Tema I. Transformaciones Geométricas

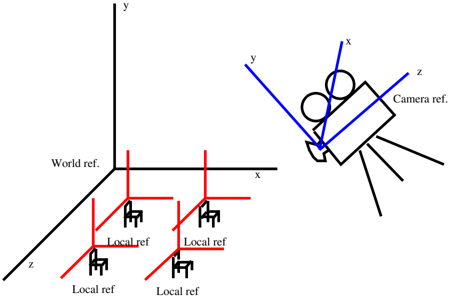
M.C. Hernandez
<mamen.hernandez@ehu.eus>
Joseba Makazaga
<joseba.makazaga@ehu.eus>

Universidad del País Vasco
Euskal Herriko Unibertsitatea

UPV/EHU

objetivo

- Transformar las propiedades geométricas de los objetos
  - Posición
  - Tamaño
  - Orientación
  - ...
- Para tener varias referencias
  - Sistema de referencia del mundo
  - Sistema de referencia de la cámara
  - Sistema de referencia local del objeto
    - Agrupamiento de objetos



Tipos de transformaciones

Transformación de modelos

- Utilizando objetos simples podremos crear objetos complejos
- Cambio del sistema de referencia del objeto al del mundo. Jerarquias.

Transformacion de vistas

- posicionamiento de la cámara en el mundo
- Cambio del sistema de referencia del mundo al de la cámara

Animaciones

- Transformaciones en el tiempo, para crear movimiento

Traslación

- $\mathbf{t} = (t_x, t_y, t_z)$ vector de movimiento

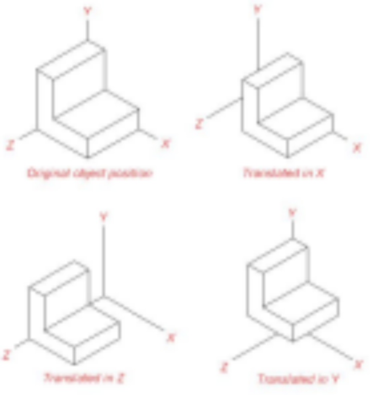
$$\mathbf{T}(\mathbf{t}) = \mathbf{T}(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Sea el punto $\mathbf{p} = (p_x, p_y, p_z, 1)^T$

$$\mathbf{T}(\mathbf{t})\mathbf{p} = (p_x + t_x, p_y + t_y, p_z + t_z, 1)^T$$

- Si $\mathbf{p}$ es un vector, $\mathbf{p} = (p_x, p_y, p_z, 0)^T$, $\Rightarrow \mathbf{T}(\mathbf{t})\mathbf{p} = \mathbf{p}$
- $\mathbf{T}^{-1}(\mathbf{t}) = \mathbf{T}(-\mathbf{t})$

Traslación



Entry Level Graphics Course Curriculum
Western Michigan University
Jason_a.cavanaugh at wmich.edu
http://grog.lab2.cc.wmich.edu/atkins/section52.htm

Escalado

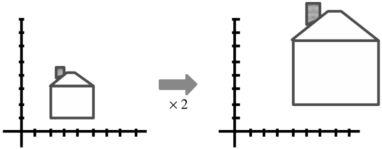
- Sean s_x, s_y, s_z los factores para el cambio de escala, cada factor para su correspondiente eje x, y, z.

$$\mathbf{S}(\mathbf{s}) = \mathbf{S}(s_x, s_y, s_z) = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

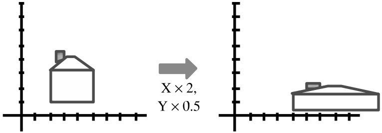
- $\mathbf{S}^{-1}(\mathbf{s}) = \mathbf{S}(1/s_x, 1/s_y, 1/s_z)$
- Si $s_x = s_y = s_z$, se dice que es un escalado *uniforme*. En otro caso, *no uniforme*
- Los escalados no uniformes aumentan la complejidad de ciertas operaciones
- Si el escalado es uniforme

$$\mathbf{S}(s, s, s) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1/s \end{pmatrix}$$

