7.5 Effets de transparence

Algorithmes de parties cachées et éclairage : surfaces opaques

- tout ou rien : visible ou invisible
- si visible : modèle éclairage à partir d'une seule surface

Prise en compte des objets qui transmettent la lumière (vitres, eau, verre…)

Quelques observations:

- un bâton plongé dans l'eau apparaît brisé : effet de réfraction
- un effet spéculaire important peut rendre inutile l'effet de transparence
- Il existe certains matériaux à travers lesquels les objets observés ne présentent pas de distorsions

1

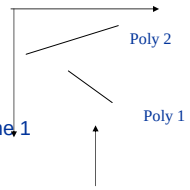
Effets de transparence

7.5.1 Transparence non réfractive

La réfraction est ignorée

Calcul de l'intensité en un point : basée sur interpolation linéaire (Newell Newell Sancha)

$$I = (1-k_1)I_1 + k_1 I_2$$



I_1 est l'intensité de la surface du polygone 1

I_2 est l'intensité de la surface du polygone 2

k_1 est la mesure de transparence du polygone 1

(indice ou facteur de transparence)

si $k_1 = 0$ polygone 1 est opaque : polygone 2 invisible

si $k_1 = 1$ polygone 1 est transparent : polygone 2 visible

2

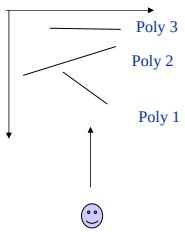
Effets de transparence

Modèle récursif (polygone 2 transparent)

si $k_1 = 1$ polygone 1 est transparent : polygone 2 visible

si $k_2 = 0$ polygone 2 est opaque : polygone 3 invisible

si $k_2 = 1$ polygone 2 est transparent : polygone 3 visible



3

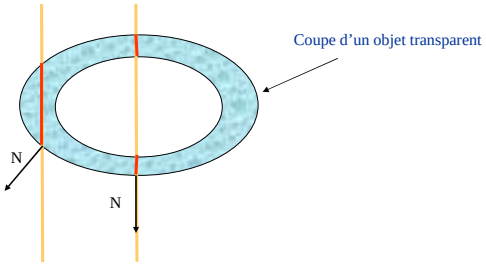
Effets de transparence

Modèle non linéaire (Kay) :

modèle linéaire peu réaliste dans le cas de surfaces gauches

prise en compte de l'épaisseur de la matière (transparence atténuée)

calcul du coefficient de transparence en fonction de la normale



4

Effets de transparence

$$k_1 = k_{min} + (k_{max} - k_{min})\,(1 - (1 - |n_z|)^p)$$

k_{min} valeur minimale de la transparence

k_{max} valeur maximale de la transparence

n_z composante en z de la normale unité à la surface

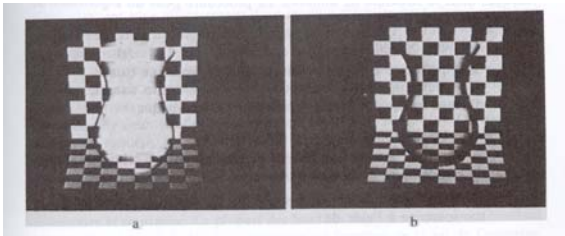
p exposant caractéristique de la transparence

calcul de l'intensité : interpolation linéaire $I = (1-k_1)I_1 + k_1 I_2$

5

Effets de transparence

Exemple: *(Cornell University. D.S. Kay)*



Les effets de transparence peuvent être incorporés dans tous les algorithmes de parties cachées (sauf z-buffer)

6

Effets de transparence

7.5.2 Transparence et z-buffer

2 tampons séparés : un pour les objets opaques et un pour les objets transparents.

Premier passage :

Pour tout polygone :

si transparent : le stocker dans une liste

si opaque : appliquer l'algorithme du z-buffer

7

Effets de transparence

Second passage

Pour chaque polygone de la liste des polygones transparents :

Regarder son altitude z

La comparer à z-buffer

si z > z-buffer, alors

ajouter son facteur de transparence k_c dans le tampon:

$$k_{new} = f\,(k_{old}\,,\,k_c)$$

calculer l'intensité $I_{new} = k_{old} I_{old} + k_c I_c$

Combiner tampons d'image opaque et intensité de transparence

8

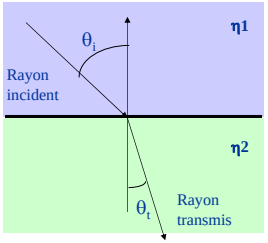
Effets de transparence

7.5.3 Transparence réfractive

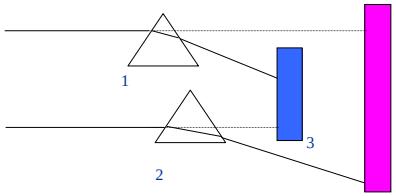
Géométrie de la réfraction

$\eta 1 \sin \theta_i = \eta 2 \sin \theta_t$

$\eta 1$ $\eta 2$ indices de réfraction



Effets de la réfraction



9

Effets de transparence

Calcul du vecteur T (rayon transmis)

