Square matrix A is orthogonal.

① $\|Au\| = \|u\|$ as $\|u\|^2 = u^T u$

② $\det(R) = \pm 1$ as $\det(R^T R) = \det(I)$

For rotation matrix, $\det(R) = r_1^T (r_2 \times r_3) = 1$

③ $A^T = A^{-1}$

④ A set of orthogonal vectors will <u>NOT</u> necessarily orthogonal after changing basis.

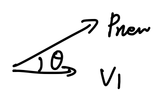
⑤ For any linearly independent set of vectors, we can <u>pick a basis</u> for those vectors s.t. they are orthonormal in new basis. (A frame, always find a transformation to a new frame)

Grahm - Schmidt Process

To make a set of vectors orthonormal

① Calculate projection onto every existing vectors.

$$proj = \frac{P_{new} \cdot V_1}{\| V_1 \|} \cdot V_1$$



② Subtract this proj from itself

$$V = P_{new} - proj$$

③ Normalize

$$V = \frac{P_{new} - proj}{\| P_{new} - proj \|}$$

Rotation Matrix.

$$R_{AB} = [\, x_{AB} \mid y_{AB} \mid z_{AB} \,]_{3\times3} = e^{\hat{\omega}\theta} \Leftarrow \begin{cases} \dot{q}(t) = \hat{\omega} q(t) \\ q(0) = q_0 \end{cases}$$

$R \in SO(3)$, $\hat{\omega} \in so(3)$

- $\hat{\omega}^T = -\hat{\omega}$
- $\hat{\omega}^3 = -\hat{\omega}$

Rodriguez: $e^{\hat{\omega}\theta} = I + \hat{\omega}\sin\theta + \hat{\omega}^2(1-\cos\theta)$ when $\|w\| = 1$

$(R = I + 2\hat{u}^2 \Rightarrow \theta = \pi,\ w = u)$

$tr(R) = 1 + \cos\theta = \sum \lambda_i$

Screw Motion (Every rigid body transformation)

$$\xi = \begin{bmatrix} -w\times q + hw \\ w \end{bmatrix} \qquad \hat{\xi} = \begin{bmatrix} \hat{w} & q\times w + hw \\ 0 & 0 \end{bmatrix}$$

given $R \Rightarrow w$

given $p \Rightarrow v = A^{-1} p$, $A = (I - e^{\hat{\omega}\theta})\hat{w} + ww^T\theta$

Exponential Properties

$$e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!} = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots + \frac{A^N}{N!} + \dots$$

① $(e^A)^T = e^{A^T}$ ② $(e^A)^{-1} = e^{-A}$

③ $e^{GAG^{-1}} = G e^A G^{-1}$

For rotation coordinates:

$w_1 = Rw$

$\hat{w} = R\hat{w}R^T$

$e^{\hat{w}\theta} = e^{R\hat{w}R^T\theta} = Re^{\hat{w}\theta}R^T$

④ $\lambda_1, \lambda_2, \dots \lambda_n$ are eigenvalues for A. $\begin{cases} \det(A-\lambda I) = 0 \\ (A-\lambda I)v = 0 \end{cases}$

$\det(A) = \lambda_1 \cdot \lambda_2 \cdot \lambda_3$

then $e^{\lambda_1}, e^{\lambda_2}, \dots e^{\lambda_n}$ are eigenvalues for e^A

$\det(e^A) = e^{\lambda_1} \cdot e^{\lambda_2} \cdots e^{\lambda_n} = e^{\sum \lambda_i} = e^{tr(A)}$

⑤ $\frac{de^{At}}{dt} = Ae^{At} = e^{At}A$

<u>Mobile axis</u>: Euler Angle.

Rotate Z, Y, X $R = R_Z R_Y R_X$

<u>Fixed axis</u>: RPY

Rotate Z, Y, X $R = R_X R_Y R_Z$.

General motion.

$$g_{AB} = \begin{bmatrix} R_{AB} & P_{AB} \\ 0 & 1 \end{bmatrix} = e^{\hat{\xi}\theta} \Leftarrow \begin{cases} \dot{q}(t) = \hat{\xi} p(t) \\ q(0) = q_0 \end{cases}$$

$g \in SE(3)$, $\hat{\xi} \in se(3)$

- $\hat{\xi} = \begin{bmatrix} \hat{w} & v \\ 0 & 0 \end{bmatrix}$ $\quad \xi = \begin{bmatrix} v \\ w \end{bmatrix}$

