Transformations géométriques

Zerari Abd El Mouméne

Cours de master – M1(Option: I.V.A.)

Université Mohamed Khider Biskra

2019 – 2020

Dr: Zerari Abd El Mouméne

Matrice

- Une matrice est un ensemble d'éléments organisé en lignes et en colonnes
- matrice *m*×*n*

$$\begin{array}{cc} & \overbrace{\hphantom{a_{00} \quad a_{01}}}^{n \text{ colonnes}} \\ m \text{ lignes} \left\{ \vphantom{\begin{bmatrix} a_{00} \\ a_{10} \end{bmatrix}} \right. & \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \end{array}$$

- Vecteur: matrice *n*×1

Opérations de bases (1)

- Transposée: faire l'échange des lignes par les colonnes

$$M = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \qquad M^T = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

$$V = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad V^T = \begin{bmatrix} x & y & z \end{bmatrix}$$

Opérations de bases (2)

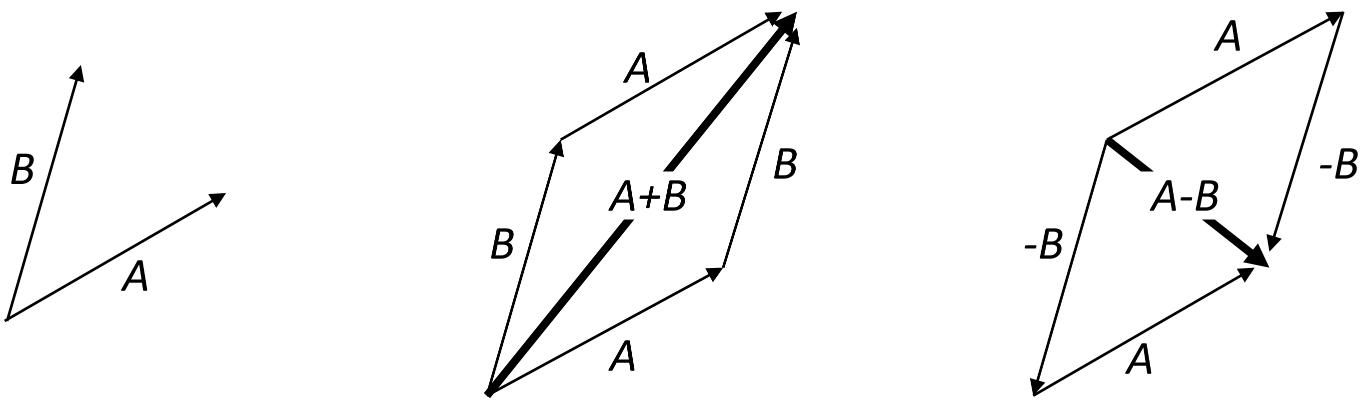
- Addition et soustraction

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$$

addition des éléments

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} - \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a-e & b-f \\ c-g & d-h \end{bmatrix}$$

Soustraction des éléments



Opérations de bases (3)

- Multiplication
  - Multiplier chaque ligne par chaque colonne
  - une matrice *m×n* peut être multiplier par une matrice *n×p* pour obtenir une matrice résultat *m×p*

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} e & f \\ g & h \end{bmatrix}=\begin{bmatrix} ae+bg & af+bh \\ ce+dg & cf+dh \end{bmatrix}$$

Opérations de bases (1)

Multiplication

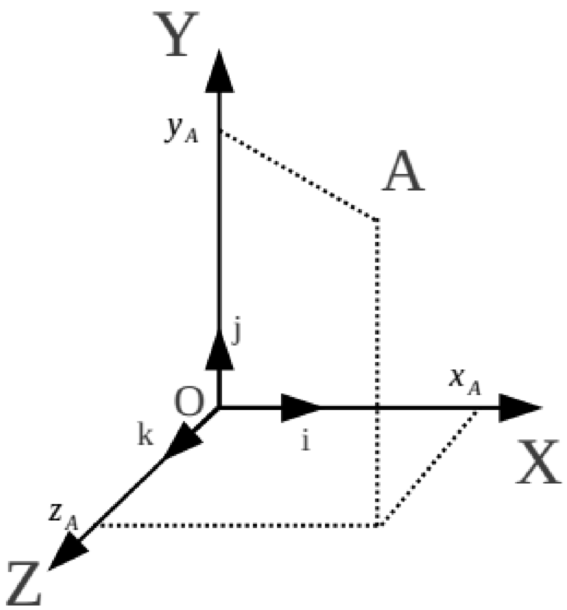
- Est ce que $AB = BA$? Peut être, mais peut être $$\begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae+bg & \ldots \\ \ldots & \ldots \end{bmatrix}$$

$$\begin{bmatrix} e & f \\ g & h \end{bmatrix}\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} ea+fc & \ldots \\ \ldots & \ldots \end{bmatrix}$$

- Alors: la multiplication n'est pas commutative!

Repères d'une scène 3D

Repère 3D



- Repère noté (O, i, j, k) (origine et vecteurs de base).
- Un point : $A = \overrightarrow{OA} = \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix}$ ou $A = (A_x, A_y, A_z)$.
- $A = O + xi + yj + zk$.

- Les repères considérés seront généralement directs :
  - Règle de la main **droite** : (Pouce,Index,Majeur) = (X, Y, Z)
- Un repère est dit orthonormé si :
  - Vecteurs (i, j, k) deux à deux orthogonaux.
  - (i, j, k) de même normes 1.

Repères d’une scène 3D

Scène 3D et repères

➢Pour la conception d’une scène 3D et de ses objets 3D, nous considérons les repères suivants :

- **Le repère de l’observateur** (ou **repère de la caméra** ; noté **Eye** dans la suite) : c’est dans ce repère qu’on définit le volume de visualisation (i.e. placement de l’écran et définition des paramètres de la projection).
- **Le repère local** (ou **repère objet** ; noté **Local** dans la suite) : on conçoit un objet indépendamment du reste de la scène. On prendra le repère le plus naturel pour définir les points de l’objet.
- **Le repère du monde** (ou **repère de scène** ; noté **World** dans la suite) : il s’agit du repère de référence de tout positionnement (les repères locaux et la caméra seront placés directement ou indirectement par rapport à World).

Repères d'une scène 3D

Scène 3D et repères (suite)

- Nous placerons les repères les uns par rapport aux autres avec des changements de repères (en général par translations, rotations, changements d'échelle). Le repère "**initial** " étant le repère de scène **World**.
- Les positions (x , y , z) des sommets des objets 3D sont données dans le repère Local de l'objet qu'ils définissent, et l'objet est placé dans la scène en déplaçant son repère local (et non directement/explicitement ses points)..
- Trois repères principaux :
  - ✓local : modèle 3D
  - ✓monde : utilisateur
  - ✓caméra : projection

Repères d'une scène 3D

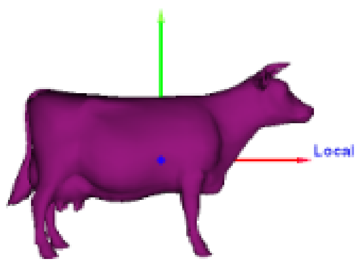
Scène 3D et repères (suite)

- Trois repères principaux :
  - ✓ local : modèle 3D
  - ✓ monde : utilisateur
  - ✓ caméra : projection
- $V_{local} \rightarrow V_{monde} \rightarrow V_{camera}$

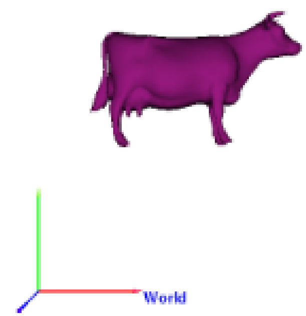
Repères d'une scène 3D

Exemple

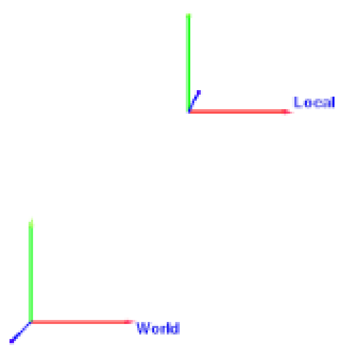
1) on dispose d'un objet défini dans son repère local



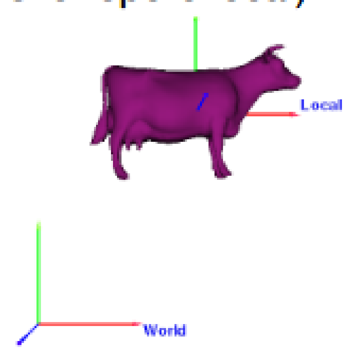
2) on souhaite



3) on place `Local` par rapport à `World`



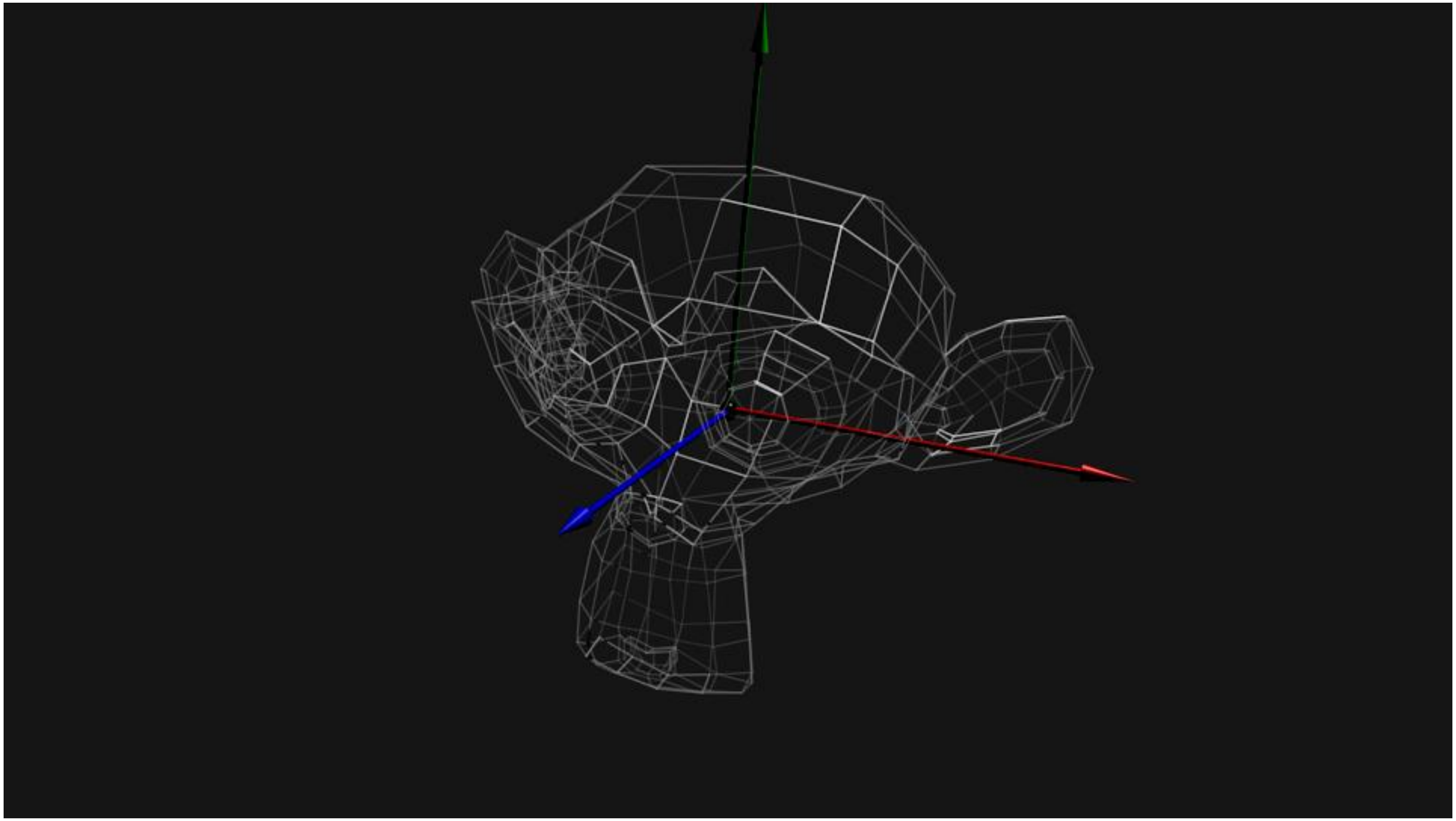
4) on obtient (points toujours exprimés dans le repère local)



