|  |  |
| --- | --- |
| **分类号 密级** | 校徽 |
| **UDC** |
|  |

**现代控制理论课程设计**

基于现代控制理论的单柔性关节机器人建模与控制

**学生姓名** 刘运涛 **学号** 18071001041

**指导教师** 任凭

**院、系、中心** 工程学院自动化及测控系

**专业年级** 2018级自动化

**验收日期** 2021年 7月 6日

**中 国 海 洋 大 学**

基于现代控制理论的单柔性关节机器人建模与控制

摘 要

|  |
| --- |
| 本文基于现代控制理论，以单柔性关节机器人的微分方程为基础，构建状态空间方程。通过计算得到状态转移矩阵的解析式，利用MATLAB对比与数值计算结果，绘制时域响应曲线。观察传递函数计算过程和结果，判断系统的零极点对消情况，以检查系统的能控能观性。构建能控能观判定矩阵验证传递函数的结果，并重构状态方程为能控标准型和能观标准型。基于能控标准型构建状态全反馈和最优控制反馈器，并与传统PD控制方式比较，设计参考输入。基于能观标准型设计状态观测器，最后再次利用李雅普诺夫稳定性判据再次判断系统稳定性。  关键词：状态空间；时域响应；能控能观性；李雅普诺夫稳定性；最优控制 |

**Modeling and control of single flexible joint robot based on modern control theory**

**Abstract**

|  |
| --- |
| Based on the modern control theory, this paper constructs the state space equation based on the differential equation of the single flexible joint robot. The analytical expression of the state transition matrix was obtained through calculation, and the time-domain response curve was drawn by using MATLAB to compare with the numerical calculation results. To check the controllability and observability of the system, we observe the calculation process and results of the transfer function and judge the pole-zero cancellation of the system. The controllable and observable matrices were constructed to verify the results of the transfer function, and the state equation was reconstructed into the controllable and observable standard forms. The state full feedback and optimal control feedbacks were constructed based on the controllable standard type, and the reference input was designed by comparing with the traditional PD control mode. Finally, Lyapunov stability criterion is used to judge the system stability again.  Keywords: State space; Time domain response; Controllability and observability；Lyapunov stability；Optimal control |

目 录

摘 要 I

Abstract II

目 录 III

**1** 背景综述 1

**1.1** 工业机器人减速器 1

**1.2** 串联弹性驱动器 1

**1.3** 参数K选择范围 2

**1.4** 伺服电机 2

**2** 微分方程与状态空间方程建模 3

**2.1** 微分方程 3

**2.2** 状态空间方程 3

**3** 状态转移矩阵 4

**3.1** 齐次解析解 4

**3.2** 非齐次解析解 4

**4** 时域响应仿真 5

**4.1** 解析解 5

**4.1.1** 齐次解析解 5

**4.1.2** 非齐次解析解 5

**4.2** 基于ode45方法的数值解 6

**4.2.1** 齐次数值解 6

**4.2.2** 非齐次数值解 6

**5** 传递函数 7

**6** 能控性与能观性 8

**6.1** 基于解析字母的能控性与能观性判定 8

**6.2** 基于参数数值的能控性和能观性判定 8

**6.3** 基于解析字母的能控能观标准型 8

**6.3.1** 能控标准型 8

**6.3.2** 能观标准型 9

**6.4** 基于参数数值的能控能观标准型 10

**7** 经典控制理论PD控制器设计 10

**7.1** 反馈电机转角 12

**7.2** 反馈负载转角 14

**8** 现代控制理论状态反馈（极点配置）控制器设计 16

**9** 参考输入 18

**10** 状态观测器 18

**11** 李亚普诺夫稳定性 19

**12** 最优控制器 20

参考文献 22

附录 23

# 背景综述

## 工业机器人减速器

随着工业的不断发展，世界各国对于工业机器人的需求也日益增加，在德国、美国提出工业4.0和工业互联网的背景下，中国也提出了“中国制造2025”的发展战略，预示着我国制造业将一步步向智能化迈进。在我国劳动力成本不断增加、传统制造业失去优势的情况下，工业机器人的发展与应用，是我国制造业转型升级的重要武器。为了保证生产质量，提高企业生产效率，实现自动化生产，工业机器人被广泛应用于电子、物流、化工等各个工业领域之中。

工业机器人最重要的基础部件：关节，也是运动的核心部件：精密减速机。精密减速器作为一种精密的动力传达机构，其利用齿轮的速度转换器，将电机的回转数减速到所要的回转数，并得到较大转矩的装置，从而降低转速，增加转矩。

按结构不同可以分为五类：谐波齿轮减速机、RV（旋转矢量）减速机、摆线针轮减速机、精密行星齿轮减速机和滤波齿轮减速机。大量应用在工业机器人关节上的减速器为 RV减速器和谐波减速器两种，行星减速机使用较少，常见的是用在DELTA上面，也少量使用在SCARA上，并且使用的都是低背隙的高端行星。

一般将RV减速器放在机座、大臂、肩部等载荷较大的位置，谐波减速器放在小臂、腕部或手部等载荷较小的位置。其中谐波减速器是通过柔性轴承使柔轮产生可控弹性变形，与刚性齿轮相啮合来传递运动和力，但柔轮会周期性发生变形，产生交变应力，易于产生疲劳破坏影响其寿命，其运动精度会随着使用时间的增长而降低。而RV减速器是由传统的摆线针轮减速器发展起来的，采用的一齿差或是少齿差传动形式，它具有结构紧凑，寿命长，传动效率高，传动精度高等一系列优点，相比于谐波减速器,其疲劳强度、刚度、寿命和回转精度都高很多。故一台工业机器人所使用的精密减速器中六到八成以上是RV减速器。

## 串联弹性驱动器

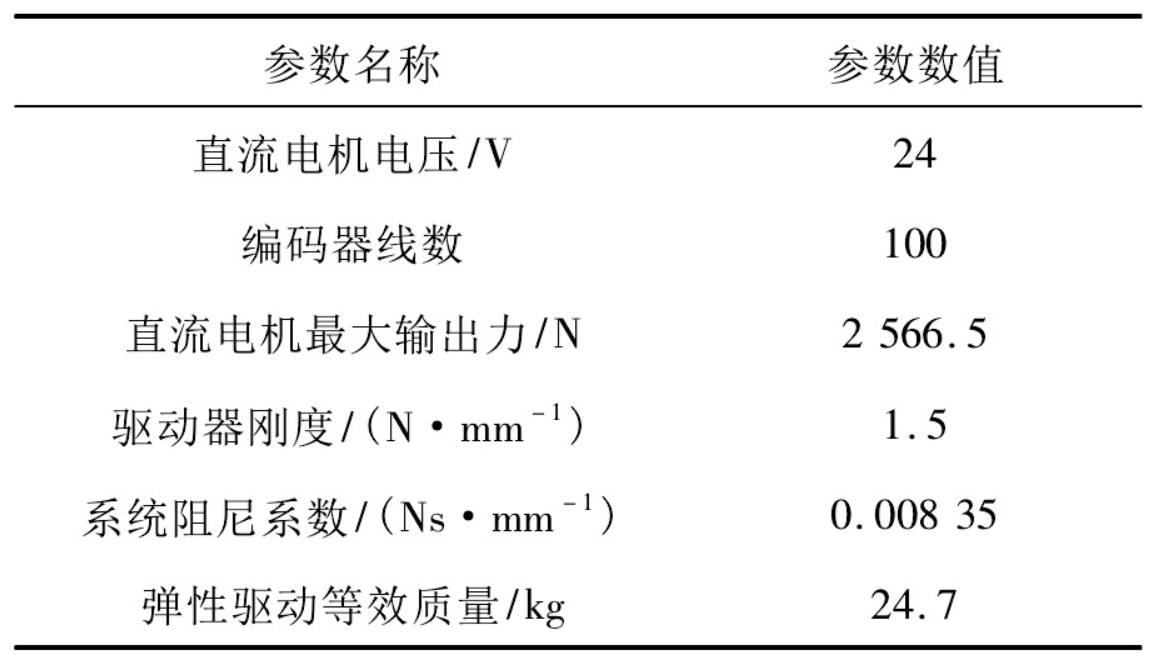
驱动装置作为机器人系统运动和驱动力的输入单元，其性能优劣对机器人系统的整体性能具有很大影响。传统机器人多采用刚性驱动器，能够精确的传递运动，而人或其它动物的驱动装置是肌肉组织，具有柔顺性,导致人或其它动物在运动、安全与能源效率等方面远远超过传统机器人系统，因此刚性驱动器已经阻碍高性能机器人的发展。为了解决这个问题,研究人员提出一种模拟生物肌肉的柔顺性驱动器—串联弹性驱动器(series elastic actuator-SEA)，其主要是在驱动装置与末端执行器之间增加具有弹性和阻尼特性的装置，解除驱动装置和末端执行器之间的耦合关系，同时使串联弹性驱动器输出力与其形变量、相对速度成一定关系。与传统的刚性驱动器相比，SEA具有被动柔顺性、阻抗低、抗冲击、力感知等优点，已经广泛地应用到各种场合。

