Waterfall, MC Escher

CC3501 Vistas y Proyecciones

Eduardo Graells-Garrido
Otoño 2023

Proyección

¿Qué es una vista en perspectiva?

Esta imagen de la Patagonia nos ayuda a definir al menos dos de sus atributos:

Las líneas paralelas convergen hacia un punto en el horizonte.

Los objetos se ven más pequeños a medida que están más lejos.

La perspectiva no siempre se trabajó correctamente.

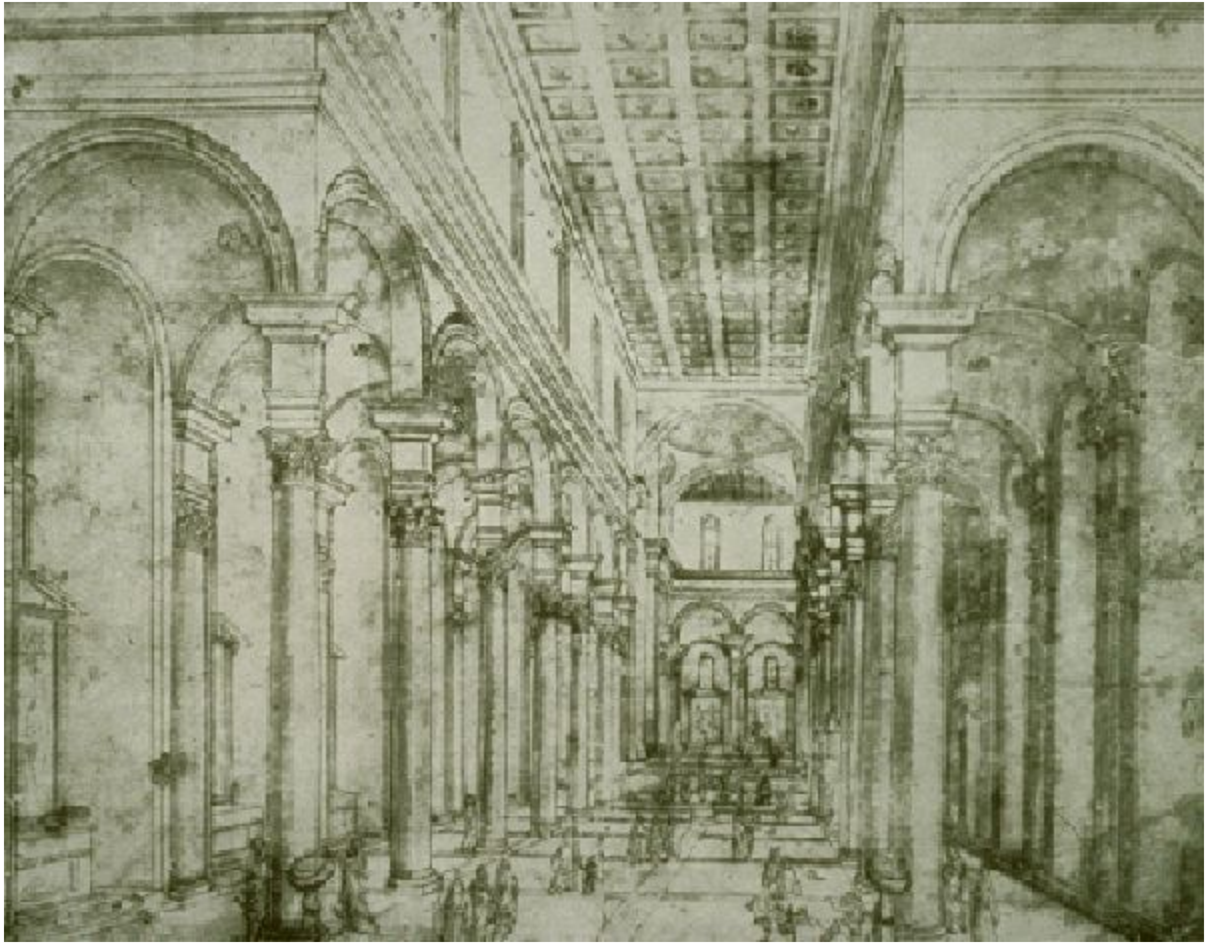
Una muestra de ello es el arte del pasado.

Lorsch Gospels 778–820. Charlemagne's Court School

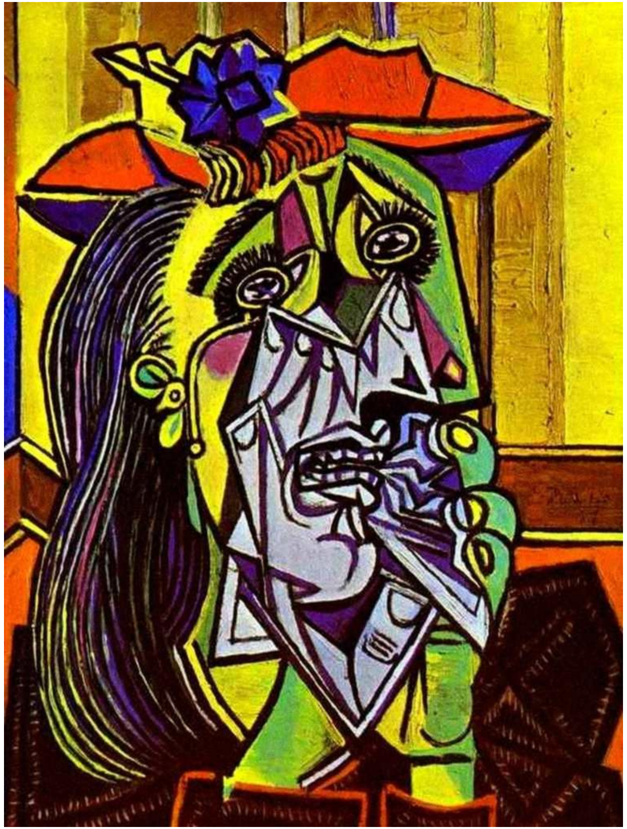
L'Annunciazione de Ambrogio Lorenzetti, 1344.

Filippo Brunelleschi fue un arquitecto considerado creador de la técnica de la perspectiva.

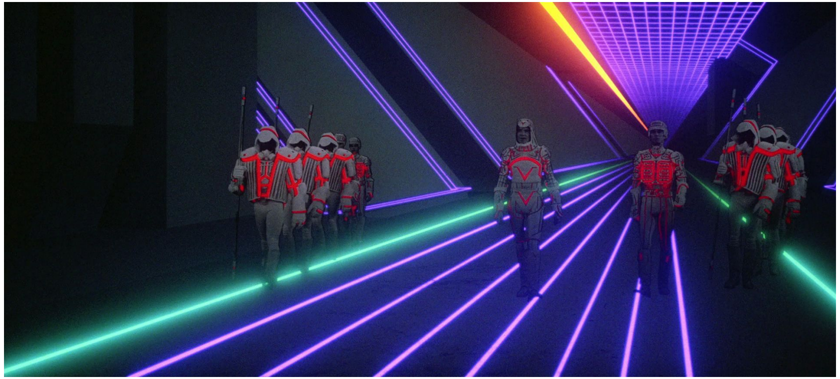
Hizo iglesias y otros establecimientos en Florencia. La del Espíritu Santo es una de ellas. El diagrama es de 1428. Su construcción empezó en 1444 y terminó en 1487.

