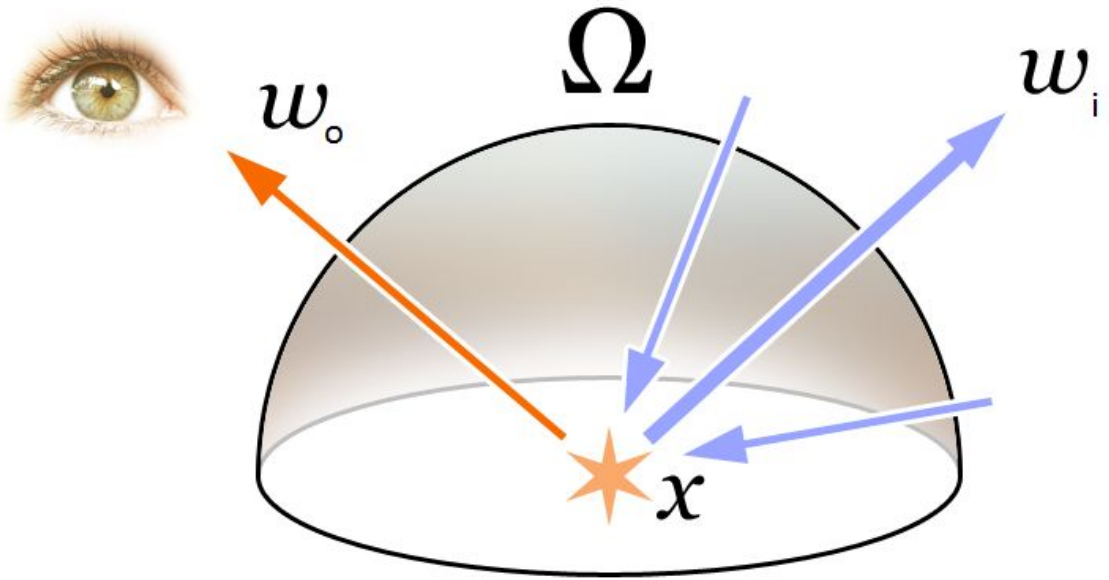
L'Empire des lumières, Magritte

CC3501 Modelos de Iluminación

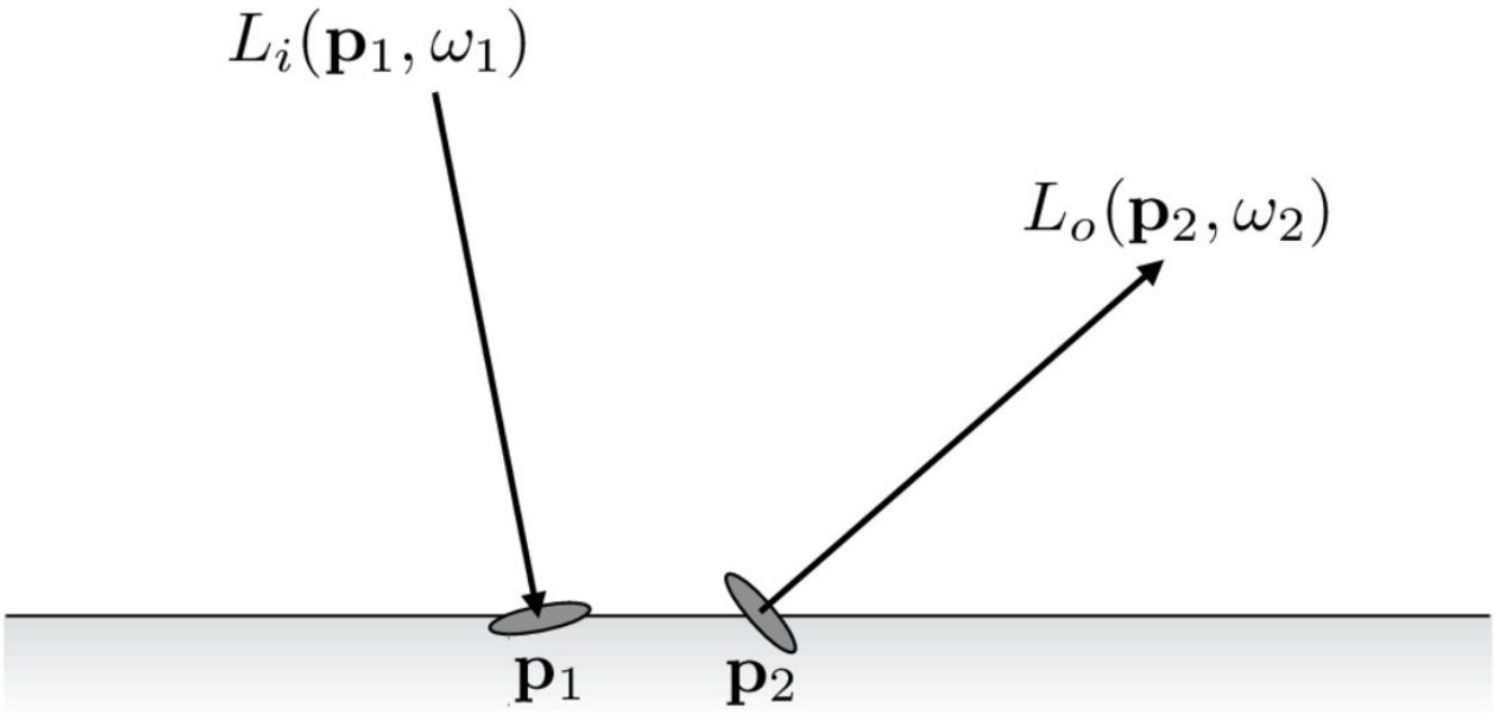
Eduardo Graells-Garrido

Otoño 2023



$$L_{\mathrm{o}}(\mathbf{x}, \omega_{\mathrm{o}}, \lambda, t) = L_{\mathrm{e}}(\mathbf{x}, \omega_{\mathrm{o}}, \lambda, t) \ + \int_{\Omega} f_{\mathrm{r}}(\mathbf{x}, \omega_{\mathrm{i}}, \omega_{\mathrm{o}}, \lambda, t) L_{\mathrm{i}}(\mathbf{x}, \omega_{\mathrm{i}}, \lambda, t)(\omega_{\mathrm{i}} \cdot \mathbf{n}) \, \mathrm{d}\, \omega_{\mathrm{i}}$$

Cuando hablamos de iluminación, se suele pensar en la ecuación de rendering. Esta ecuación describe el **resplandor** (*radiance*) emitido y reflejado por cualquier punto del espacio.



En general, el resplandor incidente no es igual al emitido.

