



Sobreado (Shading) Iluminación Global

**Capitulo 6 de Angel
Capitulo 14 y 16 de Foley**

Shading

- **Iluminación:** calcula la intensidad de luz en un punto particular de la superficie
- ***Shading*:** usa estos calculos de intensidades para sombrear o matizar toda la superficie o la escena. Se refiere al proceso de asignación de color a los pixels.

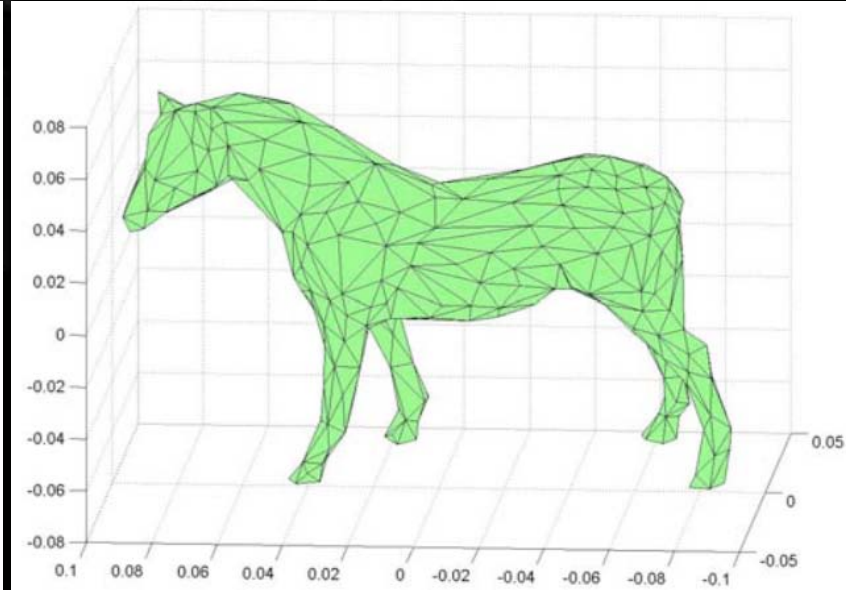
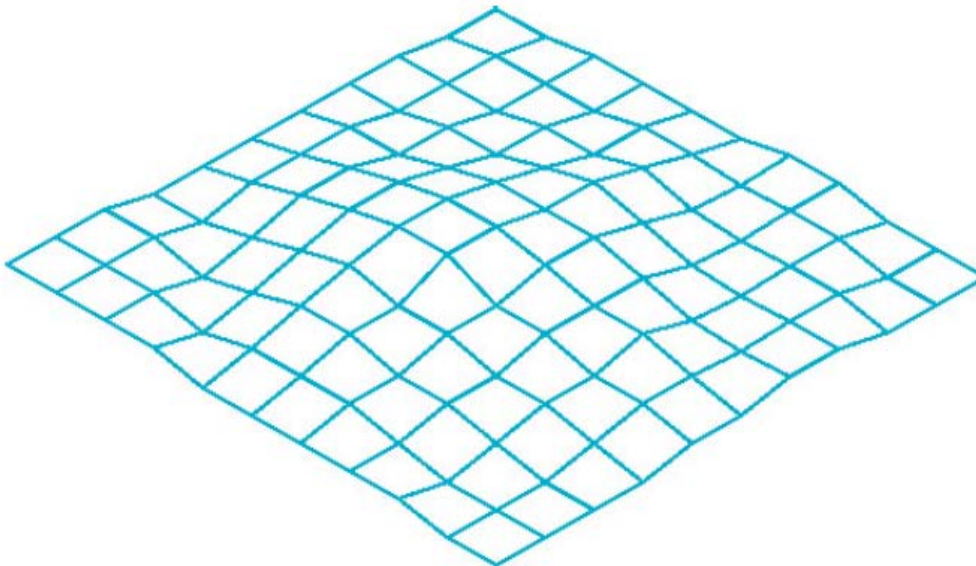
Shading de Superficies

- Puede ser aplicada en una superficie usando el modelo de iluminación local
- Superficie plana
 - Normales, intersecciones, visibilidad, proyecciones, etc.
 - Aceleración por hardware
- Superficies no planas
 - Transformada a muchos pequeños poligonos de caras planas (mesh)

Mesh poligonales



- Compuesto por un conjunto de poligonos (cuadrilateros o triangulos) conectados por sus bordes
- Modelo dominante para formas 3D (esp. objetos de formas libres)

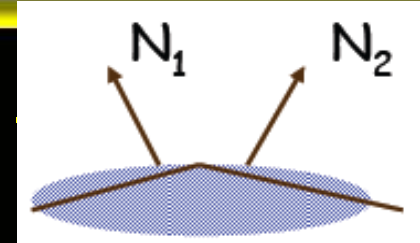


Tipos de sombreado



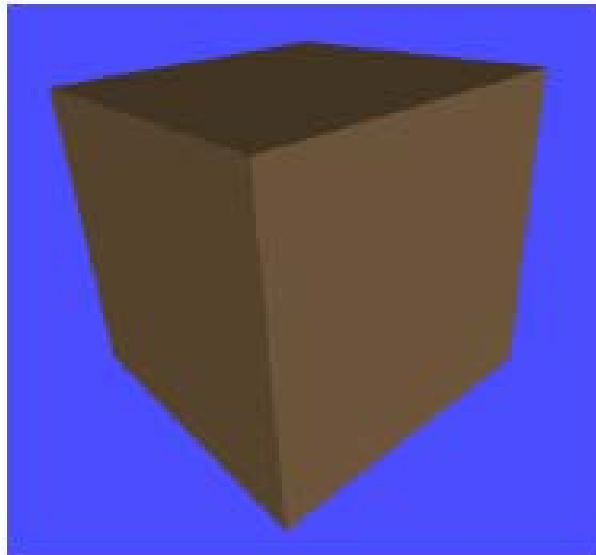
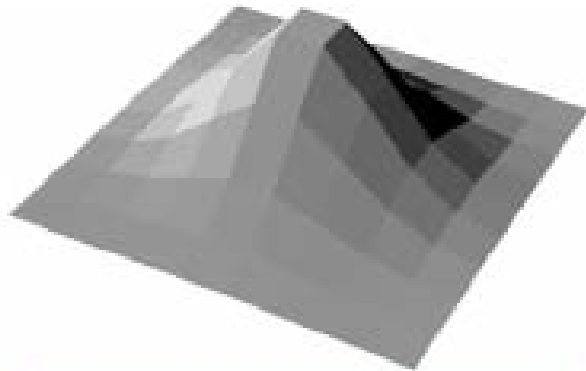
- Sombreado plano
 - Calcula la iluminación solo una vez por polígono y se aplica a todo el polígono:
`glShadeModel(GL_FLAT)`
- Interpolated/smooth/Gouraud
 - Calcula la iluminación en los bordes y se interpola: `glShadeModel(GL_SMOOTH)`
- Phong shading
 - Calcula la iluminación en cada punto del polígono

Flat Shading (sombreado plano)



- Se aplica un solo color a todo el polígono (eficiente)
- Discontinuidad de intensidades en las aristas
- A menos que
 - La fuente luminosa está en el infinito, por tanto $N \cdot L$ es constante
 - El observador está en el infinito, por tanto $N \cdot V$ es constante en toda la cara del polígono.
 - El polígono representa la superficie real que se modela y no es una aproximación a una superficie curva.
- También si se crea el objeto o la forma con un gran número de polígonos así el efecto entre polígonos es menos obvio a nuestros ojos

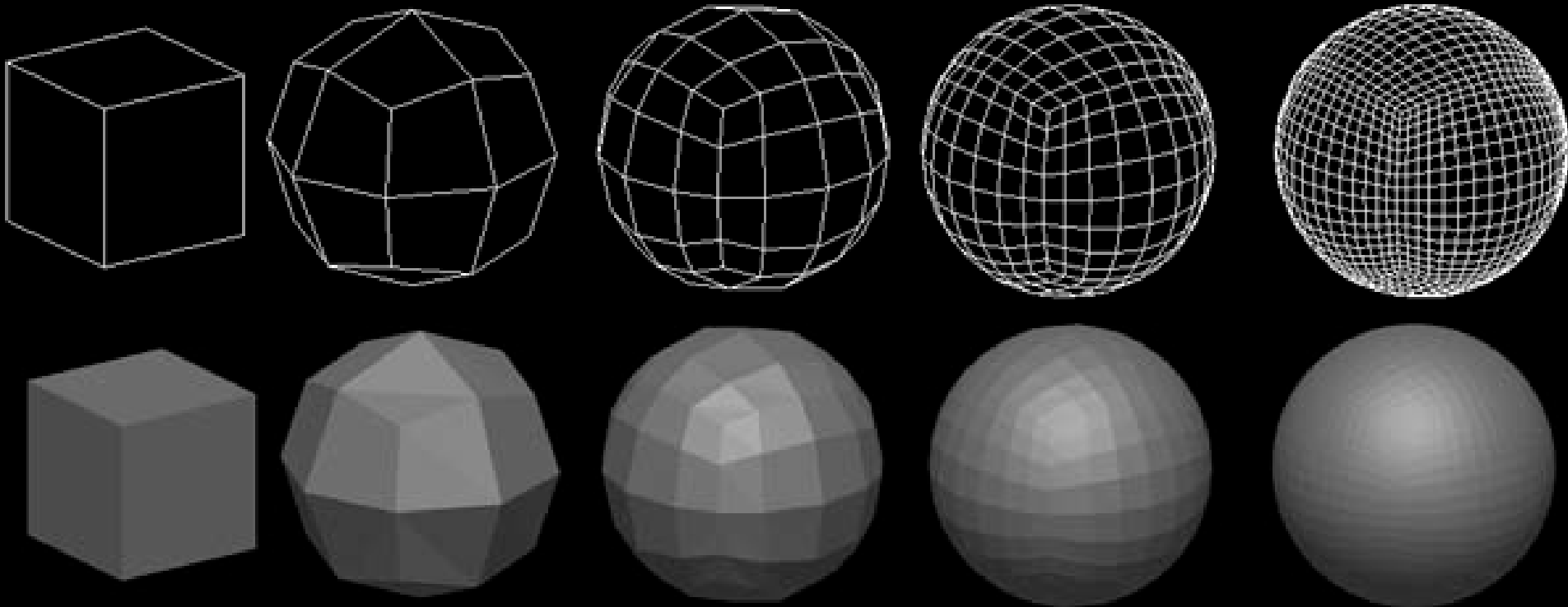
Flat Shading



Sombreado de malla poligonal

Una superficie curva se puede aproximar a otra facetada (malla poligonal)

No se logran buenos resultados en la interpolación, aunque se trabaje con una densidad alta de polígonos.

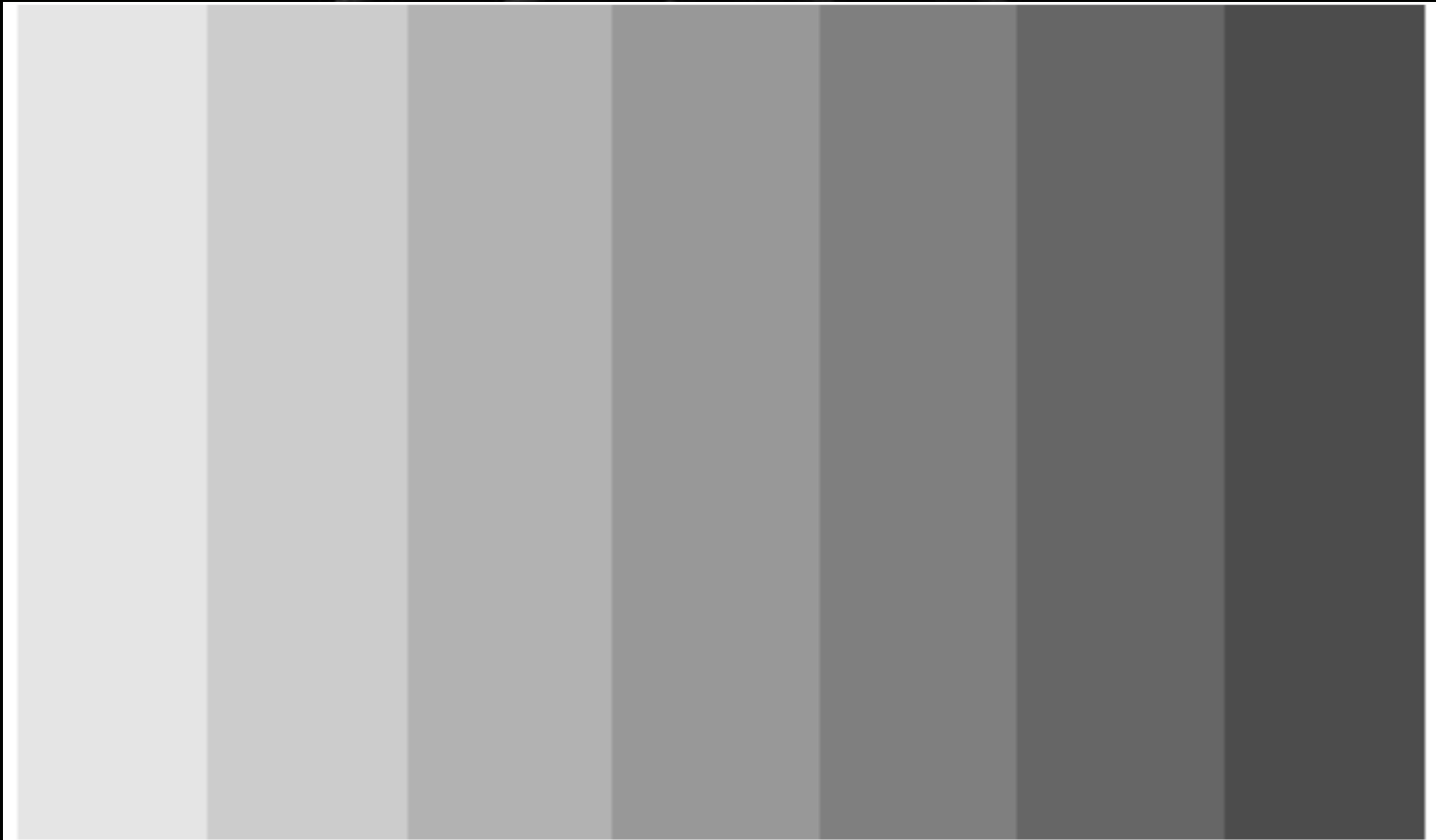


Observese que aun en la malla de más alta densidad, al sombreado los polígonos se logran distinguir las facetas

Sombreado de malla poligonal



Los problemas en la visualización de una superficie curva a través de una aproximación facetada, tienen su origen en el efecto de banda de Mach.



