

<http://tinyurl.com/ift3355>

IFT 3355: INFOGRAPHIE

LA CAMÉRA

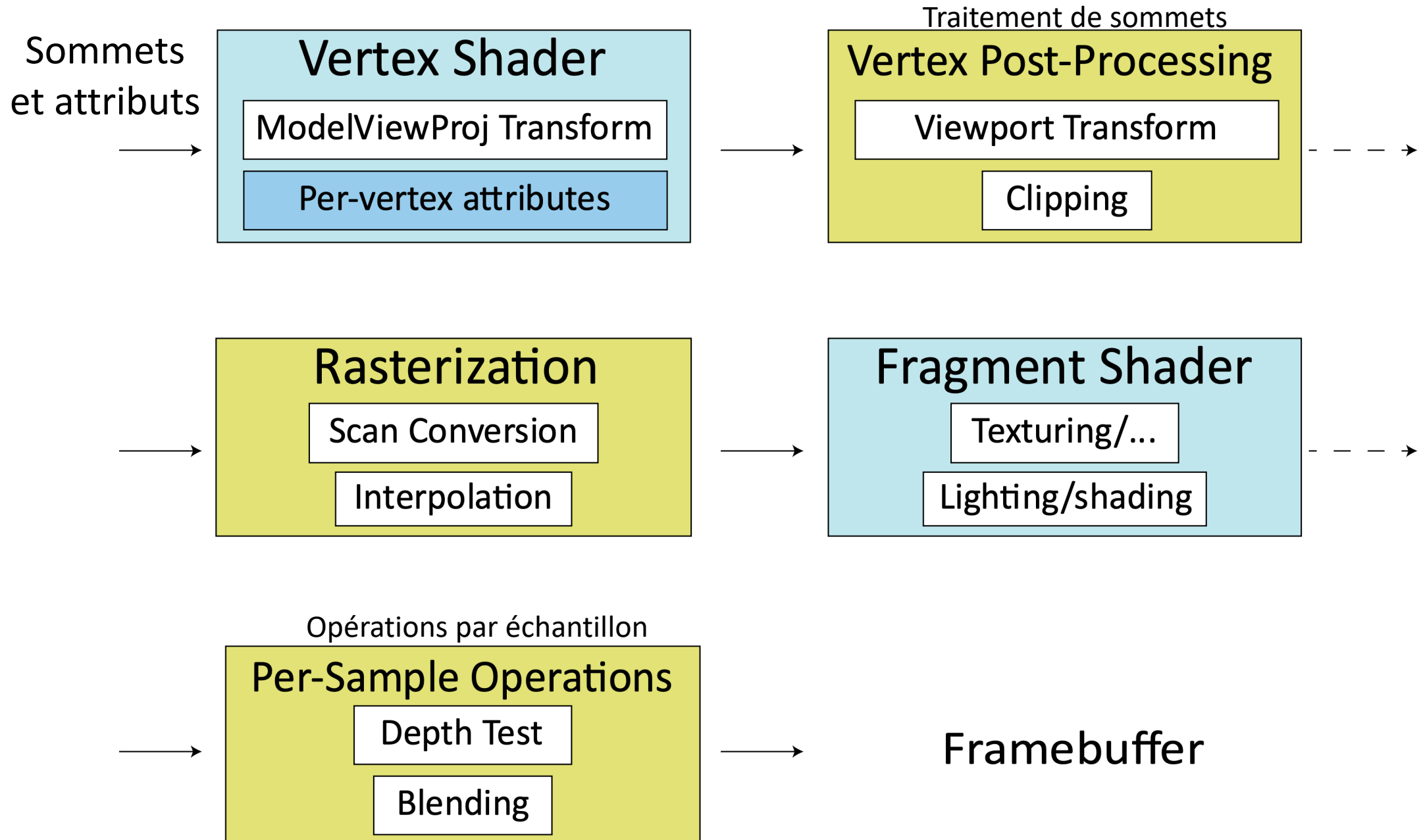
LES PROJECTIONS

Livre de référence: G:III.10; S: 7.2

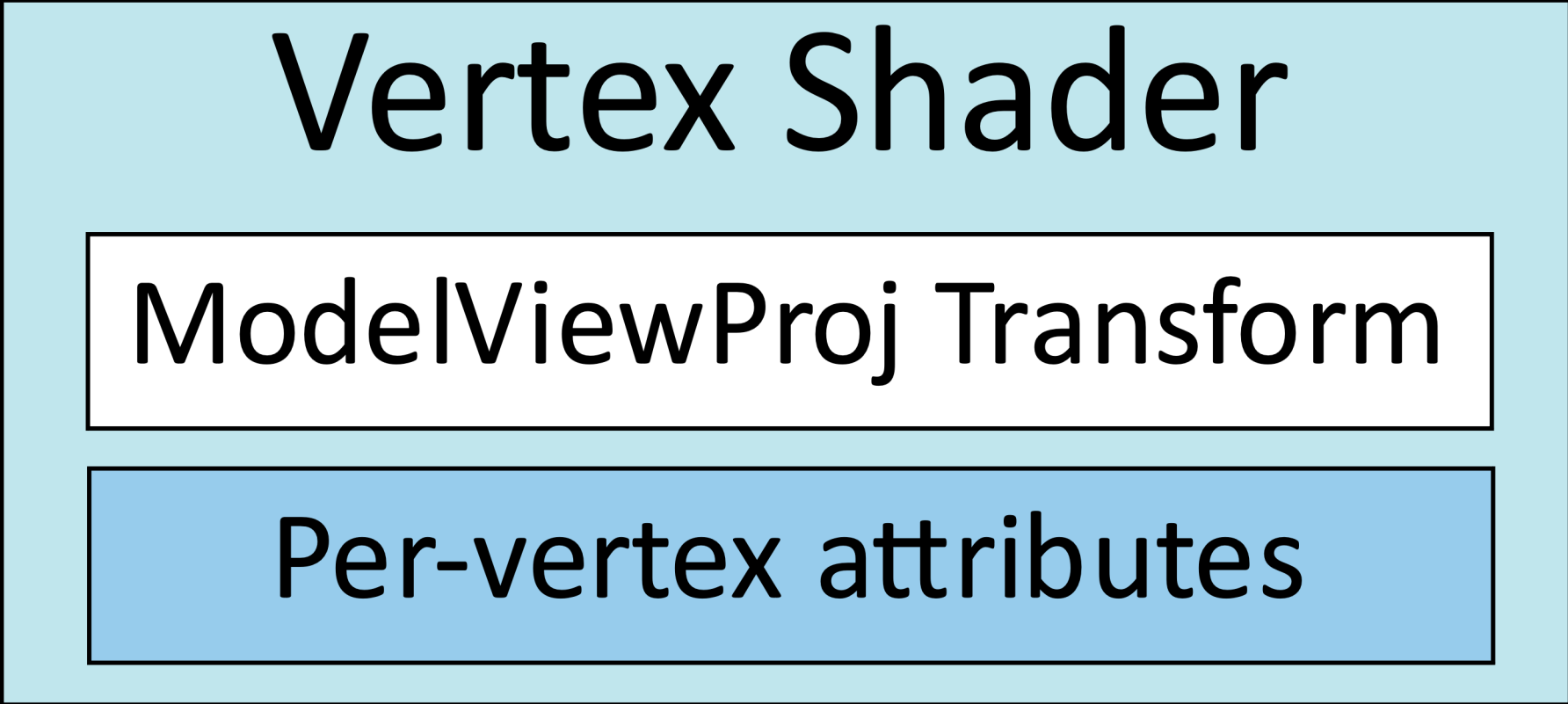
Mikhail Bessmeltsev



PIPELINE: PLUS DE DÉTAILS

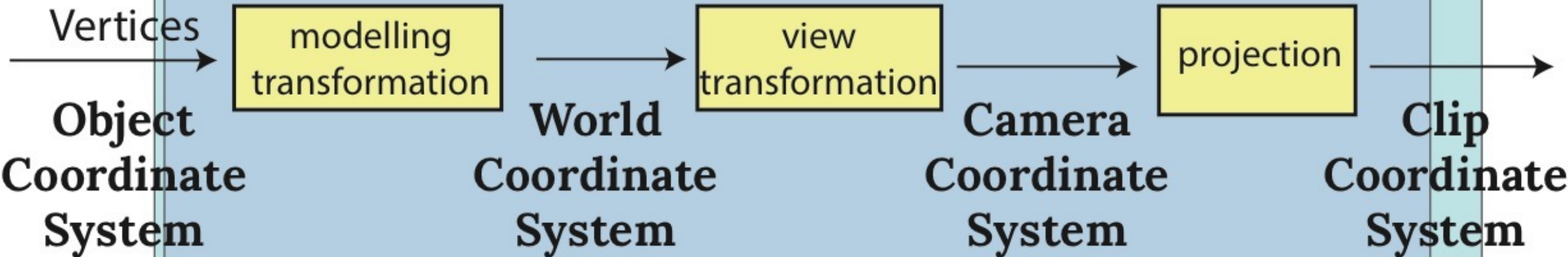


Sommets
et attributs
→

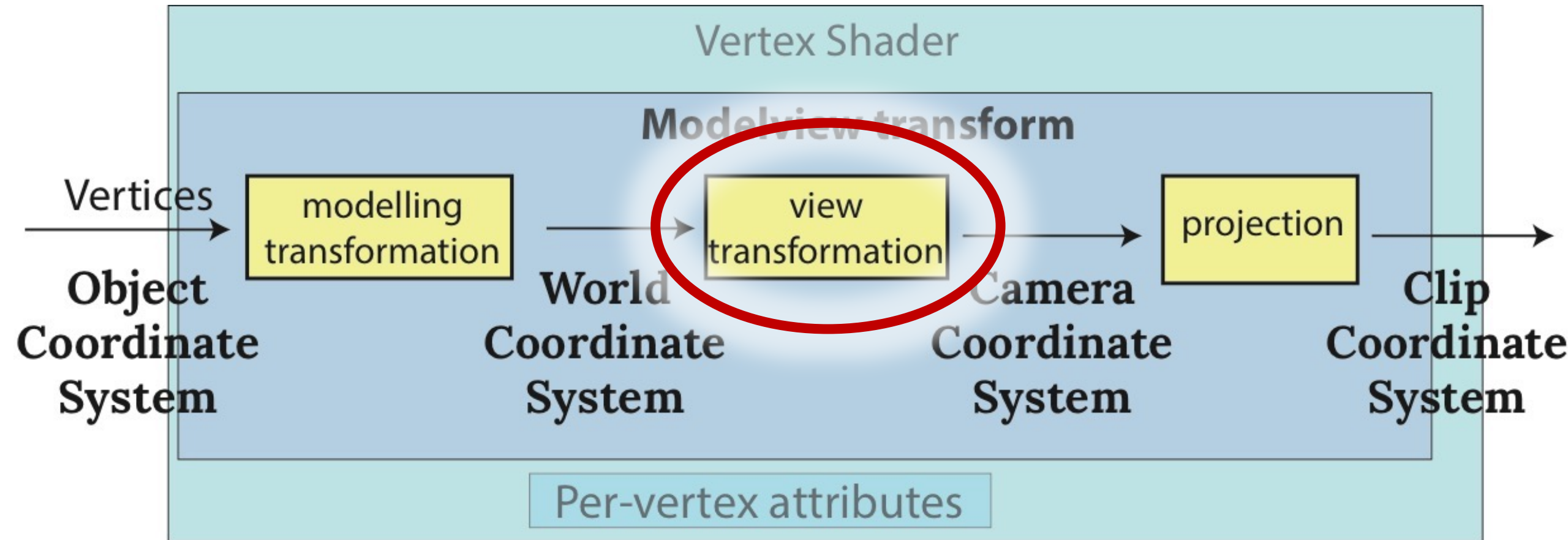


Vertex Shader

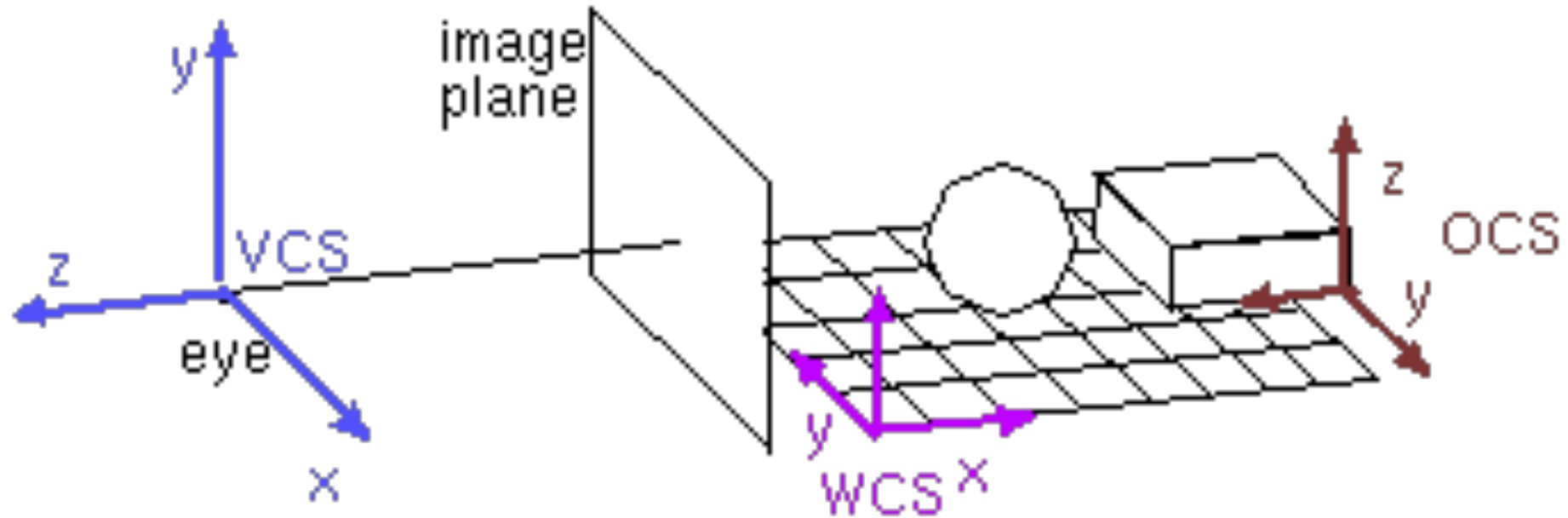
Modelview transform



VERTEX SHADER: REGARDER DE PLUS PRÈS

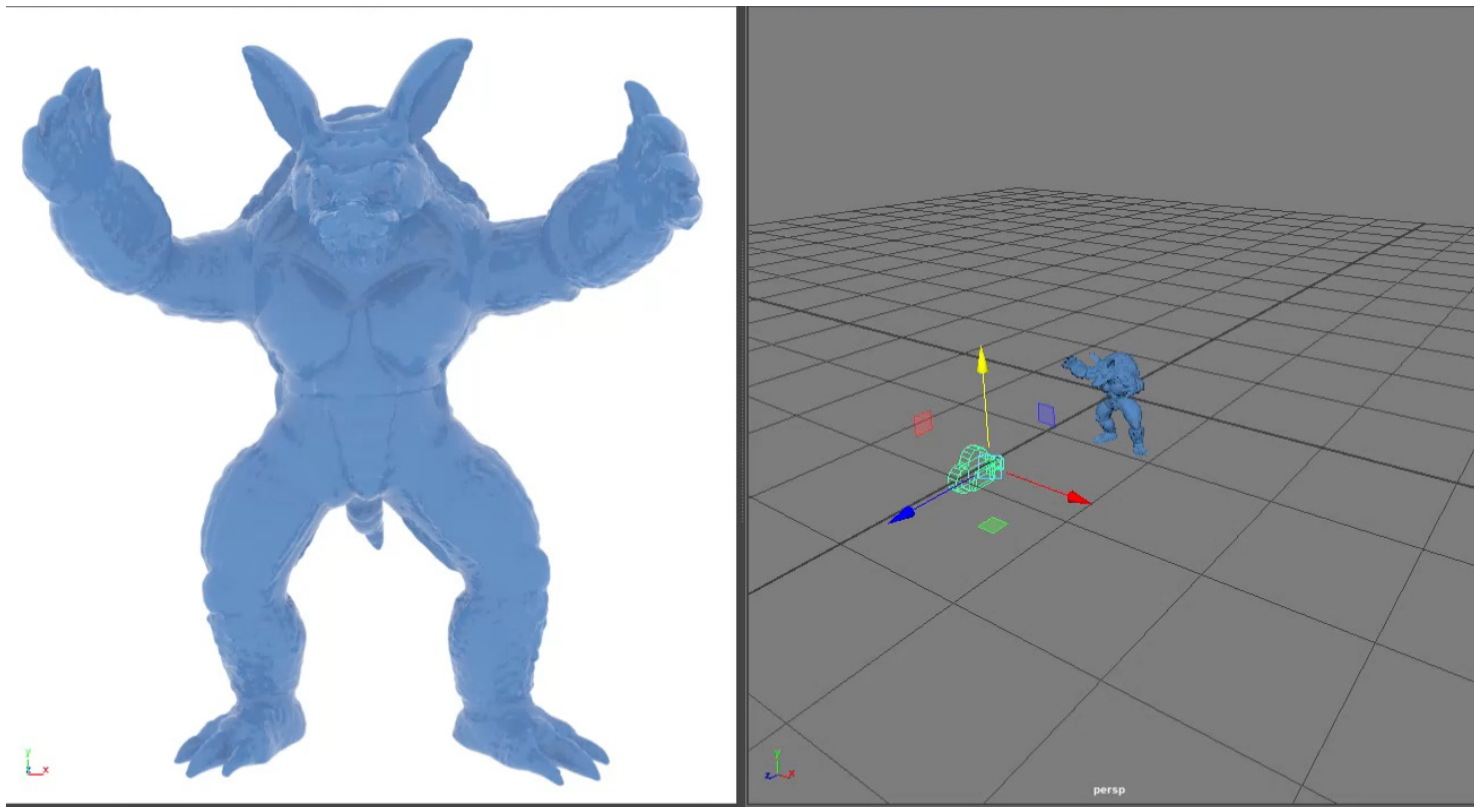


POSITIONNEMENT DE LA CAMÉRA



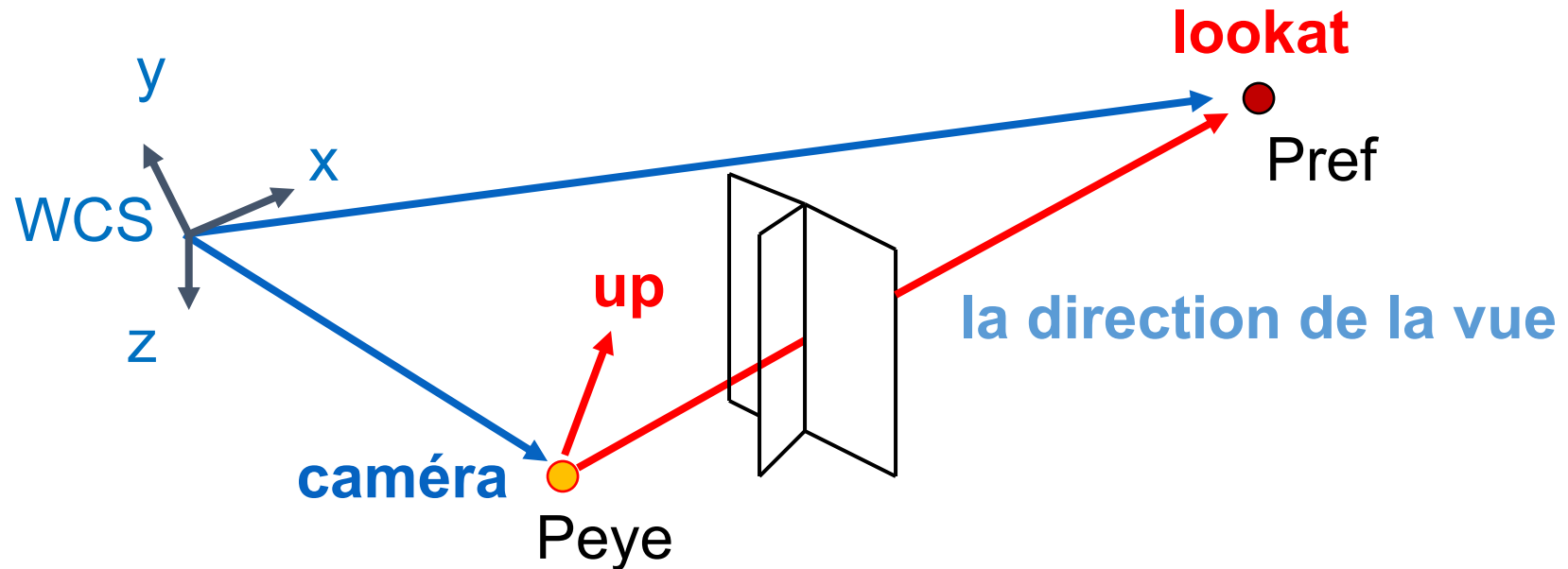
DÉFINITION DE LA CAMÉRA

- Point de l'œil (ou point focal) (*où est la caméra?*)
- Point de référence (*quel point regarde-t-elle?*)
- Le vecteur vers le haut (*up vector*)



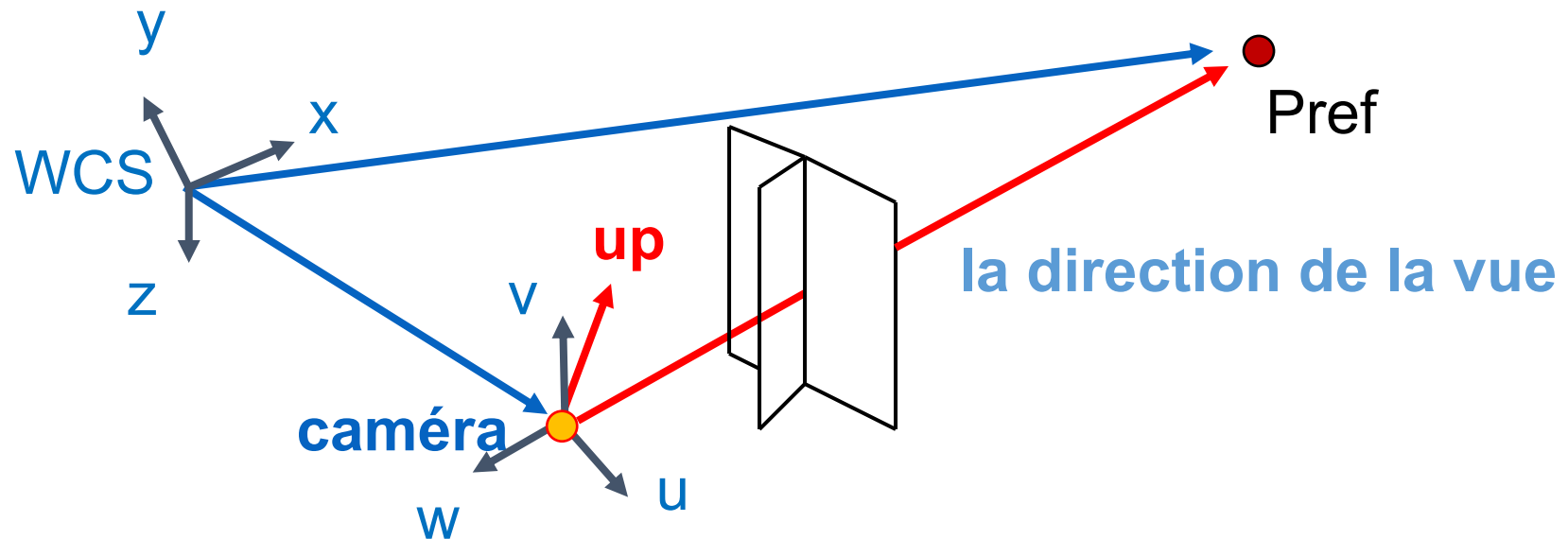
DÉFINITION DE LA CAMÉRA