在1995年，Pratt等人首先提出SEA，其由驱动装置、减速装置、弹性元件、末端执行器以及位置传感器串联而成,具有弹性特性，应用到腿足式机器人当中，实现力控制。在此基础上，研究人员依据应用场景需求对其弹性和阻尼特性进行研究,进而分别提出串联阻尼驱动器[[[1]](#endnote-1)]和串联粘弹性驱动器[1]。到目前为止，研究人员已经将各种SEA广泛地应用到外骨骼机器人、腿足式机器人、假肢机器人、仿人型机器人等领域，并为了适应不同的应用需求，发展出了各种特定的SEA。

## 参数K选择范围

串联弹性驱动器SEA的参数可简化为负载惯量、负载阻尼、电机惯量、电机阻尼、转动弹性系数等，经过文献查阅，可大致可确定其部分参数范围。文献《串联弹性驱动器力驱动力学模型稳定性和分析》[2]中对一种串联弹性驱动器的参数做了详细研究，如下表所示。

表1-1 一种串联弹性驱动器参数



## 伺服电机

伺服电机（servo motor）是指在伺服系统中控制机械元件运转的发动机，是一种补助马达间接变速装置。伺服电机可使控制速度，位置精度非常准确，可以将电压信号转化为转矩和转速以驱动控制对象。伺服电机转子转速受输入信号控制，并能快速反应，在自动控制系统中，用作执行元件，且具有机电时间常数小、线性度高、始动电压等特性，可把所收到的电信号转换成电动机轴上的角位移或角速度输出。分为直流和交流伺服电动机两大类，其主要特点是，当信号电压为零时无自转现象，转速随着转矩的增加而匀速下降。

“伺服电机”可以理解为绝对服从控制信号指挥的电机。在工业机器人中，伺服电机是工业机器人的动力系统，一般安装在机器人的“关节”处，是机器人运动的“心脏”，其功能是将电信号转换成转轴的角位移或角速度。

机器人的关节驱动离不开伺服系统，关节越多，机器人的柔性和精准度越高，所要使用的伺服电机的数量就越多。机器人对伺服系统的要求较高，必须满足快速响应、高起动转矩、动转矩惯量比大、调速范围宽，要适应机器人的形体做到体积小、重量轻、加减速运行等条件，且需要高可靠性和稳定性。机器人用伺服电机要求控制器与伺服之间的总线通讯速度快；伺服的精度高；另外对基础材料有加工要求。特别是像机器人末端执行器（手爪）应采用体积、质量尽可能小的电机，尤其是要求快速响应时，伺服电动机必须具有较高的可靠性和稳定性，能经受得起苛刻的运行条件，可进行十分频繁的正反向和加减速运行，并能在短时间内承受过载。

伺服电机主要可分为交流伺服系统和直流伺服两大类。交流伺服电动机驱动器因其具有转矩转动惯量比高、无电刷及换向火花等优点，在工业机器人中得到广泛应用。

# 微分方程与状态空间方程建模

## 微分方程

对单柔性关节机器人，设定参数如表2-1所示。

表2-1 模型参数

|  |  |  |  |
| --- | --- | --- | --- |
| 负载惯量 |  | 电机惯量 |  |
| 负载阻尼 |  | 电机阻尼 |  |
| 输入扭矩 |  | 转动弹性系数 |  |
| 负载杆转角 |  | 电机转角 |  |

根据已知模型，如图2-2所示，构建微分方程组。

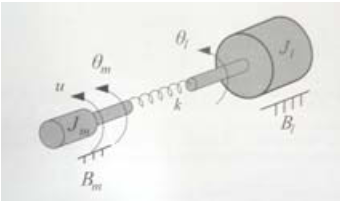


图2-2 单柔性关节机器人模型

## 状态空间方程

选取状态变量如下：

列写状态方程:

列写矩阵形式下的状态方程：

其中，，。

# 状态转移矩阵

## 齐次解析解

首先利用inv函数求，再利用ilaplace函数求状态转移矩阵。由于要求实域下的指数函数、三角函数，保留小数点后三位有效数字，利用digits设置精度为4，随后使用vpa函数就能保证始终能保留4位有效数字。 另外，由于拉普拉斯逆变换后得到的式子存在虚部，在进行时域响应绘图时，虚部无法被展示，因此采用real函数仅获取实部用于绘图。虽然缺失了虚部，但是齐次解析解与数值解得到的图像几乎没有差别，我们完全可以认为得到的实部解析解能有效反映该状态方程的时域解。齐次解析解完整形式见附录。

## 非齐次解析解

在已经得到齐次解析解的情况下，计算由控制作用下的受控项。首先将转移项中的t替换为，即将转变为。由于本设计中的输入为单位阶跃函数，因此直接将\*B后对进行0到t的积分，最后得到的结果即为受控项。最后，把受控项（带初始状态）和转移项结合，得到的结果即为非齐次解析解。

同样，非齐次解中存在虚部，我们在绘图时仍使用real函数取实部，忽略掉虚部。非齐次解析解完整形式见附录。

# 时域响应仿真

## 解析解

### 齐次解析解

利用得到的齐次解析解的实部，在给定初始条件:

绘制输出y及yd，ydd，yddd，结果如图4-1所示。

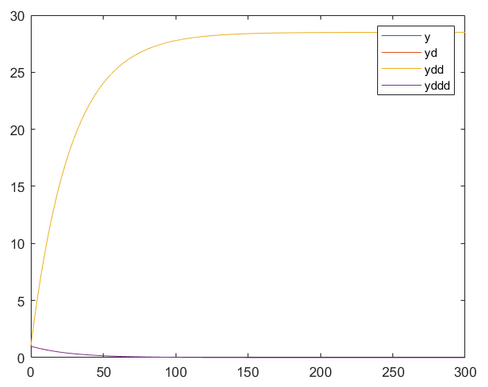


图4-1 齐次解析解图像

### 非齐次解析解

利用得到的非齐次解析解的实部，在给定初始条件:

绘制输出y及yd，ydd，yddd，结果如图4-2所示。

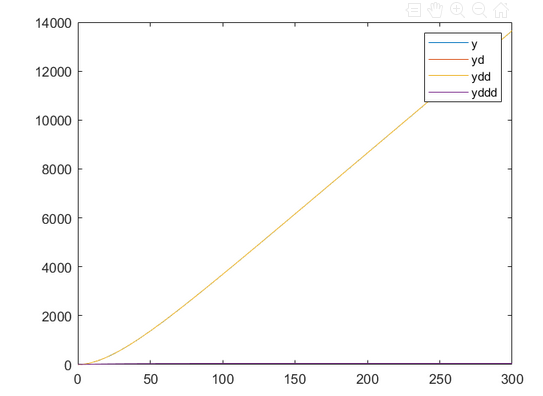


图4-2 非齐次解析解图像

## 基于ode45方法的数值解

### 齐次数值解

基于ode45，设置函数lin\_pend\_dot中，随后设置时间ti到tfl时间为300s，设置初始状态，调用ode45，绘制输出y及yd，ydd，yddd，结果如图4-3所示。

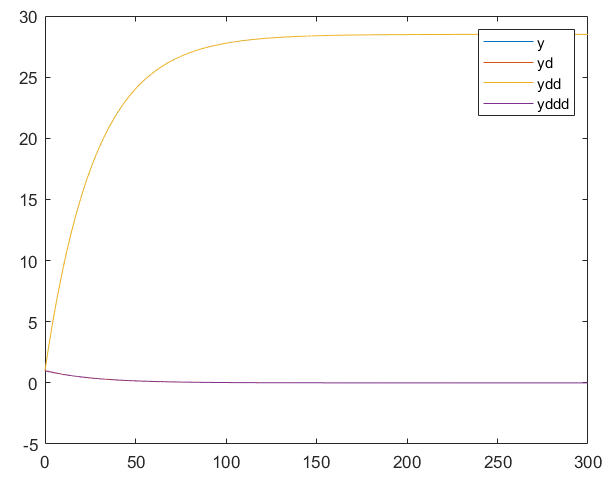


图4-3 齐次数值解图像

此时y与yd的图像重合，但最后y仍能够收敛，保证系统趋于稳定，显然，**系统内部稳定**。

并且，通过观察数值解和解析解绘制得到的图像，明显可以看出两个图像基本完全重合，存在的极细微的差别源自于解析解忽略了虚部或者MATLAB内部数值计算方法。

### 非齐次数值解

基于ode45，设置函数lin\_pend\_dot中，其中u为单位阶跃函数，随后设置时间ti到tfl时间为300s，设置初始状态，调用ode45，绘制输出y及yd，ydd，yddd，结果如图4-4所示。

