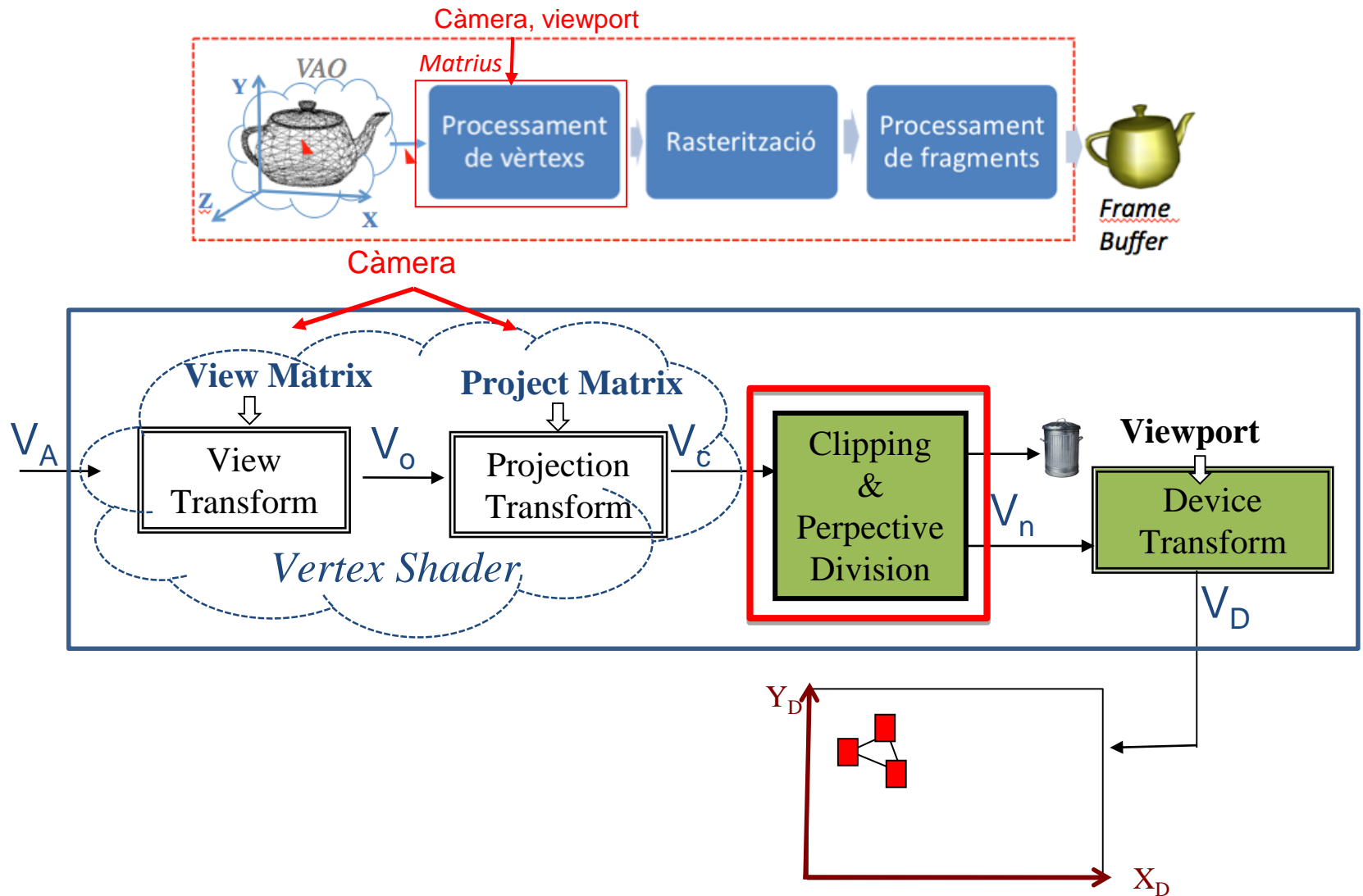


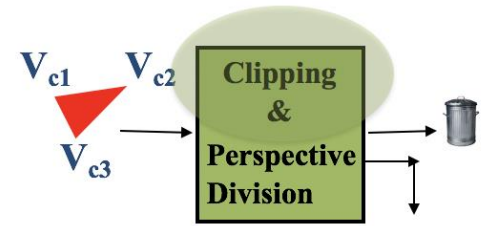
Classe 4: Procés de Visualització

- **Processament dels vèrtexs**
 - **Retallat**
 - **Divisió perspectiva**
 - **Pas a coordenades de dispositiu**
- **Rasterització**
- **Processament dels fragments: el fragment shader**

Processament de vèrtexs



Clipping



Condió per a que un Vèrtex sigui interior al volum de visió:

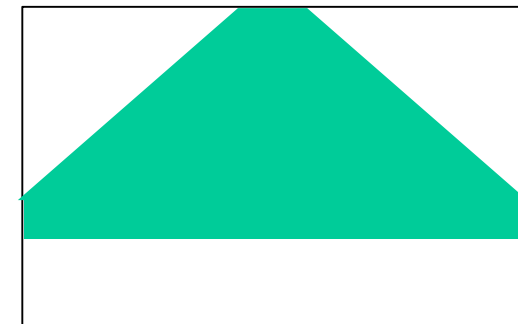
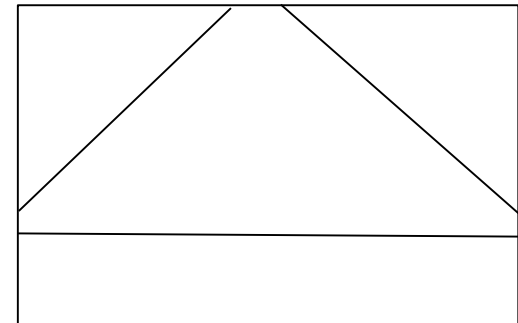
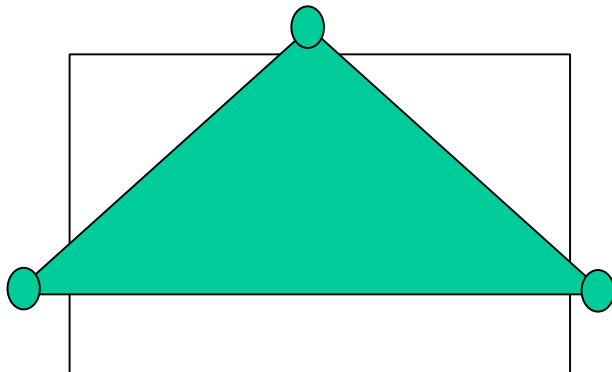
$$-w_c \leq x_c \leq w_c$$

$$-w_c \leq y_c \leq w_c$$

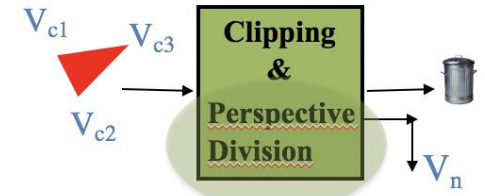
$$-w_c \leq z_c \leq w_c$$

$V_c = (x_c, y_c, z_c, w_c)$ on $w_c = 1$ en ortogonal

$V_c = (x_c, y_c, z_c, w_c)$ on $w_c = -z_o$ en perspectiva



Projecció: Ortogonal



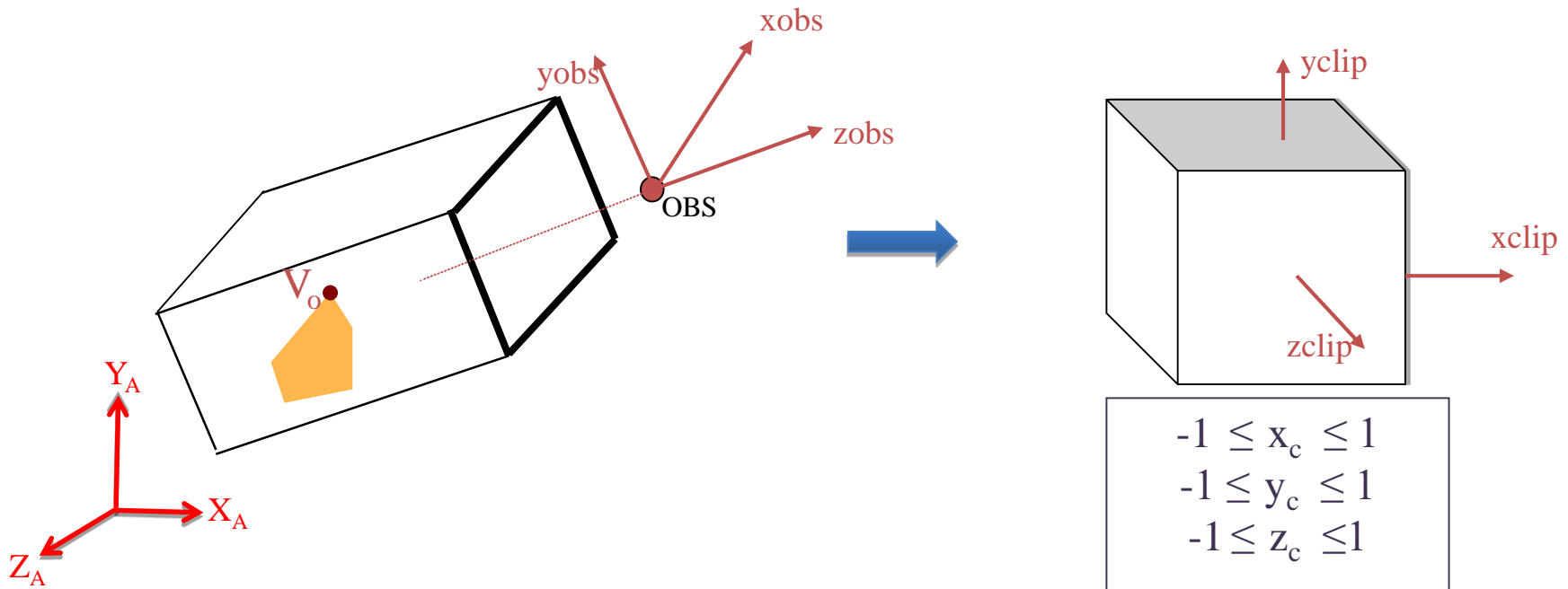
$$PM = \begin{pmatrix} a & 0 & 0 & e \\ 0 & b & 0 & f \\ 0 & 0 & c & d \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$a = 2/(r-l) \quad e = (r+l)/(r-l)$$

