EVI 21: Ray Tracing y Path Tracing

Miguel Angel Astor Romero

17 de octubre de 2016

Outline

Objetivos

Introducir y practicar los fundamentos de generación de imágenes sintéticas con los algoritmos de Ray Tracing y Path Tracing.

Objetivos Específicos

- Presentar el algoritmo de Ray Tracing de Whitted.
- Practicar la implementación de técnicas de intersección rayo-objeto.
- Presentar los fundamentos de iluminación y sombreado con funciones BRDF.
- Presentar el algoritmo de Path Tracing.
- Practicar la implementación de un método de muestreo aleatorio de Monte Carlo.

La ecuación de despliegue

$$L_o(x, \omega_o, \lambda, t) = L_e(x, \omega_o, \lambda, t) + \int_\Omega f_r(x, \omega_i, \omega_o, \lambda, t) L_i(x, \omega_i, \lambda, t)(\omega_i \cdot n) d\omega_i$$

Algoritmos de generación de imágenes sintéticas

Despliegue de polígonos

- Scan line rendering.
- Flood-fill por polígonos.
- Algoritmo del pintor.

Trazado de rayos

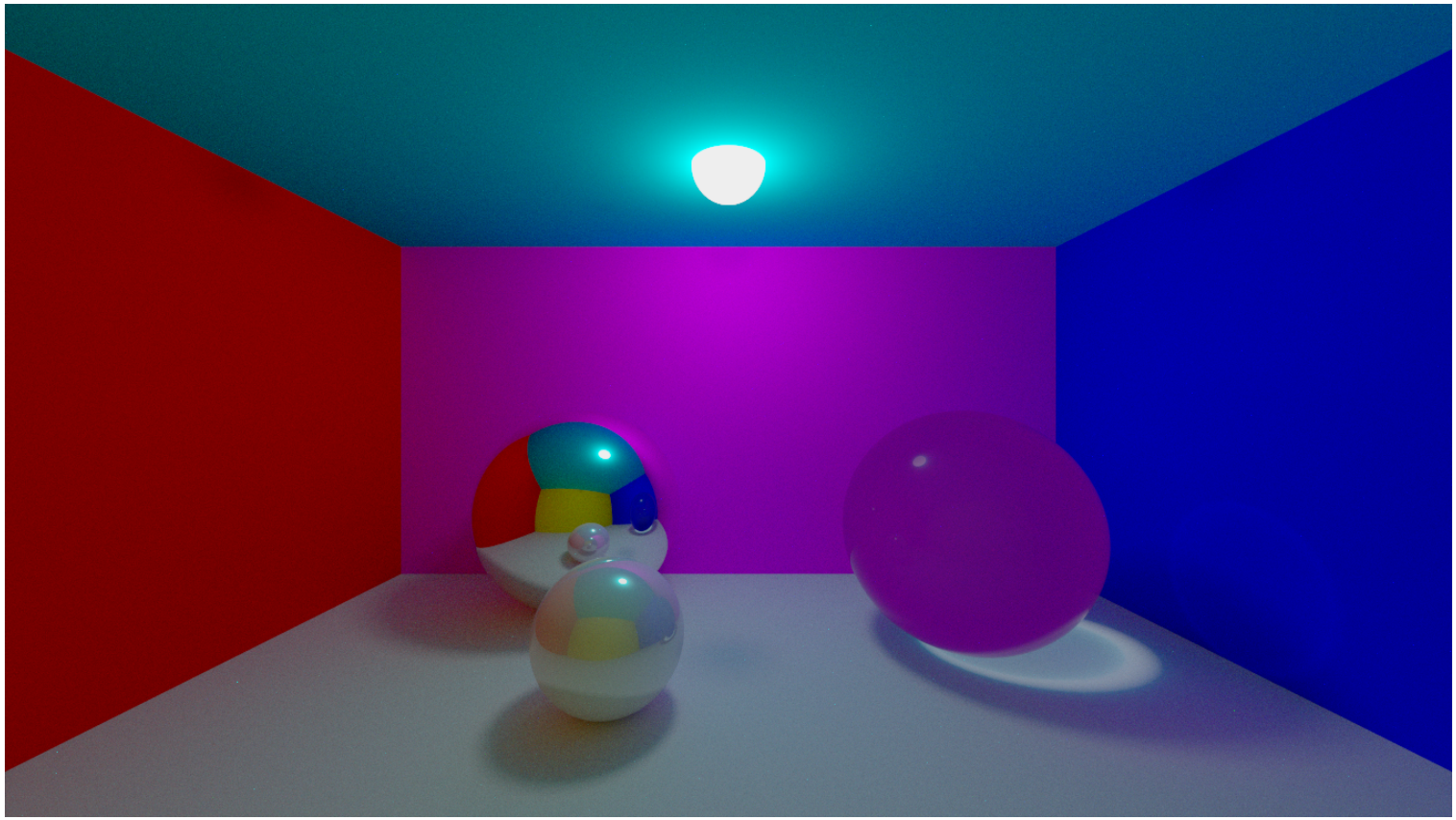
- Ray-tracing de whitted.
- Path-tracing.
- Photon Mapping.
- Metropolis Light Transport.
- Ray-marching.

Ejemplos

Despliegue de polígonos (Mirror's Edge)