Les matrices de modèle, de vue et de projection

La matrice de modèle

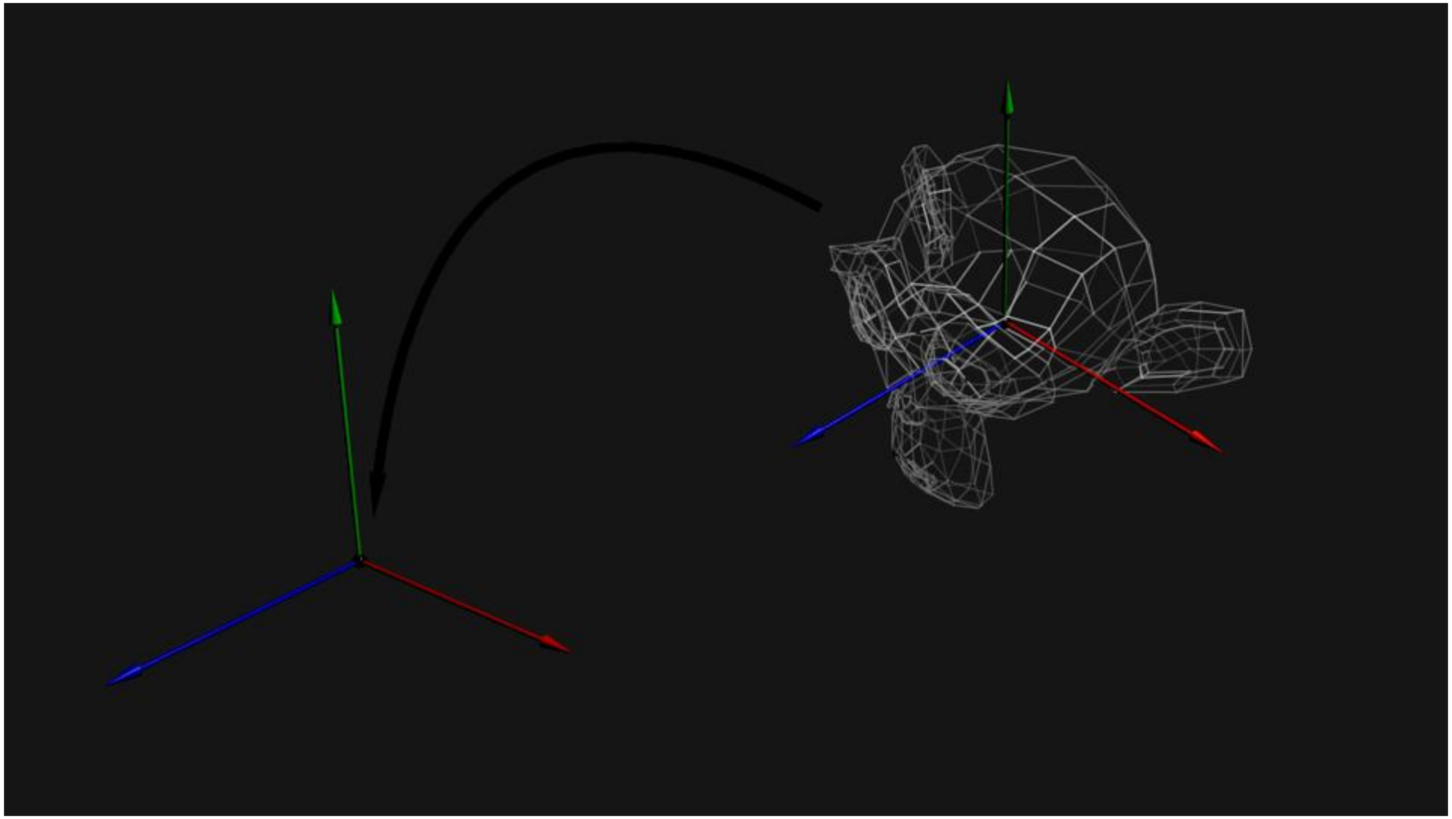
Le modèle est défini par un ensemble de sommets. Les coordonnées X, Y, Z de ces sommets sont définies par rapport au centre de l’objet.



Les matrices de modèle, de vue et de projection

La matrice de modèle (suite)

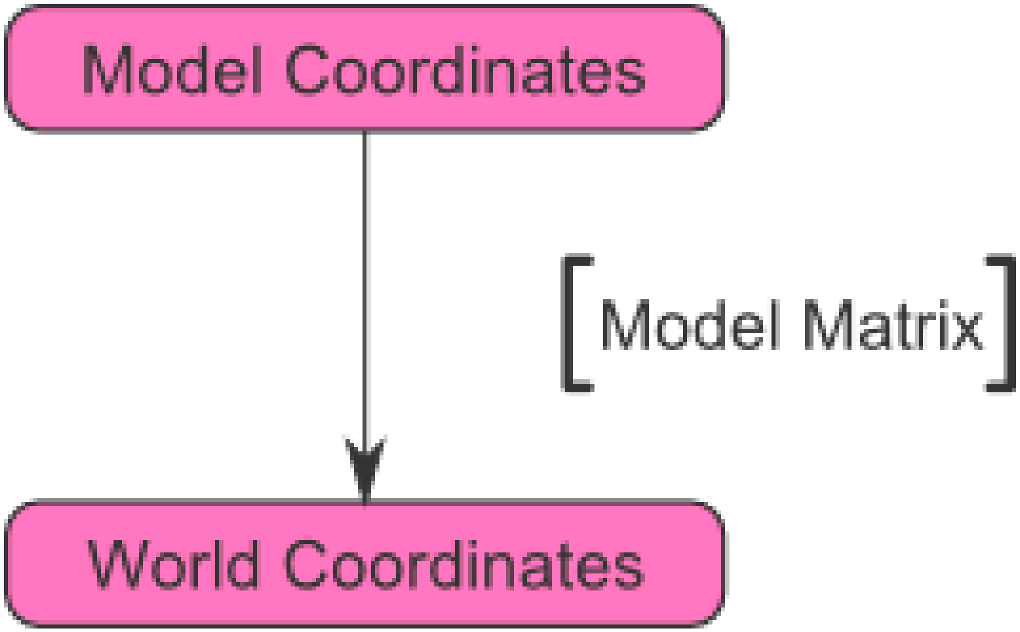
Nos sommets sont maintenant dans le repère du monde. C’est la signification de la flèche noire dans l’image ci-dessous : on s’est déplacé de l’espace du modèle (tous les sommets sont définis par rapport au centre du modèle) à l’espace monde (tous les sommets sont définis par rapport au centre du monde).



Les matrices de modèle, de vue et de projection

La matrice de modèle (suite)

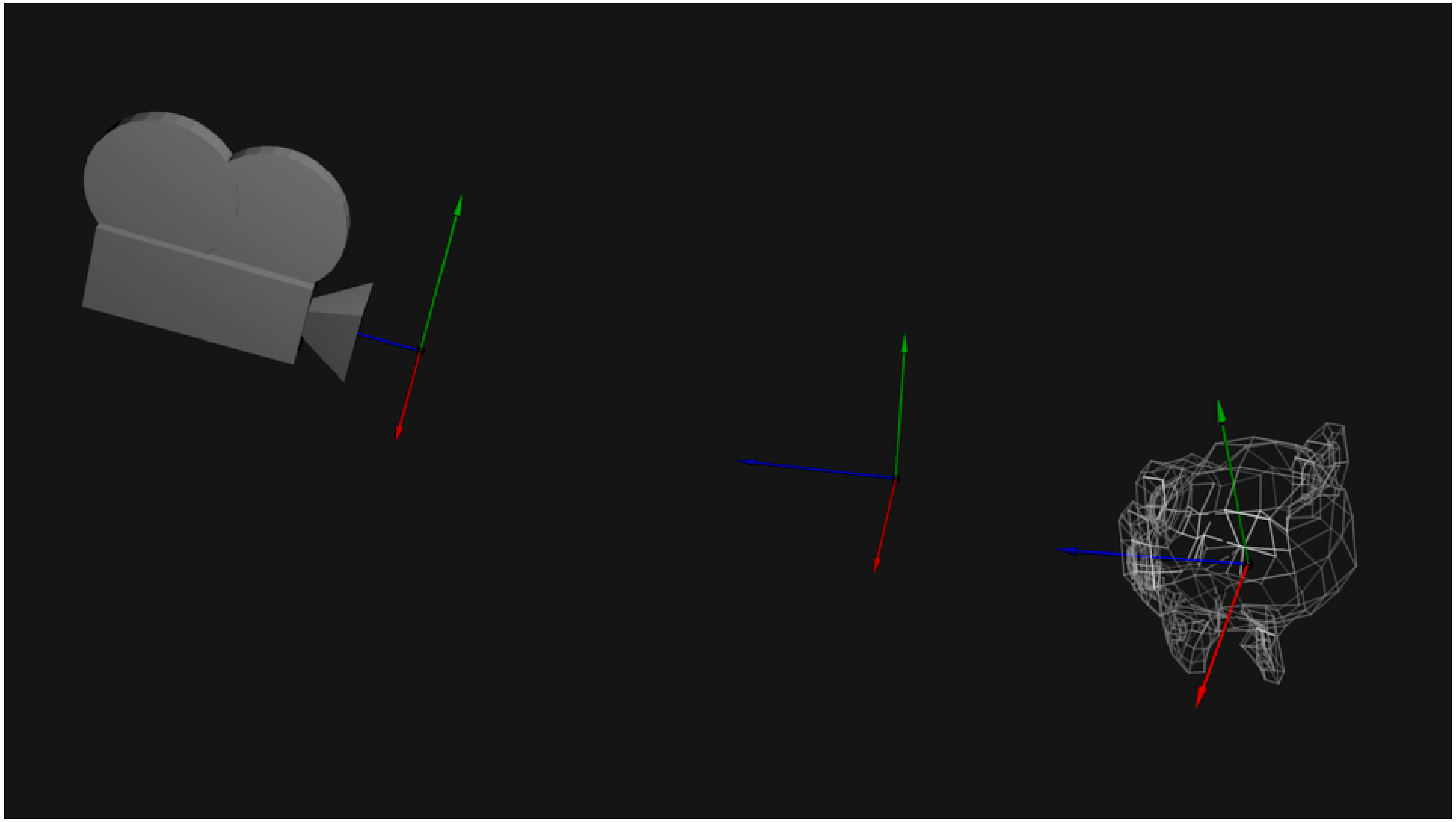
On peut résumer cela avec le diagramme suivant :



