Proyección con WebGL

Katia Leal Algara
Web: http://gsyc.urjc.es/~katia/
Email: katia.leal@urjc.es

Dept. Teoría de la Señal y Comunicaciones y Sistemas Telemáticos y Computación (GSyC)
Escuela Superior De Ingeniería De Telecomunicación (ETSIT)
Universidad Rey Juan Carlos (URJC)

Matrices de proyección

- Hasta ahora hemos visto las matrices que afectan al modelo
  - Por ejemplo: translación, rotación, escalado, …
  - Todas ellas se engloban en la matriz OpenGL denominada "***modelview matrix***".
- Veremos que la proyección también se representa por una matriz.
- A continuación veremos cómo representar varias de las proyecciones más usuales con la "matriz de proyección".
- Al final, OpenGL combinará las matrices de modelo con la matriz de proyección.

Matrices de proyección

- Todos los vértices que pasamos a OpenGL son multiplicados por una cadena de matrices.
- A lo largo del pipeline, el vértice se encontrará con tres matrices:
  - La matriz del **modelview**.
  - La matriz de la **proyección**.
  - La matriz del **viewport**.
- El vértice puede ser eliminado si se encuentra fuera del área de visión (clipping).

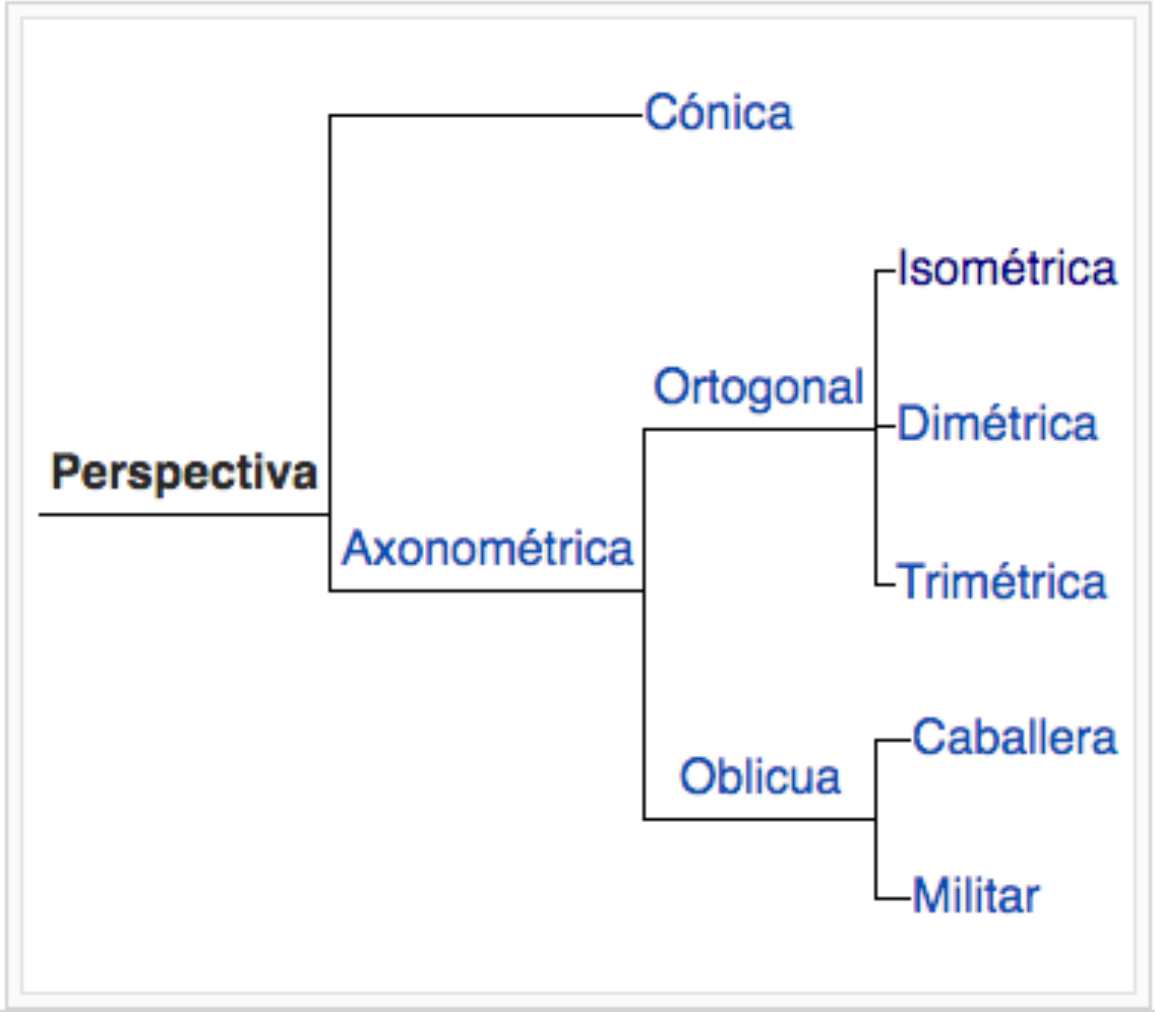
Matrices de proyección

- La matriz del **modelview** cambia las coordenadas de un vértice hacia las coordenadas de nuestra cámara (o las coordenadas de nuestro ojo).
- La matriz de **proyección** cambia las coordenadas de un vértice hacia las coordenadas de un volumen estandarizado (un cubo que se extiende desde -1 a 1).

Prespectiva Vs Proyección

- **Perspectiva:** es el arte de dibujar para recrear la profundidad y la posición relativa de los objetos comunes. En un dibujo, la perspectiva simula la profundidad
- **Proyección**: es una técnica de dibujo empleada para representar un objeto en una superficie.

Perspectiva

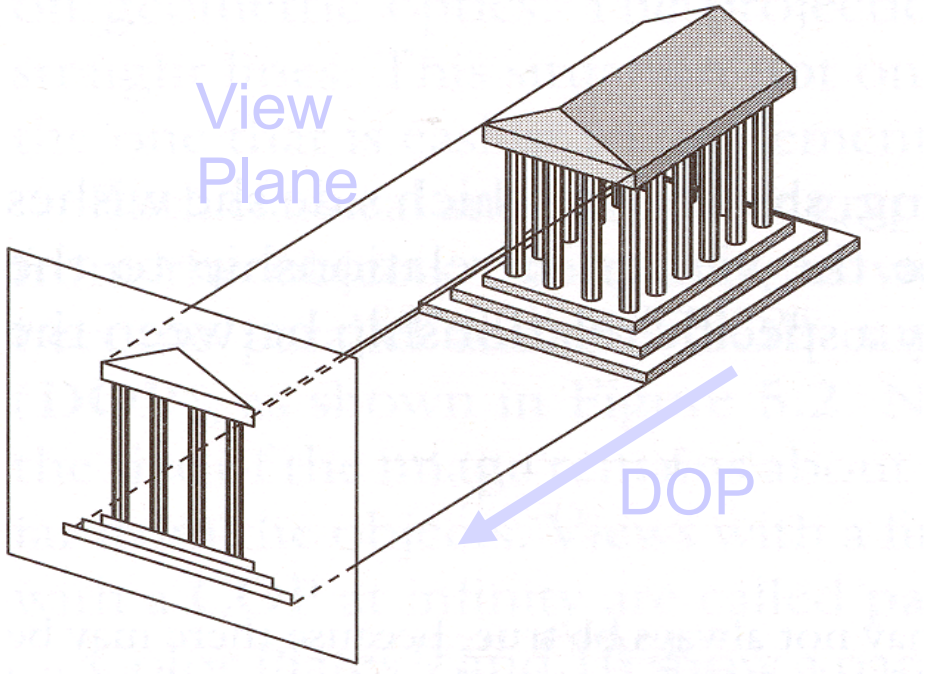


Proyección


Proyección gráfica
Proyección cónica
Proyección paralela
Proyección ortogonal
Proyección oblicua

Proyección paralela

- Trabaja situando el centro de proyección en el infinito.
- Todos los puntos tienen la misma dirección de proyección (DOP), son rectas paralelas.



