Particle Swarm Optimization of Neural Controller for Tanker Ship Steering

[1]C. K. Loo, [2]Nikos, E. Mastorakis
[1]Faculty of Engineering and Technology
Multimedia University
Melaka, Malaysia
[2]Department of Electrical Engineering and Computer Science
Hellenic Naval Academy
Piraues, GREECE

http://www1.mmu.edu.my/~ckl
http://wseas.org/mastorakis/

Abstract: - This paper discussed the implementation of Particle Swarm Optimization (PSO) to optimize a Radial Basis Function (RBF) neural controller for tanker ship control. The RBF neural controller uses reinforcement learning strategy to achieve the heading regulation of tanker ship exposed to plant changes and disturbances. However, the tuning of the neural controller design parameters are considered to be difficult and tedious due to the high nonlinearity of the ship dynamic model and the external disturbances. It is shown that PSO can provide a very promising technique for its simplicity and ease of use. Moreover, Centroidal Voronoi Tessellation (CVT) is implemented to select the starting positions of the particles strategically. The promising results from the experiment provide direct evidence for the feasibility and effectiveness of PSO for the optimization of neural controller for tanker ship heading regulation.

Key-Words: - Particle Swarm optimization, autopilots, nonlinear optimization, ship steering, neural controller, radial basis function networks.

1 Introduction

An autopilot is a ship's steering controller, which automatically manipulates the rudder to decrease the error between the reference heading angle and the actual heading angle. To improve fuel efficiency and reduce wear on ship components, autopilot systems have been developed and implemented for controlling the directional heading of ships. Often, the autopilots utilize simple control schemes such as PID control. However, manual adjustments of the PID parameters are required to compensate for disturbances acting upon the ship such as wind and currents. Once the PID parameters are fine-tuned manually, the controller will generally work well for small variations in the operating conditions. For large variations, the parameters of the autopilot must be continually modified. Such continual adjustments are necessary because the dynamics of a ship vary with, for example, speed, trim, and loading. In addition, it is useful to change the autopilot control law parameters when the ship is exposed to large disturbances resulting from changes in the wind, waves, current, and water depth. Manual adjustment of the controller parameters is often a burden on the crew. Moreover, poor adjustment may result from human error. As a result, it is of interest to have a method for automatically adjusting or modifying the underlying controller. In this paper, a Particle Swarm Optimization (PSO) of Radial Basis Function (RBF) neural controller using reinforcement learning strategy [11] is proposed to achieve the heading regulation of tanker ship exposed to plant changes and disturbances. PSO has been shown to be a promising approach for solving both unconstrained and constrained optimization problems [2][3][4][5]. Recently, several heuristics have been developed to improve the performance and set up suitable parameters for the PSO algorithm [2][3]. Some theoretical work to analyze the trajectory of particles has been carried out. A constriction factor has been proposed by Clerc and Kennedy [2] to ensure convergence. Trelea [3] reported on the trajectory analysis using dynamic systems theory. In addition, random starting configurations always lead to additional variability in PSO. The performance of PSO can be improved by strategically selecting the starting positions of the particles. It has been suggested the use of generators from Centroidal Voronoi Tessellation (CVT) as the starting points for the swarm can ensure the broad coverage of the search space and thus the solution space is fully explored for the optimal solution [6]. The same population initialization strategy, CVT will be applied in this paper.

Section II presents the steering dynamics and control of a tanker ship model. In Section III, the Radial Basis Function (RBF) neural controller is explained. Section IV describes the Trelea's PSO model. Centrodail Vornoi Tesselation algorithm is briefly described in Section V. Section VI discusses experimental results comparing the optimized neural controller. Conclusion is presented in Section VII.

