Infographie

Jean Fruitet
Jean.Fruitet@univ-mlv.fr

Centre d'InfoGraphie
Université de Marne-La-Vallée

1. INTRODUCTION

L'infographie (informatique graphique) a pour objet :

- l'analyse et le traitement d'images (amélioration d'images ; reconnaissance de formes ; cartographie thématique, ...)

- la synthèse d'image (affichage de données sous forme graphique, calcul et restitution d'images réaliste ou symbolique, visualisation de données scientifiques) sur écran cathodique (CRT) ou sur imprimante par procédé informatique.

Pour l'analyse, voir le cours de traitement d'image.

Nous consacrons cette seconde partie du cours de géométrie algorithmique à un survol rapide de outils (méthodologie / structures de données / algorithmes / interfaces) pour la création de programmes graphiques (2D et 3D).

1.1. Les étapes de la synthèse d'image.

Pour produire l'image informatique d'un phénomène —mathématique, physique, organisationnel, etc.— il est nécessaire de passer par différentes étapes :

MODELISER LE PHENOMENE
- sous forme de nombres / relations / symboles
- lois de comportement au cours du temps

AFFICHER CE MODELE SUR ECRAN EN DEUX DIMENSIONS
- par composition de primitives graphiques
  - point, segment, courbe, surface, volume
  - épaisseur, trame, couleur

INTERFACER L'AFFICHAGE AVEC L'UTILISATEUR
- l'interaction avec le programme d'affichage permet à l'utilisateur
- de modifier les caractéristiques d'affichage (changer de point de vue)
- et/ou de modifier le modèle

ANIMER LE MODELE
- la sensation d'animation est obtenue
- en modifiant le modèle et son affichage 25 fois par seconde...

1.2. L'univers et son modèle.

L'UNIVERS qui est sous-jacent à tout phénomène a des caractéristiques difficiles à représenter sur ordinateur :
- infini
- continu
- complexe / ambigü / flou
- nombreuses dimensions, dont la dimension temporelle

Un MODELE d'univers consiste en une *simplification mathématique*

On distingue différents modèles
- symbolique : langages, pictogrammes, ...
- numérique : champs de scalaires, fonctions numériques
- relationnel : schémas entités/associations et tables relationnelles
- géométrique : topologie, métrique, dimension (1D, 2D, 3D, ...)

physique : statique / cinématique (mouvement) / dynamique (force)
objet : classe, héritage, ...
etc.

Sans modélisation préalable, pas de représentation informatique...

1.3. Le modèle et sa représentation.

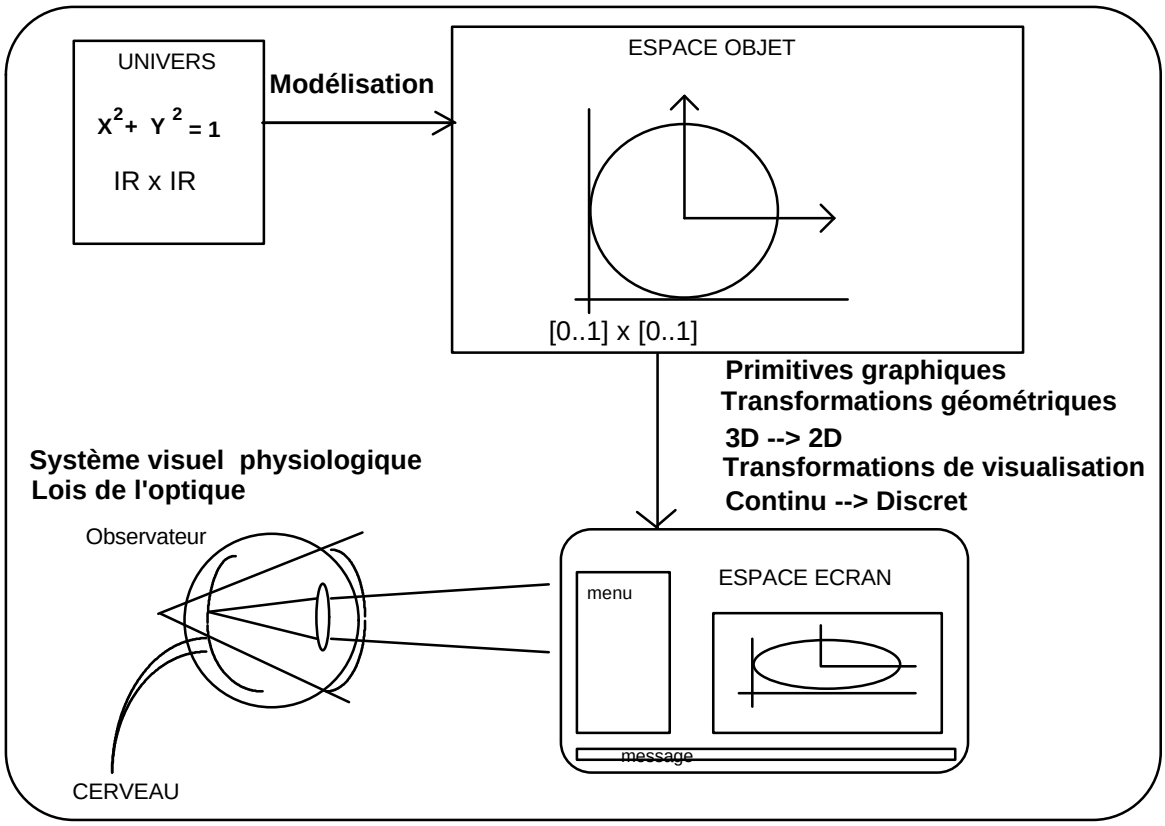
Avant tout affichage sur écran, il faut traduire/décomposer le modèle en objets graphiques (*primitives d'affichage*) et le recombiner à l'aide d'opérateurs (*transformations spatiales, projections, fenêtrage...*) dans la *mémoire d'affichage* de l'ordinateur.

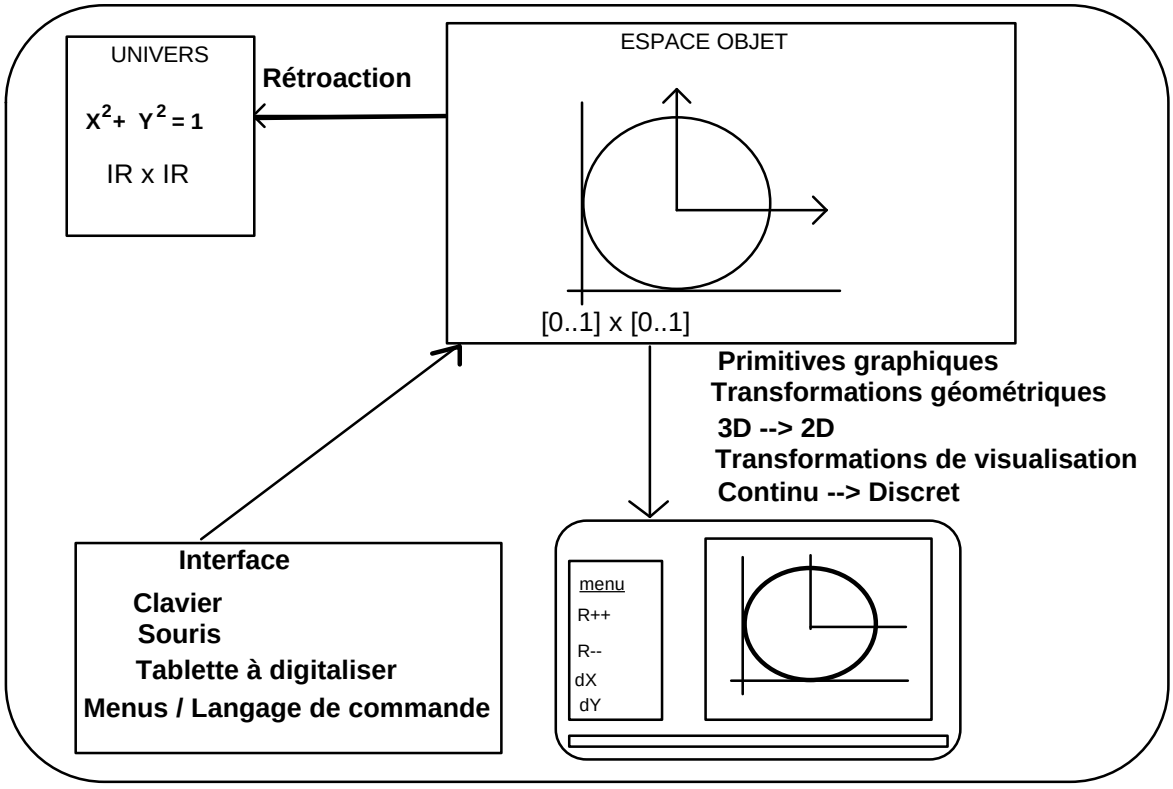
Les primitives d'affichage

Dimension	Primitive	Représentation	Structure de données	Simplexe
1D	Point Intervalle	Semi de points Intervalle	Liste de points Listes d'intervalles	Segment
2D continu (espace euclidien)	Point 2D Segment Polyligne Arc Courbe Triangle Boîte Polygone	- Nuage de points - Filaire - Polygones remplis	Ensembles Listes Arbres (Quad-Tree) Fonctions - explicites - implicites paramétriques Coniques Splines 2D Fractales	Triangle
2D raster (espace discrétisé)	Pixel Tache (connexité 4, 8)			
3D continu	Point 3D Courbe gauche Surface plane Surface gauche Facettes Polyédres Volumes : Sphère, cube, cylindre, cône	- Filaire - Par facettes polygonales remplies. - Volumique	B-REP: Liste de - Sommets - Arêtes - Faces Struct. hiérarchique - Octree - Arbre CSG MNT Quadriques Splines 3D Fractales	Tétraèdre
3D raster	Voxel Champ scalaire			

1.4. Le pipe-line de visualisation.

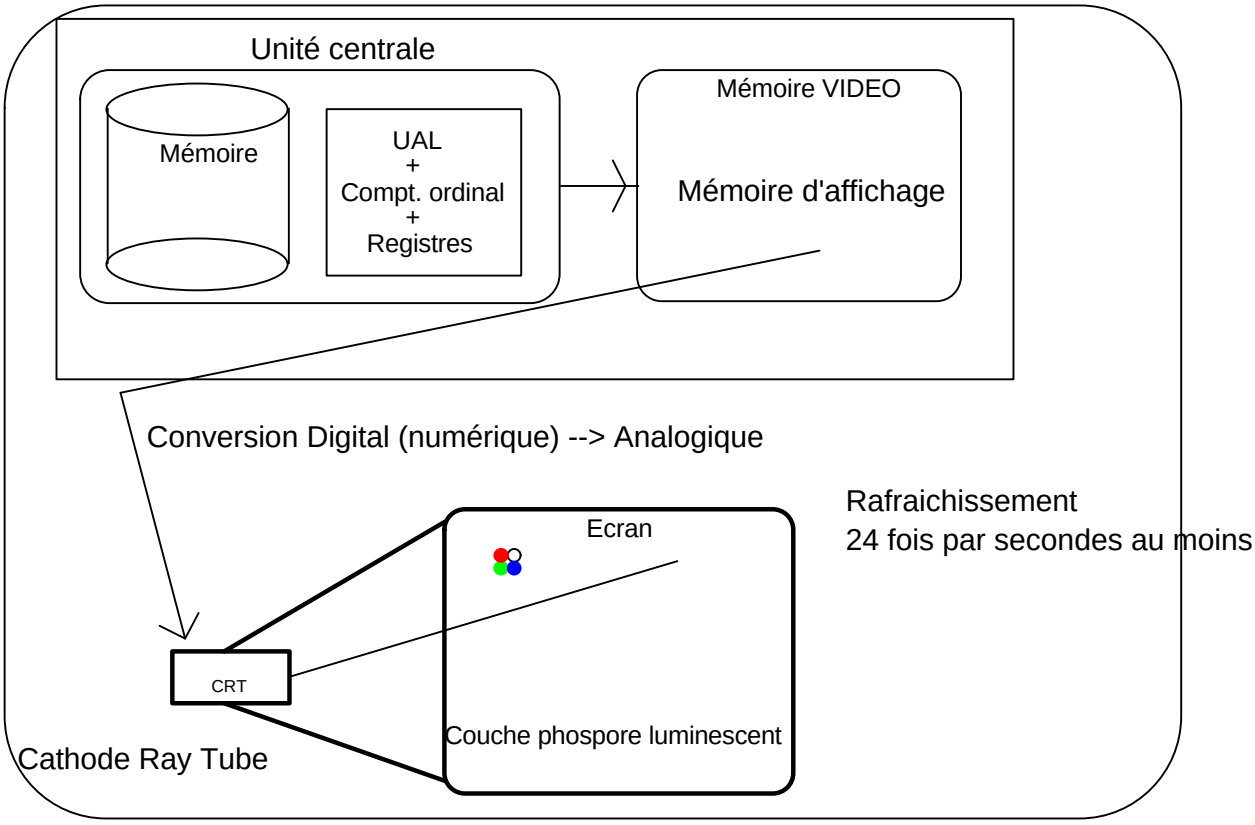
On désigne sous ce terme la succession des étapes de l'affichage


