Le lancer de rayons (*raytracing*)

Concepts fondamentaux

Christian NGUYEN

Département d'informatique
Université de Toulon

Plan

1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

Rasterization vs Raytracing

La rasterization consiste à produire une image à partir de la définition d'objets géométriques, de la projection des triangles qui les constituent et du calcul des couleurs ou des portions de texture associées en prenant en compte leur illumination.

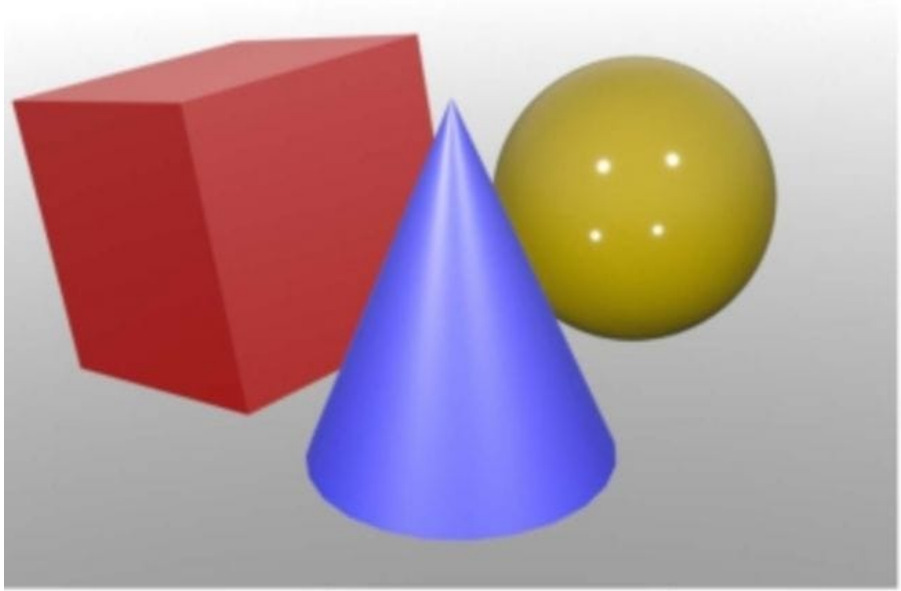
Le coût de cette approche est proportionnel au nombre de triangles (très peu de la définition de l'image finale). C'est une approche rapide et qui ne peut prétendre atteindre un rendu photoréaliste.

Le raytracing est son exact opposé. Il se base sur la définition de l'image pour calculer en chacun de ses points une couleur qui prend en compte de multiples phénomènes physiques.

Le coût de cette approche est proportionnel au nombre de pixels de l'image. Cette approche n'est pas adaptée à une visualisation « temps réel ».

Rasterization vs Raytracing

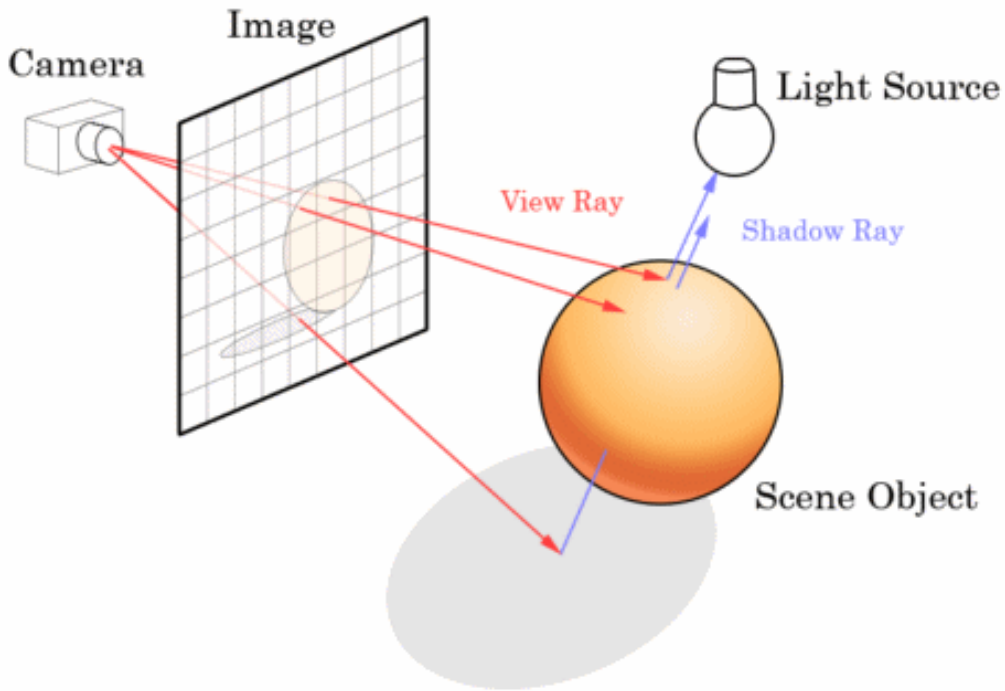
RASTERIZATION



RAY TRACING

Description d'une scène 3D transformée en une matrice 2D d'intensités.

Modélise l'interaction de la lumière et des objets en usant des principes simples de l'optique.



Elle est caractérisée par :

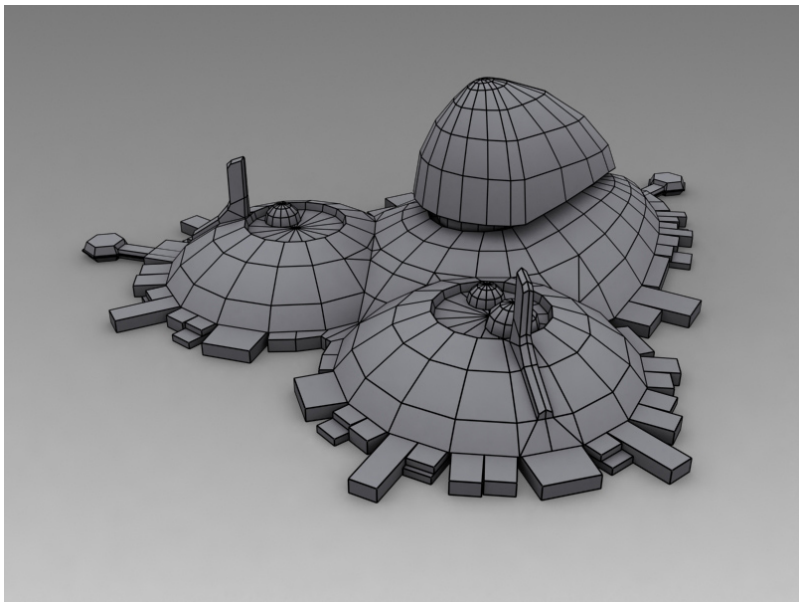
1. des objets,
2. une caméra,
3. des sources lumineuses,
4. des propriétés surfaciques et volumiques.

L'association d'un modèle projectif et d'un modèle d'illumination permet de représenter une scène 3D « réaliste » sur un support 2D.

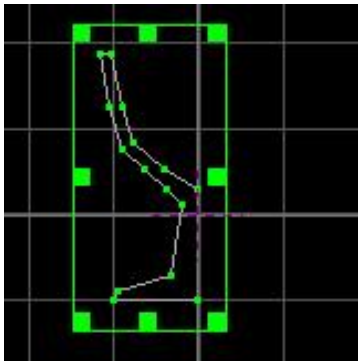
Tous les paramètres d'une scène 3D subissent diverses approximations.

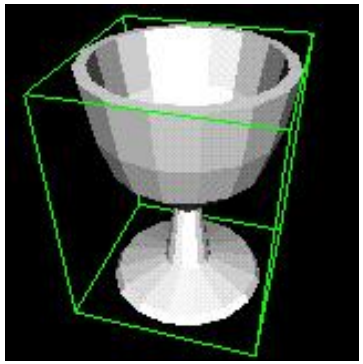
Décomposition en primitives :

- ↳ primitives : points, segments, polygones (triangles),
  - ↳ attributs sur les sommets (couleurs, normales, ...).



Manufacturés : « fil de fer », surfacique, volumique et fonctionnelle.

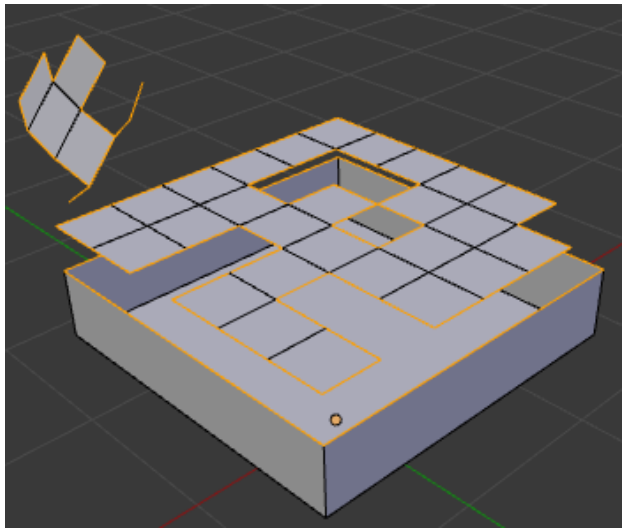




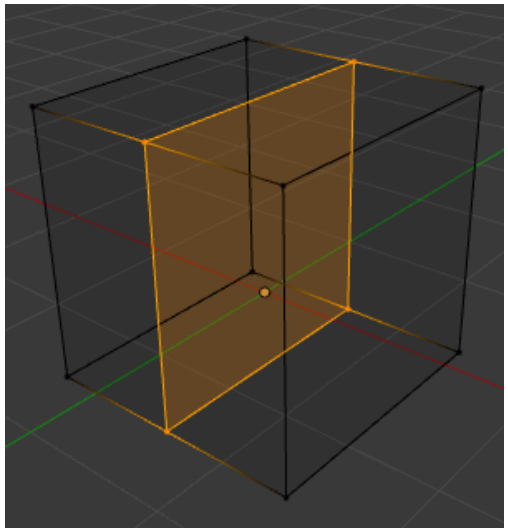
Naturels : système de particules et fractale.

Manifold vs *non manifold*

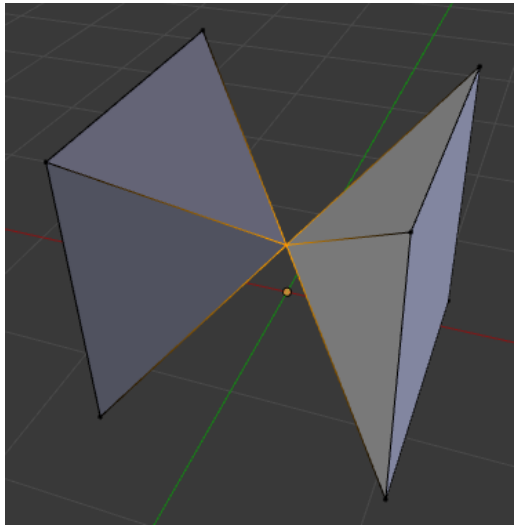
faces « pendantes »



faces internes

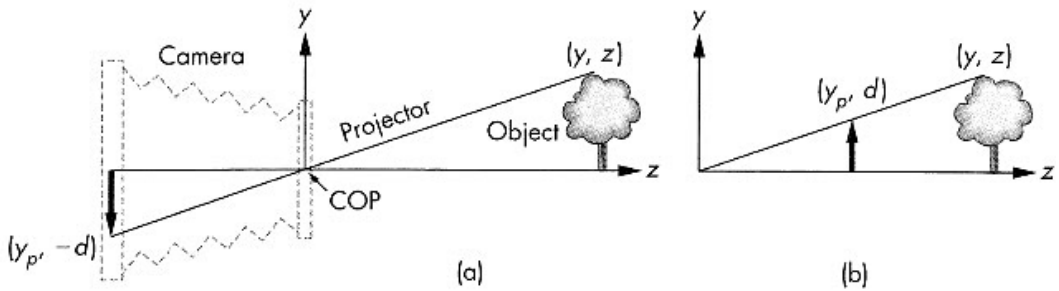


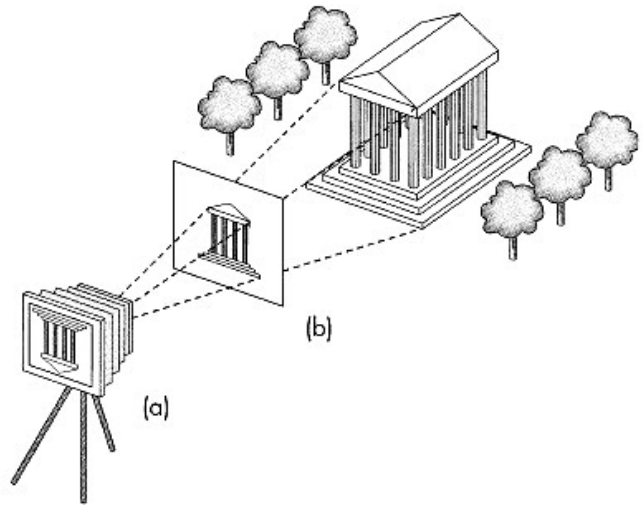
zones sans épaisseur



Modèle de la projection perspective.

Paramètres : position, orientation, ouverture, focale, plans de coupe.





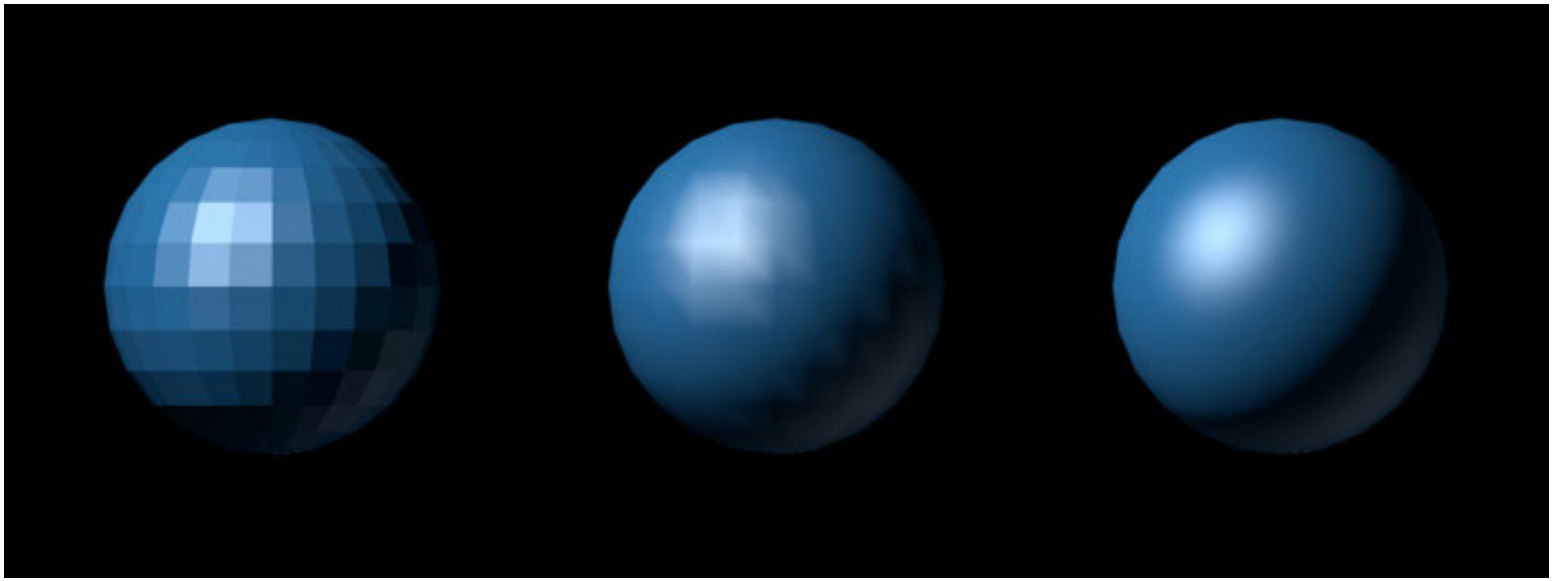
Paramètres : couleurs, intensité, localisation, taille et forme.

