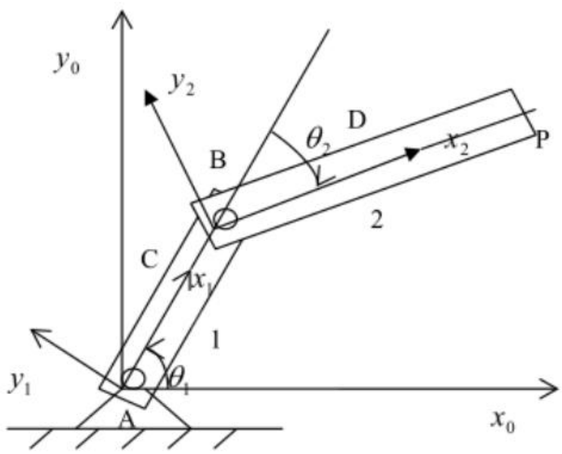
智能自主机器人与系统

二轴机械臂逆运动学求解

2150248 姚天亮 自动化

一、二连杆机械臂：

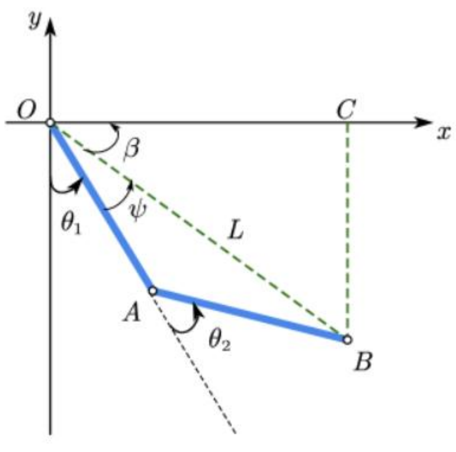
基础分析：



如图所示是本次实验课所要研究的二连杆机械臂。

为了研究方便，对参数进行如下定义：机器人广义坐标θ_1为连杆 1 与 y 轴负半轴的夹角，逆时针为正；θ_2为连杆 2 与连杆 1 延长线的夹角，逆时针为正；末端执行器笛卡尔坐标为$P(x, y)$ 。

二、问题 1. 求解二连杆机械臂角度、位置：



分析：由于两段杆长度相等，由几何关系可得：$l_1 = l_2 = l$, $\theta_2 = 2\phi$.

在$\triangle$ OAB 中使用余弦定理，得：$2l^2\cos(\pi - \theta_2) = l^2 + l^2 - L^2$,

又已知$L^2 = x^2 + y^2$.

解得：$\theta_2 = \arccos(\frac{x^2+y^2}{2l^2} - 1)$.

注意，由反三角函数的性质，这个θ_2有两个解，一个正解，一个负解。

同理可求：$\phi = \arccos(\frac{x^2+y^2}{2l^2})$.

为了计算方便，我们先规定$\beta = |\arctan(\frac{y}{x})|$.

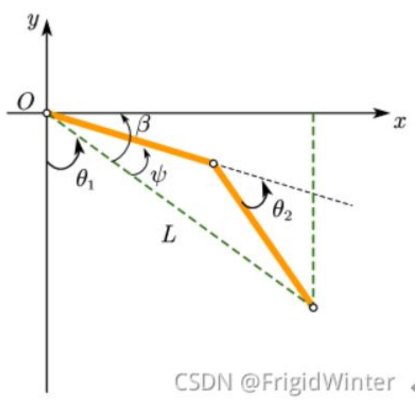
进而可得在四个象限内，平面二连杆机械臂的逆运动学解为：

$$\begin{cases} \theta_1 = \frac{\pi}{2} - (\beta \pm \psi), \theta_2 > 0\text{时取}+ \\ \theta_1 = \frac{\pi}{2} + (\beta \pm \psi), \theta_2 > 0\text{时取}- \\ \theta_1 = \frac{3\pi}{2} - (\beta \pm \psi), \theta_2 > 0\text{时取}+ \\ \theta_1 = \frac{3\pi}{2} + (\beta \pm \psi), \theta_2 > 0\text{时取}- \end{cases}$$

完成理论推导后，对应逆运动学求解部分的代码如下：

```
x = coord(1);
y = coord(2);
L1 = l(1);

fai = abs(acos(sqrt(x.^2+y.^2)/(2*L1)));
beta = abs(atan(y./x));
theta2 = acos((x.^2+y.^2)/(2*L1^2)-1);
if(x >= 0 && y >= 0)
   theta1 =  pi/2 - (beta(1) + fai(1));
elseif(x < 0 && y >= 0)
   theta1 = - pi/2 + (beta(1) - fai(1));
elseif(x < 0 && y < 0)
   theta1 = -pi/2 - (beta(1) - fai(1));
elseif(x >= 0 && y < 0)
   theta1 =  pi/2 + (beta(1) - fai(1));
end
theta = [theta1 theta2];
end
```