Soit M le vecteur unitaire perpendiculaire à N la normale dans le plan formé par le rayon incident L (-I) et la normale N, on a

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

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

$$\eta_r = \frac{\eta_1}{\eta_2}$$

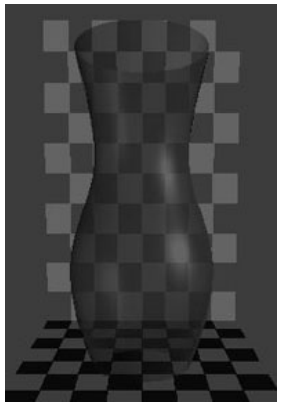
$$\vec{T} = (\eta_r \cos\theta_i - \cos\theta_t)\vec{N} - \eta_r \vec{L}$$

$$\vec{T} = \left[\eta_r(\vec{N}.\vec{L}) - \sqrt{1-\eta_r^2(1-(\vec{N}.\vec{L})^2)}\right]\vec{N} - \eta_r \vec{L}$$

Modèle utilisé dans l'algorithme de suivi de rayon (ray tracing)

10

Effets de transparence



11

Transparence sous OpenGL

- Modèle linéaire : définition d'un coefficient de transparence (mode RGBA)
  - *a=1* opaque
  - *a= 0* transparence

  ex : *glColor4f(0.0, 1.0, 1.0, 0.75);*

- Facteurs de transparence : C_{src} et C_{dest} *(4 composantes)*

- Mode transparence : activation et définition de fonctions de combinaison

glEnable (GL_BLEND);
glBlendFunc (GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);

ex : $D = C_{src}\, S + C_{dest}\, D$

glBlendEquation (GL_FUNC_ADD));

12

Facteur de blenging (Source et Destination)

Constante	Facteur de blending calculé
GL_ZERO	(0, 0, 0, 0)
GL_ONE	(1, 1, 1, 1)
GL_DST_COLOR	(R_d, G_d, B_d, A_d)
GL_SRC_COLOR	(R_s, G_s, B_s, A_s)
GL_ONE_MINUS_DST_COLOR	$(1, 1, 1, 1) - (R_d, G_d, B_d, A_d)$
GL_ONE_MINUS_SRC_COLOR	$(1, 1, 1, 1) - (R_s, G_s, B_s, A_s)$
GL_SRC_ALPHA	(A_s, A_s, A_s, A_s)
GL_ONE_MINUS_SRC_ALPHA	$(1, 1, 1, 1) - (A_s, A_s, A_s, A_s)$
GL_DST_ALPHA	(A_d, A_d, A_d, A_d)
GL_ONE_MINUS_DST_ALPHA	$(1, 1, 1, 1) - (A_d, A_d, A_d, A_d)$
GL_SRC_ALPHA_SATURATE	$(f, f, f, 1); f = \min(A_s, 1-A_d)$
GL_CONSTANT_COLOR	(R_c, G_c, B_c, A_c)
GL_ONE_MINUS_CONSTANT_COLOR	$(1, 1, 1, 1) - (R_c, G_c, B_c, A_c)$
GL_CONST_ALPHA	(A_c, A_c, A_c, A_c)
GL_ONE_MINUS_CONSTANT_ALPHA	$(1, 1, 1, 1) - (A_c, A_c, A_c, A_c)$

13

Transparence sous OpenGL

```
glColor4f(1.0, 1.0, 0.0, 0.75); /*triangle jaune*/
glColor4f(0.0, 1.0, 1.0, 0.75); /*triangle bleu*/
```

bleu d'abord

jaune d'abord

```
glBlendFunc (GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
```

14

Transparence sous OpenGL

Par défault

- source : GL_ONE et dest : GL_ZERO.
- Equivaut à pas de transparence.

La diversité des facteurs de transparence est utilisée pour

- Implémenter des filtres
- Réaliser l'antialiasing
- L'intégration de textures transparentes
- ….

15

7.6 Calcul des ombres

Contribue au réalisme d'une scène
Important en simulation (architecture, aéronautique…)
Calculs importants

Calcul des ombres
ombres propres
ombres portées

Les ombres dépendent des sources lumineuses

Attention au termes :
ombres ≠ ombrage
shadows ≠ shading

16

Ombres

Les ombres propres sont intégrées dans le modèle de Phong

Les ombres portées sont assimilables à un calcul de visibilité

- il suffit de déterminer quelles sont les surfaces cachées du point de vue de la lumière
- Le calcul est indépendant de la position de l'observateur.