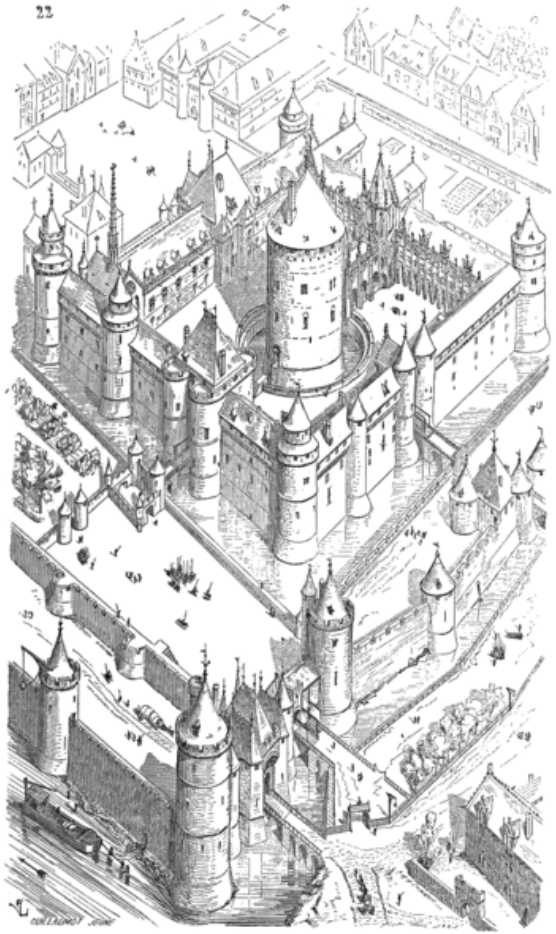
Tipos de proyección paralela

- **Ortogonal**: es aquella cuyas rectas proyectantes auxiliares son perpendiculares al plano de proyección (o a la recta de proyección), estableciéndose una relación entre todos los puntos del elemento proyectante con los proyectados.
- **Oblicua**: es aquella cuyas rectas proyectantes auxiliares son oblicuas al plano de proyección, estableciéndose una relación entre todos los puntos del elemento proyectante con los proyectados.

Proyección isométrica

- Representación visual de un objeto tridimensional en dos dimensiones, en la que los tres ejes ortogonales principales, al proyectarse, forman ángulos de 120º, y las dimensiones paralelas a dichos ejes se miden en una misma escala.
- La escala de medición es la misma en los tres ejes principales (x, y, z).
- Tiene la ventaja de permitir la representación a escala, y la desventaja de no reflejar la disminución aparente de tamaño -proporcional a la distancia- que percibe el ojo humano.
- Muy utilizado en juegos (antiguos y no tan antiguos).
- En el mundo de los juegos se la conoce por “perspectiva ¾”.

Proyección isométrica

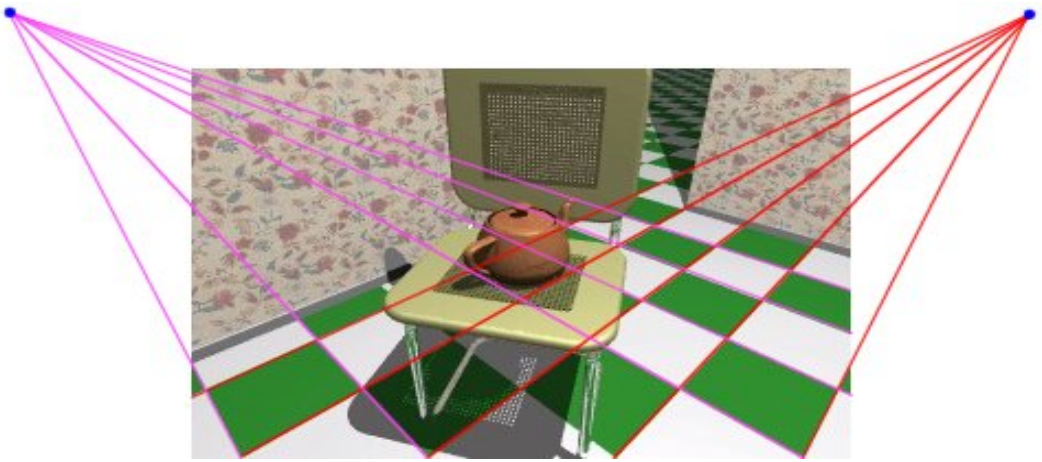


El castillo del Louvre, dibujo isométrico de Viollet-Le-Duc,(1814-1879). (Wikipedia)

Juego Syndicate (1993)

Proyección en prespectiva

- Ya se utilizaba en el renacimiento (Donatello, Brunelleschi y DaVinci).
- La idea es que los objetos lejanos los veamos más pequeños.



