机械手臂的DH模型与正运动学解

Denavit-Hartenberg(D-H)模型表示了对机器人连杆和关节进行建模的一种非常简单的方法，可用于任何机器人构型，也可用于表示在任何坐标中的变换。1955年,Denavit和Hartenberg在“ASME Journal of Applied Mechanics”发表了一篇论文，后来利用这篇论文来对机器人进行表示和建模，并导出了它们的运动方程，这已成为表示机器人和对机器人运动进行建模的标准方法。

1. DH模型的基本思想

假设机器人由一系列关节和连杆组成。这些关节可能是滑动（线性）的或旋转（转动）的，它们可以按任意的顺序放置并处于任意的平面。连杆也可以是任意的长度（包括零），它可能被弯曲或扭曲，也可能位于任意平面上。所以任何一组关节和连杆都可以构成一个我们想要建模和表示的机器人。当然，在我们的机械手臂上，关节一般是旋转的。

首先给每个关节指定一个参考坐标系，然后，确定从一个关节到下一个关节（一个坐标到下一个坐标）来进行变换的步骤。如果从基座到第一个关节，再从第一个关节到第二个关节直至到最后一个关节和末端执行器的所有变换结合起来，就得到了机器人的总变换矩阵。

DH模型通过限制原点位置和X轴的方向，人为减少了两个自由度，因此它只需要用四个参数即可表达关节之间原本是六自由度的坐标变换。DH选的四个参数都有非常明确的物理含义：

link length（连杆长度）：两个关节的轴（旋转关节的旋转轴，平移关节的平移轴）之间的公共法线长度。

link twist（连杆扭转）：一个关节的轴相对于另一个关节的轴绕它们的公共法线旋转的角度。

link offset（连杆偏移）：一个关节与下一个关节的公共法线和它与上一个关节的公共法线沿这个关节轴的距离。

joint angle（关节转角）：一个关节与下一个关节的公共法线和它与上一

个关节的公共法线绕这个关节轴的转角。

下面的图 1 是两个典型的机器人关节，虽然这种关节和连杆并不一定与任何实际机器人的关节或连杆相似，但是它们非常常见，且能很容易的表示实际机器人的任何关节。仔细看一下更有利于对 DH 模型的理解。

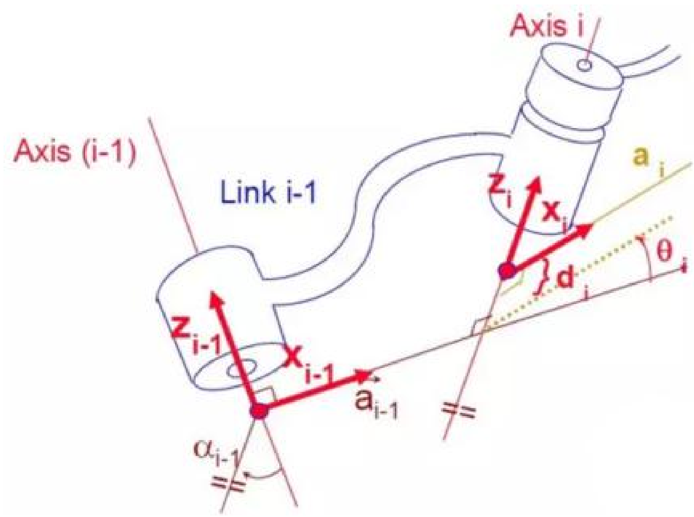

图 1 DH 模型参数说明

确定坐标系，一般有以下几个步骤：

为了用 D-H 表示法对机器人建模，第一件事就是为每个关节指定一个本地的参考坐标系。因此，对于每个关节，都必须指定一个 Z 轴和 X 轴。

指定 Z 轴，如果关节是旋转的，Z 轴位于按右手规则旋转的方向。绕 Z 轴的旋转角是关节变量;如果关节是滑动的，Z 轴为沿直线运动的方向。 沿 Z 轴的连杆长度 d 是关节变量。

指定 X 轴，当两关节不平行或相交时，z 轴通常是斜线，但总有一条距离最短的公垂线，它正交于任意两条斜线。在公垂线方向上定义本地参考坐标系的 x 轴。如果 a_n 表示 Z_{n-1} 与 Z_n 之间的公垂线，则 X_n 的方向将沿 a_n。

也有特殊情况，两关节 Z 轴平行，就会有无数条公垂线，此时可挑选与前一关节的公垂线共线的一条，可简化模型；两关节 Z 轴相交，它们之间没有公垂线（或者说公垂线距离为零）。这时可将垂直于两条轴线构成的平面的直线定义为 X 轴（相当于选取两条 Z 轴的叉积方向作为 X 轴），可简化模型。

