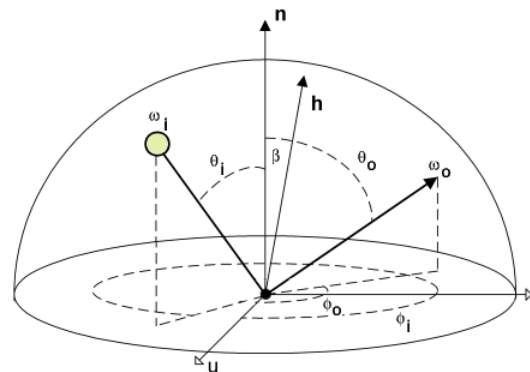


Informatique Graphique 3D & Réalité Virtuelle

Synthèse d'Images : Visibilité, Apparence, Texture

Tamy Boubekur



Rasterization & Lancer de Rayon

ALGORITHMES DE VISIBILITÉ

Visibilité

- **Objectif:** déterminer quels primitives sont visibles/cachées depuis un point donné, comme par exemple depuis :
 - une caméra > formation d'une image
 - une source de lumière > éclairage
- Détermine les 2 grandes classes d'algorithmes de rendu:

Projection

Rasterization

- Alg. du peintre
- Z-Buffer
- Ombrage Différé

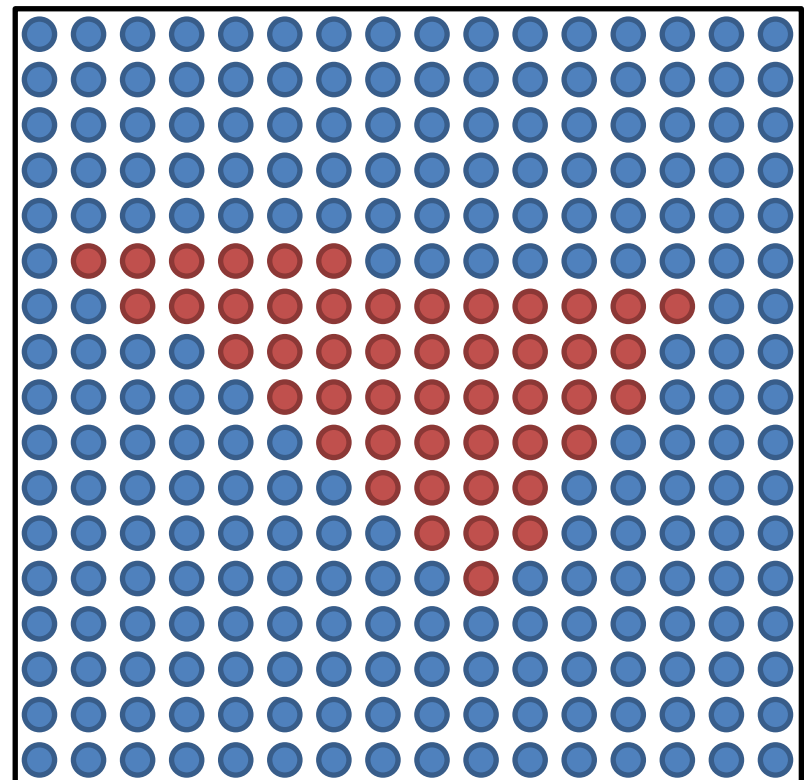
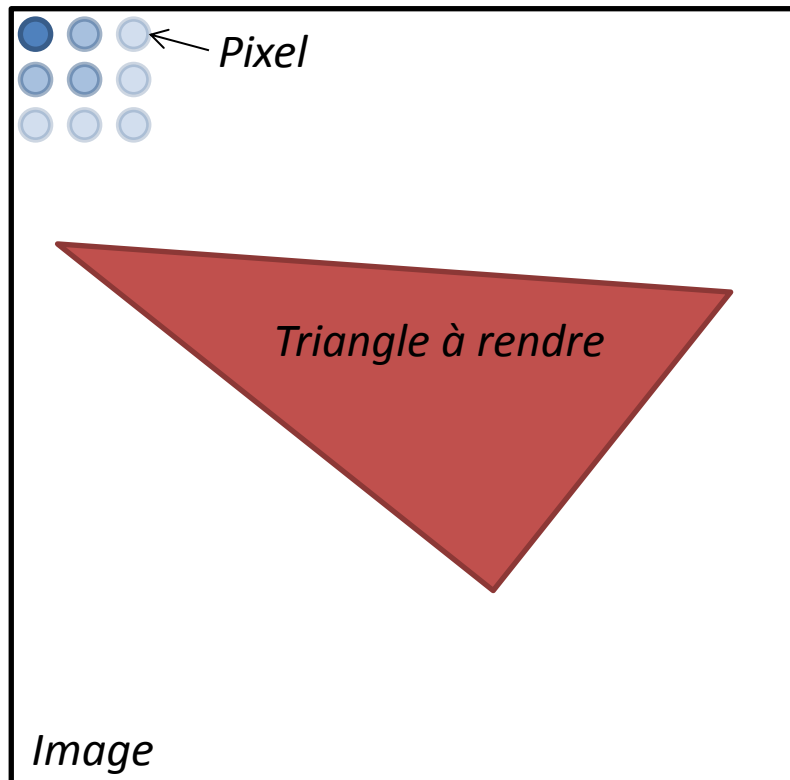
Lancer de rayon

Ray casting/tracing

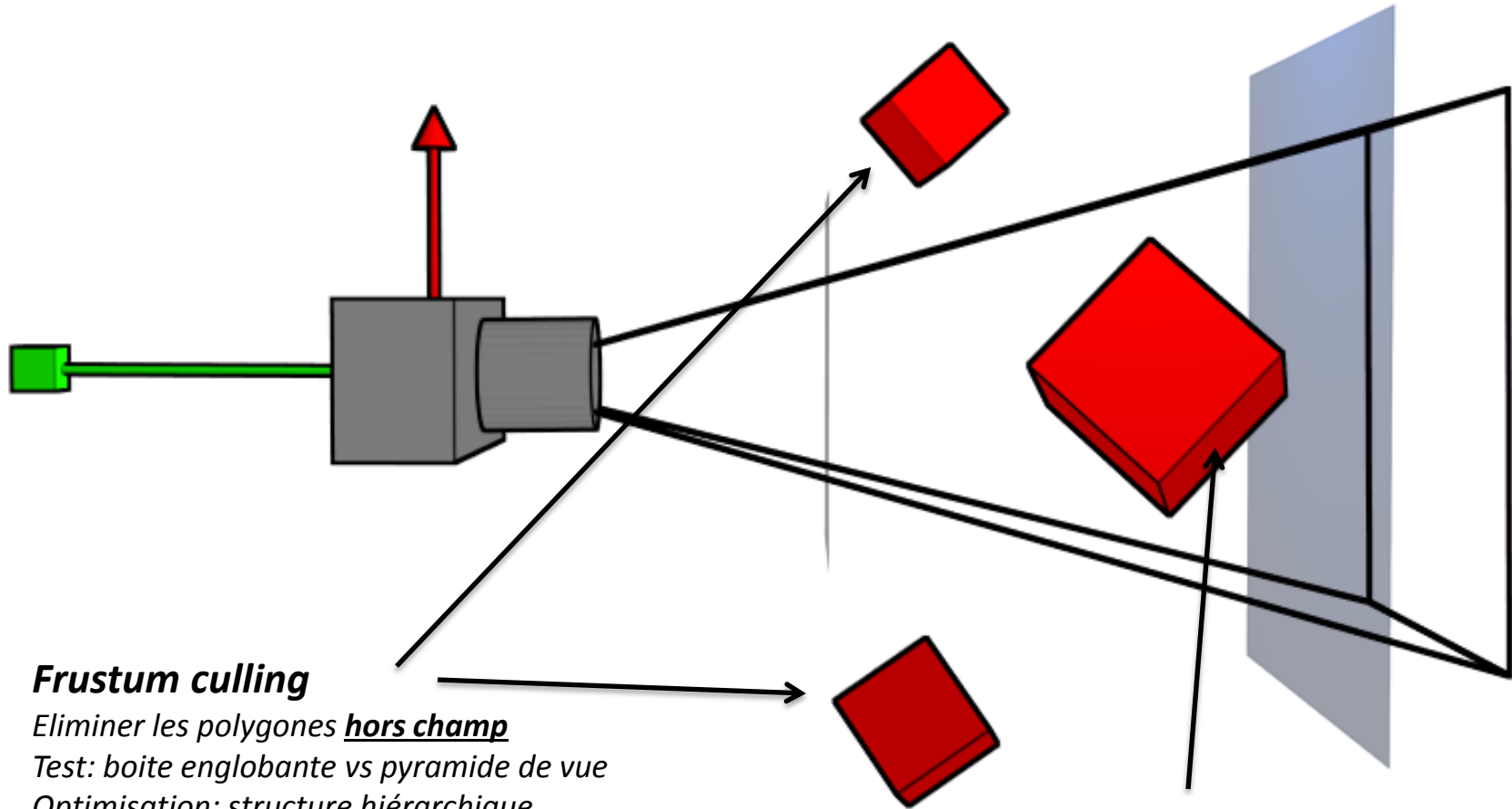
- Intersection rayon/triangle
- Récursion