La revolución en los juegos

Doom

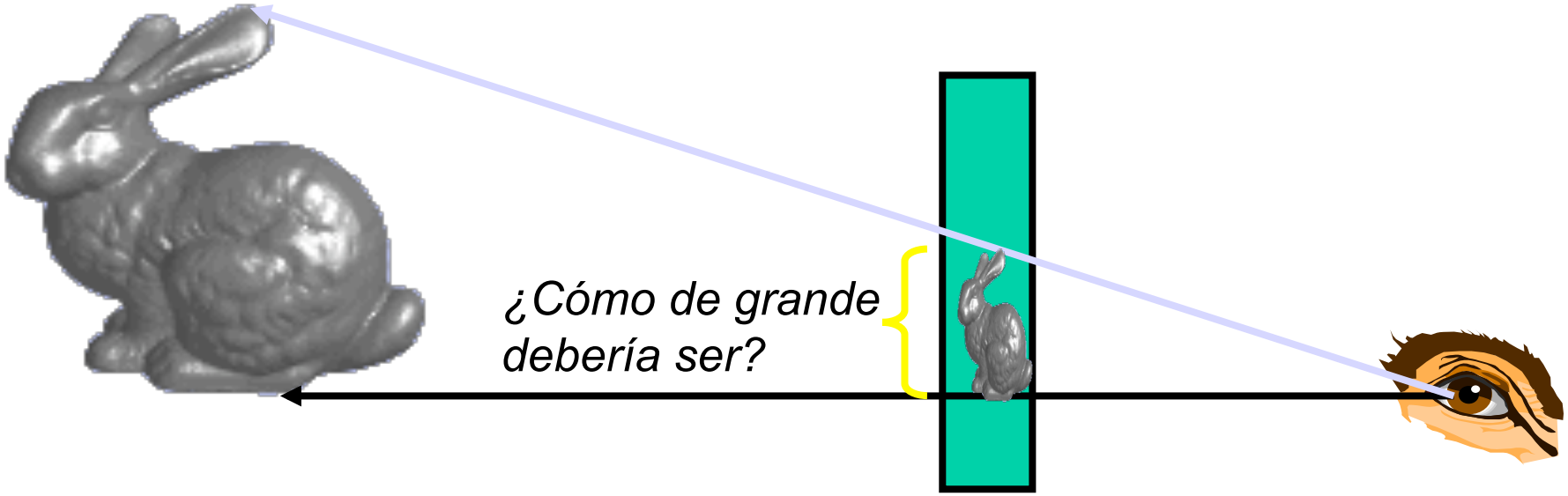
Wolfenstein 3D

Proyección de prespectiva

- Es el sistema de representación gráfico en donde un haz de rectas proyectantes que confluyen en un punto -el ojo del observador-, proyectan el cuerpo como una imagen sobre el plano auxiliar que intercepta dichas rectas.
- Este sistema de representación reproduce fielmente en un plano las imágenes del espacio, con un resultado muy similar a como lo percibimos realmente.

