Centro de Instrução Almirante Wandenkolk - CIAW
Instituto Tecnológico de Aeronáutica - ITA

Curso de Aperfeiçoamento Avançado em Sistemas de Armas

SAB: Simulação e Controle de Artefatos Bélicos

Revisão Vetorial, Matricial e Geometria Analítica

Jozias **Del Rios** Cap Eng delriosjdrvgs@fab.mil.br (12) 98177-9921

Abril 2018

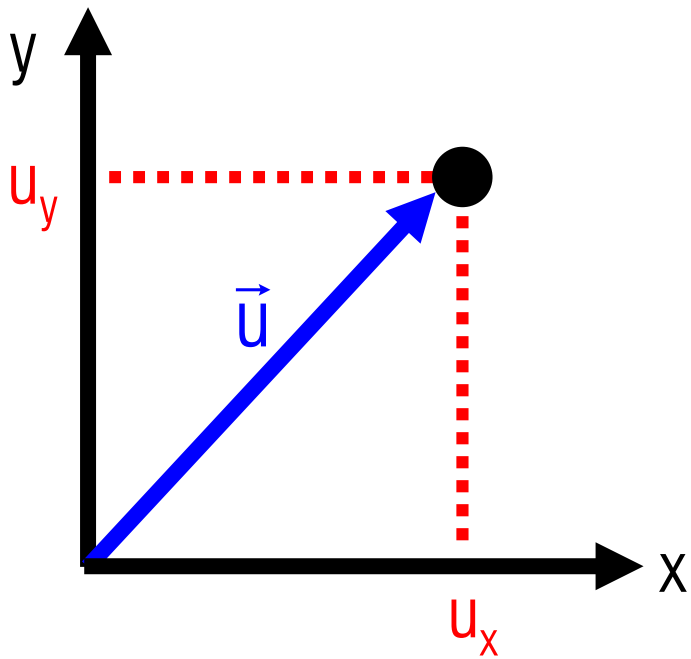
SIMULAÇÃO E CONTROLE DE ARTEFATOS BÉLICOS

Revisão Vetorial, Matricial e Geometria Analítica

Autor do Material: Jozias **DEL RIOS** – rev. 19.dez.2017

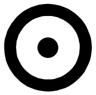
VETOR

Constituído de <u>Direção</u>, <u>Sentido</u> e <u>Módulo</u>



Escrito por coordenadas:

$$\vec{u} = \left(u_x, u_y\right)$$

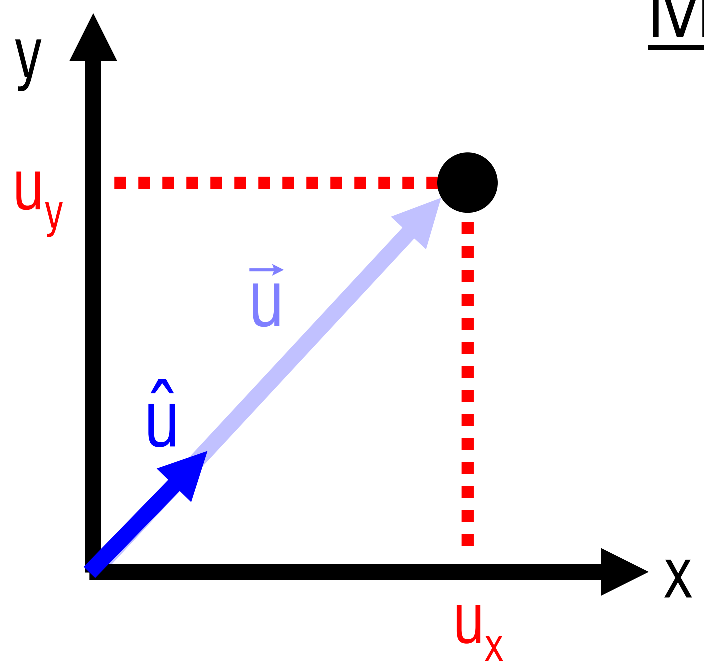

Vetor saindo da tela


Vetor entrando da tela

VERSOR

Constituído de Direção e Sentido.

Módulo unitário.



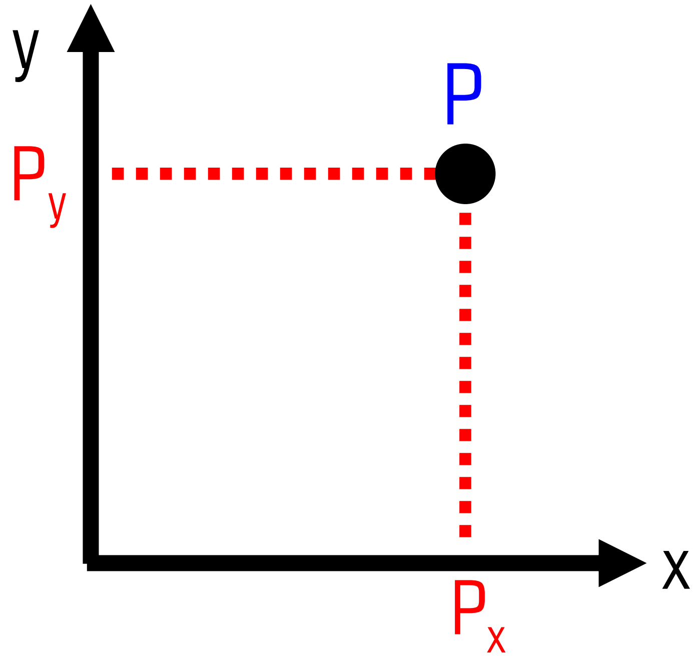
Cálculo:

$$\hat{u} = \frac{\vec{u}}{\|\vec{u}\|} = \frac{1}{\sqrt{u_x^2 + u_y^2}} \cdot \vec{u}$$

```
% MATLAB:
function d = vecdir(u)
  d = u/norm(u);
end
```

PONTO

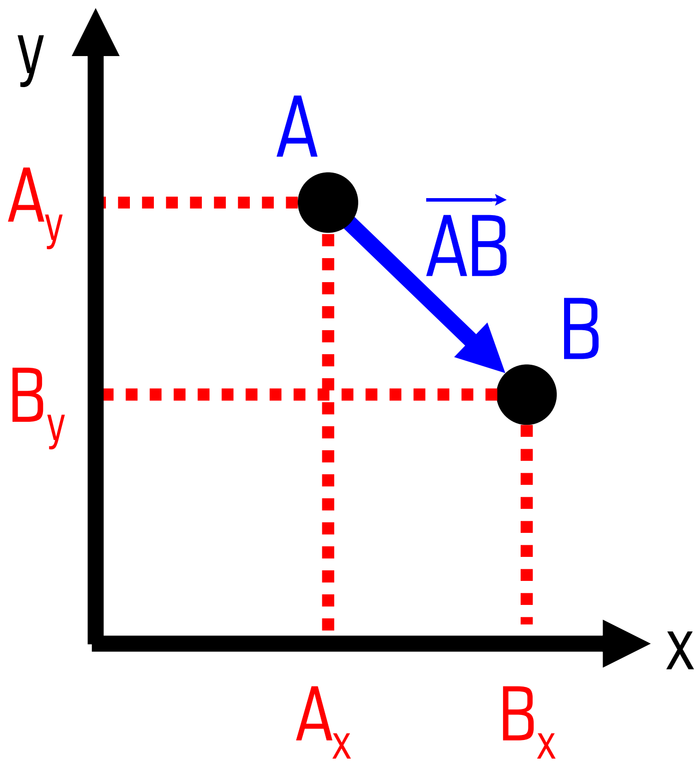
Representa um local no espaço



Escrito por coordenadas:

$$P = \left(P_x, P_y\right)$$

DESLOCAMENTO



Vetor que liga dois pontos

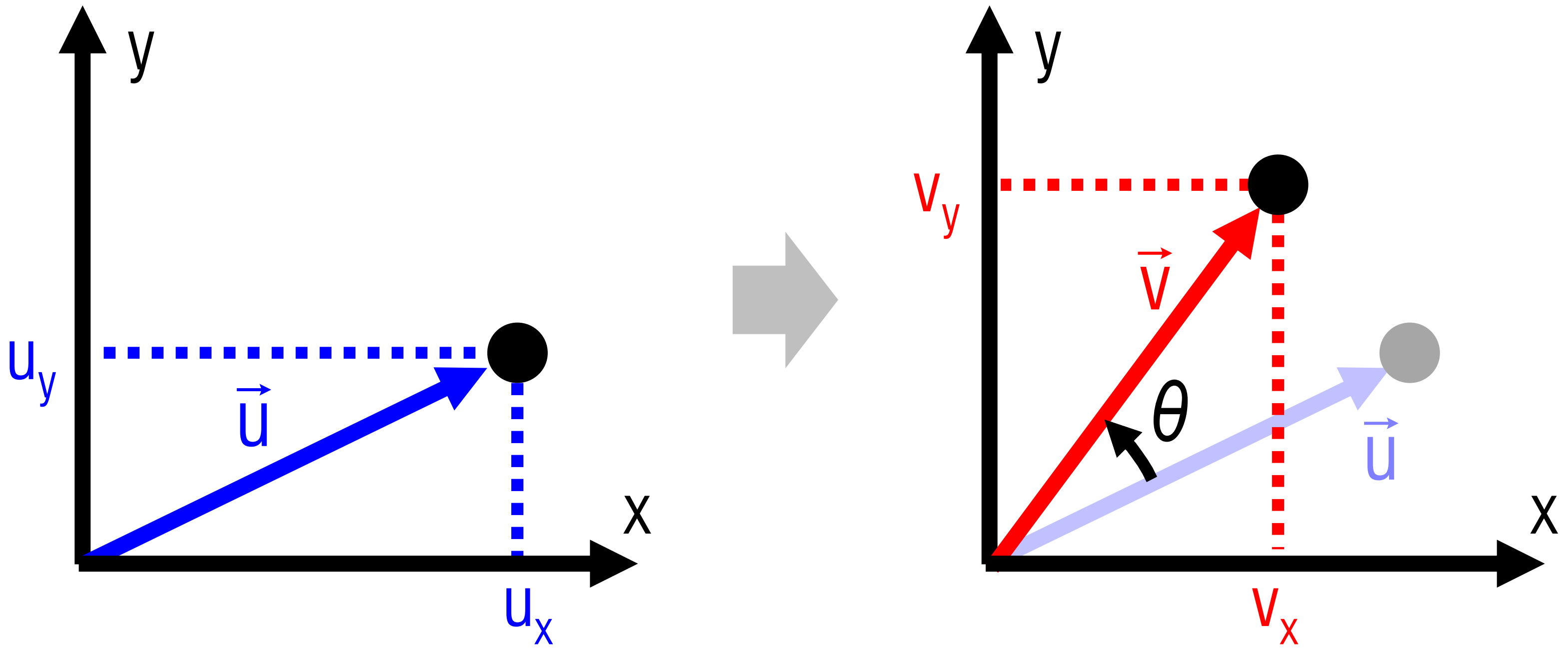
$$\overrightarrow{AB} = B - A$$

Escrito por coordenadas:

$$\overrightarrow{AB} = \left(B_x - A_x,\ B_y - A_y\right)$$

ROTAÇÃO DE VETOR NO PLANO

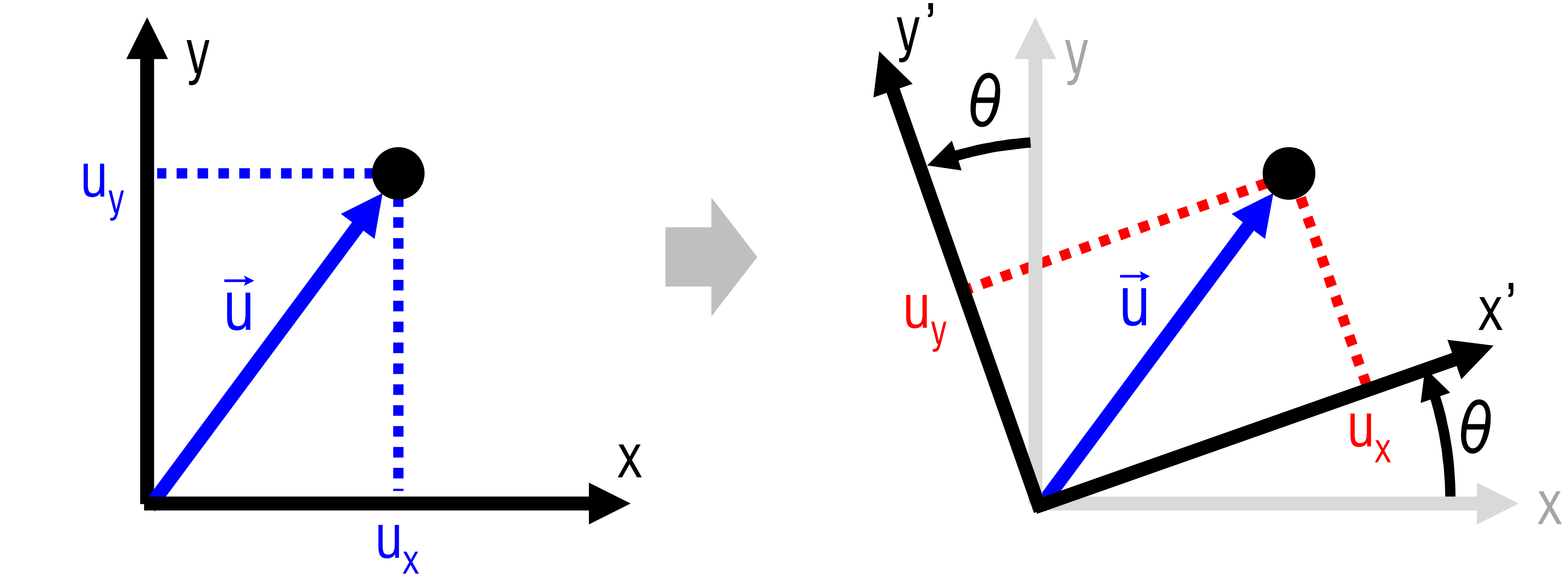
Rotação de um vetor ou ponto por um ângulo θ no plano



$$\begin{cases} v_x = u_x \cos\theta - u_y \operatorname{sen}\theta \\ v_y = u_x \operatorname{sen}\theta + u_y \cos\theta \end{cases} \Leftrightarrow \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\operatorname{sen}\theta \\ \operatorname{sen}\theta & \cos\theta \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

ROTAÇÃO DE EIXOS NO PLANO

Rotação dos eixos cartesianos por um ângulo θ

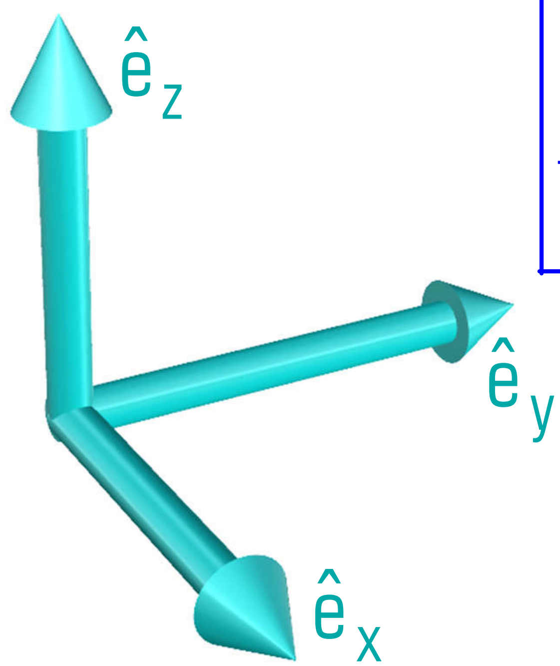


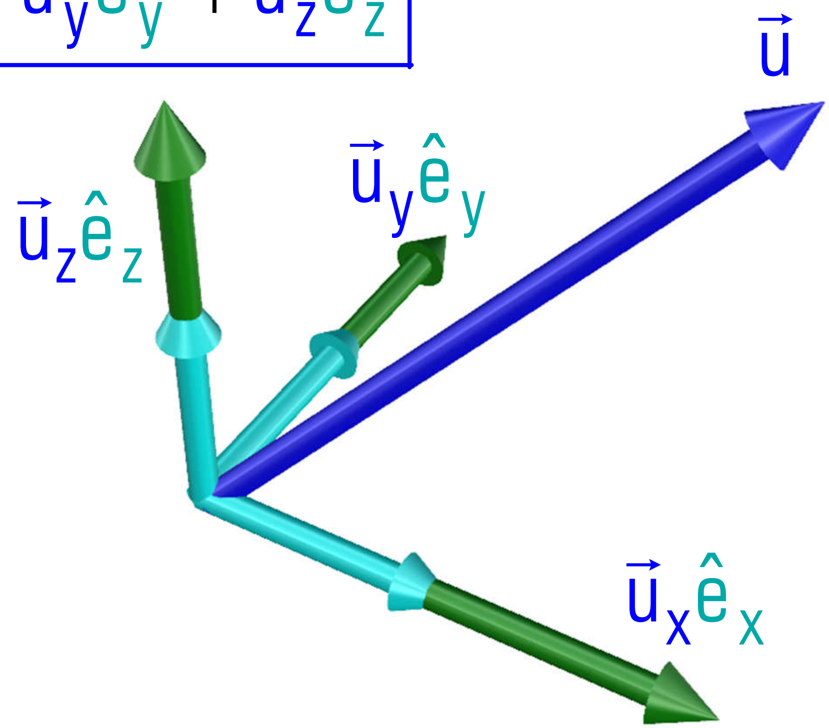
$$\begin{cases} u_x = u_x \cos\theta + u_y \operatorname{sen}\theta \\ u_y = -u_x \operatorname{sen}\theta + u_y \cos\theta \end{cases} \Leftrightarrow \begin{bmatrix} u_x \\ u_y \end{bmatrix} = \begin{bmatrix} \cos\theta & \operatorname{sen}\theta \\ -\operatorname{sen}\theta & \cos\theta \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}$$

