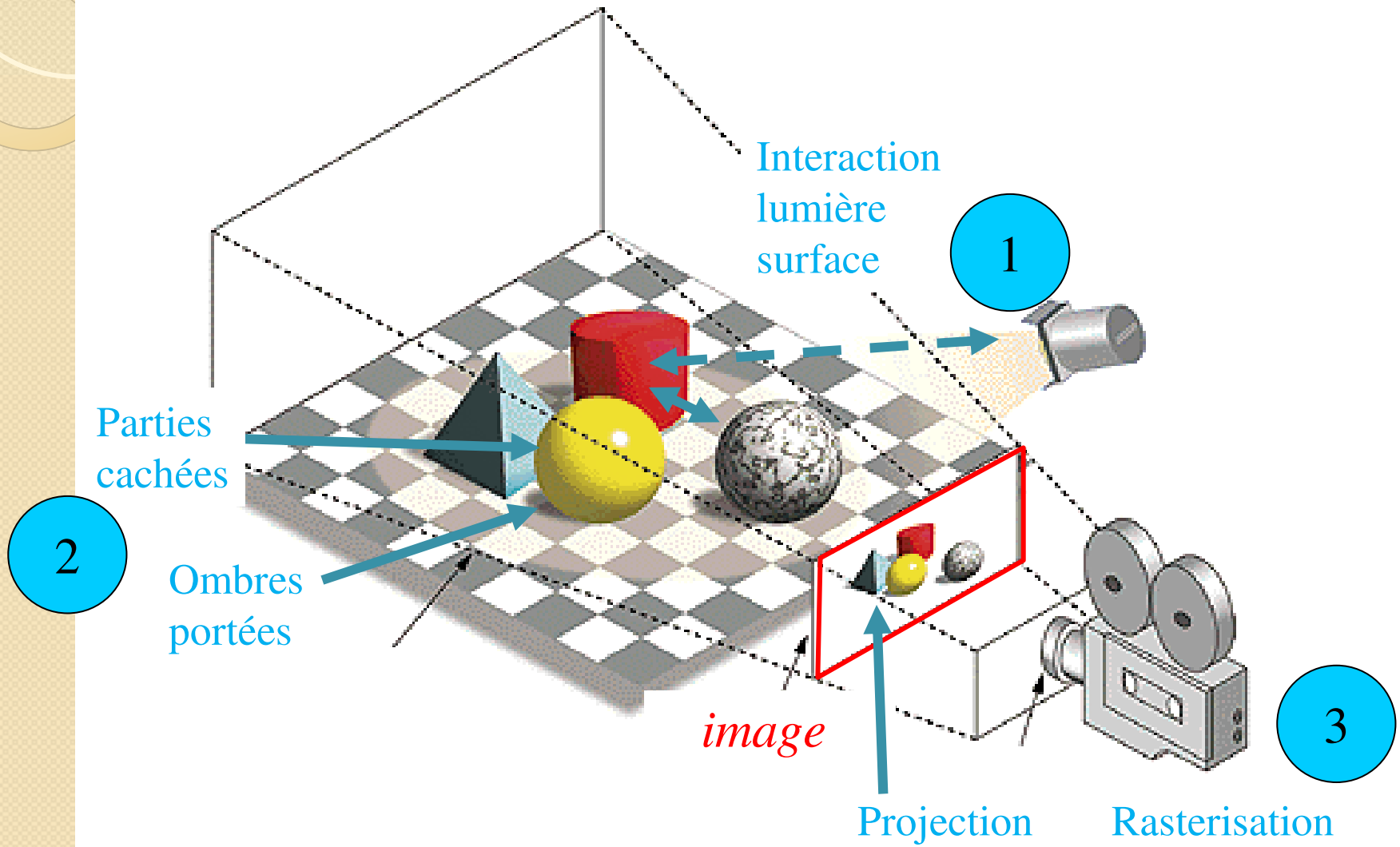
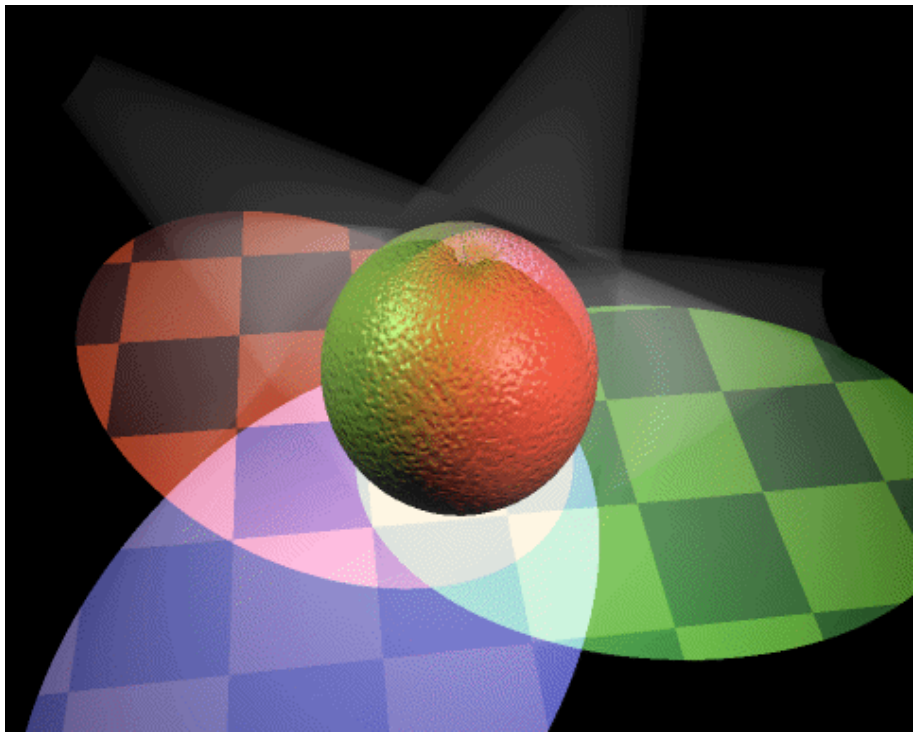
LE RENDU

L'image


Interaction
lumière
surface
1
Parties
cachées
2
Ombres
portées
image
3
Projection
Rasterisation

Le rendu (shading)

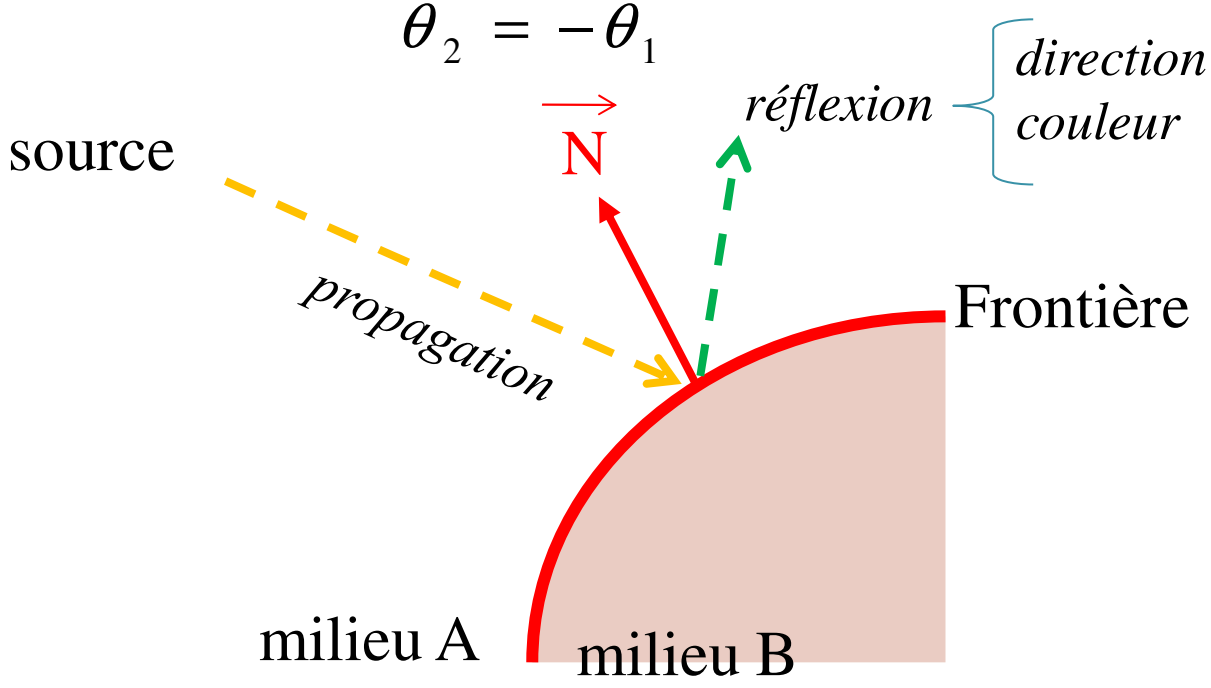
1. Modèle d'éclairage
   1. Rappels physiques
   2. Sources de lumière
   3. Modèles de Réflexions
   4. Milieu
2. Insuffisances du modèle géométrique
   1. Flat shading
   2. Gouraud shading
   3. Phong shading
3. Texture
4. Ray tracing
5. Parties cachées



1.1 Propagation de la lumière

Les lois de Snell-Descarte

$$\theta_2 = -\theta_1$$



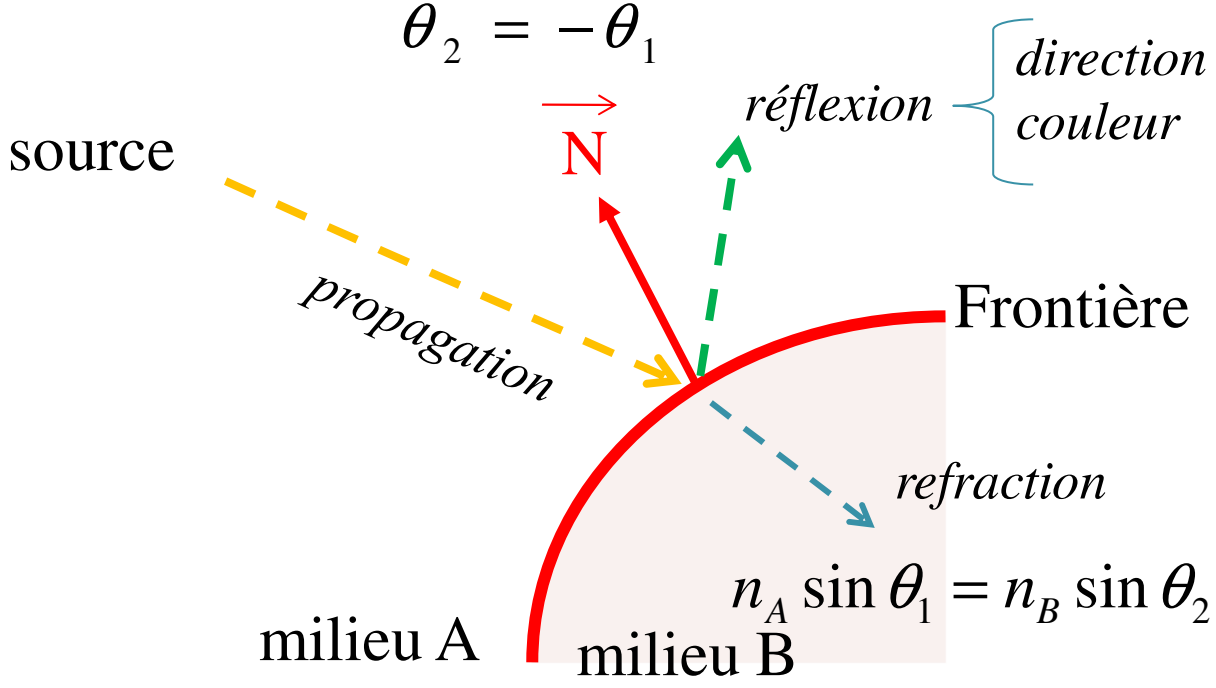
La lumière est l'ensemble des ondes électromagnétiques visibles par l'œil humain

λ = 800 nm  λ = 380 nm

Propagation de la lumière

Les lois de Snell-Descarte

$$\theta_2 = -\theta_1$$



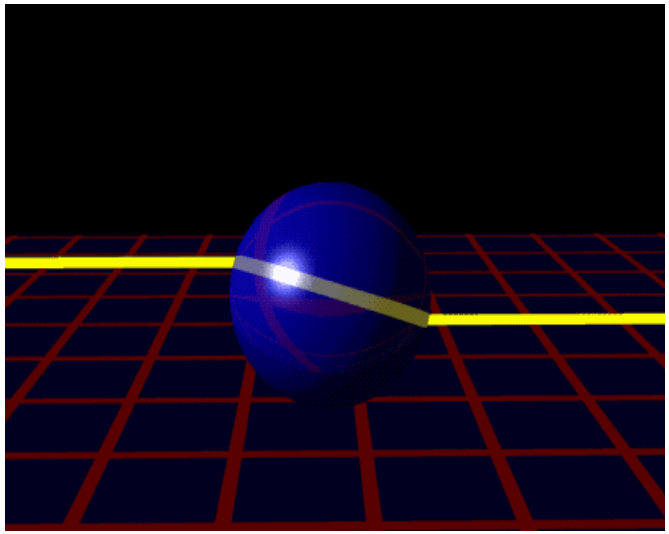
La lumière est l'ensemble des ondes électromagnétiques visibles par l'œil humain

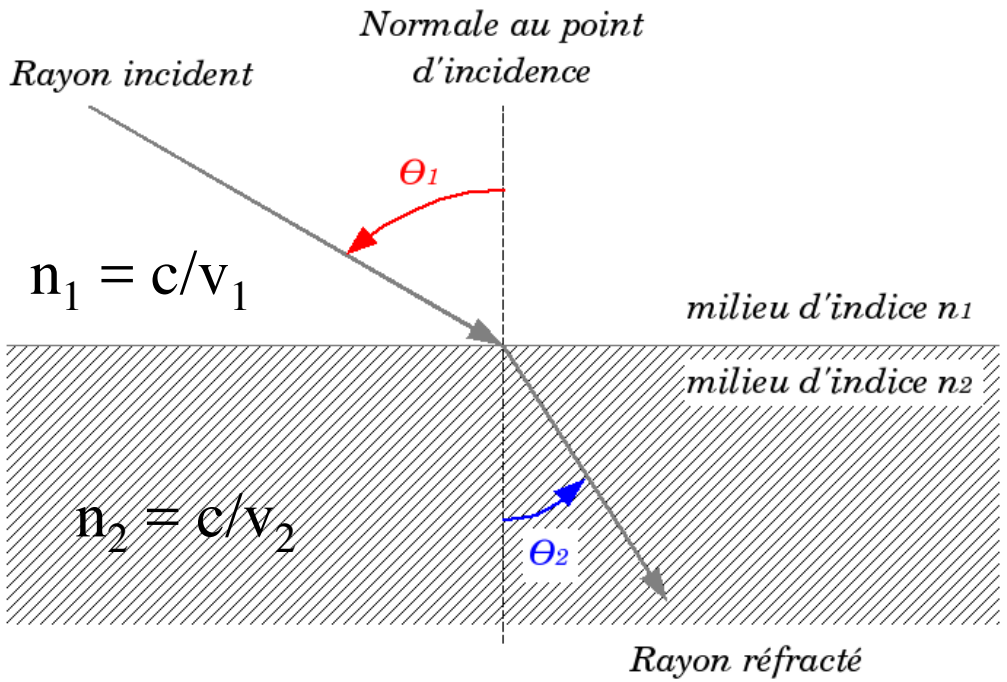
λ = 800 nm  λ = 380 nm

Transparence, réfraction

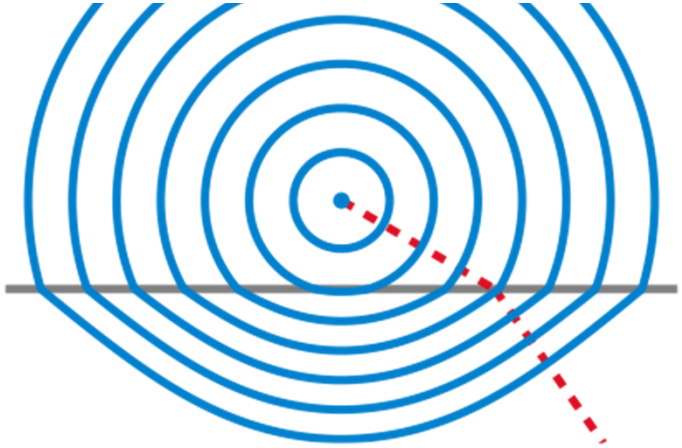
Loi de Snell-Descartes

$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$





n dépend de λ, T, structure : matériaux biréfringents



Transparence, réfraction

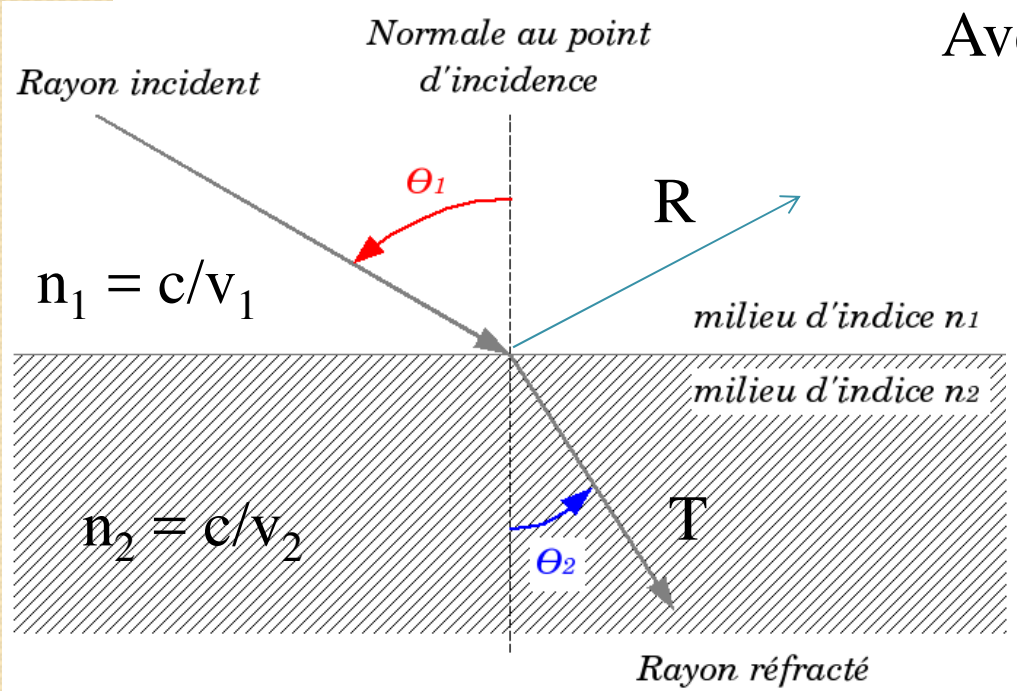
Conservation de l'énergie

T = 1 – R
T : Transmittance
R : Réflectance

Approximation de Schlick :

$$R(\theta_1) = R_0 + (1 - R_0)(1 - \cos\theta_1)^5$$

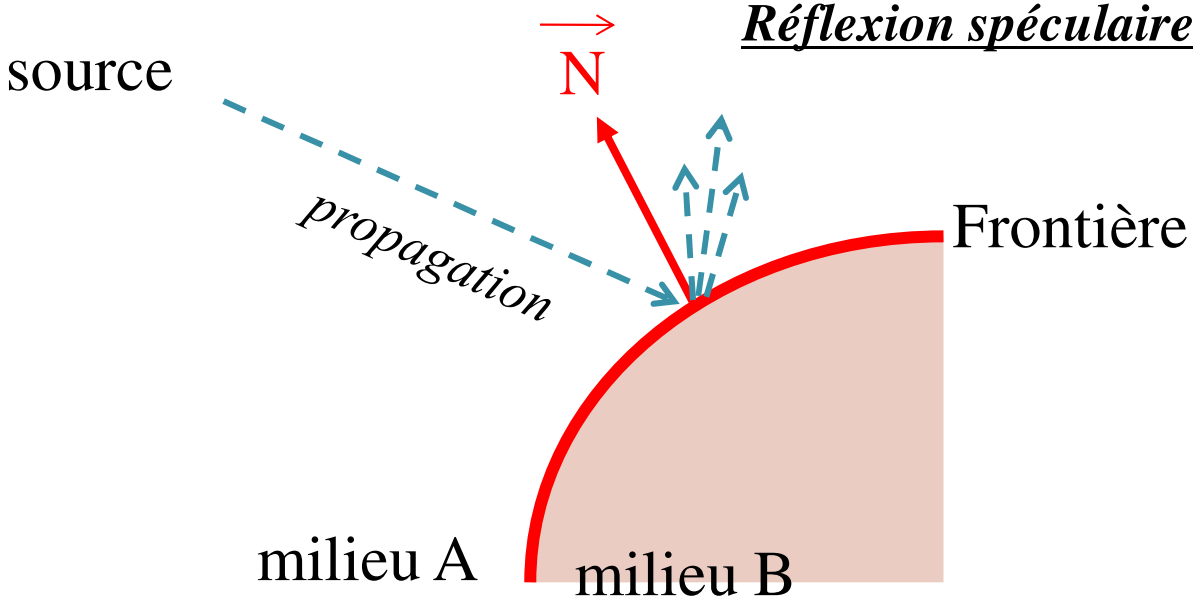
Avec $R_0 = ((n_1 - n_2)/(n_1+n_2))^2$



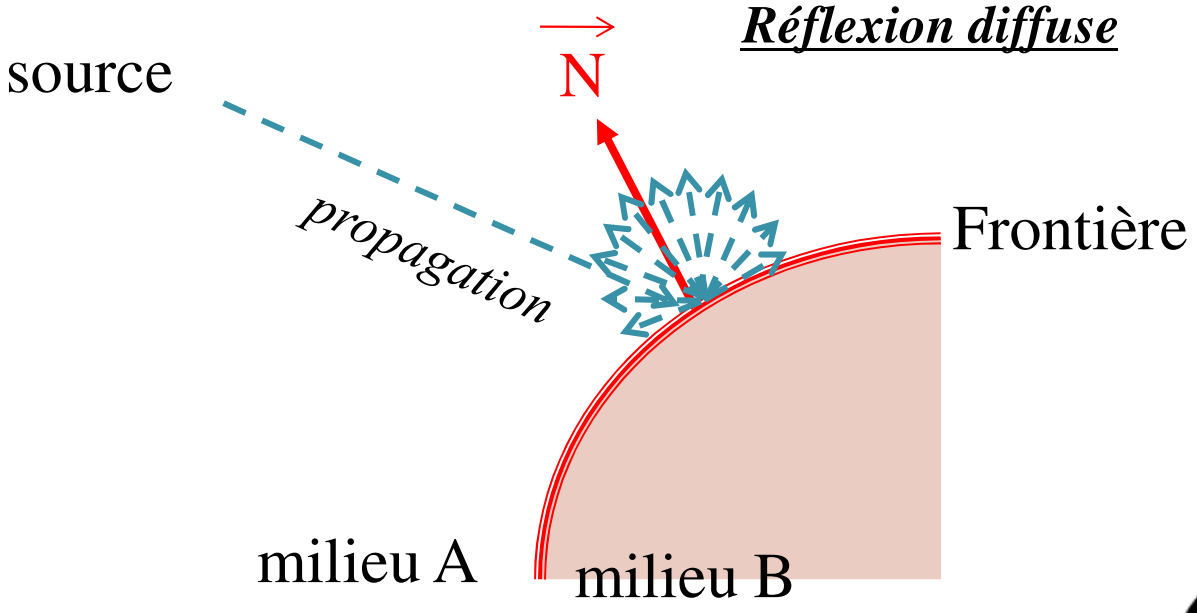
Voir les équations de Fresnel :
R dépend de l'onde (polarisation)
…