Rasterisation

- Discrétisation d'un polygone dans une grille (image)
 - 1 triangle > n **fragments**
- Plusieurs algorithmes alternatifs
 - Pixel = fragment visible
- Sans précaution, tous les polygones de la scène sont *traités*
 - > Elimination de primitives



Elimination de primitives (Culling)



Frustum culling

Eliminer les polygones hors champ

Test: boîte englobante vs pyramide de vue

Optimisation: structure hiérarchique

Occlusion culling:

- éliminer les objets occultés dans la pyramide de vue
- Compliqué > peu utilisé

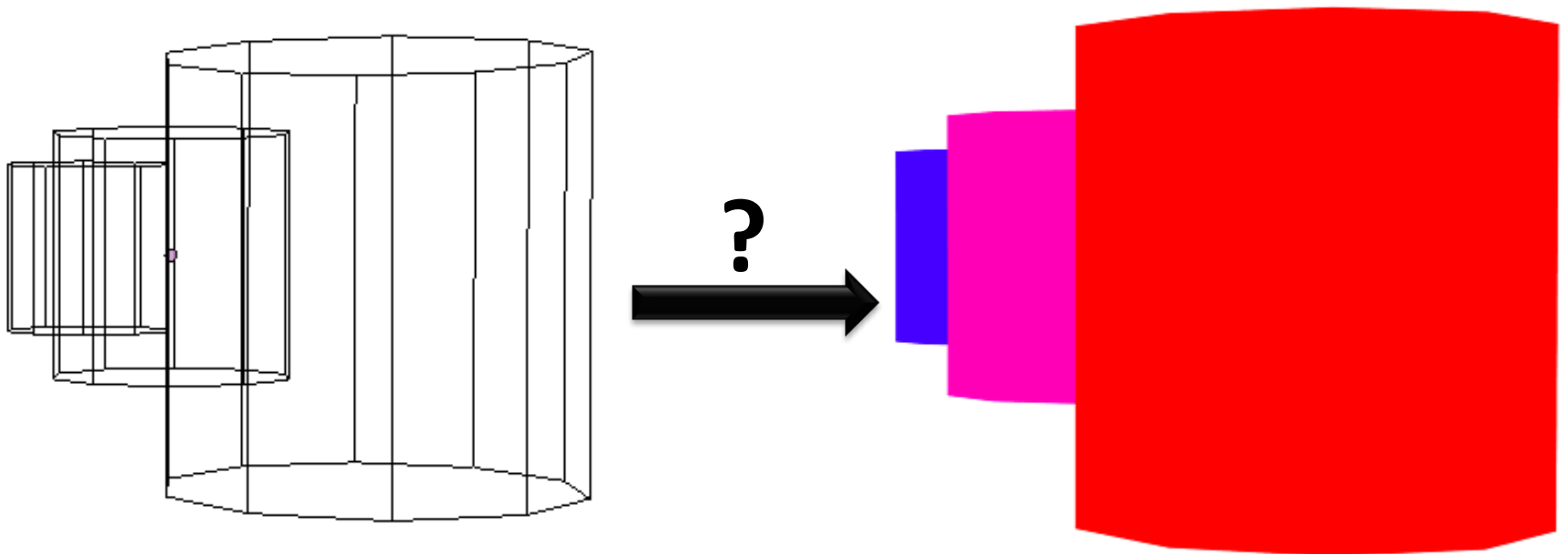
Backface culling

Elimination des faces arrières

Test: normal vs caméra orientation

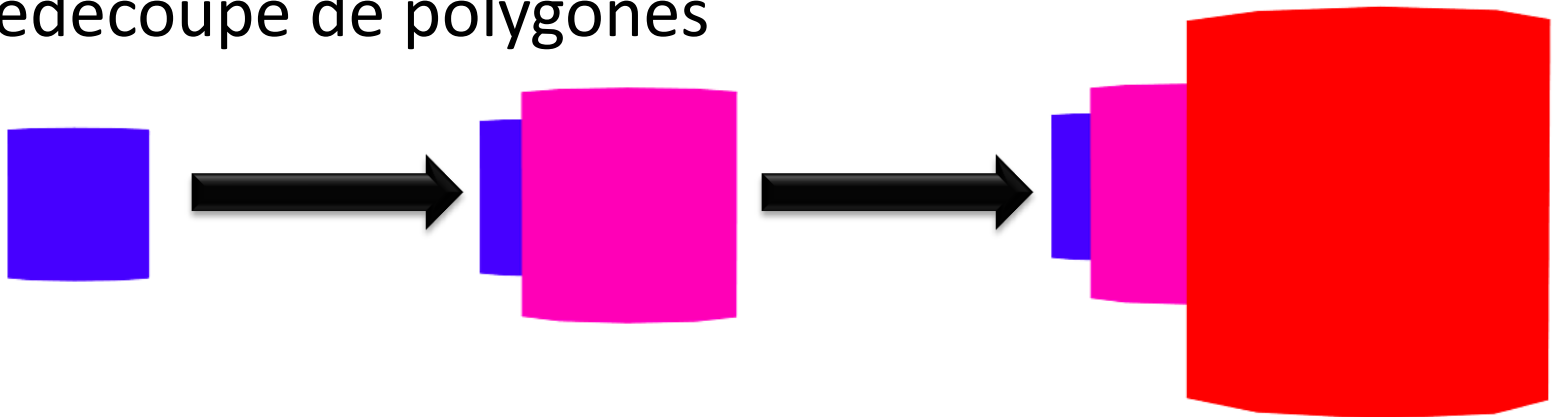
Visibilité dans le champ de vue

- Comment déterminer les parties visibles et les parties cachées de la géométrie des objets depuis un point donné (e.g. position de la caméra) ?



Algorithme du peintre

- Ordonnancement général des polygones le long de l'axe de vue
- Dessin de *loin en proche* de la liste ordonnée
- Lent > Optimisé via un **BSP-Tree**, mais *statique*
- Cas ambigu : intersection de polygones
 - Redécoupe de polygones



Rasterization avec Z-Buffer

Idée: maintenir un tampon (buffer) ZB de la même taille que le tampon couleur FB de l'écran, mais stockant pour chaque pixel la **profondeur** de la géométrie le recouvrant

Algorithme

```
Pour chaque polygone t :  
    Si t hors-champ ou t non face caméra  
        Ignorer t  
    Rasterizer t  
    Pour chaque fragment (x,y) de t :  
        c := couleur de t en (x,y)  
        z := distance fragment-caméra  
        Si ZB(x,y) > z alors :  
            FB(x,y) := c  
            ZB(x,y) := z  
        Sinon  
            Ignorer (x,y)
```



Frame-buffer
(FB)



Z-Buffer
(ZB)

😊 Rapide, support GPU, linéaire en temps

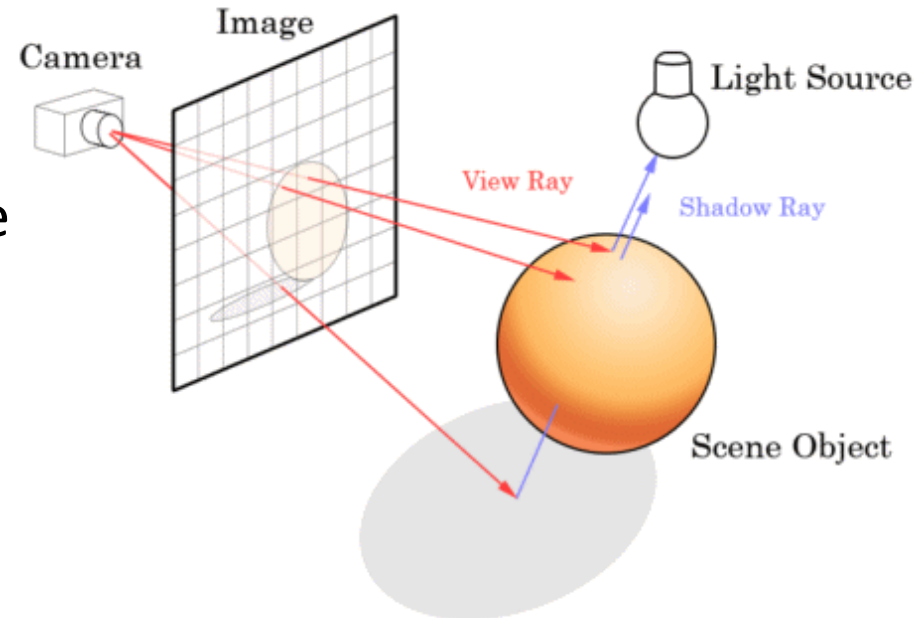
😞 Plusieurs polygones par pixel / aliasing