BASE ORTONORMAL

Base de 3 versores normais (perpendiculares) entre si

$$\boxed{\begin{aligned}\vec{u} &= \left(u_x, u_y, u_z\right)\\ \vec{u} &= u_x\hat{e}_x + u_y\hat{e}_y + u_z\hat{e}_z\end{aligned}}$$





$$\boxed{\hat{e}_x \times \hat{e}_y = \hat{e}_z}$$

$$u_x = \vec{u} \cdot \hat{e}_x$$

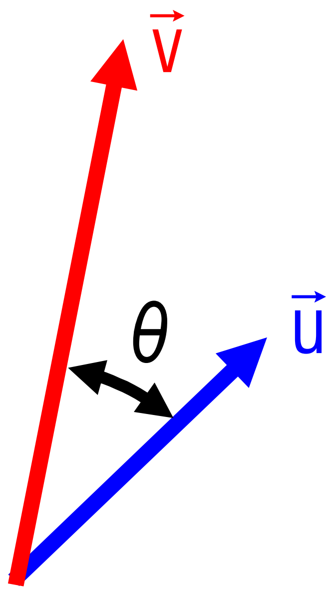
PRODUTO ESCALAR ou PRODUTO INTERNO

$$\vec{u} = \left(u_x, u_y, u_z\right) \qquad \vec{v} = \left(v_x, v_y, v_z\right)$$

$$\boxed{\vec{u} \cdot \vec{v} = u_x v_x + u_y v_y + u_z v_z}$$

$$\vec{u} \cdot \vec{v} = u\,v\cos\theta$$

```
% MATLAB: dot(u,v)
```



PRODUTO VETORIAL ou PRODUTO EXTERNO

$$\vec{u} = u_x \hat{e}_x + u_y \hat{e}_y + u_z \hat{e}_z \qquad \vec{v} = v_x \hat{e}_x + v_y \hat{e}_y + v_z \hat{e}_z$$

$$\boxed{\vec{u} \times \vec{v} = \hat{e}_x \left(u_y v_z - u_z v_y\right) + \hat{e}_y \left(u_z v_x - u_x v_z\right) + \hat{e}_z \left(u_x v_y - u_y v_x\right)}$$

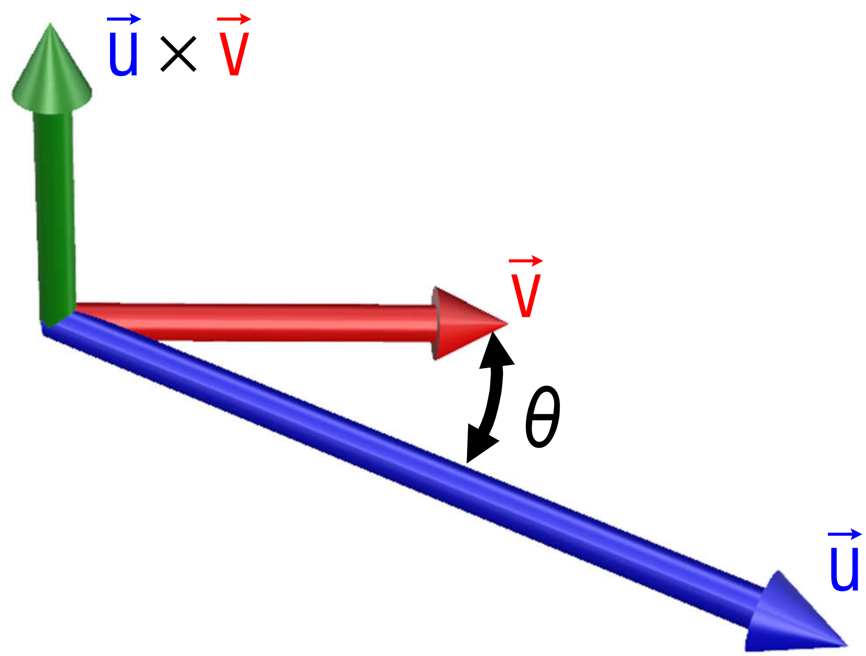
$$\vec{u} \times \vec{v} = \det \begin{pmatrix} \hat{e}_x & \hat{e}_y & \hat{e}_z \\ u_x & u_y & u_z \\ v_x & v_y & v_z \end{pmatrix}$$

$$\left\|\vec{u} \times \vec{v}\right\| = u\,v\,\mathrm{sen}\,\theta$$

Direção: regra da mão direita

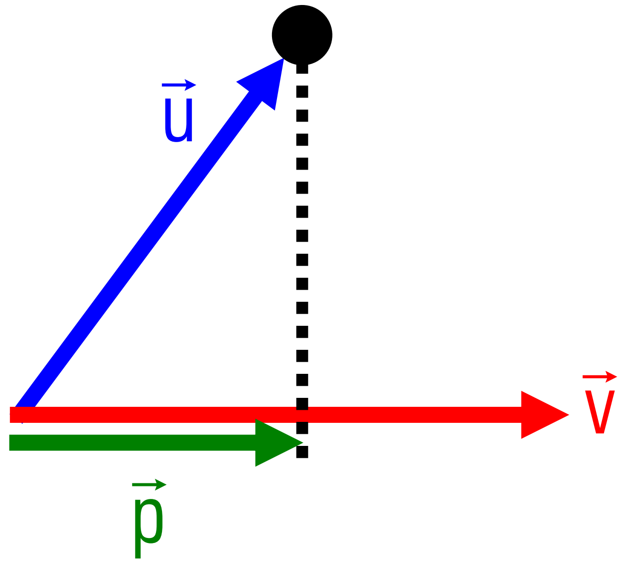
$\vec{u} \times \vec{v}$ é normal tanto a $\vec{u}$ quanto a $\vec{v}$

```
% MATLAB: cross(u,v)
```



PROJEÇÃO

$$\vec{p} = \text{proj}_{\vec{v}}\, \vec{u} \;=\; \hat{v}\left(\vec{u} \cdot \hat{v}\right)$$



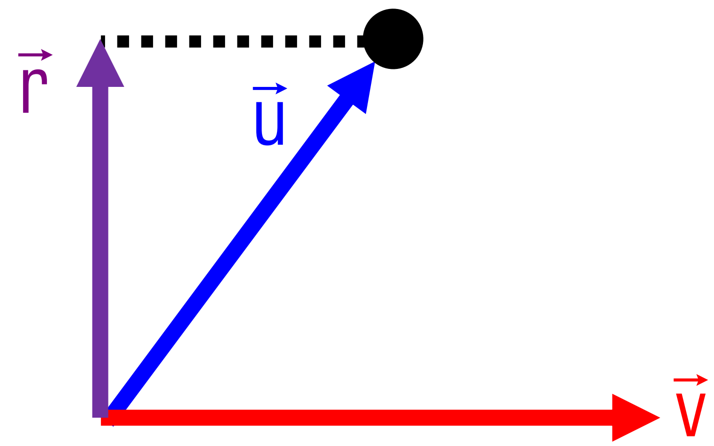
Vetor **p** é a projeção do vetor **u** na direção do vetor **v**

Logo, o vetor **p** é alinhado com o vetor **v**

```
% MATLAB:
p = vecdir(v) * dot(u, vecdir(v));
```

REJEIÇÃO

$$\vec{r} = \mathrm{rej}_{\vec{v}}\,\vec{u} = \hat{v} \times \vec{u} \times \hat{v}$$



Vetor **r** é a <u>rejeição</u> do vetor **u** na direção do vetor **v**

Logo, o vetor **r** é <u>perpendicular</u> ao vetor **v**

Reconstrução: $\vec{p} + \vec{r} = \mathrm{proj}_{\vec{v}}\,\vec{u} + \mathrm{rej}_{\vec{v}}\,\vec{u} = \vec{u}$

