CONTROL OF INVERTED PENDULUM REPORT 2016

Analog, Fuzzy and PID Control of Single Inverted Pendulum

Tang Qi

M-B5-5515-2

Department:
Electrical and Computer Engineering
Course Code:
ELCE707
Course Title:
Advanced Control

Requirements of experiment:

1. Familiar with the equipment
2. Analog control
3. Digital control (optional)
4. Fuzzy Control

1. Familiar with the equipment

1.1 System: Microsoft Windows 98 Second Version 4.10.2222 A

1.2 Computer: GenuineIntel Pentium(r) Processor 384.0MB RAM

The RAM of the processor is just fine for this task, but we should slow down the operations in case of crash.

1.3 Hardware: Bytronic© PCS1 Pendulum Control System

The PCS1consists of two separate modules linked by a connecting cable: a Carriage Module (Fig.01) and a Control Console (Fig.02). The carriage module consists of a carriage that carries a pivoted rod and mass that is driven along a 500mm track by a dc servo motor and toothed belt. The motor has an integral tachometer. The carriage position and the attitude of rod/mass assembly are measured by potentiometers

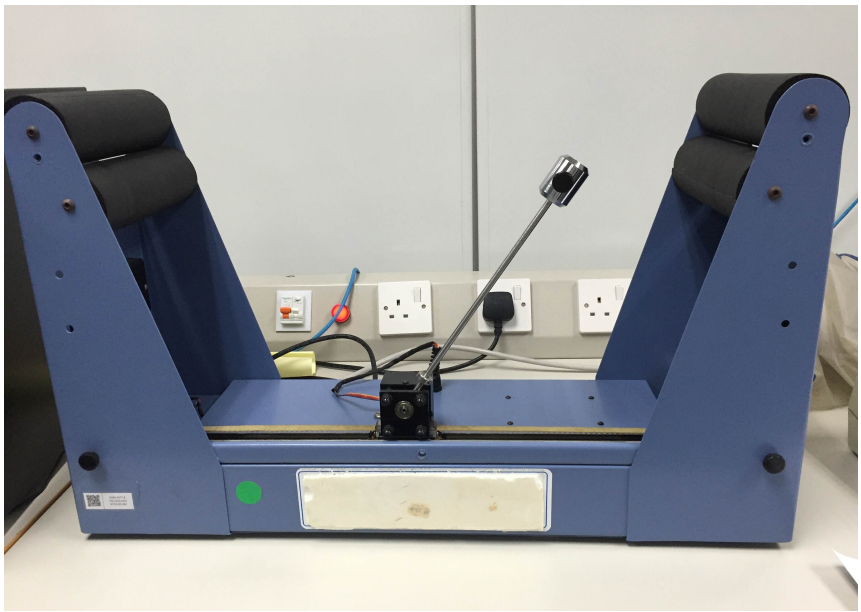

Fig.01 Carriage Module

The Control Console includes connection to the control/measurement signals (in Fig.02) and a clear mimic diagram of the overall system (Fig.03). The measurement signals are via easily accessible 4mm color-coded sockets. The color-coding is as

follows:

- Red terminals are analogue inputs and outputs.
- Blue terminals are monitoring points for feedback signals.
- Yellow terminals are the compensator connection points.
- Green connectors are ground.

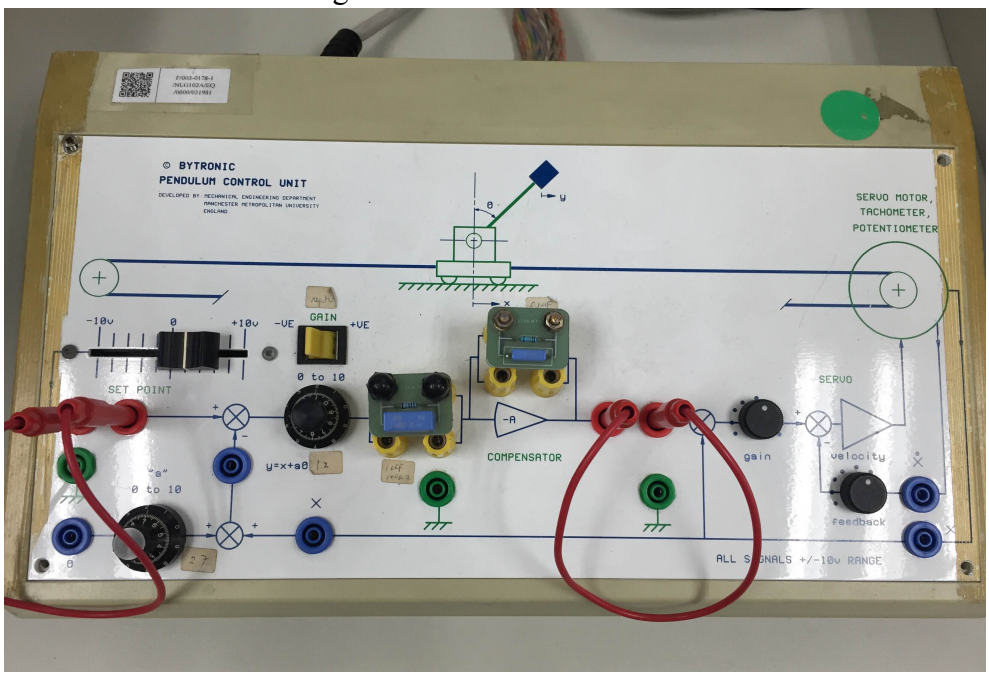

Fig.02 Control Console

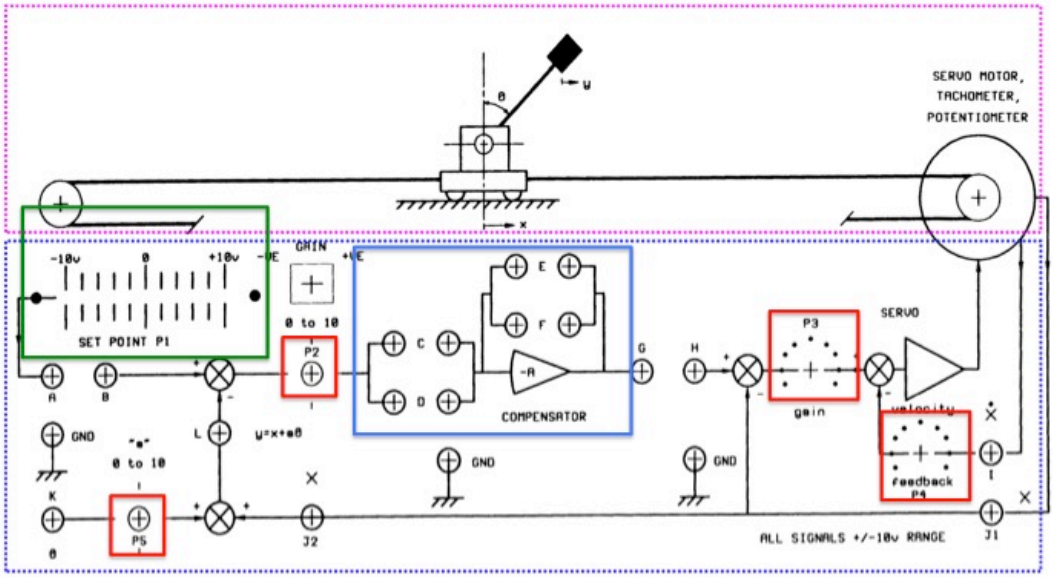

Fig.03 Mimic diagram of the overall system

The overall system contains 4 gains for: input angle theta θ , displacement y, output

and velocity feedback (from tachometer). The green part is set points of initial value for calibration.