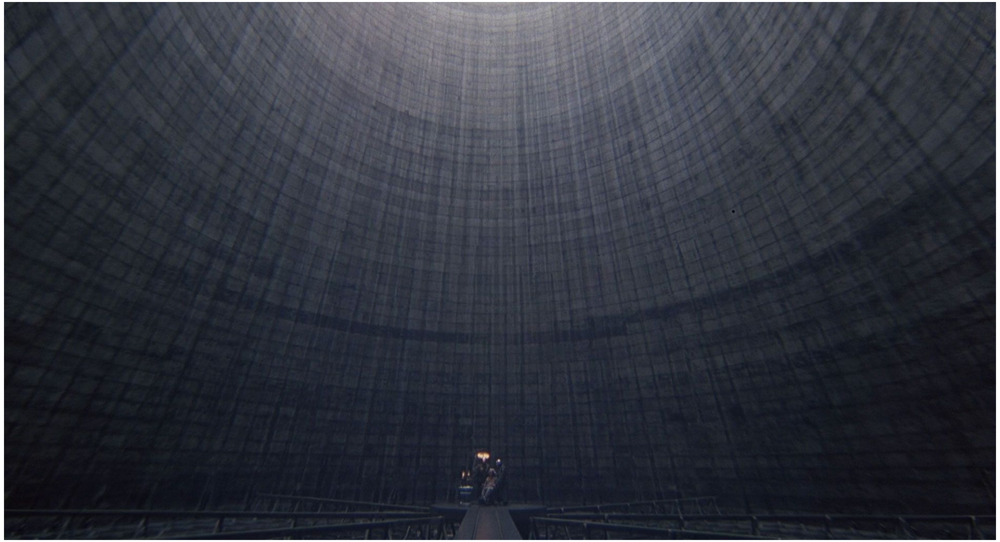


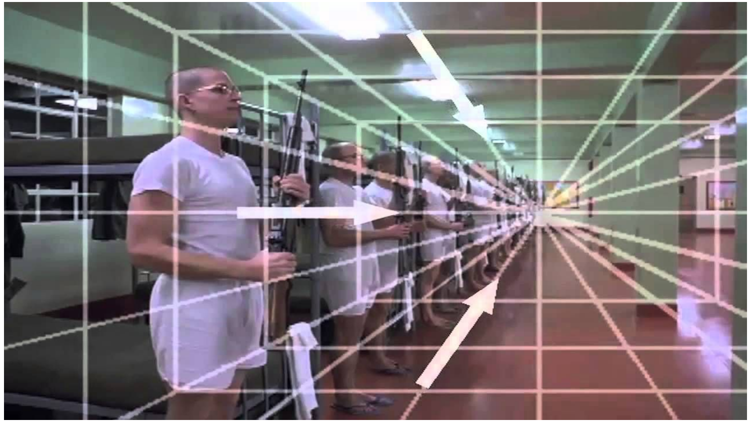
Masaccio, c. 1425.



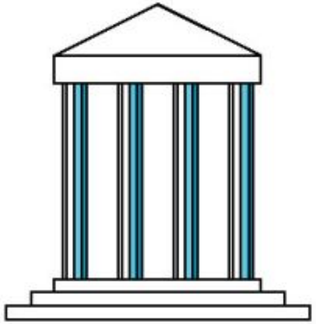
Para Picasso la perspectiva era tan manipulable como los otros elementos de la imagen.



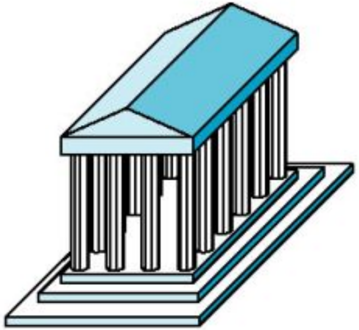




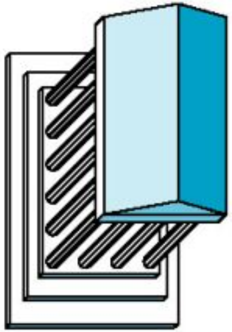
El cine hace uso de la perspectiva para aumentar el dramatismo visual de las escenas.



