Proceedings of 2017 IEEE
International Conference on Mechatronics and Automation
August 6 - 9, Takamatsu, Japan

Design and Implementation of Self-tuning Control Method for the Underwater Spherical Robot

Yanlin He[1, 2], Shuxiang Guo[1, 2, 3*], Liwei Shi[1, 2*], Huiming Xing[1], Zhan Chen[1], Shuxiang Su[1]
[1] Key Laboratory of Convergence Medical Engineering System and Healthcare Technology,
the Ministry of Industry and Information Technology, Beijing Institute of Technology,
No.5, Zhongguancun South Street, Haidian District, Beijing 100081
[2] Key Laboratory of Biomimetic Robots and Systems, Ministry of Education, Beijing Institute of Technology, No.5,
Zhongguancun South Street, Haidian District, 100081 Beijing, China.
[3] Faculty of Engineering, Kagawa University, 2217-20 Hayashi-cho, Takamatsu, Kagawa 760-8521, Japan
Email: heyanlin@bit.edu.cn, guoshuxiang@bit.edu.cn, shiliwei@bit.edu.cn
* Corresponding author

***Abstract*-Considering the complicated disturbance in underwater circumstance, usually it is difficult to solve the control problem when the robot changes its motion state or it is subject to ocean currents, its performance deteriorates since the fixed set of parameters is no longer valid for the new conditions. Thus, in this paper, an auto-tune PID (Proportional + Integral + Derivative)-like controller based on Neural Networks is applied to our amphibious spherical underwater robot, which has a great advantage on processing online for the robot due to their nonlinear dynamics. The Neural Networks (NN) plays the role of automatically estimating the suitable set of PID gains that achieves stability of the system. The NN adjusts online the controller gains that attain the smaller position tracking error. The performance of the NN-based controller is investigated in ADAMS and MATLAB cooperative simulation. The velocity of the spherical robot can be controlled to precisely track desired trajectory in body-fixed coordinate system. Additionally, real time experiments on our underwater spherical robot are conducted to show the effectiveness of the algorithm.**

***Index Terms* - Underwater Spherical Robot; Virtual Prototype; Neutral Network PID Control; Auto-tuning; Cooperative Simulation**

I. INTRODUCTION

Underwater robots have been widely used in many subsea tasks, ranging from ocean inspection to repair of underwater structures related mainly to the power and oil industry. Very often, according to the task, the underwater robot is required to continuously change its operating tool or to pick up and release loads causing a change in behaviour [1]-[10]. That results as an inherent change in its weight, buoyancy and hydrodynamic forces; and as a consequence, a decrease in the position tracking performance. In addition, underwater robots have to deal with the highly dynamical underwater environment represented in the form of ocean currents and waves in shallow water. With this in mind, when the dynamic characteristics of the system are time dependent or the operating conditions of the system vary, it is necessary to re-tune the gains to obtain the desired performance, resulting in time consumption.

As a traditional method, conventional PID controller is simple and practical, but has the disadvantage of the difficulty of adjusting parameter online [11]. NN control theory has been widely used in underwater control system. A 3-layer NN controller has been constructed for underwater robot by Yuh [12], and based on this, Lorenz [13] does research upon an underwater robot of moving up and down. Neutral network control has the strong ability of information integration and complex system control. But neutral network also has some weakness which restricts its development. For instance, it has a slow convergence rate and a long training time, which couldn't be accepted by most control system. Also, traditional neutral network doesn't satisfy the performance index of a control system, including fast response, less overshoot and so on.

Consequently, a variety of motion control methods have been proposed and the intelligent control techniques include PID control, fuzzy control, adaptive control, neural network control, or a mix of them. Research [14]-[16] present systems with a mix of neural networks and fuzzy control in which the training and rules of behaviour are based on the desired states. Their performance is described as accurate when uncertainty and perturbations take place while performing a trajectory. Although the training periods are extremely long, there are also combinations of PID controls and a smart system aimed to auto-tune the gains of different systems such as [17]-[22].

In this paper, an auto-tune PID-like controller based on an online NN is implemented on our spherical underwater robot; for trajectory tracking with unknown disturbances. Simulation results are given considering the non-linear hydrodynamics of robot; including disturbances of ocean currents. Real time experiments on spherical underwater robot are conducted to show the effectiveness of the proposed scheme. For the remaining sections of this paper in Section 2 the general system model of underwater robot and the effect of ocean currents are presented, Section 3 presents the neural network-based self-tuning PID control, Section 4 describes the co-simulation based on ADAMS and MATLAB and experimental results; Finally in Section 5 the conclusions and future works are provided.

