Research on Modeling and Simulation of Expert_PID Controlled Servo System Based on MATLAB/S-function

Fei Kang
Xi'an Institute of Optics and Precision Mechannics
Chinese Academy of Sciences(CAS)
Xi'an, Shaan Xi, China
e-mail: kangfei1213@sina.com

YanBing Liang
Xi'an Institute of Optics and Precision Mechannics
Chinese Academy of Sciences(CAS)
Xi'an, Shaan Xi, China
e-mail: lyb@opt.ac.cn

***Abstract*—The paper elaborates Expert_PID controller applied in servo system. Illustrates an S-function construction method on complex control law. Designs an Expert_PID controller based on MATLAB/S-function and writes partial core code of Expert_PID controller. Establishes the simulink model of servo system based on Expert_PID control law, gets the simulation result. Compared with the traditional discrete PID control law, the result demonstrates that Expert_PID controller can obtain excellent control qulity.**

Keywords-Expert_PID; servo system; modeling; simulation; S-function

I. INTRODUCTION

The proportional, integral and derivative controller which based on the deviation(error) of the system is referred to as a PID controller. PID control law is one of the earliest developed control strategy, it is an simple and effective control law that based on "past, present and future" of the estimated deviation. Due to its simple algorithm, robustness, high reliability, it is widely used in servo system controller designing, such as artillery, radar, theodolite and other tracking turntable. In PID controller, a crucial problem is the tuning of the PID parameters. The superior or inferior of parameters tuning will not only affects the control quality of the system, but will also affects the stability and robustness of the control system [4]. In actual servo system, due to the presence of coulomb friction, external interference, aging of executive body and other reasons, often causing uncertainties such as nonlinear, time-varying. These can lead to the PID parameters that had been tuning become bad, causing poor control performance of the system. All of these explained that the adaptive ability of the PID controller is poor. To solve the problem, for a long time, people have been seeking for a self-tuning PID controller technology to adapt to complex operating condition and high performance tracking requirement. The expert intelligent PID controller is the combination of expert experience with conventional PID control, its parameters can be tuning online. It can describe some complex algorithm to make adaptive ability of the controller greatly improved. Meanwhile, improves the tracking quality of control system [3]. Fig. 1 shows the block of expert rules tuning PID parameters in control system.

In this paper, take MATLAB as simulation tool, using S-function in simulink simulation toolbox to achieve expert

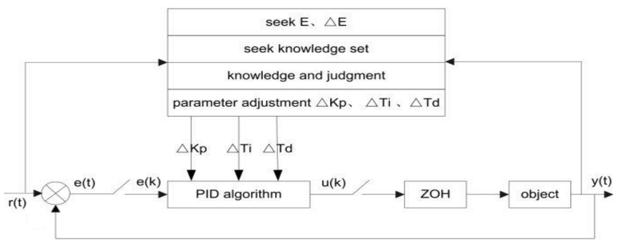

Figure 1. Expert_PID Control System.

PID control law. Take an actual position tracking servo system as studied object to study the Expert_PID control system. And also make a comparation with the discrete position PID algorithm and integral separate PID algorithm.

II. PRINCIPLE

A. PID Control Principle

PID controller is a linear and deviation control based controller. The given value r(t) and the actual output value y(t), the deviation(error) can be described like this [1]:

$$error(t) = y(t) - r(t) \tag{1}$$

PID control law can be expressed as:

$$u(t) = kp[error(t) + \frac{1}{T_I}\int_0^t error(t)dt + \frac{T_D derror(t)}{d(t)}] \tag{2}$$

The transfer function form is:

$$G(s) = \frac{U(s)}{E(s)} = k_p(1 + \frac{1}{T_I s} + T_D s) \tag{3}$$

k_p is proportional coefficient, T_I is integral time constant, T_D is derivative time constant.