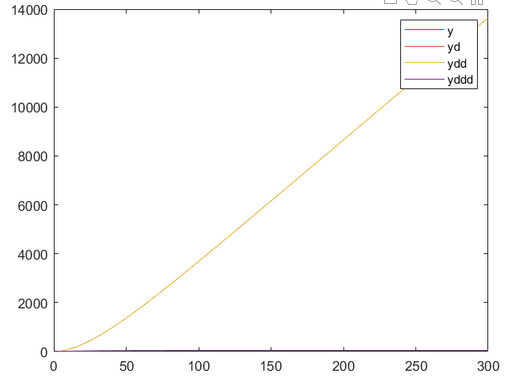


图4-4 非齐次数值解图像

此时y与yd的图像重合，y最后不能够收敛，系统不稳定，并且非齐次解的数值解结果与解析解几乎相同。因此同样可得，**系统输入输出不稳定**。

# 传递函数

利用MATLAB的tf2ss和tf函数，可直接将状态空间转化为传函，结果为：

利用zpk函数将G（s）转化为零极点模型，用以观察是否存在零极点对消现象，新的G（s）形式为：

利用状态空间求传函的方法，计算方法为：

首先计算，根据其结果可得知，状态方程中未出现零极点对消现象；随后计算，同样根据结果可以得知，输出方程中未出现零极点对消现象。显然，不存在零极点对消的现象，系统能控能观，这与后面的能控能观判定结果吻合。最后观察计算得到的传函，与式（5，1）结果完全基本相同，只是分母缺少一个极小的，但整体结果（稳定性，能控/能观性）完全不受影响。

另附和计算结果如表5-1所示。

表5-1 和计算结果

|  |  |
| --- | --- |
|  |  |
|  |  |

# 能控性与能观性

## 基于解析字母的能控性与能观性判定

利用syms创建符号型变量，构建Qc和Qg矩阵，利用det函数直接得到Qc和Qg的行列式的值分别为：

显然系统能控能观的条件是，，即，。

## 基于参数数值的能控性和能观性判定

直接将状态空间各个参数值赋给基于解析字母的能控能观的计算结果可得：

显然Qc，Qg矩阵的行列式结果均不等于0，系统能控能观。

## 基于解析字母的能控能观标准型

### 能控标准型

对于能控标准型，构建Qc矩阵，利用inv函数求Qc的逆矩阵，随后根据如下步骤计算能控规范型：

（1）计算P矩阵：

（2）计算能控规范型参数：

（3）列写能控规范型状态方程：

利用MATLAB直接计算得出和结果如表6-1所示。

表6-1 和计算结果

|  |  |
| --- | --- |
|  |  |
|  |  |

### 能观标准型

对于能观标准型，构建Qg矩阵，利用inv函数求Qg的逆矩阵，随后根据如下步骤计算能观规范型：

（1）计算T矩阵：

（2）计算能观规范型参数：

（3）列写能观规范型状态方程：

利用MATLAB直接计算得出结果如表6-2所示。

表6-2 计算结果

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

## 基于参数数值的能控能观标准型

直接将状态空间各个参数值赋给基于解析字母的能控能观标准型的计算结果可得如表6-3所示。

表6-3 基于参数数值的能控能观标准型

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| 能控标准型 |  |  |  |
| 能观标准型 |  |  |  |

# 经典控制理论PD控制器设计

根据公式（2，1）（2，2），经过拉普拉斯逆变换后可得：

带入参数可得：

化简可得传函为：

因此，初始系统的系统框图如下所示：

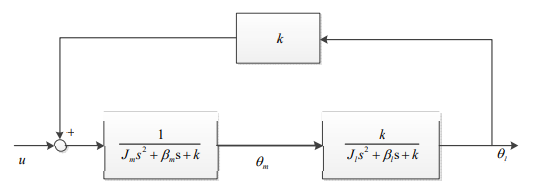


图7-1 系统框图

计算其特征多项式：

分析特征多项式，当阻尼为零时，特征多项式变为:

显然在原点和虚轴上存在重根，系统在理论上临界稳定，绘制其根轨迹入下所示。

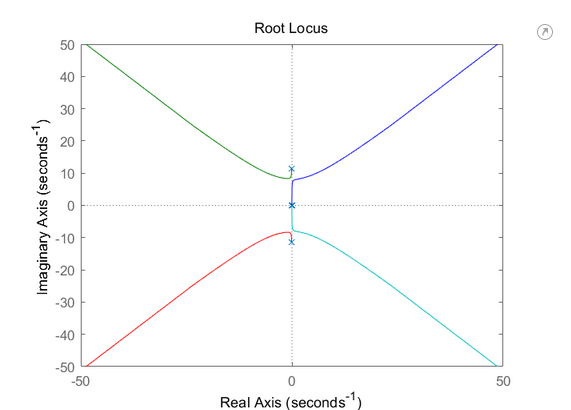


图7-2 系统根轨迹图像

## 反馈电机转角

增加PD控制器，利用反馈电机转角构建反馈，改善系统性能。对于PD调节器，令，，固定 ，利用根轨迹和时域响应判断控制效果，新的系统框图如下所示。

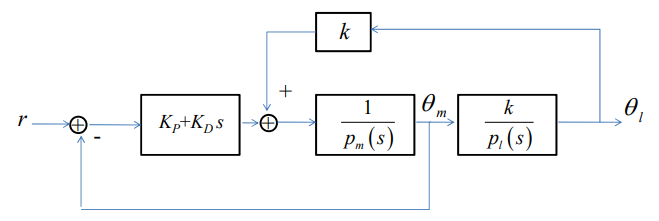


图7-2 带PD控制器的以反馈电机转角为系统反馈的系统框图

将系统框图化简，可得到系统的开环传递函数如下：



选择PD控制器的参数为：KD=0.0026，KP=0.0052，构建模型，绘制其根轨迹如下所示。

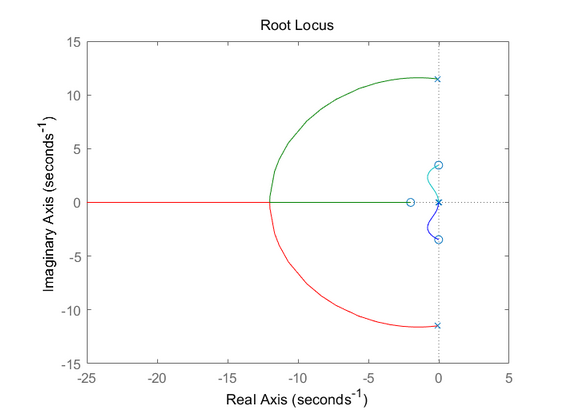


图7-3 带PD控制器的以反馈电机转角为系统反馈的根轨迹图像

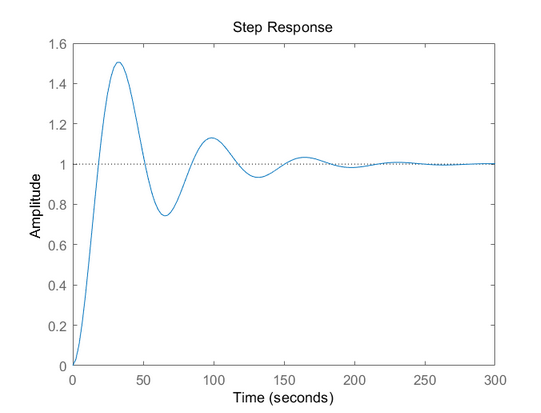


图7-4 带PD控制器的以反馈电机转角为系统反馈的阶跃响应图像

根据根轨迹图像可以看出，对于KD任何值都能令系统稳定，较大的值相对稳定性更好；从阶跃响应可以看出，此时使用PD控制器存在较大的超调，并且调节时间也较长，考虑改变KP的值，选择KP=20，再次获取其时域响应如下所示。增大KP后，有效缩短了调节时间，但超调量有所增大，总体上看系统的响应效果更佳。

