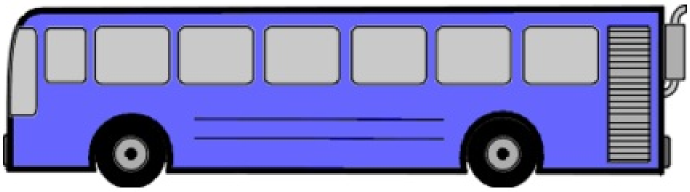
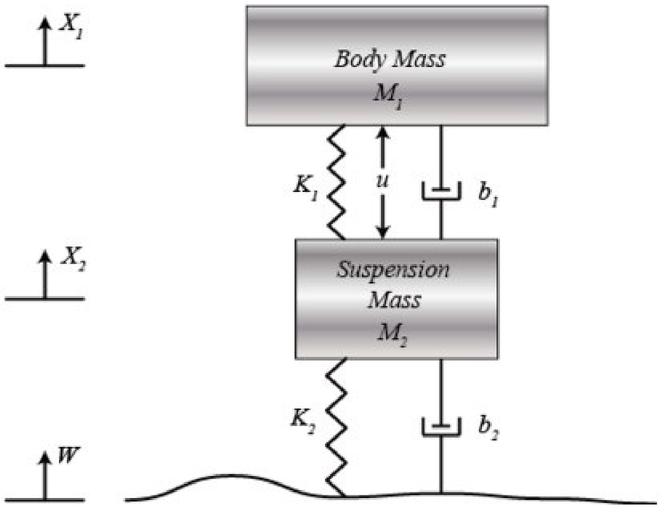
1. **系统建模**（System Modeling）

模型如下



简化起见，先建立客车的1/4个模型，即一个轮子的情况

Model of Bus Suspension System (1/4 Bus)



由牛顿第二定律有：

$$M_1\ddot{X}_1 = -b_1(\dot{X}_1 - \dot{X}_2) - K_1(X_1 - X_2) + U$$

$$M_2\ddot{X}_2 = b_1(\dot{X}_1 - \dot{X}_2) + K_1(X_1 - X_2) + b_2(\dot{W} - \dot{X}_2) + K_2(W - X_2) - U$$

参数如下

(M1)	1/4 bus body mass	2500 kg
(M2)	suspension mass	320 kg
(K1)	spring constant of suspension system	80,000 N/m
(K2)	spring constant of wheel and tire	500,000 N/m
(b1)	damping constant of suspension system	350 N.s/m
(b2)	damping constant of wheel and tire	15,020 N.s/m
(U)	control force	

传递函数

假设初始条件为0，目标输入为u、w输出为x1、x2，由动力学方程进行拉普拉斯变换得：

$$(M_1s^2+b_1s+K_1)X_1(s)-(b_1s+K_1)X_2(s)=U(s)$$

$$(b_1s+K_1)X_1(s)+(M_2s^2+(b_1+b_2)s+(K_1+K_2))X_2(s)=(b_2s+K_2)W(s)-U(s)$$

$$\begin{bmatrix}(M_1s^2+b_1s+K_1) & -(b_1s+K_1)\\ -(b_1s+K_1) & (M_2s^2+(b_1+b_2)s+(K_1+K_2))\end{bmatrix}\begin{bmatrix}X_1(s)\\ X_2(s)\end{bmatrix}=\begin{bmatrix}U(s)\\ (b_2s+K_2)W(s)-U(s)\end{bmatrix}$$

$$A=\begin{bmatrix}(M_1s^2+b_1s+K_1) & -(b_1s+K_1)\\ -(b_1s+K_1) & (M_2s^2+(b_1+b_2)s+(K_1+K_2))\end{bmatrix}$$

$$\Delta=\det\begin{bmatrix}(M_1s^2+b_1s+K_1) & -(b_1s+K_1)\\ -(b_1s+K_1) & (M_2s^2+(b_1+b_2)s+(K_1+K_2))\end{bmatrix}$$

or

$$\Delta=(M_1s^2+b_1s+K_1)\cdot(M_2s^2+(b_1+b_2)s+(K_1+K_2))-(b_1s+K_1)\cdot(b_1s+K_1)$$

求逆有

$$\begin{bmatrix}X_1(s)\\ X_2(s)\end{bmatrix}=\frac{1}{\Delta}\begin{bmatrix}(M_2s^2+(b_1+b_2)s+(K_1+K_2)) & (b_1s+K_1)\\ (b_1s+K_1) & (M_1s^2+b_1s+K_1)\end{bmatrix}\begin{bmatrix}U(s)\\ (b_2s+K_2)W(s)-U(s)\end{bmatrix}$$

$$\begin{bmatrix}X_1(s)\\ X_2(s)\end{bmatrix}=\frac{1}{\Delta}\begin{bmatrix}(M_2s^2+b_2s+K_2) & (b_1b_2s^2+(b_1K_2+b_2K_1)s+K_1K_2)\\ -M_1s^2 & (M_1b_2s^3+(M_1K_2+b_1b_2)s^2+(b_1K_2+b_2K_1)s+K_1K_2)\end{bmatrix}\begin{bmatrix}U(s)\\ W(s)\end{bmatrix}$$

