Student: Voeurn YongAnn

Institute of Robotics,
Innopolis University

Report

“Dynamics of Non-Linear Robotic Systems”

Homework-01

1. Forward Kinematic

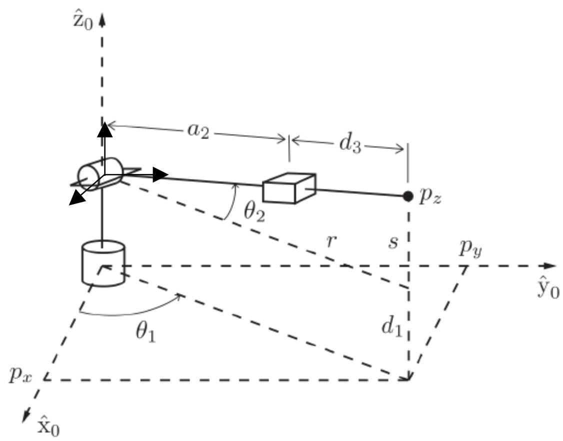

Figure 1 RRP robot

Denavit-Hartenberg convention

$$A_i = Rot_{x,\theta_i} Trans_{z_i d_i} Trans_{x,a_i} Rot_{x,a_i} \tag{1.1}$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\times \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} c_{\alpha_i} & s_{\theta_i} s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i} c_{\alpha_i} & -c_{\theta_i} s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The transformation from the base to the end-effector is given by

$$T_3^0 = T_1^0 T_2^1 T_3^2$$

Assigning the coordinate frames

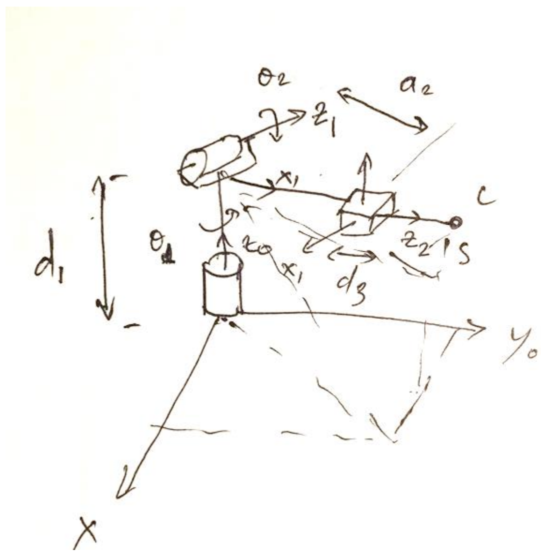

Figure 2 coordinate frame of robot for DH

Table 1 DH-parameter

J	Θ_i	d_i, mm	a_{i-1}	α_{i-1}
1	Θ_1	d_1	0	$-\pi/2$
2	Θ_2	a_1	0	$\pi/2$
3	0	d_3	0	0

$$T_1^0 = \begin{bmatrix} C_1 & 0 & -S_1 & 0 \\ S_1 & 0 & C_i & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ T_2^1 = \begin{bmatrix} C_2 & 0 & S_2 & 0 \\ S_2 & 0 & -C_2 & 0 \\ 0 & 1 & 0 & a_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, T_3^2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$T_3^0 = T_1^0 T_2^1 T_3^2 = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_y \\ R_{31} & R_{32} & R_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From MATLAB: T_3^0 =

```
[ cos(q1)*cos(q2), -sin(q1), cos(q1)*sin(q2), d3*cos(q1)*sin(q2) - a2*sin(q1)]
[ cos(q2)*sin(q1),  cos(q1), sin(q1)*sin(q2), a2*cos(q1) + d3*sin(q1)*sin(q2)]
[        -sin(q2),        0,         cos(q2),                  d1 + d3*cos(q2)]
[               0,        0,               0,                                1]
```

Geometry approach

Since $S = (a_2 + d_3)tan\theta_2$; $r^2 = S^2 + (a_2 + d_3)^2 = (a_2 + d_3)^2/cos^2(\theta_2)$

$$x_c = rcos(\theta_1) = C_1S_2d_3 - S_1d_2,$$

$$y_c = rsin(\theta_1) = S_1S_2d_3 + C_1d_2,$$

$$Z_c = d_1 + S = C_2d_3 + d_1,$$

2. Inverse Kinematic

Solve equation from forward kinematics geometry approach:

$$\theta_1 = atan2\left(\frac{x_c}{y_c}\right);$$

$$\theta_2 = atan2(s, r) + \pi/2;$$

$$d_3 = \sqrt{r^2 + S^2};$$

since $r_c = \sqrt{x_c^2 + y_c^2}$ $\quad s = z_c - d_1$.