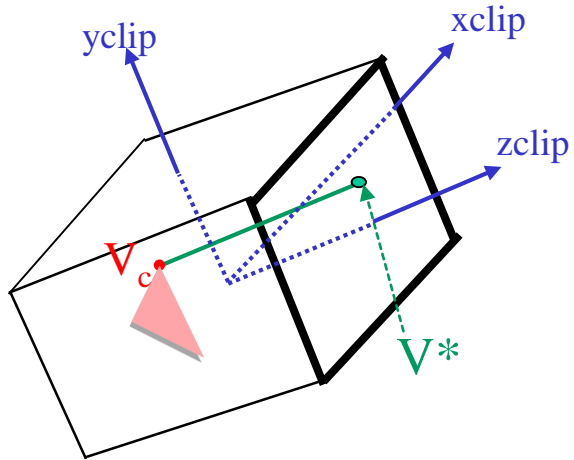
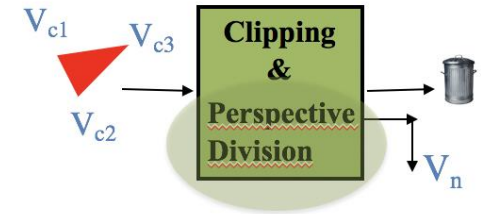
$$b = 2/(t-b) \quad f = (t+b)/(t-b)$$

$$c = 2/(z_f - z_n) \quad d = (z_n + z_f)/(z_f - z_n)$$

$$V_c = (x_c, y_c, z_c, w_c) \text{ on } w_c = 1$$



Projecció: Ortogonal



$V_c = (x_c, y_c, z_c, w_c)$ on $w_c=1$

$$\begin{aligned} -1 &\leq x_c \leq 1 \\ -1 &\leq y_c \leq 1 \\ -1 &\leq z_c \leq 1 \end{aligned}$$

Vèrtex projectat:

$$V_x^* = V_{cx} \quad V_y^* = V_{cy}$$

$$V^* = V_c / w_c = V_n$$

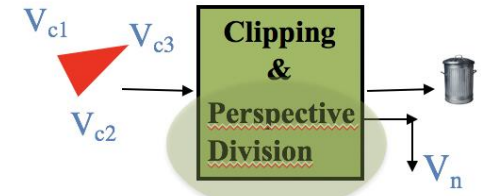
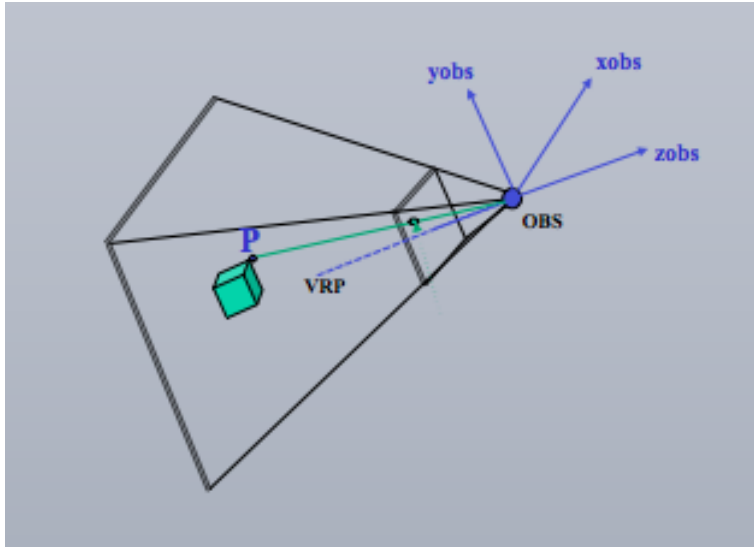
$$-1 \leq x_n \leq 1$$

$$-1 \leq y_n \leq 1$$

$$-1 \leq z_n \leq 1$$

V_z^* per càlculs posteriors i
indica distància a window

Projecció: Perspectiva



$$PM = \begin{pmatrix} 1/ra*a & 0 & 0 & 0 \\ 0 & 1/a & 0 & 0 \\ 0 & 0 & c & d \\ 0 & 0 & -1 & 0 \end{pmatrix} \quad \begin{aligned} a &= \tan(\text{FOV}/2) \\ c &= (zf+zn)/(zn-zf) \\ d &= 2*zn*zf/(zn-zf) \end{aligned}$$

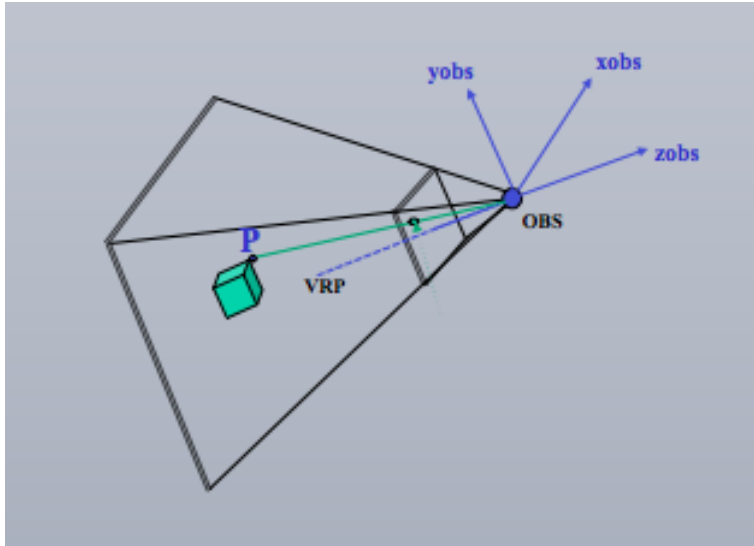
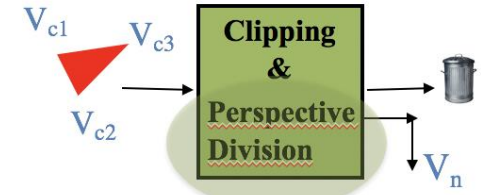
$$V_c = (x_c, y_c, z_c, w_c) \text{ on } w_c = -z_o$$