得到传递函数，若W(s) = 0,则得

$$G_1(s)=\frac{X_1(s)-X_2(s)}{U(s)}=\frac{(M_1+M_2)s^2+b_2s+K_2}{\Delta}$$

若U(s) = 0，则得

$$G_2(s)=\frac{X_1(s)-X_2(s)}{W(s)}=\frac{-M_1b_2s^3-M_1K_2s^2}{\Delta}$$

Matlab

```
>> M1 = 2500;
M2 = 320;
K1 = 80000;
K2 = 500000;
b1 = 350;
b2 = 15020;

s = tf('s');
G1 = ((M1+M2)*s^2+b2*s+K2)/((M1*s^2+b1*s+K1)*(M2*s^2+(b1+b2)*s+(K1+K2))-(b1*s+K1)*(b1*s+K1))
G2 = (-M1*b2*s^3-M1*K2*s^2)/((M1*s^2+b1*s+K1)*(M2*s^2+(b1+b2)*s+(K1+K2))-(b1*s+K1)*(b1*s+K1))

G1 =

                 2820 s^2 + 15020 s + 500000
  ------------------------------------------------------------
  800000 s^4 + 3.854e07 s^3 + 1.481e09 s^2 + 1.377e09 s + 4e10

Continuous-time transfer function.


G2 =

                 -3.755e07 s^3 - 1.25e09 s^2
  ------------------------------------------------------------
  800000 s^4 + 3.854e07 s^3 + 1.481e09 s^2 + 1.377e09 s + 4e10

Continuous-time transfer function.
```

2. **系统分析**（System Analysis）

好的客车悬浮系统拥有足够的路况保持能力，不管客车行驶在怎样的路面上，仍能提供足够的舒适度。x1-w的距离很难测量，轮胎的变形（x2-w）很微小（negligible），因此使用x1-x2的距离作为输出。