Simplifications :

- ponctuelle et d'une seule couleur.
- trajet en ligne droite, à travers des matériaux homogènes,
- interaction avec la surface des objets,

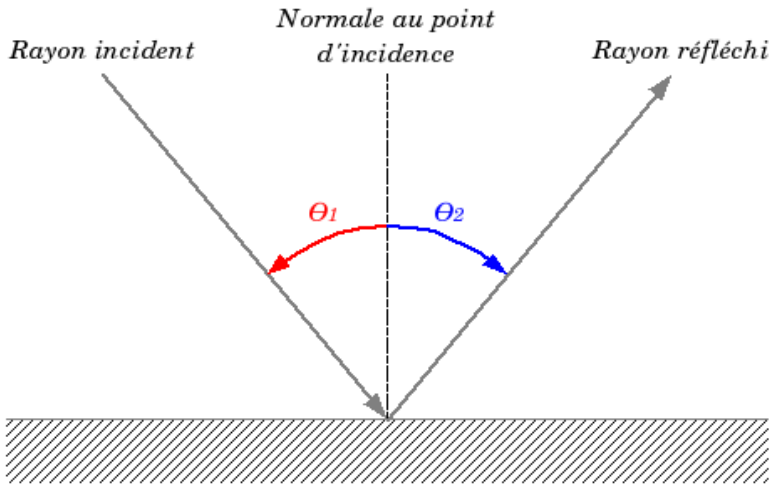
Propriétés non prises en compte : phase, longueur d'onde (diffraction, polarisation, dispersion et atténuation avec la distance).

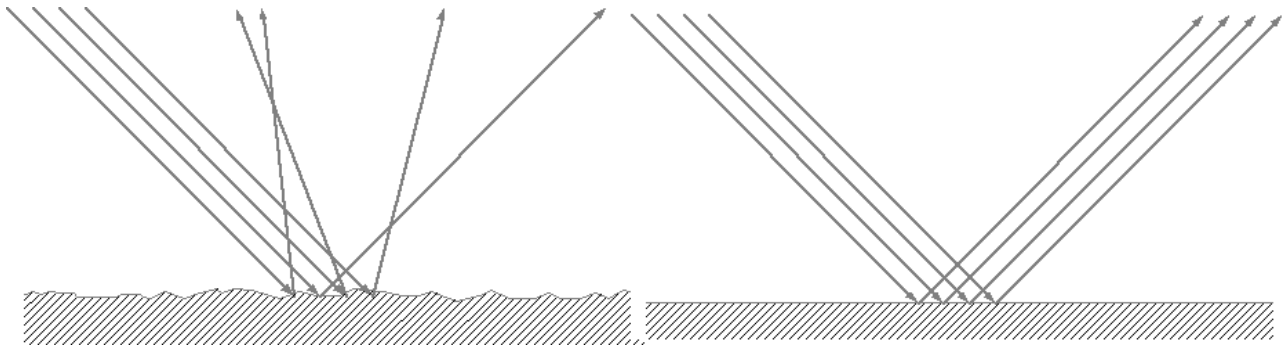
- source lumineuse : ponctuelle, à l'infini, pas d'atténuation,
- surface : interpolation des couleurs ou des normales aux faces ou aux sommets.

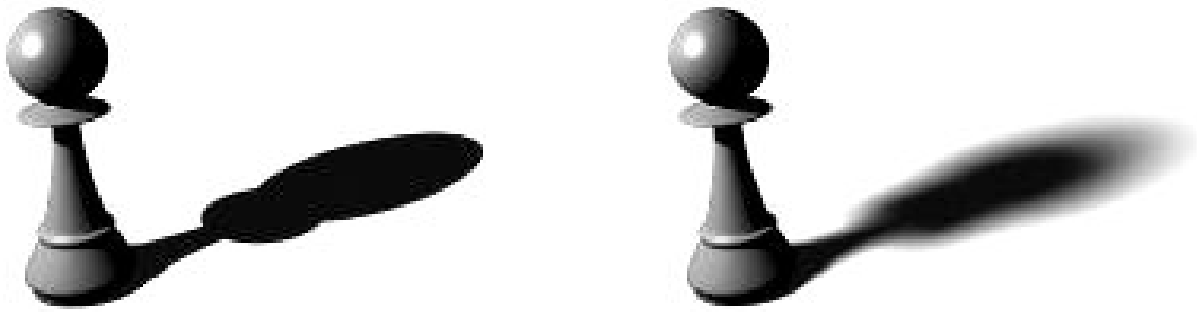


Flat, Gouraud, Phong

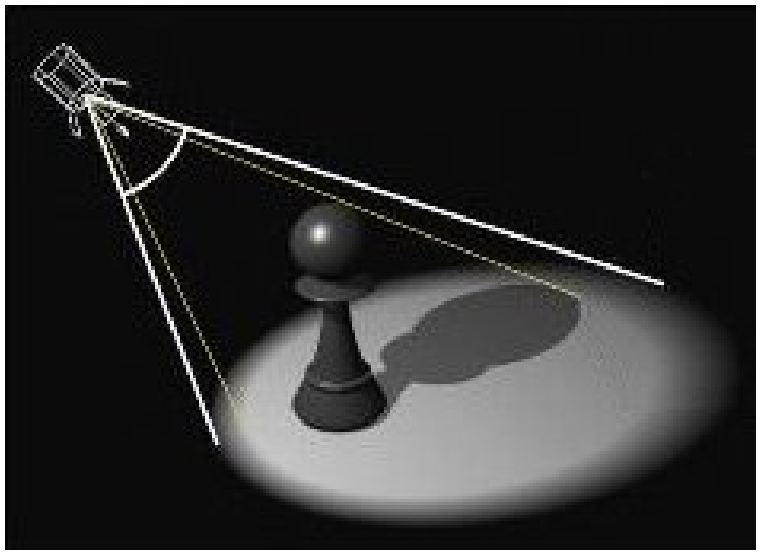
La réflexion obéit aux lois de Snell-Descartes.







Source ponctuelle vs aire lumineuse (pénombre)



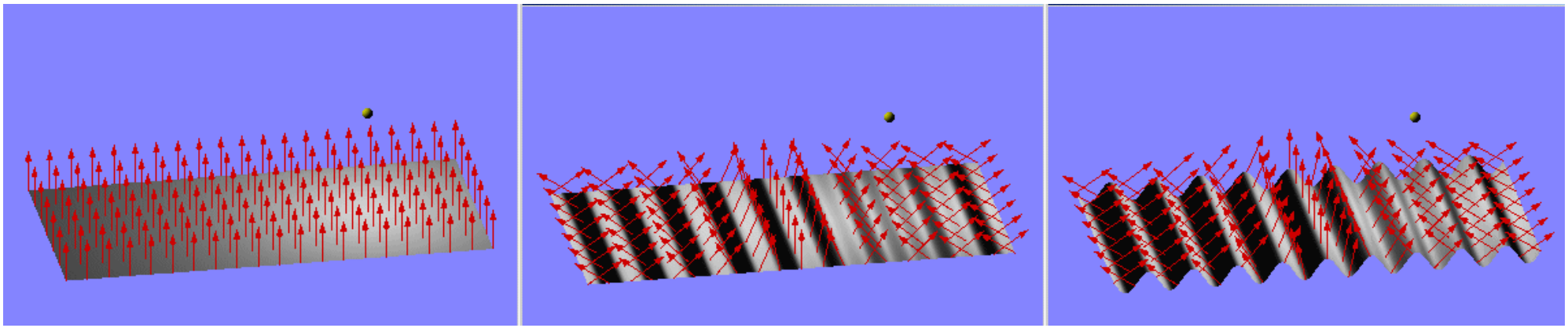
Spot lumineux

Couleur (*pattern*), transparence, reflexion, rugosité, brillance, etc.

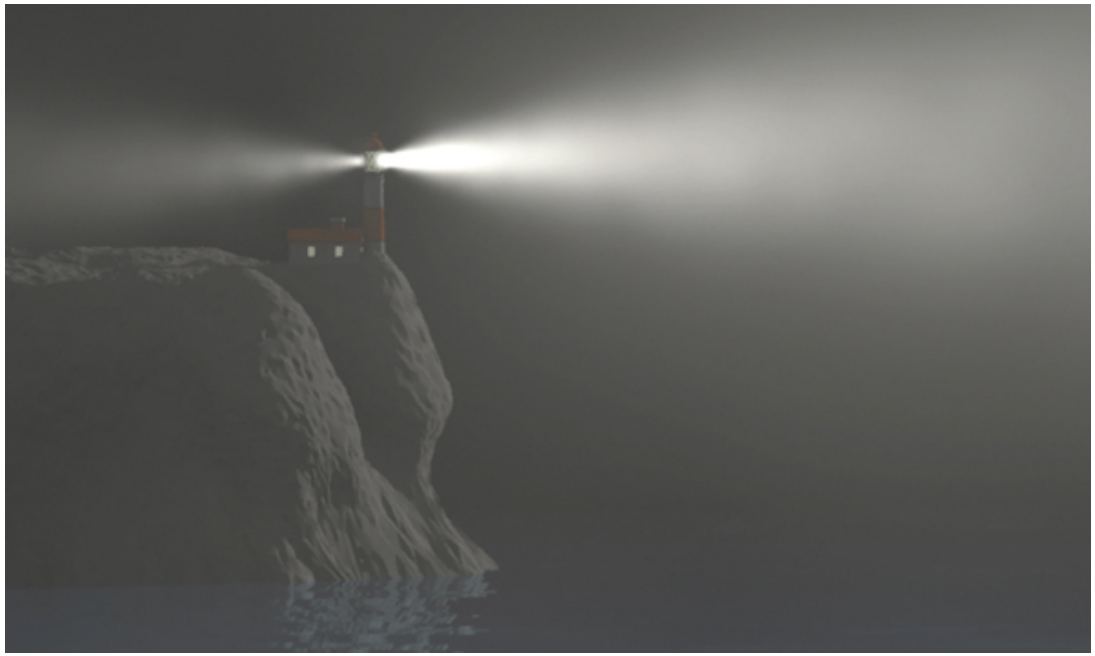
Texture mapping

Bump mapping

Perturbation des normales, par une fonction déterministe ou aléatoire.



Transparence



Brouillard

Plan

1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

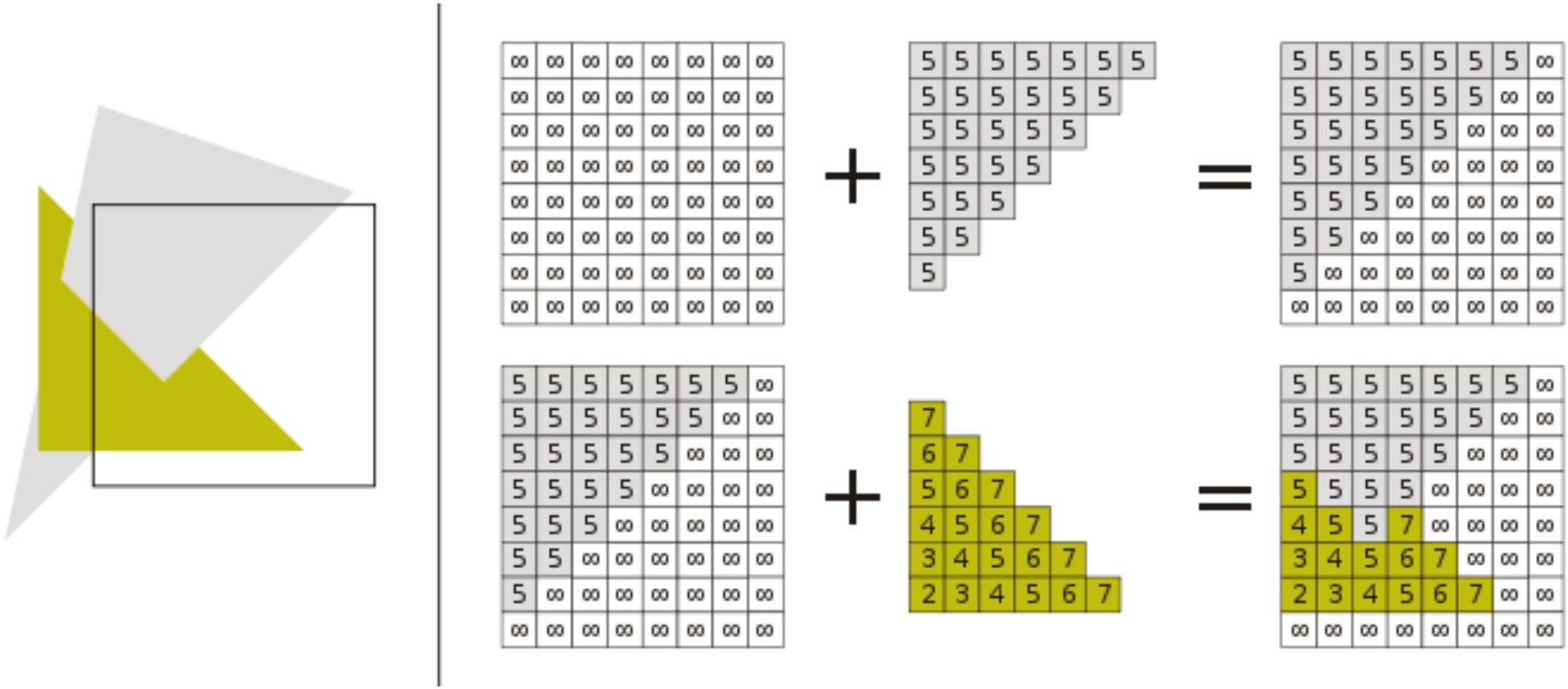
Principe général

Deux façons répandues de rendre une scène :

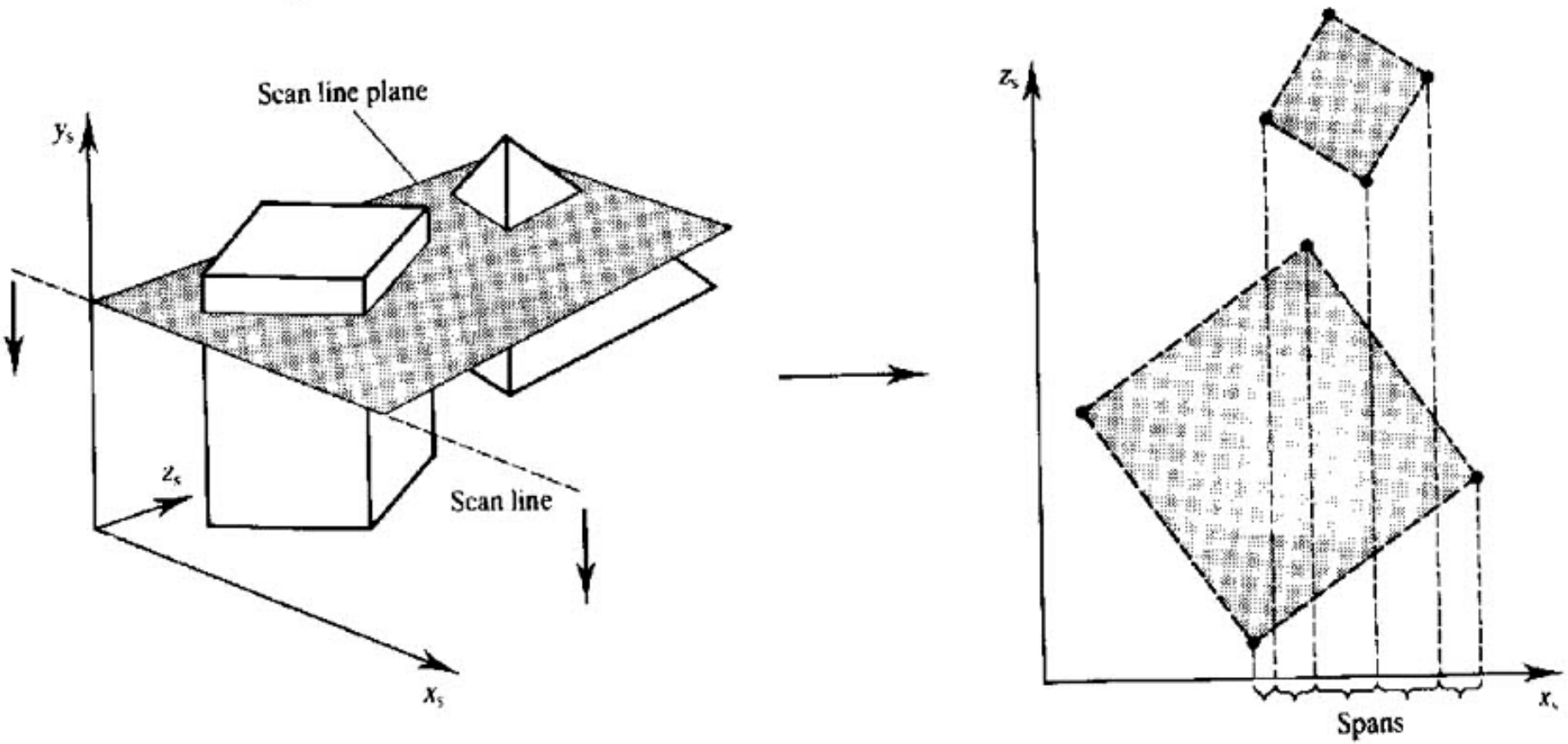
- polygone par polygone (*Z-buffer* [Catmull75]),
- dans un ordre défini par une ligne de balayage (*scan-line* [Newell72]).