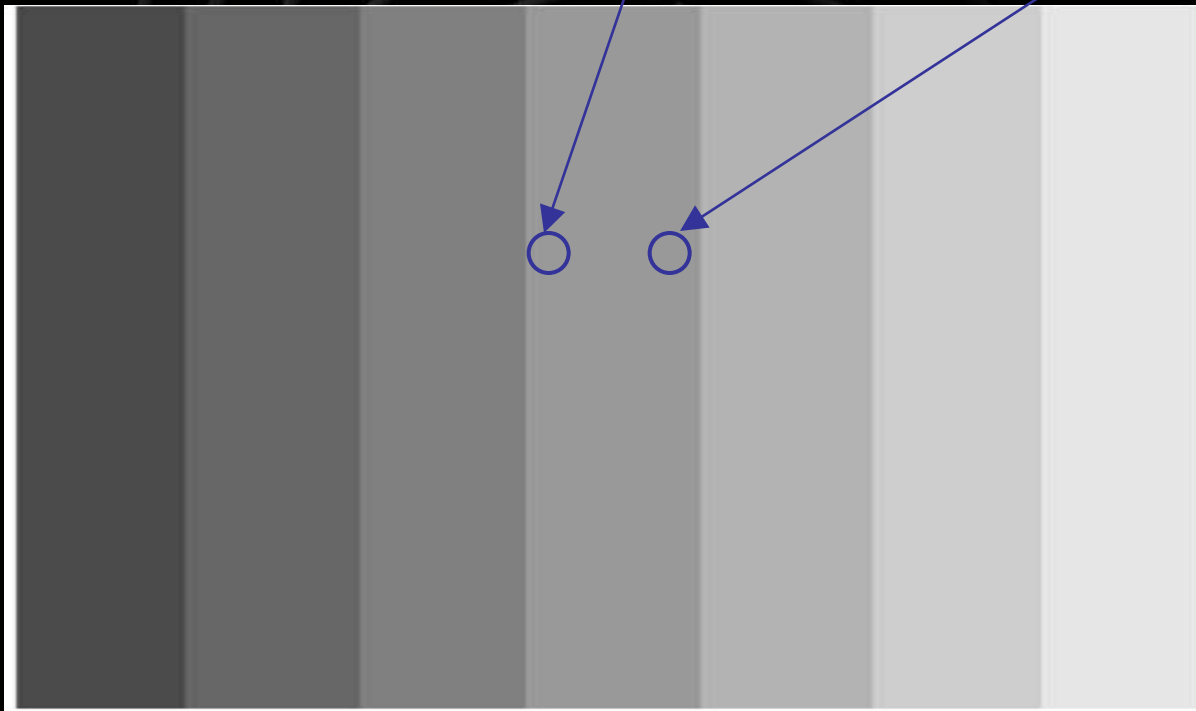


Sombreado de malla poligonal

Los problemas en la visualización de una superficie de una aproximación facetada, tienen su origen en el efecto de banda de Mach.

O sea, además de la discontinuidad existente, el ojo resalta las discontinuidades provocando que se vean aún mayores de lo que en realidad son.

Aquí vemos unas bandas. Cada una tiene un color único. Sin embargo si observamos con cuidado una de ellas vemos que la parte izquierda parece tener un color más claro que la parte derecha de la misma banda.

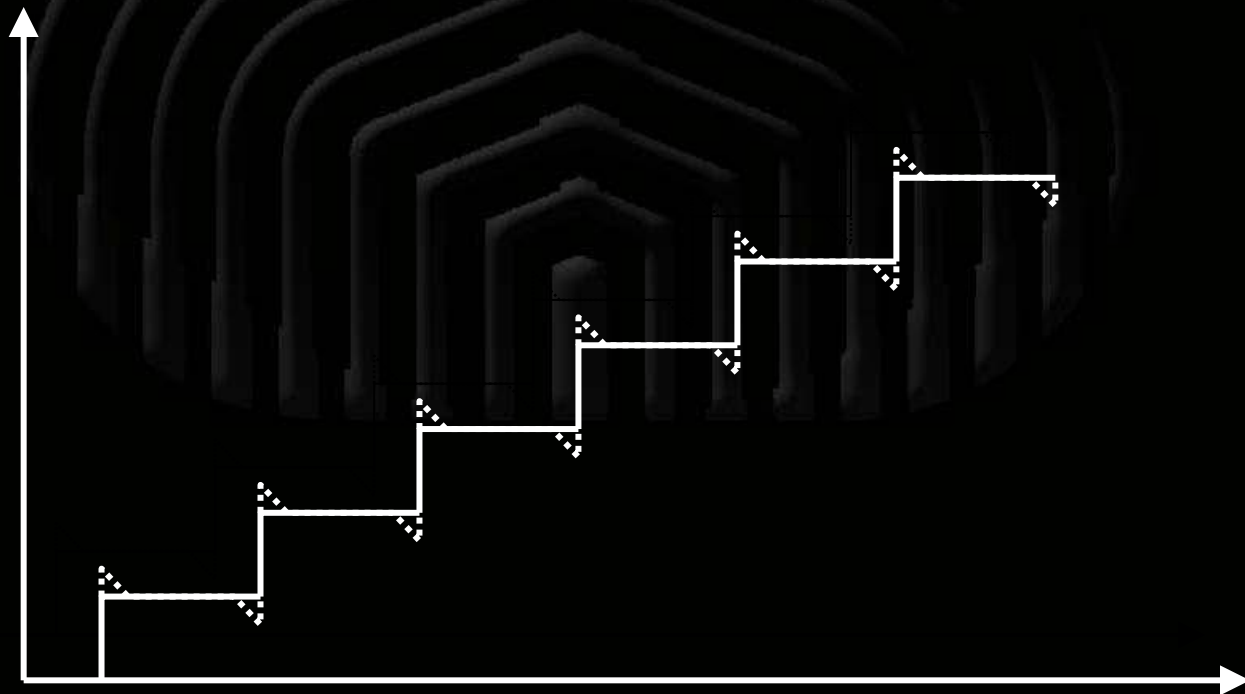


Sombreado de malla poligonal

Esquema de las intensidades reales y las percibidas.

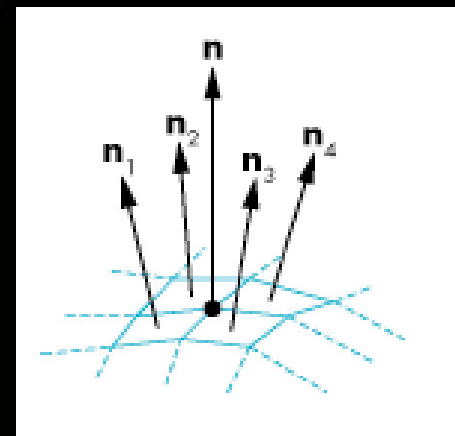
Si bien las intensidades son uniformes en cada banda, en las discontinuidades, el salto percibido es mayor al real.

Las líneas punteadas es la intensidad percibida, la línea sólida es la real.



Gouraud Shading

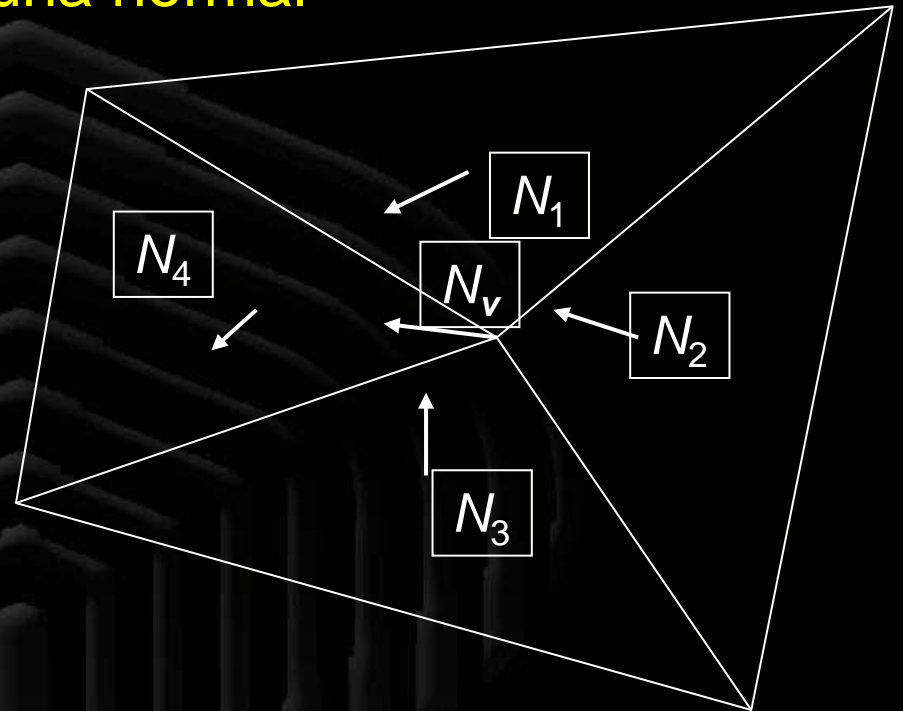
- Gouraud es una metodología que permite lograr simular continuidad en la malla poligonal, sin necesidad de tener que utilizar una malla muy densa.
- Gouraud calcula una normal ficticia para cada vértice de la malla, interpolando las normales a las caras adyacentes. Luego, calcula el color de cada vértice y por último, realiza una interpolación de los valores de los vértices para calcular los valores de luz interiores a cada polígono.



Sombreado de Gouraud

1) A cada vértice se le asigna una normal

$$\bar{N}_v = \frac{\sum_{1 \leq i \leq n} \bar{N}_i}{\left| \sum_{1 \leq i \leq n} \bar{N}_i \right|}$$

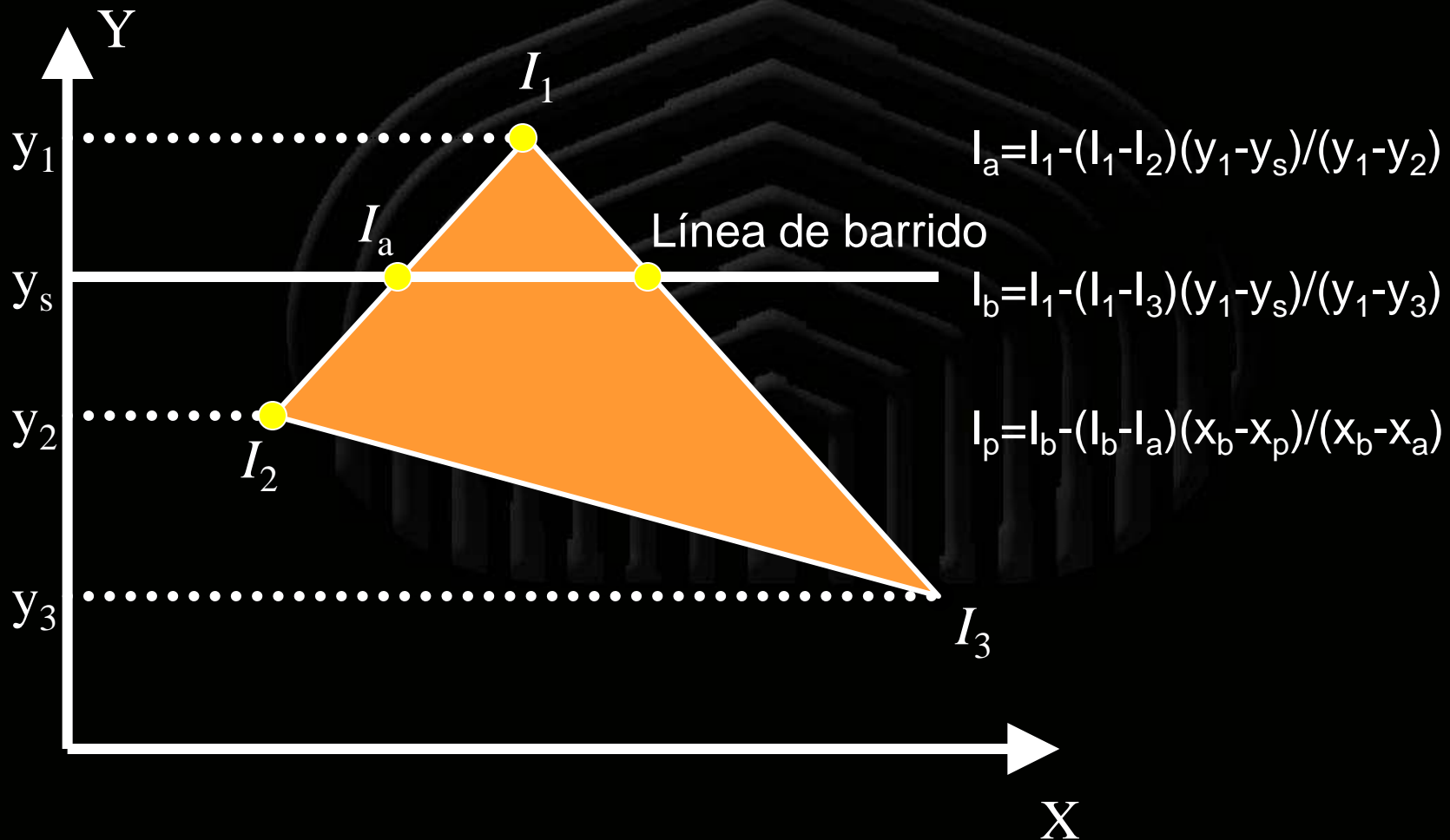


2) Se calculan las intensidades de los vértices usando algún modelo de iluminación ya visto.

