INF4034
Imagerie Numérique

Aurélien Texier
texier@esiea.fr

2020-2021

esiea
ECOLE D'INGENIEURS
DU MONDE NUMERIQUE

Introduction à l'imagerie numérique
Synthèse 3D

Intro

Introduction

- Problématique

Comment générer une image correspondant à la représentation d'un environnement virtuel ?

  - Modèles pour représenter cet environnement
  - Algorithmes pour transformer les données géométriques en images

Introduction à l'imagerie numérique
Synthèse 3D

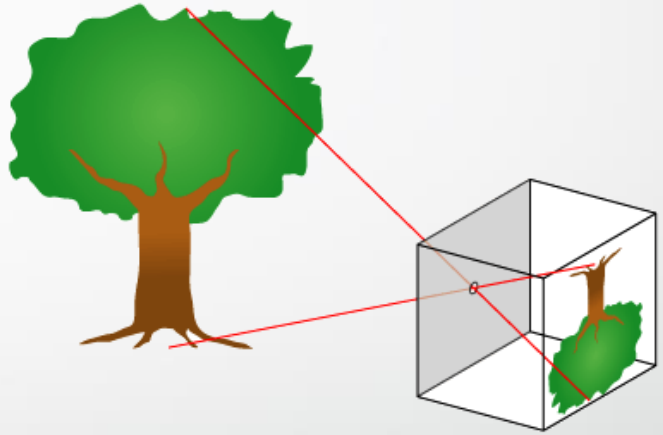
Modélisation

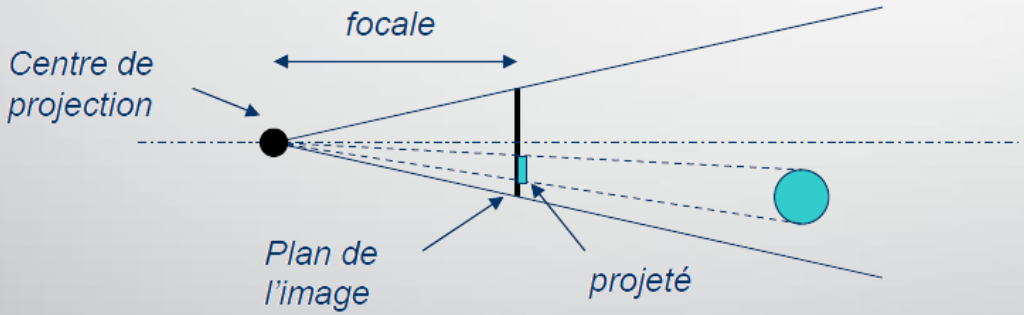
Modélisation

- Modèles pour représenter un environnement 3D
  - Caméras
    - Dispositifs qui effectuent une projection de l'espace 3D sur un plan 2D (ie. œil, appareil photo, …)
  - Objets
    - Composent la scène virtuelle
    - Différentes modélisation : surfacique, volumique, paramétrée, subdivisée…
  - Matériaux
    - Modèles pour représenter les propriétés optiques des matériaux (comment ils interagissent avec la lumière)
  - Sources lumineuses
    - Différentes façons « d'injecter » de la lumière dans la scène, plus ou moins réalistes et plus ou moins coûteuses en calcul.

Camera

- Méthode « trou d'épingle » (pinhole)
  - Projection perspective idéale





Camera

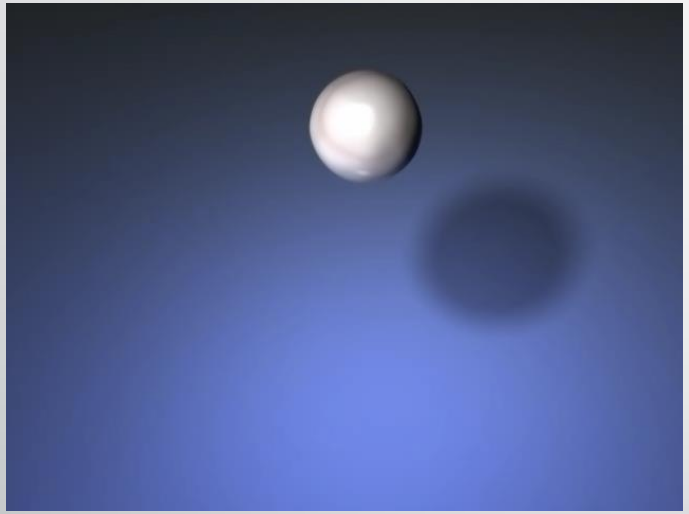
- Cas particulier : camera orthographique
  - Projection orthogonale sur le plan image
  - x = X, y = Y
  - Peut être vu comme une projection perspective dont le centre part de l'infini
  - Très utilisé en CAO pour les vues standards car
    - Préserve les distances sur X et Y,
    - Conserve les angles.

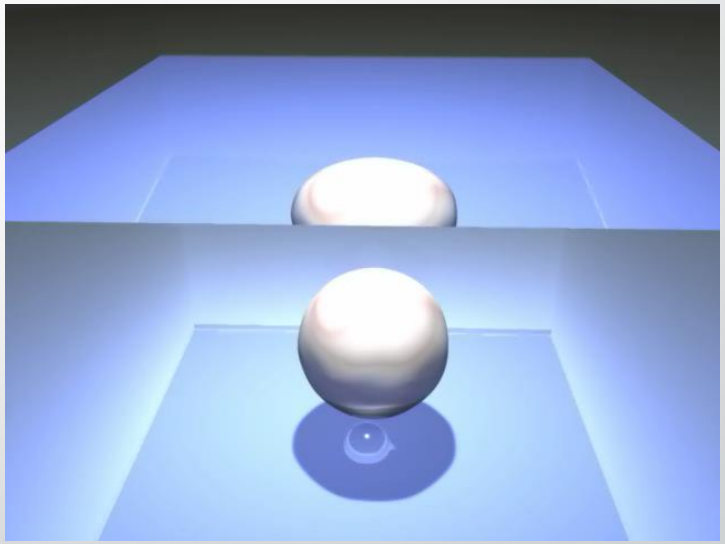
Objets 3D

- Différentes manières de les décrire :
  - Ensemble d'éléments de volume : **Voxels** (volume elements)
    - Coûteux en mémoire
    - Permet des effets que d'autres représentations ne permettent pas (feux, fumées, fluides, etc…)
    - Le compromis mémoire/calcul/qualité n'est pas intéressant pour la visualisation d'objets solides.

Objets 3D

- Voxels



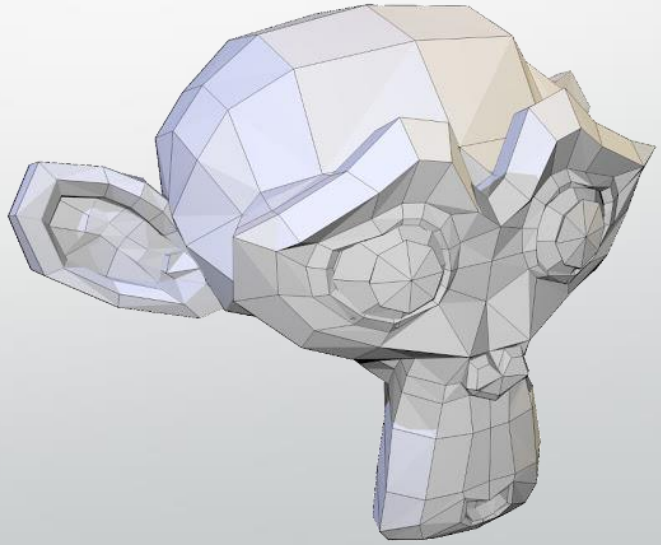


Objets 3D

- Différentes manières de les décrire :
  - Ensemble d'éléments de volume : **Voxels** (volume elements)
    - Coûteux en mémoire
    - Permet des effets que d'autres représentations ne permettent pas (feux, fumées, fluides, etc…)
    - Le compromis mémoire/calcul/qualité n'est pas intéressant pour la visualisation d'objets solides.
  - Ensemble de surfaces
    - Décrites à l'aide de surfaces élémentaires : triangles, quadrilatères, polygones eux-même composés de sommets (**Vertex**)

Objets 3D

- Facette = ensemble de points ordonnés dans le même plan (polygone)
- Ajout d'informations complémentaires telles que :
  - Normal à la surface
  - Coordonnées de texture
  - Couleur
  - Poids
  - …



