



Seminario 1

Introducción a los motores gráficos



¿Qué es un motor gráfico?



¿Qué es un motor gráfico para ti?



¿Qué es un motor gráfico?



- Motor gráfico: sistema formado por:
 - Hardware que ejecuta las tareas de procesamiento gráfico: GPU o tarjeta aceleradora de gráficos
 - Software que acepta comandos de una aplicación y construye imágenes y texto que se dirigen a la salida y a la tarjeta gráfica
- Para nosotros:
 - Conjunto de funciones, tipos de datos, configuraciones y optimizaciones de hardware y otros elementos para facilitar el desarrollo de aplicaciones gráficas (en 3D).



¿Qué es un motor gráfico?



- Otros nombres
 - Librería gráfica
 - API gráfico
 - Motor de juegos
 - Motor 3D



¿Son equivalentes?



¿Qué es un motor gráfico?



- Problemática principal:
 - La representación de un mundo 3D sobre un plano 2D hace necesarias las PROYECCIONES
 - Necesidad de métodos para añadir REALISMO
 - Para juegos es necesario TIEMPO REAL



¿Qué sabemos de gráficos?



¿Qué conceptos de gráficos conoces?



Vértices



- Elemento clave en un sistema gráfico 3D: lugar geométrico exacto, sin volumen, área o longitud
 - En el espacio cartesiano 3D viene representado por tres valores (x, y, z)
 - En Gráficos, suele utilizarse una cuarta componente: (x, y, z, w)
 - w está relacionada con el uso de coordenadas homogéneas
 - Indica un factor de escala, por eso suele omitirse cuando vale 1
- En los sistemas gráficos poligonales (la mayoría) todas las demás figuras geométricas se forman a partir de vértices (geométricamente puntos)



Polígonos



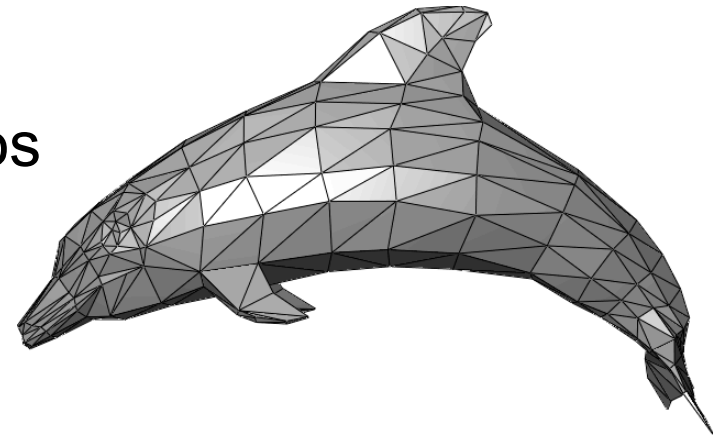
- Figura geométrica plana, cerrada, limitada por segmentos rectos consecutivos no alineados, trazados entre los vértices y que llamaremos aristas
- Todo polígono tiene al menos 3 vértices
- Un polígono con 3 vértices tiene sus vértices coplanares, es decir, definen un plano
- Dados tres vértices V_1, V_2, V_3 , podemos definir:
 - Normal al plano: $N=(N_x, N_y, N_z)=(V_2-V_1) \times (V_3-V_1)$, es decir,
$$N=\text{Det}[V_2-V_1, V_3-V_1, (i, j, k)]$$
 - Ecuación implícita del plano $Ax+By+Cz+D=0$ donde
 - $A=N_x$
 - $B=N_y$
 - $C=N_z$
 - D se obtiene despejando en la ecuación



Mallas



- Conjunto de vértices, aristas y polígonos que definen la forma de un objeto poliédrico en 3D
- Los polígonos puedes ser:
 - Triángulos
 - Cuadriláteros
 - Otros polígonos convexos





Transformaciones



- Mecanismo para alterar las las coordenadas de los puntos 3D y, en consecuencia, de las figuras geométricas
- Se definen mediante matrices cuadradas 4x4 que se multiplican por el punto para obtener un nuevo punto transformado
- Aunque los puntos sean 3D, se añade una cuarta componente (coordenadas homogéneas): $(x,y,z) \rightarrow (x,y,z,1)$
- De esta manera, cualquier transformación se puede definir mediante la composición (producto) de matrices de transformación más simples:

$$P' = T_n \cdot \dots (T_3 \cdot (T_2 \cdot (T_1 \cdot P))) = (T_n \cdot \dots \cdot T_3 \cdot T_2 \cdot T_1) \cdot P$$

- Transformaciones básicas
 - Traslación
 - Rotación
 - Escalado
- La composición de transformaciones es:
 - Asociativa
 - No conmutativa en general



