Proyecciones. Cámara.

M.C. Hernandez
<mamen.hernandez@ehu.eus>
Joseba Makazaga
<joseba.makazaga@ehu.eus>

eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

UPV/EHU

Visión y proyección

- Nuestros ojos tienen su sistema de referencia y transforman el mundo 3D en 2D.
- Realizan una *proyección*
- La gestion de la cámara la dividimos en dos partes
    - Cambio de vista: tanto posición como orientación
    - Proyección: paso de 3D a 2D.
- Usaremos coordenadas homogéneas
- Se puede animar la cámara jugando con estos cambios

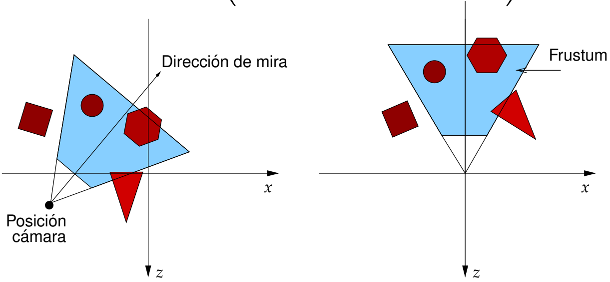
Cambio de Vista

- La posición y orientación de la cámara forma un sistema de referencia
- Necesitamos el origen $\mathbf{E}$
- Los tres ejes: $\mathbf{R}$, $\mathbf{U}$, $\mathbf{D}$. Vectores unitarios que indican dirección horizontal, vertical y profundidad, o x_c, y_c, z_c que forman un sistema dextrogiro, donde $\mathbf{R}x\mathbf{U} = \mathbf{D}$, por tanto $\mathbf{D}$ mira hacia atras.
- Los tres vectores han de formar un sistema de referencia ortogonal
  - Unitarios
  - Perpendiculares entre si

Cambio de vista

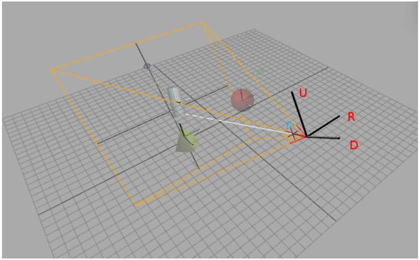
- El cambio de vista posiciona la cámara en el origen, y hace que los vectores $\{\mathbf{R}, \mathbf{U}, \mathbf{D}\}$ de la cámara pasen a ser los ejes x, y y z del sistema de referencia con el que se representa la escena.
- *Rigid body transformation*: cambio de vista $\mathbf{M}_b$:

$$\mathbf{M}_b = \begin{pmatrix} r_x & r_y & r_z & -\mathbf{R}\cdot\mathbf{E} \\ u_x & u_y & u_z & -\mathbf{U}\cdot\mathbf{E} \\ d_x & d_y & d_z & -\mathbf{D}\cdot\mathbf{E} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



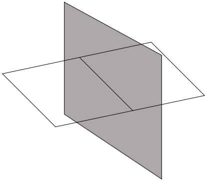
Cambio de vista

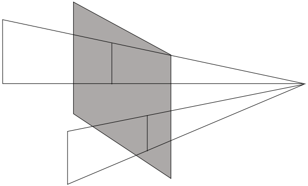
- Necesitamos $\{\mathbf{R}, \mathbf{U}, \mathbf{D}\}$ y $\mathbf{E}$!
- Posición de la cámara: $\mathbf{E}$
- Punto al que mira $\Rightarrow$ $\mathbf{D}$
- ¿$\mathbf{R}$ y $\mathbf{U}$? Three.js: camera.lookAt(point)



