# 山东大学___计算机科学与技术___学院

# 课程实验报告

| 学号：202100130052 | 姓名：刘欣月 | 班级：人工智能班 |
| --- | --- | --- |
| 实验题目：实验 1 | | |
| 实验学时：6 | 实验日期：20210510 | |
| 实验目的：MATLAB 仿真实验 | | |
| 实验环境：matlab | | |

1. 实验步骤：
2. MATLAB 理论知识和 MATLAB Bobotic Toolbox 安装省略
3. MATLAB Bobotic Toolbox 简单应用，坐标系的变换

第三章实训内容

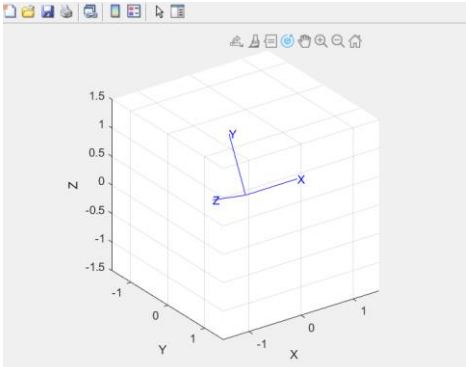
（1）练习前面的七个例题，

例题 3，1

```
>> R=rotx(30*pi/180)

R =

    1.0000         0         0
         0    0.8660   -0.5000
         0    0.5000    0.8660
```

例题 3.2

```
>> R=rotx(0)

R =

     1     0     0
     0     1     0
     0     0     1

>> R2=rotx(90);
>> tranimate(R,R2)
```




# 例题 3. 3

```
>> T=transl(1,2,3)

T =

     1     0     0     1
     0     1     0     2
     0     0     1     3
     0     0     0     1

>> p=[2,3,4];
>> T=transl(p)

T =

     1     0     0     2
     0     1     0     3
     0     0     1     4
     0     0     0     1
```

# 例题 3. 4

```
>> T=transl(3,4,5)

T =

     1     0     0     3
     0     1     0     4
     0     0     1     5
     0     0     0     1

>> p=transl(T)

p =

     3
     4
     5
```

```
>> [x,y,x]=transl(T)

x =

     5


y =

     4


x =

     5
```

## 例题 3.5 矩阵分解 R=t2r(T)

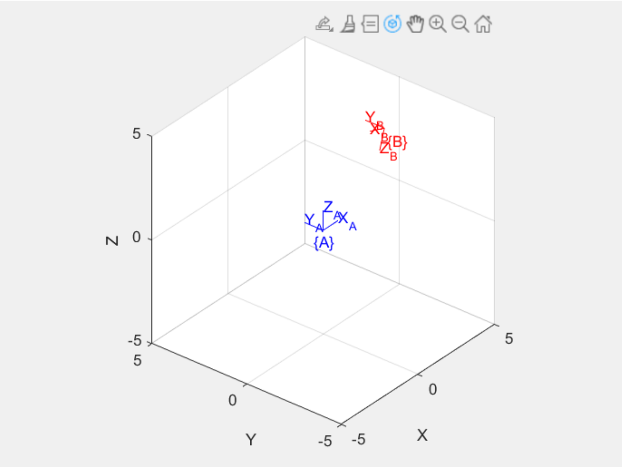
```
>> T=trotx(pi/6)*transl(3,4,5)

T =

    1.0000         0         0    3.0000
         0    0.8660   -0.5000    0.9641
         0    0.5000    0.8660    6.3301
         0         0         0    1.0000

>> R=t2r(T)

R =

    1.0000         0         0
         0    0.8660   -0.5000
         0    0.5000    0.8660
```

## 例题 3.6 矩阵分解 T=r2t(R)

```
>> R=rotz(pi/3)

R =

    0.5000   -0.8660         0
    0.8660    0.5000         0
         0         0    1.0000

>> T=r2t(R)

T =

    0.5000   -0.8660         0         0
    0.8660    0.5000         0         0
         0         0    1.0000         0
         0         0         0    1.0000
```

