Plan

4 Synthèse d'image
- Présentation
- OpenGL
- De la scène 3D à l'image 2D
- Rendu

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)
- gestion des parties (lignes/surfaces) visibles

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)
- gestion des parties (lignes/surfaces) visibles
- couleurs et textures,

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)
- gestion des parties (lignes/surfaces) visibles
- couleurs et textures,
- utilisation d'ombrage et prise en compte de propriétés d'illumination (lumières et position de l'observateur)

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)
- gestion des parties (lignes/surfaces) visibles
- couleurs et textures,
- utilisation d'ombrage et prise en compte de propriétés d'illumination (lumières et position de l'observateur)
- prise en compte de propriété de réflexion, réfraction et transparence

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)
- gestion des parties (lignes/surfaces) visibles
- couleurs et textures,
- utilisation d'ombrage et prise en compte de propriétés d'illumination (lumières et position de l'observateur)
- prise en compte de propriété de réflexion, réfraction et transparence
- prise en compte de l'atmosphère ambiante (luminosité, brouillard, ...)

Présentation

Rendu en synthèse d'image

ensemble de techniques de représentation permettant de lever les ambiguités dues au passage 3D vers 2D et/ou reproduire un aspect réel (réalisme)

- représentation en mode filaire et/ou ombré
- projection perspective avec focale plus ou moins grande (effet zoom / grand angle)
- gestion des parties (lignes/surfaces) visibles
- couleurs et textures,
- utilisation d'ombrage et prise en compte de propriétés d'illumination (lumières et position de l'observateur)
- prise en compte de propriété de réflexion, réfraction et transparence
- prise en compte de l'atmosphère ambiante (luminosité, brouillard, ...)
- ...

Rendu filaire / rendu ombré

Différents types de rendu

OpenGL : description d'une scène sous forme de triangles / quadrangles (*maillage*)

Rendu filaire / rendu ombré

Différents types de rendu

OpenGL : description d'une scène sous forme de triangles / quadrangles (*maillage*)
$\rightarrow$ deux types de rendus :

Rendu filaire / rendu ombré

Différents types de rendu

OpenGL : description d'une scène sous forme de triangles / quadrangles (*maillage*)
$\rightarrow$ deux types de rendus :

- **rendu filaire** : réprésentation des arêtes du maillage,

Rendu filaire / rendu ombré

Différents types de rendu

OpenGL : description d'une scène sous forme de triangles / quadrangles (*maillage*)
$\rightarrow$ deux types de rendus :

- **rendu filaire** : réprésentation des arêtes du maillage,
- **rendu ombré** : réprésentation des faces (triangles/quadrangles) du maillage.

Rendu filaire / rendu ombré

Différents types de rendu

OpenGL : description d'une scène sous forme de triangles / quadrangles (*maillage*)
$\rightarrow$ deux types de rendus :

- **rendu filaire** : réprésentation des arêtes du maillage,
- **rendu ombré** : réprésentation des faces (triangles/quadrangles) du maillage.

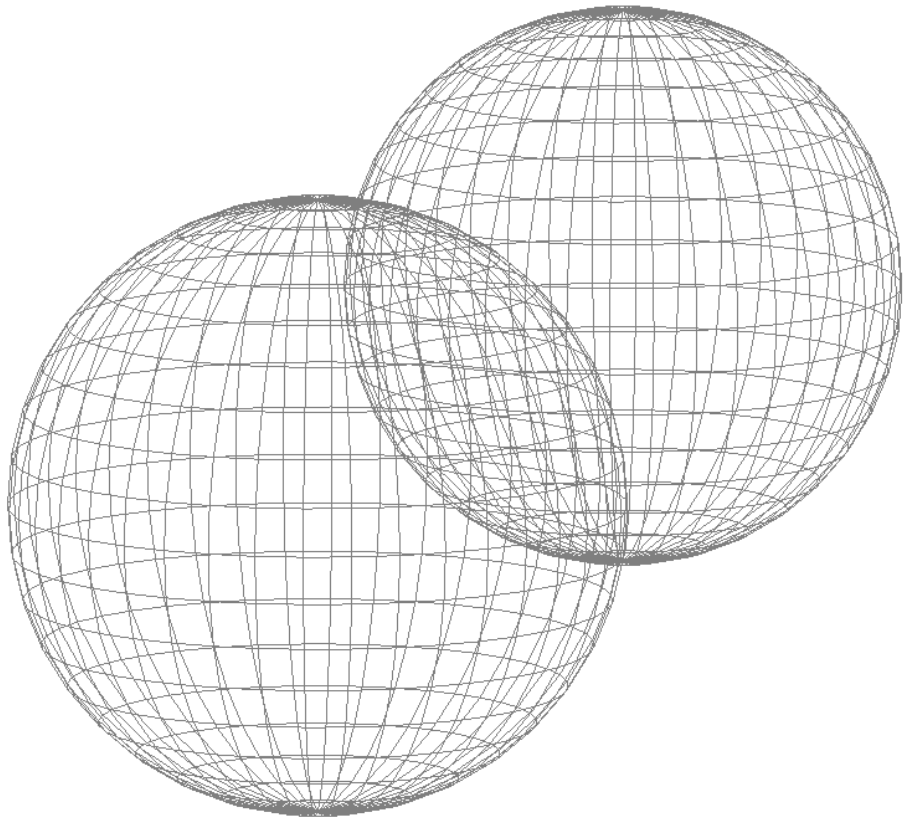
Possibilité de combiner les deux types de rendu.

Rendu filaire / rendu ombré

Différents types de rendu

Rendu filaire / rendu ombré

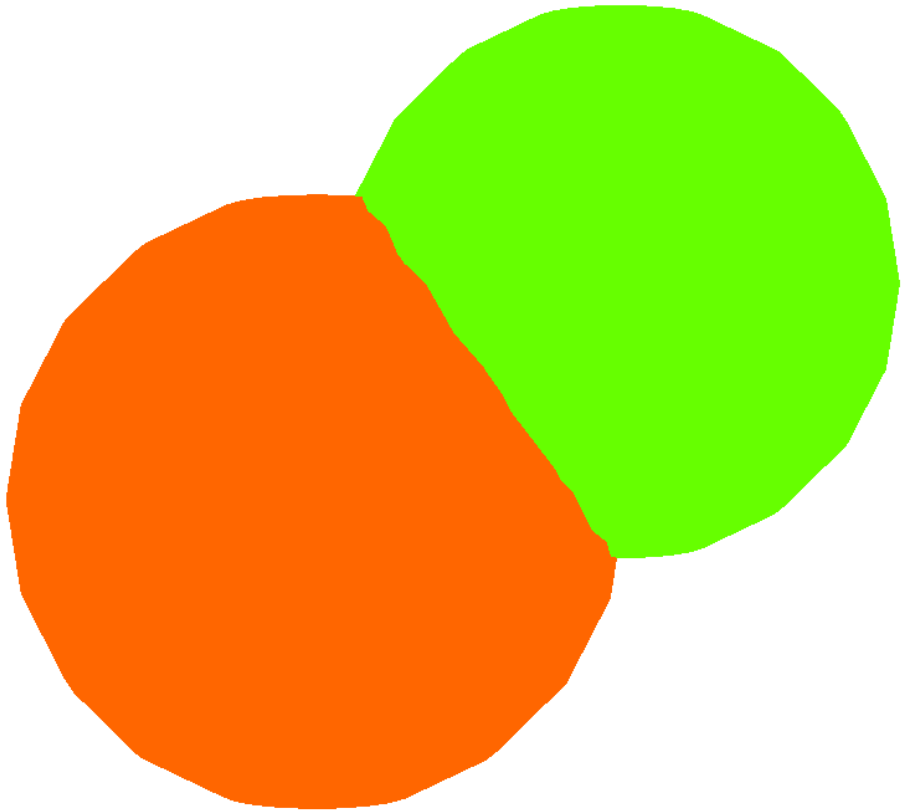
Différents types de rendu



Représentation des arêtes seules ("fil de fer" - *wireframe*)

Rendu filaire / rendu ombré

Différents types de rendu

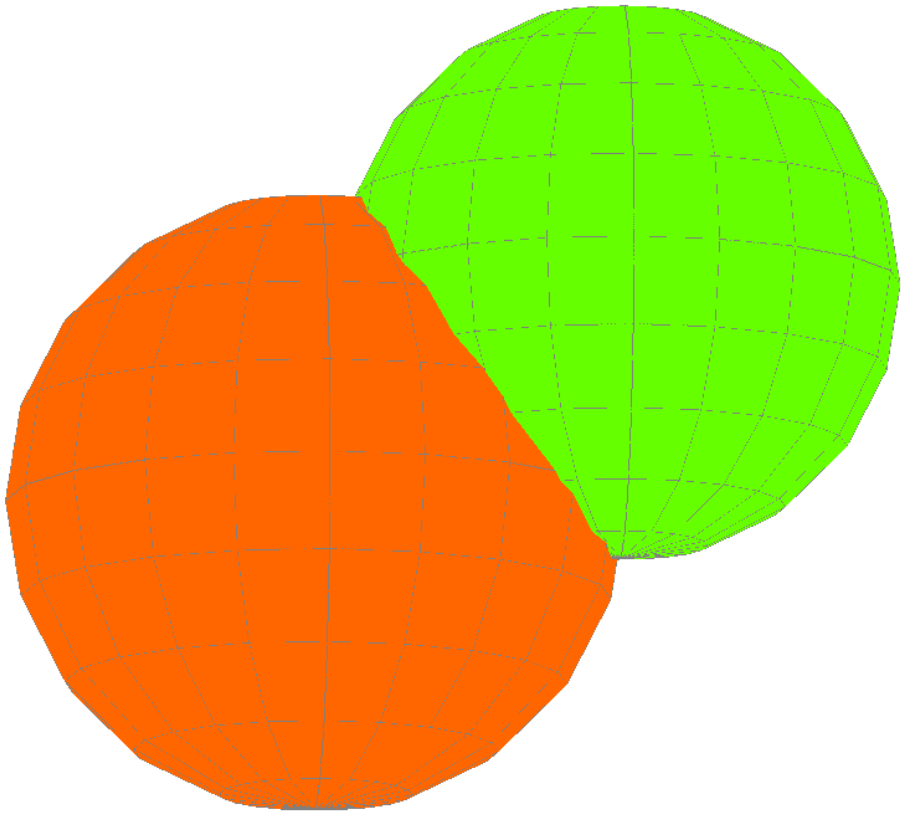


Répresentation des faces ("ombrage" - *shading*)

Couleur unique par objet

Rendu filaire / rendu ombré

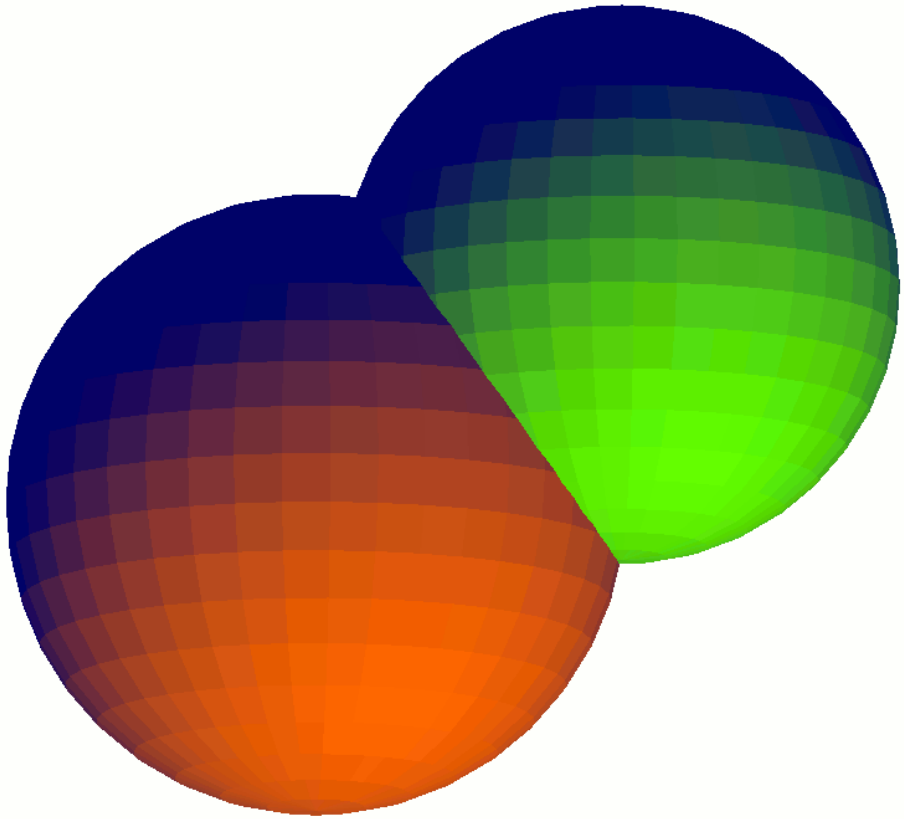
Différents types de rendu



Répresentation mixte - arêtes et faces
Couleur unique par objet

Rendu filaire / rendu ombré

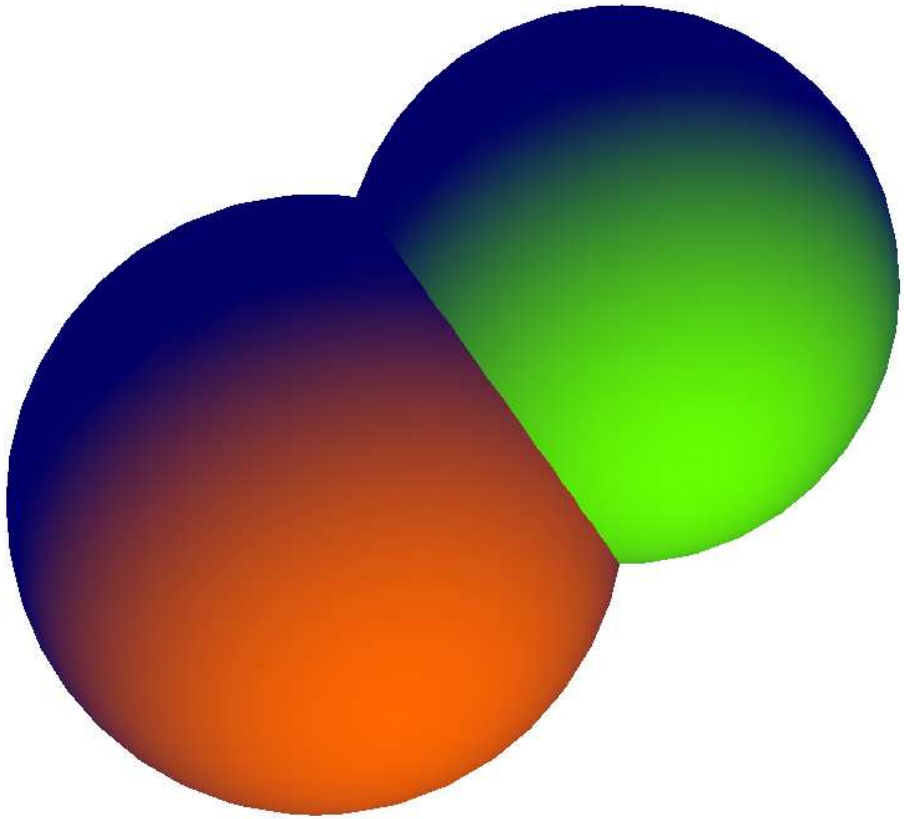
Différents types de rendu



Répresentation des faces uniquement
Couleur unique par face (*flat shading*)

Rendu filaire / rendu ombré

Différents types de rendu

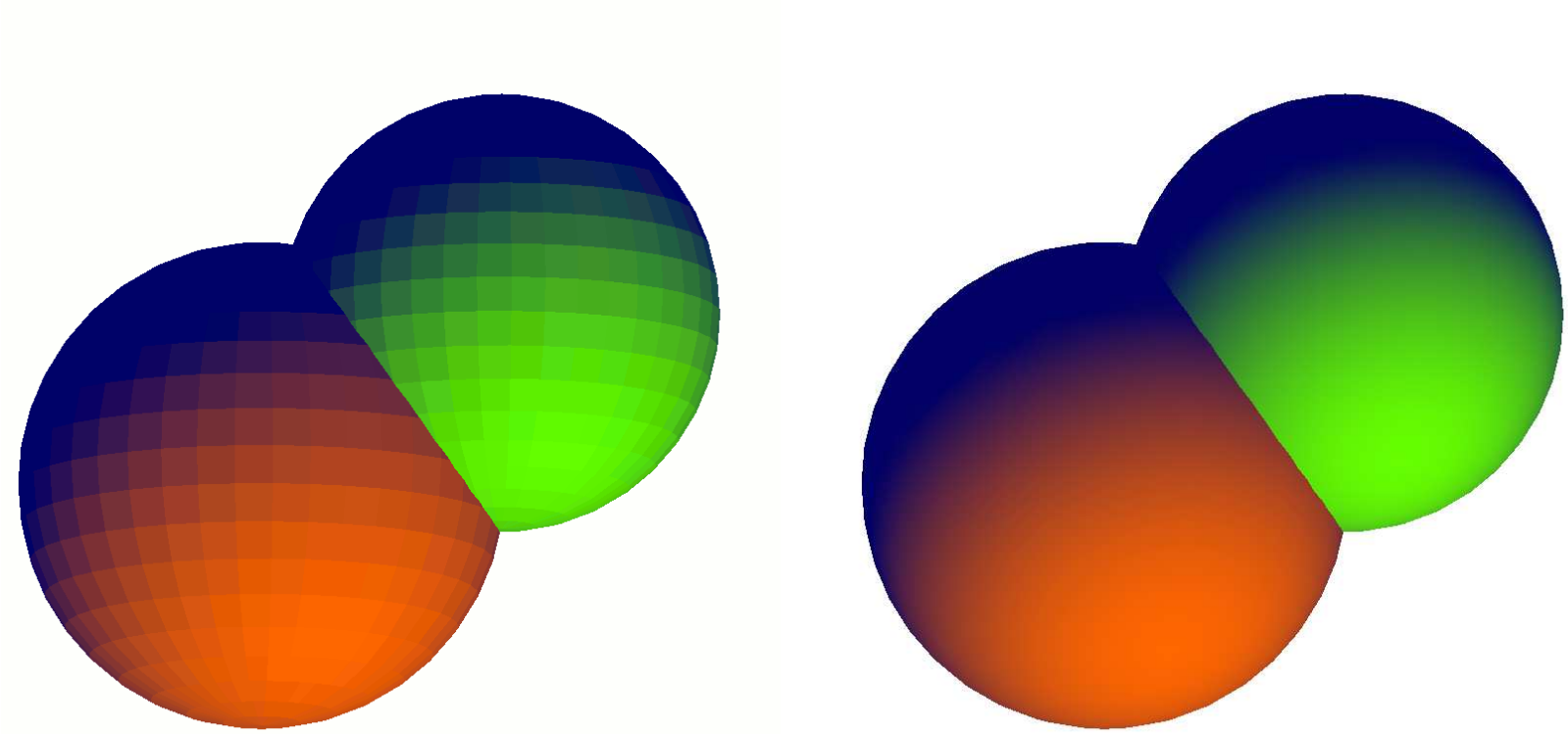


Répresentation des faces uniquement

Dégradé de couleur par face (*Gouraud shading*)

Rendu filaire / rendu ombré

Différents types de rendu



flat shading *Gouraud shading*

couleurs calculées par face/par sommet :
nécessité d'associer un vecteur normal orienté à une face/un sommet

Maillage 3D

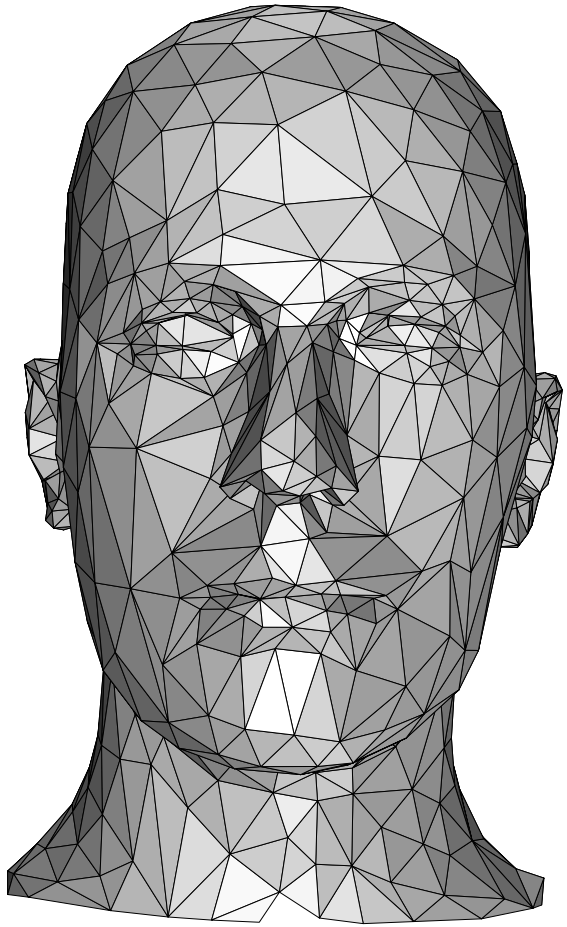
Scène OpenGL

Décomposition de la scène en triangles, quadrangles, polygones.