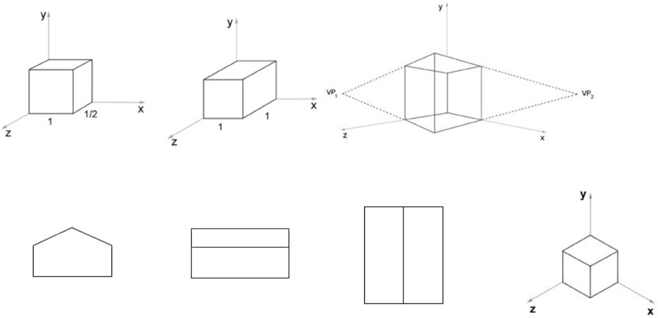
Proyecciones

- Hay que representar una escena 3D en una pantalla 2D
  - Proyección
- Hay muchos tipos de proyecciones:
  - Proyecciones paralelas
    - Plano de proyección
    - Dirección de la proyección (lo normal es que sea perpendicular al plano de proyección)
  - Proyección perspectiva
    - Plano de proyección
    - Foco de la proyección
    - Los objetos lejanos se ven menores





Tipos de proyección



- Cabinet, Cavalier, perspectiva de 2 puntos de proyección
  Alzado, planta, perfil, isometrico

http://www.mtsu.edu/ csjudy/planeview3D/tutorial.html

Tipos de proyección

- Utilizaremos dos tipos de proyección
  - Proyección ortográfica (paralela)
    - Arquitectura
    - CAD/CAM
  - Proyección perspectiva
    - Nuestra visión
    - Es la más utilizada

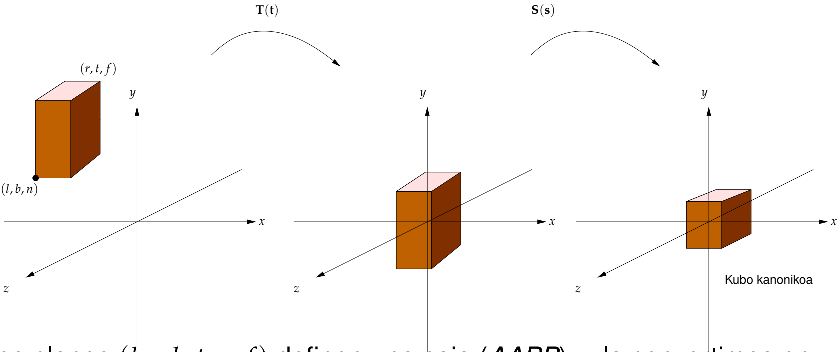
Proyeción ortográfica

- La proyección de rectas paralelas vuelven a ser paralelas
- Por ejemplo, para obtener una proyección en el plano $z = 0$:

$$\mathbf{P}_O = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Problema: no tiene inversa
- los puntos (x, y, z) y $(x, y, -z)$ tienen la misma proyección.

Proyección ortográfica



- Los planos (l, r, b, t, n, f) definen una caja (*AABB*), y la convertimos en el cubo canónico

$$\mathbf{M}_{p_o} = \mathbf{S}(\mathbf{s})\mathbf{T}(\mathbf{t}) = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2}{t-b} & 0 & 0 \\ 0 & 0 & \frac{-2}{n-f} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -\frac{r+l}{2} \\ 0 & 1 & 0 & -\frac{t+b}{2} \\ 0 & 0 & 1 & -\frac{f+n}{2} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Proyección ortográfica

$$\mathbf{M}_{p_o} = \mathbf{S}(\mathbf{s})\mathbf{T}(\mathbf{t}) = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{-2}{n-f} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- $\mathbf{t} = (\frac{-(r+l)}{2}, \frac{-(t+b)}{2}, \frac{-(f+n)}{2})$
- $\mathbf{s} = (\frac{2}{r-l}, \frac{2}{t-b}, \frac{-2}{n-f})$

- $\mathbf{M}_{p_o}^{-1} = \mathbf{T}(-\mathbf{t})\mathbf{S}(\frac{r-l}{2}, \frac{t-b}{2}, \frac{f-n}{2})$

Proyección ortográfica

- A tener en cuenta
  - $n > f$, ya que estamos viendo la parte negativa del eje z
- El cubo canónico
  - Se define mediante dos puntos: $(-1, -1, -1)$ y $(1, 1, 1)$
  - Define el *volumen canónico de la vista* ("canonical view volume")
  - A las coordenas internas al volumen se les denomina *coordenadas normalizadas de dispositivo* ("normalized device coordinates")
  - El recorte ("clipping") es muy rápido en este volumen (hardware)
  - Sistema levogiro!

