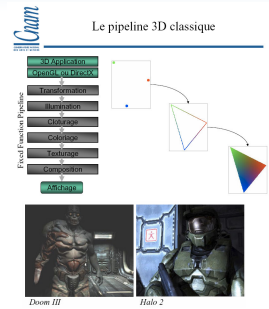
Synthèse d'Images

Elmar Eisemann
Elmar.Eisemann@inrialpes.fr

Basé sur les cours de
Frédo Durand, Barbara Cuttler, E. Boyer, H. Briceno, N. Holzschuch, Alex Meyer

Alternative au lancer de rayons

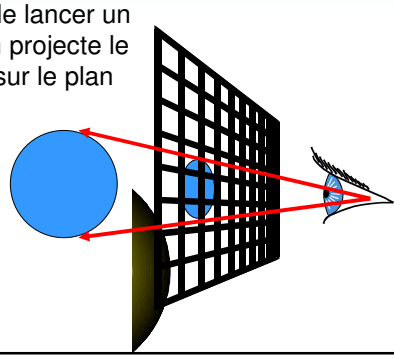
- Le pipeline graphique !
- Plus mécanique et automatisable
- Motivation pour apprendre les transformations!



http://deptinfo.cnam.fr/Enseignement/CycleSpecialisation/CAM/sicam6.pdf

Alternative au lancer de rayons

- Au lieu de lancer un rayon on projecte le triangle sur le plan image



Pourquoi? Hardware optimisé:

Ray tracing:
CPU

Graphics Hardware:
Capable de faire certains calculs très vite
Par exemple:
Gouraud shading

Pourquoi? Complexité:

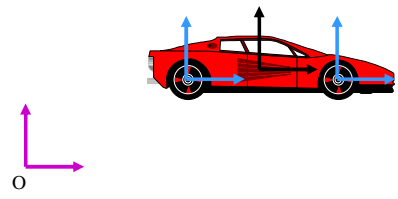
Ray tracing:
Complexité: m*n
Avec accélération:
Environ m*log(n)
Où
m= nombre de pixels
n= nombre de triangles

Graphics Pipeline:
Complexité: k+n
Où
k= nombre de pixels déssiné
n= nombre de triangles

