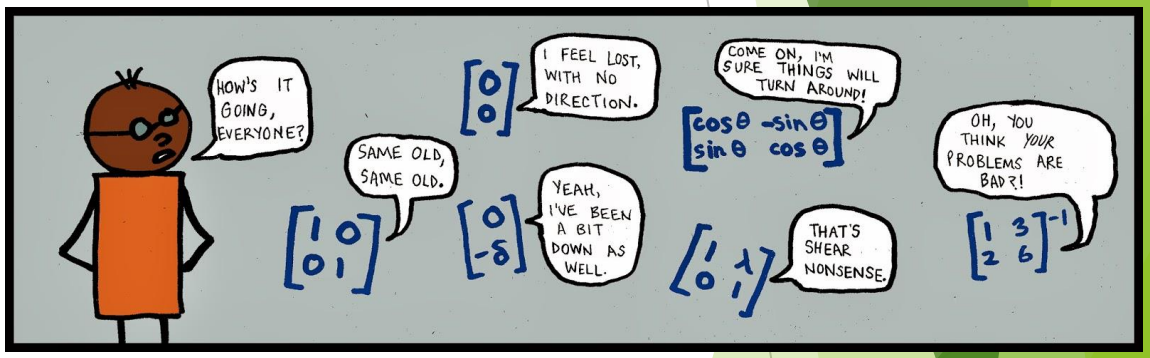
# Graphics en Game Technologie

## 4. 3D transformaties




Source: Center of Math

Robert Belleman
Computational Science Lab
Universiteit van Amsterdam
R.G.Belleman@uva.nl
(voeg a.u.b. "[GGT]" toe aan subject)

# 3D transformaties

- 3D transformatiematrices
- Samenstelling van transformatiematrices
- 3D rotatie om willekeurige as
- Orthonormale basis en veranderen coördinatensysteem
- `gluLookAt()`
- Gimbal lock

# 3D transformaties in homogene vorm

3D schaling :

$$\begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} s_x x \\ s_y y \\ s_z z \\ 1 \end{pmatrix}$$

3D rotatie om z-as:

$$\begin{pmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x\cos\phi - y\sin\phi \\ x\sin\phi + y\cos\phi \\ z \\ 1 \end{pmatrix}$$

# 3D transformaties in homogene vorm

3D x-shear $(d_y, d_z)$:

$$\begin{pmatrix} 1 & d_y & d_z & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x + d_y y + d_z z \\ y \\ z \end{pmatrix}$$

3D y-shear $(d_x, d_z)$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ d_x & 1 & d_z & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ d_x x + y + d_z z \\ z \end{pmatrix}$$

3D z-shear $(d_x, d_y)$:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ d_x & d_y & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ d_x x + d_y y + z \end{pmatrix}$$

# Translatie in homogene vorm

3D translatie:

$$\begin{pmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x + d_x \\ y + d_y \\ z + d_z \\ 1 \end{pmatrix}$$

# Samenstelling van transformatiematrices

# Voorbeeld schaling om een punt

Bereken de transformatiematrix die een voorwerp schaalt met factoren $(s_x, s_y, s_z) = (2, 3, 2)$ t.o.v. het punt $(1, 1, 0)$.

# Voorbeeld schaling om een punt

Bereken de transformatiematrix die een voorwerp schaalt met factoren $(s_x, s_y, s_z) = (2, 3, 2)$ t.o.v. het punt $(1, 1, 0)$.

$T(x_1, y_1, z_1) \cdot S \cdot T(-x_1, -y_1, -z_1) =$

# Voorbeeld schaling om een punt

Bereken de transformatiematrix die een voorwerp schaalt met factoren $(s_x, s_y, s_z) = (2, 3, 2)$ t.o.v. het punt $(1, 1, 0)$.

$$T(x_1, y_1, z_1) \cdot S \cdot T(-x_1, -y_1, -z_1) =$$

$$\begin{pmatrix} 1 & 0 & 0 & x_1 \\ 0 & 1 & 0 & y_1 \\ 0 & 0 & 1 & z_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & -z_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} =$$

$$\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 & -1 \\ 0 & 3 & 0 & -2 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

# 3D rotatie om willekeurige as

# Rotaties om willekeurige as

3D rotatiematrix om willekeurige as:

$$\boldsymbol{R}_{uvw} = \begin{pmatrix} x_u & y_u & z_u \\ x_v & y_v & z_v \\ x_w & y_w & z_w \end{pmatrix}$$

is orthogonaal: rijen zijn loodrechte eenheidsvectoren

$$\boldsymbol{u} = \begin{pmatrix} x_u \\ y_u \\ z_u \end{pmatrix}, \boldsymbol{v} = \begin{pmatrix} x_v \\ y_v \\ z_v \end{pmatrix}, \boldsymbol{w} = \begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix}$$

# Rijen orthonormale basis

1. $\|\boldsymbol{u}\| = \|\boldsymbol{v}\| = \|\boldsymbol{w}\| = 1 \Rightarrow \quad \boldsymbol{u}\cdot\boldsymbol{u} = \boldsymbol{v}\cdot\boldsymbol{v} = \boldsymbol{w}\cdot\boldsymbol{w} = 1$
2. $\boldsymbol{u} \perp \boldsymbol{v} \perp \boldsymbol{w} \Rightarrow \quad \boldsymbol{u}\cdot\boldsymbol{v} = \boldsymbol{v}\cdot\boldsymbol{w} = \boldsymbol{w}\cdot\boldsymbol{u} = 0$