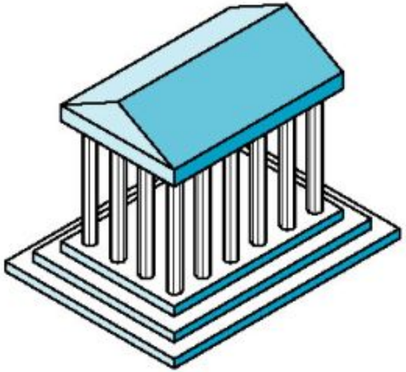
Front elevation



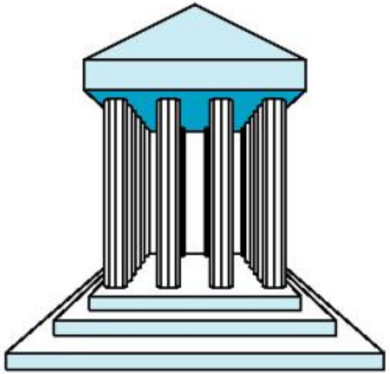
Elevation oblique



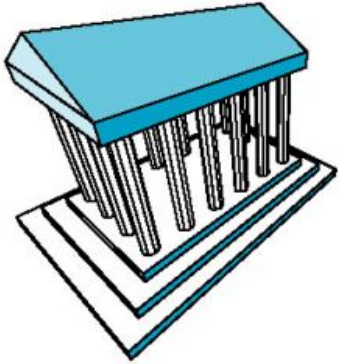
Plan oblique



Isometric

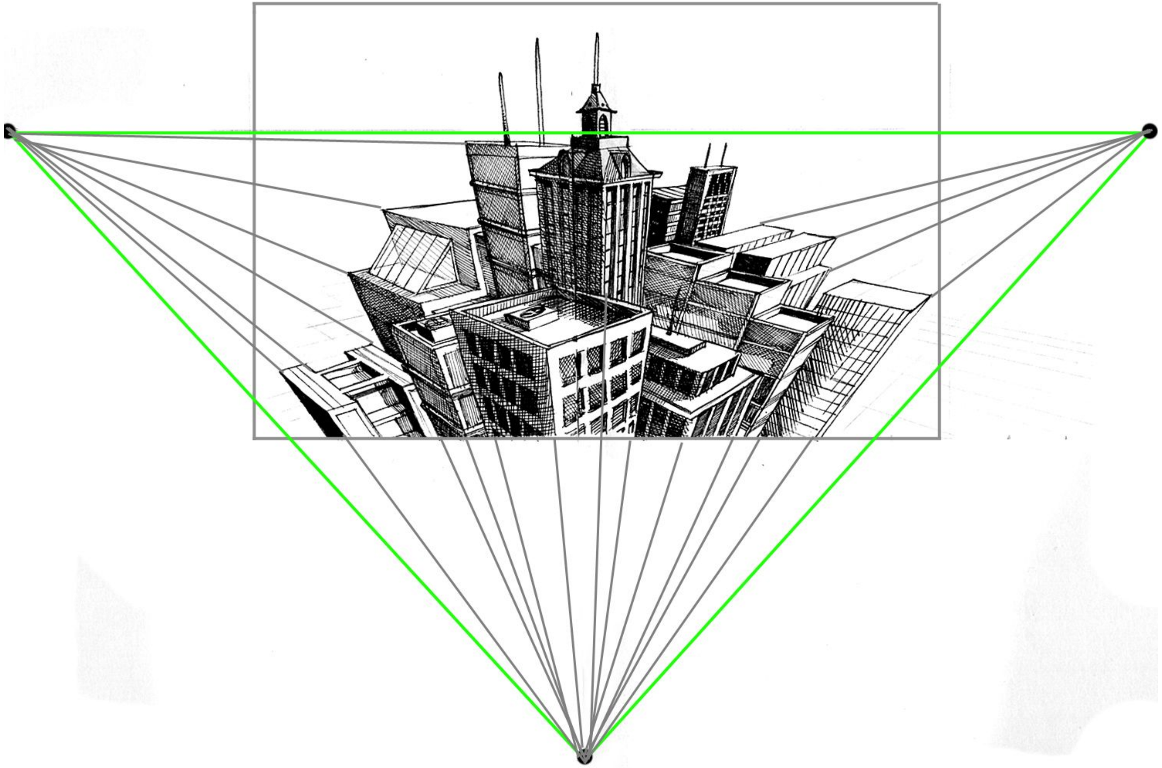


One-point perspective

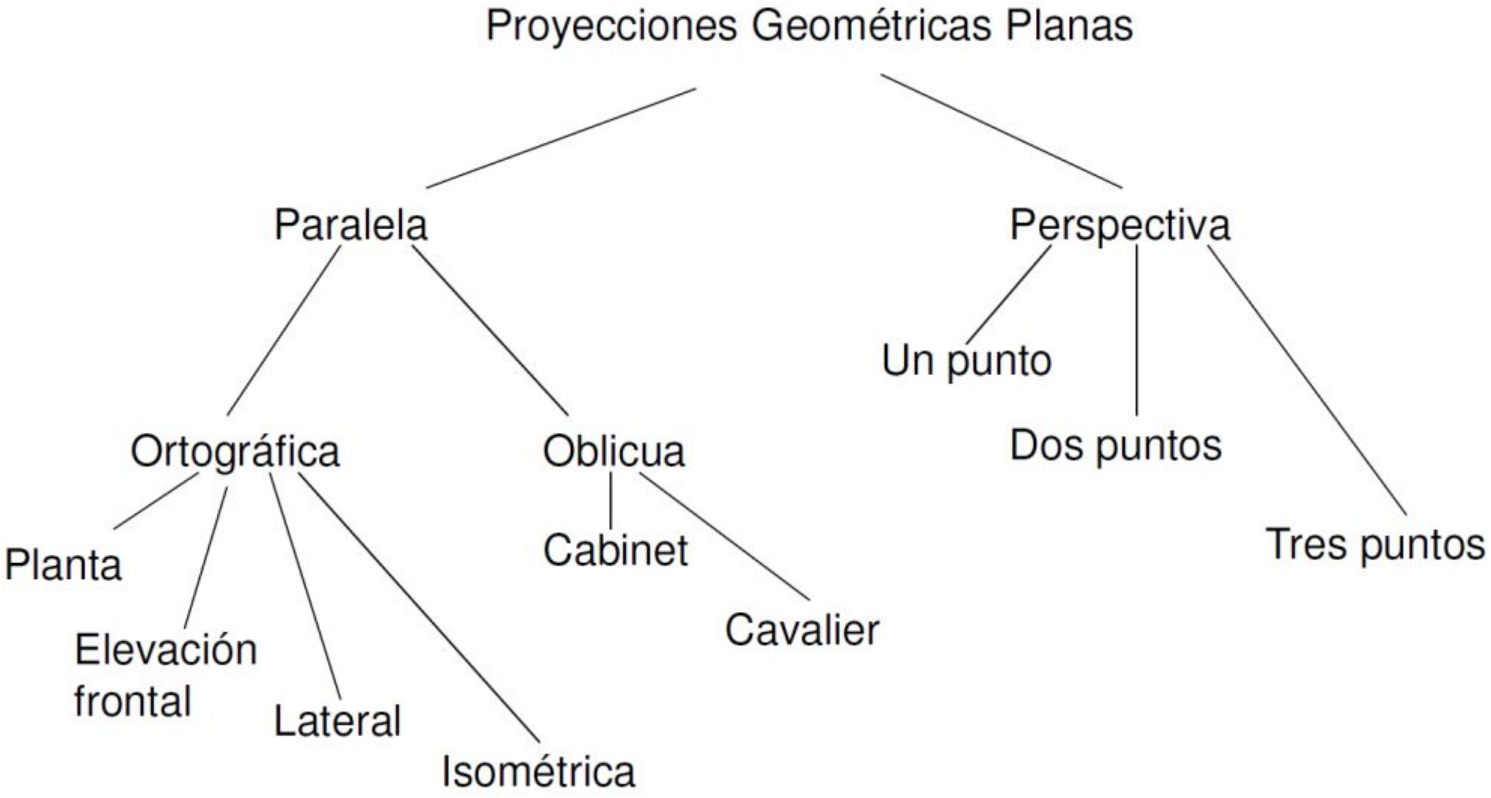


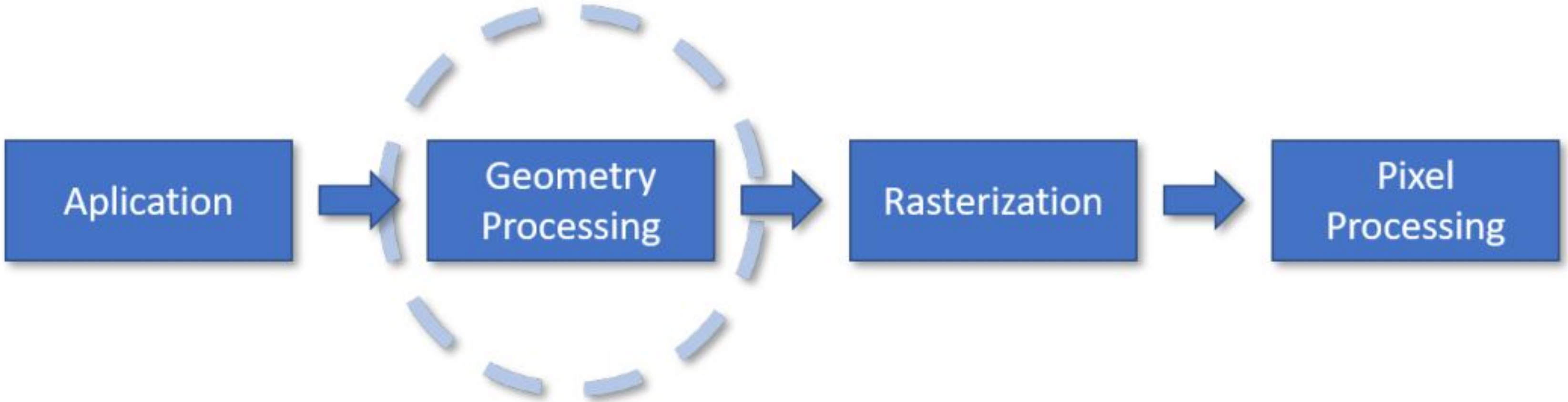
Three-point perspective

La perspectiva isométrica tiene gran éxito en distintos tipos de videojuegos.