3. Jacobian

Classical approach

$$J(q) = \begin{bmatrix} z_0 \times (o_3 - o_0) & z_1 \times (o_3 - o_1) & z_2 \\ z_0 & z_1 & 0 \end{bmatrix}$$

$$z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}; z_1 = \begin{bmatrix} s_1 \\ c_1 \\ 0 \end{bmatrix}; z_2 = \begin{bmatrix} c_1 s_2 \\ s_1 s_2 \\ c_2 \end{bmatrix};$$

$$o_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}; o_1 = \begin{bmatrix} 0 \\ 0 \\ d_1 \end{bmatrix}; o_3 = \begin{bmatrix} d_3 c_1 s_2 - a_2 S_1 \\ s_1 s_2 d_3 + a_2 c_1 \\ d_1 + d_3 c_2 \end{bmatrix};$$

$$J(q) = \begin{bmatrix} -a_2 c_1 - d_3 s_1 s_2 & d_3 c_1 c_2 & c_1 s_1 \\ -a_2 s_1 + d_3 c_1 s_1 & d_3 c_2 s_1 & s_1 s_2 \\ 0 & -s_2 d_3 & c_2 \\ 0 & -s_1 & 0 \\ 0 & c_1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

Geometry approach

$$\dot{x}_c = \dot{r}cos(\theta_1) - r\dot{\theta}_1 sin(\theta_1) = \frac{\dot{d}_3 cos\theta_2 cos\theta_1 + \dot{\theta}_2 sin\theta_2 cos\theta_1}{\cos^2\theta_2} - \dot{\theta}_1 sin\theta_1 \frac{a_2 + d_3}{cos\theta_2},$$

$$\dot{y}_c = \dot{r}sin(\theta_1) + r\dot{\theta}_1 cos(\theta_1) = \frac{\dot{d}_3 cos\theta_2 sin\theta_1 + \dot{\theta}_2 sin\theta_2 sin\theta_1}{\cos^2\theta_2} + \dot{\theta}_1 cos\theta_1 \frac{a_2 + d_3}{cos\theta_2}$$

$$\dot{Z}_c = \dot{d}_3\ tan\theta_2 - \frac{\dot{\theta}_2 d_3}{\cos^2\theta_2}$$

or

$$\begin{bmatrix} \dot{X} \\ \dot{Y} \\ \dot{Z} \end{bmatrix} = \begin{bmatrix} \frac{\partial X}{\partial \Theta_1} & \frac{\partial X}{\partial \Theta_2} & \frac{\partial X}{\partial d_3} \\ \frac{\partial Y}{\partial \Theta_1} & \frac{\partial Y}{\partial \Theta_2} & \frac{\partial Y}{\partial d_3} \\ \frac{\partial Z}{\partial \Theta_1} & \frac{\partial Z}{\partial \Theta_2} & \frac{\partial Z}{\partial d_3} \end{bmatrix} \begin{bmatrix} \dot{\Theta}_1 \\ \dot{\Theta}_2 \\ \dot{d}_3 \end{bmatrix}$$

$$\boldsymbol{V(q)} = \boldsymbol{J_v(q)\dot{q}} = \begin{bmatrix} -a_2 c_1 - d_3 s_1 s_2 & d_3 c_1 c_2 & c_1 s_1 \\ -a_2 s_1 + d_3 c_1 s_1 & d_3 c_2 s_1 & s_1 s_2 \\ 0 & -s_2 d_3 & c_2 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{d}_3 \end{bmatrix}$$

$$\boldsymbol{J_\omega} = [\boldsymbol{J\omega 1} \ \cdots \ \boldsymbol{J\omega n}] = [\boldsymbol{Z_0}\ \boldsymbol{Z_1}\ \boldsymbol{Z_2}]$$