Les matrices de modèle, de vue et de projection

La matrice de vue

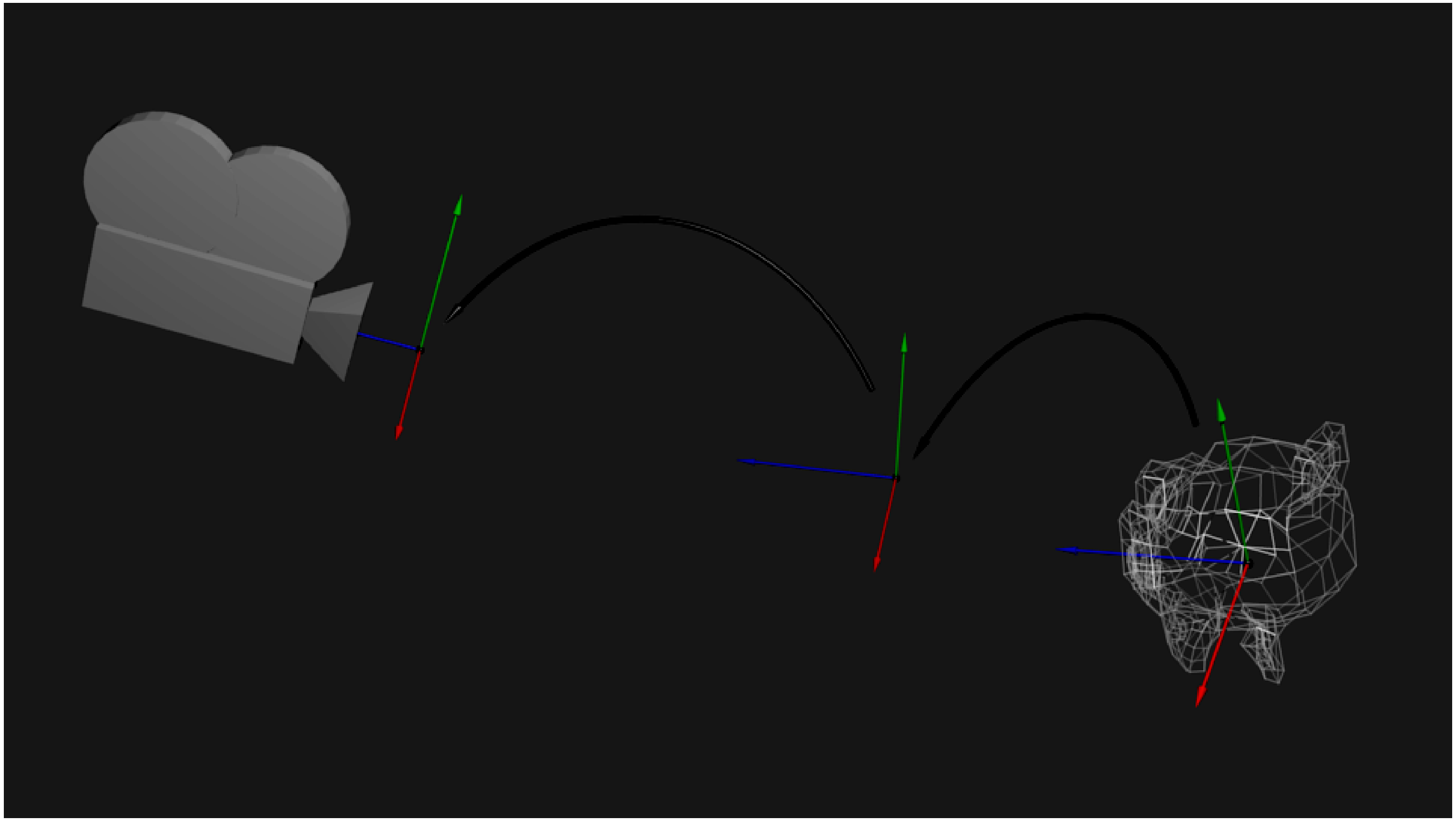
Au début, notre caméra est à l'origine dans le repère du monde. Afin de déplacer le monde, vous introduisez une nouvelle matrice, tout simplement. Exemple: déplacer la caméra t(-3,0,0) .



Les matrices de modèle, de vue et de projection

La matrice de vue

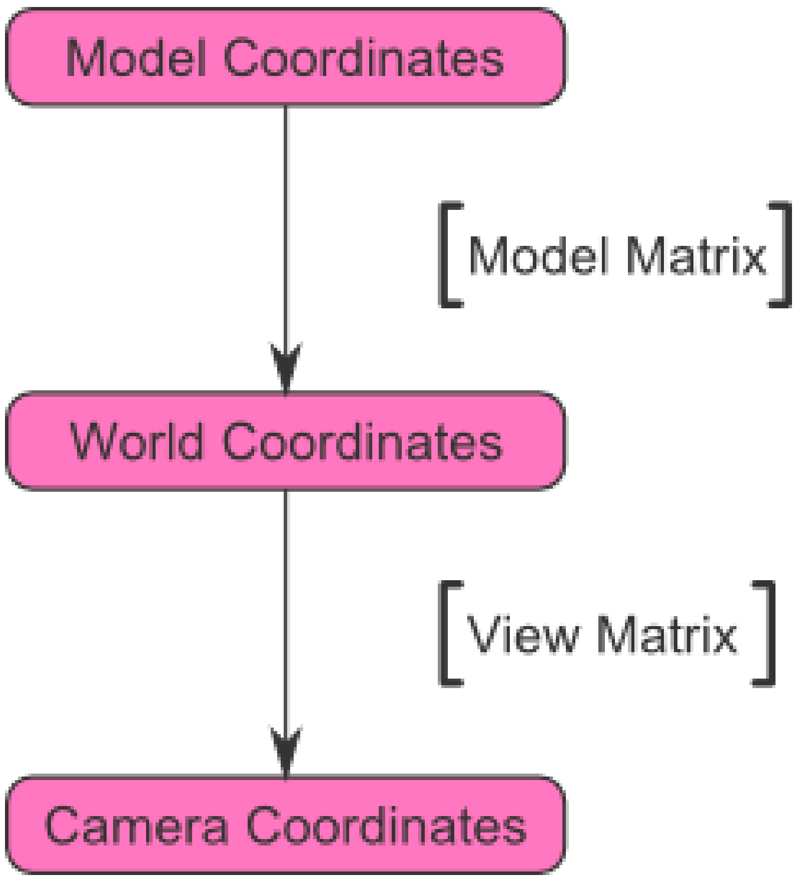
On s'est déplacé du repère du monde (tous les sommets sont définis par rapport au centre du monde) vers le repère de la caméra (tous les sommets sont définis par rapport à la caméra).



Les matrices de modèle, de vue et de projection

La matrice de vue

Voici le diagramme obligatoire :



Les matrices de modèle, de vue et de projection

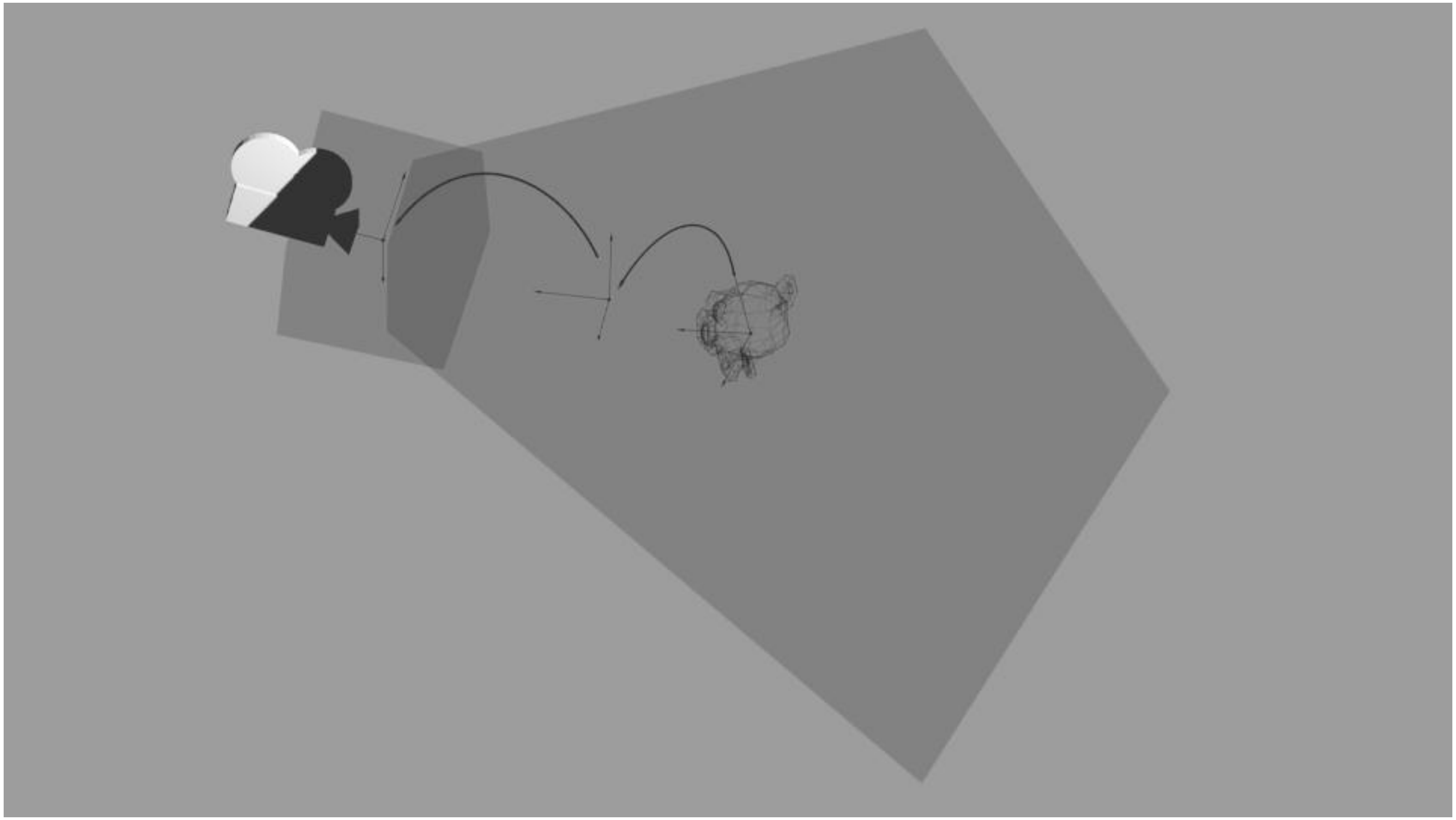
La matrice de projection

On est maintenant dans le repère de la caméra. Cela signifie qu'après toutes ces transformations, un sommet ayant les coordonnées x == 0 et y == 0 devrait être affiché au centre de l'écran. Mais on ne peut pas utiliser uniquement les coordonnées x et y pour déterminer où un objet devrait être placé à l'écran : sa distance par rapport à la caméra (z) est aussi importante ! Pour deux sommets avec les mêmes coordonnées x et y, le sommet avec la plus grande coordonnée z sera plus au centre de l'écran que l'autre.

