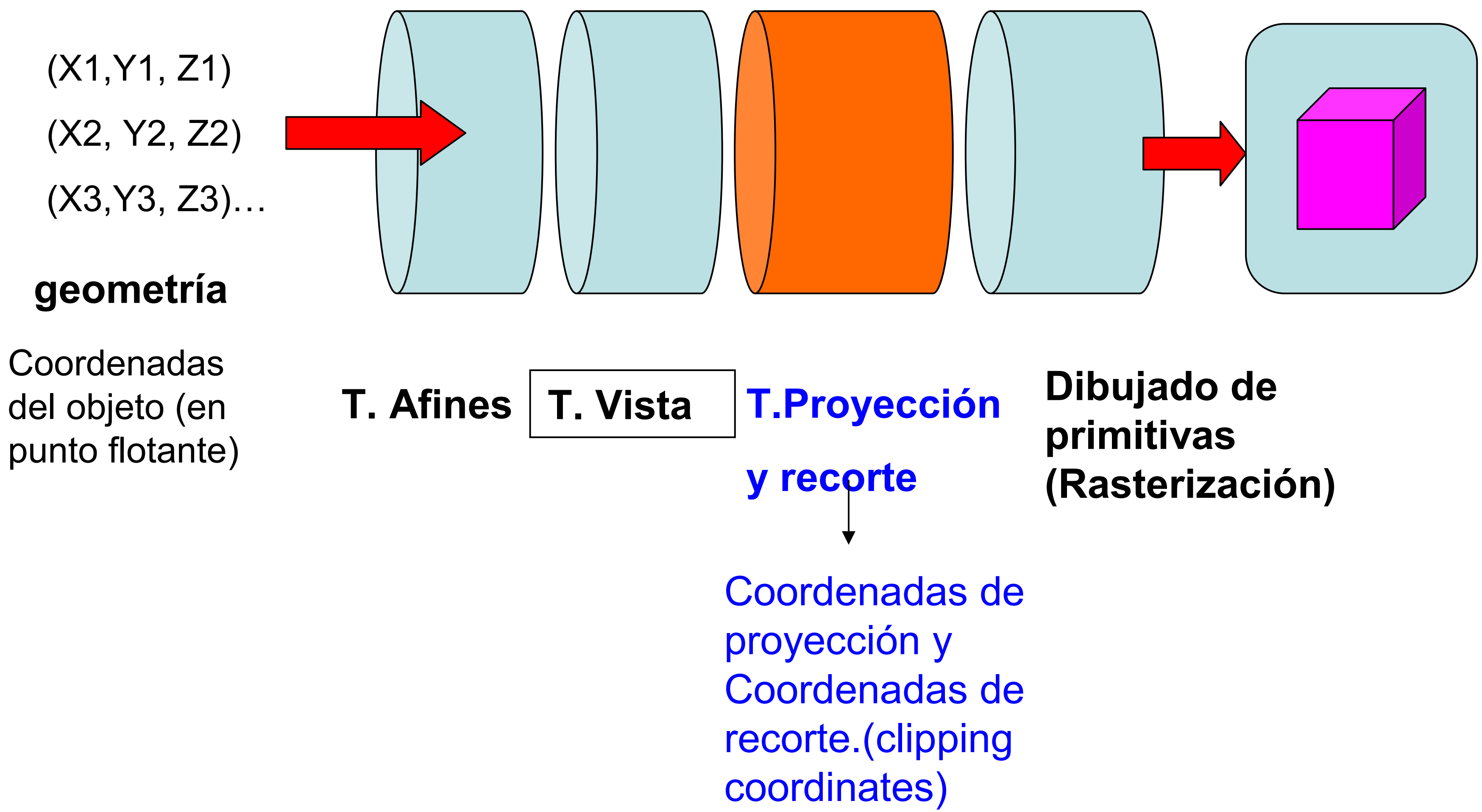
Proyecciones

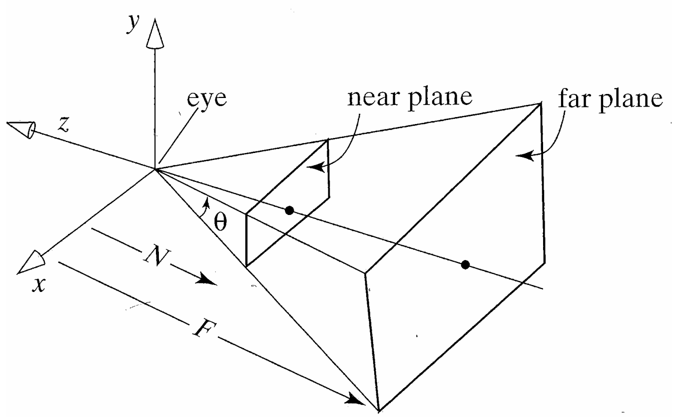


Volumen de visión

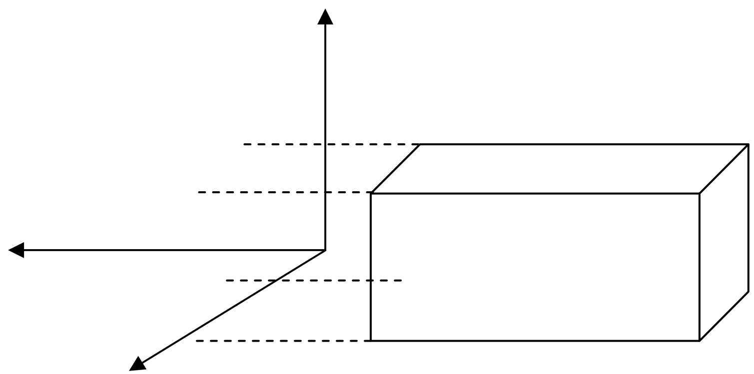
- En el sistema de coordenadas de vista, definimos el volumen de visión. La idea es definir un volumen que **determinará dos cosas**: la **porción del espacio que debe ser visualizada** y **cómo un objeto es proyectado** en el plano de proyección . La idea es una extensión del concepto de Ventana del mundo en 2D.

- La forma de este volumen tiene importancia, ya que determina el tipo de proyección a la que están sometidos los objetos de la escena.

- Hay básicamente dos tipos de volúmenes. Los de tipo paralelepípedo, que se asocian a una proyección ortográfica paralela y el volumen de tipo piramidal que se asocia a la proyección de perspectiva. Este último volumen también recibe el nombre de **frustrum**



Proyección en perspectiva.
Volumen = frustrum

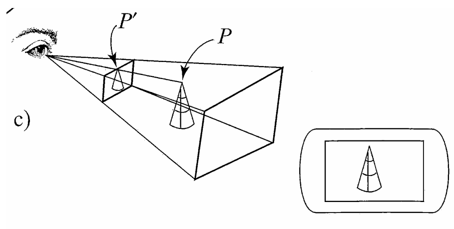


Proyección ortográfica.
Volumen = prisma

Volumen de visión(2)

- Sobre este volumen de visión se define el **plano de proyección** sobre el cual se proyectará toda la escena. La situación de este plano no es importante para la proyección y cada librería gráfica toma su propia decisión. **En opengl**, el plano de proyección está en el plano **Near** del frustrum. En caso de proyección paralela, no tiene importancia (generalmente se considera el plano XY).

c)



El volumen de visión se define respecto al sistema de coordenadas de vista. La única restricción es que para la proyección en perspectiva el plano de proyección no puede estar situado en el centro de proyección porque entonces se proyectaría la escena en un punto.

