Cours 2: *Shading*

Xavier Décoret – INF 683 - École Polytechnique

Overview

- Aspect des objects
  - Qu'est ce qui l'influence?
  - Comment le modéliser?
  - Comment le calculer efficacement?
- Textures
  - À quoi ça sert?
  - Comment ça s'utilise?
  - Qu'est ce que ça implique?

Qu'est ce qui influence l'aspect?

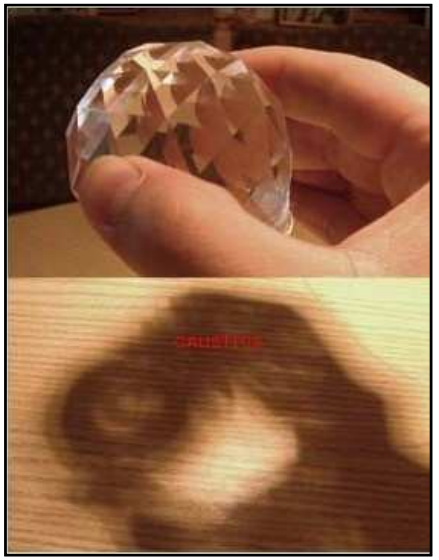
- Les matériaux constituant les objets
  - Comment ils "transmettent" la lumière incidente
- Les interactions lumineuses entre objets
  - Comment la lumière voyage des sources jusqu'à l'oeil

Fresnel effect

Isotropic highlights

Anisotropic highlights

Dispersion



Caustics

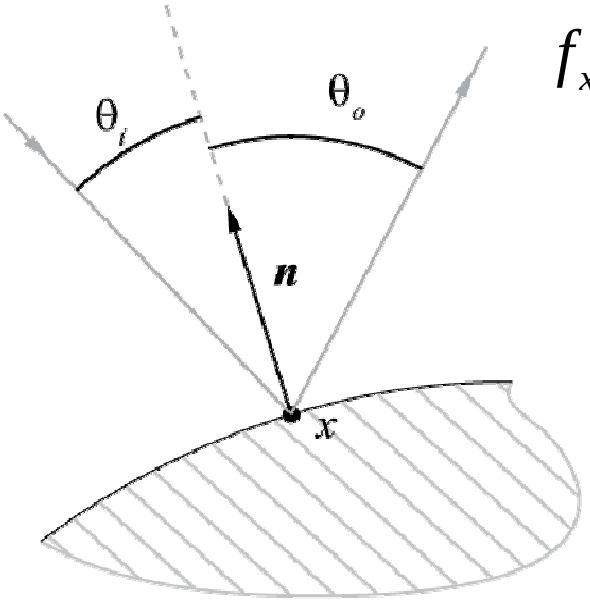
http://www.maxim-capra.com

Matériau & transmission de la lumière

Jensen, H. W et al. *A practical model for subsurface light transport*. 2001

BRDF *(Bidirectional Reflectance Function)*

Fonction de IR[4] dans IR indiquant comment la lumière incidente en un point est réfléchie en ce point

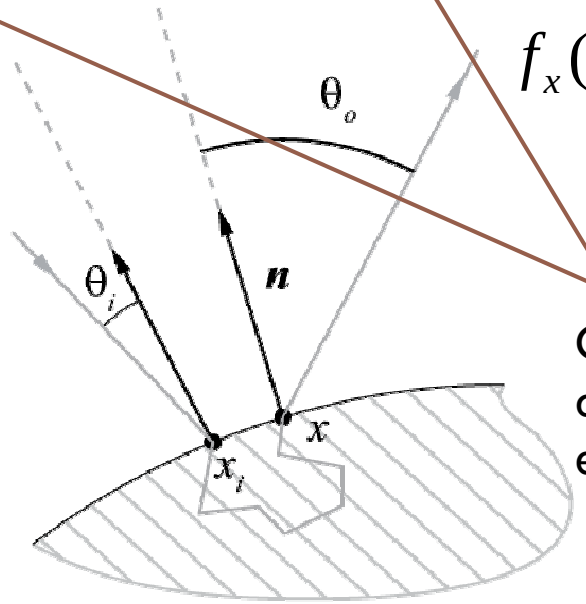


$$f_x(\theta_i, \theta_o, \lambda)$$

θ_i : incoming light
θ_i : outgoing light
λ : longueur d'onde

BSSRDF *(Subsurface Scattering)*

Fonction de IR[7] dans IR indiquant comment la lumière incidente en un autre point est réfléchie en un point



$$f_x(x_i, \theta_i, \theta_o, \lambda)$$

On ignore la dépendence en λ en pratique

Matériau & transmission de la lumière

BRDF *(Bidirectional Reflectance Function)*

BSSRDF *(Subsurface Scattering)*

http://graphics.ucsd.edu/~henrik/images/subsurf.html

Pas de BSSRDF dans ce cours

Interactions lumineuses

Illumination globale

- Le shading en un point prend en compte toutes les interactions lumineuses dans la scène

Illumination locale

- Le shading en un point ne dépend que de la source de lumière, de l'observateur, et du matériau en ce point

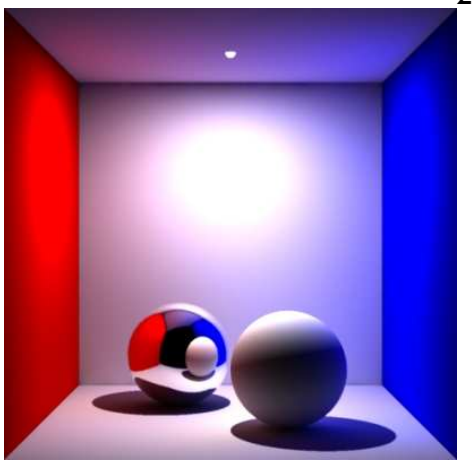
Illumination Globale (1/5)

- Clé du réalisme
  - Ombres, réflections
  - *Color bleeding*

http://www.cg-cars.com/forum/gallery/

1

2



[1] Laine et al. *Soft Shadow Volumes for Ray Tracing*, Siggraph 2005
[2] http://www.seanet.com/~myandper/abstract/sig03c09.htm

Illumination Globale (2/5)

- Compliqué et coûteux
  - *Rendering Equation* Kajiya, James T., *The rendering equation*, SIGGRAPH 1986

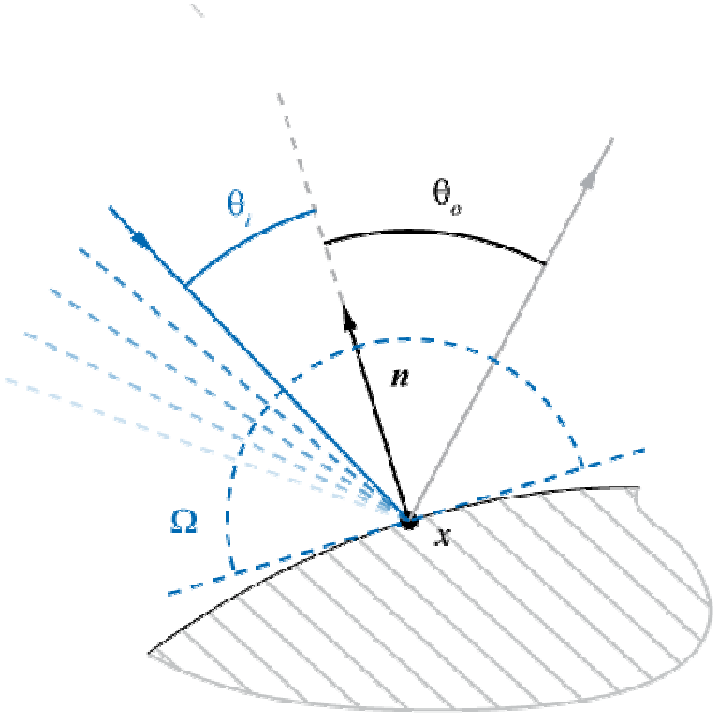
$$L_o(x,\theta_o) = L_e(x,\theta_o) + \int_\Omega f_x(\theta_i,\theta_o) L_i(x,\theta_i) \cos\theta_i d\theta_i$$

Lumière quittant x dans la direction θ_o

Lumière propre émise par x dans la direction θ_o

Bidirectional Reflectance Function en x

Lumière arrivant en x dans la direction θ_i



Illumination Globale (3/5)

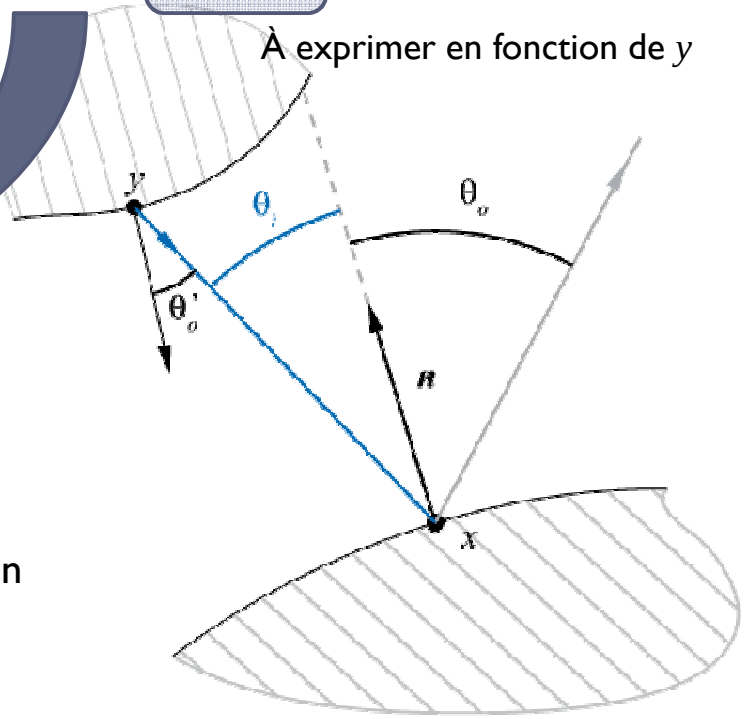
- Compliqué et coûteux
  - *Rendering Equation* Kajiya, James T., *The rendering equation*, SIGGRAPH 1986

$$L_o(x,\theta_o) = L_e(x,\theta_o) + \int_y f(x,\theta_i,\theta_o) L_o(y,\theta_o') V(x,y) \cos\theta_i d\theta_i$$

À exprimer en fonction de y

Tous les autres points y de la scène

Récursif!