Objets 3D

- Stockage en mémoire
  - Afin d'optimiser la place en mémoire, on décrit les polygones à l'aide d'index dans une liste de vertices associée
  - Exemple : fichier .obj

Objets 3D

- Cube.obj

```
v  0.0  0.0  0.0
v  0.0  0.0  1.0
v  0.0  1.0  0.0
v  0.0  1.0  1.0
v  1.0  0.0  0.0
v  1.0  0.0  1.0
v  1.0  1.0  0.0
v  1.0  1.0  1.0

vn  0.0  0.0  1.0
vn  0.0  0.0 -1.0
vn  0.0  1.0  0.0
vn  0.0 -1.0  0.0
vn  1.0  0.0  0.0
vn -1.0  0.0  0.0
```

Liste des vertices

Liste des normales

Liste des polygones

```
f  1//2  7//2  5//2
f  1//2  3//2  7//2
f  1//6  4//6  3//6
f  1//6  2//6  4//6
f  3//3  8//3  7//3
f  3//3  4//3  8//3
f  5//5  7//5  8//5
f  5//5  8//5  6//5
f  1//4  5//4  6//4
f  1//4  6//4  2//4
f  2//1  6//1  8//1
f  2//1  8//1  4//1
```

Objets 3D

- Différentes manières de les décrire :
  - Ensemble d'éléments de volume : **Voxels** (volume elements)
    - Coûteux en mémoire
    - Permet des effets que d'autres représentations ne permettent pas (feux, fumées, fluides, etc…)
    - Le compromis mémoire/calcul/qualité n'est pas intéressant pour la visualisation d'objets solides.
  - Ensemble de surfaces
    - Décrites à l'aide de surfaces élémentaires : triangles, quadrilatères, polygones eux-même composés de sommets (**Vertex**)
    - Décrites à l'aide d'**équation** : sphères, surfaces courbes, …

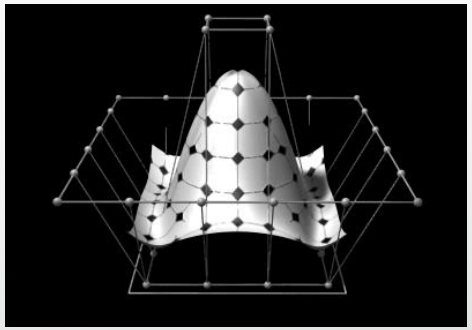
Objets 3D

- Equations : description des objets à l'aide d'objets canoniques définis par leurs équations
  - Equations cartésiennes :
    - Plan : $ax + by + cz + d = 0$;
    - Sphère de centre (c_x, c_y, c_z) et de rayon r: $(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 = r^2$
  - Equations paramétriques :
    - Sphère de centre (x_0, y_0, z_0) et de rayon r :
      - $x = x_0 + r \cos\theta \sin\phi$
      - $y = y_0 + r \sin\theta \sin\phi$
      - $z = z_0 + r \cos\phi$

avec $0 \leq \theta \leq 2\pi$ et $0 \leq \phi \leq \pi$

Objets 3D

- Courbes et surface paramétrées
  - Décrire les surface courbes complexes avec des équations
  - A la base : Bézier, ingénieur chez Renault, pour représenter les courbes de carrosserie
  - Evolution : surfaces → NURBS

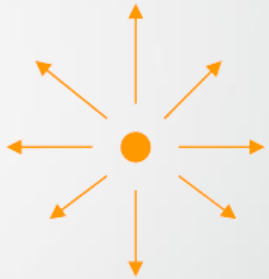


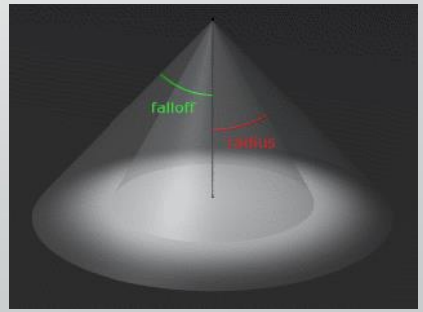
- Avantages : place en mémoire, représentation exacte, rapidité de résolution d'intersection (une seule équation par objet, cf. raytracing)
- Inconvénients : difficulté pour modéliser les objets complexes, coûts en calcul élevés (polynomes)

Sources lumineuses

- **Ponctuelle**
  - Un point génère des rayons lumineux dans toutes les directions.
- **Directionnelle**
  - La lumière ne vient que d'une direction, c'est une lumière ponctuelle à l'infini.
- **Spot**
  - Lumière ponctuelle restreinte à un cône d'émission et éventuellement un cône d'estompage.

En général : une couleur propre et une intensité. Des fois, une couleur déjà modulée par l'intensité.





Sources lumineuses

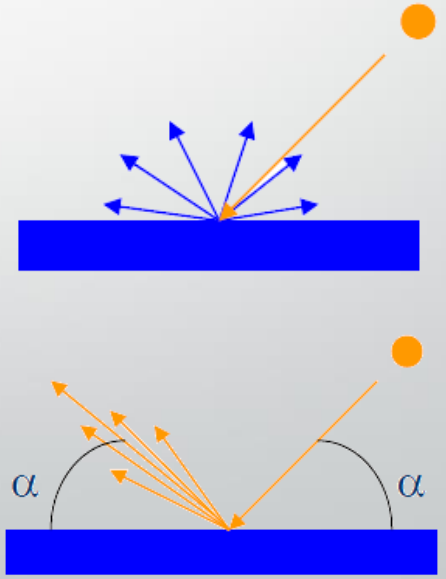
- Atténuation de la lumière avec la distance.
  - Peut être constant, linéaire, quadratique
  - Dans la réalité, en 1/d
  - En infographie, souvent "bricolé" pour obtenir l'esthétique (début constant puis une partie dégressive linéaire)

Matériaux

- Les propriétés optiques des matériaux sont approximées par des équations
  - Certaines sont plus ou moins "passe partout"
  - D ’autres plus spécifiques à certains matériaux
  - Plus ou moins coûteuses en calculs
- Principaux paramètres :
  - La direction du point d’observation
  - La normale à la surface
  - La direction des sources lumineuses
  - Un ensemble de coefficients propre au matériau considéré ainsi qu’à la source lumineuse
- Ces modèles sont indépendants de l’algorithme utilisé pour généré l’image finale (rastérisation, lancé de rayon, …)

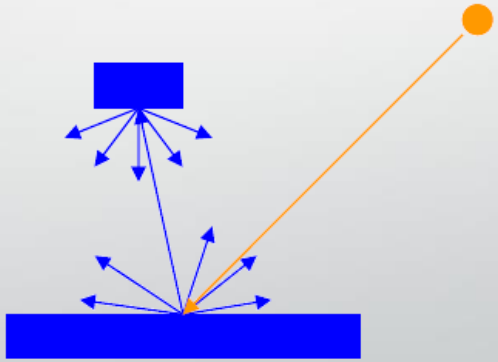
Matériaux

- Lorsqu'un matériau est illuminé par une source, il absorbe une partie de la lumière et en réfléchie l'autre partie. La partie réfléchie se décompose en plusieurs termes :
  - Réflexion diffuse
    - Uniforme, omnidirectionnelle et de la couleur de l'objet
    - Dominante pour les matériaux mates
  - Réflexion spéculaire
    - Centrée autour de la direction réfléchie de la lumière / surface, de la couleur de la lumière
    - Dominant pour les matériaux brillants



Matériaux

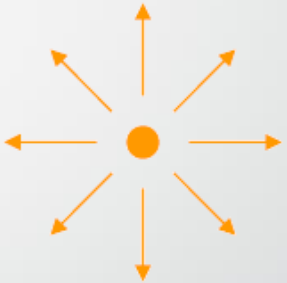
- Réflexion ambiante
  - Dans la réalité, les faces non éclairées directement par la lumière n'apparaissent pas noires, elles reçoivent une partie de la lumière réfléchie par les objets qui les entourent.
  - ➢ Approximation arbitraire de l'illumination indirecte en ajoutant un terme de valeur fixée.