Cela conditionne les algorithmes utilisés (surfaces cachées, ombrage, ...).

Z-buffer


∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
+
5 5 5 5 5 5 5
5 5 5 5 5 5
5 5 5 5 5
5 5 5 5
5 5 5
5 5
5
=
5 5 5 5 5 5 5 ∞
5 5 5 5 5 5 ∞ ∞
5 5 5 5 5 ∞ ∞ ∞
5 5 5 5 ∞ ∞ ∞ ∞
5 5 5 ∞ ∞ ∞ ∞ ∞
5 5 ∞ ∞ ∞ ∞ ∞ ∞
5 ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
5 5 5 5 5 5 5 ∞
5 5 5 5 5 5 ∞ ∞
5 5 5 5 5 ∞ ∞ ∞
5 5 5 5 ∞ ∞ ∞ ∞
5 5 5 ∞ ∞ ∞ ∞ ∞
5 5 ∞ ∞ ∞ ∞ ∞ ∞
5 ∞ ∞ ∞ ∞ ∞ ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞
+
7
6 7
5 6 7
4 5 6 7
3 4 5 6 7
2 3 4 5 6 7
=
5 5 5 5 5 5 5 ∞
5 5 5 5 5 5 ∞ ∞
5 5 5 5 5 ∞ ∞ ∞
5 5 5 5 ∞ ∞ ∞ ∞
4 5 5 7 ∞ ∞ ∞ ∞
3 4 5 6 7 ∞ ∞ ∞
2 3 4 5 6 7 ∞ ∞
∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞

Scan-line



La méthode du rendu

Deux qualités :

- modélisation d'effets visuels réalistes,
- simplicité conceptuelle et algorithmique de l'approche.

Historiquement :

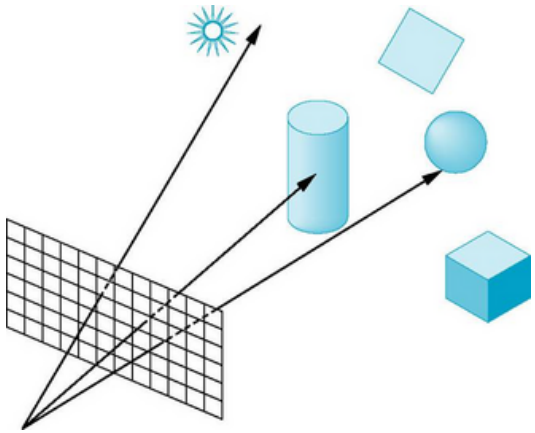
1. forward ray tracing : on lance des rayons lumineux vers la scène et on traite ceux qui traversent le plan de vue,
2. ray casting : on lance des rayons de vue depuis le plan de projection sans prendre en compte les réflexions,
3. recursive ray tracing.

Le “ray casting” ou illumination directe

[Appel 68] [Goldstein et Nagel 71] : on lance un rayon de vue depuis l'observateur à travers un pixel dans l'espace de modélisation.

On teste l'intersection de ce rayon avec tous les objets, on obtient l'objet le plus proche.

L'intersection la plus proche entre le rayon et les objets détermine la surface visible en ce point.



Algorithme

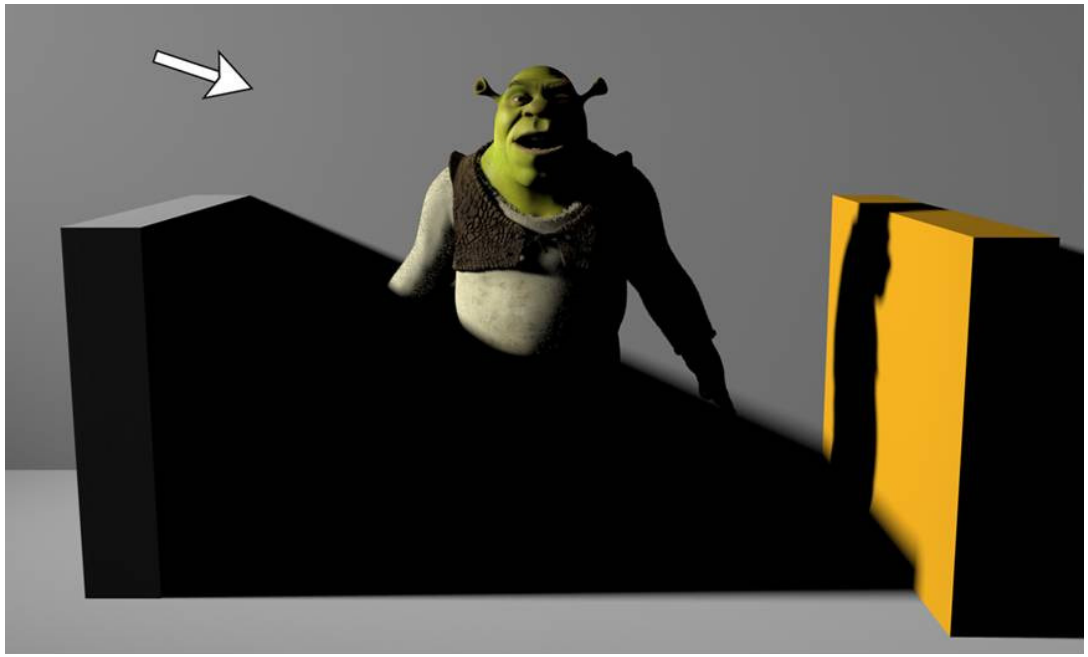
Algorithme Raycasting(*image*)

```
pour chaque ligne y de l'image faire
    pour chaque colonne x de l'image faire
        i ⟵ MAX_DIST
        générer un rayon de vue à travers le pixel(x, y)
        pour chaque objet de la scene faire
            calculer l'intersection rayon / objet
            si (intersection et plus courte distance à l'œil) alors
                sauver i
            fsi
        fpour
        si (i ≠ MAX_DIST) alors
            calculer la normale à la surface au point i
            pour chaque source lumineuse l faire
                générer un rayon de i vers l
                pour chaque objet de la scene (sauf celui de i) faire
                    calculer l'intersection rayon / objet
                    si (pas d'obstacle) alors
                        calculer la couleur et l'intensite de i en fonction de l
                    fsi
                fpour
            fpour
        sinon      ▷ aucune intersection entre le rayon de vue et la scène
            couleur et intensite du fond
        fsi
        sauver le pixel(x, y, couleur)
    fpour
fpour
```

Illumination directe

Contribution relative de chaque source lumineuse : rayons d'ombre depuis le point d'intersection d'une surface réfléchissante, ou transparente, vers chaque source lumineuse.

Tests d'intersection avec les autres objets sur le trajet vers les lumières afin de déterminer leur contribution (sauf objets transparents).



Du *ray casting* au *ray tracing*

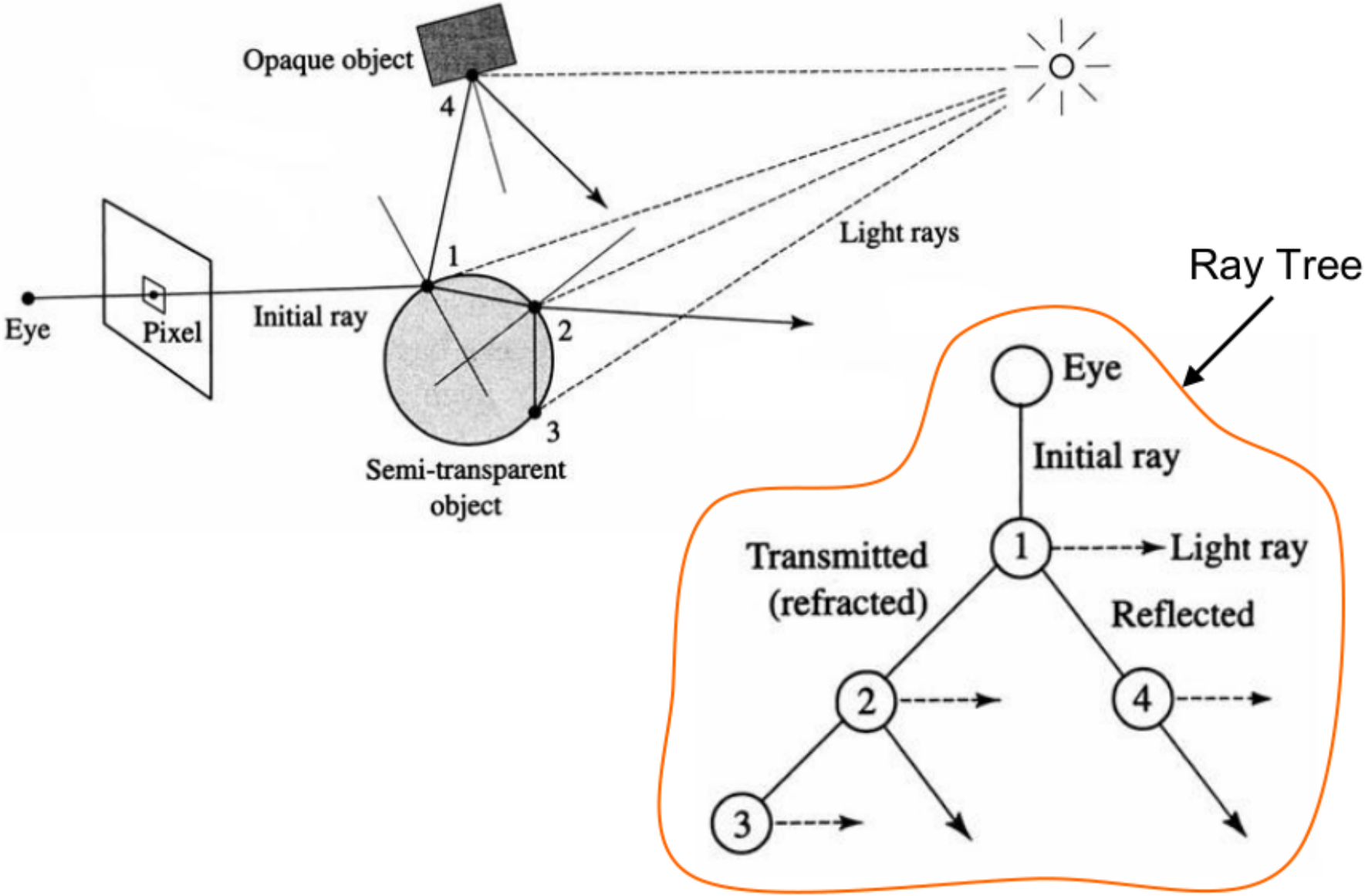
La détermination de la couleur finale au point d'intersection devient un processus fondamentalement récursif, fonction de :

- la couleur de l'objet,
- la couleur et la distance de la source lumineuse,
- la participation des rayons réfléchis et transmis (réfractés) [Whitted 80].

Les couleurs contribuant au calcul de la couleur finale sont aux feuilles de l'arbre :

- elles sont évaluées et transmises vers la racine,
- à chaque nœud de l'arbre une couleur intermédiaire est calculée.

Ray tracing



Trajet des rayons lumineux et arbre correspondant

Lancer de rayons récursif

Algorithme Raytracing(*orig*, *dir*, *prof*, *coul*)
données
    orig, *dir* : vecteur
    prof : entier
    coul : couleur (référence)
variables
    pt_inter, dir_reflexion, dir_transmission : vecteur
    coul_loc, coul_ref, coul_trans : couleur

---

si (*prof* $>$ *PROF_MAX*) **alors**
    coul $\longleftarrow$ noir
sinon
    trouver le point d'intersection le plus proche (s'il existe)
    si (pas d'intersection) **alors**
        coul $\longleftarrow$ COUL_FOND
    sinon
        coul_loc $\longleftarrow$ contribution de la coul. locale au pt d'intersection
        dir_reflexion $\longleftarrow$ calcul de la direction du rayon reflechi
        Raytracing(pt_inter, dir_reflexion, prof+1, coul_ref)
        dir_transmission $\longleftarrow$ calcul de la direction du rayon transmis
        Raytracing(pt_inter, dir_transmission, prof+1, coul_trans)
        coul $\longleftarrow$ Combiner(coul_loc, k_a, coul_ref, k_r, coul_trans, k_t)
    fsi
fsi

Plan

1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

Le modèle d'illumination

Le cas le plus simple est la réflexion spéculaire décrite par les lois de Fresnel mais qui ne s'adresse qu'à des surfaces parfaites.

Modèle théorique complexe de l'interaction entre lumière et solide : réflexion (diffuse, spéculaire et interne), dispersion, émission, absorption et transmission.

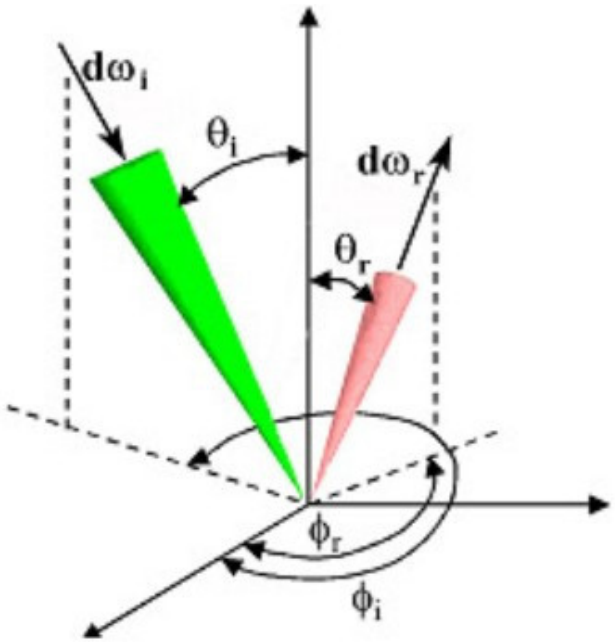
Simplification en se limitant à la modélisation des réflexions et des transmissions qui dépendent :

- de la longueur d'onde incidente (λ),
- de l'angle d'incidence,
- de la nature de la surface ainsi que de ses propriétés électriques (permittivité, perméabilité et conductivité)).