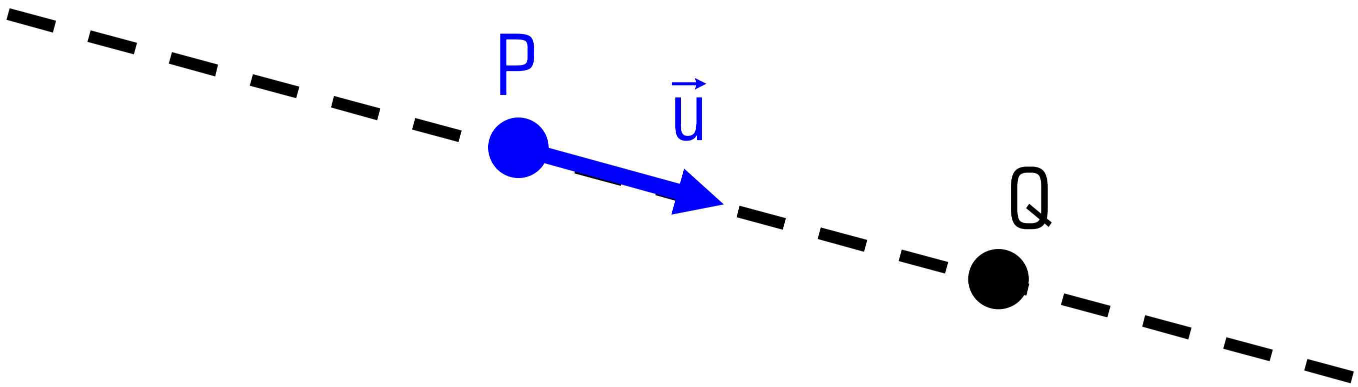
```
% MATLAB:
r = cross( vecdir(v), cross(u, vecdir(v)) );
```

FUNÇÕES MATLAB de PROJEÇÃO e REJEIÇÃO

```
% MATLAB: função de projeção de u em v
function p = vecproj(u, v)
   vn = vecdir(v);
   p = vn * dot(u, vn);
end

% MATLAB: função de rejeição de u em v
function r = vecrej(u, v)
   vn = vecdir(v);
   r = cross(vn, cross(u, vn) );
end
```

RETA



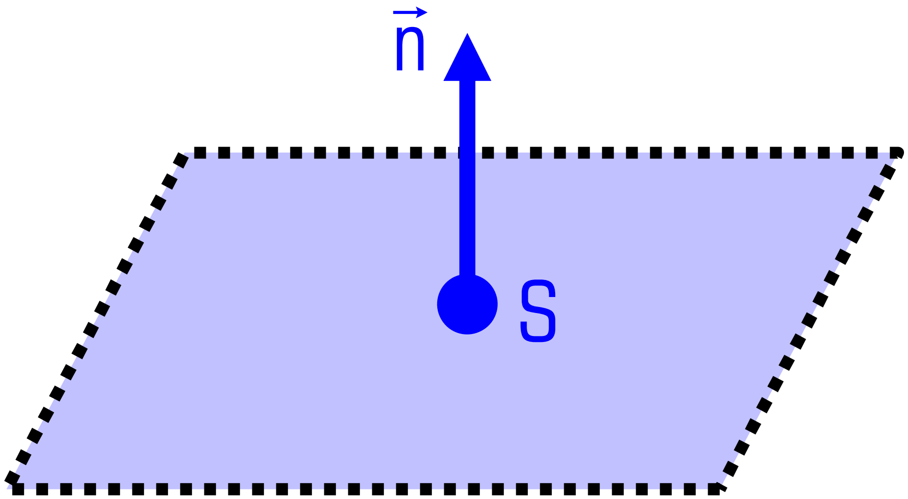
Definida por um ponto **P** qualquer na reta

e um vetor de direção **u**.

O escalar **λ** gera qualquer outro ponto **Q** na reta:

$$Q = P + \lambda \vec{u}$$

PLANO

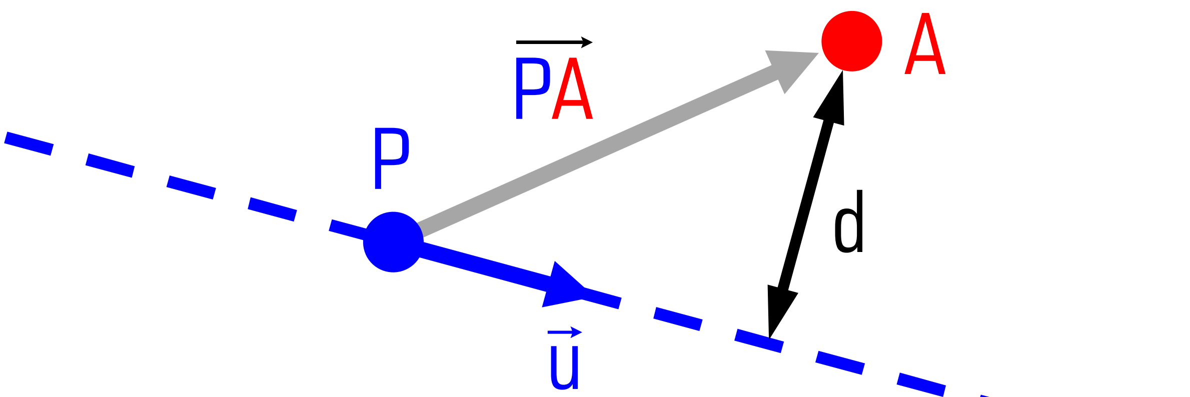


Definido por um ponto **S** qualquer contido no plano e um vetor **n** normal ao plano.

Qualquer ponto **P** contido no plano obedece a relação:

$$\vec{n} \cdot \overrightarrow{SP} = 0 \quad \Leftrightarrow \quad \vec{n} \cdot (S - P) = 0$$

DISTÂNCIA PONTO–RETA

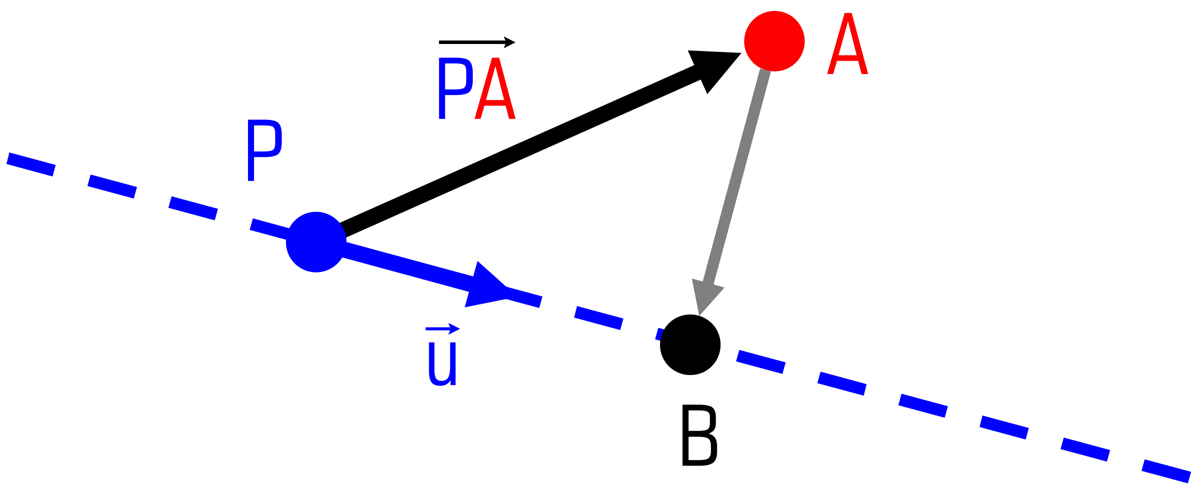


$$d = \left\| \overrightarrow{PA} \times \hat{u} \right\|$$

$$d = \left\| \mathrm{rej}_{\vec{u}}\, \overrightarrow{PA} \right\|$$

```
% MATLAB:
d = abs( cross( A - P, vecdir(u) ) );
```