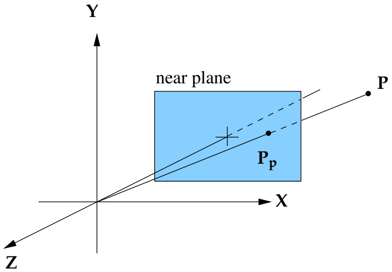
Proyección ortográfica en Three.js

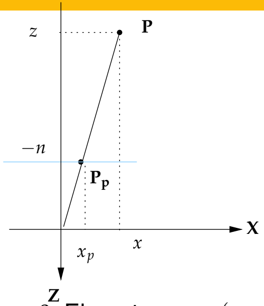
makeOrthographic (
left,right,
top, bottom,
near, far)

Proyección perspectiva

- La proyección de rectas paralelas no tiene por qué dar como resultado rectas paralelas
- Las proyecciones se realizan hacia un punto o foco
- Los objetos lejanos se ven más pequeños
- Es el tipo de proyección más utilizado

Proyección perspectiva





- Proyectaremos sobre el plano $z = -n$, donde $n > 0$. El punto $\mathbf{p}_p = (x_p, y_p, z_p)^T$ es la proyección del punto $\mathbf{p} = (x, y, z)^T$
- Sistema de referencia del observador
- Ley de triángulos similares:

$$\frac{x_p}{-n} = \frac{x}{z} \Rightarrow x_p = -n\frac{x}{z}$$

Proyección perspectiva

- $x_p = n\frac{x}{-z},\;\; y_p = n\frac{y}{-z},\;\; z_p = -n$

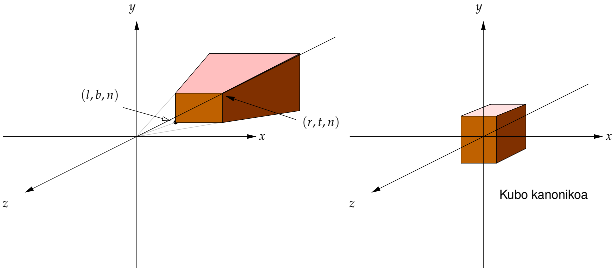
$$\mathbf{M}_p = \begin{pmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n & 0 \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

$$\mathbf{p}_p = \mathbf{M}_p\mathbf{p} = \begin{pmatrix} nx \\ ny \\ nz \\ -z \end{pmatrix} = \begin{pmatrix} n\frac{x}{-z} \\ n\frac{y}{-z} \\ -n \\ 1 \end{pmatrix}$$

- Problema: De nuevo, $\mathbf{M}_p$ no tiene inversa

Proyección perspectiva

- Mapearemos la piramide cortada al volumen canónico de vista



- Para definir la pirámide: (l, r, b, t, n, f)

Proyección perspectiva

- Tras la proyección las coordenadas proyectadas las tenemos que normalizar, para que la cuarta coordenada sea 1
- Si el punto está dentro del volumen de visión, las coordenadas normalizadas estan en $[-1, 1]$, ya que están dentro del cubo canónico. Así, si $\mathbf{p}_p = \mathbf{M}_p\mathbf{p}$, entonces $-1 \leq \frac{i_p}{w_p} \leq +1 \quad \forall i \in \{x, y, z\}$

  donde

$$\mathbf{M}_p = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Obtención de la matriz de proyección

¿De donde viene $\mathbf{M}_p$?

$$\mathbf{M}_p = \begin{pmatrix} m_{xx} & m_{xy} & m_{xz} & m_{xw} \\ m_{yx} & m_{yy} & m_{yz} & m_{yw} \\ m_{zx} & m_{zy} & m_{zz} & m_{zw} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