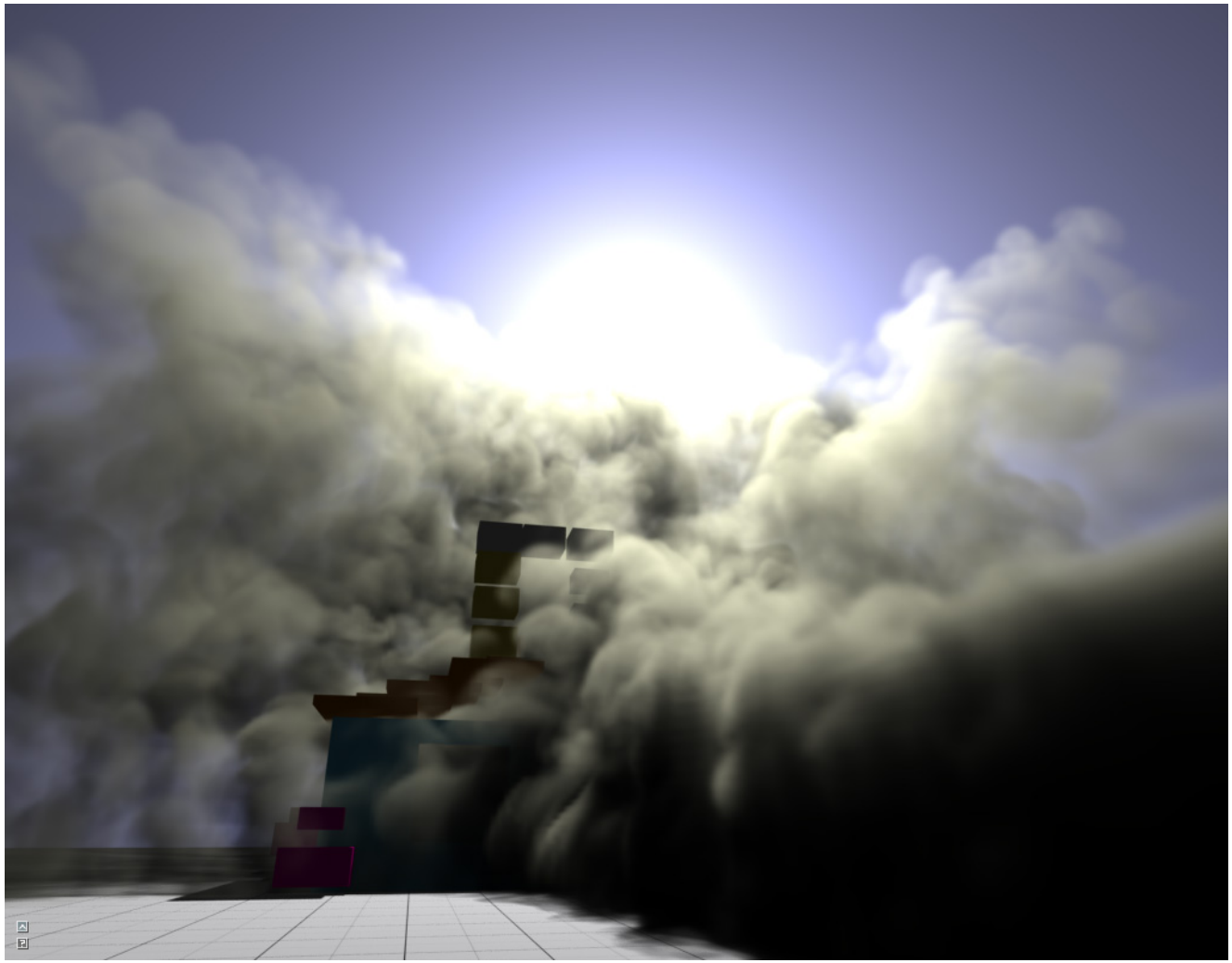
Ejemplos

Path-tracing



Ejemplos

Ray-marching

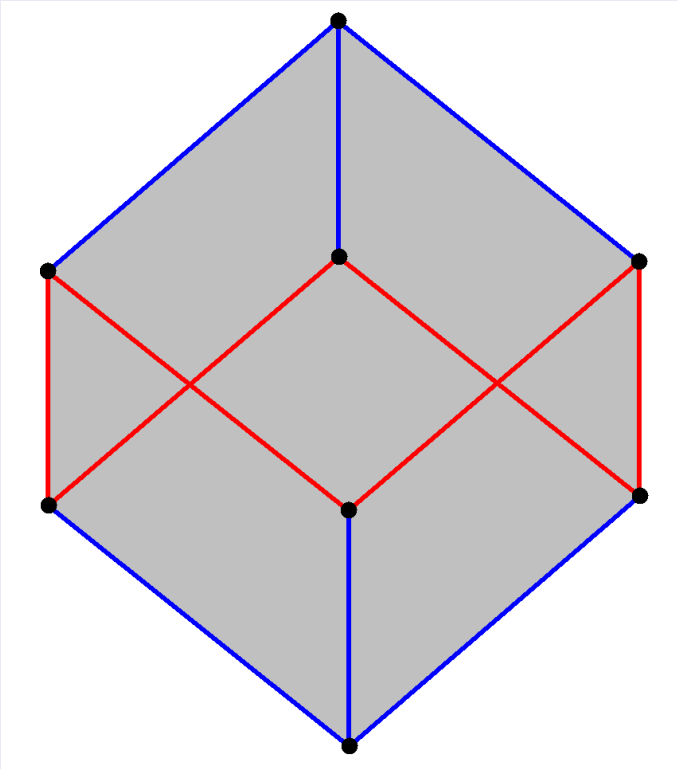


Fundamentos de generación de imágenes sintéticas

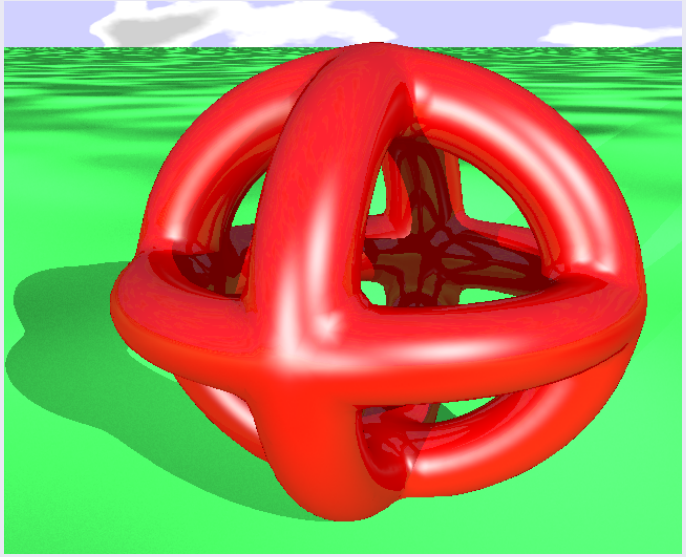
1. Vertices, polígonos y superficies implícitas o paramétricas.
2. Espacios de coordenadas.
3. Transformación entre espacios de coordenadas.
4. Proyección otrográfica y perspectiva.

Vertices, polígonos y superficies implícitas o paramétricas.

