Computação Gráfica I

CRAb – Grupo de Computação Gráfica

Departamento de Computação

UFC

Objetivos

- **Estudar**
  - **equipamentos, técnicas de programação e conceitos matemáticos**
- **Para**
  - **representação, manipulação e projeção de objetos bi- e tridimensionais**
  - **aplicar a problemas específicos**

Sumário do Curso

- Sistemas Gráficos e Modelos
- Programação Gráfica
- Input e Interação
- **Objetos Geométricos e Transformações**
- Visualização
- Pintura
- Técnicas Discretas
- Implementação de um Renderizador

Objetos Geométricos e Transformações

4.1 Escalares, Pontos e Vetores

4.1.1 A visão geométrica

- **Ponto**
  - Não tem tamanho nem forma
  - Única propriedade é sua localização
- **Escalares**
  - Tensores de ordem zero (array de um elemento)
  - Valor não muda com mudança de sistema coordenadas
  - Obedecem conjunto de operações
  - Ex: massa de uma partícula

4.1 Escalares, Pontos e Vetores

4.1.1 A visão geométrica

- **Vetores**
  - **Tensores de primeira ordem (array unidimensional com 3^1 componentes)**
  - **Têm módulo, direção e sentido**
  - **Não têm posição fixa no espaço**
  - **Transformam do sistema de coordenadas e_1 e_2 e_3 para e'_1 e'_2 e'_3 de acordo com a lei**

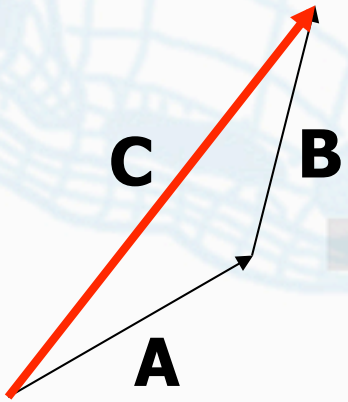
$$V'_{e'_i} = p_{e'_i e_j} V_{e_j}$$

$$\text{onde } p_{e'_i e_j} = \cos(e'_i, e_j)$$

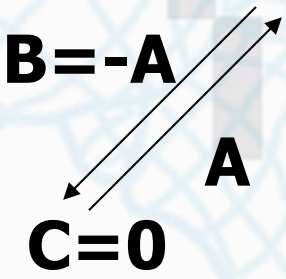
4.1 Escalares, Pontos e Vetores

4.1.1 A visão geométrica

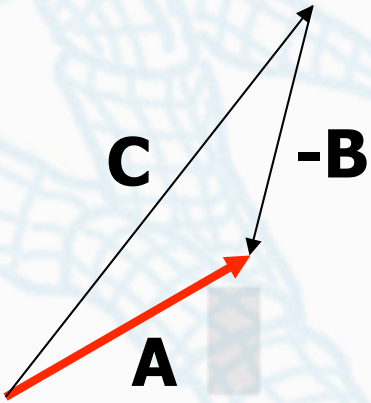
- **Vetores**



Adição vetorial



Vetor nulo

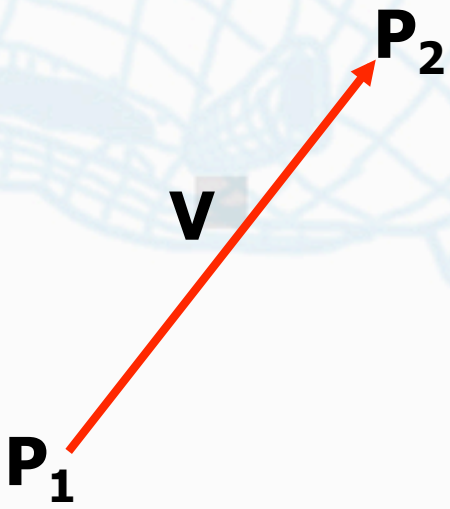


Subtração vetorial

4.1 Escalares, Pontos e Vetores

4.1.1 A visão geométrica

- Adição Ponto-Vetor (Translação do Ponto)

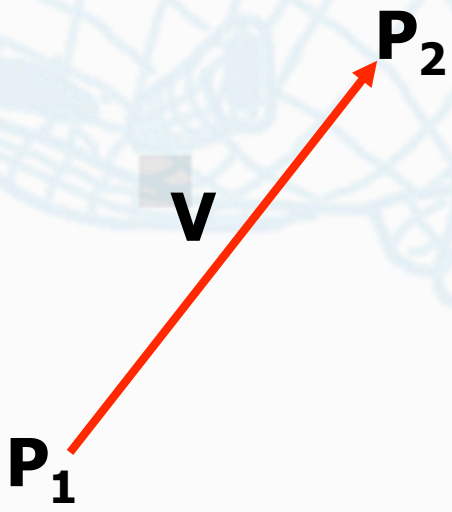


$P_2 = P_1 + V$

4.1 Escalares, Pontos e Vetores

4.1.1 A visão geométrica

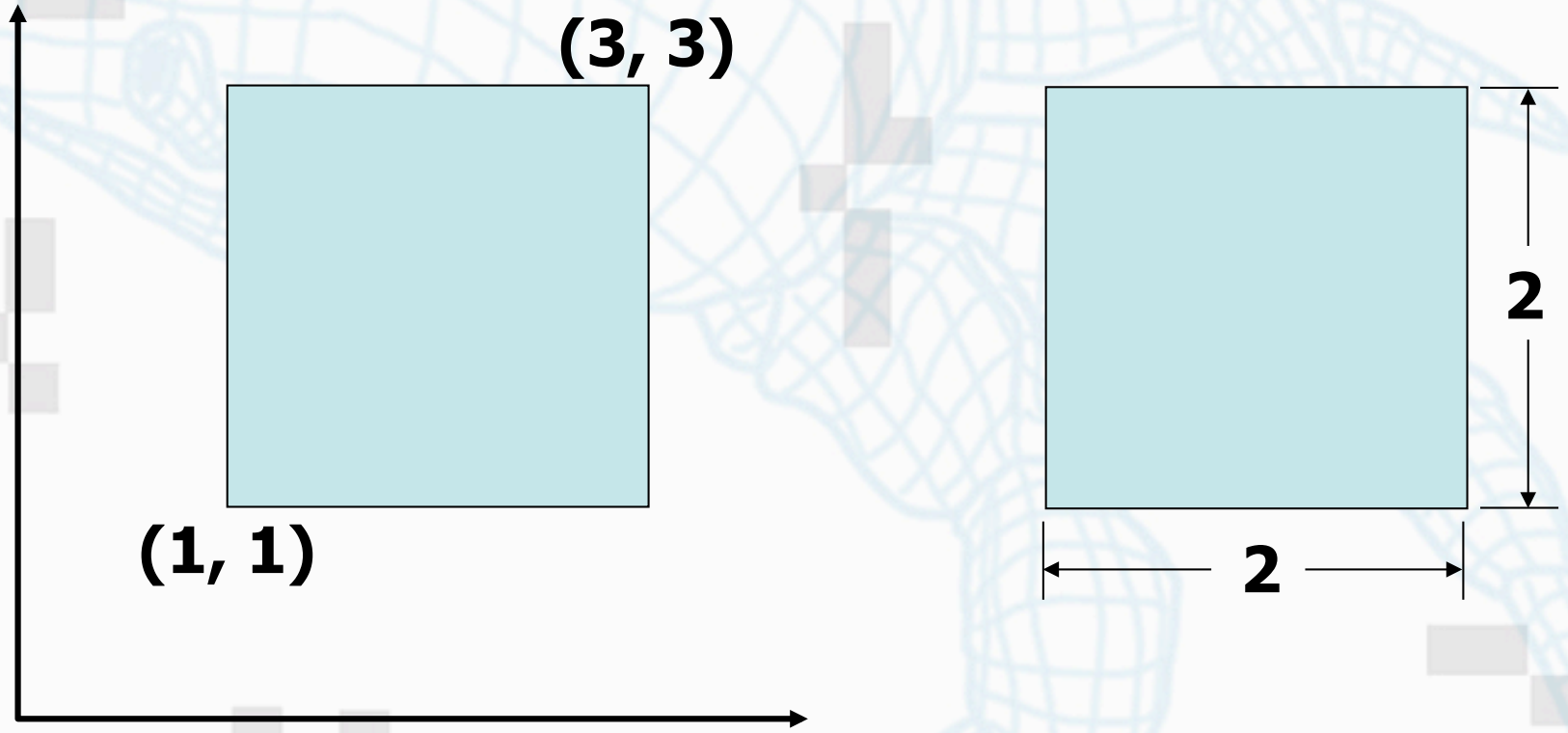
- **Subtração Ponto-Ponto**
  - **Determinação do vetor entre os pontos**



$$V = P_2 - P_1$$

4.1 Escalares, Pontos e Vetores

4.1.2 Geometria livre de coordenadas



4.1 Escalares, Pontos e Vetores

4.1.3 Visão matemática: Espaços vetoriais e Espaços afins

- Escalares, Pontos e Vetores são elementos de conjuntos especiais
  - Escalares $\in$ Campos Escalares
  - Vetores $\in$ Espaços Vetoriais
  - Pontos + Vetores $\in$ Espaços Afins

4.1 Escalares, Pontos e Vetores

- **Campo escalar, {S; +, .}**
  - **S: conjunto cujos elementos são escalares**
  - **Operações binárias: adição(+), multiplicação(.)**
  - **Propriedades**

$\forall \alpha,\beta \in S,\ \alpha + \beta \in S$ e $\alpha \,.\, \beta \in S$ (Fechamento c.r.a. + e .)

$\alpha + \beta = \beta + \alpha$ (Comutativa c.r.a. +)

$\alpha \,.\, \beta = \beta \,.\, \alpha$ (Comutativa c.r.a. .)

$\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma$ (Associativa c.r.a. +)

$\alpha \,.\, (\beta \,.\, \gamma) = (\alpha \,.\, \beta) \,.\, \gamma$ (Associativa c.r.a. .)

$\alpha \,.\, (\beta + \gamma) = (\alpha \,.\, \beta) + (\alpha \,.\, \gamma)$ (Distributiva . c.r.a. +)

$\alpha + 0 = 0 + \alpha = \alpha,\ \forall \alpha \in S$ (Elemento neutro da +)

$\alpha \,.\, 1 = 1 \,.\, \alpha = \alpha,\ \forall \alpha \in S$ (Elemento neutro da .)

$\alpha + (-\alpha) = 0,\ \forall \alpha \in S$ (Inverso aditivo)

$\alpha \,.\, \alpha^{-1} = 1,\ \forall \alpha \in S$ (Inverso multiplicativo)

4.1 Escalares, Pontos e Vetores

- **Espaço Vetorial, X sobre o Campo escalar,S**
  - **X conjunto de vetores**
  - **{X; +}: Grupo Abeliano**
    - **Obedece às seguintes propriedades $\forall x$ e $y \in X$**

Se $x + x = x \rightarrow x = 0$ (vetor nulo)
Se $z \in X$ e $x + y = x + z \rightarrow y = z$
Se $z \in X$ e $x + y = z + y \rightarrow x = z$
$\exists w \in X \mid w + x = y$
$\exists z \in X \mid x + z = y$

Propriedades de Um grupo

$x + y = y + x$

  - **X é um S-módulo, isto é**

$(\alpha + \beta)x = \alpha x + \beta x$
$\alpha(x + y) = \alpha x + \alpha y$
$\alpha\ (\beta x) = (\alpha\ .\ \beta)x$
$1x = x$

4.1 Escalares, Pontos e Vetores

- **Espaço Euclidiano**
  - **Espaço vetorial com uma medida de tamanho ou distância**

    d(x,y) = Sqrt(x . y)
- **Espaço Afim**
  - **Extensão do espaço vetorial, incluindo o objeto Ponto**
  - **Inclui as operações**
    - **Adição vetor-ponto**
    - **Subtração ponto-ponto**

4.1 Escalares, Pontos e Vetores

4.1.4 A visão da ciência da computação

- Tipos abstratos de dados
  - **vector** u, v;
  - **point** p, q;
  - **scalar** a, b;
- Linguagens orientadas a objetos (C++)
  - Classes
  - Overloading de operadores
    - q = p + a*v;

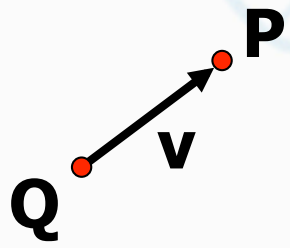
4.1 Escalares, Pontos e Vetores

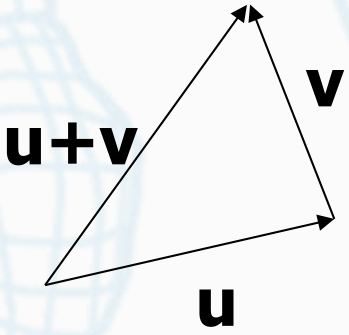
4.1.5 Tipos Abstratos de Dados geométricos

- Escalares: conjunto dos números reais
  - Operações de adição e multiplicação
- Pontos: posições no espaço 3D
- Vetores: segmentos de retas, direcionais

$v = P - Q$

$P = v + Q$





4.1 Escalares, Pontos e Vetores

4.1.6 Linhas

- Lugar geométricos dos pontos gerados por operações ponto-vetor

  $\mathbf{P}(\alpha) = \mathbf{P}_0 + \alpha\,\mathbf{d}$ (Forma paramétrica)

  $\mathbf{P}_0$ é um ponto arbitrário

  $\mathbf{d}$ é um vetor arbitrário

  α é um escalar

  - Se α for > 0 $\mathbf{P}(\alpha)$ é o raio que emana de $\mathbf{P}_0$ na direção $\mathbf{d}$

4.1 Escalares, Pontos e Vetores

4.1.7 Somas afins

$P = Q + \alpha v$

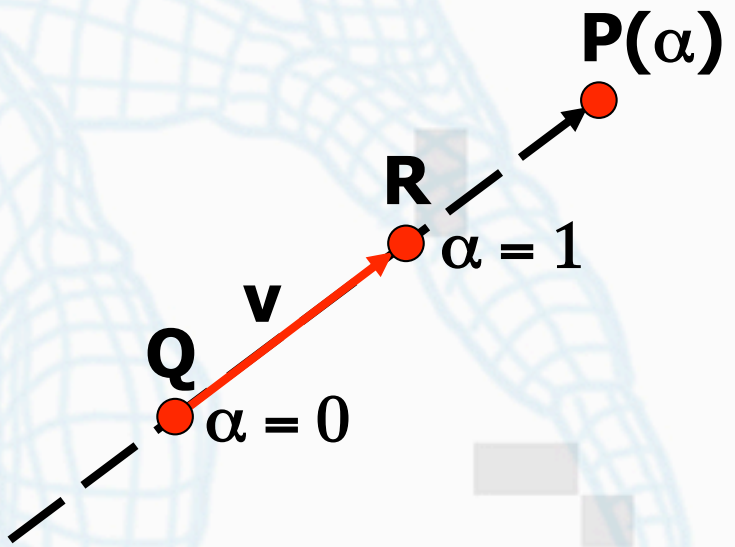
- Possível encontrar um ponto R tal que

$v = R - Q$

- Assim

$$P = Q + \alpha (R - Q)$$
$$= \alpha R + (1 - \alpha)Q$$
$$= \alpha_1 R + \alpha_2 Q$$

$\alpha_1 + \alpha_2 = 1$



4.1 Escalares, Pontos e Vetores

4.1.8 Convexidade

- Segmento de linha é um objeto convexo
- Convex hull (Fecho convexo) dos pontos

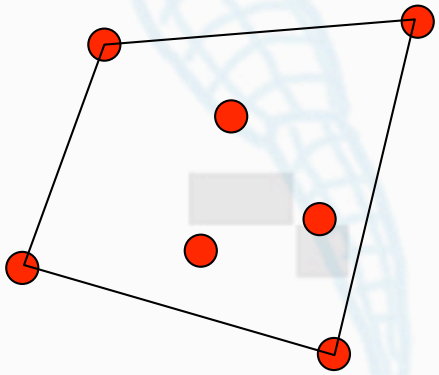
P_i, i = 1, 2, ..., n

é o conjunto de pontos P definido pela soma afim dos n pontos

$P = \alpha_1 P_1 + \alpha_2 P_2 + \ldots + \alpha_n P_n$

$\alpha_1 + \alpha_2 + \ldots + \alpha_n = 1$

$\alpha_i > 0$, i = 1, 2, ..., n



4.1 Escalares, Pontos e Vetores

4.1.9 Produtos escalares e vetoriais

4.1 Escalares, Pontos e Vetores

- **Dot product = Scalar product (a.b)**

$$\mathbf{a}\cdot\mathbf{b} = a_1b_1 + a_2b_2 + a_3b_3$$

$$\text{For } \mathbf{a} = \mathbf{b} \Rightarrow \mathbf{a}\cdot\mathbf{a} = |\mathbf{a}|^2$$

- **Angle between two nonzero vectors**

$$\mathbf{a}\cdot\mathbf{b} = |\mathbf{a}||\mathbf{b}|\cos(\theta), \quad 0 \le \theta \le \pi$$

$$\text{Cauchy - Schwarz inequality}$$

$$|\mathbf{a}\cdot\mathbf{b}| \le |\mathbf{a}||\mathbf{b}|$$

4.1 Escalares, Pontos e Vetores

- **Scalar multiplication satisfies**

[c1] $\mathbf{a}\cdot\mathbf{b}=\mathbf{b}\cdot\mathbf{a}$ (Symmetric Law)

[c2] $(k\mathbf{a})\cdot\mathbf{b}=k(\mathbf{a}\cdot\mathbf{b})$ (k = scalar)

[c3] $\mathbf{a}\cdot(\mathbf{b}+\mathbf{c})=\mathbf{a}\cdot\mathbf{b}+\mathbf{a}\cdot\mathbf{c}$ (Distributive Law)

[c4] Scalar multiplication is positive semidefinite

(i) $\mathbf{a}\cdot\mathbf{a}\geq\mathbf{0}\ \ \forall\mathbf{a}$

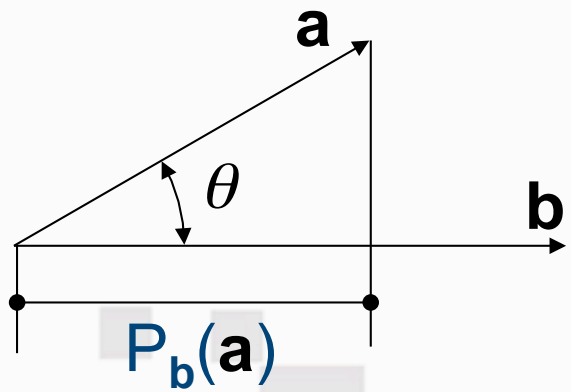
(ii) $\mathbf{a}\cdot\mathbf{a}=\mathbf{0}\Leftrightarrow\mathbf{a}=\mathbf{0}$

4.1 Escalares, Pontos e Vetores

- **Scalar projection of a onto b:** $P_b(a)$

$$P_{\mathbf{b}}(\mathbf{a}) = \frac{(\mathbf{a}\cdot\mathbf{b})}{|\mathbf{b}|} = \mathbf{a}\cdot\frac{\mathbf{b}}{|\mathbf{b}|} = |\mathbf{a}|\cos(\theta)$$

- **Illustration of** $P_b(a)$

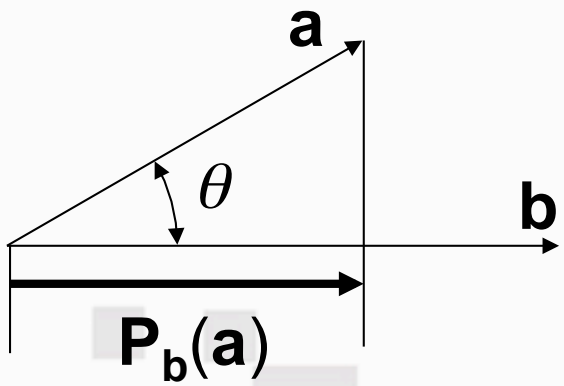


4.1 Escalares, Pontos e Vetores

- **Vector projection of a onto b: $P_b(a)$**

$$\mathbf{P_b(a)} = P_{\mathbf{b}}(\mathbf{a})\mathbf{u_b} = \frac{(\mathbf{a}\cdot\mathbf{b})}{|\mathbf{b}|}\frac{\mathbf{b}}{|\mathbf{b}|} = \frac{(\mathbf{a}\cdot\mathbf{b})\ \ \mathbf{b}}{|\mathbf{b}|^2}$$

- **Illustration of $P_b(a)$**



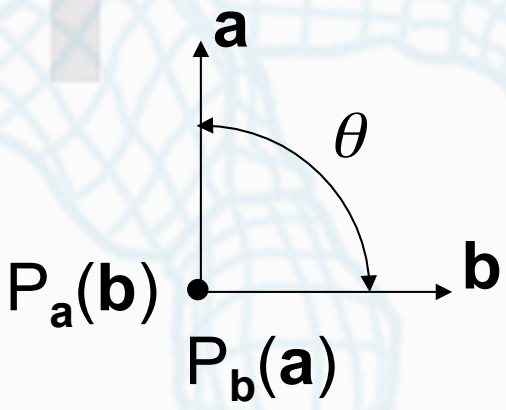
4.1 Escalares, Pontos e Vetores

- **Vectors a and b are orthogonal if**

$$P_{\mathbf{b}}(\mathbf{a}) = P_{\mathbf{a}}(\mathbf{b}) = 0 \Rightarrow$$

$$\Rightarrow |\mathbf{a}|\cos(\theta) = 0 = |\mathbf{b}|\cos(\theta) \Leftrightarrow \cos(\theta) = 0$$

$$\Rightarrow \theta = \frac{\pi}{2} \pm k\pi$$



4.1 Escalares, Pontos e Vetores

- **Basis formed by three mutually orthogonal unit vectors, e_i (i =1, 2, 3)**

$$\mathbf{e}_i \cdot \mathbf{e}_j = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

δ_{ij} - called the Kronecker symbol

4.1 Escalares, Pontos e Vetores

Theorem 1.4: Let $\mathbf{e}_1$, $\mathbf{e}_2$, $\mathbf{e}_3$ be an orthonormal basis and let

$\mathbf{a} = a_1\mathbf{e}_1 + a_2\mathbf{e}_2 + a_3\mathbf{e}_3$

$\mathbf{b} = b_1\mathbf{e}_1 + b_2\mathbf{e}_2 + b_3\mathbf{e}_3$. Then

$$\text{(i)}\ \mathbf{a}\cdot\mathbf{b} = a_1b_1 + a_2b_2 + a_3b_3 = \sum_{i=1}^{3} a_ib_i$$

$$\text{(ii)}\ |\mathbf{a}| = \sqrt{\mathbf{a}\cdot\mathbf{a}} = \sqrt{a_1^2 + a_2^2 + a_3^2} = \sqrt{\sum_{i=1}^{3} a_ia_i}$$

$$\text{(iii)}\ a_i = \mathbf{a}\cdot\mathbf{e}_i$$

4.1 Escalares, Pontos e Vetores

Let (u_1, u_2, u_3) and (v_1, v_2, v_3) be ordered bases

and let $$\mathbf{v}_j = \sum_{i=1}^{3} a_{ij}\mathbf{u}_i$$

Then (v_1, v_2, v_3) has the same orientation of (u_1, u_2, u_3) if

$\det(a_{ij}) > 0$

Right-handed basis and opposite orientation

Left-handed basis

4.1 Escalares, Pontos e Vetores

- **Vector product = a x b**

$$\mathbf{a}\times\mathbf{b}=\begin{vmatrix}\mathbf{e}_1 & a_1 & b_1\\ \mathbf{e}_2 & a_2 & b_2\\ \mathbf{e}_3 & a_3 & b_3\end{vmatrix}=\mathbf{e}_1\begin{vmatrix}a_2 & b_2\\ a_3 & b_3\end{vmatrix}-\mathbf{e}_2\begin{vmatrix}a_1 & b_1\\ a_3 & b_3\end{vmatrix}+\mathbf{e}_2\begin{vmatrix}a_1 & b_1\\ a_2 & b_2\end{vmatrix}$$

$$\mathbf{a}\times\mathbf{b}=\left(a_2b_3-a_3b_2\right)\ \mathbf{e}_1+\left(a_3b_1-a_1b_3\right)\ \mathbf{e}_2+\left(a_1b_2-a_2b_1\right)\ \mathbf{e}_3$$

4.1 Escalares, Pontos e Vetores

- **Theorem 1.5**

(i) $\|\mathbf{a}\times\mathbf{b}\| = |\mathbf{a}||\mathbf{b}|\sin\theta$ where $\theta = \angle(\mathbf{a},\mathbf{b})$

(ii) a) $(\mathbf{a}\times\mathbf{b})\perp\mathbf{a}$ and $(\mathbf{a}\times\mathbf{b})\perp\mathbf{b}$

b) If $\mathbf{a}\times\mathbf{b}\neq 0$, then $(\mathbf{a},\mathbf{b},\mathbf{a}\times\mathbf{b})$ is a right-handed linearly independent triplet

- **Theorem 1.6**

$\mathbf{a}\times\mathbf{b} = \mathbf{0}$ if and only if $\mathbf{a}$ and $\mathbf{b}$ are linearly dependent

4.1 Escalares, Pontos e Vetores

- **Vector products satisfies**

$$[E_1] \quad \mathbf{a}\times\mathbf{b} = -(\mathbf{b}\times\mathbf{a}) \qquad (\text{Anticommutative Law})$$

$$[E_2] \quad \mathbf{a}\times(\mathbf{b}+\mathbf{c}) = \mathbf{a}\times\mathbf{b}+\mathbf{a}\times\mathbf{c} \qquad (\text{Distributive Law})$$

$$[E_3] \quad (k\mathbf{a})\times\mathbf{b} = k(\mathbf{a}\times\mathbf{b}) \qquad (k = \text{scalar})$$

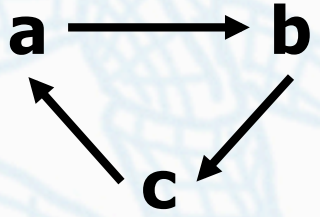
$$[E_4] \quad \mathbf{a}\times\mathbf{a} = \mathbf{0}$$

$$\text{Note}: \ \mathbf{a}\times(\mathbf{b}\times\mathbf{c}) \neq (\mathbf{a}\times\mathbf{b})\times\mathbf{c}$$

4.1 Escalares, Pontos e Vetores

- **Triple scalar product: a•bxc**

$$\mathbf{a}\cdot\mathbf{b}\times\mathbf{c}=\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix}$$



$$\mathbf{a}\cdot\mathbf{b}\times\mathbf{c} \quad = \quad \mathbf{b}\cdot\mathbf{c}\times\mathbf{a} \quad = \quad \mathbf{c}\cdot\mathbf{a}\times\mathbf{b} =$$

$$= -(\mathbf{a}\cdot\mathbf{c}\times\mathbf{b}) \quad = -(\mathbf{b}\cdot\mathbf{a}\times\mathbf{c}) \quad = -(\mathbf{c}\cdot\mathbf{b}\times\mathbf{a})$$

$$[\mathbf{abc}] = \mathbf{a}\cdot\mathbf{b}\times\mathbf{c} = \mathbf{a}\times\mathbf{b}\cdot\mathbf{c}$$

4.1 Escalares, Pontos e Vetores

- **Theorem 1.7: [abc]=0 if and only if a, b, and c are linearly dependent.**
- **Theorem 1.8:**

$$\mathbf{a}\times(\mathbf{b}\times\mathbf{c})=(\mathbf{a}\cdot\mathbf{c})\mathbf{b}-(\mathbf{a}\cdot\mathbf{b})\mathbf{c}$$

$$[F_1]\quad(\mathbf{a}\times\mathbf{b})\cdot(\mathbf{c}\times\mathbf{d})=(\mathbf{a}\cdot\mathbf{c})(\mathbf{b}\cdot\mathbf{d})-(\mathbf{a}\cdot\mathbf{d})(\mathbf{b}\cdot\mathbf{c})$$

$$[F_2]\quad(\mathbf{a}\times\mathbf{b})\times(\mathbf{c}\times\mathbf{d})=[\mathbf{abd}]\,\mathbf{c}-[\mathbf{abc}]\,\mathbf{d}$$

4.1 Escalares, Pontos e Vetores

4.1.10 Planos

- Extensão das linhas paramétricas
- Três pontos não alinhados determinam um plano único

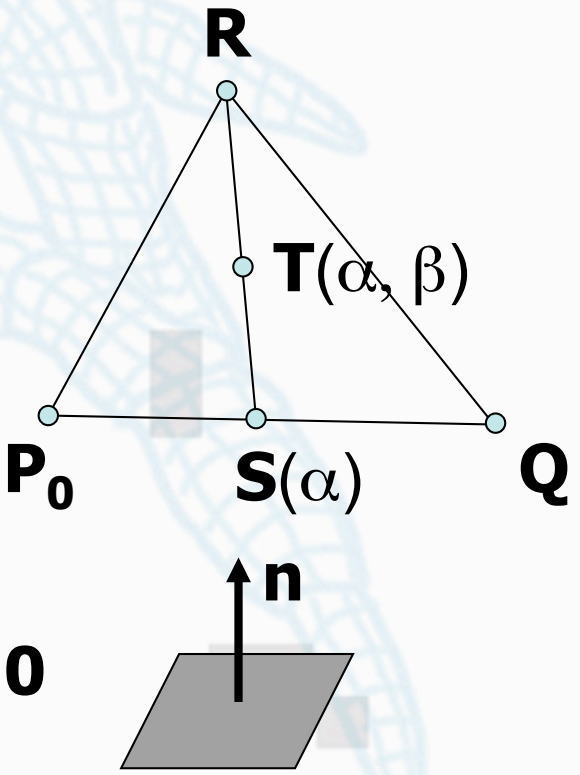
$S(\alpha) = (1-\alpha)\,P_0 + \alpha Q, \quad 0 \le \alpha \le 1$