Maillage 3D

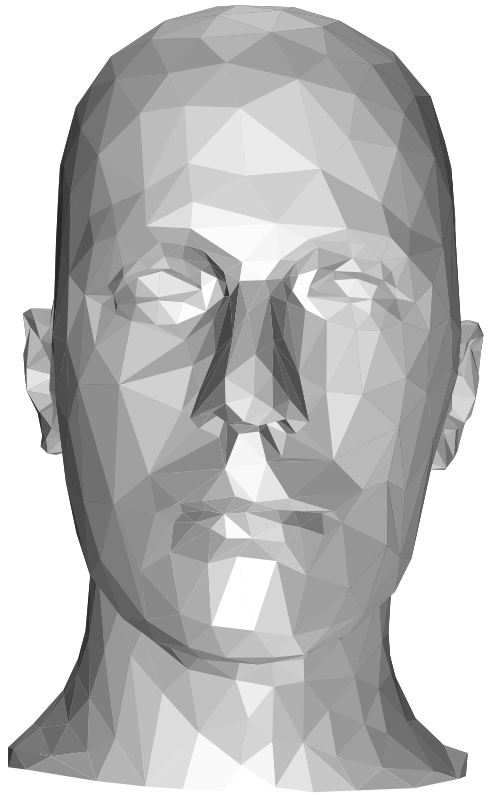
Scène OpenGL

Décomposition de la scène en triangles, quadrangles, polygones.



Maillage 3D

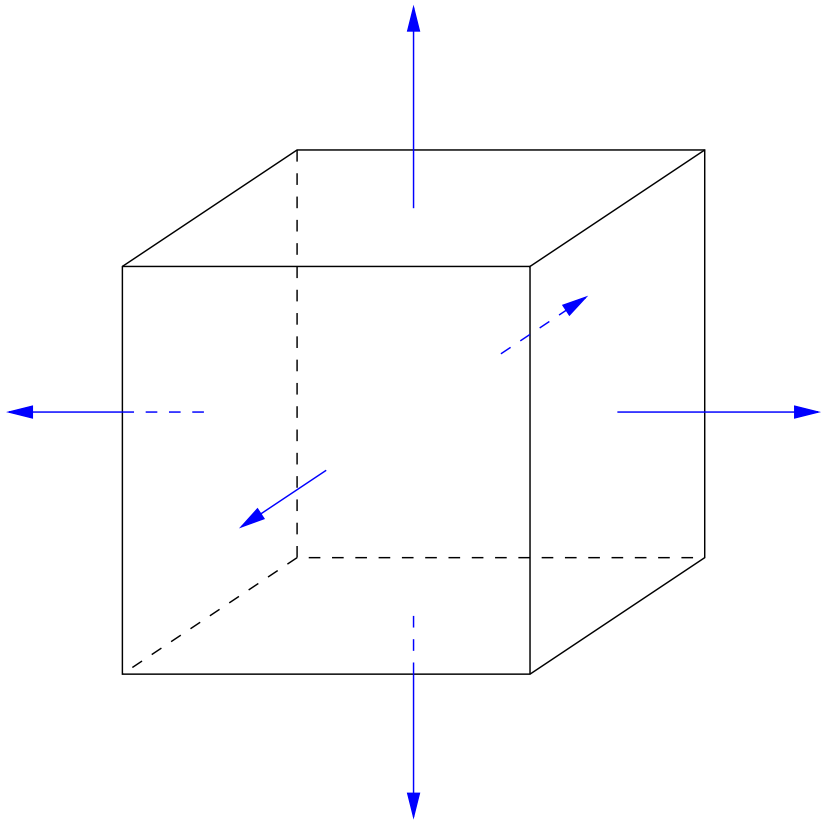
Utilisation de (vecteurs) normales



En général, pour le rendu *ombré* (par face), nécessité de connaitre les (vecteurs) normales orientées par face / par sommet

Maillage 3D

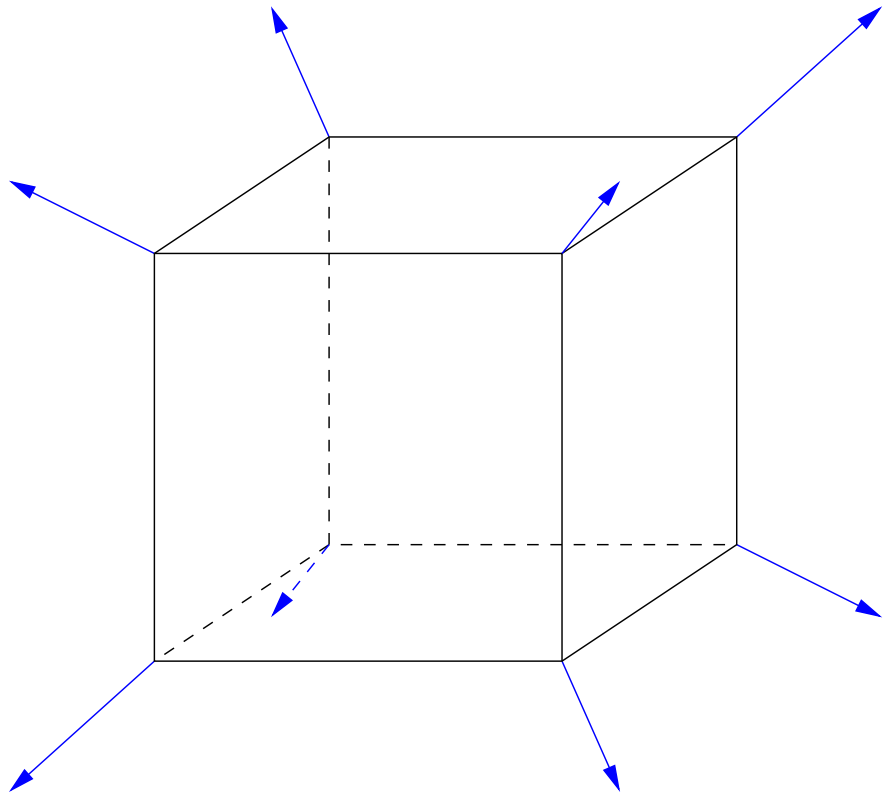
Utilisation de (vecteurs) normales



Normales aux faces *orientées* (vers l'extérieur)

Maillage 3D

Utilisation de (vecteurs) normales



Normales aux sommets *orientées* (vers l'extérieur)

Maillage 3D

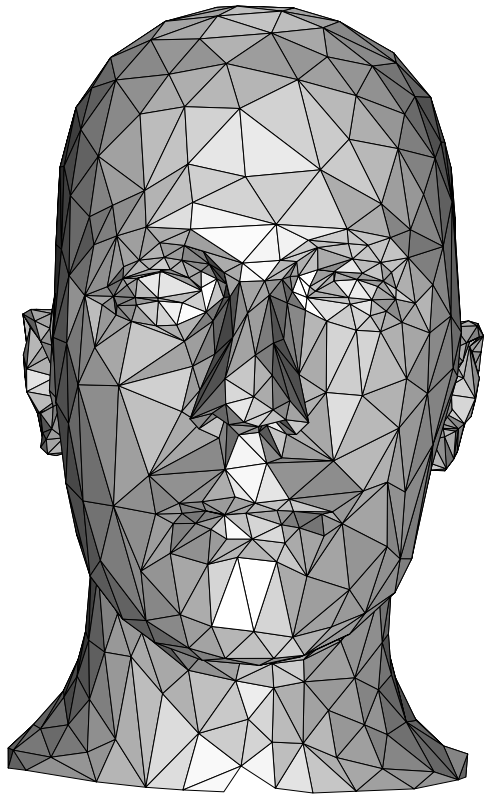
Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées

Maillage 3D

Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées

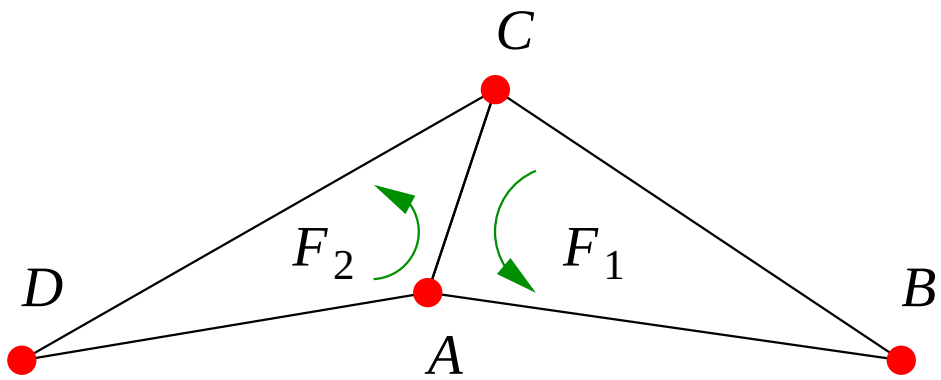


Nécessité d'avoir un maillage **orienté**

Maillage 3D

Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées



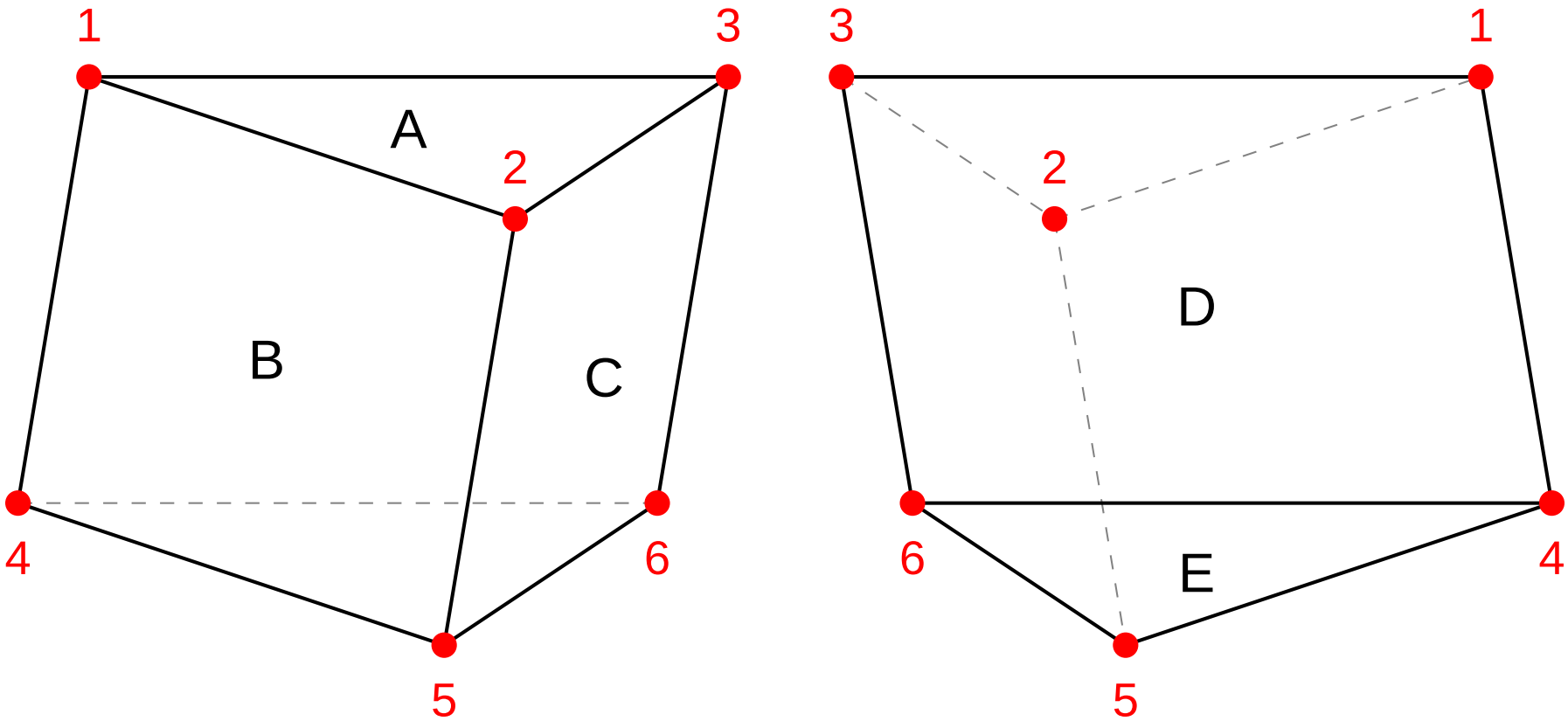
Face	Sommet S_1	Sommet S_2	Sommet S_3
F_1	B	C	A
F_2	A	C	D

Nécessité d'avoir un maillage **orienté**

Maillage 3D

Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées



Exemple 1 - prisme à base triangulaire

Maillage 3D

Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées

Sommet	Coordonnées
1	(0,0,0)
2	(0,1,0)
3	(1,1,0)
4	(0,0,1)
5	(0,1,1)
6	(1,1,1)

Face	Liste orientée des sommets
A	1 2 3
B	2 1 4 5
C	3 2 5 6
D	1 3 6 4
E	6 5 4

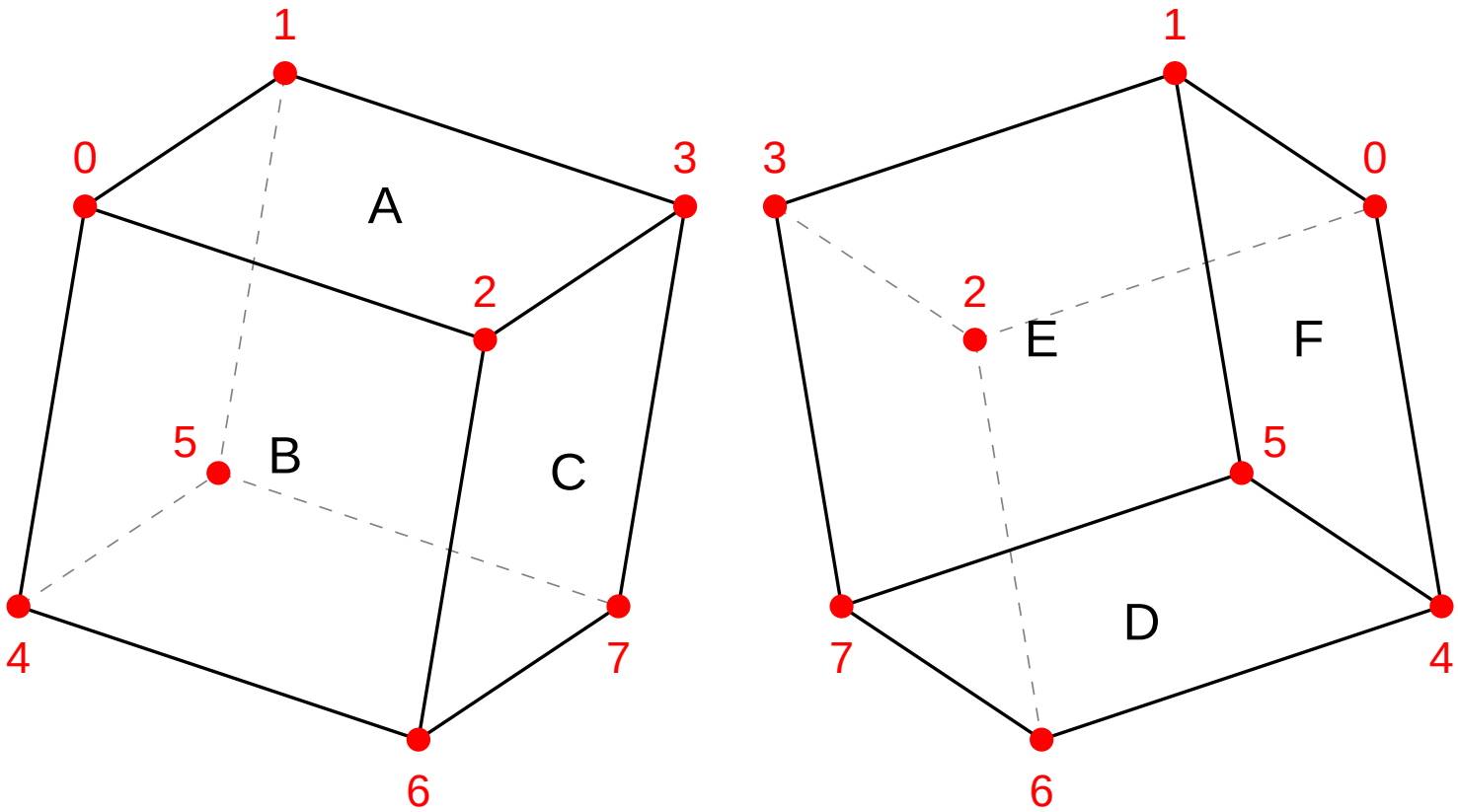
Description du maillage

Exemple 1 - prisme à base triangulaire

Maillage 3D

Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées



Exemple 2 - cube

Maillage 3D

Orientation des faces

Maillage surfacique : ensemble de faces (polygones convexes) orientées

Sommet	Coordonnées
0	(0,0,0)
1	(1,0,0)
2	(0,1,0)
3	(1,1,0)
4	(0,0,1)
5	(1,0,1)
6	(0,1,1)
7	(1,1,1)

Face	Liste orientée des sommets
A	0 2 3 1
B	6 2 0 4
C	2 6 7 3
D	4 5 7 6
E	5 1 3 7
F	0 1 5 4

Description du maillage

Exemple 2 - cube

Maillage 3D

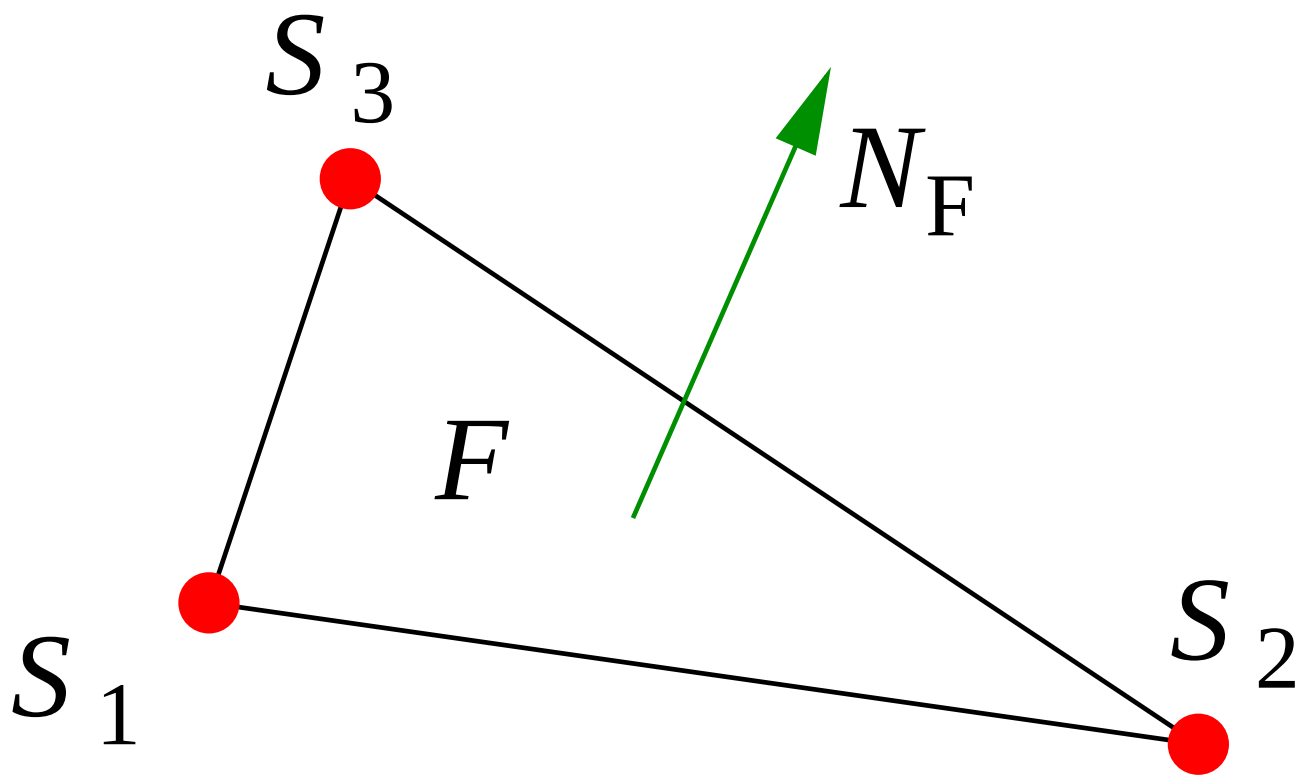
Calcul de la normale à un sommet / une face

