Iluminación Clásica en Tiempo Real

Grupo 7
Angel Santiago Avendaño Cañon
John Alejandro Pastor Sandoval
Nelson Ivan Castellanos Betancourt
Samuel Josué Vargas Castro

Agenda

1) Introducción a la iluminación clásica (John)
2) Pipeline Gráfico (Jhon)
3) Generación de una imagen básica
4) Intersección de objetos (Angel)
5) Normales de una superficie (Samuel)

6) Shading básico (ivan)

7) Lambert (Angel)

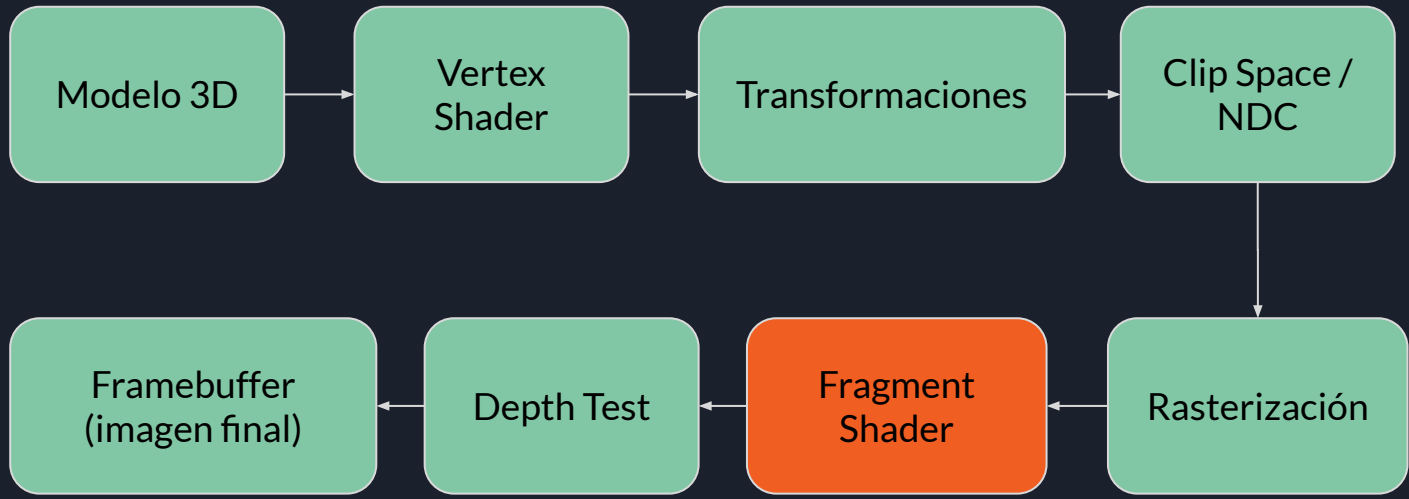
8) Phong (Samuel)

9) Blinn-Phong (Samuel)

10) Tipos de luz (iván)

11) Materiales (Angel)

Pipeline Gráfico



Introducción a la iluminación clásica

La iluminación es el proceso que determina cómo se ve un objeto cuando la luz interactúa con su superficie.

Iluminación Clásica (no física)

La iluminación clásica utiliza modelos matemáticos simples para simular el comportamiento de la luz de forma rápida y eficiente.

- Diseñada para tiempo real
- No busca exactitud física
- Prioriza rendimiento y control artístico

Generación de una imagen básica

Antes de simular iluminación o luz, necesitamos algún sistema que sea capaz de almacenar y visualizar datos numericos como colores.

PPM: es un formato en texto plano que no necesita librerías externas lo que ayuda a concentrarse en el algoritmo de trazado

Su estructura es : P3, Ancho, Alto, Max Color, seguido de filas con valores RGB.

PPM example [edit]

This is an example of a color RGB image stored in PPM format. There is a newline character at the end of each line.