- dado $\mathbf{p} = (x, y, z, 1)^T$ obtenemos $\mathbf{p}_p = (x_p, y_p, z_p, w_p)^T$.
- Las coordenadas normalizadas de dispositivo se obtienen dividiendo por la cuarta coordenada: $\mathbf{p}_n = (\frac{x_p}{w_p}, \frac{y_p}{w_p}, \frac{z_p}{w_p})^T$
- Parece tener sentido que fijemos la cuarta fila de $\mathbf{M}_p$ con los valores $(0, 0, -1, 0)$, ya que en los puntos proyectados aparece dividiendo z.

Obtención de la matriz de proyección

Primera fila de la matriz:

- La coordenada x_n para los puntos internos al volumen de visión va de -1 hasta 1. Por tanto, los puntos del plano que limita el volumen de visión por la izquierda tienen que cumplir $\frac{x_p}{w_p} = -1$. Los puntos de ese plano cumplen tambien la ecuación del plano: $x = \frac{-l}{n} z$
- La coordenada x_p no depende de y, por tanto $m_{xy} = 0$
- Dos puntos del plano que limita el volumen por la izquierda con distintas z (sean $p_1 = (\frac{-l}{n} z, y, z, 1)^T$ y $p_2 = (\frac{-l}{n} 2z, y', 2z, 1)^T$) pero con la misma proyección en el plano de proyección deben tener la misma x_n, por tanto:

$$\frac{m_{xx}\frac{-l}{n} z + m_{xz} z + m_{xw}}{-z} = \frac{m_{xx}\frac{-l}{n} 2z + m_{xz} 2z + m_{xw}}{-2z}$$

  Simplificando $2m_{xw} = m_{xw}$. Su solución: $m_{xw} = 0$

Obtención de la matriz de proyección

Primera fila de la matriz, faltan m_{xx} y m_{xz}:

- Los puntos del plano que limita el volumen por la izquierda ($x = \frac{-l}{n}z$) cumplen:

$$\frac{m_{xx}\frac{-l}{n}z + m_{xz}z}{-z} = -1$$

- Los puntos del plano que limita el volumen por la derecha ($x = \frac{-r}{n}z$) cumplen:

$$\frac{m_{xx}\frac{-r}{n}z + m_{xz}z}{-z} = 1$$

La solución de estas dos ecuaciones es:

$$m_{xx} = \frac{2n}{r-l}, \quad \text{y} \quad m_{xz} = \frac{r+l}{r-l}$$

Obtención de la matriz de proyección

Segunda fila de la matriz:

- Con los mismos razonamientos que para la x_n:
- $m_{yx} = 0$ (la y_n no depende de x)
- $m_{yw} = 0$ (dos puntos con la misma proyección...)
- La $y_n = 1$ para un punto del plano superior ($y = \frac{-t}{n} z$) y $y_n = -1$ para un punto del plano inferior ($y = \frac{-b}{n} z$). Dos ecuaciones con dos incógnitas cuya solución es:

$$m_{yy} = \frac{2n}{t-b}, \text{ y } m_{xz} = \frac{t+b}{t-b}$$

Obtención de la matriz de proyección

Tercera fila de la matriz:

- Todos los puntos que esten en el plano de proyección ($z = -n$) en las coordenadas normalizadas han de tener $z_n = -1$, independientemente de x e y, por tanto $m_{zx} = m_{zy} = 0$ y además,

  para el plano cercano:

$$\frac{z_p}{w_p} = \frac{m_{zz}(-n) + m_{zw}}{n} = -1$$

  mientras que para el lejano:

$$\frac{z_p}{w_p} = \frac{m_{zz}(-f) + m_{zw}}{f} = 1$$

- Dos ecuaciones con dos incognitas cuya solución es:

$$m_{zz} = \frac{-(f+n)}{f-n}, \text{ y } m_{zw} = \frac{-2fn}{f-n}$$

Interpretación de la matriz de Proyección

$$\mathbf{M}_p = \begin{pmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

- Escalado de x y de y a la longitud 2 (de -1 a 1)
- **Shear** en caso de que la ventana no esté centrada.
- El escalado en z es diferente
- LLeva el origen al centro del volumen normalizado