Voor orthogonale matrix $\boldsymbol{R}_{uvw}$ geldt: rij gaat over in as

$$\boldsymbol{R}_{uvw}\boldsymbol{u} = \begin{pmatrix} x_u & y_u & z_u \\ x_v & y_v & z_v \\ x_w & y_w & z_w \end{pmatrix}\begin{pmatrix} x_u \\ y_u \\ z_u \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

# Rij gaat over in as

Orthogonale matrix $\boldsymbol{R}_{uvw}$:

$$\boldsymbol{R}_{uvw}\boldsymbol{u} = \begin{pmatrix}1\\0\\0\end{pmatrix},\ \boldsymbol{R}_{uvw}\mathbf{v} = \begin{pmatrix}0\\1\\0\end{pmatrix},\ \boldsymbol{R}_{uvw}\boldsymbol{w} = \begin{pmatrix}0\\0\\1\end{pmatrix}$$

$\boldsymbol{R}_{uvw}$ voert orthonormale basis $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ over in $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$-as

Je kunt altijd rotatiematrix maken uit orthonormale basis!!!

# Willekeurige en orthogonale matrix

Voor elke $3 \times 3$ matrix geldt: as gaat over in kolom

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} a_{11} \\ a_{21} \\ a_{31} \end{pmatrix}$$

Voor orthogonale matrix $\boldsymbol{R}$ geldt: rij gaat over in as

$$\begin{pmatrix} x_u & y_u & z_u \\ x_v & y_v & z_v \\ x_w & y_w & z_w \end{pmatrix} \begin{pmatrix} x_u \\ y_u \\ z_u \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

# R orthogonaal $\leftrightarrow$ $R^T = R^{-1}$

Als $\boldsymbol{R}$ rotatiematrix, dan

- $\boldsymbol{R}^T$ ook rotatiematrix
- $\boldsymbol{R}^T = \boldsymbol{R}^{-1}$

Inverse van orthogonale matrix is Transpose

- $\boldsymbol{R}_{uvw}$ voert orthonormale basis $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ over in $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$-as
- $\boldsymbol{R}^T_{uvw}$ voert orthonormale basis $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ over in $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$-as

# $\boldsymbol{R}_{uvw}^{T}$

$\boldsymbol{R}_{uvw}^{T}$ voert orthonormale basis $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ over in $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$-as

$$\boldsymbol{R}_{uvw}^{T}\boldsymbol{x} = \begin{pmatrix} x_u & x_v & x_w \\ y_u & y_v & y_w \\ z_u & z_v & z_w \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_u \\ y_u \\ z_u \end{pmatrix} = \boldsymbol{u}$$

# Wat is rotatiematrix om vector a over hoek $\phi$?

- Maak orthonormale basis $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ met $\boldsymbol{w} = \boldsymbol{a}$
- $\begin{pmatrix} - & \boldsymbol{u} & - \\ - & \boldsymbol{v} & - \\ - & \boldsymbol{a} & - \end{pmatrix}$ voert $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{a}$ over in basis $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$
- Roteer om $\boldsymbol{z}$-as over $\phi$
- Roteer $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ terug naar $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{a}$

# Rotatiematrix om vector a over hoek $\phi$

$$\begin{pmatrix} | & | & | \\ \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{a} \\ | & | & | \end{pmatrix} \begin{pmatrix} cos\phi & -sin\phi & 0 \\ sin\phi & cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} - & \boldsymbol{u} & - \\ - & \boldsymbol{v} & - \\ - & \boldsymbol{a} & - \end{pmatrix}$$

met $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{a}$ orthonormale basis

# Maak orthonormale basis

- Gegeven: vector $\boldsymbol{a}$
- Normeer $\boldsymbol{a}$: $\boldsymbol{a} = \frac{\boldsymbol{a}}{\|\boldsymbol{a}\|}$
- Construeer $\boldsymbol{tmp}$ uit $\boldsymbol{a}$ met kleinste waarde gelijk aan 1
- $\boldsymbol{u} = \frac{\boldsymbol{tmp} \times \boldsymbol{a}}{\|\boldsymbol{tmp} \times \boldsymbol{a}\|}$
- $\boldsymbol{v} = \boldsymbol{a} \times \boldsymbol{u}$

# Rekenvoorbeeld

- Gegeven: vector $\boldsymbol{a} = (2,3,1)$
- Normeer $\boldsymbol{a}$: $\|\boldsymbol{a}\| = \sqrt{4+9+1} = 3.7 \Rightarrow \boldsymbol{a} = \begin{pmatrix} 0.54 \\ 0.8 \\ 0.27 \end{pmatrix}$
- $\boldsymbol{tmp} = (0.54, 0.8, 1)$
- $\boldsymbol{u} = \frac{\boldsymbol{tmp} \times \boldsymbol{a}}{\|\boldsymbol{tmp} \times \boldsymbol{a}\|} = (0.83, -0.56, 0)$
- $\boldsymbol{v} = \boldsymbol{a} \times \boldsymbol{u} = (0.15, 0.22, -0.96)$

# Rekenvoorbeeld stappen 1 t/m 4

Wat is rotatiematrix om vector $\boldsymbol{a} = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix}$ over hoek 45°?