$T(\beta) = (1-\beta)\,S + \beta R, \quad 0 \le \beta \le 1$

$T(\alpha,\beta) = (1-\beta)\,S + \beta R$

$= P_0 + \alpha(1-\beta)(Q-P_0) + \beta(R-P_0)$

$T(\alpha,\beta) = P_0 + \alpha' u + \beta' v \in \Delta P_0 QR$



- Se P está no plano, então

$P - P_0 = \alpha' u + \beta' v$

$n \cdot (P - P_0) = (u \times v) \cdot (P - P_0) = 0$

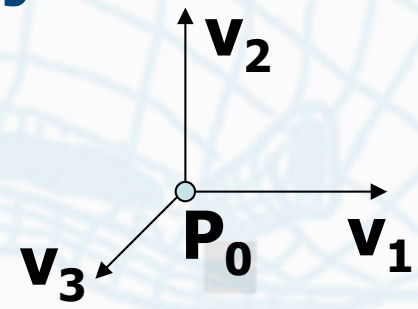
4.2 Primitivas Tridimensionais

- **Assumir**
  - **Objetos descritos por suas superfícies**
  - **Considerar objetos como ocos**
  - **Objetos descritos por um conjunto de vértices**
  - **Superfície pode ser aproximada por polígonos convexos**

4.3 Sistemas de Coordenadas e Frames

4.3.0 Frame

- É especificado por um ponto (origem) e um conjunto de vetores LI (base)



- Todo vetor w e todo ponto P podem ser escritos como

$$w = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3$$

$$P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3$$

4.3 Sistemas de Coordenadas e Frames

4.3.1 Representações e n-tuplas

Suponha que e_1, e_2, e_3 formam uma base

$v = \alpha_1 e_1 + \alpha_2 e_2 + \alpha_3 e_3$

$v = (\alpha_1, \alpha_2, \alpha_3)$ (representação 3-tupla)

4.3 Sistemas de Coordenadas e Frames

4.3.2 Mudanças de sistemas de coordenadas

$$\begin{aligned}
\mathbf{u}_1 &= \gamma_{11}\mathbf{v}_1 + \gamma_{12}\mathbf{v}_2 + \gamma_{13}\mathbf{v}_3 \\
\mathbf{u}_2 &= \gamma_{21}\mathbf{v}_1 + \gamma_{22}\mathbf{v}_2 + \gamma_{23}\mathbf{v}_3 \\
\mathbf{u}_3 &= \gamma_{31}\mathbf{v}_1 + \gamma_{32}\mathbf{v}_2 + \gamma_{33}\mathbf{v}_3
\end{aligned}$$

$$\begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \end{pmatrix} = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{pmatrix} = \mathbf{M} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{pmatrix}$$

4.3 Sistemas de Coordenadas e Frames

4.3.2 Mudanças de sistemas de coordenadas

$$\mathbf{w} = \alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2 + \alpha_3\mathbf{v}_3$$

$$\mathbf{w} = \beta_1\mathbf{u}_1 + \beta_2\mathbf{u}_2 + \beta_3\mathbf{u}_3$$

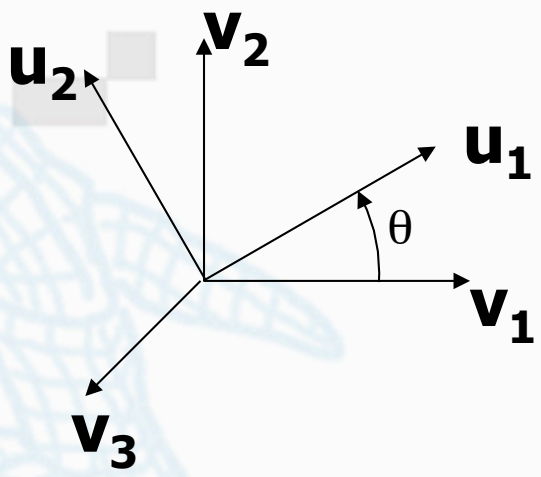
$$\mathbf{w} = \begin{bmatrix}\alpha_1 & \alpha_2 & \alpha_3\end{bmatrix}\begin{pmatrix}\mathbf{v}_1\\ \mathbf{v}_2\\ \mathbf{v}_3\end{pmatrix} = \mathbf{a}^T\begin{pmatrix}\mathbf{v}_1\\ \mathbf{v}_2\\ \mathbf{v}_3\end{pmatrix}$$

$$\mathbf{a}^T = \mathbf{b}^T\mathbf{M} \rightarrow \mathbf{a} = \mathbf{M}^T\mathbf{b}$$

$$\mathbf{w} = \begin{bmatrix}\beta_1 & \beta_2 & \beta_3\end{bmatrix}\begin{pmatrix}\mathbf{u}_1\\ \mathbf{u}_2\\ \mathbf{u}_3\end{pmatrix} = \mathbf{b}^T\begin{pmatrix}\mathbf{u}_1\\ \mathbf{u}_2\\ \mathbf{u}_3\end{pmatrix} = \mathbf{b}^T\mathbf{M}\begin{pmatrix}\mathbf{v}_1\\ \mathbf{v}_2\\ \mathbf{v}_3\end{pmatrix}$$

4.3 Sistemas de Coordenadas e Frames

4.3.3 Exemplo de mudança de representação



$$\begin{pmatrix}\mathbf{u}_1\\ \mathbf{u}_2\\ \mathbf{u}_3\end{pmatrix}=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{pmatrix}\mathbf{v}_1\\ \mathbf{v}_2\\ \mathbf{v}_3\end{pmatrix}$$

$$\mathbf{w}=1\mathbf{v}_1+2\mathbf{v}_2+4\mathbf{v}_3=\begin{bmatrix}1 & 2 & 4\end{bmatrix}\begin{pmatrix}\mathbf{v}_1\\ \mathbf{v}_2\\ \mathbf{v}_3\end{pmatrix}$$

$$\mathbf{w}=\begin{bmatrix}1 & 2 & 4\end{bmatrix}\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{pmatrix}\mathbf{u}_1\\ \mathbf{u}_2\\ \mathbf{u}_3\end{pmatrix}$$

$$\mathbf{w}=(\cos\theta+2\sin\theta)\mathbf{u}_1+(2\cos\theta-\sin\theta)\mathbf{u}_2+4\mathbf{u}_3$$

$$\mathbf{a}=\mathbf{M}^T\mathbf{b}$$

$$\begin{pmatrix}1\\ 2\\ 4\end{pmatrix}=\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{pmatrix}\beta_1\\ \beta_2\\ \beta_3\end{pmatrix}$$

$$\begin{pmatrix}\beta_1\\ \beta_2\\ \beta_3\end{pmatrix}=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{pmatrix}1\\ 2\\ 4\end{pmatrix}$$

4.3 Sistemas de Coordenadas e Frames

4.3.4 Coordenadas homogêneas

- **Ponto**

$$\mathbf{P} = \mathbf{P}_0 + x\,\mathbf{v}_1 + y\,\mathbf{v}_2 + z\,\mathbf{v}_3$$

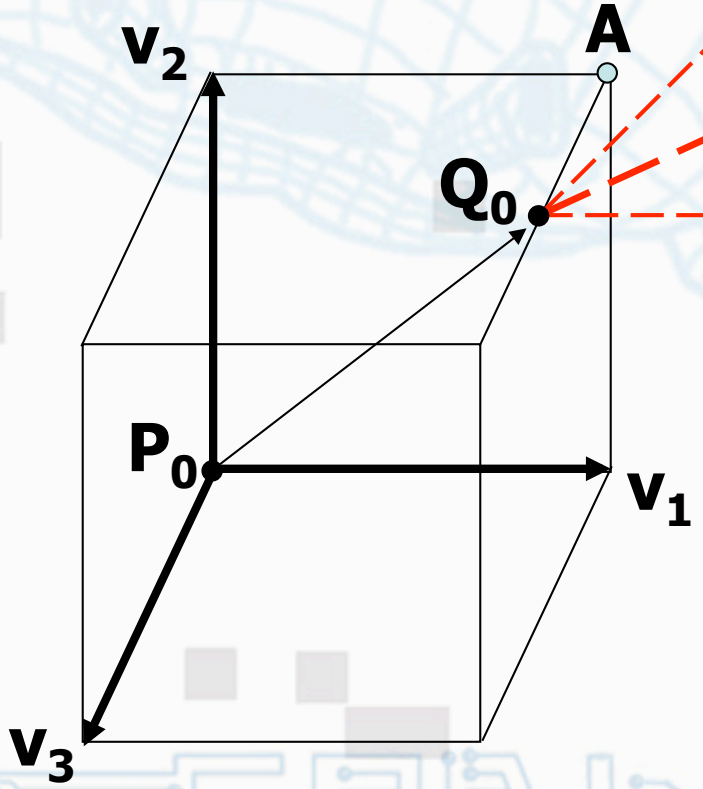
$$\mathbf{P} = \begin{bmatrix} x & y & z & 1 \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P}_0 \end{pmatrix}$$

- **Vetor**

$$\mathbf{w} = \begin{bmatrix} \delta_1 & \delta_2 & \delta_3 & 0 \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P}_0 \end{pmatrix}$$

4.3 Sistemas de Coordenadas e Frames

4.3.5 Exemplo de mudança de frames



$u_2 = v_1 + v_2$

$u_3 = v_1 + v_2 + v_3$

$u_1 = v1$

$A = P_0 + 1\,v_1 + 1\,v_2 + 0\,v_3$

$Q_0 = P_0 + 1\,v_1 + 1\,v_2 + 0.5\,v_3$

$A = Q_0 + \alpha\,u_1 + \beta\,u_2 + \gamma\,u_3$

4.3 Sistemas de Coordenadas e Frames

$$\begin{pmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \mathbf{u}_3 \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{pmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0.5 & 1 \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P}_0 \end{pmatrix} + \begin{bmatrix} \alpha & \beta & \gamma & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P}_0 \end{pmatrix}$$

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & 0 & 1 \end{bmatrix} \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \\ \mathbf{P}_0 \end{pmatrix}$$

4.3 Sistemas de Coordenadas e Frames

$$\begin{pmatrix}1\\1\\0\\1\end{pmatrix}=\begin{pmatrix}1\\1\\0.5\\1\end{pmatrix}+\begin{bmatrix}1&1&1&0\\0&1&1&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{pmatrix}\alpha\\\beta\\\gamma\\0\end{pmatrix}\Rightarrow$$

$$\begin{pmatrix}\alpha\\\beta\\\gamma\\0\end{pmatrix}=\begin{bmatrix}1&-1&0&0\\0&1&-1&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&-1\\0&1&0&-1\\0&0&1&-0.5\\0&0&0&0\end{bmatrix}\begin{pmatrix}1\\1\\0\\1\end{pmatrix}=\begin{bmatrix}1&-1&0&0.0\\0&1&-1&-0.5\\0&0&1&-0.5\\0&0&0&0\end{bmatrix}\begin{pmatrix}1\\1\\0\\1\end{pmatrix}$$

$$\begin{pmatrix}\alpha\\\beta\\\gamma\\0\end{pmatrix}=\begin{pmatrix}0\\0.5\\-0.5\\0\end{pmatrix}$$

4.3 Sistemas de Coordenadas e Frames

4.3.6 Trabalhando com representações

- Considere a mudança de representação do frame $\{i, j, k, P_0\}$ para o frame $\{u, v, w, Q_0\}$ através da transformação

  $\{a\} = [C] \{b\}$

  - $\{b\}$: representação no frame $\{i, j, k, P_0\}$
  - $\{a\}$: representação no frame $\{u, v, w, Q_0\}$
- Considere a mudança inversa

  $\{b\} = [D] \{a\} \Rightarrow [D] = [C]^{-1}$

4.3 Sistemas de Coordenadas e Frames

- **Assim**

$$[\mathbf{D}]\begin{pmatrix}1\\0\\0\\0\end{pmatrix}=\mathbf{u}=\begin{pmatrix}u_i\\u_j\\u_k\\0\end{pmatrix}\quad [\mathbf{D}]\begin{pmatrix}0\\1\\0\\0\end{pmatrix}=\mathbf{v}=\begin{pmatrix}v_i\\v_j\\v_k\\0\end{pmatrix}$$

$$[\mathbf{D}]\begin{pmatrix}0\\0\\1\\0\end{pmatrix}=\mathbf{w}=\begin{pmatrix}w_i\\w_j\\w_k\\0\end{pmatrix}\quad [\mathbf{D}]\begin{pmatrix}0\\0\\0\\1\end{pmatrix}=\mathbf{Q}_0=\begin{pmatrix}q_i\\q_j\\q_k\\1\end{pmatrix}$$

$$\Rightarrow [\mathbf{D}]\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}=[\mathbf{D}]=\begin{bmatrix}u_i&v_i&w_i&q_i\\u_j&v_j&w_j&q_j\\u_k&v_k&w_k&q_k\\0&0&0&1\end{bmatrix}$$

$$[\mathbf{C}]=[\mathbf{D}]^{-1}=\begin{bmatrix}u_i&v_i&w_i&q_i\\u_j&v_j&w_j&q_j\\u_k&v_k&w_k&q_k\\0&0&0&1\end{bmatrix}^{-1}$$

4.3 Sistemas de Coordenadas e Frames

- **Aplicando ao exemplo anterior**

$$[C] = [D]^{1} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & -0.5 \\ 0 & 0 & 1 & -0.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.3 Sistemas de Coordenadas e Frames

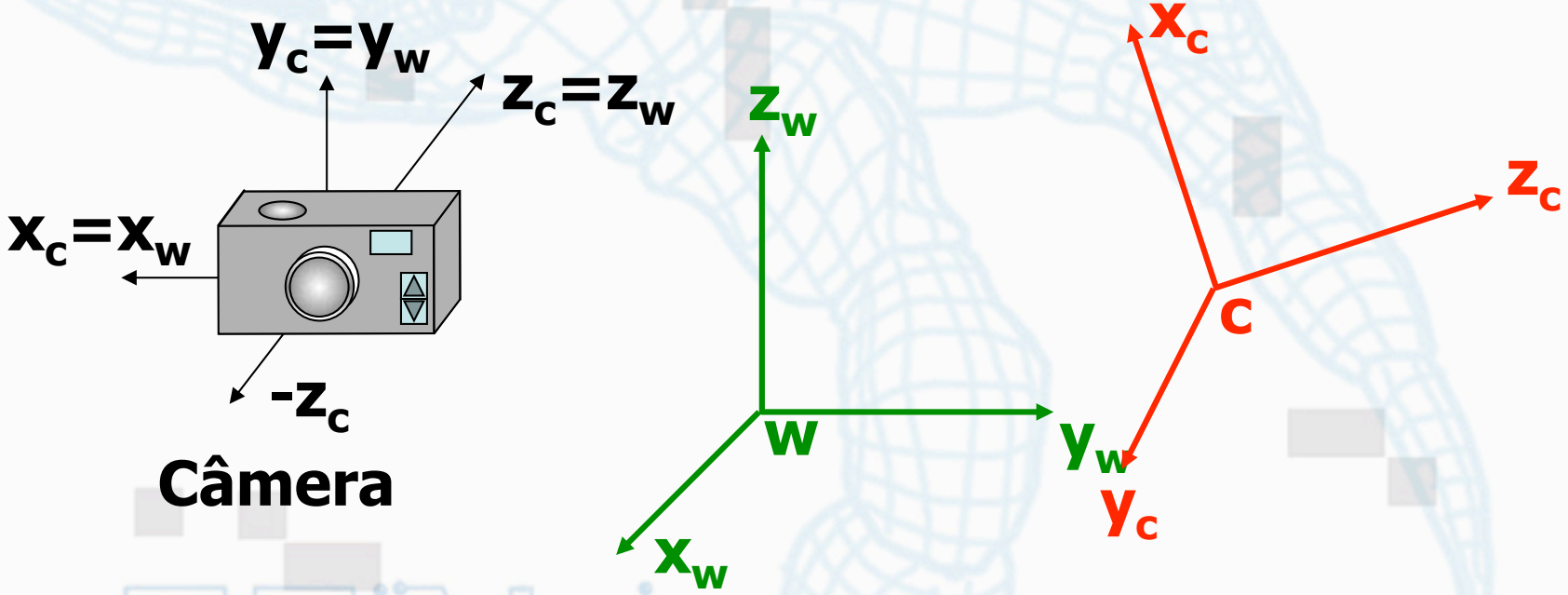
4.3.7 Frames e tipos abstratos de dados

```
point3 p, q;
vector3 v;
frame f;
v = point_sub(p, q, f);
```

4.3 Sistemas de Coordenadas e Frames

4.3.8 Frames no OpenGL

- Frame do Cenário (World Frame)
- Frame da Câmera (Camera Frame)
  - **Default**: coincide com o frame do cenário

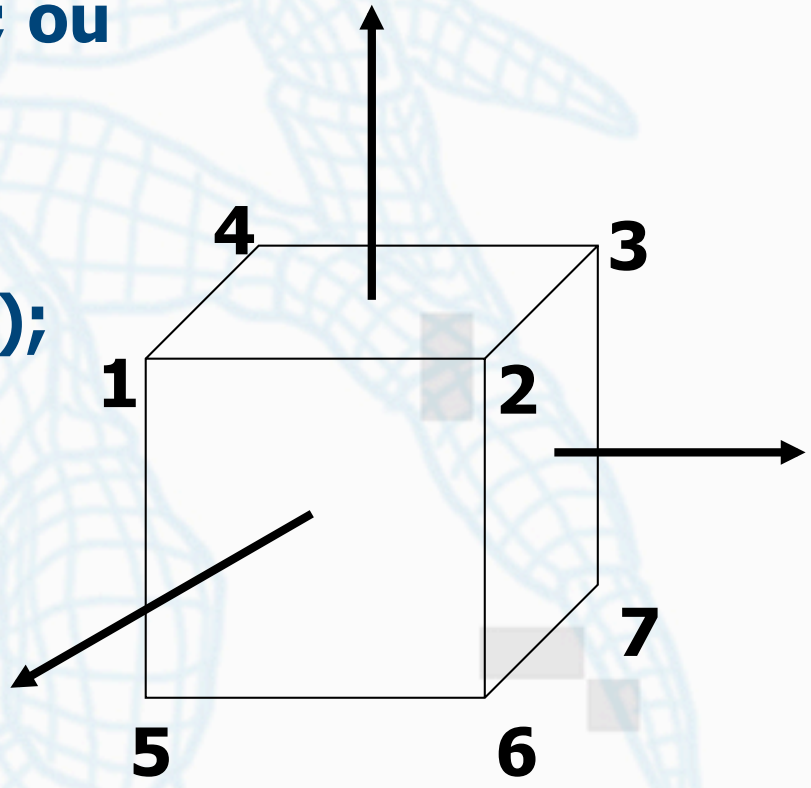


Fim da Aula 11

4.4 Modelando um Cubo Colorido

4.4.1 Modelagem de um cubo

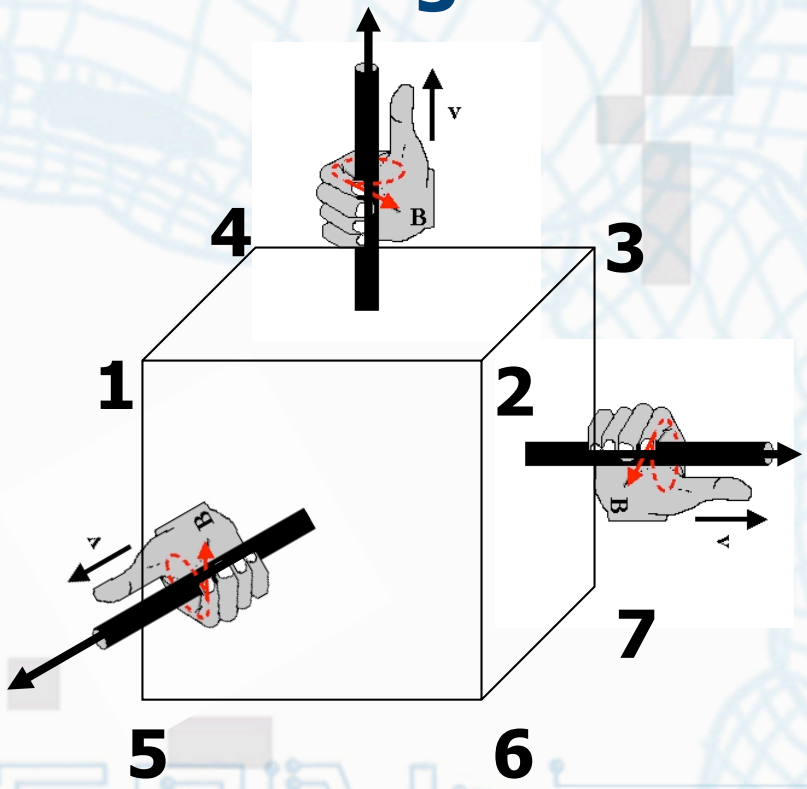
- 8 vértices:
  - GLfloat vertices[8][3]; ou
  - point3 vertices[8];
- 6 faces:
  - glBegin(GL_POLYGON);



4.4 Modelando um Cubo Colorido

4.4.2 Faces voltadas para dentro e para fora

- Obedecer regra da mão-direita

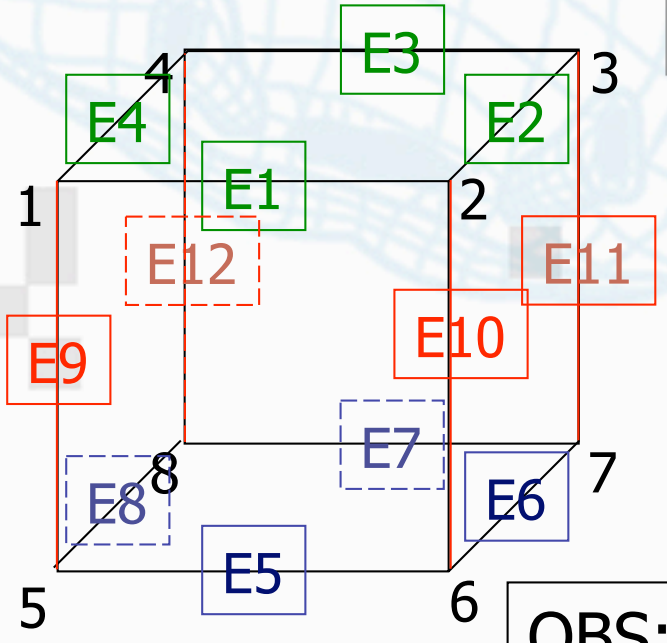


4.4 Modelando um Cubo Colorido

4.4.3 Estruturas de dados para representação de objetos

- Cubo é um poliedro composto de
  - 6 faces, cada uma das quais composta de
    - 4 arestas, cada uma das quais ligadas a
      - 2 vértices

4.4 Modelando um Cubo Colorido

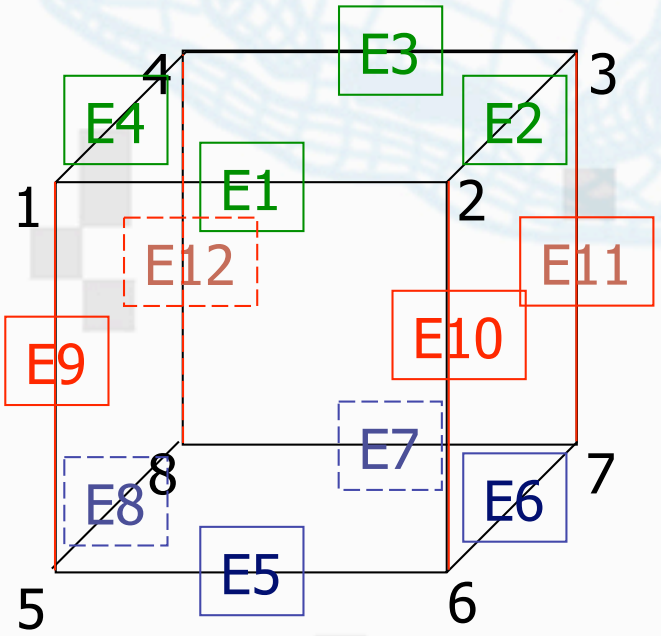


FACES

Cubo					
	F1	E11	E2	E10	E6
	F2	E4	E1	E2	E3
	F3	E10	E1	E9	E5
	F4	E9	E4	E12	E8
	F5	E6	E5	E8	E7
	F6	E12	E3	E11	E7

OBS: Pode-se definir uma face como uma seqüência de vértices. F1: 7 3 2 6

4.4 Modelando um Cubo Colorido



ARESTAS

E1	→	V1	V2
E2	→	V3	V2
E3	→	V4	V3
E4	→	V4	V1
E5	→	V5	V6
E6	→	V7	V6
E7	→	V8	V7
E8	→	V8	V5
E9	→	V5	V1
E10	→	V6	V2
E11	→	V7	V3
E12	→	V8	V4

VÉRTICES

V1	→	-1	1	1
V2	→	1	1	1
V3	→	1	1	-1
V4	→	-1	1	-1
V5	→	-1	-1	1
V6	→	1	-1	1
V7	→	1	-1	-1
V8	→	-1	-1	-1

4.4 Modelando um Cubo Colorido

4.4.4 O cubo de cores

```
typedef GLfloat point3[3];
point3 vertices[8] = { {-1.0,  1.0, 1.0}, { 1.0, 1.0,  1.0},
                       { 1.0,  1.0,-1.0}, {-1.0,1.0,-1.0},
                       {-1.0,-1.0, 1.0},  { 1.0,-1.0,  1.0},
                       { 1.0,-1.0,-1.0},  {-1.0,-1.0,-1.0}};
GLfloat colors[8][3] = {{0.0,  1.0, 1.0}, {1.0, 1.0, 1.0},
                        {1.0,  1.0, 0.0}, {0.0, 1.0, 0.0},
                        {0.0,  0.0, 1.0}, {1.0, 0.0, 1.0},
                        { 1.0, 0.0, 0.0}, {0.0, 0.0, 0.0}};
```

4.4 Modelando um Cubo Colorido

```
void quad(int a, int b, int c, int d)
{
    glBegin(GL_QUADS);
        glColor3fv  (colors[a]);
        glVertex3fv(vertices[a];
        glColor3fv  (colors[b]);
        glVertex3fv(vertices[b];
        glColor3fv  (colors[c]);
        glVertex3fv(vertices[c];
        glColor3fv  (colors[d]);
        glVertex3fv(vertices[d];
    glEnd();
}
```

4.4 Modelando um Cubo Colorido

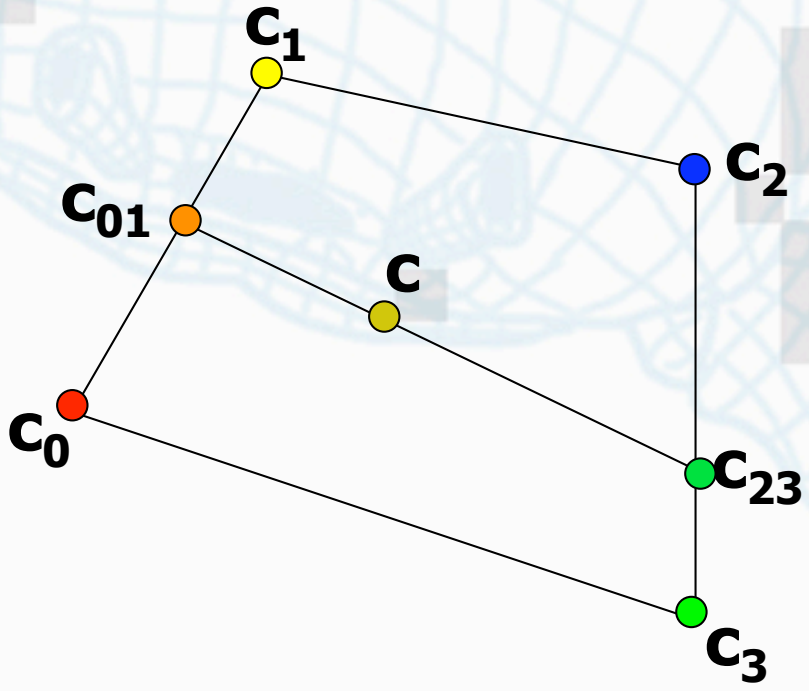
```
void colorcube(void)
{
    quad(6, 2, 1, 5); // Face x+
    quad(3, 0, 1, 2); // Face y+
    quad(5, 1, 0, 4); // Face z+
    quad(4, 0, 3, 7); // Face x-
    quad(6, 5, 4, 7); // Face y-
    quad(7, 3, 2, 6);// Face z-
}
```

Código Fonte

Código Exec

4.4 Modelando um Cubo Colorido

4.4.5 Interpolação bilinear



$$\mathbf{C}_{01}(\alpha) = (1-\alpha)\mathbf{C}_0 + \alpha\mathbf{C}_1$$

$$\mathbf{C}_{23}(\beta) = (1-\beta)\mathbf{C}_3 + \beta\mathbf{C}_2$$

$$\mathbf{C}(\alpha,\beta,\gamma) = (1-\gamma)\mathbf{C}_{01}(\alpha) + \gamma\mathbf{C}_{23}(\beta)$$

4.4 Modelando um Cubo Colorido

4.4.6 Arrays de vértices

- Encapsula informação da estrutura de dados
- Permite traçar poliedros com poucas chamadas de funções
- Tipos de arrays permitidos no OpenGL
  - vertex
  - color
  - color index
  - normal
  - texture coordinate
  - edge flag

4.4 Modelando um Cubo Colorido

- Habilitação dos arrays
  - glEnableClientState(GL_COLOR_ARRAY);
  - glEnableClientState(GL_VERTEX_ARRAY);
- Os arrays do cubo já foram definidos
  - vertices
  - Colors
- Estabelecer ponteiros para os arrays
  - glVertexPointer(3, GL_FLOAT, 0, vertices);
    - 3: número de coordenadas por vértice
    - GL_FLOAT: tipo de cada elemento do array
    - 0: offset em bytes entre vértices consecutivos
    - vertices: ponteiro para o array de vértices

4.4 Modelando um Cubo Colorido

- glColorPointer(3, GL_FLOAT, 0, colors);
  - 3: número de coordenadas por vértice
  - GL_FLOAT: tipo de cada elemento no array
  - 0: offset em bytes entre vértices consecutivos
  - colors: ponteiro para o array de cores