3) Se interpola la intensidad en cada pixel del polígono.

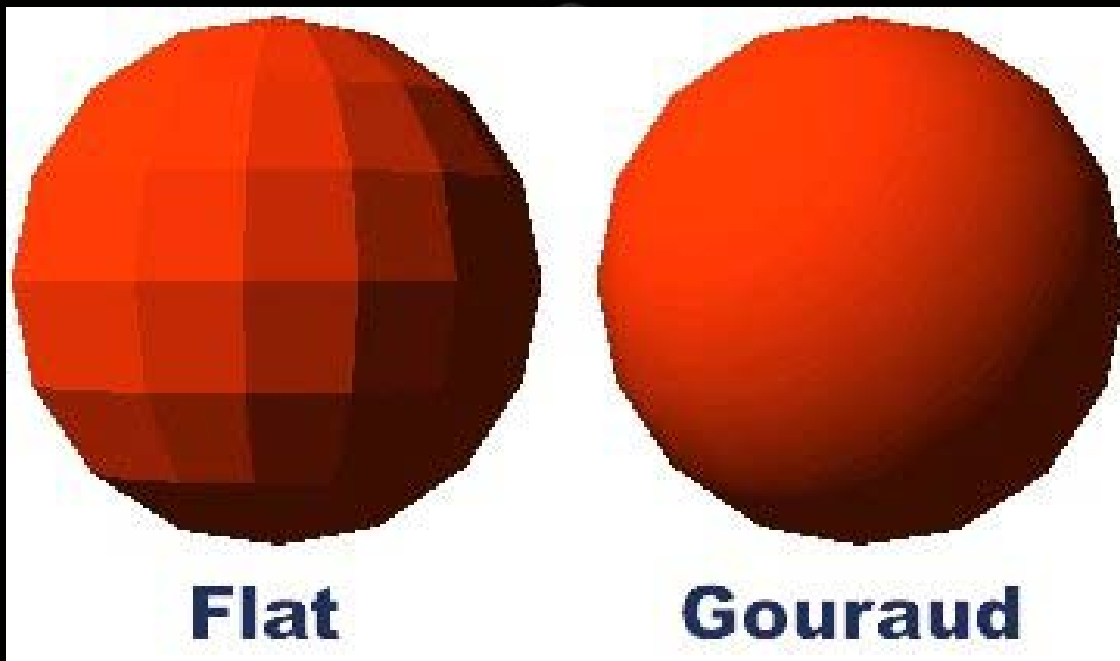
Sombreado de Gouraud

3) Se interpola la intensidad en cada pixel del polígono.





Sombreado de Gouraud



Flat vs Gouraud Shading

Flat Shading

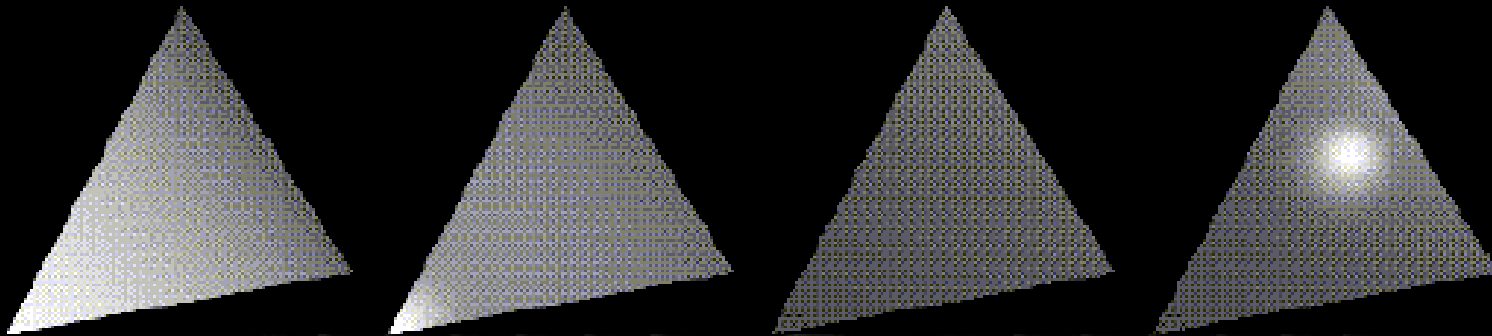


Gouraud Shading



Phong Shading

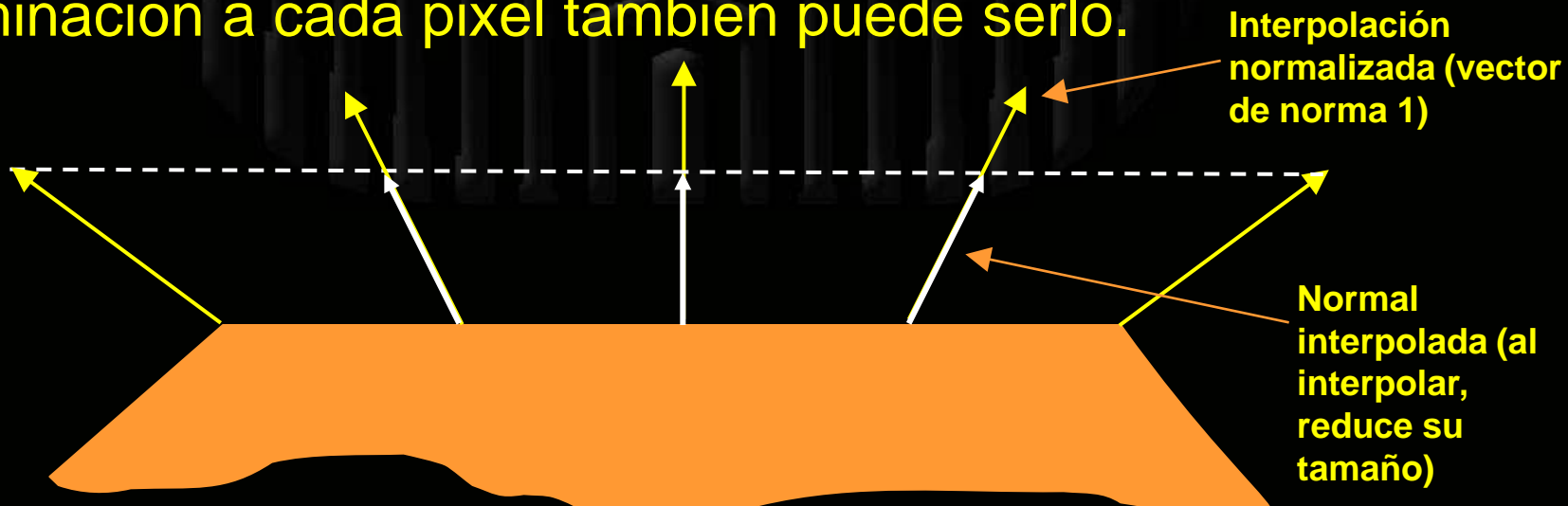
- Gouraud shading no maneja apropiadamente la iluminación especular debido a la interpolación de colores



- Phong shading: asegura el sombreado
 - Interpolando las normales en cada punto en vez de colores
 - Luego aplica el modelo de iluminación local según la normal aproximada

Sombreado de Phong

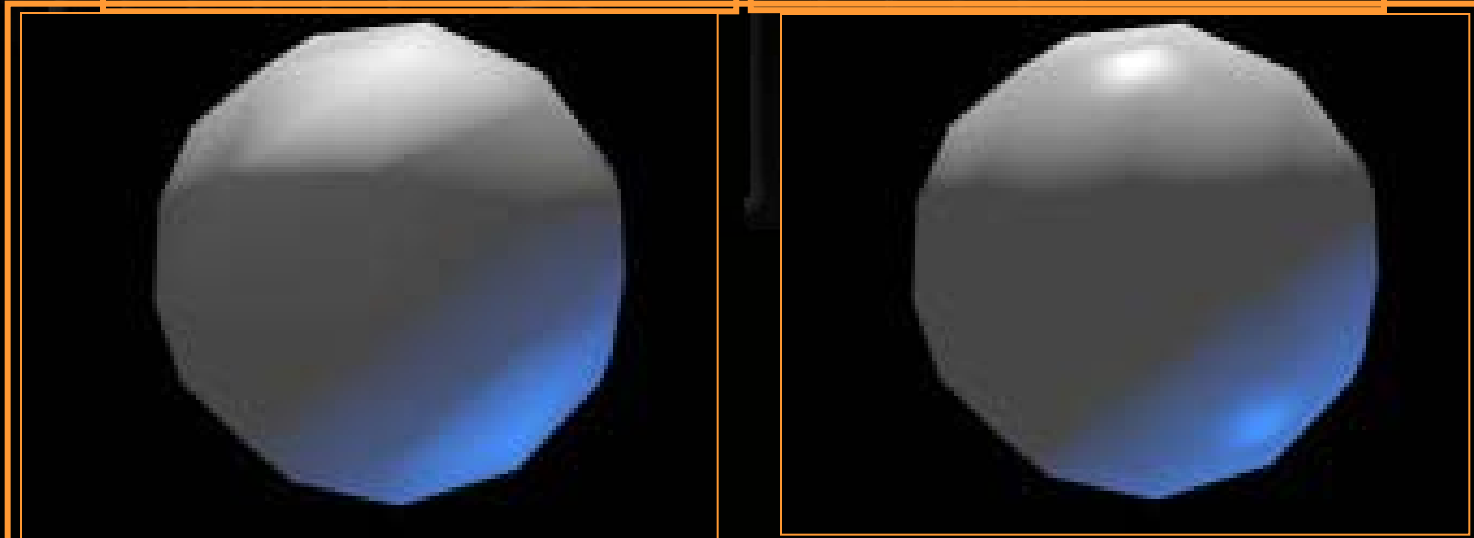
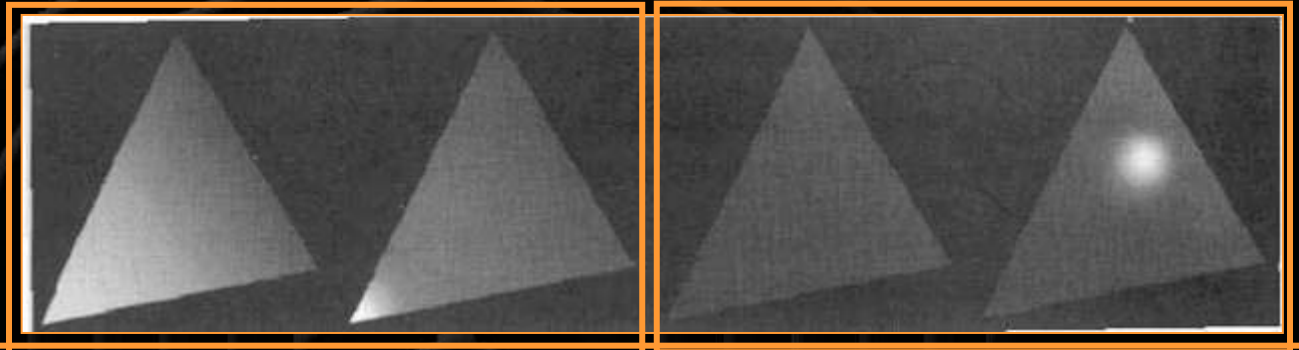
- En Gouraud se interpola el color de los vértices.
- En Phong se interpola (y normaliza) la normal de los vértices y se calcula en cada píxel el color correspondiente.
- Si se utiliza sombreado de Phong con n alto, la diferencia entre Phong y Gouraud puede llegar a ser notable.
- Normalizar un vector es costoso, y aplicar un modelo de iluminación a cada pixel también puede serlo.



Sombreado de Phong



En cada uno de los rectángulos, a la izquierda aplica Gouraud y a la derecha Phong. Si la superficie tiene un componente especular, el efecto de Phong se resalta más. En los triángulos se observa cómo Phong permite iluminar una porción de la superficie que no se iluminaría con Gouraud.



Phong vs Gouraud Shading

- Phong shading:
 - Maneja mucho mejor la iluminación especular
 - Pero es más costoso que el Gouraud Shading

iluminación global

Gran parte de la luz en el mundo real no proviene de fuentes directas.