Cela s'appelle une projection de perspective :

Les matrices de modèle, de vue et de projection

La matrice de projection



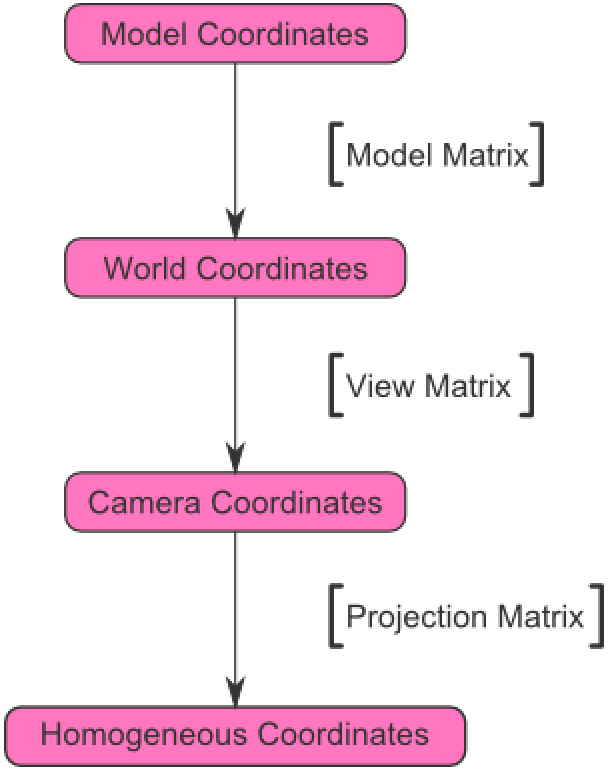
Les matrices de modèle, de vue et de projection

La matrice de projection

Et voici le diagramme final :

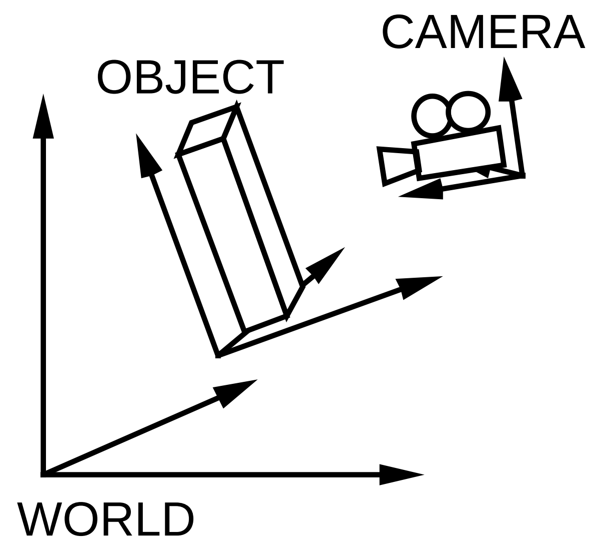
On s'est déplacé du repère de la caméra (tous les sommets sont définis par rapport à la caméra) vers le repère homogène (tous les sommets sont définis dans un petit cube). Tout ce qui est dans le cube apparaîtra à l'écran.

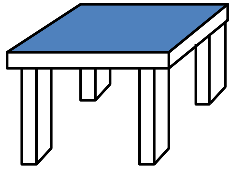
Et voici le diagramme final :



Transformations géométriques

- Repère du Monde
  - placement des objets
  - placement de la caméra (point de vue)
- Transformations
  - translation, rotation, mise à l’échelle
  - construire des scènes complexes en positionnant des objets simples (transformations successives)
  - transformer les coordonnées de l’objet en coordonnées du Monde
- Projection
  - pyramide de vue
  - projection en perspective
  - **Clipping:** consiste à ne pas calculer les objets extérieurs au <u>cône de vision</u> d'une scène,





Un pied générique est construit une seule fois, puis chaque pied de la table est une instanciation du premier, Il est ensuite déplacé à la bonne position.

Transformations géométriques

- **Modèle :** ensemble d'objets organisés pour représenter une scène devant être affichée. Chaque modèle est un ensemble de points (x,y,z).
- Représentation vectorielle des points
  - Points attachés aux primitives graphiques
  - Sommets, centres, données volumiques…
- Transformations sur ces données
  - **Transformation :** opération sur les sommets
    - Translation, rotation, changement d'échelle…
  - **Projections** :
    - Perspective, parallèle…
  - **Notation unifiée ?**

Les transformations ponctuelles
Les transformations de base

- Une fois que l'on a construit des objets graphiques, on désire généralement les manipuler, c'est -à dire changer leurs attributs.
- Les attributs d'un objet graphique sont:
  - la position
  - l'orientation
  - la taille
  - la forme
  - la couleur
  - la transparence
  - la texture

Transformations géométriques

- **Etapes d'une opération/transformation sur sommets (ex : rotation, puis translation) :**

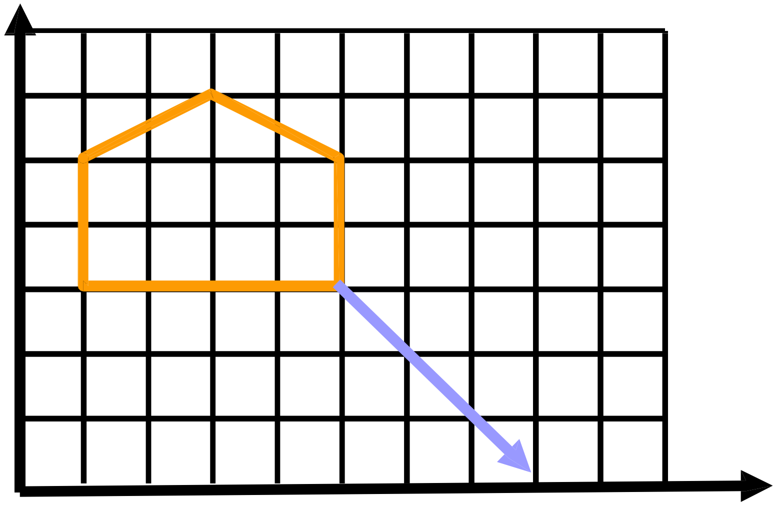
1. Soit un vecteur contenant les coordonnées d'un sommet. On mémorise tous les points du modèle de cette façon.
2. Générer une matrice 4x4 de **transformation.** Cette matrice mémorise les deux déplacements (2 matrices combinées, notion de **matrice active).**
3. Multiplier le vecteur de chaque sommet du modèle par la matrice de transformation. Si **v** est un sommet du modèle et **M** une matrice de transformation, **v' = M.v** est le sommet transformé.
4. Afficher la scène avec ces nouveaux sommets ... l'objet est déplacé !

En 2 dimensions

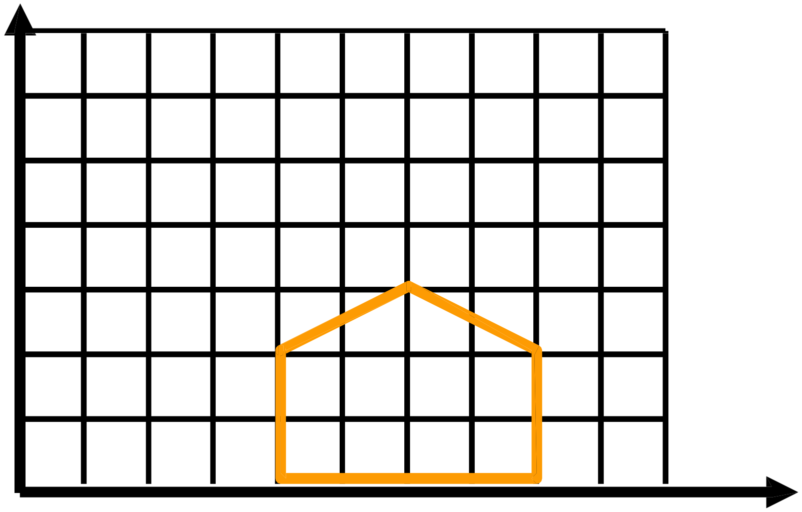
- On commence en 2D
  - Plus facile à représenter
- Chaque point est transformé:
  - $x' = f(x,y)$
  - $y' = g(x,y)$
- Comment représenter la transformation ?

Translations

- Modification simple :
  - $x' = x + t_x$
  - $y' = y + t_y$



Avant



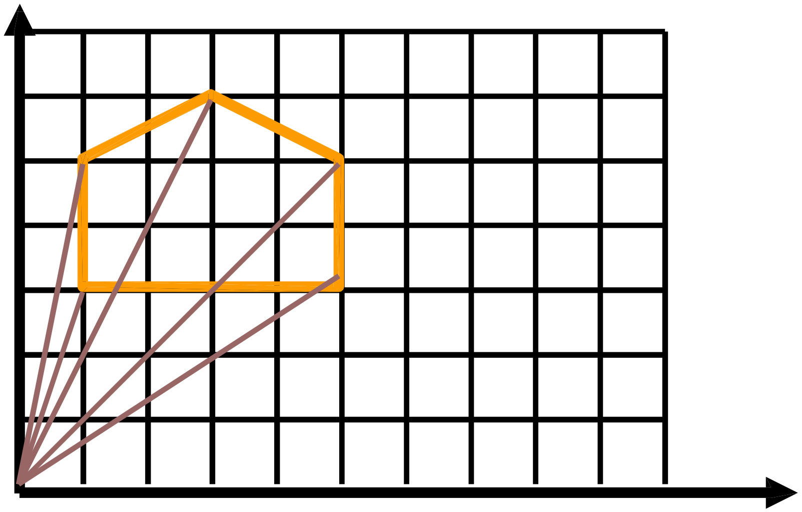
Après

Notation vectorielle

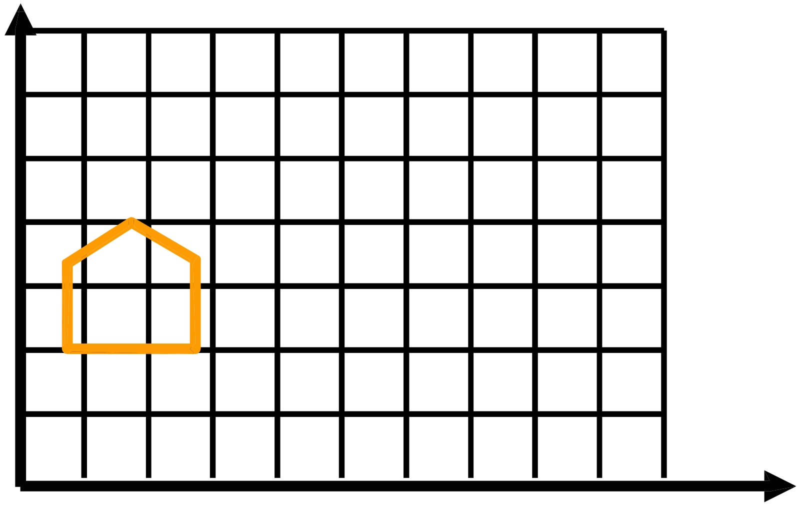
- On utilise des vecteurs pour la représentation $\begin{bmatrix} x \\ y \end{bmatrix}$
  - C’est plus simple
- Un point est un vecteur :
- Une translation est une somme vectorielle : P’ = P + T

Changement d’échelle

- Les coordonnées sont multipliées par le facteur de changement d’échelle :
  - $x' = s_x\, x$
  - $y' = s_y\, y$