1. Orthonormale basis $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{a}$ $= \begin{pmatrix} 0.83 \\ -0.56 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.15 \\ 0.22 \\ -0.96 \end{pmatrix}, \begin{pmatrix} 0.54 \\ 0.8 \\ 0.27 \end{pmatrix}$

2. $\begin{pmatrix} - & \boldsymbol{u} & - \\ - & \boldsymbol{v} & - \\ - & \boldsymbol{a} & - \end{pmatrix} = \begin{pmatrix} 0.83 & -0.56 & 0 \\ 0.15 & 0.22 & -0.96 \\ 0.54 & 0.8 & 0.27 \end{pmatrix}$

   voert $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{a}$ over in $\boldsymbol{xyz}$-as

# Rekenvoorbeeld stappen 1 t/m 4

3. Roteer om $z$-as over 45°

$$\begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0.71 & -0.71 & 0 \\ 0.71 & 0.71 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

4. Roteer $x$, $y$, $z$ terug naar $u$, $v$, $a$

$$\begin{pmatrix} | & | & | \\ \boldsymbol{u} & \boldsymbol{v} & \boldsymbol{a} \\ | & | & | \end{pmatrix} = \begin{pmatrix} 0.83 & 0.15 & 0.54 \\ -0.56 & 0.22 & 0.8 \\ 0 & -0.96 & 0.27 \end{pmatrix}$$

# Rotatiematrix om vector $(2, 3, 1)$ over hoek 45°

$$\begin{pmatrix} 0.83 & 0.15 & 0.54 \\ -0.56 & 0.22 & 0.8 \\ 0 & -0.96 & 0.27 \end{pmatrix} \begin{pmatrix} 0.71 & -0.71 & 0 \\ 0.71 & 0.71 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0.83 & -0.56 & 0 \\ 0.15 & 0.22 & -0.96 \\ 0.54 & 0.8 & 0.27 \end{pmatrix} =$$

$$\begin{pmatrix} 0.79 & -0.06 & 0.61 \\ 0.32 & 0.9 & -0.32 \\ -0.53 & 0.44 & 0.72 \end{pmatrix}$$

Vector $(2, 3, 1)$ gaat over in zichzelf

$$\begin{pmatrix} 0.79 & -0.06 & 0.61 \\ 0.32 & 0.9 & -0.32 \\ -0.53 & 0.44 & 0.72 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} \surd$$

# Draaihoek van een matrix

Draaihoek aflezen uit diagonaal matrix:

$$\cos(\text{hoek}) = \frac{\text{trace}(R) - 1}{2}$$

(N.B. $\text{trace}(R)$ van een vierkante matrix $R$ is de som van de diagonaalelementen van $R$)

$$\cos(\text{hoek}) = \text{bv. } 1/2 \Rightarrow \text{hoek} = \pi/3$$

Willekeurige rotatie-matrix:

- hoek uit diagonaal van matrix
- rotatie-as met behulp van eigenwaarde 1 en bijbehorende eigenvector

# Transformeren van normaalvectoren

Als normaalvector $\boldsymbol{n}$ getransformeerd met

$\boldsymbol{M}$, dan $\boldsymbol{Mn}$ niet loodrecht op rechthoek

Welke matrix $\boldsymbol{N}$ levert $\boldsymbol{Nn}$ wél loodrecht op rechthoek?

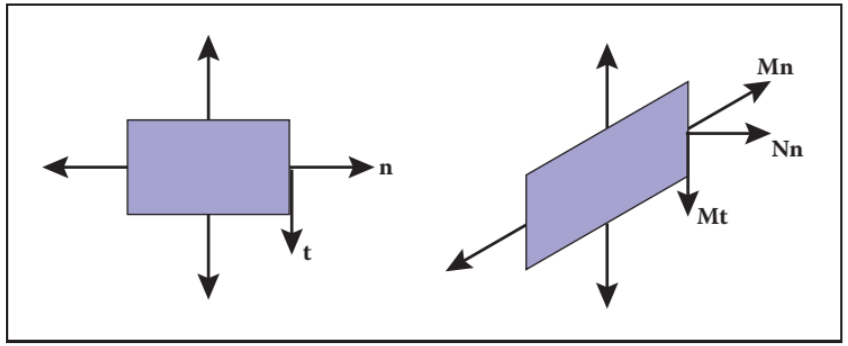



**Figure 6.17.** When a normal vector is transformed using the same matrix that transforms the points on an object, the resulting vector may not be perpendicular to the surface as is shown here for the sheared rectangle. The tangent vector, however, does transform to a vector tangent to the transformed surface.

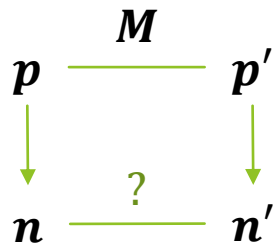
# Transformeren van normaalvectoren

Vergelijking vlak $n_1x + n_2y + n_3z + n_4 = 0$

$$\boldsymbol{n} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \\ n_4 \end{pmatrix} \text{ en } \boldsymbol{p} = \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Vlak: $\boldsymbol{n}^T\boldsymbol{p} = 0$ (vectorproduct)

# Waar gaat $\boldsymbol{n}$ in over?




$\boldsymbol{n'} = (\boldsymbol{M}^{-1})^T \boldsymbol{n}$ als $M^{-1}$ bestaat

Als $\boldsymbol{M}$ transformatiematrix van punt,

dan $(\boldsymbol{M}^{-1})^T$ transformatiematrix van normaalvector

# Rekenvoorbeeld

Gegeven: matrix $\boldsymbol{M} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$ (y-shear)

Gevraagd: matrix $\boldsymbol{N}$ voor normaalvectoren?

$$\boldsymbol{N} = (\boldsymbol{M}^{-1})^T = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}$$