PROJEÇÃO de PONTO na RETA

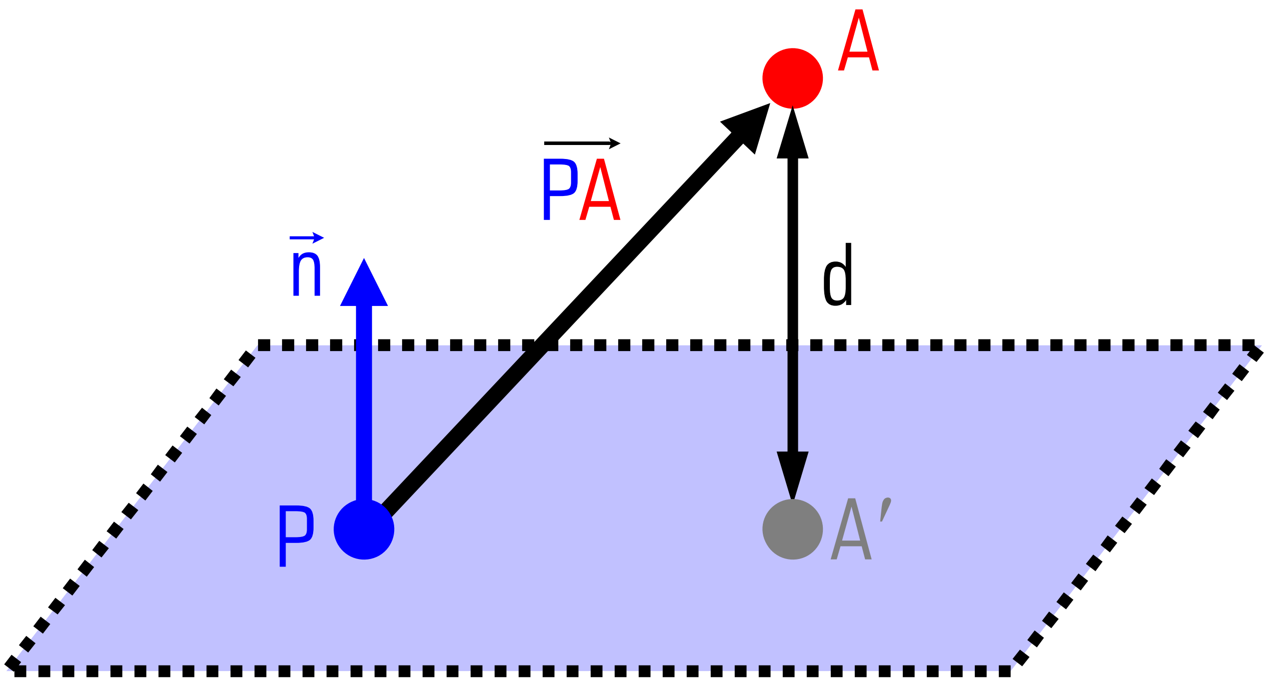


$$B = A - \mathrm{rej}_{\vec{u}}\,\overrightarrow{PA}$$

```
% MATLAB:
B = A - vecrej( A - P, u );
```

$$B = A - \hat{u} \times \overrightarrow{PA} \times \hat{u}$$

DISTÂNCIA PONTO–PLANO

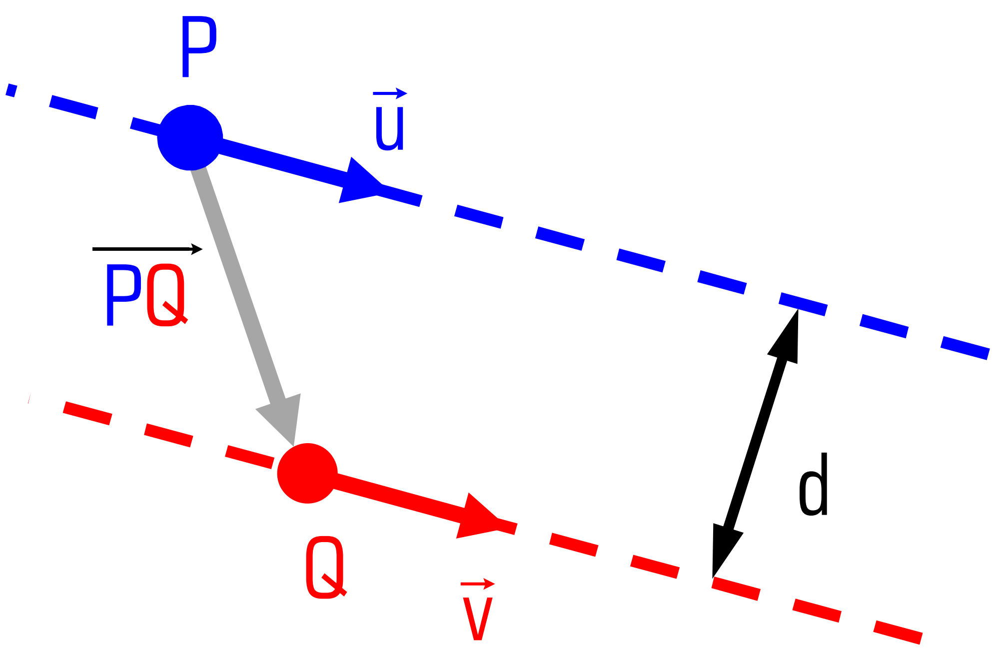


$$d = \overrightarrow{PA} \cdot \hat{n}$$

Qualquer que seja o ponto **P** no plano.

```
% MATLAB:
d = dot( A - P, vecdir(n) )
```

DISTÂNCIA entre RETAS PARALELAS



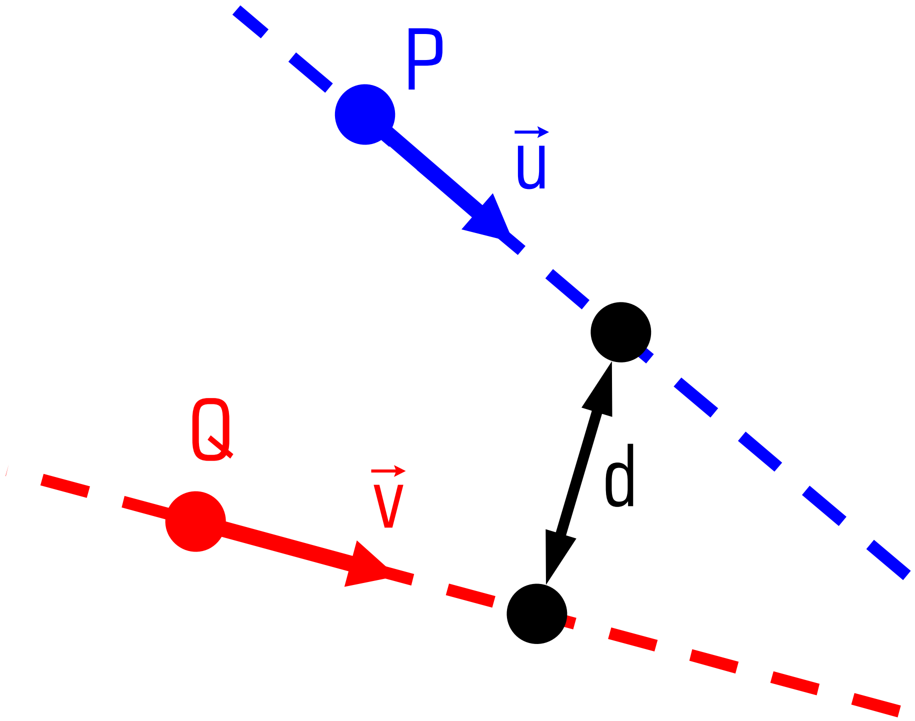
$$\vec{u} = \vec{v}$$

$$\vec{u} \times \vec{v} = 0$$

$$d = \left\| \overrightarrow{PQ} \times \hat{u} \right\|$$

```
% MATLAB:
d = abs( cross(Q-P, vecdir(u)) );
```

DISTÂNCIA RETA–RETA

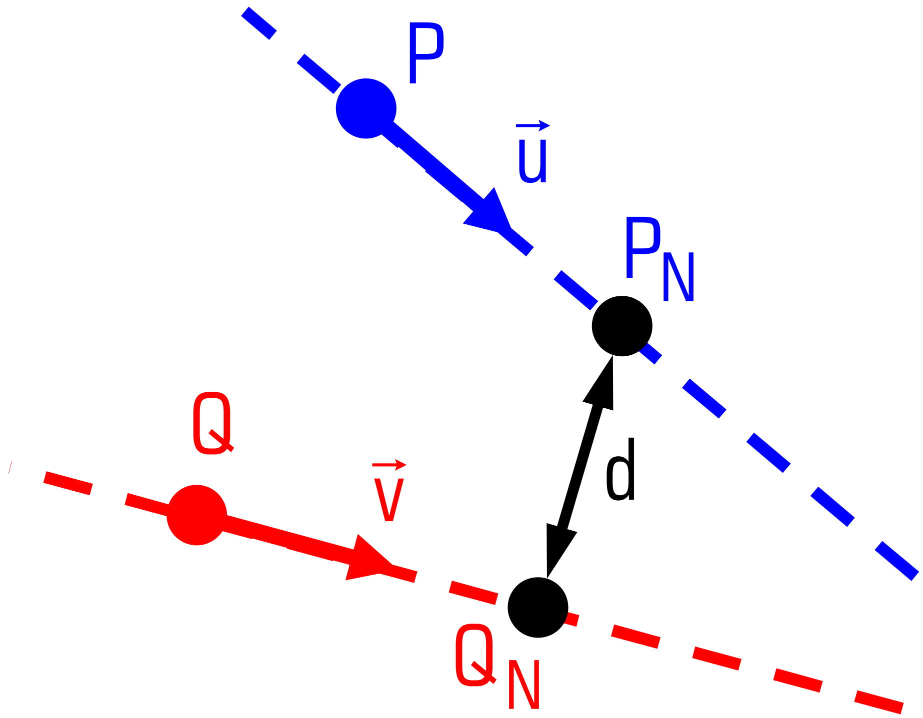


$$\vec{n} = \vec{u} \times \vec{v}$$

$$d = \left\| \overrightarrow{PQ} \cdot \hat{n} \right\|$$

```
% MATLAB:
d = abs( dot( Q-P, vecdir(cross(u,v)) ) );
```