Lancer de Rayon (Ray Tracing)

Idée: partir du point de vue et chercher pour chaque pixel le premier objet intersectant la ligne de vue au travers du pixel

Algorithme

```
Pour chaque pixel (x,y)
  r := rayon caméra-pixel
  e = +∞
  FB (x,y) = (0,0,0)
  Pour chaque polygone t
    x := intersection (r, t)
    Si x != null
      d = distance camera-x
      Si d < e
        e = d
        FB (x,y) = couleur de X
```



Accélération:

- Ranger les polygone dans un kd-tree
 - Pré-calcul en $O(n \log n)$
- Utiliser pour le kd-tree pour la recherche d'intersection
 - Coût par pixel: $O(\log n)$

😊 Simple, facilement généralisable

😞 Couteux

Intersection Ray-Triangle

IntersectionRayTriangle (o, w, p_0, p_1, p_2) :

$$e_0 = p_1 - p_0$$

$$e_1 = p_2 - p_0$$

$$n = e_0 \wedge e_1 / |e_0 \wedge e_1|$$

$$q = w \wedge e_1$$

$$a = e_0 \cdot q$$

si $n \cdot w \geq 0$ ou $|a| < \text{epsilon}$

retourner null

$$s = (o - p_0) / a$$

$$r = s \wedge e_0$$

$$b_0 = s \cdot q$$

$$b_1 = r \cdot w$$

$$b_2 = 1 - b_0 - b_1$$

si $b_0 < 0$ ou $b_1 < 0$ ou $b_2 < 0$

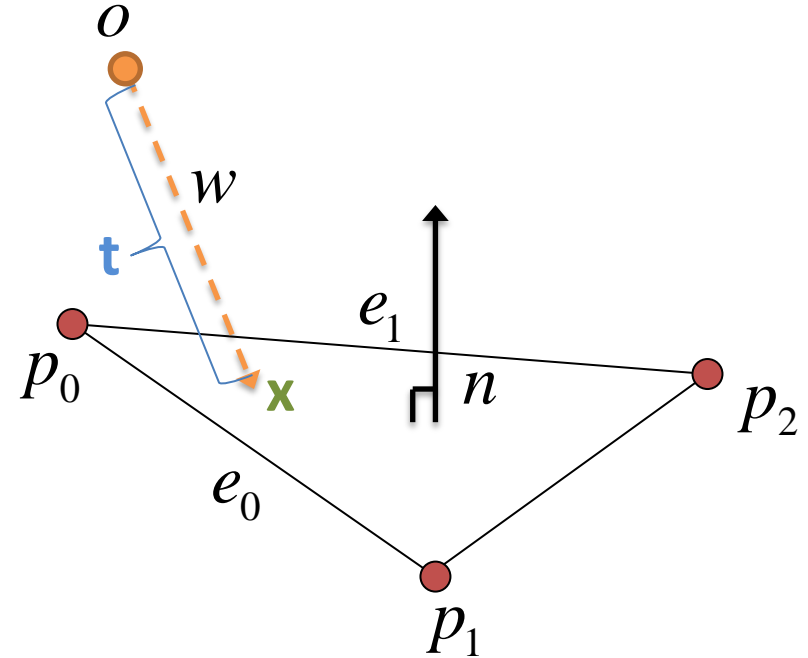
retourner null

$$t = e_1 \cdot r$$

si $t \geq 0$

retourner $[b_0, b_1, b_2, t]$

retourner null



Intersection exprimée :

- selon le triangle, en coordonnées *barycentriques*
 $x = b_0 * p_0 + b_1 * p_1 + b_2 * p_2$
- Selon la forme paramétrique du rayon : $x = o + t * w$

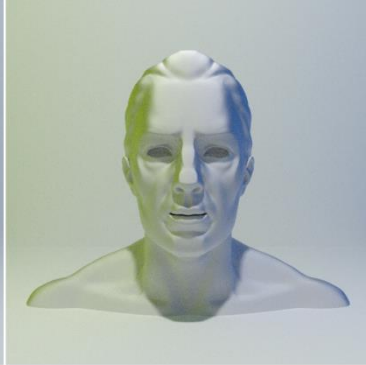
Lumière, réflectance, couleur.