Transformaciones

recuerda

$T(t_x, t_y, t_z) = \text{Traslación}$

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

$S(s_x, s_y, s_z) = \text{Escalado}$

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

$R_x(\theta), R_y(\theta), R_z(\theta) = \text{Rotación}$
un ángulo θ alrededor de los ejes

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

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

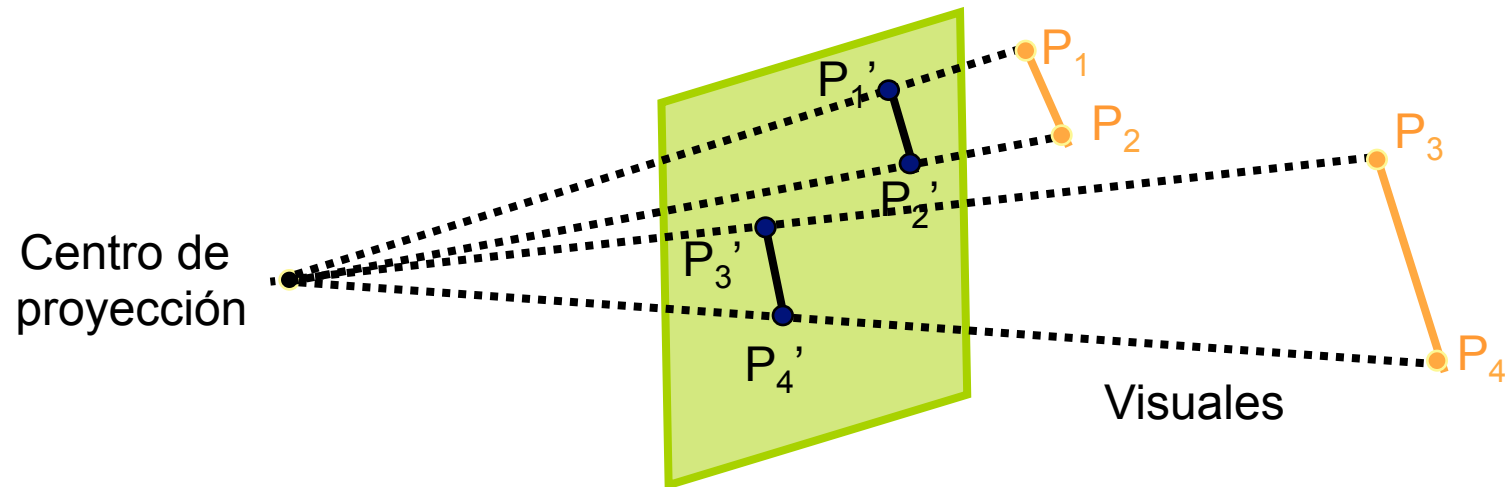
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



Proyecciones



- Proyección: tipo especial de transformación que permite representar los objetos 3D en un plano 2D (plano de proyección)
- Procedimiento básico:
 - Trazar rectas (visuales) desde un punto (centro de proyección) a los puntos del objeto
 - Intersectar cada visual con el plano de proyección