$V(x, y)$ = 1 ou 0 selon que x et y se voient ou non

Illumination Globale (4/5)

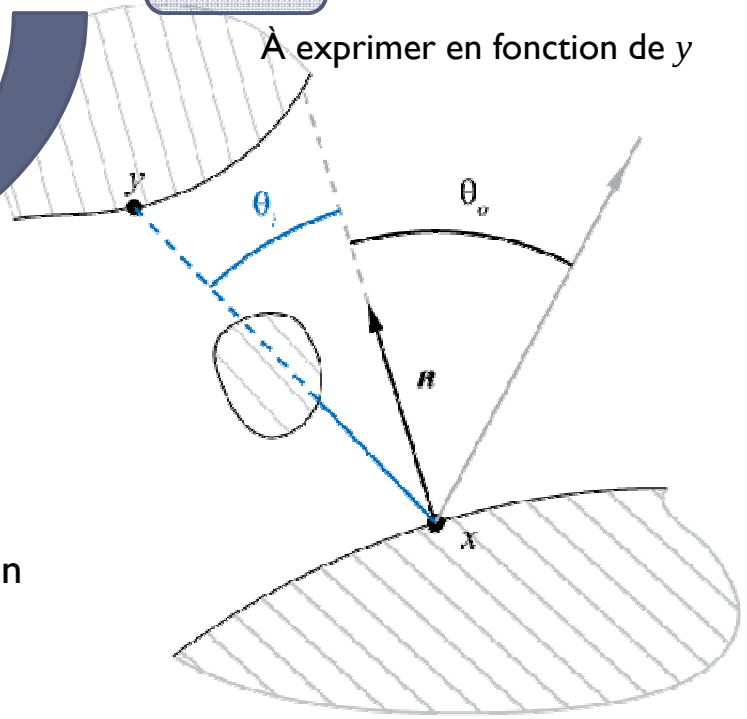
- Compliqué et coûteux
  - *Rendering Equation* Kajiya, James T., *The rendering equation*, SIGGRAPH 1986

$$L_o(x,\theta_o) = L_e(x,\theta_o) + \int_y f(x,\theta_i,\theta_o) L_o(y,\theta_o') V(x,y) \cos\theta_i d\theta_i$$

À exprimer en fonction de y

Tous les autres points y de la scène

Récursif!



$V(x, y)$ = 1 ou 0 selon que x et y se voient ou non

Illumination Globale (5/5)

- Compliqué et coûteux
  - Récursivité
  - Calcul de visibilité (cf. facteur de forme)
- Comment s'en sortir?
  - Récursivité limité (sources virtuelles, un seul rebond)
  - Approche hiérarchique
- Qu'est ce qui est utilisé actuellement?
  - Radiosité (notamment temps-réel)
  - Photon mapping
  - Precomputed Radiance Transfer

Pas dans ce cours

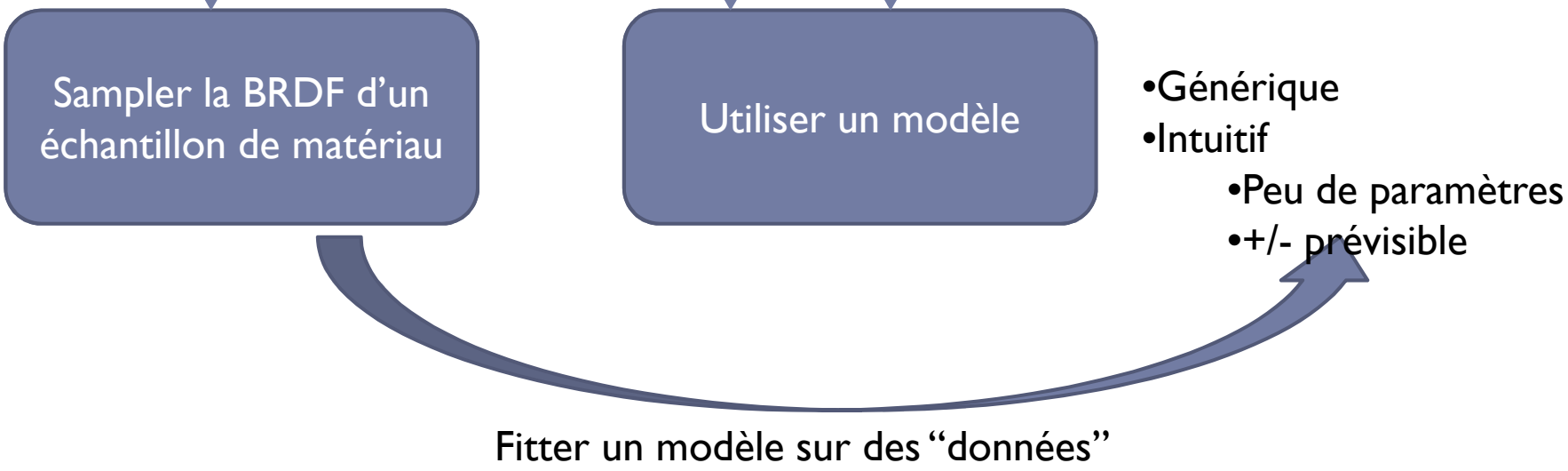
Illumination locale

- Est beaucoup plus simple et rapide à évaluer
  - Calcul efficace par les cartes graphiques
- Rate la plupart des effets visuels importants
  - Notamment, ne gère pas les occlusions
  - Pleins de "hacks" pour émuler les effets manquants

Dans ce cours, on ne considère que l'illumination locale.

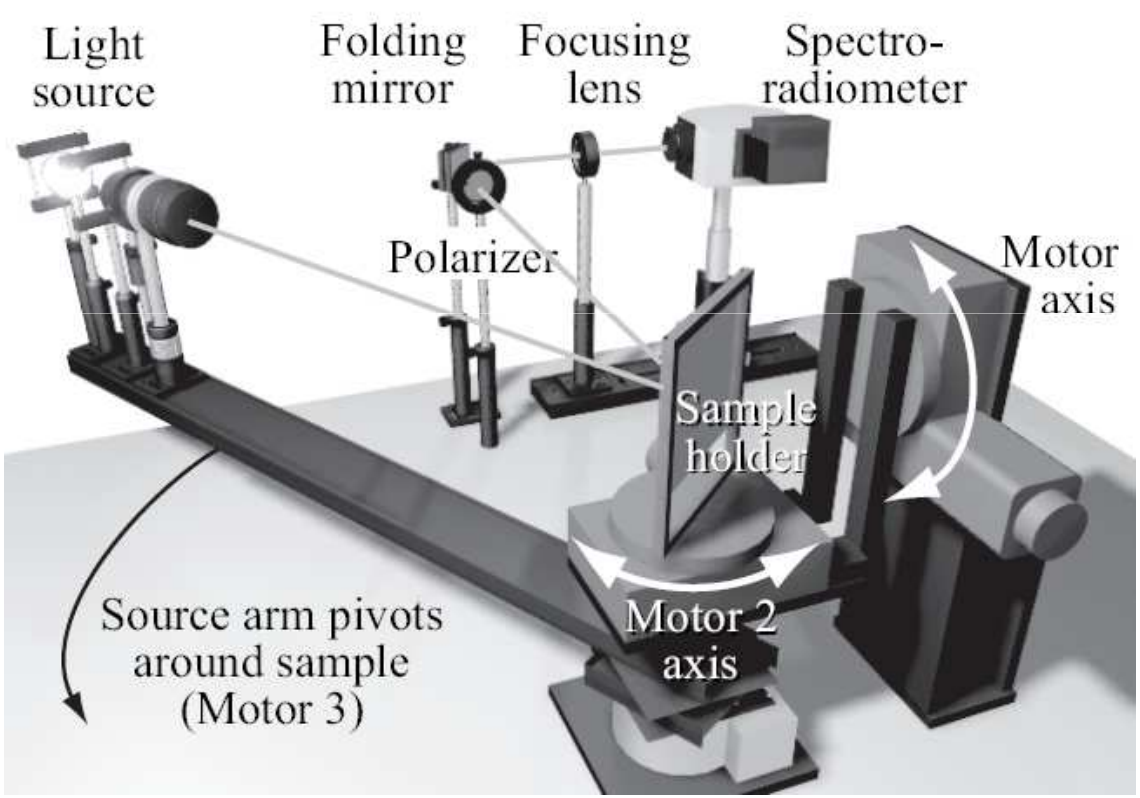
De la théorie à la pratique

- Comment spécifier les matériaux?
  - Un matériau précis du monde réel
  - Un matériau auquel pense un artiste/graphiste
- Comment évaluer une BRDF efficacement?