Initially, we check the stability of the open loop IP. We will neglect the dynamics of the carriage servo (the response time is very fast compared with the IP). The transfer function of the IP is:

$$Gp(s) = \frac{1}{1 - 0.0264s^2}$$

We can tell the stability from the root locus plot in s-plane. Using Matlab, the root locus is as shown in Fig.04:

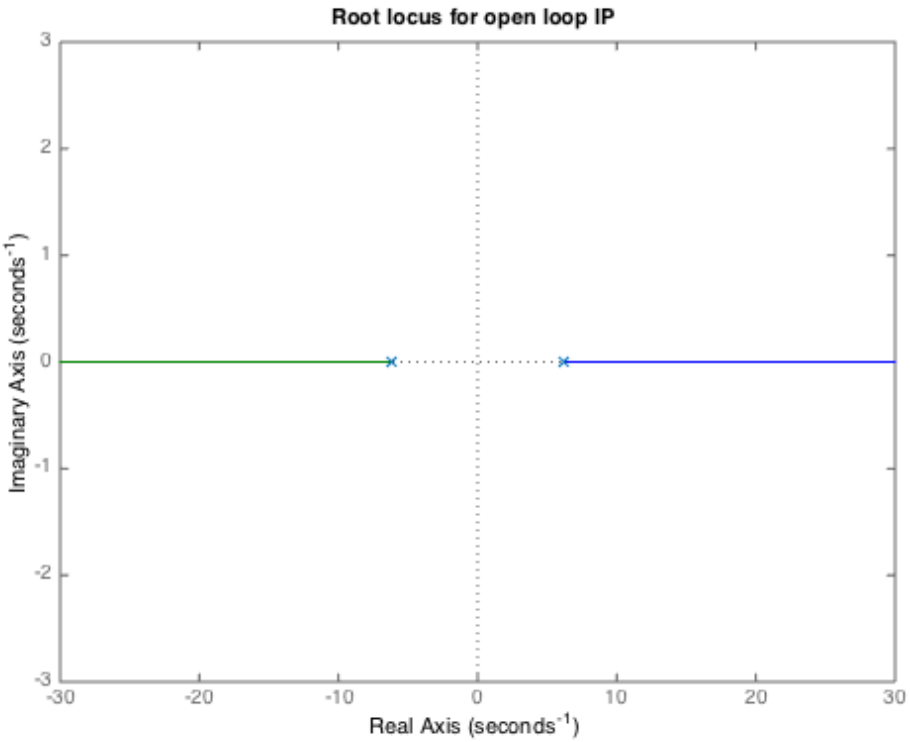

Fig.04 Mimic diagram of the overall system

The root locus of this open loop IP is not all on the left part of s-plane. Thus, we predict it unstable. The result is same if we applied positive or negative feedback gains as proportional control. Corresponding codes are as shown:

```
sys_IP=tf(1,[-0.0264 0 1]);
rlocus(sys_IP)
title('Root locus for IP ');
```

Then we plot the root locus diagram for both positive and negative gains (proportional gains of -0.5, -1.0, and -2.0)