Proyección perspectiva. Coordenada de profundidad modificada

- las coordenadas x e y se transforman mediante la interpolación lineal, pero la coordenada z no

$$
\begin{aligned}
x' &= f(x) = \frac{2}{r-l}\left(x - \frac{(r+l)z}{2n}\right) \\
y' &= g(y) = \frac{2}{t-b}\left(y - \frac{(t+b)z}{2n}\right) \\
z' &= h\left(\frac{1}{z}\right) = -\frac{2f}{f-n}\left(1 - \frac{n}{z}\right)
\end{aligned}
$$

- Tras la proyección, se realizan ciertas interpolaciones:
  - normales
  - Coordenadas textura
  - ...
- Hay que tener cuidado con la interpolación de la coordenada z
  - Interpolación perspectiva

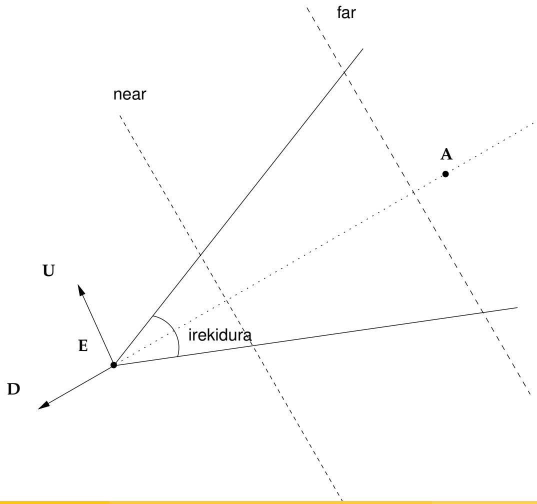
Proyección perspectiva en Three.js

```
makeFrustum: function  (
                       left, right,
                       top, bottom,
                       near, far )
```

- En la práctica, a menudo $l = -r$ y $b = -t$, por tanto establece una ventana rectangular.
- Se utilizan dos parámetros para especificar la perspectiva: apertura (*fov*) y ratio del aspecto (*aspect ratio*)

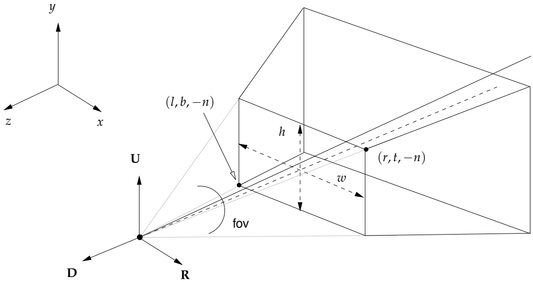
```
makePerspective: function  (
                           fov, aspect,
                           near, far )
```

Ángulo de apertura



Field of view

- Establece el volumen piramidal.
- Pide dos parámetros: fov_y y $\frac{w}{h}$



Especificación de la piramide cortada

- Sean fov_y la apertura en la dirección y, $\frac{w}{h}$ el ratio del aspecto, n la distancia al plano cercano y f al lejano, ¿Qué valores toman (l, r, b, t) ?
- A tener en cuenta la relación:

$$\frac{r-l}{t-b} = \frac{w}{h}$$

- De donde llegamos a:

$$\begin{aligned} t &= n \tan \frac{\mathrm{fov}_y}{2} \\ b &= -t \\ r &= \frac{w}{h} t \\ l &= -r \end{aligned}$$

makePerspective

- Dados (*fovy, aspect, zNear, zFar*):
  - Sea $f = \cot(\frac{\text{fovy}}{2}) = \frac{1}{\tan(\frac{\text{fovy}}{2})}$
  - La matriz de perspectiva es:

$$\mathbf{P}_p = \begin{pmatrix} \frac{f}{\text{aspect}} & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & \frac{\text{zNear}+\text{zFar}}{\text{zNear}-\text{zFar}} & \frac{2\text{zNear}*\text{zFar}}{\text{zNear}-\text{zFar}} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