Escalado



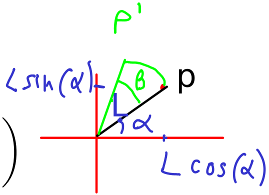
Uniforme



No uniforme

Rotación

- Coordenadas polares: $P = (L, \alpha)$
- Equivalente en coordenadas cartesianas:



$$P = (L, \alpha) \equiv \begin{pmatrix} L\cos(\alpha) \\ L\sin(\alpha) \end{pmatrix}$$

- Rotación de β grados:

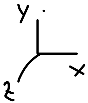
$$P' = (L, \alpha + \beta) \equiv \begin{pmatrix} L\cos(\alpha + \beta) \\ L\sin(\alpha + \beta) \end{pmatrix}$$

$$P' = \begin{pmatrix} L(\cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta)) \\ L(\cos(\alpha)\sin(\beta) + \sin(\alpha)\cos(\beta)) \end{pmatrix}$$

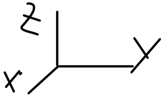
$$P' = \begin{pmatrix} \cos(\beta) & \sin(\beta)) \\ \sin(\beta) & \cos(\beta)) \end{pmatrix} \begin{pmatrix} L\cos(\alpha) \\ L\sin(\alpha) \end{pmatrix} = M \cdot P$$

Rotación

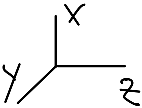
- Rotaciones respecto a los ejes x, y y z $\mathbf{R}_x(\phi), \mathbf{R}_y(\phi), \mathbf{R}_z(\phi)$:



$$\mathbf{R}_x(\phi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi & 0 \\ 0 & \sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



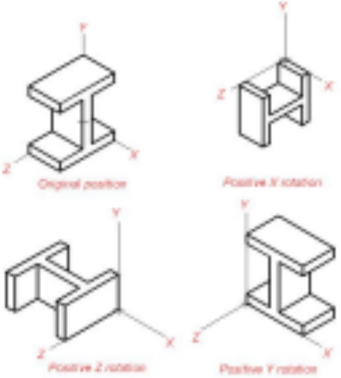
$$\mathbf{R}_y(\phi) = \begin{pmatrix} \cos\phi & 0 & \sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



$$\mathbf{R}_z(\phi) = \begin{pmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Son matrices ortonormales: $\mathbf{R}^{-1} = \mathbf{R}^T$
- Ademas, $\mathbf{R}_i^{-1}(\phi) = \mathbf{R}_i(-\phi)$

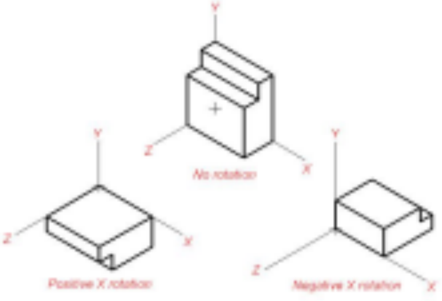
Rotación



Entry Level Graphics Course Curriculum
Western Michigan University
Jason_a.cavanaugh at wmich.edu
http://grog.lab2.cc.wmich.edu/atkins/section52.htm

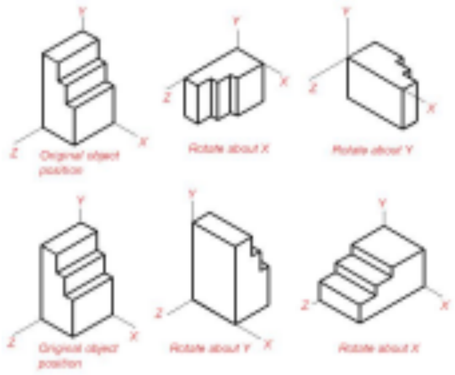
Rotación

- La dirección de la rotación:
  Regla de la mano derecha



Composición de transformaciones

- Multiplicación de matrices
- No es una operación conmutatitva: el orden importa



Composición de transformaciones

- Sea la compsición de las matrices $ABCD$:

$$\mathbf{P}' = ABCD\mathbf{P} = ABCD \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

- $\mathbf{P}' = (A(B(C(D\mathbf{P}))))$
- Podemos calcular $M = ABCD$, y entonces $\mathbf{P}' = M\mathbf{P}$

$$\begin{aligned} M &\leftarrow D \\ M &\leftarrow CM \\ M &\leftarrow BM \\ M &\leftarrow AM \end{aligned} \quad \text{ó} \quad \begin{aligned} M &\leftarrow A \\ M &\leftarrow MB \\ M &\leftarrow MC \\ M &\leftarrow MD \end{aligned}$$