- Point de l'œil (ou point focal) (*où est la caméra?*)
- Point de référence (*quel point regarde-t-elle?*)
- Le vecteur vers le haut (up vector)



DÉFINITION DE LA CAMÉRA

- Point de l'œil (ou point focal) (*où est la caméra?*)
- Point de référence (*quel point regarde-t-elle?*)
- Le vecteur vers le haut (up vector)

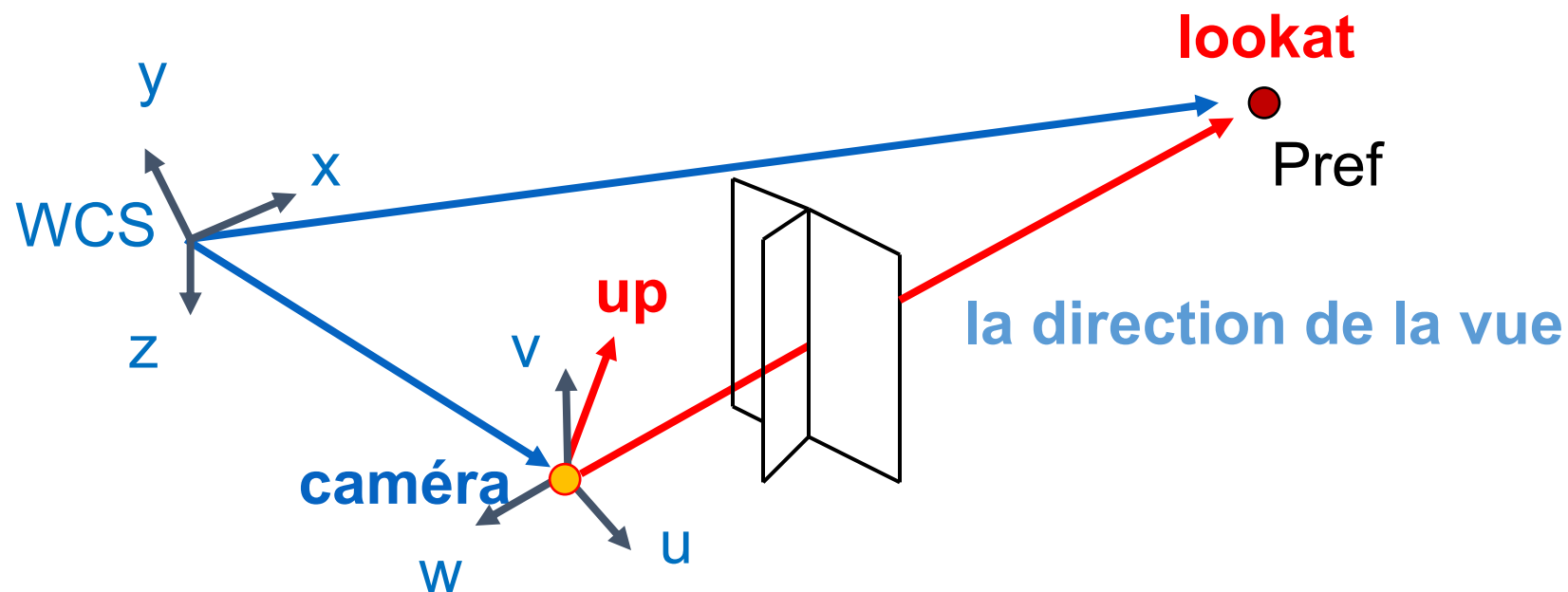


LE SYSTÈME DES COORDONNÉES DE LA CAMÉRA

- $w = \frac{p_{eye_pref}}{\|p_{eye_pref}\|}$

- $u = \frac{v_{up} \times w}{\|v_{up} \times w\|}$

- $v = w \times u$



LA MATRICE DE LA CAMÉRA

$$M_{cam} = \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix}$$

LA MATRICE DE LA CAMÉRA

$$M_{cam} = \begin{bmatrix} u_x & v_x & w_x & P_x^{eye} \\ u_y & v_y & w_y & P_y^{eye} \\ u_z & v_z & w_z & P_z^{eye} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