Modelo conceptual de la transformación de proyección

Transformación del volumen de visión en un cubo de dimensiones normalizadas

$-1 < x < 1$ $0<x<1$

$-1< y <1$ $0<y<1$

$-1 <z < 1$ $0<z<1$

Proyección (Baker)

→

Recorte usando un algoritmo eficiente (generalmente implementado por hardware)

Proyección (Foley)

→

Transformación a puerto de vista y División de perspectiva (si hace falta)

- Hay dos maneras de enfocar el problema:
- Hearn-Baker considera el problema tal que es la propia transformación de proyección la que convierte al volumen de visión en el volumen normalizado. Esto hace que la transformación que considera tenga parámetros de escalado y traslación que no tiene la transformación de proyección normal. Luego realiza el recorte y después aplica la transformación a puerto de vista 3D y la división de perspectiva si hace falta.
- Foley-Van Dam separa el problema considerando primero un conjunto de transformaciones afines que convierten el volumen de visión en el volumen canónico, tras el recorte, aplica la matriz de proyección clásica que se puede concatenar a la de transformación a puerto de vista 3D y luego realiza la división de perspectiva si hace falta

- **En cualquier caso el encadenamiento de matrices debe pararse para realizar el recorte (clipping)**

Modelo conceptual de la transformación de proyección (2)

Transformación del volumen de visión en un cubo de dimensiones normalizadas

$-1 < x < 1 \quad 0<x<1$

$-1< y <1 \quad 0<y<1$

$-1 <z < 1 \quad 0<z<1$

Coordenadas normalizadas de recorte

Proyección (Baker)

Recorte usando un algoritmo eficiente (generalmente implementado por hardware)

Proyección (Foley)

Transformación a puerto de vista y División de perspectiva (si hace falta)

Otras operaciones (cálculo sobre superficies [color, iluminación])

Ocultación de objetos

Coordenadas 3D de dispositivo

- En cualquier caso, la matriz de **modelview** se encadena con la matriz de la primera parte de la transformación de proyección (hasta el recorte)

- Vamos a dividir el resto del tema en estas partes:
- Primero clasificaremos los tipos de proyecciones y obtendremos la matriz de proyección generalizada. Esta matriz no es exactamente la que considera Hearn- Baker sino la de Foley-VanDam

- Segundo consideraremos superficialmente las transformaciones afines de la primera parte hasta el recorte

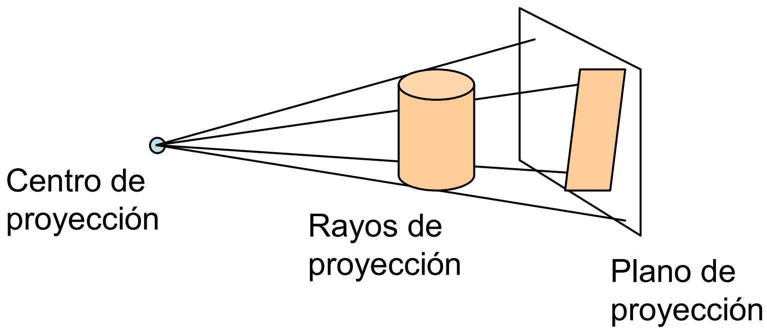
- Tercero consideraremos el recorte
- Cuarto consideraremos la transformación de puerto de vista y la división de perspectiva
- Quinto consideraremos la implementación de OpenGl de este problema

Taxonomía de las proyecciones

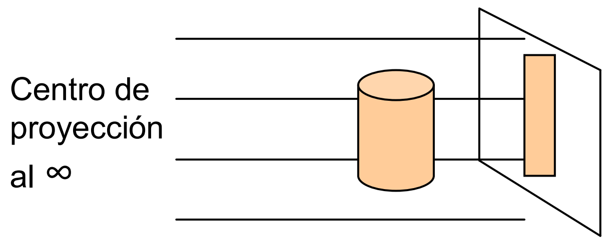
- **Proyección:** Transformación que convierte los puntos de n dimensiones en puntos de dimensiones menor que n (concretamente de 3 dimensiones a dos)
- Vamos a tratar las **proyecciones geométricas planas**
- **geométricas:** Los rayos de proyección son rectas
- **planas**: La superficie de proyección es un plano

- Hay proyecciones no geométricas (cartográficas) y también no planas (proyección sobre una cúpula o el sistema Omnimax de proyección de películas)

- Elementos de las proyecciones geométricas planas



Proyección perspectiva

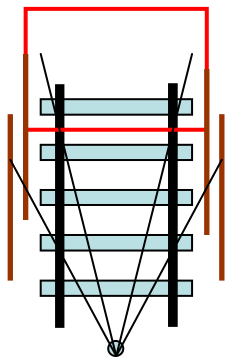


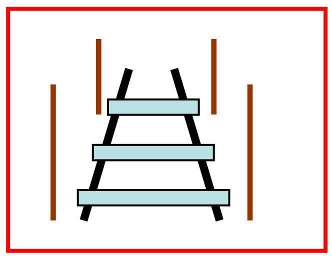
Proyección paralela

Taxonomía de las proyecciones (Proyecciones en perspectiva)

- En esta proyección el tamaño del objeto es inversamente proporcional a la distancia de dicho objeto al centro de proyección (perspective foreshortening)

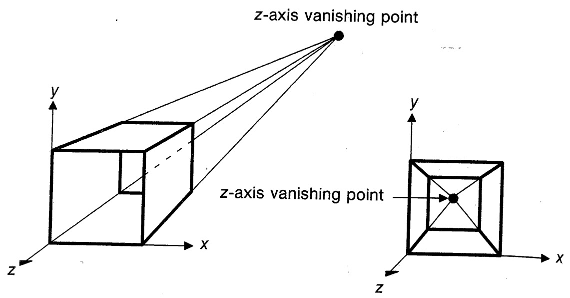
- Crea escenas realistas pero no es útil para medir distancias y los ángulos no se conservan en general. Tan solo se conservan los ángulos de líneas que son paralelas al plano de proyección.
- Las líneas paralelas que no lo son al plano de proyección no se conservan paralelas sino que son convergentes a puntos, denominados **puntos de fuga (vanishing point)**.



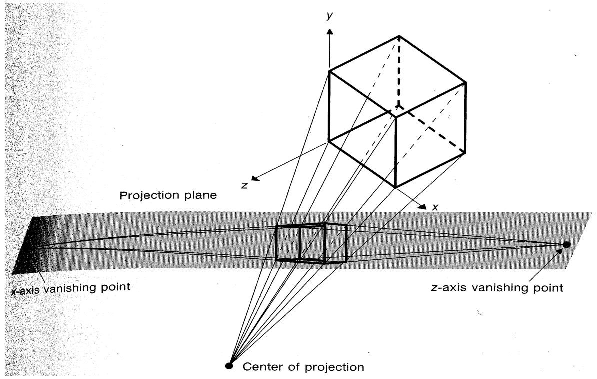


- Hay tres puntos de fuga principales que corresponden al número de ejes principales que cortan el plano de proyección. También clasificamos a las proyecciones en perspectiva como de un, dos o tres puntos de fuga.

