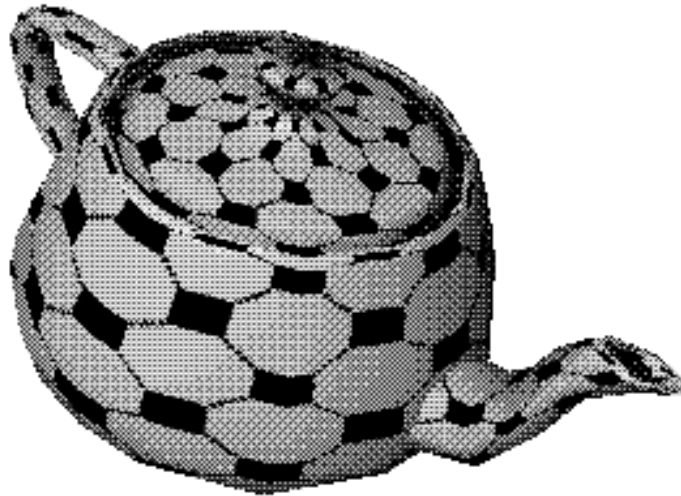


Rendu Réaliste



Alexis Nédélec

Ecole Nationale d'Ingénieurs de Brest
Technopôle Brest-Iroise, Site de la Pointe du Diable
CP 15 29608 BREST Cedex (FRANCE)
e-mail : nedelec@enib.fr

Table des Matières

Introduction	3
La Couleur	6
Textures	19
Elimination des Parties Cachées	25
Modèles d'Eclairage	27
Lissage de Gouraud	31
Lissage de Phong	33
Optique Géométrique	35
Calculs d'Intersections	45
Ray-Tracing: Introduction	57
Ray-Tracing: Approche d'Implémentation	67
Radiosité: Introduction	73
Bibliographie	87

Introduction

- ▷ Couleurs affectées aux objets
 - ◇ RGB, XYZ (CIE : Commission Internationale de l'Eclairage)
 - ◇ CMJ : Cian, Magenta, Jaune (synthèse soustractive)
 - ◇ TSL : Teinte, Saturation, Luminosité (Munsell, DIN)
- ▷ Textures
 - ◇ plaquage de textures 2D
 - ◇ sculpture de textures 3D
- ▷ Elimination des parties cachées

Introduction

- ▷ Caractéristiques des surfaces
 - ◇ géométriques
 - ◇ photométriques
- ▷ modèles d'éclairage
 - ◇ source de lumières
 - ◇ caractéristiques des surfaces
 - ◇ lois de l'optique géométrique (Snell-Descartes)
 - ◇ éclairage local sur les objets (Phong, Gouraud, ...)
 - ◇ éclairage global sur les objets (Ray-Tracing, Radiosité, ...)

Introduction



La Couleur

La couleur peut-être étudiée sous différents points de vue

- d'optique géométrique
 - ▷ réflexion, réfraction, absorption
 - ▷ spéculaire, diffuse
- d'optique physique
 - ▷ Ondes Electro-Magnétiques
 - ▷ composition spectrale (des U.V aux I.R.)
 - ▷ interférence, diffraction
 - ▷ interaction lumière/matière
 - ▷ ...

La Couleur

- études neurophysiologiques
 - ▷ globe oculaire
 - ▷ perception rétinienne
 - ▷ nerf optique, chiasma optique
 - ▷ aires visuelles primaires, secondaires
- études psychophysiques, psychologiques
 - ▷ perception : daltonisme, ...
 - ▷ interprétation : bleu/froid, rouge/chaud, ...

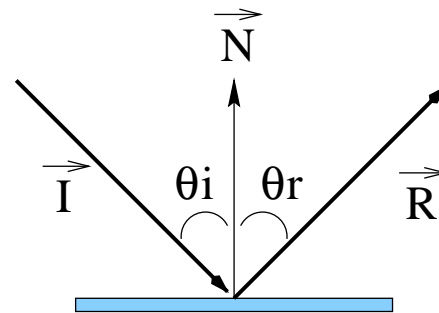
La Couleur

La couleur comme phénomènes

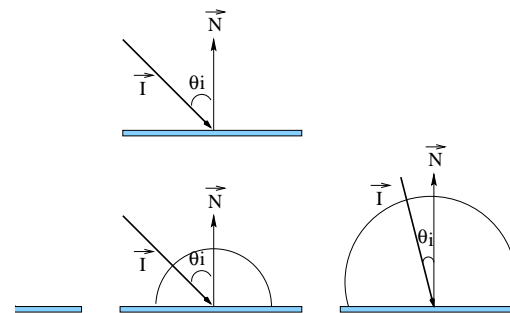
- de synthèse
 - ▷ additive : Rouge, Vert, Bleu (RGB)
 - ▷ soustractive: Cyan, Magenta, Jaune (CMY)
- de mesure
 - ▷ Commission Internationale de l'Eclairage (CIE : RGB,XYZ)
 - ▷ Teinte, Saturation, Valeur (HSV)
 - ▷ Teinte, Saturation, Luminance (HSI)
 - ▷ ...

Optique Géométrique

Réflexion spéculaire

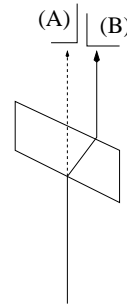


Réflexion diffuse

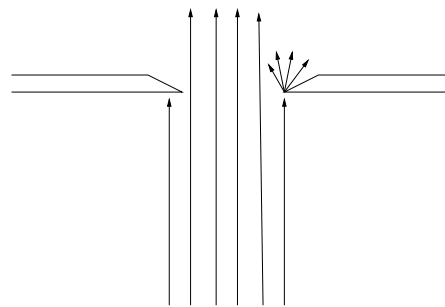


Optique Géométrique

Réfraction ou transmission

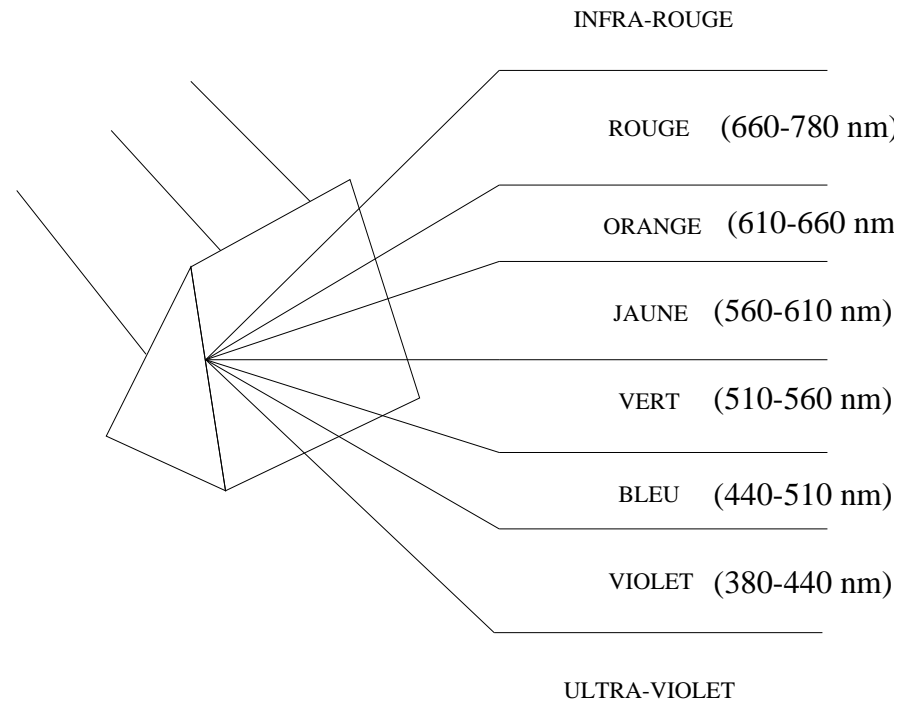


Diffraction de la lumière



Optique Physique

Composition spectrale



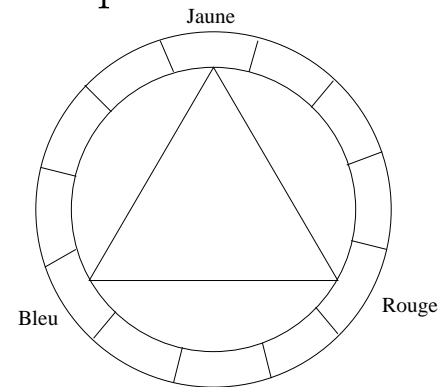
Interprétation

La couleur comme phénomène psychologique

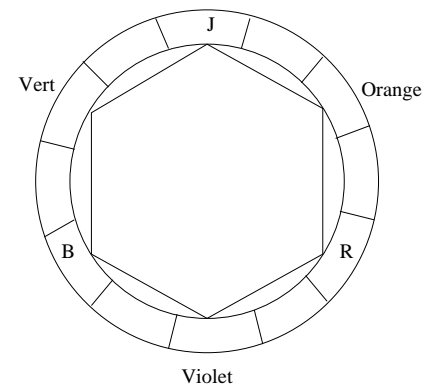
- ▷ Rouge : chaleur, force, joie, excitation, violence,...
- ▷ Orange : chaleur, gaieté, feu, soleil, lumière,...
- ▷ Jaune : or, soleil, richesse, jeunesse, vie, ldots
- ▷ Vert : apaisement, détente, repos, espoir, renouvellement, ...
- ▷ Bleu : froid, repos, calme, fraîcheur, mer, ciel,...
- ▷ Violet : rêve, utopie, spiritualité, mélancolie, maladie, ...
- ▷ Blanc : pureté, sagesse, lumière, connaissances,...
- ▷ Noir : mort, désespoir,...