– Fornecer informação da estrutura de dados

- GLubyte cubeIndices[24] = {6, 2, 1, 5,
  3, 0, 1, 2,
  5, 1, 0, 4,
  4, 0, 3, 7,
  6, 5, 4, 7,
  7, 3, 2, 6};

4.4 Modelando um Cubo Colorido

- **Desenhar os arrays**

```
for (i=0; i < 6; i++)
    glDrawElements(GL_POLYGON, 4,
    GL_UNSIGNED_BYTE, &cubeIndices[4*i]);
```

Ou

```
glDrawElements(GL_QUADS, 24,
    GL_UNSIGNED_BYTE, cubeIndices);
```

4.5 Transformações Afins (A17)

- **Transformação de um ponto P em Q**

  Q = T(P)
- **Transformação de um vetor u em v**

  v = R(u)
- **Em coordenadas homogêneas, vetores e pontos são representados como matrizes colunas 4D**

  Q = f(P) e v = f(u) (mesma transformação)

4.5 Transformações Afins (A17)

- **Restringir f a uma função linear**

  $f(\alpha p + \beta q) = \alpha f(p) + \beta f(q)$

  - **Importância**
    - **Conhecendo a transformação de vértices**
    - **Obtém-se a transformação de combinações lineares de vértices através de combinações lineares de transformações de vértices**
  - **f pode ser representada por uma matriz 4x4, [A]**

4.5 Transformações Afins (A17)

- **Matriz de Transformação não-singular**
  - **Mudança de frame**
  - **Mudança do vértice dentro do mesmo frame**
- **Matriz** **Vetor** **Ponto**

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{pmatrix} \quad \mathbf{P} = \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{pmatrix}$$

4.5 Transformações Afins (A17)

- **Transformação de vetores**
  - **Envolve apenas nove coeficientes da matriz**
  - **9 graus de liberdade**
- **Transformação de pontos**
  - **Envolve 12 coeficientes da matriz**
  - **12 graus de liberdade**
- **Transformações afins preservam linhas**
  - **Linha original:** $\mathbf{P}(\alpha) = \mathbf{P_0} + \alpha\,\mathbf{d}$
  - **Transformação:A** $\mathbf{P}(\alpha) = \mathbf{A}(\mathbf{P_0} + \alpha\,\mathbf{d}) = \mathbf{AP_0} + \alpha\,\mathbf{Ad}$
  - **Linha final:** $\mathbf{P'}(\alpha) = \mathbf{P'_0} + \alpha\,\mathbf{d'}$

4.6 Translação, Rotação e Escala (A17)

4.6.1 Translação

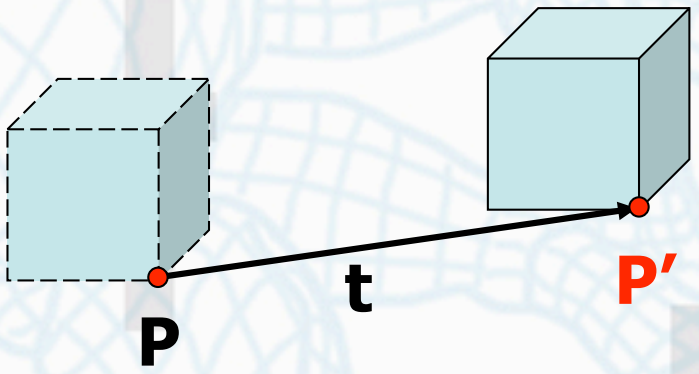
- Operação ponto-vetor

$$P' = P + t$$

$$x' = x + t_x$$

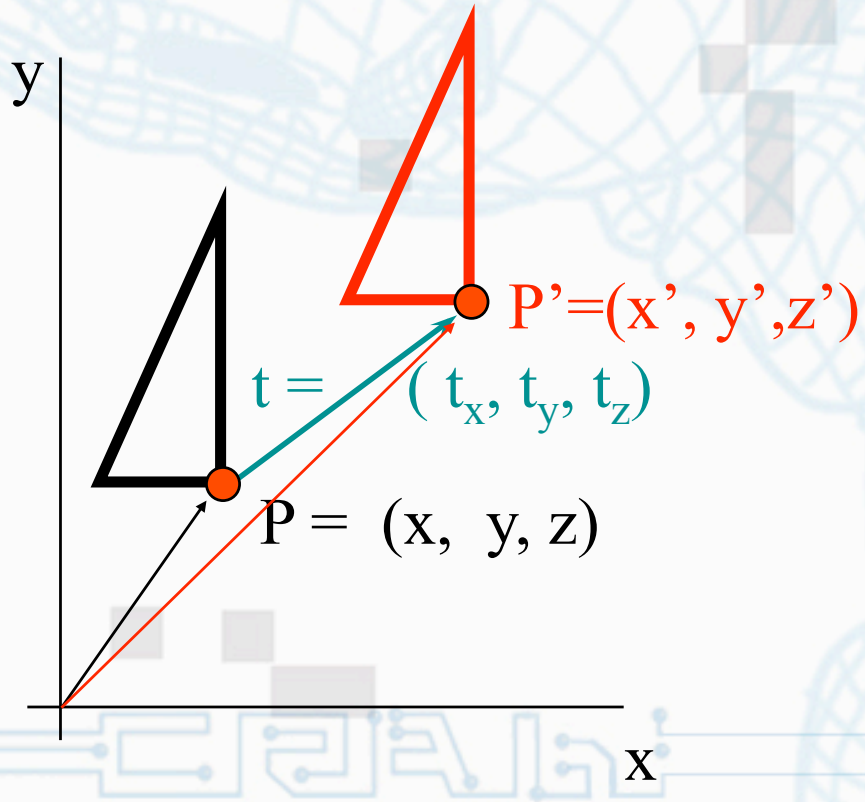
$$y' = y + t_y$$

$$z' = z + t_z$$



4.6 Translação, Rotação e Escala

- **Translação**
  - **Somar o vetor de translação $t = (t_x, t_y, t_z)$ aos vértices para reposicionar objetos**



$$
\begin{aligned}
P' &= P + t \\
x' &= x + t_x \\
y' &= y + t_y \\
z' &= z + t_z
\end{aligned}
$$

4.6 Translação, Rotação e Escala

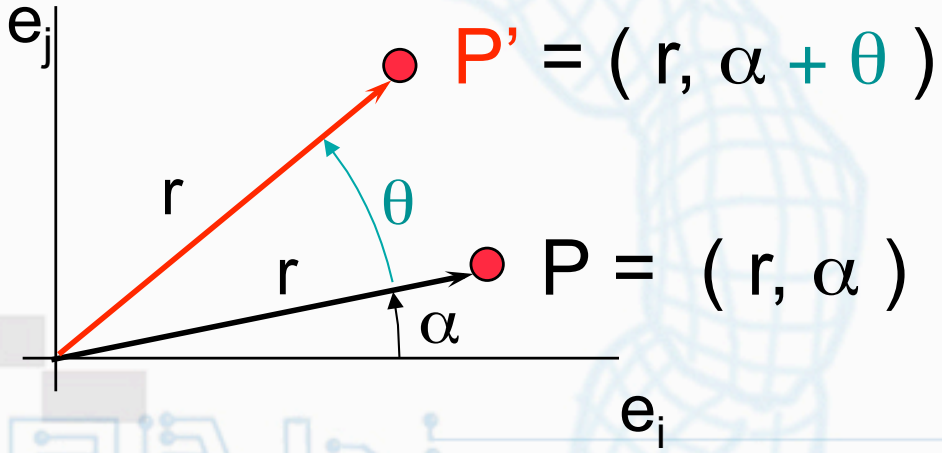
4.6.2 Rotação

vértice P em coordenadas polares

(r, α) irá para posição P' = (r, $\alpha+\theta$)

$\theta > 0$: rotação em sentido anti-horário

$\theta < 0$: rotação em sentido horário

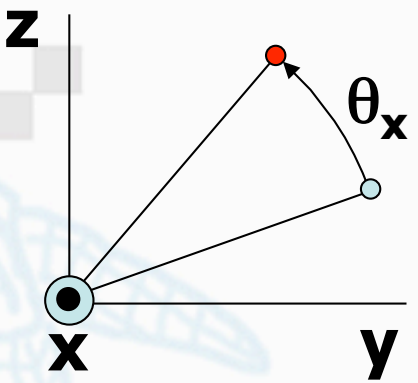


4.6 Translação, Rotação e Escala

- **Rotação em torno do eixo x**
  - **Coordenadas na forma polar:**

    $y' = r \cos(\alpha + \theta)$ $\quad y = r \cos \alpha$

    $z' = r \operatorname{sen}(\alpha + \theta)$ $\quad z = r \operatorname{sen} \alpha$



  - **Mas:**

    $\cos(\alpha + \theta) = \cos \alpha \cdot \cos \theta - \operatorname{sen} \alpha \cdot \operatorname{sen} \theta$

    $\operatorname{sen}(\alpha + \theta) = \operatorname{sen} \alpha \cdot \cos \theta + \operatorname{sen} \theta \cdot \cos \alpha$

  - **Assim:**

    $y' = r \cos \alpha \cdot \cos \theta - r \operatorname{sen} \alpha \cdot \operatorname{sen} \theta$

    $z' = r \operatorname{sen} \alpha \cdot \cos \theta + r \cos \alpha \cdot \operatorname{sen} \theta$

    $y' = y \cdot \cos \theta - z \cdot \operatorname{sen} \theta$

    $z' = z \cdot \cos \theta + y \cdot \operatorname{sen} \theta$

4.6 Translação, Rotação e Escala

Rotação em torno do eixo x

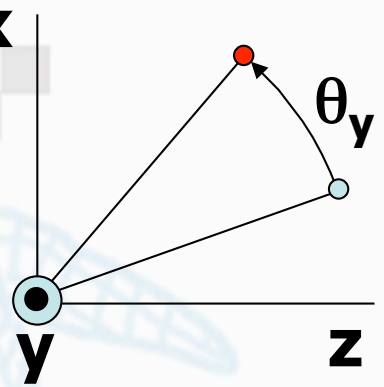
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -sen\theta_x \\ 0 & sen\theta_x & \cos\theta_x \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

4.6 Translação, Rotação e Escala

- **Rotação em torno do eixo y**
  - **Coordenadas na forma polar:**

    $z' = r \cos(\alpha + \theta)$ $\quad z = r \cos \alpha$

    $x' = r \operatorname{sen}(\alpha + \theta)$ $\quad x = r \operatorname{sen} \alpha$



  - **Mas:**

    $\cos(\alpha + \theta) = \cos \alpha \cdot \cos \theta - \operatorname{sen} \alpha \cdot \operatorname{sen} \theta$

    $\operatorname{sen}(\alpha + \theta) = \operatorname{sen} \alpha \cdot \cos \theta + \operatorname{sen} \theta \cdot \cos \alpha$

  - **Assim:**

    $z' = r \cos \alpha \cdot \cos \theta - r \operatorname{sen} \alpha \cdot \operatorname{sen} \theta$

    $x' = r \operatorname{sen} \alpha \cdot \cos \theta + r \cos \alpha \cdot \operatorname{sen} \theta$

    $z' = z \cdot \cos \theta - x \cdot \operatorname{sen} \theta$

    $x' = x \cdot \cos \theta + z \cdot \operatorname{sen} \theta$

4.6 Translação, Rotação e Escala

Rotação em torno do eixo y

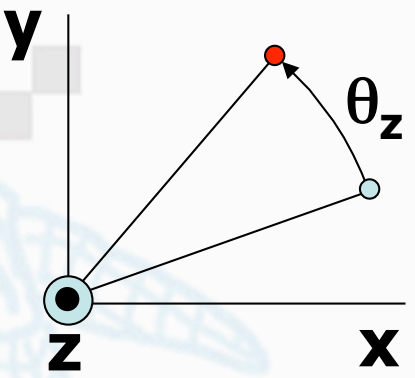
$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} \cos\theta_y & 0 & sen\theta_y \\ 0 & 1 & 0 \\ -sen\theta_y & 0 & \cos\theta_y \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

4.6 Translação, Rotação e Escala

- **Rotação em torno do eixo z**
  - **Coordenadas na forma polar:**

    $x' = r \cos(\alpha + \theta)$ $\quad x = r \cos \alpha$

    $y' = r \operatorname{sen}(\alpha + \theta)$ $\quad y = r \operatorname{sen} \alpha$



  - **Mas:**

    $\cos(\alpha + \theta) = \cos \alpha \cdot \cos \theta - \operatorname{sen} \alpha \cdot \operatorname{sen} \theta$

    $\operatorname{sen}(\alpha + \theta) = \operatorname{sen} \alpha \cdot \cos \theta + \operatorname{sen} \theta \cdot \cos \alpha$

  - **Assim:**

    $x' = r \cos \alpha \cdot \cos \theta - r \operatorname{sen} \alpha \cdot \operatorname{sen} \theta$

    $y' = r \operatorname{sen} \alpha \cdot \cos \theta + r \cos \alpha \cdot \operatorname{sen} \theta$

    $x' = x \cdot \cos \theta - y \cdot \operatorname{sen} \theta$

    $y' = y \cdot \cos \theta + x \cdot \operatorname{sen} \theta$

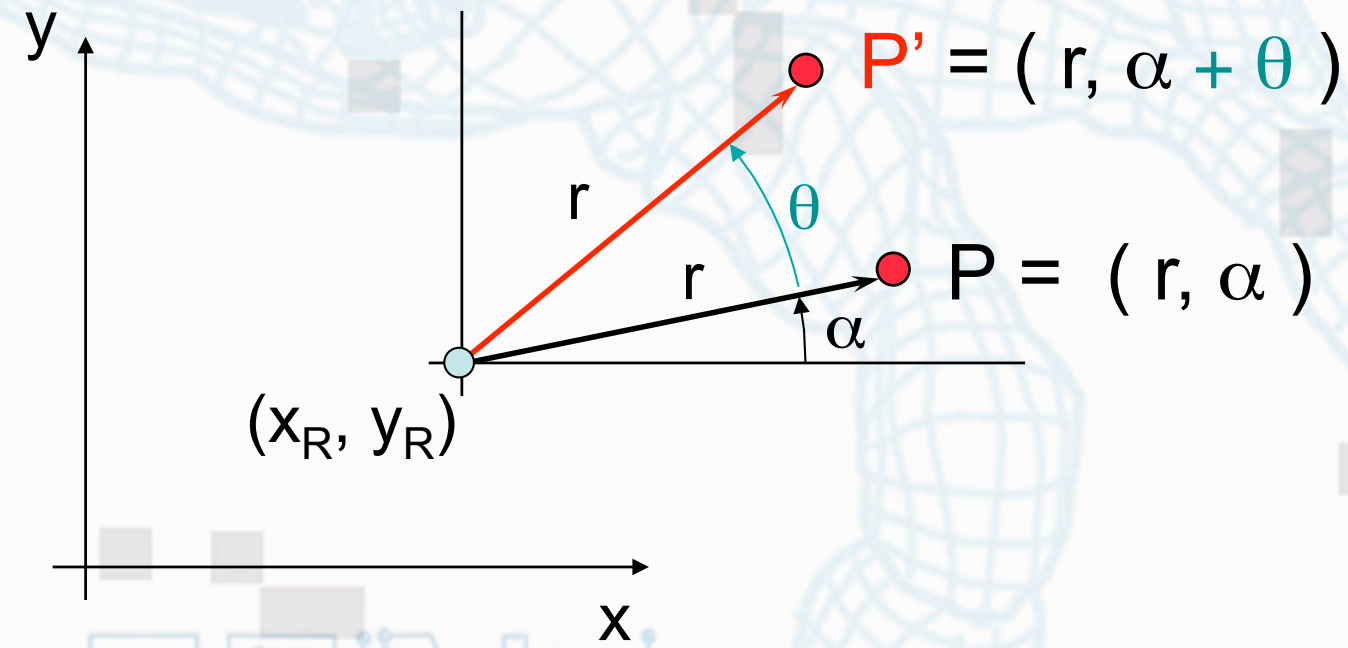
4.6 Translação, Rotação e Escala

Rotação em torno do eixo z

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} \cos\theta_z & -sen\theta_z & 0 \\ sen\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

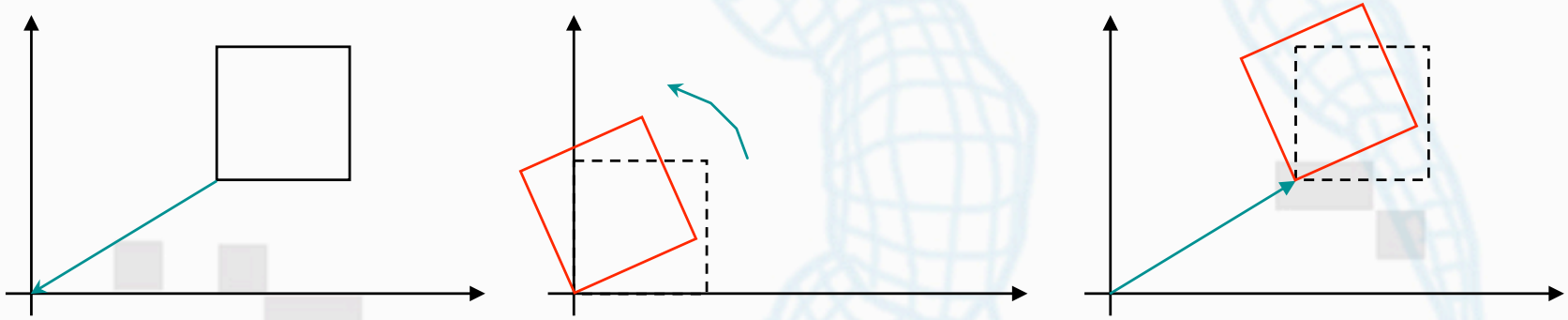
4.6 Translação, Rotação e Escala

- **Rotação em torno de um eixo // z passando pelo pivô, (x_R, y_R)**
  vértice (x, y) irá para posição (x', y')



4.6 Translação, Rotação e Escala

- **Rotação em torno do eixo //z pelo pivô, (x_R, y_R, z_R)**
  Equivale a
  - **Translação para a origem (0, 0, 0)**
  - **Rotação em relação ao eixo z**
  - **Translação de volta para (x_R, y_R, z_R)**



4.6 Translação, Rotação e Escala

- **Rotação em torno de eixo // z pelo pivô (x_R, y_R)**

$$x' = (x - x_R) \cdot \cos\theta - (y - y_R) \cdot \text{sen}\,\theta + x_R$$

$$y' = (y - y_R) \cdot \cos\theta + (x - x_R) \cdot \text{sen}\,\theta + y_R$$

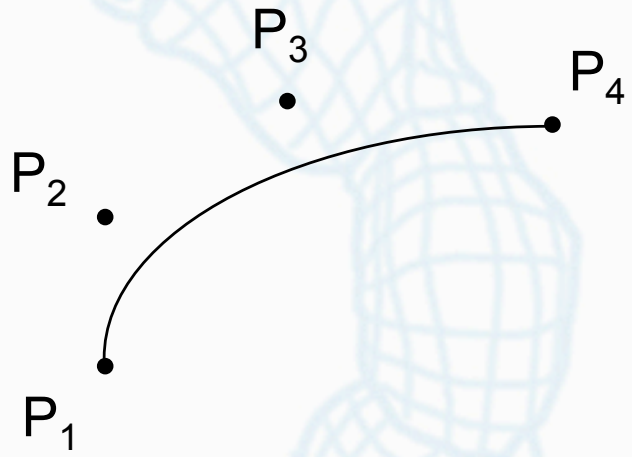
$$\begin{Bmatrix} x' \\ y' \end{Bmatrix} = \begin{bmatrix} \cos\theta & -sen\theta \\ sen\theta & \cos\theta \end{bmatrix} \begin{Bmatrix} x - x_R \\ y - y_R \end{Bmatrix} + \begin{Bmatrix} x_R \\ y_R \end{Bmatrix}$$

4.6 Translação, Rotação e Escala

- **Rotação para objetos definidos com curvas**
  - **transformações geométricas aplicadas a coordenadas de definição**
  - **implementações adaptadas à geometria do objeto**
- **Exemplos**
  - **Círculos**
    - **rotação ou translação: usar coordenadas do centro, e**
    - **escala: amplificar os raios**

4.6 Translação, Rotação e Escala

- **Rotação para objetos definidos com curvas**
  - **Segmentos de curvas**
  - **transformação: usar coordenadas de definição da curva (Pontos de controle)**



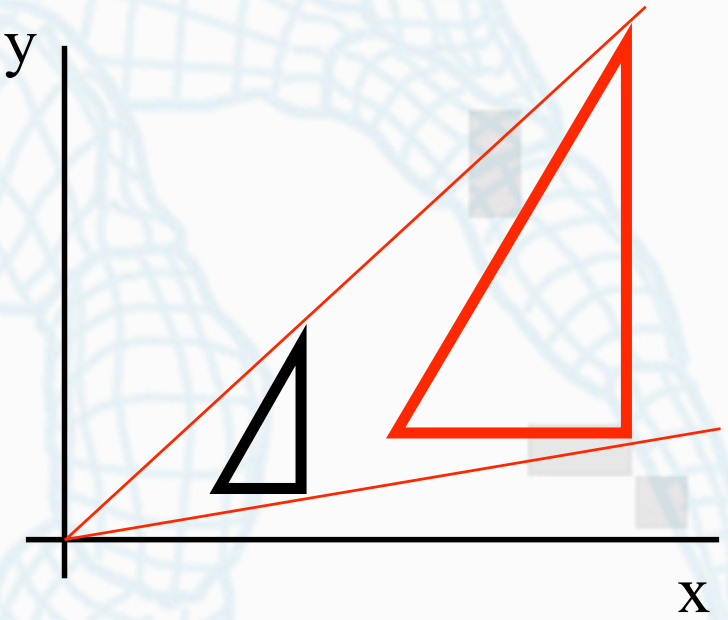
4.6 Translação, Rotação e Escala

4.6.3 Escala

- É uma transformação afim que altera a forma do objeto (non-rigid-body)
- Multiplicar os valores das coordenadas por fatores de escala s_x, s_y, s_z
- Transformação básica de escala
- $(x', y', z') = (x.s_x, y.s_y, z.s_z)$, $s_x, s_y, s_z > 0$
- Mudança uniforme de Escala quando $s_x = s_y = s_z$

4.6 Translação, Rotação e Escala

- **Efeito da transformação de escala**
  - **mudança do tamanho do objeto**
  - **deslocamento do objeto**
  - **distorção se s_x != s_y**

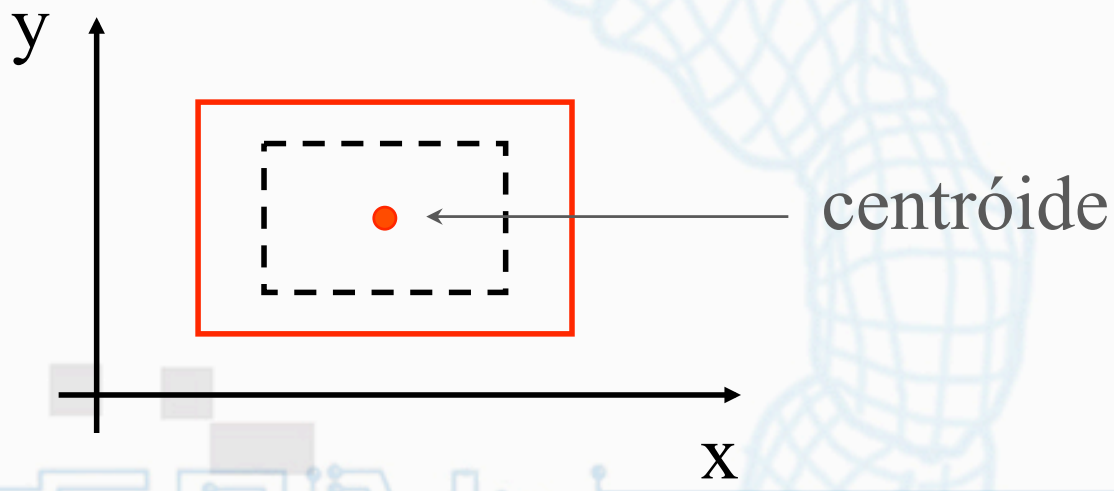


4.6 Translação, Rotação e Escala

- **Escala relativa a um ponto fixo, P_f**

  $P_f = (x_f, y_f)$

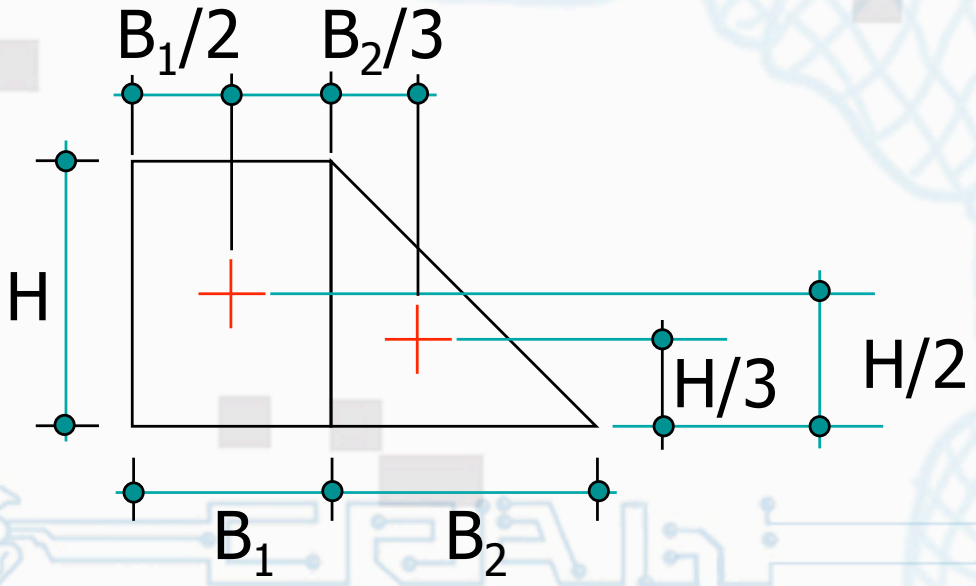
  - **se (x_f, y_f) for centróide → objeto não move**
  - **senão → deslocamento em relação a P_f**



4.6 Translação, Rotação e Escala

- **Coordenadas do centróide**

$$x_c = \frac{\int_V x\,dV}{\int_V dV}, \quad y_c = \frac{\int_V y\,dA}{\int_V dV} \text{ e } z_c = \frac{\int_V z\,dV}{\int_V dV}$$



$$x_c = \frac{\frac{B_1}{2}(B_1 H) + \left(B_1 + \frac{B_2}{3}\right)\left(\frac{1}{2}B_2 H\right)}{(B_1 H) + \left(\frac{1}{2}B_2 H\right)}$$

$$y_c = \frac{\frac{H}{2}(B_1 H) + \frac{H}{3}\left(\frac{1}{2}B_2 H\right)}{(B_1 H) + \left(\frac{1}{2}B_2 H\right)}$$

4.6 Translação, Rotação e Escala

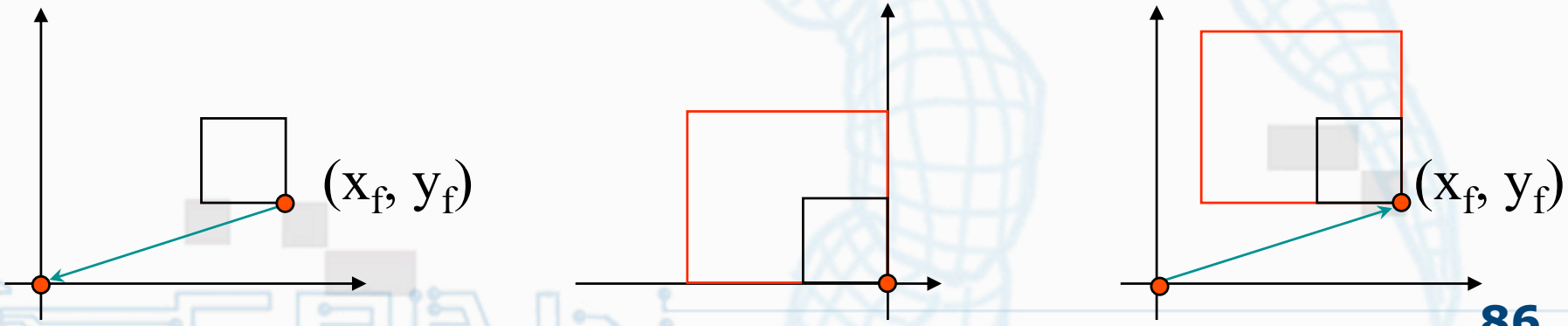
- **Escala relativa a um ponto fixo, P_f**

$$x' = (x - x_f) . s_x + x_f$$

$$y' = (y - y_f) . s_y + y_f$$

$$z' = (z - z_f) . s_z + z_f$$

  - Translação para a origem (0, 0, 0)
  - Escala em relação à origem
  - Translação de volta para (x_f, y_f)



4.7 Transformações em Coordenadas Homogêneas

- **Usando-se coordenadas homogêneas, as transformações geométricas são escritas como**

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \mathbf{M} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \Rightarrow \begin{cases} x' = m_{11}x + m_{12}y + m_{13}z + m_{14} \\ y' = m_{21}x + m_{22}y + m_{23}z + m_{24} \\ z' = m_{31}x + m_{32}y + m_{33}z + m_{34} \end{cases}$$

4.7 Transformações em Coordenadas Homogêneas

4.7.1 Translação

- **Matriz de Translação**

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- **Forma inversa (T^{-1}) com $-t_x$, $-t_y$, $-t_z$.**

4.7 Transformações em Coordenadas Homogêneas

4.7.2 Escala

- Matriz de Escala em relação a um ponto fixo, F

$$\mathbf{S} = \begin{bmatrix} S_x & 0 & 0 & (1-S_x)x_F \\ 0 & S_y & 0 & (1-S_y)y_F \\ 0 & 0 & S_z & (1-S_z)z_F \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- A escala é uniforme quando $S_x = S_y = S_z$
- Forma inversa S^{-1} com $S'_x \rightarrow \frac{1}{S_x}, S'_y \rightarrow \frac{1}{S_y}, S'_z \rightarrow \frac{1}{S_z}$