Transformeer $\boldsymbol{n} = (0, 1)$

$$\boldsymbol{Nn} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

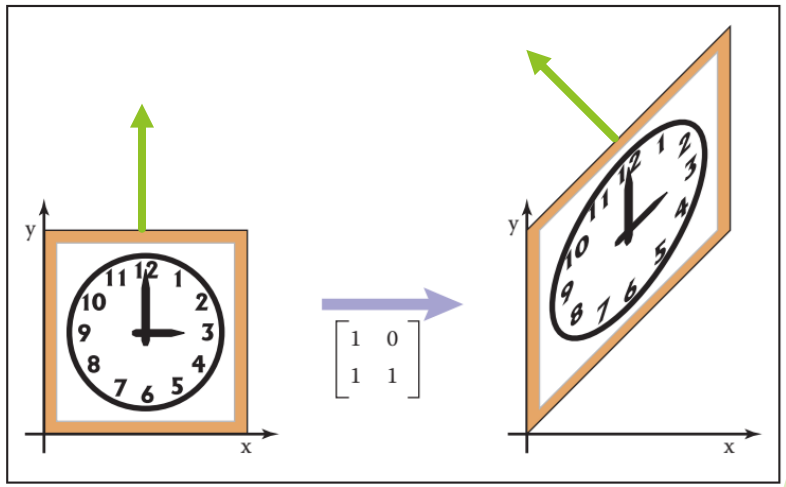



**Figure 6.4.** A *y*-shear matrix moves points up in proportion to their *x*-coordinate.

# Orthonormale basis en veranderen coördinatensysteem

# 1. Transformaties van vectoren

Gegeven $\boldsymbol{e_1} = \begin{pmatrix}1\\0\end{pmatrix}$ en $\boldsymbol{e_2} = \begin{pmatrix}0\\1\end{pmatrix}$ en afbeelding $M$:

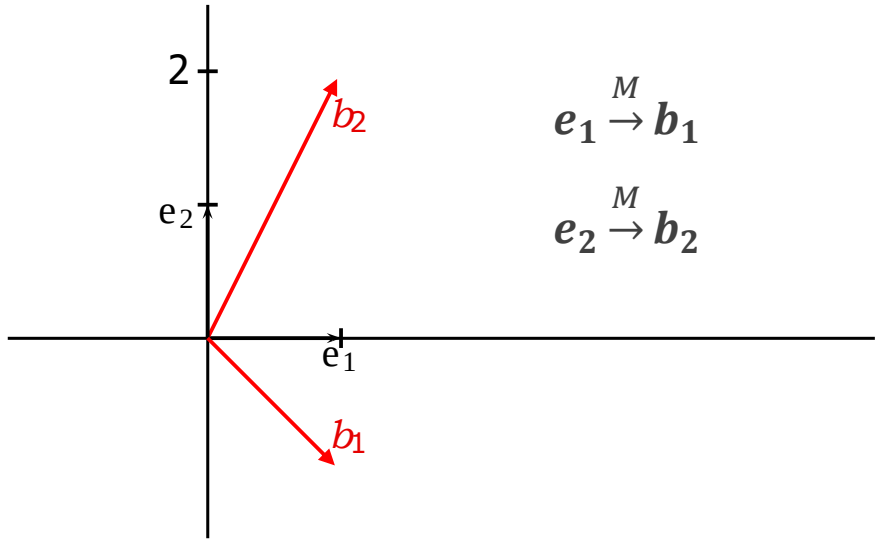
$M\boldsymbol{e_1} = \boldsymbol{b_1}, \quad M\boldsymbol{e_2} = \boldsymbol{b_2}$

Wat is $M$ ?

$$M\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}b_1\\b_1'\end{pmatrix} \qquad M\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}b_2\\b_2'\end{pmatrix} \Rightarrow$$

$$M = \begin{pmatrix}b_1 & b_2\\b_1' & b_2'\end{pmatrix}$$

# 1. Transformaties van vectoren




$$\boldsymbol{e_1} \overset{M}{\rightarrow} \boldsymbol{b_1}$$

$$\boldsymbol{e_2} \overset{M}{\rightarrow} \boldsymbol{b_2}$$

$$M = \begin{pmatrix} 1 & 1 \\ -1 & 2 \end{pmatrix}$$

# 2. Veranderen coördinatensysteem

Kies als basis $B = \{b_1, b_2\}$ met $b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$ en $b_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$

Stel de coördinaten van punt $P$ t.o.v. $B$ zijn $P_B = \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}$

$$P = \beta_1 b_1 + \beta_2 b_2 = \begin{pmatrix} b_1 & b_2 \\ | & | \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = M \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}$$

$P = M \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}$ zijn coördinaten van P t.o.v. $\boldsymbol{e_1, e_2}$

$$P = b_1 + b_2 = 1 \begin{pmatrix} 1 \\ -1 \end{pmatrix} + 1 \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix}_B = \begin{pmatrix} 2 \\ 1 \end{pmatrix}_E$$

# 2. Veranderen coördinatensysteem

$$M_{B \to E}$$

$$\begin{pmatrix} \epsilon_1 \\ \epsilon_2 \end{pmatrix}_{e_1, e_2} = M \begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}_{b_1, b_2}$$

Dus $M$ zet een punt t.o.v. $b_1, b_2$ in een punt t.o.v. $e_1, e_2$

$$\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}_{b_1, b_2} = M^{-1} \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \end{pmatrix}_{e_1, e_2}$$