$$L_i(\mathbf{p}, \omega) \neq L_o(\mathbf{p}, \omega)$$

INCIDENT

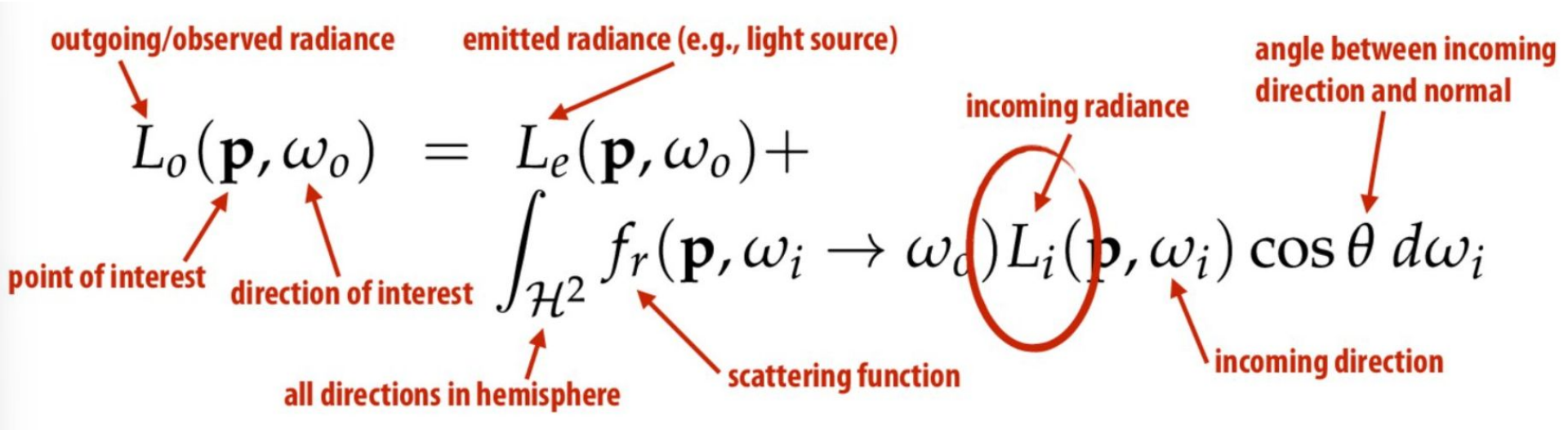
EXITANT

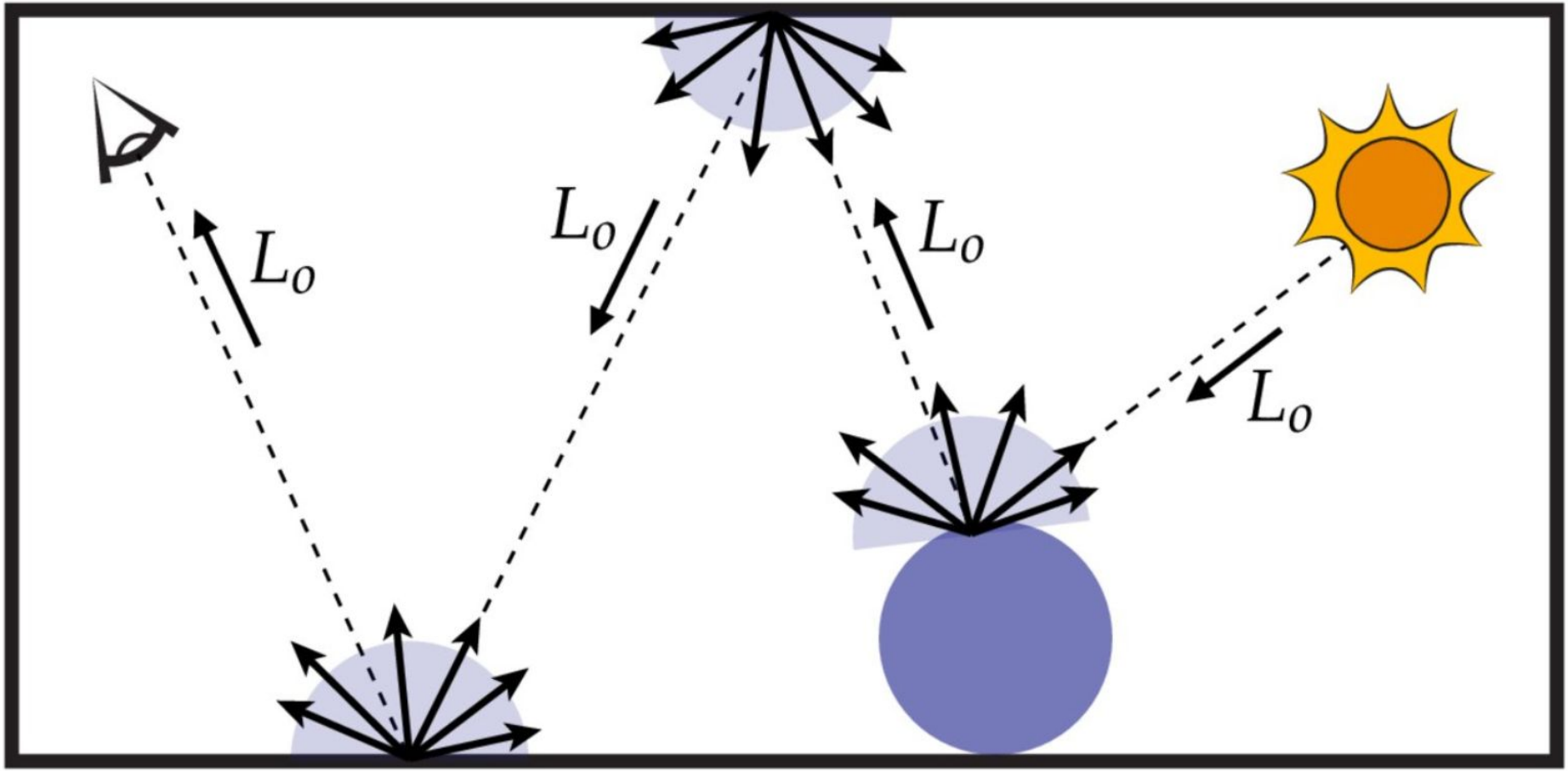
En ambos casos, la iluminación recibida y emitida por una superficie depende de muchos factores. Incluyendo **dirección** y **posición** de las **fuentes de luz**.

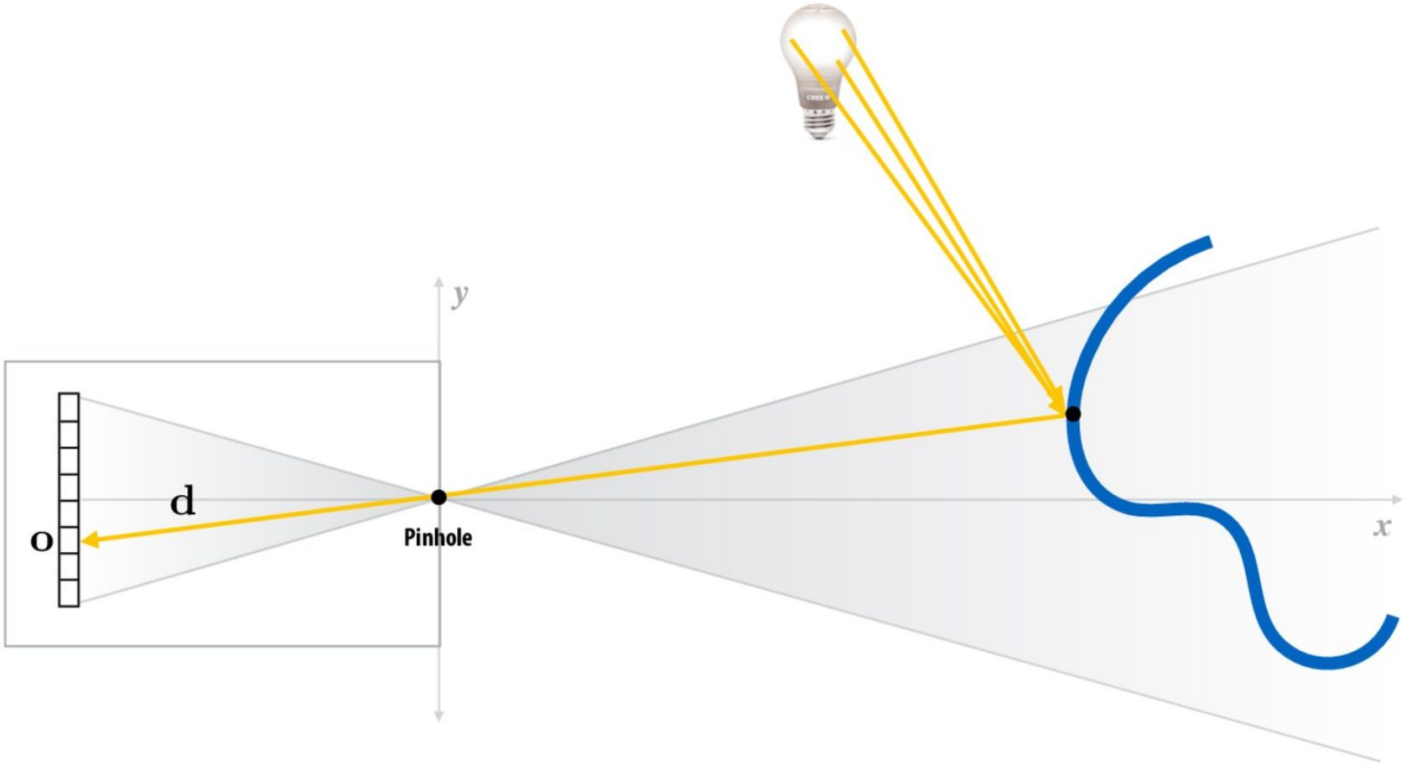
Escena renderizada con Pov-Ray, demostrando radiosidad, mapeo de fotones, blur focal, y otras capacidades fotorealistas (creado por Gilles Tran)

“Reflections”. Epic, ILMxLAB, NVIDIA

La ecuación de rendering




L_o
L_o
L_o
L_o



Más adelante veremos Ray Tracing.

Iluminación Global

Escenas anteriores involucran abundantes reflexiones y transparencias.

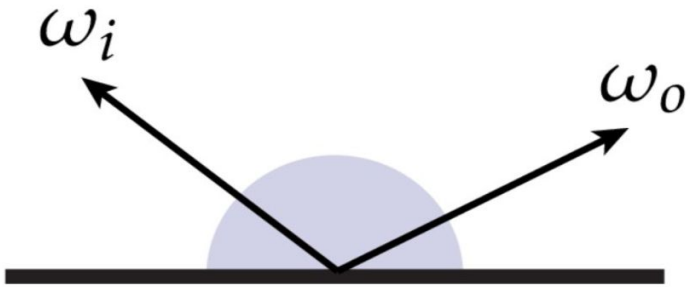
Pintar cada modelo o parte de la escena implica un conocimiento de los objetos que se ubican a su alrededor.

Las imágenes anteriores son sintéticas, y utilizan algoritmos de iluminación global para lograr un excelente nivel de realismo.

Iluminación Local

En esta clase veremos un algoritmo de iluminación local. No consideramos elementos del entorno, sólo el elemento siendo iluminado.