Pourquoi? Repère locales

Ray tracing:
Changer le rayon

Graphics Pipeline:
Automatique

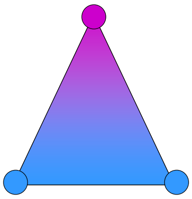


http://deptinfo.cnam.fr/Enseignement/CycleSpecialisation/CAM/sicam6.pdf

Pourquoi?
Interpolation d'information

Ray tracing:
À chaque point recalcué

Graphics Pipeline:
Interpolation directe

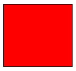


Le pipeline graphique

- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display

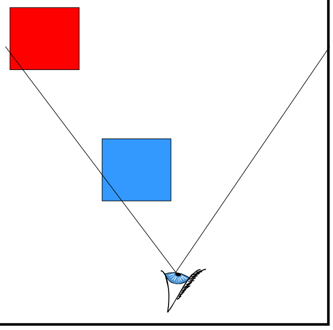
Le pipeline graphique

- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display



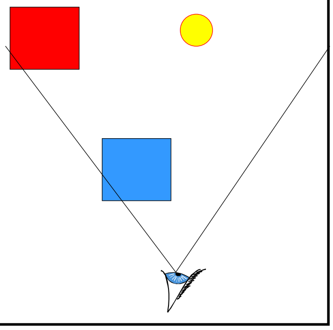
Le pipeline graphique

- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display



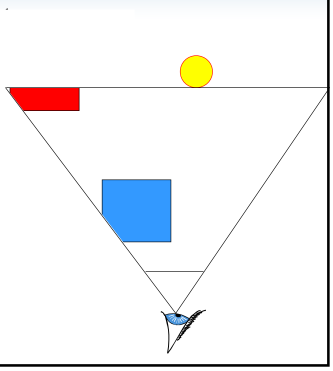
Le pipeline graphique

- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display



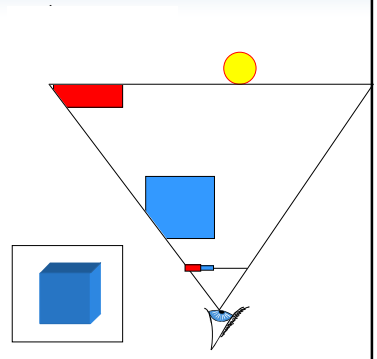
Le pipeline graphique

- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display



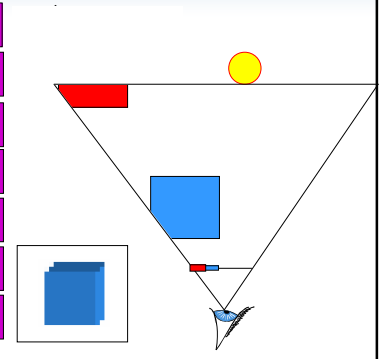
Le pipeline graphique

- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display



Le pipeline graphique

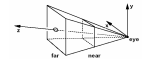
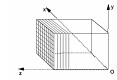
- Object Transformation
- Viewing Transformation (Perspective / Orthographic)
- Illumination (Shading)
- Clipping
- Projection (to Screen Space)
- Scan Conversion (Rasterization)
- Display



Le pipeline graphique

- Rapide: Toujours les mêmes operations
- Pas de ray-tracing!
  (Changement des coordonnées)

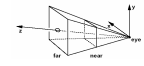
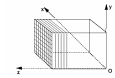
Systèmes Coordonnées

Object Transformation	World space (repère monde)	Object space (repère objet)
Viewing Transformation (Perspective / Orthographic)		Eye/Camera Space (repère camera)
Clipping		Clip Space
Projection (to Screen Space)		Screen Space (repère image)

Plan

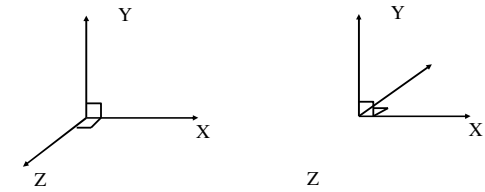
- Transformations géométriques
- Projections, caméra, perspective
- Élimination des parties cachées

Système de coordonnées

Object Transformation	World space (repère monde)	Object space (repère objet)
Viewing Transformation (Perspective / Orthographic)		Eye Space / Camera Space (repère camera)
Illumination (Shading)		
Clipping		Clip Space (NDC)
Projection (to Screen Space)		
Scan Conversion (Rasterization)		Screen Space (repère image)
Visibility / Display		

Préliminaires

Co-ordinate Systems



Right-Handed System
(Z comes out of the screen)

Left-Handed System
(Z goes in to the screen)

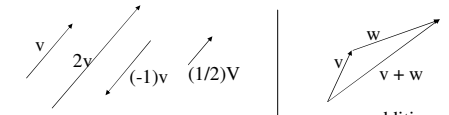
Points, P (x, y, z)

- Position relative à l'origine du système de coordonnées

Vecteurs, V (x, y, z)

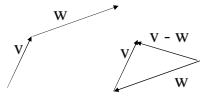
- *direction* en 3D
- Points != Vecteurs
  - *Point – Point = Vecteur*
  - *Vector+Vecteur = Vecteur*
  - *Point + Vecteur = Point*
  - *Point + Point = ?*

Vecteurs, V (x, y, z)

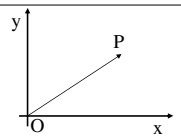


multiplication scalaire (parallel)

addition
v + w



difference
v - w = v + (-w)



Vector $\overrightarrow{OP}$

Vectors V

- Norme d'un vecteur V (x, y, z)
  - $|\underline{V}| = \sqrt{x^2 + y^2 + z^2}$
- A unit vector

$$\hat{V} = \frac{\underline{V}}{|\underline{V}|}$$

Dot Product

- $a \cdot b = |a|\,|b| \cos\theta$

  $\therefore \cos\theta = a \cdot b / |a|\,|b|$
- $a \cdot b = x_a \cdot x_b + y_a \cdot y_b + z_a \cdot z_b$