UNIVERS
$X^2 + Y^2 = 1$
IR x IR
Modélisation
ESPACE OBJET
[0..1] x [0..1]
Primitives graphiques
Transformations géométriques
3D --> 2D
Transformations de visualisation
Continu --> Discret
Système visuel physiologique
Lois de l'optique
Observateur
CERVEAU
menu
ESPACE ECRAN
message


UNIVERS
$X^2 + Y^2 = 1$
IR x IR
Rétroaction
ESPACE OBJET
[0..1] x [0..1]
Primitives graphiques
Transformations géométriques
3D --> 2D
Transformations de visualisation
Continu --> Discret
Interface
Clavier
Souris
Tablette à digitaliser
Menus / Langage de commande
menu
R++
R--
dX
dY

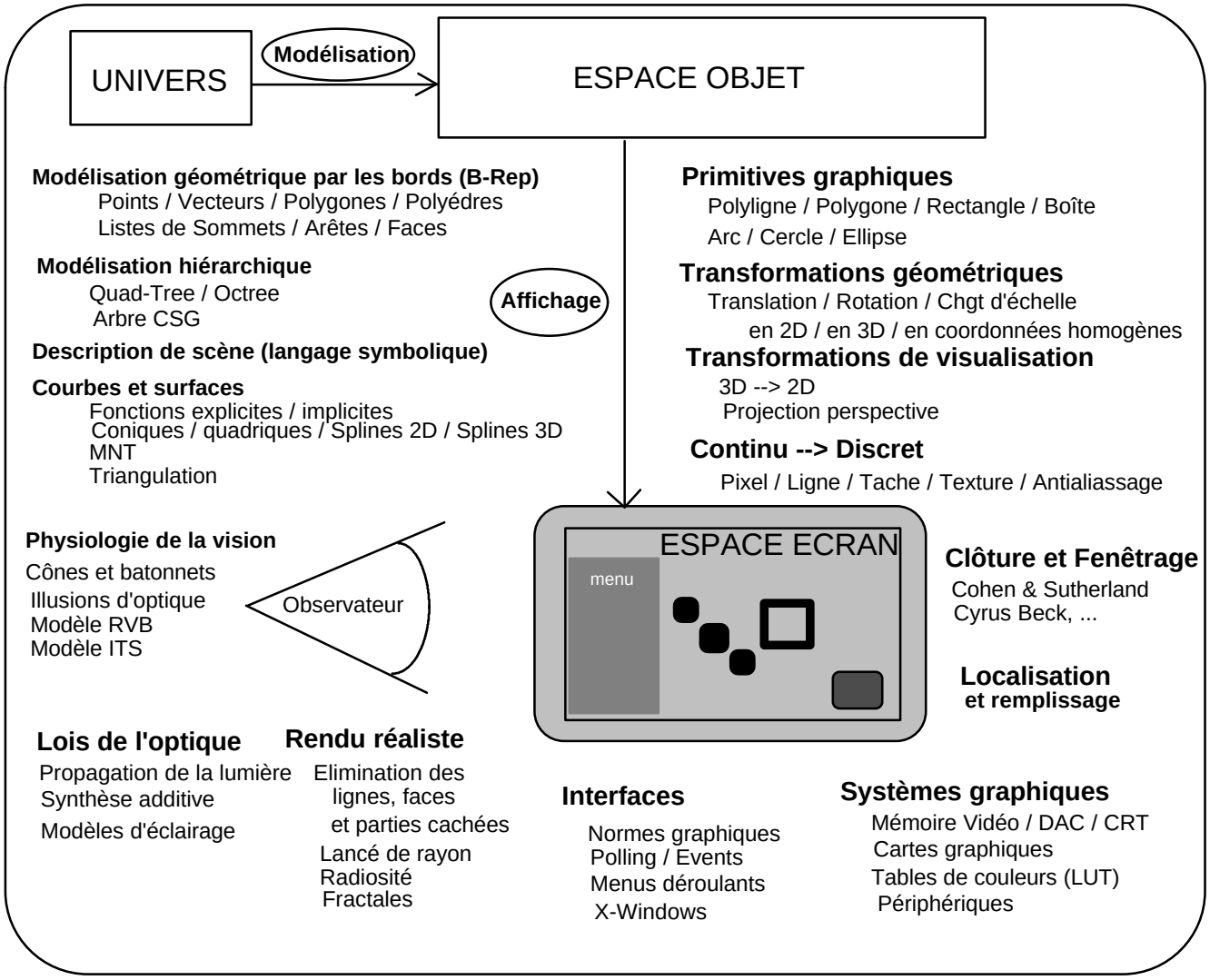
1.5. Conversion numérique / analogique.

Ce schéma fonctionnel s'appuie sur un schéma technologique



Pour simuler l'animation, il faut afficher 24 images par secondes en respectant un référentiel culturel (métaphore de la caméra), un modèle physique simplifié (lois de l'optique et de l'électromagnétisme) et physiologique (vision des couleurs).

1.6. Thèmes de l'infographie.



1.7. Un exemple d'application 2D

Affichage de la courbe y = cos(x).

Ce premier exemple va nous permettre de parcourir les étapes de l'affichage d'une fonction mathématique et de définir au passage les outils pour la programmation graphique en Turbo C sur PC.

Univers

IR --> IR
x --> y = cosinus(x).

Modèle géométrique :

Fonction explicite y=cos(x) sur l'intervalle $]-\infty,+\infty[$
Fonction paramétrique x = t; y = cos(t) sur l'intervalle $]-\infty,+\infty[$
Vu la parité et la périodicité de la fonction, on peut :
se ramener à l'intervalle $[-p,+p[$
donner la fonction en **extension**, en prenant par exemple N=12 échantillons sur l'intervalle $[-p,+p[$

Indice	*0*	*1*	*2*	*...*	*i*	*...*	*N = 12*
x	$-\Pi$	$-5\Pi/6$	$-\Pi/3$		$(2\Pi / N)*i -\Pi$		$+\Pi$
y=cos(x)	-1	$-\sqrt{3} / 2$	-1/2				-1

Passage de l'espace objet à l'espace image

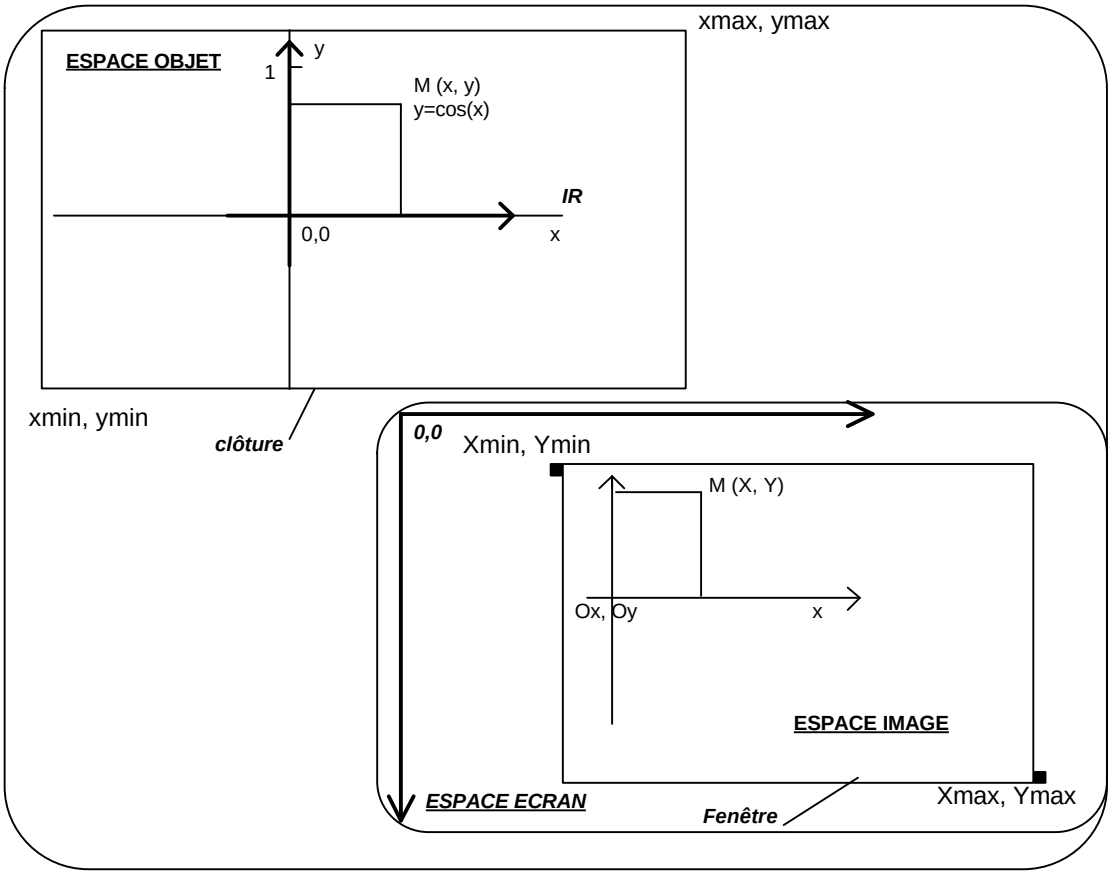
L'espace objet (ou réel) est infini, continu, euclidien, en N-dimensions.
L'espace écran est fini, discret, en 2 dimensions.

Dans l'espace objet, on peut choisir un référentiel (repère) orthogonal, normé, direct (anti-horaire 2D, ou trigonométrique) et un système de coordonnées cartésiennes pour représenter les fonctions mathématiques et leur graphe.

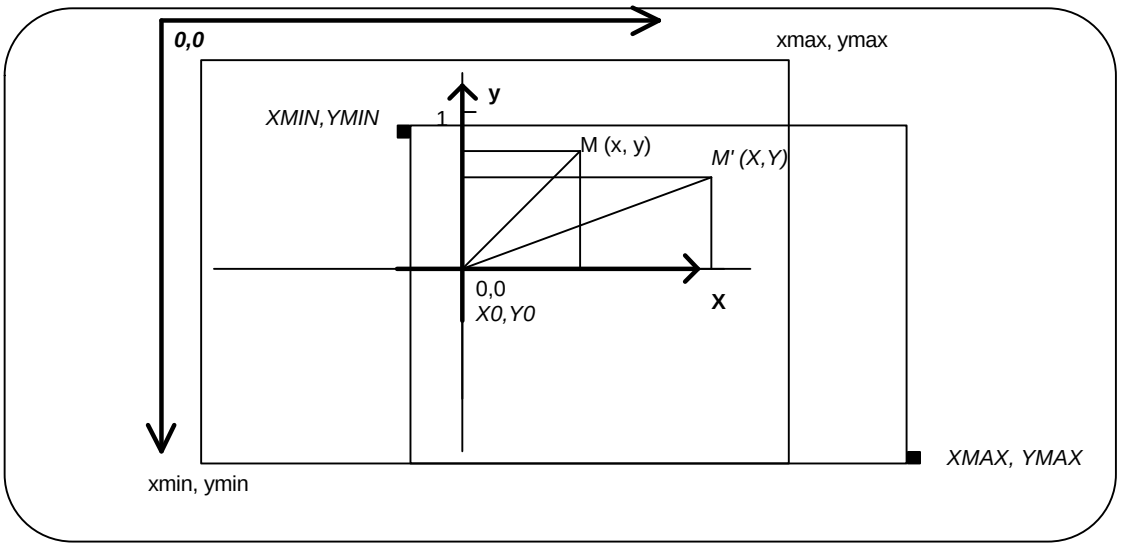
Par contre l'espace écran est défini (pour des questions technologiques... et culturelles) selon un référentiel horaire, avec origine dans le coin supérieur gauche de l'écran. Les coordonnées entières d'écran sont données en (colonne, ligne).