Premultiply	*Postmultiply*
De izquierda a derecha	De derecha a izquierda

Composición de transformaciones

- Obtener las matrices y los resultados se pueden ir componiendo en una única matriz M
- la postmultiplicación es el formalismo mas utilizado:

$$\mathbf{P}' = \begin{pmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{pmatrix} = M\mathbf{P} = M \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$$

- Para calcular M
  - Inicializar $M = I$
  - Calcular las transformaciones en orden inverso

Composición de transformaciones. Implementación

- Por ejemplo
  1. llevar el punto $\mathbf{P} = (x, y, 0)$ al origen: $\mathbf{T}(-x, -y, 0)$
  2. despues rotar en el eje z ϕ grados: $\mathbf{R}_z(\phi)$
  3. Llevar el punto a la posición inicial: $\mathbf{T}(x, y, 0)$ (deshacer la traslación inicial)

  Para este caso $\mathbf{P}' = \mathbf{T}(x, y, 0) \cdot \mathbf{R}_z(\phi) \cdot \mathbf{T}(-x, -y, 0) \cdot \mathbf{P}$

$$M \Leftarrow T_v \cdot R_z \cdot T_{-v}$$

Matrices en THREE.js

```
THREE.Matrix4.prototype={constructor:THREE.Matrix4,set:
function(a,b,c,d,e,f,g,h,k,n,p,q,m,t,s,r)
{var u=this.elements;
u[0]=a;u[4]=b;u[8]=c;u[12]=d;
u[1]=e;u[5]=f;u[9]=g;u[13]=h;
u[2]=k;u[6]=n;u[10]=p;u[14]=q;
u[3]=m;u[7]=t;u[11]=s;u[15]=r;
return this} ...
```

- Es decir, la funcion en column-mayor
- ... pero en memoria row-mayor!!!

Matrices en THREE.js

Multiplicación de matrices:

- miobjeto.matrix.multiplyMatrices(M_1, M_0);

  Matriz $\Leftarrow M_1 \cdot M_0$
- miobjeto.matrix.multiply(M)

  Matriz $\Leftarrow$ Matriz $\cdot M$

Composición de transformaciones. Vectores fila

- Podemos representar los puntos/vectores como vectores fila.
- Siendo $P = (p_x, p_y, p_z, 1)$ un vector fila
- Hay que usar las traspuestas de la matrices de transformación
- De esta forma el orden de las transformaciones (multiplicaciones de matrices) es el inverso
- $ABCD\mathbf{P}$ se convierte en $\mathbf{P}D^T C^T B^T A^T$
- OpenGL y Three.js internamente utilizan matrices traspuestas

$$M = \begin{pmatrix} m_{11} & m_{21} & m_{31} & 0 \\ m_{12} & m_{22} & m_{32} & 0 \\ m_{13} & m_{23} & m_{33} & 0 \\ t_x & t_y & t_z & 1 \end{pmatrix}$$

  - Si utilizamos las matrices como parametros es un dato a tener en cuenta
  - La mayoría de las funciones solo requieren parámetros escalares

Composición de rotaciones

- Objetivo: rotar el punto P ϕ grados respecto al eje dado por el vector $\mathbf{v}$ (que pasa por el origen)
- Utilizando las rotaciones respecto a los ejes x, y, z:
  1. Rotar α grados en el eje z, hasta que el vector $\mathbf{v}$ esté en el plano YZ: $\mathbf{R}_z(\alpha)$
  2. A continuación, rotar β grados respecto al eje x, hasta que el vector $\mathbf{v}$ se situe en el eje z: $\mathbf{R}_x(\beta)$
  3. Rotar ϕ grados respecto al eje z: $\mathbf{R}_z(\phi)$
  4. Deshacer la rotación respecto al eje x: $\mathbf{R}_x(-\beta)$
  5. Deshacer la rotación respecto al eje z: $\mathbf{R}_z(-\alpha)$

Composición de rotaciones

- Posible interpretación:
    1. Las dos primeras transformaciones realizan un cambio de sistema de referencia de forma que el eje de rotación pasa a ser el eje z del nuevo sistema de referencia
    2. La tercera transformación relaiza la rotación respecto al eje que nos interesa (ahora el eje z)
    3. Las dos últimas vuelven a realizar el cambio de sistema de referencia para pasar al sistema original
- Column-mayor o vectores columna:

$$\mathbf{R}(\phi) = \mathbf{R}_z(-\alpha)\mathbf{R}_x(-\beta)\mathbf{R}_z(\phi)\mathbf{R}_x(\beta)\mathbf{R}_z(\alpha)$$

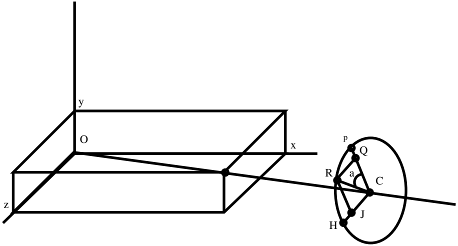
- Row-mayor o vectores fila:

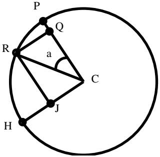
$$\mathbf{R}^T(\phi) = \mathbf{R}_z^T(\alpha)\mathbf{R}_x^T(\beta)\mathbf{R}_z^T(\phi)\mathbf{R}_x^T(-\beta)\mathbf{R}_z^T(-\alpha)$$

Rotación respecto a un eje cualquiera

- Eje $u = (x, y, z)$ y ángulo α
- Un punto P pasará a ser R:
  $R = C + \overline{CQ} + \overline{QR}$

  Pero $C = (P \cdot u)u$ y
  $\overline{QR} \equiv \overline{CJ}$
  Además, $\overline{CQ} = \cos(\alpha)\overline{CP}$
  y $\overline{CJ} = \sin(\alpha)\overline{CH}$
  Por otra parte, $\overline{CH} = u \wedge P$





Por tanto $R = (P \cdot u)u + \cos(\alpha)(P - (P \cdot u)u) + \sin(\alpha)(u \wedge P)$

Rotación respecto a un eje cualquiera

$$R = C + \overline{CQ} + \overline{QR}$$

- C: Proyección del vector P sobre el eje de rotación: $C = (P \cdot d)d$
- $\overline{CQ}$: Vector con la misma direccion que $\overline{CP}$, pero longitud menor. La rotación hace moverse al punto P en una circunferencia, la distancia al centro no varía.

  $$\cos(\alpha) = \frac{\| \overline{CQ} \|}{\| \overline{CR} \|} = \frac{\| \overline{CQ} \|}{\| \overline{CP} \|}$$

  Por tanto, $\overline{CQ} = \cos(\alpha)\overline{CP}$
- $\overline{QR}$: vector paralelo a $\overline{CH}$, que a su vez ez perpendicular al vector $\overline{CP}$. Tambien es perpendicular al plano formado por los puntos $0, P$ y C. Producto vectorial! $u \wedge P$. Ademas $\| u \wedge P \| = \| u \| \| P \| \sin(\beta) = 1 \| P \| \frac{\|\overline{CP}\|}{\|P\|} = \| \overline{CP} \|$. Por tanto:

  $$\overline{QR} = \sin(\alpha)(u \wedge P)$$

Rotación respecto a un eje cualquiera

$$R = C + \overline{CQ} + \overline{QR}$$

$$R = C + \cos(\alpha)\overline{CP} + sin(\alpha)(u \wedge P)$$

$$R = \begin{pmatrix} (P \cdot d)x \\ (P \cdot d)y \\ (P \cdot d)z \end{pmatrix} + cos(\alpha) \begin{pmatrix} P_x - (P \cdot d)x \\ P_y - (P \cdot d)y \\ P_z - (P \cdot d)z \end{pmatrix} + sin(\alpha) \begin{pmatrix} yP_z - zP_y \\ zP_x - xP_z \\ xP_y - YPx \end{pmatrix}$$

donde $(P \cdot d) = xP_x + yP_y + zP_z$. Para la primera componente tenemos:

$$\begin{aligned} R_x &= (xP_x + yP_y + zP_z)x + \cos(\alpha)(P_x - (xP_x + yP_y + zP_z)x) + \\ &\quad + \sin(\alpha)(yP_z - zP_y) \\ &= P_x(x^2 + \cos(\alpha) - \cos(\alpha)x^2) + P_y(xy - \cos(\alpha)xy - z\sin(\alpha)) + \\ &\quad P_z(xz - cos(\alpha)xz + y\sin(\alpha)) \end{aligned}$$