Sin embargo, sí podemos estudiar cómo la reflexión de la luz afecta el resplandor emitido por una superficie.



$$L_o(\mathbf{p}, \omega_o) = \int_{\mathcal{H}^2} f_r(\mathbf{p}, \omega_i, \omega_o) L_i(\mathbf{p}, \omega_i) \cos\theta \, \mathrm{d}\, \omega_i$$

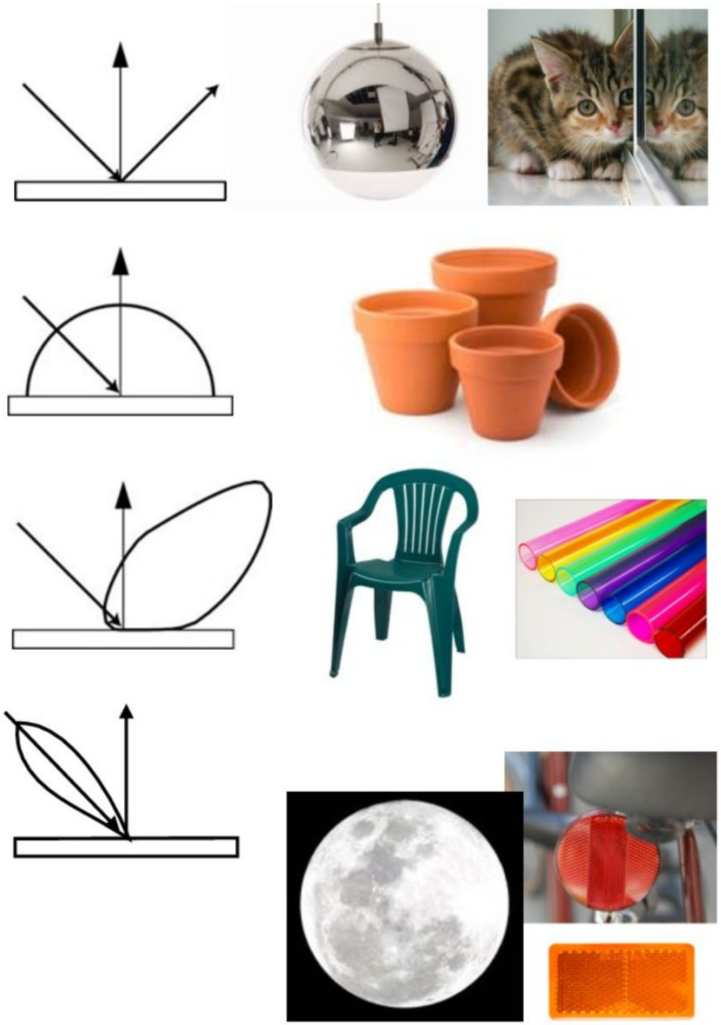
Algunas funciones de reflexión

Especular ideal - funciona como un espejo perfecto

Difusión ideal - la luz se disemina de manera uniforme en todas las direcciones

Especular brillante - la mayoría de la luz se distribuye en la dirección del reflejo

Retro-reflectiva - devuelve la luz hacia su fuente



Materials: diffuse

Materials: plastic

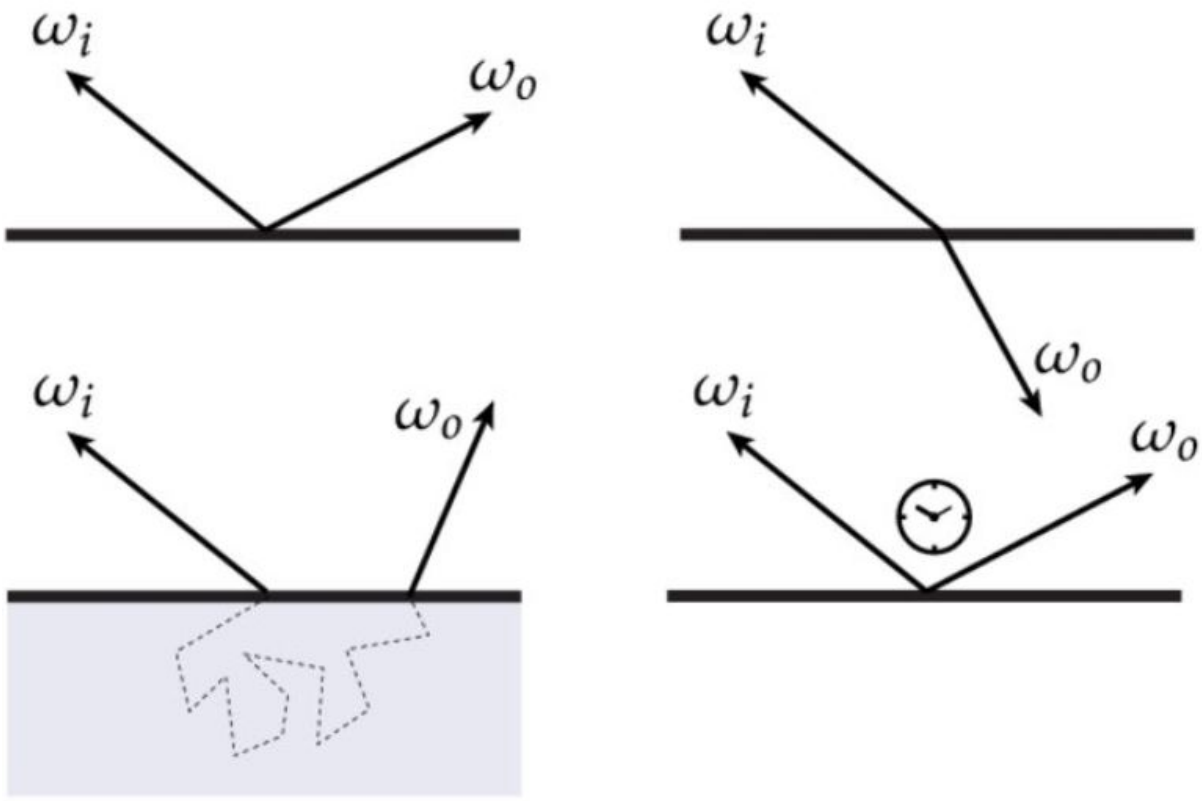
Materials: red semi-gloss paint

Materials: Ford mystic lacquer paint

Materials: mirror

Materials: gold

Materials



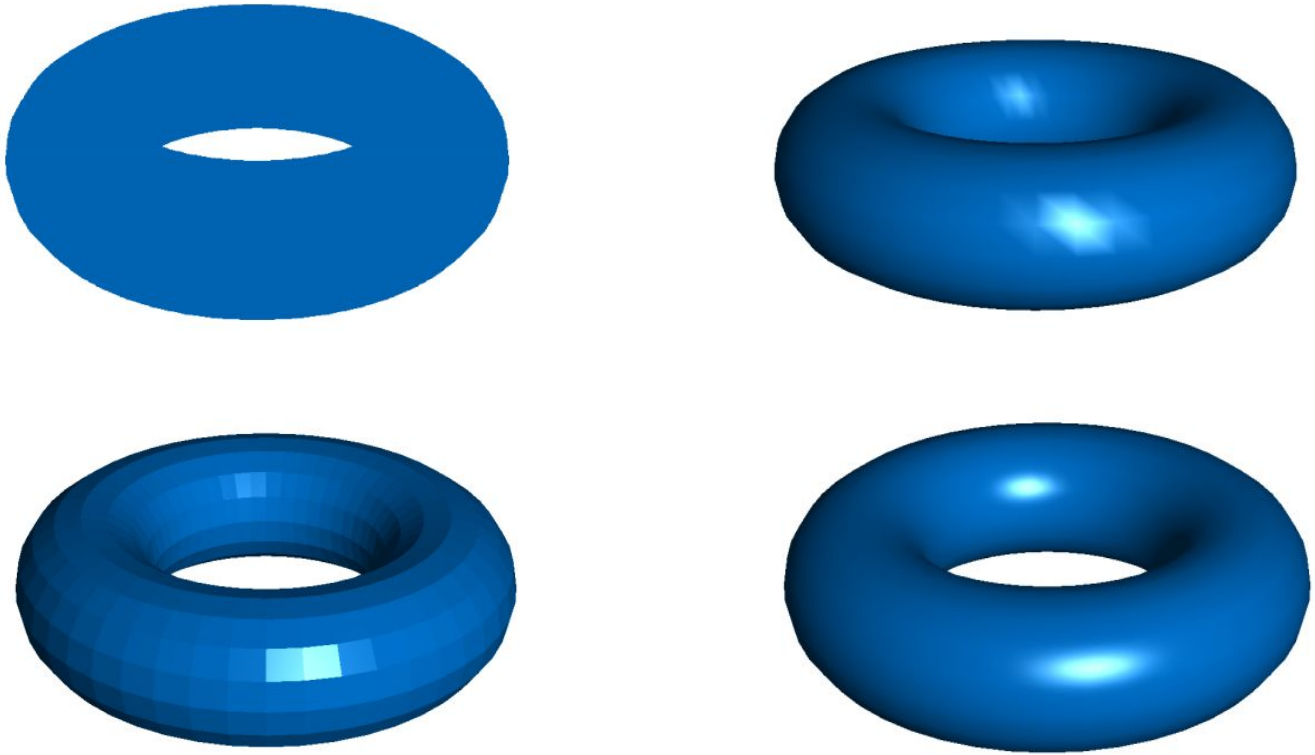
Otra dificultad: la luz no se refleja de manera simple, lo puede hacer de varias maneras.

Translucent materials: Jade

Translucent materials: skin

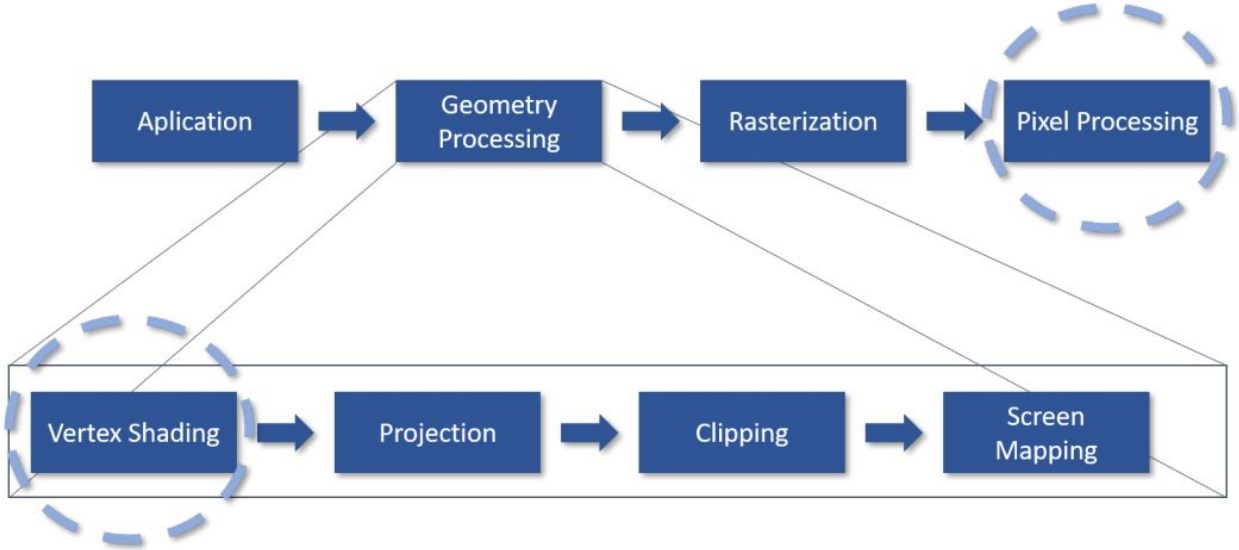
Translucent materials: leaves

Modelos Locales de Rendering

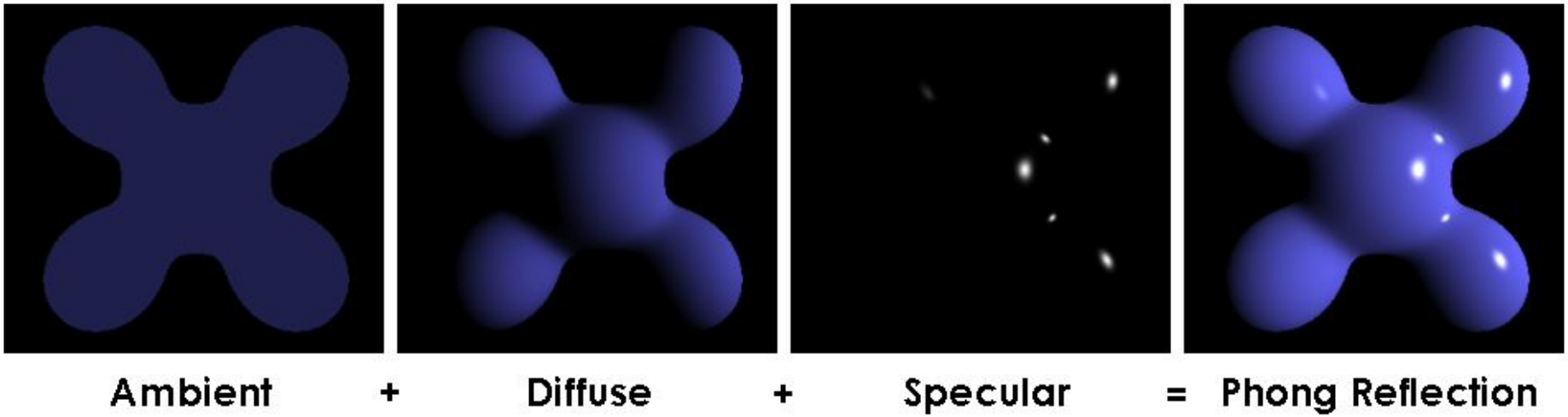


Maarten Everts – Renderizado con OpenGL, Dominio público.

Graphics Rendering Pipeline



Modelo de Phong



De acuerdo al modelo, el color final en cada pixel es la suma de los colores determinados por el material ambiental, difuso, especular y su interacción con la luz.

Modelo de Iluminación de Phong

Es un modelo de iluminación **local**

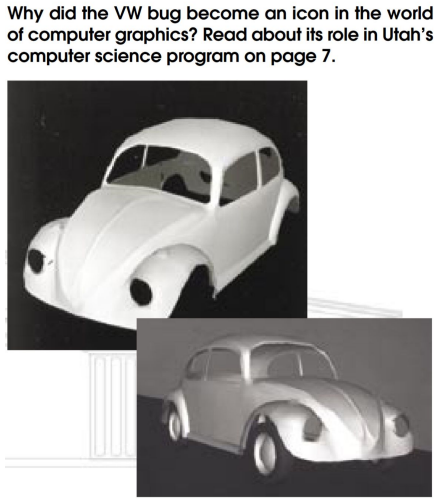
No considera **reflexiones**

No considera proyecciones de **sombras**

Es una simple **aproximación** y no se basa en modelos físicos de iluminación

Utilizado en computación gráfica por su simpleza

¡Puede ser implementado en GPU para ejecución en tiempo real!



Bui Tuong Phong
(1942 – 1975)

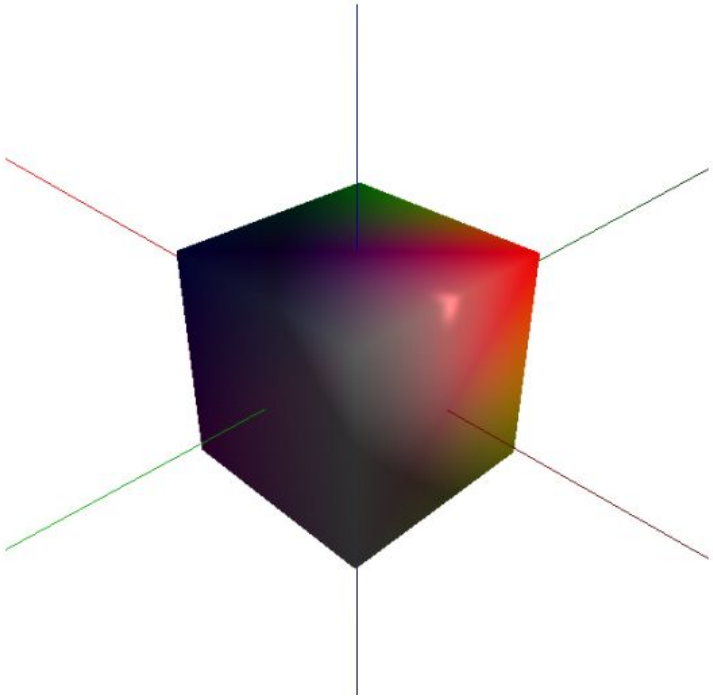

No Lighting
Per-Vertex Lighting
Per-Pixel Lighting
Unreal Engine
1
2
3
Copyright © NVIDIA Corporation 2006
Unreal © Epic

¿Qué necesitamos?

Fuente de luz

Material / color / textura

Geometría de la superficie (normales de triángulos)



Fuentes de luz

Luz Ambiental: Fuente o dirección no identificable

Luz puntual: Generada por un punto

Luz distante: Sólo apreciamos una dirección

Spotlight: Desde una fuente a una dirección específica, como una lintera

- Ángulo de corte define un cono de luz

Luz Ambiental

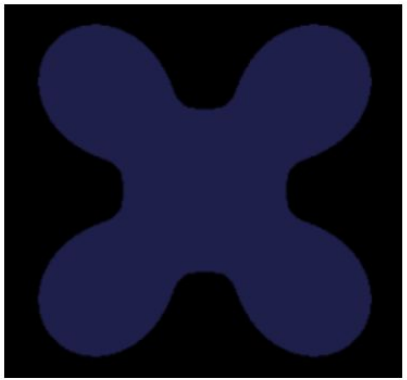
Modela un color base del objeto en la escena

La escena completa puede ser más o menos brillante regulando estos colores

Evita la oscuridad absoluta

Es independiente de las fuentes de luz

Económica computacionalmente

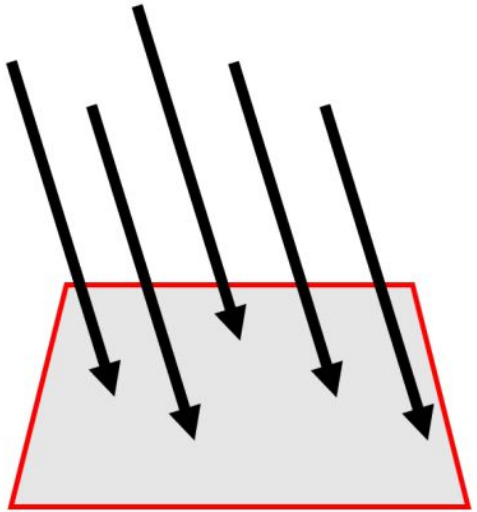


Fuente de luz direccional

Dirección es dada por un vector 3D que afecta de igual manera a toda la escena

Ejemplos de uso:

- Sol
- Luna



Fuentes de luz puntuales

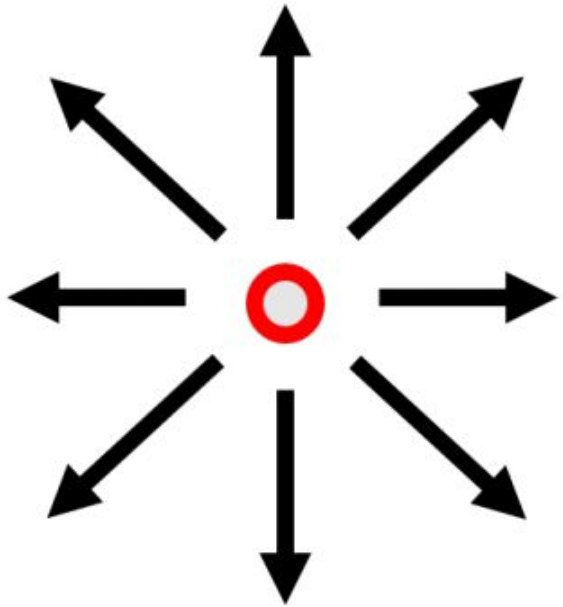
Luz es emitida en todas direcciones

Generada en un punto

Su intensidad se atenúa con la distancia $I \propto \frac{1}{\|p - p_0\|^2}$

Se suele usar un polinomio cuadrático para la atenuación, produce un mejor efecto gráfico:

$$C = \frac{1}{k_c + k_l d + k_q d^2} C_0 \qquad d = \|Q - P\|$$



Fuente de luz spotlight

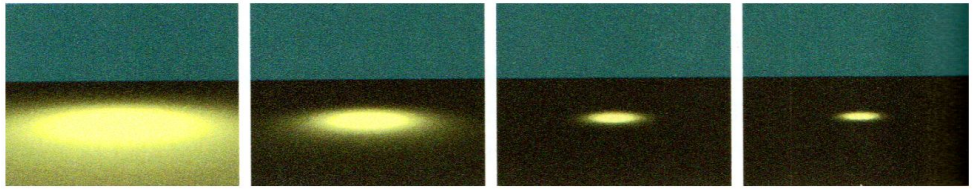
Generada en un punto, pero en una dirección preferencial

Si la fuente de luz se ubica en P y emite en la dirección R, el color resultante C en un punto Q está dado por:

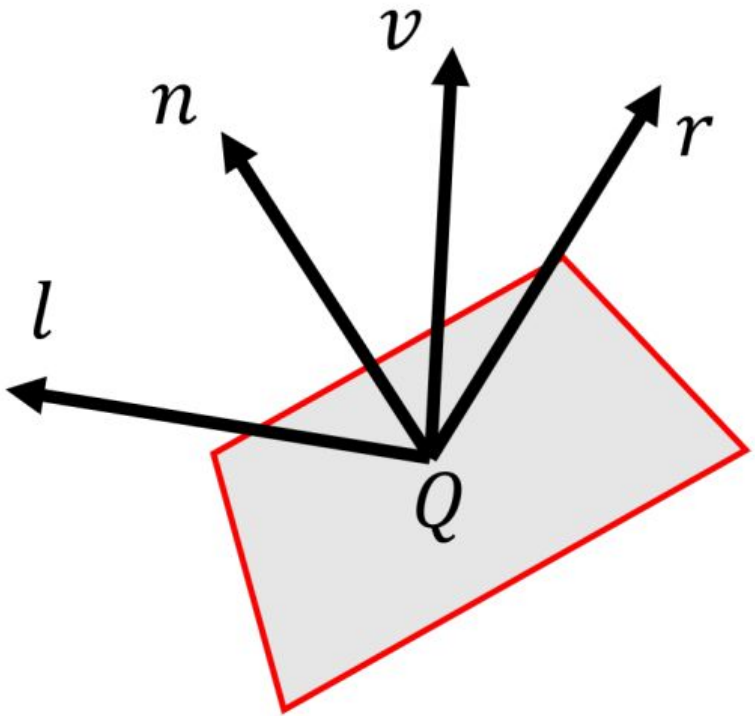
$$\mathcal{C} = \frac{\text{máx}\,(-R \cdot L, 0)^p}{k_c + k_l d + k_q d^2} \mathcal{C}_0 \qquad d = \|Q - P\|$$

$$L = \frac{P - Q}{\|P - Q\|}$$

El exponente p controla la concentración de la luz en la dirección preferencial (2, 10, 50, 100)



Modelo de iluminación de Phong

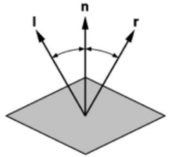


l - vector hacia la fuente de luz

n - normal de la superficie

v - vector hacia la cámara

r - reflexión de la luz l en el punto Q



$\boldsymbol{l} \cdot \boldsymbol{n} = \cos(\theta) = \boldsymbol{n} \cdot \boldsymbol{r}$

$\boldsymbol{r} = \alpha\, \boldsymbol{l} + \beta\, \boldsymbol{n}$

Solution: $\alpha = -1$ and $\beta = 2\,(\boldsymbol{l} \cdot \boldsymbol{n})$

$$\boldsymbol{r} = 2\,(\boldsymbol{l} \cdot \boldsymbol{n})\,\boldsymbol{n} - \boldsymbol{l}$$

Modelo de iluminación de Phong

Comenzar con la **luz ambiente** (igual para toda la escena).

Agregar **contribuciones de cada fuente de luz**. Saturar el resultado en el rango [0,1] (clamp) (Cada canal de color se calcula independientemente)

Las contribuciones de cada fuente de luz pueden ser descompuestas en

- Reflexión **ambiental**
- Reflexión **difusa**
- Reflexión **especular**

Un objeto puede comportarse de manera distinta según el tipo de componente de la luz: **puede tener un color específico para la reflexión ambiental, otro para la difusa y otro para la especular**.

- Estos colores pasan a constituir una descripción de material (archivos .mtl)

Reflexión Ambiental

La intensidad L_a es la misma para toda la escena $\mathcal{I}_a = \mathcal{K}_a \mathcal{L}a$

¡Pero la manera en que cada material K_a la refleja puede variar por objeto!

Aumentando $\mathcal{K}_a$

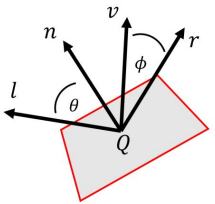
Reflexión Difusa

Un objeto reflector difuso disemina la luz a su alrededor

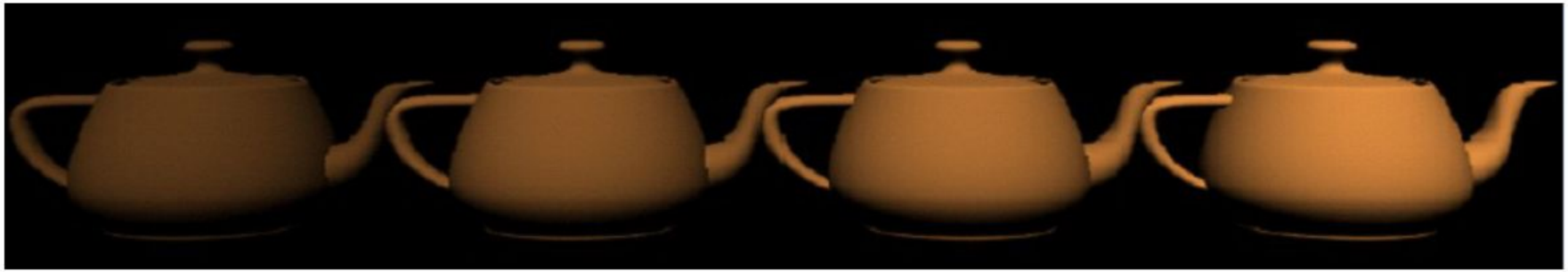
Este tipo de superficie es llamada **superficie de Lambert**

