REINFORCEMENT LEARNING CONTROL FOR SHIP STEERING USING RECURSIVE LEAST-SQUARES ALGORITHM

Zhi-peng Shen[1] , Chen Guo[1,2] and Shi-chun Yuan[1]

[1]Lab of Simulation and Control of Navigation Systems,
Dalian Maritime Univ., Dalian 116026, China;
[2]State Key Laboratory of Intelligent Technology and system,
Tsinghua University, Beijing 100084,China
(E-mail: s_z_p@263.net ; guoc@dlmu.edu.cn)

Abstract: Recursive least-squares temporal difference algorithm (RLS-TD) is deduced, which can use data more efficiently with fast convergence and less computational burden. Reinforcement learning based on recursive least-squares methods is applied to ship steering control, as provides an efficient way for the improvement of ship steering control performance. It removes the defect that the conventional intelligent algorithm learning must be provided with some sample data. The parameters of controller are on-line learned and adjusted. Simulation results show that the ship course can be properly controlled in case of the disturbances of wave, wind, current. It is demonstrated that the proposed algorithm is a promising alternative to conventional autopilots. *Copyright © 2005 IFAC*

Keywords: Recursive squares methods; Learning algorithm; Action network; Ship control ;Simulation

1. INTRODUCTION

Ship steering, in general, is a complicated control problem. From the 1920's, it has experienced many develop phases such as PID control, adaptive control and intelligent control etc. In recent years, hybrid intelligent system (HIS) which is composed of neural network (NN),fuzzy logic control (FLC), reinforcement learning (RL) and genetic algorithm (GA) has been successfully used in home appliances (Wakami, et al., 1996) and robotics (Zhou,1997), in ship steering field some primary works are also presented (Sutton,et al.,1997).

Considering the nonlinear characteristics of ship motion and the complex correlation related to ship maneuvering, velocity and the changing environments, the controlled plant has obvious uncertainties. If controller parameters can be on-line adjusted according to environment conditions, it will effectively solve the uncertainties in control. The ordinary way of neural fuzzy network that uses NN to modify control parameters is to utilize supervised learning algorithm. But such algorithm needs some sample data and the sample data should be complete and correct. Unfortunately, such detailed and precise sample data may be very expensive or even impossible to obtain in ship steering applications. Reinforcement learning needs only simple "evaluative" information, which can be easily obtained. It estimates the control effect by interacting with the environment, and trains controller network using "award" and "punish" algorithm, unlike the supervised learning given the right answer. In reinforcement learning , the most common algorithm is temporal difference (TD) learning, which wastes data and may require sampling many trajectories to reach convergence. In order to using data more efficiently and fasting convergence, least-squares (LS) methods is used in reinforcement learning. Recursive least-squares temporal difference (RLS-TD) algorithm is also deduced to solve the computational and memory problems. In this paper, the recursive least-squares(RLS) methods based reinforcement learning is applied to ship steering to learn and adjust the parameters in on-line period. Simulation results show that the ship course can be properly controlled under the disturbances of wave, wind, current and error in measure apparatus, and demonstrate the proposed algorithm is feasible.

2.THE RLS-TD ALGORITHM

The credit-assignment problem is important in reinforcement learning. When the reinforcement signal and the environmental input pattern intensively depend on the history of the controller output, the problem becomes more obvious, especially as the reinforcement signal can be attained only by a long output sequences. TD algorithm is the most famous to solve the problem which is presented by Sutton in 1988(Sutton,1988;1984). TD distributes credit through the differences between two successive predictions.

Prediction is to predict a variable through the observed data. Consider the state $s_1, s_2, \ldots, s_t, \ldots$ with the observed data $x_1, x_2, \ldots, x_t, \ldots$.,for each state transition from s_t to s_{t+1} , a reinforcement signal r_t is defined. The value function of each state is defined as follows:

$$p(s_t) = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \qquad (1)$$

where $0 < \gamma \le 1$ is a discount factor. Temporal differences are defined as the differences between two successive estimations and have the following form.