Cross Product

- Résultat est un vecteur ortho
- Calculé en utilisant les déterminants

$$\begin{vmatrix} i & j & k \\ x_v & y_v & z_v \\ x_u & y_u & z_u \end{vmatrix} \quad \begin{vmatrix} i & j & k \\ x_v & y_v & z_v \\ x_u & y_u & z_u \end{vmatrix} \quad \begin{vmatrix} i & j & k \\ x_v & y_v & z_v \\ x_u & y_u & z_u \end{vmatrix}$$

  – $i(y_v z_u - z_v y_u)$, $-j(x_v z_u - z_v x_u)$, $k(x_v y_u - y_v x_u)$
- $| a \times b | = |a||b|\sin\theta$
- $a \times a = 0$

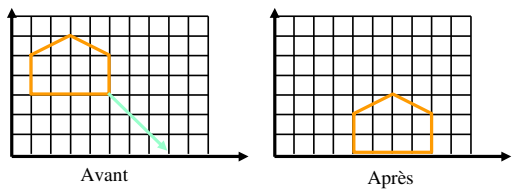
Qu'est-ce qu'on veut représenter?

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

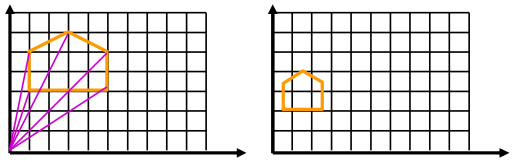


Notation vectorielle

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

Changement d'échelle

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



Notation matricielle

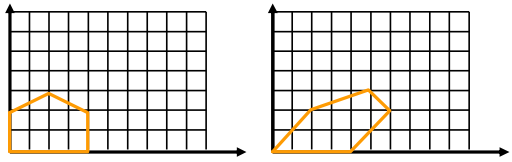
- C'est une multiplication matricielle :

P' = **S**P

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

Changement d'affinité

- Les coordonnées sont multipliées par le facteur de changement d'échelle dans une autre coordonnée:
  - $x' = x + sh_y\, y$
  - $y' = y + sh_x\, x$



Notation matricielle

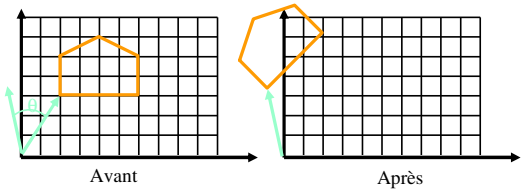
- C'est une multiplication matricielle :

P' = **SH**P

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation

- Rotation en 2D :
  - $x' = \cos\theta\, x - \sin\theta\, y$
  - $y' = \sin\theta\, x + \cos\theta\, y$



Notation matricielle

- Rotation = multiplication matricielle :

P'=**R**P

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

Unification?

P' = P + T

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

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

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

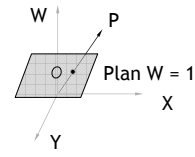
Coordonnées homogènes

- Outil géométrique très puissant :
  - Utilisé partout (Image, Vision, Robotique)
- On ajoute une troisième coordonnée, *w*
- Un point 2D devient un vecteur à 3 coordonnées :

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Coordonnées homogènes

- Deux points (x,y,w) et (x',y',w') sont égaux si et seulement si :
  - *x'/w' = x/w* et *y'/w'= y/w*
  - *Mieux: Si exist a!=0 tq: a(x,y,w)=(x',y',w')*
- *w=0:* points « à l'infini »



Translations en c. homogènes

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

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

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

Changement d'échelle

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

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

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

Rotation

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

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

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

Composition des transformations

- Les transf. rigide sont linéaires! (et encore d'autres)
- Donc concatenation en multipliant

M = (Rot)(Trans)

Rotation autour d'un point Q

- Rotation autour d'un point Q:
  - Translater Q à l'origine ($\mathbf{T}_Q$),
  - Rotation autour de l'origine ($\mathbf{R}_\Theta$)
  - Translater en retour vers Q ($\mathbf{T}_{-Q}$).

$$P' = (\mathbf{T}_{-Q})\mathbf{R}_\Theta \mathbf{T}_Q P$$

Et en 3 dimensions ?