主函数部分，由于存在两解，现有程序只计算了 θ_2为正解情况下的值，于是我们添加代码以计算 θ_2为负解情况下的的值，从而画出第二个解的机械臂位姿。



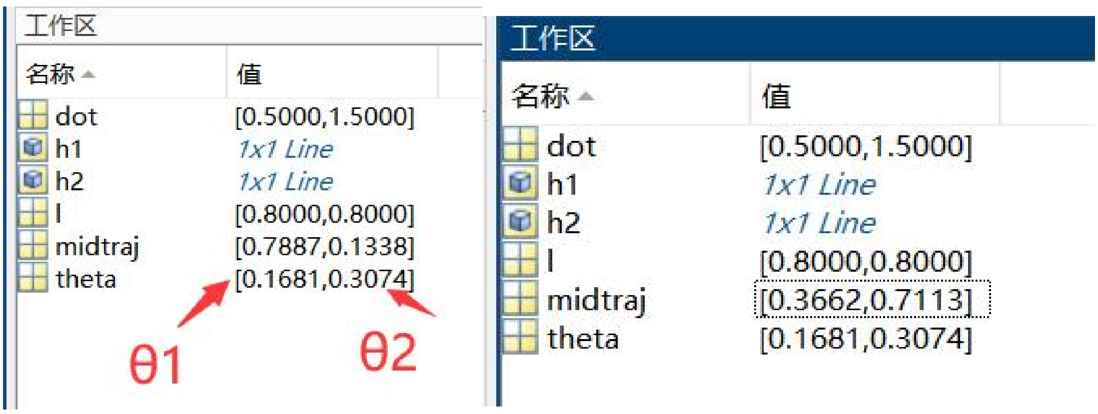
当 θ_2为负解时，对应位姿如图所示。

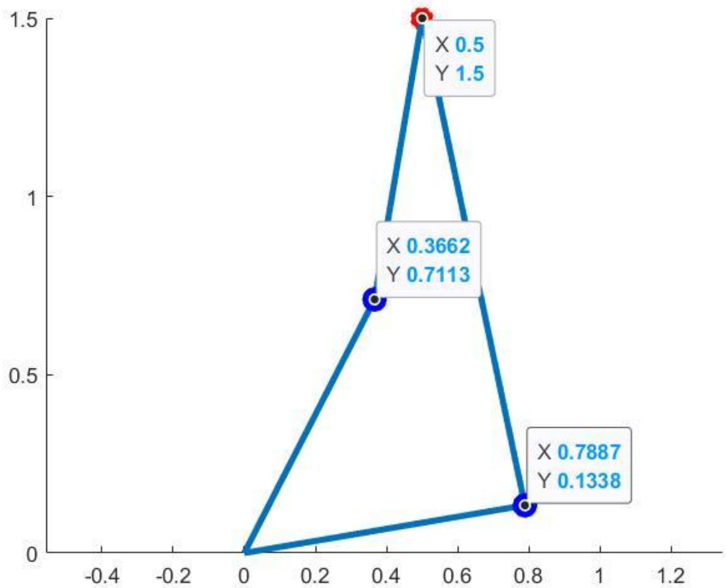
通过求解 $\theta_1 + \theta_2$，可得另一个解的情况。

代码如下：

```
%% Q1 给定点的逆运动学
    dot = [0.5 1.5];          % 给出机械臂需要到达的点
    theta = IKrob(dot, 1);   % 解出对应的关节角 <-------------------

    figure                                                           % 画图
    axis equal
    hold on
    plot(dot(1),dot(2),'r*','LineWidth',10);                         % 画目标点
    midtraj = [l(1) * cos(theta(1)),l(1) * sin(theta(1))];           % 计算关节1的位置

    h1 = line([0 midtraj(1)],[0 midtraj(2)],'LineWidth',3);          % 画杆1
    h2 = line([midtraj(1) dot(1)],[midtraj(2) dot(2)],'LineWidth',3);  % 画杆2
    plot(midtraj(1),midtraj(2),'bo','LineWidth',6);                  % 画关节1

    midtraj = [l(1) * sin(theta(1)+theta(2)),l(1) * cos(theta(1)+theta(2))];    % 计算关节1的位置
    h1= line([0 midtraj(1,1)],[0 midtraj(1,2)],'LineWidth',3);           % 画杆1
    h2= line([midtraj(1,1) dot(1)],[midtraj(1,2) dot(2)],'LineWidth',3);   % 画杆2
    plot(midtraj(1,1),midtraj(1,2),'bo','LineWidth',6);                     % 画关节1
```