Matériaux

- Paramètres

Dans les équations, on suppose que les vecteurs sont normalisés

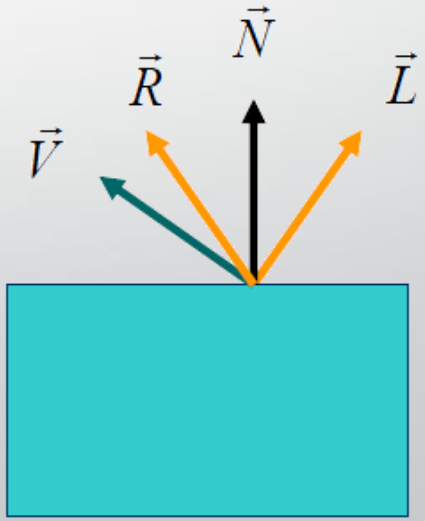


$\vec{V}$: direction du point de vue

$\vec{L}$: direction de la source lumineuse

$\vec{R}$: direction réfléchie de la lumière ($\vec{R} = 2\vec{N}(\vec{N}.\vec{L}) - \vec{L}$)

$\vec{N}$: direction normale à la surface



Matériaux

- Surface lambertienne
  - Réflexion diffuse proportionnelle au cosinus de l'angle entre la direction de la lumière et la normale à la surface

  Pour chaque lumière, on calcul l'intensité diffuse et on somme comme suit :

$$I_d = \sum_n f_{att_i} . I_i . (\vec{N} . \vec{L_i})$$

L'intensité finale donne donc :

$$I = k_a . I_a + k_d . I_d$$

avec k_a le coefficient de réflexion ambiante, k_d le coefficient de réflexion diffuse et f_{att_i} la fonction d'atténuation de la lumière i

Matériaux

- Modèle de Gouraud/Phong
  - On ajoute le terme spéculaire
    - Gouraud : Intensité spéculaire calculée au sommet puis interpolée le long de la surface
    - Phong : Intensité spéculaire calculée pour chaque pixel, en interpolant la normale. Il est plus couteux en calcul mais adapté au raytracing et aux cartes graphiques modernes.

  Pour chaque lumière, on calcul l'intensité spéculaire et on somme comme suit :

$$I_s = \sum_n f_{att_i} . I_i . \left(\vec{V} . \overrightarrow{R_i}\right)^n$$

  n est l'exposant spéculaire (shininess). Il a une valeur comprise entre 0,of et 100,of.

L'intensité finale donne : $I = k_a . I_a + k_d . I_d + k_s . I_s$ *où k_s est le coefficient de réflexion spéculaire*

Matériaux

- Modèle de Blinn-Phong
  - Dans le terme spéculaire, on remplace le produit scalaire $\vec{V}.\vec{R}$ par $\vec{N}.\vec{H}$ où $\vec{H}$ est le vecteur médian de $\vec{L}$ et $\vec{V}$

$$I_s = \sum_n f_{att_i}.I_i.\left(\vec{N}.\overrightarrow{H_i}\right)^n \qquad \vec{H} = \frac{\vec{L}+\vec{V}}{\left\|\vec{L}+\vec{V}\right\|}$$

  - Plus réaliste pour certains matériaux, moins coûteux en calcul (pas de vecteur réfléchi à calculer). C'est le modèle utilisé dans les fonctions fixes d'OpenGL et Direct3D

Matériaux

- Note

  Tous les modèles de matériaux sont données en luminance. Pour calculer des images colorées, on calcule les termes des équations pour chaque composante.

  - Ce qui donne pour la diffusion lambertienne du canal rouge :

$$C_d(rouge) = C_{d_{mat}}(rouge).\sum_n f_{att_i}.C_i(rouge).(\vec{N}.\overrightarrow{L_i})$$

  - En écriture vectorielle :

$$\overrightarrow{C_d}\begin{pmatrix} r \\ v \\ b \end{pmatrix} = \overrightarrow{C_{d_{mat}}}\begin{pmatrix} r \\ v \\ b \end{pmatrix} \otimes \left(\sum_n f_{att_i}.C_i\begin{pmatrix} r \\ v \\ b \end{pmatrix}.(\vec{N}.\overrightarrow{L_i})\right)$$

  avec $\otimes$ le produit élément par élément comme suit

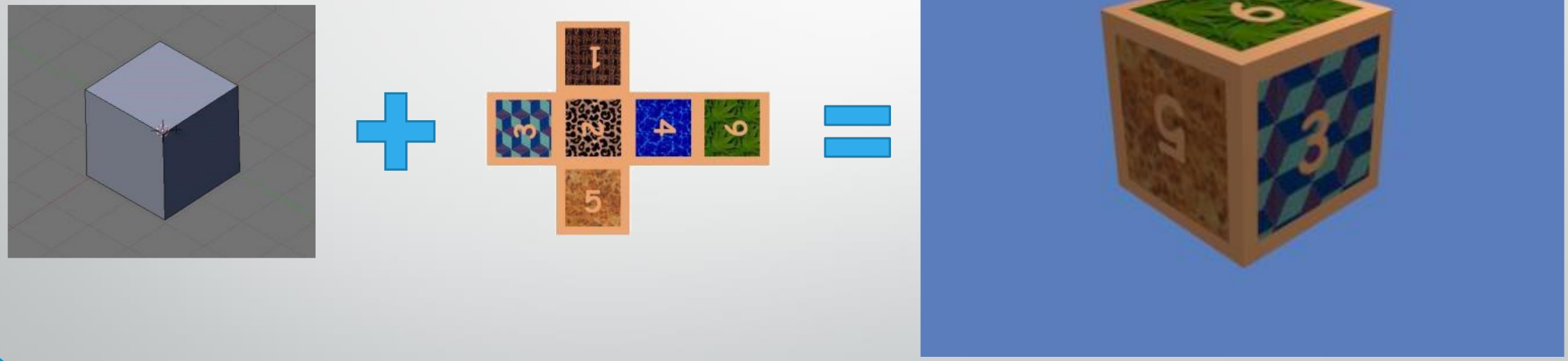
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} \otimes \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x.a \\ y.b \\ z.c \end{bmatrix}$$

Textures

- Principe : modifier les paramètres des matériaux le long de la surface en allant chercher les valeurs dans une image.
- On établie une correspondance entre la surface 3D et la texture à appliquer
  - Coordonnées de texture

Textures

- Exemple : un cube et son patron



Textures

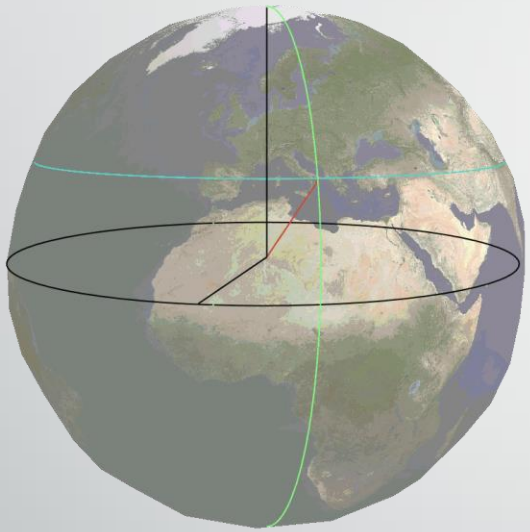
- Problèmes sur les surfaces plus complexes.
  - Exemple : sphère