Avant



Après

Notation matricielle

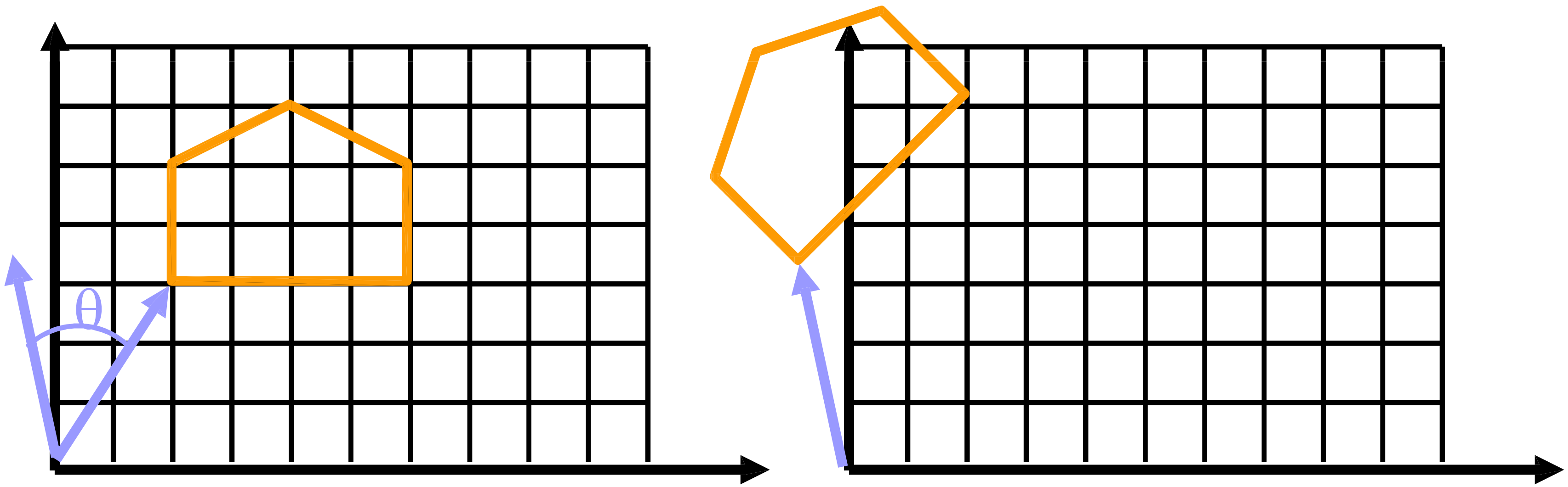
- C'est une multiplication matricielle :

P' = SP

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation

- Rotation en 2D :
  - x' = cosθ x - sinθ y
  - y' = sinθ x + cosθ y



Avant

Après

Notation matricielle

- Rotation = multiplication matricielle :

P'=RP

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Unification

- Notation simple, concise
- Mais pas vraiment unifiée
  - Addition ou bien multiplication
  - **Comment faire pour concaténer plusieurs transformations ?**
- **On veut une notation unique**
  - Qui permette de noter aussi les combinaisons de transformations
  - Comment faire ?

Coordonnées homogènes

Idée travailler dans un espace de dimension (n+1)
– appelé espace projectif
– représenter les transformations géométriques affines par des matrices

- Outil géométrique très puissant :
  - Utilisé partout en Infographie
  - aussi géométrie projective
- On ajoute une troisième coordonnée, W
- Un point 2D devient un vecteur à 3 coordonnées : $\begin{bmatrix} x \\ y \\ w \end{bmatrix}$

Passage des coordonnées usuelles aux coordonnées homogènes

(x, y, z) —> [x, y, z, 1]

[x, y, z, w] —> [x/w, y/w, z/w] , si w ≠ 0

Coordonnées homogènes

- Deux points sont égaux si et seulement si :
  - $x'/w' = x/w$ et $y'/w' = y/w$
- w=0: points « à l'infini » alors le vecteur (x, y, z,0) est une direction.
  - Très utile pour les projections, et pour certaines splines.
- w = 1, alors le vecteur (x, y, z, 1) est une position dans l'espace.

Et en 3 dimensions ?

- On introduit une quatrième coordonnée,

$$\mathbf{w} \longrightarrow \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

  - Deux vecteurs sont égaux si :
    x/w = x'/w', y/w = y'/w' et z/w=z'/w'
- Toutes les transformations sont des matrices 4x4

Translations en c. homogènes

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} x' = x + w t_x \\ y' = y + w t_y \\ w' = w \end{cases}$$

$$\begin{cases} \dfrac{x'}{w'} = \dfrac{x}{w} + t_x \\[2ex] \dfrac{y'}{w'} = \dfrac{y}{w} + t_y \end{cases}$$

Changement d'échelle

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} x' = s_x x \\ y' = s_y y \\ w' = w \end{cases}$$

$$\begin{cases} \dfrac{x'}{w'} = s_x \dfrac{x}{w} \\ \dfrac{y'}{w'} = s_y \dfrac{y}{w} \end{cases}$$

Rotation

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} x' = \cos\theta x - \sin\theta y \\ y' = \sin\theta x + \cos\theta y \\ w' = \qquad w \end{cases}$$

$$\begin{cases} \frac{x'}{w'} = \cos\theta \frac{x}{w} - \sin\theta \frac{y}{w} \\ \frac{y'}{w'} = \sin\theta \frac{x}{w} + \cos\theta \frac{y}{w} \end{cases}$$

Composition des transformations

- Il suffit de multiplier les matrices :
  - composition d'une rotation et d'une translation:

$$\mathbf{M = RT}$$

(en OpenGl il faut inverser **M = TR**)

- Toutes les transformations 2D peuvent être exprimées comme des matrices en coord. homogènes.
  - Notation très générale
- Rotation autour d'un point Q:
  - Translater **Q** à l'origine **(T_Q)**,
  - Rotation autour de l'origine **(R_Θ)**
  - Translater en retour vers Q **($- T_Q$)**.

$$\longrightarrow \quad P' = T_Q R_\Theta (-T_Q) P$$

Translations en 3D

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

$$\begin{cases} x' = x + wt_x \\ y' = y + wt_y \\ z' = z + wt_z \\ w' = w \end{cases}$$

Changement d'échelle en 3D

$$S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{cases} x' = s_x x \\ y' = s_y y \\ z' = s_z z \\ w' = w \end{cases}$$

Rotations en 3D

- Rotation : un axe et un angle
- La matrice dépend de l'axe et de l'angle
- Expression directe possible, en partant de l'axe et de l'angle, et quelques produits vectoriels
  - Passage par les quaternions
- Fait par la librairie graphique :
  - glRotatef(angle, x, y, z)

Rotation autour de l'axe x

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

L'axe x n'est pas modifié

Une rotation de $\pi/2$ change
y en *z*, et *z* en -*y*

$$R_x(\tfrac{\pi}{2}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation autour de l'axe y

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

L'axe y n'est pas modifié

Une rotation de $\pi/2$ change *z* en *x*, et *x* en -*z*

$$R_y(\tfrac{\pi}{2}) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation autour de l'axe z

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

L'axe z n'est pas modifié

Une rotation de π/2 change
x en *y*, et *y* en -*x*

$$R_z(\tfrac{\pi}{2}) = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Toutes les transformations 3D

- Toute transformation 3D s'exprime comme combinaison de translations, rotations, changement d'échelle
  - Et donc comme une matrice en coordonnées homogènes
- Fournies par la librairie graphique :
  - glTranslatef(x, y, z);
  - glRotatef(angle, x, y, z);
  - glScalef(x, y, z);

Transformations 3D (suite)

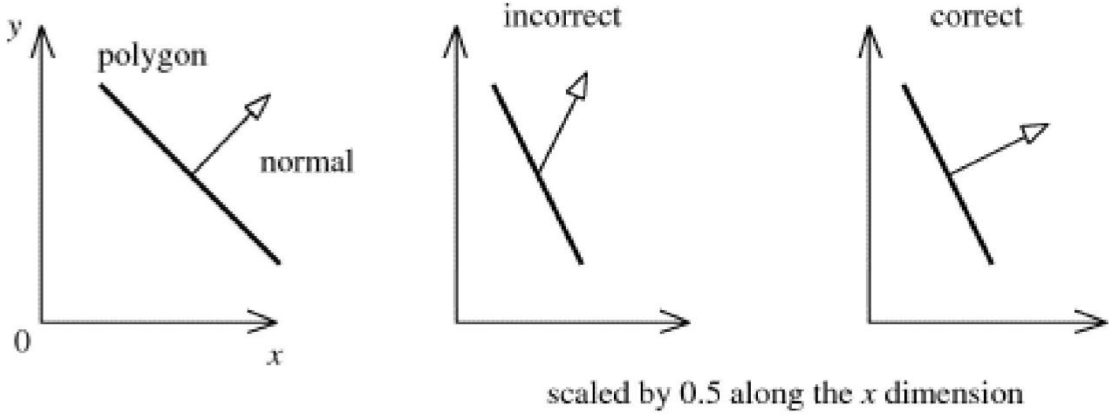
- On peut faire ses transformations soi-même :
  - glLoadIdentity();
  - glLoadMatrixf(pm);
  - glMultMatrixf(pm);
- Pile de transformations :
  - glPushMatrix();
  - glPopMatrix();

Exemple

```
drawHighLevelObject(parameters) {
    glPushMatrix()
    glRotate(…)
    glTranslate(…)
    glScale(…)
    drawSimpleShape()
    gllPopMatrix()
}
drawModel() { glPushMatrix()
    drawHighLevelObject1(…)
    glTranslate(…)
    drawHighLevelObject2(…)
    [etc…]
    gllPopMatrix()
}
```

Transformation des normales

- Vecteur normal (à la surface)
- Pas vraiment un vecteur
  - Définit une relation sur les vecteurs
  - Une forme linéaire, un co-vecteur



©R. Barzel, 2002

- Transformation en utilisant la transposée de l'inverse de M

M: est la partie linéaire de la matrice de vue (matrice 3x3 supérieure gauche).

