Chapter 2
Expert PID Control

Expert control is a control tactics to use expert knowledge and experience. Expert control was proposed firstly by Astrom in 1986 [1].

2.1 Expert PID Control

The expert PID control is to design PID parameters with characteristics of the plant and experience of the control expert, no need of modeling information.

The experience of the expert is mainly based on the error response of the system. Typical error response for a second transfer function is shown in Fig. 2.1; for the area I, III, V, VII, ..., where the absolute value of error tends to smaller, we can use unloop control; for the area II, IV, VI, VIII, ..., where the absolute value of error tends to bigger, we can use strong control or general control.

At time $k$, we consider the ideal position signal as $y_d(k)$, the output as $y(k)$, and then the tracking error is $e(k) = y_d(k) - y(k)$ at time $k$, and $e(k - 1)$ and $e(k - 2)$ represent the error at time $k - 1$ and $k - 2$, respectively, then we have

$$
\Delta e(k) = e(k) - e(k - 1)
$$
$$
\Delta e(k - 1) = e(k - 1) - e(k - 2)
$$

(2.1)

According to Fig. 2.1, we can do the following analysis:

(1) When $|e(k)| > M_1$, we can use unloop controller to minimize error quickly.
(2) When $e(k)\Delta e(k) > 0$ or $\Delta e(k) = 0$, we consider two conditions as follows:

If $|e(k)| \geq M_2$, we use strong PID controller as

$$
u(k) = u(k - 1) + k_1 \{k_p [e(k) - e(k - 1)] + k_i e(k) + k_d [e(k) - 2e(k - 1) + e(k - 2)]\}
$$

(2.2)
If $|e(k)| < M_2$, we use weak PID controller as

$$u(k) = u(k-1) + k_p[e(k) - e(k-1)] + k_i\frac{e(k)}{C_0} + k_d\frac{e(k)}{C_0}$$

(2.3)

(3) When $e(k)\Delta e(k) < 0$, $\Delta e(k)\Delta e(k-1) > 0$, or $e(k) = 0$, which indicates the absolute value of error tends to smaller or constant value, we can hold the control input.

(4) When $e(k)\Delta e(k) < 0$, $\Delta e(k)\Delta e(k-1) < 0$, which indicates the value of error is in extremism state. We consider the two conditions as follows:

If the absolute value of error is big, $|e(k)| \geq M_2$, we can use strong controller as

$$u(k) = u(k-1) + k_1 k_p e(k)$$

(2.4)

If the absolute value of error is small, $|e(k)| < M_2$, we can adopt weak controller as

$$u(k) = u(k-1) + k_2 k_p e(k)$$

(2.5)

(5) When $|e(k)| \leq \varepsilon$, which indicates the absolute value of error tends to very small, we can use PI controller to decrease the static error, where

- $u(k)$—control input at time $k$;
- $u(k-1)$—control input at time $k-1$;
- $k_1$—gain coefficient, $k_1 > 1$;
- $k_2$—gain coefficient, $0 < k_2 < 1$;
- $M_1$, $M_2$—limit values, $M_1 > M_2$;
- $\varepsilon$—positive value.
2.2 Simulation Example

Consider a plant as

\[ G_p(s) = \frac{523500}{s^3 + 87.35s^2 + 10470s} \]

The sampling time is 1 ms, using MATLAB command “c2d”, the plant can be discrete as

\[
y(k) = -\text{den}(2)y(k-1) - \text{den}(3)y(k-2) - \text{den}(4)y(k-3) \\
+ \text{num}(2)u(k-1) + \text{num}(3)u(k-2) + \text{num}(4)u(k-3)
\]

where \( \text{num}() \) and \( \text{den}() \) can be gotten by the command \text{tfdata}.

The ideal position signal is \( y_d(k) = 1.0 \). In the simulation program, due to the discretization, there is one delay in control input.

The simulation program of traditional PID controller is chap2_1.m, and the simulation results are shown in Figs. 2.2 and 2.3. The simulation program of expert PID controller is chap2_2.m, and the simulation results are shown in Figs. 2.4 and 2.5.