w作为路面的干扰（道路的高低不平等），假设为阶跃输入。对于输出x1-x2，设计要求是

- 超调小于5%

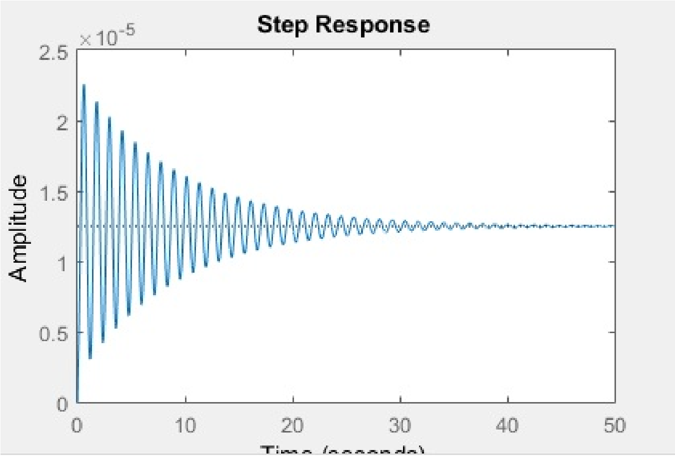
- 调节时间小于5s

即客车跑上了10cm高的路面，要求车体的振荡在+/-5mm的范围，并且在5s内恢复平衡。

2.1> 开环阶跃响应

轮胎变形

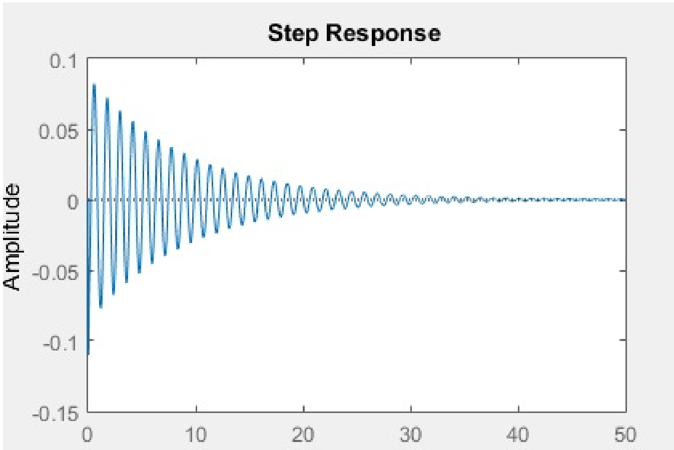
```
>> step(G1)
```



振幅很小，但调节时间太长。

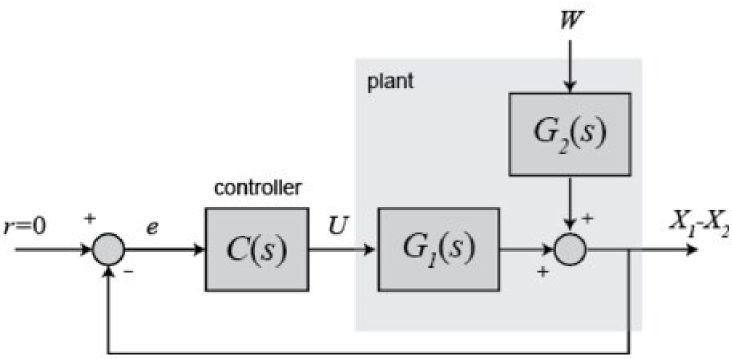
再考虑路面不平，幅值是0.1m

```
>> step(0.1*G2)
```



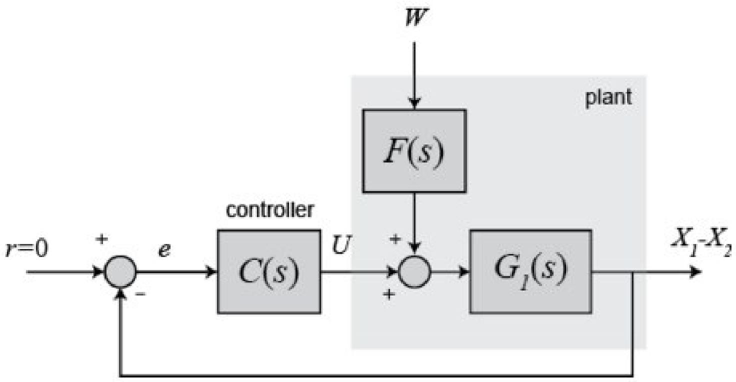
振动幅度太大，初始值甚至达到8cm，约50s的调节时间太长，远远超出要求。

解决的办法是加反馈控制器，如下框图所示



3. **PID控制器设计**（PID controller Design）

为了便编程计算，将上面框图作等效变换



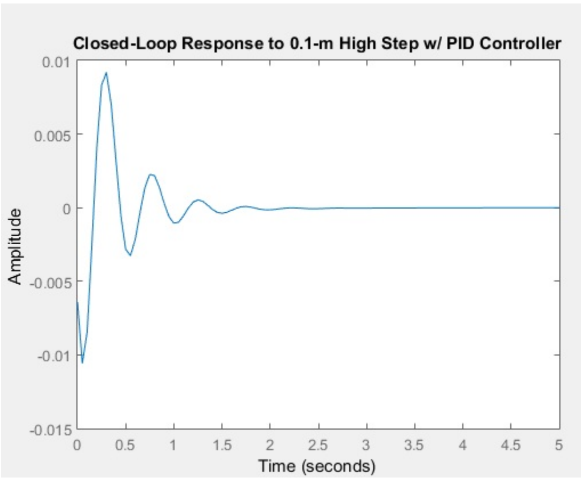
其中，F(s)G1(s) = G2(s)。

```
>> numf=[-(m1*b2) -(m1*k2) 0 0];
>> denf=[(m1+m2) b2 k2];
>> F=tf(numf,denf);
>> numf=[-(m1*b2) -(m1*k2) 0 0];
denf=[(m1+m2) b2 k2];
F=tf(numf,denf)

F =

  -3.755e07 s^3 - 1.25e09 s^2
  ---------------------------
  2820 s^2 + 15020 s + 500000

Continuous-time transfer function.
```

添加PID控制器

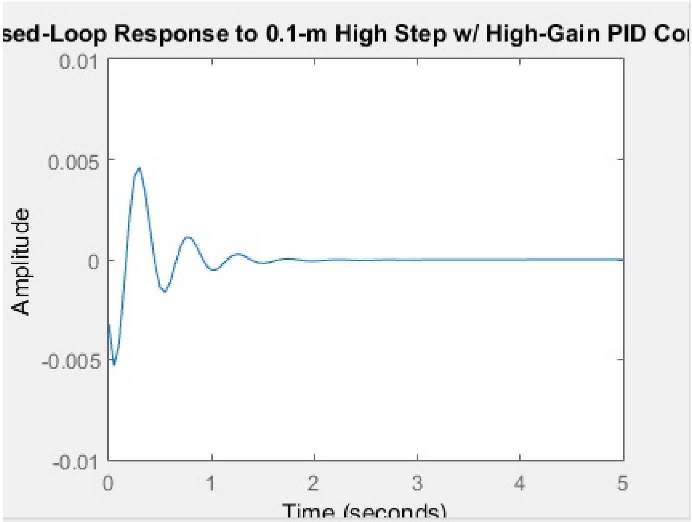
$$C(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

```
>>  Kd = 208025;
Kp = 832100;
Ki = 624075;
C = pid(Kp,Ki,Kd);
sys_cl=F*feedback(F*G1,C);
t=0:0.05:5;
step(0.1*sys_cl,t)
title('Closed-Loop Response to 0.1-m High Step w/ PID Controller')
```