Dus $M^{-1}$ zet een punt t.o.v. $e_1, e_2$ in een punt t.o.v. $b_1, b_2$

1. Transformaties van vectoren

$$\boldsymbol{e_1} \overset{M}{\rightarrow} \boldsymbol{b_1}$$

$$\boldsymbol{e_2} \overset{M}{\rightarrow} \boldsymbol{b_2}$$

2. Veranderen coördinatensysteem

$$\begin{pmatrix}\beta_1\\ \beta_2\end{pmatrix}_{b_1,b_2} \overset{M}{\rightarrow} \begin{pmatrix}\epsilon_1\\ \epsilon_2\end{pmatrix}_{e_1,e_2}$$

# Rekenvoorbeeld

$$\boldsymbol{e_1} \overset{M?}{\rightarrow} b_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \qquad \boldsymbol{e_2} \overset{M?}{\rightarrow} b_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix} \Rightarrow M = \begin{pmatrix} 1 & 1 \\ -1 & 2 \end{pmatrix}$$

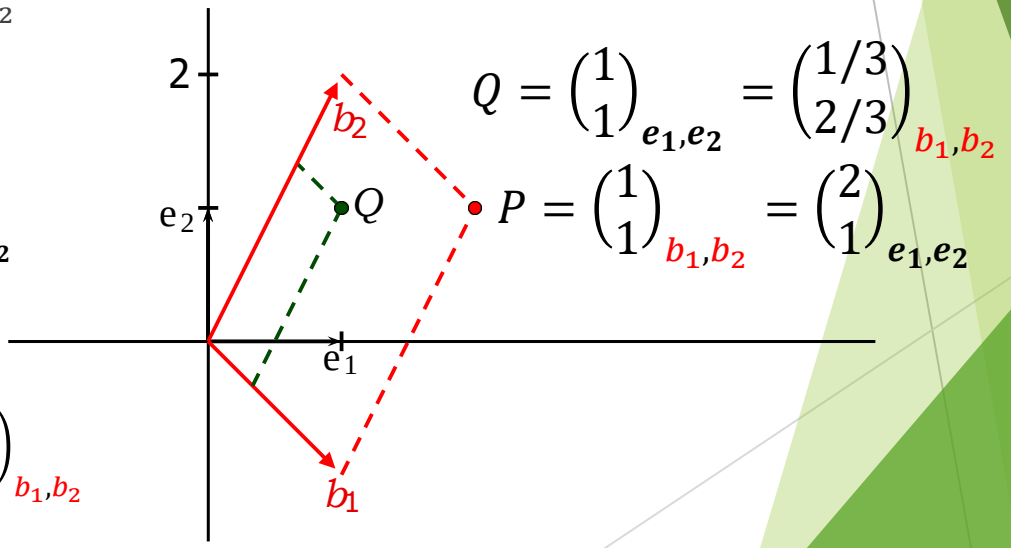
$$\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix}_{b_1,b_2} \overset{M}{\rightarrow} \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \end{pmatrix}_{e_1,e_2}$$

Gegeven $P = \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{b_1,b_2}$

Wat is $P$ t.o.v. $(\boldsymbol{e_1}, \boldsymbol{e_2})$? $P = \begin{pmatrix} 1 & 1 \\ -1 & 2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{b_1,b_2} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}_{\boldsymbol{e_1},\boldsymbol{e_2}}$

Gegeven $Q = \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{\boldsymbol{e_1},\boldsymbol{e_2}}$

Wat is $Q$ t.o.v. $(b_1, b_2)$? $Q = \frac{1}{3}\begin{pmatrix} 2 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{\boldsymbol{e_1},\boldsymbol{e_2}} = \begin{pmatrix} 1/3 \\ 2/3 \end{pmatrix}_{b_1,b_2}$




$$Q = \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{\boldsymbol{e_1},\boldsymbol{e_2}} = \begin{pmatrix} 1/3 \\ 2/3 \end{pmatrix}_{b_1,b_2}$$

$$P = \begin{pmatrix} 1 \\ 1 \end{pmatrix}_{b_1,b_2} = \begin{pmatrix} 2 \\ 1 \end{pmatrix}_{\boldsymbol{e_1},\boldsymbol{e_2}}$$

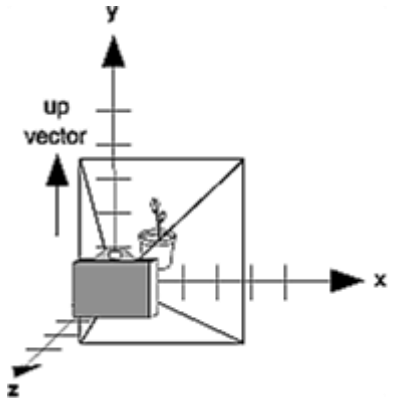
# gluLookAt()

Positioneren van een camera

# Positioneren van de camera

`gluLookAt` default positie: camera in oorsprong, kijkend in negatieve z-richting, en positieve y-as rechtop

```
gluLookAt(0.0, 0.0, 0.0,      // eye
          0.0, 0.0, -100.0,   // center
          0.0, 1.0, 0.0);     // up vector
```

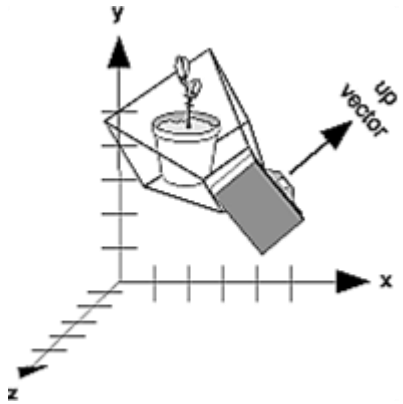



z-waarde kan elke negatieve waarde van z zijn

# Zelf gedefinieerde positie

positie: camera in $(4, 2, 1)$, kijkend naar $(2, 4, -3)$, oriëntatie vector $(2, 2, -1)$

```
gluLookAt(4.0, 2.0, 1.0,
          2.0, 4.0, -3.0,
          2.0, 2.0, -1.0);
```



# gluLookAt / m4.lookAt

Wat doet `gluLookAt(eye, lookat, up)`?