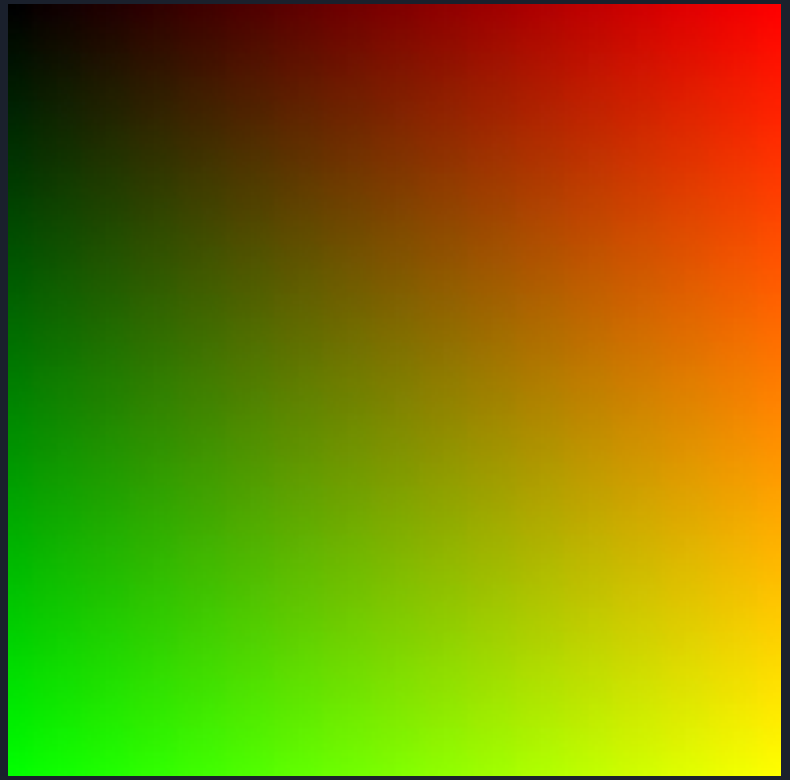
```
P3
# The P3 means colors are in ASCII, then 3 columns and 2 rows,
# then 255 for max color, then RGB triplets
3 2
255
255   0   0     0 255   0     0   0 255
255 255   0   255 255 255     0   0   0
```

Image
(magnified)

Ejemplos generación imagenes basicas

Como lo veríamos en el codigo de la imagen hay cosas a tener en cuenta:

1) Los pixels estan escritos en filas
2) Cada fila de pixeles se escriben de izquierda a derecha y de arriba hacia abajo
3) Los cálculos de luz generar valores reales continuos entre [0,1] para la imagen final estos valores los discretizamos en un espacio de 8 bits (0-255) antes de visualizarlos



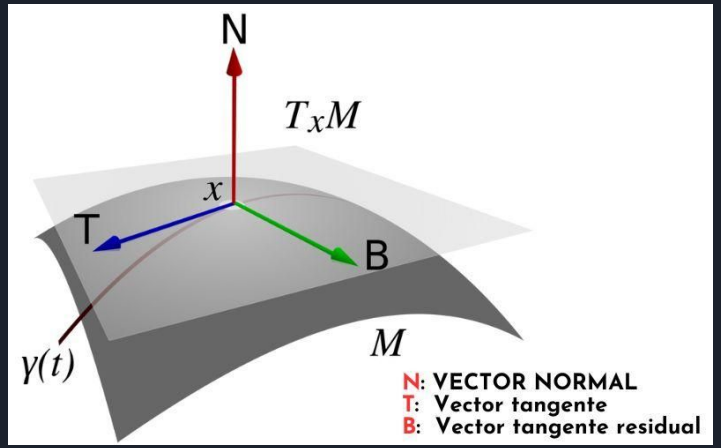
Vectores normales

Un vector normal es un vector perpendicular a una superficie en un punto específico.

- Se representa normalmente como N
- Tiene magnitud 1 (vector normalizado)
- Define la orientación de la superficie

En iluminación clásica se utilizan para:

- Calcular la iluminación difusa (Lambert)
- Determinar sombras suaves o duras



Tomado de:
https://www.lifeder.com/vector-normal/

Vectores normales

Según el objeto, el vector normal se calcula de manera distinta.

Esfera

$$N = \frac{P - C}{||P - C||}$$

Donde:
N: Vector normal
P: Punto de intersección
C: Centro de la esfera

Plano

$$N = (a, b, c)$$

Si el plano tiene la forma:
ax+by+cz+d=0.

Intersección de Objetos

Objetos Primitivos

- Esfera
- Cubo/Caja (AABB / OBB)
- Plano
- Triángulo

Son formas geométricas básicas que se pueden describir matemáticamente, fáciles de calcular intersecciones, todas las escenas se reducen a primitivos para poder calcular visibilidad e iluminación.

Shading: modelo de iluminación clásico

En el pipeline gráfico moderno, el shading ocurre en el fragment shader. Su objetivo es calcular el color final de cada fragmento a partir de:

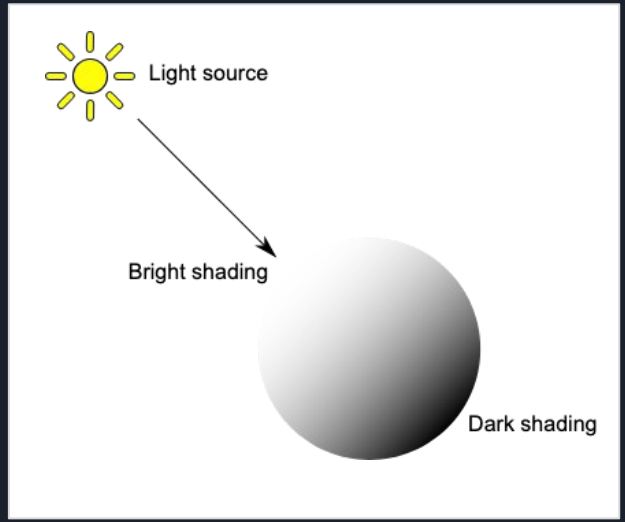
- Normales interpoladas
- Posición del fragmento
- Dirección de la luz
- Dirección de la cámara
- Propiedades del material

Una intersección ocurre cuando un rayo toca la superficie de un objeto.