## 例题 3.7

```
T1=troty(60);
T2=transl(4,0,3);
T=T2*T1;
p1=[2;4;3;1];
Ap1=T*p1;
Tr=inv(T);
Bp1=Tr*p1;
T0=transl(0,0,0);
trplot(T0,'frame','A','color','b');
axis([-5 5 -5 5 -5 5]);
hold on,
tranimate(T0,T,'frame','B','color','r');
```




（2）编写 MATLAB 程序，对教材中例题 2.6（p34），例题 2.7（p36）例题 2.9(p38) 例题 2.11(p40)的内容进行实验验证

**例 2.6** 坐标系 $F$ 沿参考坐标系的 $y$ 轴移动 10 个单位，沿 $z$ 轴移动 5 个单位。求新的坐标系位置。

$$F=\begin{bmatrix} 0.527 & -0.574 & 0.628 & 5 \\ 0.369 & 0.819 & 0.439 & 3 \\ -0.766 & 0 & 0.643 & 8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

```
Told=[0.527,-0.574,0.628,5; 0.369,0.819,0.439,3; -0.766,0,0.643
Tnew=transl(0,10,5);
Fnew=Tnew*Told;
disp(Fnew);
```

```
Told =

    0.5270   -0.5740    0.6280    5.0000
    0.3690    0.8190    0.4390    3.0000
   -0.7660         0    0.6430    8.0000
         0         0         0    1.0000

    0.5270   -0.5740    0.6280    5.0000
    0.3690    0.8190    0.4390   13.0000
   -0.7660         0    0.6430   13.0000
         0         0         0    1.0000
```

**例 2.7** 旋转坐标系中有一点 $p(2,3,4)^T$ 绕参考坐标系 $x$ 轴旋转 90°。求旋转后该点相对于参考坐标系的坐标，并用图形进行验证。

```
p1=[2,3,4]';
R=rotx(90*pi/180);
p2=R*p1;
disp(p2);
```

```
>> Untitled
     2
    -4
     3
```

**例2.9** 在该例中，假定固连在坐标系 $F_{noa}$ 上的点 $p(7,3,1)^T$ 也经历相同变换，但变换按如下不同顺序进行，求出变换后该点相对于参考坐标系的坐标。

1. 绕 $z$ 轴旋转 90°；
2. 接着平移 [4，−3，7]；
3. 接着再绕 $y$ 轴旋转 90°。

```
p1=[7,3,1,1]';
R1=trotz(90*pi/180);
T=transl(4,-3,7);
R2=troty(90*pi/180);
p2=R2*T*R1*p1;
disp(p2);
```

**例2.11** 坐标系 $B$ 先绕参考坐标系 $x$ 轴旋转 90°，然后沿当前坐标系的 $a$ 轴平移 3 英寸，然后再绕参考坐标系 $z$ 轴旋转 90°，最后沿当前坐标系 $o$ 轴平移 5 英寸。

(a) 写出描述该运动的方程。

(b) 求固连在坐标系中的点 $p(1,5,4)^T$ 相对于参考坐标系的最终位置。

```
p1=[1,5,4,1]';
R1=trotz(90*pi/180);
R2=trotx(90*pi/180);
T1=transl(0,0,3);
T2=transl(0,5,0);
p2=R1*R2*T1*T2*p1;
disp(p2);
```

```
>> Untitled
     7
     1
    10
     1
```

4. 第四章 MATLAB Robotics Toolbox 机器人建模

实训内容：

（1）练习所有例题，熟悉相关命令

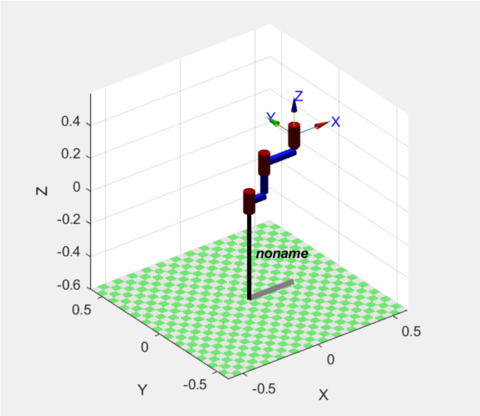
例题 4.1 定义连杆

```
>> L1.a

ans =

     0

>> L1.alpha

ans =

   -1.5708

>> L1.offset

ans =

     0
```