2 Tanker Ship Steering

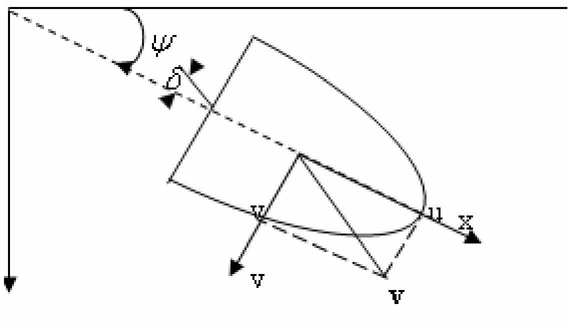


Figure 1. Tanker Ship

Assuming the "bobbing" or "bouncing" effects of the ship is neglected for large vessels, the motion of the ship is described by a coordinate system which is fixed to the ship [10]. Based on Fig. 4, a simple model which describes the dynamical behavior of the ship can be expressed by the following differential equation:

$$\dddot{\psi}(t)+\left(\frac{1}{\tau_1}+\frac{1}{\tau_2}\right)\ddot{\psi}(t)+\left(\frac{1}{\tau_1}\frac{1}{\tau_2}\right)\dot{\psi}(t) = \frac{K}{\tau_1\tau_2}\left(\tau_3\dot{\delta}(t)+\delta(t)\right) \tag{1}$$

where ψ is the heading of the ship and δ is the rudder angle. Assuming zero initial conditions, (1) can be written

$$\frac{\phi(s)}{\delta s}=\frac{K(s\tau_3+1)}{s(s\tau_1+1)(s\tau_2+1)} \tag{2}$$

where K, τ_1, τ_2 and τ_3 are parameters which are a function of the ship's constant forward velocity u and its length l as expressed below:

$$K=K_0\left(\frac{u}{l}\right) \tag{3}$$

$$\tau_i=\tau_{i0}\left(\frac{l}{u}\right), \quad i=1,2,3. \tag{4}$$

We assumes that for a tanker ship under "ballast" conditions (a very heavy ship), $K_0=5.88, \tau_{10}=-16.91, \tau_{20}=0.45, \tau_{30}=1.43,$ and $l=350$ meters [8]. For "full" conditions, $K_0=0.83, \tau_{10}=-2.88, \tau_{20}=0.38, \tau_{30}=1.07$.

The nominal speed of the tanker ship traveling in the x direction is $u=5m/s$. The maximum deviation of the rudder angle is assumed to be ± 80 degrees. This is only valid if the ship make small deviations from a straight line path, $(\delta<5^\circ)$. For $\delta>5^\circ$, an extended model given as follows should be used [8].

$$\dddot{\psi}(t)+\left(\frac{1}{\tau_1}+\frac{1}{\tau_2}\right)\ddot{\psi}(t)+\left(\frac{1}{\tau_1}\frac{1}{\tau_2}\right)H(\dot{\psi}(t)) = \frac{K}{\tau_1\tau_2}\left(\tau_3\dot{\delta}(t)+\delta(t)\right) \tag{5}$$

where $H(\dot{\psi}(t))$ is a nonlinear function of $\dot{\psi}(t)$. The function is approximated as equation (19) under steady state condition, $\dddot{\psi}=\ddot{\psi}=\dot{\delta}=0$.

$$H(\dot{\psi})=a\dot{\psi}^3+b\dot{\psi} \tag{6}$$

where a and b are real valued constants such that a is always positive. They are chosen to be 1 in this paper. For optimization purpose, the following cost function which represents the propulsive energy losses due to steering is defined.

$$J=\frac{1}{T}\left[\sum_{t=1}^{T}\left(e(t)^2+\lambda\delta(t)^2\right)\right] \tag{7}$$

The λ value is set to 0.01 in the simulation.

3 Radial Basis Function Neural Controller and Learning Mechanism Design

From Fig. 2, the inputs of the Radial Basis Function network are $x_1=e,$ and $x_2=c$ the neural controller output is $\delta=F(e,c)$ where F represents the processing by the entire radial basis function neural network. We use the error $e(k)$ and the derivative of $e(k)$ as the inputs to the radial