Taxonomía de las proyecciones (Proyecciones en perspectiva 2)

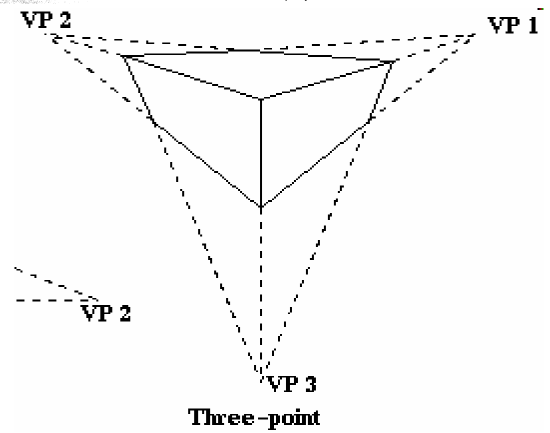


Proyección en perspectiva con un punto de fuga sobre el eje z



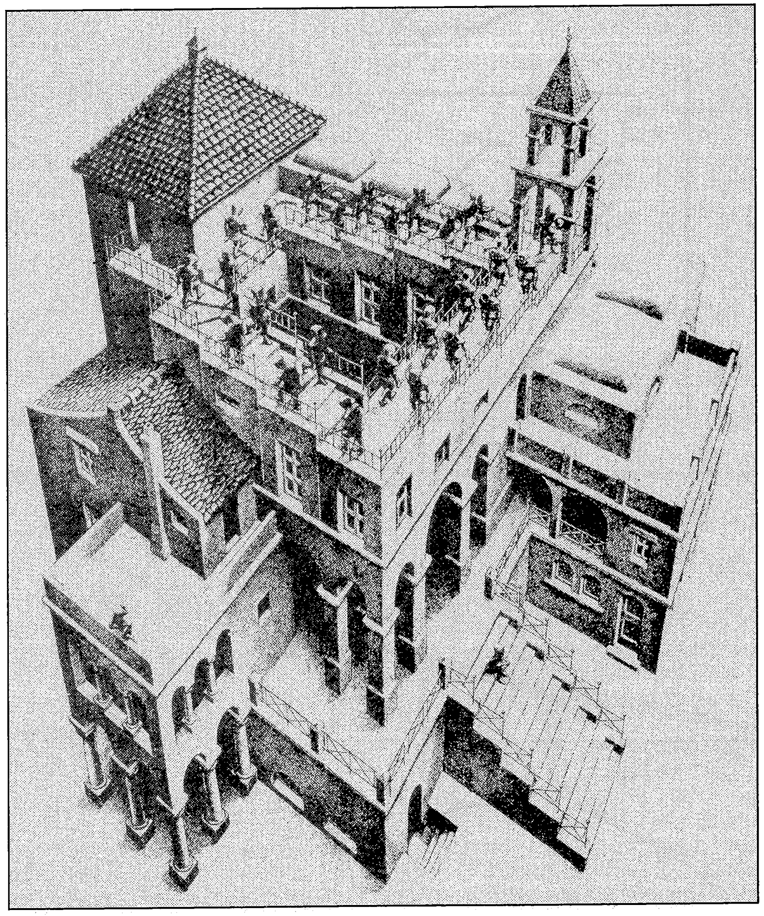
Proyección en perspectiva con dos puntos de fuga

(Fijémonos que son los ejes **x** y **z** los que intersectan con el plano de proyección)



Proyección en perspectiva con tres puntos de fuga

Taxonomía de las proyecciones (Proyecciones en perspectiva 3)



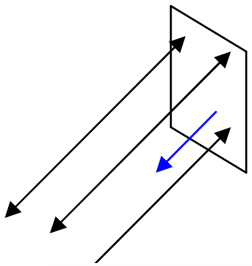
(Fijémonos que aparte de los ejes **x** y **z,** también la dirección de **y** está sometida al efecto de perspectiva)

Este grabado del artista belga Escher juega con el efecto de la perspectiva con tres puntos de fuga para crear una escena inconsistente.

Taxonomía de las proyecciones (Proyecciones paralelas)

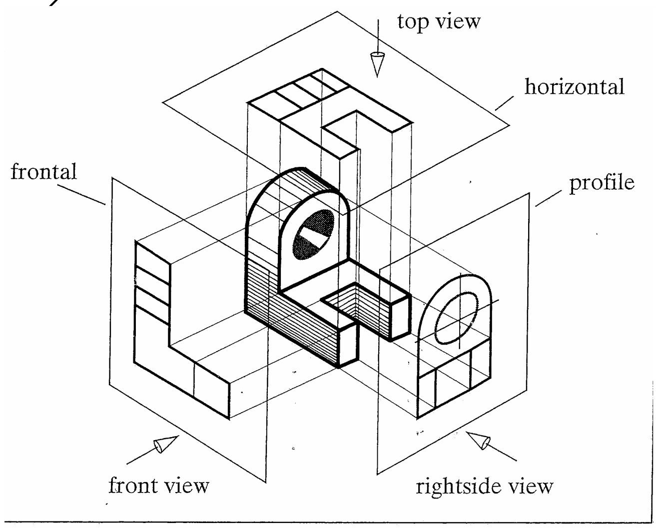
- En las Proyecciones Paralelas no aparece el efecto de empequeñecimiento con la distancia.
- Se clasifican atendiendo a la relación entre la dirección de los rayos de proyección y la normal al plano de proyección.

P.P. Ortográficas: Ambas direcciones son iguales.



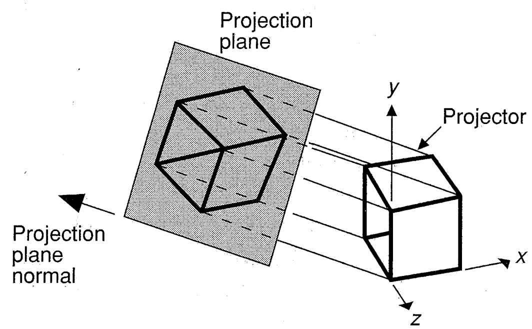
P.P. Ortográficas alineadas con los ejes de coordenadas. Hay tres: **Frontal, lateral y en planta**. En cada una de ellas el plano de proyección es perpendicular a un eje del sistema de coordenadas de vista.

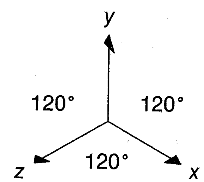
Son usadas en ingeniería y arquitectura porque en el dibujo se conservan los ángulos y la escala de distancias en cada eje es una constante.



Taxonomía de las proyecciones (Proyecciones paralelas 2)

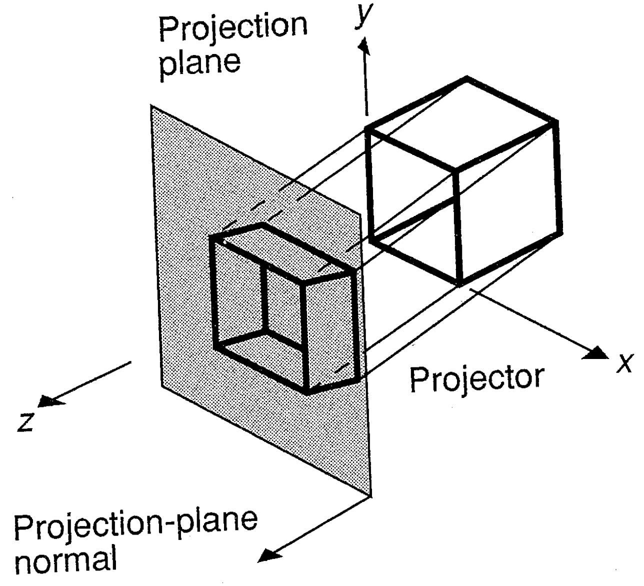
- **P.P. Ortográfica Axonométrica:** En ellas los planos de proyección no son normales a los ejes de coordenadas por lo que muestran varias caras del objeto a la vez. Se preserva el paralelismos de las líneas pero no así los ángulos. En general cada eje principal puede tener distinta escala.
- La más conocida es la **Isométrica** en la que el plano de proyección tiene ángulos iguales a los tres ejes del sistema de coordenadas de vista





Taxonomía de las proyecciones (Proyecciones paralelas 3)

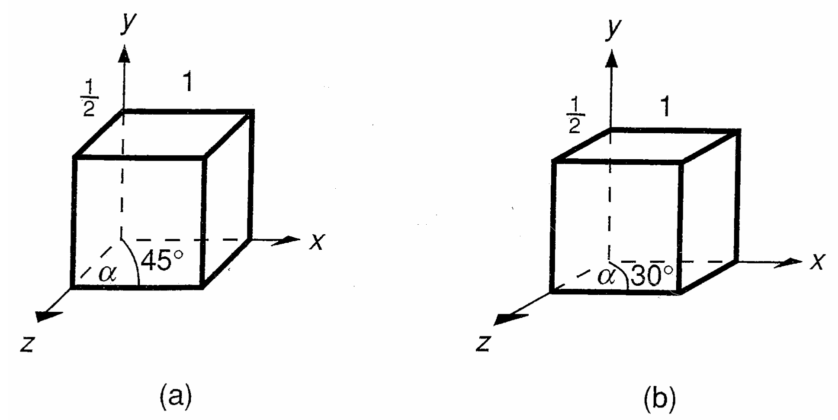
- **Proyecciones Paralelas Oblícuas:** En ellas, la normal del plano de proyección y la dirección de proyección son distintas. Generalmente el plano de proyección es normal a alguna de las direcciones principales de los ejes del sistema de coordenadas de vista.
- Las distancias y los ángulos se pueden medir en las caras paralelas al plano de proyección. En las otras caras del objeto se pueden medir distancias pero no ángulos.



Taxonomía de las proyecciones (Proyecciones paralelas 4)

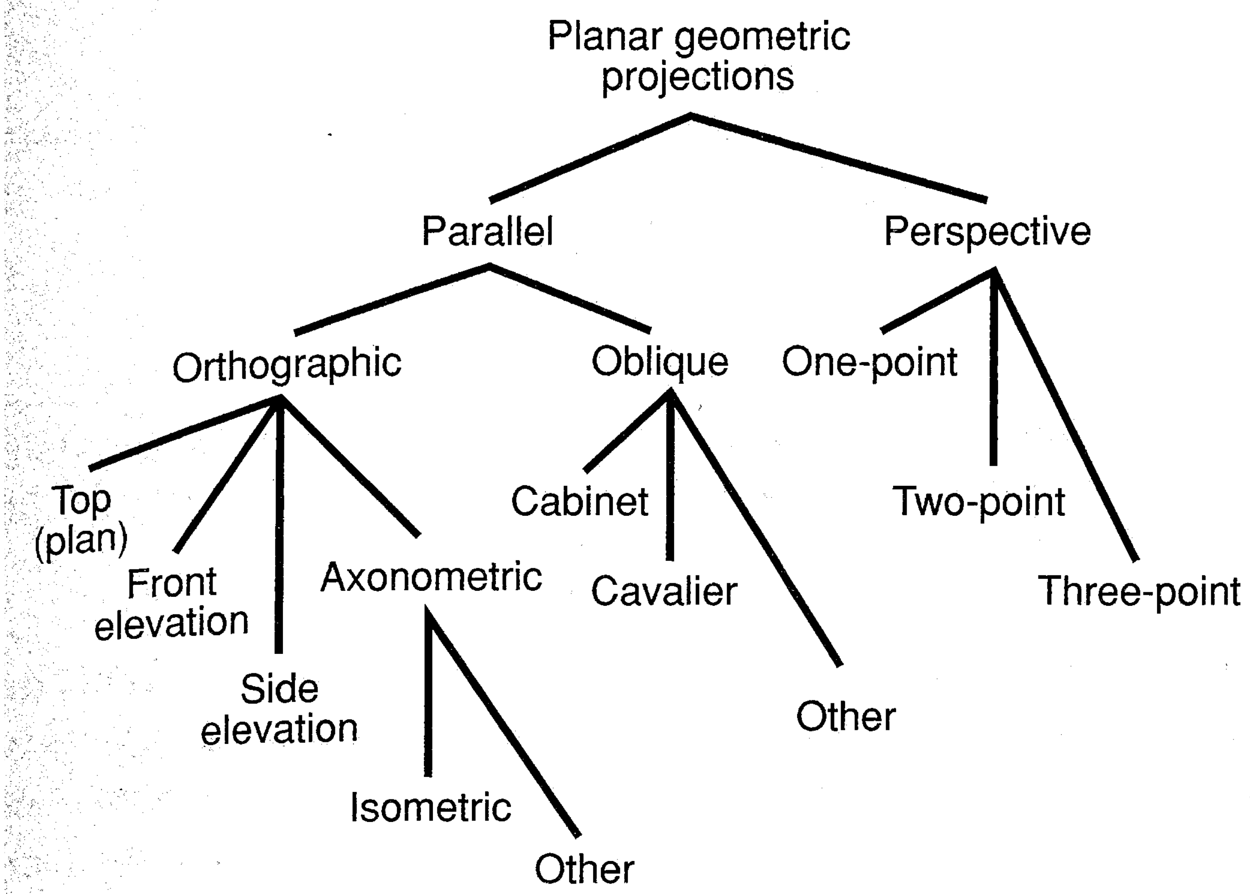
- Proyecciones Paralelas Oblícuas: **Proyección Caballera**. En ella los rayos de proyección forman un ángulo de incidencia con el plano de proyección de 45 grados.
- En ella las líneas perpendiculares al plano de proyección conservan la longitud.

- Proyecciones Paralelas Oblícuas: **Proyección de gabinete**. En ella los rayos de proyección forman un ángulo de incidencia con el plano de proyección de 63.4 grados.
- En ella las líneas perpendiculares al plano de proyección reducen a la mitad la longitud.



Proyecciones de gabinete con direcciones de proyección distintas
(Foley)

Taxonomía de las proyecciones

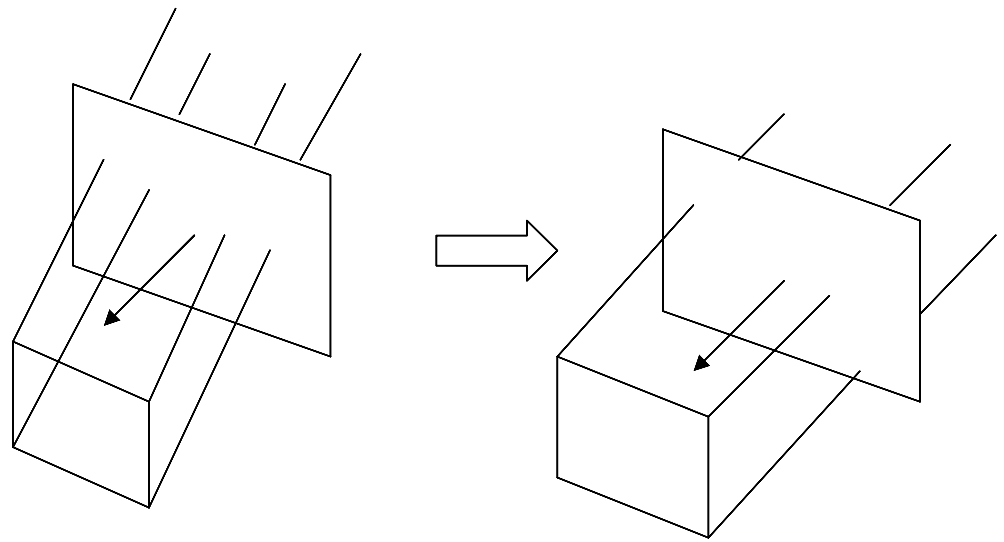


Transformación a volumen canónico (P. Paralela)

- En el caso más general suponemos que la **P. Paralela oblícua**

- **Los pasos son los siguientes:**

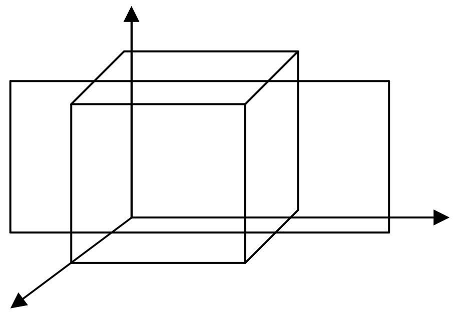
- **(Pasos 1 y 2 propuestos por Foley para llevar el S.C. V. al S. C. M.) OPCIONAL**
- **1º Traslación T(-d)**
- **2º Cambiar la orientación del S.C. Vista . (De este modo n = z , u = x v=y)**
- **3º Desplazamiento para que la dirección de proyección sea paralela al plano de proyección**



Si la proyección es
paralela ortográfica
esta transformación
no existe

Transformación a volumen canónico (P. Paralela) (2)

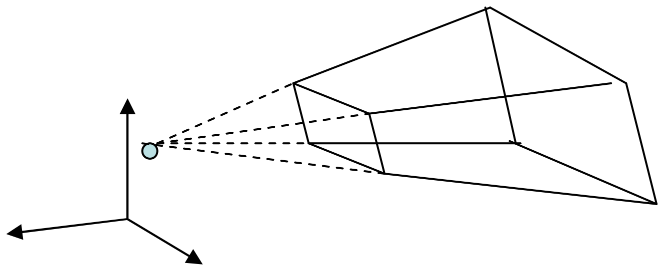
- **4º Traslación y escalado para transformar el prisma de visión en un cubo unidad**



- Se multiplican todas las matrices que provocan cada una de las transformaciones indicadas y se obtiene $\mathbf{N}_{par}$

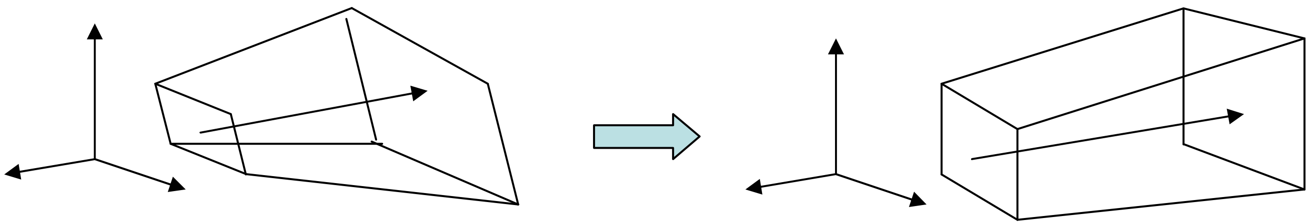
Transformación a volumen canónico (P. Perspectiva)

- En el caso más general el centro de proyección puede no estar en el origen del sistema de coordenadas (de vista) y tampoco tiene porque estar el plano de proyección normal a un eje de coordenadas (perspectiva de dos o tres puntos de fuga)



- **Los pasos son los siguientes:**

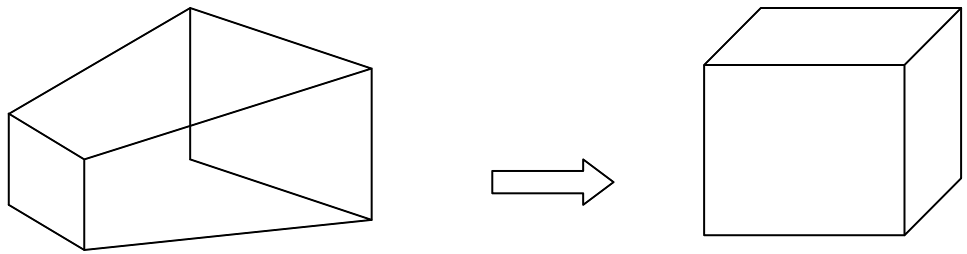
- **(Pasos 1 y 2 propuestos por Foley para llevar el S.C. V. al S. C. M.) OPCIONAL**
- **1º Traslación T(-d)**
- **2º Cambiar la orientación del S.C. Vista . (De este modo n = z , u = x v=y)**
- **3º Traslación del centro de proyección al origen**
- **4º Desplazamiento para conseguir que la línea central del volumen de visión tenga la dirección del eje z (o n en el caso del SCV)**



Transformación a volumen canónico (P. Perspectiva) (2)

- **5º Escalado para lograr una pirámide canónica (Algunos paquetes gráficos se paran aquí porque recortan sobre esta pirámide)**

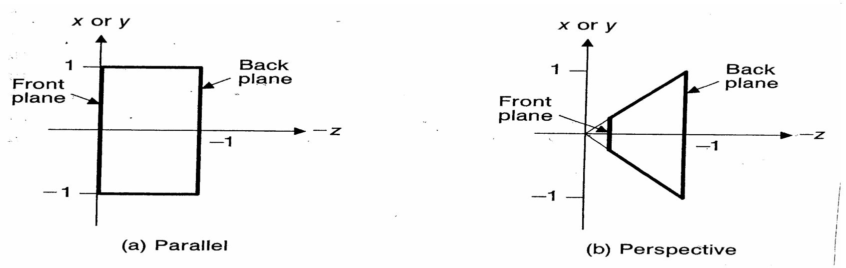
- **6º Desplazamiento para llevar la forma de pirámide truncada a la forma de cubo unidad**



- Esta transformación la denominamos $\mathbf{N}_{pers}$

Recorte

- El algoritmo de recorte está preparado para la forma del volumen canónico de un cubo unitario.
- El recorte se realiza siempre en **coordenadas homogéneas**, de otro modo puede haber problemas.

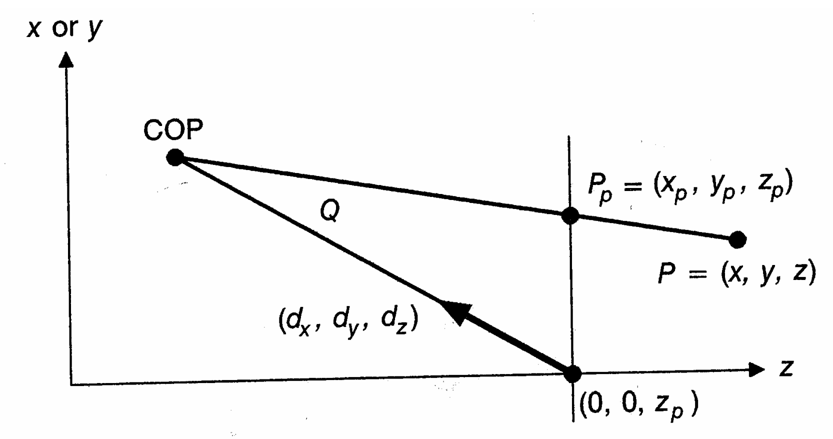


- La extensión de Cohen-Sutherland usa un código de 6 bits (arriba-abajo-derecha-izquierda-detrás-delante). Las condiciones de aceptación o rechazo triviales son las mismas que vimos en 2D. Pueden ser calculadas hasta seis intersecciones (una por cada cara).

- Igualmente Cyrus-Beck tiene una fácil adaptación a 3D. En este caso son calculados hasta seis valores del parámetro t. Los valores son clasificados y aceptados o descartados siguiendo los mismos criterios que los del algoritmo 2D hasta que quedan exactamente dos valores de t, que marcan los límites del segmento aceptado o determinan su rechazo. Para el volumen canónico unitario, las expresiones de ´cálculo del parámetro t son igual de sencillas que las vistas en la tabla para el caso 2D

Matriz general de proyección

- Como ya hemos dicho, los elementos de la proyección se sitúan respecto del sistema de coordenadas de vista.



- En la situación más general suponemos que el centro de proyección no está situado en el origen aunque el plano de proyección sí es perpendicular al eje **z** (**n** del SCV). Con lo que la P. Perspectiva solo puede ser de un punto de fuga

Desarrollo en pizarra

Matriz general de proyección (2)

$$M_{\text{general}} = \begin{bmatrix} 1 & 0 & -\frac{d_x}{d_z} & z_p\frac{d_x}{d_z} \\ 0 & 1 & -\frac{d_y}{d_z} & z_p\frac{d_y}{d_z} \\ 0 & 0 & -\frac{z_p}{Q\,d_z} & \frac{z_p^2}{Q\,d_z} + z_p \\ 0 & 0 & -\frac{1}{Q\,d_z} & \frac{z_p}{Q\,d_z} + 1 \end{bmatrix}.$$

Las matrices de proyección se distinguen de las matrices de transformaciones afines porque la última fila no es de la forma

“0 0 0 1”

La matriz general de proyección recoge los casos particulares de las proyecciones paralelas ortográficas y oblícuas y las perspectivas de un punto de fuga

	z_p	Q	(d_x	d_y	d_z)
M_{ort}	0	∞	0	0	-1
M_{per}	d	d	0	0	-1
M'_{per}	0	d	0	0	-1
Caballera	0	∞	cos α	sinα	-1
Gabinete	0	∞	cos α /2	sinα/2	-1

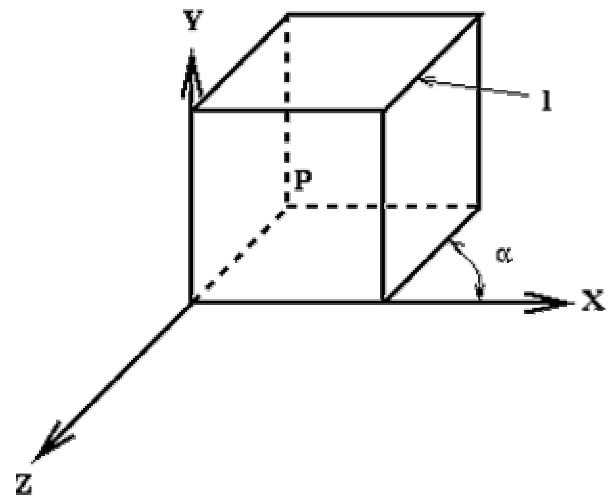
M_{per} ⟶ $$Mper = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

Matriz general de proyección (3)

- Para obtener la matriz de las proyecciones oblícuas a partir de la matriz general de proyección, asumimos que el plano xy es el plano de proyección y que el factor de perspectiva (empequeñecimiento con la distancia) es ***l*** y las líneas paralelas al eje z hacen un ángulo de α con el eje x. En estas condiciones la matriz de proyección es en realidad una matriz de desplazamiento:

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

- Y un cubo proyectado de lado ***l*** aparecería como:



- Si ***l =1*** tenemos la perspectiva caballera
- Si ***l =1/2*** tenemos la perspectiva de Gabinete

Transformación a puerto de vista

- Las coordenadas que se encuentran en el volumen normalizado de visión y sobre las que se realiza el recorte, se denominan coordenadas del **sistema de coordenadas normalizado de proyección** o también **coordenadas normalizadas de recorte**. Por tanto todos los valores de las coordenadas recortadas (y no recortadas) caen dentro de los valores $-1 \leq x \leq 1$, $-1 \leq y \leq 1$, $-1 \leq z \leq 1$ ($-1 \leq z \leq 0$)

- La trasformación a puerto de vista consiste en pasar de las coordenadas de este cubo a las llamadas **coordenadas 3D normalizadas de puerto de vista o coordenadas de pantalla 3D**. En ellas la transformación es igual que la 2D, salvo que se conserva la coordenada z que generalmente es escalada a un rango de valores entre 0 y 1 para luego ser utilizada en los **algoritmos de detección de superficies ocultas.**

- Básicamente la transformación consiste en un **escalado** sobre las coordenadas **x** e **y** y una **traslación**

- **Tras la transformación se dividen las coordenadas homogéneas por el factor w para obtener las coordenadas x, y proyectadas. Es lo que se conoce como división de perspectiva**

Frustrum y coordenada z

- La distancia entre los planos Near y Far del frustrum en la proyección de perspectiva tiene influencia en los valores que recibe la coordenada z de cada punto sometido a la proyección de perspectiva.
- Si nos fijamos en el diagrama, en él se representa en ordenadas el valor de la coordenada z

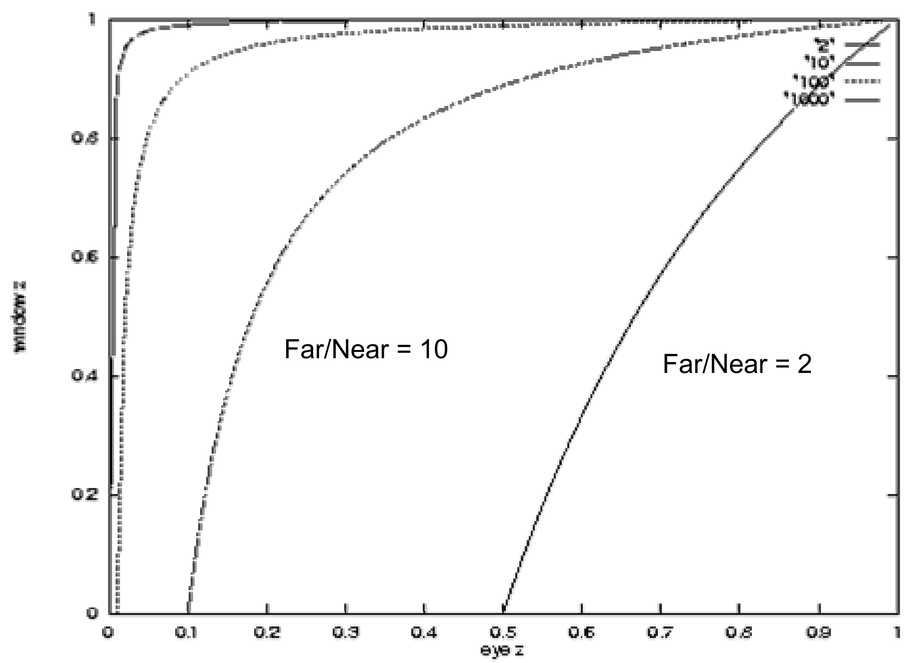


Figure 9. The relationship of window z (depth) to eye z for different far/near ratios

normalizado entre 0 y 1. Así un valor de 0 significa que dicha coordenada sitúa al objeto en el plano Near (lo más cercano posible al ojo. Un valor 1, lo sitúa en el plano Far (lo más lejano posible.

