

# Computación Gráfica

# Introducción

- Qué es la computación gráfica?

# Introducción

- Qué es la computación gráfica?
- Dónde y para qué se usa?
  - Películas
  - Videojuegos
  - Simulaciones
  - Educación
  - 3D Printing

# Películas



# Videojuegos

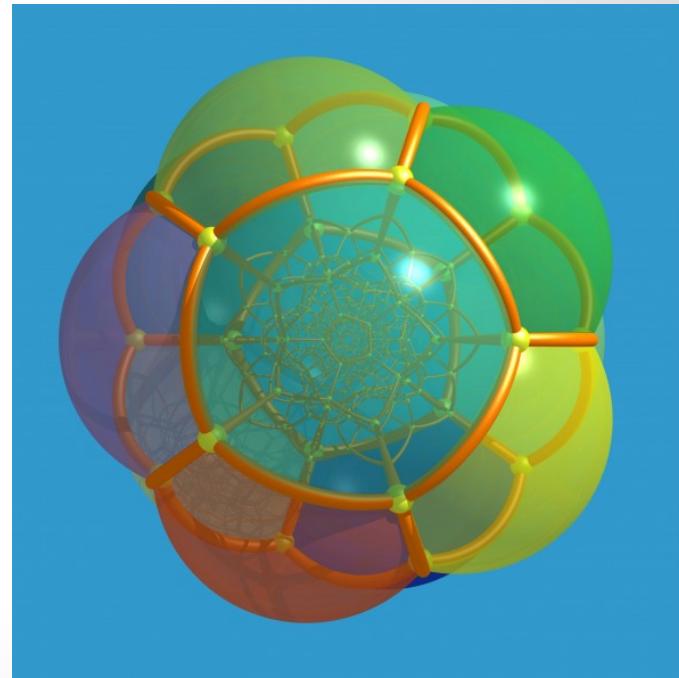
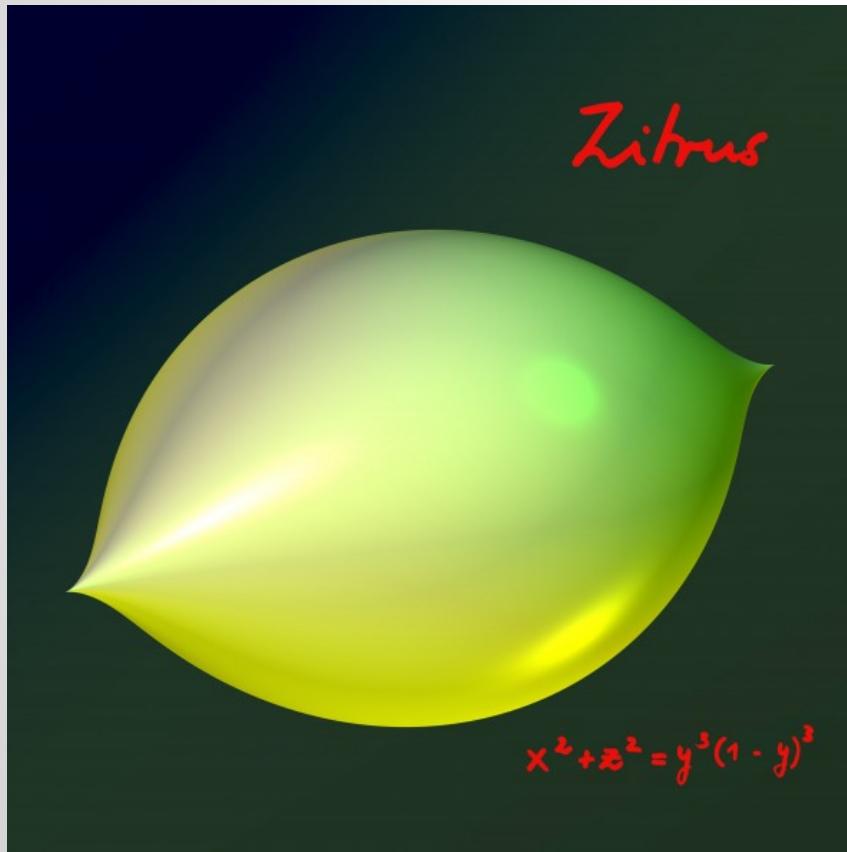