Algoritmos para abordar este problema:

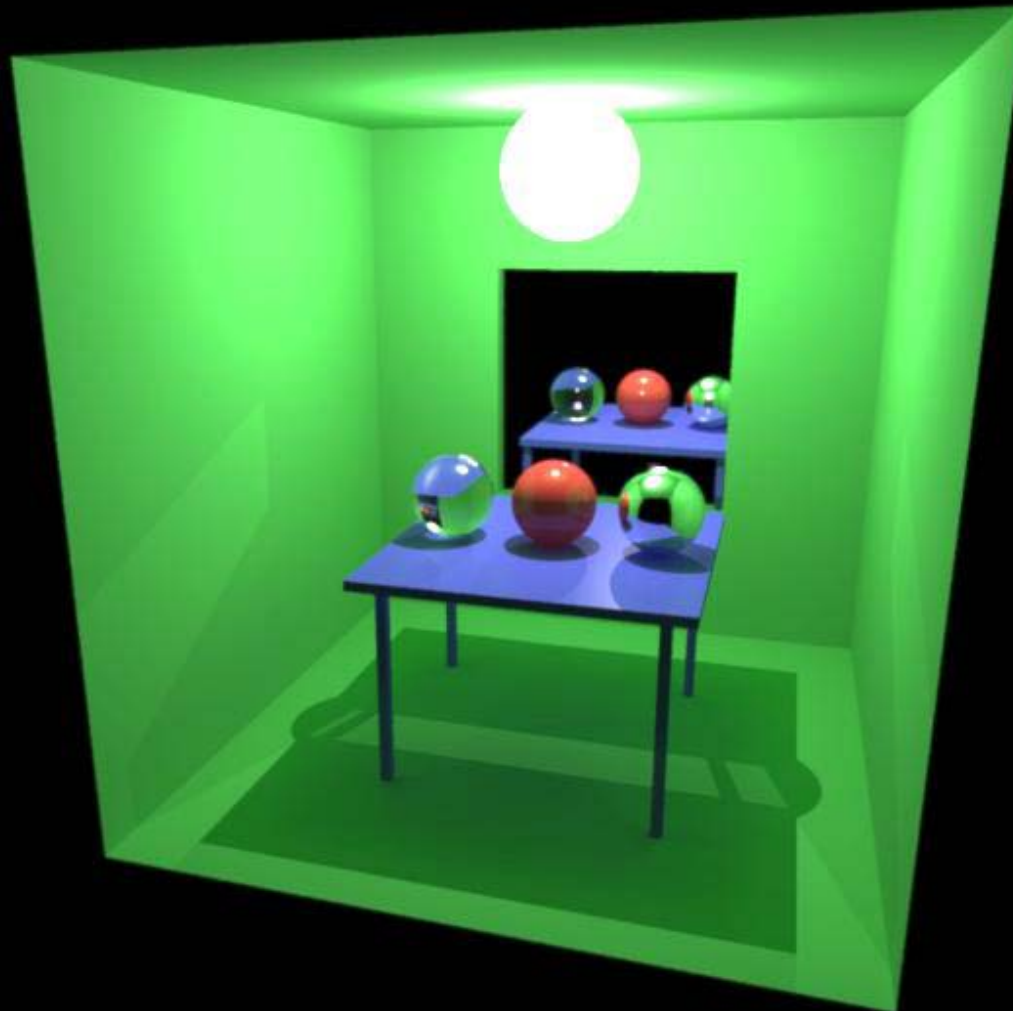
Traza de rayos recursiva (ray tracing)

Es dependiente de la ubicación del observador

Radiosidad (Radiosity)

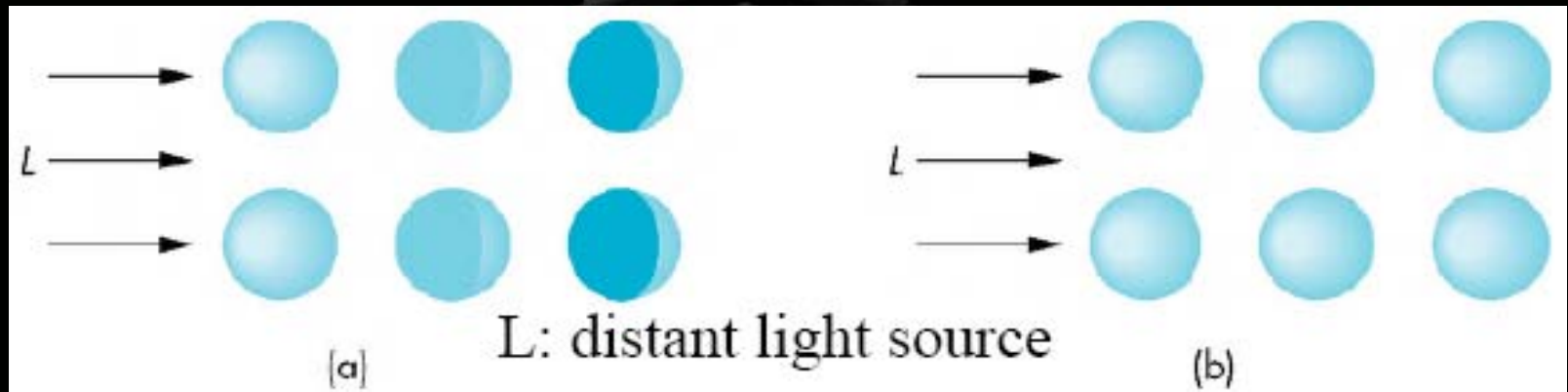
Es independiente de la ubicación del observador

iluminación global



Iluminación global

- Un ejemplo concreto:



- (a) Iluminación Global (GIM): el modelo podría dar sombra, hay multiple reflexión inter-objetos
- (b) Iluminación local (LIM): todas las esferas lucen con la misma intensidad de luz, cada rendering es independiente

GIM en OpenGL

- Usa el modelo de iluminación local de Phong y considera solo iluminación directa por fuentes de luz
- También es posible crear una fuente de luz ambiental global
- El flat shading y sombreado suave son soportados
- Las primitivas geométricas pasan por el pipeline gráfico independientemente
- Se puede usar el z-buffer
- El usuario debe simular la iluminación global con su propia implementación

Iluminación local (repaso)

- El color en un punto p se debe a la fuente de luz L
- $I = \text{global_ambient} + \text{self_emission} + \text{ambient} + \text{attenuation} * [\text{diffuse} + \text{specular}]$
- La componente de luz ambiental es independiente de alguna otra fuente de luz
- El termino emisión no es afectado por ninguna otra fuente de luz y no afecta ninguna otra superficie en la escena
- Si se trabaja en colores se maneja cada componente IR, IG, IB
- Para multiples fuentes de luz se suman las intensidades y se normaliza

Iluminación Global



- Trata de verificar como iluminar toda la escena dependientemente
- El modelo de iluminación local es necesario en el proceso
- Incorpora todas las fuente de luz en la escena, especialmente la reflejada de los objetos en la escena y no solo las que provienen de alguna fuente de luz directamente

Algoritmos de GIM



- Solución de fuerza bruta:
 - para todas las fuentes en todas las direcciones
 - trazar un rayo hasta que de en la pantalla o salga de escena
 - Computacionalmente inmanejable
- Dos algoritmos establecidos:
 - Ray tracing: discretiza el plano de la imagen
 - Radiosity: discretiza el ambiente

Imágenes con GIM



Ray Tracing



Radiosity

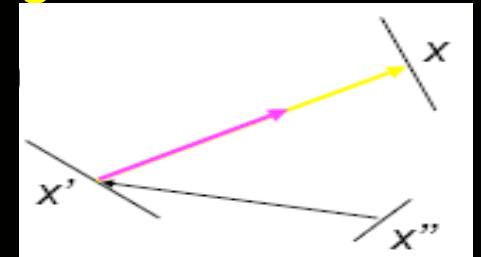
Ecuación de rendering en GIM

- Siguiendo las leyes físicas de la conservación de energía:
 - Se asume un ambiente cerrado
 - Imagina un número infinito de rayos alrededor con energía de absorción, emisión y reflexión
- Se alcanza un estado de equilibrio, el cual es calculado con la ecuación de rendering
- Ray tracing parece ser el algoritmo que usa la forma más simplificada de la ecuación de rendering

Ecuación de generación

$$I(x, x') = g(x, x') \left[\varepsilon(x, x') + \int_s \rho(x, x', x'') I(x', x'') dx'' \right]$$

- En computación gráfica, la ecuación de generación (o “rendering equation”), describe el flujo de luz a través de la escena. Está basada en la física de la luz, provee resultados teóricamente perfectos, en contraste con las técnicas vistas anteriormente, las cuales se aproximan a este ideal.
- Interpretación:
 - La intensidad de luz que sale de x' y llega a x tiene dos componentes:
 - Emisión propia a partir de x'
 - Reflexión en x' desde otro punto x''
 - La oclusión entre x y x' esta codificada en $g(x', x)$



Ecuación de generación

$$I(x, x') = g(x, x') \left[\varepsilon(x, x') + \int_s \rho(x, x', x'') I(x', x'') dx'' \right]$$

x, x', x'' son puntos en el ambiente

$I(x, x')$ es la intensidad que pasa de x' a x

$g(x, x')$ es un término geométrico.

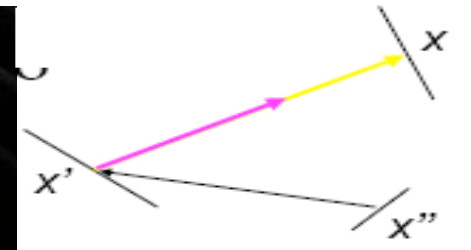
Vale 0 cuando x y x' están ocultos entre si.

Vale $1/r^2$ cuando son visibles (r =distancia(x, x'))

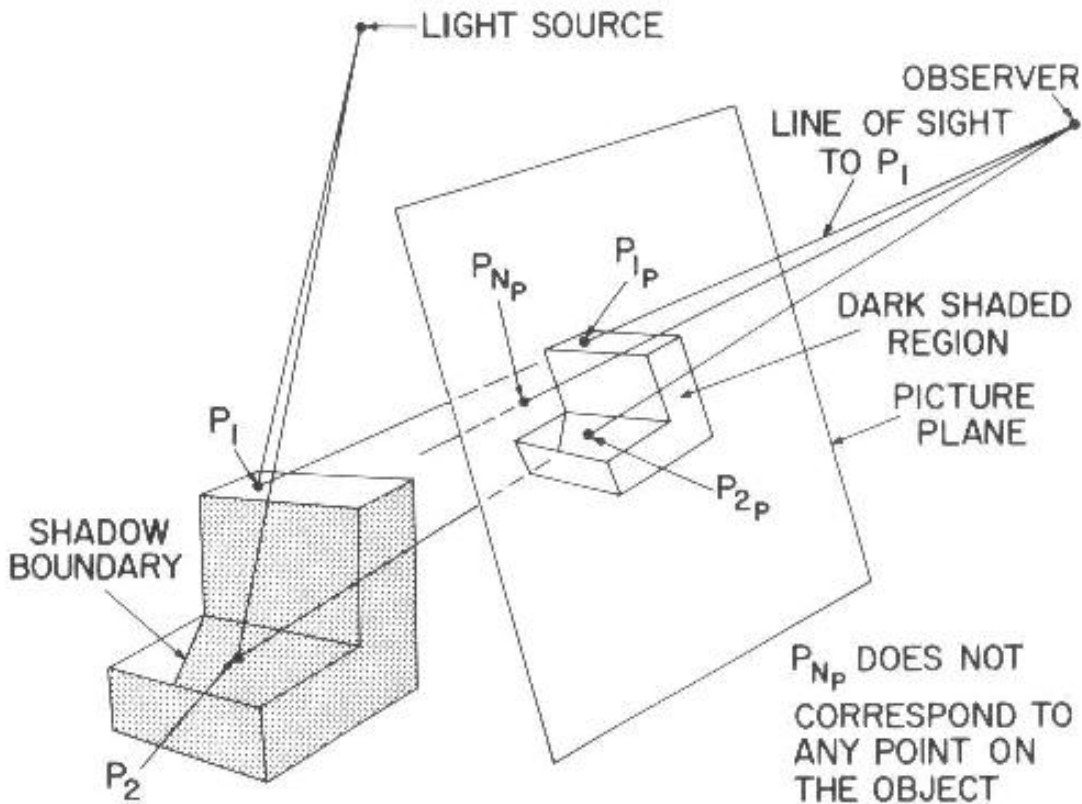
$\varepsilon(x, x')$ intensidad de luz que se emite de x' a x (si x' es fuente de luz)

$\rho(x, x', x'')$ se relaciona con la intensidad de luz reflejada en x' , que sale de x'' y llega a x

La integral refiere a que la luz que se refleja desde x' hacia x , proviene de todos los puntos del espacio (representados por la integral en x'').



ray-tracing

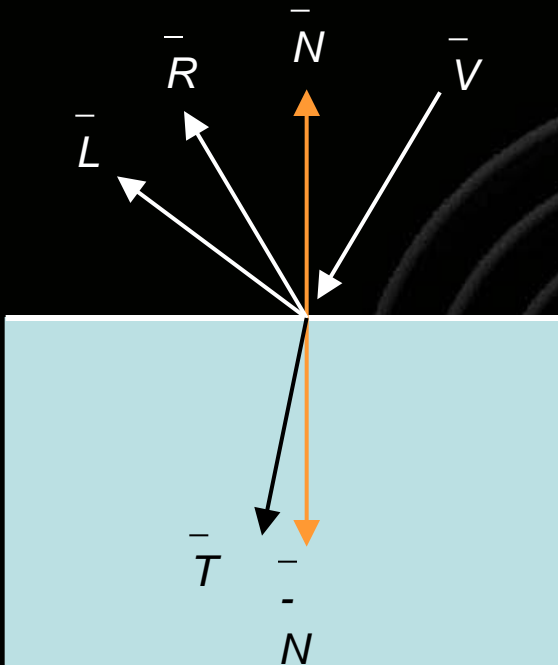


Aquí se lanzan rayos desde el observador hacia el objeto.

Luego, en el punto de intersección se lanzan rayos hacia las fuentes luminosas (llamados rayos de sombra).

Si uno de esos rayos interseca un objeto, dicho objeto está bajo sombra de la fuente de luz correspondiente.