$$-w_c \leq x_c \leq w_c$$

$$-w_c \leq y_c \leq w_c$$

$$-w_c \leq z_c \leq w_c$$

Projecció: Perspectiva

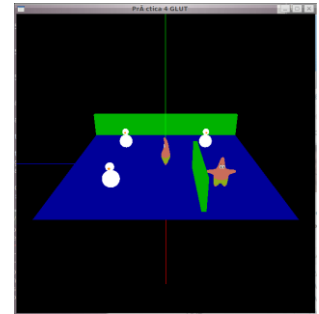


Vèrtex projectat:

$$\mathbf{V}^* = \mathbf{V}_c / w_c = -\mathbf{V}_c / z_o$$

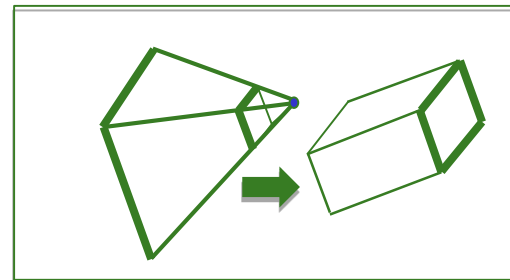
$$x^* = -x_c / z_o \quad y^* = -y_c / z_o \quad z^* = -z_c / z_o$$

Inversament proporcional a distància a observador 😊



$$\mathbf{V}_c = (x_c, y_c, z_c, w_c) \text{ on } w_c = -z_o$$

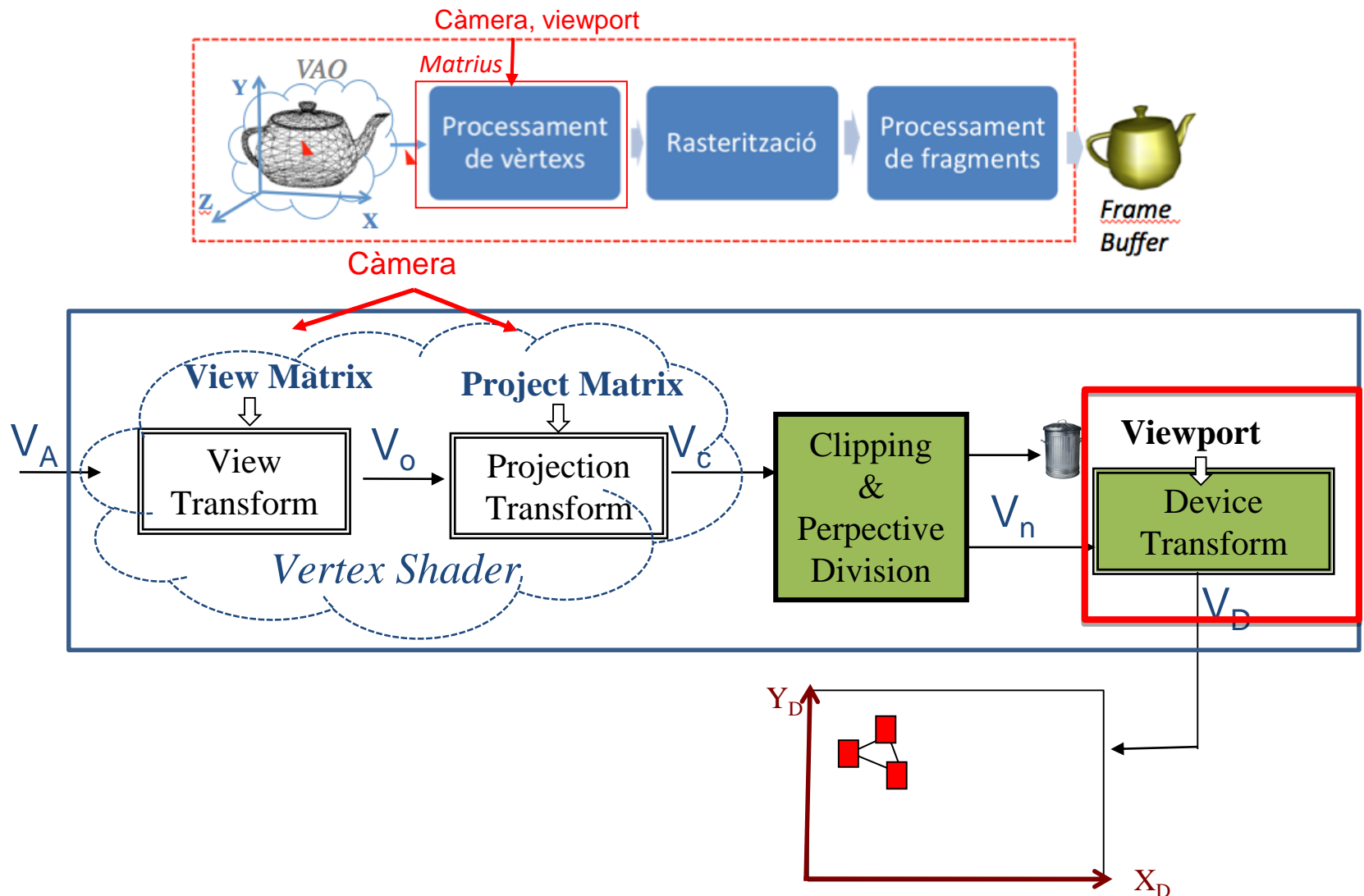
$$\begin{aligned} -w_c &\leq x_c \leq w_c \\ -w_c &\leq y_c \leq w_c \\ -w_c &\leq z_c \leq w_c \end{aligned}$$