DRAGON AGE  
**INQUISITION**

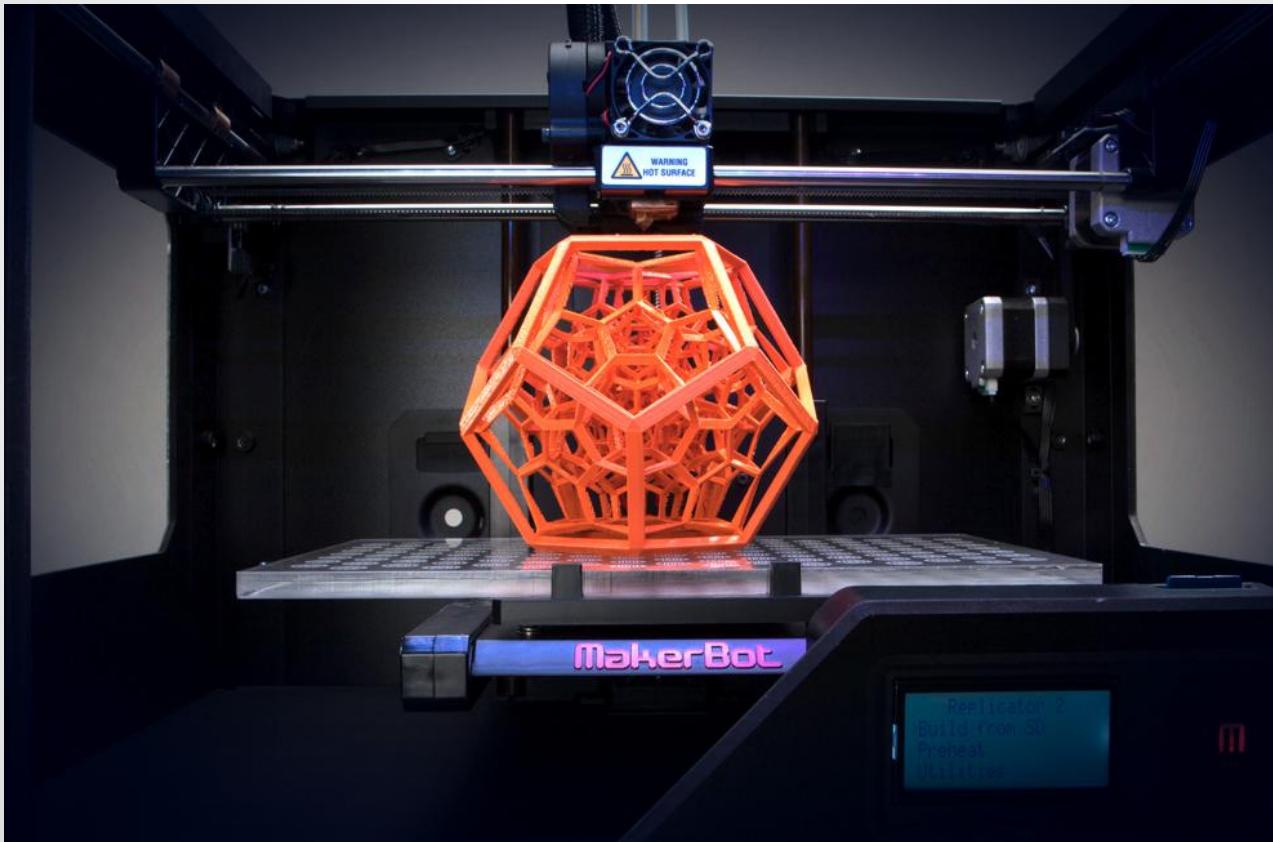
# Simulaciones



# Educación



# 3D Printing



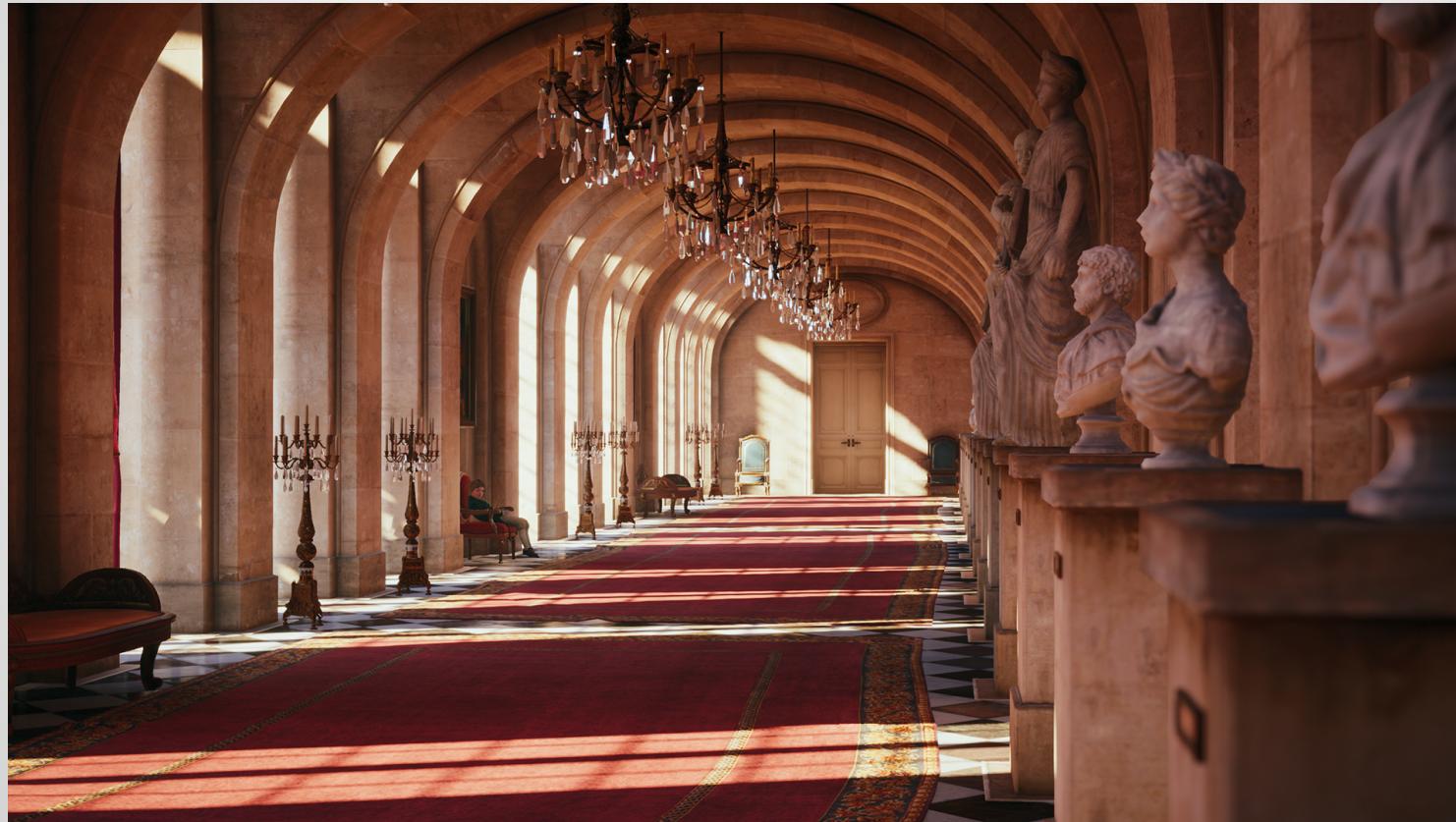
# Introducción

- Qué es la computación gráfica?
- Dónde y para qué se usa?
  - Películas
  - Videojuegos
  - Simulaciones
  - Educación
  - 3D Printing
- Tiempos de render: prerender vs. realtime