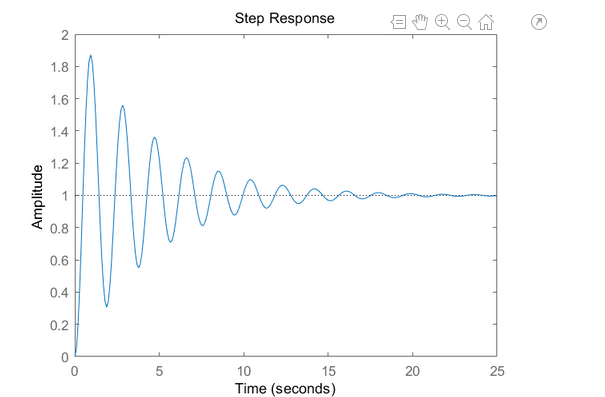


图7-5 带PD控制器的增大Kd以反馈负载转角为系统反馈的阶跃响应图像

## 反馈负载转角

增加PD控制器，利用反馈负载转角构建反馈，改善系统性能。对于PD调节器，令，，固定 ，利用根轨迹和时域响应判断控制效果，新的系统框图如下所示。

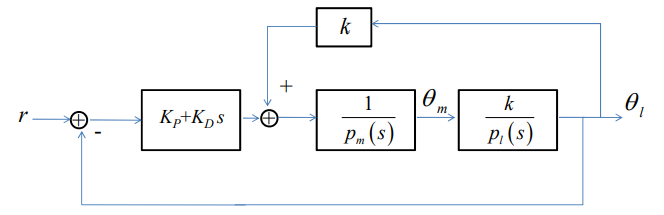


图7-6 带PD控制器的以反馈负载转角为系统反馈的系统框图

选择PD控制器的参数为：KD=0.0009，KP= 0.0018，构建模型，绘制其根轨迹如下所示。

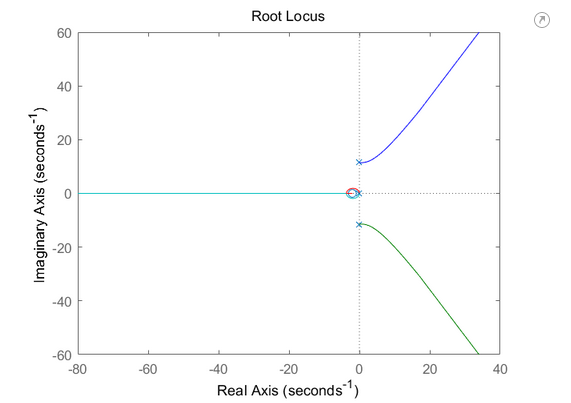


图7-7 带PD控制器的以反馈负载转角为系统反馈的根轨迹图像

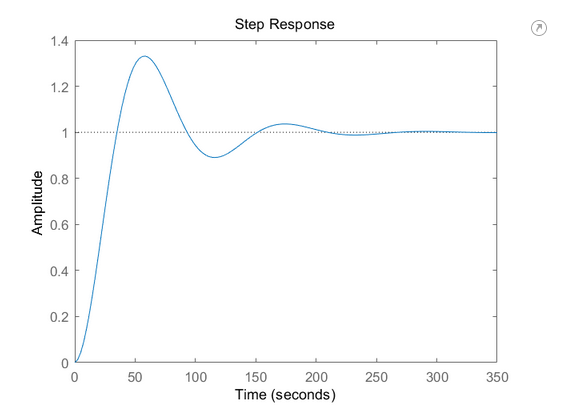


图7-8 带PD控制器的以反馈负载转角为系统反馈的阶跃响应图像

根据根轨迹图像可以看出：对于KD ,任何值都能令系统稳定，较大的值相对稳定性更好；从阶跃响应也可以看出，此时使用PD控制器存在较大的超调，并且调节时间也较长，考虑使用PID调节器，选择PID的参数为：

再次获取其时域响应如下所示。加入积分I后，有效改善了超调，但调节时间有所增长，总体上看系统的响应效果更佳。

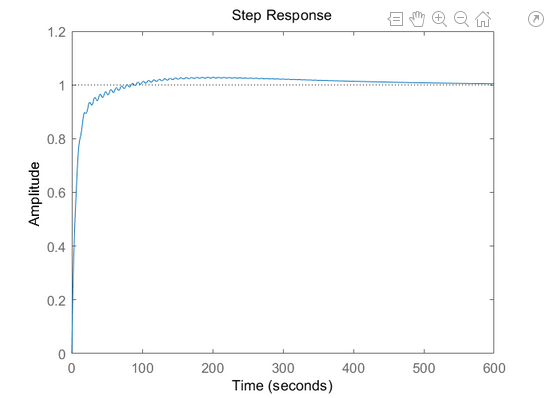


图7-9 带PID控制器的以反馈负载转角为系统反馈的阶跃响应图像

为了对比方便，选择一组不稳定的极点，将PD控制器的参数为：KD=0.2，KP= 0.4，构建模型，其不稳定的阶跃响应图像如下所示。

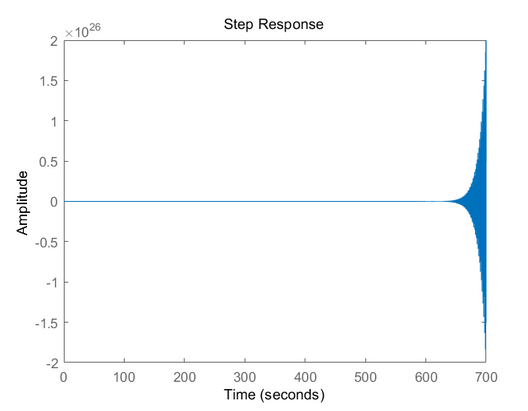


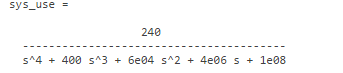
图7-10 带PD控制器的以反馈负载转角为系统反馈的不稳定的阶跃响应图像

# 现代控制理论状态反馈（极点配置）控制器设计

对单柔性关节机器人设计状态反馈，首先判定其能控性，根据第六节的结果可知，本设计的模型具备能控性，因此可以利用极点配置法配置状态反馈。能控标准型结果如表6-3所示，此处不再展示。

由于当系统只拥有纯虚根的时候响应才能达到无超调，所以设定系统极点时，应使其虚部尽量接近于零，这样的系统才可能够达到快速响应无超调的目的。对于反馈矩阵K的求取，可以直接利用MATLAB自带的K=acker(A,B,M)，函数，利用Ackermann公式，求解状态反馈阵K,其中输入的参数A、B为系统的状态空间模型矩阵，向量M中是期望的闭环极点位置，返回值是增益向量。

本次设计选择的极点为M=[-100; -100; -100 ; -100 ]，经计算得到状态反馈阵K为[4.1367e5 0.1664e5 0.0029e5 0.002e5]，将其带入原系统，计算A-BK，新的传函如下所示：



绘制阶跃响应曲线结果如下所示。

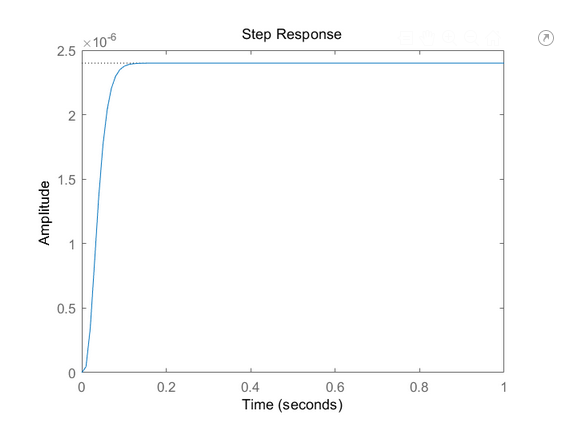


图8-1 带状态反馈的系统阶跃响应图像

利用ode45数值解检验当初始状态不在稳定状态时，观察是否能够回到稳定状态，设置初始状态为X0=[1 1 1 1]时，绘制图像如下所示。

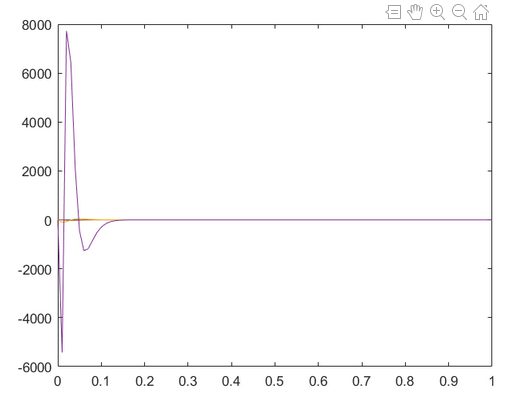


图8-2 状态反馈的系统回到稳定状态的图像

显然，在初始状态X0=[1 1 1 1]，经过全状态反馈控制，系统可以回到原点，即平衡状态。

# 参考输入

在第八节极点配置法设计的状态反馈控制器基础上，设计参考输入v，使负载角稳定在0.41弧度，所以输出值应为0.41rad。经过对参数的试凑，最后得到单位阶跃响应的输出结果为0.41，如下所示。