Simply, the role of the PID is: (1) proportional-multiples deviation signal. Once it produced, the controller generates a control signal to decrease the deviation immediately. (2) Integral-be mainly used to eliminate static error, improve system type. Integral effect is strong or weak decided by integral time constant T_I . If T_I is small, integral effect is strong, else it's weak. (3) derivative-represents the changing rate of the deviation signal. It can generate a early effective correction signal before deviation signal becomes too big. Meanwhile, it can accelerate movement of the system, reduce adjustment time. But in modern servo system, we actually use computer control technology. So the discrete PID algorithm can be expressed as following:

$$t \approx kT \qquad (k = 0,1,2,...) \tag{4}$$

$$\int_0^t error(t)dt \approx T\sum_{j=0}^{k} error(jT) = T\sum_{j=0}^{k} j \tag{5}$$

$$\frac{derror(t)}{d(t)} \approx \frac{error(kT)-error((k-1)T)}{T} = \frac{error(k)-error(k-1)}{T} \tag{6}$$

Discrete PID expression is:

$$u(k) = k_p error(k) + k_i \sum_{j=0}^{k} error(j)T + k_d \frac{error(k)-error(k-1)}{T} \tag{7}$$

Among equations (4), (5), (6), (7):

$$k_i = \frac{k_p}{T_I}, k_d = k_p T_D \tag{8}$$

T is sampling time, k is sampling number, k=1, 2, …, $error(k)$ is deviation signal in the time k, $error(k-1)$ is deviation signal in the time (k-1) [2].

B. *Expert_PID Control Principle*

The essence of Expert_PID control is that: getting a variety of knowledge of the controlled object and the control law, there is no need to know the exact model of the controlled object. A typical linear system unit step response error curve is shown in Fig. 2 [6].

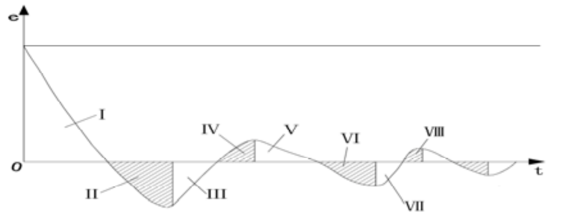

Figure 2. Typical Linear System Unit Step Response Error Curve.

Using expert experence to turning the PID parameters, where $e(k)$ represents the present sampling error, $e(k-1), e(k-2)$ separately represent unit delay and two order delay sampling time error, there is the following definition:

$$\Delta e(k) = e(k) - e(k-1) \tag{9}$$

$$\Delta e(k-1) = e(k-1) - e(k-2) \tag{10}$$

Making sorts of the error and designing corresponding control law according to the error and its trend. The expert rules have five, they are shown as following:

(1). If $|e(k)| > M1$, that represents the absolute value of current error is very lagre, implement open loop control. Giving controller a corresponding constant value to adjust the error rapidly. (2) If $e(k)*\Delta e(k) > 0$ or $e(k) = 0$, that represents the absolute value of error is increasing or the error is 0. a. If $|e(k)| > M2$, the error is large, implement a fairly strong control, the controller can be described as:

$$u(k) = u(k-1) + k1*\{kp*[e(k)-e(k-1)] + ki*e(k) + kd*[e(k) - 2*e(k-1) + e(k-2)]\} \tag{11}$$

b. If $|e(k)| < M_2$, the error is small, implement a general control, the controller output is:

$$u(k) = u(k-1) + kp*[e(k) - e(k-1)] + ki*e(k) + kd*[e(k) - 2*e(k-1) + e(k-2)] \tag{12}$$

(3). If $e(k)*\Delta e(k) < 0$ and $\Delta e(k)*\Delta e(k-1) > 0$ or $e(k) = 0$, that represents the absolute value of error is decresing or the error is 0. At this time, the controller output is kept constant, that is to say, using the traditional discrete PID control.

(4). If $e(k)*\Delta e(k) < 0$ and $\Delta e(k)*\Delta e(k-1) < 0$, that represents the error is at extremal state. a. If $|e(k)| > M2$ the controller is:

$$u(k) = u(k-1) + k1*kp*em(k) \tag{13}$$

b. If $|e(k)| < M2$ the controller output is:

$$u(k) = u(k-1) + k2*kp*em(k) \tag{14}$$

(5). If $|e(k)| <= E1$, that represents the absolute value of error is small. At this time, join integral role to eliminate error. In the formulas given above, variable $em(k)$ is the error's extremum, k1 is gain amplification coefficient, k2 is gain suppression coefficient, k1>1, 0<k2<1; M1,M2,E1 is error defined line, M1>M2>E1>0. In Fig. 2: Ⅰ,Ⅲ,Ⅴ,Ⅶ is the absolute value of error decreasing area where needs keep waiting, that is to say, we can implement open loop control. Ⅱ,Ⅳ,Ⅵ,Ⅷ is the absolute value of error increasing area where we can exert corresponding control law to decrease or eliminate the error according to it is large or small [6].

III. S-FUNCTION AND ITS USE MECHANISM

S-function module is a senior function module in simulink, it is the core of the dynamic simulink system. In a word, S-function is the most attractive aspect of simulink. As long as a researched system can be described by the MATLAB programming language, we can construct its S-function. When the system uses a complex control law or the controlled object is a complex model, there is no existing module available, usually we use MATLAB programming language to write a large number of complex and cumbersome source code for simulation. If we realize the control law or the object model by S-function, we can avoid the original method of programming directly, make modeling and simulation process more intuitive, simple and easy to debug. S-function call format is:
sys=function_name (t, x,u,flag,paramt1,paramt2…paramtN);
Wherein function_name is the controller or object name. t, x, u are respectively the current moment, the status vector and the input vector. The value of the variable flag control the information of returned variable sys, paramt1, paramt2… paramtN is user-defined parameters. Simulink calls different functions to complete the system dynamic simulation according to the value of the flag. Following is the value of flag and corresponding function of S-function [7].

- Flag=0, complete system initialization.
- Flag=1, return status derivative of continuous system.
- Flag=2, update the status of the discrete system.
- Flag=3, return output vector of the system.
- Flag=4, update the system next sampling time.
- Flag=9, terminate the simulation.

During initialization, the initialization function completes the update of the sizes structure, the meaning of each variable in the sizes structure is:

- sizes.NumContStates, number of continuous status.
- sizes.NumDiscStates, number of discrete status.
- sizes.NumOutputs, number of output vector.
- sizes.NumInputs, number of input vector.
- sizes.NumSampleTimes, number of sampling time.

In addition to the variable sys, we should also set the initial status x0, description variable str and sampling period variable ts. The variable ts is a double-column matrix, each row represents a sampling period. For continuous system and single sampling period system, the variable matrix is [t1,t2], where t1 is the sampling period, t1=-1 demostrates the controller inherit the sampling period of input signal, t2 is offset, generally taken to be 0. For continuous system, ts=[-1, 0].

IV. EXPERT_PID CONTROLLER DESIGN

Because Expert_PID control law can not be simply described with the transfer function, the following procedure is the implementation of Expert_PID control law in S-function format. After encapsulation, it can be used as the controller of the system. So we can establish simulink model of servo system for simulation. Expert_PID controller is designed to follow the expert rules mentioned above. The format of S-function is: function [sys, x0, str, ts] = Expert_PID(t, x, u, flag) [7].

A. *Initialization Phase*

The module uses discrete time domain design. Expert_PID controller has three inputs, they are respectively u(1), u(2), u(3). They separately represent the current error sampling signal, unit delay error sampling signal, second order delay error sampling signal. The controller has one output, sys, that is regulated variable. The controller has three discreat status they are respectively x(1), x(2), x(3), they separately represent proportional, derivation, integral. Its initial status is x0=[0;0;0]. Because the controller output is directly related to the input, it has direct feedthrough. The controller has one sampling time T=0.001s. So Expert_PID controller has the following block diagram, shown in Fig. 3. The controller sizes structure variables value are set as:

- NumContStates= 0.
- NumDiscStates= 3.
- NumOutputs= 1.
- NumInputs= 3.
- DirFeedthrough= 1.
- NumSampleTimes= 1.

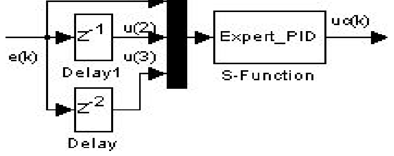

Figure 3. Expert_PID Controller Block Diagram.

B. *The Controller Output Phase*

At this stage, assign controller output variable uc to the system variable sys, as the output function return value. Take expert rule 1 and rule 2 for example, give partial core code as following, where uc_1 is the controller output value of last sampling time [8].

```
function sys = mdlOutputs(t, x, u)
// Expert rule 1.
if abs(x(1))>0.50            uc=0.50;
elseif abs(x(1))>0.25        uc=0.25;
elseif abs(x(1))>0.10        uc=0.10;
elseif abs(x(1))>0.02        uc=0.02;
end
// Expert rule 2. kp=0.0085; ki=0.0145; kd=0.02.
if (x(1)*x(2)>0|(x(2)==0)
    if abs(x(1))>=0.01
        uc=uc_1+2*(kp*(u(1)-u(2))+ki*u(1)+kd*(u(1)-
            2*u(2)+u(3)));
    else    uc=uc_1+kp*(u(1)-u(2))+ki*u(1)+kd*(u(1)-
            2*u(2)+u(3));
    end  end
sys=uc;
```

C. *Discrete Status Update Phase*

For discrete PID status, dynamic characteristics of the controller update at this stage. In Expert_PID control module, update the value of each status according to the expert rules. The status change with the changing of sampling time and input error. Among, x(1)=u(1) is the proportional part; x(2)= (u(1)-u(2))/T is the derivative part; x(3)= x(3)+u(1)*T is the integral part. Whenever, there is a new input of error sampling signal, the entire status update. T is the sampling time. The returned value of discrete status update function is the vecter:

sys=[u(1); (u(1)-u(2))/T; x(3)+u(1)*T].

V. SYSTEM MODELING AND SIMULATION

Transfer function of the controlled object in servo system position loop is:

$$G_p(s) = \frac{8000}{0.001s^2 + s} \tag{15}$$

Establish the simulink model of the servo system based on the Expert_PID control law previously described. Fig. 4 shows the position loop simulink model of the servo system [5]. Demanding that the system steady-state tracking error is less than 0.001. The maximum value of controller output is set to be 2. The controlled object is continuous model, the controller is discrete digitizing. The sample time T=0.001s. Simulation results are shown on Fig. 5 and Fig. 6. Simulation results expressed that in the range of allowable error, Expert_PID control system can track the step and sine signal perfectly, the stead-state error is far less than 0.001. Fig. 7 shows the curve of tracking performance comparations between Expert_PID (No.3), discrete position PID(No.1)and discrete integral separation PID(No.2).

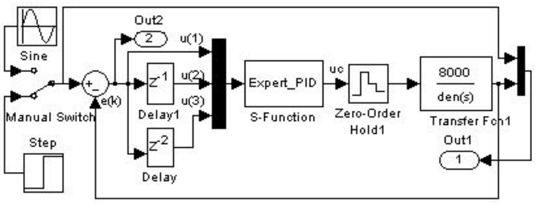

Figure 4. The Simulink Model of the Servo System.

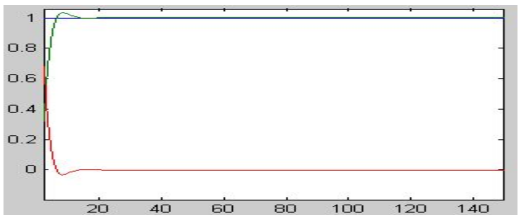

Figure 5. Step Signal Tracking and Error Curve.

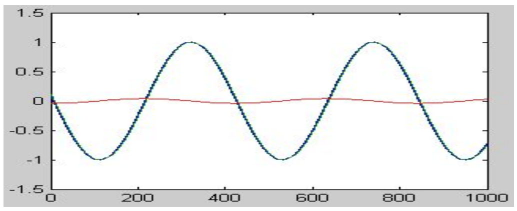

Figure 6. Sine Signal Tracking and Error Curve.