4.7 Transformações em Coordenadas Homogêneas

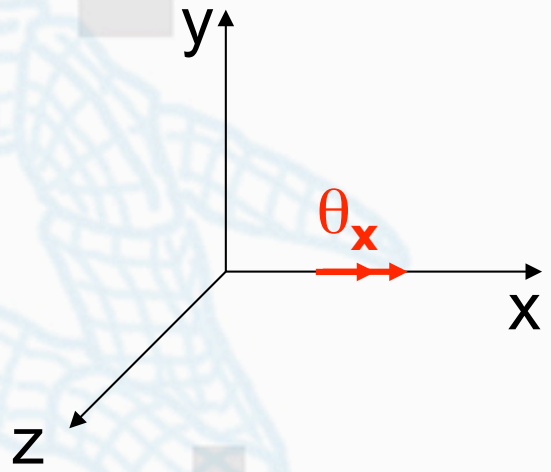
4.7.3 Rotação

- Em 3D, as rotações podem ser sobre qualquer reta
- As rotações mais simples são sobre os eixos de coordenadas
- Outras rotações são equivalentes a combinações das rotações dos 3 eixos-coordenados (mais T e T').
- Para todas as rotações 3D sobre eixos de coordenadas
  - $\theta > 0$: rotação anti-horária (a partir da origem) na regra da mão-direita

4.7 Transformações em Coordenadas Homogêneas

- **Rotação em relação ao eixo-x**

$$\mathbf{R}_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta_x & -sen\theta_x & 0 \\ 0 & sen\theta_x & \cos\theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



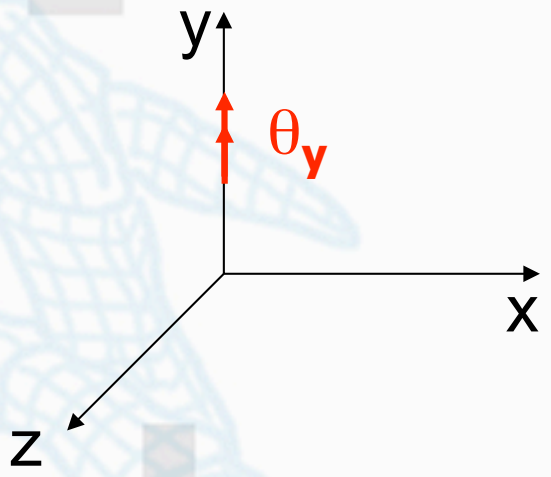
- **Rotações sobre os outros dois eixos são obtidas com permutação cíclica**

  x → y → z → x

4.7 Transformações em Coordenadas Homogêneas

– **Rotação em relação ao eixo-y**

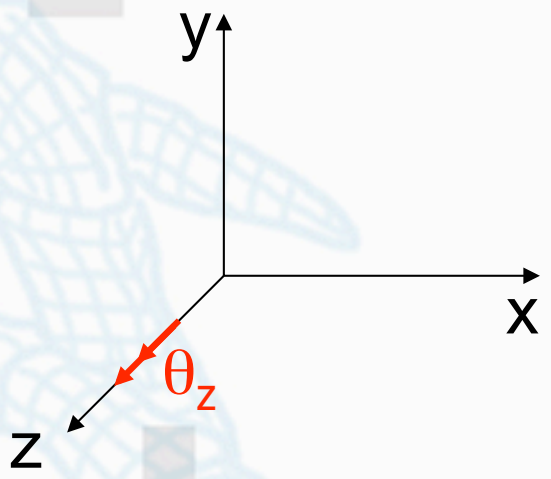
$$\mathbf{R}_y = \begin{bmatrix} \cos\theta_y & 0 & sen\theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -sen\theta_y & 0 & \cos\theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



4.7 Transformações em Coordenadas Homogêneas

– **Rotação em relação ao eixo-z**

$$\mathbf{R}_z = \begin{bmatrix} \cos\theta_z & -sen\theta_z & 0 & 0 \\ sen\theta_z & \cos\theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



4.7 Transformações em Coordenadas Homogêneas

- Obtém-se a inversa de R_x, R_y ou R_z com

  $\theta \rightarrow -\theta$

  como
  - $\cos(-\theta) = \cos(\theta)$
  - $\text{sen}(-\theta) = -\text{sen}(\theta)$,

  $$R^{-1} = R^T$$

- R são matrizes ortonormais

4.7 Transformações em Coordenadas Homogêneas

- **Outras rotações são obtidas por composição de matrizes de transformação**
  - **Transformação do eixo de rotação em um eixo de coordenada**
  - **Rotação especificada em torno do eixo transformado**
  - **Transformação do eixo de rotação de volta para a posição e orientação originais**

4.7 Transformações em Coordenadas Homogêneas

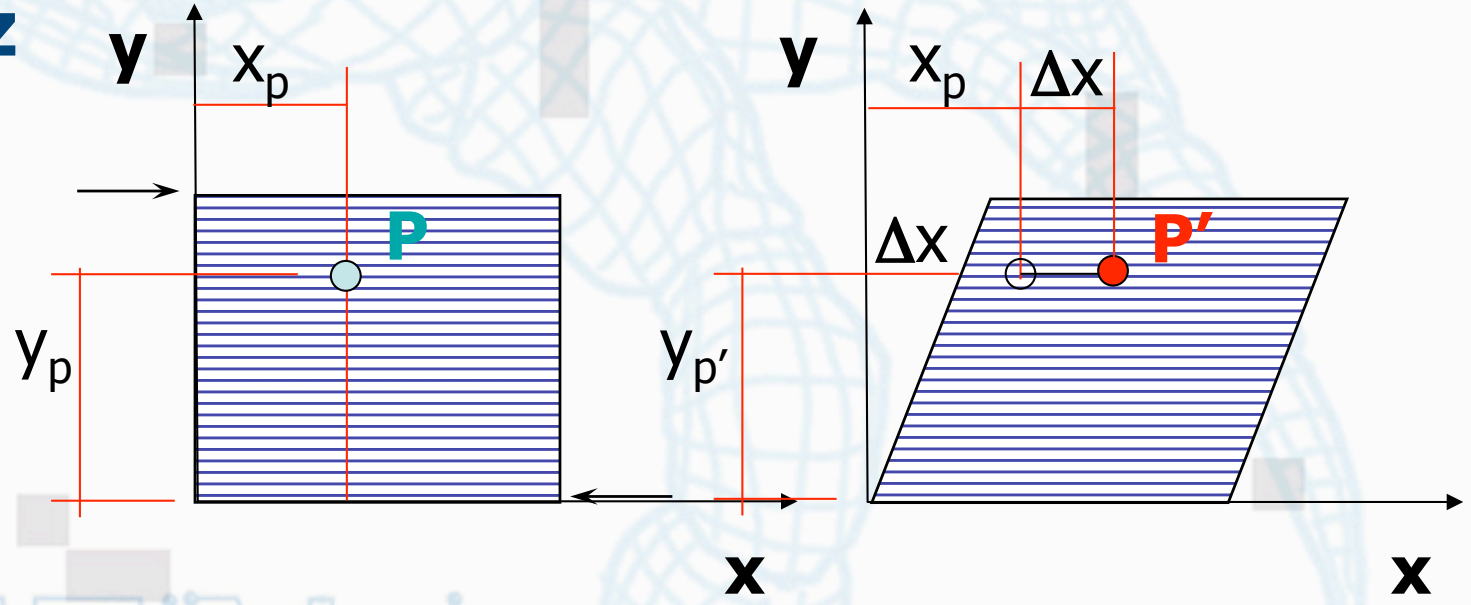
4.7.4 Cisalhamento (Shear)

- Cisalhamento no plano xy na direção x

$x' = x+\Delta x = x + tg(\gamma)y = x + sh_x y$

$y' = y$

$z' = z$



4.7 Transformações em Coordenadas Homogêneas

- **Cisalhamento no plano xy na direção x**

$$S^{x}_{xy} = \begin{bmatrix} 1 & \mathrm{sh}_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

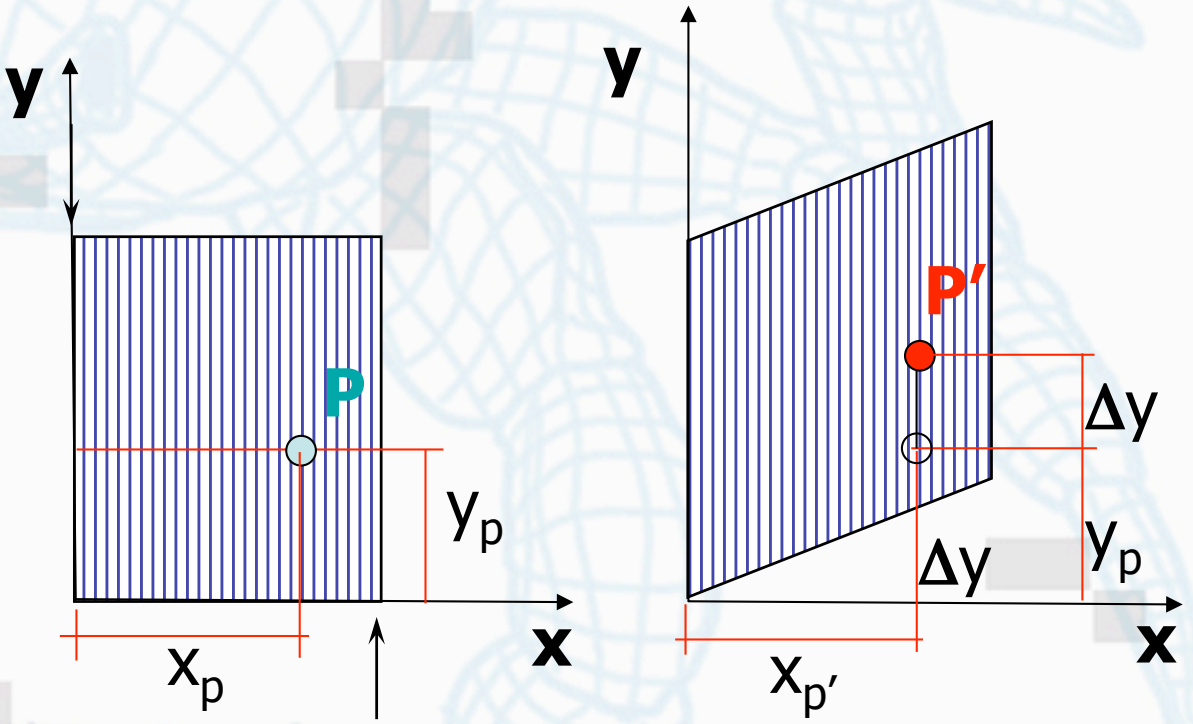
4.7 Transformações em Coordenadas Homogêneas

- **Cisalhamento no plano xy na direção y**

$x' = x$

$y' = y+\Delta y = y + tg(\gamma)x = y + sh_y x$

$z' = z$



4.7 Transformações em Coordenadas Homogêneas

- **Cisalhamento no plano xy na direção y**

$$S_{xy}^{y} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \mathrm{sh}_y & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

– **Cisalhamento no plano xz na direção x**

$$S^{x}_{xz} = \begin{bmatrix} 1 & 0 & \mathrm{sh}_x & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

– **Cisalhamento no plano xz na direção z**

$$S_{xz}^{z} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \mathrm{sh}_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

– **Cisalhamento no plano yz na direção y**

$$S^{y}_{yz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & \mathrm{sh}_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

– **Cisalhamento no plano yz na direção z**

$$S^{z}_{yz} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & sh_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fim da Aula 12

4.7 Transformações em Coordenadas Homogêneas

4.7.5 Outras Transformações

- **Reflexão**: imagem de objeto em relação a um plano
- **Espelho xz:** $y \rightarrow -y$
  - Equivale a $s_x = s_z = 1, s_y = -1$

$$S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- **Espelho yz:** $x \rightarrow -x$
  - Equivale a $s_x = -1, s_y = s_y = 1$

$$S = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

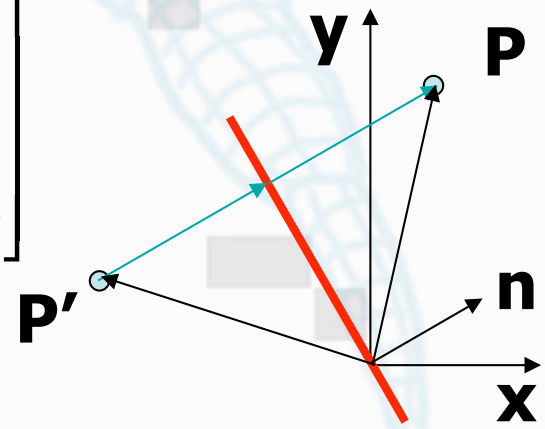
– **Espelho xy: z → - z**
  - **Equivale a $s_x = s_y = 1, s_z = -1$**

$$\mathbf{S} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

– **Espelho cuja normal é n = (n_x, n_y, n_z):**

$$\mathbf{P}'_i = \mathbf{P}_i - 2(\mathbf{P}_k\mathbf{n}_k)\mathbf{n}_i = (\delta_{ik} - 2\mathbf{n}_i\mathbf{n}_k)\mathbf{P}_k$$

$$\mathbf{S_n} = \begin{bmatrix} (1-2\mathbf{n}_1\mathbf{n}_1) & -2\mathbf{n}_1\mathbf{n}_2 & -2\mathbf{n}_1\mathbf{n}_3 \\ -2\mathbf{n}_2\mathbf{n}_1 & (1-2\mathbf{n}_2\mathbf{n}_2) & -2\mathbf{n}_2\mathbf{n}_3 \\ -2\mathbf{n}_3\mathbf{n}_1 & -2\mathbf{n}_3\mathbf{n}_2 & (1-2\mathbf{n}_3\mathbf{n}_3) \end{bmatrix}$$



4.7 Transformações em Coordenadas Homogêneas

- **Reflexão relativa à origem:**

  $x \rightarrow -x$ e $y \rightarrow -y$

  - **Equivale a $s_x = -1$, $s_y = -1$**

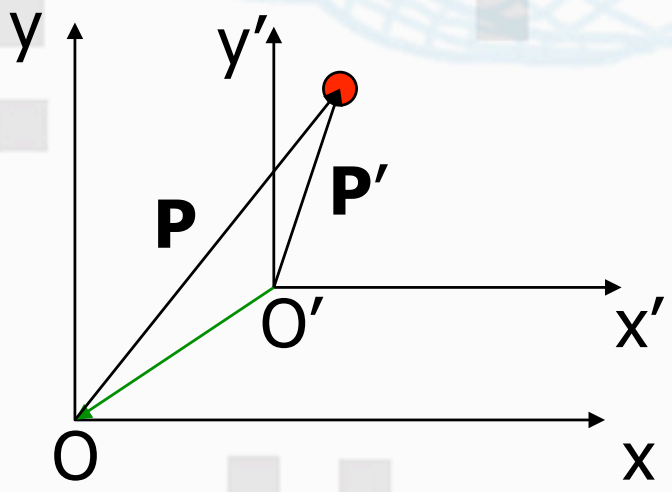
4.7 Transformações em Coordenadas Homogêneas

Transformações entre sistemas de coordenadas

- Usado em
  - Transformações de Visualização
  - Transformações de Modelagem
- Transformação entre sitemas paralelos com origens distintas
- Transformações entre sistemas centrados na mesma origem com orientações distintas

4.7 Transformações em Coordenadas Homogêneas

- **Transformação entre sistemas paralelos com origens distintas**
  - **Aplicar translação $T_{o'o}$ aos pontos da cena**

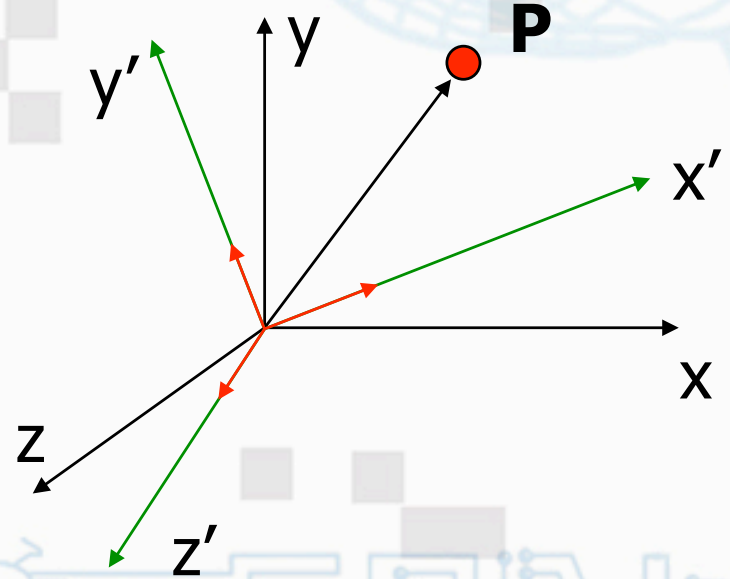


$$\mathbf{P}' = \overrightarrow{O'O} + \mathbf{P}$$

$$\mathbf{P}' = \begin{bmatrix} 1 & 0 & 0 & -O'_x \\ 0 & 1 & 0 & -O'_y \\ 0 & 0 & 1 & -O'_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \\ 1 \end{pmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

- **Transformações entre sistemas centrados na mesma origem com orientações distintas**
  - **Construir matriz de transformação cujas linhas são vetores unitários nas direções de x' y' e z' representados por suas componentes no sistema x y z**



$$\mathbf{i}' = \begin{pmatrix} i'_x \\ i'_y \\ i'_z \end{pmatrix} \qquad \mathbf{j}' = \begin{pmatrix} j'_x \\ j'_y \\ j'_z \end{pmatrix} \qquad \mathbf{k}' = \begin{pmatrix} k'_x \\ k'_y \\ k'_z \end{pmatrix}$$

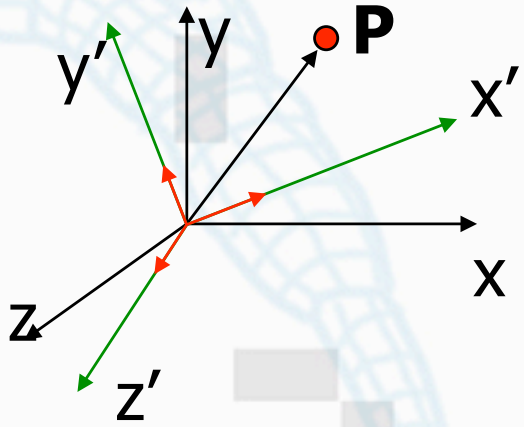
$$\mathbf{P}' = \begin{bmatrix} i'_x & i'_y & i'_z & 0 \\ j'_x & j'_y & j'_z & 0 \\ k'_x & k'_y & k'_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} \mathrm{P}_x \\ \mathrm{P}_y \\ \mathrm{P}_z \\ 1 \end{pmatrix}$$

4.7 Transformações em Coordenadas Homogêneas

- **Assim**

$$\begin{pmatrix}1\\0\\0\\1\end{pmatrix}=\begin{bmatrix}i_x & i_y & i_z & 0\\ j_x & j_y & j_z & 0\\ k_x & k_y & k_z & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{pmatrix}i_x\\i_y\\i_z\\1\end{pmatrix} \qquad \begin{pmatrix}0\\1\\0\\1\end{pmatrix}=\begin{bmatrix}i_x & i_y & i_z & 0\\ j_x & j_y & j_z & 0\\ k_x & k_y & k_z & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{pmatrix}j_x\\j_y\\j_z\\1\end{pmatrix}$$

$$\begin{pmatrix}0\\0\\1\\1\end{pmatrix}=\begin{bmatrix}i_x & i_y & i_z & 0\\ j_x & j_y & j_z & 0\\ k_x & k_y & k_z & 0\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{pmatrix}k_x\\k_y\\k_z\\1\end{pmatrix}$$



$\mathbf{T}$ é Ortonormal $\rightarrow$ $\mathbf{T}^{-1}=\mathbf{T}$ $\rightarrow$ $\mathbf{P}'=\mathbf{T}\cdot\mathbf{P}$ e $\mathbf{P}=\mathbf{T}^{T}\cdot\mathbf{P}'$

4.8 Concatenação de Transformações

4.8.1 Rotação em torno de um ponto fixo

- Utilizar concatenação de 3 operações
  - Translação do ponto fixo para a origem
    - $T(-P_F)$
  - Rotação em torno do eixo passando pela origem
    - $R(\theta)$
  - Translação de volta para a posição do ponto fixo
    - $T(P_F)$
  - $P' = [T(P_F)\ R(\theta)\ T(-P_F)]\ P$

4.8 Concatenação de Transformações

4.8.2 Rotação geral

- É possível expressar como concatenação de rotações em torno de z, y e x
  - $R = R_x R_y R_z$

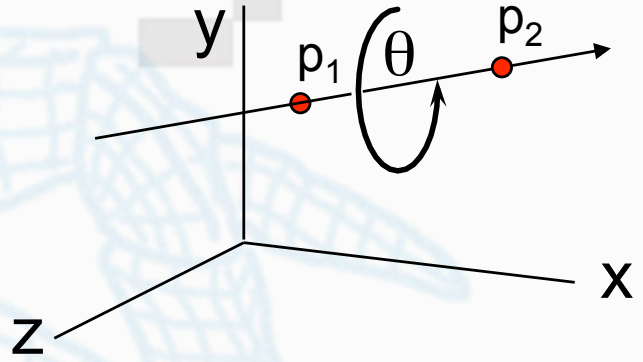
4.8 Concatenação de Transformações

4.8.3 Transformação de instância

- Usar um objeto protótipo
- Gerar instância desse objeto
- Aplicar transformações para a forma desejada

4.8 Concatenação de Transformações (A18)

4.8.4 Rotação sobre um eixo arbitrário



- Definir o eixo de rotação
  - dois pontos sobre a reta ou
  - um ponto e um vetor-direção
- O segmento P_1P_2 ("segmento de reta orientado") define um vetor

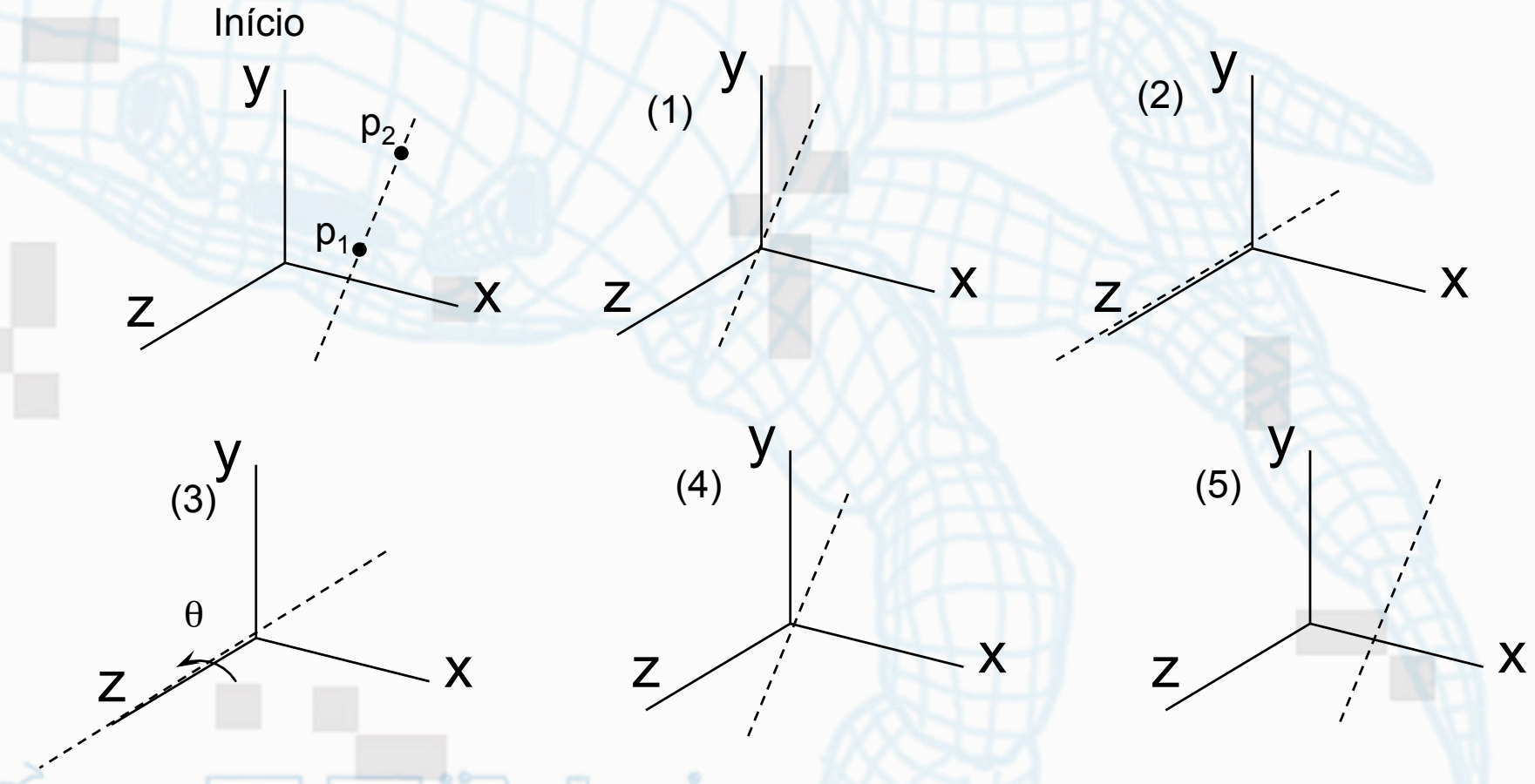
  $V = (x_2-x_1, y_2-y_1, z_2-z_1)$
- Determinar uma forma de rotação geral de matriz usando operações com vetores

4.8 Concatenação de Transformações (A18)

- **Rotação sobre um eixo arbitrário**
  1. Translação do eixo de rotação para a origem
  2. Rotações para alinhar o eixo de rotação com o eixo z (ou outro eixo)
  3. Rotação sobre o eixo z
  4. Inversão das rotações no passo 2
  5. Inversão da translação do passo 1

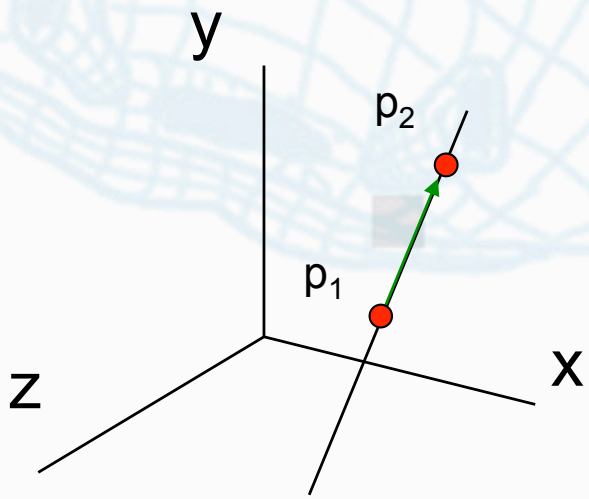
4.8 Concatenação de Transformações (A18)

– Rotação sobre um eixo arbitrário



4.8 Concatenação de Transformações (A18)

- **Rotação sobre um eixo arbitrário**
  - **Coordenadas de p_1 e p_2 determinam o vetor unitário û ao longo do eixo de rotação**



$$\mathbf{V} = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$$

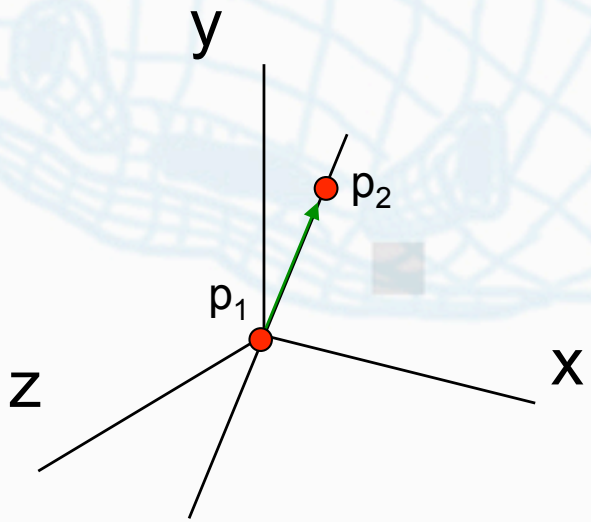
$$\mathbf{u} = \frac{\mathbf{V}}{|\mathbf{V}|} = (a, b, c)$$

$$a = \frac{x_2 - x_1}{|\mathbf{V}|}, b = \frac{y_2 - y_1}{|\mathbf{V}|}, c = \frac{z_2 - z_1}{|\mathbf{V}|}$$