L'ensemble de ces propriétés sont les paramètres de la fonction de distribution probabiliste qui caractérise la réflectivité bidirectionnelle (BRDF) :

$$R_{bd}(\lambda, \Phi_i, \Theta_i, \Phi_r, \Theta_r)$$

Relations entre lumière incidente de direction (Φ_i, Θ_i) et lumière réfléchie dans la direction (Φ_r, Θ_r), avec Φ azimut [1] et Θ zénith [2]. A noter que cette fonction dépend aussi de la longueur d'onde λ.

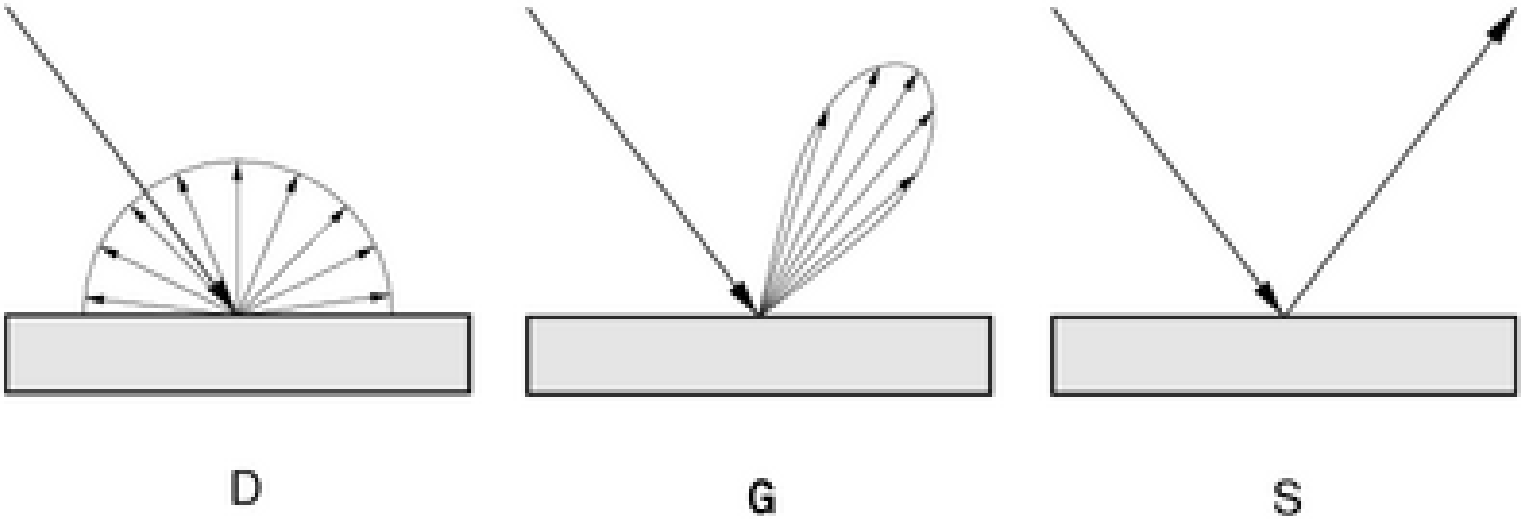


Relation entre l'énergie incidente et l'intensité transmise

1. de l'arabe as-simt, qui signifie « direction ».
2. du latin zenit, issu de l'écrit incorrect de l'arabe samt al-ras

Subdivision de la BRDF en trois composantes [He *et al.* 91] :

- contribution diffuse idéale (D, réflexions multiples dûes aux interactions sur et sous la surface),
- contribution diffuse directionnelle (G, réflexions du 1er ordre),
- contribution spéculaire (S).



Subdivision de la BRDF

Rayons réfléchis

Leur coût est proportionnel au niveau de réflexion.

Par la suite, on ne distinguera que deux types de réflexion : la réflexion diffuse idéale et la réflexion spéculaire.

On suppose que le matériau est suffisamment opaque pour que l'on puisse ignorer :

- une éventuelle partie transmise à travers le matériau,
- un décalage spatial notable entre le point d'arrivée du photon incident et celui du photon réémis.

↳ retransmission immédiate dans une direction donnée (vs énergie absorbée, dispersée puis réemise dans toutes les directions).

Réflexion diffuse

Une infinité de rayons de réflexion répartis uniformément dans toutes les directions.

Composante diffuse I_d simple à calculer : produit scalaire des vecteurs unitaires $\vec{N}$ par $\vec{L}$ (loi de Lambert).

$$I_d = I_o I_i k_d \cos\alpha,\ 0 \leqslant \alpha \leqslant 2\pi$$

k_d coefficient de réflexion empirique qui dépend d'une longueur d'onde. Remarque : I_d est constant pour une surface plane.

Dans le cas de plusieurs sources lumineuses L_n, on a :

$$I_d = k_d \sum_n I_{i,n}(L_n \cdot N)$$

Réflexion spéculaire

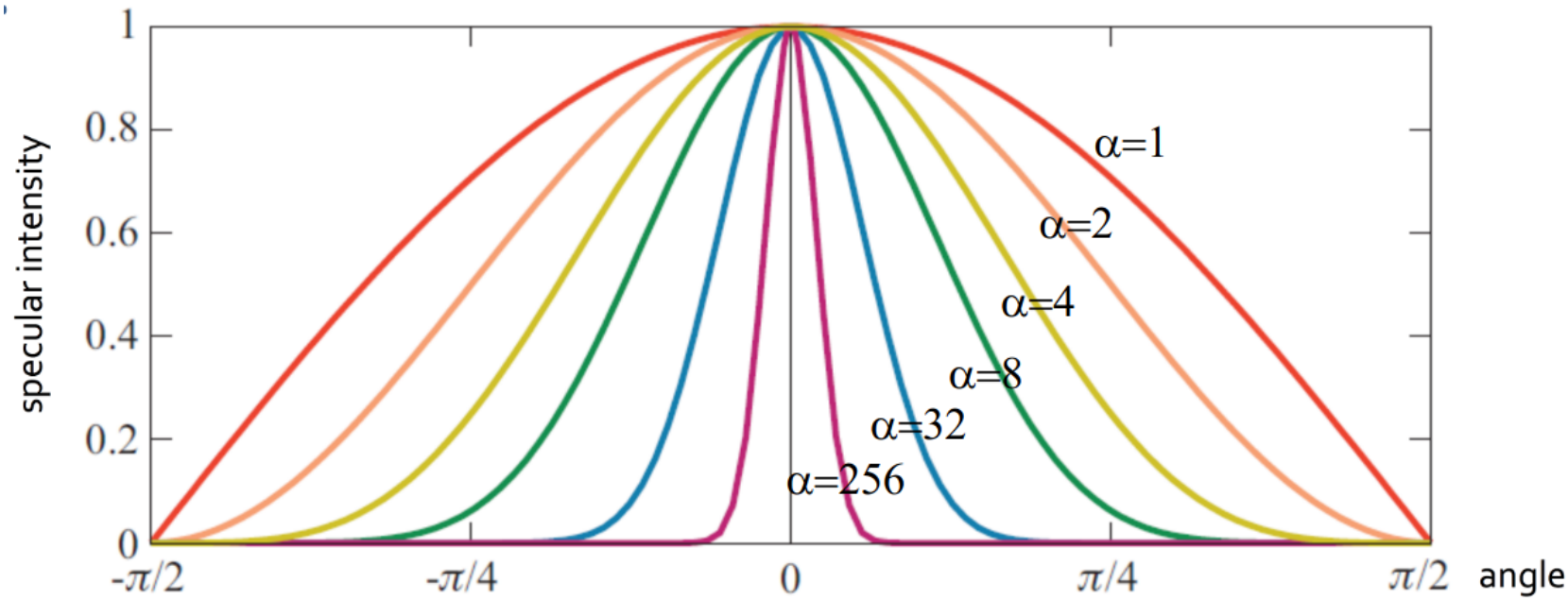
Surface spéculaire idéale : angle d'incidence = angle de réflexion, lumière réfléchie dans une seule direction.

Première approximation raisonnable mais empirique de ce phénomène [Phong 75] :

$$\begin{aligned} I_s &= W(\alpha)\cos^n(\gamma) \\ &= I_i k_s (R \cdot V)^n \end{aligned}$$

- α angle d'incidence,
- γ angle entre $\vec{R}$ (réflexion) et $\vec{V}$ (vue),
- $W(\alpha)$ ratio entre lumière incidente et lumière réfléchie, remplacé par $I_i k_s$,
- n propriétés physiques de la surface (miroir parfait : $n = +\infty$).

Réflexion spéculaire



Propriétés spéculaires des surfaces ($\cos^n(\alpha)$)

[Blinn 77] modèle physiquement moins empirique qui prend mieux en compte les rayons fortement incidents mais occasionne tant de calculs supplémentaires que les concepteurs continuent d'utiliser le modèle de Phong.

Lumière ambiante

Troisième composante utilisée dans les modèles d'illumination : modélise les multiples réflexions de la lumière.

Coûteuse à calculer : simple approximation par une constante, associée à un coefficient de réflexion ambiant qui est une propriété du matériau.

Permet de prendre en compte l'illumination d'une surface dans l'ombre ou ne faisant face à aucune source lumineuse.

Modèle d'illumination de Phong

Ces trois composantes ont été pris en compte dans un modèle d'éclairement qui détermine l'énergie réémise par un point d'une facette suivant la formule :

$$I = I_o I_a k_a + I_i (I_o k_d (L \cdot N) + k_s (R \cdot V)^n)$$

- I_a contribution ambiante,
- I_i intensité de la lumière incidente,
- k_d et k_s contributions diffuse et spéculaire, toutes deux dépendant du matériau de la surface,
- n coefficient de surbrillance.

Application au modèle RVB

- couleur des surfaces en jouant sur les coefficients de réflexion diffuse,
- modélisation des reflets comme des reflets blancs (pour une lumière blanche).

$$
\begin{aligned}
I_r &= I_o I_a k_{ar} + I_i (I_o k_{dr} (L \cdot N) + k_s (R \cdot V)^n) \\
I_v &= I_o I_a k_{av} + I_i (I_o k_{dv} (L \cdot N) + k_s (R \cdot V)^n) \\
I_b &= I_o I_a k_{ab} + I_i (I_o k_{db} (L \cdot N) + k_s (R \cdot V)^n)
\end{aligned}
$$

Remarque : le terme spéculaire est commun aux trois équations.

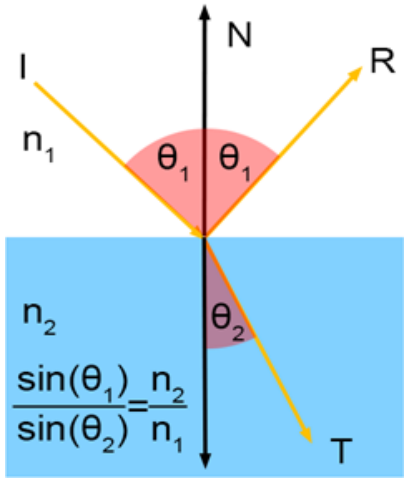
La synthèse des couleurs par trois composantes reste une approximation.

Les rayons transmis

Ne prend en compte que l'angle d'incidence et la différence de densité entre les deux milieux au niveau de la surface de l'objet (indice de refraction) :

$$n_1 \sin \Theta_i = n_2 \sin \Theta_t$$

avec n_1, n_2 les indices des milieux de propagation séparés par la surface semi-transparente (ex. dioptre air-eau, $\Theta_t < \Theta_i$, $n_1 = 1$, $n_2 = 1,3$).

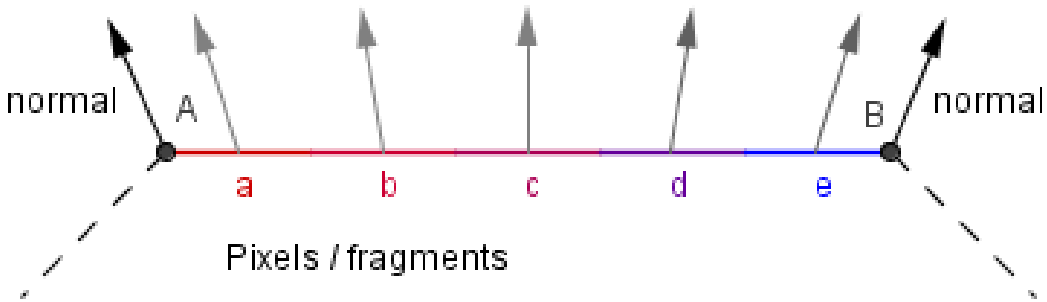


Ombrage d'une surface

À cause de l'approximation des surfaces par un maillage polygonale, le modèle manque de réalisme (polygone d'une couleur unie).

[Gouraud 71] conservation, au niveau des sommets, des normales réelles de la surface approximée : couleur réelle calculée puis interpolée linéairement sur tout le polygone (mais brutales variations d'intensité à la frontière des polygones).

Phong interpole linéairement la normale à la surface et donc les calculs d'intensité en chaque pixel.

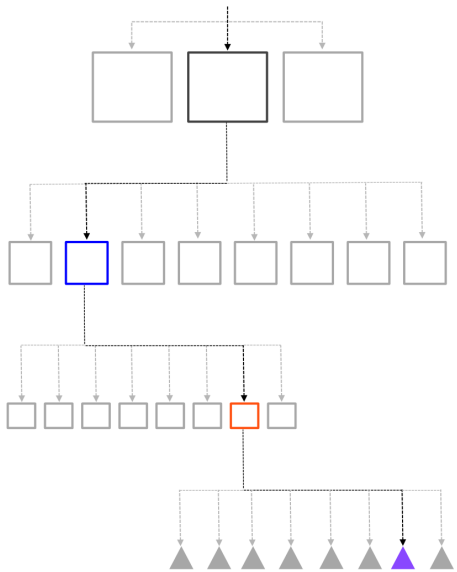


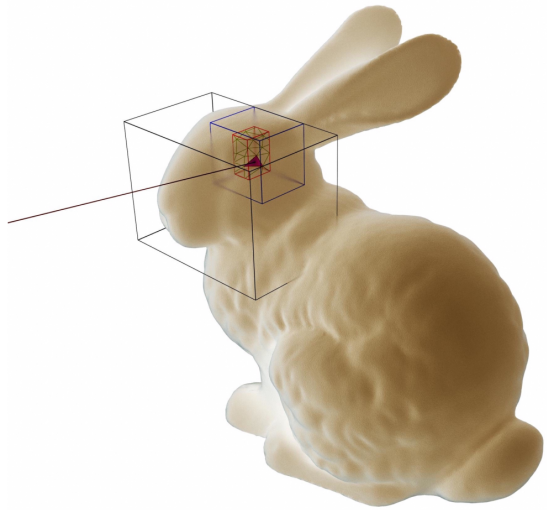
Sur le plan calculatoire

Pour 1920×1080 pixels on a :

- environ 2 M rayons de vue,
- 4 à 6 fois plus de rayons d'ombre,
- 10 fois plus de rayons réfléchis et transmis.

[Whitted 80] jusqu'à 95% du temps de calcul utilisé dans les calculs d'intersection pour une scène d'une complexité moyenne.
↳ utilisation de la notion de volume englobant [Clark 76].





Plan

1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

Intersection segment de droite / plan

Soit un segment de droite défini par $P_1(x_1, y_1, z_1)$ et $P_2(x_2, y_2, z_2)$, l'équation paramétrique du segment de droite est :

$$\begin{aligned} x &= x_1 + (x_2 - x_1)t = x_1 + it \\ y &= y_1 + (y_2 - y_1)t = y_1 + jt \\ z &= z_1 + (z_2 - z_1)t = z_1 + kt \end{aligned}$$

D'autre part, l'équation implicite d'un plan est :

$$Ax + By + Cz + D = 0$$

Un vecteur normal à ce plan est (A, B, C) (que l'on peut normer).