Traza de rayos recursiva (Whitted)

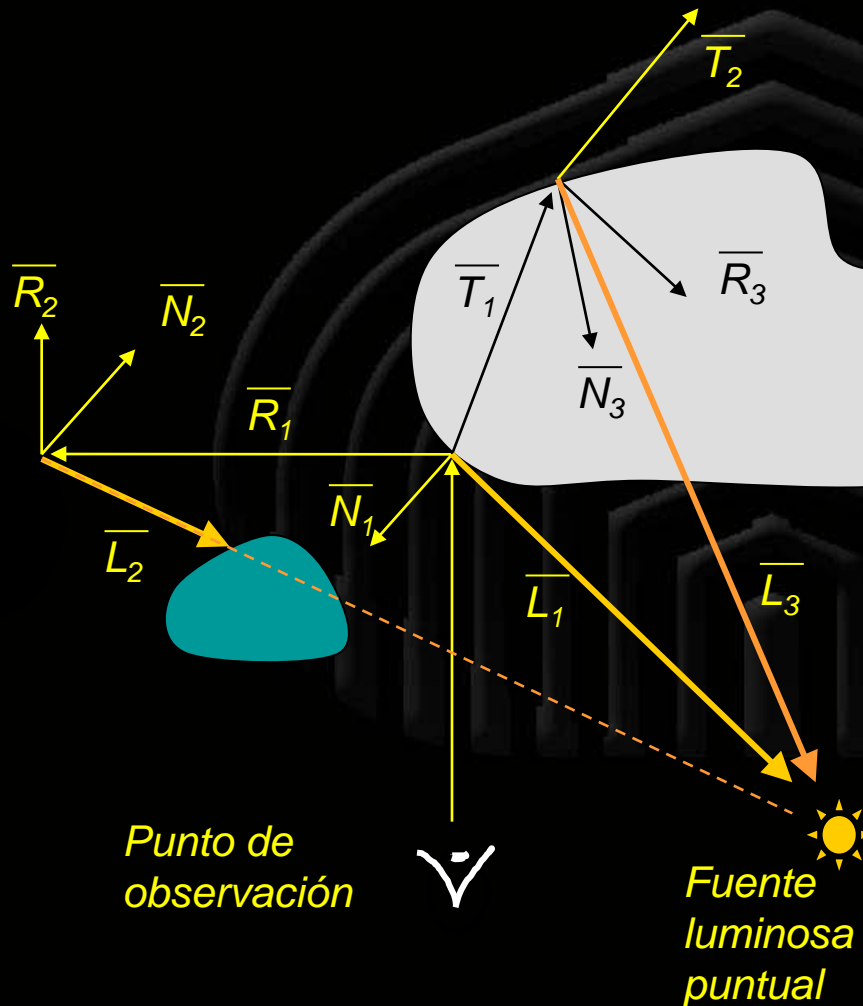


Aquí se ve un rayo v que sale del observador e incide en una superficie.

Además de los rayos de sombra L , se generan rayos de reflexión R y de refracción T (si el objeto es transparente). Se llaman rayos secundarios.

Rayos primarios son los que parten del observador.

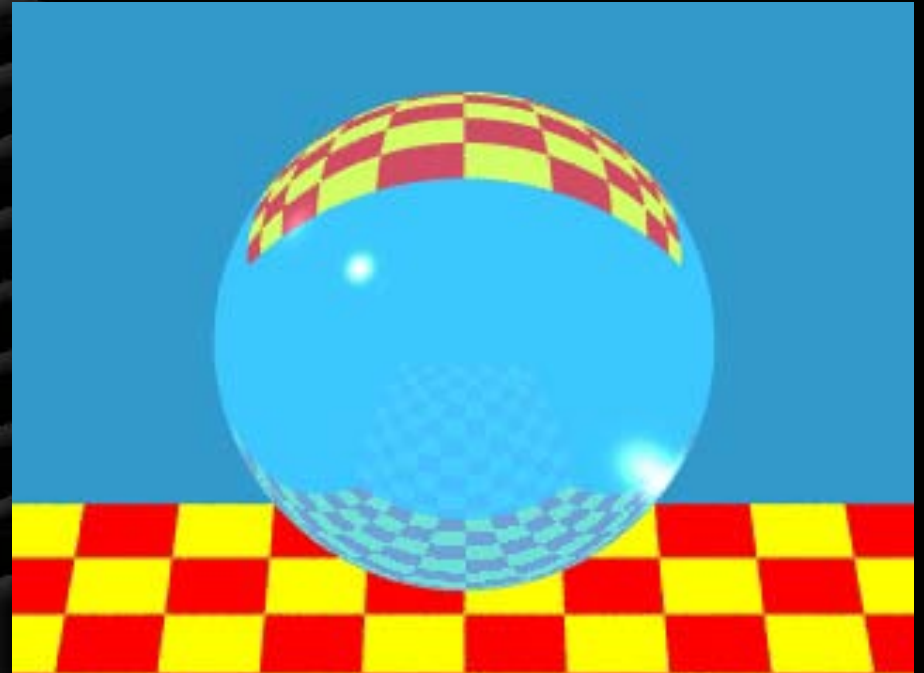
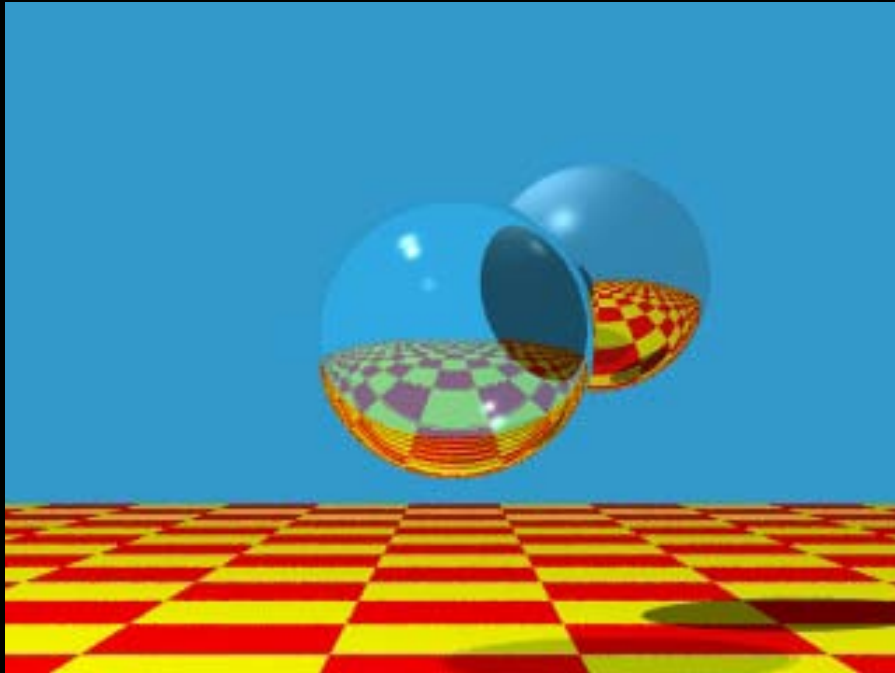
Traza de rayos recursiva (Whitted)



Aquí se ven los primeros pasos de la recursión del algoritmo de Whitted. El rayo que sale del observador, incide en un punto de una superficie. De él salen 3 rayos L_1, R_1 y T_1 , estos últimos inciden nuevamente en 2 superficies.


El objetivo es saber qué ve el observador en ese punto. La luz que le llega al observador se compone de la proveniente directamente de la fuente luminosa (y reflejada en la superficie), así como de la reflejada de forma especular por la superficie y de la refractada, las cuales a su vez vienen de otras reflexiones y refracciones.

Traza de rayos recursiva (Whitted)



Aquí se aprecian imágenes que para ser generadas se ha precisado realizar árboles de profundidad de un nivel relativamente hondo (4 o 5 niveles).

Traza de rayos recursiva (Whitted)


$$I_{\lambda} = I_{a\lambda} k_a O_{d\lambda} + \sum_{1 \leq i \leq m} S_i f_{att_i} I_{p\lambda_i} \left[k_d O_{d\lambda} (\bar{N} \cdot \bar{L}_i) + k_s O_{s\lambda} (\bar{R}_i \cdot \bar{V})^n \right] + k_s I_{r\lambda} + k_t I_{t\lambda}$$

Dos términos nuevos agregados a la ecuación de iluminación

$I_{r\lambda}$ Intensidad del rayo reflejado.

$I_{t\lambda}$ Intensidad del rayo transmitido refractado.

k_t coeficiente de transmisión (entre 0 y 1).

$I_{r\lambda}$ e $I_{t\lambda}$ se calculan recursivamente y se multiplican por el inverso de la distancia.

Traza de rayos recursiva (Whitted)

seudocódigo

(1)

A continuación veremos el pseudocódigo del algoritmo de Whitted. Es relativamente fácil de implementar y los resultados son bastante buenos aunque no válidos físicamente.

Seleccionar el centro de proyección y la ventana en el plano de vista;

for (*cada línea de barrido en la imagen*) {

for (*cada pixel en la línea de barrido*) {

determinar rayo por centro de proyección y pixel;

 pixel=traza_RR(rayo, 1);

 }

}

Traza de rayos recursiva (Whitted)

seudocódigo

(2)

/ Intersecar rayo con los objetos y calcular la sombra en la intersección más cercana. */*

/ La profundidad es la profundidad actual en el árbol de rayos */*

color_RR **traza**_RR (rayo_RR, **int** profundidad)

{

determinar la intersección más cercana de rayo con un objeto;

if (*Hay objeto intersecado*) {

calcular normal en la intersección;

return **sombra**_RR(*obj. intersecado más cercano, rayo, intersección, normal, profundidad*);

}

else

return VALOR_FONDO

}

Traza de rayos recursiva (Whitted)

seudocódigo

(3)




```
/* Calcular sombra en un punto, con rayos para sombras, reflex. y refrac. */  
color_RR sombra_RR (objeto, rayo, punto, normal, int profundidad)  
{  
    color = término del ambiente;  
    for (cada luz) {  
        rayo_s = rayo desde punto a la luz;  
        if (el producto punto de normal y direc. de luz es positivo) {  
            Calcular cuánta luz es bloqueada por sup. Opacas y transp., y  
            usarlo para escalar los términos difusos y especulares antes de  
            añadirlos a color;  
        }  
    }  
}
```

...

Traza de rayos recursiva (Whitted)

seudocódigo

(4)

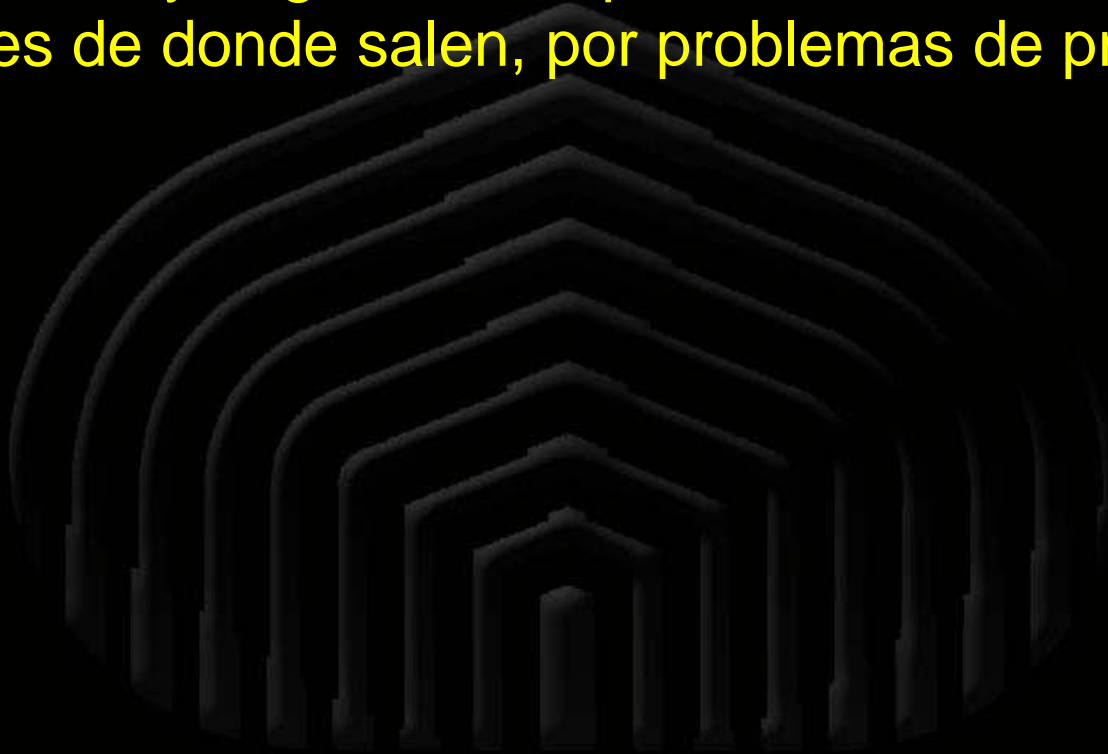


```
if (profundidad < profundidad_max) { /*Regresar si la prof. es excesiva */
    if (objeto es reflejante) {
        rayo_r = rayo en la dirección de reflexión desde punto;
        color_r = traza_RR (rayo_r, profundidad + 1);
        escalar color_r por el coeficiente especular y añadir a color; }
    if (objeto es transparente) {
        rayo_t = rayo en la dirección de refracción desde punto;
        if (no ocurre la reflexión interna total) {
            color_t = traza_RR (rayo_t, profundidad + 1);
            escalar color_t por el coeficiente de transmisión y
                añadir a color;
        }
    }
}
return color; /* Devolver color del rayo. */
}
```

Traza de rayos recursiva (Whitted)



Problema: Los rayos generados pueden intersecar las mismas superficies de donde salen, por problemas de precisión.



Métodos de Radiosidad (Radiosity)



Ray tracing contempla un término de iluminación ambiental no direccional. Esto es para contemplar otras contribuciones luminosas.

Hay métodos basados en la termodinámica, (emisión y reflexión de radiación) que hace innecesaria la iluminación ambiental.