Este se define como :

- r(t) = O + td
- o: Origen del rayo
- d: Dirección (Normalizada)
- t: Distancia a lo largo del rayo

Y se busca un t > 0 donde el rayo intersecta al objeto



Lambert

El modelo de lambert describe como la luz se refleja de forma difusa en una superficie mate

- La luz se dispersa en todas las direcciones
- El brillo no depende del observador
- Solo depende del ángulo entre la luz y la superficie

Se basa en un concepto simple, entre más directa llega la luz a la superficie, más iluminada se ve

- Luz perpendicular - Máxima iluminación
- Luz rasante - Iluminación baja
- Luz detrás - Sin iluminación

Para trabajar se usa el vector normal de la superficie y la dirección de la luz

se usa la fórmula para la iluminación

$$I_d = k_d(N \cdot L)$$

Funciona gracias a la propiedad del producto punto

A dot B = |A|*|B|*cos(theta)

https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEhzkMVK7NPpcGF0D226HqcXY20mSeb9aW-DnPJQzpSaFY9EGkOthLX9nrq7MhBH8bYaV1p3NrejGR3KrtzXGsuCk3oLe5URFZQ3k6KplZo9oxt4zjL3O4eQXdL-33RjkMZXomnyQg/s400/SNielson_Light_and_Form_Basics2.jpg

Phong

El modelo Phong es un modelo empírico de iluminación que simula cómo la luz interactúa con una superficie combinando tres componentes:

Luz ambiental

Simula la luz indirecta del entorno.

$$I_a = k_a I_a$$

Luz difusa (Lambert)

Depende del ángulo entre la normal N y la luz L:

$$I_d = k_d (N \cdot L)$$

Luz Especular (Brillo)

Simula los reflejos brillantes:

$$I_s = k_s (R \cdot V)^n$$

R = vector reflejado

V = dirección hacia la cámara

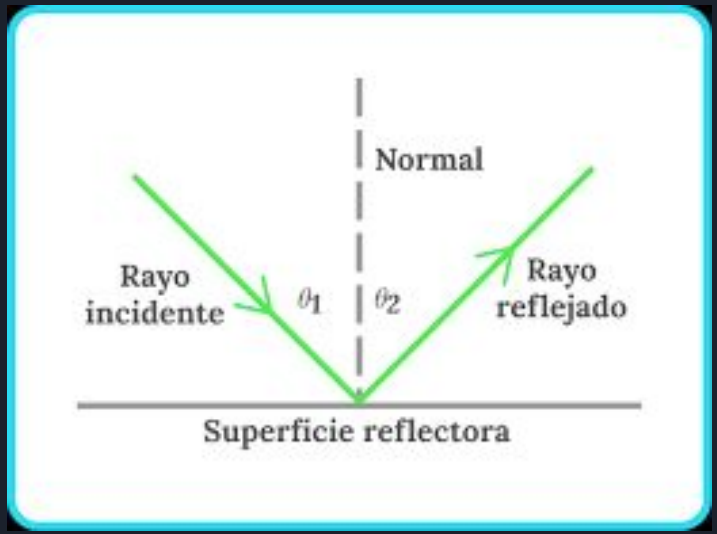
n = coeficiente de brillo (shininess)

Vector reflejado

El vector reflejado (R) es la dirección en la que la luz rebota después de golpear una superficie. Se calcula a partir de la dirección de la luz y la normal de la superficie, siguiendo la ley de reflexión: el ángulo de incidencia es igual al ángulo de reflexión.

$$R = 2(N \cdot L)N - L$$

Donde:
L es la dirección hacia la luz normalizada
N es la normal
R es el vector reflejado



Tomado de:
https://multimedia.uned.ac.cr/pem/fisica_ciencias_agronomicas/pag/mod_4.html#prettyPhoto/0/

Blinn-Phong

El modelo Blinn-Phong es una mejora computacional del modelo Phong que optimiza el cálculo del brillo especular. Fue propuesto por James F. Blinn. La diferencia principal está en cómo se calcula el reflejo especular.