Vertices y polígonos



Superficies paramétricas

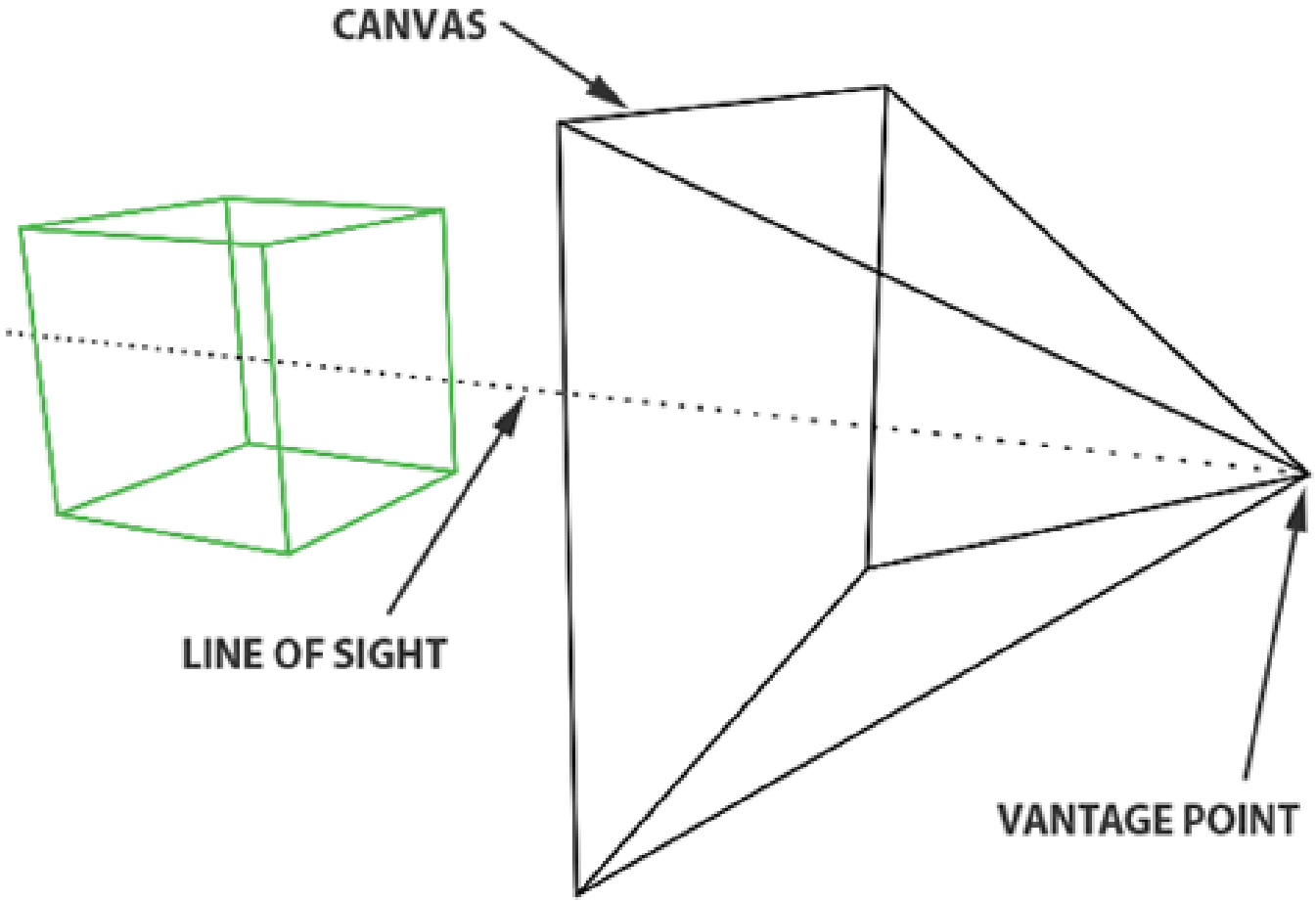


Espacios de coordenadas

Se definen 4 espacios de coordenadas:

- Coordenadas de objeto.
- Coordenadas de mundo.
- Coordenadas de ojo/vista.
- Coordenadas de imagen.

Transformación entre espacios de coordenadas



Transformación entre espacios de coordenadas

- Matriz de modelado pasa de coordenadas de objeto a mundo.
- Matriz de vista pasa de coordenadas de mundo a ojo.
- Matriz de proyección pasa de coordenadas de ojo a imagen.

IMPORTANTE: Las transformaciones en 3D se aplican con coordenadas homogéneas

- Agregar un componente adicional en 1 a los puntos, ó en 0 a los vectores.

$$\vec{p} = (x, y, z, 1)$$

$$\vec{v} = (x, y, z, 0)$$

Transformaciones afines

Traslación

$$T = \begin{pmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotación

$$R_y = \begin{pmatrix} cos(\theta) & 0 & sen(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -sen(\theta) & 0 & cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Escalamiento

$$S = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Shear

$$S_{xy} = \begin{pmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Proyección ortográfica y perspectiva

orthographic projection

perspective projection

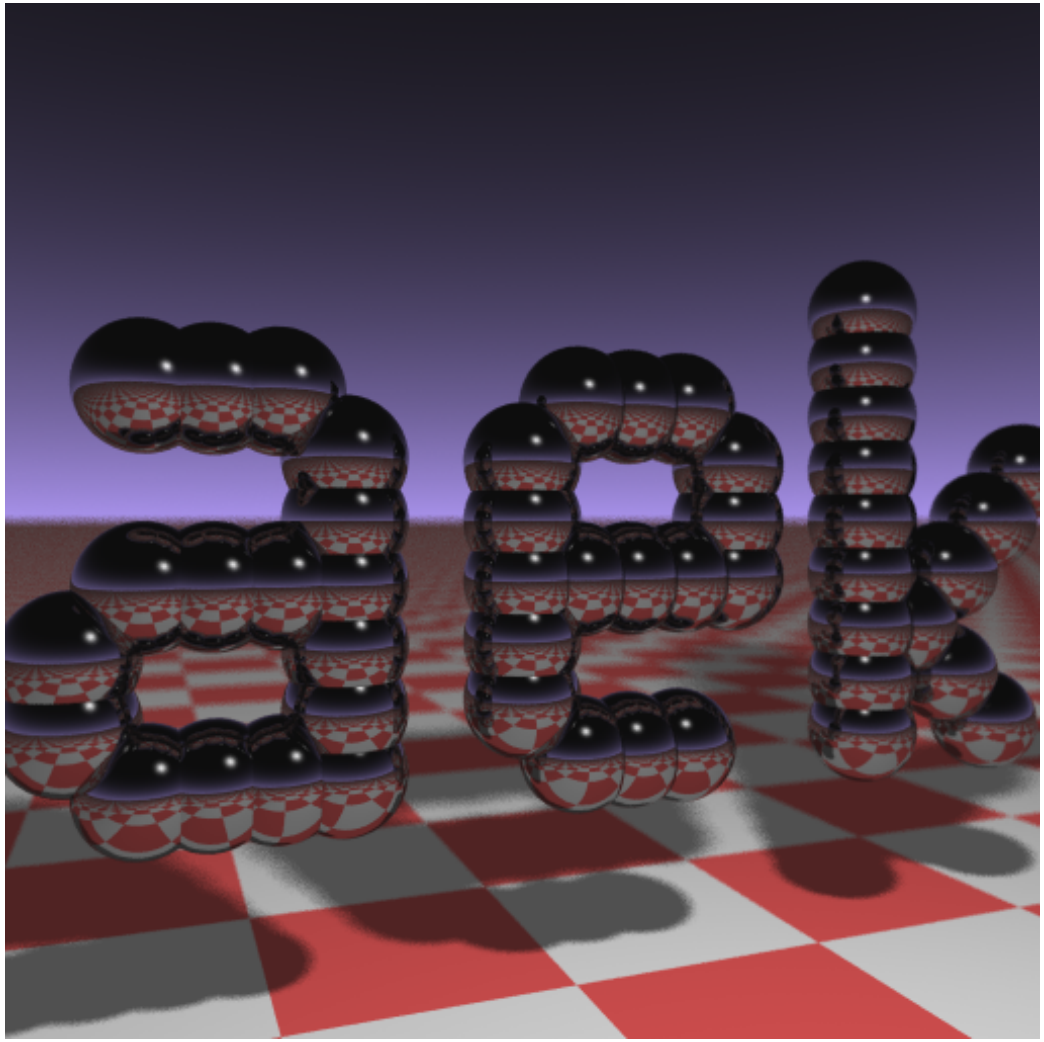
Historia

- Arthur Appel introduce el algoritmo de Ray-casting en 1968.
  - Permitía despliegue de figuras paramétricas con iluminación y sombreado local.
- Turner Whitted extiende el algoritmo en 1979 y lo llama Ray-tracing recursivo.
  - Permite calcular sombreado globalmente e interacciones especulares.
- James Kajiya presenta la ecuación de despliegue e introduce el algoritmo de Path-tracing como una solución numérica a la misma.
- Mucho desarrollo posterior:
  - Path tracing bidireccional de Lafortune en 1991.
  - Metropolis Light Transport de Veach y Guibas en 1997.
  - Photon Mapping de Jensen en 1998.

Características de Ray-tracing

- Es un algoritmo muy "sencillo" de comprender e implementar.
- Facil de extender para producir distintos efectos visuales.
- Puede desplegar superficies paramétricas o implícitas facilmente.
- Puede representar efectos especulares realistas (reflexión y refracción).