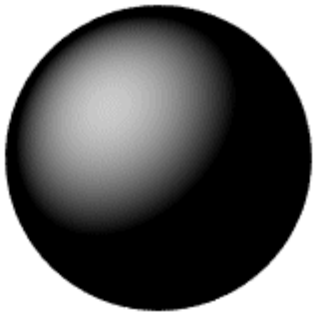
Réflexion spéculaire

$$\theta_2 = -\theta_1$$

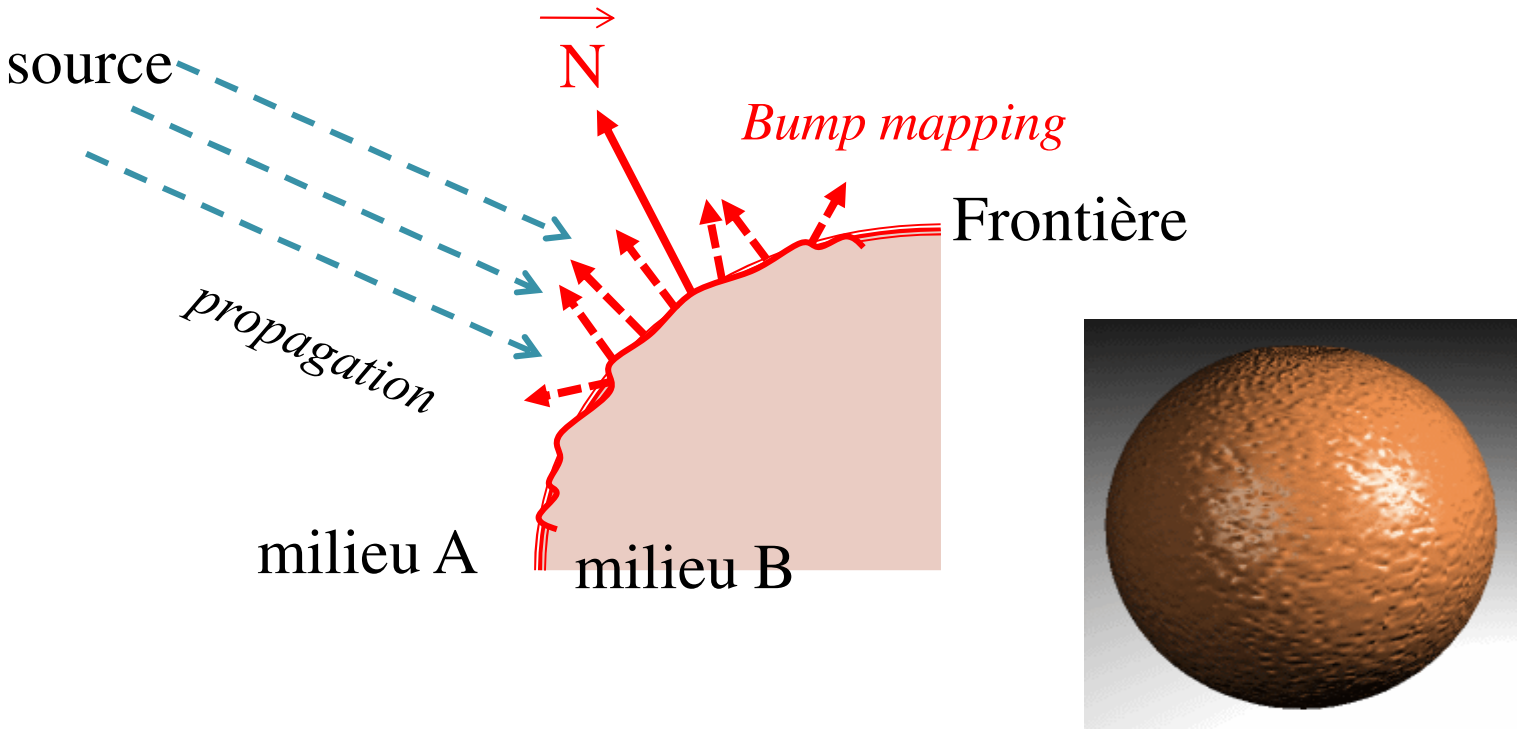


Réflexion diffuse



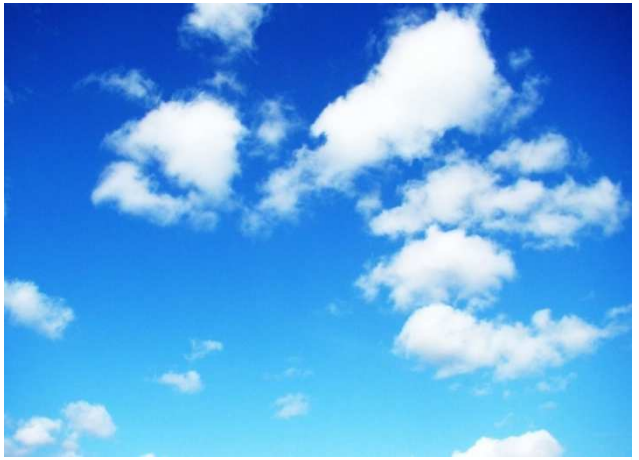


Texture ➔ Bump mapping



Réflexion spéculaire ou diffuse





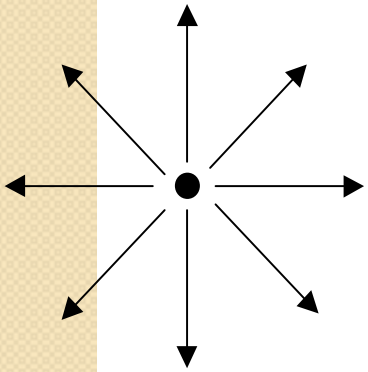
I.2 Source de lumière

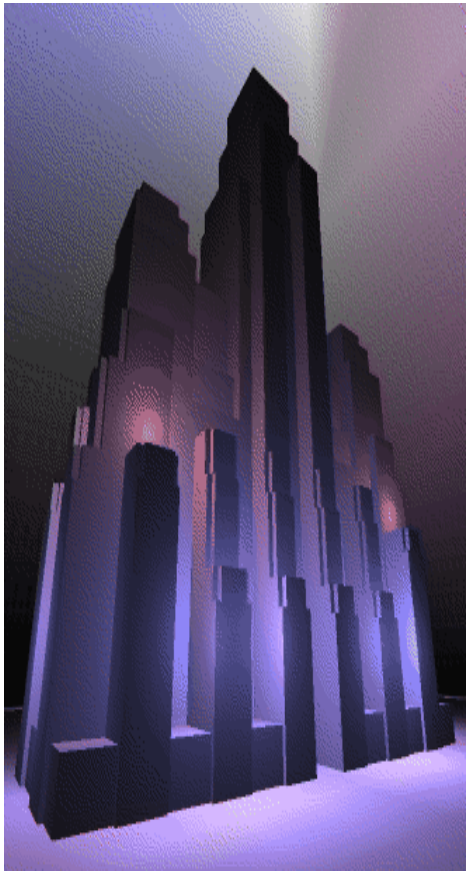


Sources ponctuelles

Isotrope

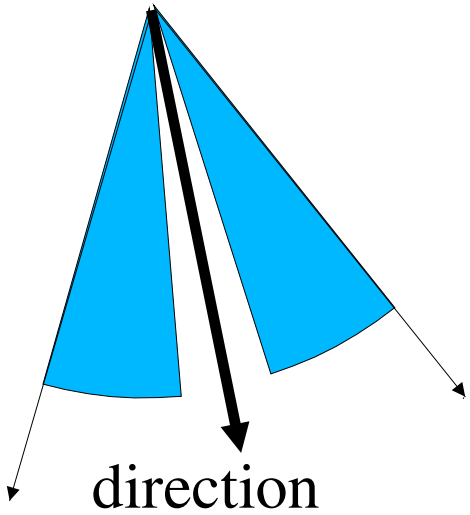
Atténuation en fonction de la distance

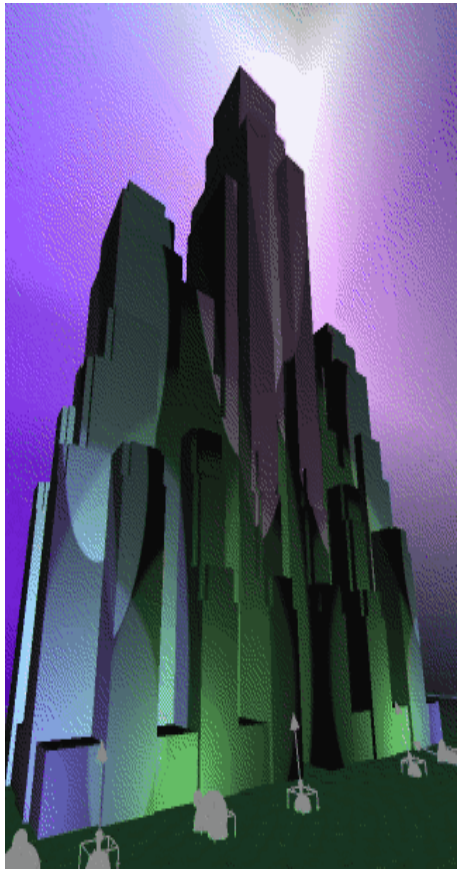




Directionnelle

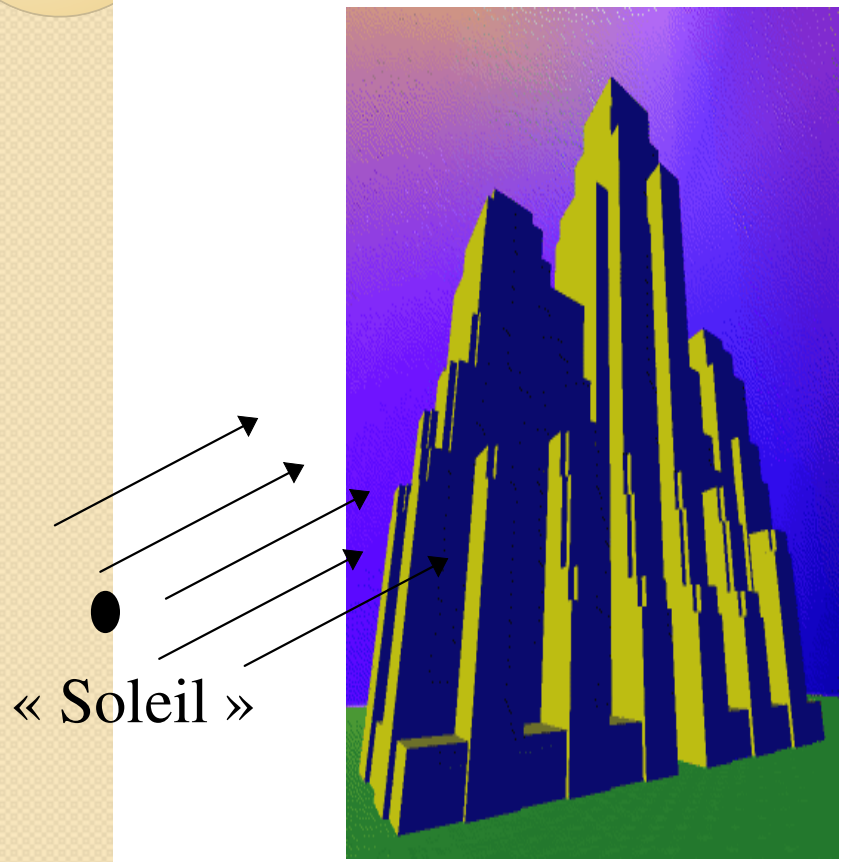
Atténuation en fonction de la direction





Autres sources

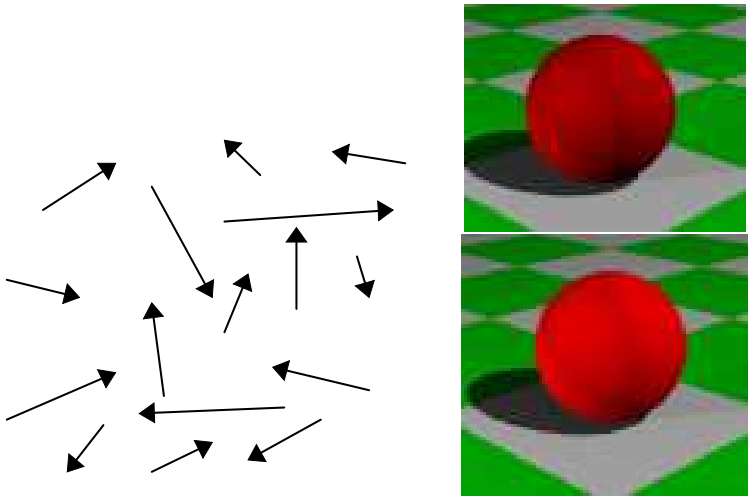
Source directionnelle



Simplification

« Eclairage » ambiant

Isotrope et Uniforme

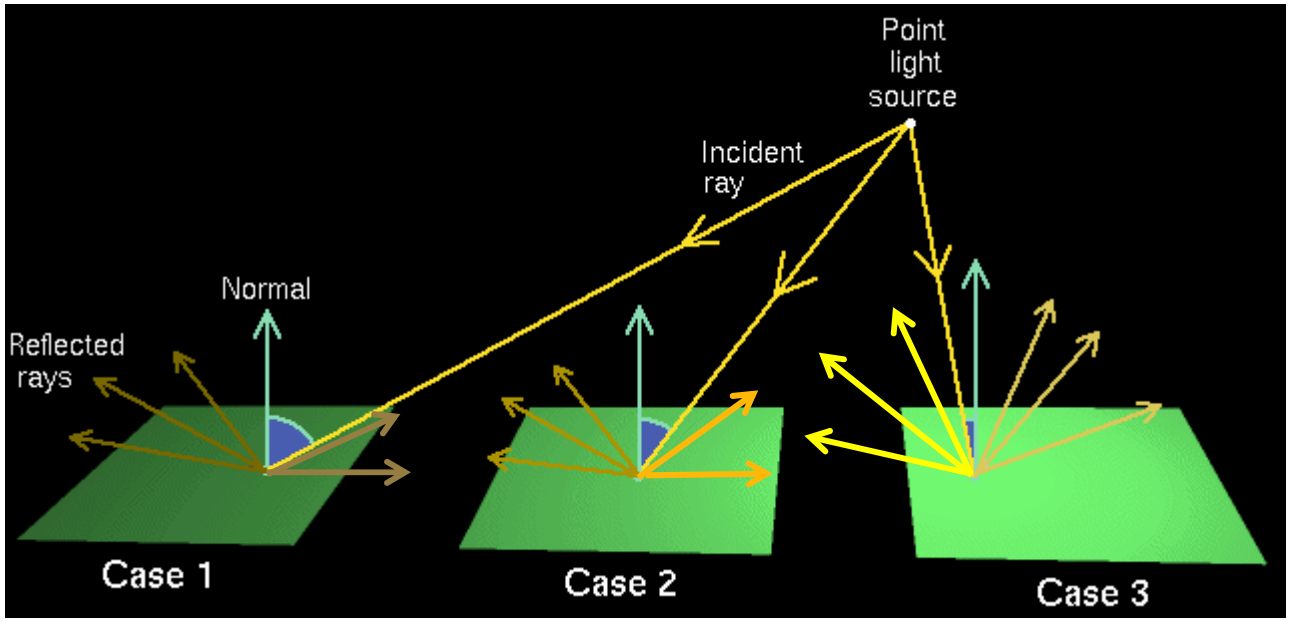


Simplification

On ne peut pas la négliger…

I.3. Modèles de réflexions/surfaces

Réflexion diffuse

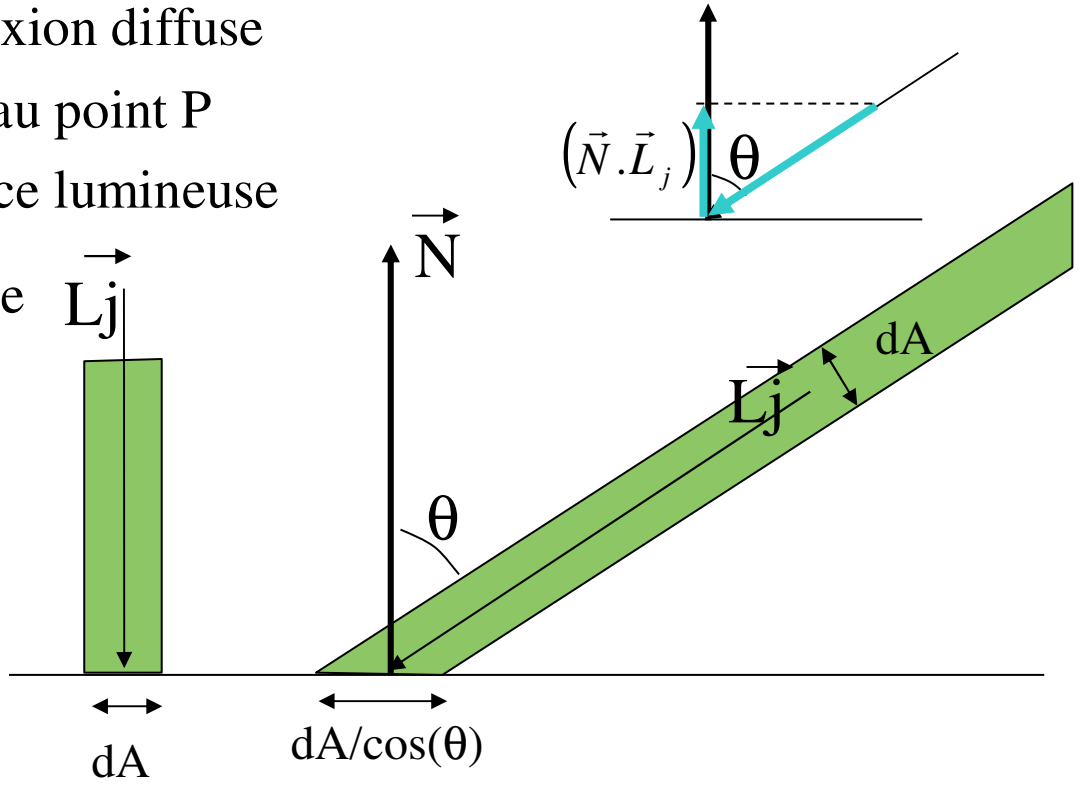


- La **réflexion diffuse** dans toutes les directions (indépendant de l'observateur)
- L'intensité dépend de l'angle entre la source et la normale à la surface
- Réaliste pour les matériaux mats (« rugosité fine » de surface)

Réflexion diffuse : Loi de Lambert

$$I_d = \max\left(-K_d \sum_{sources} \langle \vec{N}.\vec{L_j} \rangle I_j, 0\right)$$

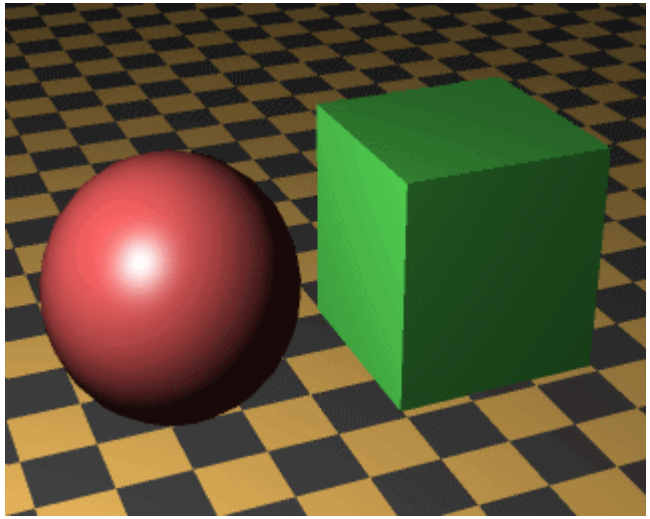
- K_d : coefficient de réflexion diffuse
- N : normale à la surface au point P
- L_j : direction de la source lumineuse
- I_j : intensité de la source

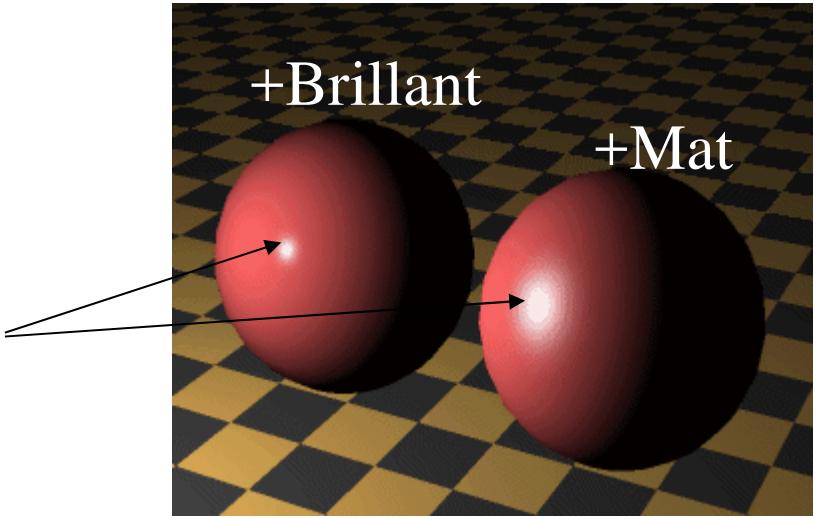


Réflexion spéculaire

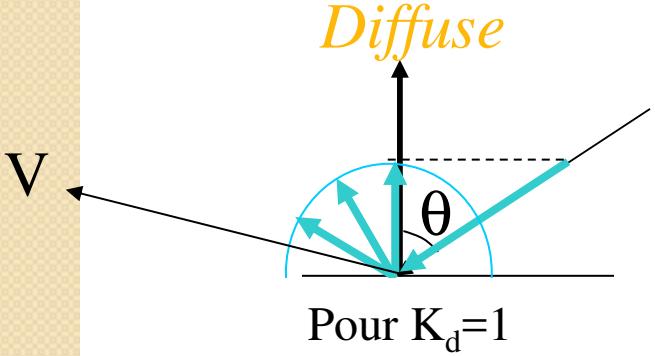
spéculaire : relatif au miroir…

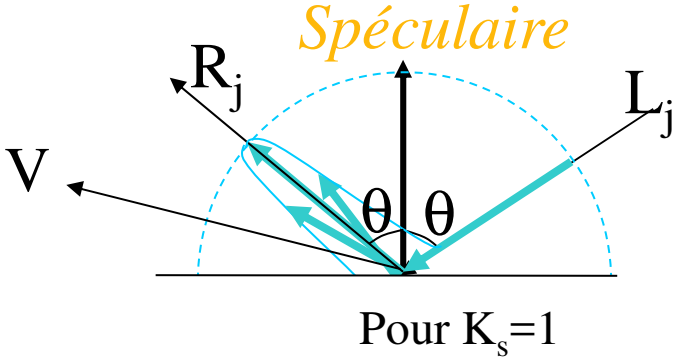
ici le reflet de la source de lumière





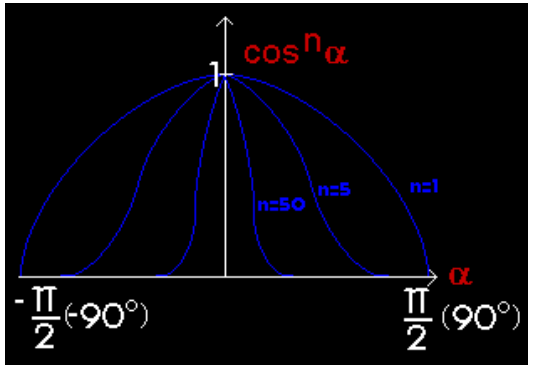
Pour un observateur dans la direction V





Intensité spéculaire

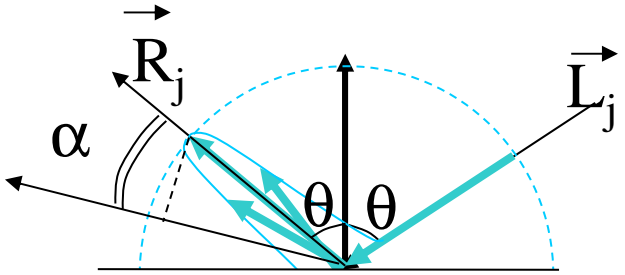
Phong $$I_s = \sum_{sources} K_s \langle \vec{R_j}.\vec{V} \rangle^n I_j$$



n dépend du type de matériau

1 : mat, 200 : brillant

C'est l'intensité vue ➔

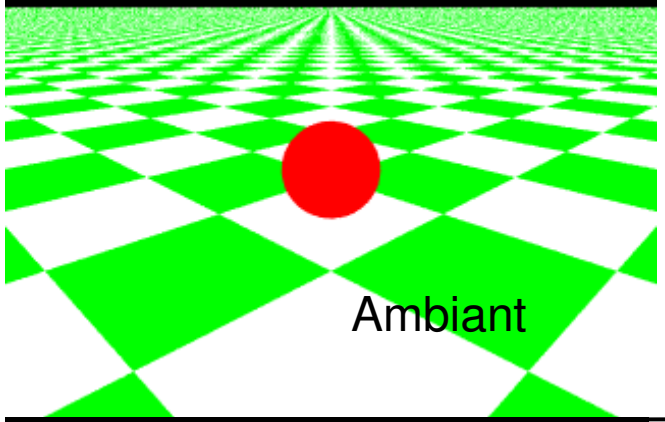


$$\vec{R_j} = \vec{L_j} - 2\langle \vec{N}.\vec{L_j} \rangle \vec{N}$$

Pour K_s=1

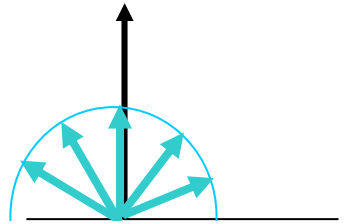
Réflexions

Diffuse la lumière ambiante



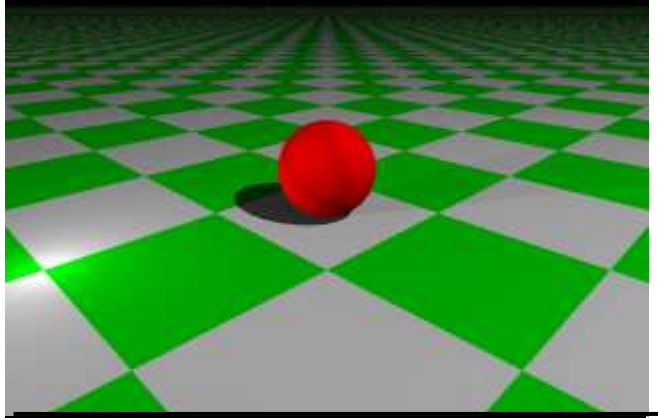
« Radiosité simplifiée »

$$I_i = K_{ai} I_a$$



Pour K_{ai}=1

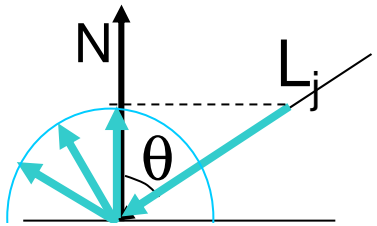
Réflexion diffuse



Matériau Mat

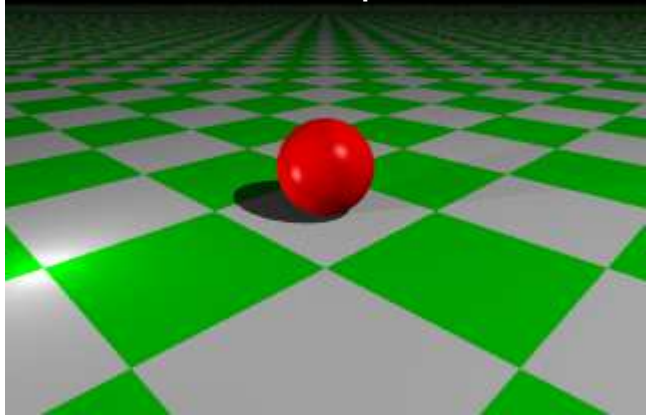
$$I_d = K_d \sum_{sources} (\vec{N}.\vec{L_j})\, I_j$$