Radiosidad: tasa con la que la energía parte de una superficie. Se compone de la tasa de energía emitida + la reflejada.

$$\text{Energía/unidad de tiempo/unidad de área} = \text{W/m}^2$$

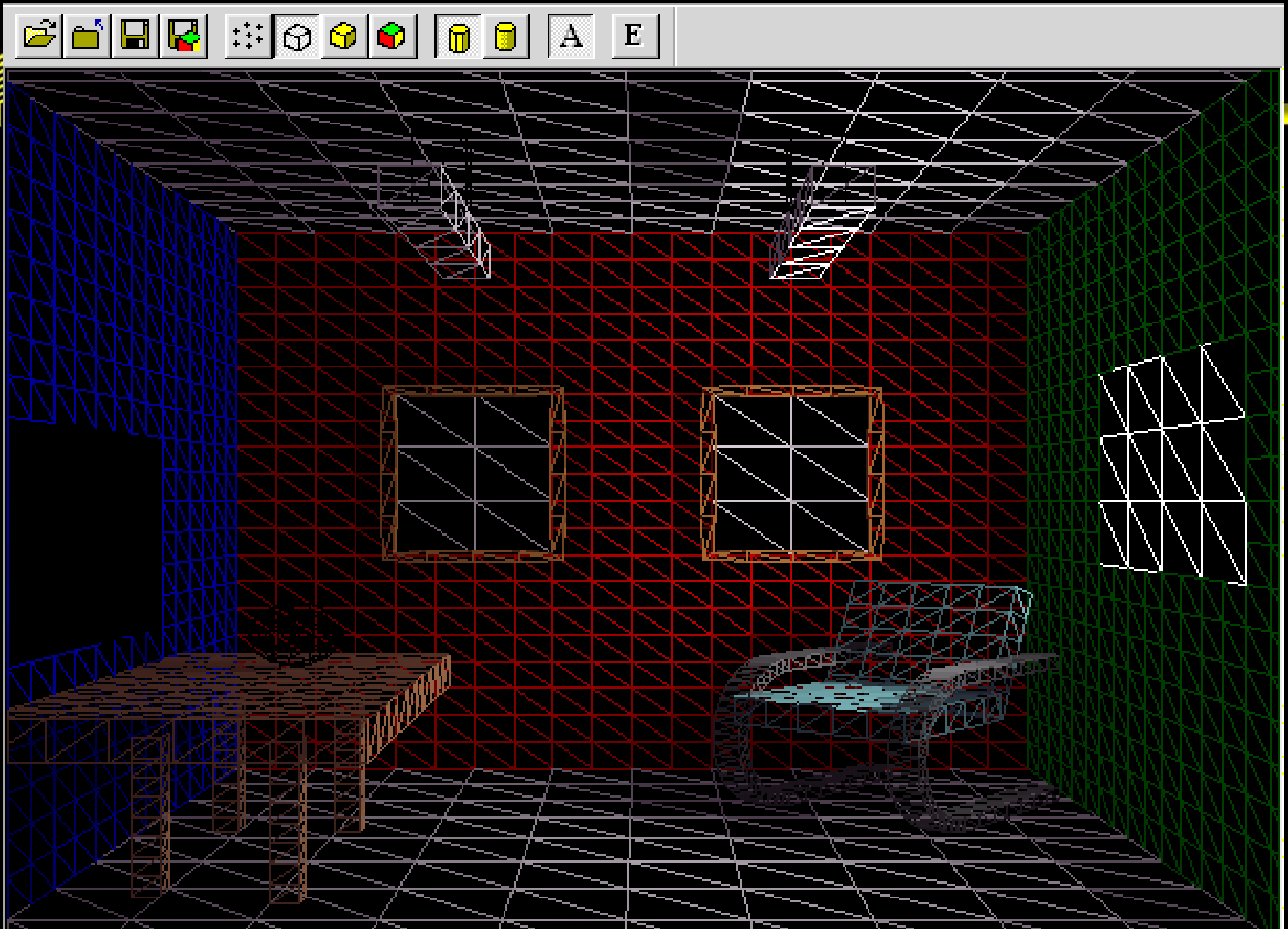


Métodos de Radiosidad

Las interacciones entre superficies se calculan de forma independiente de la vista.

Luego de este cálculo, se generan las vistas, solo determinando cuáles son las superficies visibles y el sombreado por interpolación (por ejemplo, con Gouraud).

Métodos de Radiosidad



Ready

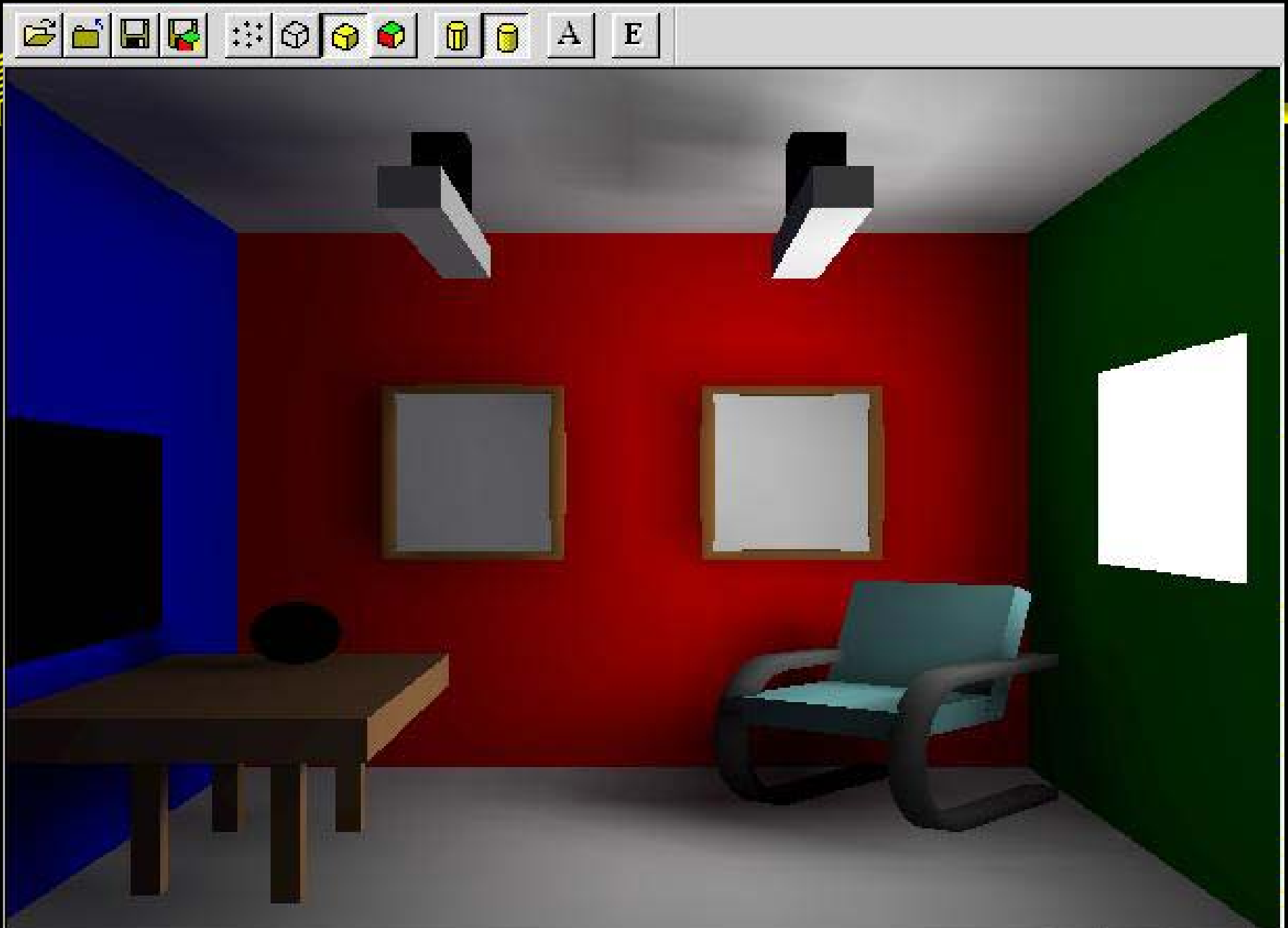
NUM

Métodos de Radiosidad



NUM

Métodos de Radiosidad



Ready

NUM

Métodos de Radiosidad

- Todas las superficies pueden emitir y reflejar luz.
- Por tanto, las fuentes luminosas son áreas.
- Al ambiente se lo divide en un número finito de parches discretos.
- En cada parche, la luz es emitida y reflejada de forma uniforme.

Ecuación de Radiosidad

$$B_i = E_i + \rho_i \sum_{1 \leq j \leq n} B_j F_{j-i} (A_j / A_i)$$

B_i = radiosidad de los parche i

E_i = tasa con que el parche i emite luz.

ρ_i = reflectividad del parche i (sin dimensiones).

F_{j-i} = Factor de forma, fracción de energía que va del parche j al i (sin dimensiones)

A_i = Área del parche i

Ecuación de Radiosidad

$$B_i = E_i + \rho_i \sum_{1 \leq j \leq n} B_j F_{j-i} (A_j / A_i)$$

- La energía que parte de un parche = suma de la energía emitida + la reflejada.
- Luz reflejada = luz incidente x reflectividad ρ .
- Luz incidente = Suma de luz que parte de toda el área de cada parche en el ambiente, escalada a la fracción de luz que llega a un área unidad del parche receptor.
- $B_j F_{j-i}$ = cantidad de luz que parte de un área unidad de A_j y llega a toda el área A_i .
- $B_j F_{j-i} (A_j / A_i)$ = cantidad de luz que parte de toda el área A_j y llega al área unidad de A_i .

Ecuación de Radiosidad

$$B_i = E_i + \rho_i \sum_{1 \leq j \leq n} B_j F_{j-i} (A_j/A_i)$$

En los ambientes difusos, los factores de forma cumplen una regla útil:

$$A_i F_{i-j} = A_j F_{j-i} \quad \Rightarrow \quad F_{i-j} = A_j F_{j-i} / A_i$$

Por tanto, la ecuación anterior se simplifica:

$$B_i = E_i + \rho_i \sum_{1 \leq j \leq n} F_{i-j} B_j$$

O, de forma equivalente:

$$B_i - \rho_i \sum_{1 \leq j \leq n} F_{i-j} B_j = E_i$$

Ecuación de Radiosidad

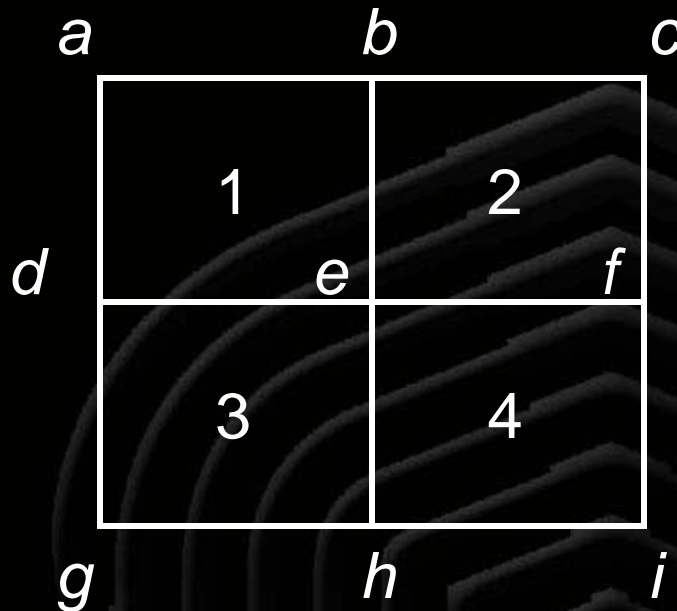
$$B_i - \rho_i \sum_{1 \leq j \leq n} F_{i-j} B_j = E_i$$

Esto permite generar un sistema de ecuaciones lineales:

$$\begin{array}{cccccc} 1 - \rho_1 F_{1-1} & -\rho_1 F_{1-2} & \dots & -\rho_1 F_{1-n} & B_1 & E_1 \\ -\rho_2 F_{2-1} & 1 - \rho_2 F_{2-2} & \dots & -\rho_2 F_{2-n} & B_2 & E_2 \\ \cdot & \cdot & & \cdot & \cdot & \cdot \\ \cdot & \cdot & & \cdot & \cdot & \cdot \\ \cdot & \cdot & & \cdot & \cdot & \cdot \\ -\rho_n F_{n-1} & -\rho_n F_{n-2} & \dots & -\rho_n F_{n-n} & B_n & E_n \end{array} * =$$

Se puede resolver por Gauss-Seidel

Determinación de la radiosidad de los vértices



A partir de la radiosidad de los parches, se puede inferir la radiosidad de los vértices según las ecuaciones de abajo.