Cas d'un maillage quelconque

Maillage 3D

Calcul de la normale à un sommet / une face

Cas d'un maillage quelconque



Normale orientée N_F pour une face $F = [S_1, S_2, S_3]$

Maillage 3D

Calcul de la normale à un sommet / une face

Cas d'un maillage quelconque

$$W = \overrightarrow{S_1 S_2} \wedge \overrightarrow{S_1 S_3}$$

et

$$N_F = \frac{1}{\|W\|} W$$

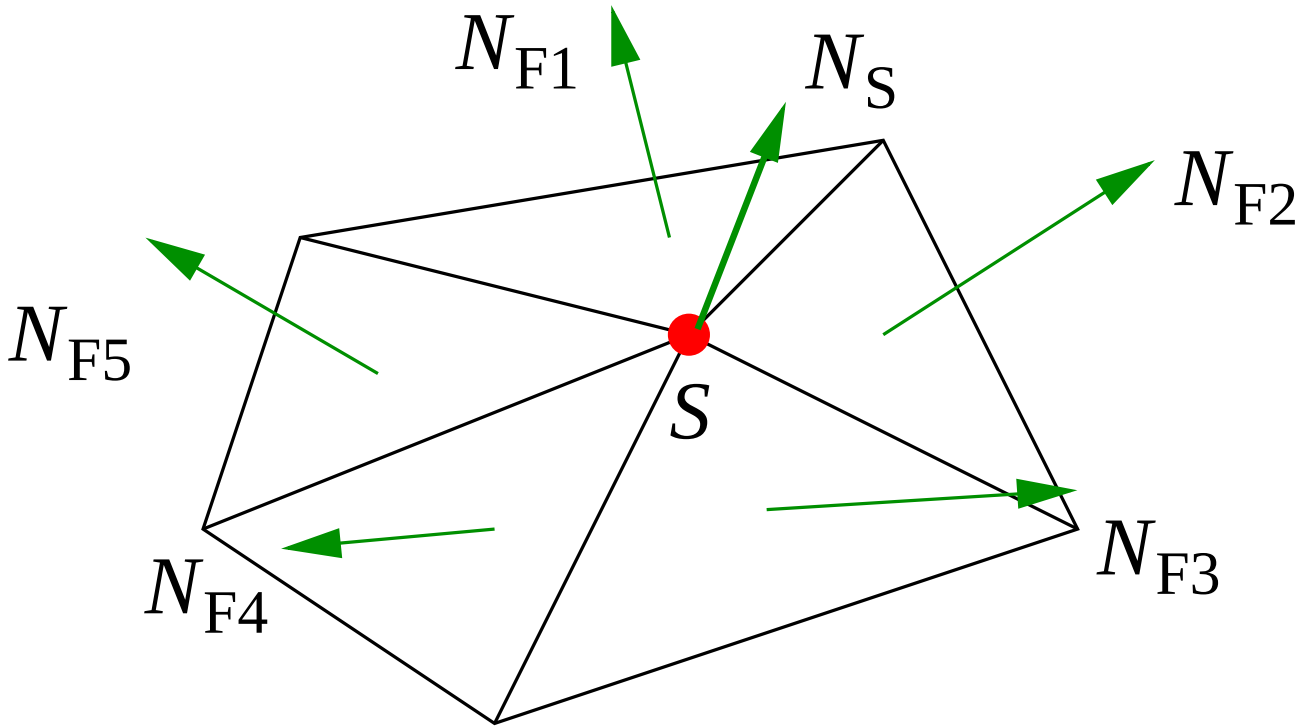
N_F vecteur unitaire (de longueur 1)

Normale orientée N_F pour une face $F = [S_1, S_2, S_3]$

Maillage 3D

Calcul de la normale à un sommet / une face

Cas d'un maillage quelconque



Normale orientée N_S pour un sommet $S \in \{F_1, \ldots, F_M\}$

Maillage 3D

Calcul de la normale à un sommet / une face

Cas d'un maillage quelconque

$$W = \sum_{i=1}^{M} \alpha_i N_{F_i}$$

et

$$N_S = \frac{1}{\|W\|} W$$

$$\alpha_i = 1 \quad \text{ou} \quad \alpha_i = \text{aire}(F_i) \quad \text{ou} \quad \alpha_i = \text{angle}(S, F_i)$$

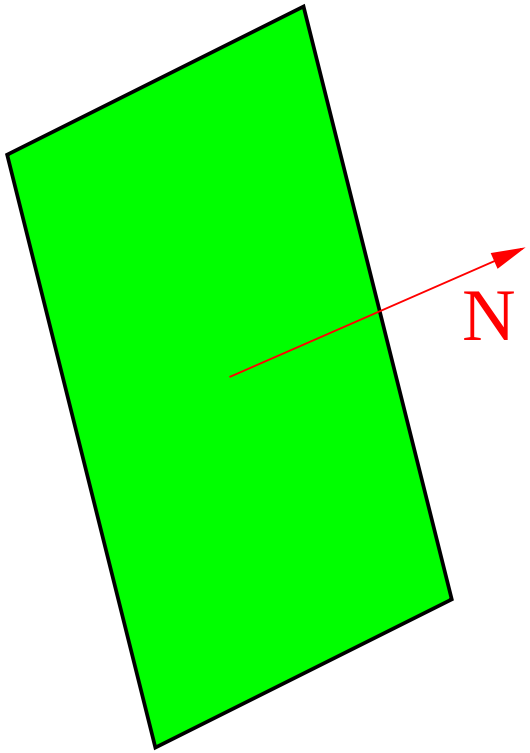
Normale orientée N_S pour un sommet $S \in \{F_1, \ldots, F_M\}$

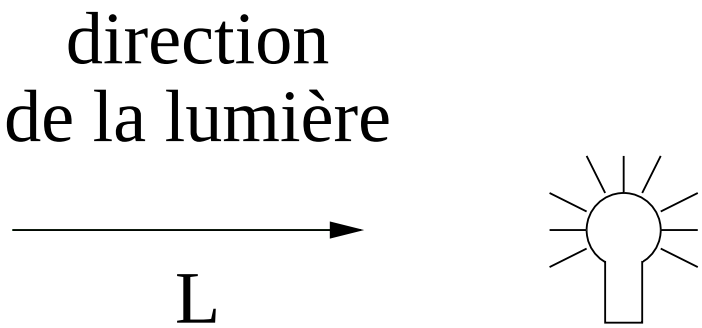
Modèle d'illumination

Exemple du modèle de Lambert

Modèle d'illumination

Exemple du modèle de Lambert



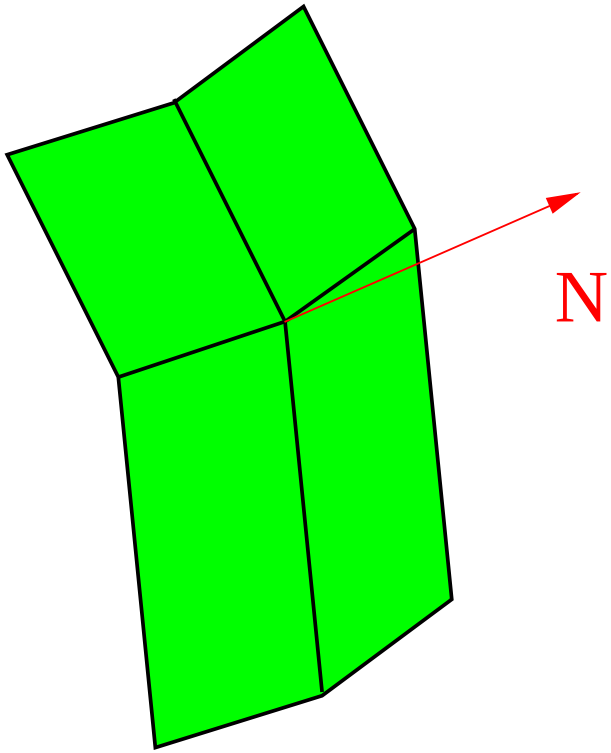


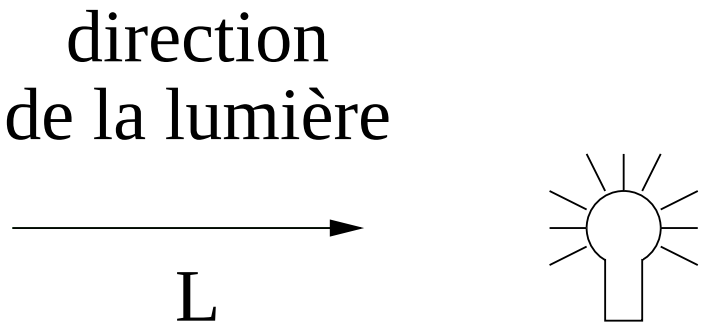
$$\text{Couleur} = C(\langle \overrightarrow{N}, \overrightarrow{L} \rangle)$$

normale orientée et unitaire N associée à chaque face

Modèle d'illumination

Exemple du modèle de Lambert





$$\text{Couleur} = C(\langle \overrightarrow{N}, \overrightarrow{L} \rangle)$$

normale orientée et unitaire N associée à chaque sommet

Modèle d'illumination

Exemple du modèle de Lambert

$$\text{Calcul de la couleur} = C(\langle \overrightarrow{N}, \overrightarrow{L} \rangle)$$

N et L vecteurs unitaires

Modèle d'illumination

Exemple du modèle de Lambert

C_1 : couleur de l'objet
C_2 : couleur noir

$$\text{Calcul de la couleur} = C(\langle \overrightarrow{N}, \overrightarrow{L} \rangle)$$
N et L vecteurs unitaires

Modèle d'illumination

Exemple du modèle de Lambert

C_1 : couleur de l'objet
C_2 : couleur noir

calculer $I = \langle \overrightarrow{N}, \overrightarrow{L} \rangle$ $\qquad (-1 \leq I \leq 1)$

Calcul de la couleur $= C(\langle \overrightarrow{N}, \overrightarrow{L} \rangle)$
N et L vecteurs unitaires

Modèle d'illumination

Exemple du modèle de Lambert

C_1 : couleur de l'objet
C_2 : couleur noir

calculer $I = \langle \overrightarrow{N}, \overrightarrow{L} \rangle \qquad (-1 \leq I \leq 1)$

si $I \geq 0$ alors Couleur $= I \times C_1 + (1 - I) \times C_2$
sinon Couleur $= C_2$

$$\text{Calcul de la couleur} = C(\langle \overrightarrow{N}, \overrightarrow{L} \rangle)$$

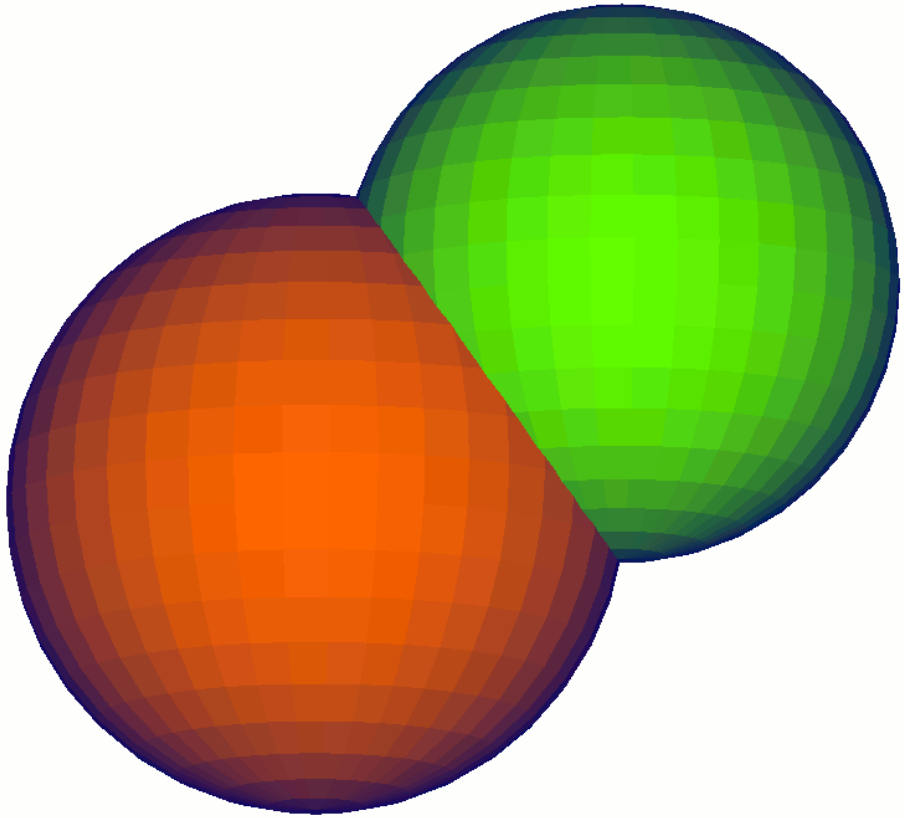
N et L vecteurs unitaires

Modèle d'illumination

Exemple du modèle de Lambert - utilisation d'une normale orientée par face

Modèle d'illumination

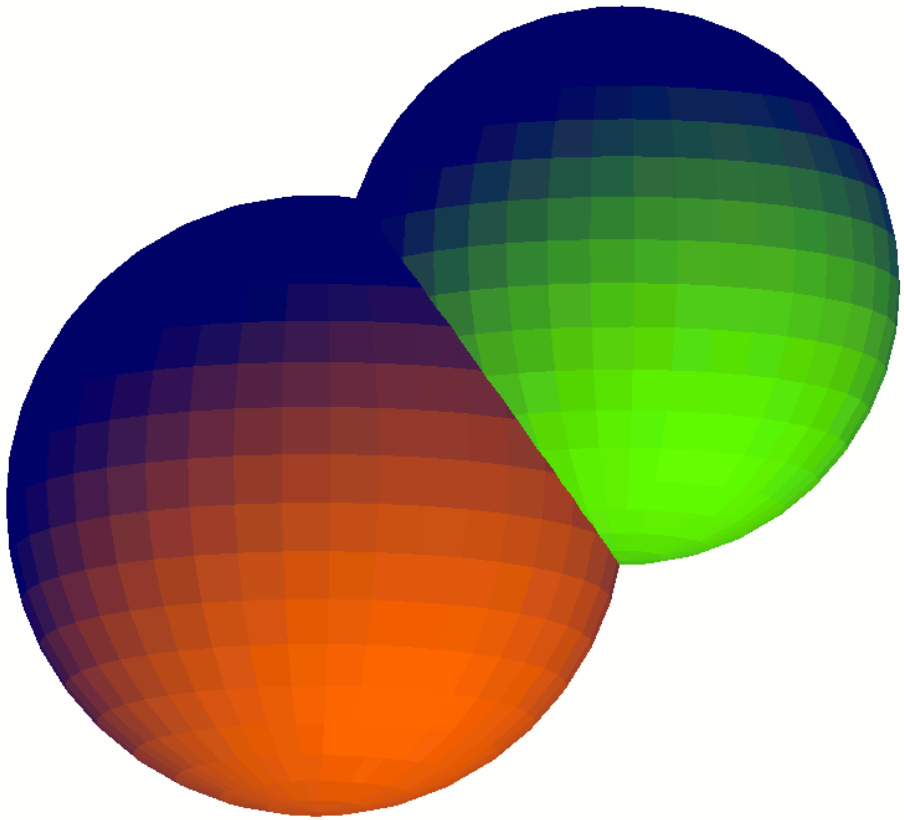
Exemple du modèle de Lambert - utilisation d'une normale orientée par face



Direction d'éclairement L égale à la direction de vue

Modèle d'illumination

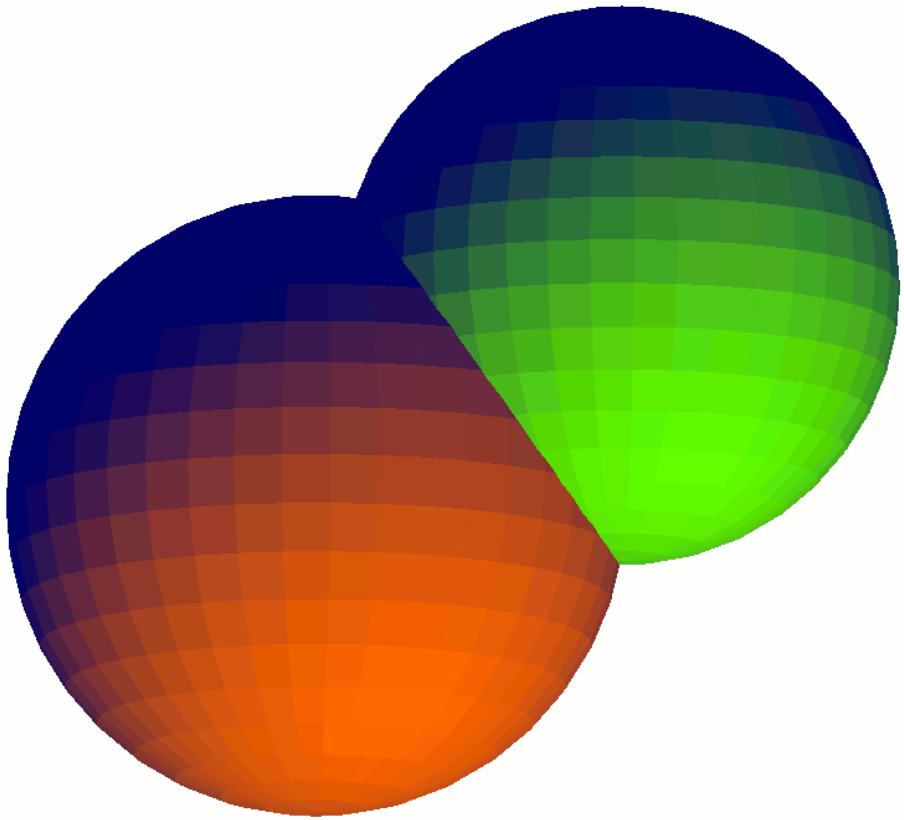
Exemple du modèle de Lambert - utilisation d'une normale orientée par face



Une autre direction d'éclairement

Modèle d'illumination

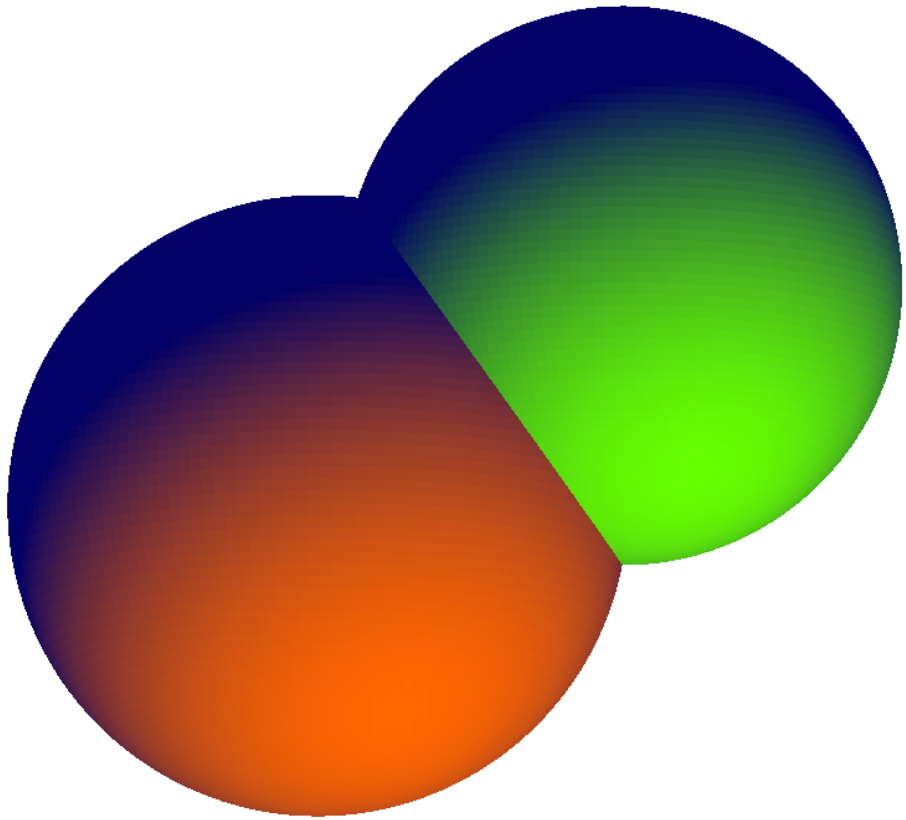
Exemple du modèle de Lambert - utilisation d'une normale orientée par face



Facettisation grossière

Modèle d'illumination

Exemple du modèle de Lambert - utilisation d'une normale orientée par face



Facettisation fine

Rendu ombré

Méthode de Gouraud

Méthode utilisant les normales aux sommets et permettant d'avoir un dégradé continu sur un ensemble de faces.