Bouwt matrix die wereldcoördinaten omzet in eye/camera-coördinaten

eye/camera coördinatensysteem:

$w = eye - lookat$

$u = up \times w$

$v = w \times u$

We zoeken rotatiematrix R die: $u \overset{R}{\rightarrow} x$ $v \overset{R}{\rightarrow} y$ $w \overset{R}{\rightarrow} z$

$R$ voert dan wereld- over in camera-coördinaten

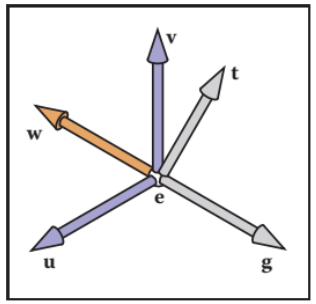



**Figure 7.6.** The user specifies viewing as an eye position **e**, a gaze direction **g**, and an up vector **t**. We construct a right-handed basis with **w** pointing opposite to the gaze and **v** being in the same plane as **g** and **t**.

# gluLookAt / m4.lookAt

We zoeken rotatiematrix R die: $u \xrightarrow{R} x \qquad v \xrightarrow{R} y \qquad w \xrightarrow{R} z$

We weten dat $u$, $v$ en $w$ orthonormaal stelsel van vectoren

$$R = \begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{pmatrix}$$

$$\begin{pmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{pmatrix} \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

# Transformatie die wereldcoördinaten overvoert in cameracoördinaten

Eerst translatie van $\boldsymbol{e}$ naar oorsprong, dan gevonden rotatie $R$

$$\begin{pmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ w_x & w_y & w_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -e_x \\ 0 & 1 & 0 & -e_y \\ 0 & 0 & 1 & -e_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

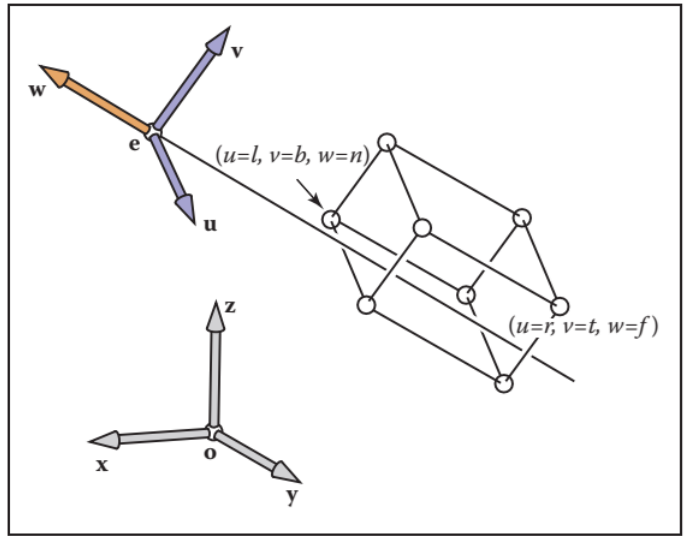



**Figure 7.7.** For arbitrary viewing, we need to change the points to be stored in the "appropriate" coordinate system. In this case it has origin **e** and offset coordinates in terms of **uvw**.

# Wat doet gluLookAt(oog, lookat, up)?

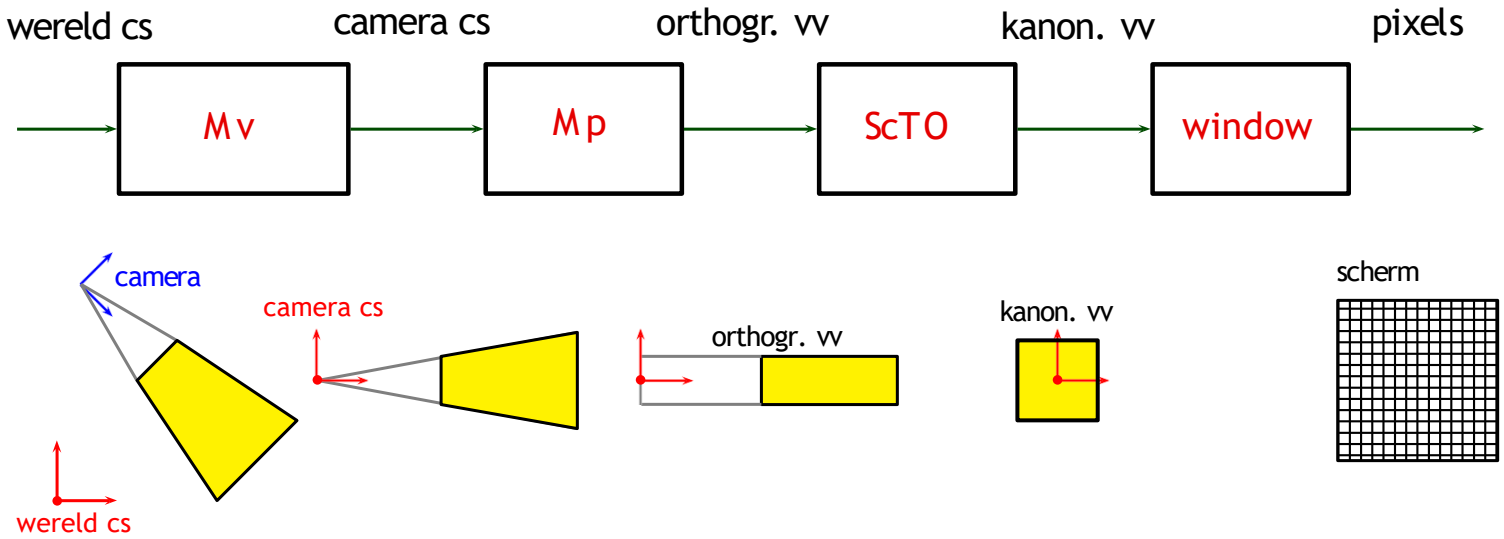
```
eye  =  [4, 4, 4];
look =  [0, 1, 0];
up   =  [0, 1; 0];

w = eye - look;  w = w/norm(w);
u = cross(up,w); u = u/norm(u);
v = cross(w,u);  v = v/norm(v);
dx = -eye'*u
dy = -eye'*v
dz = -eye'*w
V = [u' dx; v' dy; w' dz; 0 0 0 1]
```

