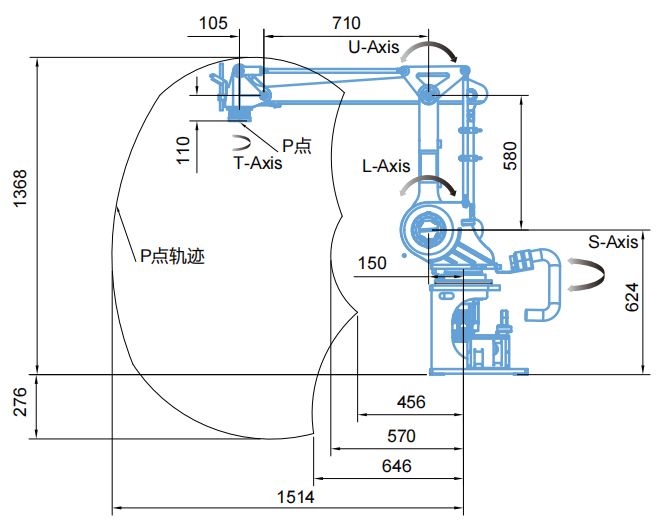
1. **机型简介**



**\_**

**\_**

**\_**

**+**

**+**

**+**

**\_**

**+**

图1.1设计参数

表1.1 ECR20产品参数表

|  |  |  |
| --- | --- | --- |
| 产品参数表 | | |
| 结构形式 | | 多关节机器人(RRRR) |
| 自由度 | | 四轴 |
| 有效负载 | | 20KG |
| 最大活动半径 | | 1521mm |
| 重复定位精度 | | ±0.06mm |
| 运动范围 | J1 | ±150° |
| J2 | -98°, +43° |
| J3 | -104°, +22° |
| J4 | ±360° |
| 本体重量 | | 150KG |
| 环境条件 | 温度 | 0~45℃ |
| 湿度 | 20~80%RH(不结露) |
| 振动 | 4.9m/s2以下 |
| ﹡不可有引火性及腐蚀性气液体  ﹡不可涉及水、油、粉  ﹡不可靠近电磁气源头 | |
| 电源容量 | | 4.15 KVA |

本简介所涉及的产品资料仅供参考，最终图纸请以实际签订的合同内容为准。

1. **ECR20四轴机器人正运动学**

**2.1 符号推理**

|  |
| --- |
| 杆件简图(几何) |

图2.1正解杆件简图

根据设计参数，可知上图中杆件尺寸为

****

通过几何关系，依次推算出各端点的空间位姿描述为

|  |  |
| --- | --- |
| O1 |  |
| O2 |  |
| O3 |  |
| M |  |
| N |  |
| O4 |  |

**2.2 测试样例**

由参数d1=624，d2=15，d5=110，a1=150，a2=580，a3=710，a4=105，可计算零位时(q1=q2=q3=q4=0)，末端TCP(即O4点)位姿：



