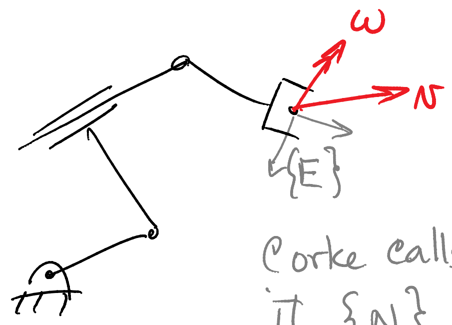
Goal is $\left\{ \begin{matrix} \text{force} \\ \text{velocity} \end{matrix} \right\}$ relationships:

Given $\dot{q}$, determine (v, ω) of end-effector



Corke calls it {N}

$$\boxed{\nu = J(q)\,\dot{q}}$$ eq(8.2)

$\tau = J^T(q)\, g \qquad g = \begin{bmatrix} f \\ m \end{bmatrix}_{6\times1}$ ← wrench

where $\nu = \begin{bmatrix} v \\ \omega \end{bmatrix}_{(6\times1)}$ ← a.k.a. spatial velocity

ν a.k.a. the "twist" of the end effector.

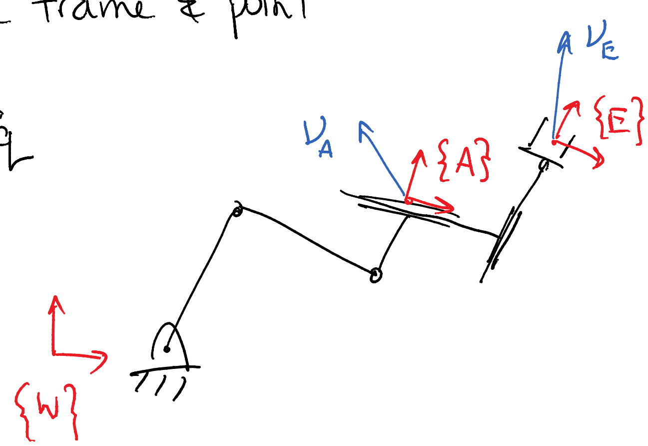
Notes: ν can be expressed in any convenient frame

J can be derived for any point on the robot

∴ You should specify the frame & point

e.g. $\quad {}^{W}\nu_{Eorg} = {}^{W}J_{Eorg}\,\dot{q}$

$${}^{E}\nu_{Aorg} = {}^{E}J_{Aorg}\,\dot{q}$$



Importance of Jacobian.

$v = J\dot{q}$ $\Longrightarrow$ allows us to know velocity in workspace from joint velocities

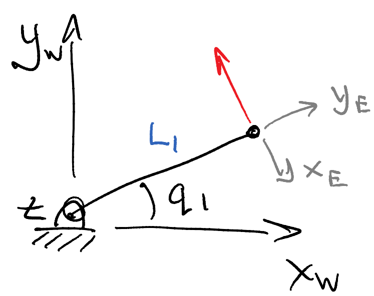
$\dot{q} = J^{-1} v$ $\Longrightarrow$ allows us to control the velocity in the workspace.

It identifies "singular" configurations, where small workspace velocities $\Longrightarrow$ large joint velocities

Planar examples:

1R-Planar robot

speed of end-effector $= \dot{q}_1 L_1$



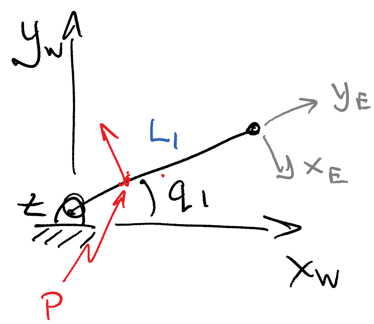
$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} -L_1 s_1 \\ L_1 c_1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \dot{q}_1$$

Note dependence on q.

Note v & ω are expressed in $\{W\}$. Express in $\{E\}$:

$$
{}^{E}\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}_{E_{org}} = \underbrace{\begin{bmatrix} -L_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}}_{{}^{E}J_{E_{org}}} \dot{q}_1
$$

Change the point.



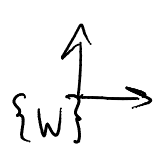
Now speed is $\frac{L_1}{3}\dot{q}_1$
but direction is
the same.