PONTOS MAIS PRÓXIMOS ENTRE RETAS



$$\vec{n} = \vec{u} \times \vec{v}$$

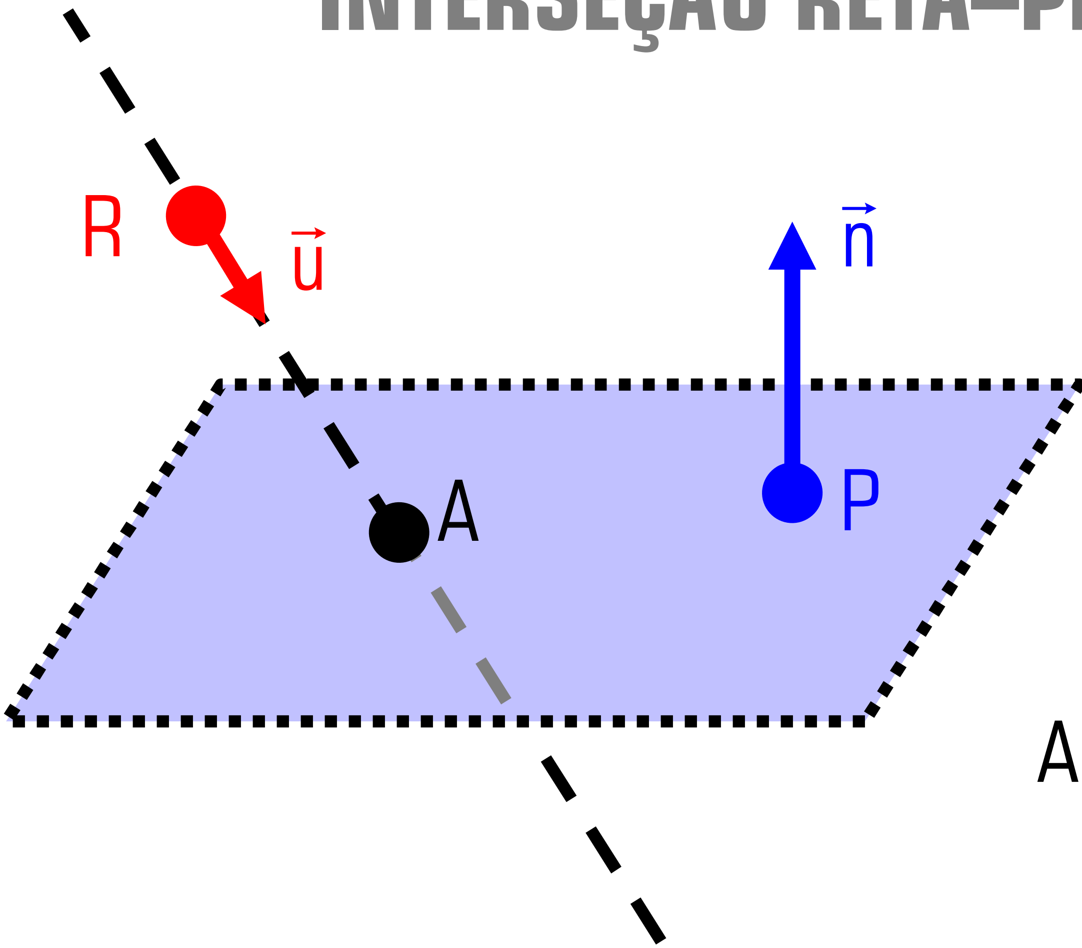
$$\vec{n}_u = \vec{u} \times \vec{n}$$

$$\vec{n}_v = \vec{v} \times \vec{n}$$

$$P_N = P + \vec{u}\,\frac{\overrightarrow{PQ} \cdot \vec{n}_v}{\vec{u} \cdot \vec{n}_v}$$

$$Q_N = Q + \vec{v}\,\frac{\overrightarrow{PQ} \cdot \vec{n}_u}{\vec{v} \cdot \vec{n}_u}$$

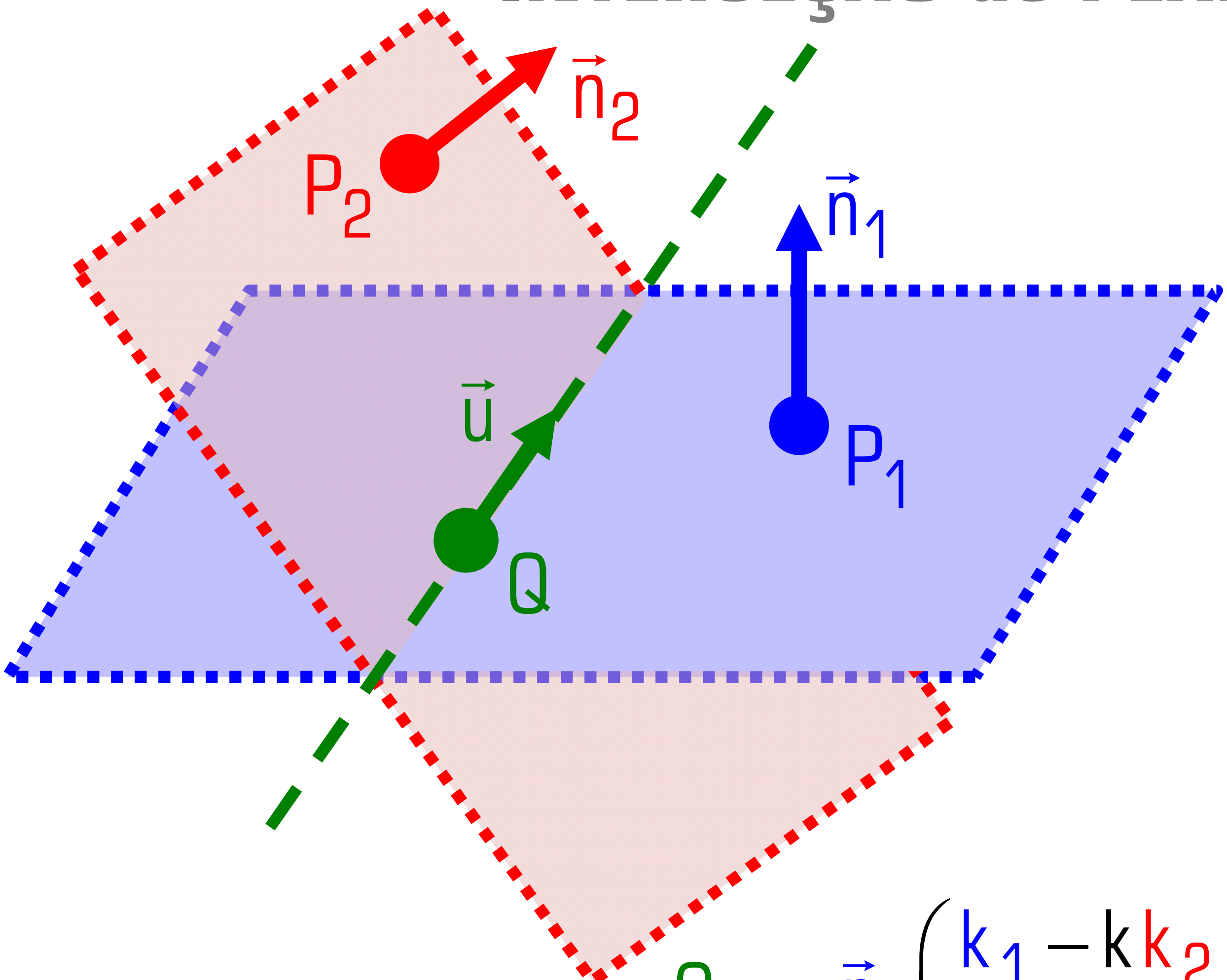
INTERSEÇAO RETA–PLANO



$$A = R + \vec{u} \cdot \frac{\overrightarrow{RP} \cdot \vec{n}}{\vec{u} \cdot \vec{n}}$$

```
% MATLAB:
A = R + u*dot(R-P, n) / dot(u, n)
```