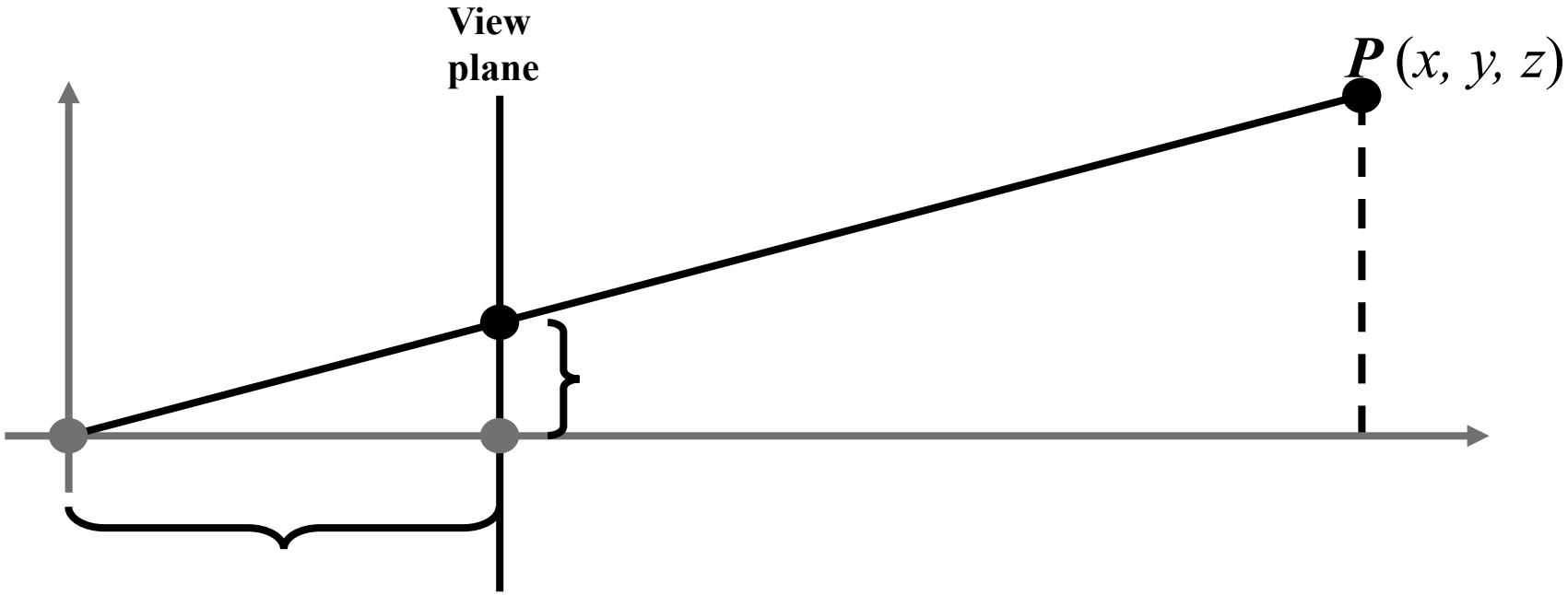
Proyección de prespectiva

- Cuando hacemos gráficos en 3D, pensamos que el monitor es cómo una ventana 2D a nuestro mundo en 3D.



Proyección de prespectiva

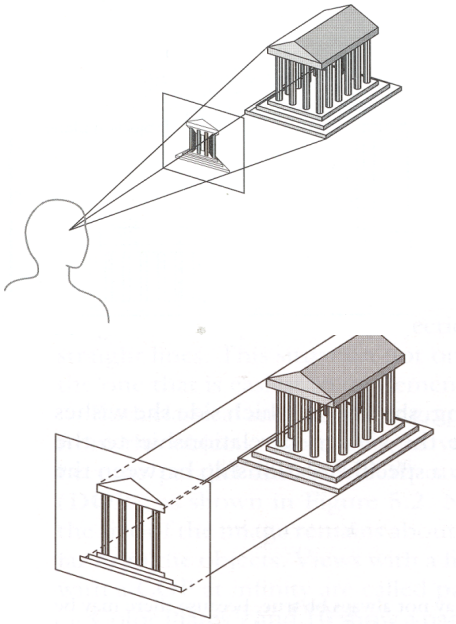
- Las matemáticas se basan en igualdad de triángulos.



Perspectiva Vs Paralela

- La proyección en perspectiva:
  - El tamaño depende de la distancia. Parece real.
  - Las distancias y los ángulos no se conservan (en general).
  - Las líneas paralelas no se representan como paralelas (en general)

- La proyección paralela:
  - Es ideal para realizar medidas.
  - Las líneas paralelas siguen siendo paralelas.
  - Los ángulos no tienen que conservarse.
  - No parece real.