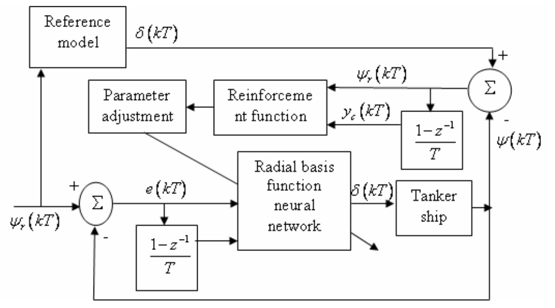


Figure 2. Reinforcement learning and RBF neural controller for ship heading regulation

basis function network and ψ_r, ψ represent the desired plant output and actual plant output.

$$e = \psi_r - \psi$$
$$\dot{e} = \dot{\psi}_r - \dot{\psi} \qquad (8)$$

A backward difference approximation to the derivative is used for $\dot{e}(k)$

$$\dot{e}(k) \approx \frac{e(kT) - e(kT-T)}{T} = c(kT) \qquad (9)$$

where T = 10 sec and k is an index for the time step and T is the sampling period of the digital controller. As is standard in discrete-time systems, we use "k" rather than "kT" as the argument for the signals. Let $\mathbf{x}(k) = [e(k), c(k)]$ as the input vector to the i^{th} receptive field unit and its output denoted with $R_i(\mathbf{x})$. The receptive field unit has a "weight" which we denote by b_i. Assume that there are n_R receptive field units. From Figure,

$$\delta(k) = F(e(k), c(k)) = \sum_{i=1}^{n_R} b_i R_i(\mathbf{x}(k)). \qquad (10)$$

The receptive field units are chosen as

$$R_i(\mathbf{x}) = \exp\left(-\frac{\left|\mathbf{x} - \mathbf{c}^i\right|^2}{\left(\sigma^i\right)^2} \right) \qquad (11)$$

where $\mathbf{c}^\mathbf{i} = \left[c_1^i, c_2^i\right]$, σ^i is a scalar.

In this paper, we use the $n_R = 121$ and create a uniform grid for the $\mathbf{c}^\mathrm{i}$ centers, assume that $e(k) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ and $c(k) \in [-0.01, 0.01]$. For the receptive field units we use spreads $\sigma_1^i = 0.7\frac{\pi}{\sqrt{n_R}}$ and $\sigma_2^i = 0.7\frac{0.02}{\sqrt{n_R}}$. The tuning of the receptive field unit weights $b_i, i = 1, 2, \ldots, n_R$ is considered. The receptive field unit weights are initialized to be zero to represent that the neural network knows little about how to control the ship heading. The weights are adjusted based on reinforcement learning [11]. The controller learns quantifies the relative success or failure of its actions. If the action it took tended to lead it closer to its reference model, then it strengthens the tendency to pick that action again, otherwise, the action selection tendency will be weaken for the unsuccessful case. To define the reinforcement signal, first define a reference model

$$G(s) = \frac{1}{150s + 1} \qquad (12)$$

The reference model indicates a smooth first order response for changes in the desired ship heading. The reference model output $\psi_m(kT)$ is discretized (15) and computed.

We use

$$y_e(kT) = \psi_m(kT) - \psi(kT)$$
$$y_c(kT) = \frac{e(kT) - e(kT-T)}{T} \qquad (13)$$

as inputs to the reinforcement function. Define

$$J_R\left(y_e(kT), y_c(kT)\right) = \begin{cases} \bar{J}_R\left(y_e(kT), y_c(kT)\right) \\ \quad \text{if } \left|\bar{J}_R\left(y_e(kT), y_c(kT)\right)\right| \geq \alpha & (14) \\ 0 & (16) \\ \quad \text{if } \left|\bar{J}_R\left(y_e(kT), y_c(kT)\right)\right| < \alpha \end{cases}$$

where

$$\bar{J}_R\left(y_e(kT), y_c(kT)\right) = \eta\left(-\eta_e y_e(kT) - \eta_c y_c(kT)\right)$$

and $\alpha = 0.005$ is chosen. Here, η, η_e and η_c and (17) considered as design parameters. The parameters η_e and η_c are adjusted to indicate the importance of achieving tracking and deviations in tracking, respectively. The parameter η is the adaptation gain; for small value, adaptation will be slow, for large value may lead to instabilities. The receptive field unit weights will be adjusted according to

$$b_i(kT) = b_i(kT-T) + J_R(kT)R_i(kT-T) \quad (15)$$

It is proposed to use particle swarm optimization (PSO) technique to optimize the three design parameters η, η_e and η_c to improve the performance of adaptive neural control.