On peut intégrer les ombres portées dans le modèle de Phong:

$$I = I_a \,.\, k_a + \Sigma_j S_j f_{att\,j} \; I_{p\,j} \, [\, k_d \, (N.L_j) + k_s \, (R_j.V)^n \,]$$

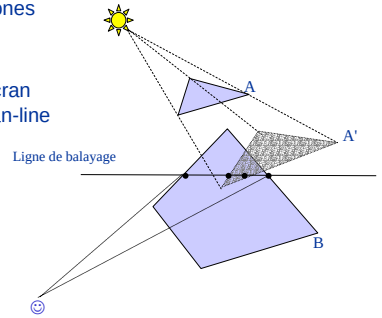
$S_j = 0$ si la lumière n'atteint pas ce point
$S_j = 1$ si la lumière atteint ce point

17

Ombres

algorithme scan-line pour les ombres

- projection sur les polygones
- création de polygones d'ombre
- traités dans le repère écran avec l'algorithme du scan-line



18

Ombres

Algorithme de calcul d'ombres en 2 passages

- basé sur l'algorithme de Weiler Atherton (scène)

  nécessite le découpage de polygone
  gestion des polygones dans la base de données

- basé sur le z-buffer (image)

  nécessite de la mémoire
  doit être optimisé (calcul de l'ombre pour un pixel non visible)
  beaucoup de transformations

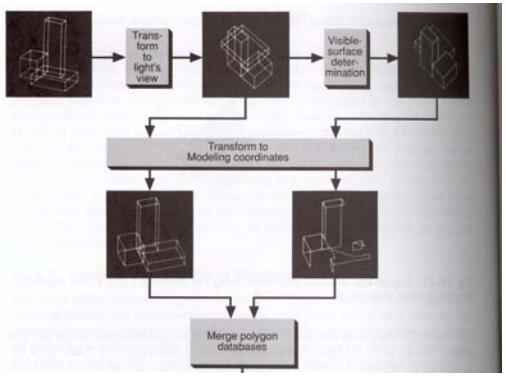
19

Ombres

Dans le repère scène (object-precision)

(Cornell University Atherton, Weiler, Greenberg)

1

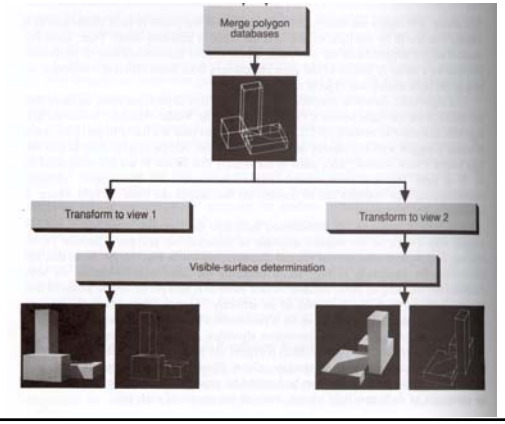


20

Ombres

Suite *(Cornell University Atherton, Weiler, Greenberg)*

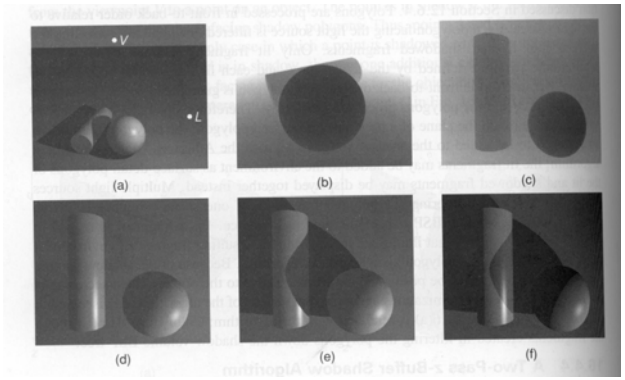
2



21

Ombres

Dans le repère image (image-precision) *(Columbia University)*

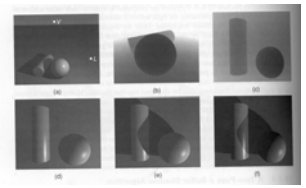


22

Ombres

Dans le repère image (image-precision)

a) vue générale
b) z-buffer éclairage
c) z-buffer observateur

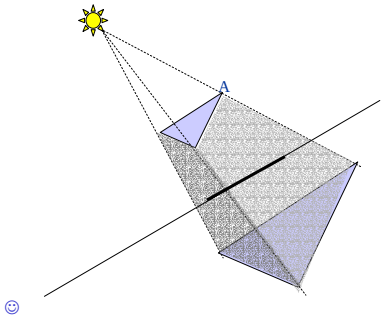


d) image sans ombre
e) image avec ombre
f) image avec ombre après calcul de l'illumination

23

Ombres

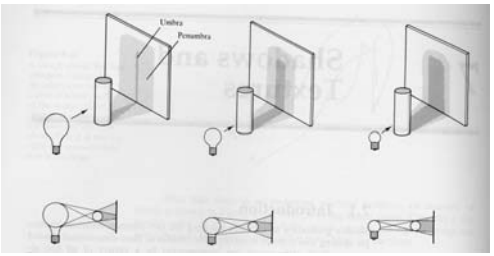
Volumes d'ombre



24

Ombres

Calcul de la pénombre : source non ponctuelle



25