超调为9mm大于5mm，但是调节时间已经满足要求。继续选择Kp、Ki、Kd值，这里取原值的两倍

```
>> Kd=2*Kd;
Kp=2*Kp;
Ki=2*Ki;
C=pid(Kp,Ki,Kd);
sys_cl=F*feedback(F*G1,C);
step(0.1*sys_cl,t)
title('Closed-Loop Response to 0.1-m High Step w/ High-Gain PID Controller')
>> axis([0 5 -.01 .01])
```



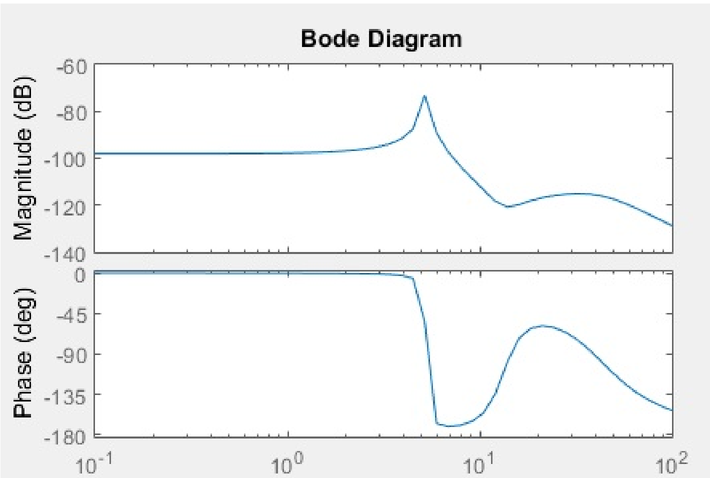
如图，此时满足设计要求。

4. **控制器设计中的频率法**（Frequency Methods for Controller Design）

4.1> 开环响应的Bode图

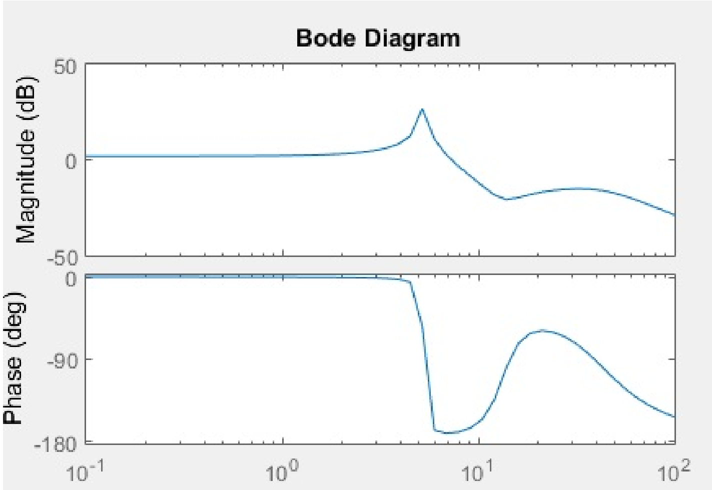
频率法的主要思想是根据开环传递函数的bode如来估计闭环时的响应，通过添加控制器来改变开环传递函数的Bode图进而闭环响应也会改变从而实现满足要求的闭环响应。

```
>> w = logspace(-1,2);
bode(G1,w)
```



为了在不同自然频率下呈现系统，令低频阶段的幅值为0，通过调整增益K来增加响应的幅度，同时K的增加不会影响相位曲线。幅值曲线需上移100dB = 20*logK，即K = 100000。

```
>> K=100000;
bode(K*G1,w)
```



如上，与分析一致。

4.2> 添加前向控制（lead control）

在5rad/sec相位曲线是凹的，首先在这个区域增加正相位，**大的相位裕度会使闭环响应超调减小**，目标增加140deg的正相位，因为一个前向控制器（lead controller）所能增加的相位不超过90deg，因此使用两个前向控制器。通过下列步骤来确定前向控制的参数T和a