(1) Program of traditional PID Controller: chap2_1.m.

```matlab
% Expert PID Controller
clear all;
close all;
ts=0.001;
sys=tf(5.235e005,[1,87.35,1.047e004,0]); % Plant
dsys=c2d(sys,ts,'z');
[num,den]=tfdata(dsys,'v');
```

![Fig. 2.2 Step response with traditional PID control](image)
Fig. 2.3 Error response with traditional PID control

Fig. 2.4 Step response with expert PID control

Fig. 2.5 Error response with expert PID control
u_1=0; u_2=0; u_3=0; u_4=0;
y_1=0; y_2=0; y_3=0;
ei=0;

kp=0.96; ki=0.03; kd=0.01;

error_1=0;
for k=1:1:500
time(k)=k*ts;
yd(k)=1.0;
y(k)=-den(2)*y_1-den(3)*y_2-den(4)*y_3+num(1)*u_1+num(2)*u_2+num(3)*u_3+num(4)*u_4;
error(k)=yd(k)-y(k); % Calculating P
derror(k)=error(k)-error_1; % Calculating D
ei=ei+error(k)*ts;
u(k)=kp*error(k)+kd*derror(k)/ts+ki*ei; % PID Controller
u_4=u_3; u_3=u_2; u_2=u_1; u_1=u(k);
y_3=y_2; y_2=y_1; y_1=y(k);
error_1=error(k);
end
figure(1);
plot(time,yd,'b',time,y,'r','linewidth',2);
xlabel('time(s)'); ylabel('r,y');
figure(2);
plot(time,yd-y,'r','linewidth',2);
xlabel('time(s)'); ylabel('error');

(2) Program of expert PID control: chap2_2.m

% Expert PID Controller
clear all;
close all;
ts=0.001;

sys=tf(5.235e005, [1, 87.35, 1.047e004, 0]); % Plant
dsys=c2d(sys, ts, 'z');
[num, den] = tfdata(dsys, 'v');

u_1=0; u_2=0; u_3=0; u_4=0;
y_1=0; y_2=0; y_3=0;
ei=0;
error_1=0; derror_1=0;
kp=0.6;ki=0.03;kd=0.01;
for k=1:1:500
    time(k)=k*ts;
    yd(k)=1.0; %Tracing Step Signal
    %Linear model
    y(k)=-den(2)*y_1-den(3)*y_2-den(4)*y_3+num(1)*u_1+num(2)*u_2+num(3)*u_3+num(4)*u_4;
    error(k)=yd(k)-y(k); % Calculating P
    derror(k)=error(k)-error_1; % Calculating D
    ei=ei+error(k)*ts; % Calculating I
    u(k)=kp*error(k)+kd*derror(k)/ts+ki*ei; %PID Controller

%Expert control rule
    if abs(error(k))>0.8 %Rule1:Unclosed control rule
        u(k)=0.45;
    elseif abs(error(k))>0.40
        u(k)=0.40;
    elseif abs(error(k))>0.20
        u(k)=0.12;
    elseif abs(error(k))>0.01
        u(k)=0.10;
    end

    if error(k)*derror(k)>0 & (derror(k)==0) %Rule2
        if abs(error(k))>=0.05
            u(k)=u_1+2*kp*error(k);
        else
            u(k)=u_1+0.4*kp*error(k);
        end
    end

    if (error(k)*derror(k)<0 & derror(k)*derror_1>0) | (error(k)==0) %Rule3
        u(k)=u(k);
    end

    if error(k)*derror(k)<0 & derror(k)*derror_1<0 %Rule4
        if abs(error(k))>=0.05
            u(k)=u_1+2*kp*error(k);
        else
            u(k)=u_1+0.6*kp*error(k);
        end
    end

    if abs(error(k))<=0.001 %Rule5:Integration separation PI control
        u(k)=0.5*error(k)+0.010*ei;
    end
2.2 Simulation Example

\[ u_4 = u_3; u_3 = u_2; u_2 = u_1; u_1 = u(k); \]
\[ y_3 = y_2; y_2 = y_1; y_1 = y(k); \]
\[ \text{error}_1 = \text{error}(k); \]
\[ \text{derror}_1 = \text{derror}(k); \]
end

figure(1);
plot(time,yd,'r',time,y,'b:','linewidth',2);
xlabel('time(s)');ylabel('r,y');
legend('Ideal position','Practical position');

Reference

Intelligent Control Design and MATLAB Simulation
Jinkun, L.
2018, XV, 290 p. 137 illus., 95 illus. in color., Hardcover
ISBN: 978-981-10-5262-0