LA MATRICE DE LA CAMÉRA

$$M_{cam} = \begin{bmatrix} u_x & v_x & w_x & P_x^{eye} \\ u_y & v_y & w_y & P_y^{eye} \\ u_z & v_z & w_z & P_z^{eye} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{view} = M_{cam}^{-1} = \dots$$

THREE.JS

```
camera = new THREE.OrthographicCamera( /*some parameters*/ );  
camera.position.set(30,0,0);  
camera.up = new THREE.Vector3(0,0,1);  
camera.lookAt(new THREE.Vector3(0,0,0));
```


DÉPLACER LE MONDE OU LA CAMÉRA?

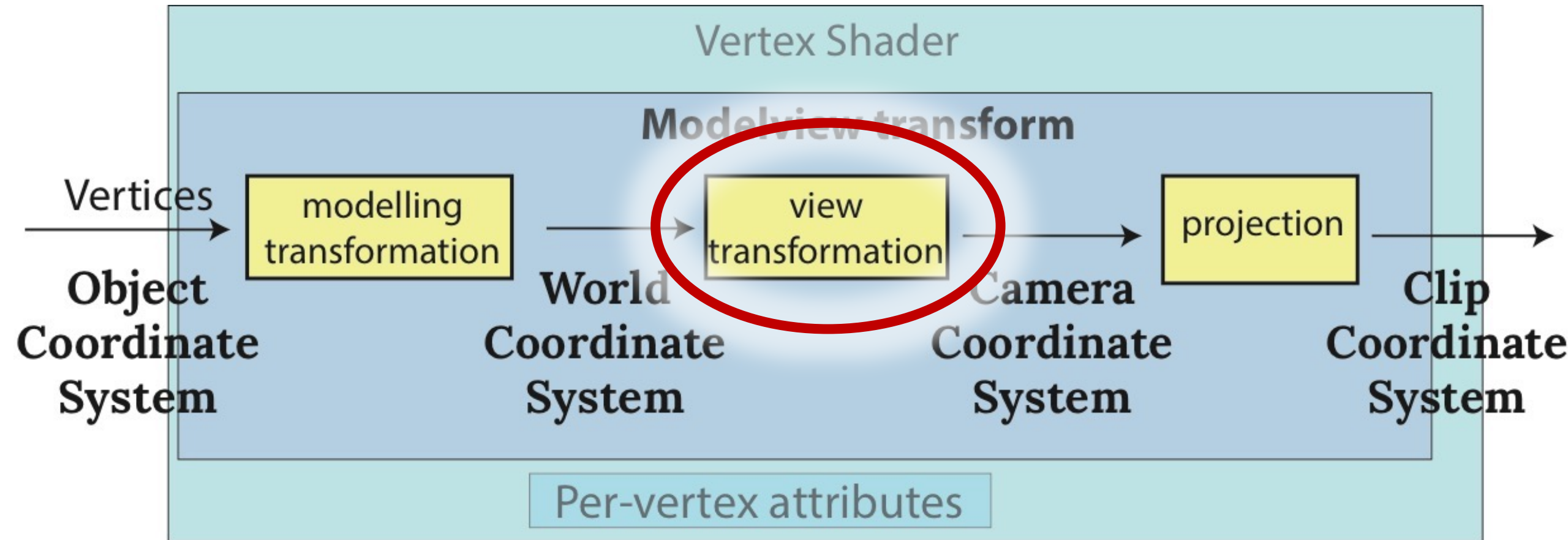
Deux opérations équivalentes!

- Déplacer la caméra vers l'avant = déplacer le monde en arrière

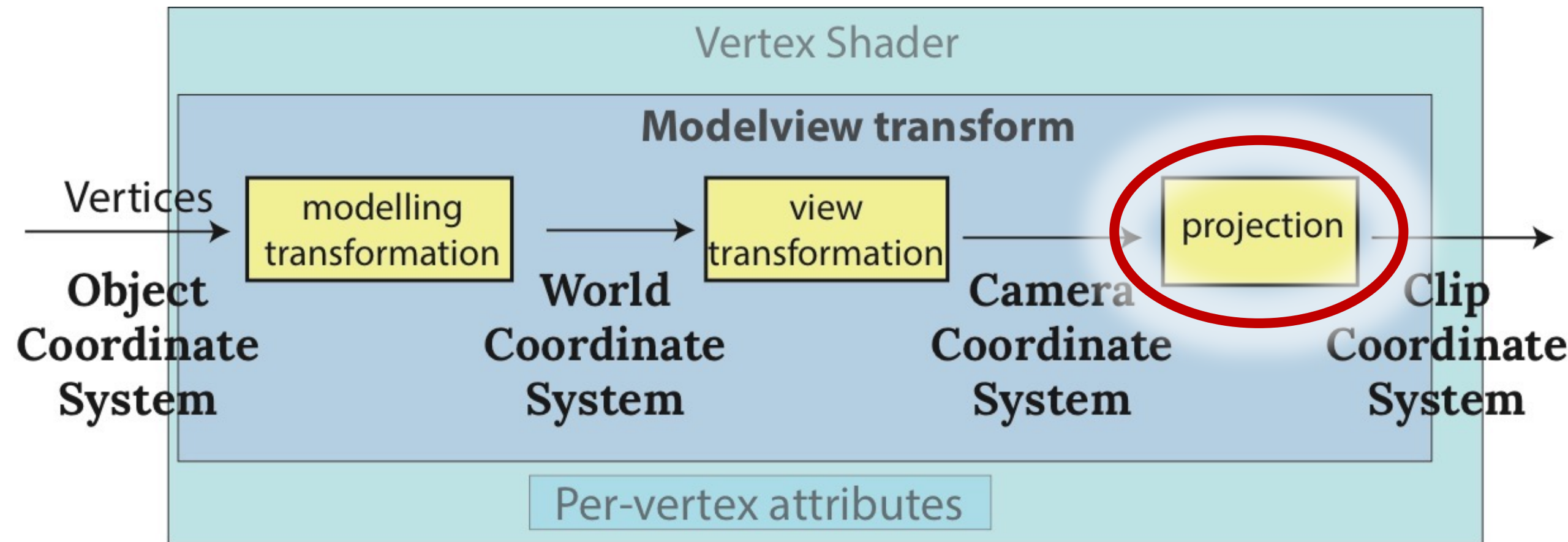
Exemple

- La caméra OpenGL initiale: à l'origine, regardant dans l'axe $-z$
- Créer un carré de taille unité parallèle à la caméra, $z = -10$
- Déplacer en direction de z par 3
 - Déplacer la caméra à $z = -3$
 - Garder la caméra, mais déplacer le carré à $z = -7$
- La même image!

VERTEX SHADER: REGARDER DE PLUS PRÈS



VERTEX SHADER: REGARDER DE PLUS PRÈS



LES PROJECTIONS FONT LA DIFFÉRENCE.



85mm @ 200cm

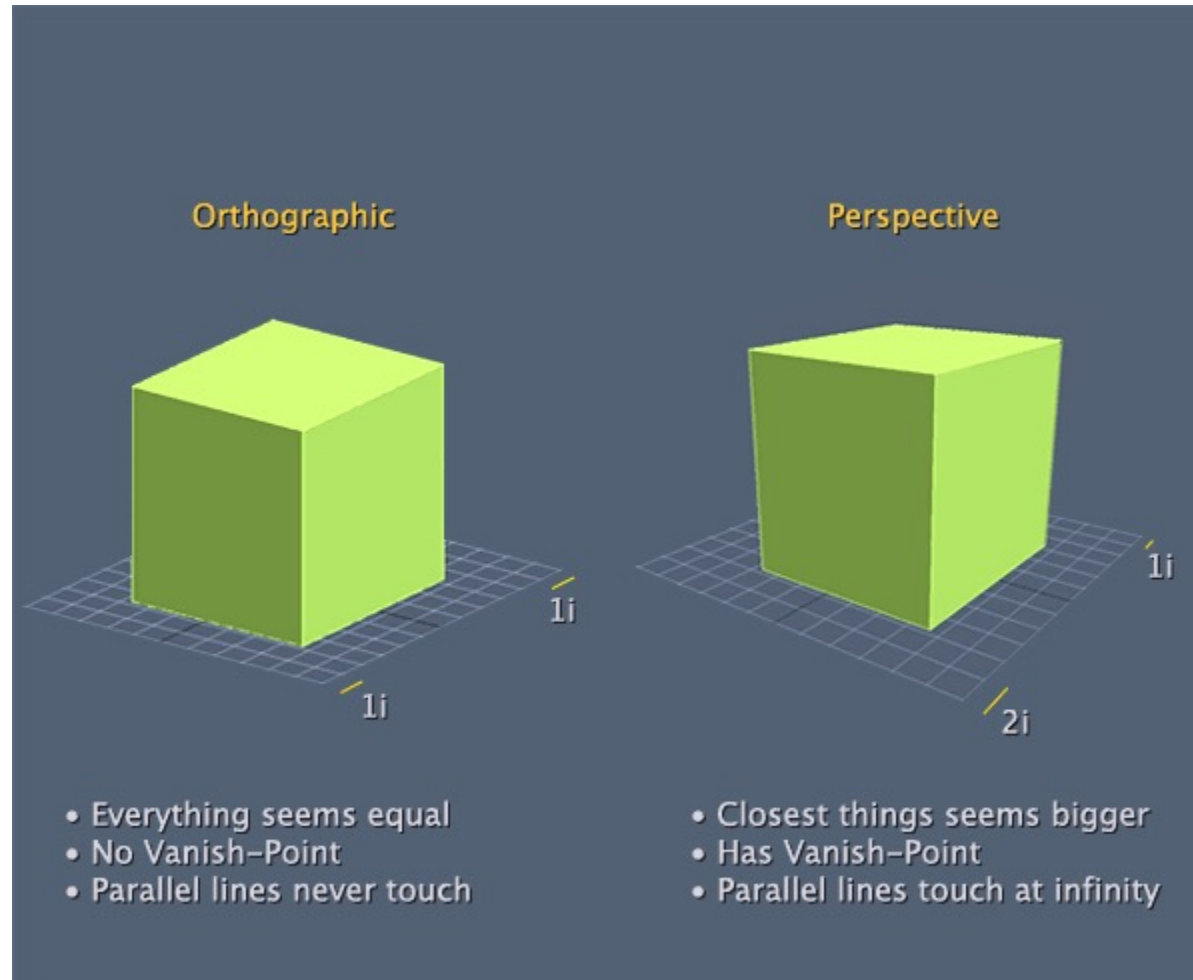
35mm @ 85cm

16mm @ 40cm

12mm @ 30cm

8mm @ 20cm

PROJECTION PERSPECTIVE



UNE PROJECTION



COMMENT CONVERTIR 3D EN 2D?