Perspectiva

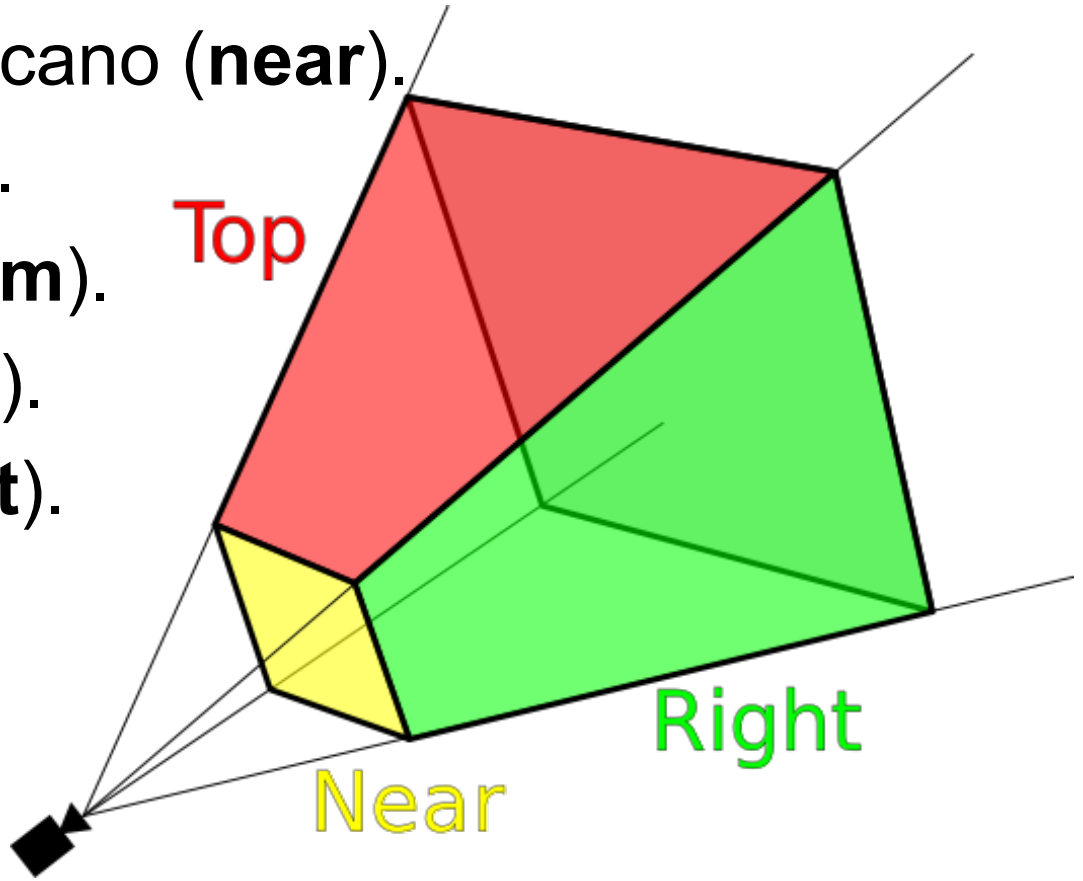
- Veamos cómo poner los parámetros de la proyección (p.e. field of view).
- Normalmente solemos utilizar una proyección de perspectiva:
  - Vemos los objetos lejanos más pequeños.
  - Se pierden en un punto central lejano.
  - Esto define lo que llamamos: ***view frustum***.
- Ya sabemos que existen otras proyecciones: **orthográfica**, **isométrica.**

View fustrum

- En gráficos 3D se usa el frustum de seis lados para simular el comportamiento de una cámara virtual.
- Así se consiguen resultados de planos semejantes a la realidad.
- En concreto, el software de renderizado usa ésta figura geométrica para calcular la parte del escenario virtual que ve la cámara u observador imaginario.

View fustrum

- Plano de recorte lejano (**far**).
- Plano de recorte cercano (**near**).
- Plano superior (**top**).
- Plano inferior (**bottom**).
- Plano izquierdo (**left**).
- Plano derecho (**right**).



Cómo funciona en OpenGL

- OpenGL tiene muchas matrices de transformaciones y estas funcionan mediante pilas.
- Las dos matrices más importantes son: `GL_MODELVIEW` y `GL_PROJECTION`.
- Los puntos se multiplican primero por la matriz de modelview y después por la de proyección.
  - `GL_MODELVIEW`: representa la transformación del objeto a la cámara.
  - `GL_PROJECTION`: representa la transformación de la cámara a la ventana (*world-coordinates to screen coordinates*).
  - `glViewport(0,0,w,h)`: es la transformación de la ventana a la pantalla física.

Cómo funciona en OpenGL

- **Syntax:** *void gl.viewport(x, y, width, height);*
- **Parameters**
  - **x**: a GLint specifying the horizontal coordinate for the lower left corner of the viewport origin. Default value: 0.
  - **y**: a GLint specifying the vertical coordinate for the lower left corner of the viewport origin. Default value: 0.
  - **width**: a non-negative Glsizei specifying the width of the viewport. Default value: width of the canvas.
  - **height**: a non-negative Glsizei specifying the height of the viewport. Default value: height of the canvas.
- **Return value:** none.
- **Errors thrown:** if either *width* or *height* is a negative value, a gl.INVALID_VALUE error is thrown.

Cómo lo hacemos con OpenGL

- Tenemos que tener en cuenta tres aspectos o pasos fundamentales:
  - **Situar la cámara**:
    - Define la matriz de model-view.
  - **Seleccionamos una lente**:
    - Define la matriz de proyección.
  - **Definimos el volumen a representar**:
    - Define el área que podemos evitar al dibujar (clipping).