Rendu ombré

Méthode de Gouraud

Méthode utilisant les normales aux sommets et permettant d'avoir un dégradé continu sur un ensemble de faces.
Triangle T de sommets $[A, B, C]$
Point P de T :

$$P = u\ A + v\ B + w\ C \text{ avec } u, v, w \geq 0 \text{ et } u + v + w = 1$$

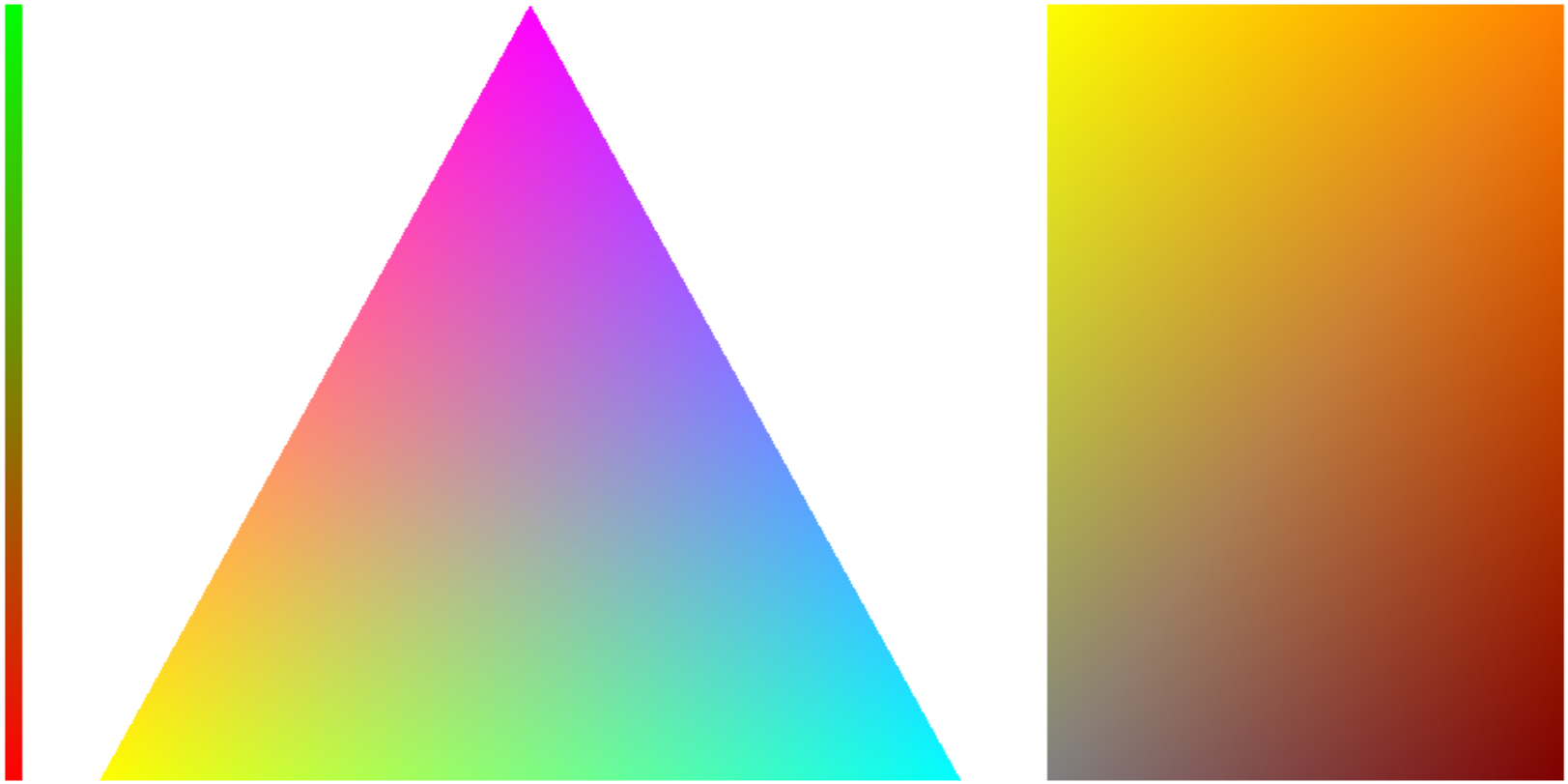
(u, v, w) coordonnées barycentriques de P dans T

$$\left\{ \begin{array}{lcll} \text{sommet } A & + & \text{normale } N_A & : \text{ couleur } C_A \\ \text{sommet } B & + & \text{normale } N_B & : \text{ couleur } C_B \\ \text{sommet } C & + & \text{normale } N_C & : \text{ couleur } C_C \end{array} \right\}$$

$$\rightarrow \text{ point } P \ : \ \text{couleur } C_P = u\ C_A + v\ C_B + w\ C_C$$

Rendu ombré

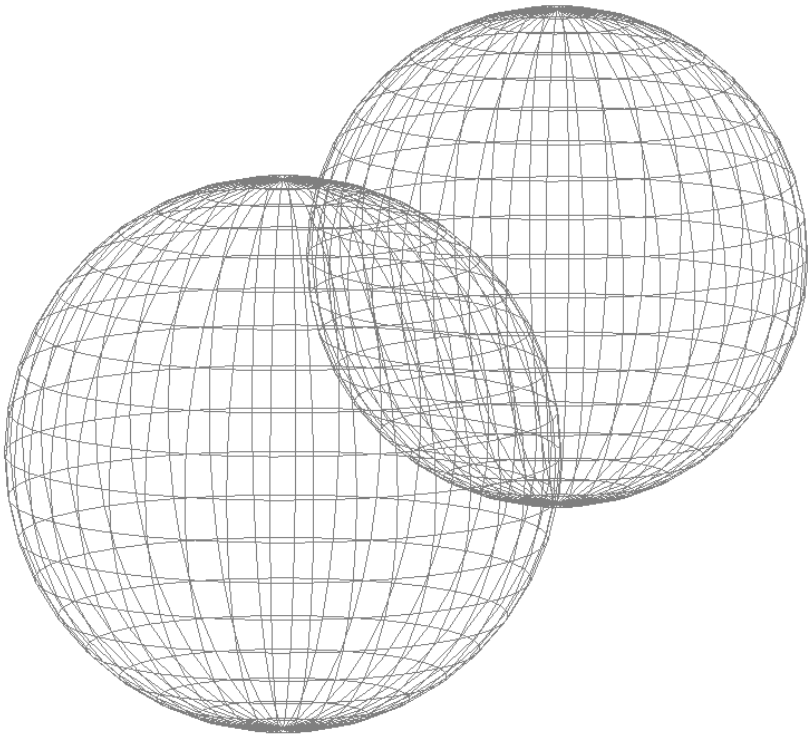
Méthode de Gouraud

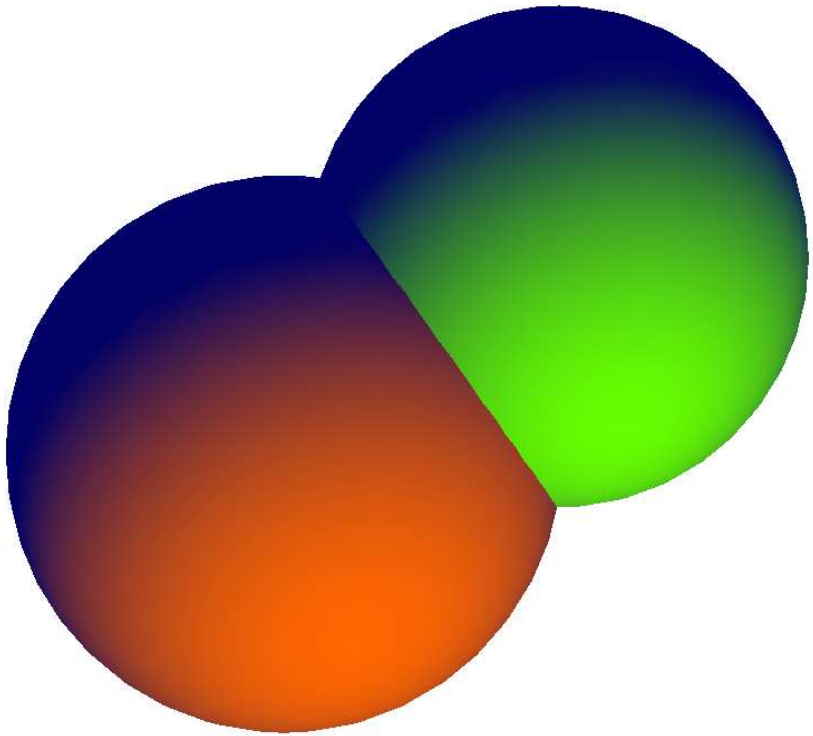


Exemples

Rendu ombré

Méthode de Gouraud

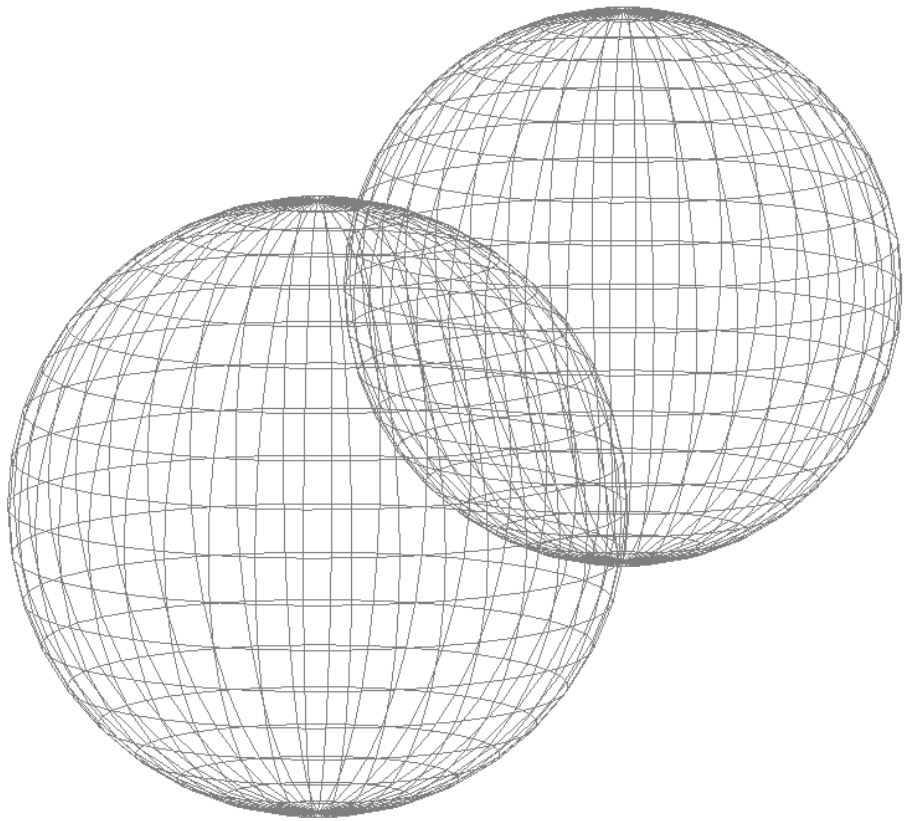




Exemples

Rendu filaire / rendu ombré

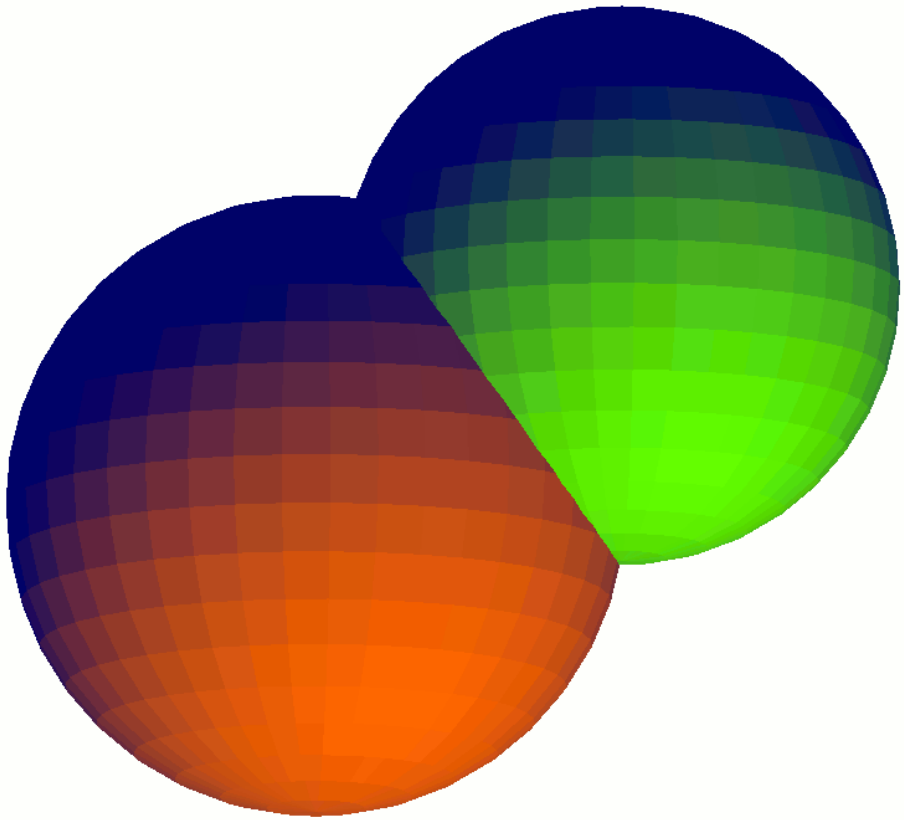
Comparatif



Deux sphères facettisées avec 800 faces par sphère

Rendu filaire / rendu ombré

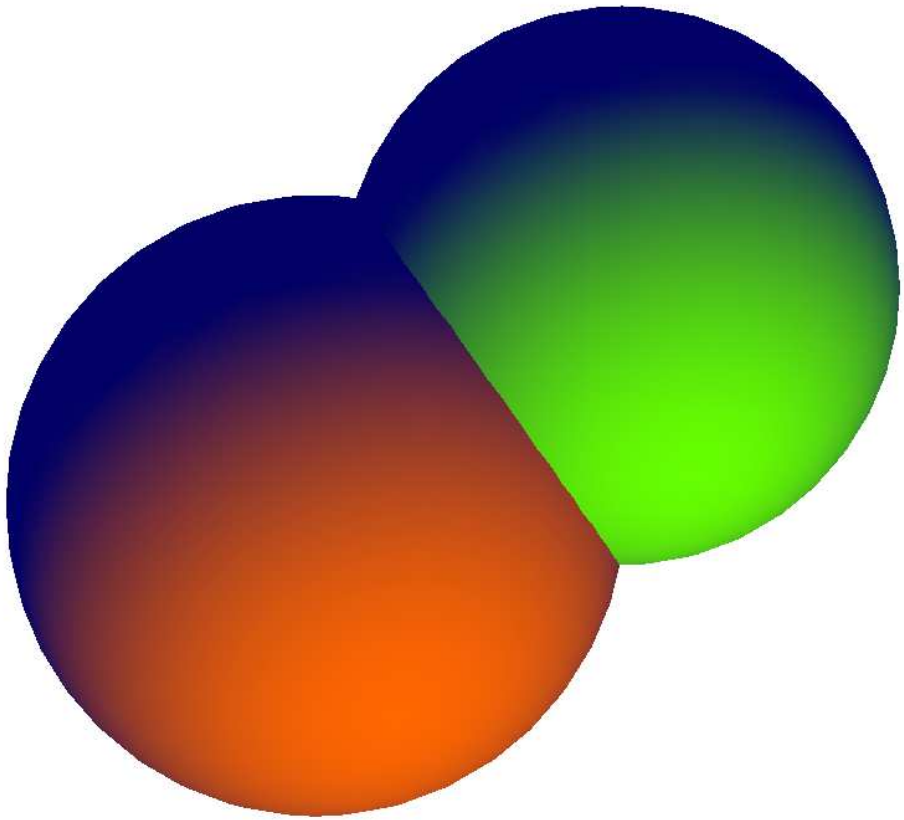
Comparatif



Ombrage constant par face

Rendu filaire / rendu ombré

Comparatif



Ombrage de Gouraud

Rendu filaire / rendu ombré

Code OpenGL

```
/* ------------------------------------------------------------ */
/* tracé de la face triangulaire avec une couleur unie          */
/* ------------------------------------------------------------ */
/* calcul de la normale N à une face                            */
...
/* calcul de la couleur (R,G,B) */
/* en fonction de la normale N  */
...
 /* affecter la couleur (R,G,B) au triangle */
glColor3d(R,G,B);
glBegin(GL_TRIANGLES);
   glVertex3d(xA,yA,zA);
   glVertex3d(xB,yB,zB);
   glVertex3d(xC,yC,zC);
glEnd();
```

Rendu filaire / rendu ombré

Code OpenGL

```
/* ------------------------------------------------------------ */
/*                 tracé de la face triangulaire                */
/*          avec une couleur spécifique par sommet              */
/* ------------------------------------------------------------ */
glBegin(GL_TRIANGLES);
  /* calcul de la normale NA au sommet A */
  ...
  /* calcul de la couleur (RA,GA,BA) */
  /* en fonction de la normale NA    */
  ...
  /* affecter la couleur (RA,GA,BA) au sommet A*/
  glColor3d(RA,GA,BA);
  glVertex3d(xA,yA,zA);

  ...
```

Rendu filaire / rendu ombré

Code OpenGL

```
...

/* calcul de la normale NB au sommet B */
...
/* calcul de la couleur (RB,GB,BB) */
/* en fonction de la normale NB    */
...
/* affecter la couleur (RB,GB,BB) au sommet B*/
glColor3d(RB,GB,BB);
glVertex3d(xB,yB,zB);

...
```

Rendu filaire / rendu ombré

Code OpenGL

```
    ...


    /* calcul de la normale NC au sommet C */
    ...
    /* calcul de la couleur (RC,GC,BC) */
    /* en fonction de la normale NC    */
    ...
    /* affecter la couleur (RC,GC,CC) au sommet C*/
    glColor3d(RC,GC,BC);
    glVertex3d(xC,yC,zC);

glEnd();
```

Gestion des parties visibles

Gestion des parties visibles

Représentation en mode `ombré` - remplissage des faces

Gestion des parties visibles

Représentation en mode `ombré` - remplissage des faces
deux principaux algorithmes :

Gestion des parties visibles

Représentation en mode `ombré` - remplissage des faces
deux principaux algorithmes :

- scène simple : **algorithme du peintre**

Gestion des parties visibles

Représentation en mode `ombré` - remplissage des faces
deux principaux algorithmes :

- scène simple : **algorithme du peintre**
- scène complexe : **algorithme du z-buffer**

Gestion des parties visibles

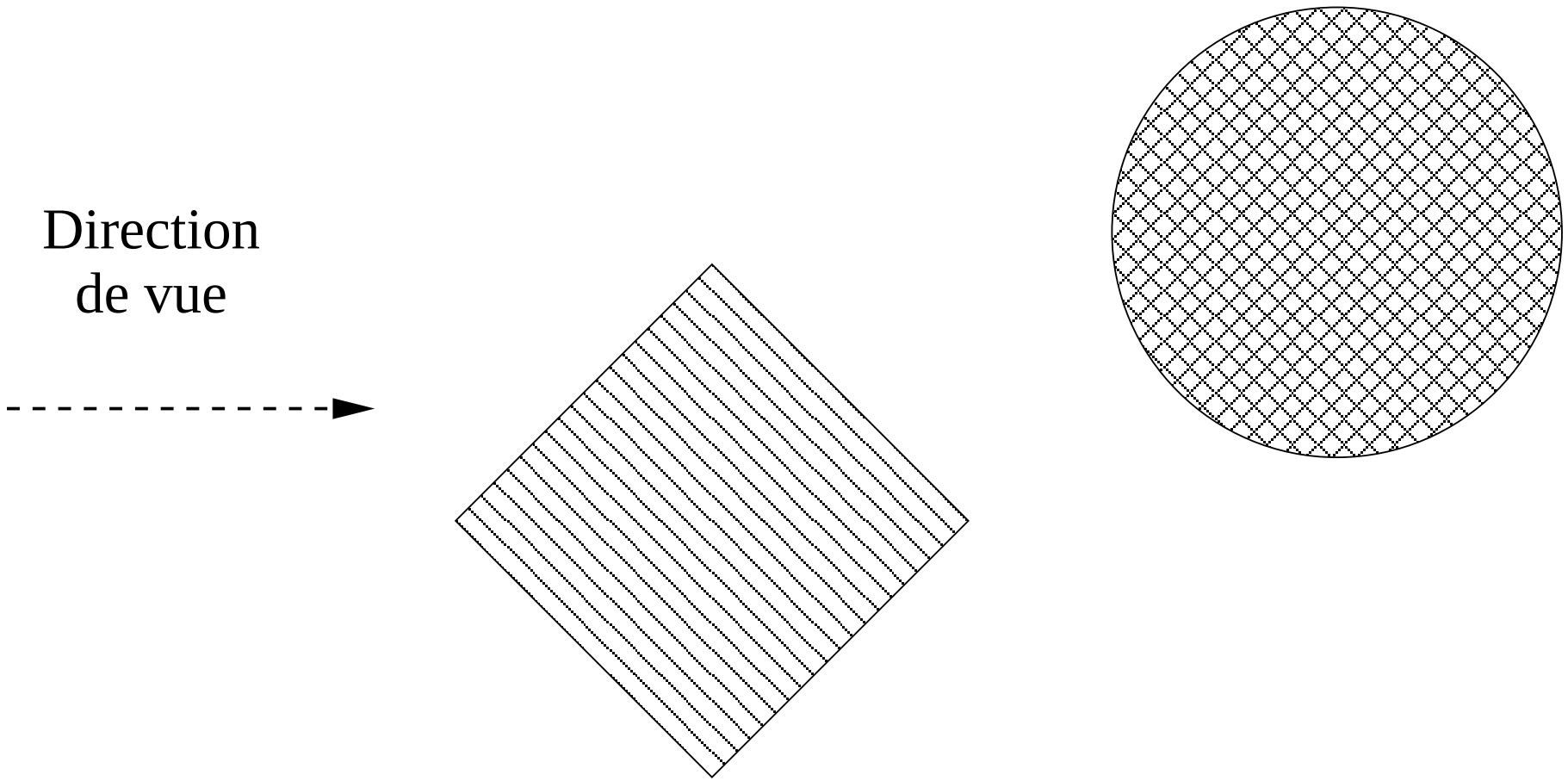
Algorithme du peintre

Tri des objets suivant leurs positions par rapport à la caméra

Gestion des parties visibles

Algorithme du peintre

Tri des objets suivant leurs positions par rapport à la caméra

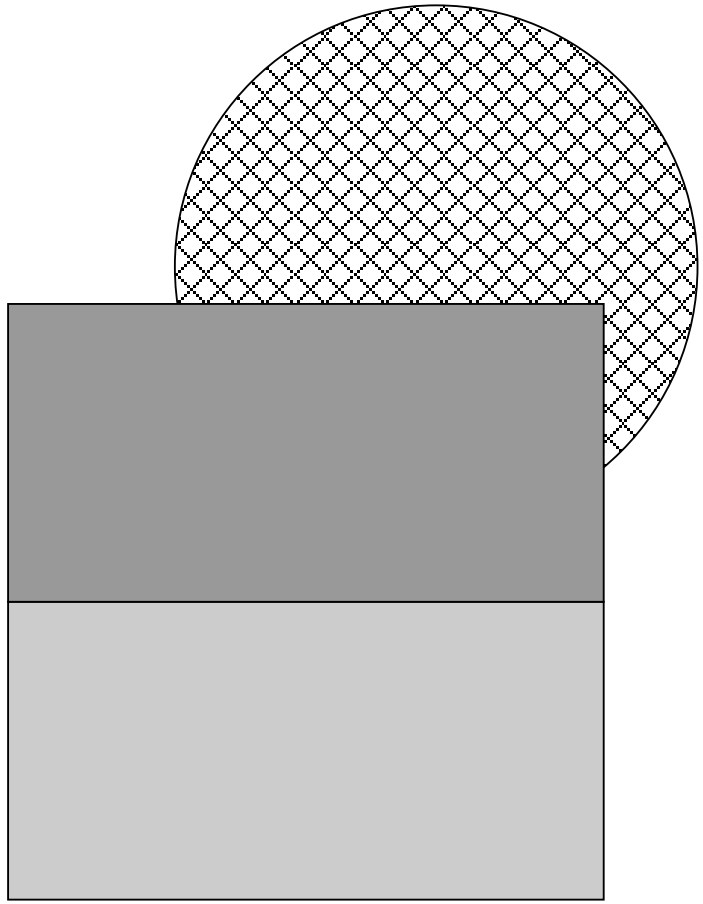


La scène et la direction de vue

Gestion des parties visibles

Algorithme du peintre

Tri des objets suivant leurs positions par rapport à la caméra



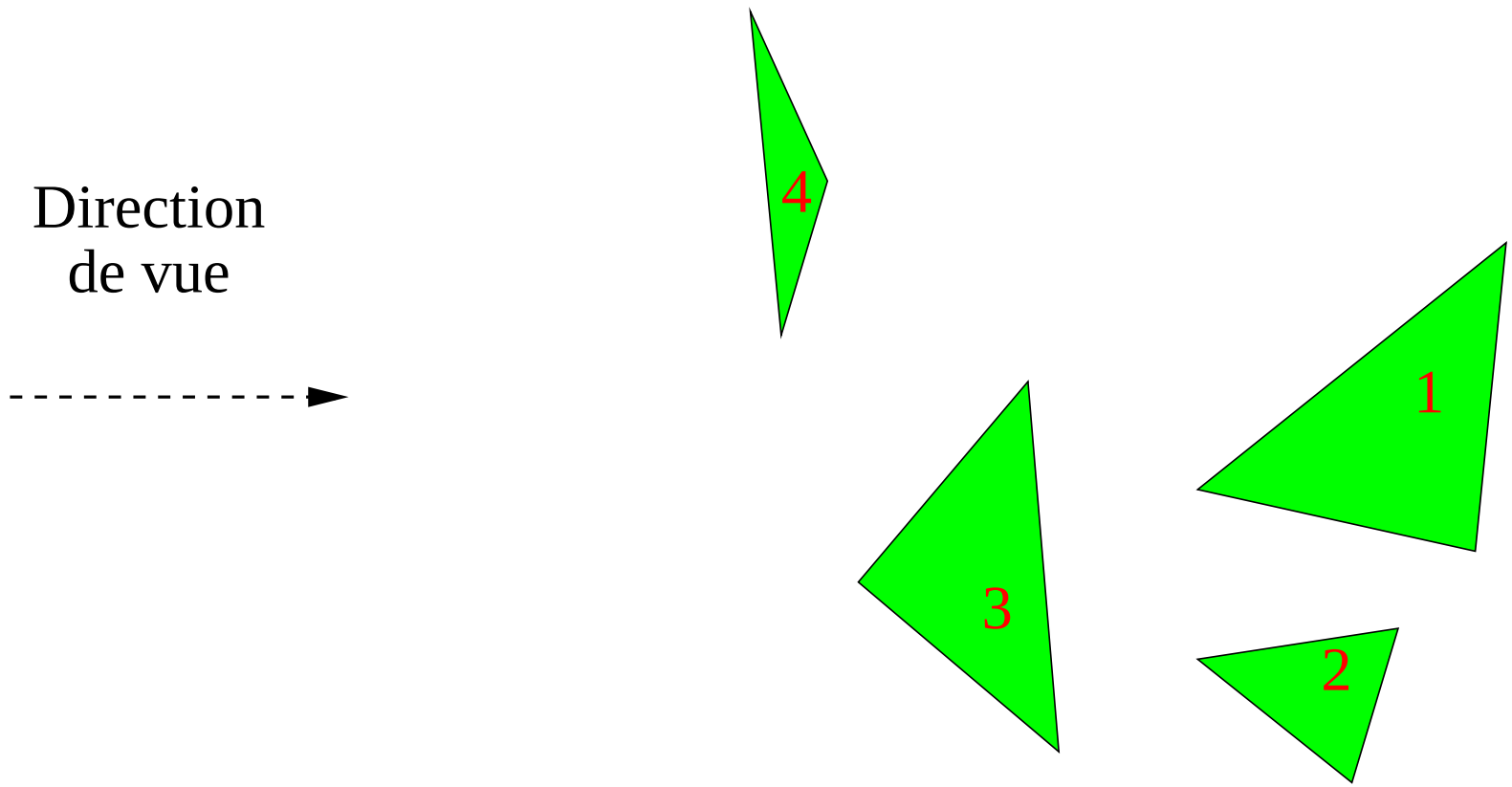
Vue depuis la caméra

Gestion des parties visibles

Algorithme du peintre

Gestion des parties visibles

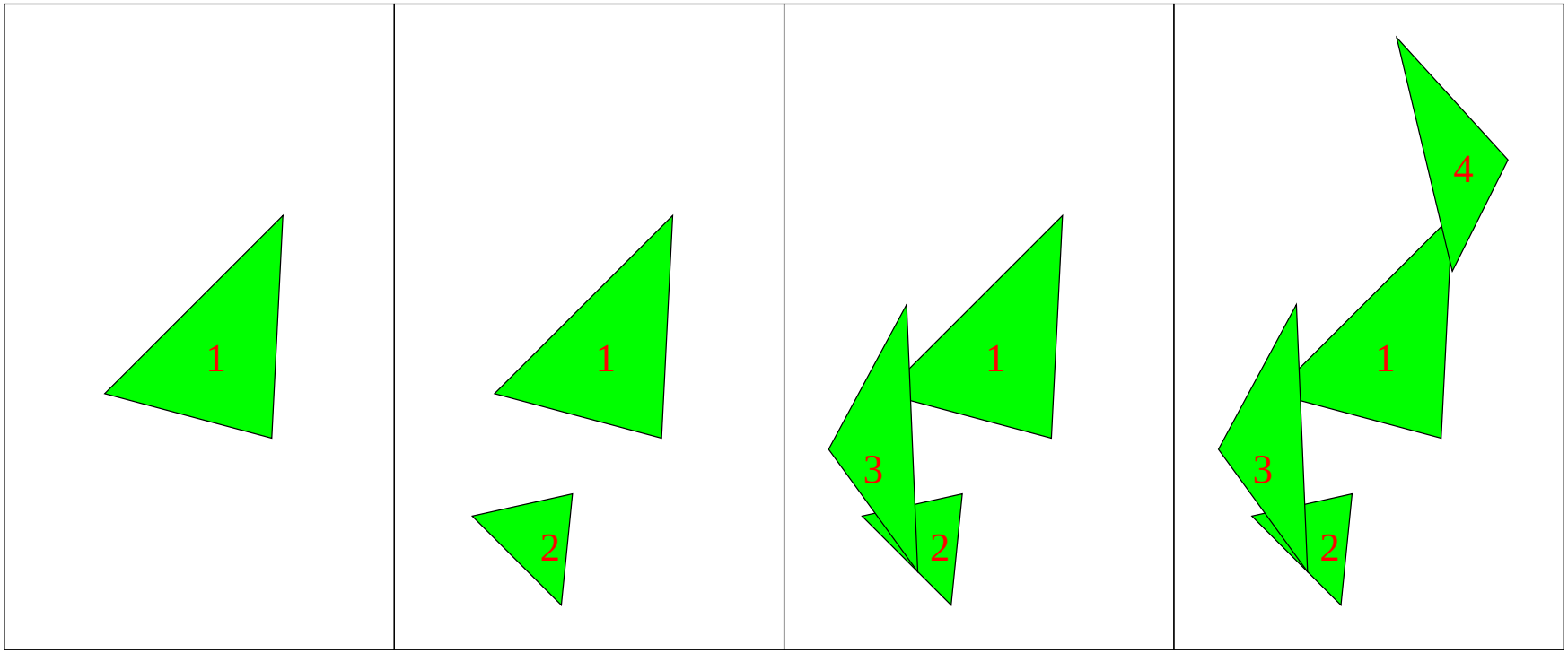
Algorithme du peintre



ordonnancement des triangles vus depuis l'observateur : tri en z

Gestion des parties visibles

Algorithme du peintre



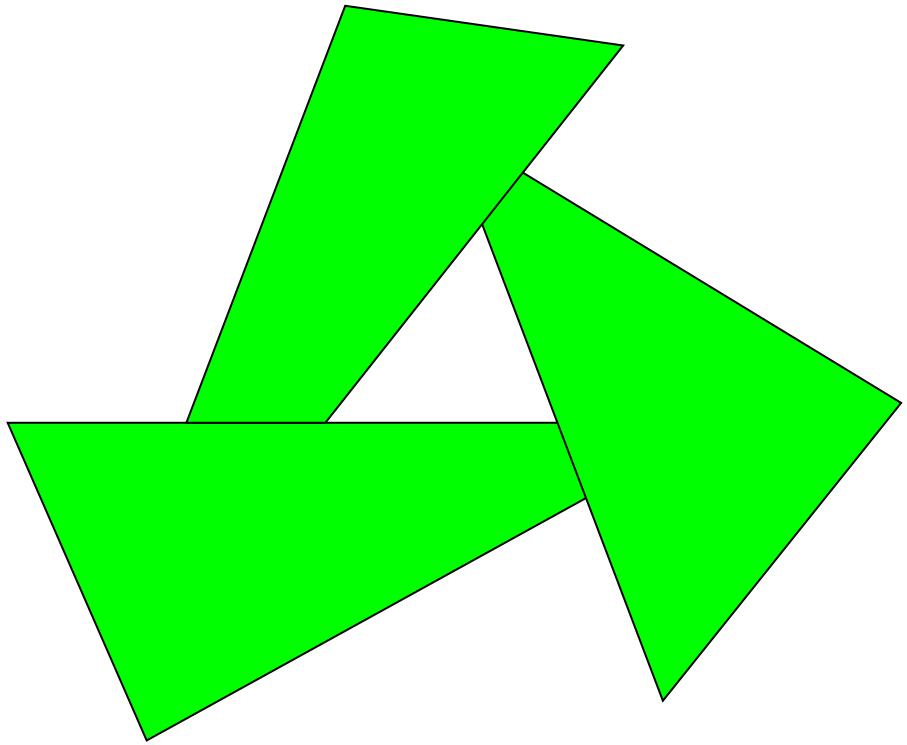
affichage des triangles du plus loin au plus proche

Gestion des parties visibles

Algorithme du peintre - limites

Gestion des parties visibles

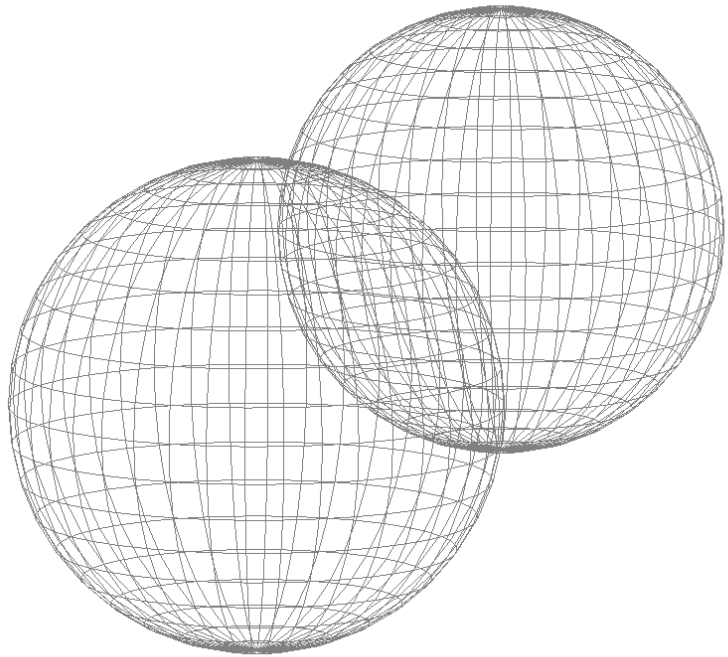
Algorithme du peintre - limites



Objets se cachant mutuellement

Gestion des parties visibles

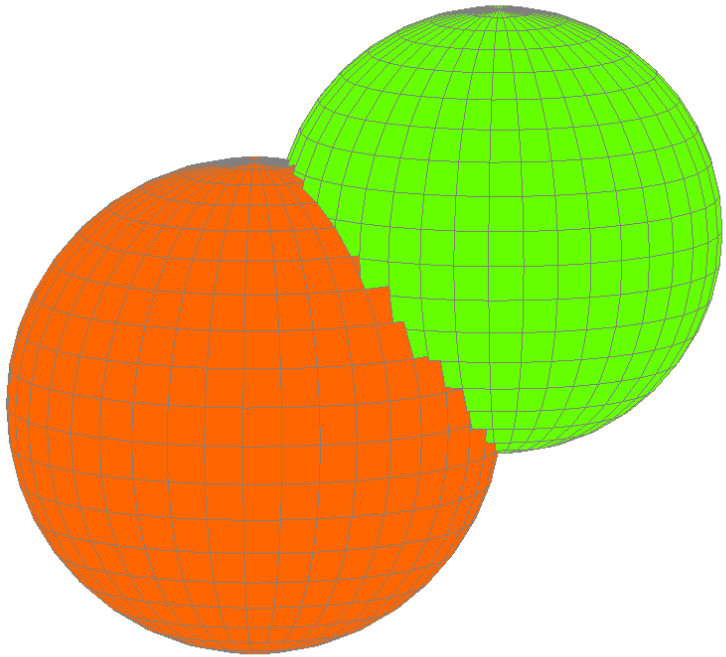
Algorithme du peintre - limites



Objets s'intersectant
Vue en mode filaire

Gestion des parties visibles

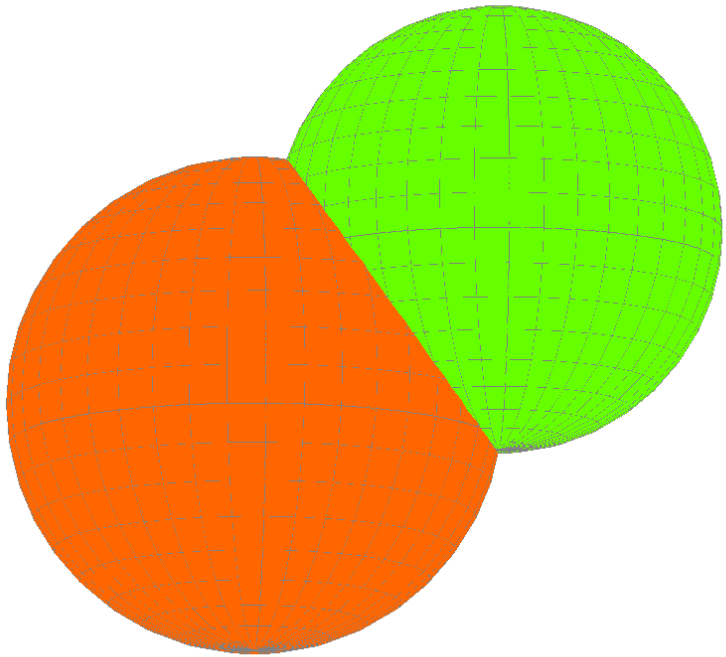
Algorithme du peintre - limites



Vue en face ombrée
algorithme du peintre

Gestion des parties visibles

Algorithme du peintre - limites



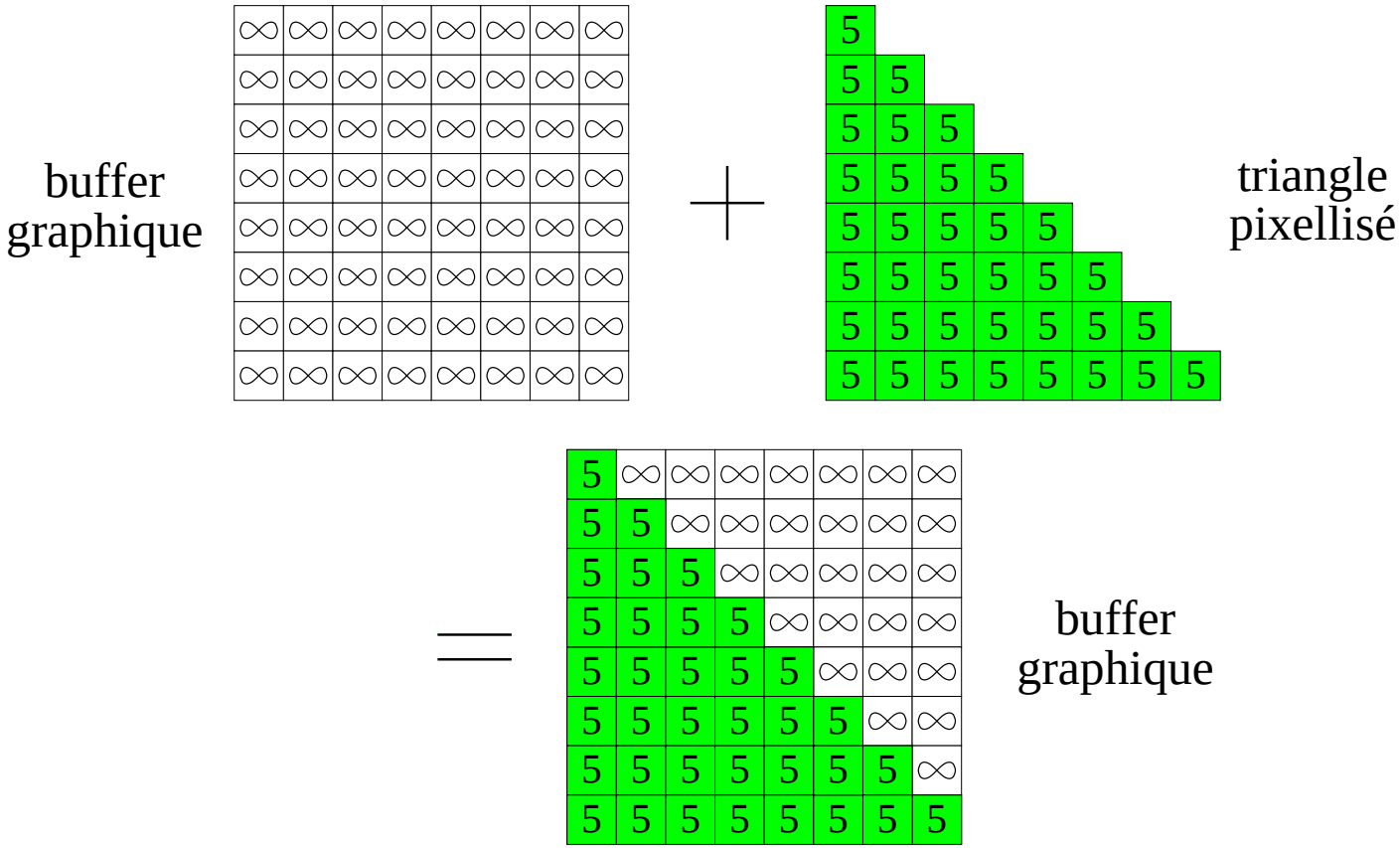
Vue en face ombrée
algorithme du z-buffer

Gestion des parties visibles

Algorithme du z-buffer

Gestion des parties visibles

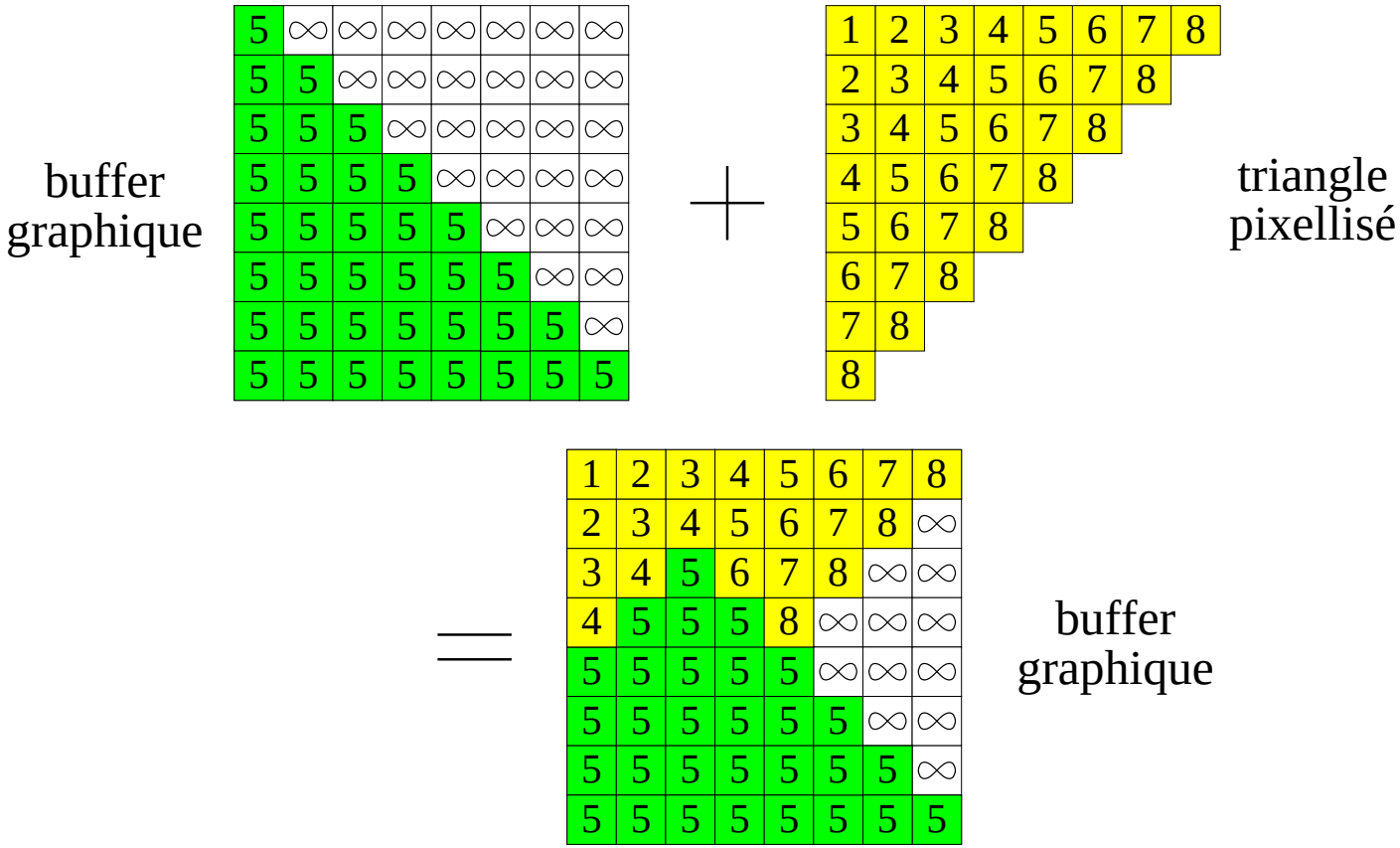
Algorithme du z-buffer



Dessin de deux triangles : le triangle 1

Gestion des parties visibles

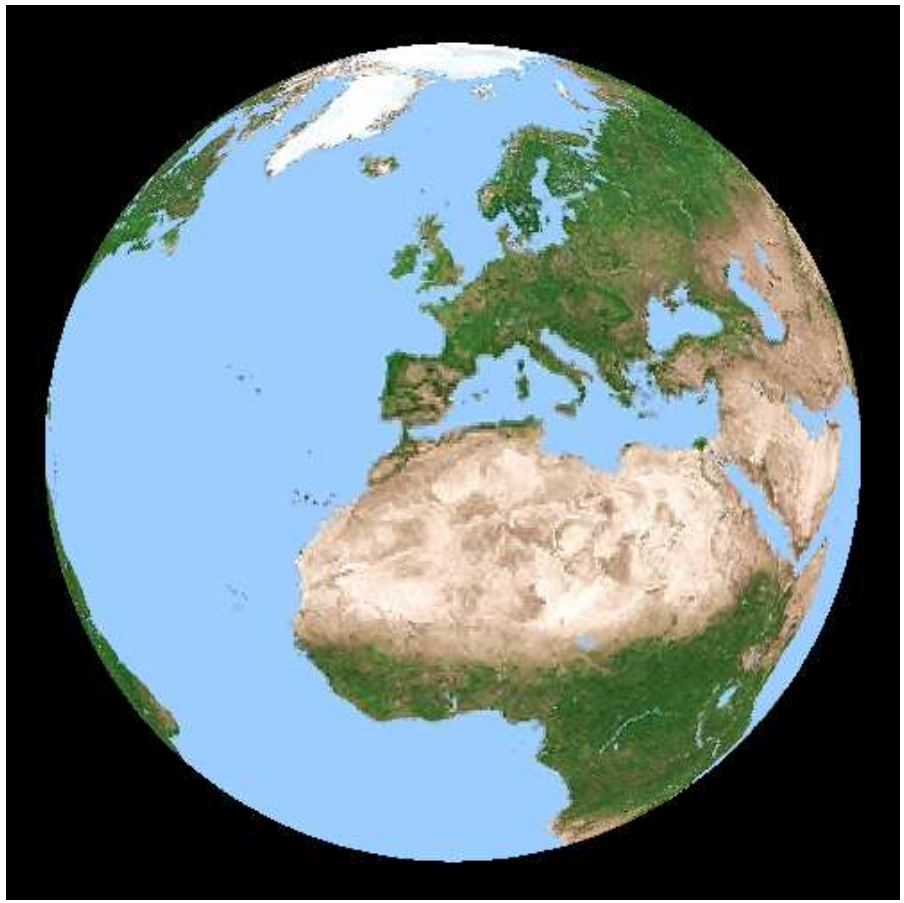
Algorithme du z-buffer



Dessin de deux triangles : le triangle 2

Texture

Texture



Exemples

Texture

Principe

Texture

Principe

1. créer une texture à partir d'une image,

Texture

Principe

1. créer une texture à partir d'une image,
2. appliquer (une partie de) l'image-texture sur une face (triangle/quadrangle/polygone) en mettant en correspondance chaque sommet de la face avec un point de la texture

Texture

Principe

1. créer une texture à partir d'une image,
2. appliquer (une partie de) l'image-texture sur une face (triangle/quadrangle/polygone) en mettant en correspondance chaque sommet de la face avec un point de la texture

Possibilité d'utiliser plusieurs images-textures dans une même scène

Texture

Coordonnées image-texture

Texture

Coordonnées image-texture

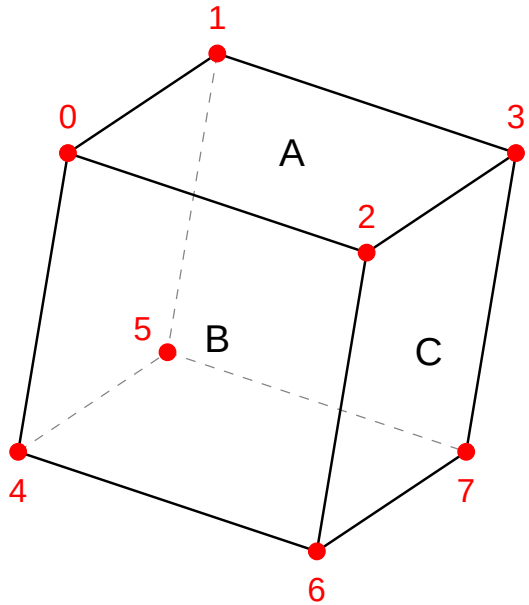
(X, Y) coordonnées d'un point de l'image-texture
avec $0 \leq X \leq 1$ et $0 \leq Y \leq 1$

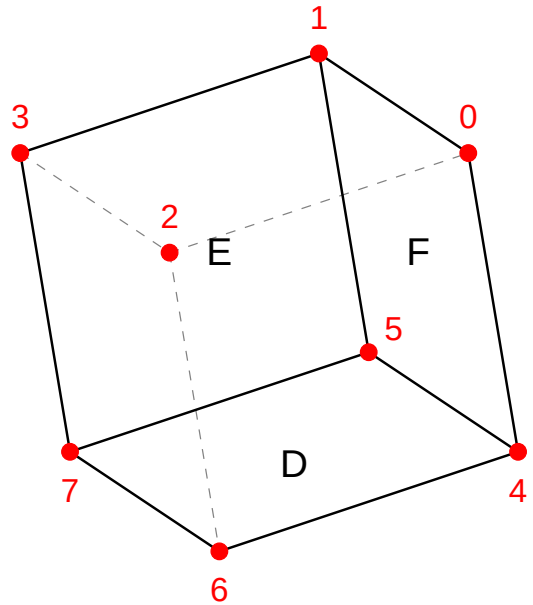
Texture

Exemple

Plaquer l'image-texture sur chaque face du cube

Texture





Cube

Texture

Exemple

```
void cube()
{
 // les 8 sommets
 GLdouble S[8][3] = {
  {0.0,0.0,0.0},{1.0,0.0,0.0},
  {0.0,1.0,0.0},{1.0,1.0,0.0},
  {0.0,0.0,1.0},{1.0,0.0,1.0},
  {0.0,1.0,1.0},{1.0,1.0,1.0}
 };

 // les 6 faces
 GLint F[6][4] = {
  {0,2,3,1},{6,2,0,4},{2,6,7,3},
  {4,5,7,6},{5,1,3,7},{0,1,5,4}
 };
```

Texture

Exemple

```
// objet cube
glBegin(GL_QUADS);
for (int i=0; i<6; i++)
{
 // les 4 sommets de la face i

 glVertex3dv(S[F[i][0]]);

 glVertex3dv(S[F[i][1]]);

 glVertex3dv(S[F[i][2]]);

 glVertex3dv(S[F[i][3]]);
}
glEnd();
}
```

Texture

Exemple

Correspondance image-quadrangle

Résultat du texturage

Texture

Exemple

```
// objet cube
glBegin(GL_QUADS);
for (int i=0; i<6; i++)
{
 // les 4 sommets de la face i
 glTexCoord2d(0.0,0.0);
 glVertex3dv(S[F[i][0]]);
 glTexCoord2d(0.0,1.0);
 glVertex3dv(S[F[i][1]]);
 glTexCoord2d(1.0,1.0);
 glVertex3dv(S[F[i][2]]);
 glTexCoord2d(1.0,0.0);
 glVertex3dv(S[F[i][3]]);
}
glEnd();
}
```

Texture

Exemple

Correspondance image-quadrangle

Résultat du texturage

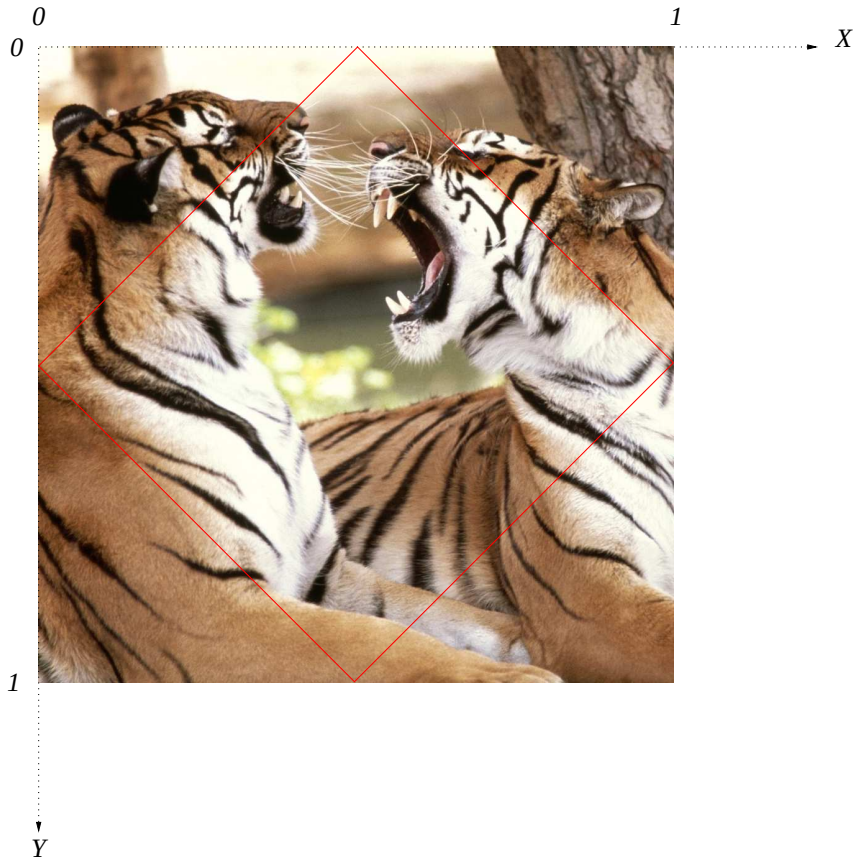
Texture

Exemple

```
// objet cube
glBegin(GL_QUADS);
for (int i=0; i<6; i++)
{
 // les 4 sommets de la face i
 glTexCoord2d(0.0,0.0);
 glVertex3dv(S[F[i][0]]);
 glTexCoord2d(0.0,0.5);
 glVertex3dv(S[F[i][1]]);
 glTexCoord2d(0.5,0.5);
 glVertex3dv(S[F[i][2]]);
 glTexCoord2d(0.5,0.0);
 glVertex3dv(S[F[i][3]]);
}
glEnd();
}
```

Texture

Exemple



Correspondance image-quadrangle

Résultat du texturage

Texture

Exemple

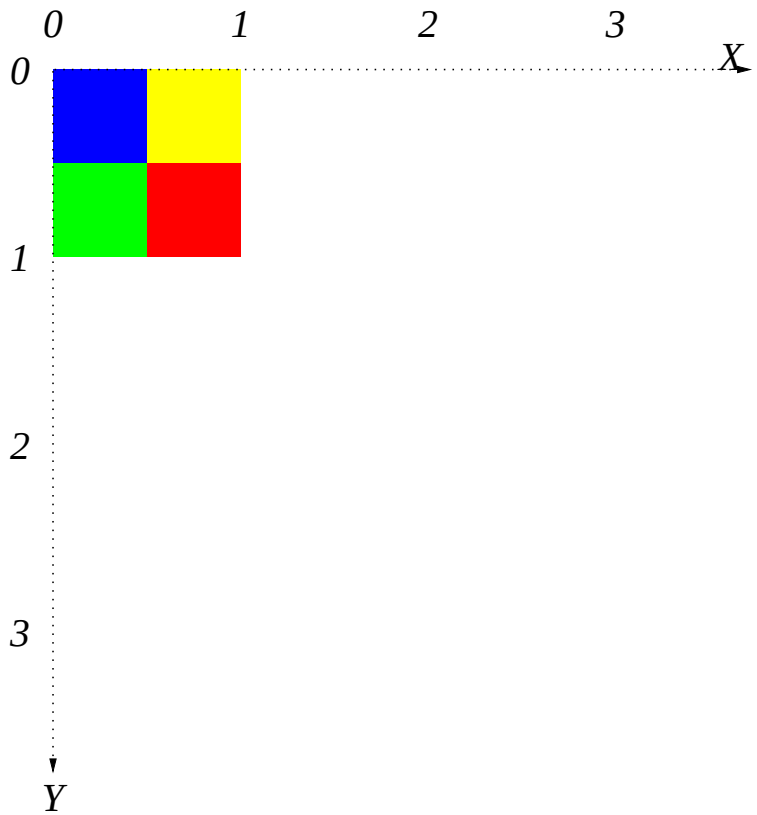
```
// objet cube
glBegin(GL_QUADS);
for (int i=0; i<6; i++)
{
 // les 4 sommets de la face i
 glTexCoord2d(0.5,0.0);
 glVertex3dv(S[F[i][0]]);
 glTexCoord2d(0.0,0.5);
 glVertex3dv(S[F[i][1]]);
 glTexCoord2d(0.5,1.0);
 glVertex3dv(S[F[i][2]]);
 glTexCoord2d(1.0,0.5);
 glVertex3dv(S[F[i][3]]);
}
glEnd();
}
```