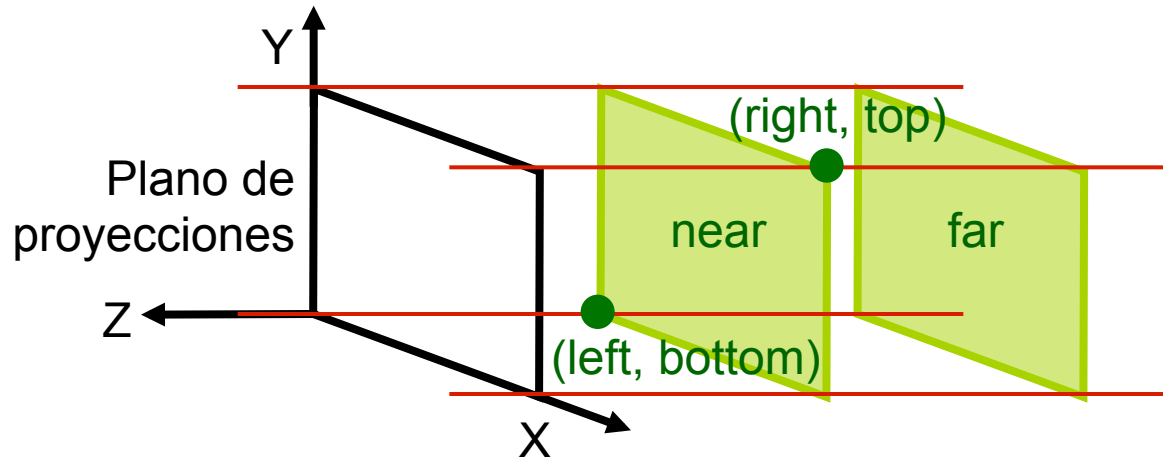




Proyecciones paralelas



Centro de proyección infinito → Visuales paralelas



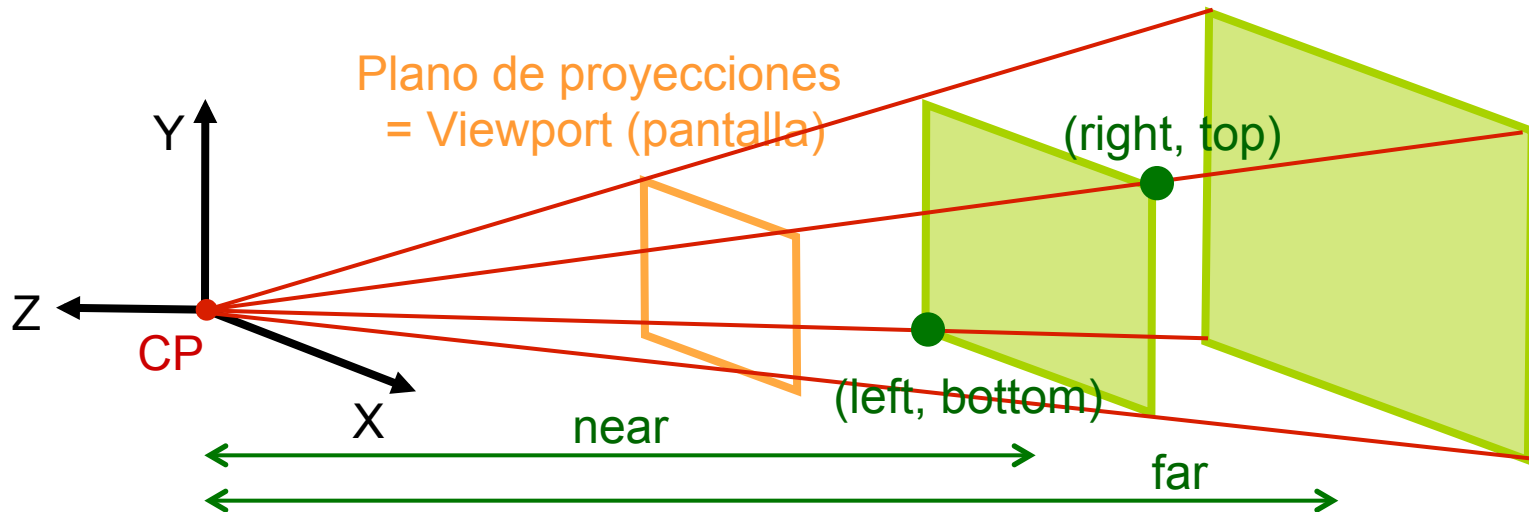
$$P_{orthogonal} = \begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bottom} & 0 & -\frac{top + bottom}{top - bottom} \\ 0 & 0 & \frac{2}{near - far} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Proyecciones perspectivas



Centro de proyección finito → Visuales convergen en CP



$$P_{perspective} = \begin{bmatrix} \frac{2 \cdot \text{near}}{\text{right} - \text{left}} & 0 & \frac{\text{right} + \text{left}}{\text{right} - \text{left}} & 0 \\ 0 & \frac{2 \cdot \text{near}}{\text{top} - \text{bottom}} & \frac{\text{top} + \text{bottom}}{\text{top} - \text{bottom}} & 0 \\ 0 & 0 & -\frac{\text{far} + \text{near}}{\text{far} - \text{near}} & -\frac{2 \cdot \text{far} \cdot \text{near}}{\text{far} - \text{near}} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$



¿Qué es el pipeline?



¿Qué es el pipeline o tubería 3D?