# gluLookAt

```
gluLookAt(4,4,4, 0,1,0, 0,1,0);
glGetFloatv(GL_MODELVIEW_MATRIX, mat);
```

$$\begin{pmatrix} 0.7071 & 0 & -0.7071 & 0 \\ -0.3313 & 0.8835 & -0.3313 & -0.8835 \\ 0.6247 & 0.4685 & 0.6247 & -6.8716 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

# Grafische pijplijn



# Grafische pijplijn

"To specify viewing, modeling, and projection transformations, you construct a 4 × 4 matrix $M$, which is then multiplied by coordinates of each vertex $v$ in the scene to accomplish the transformation $v' = Mv$."

# Welke ModelViewmatrix onstaat hier?

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glMultMatrixf(N);
glMultMatrixf(M);
glMultMatrixf(L);
glBegin(GL_POINTS);
glVertex3f(v);
glEnd();
```

$$v' = NML(v)$$

De laatste transformatie in de code wordt als eerste op de vertices uitgevoerd

# Gimbal lock

# Wat is gimbal lock?

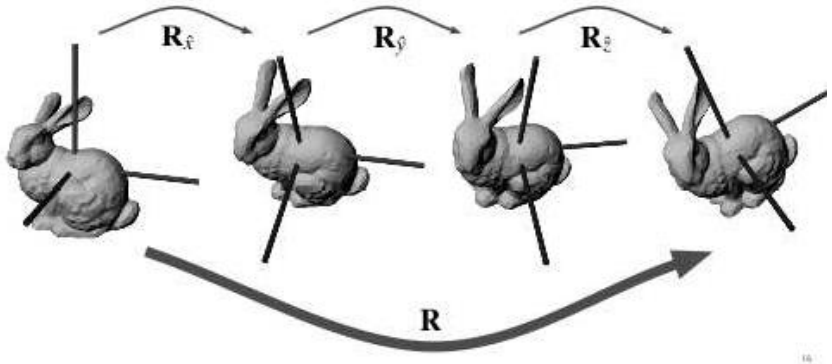
- Gimbal lock treedt op bij rotatie van object met Euler rotatiematrices
- Gimbal lock is fenomeen waarbij twee rotatie-assen van object in dezelfde richting staan
  - het gevolg is dat het object niet meer roteert zoals je zou verwachten
- Gimbal lock is frustrerend probleem van iedere beginnende CG artiest
- Zie ook http://en.wikipedia.org/wiki/Gimbal_lock

# Euler rotaties om oriëntatie aan te geven

- Willekeurige rotatie kan samengesteld worden door 3 rotatiematrices om as:

  $R = R_z R_y R_x$

- Resulterende oriëntatie is afhankelijk van de volgorde van rotaties
- De volgorde maakt uit! XYZ, XZY, YXZ, YZX, ZXY, ZYX



# Gimbal lock

- Gimbal lock ontstaat als assen van twee gimbals in dezelfde richting staat: één vrijheidsgraad verloren
- Gimbal is hardware implementatie van Euler hoeken

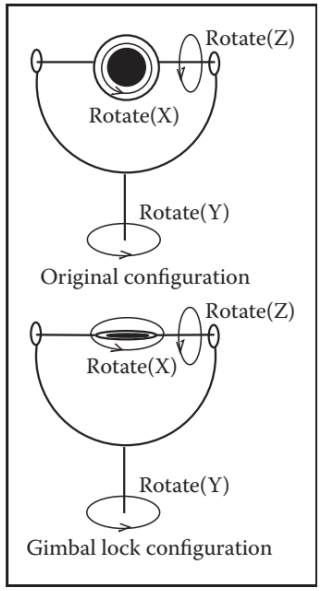


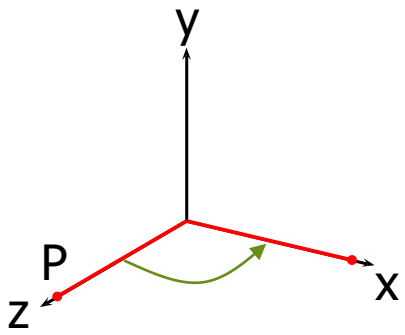

**Figure 16.11.** In this example, gimbal lock occurs when a 90 degree turn around axis Z is made. Both X and Y rotations are now performed around the same axis leading to the loss of one degree of freedom.


Gimbal used in the Apollo spacecraft.
Source: Pinterest

# Wiskunde van gimbal lock

Gimbal lock ontstaat als as wordt overgevoerd in andere as

Bijvoorbeeld rotatie over 90° om y-as voert z-as over in x-as ⇒ rotatie om x-as = rotatie om z-as



# 3D rotatiematrices

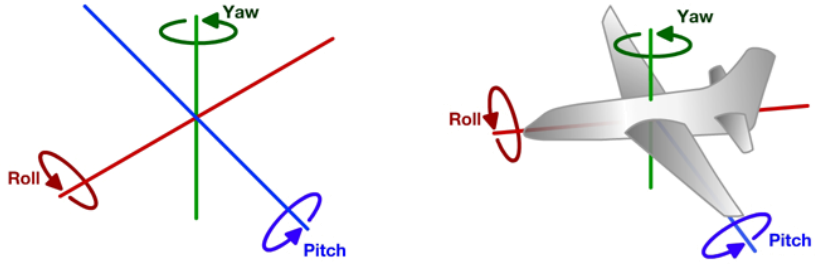
$$\boldsymbol{R}_x(\text{pitch}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix}$$

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

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

Volgorde is belangrijk!

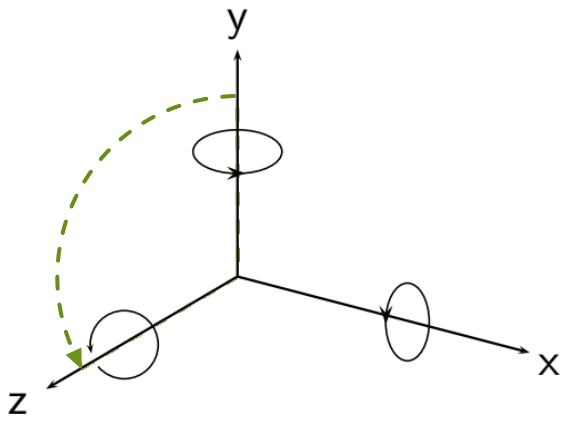
$$R = R_{pitch} R_{yaw} R_{roll}$$