Texture

Périodisation

Texture

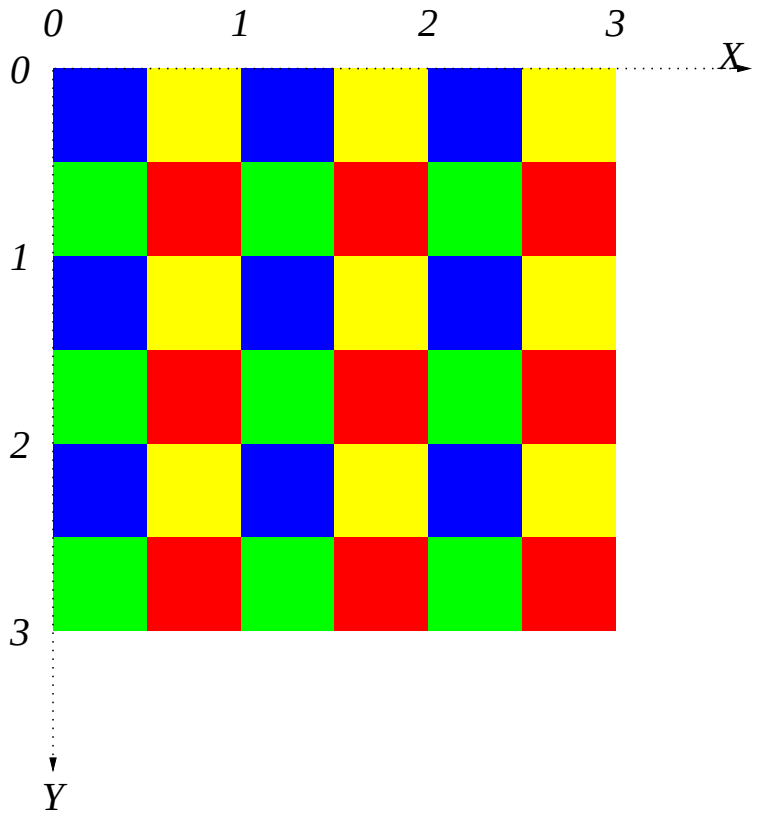
Périodisation



Texture initiale

Texture

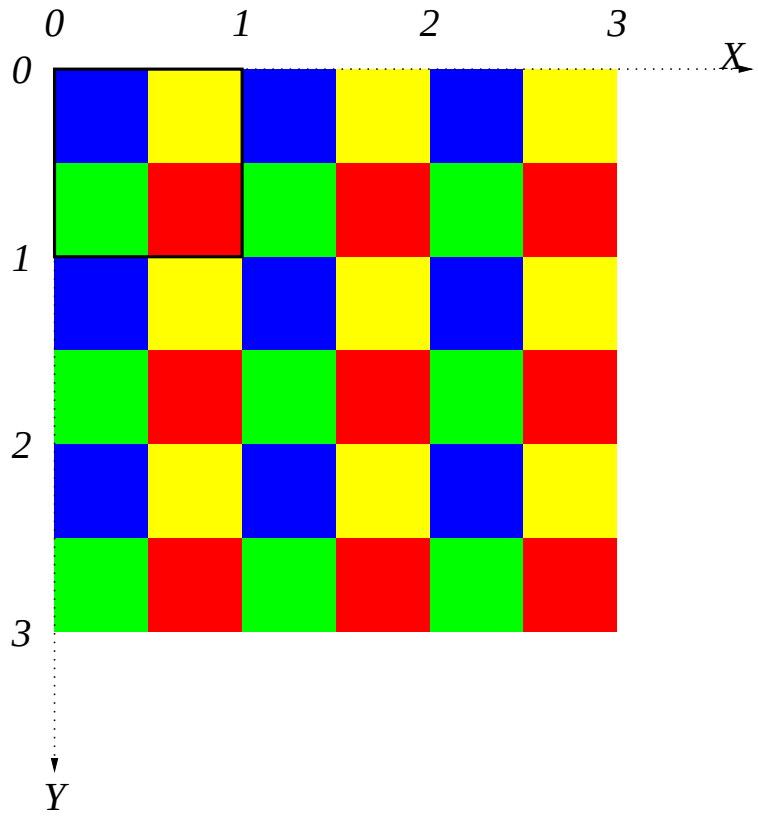
Périodisation



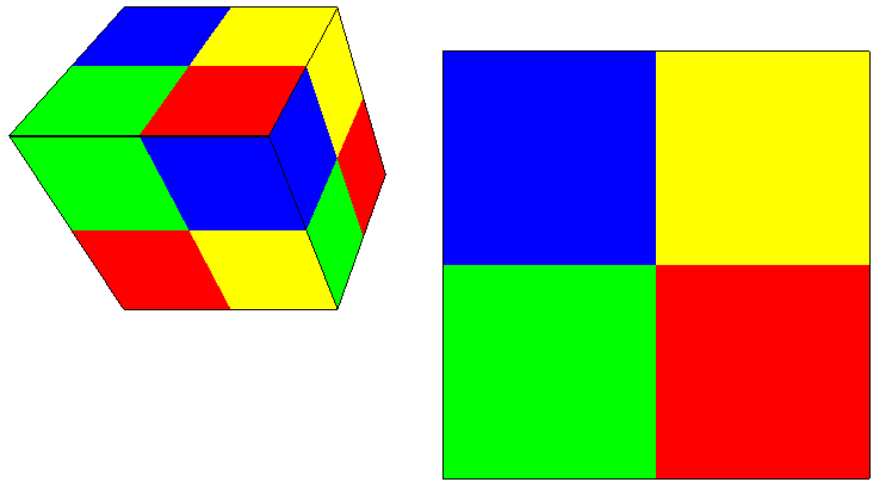
Texure *"périodisée"*

Texture

Périodisation



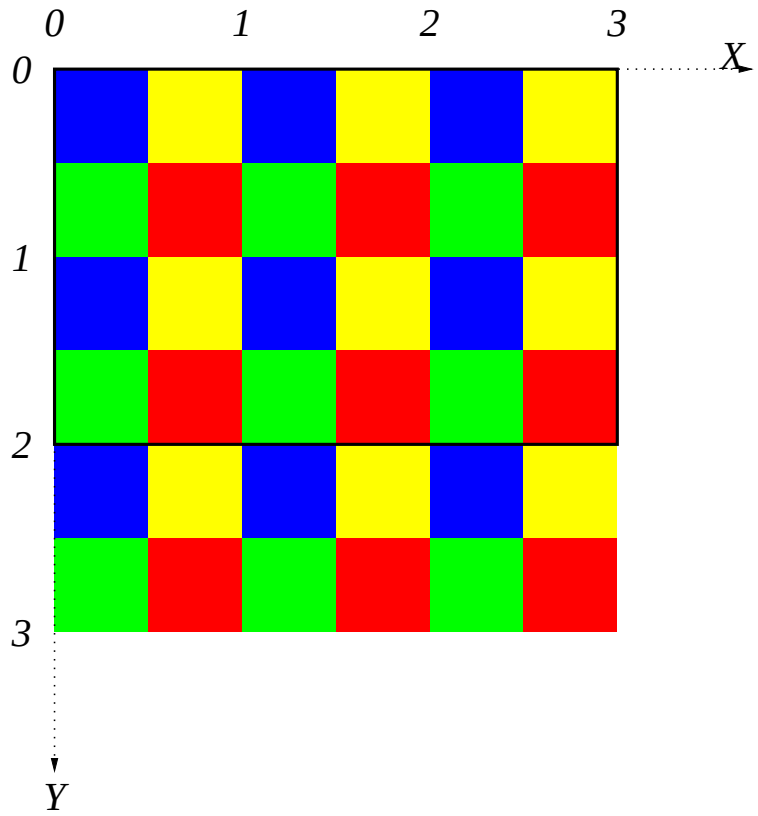
Correspondance image-quadrangle



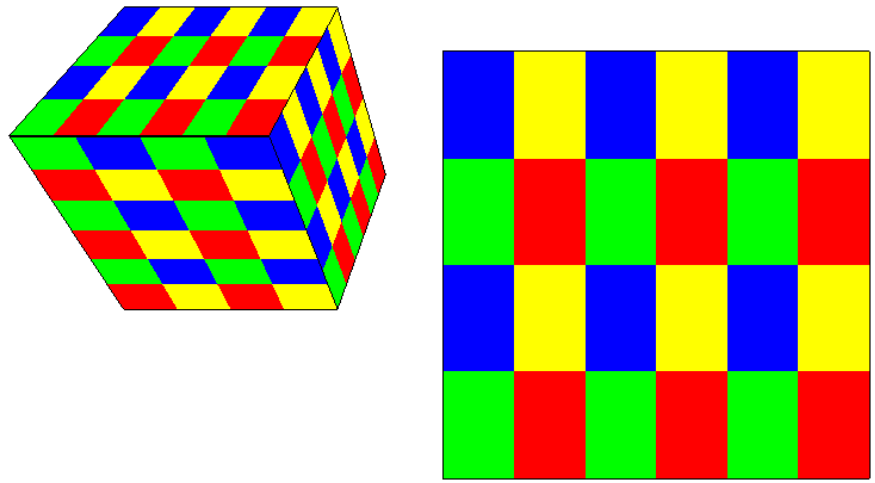
Résultat du texturage

Texture

Périodisation



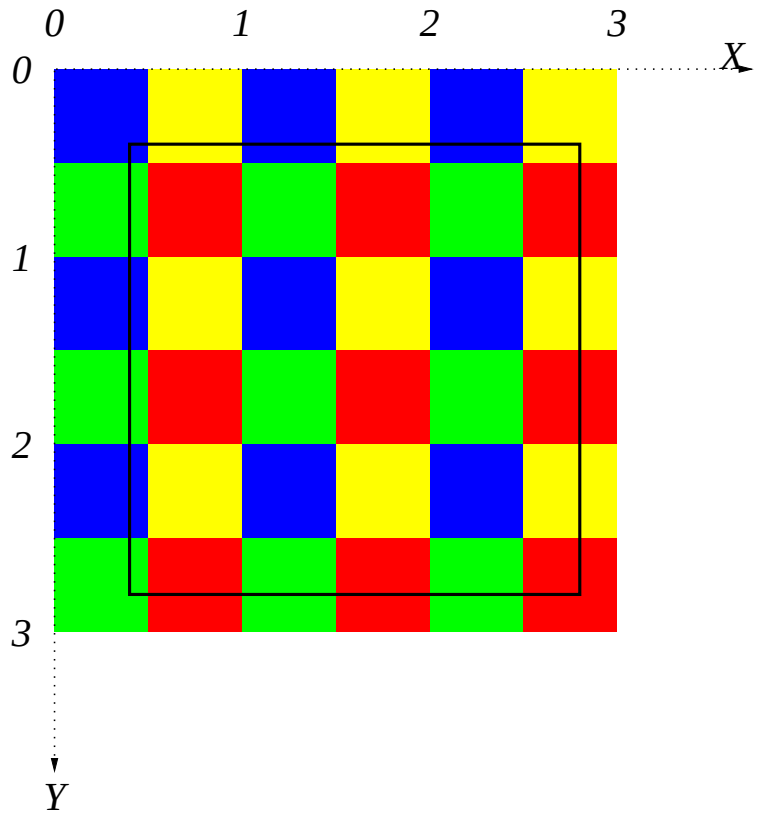
Correspondance image-quadrangle



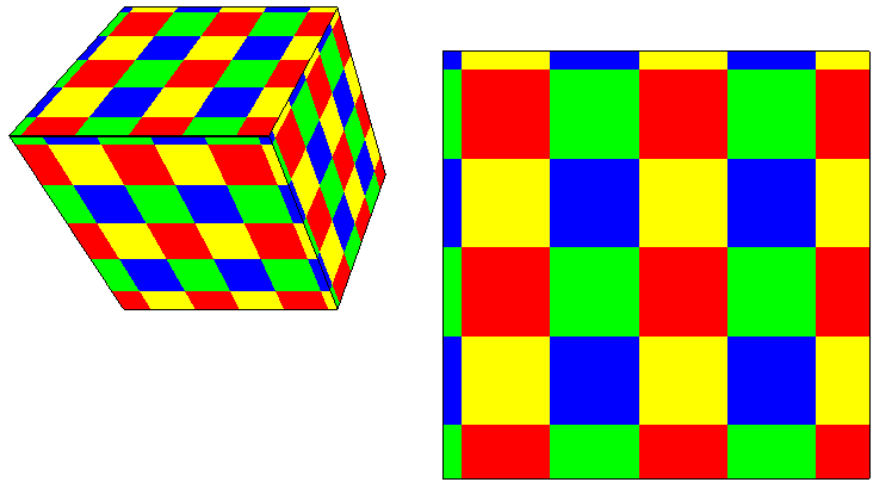
Résultat du texturage

Texture

Périodisation



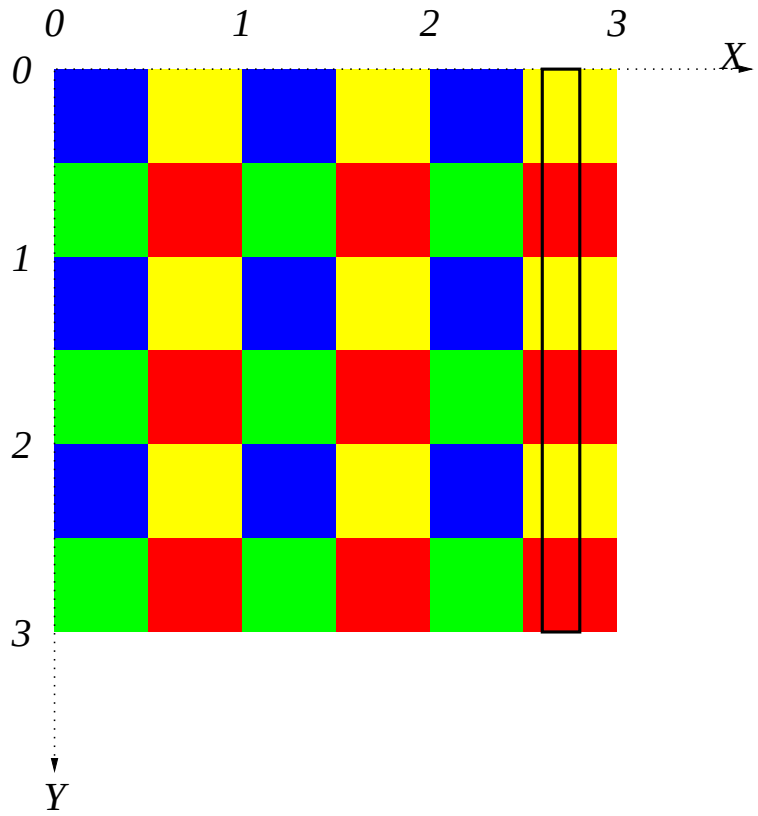
Correspondance image-quadrangle



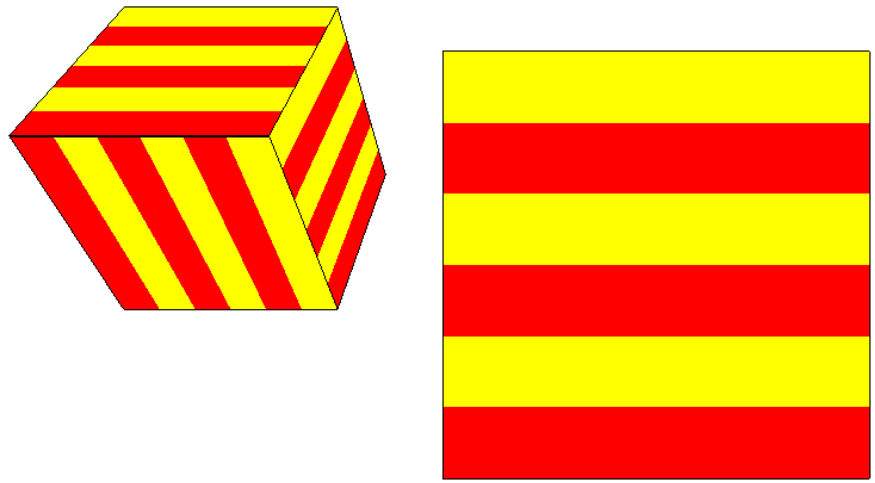
Résultat du texturage

Texture

Périodisation



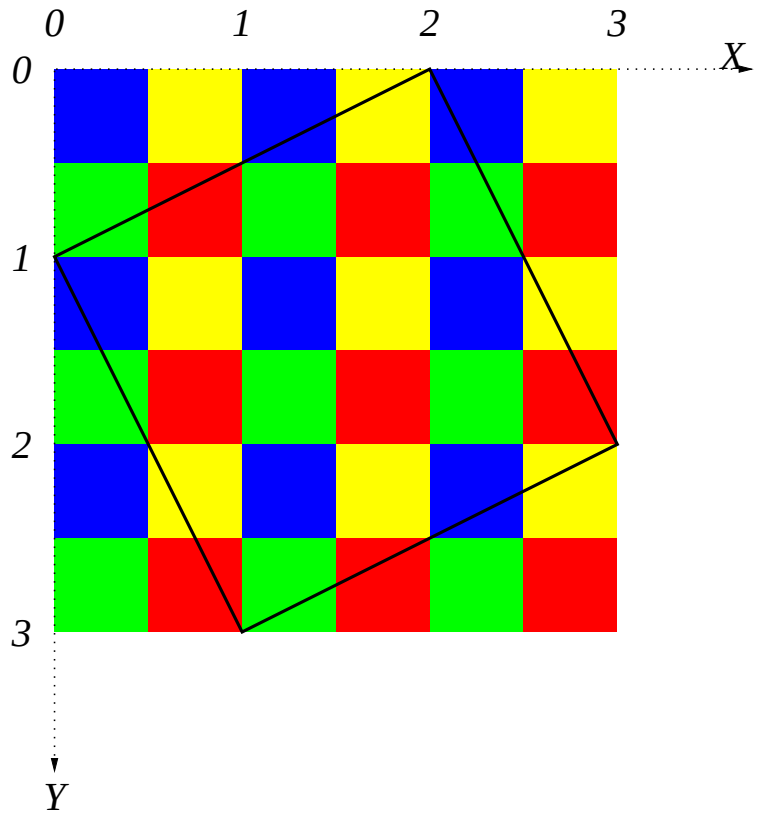
Correspondance image-quadrangle



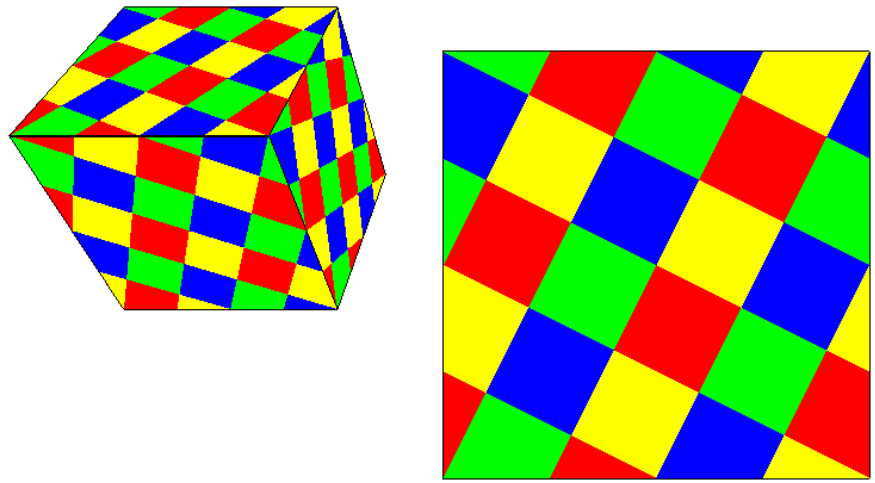
Résultat du texturage

Texture

Périodisation



Correspondance image-quadrangle



Résultat du texturage

Texture

Plaquage sur une surface

Texture

Plaquage sur une surface

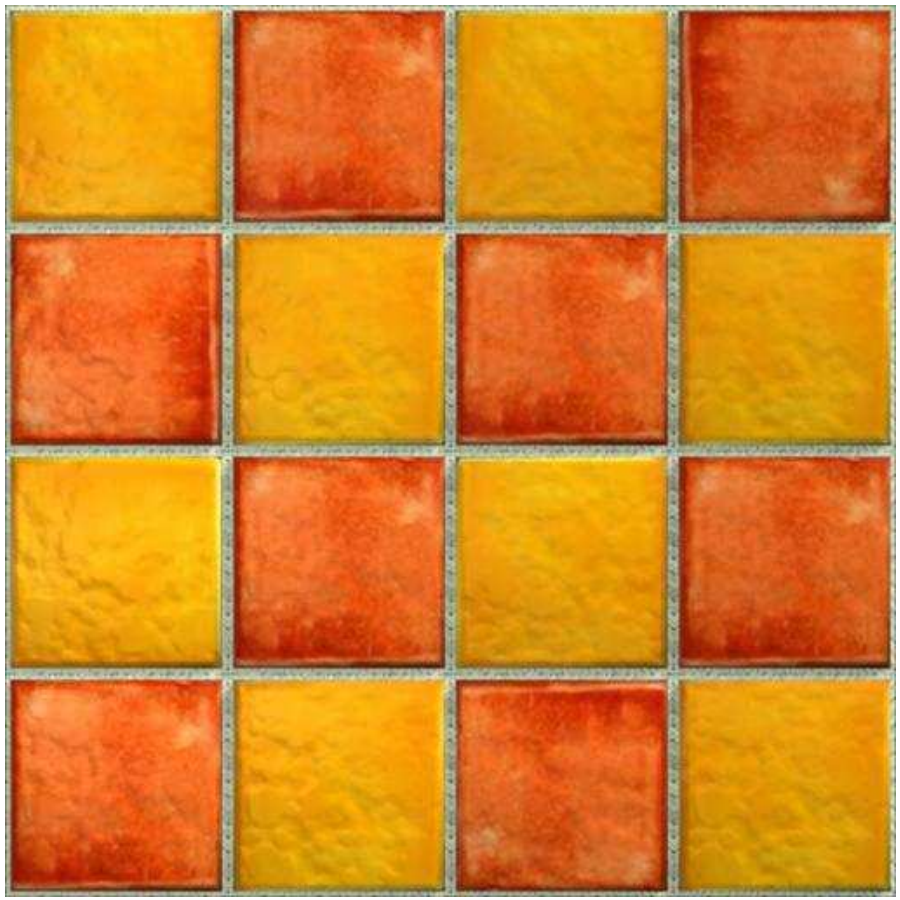
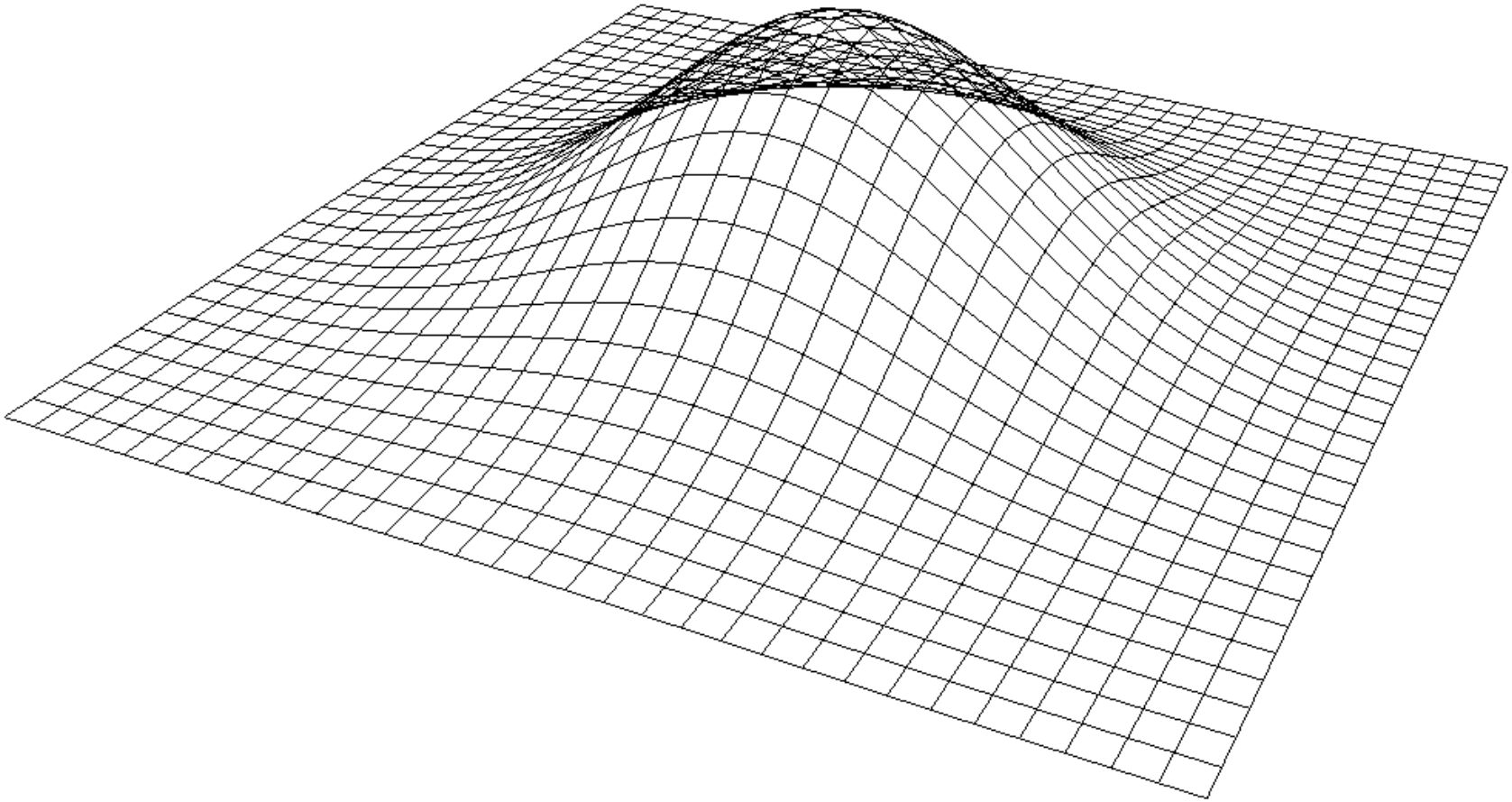


Image texture

Texture

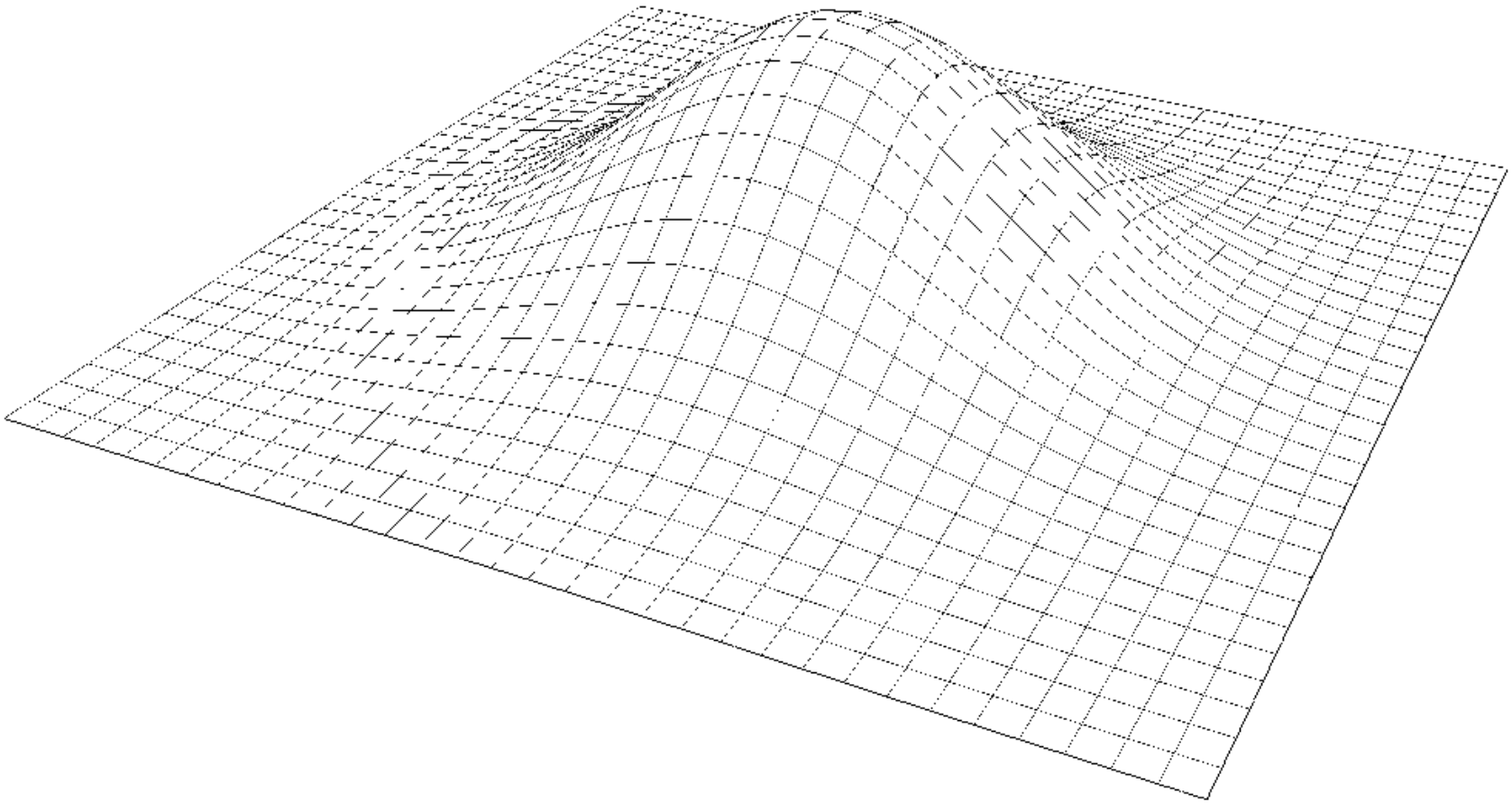
Plaquage sur une surface



Surface

Texture

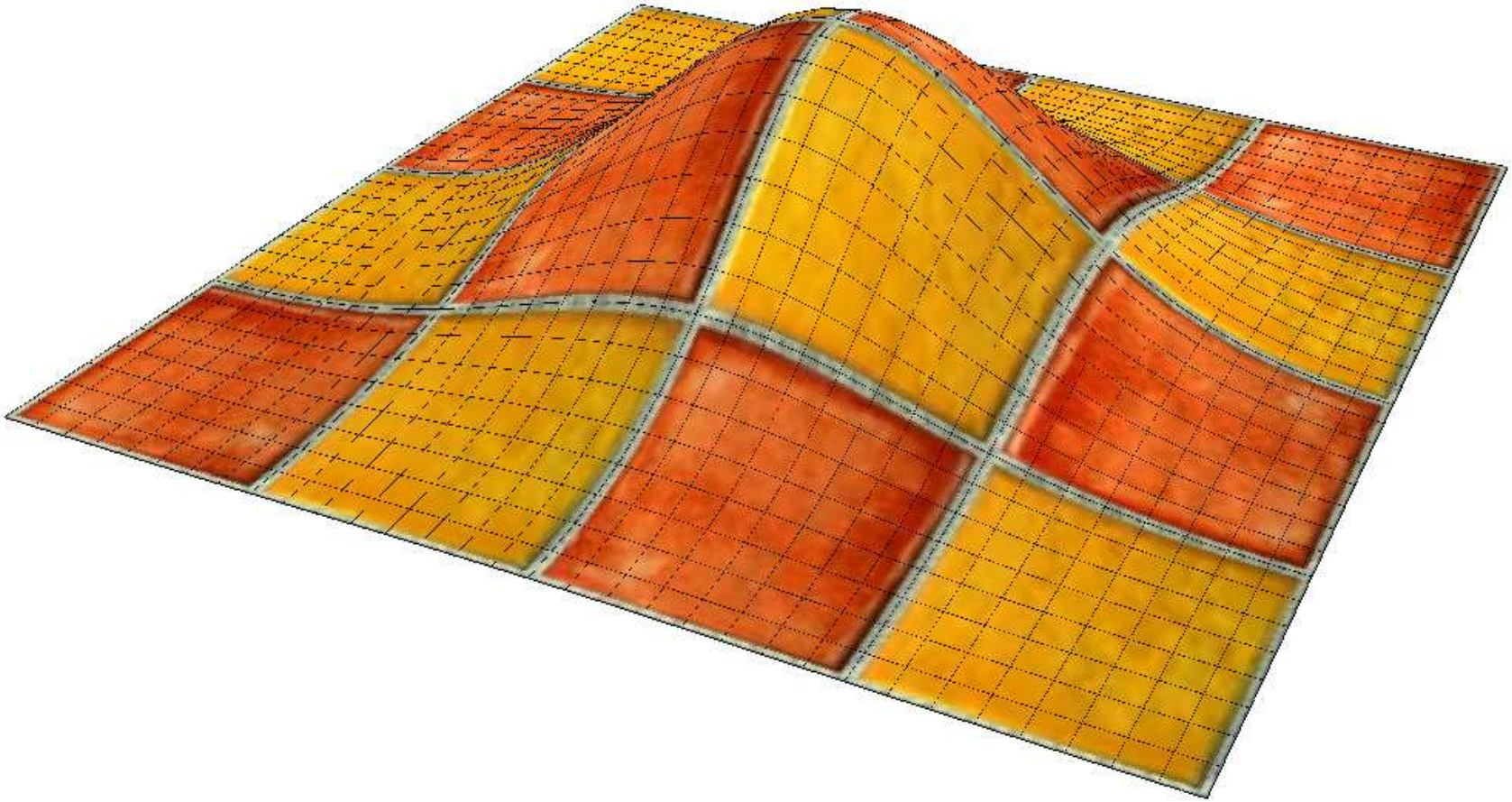
Plaquage sur une surface



Surface

Texture

Plaquage sur une surface



Surface texturée

Texture

Plaquage sur une surface

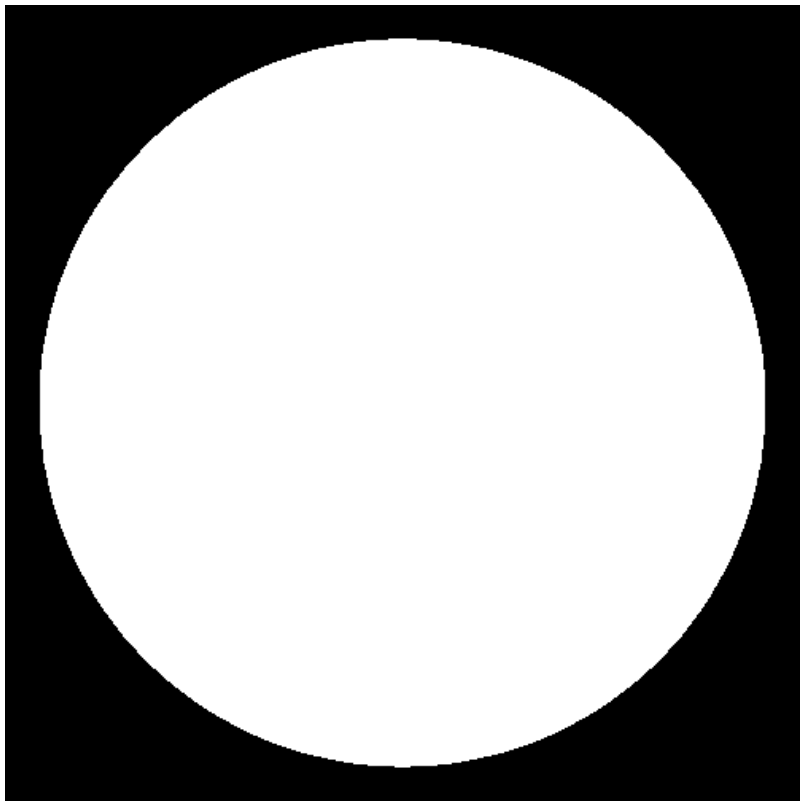
Surface texturée

Texture

Mélange couleur et texture

Texture

Mélange couleur et texture

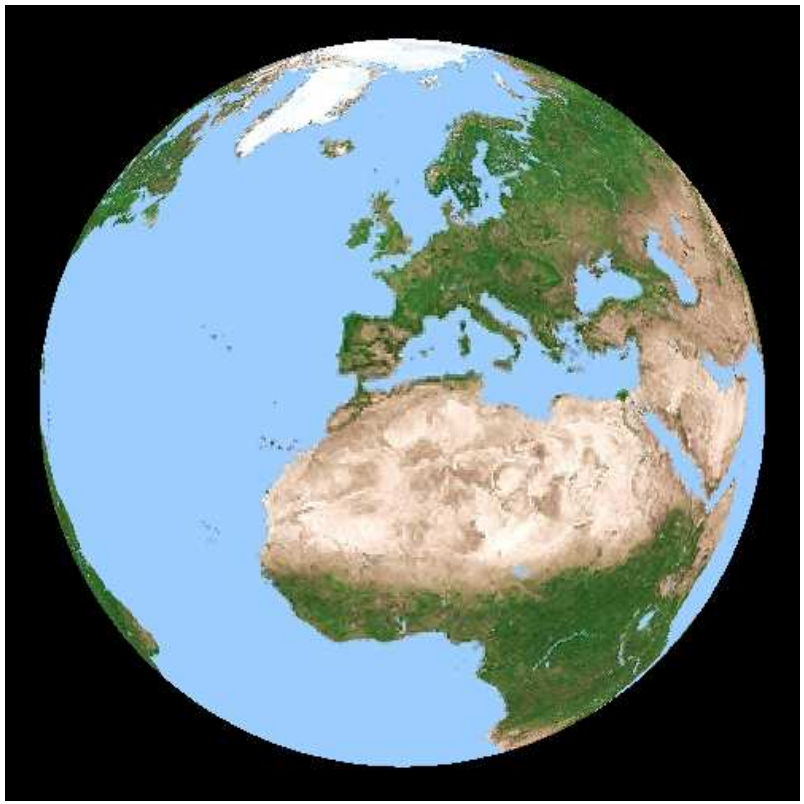


Couleur blanc sans texture

Dégradé sans texture

Texture

Mélange couleur et texture



Couleur blanc avec texture

Dégradé avec texture

Texture

Texture en OpenGL - création

```
// ----- définir l'image de la texture ----- //
int L = 3, H = 2;  // dimensions de l'image
//    noir    |    rouge     |     jaune     |