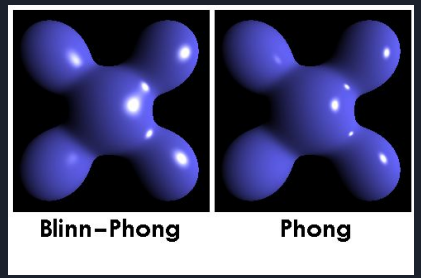
En lugar de usar el vector reflejado, se usa el vector medio (H):

$$H = \frac{L + V}{||L + V||}$$

Posteriormente se calcula la iluminación especular de la siguiente manera:

$$(N \cdot H)^n$$

De esta manera, se tiene la siguiente comparación entre los modelos de iluminación, donde se puede ver que el modelo Blinn-Phong es más eficiente y estable numéricamente, ya que reemplaza el cálculo del vector reflejado por el vector medio entre la luz y la cámara. Esto reduce el costo computacional y mantiene un resultado visual muy similar al modelo Phong, por lo que es ampliamente utilizado en gráficos en tiempo real.



Tomado de: https://commons.wikimedia.org/wiki/File:Blinn_phong_comparison.png

Tipos de luz

En el modelo de iluminación clásico, las luces se clasifican según su comportamiento geométrico y su dependencia espacial.

Las tres más usadas en motores en tiempo real son:

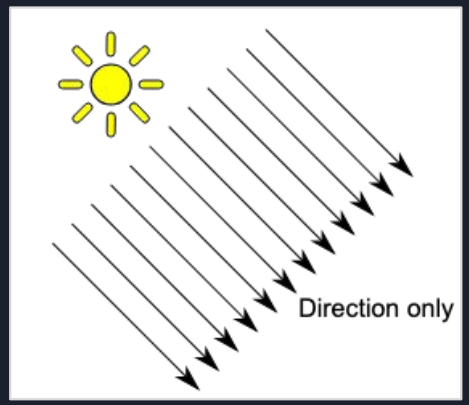
- Luz Direccional
- Luz Puntual (Point Light)
- Luz Spot
- Area Lights
- Emissive materials
- Ambient light

La diferencia fundamental entre ellas está en cómo se calcula el vector hacia la luz L y si existe atenuación por distancia. Estas luces pueden operar en tres modos:

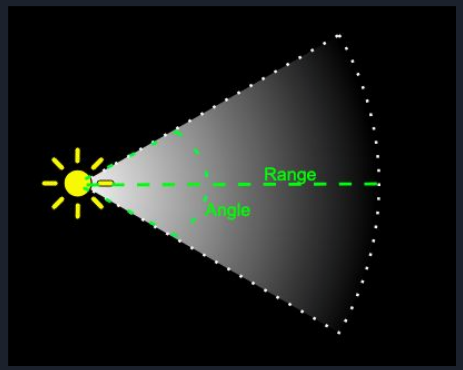
- Realtime → calculadas cada frame en GPU
- Baked → precalculadas y almacenadas en lightmaps
- Mixed → combinación de ambas

Tipos de luz

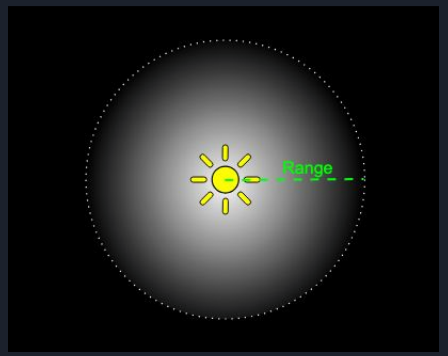
Direccional



Luz Spot



Luz Puntual



Imágenes tomadas de: https://docs.unity3d.com/es/2018.4/Manual/Lighting.html

Materiales

Un material define la forma en como una superficie interactúa con la luz.

Las propiedades de un material determina:

- Cuánta luz se refleja
- En qué dirección
- Con qué color e intensidad

Mientras que la geometría define la ubicación del objeto, un material define como se ve

En iluminación clásica, se divide en

- Difuso (Lambert)
- Especular
- Ambiental

Un material también define como la luz rebota en una superficie y hacia qué dirección.

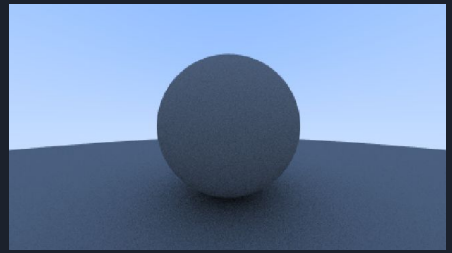
Para un rayo se tiene que:

$$R=L-2(L\cdot N)N$$

Tipos de materiales simples:

Mate:

- Difuso:
  - Con poco o nada de reflexiones (Especular)
- Plástico:
  - Difuso + Especular moderado
- Metal:
  - Especular dominante
  - El color viene del reflejo



Ejemplo de un material difuso, tomado de: https://raytracing.github.io/books/RayTracingInOneWeekend.html