Para las tres componentes, en forma matricial:

$$\begin{pmatrix} R_x \\ R_y \\ R_z \end{pmatrix} = \begin{pmatrix} \cos\alpha + (1-\cos\alpha)x^2 & (1-\cos\alpha)xy - z\sin\alpha & (1-\cos\alpha)xz + y\sin\alpha \\ (1-\cos\alpha)xy + z\sin\alpha & \cos\alpha + (1-\cos\alpha)y^2 & (1-\cos\alpha)yz - x\sin\alpha \\ (1-\cos\alpha)xz - y\sin\alpha & (1-\cos\alpha)yz + x\sin\alpha & \cos\alpha + (1-\cos\alpha)z^2 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix}$$

Transformaciones en Three.js

- cada objeto puede tener definidos: scale, rotation, translation

  Scale: x, y, z

  Rotation: x, y, z

  Translation: x, y, z

- Resultado final:

  $T_{xyz} \cdot R_x \cdot R_y \cdot R_z \cdot S_{xyz} \cdot Objeto$

- En caso de que no queramos ese orden?

1. Gestión manual de matrices
2. Mediante correcta gestión del Grafo de Escena

Aplicando transformaciones

- Sólo tendremos en cuenta la translación, la rotación y el escalado.
- La composición de trasformaciones se puede representar mediante una única matriz (mediante la multiplicación de matrices)
- Sea $\mathbf{M}$ la composición de transformaciones

$$\mathbf{X} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & t_x \\ m_{10} & m_{11} & m_{12} & t_y \\ m_{20} & m_{21} & m_{22} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{SR} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

donde $\left\{ \begin{array}{ccc} \mathbf{S} & \text{escalado} & \mathbf{S}(\mathbf{s}) \\ \mathbf{R} & \text{Rotación} & \mathbf{R}(\mathbf{n}, \alpha) \\ \mathbf{T} & \text{Traslación} & \mathbf{T}(\mathbf{t}) \end{array} \right.$

Aplicando transformaciones

- Si $\mathbf{P}$ es un punto: $\mathbf{P}' = \mathbf{M}\mathbf{P} = \begin{pmatrix} \mathbf{S}\mathbf{R} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix} \mathbf{P} = \mathbf{S}\mathbf{R}\mathbf{P} + \mathbf{T}$
- La inversa: $\mathbf{P} = \mathbf{M}^{-1}\mathbf{P}' = \begin{pmatrix} \mathbf{S}^{-1}\mathbf{R}^T & -\mathbf{S}^{-1}\mathbf{R}^T\mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix} \mathbf{P}' = \mathbf{S}^{-1}\mathbf{R}^T(\mathbf{P}' - \mathbf{T})$
- Si $\mathbf{V}$ es un vector: $\mathbf{V}' = \mathbf{M}\mathbf{V} = \mathbf{S}\mathbf{R}\mathbf{V}$ (sin traslaciones)

Aplicando transformaciones: composición

- Supongamos que a un punto $\mathbf{P}$ queremos aplicarle primeramente $\mathbf{M}_1$, y despues $\mathbf{M}_2$

$$
\begin{aligned}
\mathbf{P}' &= \mathbf{M}_2 \cdot \mathbf{M}_1 \cdot \mathbf{P} \\
\mathbf{P}' &= \begin{pmatrix} \mathbf{S}_2\mathbf{R}_2 & \mathbf{T}_2 \\ \mathbf{0}^T & 1 \end{pmatrix} \cdot \begin{pmatrix} \mathbf{S}_1\mathbf{R}_1 & \mathbf{T}_1 \\ \mathbf{0}^T & 1 \end{pmatrix} \cdot \mathbf{P} \\
\mathbf{P}' &= \begin{pmatrix} \mathbf{S}_2\mathbf{R}_2 & \mathbf{T}_2 \\ \mathbf{0} & 1 \end{pmatrix} \cdot (\mathbf{S}_1\mathbf{R}_1\mathbf{P} + \mathbf{T}_1) \\
\mathbf{P}' &= \mathbf{S}_2\mathbf{R}_2(\mathbf{S}_1\mathbf{R}_1\mathbf{P} + \mathbf{T}_1) + \mathbf{T}_2 \\
\mathbf{P}' &= \mathbf{S}_2\mathbf{S}_1\mathbf{R}_2\mathbf{R}_1\mathbf{P} + \mathbf{S}_2\mathbf{R}_2\mathbf{T}_1 + \mathbf{T}_2
\end{aligned}
$$