Pipeline

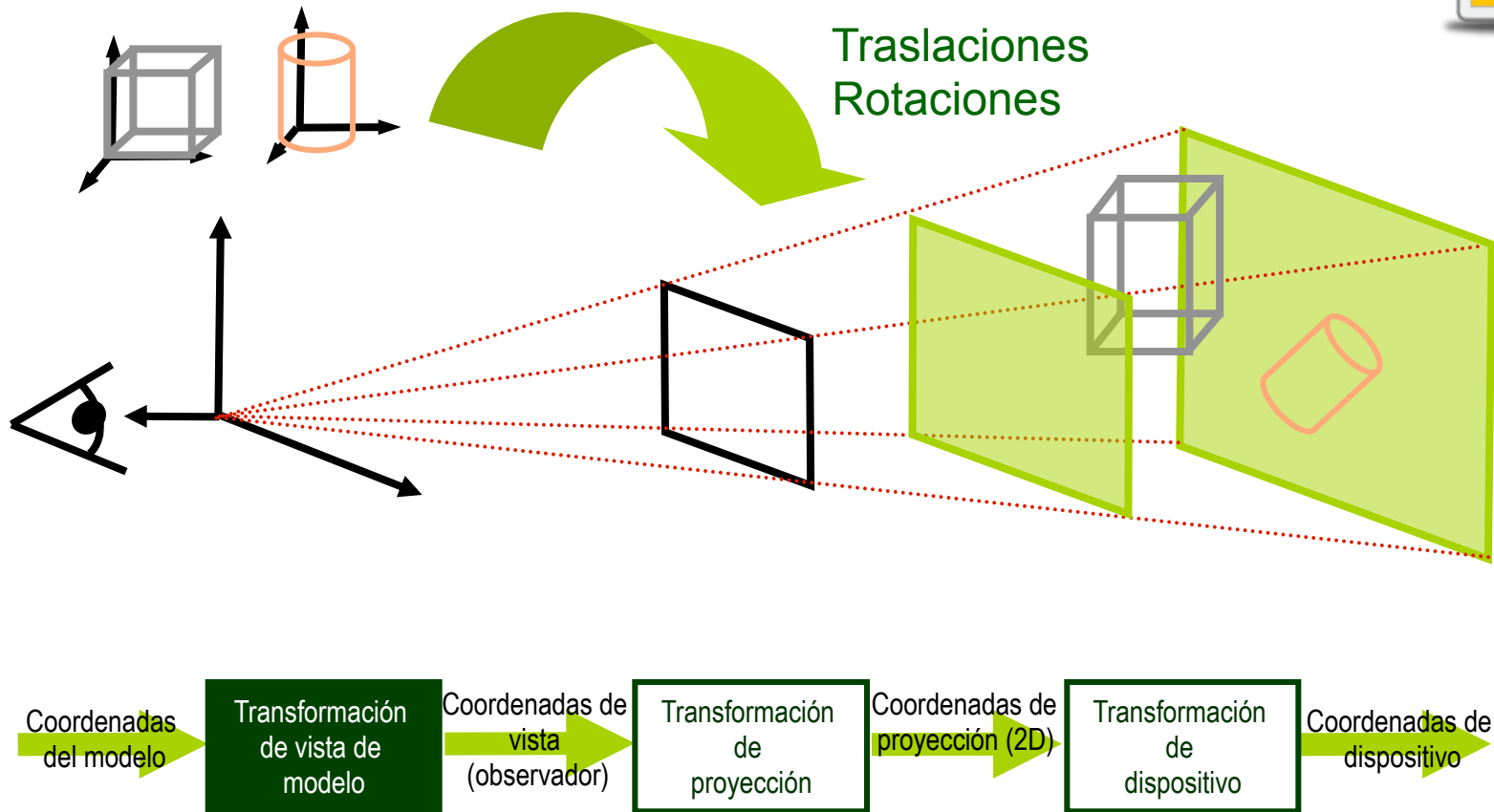


- **Tubería o pipeline 3D:** proceso que siguen los datos geométricos para obtener una representación 2D realista de los datos 3D
- Las operaciones básicas para reducir una dimensión son:
 - Transformaciones geométricas básicas:
 - Traslación
 - Escalado
 - Rotación
 - Proyecciones