APPARENCE

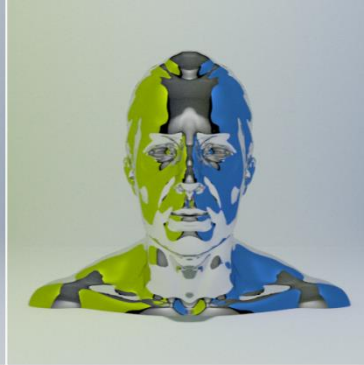
Apparence



Surface Diffuse



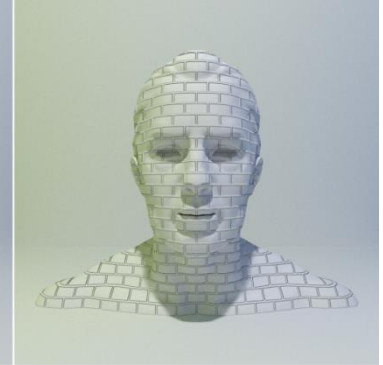
Surface *Glossy*



Surface Spéculaire

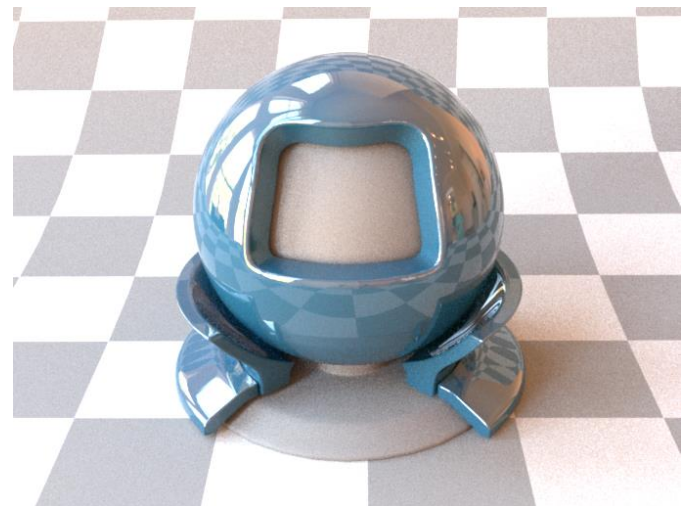


Carte Couleur

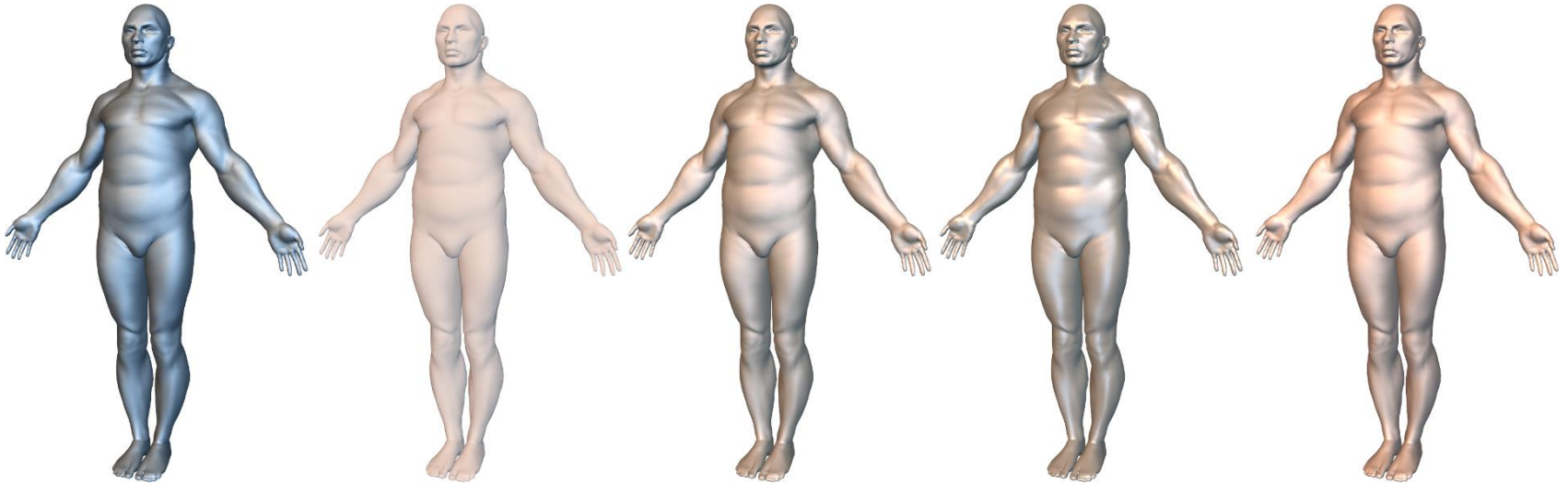


Carte Relief

- Matériaux
- Textures
 - Variation des paramètres des matériaux sur la surface
- *Meso-* et *micro-structure* de la surface



Apparence

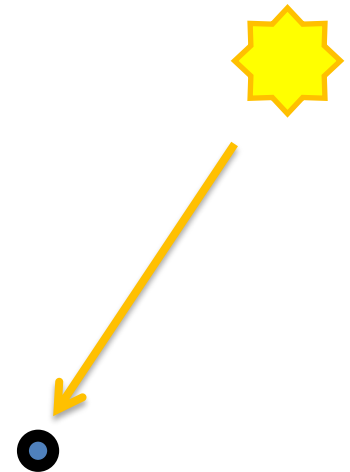


- Couleur en un point p :
 - depuis un point de vue donné
 - pour une scène donnée
- Fonction de:
 - L'éclairage (illumination) en p
 - La **réflectance** du matériau en p

Eclairage

Radiance en un point:

- Mesure radiométrique décrivant la quantité d'énergie passant en un point pour une direction donnée.
- Exprimée en Watt par Stéradian par Mètre-carré ($\text{W} / \text{m}^2 \text{sr}$)



Champ de Lumière

- Ou *Light Field*
- $L(\mathbf{p}, \boldsymbol{\omega}), \mathbf{p} \in \mathbb{R}^3, \boldsymbol{\omega} \in S^2$
- *Lumière émise en \mathbf{p} , dans la direction $\boldsymbol{\omega}$*
- Résulte
 - *Des sources primaires (surfaces emissives, source virtuelles)*
 - *Du transport lumineux dans la scène*

Sources Virtuelles de Lumière

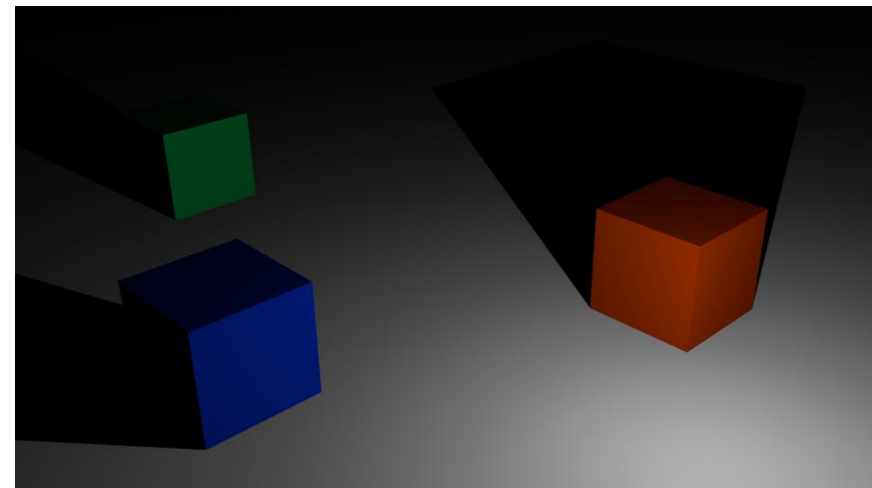
- Intensité
- Couleur:
 - En général: un triplet RVB attaché à la source
 - Modélisation physique: **spectre complet**
- Type :
 - Sources **ponctuelle** : définit par une position
 - Emet de l'énergie dans toutes les directions
 - Source **directionnelle** : une position + une direction
 - Rayons parallèles
 - Souvent utilisée pour modéliser la lumière du soleil
 - **Spot**: portion angulaire d'une source ponctuelle
 - **Source Etendue** (*Area Light*) : un morceau de surface émettant de la lumière
 - Ombres douces
 - Peut-être défini à partir de n'importe quelle géométrie de la scène



Atténuation Lumineuse

- Modélise l'énergie reçu en un point à une distance d de la source
- Typiquement caractérisé en information par un triplet de valeurs:
 - a_c le coefficient d'atténuation constante
 - a_l le coefficient d'atténuation linéaire
 - a_q le coefficient d'atténuation quadratique

$$L(d) = \frac{L}{a_c + a_l \cdot d + a_q \cdot d^2}$$



Environnement Lumineux

Analytique

- Ensemble de sources lumineuses

Echantillonné

- Exemple: fonction (hémi)-sphérique décrivant l'éclairage (supposé) infiniment distant
 - Capturée via un *lightprobe*

ou un panorama de photos

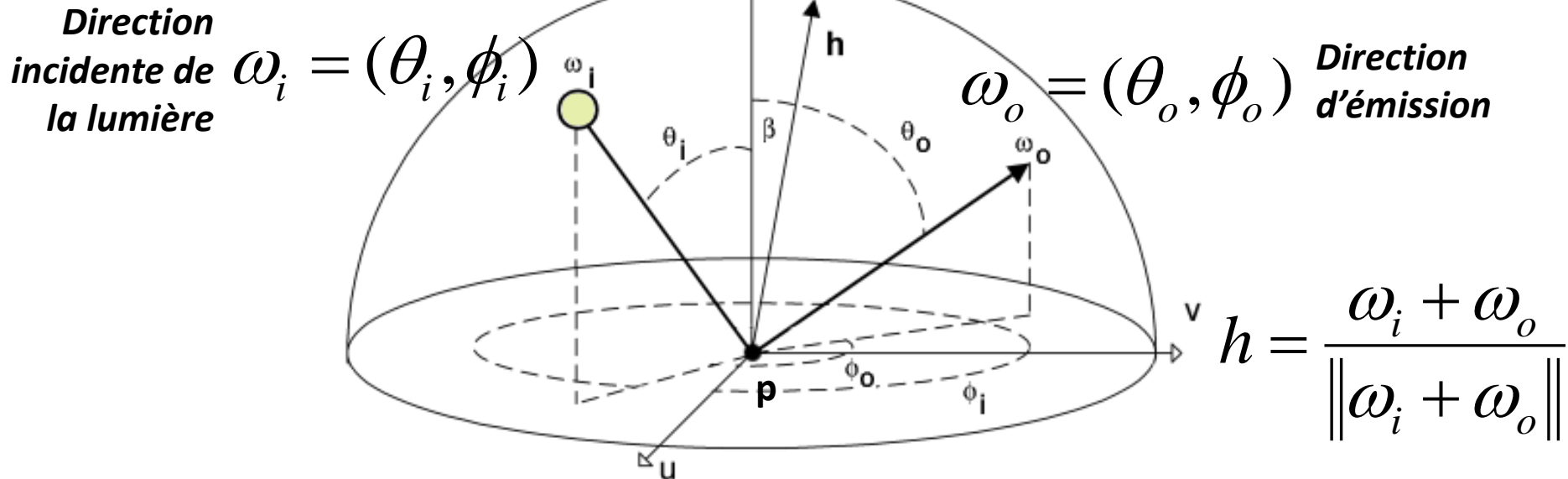
- En général en haute dynamique (imagerie HDR)
- Analyse fréquentielle