$$\hat{r}_t = r_t + \gamma \widetilde{p}(s_{t+1}) - \widetilde{p}(s_t) \qquad (2)$$

where s_{t+1} is the successive state of s_t , $\widetilde{p}(s)$ denotes the estimate of the value function $p(s)$ .Consider a general linear function approximator with a fixed basis function vector, the estimated value function can be denoted as

$$\widetilde{p}(x_t) = \phi^T(x_t) W_t \qquad (3)$$

where $\phi(x_t) = (\phi_1(x_t), \phi_2(x_t), \cdots, \phi_n(x_t))^T$ is the state's observed data vector , $W_t = (w_1, w_2, \cdots, w_n)^T$ is the weight vector. Based on gradient-descent methods., the corresponding incremental weight update rule is

$$W_{t+1} = W_t + \eta(r_t + \gamma\phi^T(x_{t+1})W_t - \phi^T(x_t)W_t))Z_t \qquad (4)$$

where η is the learning factor , $Z_{t+1} = \gamma\lambda Z_{t+1} + \phi(x_t)$ is the eligibility trace vector. The above linear TD algorithm is proved to converge with probability 1(Brartke. and Barto,1996) which satisfies the following equation.

$$E[A(X_t)]W - E[B(X_t)] = 0 \qquad (5)$$

where $A(X_t) = Z_t(\phi(x_t) - \gamma\phi(x_{t+1})^T)$, $B(X_t) = Z_t r_t$ .The update equation (4) shows that the changes to W depend only on the most recent trajectory, and after those changes are made, the trajectory and its rewords are simply forgotten. The approach, while requiring little computation per iteration, wastes data and may require sampling many trajectories to reach convergence. In order to using data more efficiently and fasting convergence, the least-squares(RL) methods is combined with TD. For the estimate of the value function $\widetilde{p}(s)$ discussed above, when linear function approximators are used, the least-squares estimation problem of (5) has the following objective function.

$$J = \left\| \sum_{t=1}^{T} A(X_t)W - \sum_{t=1}^{T} B(X_t) \right\|^2 \qquad (6)$$

where $A(X_t), B(X_t)$ are defined as (5), $\|\cdot\|$ is a Euclid norm. Then the least-squares estimate of the weight vector W is computed according to the following equation.

$$W = A_T^{-1} B_T = (\sum_{t=1}^{T} A(X_t))^{-1} \sum_{t=1}^{T} B(X_t) \qquad (7)$$

where

$$A_T = \sum_{t=0}^{T} A(X_t) = \sum_{t=0}^{T} Z_t(\phi(x_t) - \gamma\phi(x_{t+1})^T),$$

$$B_T = \sum_{t=1}^{T} B(X_t) = \sum_{t=1}^{T} Z_t r_t$$

As is well known in system identification, adaptive filtering and control, recursive least-squares methods are commonly used to solve the computational and memory problems of least-squares algorithms. In order to deduce the RLS-TD algorithm based on the above idea., the matrix inverse lemma is first given as follows:

if $A \in R^{n \times n}, B \in R^{n \times 1}, C \in R^{1 \times n}$ and A is invertible, then

$$(A + BC)^{-1} = A^{-1} - A^{-1}B(I + CA^{-1}B)^{-1}CA^{-1} \qquad (8)$$

let $P_t = A_t^{-1}$, $P_0 = \delta I$, $K_{t+1} = P_{t+1} Z_t$

where δ is a positive number and I is the identity matrix. According to equation (7) and (8), the weight update rules of RLS-TD are given by

$$K_{t+1} = P_t Z_t / (1 + (\phi(x_t) - \gamma\phi(x_{t+1}))^T P_t Z_t) \qquad (9)$$

$$W_{t+1} = W_t + K_{t+1}(r_t - (\phi(x_t) - \gamma\phi(x_{t+1}))^T W_t) \qquad (10)$$

$$P_{t+1} = P_t - P_t Z_t (1 + (\phi(x_t) - \gamma\phi(x_{t+1}))^T P_t Z_t))^{-1} \\ ((\phi(x_t) - \gamma\phi(x_{t+1}))^T P_t \qquad (11)$$

With the initial conditions and the successive observed data, the weight vector W can be estimated by equation (9), (10) and (11), as a result the estimated value function $\widetilde{p}(x_t)$ is attained.