4.8 Concatenação de Transformações (A18)

- **Rotação sobre um eixo arbitrário**

  <u>Passo 1</u>: Translação de p_1 para a origem

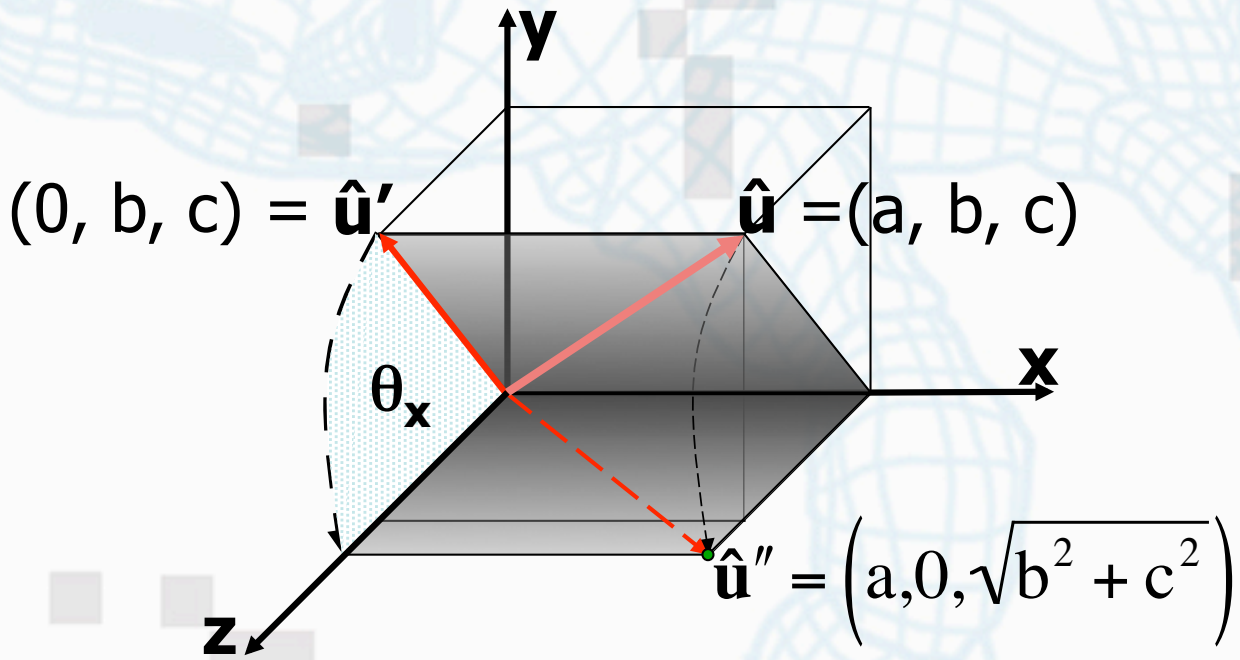


$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & -z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.8 Concatenação de Transformações (A18)

<u>Passo 2.1</u>: Posicionamento do eixo de rotação no plano xz

- **determinar ângulo θ_x entre û' e k**
- **girar de um ângulo θ_x sobre o eixo x**



4.8 Concatenação de Transformações (A18)

- **determinar ângulo θ_x entre û' e k**
  - **Encontre o cos θ_x**

$$\hat{\mathbf{u}}' \cdot \mathbf{k} = |\hat{\mathbf{u}}'||\mathbf{k}|\cos\theta_x \Rightarrow \cos\theta_x = \frac{\hat{\mathbf{u}}' \cdot \mathbf{k}}{|\hat{\mathbf{u}}'|1} = \frac{c}{d}$$

$$\text{onde } d = |\hat{\mathbf{u}}'| = \sqrt{b^2 + c^2}$$

  - **Encontre o sen θ_x**

$$sen\theta_x = \sqrt{1 - \left(\frac{c}{d}\right)^2} = \sqrt{\frac{d^2 - c^2}{d^2}} = \frac{b}{d}$$

4.8 Concatenação de Transformações (A18)

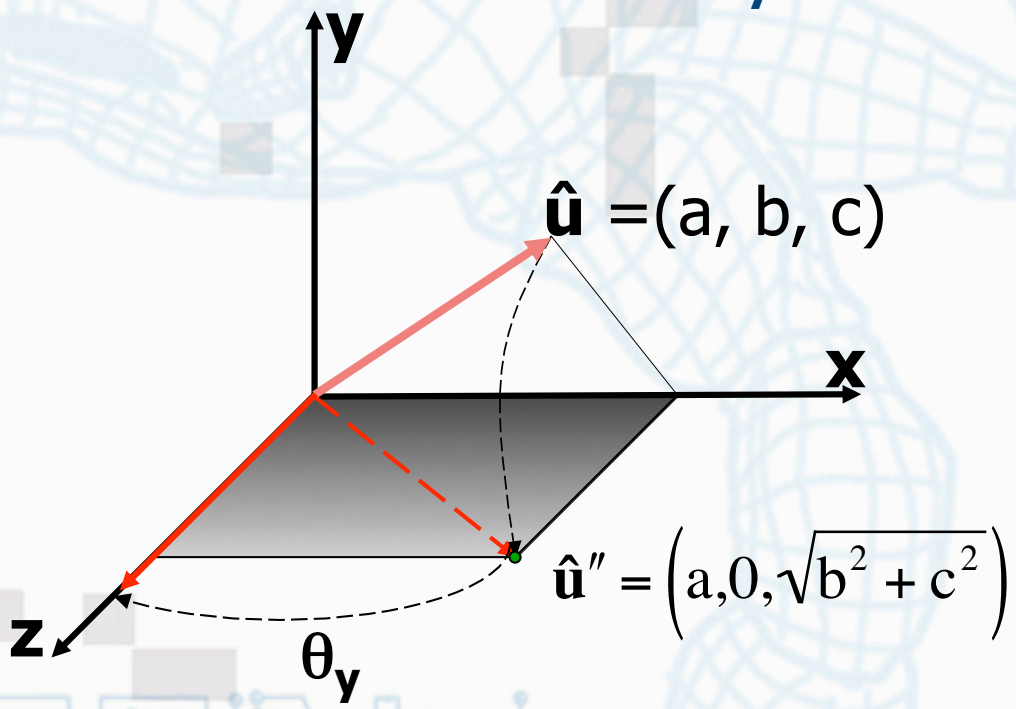
– girar de um ângulo θ_x sobre o eixo x

$$\mathbf{R}_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & c/d & -b/d & 0 \\ 0 & b/d & c/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.8 Concatenação de Transformações (A18)

<u>Passo 2.2</u>: Alinhamento do eixo de rotação com o eixo z

- **determinar ângulo θ_y entre û" e k**
- **girar û" de um ângulo $-\theta_y$ sobre o eixo y**



4.8 Concatenação de Transformações (A18)

- **determinar ângulo θ_y entre û" e k**
  - **Encontre o cos θ_y**

$$\cos\theta_y = \frac{\hat{\mathbf{u}}''\cdot\mathbf{k}}{|\hat{\mathbf{u}}''||\mathbf{k}|} = \frac{d}{\sqrt{a^2+d^2}}$$

$$\text{mas } \sqrt{a^2+d^2} = \sqrt{a^2+b^2+c^2} = 1 \Rightarrow \cos\theta_y = d$$

  - **Encontre o sen θ_y**

$$sen\theta_y = a \Rightarrow sen\left(-\theta_y\right) = -a$$

4.8 Concatenação de Transformações (A18)

– girar de um ângulo – θ_y sobre o eixo x

$$\mathbf{R}_y(-\theta_y) = \begin{bmatrix} d & 0 & -a & 0 \\ 0 & 1 & 0 & 0 \\ a & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.8 Concatenação de Transformações (A18)

Passo 3: Rotação de θ sobre o eixo z

$$\mathbf{R}_z(\theta) = \begin{bmatrix} \cos\theta & -\text{sen}\,\theta & 0 & 0 \\ \text{sen}\,\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.8 Concatenação de Transformações (A18)

- **Passo 4**: Inverso do passo 2.2, $R_y(\theta_y)$
- **Passo 5**: Inverso do passo 2.1, $R_x(-\theta_x)$
- **Passo 6**: Inverso do passo 1, $T(P_1)$
- **Assim a Rotação em torno do eixo P_1P_2**

$$\mathbf{M} = \mathbf{T}(\mathbf{P}_1)\mathbf{R}_x(-\theta_x)\mathbf{R}_y(\theta_y)\mathbf{R}_z(\theta)\mathbf{R}_y(-\theta_y)\mathbf{R}_x(\theta_x)\mathbf{T}(-\mathbf{P}_1)$$

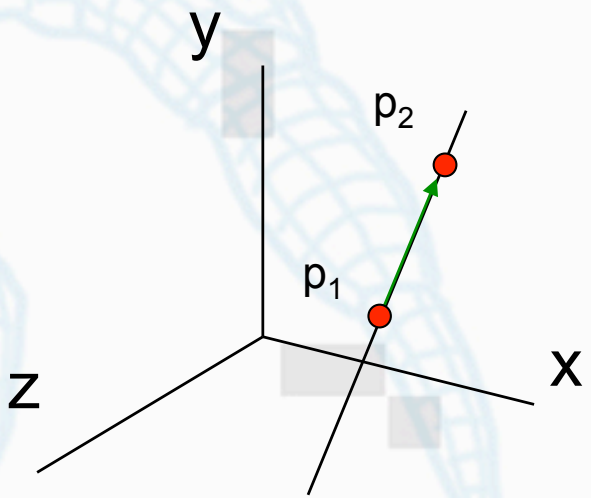
$$\mathbf{M} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.8 Concatenação de Transformações (A18)

- **Rotação sobre um eixo arbitrário**

  Maneira alternativa

  - **Construir uma base ortogonal i' j' k' tal que k' seja o vetor unitário na direção P_1P_2**
  - **Transformar os pontos da cena para esse sistema de coordenadas**
  - **Girar em torno de z'**
  - **Transformar os ponto para o sistema original**



4.9 Matrizes de Transformação no OpenGL (A19)

4.9.1 A Matriz de transformação corrente

- Aplicada aos vértices subseqüentes à sua montagem
- É uma matriz 4 x 4
- Inicializada como a matriz identidade
- Modificada por pós-multiplicação de matrizes
  - C ← CT (Translação)
  - C ← CS (Escala)
  - C ← CR (Rotação)

4.9 Matrizes de Transformação no OpenGL (A19)

- **Definida diretamente com o comando glLoadMatrix{fd}(M);**
  - **C ← M**
  - **M é um array com 16 elementos de tipo float ou double que compõem 4 a 4 as colunas de C**

$$\mathbf{M} = \begin{bmatrix} m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \\ m_4 & m_8 & m_{12} & m_{16} \end{bmatrix}$$

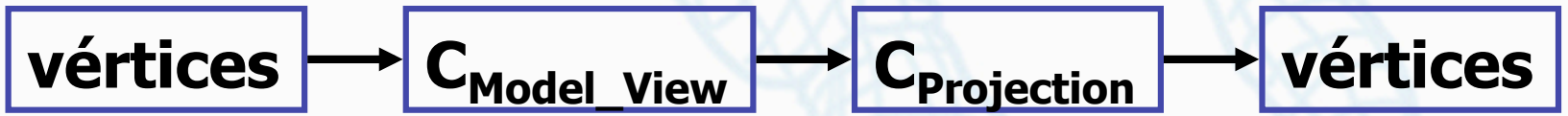
4.9 Matrizes de Transformação no OpenGL (A19)

- **Modificada com o comando glMultMatrix{fd}(M);**
  - **C ← CM**
  - **M é um array com 16 elementos de tipo float ou double que compõem 4 a 4 as colunas da matriz**

$$\mathbf{M} = \begin{bmatrix} m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \\ m_4 & m_8 & m_{12} & m_{16} \end{bmatrix}$$

4.9 Matrizes de Transformação no OpenGL (A19)

- Os vértices são transformados pela matriz corrente resultante de dois modos
  - GL_PROJECTION (modo de projeção)
  - GL_MODELVIEW (modo de modelagem e visualização)
  - glMatrixMode(modo): Seleciona um dos dois modos



- $C \leftarrow C_{Projection} C_{Model\_View}$

4.9 Matrizes de Transformação no OpenGL (A19)

4.9.2 Rotação, translação e escala

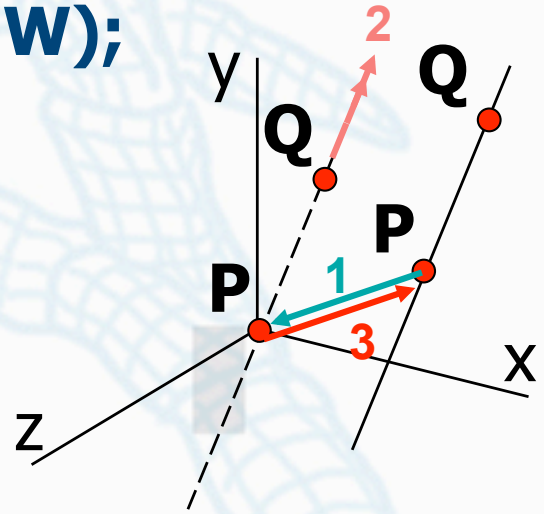
- glRotatef(ângulo, vx, vy, vz);
  - Ângulo em graus
  - (vx, vy, vz) vetor na direção da rotação
- glTranslatef(dx, dy, dz);
  - (dx, dy, dz) vetor de deslocamento
- glScalef(sx, sy, sz);
  - sx, sy, sz: fatores de escala