1.确定所需的正相位：每个控制器 140/2 = 70deg

2.确定增加相位处的频率：5.0rad/sec

3.根据下式计算a值

$$a = \frac{1-\sin 70^o}{1+\sin 70^o} = 0.031091$$

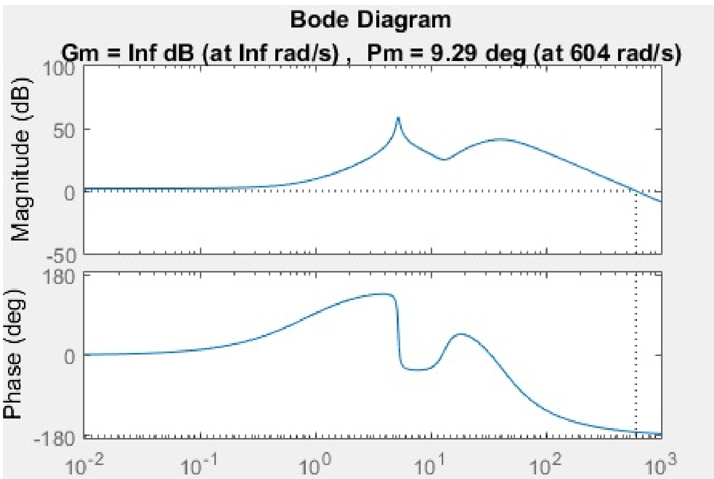
4.根据下式计算T和aT

$$T = \frac{1}{W\sqrt{a}} = \frac{1}{5\sqrt{0.031091}} = 1.13426$$

$$aT = \frac{\sqrt{a}}{W} = \frac{\sqrt{0.031091}}{5} = 0.035265$$

观察添加前向控制的Bode图

```
>> a = (1-sin(70/180*pi))/(1+sin(70/180*pi));
w=5;
T=1/(w*sqrt(a));
aT=sqrt(a)/w;
numc = conv([T 1], [T 1]);
denc = conv([aT 1], [aT 1]);
C = tf(numc,denc);
margin(K*C*G1)
```



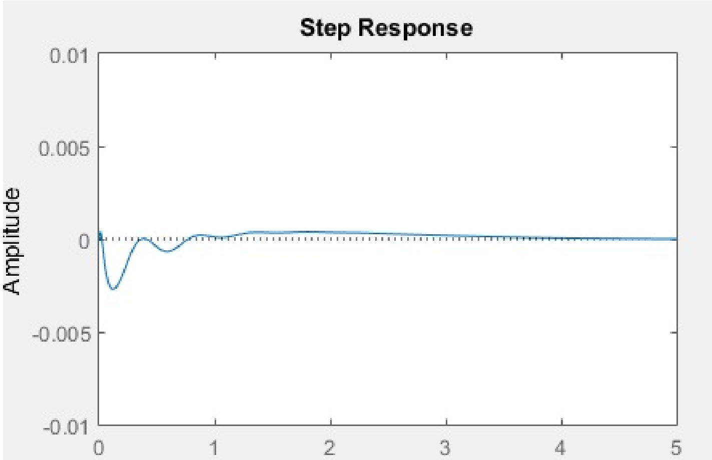
由图，相位裕度已经足够达到设计要求了（相位裕度大吗？？？）。

闭环传递函数为

```
sys_cl = F*feedback(G1,K*C);
```

观察此时系统的闭环响应

```
>> t=0:0.01:5;
step(0.1*sys_cl,t)
axis([0 5 -.01 .01])
```



如图说示，已经满足要求。

4. **形位空间控制器设计**（State-Space Controller Design）

令Y = X1 - X2，系统的状态空间方程如下：

$$
\begin{bmatrix} \dot{X}_1 \\ \ddot{X}_1 \\ \dot{Y}_1 \\ \ddot{Y}_1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{-b_1 b_2}{M_1 M_2} & 0 & \left[\frac{b_1}{M_1}\left(\frac{b_1}{M_1}+\frac{b_1}{M_2}+\frac{b_2}{M_2}\right)-\frac{K_1}{M_1}\right] & \frac{-b_1}{M_1} \\ \frac{b_2}{M_2} & 0 & -\left(\frac{b_1}{M_1}+\frac{b_1}{M_2}+\frac{b_2}{M_2}\right) & 1 \\ \frac{K_2}{M_2} & 0 & -\left(\frac{K_1}{M_1}+\frac{K_1}{M_2}+\frac{K_2}{M_2}\right) & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ \dot{X}_1 \\ Y_1 \\ \dot{Y}_1 \end{bmatrix} +
$$

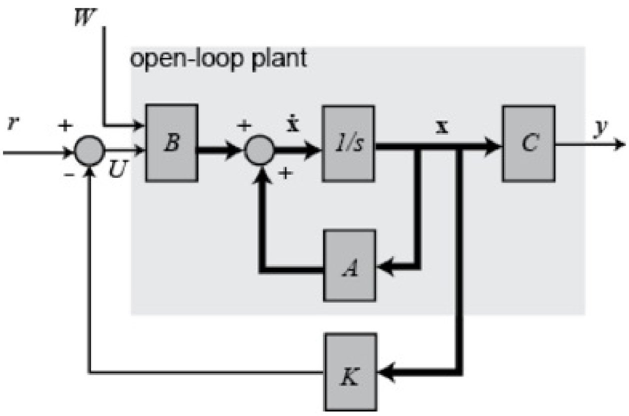
$$
\begin{bmatrix} 0 & 0 \\ \frac{1}{M_1} & \frac{b_1 b_2}{M_1 M_2} \\ 0 & \frac{-b_2}{M_2} \\ \left(\frac{1}{M_2}+\frac{1}{M_2}\right) & \frac{-K_2}{M_2} \end{bmatrix} \begin{bmatrix} U \\ W \end{bmatrix}
$$

$$
Y = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_1 \\ \dot{X}_1 \\ Y_1 \\ \dot{Y}_1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} U \\ W \end{bmatrix}
$$

Matlab:

```
>> m1 = 2500;
m2 = 320;
k1 = 80000;
k2 = 500000;
b1 = 350;
b2 = 15020;

A=[0                 1    0                                                 0
  -(b1*b2)/(m1*m2)   0   ((b1/m1)*((b1/m1)+(b1/m2)+(b2/m2)))-(k1/m1)   -(b1/m1)
   b2/m2             0   -((b1/m1)+(b1/m2)+(b2/m2))                       1
   k2/m2             0   -((k1/m1)+(k1/m2)+(k2/m2))                       0];
B=[0                 0
   1/m1              (b1*b2)/(m1*m2)
   0                -(b2/m2)
   (1/m1)+(1/m2)    -(k2/m2)];
C=[0 0 1 0];
D=[0 0];
sys=ss(A,B,C,D);
```

设计full-state controller（假设系统的状态变量都是可测的），框图如下：



闭环系统的特征多项式是（SI - (A-B[1 0]'K))而非（SI - (A-BK))因为控制器只能控制力输入U而不能控制路面阻抗，B矩阵为4*2型，只需令其第二列元素全部为0。为了消除稳态误差，必须有积分动作（integral action），因此增加一个额外的系统变量，设为 int(Y_1) （Y_1的积分）。新的状态空间方程为

$$\dot{\mathbf{x}} = \left(A - B\begin{bmatrix}1\\0\end{bmatrix}K\right)\mathbf{x} + B\begin{bmatrix}U\\W\end{bmatrix}$$

$$y = C\mathbf{x}$$

重新确定A、B、C、D

```
>> Aa = [[A,[0 0 0 0]'];[C, 0]];
Ba = [B;[0 0]];
Ca = [C,0];
Da = D;
sys=ss(Aa,Ba,Ca,Da);
```

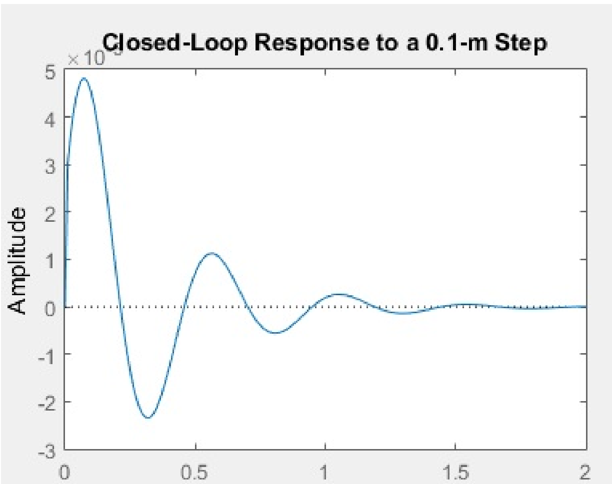
添加K为

```
>> K = [0 2.3e6 5e8 0 8e6]
```

观察闭环响应

```
>> t = 0:0.01:2;
sys_cl = ss(Aa-Ba(:,1)*K,-0.1*Ba,Ca,Da);
step(sys_cl*[0;1],t)
title('Closed-Loop Response to a 0.1-m Step')
```