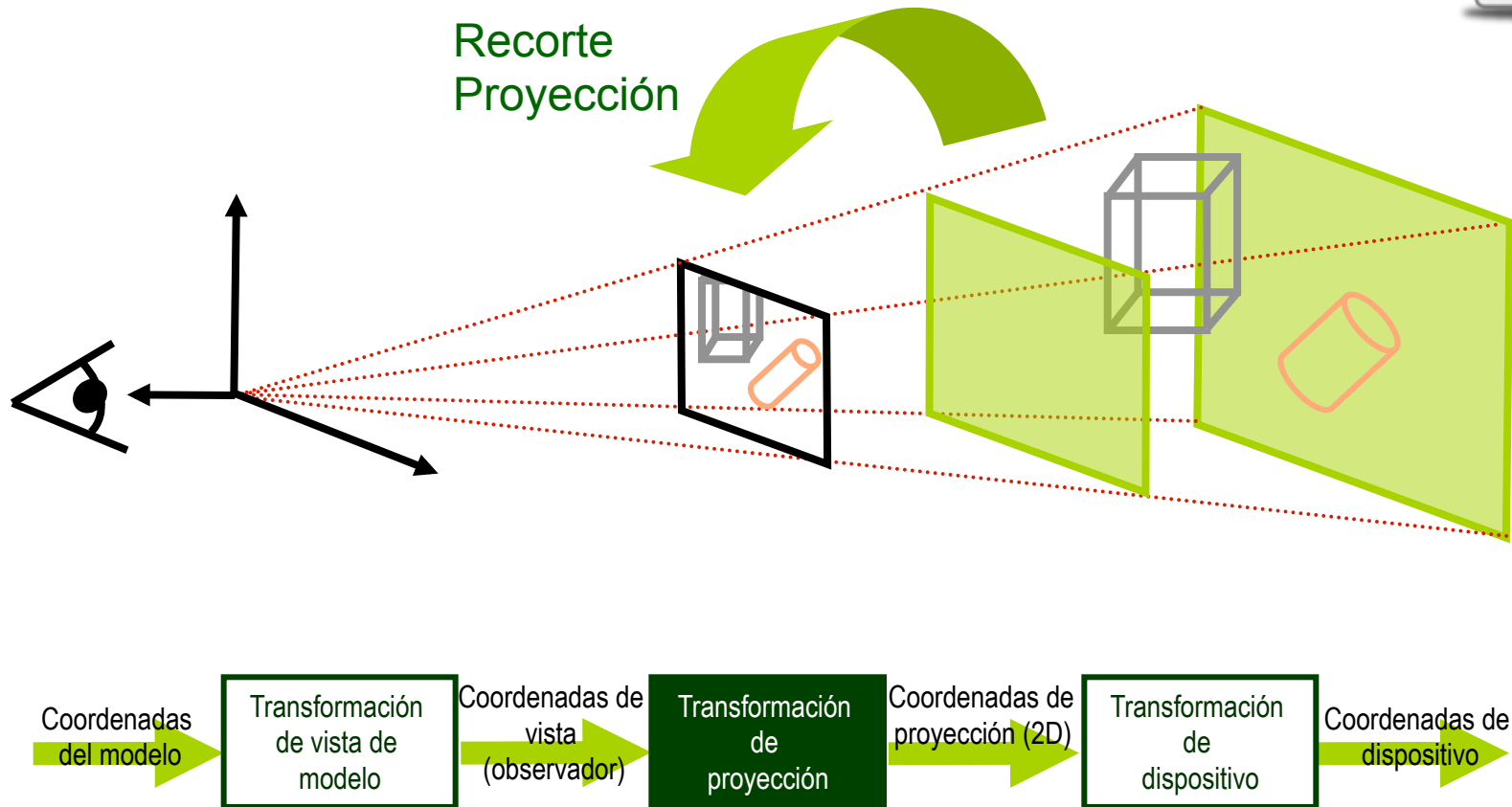


Pipeline



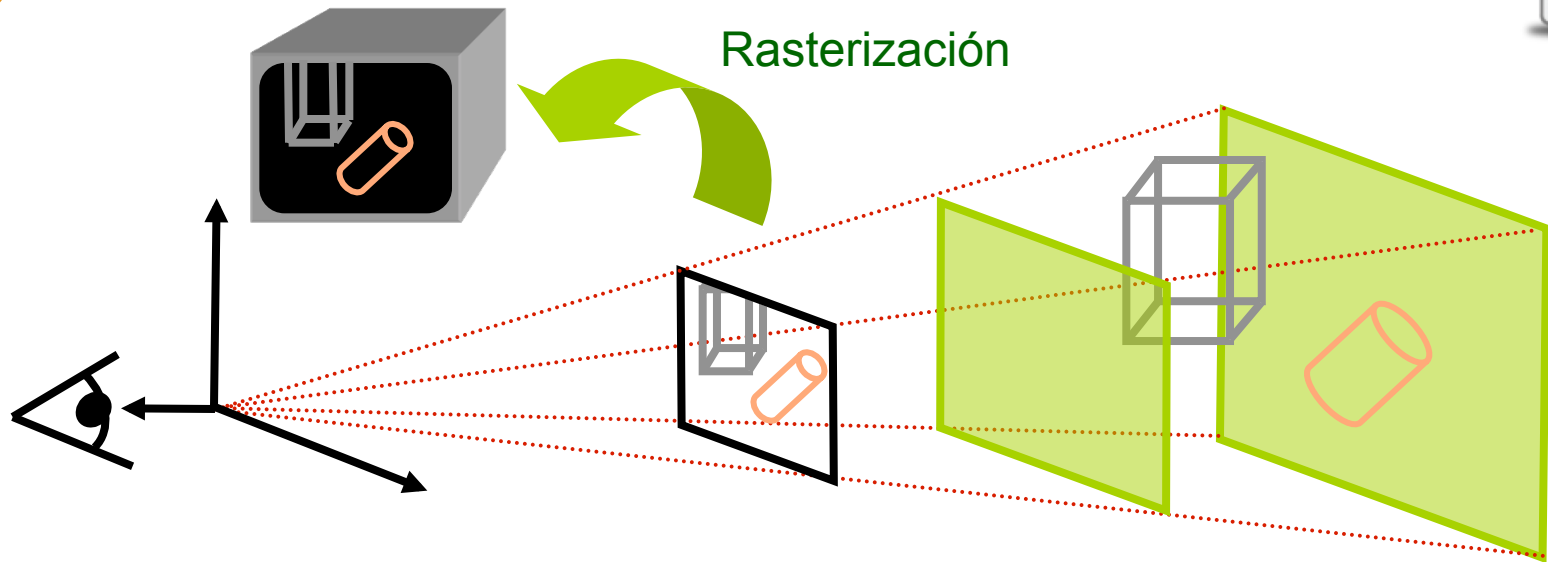


Pipeline





Pipeline

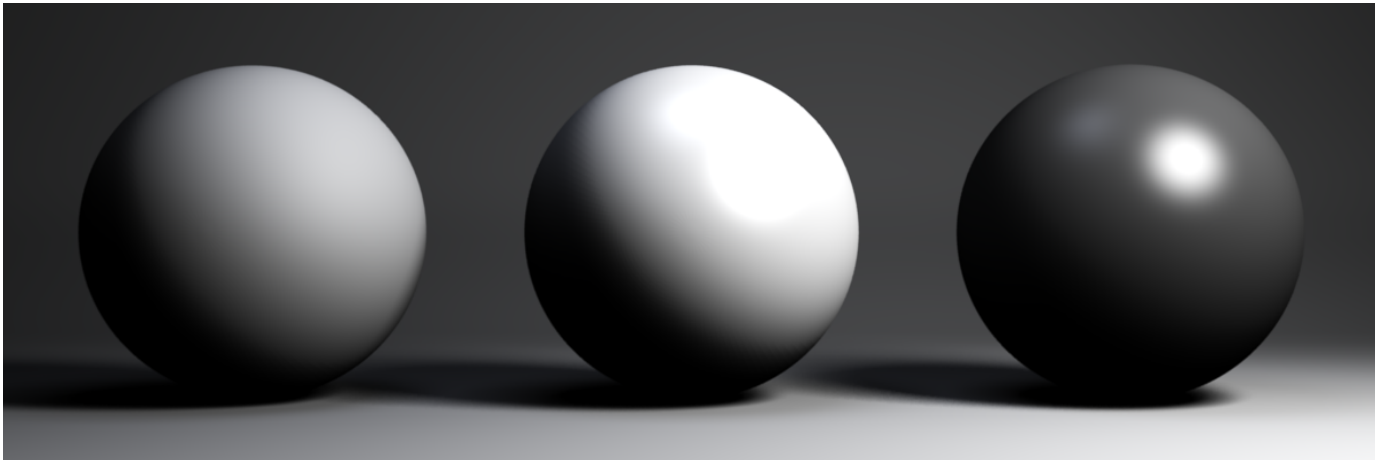




Atributos: materiales



- Material
 - Color
 - Transparencia
 - Reflexión difusa
 - Reflexión especular
- Normales
- Modelo de reflexión