$$\boldsymbol{Z_{i-1}} = \boldsymbol{R_{i-1}K}$$

Since $\boldsymbol{Z_0} = \boldsymbol{K} = (\boldsymbol{0}, \boldsymbol{0}, \boldsymbol{1})^T$; $\boldsymbol{z_1} = \begin{bmatrix} \boldsymbol{s_1} \\ \boldsymbol{c_1} \\ \boldsymbol{0} \end{bmatrix}$; $\boldsymbol{z_2} = \boldsymbol{0}$ (prismatics joint)

$$\boldsymbol{J_\omega(q)} = \begin{bmatrix} 0 & -s_1 & 0 \\ 0 & c_1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

4. Singular configuration

Singularities:

$$\det(J_v)=0$$

$$
\begin{aligned}
det\,(J) &= S_2d_3(S_1S_2(-S_1S_2d_3 - C_1d_2) - C_1S_2(C_1S_2d_3 - S_1d_2)) \\
&\quad +C_2((-S_1S_2d_3 - C_1d_2)(S_1C_2d_3) - (C_1S_2d_3 - S_1d_2)(C_1C_2d_3)) \\
&= S_2d_3[-S_1^2S_2^2d_3 - S_1S_2C_1d_2 - C_1^2S_1^2d_3 + C_1S_1S_2d_2] \\
&\quad +C_2[-S_1^2S_2C_2d_3^2 - C_1S_1C_2d_2d_3 - C_1^2S_2C_2d_3^2 + C_1S_1C_2d_2d_3] \\
&= S_2d_3[-d_3S_2^2[C_1^2 + S_1^2] + C_2[-S_2C_2d_3^2[S_1^2 + C_1^2]] \\
&= S_2d_3[-d_3S_2^2] - C_2^2S_2d_3^2 = -S_2^2d_3^2 - C_2^2S_2d_3^2 = -S_2d_3^2[S_2^2 + C_2^2] \\
det\,(J) &= -S_2d_3^2
\end{aligned}
$$

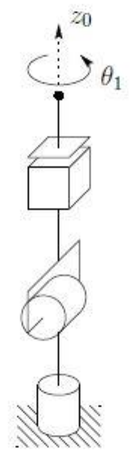

Figure 3 Singularity of RRP robot

When d3 = 0, the arm cannot move in Cartesian Z direction
When s2 = 0, the arm is tangent to the workspace inner boundary; it cannot move along shoulder axis direction. Thus any rotation about the base leaves this point fixed.

5. Velocity of the tool frame

The velocity of the tool frame when joint variables are changing with time as follows:

$$\theta_1(t) = sin(t);\ \theta_2(t) = cos(2t);\ d3(t) = sin(3t)$$

In this work the equation from geometry approach is used to calculate velocity of the tool frame..

By using MATLAB, we got following equation and graph:

$$
\begin{aligned}
\dot{x} = {} & 3\mathrm{cos(3t)cos(sin(t))sin(cos(2t))} - \mathrm{cos(sin(t))cos(t)} \\
& - \mathrm{sin(3t)sin(sin(t))sin(cos(2t))cos(t)} \\
& - 2\mathrm{sin(2t)sin(3t)cos(sin(t))cos(cos(2t))}
\end{aligned}
$$

$$
\begin{aligned}
\dot{y} = {} & 3cos(3t)sin(sin(t))sin(cos(2t)) - sin(sin(t))cos(t) \\
& + sin(3t)cos(sin(t))sin(cos(2t))cos(t) \\
& - 2sin(2t)sin(3t)sin(sin(t))cos(cos(2t))
\end{aligned}
$$

$$\dot{z} = 3cos(3t)cos(cos(2t)) + 2sin(2t)sin(3t)sin(cos(2t))$$

By the way this equation can be obtain by compute V(q)=J(q) $\dot{q}$.

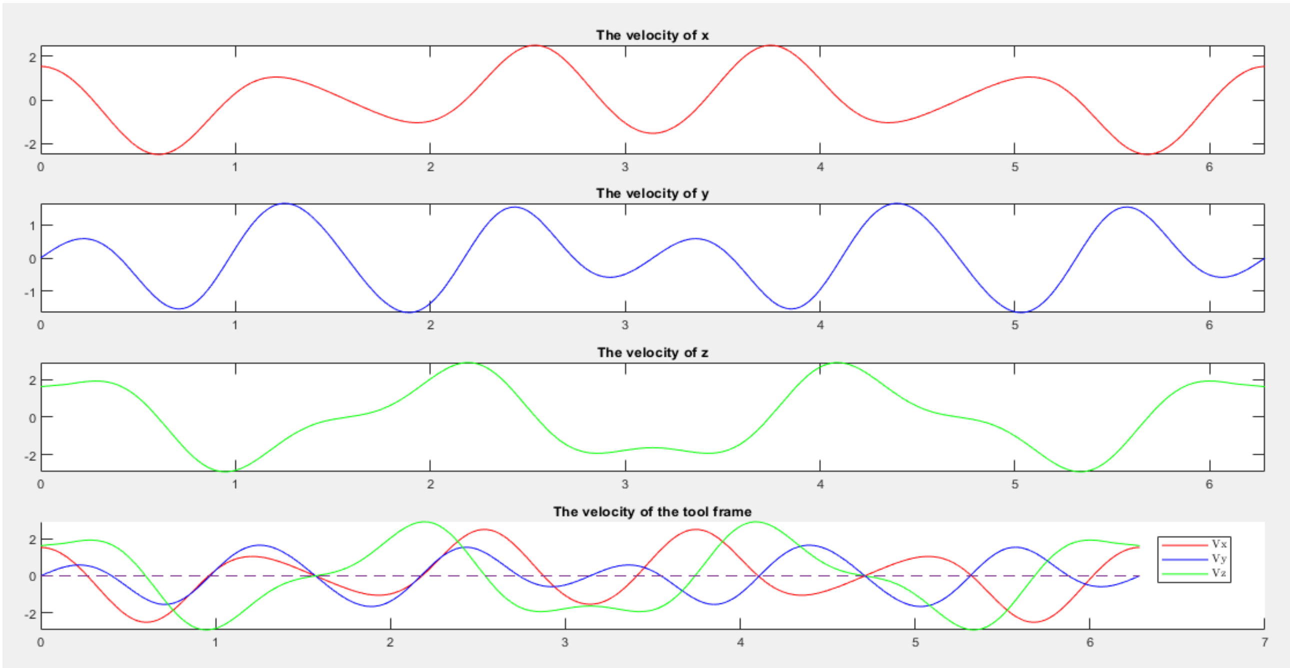

Figure 4 Velocity of tool frame

6. Joint trajectory

The tool coordinates changing with time as follows:

$$p_x(t) = 2a_2 sin(t), \qquad p_y(t) = 2a_2\, cos(2t), \qquad p_z(t) = d_1 sin(3t).$$

By applying this equation to IK solution:

$$\theta_1 = atan2\left(\frac{\sin(t),}{\cos(2t)}\right);$$

$$\theta_2 = atan2\left(2\sqrt{a_2^2 * \sin(t)^2 + a_2^2 * \cos(2t)^2}, 2d_1 sin(3t) - d_1\right) + \pi/2;$$

$$d_3 = 2\sqrt{a_2^2 \sin(t)^2 + 2a_2^2 \cos(2t)^2}$$

This equation is not defined is due to atan2 .

$$\operatorname{atan2}(y,x) = \begin{cases} \arctan(\frac{y}{x}) & \text{if } x > 0, \\ \arctan(\frac{y}{x}) + \pi & \text{if } x < 0 \text{ and } y \geq 0, \\ \arctan(\frac{y}{x}) - \pi & \text{if } x < 0 \text{ and } y < 0, \\ +\frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0, \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0, \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0. \end{cases}$$

$$\sin(t) = 0 \text{ ;}\cos(2t) = 0 \text{ or } t{=}\pi/4 \text{ and}$$

$$\sin(t)^2 + \cos(2t)^2{=}0 \text{ or } t{=}\ \pi/2$$

$$2d_1 sin(3t) - d_1 {=}0 \text{ or } t = \pi/18.$$

With Jacobian:

We have the singularity case

$$\det(J) = -S_2 d_3^2 = 0$$

$$atan2\left(2\sqrt{\mathrm{a}_2^2 * \sin(t)^2 + \mathrm{a}_2^2 * \cos(2t)^2}, 2d_1 sin(3t) - d_1\right) + \pi/2 = 0$$

By using the atan2 property

$$2\sqrt{\mathrm{a}_2^2 * \sin(t)^2 + \mathrm{a}_2^2 * \cos(2t)^2} = 0$$

$$2d_1 sin(3t) - d_1 < 0$$

d3 is always bigger than zero so the singularity points it when t= $\frac{\pi}{2}$ and t< $\frac{\pi}{18}$

This graph below is obtain by deriving the equation above respect to time t.

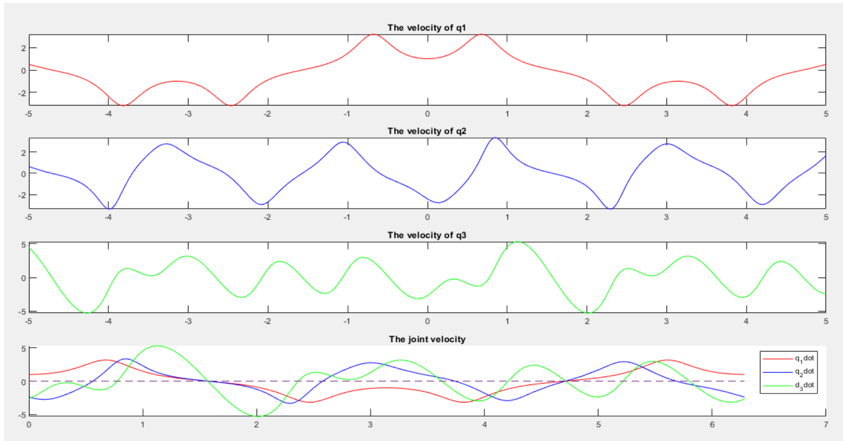

Figure 5 velocity of each joint

- Inverse Jacobean approach:

Algorithm 6.1. Inverse kinematics iteration technique.

1. *Set the initial counter* $i = 0$.
2. *Find or guess an initial estimate* $\mathbf{q}^{(0)}$.
3. *Calculate the residue* $\delta\mathbf{T}(\mathbf{q}^{(i)}) = \mathbf{J}(\mathbf{q}^{(i)})\,\delta\mathbf{q}^{(i)}$.

   If every element of $\mathbf{T}(\mathbf{q}^{(i)})$ *or its norm* $\|\mathbf{T}(\mathbf{q}^{(i)})\|$ *is less than a tolerance,* $\|\mathbf{T}(\mathbf{q}^{(i)})\| < \epsilon$ *then terminate the iteration. The* $\mathbf{q}^{(i)}$ *is the desired solution.*
4. *Calculate* $\mathbf{q}^{(i+1)} = \mathbf{q}^{(i)} + \mathbf{J}^{-1}(\mathbf{q}^{(i)})\;\delta\mathbf{T}(\mathbf{q}^{(i)})$.
5. *Set* $i = i + 1$ *and return to step 3.*

Since: $$\boldsymbol{V}(\boldsymbol{q}) = \boldsymbol{J_v}(\boldsymbol{q})\dot{\boldsymbol{q}}$$

That implies: $$\boldsymbol{\delta q} = \boldsymbol{J}^{-1}\boldsymbol{\delta p}$$

$$\boldsymbol{\delta q} = \boldsymbol{J_v^{-1}}(\boldsymbol{q})\begin{bmatrix}\delta p_x \\ \delta p_y \\ \delta p_z\end{bmatrix}$$

$$\boldsymbol{q_{i+1}} \;=\; \boldsymbol{q_i} \;+\; \boldsymbol{J}^{-1}(\boldsymbol{q_i})\;\boldsymbol{\delta p}.$$

This sub-section I calculated only inverse Jacobean (`HW3_calculation.m`).

MATLAB code:

- `HW3_calculation.m`
  `is for calculation of each section(fk,invk,J,Jinv)`

- `joint_trajectory.m`
  `is for making graph of joint trajectory`

- `velocity_toolframe_graph.m`
  `is for making graph of velocity tool frame`

GitHub link: https://github.com/yongan007/Hw3_Nonlinear_dynamics