Especificación de los planos del Frustum

- Basado en el trabajo de (Gribb & Hartmann, 2001)
- Toma en cuenta las matrices de proyección y cambio de vista
- Sea la matriz de proyección $\mathbf{M}_{o|p}$, la matriz de cambio de vista $\mathbf{M}_b$, y $\mathbf{M} = \mathbf{M}_{o|p}\mathbf{M}_b$
- El punto $\mathbf{P} = (x, y, z, 1)^T$ se transorma en otro punto $\mathbf{T} = \mathbf{MP} = (x', y', z', w')^T$
- La cuarta coordenada del punto $\mathbf{T}$, w', no va a ser 1, debido a la proyección.
    - División de todas las coordenadas por la cuarta componente w'
- Sea $\mathbf{U} = (u_x, u_y, z_y)$, tal que: $\begin{cases} u_x & = & \frac{x'}{w'} \\ u_y & = & \frac{y'}{w'} \\ u_z & = & \frac{z'}{w'} \end{cases}$

Especificación de los planos del Frustum

- El punto $\mathbf{U}$ se encuentra en coordenadas normalizadas de dispositivo
- Será visible si el punto esta dentro del cubo unitario (*clipping cube*), es decir, $-1 \leq u_i \leq 1,\ i \in x, y, z$
- Analicemos la parte derecha del plano izquierdo, donde $-1 \leq u_x$

$$-1 \leq u_x \Leftrightarrow -1 \leq \frac{x'}{w'} \Leftrightarrow x' + w' \geq 0 \Leftrightarrow$$
$$(\mathbf{m}_x \cdot \mathbf{P}) + (\mathbf{m}_w \cdot \mathbf{P}) \geq 0 \Leftrightarrow (\mathbf{m}_x + \mathbf{m}_w) \cdot \mathbf{P} \geq 0$$

donde $\mathbf{m}_x$ y $\mathbf{m}_w$ son la primera y cuarta fila de la matriz $\mathbf{M}$

- La espresión $(\mathbf{m}_x + \mathbf{m}_w) \cdot \mathbf{P} \geq 0$ representa el plano izquierdo del *frustum* $\mathbf{n}X = d$, donde $\begin{cases} n_x &= & m_{xx} + m_{wx} \\ n_y &= & m_{xy} + m_{wy} \\ n_z &= & m_{xz} + m_{wz} \\ d &= & -(m_{xw} + m_{ww}) \end{cases}$

Especificación de los planos del Frustum

- Hay que cambiar el signo del vector normal para que apunte hacia fuera, de esta forma, los seis planos del frustum (l,r,b,t,n,f) son:

$l \begin{cases} n_x = -m_{wx} - m_{xx} \\ n_y = -m_{wy} - m_{xy} \\ n_z = -m_{wz} - m_{xz} \\ d = m_{ww} + m_{xw} \end{cases}$	$r \begin{cases} n_x = -m_{wx} + m_{xx} \\ n_y = -m_{wy} + m_{xy} \\ n_z = -m_{wz} + m_{xz} \\ d = m_{ww} - m_{xw} \end{cases}$
$b \begin{cases} n_x = -m_{wx} - m_{yx} \\ n_y = -m_{wy} - m_{yy} \\ n_z = -m_{wz} - m_{yz} \\ d = m_{ww} + m_{yw} \end{cases}$	$t \begin{cases} n_x = -m_{wx} + m_{yx} \\ n_y = -m_{wy} + m_{yy} \\ n_z = -m_{wz} + m_{yz} \\ d = m_{ww} - m_{yw} \end{cases}$
$n \begin{cases} n_x = -m_{wx} - m_{zx} \\ n_y = -m_{wy} - m_{zy} \\ n_z = -m_{wz} - m_{zz} \\ d = m_{ww} + m_{zw} \end{cases}$	$f \begin{cases} n_x = -m_{wx} + m_{zx} \\ n_y = -m_{wy} + m_{zy} \\ n_z = -m_{wz} + m_{zz} \\ d = m_{ww} - m_{zw} \end{cases}$