INTERSEÇAO de PLANOS



$$\boxed{\vec{u} = \vec{n}_1 \times \vec{n}_2}$$

$$k = \vec{n}_1 \cdot \vec{n}_2$$

$$k_1 = P_1 \cdot \vec{n}_1$$

$$k_2 = P_2 \cdot \vec{n}_2$$

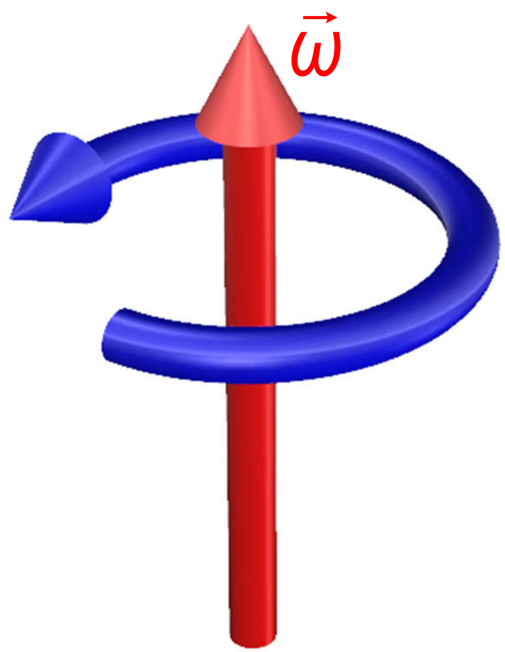
$$Q = \vec{n}_1 \left(\frac{k_1 - k k_2}{1 - k^2} \right) + \vec{n}_2 \left(\frac{k_2 - k k_1}{1 - k^2} \right)$$

VELOCIDADE ANGULAR

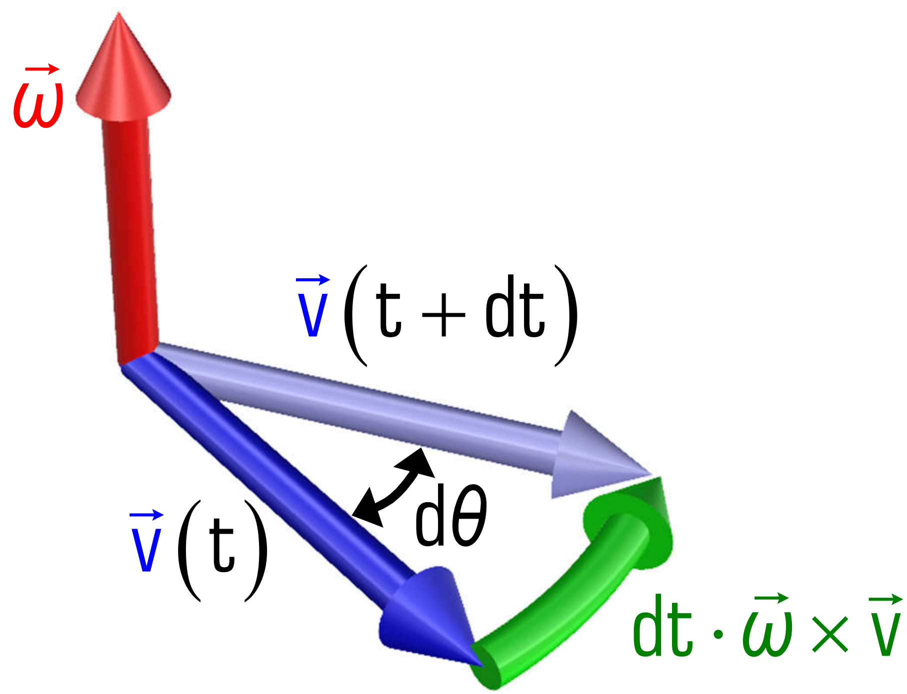
<u>Vetor</u>:

- Direção é um eixo.
- Módulo é uma velocidade de rotação angular (rad/s).

<u>Aplicar a regra da mão direita</u>



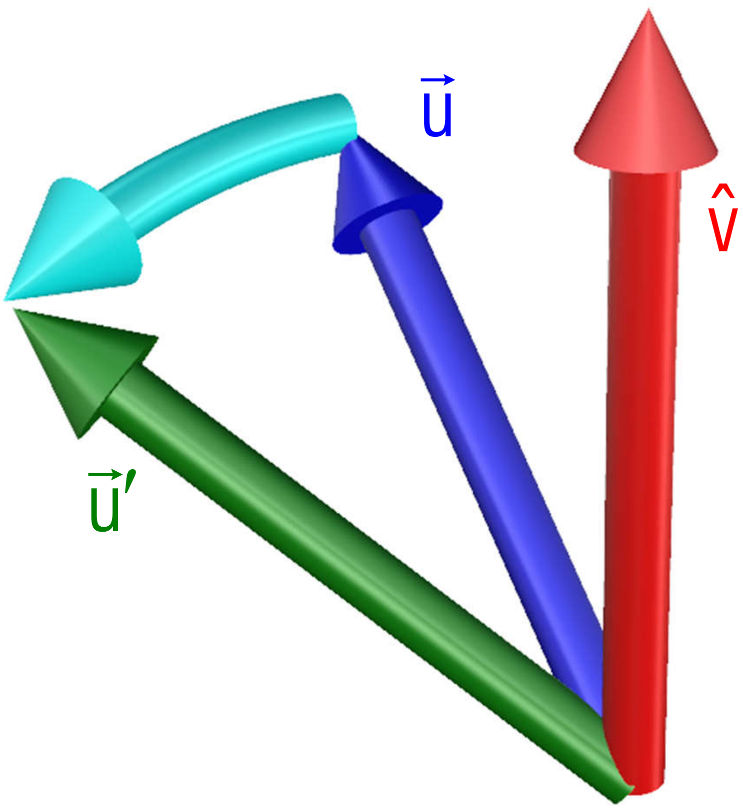
$$d\theta = \omega \, dt$$

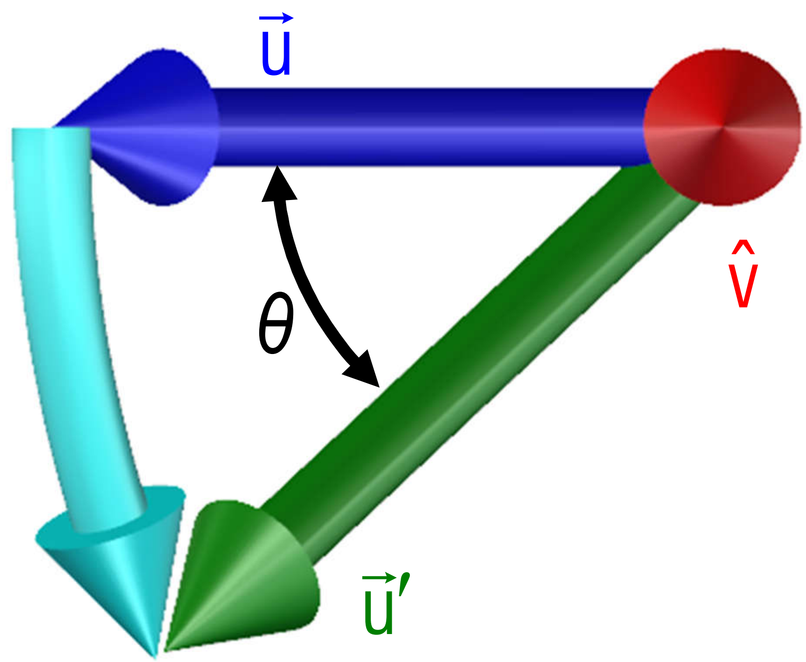


FÓRMULA DE ROTAÇÃO DE RODRIGUES

Rotação de um vetor $\vec{u}$ por um ângulo θ ao redor do eixo $\hat{v}$

$$\boxed{\vec{u}' = \vec{u}\cos\theta + (\hat{v}\times\vec{u})\,\mathrm{sen}\,\theta + \hat{v}(\vec{u}\cdot\hat{v})(1-\cos\theta)}$$





FÓRMULA DE ROTAÇÃO DE RODRIGUES

$$\boxed{\vec{u}' = \vec{u}\cos\theta + \left(\hat{v}\times\vec{u}\right)\operatorname{sen}\theta + \hat{v}\left(\vec{u}\cdot\hat{v}\right)\left(1-\cos\theta\right)}$$

```
% MATLAB:
% Rotação do vetor u por um ângulo q ao redor do eixo vetor v

function ur = vecrot(u, v, q)
  vn = vecdir(v);
  ur = u * cos(q) + ...
       cross(vn,u) * sin(q) + ...
       vn * dot(u,vn) * (1-cos(q));
end
```

FRD: FORWARD, RIGHT, DOWN

Base ortonormal fixada ao corpo (veículo)