4 Particle Swarm Optimization (PSO)

The particle swarm optimization (PSO) is a parallel evolutionary computation technique, originally developed by Kennedy and Eberhart [1][2]. The PSO algorithm is initialized with a population of random candidate solutions, conceptualized as particles. Each particle is assigned a randomized velocity and is iteratively moved through the problem space. Each particle is attracted towards the location of the best fitness so far by the particle itself and by the location of the best fitness achieved so far across the population. The algorithm is described in the following:

(i) Initialize a population of random positions and velocities in the n-dimensional problem space.

(ii) Evaluate the fitness value for each particle.

(iii) Each particle's fitness evaluation is compared with the current particle's pbest. If current value is better than pbest, update the pbest with the current value in n-dimensional space.

(iv) Compare fitness evaluation with the population's global best fitness, gbest, then reset gbest to the current particle's array index and value.

(v) The velocities and positions of the particle are updated as follows:

$$\boldsymbol{v}_i(t+1) = w\boldsymbol{v}_i(t) + c_1 rand_1(\boldsymbol{pbest}_i - \boldsymbol{x}_i(t))$$
$$+ c_2 rand_2(\boldsymbol{gbest} - \boldsymbol{x}_i(t)) \quad (16)$$
$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \boldsymbol{v}_i(t+1)$$

(vi) Loop to (ii) until a stopping criterion is met.

The vector $\boldsymbol{x}_i = [x_{i1}, x_{i2}, x_{i3}, \ldots, x_{in}]^T$ stands for the position of the i^{th} particle, $v_i = [v_{i1}, v_{i2}, v_{i3}, \ldots, v_{in}]^T$ represents the velocity of the ith particle. The variable w is the inertia weight, c_1 and c_2 are positive constants; $rand_1$ and $rand_2$ are uniform random numbers in the range [0; 1], generated anew for each dimension, $i = 1, \ldots, n$ of the particle i. The inertia weight w represents the momentum of the particles. The constants c_1 and c_2 stands for the controlling factors of the "cognition" and "social" parts that attract each particle toward pbest and gbest. The "cognition" factor governs the independent behavior of the particle itself whereas the "social" factor determines the collective behavior of the particles. In the paper [2], Clerc and Kennedy introduced a constriction coefficient k for swarm system stability and convergence. The velocity equation is updated according to

$$\boldsymbol{v}_i(t+1)$$
$$= k\left[\boldsymbol{v}_i(t) + c_1 rand_1(\boldsymbol{pbest}_i - \boldsymbol{x}_i(t)) + c_2 rand_2(\boldsymbol{gbest} - \boldsymbol{x}_i(t))\right] \quad (17)$$

where $k = \dfrac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}$ with $\varphi = c_1 + c_2 > 4$

and k is a function of c_1 and c_2. By default, φ is set to $4.1(c_1 = c_2 = 2.05)$, and the construction coefficient k is 0.729. This is equivalent to $w = 0.729, c_1 = c_2 = 1.494$ in equation (16). In this paper, the PSO is named as Clerc model. Trelea [3] also presented convergence analysis for a one-dimensional particle. The parameter settings reported in his simulations were $w = 0.85, c_1 = c_2 = 1.7$. It is named as Trelea model in this paper.

5 Centroidal Voronoi Tessalations

A group of points in the search space is designed to be the set of solution generators. Particle initialization in PSO can be thought as a process to allocate the solution generators in the search space. The space is partitioned into compartments with particle as their centroid. The particles should be initialized so that they are distributed as evenly as possible throughout the space to ensure broad coverage of the search spaces. The standard method of particle initialization in PSO fails to accomplish this goal, especially in high-dimensional spaces [6]. Centroidal Voronoi Tessellation (CVT) is a way to partition a space into compartments [6]. It has been shown that CVT can initialize the PSO particles