Transformación de cuerpos rígidos. (Rigid-body transformation)

- No cambia tamaños/ángulos relativos del objeto
- Se obtiene mediante combinación de traslaciones y rotaciones:

$$\mathbf{X} = \mathbf{T}(\mathbf{t})\mathbf{R} = \begin{pmatrix} m_{00} & m_{01} & m_{02} & t_x \\ m_{10} & m_{11} & m_{12} & t_y \\ m_{20} & m_{21} & m_{22} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- $\mathbf{X}^{-1} = (\mathbf{T}(\mathbf{t})\mathbf{R})^{-1} = \mathbf{R}^{-1}\mathbf{T}(\mathbf{t})^{-1} = \mathbf{R}^T\mathbf{T}(-\mathbf{t})$
- Mediante la notación matricial:

$$\mathbf{X} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

$$\mathbf{X}^{-1} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0}^T & 1 \end{pmatrix}$$

Aplicando transformaciones: normales

- Dado el plano $Ax + By + Cz + D = 0$ si $\sqrt{A^2 + B^2 + C^2} = 1$ tenemos que $n = (A, B, C)^T$ es el vector normal al plano y que cualquier punto p del plano cumple $n_a P = 0$, donde: $n_a = (A, B, C, D)$ y

$$P = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

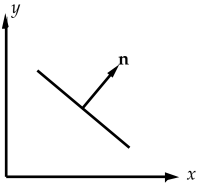
- Tras una transformación M el plano se transforma en otro plano, por tanto sus puntos cumplirán $n'_a p' = 0$, donde $n'_a = (A', B', C', D')$ y $p' = Mp$
- n'_a no es única ($kn'_a p' = 0$!), pero, $n_a \mathbf{M}^{-1} \mathbf{M} p = 0$ es una de las posibilidades. $n'_a = (A', B', C', D') = n_a \mathbf{M}^{-1}$.
- necesitamos $\mathbf{M}^{-1} = \begin{pmatrix} \mathbf{S}^{-1}\mathbf{R}^T & -\mathbf{S}^{-1}\mathbf{R}^T\mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix}$

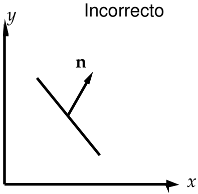
Aplicando transformaciones: normales

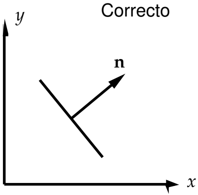
- podemos trabajar con vectores columna, $\mathbf{N} = (\mathbf{M}^{-1})^T$:

$$\begin{pmatrix} A' \\ B' \\ C' \\ D' \end{pmatrix} = (M^{-1})^T \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix}$$

- Si solo busco el vector normal n, entonces la cuarta fila no me interesa, además, si no hay cambio de escala, entonces $\mathbf{N} == \mathbf{M}$
- Si en la matriz $\mathbf{M}$ hay un cambio de escala *uniforme* $n' = \frac{\mathbf{M}n}{\|\mathbf{M}n\|}$
- Si hubiera cambio de escala no uniforme: hay que calcular $(\mathbf{M}^{-1})^T$!







Transformacion de planos

- queremos aplicar la transformación $\mathbf{M}$ al plano $\mathbf{n} \cdot \mathbf{X} = d$
  $\mathbf{M} = \begin{pmatrix} \mathbf{SR} & \mathbf{T} \\ \mathbf{0}^T & 1 \end{pmatrix}$
- Supongamos que el cambio de escala es uniforme, $\mathbf{S} = \mathbf{S}(s)$ y $\mathbf{S}^{-1} = \mathbf{S}(\frac{1}{s})$
- Nuevo vector perpendicular: $\mathbf{n}' = (\mathbf{M}^{-1})^T\mathbf{n} = \mathbf{S}^{-1}\mathbf{R}\mathbf{n}$
- Multiplicando el nuevo normal a un punto transformado, obtendremos la nueva distancia d':