4.9 Matrizes de Transformação no OpenGL (A19)

4.9.3 Rotação em torno de um ponto fixo

- Composição de transformações

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(Px, Py, Pz);
glRotatef(ângulo, vx, vy, vz);
glTranslatef(-Px, -Py, -Pz);
```

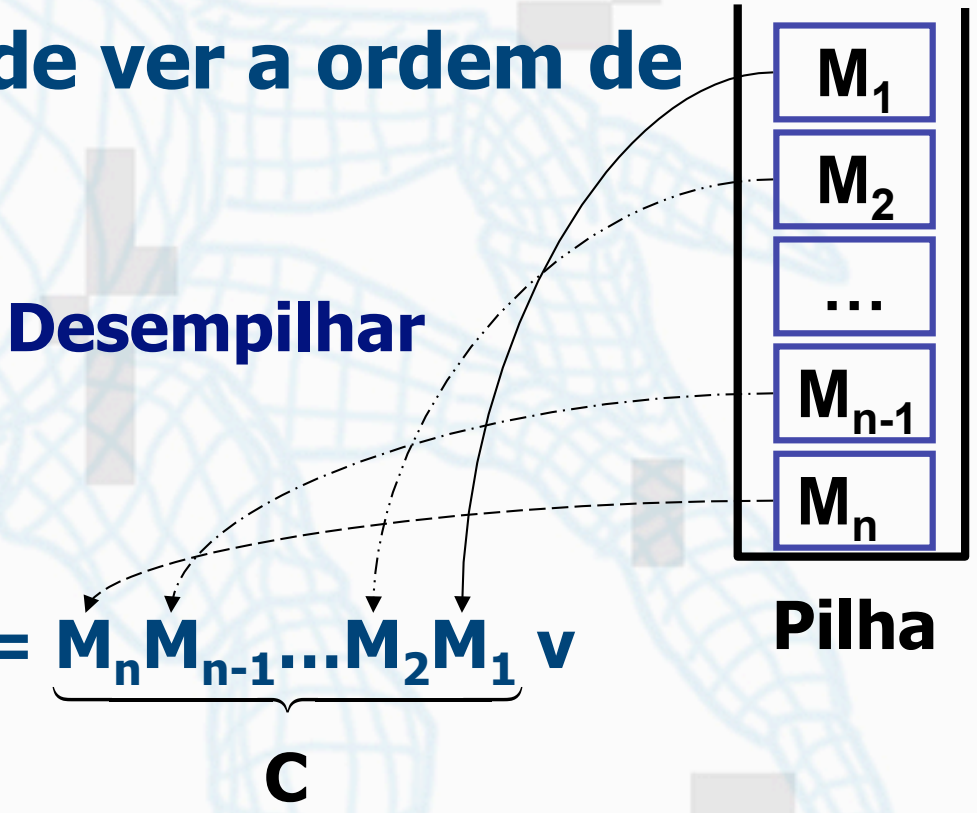


- $v' = T(P_x, P_y, P_z)\ R(\alpha, v_x, v_y, v_z)\ T(-P_x, -P_y, -P_z)\ v$

4.9 Matrizes de Transformação no OpenGL (A19)

4.9.4 Ordem de transformações

- Duas maneiras de ver a ordem de transformações
  - $C \leftarrow CM_n$
  - $C \leftarrow CM_{n-1}$
  - ...
  - $C \leftarrow CM_2$
  - $C \leftarrow CM_1$



4.9 Matrizes de Transformação no OpenGL (A19)

4.9.5 Rotação de um cubo sem translação

- Funções Callback
  - glutDisplayFunc (display)
  - glutIdleFunc (spincube)
  - glutMouseFunc (mouse)
  - glutKeybordFunc(keyboard)

4.9 Matrizes de Transformação no OpenGL (A19)

- Função **display**
  - Aplica as matrizes de rotação em torno dos eixos x y e z
- Função **spincube**
  - Incrementa o ângulo de rotação em torno do eixo selecionado e chama **display**
- Função **mouse**
  - Seleciona o eixo de rotação
- Função **keyboard**
  - Encerra o programa pressionando ‘q’ ou ‘Q’

Código Fonte Código Exec

4.9 Matrizes de Transformação no OpenGL (A19)

4.9.6 Loading, pushing, e popping de matrizes

- Loading matix M como matriz corrente
  - glLoadMatrixf(M);
- Salvando a matriz corrente na pilha
  - glPushMatrix()
- Retirando a matriz da pilha e tornando-a corrente
  - glPopMatrix()

4.10 Interfaces para Aplicações 3D (A19)

4.10.1 Usando áreas do screen

- Motion callback retorna: botão ativado e posição do mouse
- Botão esquerdo pressionado
  - Mouse no centro: nenhuma rotação
  - Mouse movendo para cima: rotação y+
  - Mouse movendo para baixo: rotação y-
  - Mouse movendo para direita: rotação x+
  - Mouse movendo para esquerda: rotação x-
  - Mouse movendo para os cantos: rotação combinada x y

4.10 Interfaces para Aplicações 3D (A19)

- **Botão direito pressionado**
  - Mouse no centro: nenhuma translação
  - Mouse movendo para cima: translação y+
  - Mouse movendo para baixo: translação y-
  - Mouse movendo para direita: translação x+
  - Mouse movendo para esquerda: translação x-
  - Mouse movendo para os cantos: translação combinada x y
- **Botão do meio pressionado**
  - Mouse movendo para direita: translação z+
  - Mouse movendo para esquerda: translação z-

4.10 Interfaces para Aplicações 3D (A19)

- **Velocidade do movimento**
  - **Quanto mais longe do centro mais rápido o movimento**

Código Fonte

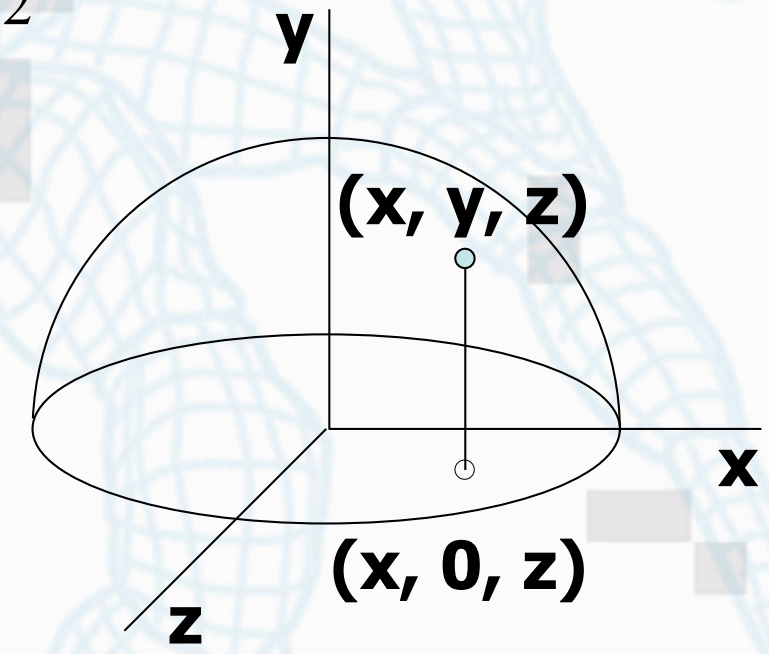
Código Exec

4.10 Interfaces para Aplicações 3D (A19)

4.10.2 Uma Trackball virtual

- Dada uma posição (x, z) no plano
  - Calcule o ponto no hemisfério

$$y = \sqrt{1 - x^2 - z^2}$$



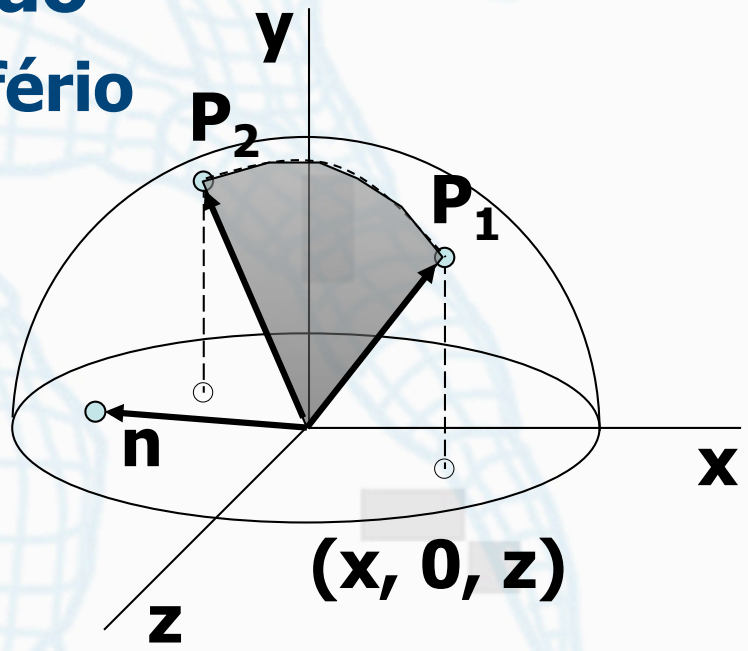
4.10 Interfaces para Aplicações 3D (A19)

4.10.2 Uma Trackball virtual

- Primeiro clique no botão do mouse
  - Define ponto P_1 no hemisfério
- Mouse move, liberar botão
  - Define ponto P_2 no hemisfério
- Calcula vetor normal
  - $n = P_1 \times P_2$

Código Fonte

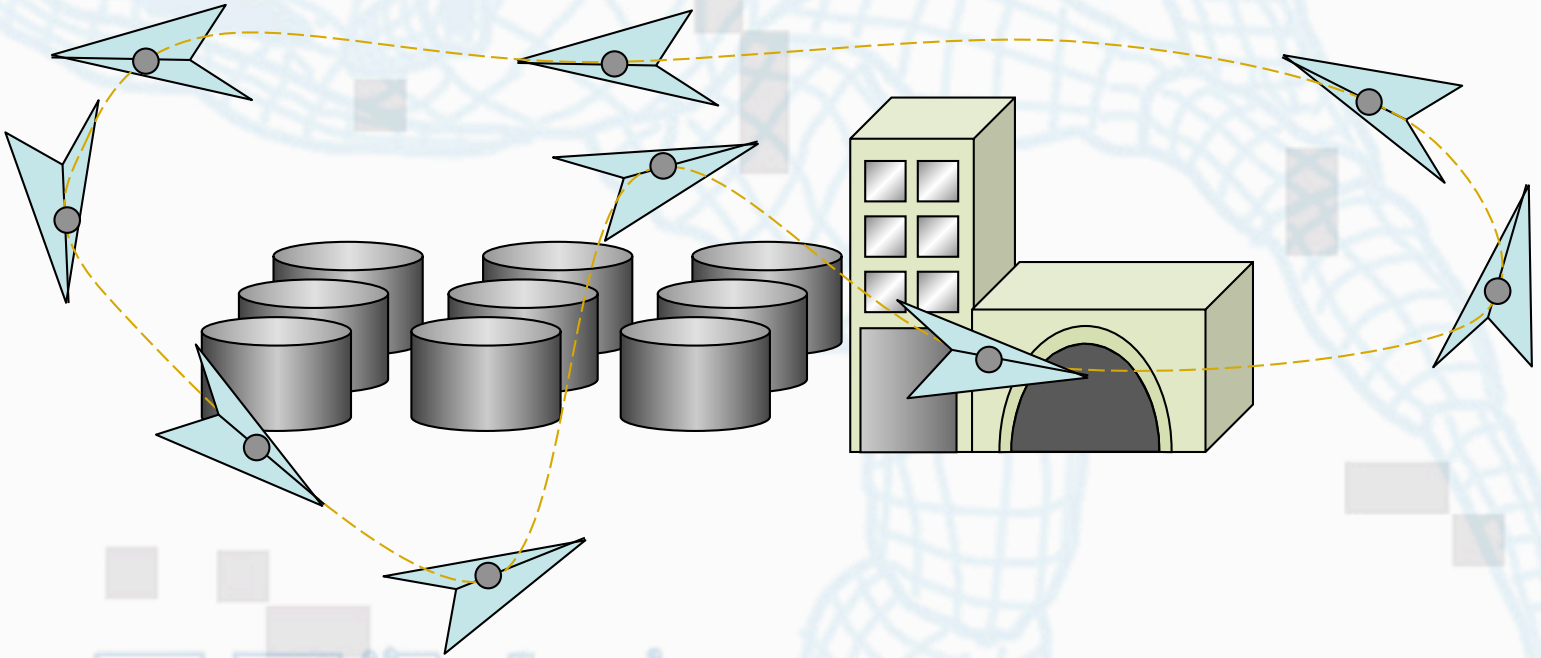
Código Exec



4.10 Interfaces para Aplicações 3D (A19)

4.10.3 Rotações suaves

- A partir dos key-frames indicados
  - gerar aviões em posições intermediárias ao longo da rota, interpolando as orientações

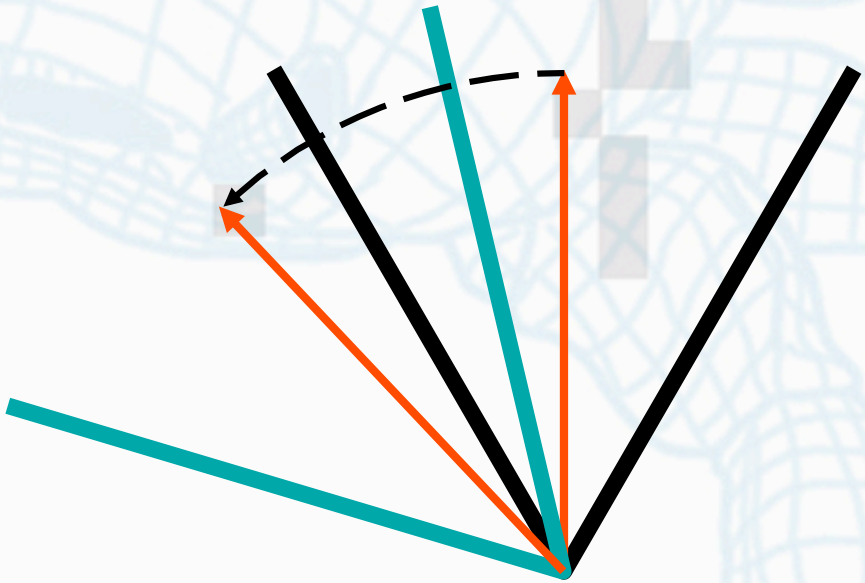


4.10 Interfaces para Aplicações 3D (A19)

- As orientações dos aviões nos key-frames são obtidas através de matrizes de rotação
  - $R = R_x(\theta_x)\, R_y(\theta_y)\, R_z(\theta_z)$
    - $E=(\theta_x, \theta_y, \theta_z)$: Ângulos de Euler
- Considere dois conjuntos de ângulos de Euler
  - E_i e E_{i+1}
  - Calcular E_k = Interpolação(E_i, E_{i+1})
  - Animação não ficará suave

4.10 Interfaces para Aplicações 3D (A19)

- Modificar a orientação interpolando entre os vetores vermelhos como no caso da trackball



4.10 Interfaces para Aplicações 3D (A19)

4.10.4 Rotações incrementais

- Como no caso da trackball virtual
  - Achar o círculo ao longo da superfície esférica que leva de uma orientação à outra
  - Incrementar o ângulo de rotação ao longo do eixo de rotação v = (dx, dy, dz)

Fim da Aula 13

4.11 Quaternions (A19)

Introduction

- Quaternions are commonly used to represent rotations
- They were introduced by William Hamilton (1805-1865) [1]
- Quaternions were conceived as Geometrical Operators
- A Complete Calculus of Quaternions was introduced by Hamilton [2]

Definition of Vector

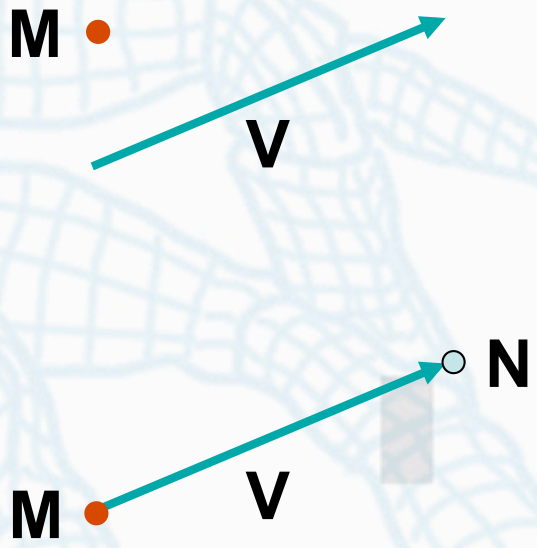
- It is a line segment with orientation
- It represents the relative position of two points
  - Example: point N with respect to M

Hamilton's Motivation for Quaternions

- **Create a Mathematical Concept to represent**
  - **RELATIONSHIP between two VECTORS**
  - **Similar to the way a Vector represents**
    - **RELATIONSHIP between two POINTS**

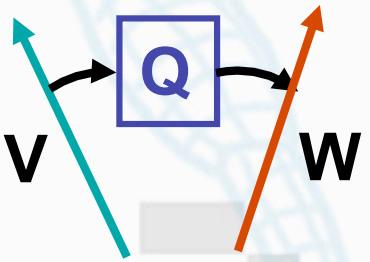
Vector applied to Point

- Given
  - Point M and
  - Vector V
- Application of V over M results in
  - Unique Point N



Quaternion applied to Vector

- **Hamilton wanted that given**
  - **Vector *V* and**
  - **Quaternion *Q***
- **Application of *Q* over *V* results in**
  - **Unique Vector *W***



Quatenion Rationale

- Vector is completely defined by
  - Length
  - Orientation
- To define a vector in terms of another vector → a Quaternion has to represent
  - Relative Length
  - Relative Orientation

Definition of Scalar

- **Scale**
  - **RATIO between the lengths of two PARALLEL vectors A and B**

s = Scale

  - **RELATIVE LENGTH of vectors**

Scalar - Vector Operations

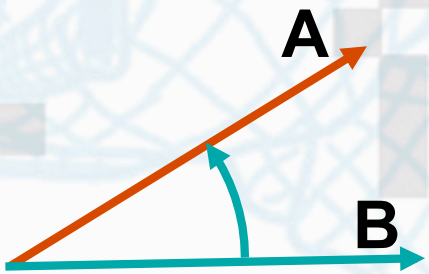
- $S = \dfrac{\mathbf{A}}{\mathbf{B}}$ **S is the Quotient between two PARALLEL vectors A and B**

$$\mathbf{A} = S \Diamond \mathbf{B} = \frac{\|\mathbf{A}\|}{\|\mathbf{B}\|}\mathbf{B} = s\mathbf{B}$$

- **A Scalar is an Operator that**
  - **Changes the SCALE of the vector**
  - **Keeps its orientation unchanged**

Definition of Versor

- **Versor**
  - **QUOCIENT between two NON-PARALLEL vectors A and B of EQUAL LENGTH**



  - **RELATIVE ORIENTATION of vectors**

Versor - Vector Operations

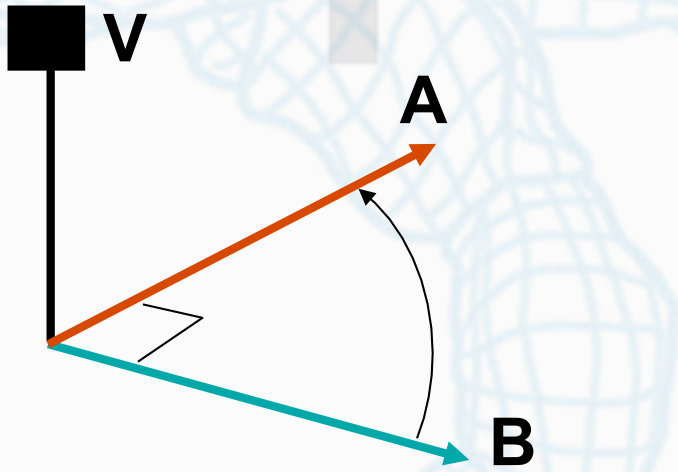
- $V = \frac{\mathbf{A}}{\mathbf{B}}$ ***V*** **is the Geometric Quotient between two NON-PARALLEL vectors of EQUAL LENGTH A and B**

$$\mathbf{A} = V \langle\rangle \mathbf{B}$$

- **A Versor is an Operator that**
  - **Changes the ORIENTATION of the vector**
  - **Keeps its LENGTH unchanged**

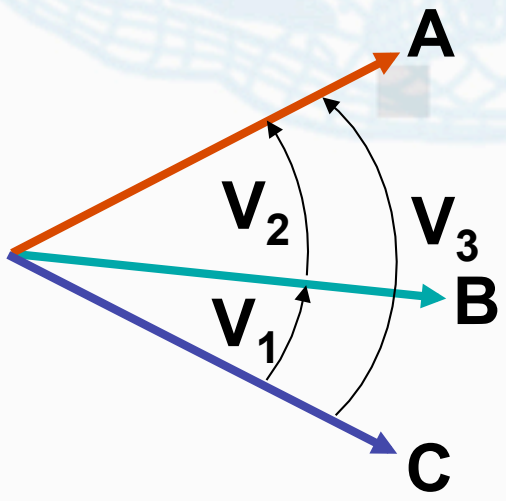
Right Versors

- A *Right Versor* is a Versor that applies a 90^o rotation
- Vector length is left unchanged as in any other Versor application



Composing Versors

- **Versor composition is the consecutive application of two versors operators**



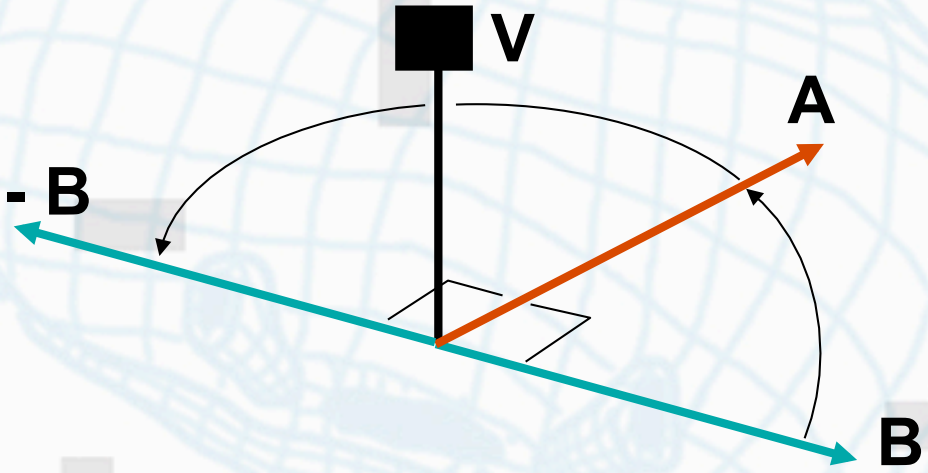
$$\mathbf{A} = V_2 \lozenge \mathbf{B}$$

$$\mathbf{B} = V_1 \lozenge \mathbf{C}$$

$$\mathbf{A} = V_2 \lozenge V_1 \lozenge \mathbf{C} = V_3 \lozenge \mathbf{C}$$

$$V_3 = V_2 \lozenge V_1$$

Composing Right Versors



$$\mathbf{A} = V \lozenge \mathbf{B}$$

$$-\mathbf{B} = V \lozenge \mathbf{A} = V \lozenge V \lozenge \mathbf{B}$$

$$-1 = V \lozenge V$$

- **(-1) is the INVERSION operator**
  - **Double application of a right versor to a vector**
  - **Inverts the direction of a vector**

Definition of Quaternion

- **Geometrical Quotient of two vectors *A* and *B***

$$Q = \frac{\mathbf{A}}{\mathbf{B}} \Leftrightarrow \mathbf{A} = Q \lozenge \mathbf{B}$$

- **Operating on a vector, changes its**
  - **ORIENTATION**
  - **LENGTH**

Quaternion Characteristics

- **Axis(Q)** = Unit Vector perpendicular to the plane of rotation
- **Angle(Q)** = Angle between the vectors in the quotient
- **Index(Q)** = In a Right Quaternion is the Axis(Q) multiplied by the length ratio of the two vectors in the quotient

Representation of Quaternions

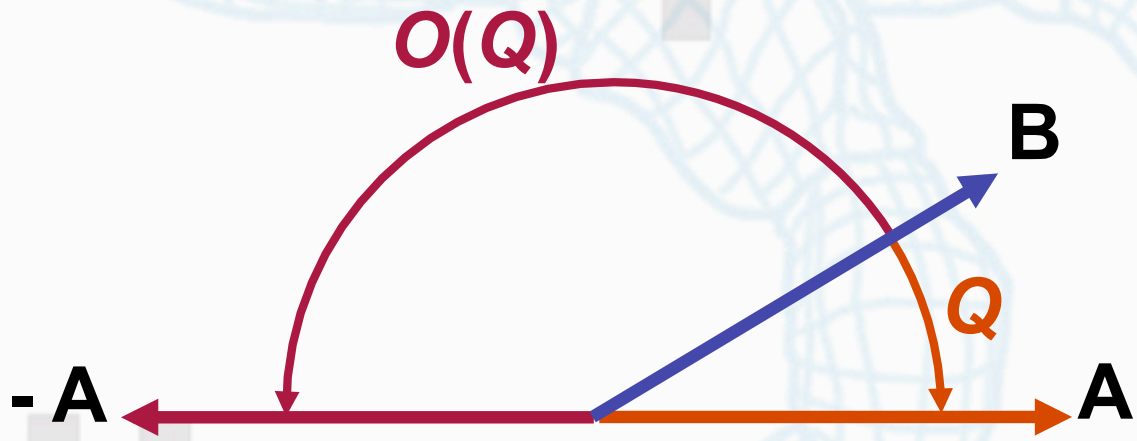
- **Quaternion = "*A set of Four*"**
- **From**
  - **the Latin *Quaternio***
  - **the Greek τετρακτυς (ΤΕΤΡΑΚΤΥΣ)**
- **The combined operation of *Scalar* and *Versor* requires 4 numbers:**
  - **1 for Scale**
  - **1 for Angle**
  - **2 for Orientation (common plane)**
- **Quaternion = Scalar ∪ Versor**

Opposite Quaternions

- The quaternion *Q*
- has an *Opposite* quaternion *O*(*Q*)

$$Q = \frac{\mathbf{A}}{\mathbf{B}}$$

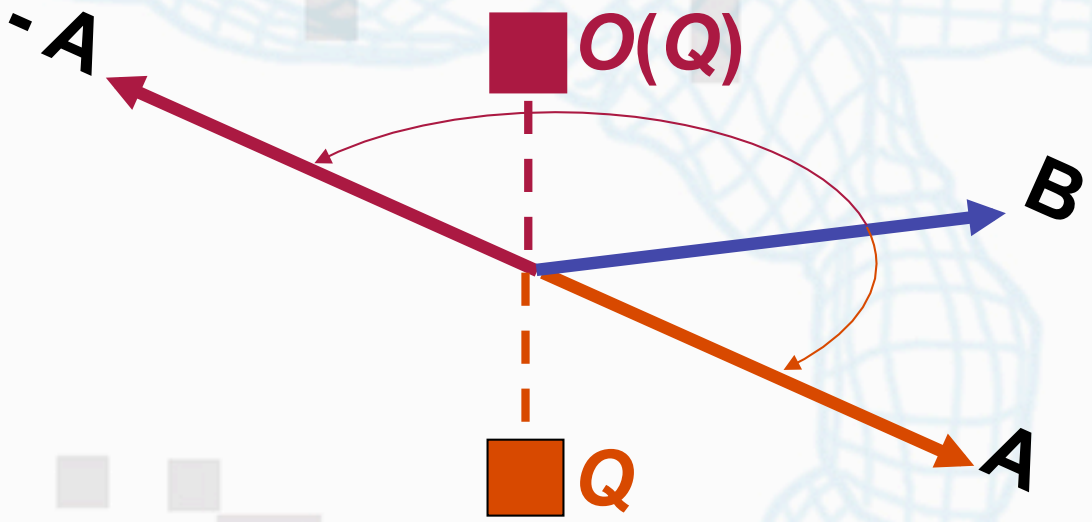
$$O(Q) = \frac{-\mathbf{A}}{\mathbf{B}} = -Q$$



Opposite Quaternion Properties

$$Angle(Q) + Angle(O(Q)) = \pi$$

$$Axis(Q) = -Axis(O(Q))$$



Reciprocal Quaternions

- **The quaternion Q:**

$$Q = \frac{\mathbf{A}}{\mathbf{B}} \Rightarrow \mathbf{A} = Q \lozenge \mathbf{B}$$

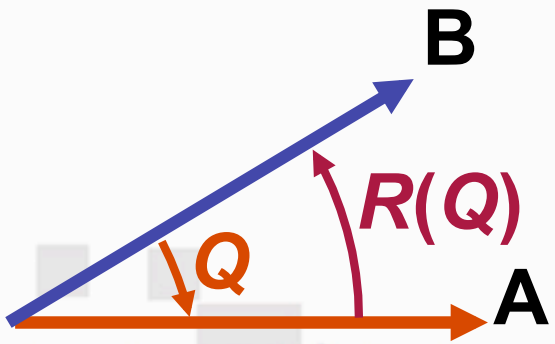
- **Reciprocal of Q:**

$$R(Q) = Q^{-1} = \frac{\mathbf{B}}{\mathbf{A}} \Rightarrow \mathbf{B} = Q^{-1} \lozenge \mathbf{A}$$

- **Composition**

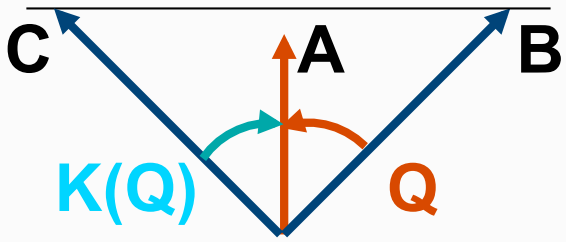
$$R(Q) \lozenge Q = \mathbf{1}$$

(1) is an Identity Operator