# Prerender (~29hs)



# Realtme (~16.66ms)



# Arquitecturas principales

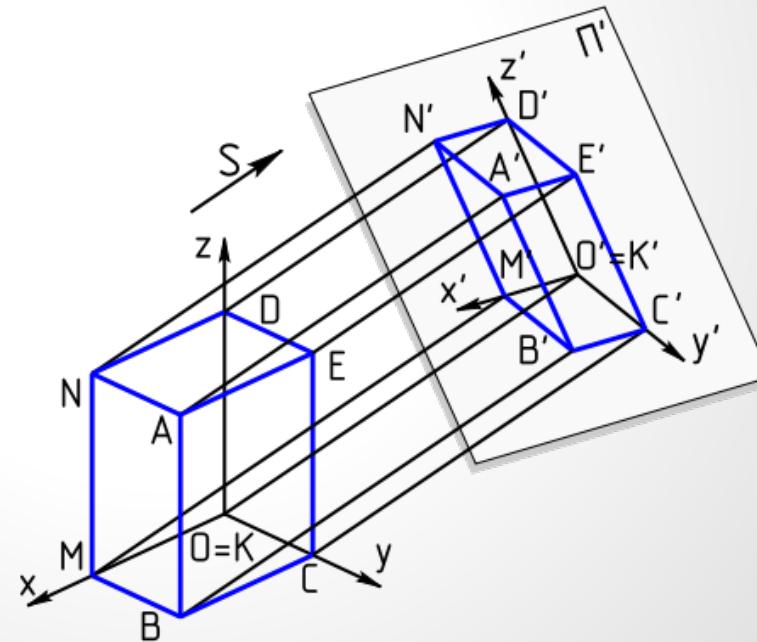
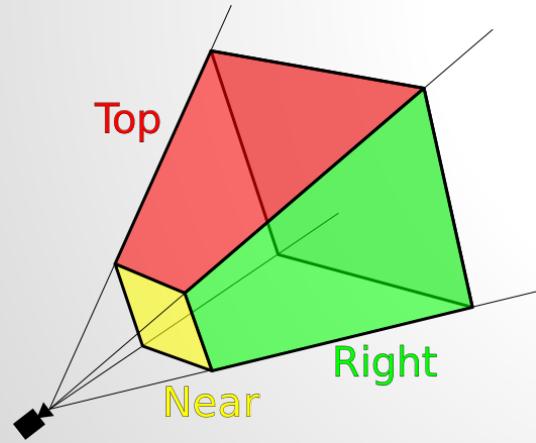
- Rasterizer
- Raycaster y derivados

# Rasterizer vs Raytracer



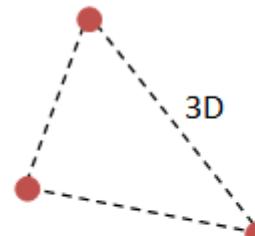
# Rasterizer

- Proyección de vértices y triángulos a espacio cámara

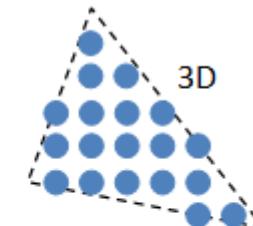


# Rasterizer

- Proyección de vértices y triángulos a espacio cámara
- Dibujado de pixeles sobre triángulos proyectados



Rasterizer



A primitive is formed by one or more vertices. Vertices are not aligned to the pixel-grid