Un plan peut donc être défini par sa normale unitaire et un point.

Intersection segment de droite / plan

On cherche la valeur de t pour laquelle la droite coupe le plan. On substitue donc les coordonnées (paramétrées) de la droite dans l'équation (implicite) du plan :

$$t = -\frac{A.x_1 + B.y_1 + C.z_1 + D}{A.i + B.j + C.k}$$

Ce qui revient à chercher le cosinus de l'angle entre la normale du plan et le vecteur directeur de la droite :

$$V_1.V_2 = \|V_1\|\|V_2\| \cos(\alpha)$$

Intersection segment de droite / sphère

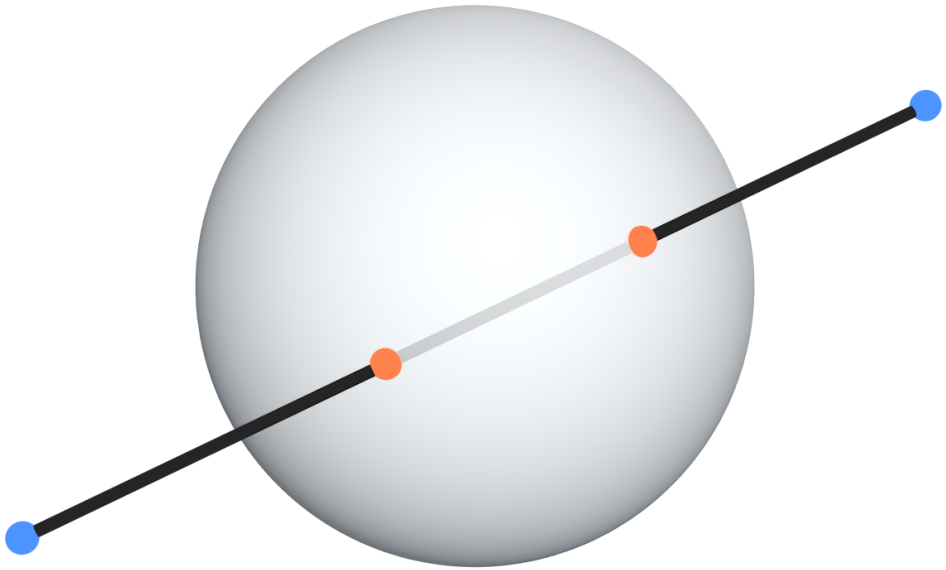
Soit un segment de droite défini par :

$$I = A + tD$$

A un point de la droite et D son vecteur directeur unitaire.

D'autre part, l'équation d'une sphère de centre (l, m, n) et de rayon r est :

$$(x - l)^2 + (y - m)^2 + (z - n)^2 = r^2$$



Intersection segment de droite / sphère

En substituant x, y et z dans l'équation implicite de la sphère on obtient une équation quadratique en t de la forme :

$$at^2 + bt + c = 0$$

- $\Delta < 0$: le segment de droite n'intersecte pas la sphère
- $\Delta = 0$: le segment est tangent à la sphère.
- $\Delta > 0$: les racines réels donnent les points d'intersection avant et arrière.

Normale N à la surface sphérique au point d'intersection de coordonnées (x_i, y_i, z_i) :

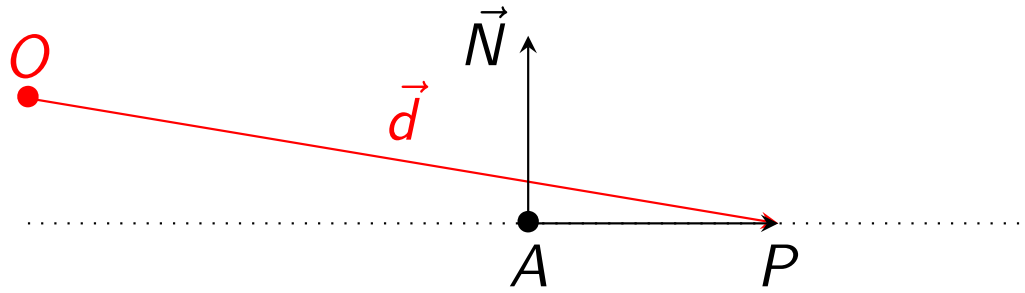
$$N = (\frac{x_i - l}{r}, \frac{y_i - m}{r}, \frac{z_i - n}{r})$$

Intersection segment de droite / triangle

Une approche : calculer l'intersection rayon / plan puis vérifier si le point d'intersection est dans le triangle.

Soit un triangle définit par ses 3 sommets A, B et C. Sa normale N est définie par $N = AB \wedge AC$.

On vérifie que le vecteur $\overrightarrow{AP}$ est perpendiculaire à $\vec{N}$:



$$\overrightarrow{AP}.\vec{N} = P.\vec{N} - A.\vec{N} = 0$$

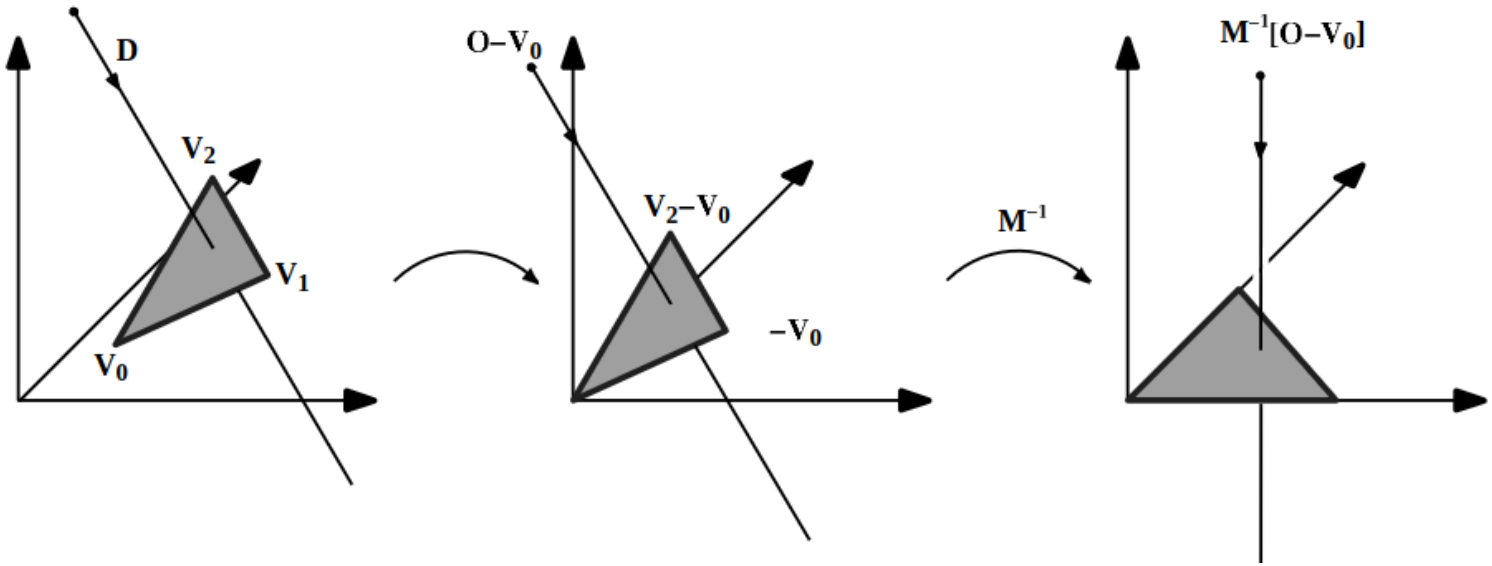
Soit la demi-droite (ou rayon) d'équation $P = O + t\ \vec{d}$, on a :

$$(O + t\ \vec{d}).\vec{N} = A.\vec{N} \Leftrightarrow t = \frac{A.\vec{N} - O.\vec{N}}{\vec{d}.\vec{N}} = \frac{\overrightarrow{OA}.\vec{N}}{\vec{d}.\vec{N}}$$

Intersection segment de droite / triangle

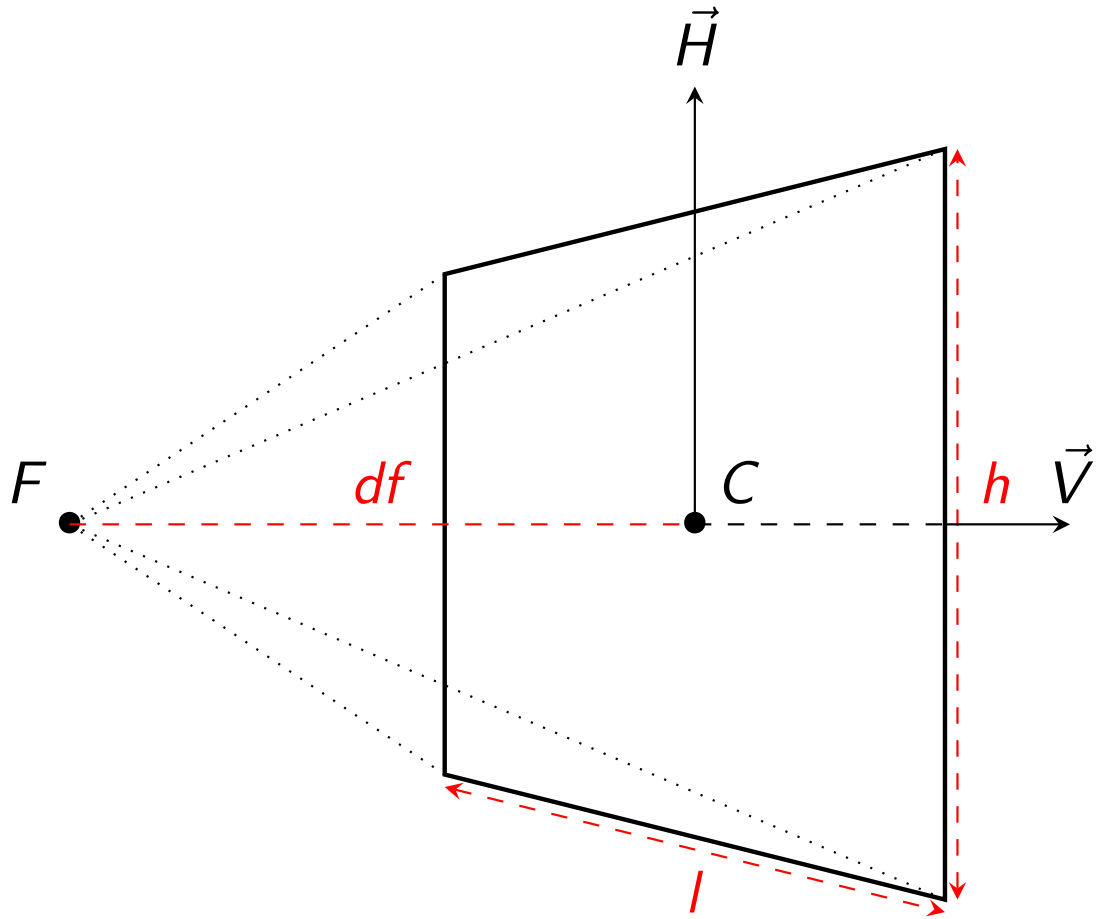
Il existe une méthode de calcul rapide de l'intersection, sans nécessiter le pré-calcul de l'équation du plan contenant le triangle [Möller-Trumbore 97].

L'algorithme translate l'origine du rayon puis change la base pour donner un vecteur (t, u, v), où t est la distance au plan dans lequel se trouve le triangle et (u, v) représente les coordonnées à l'intérieur du triangle.



Le modèle de caméra et l'écran

L'écran constitue la base de la pyramide régulière à base rectangulaire constituant la caméra.



Le modèle de caméra et l'écran

Les deux principaux paramètres définissant une caméra sont sa position (qui donne le centre C de l'écran) et la direction vers laquelle elle pointe (qui donne le vecteur normal V au plan de projection).

Pour finir de régler l'orientation de la caméra (et donc de l'écran), il faut préciser sa verticale : un vecteur normalisé H, perpendiculaire au vecteur de direction de vue et dans le plan de projection.

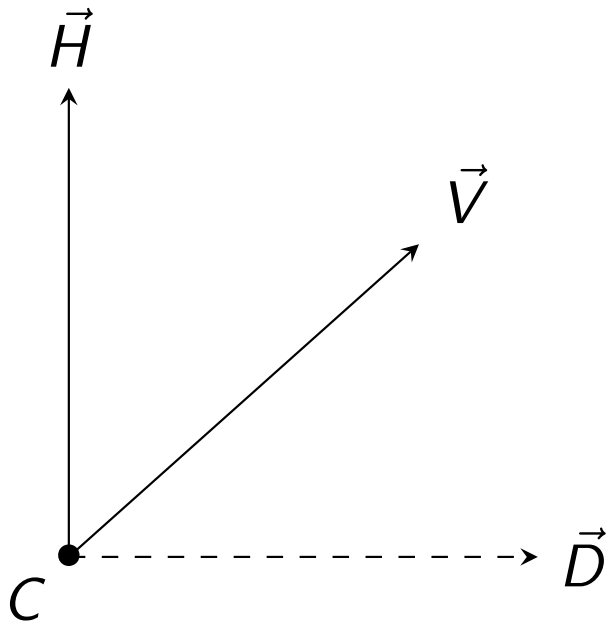
Enfin, il faut donner la taille de l'écran : sa largeur l et sa hauteur h.

Déplacement dans l'écran

Déplacement verticale : $P = C + tH,\ t \in [-h/2, h/2]$

Pour déterminer le déplacement horizontal, on définit un troisième vecteur D, perpendiculaire à H et V, dans le même plan que H :

$$D = V \wedge H$$

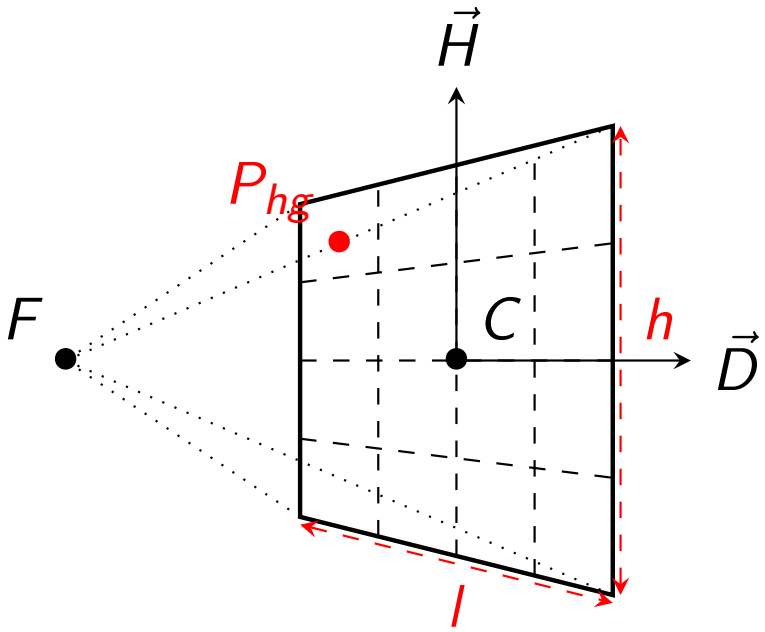


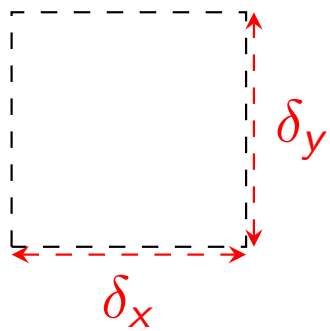
Déplacement dans l'écran

Il reste à discrétiser notre écran en « carreaux » et à calculer le centre de chacun de ces rectangles.