II. MODELING OF THE SPHERICAL UNDERWATER ROBOT

As introduced in references [1]-[6], we developed an amphibious spherical robot capable of motion on land and underwater to perform complicated operations. The underwater movement mechanisms of this amphibious spherical robot are the same as those presented in previous reports, by changing the directions and propulsive forces of its four water-jet propellers, the robot can not only move forward or backward but can also rotate clockwise or counter clockwise, with the ability to ascend, dive, or float in the underwater environment.

A. *Kinematic model*

We define an inertial coordinate frame with respect to the ground, denoted *XYZ*, to facilitate analysis of the spherical robot on the horizontal plane, as shown in Fig.1. The origin is at point O. The body coordinate axes *XsYsZs* are parallel to *XYZ*, respectively. The axes are attached to the sphere, with their origin at the center. The set of generalized coordinates describing the sphere consists of 1) the coordinates of the contact point O on the plane, and 2) any set of variables describing the orientation of the sphere [23]-[27].

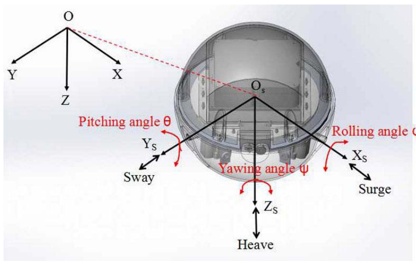


Fig.1 Coordinates setup of the amphibious spherical robot

Let ξ_0, η_0, and ζ_0 be the three projections of the body center point O_s onto the inertial coordinate frame and the vector components of the inertial coordinate frame. The position and orientation of the underwater spherical robot vector in earth fixed coordinate can be expressed by vectors $\boldsymbol{\eta_1}=(x,y,z)$ and $\boldsymbol{\eta_2}=(\varphi,\theta,\psi)$□respectively, where *x*, *y* and *z* represent the Cartesian position in the Earth-fixed frame and and φ represents the roll angle, θ the pitch angle and ψ the yaw angle. At the same time, the linear velocity and angular velocity of the underwater spherical robot in robot-fixed coordinate system can be expressed as $\boldsymbol{v_1}=(u,v,w)$ and $\boldsymbol{v_2}=(p,q,r)$ respectively. In the inertial coordinate frame {O}, $\boldsymbol{v_1}$ is the velocity of the robot, $\boldsymbol{\omega}$ is the angular velocity of robot, the projections of $\boldsymbol{v_1}$ onto the three axes of the body coordinate system {O} are μ, υ, and ω, respectively, and the projections of $\boldsymbol{v_2}$ onto the three axes of the body coordinate system {O} are *p*, *q*, and *r,* respectively.

The relationship between velocities on the fixed and Equations are [28]

$$\begin{bmatrix} \dot{\eta}_1 \\ \dot{\eta}_2 \end{bmatrix} = \begin{bmatrix} J_1(\eta_2) & 0 \\ 0 & J_2(\eta_2) \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \tag{1}$$

where $J_1(\eta_2)$ is the transformation matrix which related to the function of Euler angles, and can be expressed as [30].

$$J_1(\eta_2) = \begin{pmatrix} c\psi c\theta & c\psi s\theta s\varphi - s\psi c\varphi & c\psi s\theta c\varphi + s\psi c\varphi \\ s\psi c\theta & s\psi s\theta s\varphi - c\psi c\varphi & s\psi s\theta c\varphi - c\psi s\varphi \\ -s\theta & c\theta s\varphi & c\theta c\varphi \end{pmatrix} \tag{2}$$

where $s(\cdot) = sin(\cdot)$, $c(\cdot) = cos(\cdot)$, and $J_2(\eta_2)$ can be expressed as:

$$J_2(\eta_2) = \begin{bmatrix} 1 & \sin\varphi\tan\theta & \cos\varphi\tan\theta \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi/\cos\theta & \cos\varphi/\cos\theta \end{bmatrix} \tag{3}$$

The components of the driving forces and diving torque in the body coordinate system {O} are F_{Tx}, F_{Ty}, and F_{Tz} and M_{Tx}, M_{Ty}, and M_{Tz}, respectively. The external resultant force and external resultant torque of the robot when underwater are given below:

$$F = F_d + B + G + \sum_{i=1}^{n} F_{Ti} \tag{4}$$

$$M = M_d + M_B + M_G + \sum_{i=1}^{n} M_{Ti} \tag{5}$$

where F_d is the drag force of water, *B* is the buoyancy force, *G* is force due to gravity, F_{Ti} is the actuation force of the water-jet propeller, and the *M* terms are the corresponding moments.

B. *Hydrodynamic Model*

Hydrodynamic forces and moments are one of the main causes of damping, and equations of motion expressed on the Equation [8]

$$M\dot{v} + C(v)v + D(v)v + G(\eta) = \tau \tag{6}$$

$$\dot{\eta} = J(\eta)v \tag{7}$$

where *M* denotes the inertial matrix (including the added mass), *C* is the Coriolis matrix and centripetal forces (including the effects of added mass), *D* refers to the damping matrix, *G* represents the vector of gravitational forces, and τ is the input control vector.

C. *Ocean currents*

Ocean current is generated by wind, tides, variation of densities and re-circulation of water, among others. The main objective of this work is not to generate a detailed report of this phenomena; nevertheless, it is appropriate to highlight the model of induced ocean currents. In the previous work, the equations of motion are represented in terms of relative velocity of the vehicle and the currents,

$$v_r = v - v_{CI} \tag{8}$$

where $v_{CI}=[u_c\ v_c\ w_c\ 0\ 0\ 0]^T$ is a non-rotational vector of the current velocity according to Equation (1). Note that the linear velocity on the fixed frame can be transformed to linear velocity in the equation by applying the elemental rotation matrices. Let $[u_c^E \quad v_c^E \quad w_c^E]^T$ be the current velocity referenced to the Earth-fixed frame. Then the components of the linear velocity on the equation are calculated as follows,

$$\begin{bmatrix} u_c \\ v_c \\ w_c \end{bmatrix} = J_1^T(\eta_2) \begin{bmatrix} u_c^E \\ v_c^E \\ w_c^E \end{bmatrix} \tag{9}$$

Suppose the current velocity in the equation as constant or at least with a minimum variation, so that:

$$\dot{v}_{CI} = 0 \rightarrow \dot{v}_r = \dot{v} \tag{10}$$

Then, the relative equations of motion become:

$$M\dot{v}+C(v_r)v_r+D(v_r)v_r+G(\eta)=\tau \tag{11}$$

Now, the current velocity in the Earth-fixed frame $[u_c^E \quad v_c^E \quad w_c^E]^T$ can be related to the mean velocity of the current V_C through two angles: α (angle of attack), β (sideslip angle), describing the orientation of v_{CI} around the axes y and z respectively as follows:

$$u_c^E = Vc\cos(\alpha)\cos(\beta) \tag{12}$$

$$v_c^E = Vc\sin(\beta) \tag{13}$$

$$\omega_c^E = Vc\sin(\alpha)\cos(\beta) \tag{14}$$

where Vc is the average currents velocity in the earth-fixed reference frame.

III. Self-Tuning Neural Network for PID Control

The tuning of PID (Proportional + Integral + Derivative) controllers depends on adjusting its parameters (K_p, K_i, K_d), so that the performance of the system under control becomes robust and accurate according to the established performance criteria. The proposed auto-tuning algorithm is based on NN which exhibit the characteristics such as parallelism and generalization, non-linearity, adaptability and fault tolerance.

A block diagram of the auto-tuning control with artificial neural network (NN) is shown in Fig.2 [28].

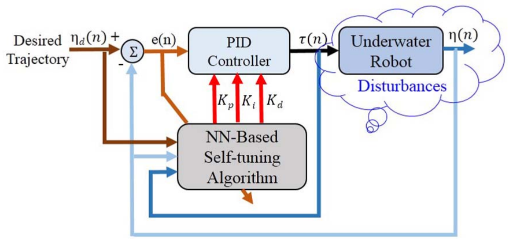


Fig.2 Block diagram of an auto-tuned PID control

The algorithm used as auto-tuning is the back propagation method, chosen for its ability to adapt to changing environments. Operation begins applying the inputs to the network (see Fig.3), this is propagated from the first layer to the hidden layers in, up to produce an output (K_p, K_i and K_d). The output signal is compared to the desired output and an error signal is calculated for each of the outputs, this is shown in Fig. 2. The error outputs back propagate, starting from the output layer, to all neurons in the hidden layer that contribute directly to the output; however, the hidden layer neurons receive only a fraction of the total error signal. This process repeats iteratively, layer by layer, until all neurons in the network has received an error signal describing its relative contribution to the total error.

Figure 3 presents the topology of the NN used to auto-tune the PID control gains implemented on the ROV. Its structure shows seven neurons on the input layer, three neurons on the hidden layer, and finally another three neurons on the output layer. The neurons placed on the output layer correspond to the PID gains: K_p, K_i, K_d.

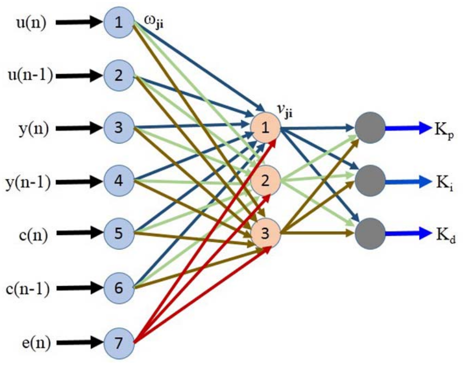


Fig.3 Block diagram of the implemented back propagation NN

where *u(n)* and *u(n-1)* are reference inputs (desired trajectory), *y(n)* and *y(n-1)* are reference outputs (real trajectory), *c(n)* and *c(n-1)* correspond to the control signals, ω_{ji} are the weights of the hidden layer, and v_{ji} are the weights of the output layer.

The back-propagation algorithm looks for the minimum of the error function in weight space using the method of gradient descent. The combination of weights which minimizes the error function is considered to be a solution of the learning problem, the output layer neuron value will be expressed as:

$$u(n)=1/1+e^{-r} \tag{15}$$

where $r=\sum_{j=`}^{3} v_j h_j,\quad h_j=1/1+e^{-S_j},\quad S_j=\sum_{i=1}^{3} w_{ji}x_i$

The criteria used to minimize the error as following [27]:

$$E(n)=\frac{1}{2}\sum_{k=1}^{t}\left(y_r(n)-y(n)\right)^2 \tag{16}$$

The minimization procedure consists, as it is known, in a movement in the negative gradient direction of the function *E(n)* with respect to the weighting coefficients v_{ji} and ω_{ji}. The weighting coefficients of the input layer are

$$\frac{\partial E(n)}{\partial v_{ji}}=\delta^1 h_j \frac{\partial e_y}{\partial e_u} \tag{17}$$

The weighting coefficient of the hidden layer are

$$\frac{\partial E(n)}{\partial w_{ji}}=-\delta_j^2 x_i \frac{\partial e_y}{\partial e_u} \tag{18}$$

Using Equations (17) and (18), the adjustments of weighting coefficients v_{ji} (Equation (19)), ω_{ji} (Equation (20)) can be made by means of the expressions:

$$v_{ji}(n+1)=v_{ji}(n)+\left(a\frac{\partial e_y}{\partial e_u}\right)\delta^1 h_j \tag{19}$$

$$w_{ji}(n+1)=w_{ji}(n)+\left(a\frac{\partial e_y}{\partial e_u}\right)\delta_j^2 x_j \tag{20}$$

where a is the learning coefficient, ω_{ji} *(n+1)* is a vector of weights for the hidden layer, v_{ji} *(n+1)* is the vector of weights of the output layer and equivalent gain $\partial e_y/\partial e_u$ is unknown.

IV. Numerical Simulation and Experimental Results

A. Co-Simulation and Analysis of Underwater Motion

The auto-tuned PID was evaluated using ADAMS and MATLAB/SIMULINK software, firstly, three-dimensional model of the robot was imported from SolidWorks, then applied the mathematical model of robot, next the force model and underwater motion of the robot was set up in ADAMS, and then the integrated co-simulation system was constructed in MATLAB/SIMULINK. Fig.4 showed the construction of the co-simulation system. In this system MATLAB software solved the control strategy like PID or auto-PID, while ADAMS software provided the model of dynamics of mechanical system.

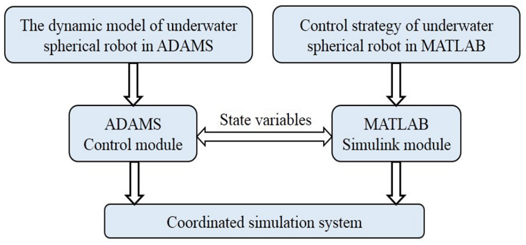
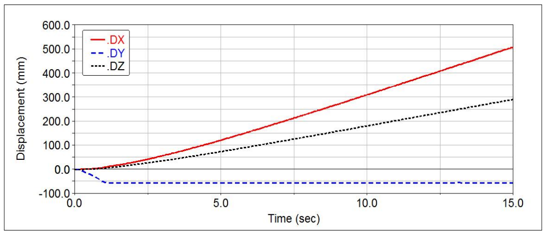
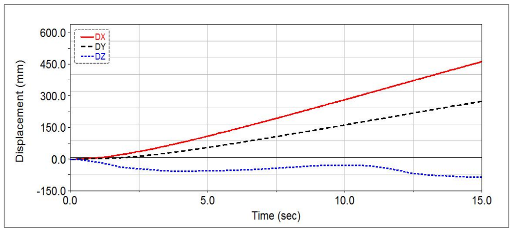