Sampler une BRDF

- On peut utiliser un gonioreflectomètre



Si si, ça existe!

http://www.graphics.cornell.edu/~westin/

Modèles de BRDF

- Modèles simplifiés
  - Gouraud
  - Phong
  - Blinn-Phong
- Modèles complexes
  - Un exemple pour le cours
    - Cook-Torrance
  - D'autres pour la culture
    - Ward, Torrance-Sparrow, Kubelka-Munk, …

http://en.wikipedia.org/wiki/Specular_highlight

Modèles simplifiés

- Somme de 3 termes
  - Ambient
  - Diffus
  - Spéculaire

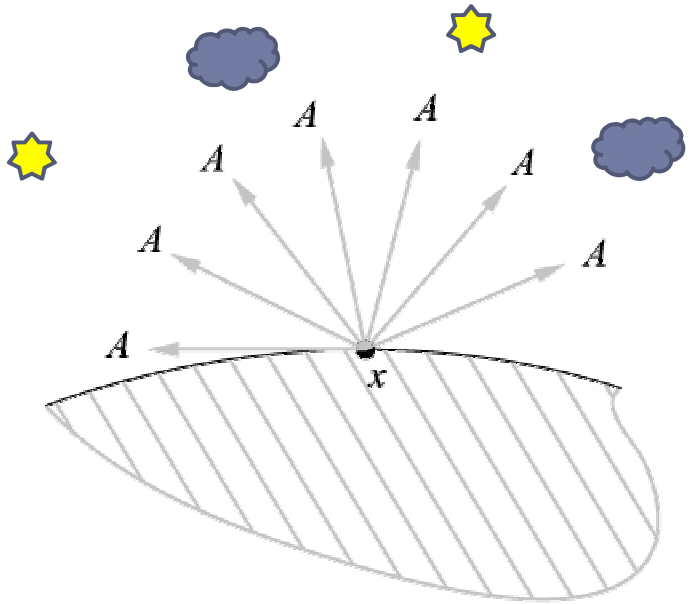
Pour chaque terme, il y a:
- une intensité de source
- un coefficient d'absorption
- une fonction de transfert

- Deux formules pour le spéculaire
  - Phong
  - Blinn-Phong
- Trois façons de les évaluer
  - Par face
  - Par sommet
  - Par fragment

Terme ambient (1/2)

- Représenter la lumière "dans la scène" i.e. "l'ambiance"
  - Lumière provenant du dôme céleste
  - Lumière que la scène réfléchit sur elle-même
- Modélisation
  - Ne dépend pas des angles d'incidence et de reflection

$$A = I_a K_a$$



Terme ambient (2/2)

- Très simpliste
  - N'a pas vraiment d'interprétation physique
  - Ne donne pas d'indications sur la forme des objets

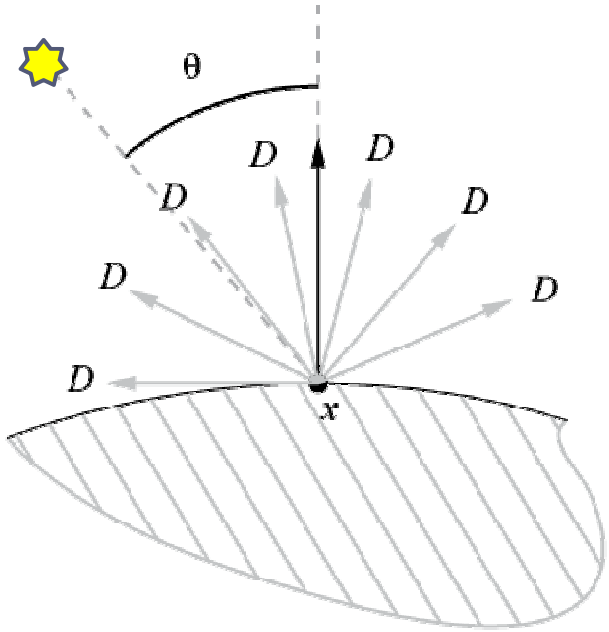
K_a croissant

- Mais bien pratique
  - Émulation bon marché de l'illumination globale

Terme diffus (1/2)

- Représenter un transfert isotrope d'énergie
  - Réflection égale dans toutes les directions
  - Conservation de l'énergie
- Modélisation
  - Fait intervenir l'orientation locale de la surface

$$D = I_d K_d \cos\theta$$



Terme diffus (2/2)

- Hypothèse isotrope *a.k.a* de matériau lambertien
- Le shading varie le long de la surface
  - Renseigne sur la courbure (donc la forme)de l'objet

K_d croissant

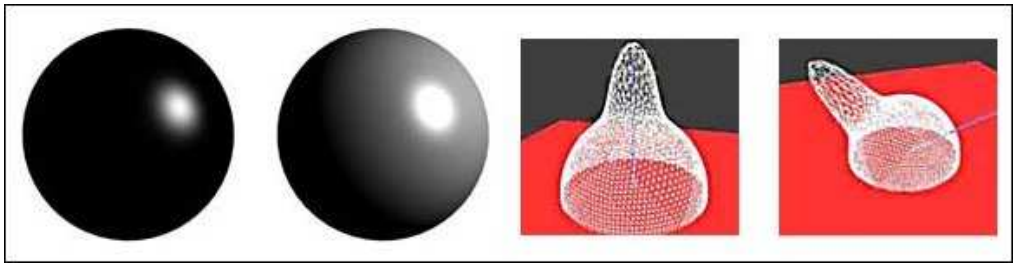
  - Mais ne dépend pas de la position de l'observateur

Terme spéculaire (1/2)

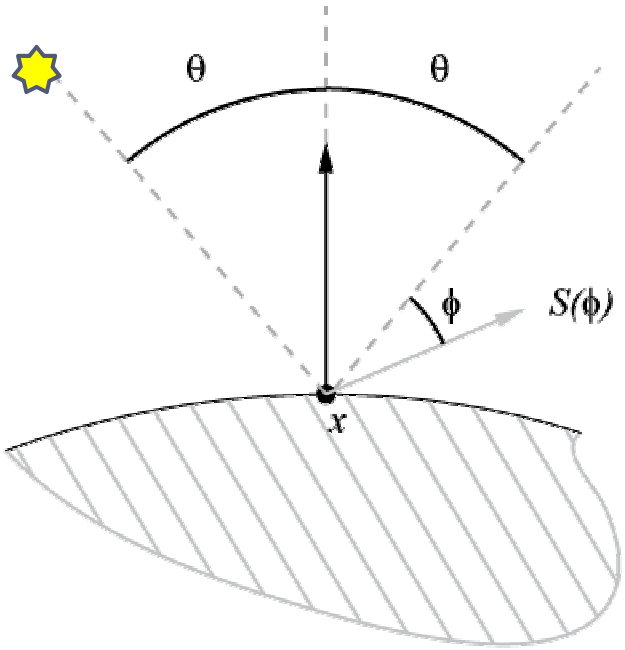
- Représenter les surfaces "brillantes" (*glossy*)
  - Cas idéal des miroirs
- Modélisation
  - Autour de la réflection miroir
  - Avec une décroissance exponentielle