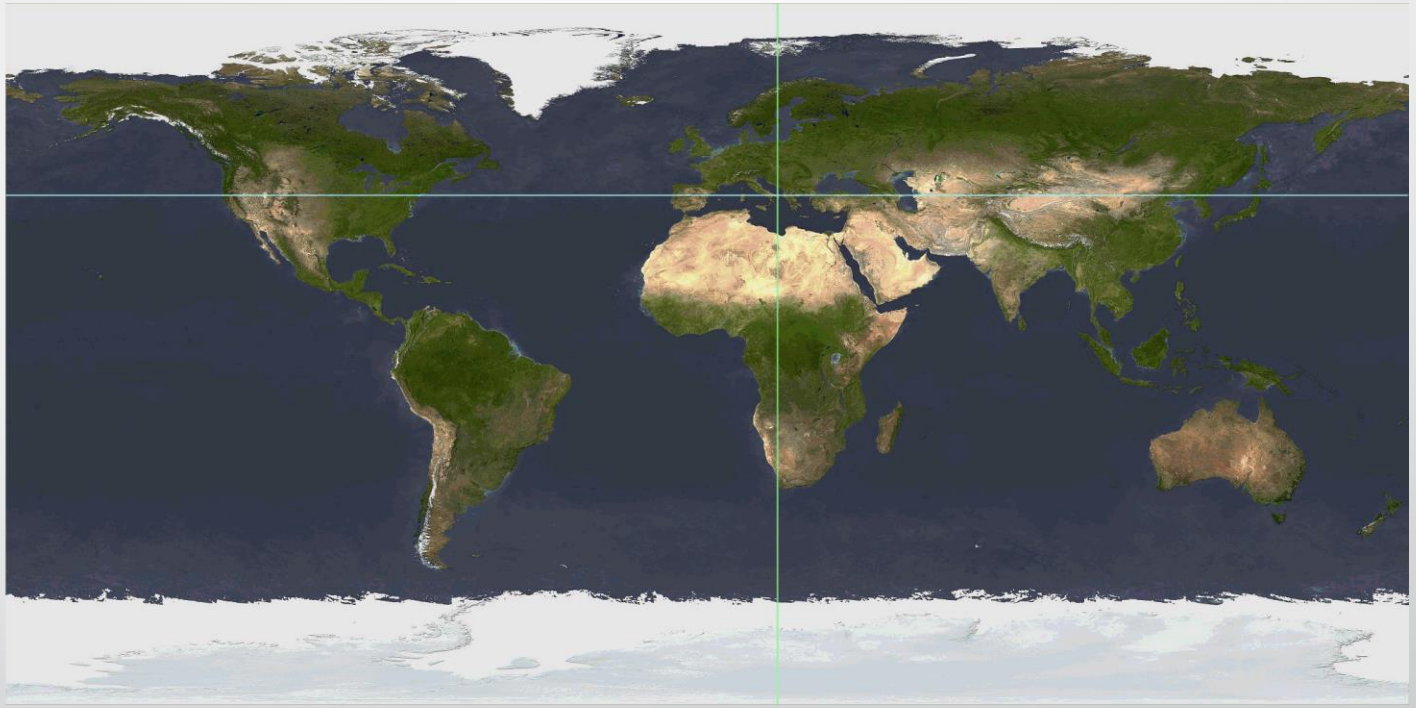
$$u = \sin\theta \,.\cos\phi = \frac{x}{\sqrt{x^2 + y^2 + z^2}}$$

$$v = \sin\theta \,.\sin\phi = \frac{y}{\sqrt{x^2 + y^2 + z^2}}$$

  - Mais on admet des déformations : aux pôles, une grande partie de l'image est contractée.

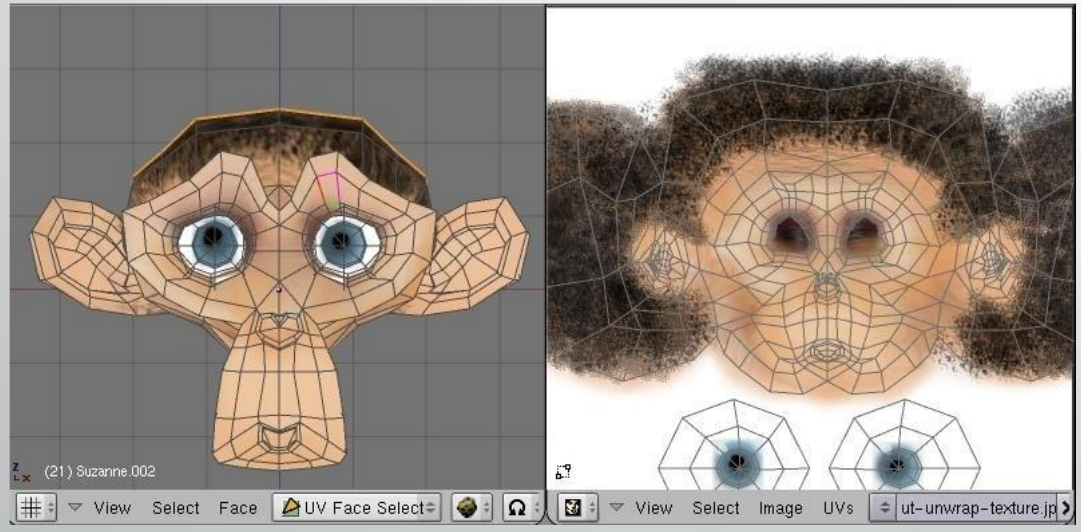
Textures





Textures

- Pour des meshes encore plus compliqués, impossible d'automatiser la génération des coordonnées UV
  - Fait à la main par les infographistes



Introduction à l'imagerie numérique
Synthèse 3D

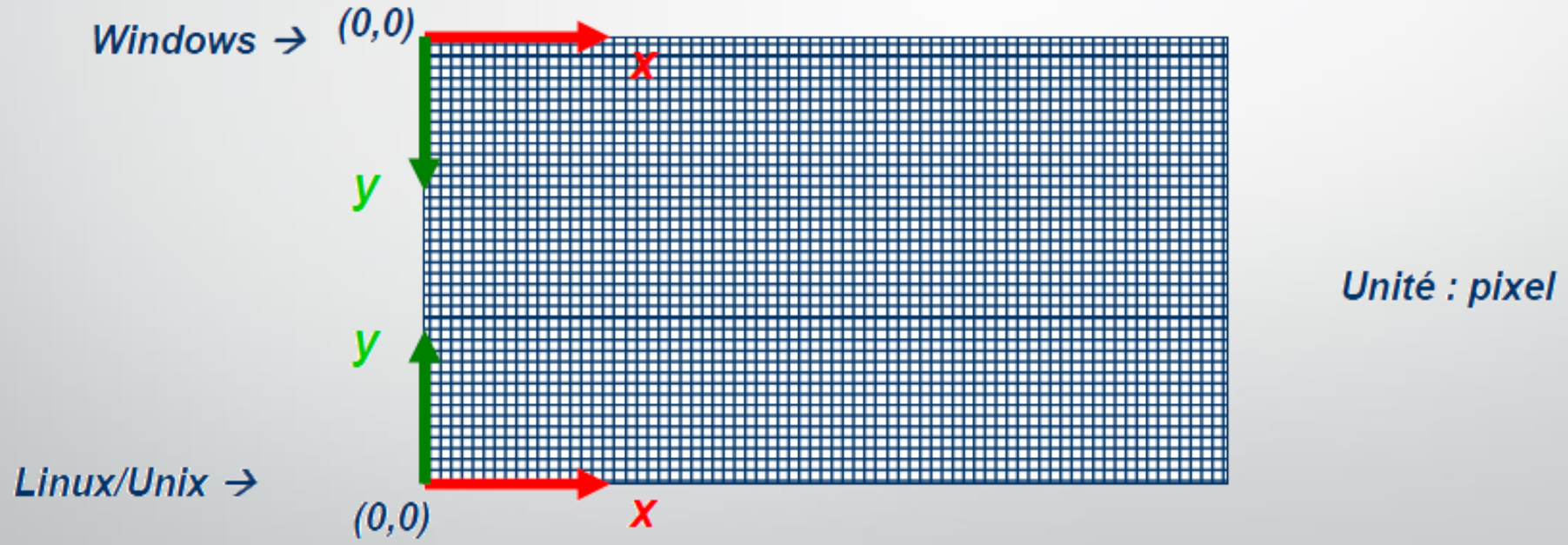
Mathématiques

Mathématiques

- Besoin d'outils mathématiques pour le calcul dans l'espace
  - Repères
    - Repère de l'écran
    - Repère de l'image
    - Repère de la scène
    - Repère de l'objet
  - Transformations classiques
    - Translation
    - Rotations
      - Matrices
      - Axes
      - Quaternions
  - Géométrie projective et matrices homogènes