 - Phong
 - Blinn ...



Atributos: texturas



- Texturas
 - Mapeado de texturas (texture mapping)
 - Mapeado del entorno (environment mapping)
 - Texturas sólidas (solid textures)
 - Texturas por normales (bump mapping)

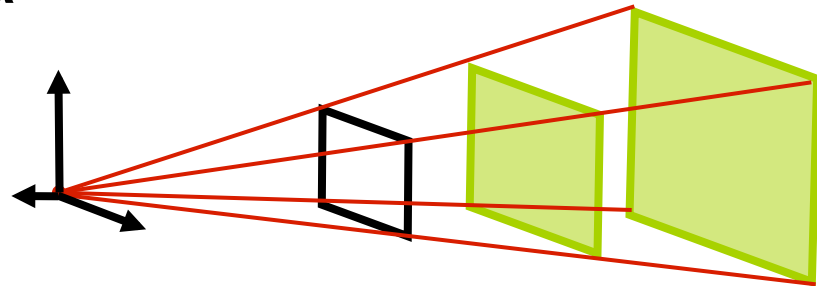
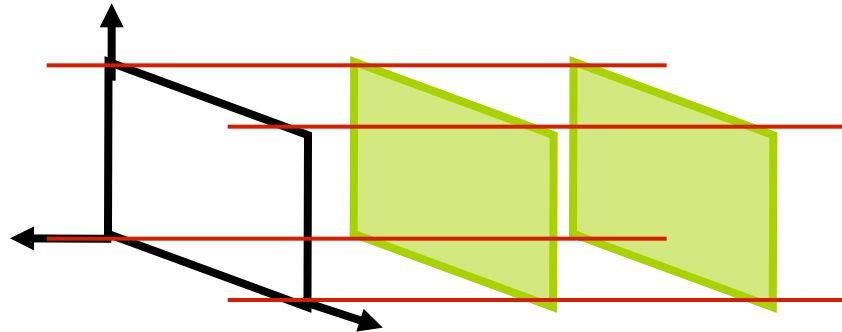