Business Card Ray-tracer



Definición de rayo

- Un rayo es un segmento de linea orientado con las siguientes propiedades:

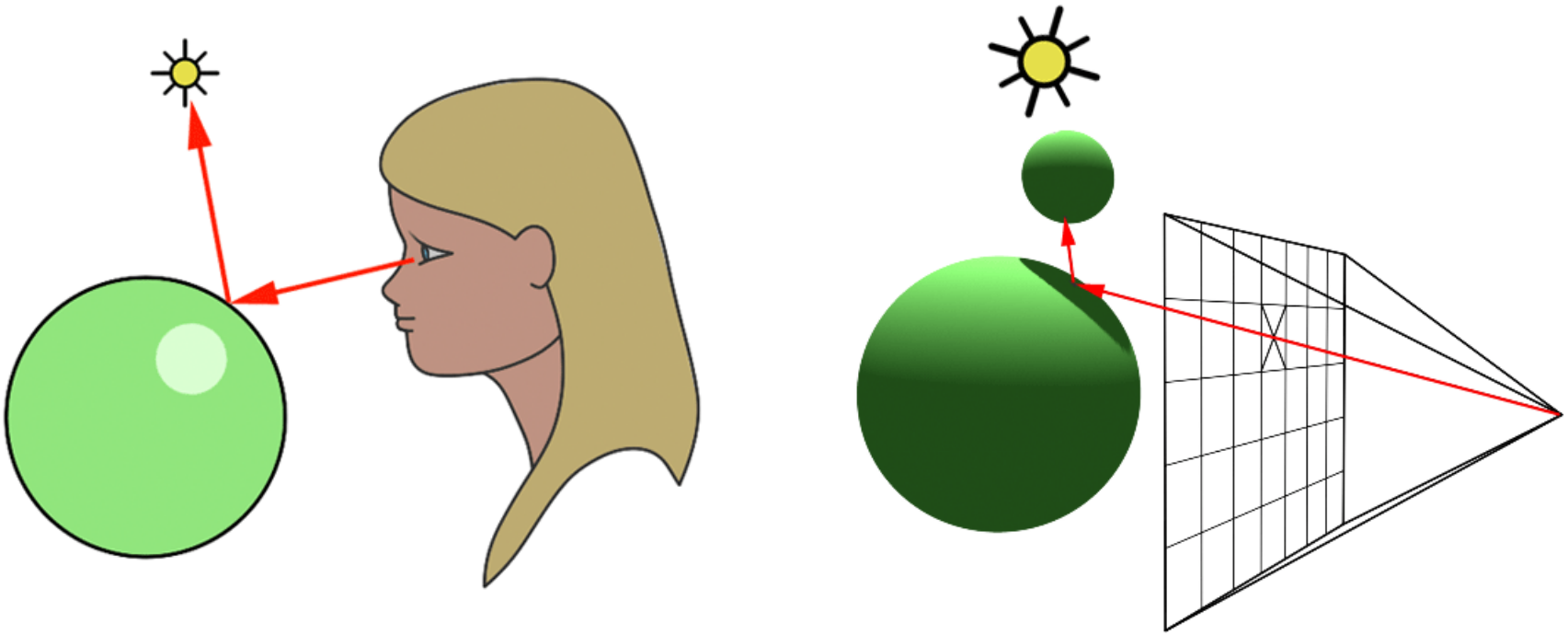
$$r(t) = o + \vec{d}t$$

- o Origen del rayo.
- $\vec{d}$ Dirección.
- t Desplazamiento.

El algoritmo de Ray-tracing de Whitted

```
Por cada fila de pixeles
  Por cada pixel de la fila
    Generar rayo primario.
    Buscar intersección más cercana.
    Si superficie es difusa
       Por cada fuente de luz
          Generar rayo de sombra.
          Calcular modelo de iluminación.
    Sino
       Calcular rayo reflejado.
       Calcular rayo refractado.
    Color final = iluminación + reflexión + refracción.
```

Que hace exactamente el algoritmo



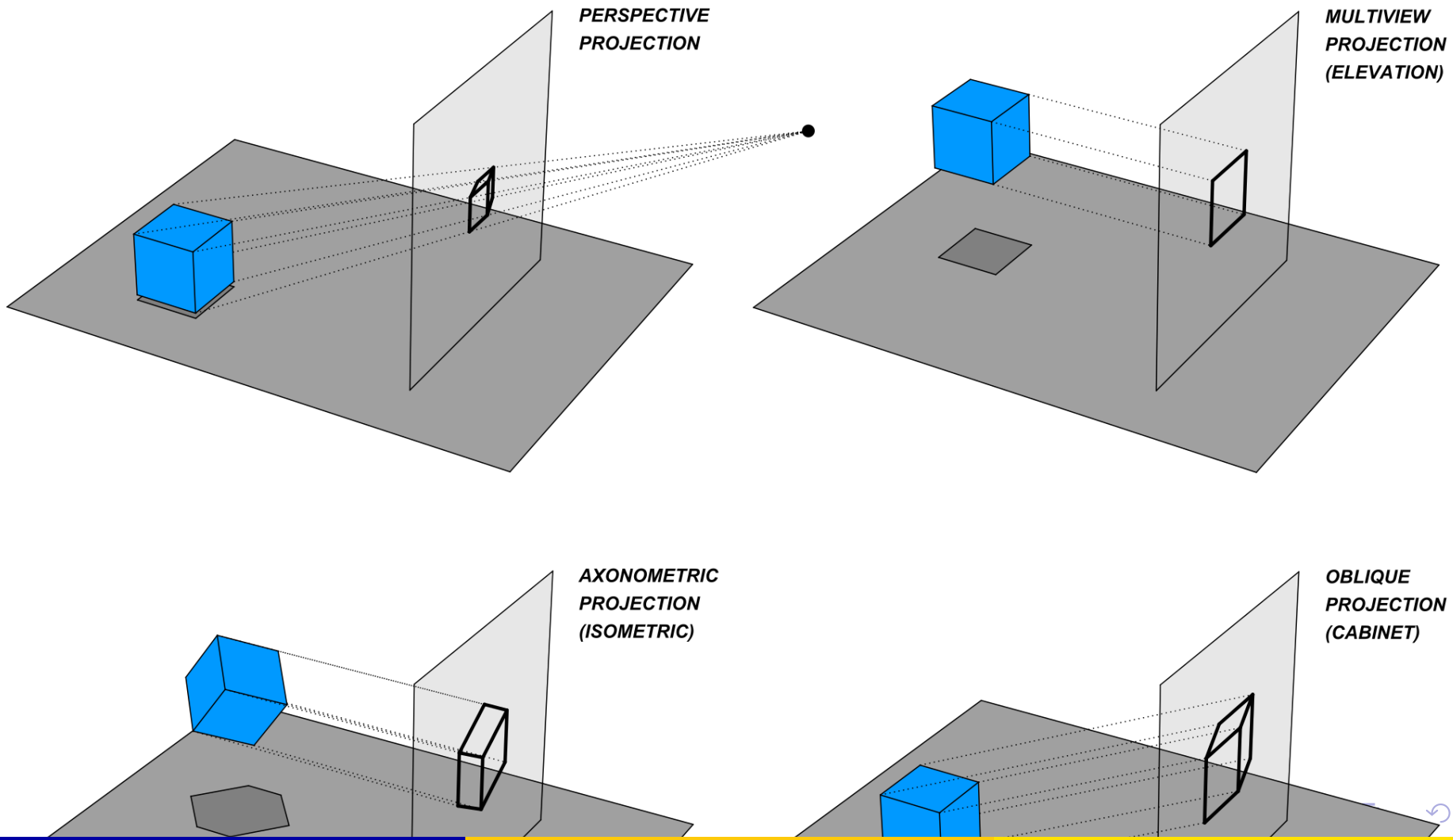
Generación de los rayos primarios

Para poder generar los rayos primarios es necesario primero definir unas cuantas cosas:

- El tipo de proyección:
  - Ortográfica o perspectiva.
- Un punto de visión.
  - De forma general, hay que establecer una "cámara".
- Un plano de proyección.