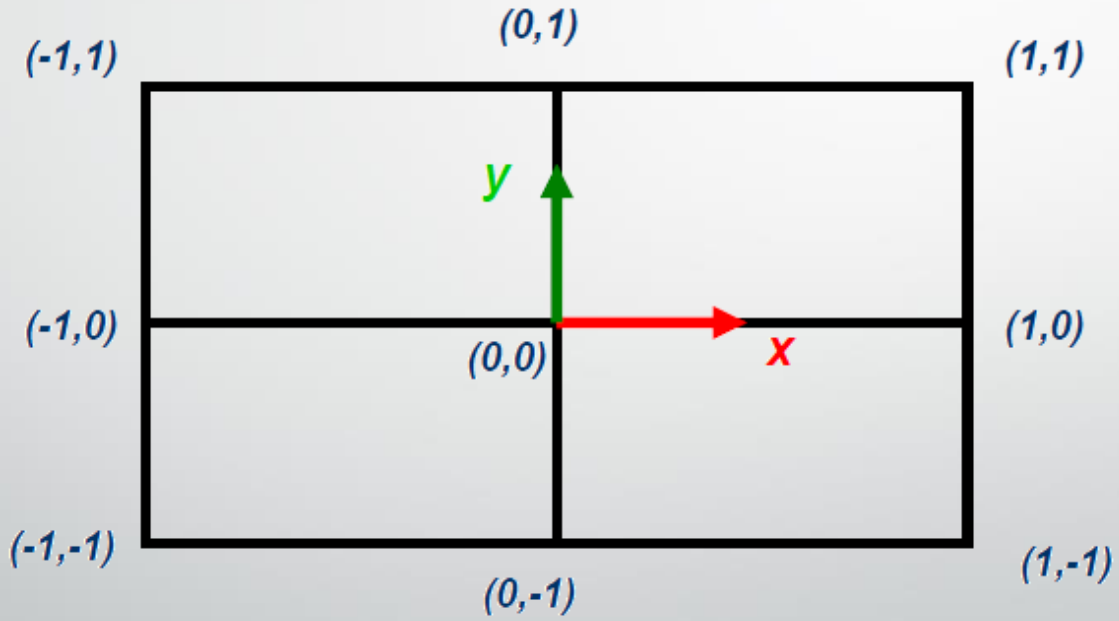
Repères

- Repère écran :



Repères

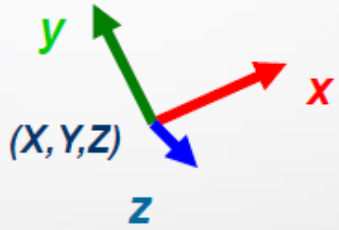
- Repère image (plan de projection) :

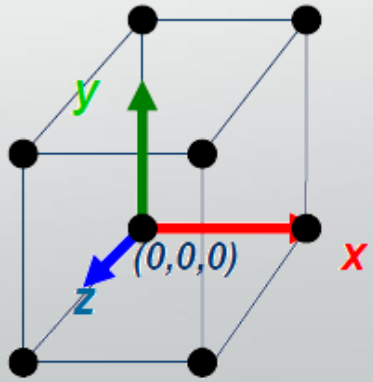


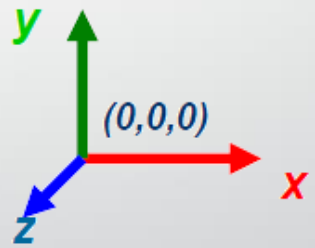
Unité : celle fixée dans le monde

Repères

- Repère de la scène (ou repère monde)
  - Référence dans laquelle les coordonnées des objets sont exprimées (camera, lumières et objets 3D)
- Repère objet
  - Repère dans lequel les vertices de l'objet sont exprimées







Transformations classiques

- Translations

$$M'\begin{bmatrix}x'\\y'\\z'\end{bmatrix} = M\begin{bmatrix}x\\y\\z\end{bmatrix} + T\begin{bmatrix}t_x\\t_y\\t_z\end{bmatrix} = \begin{bmatrix}x+t_x\\y+t_y\\z+t_z\end{bmatrix}$$

- Rotations

$$R_x(\alpha) = \begin{bmatrix}1 & 0 & 0\\0 & \cos(\alpha) & -\sin(\alpha)\\0 & \sin(\alpha) & \cos(\alpha)\end{bmatrix} R_y(\beta) = \begin{bmatrix}\cos(\beta) & 0 & \sin(\beta)\\0 & 1 & 0\\-\sin(\beta) & 0 & \cos(\beta)\end{bmatrix} R_z(\gamma) = \begin{bmatrix}\cos(\gamma) & -\sin(\gamma) & 0\\\sin(\gamma) & \cos(\gamma) & 0\\0 & 0 & 1\end{bmatrix}$$

$$M'\begin{bmatrix}x'\\y'\\z'\end{bmatrix} = R_x\begin{bmatrix}1 & 0 & 0\\0 & \cos(\alpha) & -\sin(\alpha)\\0 & \sin(\alpha) & \cos(\alpha)\end{bmatrix}.M\begin{bmatrix}x\\y\\z\end{bmatrix}$$

Transformations classiques

- Propriétés des matrices de rotation
  - Ne commutent pas !!! Attention à l'ordre de composition.
  - L'inverse est égale à la transposée. Trouver le transposer est plus rapide que de calculer son inverse. On peut voir la matrice de rotation comme une matrice de passage d'un repère cartésien à un autre.
  - Le déterminant vaut 1.

Transformations classiques

- Autres représentations des rotations
  - Rotation autour d'un axe
    - On donne une direction (vecteur normalisé) et un angle de rotation autour de cet axe.
  - Angle d'Euler
    - Compositions successives de rotations
  - Quaternion
    - Objets mathématiques ayant une algèbre propre
    - Quadruplet (x, y, z, w)
    - ➢ Algorithmes et conversions sur le net (FAQ quaternion)

Transformations classiques

- Echelle/homothétie par rapport à l'origine

$$M' \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = H_{xyz} \begin{bmatrix} h_x & 0 & 0 \\ 0 & h_y & 0 \\ 0 & 0 & h_z \end{bmatrix} . M \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} h_x . x \\ h_y . y \\ h_z . z \end{bmatrix}$$

Transformations classiques

- Projection
  - La projection d'un point 3D sur un plan 2D peut être retrouvée à l'aide des équations de projection vues précédemment à savoir :

$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z}$$

Problème : pas de représentation vectorielle possible

Transformations classiques

- Constat :
  - Représentation des opérations de transformations sur des dimensions différentes.
  - Opérateurs à utiliser différents selon la transformation voulue.
  - Problèmes de représentation informatique de ces transformations.

➢ Solution : utiliser une représentation homogène des transformations et utiliser le même opérateur mathématiques pour les composer.

Représentation homogène

- Représentation homogène des transformations
  - Résultats d'une branche particulière des mathématiques : la géométrie projective.
  - Représentation des vecteurs en coordonnées homogènes : on ajoute une coordonnée supplémentaire aux coordonnées cartésiennes

Cartésien vers homogène $[x \quad y \quad z] \rightarrow [x \quad y \quad z \quad 1]$

Homogène vers cartésien $[x \quad y \quad z \quad w] \rightarrow [x/w \quad y/w \quad z/w]$

Remarque : Il y a une infinité de représentations d'un point cartésien en coordonnées homogènes

$$[x \quad y \quad z \quad w] \Leftrightarrow [k.x \quad k.y \quad k.z \quad k.w]$$

Représentation homogène

- Translations

$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

On obtient X', le translaté de X par T avec la relation suivante : $X' = T.X$

On a bien :

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ z + t_z \\ 1 \end{bmatrix}$$

Représentation homogène

- Rotations

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

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

On obtient X', la transformée de X par R avec la relation suivante : $X' = R.X$

Représentation homogène

- Echelles/homothéties

$$H_{xyz} = \begin{bmatrix} h_x & 0 & 0 & 0 \\ 0 & h_y & 0 & 0 \\ 0 & 0 & h_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

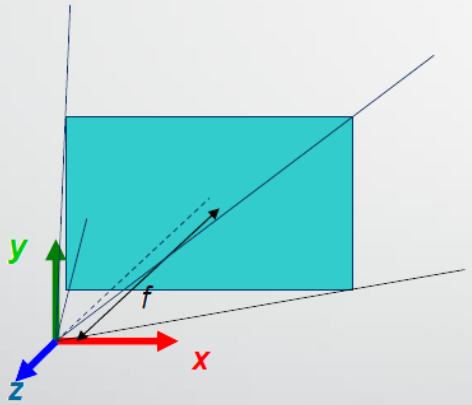
On obtient X', la transformée de X par H avec la relation suivante : $X' = H.X$

On a bien :

$$\begin{bmatrix} h_x & 0 & 0 & 0 \\ 0 & h_y & 0 & 0 \\ 0 & 0 & h_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x.h_x \\ y.h_y \\ z.h_z \\ 1 \end{bmatrix}$$

Représentation homogène

- Projection perspective
  - Par rapport à l'origine, sur un plan orthogonal à z, située à une distance f de l'origine, orientée vers z (repère caméra standardisé)



$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & {}^{1}/_{f} & 0 \end{bmatrix}$$

On obtient X', le projeté de X par P avec la relation suivante : $X' = P.X$

Représentation homogène

- Projection perspective
  - On a bien :

$$X' = P\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 1/f & 0\end{bmatrix}.X\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix} = \begin{bmatrix}x\\ y\\ z\\ z/f\end{bmatrix} = \begin{bmatrix}f.x/z\\ f.y/z\\ f\\ 1\end{bmatrix}$$

  - Cas particulier : projection orthographique

$$X' = P\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & f\\ 0 & 0 & 0 & 1\end{bmatrix}.X\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix} = \begin{bmatrix}x\\ y\\ f\\ 1\end{bmatrix}$$

Mathématiques

- Transformation des sommets (vertices) d'un objet

$$X' = P.C^{-1}.M.X$$

  - Avec C la matrice de transformation de la caméra, M la matrice de transformation de l'objet toutes deux exprimées dans le repère monde
  - Remarque 1 : X' est exprimé en coordonnées homogènes dans le repère image, les coordonnées x et y doivent être divisées par w avant d'être utilisées
  - Remarque 2 : le produit $C^{-1}.M$ est souvent représenté sous forme d'une seule matrice appelée « ModelView Matrix » en OpenGL par exemple.