The dynamic model of underwater spherical robot in ADAMS
Control strategy of underwater spherical robot in MATLAB
ADAMS Control module
State variables
MATLAB Simulink module
Coordinated simulation system

Fig.4 The construction of co-simulation system

We assume that the amphibious spherical robot is moving in water, perturbed by water currents of considerable intensity. The first perturbation takes place and its magnitude is $Vc = 0.001$ m/s with orientation of $a = 0$ and $b = 0$, and the second perturbation goes with a magnitude of $Vc = 0.001$ m/s and an orientation of $a = 0$ and $b = pi/2$, as set in Equation (15)-(17) We take the origin of the virtual prototype vector coordinate system as the origin of the ground coordinate system (0, 0, 0), select the point (500, 300, -80) as the target point of the robot motion. By trial and error method, three parameters of the PID control are set up to make the algorithm converge, combining MATLAB and ADAMS simulation at the same time, we observe the motion of robot in real-time. Then through mathematical operation, the moving trajectory of the centroid of robot in different directions are shown in Fig.5 (a), and the attitude angle in three different directions of robot are shown in Fig.6 (a). In order to demonstrate the effectiveness of the designed neutral network-based self-tuning PID control method, we also simulated the underwater motion with traditional PID control, the moving trajectory results and the attitude angle results are shown in Fig. 6 (b) and Fig. 7 (b).

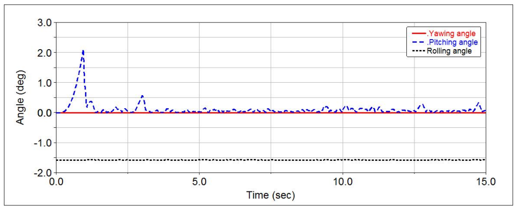


(a) NN-Based Self-Tuning PID control

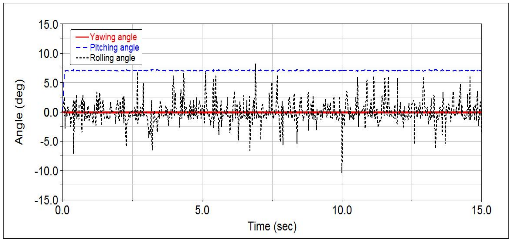


(b) PID control

Fig.5 Displacement curve of the centroid in different directions



(a) NN-based self-tuning PID control



(b) PID control

Fig.6 Vibration amplitude during underwater motion

By obtaining the displacement and time curve of centroid of robot, we measured the final location coordinate of the centroid of robot is (501, 295, -78) when applied neutral network-based self-tuning PID control, the motion errors of robot in three directions about 0.02%, 1.67% and 2.5%, respectively. However, when applied traditional PID control, the final location coordinate of the centroid of robot is (495, 290, -74.6), the motion errors of robot in three directions about 1.0%, 3.33% and 6.7%, respectively. So the NN-based self-tuning PID has better performance than the traditional PID, especially for vertical motion. For the attitude angle of robot, the maximum yawing angle of the NN-based self-tuning PID is less than one of the traditional PID. Additionally, the variety of the pitching angle and rolling angle of NN-based self-tuning PID control is less than the results with traditional PID control. For these reasons, it is feasible to conclude that the NN-based self-tuning PID control has better performance, and the NN-based self-tuning PID control algorithm could satisfy the requirement of underwater movement in precision.

The next set of Fig.7 indicate the gains (Kp, Ki, Kd) obtained by the neural network in the direction of Z. As can be see, neurons start working from time zero since they absorbed the first perturbation in length, and respond to the abrupt change presented when the second perturbation is introduced, thus allowing change of the neural network in order to compensate for the lack of gains in the DOF corresponding to the alterations.