Cercle Chromatique

Cercle chromatique des couleurs primaires



Cercle chromatique des couleurs secondaires



Cercle Chromatique

Harmonisation de couleurs par contraste

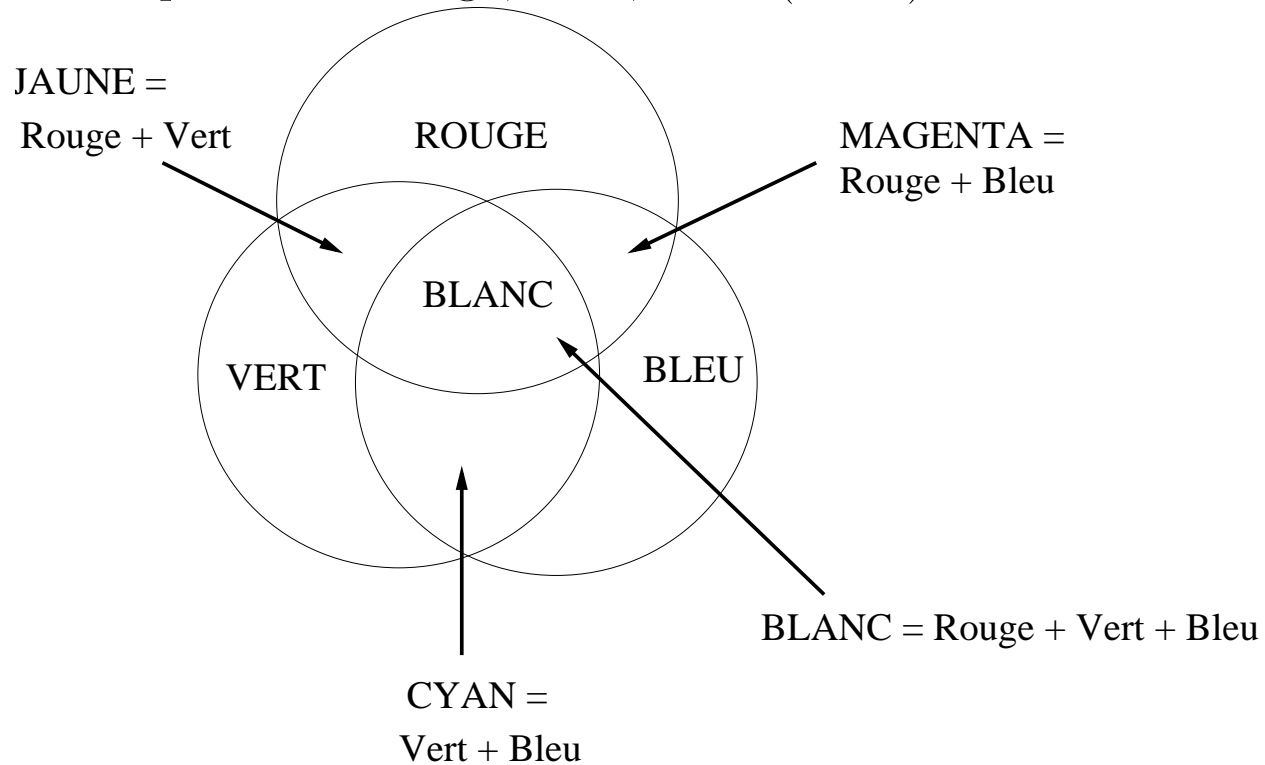
- ▷ Harmonie objective : perception de la couleur complémentaire
- ▷ Harmonie subjective : chaud-froid, ...

Que l'on peut retrouver sur le cercle chromatique

- ▷ Deux couleurs : diamétralement opposées (Rouge, Vert)
- ▷ Trois couleurs : triangle (Jaune, Bleu, Rouge)
- ▷ Quatres couleurs : carré (Jaune, Rouge-Orange, Violet, Bleu-Vert)
- ▷ Six couleurs : Hexagone (Jaune, Orange, Rouge, Violet, Bleu, Vert)

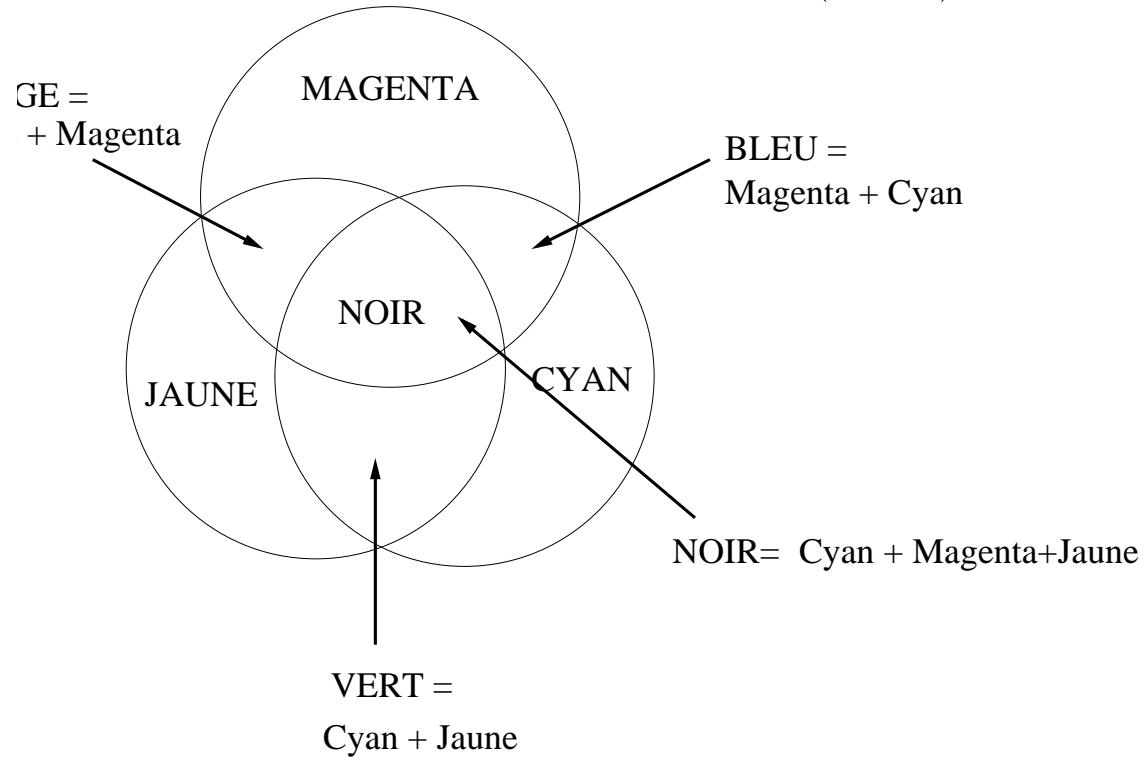
Synthèse Additive

A partir des composantes Rouge, Vert, Bleu (RGB)



Synthèse soustractive

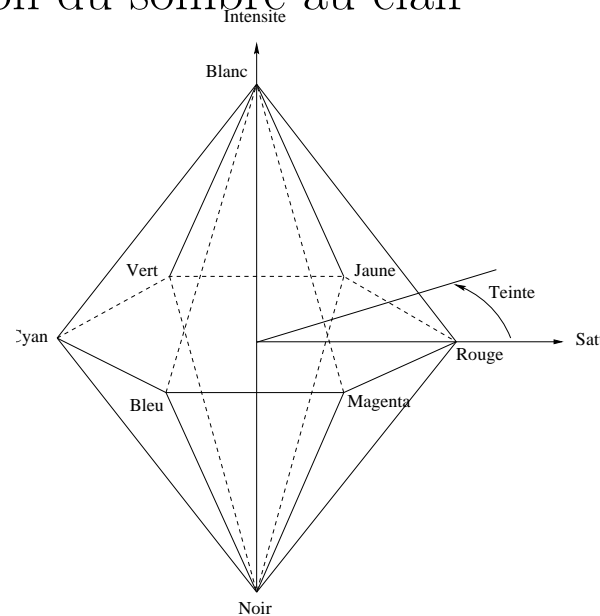
A partir des composantes Cyan, Magenta, Jaune (CMJ)



Système TSL

Teinte, Saturation, Luminance (Hue, Saturation, Value)

- ▷ Teinte : qualité de la couleur (rouge, vert, bleu, etc.)
- ▷ Saturation : pureté (intensité) de la couleur
- ▷ Luminance : variation du sombre au clair



Affichage

Deux types d'affichage : indexée (2, 4, 8 bits) ou direct (16, 24 bits)

Affichage indexé de la couleur

- ▷ Color LookUp Table (CLUT) : table d'index (4, 16, 256 entrées)
- ▷ Valeur d'index le plus proche de la couleur demandée
- ▷ conversion RAMDAC
 - ◇ RAM : conversion de l'index en valeurs RGB
 - ◇ DAC : conversion RGB en signal vidéo (Digital-Analog Converter)

Affichage direct de la couleur

- ▷ mode 24 bits : 3 canaux de 8 bits (16,7 millions de couleurs)
- ▷ mode 16 bits : 3 canaux de 5 bits (32.768 couleurs)

Textures

On distingue différents types de plaquage de textures

- ▷ Textures 2D : plaquage d'un motif rectangulaire sur une surface 3D
- ▷ Textures 3D : texture définie en 3D, on sculpte l'objet dans le bloc de texture
- ▷ HyperTextures, Texels : rendu d'objets à micro-structures (poils, feu, ...)
- ▷ Textures synthétiques : calculées mathématiquement (briques, rochers, ...)
- ▷ Textures numériques : à partir de textures existantes

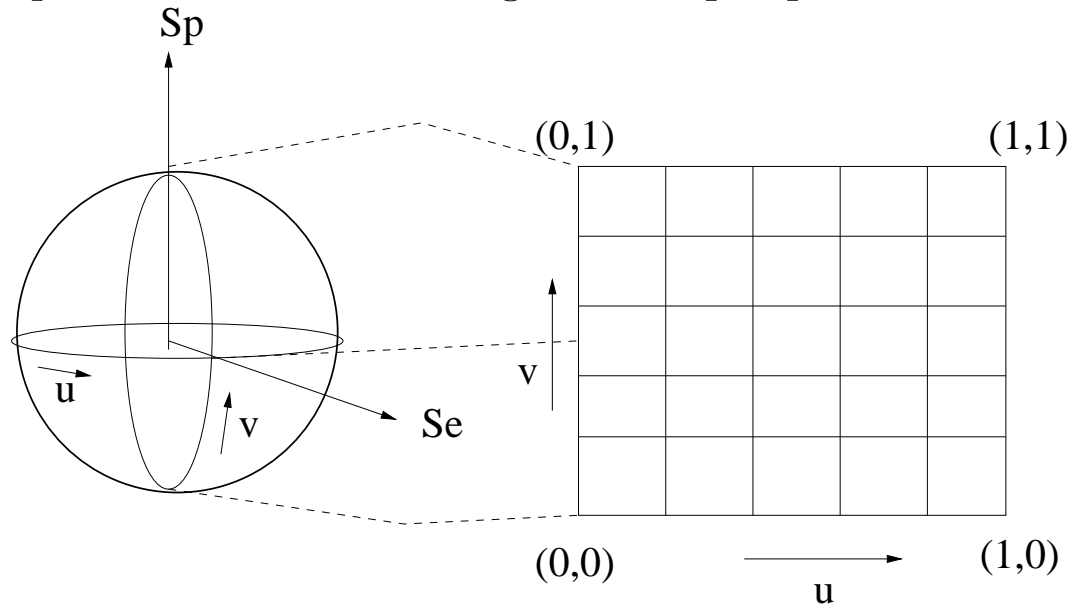
Intérêt du plaquage de texture : Degré de réalisme sur volumes simples

Problème du mapping, établir une correspondance **exacte** entre

- ▷ trame rectangulaire à mapper
- ▷ et surface **quelconque** (de la sphère aux surfaces “libres”)

Plaquage de Textures

Texture définie par une trame rectangulaire à plaquer sur une surface 3D



Problème : Trouver une correspondance Point de Surface / Point de Trame

Plaquage de Textures

Un point de trame est défini par 2 paramètres $u, v \in [0, 1]$

Caractéristiques d'une sphère

▷ Equateur : $\vec{S}_e(X_e, Y_e, Z_e)$

▷ Pôle : $\vec{S}_p(X_p, Y_p, Z_p)$

En chaque point de sphère on connaît la normale : $\vec{N}(X_n, Y_n, Z_n)$

Correspondance en longitude (détermination du paramètre v)

$$\phi = \arccos(-\vec{N} \cdot \vec{S}_p)$$

$$v = \frac{\phi}{\pi}$$

Plaquage de Textures

Correspondance en latitude (détermination du paramètre u)

$$\theta = \arccos\left(\frac{-\vec{S}_e \cdot \vec{N}}{\sin \phi}\right) \cdot \frac{1}{2\pi}$$

Test pour paramétrer u dans l'intervalle $[0,1]$

- ▷ $(\vec{S}_p \wedge \vec{S}_e) \cdot \vec{N} > 0 \Rightarrow u = \theta$
- ▷ $(\vec{S}_p \wedge \vec{S}_e) \cdot \vec{N} < 0 \Rightarrow u = 1 - \theta$

Plaquage de Textures

Modes de projection de textures

- ▷ Projection Planaire : enveloppe de papier
- ▷ Projection Cylindrique : étiquette de bouteille
- ▷ Projection Sphérique : globe terrestre

Autre type de plaquages

- ▷ plaquage de relief (bump mapping): perturbation de normales
 - ◇ briques, ...
- ▷ plaquage d'environnement: surimpression de scène sur objet
 - ◇ objet réfléchissant
- ▷ plaquage d'opacité : objet opaque et transparent
 - ◇ effets de grilles, ...