Conjugate Quaternion

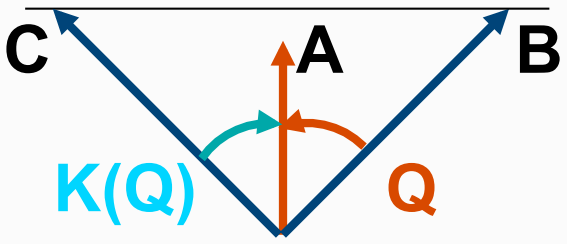
- Given
  - Vectors A and B and
  - Quotient Q=A/B
- Geometric reflection of vector *B* over vector *A* will be vector *C*



$$Q = \frac{\mathbf{A}}{\mathbf{B}} \Rightarrow \mathbf{A} = Q \Diamond \mathbf{B}$$

Conjugate Quaternion

- **Conjugate Quaternion K(Q)=A/C**
  - **Angle (K(Q)) = Angle (Q)**
  - **Axis (K(Q)) = - Axis (Q)**



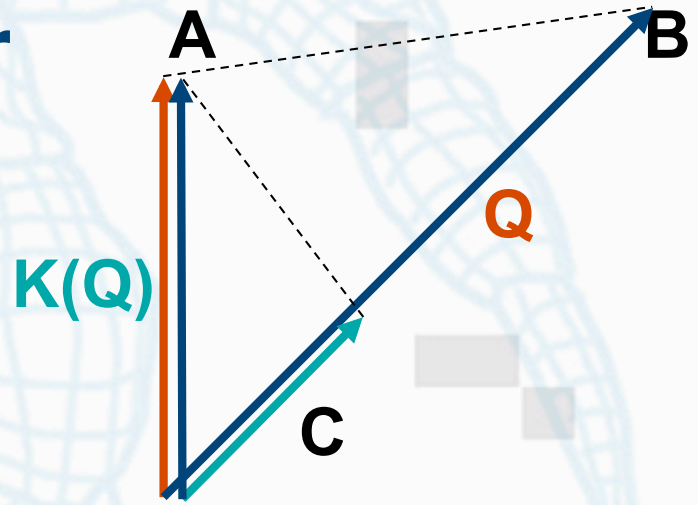
$$K(Q) = \frac{\mathbf{A}}{\mathbf{C}} \Rightarrow \mathbf{A} = K(Q) \lozenge \mathbf{C}$$

Norm of a Quaternion

- The *Norm* is the composition of a Quaternion with its Conjugate
  - The rotation of K(Q) compensates the rotation of Q
  - Producing a parallel vector
  - N(Q) is a scalar operator

$$N(Q) = \frac{\mathbf{C}}{\mathbf{B}} = \frac{\mathbf{C}}{\mathbf{A}}\frac{\mathbf{A}}{\mathbf{B}} = \left(\frac{\|\mathbf{A}\|}{\|\mathbf{B}\|}\right)^2$$

$$\mathbf{C} = N(Q)\lozenge\mathbf{B} = K(Q)\lozenge Q\lozenge\mathbf{B}$$

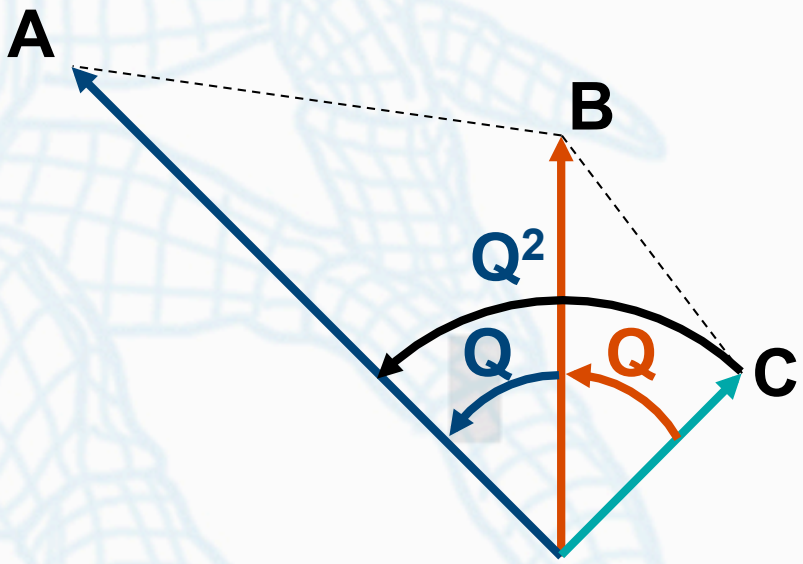


Square of a Quaternion

- **Defined as applying Q twice**

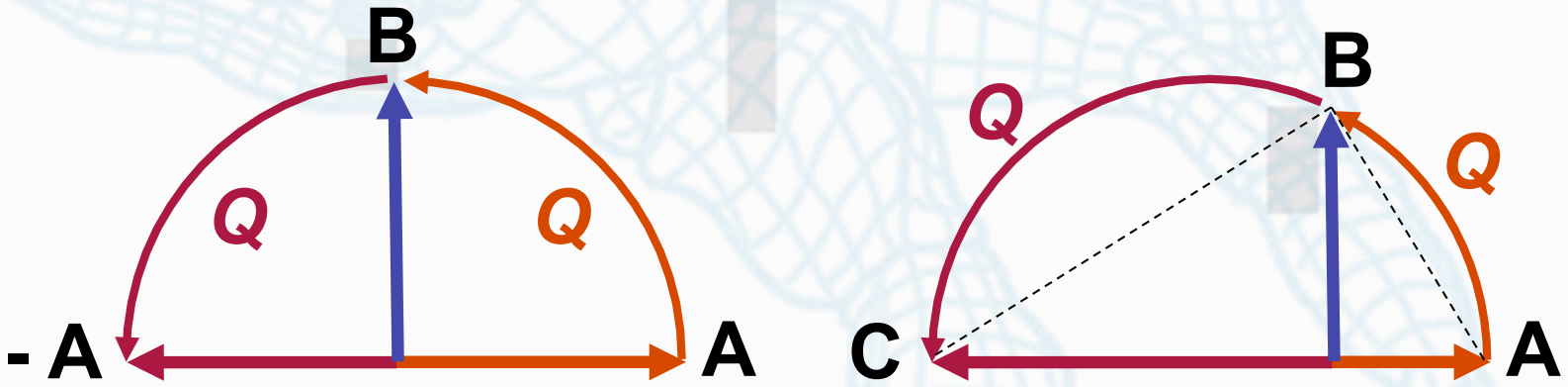
$$Q^2 = \frac{\mathbf{A}}{\mathbf{C}} = \frac{\mathbf{A}}{\mathbf{B}}\frac{\mathbf{B}}{\mathbf{C}}$$

$$\mathbf{A} = Q^2 \lozenge \mathbf{C} = Q \lozenge Q \lozenge \mathbf{C}$$



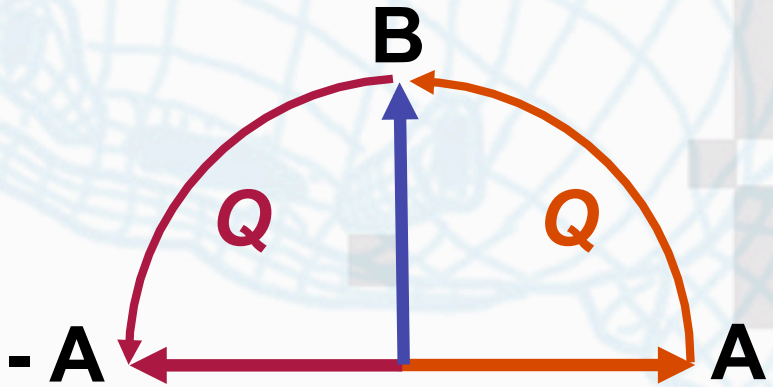
Composing Right Quaternions

- **Succesive application of a Right Quaternion over a Vector**
  - **results in a Vector in the opposite direction**



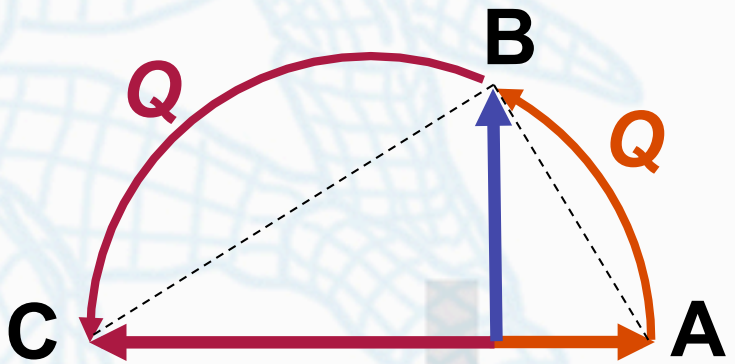
Composing Right Quaternions

- **The square of any right quaternion is a NEGATIVE scalar operator**



$$\left(\frac{\mathbf{B}}{\mathbf{A}}\right)^2 = \frac{-\mathbf{A}}{\mathbf{A}} = -1$$

$$-\mathbf{A} = Q\lozenge(Q\lozenge\mathbf{A})$$



$$\frac{\mathbf{C}}{\mathbf{A}} = \left(\frac{\mathbf{B}}{\mathbf{A}}\right)^2 = -\left(\frac{\|\mathbf{B}\|}{\|\mathbf{A}\|}\right)^2$$

$$\mathbf{C} = Q^2\mathbf{A}$$

Versor of a Quaternion

- ***Versor of* a Vector**
  - **Unit vector parallel to the vector**

$$U(\mathbf{A}) = \frac{\mathbf{A}}{\|\mathbf{A}\|} = \hat{\mathbf{A}}$$

- ***Versor of* a Quaternion**
  - **Quotient of the Versors of the vectors**

$$U(Q) = U\left(\frac{\mathbf{A}}{\mathbf{B}}\right) = \frac{U(\mathbf{A})}{U(\mathbf{B})} = \frac{\hat{\mathbf{A}}}{\hat{\mathbf{B}}}$$

- **It is the part of the Quaternion that represents Relative Orientation**

Tensor of a Quaternion

- ***Tensor of* a Vector**
  - **Length of the vector**

$$T(\mathbf{A}) = \|\mathbf{A}\|$$

- ***Tensor of* a Quaternion**
  - **Quotient of the Tensors of the vectors**

$$T(Q) = T\left(\frac{\mathbf{A}}{\mathbf{B}}\right) = \frac{T(\mathbf{A})}{T(\mathbf{B})} = \frac{\|\mathbf{A}\|}{\|\mathbf{B}\|}$$

- **It is the part of the Quaternion that represents Relative Scale**

Tensor and Versor of a Quaternion

- **Versor operator**
  - **Applies *VERSION* to a vector**
  - **Changes vector's orientation**
- **Tensor operator**
  - **Applies *TENSION* to a vector**
  - **Stretches the vector and change its length**

Tensor and Versor of a Quaternion

- **Vector decomposed in Versor and Tensor parts**

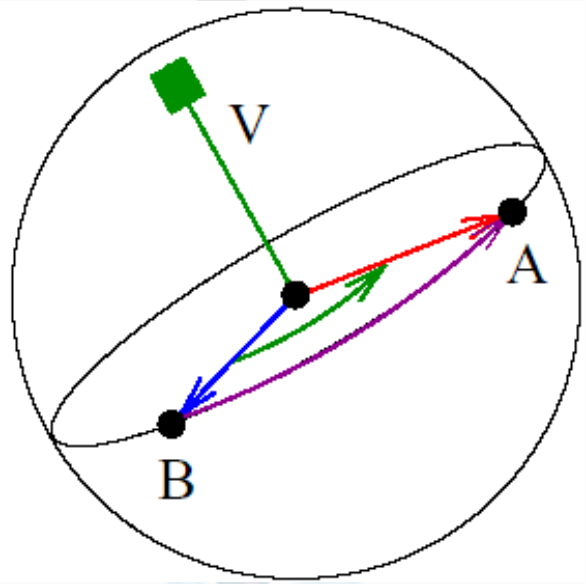
$$\mathbf{A} = T(\mathbf{A}) \lozenge U(\mathbf{A}) = \|\mathbf{A}\| \lozenge \hat{\mathbf{A}}$$

- **Quaternion decomposed in Versor and Tensor parts**

$$Q = T(Q) \lozenge U(Q) = \frac{T(\mathbf{A})}{T(\mathbf{B})} \lozenge \frac{U(\mathbf{A})}{U(\mathbf{B})}$$

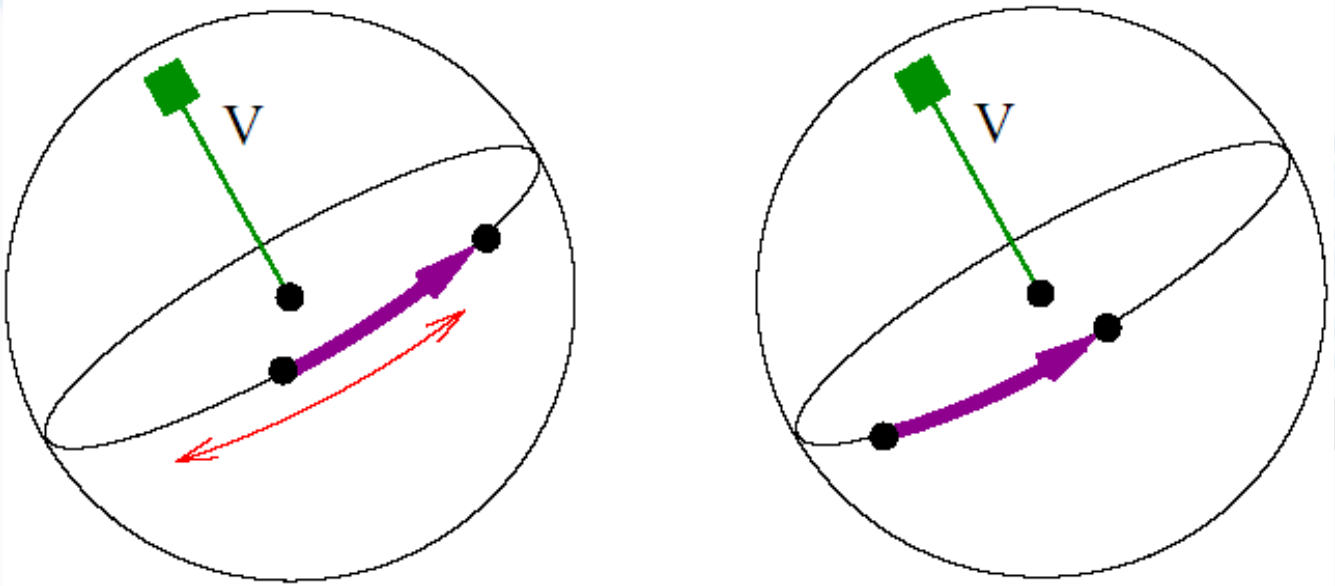
Vector - Arcs

- ***Versors*** **represented on the surface of a unit sphere**
- **Versor *V* moves point *B* to point *A***
- **Minimum Arc joining points *B* and *A* is defined as *Vector-Arc***



Sliding Vector - Arcs

- **Vectors can be translated on a plane**
- ***Vector arcs* can freely slide along the great circle and still represent the SAME *Versor***



Composition of Biplanar Versors

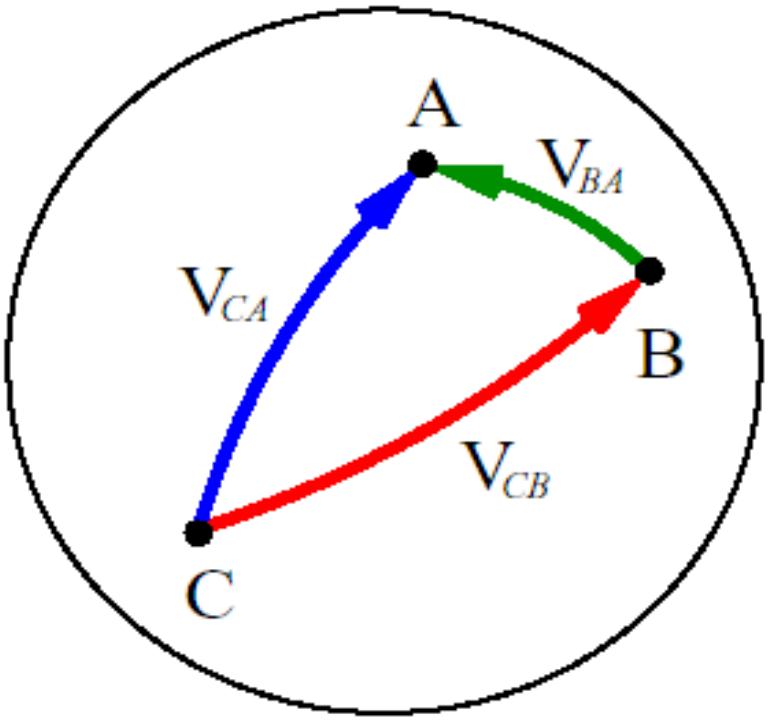
$$V_{BA} = \frac{\vec{A}}{\vec{B}}$$

composed with

$$V_{CB} = \frac{\vec{B}}{\vec{C}}$$

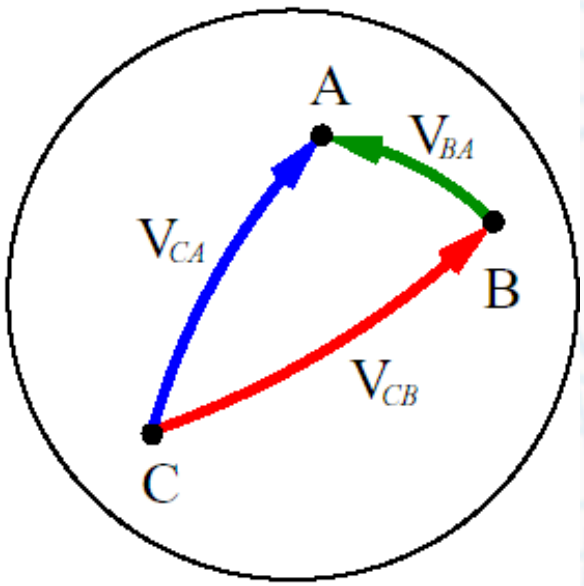
results in the versor

$$V_{CA} = \frac{\vec{A}}{\vec{B}} \diamond \frac{\vec{B}}{\vec{C}} = \frac{\vec{A}}{\vec{C}}$$



Multiplication and Division of Biplanar Versor

- ***Spherical Triangle ABC***
  - **Used to define versor operations**
  - **Analogous to parallelogram rule for vectors**

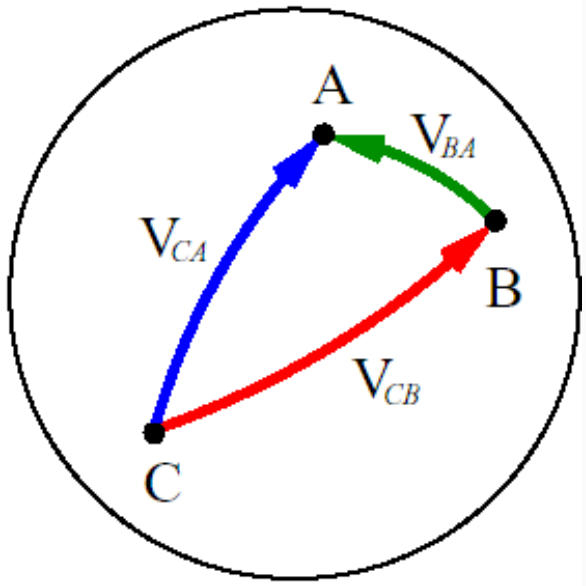


Multiplication and Division of Biplanar Versor

- ***Multiplication of Versors***
  - **like the sum of vectors**

$$V_{CA} = V_{BA} V_{CB}$$

- ***Division of Versors***
  - **like the difference of vectors**



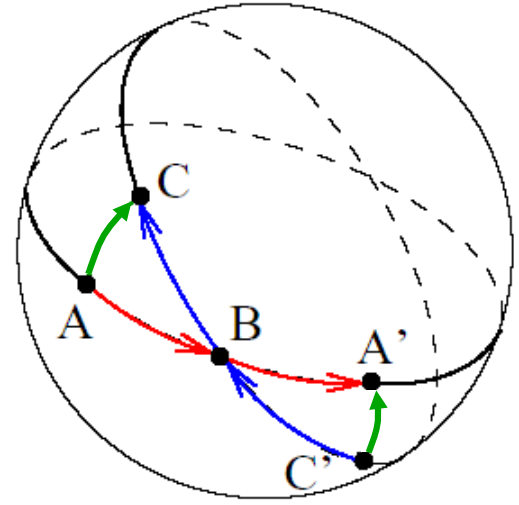
$$V_{BA} = \frac{V_{CA}}{V_{CB}} = \frac{\frac{\mathbf{A}}{\mathbf{C}}}{\frac{\mathbf{B}}{\mathbf{C}}} = \frac{\mathbf{A}}{\mathbf{C}} \frac{\mathbf{C}}{\mathbf{B}} = \frac{\mathbf{A}}{\mathbf{B}}$$

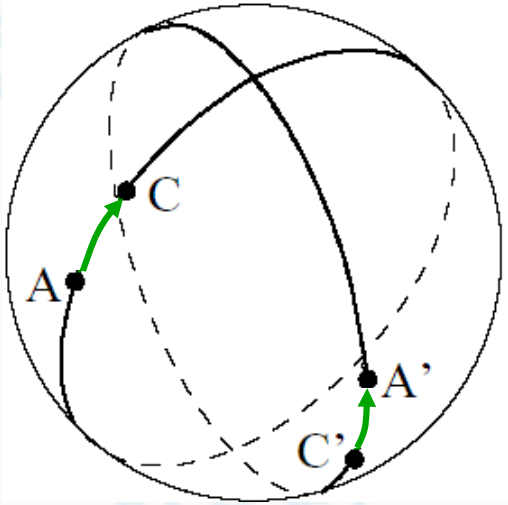
Versor Composition is Non-Commutative

- $V_{AC} = V_{BC}V_{AB} \neq$

  $\neq V_{AB}V_{BC} = V_{BA'}V_{C'B} = V_{C'A'}$

- Versors V_{AC} and $V_{C'A'}$ have
  - Same angle
  - Different axes (different planes)





Fim da Aula 14

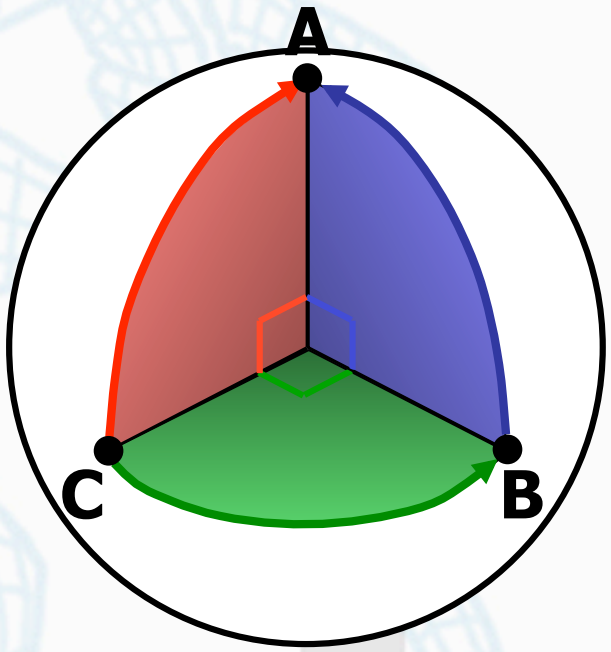
Composition of two Orthogonal Right Versors

- **Multiplication of two orthogonal Right Versors produce a Right Versor orthogonal to them**

$$V_{BA}V_{CB} = V_{CA}$$

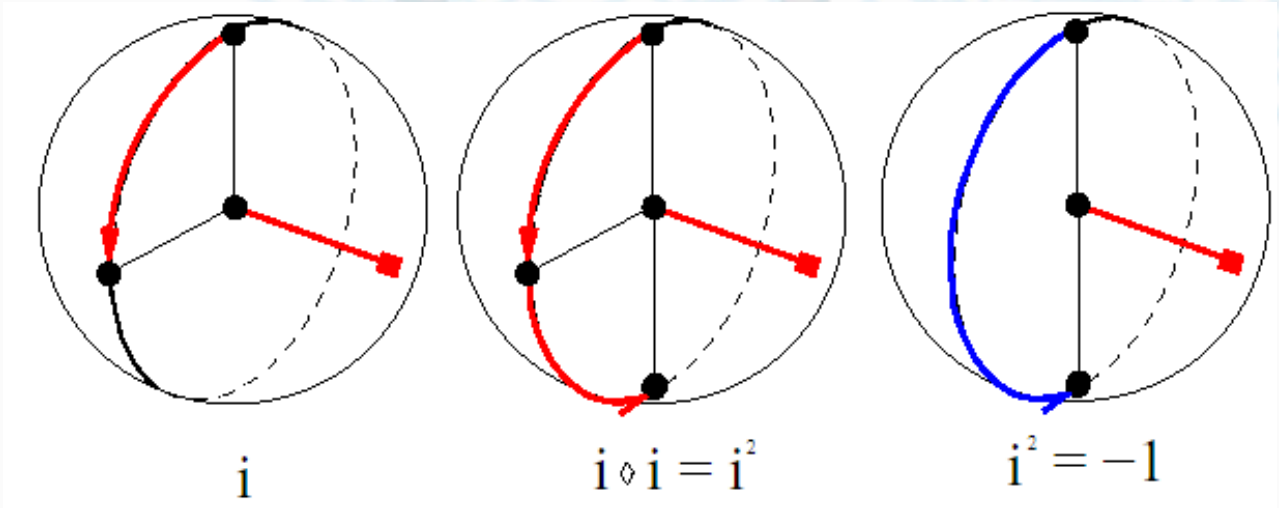
- **When order is reversed**

$$V_{CB}V_{BA} = -V_{CA}$$

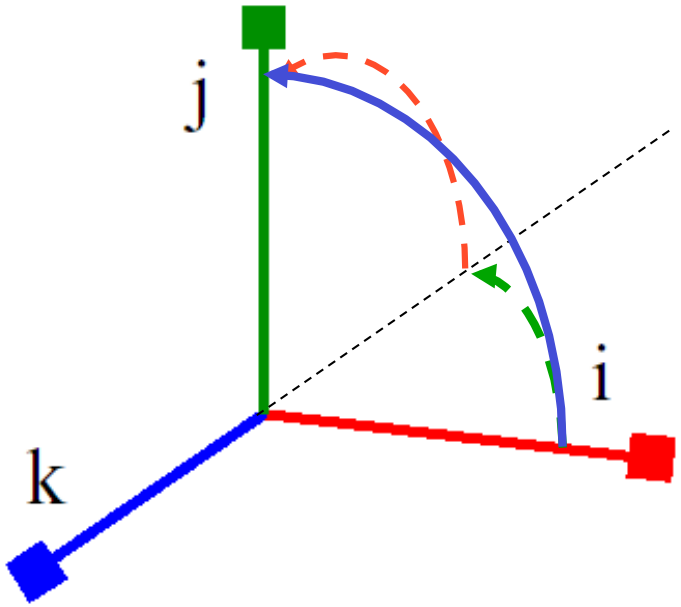


Square of Elementary Versors

- The *Square of* an operator is the operator applied twice
- The square of Right Versors is always the (-1) Operator



Elementary Versors

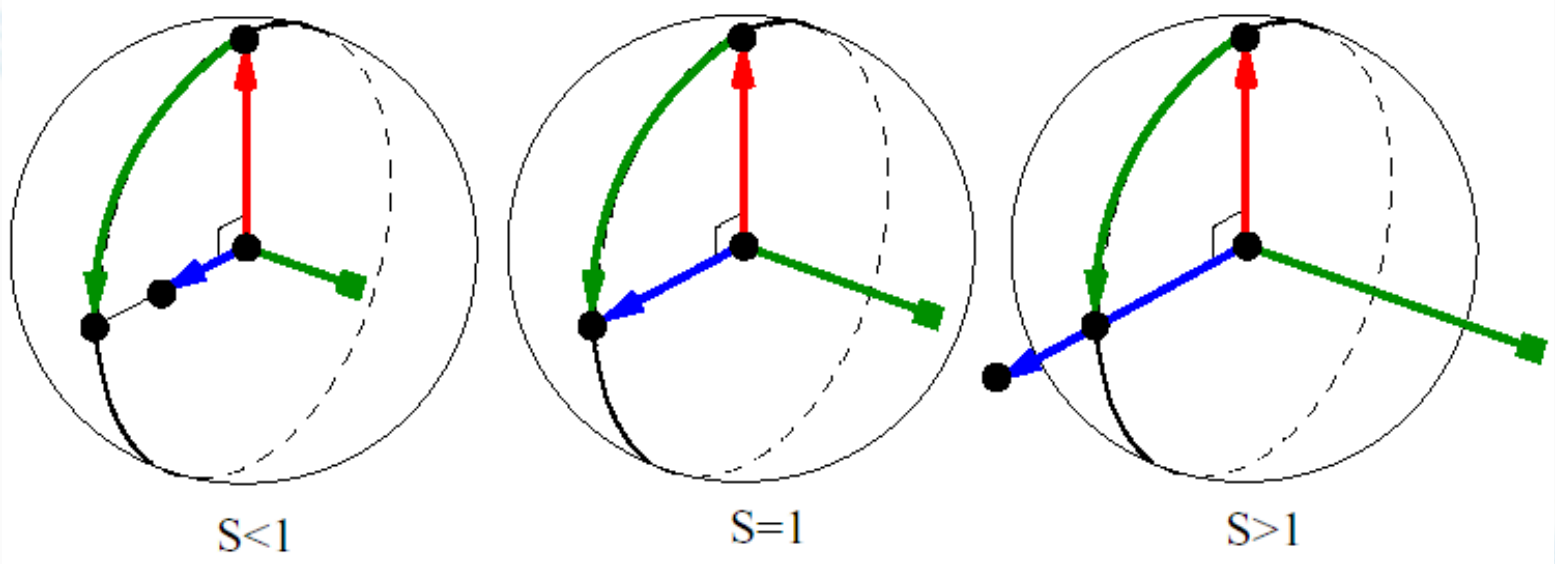


Composition of Elementary Versors

right-hand	self
$i \cdot j = k$	$i \cdot i = -1$
$j \cdot k = i$	$j \cdot j = -1$
$k \cdot i = j$	$k \cdot k = -1$

Index of Right Quaternions

- **Is the Axis of the quaternion Scaled by the ratio of lengths**

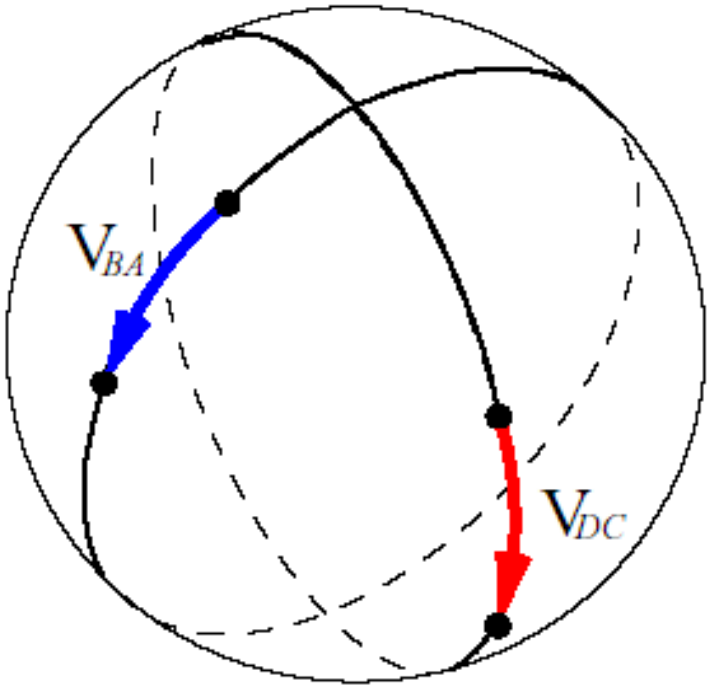


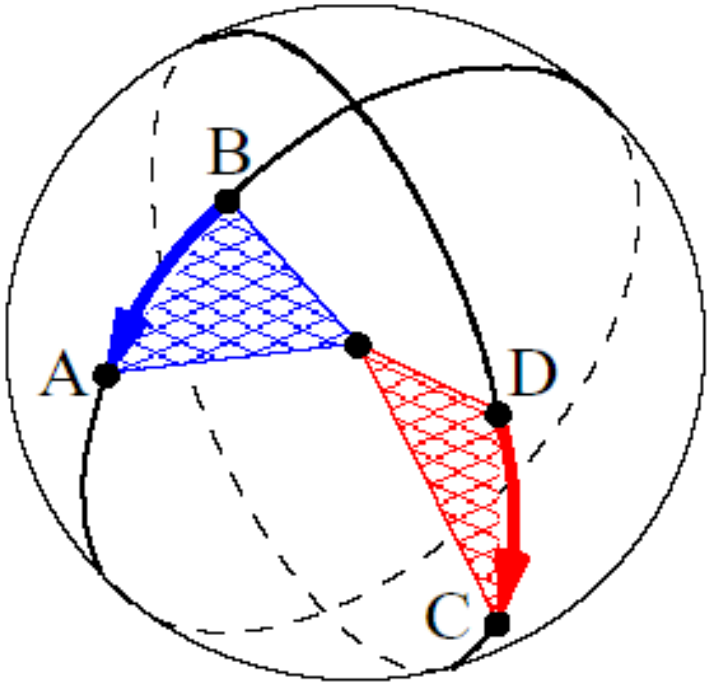
Sum of Versors

- **Versors are Quotients**
  - **They can be summed ONLY when they have a COMMON DENOMINATOR**
  - **A Common Denominator can ALWAYS be found**

$$\left.\begin{aligned} V_{BC} &= \frac{\vec{C}}{\vec{B}} \\ V_{BA} &= \frac{\vec{A}}{\vec{B}} \end{aligned}\right\} \quad V_{BC} + V_{BA} = \frac{\vec{C}}{\vec{B}} + \frac{\vec{A}}{\vec{B}} = \frac{\vec{C} + \vec{A}}{\vec{B}}$$

Getting a Common Denominator



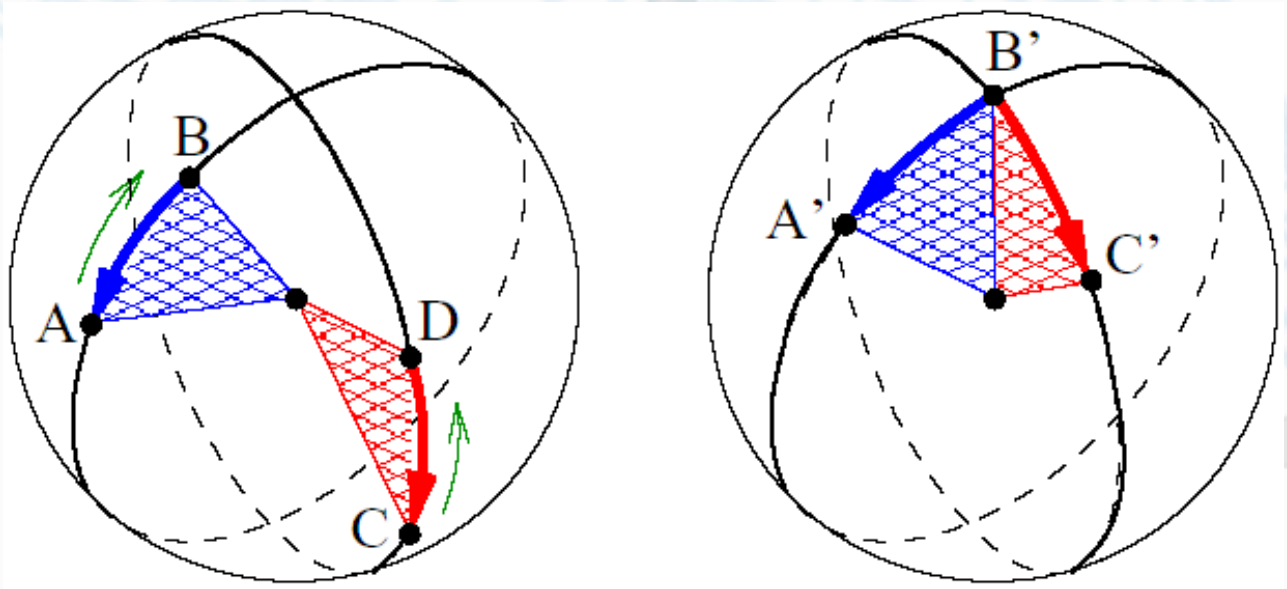


$$V_{BA} = \frac{\vec{A}}{\vec{B}}$$

$$V_{DC} = \frac{\vec{C}}{\vec{D}}$$

Getting a Common Denominator

- **Slide both versors along their great circles until their origins coincide**
- **The vector *B'* in the intersection is the common denominator**



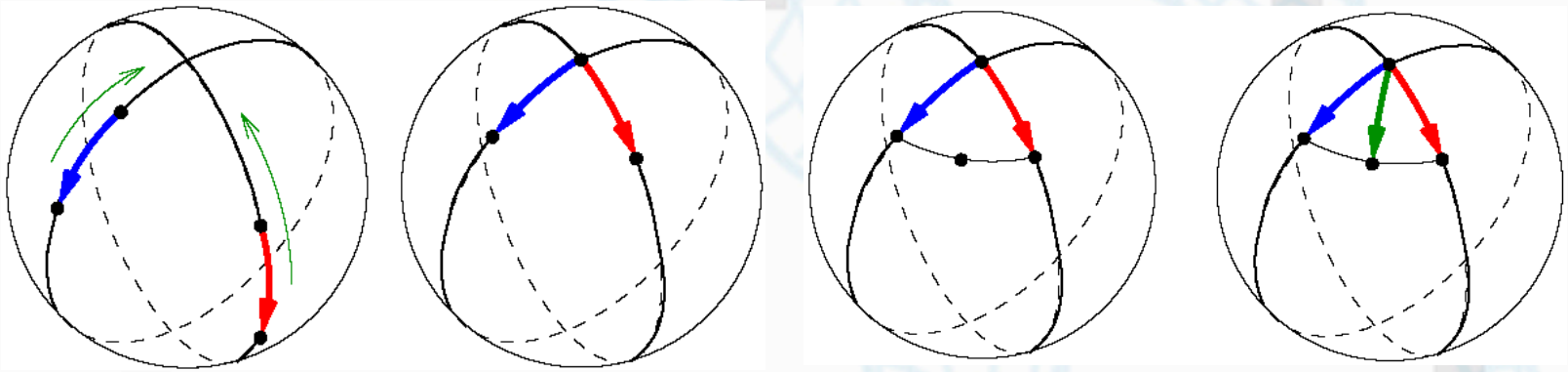
Geometrical Interpretation of the Sum

- **As with Vectors**

  1) Get a common denominator SLIDING both V*ector-Arcs* to a common origin

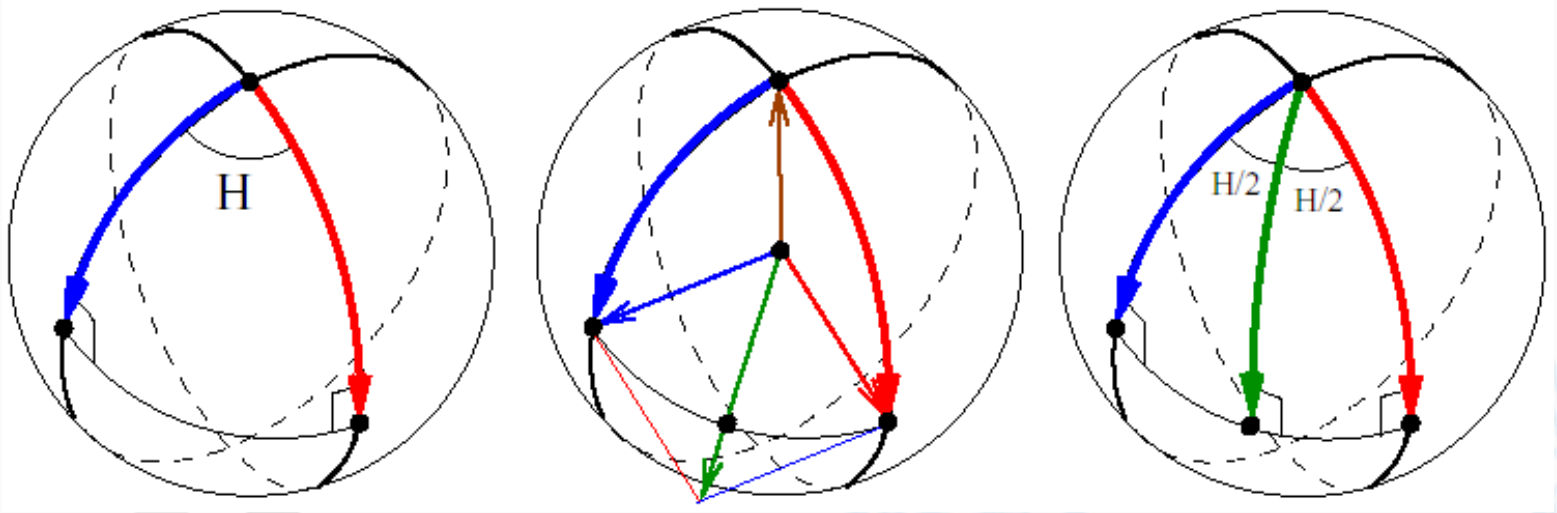
  2) Add the two vectors in the numerator

  3) Get the new Quotient



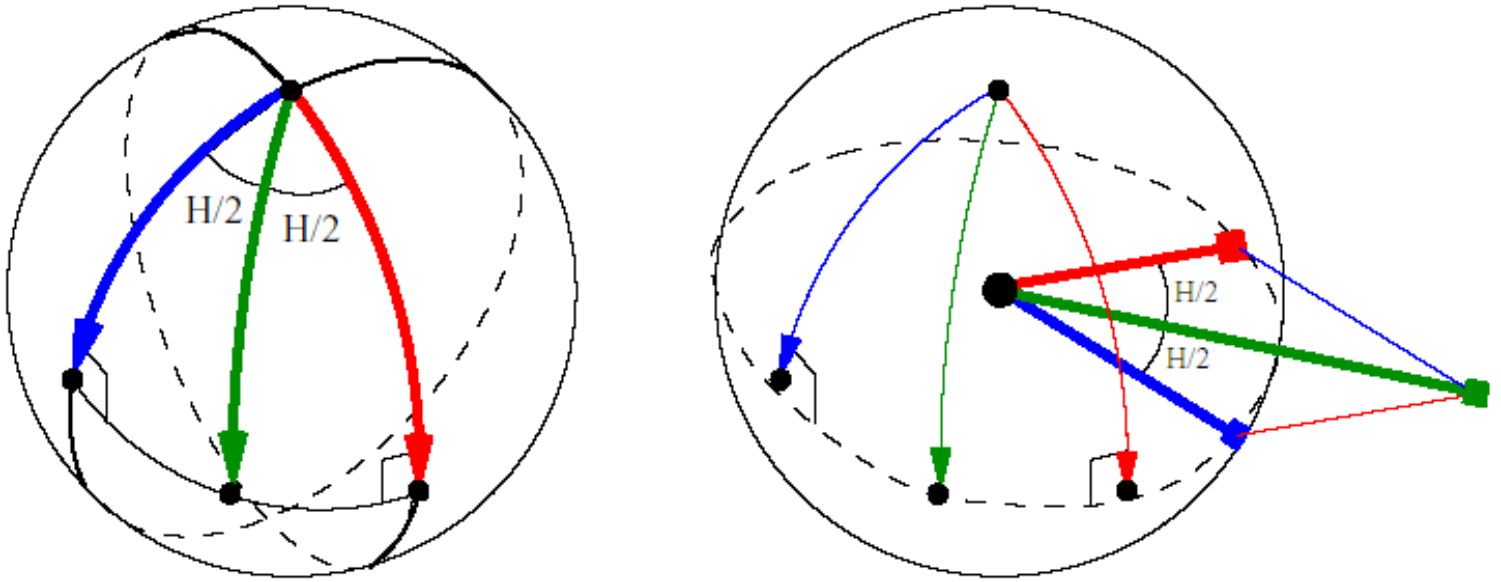
Sum of two Right Versors

- It is always a right quaternion
- Its plane BISECTS those of the original two versors
- and has a Scalar characteristic > 1



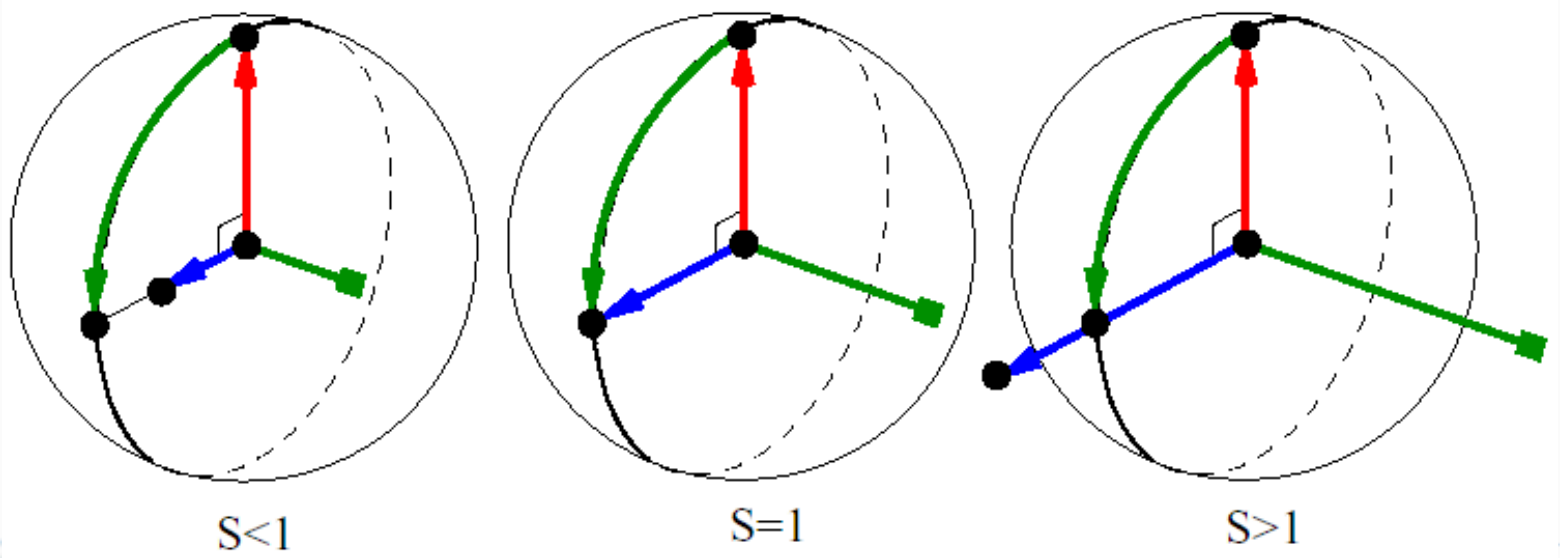
Sum of two Right Versors

- ***Index*** **of resulting Versor = sum of indices of two versors**



Multiplying a Right Versor by a Scalar

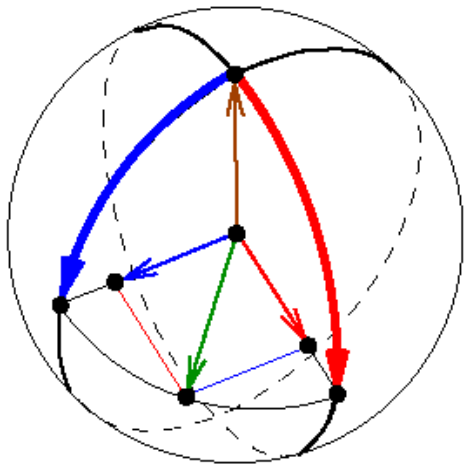
- **Multiplication by a Scalar**
  - **Affects only Scalar part of the Right Versor**
  - **Modifies the length ratio of the vectors in the Quotient**



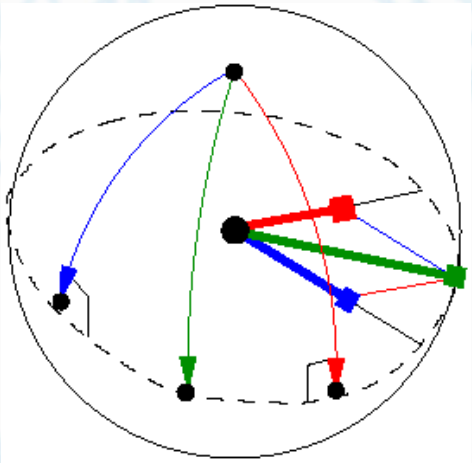
Right Versor in terms of Orthogonal Right Versor

- If the Orthogonal Right Versors i, j, k are multiplied by Scalars *x, y, z*
- Their sum will be a Right Versor whose axis has (*x, y, z*) as components

$$Q = x\,\mathbf{i} + y\,\mathbf{j} + z\,\mathbf{k}$$



$$x^2 + y^2 + z^2 = 1$$

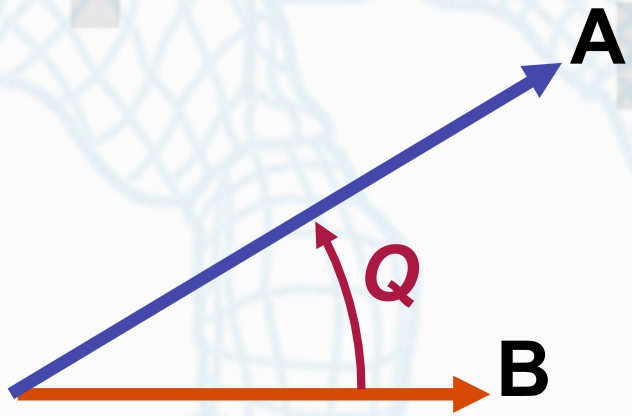


Scalar and Right Parts of Quaternions

- A Quaternion operator applied to a vector B
- performs an operation that produces another vector A

$$Q = \frac{\vec{A}}{\vec{B}}$$

$$\vec{A} = Q \diamond \vec{B}$$

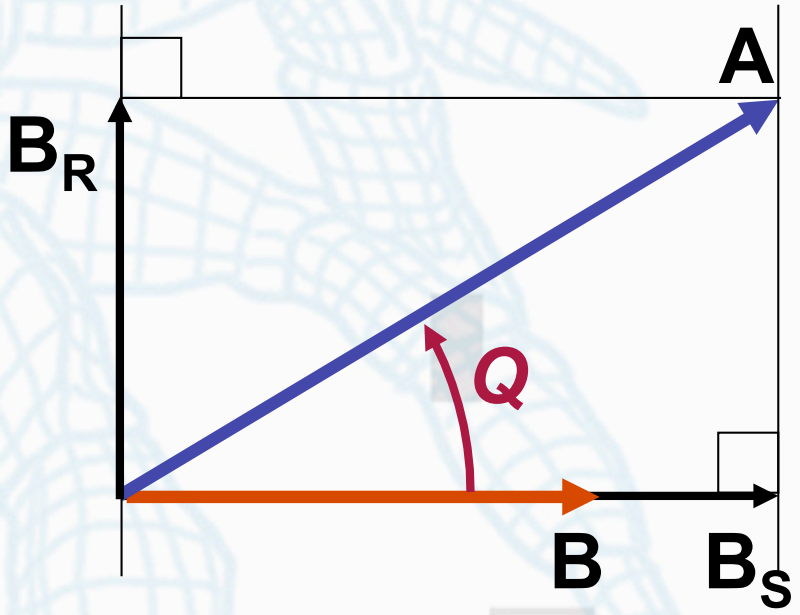


Scalar and Right Parts of Quaternions

- The new vector A can be expressed as a sum of two orthogonal vectors

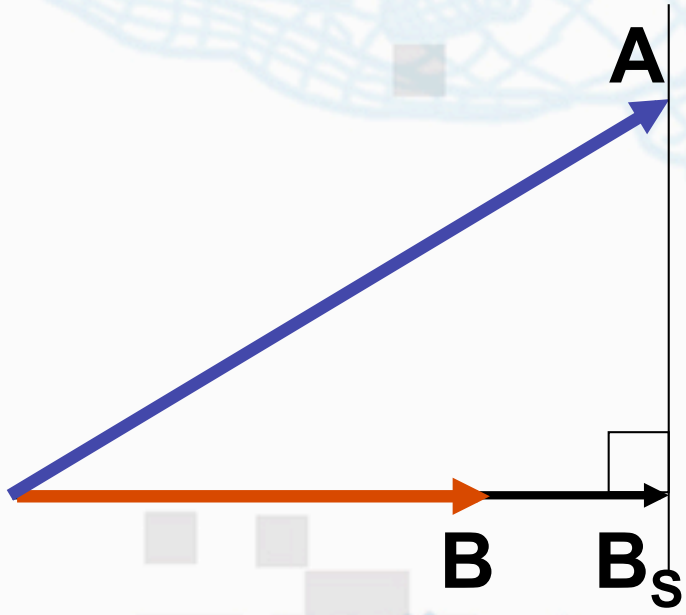
  $A = B_S + B_R$

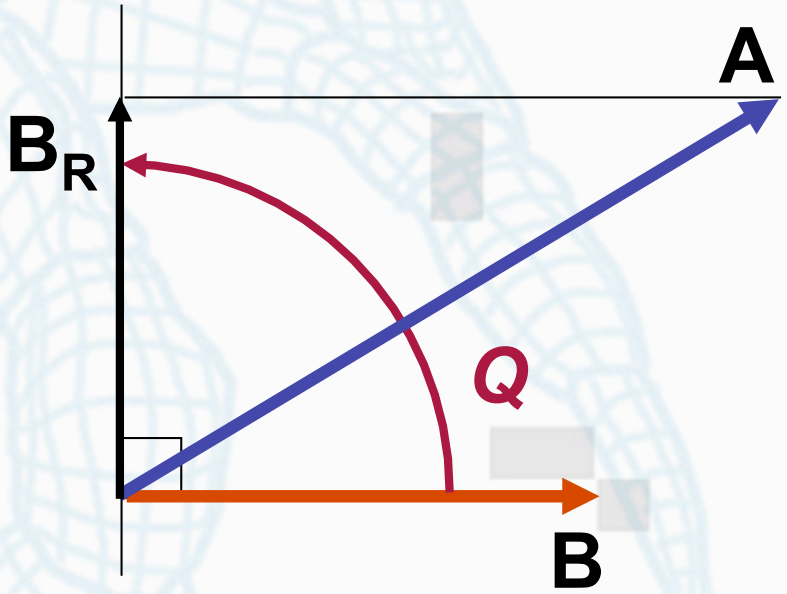
  - B_S parallel to B and
  - B_R orthogonal to B



Scalar and Right Parts of Quaternions

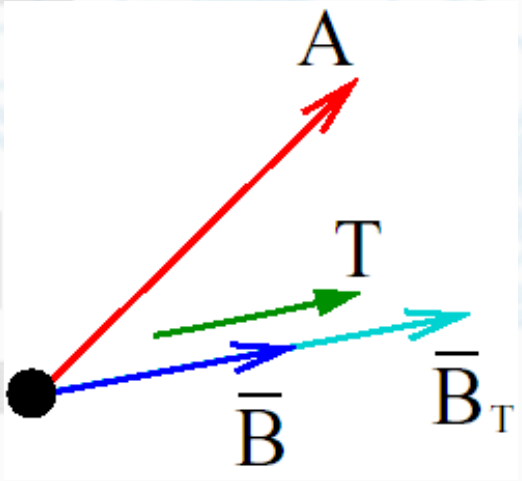
- **Apply Scalar operator to B**
  - **Obtain B_S**
- **Apply Right quaternion to *B***
  - **Obtain B_R**

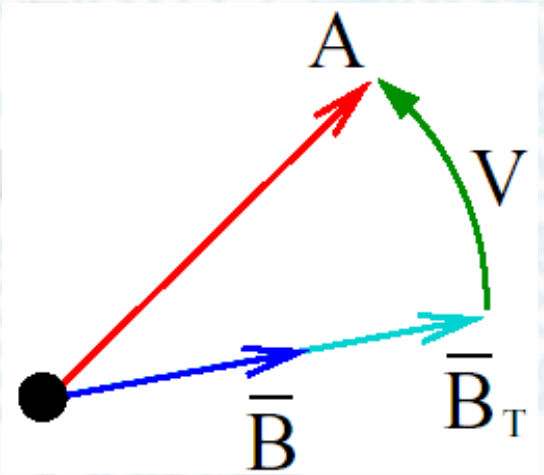




Tensor and Versor Part of a Quaternion

- **Same operation can be decomposed in Tensor and Versor Operators**





$$\mathbf{B}_T = T \langle\rangle \mathbf{B} \qquad \mathbf{A} = V \langle\rangle \mathbf{B}_T$$