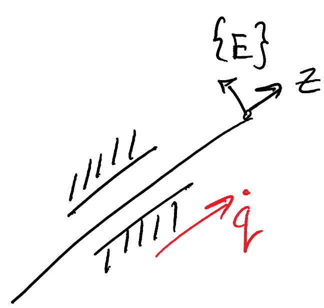
$$
{}^{W}v_P = \frac{1}{3}\,{}^{W}J_{E_{org}}\,\dot{q}_1
$$

$$
{}^{E}v_P = \frac{1}{3}\,{}^{E}J_{E_{org}}\,\dot{q}_1
$$

1P robot:

$$
{}^{W}\begin{bmatrix} v_x \\ v_y \\ v_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}_{E_{org}} = \underbrace{\begin{bmatrix} {}^{W}z_x \\ {}^{W}z_y \\ {}^{W}z_z \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{{}^{W}J_{E_{org}}} \dot{q}
$$





Change the point.

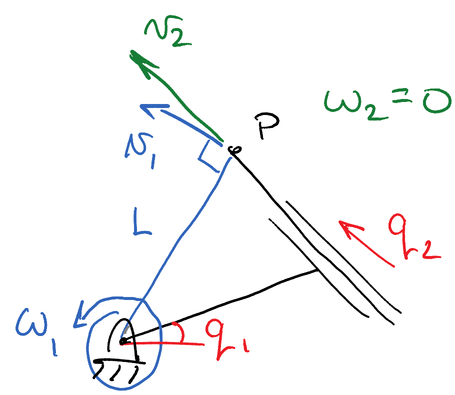
$$^{w}\nu_P = {}^{w}\underline{\underline{J_{E_{org}}}}\,\dot{q}$$

← No change since link is translating.

Put two links in series. What happens?

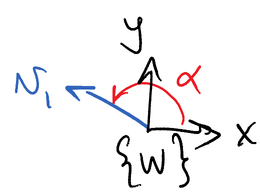
ν is a vector, so link effect add.

Consider each link in isolation



$$\nu = \nu_1 + \nu_2$$

$$\omega = \omega_1 + \omega_2$$



$$\|\nu_1\| = L_1 \dot{q}_1$$

$$^{w}\begin{bmatrix} \nu_x \\ \nu_y \\ \nu_z \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}_P = \begin{array}{c} \left[\begin{array}{c:c} L_1 c_\alpha & {}^{w}z_x \\ L_1 s_\alpha & {}^{w}z_y \\ 0 & 0 \\ \hdashline 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{array}\right] \\ \begin{array}{cc} \dot{q}_1 & \dot{q}_2 \end{array} \end{array} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

More joints? Just keep adding columns to J and elements to $\dot{q}$.

How do we get the columns of J for complex 3D robots?

General idea:

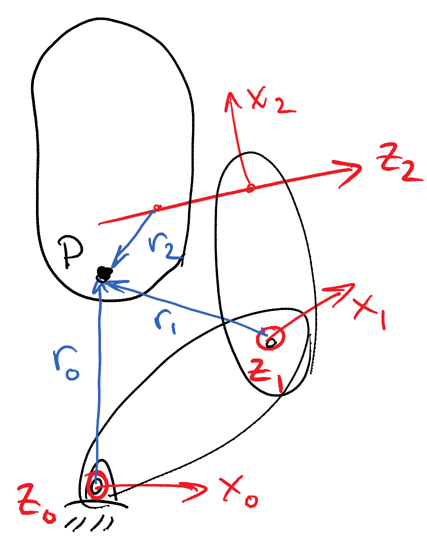
prismatic joints – the v rows of J are $\hat{z}$-vectors
the ω rows of J are $0_{3\times1}$.

revolute joints – the v rows of J can be constructed from $\hat{z}$ and a vector from $\hat{z}$ to the point
the ω rows of J are $\hat{z}$-vectors.

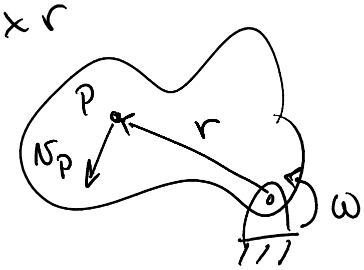
Example.

For prismatic joints we need only the $\hat{z}$ directions

For revolute joints we need $\hat{z}$ directions AND vectors from z-axes to P.



Recall: for a pinned body the velocity of a point p is $v_p = \omega \times r$



Consider the robot with only one joint moving at a time.

Assume all joints are revolute.