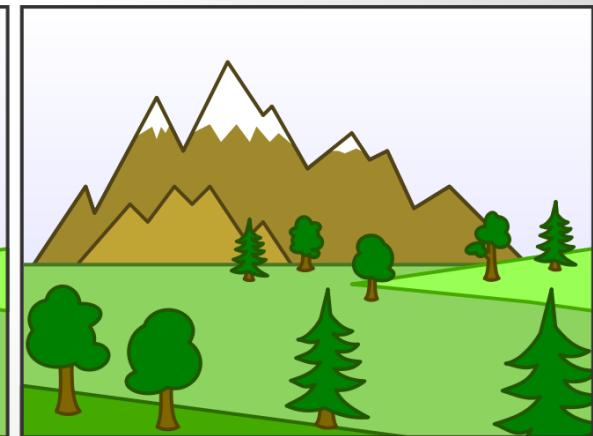
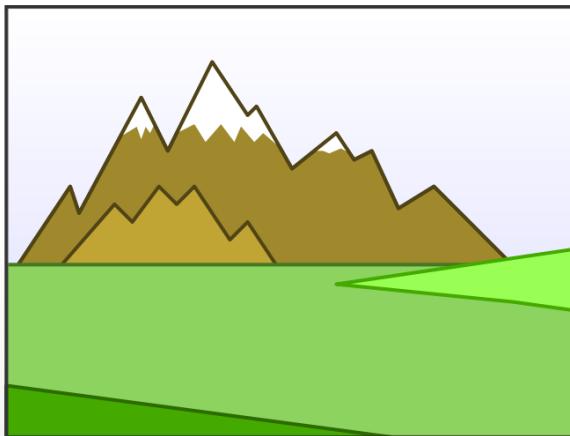
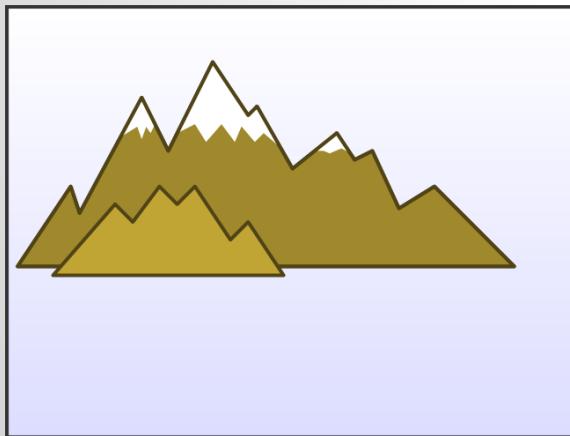
A fragment is aligned to the pixel-grid with a depth

# Rasterizer

- Proyección de vértices y triángulos a espacio cámara
- Dibujado de pixeles sobre triángulos proyectados
- Hidden surface removal

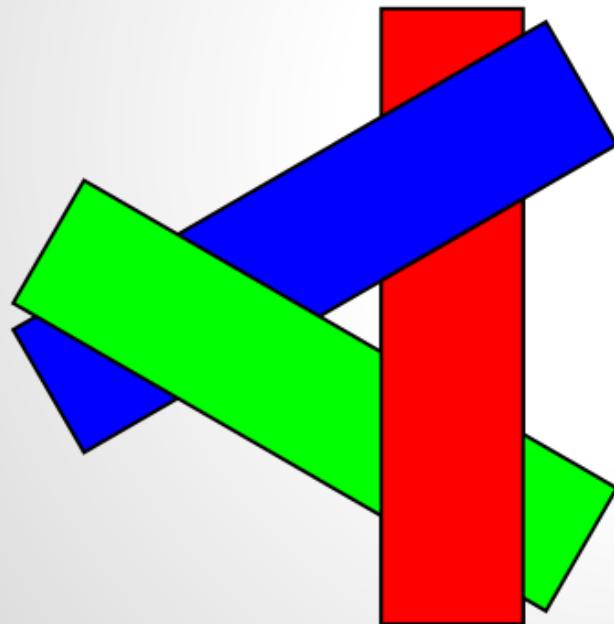
# HSR: Algoritmo del pintor

Ordena todos los polígonos y luego los dibuja en ese orden.



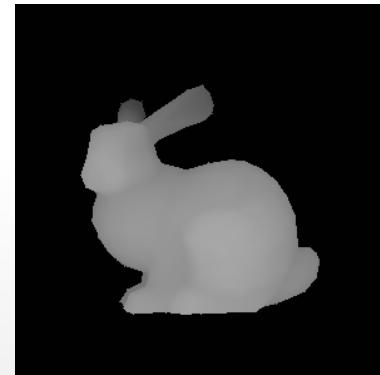
# HSR: Algoritmo del pintor

Tiene casos problemáticos!