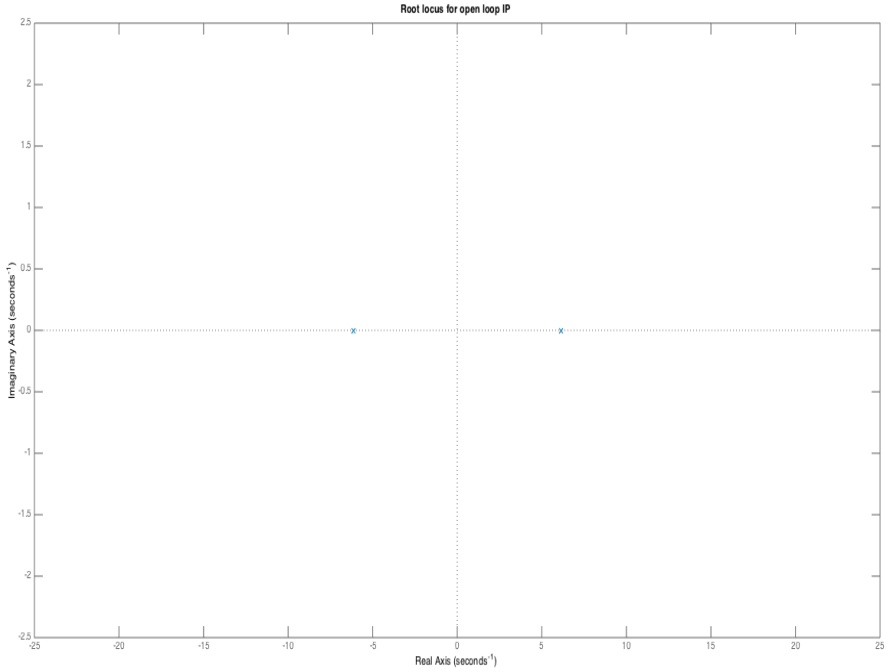

Fig.05 Root locus with positive proportional gain K=10

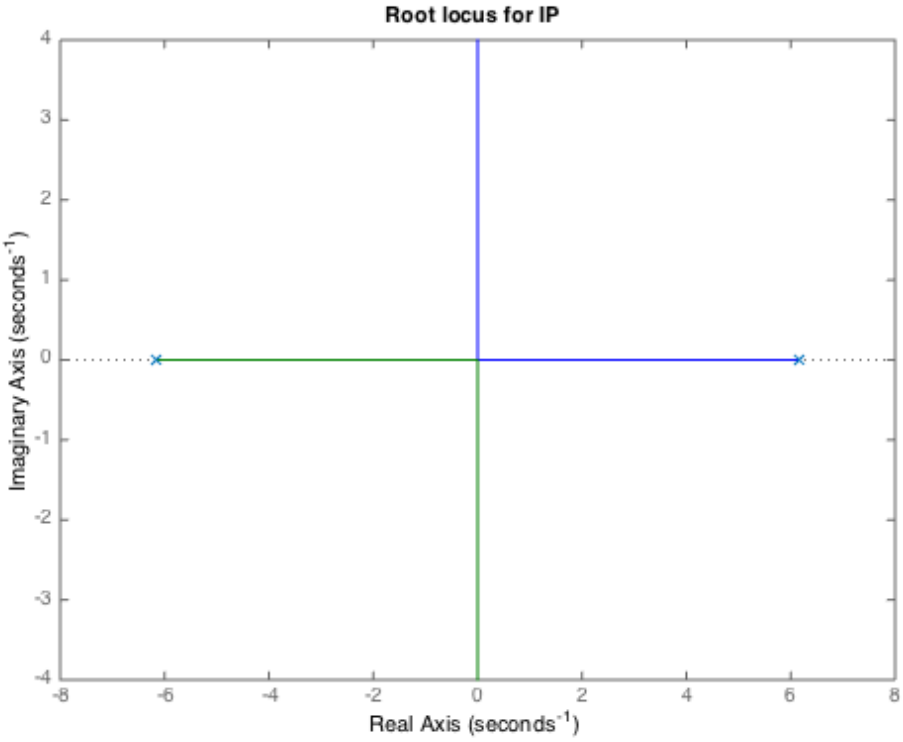

Fig.06 Root locus with negative proportional gain K= -0.5

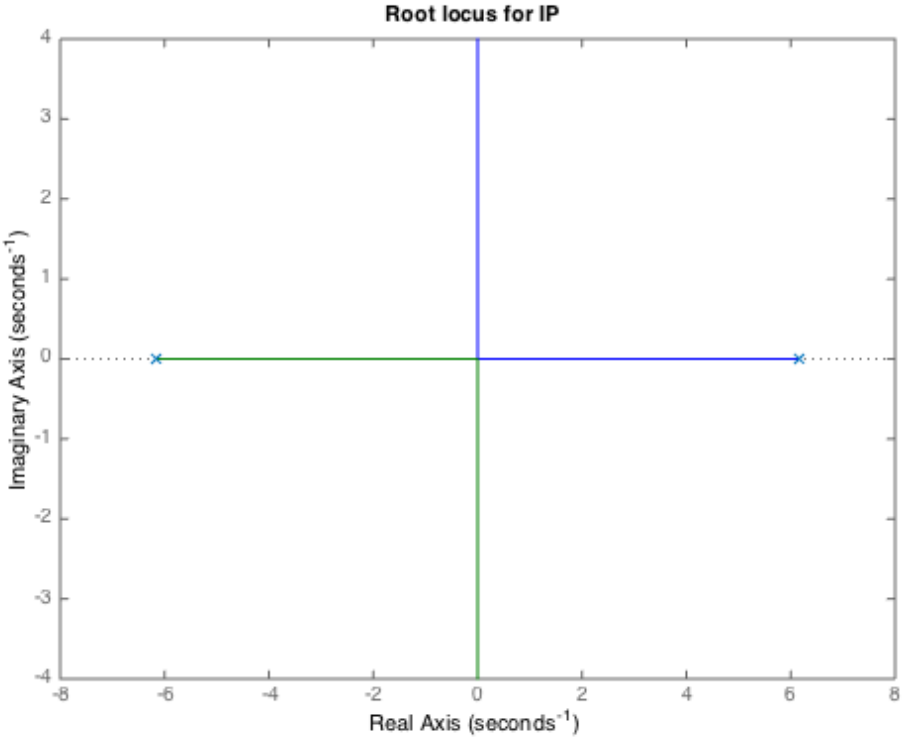

Fig.06 Root locus with negative proportional gain K= -1

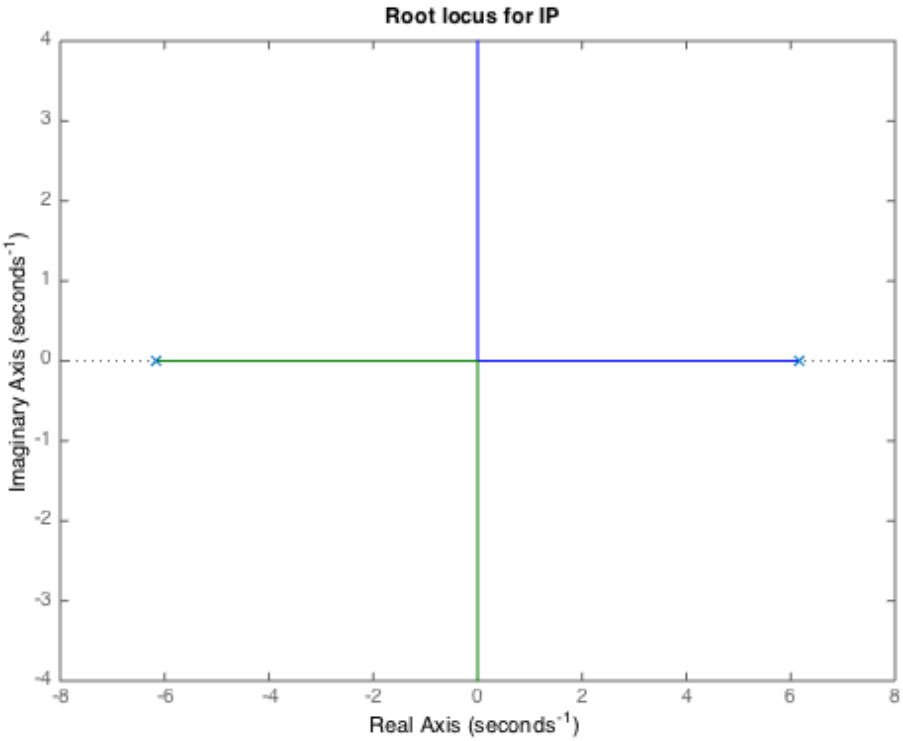

Fig.06 Root locus with negative proportional gain K= -2

No matter how we change K, the IP is not stable, thus we need to add a phase lead compensator to make the system stable. The blue part in Fig.03 is a phase lead compensator as shown in Fig.07, used to make stabilization possible according to corresponding system transfer function.

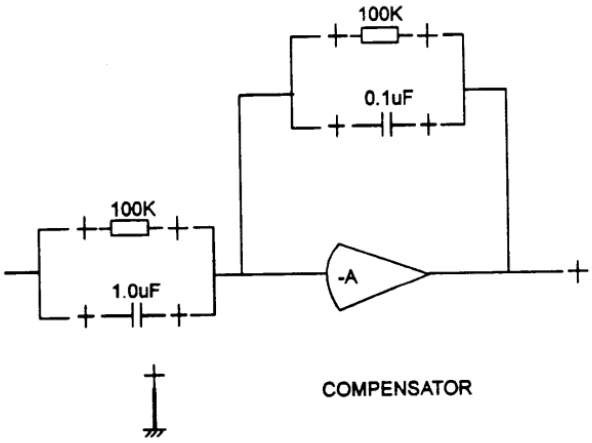

Fig.07 Phase lead compensator

Doing KVL, we can get the transfer function of the compensator:

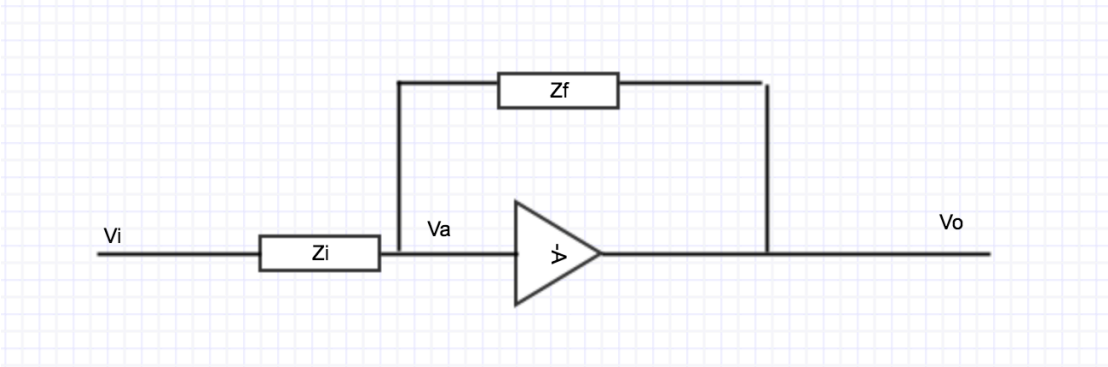

Fig.08 KVL diagram of phase lead compensator

$$\frac{Vi}{Zi} = \frac{Va\text{-}Vo}{Zf}$$

$$Vo = -A*Va$$

$$\frac{Vo}{Vi} = -\frac{Zf}{Zi} = -\frac{1+0.1s}{1+0.01s}$$

$$Gc(s) = \frac{1+0.1s}{1+0.01s}$$

Now we add phase lead compensator to the IP, we plot the new root locus to predict the stability, as shown in

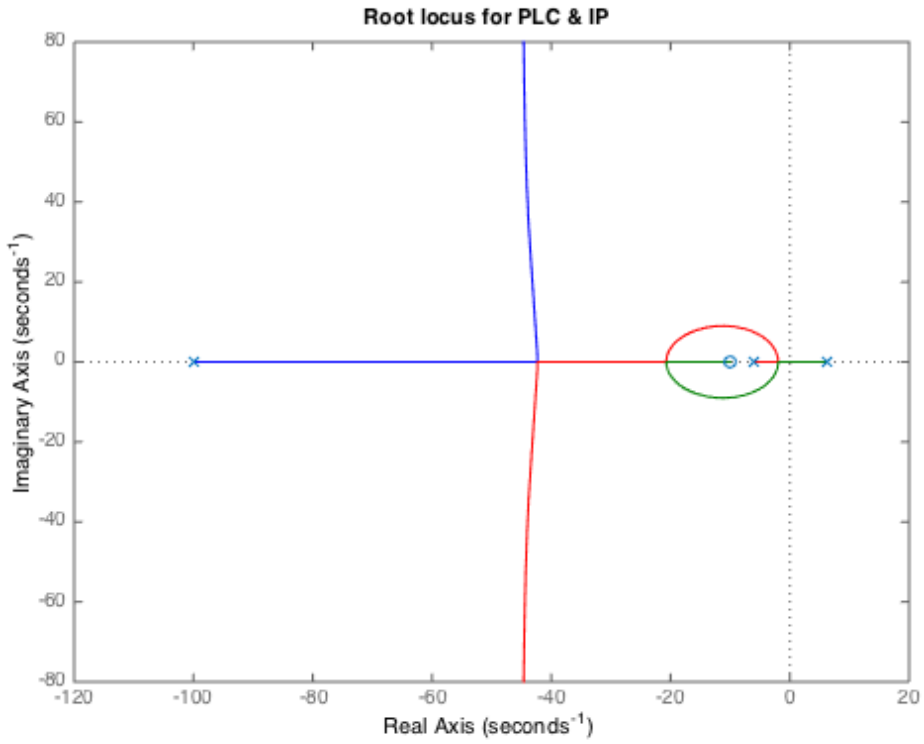

Fig.09 Root locus for open loop IP and compemsator

The system now can be stable under required overshoot and rise time. Corresponding codes are as following:

```
sys_IP=tf(1,[-0.0264 0 1]);
sys_PLC=tf([0.1 1],[0.01 1]);
sys_all=sys_IP*sys_PLC;
rlocus(-2*sys_all)
title('Root locus for PLC & IP ');
```

According to the dynamic indicator of system, we have formulas:

$$t_r \approx \frac{1.8}{\omega_n}$$

$$M_p = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}}$$

If we choose the overshoot to be no more than 5 % ($\zeta < 0.7$) and rise time to be less than 1s ($\omega_n > 1.8$). Adding unit feedback, we tried proportional gains of -1.5 and 02 . Step results are as shown in Fig.10 and Fig.11.