$$S(\phi) = I_s K_s (\cos\phi)^n$$

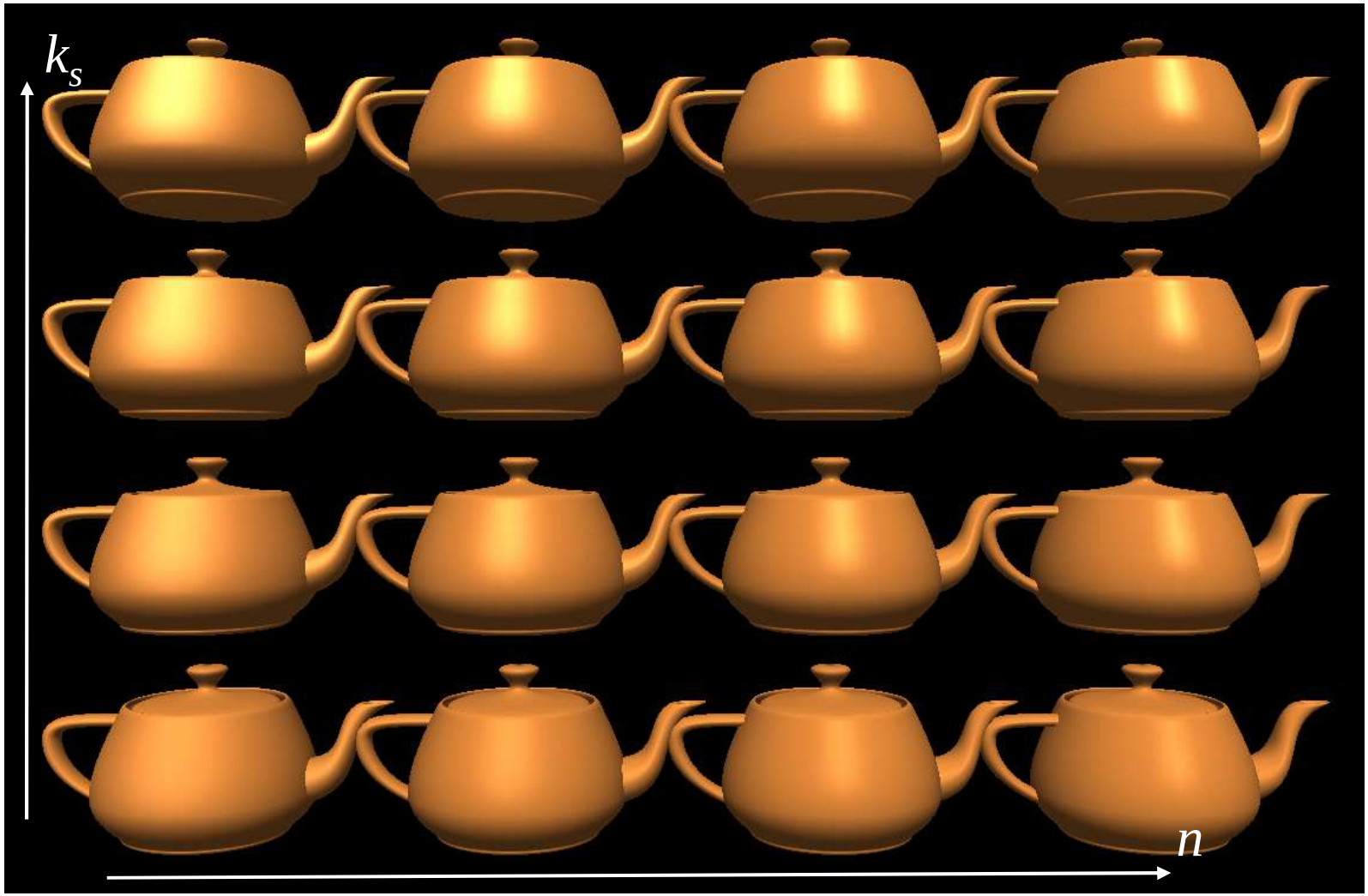
n : *shininess*



spéculaire | spéculaire+diffus | Visualisation de S(ϕ) *a.k.a* lobes spéculaires

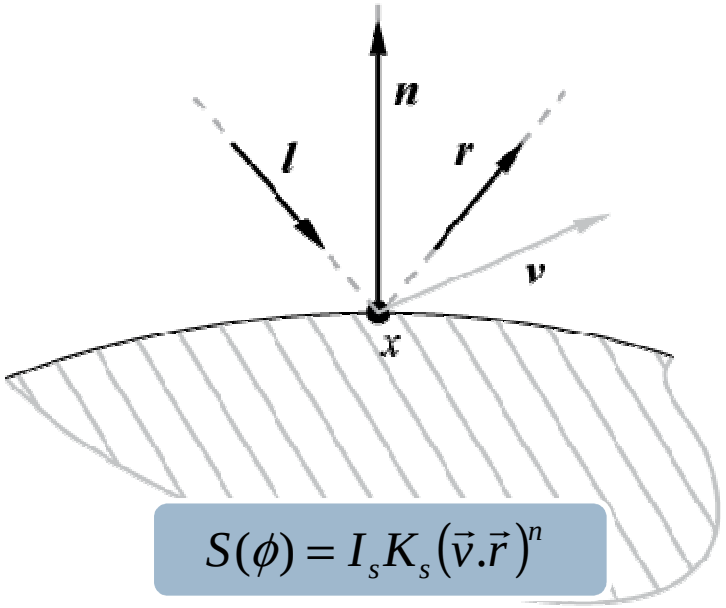


Terme spéculaire (2/2)

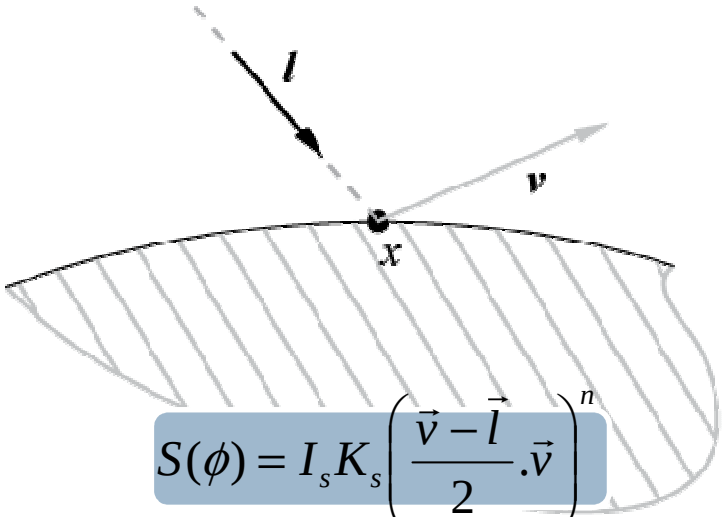


Phong & Blinn-phong (1/2)

- La formule précédente = modèle de Phong
  - Il n'est pas physiquement correct
  - Il "rend" bien en pratique et est simple à évaluer
- Une simplification possible = modèle de Blinn-Phong
  - Utilise l'angle avec le *half-vector* pour simplifier le calcul



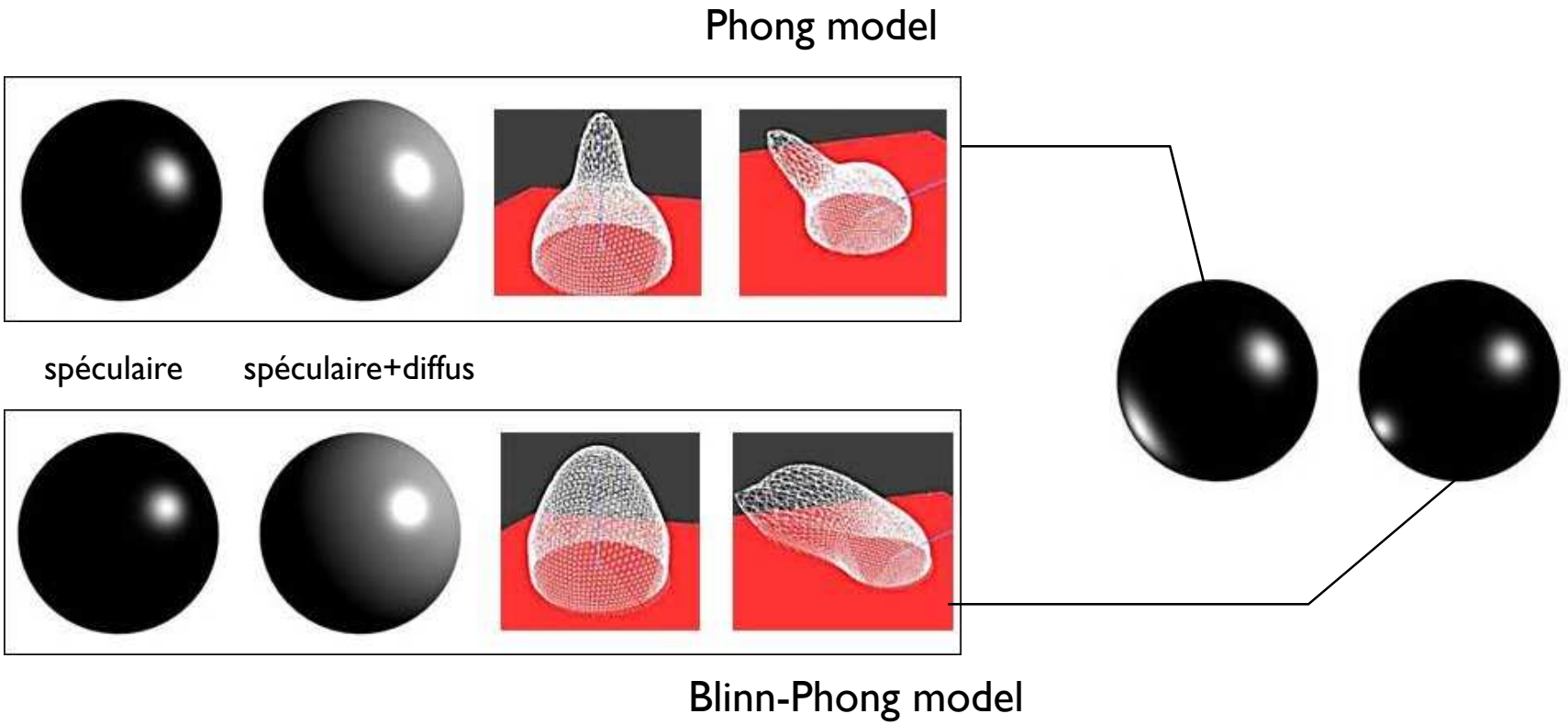
$$S(\phi) = I_s K_s (\vec{v}.\vec{r})^n$$



$$S(\phi) = I_s K_s \left(\frac{\vec{v} - \vec{l}}{2} . \vec{v} \right)^n$$

Phong & Blinn-phong (2/2)

- La différence est une question de goût!
- Blinn-phong a tendance a être plus prévisible



Remarques

- Pourquoi trois coefficients d'absorption?
  - Après tout la lumière est la même?!?!
- C'est juste un modèle
  - Le but c'est d'offrir généricité et simplicité

- On peut faire plus compliqué comme le montre le transparent suivant…

Cook-Torrance

Cook L. and Torrance K. *A Reflectance Model For Computer Graphics*. 1981

Facet slope distribution function

$$D = \frac{e^{-\left(\frac{\tan\alpha}{m}\right)^2}}{4m^2\cos^4\alpha}$$

- D est la fraction des facettes orientées dans la direction de H
- Il y a plusieurs modèles (formules) possibles
- Le plus général est celui de Beckmann (un paramètre: la pente *m*)