Pour cela, soient $\delta_x = l/n_c$ et $\delta_y = h/n_l$, avec n_c et n_l nombre de colonnes et de lignes, alors :

$$P_{hg} = C + H(h/2 - \delta_y/2) - D(l/2 - \delta_x/2)$$
$$P_{xy} = P_{hg} - H(y.\delta_y) + D(x.\delta_x)$$

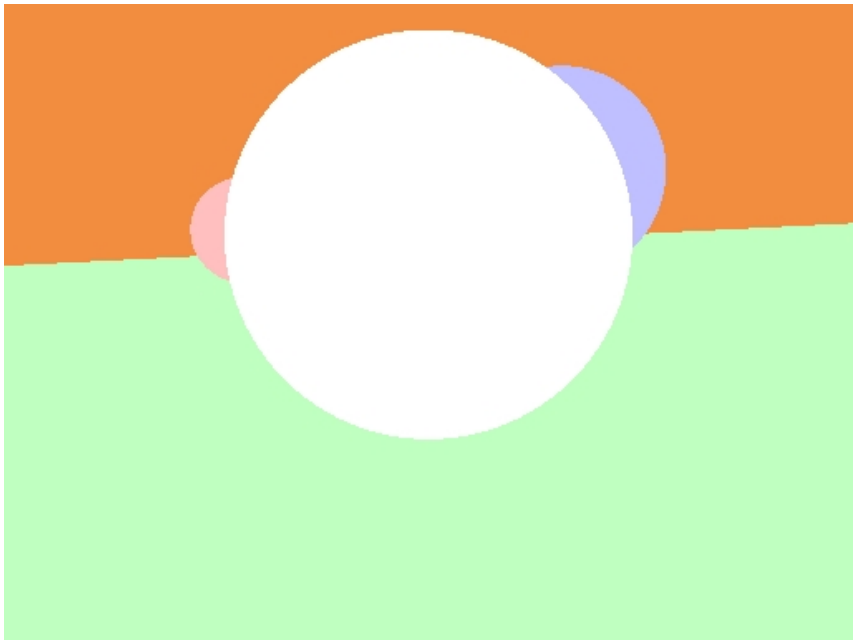




Le rayon et image obtenue

Le vecteur directeur du rayon qui part de P_{xy} vaut $P_{xy} - F$, avec $F = C - V.d_f$ position du foyer (constante).

En évitant tout calcul d'éclairage, on obtient :



Que la lumière soit

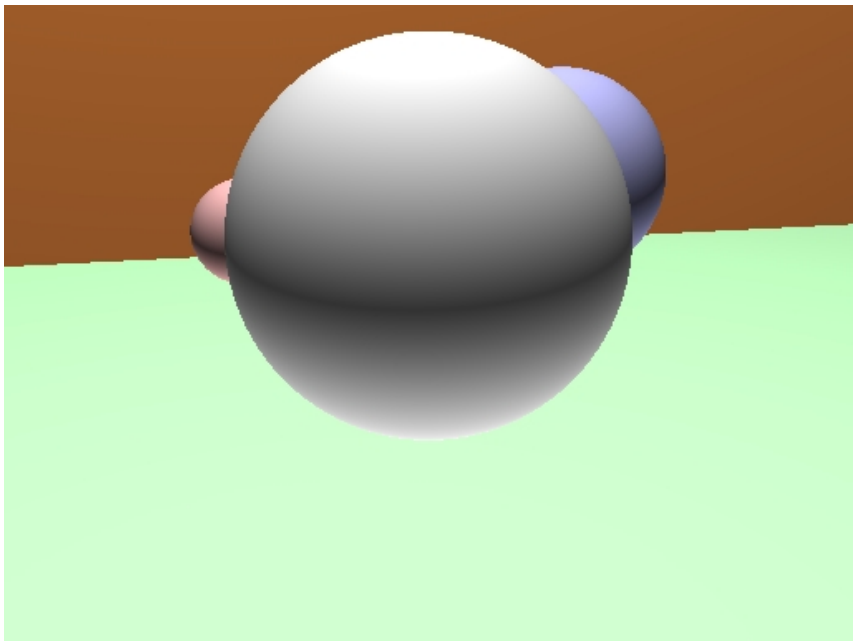
Pour donner plus de relief à l'image, on calcule la quantité de lumière qui atteint l'objet au point d'intersection du rayon.

Dans le modèle de Phong, l'éclairage d'un point est égal à la somme de la contribution de chaque lumière de la scène.

La valeur diffuse de l'éclairage sera proportionnelle au rapport de l'angle d'incidence à la tangente (étalement, dispersion d'énergie).

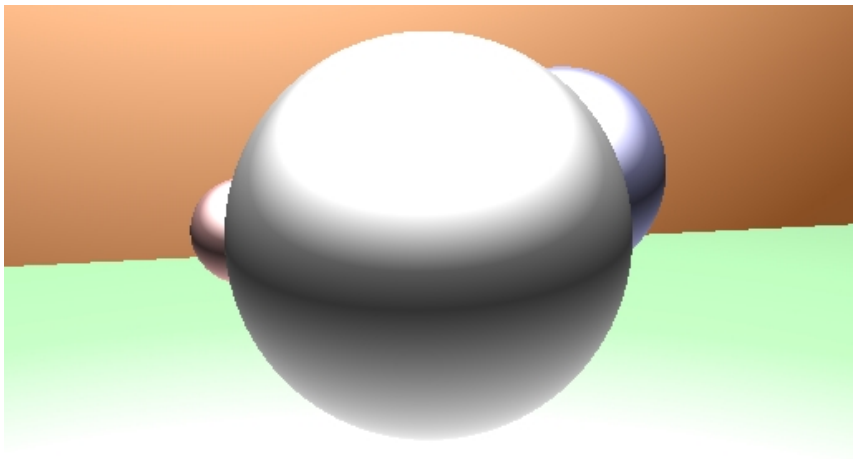
Lumière diffuse

$$I_d = I_o(I_a k_a + I_i k_d \vec{L} \cdot \vec{N})$$



Lumière ponctuelle complète

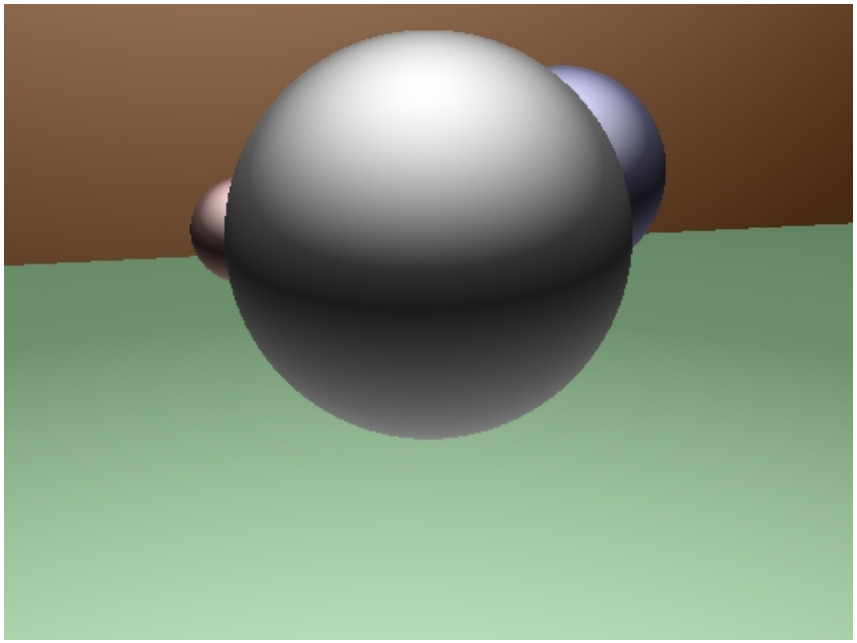
$$I = I_o I_a k_a + I_i (I_o k_d (\vec{L} \cdot \vec{N}) + k_s (\vec{R} \cdot \vec{V})^n)$$



Contrôle de la luminosité

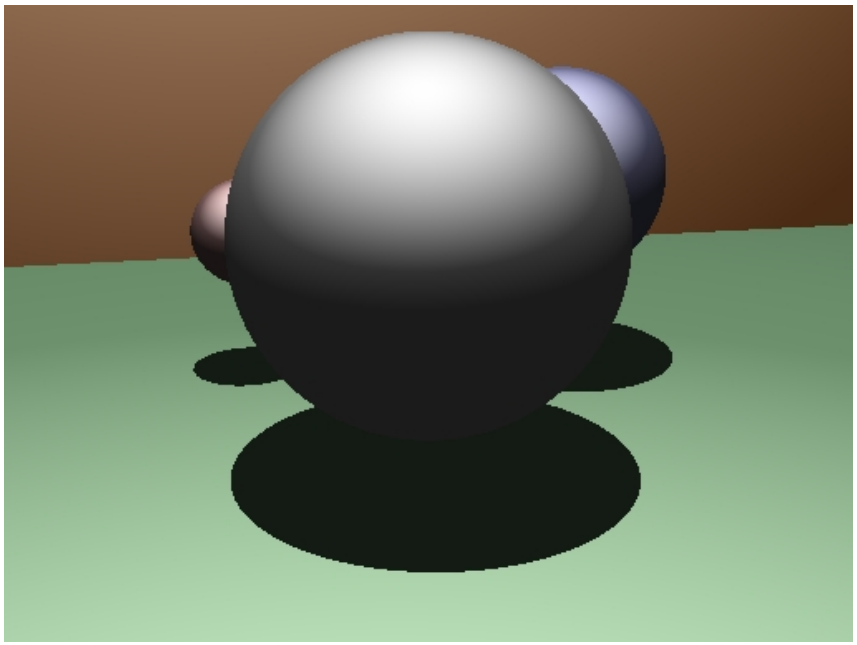
Sans correction, les couleurs obtenues risquent d'être saturées ($c_i \notin [0, 1]$, somme d'illuminations toutes maximales).

Il faut donc normaliser la luminosité de l'image, par exemple en cherchant la valeur maximale de chaque teinte.



Au début étaient les ténèbres

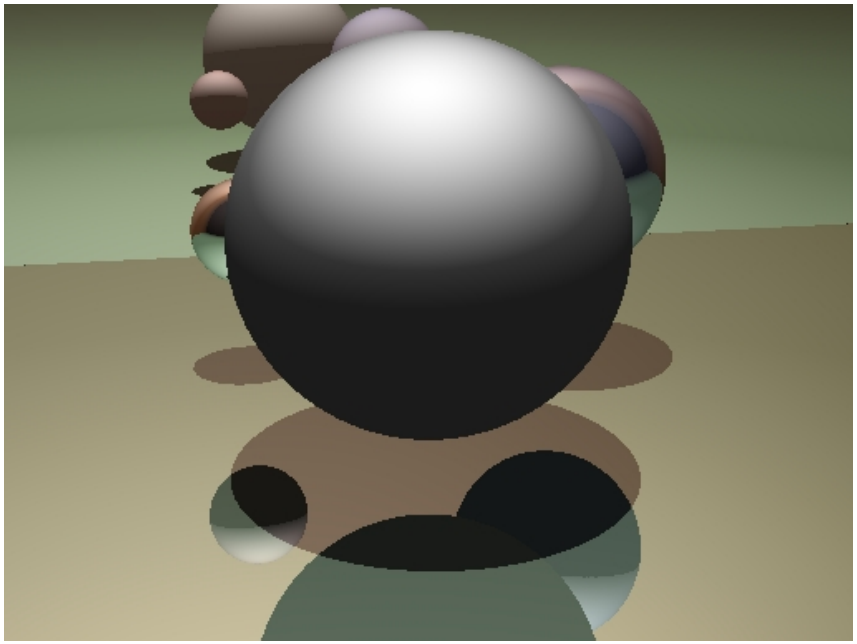
Algorithme Ombre portée(I, S)
creer un rayon R depuis I vers la lumiere S
creer la liste L des intersections de R
si (L n'est pas vide) **alors**
    si ($L[0] < D(I, S)$) **alors**
        retourner couleur noire
    sinon
        retourner l'illumination normale



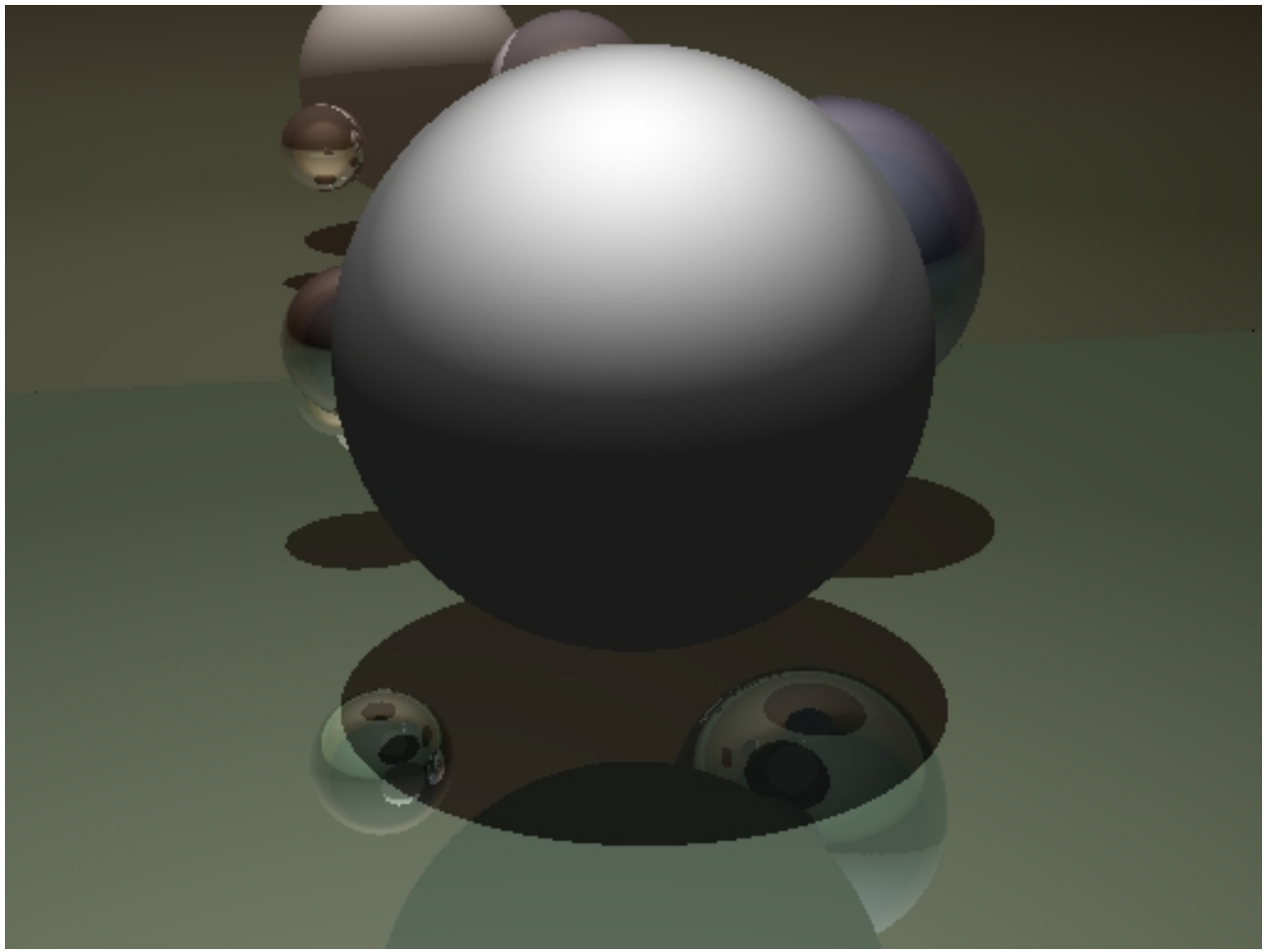
Lancer de rayons récursif

En prenant en compte le coefficient de réflexion $k \in [0, 1]$ de l'objet, on mélange les couleurs suivant la formule :

$$M = C_1 \cdot (1 - k) + C_2 \cdot k$$



Réflexions (profondeur 3)



La réfraction : un pas de trop ?