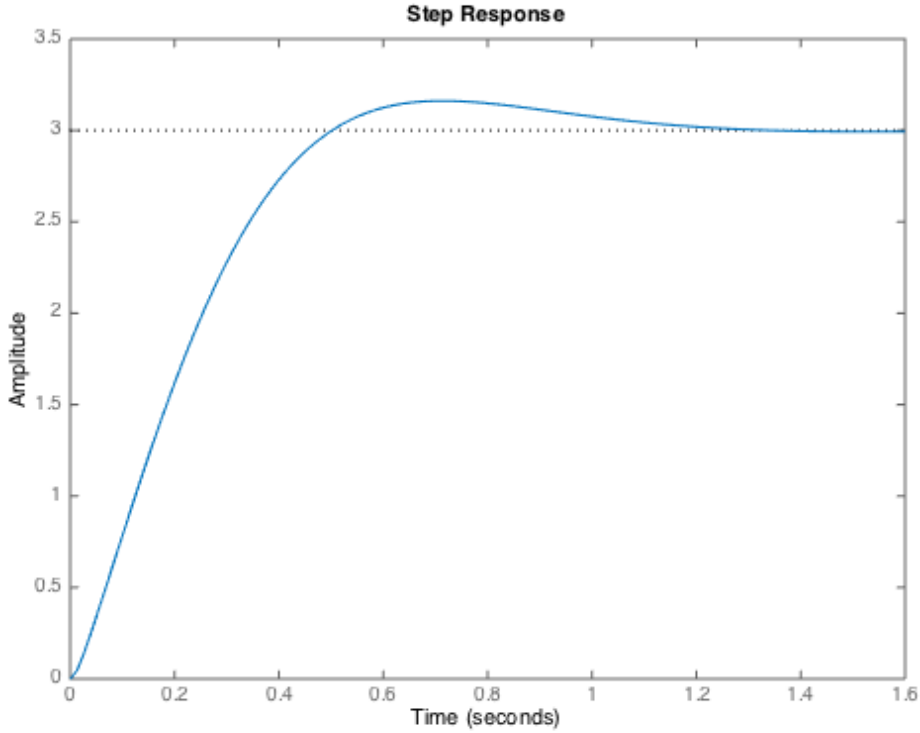

Fig.10 Step response of close loop system with negative proportional gain K= -1.5

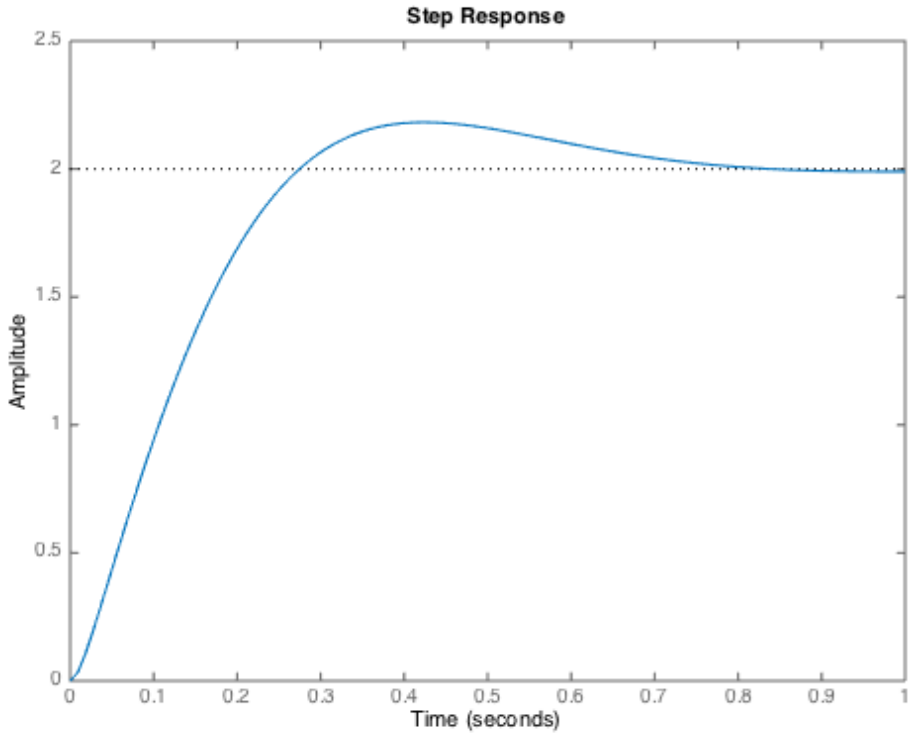

Fig.11 Step response of close loop system with negative proportional gain K= -2

1.4 Software: Bytronic Interface Devices IIC & Matlab 6.1

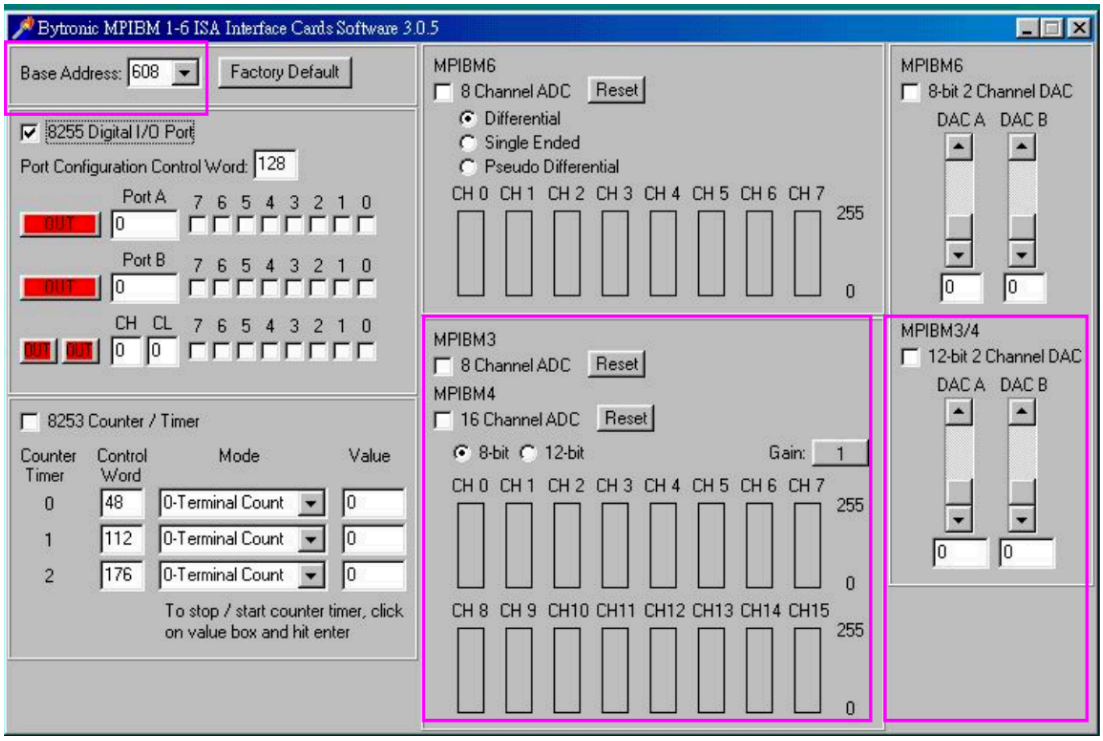

Fig.12 Bytronic Interface Devices IIC

Bytronic Interface Devices IIC: Corresponding Address 608. For the channel, we use 8 channels ADC in 12 bit (2^12=4096) as so the output DAC is also 12 bit of 2 channels. Different channel reflects different characteristics and the first channel represents the position (0~4096).

To digitally control the position of the pendulum horizontally, we can input digital values in the DAC A. Ideally, to the utmost left the value is 0 and to the utmost right position the value is 4095. However, in the analog process, the situation is slightly different.

The Matlab version in the lab computer is Matlab 6.0, launched in 2000 and is quite mature for this IP control task. However, some orders are slightly different. For example, the AND command, not && but &.

2. Analog control

The analog control is based on the adjustment of the four gains and the default set point. Observing the Control Console, we can find that the calibration of two input gains (for angle θ and for displacement y) are more detailed than the other two gains (for output and feedback) for the reason that gains of the inputs are more essential. From the analysis of simulation, we find the increase of output gain only causes slightly improvement of response time (when g0= 4, g1=0.8. So do the gain on feedback. All of the four gains will have an influence on the IP system.

Reading the instrument of the Carriage Module, we were suggested to follow the procedures:

1. Position the equipment on a desk with the carriage module in the inverted pendulum position.
2. Position the set point potentiometer to the middle (0V).
3. Turn on the power.
4. To balance the pendulum the variables "a", "gain" and the compensator all need to be designed and calculated. We could quickly get the unit operational, follow these approximate control panel settings below:
    a) Mass positioned at the top of the rod.
    b) Set "a" to 2.7.
    c) Set "gain" to 1.2.
    d) Switch the gain to negative.
    e) To allow quick set-up, compensator components have been supplied with the PCS.