Specular term

$$R = \frac{F}{\pi}\frac{D}{N.L}\frac{G}{N.V}$$

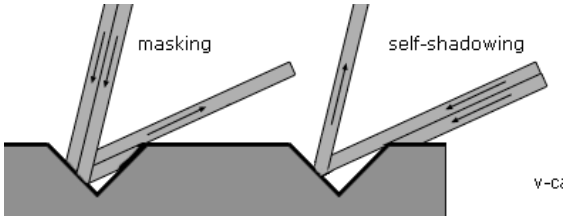
Fresnel term

$$F = (1 + E.N)^{\lambda}$$

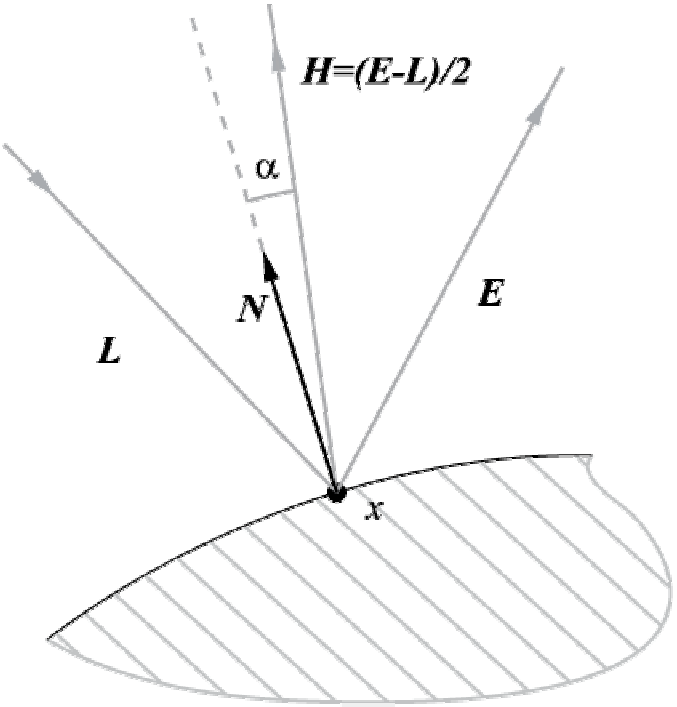
- F indique comment une micro-facette réfléchi la lumière
- Ca dépend de la longueur d'onde
- Je ne suis pas sûr de la formule!

Geometrical attenuation factor

- G modélise comment les facettes se masquent et s'ombrent mutuellement



$$G = \min\left(1, \frac{2(H.N)(E.N)}{E.H}, \frac{2(H.N)(L.N)}{E.H}\right)$$



Shading en OpenGL (1/2)

- Comment les pixels sont-ils produits?
  - CPU : appel à des fonctions de l'API pour
    - spécifier les attributs de(s) la(les) lampe(s)
    - spécifier les attributs du maillage
      - K_a, K_d, K_s, n — par "maillage" i.e. par `glBegin()/glEnd()`
      - Normale — par sommet
      - Position 3D — par sommet

par face si on spécifie la même normale aux sommets
(en exploitant le fait que OpenGL est une machine à états)

Un exemple…

```
const float Lw[4] = { 0.0f,0.0f,8.0f,1.0f };
glLightfv(GL_LIGHT0,GL_POSITION,Lw);

const float red[3]   = { 1.0f,0.0f,0.0f };
const float blue[3]   = { 0.0f,0.0f,0.5f };
const float green[3]  = { 0.0f,1.0f,0.0f };
const float yellow[3] = { 1.0f,1.0f,0.0f };
const float black[3]  = { 0.0f,0.0f,0.0f };
glMaterialfv(GL_FRONT_AND_BACK,GL_AMBIENT,blue);
glMaterialfv(GL_FRONT_AND_BACK,GL_SPECULAR,yellow);
glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION,black);
glMaterialf(GL_FRONT_AND_BACK,GL_SHININESS,20.0f);

glBegin(GL_TRIANGLES);
// A normal per face
glNormal3f(0,0,1);
glVertex2f(0,0);
glVertex2f(2,0);
glVertex2f(1,1);

// A normal per vertex
glNormal3fv(-1, 0,1);glVertex2f(0, 0);
glNormal3fv( 1, 0,1);glVertex2f(2, 0);
glNormal3fv( 0,-1,1);glVertex2f(1,-1);
glEnd();
```

Shading en OpenGL (1/2)

- Comment les pixels sont-ils produits?
  - CPU : appel à des fonctions de l'API pour
    - spécifier les attributs de(s) la(les) lampe(s)
    - spécifier les attributs du maillage
      - K_a, K_d, K_s, n
      - Normale
      - Position 3D
  - GPU : *hardware* spécialisé qui
    - projette les sommets
    - rasterize l'intérieur de la projection
    - calcule une couleur pour chaque fragment
    - combine les fragments pour obtenir les pixels

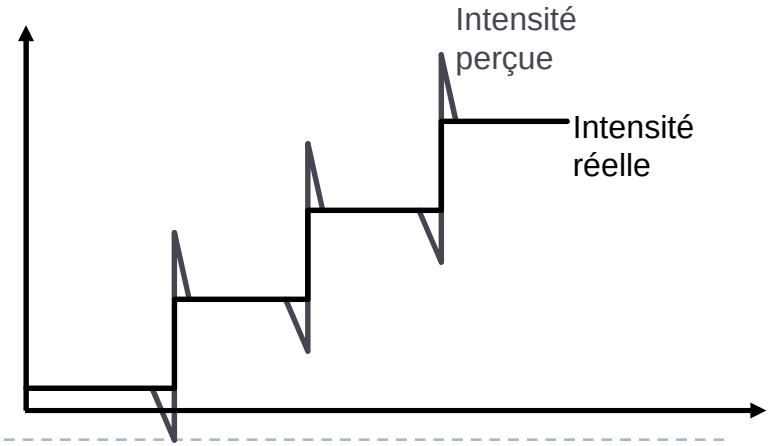
Shading en OpenGL (2/2)

- Comment les pixels sont-ils *shaded*?
  - *Flat shading*
    - Applique Phong pour produire une couleur par face
  - *Gouraud shading*
    - Applique Phong pour produire une couleur par sommet
    - Interpole la couleur pour les fragments intérieurs
  - *Phong shading* ⟷ *2 sens pour Phong!*
    - Interpole les paramètres du modèle de Phong
      - Normale
      - Vecteur vers la source (*light vector*)
    - Applique Phong pour produire une couleur par fragment

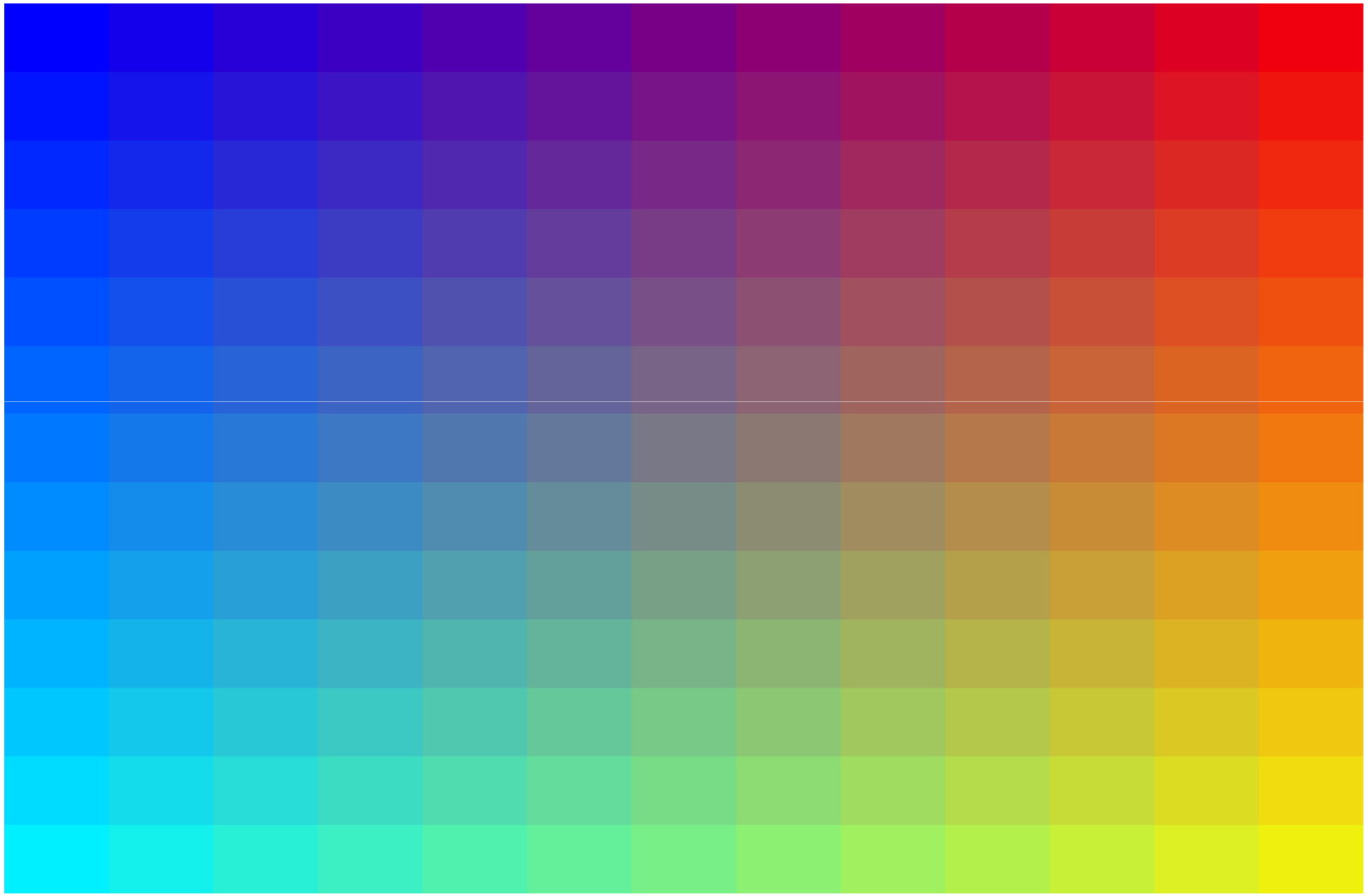
Flat *vs.* Gouraud shading

- Le *Flat shading* produit des objets "facettisés"
  - Il faut fortement tesseler les surfaces