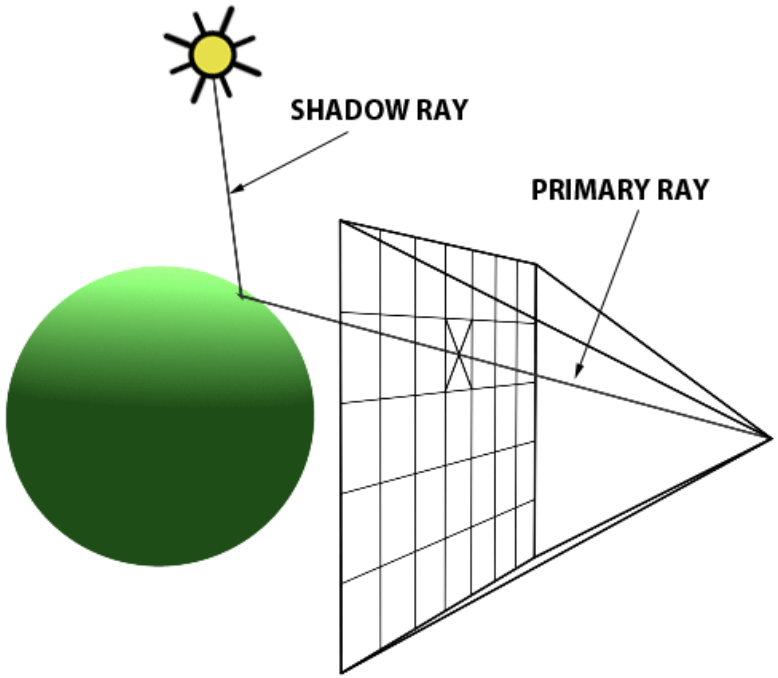
Proyección ortográfica.

En este caso solo es necesario definir el plano de proyección.



Proyección en perspectiva

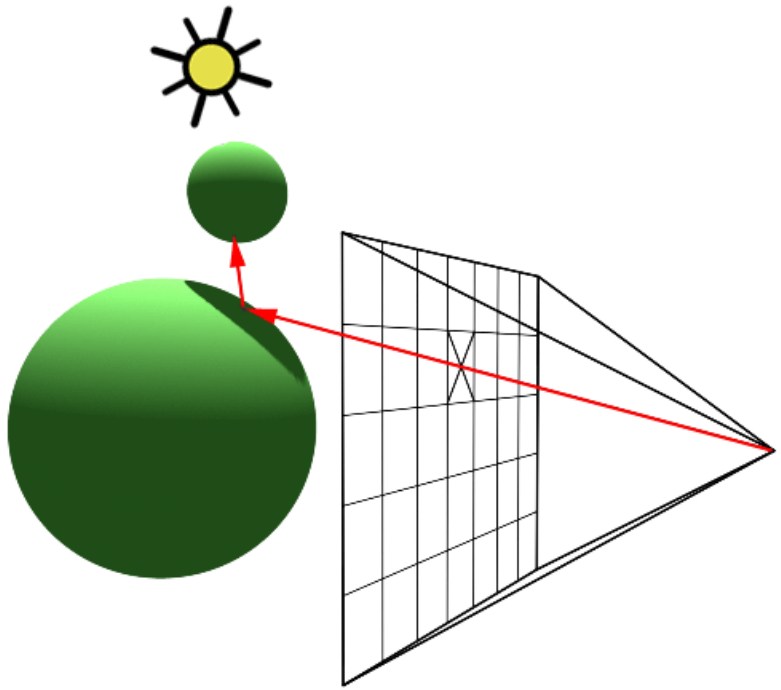
Definimos un punto arbitrario como punto de visión. Este será el origen de todos los rayos primarios.



El plano de proyección se asume que posee normal igual a $(0, 0, \pm 1)$ y que el punto $(0, 0, 0)$ está sobre el plano.

Cálculo de la dirección del rayo

En proyección perspectiva, por cada pixel se calcula el vector que va desde el origen del rayo hasta el centro del pixel.
Es posible muestrear más de un rayo por pixel (usando coordenadas sub-pixel) para reducir el *aliasing* de la imagen final.



Muestreo de un pixel (con supersampling)

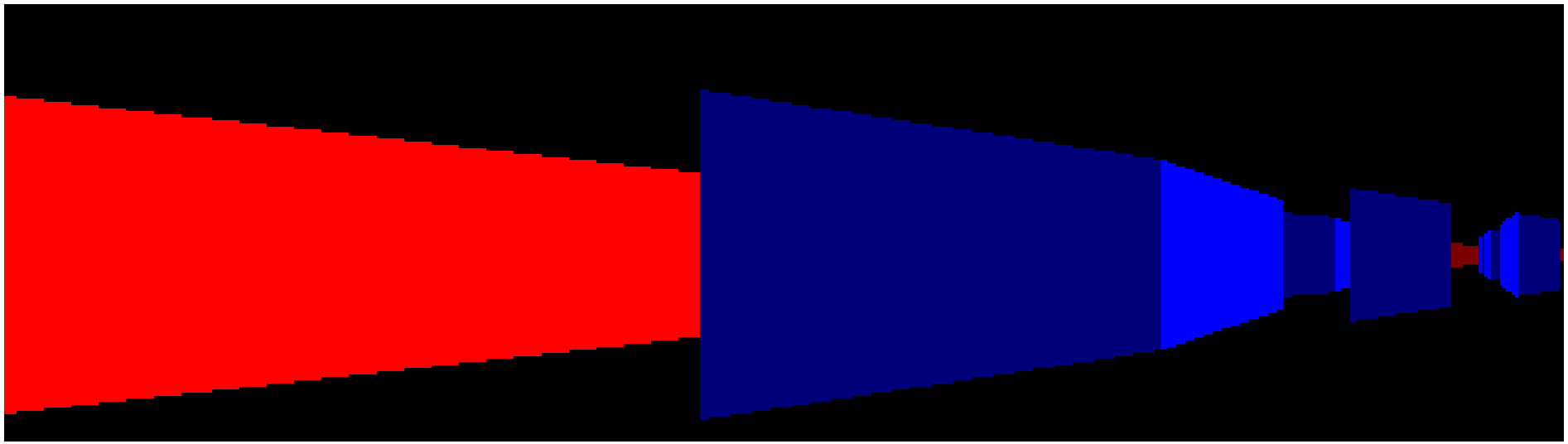
```
vec2 sample_pixel(int i, int j, float w,
  float h, float a_ratio, float fov) {

  float pxNDC, pyNDC, pxS, pyS;
  pyNDC = (static_cast<float>(i) + random01()) / h;
  pyS = (1.0f - (2.0f * pyNDC)) *
    glm::tan(radians(fov / 2.0f));
  pxNDC = (static_cast<float>(j) + random01()) / w;
  pxS = (2.0f * pxNDC) - 1.0f;
  pxS *= a_ratio * glm::tan(radians(fov / 2.0f));

  return vec2(pxS, pyS);
}
```

Consideraciones

El origen de los rayos debe estar "centrado" con respecto al plano de imagen. De lo contrario la imagen final saldrá distorsionada.