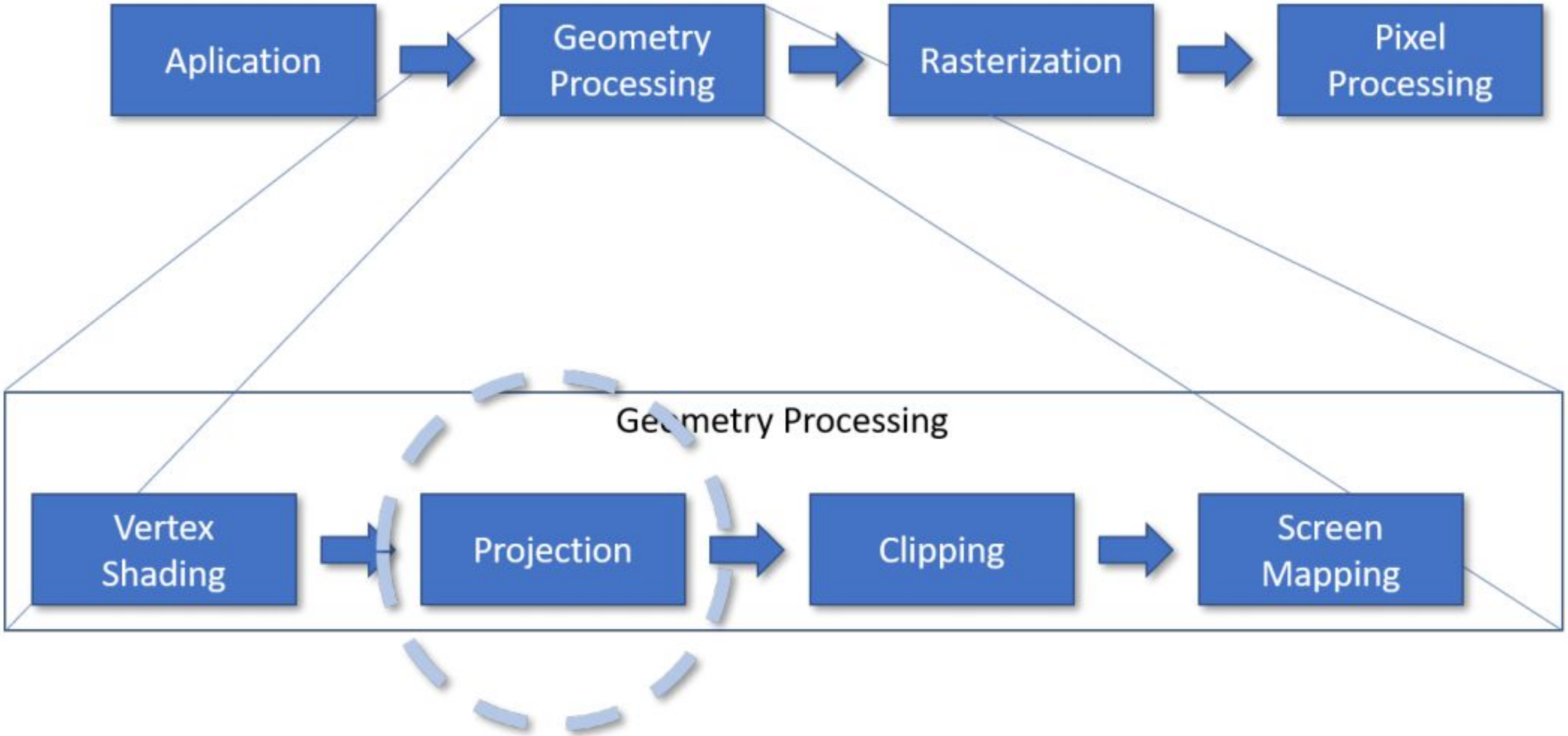


¡Hay múltiples tipos de perspectivas! Con uno, dos, e incluso tres puntos de fuga.

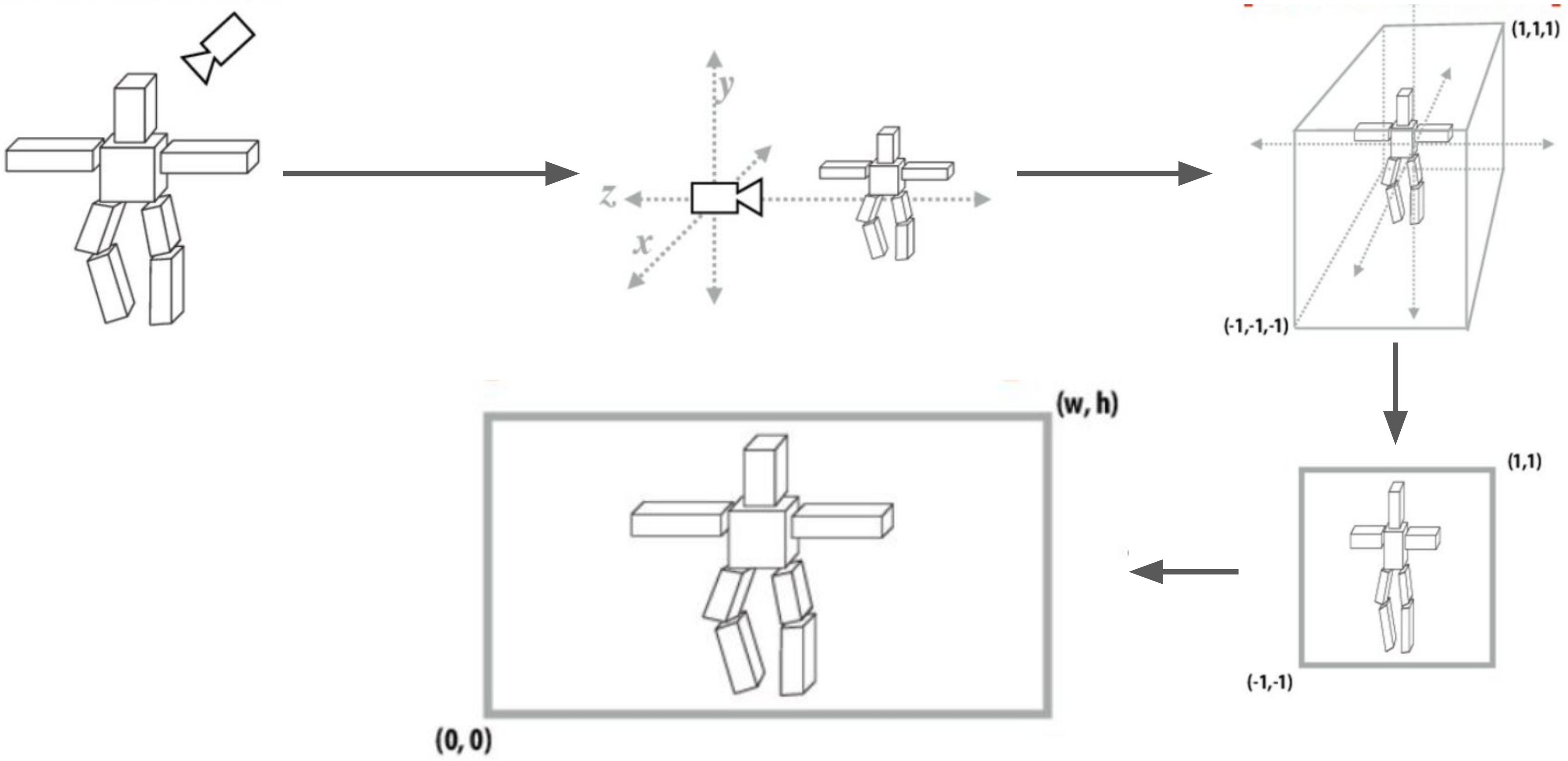

Proyecciones Geométricas Planas
Paralela
Perspectiva
Ortográfica
Oblicua
Un punto
Dos puntos
Tres puntos
Planta
Elevación frontal
Lateral
Isométrica
Cabinet
Cavalier