Una vez obtenida la radiosidad de los vértices, se puede aplicar Gouraud o Phong para que no se note el facetado provocado por los parches.

$$B_e = (B_1 + B_2 + B_3 + B_4)/4$$

$$(B_b + B_e)/2 = (B_1 + B_2)/2 \quad \Rightarrow \quad B_b = (B_1 + B_2 - B_e)$$

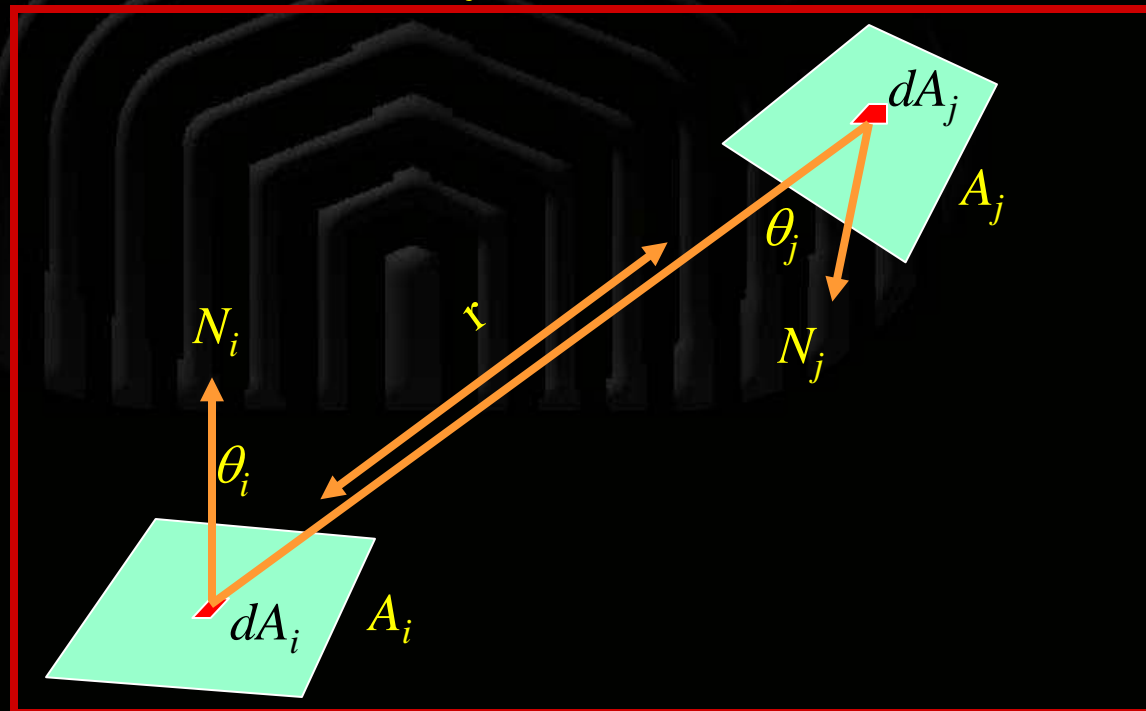
$$(B_a + B_e)/2 = B_1 \quad \Rightarrow \quad B_a = 2B_1 - B_e$$

Cálculo de los Factores de Forma

El factor de forma de un área diferencial dA_i a un área diferencial dA_j es:

$$dF_{di-dj} = \cos\theta_i \cos\theta_j H_{ij} dA_j / \pi r^2$$

H_{ij} es 1 o 0 dependiendo de si dA_j es visible desde dA_i



Cálculo de los Factores de Forma

$$dF_{di-dj} = \cos\theta_i \cos\theta_j H_{ij} dA_j / \pi r^2$$

$$F_{di-j} = \int_{A_j} \cos\theta_i \cos\theta_j H_{ij} / \pi r^2 dA_j$$

$$F_{i-j} = (1/A_i) \int_{A_i} \int_{A_j} \cos\theta_i \cos\theta_j H_{ij} / \pi r^2 dA_j dA_i$$

Suponemos que el punto central de un parche tipifica los demás puntos del parche.

=>

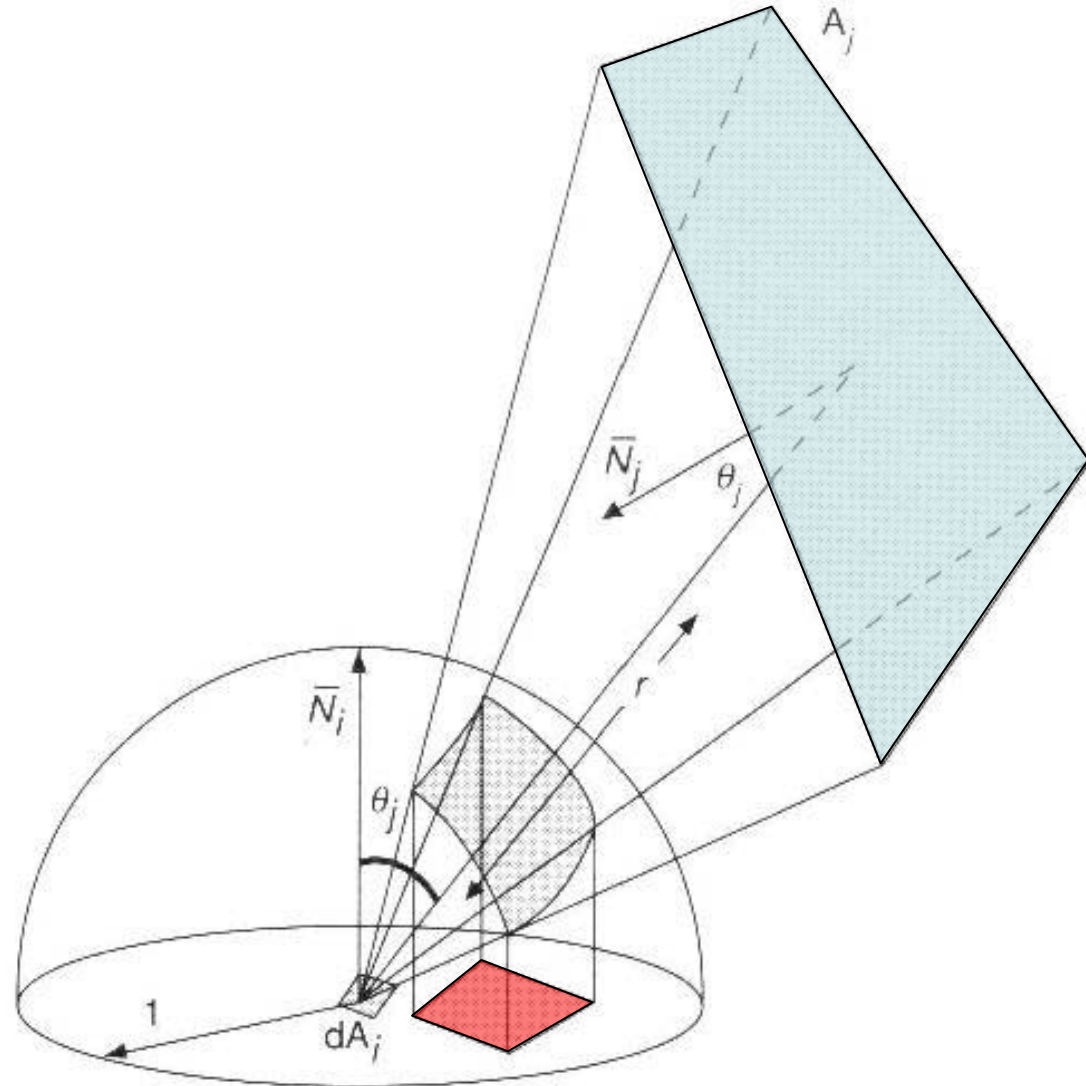
F_{i-j} se puede aproximar con F_{di-j} calculado para un dA_j en el centro del parche i .

Cálculo de los Factores de Forma

Calcular F_{di-j} equivale

a:

- 1) proyectar el área visible de A_j desde dA_i sobre el hemisferio unidad.
- 2) Hacer una nueva proyección sobre el círculo unidad.
- 3) Dividir el área hallada por el área del círculo.

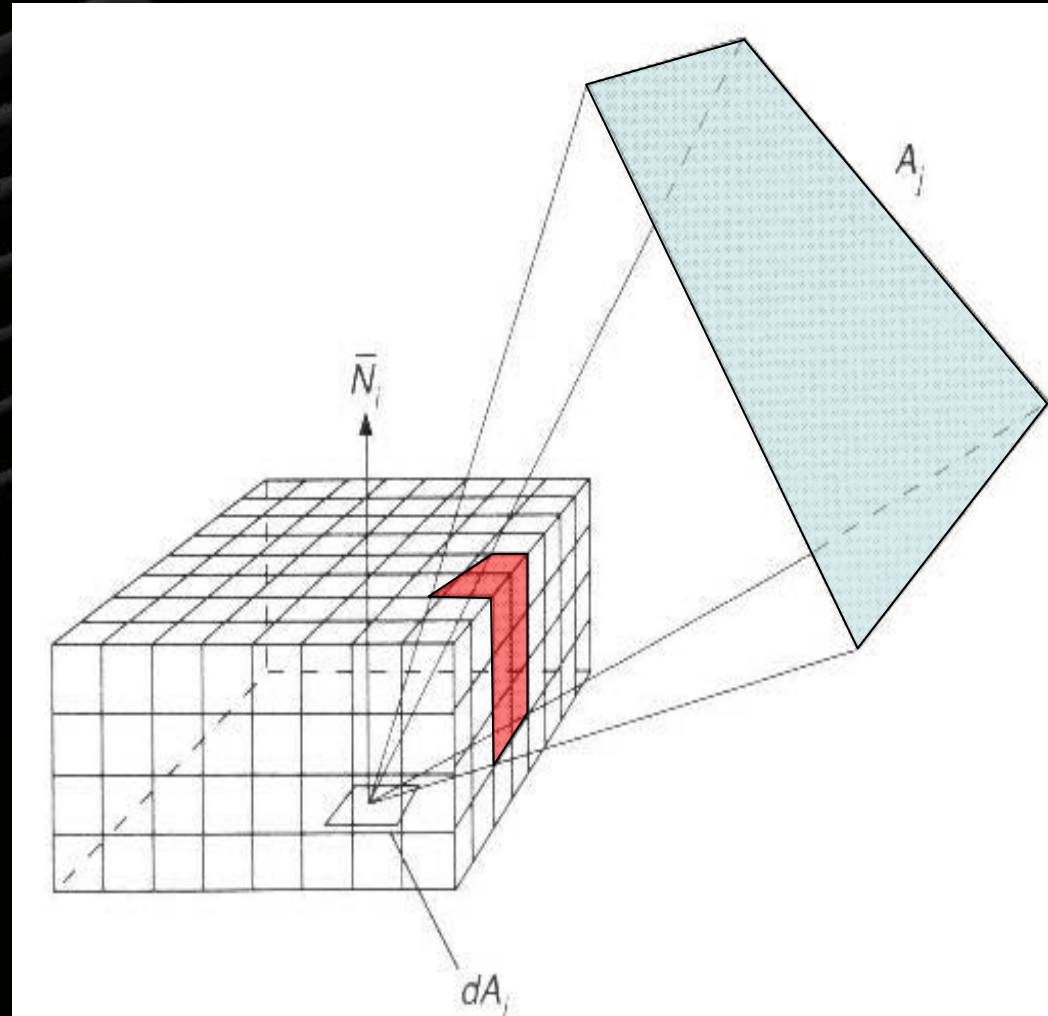


Cálculo de los Factores de Forma

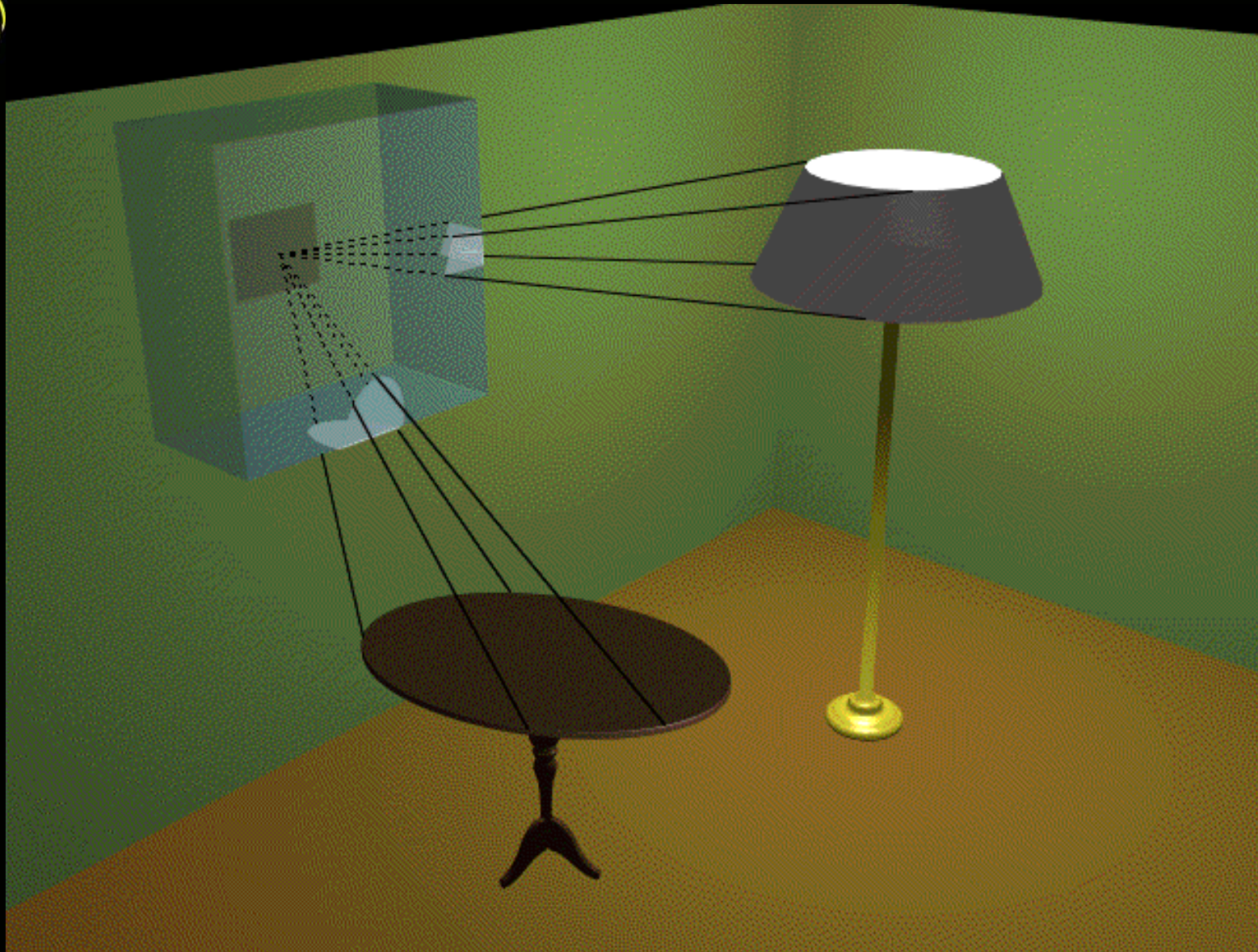


Calcular F_{di-j} equivale a:

- 1) proyectar el área visible de A_j desde dA_i sobre un **hemicubo** de centro en dA_i .
- 2) Cada uno de los parches se proyectan sobre la cara apropiada del hemicubo.
- 3) Cada celda del hemicubo tiene un F_{deF} propio, debido a su posición. F_{di-j} se calcula sumando los F_{deF} de todas las celdas que contengan el identificador del parche j .