Métodos para calcular la intersección

- Las intersecciones se pueden calcular de manera algebraica o geométrica.
- En cualquier caso, lo que se desea obtener es el parámetro t a utilizar en la ecuación del rayo.
- El rayo puede intersectar la superficie en las siguientes cantidades de puntos:
  - Uno (pe: intersección rayo-plano perpendiculares)
  - Dos (pe: intersección rayo-esfera no tangencial)
  - Ninguno (pe: intersección rayo-plano paralelos)
  - Infinitos (pe: intersección rayo-plano paralelos y sobrepuestos)

Intersección rayo-plano (método algebraico)

Un plano se puede definir de forma implícita como un vector normal $\vec{n}$ y un punto p ubicado sobre el plano:

$$(p - p_0) \cdot \vec{n} = 0$$

Un rayo se define paramétricamente como:

$$o + \vec{d}t = p$$

Sustituimos p en la ecuación del plano:

$$(o + \vec{d}t - p_0) = 0$$

Y luego despejamos t:

$$t = \frac{(p_0 - o) \cdot \vec{n}}{\vec{d} \cdot \vec{n}}$$

Intersección rayo-esfera (método algebraico)

Una esfera se define como:

$$x^2 + y^2 + z^2 - r = 0$$

Considerando que x, y y z son las coordenadas de un punto p, entonces podemos reescribir como:

$$p^2 - r^2 = 0$$

Luego sustituimos P con la ecuación del rayo

$$(o + \vec{d}t)^2 - r^2 = o^2 + (\vec{d}t)^2 + 2o\vec{d}t - r^2 = 0$$

Que es una fórmula cuadrática:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

La función bidireccional de distribución de reflectancia (BRDF)

- Es una función que describe como se refleja la luz en una superficie.
- Se utiliza para aproximar una función más general llamada Función Bidireccional de Distribución de Dispersión y Reflectancia Superficial (BSSRDF).
- Se divide en dos componentes:
  1. Especular.
  2. Difuso.

La BRDF de Phong

Definida por Bui Tuong Phong, es una BRDF básica que puede usarse para aproximar el aspecto visual de materiales satinados como el plástico.

- No es físicamente correcta (quebranta la ley de conservación de la energía).
- Asume que las superficies son reflectores lambertianos perfectos.

Componente difuso

$$d = max(\vec{n} \cdot \vec{l}, 0)$$

$$color = material * luz * d$$

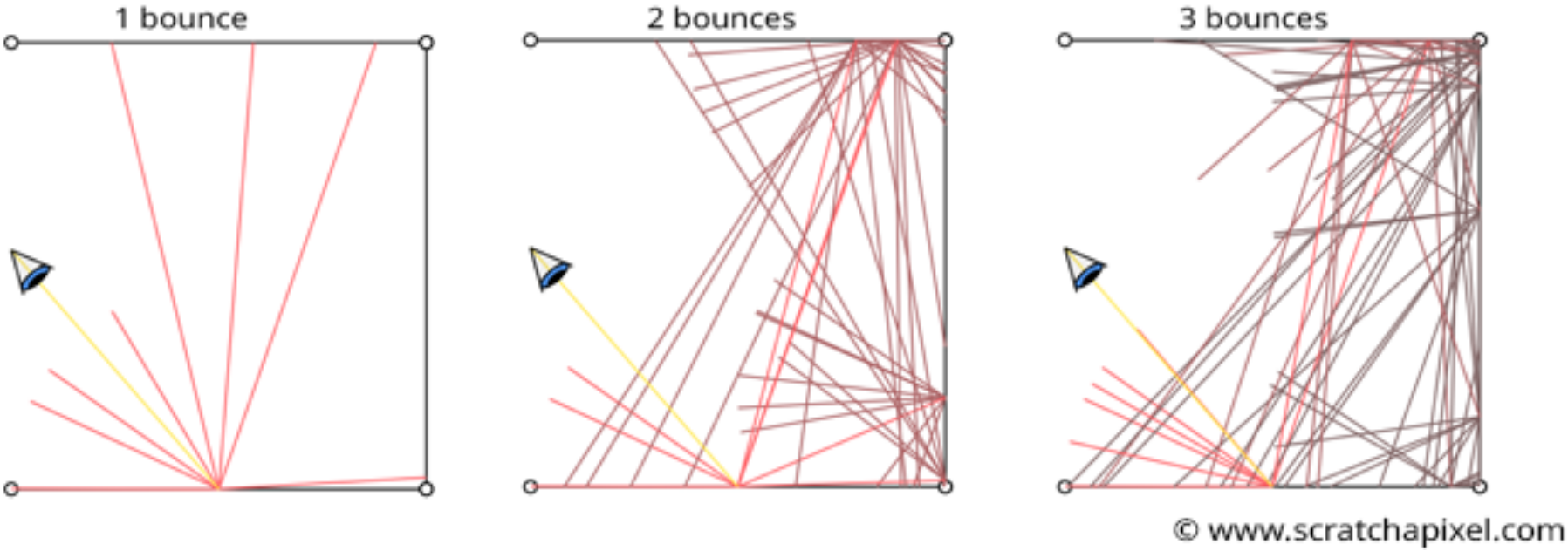
Componente especular

$$\vec{r} = reflejar(\vec{l}, \vec{n})$$

$$s = max(\vec{r} \cdot \vec{i}, 0)^b$$

$$color = material * luz * s$$

Una imagen



La estrctura de representación de materiales

Para calcular la iluminación y el sombreado se definen por lo menos las siguientes propiedades para los materiales de las superficies:

Color difuso Color "principal" del objeto.

Color especular Color de los brillos especulares (blanco en materiales no metálicos).

Color emisivo Color de la luz que emite el objeto (negro salvo en fuentes de luz).

Brillo Intensidad de los brillos especulares.

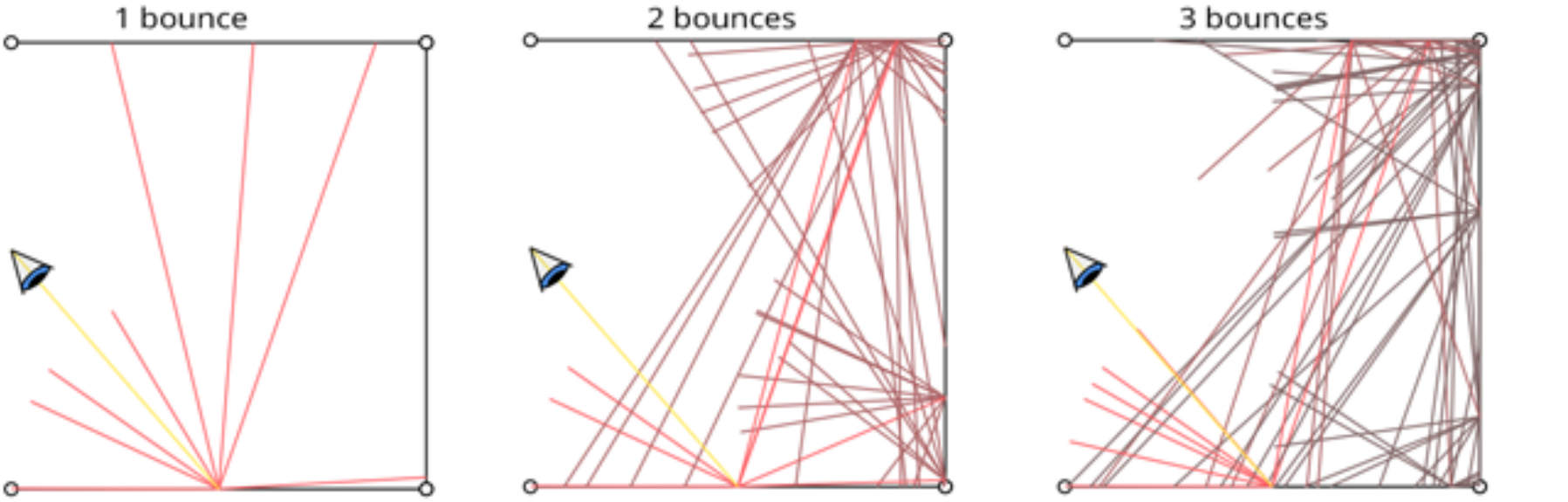
Reflectividad Cantidad de luz que refleja el material especularmente.