3 REINFORCEMENT LEARNING CONTROL FOR SHIP STEERING

Unlike the supervised learning problem in which the correct "target" output values are given for each input pattern to instruct the network's learning, in reinforcement learning only simple "evaluative " or

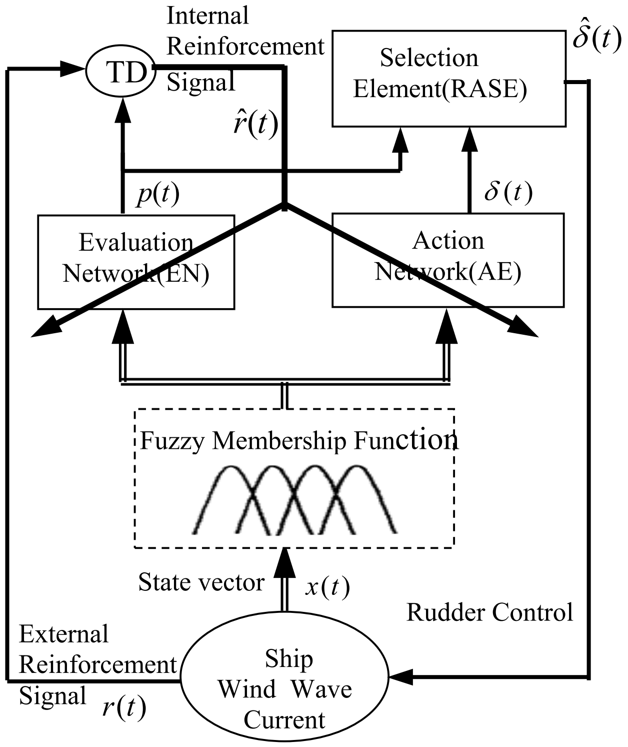

Fig.1 The structure of reinforcement learning

"critic" information are needed for learning. In the extreme case, these is only a single bit of information, to indicate whether the output is right or wrong. This is very significant in ship steering control. Under the disturbances of wave, wind, current and error in measure apparatus, only fuzzy information indicating current control effect such as good, normal or bad etc are provided. Depending on reinforcement learning, ship control effect can be improved in a certain extent by on-line adjusting the rudder angle and control parameters.

The control structure is illustrated in Fig.1. It is composed of evaluation network(EN), action network (AN) and random action selection element (RASE). Evaluation network is used to evaluate current action according to external reinforcement signal $r(t)$ and environmental state data $x(t)$, and outputs evaluation signal $p(t)$. The internal reinforcement signal $\hat{r}(t)$ is provided by TD algorithm. Action network generates control action $\delta(t)$ according to environmental state data $x(t)$ and evaluation signal $p(t)$, and trains the network's weights. Random action selection element selects a control action $\hat{\delta}(t)$ from the action set according to $\delta(t)$, and acts on ship.

3.1 Action Network

Action network is a general fuzzy cerebellar model articulation controller (GFCMAC). Its output is the rudder angle δ used to control ship motion. Fig.2 illustrates the structure of the action network. For ship course control, course angle ψ and yaw rate γ should be firstly considered. So the inputs of this controller are $e = \psi_d - \psi$ (ψ_d is the set course, ψ is the actual course) and $\gamma = \frac{d\psi}{dt}$. The output is the computed rudder δ .

Mapping $X \Rightarrow M$ uses the rule of conventional CMAC.

$$m_{i,k} = \left\lfloor \frac{q_i + N_g - k}{N_g} \right\rfloor \cdot N_g + k \qquad (12)$$

where $m_{i,k}$ is the address of mapping from input vector $q_i(x_j)$ to middle variable m , N_g is the number of excited units, k is the ordinal number of excited unit and $k = 0 \sim (N_g - 1)$. $\lfloor \cdot \rfloor$ is the floor function. The membership function of input variable is defined as Gaussian function.

$$\mu_j(x) = \exp\left(\frac{\left\| x - c_j \right\|^2}{\sigma_j} \right) \qquad (13)$$

When the mapping $X \Rightarrow M$ is determinate, the position a of input vector in A is given by searching the table, and shown as

$$a = E(\Lambda, \mu) \qquad (14)$$

where $\Lambda = m_1 \otimes m_2 \otimes \cdots \otimes m_n$ is tensor product operator , $\mu = \prod_{i=1}^{n} \mu_i$. And the excited unit address is denoted as

$$a_i = \{a_0, a_1, \cdots, a_{N_g - 1}\} \qquad (15)$$

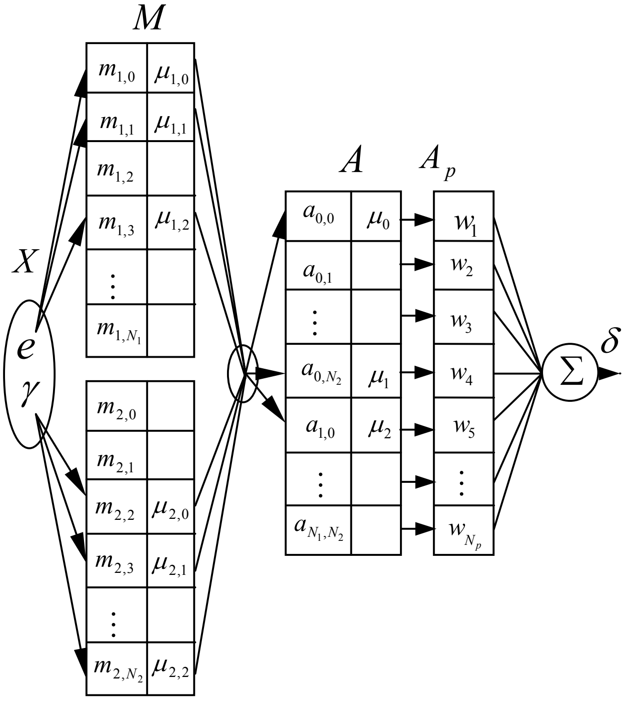

Fig.2. The structure of action network

The output mapping is

$$\delta = a^T h(x) w \qquad (16)$$

where x is the sample input vector, a^T is the address vector of excited units. And $h(x)$ is the weight vector..

3.2 Evaluation Network

Because of the dynamic effect of the ship, reinforcement signal $r(t)$ produced by current control value $\delta(t)$ can only be known at time step $t+1$. The action of evaluation network is to predict the possible operation state of the ship according to current input information, so the action network can learn and modify parameters in advance to improve the control performance. In this paper, the evaluation network is a radial basis functions neural network , and its inputs is the same as action network. As radial basis functions neural network is linear , it is convenient to use the RLS-TD algorithm to predict the value function.

3.3 Reinforcement Signal

Using the conventional learning rule(for example BP algorithm) to train the neural fuzzy network on-line, the correct "target" output value $\hat{\delta}(t)$ should be given for each input pattern $x(t)$ to instruct the network's learning. But it is very difficult to satisfy such a requirement, so we turn to use reinforcement learning algorithm. Reinforcement learning makes the neural fuzzy network posses adaptive ability. The closed loop control effect is described by a reinforcement signal r . Unlike the supervised learning , r is only the fuzzy evaluation to current control effect. In the extreme case, it is even a two-value number $r \in \{-1,1\}$, $r=-1$ means "failure" and $r=1$ means "success". r can also be a continuous number in the range $[-1,1]$, corresponding to different degree of success or failure. the larger r , the better control effect. For ship course control in this paper, r is defined as

$$r = 1 - 2\left|\frac{e}{e_{\max}}\right| \qquad （17）$$

where $e = \psi_d - \psi$ is the course error(ψ_d is the set course, ψ is the actual course). If $e > e_{\max}/2$ then $r<0$. If $e < e_{\max}/2$ then $r>0$. The meaning of equation(19) is that for ship course control the less the error ,the larger the reinforcement signal r and the better the control effect. And the degree of error can be shown in detail for r is a continuous number in the range $[-1,1]$.

3.4 Parameter Learning

Because the output $p(t)$ of evaluation network is the predicted reinforcement signal used to predict $r(t)$,its learning should be prior to the action network. The previous proposed RLS-TD algorithm is used to learn the weights of the evaluation network by equation (9),(10)and (11) .

The goal of the action network is to maximize the evaluation signal $p(t)$ of every state. The corresponding incremental weight update rule is

$$\Delta w_s(t) = \eta \frac{\partial p(t)}{\partial w_s(t)} = \eta \frac{\partial p(t)}{\partial \delta(t)} \frac{\partial \delta(t)}{\partial w_s(t)} \qquad (18)$$

where η is the learning factor. Here stochastic real valued algorithm is used to estimate the gradient information. The output δ of action network dose not directly act on ship. In stead, it is treated as a expected rudder angle. The actual rudder angle is chosen by exploring a range $\sigma(t)$ around δ . $\sigma(t)$ is a variable of Gaussian probabilistic distribution.

$$\sigma(t) = F[p(t)] = \frac{K}{1+e^{p(t)}} \qquad (19)$$

where K is a scaling coefficient. Once $\sigma(t)$ is confirmed, the actual rudder angle $\hat{\delta}(t)$ can be calculated.

$$\hat{\delta}(t) \sim N(\delta(t), \sigma(t)) \qquad (20)$$

where $N(\cdot)$ is a normal distribution function. And the gradient information is estimated by

$$\frac{\partial p(t)}{\partial \delta(t)} = a\hat{r}(t) \cdot [\frac{\hat{\delta} - \delta)}{\sigma}]_t \qquad (21)$$

where $(\hat{\delta} - \delta)\big/\sigma$ indicates the standard error between actual rudder angle $\hat{\delta}(t)$ and expected rudder angle $\delta(t)$. Equation (21) shows that if $\hat{r}(t) > 0$, actual rudder angle $\hat{\delta}(t)$ is better than expected rudder angle $\delta(t)$,and $\delta(t)$ should be close to $\hat{\delta}(t)$; vice versa. Once $\partial p(t)\big/\partial \delta(t)$ is attained, the weights of the action network can be learned by equation (18).