- Maintenant, toute la scène est dans le système de coordonnées de la caméra (homogènes)
- Comment obtenir les coordonnées homogènes 2D?
- Une méthode: se débarrasser de la coordonnée z

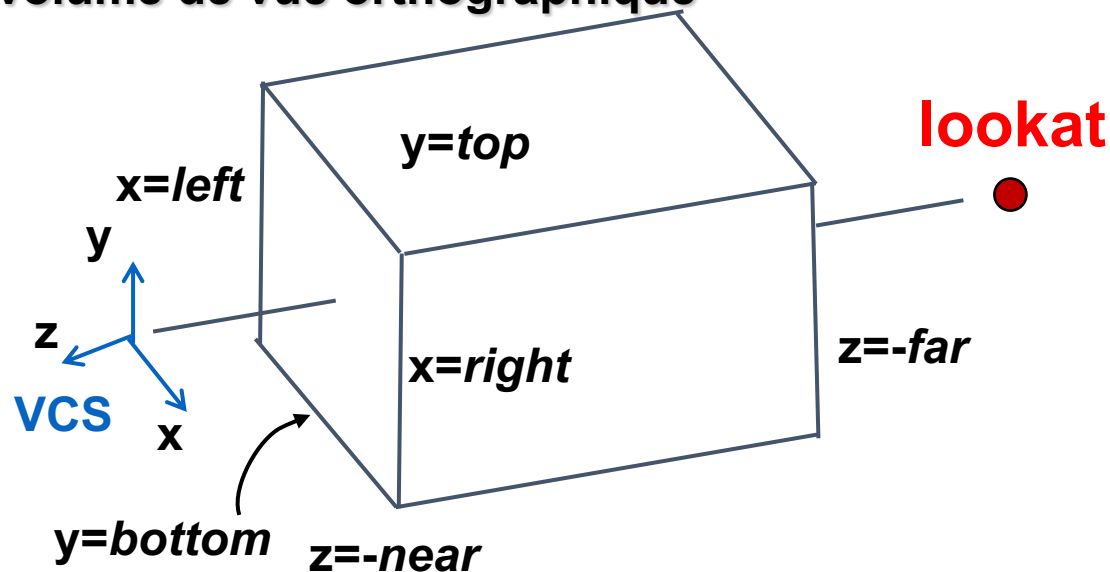
$$M = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

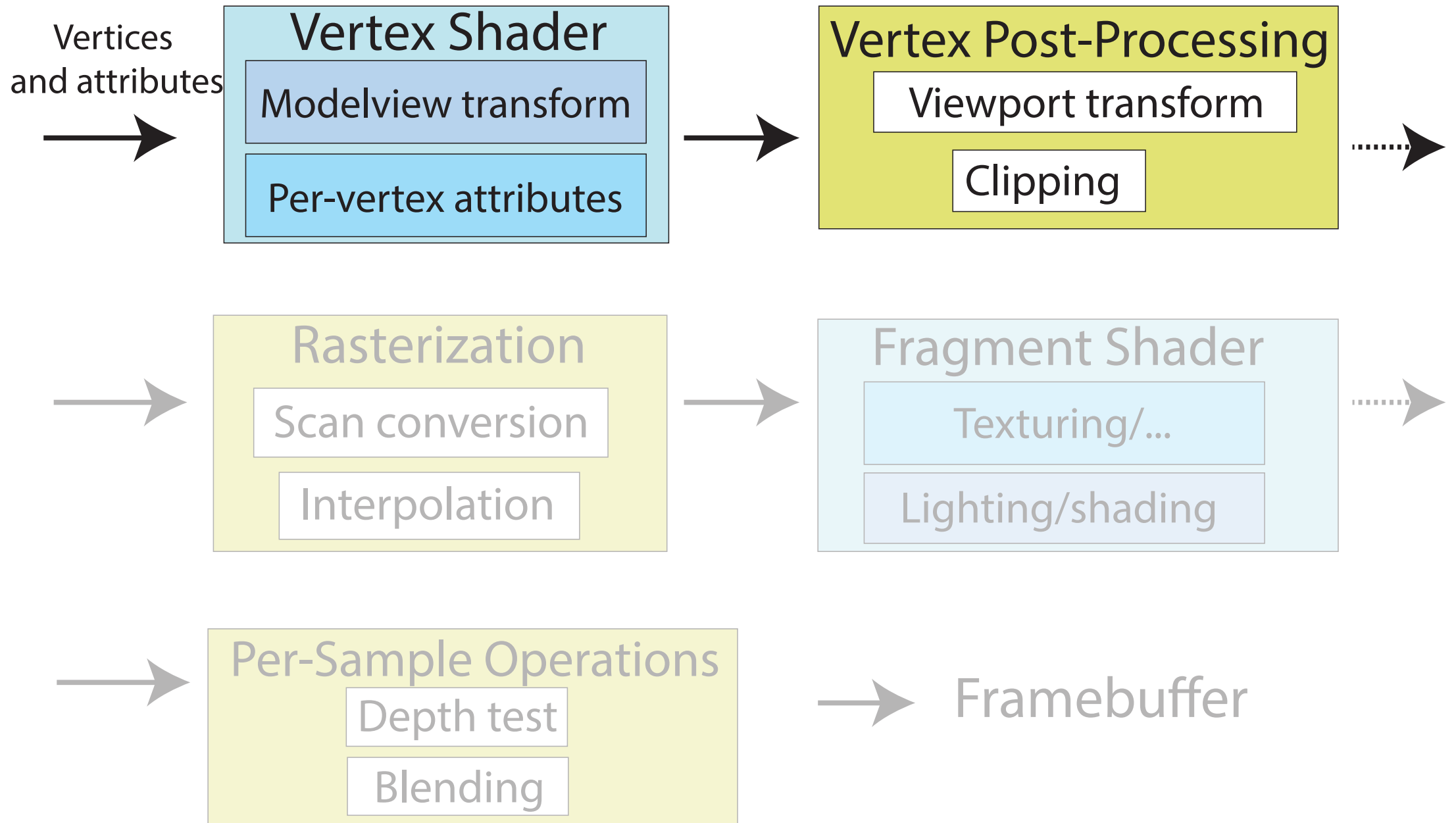
Des problèmes?

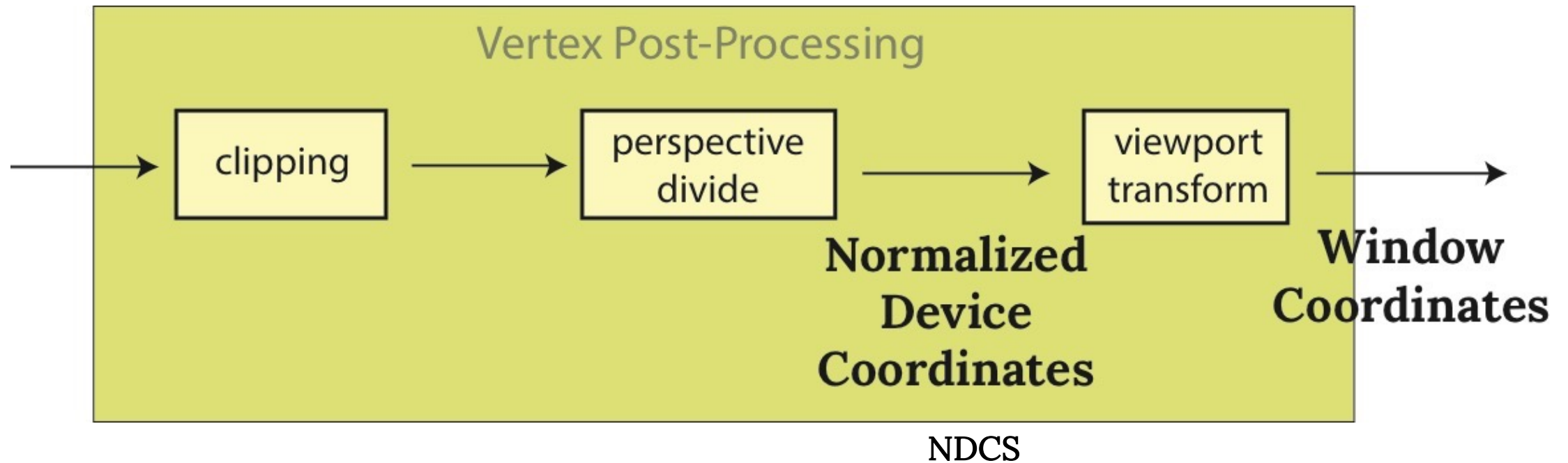
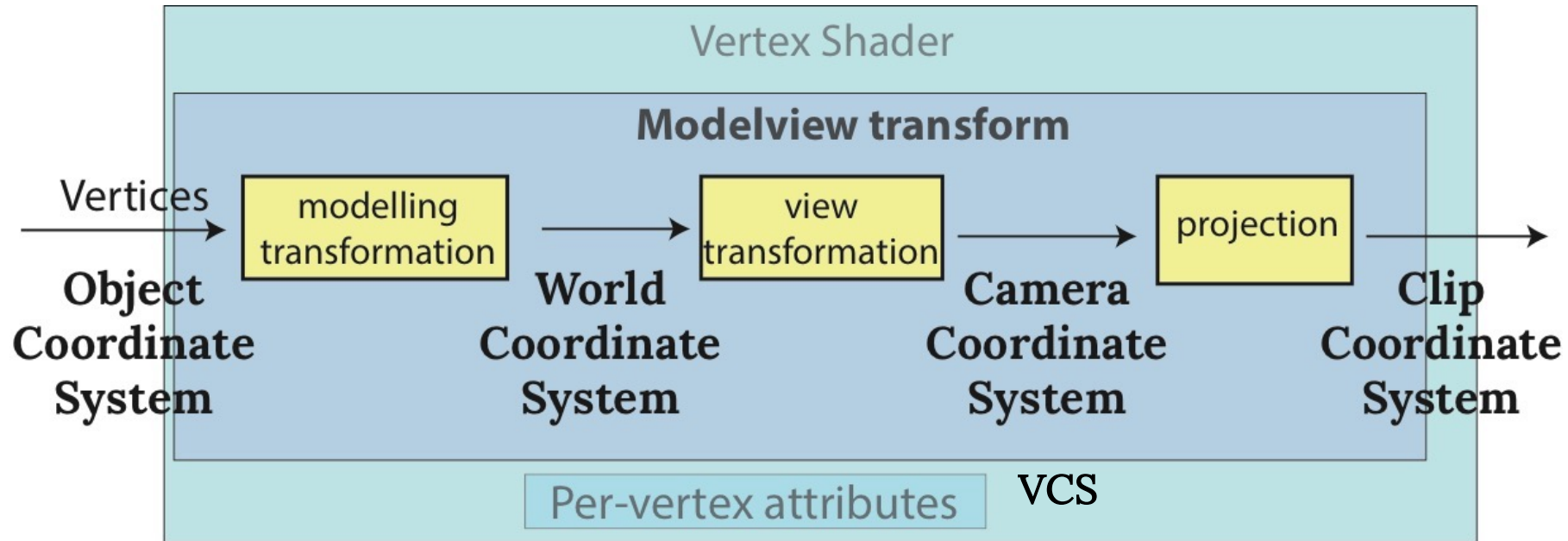
LE VOLUME DE VUE PROJECTION ORTHOGRAPHIQUE

- = Projection parallèle
- Il spécifie le champ de vue, utilisé pour le *clipping*
- Il limite le domaine de z stocké pour le test de profondeur

Le volume de vue orthographique

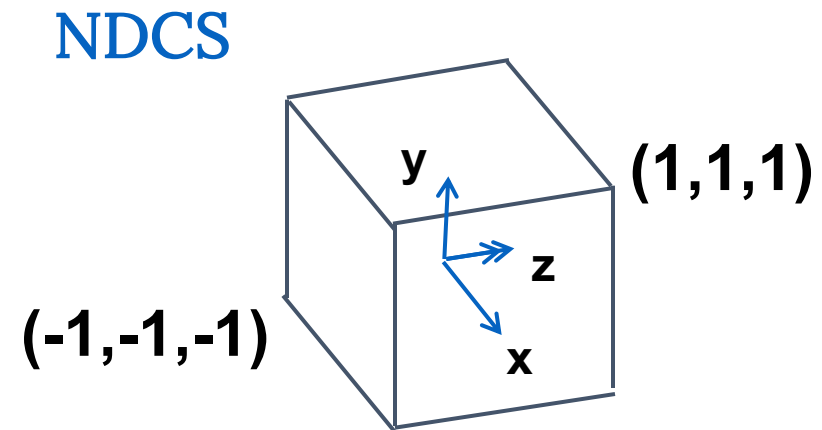
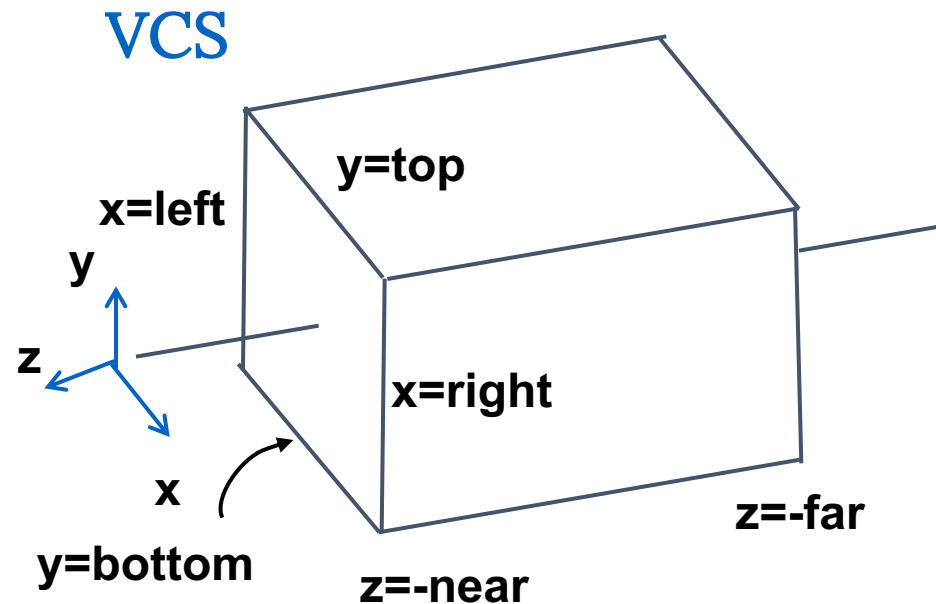






L'AXE Z

- Un flip de l'axe z change *handedness* du système des coordonnées
 - RHS (règle de la main droite) avant la projection
 - les coordonnées de la vue/caméra, du monde
 - LHS (règle de la main gauche) après
 - clip, les coordonnées de l'appareil normalisées



POURQUOI ON A BESOIN DES AVANT-PLAN & ARRIÈRE-PLAN?