N normale à la surface
Lj rayon de la lumière j



Pour K_d=1

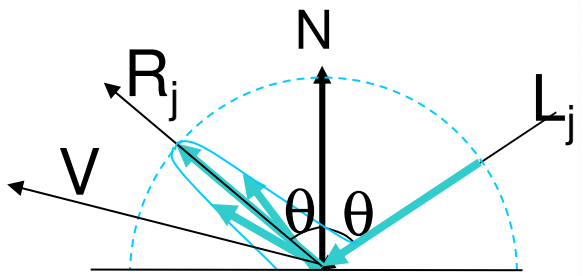
Réflexion spéculaire



Brillant/Lisse

$$I_s = K_s \sum_{sources} \langle \vec{R_j}.\vec{V} \rangle^n I_j$$

$$\vec{R_j} = \vec{L_j} - 2\langle \vec{N}.\vec{L_j} \rangle \vec{N}$$

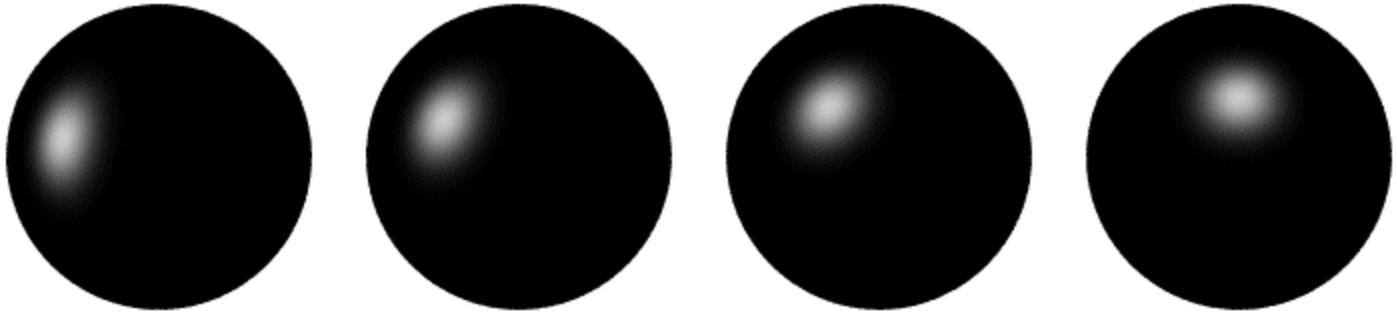


V direction de l'observateur

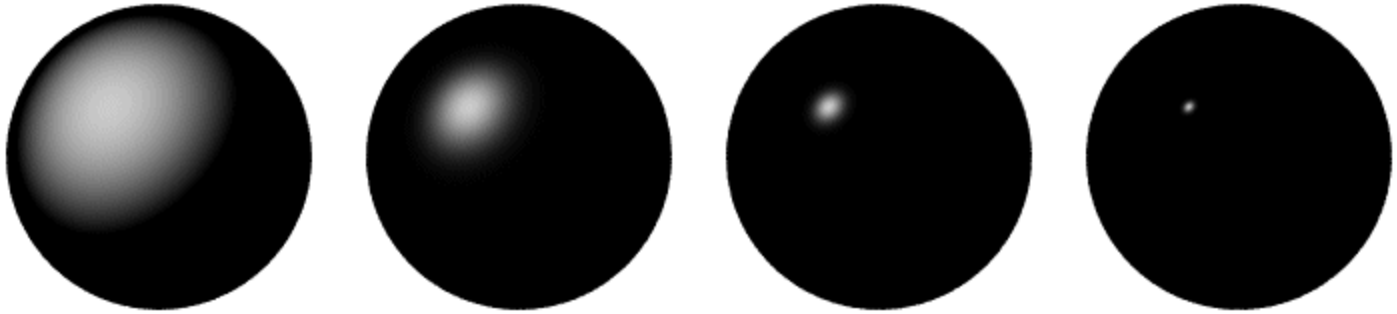
Pour K_s=1

Réflexion spéculaire
(Modèle de Phong)

$$I_s = K_s \sum_{sources} \langle \vec{R_j}.\vec{V} \rangle^n I_j$$



Si on déplace une source lumineuse ou…



Si on change la brillance (n)

Matériaux

Métal : $K_a = 0.2$, $K_d = 0.7$
K_s =0.75 n=80 …

Verre : $K_a = 0$, $K_d = 0$
K_s =1 n=400…

Nuage : $K_a = 0.7$, $K_d = 0$
$K_s = 0$…

Bois : $K_a = 0.$, $K_d = 0.3$
K_s =0.2 n=10…

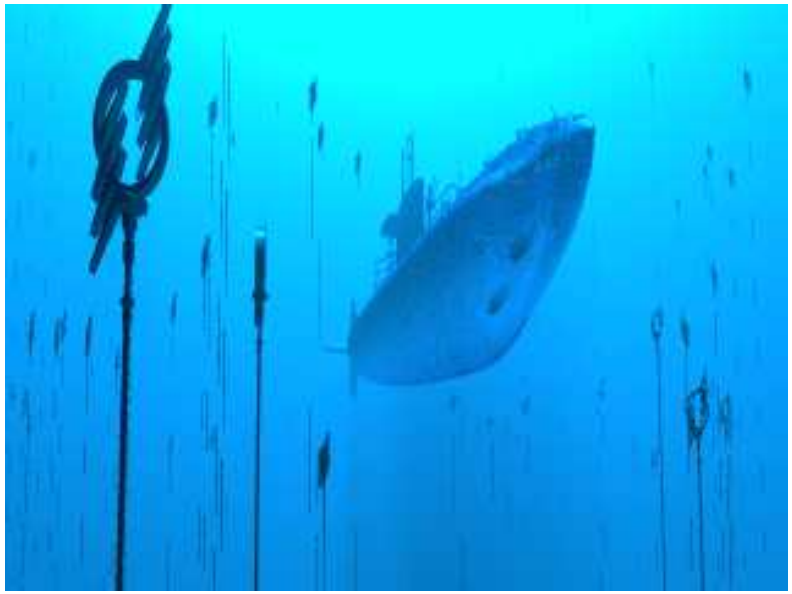
Exemple de fichier (.mtl)

```
# Blender MTL File: 'None'
# Material Count: 2

newmtl Material
Ns 96.078431
Ka 1.000000 1.000000 1.000000
Kd 0.640000 0.640000 0.640000
Ks 0.500000 0.500000 0.500000
Ke 0.000000 0.000000 0.000000
Ni 1.000000
d 1.000000
illum 2

newmtl Material.001
Ns 96.078431
Ka 1.000000 1.000000 1.000000
Kd 0.640000 0.640000 0.640000
Ks 0.500000 0.500000 0.500000
Ke 0.000000 0.000000 0.000000
Ni 1.000000
d 1.000000
illum 2
map_Kd  icon.png
```

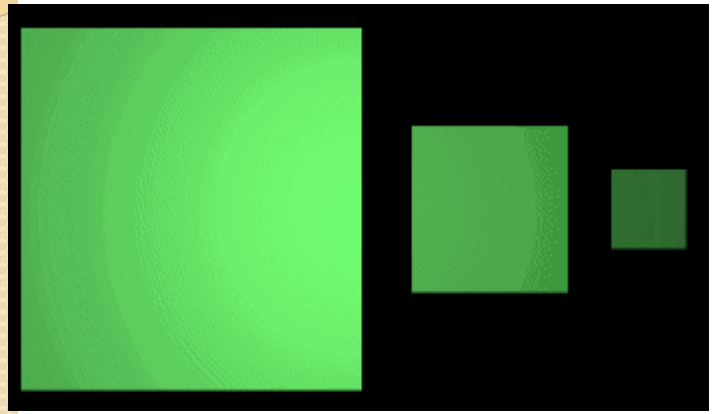
Le "#" spécifie une ligne de commentaires

https://fr.wikipedia.org/wiki/Material_Template_Library

1.4. Milieu traversé : atténuation



Atténuation des sources lumineuses



$$I_{att} = f_{att} I_d$$

$$f_{att} = \frac{1}{D_l^2}$$

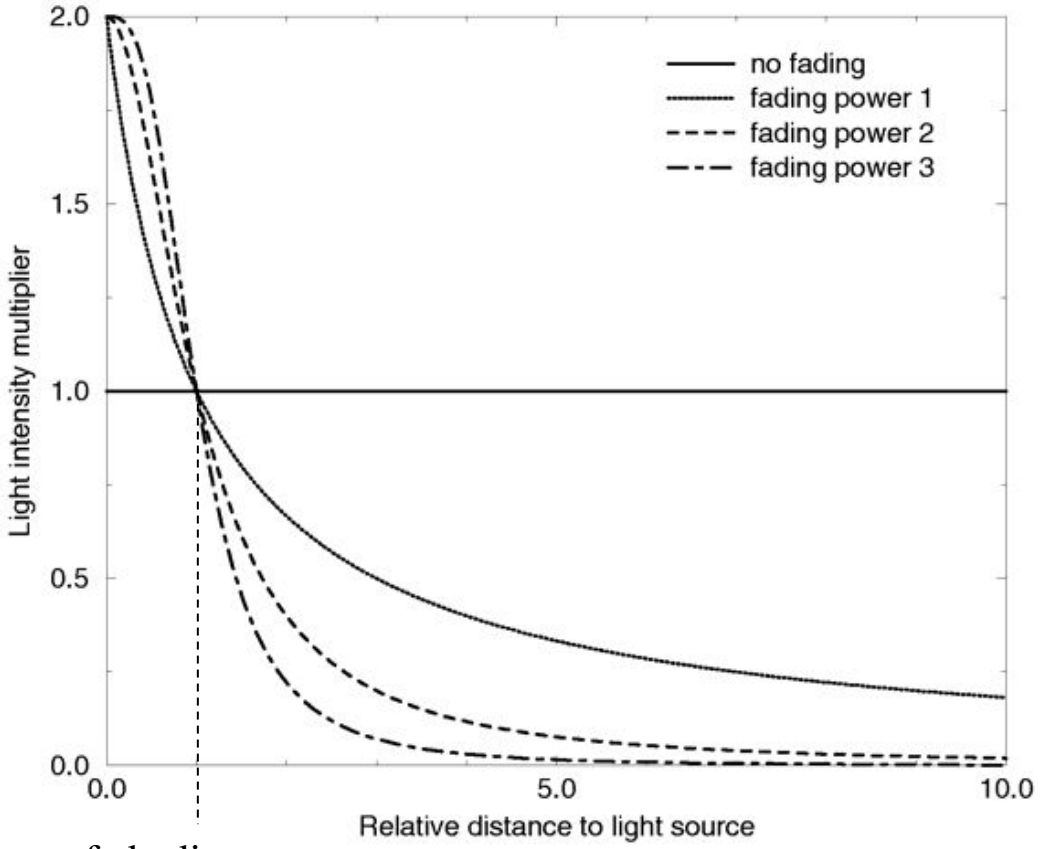
$$f_{att} = \min\left(\frac{1}{c_1 + c_2 D_l + c_3 D_l^2}, 1\right)$$

Effet du milieu traversé

D_l : distance à la source de lumière

Atténuation (light fading)

$$attenuation = \frac{2}{1 + \left(\frac{d}{fadeXdistance}\right)^{fadeXpower}}$$

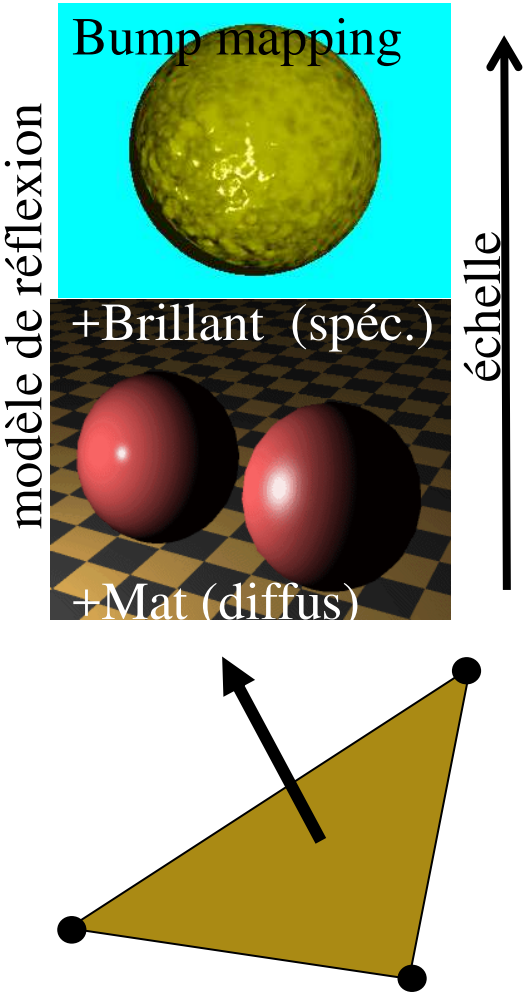


fade distance

2. INSUFFISANCES DU MODÈLE GÉOMÉTRIQUE

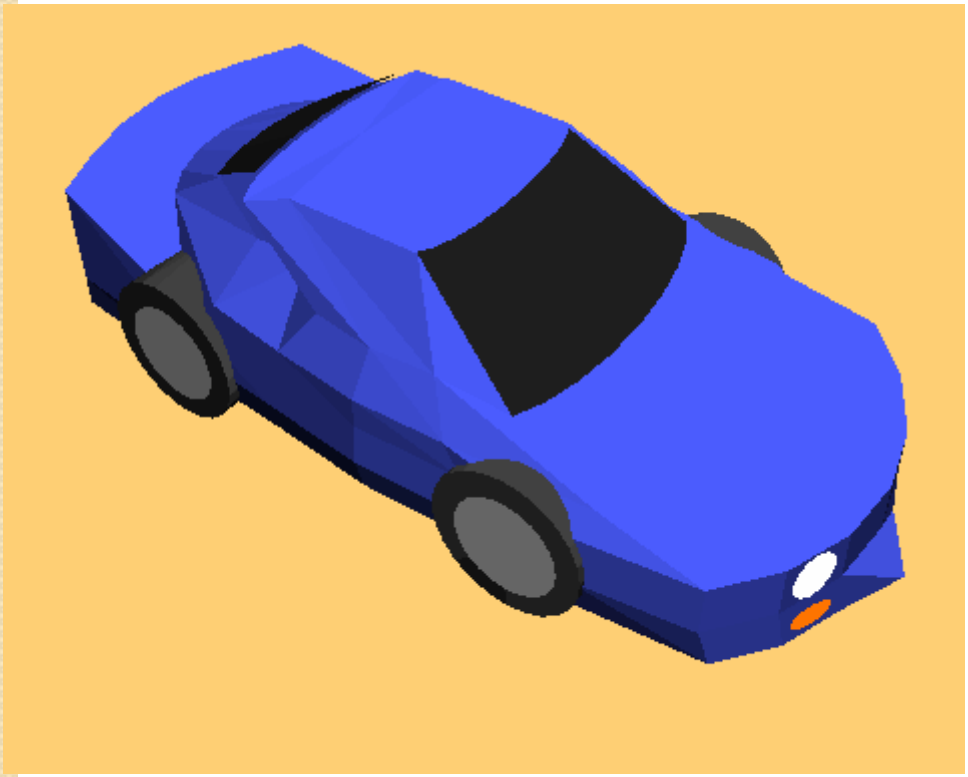
2. Insuffisances du modèle géométrique

- Géométrie de la surface :
  - Le « Bump Mapping »
  - réflexion diffuse/spéculaire
- Géométrie dégradée (Jeux vidéo…)
  - « shading » : flat, Gouraud, Phong

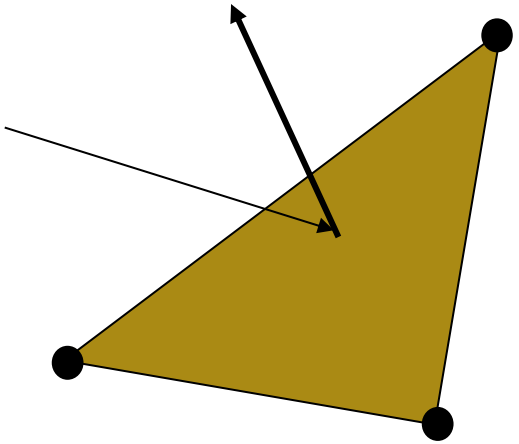


2.1. Affichage : « ombrage » plat (Flat shading)

- Intensité uniforme



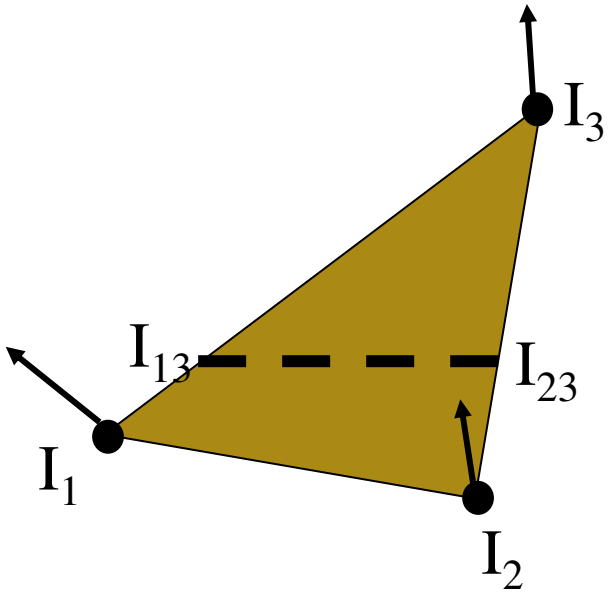
réflexion diffuse

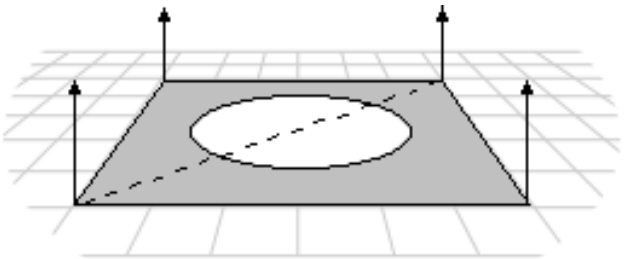


couleur uniforme
sur chaque facette

2.2. Ombrage de Gouraud

- Principe
  - interpolation des intensités
- Algorithme
  - calcul des normales des facettes
  - calcul des normales aux sommets
  - intensité lumineuse des sommets
  - interpolation des intensités
- Inconvénients
  - polygones convexes
  - pas de surfaces brillantes
  - dépend de l'orientation





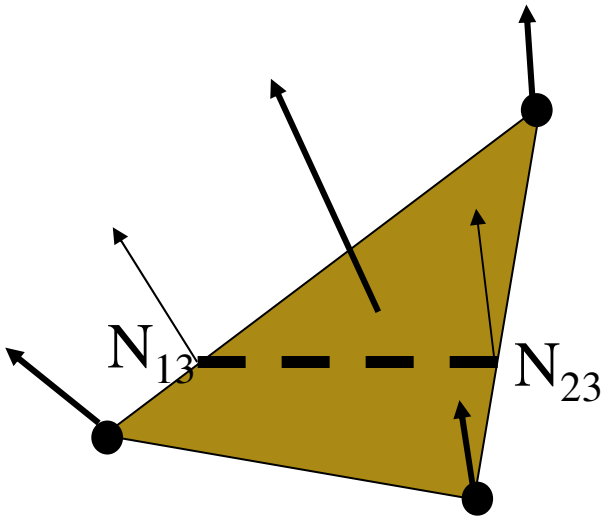
Exemple : Flat vs Gouraud

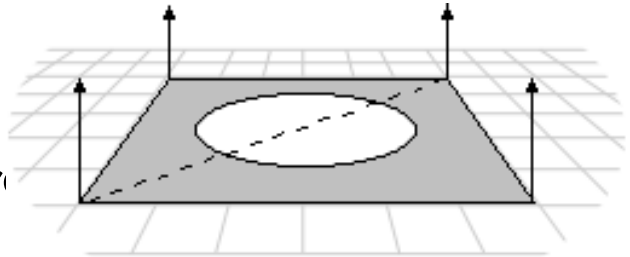
Flat shading

Gouraud (interpolated) shading

2.3. Ombrage de Phong

- Principe
  - Interpolation des normales
- Algorithme
  - calculer les normales aux polygones
  - calculer les *normales* aux sommets
  - calculer les *normales* de tous les points
  - calculer l'intensité lumineuse en tous les points
- Avantage : permet de « capturer » reflets (dans le cas d'un modèle spéculair



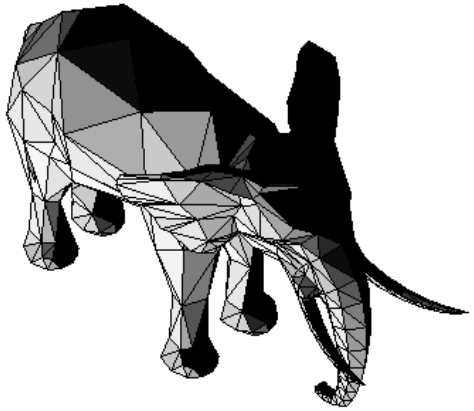


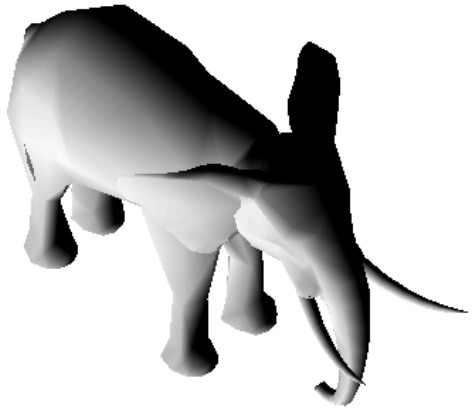
Exemple : Gouraud vs Phong

Gouraud

Phong

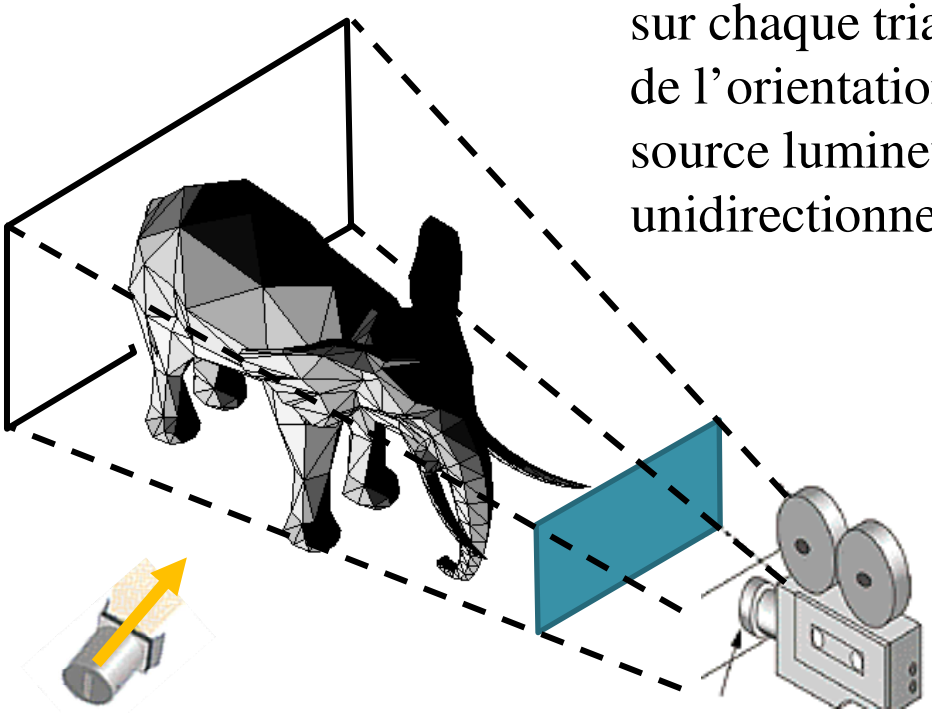
EXEMPLE : Utilisation des normales pour l'éclairage directionnel