给Ba乘0.1表示地面高度增加0.1m时的情况



观察响应满足要求，也可以尝试下别的K值。

5. **Simulink建模**（Simulink Modeling）

动态表达式

$$M_1\ddot{X}_1 = -b_1(\dot{X}_1 - \dot{X}_2) - K_1(X_1 - X_2) + U$$

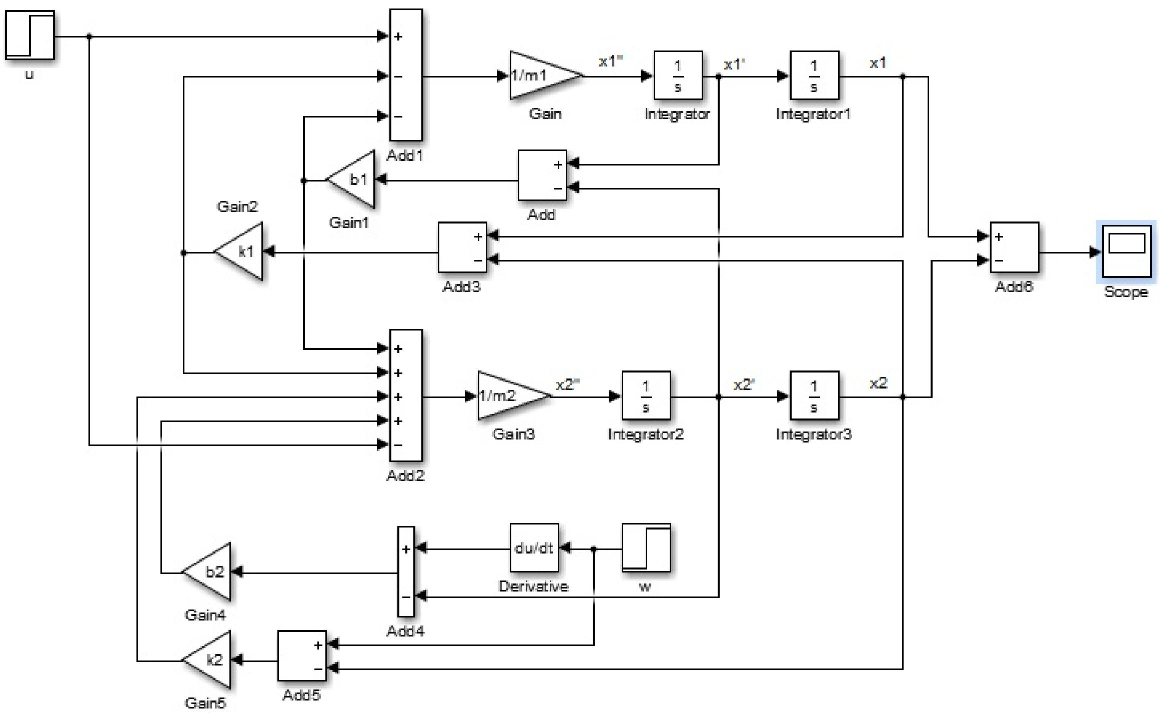
$$M_2\ddot{X}_2 = b_1(\dot{X}_1 - \dot{X}_2) + K_1(X_1 - X_2) + b_2(\dot{W} - \dot{X}_2) + K_2(W - X_2) - U$$

仍从积分环节开始建模开始建模

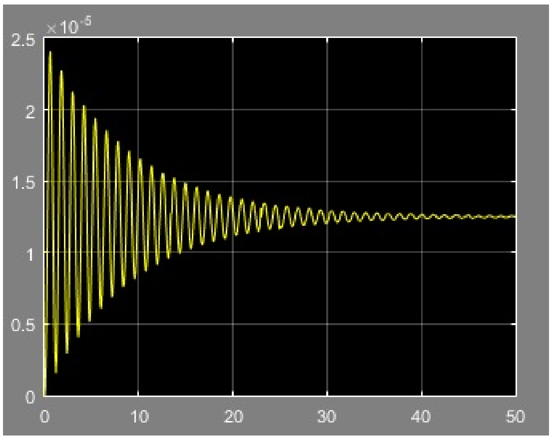
$$\int\int \frac{d^2x_1}{dt^2}\,dt = \int \frac{dx_1}{dt}\,dt = x_1$$

$$\int\int \frac{d^2x_2}{dt^2}\,dt = \int \frac{dx_2}{dt}\,dt = x_2$$

Simulink模型如下



其中，u的step time改为0，暂时先不考虑地面情况的干扰，设置w的step time改为0，final time为0。仿真时间设为50s，点击运行



下一节将通过控制U来提高系统的响应特征。

6. **Simulink控制器设计**（Simulink Controller Design）

6.1> 抽取线性模型到Matlab

这次使用 linmod 函数来实现抽取Simulink中的模型到Matlab中。将Simulink的输入u与输出x1-x2分别换位int block和output block，保存模型为suspmod.mdl。

先在Matlab中输入对应参数值

```
>> m1 = 2500;
m2 = 320;
k1 = 80000;
k2 = 500000;
b1 = 350;
b2 = 15020;
```

继续输入

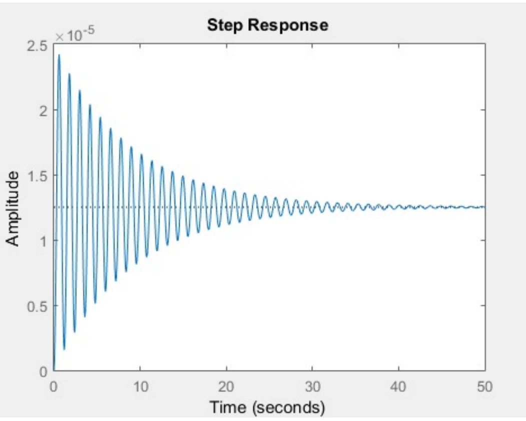
```
>> [A,B,C,D]=linmod('suspmod')
[num,den]=ss2tf(A,B,C,D)
```

就可以观察到Simulink模型转化成状态空间函数的参数值A,B,C,D和传递函数的参数值num,den。

```
A =

   1.0e+03 *

         0         0    0.0010         0
         0         0         0    0.0010
   -0.0320    0.0320   -0.0001    0.0001
    0.2500   -1.8125    0.0011   -0.0480


B =

   1.0e-03 *

         0
         0
    0.4000
         0


C =

     1    -1     0     0


D =

     0
```

```
num =

         0         0    0.0004    0.0188    0.6250


den =

   1.0e+04 *

    0.0001    0.0048    0.1851    0.1721    5.0000
```

观察阶跃响应来验证模型



如图，与Simulink中所得一样，抽取无误。

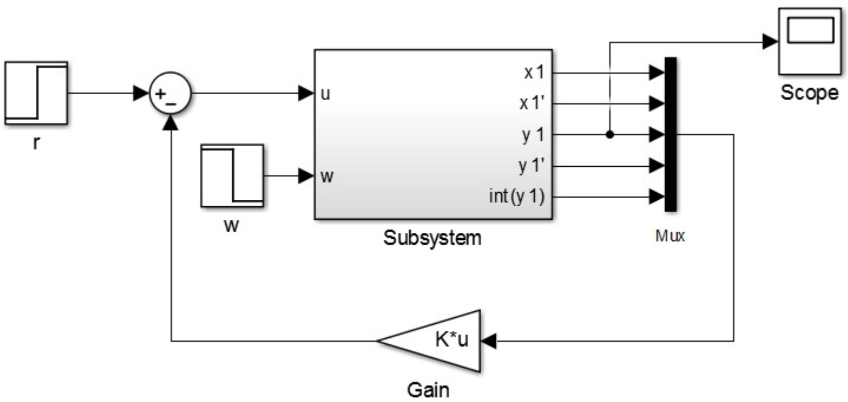
6.2> 实现全状态反馈控制（Implementing full state-feedback control）
在形位空间控制器一节确定了新的状态量

$$\left[X_1 \quad \frac{dX_1}{dt} \quad Y_1 = X_1 - X_2 \quad \frac{dY_1}{dt} \quad \int Y_1 \, dt\right]^T$$

控制器使用的反馈增益为 K = [0 2.3E6 5E8 0 8E6]。

如图，将原Simulink模型对应的输入量及状态量分别以input block及output block（双击输入输出模块，可以改变端口顺序，在封装子系统中会以该顺序呈现输入输出）引出

封装为子系统



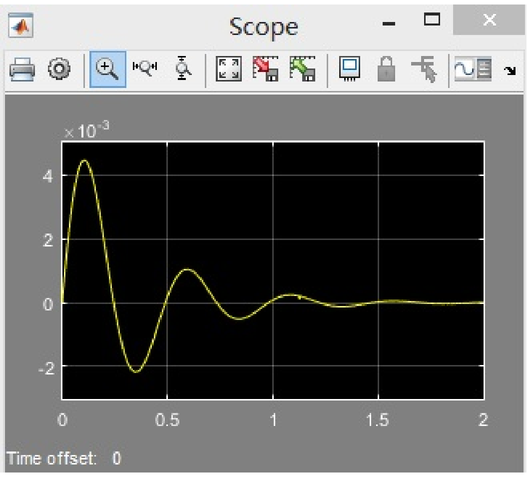
suspmod_control.slx

其中，设置r的step time改为0， final time为0，设置w的step time改为0，final time为-0.1。添加Mux作为多输出标量转化为一个向量。新版的Gain的输入为向量/矩阵时应选择Matrix gain，新版的Matlab将其与gain合在一起，双击gain，Multiplication改为：Matrix(K*u) (u vector) 或 Matrix(K*u)。

在Matlab中输入K值

```
>> K = [ 0 2.3e6 5e8 0 8e6 ];
```

设置仿真时间为2s，点击运行，观察y1图线



如上，满足要求。

（此处有个问题，将w的final time改为0.1，运行程序所得y1图线应该与形位空间控制器一节所得的一样，但观察发现两图线关于x轴对称？？）