例题 4.2 轴机械臂

```
clear;
clc;
L1=Link([pi/4,0,0,0,0],'modified');
L2=Link([pi/2,0.2,0.1,0,0],'modified');
L3=Link([0,0.1,0.2,0,0],'modified');
robot=SerialLink([L1,L2,L3]);%用定义好的关节建立机器人
robot.display();%显示建立的机器人DH参数
theta=[0 0 0];%6个关节的角度变量值都设为0，可以更改
robot.plot(theta);%显示机器人的图像
```

```
robot =

noname:: 3 axis, RRR, modDH, slowRNE
+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          0|          0|          0|
|  2|         q2|        0.2|        0.1|          0|          0|
|  3|         q3|        0.1|        0.2|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+
```

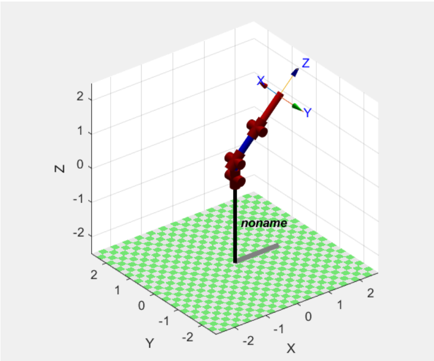



例题 4.3 定义 6 轴机械臂

```
L1=Link('d',0,'a',0,'alpha',pi/2);
L2=Link('d',0,'a',0.5,'alpha',0,'offset',pi/2);
L3=Link('d',0,'a',0,'alpha',pi/2,'offset',pi/4);
L4=Link('d',1,'a',0,'alpha',-pi/2);
L5=Link('d',0,'a',0,'alpha',pi/2);
L6=Link('d',1,'a',0,'alpha',0);
robot=SerialLink([L1,L2,L3,L4,L5,L6]);%用定义好的关节建立机器人
robot.display();%显示建立的机器人DH参数
theta=[0 0 0 0 0 0];%6个关节的角度变量值都设为0，可以更改
robot.plot(theta);%显示机器人的图像
```

noname:: 6 axis, RRRRRR, stdDH, slowRNE

| j | theta | d | a | alpha | offset |
|---|---|---|---|---|---|
| 1 | q1 | 0 | 0 | 1.5708 | 0 |
| 2 | q2 | 0 | 0.5 | 0 | 1.5708 |
| 3 | q3 | 0 | 0 | 1.5708 | 0.785398 |
| 4 | q4 | 1 | 0 | -1.5708 | 0 |
| 5 | q5 | 0 | 0 | 1.5708 | 0 |
| 6 | q6 | 1 | 0 | 0 | 0 |

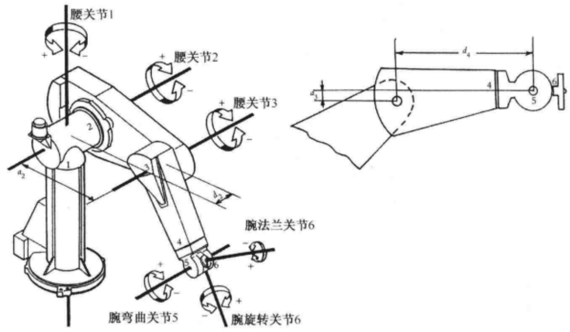



（2）编写 MATLAB 程序，对教材中习题 2.33（P81）机器人建模，并用图展示处建模结果

2.33 对于如图 P. 2. 33 所示的 6 轴机器人 Unimation Puma 562:

- 基于 D-H 表示法建立坐标系。
- 填写参数表。
- 写出所有的 $A$ 矩阵。
- 根据下列数值求 ${}^{R}T_{H}$ 矩阵:

已知具体数据: 基座高度 = 27 in, $d_2 = 6$ in, $a_2 = 15$ in, $a_3 = 1$ in, $d_4 = 18$ in, $\theta_1 = 0°$, $\theta_2 = 45°$, $\theta_3 = 0°$, $\theta_4 = 0°$, $\theta_5 = -45°$, $\theta_6 = 0°$。



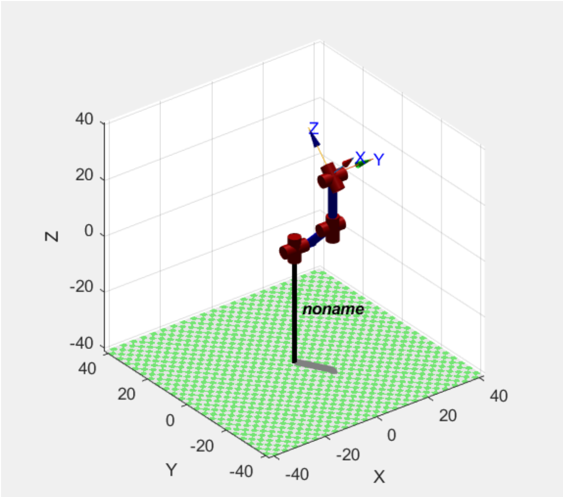

| # | $\theta$ | $d$ | $a$ | $\alpha$ |
|---|---|---|---|---|
| 0−1 | | | | |
| 1−2 | | | | |
| 2−3 | | | | |
| 3−4 | | | | |
| 4−5 | | | | |
| 5−6 | | | | |

图 P. 2. 33 Puma 562 机器人

| # | $\theta$ | D | a | $\alpha$ |
|---|---|---|---|---|
| 0-1 | $\theta_1$+90 | 0 | 0 | −90 |
| 1-2 | $\theta_2$ | 6" | 15" | 0 |
| 2-3 | $\theta_3$ | 0 | 1" | −90 |
| 3-4 | $\theta_4$ | 18" | 0 | 90 |
| 4-5 | $\theta_5$ | 0 | 0 | −90 |
| 5-6 | $\theta_6$ | 0 | 0 | 0 |

```
L1=Link('d',0,'a',0,'alpha',pi/2);
L2=Link('d',6,'a',15,'alpha',0,'offset',pi/2);
L3=Link('d',0,'a',1,'alpha',pi/2,'offset',pi/4);
L4=Link('d',18,'a',0,'alpha',-pi/2);
L5=Link('d',0,'a',0,'alpha',pi/2);
L6=Link('d',1,'a',0,'alpha',0);
robot=SerialLink([L1,L2,L3,L4,L5,L6]);%用定义好的关节建立机器人
robot.display();%显示建立的机器人DH参数
theta=[90*pi/180 45*pi/180 0 0 -45*pi/180 0];%6个关节的角度变量
robot.plot(theta);%显示机器人的图像
```

```
noname:: 6 axis, RRRRRR, stdDH, slowRNE
+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          0|     1.5708|          0|
|  2|         q2|          6|         15|          0|     1.5708|
|  3|         q3|          0|          1|     1.5708|   0.785398|
|  4|         q4|         18|          0|    -1.5708|          0|
|  5|         q5|          0|          0|     1.5708|          0|
|  6|         q6|          1|          0|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+
```




5. 第五章基于 MATLAB Bobotics Toolbox 机器人正逆运动学分析

实训内容

（1）练习本章例题，熟悉相关命令

```
>> mdl_puma560;
>> p560=SerialLink(p560);
>> fkine(p560,qz)


ans =
         1         0         0    0.4521
         0         1         0     -0.15
         0         0         1    0.4318
         0         0         0         1
>> q=[0 -pi/4 -pi/4 0 pi/8 0];
>> T=fkine(p560,q)


T =
    0.3827         0    0.9239    0.7371
         0         1         0   -0.1501
   -0.9239         0    0.3827   -0.3256
         0         0         0         1
>> qi=ikine(p560,T)

qi =

    0.0000   -0.7854   -0.7854    0.0000    0.3927   -0.0000
```

（2）更改 q 的位置，重新利用该方法进行正逆运动学分析

```
>> mdl_puma560;
>> p560=SerialLink(p560);
>> fkine(p560,qz)


ans =
         1         0         0    0.4521
         0         1         0     -0.15
         0         0         1    0.4318
         0         0         0         1
>> q=[0 pi/4 pi/4 0 pi/8 0];
>> T=fkine(p560,q)


T =
   -0.3827         0   -0.9239   -0.1265
         0         1         0     -0.15
    0.9239         0   -0.3827    0.3256
         0         0         0         1
>> qi=ikine(p560,T)

qi =

   -0.0000    0.7854    0.7854    0.0000    0.3927   -0.0000
```

## 6. 第六章基于 MATLAB Robotics Toolbox 机器人轨迹规划

（1）练习本章例题，熟悉相关命令

<1>关节空间轨迹规划

```
%关节空间轨迹规划
clear;
clc;
ML1=Link([0,0.4967,0,0,0],'modified');
ML2=Link([-pi/2,-0.18804,0.2,3*pi/2,0],'modified');
ML3=Link([0,0.17248,0.79876,0,0],'modified');
ML4=Link([0,0.98557,0.25126,3*pi/2,0],'modified');
ML5=Link([0,0,0,pi/2,0],'modified');
ML6=Link([0,0,0,pi/2,0],'modified');
robot=SerialLink([ML1 ML2 ML3 ML4 ML5 ML6],'name','Fanuc M20ia');
%给定末端执行器的初始位置
p1=[0.617222144    0.465154659    -0.63456124    -0.254420286
    -0.727874557    0.031367208    -0.684992502   -1.182407321
    -0.298723039    0.884673523    0.357934776    -0.488241553
    0 0 0 1];
p2=[-0.504697849    -0.863267623    -0.007006569    0.664185871
    -0.599843651    0.356504321    -0.716304589    -0.35718173
    0.620860432    -0.357314539   -0.697752567    2.106929688
    0 0 0 1];
%利用运动学反解 ikine 求解各个关节转角
init_ang=robot.ikine(p1);%使用运动学得带反解的算法计算得到初始的关节角度
targ_ang=robot.ikine(p2);%使用运动学迭代反解的算法计算得到目标关节角度
%利用五次多项式计算关节速度和加速度
step=40;
[q,qd,qdd]=jtraj(init_ang,targ_ang,step);

```

```
%显示机器人姿态随时间的变化
subplot(3,3,[1,4,7]);
robot.plot(q);

%显示机器人的姿态随时间的变化
subplot(3,3,2);
i=1:6;
plot(q(:,i));
title('初始位置 各个关节角度随时间的变化 目标位置');
grid on;
subplot(3,3,5);
i=1:6;
plot(qd(:,i));
title('各个关节角速度随时间的变化');
grid on;
subplot(3,3,8);
i=1:6;
plot(qdd(:,i));
title('各个关节角加速度随时间的变化');
grid on;
%显示末端执行器的位置
subplot(3,3,3);
hold on
grid on
title('末端执行器在三维空间中的位置变化');
for i=1:step
    position=robot.fkine(q(i,:));
plot3(position.t(1),position.t(2),position.t(3),'b','MarkerSize',5);
end

%显示末端执行器的线速度和角速度
subplot(3,3,6);
hold on
grid on
title('末端执行器速度大小随时间的变化');
vel=zero(step,6);
vel_velocity=zeros(step,1);
vel_angular_velocity=zero(step,1);
for i=1:step
    vel(i,:)=robot.jacob0(q(i,:))*qd(i,:)';
    vel_velocity(i)=sqrt(vel(i,1)^2+vel(i,2)^2+vel(i,3)^6);
    vel_angular_velocity(i)=sqrt(vel(i,4)^2+vel(i,5)^2+vel(i,3)^6);
end
x=linspace(1,step,step);
```

```
plot(x,vel_velocity);
subplot(3,3,9);
hold on
grid on
title('末端执行器角速度大小随时间的变化');
x=linspace(1,step,step);
plot(x,vel_angular_velocity);
```