4 SIMULATION RESULTS

The above algorithm is used in ship steering control. When training data are available no-line, the on-line supervised learning algorithm can perform very well (Yang,2002). But considering the real status of ship navigation, considerable error in the measurement signal may exist, the precise information is not easily obtained. Reinforcement learning that only needs simple fuzzy feedback information has practical meaning in this case

Ship motion can be described either in state space mode or by input-output model. The former can deal with multivariable problem of ship steering control and the disturbances caused by waves, wind and currents directly and more accurately, but the computation burden is more heavy. The latter is also called response model, it omits the sway velocity but grasps the main characteristics of ship dynamics: $\delta \to \dot{\psi} \to \psi$,and the obtained differential equation can still preserve the nonlinear. The disturbances of wind, waves can even be converted to a kind of equivalent disturbance rudder angle as an input signal. In fact, response model is an extension of the linear Nomoto model. The second-order Nomoto model is

$$\ddot{\psi} + \frac{1}{T}\dot{\psi} = \frac{K}{T}\delta \tag{22}$$

To some unstable ship, $\dot{\psi}/T$ must be replaced with a non-linear term $(K/T)H(\dot{\psi})$ and $H(\dot{\psi}) = a\dot{\psi} + \beta\dot{\psi}^3$ So the second-order non-linear ship response model is expressed

$$\ddot{\psi} + \frac{K}{T}H(\dot{\psi}) = \frac{K}{T}\delta \tag{23}$$

parameters a, β and K, T is related to ship's elocity. In the paper, simulation studies are made refer to cargo ship .Set the initial ship speed $V_0 = 7.2m/s$,then the dimension parameters are, $T = 100s$, $K = 0.16\ 1/s$.

Fig.3 shows the control curve result when set course is 40˚, wind force is Beaufort 3 and wind direction is 30˚ .While Fig.4 and Fig 5 show the control curve result when set course is 10˚~ -10˚ ~ 30˚ ~ 0˚.The curves indicate that the course tracking is fast, control action reasonable and meet the performance of ship steering. The control result is partial satisfied. For further test of performance of the proposed algorithm, simulate the case that the instrument has measurement error by adding a constant disturbance (3˚). Fig.6 is the control curve where the constant disturbance is added at time step 200s and set course is 30˚. It is easily to see that the reinforcement learning can evidently reduce the static control error, but for the temporary control error it has little help.

5 CONCLUSIONS

In this paper, recursive least-squares temporal difference algorithm(RLS-TD) is deduced, which uses data more efficiently with fast convergence and less computational burden compared to conventional temporal difference algorithm.. Reinforcement learning based on recursive least-squares algorithm is applied to ship steering control, as provides an efficient way for the improvement of ship steering control performance. It removes the defect that the conventional intelligent algorithm learning must be provided with some sample data. The parameters of controller are on-line learned and adjusted. It can deal with the uncertainty of ship control in a way. Simulation results show that the ship course can be properly controlled in case of the disturbances of wave, wind, current and error in measure apparatus exist. It is demonstrated that the proposed algorithm is a promising alternative to conventional autopilots.

ACKNOWLEDGMENTS

This work was supported in part by the Ministry of Communications of P.R.C. under grant (95-05-05-32), and in part by the Foundation of The State Key Lab. of Intelligent Technology and Systems, Tsinghua University, China, under grant (0107).

REFERENCES

Brartke. S.J. and Barto.(1996) A. Linear least-squares algorithms for temporal difference learning. *Machine Learning*, **22**, 33-57

Sutton R, Rdserts G N and Taylor D. H.(1997). Tuning fuzzy ship autopilots using artificial neural networks. *Trans.Inst. Meas.Contr*, **19(2)**:94 -106

Sutton R and Marsden G D. (1997).Fuzzy autopilot optimized using a genetic algorithm. *J. Navigation,* **50(1)**:120-131

Sutton R.S.(1988). Learning to predict by the methods of temporal differences, *Mach. Learn*, Vol.**3**,pp.9~44

Sutton R.S.(1984). Temporal credit assignment in reinforcement learning. *PhD thesis*, University of Massachusetts, Amherst, MA

Wakami N, Nomura H and Araki S. (1996).Fuzzy logic for home appliance. *Fuzzy logic and Neural Network Handbook*. McGraw-Hill Inc.