$\mathbf{b}_x$ = forward

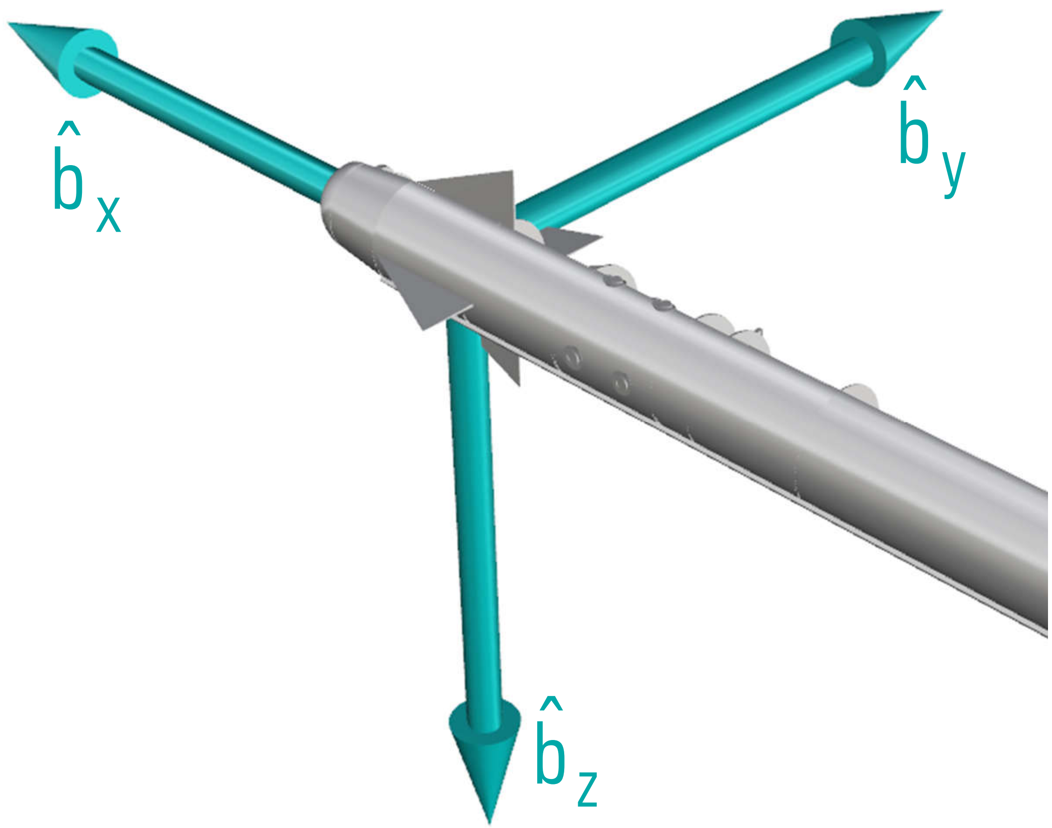
$\mathbf{b}_y$ = right

$\mathbf{b}_z$ = down

$\mathbf{b}_x = (1, 0, 0)$

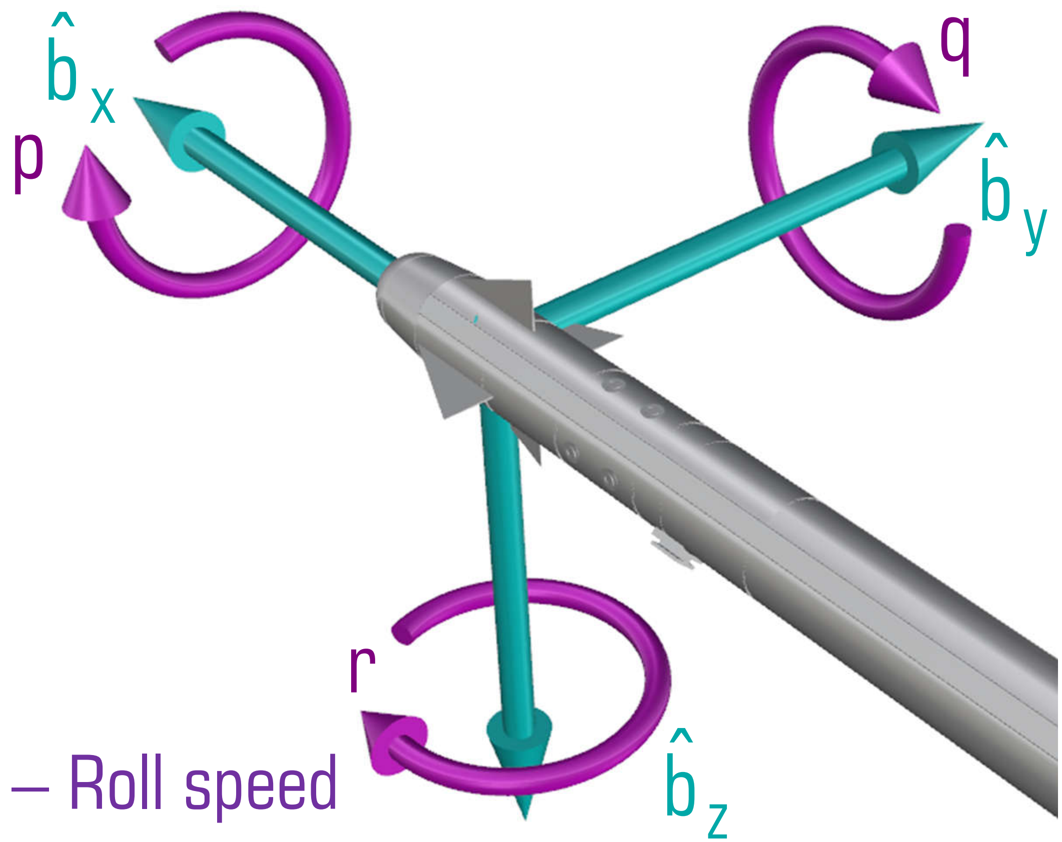
$\mathbf{b}_y = (0, 1, 0)$

$\mathbf{b}_z = (0, 0, 1)$



VELOCIDADES de ARFAGEM, GUINADA e ROLAMENTO

Velocidades angulares em que o veículo <u>gira</u> tem intensidades (p,q,r) ao redor dos eixos **FRD** fixos ao corpo.



p – Velocidade de Rolamento – Roll speed

q – Velocidade de Arfagem – Pitch speed

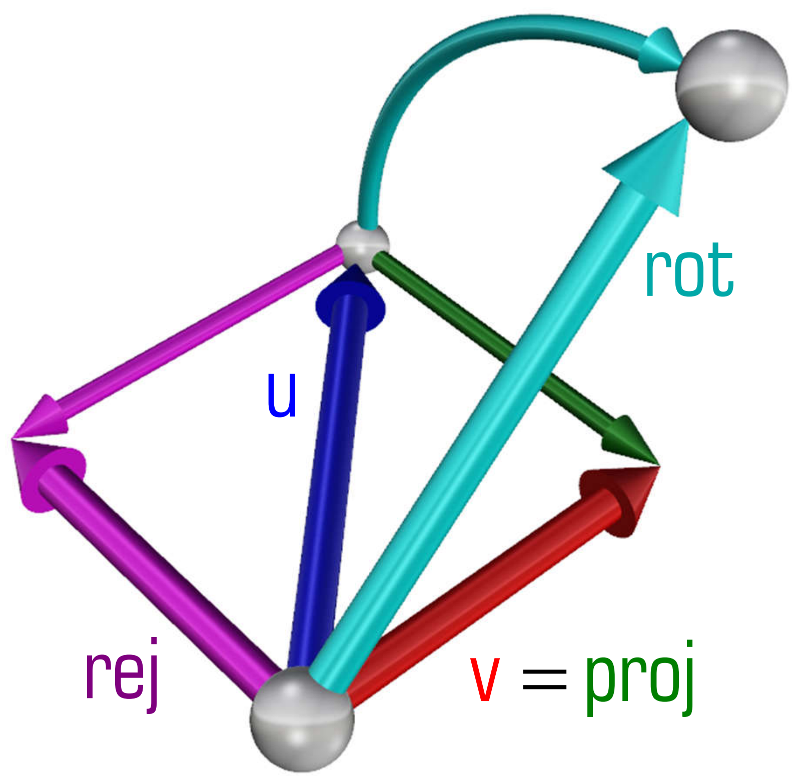
r – Velocidade de Guinada – Yaw speed

FORMA MATRICIAL DE VETOR

Coordenadas do vetor escritas como matriz coluna:

$$\vec{u} = \left(u_x, u_y, u_z\right) = u_x \hat{e}_x + u_y \hat{e}_y + u_z \hat{e}_z = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^T = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix}$$

```
% operações em MATLAB:
u = [1 1 0]';
v = [1 0 0]';
d = dot(u,v);
c = cross(u,v)
proj = v*dot(u,v)
rej = cross(v, cross(u,v))
rot = vecrot(u, v, pi/2)
% d=1    c=[0 0 -1]'
% proj=[1 0 0]'  rej=[0 1 0]'
% rot=[1 0 1]'
```



MATRIZ ANTISSIMÉTRICA

Matriz antissimétrica para auxiliar o produto vetorial

$$\vec{u} = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^T$$

$$\vec{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$$

$$\tilde{u} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix}$$

$$\vec{u} \times \vec{v} = \tilde{u}\,\vec{v} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} u_y v_z - u_z v_y \\ u_z v_x - u_x v_z \\ u_x v_y - u_y v_x \end{bmatrix}$$