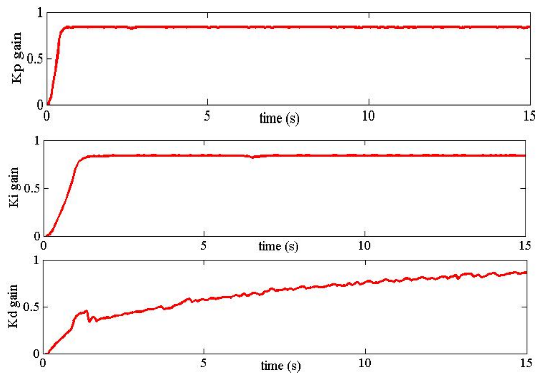
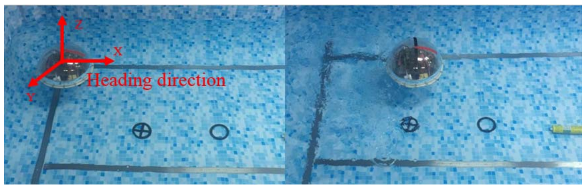


Fig.7 PID gain time behavior in z-coordinate

B. *Experimental Results and Analysis of Underwater Motion*

To evaluate the improved amphibious spherical robot, we carried out several underwater experiments by using the NN-based self-tuning PID control method with and without disturbances in a pool, which was also done using the traditional PID control method. These experiments were carried out in a same pool and each one was repeated twenty times. Control signals directed the robot along the desired trajectory, and the robot was programmed to move forward in a distance of 3 m in a pool. The displacement distance and the attitude angle were measured by a ruler and an IMU sensor. Fig.8 shows a video sequence of the horizontal forward underwater motion without disturbances, in which one pair of water-jet propellers provided the same propulsive forces, and the yawing angle was used as feedback data for the control algorithm. We recorded the time and the displacement in the forward motion experiment, and calculated the average velocity. Fig.9 and Fig.10 shows the trajectory tracking result and yawing angle of NN-Based Self-Tuning PID controller vs PID without disturbances. Fig.11 and Fig.12 shows the trajectory tracking result and yawing angle of NN-Based Self-Tuning PID controller vs PID with disturbances.



(a) At 0s (b) At 5s

Fig. 8 Underwater horizontal forward motion

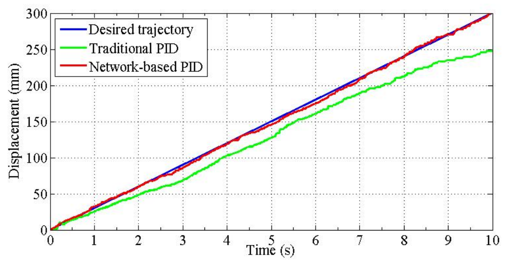


Fig.9 Trajectory tracking result of NN-based self-tuning PID control vs PID (without disturbances)

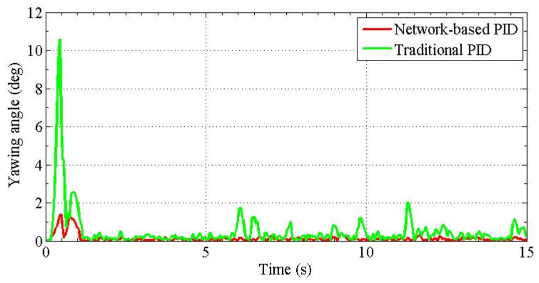


Fig.10 Experimental results of yawing angle of NN-based self-tuning PID control vs PID (without disturbances)

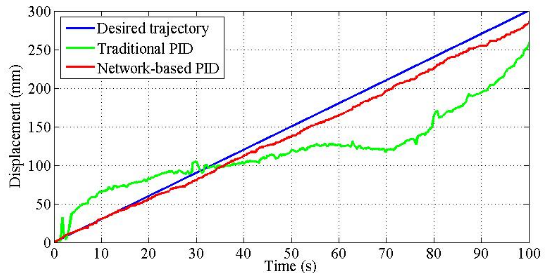


Fig.11 Trajectory tracking results of NN-based self-tuning PID control vs PID (with disturbances)

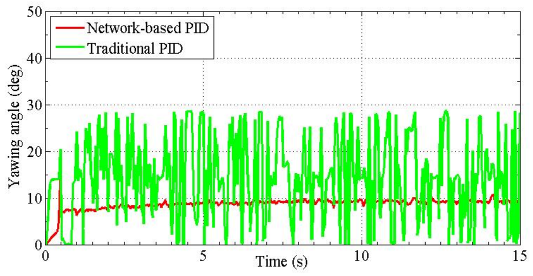


Fig.12 Experimental results of yawing angle of NN-based self-tuning PID vs PID (with disturbances)