下图为第一小问结果展示：



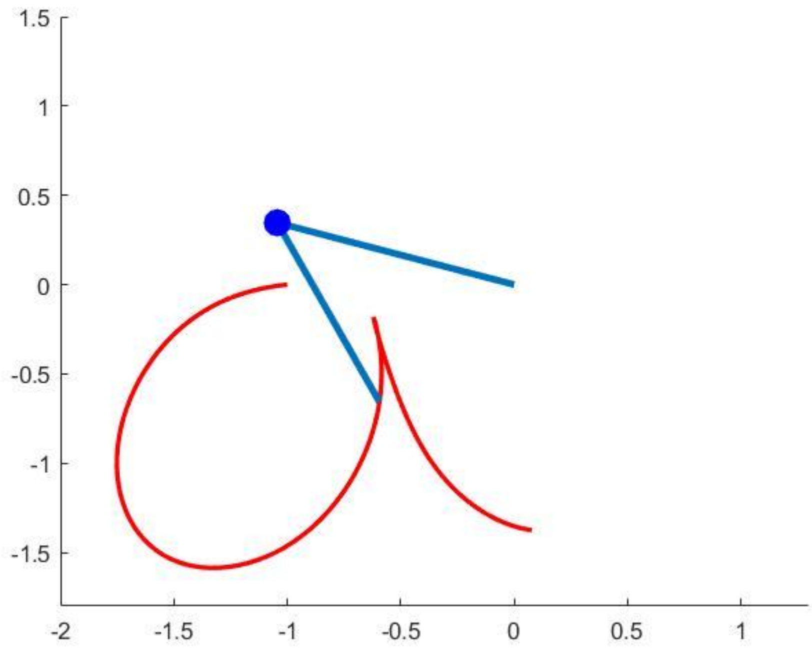


三、问题 2. 根据机械臂逆运动学求解关节空间中的轨迹

分析：本题需要将每个点的空间位置使用第一小题的方法进行反解，进而得到两个关节空间位置。为此，只需通过一个循环，遍历所有点，从而实现对每个点空间位置求解，最终通过求解，我们可以得到一个完整连续的二连杆机械臂的末端按照 a 形轨迹进行运动的运动动画，代码如下。

```
for k=1:trajactory_length
    thetaA(k,:)=IKrob(trajcoord(k,:),1);
    midtrajA(k,:) = [l(1) * sin(thetaA(k,1)) l(1) * cos(thetaA(k,1))];
end
```

效果图如下：



四、问题 3. 利用逆 Jacobian 矩阵求解关节空间的位置

分析：Jacobian 矩阵就是对 x、y 的式子进行求偏导，并进行相应操作，实现从关节角速度向末端位置速度的转变。

已知机器人末端的坐标$P(x_p, y_p)$，可表示为：

$$x_p = l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2)$$

$$y_p = l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2)$$

通过求导，可得对应的 Jacobian 矩阵为：

$$\begin{bmatrix} -l_1 \sin\theta_1 - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix} = J(\theta_1, \theta_2)$$

由此，为了实现求解关节空间的位置，我们可以通过遍历二连杆臂上的所有点，通过迭代计算 Jacobian 阵，实现点坐标的实时更新，进而完成运动轨迹模拟。
整体流程为:
1. 计算 Jacobian 矩阵;
2. 根据和 Jacobian 矩阵计算关节角速度;
3. 根据时间更新关节角度;
4. 根据更新后的关节角度计算新的笛卡尔坐标;
5. 重复这个过程可以实现连续时间下二连杆机械臂的运动轨迹模拟。

对应的 Jacobian 矩阵的代码为:

```
JacobiMatrix = [l(1)*sin(theta(1))+l(2)*sin(theta(1)+theta(2)) l(2)*sin(theta(1)+theta(2));
    -l(1)*cos(theta(1))-l(2)*cos(theta(1)+theta(2)) -l(2)*cos(theta(1)+theta(2))];
```

在主函数中，所写代码如下:

```
%% 逆雅可比矩阵求解（可以先用逆运动学求机械臂初始姿态）<----------------

    thetaB  = zeros(trajactory_length,2);                % 初始化机械臂关节角
    midtrajB = zeros(trajactory_length,2);               % 初始化关节1的位置

    for i=1:trajactory_length-1
        JacobiMatrix=Jacobi(thetaB(i,:),1);
        dtheta=JacobiMatrix\trajspeed(i,:)';
        thetaB(i+1,:)=thetaB(i,:)+dtheta'*dt;
        midtrajB(i+1,:)=[l(1)*sin(thetaB(i+1,1)) l(1)*cos(thetaB(i+1,1))];
    end
```

效果为: