

Gràfics i Visualització de Dades

Tema 5: Il·luminació (part 1)

Anna Puig

NOTA: Aquestes transparències es corresponen a les explicacions del tema 5 del llibre de referència bàsica

[Angel2011] Edward Angel, Dave Shreiner, **Interactive Computer Graphics: A Top-Down Approach with Shader-Based OpenGL, 6/E**, ISBN-10: 0132545233. ISBN-13: 9780132545235, Addison-Wesley, 2011

Objectius

- En aquest tema, s'analitzarà com **il·luminar** els objectes per a que semblin tri-dimensionals
- S'introduiran els **tipus d'interacció** entre llums i materials
- S'explicarà com construir un **model local de càcul d'il·luminació** (el model de Phong-Blinn) que pot ser utilitzat directament en el hardware gràfic
- S'analitzaran diferents **implementacions** en la CPU, en el vertex-shader i en el fragment-shader que codifiquen el shading poligonal.

Índex

5.1. Introducció

5.2. Interacció de les llums i els objectes

5.3. Llums

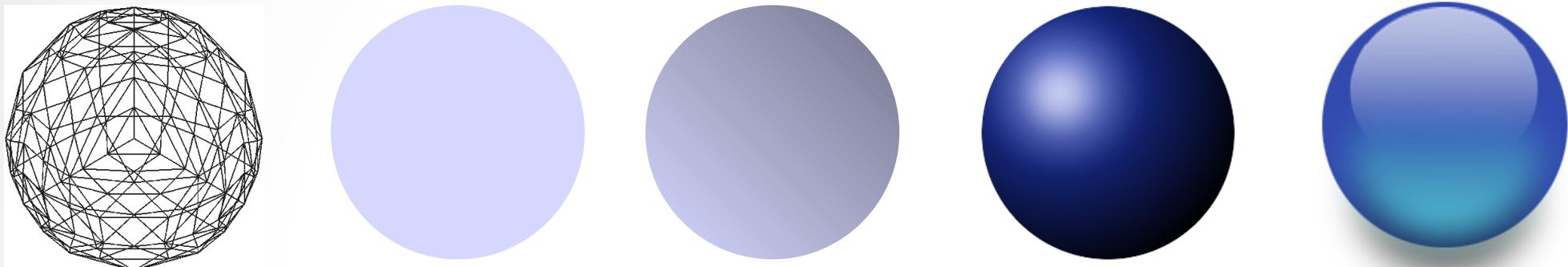
5.4. Materials

5.5. El model de Phong

5.6. Shading poligonal

3.1. Introducció

Suposeu que construïm una esfera amb molts polígons:



- Per què les tres darreres semblen més una esfera real?
 - Les **interaccions** entre les **llums** i el **material** de l'esfera fa que cada punt que veu l'observador tingui un color diferent.
 - Per això és necessari considerar:
 - Els punts de llum
 - Les propietats del material de l'objecte
 - La posició de l'observador
 - L'orientació de la superfície

Què és la llum? (1)

La **llum** es pot considerar com petits paquets d'energia (**fotons**)

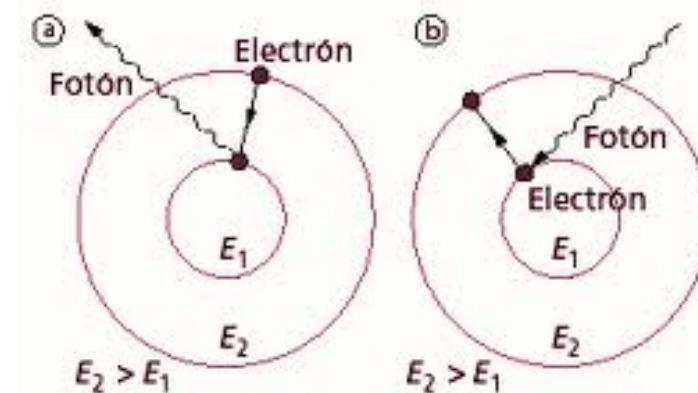
- Cada àtom té un nucli i electrons circulant a diferents òrbites
- Quan un electró salta d'òrbita alta a una baixa s'emet energia (**Llei de Plank**)
- Quan un fotó col·lisiona amb l'àtom del costat i un electró puja d'òrbita, s'absorbeix la llum.



<http://www.overclock.net/art-graphics/251218-kerkythea-shaded-lightsource-test.html>

La **longitud d'ona (color)** de la llum emesa o absorbida correspon al canvi d'orbital.

La **natura** dels àtoms i els seus electrons (**material**) controla el tipus d'emissió de llum



Què és la llum? (1)

Els diferents nivells d'energia dels electrons dels diferents materials permeten reflectir diferents longituds d'ona:

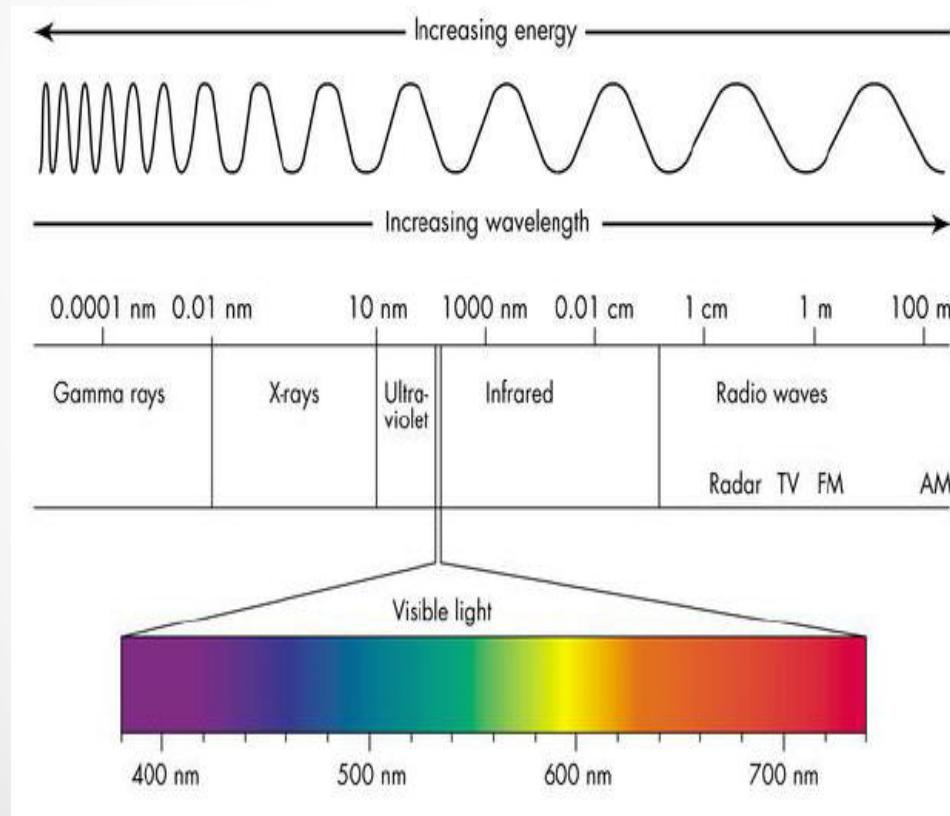
- Els **metalls** perden electrons fàcilment ja que per ells circulen relativament lliures i ocupen diferents nivells d'energia. Son més reflectants.
- Els materials **aïllants** tenen els electrons més estables i poden absorbir més llum sense canviar les òrbites dels seus



Què és la llum? (2)

La **llum** també es pot considerar com a un **conjunt d'ones**:

- La longitud d'ona és la distància entre un màxim i un altre de l'ona
- La longitud d'ona visible va de 400nm a 700nm aproximadament



La llum visible té una longitud d'ona a 400nm – 700nm, una porció petita de l'espectre de la llum

Només ens interessa en l'espectre visible i definir/ agrupar rangs amplis de longitud d'ona, aproximadament:

Vermell (R), Verd (G) I Blau (B)

Índex

5.1. Introducció

5.2. Interacció de les llums i els objectes

5.3. Llums

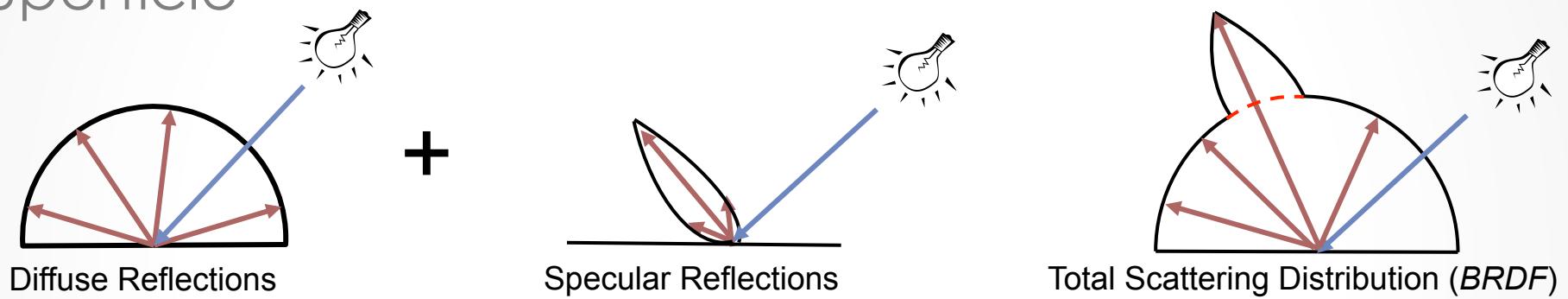
5.4. Materials

5.5. El model de Phong

5.6. Shading poligonal

5.2. Interacció de la llum i els materials dels objectes

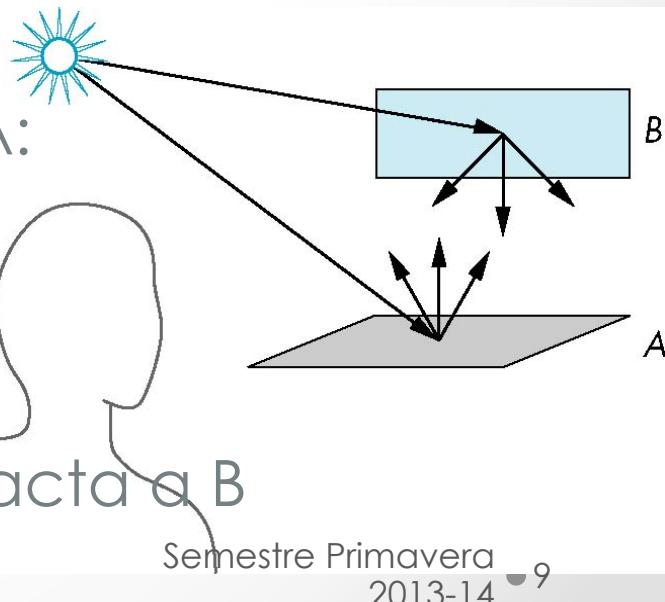
Comportament de la llum quan impacta en una superfície



Scattering:

Quan la llum impacta en la superfície A:

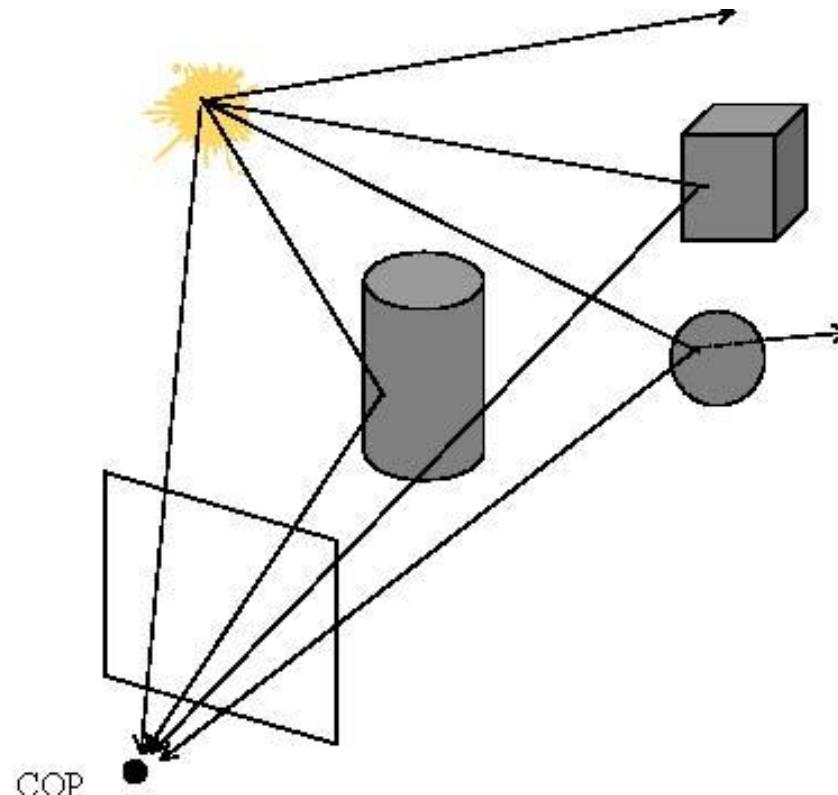
- Una part rebota (scattering)
- Una part s'absorbeix
- Una part de la llum rebotada impacta a B



5.2. Interacció de la llum i els materials dels objectes

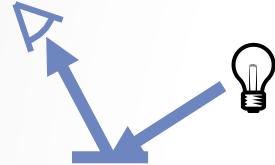
Com es calcula?

- Substitució de l'observador pel pla de projecció

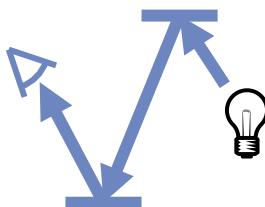


5.2. Interacció de la llum i els materials dels objectes

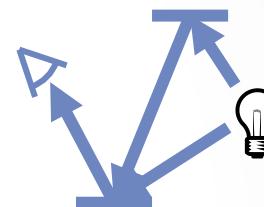
- Tipus d'il·luminacions calculades:



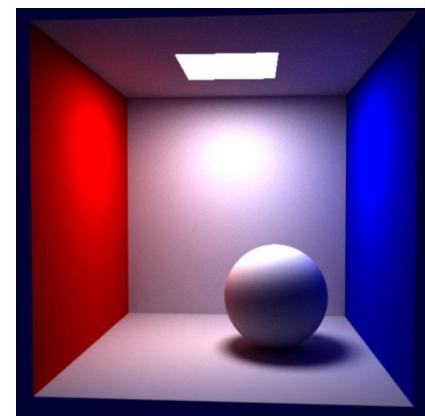
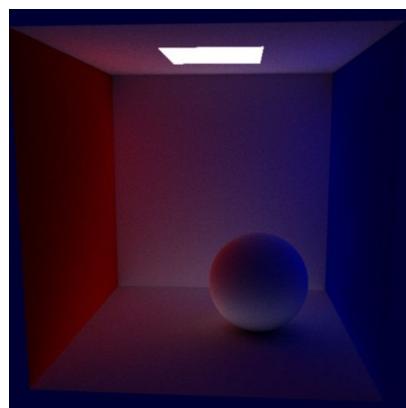
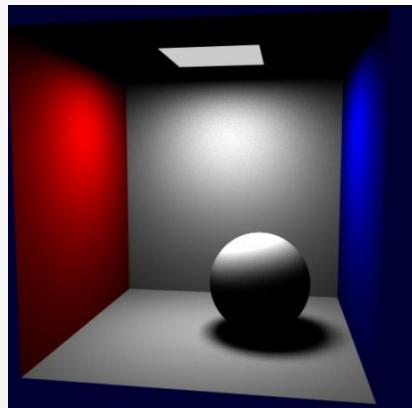
Il·luminació directa



Il·luminació indirecta

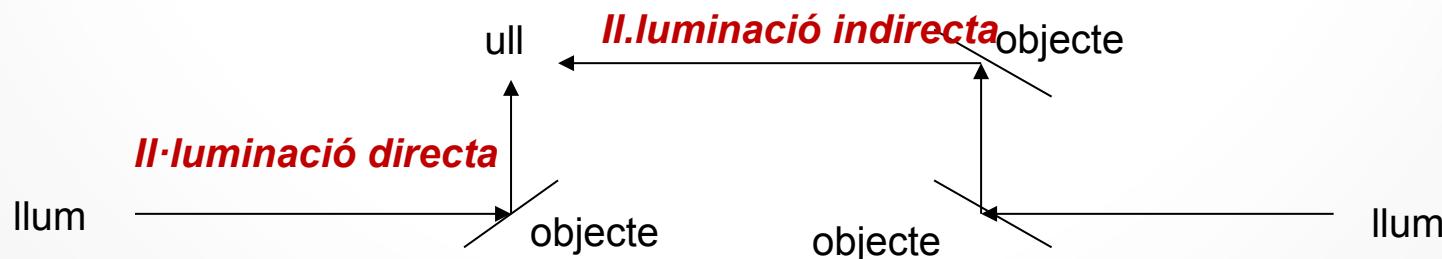


Il·luminació global



5.2. Interacció de la llum i els materials dels objectes

- Llums iombres
 - La major part de la llum reflectida per una superfície prové **directament** de les fonts de llum.
 - Quan la llum que prové d'una font de llum es veu ocluïda per una altra superfície es forma una **ombra**.
 - En el pipeline clàssic no es tenen en compte lesombres.
- Reflexions entre objectes (il·luminació **indirecte**)
 - Quan la llum rebota en altres objectes i impacte a una altra superfície, afegeix llum a la il·luminació directe.



5.2. Interacció de la llum i els materials dels objectes



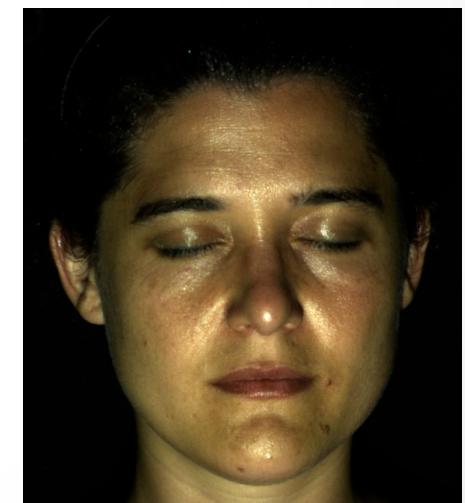
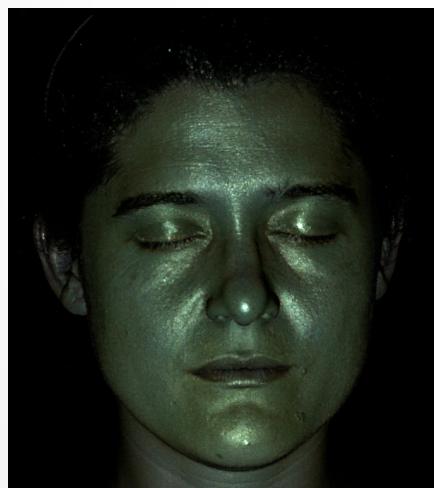
Il·luminació directa



Il·luminació indirecta



Il·luminació global



S. Nayar, G. Krishnan, M. Grossberg, and R. Raskar. "Fast Separation of Direct and Global Components of a Scene using High Frequency Illumination," SIGGRAPH '06

<http://www.cs.columbia.edu/CAVE/projects/separation/videos/Separation.mov>

5.2. Interacció de la llum i els materials dels objectes

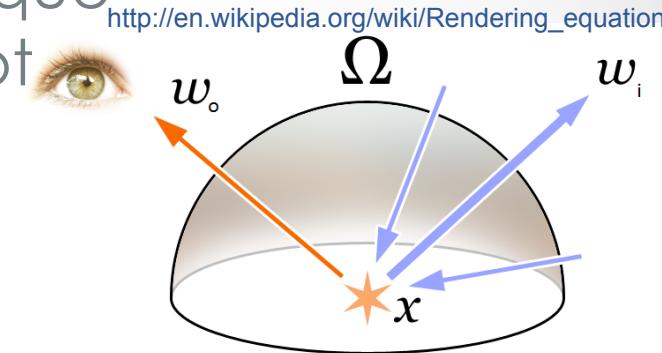
Equació del rendering (BRDF): equació que serveix per a calcular la il·luminació a tot punt de l'escena

El scattering i l'absorció infinit de la llum es poden descriure per l'equació del rendering:

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

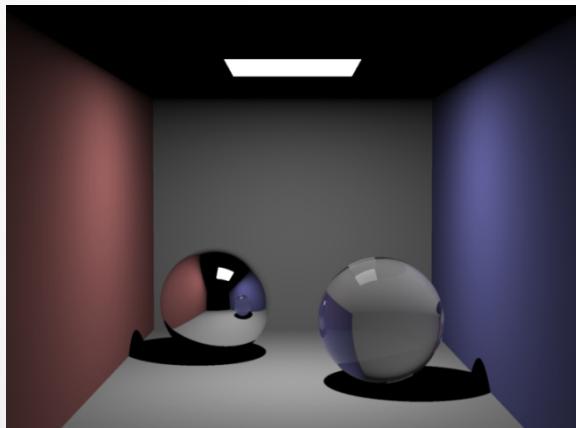
- No es pot resoldre en general
- Normalment es fan simplificacions per a poder resoldre-la:
 - El mètode de RayTracing és el cas especial per a superfícies perfectament reflectants
- L'equació del rendering és **global** i inclou:
 - Ombres, Scattering múltiple entre objectes

Kajiya, James T. (1986) ["The rendering equation"](#), Siggraph 1986: 143

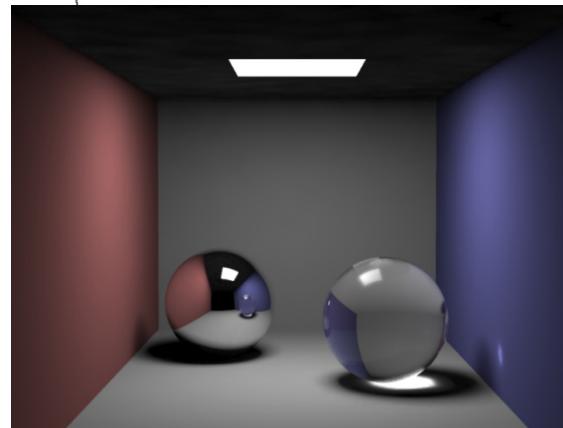


5.2. Interacció de la llum i els materials dels objectes

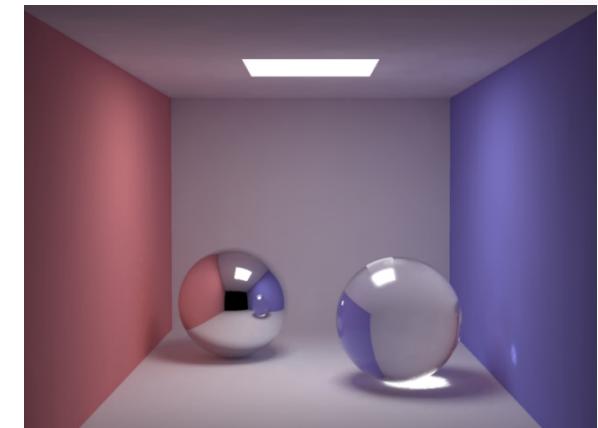
Efectes globals



Direct illumination + specular reflection
Ray trace



+ soft shadows and caustics
Ray trace + caustic photon map



+ diffuse reflection (color blending)
Ray trace + caustic and diffuse photon maps

<http://graphics.ucsd.edu/~henrik/images/global.html>

5.2. Interacció de la llum i els materials dels objectes

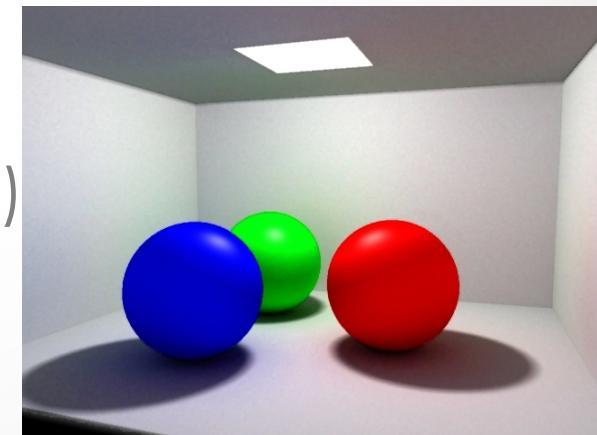
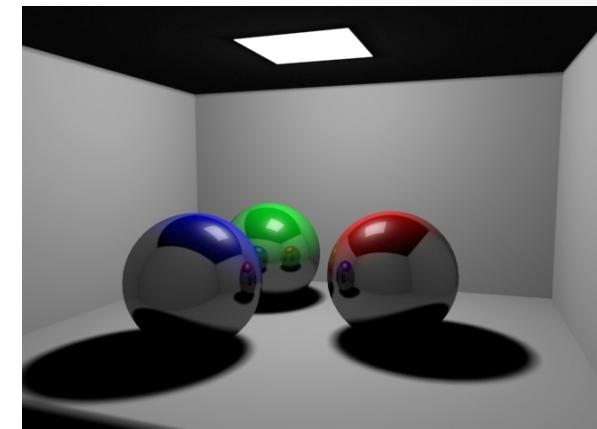
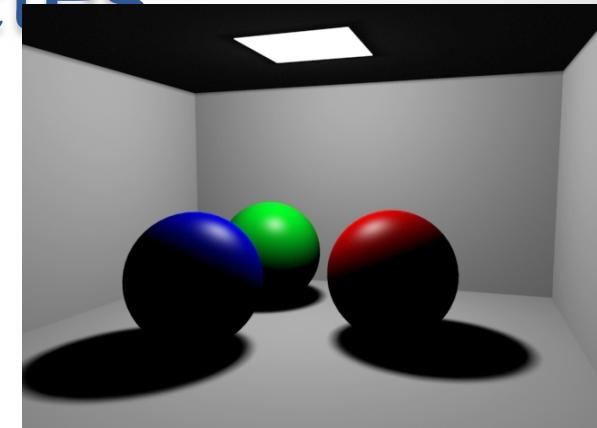
Rendering local: té en compte la llum directa de les fonts de llum i modela alguns efectes indirectes:

- Ràpid i fàcil d'adaptar al pipeline gràfic
- Menys realisme

Rendering global: té en compte la il·luminació amb totes les superfícies i les llums de l'escena.

- Permet reflexions entre objectes, refraccions, càustiques, efectes de mitjans participatius (boira, aigua, etc.)

http://www.spot3d.com/vray/help/200R1/examples_GI.htm



Índex

5.1. Introducció

5.2. Interacció de les llums i els objectes

5.3. Llums

5.4. Materials

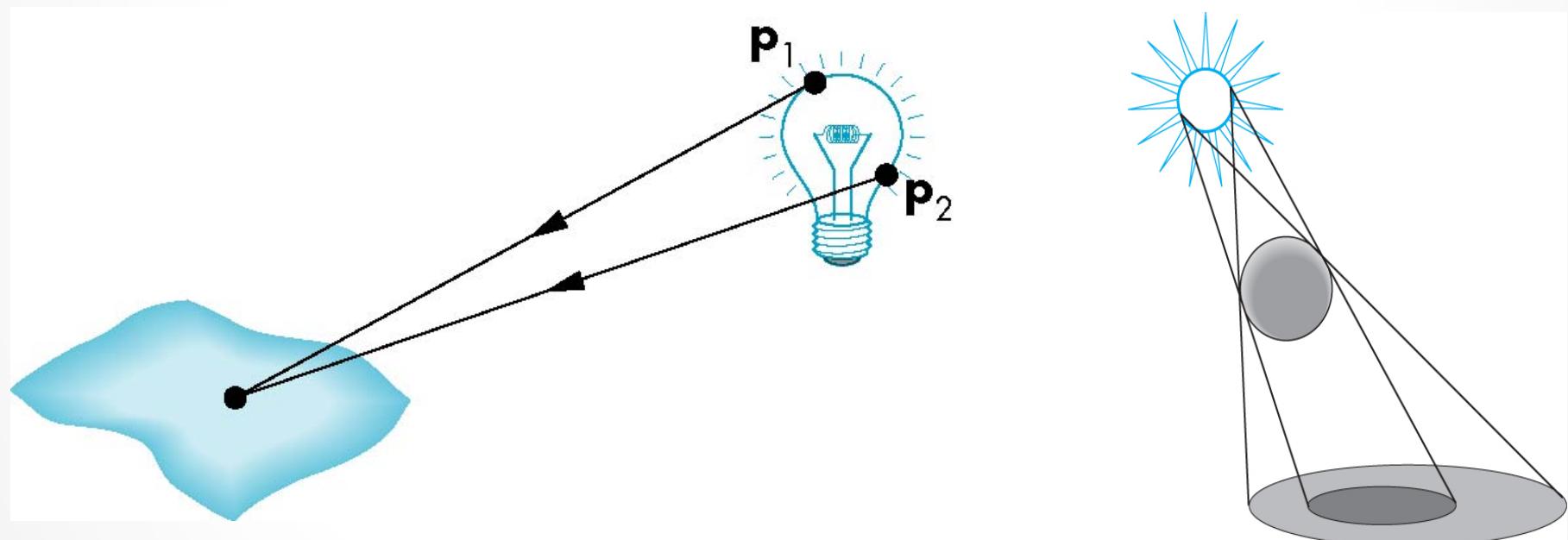
5.5. El model de Phong

5.6. Shading poligonal

5.3. Llums

Un **focus de llum** és un objecte que emet llum.

Les llums reals són difícils de modelar per què cal integrar l'emissió de la llum de tota la seva superfície

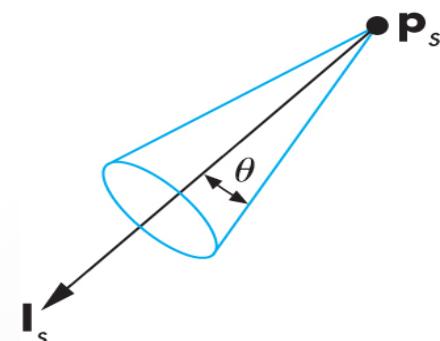
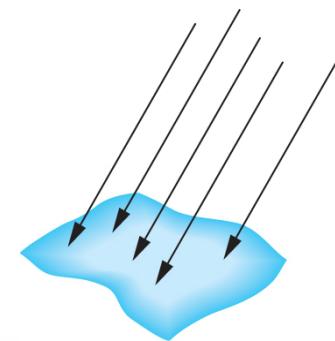
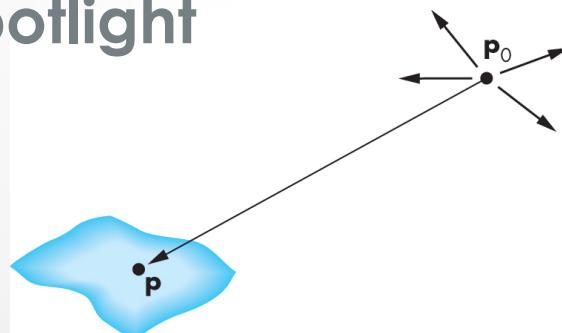


Tota llum emet en les tres components de color $\mathbf{L} = [I_r \ I_g \ I_b]$

5.3. Llums

Tipus de llums:

- Llum **ambient**: quantitat de llum constant en tots els punts de l'escena. Modela la component difusa de la llum.
- Llum **puntual**: No tenen dimensió, ni direcció (emeten en forma radial). Es caracteritzen per la posició i la seva emissió (o intensitat) Exemple: sol, bombetes, etc.
- Llum **direccional**: focus a l'infinít, es caracteritzen per la seva direcció i la seva intensitat
- **Spotlight**



5.3. Llums

Llum ambient:

És el resultat de múltiples interaccions entre llums i els objectes de l'entorn

Components difuses de la llum

El color percepbut depèn tant del color de la llum com de les propietats del material de l'objecte

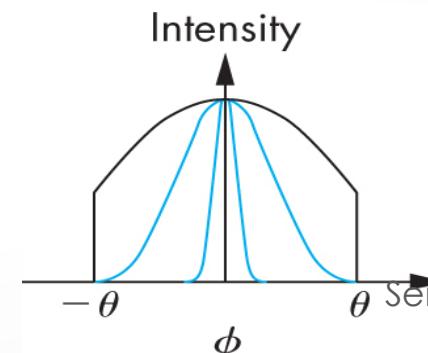
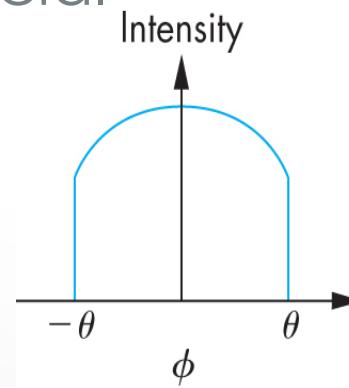
Es caracteritza amb les tres components de color R, G, B

$$I_a = [I_{ar} \mid I_{ag} \mid I_{ab}]$$

5.3. Llums

Atenuació en profunditat:

- La quantitat de llum rebuda per una superfície és inversament proporcional a la distància de la superfície al focus de llum $1/d^2$
- Es pot aplicar un factor de $1/(a + bd + cd^2)$ a les components de la llum, tan difusa com especular, que arriben a la superfície, on a, b, c són paràmetres que es determinen segons l'escena.
- en spotlights l'atenuació pot ser concentrada al centre del con, exponencial



Índex

5.1. Introducció

5.2. Interacció de les llums i els objectes

5.3. Llums

5.4. Materials

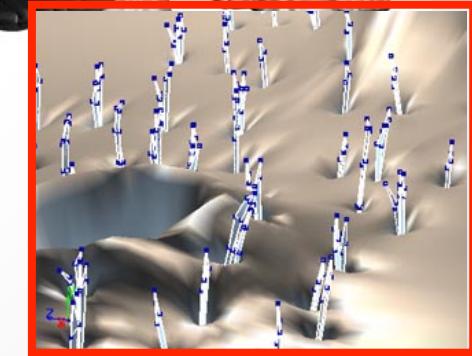
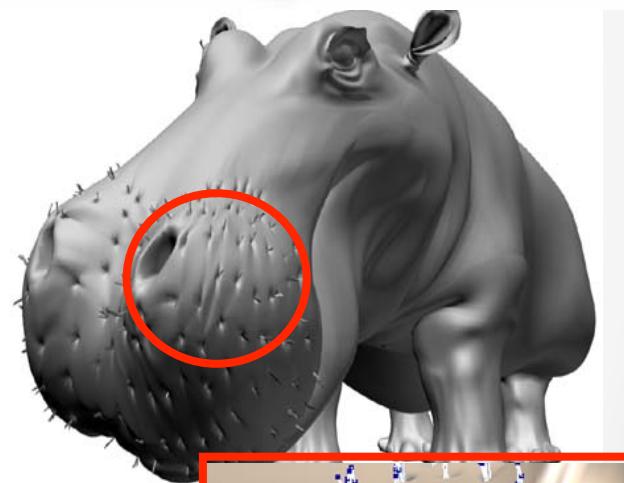
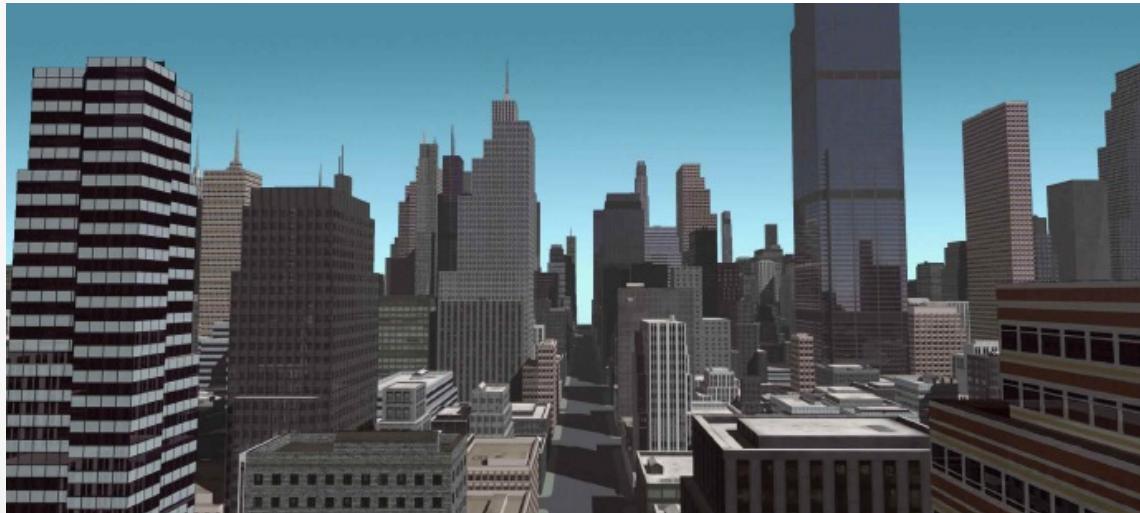
5.5. El model de Phong

5.6. Shading poligonal

5.4. Materials

El realisme implica definir un grau de detall en els objectes:

- modelant-los
- utilitzant **materials i textures**



5.4. Materials

Els materials dels objectes modelen l'aspecte de l'objecte

Els materials defineixen el tipus d'interacció de la llum amb l'objecte



Plastic



Metal



Matte

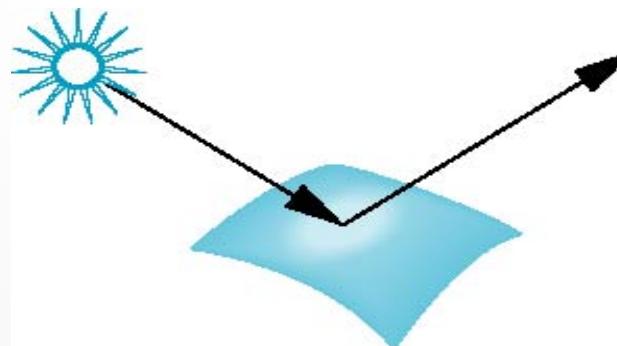
5.4. Materials

Un objecte pot:

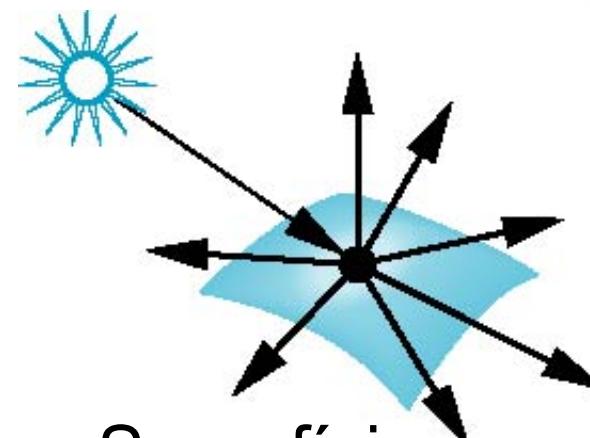
- **emetre llum**
- **absorbir llum**
- **reflectir llum**

en una superfície suau, la llum que es reflecteix es concentra en una direcció (mirall)

una superfície rugosa reflecteix la llum en totes direccions.



Superfície suau

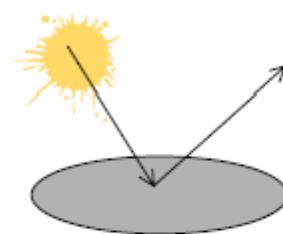


Superfície rugosa

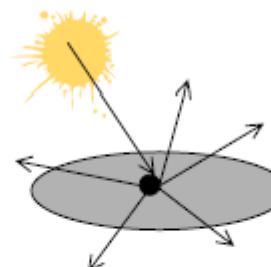
5.4. Materials

Reflexions en superfícies:

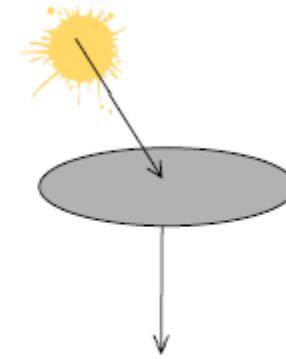
- **especulars**: es veuen molt brillants en un punt, ja que la llum reflectida es concentra en un rang d'angles propers a l'angle de reflexió.
- **difuses**: reflecteixen la llum en totes direccions i tenen la mateixa aparença en tots els observadors.
- **transmeses**: part de la llum penetra la superfície (refraccions) i es transmet cap a l'interior



especular



difusa



translucida

5.4. Materials

- **Propietats que defineixen un material**

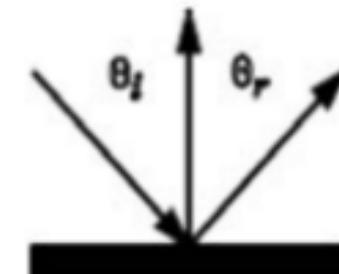
- Component difusa K_d (R, G, B)
- Component ambient K_a (R, G, B)
- Opacitat alfa = 0 (transparent) 1 (opac)
- Component especular $K_s(R, G, B)$
- Exponent de reflexió especular (shininess) $\alpha_r = 1...500$



5.4. Materials

Mirall pur:

- reflexió especular ideal
- llei de reflexió



Difusa:

- superfícies “mate”
- llei de Lambert



Especular:

- superfícies brillants i “glossy”
- models de Phong, Blinn-Phong



Índex

5.1. Introducció

5.2. Interacció de les llums i els objectes

5.3. Llums

5.4. Materials

5.5. El model de Phong

5.6. Shading poligonal

5.5. Model de Phong

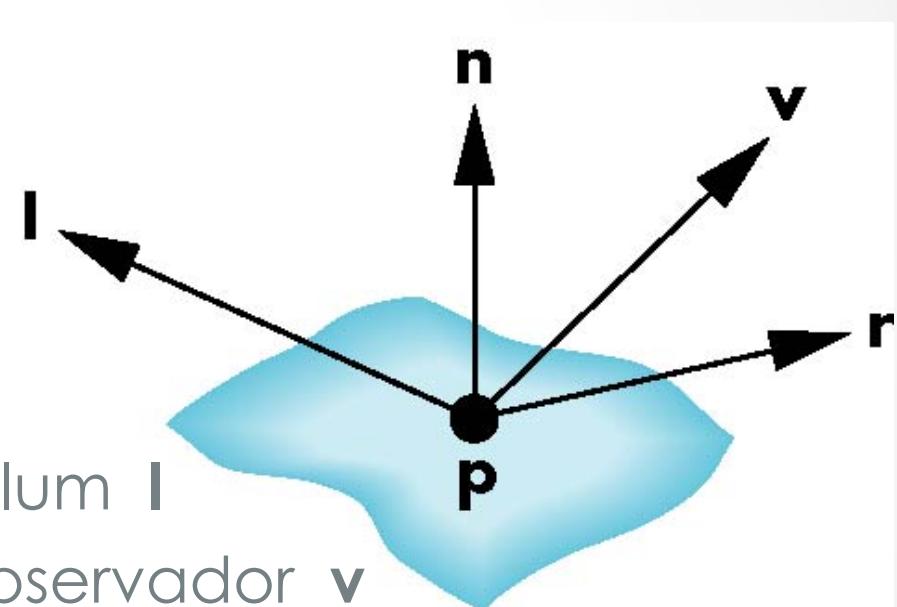
Model d'il·luminació **local** que es pot calcular de forma molt eficient integrat dins del pipeline gràfic.

Modela tres components:

- ambient
- difusa
- specular

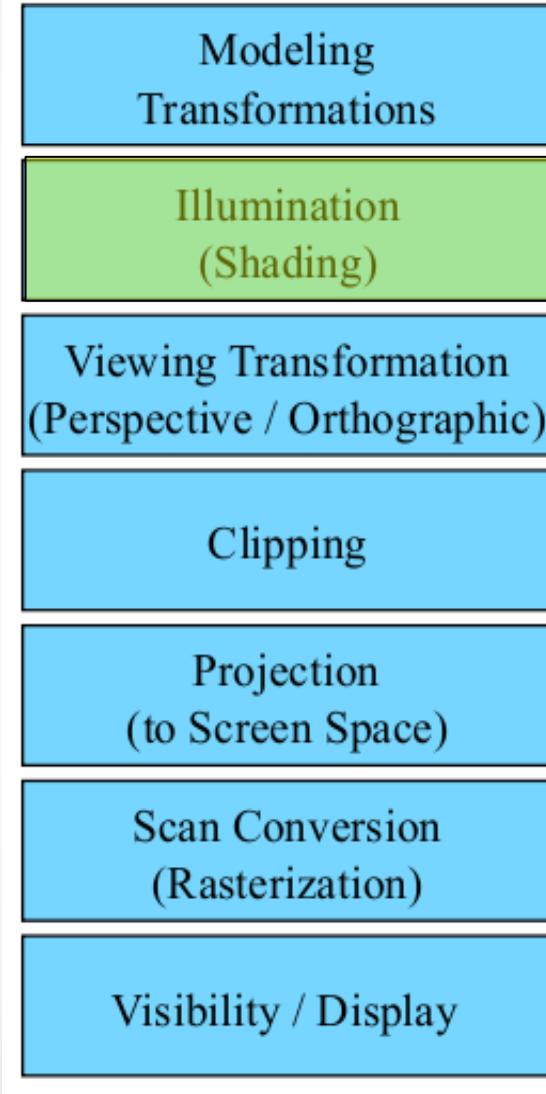
Utilitza quatre vectors:

- del punt de la superfície a la llum \mathbf{I}
- del punt de la superfície a l'observador \mathbf{v}
- la normal al punt \mathbf{n}
- el vector reflectit \mathbf{r}



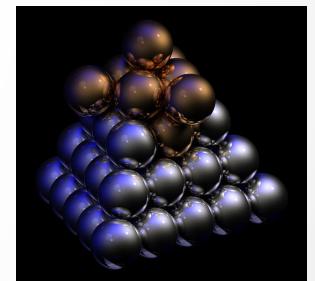
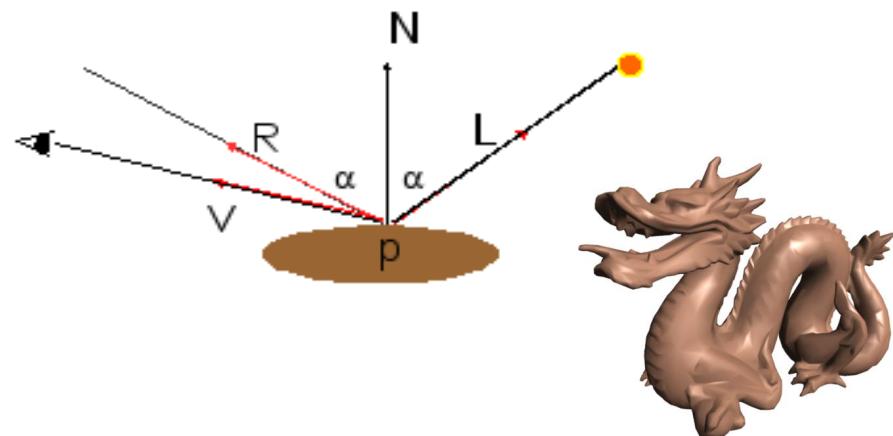
Considera que les llums emeten **llum difusa, specular i ambient**

5.5. Model de Phong



A partir dels vèrtexs dels objectes ja transformats, es calcula el ·luminació (**shading**) de cadascun segons les propietats dels materials, la geometria, les textures i les llums

- S'usen models **d'il·luminació locals** (Diffuse, Ambient, Phong, Blinn, etc.)



$$I_{rgb}(P, \vec{v}) = ka Ia_{rgb} Od_{rgb} + I(kd Od_{rgb} \vec{N} \cdot \vec{L} + ks Os_{rgb} (\vec{R} \cdot \vec{V})^n)$$

5.5. Model de Phong

- **Component ambient:**

La llum **ambient** modela les diferents interaccions entre la llum i els objectes de l'escena

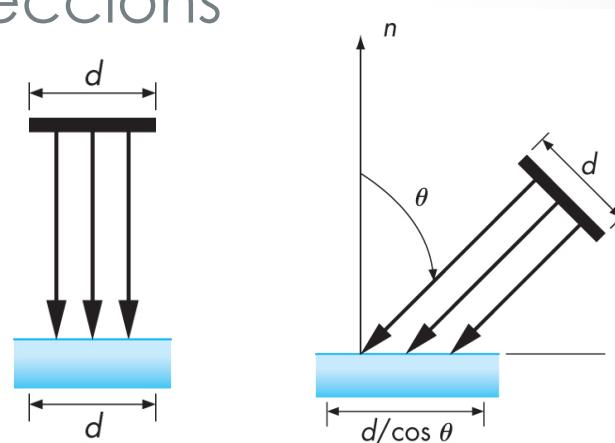
La llum ambient és **constant** a tots els punts, encara que es veu afectada per les propietats del material de cada objecte

S'utilitza una intensitat de **llum ambient global** I_{ag}

També s'afegeix el component I_a de cadascuna de les llums que il·luminen un punt.

5.5. Model de Phong

Llei de Lambert: Modela la llum **difusa** pura, on la llum es reflexa igualment a totes les direccions



La quantitat de llum reflectida és proporcional a la component vertical de la llum incident

La llum reflectida és proporcional al $\cos \theta_i$

$$\cos \theta_i = \mathbf{l} \cdot \mathbf{n} \quad (\text{amb els vectors normalitzats})$$

El coeficient K_d del material indica quanta llum de cada component (R, G, B) es reflexa en cada canal

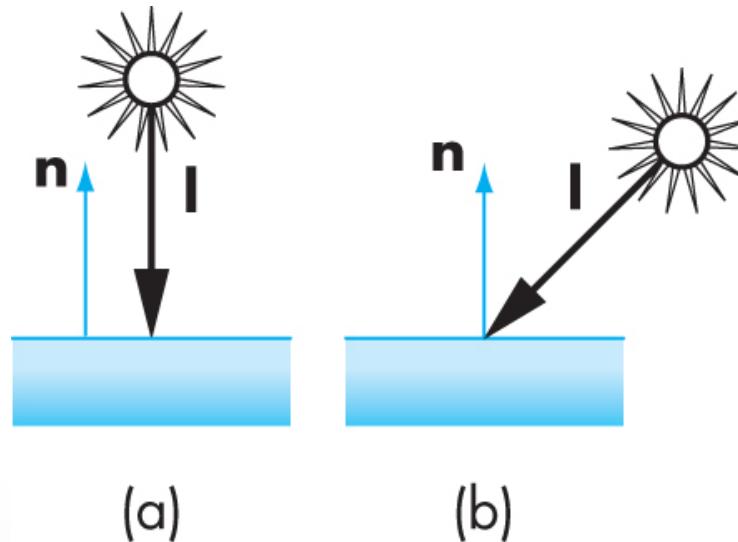
5.5. Model de Phong

La llum (o intensitat reflectida) serà:

$$\text{Color difús}_d = K_d (I \cdot n) I_d$$

Si es té en compte l'atenuació en profunditat seria:

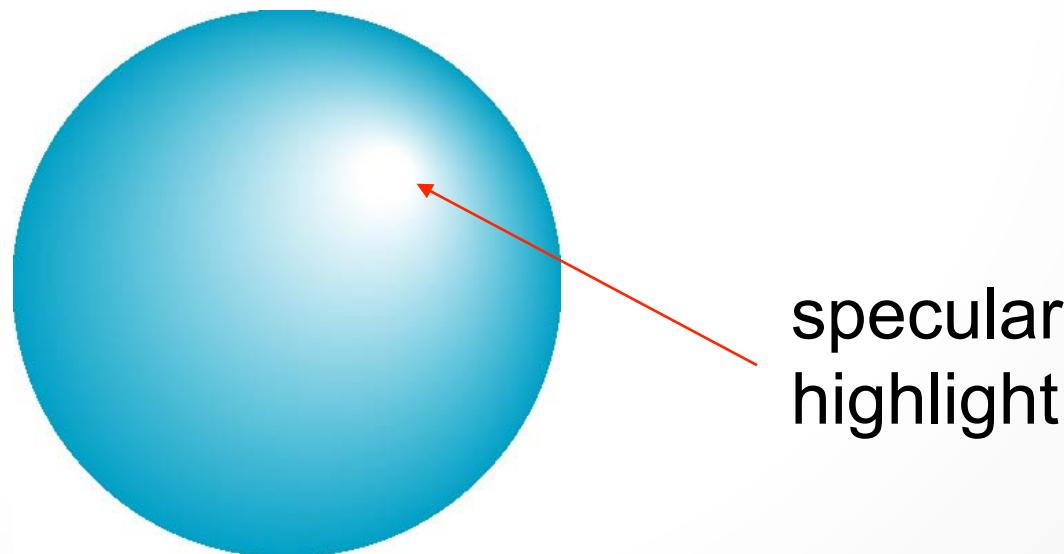
$$\text{Color difús}_d = (K_d / (a+bd+cd^2)) * (I \cdot n) I_d$$



5.5. Model de Phong

La majoria de superfícies no són idealment difuses ni idealment especulars

Les superfícies suaus mostren brillantors (*highlights*) especulars produïdes per que la llum incident es reflecteix en direccions concentrades prop de la direcció de la reflexió pura

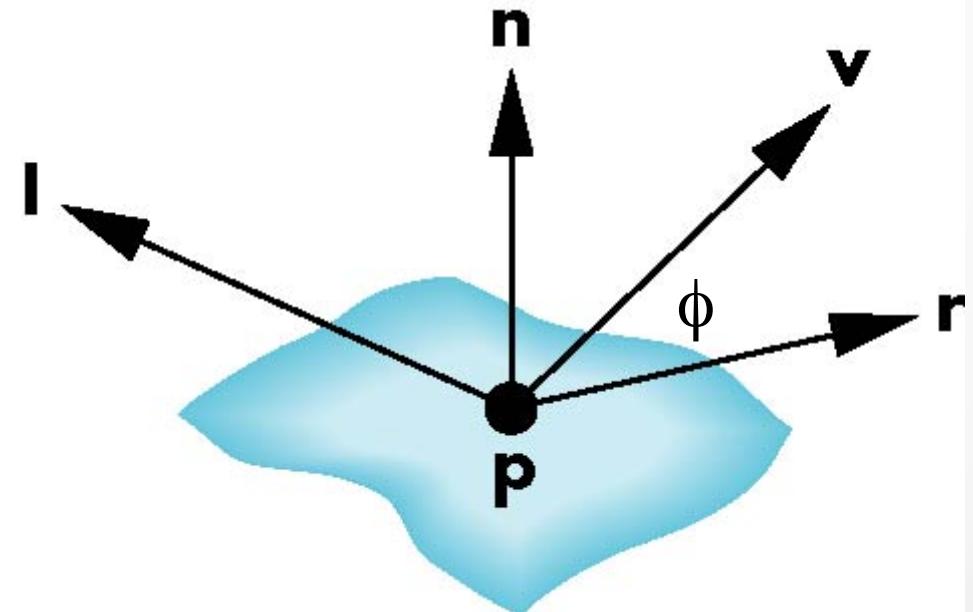


5.5. Model de Phong

Phong va proposar utilitzar un terme que es decrementés quan l'angle entre l'observador i la reflexió ideal augmentés.