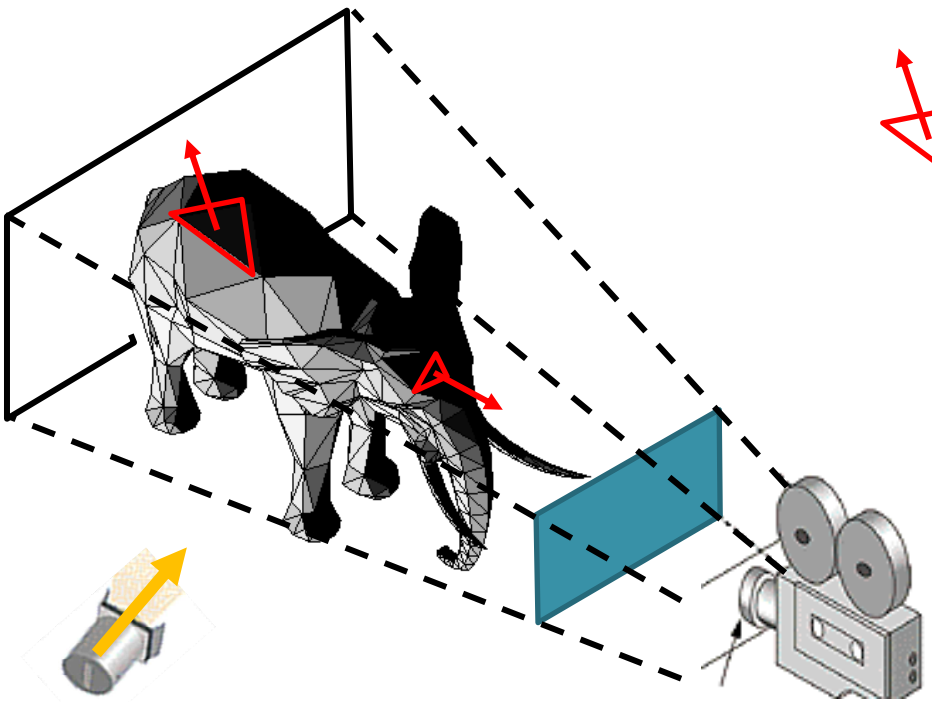
Le « flat shading »

Le flat shading est un modèle d'éclairage d'une surface triangulée qui donne l'intensité sur chaque triangle en fonction de l'orientation relative d'une source lumineuse unidirectionnelle (ex: le soleil).



Le « flat shading »

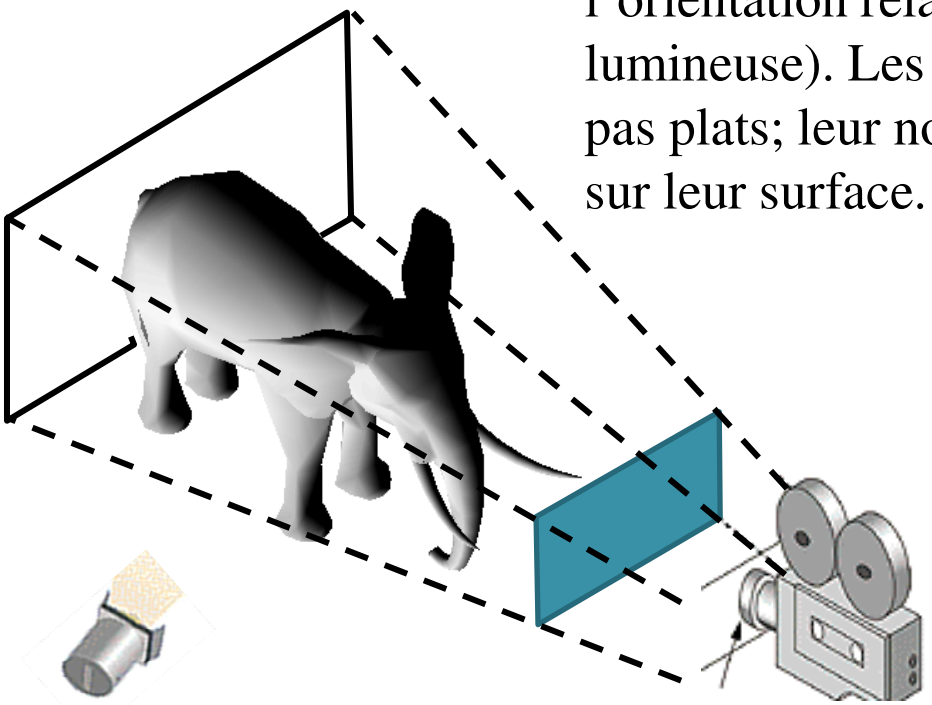
$$I = \min(0, -\vec{V}lum.\vec{N})$$



Ombrage
« flat »

Le « Smooth shading »

Le smooth shading calcule l'intensité en chaque point des triangles (en fonction de l'orientation relative d'une source lumineuse). Les triangles ne sont pas plats; leur normale peut varier sur leur surface.

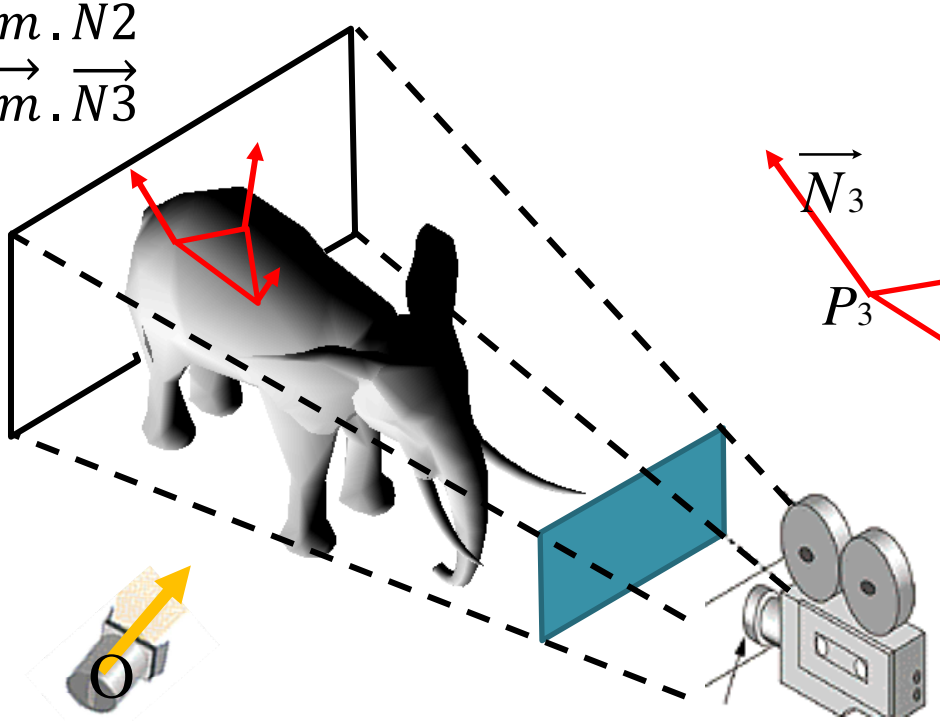


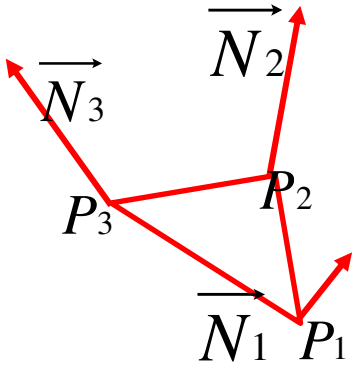
Le « Smooth shading »

Gouraud

$$I1 = -\overrightarrow{Vlum}.\overrightarrow{N1}$$
$$I2 = -\overrightarrow{Vlum}.\overrightarrow{N2}$$
$$I3 = -\overrightarrow{Vlum}.\overrightarrow{N3}$$

$$I(u,v) = (1\text{-}u\text{-}v)*I_1 + u*I_2 + v*I_3$$





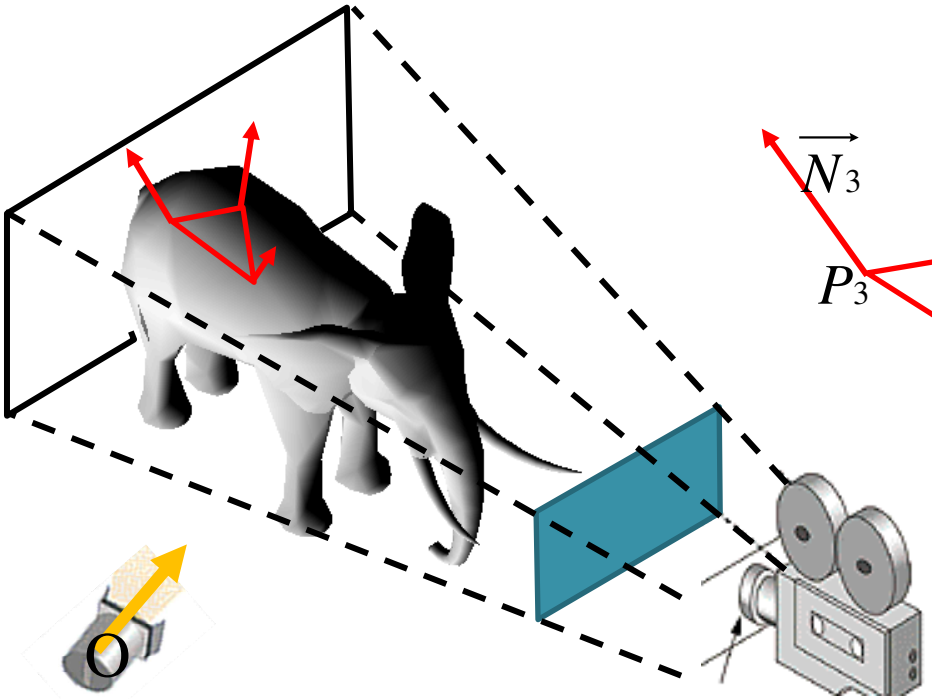
Ombrage « smooth »

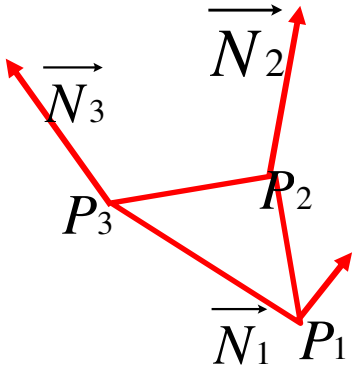
Le « Smooth shading »

Phong

$$\vec{N}(u,v) = (1-u-v) * \vec{N}1 + u * \vec{N}2 + v * \vec{N}3$$

$$I(u,v) = -\vec{V}_{lum}.\vec{N}(u,v)$$

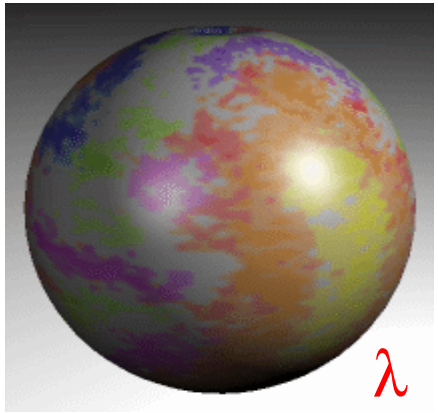




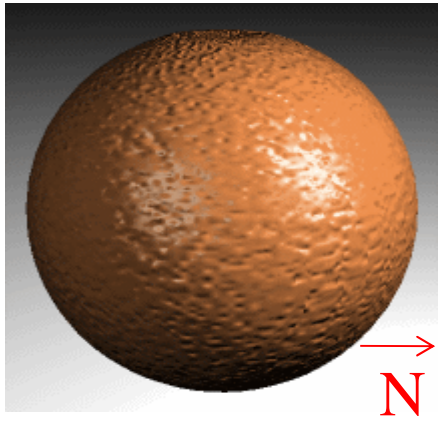
Ombrage « smooth »

3.TEXTURES

3. Les textures

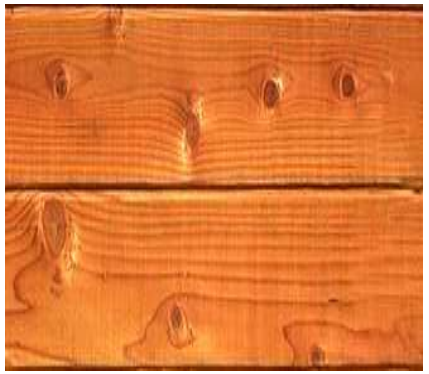


Color mapping



Bump mapping

Motif unique



Motif répété

Perturbateurs : aléas…

SKIP

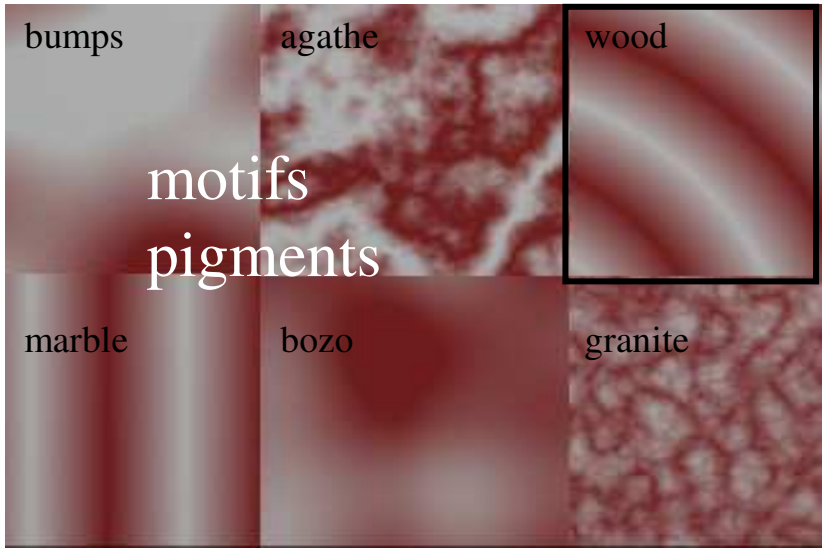
Exemple

Répétition, taille, aléas…

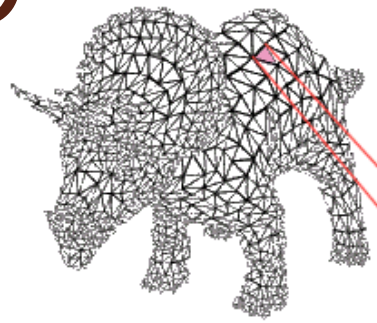
pigments

normales

Surface échelle méso



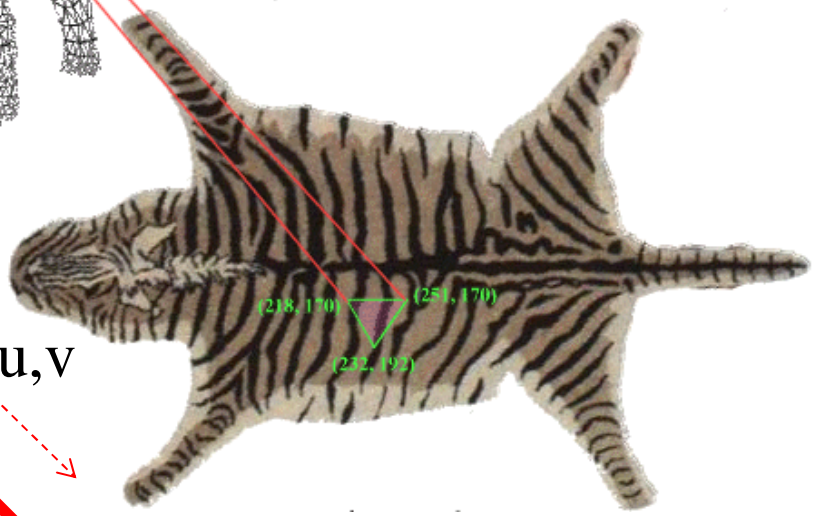
Mapping en 3D



Pour chaque triangle du modèle 3D prendre la texture sur la « photo »

3D

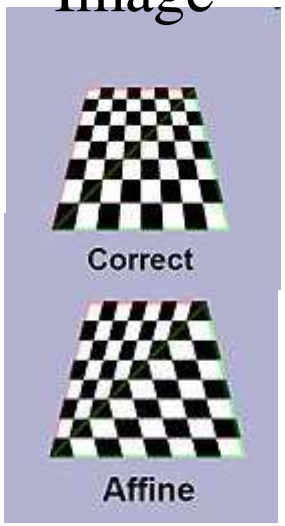
x,y,z



u,v

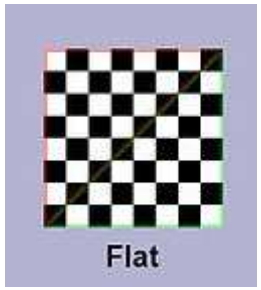
i,j

Image



Projection sur le plan de l'image

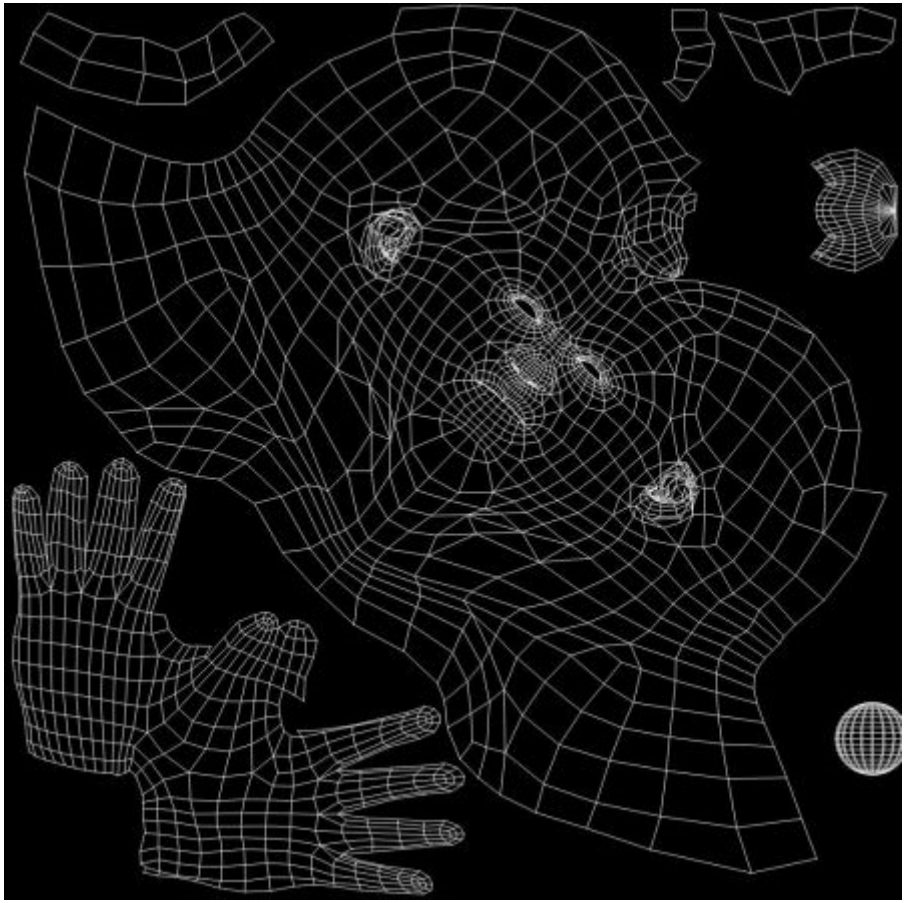
erreur



Texture

Pendant la rasterisation, on interpole les coordonnées sur la carte de texture

Texture : pigment mapping

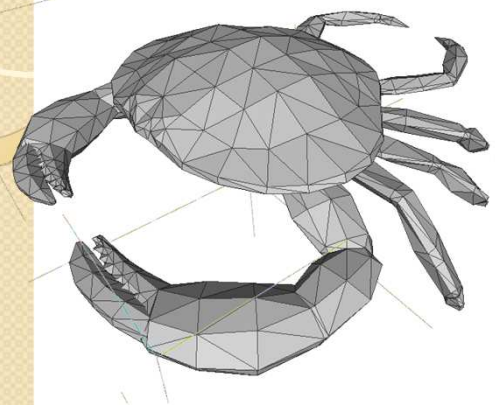


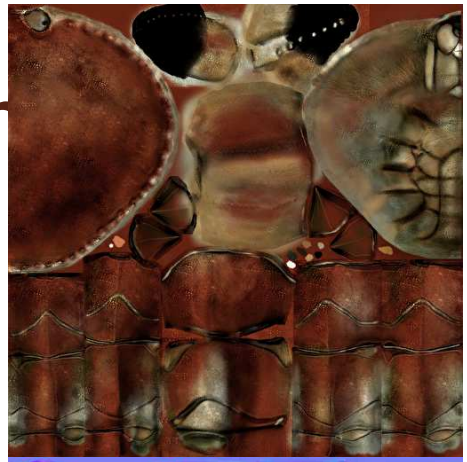
Texture, dans la pratique…

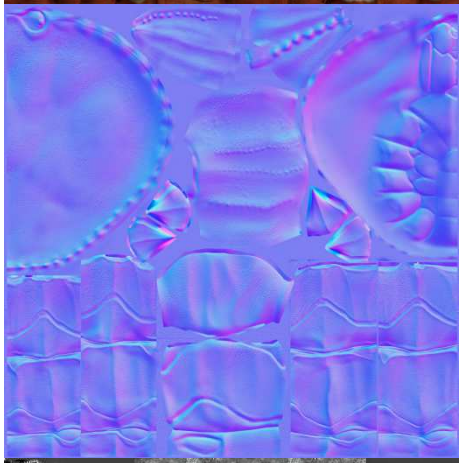
Blender 2.7 Tutorial #13 :
UV Mapping (Unwrapping for Image Textures)