K_d es el coeficiente difuso del material

El ángulo de incidencia es relevante

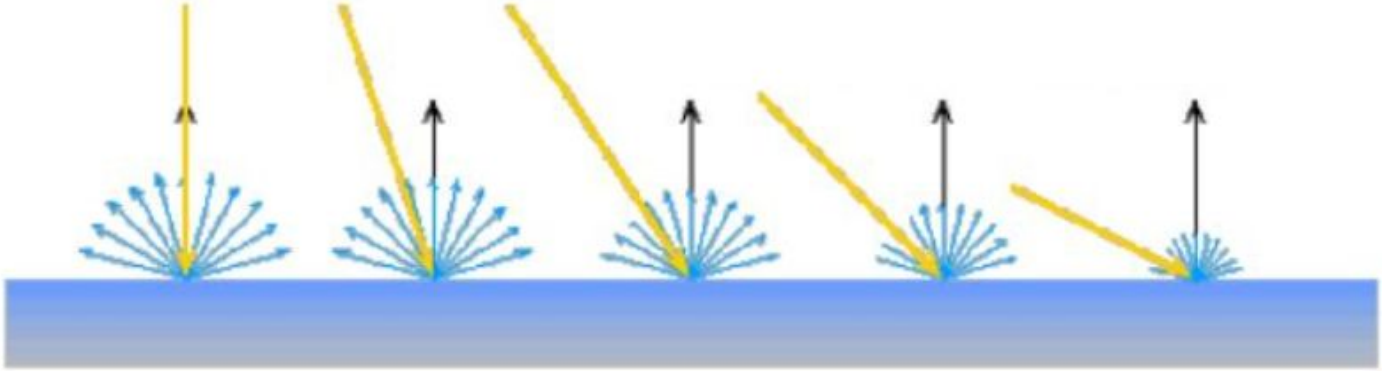


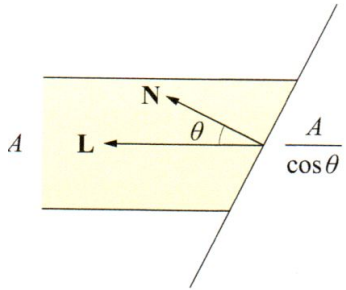
$$\mathcal{I}_d = \frac{\mathcal{K}_d \mathcal{L}_d}{k_c + k_l d + k_q d^2}(l \cdot n) \qquad d = \|Q - P\|$$



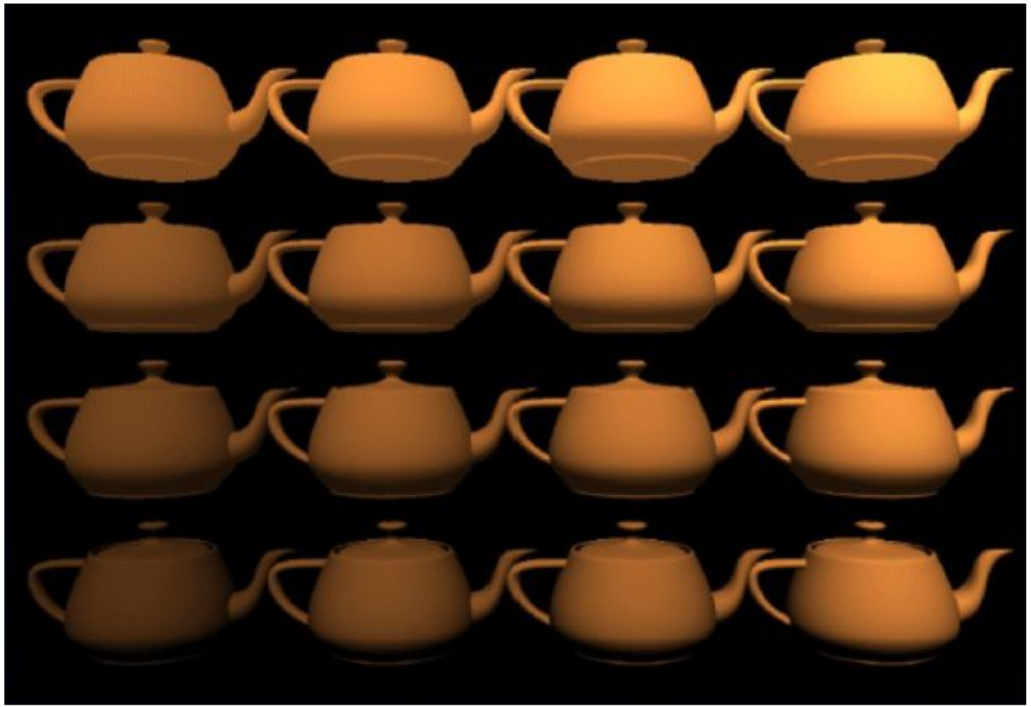
Aumentando $\mathcal{K}_d$

Reflexión difusa – Superficie de Lambert





Reflexión ambiental y difusa



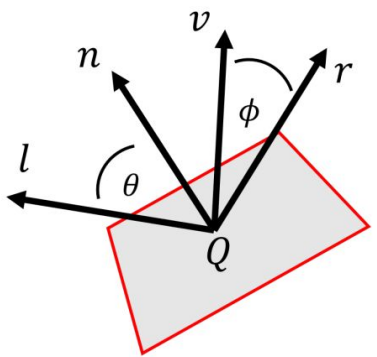
Hacia la derecha aumenta $\mathcal{K}_d$
Hacia arriba aumenta $\mathcal{K}_a$

Reflexión Especular (“brillitos”)

Se define de manera análoga (intensidad de luz, características del material)

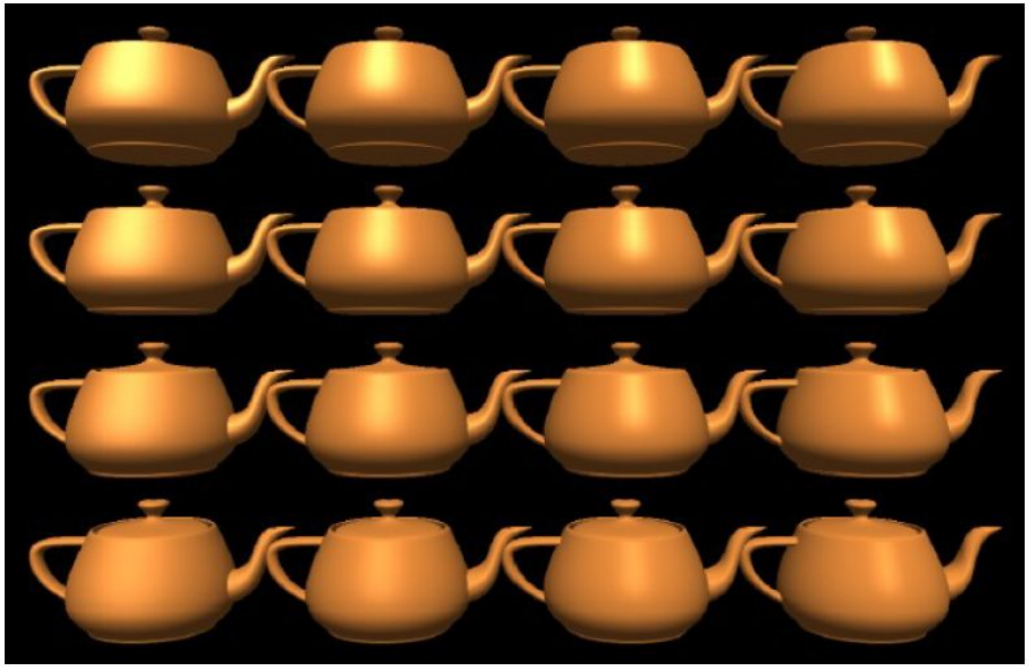
Sin embargo, se utilizan los vectores hacia la vista y de reflexión

Es **especular** pero no estamos hablando de un espejo



$$\mathcal{I}_s = \frac{\mathcal{K}_s \mathcal{L}_s}{k_c + k_l d + k_q d^2} (v \cdot r)^{\alpha} \qquad d = \|Q - P\|$$

Reflexión especular



Hacia la derecha aumenta α
Hacia arriba aumenta K_s

Modelo de iluminación de Phong

$$\mathcal{I} = \mathcal{K}_a\mathcal{L}_a + \frac{1}{k_c + k_l d + k_q d^2}(\mathcal{K}_d\mathcal{L}_d(l \cdot n) + \mathcal{K}_s\mathcal{L}_s(v \cdot r)^\alpha)$$

$$d = \|Q - P\|$$

Importante

- $\mathcal{L}_a$, $\mathcal{L}_d$, $\mathcal{L}_s$, $\mathcal{K}_a$, $\mathcal{K}_d$ y $\mathcal{K}_s$ se aplican para cada independientemente para R, G y B
- Luego, $\mathcal{L}_a$, $\mathcal{L}_d$, $\mathcal{L}_s$, $\mathcal{K}_a$, $\mathcal{K}_d$ y $\mathcal{K}_s$ son representadas compactamente como colores