- C'est pareil
- On introduit une quatrième coordonnée, *w*
  - Deux vecteurs sont égaux si : *x/w = x'/w'*, *y/w = y'/w'* et *z/w=z'/w'*

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

- Toutes les transformations sont des matrices 4x4

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 = axe et angle
- Les rotations autour d'un axe de coordonnées ont une expression simple
  - Les autres rotations s'expriment comme combinaison de ces rotations simples

Rotation autour *x*-axis

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

Axe x ne Change pas

Verification rapide: une rotation de π/2
Doit changer *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 *y*-axis

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

y-axis ne Change pas

Verification rapide: une rotation de π/2
Doit changer *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 about *z*-axis

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

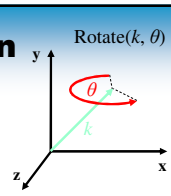
z-axis ne Change pas

Verification rapide: une rotation de π/2
Doit changer *x* en *y*, and *y* en *-x*

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

Rotation

- Autour (k_x, k_y, k_z)
  (Formule de Rodrigues)



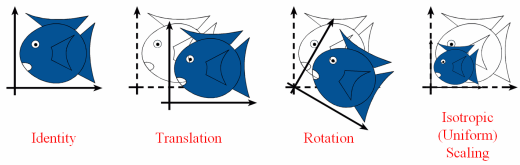
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} k_xk_x(1\text{-}c)+c & k_zk_x(1\text{-}c)-k_zs & k_xk_z(1\text{-}c)+k_ys & 0 \\ k_yk_x(1\text{-}c)+k_zs & k_zk_x(1\text{-}c)+c & k_yk_z(1\text{-}c)-k_xs & 0 \\ k_zk_x(1\text{-}c)-k_ys & k_zk_x(1\text{-}c)-k_xs & k_zk_z(1\text{-}c)+c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

où $c = \cos\theta$ & $s = \sin\theta$

Toutes les transformations 3D

- Transformation rigide 3D= combinaison de translations, rotations.
- En plus on peut changer l'échelle
  - Donc des matrices en coordonnées homogènes

Transformations simples



- Peut-être combinés
- Inversible?

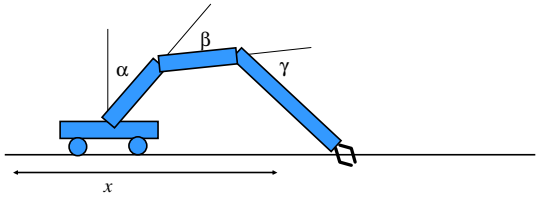
Oui, sauf scale = 0

Attention !

- La multiplication de matrice n'est pas commutative
- L'ordre des transformations est important
  - Rotation puis translation aura un effet très différent de translation puis rotation
  - Une source de bugs **très** courante

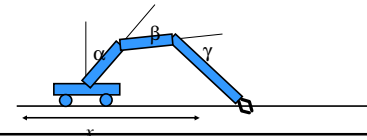
Définition d'objets complexes

Notre problème :



Définir un objet complexe

- L'objet est défini comme une combinaison d'objets plus petites
  - Exemples : main, bras, corps…
- On veut facilement le manipuler:
  - Maintenir connectivité:
    - si je bouge le bras, la main suit
  - Utiliser les paramètres naturels : x,α,β,γ



Comment faire ?

- Coordonnées relatives
  - roue par rapport à la voiture
  - boulons par rapport à la roue
- Presque personne n'utilise des coordonnées absolues dans la vie

Coordonnées relatives

- Utiliser les coordonnées relatives :
  - La position de l'avant-bras est donné en fonction du bras

Coordonnées relatives

- Comment revenir aux coordonnées absolues?
  - Garder une matrice courrante qui transforme relative en absolue:
    - Translation sur la position du bras
    - Dessiner le bras
    - Concatentation avec translation sur la position de l'avant-bras par rapport au bras
    - Concatenation avec Rotation
    - Dessiner l'avant-bras

Coordonnées relatives

- Problème: Je veux revenir aux coordonnées du corps pour dessiner l'autre bras

Coordonnées relatives

- Garder l'information sur les transformations successives
  - **M** = model vers vue
  - **M'=MT** (translation par x)
  - Dessin voiture
  - **M''=M'T1** (translation au roue 1)
  - Dessiner roue à (0,0)
  - return to **M'**
  - **M'''=M'T2** (translation au roue 2)
  - Dessiner roue à (0,0)

Idées???