On stocke (souvent) la profondeur dans une représentation en point fixe, donc toutes les valeurs doivent être dans un intervalle limité (0..1)

- Avant-plan:
 - Éviter la singularité de la perspective (division par zéro, ou les nombres trop petits)
- Arrière-plan:
 - Éviter les artefacts de précision numérique pour les objets trop loin

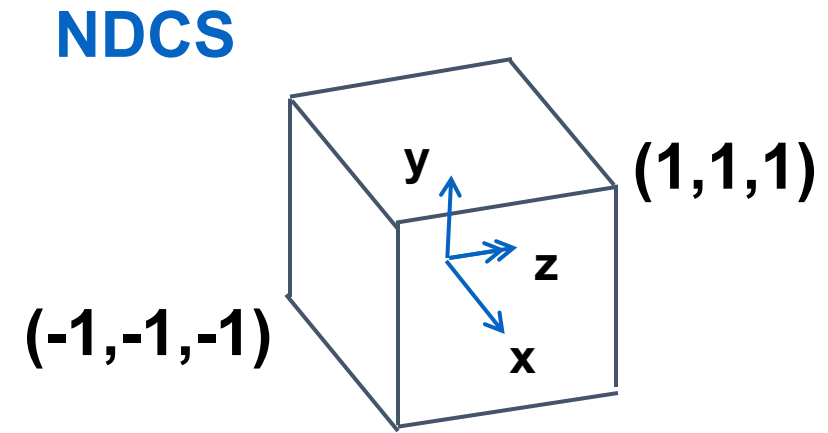
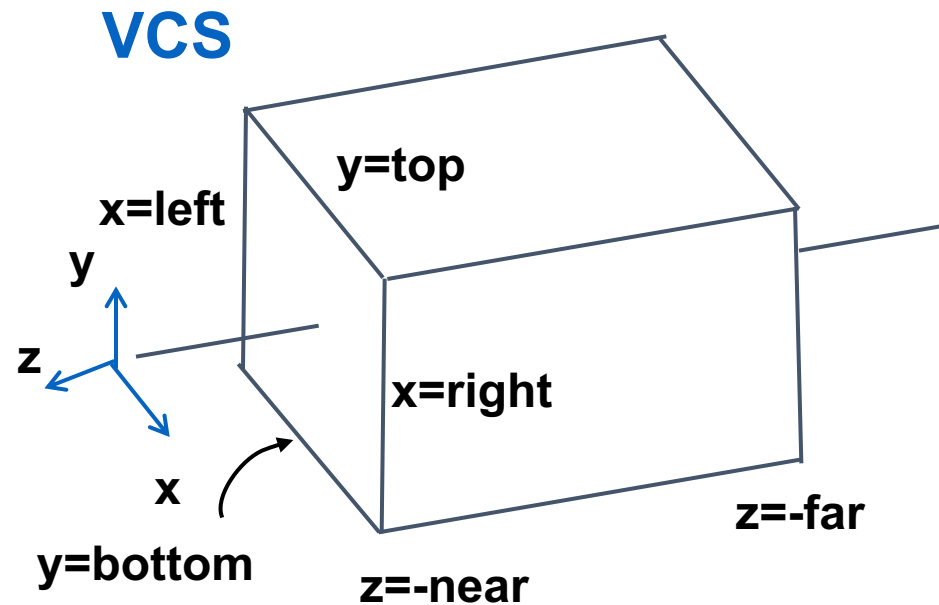
TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, déplacer, refléter

$top \rightarrow 1$

$bottom \rightarrow -1$

$$y' = a \cdot y + b$$



TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, déplacer, refléter

$$top \rightarrow 1$$

$$bottom \rightarrow -1$$

$$y' = a \cdot y + b$$

- Ou résoudre deux équations:

$$\begin{cases} 1 = a \cdot top + b \\ -1 = a \cdot bottom + b \end{cases}$$

\Rightarrow

$$a = \frac{2}{top - bottom}$$

$$b = \frac{-top - bottom}{top - bottom}$$

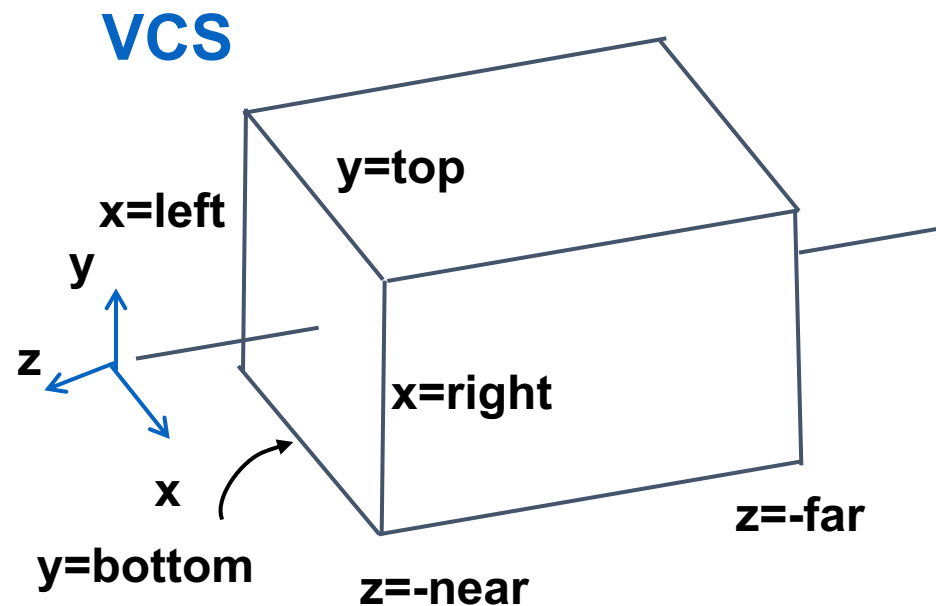
TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

$top \rightarrow 1$
 $bottom \rightarrow -1$

$$y' = a \cdot y + b$$

$$a = \frac{2}{top - bottom}$$

$$b = \frac{-top - bottom}{top - bottom}$$



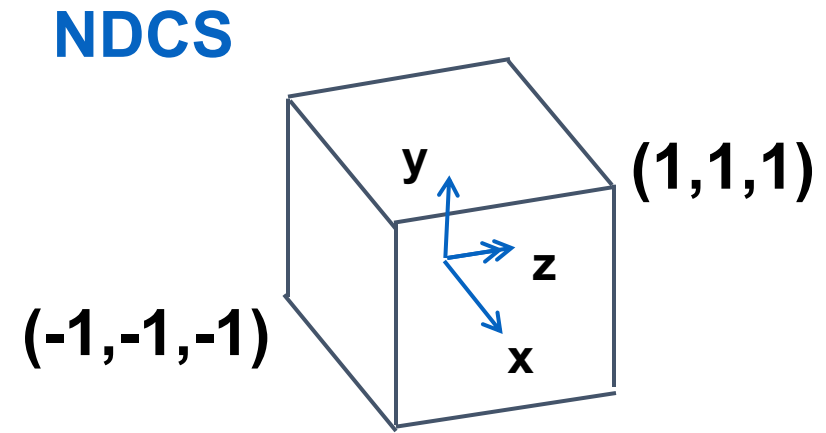
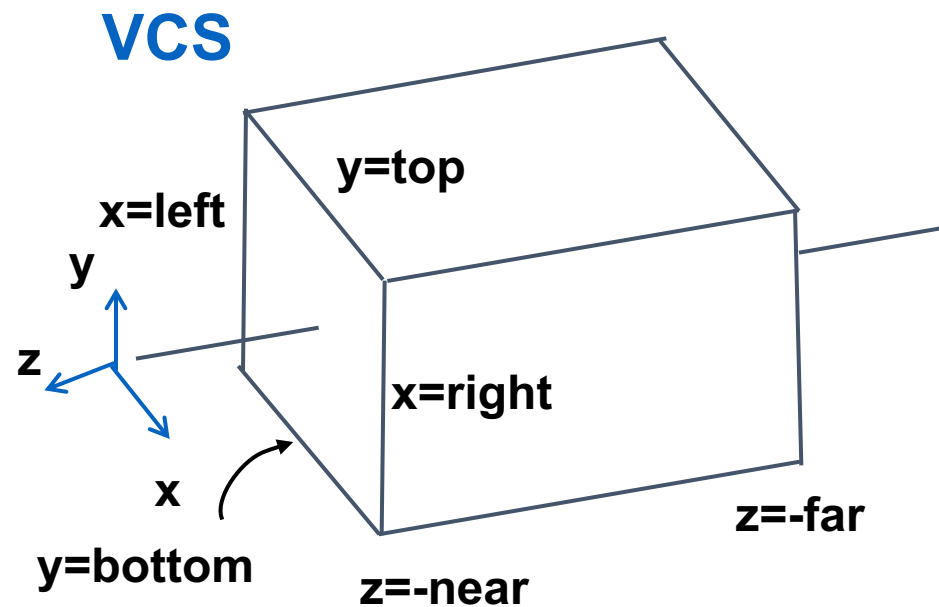
La même idée pour gauche+droite, arrière+avant

TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, déplacer, refléter

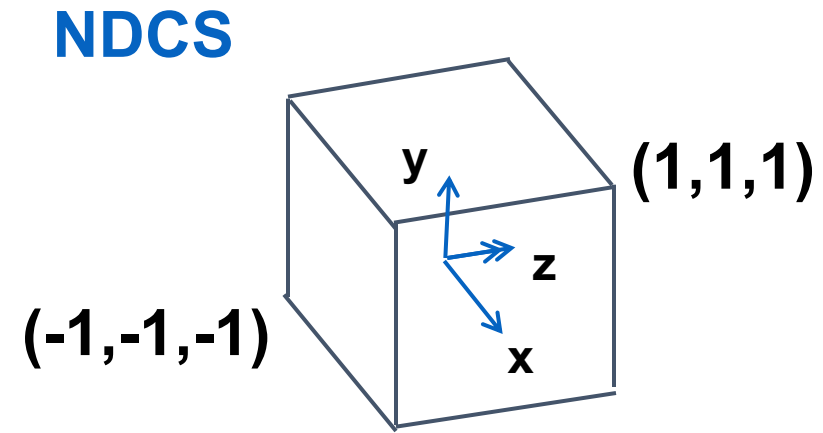
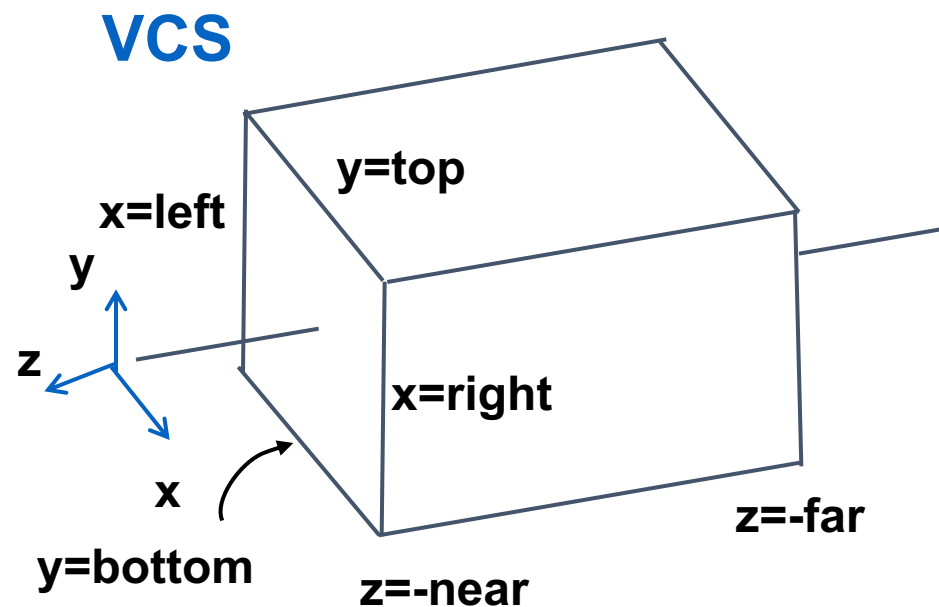
$$P' = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bot} & 0 & -\frac{top + bot}{top - bot} \\ 0 & 0 & \frac{-2}{far - near} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix} P$$