Equation du rendu

$$\begin{array}{c} \text{Intensité} \\ \text{renvoyée} \end{array} \downarrow L_o(\omega_o) = \begin{array}{c} \text{Emission} \\ \text{intrinsèque} \end{array} \downarrow L_e(\omega_o) + \int_0^{2\pi} \int_0^{\pi/2} \begin{array}{c} \text{Lumière} \\ \text{Reçu} \end{array} \downarrow L_i(\omega_i) \begin{array}{c} \text{Réflectance} \end{array} \downarrow f(\omega_i, \omega_o) \cos \theta_i d\theta_i d\phi_i$$

(Omis ensuite)



Equation du rendu

Simplification pour une **Source ponctuelle** unique

$$L_o(\omega_o) = L_i(\omega_i) f(\omega_i, \omega_o) (n \cdot \omega_i)$$

Plusieurs sources ponctuelles

$$L_o(\omega_o) = \sum_i L_i(\omega_i) f(\omega_i, \omega_o) (n \cdot \omega_i)$$

Evaluation dans le cas non ponctuel (e.g. sources étendues)
par la méthode de *Monte-Carlo*

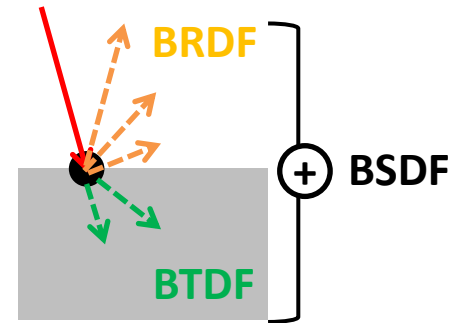
Eclairage direct

Considérer un point de surface indépendamment

- Avec ou sans ombre porté
- Pas d'échange lumineux avec les autres points de la scène
- Modélisation du matériau par une BRDF (*Bidirectional Reflectance Distribution Function*)
 - Plusieurs modèles analytiques existent
 - Paramètres
 - uniforme sur une surface
 - Ou variant et spécifié par une carte sur la surface
 - carte couleur/albedo diffus
 - Carte de brillance
 - etc

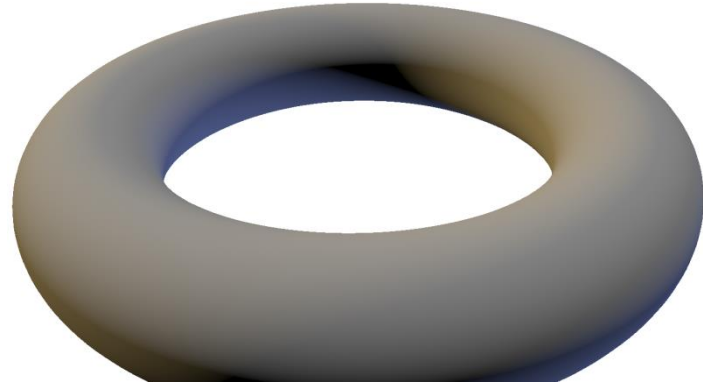
Réflectance

- Définit en un point par la fonction de distribution de réflectance bidirectionnelle ou **BRDF**
 - **BRDF** : composante **réflective** de la **BSDF (dispersion)**
 - Pour l'instant, on oublie la composante transmissive (**BTDF**)
- Défini la réflexion de l'énergie reçu (radiance)
- Composantes classiques :
 - **Diffuse** : distribution de l'énergie dans toutes les directions
 - **Spéculaire/Fresnel** : réflexion directionnelle
 - Miroir : réflexion spéculaire parfaite

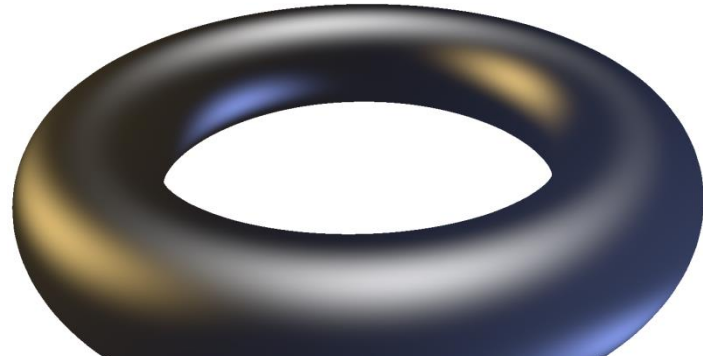


Compositions

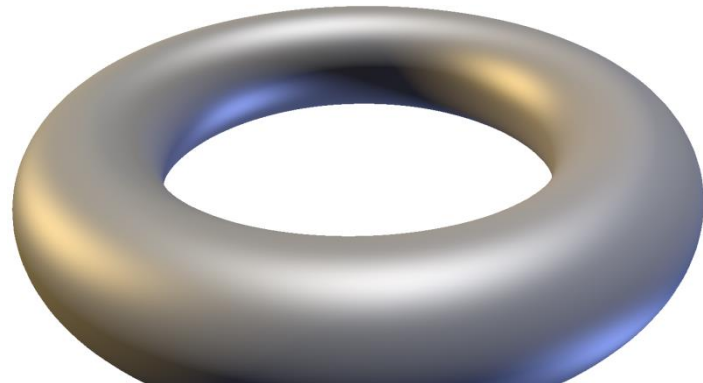
Terme diffus



Terme spéculaire



Diffus + Spéculaire



BRDF

- Définit la **micro-structure** d'un matériau
- Cas classique : une fonction à 4 dimensions

$$f : S^2 \times S^2 \rightarrow [0,1]$$

$$\omega_i \times \omega_o \rightarrow r$$

$$\omega_i = (\theta_i, \phi_i) \quad \omega_o = (\theta_o, \phi_o) \quad \omega_h = \frac{\omega_i + \omega_o}{\|\omega_i + \omega_o\|}$$

Lumière incidente *Direction d'émission* *HalfVector*

- Valeur en inférieure ou égale à 1

$$L_o(\omega_o) = L_i(\omega_i) f(\omega_i, \omega_o) (n \cdot \omega_i)$$

BRDF Physiquement Plausibles

- Respecte la **réciprocité d'Helmutz**

$$f(\omega_i, \omega_o) = f(\omega_o, \omega_i)$$

- **Conservative**

$$\int_{\Omega} f(\omega_i, \omega_o) \cos \theta_o d\omega_o \leq 1$$

- **Positivité**

$$f(\omega_i, \omega_o) \geq 0, \forall \omega_i, \omega_o$$

Propriétés

- Isotropie/Anisotropie
- Nombre réduit de paramètres
- Séparabilité « diffus/spéculaire »
$$f(\omega_i, \omega_o) = f^d(\omega_i, \omega_o) + f^s(\omega_i, \omega_o)$$
- Evaluable par la méthode de Monte-Carlo

Note sur la couleur

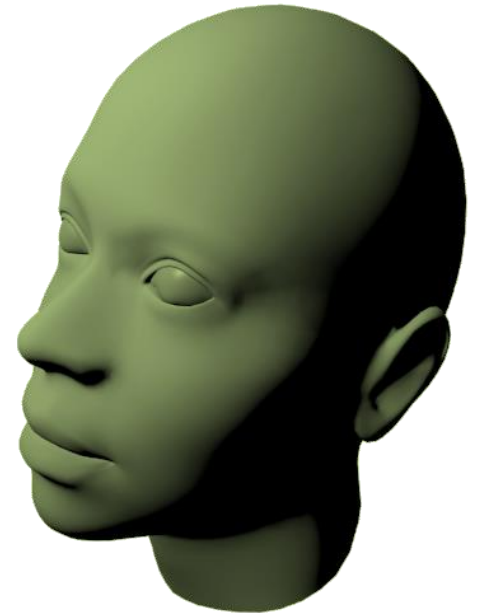
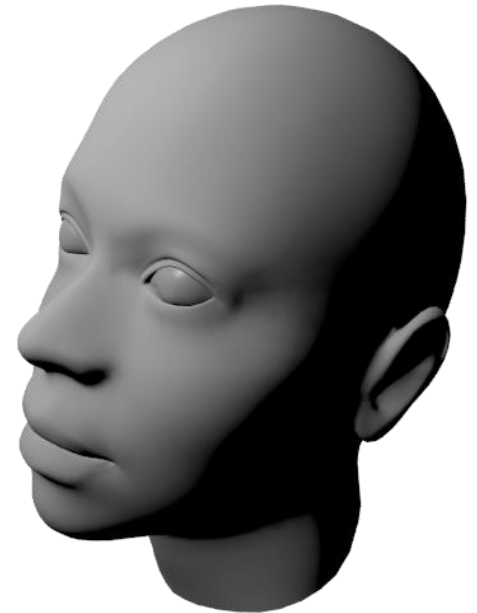
- Pour l'instant, on considère le cas où la BRDF s'applique aux trois canaux couleur de la même manière
- Albedo diffuse : couleur de base du matériau
- Couleur spéculaire :
 - Modèles physiques : dépend du coefficient de Fresnel et de la conductivité du matériau
 - Modèles empirique : spécifiée indépendamment

BRDF diffuse

- Modèles de Lambert

$$f^d(\omega_i, \omega_o) = \frac{k_d}{\pi} \text{ Coefficient Diffus}$$

- Standard
- Indépendant du point de vue
- Réutilisé dans la plupart des autres modèles, qui se concentrent sur les réflexions spéculaires dépendante du point de vue.