Caractéristiques de surfaces

- ▷ couleur, brillance : spéculaire (reflets), diffuse, ambiente
- ▷ texture : représentation de la matière
- ▷ transparence : taux de lumière transmise à travers la surface
- ▷ réflectance : taux de lumière réfléchi par l'objet
- ▷ rugosité : taux de réflexion dans toutes les directions
- ▷ radiance : lumière émise par la surface indépendamment des sources lumineuses

Élimination des Parties Cachées

Tampon de profondeur (Z-Buffer)

- ▷ introduite par Catmull (1974)
- ▷ pour des surfaces quelconques
- ▷ codage par pixel (n-bits couleur, p-bits profondeur)
- ▷ 24-32 bits / pixel pour la profondeur

La profondeur

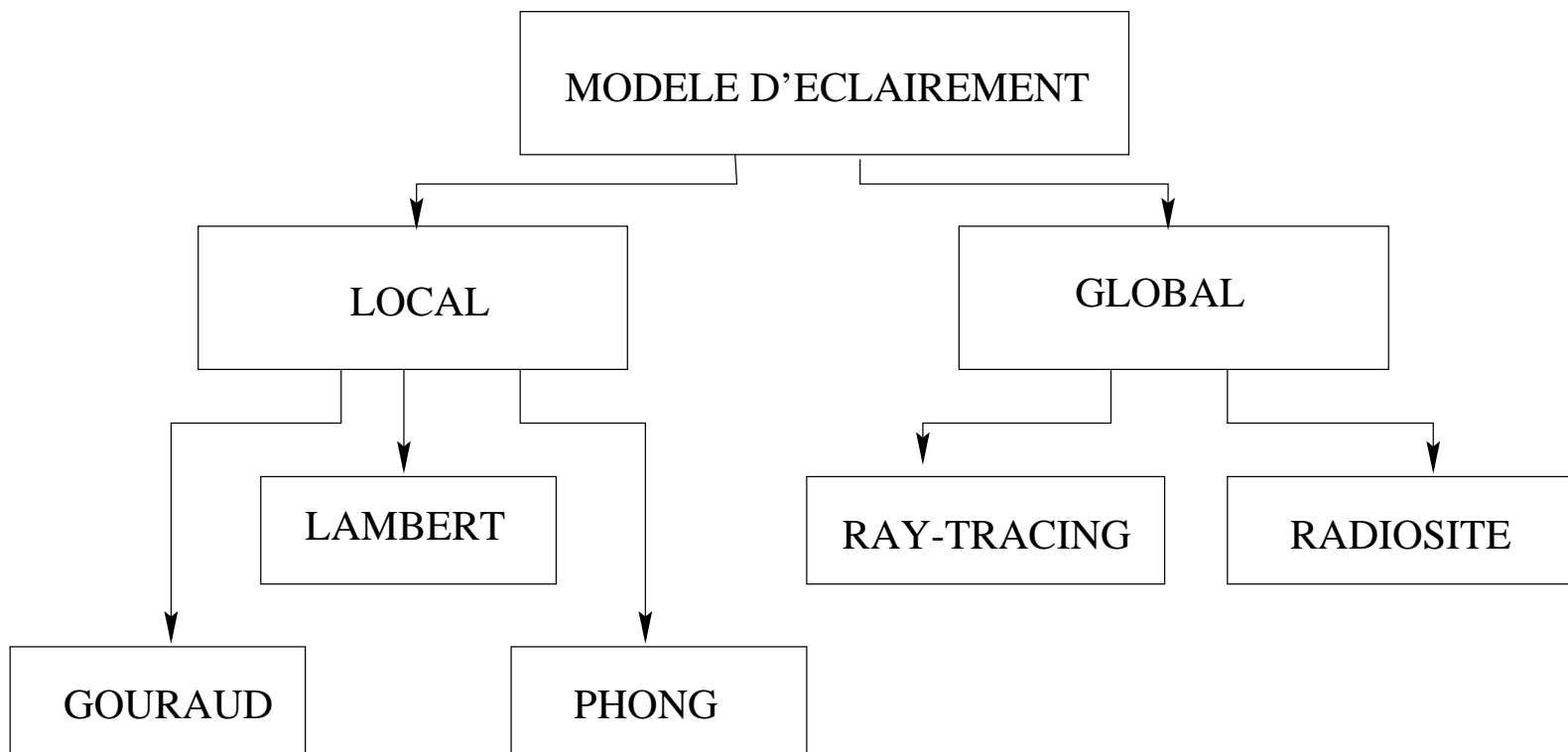
- ▷ représente la distance Œil/Pixel/Objet
- ▷ est initialisée au maximum possible $2^p - 1$

Élimination des Parties Cachées

Algorithme du Z-Buffer

```
POUR chaque pixel(x,y) ecran FAIRE
{
  z(x,y) <- zmax;
  couleur(x,y) <- background(x,y);
}
POUR chaque objet O de la scene FAIRE
  POUR chaque pixel(x,y) sur lequel O se projette FAIRE
  {
    z0 <- distanceObjet(O,x,y);
    SI z0 < z(x,y) ALORS
    {
      z(x,y) <- z0;
      couleur(x,y) <- couleurObjet(O,x,y);
    }
  }
}
```

Modèles d'Eclairage



Modèles d'Eclairage

Simulation du comportement de la lumière de la source à l'œil

- ▷ modèles empiriques (Lambert, Phong, Gouraud)
- ▷ modèles physiques (Lancer de rayons, Radiosité)

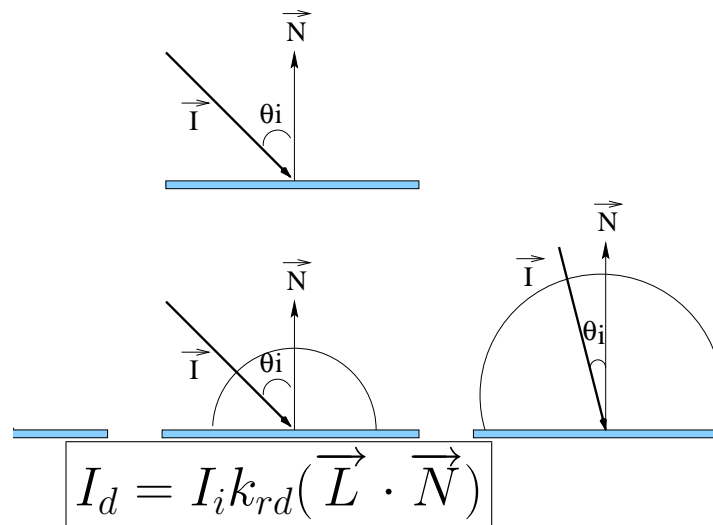
Modèle de Lambert

- ▷ une couleur par polygone
- ▷ contour des facettes visibles

Elimination de la visibilité des facettes par les méthodes de lissage

- ▷ Gouraud : interpolation de la couleur
- ▷ Phong : interpolation des normales aux faces

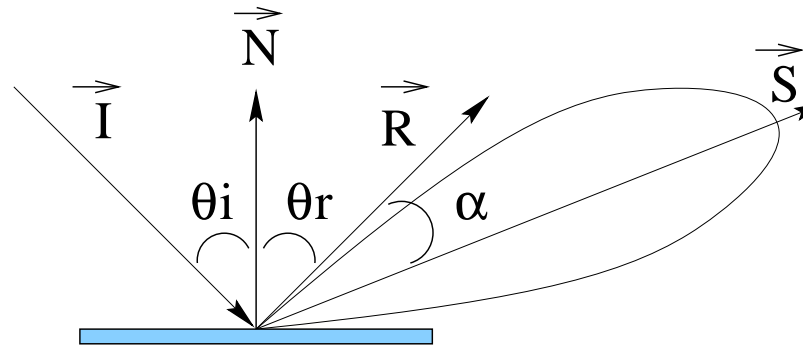
Modèle de Lambert



- ▷ I_i : intensité de la lumière incidente
- ▷ $k_{rd} \in [0, 1]$: coefficient de réflexion diffuse
- ▷ \vec{N} : normale à la surface
- ▷ \vec{L} : rayon issu de la source lumineuse

Modèle de Phong

Réflexion spéculaire

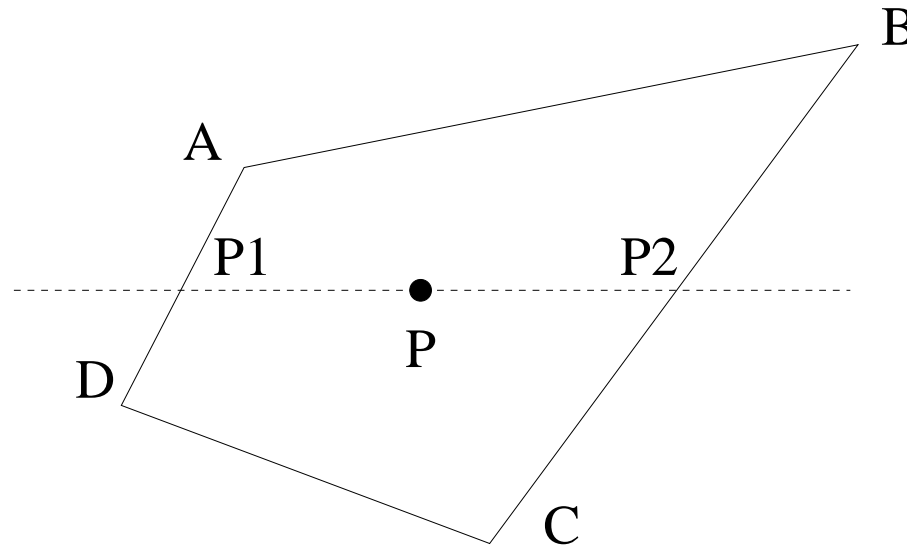


$$I_s = I_i k_s (\vec{R} \cdot \vec{S})^n$$

- ▷ $k_s \in [0, 1]$: coefficient de réflexion spéculaire
- ▷ \vec{S} : rayon dans la direction de l'observateur
- ▷ n : plus n est grand, plus la réflexion est spéculaire