Parámetros de la cámara/estructura de datos

Especificaremos:

- Posición y orientación (en el sitema de referencia del mundo)
  - Posición $\mathbf{V}$
  - Punto de atención $\mathbf{A}$
  - Vector hacia arriba $\mathbf{up}$
    - A veces debe ser unitario
    - No tiene por qué ser perpendicular a $\mathbf{VA}$ pero no puede ser paralelo al mismo.
  - Eje z de la cámara
    - *Near plane* d_n
    - *Far plane* d_f
  - Independientes al sistema
    - Apertura
    - *Aspect ratio* anchura/altura $a = \frac{w}{h}$
- Tipo de proyección (ortogonal/perspectiva)

Resumiendo

Pipeline

- Cambio del sistema local al sitema del mundo $\mathbf{M}_L$
- Cambio del sistema del mundo al sistema de la cámara
  - Cambio de vista: $\mathbf{M}_b$
- Cambio al cubo normalizado: $\mathbf{M}_{o|p}$
- *Frustum culling* (ya se verá)
- Sistema de coordenadas de la pantalla
  - Encuadre: $\mathbf{M}_v$

$$\mathbf{P}' = \mathbf{M}_v \mathbf{M}_{o|p} \mathbf{M}_b \mathbf{M}_L \mathbf{P}$$

Control de cámara

- Hay muchas formas de mover la cámara
  - Modo de analisis (*examine, circle, arc*)
  - Mover la cámara (*crane, dolly*)
  - Zoom
  - Rotacion
    - Cámara fija (*pan, tilt*)
    - Vuelo de cámara, aviones (*yaw, heading, pitch*)
  - Paseo
  - Seguir a un objeto
  - ...

Control de cámara

- La situación de la cámara se fija totalmente con los parámetros que en un momento tiene su estructura de datos
- La interface de usuario qué tarea especifica
- Cómo se mueve la cámara, es decir, cómo decidir la nueva situación de la cámara
    - Situación actual
    - Interacción con el usuario

Modo de analisis

- La posición de la cámara se mueve en la superficie de una esfera cuyo centro es $\mathbf{A}$
- El usuario elige el punto de atención o centro de la esfera.

Modo de analisis

Arc over-under

- Moverse en función de los meridianos (entre los polos)
  - Rotación de $\mathbf{V}$ y $\mathbf{up}$ en torno al eje $\mathbf{e}$, el cual es paralelo al plano XZ del mundo, perpendicular a $\mathbf{VA}$ y que pasa por $\mathbf{A}$.
  - Si la cámara se aproxima demasiado a un polo $\rightarrow$ singularidad

Arc left-right

- Moverse en función de los paralelos
  - Rotar $\mathbf{V}$ y $\mathbf{up}$ en torno al eje $\mathbf{e}$, que es paralelo al eje Y del mundo y que pasa por $\mathbf{A}$

¿Cuanto rotamos?

- Factor constante
- Factor dinámico (por ejemplo si rotamos con el raton)

Movimiento de la cámara

- Se trata de mover los puntos $\mathbf{V}$ y $\mathbf{A}$

$$\begin{aligned} \mathbf{V}' &= \mathbf{T}(\mathbf{t})\mathbf{V} \\ \mathbf{A}' &= \mathbf{T}(\mathbf{t})\mathbf{A} \end{aligned}$$

- ¿Qué sistema de referencia hay que utilizar?
- El de la cámara
  - *crane*: En el eje $\mathbf{R}$ (izda/dcha): $\mathbf{t} = \pm \text{step} \cdot \mathbf{R}$
  - *dolly*: En el eje $\mathbf{U}$ (arriba/abajo): $\mathbf{t} = \pm \text{step} \cdot \mathbf{U}$
  - *truck*: En el eje $\mathbf{D}$ (avanzar/retroceder): $\mathbf{t} = \pm \text{step} \cdot \mathbf{D}$
- Equivalente a volar
  - Si un objeto está en el suelo, mirando hacia abajo, avanzaría hacia el interior del suelo
  - Mirar izquierda/derecha, pero avanzar hacia delante

Cambio de posición de la cámara

- Mover los puntos $\mathbf{V}$ y $\mathbf{A}$

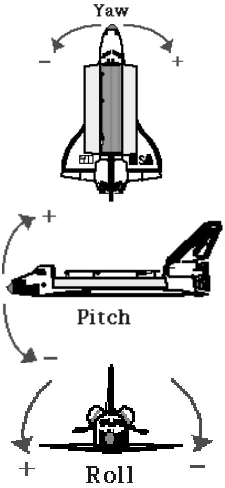
$$\begin{aligned} \mathbf{V}' &= \mathbf{T}(\mathbf{t})\mathbf{V} \\ \mathbf{A}' &= \mathbf{T}(\mathbf{t})\mathbf{A} \end{aligned}$$

- ¿Qué sistema de referencia hay que utilizar?
- El del mundo
  - En el eje x (izda/dcha del mundo): $\mathbf{t} = (\pm\text{step}, 0, 0)^T$
  - En el eje y (arriba/abajo): $\mathbf{t} = (0, \pm\text{step}, 0)^T$
  - En el eje z (adelante/atras): $\mathbf{t} = (0, 0, \pm\text{step})^T$
- Problemas
  - Si el suelo no es regular!

Dolly y Zoom

- Los cambios *Dolly* (adelantar, retrasar) se puede ver como un *zoom*. Pero no son iguales.
- *Dolly* no consigue el efecto de profuncidad, ya que la perspectiva no cambia.
- Para hacer *Zoom* hay que cambiar la apertura “field of view”
    - Los ángulos grandes (70 80) crean distorsiones

Giros de cámara



Heading,yaw: $\mathbf{A}' = \mathbf{R_U}(\alpha)\mathbf{A}$, $\mathbf{up}' = \mathbf{R_U}(\alpha)\mathbf{up}$

pitch: $\mathbf{A}' = \mathbf{R_R}(\alpha)\mathbf{A}$, $\mathbf{up}' = \mathbf{R_R}(\alpha)\mathbf{up}$

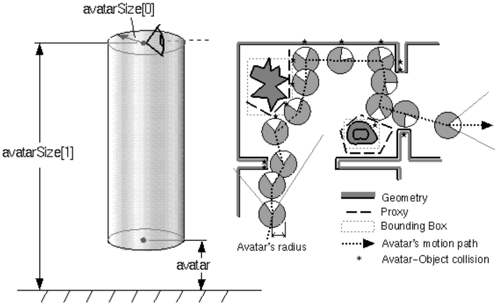
roll: $\mathbf{A}' = \mathbf{R_D}(-\alpha)\mathbf{A}$, $\mathbf{up}' = \mathbf{R_D}(-\alpha)\mathbf{up}$

http://liftoff.msfc.nasa.gov/academy/rocket_sci/shuttle/attitude/pyr.html

Avatar

- Tres paámetros:
  - Altura del ojo (desde el suelo)
    - La cámara se situa en el ojo
  - Altura de la rodilla
    - Para decidir si al tropezar con un objeto se puede subirse encima del objeto o no.
  - Radio
    - Colisiones
- Se modela cmo un cilindro (que va desde la rodilla a la cabeza)

Avatar


avatarSize[0]
avatarSize[1]
avatar
Avatar's radius
Geometry
Proxy
Bounding Box
Avatar's motion path
*
Avatar-Object collision