BRDF de Phong

- Terme spéculaire

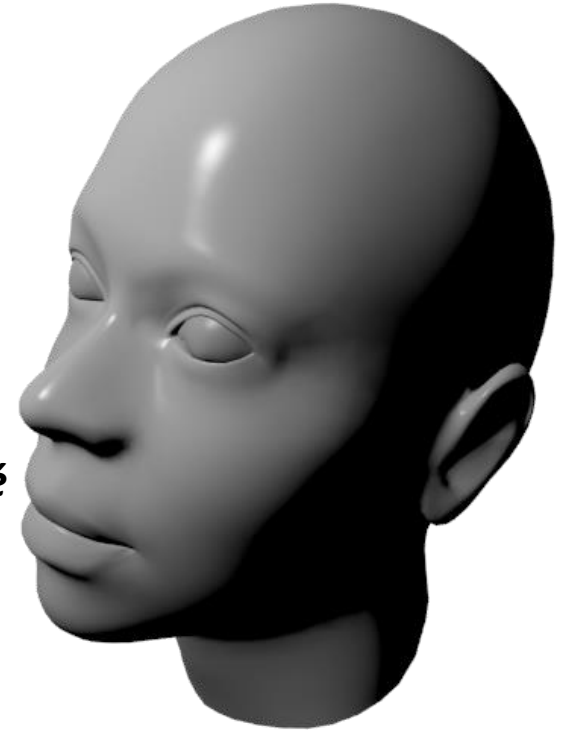
$$f^s(\omega_i, \omega_o) = k_s (r \cdot \omega_o)^s$$

$$r = 2n(\omega_i \cdot n) - \omega_i$$

Coefficient de spécularité

Brillance

- Indice de **brillance** (*shininess*)
- Terme diffus : Lambert
- *Modèle empirique non conservatif*

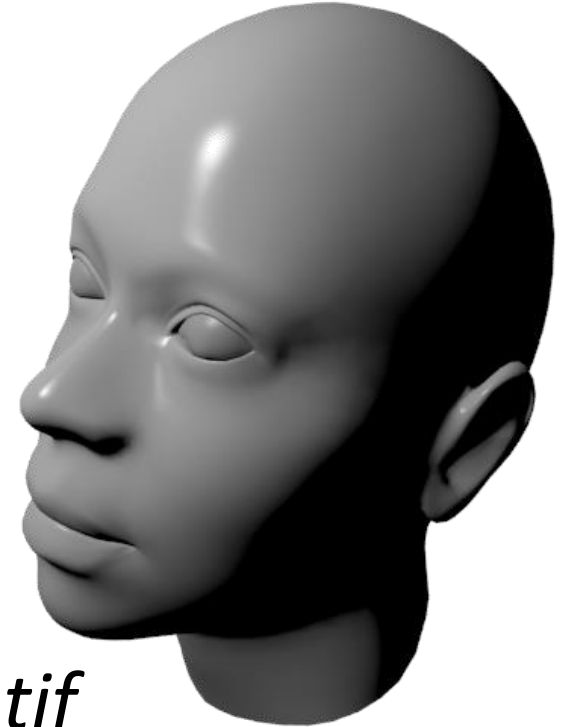


BRDF de Blinn-Phong

- Modèle de Phong modifié

$$f^s(\omega_i, \omega_o) = k_s (n \cdot \omega_h)^s$$

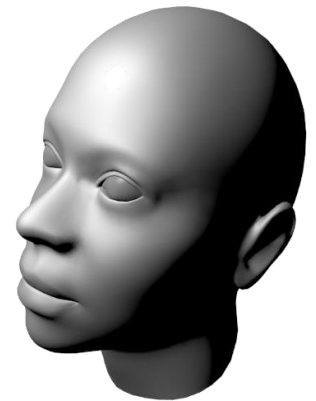
- Simple, efficace
- *Modèle empirique non conservatif*
- Normalisé en 2008



Modèles à Micro-Facettes

$$f^s(\omega_i, \omega_o) = \frac{\overset{\substack{\text{Distribution de} \\ \text{microfacettes} \searrow}}{D(\omega_h)} \overset{\substack{\text{Terme de} \\ \text{Fresnel} \searrow}}{F(\omega_i, \omega_h)} \overset{\substack{\text{Terme} \\ \text{Géométrique} \searrow}}{G(\omega_i, \omega_o, \omega_h)}{4(n \cdot \omega_i)(n \cdot \omega_o)}$$

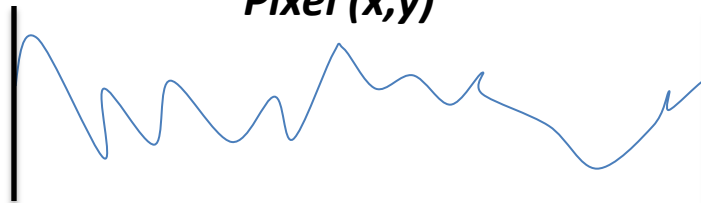
- Modèle statistique de la micro-géométrie
- $\alpha \in [0,1]$ la rugosité du matériau
 - En pratique souvent élevée au carrée
- Caractérisation géométrique



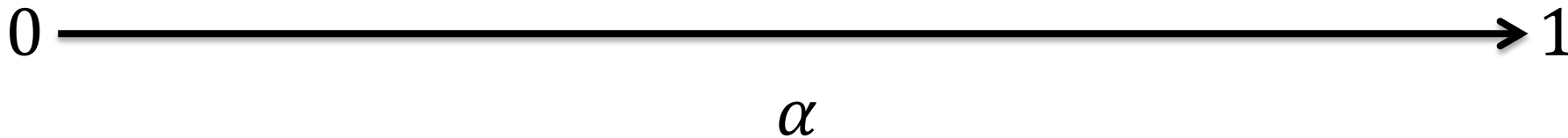
Pixel (x-1,y)

Pixel (x,y)

Pixel (x+1,y)



Rugosité



Distribution de micro-facettes

- D : Modèles la **distribution de normales** de la surface à l'échelle microscopique
- Exemple :
 - Distribution de Beckmann (BRDF Cook-Torrance, 1981-1982)
 - Distribution GGX/Trowbridge-Reitz
 - Et variantes
- Idéalement : F et G doivent être dérivés de D

Distribution de Beckmann

$$D(\omega_h) = \frac{1}{\pi \alpha^2 (n \cdot \omega_h)^4} e^{\frac{(n \cdot \omega_h)^2 - 1}{\alpha^2 (n \cdot \omega_h)^2}}$$

- Employée par la **BRDF de Cook-Torrance**
- Distribution des ondes électromagnétiques
- Déterministe
- Supposition :
 - toutes les facettes ont la même aire
 - Toute facette a une facette symétrique par rapport à la normale

Distribution GGX

Introduit par Trowbridge et Reitz (1975),
généralisé par Burley (2012)

$$D(\omega_i, \omega_o) = \frac{\alpha_p^2}{\pi \left(1 + \left(\alpha_p^2 - 1 \right) \cdot (n \cdot \omega_h)^2 \right)^2}$$

Standard industriel (Disney, Unreal Engine, etc)

Terme de Fresnel

Distingue les matériaux

- conducteurs
- diélectriques

Approximation de Schlick [1993]

$$F(\omega_i, \omega_h) = F_0 + (1 - F_0)(1 - \max(0, \omega_i \cdot \omega_h))^5$$

Avec $F_0 \in \mathbb{R}$ le coefficient de réflexion de Fresnel, dépendant du matériau:

- Plastique (diélectrique) : 0.3 à 0.5
- Aluminium (conducteur) : [0.91, 0.92, 0.92], « reflet coloré », variance significative selon la longueur d'onde