- Le *Flat shading* produit des *Mach bands*
  - L'oeil humain "perçoit" les discontinuités d'intensité



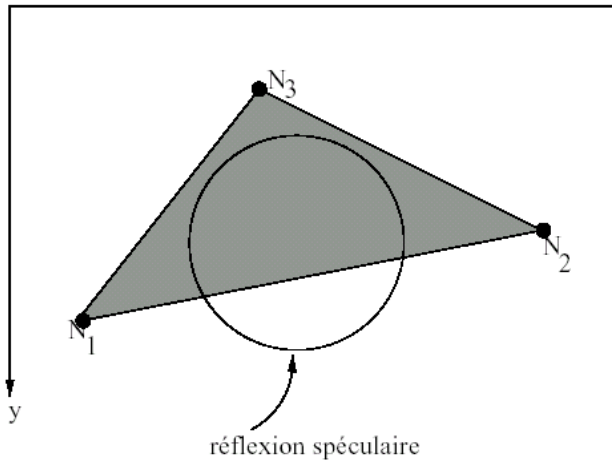
Mach Banding



Gouraud *vs.* Phong shading

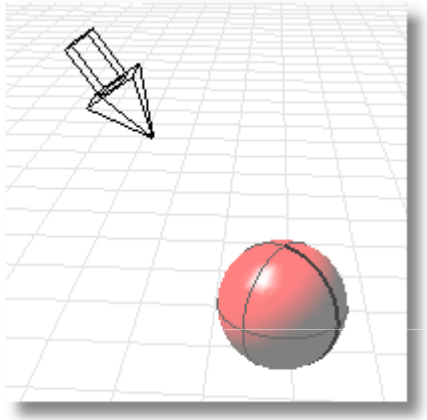
Per vertex *Per pixel*

- Phong est plus coûteux
  - En général, il y a plus de pixels que de sommets
- Phong est plus joli
  - Capture les spécularités inter-face ratées par Gouraud
  - N'est cependant pas exact
    - L'interpolation du *light vector* est une approximation