Indice de refracción 1 en el caso del aire.

¿Que es Path-tracing?

Es una extensión del algoritmo de ray-tracing diseñada por James Kajiya para resolver numéricamente la ecuación de *rendering*.

- Mientras que Ray-tracing calcula rayos recursivamente solo al intersectar superficies especulares, Path-tracing calcula rebotes de los rayos luminosos al chocar sobre superficies difusas.
- Esto se realiza mediante técnicas de muestreo aleatorio.



Muestreo de Monte Carlo

Dada una integral de esta forma:

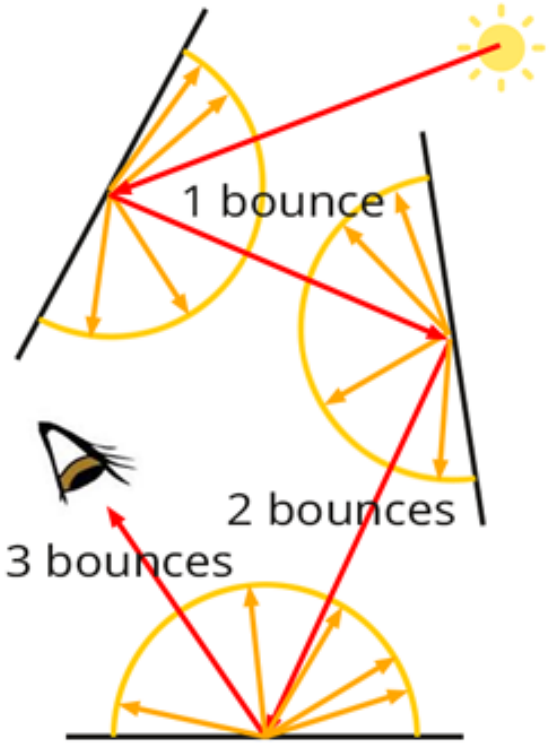
$$F = \int_a^b f(x)dx$$

Esta puede aproximarse con el siguiente estimador, usando N muestras:

$$F = \frac{1}{N} \sum_{i=0}^{N-1} \frac{f(X_i)}{pdf(X_i)}$$

Extendiendo el ray-tracing con muestreo de Monte Carlo

La idea es muestrear la semiesfera ubicada sobre el punto de intersección utilizando un estimador de Monte Carlo y utilizar la dirección muestreada como dirección de un rayo reflejado de forma difusa.



© www.scratchapixel.com

Consideraciones

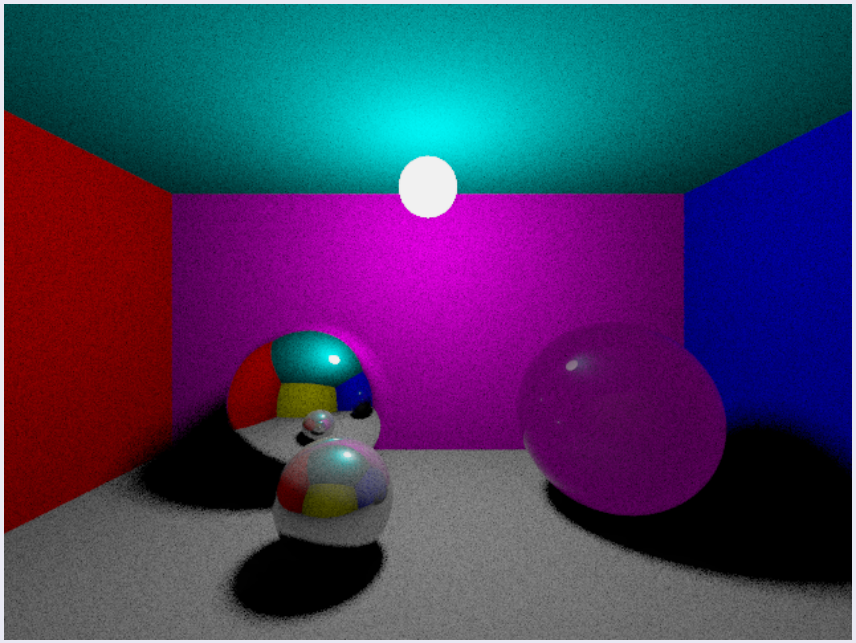
- Se calcula una sola dirección de rebote por cada intersección en el camino.
- Si las direcciones se muestrean de manera uniforme, la PDF es constante:

$$PDF = \frac{1}{2\pi}$$

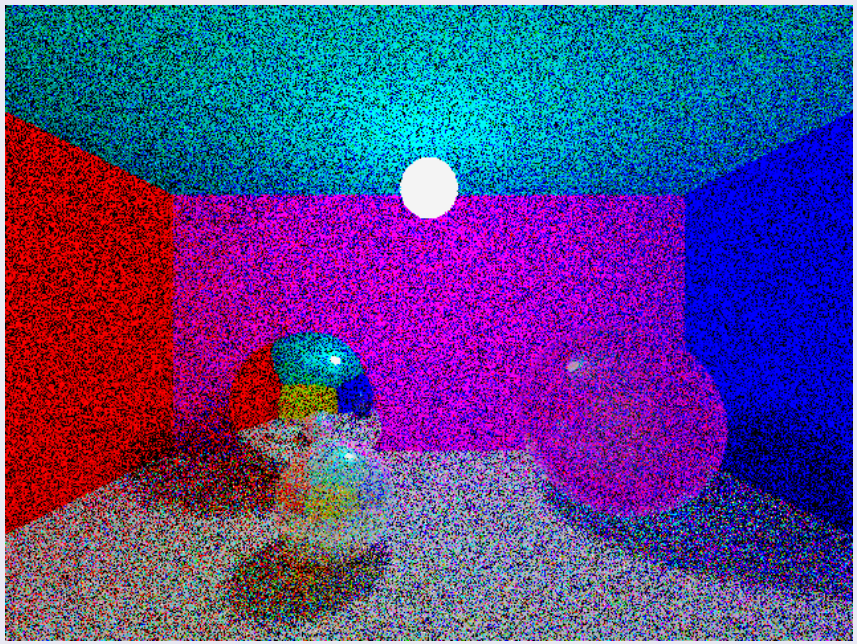
- Las direcciones se muestrean en una semiesfera ideal y luego se convierten a coordenadas de mundo.