Modelos de Sombreado

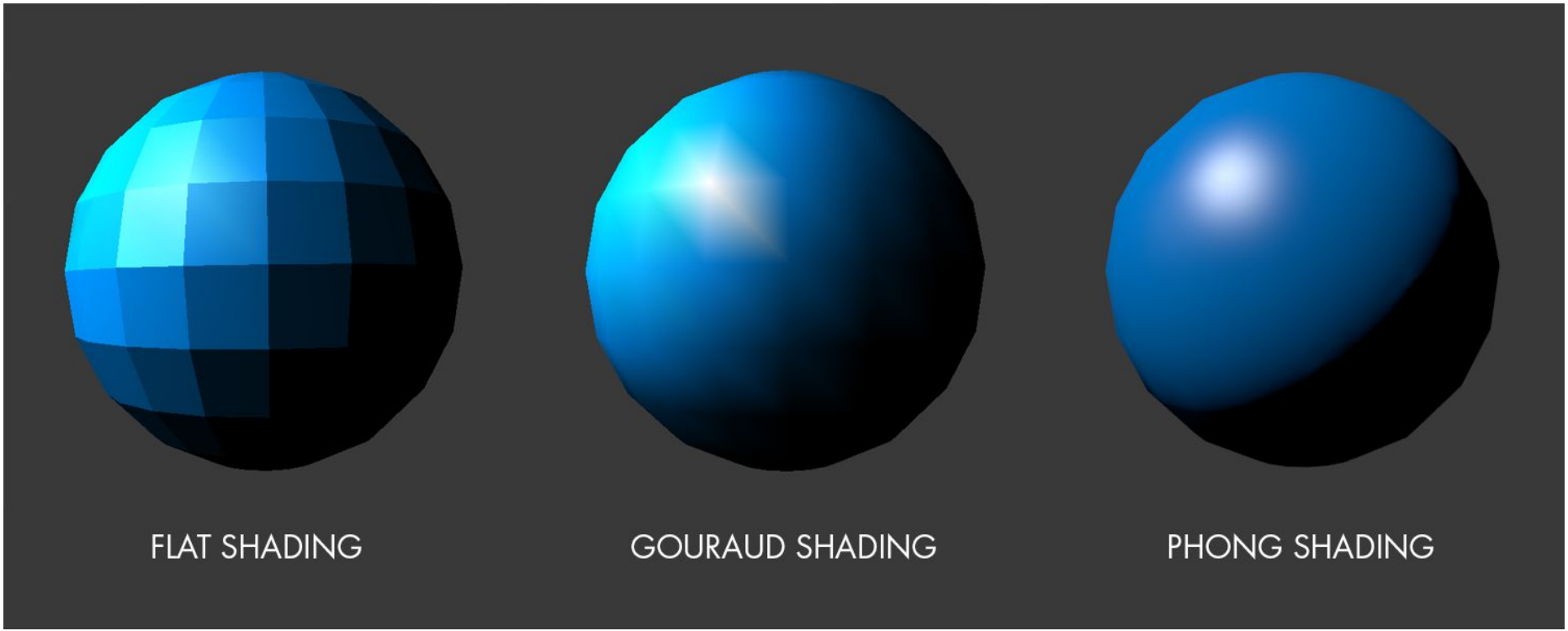
El modelo de iluminación de Phong nos indica cómo calcular el color final para un punto arbitrario en la superficie

Sin embargo, **nuestros modelos son discretizados**, por lo que conocemos solo algunas de las posiciones de cada trozo de superficie.

Tenemos distintas estrategias:

- **Flat**: Asigna un único color a cada cara poligonal
- **Gouraud**: Calcula el color para los vértices que definen cada cara poligonal
- **Phong**: Calcula el color para cada pixel contenido en la cara poligonal.

Modelos de Sombreado



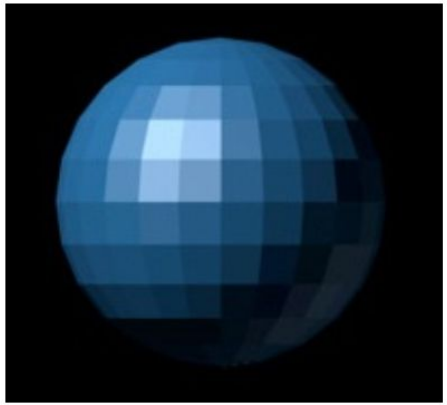
Sombreado Plano

Cada vértice del triángulo puede tener distinto color

Simplemente escogemos uno, y con ese pintamos el triángulo completo.

Con muchos triángulos, o caras poligonales en general, podemos apreciar degradaciones generadas por las fuentes de luz

Es el método más económico



Sombreado de Gouraud

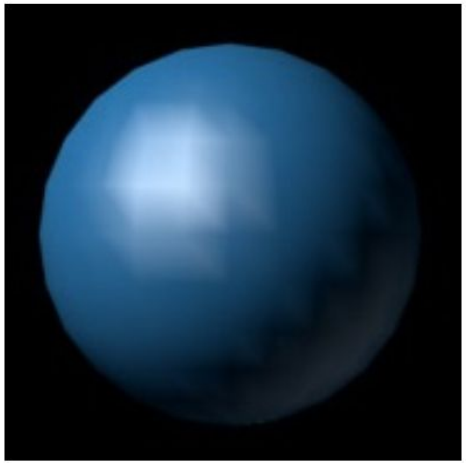
Como cada vértice del triángulo puede tener distinto color, colores en el interior del triángulo pueden ser simplemente interpolados

Se produce un mejor efecto en superficies suaves

Podemos aproximar la normal en cada vértice como el promedio normalizado de las normales de las caras vecinas

Se requiere conocimiento sobre qué caras comparten determinado vértice

- Más adelante veremos estructuras de datos para mallas poligonales

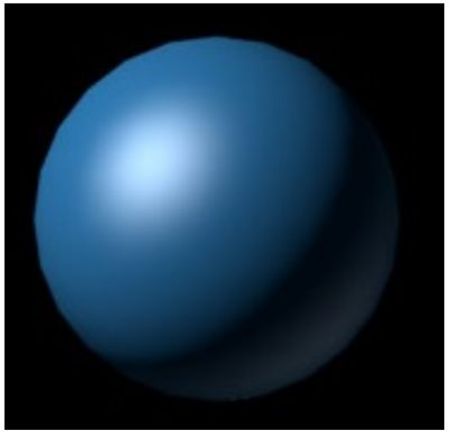


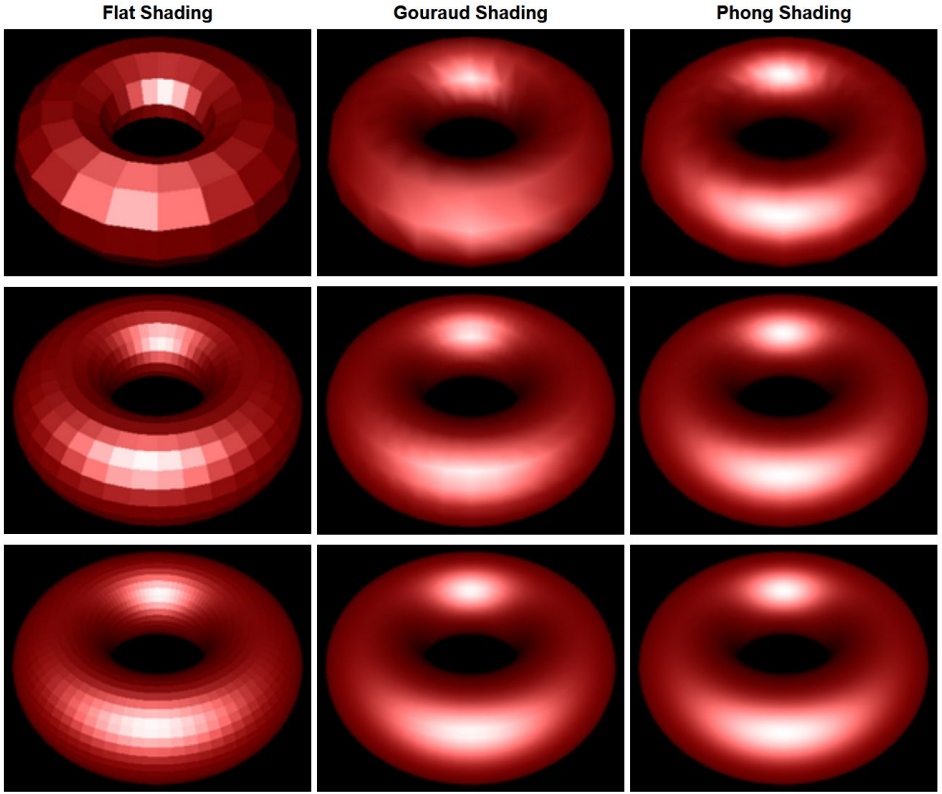
Sombreado de Phong

A diferencia del método de Gouraud, aquí interpolaremos la normal al interior de cada triángulo, y con ella calcularemos el color asociado a cada píxel.