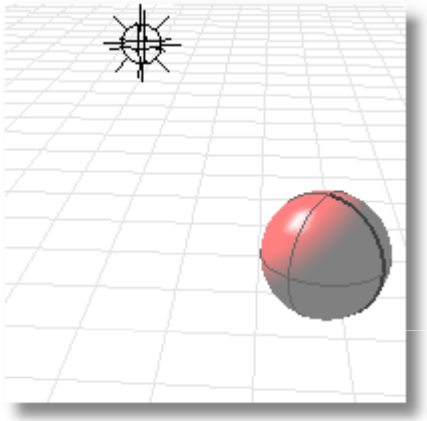


Les modèles de lampes

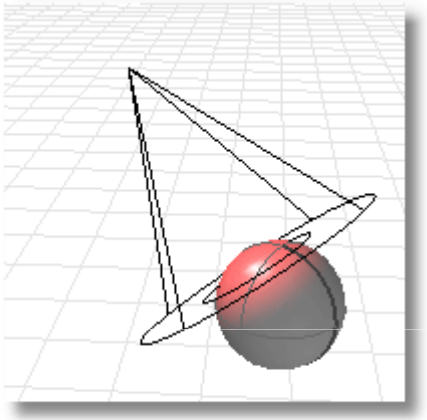
Directionnelle

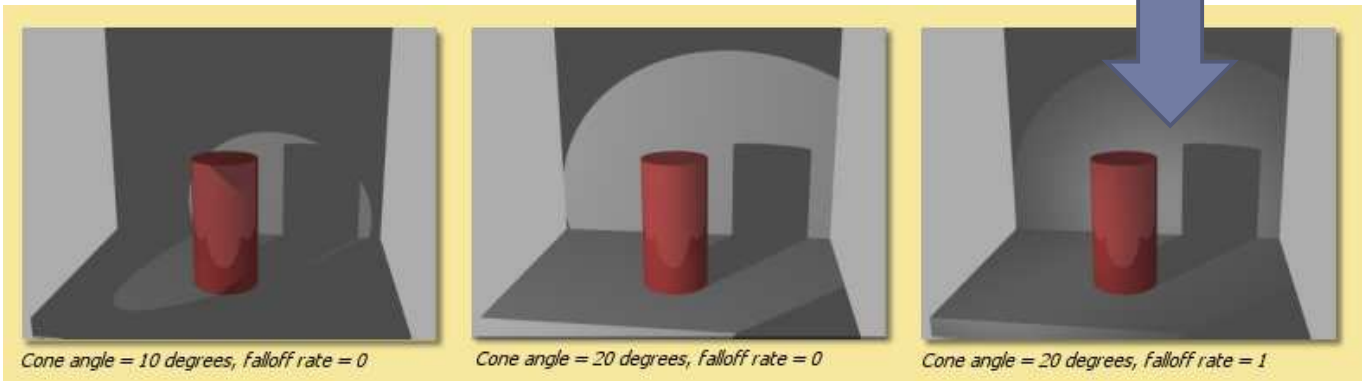


Ponctuelle

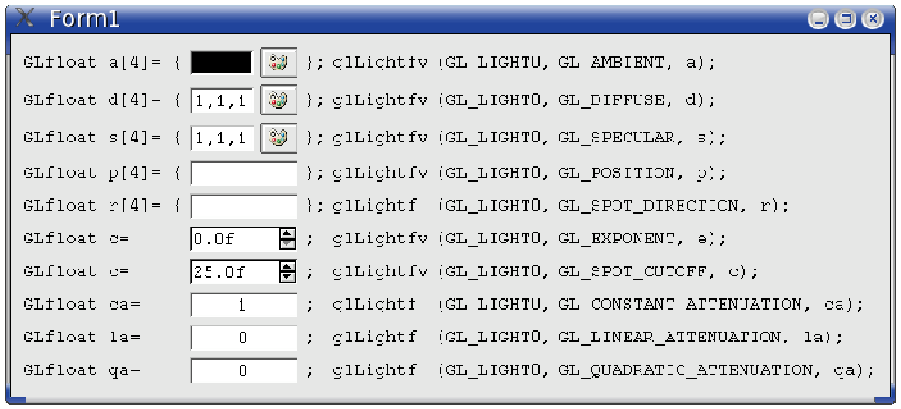


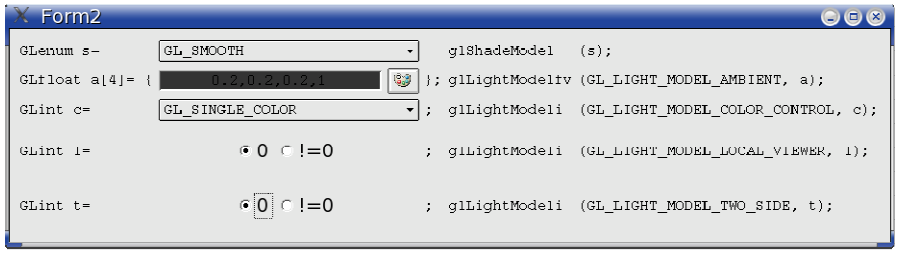
Spot

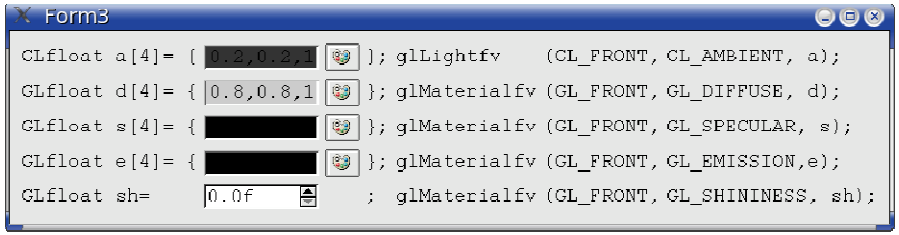




Les paramètres OpenGL

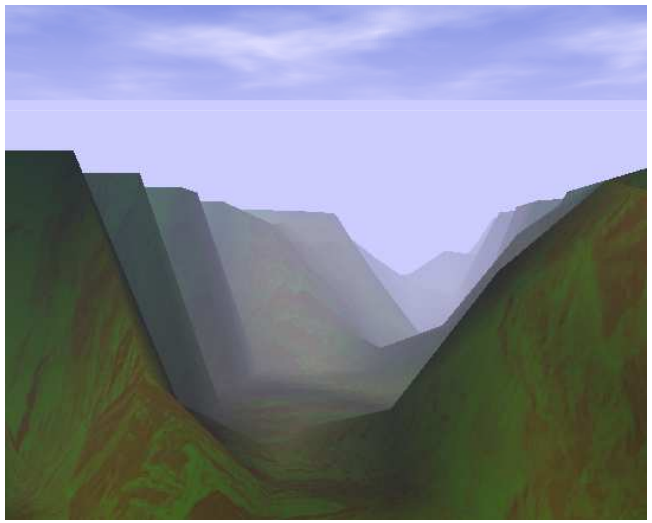






Hacks pour émuler des effets complexes

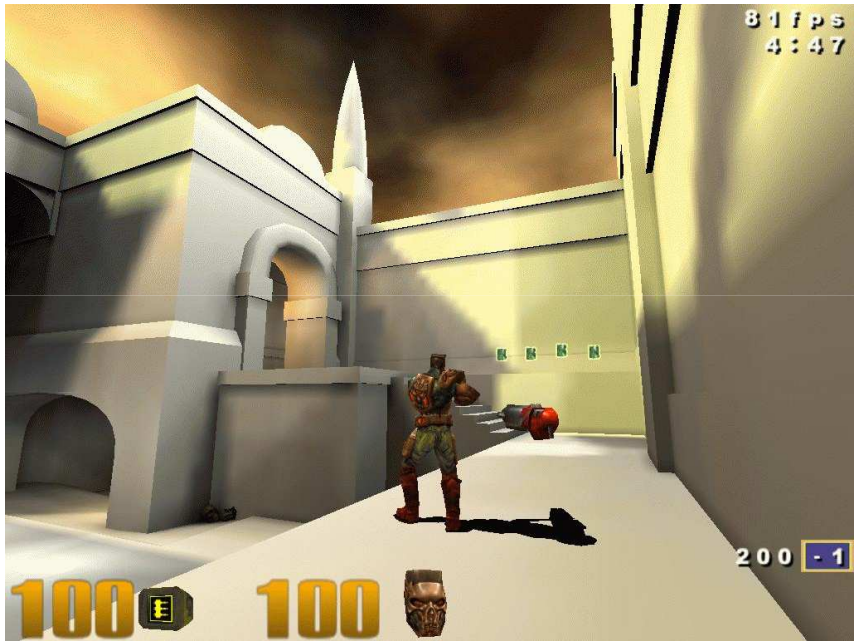
- Le terme ambient simule (sic) l'illumination globale
- Effets Atmosphériques
  - La couleur du fragment est mélangée avec le fond en fonction: *blended*
    - de la distance à l'oeil (z)
    - du modèle d'attenuation choisie
  - Ça sert d'autres buts
    - cf. cours visibilité
    - cf. cours niveaux de détails



- Textures

Textures (1/3)

Sans



Avec

Textures (2/3)

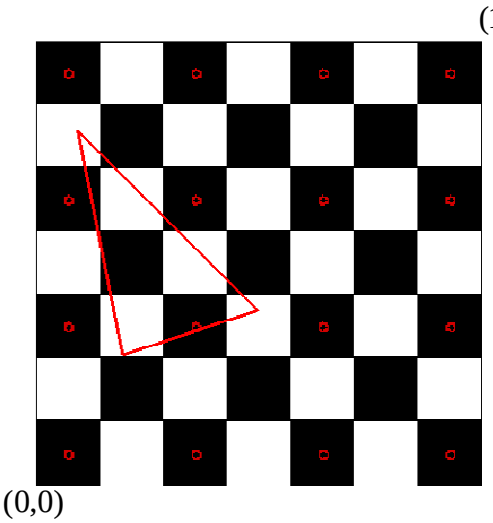
- Souvent présentées par analogie avec du papier peint que l'on peut coller sur une surface
- C'est une vision historique
- Aujourd'hui, il vaut mieux voir les textures autrement…

Textures (3/3)

~~Lookup table~~

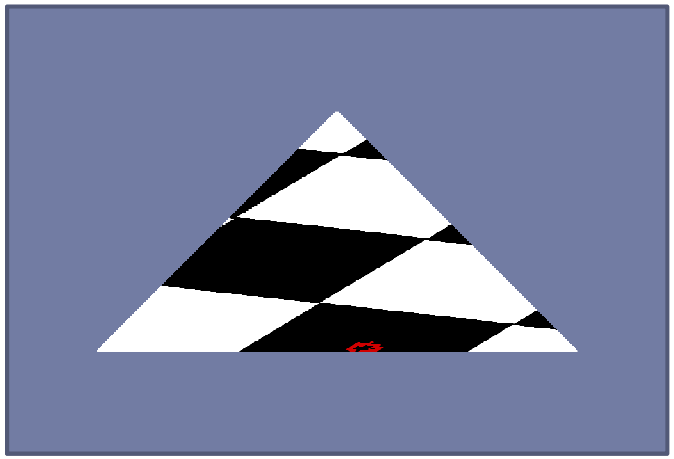
- Une texture est un champ scalaire discret sur la surface
  - Défini sur un domaine linéaire/rectangulaire/cubique
  - Mappé sur la surface par des coordonnées textures
    - Spécifiées en chaque sommet `glTexCoord{123}{fi}`
    - Interpolées pour chaque fragment

1D 2D 3D



domaine

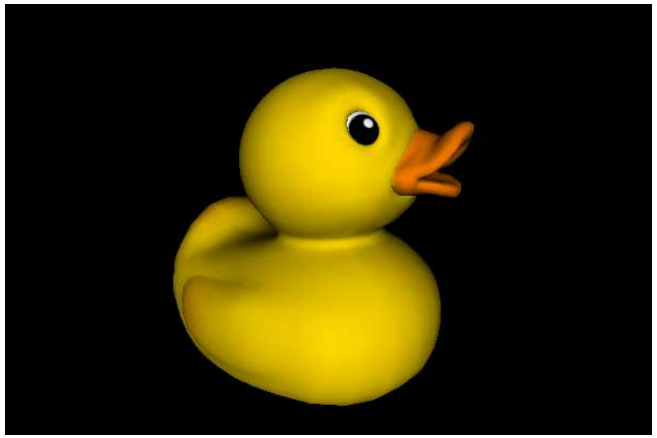
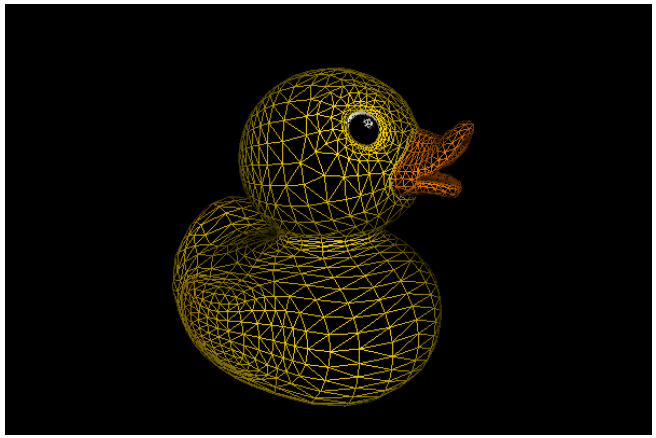
```
glBegin(GL_TRIANGLES);
glTexCoord2f(0.2f,0.3f);
glVertex2f(-1.0f,0.0f);
glTexCoord2f(0.5f,0.4f);
glVertex2f(+1.0f,0.0f);
glTexCoord2f(0.1f,0.8f);
glVertex2f( 0.0f,1.0f);
glEnd();
```



Vue 3D

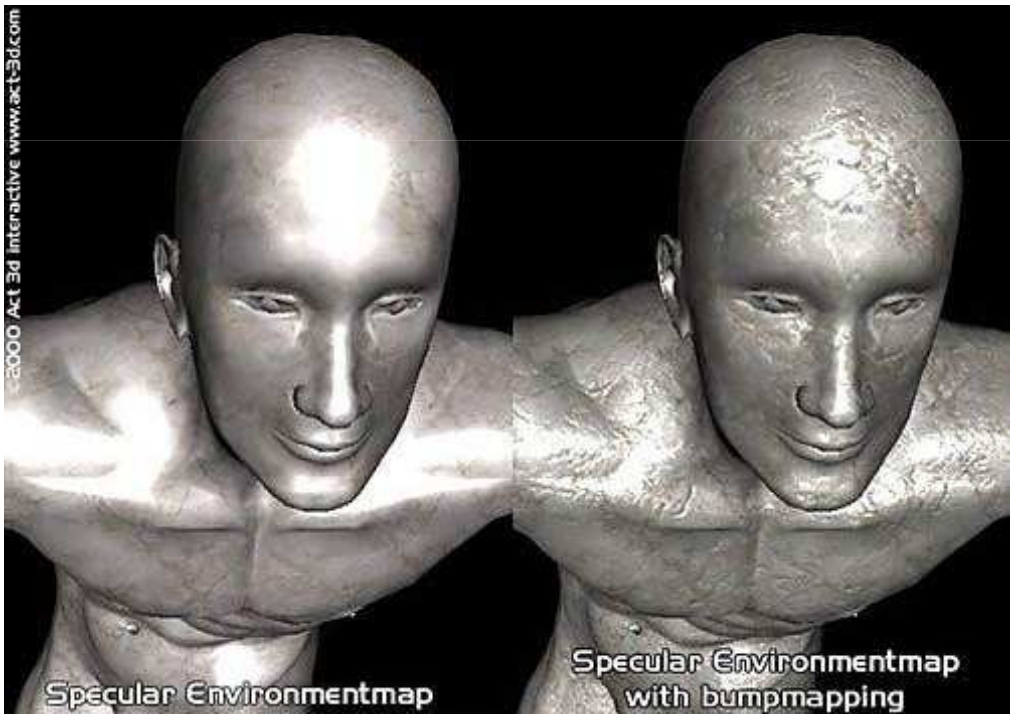
Textures (3/3)

- On peut l'utiliser pour spécifier:
  - La couleur ambient/diffuse

Textures (3/3)

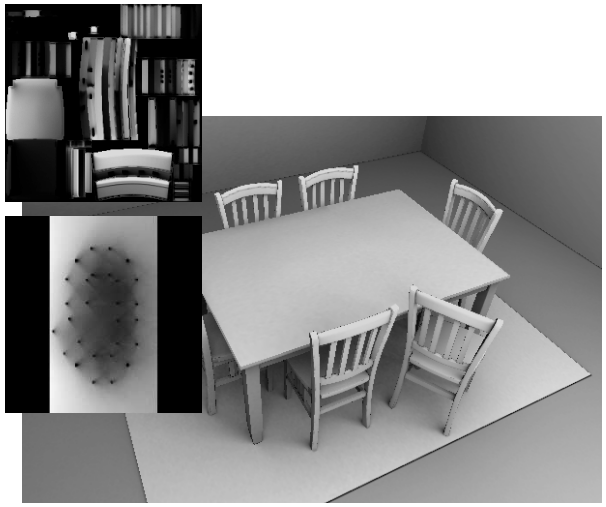
- On peut l'utiliser pour spécifier:
  - La couleur ambient/diffuse
  - La normale (*bump mapping*)



Textures (3/3)

- On peut l'utiliser pour spécifier:
  - La couleur ambient/diffuse
  - La normale (*bump mapping*)
  - Un illumination précalculée (*light map*)

Diffuse mappée sur la scène



Lightmap mappées sur la scène

Combinées

Textures (3/3)

- On peut l'utiliser pour spécifier:
  - La couleur ambient/diffuse
  - La normale (*bump mapping*)
  - Un illumination précalculée (*light map*)
  - Ce qu'on veut en fait (cf. cours sur les *shaders*)

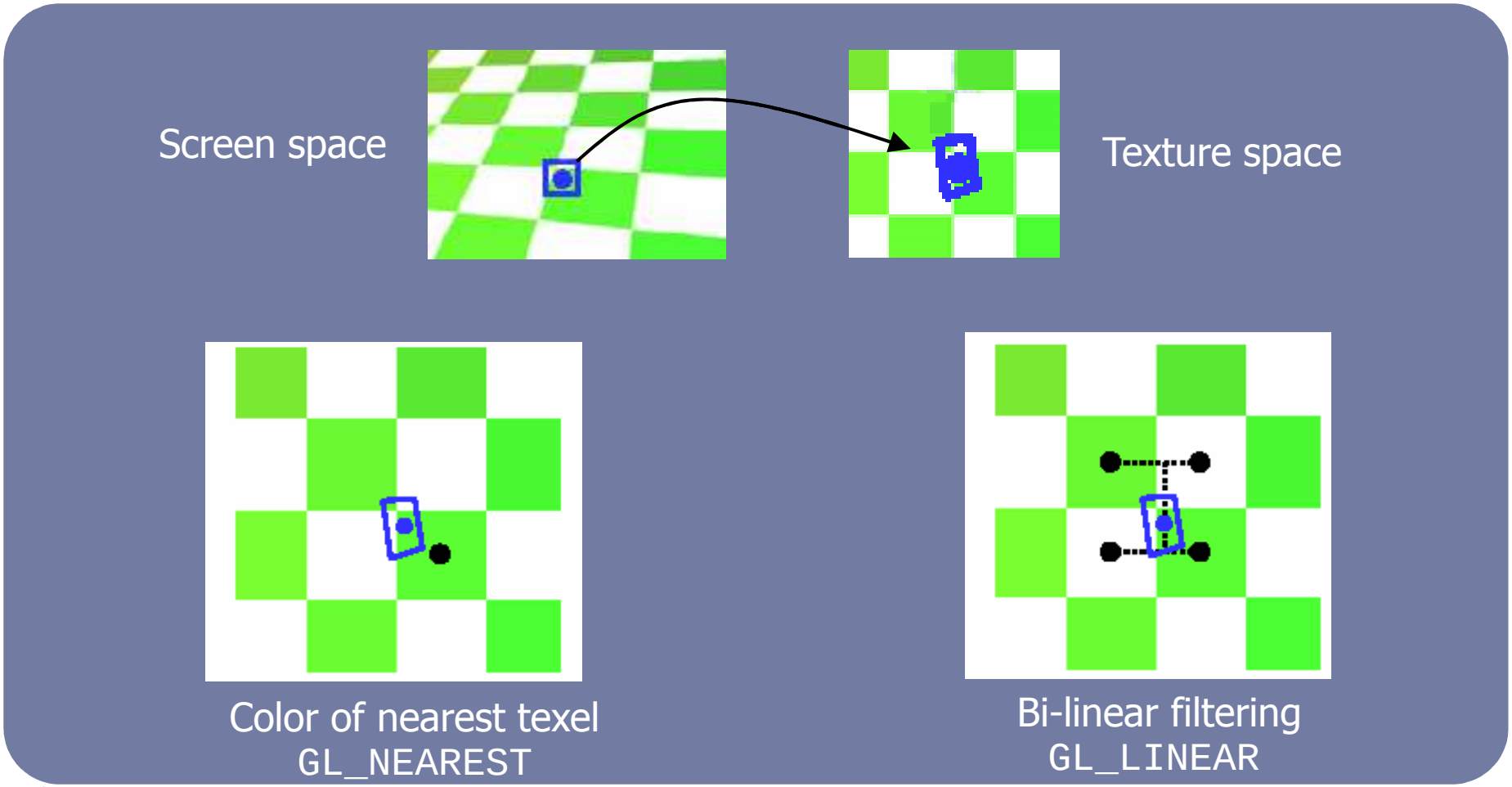
Texture mapping & Aliasing

- Undersampling *magnification*
  - Prendre le plus proche ou interpoler les voisins
- Supersampling *minification*
  - Solution idéale
    - Integration des échantillons samples → coûteux
  - Solution pratique
    - Pré-filtrer les textures → *mipmaps*
    - Interpoler entre les niveaux (les voisins)

Undersampling

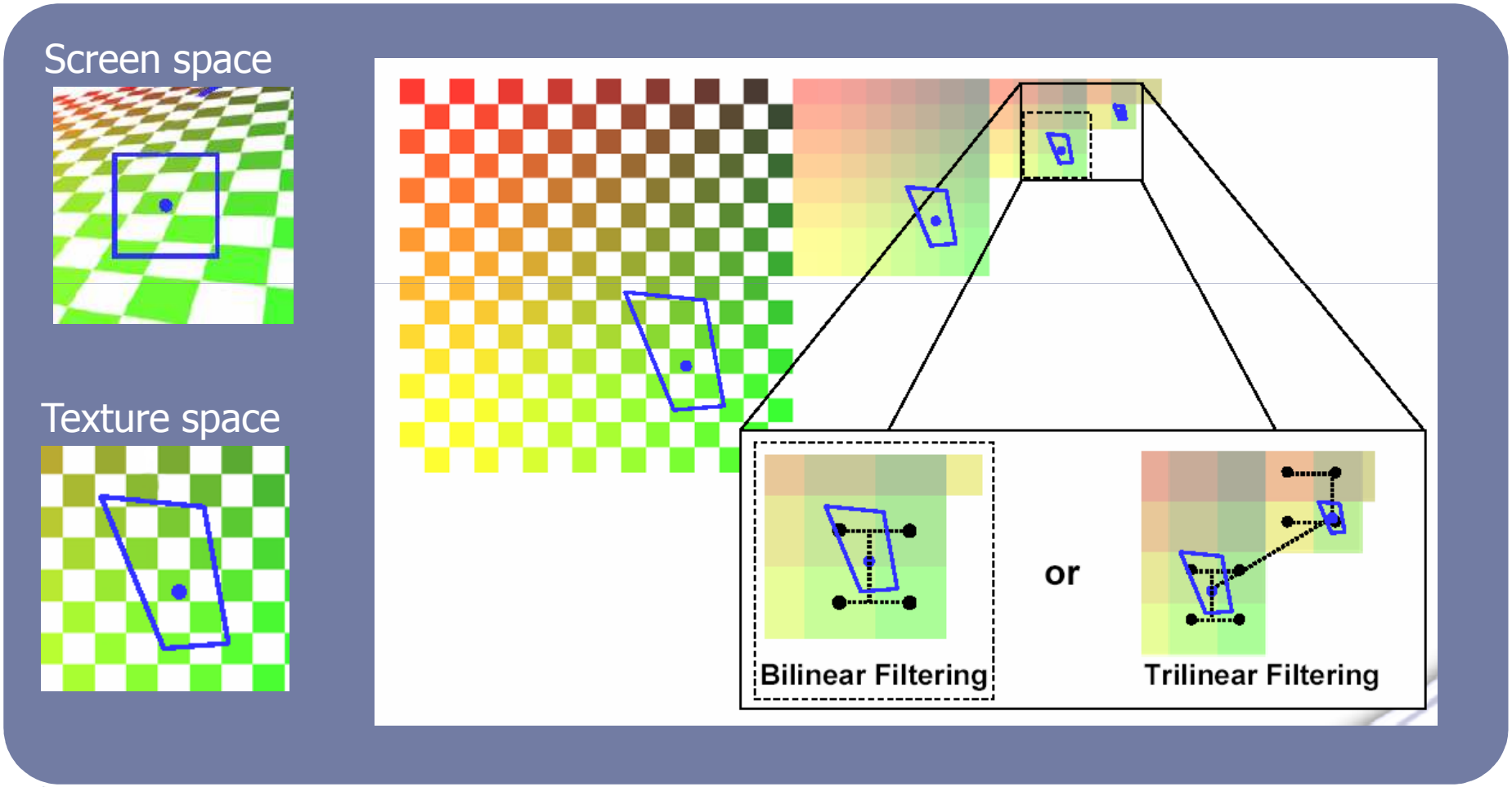
picture element *texture element*

- Pixel "plus petit" qu'un texel



Supersampling

- Pixel "plus grand" qu'un texel



C’est fini pour aujourd’hui

C’est l’heure de la pause