Terme Géométrique

- Modélise les effets d 'ombrage des micro-facettes
- Dépend de la rugosité et de la distribution D

Terme Géométrique Cook-Torrance

Distingue les effets de masquage et de l'ombrage inter-facettes

$$G = \min\left[1, \underbrace{\frac{2(n \cdot h)(n \cdot \omega_i)}{(\omega_o \cdot h)}}_{\text{Ombrage}}, \underbrace{\frac{2(n \cdot h)(n \cdot \omega_o)}{(\omega_o \cdot h)}}_{\text{Masquage}}\right]$$

Terme Géométrique GGX

Terme géométrique associé à la distribution de micro-facettes GGX

$$G(\omega_i, \omega_o) = G^1(\omega_i)G^1(\omega_o)$$

$$G^1(\omega) = \frac{2(n \cdot \omega)}{n \cdot \omega + \sqrt{\alpha^2 + (1 - \alpha^2)(n \cdot \omega)^2}}$$

Remarque

- Dans le cas purement physique : F et G doivent être dérivés de D

Matériaux métalliques

- Conducteurs
- Couleur réfléchie modulée par le matériau
 - En pratique si matériau métallique : réflexion avec la couleur de base
 - Sinon : la lumière incidente conserve sa teinte

« Un » modèle pour le rendu basé - physique

- Paramètre Matériau « PBR » (*Physically Based Rendering*)
 - Albedo (couleur diffuse moyenne)
 - Rugosité (entre 0 et 1)
 - « Métallique »
 - valeur binaire (conducteur ou pas)
- Voir le modèle plus complet proposé par Disney
 - *Physically-based Shading at Disney, SIGGRAPH 2012*

BRDF Basée-Donnée

- Échantillonnée à partir d'un véritable matériel
- Pas de forme analytique simple
- Représentée par un nombre réduit de coefficients une fois projetée une base de fonctions spécifique:
 - harmoniques sphériques
 - ondelettes

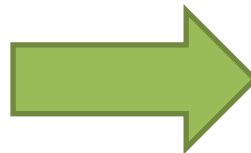
TEXTURES

Définition

- Une carte de valeurs (*map*)
 - Albedo, définit une distribution de points colorés sur les surfaces (*color map*) ; autre paramètre de la BRDF aussi modulables via une carte (rugosité, brillance)
 - Normal, définit une distribution de vecteur normaux sur une surfaces (*normal map*)
 - Déplacement, définit une distribution de vecteurs de déplacements sur une surface (*displacement map*)
- Textures 2D (carte scalaire ou vectorielle) plaquée sur une surface 3D via sa *paramétrisation UV*

Plaquage de texture

Texture couleur (image 2D RGB)



Maillage texturé



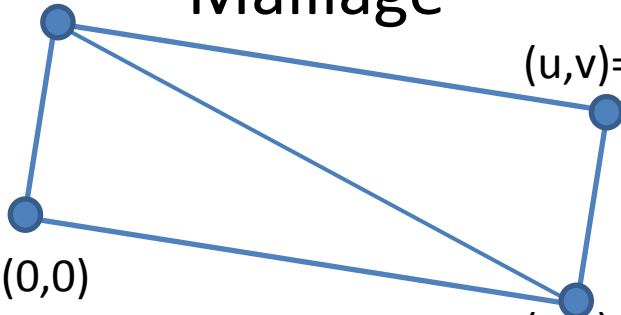
$(u,v)=(0,1)$

Maillage

$(u,v)=(1,1)$

$(u,v)=(0,0)$

$(u,v)=(1,0)$



Coordonnées paramétriques

*a.k.a « Coordonnées de texture »,
a.k.a « Coordonnées uv »*

$$(u, v) \in \mathbb{R}^2 \quad \text{par convention:} \quad (u, v) \in [0,1]^2$$

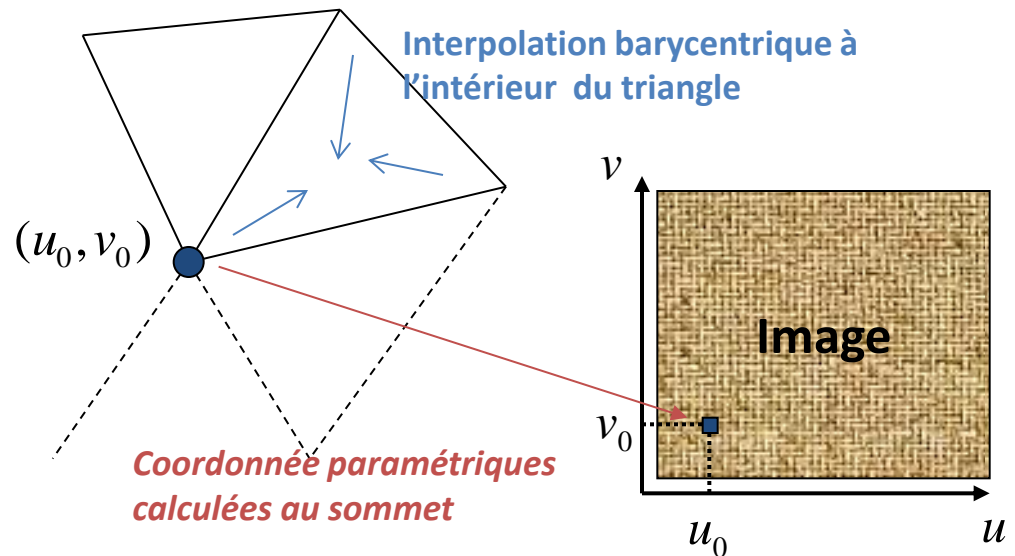
Définition d'une propriété de surface à partir d'une fonction bivariable:

$f :$

$$\mathbb{R}^2 \rightarrow \mathbb{R}^n$$

$$u, v \rightarrow c$$

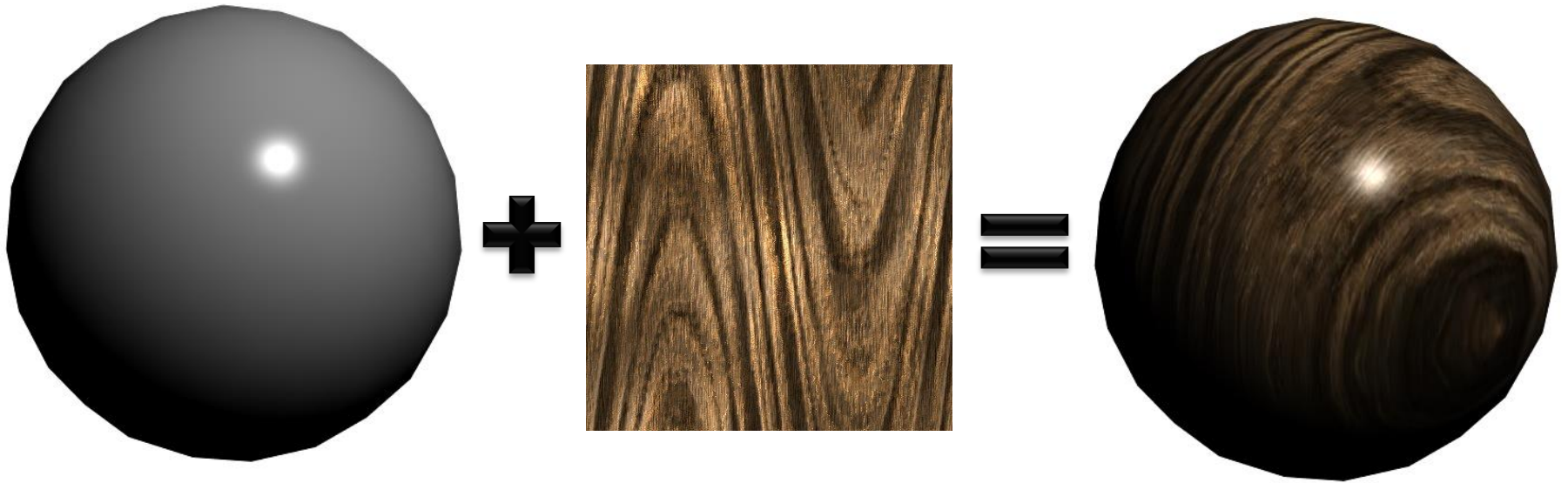
Exemple : valeur d'un pixel dans une image (« texture mapping »)