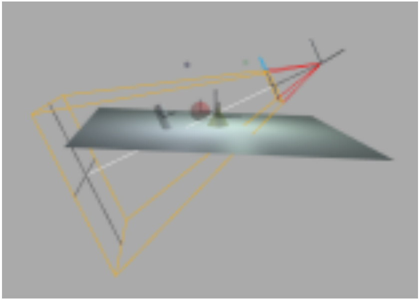
$$\begin{aligned}
d' &= \mathbf{n}' \cdot (\mathbf{MX}) = (\mathbf{n}')^T(\mathbf{RSX} + \mathbf{T}) = (\mathbf{n}')^T\mathbf{RSX} + \mathbf{n}' \cdot \mathbf{T} \\
d' &= (\mathbf{n}^T\mathbf{R}^T\mathbf{S}^{-1})\mathbf{RSX} + \mathbf{n}' \cdot \mathbf{T} \\
d' &= \mathbf{n}^T\mathbf{S}^{-1}\mathbf{S}\mathbf{R}^T\mathbf{R}\mathbf{X} + \mathbf{n}' \cdot \mathbf{T} \\
d' &= \mathbf{n}^T\mathbf{X} + \mathbf{n}' \cdot \mathbf{T} = \mathbf{n} \cdot \mathbf{X} + \mathbf{n}' \cdot \mathbf{T} = d + \mathbf{n}' \cdot \mathbf{T}
\end{aligned}$$

El nuevo plano

$$\begin{cases} \mathbf{n}' \cdot \mathbf{X} = d' & \text{Caso de que no haya cambio de escala(s=1)} \\ \frac{\mathbf{n}'}{\|\mathbf{n}'\|} \cdot \mathbf{X} = \frac{d+\mathbf{n}'\mathbf{T}}{\|\mathbf{n}'\|} & \text{otro caso} \end{cases}$$

Cambio de sistema de referencia

- Supongamos que definimos el sitema de referencia L:
  - el origen del sistema de referencia $\mathbf{o} = (o_x, o_y, o_z, 1)^T$
  - Tres vectores de dirección, formando una base ortonormal:
    - $\mathbf{R} = (r_x, r_y, r_z, 0)^T$ vector hacia la "derecha"
    - $\mathbf{U} = (u_x, u_y, u_z, 0)^T$ vector hacia "arriba"
    - $\mathbf{D} = (d_x, d_y, d_z, 0)^T$ vector hacia "atras"
  - ¿Cómo pasamos de un sistema a otro?

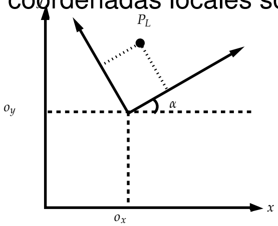


Cambio de sistema de referencia

- Para conocer las coordenadas mundo P_M de un un punto P_L en el sistema de referencia L:

$$P_M = \begin{pmatrix} r_x & u_x & d_x & o_x \\ r_y & u_y & d_y & o_y \\ r_z & u_z & d_z & o_z \\ 0 & 0 & 0 & 1 \end{pmatrix} P_L = \begin{pmatrix} \mathbf{R} & \mathbf{U} & \mathbf{D} & \mathbf{o} \\ 0 & 0 & 0 & 1 \end{pmatrix} P_L$$

- **Ejercicio.** Supongamos que el origen de la imagen $2D$ de la izquierda es $\mathbf{o} = (4,4)$ y el ángulo $\alpha = \pi/4$. ¿Qué coordenadas tendrá P_L en el mundo, si sus coordenadas locales son $(1,1)$?



Cambio de sistema de referencia

- A menudo, hace falta la transformación inversa.
- Si las coordenadas mundo de un punto son P_M, ¿Cuales son las coordenadas locales P_L?

$$P_M = \begin{pmatrix} \mathbf{R} & \mathbf{U} & \mathbf{D} & \mathbf{o} \\ 0 & 0 & 0 & 1 \end{pmatrix} P_L$$

$$P_L = \begin{pmatrix} \mathbf{R} & \mathbf{U} & \mathbf{D} & \mathbf{o} \\ 0 & 0 & 0 & 1 \end{pmatrix}^{-1} P_M$$

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

Cambio de sistema de referencia

- En esta imagen podemos ver las transformaciones y el orden que utiliza OpenGL.