Définition d'écran : c'est le nombre de lignes et de colonnes adressables.
Sur un écran PC en mode graphique standard VGA 16 couleurs :
Nombre de colonnes par ligne 640 (de 0 à MaxX=639)
Nombre de lignes par page écran 480 (de 0 à MaxY=479)
Si la forme du pixel (picture element) n'est pas carrée les cercles sont affichés comme des ellipses et les carrés comme des rectangles...
Il est donc nécessaire :
- de calculer par avance les coordonnées des points (x,y=f(x)) pour toutes les valeurs de l'intervalle de définition
- d'initialiser le mode d'affichage graphique (caractéristiques d'écran, clôture et fenêtrage)
- d'effectuer le passage de l'espace objet à l'espace image, qui peut être par exemple défini en coordonnées flottantes normalisées [0,1] x [0,1]
- d'effectuer le passage de l'espace image à l'espace écran (changement de repère)
- pour chaque couple de coordonnées écran, d'appeler les primitives d'affichage.

Et bien sûr si on ne dispose que de primitives point ou ligne, il faut décomposer le graphe en une succession de segments.



Passage de l'espace objet à l'espace image



Dans l'espace objet dans l'espace image

sur l'axe des X $$\frac{(x - x\min)}{(x\max - x\min)} = \frac{(X - XMIN)}{(XMAX - XMIN}$$

sur l'axe des Y $$\frac{(y - y\min)}{(y\max - y\min)} = \frac{(Y - YMIN)}{(YMAX - YMIN}$$

soit $X = (x - x\min)(XMAX - XMIN) / (x\max - x\min) + XMIN$

et $Y = (y - y\min)(YMAX - YMIN) / (y\max - y\min) + YMIN$

Il faut ensuite passer dans l'espace écran, c'est-à-dire inverser l'orientation de l'axe des Y et en coordonnées entières !

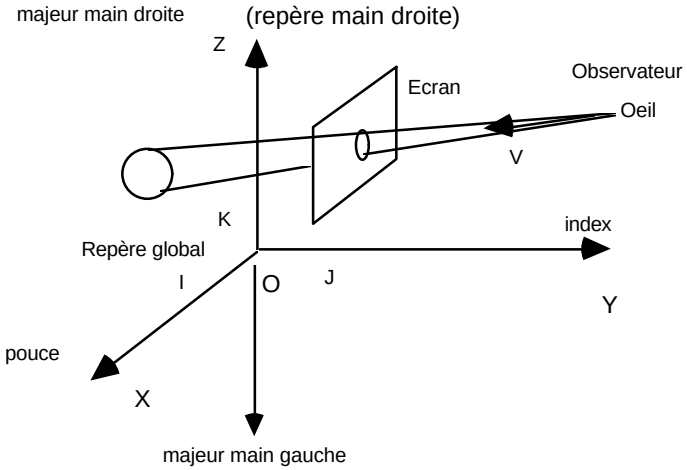
soit $Xécran = \lfloor (x - x\min)(XMAX - XMIN) / (x\max - x\min) + \rfloor XMIN$

et $Yécran = \mathrm{MaxY} - \lfloor (y - y\min)(YMAX - YMIN) / (y\max - y\min) + YMIN \rfloor$

2. NOTIONS GEOMETRIQUES

2.1. Notion de repère

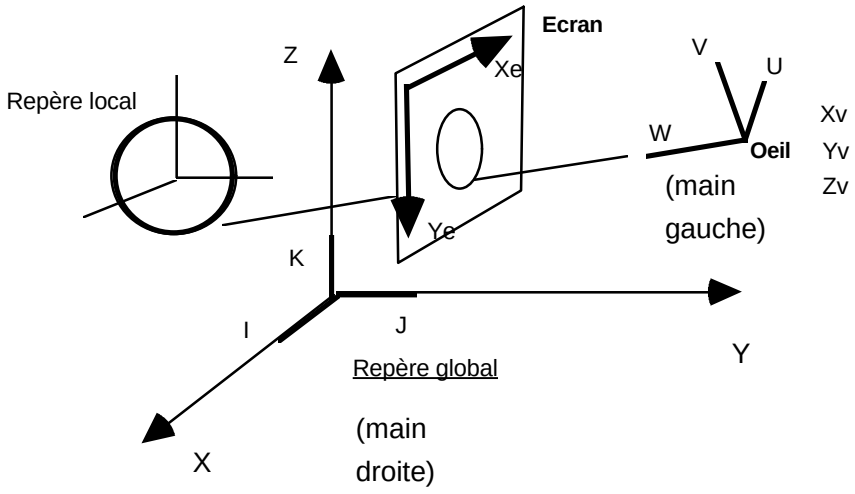
Une scène est une collection d'objets en 3 dimensions. Le repère associé à la scène est le repère global
Le repère mathématique usuel est un repère main droite (pouce -> X ; index --> Y ; majeur -> Z). En synthèse d'image on lui préfère souvent un repère main gauche avec les mêmes conventions (pouce -> X ; index --> Y ; majeur -> Z), pour lequel l'axe Z pointe dans le sens opposé du repère main droite.



2.2. Visualisation

L'opération de visualisation est définie par la position de l'oeil de l'observateur (E) et un axe de visée V si l'oeil est à une distance finie.
Si l'observateur est à l'infini, on parle de direction de visée.

Tous les déplacement des objets de la scène ou les déplacements de l'observateur se traduisent par le recalcul des projections sur l'écran. Il est donc nécessaire de passer constamment d'un repère à l'autre.
Ce passage est obtenu par composition de transformations spatiales élémentaires : translations, rotations, changements d'échelle, homothétie...
Pour des raisons d'efficacité, on cherchera à représenter ces transformations par des produits matriciels, qui sont programmés de façon efficace et même implantés par matériel sur les machines graphiques spécialisées.

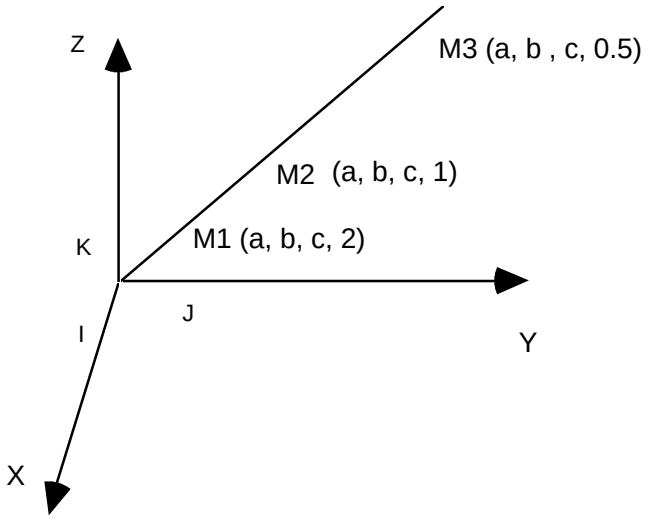


2.3. Le système de coordonnées homogènes

Les transformations de IR^3 dans IR^3 peuvent s'exprimer par des matrices 3x3, sauf la translation. Par contre dans IP^4, l'espace projectif de IR^3, toutes les transformations s'expriment par des matrices 4x4.

Les coordonnées homogènes d'un point de IR^3 sont constituées d'un quadruplet (x, y, z, t).

- si t _ 0, M de coordonnées homogènes (x, y, z, t) est le point de IR^3 de coordonnées cartésiennes (x/t, y/t, z/t).
- si t = 0, le point M est "à l'infini" et le triplet (x, y, z) s'interprète comme un vecteur.



Pour tous les points à distance finie de IR^3 les quadruplets (x, y, z, t) et (x/t, y/t, z/t, 1) représentent le même point. On choisira donc en général la seconde forme.; cependant on peut choisir une valeur de **t >1** pour coder avec des entiers les coordonnées réelles.

Exemple : ${}^t(0.2, 1.3, 2.5, 1) = {}^t(2, 13, 25, 10)$.

2.4. Variétés affines de IR^3 en coordonnées homogènes.

La droite

Une droite est définie par deux points A et B

A : ${}^t(x_A, y_A, z_A, 1)$;

B : ${}^t(x_B, y_B, z_B, 1)$;

et un point courant par M : ${}^t(x, y, z, 1)$

Représentation paramétrique de la droite :

$x - x_A = \lambda (x_B - x_A)$

$y - y_A = \lambda (y_B - y_A)$

$z - z_A = \lambda (z_B - z_A)$

Segment [A B] $0 < \lambda \leq 1$

Le plan

Un plan est défini par un point M_0 et une normale **n** au plan.

L'équation du plan est donc :

$$\mathbf{M_0 M} \cdot \mathbf{n} = 0 \quad (\cdot : \text{produit scalaire})$$

M_0 : ${}^t(x_0, y_0, z_0, 1)$

n : ${}^t(a, b, c, 0)$

On exprime ce produit scalaire nul :

$a (x-x_0) + b (y-y_0) + (c (z-z_0) = 0$

soit, sous forme d'un produit de matrices

$$[a\ b\ c\ d]\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = 0$$

avec $d = -(a x_0 + b y_0 + cz_0)$

Puissance d'un point par rapport à un plan

Soit P un plan d'équation $ax + by + cz + d = 0$
et M un point de coordonnées ${}^t(X, Y, Z, 1)$
Le nombre réel $P(M) = aX + bY + cZ + d$ est la "puissance de M par rapport à P". Le signe de P(M) détermine deux demi-espaces séparés par le plan P.

Intérieur d'un polyèdre convexe

Un point est intérieur à un polyèdre convexe dont toutes les normales aux faces du polyèdre sont orientées vers l'extérieur si la puissance de ce point par rapport à toutes ces faces est négative.

Faces visibles

Une face pour laquelle la puissance de l'oeil est négative n'est pas visible par un observateur extérieur au polyèdre... Les faces vues par l'observateur sont celles pour lesquelles $P_F(Oeil) > 0$.

2.5. Eléments de calcul géométrique

Produit scalaire

a, b deux vecteurs d'angle a

$a \cdot b = |a|\ |b|\ \cos(a)$

a : ${}^t(xa\ ya\ za\ 0)$ b : ${}^t(xb\ yb\ zb\ 0)$

$a \cdot b = xa\ xb + ya\ yb + za\ zb$

Déterminant

$$D = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix} = a_1 b_2 - a_2 b_1$$

$$D = \begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix} = a_1 \begin{vmatrix} b_2 & c_2 \\ b_3 & c_3 \end{vmatrix} - a_2 \begin{vmatrix} b_1 & c_1 \\ b_3 & c_3 \end{vmatrix} + a_3 \begin{vmatrix} b_1 & c_1 \\ b_2 & c_2 \end{vmatrix}$$

$$\sum_{j=1}^{n} (-1)^{i+j} a_{ij} D^*_{ij}$$

avec D^*_{ij} déterminant de la matrice obtenue en supprimant la ième ligne et la jème colonne dans (a_{ij}).