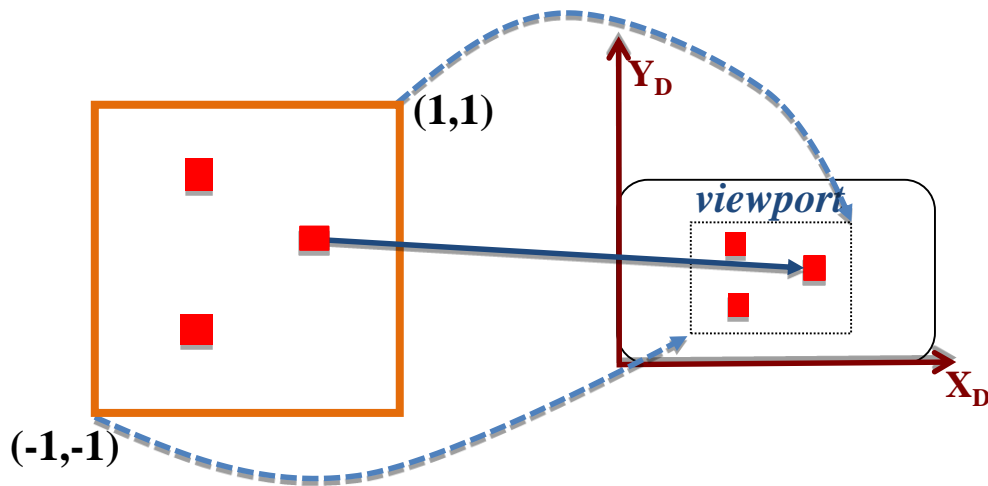
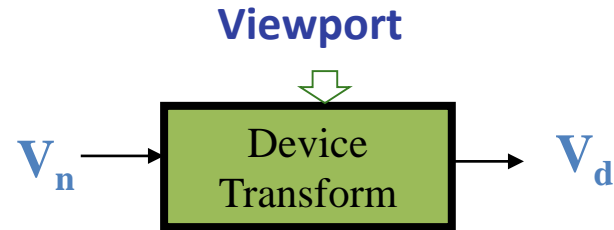
$$\begin{aligned} -1 &\leq x_n \leq 1 \\ -1 &\leq y_n \leq 1 \\ -1 &\leq z_n \leq 1 \end{aligned}$$

$$\mathbf{V}^* = \mathbf{V}_c / w_c \rightarrow \mathbf{V}_n$$

Transformació món-dispositiu



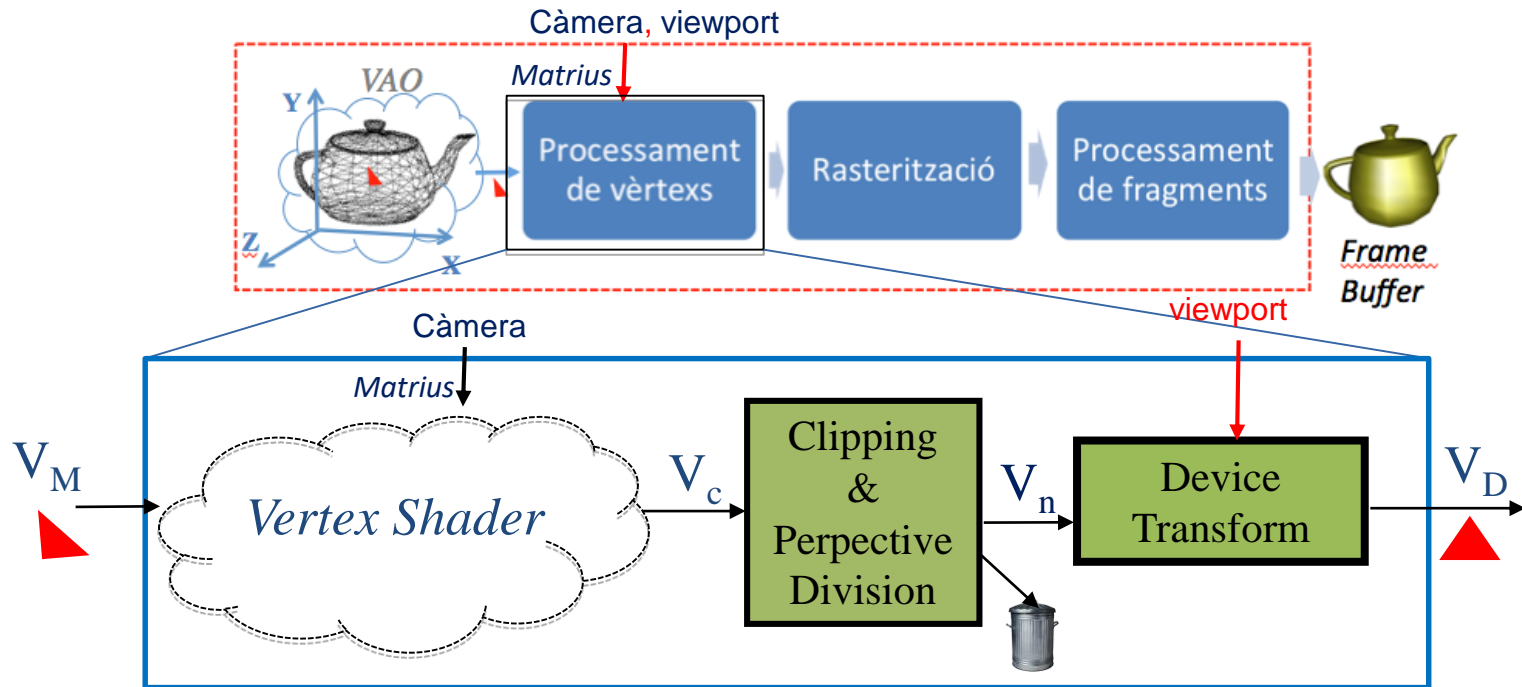
Transformació món-dispositiu



Depth range transformation

- $V_n.z \ [-1,1] \rightarrow [0,1]$, per defecte.

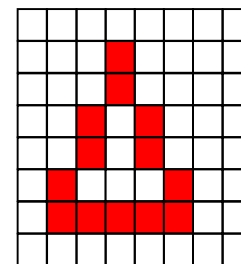
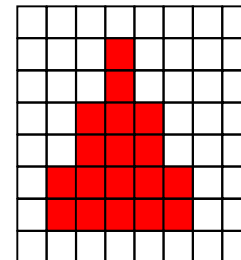
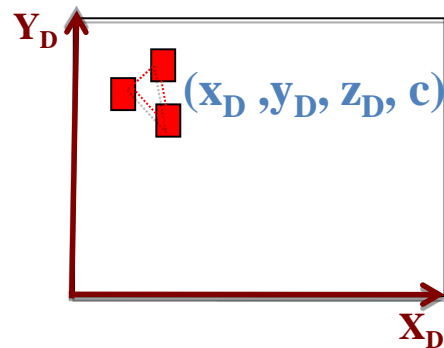
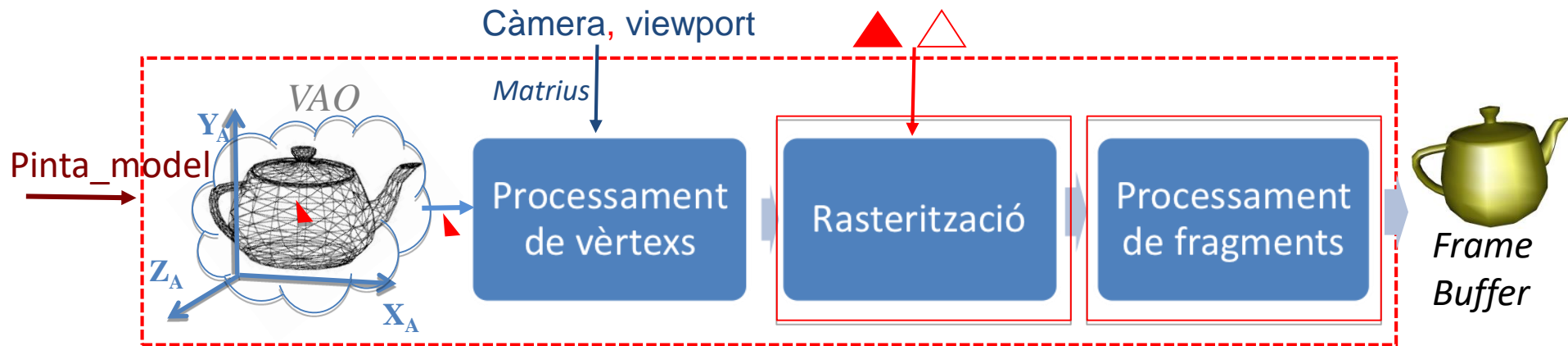
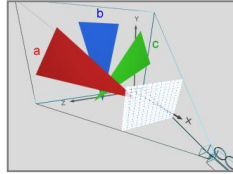
Processament de vèrtexs: resum



Classe 4: Procés de Visualització

- Processament dels vèrtexs
 - Retallat
 - Divisió perspectiva
 - Pas a coordenades de dispositiu
- **Rasterització**
- **Processament dels fragments: el fragment shader**

Procés de visualització

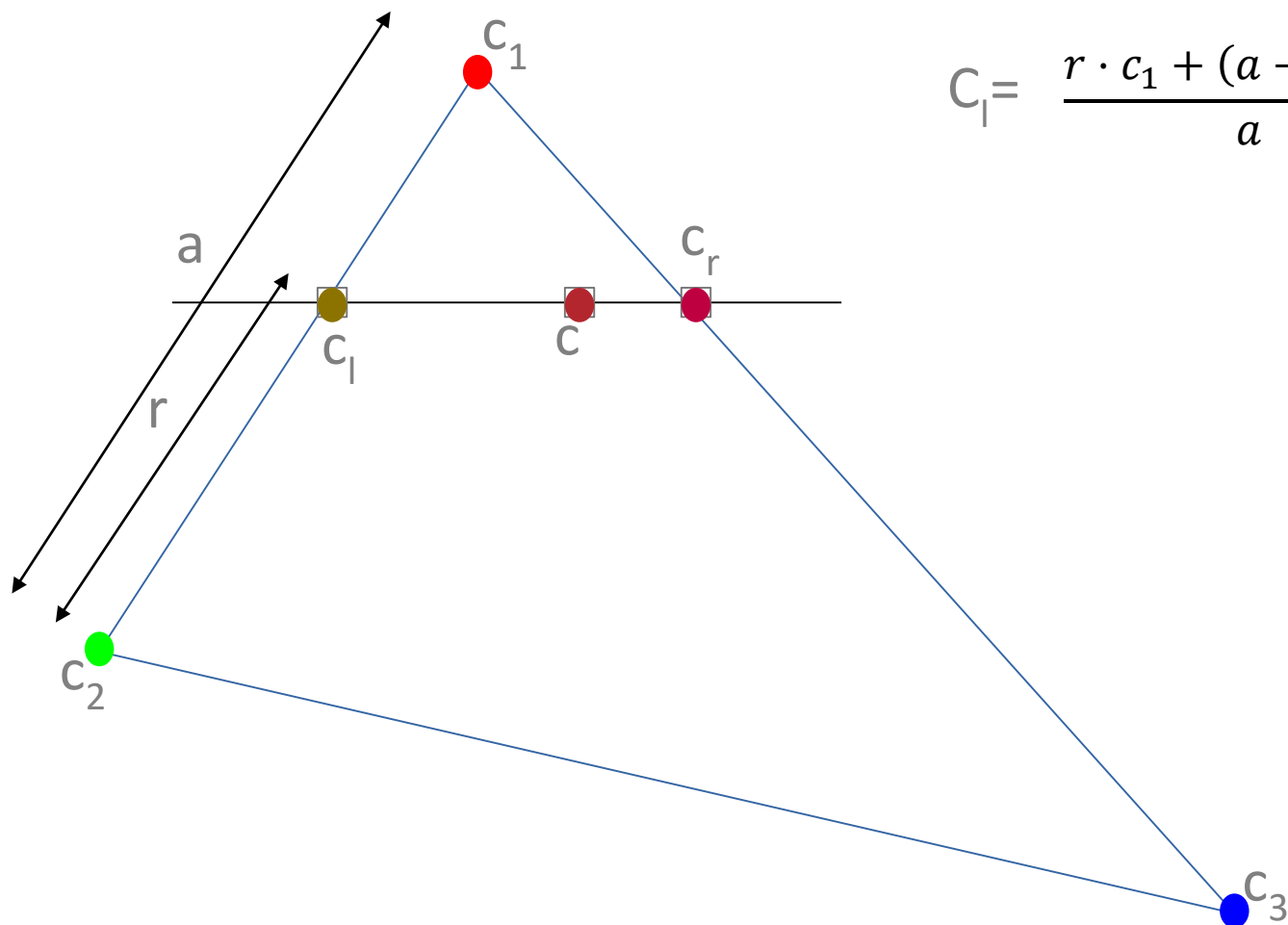
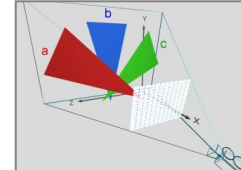