$$I_r \sim k_s I \cos^\alpha \phi$$

Exponent de reflexió
especular



5.5. Model de Phong

La normal es determina per l'orientació local de la superfície

Cal calcular l'angle de reflexió

(1) angle d'incidència = angle de reflexió

$$\mathbf{l} \cdot \mathbf{n} = \mathbf{n} \cdot \mathbf{r}$$



(2) Els tres vectors són copланars: $\mathbf{r} = \mathbf{a} * \mathbf{l} + \mathbf{b} * \mathbf{n}$

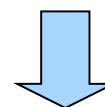


Sustituïnt \mathbf{r} a l'equació (1) dóna $\mathbf{b} = (\mathbf{l} \cdot \mathbf{n}) - \mathbf{a} * (\mathbf{l} \cdot \mathbf{n})$

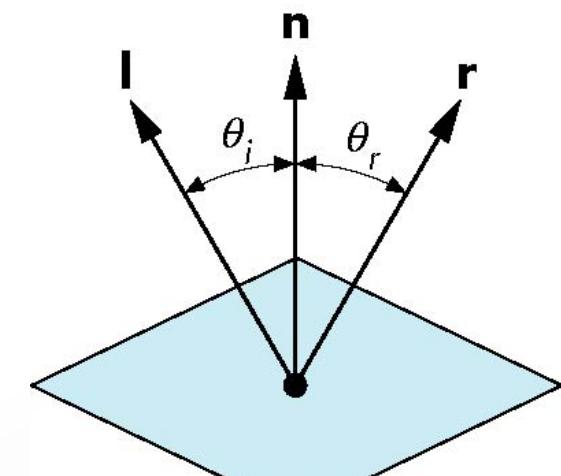
(3) Tots els vectors estan normalitzats $\mathbf{r} \cdot \mathbf{r} = 1$:



$$\mathbf{a} = -1; \mathbf{b} = 2 * (\mathbf{l} \cdot \mathbf{n})$$



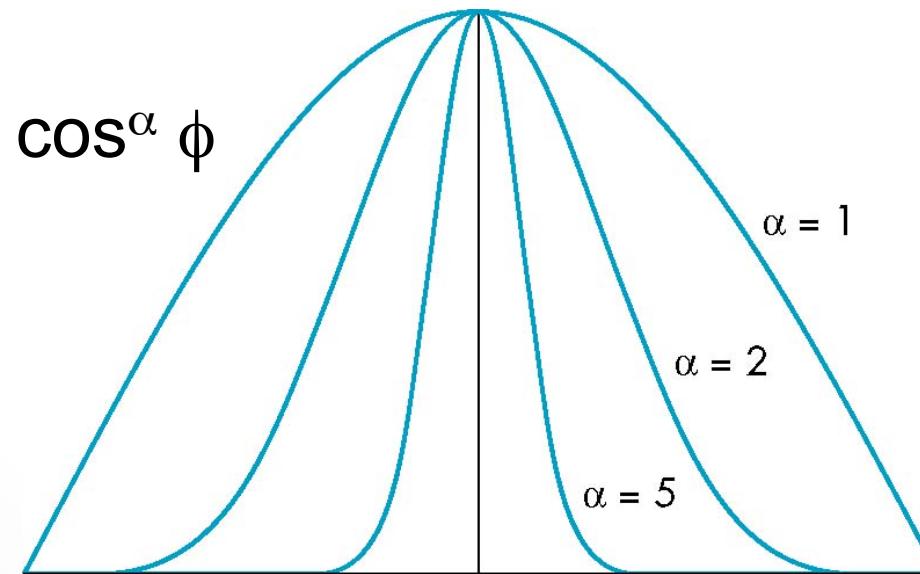
$$\mathbf{r} = 2 (\mathbf{l} \cdot \mathbf{n}) \mathbf{n} - \mathbf{l}$$



5.5. Model de Phong

L'exponent de reflexió espectral:

Els valors de α_s entre 100 i 200 corresponen als metalls.
Valors entre 5 i 10 donen una aparença de plàstic.



5.5. Model de Phong

En el model de Phong es **sumen** les contribucions de cadascuna de les fonts de llum.

Cada **llum** té termes separats de component difusa, component especular i component ambient. Cadascun d'ells està codificat en un color RGB.

Llavors, per a cada font de llum es tenen 9 valors:

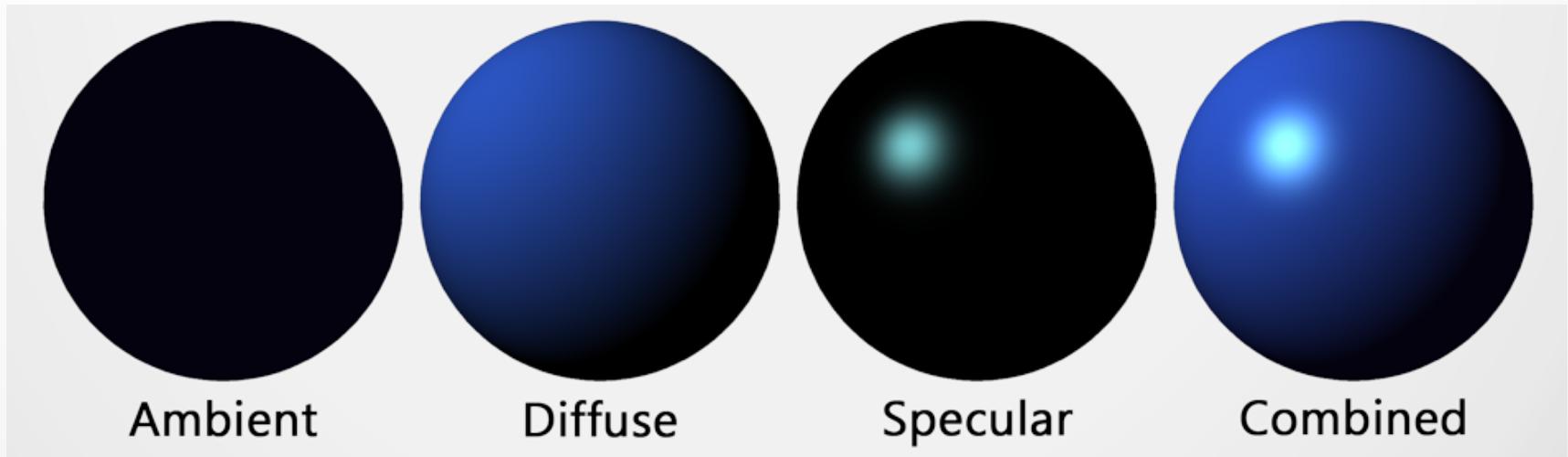
$$I_{dr}, I_{dg}, I_{db}, I_{sr}, I_{sg}, I_{sb}, I_{ar}, I_{ag}, I_{ab}$$

5.5. Model de Phong

Les propietats dels **materials** corresponen a les nou components de les llums

$$k_{dr}, k_{dg}, k_{db}, k_{sr}, k_{sg}, k_{sb}, k_{ar}, k_{ag}, k_{ab}$$

- Es modela la seva transparència (alpha)
- Es modela l'exponent de reflexió espectral α_r

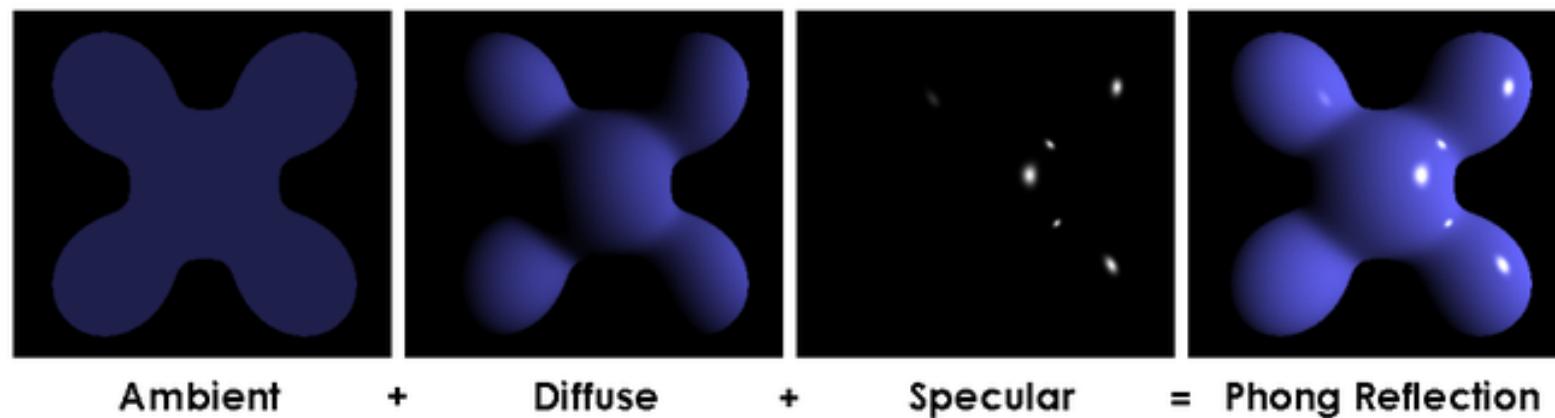
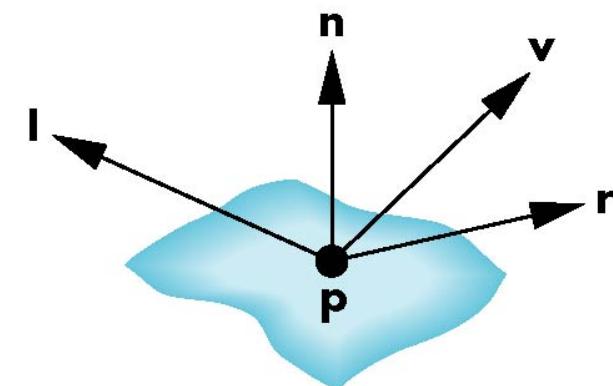


<http://www.youtube.com/watch?v=p34fKUqVoLI&feature=related>

5.5. Model de Phong

Per a cada llum i per a cada color, el model de Phong es pot escriure com (ignorant l'atenuació en profunditat)

$$I = k_d I_d \mathbf{l} \cdot \mathbf{n} + k_s I_s (\mathbf{v} \cdot \mathbf{r})^{\alpha s} + k_a I_a$$

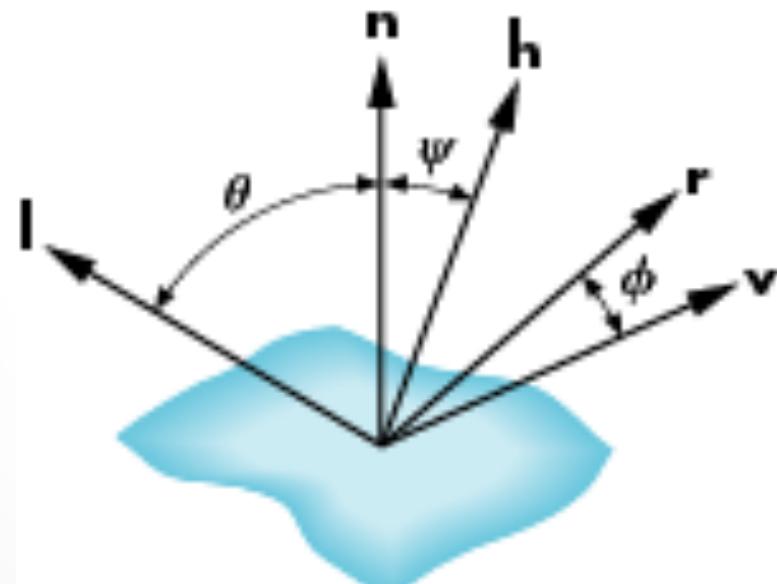


<http://www.youtube.com/watch?feature=endscreen&NR=1&v=hDAcp9bStRQ>

5.5. Model de Phong

El terme espectral de Phong és problemàtic ja que s'ha de recalcular un nou vector de **reflexió** i el vector de visió a cada vèrtex.