Scalar and Right versus Tensor and Versor

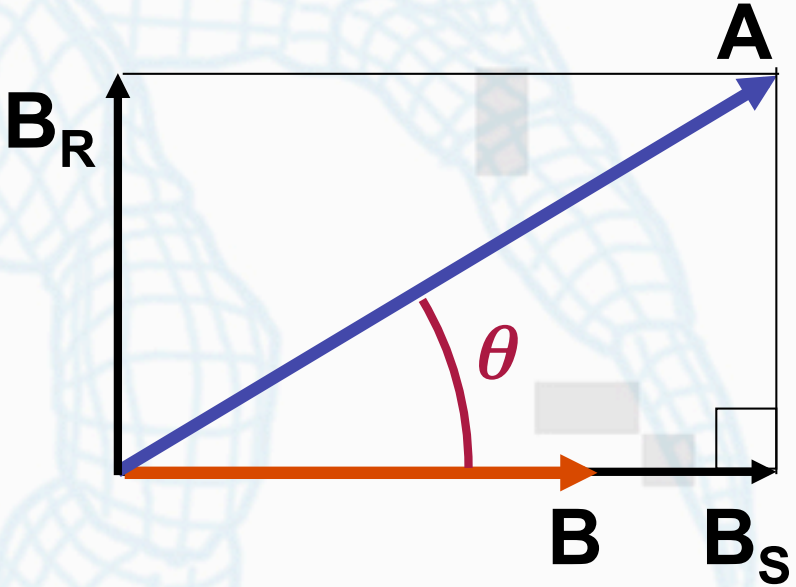
- **Scalar** and **Right** parts are representations in **rectangular** coordinates
- **Tensor** and **Versor** parts are representations in **polar** coordinates

Quaternions as Four Coefficients

- Let L be the Ratio of lengths between vectors A and B $L = \frac{\|\mathbf{A}\|}{\|\mathbf{B}\|}$
- Thus, the scalar factor S is

$$S = \frac{\|\mathbf{B}_s\|}{\|\mathbf{B}\|} = \frac{\|\mathbf{A}\|\cos(\theta)}{\|\mathbf{B}\|}$$

$$S = L\cos(\theta)$$

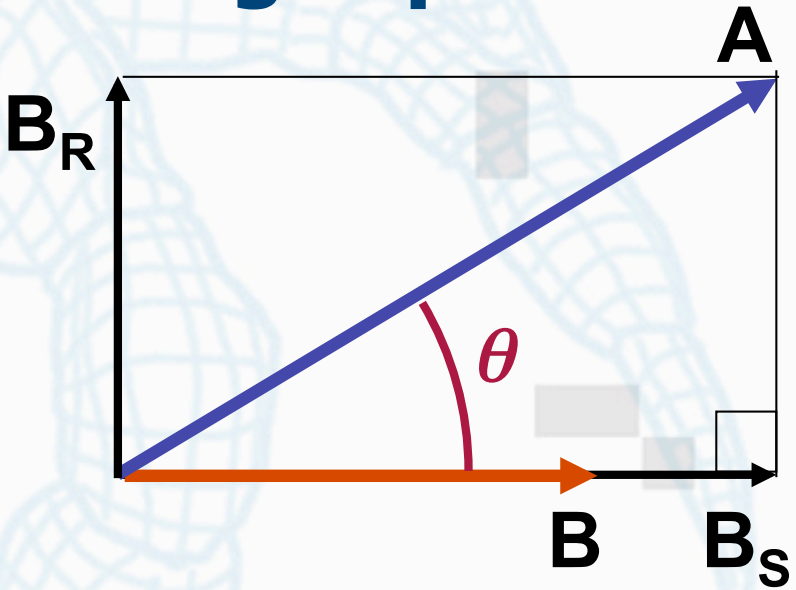


Quaternions as Four Coefficients

- **Let L be the Ratio of lengths between vectors A and B** $L = \frac{\|\mathbf{A}\|}{\|\mathbf{B}\|}$
- **Thus, the Tensor of the Right part**

$$R = \frac{\|\mathbf{B}_\mathbf{R}\|}{\|\mathbf{B}\|} = \frac{\|\mathbf{A}\|\sin(\theta)}{\|\mathbf{B}\|}$$

$$R = L\sin(\theta)$$



Quaternions as Four Coefficients

- **Then, the Quaternion Q can be written as**

  Q = x i +y j +z k +w

$$w = L\cos(\theta)$$

$$\sqrt{x^2 + y^2 + z^2} = L\sin(\theta)$$

- **w (real) is Scalar part**
- **(x i +y j +z k) is the Right part**

Product of Quaternions

- **Two quaternions *Q*1 and *Q*2 are composed by**

$$Q_1 \diamond Q_2 = T(Q_1)U(Q_1) \diamond T(Q_2)U(Q_2)$$

- **That is equivalent to**

$$Q_1 \diamond Q_2 = T(Q_1)T(Q_2) \cdot U(Q_1) \diamond U(Q_2)$$

Product of Quaternions

- **Consider the composition**

  $Q = Q1 \lozenge Q2$

  - **Tensor $T(Q)=T(Q1)T(Q2)$**
  - **Versor $U(Q) = U(Q1) \lozenge U(Q2)$**

Representation by four coefficients

- Let the representation of *P* and *Q* be

$$P = x_p\,\mathbf{i} + y_p\,\mathbf{j} + z_p\,\mathbf{k} + w_p$$

$$Q = x_q\,\mathbf{i} + y_q\,\mathbf{j} + z_q\,\mathbf{k} + w_q$$

- Their composition P ◊ Q is

$$P \diamond Q = \mathbf{L}(P)Q = \begin{bmatrix} w_p & -z_p & y_p & x_p \\ z_p & w_p & -x_p & y_p \\ -y_p & x_p & w_p & z_p \\ -x_p & -y_p & -z_p & w_p \end{bmatrix} \cdot \begin{bmatrix} x_q \\ y_q \\ z_q \\ w_q \end{bmatrix}$$

Representation by four coefficients

- Let the representation of *P* and *Q* be

$$P = x_p\,\mathbf{i} + y_p\,\mathbf{j} + z_p\,\mathbf{k} + w_p$$

$$Q = x_q\,\mathbf{i} + y_q\,\mathbf{j} + z_q\,\mathbf{k} + w_q$$

- Their composition P ◊ Q is

$$P \diamond Q = \mathbf{R}(Q)P = \begin{bmatrix} w_q & z_q & -y_q & x_q \\ -z_q & w_q & x_q & y_q \\ y_q & -x_q & w_q & z_q \\ -x_q & -y_q & -z_q & w_q \end{bmatrix} \cdot \begin{bmatrix} x_p \\ y_p \\ z_p \\ w_p \end{bmatrix}$$

Rotating a Vector (Finally !!)

- A Quaternion q = (x, y, z, w) rotates a Vector v by using the product

$$v' = q \lozenge v \lozenge q^{-1} = q \lozenge (v \lozenge q^{-1})$$

- Which can be reduced to a Matrix-Vector multiplication L(q) R(q^{-1})v

$$\begin{bmatrix} (w^2 + x^2 - y^2 - z^2) & (2xy - 2wz) & 2xz + 2wy & 0 \\ (2xy + 2wz) & (w^2 - x^2 + y^2 - z^2) & (2yz - 2wx) & 0 \\ (2xz - 2wy) & (2yz + 2wx) & (w^2 - x^2 - y^2 + z^2) & 0 \\ 0 & 0 & 0 & (w^2 + x^2 + y^2 + z^2) \end{bmatrix}$$

Quaternion Algebra

- **Consider quaternions**

$$Q_0 = w_0 + x_0 i + y_0 j + z_0 k = (w_0, \mathbf{v}_0)$$
$$Q_1 = w_1 + x_1 i + y_1 j + z_1 k = (w_1, \mathbf{v}_1)$$

- **Sum and Subtraction of quaternions**

$$\begin{aligned} Q_0 \pm Q_1 &= (w_0 \pm w_1) + (x_0 \pm x_1) i + (y_0 \pm y_1) j + (z_0 \pm z_1) k \\ Q_0 \pm Q_1 &= (w_0 \pm w_1, \mathbf{v}_0 \pm \mathbf{v}_1) \end{aligned}$$

Quaternion Algebra

- **Multiplications of primitive elements**

$$i^2 = j^2 = k^2 = -1$$

$$ij = -ji = k$$

$$jk = -kj = i$$

$$ki = -ik = j$$

Quaternion Algebra

- **Multiplication of quaternions**

$$
\begin{aligned}
Q_0Q_1 &= \left(w_0 + x_0 i + y_0 j + z_0 k\right)\left(w_1 + x_1 i + y_1 j + z_1 k\right) \\
&= \left(w_0 w_1 - \left(x_0 x_1 + y_0 y_1 + z_0 z_1\right)\right) \\
&+ \left(w_0 x_1 + x_0 w_1 + y_0 z_1 - z_0 y_1\right) i \\
&+ \left(w_0 y_1 + y_0 w_1 + z_0 x_1 - x_0 z_1\right) j \\
&+ \left(w_0 z_1 + z_0 w_1 + z_0 x_1 - x_0 z_1\right) k \\
Q_0Q_1 &= \left(\left(w_0 w_1 - \mathbf{v}_0 \cdot \mathbf{v}_1\right), w_0 \mathbf{v}_1 + w_1 \mathbf{v}_0 + \mathbf{v}_0 \times \mathbf{v}_1\right)
\end{aligned}
$$

Quaternion Algebra

- **Conjugate of quaternions**

$$K(Q) = \mathrm{Q}^* = \left(\mathrm{w} + \mathrm{xi} + \mathrm{yj} + \mathrm{zk}\right)^*$$
$$= \left(\mathrm{w} - \mathrm{xi} - \mathrm{yj} - \mathrm{zk}\right)$$

$$K(K(Q)) = \left(\mathrm{Q}^*\right)^* = Q$$

$$K(Q_1 Q_2) = \left(Q_1 Q_2\right)^* = Q_2^* Q_1^*$$

Quaternion Algebra

- **Norm of quaternions**

$$N(Q) = N(\mathrm{w} + \mathrm{xi} + \mathrm{yj} + \mathrm{zk})$$

$$= \mathrm{w}^2 + \mathrm{x}^2 + \mathrm{y}^2 + \mathrm{z}^2$$

$$N(K(Q)) = N(\mathrm{Q}^*) = N(Q)$$

$$N(Q_1 Q_2) = N(Q_1) N(Q_2)$$

Quaternion Algebra

- **The multiplicative inverse, Q^{-1}**

$$QQ^{-1} = Q^{-1}Q = 1$$

$$Q^{-1} = \frac{Q^*}{N(Q)}$$

$$\left(Q^{-1}\right)^{-1} = Q$$

$$\left(Q_1 Q_2\right)^{-1} = Q_2^{-1} Q_1^{-1}$$

Quaternion Algebra

- **The Selection function, W(Q)**

$$W(Q) = W\left(w + xi + yj + zk\right) = w$$

$$W(Q) = \left(Q + Q^{*}\right)/2$$

- **Dot product of two quaternions**

$$Q_0 \cdot Q_1 = w_0 w_1 + x_0 x_1 + y_0 y_1 + z_0 z_1$$

$$= W\left(Q_0 Q_1^{*}\right)$$

Quaternion Algebra

- **Unit quaternion has N(Q)=1**
- **Represented by**

$$Q = \cos(\theta) + \mathbf{u}\sin(\theta)$$

$$\mathbf{u} = u_x i + u_y j + u_z k$$

$$\left\| \left(u_x, u_y, u_z \right) \right\| = 1$$

$$\mathbf{uu} = -1 \quad \text{- Quaternion product}$$

Quaternion Algebra

- **Extension of Euler's identity**

$$\exp(\mathbf{u}\theta) = \cos(\theta) + \mathbf{u}\sin(\theta)$$

- **Power of a unit quaternion**

$$Q^t = (\cos(\theta) + \mathbf{u}\sin(\theta))^t = \exp(\mathbf{u}t\theta)$$

$$Q^t = \cos(t\theta) + \mathbf{u}\sin(t\theta)$$

Quaternion Algebra

- **Extension of Euler's identity**

$$\exp(\mathbf{u}\theta) = \cos(\theta) + \mathbf{u}\sin(\theta)$$

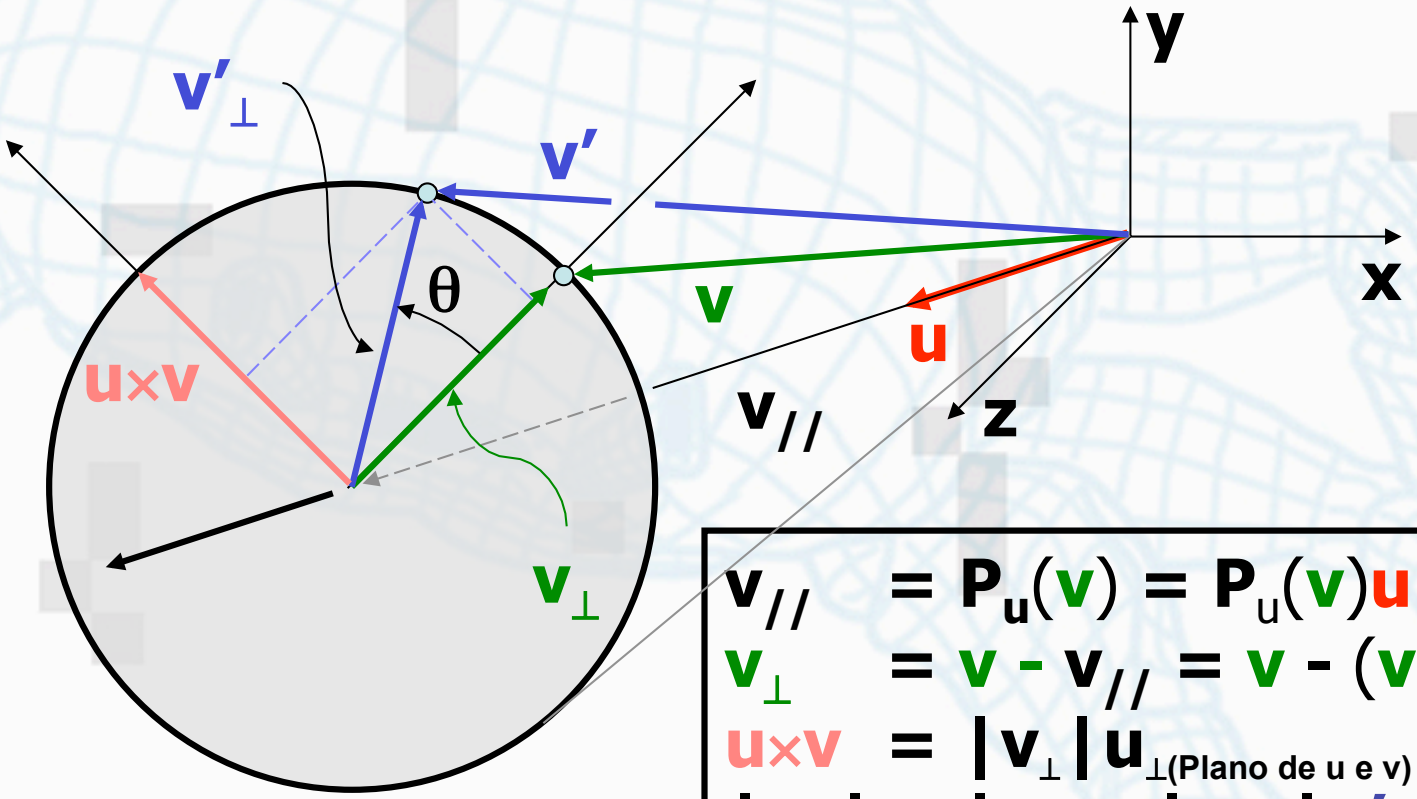
- **Logarithm of a unit quaternion**

$$\log(Q) = \log(\cos(\theta) + \mathbf{u}\sin(\theta))$$

$$\log(Q) = \log(\exp(\mathbf{u}\theta)) = \mathbf{u}\theta$$

- **Non-commutativity of quaternion multiplication disallows log and exp identities**

Rotation around an axis



Plane perpendicular to the direction of u

$$
\begin{aligned}
\mathbf{v}_{//} &= P_u(\mathbf{v}) = P_u(\mathbf{v})\mathbf{u} = (\mathbf{v}.\mathbf{u})\mathbf{u} \\
\mathbf{v}_{\perp} &= \mathbf{v} - \mathbf{v}_{//} = \mathbf{v} - (\mathbf{v}.\mathbf{u})\mathbf{u} \\
\mathbf{u}\times\mathbf{v} &= |\mathbf{v}_{\perp}|\,\mathbf{u}_{\perp(\text{Plano de u e v})} \\
|\mathbf{v}_{\perp}| &= |\,\mathbf{u}\times\mathbf{v}\,| = |\mathbf{v}'_{\perp}| \\
\mathbf{v}'_{\perp} &= |\mathbf{v}'_{\perp}|\cos\theta\,\mathbf{u}_{\mathbf{v}\perp} + |\mathbf{v}'_{\perp}|\,\text{sen}\,\theta\,\mathbf{u}_{\mathbf{u}\times\mathbf{v}} \\
&= (\mathbf{v} - (\mathbf{v}.\mathbf{u})\mathbf{u})\cos\theta + \mathbf{u}\times\mathbf{v}\,\text{sen}\,\theta
\end{aligned}
$$

Rotation around an axis

- **Decomposition of Rotated vector**

$$\mathbf{v}' = \mathbf{v}_{//} + \mathbf{v}'_{\perp}$$

$$= (\mathbf{v}\cdot\mathbf{u})\mathbf{u} + (\mathbf{v} - (\mathbf{v}\cdot\mathbf{u})\mathbf{u})\cos(\theta) + (\mathbf{u}\times\mathbf{v})\sin(\theta)$$

$$= \mathbf{v}\cos(\theta) + ((\mathbf{v}\cdot\mathbf{u})\mathbf{u})(1-\cos(\theta)) + (\mathbf{u}\times\mathbf{v})\sin(\theta)$$

Rotation around an axis

- **Rotation using a unit quaternion**

  $Q=\cos(\theta)+u\sin(\theta)$

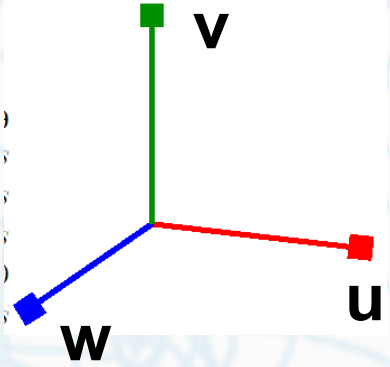
- **Vector v is rotated of 2θ about the axis defined by the unit vector u**

$$\mathbf{v}' = R(\mathbf{v}) = Q\mathbf{v}Q^{*}$$

Rotation around an axis

- **Conditions to prove**
  - **v′ is a 3D vector**
    W(v′) = 0
  - **R is a length preserving operator**
    N(v′) = N(v)
  - **R is a linear operator**
    R(αv + w) = αR(v) + R(w)
  - **R(u) = u**

Rotation around an axis



- u, v, w are orthonormal 3D vectors
- Suppose v is rotated by ϕ with the unit quaternion of axis u

$v' = qvq^* \rightarrow v.v' = \cos(\phi)$

Rotation around an axis

$$\cos(\phi) = \mathbf{v} \cdot (q\mathbf{v}q^*) = W(\mathbf{v}^* q\mathbf{v}q^*)$$

$$= W\left[-\mathbf{v}(\cos(\theta) + \mathbf{u}\sin(\theta))\mathbf{v}(\cos(\theta) - \mathbf{u}\sin(\theta))\right]$$

$$= W\left[(-\mathbf{v}\cos(\theta) - \mathbf{vu}\sin(\theta))(\mathbf{v}\cos(\theta) - \mathbf{vu}\sin(\theta))\right]$$

$$= W\begin{bmatrix} -\mathbf{v}^2\cos(\theta)^2 + \mathbf{v}^2\mathbf{u}\sin(\theta)\cos(\theta) - \\ -\mathbf{vuv}\sin(\theta)\cos(\theta) + (\mathbf{vu})^2\sin(\theta)^2 \end{bmatrix}$$

$$= W\left[\cos(\theta)^2 - \sin(\theta)^2 - (\mathbf{u} + \mathbf{vuv})\sin(\theta)\cos(\theta)\right]$$

Rotation around an axis

$$\mathbf{vu} = -\mathbf{w}$$

$$\mathbf{vuv} = -\mathbf{wv} = -(-\mathbf{u}) = \mathbf{u}$$

$$\cos(\phi) = W\lfloor\cos(\theta)^2 - \sin(\theta)^2 - 2\mathbf{u}\sin(\theta)\cos(\theta)\_$$

$$= \cos(\theta)^2 - \sin(\theta)^2 = \cos(2\theta)$$

- **Therefore $\phi = 2\theta$**

Fim da Aula 15