In the analog control, we need the Control Console but not digital fuzzy control output from Matlab dacout, thus we need to de-plug the color cable for data transformation. The connection of the data transfer cable will cause a voltage input to the Control Console, in this case, there would be two input sources, one is analog input (angle θ and displacement y) and the other is noise, coming from computer voltages via data cable.

Practically, for the first time I did the experiment, I find two settings different from previously expected, the set point and the displacement gain. The set point potentiometer should no loner be set to the middle (0 V), considering that in analog control, ADC is used. The analog data are transferred to digital data and operate in voltages. For the displacement y, the corresponding voltage for digital analysis ranges from -10V to 10V. Thus, same as the normal potentiometer, we need to default the Control Console, over 20 years after production.

In practice, I find that the adjustment of gains fails to have an orderly effect on IP, and asked for help from lab administrator. It turned out that it was due to poor contact of

red terminals and dis-connect of data link inside Control Console.

After maintenance, the IP state is most stable when five parameters set as:

Set point: 2V Gain for angle θ : 2.9 Gain for IP displacement y: 1.2 Output gain: as showed in Fig.13 Feedback gain: as showed in Fig.13

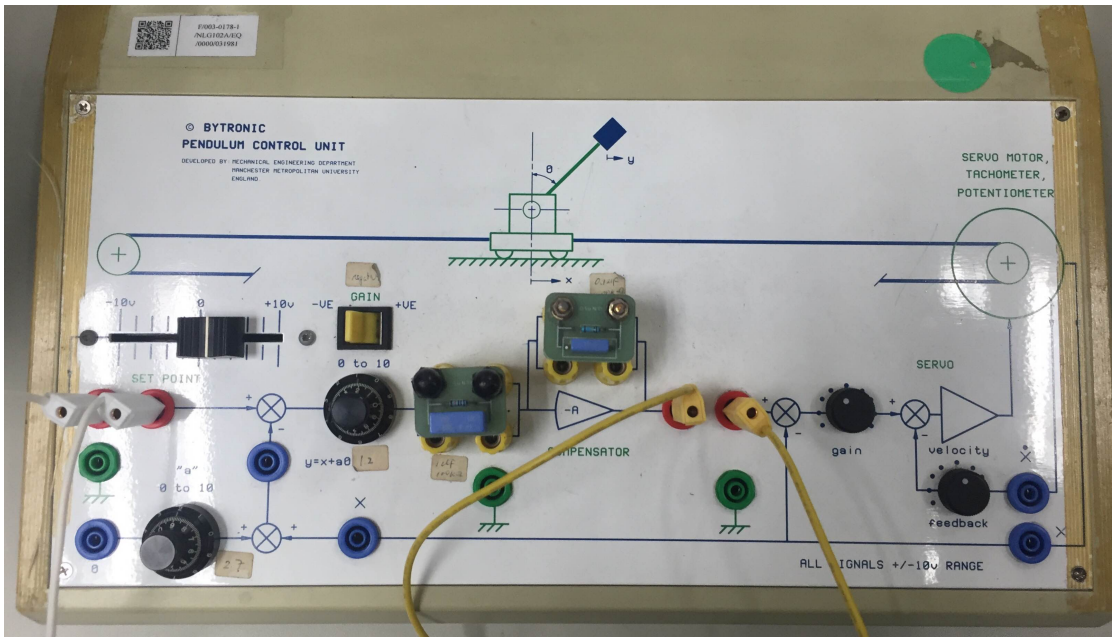

Fig.13 Setting of Control Console for analog control

3. Fuzzy Control

3.1 Fuzzy Controller design

As indicated in Simulation part, the linear transformation with 5*5 rule base already reached a satisfying result. Hence, in the experiment, I still use the triangular membership function and 5*5 rule base. The rule base table and membership function are as shown:

TABLE.01 Rule Table for the Inverted Pendulum with Rules That Are " ON" Highlighted:

"force" u		"change-in-error" $\dot{e}$				
		-2	-1	0	1	2
"error" e	-2	2	2	2	1	0
	-1	2	2	1	0	-1
	0	2	1	[0]	[-1]	-2
	1	1	0	-1	-2	-2
	2	0	-1	-2	-2	-2

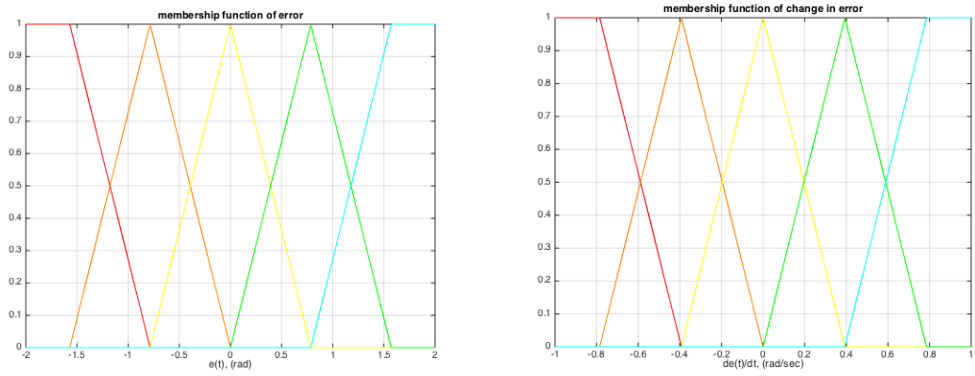

Fig.14 Input Membership Functions with Input Values (e & e')

Next is the inference step, which is determining conclusions according to the weight of each condition. Because we are using triangular membership function and each condition is of same weight, we are going to using the minimum of to represent the premise to certain situation and there should be always 2*2 conclusions reached (including repetitive conclusions).

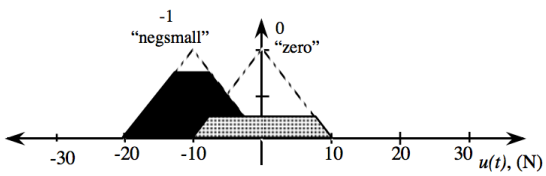

FIGURE 2.15 Implied fuzzy sets.

Fig.15 Interference to get decision

The fuzzy controller uses the minimum operator to represent both the "and" in the

premise and the implication and COG de-fuzzification).

From the simulation result, we to tune the "derivative gain to be g0=5, g1=0.3 and h=2 and achieved satisfying result, as shown in Fig.16 Corresponding codes are in Appendix I.

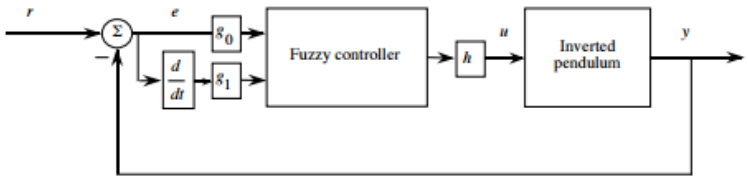

Fig.16 Improved fuzzy control system design for IP

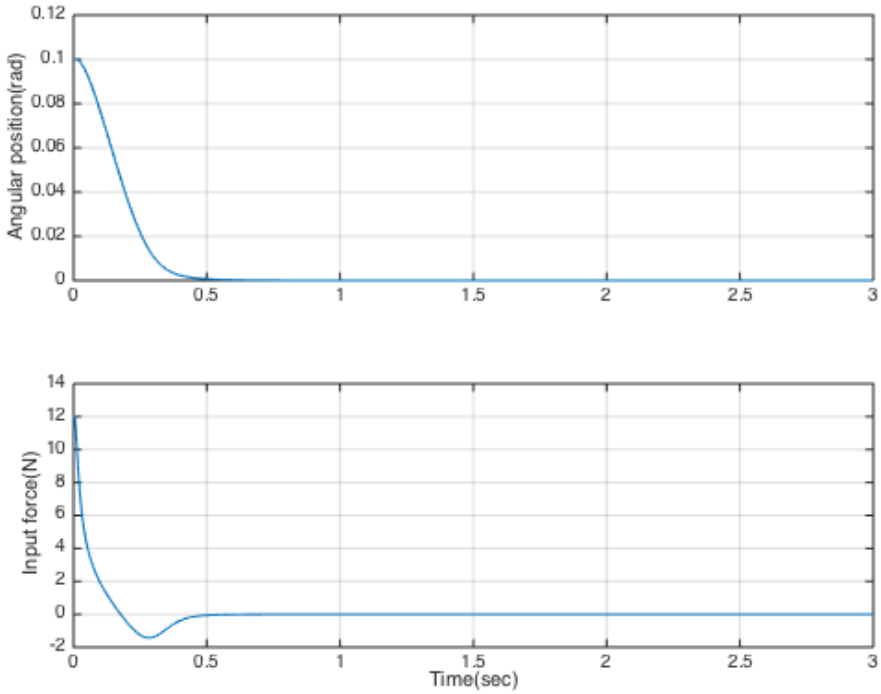

Fig.17 g0=5, g1=0.3, h=2 in simulation

With such gain and result in Fig.17, the system will be stable within 0.5s.