给每个关节都附上单独的坐标系之后，如下图所示。

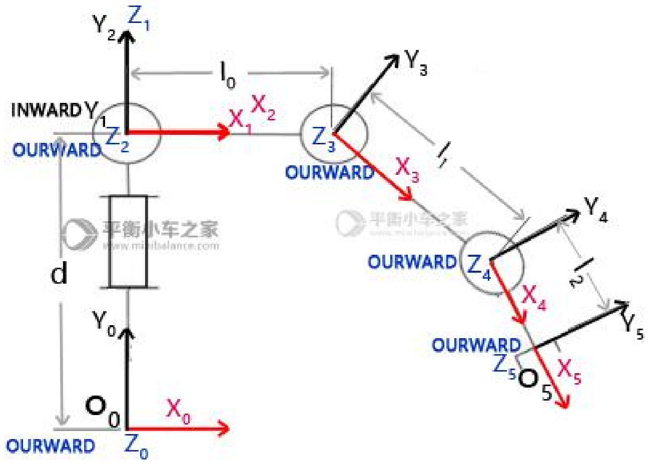

图 2 舵机版机械手臂

确定坐标系以后，我们可以用更简洁的方法来表示上面很绕口的四个参数：

Link length a_{i-1} :distance of(z_{i-1} , z_i)along x_{i-1}

Link twist α_{i-1} :angle of(z_{i-1}, z_i)abot x_{i-1}

Link offset d_i :distance of(x_{i-1}, x_i)along z_i

Joint angle θ_i :angle of(x_{i-1}, x_i)about z_i

接下来我们就可以填写机械臂的 DH 参数表了。

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	-90°	0	d	θ_0
2	90°	0	0	θ_1
3	0	l_0	0	θ_2
4	0	l_1	0	θ_3
5	0	l_2	0	0

根据以下转换公式：

$$
{}^{i-1}_{i}T=\begin{bmatrix}\cos\theta_i & -\sin\theta_i & 0 & a_{i-1}\\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i\\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

我们可以从每个关节的四个 DH 参数出发，最后得到一个机械臂的正运动学公式。我们现在的任务就是求末端执行器相对于 O_0 的 T 矩阵。按照上文提到的方法，我们首先把 DH 表带入转换公式，把两个相邻关节之间的 T 矩阵写出来。

$$
{}^{0}_{1}T=\begin{bmatrix}\cos\theta_0 & -\sin\theta_0 & 0 & 0\\ 0 & 0 & 1 & d\\ -\sin\theta_0 & -\cos\theta_0 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

$$
{}^{1}_{2}T=\begin{bmatrix}\cos\theta_1 & -\sin\theta_1 & 0 & 0\\ 0 & 0 & -1 & 0\\ \sin\theta_1 & \cos\theta_1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

$$
{}^{2}_{3}T=\begin{bmatrix}\cos\theta_2 & -\sin\theta_2 & 0 & l_0\\ \sin\theta_2 & \cos\theta_2 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

$$
{}^{3}_{4}T=\begin{bmatrix}\cos\theta_3 & -\sin\theta_3 & 0 & l_1\\ \sin\theta_3 & \cos\theta_3 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

$$
{}^{4}_{5}T=\begin{bmatrix}1 & 0 & 0 & l_2\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}
$$

得到旋转矩阵之后，根据下面的公式即可得到末端执行器的姿态：

$$
\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}={}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\begin{bmatrix}0\\ 0\\ 0\\ 1\end{bmatrix}
$$

下面是 Matlab 代码：

```
clear;
```

```
clc;
syms theta0 theta1 theta2 theta3 alpha;
syms l0 l1 l2 d;
syms x y z;
T01=[cos(theta0) -sin(theta0) 0 0;0 0 1 d;-sin(theta0) -cos(theta0) 0 0;0
0 0 1];
T12=[cos(theta1) -sin(theta1) 0 0;0 0 -1 0;sin(theta1) cos(theta1) 0 0;0
0 0 1];
T23=[cos(theta2) -sin(theta2) 0 l0;sin(theta2) cos(theta2) 0 0;0 0 1 0;0
0 0 1];
T34=[cos(theta3) -sin(theta3) 0 l1;sin(theta3) cos(theta3) 0 0;0 0 1 0;0
0 0 1];
T45=[1 0 0 l2;0 1 0 0;0 0 1 0;0 0 0 1];
T05=T01*T12*T23*T34*T45
T04=T01*T12*T23*T34
T5=T05*[0;0;0;1]
T4=T04*[0;0;0;1]
x=T5(1,1)
y=T5(2,1)
z=T5(3,1)
alpha=asin((T5(2,1)-T4(2,1))/l2)
```

Copy 到 matlab 上面可以直接运行，由此，在知道每个关节的转角的前提下就可以知道当前末端执行器的姿态了。θ_0，θ_1，θ_2，θ_3下面是结果：

```
x =
l0*cos(theta0)*cos(theta1) -
l2*(cos(theta3)*(cos(theta0)*sin(theta1)*sin(theta2) -
cos(theta0)*cos(theta1)*cos(theta2)) +
sin(theta3)*(cos(theta0)*cos(theta1)*sin(theta2) +
cos(theta0)*cos(theta2)*sin(theta1))) -
l1*(cos(theta0)*sin(theta1)*sin(theta2) -
cos(theta0)*cos(theta1)*cos(theta2))
y =
d + l1*(cos(theta1)*sin(theta2) + cos(theta2)*sin(theta1)) +
l0*sin(theta1) + l2*(cos(theta3)*(cos(theta1)*sin(theta2) +
cos(theta2)*sin(theta1)) + sin(theta3)*(cos(theta1)*cos(theta2) -
sin(theta1)*sin(theta2)))
```

```
z =
l1*(sin(theta0)*sin(theta1)*sin(theta2) -
cos(theta1)*cos(theta2)*sin(theta0)) +
l2*(cos(theta3)*(sin(theta0)*sin(theta1)*sin(theta2) -
cos(theta1)*cos(theta2)*sin(theta0)) +
sin(theta3)*(cos(theta1)*sin(theta0)*sin(theta2) +
cos(theta2)*sin(theta0)*sin(theta1))) - l0*cos(theta1)*sin(theta0)
alpha =
asin(cos(theta3)*(cos(theta1)*sin(theta2) + cos(theta2)*sin(theta1)) +
sin(theta3)*(cos(theta1)*cos(theta2) - sin(theta1)*sin(theta2)))
```