Produit vectoriel

$$\sum_{j=1}^{n} (-1)^{i+j} a_{ij} D^*_{ij}$$

a et *b* deux vecteurs d'angle α

$a \wedge b = n|a||b|\sin(\alpha)$

n normale au plan déterminé par (O, a, b)

expression analytique du produit vectoriel :

$$a \vee b = \begin{vmatrix} i & j & k \\ x_a & y_a & z_a \\ x_b & y_b & z_b \end{vmatrix} = i(y_a z_b - y_b z_a) - j(x_a z_b - x_b z_a) + k(x_a b - x_b y_a)$$

2.6. Transformations géométriques

Les transformations géométriques usuelles en coordonnées homogènes sont représentées par des matrices 4x4

f : P^4 --> P^4 M(f) est la matrice 4x4 de f
R est le repère (O, i, j, k) associé à IR^3

Translation

T : ${}^t(tx\ ty\ tz\ 0)$ un vecteur de IR^3
La translation de vecteur T est une application

t_T : P^4 --> P^4
X --> X + T avec X ${}^t(x\ y\ z\ 1)$

s'exprime par le produit matriciel $M(t_T) \times X$

La matrice $M(t_T)$ est donnée par

$$M(t_T) = \begin{vmatrix} 1 & 0 & 0 & tx \\ 0 & 1 & 0 & ty \\ 0 & 0 & 1 & tz \\ 0 & 0 & 0 & 1 \end{vmatrix} \quad \text{et } X = \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix}$$

(matrice 4x4) (Vecteur colonne)

Les coordonnées du point translaté sont données par $X' = M(t_T) \times X$

Matrice inverse T^{-1} : Translation de vecteur opposé

$$\begin{vmatrix} 1 & 0 & 0 & -tx \end{vmatrix}$$

$$M^{-1}(t_T) = \begin{vmatrix} 0 & 1 & 0 & -ty \\ 0 & 0 & 1 & -tz \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Changement d'échelle

c : P^4 --> P^4
$^t(x\ y\ z\ 1)$ --> $^t(k1\ x,\ k2\ y,\ k3\ z,\ 1)$

$$M(c) = \begin{vmatrix} k_1 & 0 & 0 & 0 \\ 0 & k_2 & 0 & 0 \\ 0 & 0 & k_3 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

dont l'inverse est

$$M^{-1}(c) = \begin{vmatrix} 1/k_1 & 0 & 0 & 0 \\ 0 & 1/k_2 & 0 & 0 \\ 0 & 0 & 1/k_3 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Cas particulier de changement d'échelle : l'homothétie
$k_1 = k_2 = k_3 = k$

Rotations autour d'un axe du repère

r : P^4 --> P^4 d'angle α

Rotation autour de Ox d'un angle α

$$R_x(\alpha) = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Rotation autour de Oy d'un angle α

$$R_y(\alpha) = \begin{vmatrix} \cos(\alpha) & 0 & \sin(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) & 0 \end{vmatrix}$$

$$\begin{vmatrix} 0 & 0 & 0 & 1 \end{vmatrix}$$

Rotation autour de Oz d'un angle α

$$R_z(\alpha) = \begin{vmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Rotations inverses autour d'un axe du repère

Il suffit de *changer le signe des termes en* $\sin(\alpha)$

Rotation d'angle teta autour d'un axe quelconque D

Schéma général

a) Amener D à l'origine : Translation T

b) Rotations pour amener D à coïncider avec l'un des axes Rx(alpha) et Ry(beta)

c) Rotation d'angle teta autour de l'axe D confondu avec Oz : Rz(teta)

d) Rotations et translation inverses pour ramener D en position initiale.

$M_{(RD)} = T \times R_x(\alpha) \times R_y(\beta) \times R_z(\theta) \times R_y^{-1}(\beta) \times R_x^{-1}(\alpha) \times T^{-1}$

D est donnée par deux points

M1 $(x_1, y_1, z_1, 1)$

M2 $(x_2, y_2, z_2, 1)$

u vecteur unitaire de D

$u = {}^t((x_2-x_1)/U, (y_2-y_1)/U, (z_2-z_1)/U, 0)$

avec $U = SQRT((x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2)$

soit $u = {}^t(a, b, c, 0)$

avec $a = (x_2-x_1)/U$ $b = (y_2-y_1)/U$ $c = (z_2-z_1)/U$

et $d = SQRT(a^2+b^2+c^2)$

Tous calculs faits on trouve

$$T(-OM_1) = \begin{vmatrix} 0 & 1 & 0 & -x1 \\ 0 & 0 & 1 & -y1 \\ 0 & 0 & 0 & -z1 \\ 0 & 0 & 0 & -1 \end{vmatrix}$$

$$R_x(\alpha) = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & c/d & -b/d & 0 \\ 0 & b/d & c/d & 0 \end{vmatrix}$$

$$\begin{vmatrix} 0 & 0 & 0 & 1 \end{vmatrix}$$

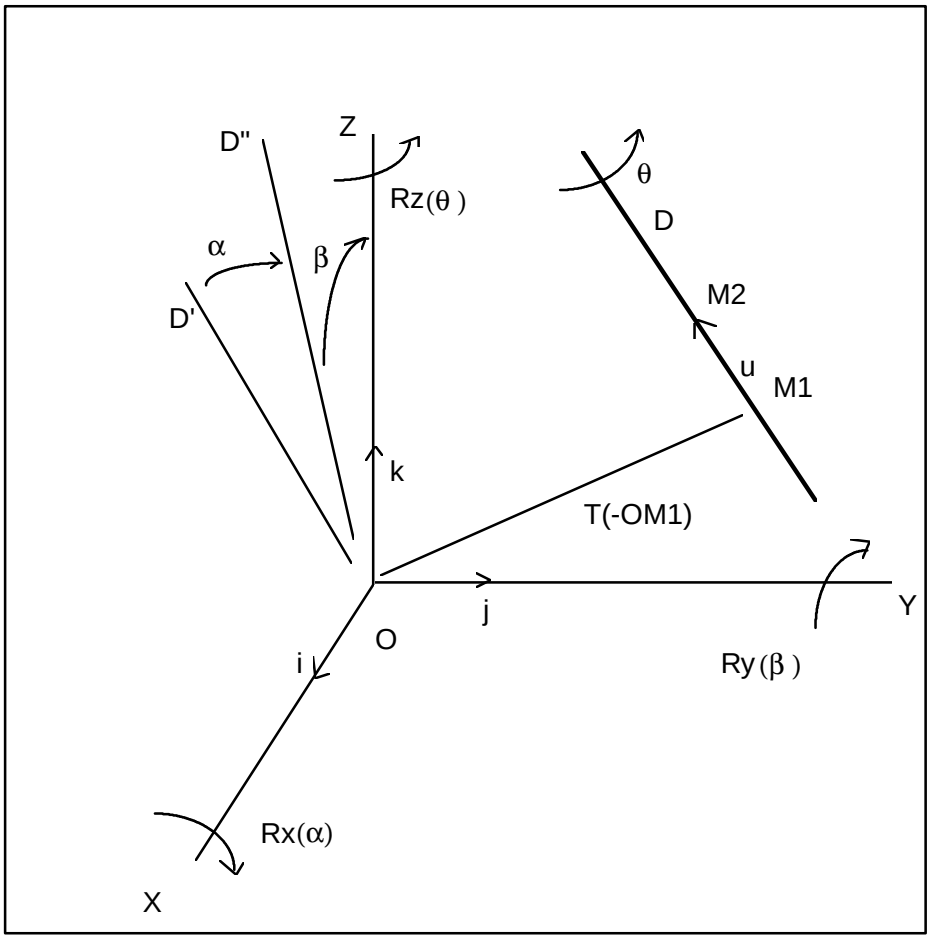
$$R_y(\beta) = \begin{vmatrix} d & 0 & -a & 0 \\ 0 & 1 & 0 & 0 \\ a & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Et pour la rotation autour de Oz d'angle q:

$$R_z(\theta) = \begin{matrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

On obtient la matrice

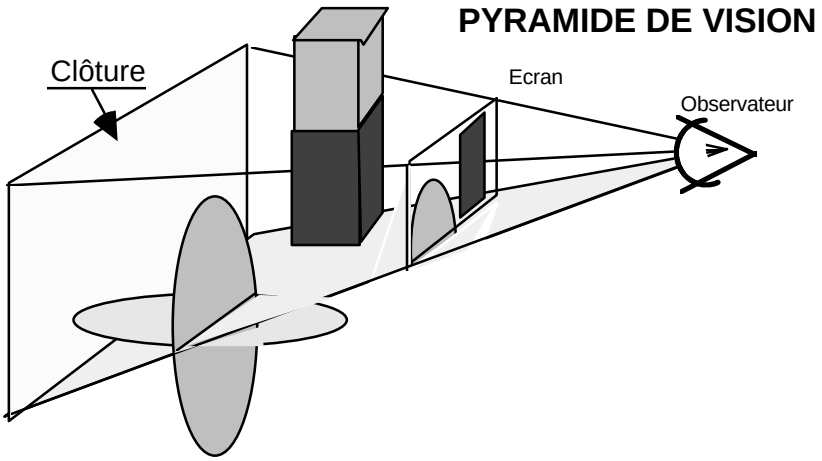
$$M_{(RD)} = T \times R_x(\alpha) \times R_y(\beta) \times R_z(\theta) \times R_y^{-1}(\beta) \times R_x^{-1}(\alpha) \times T^{-1}$$



3. PROJECTIONS

3.1. Modèle de la caméra et pipe line de visualisation

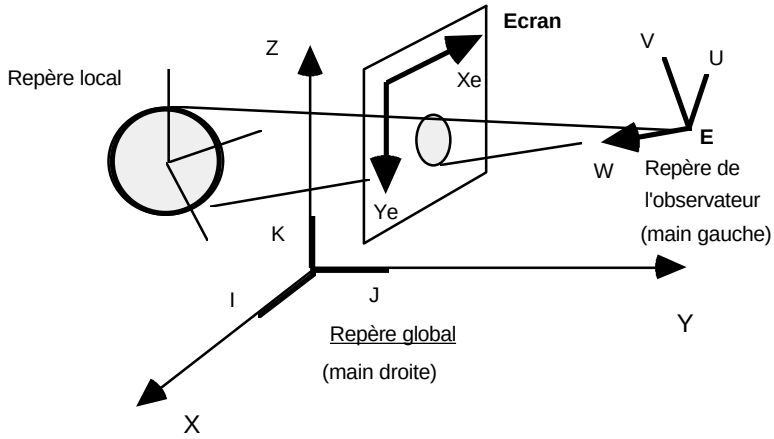
Une scène est une collection d'objets virtuels en 3 dimensions vus par un observateur à travers la fenêtre en 2 dimensions de l'écran d'ordinateur.
La visualisation implique de déterminer les objets de la scène contenus dans la *pyramide de vision* (**découpage**), puis de *projeter* ces objets sur le plan image (**projection perspective** par exemple), et enfin d'afficher cette image en la transformant en *primitives graphiques* dans l'espace écran (**pixels**).



L'opération de visualisation est définie dans l'espace objet par la position des objets, celle de l'observateur et un *axe de visée* si l'oeil est à une distance finie. Si l'observateur est à l'infini, on parle de *direction de visée*.

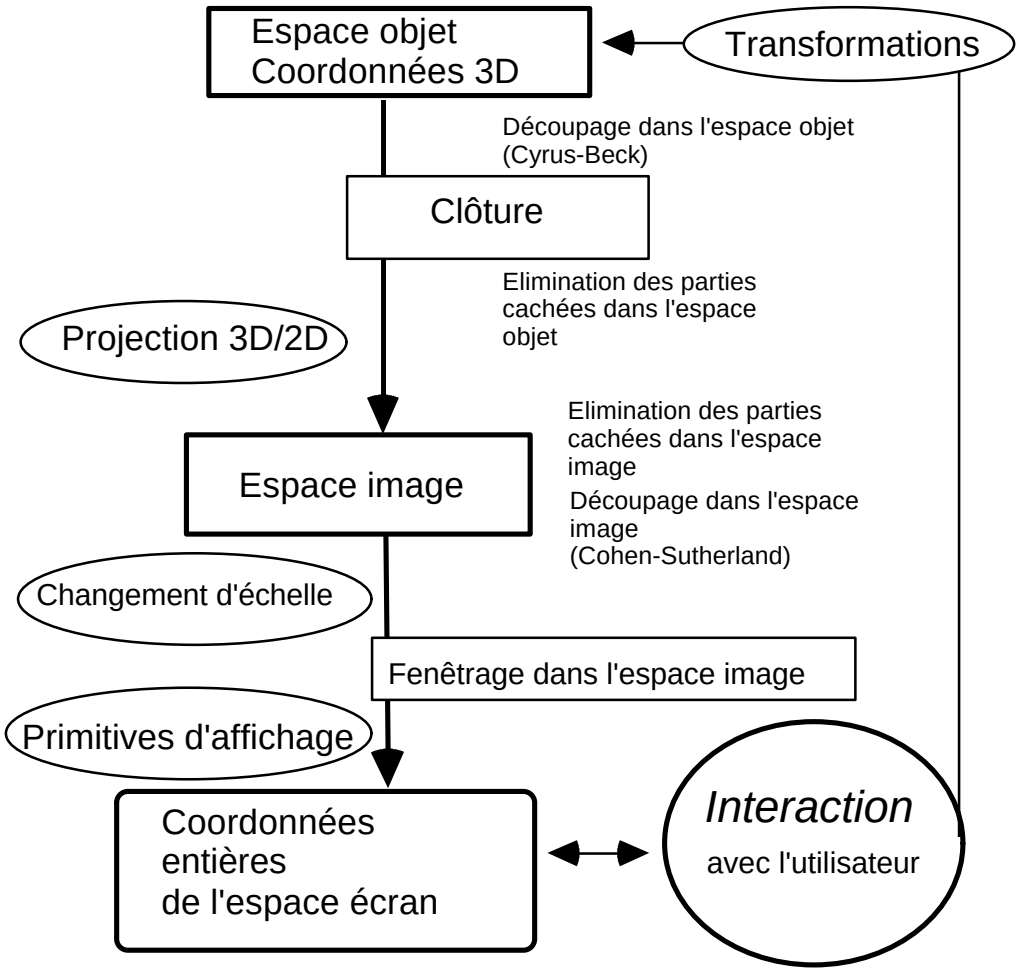
Tous les déplacement (objets de la scène ou observateur) se traduisent par le recalcul des projections sur l'écran.

La fluidité des déplacements est obtenue en recalculant et en affichant l'image au moins 25 fois par seconde.



Coordonnées d'un point dans le repère objet global	x y z	Coordonnées d'un point dans le repère local de l'observateur	xv yv zv

Pipe line de visualisation



3.2. Classification des projections

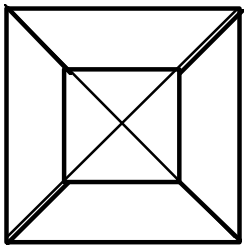
Les projections se divisent en deux grandes classes suivant la position du centre de projection :
- projections perspectives ayant leur centre de projection à distance finie
- projections parallèles ayant leur centre de projection à distance infinie.

Projections perspectives

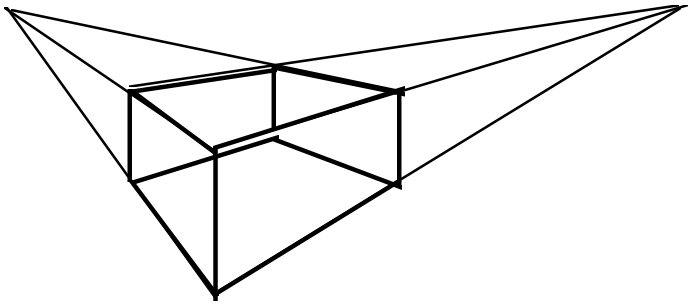
Le centre de projection est à distance finie du plan de projection
--> vision “humaine”.

Exemple : projection d’un cube

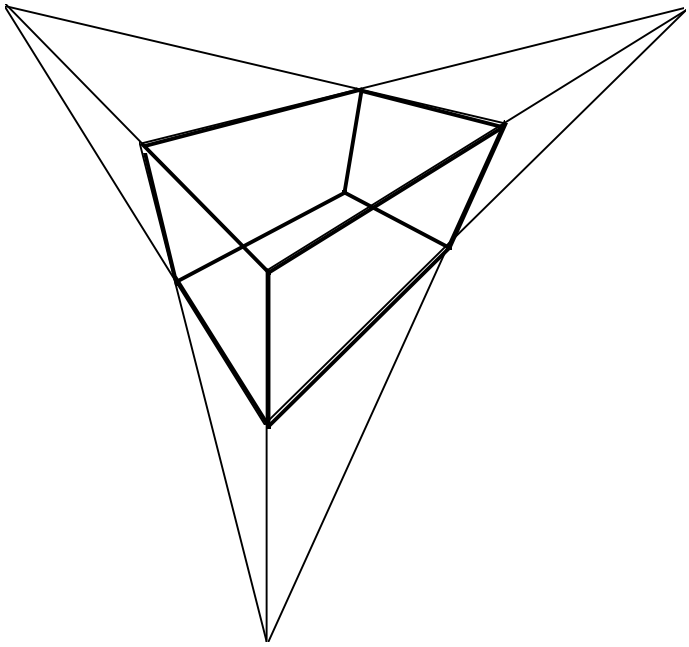
Projection à un point de fuite, deux faces parallèles au plan de projection



Deux points de fuites, quatre arètes parallèles au plan de projection



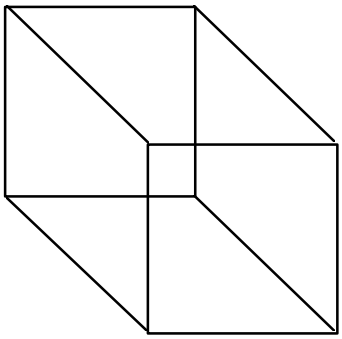
Trois points de fuite, aucune arète n'est parallèle
au plan de projection

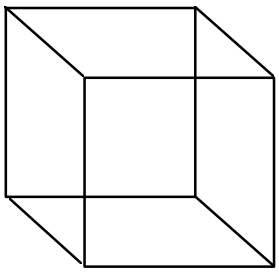


Projections parallèles

Le point de fuite est à l'infini.
- projection axonométrique : plan de projection perpendiculaire à la direction de projection
  - dimétrique
  - isométrique
  - orthographique
    - vue de dessus
    - vue de côté
    - vue de face
- projection oblique : le plan de projection n'est pas perpendiculaire à la direction de projection.
  - projection cavalière (45°)
  - projection cabinet : les distances fuyantes sont divisées par 2.

Perspective cavalière





Projection cabinet

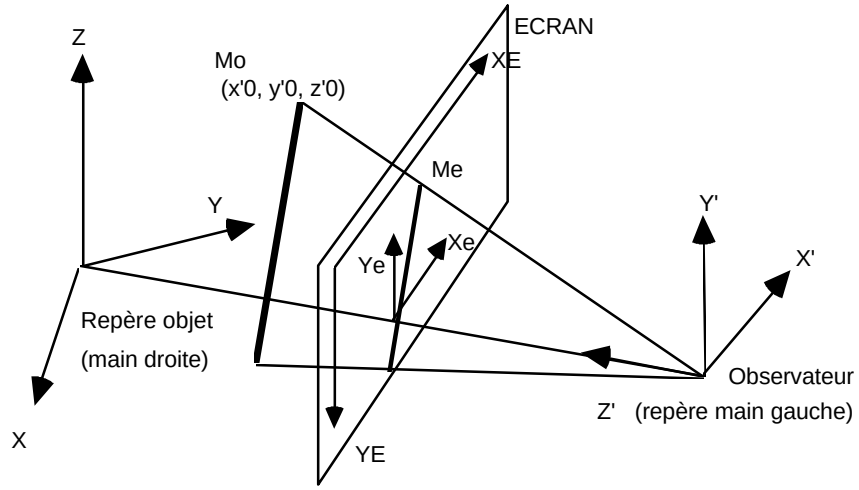
3.3. Calcul des projections perspectives

L'écran est perpendiculaire à la droite de visée (droite de l'oeil au centre du monde objet).

On connaît :

- la distance D de l'observateur à l'écran (plan de projection)
- la position de l'observateur dans le repère objet
- la position des points dans l'espace objet

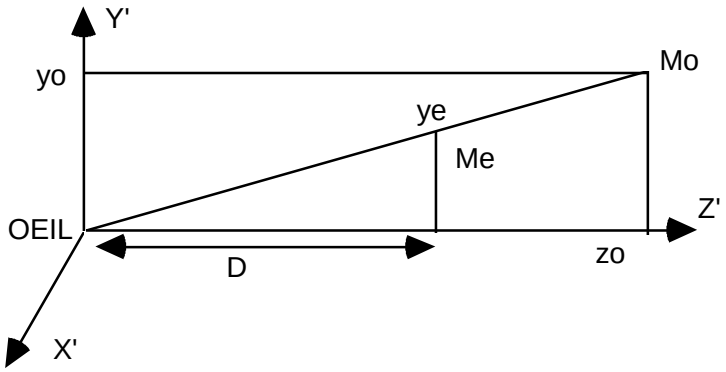
Il s'agit de déterminer les positions des projetés dans l'espace écran.



Le point Mo (xo,yo, zo) dans le repère (OEIL, X', Y', Z') de l'observateur se projette en Me(xe,ye) dans le repère image.

yo / ye = zo / D
xo / xe = zo / D

avec D *distance de l'observateur à l'écran*.



Pour le point M(x,y,z) on ramène la transformation perspective au repère observateur :
xe = x . (D/z)
ye = y . (D/z)

En coordonnées homogènes la matrice de transformation s'écrit :

$$\begin{vmatrix} X \\ Y \\ Z \\ W \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/D & 0 \end{vmatrix} \times \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix}$$

$$\Rightarrow \begin{vmatrix} X \\ Y \\ Z \\ W \end{vmatrix} = \begin{vmatrix} x \\ y \\ z \\ z/D \end{vmatrix}$$

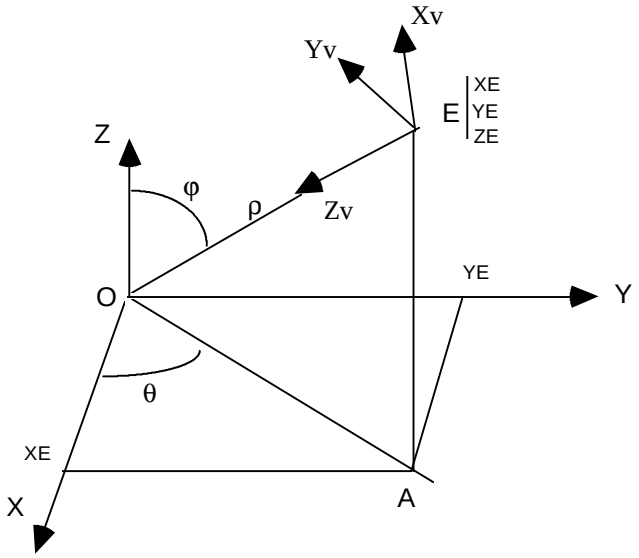
En passant dans IR^3 on retrouve :

$X = x /(z/D)$; $Y=y/(z/D)$; $Z = z/ (z/D) = D$.

Passage de l'espace objet à l'espace observateur

On peut décrire la position de l'observateur en coordonnées sphériques dans le repère objet.

$X_E = \rho \sin(\varphi) \cos(\theta)$
$Y_E = \rho \sin(\varphi) \sin(\theta)$
$Z_E = \rho \cos(\varphi)$



On se propose de déterminer la matrice **V** telle que

$$\begin{vmatrix} Xv \\ Yv \\ Zv \\ 1 \end{vmatrix} = \mathbf{V} \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix}$$

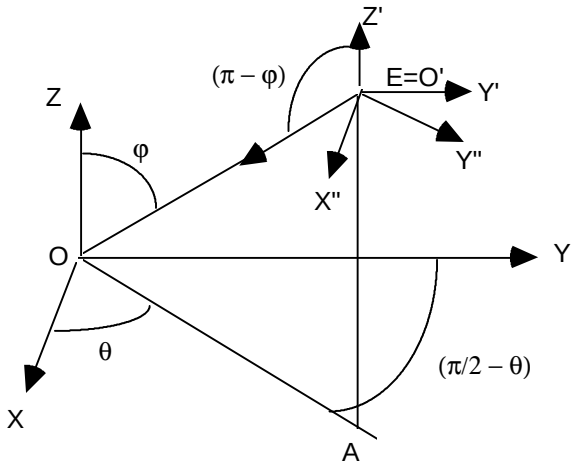
de passage du repère objet au repère de vision (observateur).

V est un produit de matrices élémentaires.

1) Translation de vecteur -OE, fait passer le point O au point E.

$$T = \begin{vmatrix} 1 & 0 & 0 & -xe \\ 0 & 1 & 0 & -ye \\ 0 & 0 & 1 & -ze \end{vmatrix}$$

$$\begin{vmatrix} 0 & 0 & 0 & 1 \end{vmatrix}$$



2) Rotation d'axe OZ pour amener OY dans le plan (OEA) d'angle (π /2 - θ)

$$R_Z(\pi/2 - \theta) = \begin{vmatrix} \sin(\theta) & -\cos(\theta) & 0 & 0 \\ \cos(\theta) & \sin(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

3) Rotation d'axe OX pour placer l'axe OZ (O'Z') dans la direction EO d'angle (φ–π)

$$R_X(\varphi-\pi) = \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & -\cos(\varphi) & \sin(\varphi) & 0 \\ 0 & -\sin(\varphi) & -\cos(\varphi) & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

4) Inversion de la direction de OX

$$M_{yz} = \begin{vmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Et finalement

$$V = T \times R_Z \times R_X \times M_{yz}$$

$$V = \begin{vmatrix} -\sin(\theta) & -\cos(\theta) & 0 & -xe \\ -\cos(\varphi)\cos(\theta) & -\cos(\varphi)\sin(\theta) & \sin(\varphi) & -ye \\ -\sin(\varphi)\cos(\theta) & -\sin(\varphi)\sin(\theta) & -\cos(\varphi) & -ze \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Finalement

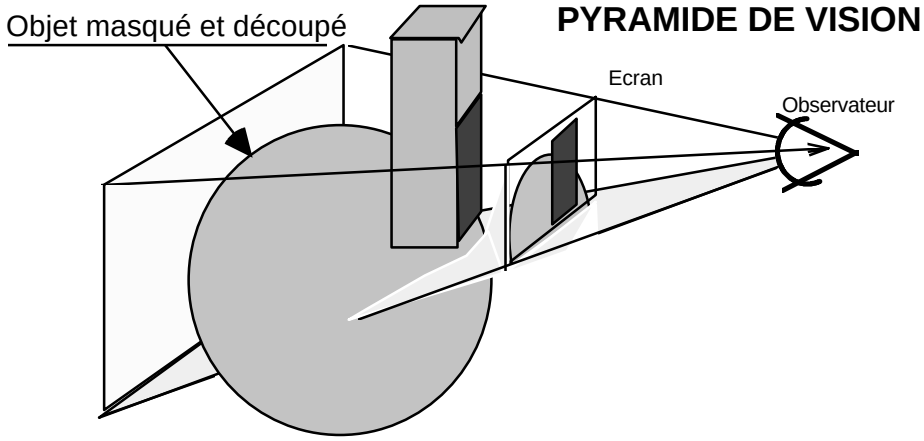
$$\begin{vmatrix} Xv \\ Yv \\ Zv \\ 1 \end{vmatrix} = \mathbf{V} \begin{vmatrix} x \\ y \\ z \\ 1 \end{vmatrix}$$

Puis on procède à la projection perspective vue plus haut.

4. ELIMINATION DES FACES CACHEES

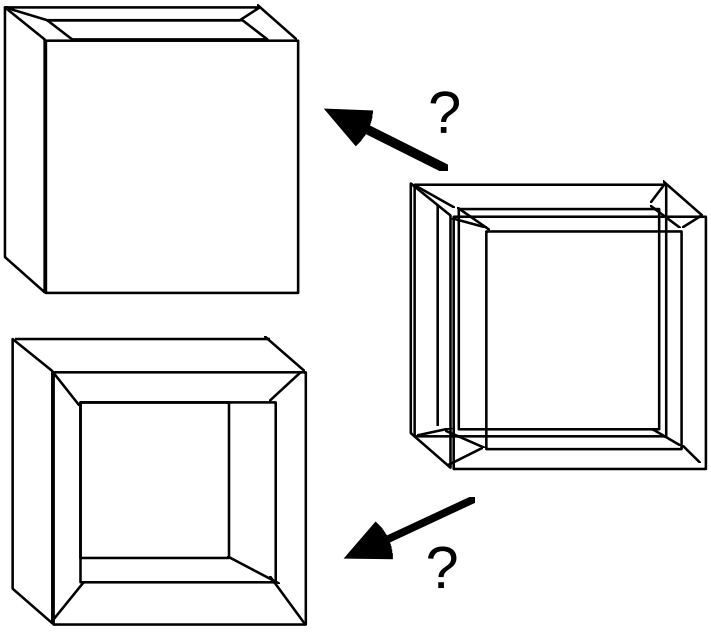
4.1. Que voit-on de la scène

Une scène est déterminée par un point de vue : seuls les objets contenus dans la *pyramide de vision* seront affichés, après élimination des lignes et faces cachées par d'autres objets.



4.2. Représentation fil de fer

Les objets sont "représentés par les arètes. C'est une représentation ambigüe.



L'élimination des lignes (arètes) cachées est adaptée aux périphériques vecteurs :

- tables à tracer
- écrans calligraphiques (vecteurs)

L'élimination des faces cachées convient aux écrans en mode mosaïque [raster]

--> visualisation 3D des objets opaques.

Les algorithmes dépendent beaucoup des structures de données et de la façon dont les objets sont représentés dans la mémoire de l'ordinateur.

4.3. Représentation des objets 3D

B-REP : représentation par les bords

-> approximation des objets par des polygones plans : polyèdres

Table des sommets

x	y	z	indice
10,5	-5,0	25,7	*0*
10,5	0,0	25,7	*1*
12,0	5,5	15,0	*2*
			3
			4
			...
			n

Coordonnées réelles 3D

Table des arètes

S1	S2	indice
0	2	*0*
2	4	*1*
4	0	*2*
		...
		m

Indices dans la table des sommets

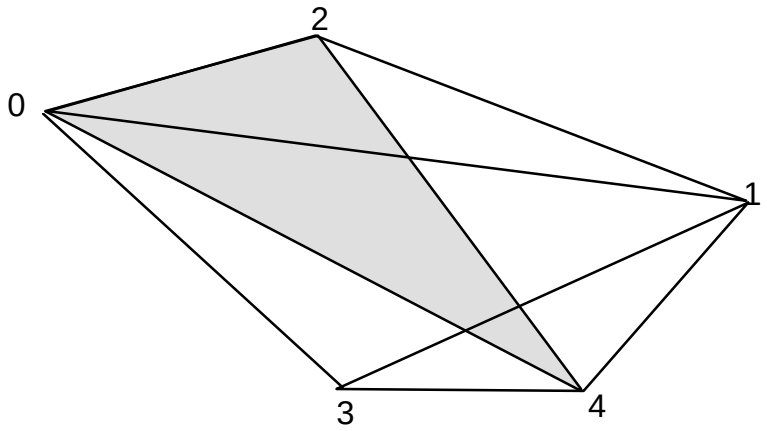
Table des faces

S1	S2	S3	...	indice
0	2	4		*0*
				1
				2
				...
				q

Indices dans la table des sommets
ou dans celle des arètes

GEOMETRIE

TOPOLOGIE



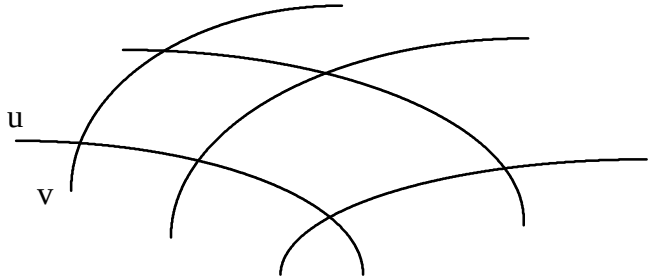
Facettes triangulaires et polyèdres convexes permettent une représentation simple et efficace d'une scène pouvant contenir plus de 10 000 facettes...

Patches ou carreaux

- de Coons
- de Bézier
- Splines

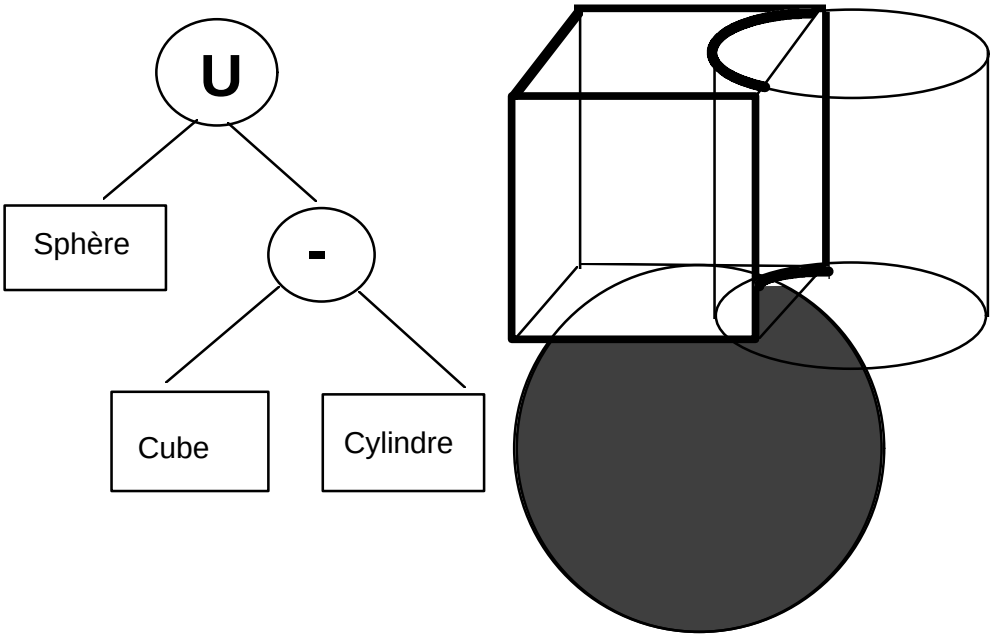
Un carreau est une partie élémentaire de surface délimitée par 4 courbes frontières.
Les courbes sont paramétrées :

p(x, y, z) avec x = f(u,v) ; v = g(u,v) ; z = h(u, v).



Cela permet de générer des formes "libres", des surfaces gauches.
Par contre les calculs d'intersection sont assez coûteux.

Arbres C S G [constructive solid geometry]



Les noeuds de l'arbre sont des opérations ensemblistes, les feuilles des primitives solides.

4.4. Algorithme d'élimination de faces cachées

On distingue
a) les algorithmes dans l'espace objet
b) les algorithmes dans l'espace image.

Tous utilisent la notion de tri géométrique. Il s'agit de définir un ordre de priorité sur les objets pour déterminer ceux qui sont **visibles**.
Tous exploitent plus ou moins la notion de c*ohérence d'image* pour limiter les calculs d'intersection.
Quand certains paramètres varient peu ou lentement, l'image se modifie peu d'un pixel à l'autre.

--> cohérence entre les arètes
--> cohérence entre les faces
--> cohérence d'éclairage
--> d'une ligne d'image à la suivante [*scan line* : ligne de balayage de l'image]

--> d'une page d'écran à l'autre : les images successives d'une scène sont très semblables si le point de vue varie peur.

--> cohérence de profondeur : une superposition par projection correspond presque toujours à des profondeurs différentes.

Caractéristiques des algorithmes d'élimination de faces cachées

Dans l'espace objet :

- résolution géométrique indépendante de l'affichage, dans la clôture de la pyramide de vision
- calculs en coordonnées réelles 3D
- comparaison de chaque facette avec toutes les autres ($o(n^2)$)

Dans l'espace image :

- en fonction de la résolution du périphérique d'affichage (écran)
- dans les limites de la fenêtre d'affichage
- en coordonnées entières si possible
- comparaison de chaque facette avec chaque pixel (o(n.r))
  - r : résolution de l'écran = MAXLIGNE x MAXPIXEL_LIGNE
  - n : nombre de facettes

Il s'agit de détecter si un point de coordonnées (x, y) est intérieur à une facette.
La prise en compte de la cohérence spatiale améliore les algorithmes dans l'espace image.

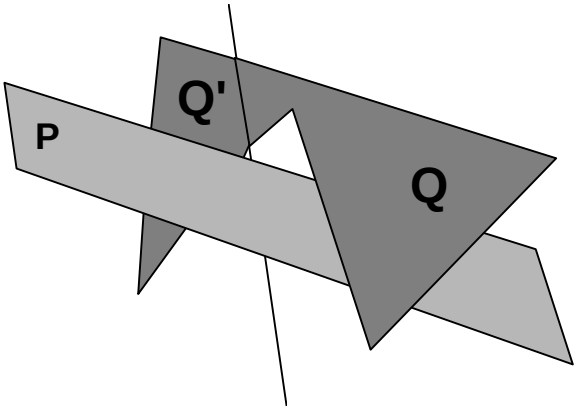
Algorithmes dans l'espace objet

Ce sont les plus anciens. [Galimberti & Montanari 1969]

Une scène est constituée d'objets polyédriques non intersectants opaques ; chaque face a une normale orientée vers l'extérieur. On détermine les faces arrières à l'aide des normales. On supprime les arêtes appartenant à deux faces arrières. Pour les arêtes restantes, il faut tester si aucune face ne les cache.

Algorithme du peintre [Newell, Newell & Sancha]

Cet algorithme repose sur un tri des facettes selon la distance à l'observateur.
Les facettes les plus éloignées sont affichées en premier ; les facettes (opaques) les plus proches venant recouvrir les facettes les plus éloignées...



Une scène est constituée de polyèdres ; les faces sont des polygones classés en triant les sommets les plus éloignés de chaque face selon leur profondeur. En cas de chevauchement, les faces sont découpées selon le plan contenant l'autre face.
Puis on projette les faces classées sur le plan de l'écran en commençant par celles qui sont les plus éloignées.

La force de cet algorithme est sa *simplicité*, mais de nombreuses faces cachées sont affichées inutilement.

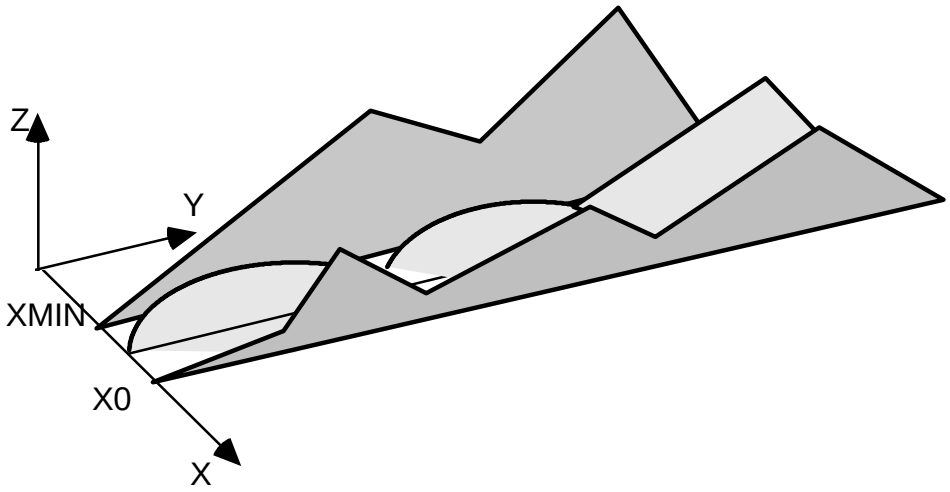
Algorithmes mixtes

Algorithme de ligne de crête

[Wright 73 - dans l'espace image] [Williamson 72 - dans l'espace objet]

Cet algorithme est utile pour visualiser les **fonctions** de deux variables et les MNT (modèle numérique de terrain).

L'idée est de couper la surface à représenter par des plans d'équation x=Cte.



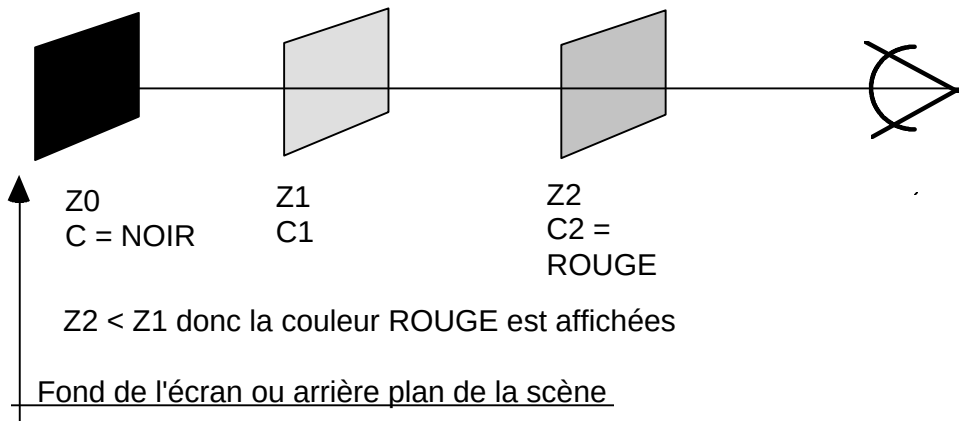
```
        LIGNE_DE_CRETE : tableau d'altitudes réelles

Fonction CRETE(altitude f(x,y))
{
        /* Afficher la courbe la plus proche de l'observateur */
        Pour y=YMIN à YMAX faire
                Afficher f(x0, y);
        /* Initialiser un tableau à une dimension LIGNE_DE_CRETE
                avec les valeurs de f(x0,y) */
        Pour y=YMIN à YMAX faire
                LIGNE_DE_CRETE[y]= f(x1, y);
        Pour j=1 à XMIN faire
                Pour y=YMIN à YMAX faire
                        Si f(xj, y) > LIGNE_DE_CRETE[y]
                                {
                                        Afficher f(xj, y);
                                        LIGNE_DE_CRETE[y]= f(xj, y);
                                }
}
```

Algorithmes dans l'espace image

Algorithme du Z-buffer [tampon de profondeur]

C'est un algorithme dans l'espace image, simple (il est souvent "câblé" sur les machines haut de gamme), mais il est gourmand en espace mémoire et présente des difficultés d'anti-aliassage.

L'image est une matrice de pixels.
Pour chaque pixel on recherche la valeur de couleur, qui dépend de la profondeur, c'est-à-dire de la distance à l'oeil de l'observateur de l'objet représenté par ce pixel.



Algorithme du Z-buffer

```
        mpl : maximum de pixels par ligne d'écran ;
        mlécran : maximum de lignes d'écran;
        Z : tableau [mpl, mlécran] de réels ;
        C : tableau [mpl, mlécran] de couleurs ;

Fonction Z_BUFFER (Scène, Oeil)
/* modifie le tableau C */
{
        réel z; entiers i, j; objet E;
```

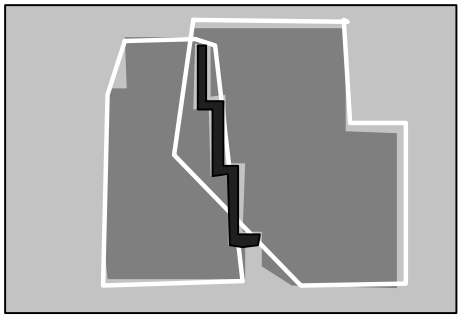
```
        Pour chaque pixel (i,j) faire
        {
                Z[i,j] = INFINI ;
                C[i,j] = COULEUR_FOND;
        }

        Pour chaque objet E de la scène faire
        {
                Calculer la projection de E sur l'écran ;
                Pour chaque pixel de cette projection faire
                {
                        z = Profondeur de E en (i,j);
                        Si (z<Z[i,j])
                        {
                                Z[i,j] = z;
                                C[i,j] = Couleur de E en (i,j);
                        }
                }
        }
}
```

Les avantages du Z-BUFFER

- Rapide ; câblé ; tire profit de la cohérence de scène
- Compatible avec le lissage de Phong ou de Gouraud.

Les inconvénients du Z-BUFFER

- Les éléments sont affichés dans un ordre quelconque
- L'image est connue à la résolution de la mémoire d'image, donc il peut y avoir des défaut d'aliassage.
- Il n'est pas possible de représenter les ombres portées, les transparences et les réflexions.
- Pour une image 1024 x 1024 il faut 1024 x 1024 x 2 octets pour le tampon de profondeur (2 MO).

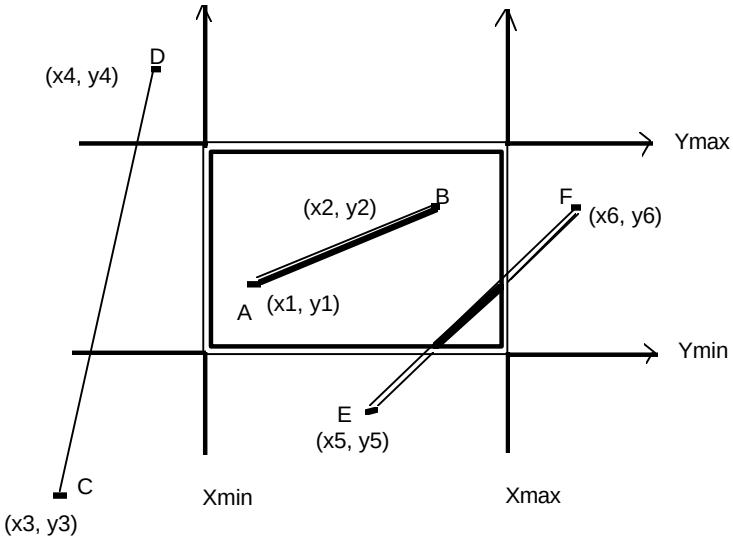


Quelle couleur domine à la frontière des deux polygones ?

5. FENETRAGE

5.1. Découpage d'un segment [clipping] Algorithme de Cohen et Sutherland - Hodgman

On appelle fenêtrage (ou découpage) l'opération qui consiste à découper un objet graphique (segment) selon les bords de la fenêtre de visualisation.
Pour déterminer si un segment est visible on considère les coordonnées de ses extrémités par rapport aux droites constituant les bords de la fenêtre de diagonale ((Xmin, Ymin) (Xmax, Ymax))



A chaque point (x, y) est associé un code de 4 chiffres binaires

CODE (x, y)=	Cxmin	Cxmax	Cymin	Cymax
				0 si y <= Ymax
				1 si y > Ymax
			0 si y >= Ymin	
			1 si y < Ymin	
		0 si x <= Xmax		
		1 si x > Xmax		
	0 si x >= Xmin			
	1 si x < Xmin			

Avec cette codification les codes des points (x1, y1) et (x2, y2) est 0000 ;
le code de (x3, y3) est 1010 ; le code de (x4, y4) est 1001
le code de (x5, y5) est 0010 ; le code de (x6, y6) est 0100

Condition nécessaire et suffisante pour qu'un segment [AB] d'extrémités (x_A, y_A) et (x_B, y_B)soit entièrement visible :

- les deux codes sont nuls

((code(x_A, y_A)==0) && (code(x_B, y_B) == 0))

Condition suffisante pour que le segment [CD] soit entièrement invisible

- un bit commun différent de zéro

((code(x_C, y_C) & code(x_B, y_B)) != 0)

Le segment [EF] est en partie visible, pour déterminer quelle portion conserver, on peut procéder comme suit :

- partager [EF] en deux segments [EM] et [MF] égaux
- rappeler récursivement l'opération de fenêtrage sur chaque nouveau segment.

La condition d'arrêt de la récursion, c'est la résolution de l'écran : un pixels d'écart entre les extrémités du segment.

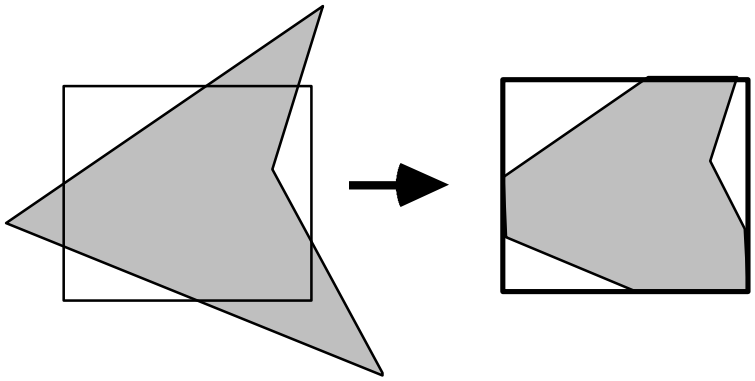
5.2. Découpage d'un polygone

Le polygone est décrit par la liste de ses sommets qui définit une liste triée de ses arcs.
Lsom (s1, s2, s3, ..., sn)
Larc (s1s2, s2s3, ..., sns1)
L'algorithme parcourt la liste de sommets et examine à chaque pas la relation entre les sommets et la ligne de découpe. Il crée une nouvelle liste de sommets en ajoutant 0, 1 ou 2 sommets suivant la stratégie ci-dessous.

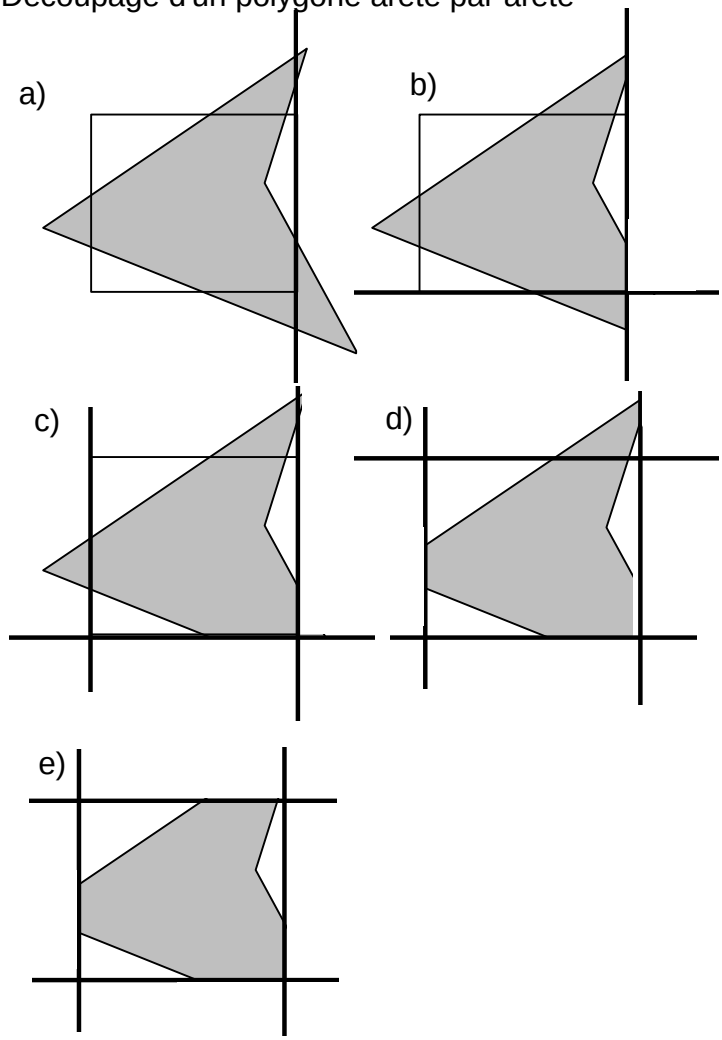
-prendre un sommet S_fin d'un arc (S_ini, S_fin). S_ini a déjà été traité. Examiner la stratégie concernant S_fin :

S_fin S_ini	intérieur	extérieur
intérieur	ajouter + 1 sommet	ajouter le point d'intersection +1 sommet
extérieur	ajouter le pont d'intersection et le S_fin +2 sommets	ne rien faire +0 sommet

Il suffit de répéter cette démarche pour chaque ligne de découpe. L'algorithme s'étend au découpage par n'importe quel polygone convexe.



Découpage d'un polygone arète par arète



6. ESTOMPAGE

Soit une triangulation 3D et une source de lumière ponctuelle à une distance infinie. L'éclairage de chaque facette triangulaire (assimilée à une surface mate) dépend de l'orientation de la normale à la facette par rapport à la direction du rayon lumineux incident.

$$I_{Diffuse} = I_{Ponctuelle}(\vec{L}.\vec{N})$$
$$\text{avec } \|\vec{L}\| = \|\vec{N}\| = 1$$

On peut donc exprimer l'éclairement de chaque facette, à un facteur I_p près, comme un produit mixte de trois vecteurs :

$$\vec{u} = \mathrm{AB}/\|\mathrm{AB}\| \ ; \ \vec{v} = \mathrm{BC}/\|\mathrm{BC}\| \ ; \ \vec{w} = \vec{L}/\|\vec{L}\|$$

puisque $\vec{N} = \vec{u} \wedge \vec{v}$ (produit vectoriel de AB par BC normalisés)

En exprimant les coordonnées de chaque vecteur

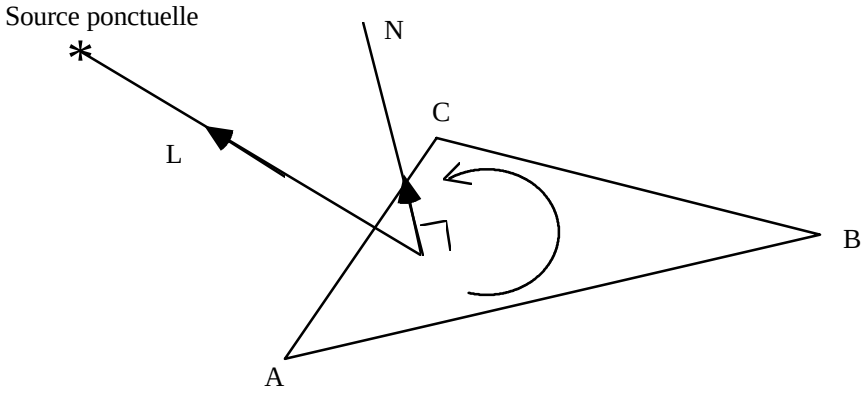
$$\vec{u}\begin{bmatrix} a \\ b \\ c \end{bmatrix} ; \vec{v}\begin{bmatrix} d \\ e \\ f \end{bmatrix} ; \vec{w}\begin{bmatrix} i \\ j \\ k \end{bmatrix}$$

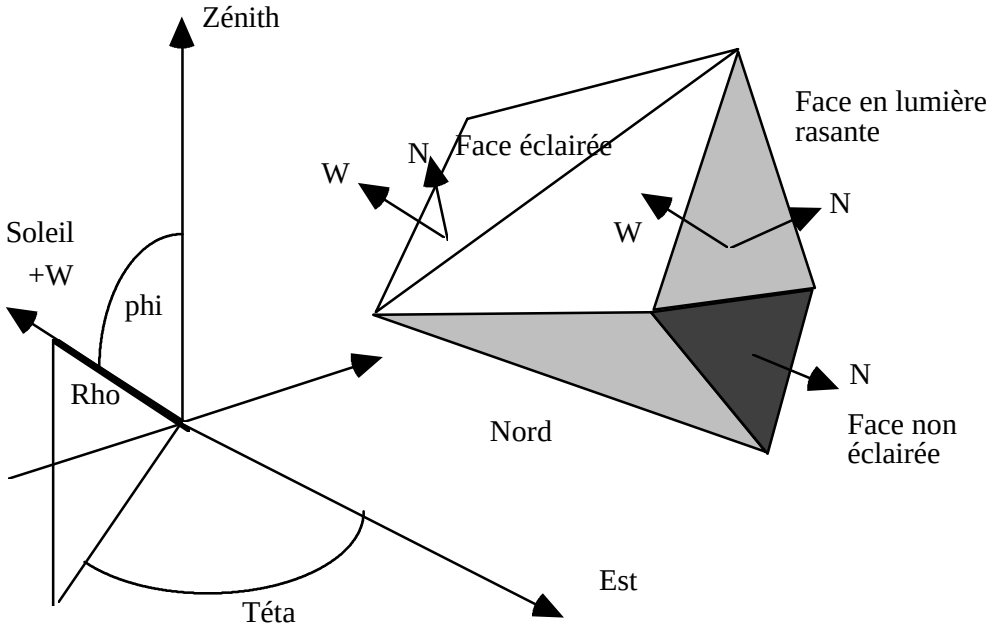
le produit mixte $(\vec{u} \wedge \vec{v}).\vec{w}$ est égal au déterminant

$$\Delta = \begin{vmatrix} a & d & i \\ b & e & j \\ c & f & k \end{vmatrix} = i(bd - ce) - j(af - cd) + k(ae - bd)$$

Si $\Delta > 0$ la facette est éclairée, sinon elle est sombre...

Selon la loi de Lambert la quantité de lumière diffuse reçue par l'observateur est indépendante de la position de celui-ci, pourvu que la facette soit visible :
Pour déterminer quelles sont les facettes visibles, il suffit d'appliquer le même raisonnement que précédemment en calculant le produit mixte de la normale à chaque facette avec la direction de visée.





6.1. Algorithme

1) Calculer les coordonnées de la source lumineuse (ou plutôt son coefficient directeur) en coordonnées sphériques

$$\vec{w}\begin{cases} x = r\cos(q)\sin(j) \\ y = r\sin(q)\sin(j) \\ z = r\cos(j) \end{cases} \quad \text{avec } r = 1 \text{ et } x^2 + y^2 + z^2 = 1$$

2) Pour chaque triangle, calculer la normale N :

Soient A, B et C trois sommets

$\vec{u} = AB$; $\vec{v} = BC$

$\vec{N} = \frac{\vec{u}}{\|\vec{u}\|} \wedge \frac{\vec{v}}{\|\vec{v}\|}$ (Si < 0 calculer $\frac{\vec{v}}{\|\vec{v}\|} \wedge \frac{\vec{u}}{\|\vec{u}\|}$)

puis calculer l'intensité lumineuse diffuse

$I_{Diffuse} = (\frac{\vec{u}}{\|\vec{u}\|} \wedge \frac{\vec{v}}{\|\vec{v}\|}).\vec{w}$

$I_{Diffuse} \in [-1,+1]$. Si $I_{Diffuse} \leq 0,$ la facette est dans l'ombre

3) Calculer les coordonnées de l'observateur

$$\vec{e}\begin{cases} x_e = g\cos(a)\sin(b) \\ y_e = g\sin(a)\sin(b) \quad \text{avec } g > 0 \\ z_e = g\cos(b) \end{cases}$$

Pour chaque triangle éclairé calculer la visibilité :

$\Delta_{Visible} = (\frac{\vec{u}}{\|\vec{u}\|} \wedge \frac{\vec{v}}{\|\vec{v}\|}).\frac{\vec{e}}{\|\vec{e}\|}$

Si $\Delta_{Visible} > 0$ la facette est visible sinon invisible..

4) Pour toutes les facette éclairées et visibles attribuer une couleur d'intensité proportionnelle à l'éclairement I ; si on dispose de 16 couleurs (ou plutôt de 16 gris en valeur de luminosité croissante) :

Si I<=0 Couleur = 0

et si I>0

$$\text{Couleur} = \lceil 15\ I \rceil \text{ avec } I \in]0,1]$$

5) Afficher les facettes par l'algorithme du peintre en commençant par celles qui sont le plus éloignées de l'observateur.

Bibliographie

BURGER P., DUNCAN G., “Interactive Computer Graphics, Fonctional, Procedural and Device-level methods”, Addison Wesley - 1989
FOLEY J.D., VAN DAM A., FEINER, HUGHES “Computer Graphics, Principles and Practice”, Addison Wesley -1982 -1992
LIEBLING T., ROTHLISBERGER H. : “Infographie et applications”, Masson - 1988
PEROCHE B., GHAZANFARPOUR D., ARGENCE J., MICHELUCCI D., “La synthèse d’image”, Hermès - 1990
SCHWEIZER Philippe, “Infographie 1 et 2”, Presses Polytechniques Romandes - 1987

Table des matières