Blinn va proposar una aproximació usant el vector mig, que és més eficient.



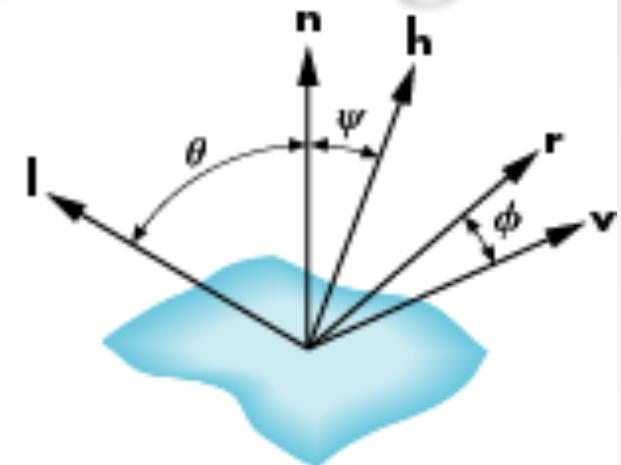
$$\mathbf{h} = (\mathbf{I} + \mathbf{v}) / |\mathbf{I} + \mathbf{v}|$$

\mathbf{h} és el vector mig normalitzat entre \mathbf{I} i \mathbf{v}

5.5. Model Blinn-Phong

Model de Blinn-Phong:

$$I = k_d I_d \mathbf{I} \cdot \mathbf{n} + k_s I_s (\mathbf{n} \cdot \mathbf{h})^\beta + k_a I_a$$

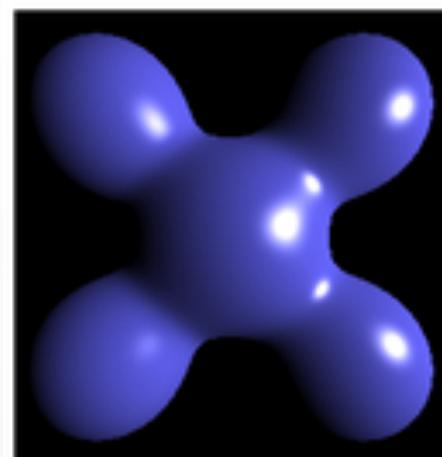


β s'usa per calcular la **shininess**

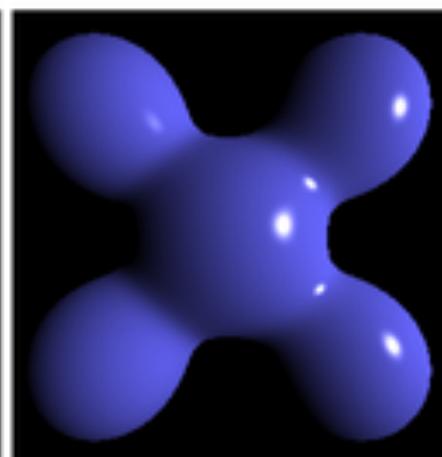
L'angle mig és un mig de l'angle entre \mathbf{r} i \mathbf{v} si els vectors són copланars.

És el que s'utilitza en el OpenGL estàndard.

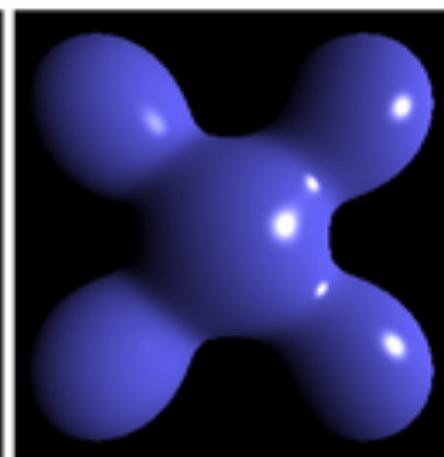
<http://www.youtube.com/watch?feature=endscreen&NR=1&v=nkM9v67z5J4>



Blinn-Phong



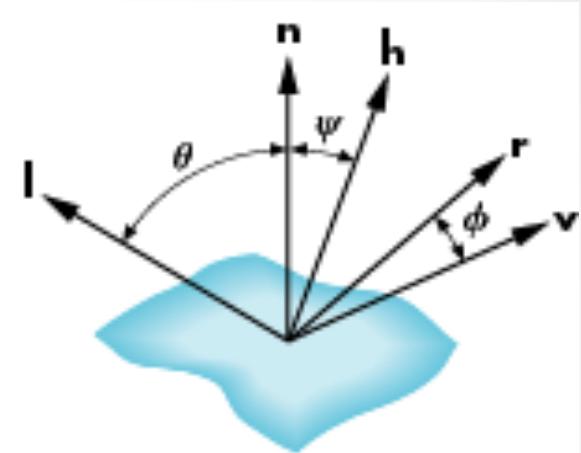
Phong



**Blinn-Phong
(higher exponent)**

5.5. Model Blinn-Phong

La intensitat total en un punt es calcula com la contribució de totes les llums de l'escena en el punt més la intensitat ambient global de l'escena.



$$I_{total} = I_{a_{global}} K_a + \sum_{i=1}^{numLlums} \frac{1.0}{a_i d_i^2 + b_i d_i + c_i} (I_{d_i} K_d \cdot \max(l \cdot n, 0) + I_{s_i} K_s \cdot \max((n \cdot h)^\beta, 0) + I_{a_i} K_a)$$

Índex

5.1. Introducció

5.2. Interacció de les llums i els objectes

5.3. Llums

5.4. Materials

5.5. El model de Phong

5.6. Shading poligonal

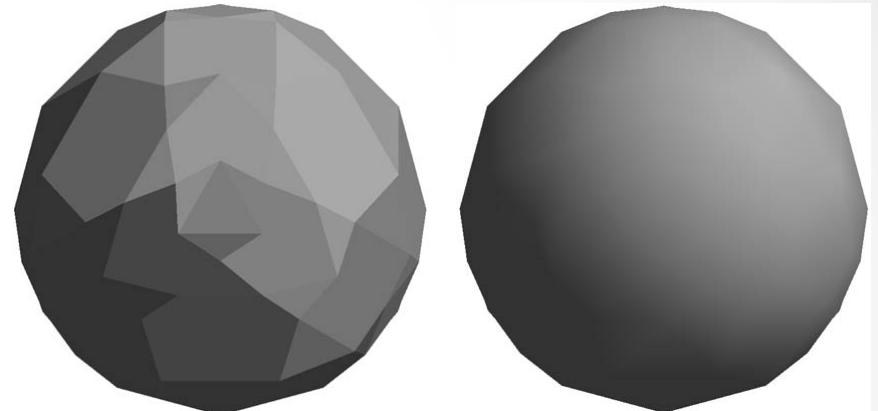
5.6. Shading poligonal

El *shading* poligonal pot ser:

- **flat shading**
- **suau (smooth i Gouraud)**
- **Phong shading**

Es necesiten:

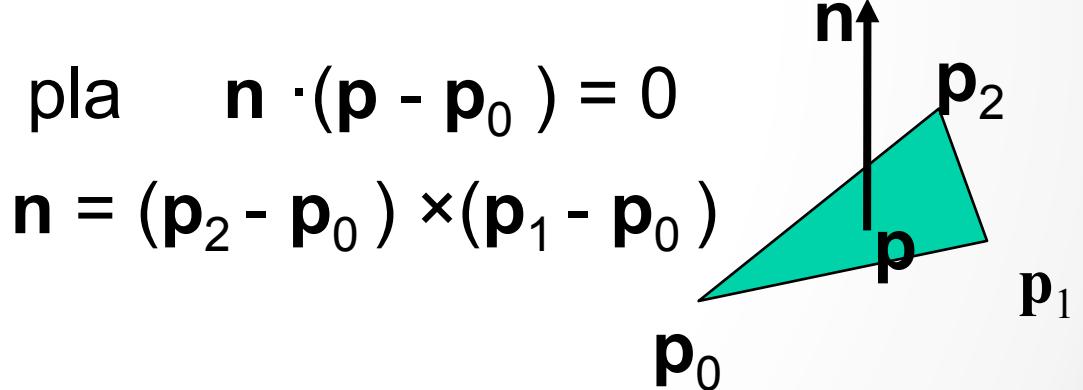
- les normals
 - les llums
 - les propietats dels materials
- S'envia tota la informació als shaders (vertex i fragment)



5.6. Shading poligonal

Flat shading:

- tots els punts d'una cara es pinten del mateix color, segons la normal al pla de la cara.