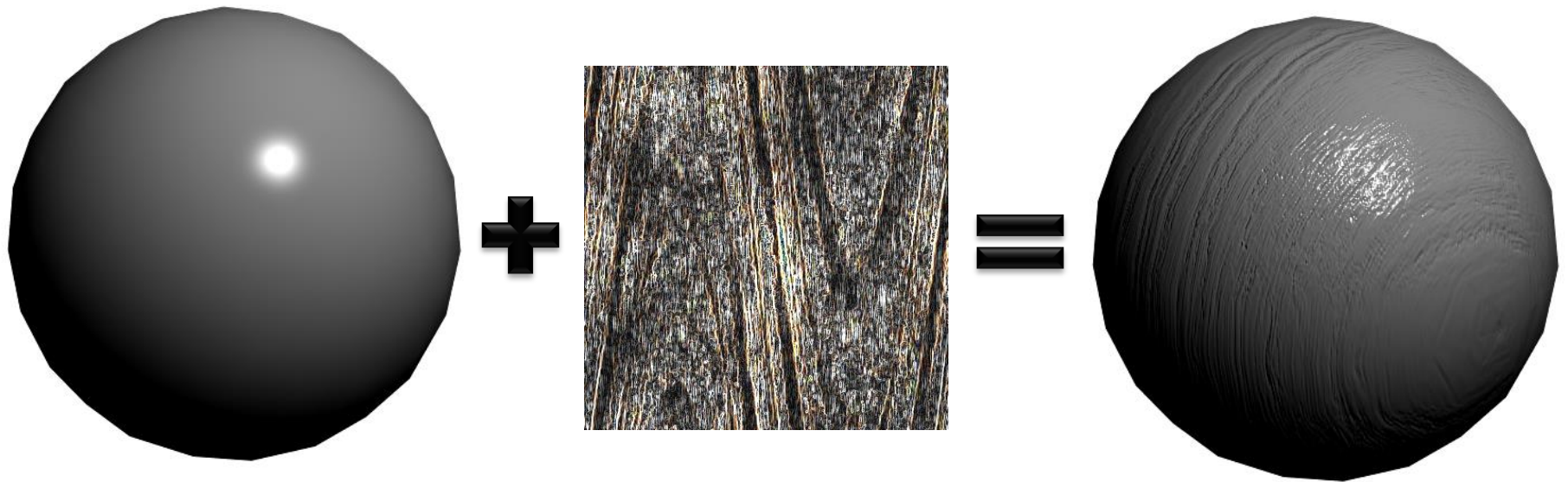
Définition de coordonnées paramétriques continues sur l'ensemble des sommets d'un maillage : **Paramétrisation**

Exemples d'algorithme de paramétrisation: LSCM, Floater

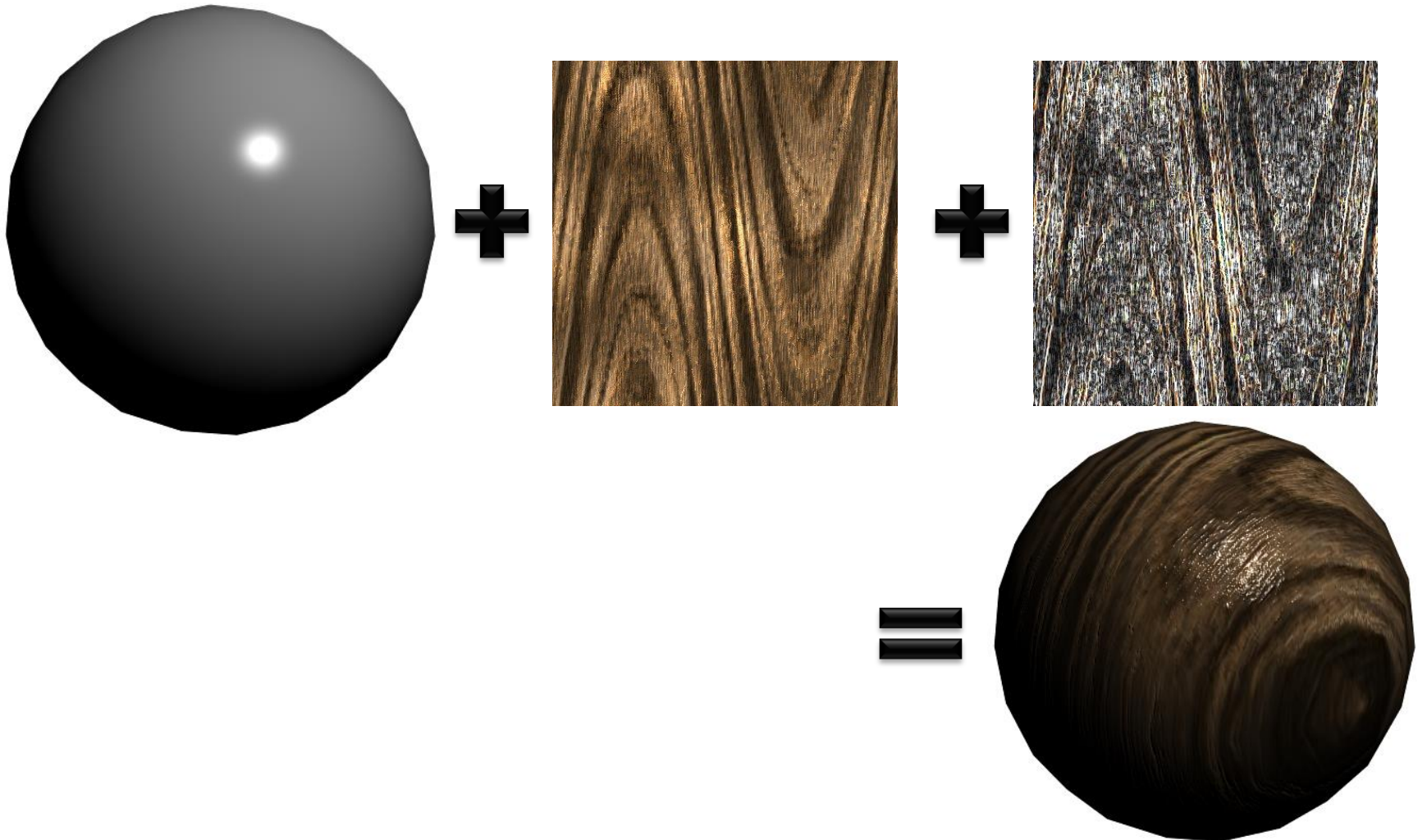
Texture couleur



Texture normale



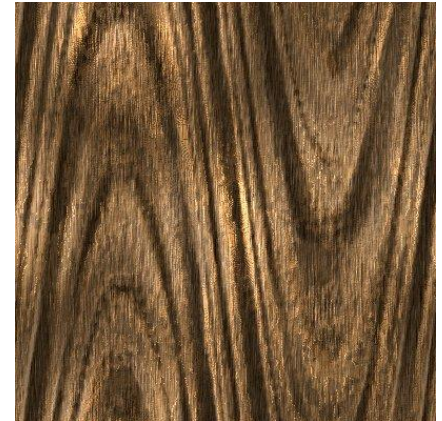
Couleur et normale



Type de textures

Texture 512x512 RGB

- 1D, 2D, 3D, etc
 - scalaire, vectorielle,
 - couleur, alpha, paramètre de BRDF
- 2 types principaux
 - Carte
 - Procédural



Carte de Texture

- « Image » prédéfinis, artificielle ou capturée (photo)
- Coût mémoire parfois élevé
 - Compression/décompression à la volée
- Nécessite un filtrage pour éviter les effets d'aliasing, de moiré, etc
- Evaluation efficace :
 - Accès mémoire + filtrages
- Pré-calculs possibles
 - Traitement d'image (rehaussement, débruitage, etc)
 - Calcul d'une pyramide (mip-map)
 - Quantification et compression

Texture Procédurale

- Une fonction évaluable en tout point d'un domaine
- Léger en mémoire : code la fonction + paramètres
- Couteux à l'évaluation
- A priori, une infinité de fonctions possible, mais quelques propriétés sont souhaitables
 - Déterministe
 - Variation naturelle (pseudo-aléatoire)
 - Faible nombre de paramètres intuitifs
 - Evaluations dé-corrélées les unes des autres
 - Calcul parallèle / GPU
- Exemple : **bruit de Perlin**, bruit en ondelette, bruit de Gabor

Bruit de Perlin

- Introduit par Ken Perlin en 1985
- Bruit de gradient
- Donne une apparence pseudo-aléatoire aux surfaces
- Multi-échelle : sommes de bruit de Perlin à plusieurs échelles