Lissage de Gouraud

$$I_1 = I_D \frac{Y_A - Y_1}{Y_A - Y_D} + I_A \frac{Y_1 - Y_D}{Y_A - Y_D}, I_2 = I_C \frac{Y_B - Y_2}{Y_B - Y_C} + I_B \frac{Y_2 - Y_C}{Y_B - Y_C}$$



$$I_P = I_2 \frac{X_P - X_1}{X_2 - X_1} + I_1 \frac{X_2 - X_P}{X_2 - X_1}$$

Lissage de Gouraud

```
POUR chaque polygone FAIRE
    Calcul de la normale au polygone
POUR chaque sommet (A,B,...) de polygone FAIRE
    Calcul de normale : moyenne des normales aux faces incidentes
POUR chaque sommet (A,B,...) de polygone FAIRE
    Calcul de couleur par un modele d'eclairément local (lambert)
POUR chaque point P FAIRE
    SI P appartient a une arete [A,B] de polygone ALORS
        Calcul de la couleur  $I_p$  par interpolation lineaire entre [IA,IB]
    SINON
        Calcul de la couleur  $I_p$  en balayage horizontal par interpolation entre [I1,I2]
```


Lissage de Phong

POUR chaque polygone FAIRE

 Calcul de la normale au polygone

POUR chaque sommet (A,B,...) de polygone FAIRE

 Calcul de normale : moyenne des normales (NA,NB,...)aux faces incidentes

POUR chaque point P appartenant a une arete FAIRE

 Calcul de la normale NP_i par interpolation lineaire entre [A,B]

$$NP_i(t) = t NA + (1-t) NB$$

POUR chaque point P interieur au polygone FAIRE

 Calcul de la normale en ce point par interpolation lineaire

$$NP = t NP_1 + (1-t) NP_2$$

POUR chaque point P ou la normale est determine

 Calcul de couleur par un modele d'eclairément local (lambert)

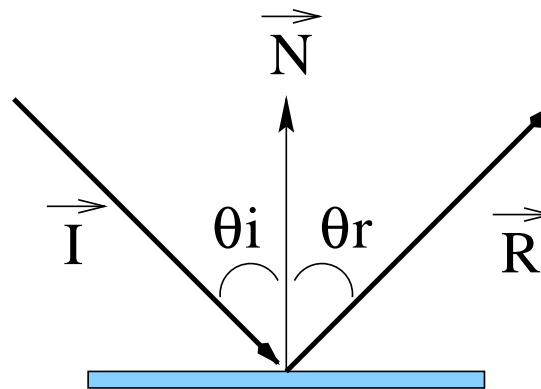
Comparaison

Entre les lissages de Gouraud et Phong

- ▷ interpolation par rapport à la couleur/normale
- ▷ polygones convexes dans les deux cas
- ▷ dépendance/indépendance couleur sur orientation des polygones
- ▷ Phong : meilleur rendu réaliste mais plus coûteux en temps de calcul

Optique Géométrique

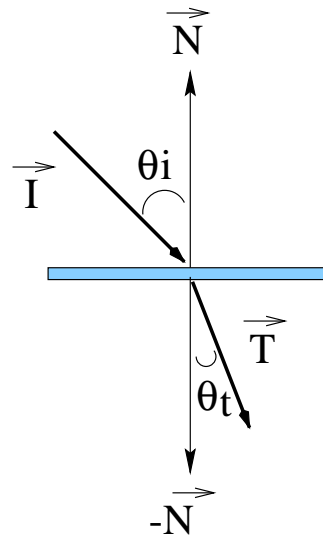
Lois de Snell–Descartes : Réflexion



- ▷ $\theta_i = \theta_r$
- ▷ $\vec{I}, \vec{R}, \vec{N}$: dans le même plan d'incidence

Optique Géométrique

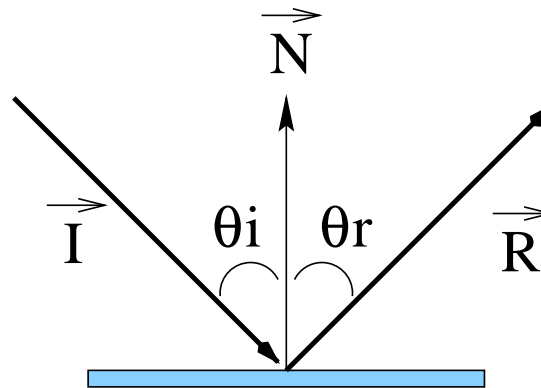
Lois de Snell–Descartes : Réfraction



▷ $n_i \sin \theta_i = n_t \sin \theta_t$

▷ $\vec{I}, \vec{T}, \vec{N}$: dans le même plan d'incidence

Réflexion Spéculaire



Hypothèse

▷ \vec{I} , \vec{N} sont connus : $\cos \theta_i = -\vec{I} \cdot \vec{N}$, $\cos \theta_r = \vec{N} \cdot \vec{R}$

Lois de la réflexion

1. $\theta_i = \theta_r$: $-\vec{I} \cdot \vec{N} = \vec{N} \cdot \vec{R}$

2. \vec{I} , \vec{R} , \vec{N} même plan d'incidence : $\vec{R} = \alpha \vec{I} + \beta \vec{N}$

Réflexion Spéculaire

On en déduit

$$\triangleright \vec{N} \cdot \vec{R} = \vec{N} \cdot (\alpha \vec{I} + \beta \vec{N})$$

$$\triangleright -\vec{I} \cdot \vec{N} = \vec{N} \cdot (\alpha \vec{I} + \beta \vec{N})$$

Conditions

$$\triangleright \vec{N} \text{ normalisé : } -\vec{I} \cdot \vec{N} = \alpha(\vec{N} \cdot \vec{I}) + \beta$$

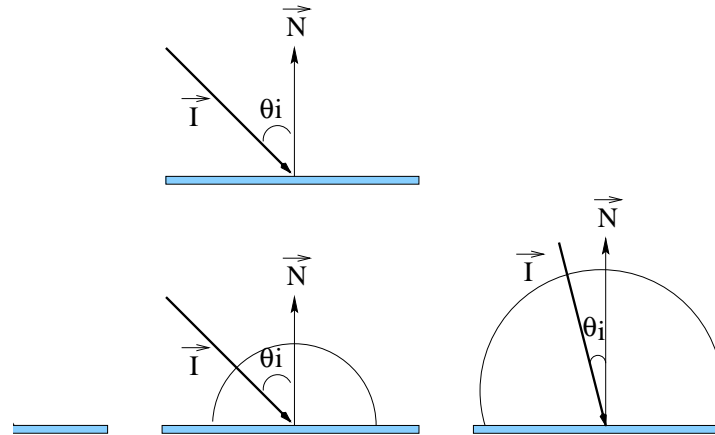
$$\triangleright \text{On choisit } \alpha = 1 : \beta = -2(\vec{I} \cdot \vec{N})$$

D'où

$$\boxed{\vec{R} = \vec{I} - 2(\vec{I} \cdot \vec{N})\vec{N}}$$

\vec{I}, \vec{N} connus, on trouve \vec{R} (C.Q.F.D.)

Réflexion Diffuse

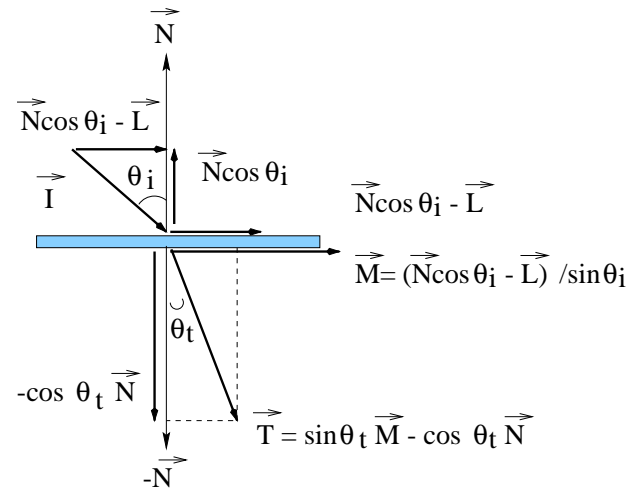


$$I_d = I_i \cos \theta$$

- ▷ \vec{I}, \vec{N} connus
- ▷ $k_{rd} \in [0, 1]$: connu
- ▷ $\vec{L} = -\vec{I}$

$$I_d = I_i k_{rd} (\vec{L} \cdot \vec{N})$$

Réfraction Spéculaire



Hypothèse

▷ \vec{I}, \vec{N} : connus

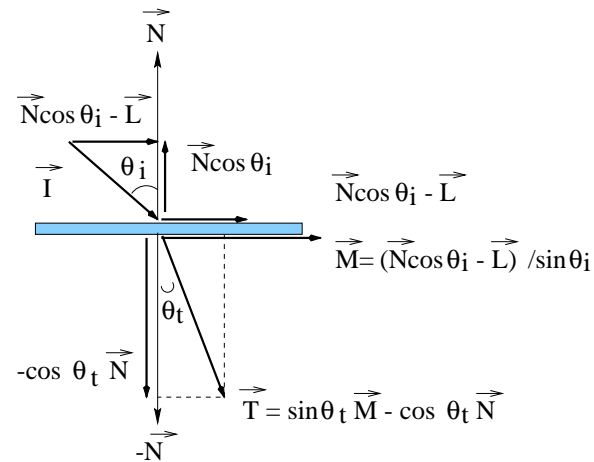
Lois de Snell–Descartes

1. $n_i \sin \theta_i = n_t \sin \theta_t$
2. $\vec{I}, \vec{T}, \vec{N}$: dans le même plan d'incidence

Réfraction Spéculaire

Première loi : $\eta_{it} = \frac{n_i}{n_t} = \frac{\sin \theta_t}{\sin \theta_i}$

Deuxième loi : $\vec{T} = \alpha \vec{I} + \beta \vec{N}$



$$\vec{T} = \sin \theta_t \vec{M} + (-\cos \theta_t \vec{N})$$

Réfraction Spéculaire

\vec{M} unitaire

$$\vec{M} = \frac{\cos \theta_i \vec{N} - \vec{L}}{\sin \theta_i}$$

$$\vec{T} = \sin \theta_t \left(\frac{\cos \theta_i \vec{N} - \vec{L}}{\sin \theta_i} \right) - \cos \theta_t \vec{N}$$

$$\vec{T} = (\eta_{it} \cos \theta_i - \cos \theta_t) \vec{N} - \eta_{it} \vec{L}$$

avec

$$\triangleright \cos \theta_t = \sqrt{1 - \sin^2 \theta_t} = \sqrt{1 - (\eta_{it} \sin \theta_i)^2}$$

$$\triangleright \cos \theta_t = \sqrt{1 - \eta_{it}^2 (1 - \cos^2 \theta_i)}$$

On en déduit

$$\vec{T} = [\eta_{it} (\vec{N} \cdot \vec{L}) - \sqrt{1 - \eta_{it}^2 (1 - (\vec{N} \cdot \vec{L})^2)}] \vec{N} - \eta_{it} \vec{L}$$

Réfraction Spéculaire

Deuxième loi : $\vec{T} = \alpha \vec{I} + \beta \vec{N}$

d'où

$$\triangleright \alpha = \eta_{it} (\vec{L} \cdot -\vec{I})$$

$$\triangleright \beta = \eta_{it} (\vec{N} \cdot \vec{L}) - \sqrt{1 - \eta_{it}^2 (1 - (\vec{N} \cdot \vec{L})^2)}$$

\vec{I}, \vec{N} connus, on trouve \vec{R} (C.Q.F.D.)