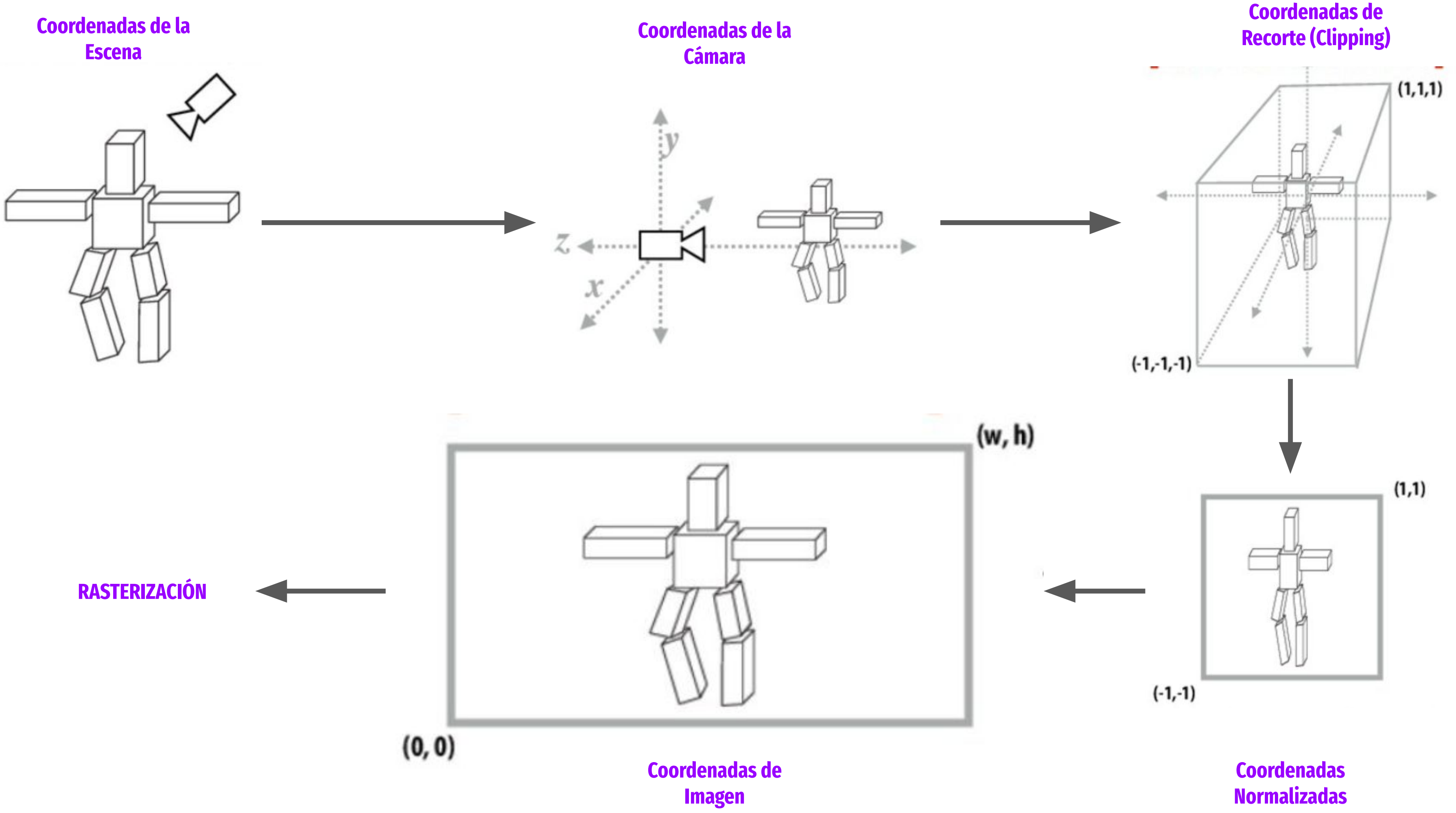


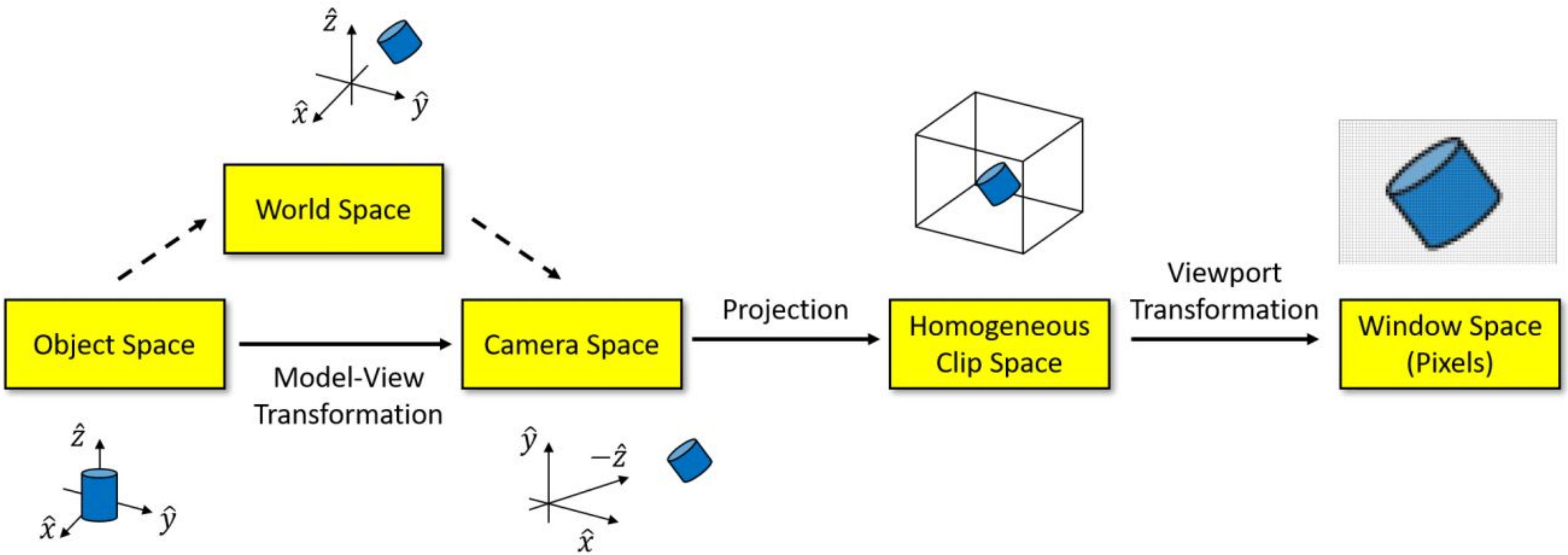
¿Por qué es relevante esto dentro del pipeline?

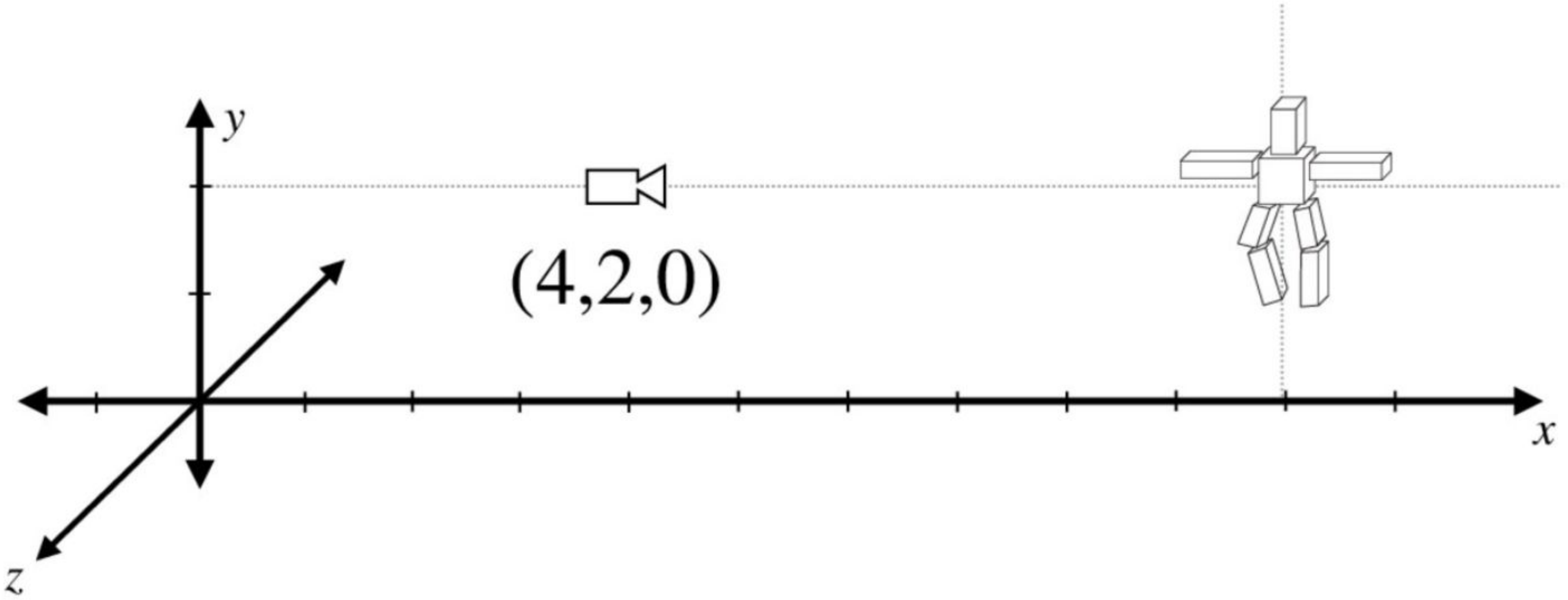


La proyección es una etapa importante del procesamiento geométrico.