En cambio en el eje de abcisas aparece normalizada la distancia real del objeto al plano de proyección (plano Near), es decir, un objeto con eyez =0 está en el plano Near y un objeto con z=1 está en el plano Far.

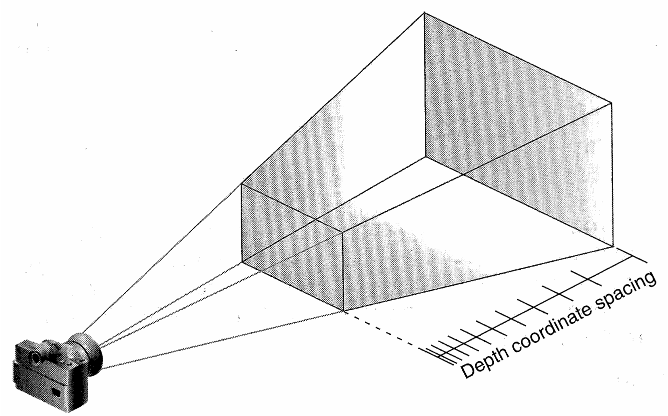
En el diagrama se muestran curvas para distintas relaciones de las distancias Far / Near

Frustrum y coordenada z (2)

- Si la proyección fuera ortográfica, la relación sería lineal. Es decir, conforme desplazamos el objeto de Near a Far, la coordenada z variaría de 0 a 1 uniformemente.
- Como la proyección es en perspectiva, esta variación no es lineal. Además contra mayor es la relación Far/Near, menos lineal es.
- Con una relación Far/Near = 1000, podemos observar que casi toda la precisión de z se concentra cerca del plano Near, con lo que al aumentar un poco la distancia, la coordenada z se satura. Esto se traduce en que en esta circunstancia se es capaz de discriminar qué objetos están delante de otros, si estamos cerca del Plano Near, mientras que a poco que nos alejemos de él, ya no podemos distinguir profundidad. Esto sería equivalente a la pérdida de profundidad de campo de un objetivo gran angular (sin su beneficioso efecto)

- Por el contrario una relación Far/Near = 2 no distinguirá qué objetos están delante o detrás hasta la mitad del frustrum, siendo a partir de ahí un comportamiento casi lineal

- Así la mejoría de este problema consiste en mover el plano Near alejándolo del ojo para hacer la relación Far/Near más pequeña. El efecto que tiene esto es que no hay tanta precisión de profundidad entre los objetos cercanos y algunos pueden desaparecer si se utiliza un algoritmo de ocultación de objetos basados en esta coordenada z



Proyecciones en la OpenGl

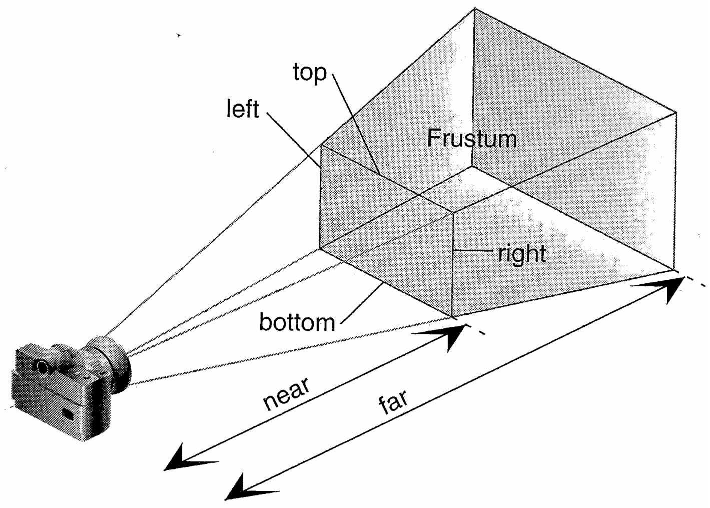
En OpenGl todo el proceso de proyección descrito está oculto. Por tanto no hay que especificar la aplicación de transformaciones para normalizar el cubo de visión, ni para el recorte. Lo único que hay que especificar es el volumen original de visión.

Para el caso de **proyección en perspectiva** usamos :

```
void glFrustrum( Gldouble left, Gldouble right, Gldouble bottom, Gldouble top, Gldouble near, Gldouble far);
```

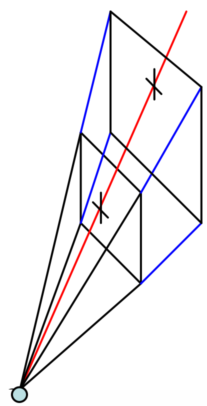
Donde todos los valores deben ser positivos.

La configuración normal que se emplea en las aplicaciones que simulan una vista tal y como la vería una persona, es la que se muestra en el dibujo. En él la línea que une el centro de proyección con el centro del plano Near (llamada línea de vista), también pasa por el centro del plano Far



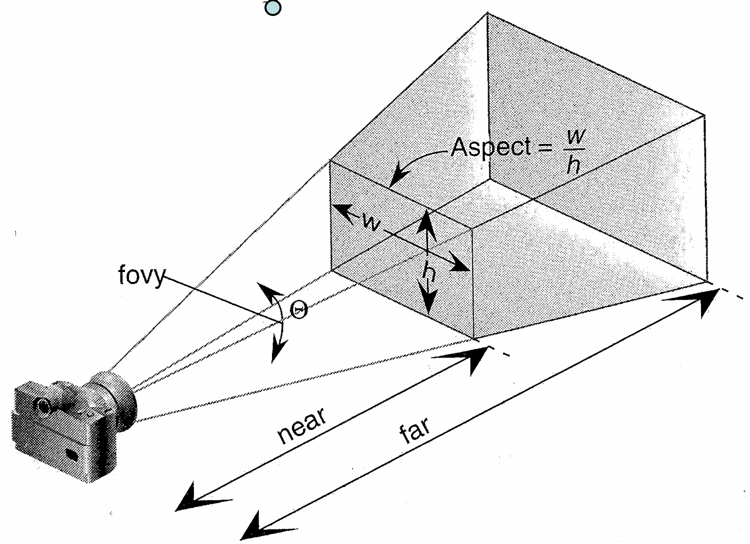
Proyecciones en la OpenGl (2)

- Sin embargo podemos establecer configuraciones donde el frustrum no es simétrico en los ejes X e Y respecto a la línea de vista.



- Si queremos usar siempre la configuración normal (frustrum simétrico respecto de la línea de visión), podemos usar la instrucción:

```
void gluPerspective(Gldouble fovy, Gldouble
    aspect, Gldouble Near, Gldouble Far);
```



- Los valores de `fovy` deben estar entre 0.0 y 180.0 grados sexagesimales.
- Si conocemos la distancia a la que se encuentra el objeto (***d***) y la altura del plano Far que contiene al objeto (***h***), el ángulo en radianes es:

- **Θ** = 2.0 * atan2(***h***/2,***d***)

Proyecciones en la OpenGl (3)

¿Qué unidades de medida se deben emplear para establecer la distancia entre los planos Near y Far? En otras palabras ¿Importan las unidades de medida en la configuración del volumen de visión?

NO. **La transformación de proyección se basa en relaciones entre medidas** y por tanto es adimensional. Esto quiere decir que la distancia entre los planos Near y Far localizados en por ej. 1.0 y 20.0 pueden ser metros, pulgadas, kilómetros o cualquier otra medida. Es cosa nuestra.

Transformación de **proyección ortográfica**.

Para esta proyección se usa la instrucción:

```
void glOrtho(Gldouble left, Gldouble right, Gldouble botton, Gldouble top, Gldouble near, Gldouble far);
```

Se crea una matriz cuya dirección de proyección es paralela al eje z y punto de vista mira hacia el eje z negativo.

Si estamos en el mundo 2D usamos

```
void glOrtho2D glOrtho(Gldouble left, Gldouble right, Gldouble botton, Gldouble top);
```

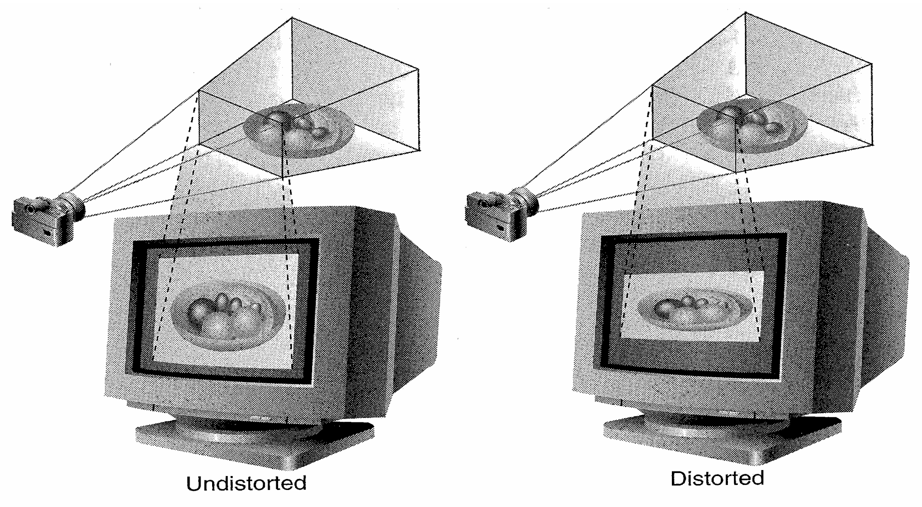
Esta instrucción es igual que la anterior solo que no realiza recorte sobre la coordenada z porque asume que tiene un valor entre -1.0 y 1.0

Proyecciones en la OpenGl (4)

Como ya vimos en el capítulo anterior, se define un puerto de vista 2D con

```
void glViewPort(Glint x, GLint y, GLsizei width, Glsizei height);
```

Si la razón entre altura y anchura del plano Near (plano de proyección) es diferente a la razón entre altura y anchura del puerto de vista, aparecerá una distorsión del dibujo.

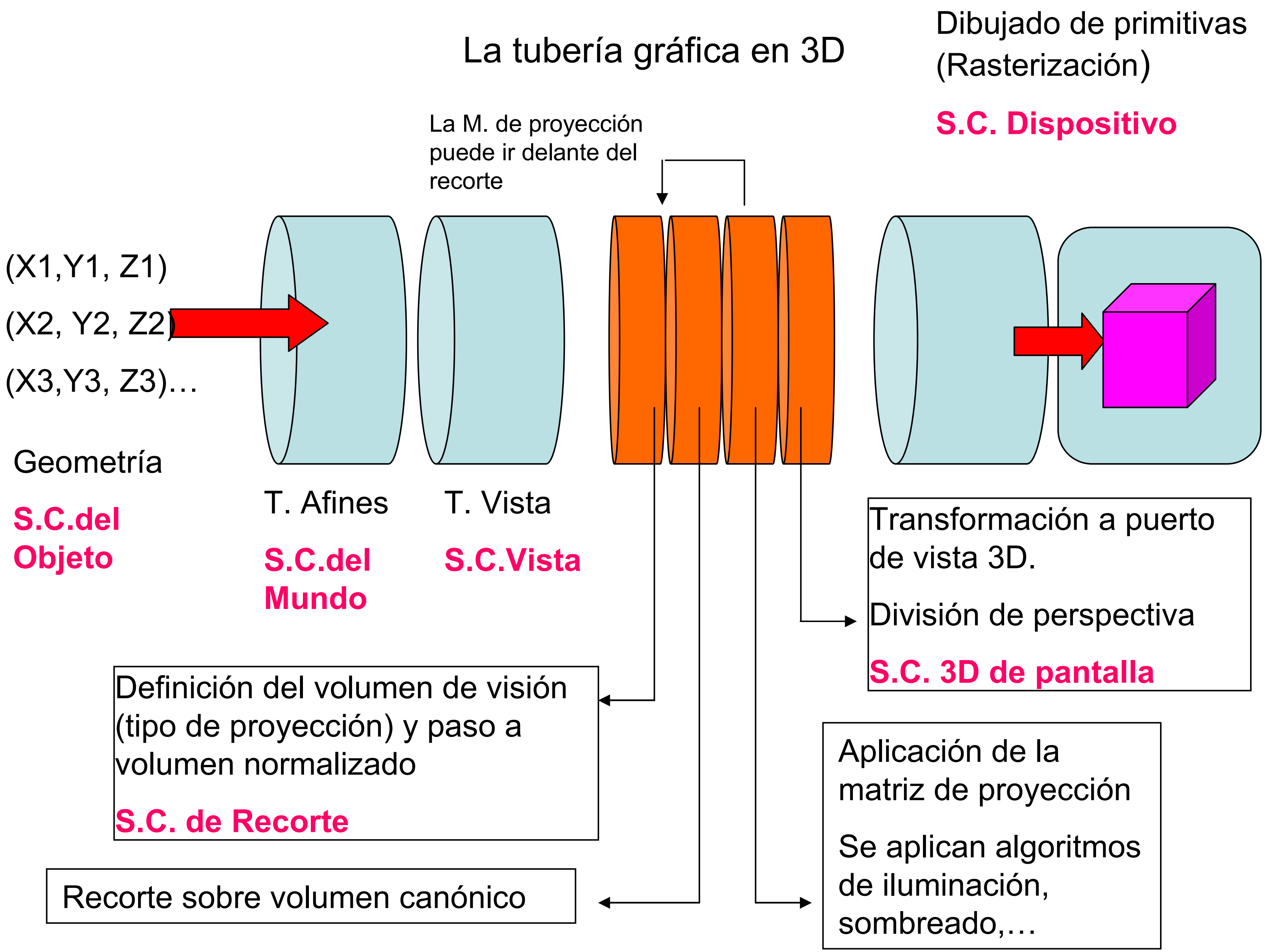


Es en esta fase donde se realiza la **división de perspectiva** y se almacena la coordenada z en el **buffer de profundidad o z-buffer** para ser usada por los algoritmos de ocultación. Podemos establecer el rango de los valores de z, que por defecto van desde z=0 para puntos sobre el plano Near hasta z = 1 para puntos situados en el plano Far.. El cambio de rango se hace con

```
void glDepthRange(int near, int far);
```

Ejemplo

- Ejecutamos un ejemplo de la página de Nate Robbins para proyecciones en perspectiva


La tubería gráfica en 3D
Dibujado de primitivas
(Rasterización)
S.C. Dispositivo
La M. de proyección
puede ir delante del
recorte
(X1,Y1, Z1)
(X2, Y2, Z2)
(X3,Y3, Z3)…
Geometría
S.C.del
Objeto
T. Afines
S.C.del
Mundo
T. Vista
S.C.Vista
Transformación a puerto
de vista 3D.
División de perspectiva
S.C. 3D de pantalla
Definición del volumen de visión
(tipo de proyección) y paso a
volumen normalizado
S.C. de Recorte
Aplicación de la
matriz de proyección
Se aplican algoritmos
de iluminación,
sombreado,…
Recorte sobre volumen canónico