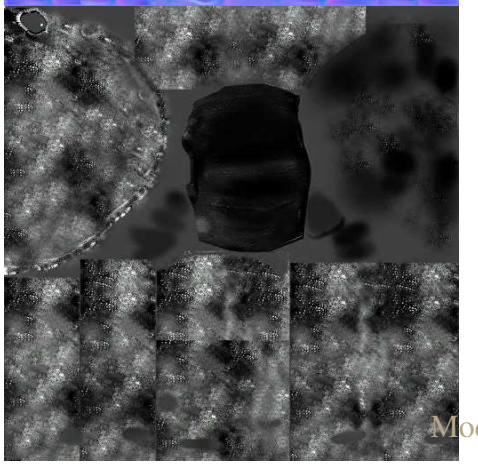
BornCG 2014

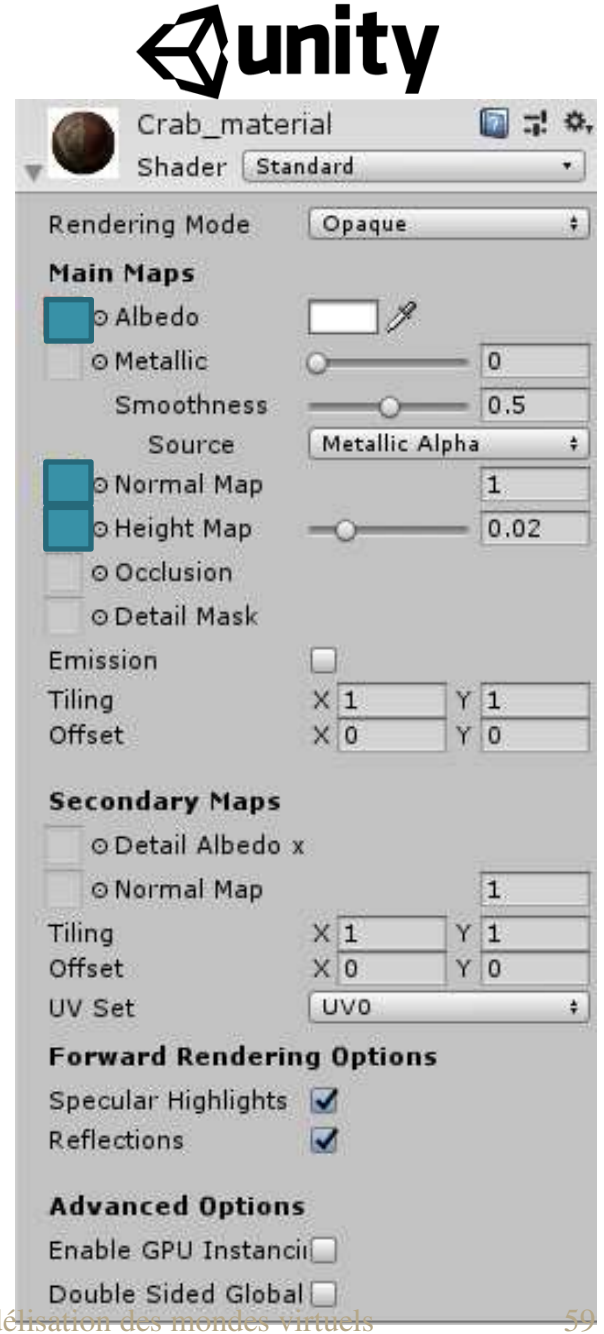
Exemple m










unity
Crab_material
Shader
Standard
Rendering Mode
Opaque
Main Maps
Albedo
Metallic
0
Smoothness
0.5
Source
Metallic Alpha
Normal Map
1
Height Map
0.02
Occlusion
Detail Mask
Emission
Tiling
X 1
Y 1
Offset
X 0
Y 0
Secondary Maps
Detail Albedo x
Normal Map
1
Tiling
X 1
Y 1
Offset
X 0
Y 0
UV Set
UV0
Forward Rendering Options
Specular Highlights
Reflections
Advanced Options
Enable GPU Instanci
Double Sided Global

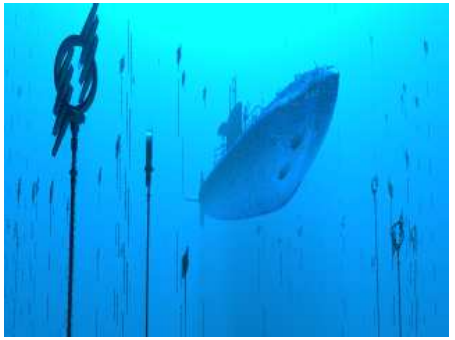
Pour aller plus loin…

- Chap « Illumination and Shading »
  - Computer Graphics – Foley, Van Dam…Addison Wesley
- Édition de texture

  Blender 2.7 Tutorial #13 :
  UV Mapping (Unwrapping for Image Textures)
- PovRay

  https://people.minesparis.psl.eu/olivier.stab/SIRV/

- Unity, (Brackeys) :
  - Lighting in Unity,
  - RealTime Lighting in Unity



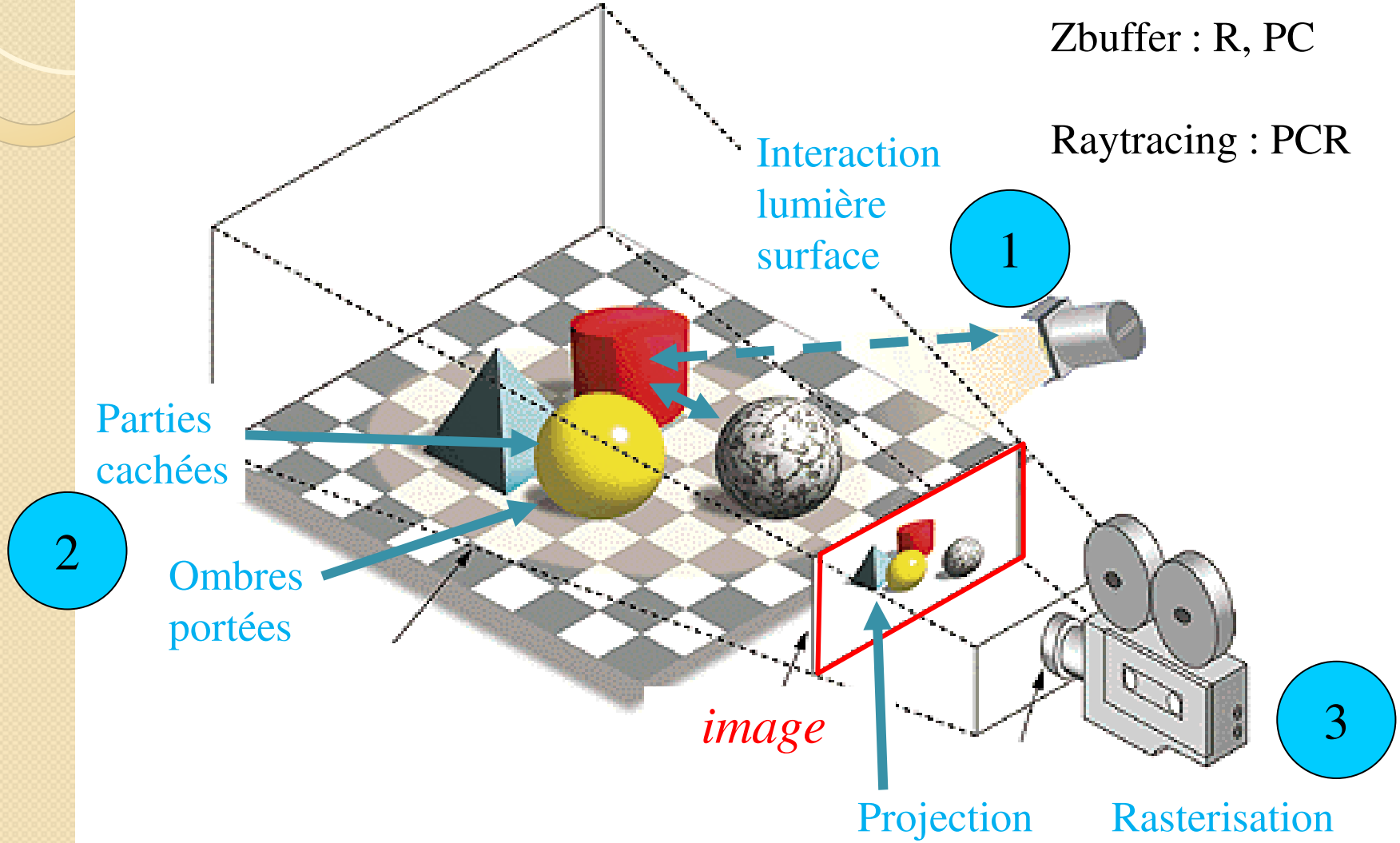
4. RAY TRACING & PC

L'image

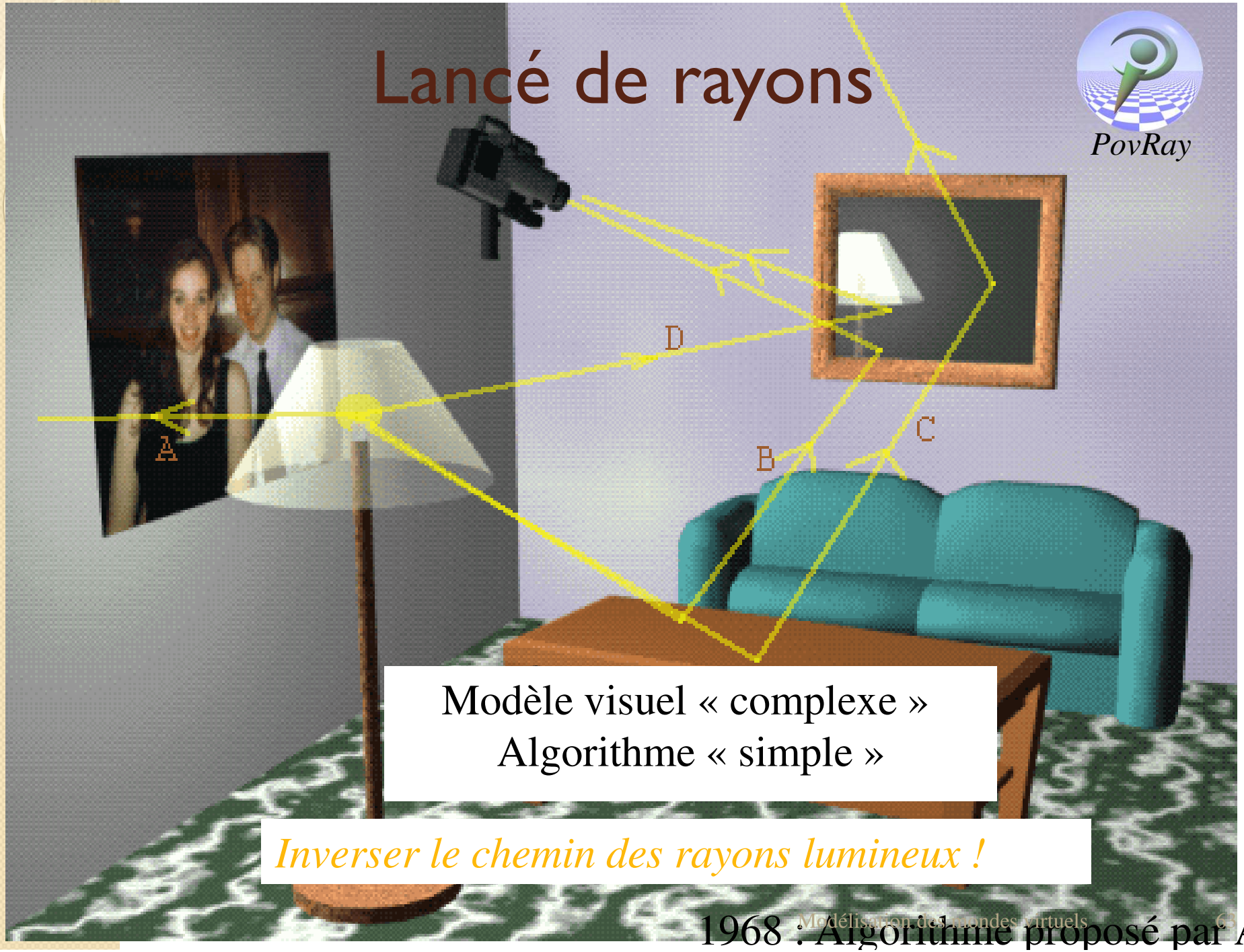
A.Peintre : PC, R

Zbuffer : R, PC

Raytracing : PCR



Lancé de rayons



Modèle visuel « complexe »
Algorithme « simple »

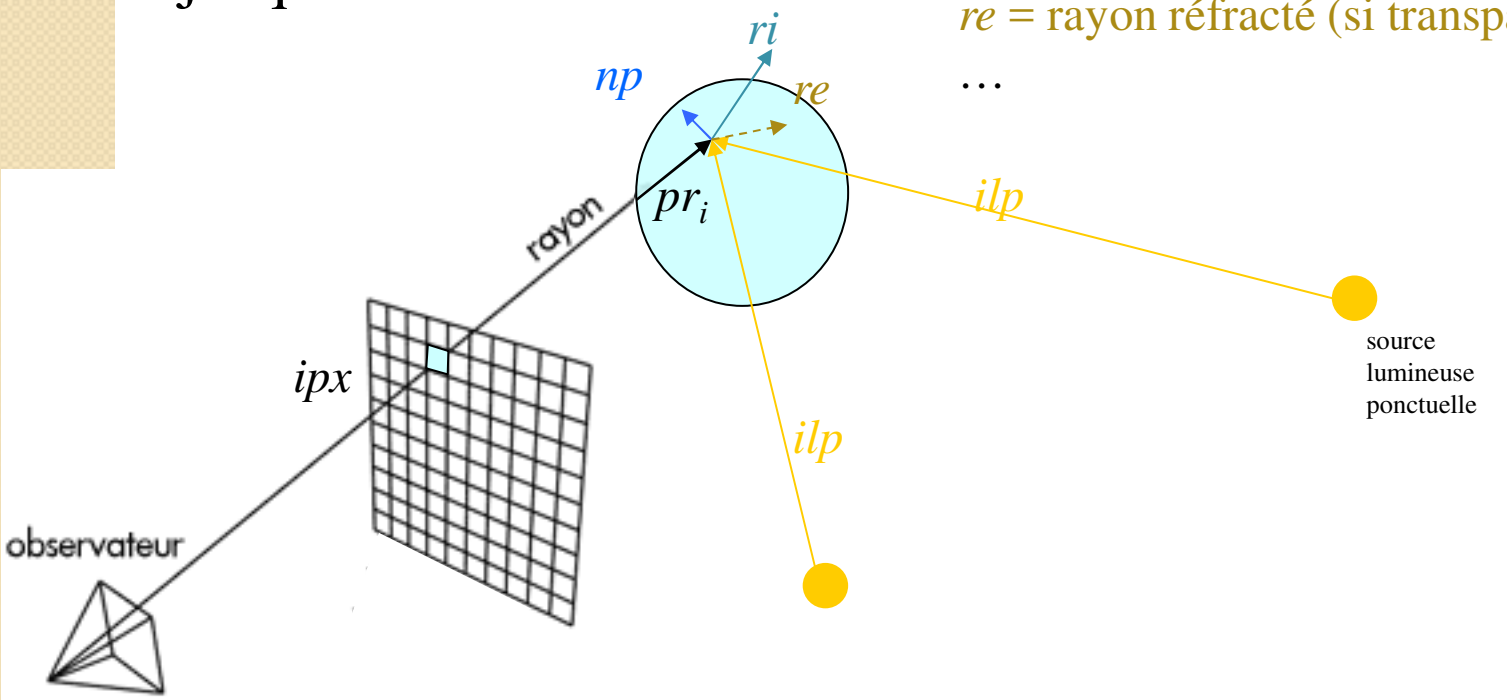
Inverser le chemin des rayons lumineux !

1968 : Algorithme proposé par A

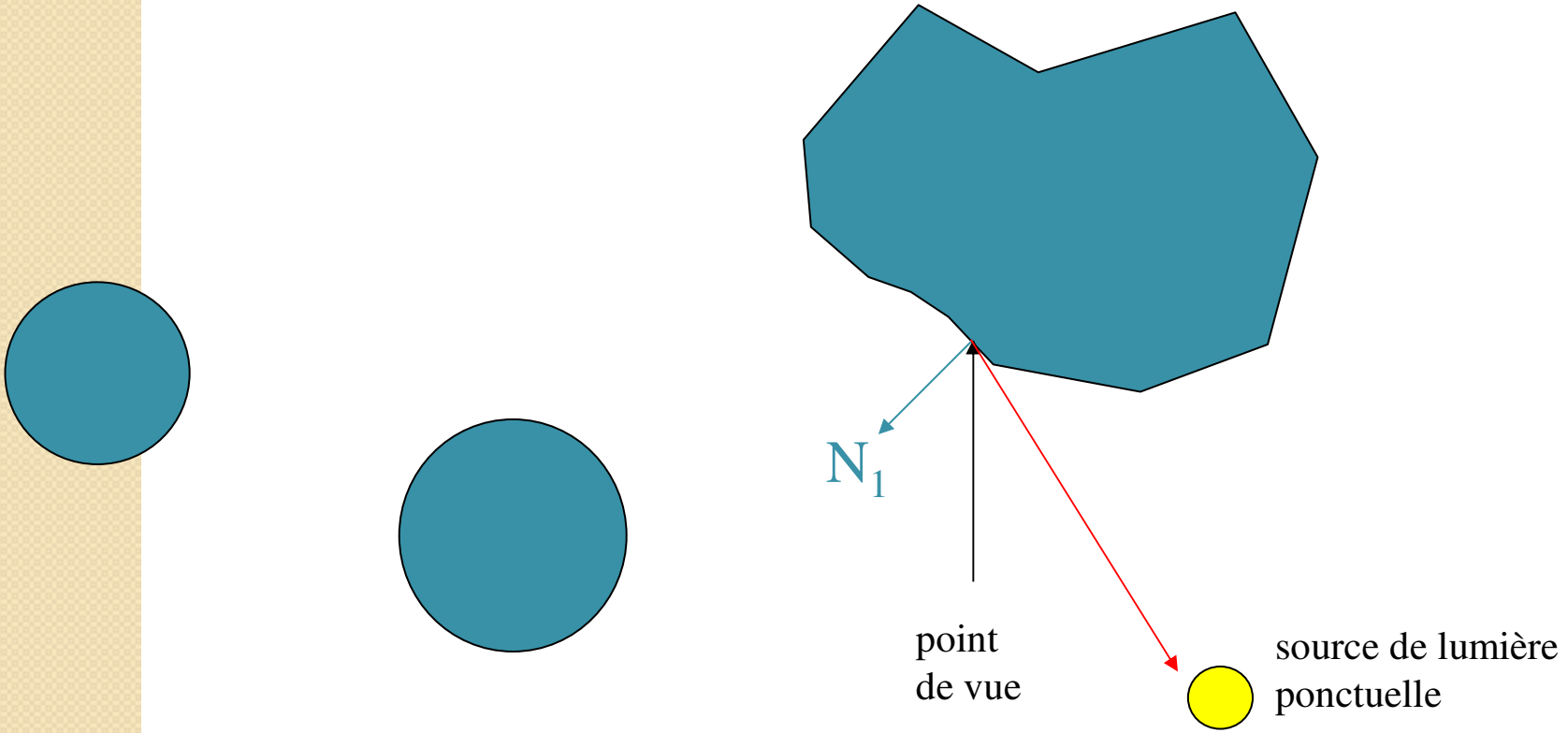
Lancé de rayon
(Ray tracing)

Principe : modéliser le trajet des rayons de la lumière jusqu'à l'oeil

ipx = intensité lumineuse du pixel
r = lancé d'un rayon partant de *ipx*
pr_i = intersection de r avec la géométrie
np = normale en pr_i
ilp=éclairage de la surface en pr_i
ri = rayon réfléchi en pr_i (si miroir)
re = rayon réfracté (si transparence)
…

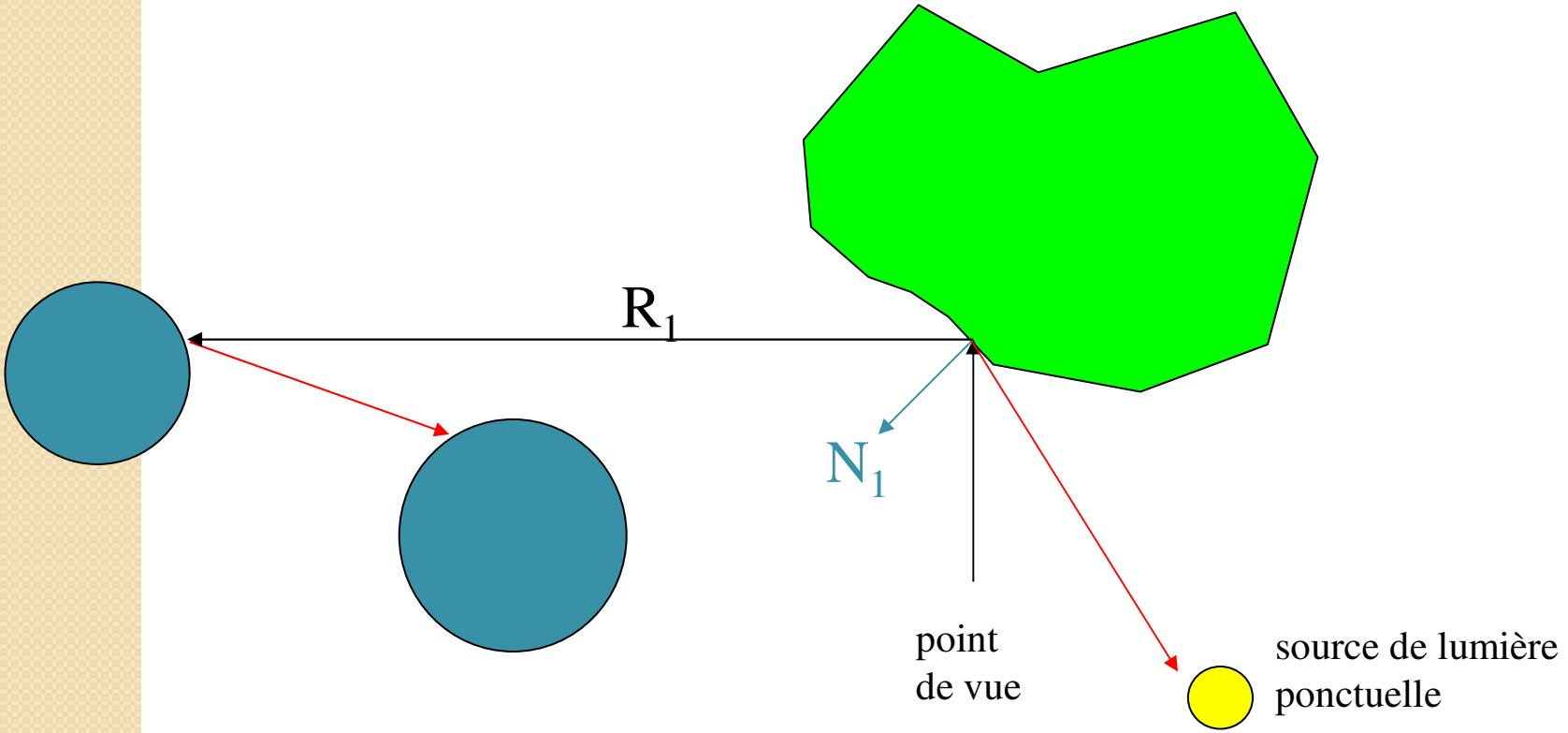


Principe : lancé de rayon



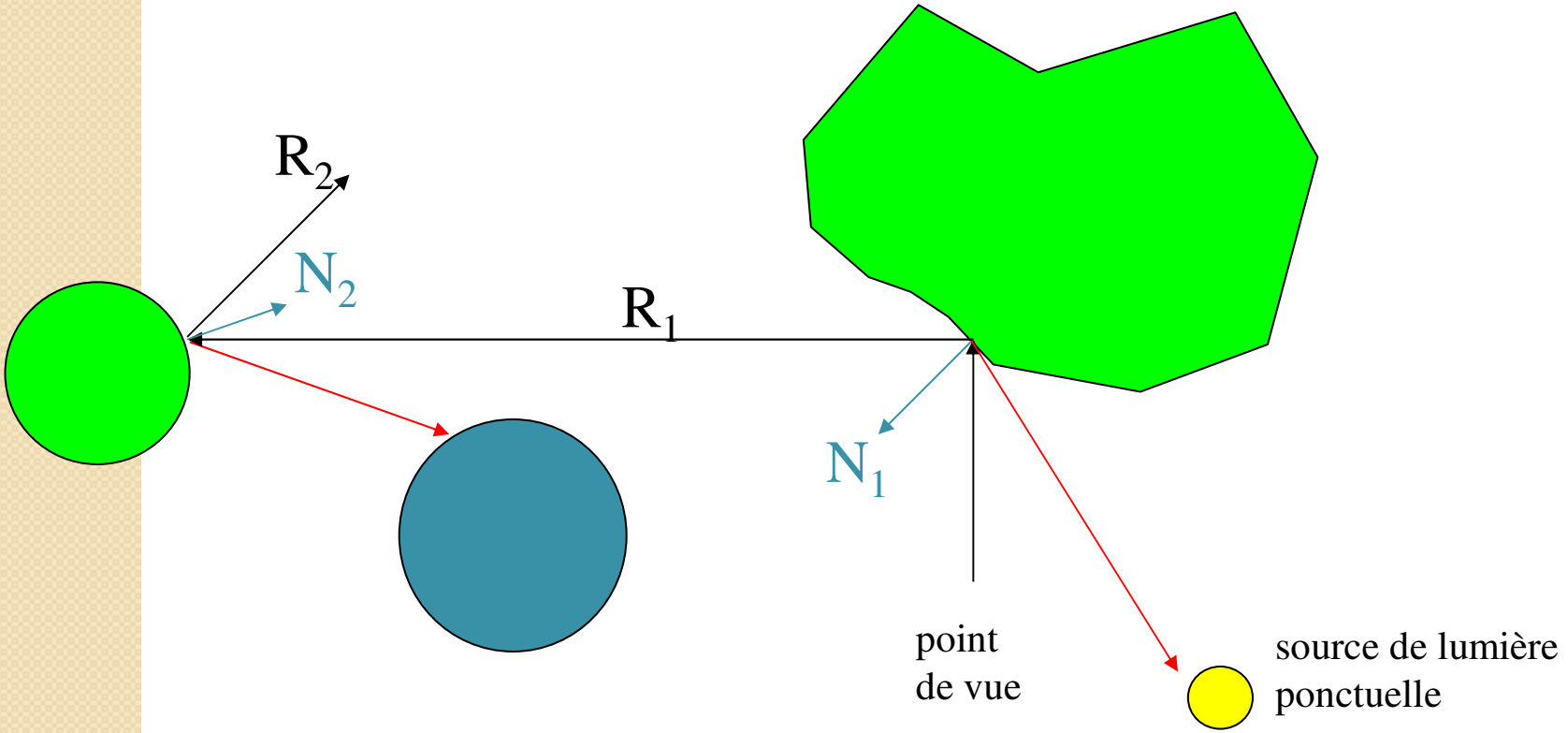
from CG principles & practice Foley, Van Dam …