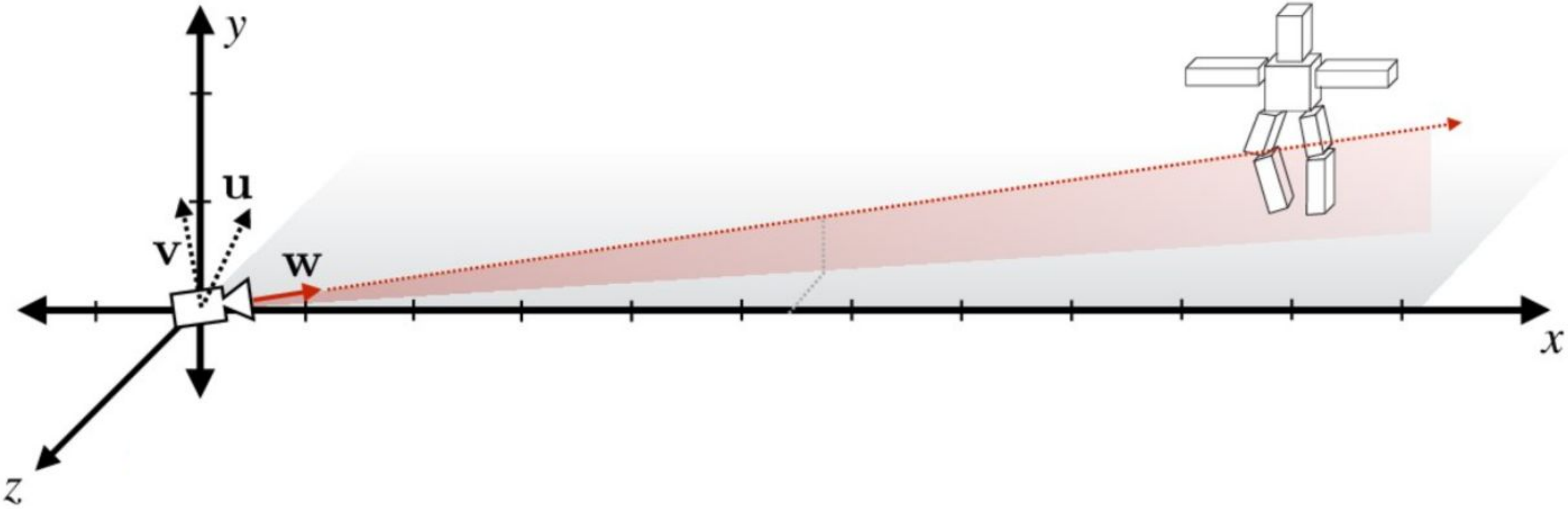

y
z
x
(1,1,1)
(-1,-1,-1)
(1,1)
(-1,-1)
(w, h)
(0, 0)


Coordenadas de la Escena
Coordenadas de la Cámara
y
z
x
Coordenadas de Recorte (Clipping)
(1,1,1)
(-1,-1,-1)
(1,1)
(-1,-1)
Coordenadas Normalizadas
(w, h)
(0, 0)
Coordenadas de Imagen
RASTERIZACIÓN


World Space
Object Space
Model-View
Transformation
Camera Space
Projection
Homogeneous
Clip Space
Viewport
Transformation
Window Space
(Pixels)
$\hat{z}$
$\hat{x}$
$\hat{y}$
$-\hat{z}$



¿Cómo podemos transformar esta escena para que el sistema de coordenadas transformado esté en el origen, con la cámara apuntando hacia el eje -z?

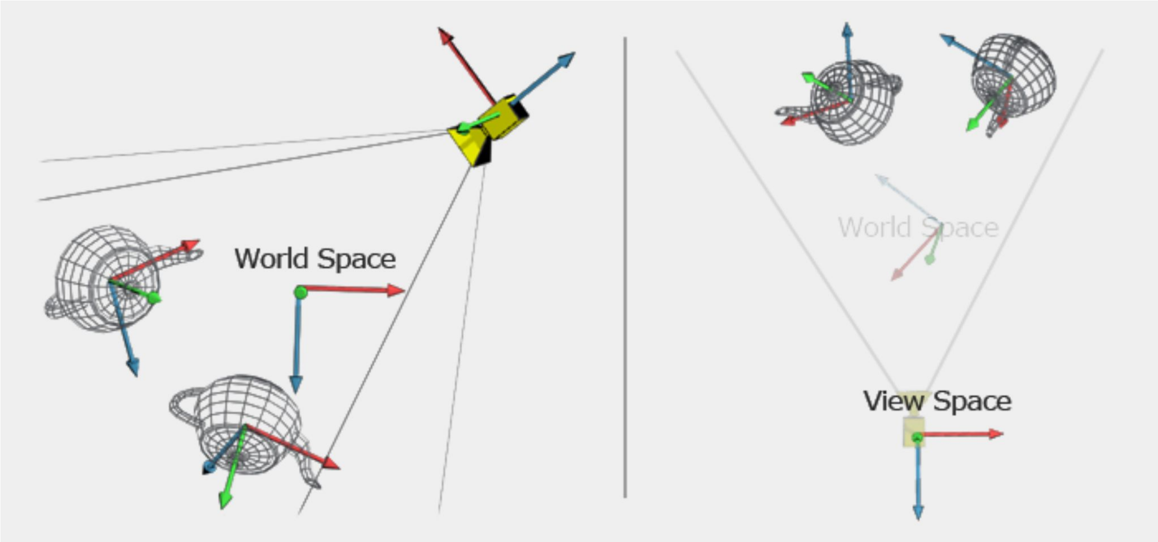


¿Qué necesitamos hacer para que la cámara apunte en la misma dirección, pero que el eje -w coincida con el eje z?

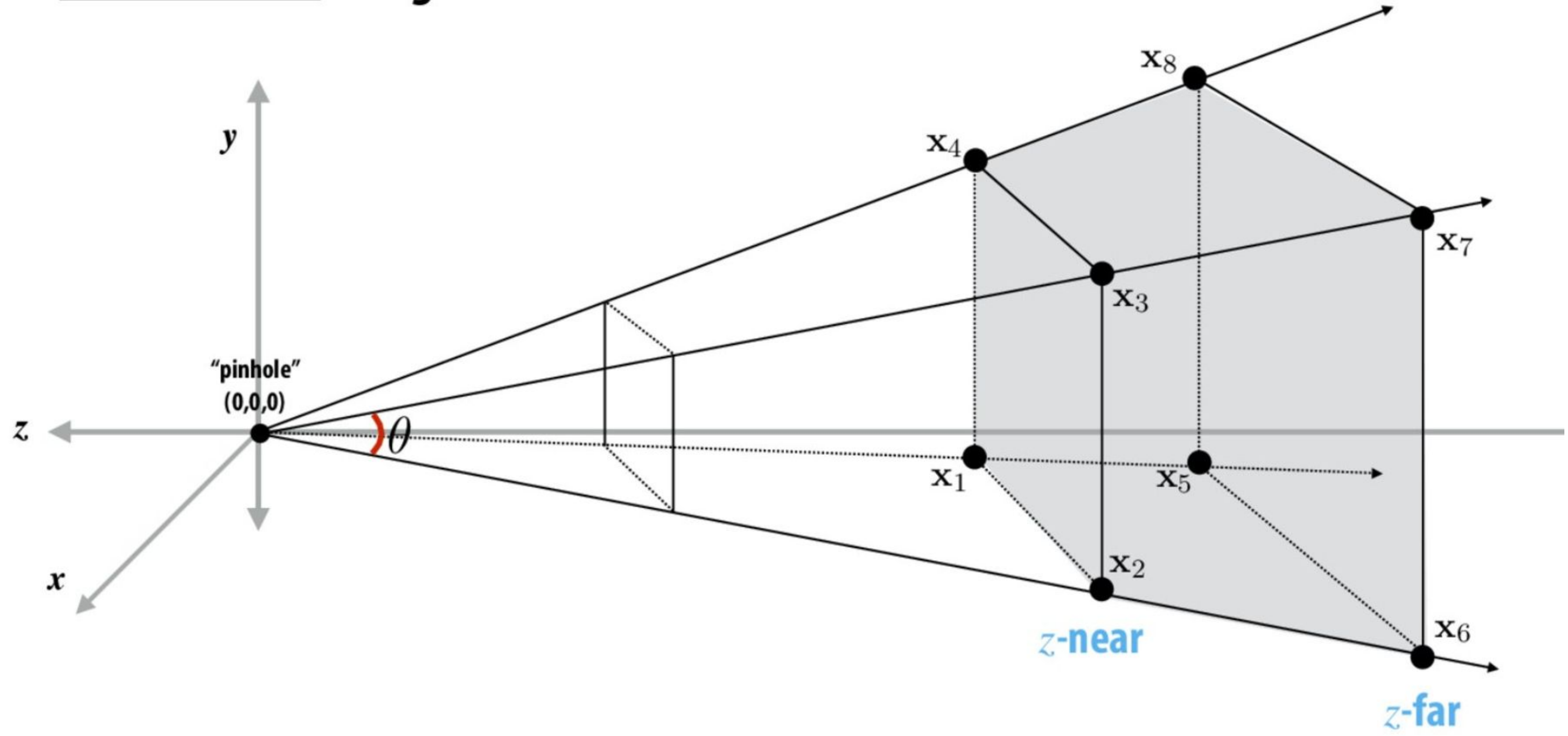
El primer paso es construir los vectores u, v y w. Con ello podemos construir la rotación necesaria para nuestros fines.

$$\begin{bmatrix} Transform\_XAxis.x & Transform\_YAxis.x & Transform\_ZAxis.x & Translation.x \\ Transform\_XAxis.y & Transform\_YAxis.y & Transform\_ZAxis.y & Translation.y \\ Transform\_XAxis.z & Transform\_YAxis.z & Transform\_ZAxis.z & Translation.z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

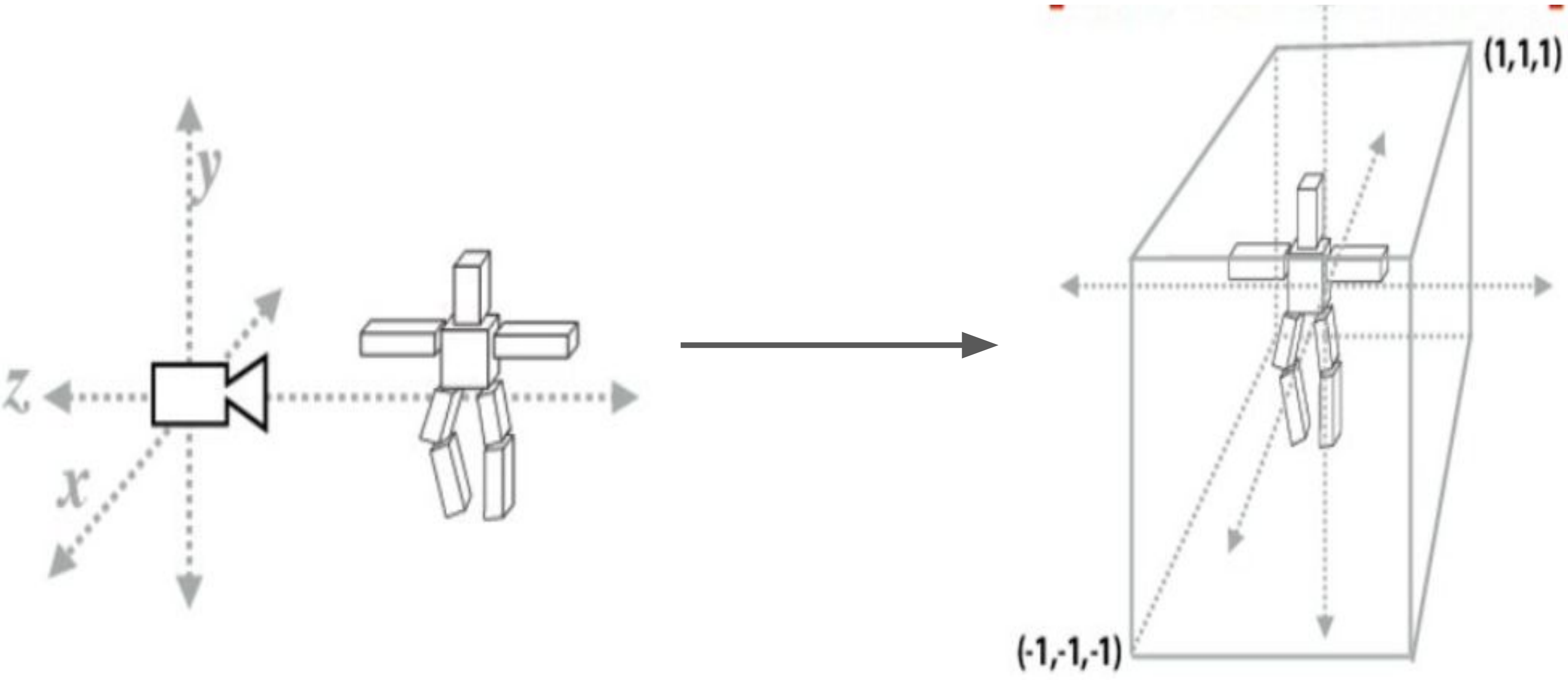
Repaso: una manera de definir las transformaciones de 4x4 es verlas como **un cambio de sistema de coordenadas**, en los que especificamos nuestro nuevo sistema como tres ejes y una traslación.



```python
def lookAt(eye, at, up):

    forward = (at - eye)
    forward = forward / np.linalg.norm(forward)

    side = np.cross(forward, up)
    side = side / np.linalg.norm(side)

    newUp = np.cross(side, forward)
    newUp = newUp / np.linalg.norm(newUp)

    return np.array([
            [side[0],        side[1],     side[2], -np.dot(side, eye)],
            [newUp[0],      newUp[1],    newUp[2], -np.dot(newUp, eye)],
            [-forward[0], -forward[1], -forward[2], np.dot(forward, eye)],
            [0,0,0,1]
        ], dtype = np.float32)
```



Luego de posicionar la cámara, definimos lo que se conoce como "**view frustum**", el poliedro de vista efectiva, definido por el ángulo de apertura de la cámara y los planos z-near y z-far. Ello nos permitirá definir nuestra **transformación de perspectiva**.



Recordemos que queremos que el frustum se exprese en coordenadas normalizadas, es decir, un cubo con sus vértices en posiciones unitarias.