Cálculo de los *Factores de Forma*

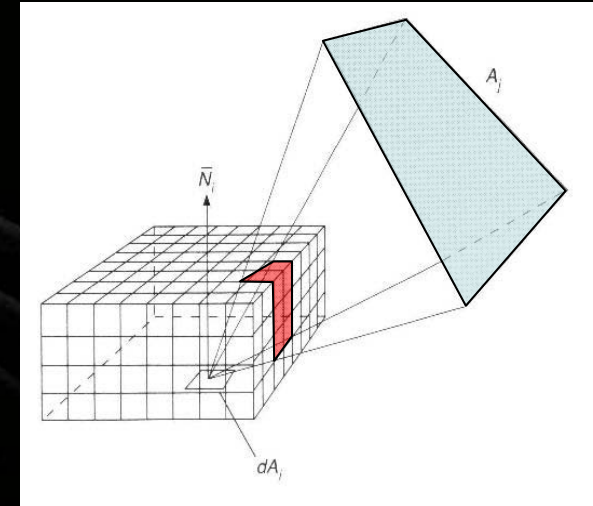


Cálculo de los *Factores de Forma*

Este algoritmo se puede realizar aplicando el algoritmo de z-buffer a cada lado del hemisucubo, y registrando el código del parche en lugar del color.

Se puede aprovechar el hardware existente para la memoria de profundidad z.

Pueden haber artefactos de discretización, por usar operaciones de precisión de la imagen.



Soluciones parciales = Refinamiento progresivo

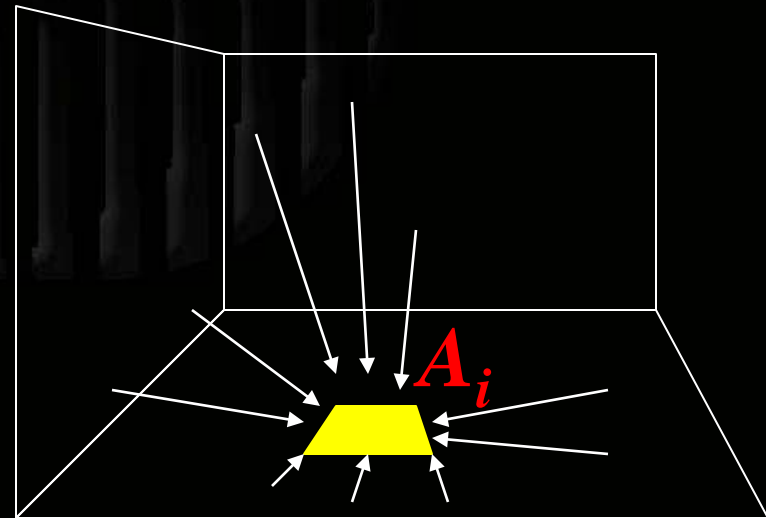
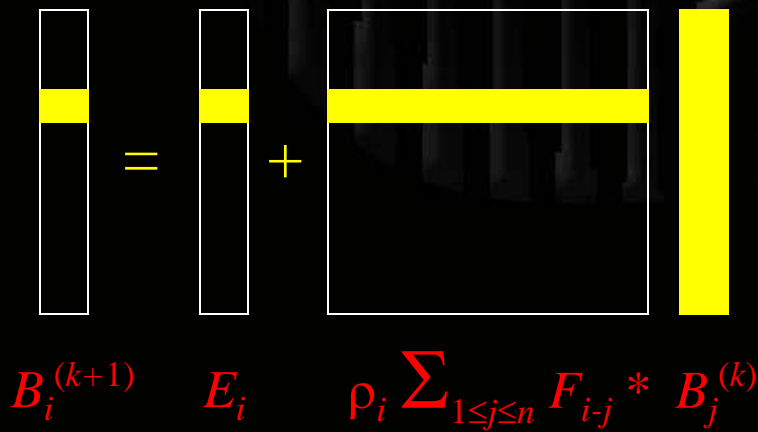


En el método anterior, hay n filas del tipo:

$$B_i^{(k+1)} = E_i + \rho_i \sum_{1 \leq j \leq n} F_{i-j} B_j^{(k)}$$

se **reune** la energía del ambiente sobre el parche i .

Por cada fila i procesada, se obtiene un nuevo valor para el parche i .



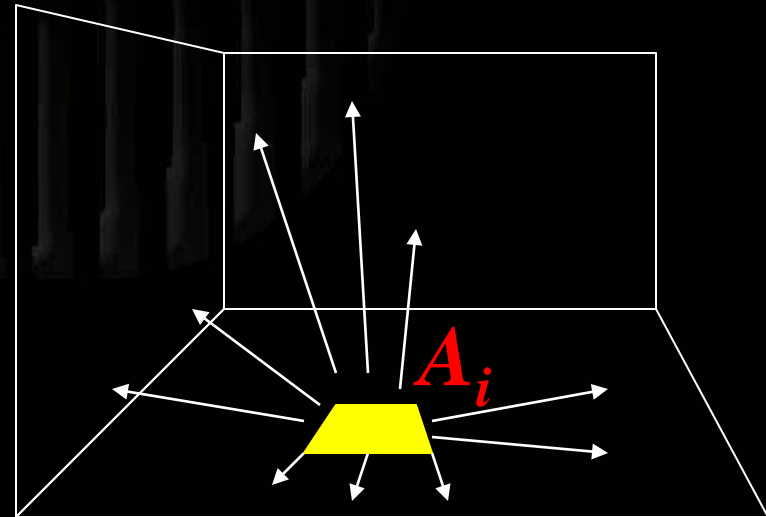
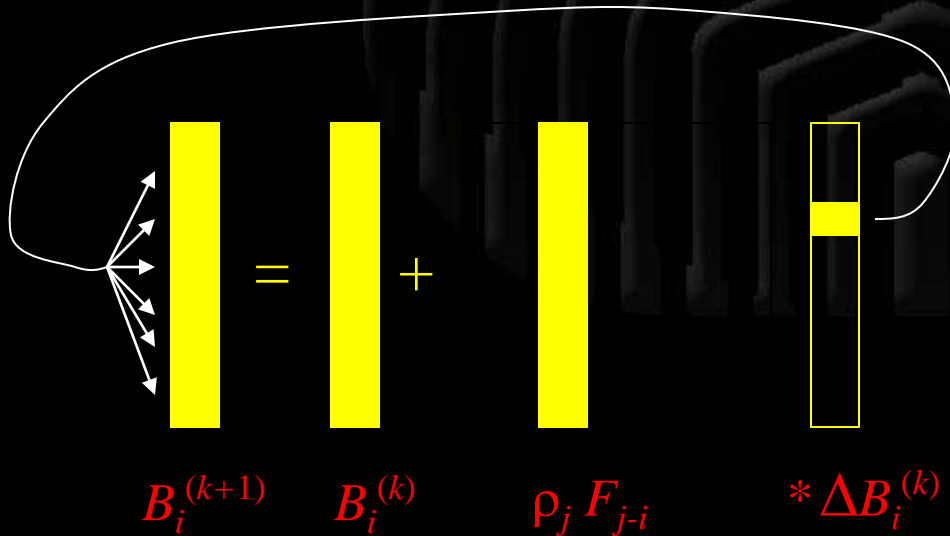
Soluciones parciales = Refinamiento progresivo



En este nuevo método, se **irradia** la energía del parche i a todo el ambiente.

$$\text{For } (j = 1:n) \left\{ B_j^{(k+1)} = B_j^{(k)} + \rho_j F_{j-i} \Delta B_i \right\}$$

Por cada columna i procesada, se obtiene un nuevo valor para todos los parches.



Soluciones parciales = Refinamiento progresivo



PROGRESSIVE SOLUTION

The above images show increasing levels of global diffuse illumination. From left to right: 0 bounces, 1 bounce, 3 bounces.

Soluciones parciales = Refinamiento progresivo



- El cálculo de F_{j-i} precisa de un hemicubo diferente por j . Esto se puede simplificar, haciendo:

$$\text{For } (j = 1:n) \quad \{ B_j^{(k+1)} = B_j^{(k)} + \rho_j F_{i-j}(A_i/A_j) \Delta B_i \}$$

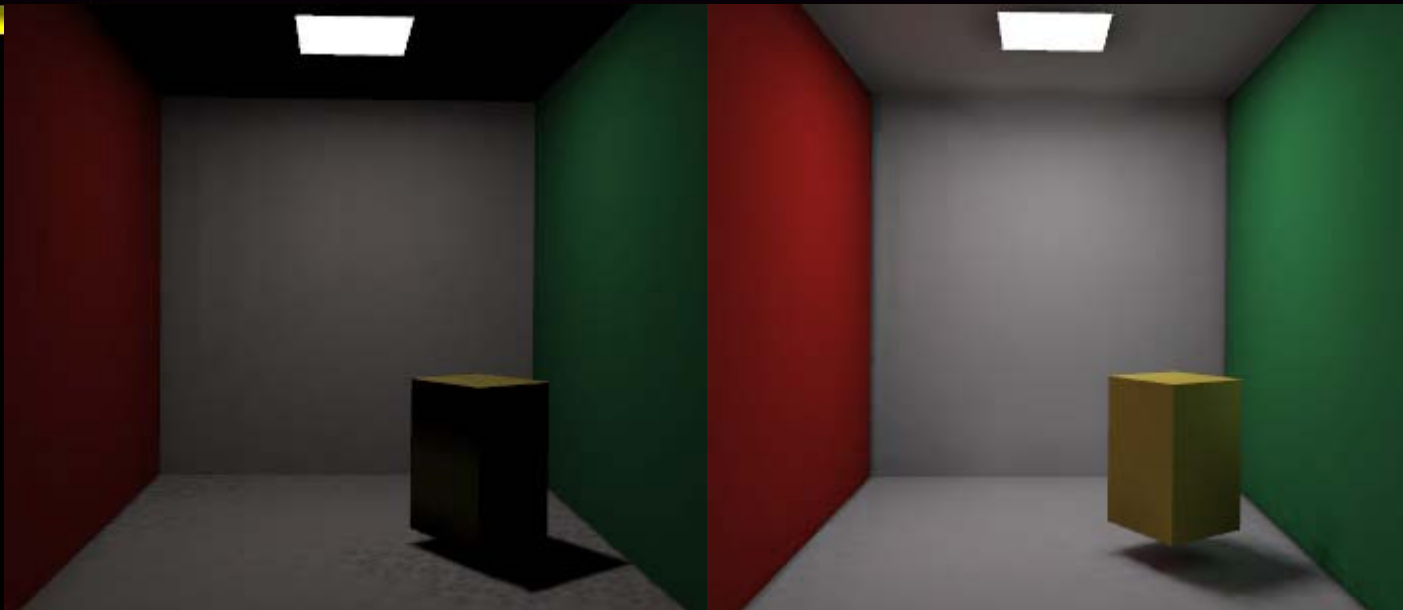
Este nuevo cálculo solo precisa un hemicubo centrado en el parche A_j

- Cuando se irradia por primera vez del parche A_j , este irradia $E_j = \Delta B_j$
- Las siguientes veces que se irradie del parche A_j , solo se considera el incremento ΔB_j logrado desde la última vez que se irradizó.
- Conviene elegir los parches que al irradiar den el mayor impacto, los que tengan más energía a irradiar.

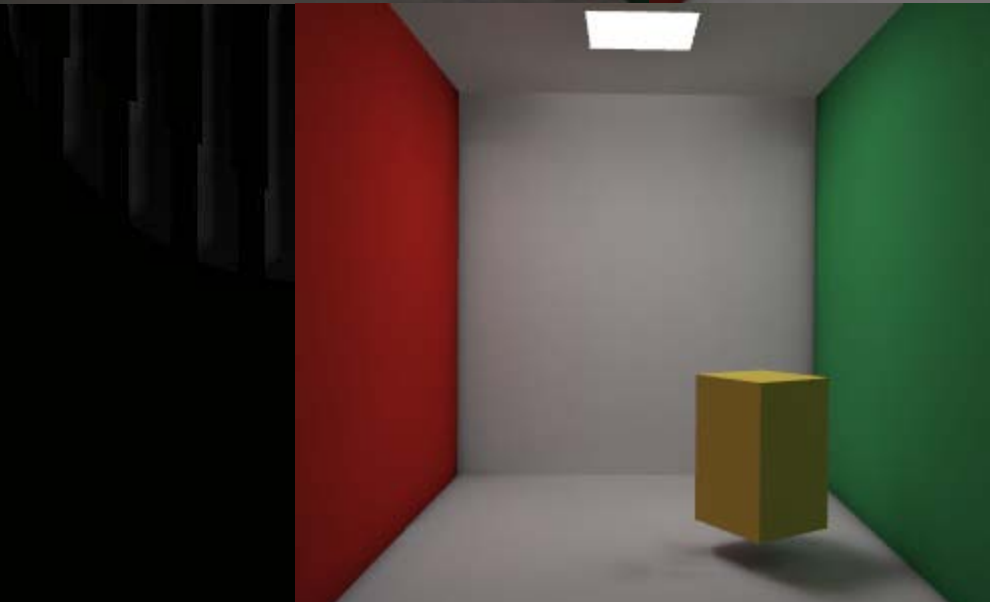
Radiosity



1



32



320

