倒立摆系统与自动控制实验

实验指导书（硕士研究生专用）

广西大学机械工程学院 机械电子工程专业

二〇〇七年十二月

内容简介

这是一本为工科高年级学生和研究生编写的实验与实践教科书，可以作为控制系统领域各门控制课程的配套实验教材。本书是基于固高摆系统完成的。

在控制实践中，首先是关于控制对象的知识获取与表达，也就是控制对象模型结构的选取与建模。因此本书第一部分安排了直线一级摆系统（第一章）、环形一级摆系统（第二章）和直线一级顺摆（第三章）的动力学建模与实验，同时分别采用了牛顿－欧拉方法和拉各朗日方法等两种方法。关于控制器的知识获取与表达，也就是控制器的结构与参数设计，在经典控制论实验中将直线一级倒立摆当作简单的单输入单输出系统（忽略了小车位移的控制）采用了 PID，根轨迹，频域响应三种控制器设计方法进行了控制器结构设计和参数设计；在现代控制论实验和最优控制实验中，考虑了小车位移的控制，将直线一级倒立摆当作单输入多输出系统，分别采用了极点配置法和线性二次型最优控制策略，进行控制器结构和参数设计。这些实验内容都按照上面的顺序编排，是本书的第二部分，也是本书的主体。第二部分的实验内容完全与《现代控制工程》的教学内容配套，所使用的实验软件平台也是 MATLAB，适用了基础实验课程的需要。本书的第三部分为智能控制实验和复杂系统控制器设计和调整，主要进行神经网络 PID 控制实验，能量控制策略起摆实验，环形串联两级倒立摆控制实验。而直线三级倒立摆和环形并联两级倒立摆的控制器结构设计和参数调整非常复杂和困难，在第三部分中只是给出了系统的数学模型，以供控制理论的研究生和教师进行研究时参考。第三部分所使用的软件平台为 Borland C+3.1，主要目的是为研究者提供一个开放式的研究开发平台，方便研究者采用 C 语言来实现比较复杂的控制算法。对于那些对实时控制编程感兴趣的学生和老师来说，第三部分也是值得仔细阅读的一章。

本书适合高年级本科生、研究生、工程技术人员及计算机控制系统开发人员使用。

序言

摆是进行控制理论研究的典型实验平台，可以分为倒立摆和顺摆。由于倒立摆系统的控制策略和杂技运动员顶杆平衡表演的技巧有异曲同工之处，极富趣味性，而且许多抽象的控制理论概念如系统稳定性、可控性和系统抗干扰能力等等，都可以通过倒立摆系统实验直观的表现出来，因此在欧美发达国家的高等院校，它已成为必备的控制理论教学实验设备。学习自动控制理论的学生通过倒立摆系统实验来验证所学的控制理论和算法，非常的直观、简便，在轻松的实验中对所学课程加深了理解。

倒立摆不仅仅是一种优秀的教学实验仪器，同时也是进行控制理论研究的理想实验平台。由于倒立摆系统本身所具有的高阶次、不稳定、多变量、非线性和强耦合特性，许多现代控制理论的研究人员一直将它视为典型的研究对象，不断从中发掘出新的控制策略和控制方法，相关的科研成果在航天科技和机器人学方面获得了广阔的应用。二十世纪九十年代以来，更加复杂多种形式的倒立摆系统成为控制理论研究领域的热点，每年在专业杂志上都会有大量的优秀论文出现。

固高科技有限公司（以下简称固高科技）为高等院校的自动控制教学提供了整套基于摆系统的实验解决方案。包括各种摆的开发生产、实验内容的安排和配置，以及对应的自动控制理论教学内容和相关经典教材的推荐。固高科技开发生产的倒立摆系列包括直线运动型和圆周运动型两个系列，主要特点包括：

开放性：采用四轴运动控制板卡，机械部分和系统硬件部分非常容易扩展，可以根据用户需要进行配置。系统软件接口充分开放，用户不仅可以使用配套的实验软件，而且可以根据自己的实际需要扩展软件的功能。

模块化：系统的机械部分可以选用直线或者旋转平台，根据实际需要配置成成一级、二级或者三级倒立摆。而三级摆可以方便地改装成两级摆，两级摆可以改装成一级摆。系统实验软件同样是基于模块化的思想设计，用户可以根据需要增加或者修改相应的功能模块。

简易安全：摆系统包括运动控制板卡、电控箱（旋转平台系统中和机械本体联在一起）、机械本体和微型计算机几个部分组成，安装升级方便。同时在机械、运动控制板卡和实验软件上都采取了积极措施，保证实验时人员的安全可靠和仪器安全。

方便性：倒立摆系统易于安装、升级，同时软件界面操作简单。

先进性：采用工业级四轴运动控制板卡作为核心控制系统，先进的交流伺服电机作为驱动，检测元件使用光电码盘而不使用电位计。系统设计符合当今先进的运动控制发展方向。

固高摆系统适应如下课程的实验：自动控制原理，现代控制理论，现代控制工程，最优控制，非线性系统控制，智能控制，模糊控制和神经网络控制等等。教师可以根据学生实际情况开设相应的实验课程。

注意事项

1. 本指导书中例程的系统参数只具有指导意义，请同学自己完成控制系统设计及实现。
2. 由于倒立摆系统是典型的多变量，强耦合，非线性系统，在经典控制器设计时在直线平台上可能导致小车“撞墙”，因此要注意仔细设计和实验。在线性控制器设计时，由于系统的非线性影响，控制器仿真成功不一定就能实际控制倒立摆系统，因此控制器参数也要仔细调整。

目 录

第一章　直线一级倒立摆的牛顿－欧拉方法建模

系统建模可以分为两种：机理建模和实验建模。实验建模就是通过在研究对象上加上一系列的研究者事先确定的输入信号，激励研究对象并通过传感器检测其可观测的输出，应用数学手段建立起系统的输入－输出关系。这里面包括输入信号的设计选取，输出信号的精确检测，数学算法的研究等等内容。机理建模就是在了解研究对象的运动规律基础上，通过物理、化学的知识和数学手段建立起系统内部的输入－状态关系。

对于倒立摆系统，由于其本身是自不稳定的系统，实验建模存在一定的困难。但是经过小心的假设忽略掉一些次要的因素后，倒立摆系统就是一个典型的运动的刚体系统，可以在惯性坐标系内应用经典力学理论建立系统的动力学方程。下面我们采用其中的牛顿－欧拉方法建立直线型一级倒立摆系统的数学模型。

微分方程的推导

在忽略了空气阻力，各种摩擦之后，可将直线一级倒立摆系统抽象成小车和匀质杆组成的系统，如下图 1-1 所示。

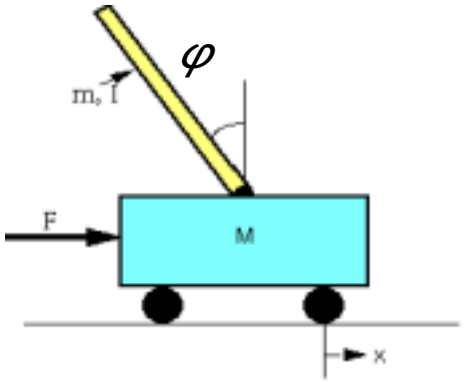


图 1-1　直线一级倒立摆系统

我们不妨做以下假设：

M　　小车质量

m　　摆杆质量

b　　小车摩擦系数

l　　摆杆转动轴心到杆质心的长度

I　　摆杆惯量

F　　加在小车上的力

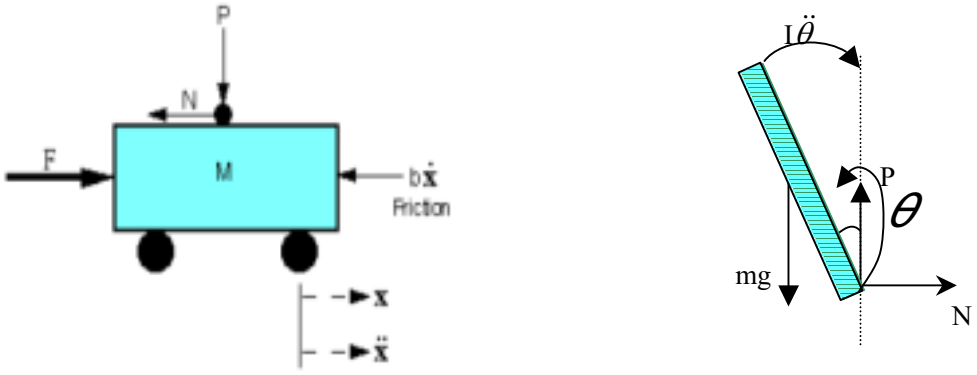
x　　小车位置

φ　　摆杆与垂直向上方向的夹角

θ　　摆杆与垂直向下方向的夹角（考虑到摆杆初始位置为竖直向下）

下图是系统中小车和摆杆的受力分析图。其中，N 和 P 为小车与摆杆相互作用力的水平和垂直方向的分量。

注意：在实际倒立摆系统中检测和执行装置的正负方向已经完全确定，因而矢量方



(a) (b)

图 1-2 （a）小车隔离受力图 （b） 摆杆隔离受力图

向定义如图所示，图示方向为矢量正方向。

分析小车水平方向所受的合力，可以得到以下方程：

$$M\ddot{x} = F - b\dot{x} - N$$

由摆杆水平方向的受力进行分析可以得到下面等式：

$$N = m\frac{d^2}{dt^2}(x + l\sin\theta)$$

即: $N = m\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta$

把这个等式代入上式中，就得到系统的第一个运动方程：

$$(M+m)\ddot{x} + b\dot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F \qquad (1.1)$$

为了推出系统的第二个运动方程，我们对摆杆垂直方向上的合力进行分析，可以得到下面方程：

$$P - mg = m\frac{d^2}{dt^2}(l\cos\theta)$$

即：$P - mg = -ml\ddot{\theta}\sin\theta - ml\dot{\theta}^2\cos\theta$

力矩平衡方程如下：

$$-Pl\sin\theta - Nl\cos\theta = I\ddot{\theta}$$

注意：此方程中力矩的方向，由于 $\theta = \pi + \phi, \cos\phi = -\cos\theta, \sin\phi = -\sin\theta$，故等式前面有负号。

合并这两个方程，约去 P 和 N，得到第二个运动方程：

$$(I+ml^2)\ddot{\theta}+mgl\sin\theta=-ml\ddot{x}\cos\theta \tag{1.2}$$

设$\theta=\pi+\phi$（ϕ是摆杆与垂直向上方向之间的夹角），假设ϕ与1（单位是弧度）相比很小，即ϕ《1，则可以进行近似处理：$\cos\theta=-1$，$\sin\theta=-\phi$，$(\frac{d\theta}{dt})^2=0$。用u来代表被控对象的输入力F，线性化后两个运动方程如下：

$$\begin{cases}\left(I+ml^2\right)\ddot{\phi}-mgl\phi=ml\ddot{x}\\(M+m)\ddot{x}+b\dot{x}-ml\ddot{\phi}=u\end{cases} \tag{1.3}$$

传递函数

对方程组(1.3)进行拉普拉斯变换，得到

$$\begin{cases}\left(I+ml^2\right)\Phi(s)s^2-mgl\Phi(s)=mlX(s)s^2\\(M+m)X(s)s^2+bX(s)s-ml\Phi(s)s^2=U(s)\end{cases} \tag{1.4}$$

注意：推导传递函数时假设初始条件为0。

由于输出为角度ϕ，求解方程组（1.4）的第一个方程，可以得到

$$X(s)=[\frac{(I+ml^2)}{ml}-\frac{g}{s^2}]\Phi(s)$$

把上式代入方程组（1.4）的第二个方程，得到

$$(M+m)\left[\frac{(I+ml^2)}{ml}-\frac{g}{s}\right]\Phi(s)s^2+b\left[\frac{(I+ml^2)}{ml}+\frac{g}{s^2}\right]\Phi(s)s-ml\Phi(s)s^2=U(s)$$

整理后得到传递函数：

$$\frac{\Phi(s)}{U(s)}=\frac{\frac{ml}{q}s^2}{s^4+\frac{b(I+ml^2)}{q}s^3-\frac{(M+m)mgl}{q}s^2-\frac{bmgl}{q}s}$$

其中

$$q=[(M+m)(I+ml^2)-(ml)^2]$$

状态空间方程

系统状态空间方程为

$$\dot{X} = AX + Bu$$
$$y = CX + Du$$

方程组（3）对$\ddot{x},\ddot{\phi}$解代数方程，得到解如下：

$$\begin{cases} \dot{x} = \dot{x} \\ \ddot{x} = \dfrac{-(I+ml^2)b}{I(M+m)+Ml^2}\dot{x} + \dfrac{m^2gl^2}{I(M+m)+Ml^2}\phi + \dfrac{(I+ml^2)}{I(M+m)+Ml^2}u \\ \dot{\phi} = \dot{\phi} \\ \ddot{\phi} = \dfrac{-mlb}{I(M+m)+Ml^2}\dot{x} + \dfrac{mgl(M+m)}{I(M+m)+Ml^2}\phi + \dfrac{ml}{I(M+m)+Ml^2}u \end{cases}$$

整理后得到系统状态空间方程：

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \dfrac{-(I+ml^2)b}{I(M+m)+Ml^2} & \dfrac{m^2gl^2}{I(M+m)+Ml^2} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \dfrac{-mlb}{I(M+m)+Ml^2} & \dfrac{mgl(M+m)}{I(M+m)+Ml^2} & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{I+ml^2}{I(M+m)+Ml^2} \\ 0 \\ \dfrac{ml}{I(M+m)+Ml^2} \end{bmatrix} u$$

$$y = \begin{bmatrix} x \\ \phi \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u$$

开环系统仿真

实际系统的模型参数如下:

M	小车质量	1.096 Kg
m	摆杆质量	0.109 Kg
b	小车摩擦系数	0 .1N/m/sec
l	摆杆转动轴心到杆质心的长度	0.2 5m
I	摆杆惯量	0.0034 kg*m*m
T	采样频率	0.005 秒

注意：在进行实际系统的 MATLAB 仿真时，请将采样频率改为实际系统的采样频率。请用户自行检查系统参数是否与实际系统相符，否则请改用实际参数进行实验。

实验步骤

1. 在 MATLAB Command 窗口中，首先进入到倒立摆系统 MATLAB 仿真文件的路径。
2. 输入 pll_open_t.m 可以看到系统传递函数模型在冲击输入下的开环响应。

3. 输入 pll_open_s.m 可以看到系统状态空间模型在阶跃输入下的开环响应。

4. 如果要修改系统参数或者输入信号，请打开相应文件进行编辑，然后在进行相应的实验。

5. 完成试验报告，分析系统的开环响应特性

第二章 直线一级顺摆的建模

微分方程的推导

当直线一级倒立摆摆杆自由下垂时，如何控制小车在按给定的运动规律运动的同时保持摆杆的自由下垂状态，是起重运输过程中必须解决的问题。这里，暂且将此类问题定义为**顺摆**问题。与直线一级倒立摆类似，可将直线一级顺摆系统抽象成小车、匀质摆杆组成的刚体系统（参考图 1-1 所示）。假设

M ：小车的质量

m_1：摆杆的质量

l_1：摆杆转动中心到杆质心的距离

x：小车的位置（水平向右为正）

θ_1：摆杆与垂直向下方向的夹角（顺时针为正）

系统的拉格朗日方程为：

$$L(q,\dot{q}) = T(q,\dot{q}) - V(q,\dot{q})$$

其中，L 为拉格朗日算子，q 为系统的广义坐标，T 为系统的动能，V 为系统的势能。拉格朗日方程由广义坐标 q_i 和 L 表示为：

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = f_i$$

其中，$i = 1,2,3\cdots n$，f_i 为系统沿该广义坐标方向上的外力，在本系统中，设系统的两个广义坐标分别是 x,θ_1。

首先计算系统的动能：

$$T = T_{m1} + T_{m2}$$

T_M 为小车动能，T_{m1} 为摆杆动能。

$$T_M = \frac{1}{2}M\dot{x}^2$$

$$T_{m1}^{'}=\frac{1}{2}m_1\left(\left(\frac{d(x-l_1\sin\theta_1)}{dt}\right)^2+\left(\frac{d(l_1\cos\theta_1)}{dt}\right)^2\right)=\frac{1}{2}m_1\dot{x}^2-m_1l_1\dot{x}\dot{\theta}_1\cos\theta_1+\frac{1}{2}m_1l_1^2\dot{\theta}_1^2$$

$$T_{m1}^{''}=\frac{1}{2}J_p\omega_1^2=\frac{1}{2}\left(\frac{1}{3}m_1l_1^2\right)\dot{\theta}_1^2=\frac{1}{6}m_1l_1^2\dot{\theta}_1^2$$

则

$$T_{m1}=T_{m1}^{'}+T_{m1}^{''}=\frac{1}{2}m_1\dot{x}^2-m_1l_1\dot{x}\dot{\theta}_1\cos\theta_1+\frac{2}{3}m_1l_1^2\dot{\theta}_1^2$$

可以得到系统动能：

$$T=\frac{1}{2}M\dot{x}^2+\frac{1}{2}m_1\dot{x}^2-m_1l_1\dot{x}\dot{\theta}_1\cos\theta_1+\frac{2}{3}m_1l_1^2\dot{\theta}_1^2$$

系统的势能为：(以摆杆自由下垂的位置为零势能位置)

$$V=m_1gl_1(1-\cos\theta_1)$$

从而拉格朗日算子为:

$$L=\frac{1}{2}M\dot{x}^2+\frac{1}{2}m_1\dot{x}^2-m_1l_1\dot{x}\dot{\theta}_1\cos\theta_1+\frac{2}{3}m_1l_1^2\dot{\theta}_1^2+m_1gl_1(1-\cos\theta_1)$$

由于在广义坐标θ_1上无外力作用，有以下等式成立：

$$\frac{d}{dt}\left(\frac{\partial L}{\partial\dot{\theta}_1}\right)-\frac{\partial L}{\partial\theta_1}=0$$

得到下式：

$$m_1l_1\cos\theta_1\ddot{x}+\frac{4}{3}m_1l_1^2\ddot{\theta}_1+m_1gl_1\sin\theta_1=0$$

令

$$u=\ddot{x}$$

取平衡位置时各变量的初值为零,

$$X=(x,\dot{x},\theta_1,\dot{\theta}_1)=(0,0,0,0)$$

得到

$$\ddot{x}=u$$

$$\ddot{\theta}_1=-\frac{3g}{4l_1}\theta_1-\frac{3}{4l_1}u$$

系统的状态空间模型

取状态变量如下：

$$x_1 = x$$
$$x_2 = \dot{x}$$
$$x_3 = \theta_1$$
$$x_4 = \dot{\theta}_1$$

由(5)，(6)式得到状态空间方程如下：

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\frac{3g}{4l_1} & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ -\frac{3}{4l_1} \end{bmatrix} u$$

$$y = \begin{bmatrix} x \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u$$

系统开环响应仿真

实际系统的模型参数如下:

M	小车质量	1.096 Kg
m	摆杆质量	0.109 Kg
l_1	摆杆转动轴心到杆质心的长度	0.25m
I	摆杆惯量	0.0034 kg*m*m
T	采样频率	0.005 秒

注意：在进行实际系统的 MATLAB 仿真时，请将采样频率改为实际系统的采样频率。请用户自行检查系统参数是否与实际系统相符，否则请改用实际参数进行实验

实验步骤

1. 在 MATLAB Command 窗口中，首先进入到倒立摆系统 MATLAB 仿真文件的路径。

2. 输入 pc1_open_s.m 可以看到系统状态空间模型在阶跃输入下的开环响应。

3. 如果要修改系统参数或者输入信号，请打开相应文件进行编辑，然后在进行相应的实验。

4. 完成试验报告，分析系统的开环响应特性

第三章 固高倒立摆系统 MATLAB 实验软件

固高倒立摆系统实验控制软件工作在 Windows 2000 操作系统环境下，不支持 Windows98 和 95。系统运行时需要 MATLAB 6 Release 12.1 以及 SIMULINK 4.1 支持。用户使用控制软件前需自行安装上述软件。

固高倒立摆系统 MATLAB 实验软件的安装

第一步：倒立摆系统控制软件以光盘的形式分发，请将光盘插入驱动器

第二步：在光盘上找到软件安装目录，双击“setup.exe”,启动安装程序。系统出现图3-1 提示：

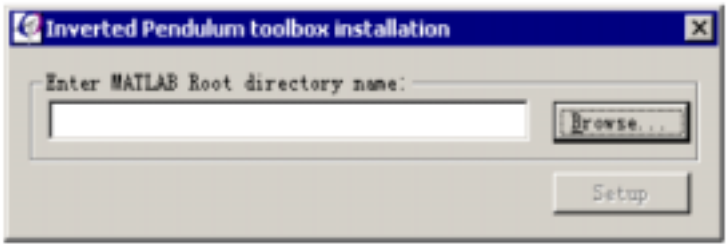


图3-1 安装提示对话框

第三步：安装程序需要知道 MATLAB 软件所在的软件根目录路径，点击“Browse …”按钮，系统会现图3-2 浏览对话框，找到 MATLAB 软件路径并选中。

图3-2 浏览对话框

第四步：点击确定，然后点击“setup”按钮，系统会出现进度框。软件自动安装于

MATLAB\Toolbox\pendPCI 目录下。如果此目录存在，安装可能失败。

第五步：安装完毕，可以启动 SIMULINK 使用本软件。

MATLAB 以及 SIMULINK 应用的基本介绍

下面以直线型一级倒立摆 LQR 算法为例介绍 MATLAB 编程：

```
M = .5;                                  //变量定义及初始化
m = 0.2;
b = 0.1;
I= 0.006;
g = 9.8;
l = 0.3;

p = (M+m)*(I+m*l^2)-(m*l)^2;      %simplifies input

A = [0          1                    0                  0;                  //矩阵定义及初始化
     0    -(I+m*l^2)*b/p    (m^2*g*l^2)/p       0;
     0          0                    0                  1;
     0     -(m*l*b)/p         m*g*l*(M+m)/p    0]
B = [         0;
       (I+m*l^2)/p;
             0;
          m*l/p]
C = [1 0 0 0;
       0 0 1 0]
D = [0;
       0]

  %close loop system response for step signal
x=20;
y=100;
Q=[x 0 0 0;
    0 0 0 0;
    0 0 y 0;
    0 0 0 0];
R = 1;
K = lqr(A,B,Q,R)                                    //lqr 算法，直接调用控制工具箱的函数

Ac = [(A-B*K)];
Bc = [B];
Cc = [C];
Dc = [D];

T=0:0.005:5;
```

```
U=0.2*ones(size(T));
Cn=[1 0 0 0];
Nbar=rscale(A,B,Cn,0,K)                    //自定义函数
Bcn=[Nbar*B];
[Y,X]=lsim(Ac,Bcn,Cc,Dc,U,T);              //仿真

plot(T,Y)                                  //图形显示
legend('Cart','Pendulum')
```

自定义函数必须存成与函数名相同的 m 文件。其中的自定义函数如图3-3 所示：

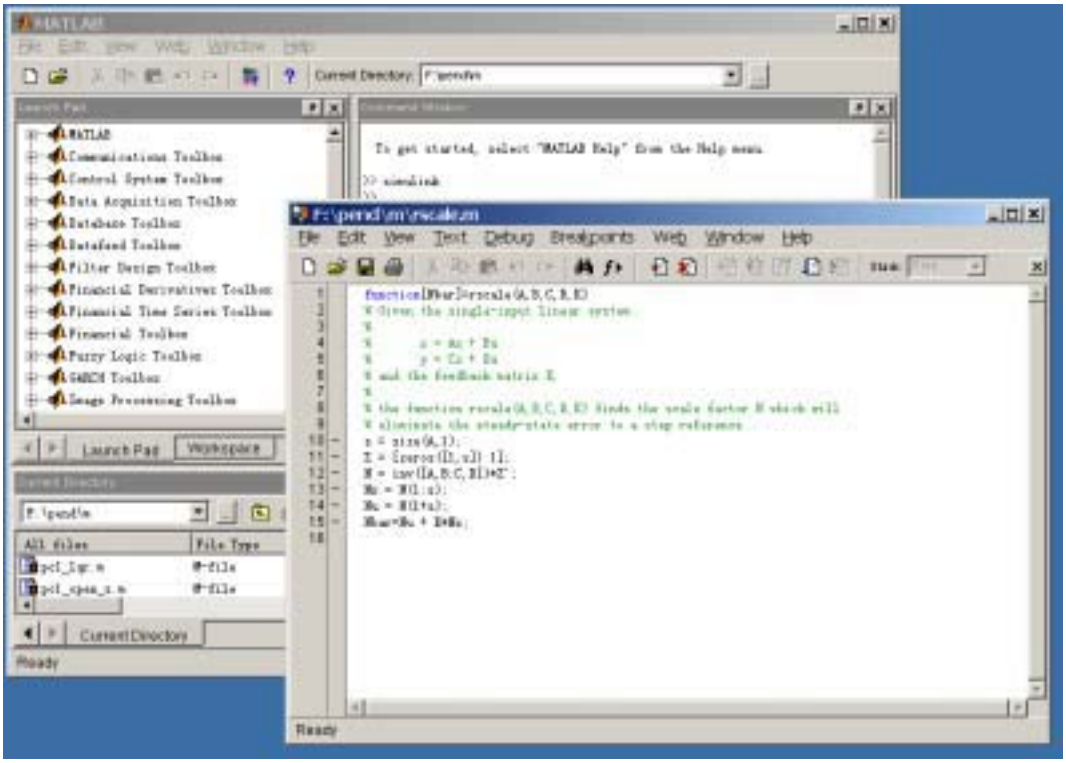


图3-3 自定义函数窗口

在 MATLAB Editor 窗口左上角显示函数存储在“F:/pend/m/rascale.m”中。

关于 MATLAB 编程更详细的信息，请参看 MATLAB 帮助及相关书籍。

SIMULINK 是基于鼠标拖动的图形化，模块化设计的系统分析工具。

点击 SIMULINK Library browser 的“File”下拉菜单，点击“File/New/Model”，或者直接按热键“Ctrl-N”，应用程序生成一个新窗口，用户可以将 SIMULINK Library 中的模块用鼠标拖动到新窗口中。

通过鼠标可以将两个模块连接起来。以下面的简单例子来说：

将模块信号发生器“SIMULINK/Sources/Signal Operator”、示波器“SIMULINK /Sinks/Scope”、封装子系统“SIMULINK/Subsystems/Subsystem”拖到新窗口中，然后按住鼠标，将系统输入输出连接起来。其中封装子系统默认含有输入“in1”、输出“out1”模块，用户可以自行增删输入输出。将线性控制系统模块“Control System Toolbox/LMI System”增加到子系统中，并且将输入输出连接起来，则系统如图3-4 所示：

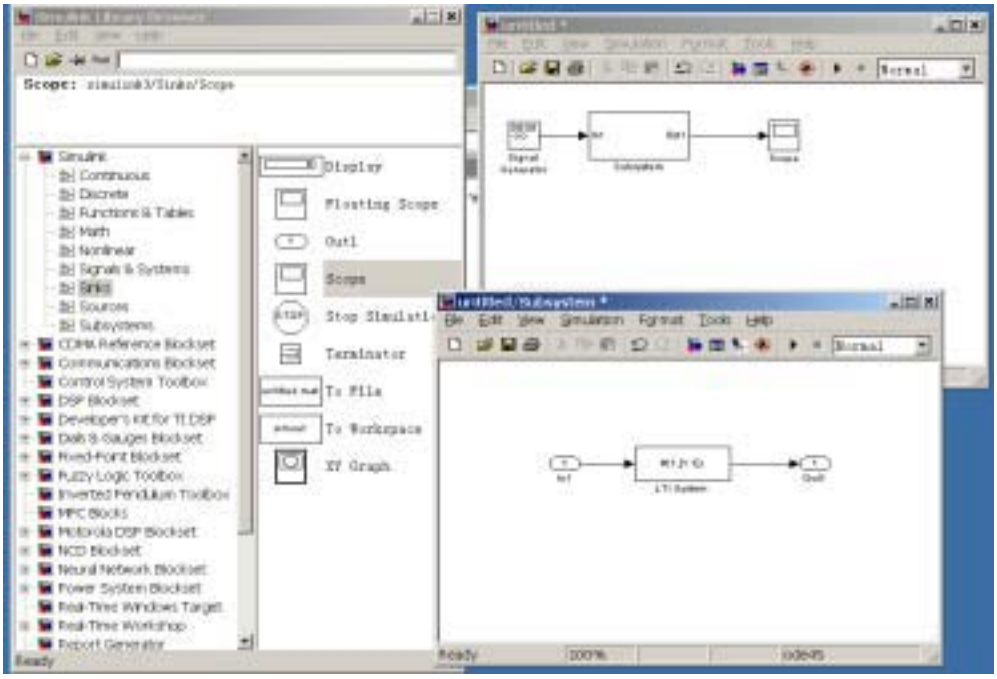


图3-4 SIMULINK 实例窗口

双击需要修改参数的模块进行参数修改，修改完毕就可以进行系统的仿真以确定系统的性能和特性了。

例如假设线性系统为：

$$G(s)=\frac{2s+1}{s^2+3s+10}$$

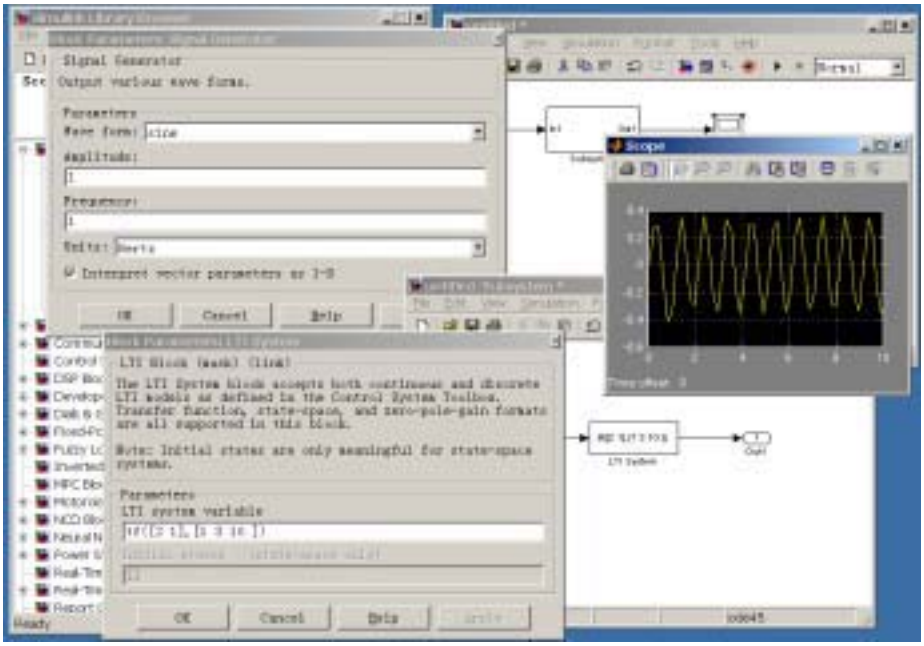


图3-5 系统示波器输出

采用幅值为 1，频率为 1 赫兹的正弦信号作为实验信号，系统的输出如示波器（图3-5）所示。

S-Function 是为方便用户扩展 SIMULINK 模块库而设计的，它有专门的格式。用户可以用 MATLAB Editor 手动编写 S－Function，也可以通过 S-Function Builder 集成环境定制。

关于 S-Function 以及封装子系统的知识请参看 MATLAB 的联机帮助以及相关书籍。

固高倒立摆系统工具箱

固高倒立摆实时控制工具箱软件包括板卡相关功能函数、简单的例子、直线型单级倒立摆以及二级倒立摆的实时控制示例程序几个部分。用户可以根据说明自己设计更新的算法，或者将之移植到环型倒立摆系统上。实时控制示例程序包括一级倒立摆 LQR 算法、极点配置算法、PID 控制、根轨迹法、频域设计法五种以及二级倒立摆 LQR 算法。

注意：因为 Windows 2000 是多任务的操作系统，为了保证倒立摆系统控制输出的实时性，请在运行实控程序前关闭其余的应用程序，并且在运行过程中禁止启动其它的应用程序。

实验步骤如下：

1. 检查电源线、数据线正确安装。关闭电控箱电源。将小车放在导轨中间。
2. 保证倒立摆杆垂直向下稳定。
3. 打开电控箱开关，接通电源。
4. 打开示例程序，设置正确地控制参数，开始实控。
5. 用手将倒立摆杆柔和地扶起，当电机启动后，手轻轻放开。
6. 对于经典控制理论设计的控制算法，只能保证倒立摆杆平衡，不能控制小车的位置。实验时需手动保证小车不要“撞墙”。
7. 实验结束，关闭程序，关闭电控箱。

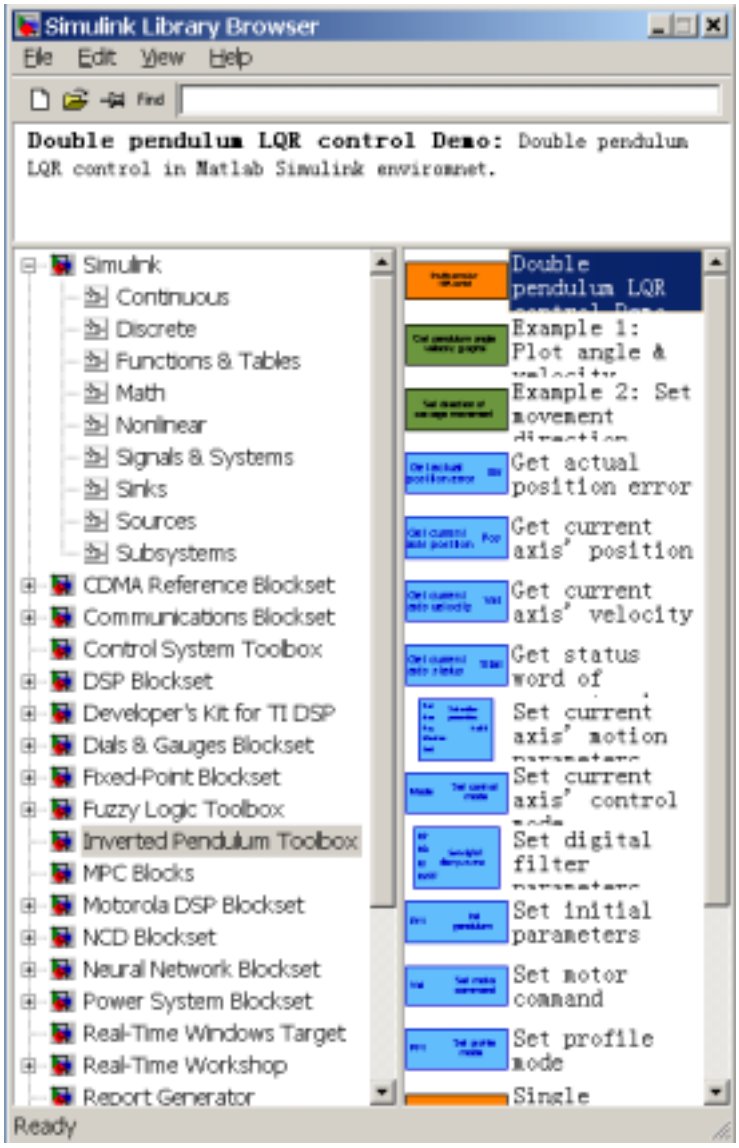


图3-6 固高摆系统的 SIMULINK 模块库

下面以直线型一级倒立摆 LQR 算法实时控制示例程序为例子，介绍倒立摆系统实时控制工具箱的应用及使用方法。

首先在 Windows 2000 操作系统环境下启动 MATLAB 应用程序，在 Command Windows 窗口中键入 SIMULINK 命令，启动 SIMULINK 应用程序。

成功安装倒立摆系统控制软件后，在 SIMULINK 的模块库中，添加了“Inverted Pendulum Toolbox”模块库，单击模块库，在右边窗口中模块库展开包括示例程序五个（红色），简单

的例子两个（绿色），封装的 S－Function 模块十个（蓝色）（图3-6 所示）。封装的 S－Function 主要是运动控制板卡的功能函数的封装，如果不是对于 C 语言和 MATLAB 的 CMEX 功能不是很熟悉，建议用户不要修改，并且不要修改相应的头文件和 c 语言源代码。

双击“Single Pendulum LQR Control Demos”模块，仿真程序如下：

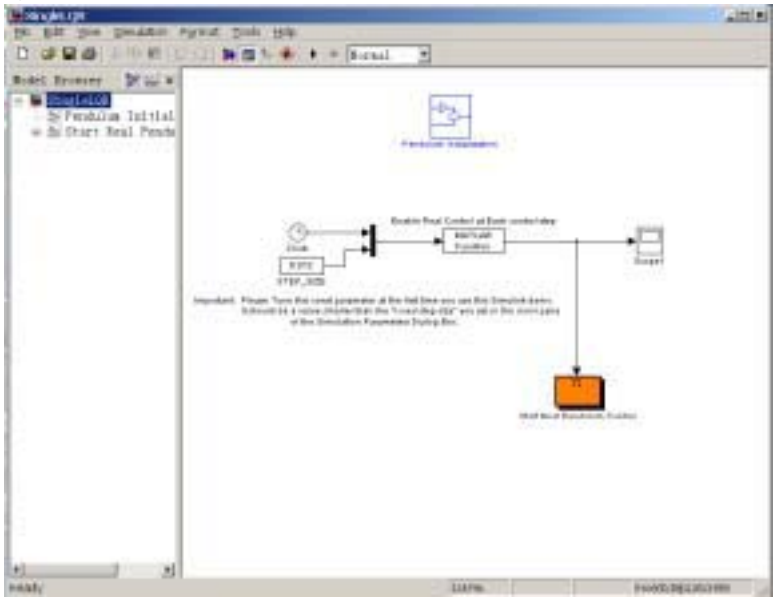


图3-7 直线一级摆的 LQR 仿真程序

仿真模块包括“Pendulum Initializaton”，“Enbale Real Control at Each Step”以及“Start Real Pendulum Control”三个主要的部分。其中“Enbale Real Control at Each Step”主要是设定控制时钟，使板卡控制输出使能。“Pendulum Initializaton”部分是运动控制板卡的初始化，其模块结构如下：

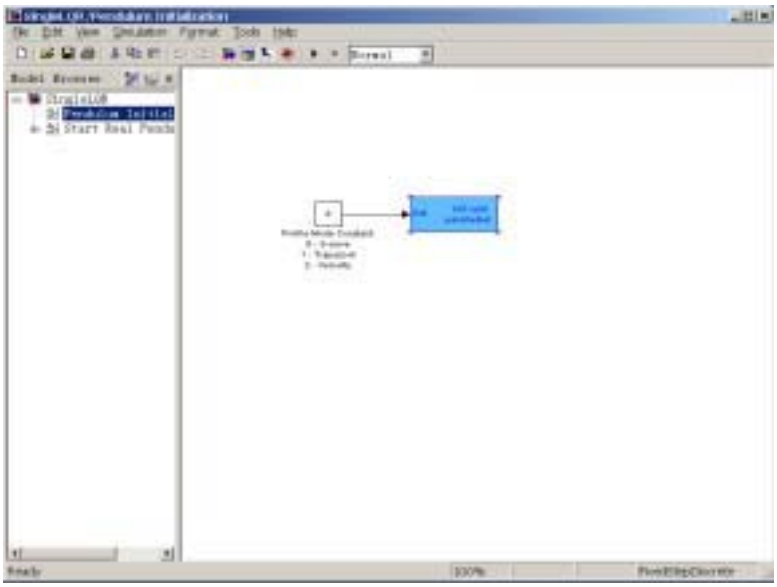


图3-8 Pendulum Initialization 模块结构图

双击蓝色 S—Function 模块，可以获得该模块的功能说明。

通过“Profile Mode Constant”模块可选择板卡的控制模式，默认为速度控制模式。关于板卡的工作模式的意义及初始化设定过程，请参见相关板卡的说明书。

实时控制示例程序最主要的部分是”Start Real Pendulum Control“，其结构如图3-9 所示。

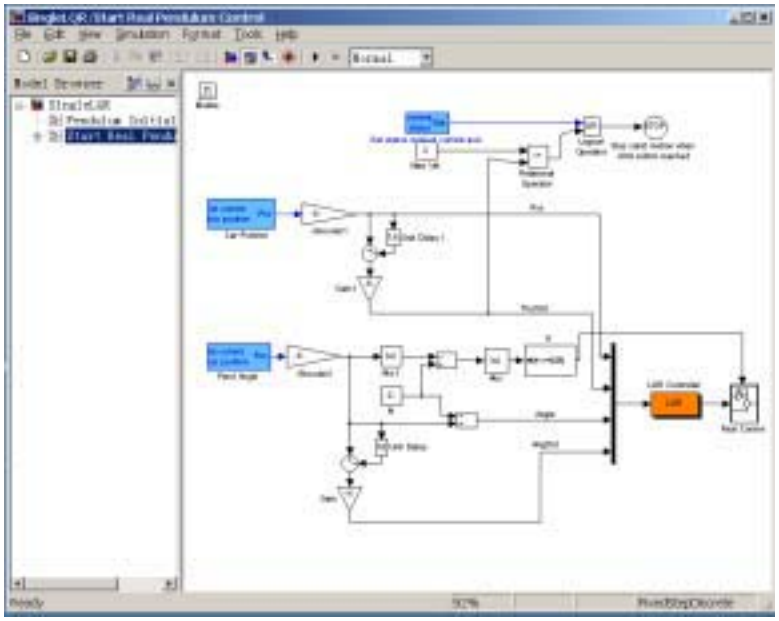


图3-9 LQR 控制模块主体结构

在数据采集和电机控制输出中使用了通过 S-Function 封装好的板卡相关功能函数。算法控制部分包括控制逻辑判断，控制算法等。其中的速度量通过延时环节差分得到：

$$PosDot(k) = \frac{[Pos(k-1) - Pos(k)]}{T_{Sample\ \ Period}}$$

$$AngDot(k) = \frac{[Angle(k-1) - Angle(k)]}{T_{Sample\ \ Period}}$$

实时控制输出模块为 “Real Control”，其内部结构如图3-10 所示。

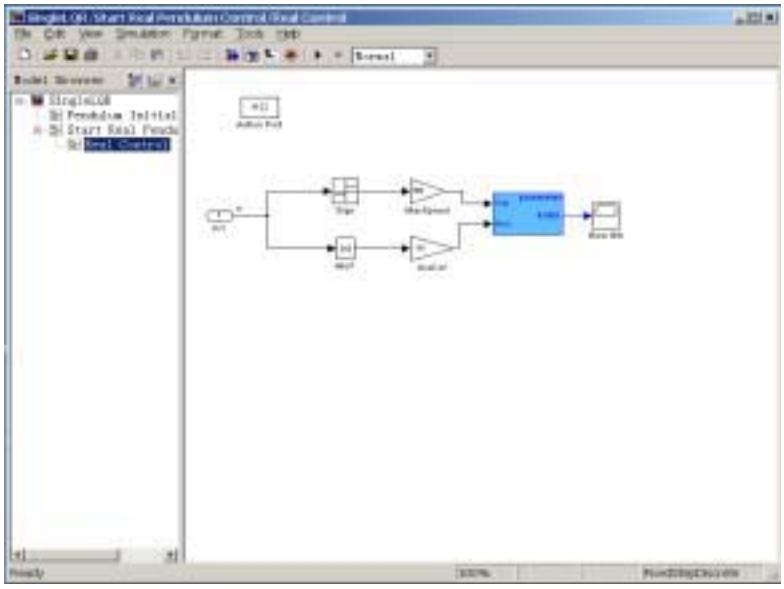


图3-10 实时控制输出模块

该模块通过逻辑判断触发 “Action Port”，将控制算法的结果包括理论运算的的速度和加速度输出到板卡。关于运动控制板卡的速度工作模式下功能调用请参见相关说明书。

控制算法模块将得到的系统输入运用控制理论的相关算法得出理论的系统输出，对于LQR 控制器，其算法为：

$$OutPut = Pos * Kx + PosDot * kx' + Angle * Ka + AngDot * Ka'$$

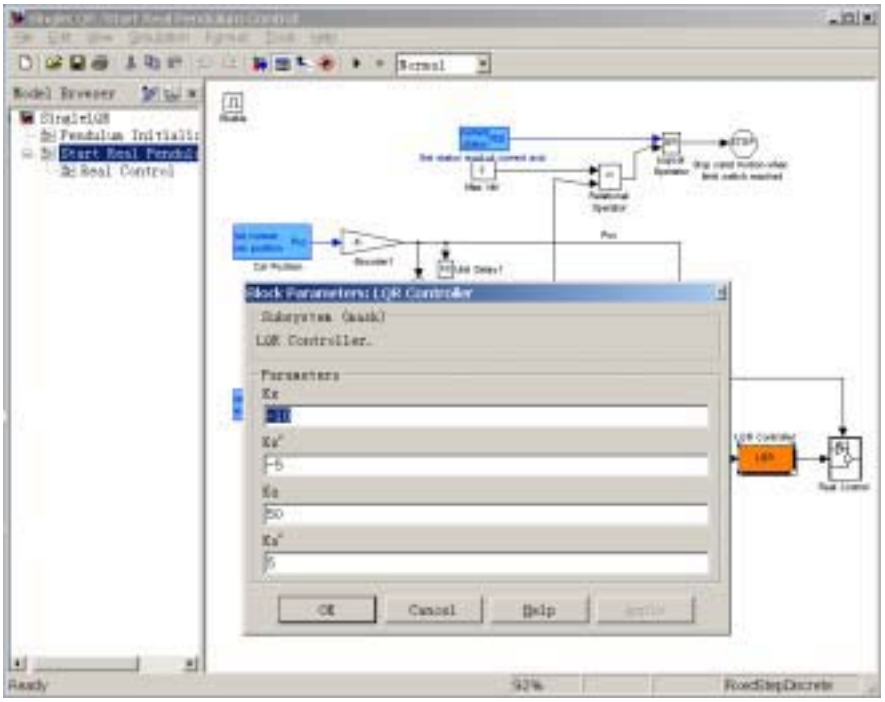


图3-11　LQR 控制器参数设置对话框

双击“LQR Controller”，设置控制器参数如下(图3-11 所示)：

其默认值为$[Kx \quad Kx' \quad Ka \quad Ka'] = [-10 \quad -5 \quad 50 \quad 5]$

点击“确定”按钮，使所设控制器参数生效。然后点击菜单条上“SIMULINK”菜单中的“Start”菜单。或者直接按“Ctrl－T”热键，系统开始进行实时控制。

其余示例程序全部类似。用户可以根据提示完成相应的实验。关于在 SIMULINK 环境下实时控制的相关主题请参看 MATLAB 联机帮助。

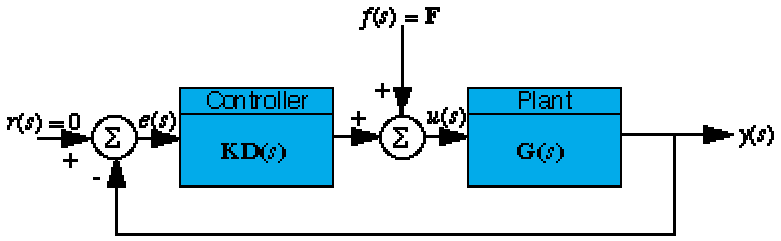
第四章 经典控制理论实验

经典控制理论的研究对象主要是单输入单输出的系统，控制器设计时一般需要有关于被控对象的较精确模型。PID 控制器因其结构简单，容易调节，在工业控制上应用较广。根轨迹法通过研究系统开环传递函数的零极点，设计控制器从而将闭环系统的传递函数极点设计在希望的位置上，从而满足设计者对于系统闭环稳态性能和瞬态性能的要求。频域响应法通过研究系统开环传递函数的频域响应，依据 nyquist 图或者 bode 图设计控制器从而使闭环系统满足设计者对于频域性能的要求。前面我们已经得到了倒立摆系统的比较精确的动力学模型，下面我们针对直线型一级倒立摆系统摆杆的平衡控制应用上述三种方法设计控制器。

实验一　PID 控制器设计与调节

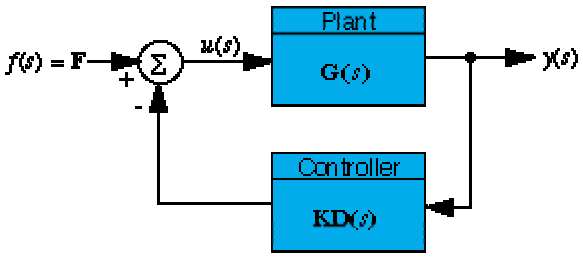
理论分析

首先，对于倒立摆系统输出量为摆杆的角度，它的平衡位置为垂直向上。系统控制结构框图如下：



图中 $KD(s)$ 是控制器传递函数， $G(s)$ 是被控对象传递函数。

考虑到输入 $r(s)=0$，结构图可以很容易的变换成



该系统的输出为

$$y(s)=\frac{G(s)}{1+KD(s)G(s)}F(s)=\frac{\dfrac{num}{den}}{1+\dfrac{(numPID)(num)}{(denPID)(den)}}F(s)$$

$$=\frac{num(denPID)}{(denPID)(den)+(numPID)(num)}F(s)$$

其中， num ——被控对象传递函数的分子项

den ——被控对象传递函数的分母项

$numPID$ ——PID 控制器传递函数的分子项

$denPID$ ——PID 控制器传递函数的分母项

被控对象的传递函数是

$$\frac{\Phi(s)}{U(s)}=\frac{\frac{ml}{q}s^2}{s^4+\frac{b(I+ml^2)}{q}s^3-\frac{(M+m)mgl}{q}s^2-\frac{bmgl}{q}s}=\frac{num}{den}$$

其中 $q=[(M+m)(I+ml^2)-(ml)^2]$

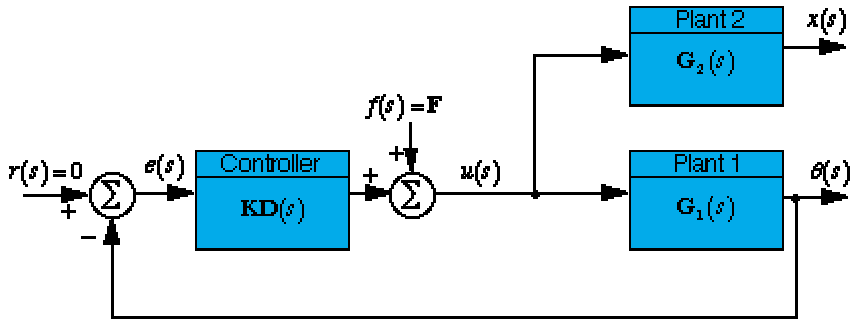
PID 控制器的传递函数为

$$KD(s)=K_Ds+K_P+\frac{K_I}{s}=\frac{K_Ds^2+K_Ps+K_I}{s}=\frac{numPID}{denPID}$$

需仔细调节 PID 控制器的参数，以得到满意的控制效果。

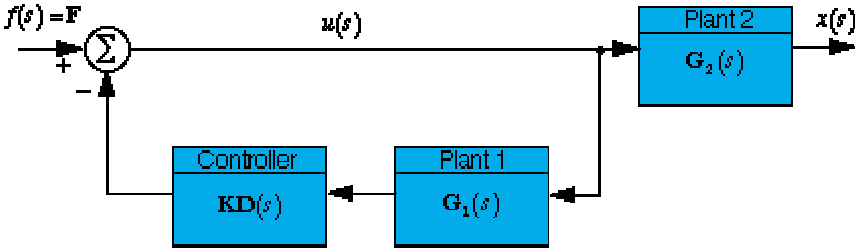
前面的讨论只考虑了摆杆角度，那么，在我们施加控制的过程中，小车位置如何变化呢？

考虑小车位置，得到改进的系统框图如下：



其中，$G_1(s)$ 是摆杆传递函数，$G_2(s)$ 是小车传递函数。

由于输入信号 $r(s)=0$，所以可以把结构图转换成：



其中，反馈环代表我们前面设计的控制器。

小车位置输出为：

$$X(s)=\frac{G_2(s)}{1+KD(s)G_1(s)}F(s)=\frac{\frac{num_2}{den_2}}{1+\frac{(numPID)(num_1)}{(denPID)(den_1)}}F(s)$$
$$=\frac{(num_2)(denPID)(den_1)}{(denPID)(den_1)(den_2)+(numPID)(num_1)(den_2)}F(s)$$

其中，num_1，den_1，num_2，den_2 分别代表被控对象 1 和被控对象 2 传递函数的分子和分母。*numPID* 和 *denPID* 代表 PID 控制器传递函数的分子和分母。

下面我们来求 $G_2(s)$，根据前面的推导：

$$X(s)=[\frac{(I+ml^2)}{ml}-\frac{g}{s^2}]\Phi(s)$$

可以推出小车位置的传递函数为

$$G_2(s)=\frac{X(s)}{U(s)}=\frac{\frac{(I+ml^2)}{q}s^2-\frac{mgl}{q}}{s^4+\frac{b(I+ml^2)}{q}s^3-\frac{(M+m)mgl}{q}s^2-\frac{bmgl}{q}s}$$

其中 $q=[(M+m)(I+ml^2)-(ml)^2]$

可以看出，$den_1=den_2=den$，小车的传递函数可以简化成：

$$X(s)=\frac{(num_2)(denPID)}{(denPID)(den\ )+k(numPID)(num_1)}F(s)$$

实验步骤

1. 根据建模结果仔细计算并寻找合适的理论 PID 控制参数。
2. 进入 matlab command 窗口，键入 pll-pid.m，进行仿真实验。通过调节 PID 参数请仔细观察思考控制器参数对系统瞬态响应和稳态响应的影响。找到几组合适的控制器参数作为实际控制的参数。
3. 进入 matlab simulink 窗口，点击 Inverted Pendulum Toolbox,在其右边点击 Single Pendulum PID Control Demo,进行仿真实验。在 Start Real Pendulum Control/PID 模块修改 PID 控制器参数以及控制周期。
4. 实控前，请仔细检查系统硬件连接，仔细阅读使用说明书。

5. 通过调整参数可以控制摆杆竖直向上，此时可能需用手轻轻扶一下摆杆，以避免小车“撞墙”。
6. 如果控制效果不理想，调整控制器参数，直到获得较好的控制效果。
7. 认真完成实验并提交试验报告。分析理论结果与实际结果的差异。

实验二　根轨迹法控制器设计与调节

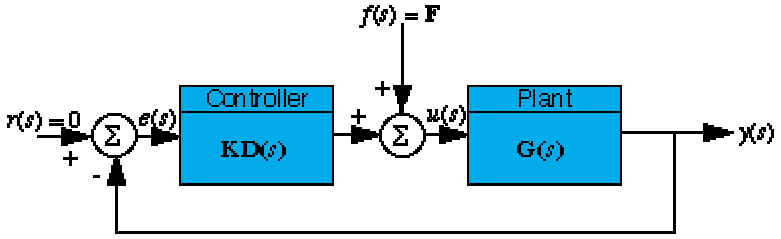
理论分析

前面我们已经得到了倒立摆系统的开环传递函数，输入为对小车的推力，输出为倒立摆系统摆杆的角度，被控对象的传递函数是

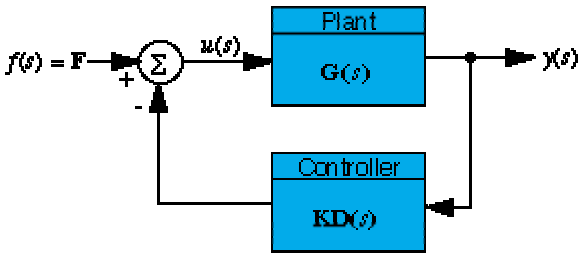
$$\frac{\Phi(s)}{U(s)}=\frac{\frac{ml}{q}s}{s^3+\frac{b(I+ml^2)}{q}s^2-\frac{(M+m)mgl}{q}s-\frac{bmgl}{q}}$$

其中 $q=[(M+m)(I+ml^2)-(ml)^2]$

给系统施加脉冲扰动，输出量为摆杆的角度时，系统框图如下：



考虑到输入 $r(s)=0$，结构图可以很容易的变换成



该系统的输出为

$$y(s)=\frac{G(s)}{1+KD(s)G(s)}F(s)=\frac{\frac{num}{den}}{1+k\frac{(numlead)(numlag)(num)}{(denlead)(denlag)(den)}}F(s)$$

$$=\frac{num(denlead)(denlag)}{(denlead)(denlag)(den)+k(numlead)(numlag)(num)}F(s)$$

其中，num ——被控对象传递函数的分子项

den ——被控对象传递函数的分母项

$numlead$ 和 $denlead$ ——控制器超前环节传递函数的分子项

$numlag$ 和 $denlag$ ——控制器滞后环节传递函数的分子项和分母项

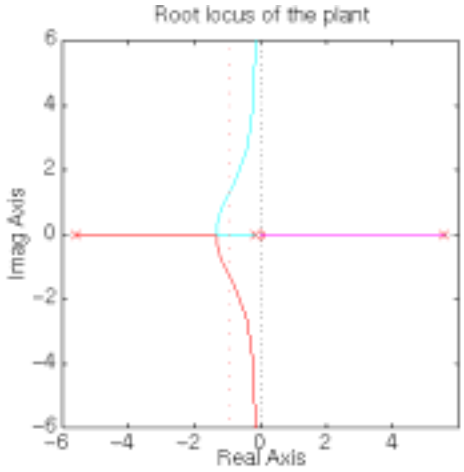
k ——控制器增益

闭环传递函数可以由 Matlab 程序求出。

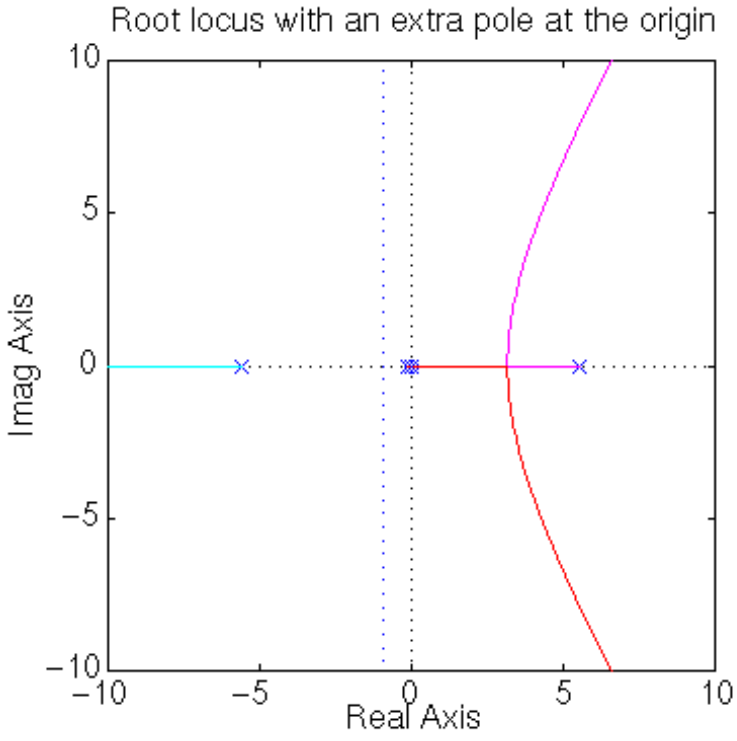
这里我们假设系统开环传递函数为

$$G(s)=\frac{4.5455s}{s^3+0.1818s^2-31.1818s-4.4545}$$

首先画出系统开环传递函数的根轨迹图如下，可以看出闭环传递函数的一个极点位于右半平面，并且有一条根轨迹起始于该极点，并沿着实轴向左跑到位于原点的零点处，这意味着无论增益如何变化，这条根轨迹总是位于右半平面，即系统总是不稳定的。



为了解决这个问题，我们在原点处增加一个额外的极点，使得原点处的零极点对消掉，可以绘出新的根轨迹。根轨迹如下图



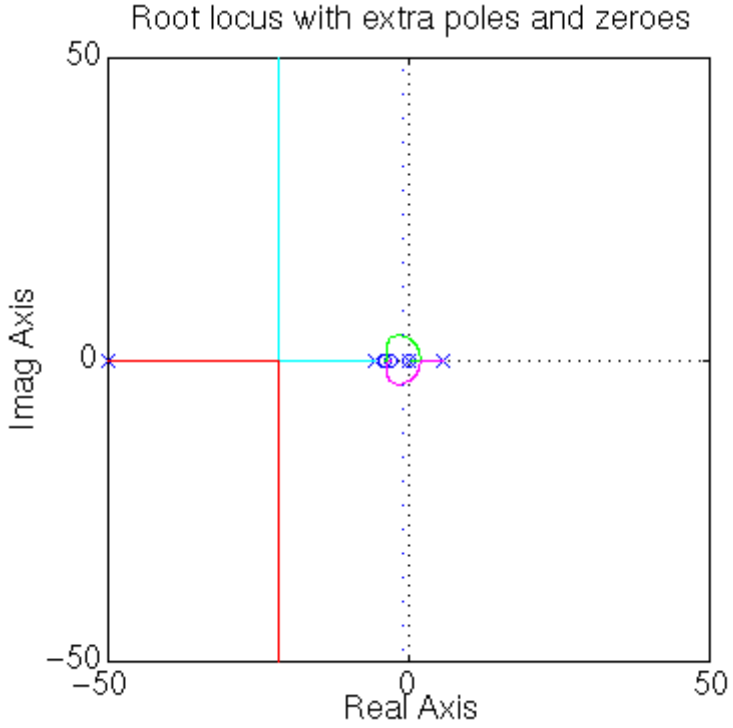
增加一个位于原点的极点后系统共有一个零点，四个极点，分别为

零点　　0

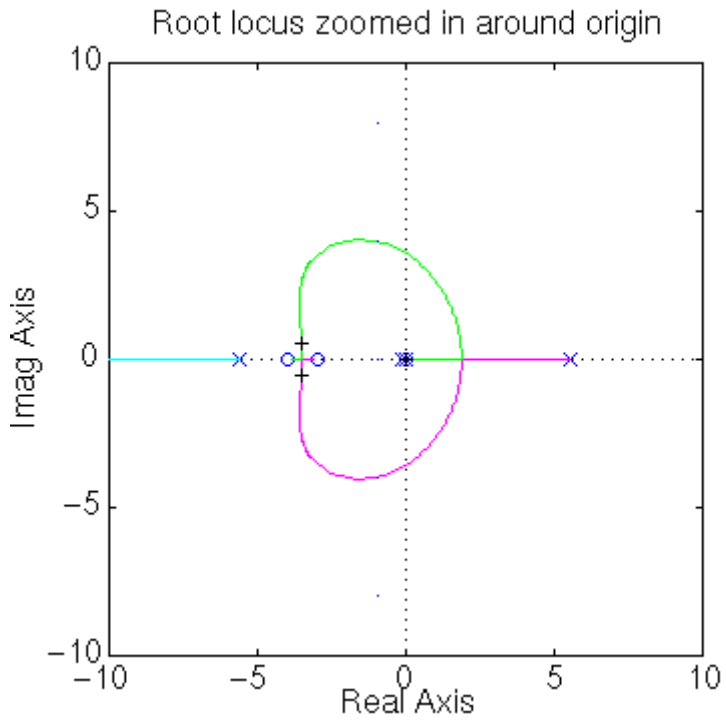
极点　　0　　-5.6041　　5.5651　　-0.1428

这说明有三根渐近线，一根在负实轴方向上，另外两根与第一根夹角为 120 度。在上面这种情况下，有两根根轨迹永远不会到达左半平面，我们必须通过在控制器中增加一个零点来把渐近线的数目减少到 2。但是如果只增加一个零点的话，渐近线的交点为[(-5.6041+5.5651-0.1428+0+0)-(0+0+z2)]/2，这意味着两根渐近线离开实轴的位置是-0.1-(1/2)z2，即使 z2 取得足够小（假定 z2 靠近原点），根轨迹处在左半平面的部分很少，很快就进入右半平面了（渐近线分离点约为-0.1），不能满足设计要求。

解决这个问题的方法是在左半平面再增加一个远离其他零极点的极点，为了保证渐近线的数目为 2，同时再增加一个零点，要求其中极点相对较大而零点相对较小，具体数值没有要求。经过多次实验后可以得到一组零极点（不是唯一的），校正后系统的根轨迹如下

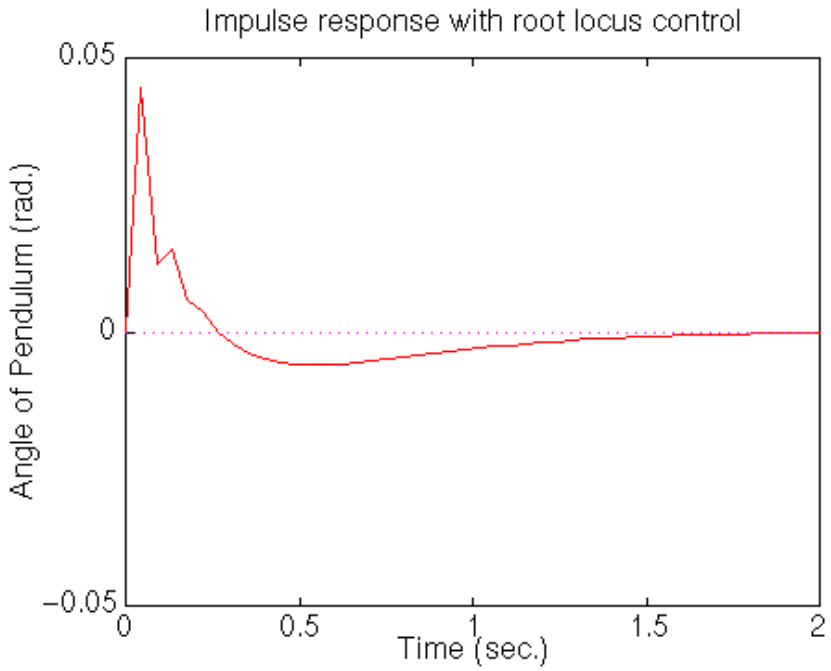


为了看得更清楚，改变坐标轴使图形放大：



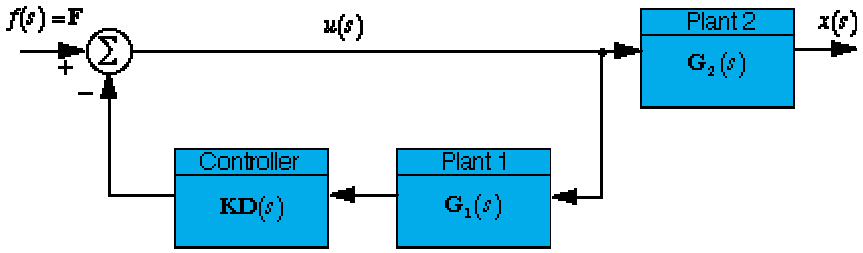
我们只要在根轨迹上选择一对位于左半平面的共扼复根和一个负实根，就可以得到稳定的系统。同时可以根据所选的根求解出系统增益，解出控制器的传递函数。

校正后摆杆位置的脉冲扰动的响应曲线如下图所示。可以看出系统响应满足指标要求。



前面讨论的输出量只考虑了摆杆位置，小车位置响应的推导如下：

根据前面的推导，得到改进的小车位置系统框图如下图：



小车位置输出为

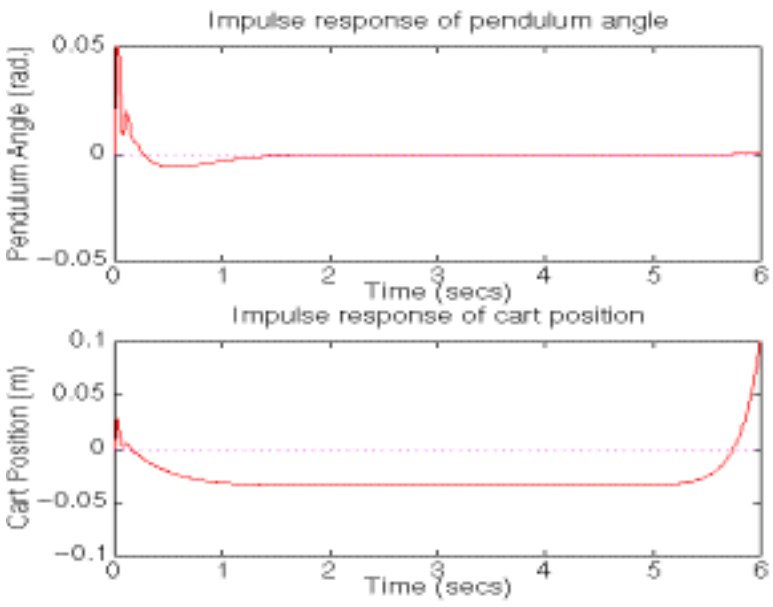
$$X(s)=\frac{G_2(s)}{1+KD(s)G_1(s)}F(s)=\frac{\frac{num_2}{den_2}}{1+k\frac{(numlead)(numlag)(num_1)}{(denlead)(denlag)(den_1)}}F(s)$$

$$=\frac{(num_2)(denlead)(denlag)(den_1)}{(denlead)(denlag)(den_1)(den_2)+k(numlead)(numlag)(num_1)(den2)}F(s)$$

其中，num_1，den_1，num_2，den_2 分别是被控对象 1（摆杆）和被控对象 2（小车）传递函数的分子和分母。可以看出，$den_1=den_2=den$，X 和 F 之间的传递函数可以简化成：

$$X(s)=\frac{(num_2)(denlead)(denlag)}{(denlead)(denlag)(den)+k(numlead)(numlag)(num_1)}F(s)$$

根据位置的传递函数和控制器传递函数可以求出小车位移。摆杆角度和小车位移的响应曲线如下：



实验步骤

1. 根据建模结果仔细计算并寻找合适的理论控制器参数。

2. 进入 matlab command 窗口，键入 pll-root.m，进行仿真实验。通过调节参数请仔细观察思考控制器参数对系统瞬态响应和稳态响应的影响。找到几组合适的控制器参数作为实际控制的参数。

注意：此程序中的对象参数请用户根据实际系统情况自行调节。程序中只是加入了两个零点 $-z_1,-z_2$、两个极点 $-p_1,-p_2$，请您根据自己的设计修改参数。

程序运行时，用户根据需要在根轨迹图中用鼠标选择轨迹上的一点。程序中 k 是控制器的增益，系统在该增益处对应的极点为 poles，selected_point 是您选择的位于根轨迹上的点，所有这些参数都显示在 Matlab Command 窗口中。

3. 用 Matlab 中的[NUMd,DENd] = C2DM(NUM,DEN,Ts,'method')命令把控制器传递函数离散化，并转换成标准形式

$$G(z)=\frac{b_0z^m+b_1z^{m-1}+\cdots+b_{m-1}+b_m}{z^n+a_1z^{n-1}+\cdots+a_{n-1}z+a_n}\ (m\le n)$$

4. 进入 matlab simulink 窗口，点击 Inverted Pendulum Toolbox,在其右边点击 Single Pendulum Transfer Function Control Demo,进行仿真实验。在 Start Real Pendulum Control/Transfer Function Controller 模块修改根轨迹法控制器参数以及控制周期。

注意：其中

$$numerater=\begin{bmatrix}b_0 & b_1 & b_2 & b_3 & b_4\end{bmatrix},denoninator=\begin{bmatrix}1 & a_1 & a_2 & a_3 & a_4\end{bmatrix}$$

5. 实控前，请仔细检查系统硬件连接，仔细阅读使用说明书。

6. 通过调整参数可以控制摆杆竖直向上，此时可能需用手轻轻扶一下摆杆，以避免小车“撞墙”。

7. 如果控制效果不理想，调整控制器参数，直到获得较好的控制效果。

8. 认真完成实验并提交试验报告。分析理论结果与实际结果的差异。

实验三　频域响应法控制器设计与调节

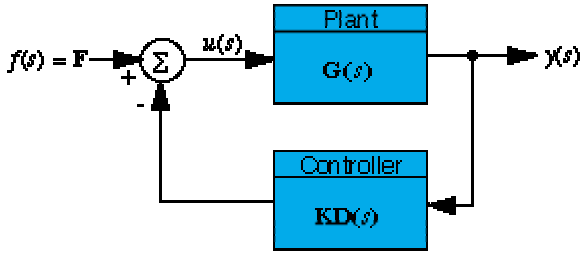
理论分析

前面我们已经得到了倒立摆系统的开环传递函数，输入为对小车的推力，输出为倒立摆系统摆杆的角度，被控对象的传递函数是

$$\frac{\Phi(s)}{U(s)}=\frac{\frac{ml}{q}s^2}{s^4+\frac{b(I+ml^2)}{q}s^3-\frac{(M+m)mgl}{q}s^2-\frac{bmgl}{q}s}$$

其中 $q=[(M+m)(I+ml^2)-(ml)^2]$

现在，我们用 Nyquist 图来设计倒立摆系统的控制器。引入控制器后系统的方框图如下：



闭环传递函数可以由 Matlab 程序求出。

这里我们假设系统在没有加控制器时开环传递函数为

$$G(s)=\frac{4.5455s}{s^3+0.1818s^2-31.1818s-4.4545}$$

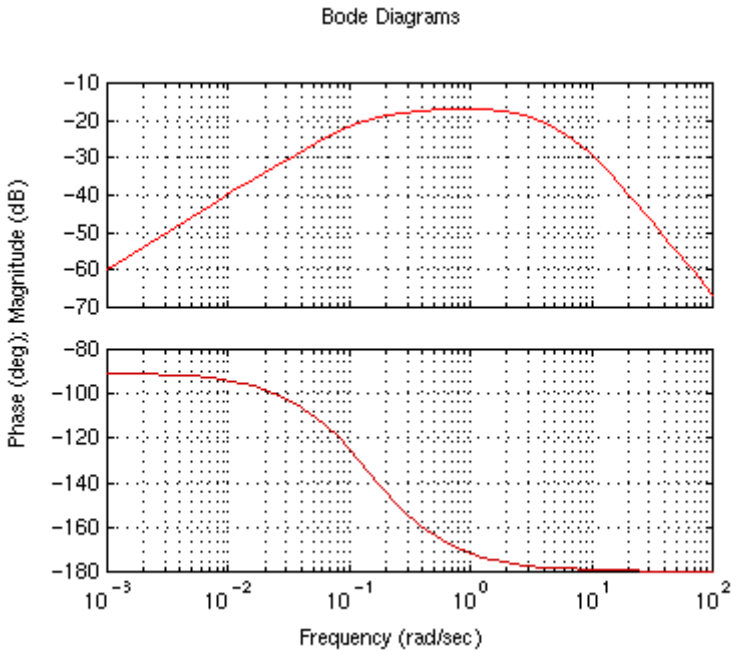
校正前系统的零极点为

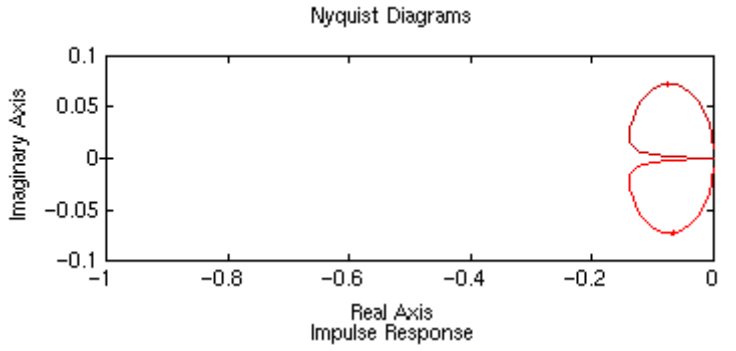
零点　　0　　0

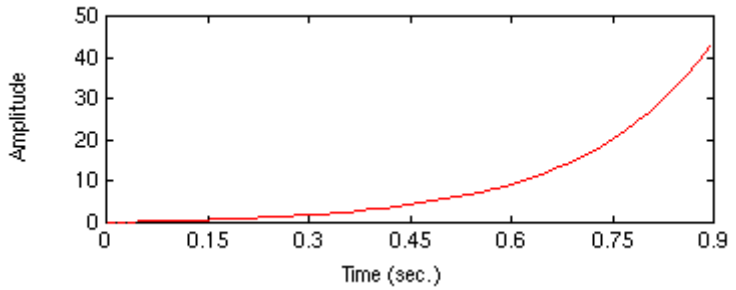
极点　　0　　-5.6041　　5.5651　　-0.1428

可以看出，其中一对位于原点的零极点可以对消，还有一个正实数极点位于右半 S 平面。根据 Nyquist 稳定性判据，闭环系统稳定的充分必要条件是：当 ω 从 $-\infty$ 连续增大到 $+\infty$ 时，$G(j\varpi)$ 沿逆时针方向包围-1 点 p 圈，这里 p 是开环传递函数在右半 S 平面的极点数目。此处 p 为 1，则 Nyquist 曲线要逆时针绕-1 点 1 圈，系统才稳定。

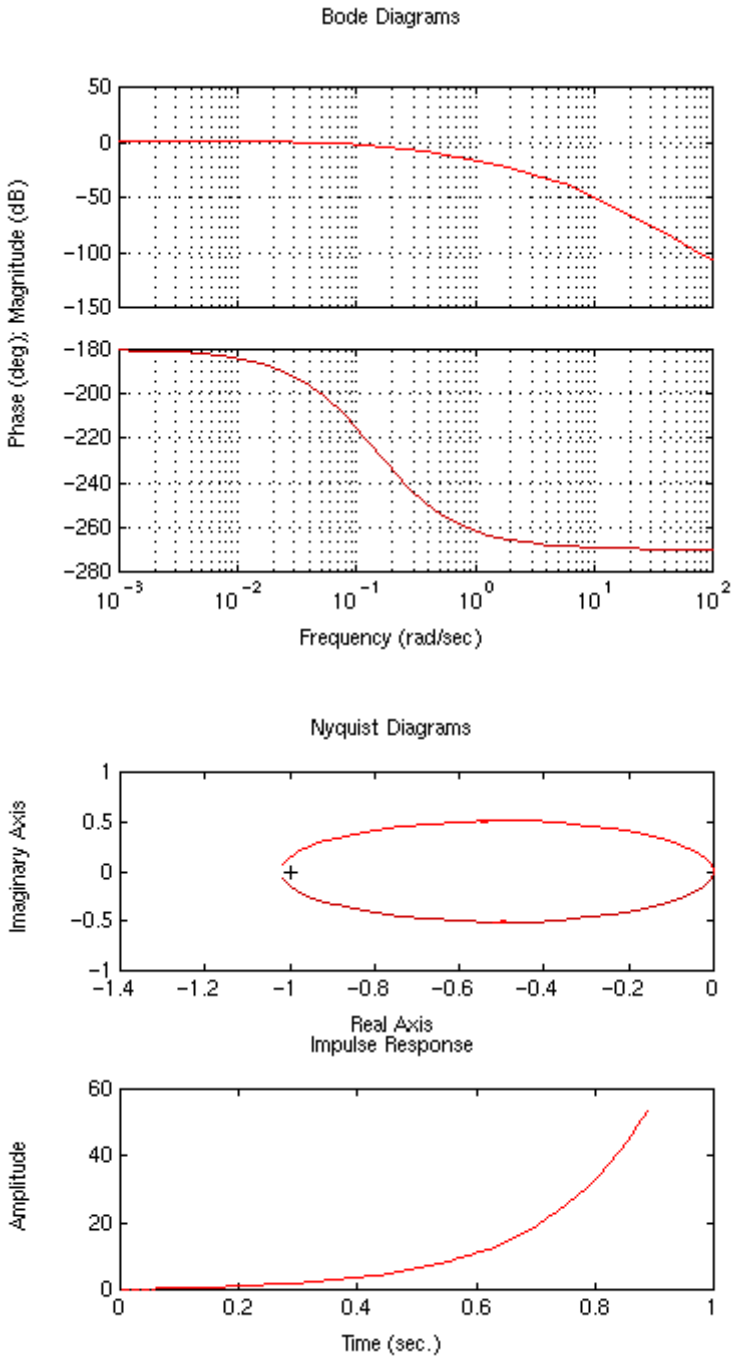
未校正系统的 Bode 图、Nyquist 图以及摆杆位置的脉冲响应曲线如下：



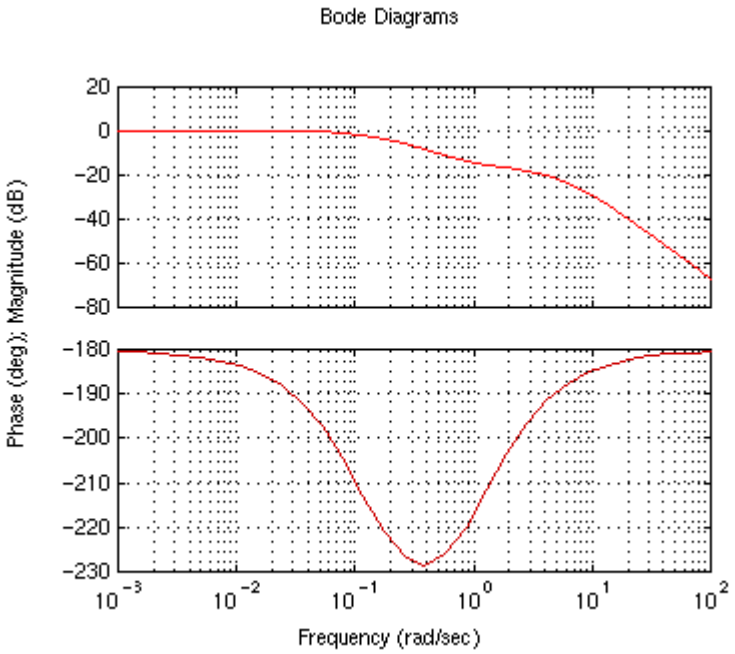


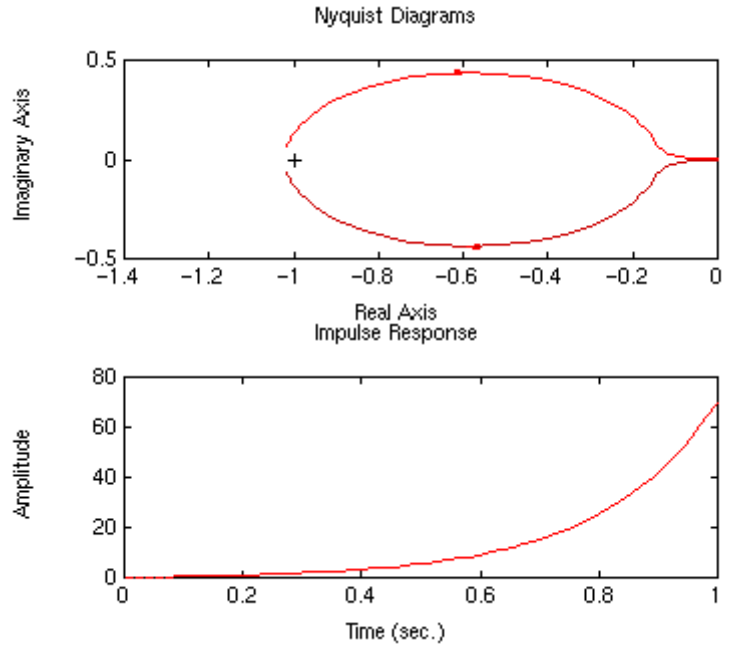


由于 Nyquist 曲线没有包围-1 点，所以闭环系统是不稳定的。为了使系统稳定，首先通过控制器引入一个积分环节，使得原点处的零点被对消掉（现在原点处共有两个零点和两个极点了），曲线就变成下图的情况：

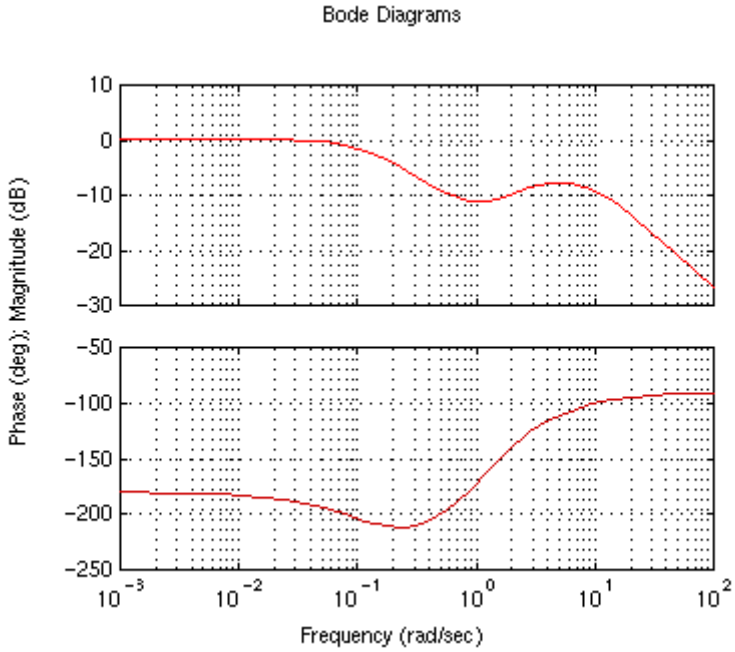


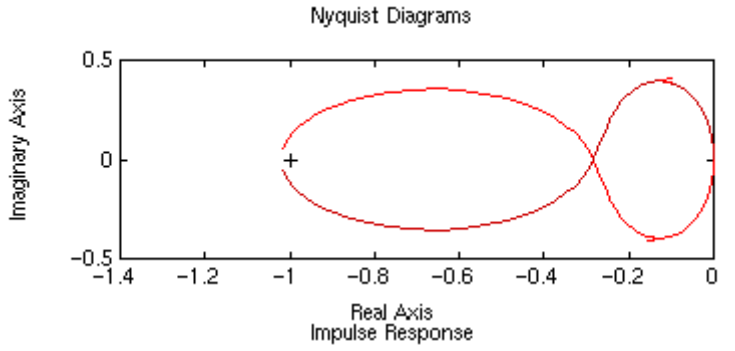
图中 Nyquist 曲线顺时针方向绕-1 点一圈。现在右半平面有两个极点，我们需要增加相位使曲线逆时针绕-1 点一周。增加零点可以增加相位，因此给控制器增加一个位于-1 点的零点，则图形如下：

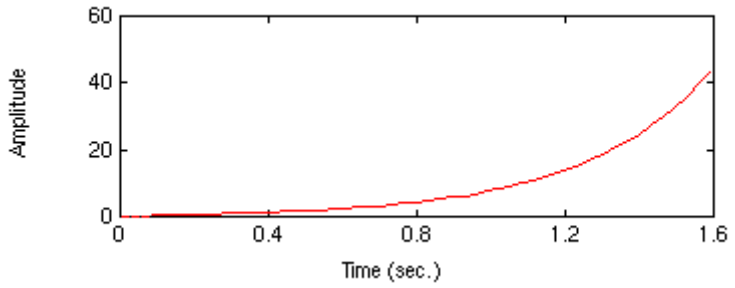




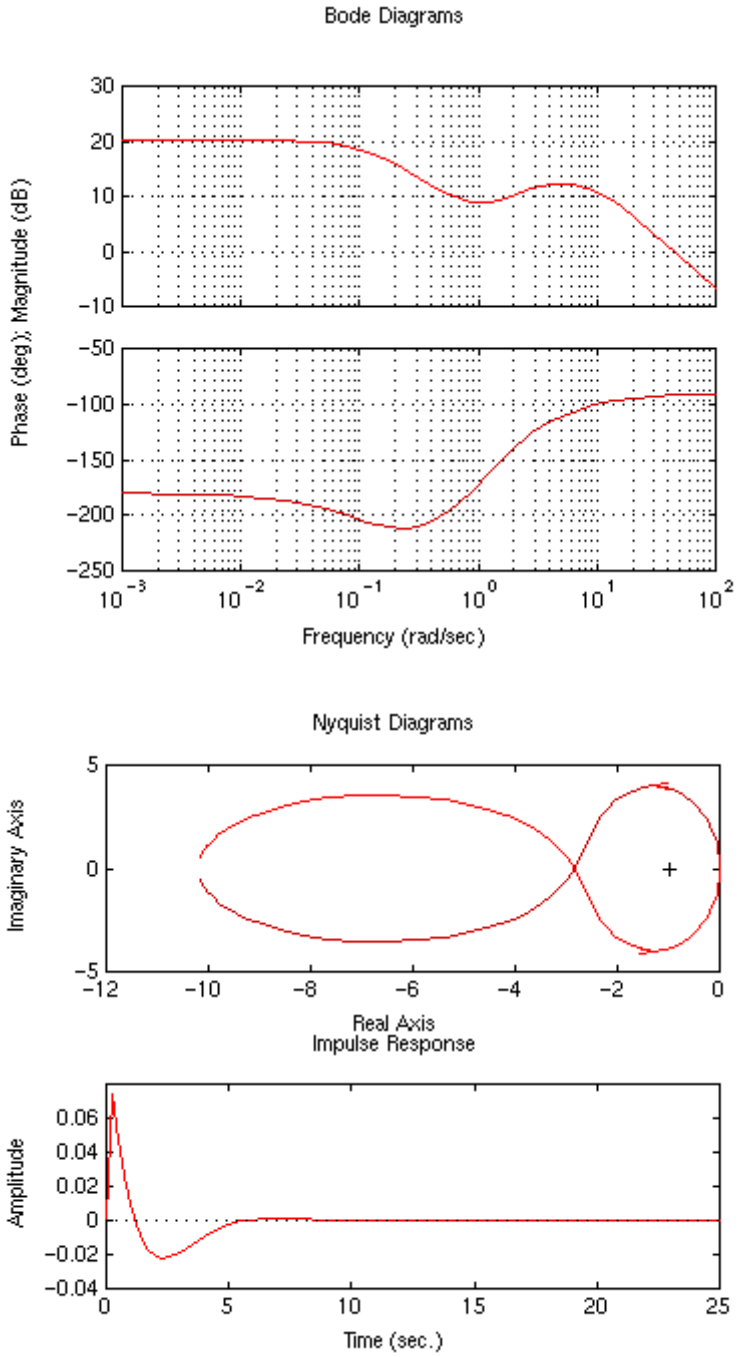
可以看出，相位仍然不够，绕-1 的曲线还是顺时针方向的。那么，我们再增加第二个零点，使得图形成为如下形式：



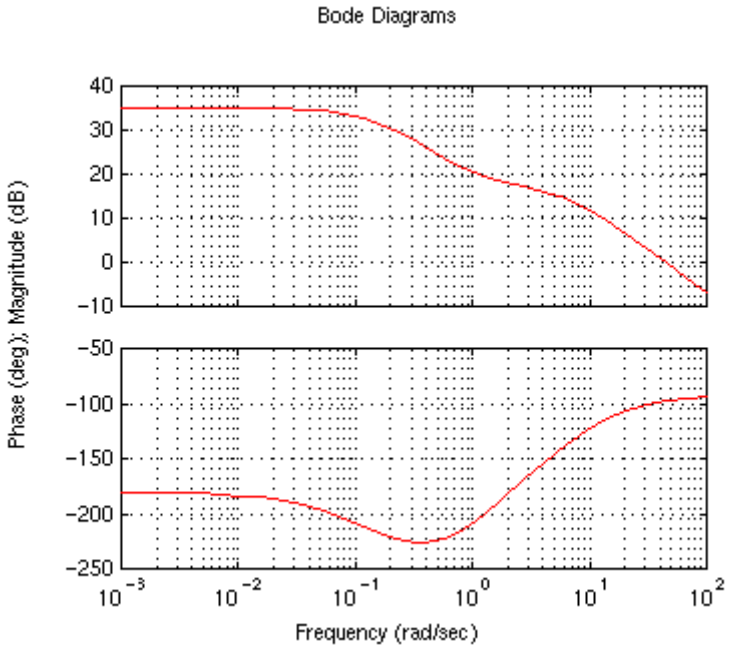


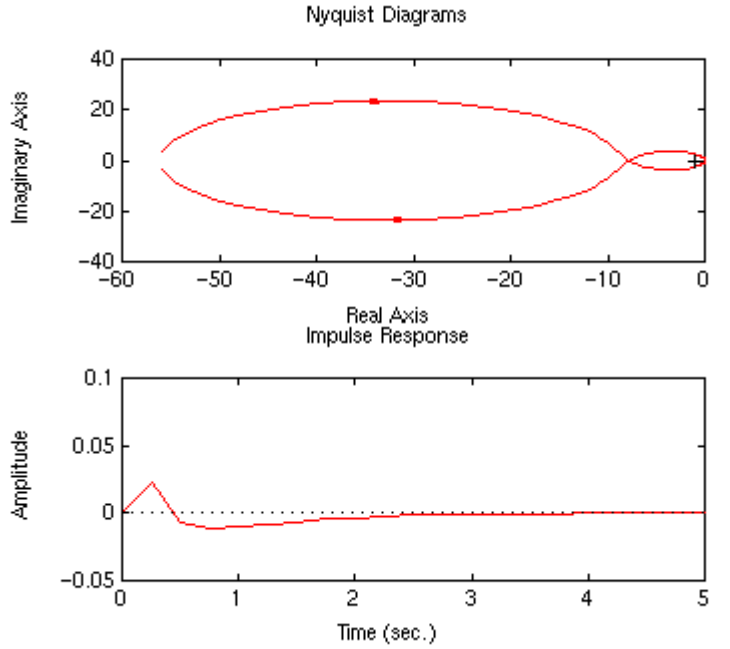


Nyquist 曲线仍然是顺时针绕-1 点。现在我们采用增加增益的方法，使得 Nyquist 曲线被向左侧拉伸，最终使曲线逆时针方向的圆圈围绕-1 点。图形如下：



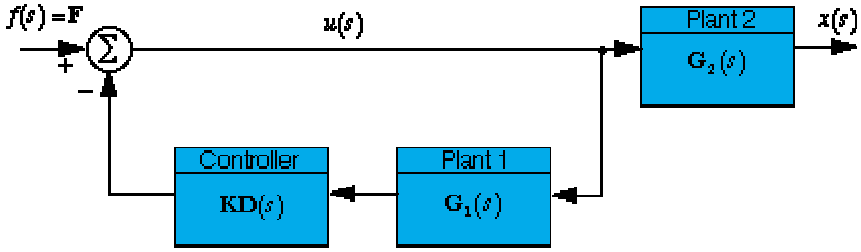
Nyquist 曲线逆时针绕-1 点一圈，系统是稳定的。然而如果要求系统的稳定时间小于 5 秒，则现有控制器不满足性能指标要求。我们通过修改控制器的极点来改善系统的响应。适当改变零极点组合，可以得到图形类似下图：在设计时，我们使 Bode 图变平滑，同时，你会注意到 Nyquist 曲线变得更椭圆了。





从响应曲线可以看出，系统的脉冲响应指标满足前面的设计要求。

和前面一样，小车位移的方框图和传递函数如下：



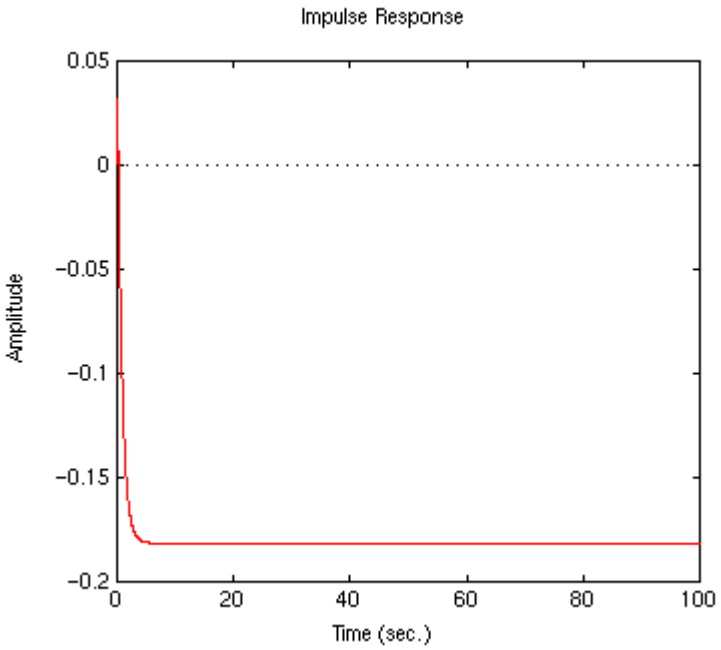
$$x(s)=\frac{G_2(s)}{1+KD(s)G_1(s)}F(s)=\frac{\frac{num_2}{den_2}}{1+\frac{k(numcontroller)(num_1)}{(dencontroller)(den_1)}}F(s)$$

$$=\frac{(num_2)(dencontroller)(den_1)}{(dencontroller)(den_1)(den_2)+k(numcontroller)(num_1)(den_2)}F(s)$$

化简后，$X(s)=\frac{(num_2)(dencontroller)}{(dencontroller)(den)+k(numcontroller)(num_1)}F(s)$

其中，*numcontroller* 和 *dencontroller* 分别为控制器的分子和分母。

小车位置响应曲线如下所示：



实验步骤

1. 据建模结果仔细计算并寻找合适的理论控制器参数。
2. 进入 matlab command 窗口，键入 pll-freq.m，进行仿真实验。通过调节参数请仔细观察思考控制器参数对系统瞬态响应和稳态响应的影响。找到几组合适的控制器参数作为实际控制的参数。

注意：此程序中的对象参数请用户根据实际系统情况自行调节。请您根据自己的设计修改控制器参数。

程序运行时，用户根据需要在 Matlab Command 窗口中输入控制器的传递函数及增益。然后就可以看到系统的 Nyquist 图及时域响应。

3. 用 Matlab 中的[NUMd,DENd] = C2DM(NUM,DEN,Ts,'method')命令把控制器传递函数离散化，并转换成标准形式

$$G(z)=\frac{b_0 z^m+b_1 z^{m-1}+\cdots+b_{m-1}+b_m}{z^n+a_1 z^{n-1}+\cdots+a_{n-1}z+a_n}\ (m\le n)$$

4. 进入 matlab simulink 窗口，点击 Inverted Pendulum Toolbox,在其右边点击 Single Pendulum Transfer Function Control Demo,进行仿真实验。在 Start Real Pendulum Control/Transfer Function Controller 模块修改控制器参数以及控制周期。

   注意：其中

$$numerater=\begin{bmatrix}b_0 & b_1 & b_2 & b_3 & b_4\end{bmatrix}, denoninator=\begin{bmatrix}1 & a_1 & a_2 & a_3 & a_4\end{bmatrix}$$

5. 实控前，请仔细检查系统硬件连接，仔细阅读使用说明书。
6. 通过调整参数可以控制摆杆竖直向上，此时可能需用手轻轻扶一下摆杆，以避免小车“撞墙”。
7. 如果控制效果不理想，调整控制器参数，直到获得较好的控制效果。
8. 认真完成实验并提交试验报告。分析理论结果与实际结果的差异。

第五章 现代控制理论实验—状态空间极点配置

经典控制理论的研究对象主要是单输入单输出的系统，控制器设计时一般需要有关于被控对象的较精确模型。现代控制理论主要是依据现代数学工具，将经典控制理论的概念扩展到多输入多输出系统。极点配置法通过设计状态反馈控制器将多变量系统的闭环系统极点配置在期望的位置上，从而使系统满足工程师提出的的瞬态和稳态性能指标。前面我们已经得到了倒立摆系统的比较精确的动力学模型，下面我们针对直线型一级倒立摆系统应用极点配置法设计控制器。

理论分析

前面我们已经得到了倒立摆系统的非线性动力学方程。

$$\begin{cases} \ddot{\theta} = \dfrac{mL(m+M)g}{I(m+M)+mML^2}\theta - \dfrac{mL}{I(m+M)+mML^2}u \\ \ddot{x} = -\dfrac{m^2L^2g}{I(m+M)+mML^2}\theta + \dfrac{I+mL^2}{I(m+M)+mML^2}u \end{cases}$$

以上式是两元联立二阶常微分方程，如果取状态变量为：

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \theta \\ \dot{\theta} \\ x \\ \dot{x} \end{bmatrix}$$

即摆杆的角度和速度以及小车的位置和速度四个状态变量。则系统状态方程为：

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \dfrac{mgL(M+m)}{I(M+m)+MmL^2}x_1 + \dfrac{-mL}{I(M+m)+MmL^2}u \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \dfrac{-m^2gL^2}{I(M+m)+MmL^2}x_1 + \dfrac{I+mL^2}{I(M+m)+MmL^2}u \end{cases}$$

将上式写成向量和矩阵的形式，就成为线性系统的状态方程

$$\dot{x} = Ax + Bu$$

$\dot{x}$ 是四维的状态向量，而系统矩阵 A 和输入矩阵 B 为下列形式

系统矩阵 A=$\begin{bmatrix} 0 & 1 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ b & 0 & 0 & 0 \end{bmatrix}_{4\times4}$　　输入矩阵 B=$\begin{bmatrix} 0 \\ c \\ 0 \\ d \end{bmatrix}_{4\times1}$

其中参数 a、b、c、d 为下列表达式确定的常数。

$$a=\frac{mL(m+M)g}{I(m+M)+mML^2} \qquad c=-\frac{mL}{I(m+M)+mML^2}$$

$$b=-\frac{m^2L^2g}{I(m+M)+mML^2} \qquad d=\frac{I+mL^2}{I(m+M)+mML^2}$$

选择摆杆的倾斜角度θ和小车的水平位置 x 作为倒立摆杆/小车系统的输出，则输出方程为

$$y=\begin{bmatrix}\theta\\ x\end{bmatrix}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix}=Cx$$

所谓状态反馈，就是用状态向量与一个系数矩阵的积作为控制向量

$$u=Kx$$

控制力 u 是一个加给小车水平方向的力 u，状态变量有四个，所以反馈系数是个 1X4 阶的矩阵

$$K=\begin{bmatrix}k_1 & k_2 & k_3 & k_4\end{bmatrix}$$

则系统状态反馈控制力可用状态变量与各自系数 k_1、k_2、k_3、k_4 乘积之和的形式表示，即

$$u=k_1x_1+k_2x_2+k_3x_3+k_4x_4$$

$$=k_1\theta+k_2\dot{\theta}+k_3x+k_4\dot{x}$$

状态反馈控制系统的方块图5-1 所示：

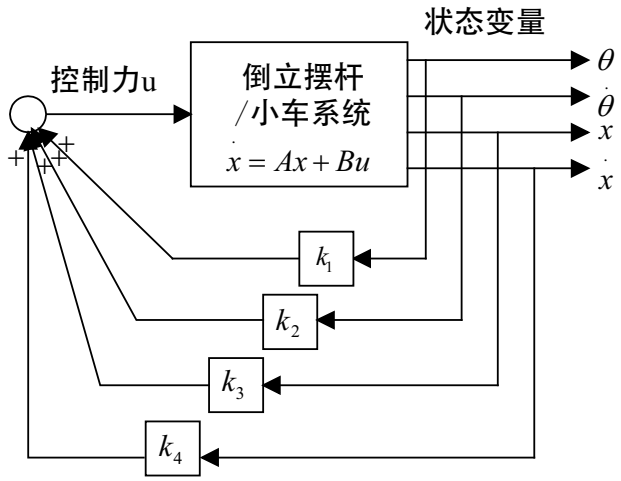


图5-1 状态反馈控制系统的方块图

根据极点配置法确定反馈系数

极点配置法是以线性系统为对象设计状态反馈控制器，使闭环控制系统的特征根（极点）分布在指定位置的控制器设计方法。

假定倒立摆杆/小车系统的参数如下：

摆杆的质量 m=0.07kg

长　　度 2L=0.4m　　$I=\frac{mL^2}{3}$

小车的质量 M=1.32kg

重力加速度 g=10 m/s^2

得到系统矩阵 A 和输入矩阵 B 为

$$A=\begin{bmatrix} 0 & 1 & 0 & 0 \\ 38.1825 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -0.3847 & 0 & 0 & 0 \end{bmatrix} \qquad B=\begin{bmatrix} 0 \\ -2.8037 \\ 0 \\ 0.7477 \end{bmatrix}$$

矩阵 A 的特征值是方程 $|Is-A|=0$ 的根：

$$|Is-A|=\begin{bmatrix} s & -1 & 0 & 0 \\ -38.1825 & s & 0 & 0 \\ 0 & 0 & s & -1 \\ -0.3847 & 0 & 0 & s \end{bmatrix}=0$$

因此，该系统的特征根 $s_1 \sim s_4$ 分别为

$$s_1, s_2, s_3, s_4 = 0, \quad 0, \quad 6.18, \quad -6.18$$

特征根之一 s_3 的实部是正值，所以该系统是不稳定的。由此可知：u=0 时，倒立摆系统是不稳定的系统。对这一不稳定系统应用状态反馈，可使摆杆垂直且使小车处于基准位置，即达到稳定状态。

在用状态方程表示的系统中，应用状态反馈构成的控制系统的特征根，以矩阵（A+BK）的特征值给出。系统稳定的充要条件是所有特征值都要处于复平面的左半平面。

矩阵（A+BK）的特征值是方程式 $|Is-(A+BK)|=0$ 的根：

$$\left|\begin{bmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \\ 0 & 0 & s & 0 \\ 0 & 0 & 0 & s \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 & 0 \\ a & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ b & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 \\ c \\ 0 \\ d \end{bmatrix} \times \begin{bmatrix} k_1 & k_2 & k_3 & k_4 \end{bmatrix}\right| = 0$$

这是 s 的四次代数方程式，可表示为

$$s^4 - (ck_2 + dk_4)s^3 - (a + ck_1 + dk_3)s^2 + (ad - bc)k_4 s + (ad - bc)k_3 = 0$$

适当选择反馈系数 k_1、 k_2、 k_3、 k_4，系统的特征根可以取得所希望的值。

把四个特征根 λ_1、 λ_2、 λ_3、 λ_4 设为四次代数方程式的根，则有

$$\begin{aligned} &s^4 - (\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)s^3 \\ &+ (\lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_3\lambda_4 + \lambda_4\lambda_1 + \lambda_1\lambda_3 + \lambda_2\lambda_4)s^2 \\ &- (\lambda_1\lambda_2\lambda_3 + \lambda_2\lambda_3\lambda_4 + \lambda_1\lambda_3\lambda_4 + \lambda_4\lambda_1\lambda_2)s \\ &+ \lambda_1\lambda_2\lambda_3\lambda_4 = 0 \end{aligned}$$

比较两式有下列联立方程式

$$\begin{aligned} &ck_2 + dk_4 = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 \\ &-(a + ck_1 + dk_3) = \lambda_1\lambda_2 + \lambda_2\lambda_3 + \lambda_3\lambda_4 + \lambda_4\lambda_1 + \lambda_1\lambda_3 + \lambda_2\lambda_4 \\ &-(ad - bc)k_4 = \lambda_1\lambda_2\lambda_3 + \lambda_2\lambda_3\lambda_4 + \lambda_1\lambda_3\lambda_4 + \lambda_4\lambda_1\lambda_2 \\ &(ad - bc)k_3 = \lambda_1\lambda_2\lambda_3\lambda_4 \end{aligned}$$

如果给出的 λ_1、 λ_2、 λ_3、 λ_4 是实数或共轭复数，则联立方程式的右边全部为实数。据此可求解出实数 k_1、 k_2、 k_3、 k_4。

当将特征根指定为下列两组共轭复数时

$$\lambda_1,\ \lambda_2,\ \lambda_3,\ \lambda_4 = -1 \pm 2j,\ -2 \pm j$$

利用方程式可列出关于 k_1、 k_2、 k_3、 k_4 的方程组：

$$\begin{aligned} &-2.8037k_2 + 0.7477k_4 = -6 \\ &-38.1925 + 2.8037k_1 - 0.7477k_3 = 18 \end{aligned}$$

$$\begin{aligned} &-27.4766k_4 = -30 \\ &27.4766k_3 = 25 \end{aligned}$$

求解后得

$$k_1 = 20.2846 \qquad k_2 = 2.4316$$
$$k_3 = 0.9099 \qquad k_4 = 1.0918$$

即施加在小车水平方向的控制力 u：

$$u = 20.2846\theta + 2.4316\dot{\theta} + 0.9099x + 1.0918\dot{x}\,[\mathrm{N}]$$

上式给出的状态反馈控制器，可以使处于任意初始状态的系统稳定在平衡状态，即所有的状态变量θ、$\dot{\theta}$、x及$\dot{x}$都可稳定在零的状态。这就意味着即使在初始状态或因存在外扰时，摆杆稍有倾斜或小车偏离基准位置，依靠该状态反馈控制也可以使摆杆垂直竖立，使小车保持在基准位置。

相对平衡状态的偏移，得到迅速修正的程度要依赖于指定的特征根的值。一般来说，将指定的特征根配置在原点的左侧，离原点越远，控制动作就越迅速，但相应地需要更大的控制力和快速的灵敏度。

下表所示为在复平面上指定的特征根配置成三种情况分别得到的反馈系数k_1、k_2、k_3、k_4。这显示出特征根越往复平面的左侧配置，k_1、k_2、k_3、k_4的值就越大这一倾向。

符号	指定特征值	反馈系数
①×	$\lambda_{1,2} = -1 \pm 2j$ $\lambda_{3,4} = -2 \pm j$	$k_1 = 20.2846, k_2 = 2.4316$ $k_3 = 0.9099, k_4 = 1.0918$
②●	$\lambda_{1,2} = -4 \pm 3j$ $\lambda_{3,4} = -5 \pm j$	$k_1 = 66.6537, k_2 = 10.8650$ $k_3 = 23.6565, k_4 = 16.6687$
③Θ	$\lambda_{1,2} = -7.4527 \pm 9.666j$ $\lambda_{3,4} = -3.1538 \pm 1.8334j$	$k_1 = 124.1694, k_2 = 18.3899$ $k_3 = 70.7099, k_4 = 40.5897$

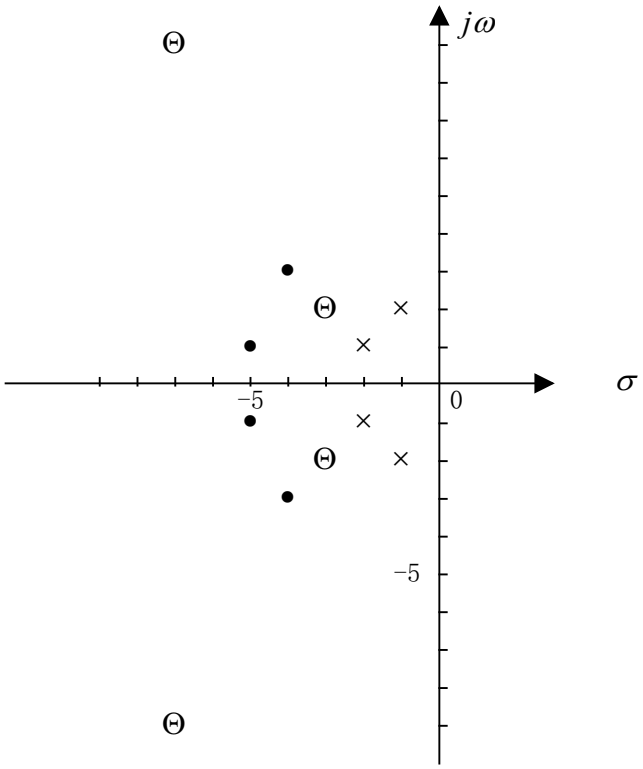


图5-2 极点配置法指定的特征根和得到的反馈系数

总之，用极点配置法指定的特征根和用极点配置法得到的控制系统的特性两者密切相关，在实际控制系统中，若由极点配置法决定反馈系数，必须反复进行这样的仿真，以得到满足更具体的控制目标和硬件上的制约的控制系统。

实验步骤

1. 根据建模结果仔细计算并寻找合适的理论控制器参数。
2. 进入 matlab command 窗口，键入 pll-pole.m，进行仿真实验。通过调节参数请仔细观察思考控制器参数对系统瞬态响应和稳态响应的影响。找到几组合适的控制器参数作为实际控制的参数。

注意：此程序中的对象参数请用户根据实际系统情况自行调节。请您根据自己的设计修改控制器参数。

3. 进入 matlab simulink 窗口，点击 Inverted Pendulum Toolbox,在其右边点击 Single Pendulum Pole Placement Control Demo,进行仿真实验。在 Start Real Pendulum Control/Pole Placement Controller 模块修改控制器参数以及控制周期。
4. 实控前，请仔细检查系统硬件连接，仔细阅读使用说明书。

5. 如果控制效果不理想，调整控制器参数，直到获得较好的控制效果。

6. 认真完成实验并提交试验报告。分析理论结果与实际结果的差异。

第六章 最优控制理论—LQR 控制器设计与调节

最优控制理论主要是依据庞德里亚金的极值原理，通过对性能指标的优化寻找可以使目标极小的控制器。其中线性二次型性能指标因为可以通过求解 Riccatti 方程得到控制器参数，并且随着计算机技术的进步，求解过程变得越来越简便，因而在线性多变量系统的控制器设计中应用较广。利用线性二次型性能指标设计的控制器称作 LQR 控制器。前面我们已经得到了直线一级倒立摆系统的比较精确的动力学模型，下面我们针对直线型一级倒立摆系统应用 LQR 法设计与调节控制器，控制摆赶保持倒立平衡的同时，跟踪小车的位置。对于直线一级顺摆系统的最优控制器的设计，可以完全参考本章的方法。

理论分析

前面我们已经得到了直线一级倒立摆系统的系统状态方程。假定倒立摆系统状态方程为：

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.1818 & 2.6727 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.4545 & 31.1818 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 1.8182 \\ 0 \\ 4.5455 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} u$$

应用线性反馈控制器，控制系统结构如下图。图中 R 是施加在小车上的阶跃输入，四个

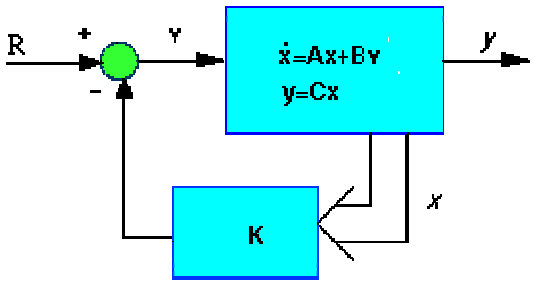


图6-1 线性反馈控制器

状态量 x、$\dot{x}$、θ 和 $\dot{\theta}$ 分别代表小车位移、小车速度、摆杆位置和摆杆角速度，输出 $y=[x,\theta]'$ 包括小车位置和摆杆角度。设计控制器使得当给系统施加一个阶跃输入时，摆杆会摆动，然

后仍然回到垂直位置，小车可以到达新的指定位置。

系统的开环极点可以用 Matlab 程序求出。开环极点为 0， -0.1428，5.5651，-5.6041，可以看出，有一个极点 5.5651 位于右半 S 平面，这说明开环系统不稳定。

假设全状态反馈可以实现（四个状态量都可测），找出确定反馈控制规律的向量 K。用 Matlab 中的 lqr 函数，可以得到最优控制器对应的 K。lqr 函数允许你选择两个参数——R 和 Q，这两个参数用来平衡输入量和状态量的权重。最简单的情况是假设 $R=1$，$Q=C'*C$。当然，也可以通过改变 Q 矩阵中的非零元素来调节控制器以得到期望的响应。(有关 lqr 的理论知识及具体设计思路可参考《现代工程控制》一书)

$$Q=C'*C=\begin{bmatrix}1&0&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix}$$

其中， $Q_{1,1}$ 代表小车位置的权重，而 $Q_{3,3}$ 是摆杆角度的权重，输入的权重 R 是 1。

下面来求矩阵 K，Matlab 语句为 $K=lqr(A,B,Q,R)$，求得

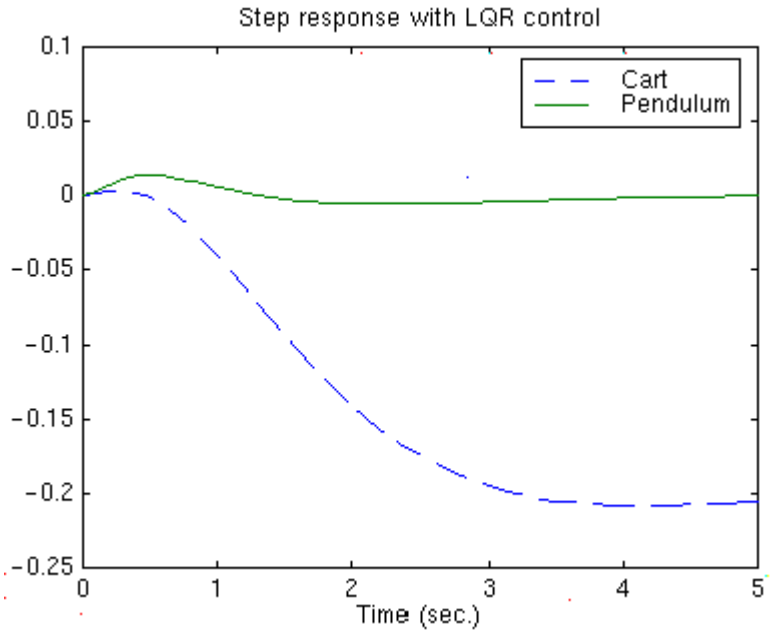


图6-2 LQR 控制的阶跃响应

$$K=[\ -1.0000\quad -1.6567\quad 18.6854\quad 3.4594]$$

LQR 控制的阶跃响应如图6-2 所示

其中，实线代表摆杆的角度，虚线代表小车位置。从图中可以看出，响应的超调量很

小，但稳定时间和上升时间偏大，小车的位置没有跟踪输入，而是向相反方向移动，这个问题留给同学们讨论，在这里我们来缩短稳定时间和上升时间。

可以发现，Q 矩阵中，增加 $Q_{1,1}$ 使稳定时间和上升时间变短，并且使摆杆的角度变化减小。这里取 $Q_{1,1}$=5000，　$Q_{3,3}$=100，则 $K=[\ -70.7107\ \ -37.8345\ \ 105.5298\ \ \ 20.9238]$，响应曲线如下：

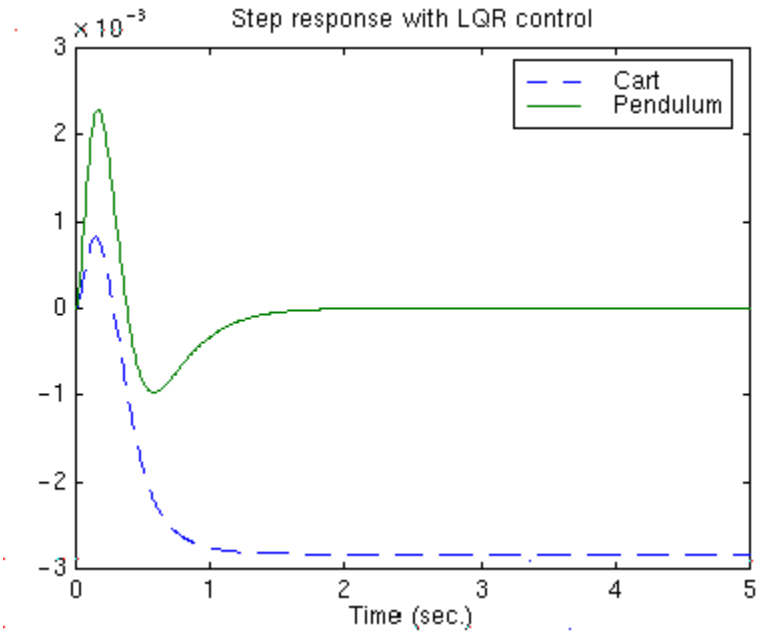


图6-3 LQR 控制的阶跃响应

如果再增大 $Q_{1,1}$ 和 $Q_{3,3}$，系统的响应还会改善。但在保证 $Q_{1,1}$ 和 $Q_{3,3}$ 足够小的情况下，

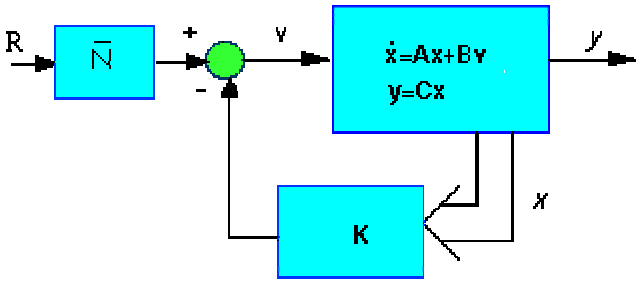


图6-4 LQR 控制器框图

系统响应已经满足要求了。

现在，要消除稳态误差。在前面的设计方法中，是把输出信号反馈回来乘以一个系数矩阵 K，然后与输入量相减得到控制信号。这就使得输入与反馈的量纲不一致，因此，为了使输入与反馈的量纲互相匹配，给输入乘以增益 $Nbar$，如图6-4 所示：

用函数 *rscale* 来计算 *Nbar* ： $Nbar = rscale(A, B, Cn, 0, K) = -70.7107$，可以看出，实际上 *Nbar* 和 *K* 向量中与小车位置 *x* 对应的那一项相等。此时系统的响应曲线如图6-5，小车位置跟踪输入信号，并且，摆杆超调足够小，稳态误差满足要求，上升时间和稳定时间

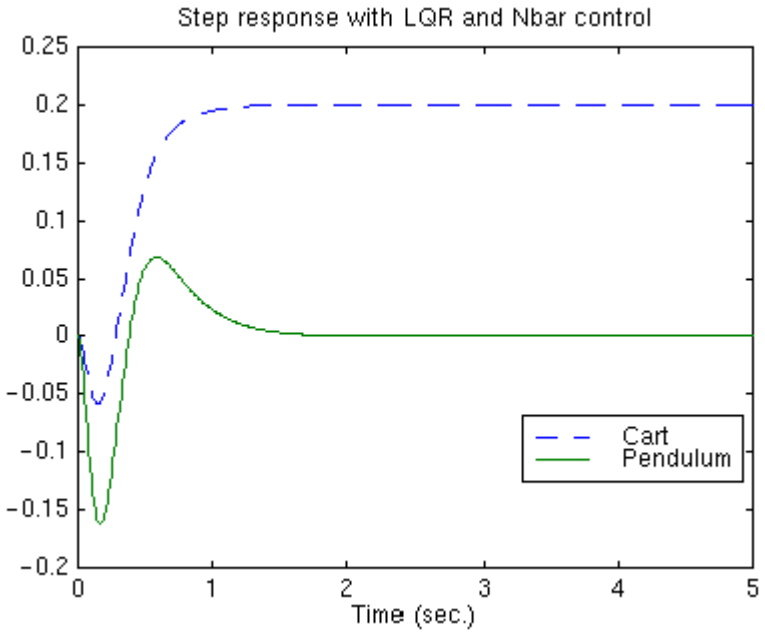


图6-5 系统响应曲线

也符合设计指标。

实验步骤

1. 根据建模结果仔细计算并寻找合适的理论控制器参数。
2. 进入 matlab command 窗口，键入 pll-lqr.m，进行仿真实验。通过调节参数请仔细观察思考控制器参数对系统瞬态响应和稳态响应的影响。找到几组合适的控制器参数作为实际控制的参数。

注意：此程序中的对象参数请用户根据实际系统情况自行调节。请您根据自己的设计修改控制器参数。

3. 入 matlab simulink 窗口，点击 Inverted Pendulum Toolbox，在其右边点击 Single Pendulum LQR Control Demo，进行仿真实验。在 Start Real Pendulum Control/LQR 模块修改控制器参数以及控制周期。
4. 实控前，请仔细检查系统硬件连接，仔细阅读使用说明书。
5. 如果控制效果不理想，调整控制器参数，直到获得较好的控制效果。

6. 认真完成实验并提交试验报告。分析理论结果与实际结果的差异。

第七章 PID 神经网络控制理论与实验

智能控制理论随着计算机技术的发展而逐步发展，因为智能控制器针对现实系统中的非线性、分布参数、不可建模等特性具有一定的泛化、自适应、抗干扰能力，工程技术人员和研究工作者对之非常关注。下面我们针对固高摆系统介绍 PID 神经网络控制器的控制算法原理。

理论分析

PID 神经网络控制器的结构如图7-1 所示。

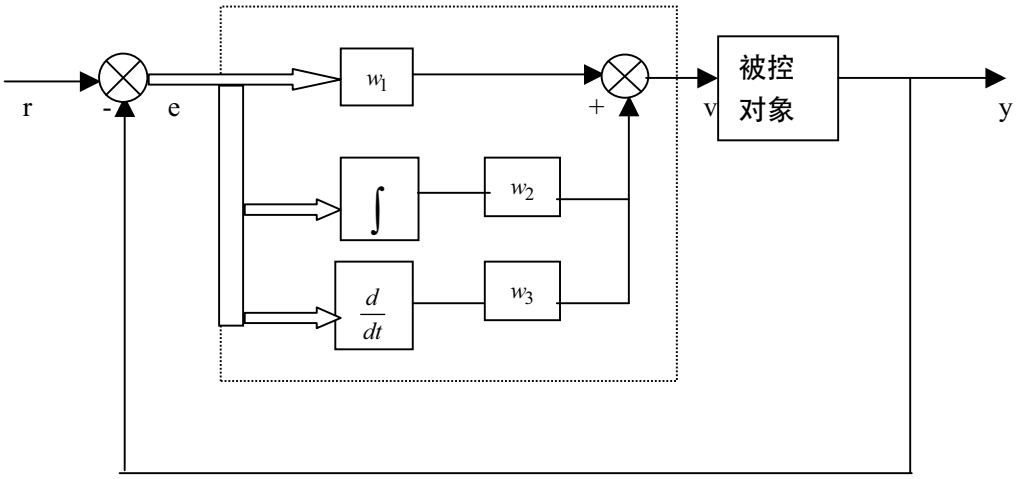


图7-1 PID 神经网络控制器结构图

PID 神经网络有三层结构，其结构如图：

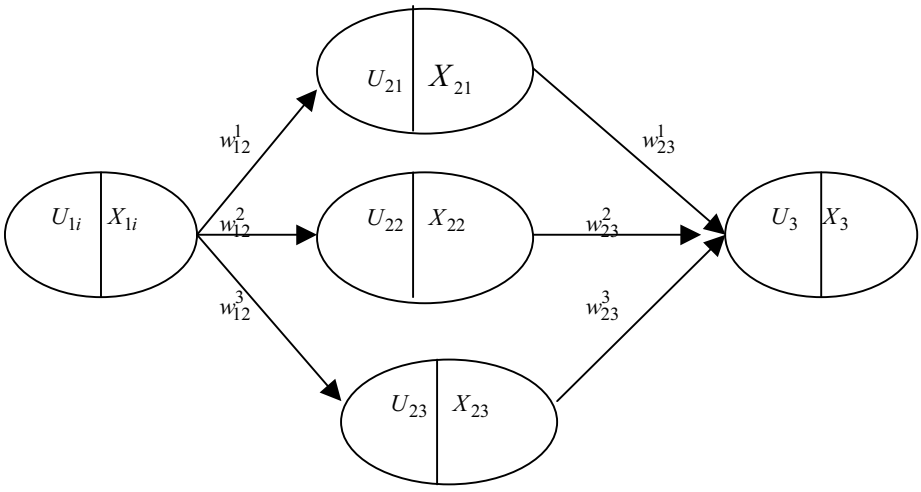


图7-2 PID 神经网络结构图

其中：

$$U_1(k) = e(k)$$

$$X_1(k) = \begin{cases} 1 & U_1(k) > 1 \\ U_1(k) & -1 \le U_1(k) \le 1 \\ -1 & U_1(k) < -1 \end{cases}$$

$$U_{2i}(k) = w_{12}^i(k) * X_1(k)$$

$$X_{21}(k) = \begin{cases} 1 & U_{21}(k) > 1 \\ U_{21}(k) & -1 \le U_{21}(k) \le 1 \\ -1 & U_{21}(k) < -1 \end{cases}$$

$$X_{22}(k) = \begin{cases} 1 & X_{22}(k) > 1 \\ X_{22}(k-1) + U_{21}(k) & -1 \le X_{22}(k) \le 1 \\ -1 & X_{22}(k) < -1 \end{cases}$$

$$X_{23}(k) = \begin{cases} 1 & X_{23}(k) > 1 \\ U_{23}(k) - U_{23}(k-1) & -1 \le X_{23}(k) \le 1 \\ -1 & X_{23} < (k) - 1 \end{cases}$$

$$U_3(k) = \sum_{i=1}^{3} w_{23}^i(k) * X_{2i}(k)$$

$$X_3(k) = \begin{cases} 1 & U_3(k) > 1 \\ U_3(k) & -1 \le U_3(k) \le 1 \\ -1 & U_3(k) < -1 \end{cases}$$

PID 神经网络的学习算法

采用 BP 算法，则可以得到

$$J = E = \frac{1}{M}\sum_{K=1}^{M} E_K = \frac{1}{M}\sum_{K}^{M} e^2(K)$$

根据 BP 算法

$$
\begin{aligned}
w_{23}^{i}(k+1) &= w_{23}^{i}(k) - \eta \frac{\partial J}{\partial w_{23}^{i}(k)} \\
&= w_{23}^{i}(k) - \eta \frac{\partial J}{\partial E_K} \frac{\partial E_K}{\partial y(k)} \frac{\partial y(k)}{\partial v(k)} \frac{\partial v(k)}{\partial X_3(k)} \frac{\partial X_3(k)}{\partial U_3(k)} \frac{\partial U_3(k)}{\partial w_{23}^{i}(k)} \\
&= w_{23}^{i}(k) - \eta\{-\frac{2}{M}\sum_{k=1}^{M}[r(k)-y(k)]\} * \frac{y(k+1)-y(k)}{v(k+1)-v(k)} * 1 * X_{2i}(k) \\
&= w_{23}^{i}(k) - \eta\sum_{k=1}^{M}\sigma_{23}(k) X_{2i}(k)
\end{aligned}
$$

其中 $\frac{\partial J}{\partial E_K}\frac{\partial E_K}{\partial y(k)} = -\frac{2}{M}\sum_{k=1}^{M}[r(k)-y(k)]$

$$
\frac{\partial y(k)}{\partial v(k)} = \frac{y(k+1)-y(k)}{v(k+1)-v(k)}
$$

$$
\frac{\partial v(k)}{\partial X_3(k)}\frac{\partial X_3(k)}{\partial U_3(k)} = 1
$$

$$
\frac{\partial U_3(k)}{\partial w_{23}^{i}(k)} = X_{2i}(k)
$$

$$
\sigma_{23}(k) = -\frac{2}{M}[r(k)-y(k)] * \frac{y(k+1)-y(k)}{v(k+1)-v(k)}
$$

$$
\begin{aligned}
w_{12}^{i}(k+1) &= w_{12}^{i}(k) - \eta \frac{\partial J}{\partial w_{12}^{i}(k)} \\
&= w_{12}^{i}(k) - \eta \frac{\partial J}{\partial E_K} \frac{\partial E_K}{\partial y(k)} \frac{\partial y(k)}{\partial v(k)} \frac{\partial v(k)}{\partial X_3(k)} \frac{\partial X_3(k)}{\partial U_3(k)} \frac{\partial U_3(k)}{\partial X_{2i}(k)} \frac{\partial X_{2i}(k)}{\partial U_{2i}(k)} \frac{\partial U_{2i}(k)}{\partial w_{12}^{i}(k)} \\
&= w_{12}^{i}(k) - \eta\{-\frac{2}{M}\sum_{k=1}^{M}[r(k)-y(k)]\} * \frac{y(k+1)-y(k)}{v(k+1)-v(k)} * \\
&\qquad w_{23}^{i}(k) * \frac{X_{2i}(k+1)-X_{2i}(k)}{U_{2i}(k+1)-U_{2i}(k)} * X_1(k) \\
&= w_{12}^{i}(k) - \eta\sum_{k=1}^{M}\sigma_{12}(k) X_1(k)
\end{aligned}
$$

其中$\frac{\partial J}{\partial E_K}\frac{\partial E_K}{\partial y(k)} = -\frac{2}{M}\sum_{k=1}^{M}[r(k)-y(k)]$

$$\frac{\partial y(k)}{\partial v(k)} = \frac{y(k+1)-y(k)}{v(k+1)-v(k)}$$

$$\frac{\partial v(k)}{\partial X_3(k)}\frac{\partial X_3(k)}{\partial U_3(k)} = 1$$

$$\frac{\partial U_3(k)}{\partial X_{2i}(k)} = w_{23}^{i}(k)$$

$$\frac{\partial X_{2i}(k)}{\partial U_{2i}(k)} = \frac{X_{2i}(k+1)-X_{2i}(k)}{U_{2i}(k+1)-U_{2i}(k)}$$

$$\frac{\partial U_{2i}(k)}{\partial w_{12}^{i}(k)} = X_1(k)$$

$$\sigma_{12}^{i}(k) = -\frac{2}{M}[r(k)-y(k)] * \frac{y(k+1)-y(k)}{v(k+1)-v(k)}\frac{X_{2i}(k+1)-X_{2i}(k)}{U_{2i}(k+1)-U_{2i}(k)} * w_{23}^{i}(k)$$

实验步骤

1. 实控前，请仔细检查系统硬件连接，仔细阅读使用说明书。
2. PID 控制参数的初始值在文件 Ctr.h 中设置，必要时可修改。
3. 如果不需要修改，可直接键入 PendNeural 启动程序
4. 选择合适的学习步长 yita（初始值已设为 0.001），
5. 根据屏幕提示进行控制，注意由于没有控制小车位置，所以需用手轻轻扶一下摆杆，以避免小车“撞墙”。这时可观察摆杆的平衡稳定性，比较与以前用传统 PID 控制时的区别。
6. 如果控制效果不理想，调整步长和网络权值的初始值，直到获得较好的控制效果。
7. 认真完成实验并提交试验报告。分析理论结果与实际结果的差异。

附录 1 固高摆控制系统的硬件和软件

A.1 控制系统硬件

固高摆控制系统硬件框图如图 A-1 所示，包括计算机、运动控制卡、伺服系统、倒立摆

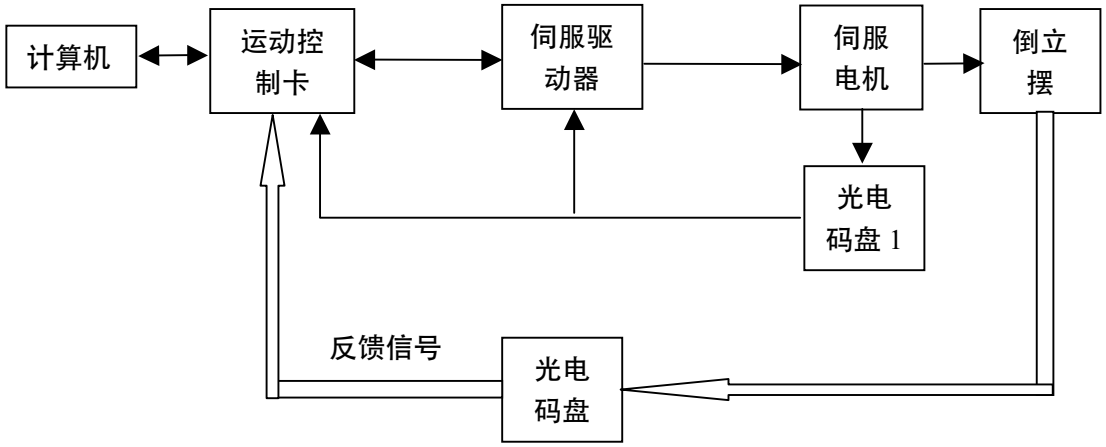


图 A-1 固高摆系统的硬件框图

本体和光电码盘反馈测量元件等几大部分，组成一个闭环系统。图中光电码盘 1 由伺服电机自带，对于直线型倒立摆，可以根据该码盘的反馈通过换算获得小车的位移，小车的速度信号可以通过差分法得到；对于环形倒立摆，则可以根据该码盘的反馈和机构的减速比直接求得转动小车的角位移。各个摆杆的角度由光电码盘测并直接反馈到控制卡，速度信号可以通过差分方法得到。计算机从运动控制卡中实时读取数据，确定控制决策（电机的输出力矩），并发送给运动控制卡。运动控制卡经过 DSP 内部的控制算法实现该控制决策，产生相应的控制量，使电机转动，带动小车运动，保持摆杆平衡。

A.2 控制系统软件

所有固高摆系统的控制软件均基于固高科技有限公司开发运动控制器而开发。除最初推出的部分倒立摆采用的是 GM-400 ISA 总线四轴伺服运动控制卡，其它摆均采用固高科技最新推出的 GT-400-SV PCI 总线四轴伺服运动控制卡。因此，本节只介绍基于 GT-400-SV-PCI 运动控制卡的控制系统软件。

基于 GT-400-SV-PCI 控制卡的控制系统软件分为 MATLAB 和 DOS 两个版本。MATLAB 版控制系统软件说明在第四章中已经做过说明，下面主要就固高摆的 DOS 控制软件进行说明。

DOS 版控制系统软件说明

DOS 版控制系统软件用标准 C 语言编写而成。有关 C 语言程序设计的内容可以参考各种介绍 C 语言的书籍。本节主要针对固高摆系统介绍如何基于 GT-400-SV-PCI 运动控制函数库编写控制软件。

基于 GT-400-SV-PCI 卡的 DOS 环境下的摆控制软件（演示软件）在 Borland C++3.1 平台上设计和编译，整个程序由六个模块组成，每个模块的功能描述如下：

Pend.cpp	系统软件主程序
Ctrl.cpp	实时控制模块
Sv_pci.cpp	GT-400-SV-PCI 板卡初始化模块
Chinese.cpp	汉字显示驱动程序
Draw.cpp	Dos 环境下采样数据显示
Face.cpp	Dos 环境下用户操作界面和交互命令处理

控制系统软件主程序

主要代码流程如下：

```
int main( )                    //C 语言程序入口
{
    sv_init();                 //初始化 GT-400-SV-PCI 运动控制板
    init_graph();              //初始化 DOS 环境下的视频显示为图形方式
    ReadLibrary();             //调入显示汉字库

    init_bar();                //创建和显示图形用户界面
    init_rect();
    display_panel();

    init_interrupt();          //初始化 DOS 定时器中断
    handle();                  //人机交互命令处理模块
    restore_interrupt();       //恢复 DOS 中断

    SaveLibrary();             //保存显示汉字库
    closegraph();              //关闭图形显示模式
    return 0;                  //程序结束返回
}
```

上述主程序调用了其它功能模块中定义的功能函数。下面就与运动控制器相关的功能模块以及与实时控制相关的功能模块进行详细说明，其它如创建和显示图形用户界面和人机交互命令处理在此不做详细说明，如果读者有兴趣，可以参考相关 C 语言教材。

GT-400-SV-PCI 板卡初始化模块

GT-400-SV-PCI 板卡初始化模块 Sv_pci.cpp 中定义了运动控制器的初始化函数 sv_init( )，其详细代码解释如下：

```
int sv_init( )                 //运动控制卡初始化函数入口
{
rtn=GT_Close();                //先关闭一次控制卡，确保下面函数正常执行
rtn=GT_Open();                 //打开控制卡，准备运行
rtn=GT_Reset();                //复位控制卡，确保控制卡能正常执行
```

```
rtn=GT_LmtSns(Snslmt);          //设置限位开关有效电平
rtn=GT_LmtsOn();                //打开限位开关自动监控功能
rtn=GT_EncSns(Snsenc);          //设置光电增量码盘反馈的计数方向
rtn=GT_SetSmplTm(SAM*1000000);  //设置运动控制卡伺服控制周期
rtn=GT_Axis(1);                 //设置当前工作轴为 1 轴，下面的命令只对 1 轴有效
rtn=GT_PrflT();                 //设置 1 轴的轨迹规规划方式为 T 型曲线模式
rtn=GT_ClrSts();                //清除 1 轴状态字
rtn=GT_SetKp(Kp);               //设置 1 轴滤波器的 P 参数
rtn=GT_SetKi(Ki);               //设置 1 轴滤波器的 I 参数
rtn=GT_SetKd(Kd);               //设置 1 轴滤波器的 D 参数
rtn=GT_SetKvff(Kvff);           //设置 1 轴滤波器的 Kvff(速度前馈)参数
rtn=GT_SetKaff(Kaff);           //设置 1 轴滤波器的 Kaff(加速度前馈)参数
rtn=GT_SetILmt(Ilmt);           //设置当前轴的伺服滤波器误差积分饱和值
rtn=GT_SetMtrBias(MBias);       //设置当前轴的伺服滤波器输出零点偏移值
rtn=GT_Update();                //当前轴参数更新
rtn=GT_ClrSts();                //清除 1 轴状态字
rtn=GT_SetPos(0);               //设置当前轴的目标位置
rtn=GT_SetVel(2);               //设置当前轴的目标速度
rtn=GT_SetAcc(0.3);             //设置当前轴的加速度
rtn=GT_SetJerk(0.2);            //设置当前轴的加加速度
rtn=GT_Update();                //当前轴参数更新
return 0;                       //返回
}
```

上述运动控制函数的详细功能说明和相关参数的定义请参考运动控制器编程手册。

实时控制模块 Ctrl.cpp 中定义了定义了定时器中断服务程序 handler，定时器中断初始化函数 init_interrupt()和中断恢复函数 restore_interrupt()。

```
const unsigned short INTR = 0X1C;    //系统中断服务程序地址
void interrupt(*oldhandler)(...);    //旧中断服务程序的句柄定义
```

定时器中断服务程序 handler

```
void interrupt handler(...)          //中断服务程序入口
{
    disable();                       //关闭中断（中断不能嵌套发生）
    if (mutex_event == 0)
        handle_pendulum();           //实时控制算法模块
    enable();                        //打开中断，允许下次中断发生
}
```

定时器中断初始化函数

```
void init_interrupt()
{
    disable();                  //关闭中断（在初始化中断过程中严禁产生中断）
    outportb(0x43,0x36);        //写中断控制字：0x43 为中断控制寄存器的内存映射地址
                                //写入 0x36 表示下面的字节用于设置 2 号实时时钟周期
    outportb(0x40,0xf6);        //写数据寄存器：0x40 为数据寄存器内存映射地址，写入的
    outportb(0x40,0x1b);        //数据为 16 位时钟细分比率，首先写入低字节 0xf6，
```

```
                                    //随后写入高字节 0x1b
        oldhandler = getvect(INTR);  //保存原有的中断服务程序句柄
        setvect(INTR, handler);  //设置新的中断服务程序句柄
        enable();                   //打开中断
    }
```

上述中断初始化代码创建一个 6 毫秒的定时器中断，即当打开中断后，将每隔 6 毫秒对摆系统进行一次实时控制。6 毫秒的定时时间是通过向数据寄存器写入 0x1bf6=7158 获得的，因为计算机系统的 2 号实时时钟频率为 1.193MHz，**1193000*0.006=7158**。

中断恢复函数

```
void restore_interrupt()
{
    disable();                      //关闭中断
    setvect(INTR,oldhandler);     //恢复旧的的中断服务程序
    enable();                       //打开中断
}
```

实时控制算法模块（以直线两级倒立摆为例）

```
void handle_pendulum()
{
    int i;
    long actl_pos;

    //分别采集摆杆 2、摆杆 1 和驱动电机的当前位置，并将它们转化为相应的量纲。
    //这些数据分别从运动控制器的 3、2、1 通道输入。
    GT_Axis(3);
    rtn=GT_GetAtlPos(&actl_pos);
    angle2 = -ENCODE2 * actl_pos;
    GT_Axis(2);
    rtn=GT_GetAtlPos(&actl_pos);;
    angle =    -ENCODE2 * actl_pos;
    GT_Axis(1);
    rtn=GT_GetAtlPos(&actl_pos);;
    pos = ENCODE1 * actl_pos;

    //根据上次控制周期的采样值和时钟周期计算各运动部件的速度，并更新变量
    posDot = (pos - pos0) / INTPERIOD;
    angleDot = (angle - angle0) / INTPERIOD;
    angleDot2 = (angle2 -angle02 ) / INTPERIOD;
    pos0 = pos;
    angle0 = angle;
    angle02 = angle2;

    //进行安全检查，如果发现异常（如速度过大等），返回
    if ( handle_safety() == -1 )    return;
```

```
    //规一化摆杆 1 角度，使之在 0~360 之间
    ang_2pi = angle;
    while (ang_2pi < 0)       {ang_2pi += 2 * M_PI;};
    while (ang_2pi >= (2*M_PI)) {ang_2pi -= 2 * M_PI;};

    //规一化摆杆 2 角度，使之在 0~360 之间
    ang_2pi2 = angle2;
    while (ang_2pi2 < -M_PI)     {ang_2pi2 += 2 * M_PI;};
    while (ang_2pi2 >= M_PI)     {ang_2pi2 -= 2 * M_PI;};
    ang_2pi2 += ang_2pi - M_PI;

    vel=0;    acc=0;          //初始化控制变量

    switch (start)            //根据不同的操作命令，进行不同的处理
    {
        case 1:
            enable_servo();     //使能伺服，并修改运动控制模式到速度控制模式
            start = 2;          //进入控制状态
            return;

        case 2://检测倒立摆是否在平衡位置
        if ((fabs(ang_2pi-M_PI) <= entryAngle) && (fabs(ang_2pi2) <= entryAngle) )
                anti_cran(); //如果是，计算保持在该平衡位置所需的控制量
            break;
        default:
            break;
    }

    rtn=GT_ClrSts();          //清除控制器状态字
    rtn=GT_SetVel(vel);       //设置所需的速度;
    rtn=GT_SetAcc(acc);       //设置控制加速度；
    rtn=GT_Update();          //刷新数据，施加控制
}
void enable_servo()
{
    GT_AxisOn();      //使能伺服
    GT_PrflV();       //设置当前轴（1 轴）为速度控制模式
    GT_Update();      //刷新参数
}
void anti_cran()
{
    //根据状态反馈计算控制输出
    v =     - (pos_param[0].value * (pos+offset)
        + pos_param[2].value * (ang_2pi-M_PI)
```

```
            + pos_param[4].value * (ang_2pi2)
            + pos_param[1].value * posDot
            + pos_param[3].value * angleDot
            + pos_param[5].value * (angleDot2+angleDot)
            );

        //将控制量转化为运动控制器可以接受的量纲
        acc = fabs(v * ACC_COF );
        if (v > 0 )      vel =    10 * VEL_COF;
        if (v < 0 )      vel = -10 * VEL_COF;
    }
```

显然，如果您要加入自己的算法到该系统，可以直接修改该函数。需要说明的是，在对摆系统进行数学建模时已经明确，**所有摆系统是一个力控设备，根据牛顿第二定律，力与加速度成正比，因此给系统施加的控制量就是加速度，而速度则是一个事先设定的较大数值，在控制过程中，一般不会达到这个速度值。**如何选择这个速度值，需要一定的实验数据支持和经验。

```
    int handle_safety() //安全处理模块
    {
        unsigned short swt;
        rtn=GT_GetSts(&swt);          //查询控制卡状态

        //检查是否到达软限位位置或硬限位（限位开关）位置
        if ((fabs(pos) >= LIMIT) ||( (swt & 0x20) == 0x20 )||( (swt & 0x40) == 0x40 ))
        {//如果是，关闭控制器；并复位变量，不终止整个程序的运行
            safety = 1;       start = 0;  pend = 0; motion = 0;     offset = 0;       vel = 0;    acc = 0;
            GT_Close();
        }

        //检查当前运动速度是否超过最高安全速度
        if (fabs(posDot) > MAX_SPEED)
        {//如果是，关伺服和控制器，返回-1，最终终止整个程序的运行
            safety = ERR_2;     rtn=GT_AxisOff(); pend = 0;        start = 0;
            vel = 0;          acc = 0;          GT_Close();
            return -1;
        }

        //检查急停按钮（空格键）是否按下
        if (safety == ERR_1)
        {//如果按下，关伺服和控制器，返回-1，最终终止整个程序的运行
            rtn=GT_AxisOff();//error_msg(rtn);
            pend = 0;         start = 0;
            vel = 0;          acc = 0;
            GT_Close();
            return -1;
```

```
        }

        return 0;
    }
```

实际提供的演示控制源代码与上面的代码并不完全一致，请在阅读时注意。上面程序中所用到的许多变量，在此也没有定义，具体含义和类型请对照实际源代码。

本附录介绍的内容对于利用固高摆进行控制理论研究是必不可少的。因为本书的所有实验和研究项目都是借助于这些软件，所以读者必须熟练掌握本附录所介绍的内容。