$$n' = \left(M^T\right)^{-1} \cdot n$$

Rastérisation

Rastérisation

– Pour chaque primitive P_i, trouver les rayons intersectant P_i

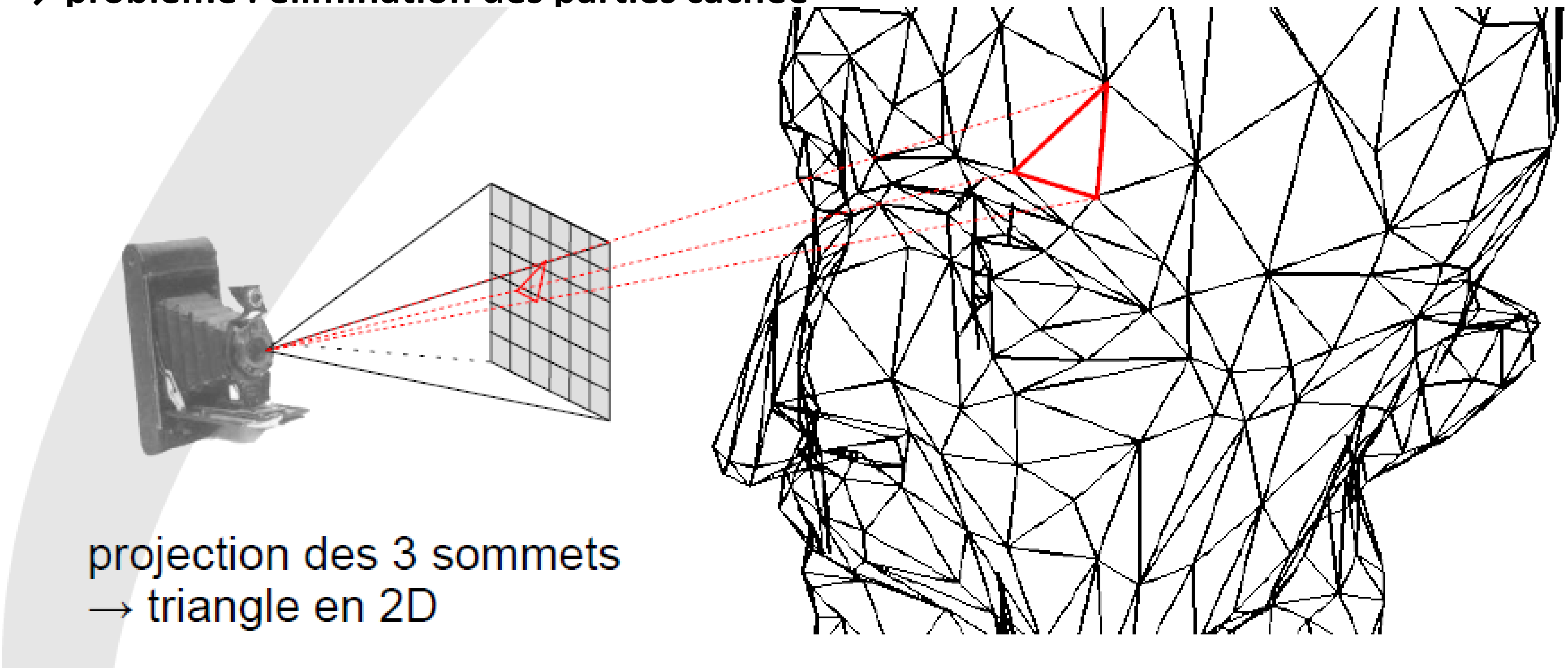
– Rendu en deux étapes

• projection des primitives sur l'écran (*forward projection)*

• discrétisation (conversion des primitives 2D en pixels)

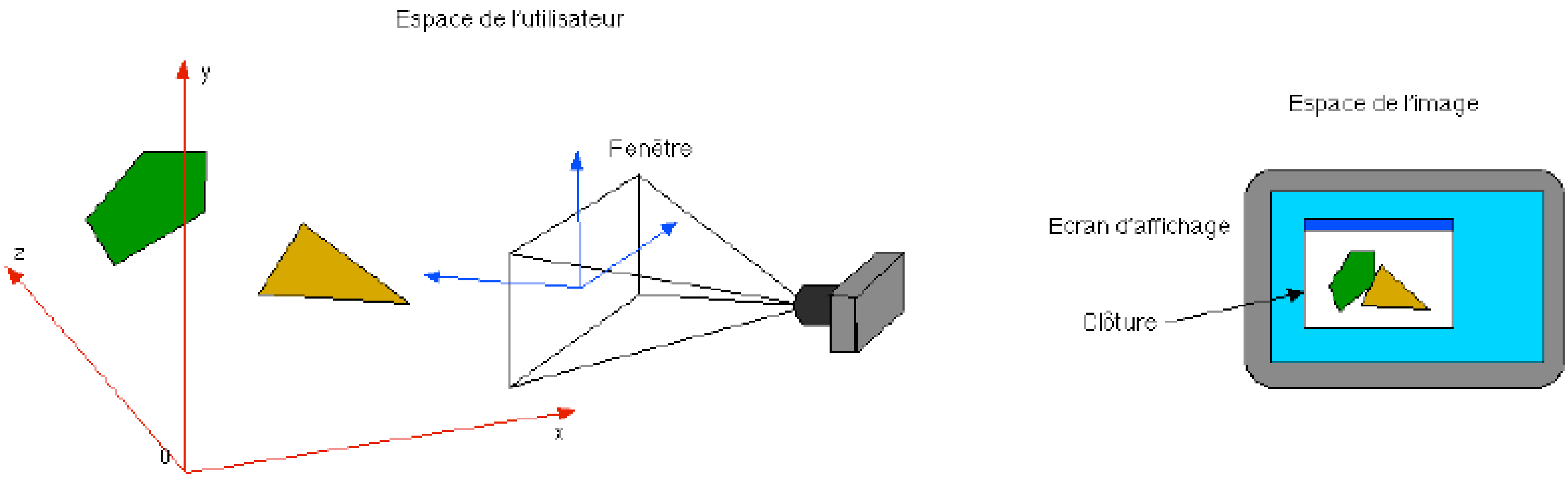
– scène = ensemble de primitives ≪ rasterisables ≫

→ problème : élimination des parties cachée



Passage de la scène à l’écran

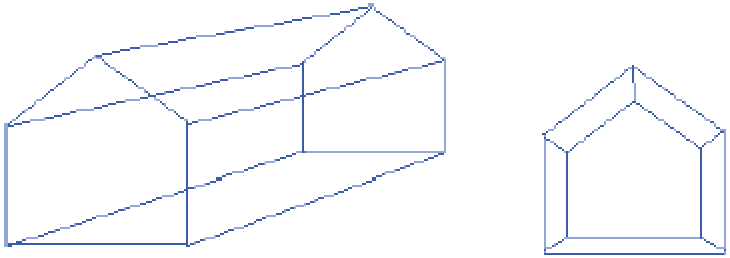
L’espace de l'utilisateur et l'espace de l'image



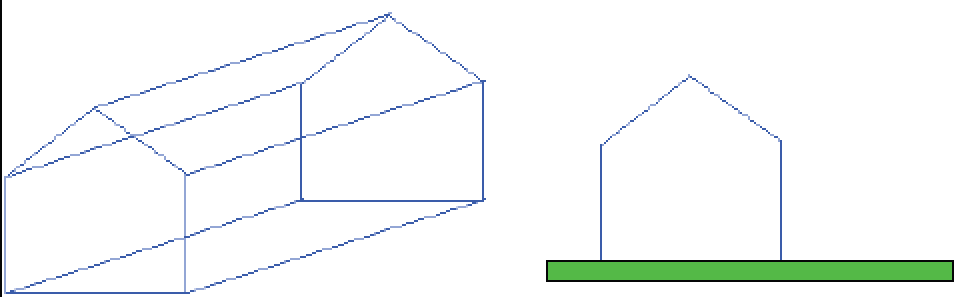
Passage de la scène à l’écran

Passage de l'espace 3D à l'espace 2D

- L'opération mathématique est une projection.
- On utilise couramment deux types de projection :
  - projection avec perspective (plus réaliste)



  - projection orthogonale (plus commode parfois lors de la création et dans des domaines comme la CAO)

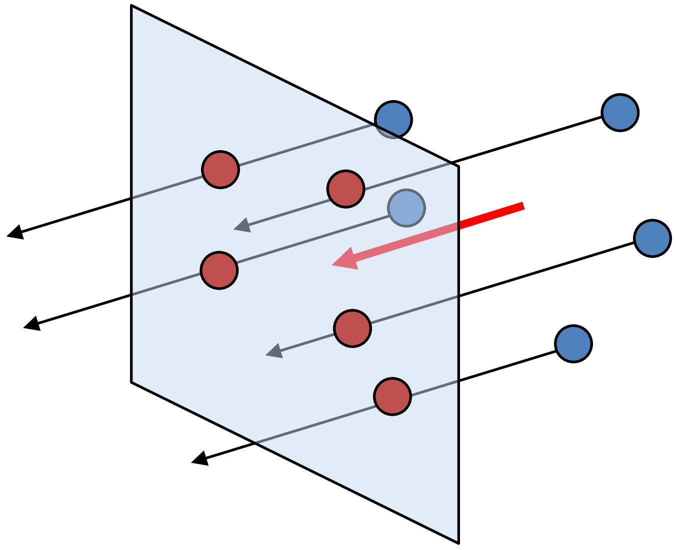


Projections

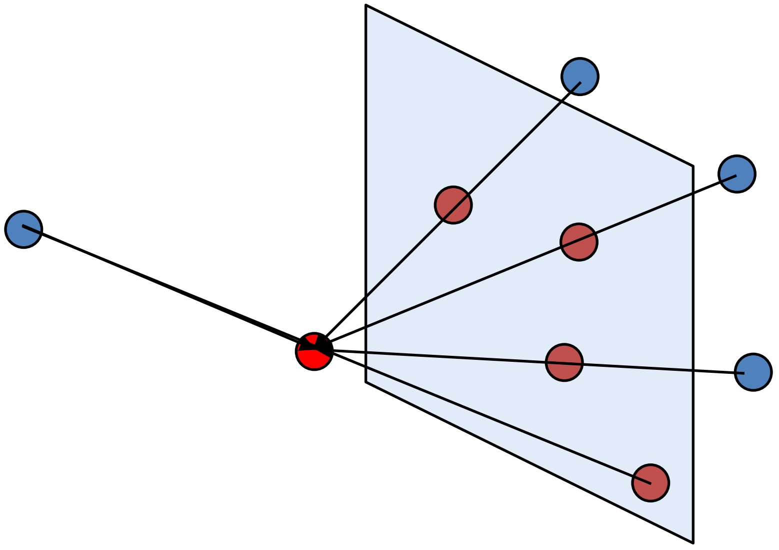
- Projection réduit le domaine $P_n \in \Re^n \rightarrow \Re^m$ où $m < n$
  - typiquement en infographie, n=3 et m=2
- Un projecteur est un segment reliant P_n à un centre de projection
- L'intersection d'un projecteur avec la surface de projection correspond à P_m
- Lorsque cette surface est planaire, on parle de projection planaire

Projection planaire

- On divise les projections planaires en
  - projection parallèle
  - projection perspective



Parallèle



Perspective

Projection parallèle

- Centre de projection est à l'infini
- Direction de projection

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} - \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \\ z_2 - z_1 \\ 0 \end{bmatrix}$$