1. **ECR20四轴机器人逆运动学**

**3.1 符号推理**

已知TCP(即O4点)位姿描述为：

****

暂时定义****

将前两个式子左右对应相加，可得到如下等式：



可求得(参考https://blog.csdn.net/fengyu19930920/article/details/81144042 2.1.2节)：



当d2=0时，若X4>Y4，取-号；若X4<Y4，取+号。

其中通过将前两个式子左右两端求平方，然后对应相加可得W：

****

根据q1可求得q4：

****

再通过几何法，继续求解q2、q3。几何法杆件简图如下：

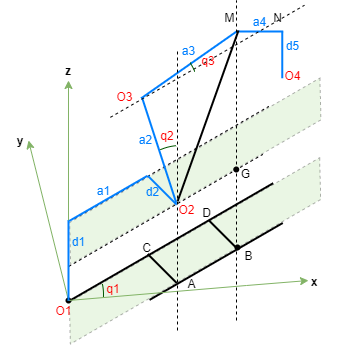


图3.1逆解空间杆件简图

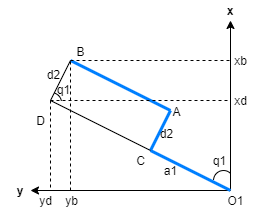


图3.2逆解地面投影简图

其中M点地面投影为B点，O2点地面投影为A点，偏置a1地面投影为线段O1C，偏置d2地面投影为线段AC，线段O2G地面投影为线段AB。

有TCP(即O4点)位姿，可推得M点空间坐标为：

****

由于B点是M点地面投影

****

地面D点坐标

****

因此，在地面投影中



图3.1中，MB=ZM，BG=d1，因此







在ΔMO2G中,

在ΔMO2O3中，根据余弦定理可得

此时可知，q2=∠MO2G+∠MO2O3-90°。

在ΔMO2O3中，根据余弦定理可得

此时可知，q3=∠MO3O2-(90°-q2)。

至此已求得全部关节位移。

**3.2 测试样例**

已知TCP位姿为：



代入DH数据可得关节位移：



1. **代码实现**
   1. **MATLAB代码**
      1. **正解函数**
2. % 正解函数
3. % 输入：各关节位移
4. %    旋转轴位移值单位为°,移动轴位移值单位为mm
5. % 输出：TCP位姿描述(XYZABC)
6. %    XYZ输出值单位为mm,ABC输出值单位为°
7. function [p]=ECR20\_1500\_fkine(q)
8. % 连杆参数
9. a1=150;
10. a2=580;
11. a3=710;
12. a4=105;
13. d1=624;
14. d2=15;
15. d5=110;
16. q=deg2rad(q);
17. % 位姿计算
18. x=cos(q(1))\*(a1 - a2\*sin(q(2)) + a3\*cos(q(3)) + a4)+sin(q(1))\*d2;
19. y=sin(q(1))\*(a1 - a2\*sin(q(2)) + a3\*cos(q(3)) + a4)-cos(q(1))\*d2;
20. z=d1 + a2\*cos(q(2)) + a3\*sin(q(3)) - d5;
21. a=rad2deg(q(1) - q(4));
22. b=0;
23. c=-180;
24. % 结果输出
25. p=[x,y,z,a,b,c];
26. end
    * 1. **逆解函数**
27. % 逆解函数
28. % 输入：TCP位姿描述(XYZABC)
29. %       XYZ输出值单位为mm,ABC输出值单位为°
30. % 输出：各关节位移
31. %       旋转轴位移值单位为°,移动轴位移值单位为mm
32. function [q]=ECR20\_1500\_ikine(p)
33. % 连杆参数
34. a1=150;
35. a2=580;
36. a3=710;
37. a4=105;
38. d1=624;
39. d2=15;
40. d5=110;
41. % TCP位姿
42. px=p(1);
43. py=p(2);
44. pz=p(3);
45. pa=deg2rad(p(4));
46. pb=deg2rad(p(5));
47. pc=deg2rad(p(6));
48. % 求关节转角1
49. W=sqrt(px^2+py^2-d2^2);
50. q1=atan2(W-d2,-W-d2)-atan2(px+py,-sqrt((W-d2)^2+(-W-d2)^2-(px+py)^2));
51. if q1>pi
52. q1=q1-pi\*2;
53. elseif q1<-pi
54. q1=q1+pi\*2
55. end
56. % 求关节转角4
57. q4=q1 - pa;
58. q4=atan2(sin(q4), cos(q4));
59. % 求关节转角2
60. px=px - cos(q1)\*a4 - sin(q1)\*d2;
61. py=py - sin(q1)\*a4 + cos(q1)\*d2;
62. pz=pz + d5 - d1;
63. temp=sqrt(px^2 + py^2)-a1;
64. q21=atan2(pz, temp);
65. q22=acos((a2^2 + temp^2 + pz^2 - a3^2)/(2\*a2\*sqrt(temp^2+pz^2)));
66. q2=q21 + q22 - pi/2;
67. % 求关节转角3
68. c3=(a2^2 + a3^2 - (temp^2+pz^2))/(2\*a2\*a3);
69. q3=acos(c3) - (pi/2-q2);
70. q=rad2deg([q1,q2,q3,q4]);
71. end
    1. **ST代码**
       1. **正解函数**
72. FUNCTION FN\_RRRR\_FK\_ECR : CoordPoint
73. VAR\_INPUT
74. sActAngle : AxisAngle; *//待正解的四轴各轴弧度*
75. a1,a2,a3,a4 : LREAL;  *//DH参数值-a*
76. d1,d2,d5 : LREAL;  *//DH参数值-d*
77. END\_VAR
78. VAR
79. s1,s2,s3,c1,c2,c3 : LREAL;
80. END\_VAR
81. */////////////////////////////////////////////////////*
82. sActAngle.lrAxis1 := sActAngle.lrAxis1\*deg2rad;
83. sActAngle.lrAxis2 := sActAngle.lrAxis2\*deg2rad;