... VOICI LA MEILLEURE FAÇON



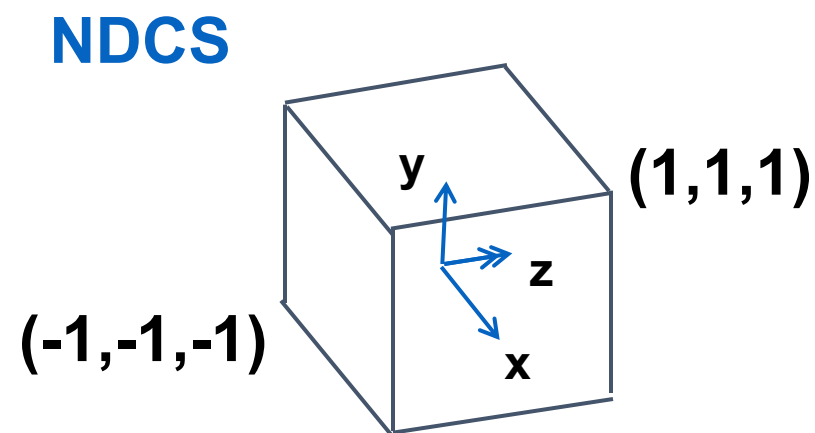
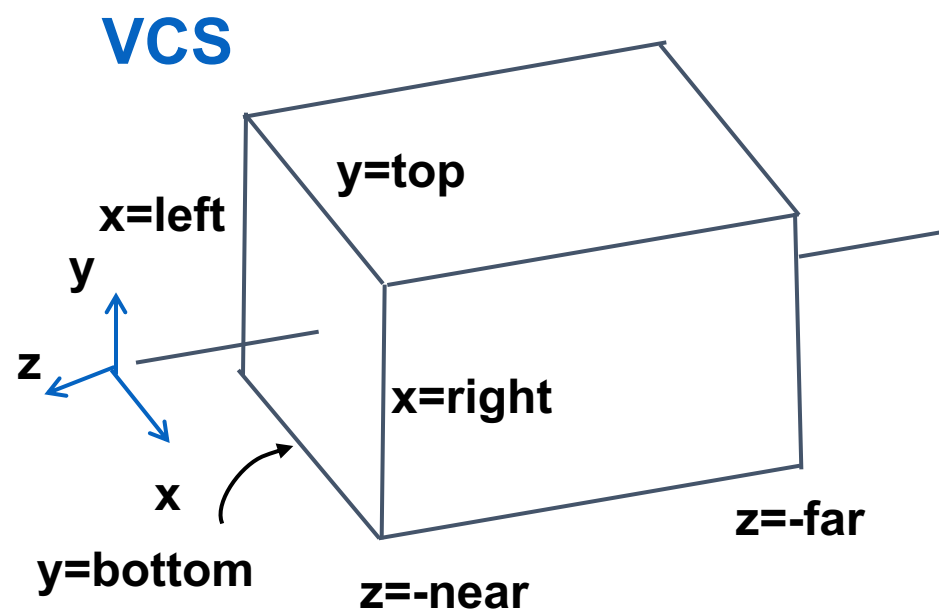
... VOICI LA MEILLEURE FAÇON

- Au début, déplacer les sommets, puis changer l'échelle!



... VOICI LA MEILLEURE FAÇON

- $M = Scale \left(\frac{2}{r-l}, \frac{2}{t-b}, \frac{-2}{f-n} \right) \cdot Tr \left(-\frac{r+l}{2}, -\frac{t+b}{2}, \frac{f+n}{2} \right)$



TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, déplacer, refléter

$$P' = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bot} & 0 & -\frac{top + bot}{top - bot} \\ 0 & 0 & \frac{-2}{far - near} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix} P$$

TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, **déplacer**, refléter

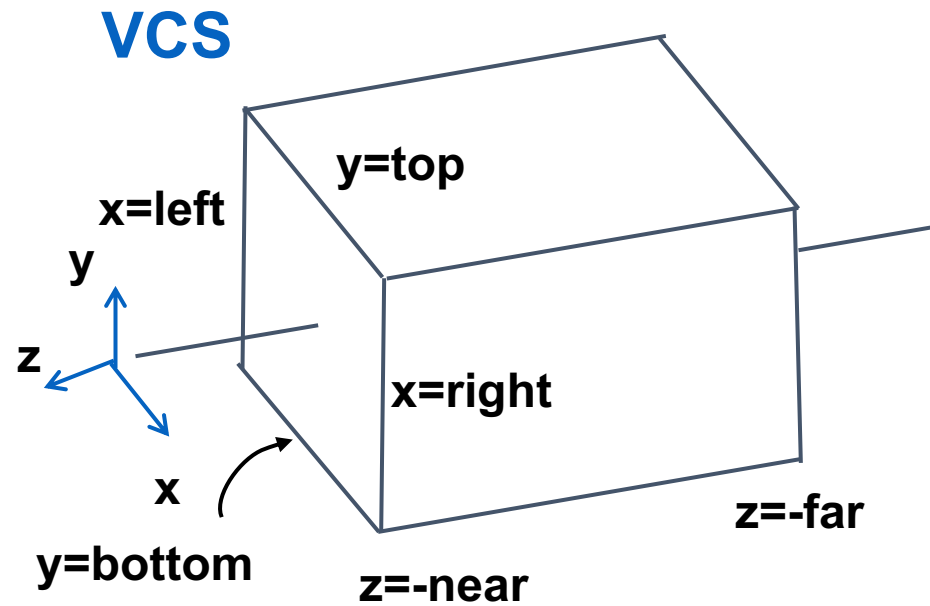
$$P' = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bot} & 0 & -\frac{top + bot}{top - bot} \\ 0 & 0 & \frac{-2}{far - near} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix} P$$

TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, déplacer, refléter

$$P' = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bot} & 0 & -\frac{top + bot}{top - bot} \\ 0 & 0 & \frac{-2}{far - near} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix} P$$

QU'EST-CE QUI CHANGE SI LA CAMÉRA
EST DÉPLACÉE EN ARRIÈRE?



PROJECTION ORTHOGRAPHIQUE

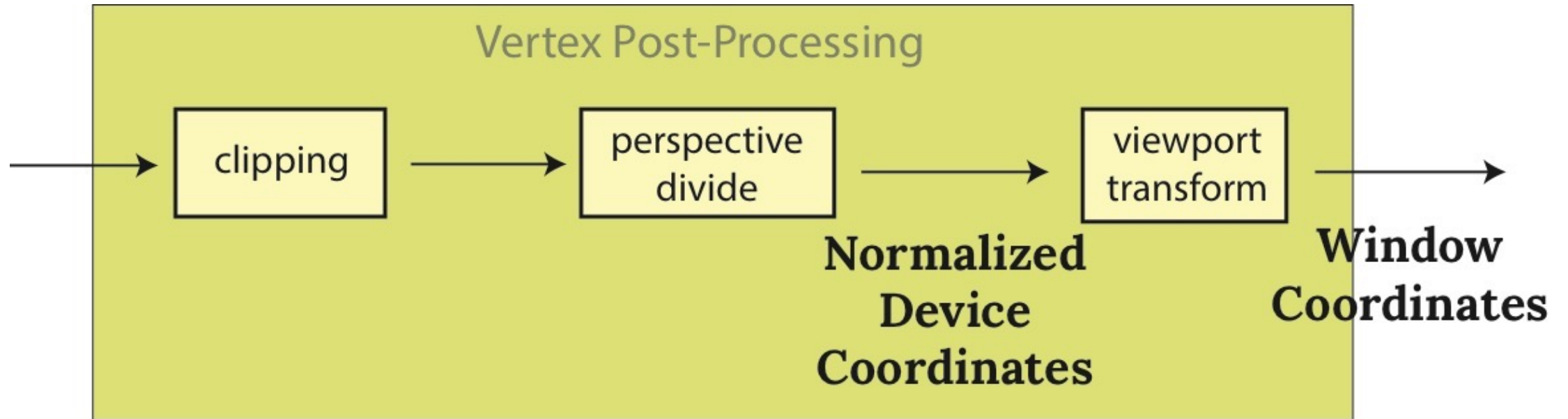
A silhouette of a person holding a large telephoto lens against a clear blue sky. The person's head and shoulders are visible on the right, and the lens extends horizontally across the frame to the left. The lighting is bright, creating a high-contrast silhouette effect.

= prendre une photo de très loin avec un bon zoom

THREE.JS

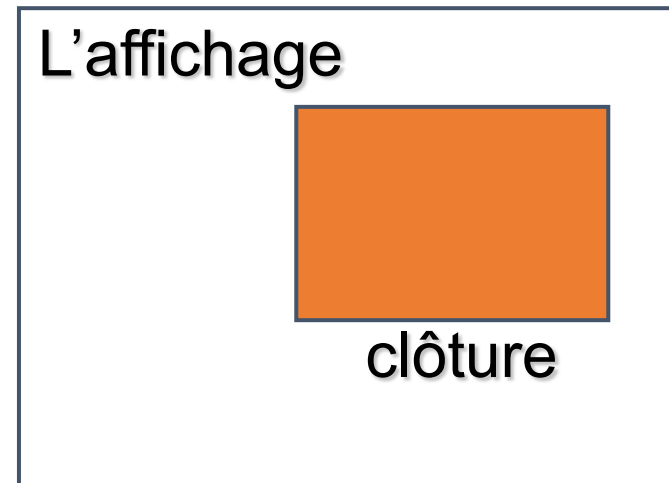
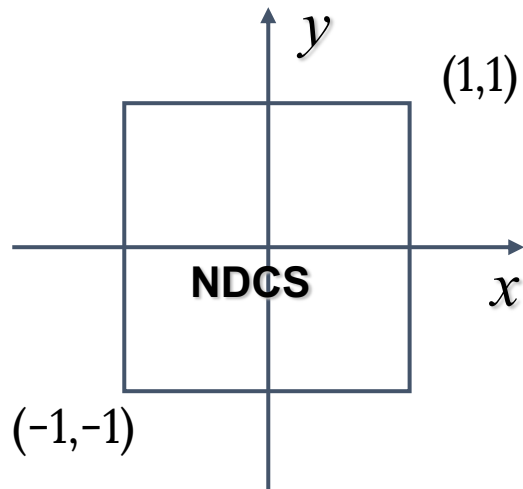
```
var camera = new THREE.OrthographicCamera( width / - 2, width / 2,  
height / 2, height / - 2, 1, 1000 );  
scene.add( camera );
```

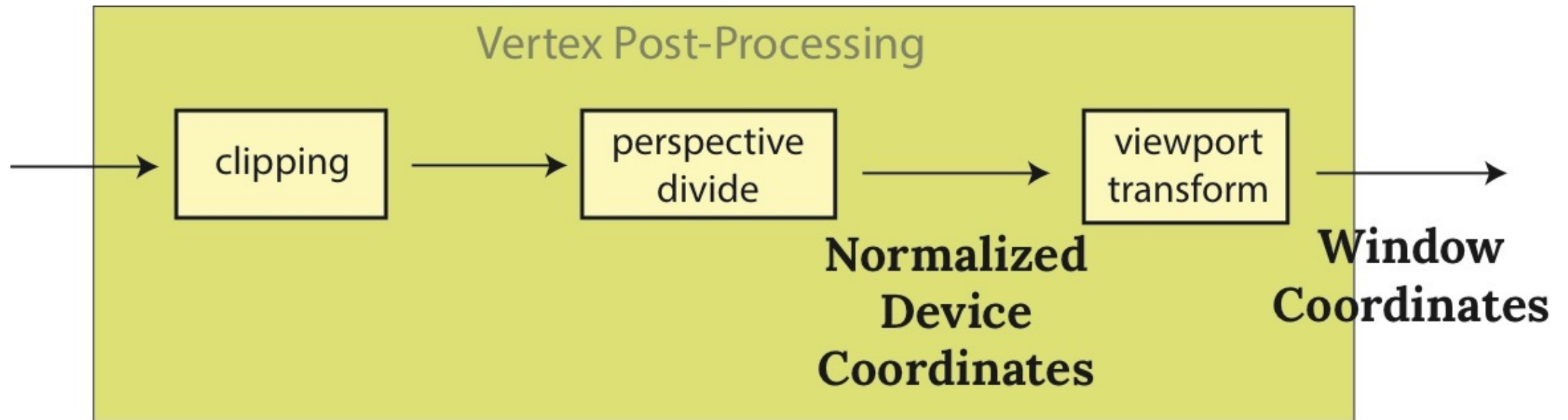
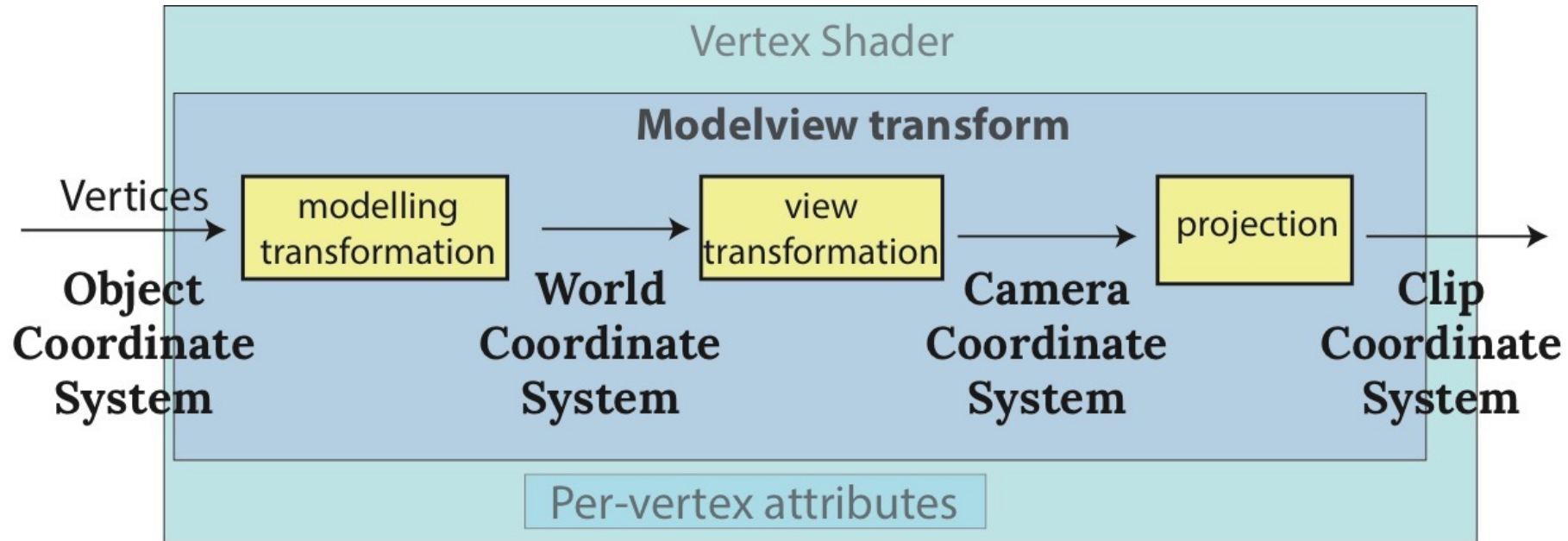
VIEWPORT TRANSFORM



(AUTO) TRANSFORMATION DE LA CLÔTURE (VIEWPORT)

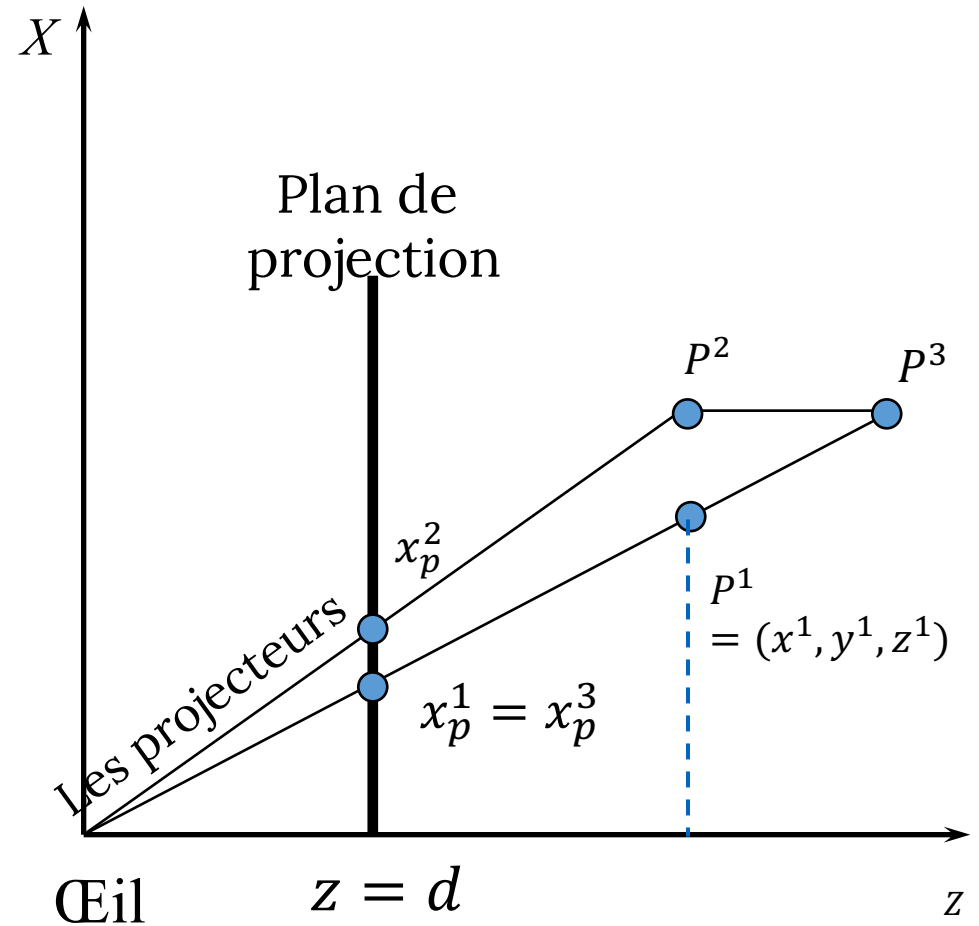
- Convertir les coordonnées de l'appareil normalisées (NDC) en coordonnées de la fenêtre
- I.e. ça:





STÉNOPÉ (LA PERSPECTIVE)

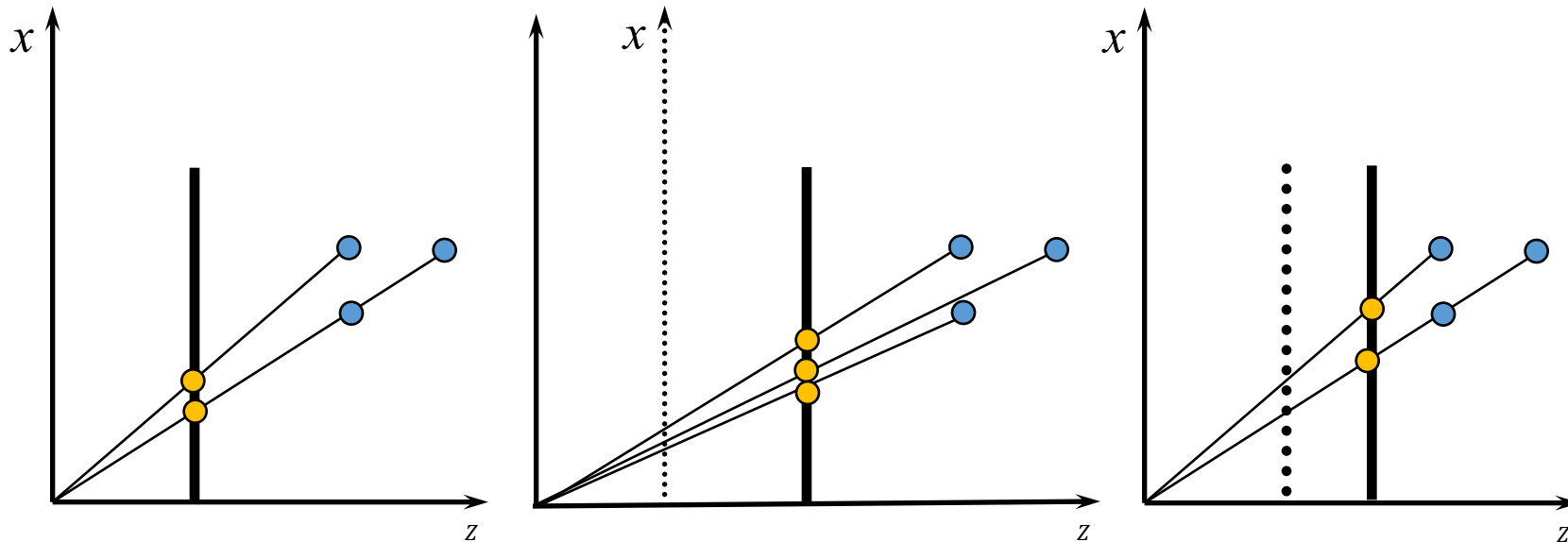
- Une vue d'un **point à distance finie**
- Sans perte de généralité :
 - L'œil à l'origine
 - Le plan de projection est $z = d$
- Pour $P = (x, y, z)$ la similitude des triangles donne:



$$\frac{x}{z} = \frac{x_p}{d} \text{ et } \frac{y}{z} = \frac{y_p}{d} \Rightarrow x_p = \frac{x}{z/d} \text{ et } y_p = \frac{y}{z/d}$$

STÉNOPÉ (LA PERSPECTIVE)

- Quelle est la différence entre:
 - Déplacer le plan de projection
 - Déplacer l'œil (le centre de projection)?



PROJECTION PERSPECTIVE

- En coordonnées homogènes:

$$P(x, y, z, 1) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \\ z/d \end{pmatrix}$$

- En coordonnées euclidiennes:

$$\left(\frac{x}{z/d}, \frac{y}{z/d}, \frac{z}{z/d} \right) = \left(\frac{x}{z/d}, \frac{y}{z/d}, d \right) = (x_p, y_p, d).$$

- P est singulier: $\det(P) = 0$

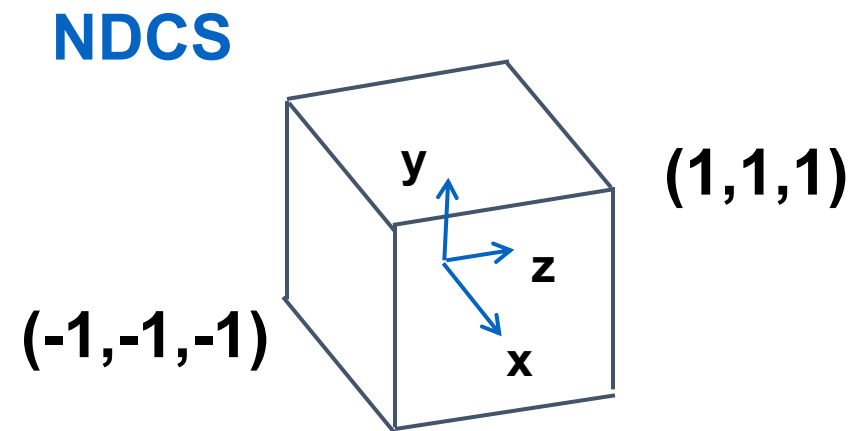
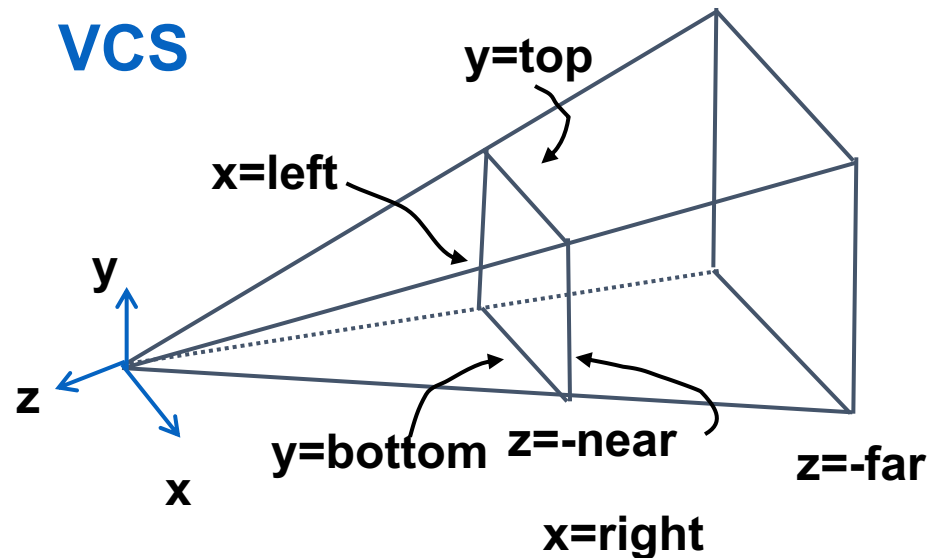
UNE MATRICE DE PROJECTION SINGULIÈRE

- On ne peut pas l'inverser!
- On ne peut pas faire le test de profondeur
 - Tous les z sont les mêmes
- Similaire à notre première matrice de projection orthographique:

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

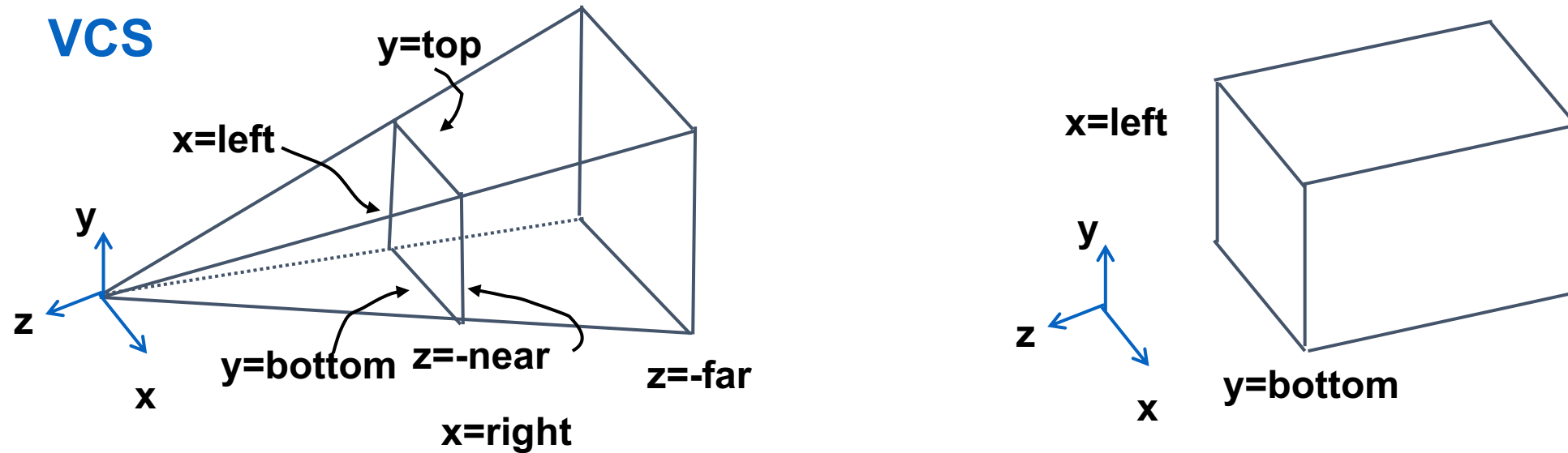
LA MATRICE DE LA PROJECTION PERSPECTIVE OPENGL