//    bleu    |    magenta   |     blanc     |
GLubyte ImageT = {
  0,  0,  0 , 255,  0,  0 , 255,255,  0 ,     0,0,0 ,
  0,  0,255 , 255,  0,255 , 255,255,255 ,     0,0,0
}

// Remarque : chaque ligne de l'image doit etre codée
// sur un nombre d'octets (GLubyte) multiple de 4
// ici, une ligne = 3 pixels RGB = 9 octets
// --> on complète le codage de chaque ligne avec 3
// octets supplémentaires pour avoir 12 = 3 x 4 octets
```

Texture

Texture en OpenGL - création

```
/* active le texturing */
glEnable(GL_TEXTURE_2D);
/* genere un numero de texture */
glGenTextures(1,&Num);
/* selectionne ce numero */
glBindTexture(GL_TEXTURE_2D,Num);
```

Texture

Texture en OpenGL - création

```
/* creation de la texture à partir du tableau ImageT */
glTexImage2D (
  GL_TEXTURE_2D,     /* Type : texture 2D */
  0,                 /* Mipmap : aucun */
  3,                 /* ModeRGB (3 composantes) */
  L,                 /* Largeur */
  H,                 /* Hauteur */
  0,                 /* Largeur du bord : 0 */
  GL_RGB,            /* Format : RGB */
  GL_UNSIGNED_BYTE,  /* composante de type GLubyte */
  ImageT             /* buffer de l'image de texture */
);
```

Texture

Texture en OpenGL - création

```
/* definit le type de texture (2D)
   et comment elle est appliquee */
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER,
                GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER,
                GL_NEAREST);
```

Texture

Texture en OpenGL - utilisation d'une texture

```
void dessin()
{
  glEnable(GL_TEXTURE_2D); /* active le texturing */
  /* selectionne la texture de numero Num */
  glBindTexture(GL_TEXTURE_2D,Num);
  ...

  /* associer la coordonnée texture (tx,ty) et */
  /* la couleur (Cr,Cg,Cb) au point (x,y,z)    */
  glTexCoord2d(tx,ty);
  glColor3d(Cr,Cg,Cb);
  glVertex3d(x,y,z);
  ...
  glDisable(GL_TEXTURE_2D); /* désactive le texturing */
}
```