3.2 Fuzzy Control Experiment

Before the DDC fuzzy control, first we need to de-plug the cable to the control console, to noise of avoid analog input. I adjusted my previous code of simulation to do the experiment referring to *Lecture Notes (Lab): IntroductionFile*. Three adjustments are made:

1. Computer connection: In fuzzy control simulation, we use angular position (e or angle θ) and angular speed (de) as input. We set initial values and use ODE (Runge-Kutta 4th Order ODE) for modeling. In experiment, we are using digital

fuzzy control and real time (update every 0.01s) data feedback, thus, no need of modeling but angle θ and displacement y provided by inner potentiometer of Carriage Module. Thus, we need to connect the colorful data cable for data transformation.

2. De-plug of red terminal (Control Console) and closure of Bytronic Interface Devices IIC.

3. Adjust the code so that it reads analog input and gives digital output and delete related modeling part (initial values and ODE)

Then we need to try different gains to make the system stable in a short time (IP stands upright). As we have learned from simulation, gains of angle and output are better to be larger and gains of angular speed is better to be smaller. Currently, the best result is reached when parameters are set as:

Gains of angular position g0: 1.5 Gains of angular speed g1: 3 Gains of angular speed h: 1.3

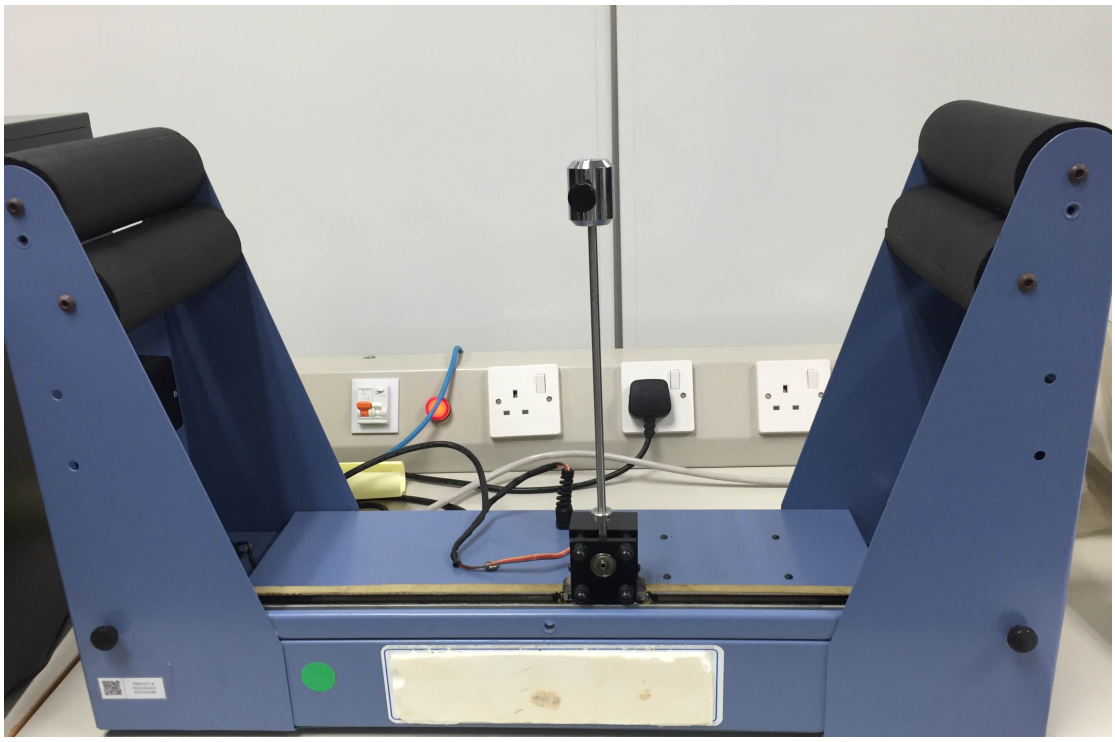

Fig.18 Stable system under fuzzy control

External force is applied three times (including the initial one) and the IP will always get stabilized at position -3, force and corresponding position tracks (are as shown in Fig.19 & Fig.20). Because data are examined by inner potentiometer, thus the vertical axis unit is (-10V~+10V/4096) and the horizontal axis unit is 0.01s.

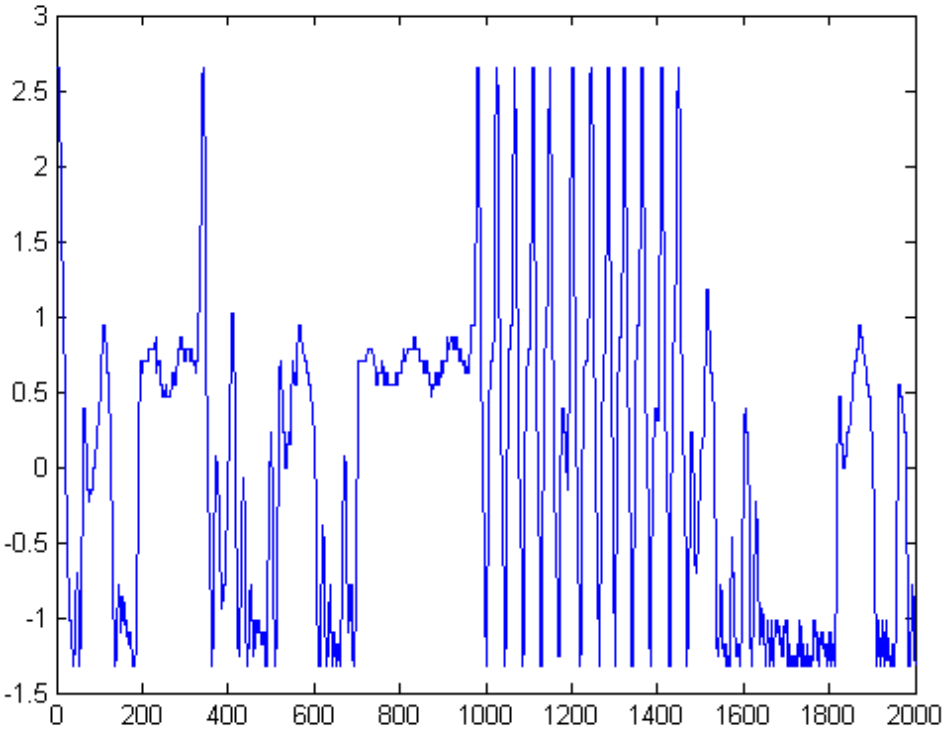

Fig.19 Disturbance force (measured by potentiometer) applied on the bob

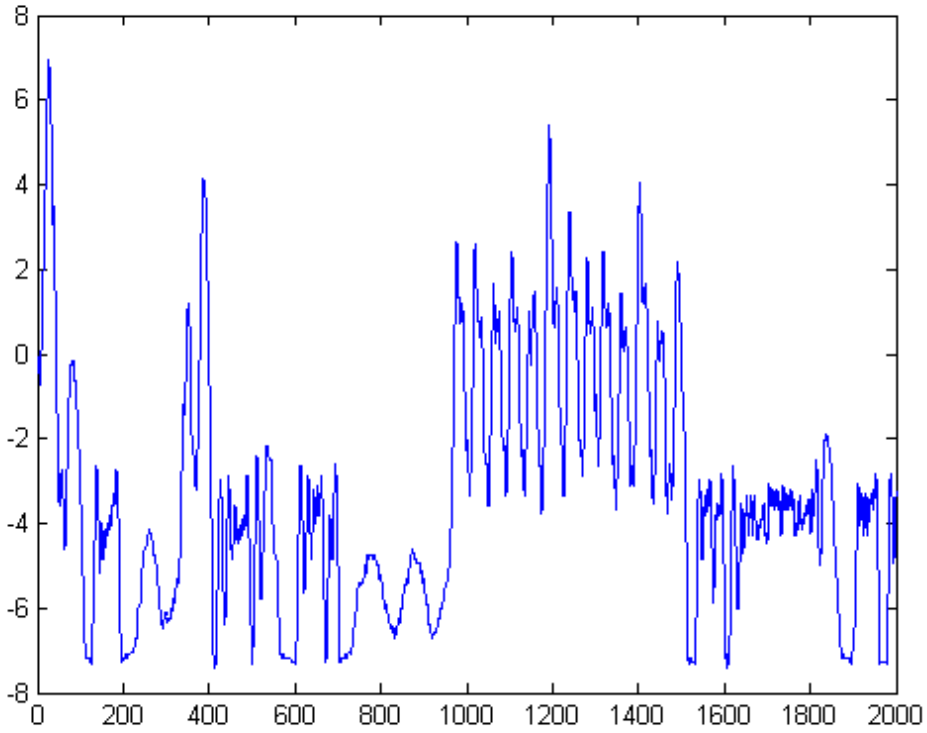

Fig.20 Position of bob (measured by potentiometer)

4.PID control

In the design process we will assume a single-input, single-output plant as described by the following transfer function. And we will attempt to control the pendulum's angle without regard for the cart's position.