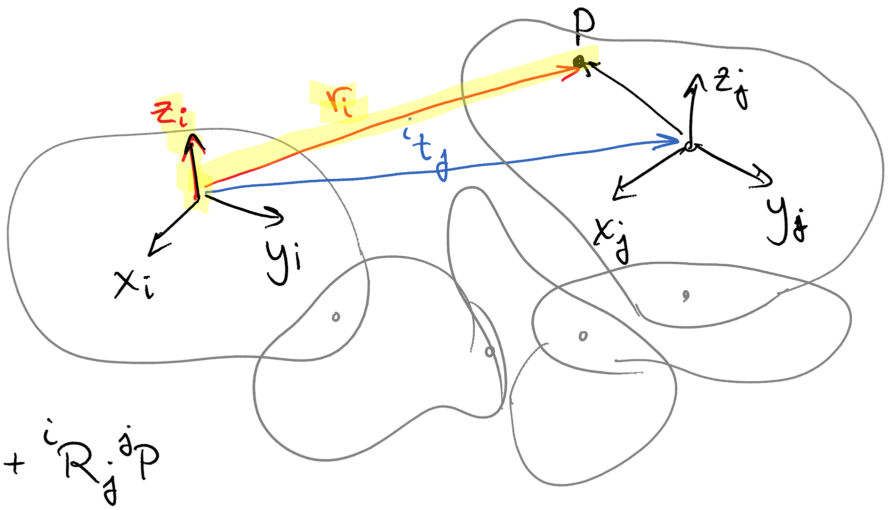
$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \hline \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \left[\begin{array}{c|c|c} \hat{z}_0 \times r_0 & \hat{z}_1 \times r_1 & \hat{z}_2 \times r_2 \\ \hline \hat{z}_0 & \hat{z}_1 & \hat{z}_2 \end{array}\right] \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}$$

Assume joint 1 is prismatic

$$\begin{bmatrix} v_x \\ v_y \\ v_z \\ \hline \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \left[\begin{array}{c|c|c} \hat{z}_0 \times r_0 & \hat{z}_1 & \hat{z}_2 \times r_2 \\ \hline \hat{z}_0 & 0 & \hat{z}_2 \end{array}\right] \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix}$$

How do you find r_k and $\hat{z}_k$?

$${}^{i}T_j = \begin{bmatrix} {}^{i}R_j & {}^{i}t_j \\ 0\,0\,0 & 1 \end{bmatrix} \qquad {}^{i}R_j = \begin{bmatrix} {}^{i}\hat{x}_j & {}^{i}\hat{y}_j & {}^{i}\hat{z}_j \end{bmatrix}$$



$$r_i = {}^{i}t_j + {}^{i}R_j\,{}^{j}P$$

$${}^{i}T_j = {}^{i}A_{i+1}(q_{i+1})\,{}^{i+1}A_{i+2}(q_{i+2}) \cdots {}^{j-1}A_j(q_j)$$ $\Leftarrow$ known from kinematic model, e.g. DH.

${}^{i}T_j$ contains ${}^{i}R_j$ & ${}^{i}t_j$

You must also know ${}^{j}P$

Construct J assuming ${}^{j}P$ is known

for $i = j$ to 1

Compute ${}^{i}T_j$ and r_j

If joint is revolute, col j of J is $\begin{bmatrix} \hat{z}_j \times r_j \\ \hat{z}_j \end{bmatrix}$

Else, col j. of J is $\begin{bmatrix} \hat{z}_j \\ 0 \end{bmatrix}$

end

A few details:

If P is on link $j < N$, columns $j+1, \dots, N$ are $0_{6\times 1}$

The quantities r_j and $\hat{z}_j$ must be expressed w.r.t. a common frame, say $\{A\}$. Then the Jacobian equation is: $\boxed{{}^{A}\nu = {}^{A}J_p(q)\,\dot{q}}$

Corke's text uses $\{E\} = \{N\}$ and $\{W\} = \{0\}$

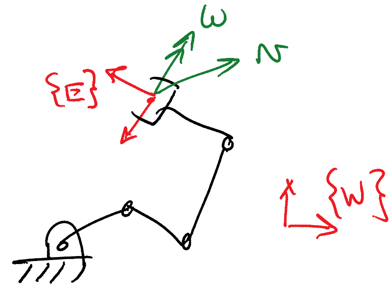
equation (8.2) $\rightarrow$ $\nu = J(q)\,\dot{q}$

more precisely it is $\boxed{{}^{0}\nu_N = {}^{0}J_N(q)\,\dot{q}}$ This notation is used in section 8.1.2 w/o intro.

Matlab function

jacob0 returns ${}^0J_N(q)$

zero



$$\Rightarrow \quad \text{jacob0}(q) * \dot{q} = \begin{bmatrix} {}^0v_N \\ {}^0\omega_N \end{bmatrix} = {}^0\nu_N$$