Yang Guoxun.(2002). Study on ship motion hybrid intelligent control and its interacting simulation based on virtual reality. *PhD thesis*, Dalian Maritime University ,China.

Zhou Changjiu.(1997). Hybrid fuzzy intelligent control and its application to double legs robot. *PhD thesis*, Dalian Maritime University ,China.

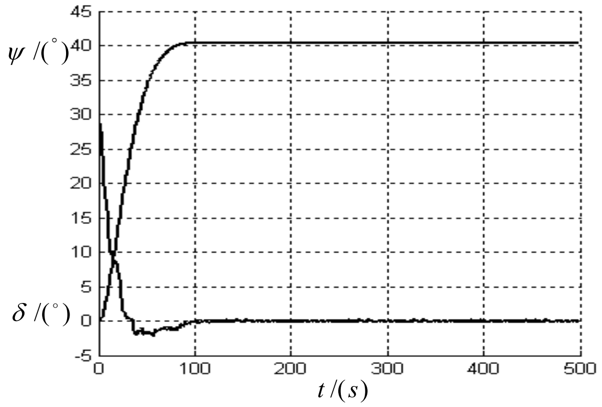

Fig.3 control curve ,course 40°

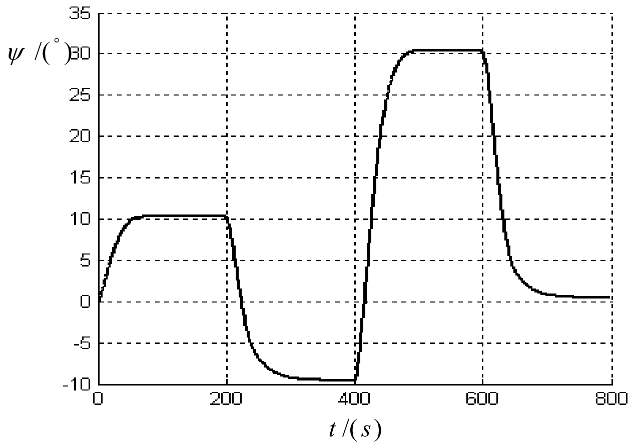

Fig.4 control curve ,course 10°~ -10° ~ 30° ~ 0°

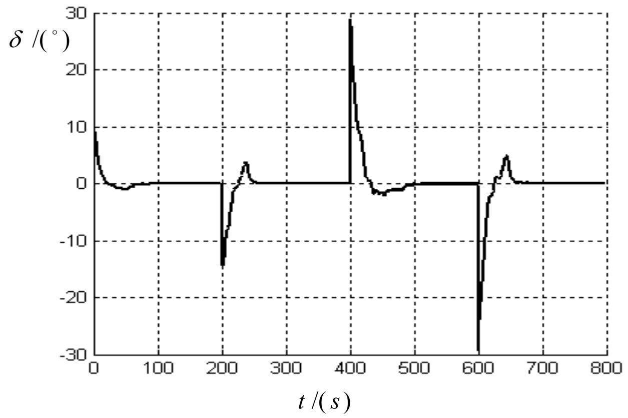

Fig.5 control curve ,course 10°~ -10° ~ 30° ~ 0°

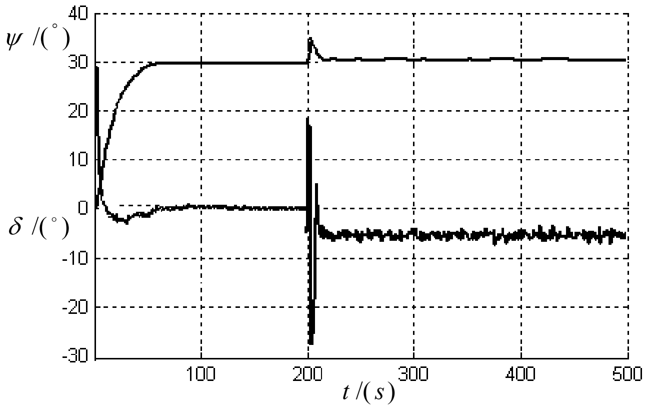

Fig.6 control curve , constant disturbance