Conclusion

Pour calculer la direction de Réflexion / Réfraction il faut connaître

- ▷ les normales en tout point d'une surface
- ▷ les coefficients photométriques de surface
- ▷ appliquer les lois optique géométrique

$$\vec{R} = \vec{I} - 2(\vec{I} \cdot \vec{N})\vec{N}$$

$$\vec{T} = [\eta_{it}(\vec{N} \cdot \vec{L}) - \sqrt{1 - \eta_{it}^2(1 - (\vec{N} \cdot \vec{L})^2)}] \vec{N} - \eta_{it} \vec{L}$$

Calculs d'Intersections

Détermination du point d'intersection d'une droite (rayon) avec une surface quelconque

▷ objet implicite : $f(x, y, z) \leq 0$

◇ quadriques

◇ métaballs

◇ ...

▷ un arbre octal (intersection avec le cube courant)

▷ un objet transformé (appliquer la transformation au rayon)

Droite / Sphère

Soit

▷ $\vec{R}_0(X_0, Y_0, Z_0)$: Origine de droite

▷ $\vec{R}_d(X_d, Y_d, Z_d)$: Direction de droite

▷ \vec{R}_d normalisé: $X_d^2 + Y_d^2 + Z_d^2 = 1$

\vec{R}_d normalisé: $X_d^2 + Y_d^2 + Z_d^2 = 1$

On définit une droite paramétrique

$$\vec{R}(t) = \vec{R}_0 + t\vec{R}_d$$

avec $t \in [0, \infty]$

Droite / Sphère

Soit

▷ $\vec{S}_c(X_c, Y_c, Z_c)$: Centre de sphère

▷ $\vec{S}_r(X_r, Y_r, Z_r)$: Rayon de sphère

On définit les points, $\vec{S}(X, Y, Z)$, d'une sphère analytique

$$(X - X_c)^2 + (Y - Y_c)^2 + (Z - Z_c)^2 = S_r^2$$

et on obtient le système à résoudre

$$\begin{cases} (X - X_c)^2 + (Y - Y_c)^2 + (Z - Z_c)^2 = S_r^2 \\ X = X_0 + tX_d \\ Y = Y_0 + tY_d \\ Z = Z_0 + tZ_d \end{cases}$$

Droite / Sphère

définissant une équation du second degré en t

$$At^2 + Bt + C = 0$$

avec dans ce cas

$$A = X_d^2 + Y_d^2 + Z_d^2$$

$$B = 2(X_d(X_0 + X_c) + Y_d(Y_0 + Y_c) + Z_d(Z_0 + Z_c))$$

$$C = (X_0 - X_c)^2 + (Y_0 - Y_c)^2 + (Z_0 - Z_c)^2 - S_r^2$$

Deux solutions

- ▷ Pas de racines réelles : pas d'intersection
- ▷ Deux racines distinctes : intersection

Le cas tangent sera ignoré en Ray-Tracing

Droite / Sphère

Soit t_i , la valeur d'intersection la plus proche

Coordonnées du point d'intersection

$$X_i = X_0 + t_i X_d$$

$$Y_i = Y_0 + t_i Y_d$$

$$Z_i = Z_0 + t_i Z_d$$

Normale au point d'intersection $\vec{N}(X_n, Y_n, Z_n)$

$$X_n = \frac{X_i - X_c}{S_r}$$

$$Y_n = \frac{Y_i - Y_c}{S_r}$$

$$Z_n = \frac{Z_i - Z_c}{S_r}$$

Droite / Sphère

Pour un calcul d'intersection on doit donc

1. calculer les coefficients A, B, C
2. résoudre l'équation $At^2 + Bt + C = 0$
3. déterminer les deux racines t_1, t_2 si elles existent
4. retenir l'intersection la plus proche
5. calculer la normale \vec{N} en ce point

Ce principe s'applique aux quadriques plus générales

Droite / Quadriques

Equation de surface

$$F(X, Y, Z) = [X \ Y \ Z \ 1] \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = 0$$

en considérant

- ▷ $[Q]$: matrice $[4,4]$ avec $A_{ij} = A_{ji}$ ($i \neq j$)
- ▷ $[P]$: vecteur homogène

Droite / Quadriques

En substituant l'équation paramétrique de droite

$$X = X_0 + tX_d$$

$$Y = Y_0 + tY_d$$

$$Z = Z_0 + tZ_d$$

avec l'équation précédente

$$F(X, Y, Z) = [P]^T \cdot [Q] \cdot [P] = 0$$

On obtient les coefficients

$$A_q = [R_d]^T \cdot [Q] \cdot [R_d]$$

$$B_q = 2[R_d]^T \cdot [Q] \cdot [R_0]$$

$$C_q = [R_0]^T \cdot [Q] \cdot [R_0]$$

Droite / Quadriques

d'où le résultat suivant pour les trois coefficients de l'équation du second degré

$$A_q = A_{11}X_d^2 + 2A_{12}X_dY_d + 2A_{13}X_dZ_d + A_{22}Y_d^2 + 2A_{23}Y_dZ_d + A_{33}Z_d^2$$

$$B_q = 2(A_{11}X_0X_d + A_{12}(X_0Y_d + X_dY_0) + A_{13}(X_0Z_d + X_dZ_0) + A_{14}X_d + A_{22}Y_0Y_d + A_{23}(Y_0Z_d + Y_dZ_0) + A_{24}Y_d + A_{33}Z_0Z_d + A_{34}Z_d)$$

$$C_q = A_{11}X_0^2 + 2A_{12}X_0Y_0 + 2A_{13}X_0Z_0 + 2A_{14}X_0 + A_{22}Y_0^2 + A_{23}Y_0Z_0 + 2A_{24}Y_0 + A_{33}Z_0^2 + 2A_{34}Z_0 + A_{44}$$

On peut donc calculer : $A_q t^2 + B_q t + C_q = 0$

Droite / Quadriques

Si intersection, on en déduit : $\vec{P}_i(X_i, Y_i, Z_i)$

$$X_i = X_0 + t_i X_d, \quad Y_i = Y_0 + t_i Y_d, \quad Z_i = Z_0 + t_i Z_d$$

et la normale à la quadrique : $\vec{N}(X_n, Y_n, Z_n)$

$$X_n = \frac{\partial F}{\partial X} = 2(A_{11}X_i + A_{12}Y_i + A_{13}Z_i + A_{14})$$

$$Y_n = \frac{\partial F}{\partial Y} = 2(A_{12}X_i + A_{22}Y_i + A_{23}Z_i + A_{24})$$

$$Z_n = \frac{\partial F}{\partial Z} = 2(A_{13}X_i + A_{23}Y_i + A_{33}Z_i + A_{34})$$

Conclusion

Calculs d'intersection possible droite / surface

▷ avec un objet implicite : $f(x, y, z) \leq 0$

◇ pour les quadriques simples

◇ métaballs

▷ surfaces de révolution et d'extrusion

▷ arbre octal (intersection avec le cube courant)

▷ objet transformé (appliquer la transformation au rayon)

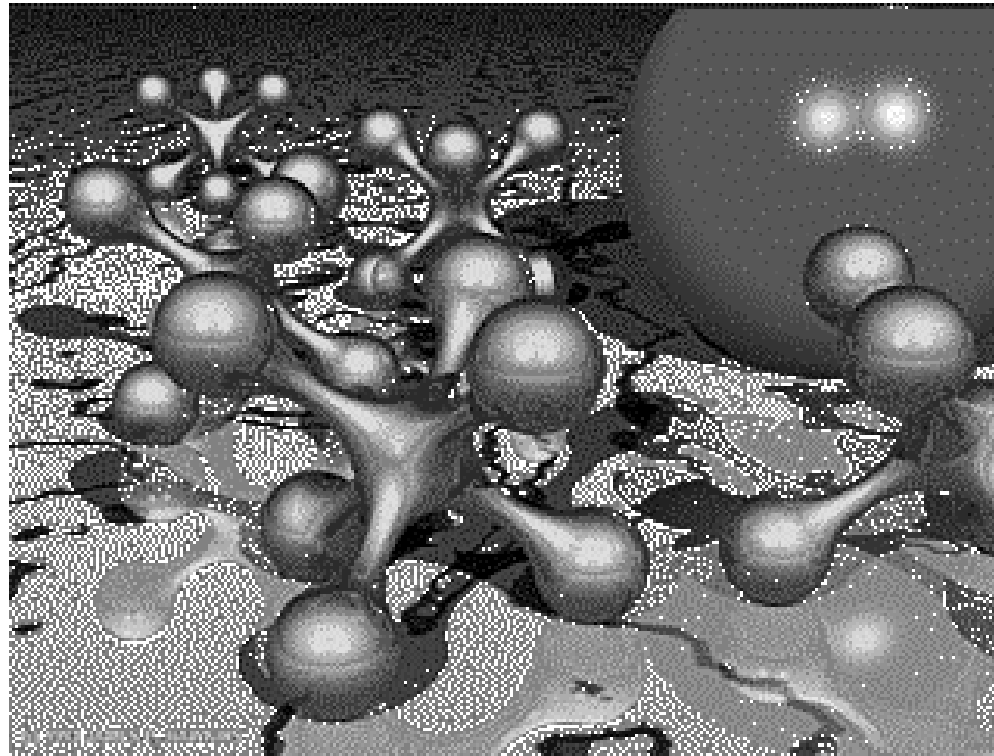
calculs d'intersection plus complexes

▷ courbe / surface

▷ surface / surface

▷ ...

