哈爾濱工業大學(深圳)
HARBIN INSTITUTE OF TECHNOLOGY, SHENZHEN

《机器人学导论》实验(二)报告

专业：　　自动化

班级：　　2103206　　班

姓名：　　吴俊达

学号：　　210320621

2024 年 5 月

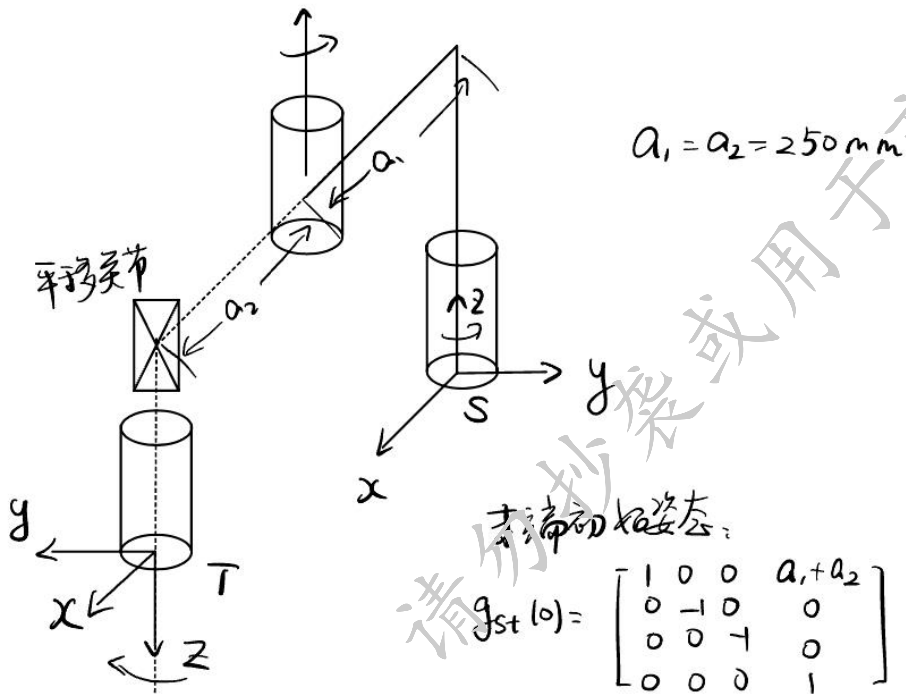
首先说明：

1. 所选机器人为 SCARA 机器人，接下来的分析均基于该机器人进行。

2. 我一开始对“门形轨迹”理解有误，误以为是要在 xOy 平面上呈现出一个门字形轨迹，所以采了 8 个点。解算出的轨迹和这 8 个点是吻合的。

一、 绘制简图

图中 S 为世界坐标系（或称 spatial frame），T 为工具坐标系（Tool Frame）。



二、 推导机器人正运动学

由指数积公式，$g=e^{\xi_1\theta_1}e^{\xi_2\theta_2}e^{\xi_3\theta_3}e^{\xi_4\theta_4}g(0)$

其中 $g(0)=\begin{bmatrix}1&0&0&a_1+a_2\\0&-1&0&0\\0&0&-1&0\\0&0&0&1\end{bmatrix}$（式中 a_1＝a_2＝250mm）

$$\xi_1=\begin{bmatrix}-\omega\times q\\ \omega\end{bmatrix}=\begin{bmatrix}0&0&0&0&0&1\end{bmatrix}^T$$

$\xi_2=\begin{bmatrix}-\omega_2\times q_2\\ \omega_2\end{bmatrix}$，$\omega_2=\begin{bmatrix}0\\0\\1\end{bmatrix},q_2=\begin{bmatrix}a_1\\0\\0\end{bmatrix}$，因此 $\xi_2=\begin{bmatrix}0&-a_1&0&0&0&1\end{bmatrix}^T$

第 3 轴为沿着-z 轴的平移关节，因此 $\xi_3=\begin{bmatrix}0&0&-1&0&0&0\end{bmatrix}^T$

$$\xi_4 = \begin{bmatrix} -\omega_4 \times q_4 \\ \omega_4 \end{bmatrix}，\ \omega_4 = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}, q_4 = \begin{bmatrix} a_1 + a_2 \\ 0 \\ 0 \end{bmatrix}，因此\ \xi_4 = \begin{bmatrix} 0 & a_1 + a_2 & 0 & 0 & 0 & -1 \end{bmatrix}^T$$

计算出上述 $e^{\xi_1\theta_1}, e^{\xi_2\theta_2}, e^{\xi_3\theta_3}, e^{\xi_4\theta_4}$，再代入指数积公式即可得解。计算过程用 MATLAB 实现如下：

```
syms theta1 theta2 theta3 theta4 a1 a2 real
q1 = [0;0;0];
q2 = [a1;0;0];
q4 = [a1+a2;0;0]; % 轴交点

w1 = [0;0;1];
w2 = [0;0;1];
w4 = [0;0;-1]; % 轴向向量

xi1 = [-cross(w1,q1); w1];
xi2 = [-cross(w2,q2); w2];
xi3 = [0;0;-1; 0;0;0]; % 平移关节
xi4 = [-cross(w4,q4); w4]; % 各关节旋量坐标
xi = [xi1 xi2 xi3 xi4];

g_init = [1,0,0,a1+a2;
          0,-1,0,0;
          0,0,-1,0;
          0,0,0,1]; % 初始位形

% forward
g_temp = Transformationsym(xi1,theta1) * Transformationsym(xi2,theta2) *
Transformationsym(xi3,theta3)  *Transformationsym(xi4,theta4) * g_init;
g_st = simplify(g_temp, 'Steps', 100);
% this method helps to get more simplified form
disp(g_st)
```

这个函数中调用了 Transformationsym 函数，其定义为：

```
function g = Transformationsym(xi, theta)
    xi_wedge = mywedge(xi).*theta;
    exp_xi = expm(xi_wedge);
    g = simplify(exp_xi,'Criterion','preferReal', 'Steps', 30)
end
```

其中又调用了函数 mywedge，其定义为（兼有 6 维向量的 wedge 功能和 3 维向量的 hat 功能）

```
function b=mywedge(a)
if size(a) == [6,1]
    b = subs(zeros(4,4));
```

```
    b(1,2) = -a(6,1);
    b(1,3) = a(5,1);
    b(2,1) = a(6,1);
    b(2,3) = -a(4,1);
    b(3,1) = -a(5,1);
    b(3,2) = a(4,1);

    b(1,4) = a(1,1);
    b(2,4) = a(2,1);
    b(3,4) = a(3,1);
elseif size(a) == [3,1]
    b = subs(zeros(3,3));

    b(1,2) = -a(3,1);
    b(1,3) = a(2,1);
    b(2,1) = a(3,1);
    b(2,3) = -a(1,1);
    b(3,1) = -a(2,1);
    b(3,2) = a(1,1);
end
```

计算得：

$$e^{\xi_1\theta_1}=\begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},e^{\xi_2\theta_2}=\begin{bmatrix} c_2 & -s_2 & 0 & -a_1(c_2-1) \\ s_2 & c_2 & 0 & -a_1 s_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},e^{\xi_3\theta_3}=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\theta_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$e^{\xi_4\theta_4}=\begin{bmatrix} c_4 & s_4 & 0 & -(a_1+a_2)(c_4-1) \\ -s_4 & c_4 & 0 & (a_1+a_2)s_4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$，最终的姿态矩阵

$$g_{st}=\begin{bmatrix} \cos(\theta_1+\theta_2-\theta_4) & \sin(\theta_1+\theta_2-\theta_4) & 0 & a_2c_{12}+a_1c_1 \\ \sin(\theta_1+\theta_2-\theta_4) & -\cos(\theta_1+\theta_2-\theta_4) & 0 & a_2s_{12}+a_1s_1 \\ 0 & 0 & -1 & -\theta_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

三、 用正运动学解算轨迹并绘制

利用下面的代码解算轨迹并绘制：

```
a1 = 250;
a2 = 250;
q1 = [0;0;0];
q2 = [a1;0;0];
q4 = [a1+a2;0;0]; % 轴交点

w1 = [0;0;1];
w2 = [0;0;1];
```

```
w4 = [0;0;-1]; % 轴向向量

xi1 = [-cross(w1,q1); w1];
xi2 = [-cross(w2,q2); w2];
xi3 = [0;0;-1; 0;0;0]; % 平移关节
xi4 = [-cross(w4,q4); w4]; % 各关节旋量坐标
xi = [xi1 xi2 xi3 xi4];

g_init = [1,0,0,a1+a2;
          0,-1,0,0;
          0,0,-1,0;
          0,0,0,1]; % 初始位形
theta = zeros(4,1);
last_loc = zeros(3,1);
cnt = 0;
step = 1;
ind = 1;
actural_points = [329.008,-59.002,-31.059;
    329.008,-139.002,-31.059;
    329.008,-139.003,-84.259;
    449.008,-139.003,-53.559;
    449.008,140.997,-53.559;
    335.008,156.997,-53.559;
    335.008,156.997,-83.559;
    335.008,156.997,-50.559]; % 实际规定的位置点

while ind<=size(loggeddata2,1)
    cnt =cnt+1;
    theta = loggeddata2(ind,5:8);
    gd = Fkine(xi,theta,g_init);
    location_all(cnt,1:3) = gd(1:3,4);
    step = min(round(6/ norm(gd(1:3,4) - last_loc)),100);  % 此步骤为根据前后两点距离改变步长，大致为每 6mm 绘制一个点
    last_loc = gd(1:3,4);
    ind = ind + step;
end

plot3(location_all(:,1),location_all(:,2),location_all(:,3)); % 画出三维线图来表示运动轨迹
hold on;
scatter3(actural_points(:,1),actural_points(:,2),actural_points(:,3)); % 三维散点图表示路径点
for k=1:1:8
    text_str = sprintf('p%d',k); % 添加各路径点标签，以 p1-p8 表示
    text(actural_points(k,1),actural_points(k,2),actural_points(k,3),text_str);
```

```
end
hold off
```

其中的正运动学函数：

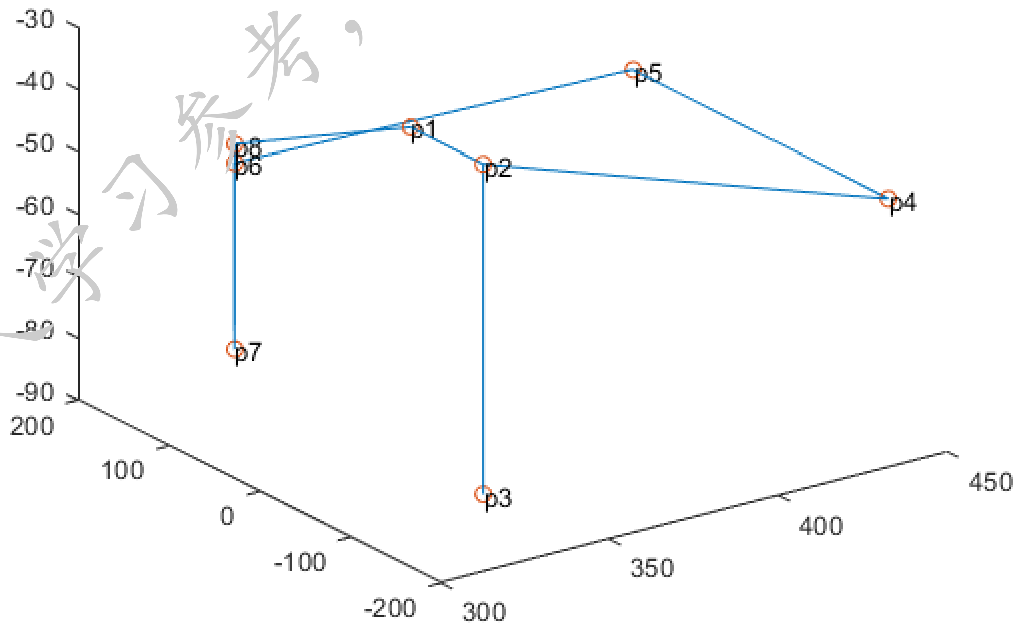
```
function g_st = Fkine(Xi,theta,g0)
    n = size(Xi,2);
    g_st = g0;
    for i = n:-1:1 % exp prod, multiply on the left, inverse sequence
        g_st = Transformation(Xi(:,i),theta(i))*g_st;
    end
end
```

其调用的 Transformation 函数：

```
function g = Transformation(xi, theta)
    v = xi(1:3);
    w = xi(4:6);
    if w==[0;0;0]
        g = [eye(3) v.*theta; 0 0 0 1];
    else
        theta = theta * pi / 180;
        w_hat = mywedge(w);
        R = eye(3) + sin(theta)*w_hat + (1-cos(theta))*w_hat^2; % Rodrigue's formula
        p = (eye(3) - R)*cross(w,v) + w * w' * v * theta;
        g = [R p; 0 0 0 1];
    end
end
```

其中又调用了函数 mywedge，其定义如上一部分。

绘制结果是：



可见和预先设定的路径点 p1-p8（见第四部分）相吻合。

四、 写出能够实现机器人码垛功能的宏指令集

宏指令集如下。简单介绍：①设置系统速度；②定义 8 个路径点；③给出 8 个路径点的坐标；④首先移动到一个较高点 p8，然后运行至起点；在运行至 p3 时启动吸嘴，吸起待放置物体，然后上移至 p2，依次移动至 p7[终点]，最后放下物体，从终点抬起至 p8。

```
System.Speed 10
Location p1
Location p2
Location p3
Location p4
Location p5
Location p6
Location p7
Location p8


p1=329.008,-59.002,-31.059,0.000,180.000,-43.575
p2=329.008,-139.002,-31.059,0.000,180.000,-43.575
p3=329.008,-139.003,-84.259,0.000,180.000,-43.575
p4=449.008,-139.003,-53.559,0.000,180.000,-43.575
p5=449.008,140.997,-53.559,0.000,180.000,-43.575
p6=335.008,156.997,-53.559,0.000,180.000,-43.575
p7=335.008,156.997,-83.559,0.000,180.000,-43.575
p8=335.008,156.997,-50.559,0.000,180.000,-43.575

Move.Line p8
Move.Line p1
Move.Line p2
Move.Line p3
Move.WaitForEOM
IO.Set DOUT(20103),1
Move.Line p2
Move.Line p4
Move.Line p5
Move.Line p6
Move.Line p7
Move.WaitForEOM
IO.Set DOUT(20103),0
WaitTime 250
Move.Line p8
```