Mapper (faire correspondre) la pyramide tronquée à un cube NDCS. Après, z sera ignorée. On doit changer l'échelle/déplacer/faire cisaillement → une transformation générique



LA PYRAMIDE TRONQUÉE DEVIENT UNE BOÎTE

Poursuit comme avec la projection orthographique



Avant:

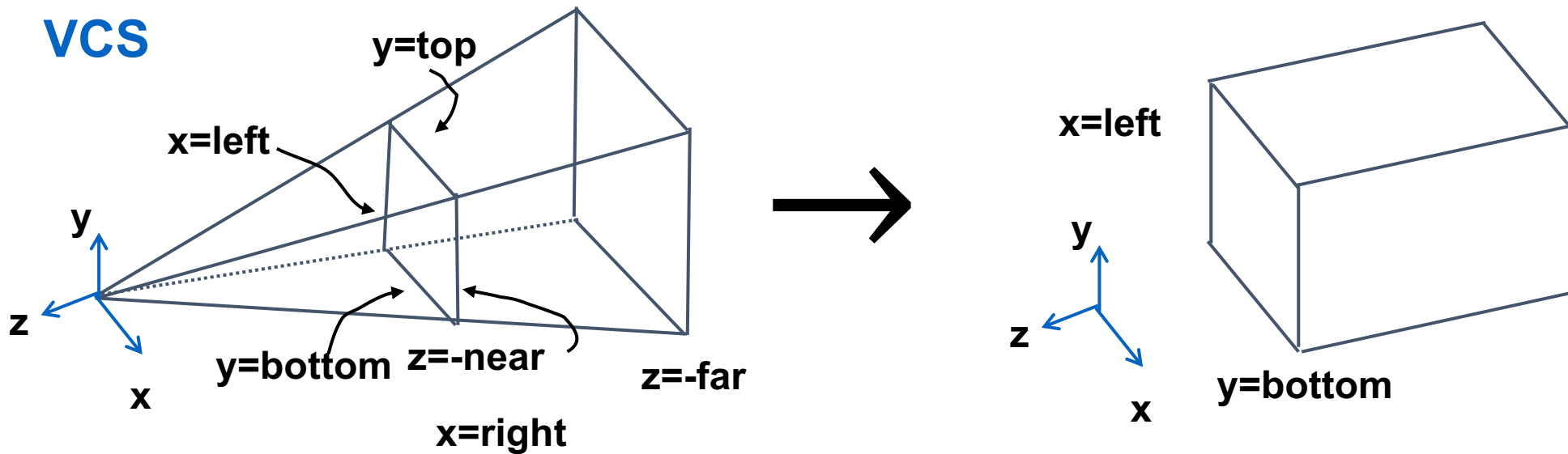
TRANSFORMATIONS POUR UNE PROJECTION PARALLÈLE

- Changer d'échelle, déplacer, refléter

$$P' = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bot} & 0 & -\frac{top + bot}{top - bot} \\ 0 & 0 & \frac{-2}{far - near} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix} P$$

LA PYRAMIDE TRONQUÉE DEVIENT UNE BOÎTE

$$M_p = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{n+f}{n} & -f \\ 0 & 0 & \frac{1}{n} & 0 \end{pmatrix}$$



LA STRUCTURE DE LA MATRICE

$$\begin{pmatrix} \blacksquare & 0 & \blacksquare & \blacksquare \\ 0 & \blacksquare & \blacksquare & \blacksquare \\ 0 & 0 & \blacksquare & \blacksquare \\ 0 & 0 & -1 & \blacksquare \end{pmatrix}$$

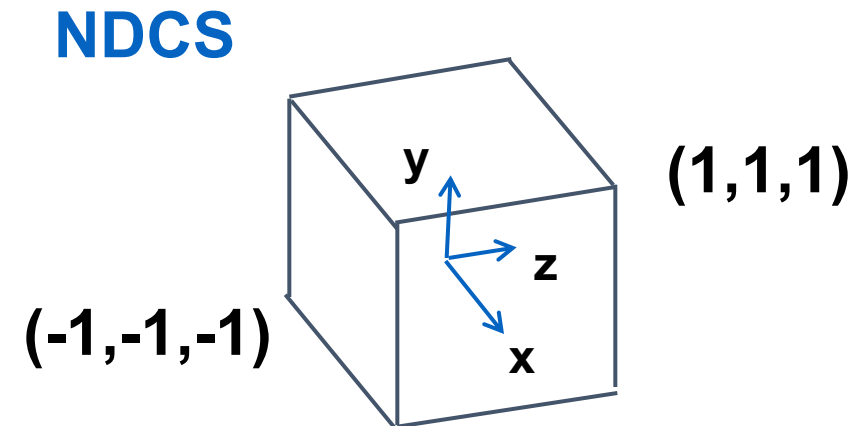
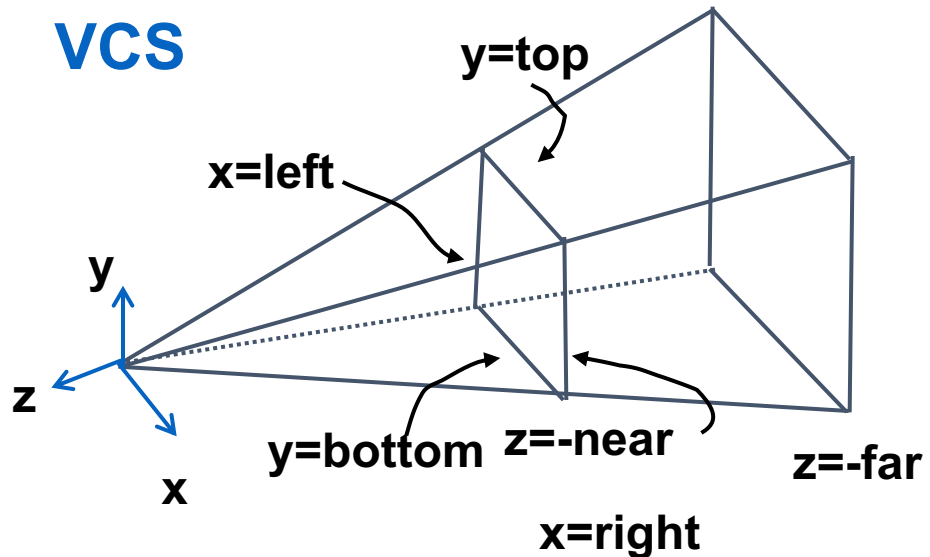
UN SYSTÈME LINÉAIRE

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} E & 0 & A & 0 \\ 0 & F & B & 0 \\ 0 & 0 & C & D \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ w' \end{pmatrix} = \begin{pmatrix} Ex + Az \\ Fy + Bz \\ Cz + D \\ -z \end{pmatrix}$$

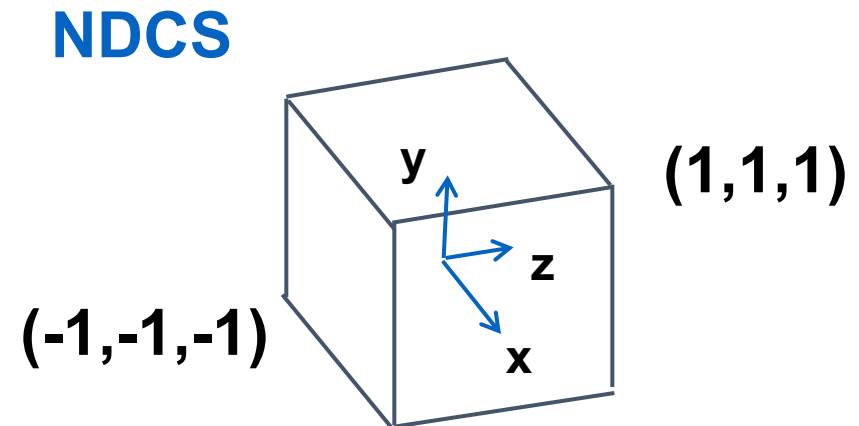
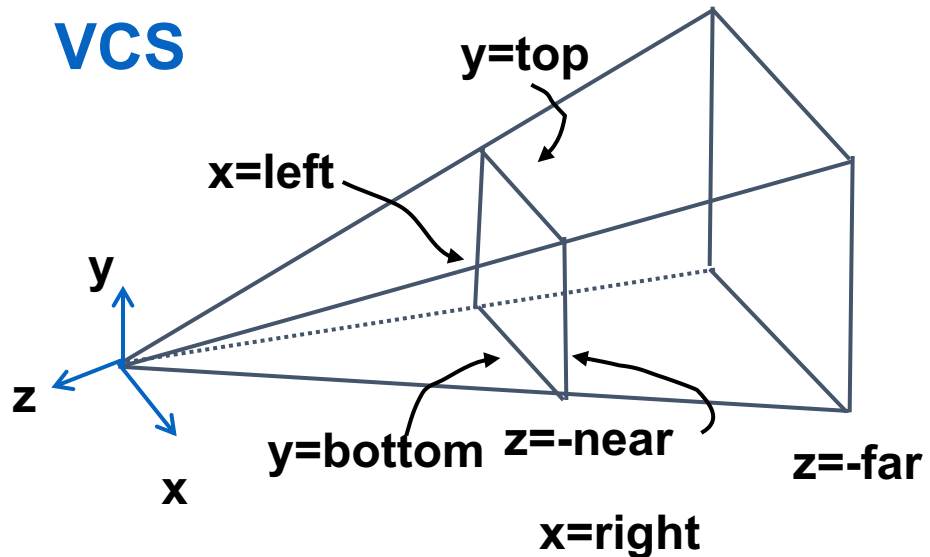
UN SYSTÈME LINÉAIRE

$$\begin{aligned}x = \textit{left} &\rightarrow \frac{x'}{w'} = -1; x = \textit{right} \rightarrow \frac{x'}{w'} = 1 \\y = \textit{top} &\rightarrow \frac{y'}{w'} = 1; y = \textit{bottom} \rightarrow \frac{y'}{w'} = -1 \\z = -\textit{near} &\rightarrow \frac{z'}{w'} = 1; z = -\textit{far} \rightarrow \frac{z'}{w'} = -1\end{aligned}$$



UN SYSTÈME LINÉAIRE

$$\begin{aligned}x = \textit{left} &\rightarrow \frac{x'}{w'} = -1; x = \textit{right} \rightarrow \frac{x'}{w'} = 1 \\y = \textit{top} &\rightarrow \frac{y'}{w'} = 1; y = \textit{bottom} \rightarrow \frac{y'}{w'} = -1 \\z = \textit{-near} &\rightarrow \frac{z'}{w'} = 1; z = \textit{-far} \rightarrow \frac{z'}{w'} = -1\end{aligned}$$



UN SYSTÈME LINÉAIRE

$$\begin{pmatrix} x' \\ y' \\ z' \\ w' \end{pmatrix} = \begin{pmatrix} Ex + Az \\ Fy + Bz \\ Cz + D \\ -z \end{pmatrix}$$

$$y = \text{top} \rightarrow \frac{y'}{w'} = 1$$

$$\frac{y'}{w'} = \frac{Fy + Bz}{w'},$$
$$1 = F \frac{y}{-z} + B \frac{z}{-z},$$

$$1 = \frac{Fy + Bz}{w'},$$
$$1 = F \frac{y}{-z} - B,$$

$$1 = \frac{Fy + Bz}{-z},$$
$$1 = F \frac{\text{top}}{-(-\text{near})} - B,$$

$$1 = F \frac{\text{top}}{\text{near}} - B$$

L'EXEMPLE DE LA PERSPECTIVE

le volume de vue

- left = -1, right = 1
- bottom = -1, top = 1
 - near = 1, far = 4

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -5/3 & -8/3 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

L'EXEMPLE DE LA PERSPECTIVE

$$\begin{bmatrix} 1 \\ -1 \\ -5z_{VCS}/3 - 8/3 \\ -z_{VCS} \end{bmatrix} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & -5/3 & -8/3 \\ & & & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ z_{VCS} \\ 1 \end{bmatrix}$$

$/w$

$$\begin{aligned} x_{NDCS} &= -1/z_{VCS} \\ y_{NDCS} &= 1/z_{VCS} \\ z_{NDCS} &= \frac{5}{3} + \frac{8}{3z_{VCS}} \end{aligned}$$

PROJECTION PERSPECTIVE

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