Comparación

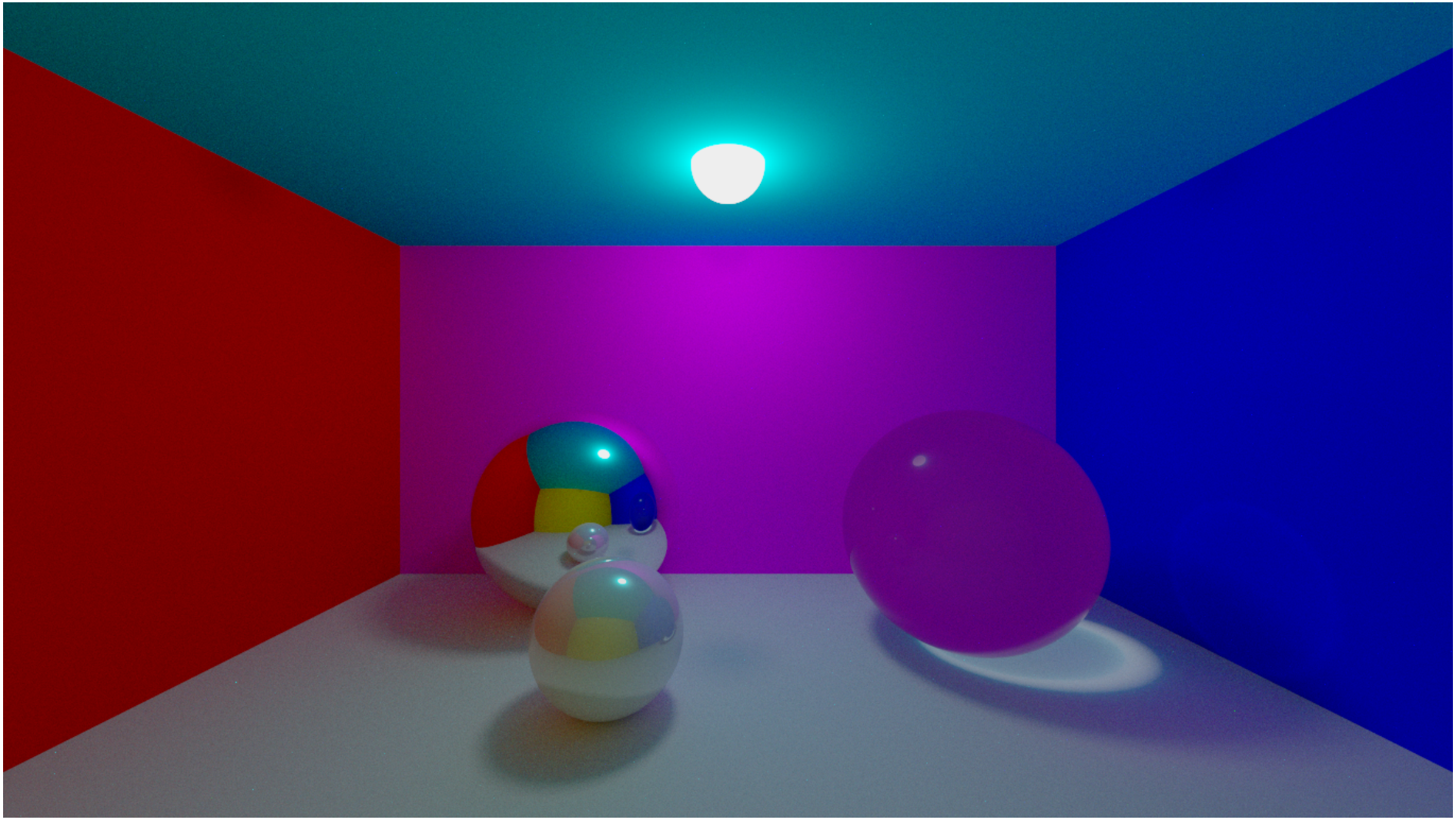
Ray-tracing



Path-tracing

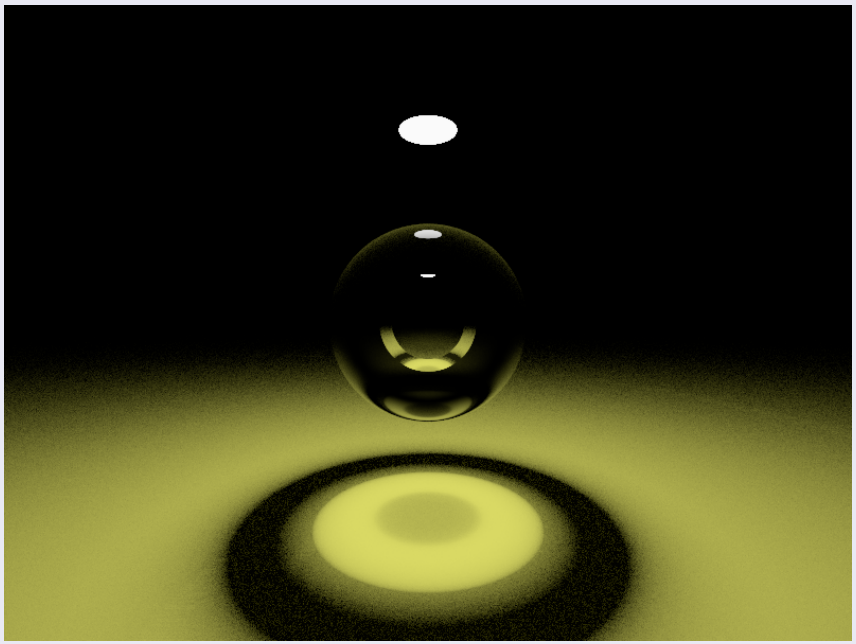


Una imagen más

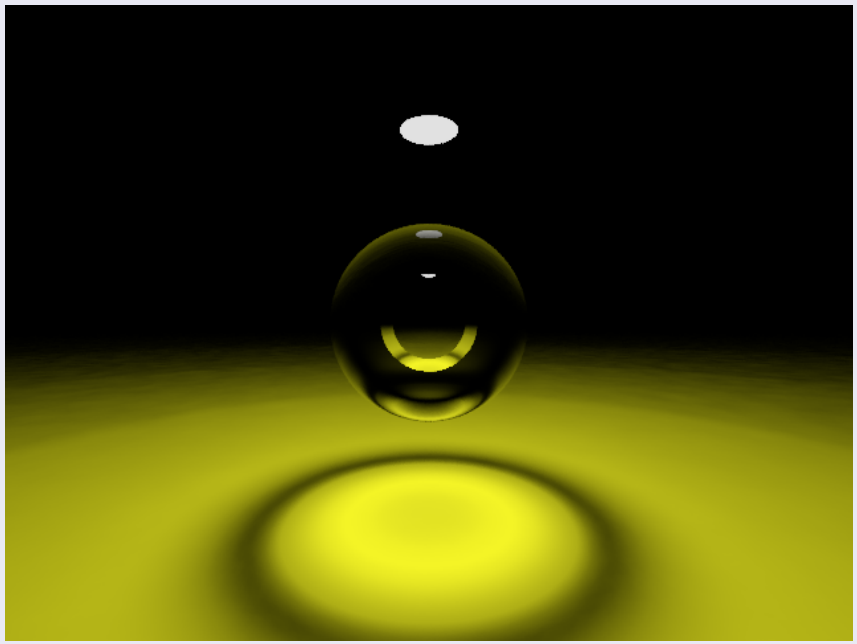


Reducción del ruido

Path-tracing



Photon Mapping



Referencias

- Foley, James D., et al., Introducción a la Graficación por Computador, Addison-Wesley, 1996.
- Jensen, Henrik Wann, Realistic image synthesis using photon mapping, AK Peters, Ltd., 2001.

Contactos

Prof. Miguel A. Astor

- miguel.astor@ciens.ucv.ve
- miguel.a.astor@ucv.ve

¿Preguntas?