Introduction à l'imagerie numérique
Synthèse 3D

Algorithme de rendu

Clipping

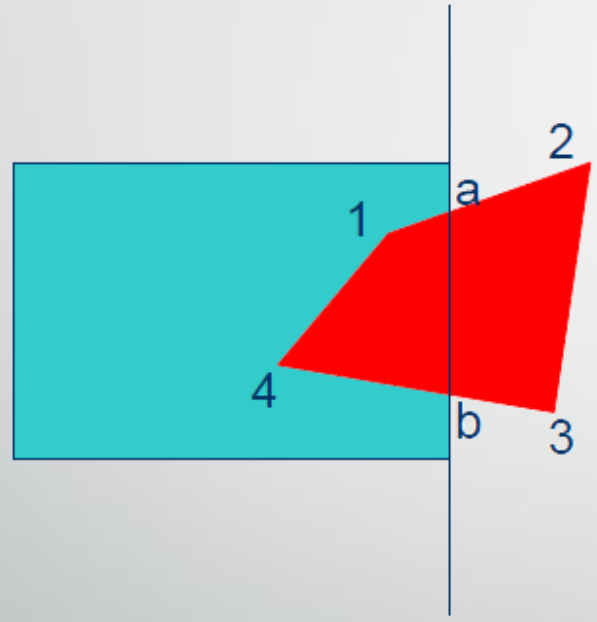
- Projection sur un plan image → comment traiter les points en dehors?
  - Clipping : Sutherland-Hodgeman
  - But : éliminer les parties du polygone en dehors de l'image

Clipping

- On traite successivement chaque bord de l'image.
- Pour chaque segment du polygone, 4 possibilités :
  - Départ et arrivée dans l'image
    - Ajout du point d'arrivée au polygone clippé.
  - Départ dans l'image et arrivée en dehors
    - Ajout de l'intersection entre le segment et le bord de l'image au polygone clippé.
  - Départ et arrivée hors de l'image
    - Ne rien ajouter au polygone clippé.
  - Départ hors de l'image et arrivée dedans
    - Ajout de l'intersection entre le segment et le bord de l'image ET le point d'arrivée au polygone clippé.

Clipping

- Exemple de clipping



itération	Segment	Départ	Arrivée	Polygone clippé
1	1,2	Dedans	Dehors	a
2	2,3	Dehors	Dehors	a
3	3,4	Dehors	Dedans	a,b,4
4	4,1	Dedans	Dedans	a,b,4,1

Rastérisation

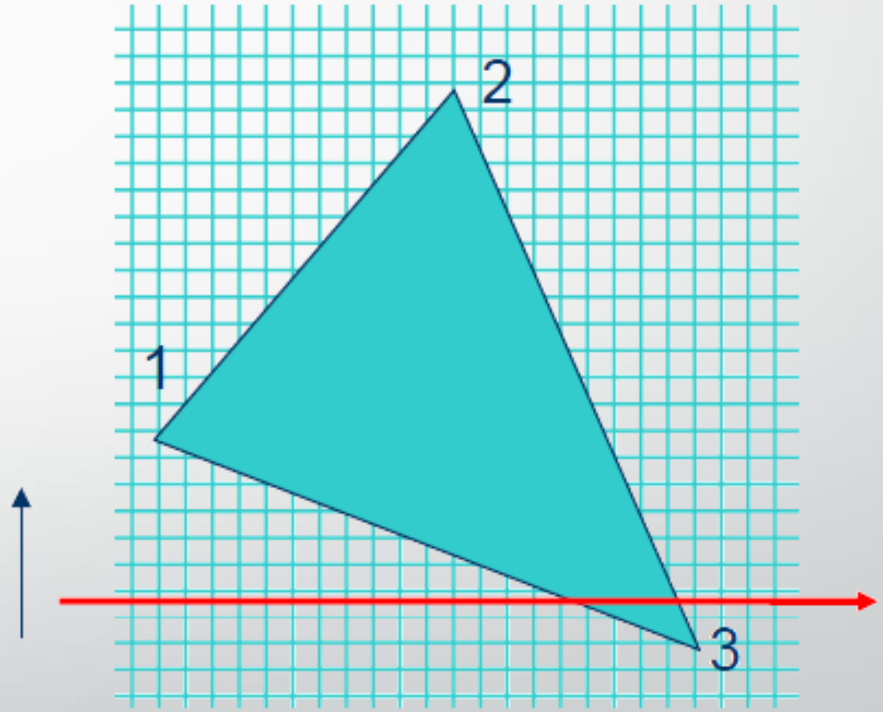
- Remplissage (rendu 2D) d'un polygone
  - Procédé dit de rastérisation : conversion de données vectorielles en une image matricielle
  - Méthode la plus répandue : algorithme Scanline
    - Principe : balayer le polygone par des lignes horizontales et déterminer pour chaque pixel de la ligne si l'on se trouve à l'intérieur ou à l'extérieur du polygone

Rastérisation

- Exemple sur un triangle
  - Pour déterminer si un pixel est à l'intérieur du triangle
    - Tests sur les équations des droites des segments du triangle

    OU

    - Test avec la méthode des aires



Rastérisation

- Equations des segments
  - De la forme

$$y = ax + b$$

  - Avec deux points, se retrouve trivialement en résolvant

$$\begin{cases} y_1 = ax_1 + b \\ y_2 = ax_2 + b \end{cases}$$

  - Appartenance au demi-plan

$$ax - y + b > 0$$

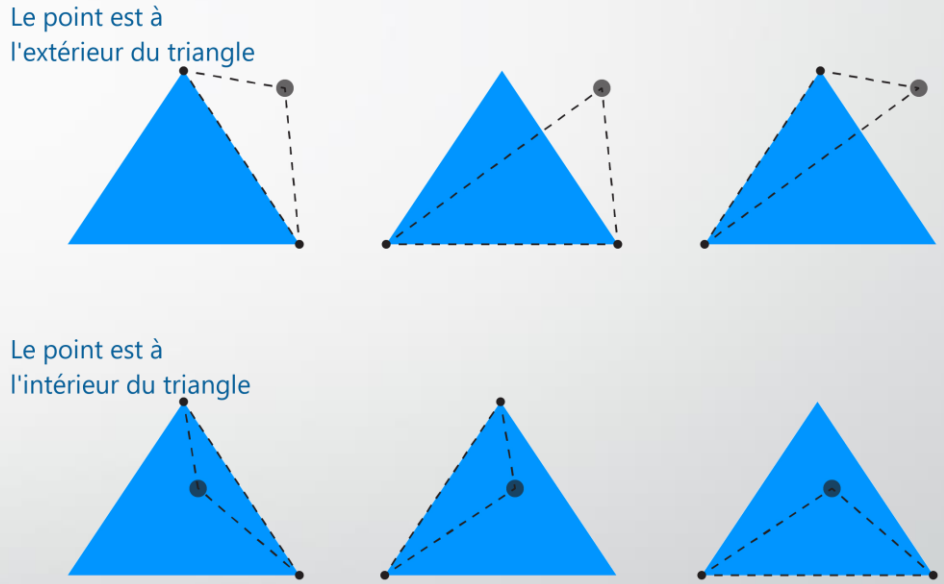
  - Donc un point (x, y) appartient au triangle s'il satisfait au système

$$\begin{cases} a_1x - y + b_1 > 0 \\ a_2x - y + b_2 > 0 \\ a_3x - y + b_3 > 0 \end{cases}$$