Cámara



- Proyección
 - Paralela
 - Perspectiva
- Atributos cámara
 - Posición
 - Orientación
 - Enfoque
 - Zoom
 - Amplitud de campo



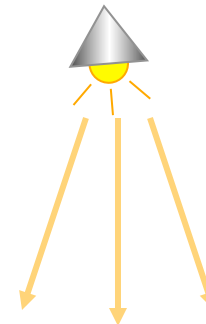
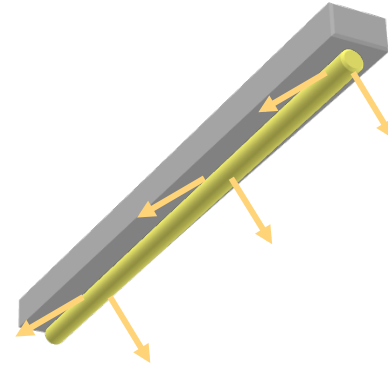
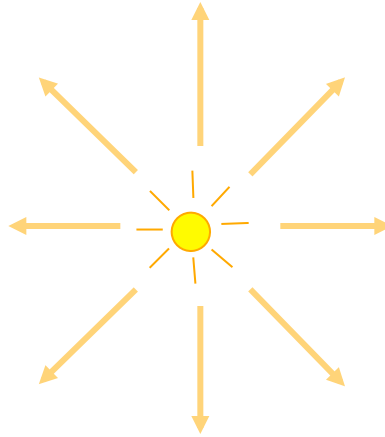
Volumen de recorte
(frustum)



Luces



- Tipo de luz
 - Puntual
 - Distribuida
 - Dirigida
- Atributos
 - Color
 - Intensidad
 - Posición
 - Atenuación
 - Dirección (si es dirigida)
 - Forma (si es distribuida)

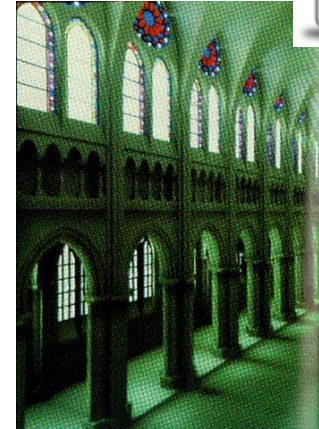
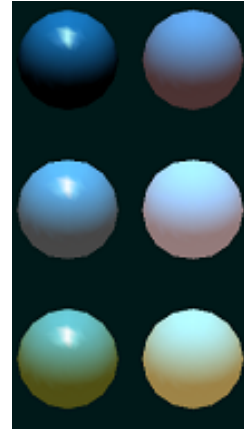




Render



- Algoritmo del pintor
- Z-buffer
- Trazado de rayos (Raytracing)
- Radiosidad (Radiosity)
- Mejora del rendimiento
 - Recortado (clipping)
 - Eliminación de caras ocultas (backface culling)
 - Utilización de doble buffer
 - Vertex shaders + pixel shaders
- Shading
 - Plano (flat)
 - Gouraud
 - Phong

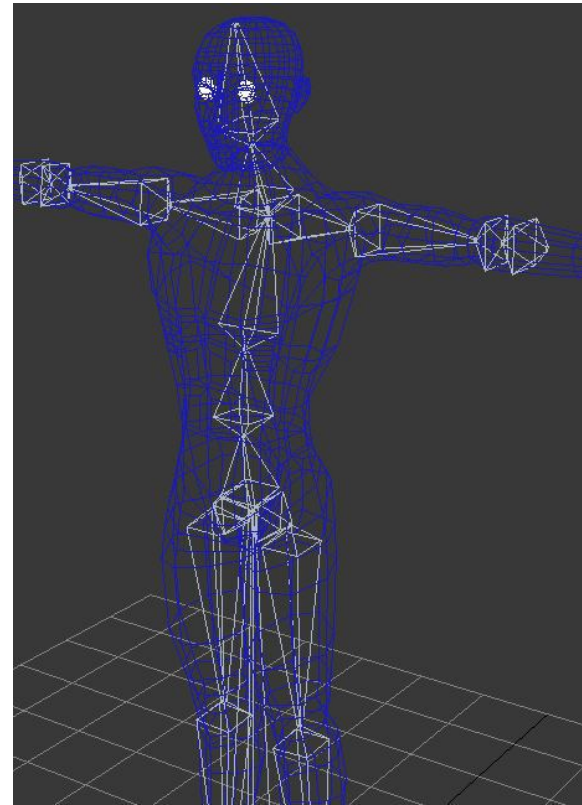




Animación



- Animación de modelos
- Animación por interpolación
- Animación de estructuras articuladas
- Animación de objetos blandos
- Animación por sistemas de partículas