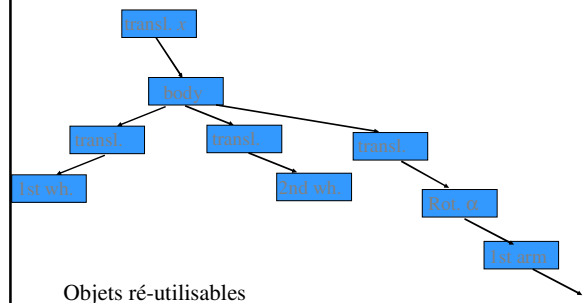
Pile de transformations

- OpenGL:
  - `popmatrix(), pushmatrix()`
- SPHIGS:
  - `openStructure(), closeStructure()`
- Postscript:
  - `gsave, grestore`

Définition hiérarchique

- Définition hiérarchique des objets
- Dessiner l'objet = traversée de la hiérarchie

Définition hiérarchique

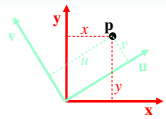


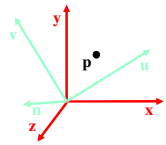
Objets ré-utilisables

- Une seule procédure pour toutes les roue

Changer le repère

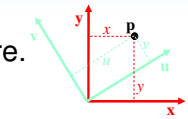
- Comment trouver la transformation d'un repère dans l'autre?



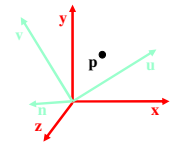


Changer le repère

- Translater une origine dans l'autre.

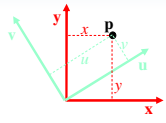


- Et l'orientation?

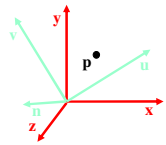


Change of Orthonormal Basis

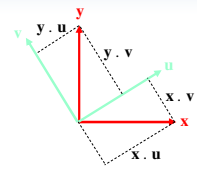
- Given:
  coordinate frames **xyz** and **uvn**
  point **p** = (*x*,*y*,*z*)

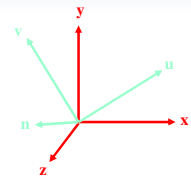


- Find:
  p = (*u*,*v*,*n*)



Change of Orthonormal Basis





$$\begin{aligned}
\mathbf{x} &= (\mathbf{x}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{x}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{x}\cdot\mathbf{n})\,\mathbf{n} \\
\mathbf{y} &= (\mathbf{y}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{y}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{y}\cdot\mathbf{n})\,\mathbf{n} \\
\mathbf{z} &= (\mathbf{z}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{z}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{z}\cdot\mathbf{n})\,\mathbf{n}
\end{aligned}$$

Change of Orthonormal Basis

$$\begin{aligned}
\mathbf{x} &= (\mathbf{x}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{x}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{x}\cdot\mathbf{n})\,\mathbf{n} \\
\mathbf{y} &= (\mathbf{y}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{y}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{y}\cdot\mathbf{n})\,\mathbf{n} \\
\mathbf{z} &= (\mathbf{z}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{z}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{z}\cdot\mathbf{n})\,\mathbf{n}
\end{aligned}$$

Substitute into equation for *p*:

$$\mathbf{p} = (x,y,z) = x\,\mathbf{x} + y\,\mathbf{y} + z\,\mathbf{z}$$

$$\begin{aligned}
\mathbf{p} = {} & x\,[\,(\mathbf{x}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{x}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{x}\cdot\mathbf{n})\,\mathbf{n}\,] + \\
& y\,[\,(\mathbf{y}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{y}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{y}\cdot\mathbf{n})\,\mathbf{n}\,] + \\
& z\,[\,(\mathbf{z}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{z}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{z}\cdot\mathbf{n})\,\mathbf{n}\,]
\end{aligned}$$

