Espacio Clip, NDC y División por W

Nicolas Quezada Mora
Juan Sebastian Tinjaca Mesa
Jeronimo Bermudez Hernandez

¿Cómo un mundo 3D infinito termina convertido en píxeles dentro de un rectángulo?

En gráficos por computadora, un motor 3D debe transformar un mundo tridimensional continuo en una imagen bidimensional compuesta por píxeles.
La cámara virtual observa una escena con profundidad infinita, pero la pantalla solo puede representar un área finita.

Para resolver este problema, el pipeline gráfico no proyecta directamente la escena a la pantalla.

Concepto

Espacio Clip:
Se determina qué geometría es visible; lo demás se descarta.

División por W:
Las coordenadas se dividen por su profundidad, generando la perspectiva
(cercano grande, lejano pequeño).

Espacio NDC:
Todo lo visible se normaliza al rango −1 a 1 dentro de un cubo estándar, que luego se convierte en píxeles.

Fundamento matemático

Un vértice en espacio de cámara se representa en coordenadas homogéneas:
$P_{view} = (x, y, z, 1)$
Se aplica la matriz de proyección:
$P_{clip} = M_{proj} * P_{view} = (x_c, y_c, z_c, w_c)$
El componente wcw_cwc queda relacionado con la profundidad ($w_c \approx -z$)
El hardware realiza la división por W:
$x_{ndc} = x_c / w_c$ $y_{ndc} = y_c / w_c$ $z_{ndc} = z_c / w_c$
Lo que produce:
$x_{ndc} \propto x / z$
$y_{ndc} \propto y / z$
Esta relación genera la perspectiva y transforma el frustum de visión en el cubo normalizado [−1,1][-1,1][−1,1] (NDC)

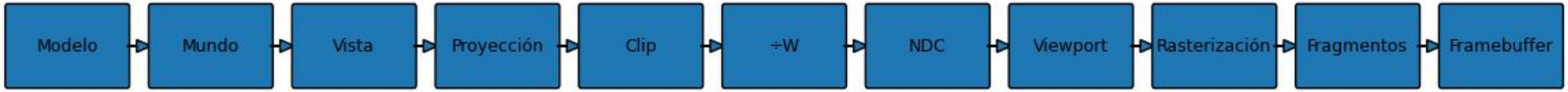
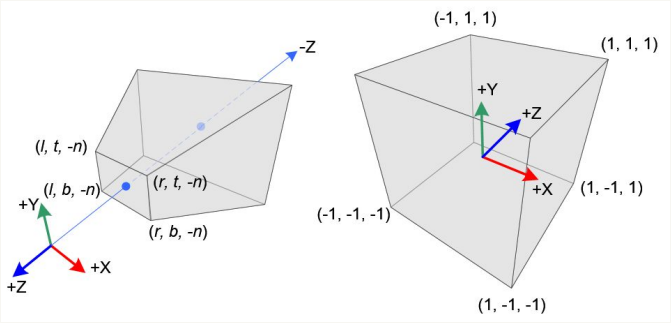


Diagrama de pipeline

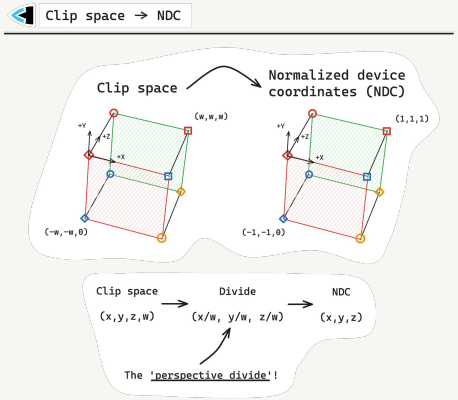
Ejemplos visuales

Antes de rasterizar, la GPU trabaja en clip space homogéneo (x, y, z, w) donde recorta (clipping), y después hace la división por W para llegar a NDC

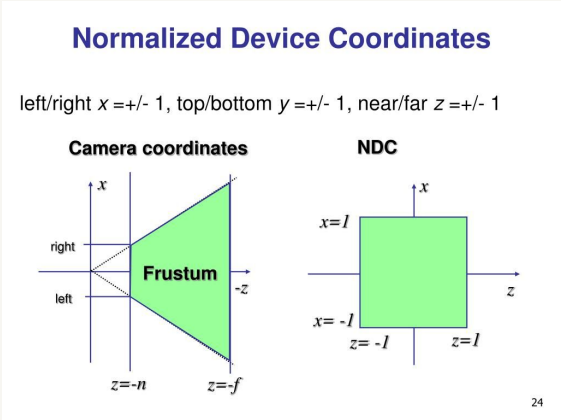
Frustum → Cubo canónico



Clip space -> NDC

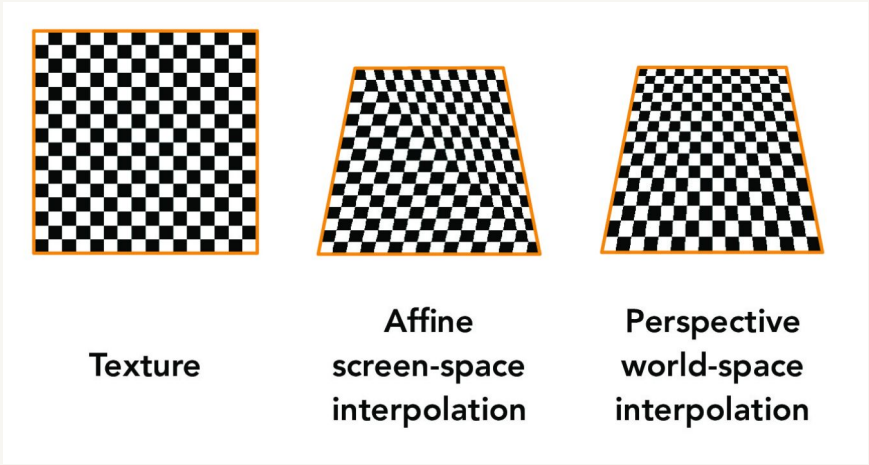


MODERN WORLD

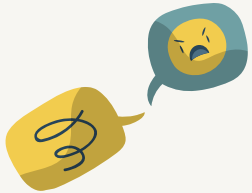


Errores comunes:

La textura se deforma cuando la interpolación no es perspective-correct



Conexión con motores gráficos reales



- **En un motor real, la “salida” del Vertex Shader es *clip space***: se entrega un vector(x,y,z,w) y la GPU ya sabe **dónde recortar (clip)** y luego **rasterizar**
- **La “División por W” la hace el pipeline** al final del VS/ensamblado, convirtiendo *clip space* → **NDC** (Normalized Device Coordinates).