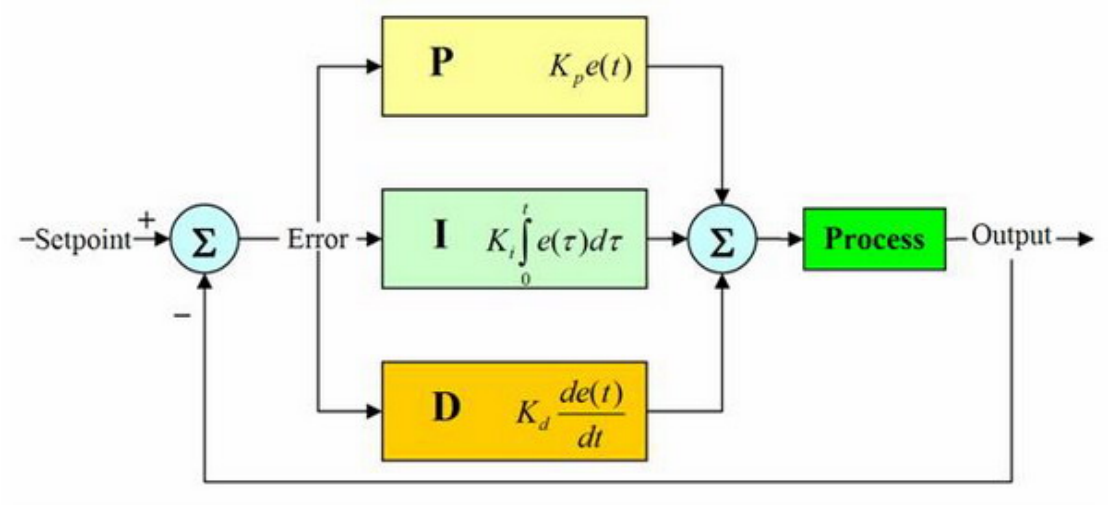

Fig.21 PID control principle

For the simulation of PID control, I choose to model the controlling parameters mathematically with estimated physical parameters.

Referring to Ref., we can get the transfer function of pole angle θ to input force expressed as:

$$\frac{\theta(s)}{F(s)} = -\frac{\frac{mLs^2}{q}}{s^4 + \frac{b(mL^2+J)}{q}s^3 - \frac{(M+m)mgL}{q}s^2 - \frac{bmgL}{q}s}$$

Apply the following physical parameters:

TABLE.02 Physical parameters of IP system

M	cart mass	1.0Kg
m	pole mass	0.1Kg
b	cart friction coefficient	01N(m/sec)
L	pole length	0.4m
J	rotational inertia	0.0053Kg*mˆ2

We can get the transfer function as:

$$G_\theta(s) = \frac{\theta(s)}{u(s)} = \frac{0.7329\ s^2}{s^4 + 0.0976\ s^3 - 20.1557\ s^2 - 1.8323\ s}$$

Then we can get the fist design of PID. Corresponding codes are in Appendix III.
To determine the values of Kp, Ki and Kd, we tried in separately weighted and imply an impulse disturbance to the system.