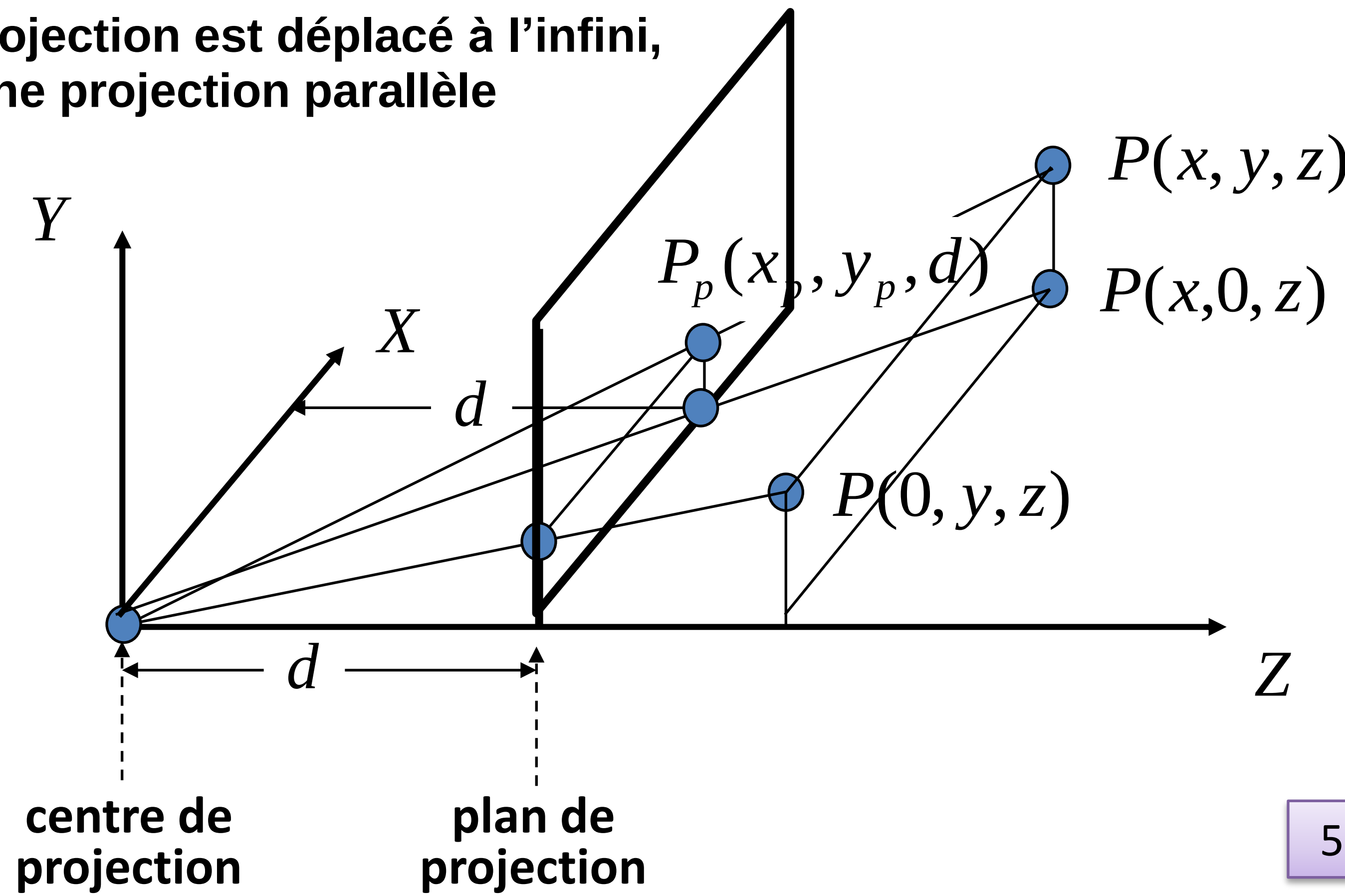
direction ou point à l'infini

- Projecteurs sont parallèles entre eux
- Lignes parallèles en 3D demeurent parallèles après projection

Projection perspective

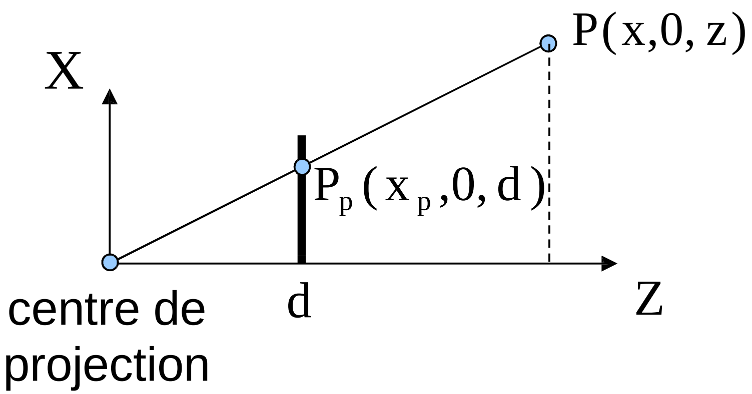
- Centre de projection est à une distance finie
- Taille d'un objet augmente lorsque la distance au centre de projection diminue
- Lignes parallèles en 3D ne sont plus parallèles après projection

- Si le centre de projection est déplacé à l'infini,
  on obtient une projection parallèle

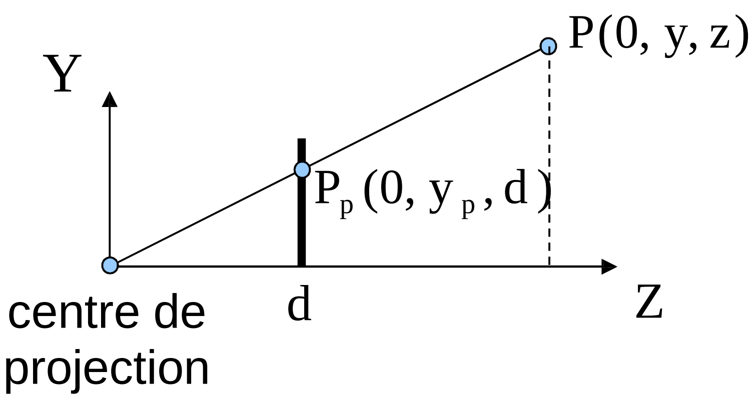


Projection perspective

- Théorème de Thales : Triangles semblables



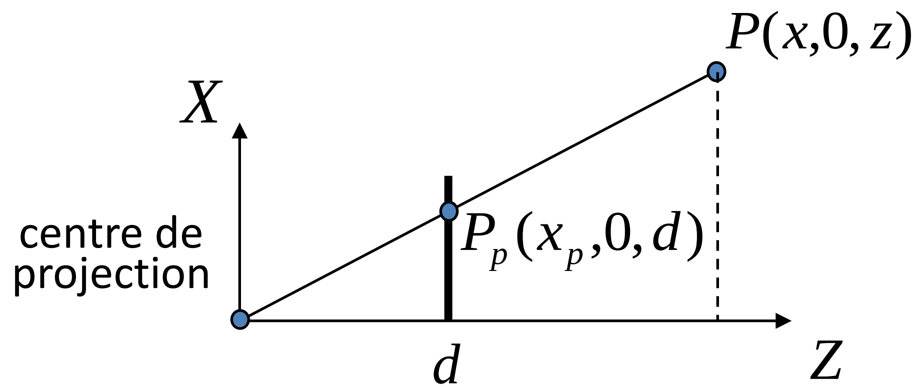
$$\frac{x_p}{d} = \frac{x}{z} \quad : \quad x_p = \frac{x}{z/d}$$

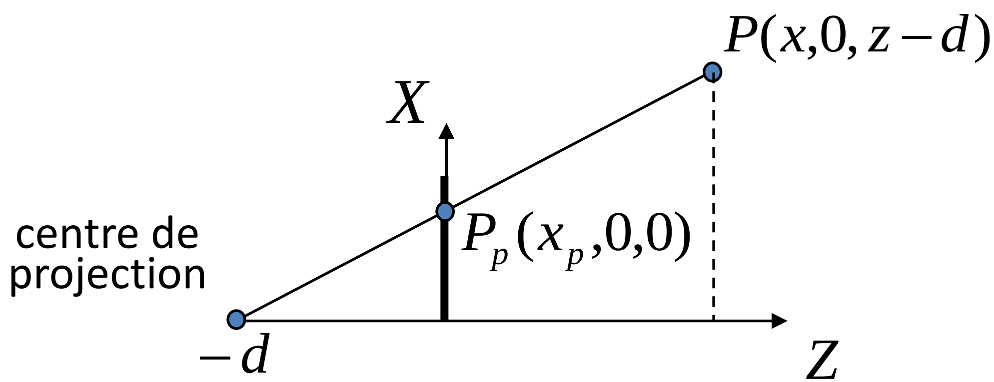


$$\frac{y_p}{d} = \frac{y}{z} \quad : \quad y_p = \frac{y}{z/d}$$

Projection perspective

Plan de projection à *z*=0





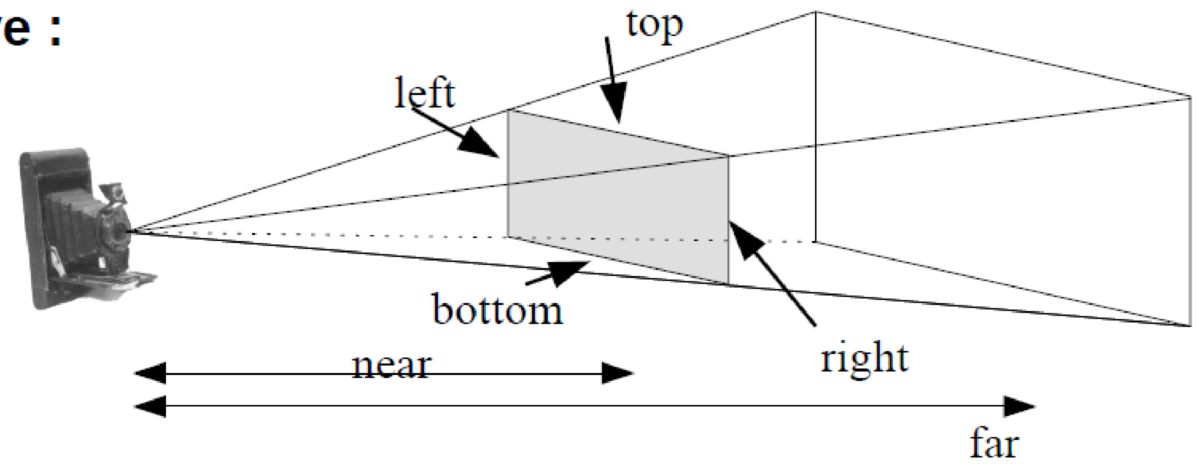
Projection perspective

- Projection sur le plan z=0, avec le centre de projection placé à z=-d:

$$\begin{bmatrix} x_p \\ y_p \\ d \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1/d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