<2>笛卡尔空间轨迹规划

```
clear;
clc;
ML1=Link([0,0.4967,0,0,0],'modified');
ML2=Link([-pi/2,-0.18804,0.2,3*pi/2,0],'modified');
ML3=Link([0,0.17248,0.79876,0,0],'modified');
ML4=Link([0,0.98557,0.25126,3*pi/2,0],'modified');
ML5=Link([0,0,0,pi/2,0],'modified');
ML6=Link([0,0,0,pi/2,0],'modified');
robot=SerialLink([ML1 ML2 ML3 ML4 ML5 ML6],'name','Fanuc M20ia');

%给定末端执行器的初始位置
p1=[0.617222144    0.465154659   -0.63456124   -0.254420286
    -0.727874557   0.031367208   -0.684992502  -1.182407321
    -0.298723039   0.884673523   0.357934776   -0.488241553
    0 0 0 1];
p2=[-0.504697849   -0.863267623   -0.007006569   0.664185871
    -0.599843651   0.356504321   -0.716304589   -0.35718173
    0.620860432    -0.357314539  -0.697752567    2.106929688
    0 0 0 1];
step=40;
Tc=ctraj(p1,p2,step);

%显示机器人姿态随时间的变化
subplot(3,3,[1,4,7]);
q=zeros(step,6);
for i=1:step
    q(i,:)=robot.ikine(Tc(:,:,i));
end
robot.plot(q);

qd=zeros(step-1,6);
for i=2:step
    qd(i,1)=q(i,1)-q(i-1,1);
    qd(i,2)=q(i,2)-q(i-1,2);
```

```
    qd(i,3)=q(i,3)-q(i-1,3);
    qd(i,4)=q(i,4)-q(i-1,4);
    qd(i,5)=q(i,5)-q(i-1,5);
    qd(i,6)=q(i,6)-q(i-1,6);
end


qdd=zeros(step-2,6);
for i=2:step-1
    qdd(i,1)=qd(i,1)-qd(i-1,1);
    qdd(i,2)=qd(i,2)-qd(i-1,2);
    qdd(i,3)=qd(i,3)-qd(i-1,3);
    qdd(i,4)=qd(i,4)-qd(i-1,4);
    qdd(i,5)=qd(i,5)-qd(i-1,5);
    qdd(i,6)=qd(i,6)-qd(i-1,6);
end

%显示机器人关机运动状态
subplot(3,3,2);
i=1:6;
plot(q(:,i));
title('初始位置 各个关节角度随时间的变化 目标位置');
grid on;
subplot(3,3,5);
i=1:6;
plot(qd(:,i));
title('各个关节角速度随时间的变化');
grid on;
subplot(3,3,8);
i=1:6;
plot(qdd(:,i));
title('各个关节角加速度随时间的变化');
grid on;

%显示末端执行器的位置
subplot(3,3,3);
hold on
grid on
title('末端执行器在三维空间中的位置变化');
for i=1:step
    position=robot.fkine(q(i,:));
plot3(position.t(1),position.t(2),position.t(3),'b','MarkerSize',5);
end
```

```
%显示末端执行器的线速度和角速度
subplot(3,3,6);
hold on
grid on
title('末端执行器速度大小随时间的变化');
vel_velocity=zeros(step,1);
for i=2:step
    vel_velocity(i)=sqrt(Tc(1,4,i)-Tc(1,4,i-1)^2+(Tc(2,4,i)-Tc(2,4,i))^2+(Tc(3,4,i)-Tc(3,4,i-1))^2);
end
x=linspace(1,step,step);
plot(x,vel_velocity);

subplot(3,3,9);
hold on
grid on
title('末端执行器角速度大小随时间的变化');
vel_acceleration=zeros(step-2,1);
for i=3:step
    vel_acceleration(i-2)=sqrt((Tc(1,4,i)-Tc(1,4,i-1)-(Tc(1,4,i-1)-Tc(1,4,i-2)))^2+(Tc(2,4,i)-Tc(2,4,i-1)-(Tc(2,4,i-1)-Tc(2,4,i-2)))^2+(Tc(3,4,i)-Tc(3,4,i-1)-(Tc(3,4,i-1)-Tc(3,4,i-2)))^2);
end
x=linspace(1,step-2,step-2);
plot(x,vel_acceleration);
```