84. sActAngle.lrAxis3 := sActAngle.lrAxis3\*deg2rad;
85. sActAngle.lrAxis4 := sActAngle.lrAxis4\*deg2rad;
86. *//展开计算---减小计算量*
87. s1 := SIN(sActAngle.lrAxis1);
88. s2 := SIN(sActAngle.lrAxis2);
89. s3 := SIN(sActAngle.lrAxis3);
90. c1 := COS(sActAngle.lrAxis1);
91. c2 := COS(sActAngle.lrAxis2);
92. c3 := COS(sActAngle.lrAxis3);
93. *//TCP位姿*
94. FN\_RRRR\_FK\_ECR.X := c1\*(a1-a2\*s2 + a3\*c3 + a4) + d2\*s1;
95. FN\_RRRR\_FK\_ECR.Y := s1\*(a1-a2\*s2 + a3\*c3 + a4) - d2\*c1;
96. FN\_RRRR\_FK\_ECR.Z := d1 + a2\*c2 + a3\*s3 - d5;
97. FN\_RRRR\_FK\_ECR.A := (sActAngle.lrAxis1-sActAngle.lrAxis4)\*rad2deg;
98. FN\_RRRR\_FK\_ECR.B := 0;
99. FN\_RRRR\_FK\_ECR.C := 0;
100. *//附加轴位姿*
101. FN\_RRRR\_FK\_ECR.Aux[0] := sActAngle.lrAxis7;
102. FN\_RRRR\_FK\_ECR.Aux[1] := sActAngle.lrAxis8;
103. FN\_RRRR\_FK\_ECR.Aux[2] := sActAngle.lrAxis9;
     * 1. **逆解函数**
104. FUNCTION FN\_RRRR\_IK\_ECR : AxisAngle
105. VAR\_INPUT
106. cCoordPoint  : CoordPoint; *//当前空间位置姿态*
107. InCurAxisAngle : AxisAngle; *//上一次关节角度*
108. config   : ARRAY[0..2] OF INT; *//姿势选择*
109. a1,a2,a3,a4  : LREAL;  *//DH参数值-a*
110. d1,d2,d5  : LREAL;  *//DH参数值-d*
111. END\_VAR
112. VAR
113. px,py,pz,pa : LREAL;
114. theta1,theta2,theta3,theta4 : LREAL;
115. Q21,Q22,temp : LREAL;
116. END\_VAR
117. *////////////////////////////////////////////////////*
118. px := cCoordPoint.X;
119. py := cCoordPoint.Y;
120. pz := cCoordPoint.Z;
121. pa := cCoordPoint.A\*Deg2Rad;
122. *//求关节转角1*
123. temp := SQRT(EXPT(px, 2)+EXPT(py, 2) - EXPT(d2, 2));
124. theta1 := NW\_atan2(temp-d2,-temp-d2) -
125. NW\_atan2(px+py, config[0]\*SQRT(EXPT(temp-d2, 2) + EXPT(-temp-d2, 2) - EXPT(px+py, 2)));
126. *//求关节转角4*
127. theta4 := theta1 - pa;
128. theta4 := NW\_atan2(SIN(theta4), COS(theta4));
129. IF (theta4 - InCurAxisAngle.lrAxis4\*Deg2Rad) > PI THEN
130. theta4 := theta4 - PI2;
131. ELSIF (theta4 - InCurAxisAngle.lrAxis4\*Deg2Rad) < -PI THEN
132. theta4 := theta4 + PI2;
133. END\_IF
134. *//求关节转角2*
135. px := px - COS(theta1)\*a4 - SIN(theta1)\*d2;
136. py := py - SIN(theta1)\*a4 + COS(theta1)\*d2;
137. pz := pz + d5 - d1;
138. temp := SQRT(px\*px + py\*py)-a1;
139. Q21 := NW\_atan2(pz, temp);
140. Q22 := ACOS((a2\*a2 + temp\*temp + pz\*pz - a3\*a3)/(2\*a2\*SQRT(temp\*temp+pz\*pz)));
141. theta2 := Q21+Q22-PIHalf;
142. *//求关节转角3*
143. theta3 := ACOS((a2\*a2 + a3\*a3 - temp\*temp - pz\*pz)/(2\*a2\*a3)) - (PIHalf-theta2);
144. FN\_RRRR\_IK\_ECR.lrAxis1 := theta1\*Rad2Deg;
145. FN\_RRRR\_IK\_ECR.lrAxis2 := theta2\*Rad2Deg;
146. FN\_RRRR\_IK\_ECR.lrAxis3 := theta3\*Rad2Deg;
147. FN\_RRRR\_IK\_ECR.lrAxis4 := theta4\*Rad2Deg;
148. FN\_RRRR\_IK\_ECR.lrAxis5 := 0;
149. FN\_RRRR\_IK\_ECR.lrAxis6 := 0;
150. FN\_RRRR\_IK\_ECR.lrAxis7 := cCoordPoint.Aux[0];
151. FN\_RRRR\_IK\_ECR.lrAxis8 := cCoordPoint.Aux[1];
152. FN\_RRRR\_IK\_ECR.lrAxis9 := cCoordPoint.Aux[2];