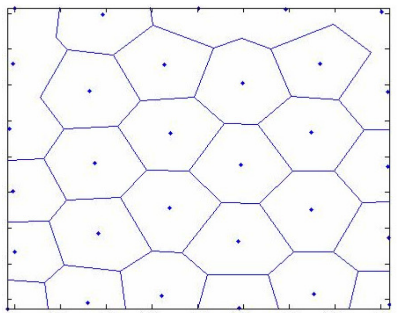


Figure 3. Centroidal Voronoi Tesselations for 30 points

evenly in the solution space and lead to improved performance [6]. Two of the most well-known algorithms for computing CVTs are Mac Queen's method [6] and Lloyd's method [7]. Lloyd's algorithm is deterministic and requires only a few iterations, but each one is computationally expensive. In this paper, Lloyd's algorithm is chosen to compute CVT using the source codes available from the authors [7]. The details of the algorithm are explained in the literature [7]. Figure 3 shows the CVTs for 30 points of generators to be used as initial population of particles in PSO for the experiment to be discussed in this paper.

6 Experimental Results

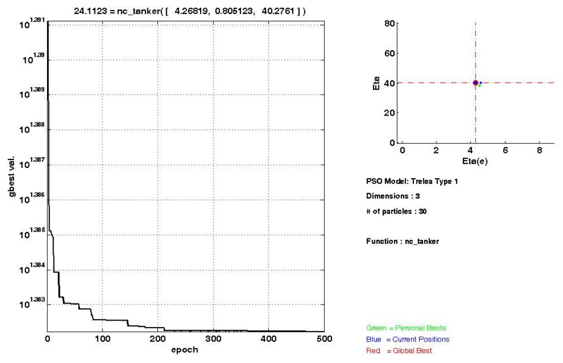


Figure 4. Result for the neural controller optimization of tanker ship steering using Trelea PSO model.

The nonlinear process model given in equation (9) is used to emulate the "real" ship dynamics. The PSO models will perform the neural controller optimization for the three design parameters η, η_e and η_c based on the cost function defined in equation (14). The PSO algorithms were implemented in Matlab [10]. During the numerical experiments, the Trelea PSO model was run with an initial population of particles generated by CVTs shown in Figure 3. All the running trials were carried out with a population of 30 particles and 200 generations. The results of the Trelea PSO model is shown in Figure 4. From the results obtained, the PSO model presents the ability to optimize the neural controller of tanker ship steering. Table 1 summarizes the results.

Table 1 Summary of results

PSO model (Clerc)	η	η_e	η_c	**gbest (J)**
	4.2682	0.8051	40.2761	24.1123

The optimized neural controller is utilized for the tanker ship steering regulation under different operating conditions such as wind disturbances, changes of speed, existence of sensor noise and ship weight changes.

Nominal conditions

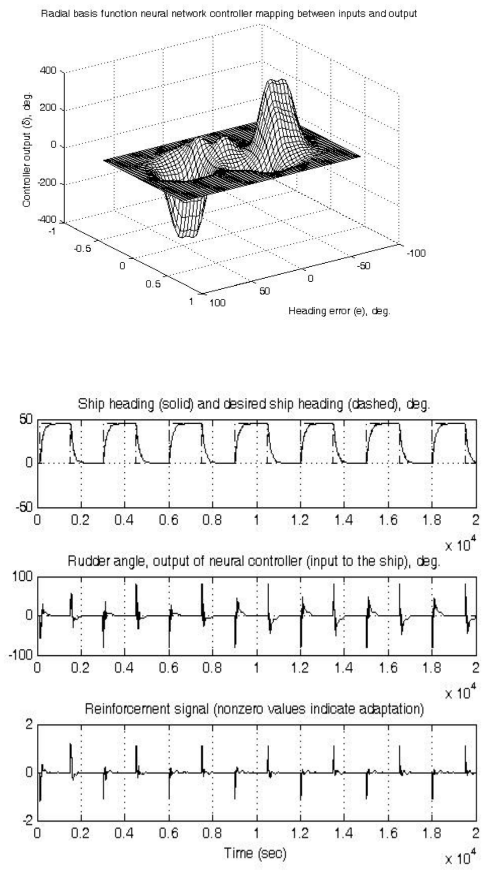


Figure 5. Controller response surface and closed-loop response resulting from using the optimized neural controller for tanker ship steering under nominal condition.