Conclusion

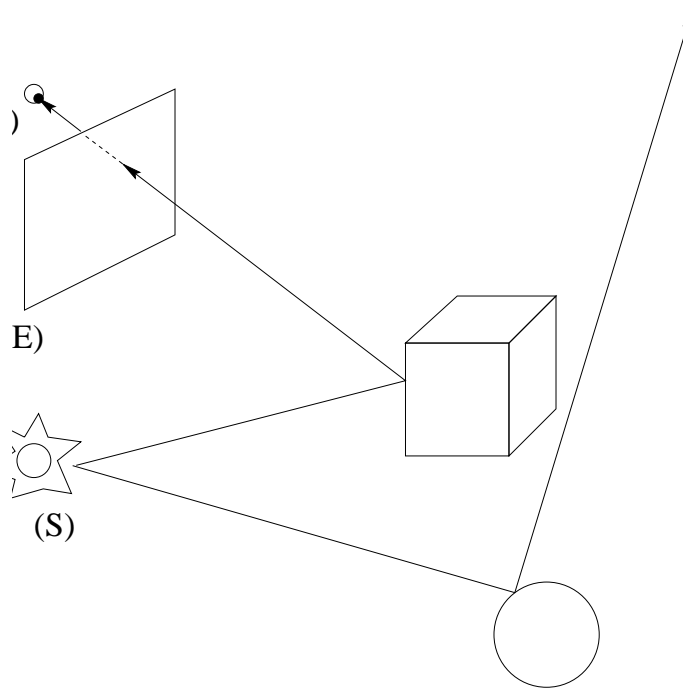


Courtesy of Jim Hawkins:

<http://www.exit109.com/~jimh>

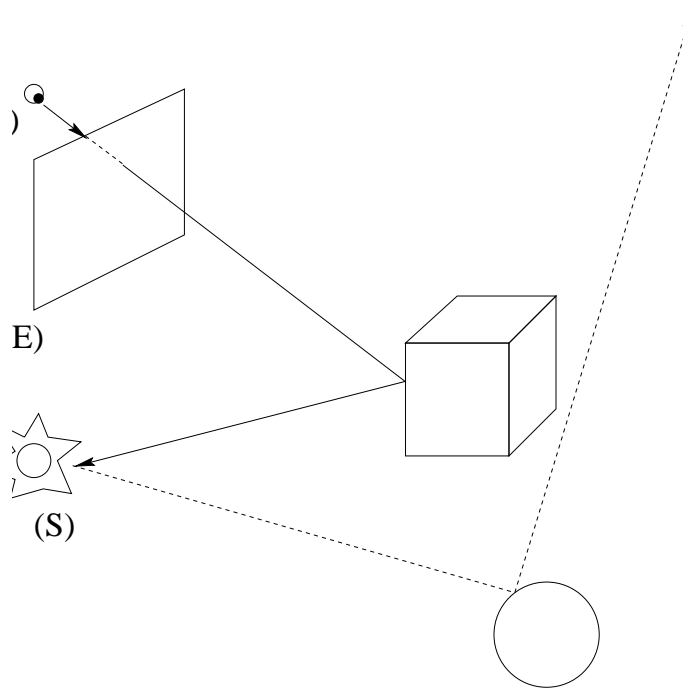
Ray-Tracing

Suivi des rayons lumineux éclairant une scène : de la Source à l'Observateur



Ray-Tracing

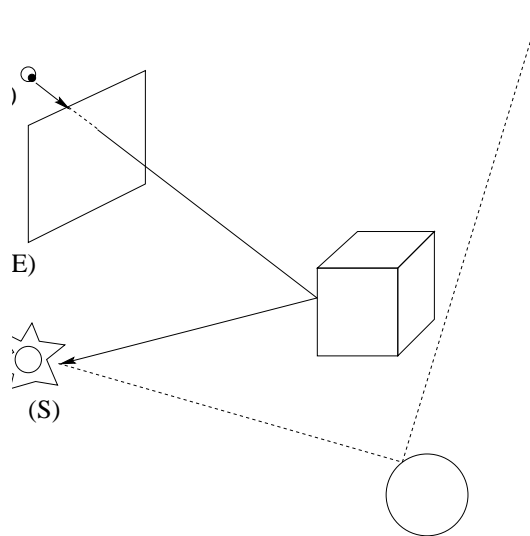
Réciprocité des chemins optiques : de l'Observateur à la Source



Ray-Tracing

Pour chaque rayon (droite œil / pixel)

- ▷ Calcul d'intersection (rayon / scène)
- ▷ Calcul réflexion / réfraction
- ▷ Calcul d'éclairage au point d'intersection le plus proche de l'écran

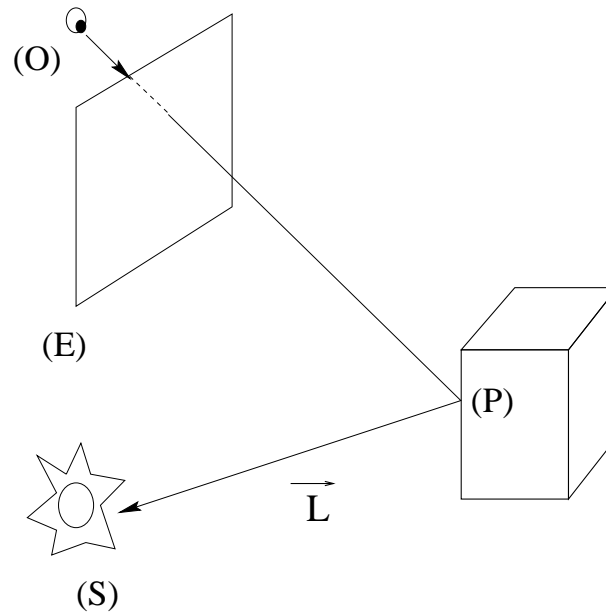


Eclairage de Source

Calcul d'éclairage / Ombrage : **Shadow Feeler**

Hypothèse

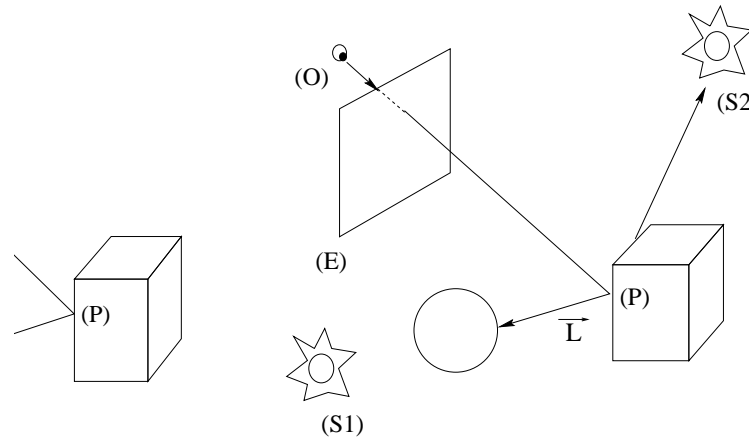
- ▷ On connaît la position de la Source
- ▷ On détermine le rayon (P / Source)



Eclairage de Source

Principe du Shadow Feeler

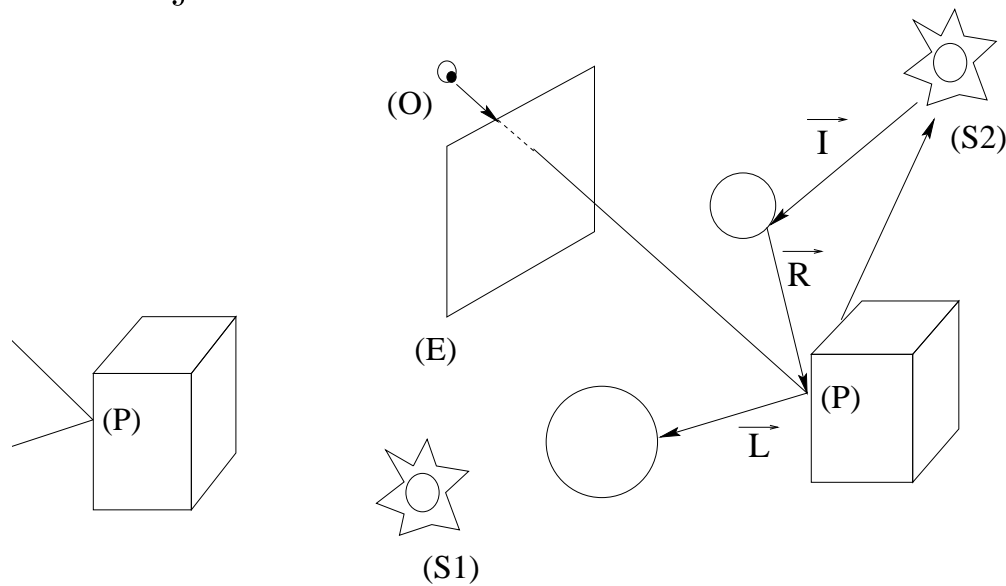
- ▷ pour chaque rayon (P / Source)
- ▷ Calcul d'intersection (rayon / scène)
- ▷ Si intersection avec la scène : ombre au point P (shadow Feeler)
- ▷ Sinon calcul d'intensité lumineuse en P du à la source



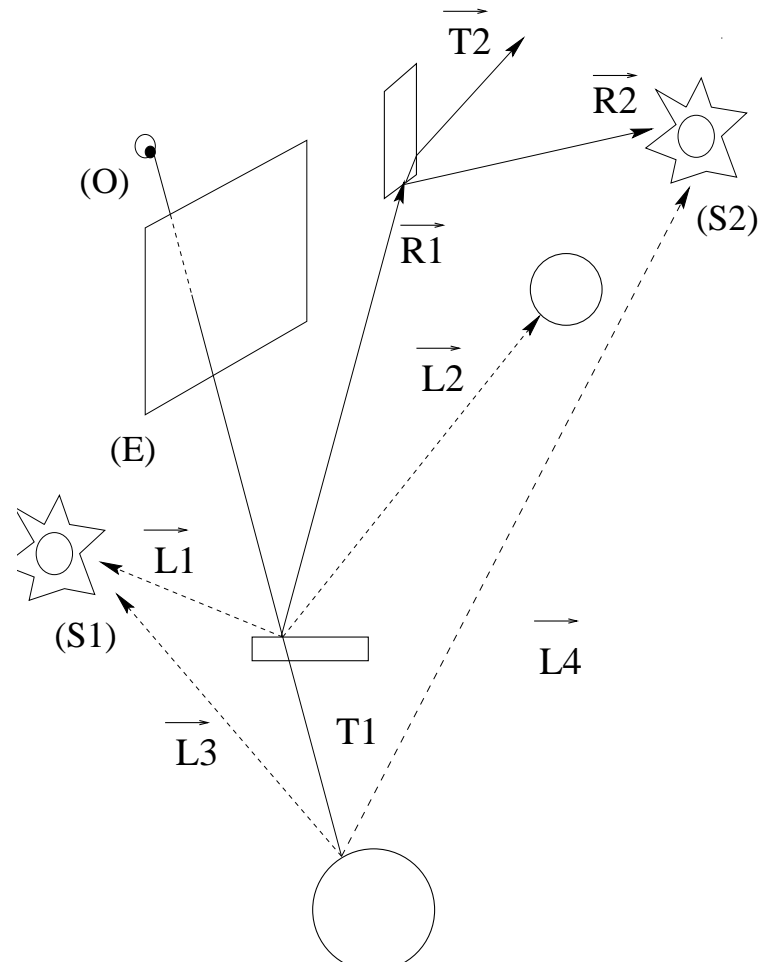
Eclairage entre objets

Au point d'intersection le plus proche de l'écran

- ▷ Calcul d'éclairage du aux sources
- ▷ Calcul d'éclairage des objets entre eux