Fragments: $\{(x_D, y_D, z_D, c)_f\}$



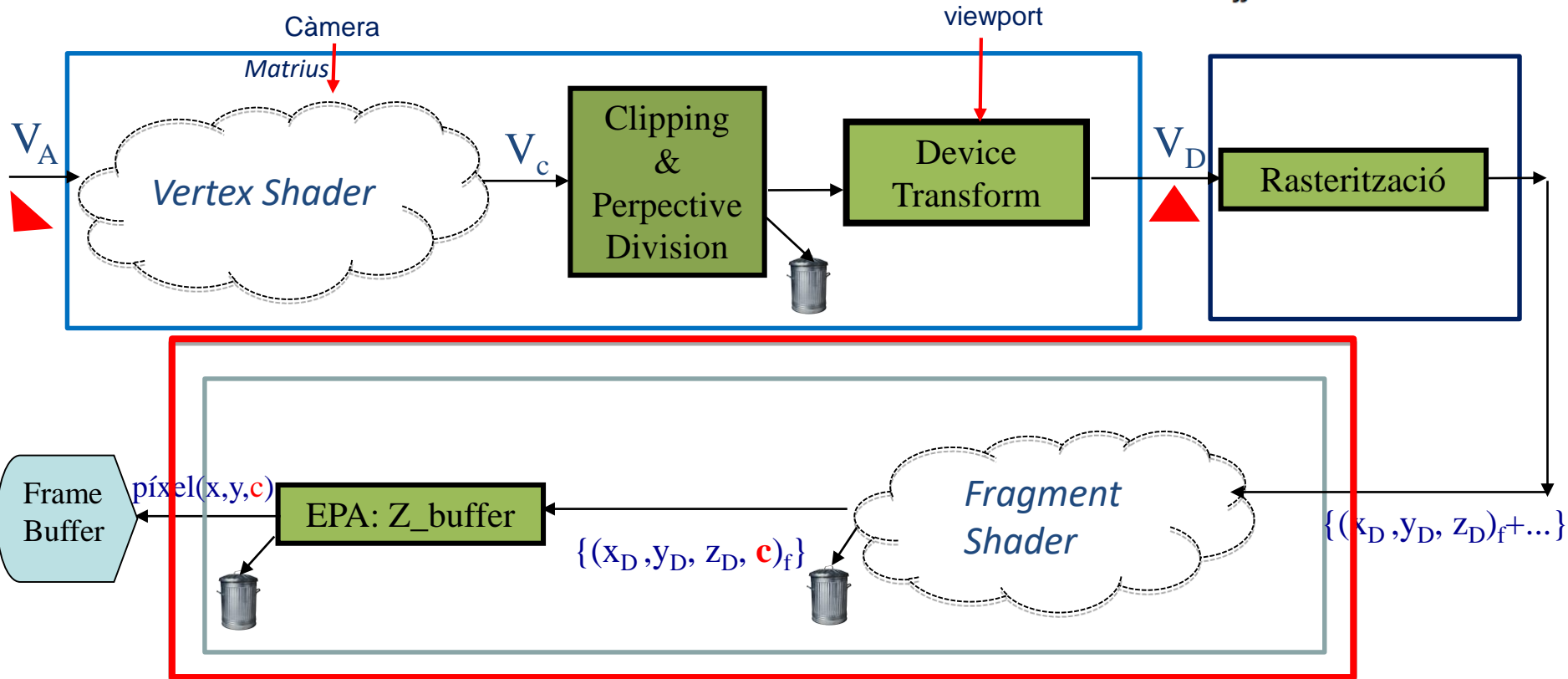
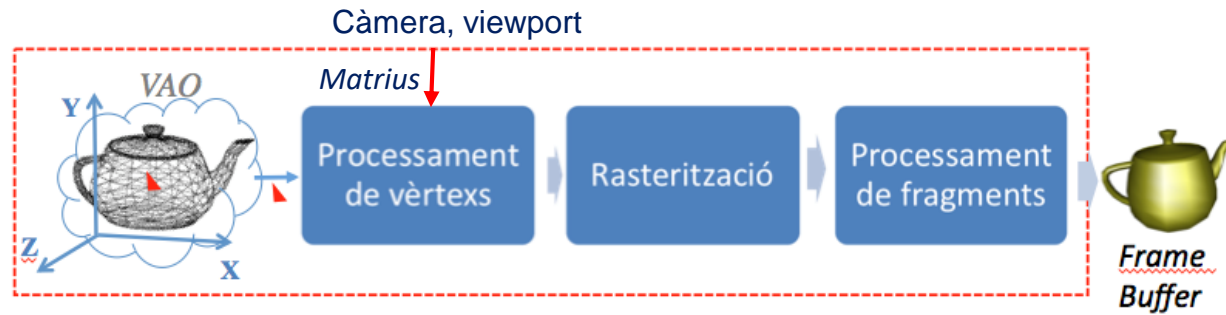
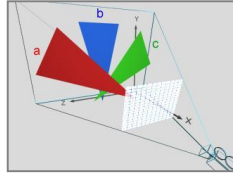
píxel(x,y,c)