Principe : lancé de rayon



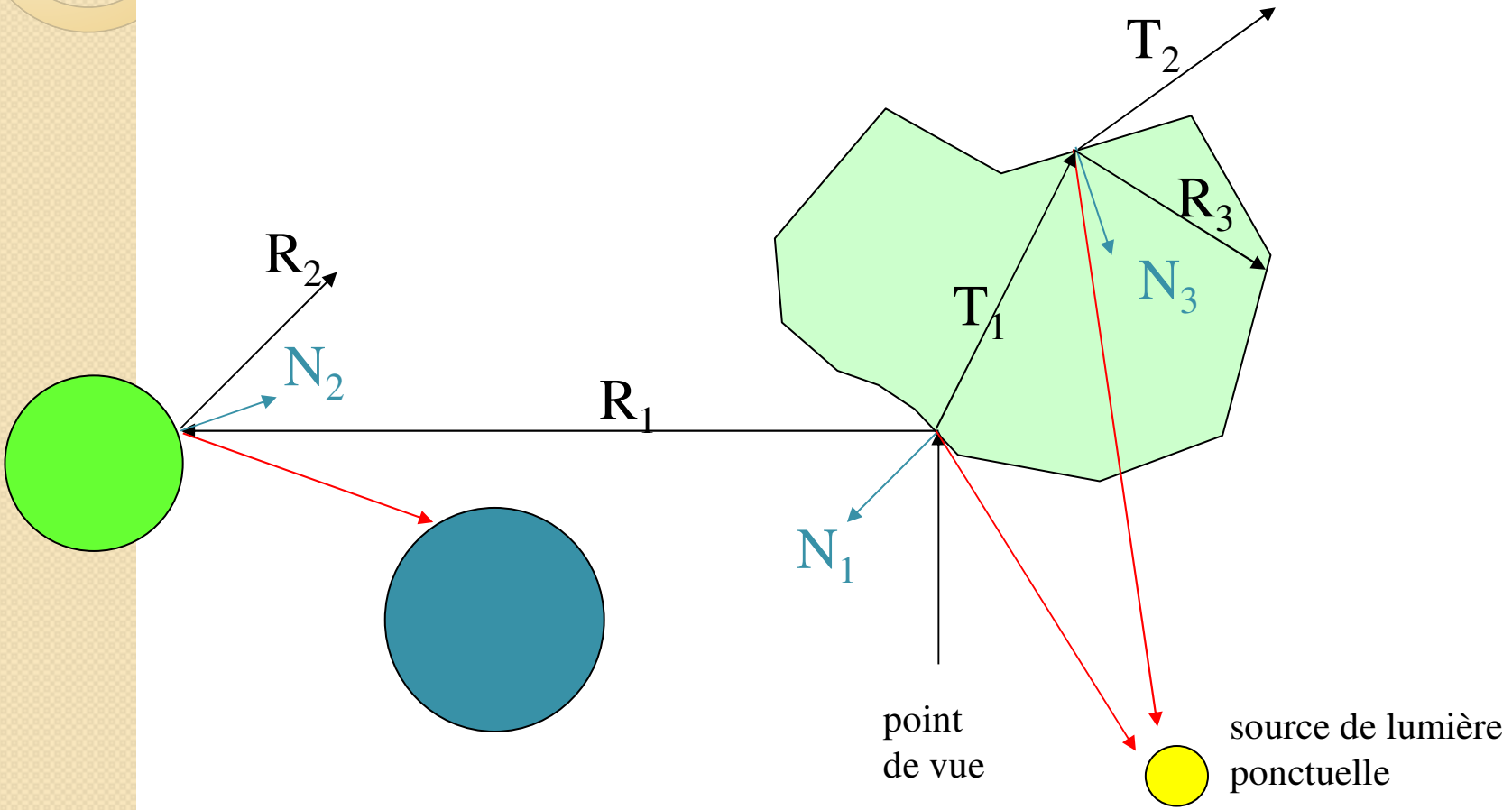
from CG principles & practice Foley, Van Dam …

Principe : lancé de rayon



from CG principles & practice Foley, Van Dam …

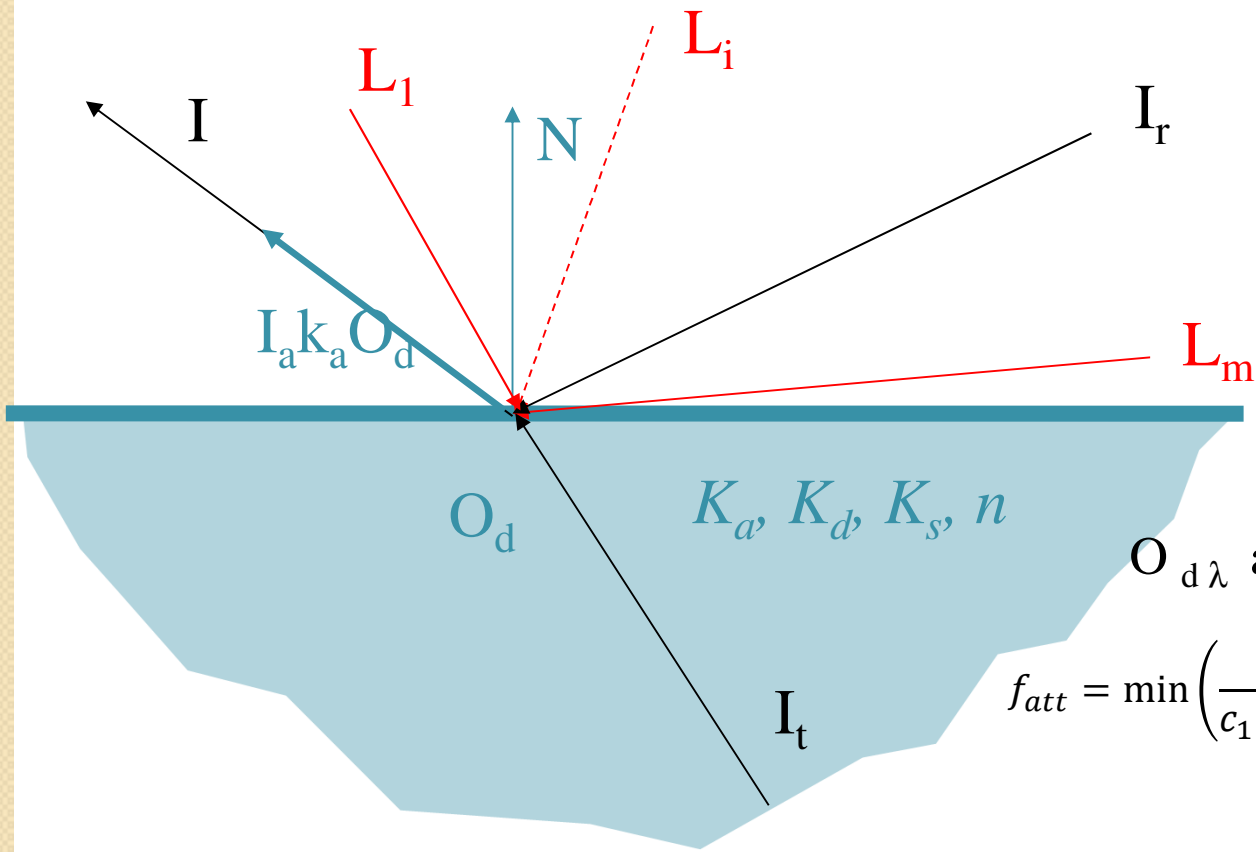
Principe : lancé de rayon



from CG principles & practice Foley, Van Dam …

Définition récursive ➔ lancé de rayons

$$I_\lambda = \underbrace{I_{a\lambda} K_a O_{d\lambda}}_{\text{ambiante}} + \underbrace{\sum_{i=1}^{m} S_i f_{att\_i} I_{p\lambda i} \left[\underbrace{K_{di} O_{d\lambda} \left(\vec{N}.\vec{L}_i\right)}_{\text{diffuse}} + \underbrace{K_s \left(\vec{R}_i.\vec{V}\right)^n}_{\text{spéculaire}} \right]}_{\text{sources}} + \underbrace{K_r I_{r\lambda}}_{\text{refléchie}} + \underbrace{K_t I_{t\lambda}}_{\text{réfractée}}$$

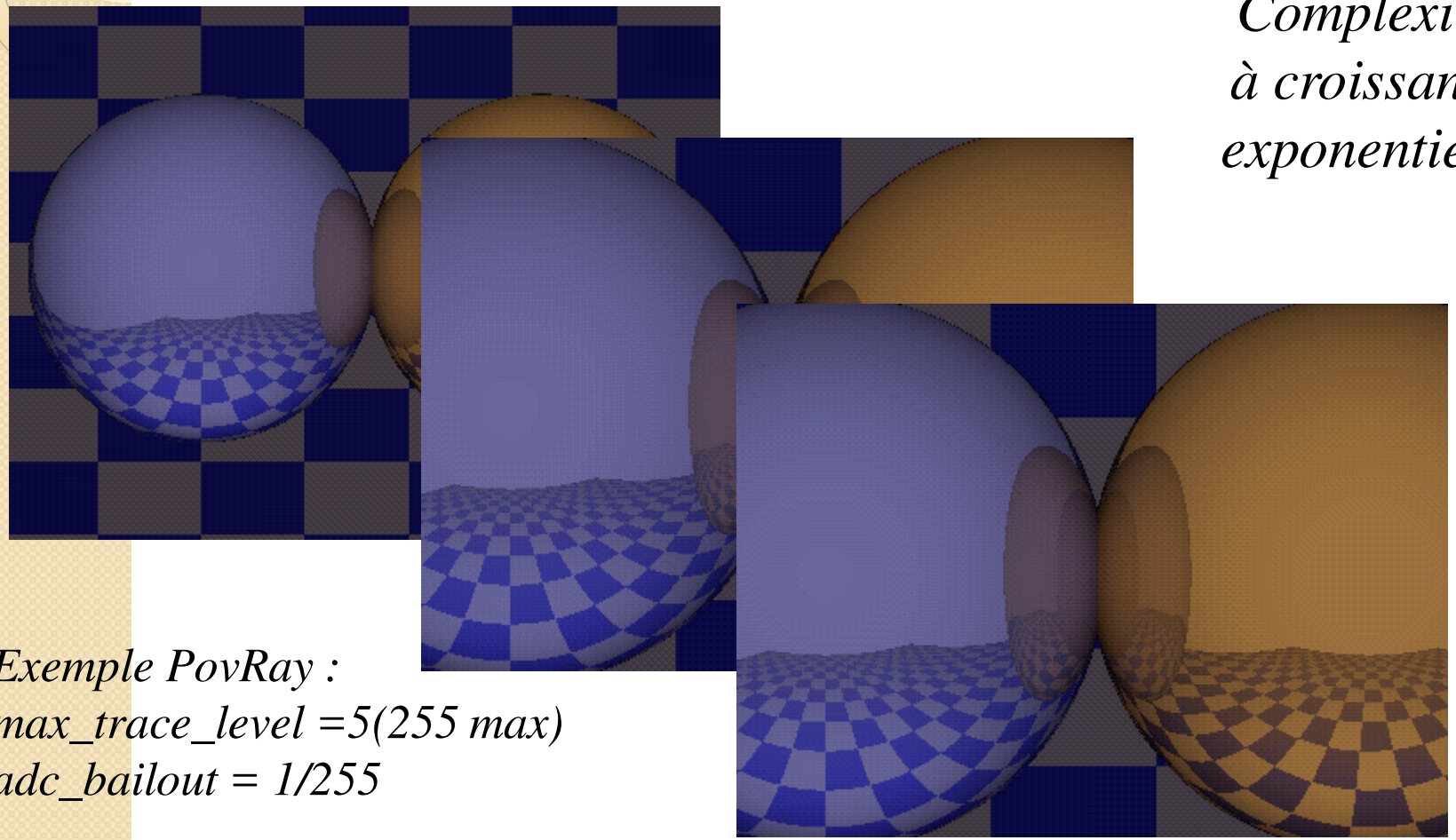


$S_i = 0$ si obstacle
$S_i = 1$ sinon

$O_{d\lambda}$ avec λ = R, V ou B

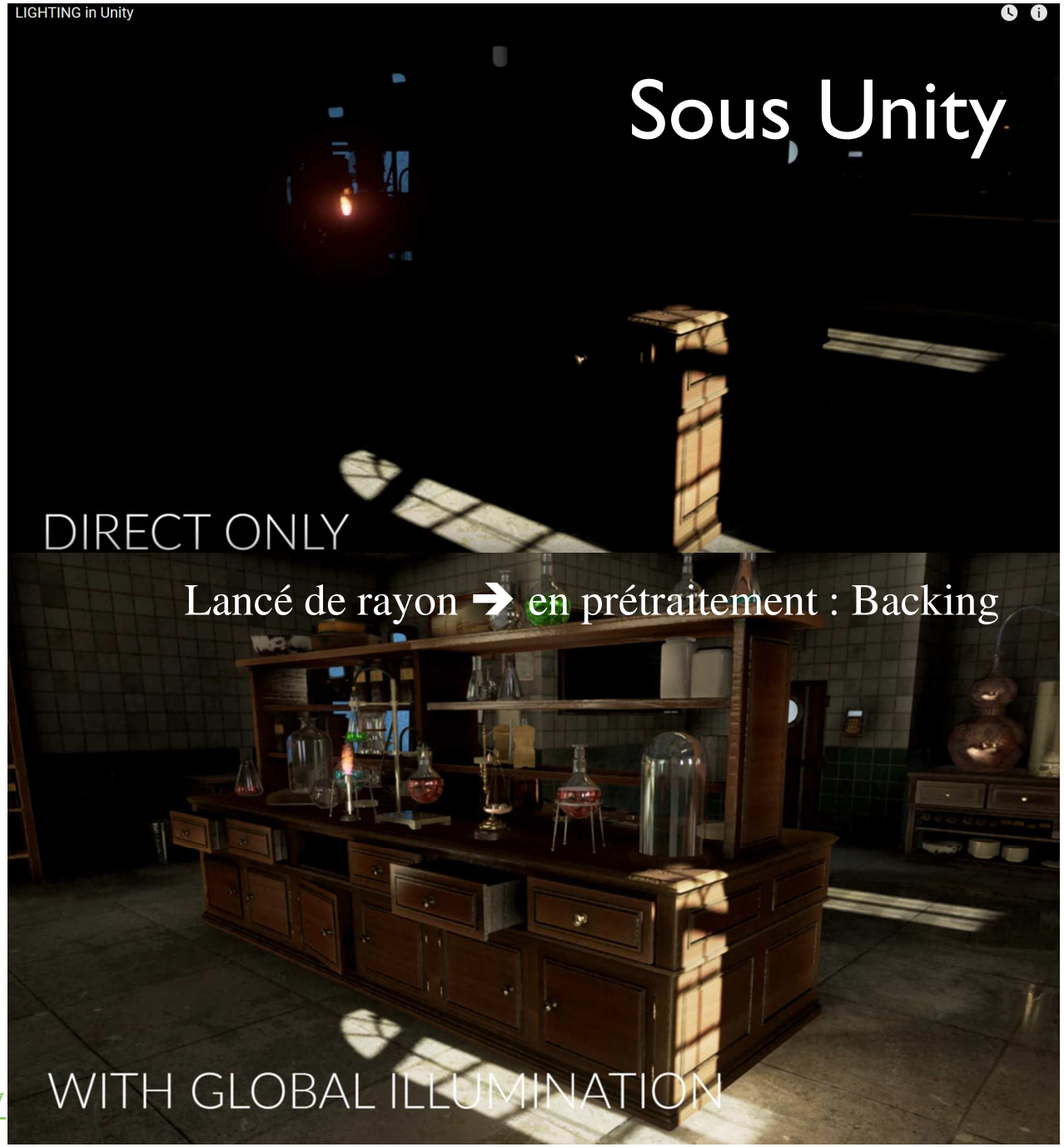
$$f_{att} = \min\left(\frac{1}{c_1 + c_2 D_l + c_3 D_l^2}, 1\right)$$

Jusqu'où ?



Complexité à croissance exponentielle

Exemple PovRay :
max_trace_level =5(255 max)
adc_bailout = 1/255



Brackeys :

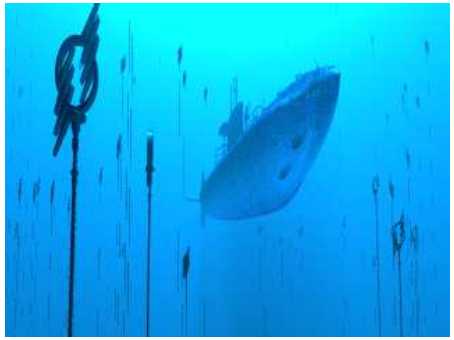
- Lighting in Unity,
- RealTime Lighting in Unity
- …

Pour aller plus loin…

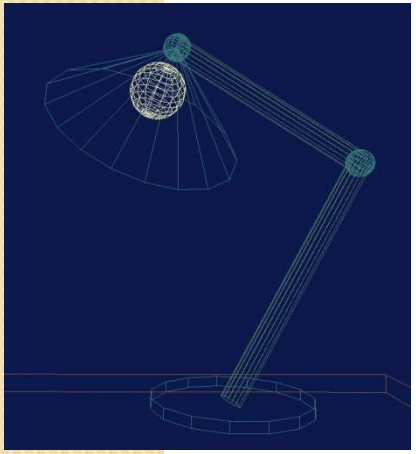
- Chap « Illumination and Shading »
  - Computer Graphics – Foley, Van Dam…Addison Wesley
- Édition de texture

  Blender 2.7 Tutorial #13 :
  UV Mapping (Unwrapping for Image Textures)
- PovRay

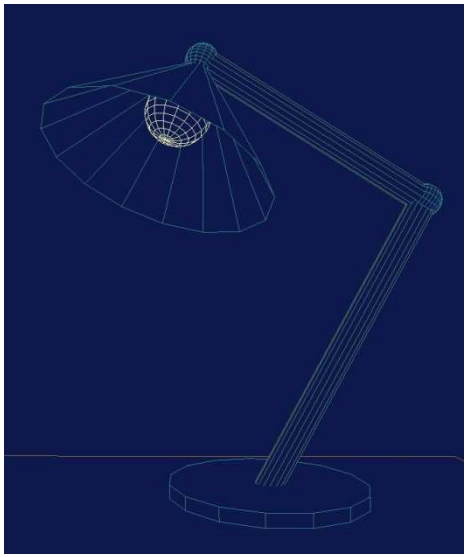
  https://people.minesparis.psl.eu/olivier.stab/SIRV/
- Unity, (Brackeys) :
  - Lighting in Unity,
  - RealTime Lighting in Unity