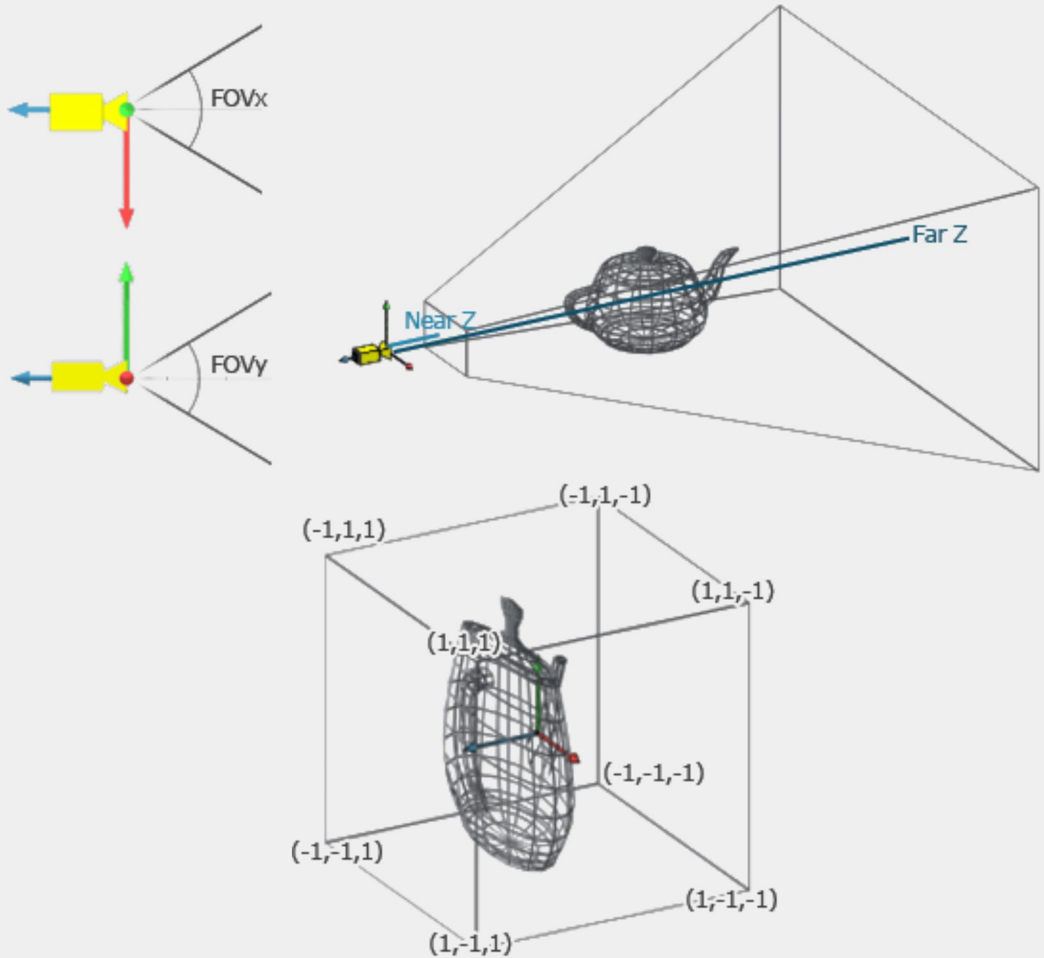
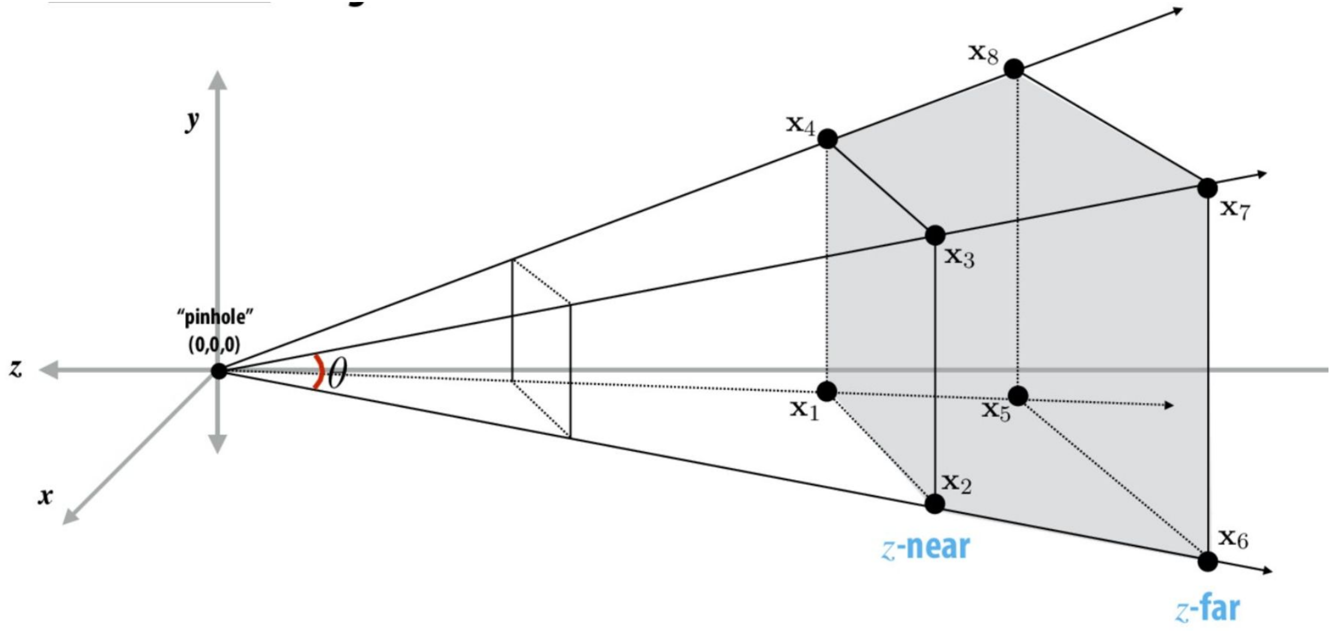


Figure 11: Perspective Projection



Si definimos como l (izquierda), r (derecha), b (abajo), t (arriba), n (cerca), f (lejos) las distintas componentes del cubo, entonces podemos definir la transformación de perspectiva con una matriz que combina escalamiento,

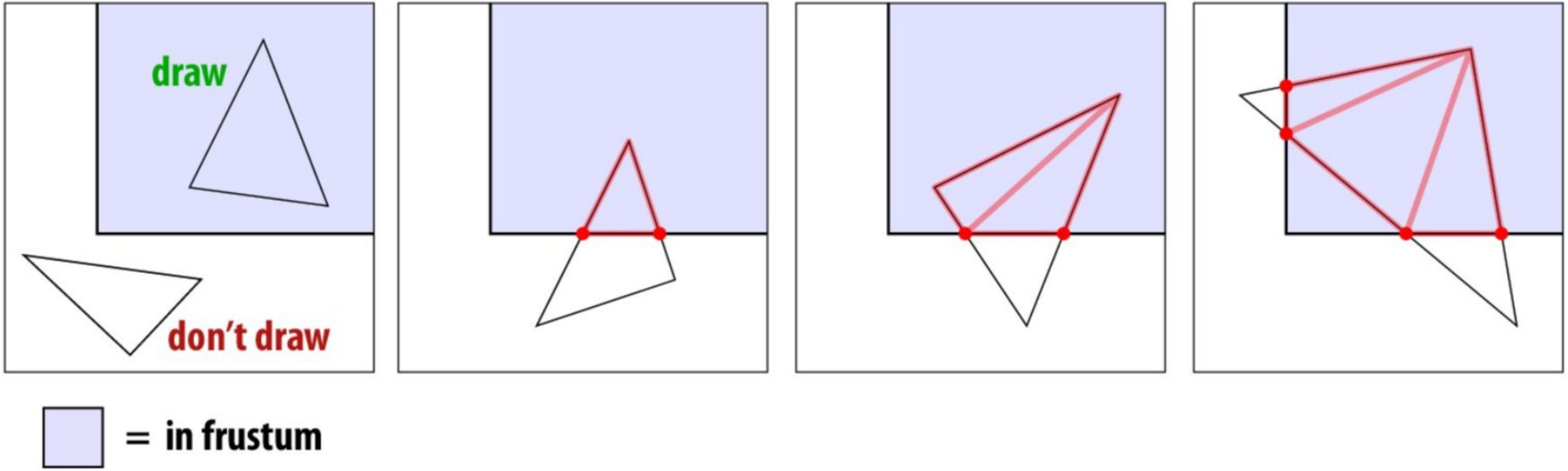
$$\begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

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

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

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

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



El frustum nos permite decidir qué graficar. Al descartar elementos que estén fuera de nuestra vista, no desperdiciamos procesamiento. Utilizar coordenadas basadas en el cubo normalizado facilita esta eficiencia.

Veamos el código

¿Preguntas?

Propuesto: construir la matriz de perspectiva ortográfica.