The position of all three axes, with the right-hand rule for its rotations.
Source: touringmachine.com

# $R_x(90°)$ voert y-as over in z-as

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos 90° & -\sin 90° \\ 0 & \sin 90° & \cos 90° \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$



# Voorbeeld: $R_z(r)R_x(90°)R_y(h)$

Bekijk de volgende 3 rotaties:

1. rotatie om y-as met hoek $h$
2. rotatie om x-as met hoek 90°
3. rotatie om z-as met hoek $r$

# Voorbeeld: $R_z(r)R_x(90°)R_y(h)$

$$\begin{pmatrix} \cos r & -\sin r & 0 \\ \sin r & \cos r & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos 90° & -\sin 90° \\ 0 & \sin 90° & \cos 90° \end{pmatrix} \begin{pmatrix} \cos h & 0 & \sin h \\ 0 & 1 & 0 \\ -\sin h & 0 & \cos h \end{pmatrix} =$$

$$\begin{pmatrix} \cos (r+h) & -\sin (r+h) & 0 \\ \sin (r+h) & \cos (r+h) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Deze matrix is maar van één hoek (r+h) afhankelijk ⇒

Dit heet gimbal lock: één vrijheidsgraad is verloren

# Quaterions om oriëntatie weer te geven

- $q = (s, x, y, z)$
- $q = (s; \boldsymbol{v})$ met $\boldsymbol{v} = (x, y, z)$
- $|q| = \sqrt{s^2 + \boldsymbol{v}^2} = 1$
- Een rotatie met hoek $\phi$ om een genormaliseerde vector $\boldsymbol{n}$ wordt weergegeven door:

  $q = (\cos\left(\frac{\phi}{2}\right); \sin\left(\frac{\phi}{2}\right)\boldsymbol{n})$
- Roteren van punt $\boldsymbol{p}$:
  - Quaternion notatie van $\boldsymbol{p}$ is $q_p = (0;\ \boldsymbol{p})$
  - $q'_p = q \cdot q_p \cdot q^{-1} = (0; \boldsymbol{w})$, waar $q^{-1} = \left(\frac{1}{|q|}\right)(s; -\boldsymbol{v})$
  - $\boldsymbol{w}$ is het geroteerde punt

Tip!
YouTube video:
10 mins GameDev tips - Quaternions

# Tentamenvragen

1. Gegeven een orthonormale basis $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$. Wat voor matrix is

   $\boldsymbol{M_{uvw}} = \begin{pmatrix} x_u & y_u & z_u \\ x_v & y_v & z_v \\ x_w & y_w & z_w \end{pmatrix}$ en waar worden $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ in overgevoerd?

2. Wat is de inverse van $\boldsymbol{M_{uvw}}$?

3. Welke matrix voert de vectoren $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$ over in de vectoren $\boldsymbol{a} = (1,2,3), \boldsymbol{b} = (1,1,1)$ en $\boldsymbol{c} = (2,1,0)$?

# Verzin een tentamenvraag!

- Verzin een tentamenvraag n.a.v. dit college
- Dien in op dit formulier
- Als je vraag wordt gebruikt tijdens een (deel)tentamen ontvang jij een half punt extra op je eindcijfer voor dat (deel)tentamen (bv.: had je een 7, dan krijg je een 7,5)!*

*: Vragen die voorgaande jaren al eens op een (deel)tentamen voor dit vak zijn gesteld zijn van deze regeling uitgesloten.