5. Parties cachées & rendu (Brep)



Wireframe



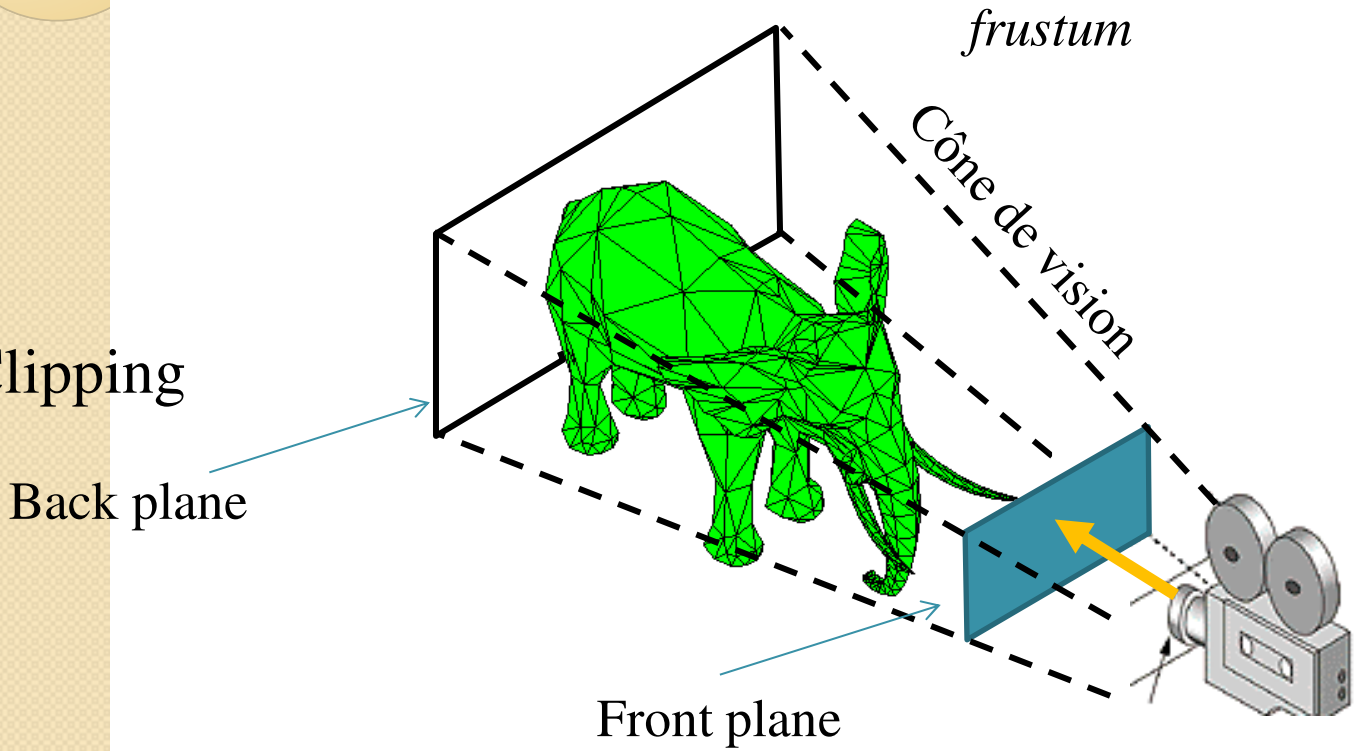
Hidden Lines

Flat Shading

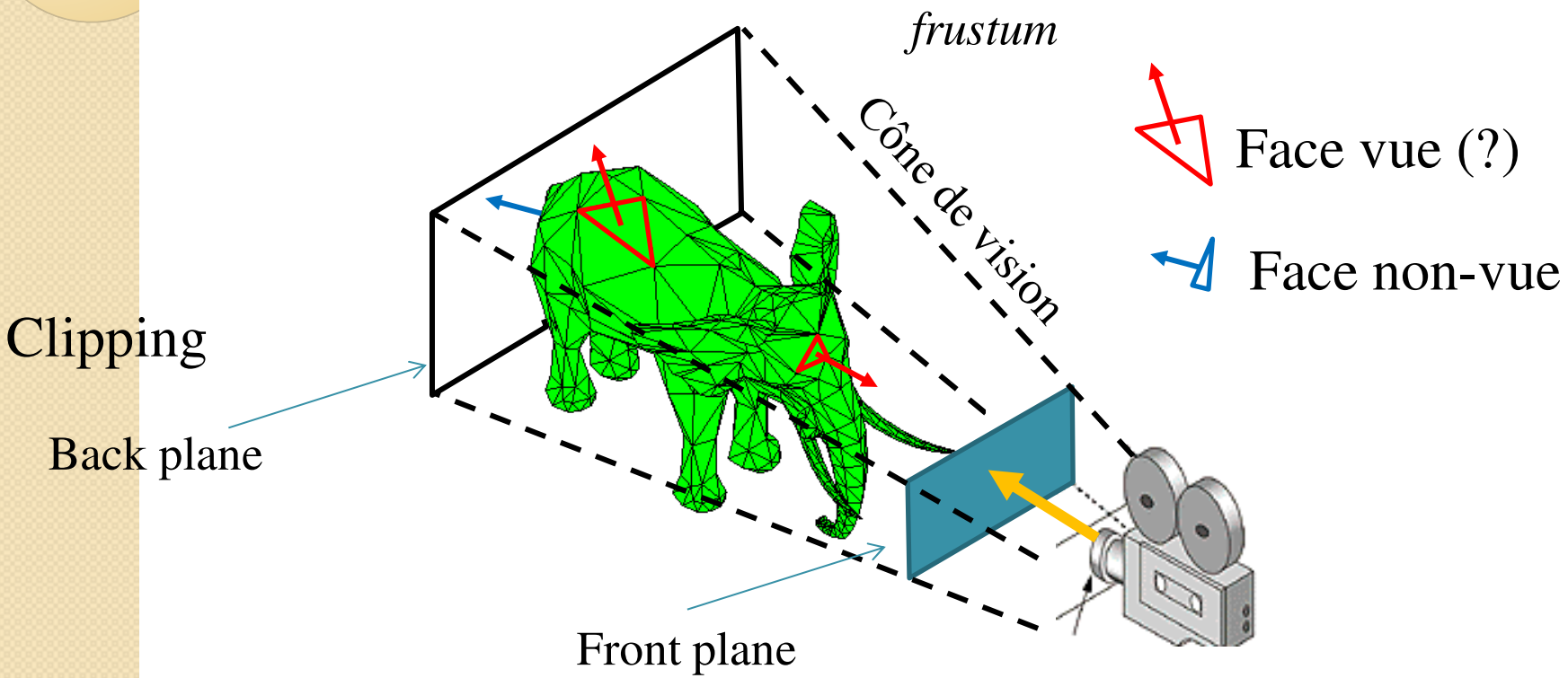
Antialiasing

Gouraud Shading

Prétraitement : le Clipping

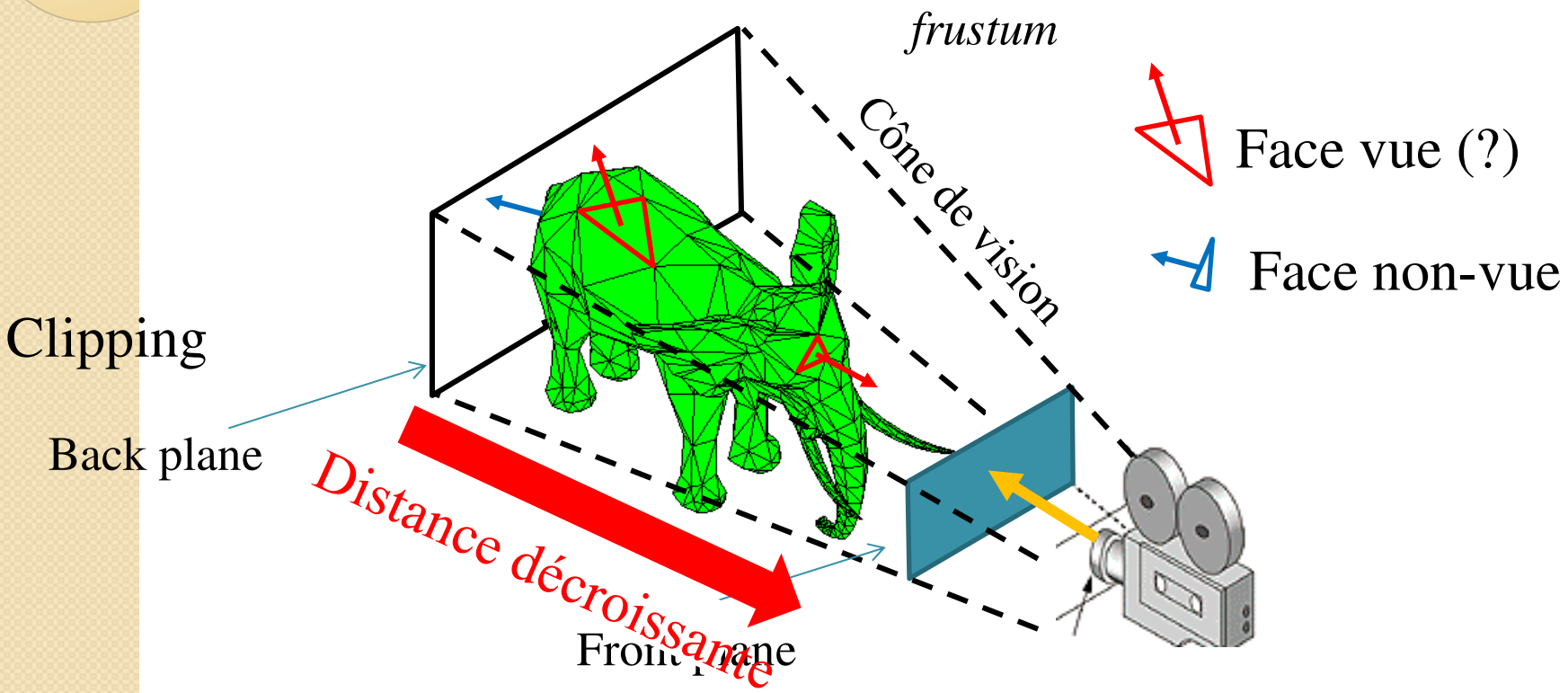


Prétraitement : Le Back face culling



Dans quel cas le « back-face culling » résout-il le problème des parties cachées ?

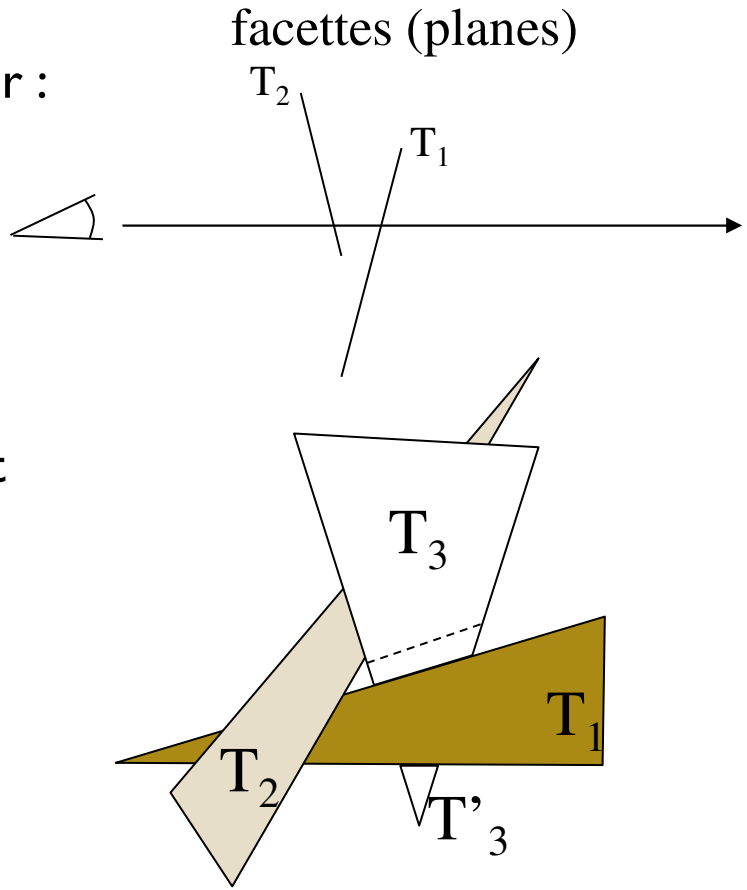
L'algorithme du peintre



Algorithme du peintre

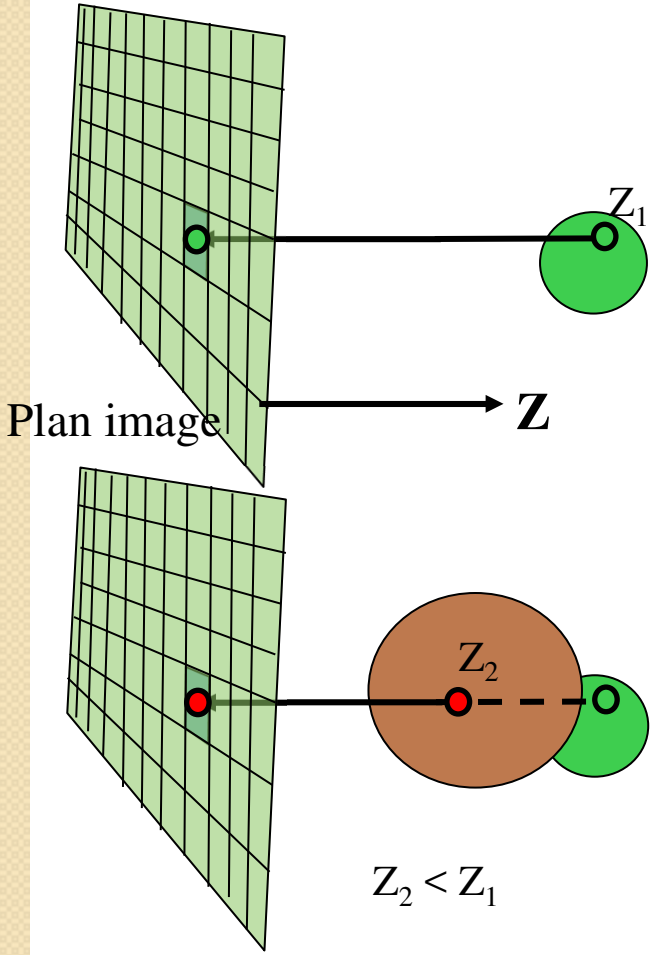
- Liste ordonnée de facettes
  - trier les facettes suivant leur profondeur : Z-max
  - afficher en Z-max décroissant
  - ambiguïtés : problème du recouvrement cyclique…

$T_2 < T_1$

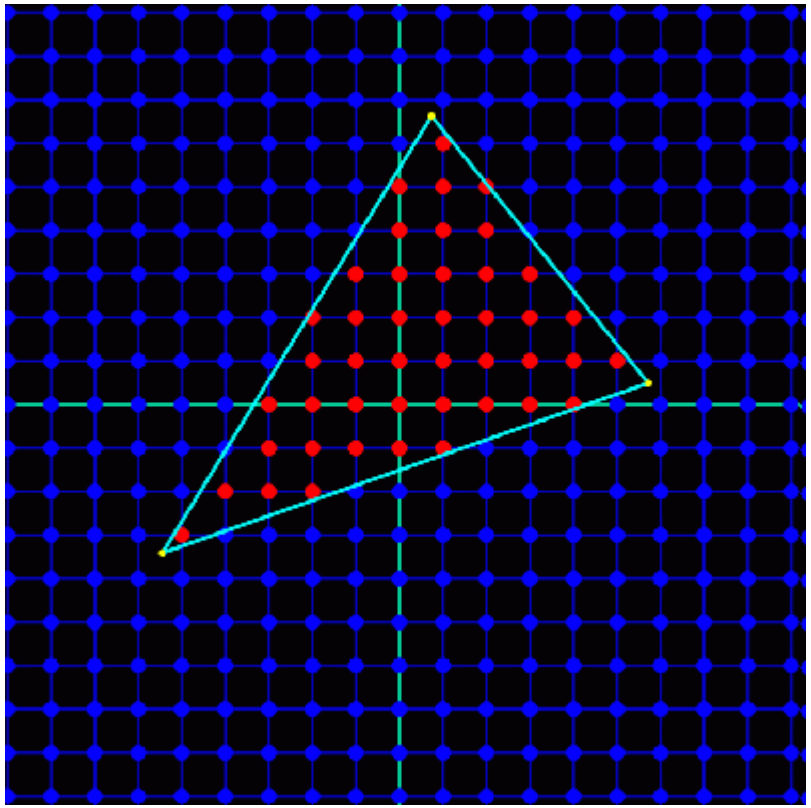


Z-buffer* : ordre de profondeur

Distance min(Z_i) par pixel?



Pendant la « pixelisation »



A la résolution écran

**Edwin Catmull 1974*

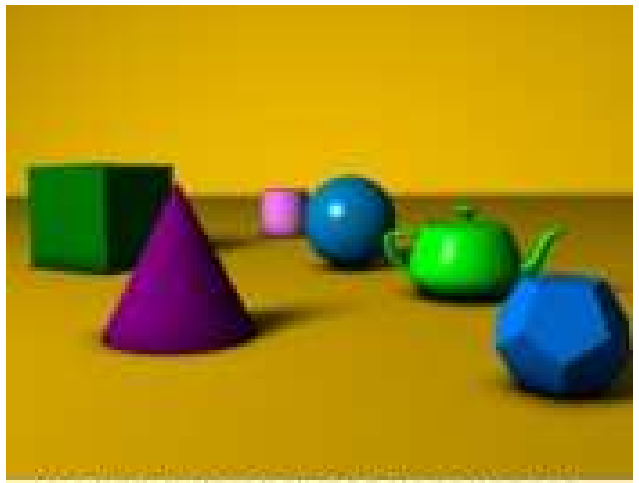
Z-buffer (exemple)

5	10	10	10	10	10	10
4	10	10	10	10	10	10
3	10	4	4	4	10	10
2	10	4	4	4	10	10
1	10	4	4	4	10	10
0	10	10	10	10	10	10
	0	1	2	3	4	5

y, x

5	10	10	10	10	10	10
4	10	10	10	10	10	10
3	10	4	4	4	10	10
2	10	4	4	2	2	2
1	10	4	4	2	2	2
0	10	10	10	2	2	2
	0	1	2	3	4	5

y, x



A simple three dimensional scene

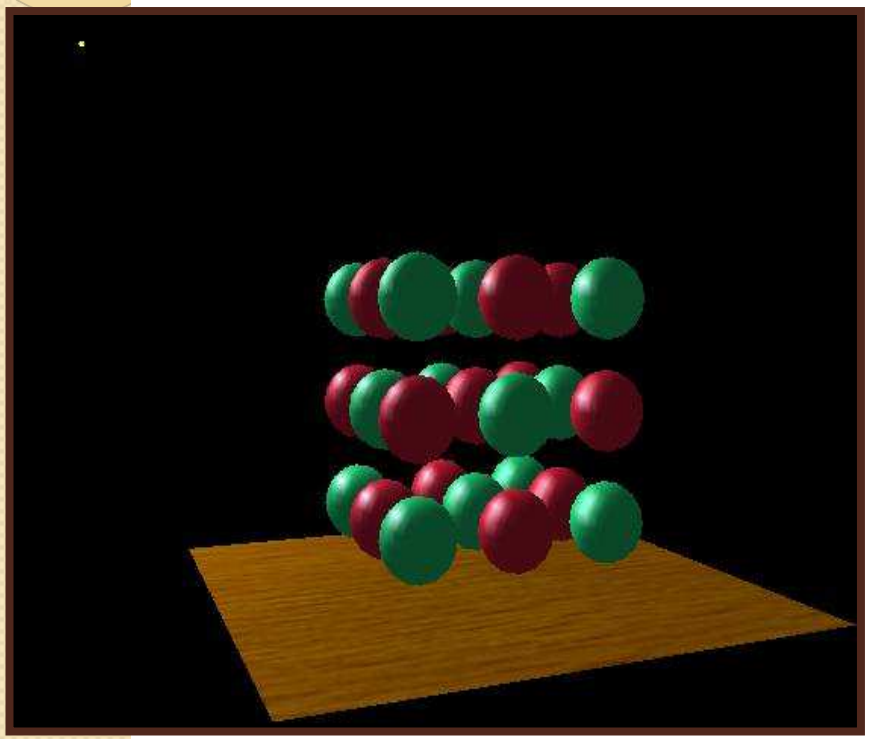
5	10	10	8	8	8	10
4	10	10	8	8	8	10
3	10	4	4	4	8	10
2	10	4	4	2	2	2
1	10	4	4	2	2	2
0	10	10	10	2	2	2
	0	1	2	3	4	5

y, x

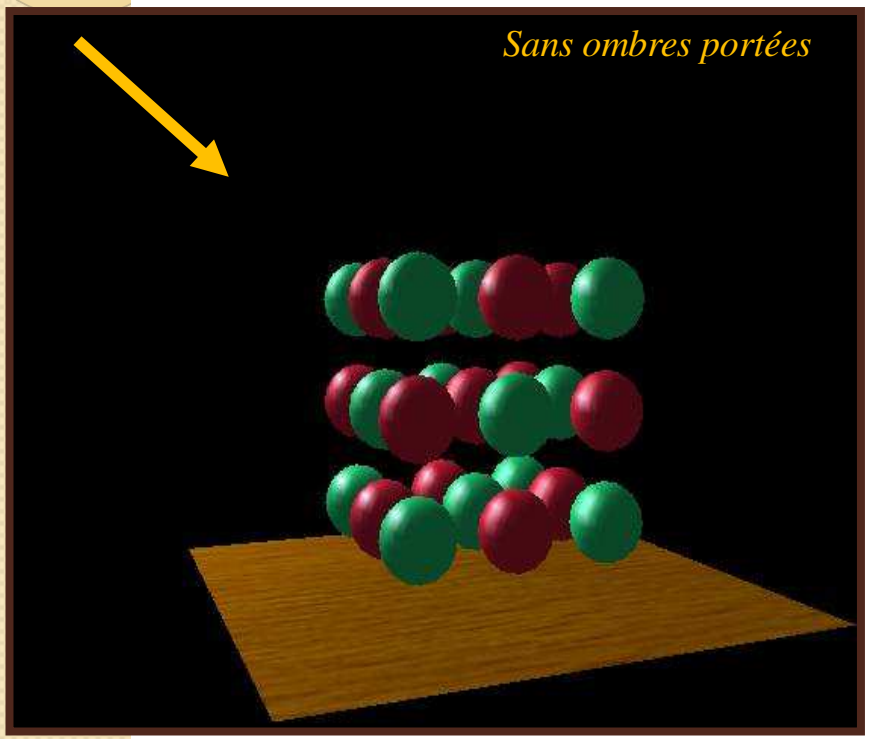
5	10	10	8	8	8	10
4	10	10	8	8	8	10
3	10	4	3	3	3	3
2	10	4	3	2	2	2
1	10	4	3	2	2	2
0	10	10	3	2	2	2
	0	1	2	3	4	5

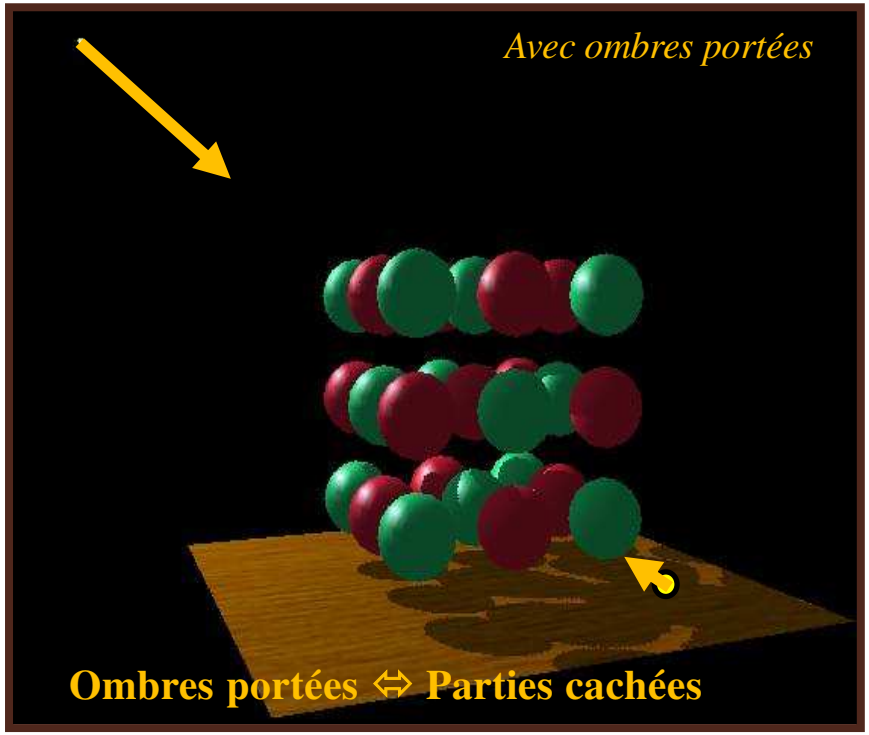
y, x

Z-buffer representation



Ombres portées



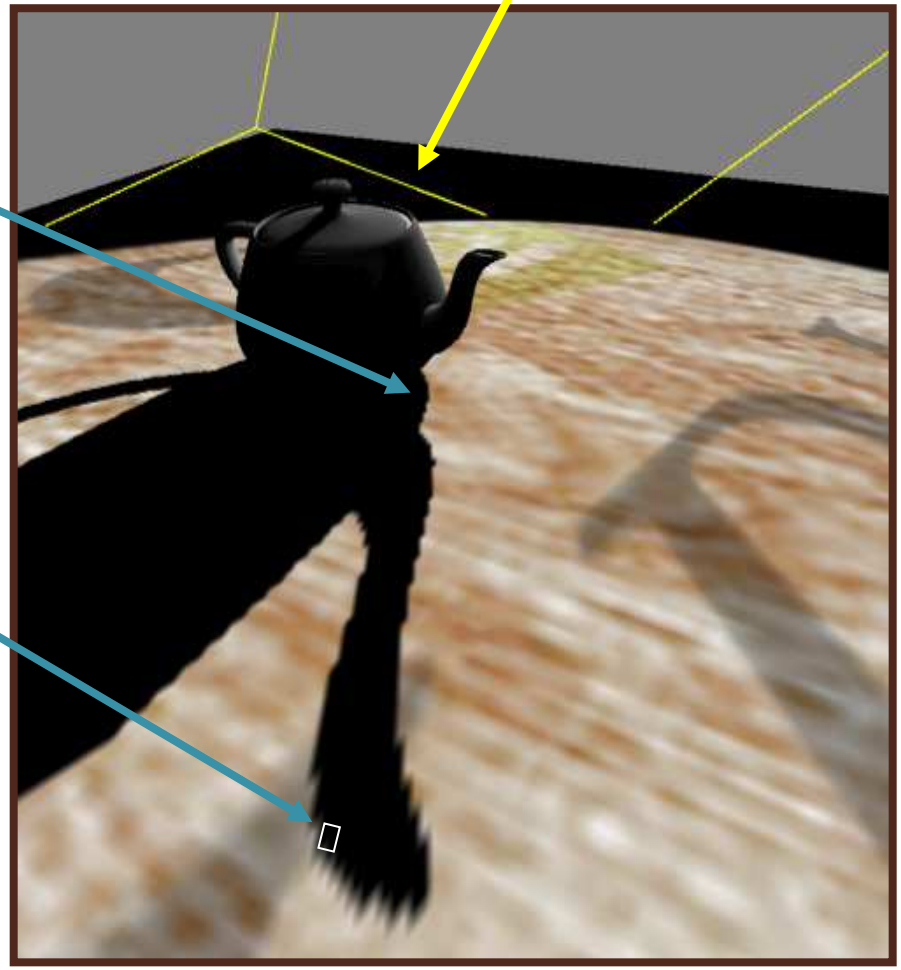


Combien de fois l'algorithme de parties cachées doit-il être appelé ?