Change of Orthonormal Basis

$$\begin{aligned}
\mathbf{p} = {} & x\,[\,(\mathbf{x}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{x}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{x}\cdot\mathbf{n})\,\mathbf{n}\,] + \\
& y\,[\,(\mathbf{y}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{y}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{y}\cdot\mathbf{n})\,\mathbf{n}\,] + \\
& z\,[\,(\mathbf{z}\cdot\mathbf{u})\,\mathbf{u} + (\mathbf{z}\cdot\mathbf{v})\,\mathbf{v} + (\mathbf{z}\cdot\mathbf{n})\,\mathbf{n}\,]
\end{aligned}$$

Rewrite:

$$\begin{aligned}
\mathbf{p} = {} & [\,x\,(\mathbf{x}\cdot\mathbf{u}) + y\,(\mathbf{y}\cdot\mathbf{u}) + z\,(\mathbf{z}\cdot\mathbf{u})\,]\,\mathbf{u} + \\
& [\,x\,(\mathbf{x}\cdot\mathbf{v}) + y\,(\mathbf{y}\cdot\mathbf{v}) + z\,(\mathbf{z}\cdot\mathbf{v})\,]\,\mathbf{v} + \\
& [\,x\,(\mathbf{x}\cdot\mathbf{n}) + y\,(\mathbf{y}\cdot\mathbf{n}) + z\,(\mathbf{z}\cdot\mathbf{n})\,]\,\mathbf{n}
\end{aligned}$$

Change of Orthonormal Basis

$$\mathbf{p} = [\, x\,(\mathbf{x}\cdot\mathbf{u}) + y\,(\mathbf{y}\cdot\mathbf{u}) + z\,(\mathbf{z}\cdot\mathbf{u})\,]\,\mathbf{u} +$$
$$[\, x\,(\mathbf{x}\cdot\mathbf{v}) + y\,(\mathbf{y}\cdot\mathbf{v}) + z\,(\mathbf{z}\cdot\mathbf{v})\,]\,\mathbf{v} +$$
$$[\, x\,(\mathbf{x}\cdot\mathbf{n}) + y\,(\mathbf{y}\cdot\mathbf{n}) + z\,(\mathbf{z}\cdot\mathbf{n})\,]\,\mathbf{n}$$

$$\mathbf{p} = (u,v,n) = u\,\mathbf{u} + v\,\mathbf{v} + n\,\mathbf{n}$$

Expressed in **uvn** basis:

$$u = x\,(\mathbf{x}\cdot\mathbf{u}) + y\,(\mathbf{y}\cdot\mathbf{u}) + z\,(\mathbf{z}\cdot\mathbf{u})$$
$$v = x\,(\mathbf{x}\cdot\mathbf{v}) + y\,(\mathbf{y}\cdot\mathbf{v}) + z\,(\mathbf{z}\cdot\mathbf{v})$$
$$n = x\,(\mathbf{x}\cdot\mathbf{n}) + y\,(\mathbf{y}\cdot\mathbf{n}) + z\,(\mathbf{z}\cdot\mathbf{n})$$

Change of Orthonormal Basis

$$u = x\,(\mathbf{x}\cdot\mathbf{u}) + y\,(\mathbf{y}\cdot\mathbf{u}) + z\,(\mathbf{z}\cdot\mathbf{u})$$
$$v = x\,(\mathbf{x}\cdot\mathbf{v}) + y\,(\mathbf{y}\cdot\mathbf{v}) + z\,(\mathbf{z}\cdot\mathbf{v})$$
$$n = x\,(\mathbf{x}\cdot\mathbf{n}) + y\,(\mathbf{y}\cdot\mathbf{n}) + z\,(\mathbf{z}\cdot\mathbf{n})$$

In matrix form:

$$\begin{pmatrix} u \\ v \\ n \end{pmatrix} = \begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ n_x & n_y & n_z \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

where:
$u_x = \mathbf{x}\cdot\mathbf{u}$
$u_y = \mathbf{y}\cdot\mathbf{u}$
etc.