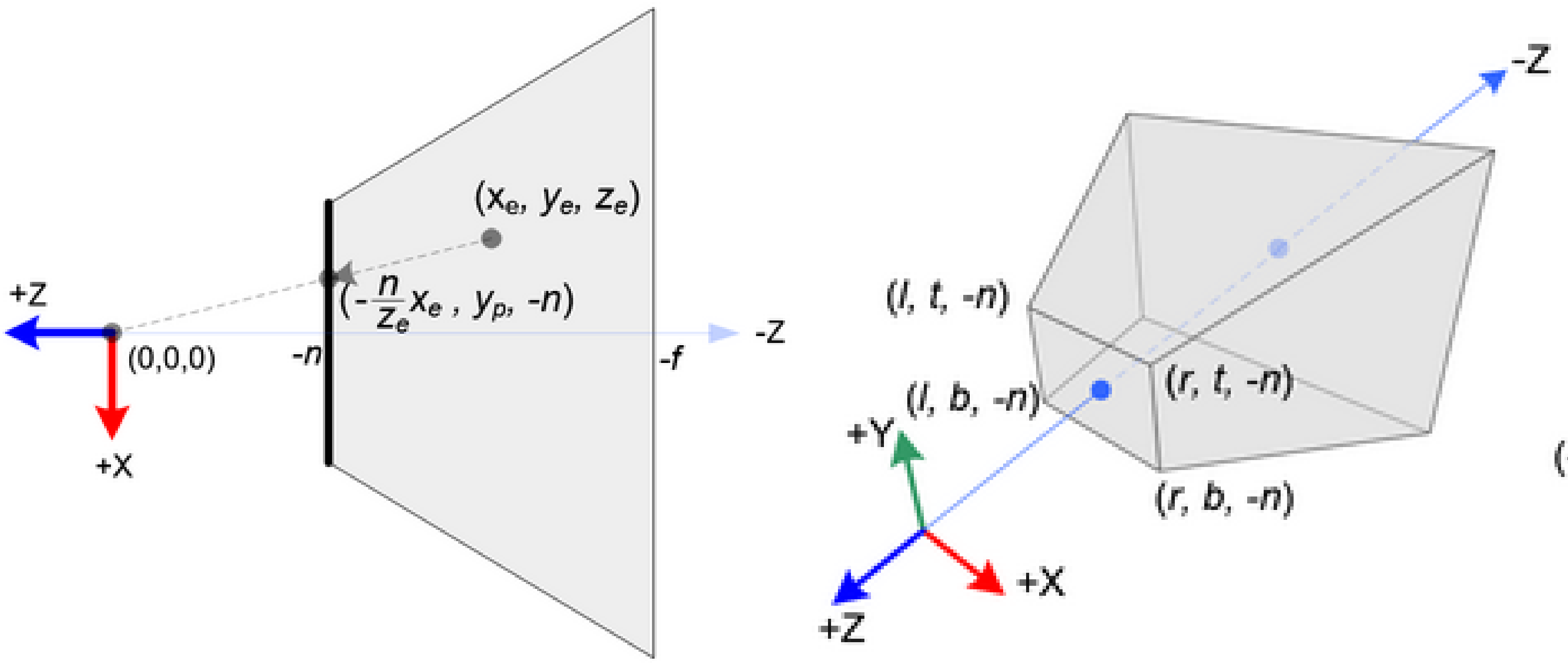
Projection perspective

- **Projection perspective :**



$$\begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

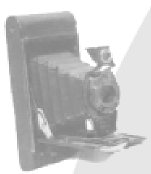
Projection perspective

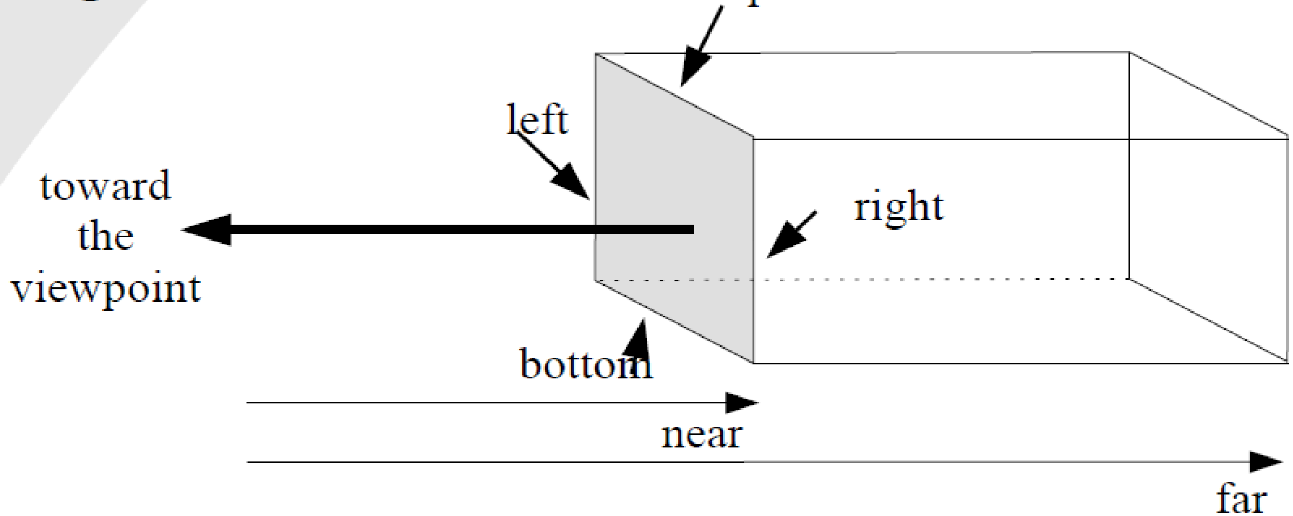


Pour plus de détail consulté le site:
http://www.songho.ca/opengl/gl_projectionmatrix_mathml.html

Projection perspective

- **Projection orthogonale**

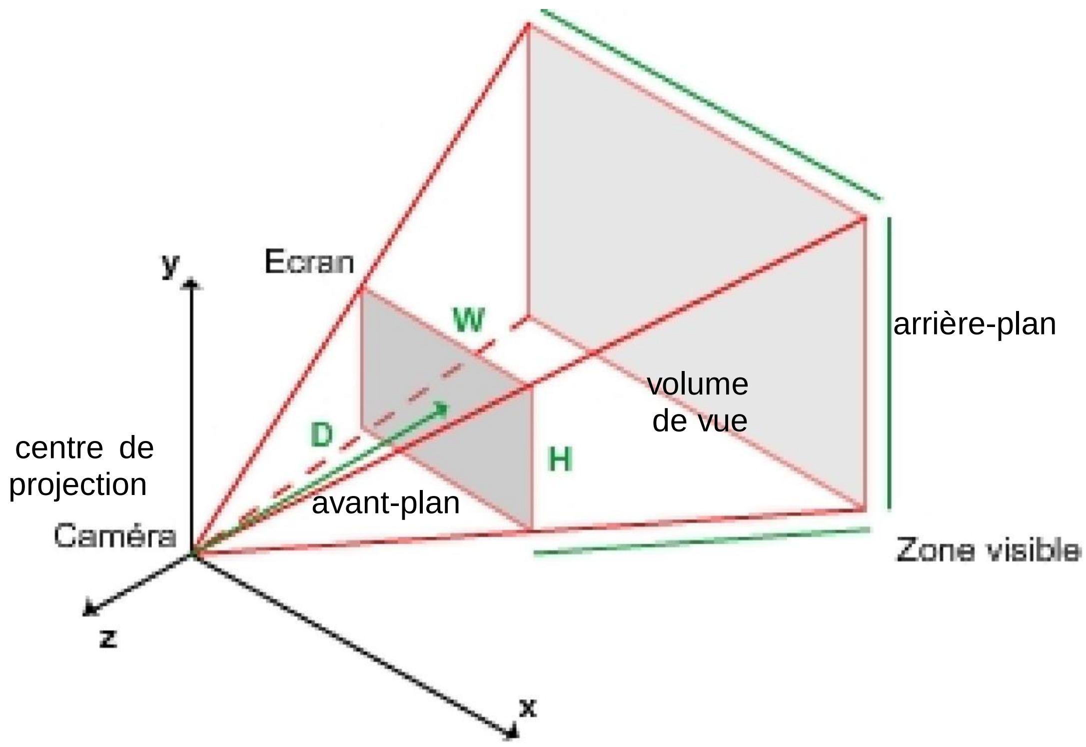




$$\begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & -\frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Pyramide de vue

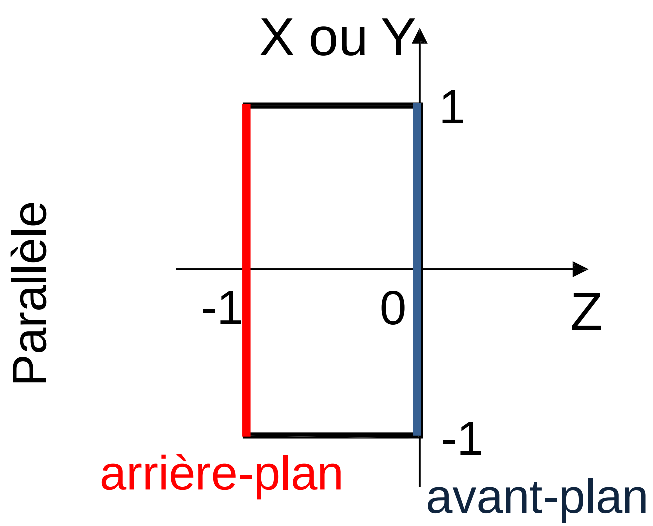
- Clipping avec les six plans définissant le volume de vue
- Projection des survivants au clipping sur la fenêtre
- Transformations en coordonnées d'affichage

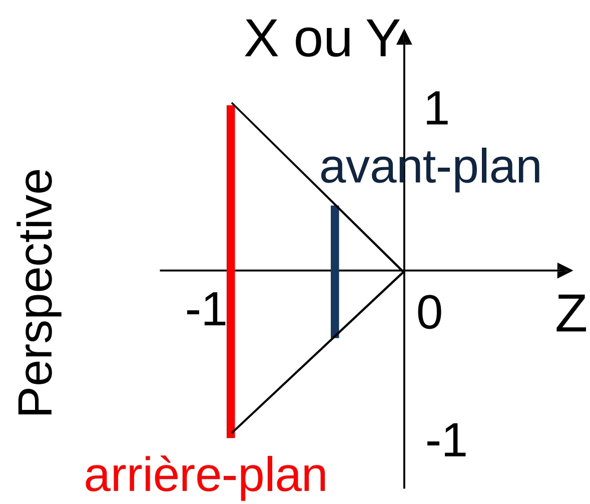


Forme Canonique de la pyramide de vue

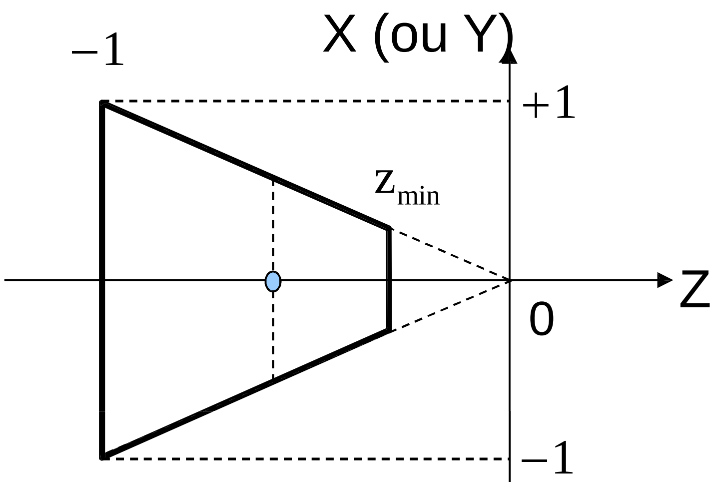
Volume de vue canonique : les faces proche et lointaine du parallélépipède sont parallèles à *xOy*. La caméra est en O.

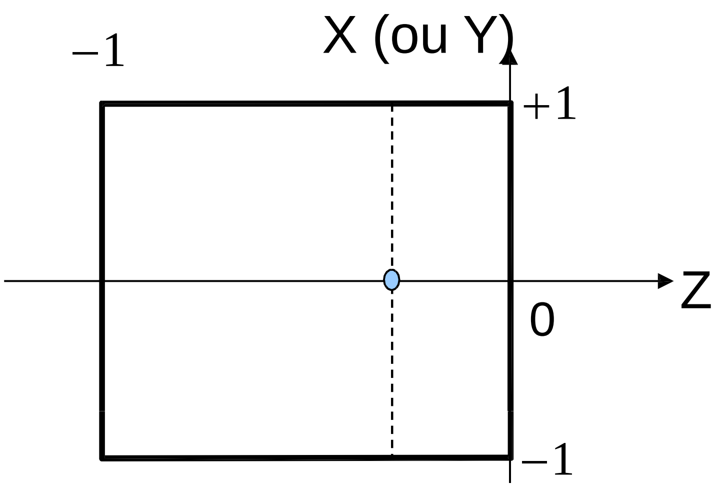
- Le clipping avec des plans arbitraires peut être coûteux, alors on transforme la pyramide de vue dans une forme canonique
- + Clipping sera simplifié
- - Transforme des points qui pourraient être clippés





Forme Canonique de la pyramide de vue





$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{1+z_{min}} & \frac{-z_{min}}{1+z_{min}} \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad \text{pour } z_{min} \neq -1$$

Cohen-Sutherland en 3D (6 bits)

$$x < -1 \quad ; \quad x > 1$$
$$y < -1 \quad ; \quad y > 1$$
$$z < -1 \quad ; \quad z > 0$$