Texture d'ombre

problème d' « aliasing » avec le Z-buffer

source de lumière



Contours très précis près de la source

Contour très grossier loin de la source

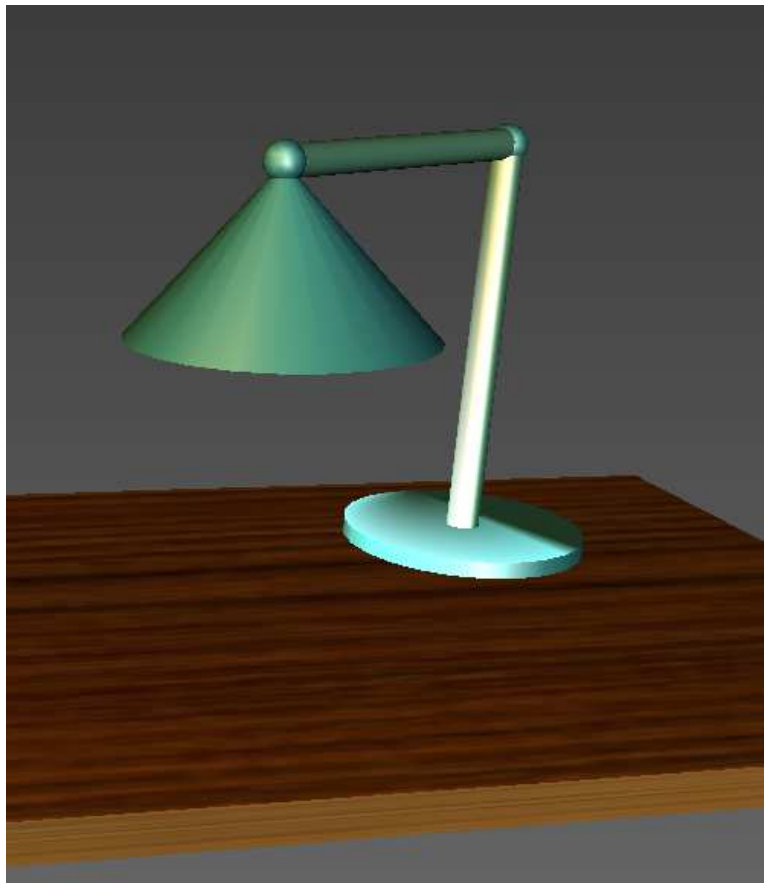
Un seul pixel dans le Zbuffer de la source

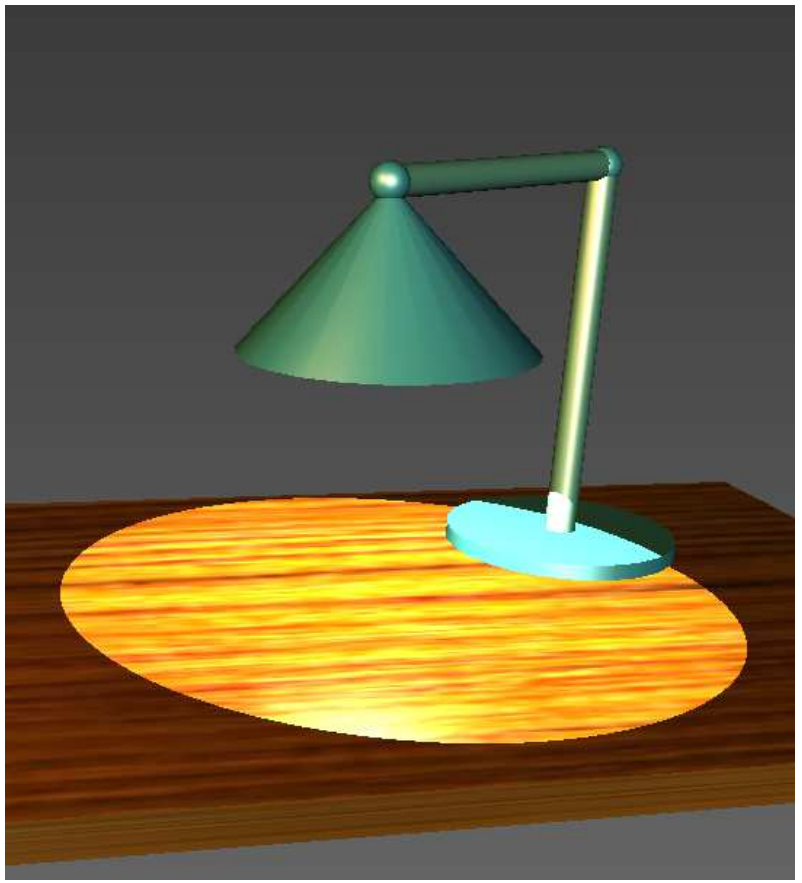
Solution : Irregular Z-buffer

Brackeys :

- Lighting in Unity,
- RealTime Lighting in Unity

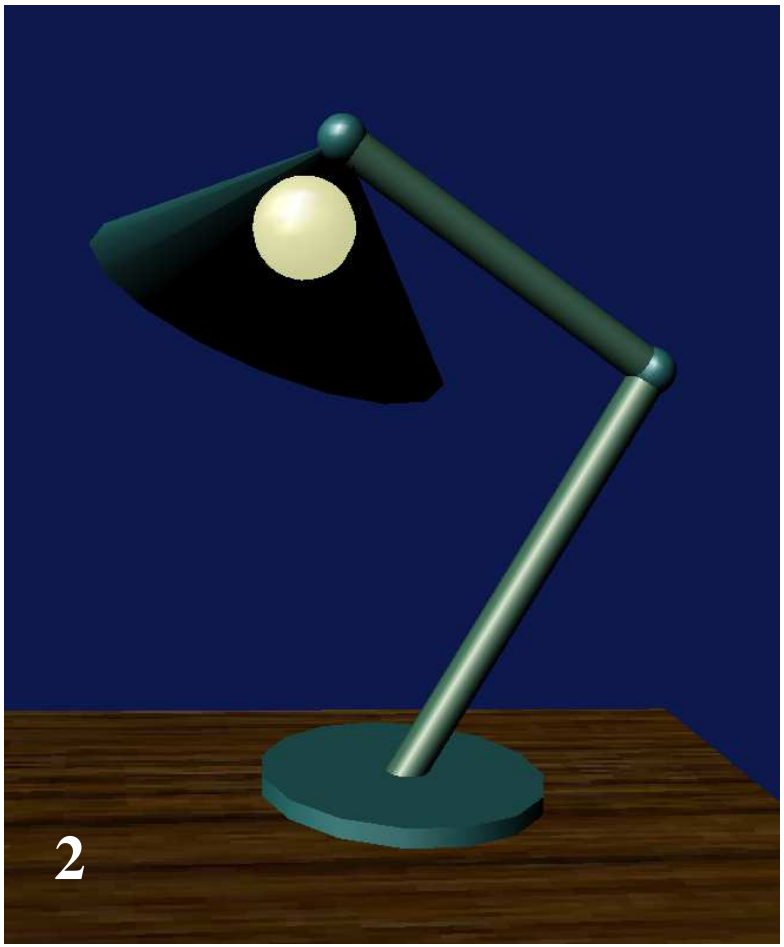
Exercice





Pourquoi l'éclairage de la table n'est pas réaliste sur l'image de g

Exercice

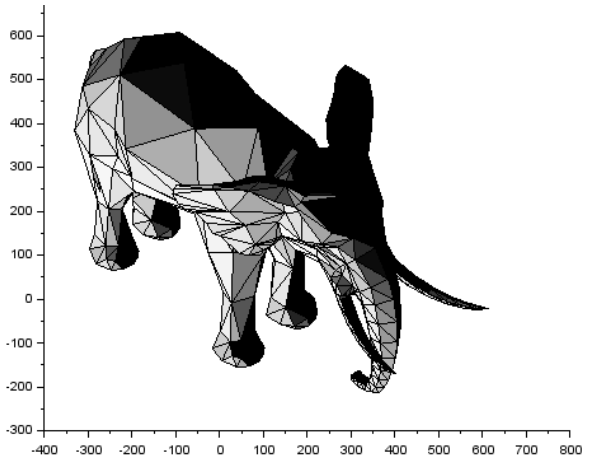


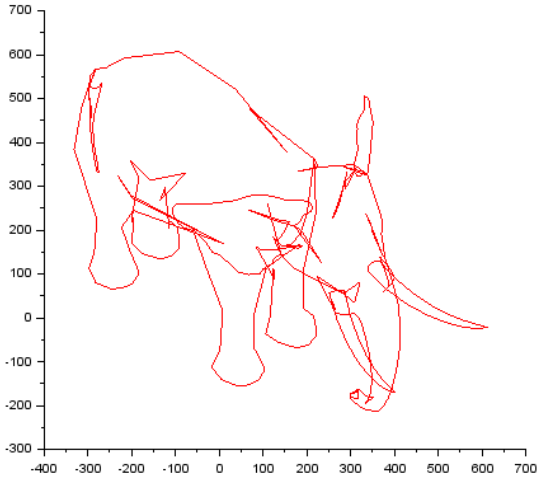
Localisez et interprétez les erreurs sur les images 1 et 2 ?

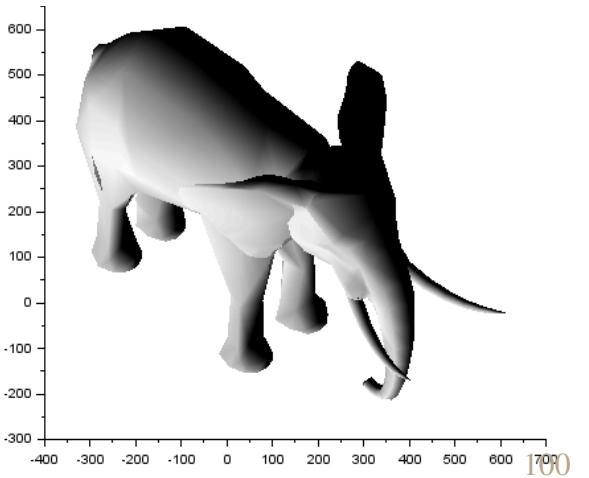
TP 3 : Shader

Eclairage d'une triangulation (3D)

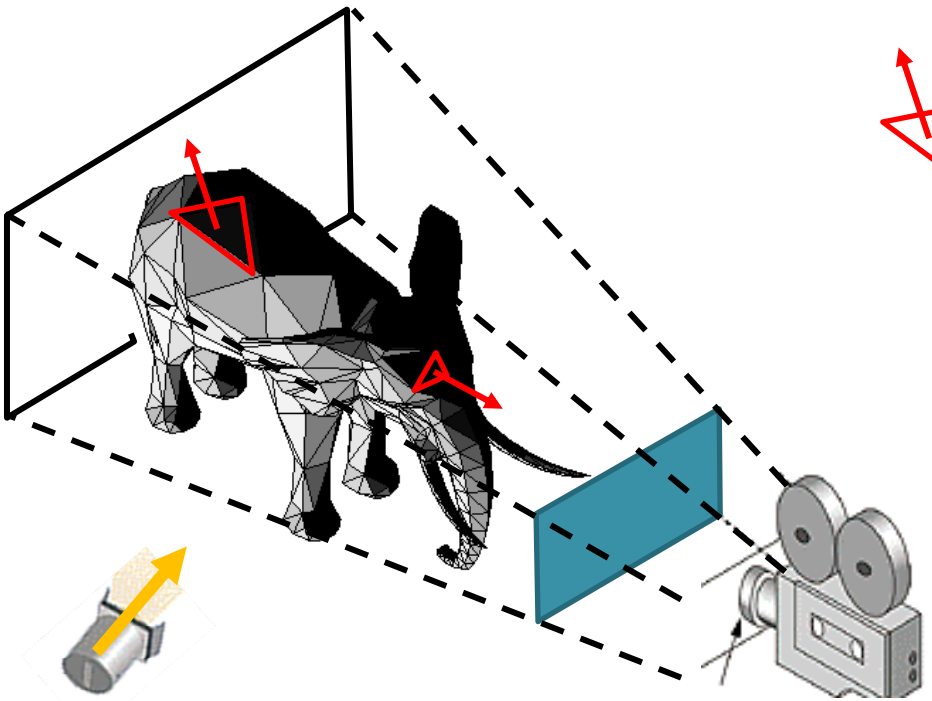
1. FLAT SHADING
2. SMOOTH SHADING
3. Silhouettes
4. FLAT and SMOOTH (critère d'angle)





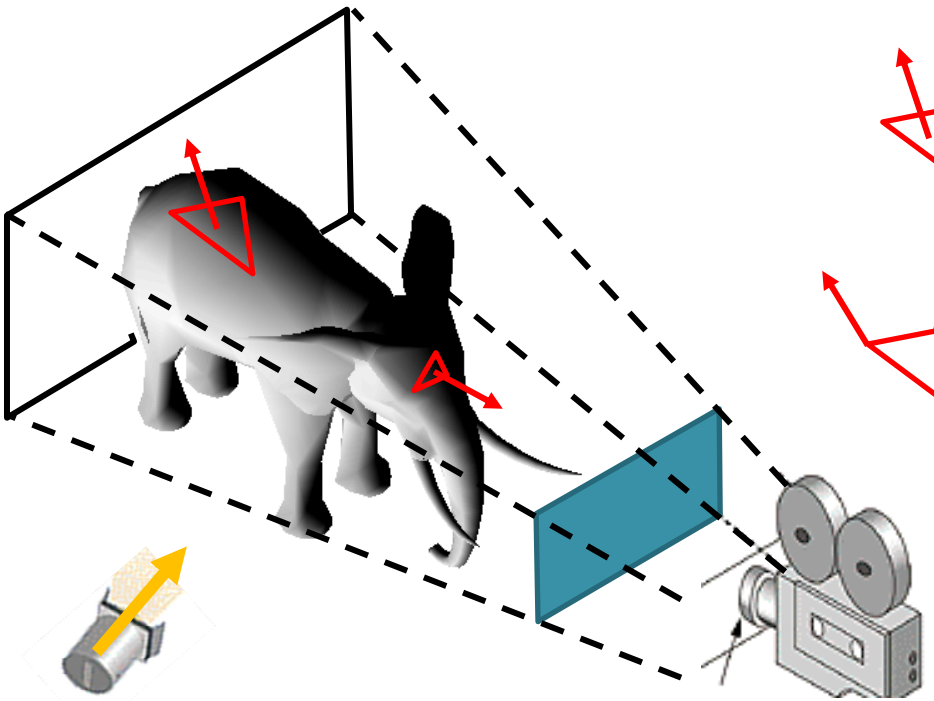


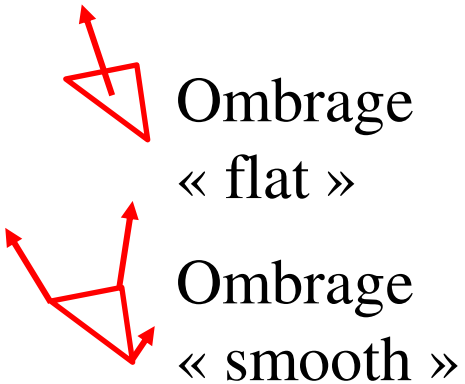
Le « flat shading »



Ombrage
« flat »

Le « gouraud shadding »

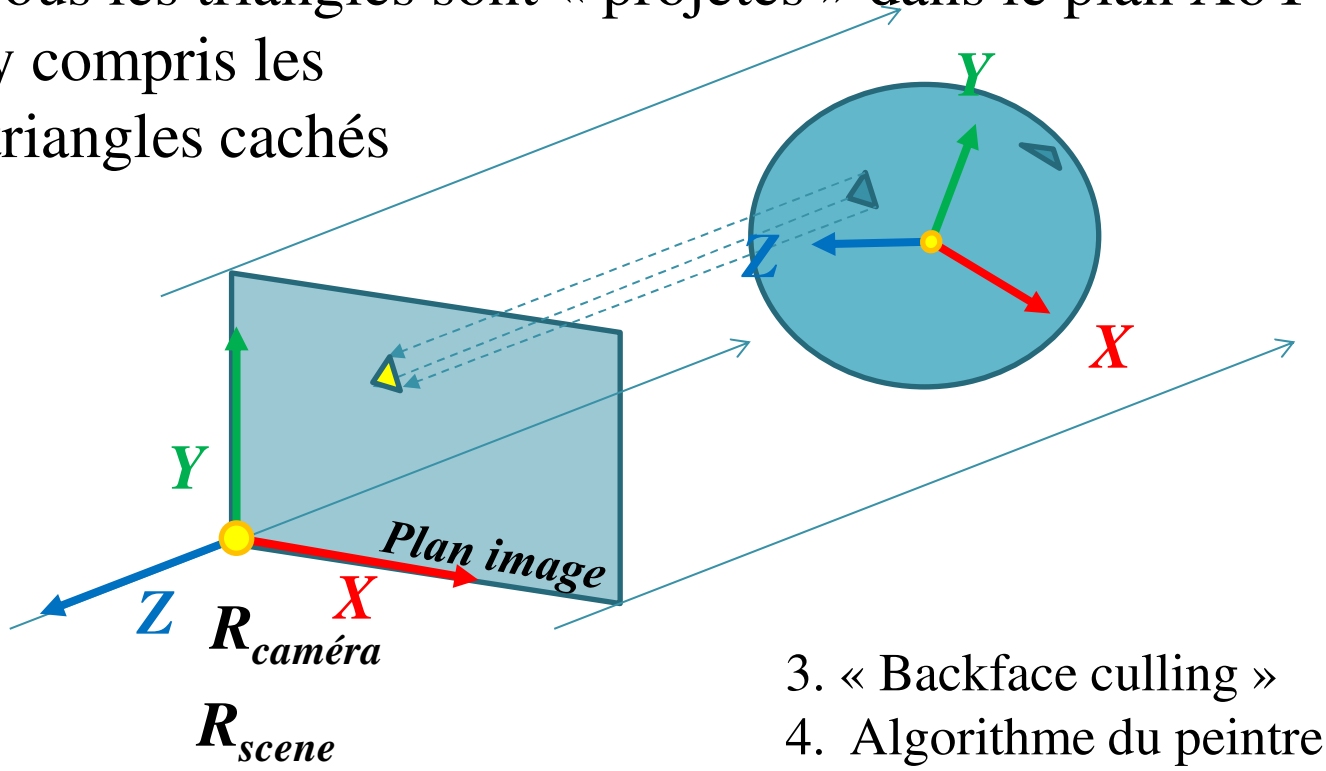




TP 2 : Pré-requis

- Triplot2d(XYZcoord, itrnoe, couleur)

Tous les triangles sont « projetés » dans le plan XoY y compris les triangles cachés

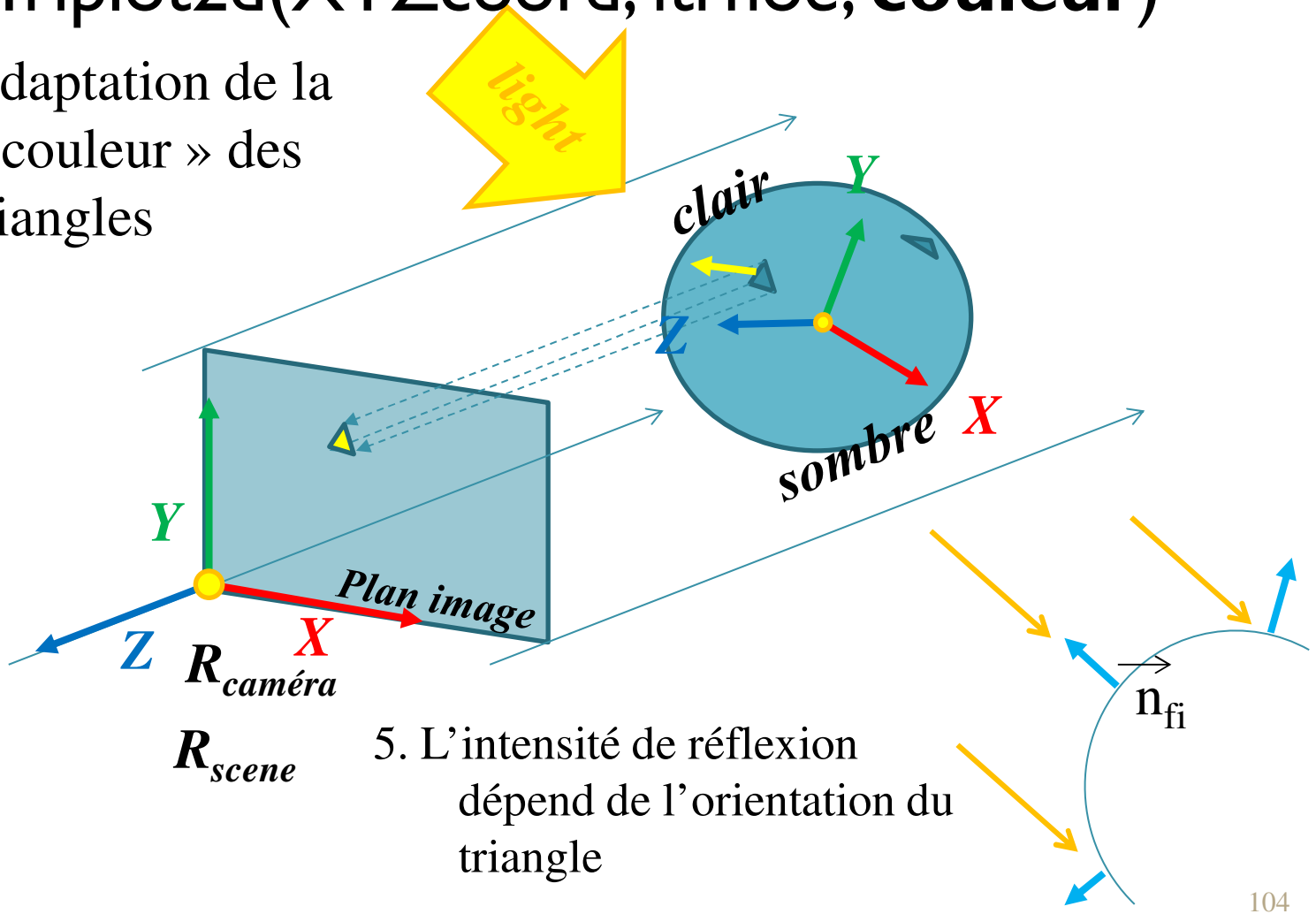


3. « Backface culling »
4. Algorithme du peintre

TP 2 : Pré-requis (2)

- Triplot2d(XYZcoord, itrnoe, **couleur**)

Adaptation de la « couleur » des triangles

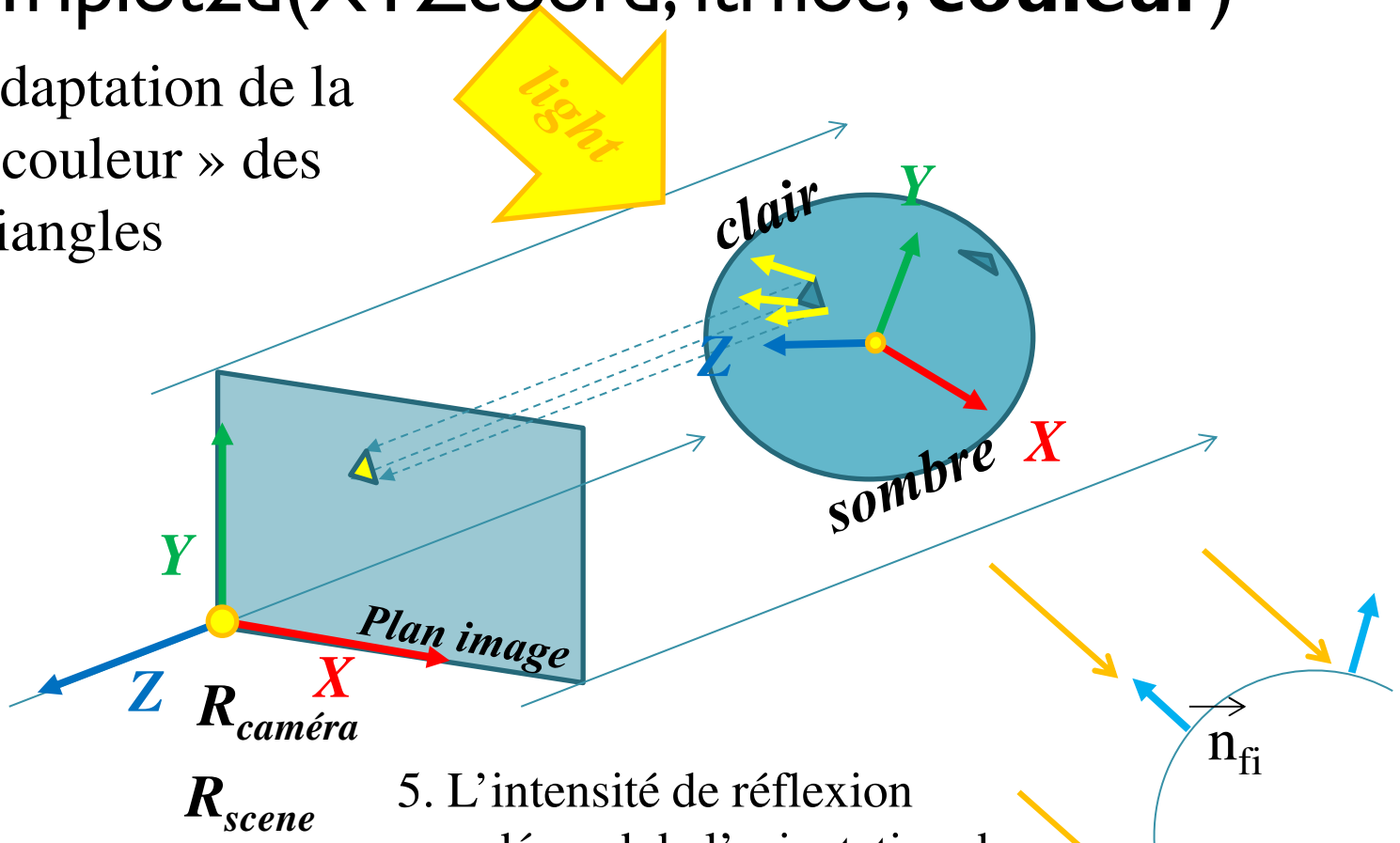


5. L'intensité de réflexion dépend de l'orientation du triangle

TP 2 : Pré-requis (3)

- Triplot2d(XYZcoord, itrnoe, **couleur**)

Adaptation de la « couleur » des triangles



5. L'intensité de réflexion dépend de l'orientation du triangle et de ses voisins