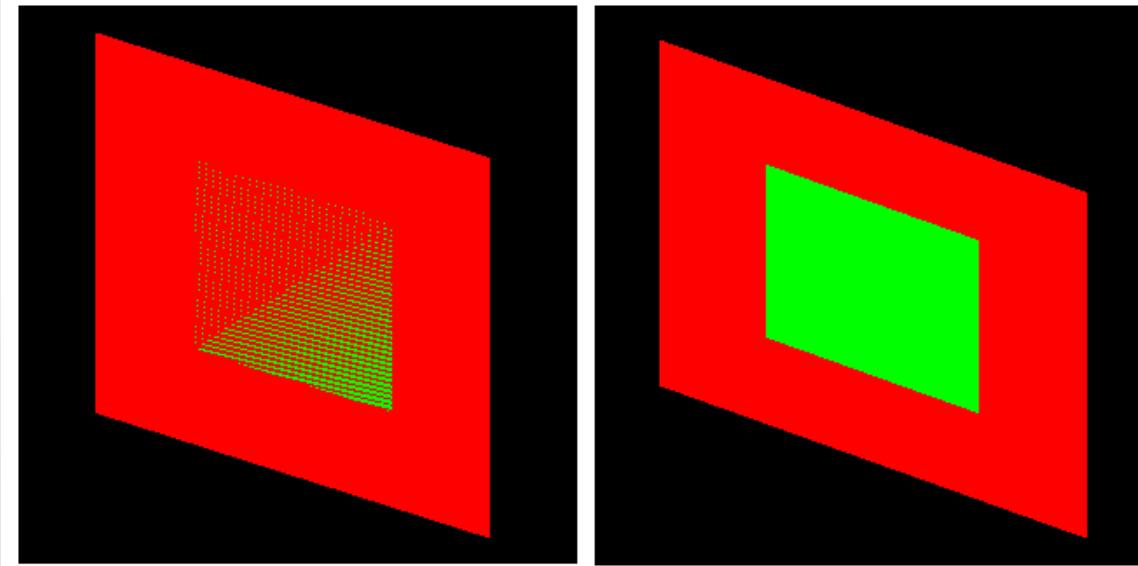
# HSR: Z-Buffer

- Cada pixel guarda su profundidad en un buffer.
- Al dibujar, el algoritmo decide si pintar o no en base al valor del buffer actual



# HSR: Z-Buffer

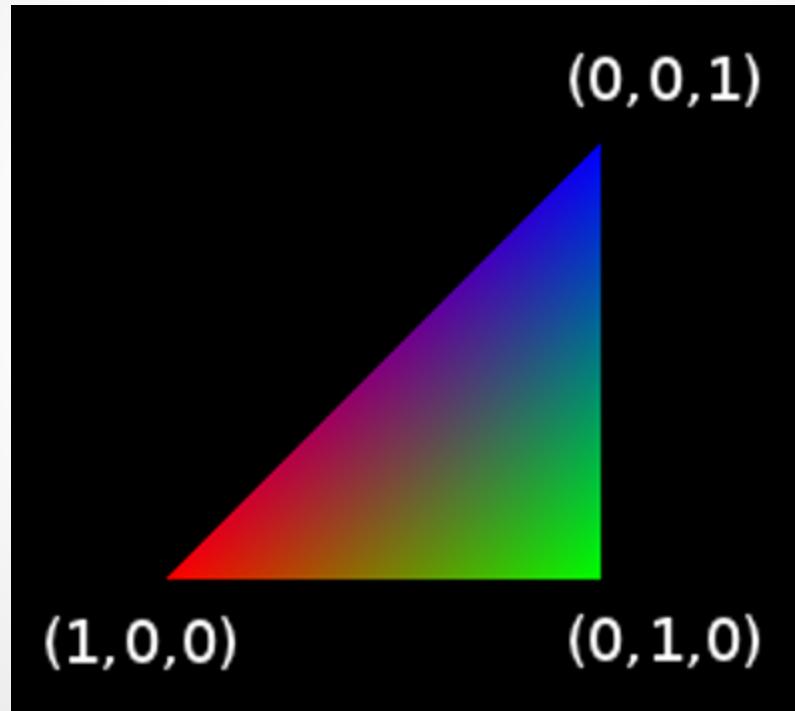
- La precisión del buffer es importante!