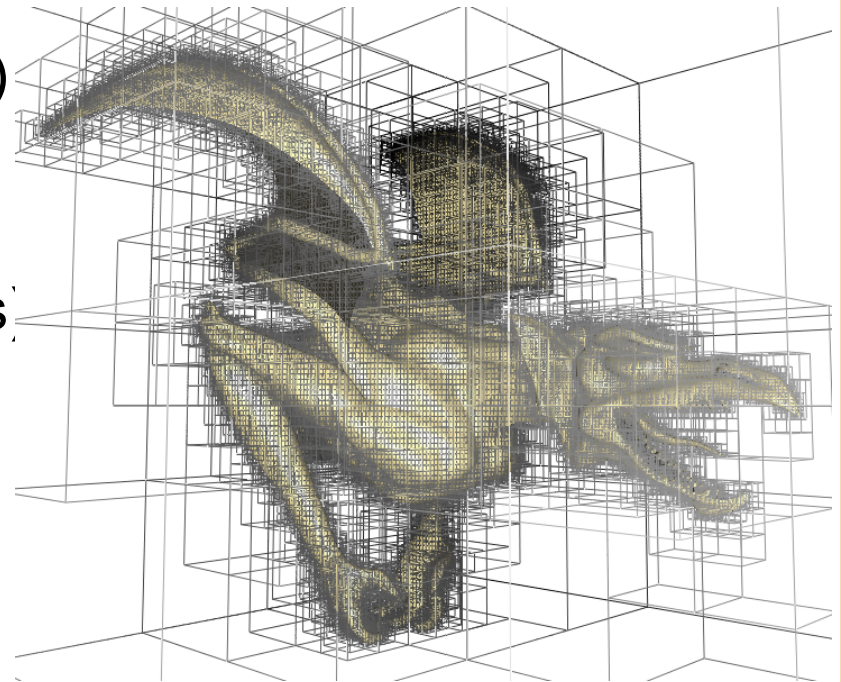




Más



- Estructuras
 - Volúmenes envolventes
 - Árboles octales (octrees)
 - Árboles BSP
 - Grafos
 - Niveles de detalle (LODs)
- Funciones
 - Intersecciones
 - Colisiones
 - Visibilidad
 - Iluminación y render
 - Carga y movimiento





¿Cómo organizamos todo esto?



¿Cómo organizamos todo esto para
construir un motor gráfico?



¿Qué nos proporciona un motor ?



- Estructura de datos para almacenar los objetos de la escena y sus relaciones
- Funciones para importar datos de otros programas (modelos, texturas...)
- Funciones y estructuras de datos para implementar la tubería 3D
- Interfaz, para facilitar y optimizar la visualización y el acceso a la tarjeta gráfica



Pero eso ¿no lo hace ya OpenGL?



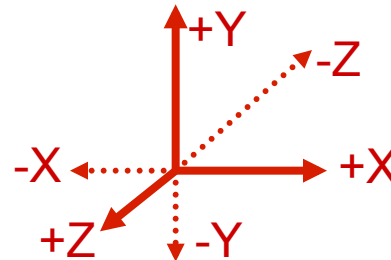
¿Es OpenGL un motor gráfico?



Nuestro motor gráfico: TAgEngine



- Motor orientado a objetos en C++ (u otro lenguaje OO)
- No implementaremos la visualización: la hará OpenGL (u otra librería gráfica)
- Utilizaremos el sistema de coordenadas y la mayoría de las convenciones gráficas de OpenGL: Dextrógiro





TAGengine



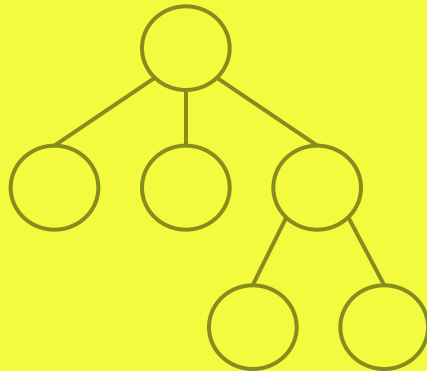
Aplicación



Interfaz



Árbol de la escena



Gestor de recursos