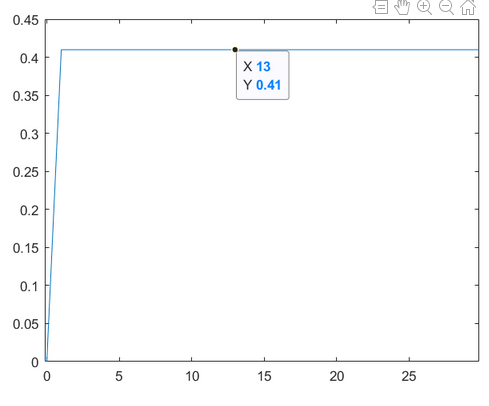
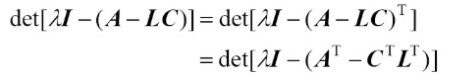


图9-1 稳定在0.41的阶跃响应曲线

# 状态观测器

倘若状态不能直接反馈，则需要采用极点配置法设计全维状态观测器，观测器极点位置应当比控制器极点位置距离虚轴更远。由于极点离虚轴更远是指的发挥主要作用的极点，因此在极点配置时优先考虑离虚轴更远的一组极点。

在MATLAB中，实际上计算状态观测器仍然可以用acker函数，仅仅是在输入参数将原来的A，B变成A'，C'即可。那是由于sI-(A-LC)进行转置后行列式是不变的，所以可以转置为sI-(A'-C'L')L'就相当于K，所以这里要求的L阵，只要转置一下就可以了。（L矩阵即为教材上状态观测器的G矩阵）



状态反馈后的极点为s=-100，观测器极点位置应当比控制器极点位置距离虚轴更远，所以本次设计选择的第一组极点为P1=[-120; -130; - 140; -150 ]，经计算得到状态观测阵L1为[0.001;0.0109;0.0807;2.6050]；第二组极点为P2=[-300; -310; -320 ; -330 ]，经计算得到状态观测阵L2为[0;0.0059;0.1038;8.1038].将其带入原系统，计算A-BL，借助ode45求响应曲线如下所示。

表10-1 两组不同极点对应的响应图像

|  |  |
| --- | --- |
|  |  |
| 第一组极点对应的阶跃响应图像 | 第二组极点对应的阶跃响应图像 |

观察两组响应的图像可得，实际上在单位阶跃输入的情况下，由于状态反馈时配置的极点已经让系统的稳定性可以得到绝对的保障（4个s=-100）,因此在此处设置的观测器的极点还需要更加靠左，意味着观测器的稳定程度还会进一步提高，因此两组极点虽然相差100左右，但是实际上两组单位阶跃响应几乎可以说是完全相同。

# 李亚普诺夫稳定性

基于单柔性关节机器人的参数，证明A-BK系统的李雅普诺夫稳定性。利用MATLAB自带的lyap函数，只需要给定A和Q矩阵的参数，可以直接解出P和D。基于线性定常连续系统渐进稳定性的判别方法：

线性定常系统：

①渐近稳定的充要条件: A的特征值全部在左半开平面内;

②渐近稳定的充要条件:对任意正定阵Q ,存在正定阵P满足李雅普诺夫方程：

给定Q矩阵为单位矩阵，利用lyap函数解得：

表11-1 求解结果

|  |  |
| --- | --- |
|  |  |
| 正定矩阵P | 特征值D |

显然P矩阵是正定矩阵，根据李雅普诺夫稳定充要条件②可得，单柔性关节机器人系统渐近稳定。同样通过观察特征值D也得发现，此时A的特征值全部在左半平面，根据李雅普诺夫稳定充要条件①可得，同样单柔性关节机器人系统渐近稳定。

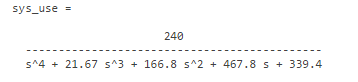
# 最优控制器

LQR控制器是一种常用的控制器，相比较直接设置特征值的控制器来说有很多优点。一般来说，我们需要从以下这几个维度来评估控制器：超调量、使系统达到稳定的时间、系统稳定后误差是否收敛。LQR控制器可以通过设置一个代价函数，将设计控制器转化为一个优化问题，通过这种方法可以直接影响输入输出误差以及控制量，从而达到最优控制。

MATLAB中自带的lpr函数可以在给定Q和R的情况下，直接得出最优反馈增益矩阵K，唯一正定解P（若矩阵A-BK是稳定矩阵，则总有正定解P存在）和矩阵A-BK的特征值E（直接观察极点位置判断稳定性）。其中Q为给定的半正定实对称常数矩阵，Q越大收敛速度越快；R为给定的正定实对称常数矩阵，R越大收敛效果越好。

在给定Q为4阶单位矩阵，R为1时，计算得到最优反馈增益矩阵

K1=[-0.3038 0.8538 1.7180 1.0725]，将其带入原系统后得到的新系统的传函如下所示:



其单位阶跃响应如下所示。

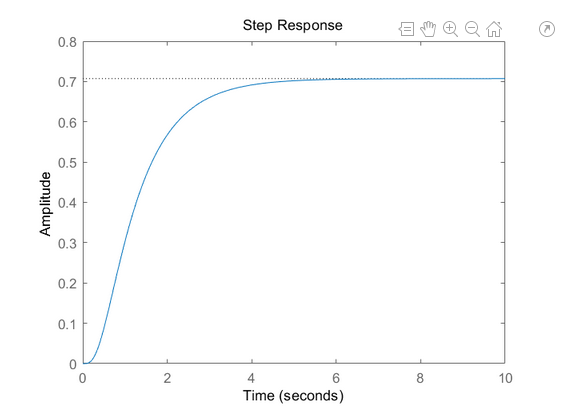


图12-1 最优状态反馈的系统阶跃响应图像

可以看到，在加入最优反馈矩阵K1后，实现了系统快速无超调。利用ode45数值解检验当初始状态不在稳定状态时，观察是否能够回到稳定状态，设置初始状态为X0=[1 1 1 1]时，绘制图像如下所示。

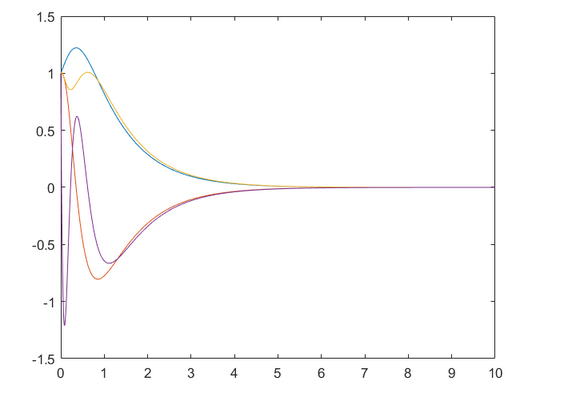


图12-2 最优控制状态反馈的系统回到稳定状态的图像

显然，在初始状态X0=[1 1 1 1]，经过全状态反馈控制，系统可以回到原点，即平衡状态。

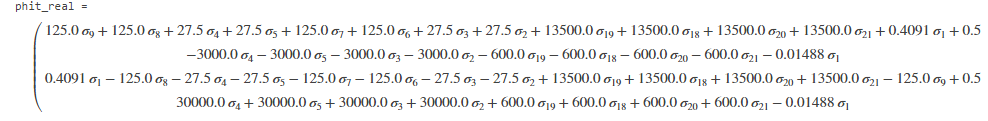
参考文献

1. 孙宁,程龙.串联弹性驱动器设计、建模及在机器人上的应用[J].自动化学报,2020,2(10):1-2.
2. 马洪文,王立权,赵朋,等.串联弹性驱动器力驱动力学模型和稳定性分析[J].哈尔滨工程大学学报,2012,33(11):5.