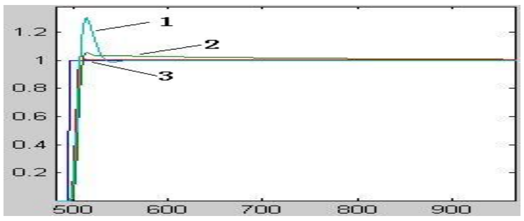

Figure 7. Unit Step Signal Tracking Performance Comparation.

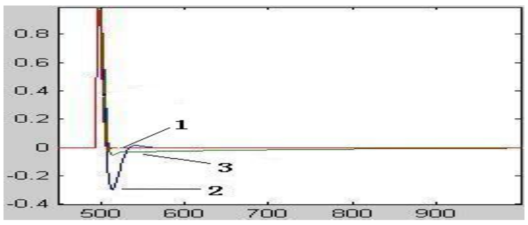

Figure 8. Unit Step Signal Tracking Error Comparation.

Fig. 8 shows the curve of tracking error of the three control law. The result is that Expert_PID(No.1) controlled system has the smallest steady-state error and the dynamic error tracking curve can be the best convergence. The integral separation(No.3) tracking effect is not as good as the Expert_PID. The discrete position PID(No.2) tracking effect is poor, to some degree. The three control laws dynamic tracking performance are shown in TABLE 1 [1] , the key performance indicators tell us Expert_PID control performance is the best of the three control laws.

TABLE I. DYNAMIC PERFORMANCE COMPARATION.

Control Law / **Dynamic Performance**	**Expert_PID Control**	**Discrete Position PID**	**Discrete Integral Separation PID**
Overshoot $\sigma\%$	1.2%	32.6%	3.9%
Rise time $\mathrm{tr}(s)$	0.0053	0.0060	0.0087
Adjust time $ts(s)$	0.0072	0.0336	0.0124

VI. CONCLUSION

Gives an introduction of traditional discrete PID control and Expert_PID control principle. Applies Expert_PID controller in an actual servo system position loop designing. To realize Expert_PID control law, using MATLAB S-function. Then, establishing the servo system simulink model. The simulation results tell that Expert_PID can obtain the best dynamic and steady-state performance compared with discrete position PID and discrete integral separation PID. Futher more, its dynamic tracking error convergence speed is the best of the three. The successful application of Expert_PID control demonstrates its adaptive capacity, robustness and excellent control quality.

ACKNOWLEDGMENT

First and foremost, I would like to show my deepest gratitude to my supervisor, researcher YanBing Liang, a respectable, responsible and resourceful scholar, who has provided me with valuable guidance in servo system modeling. Without his enlightening instruction, impressive kindness and patience, I could not have completed this thesis. His keen and vigorous academic observation enlightens me not only in this thesis but also in my future study. I shall extend my thanks to my girlfriend and other classmates for their kindness and help.

REFERENCES

[1] ShouSong Hu, Principle of Automatic Control(the fifth edition), Science Press, June 2007.

[2] ShouHong Lai, Microcomputer Control Technology, Machinery Industry Press, May 2000.

[3] YaoNan Wang, Intelligent Control Theory and Applications, Machinery Industry Press, February 2008

[4] DaYong Yang, Ming Li, Simulation and application of expert PID control in the process control device, Computer and Modernization, 2008.

[5] ZhongSen Huang, Jing Huang, Calculation and Simulation of control systems by MATLAB(the third edition), National Defence Industry Press.

[6] JinKun Liu, Advanced PID control MATLAB simulation(the third edition), Electronic Industry Press, March 2011.

[7] Simulink Dynamic System Simulation for Matlab Writing S-Functions As M-Files .THE MATH WORKS, 2009.

[8] ZhiQiang He, Parameters Tuning of PID Controller And Its Application. ZhejiangUniversity, March 2008.