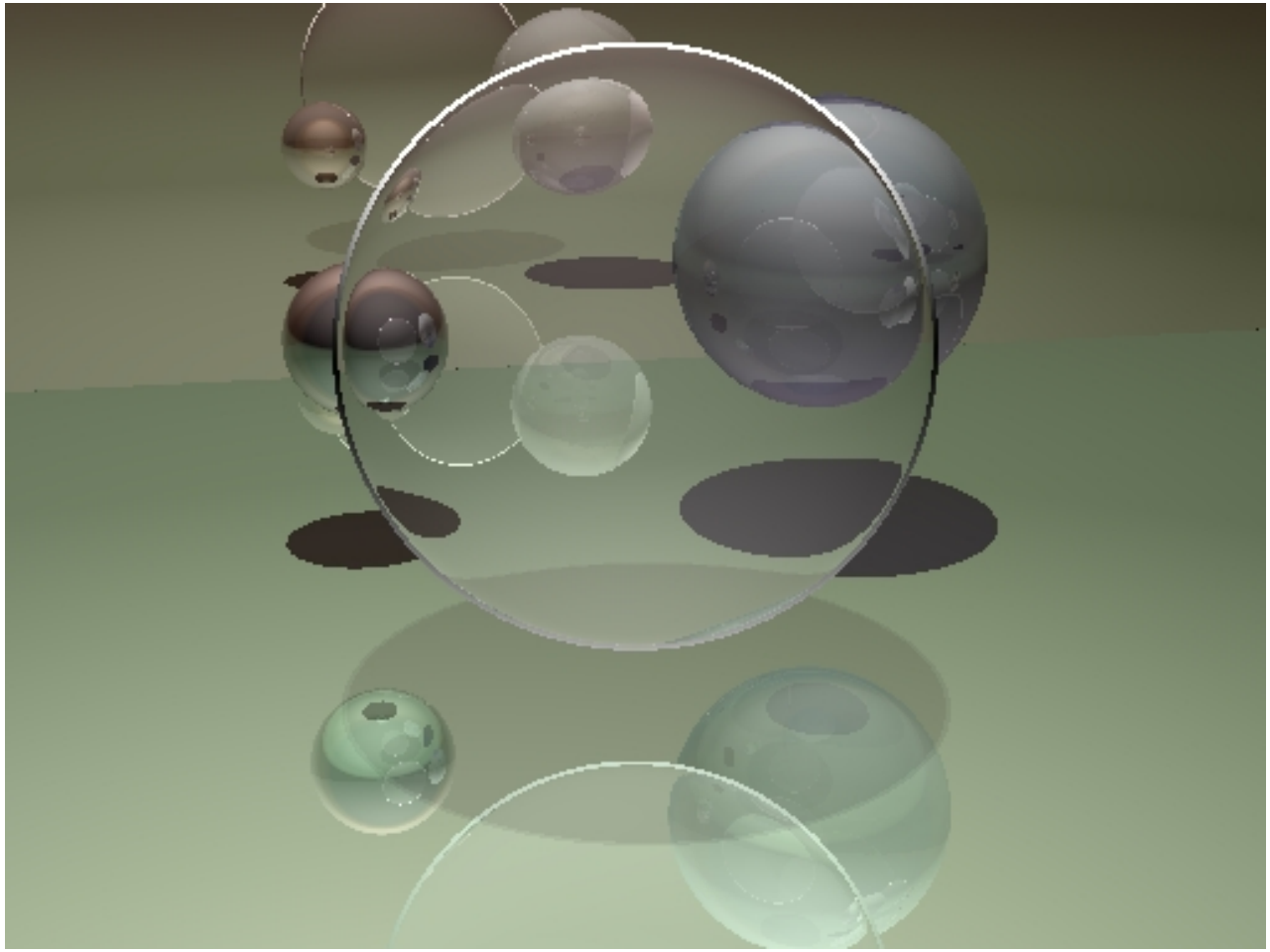
Il faut pouvoir mémoriser à tout instant l'indice de réfraction du milieu dans lequel le rayon progresse : réfléchi il conserve l'indice ; réfracté il prend l'indice du nouveau milieu.

Il faut faire le distingo entre intérieur et extérieur afin de pouvoir réorienter le vecteur normal pour qu'il soit dirigé du côté d'où vient le rayon incident.

Si l'angle d'incidence est trop ouvert (angle critique de la surface) et que le changement de milieu augmente l'angle qu'il doit faire avec la normale alors le rayon est réfléchi.

Simplification des algorithmes de détermination de l'ombre portée (car il devient très difficile de déterminer le chemin qui mène à une source lumineuse).
↳ filtrage de la couleur émise par le rayon par la couleur de la surface traversée (pondérée par le coefficient de transparence) en les multipliant entre elles.

La réfraction : un pas de trop ?



Plan

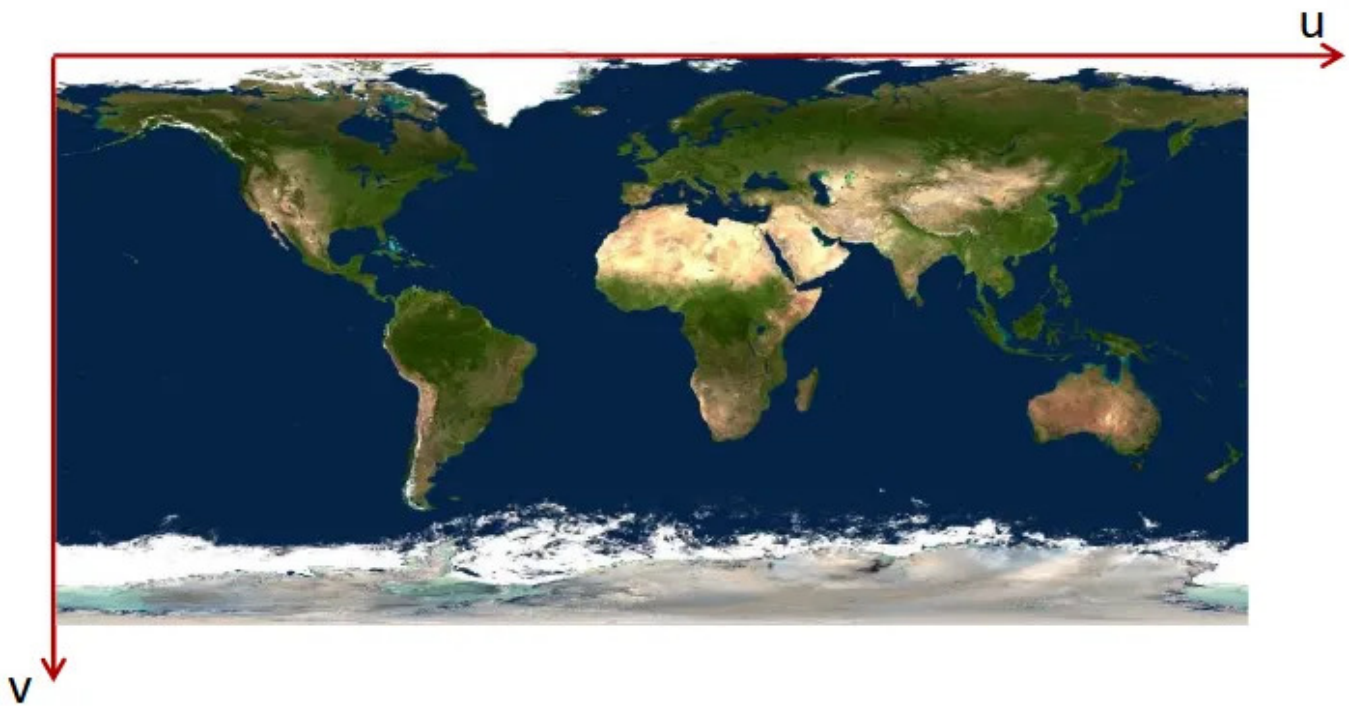
1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

Introduction

Les textures ajoutent un réalisme certain aux images obtenues.

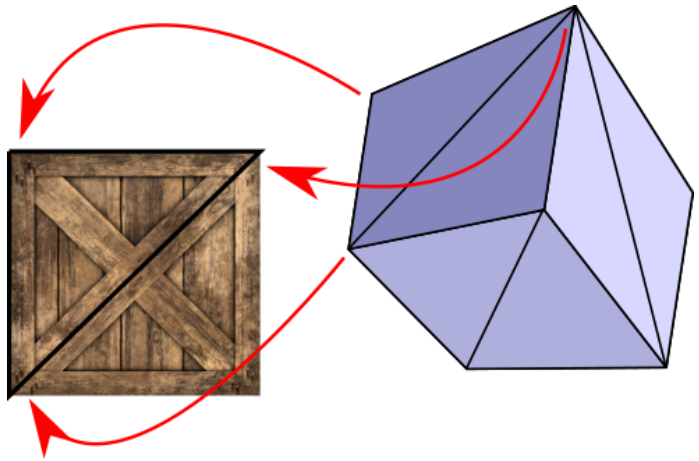
La première étape de ce processus est d'établir une correspondance entre des points d'un objet (3D) et des points d'une texture.

Une texture est définie dans un espace plan (U, V) et ses coordonnées sont comprises dans l'intervalle normalisé $[0, 1]$, afin d'être indépendantes de la taille de la texture.



Correspondance

Dans le cas des polygones triangle et quadrilatère, on associe une coordonnée de la texture à chaque sommet.

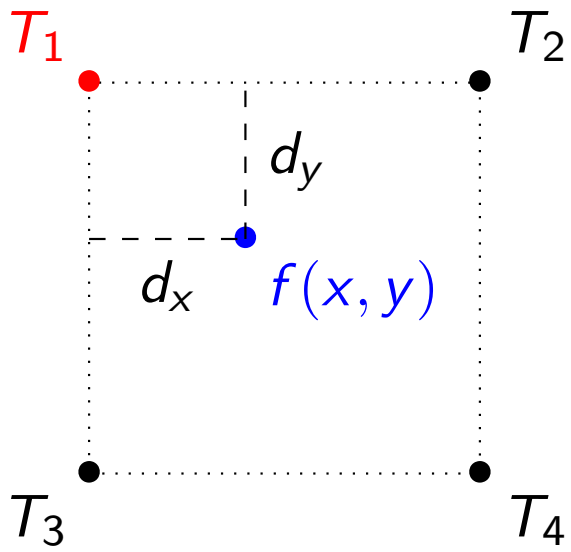


Puis, pour chaque point d'intersection P, on calcule les coordonnées correspondantes de la texture associée, soit directement, soit par interpolation.

Interpolations

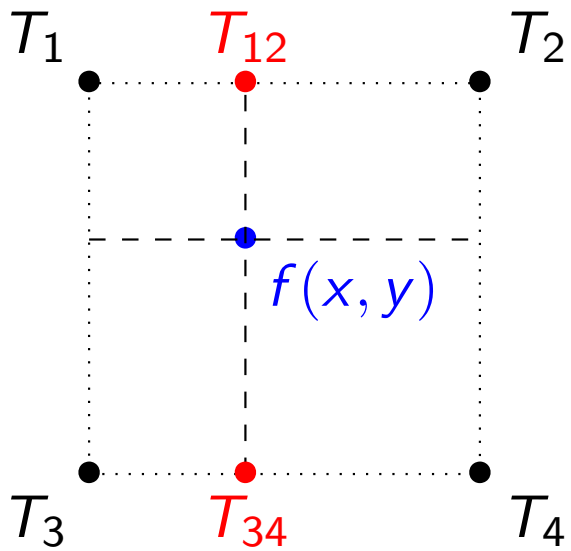
L'interpolation peut être de deux types : au plus proche ou bilinéaire.

L'interpolation la plus simple consiste à trouver le texel le plus proche.



Interpolation bilinéaire

L'interpolation bilinéaire permet de calculer la valeur d'une fonction en un point quelconque, à partir de ses deux plus proches voisins dans chaque direction.

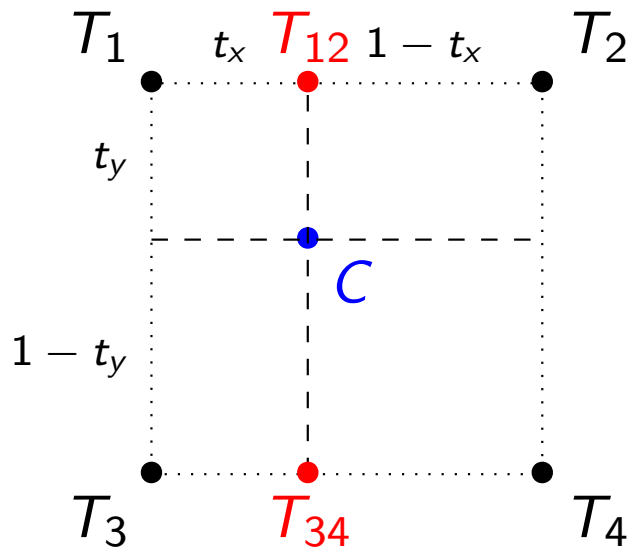


$$f(x, y) = ax + by + cxy + d$$

a, b, c, d sont des constantes déterminées à partir des quatre voisins (x_1, y_1), (x_1, y_2), (x_2, y_1), (x_2, y_2) du point (x, y) dont on cherche la valeur.

Interpolation bilinéaire

Dans l'espace (discret) d'une texture :



$$T_{12} = (1 - t_x) \cdot T_1 + t_x T_2$$
$$T_{34} = (1 - t_x) \cdot T_3 + t_x T_4$$

Ce qui nous permet d'interpoler linéairement C :

$$C = (1 - t_y) \cdot T_{12} + t_y T_{34}$$

Cas des triangles

On paramétrise un triangle par le calcul du barycentre de ses sommets.

Soit P_0, P_1 et P_2 les sommets du triangle, et β_0, β_1 et β_2 les coordonnées barycentriques d'un point P, alors :

$$P = \beta_0 P_0 + \beta_1 P_1 + \beta_2 P_2, \text{ avec } \beta_0 + \beta_1 + \beta_2 = 1$$

On calcule les coordonnées (s, t) de la texture, en résolvant le système linéaire suivant en (β_1, β_2) :

$$P = (1 - \beta_1 - \beta_2) P_0 + \beta_1 P_1 + \beta_2 P_2$$

$$x = (1 - \beta_1 - \beta_2)x_0 + \beta_1 x_1 + \beta_2 x_2$$
$$y = (1 - \beta_1 - \beta_2)y_0 + \beta_1 y_1 + \beta_2 y_2$$

Ou encore :

$$\beta_1(x_1 - x_0) + \beta_2(x_2 - x_0) = x - x_0$$
$$\beta_1(y_1 - y_0) + \beta_2(y_2 - y_0) = y - y_0$$

C'est la transformation linéaire :

$$M \cdot \beta = u - u_0 \Leftrightarrow \beta = M^{-1}(u - u_0)$$

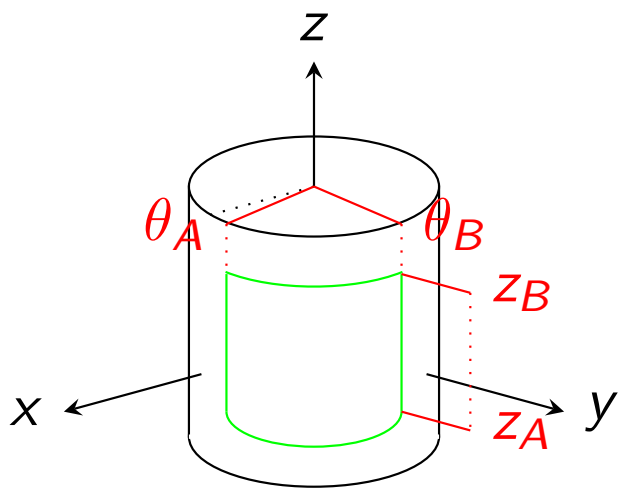
car la matrice M est inversible [3], donc :

$$\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \frac{1}{\det M} \begin{pmatrix} y_2 - y_0 & x_0 - x_2 \\ y_0 - y_1 & x_1 - x_0 \end{pmatrix} \begin{pmatrix} x - x_0 \\ y - y_0 \end{pmatrix}$$

3. car $u_1 - u_0$ et $u_2 - u_0$ sont linéairement indépendants, si ce n'était pas le cas u_0, u_1 et u_3 seraient colinéaires et ne formeraient pas un triangle.