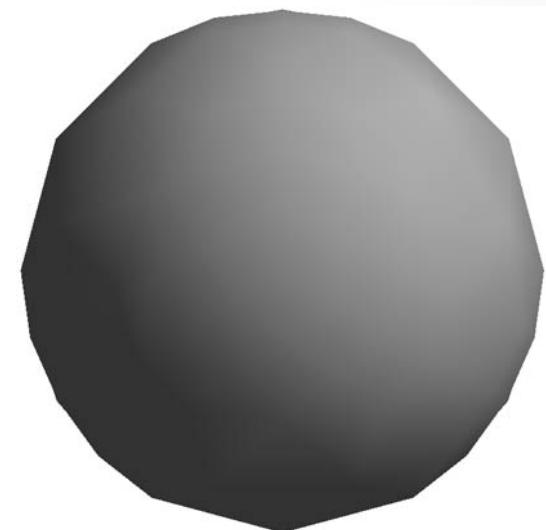


$$\text{normalització } n = n / |n|$$

5.6. Shading poligonal

Smooth shading:

- Es calcula la il·luminació a cada vèrtex i s'interpola el color a cada punt de la cara (segons els seus vèrtexs frontera)
- Cas de superfícies **paramètriques**: es pot calcular la directament la normal a cada vèrtex

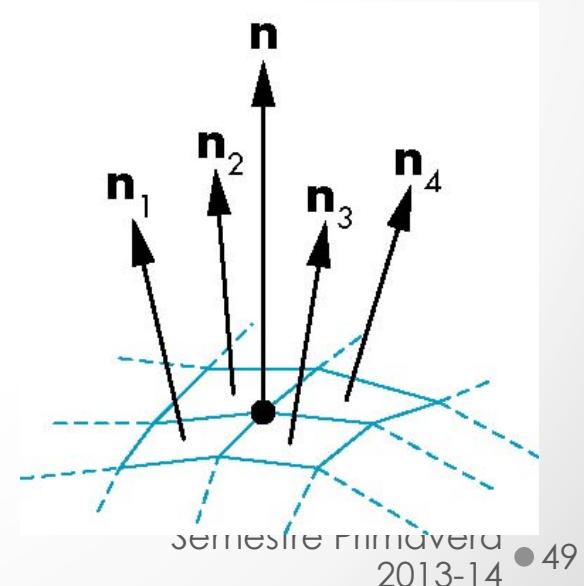


5.6. Shading poligonal

Gouraud Shading

- Usat en models poligonals per calcular les normals a cada vèrtex
- Es calcula el promig de les normals a cada vèrtex
- S'aplica el model de Blinn-Phong a cada vèrtex
- S'interpolen les intensitats de cada vèrtexs a cada polígon

$$\mathbf{n} = (\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4) / |\mathbf{n}_1 + \mathbf{n}_2 + \mathbf{n}_3 + \mathbf{n}_4|$$



5.6. Shading poligonal

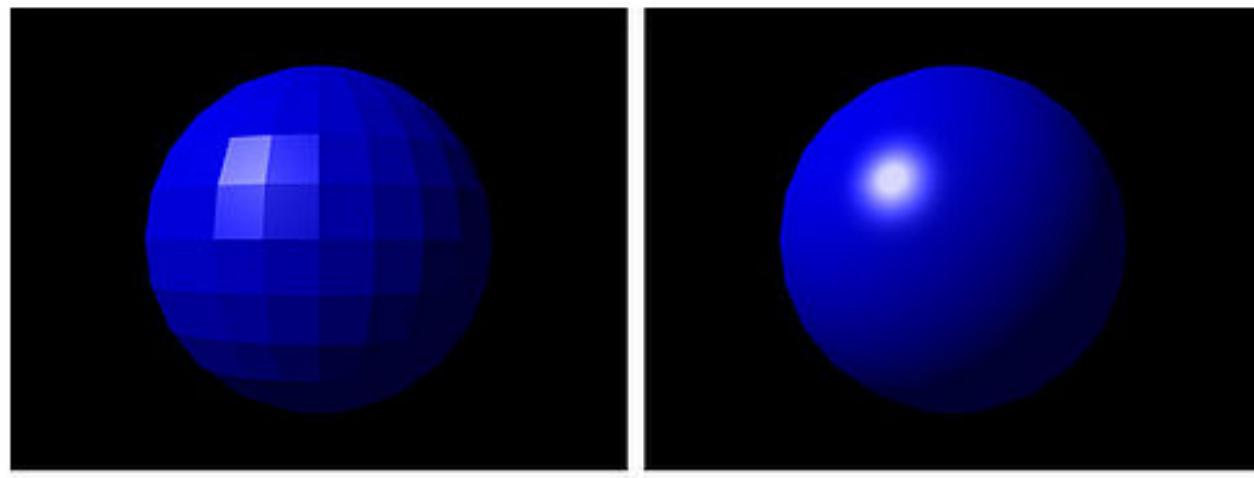
On es calcula la il·luminació?

- En la CPU o en la GPU
- Si es calcula la il·luminació a cada vèrtex:
 - Es pot calcular a la CPU el color de cada vèrtex i enviar-ho al vèrtex shader com un atribut del vèrtex
 - Es pot calcular a la GPU en el vertex shader, enviant prèviament els paràmetres al vertex shader
- En general, per defecte, la il·luminació de cada vèrtex s'interpola si es passa el color al fragment shader amb una variable de tipus *varying* (**smooth shading**)
- Si s'usa com a variable *uniform* llavors s'obté el **flat shading**

5.6. Shading poligonal

Phong shading

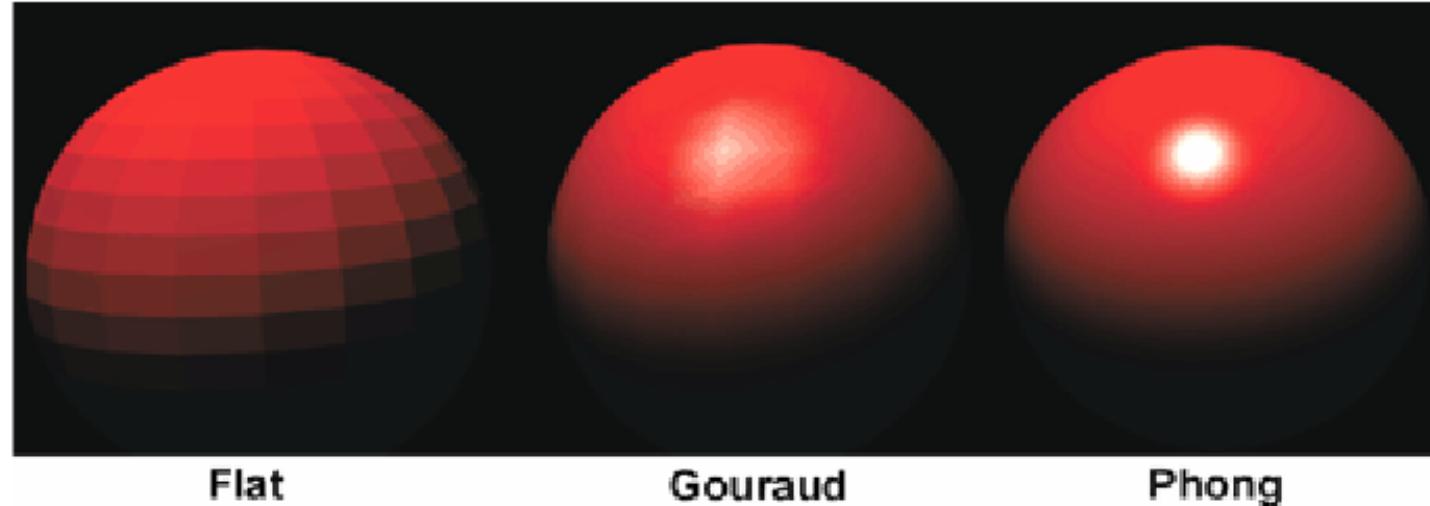
- Es troben les normals a cada vèrtex
- S'interpolen les normals en les arestes
- S'interpolen les normals en els punts interiors del polígon
- S'aplica el model de Blinn-Phong a cada fragment discretitzat



5.6. Shading poligonal

Si els polígons aproxiuen una superfície amb **curvatures molt altes**, el Phong shading es veu més suau que el Gouraud

El Phong shading implica més càlculs que el Gouraud shading

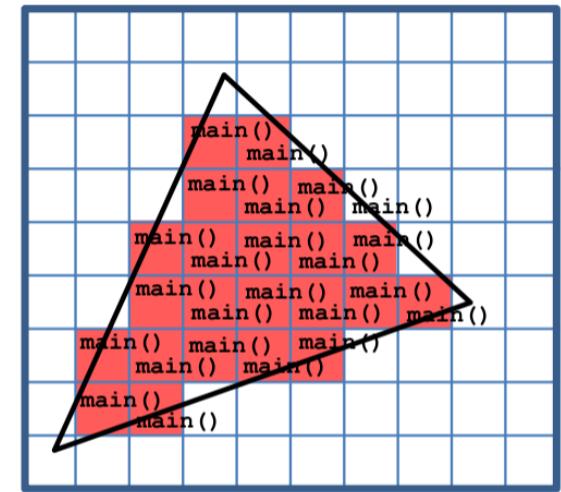
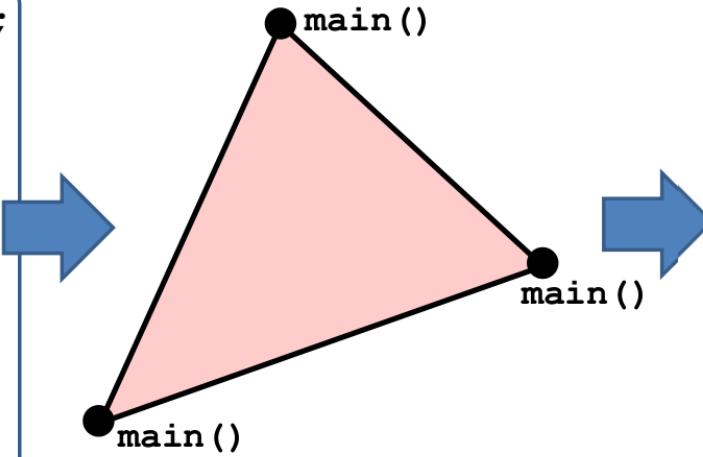


Vertex shader o fragment shader?

```
glBegin(GL_TRIANGLES);

    glVertex3f(0,0,0);
    glVertex3f(1,0,0);
    glVertex3f(1,1,0);

glEnd();
```



5.6. Shading poligonal

A nivell d'implementació a la GPU:

- Si es calcula el model de Blinn-Phong a cada vèrtex en el vertex shader:
 - S'interpola el **color en el fragment shader** (varying)
 - **Gouraud**
- Si s'obté la **normal interpolada al fragment shader** i allà es calcula el color:
 - **Phong shading**

5.6. Shading poligonal

Toon shading:

- Tècnica no-fotorealista
- Usa pocs colors (tons) en funció de l'angle entre la llum i la normal de l'objecte
- Els tons s'escullen en funció del cosinus de l'angle entre la llum i la normal a la superfície.
- Si la normal és propera a la direcció de la llum, s'escull un to més clar. Si és llunyana, s'escull un to més fosc.



5.6. Shading poligonal

- **Toon shading:** Implementable en el fragment shader directament: interpolant el **color** o les normals.

```
// vertex shader
uniform vec3 lightDir;
out varying float intensity;
void main()
{
    intensity = dot(lightDir,gl_Normal);
    ...
}
```



```
// fragment shader
in float intensity;
void main()
{
    vec4 color;
    if (intensity > 0.95)
        color = vec4(1.0,0.5,0.5,1.0);
    else if (intensity > 0.5)
        color = vec4(0.6,0.3,0.3,1.0);
    else if (intensity > 0.25)
        color = vec4(0.4,0.2,0.2,1.0);
    else
        color = vec4(0.2,0.1,0.1,1.0);
    gl_FragColor = color;
}
```

5.6. Shading poligonal

- **Toon shading:** Implementable en el fragment shader directament: interpolant el color o les **normals**.

```
// vertex shader
out vec3 normal;

void main()
{
    normal = gl_Normal;
    ...
}
```



```
// fragment shader
uniform vec3 lightDir;
in vec3 normal;

void main()
{
    float intensity;
    vec4 color;
    intensity = dot(lightDir,normalize(normal));

    if (intensity > 0.95)
        color = vec4(1.0,0.5,0.5,1.0);
    else if (intensity > 0.5)
        color = vec4(0.6,0.3,0.3,1.0);
    else if (intensity > 0.25)
        color = vec4(0.4,0.2,0.2,1.0);
    else
        color = vec4(0.2,0.1,0.1,1.0);
    gl_FragColor = color;
```