Revolute

$$\xi = \begin{bmatrix} -w\times q \\ w \end{bmatrix}$$

$$\hat{\xi} = \begin{bmatrix} \hat{w} & -w\times q \\ 0 & 0 \end{bmatrix}$$

$$e^{\hat{\xi}\theta} = \begin{bmatrix} e^{\hat{w}\theta} & (I - e^{\hat{w}\theta})q \\ 0 & 1 \end{bmatrix}$$

Prismatic

$$\xi = \begin{bmatrix} v \\ 0 \end{bmatrix}$$

$$\hat{\xi} = \begin{bmatrix} 0 & v \\ 0 & 0 \end{bmatrix}$$

$$e^{\hat{\xi}\theta} = \begin{bmatrix} I & v\theta \\ 0 & 1 \end{bmatrix}$$

$v = \frac{p}{\|p\|}$

$\theta = \|p\|$

$$e^{\hat{\xi}\theta} = \begin{bmatrix} e^{\hat{w}\theta} & (I - e^{\hat{w}\theta})(w\times v) + ww^T v\theta \\ 0 & 1 \end{bmatrix}$$

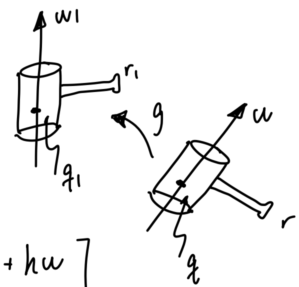
Transformation:

① points: $\bar{q}_1 = g \cdot \bar{q} \rightarrow \begin{bmatrix} \bar{q}_1 \\ 1 \end{bmatrix} = \begin{bmatrix} R & P \\ 0 & 1 \end{bmatrix} \begin{bmatrix} q \\ 1 \end{bmatrix}$

② vectors: $w_1 = Rw \longrightarrow \hat{w}_1 = R \hat{w} R^T$

③ twists: $\hat{\xi}_1 = g \circ \hat{\xi} \circ g^{-1}$

$$e^{\hat{\xi}_1 \theta} = g \circ e^{\hat{\xi}\theta} \circ g^{-1}$$



④ twist coordinate

$$\underbrace{\begin{bmatrix} -\hat{w}_1 q + h w_1 \\ w_1 \end{bmatrix}}_{\xi_1} = \underbrace{\begin{bmatrix} R & \hat{P} R \\ 0 & R \end{bmatrix}}_{Adg} \underbrace{\begin{bmatrix} -\hat{w} q + h w \\ w \end{bmatrix}}_{\xi}$$

---

vector · vector ⇒ scalar [dot / inner product]

vector × vector ⇒ vector [cross product].

vector · vector^T ⇒ matrix [outer product]

$tr(xx^T) = x^T x$, x is column vector.

Tricks:

① Calculate tagent of $p(x) \Rightarrow \frac{dp(x)}{dx}$.

② When need to find an expression for the XYZ positison specify the origin and the frame first.

③ Give 2 orthonormal vectors, $V_1 \times V_2 = V_3$.

④ $e^{\theta i} = \cos\theta + i \sin\theta$

⑤ triple product

- $(a \times b) \times c = -c \times (a \times b)$
- $(a \times b) \times c = a \times (b \times c) - b \times (a \times c)$
- $(a \times b) \times c = b(a \cdot c) - c(a \cdot b)$

⑥ $R_x(\theta_1) R_x(\theta_2) = R_x(\theta_1 + \theta_2)$

⑦ $(I - \hat{a})^{-1}(I + \hat{a}) \in SO(3)$

⑧ $Av = b \qquad v = (A^T A)^{-1} A^T b$

⑨ $\hat{w}^T \hat{w} = (w^T w) I - w w^T$

$$\|w\|_2^2 = \frac{1}{2} tr(\hat{w}^T \hat{w})$$

ROS: provide service expected from OS

graph: peer-to-peer network of processes.

- Query the camera sensing loop for a single image
- Use a Vision algo to compute the location of obj
- Compute joint angles to move the arm to the location
- Send position commands to each joint control loops
- Signal the gripper control loop to grab

each control loop = node

rospack find [package name]

catkin_make

catkin_create_pkg [name]

rosrun [pkg. name] [exe_name]

node $\xrightarrow{msg}$ topic → node

---

rosnode list

rosnode info /[name]

Rqt /node $\xrightarrow{topic}$ /node

rosservice call [name] [argv]

```
rospy.init_node('name')
try:
    talker()
def talker()
    pub = rospy.publisher('topic', [DataTy], qsize=10)

    r = rospy.rate(10)
    while not down:
        pubstring = 'string' (rospy.get_time())  < pub.publish(pubstring)
def listener():
    rospy.subscriber('topic', [DataTy], callback)
def callback(message):
        print(msg)
```

---

request

response