TABLE.03 Parameters of Kp, Ki, Kd

	Kp	Ki	Kd
Trial 1	1	1	1
Trial 2	100	1	1
Trial 3	100	1	20

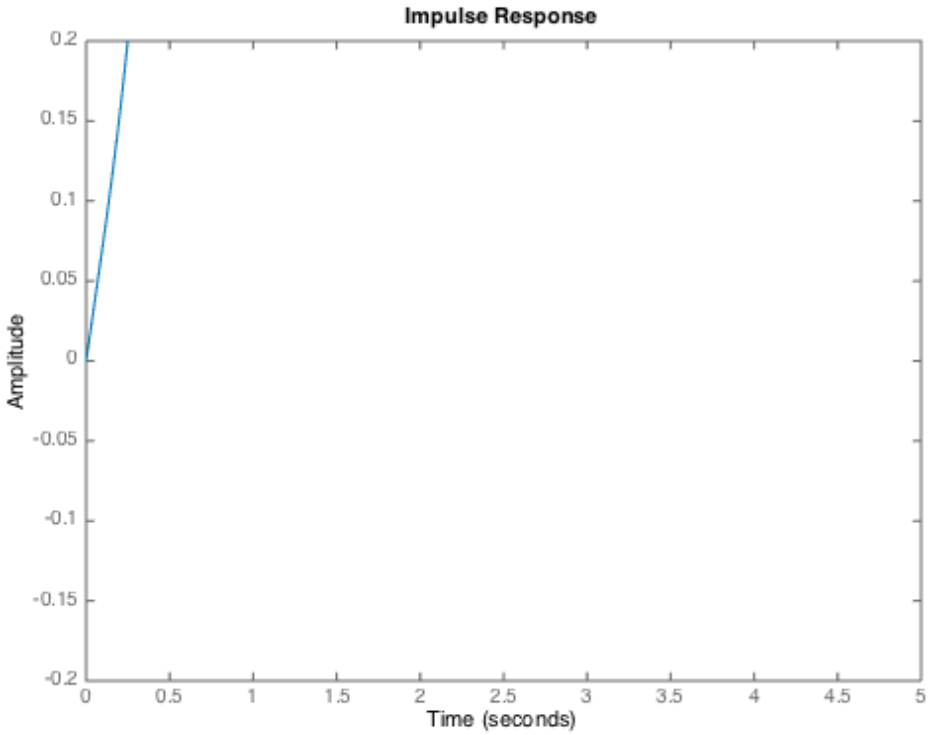

Fig.22 PID Impulse response on IP when Kp=1, Ki=1, Kd=1

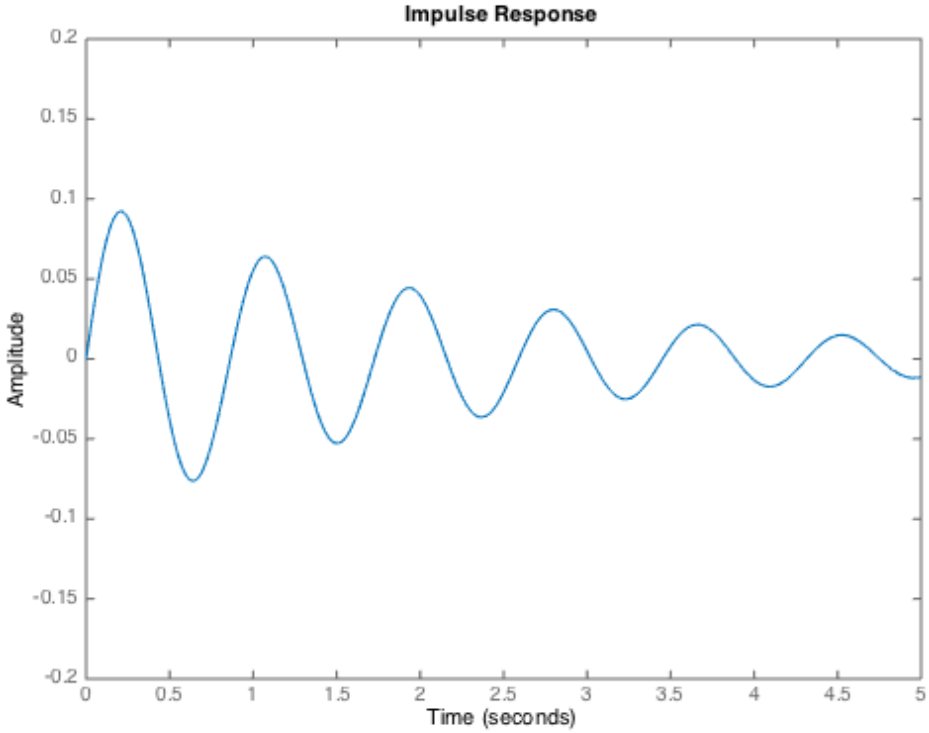

Fig.23 PID Impulse response on IP when Kp=100, Ki=1, Kd=1

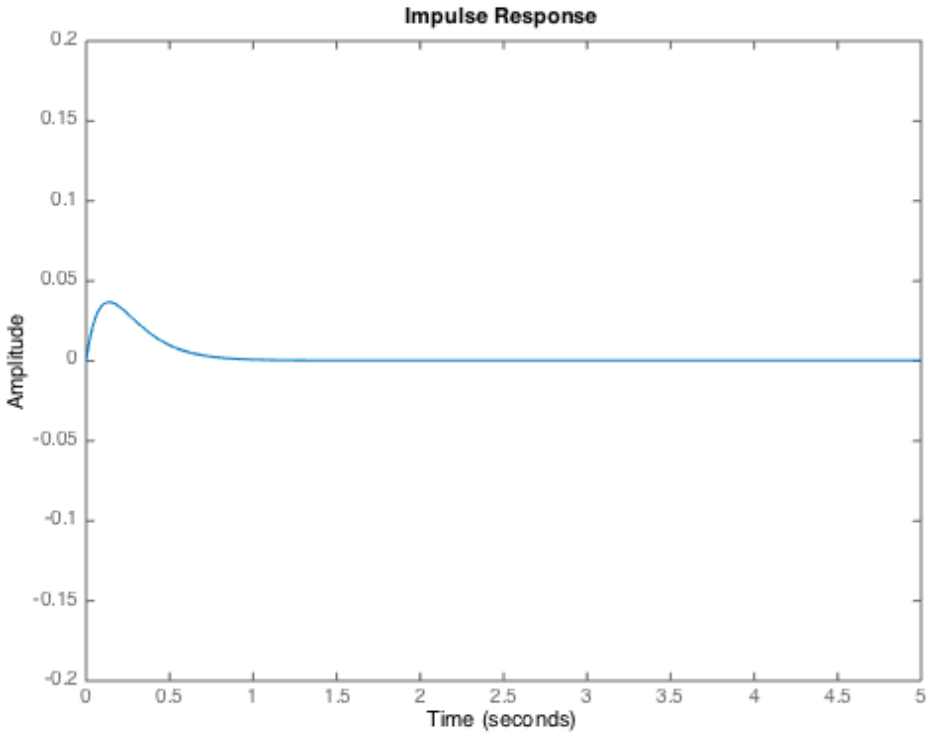

Fig.24 PID Impulse response on IP when Kp=100, Ki=1, Kd=20

The response is not stable in Trial 1 (Fig.22) and the proportional gain Kp is increased to modify the response. We can find the system in Trial 2 (Fig.23) is oscillating and the settling time is too long (over 5s) However, the steady state error is getting close to zero quickly. The settling time is getting better when Kd is increased in Trial 3 (Fig.24) and we find now the system has less than 1s settling time and small overshoot.

More specifically, the modification of Kp,Ki,Kd can usually affect the system :

TABLE.04 Parameters of Kp, Ki, Kd affects on system

Effects of	(on) Rise Time	Overshoot	Settling time	Steady-state error	Stability
↑ Kp	Decrease ↓	Increase ↑	Small Increase ↗	Decrease ↓	Degrade ↓
↑ Ki	Small Decrease ↘	Increase ↑	Increase ↑	Large Decrease ↓↓	Degrade ↓
↑ Kd	Small Decrease ↘	Decrease ↓	Decrease ↓	Minor Change →	Improve ↑

In certain cases, the increase of Kd will decrease the stability of system.

Apply the PID on the Bytronic© PCS1 Pendulum Control System, the inverted pendulum will go to the left end and stop. More conditions and parameters might be considered and imported to maintain the bob stands upright in future work.

References

1. Lecture Notes 3.2: Fuzzy_2File
2. Course Readings L1: Lab Maunal_Inverted Pedulum (2in1)File
3. PID 控制的一级倒立摆课程设计. (n.d.). Retrieved April 14, 2016, from http://wenku.baidu.com/view/13c1ba74a417866fb84a8eeb.html

Appendix I: Matlab Code for Fuzzy Control Simulation

```
ruleBase=[5 5 5 4 3; 5 5 4 3 2;5 4 3 2 1;4 3 2 1 1;3 2 1 1 1];
center=[-20 -10 0 10 20];  % set center for MF, assume the range to be
-20~20
x1=0.1;%initial angle
x2=0;% initial angular speed
interval=0.001;
t=0:interval:3; % time
force=zeros(1,length(t));
initForce=0;

theta(1)=x1;
thetaSpeed(1)=x2;
g0=5;
g1=0.3;
h=2;
for k=1:length(t)
    x1=g0*x1;
    x2=g1*x2;
    [mf1,mf2]=mfn(x1,x2);
    for i=1:5
        for j=1:5
            premise(i,j)=min(mf1(i),mf2(j));
        end
    end
    area=10-(1-premise)*10.*(1-premise);

    num=0;
    den=0;
    for i=1:5
        for j=1:5
          num=num+area(i,j)*center(ruleBase(i,j));
          den=den+area(i,j);
        end
    end
    u=num/den;
    u=h*u;
    force(k)=u;
    [x1,x2]=rk4(theta(k),thetaSpeed(k),interval,initForce);
    initForce=u*(1-exp(-100*t(k)));
    if k<length(t);
        theta(k+1)=x1;
        thetaSpeed(k+1)=x2;
    end
```

```
end
 subplot(2,1,1)
 plot(t,theta)
 ylabel('Angular position(rad)')
 grid on;
 subplot(2,1,2)
 plot(t,force)
 ylabel('Input force(N)')
 xlabel('Time(sec)')
 grid on;
```

Appendix II: Matlab Code for Fuzzy Control Experiment

```
clear;
dacout(608,0,2047);
ruleBase=[...
    5 5 5 4 3
    5 5 4 3 2
    5 4 3 2 1
    4 3 2 1 1
    3 2 1 1 1]
center=[-20 -10 0 10 20];
g0=1;
g1=2.3;
h=1.2;
r=-1;
% y, x,theta can be read, we get the corresponding voltage
tic;
for s=1:1:2000 % step
    e(s)=g0*(adcinp(608,3,1,1)/204.8-10);% theta
%     x(i)=adcinp(608,1,1,1)/204.8-10; % distance of chart
    y=adcinp(608,0,1,1)/204.8-10;
    e(s+1)=g0*(y-r);
    de(s)=g1*(e(s+1)-e(s));% angular speed
    [mf1,mf2]=mfn(e(s),de(s));
    for i=1:5
        for j=1:5
            premise(i,j)=min(mf1(i),mf2(j));
        end
    end

    area=10-(1-premise)*10.*(1-premise);
    num=0;
    den=0;
    for i=1:5
        for j=1:5
          num=num+area(i,j)*center(ruleBase(i,j));
          den=den+area(i,j);
        end
    end
    u=num/den;
    force(s)=h*(u+10)*204.8;
    dacout(608,0,force(s));
    time(s)=toc;
end
```

Appendix III Matlab Code for PID Control Simulation

```
% transfer function for IP angle
num=[0.7329 0 0];
den=[1 0.0976 -20.1557 -1.8323 0];

% tranfer function of PID controller
Kd=20;
Kp=100;
Ki=1;

numPID=[Kd Kp Ki];
denPID=[1 0];
numc=conv(num,denPID);
denc=polyadd(conv(denPID,den),conv(numPID,num));

% impulse response of PID controller
t=0:0.005:5
impulse(numc,denc,t);
```

```
function [poly]=polyadd(poly1,poly2)
if length(poly1)<length(poly2)
    short=poly1;
    long=poly2;
else
    short=poly2;
    long=poly1;
end
    mz=length(long)-length(short);
if mz>0
    poly=[zeros(1,mz),short]+long;
else
    poly=long+short;
end
```

Appendix IV Matlab Code for PID Control

```
dacout(608,0,2047);
kd=20;
kp=100;
ki=1;
T=0.01;
angle = zeros(1,3001);%get angle of bob
angle_sum = zeros(1,3001);
for k=1:1:3000
    y = (adcinp(608,3,1,1)/204.8)-10; % get position of bob
    angle(k+1) = y;
    angle_sum(k+1)=angle_sum(k)+angle(k);

force(k)=kd*(angle(k+1)-angle(k))+kp*angle(k+1)+ki*angle_sum(k+1)*T;
    dacout(608,0,force(k));
end
```