# Rasterizer

- Proyección de vértices y triángulos a espacio cámara
- Dibujado de pixeles sobre triángulos proyectados
- Hidden surface removal, Z buffer
- Interpolación de atributos de vértice a espacio triángulo

# Rasterizer



# Mipmaps

Segun la profundidad del pixel, se utilizan versiones reducidas de una textura

Esto acelera el lookup,  
aunque genera peso adicional  
en memoria.



# Rasterizer

Ventajas:

- Eficiente
- “Fácil” de implementar en GPU

Desventajas:

- Es difícil simular reflejos y fenómenos complejos.
- Modelo de cámara poco flexible

# Raycaster

- Discretizar la pantalla y disparar rayos por cada punto de la grilla.

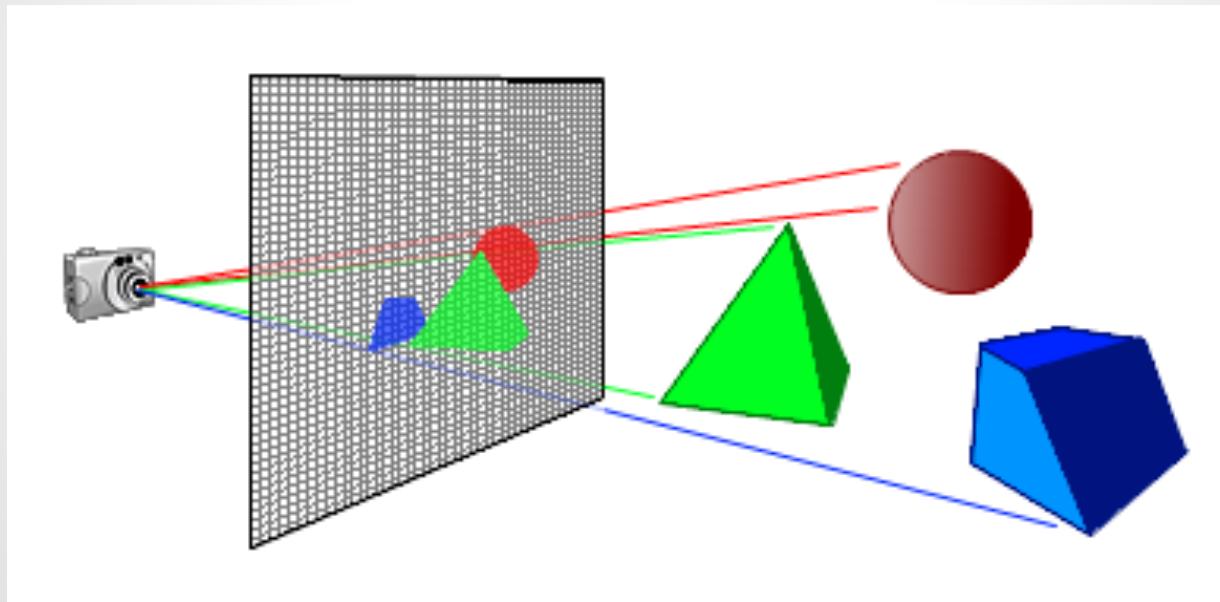
# Raycaster

- Discretizar la pantalla y disparar rayos por cada punto de la grilla.
- Se calculan puntos de colisión con objetos geométricos.

# Raycaster

- Discretizar la pantalla y disparar rayos por cada punto de la grilla.
- Se calculan puntos de colisión con objetos geométricos.
- Utilizando la información de la colisión, se puede decidir el color final de la muestra.