Remarque : l'ordre des points détermine le sens de l'inégalité

Rastérisation

- Méthode des aires
  - On considère les aires des triangles formés avec le point à tester et les sommets du triangle.
  - Si la somme des aires des triangles inscrits est égale à l'aire du triangle, le point P est à l'intérieur



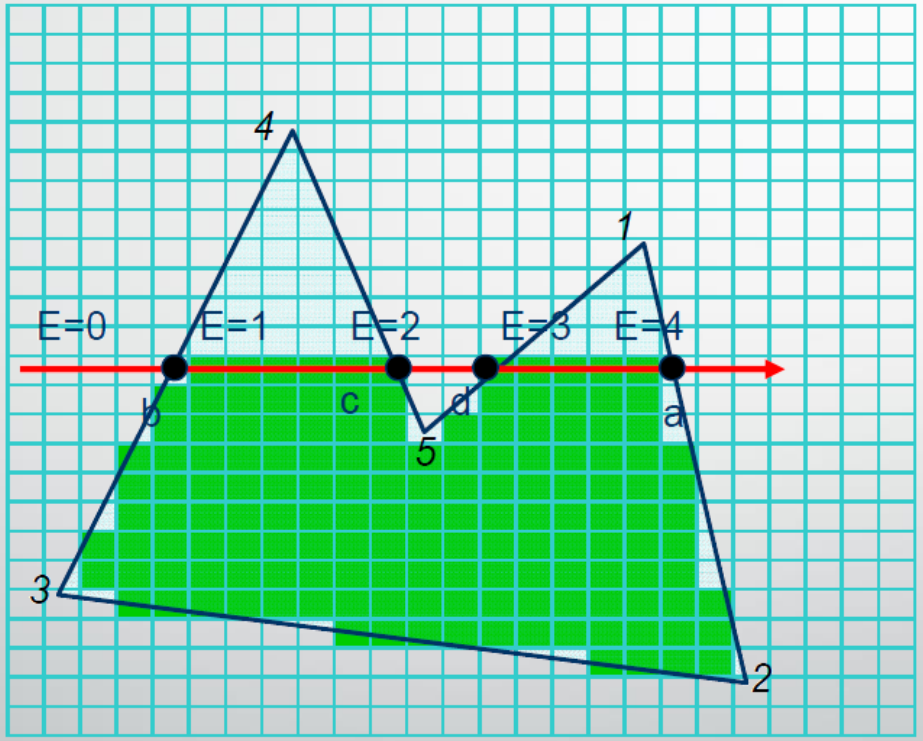
NB : cette méthode est aussi très utile pour le raytracing !

Rastérisation

- Généralisation du scanline à des polygones quelconques
  - Maintenir une liste des segments intersectés par la scanline
  - Trier la liste en fonction de l'intersection sur x avec les segments (un tri à bulle marche très bien ici)
  - Compter le nombre de segments traversés
    - Si le nombre est impair, on est dans le polygone
    - Sinon on est dehors

Rastérisation

- Exemple : scanline d'un polygone quelconque



E : nombre de segments traversés

Liste triée :

Segment	Intersection X
3,4	b
4,5	c
5,1	d
1,2	a

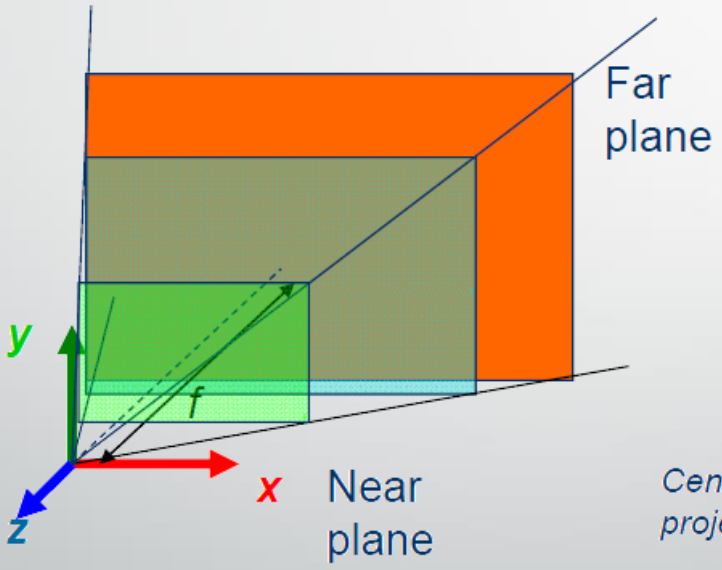
Rastérisation

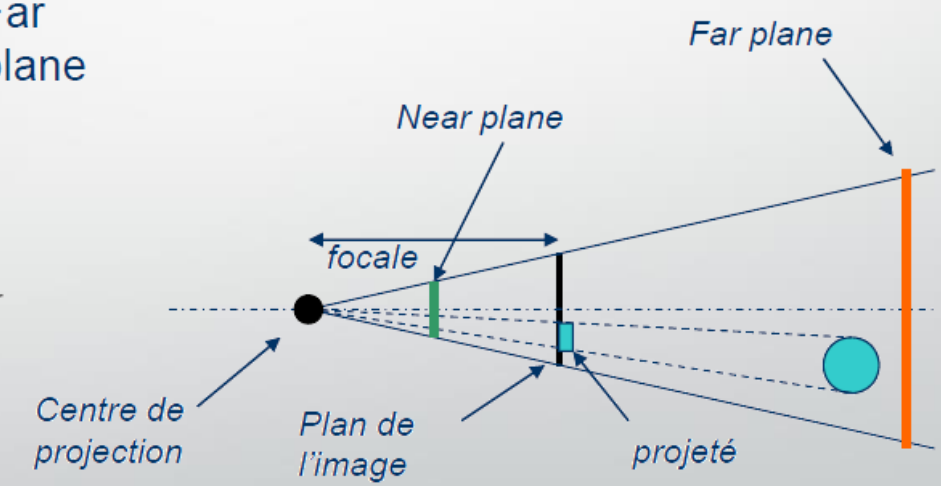
- Jusque là, on sait transformer, projeter, clipper et dessiner les polygones dans le plan image
  - Cependant, comment faire si deux polygones sont projetés au même endroit sur l'image ou s'ils se croisent dans l'espace 3D ? Comment déterminer les parties qui sont cachées ?

➢ Maintenir pour chaque pixel sa profondeur dans un buffer puis tester pour chaque pixel à dessiner s'il est devant ou derrière un pixel éventuellement déjà dessiné, c'est le Z buffer

Rastérisation

- On restreint le volume de projection entre 2 plans de l'espace parallèle au plan image : le plan de clipping proche et le plan de clipping lointain (near et far)