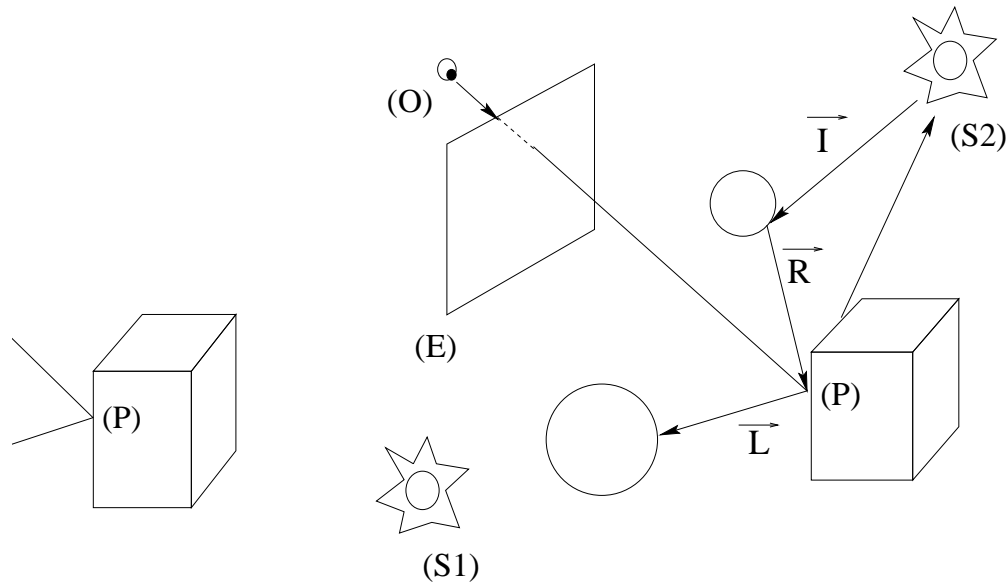


Réflexion / Réfraction



Réflexion / Réfraction

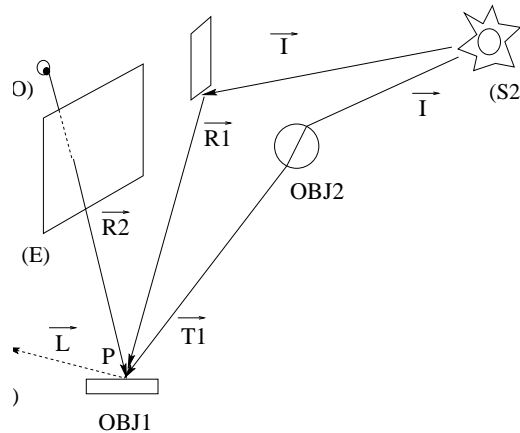
- ▷ Ray-Casting: Lancer de Rayons
 - ◇ intersection œil/pixel et shadow-feeler
- ▷ Ray-Tracing: Suivi de Rayons
 - ◇ Lancer de Rayons + réflexion/transmission



Réflexion / Réfraction

Principe identique pour réflexion (\vec{R}) et réfraction (\vec{T})

1. Calcul du rayon \vec{R} (resp. \vec{T}) au point P de la surface
 - ▷ connaître la normale \vec{N}
 - ▷ appliquer les lois de l'optique géométrique
2. \vec{R} devient le rayon incident \vec{I}
3. Récursivité du traitement



Réflexion / Réfraction

Connaître les propriétés des surfaces

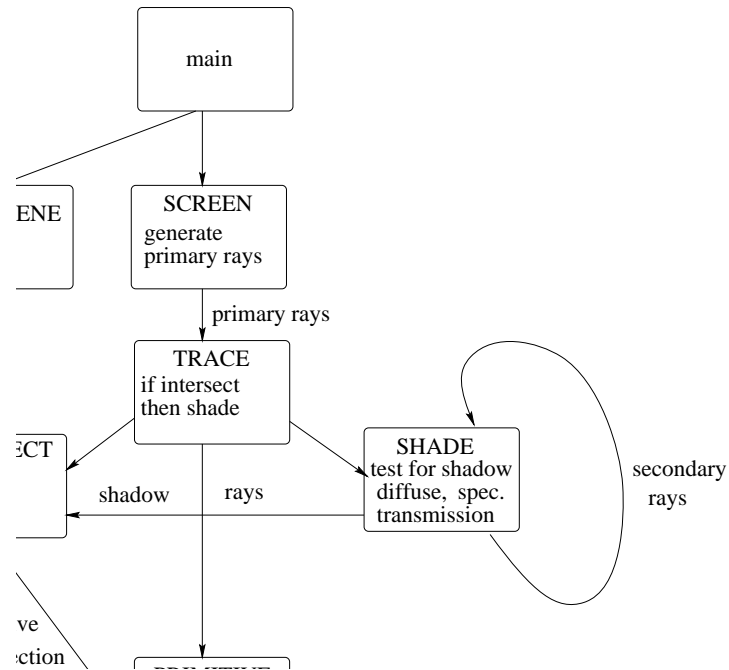
- ▷ Géométriques : Normales aux surfaces
- ▷ Photométriques :
 - ◇ coefficient de réflexion (k_r)
 - ◇ coefficient de transmission (k_t)
 - ◇ coefficient d'absorption (k_{abs})

Ces coefficients se définissent sur un intervalle : $[0, 1]$

Problème du Ray-tracing: mais jusqu' où s'arrêteront-ils ?

- ▷ On sort de la scène
- ▷ On limite la récursivité
- ▷ On dépasse un seuil de perception

Approche d'Implémentation



Approche d'Implémentation

Module **Intersection**

```
int intersect(Ray ray , CSGTree* scene , PointList* inter) {

    if (scene->composite) {                                /* recursivite CSG */
        nb_g = intersect( ray, scene->fg, inter_g);
        if (nb_g == 0 && scene->opbool != "UNION")
            return 0;
        else {
            nb_d = intersect( ray , scene->fd , inter_d); /* recursivite CSG */
            return composite_intersect( inter_g, nb_g,
                                       inter_d, nb_d,
                                       scene->opbool );
        }
    }
    else return prim_intersect(ray, scene->primitive,inter);
}
```

Approche d'Implémentation

Module **Trace**

```
raytrace(int recurs, float threshold, Ray ray, Pixel color) {  
  
    if (intersect(ray, scene, inter) ) {  
        point = find_closest(inter);  
        normal = surf_norm(Surf *surf , point);  
        shadow( recurs, threshold, ray,  
              point, normal, inter, color);  
    }  
    else background ( ray, color);  
}
```

Approche d'Implémentation

Module **shade**

```
shadow( int recurs, float threshold, Ray ray, Point point, Vect normal, Pixel color) {
  for(source=0;sources<=nb_sources;sources++) { /* Shadow Feeler */
    light = compute_ray_source(point);
    if ( intersect(light,scene,inter)==0 || first_intersect(inter)>lightpos )
      color += dotproduct(normal,light)*colorlight;
  } /* Fin Shadow Feeler */
  if (recurs + 1 < recurs_max) { /* recursivite Reflexion / Transmission */
    kr=surf_refl(surf , point);
    if (seuil*kr > seuil_min) { /* seuil de perception */
      refl=reflexion(point , normal);
      raytrace(recurs + 1, seuil*kr , refl , color);
    }
    kt=surf_refr(surf , point);
    if (seuil*kt > threshold_min) { /* seuil de perception */
      refr=refraction(point , normal);
      raytrace(recurs + 1, threshold*kt , refr , color);
    }
  } /* Fin recursivite Reflexion - Transmission */
}
```

Approche d'Implémentation

Ray Tracing

- ▷ Arbre C.S.G.
- ▷ Arbre Réflexion / transmission
- ▷ calcul d'intersections avec C.S.G.
- ▷ calcul Réflexion / transmission
- ▷ calcul d'intensité lumineuse

Implémentation des modules

- ▷ **Intersection** : Récursivité C.S.G.
- ▷ **Suivi de rayons** : Récursivité Réflexion / Transmission
- ▷ **Eclairement** : calcul d'intensité (couleur Pixel)

Introduction

Radiosité d'une surface

- ▷ la **quantité** de lumière (énergie) quittant cette surface
- ▷ due à la **quantité** de lumière **émise**
- ▷ et **réfléchi**e par cette surface

$$B_i = E_i + \rho_i \sum_{j=1}^n F_{ij} B_j$$

- ▷ E_i : Energie lumineuse émise par la surface i
- ▷ ρ_i : coefficient de réflexion de la surface i
- ▷ F_{ij} : **Facteur de forme**, entité géométrique définissant la fraction de lumière quittant i et atteignant la surface j

Forme matricielle

Pour un ensemble de n surfaces

$$\begin{bmatrix} 1 - \rho_1 F_{11} & \cdots & -\rho_1 F_{1i} & \cdots & -\rho_1 F_{1n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 - \rho_i F_{i1} & \cdots & 1 - \rho_i F_{ii} & \cdots & -\rho_i F_{in} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 - \rho_n F_{n1} & \cdots & -\rho_n F_{ni} & \cdots & 1 - \rho_n F_{nn} \end{bmatrix} \begin{bmatrix} B_1 \\ \cdots \\ B_i \\ \cdots \\ B_n \end{bmatrix} = \begin{bmatrix} E_1 \\ \cdots \\ E_i \\ \cdots \\ E_n \end{bmatrix}$$

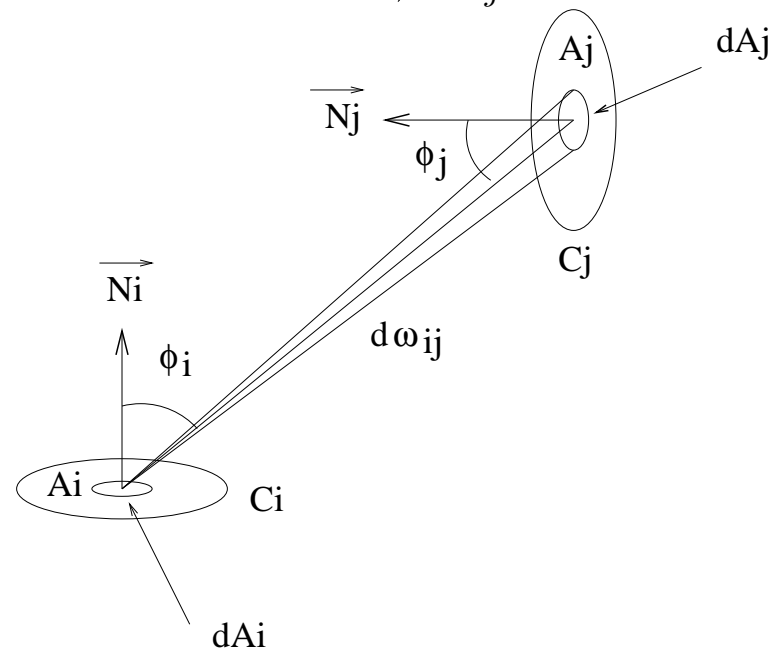
Remarques

- ▷ E_i : non-nul si surface rayonnant de l'énergie
- ▷ E_i, ρ_i : dépendent de la longueur d'onde lumineuse utilisée
- ▷ $F_{ii} = 0$: pour les surfaces planes ou convexe

Problème : calcul de Facteur de Forme (F_{ij})

Angle solide

Entre deux surfaces élémentaires : dA_i, dA_j



$$\text{Angle solide : } d\omega_{ij} = \cos \phi_j \frac{dA_j}{r^2}$$

Surface élémentaire

Soit dE_i l'énergie élémentaire quittant la surface élémentaire dA_i et atteignant la surface élémentaire dA_j

on trouve alors l'énergie rayonnée par la surface élémentaire

$$dE_i dA_i = (I_i \cos \phi_i d\omega_{ij}) dA_i$$

$$dE_i dA_i = (I_i \cos \phi_i \cos \phi_j \frac{dA_j}{r^2}) dA_i$$

et donc

$$dE_i dA_i = (I_i \cos \phi_i \cos \phi_j) \frac{dA_i dA_j}{r^2}$$

Surface Totale

Expression du facteur de forme correspondant

$$F_{dA_i dA_j} = \frac{\text{Energie rayonnee par } dA_i \text{ atteignant } dA_j}{\text{Energie totale quittant } dA_i \text{ en toute direction}}$$

Sur une surface lambertienne idéale $E_i = I_i \pi$

$$F_{dA_i dA_j} = \left(I_i \cos \phi_i \cos \phi_j \frac{dA_i dA_j}{r^2} \right) (I_i \pi dA_i)^{-1}$$

$$F_{dA_i dA_j} = \frac{\cos \phi_i \cos \phi_j dA_j}{\pi r^2}$$

Intégration du facteur de forme élémentaire sur la surface totale A_j

$$F_{dA_i A_j} = \int_{A_j} \frac{\cos \phi_i \cos \phi_j dA_j}{\pi r^2}$$

Surface Totale

Facteur de forme d'échange d'énergie entre les deux surfaces A_i, A_j

$$F_{A_i A_j} = \frac{1}{A_i} \int_{A_i} \left(\int_{A_j} \frac{\cos \phi_i \cos \phi_j}{\pi r^2} dA_j \right) dA_i$$

Ce facteur de forme est calculé en moyenne ($\frac{1}{A_i}$)

▷ réciprocité des échanges thermiques : $A_i F_{ij} = A_j F_{ji}$

▷ normalisation des facteurs de forme : $\sum_{k=1}^n F_{ik} = 1, i \in [1, n]$

▷ surfaces planes, convexes : $F_{ii} = 0$

Calcul

Problème: calcul de $F_{A_i A_j} = \frac{1}{A_i} \int_{A_i} \left(\int_{A_j} \frac{\cos \phi_i \cos \phi_j}{\pi r^2} dA_j \right) dA_i$

Restrictions

1. distance (r) très grande entre les 2 surfaces
2. projection sur une hémisphère \equiv projection sur un héli-cube

Signification

1. intégration sur A_j ne varie pas sur A_i
2. calcul basé sur la notion d'angle solide

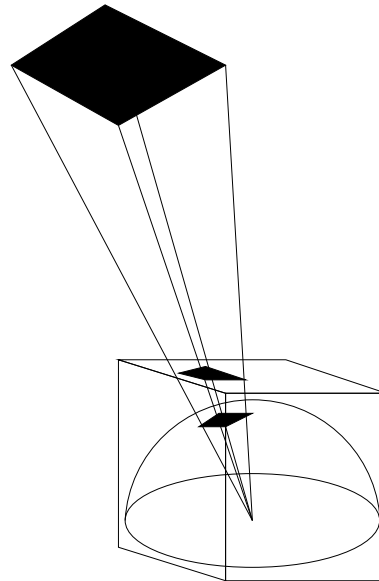
Première approximation : $F_{ij} \approx \frac{A_i}{A_i} \left(\int_{A_j} \frac{\cos \phi_i \cos \phi_j}{\pi r^2} dA_j \right)$

D'où : $F_{ij} = \int_{A_j} \frac{\cos \phi_i \cos \phi_j}{\pi r^2} dA_j$

Hémi-cube

On assimile une hémisphère à un hémi-cube

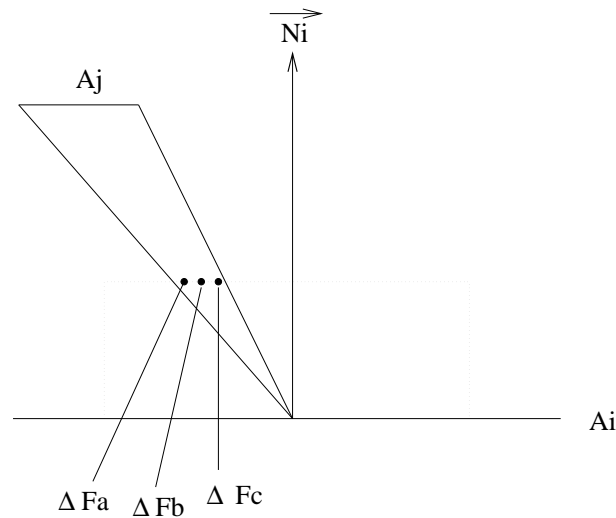
- ▷ construit autour du centre de chaque surface
- ▷ axe Oz orienté suivant la normale au centre
- ▷ subdivision de chaque face de l'hémi-cube en éléments appelés "pixels"



Hémi-cube

Chaque surface (A_j) peut alors être projetée sur cet hémi-cube

Chaque pixel de l'hémi-cube sera considéré comme une surface élémentaire définissant un facteur de forme élémentaire ΔF_q



$$F_{ij} = \Delta F_a + \Delta F_b + \Delta F_c$$

Avantages

Facteur de forme d'un "pixel"

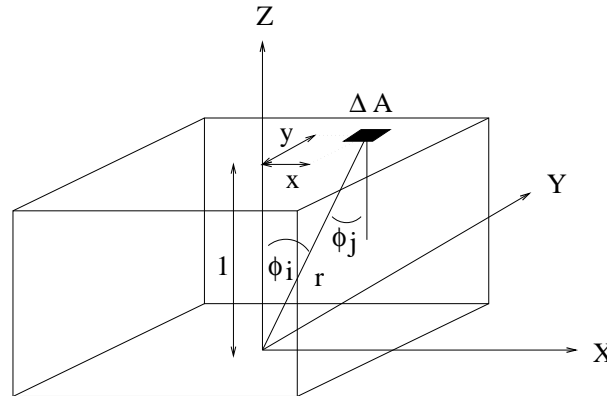
$$\Delta F_{ij} = \Delta F_q = \frac{\cos \phi_i \cos \phi_j}{\pi r^2} \Delta A$$

où ΔA : surface d'un pixel

- ▷ prédétermination d'un facteur de forme d'un "pixel", donc d'un hémi-cube
- ▷ facteur de forme d'une surface A_i défini par la somme des pixels de l'hémi-cube couvert par la projection de A_j

$$F_{ij} = \sum_q \Delta F_q$$

Calcul



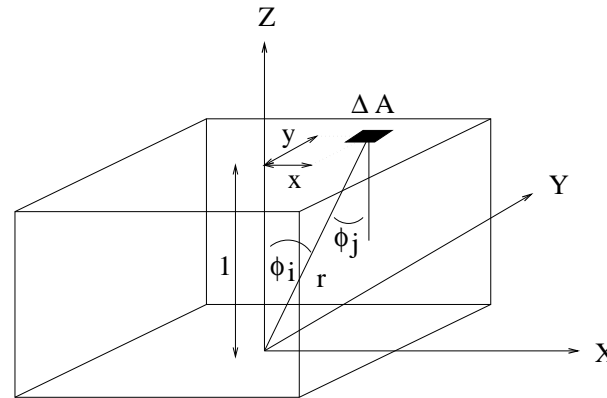
$$\triangleright r = (x^2 + y^2 + 1)^{\frac{1}{2}}$$

$$\triangleright \cos \phi_i = \cos \phi_j = \frac{1}{(x^2 + y^2 + 1)^{\frac{1}{2}}}$$

$$\Delta F_q = \frac{\cos \phi_i \cos \phi_j}{\pi r^2} \Delta A$$

$$\boxed{\Delta F_q = \frac{1}{\pi(x^2 + y^2 + 1)^2} \Delta A}$$

Calcul



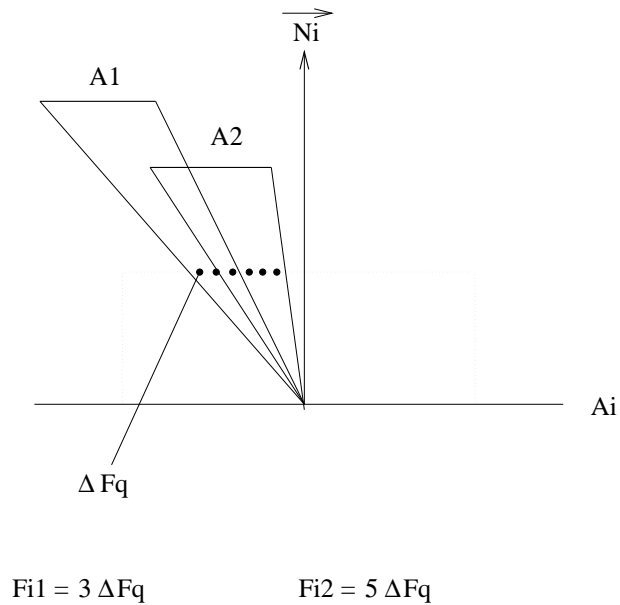
Pour un pixel positionné sur un côté de l'hémi-cube

$$\Delta F_q = \frac{z}{\pi(y^2 + z^2 + 1)^2} \Delta A$$

$$\Delta F_q = \frac{z}{\pi(x^2 + z^2 + 1)^2} \Delta A$$

Conclusion

Problème : Projection de deux surfaces recouvrant les mêmes pixels



On retiendra ceux de la surface la plus proche

Conclusion

Inconvénient de la méthode

- ▷ échantillonnage du à la subdivision de l'hémi-cube : aliasing
- ▷ erreurs de calculs sur les facteurs de formes rapprochés
- ▷ utilisation hybride Ray-Tracing / Radiosité

Bibliographie

Les bonnes adresses

- ▷ <http://www.swin.edu.au/astronomy/pbourke>: de la théorie à la pratique
- ▷ <http://www.linux3D.org> : de nombreux outils
- ▷ <http://www.graphics.cornell.edu/online/research> : de belles images
- ▷ <ftp://ftp-graphics.stanford.edu/pub/Graphics> : modeleurs, ray-traceurs, ...
- ▷ <http://www.povray.org> : ray-tracing
- ▷ <ftp://ftp.gwu.edu/pub/graphics/BMRT> : RenderMan compliant ray-tracer/radiosity
- ▷ <http://www.acm.org/jgt> : Journal of Graphics Tools
- ▷ <http://www.springer.de/comp/index.html> : bibliographie informatique
- ▷ <ftp://ftp-graphics.stanford.edu/pub/Graphics/Papers/GraphicsFTP.txt> : des liens
- ▷ <http://www.education.siggraph.org/materials/hypgraph/toc.htm> : sur tous les sujets

Rendu Réaliste

Les ouvrages

Foley / Van Dam / Feiner / Hughes

“Computer Graphics, principles & practice”,
Addison-Wesley 1990

A. S. Glassner

“An Introduction To Ray-Tracing”,
Academic Press 1991

N. Wilt

“Object Oriented Ray-Tracing in C++”,
John Wiley & Sons Inc. , Professional Computing 1994