Change of Orthonormal Basis

$$\begin{pmatrix} u \\ v \\ n \end{pmatrix} = \begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ n_x & n_y & n_z \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = M \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

What's $\mathbf{M^{-1}}$, the inverse?

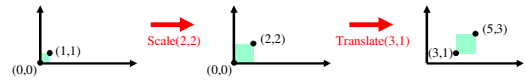
$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_u & x_v & x_n \\ y_u & y_v & y_n \\ z_u & z_v & z_n \end{pmatrix} \begin{pmatrix} u \\ v \\ n \end{pmatrix}$$

$u_x = \mathbf{x}\cdot\mathbf{u} = \mathbf{u}\cdot\mathbf{x} = x_u$

$$\mathbf{M^{-1}} = \mathbf{M^T}$$

Exemple

Échelle et translater

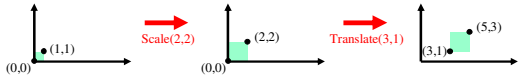


Concatenation de matrices: $p' = T(Sp) = TSp$

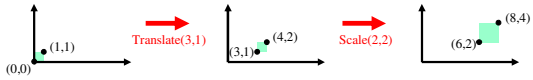
$$TS = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

PAS commutative

Échelle et translater: $p' = T(Sp) = TSp$



Translater et échelle: $p' = S(Tp) = STp$



PAS commutative

$$TS = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

$$ST = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 6 \\ 0 & 2 & 2 \\ 0 & 0 & 1 \end{pmatrix}$$

Exemple tableau

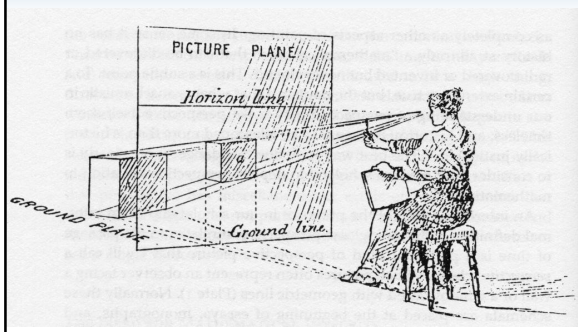
Plan

- Transformations géométriques
- Projections, caméra, perspective
- Élimination des parties cachées

Projections et perspectives

- Affichage d'un modèle 3D
- Fenêtre 2D
- Transformation des coordonnées 3D du modèle vers les coordonnées 2D des pixels

Projection perspective



Projections/Camera

- Rappel: Une projection est:
- Origine:
  - position de l'observateur
- Direction de vue:
  - vers laquelle l'observateur est tourné
- Direction verticale:
  - la verticale pour l'observateur

Différent types de projections

- Projections parallèles :
  - Pas de rétrécissement des objets
  - Plusieurs types de projections parallèles :
    - isométrique, cavalière,…
- Projection perspective :
  - Les objets lointains sont plus petits
  - Plusieurs types :
    - Un, deux ou trois points de fuite

Décrire une projection

- Projection par une application linéaire P
- Q point, sa projection PQ=(x,y,z,w)
- (x/w,y/w) position sur l'écran
- z/w une profondeur

Projection ortho/parallèle

- Vue dans la direction de l'axe –Z
- Qu'est-ce P???

Projection parallèle

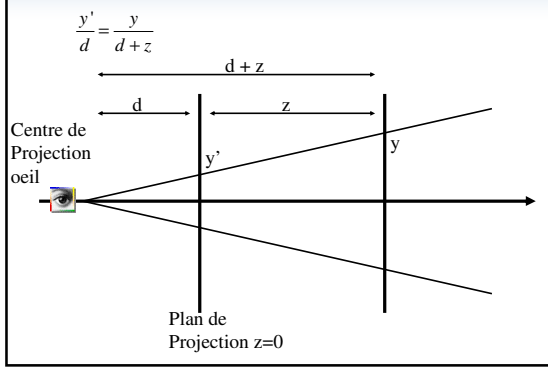
- Projection parallèle sur le plan *z=0* :
  - *x'=x, y'=y, w'=w*
- Matrice de projection P:

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

Projection Perspective

- Un peu plus compliqué.
- Vue dans direction de l'axe –Z.
- Plan de projection Z=0
- Origine (0,0,-d,1)

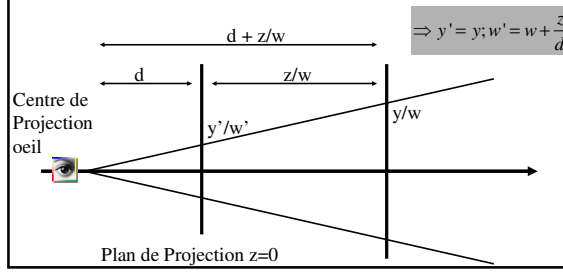
Projection Perspective



Projection Perspective en coordennées homogenes

$$\frac{\frac{y'}{w'}}{d} = \frac{\frac{y}{w}}{d+\frac{z}{w}} \Rightarrow \frac{y'}{dw'} = \frac{y}{wd+z} \Rightarrow \frac{y'}{w'} = \frac{1}{\frac{1}{d}} * \frac{y}{wd+z} = \frac{y}{w+\frac{z}{d}}$$

$$\Rightarrow y' = y; w' = w + \frac{z}{d}$$



Projection perspective

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

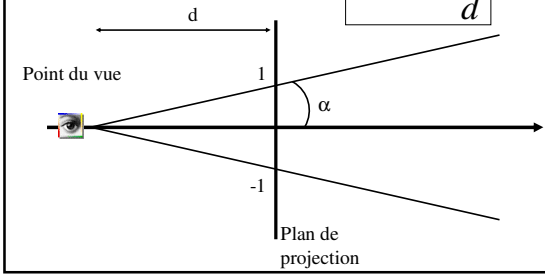
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix}$$

Ouverture de vue

- « d » n'est pas intuitif
- Ouverture de vue : plus simple
  - Angle
  - Exprime la largeur du champ visuel

Ouverture

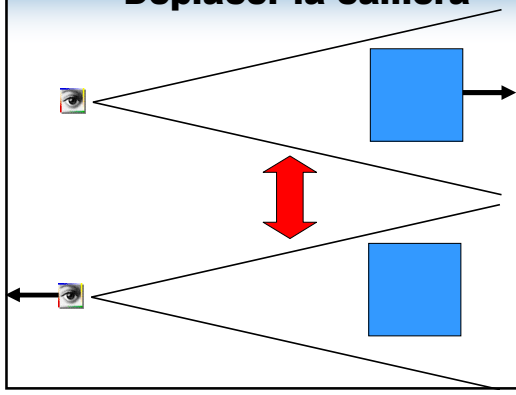
$$\tan\alpha = \frac{1}{d}$$



Autres positions de la caméra

- Comment passer à un modèle général ?
  - Point de vue quelconque, direction quelconque…

Déplacer la caméra



Attention

- Open GL utilise une facon bizarre de multiplier des Matrices!

```
glMultMatrix(A);
glMultMatrix(B);
glVertex3f(v);
```

Donne ABv et pas BAv

Attention

- Open GL utilise une facon bizarre de multiplier des Matrices!

```
glMultMatrix(A);
glMultMatrix(B);
glVertex3f(v);
```

Donne ABv et pas BAv

Avantage: On peut d'abord préciser la caméra

Attention

- Néanmoins ce n'est pas si grave…
- Ordre du code = penser en local
- Ordre inverse = penser en global
- Expliqué en TP…

Slides

- Contributions de:
  - **Briceno, H., Notes du cours SI, UFRIMA**
  - **Boyer, E., Notes du cours SI , UFRIMA**
  - **Holzschuch, N., « Notes du cours DEA-IVR, ENSIMAG, Création d'Images Virtuelles ». 2005-2006**
  - **Frédo Durand and Barbara Cuttler, SI, MIT**
- **Images taken from various sources:**
  IF ANY IMAGE IN THIS PRESENTATION IS NOT ALLOWED TO BE USED, PLEASE CONTACT ME AND I WILL DELETE IT!
  TO MY BEST KNOWLEDGE ALL IMAGES CAN BE USED FOR UNIVERSITY COURSES.