# Raycaster



# Raycaster

Ventajas:

- Muy extensible (raytracing, pathtracing)
- Arquitectura más sencilla (no hay HSR)
- Fácil de paralelizar
- Mayor precisión

Desventajas:

- Más complejo computacionalmente
- Búsqueda de intersecciones rayo-tríangulo son complejas y poco cache-friendly

# Raycasting: algoritmo general

```
for each pixel on the screen ps
    let v be the vector from our virtual eye through ps
    let dmin (the minimum distance to an element) be set to max float
    let emin (the element of the scene with the minimum distance) be set to NULL
    for each element in our scene e
        let p be the point of the intersection of v and e
        if p exists then
            let d be the distance of p from the eye
            if d < dmin then
                dmin := d;
                emin := e;
            end;
        end;
    end;
    if emin <> NULL then
        let ps be the color of emin
    else
        let ps be black
    end;
end.
```

# Raycasting: Cámara simple

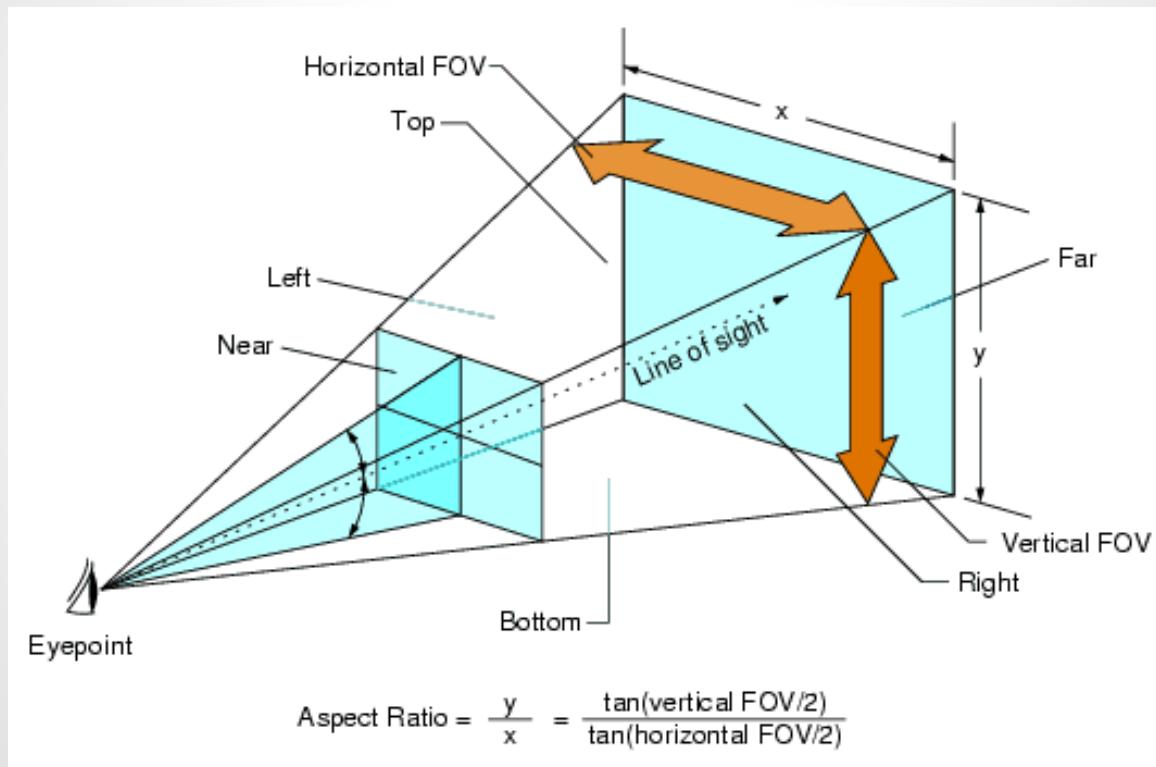
- El modelo más sencillo de cámara es la cámara pinhole.
- No usa lente, y la apertura es un único punto. Así, todos los rayos salen siempre del mismo punto, en distintas direcciones.

# Raycasting: Cámara simple

Se la define con:

- Posición ( $x, y, z$ )
- Orientación ( $x', y', z'$ )
- Normal ( $x'', y'', z''$ )
- Field of view (fov)

# Raycasting: Cámara simple



# Raycasting: Cámara simple

Ventajas:

- Simple de implementar

Desventajas:

- Introduce distorsiones hacia los extremos de la imagen.

# Caso extremo de field of view