Rasterització: interpolació



$$c_r = \frac{r \cdot c_1 + (a - r) \cdot c_2}{a}$$

Processament de fragments



Processament de fragments: El fragment Shader

Fragment Shader

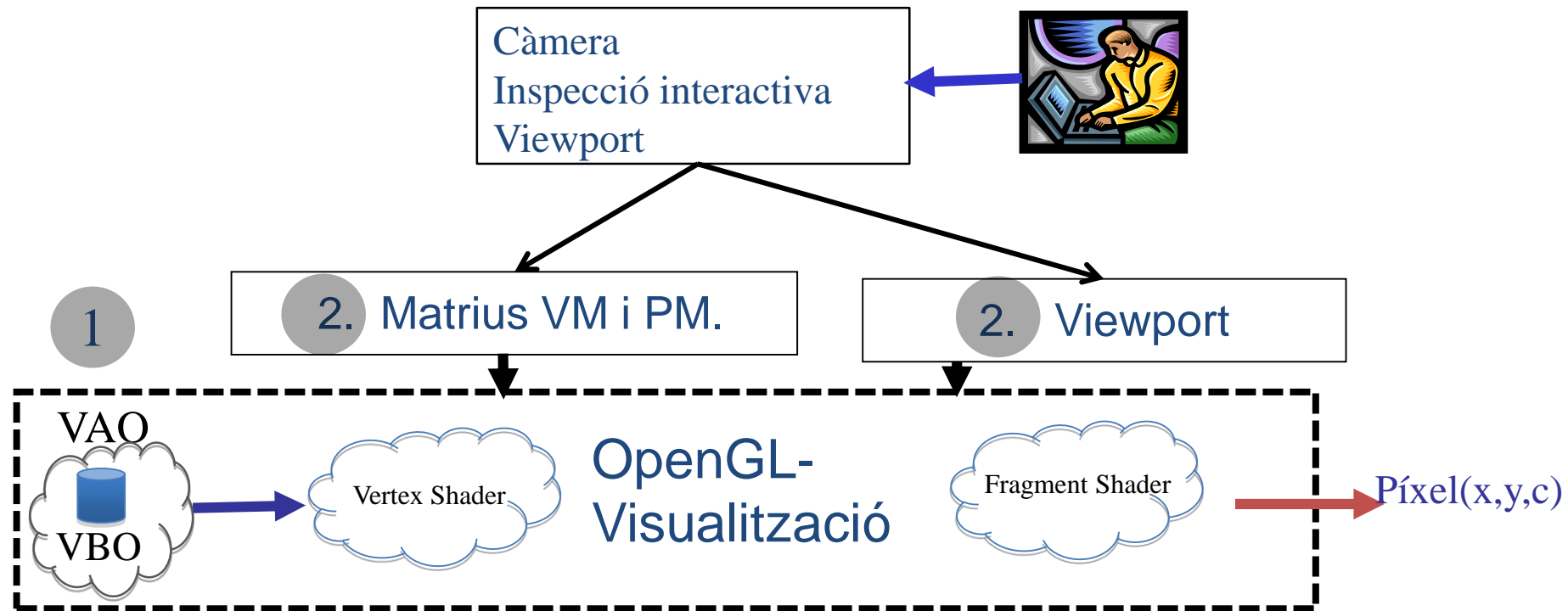
```
#version 330 core

in ...

out vec4 FragColor;

void main() {
    FragColor = vec4(0, 0, 0, 1);
}
```

Pintar/visualitzar en OpenGL 3.3 (resum)



3. Pinta_Model()

// Activa VAO i crida a glDrawArrays(...)

Classe 4: Conceptes i preguntes

- El procés de visualització projectiu: blocs funcionals que l'integren, ordre dels processos, sistemes de coordenades.
- Diferència entre vèrtex i fragment.
- Què cal fer, com a mínim, en el fragment shader?
- Què són les coordenades normalitzades?
- Què és i com funciona el retallat? Per què cal?
- Com i quan es passa a coordenades de dispositiu? Què són aquestes coordenades exactament?
- Què passa amb els `out` addicionals que puguem afegir al vertex shader? Com arriba aquesta informació al fragment shader?