As we can see from Figures 9 and 11, when the robot complete underwater horizontal forward motion without disturbances, the final trajectory were basically along the initial setting with NN-based self-tuning PID control method and the traditional PID control method, and the yawing angle of robot within the scope of our requirements. However, when the robot complete underwater horizontal forward motion with disturbances, we compared the experimental results with NN-based self-tuning PID and the traditional PID, as shown in Figures 10 and 12, of which the maximum error of trajectory tracking was about 70cm for traditional PID. For the NN-based self-tuning PID control method, we could see that the position error was greatly reduced and the maximum error was about 30cm. The maximum error of yawing angle was about 5.2deg with traditional PID control method, however, for the NN-based self-tuning PID control method, the position error was greatly reduced and the maximum error was about 2.7deg. Consequently, the NN-based self-tuning PID was more helpful in improving the control accuracy of robot when swimming underwater with some disturbances.

V. Conclusions and Future Work

The actual work presents the development of a control algorithm to automatically tune the gains of a PID control, based on a neural network. The control algorithm was implemented on our amphibious spherical robot for trajectory tracking with unknown disturbances. The algorithm performance was evaluated in two instances: a numerical simulation and implemented on the underwater spherical robot in real-time. The numerical simulation took place with the non-linear hydrodynamics of robot; including disturbances of ocean currents in different directions. In order to validate in real-time the auto-tuned PID control. A comparative study between the conventional PID and the auto-tuned PID was discussed. The study took into consideration two criterions to assess the performance of each controller: position tracking error and yawing angle, leading to the conclusion that the proposed NN-based PID controller attained the best performance with less trajectory tracking error and yawing angle. So the neutral network based PID controller has better performance of underwater spherical robot, especially when the robot swimming in underwater with some disturbances. Future research will involve some more complicated underwater experiments with different frequencies and velocities.

ACKNOWLEDGEMENTS

This work was supported by National Natural Science Foundation of China (61503028, 61375094), and Excellent Young Scholars Research Fund of Beijing Institute of Technology (2014YG1611). This research project was also partly supported by National High Tech. Research and Development Program of China (No.2015AA043202).

REFERENCES

[1] M. Li, S. Guo, H. Hirata, H. Ishihara, "Design and performance evaluation of an amphibious spherical robot," *Robotics and Autonomous Systems*, vol. 64, pp. 21-34, 2015.

[2] S. Guo, Y. He, L. Shi, S. Pan, R. Xiao, P. Guo, "Modal and fatigue analysis of critical components of an amphibious spherical robot," *Microsystem Technologies*, pp. 1-15, doi: 10.1007/s00542-016-3083-0, 2016.

[3] Y. He, L. Shi, S. Guo, S. Pan, Z. Wang, "Preliminary mechanical analysis of an improved amphibious spherical father robot," *Microsystem Technologies*, pp. 1-16, doi: 10.1007/s00542-015-2504-9, 2015.

[4] Y. He, L. Shi, S. Guo, S. Pan, Z, Wang, "3D Printing Technology-based an Amphibious Spherical Underwater Robot", *Proceedings of 2014 IEEE International Conference on Mechatronics and Automation,* pp. 1382-1387, 2014.

[5] L. Shi, S. Guo, S. Mao, C. Yue, M. Li, K. Asaka, "Development of an Amphibious Turtle-Inspired Spherical Mother Robot," *Journal of Bionic Engineering*, vol.10, no.4, pp. 446-455, 2013.

[6] S. Pan, S. Guo, L. Shi, Y. He, Z, Wang, Q, Huang, "A spherical robot based on all programmable SoC and 3-D printing," *Proceedings of 2014 IEEE International Conference on Mechatronics and Automation*, pp. 150-155, 2014.

[7] Y. Li, S. Guo, C. Yue, "Preliminary Concept of a Novel Spherical Underwater Robot", *International Journal of Mechatronics and Automation*,vol.5, no.1, pp. 11-21, 2015.

[8] M. Li, S. Guo, H. Hirata, H. Ishihara, "A roller-skating/walking mode-based amphibious robot," *Robotics and Computer-Integrated Manufacturing*, vol. 44, pp. 17-29, 2017.

[9] C. Yue, S. Guo, L. Shi, "Design and Performance Evaluation of a Biomimetic Microrobot for the Father-son Underwater Intervention Robotic System", *Microsystem Technologies*, vol.22, no.4, pp. 831-840, 2015.

[10] J. Yuh, "A neural net controller for underwater robotic vehicles". *Oceanic Engineering*, vol. 15, no.3, pp. 161-166, 1990

[11] J. Lorentz and J. Yuh, "A survey and experimental study of neural network AUV control", *Proceedings of the 1996 IEEE Symposium on Autonomous Underwater Vehicle Technology,* pp.109-116, 1996.

[12] X. Song, F. Liu, Z. Zou, Y.M. Zhu, J.Yin, F. Xu, "Nonlinear Underwater Robot Controller Design With Adaptive Disturbance Prediction and Smoother", *Journal of Computational Intelligence Systems*, vol. 4, no.4, pp. 634-643, 2012.

[13] A. Marzbanrad, M. Eghtesad, R. Kamali, "A Robust Adaptive Fuzzy Sliding Mode Controller For Trajectory Tracking of ROVs", *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 2863-2870, 2011.

[14] N. Kha, K.K. Ahn, "Position Control of Shape Memory Alloy Actuators by Using Self Tuning Fuzzy PID Controller", *International Journal of Control Automation and Systems*, vol.4, no.6, pp. 756-762, 2006.

[15] J. Paulusova, L. Orlicky, M.Dubravska, "Self-tuning fuzzy PID controller", *Proceedings of the International Conference on Process Control*, pp. 304–308, 2013.

[16] E. Dong, S. Guo, X.Lin,; Li, X. Y.Wang, "A Neural Network-Based Self-Tuning PID Controller of an Autonomous Underwater Vehicle", *Proceedings of 2012 IEEE International Conference on Mechatronics and Automation*, pp. 898–903, 2012.

[17] L. Liu, J. Luo, "Research of PID Control Algorithm Based on Neural Network", *Energy Procedia*, vol.13, pp. 6988-6993, 2011.

[18] K. Sinthipsomboon, I. Hunsacharoonroj, J. Khedari,; W. Pongaen, P. Pratumsuwan, "A hybrid of fuzzy and fuzzy self-tuning PID controller for servo electro-hydraulic system", *Proceedings of the 6th IEEE Conference on Industrial Electronics and Applications*, pp. 220-225, 2011.

[19] T. Ai, J. Yu, Y. Liu, J. Zhou, "Study on Neural Network Self-Tuning PID Control for Temperature of Active Solar House Heating System", *Proceedings of the International Workshop on Intelligent Systems and Applications*, pp. 1-4, 2010.

[20] M. Sasaki, A. Asai, T. Shimizu, S.Ito, "Self-tuning control of a two-link flexible manipulator using neural networks", *Proceedings of the ICCAS-SICE*, pp. 2468-2473, 2009.

[21] S. Guo, J. Du, X. Lin, and C. Yue, "Adaptive fuzzy sliding mode control for spherical underwater robots", *Proceedings of 2012 IEEE International Conference on Mechatronics and Automation*, pp. 1681-1685, 2012.

[22] T. Kuo, Y. Huang, C. Chen, C. Chang, "Adaptive Sliding Mode Control with PID Tuning for Uncertain Systems", *Engineering Letters*, vol.16, no.3, pp. 311-315, 2008.

[23] C. Karakuzu, "Parameter Tuning of Fuzzy Sliding Mode Controller Using Particle Swarm Optimization", *International Journal of Innovative Computing Information & Control,*vol.6, no.10, pp.4755-4770, 2010.

[24] A.N. Ponce, A. Behar, A. Hernández, V. Sitar, "Neural Networks for Self-Tuning Control Systems", *Acta Polytechnic*a, vol.44, no.1, pp.49-52, 2004.

[25] Y. Xu, C. Xie, and D. Tong, "Adaptive synchronization for dynamical networks of neutral type with time-delay", *Optik-International Journal for Light and Electron Optics*, vol. 125, no.1, pp. 380-385, 2014.

[26] A.M. Lekkas, T.I. Fossen, "Trajectory Tracking and Ocean Current Estimation for Marine Underactuated Vehicles", *Proceedings of 2014 IEEE International Conference on Control Applications*, pp. 905-910, 2014.

[27] E. Dong, Z. Chen, Z. Yuan., "Control and synchronization of chaos systems based on neural network PID controller", *Journal of Engineering and Technology*, vol. 37, no. 3, pp.646-650, 2007.

[28] H. A. Rodrigo, G. V. L. Govinda, S. J. Tomás, ".Neural Network-Based Self-Tuning PID Control for Underwater Vehicles", *Sensors*, vol.16, no.9, 2016.