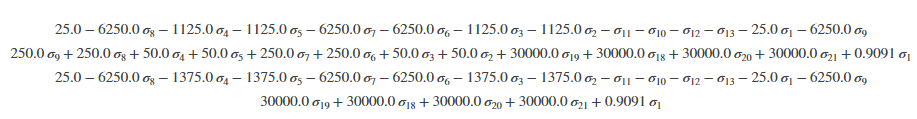
附录

（1）齐次解析解：

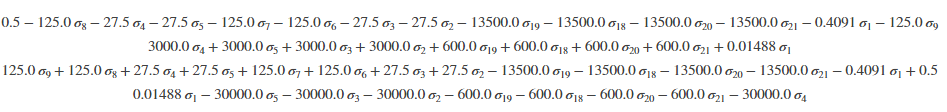
第一列：



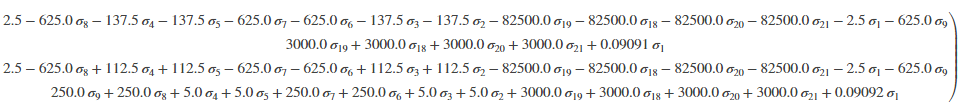
第二列：



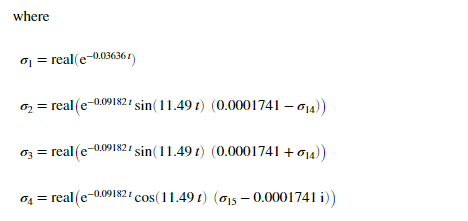
第三列：

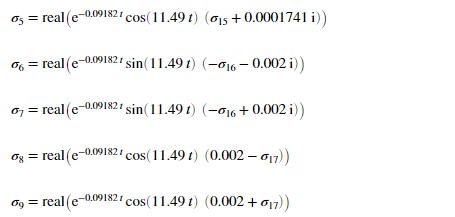


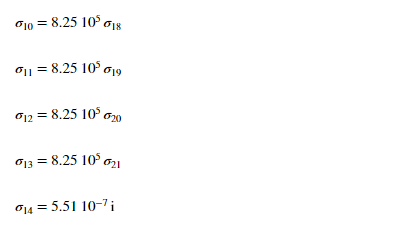
第四列：

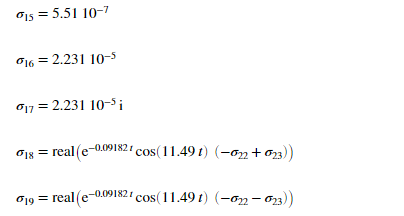


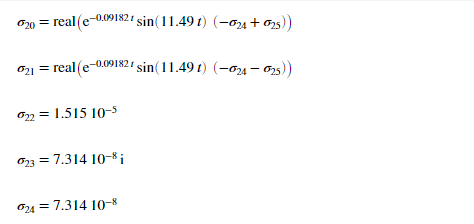
部分参数：





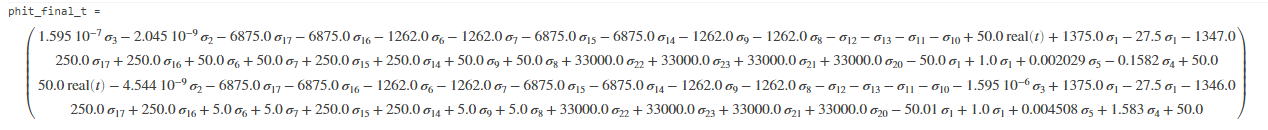


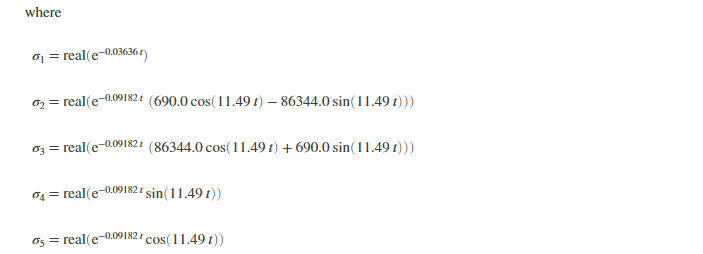


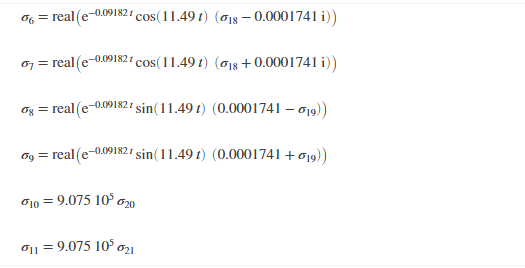


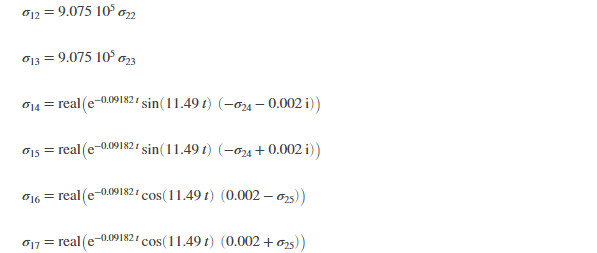


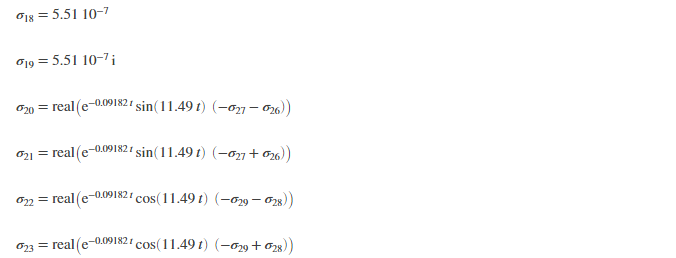
（2）非齐次解析解：

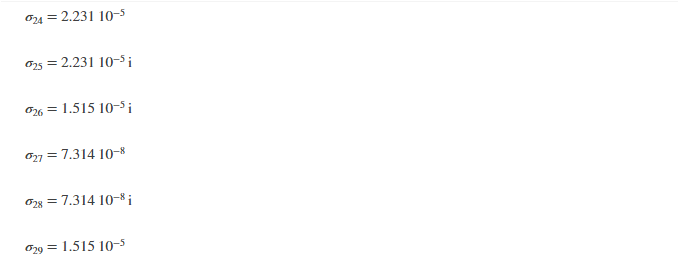












（3）程序

clear all

syms s tao t

global k Jl Jm Bl Bm K K1 k1 k2 k3 k4 k1\_1 k2\_1 k3\_1 k4\_1 L1 L2

k = 6; Jl= 0.5; Jm= 0.05; Bl= 0.01; Bm= 0.01;

[x1 ,x2, x3, x4]=deal(1);

ti = 0;

tfl = 300;

A=[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm];

B=[0;0;0;1/Jm];

C=[1 0 0 0];

D=zeros(1);

I=eye(4);

digits(4);

E=s\*I-A;%求状态转移矩阵

E\_det=det(E);

H=collect(inv(E));%inv求逆，collect

E\_inv=inv(E);

phit=vpa(ilaplace(collect(inv(E))));%损失精度

% phit\_test=simplify(expm(A\*t))

%齐次解析解

phit\_real=simplify(vpa(real((ilaplace(inv(E)))),4)); %ilaplace拉普拉斯逆变换

% rewrite(phit\_real,'cos')

phit\_real\_plot=vpa(phit\_real\*[x1 ;x2; x3; x4])

% [v,d]=eig(A) %V为特征向量矩阵，D为特征值矩阵

%零极点对消分析

[v,d]=eig(A); %V为特征向量矩阵，D为特征值矩阵

[z,p,kl]=ss2zp(A,B,C,D);

sys2=zpk(z,p,kl)

det\_E=factor(det(E));%det求行列式 factor因式分解

E\_adj=adjoint(E);

E\_zata=E\_adj\*B;

Xa=E\_inv\*B;

Ya=C\*E\_inv

%状态方程中未出现零极点对消现象

Ga=C\*Xa

%输出方程中未出现零极点对消现象

%非齐次解析解

phit\_seta\_2=vpa(ilaplace(inv(E)\*B\*(1/s)));%

phit\_final\_t=simplify(vpa(real((phit\*[x1; x2; x3; x4]+phit\_seta\_2))))

% phit\_1=subs(phit,t,t-tao);%将变量t替换为t-tao

% phit\_sata=int(phit\_1\*B,tao,0,t);%带符号类型t——10

% phit\_final\_t=simplify(vpa(real((phit\*[x1; x2; x3; x4]+phit\_sata))));

% phit\_final\_num=subs(phit\_final\_t,t,300);%将变量t替换为10

%受控项

% phit\_1=subs(phit,t,t-tao);%将变量t替换为t-tao

% phit\_sata=int(phit\_1\*B,tao,0,t);%带符号类型t——10

% phit\_final\_t=simplify(vpa(real((phit\_sata))));

phit\_seta\_2=vpa(ilaplace(inv(E)\*B\*(1/s)));%

phit\_final\_t=simplify(vpa(real((phit\*[x1; x2; x3; x4]+phit\_seta\_2))));

t=ti:0.01:tfl;

for i=1:4

phit\_plot\_1=str2func(['@(t)', vectorize(phit\_final\_t(i))]);% 变为函数句柄

plot(t,phit\_plot\_1(t))

hold on

end

legend('y','yd','ydd','yddd')

hold off

%齐次解析解绘图

%内部稳定

t=ti:0.01:tfl;

for i=1:4

phit\_plot=str2func(['@(t)', vectorize(phit\_real\_plot(i))]);% 变为函数句柄

plot(t,phit\_plot(t))

hold on

end

legend('y','yd','ydd','yddd')

hold off

%非齐次解析解绘图

%输入输出稳定

t=ti:0.01:tfl;

for i=1:4

phit\_plot\_1=str2func(['@(t)', vectorize(phit\_final\_t(i))]);% 变为函数句柄

plot(t,phit\_plot\_1(t))

hold on

end

legend('y','yd','ydd','yddd')

hold off

%齐次数值解

[time\_lin,sol\_lin]=ode45(@lin\_pend\_dot,[ti tfl],[x1 x2 x3 x4]);

plot(time\_lin,sol\_lin);

legend('y','yd','ydd','yddd');

function xdot = lin\_pend\_dot(t,x)

global k Jl Jm Bl Bm

xdot =[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm]\*x; % very important

end

%非齐次数值解

[time\_lin,sol\_lin]=ode45(@lin\_pend\_dot1,[ti tfl],[x1 x2 x3 x4]);

plot(time\_lin,sol\_lin);

legend('y','yd','ydd','yddd');

function xdot1 = lin\_pend\_dot1(t,x)

global k Jl Jm Bl Bm

xdot1 =[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm]\*x+[0;0;0;1/Jm]\*stepfun(t,0); % very important

end

%带参数判别

%能控性

Qc=[B A\*B (A^2)\*B (A^3)\*B];

Qc\_det=det(Qc) %rank(Qc)

%系统能控

Qg=[C;C\*A;C\*A^2;C\*A^3];

Qg\_det=det(Qg)%rank(Qg)

%系统能观

%无参数判别

clear all

syms k Jl Jm Bl Bm

A=[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm];

B=[0;0;0;1/Jm];

C=[1 0 0 0];

Qc=[B A\*B (A^2)\*B (A^3)\*B];

Qc\_det=det(Qc) %rank(Qc)

Qg=[C;C\*A;C\*A^2;C\*A^3];

Qg\_det=det(Qg)%rank(Qg)

%能控条件 K^2/((Jl^2)(Jm^4))不等于0

%能观条件 K^2/((Jl^2）不等于0

clear all

syms k Jl Jm Bl Bm

A=[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm];

B=[0;0;0;1/Jm];

C=[1 0 0 0];

%不带参数的能控标准型

Qc=[B A\*B (A^2)\*B (A^3)\*B];

p1=[0 0 0 1]\*(inv(Qc));

P=[p1;p1\*A;p1\*(A^2);p1\*(A^3)];

A\_ba\_a=P\*A\*inv(P)

B\_ba\_a=P\*B

%不带参数的能观标准型

Qg=[C;C\*A;C\*(A^2);C\*(A^3)];

T1=inv(Qg)\*[0;0;0;1];

T=[T1 A\*T1 (A^2)\*T1 (A^3)\*T1];

A\_ba\_b=inv(T)\*A\*T

B\_ba\_b=inv(T)\*B

C\_ba\_b=C\*T

%带参数的能控标准型

Qc=[B A\*B (A^2)\*B (A^3)\*B];

p1=[0 0 0 1]\*(inv(Qc));

P=[p1;p1\*A;p1\*(A^2);p1\*(A^3)];

A\_ba\_a=P\*A\*inv(P)

B\_ba\_a=P\*B

%带参数的能观标准型

Qg=[C;C\*A;C\*(A^2);C\*(A^3)];

T1=inv(Qg)\*[0;0;0;1];

T=[T1 A\*T1 (A^2)\*T1 (A^3)\*T1];

A\_ba\_b=inv(T)\*A\*T

B\_ba\_b=inv(T)\*B

C\_ba\_b=C\*T

clear all

syms s

k = 6; Jl= 0.5; Jm= 0.05; Bl= 0.01; Bm= 0.01;

a=2;

Kp=0.0018;

% Kp=0.4;%不稳定

Kd=Kp/a;

pl=Jl\*s^2+Bl\*s+k;

pm=Jm\*s^2+Bm\*s+k;

PD\_simulink=Kp+Kd\*s;

pl\_simulink=k/pl;

pm\_simulink=1/pm;

pms=Jm\*s^2+Bm\*s+k;

pls=Jl\*s^2+Bl\*s+k;

gp1=1/pms;

gp2=k/pls;

gp3=gp1\*gp2/(1-k\*gp1\*gp2);

%计算系统传函--未增加PD校正环节，未加反馈

tf\_zata\_l\_un=collect(simplify(gp3));

[I,D]=numden(sym(tf\_zata\_l\_un));

I=eval(I); %分子

num\_un=I;

D=eval(D); %分母

den\_un=sym2poly(D);

sys\_un=tf(num\_un,den\_un);

rlocus(sys\_un);

% %计算系统传函--增加PD校正环节，未加反馈

% tf\_zata\_l\_pd=simplify(PD\_simulink\*(pl\_simulink\*pm\_simulink/(1-pl\_simulink\*pm\_simulink\*k)));

% [I,D]=numden(sym(tf\_zata\_l\_pd));

% I=eval(I); %分子

% num\_pd=sym2poly(I);

% D=eval(D); %分母

% den\_pd=sym2poly(D);

% sys\_pd=tf(num\_pd,den\_pd)

% rlocus(sys\_pd)

%电机转角zata\_m

%计算系统传函--增加PD校正环节,增加反馈

a=2;

Kp1=0.0052; %调节时间较长

Kp=20;%性能好的极点

Kd=Kp/a;

Kd1=Kp1/a;

pl=Jl\*s^2+Bl\*s+k;

pm=Jm\*s^2+Bm\*s+k;

G0H=(Kd\*(s+a)\*pl)/(pl\*pm-k^2);%开环传函

PD\_simulink=Kp+Kd\*s;

PD\_simulink1=Kp1+Kd1\*s;

pl\_simulink=k/pl;

pm\_simulink=1/pm;

R\_11=(PD\_simulink1\*pm\_simulink);

R\_21=R\_11/(1+R\_11);

R\_31=R\_21\*pl\_simulink;

R\_41=k\*(1/PD\_simulink1);

R\_51=simplify(R\_31/(1-R\_31\*R\_41));

R\_1=(PD\_simulink\*pm\_simulink);

R\_2=R\_1/(1+R\_1);

R\_3=R\_2\*pl\_simulink;

R\_4=k\*(1/PD\_simulink);

R\_5=simplify(R\_3/(1-R\_3\*R\_4));

tf\_zata\_m=simplify(G0H);

[I,D]=numden(sym(tf\_zata\_m));

I=eval(I); %分子

num=sym2poly(I);

D=eval(D); %分母

den=sym2poly(D);

sys\_zata\_m=tf(num,den);

rlocus(sys\_zata\_m)

tf\_zata\_m1=simplify(R\_51);

[I1,D1]=numden(sym(tf\_zata\_m1));

I1=eval(I1); %分子

num1=sym2poly(I1);

D1=eval(D1); %分母

den1=sym2poly(D1);

sys\_zata\_m1=tf(num1,den1);

step(sys\_zata\_m1)

tf\_zata\_m2=simplify(R\_5);

[I2,D2]=numden(sym(tf\_zata\_m2));

I2=eval(I2); %分子

num2=sym2poly(I2);

D2=eval(D2); %分母

den2=sym2poly(D2);

sys\_zata\_m2=tf(num2,den2);

step(sys\_zata\_m2)

%负载转角zata\_l

%计算系统传函--增加PD校正环节,增加反馈

R\_1=(pm\_simulink\*pl\_simulink);