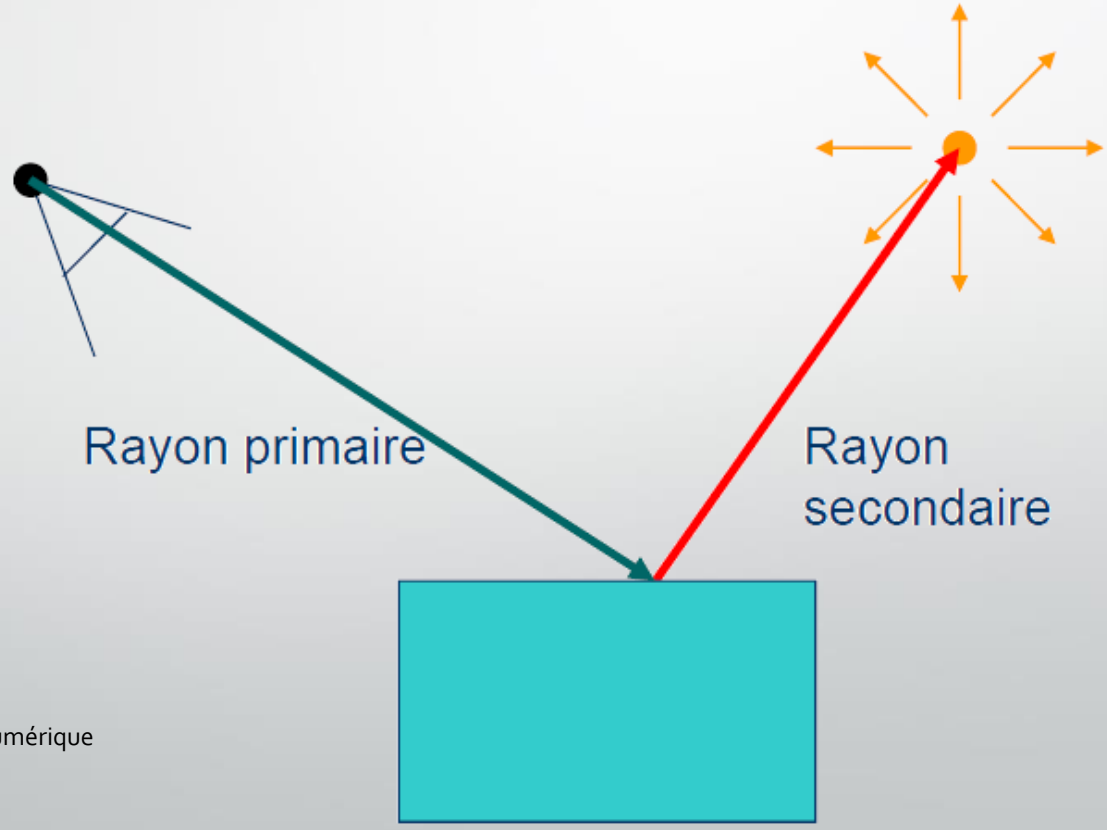


Raytracing

- Méthode de rendu par lancer de rayons
  - Beaucoup plus proche de la réalité : modélisation du comportement de la lumière.
  - Beaucoup plus simple à implémenter que les rendus par scanline.
  - Permet d'atteindre un grand niveau de réalisme au détriment des performances (effets spéciaux cinéma, films en images de synthèse, … → plusieurs minutes de calcul par image)
  - De façon générale, pas adapté au rendu temps réel (des exceptions existent, cf. OpenRT)
  - Permet de gérer intrinsèquement des effets complexes comme la réflexion, la réfraction et les ombres portées.
  - Supporte intrinsèquement les objets définis par des équations (pas besoin de les transformer en ensemble de facettes avant de les afficher).

Raytracing

- Principe : effectuer le chemin inverse de la lumière



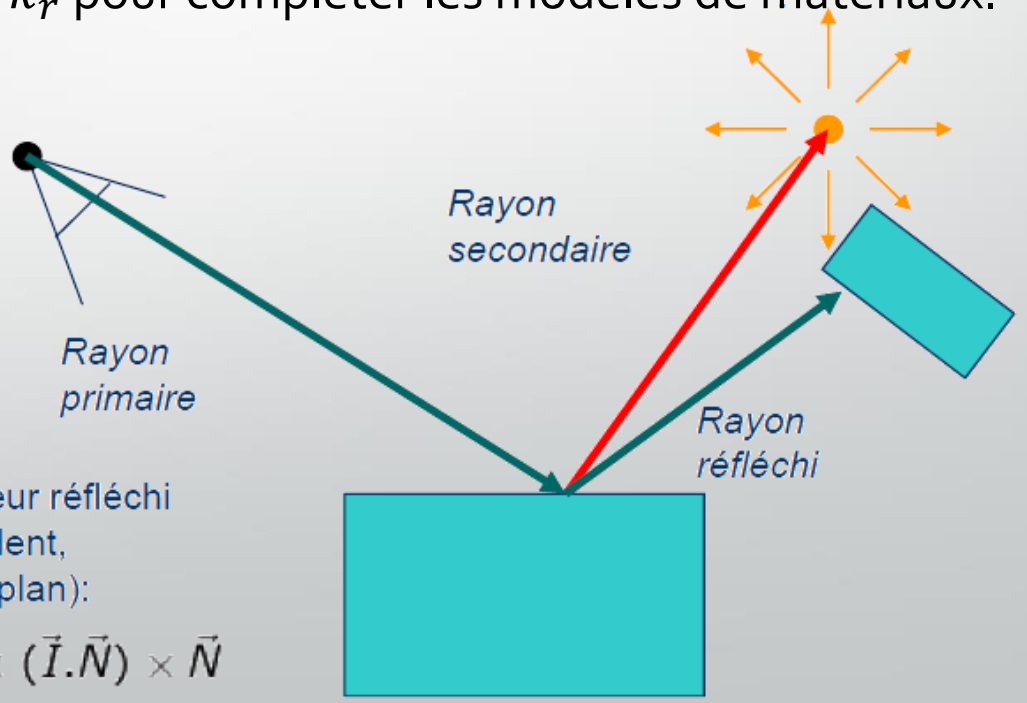
Raytracing

Algorithme de base

- Pour chaque pixel de l'image.
  - Calculer le rayon primaire.
  - Trouver les intersections du rayon primaire avec les objets de la scène.
  - Trier les intersections selon la profondeur, et récupérer le plus proche.
  - Pour chaque lumière de la scène.
    - Calculer le rayon secondaire.
    - Si le rayon n'intersecte pas d'objet entre son point de départ et la source lumineuse, calculer la contribution de la lumière selon le modèle de matériau utilisé.
  - Calculer la couleur finale et la stocker dans le pixel.

Raytracing

- Amélioration
  - Calcul des réflexions : si le matériau de l'objet touché par un rayon primaire est réflectif, on relance un rayon primaire dans la direction réfléchie du point de vue. On se donnera un coefficient de réflexion k_r pour compléter les modèles de matériaux.

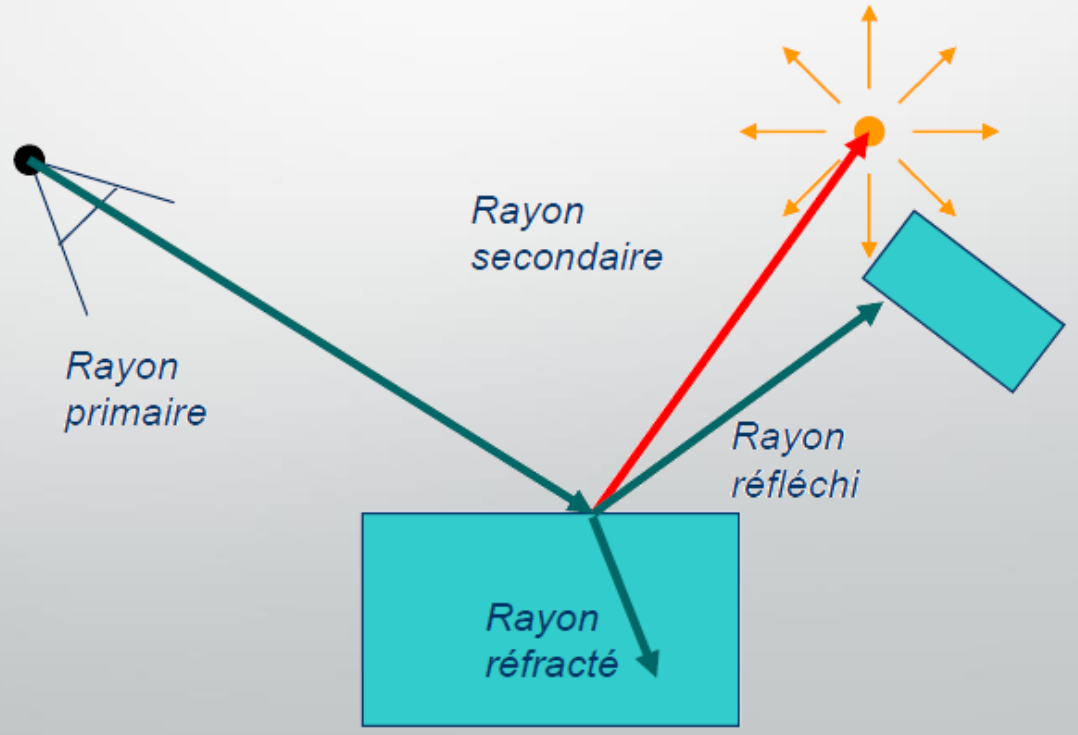


Calcul du vecteur réfléchi
(I=vecteur incident,
N=normale au plan):

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

Raytracing

- Réfraction
  - On calcul le rayon réfracté selon la loi de Snell et on lance un rayon primaire dans cette direction



Raytracing

- Loi de Snell Descartes

$$n_1.\sin(\theta_1)=n_2.\sin(\theta_2)$$

et

$$\overrightarrow{D_2}=\left(\frac{n_1}{n_2}\right).\overrightarrow{D_1}+\left(\frac{n_1}{n_2}\cos\theta_1-\cos\theta_2\right).\vec{N}$$