Se produce un excelente efecto en superficies suaves

Significativamente más costoso



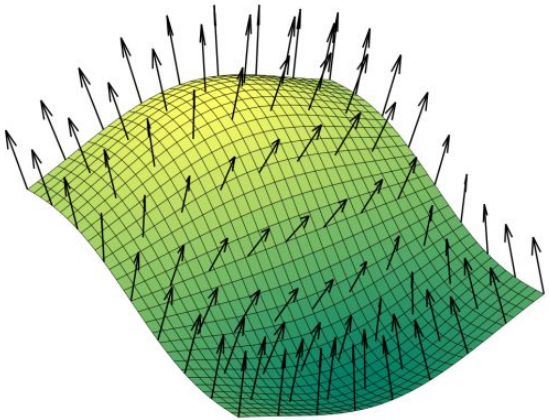


La resolución es importante.

Vectores Normales

El vector normal a una cara nos entrega información de su orientación en el espacio

Cuando se transforma el modelo, se deben convertir vértices y normales

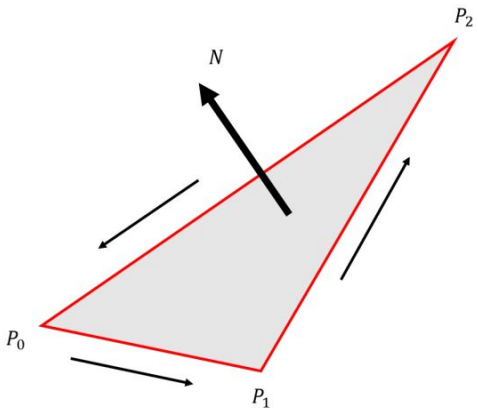


By Nicoguaro - Own work, CC BY 4.0

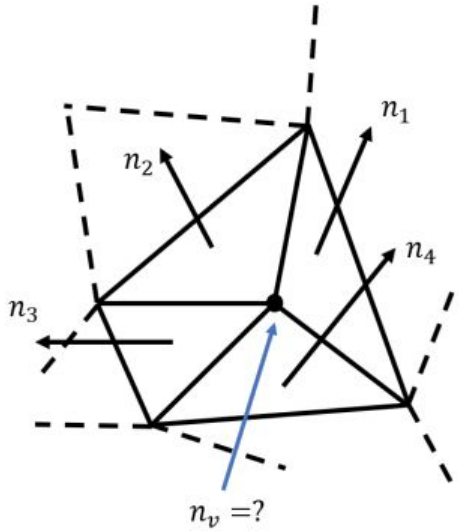
Normal de un triángulo (en caso de no tener normales)

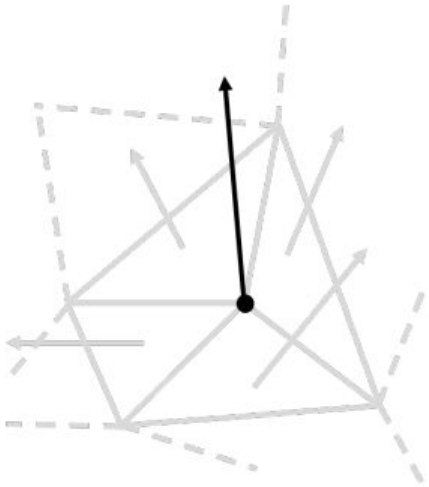
Dado un triángulo cuyos vértices están ordenados en sentido contrario a las manecillas del reloj (counter-clockwise), podemos calcular la normal hacia afuera con la ecuación

$$N = \frac{(P_1 - P_0) \times (P_2 - P_0)}{\|(P_1 - P_0) \times (P_2 - P_0)\|}$$



Normal en un vértice (en caso de no tener normales)



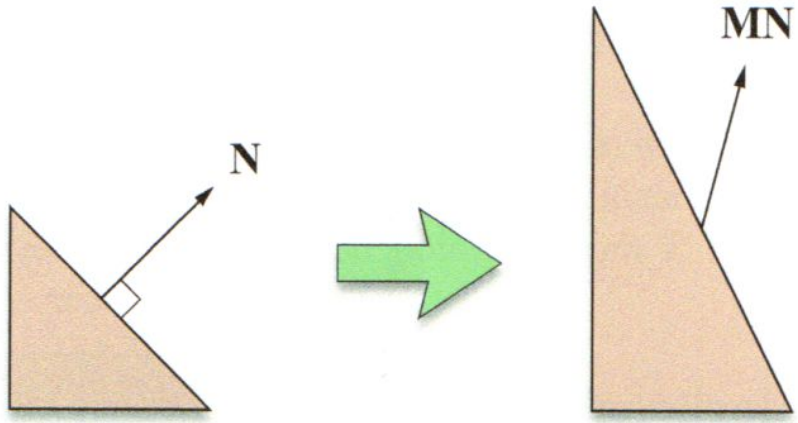


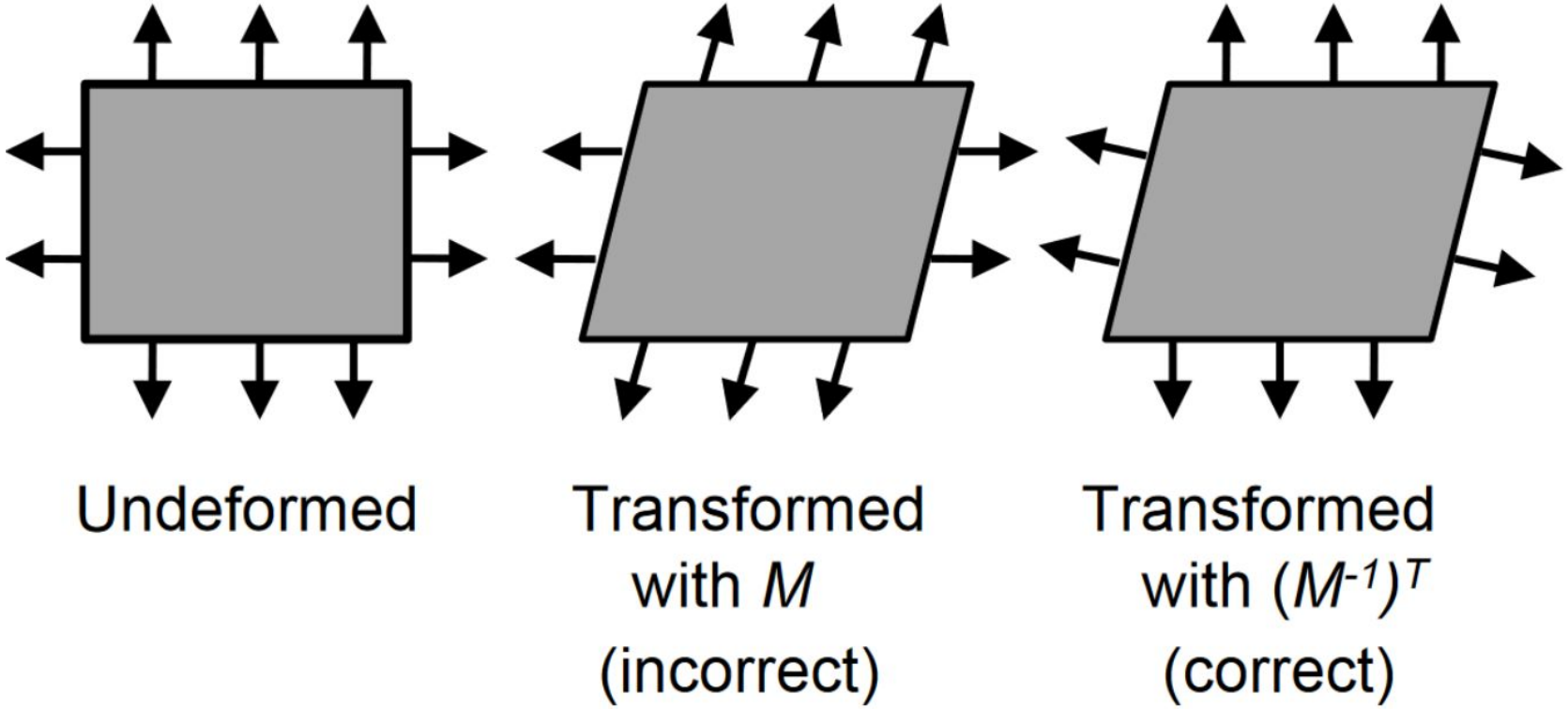
Aproximación de Gouraud: la normal de un vértice se aproxima como el promedio normalizado de las normales de las caras vecinas

$$n_v = \frac{n_1 + n_2 + n_3 + \ldots}{\|n_1 + n_2 + n_3 + \ldots\|}$$

Problema: transformando vectores normales

Si la transformación no es ortogonal, la dirección de las normales se verá afectada:





Algunas transformaciones deforman la normal o bien no se aplican correctamente. **¿Qué hacer en este caso?**
Propuesto: leer por qué la operación (M^(-1))^T funciona:
http://www.lighthouse3d.com/tutorials/glsl-12-tutorial/the-normal-matrix/

¿Preguntas?