R\_2=R\_1/(1-k\*R\_1);

% R\_3=R\_2\*(0.0018+0.089\*s+7.23\*10^(-6)/s);

R\_3=R\_2\*PD\_simulink;

R\_4=R\_3/(1+R\_3);

g0h=(Kd\*(s+a)\*k)/(pl\*pm-k^2);%开环传函

pd=0.0018+0.089\*s+7.23\*10^(-6)/s;

gspd2=pd\*gp3/(1+gp3\*pd/gp2);

tf\_zata\_l=simplify(g0h);

[I,D]=numden(sym(tf\_zata\_l));

I=eval(I); %分子

num=sym2poly(I);

D=eval(D); %分母

den=sym2poly(D);

sys\_zata\_l=tf(num,den);

rlocus(sys\_zata\_l)

Kp=0.0018;

Kd=Kp/a;

PD\_simulink=Kp+Kd\*s;

tf\_zata\_l1=simplify(R\_4);

[I2,D2]=numden(sym(tf\_zata\_l1));

I2=eval(I2); %分子

num2=sym2poly(I2);

D2=eval(D2); %分母

den2=sym2poly(D2);

sys\_zata\_l1=tf(num2,den2);

step(sys\_zata\_l1)

%选择一组不稳定极点

Kp=0.4;

Kd=Kp/a;

PD\_simulink=Kp+Kd\*s;

R\_1=(pm\_simulink\*pl\_simulink);

R\_2=R\_1/(1-k\*R\_1);

% R\_3=R\_2\*(0.0018+0.089\*s+7.23\*10^(-6)/s);

R\_3=R\_2\*PD\_simulink;

R\_4=R\_3/(1+R\_3);

tf\_zata\_l2=simplify(R\_4);

[I3,D3]=numden(sym(tf\_zata\_l2));

I3=eval(I3); %分子

num3=sym2poly(I3);

D3=eval(D3); %分母

den3=sym2poly(D3);

sys\_zata\_l2=tf(num3,den3);

step(sys\_zata\_l2)

%PID整定

PD\_simulink=0.0018+0.089\*s+7.23\*10^(-6)/s;

R\_1=(pm\_simulink\*pl\_simulink);

R\_2=R\_1/(1-k\*R\_1);

% R\_3=R\_2\*(0.0018+0.089\*s+7.23\*10^(-6)/s);

R\_3=R\_2\*PD\_simulink;

R\_4=R\_3/(1+R\_3);

tf\_zata\_l2=simplify(R\_4);

[I3,D3]=numden(sym(tf\_zata\_l2));

I3=eval(I3); %分子

num3=sym2poly(I3);

D3=eval(D3); %分母

den3=sym2poly(D3);

sys\_zata\_l2=tf(num3,den3);

step(sys\_zata\_l2)

%李雅普诺夫稳定性判定

% AX + XA' = -C

% 这是函数的内部定义式，恰好与理论定义的转置是反着的

P = lyap(A', I) % 一般令Q=I（I指单位阵）

all(eig(P)>0&imag(eig(P))==0)

%取Q=I时，此时P为正定矩阵，系统稳定

[V,D]=eig(A) % D的对角线上即为特征值

%此时A的特征值全部在左半平面，系统稳定

%全状态反馈

Qc=ctrb(A,B); % 求取系统的能控矩阵

rank(Qc);

%能控标准型

p1=[0 0 0 1]\*(inv(Qc));

P=[p1;p1\*A;p1\*(A^2);p1\*(A^3)];

A\_ba\_a=P\*A\*inv(P);

B\_ba\_a=P\*B;

C\_ba\_c=C\*inv(P);

%状态反馈矩阵

% p1=-0.5;p2=-0.5;p3=-0.5;p4=-0.5;

p1=-100;p2=-100;p3=-100;p4=-100;

% K=place(A,B,[p1,p2,p3,p4])

M=[p1;p2;p3;p4]; %新的极点

K=acker(A,B,M); % Ackermann公式,求解状态反馈阵K,其中，A、B为系统的状态空间模型矩阵，向量P中是期望的闭环极点位置，返回值是增益向量。

k1=K(1);

k2=K(2);

k3=K(3);

k4=K(4);

%利用ode45的数值解判断非零初值能否回到稳定状态

[time\_lin,sol\_lin]=ode45(@state\_feedback\_fun,[0 1],[x1 x2 x3 x4]);%

plot(time\_lin,sol\_lin);

function xdot = state\_feedback\_fun(t,x)

global k Jl Jm Bl Bm K K1

xdot =[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm]\*x-[0;0;0;1/Jm]\*K\*x;

end

%阶跃输入响应

sys=ss(A-B\*K,B,C,D);

[z,p,k]=ss2zp(A-B\*K,B,C,D);

sys2=zpk(z,p,k);

t=0:0.01:1;

[num,den]=ss2tf(A-B\*K,B,C,D);

sys\_use=tf(num,den);

step(sys\_use,t)

% % [y,t,x]=lsim(sys,u,t,x0);%lsim任意输入u的响应，x0状态初始值，y输出

[y,t,x]=lsim(sys,stepfun(t,0),t,[x1 x2 x3 x4]);%lsim任意输入u的响应，x0状态初始值，y输出

for i=1:4

plot(t,x(:,i))

hold on

end

% legend('y','yd','ydd','yddd')

hold off

%参考输入

t=0:1:30;

[y,t,x]=lsim(sys,170820\*ones(1,31),t);

plot(t, y)

%状态观测器

P1=[-120; -130; - 140; -150];

P2=[-300; -310; -320 ; -330];

% 判断系统是否完全能观

E\_obs = obsv(A,C); % 求解能观性矩阵

E\_val = rank(E\_obs); % 根据能控性矩阵是否满秩判断能观性

L1 = (acker(A',C',P1))' % 利用acker求状态增益矩阵

L2= (acker(A',C',P2))'

%第一组极点

[time\_lin,sol\_lin]=ode45(@observer,[0 10],[x1 x2 x3 x4]);%

plot(time\_lin,sol\_lin);

function xdot = observer(t,x)

global k Jl Jm Bl Bm L1

xdot =([0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm]-L1\*[1 0 0 0])\*x+[0;0;0;1/Jm]\*stepfun(t,0)+(L1\*[1 0 0 0])\*x; %very important

end

%第二组极点

[time\_lin1,sol\_lin1]=ode45(@observer1,[0 10],[x1 x2 x3 x4]);

plot(time\_lin1,sol\_lin1);

function xdot = observer1(t,x)

global k Jl Jm Bl Bm L2

xdot =([0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm]-L2\*[1 0 0 0])\*x+[0;0;0;1/Jm]\*stepfun(t,0)+(L2\*[1 0 0 0])\*x; %very important

end

%LPR控制器

Q=[1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1]; %Q为给定的半正定实对称常数矩阵,Q越大收敛速度越快

R=1;%R为给定的正定实对称常数矩阵，R越大收敛效果越好

% K1为最优反馈增益矩阵；

% S为对应Riccati方程的唯一正定解P（若矩阵A-BK是稳定矩阵，则总有正定解P存在）；

% E为矩阵A-BK的特征值

[K1,S,E] = lqr(A,B,Q,R);

k1\_1=K1(1);

k1\_2=K1(2);

k1\_3=K1(3);

k1\_4=K1(4);

sys=ss(A-K1\*B,B,C,D);

t=0:0.01:10;

[num,den]=ss2tf(A-B\*K1,B,C,D);

sys\_use=tf(num,den)

step(sys\_use,t)

% [y,t,x]=lsim(sys,u,t,x0);%lsim任意输入u的响应，x0状态初始值，y输出

t=0:0.01:2;

u=zeros(size(t));%判断是否会回到平衡点（0,0）stepfun(t,0)

[y,t,x]=lsim(sys,u,t,[x1 x2 x3 x4]);%lsim任意输入u的响应，x0状态初始值，y输出

plot(t,x(:,1))

legend('y')

% hold off

%利用ode45的数值解判断非零初值能否回到稳定状态

[time\_lin,sol\_lin]=ode45(@state\_feedback\_fun1,[0 10],[x1 x2 x3 x4]);%

plot(time\_lin,sol\_lin);

function xdot = state\_feedback\_fun1(t,x)

global k Jl Jm Bl Bm K K1

xdot =[0 1 0 0;-k/Jl -Bl/Jl k/Jl 0; 0 0 0 1; k/Jm 0 -k/Jm -Bm/Jm]\*x-[0;0;0;1/Jm]\*K1\*x; %最优控制 very important

end

1. [↑](#endnote-ref-1)