Cas des cylindres

Soient un point $P(x, y, z)$ et (s, t) les coordonnées de la texture.



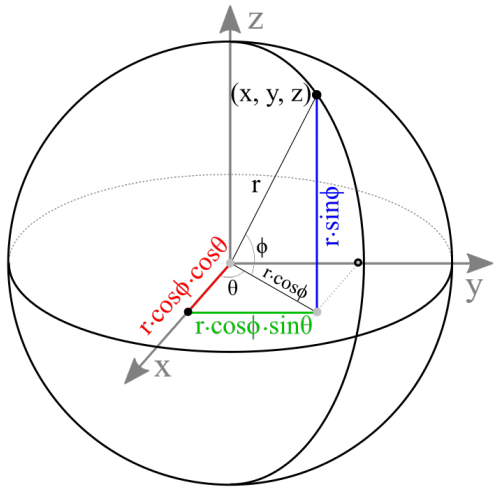
$$x = r\cos\theta,\ \ y = r\sin\theta \text{ avec } \theta \in [\theta_A, \theta_B],\ \ z \in [z_A, z_B]$$

On en déduit les coordonnées (s, t) :

$$s = \frac{\theta - \theta_A}{\theta_B - \theta_A} \text{ et } t = \frac{z - z_A}{z_B - z_A}$$

Cas des sphères

Soient un point $P(x, y, z)$ et (s, t) les coordonnées de la texture.



$$x = r \sin\theta \cos\phi, \;\; y = r \sin\theta \sin\phi, \;\; z = r \cos\theta$$

avec

$$\theta \in [0, \pi], \phi \in [0, 2\pi]$$

On en déduit les coordonnées (s, t) :

$$s = \frac{\theta}{\pi} = \frac{\cos^{-1}(z/r)}{\pi}, \;\; t = \frac{\phi}{2\pi} = \frac{\cos^{-1}(x/r \sin(\pi s))}{2\pi}$$

Plan

1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

Limites

Calculs coûteux :

- cohérence spatiale perdue car on lance des rayons indépendamment les uns des autres,
- les calculs d'intersection se font en flottant et consistent à chercher les racines de fonctions polynomiales.

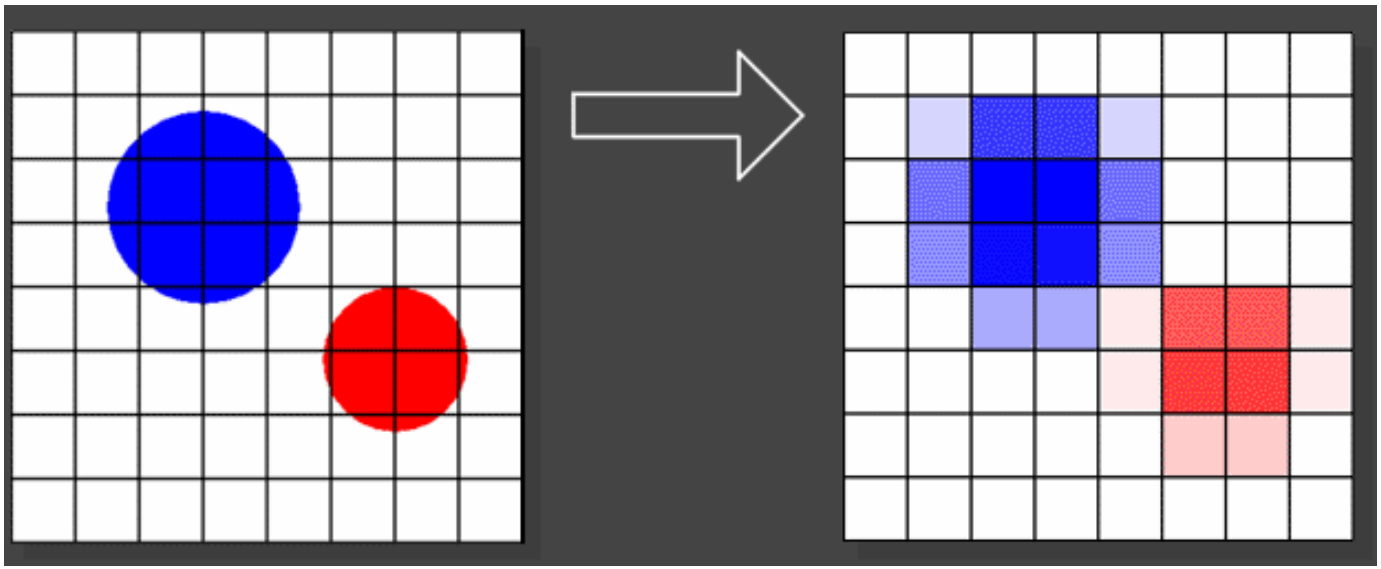
Ignore les petits objets :

- le sur-échantillonnage ne résoud rien car il s'active à partir d'une différence de couleur entre deux pixels voisins

Limites

Crénelage (*aliasing*)

- un filtrage pondéré n'est pas possible car on ne sait pas a priori quel pourcentage du pixel occupe la surface,
- sur-échantillonnage (*supersampling*), adapté (*adaptative supersampling*) : se fait lorsque la différence de couleur entre deux proches pixels est supérieure à un seuil,
- *jittering* : évite certains effets d'une grille fixe en décalant légèrement les rayons sur-échantillonnées par perturbation.



Limites

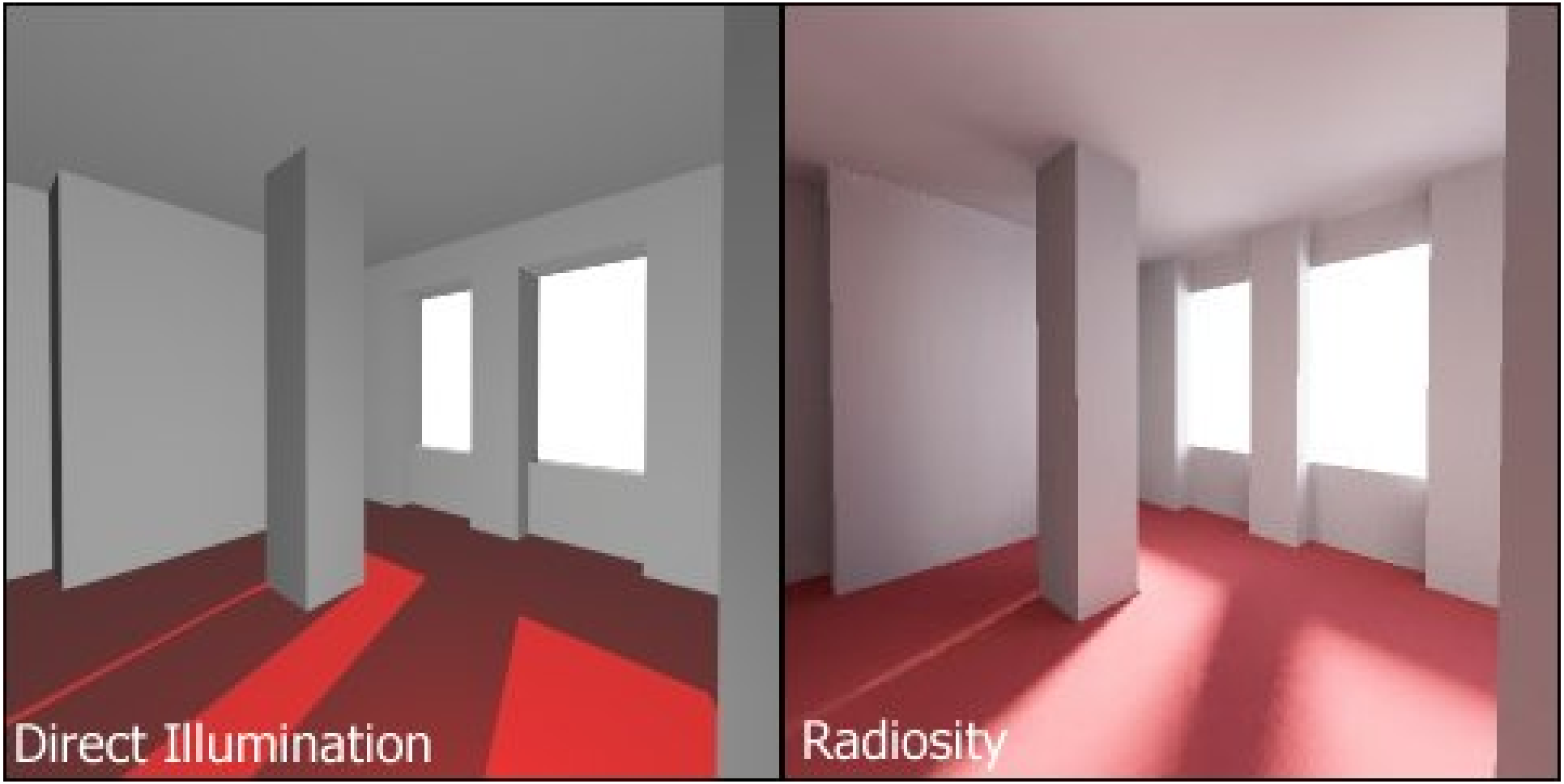
Lumière source de nombreuses approximations :

- une seule couleur,
- pas d'interférence ou de diffraction,
- source infiniment petite,
- illumination en tout ou rien.

De nombreux problèmes :

- arc en ciel,
- pénombre,
- illumination spéculaire,
- lumière ambiante (radiosité),
- réfraction sans focalisation,
- surbrillance, ...

Limites



Directions de recherche principales

- amélioration des algorithmes d'intersection : lancé de faisceaux, subdivision spatiale, adaptation de l'algorithme d'intersection aux « vrais » objets (pas d'approximation polyhédrique),
- antialiasage : le lancé de faisceaux, le lancé de cônes,
- augmentation de la palette d'effets optiques : modèle atmosphérique, combinaison avec la radiosité, illumination globale,
- prise en compte des petits objets : système de particules pour la poussière, les gouttes de pluie, ...

Plan

1. Introduction
2. Le rendu
3. Modèles d'illumination
4. Géométrie du lancer de rayons
5. Placage de texture
6. Limites et améliorations
7. POV-Ray

POV-Ray

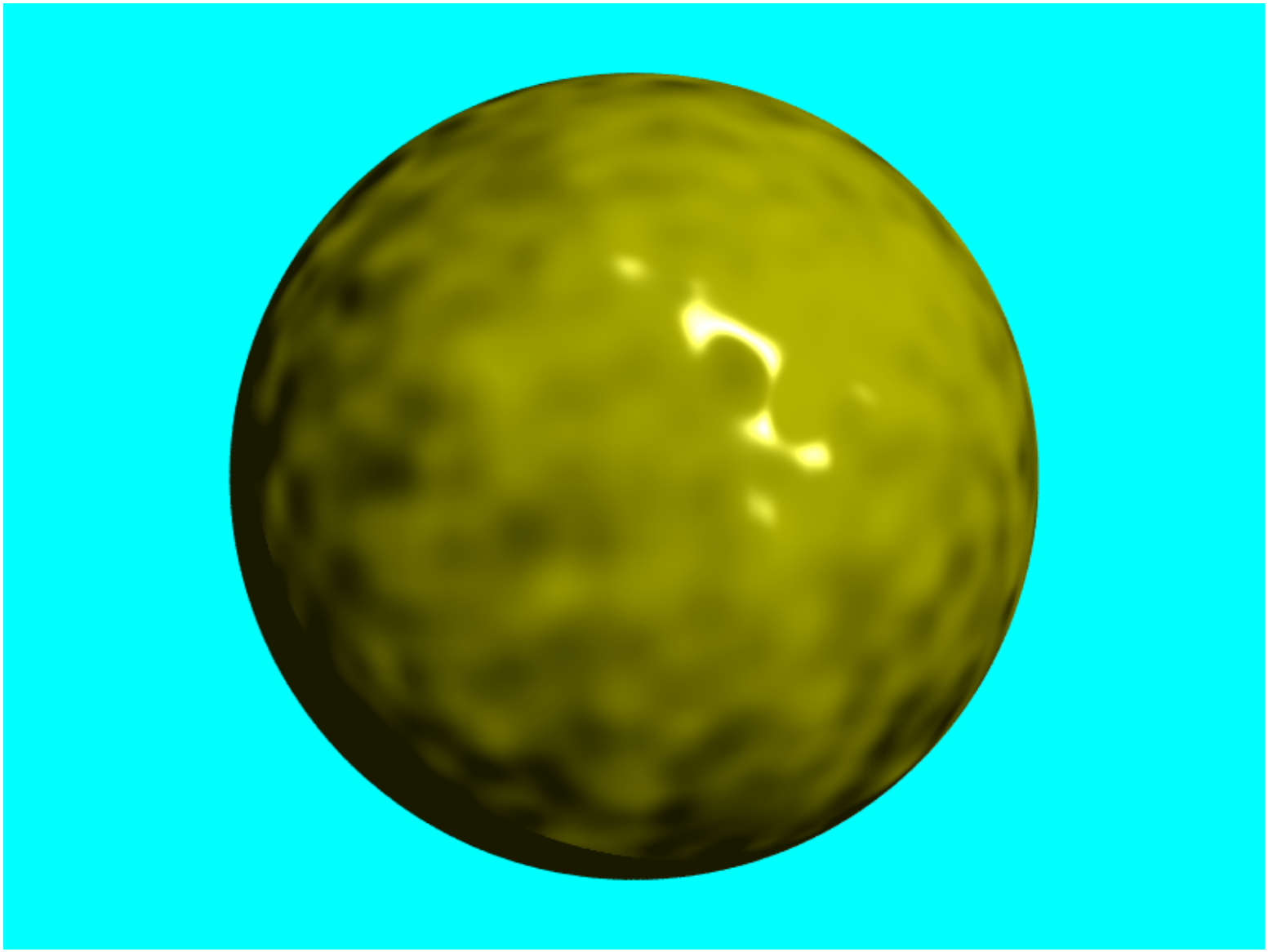
Persistence of Vision Ray Tracer est gratuit et portable.

Principales caractéristiques :

- language de description de scènes 3D photoréalistes,
- nombreux fichiers d'exemples,
- fichiers en-tête définissant des formes, des couleurs et des textures,
- grande qualité d'images (résolution jusqu'à 4096×4096 en 48 bits),
- visualisation durant les calculs,
- possibilité d'interrompre le traitement et de le reprendre par la suite.

```
#include "colors.inc"    // The include files contain
#include "shapes.inc"    // pre-defined scene elements
#include "textures.inc"

camera {
  location  <0, 2, -3>
  look_at   <0, 1,  2>
}

light_source { <2, 4, -3> color White}

background { color Cyan }

sphere {
  <0, 1, 2>, 2
  texture {
    pigment {color Yellow}  // pre-defined in COLORS.INC
    normal {bumps 0.4   scale 0.2}
    finish {phong 1}
}
```

Copie d'écran



La totale