The first simulation study concerns the tanker ship steering control under nominal condition, where we have ballast conditions (heavy weight), no wind, no sensor noise, and a nominal speed of 5 m/s. The close-loop response is shown in Fig. 5. Significant overshoot of the desired response is observed at early stage of simulation. It is due to the lack of initial knowledge about how to control the ship heading. The neural controller perform on-line reinforcement learning effectively and eventually the response is improved, overshoot is reduced until ultimately the tanker can tracks closely the desired heading ψ_m .The learned input-output of controller surface is shown in Fig 5.

Effects of Wind Disturbances

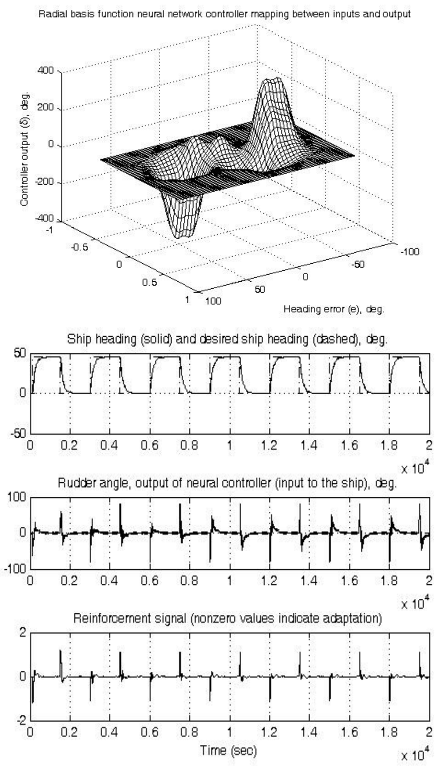


Figure 6. Controller response surface and closed-loop response resulting from using the optimized neural controller for tanker ship steering under wind disturbances.

The wind disturbance on effects on the ship is studied in this experiment. An additional sinusoidal disturbance effect is added to the rudder angle of the tanker ship. In Fig. 6, we found that the neural controller is able to compensate the wind disturbance effects to achieve very good regulation of the ship heading. The neural controller learns how to shape the mapping to reduce the effects of the wind disturbances by on-line reinforcement learning, even though the precise characteristics of the wind disturbance are not known. The final neural controller response surface is shown in Fig. 6.

Effects of Speed Change

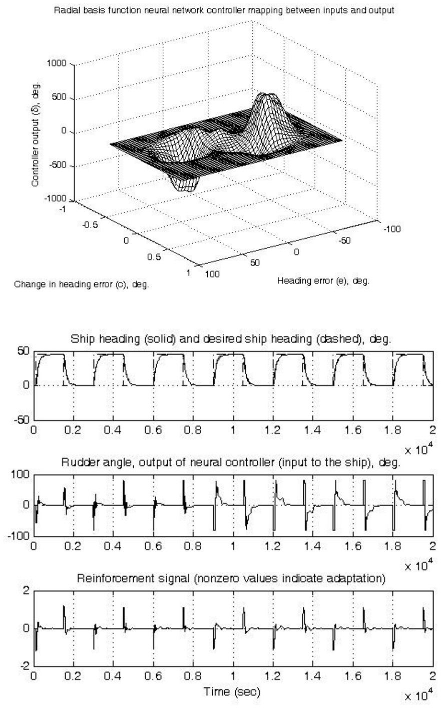
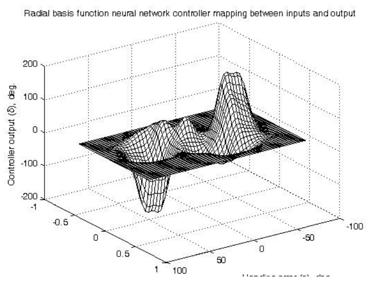


Figure 7. Controller response surface and closed-loop response resulting from using the optimized neural controller for tanker ship steering under speed change.

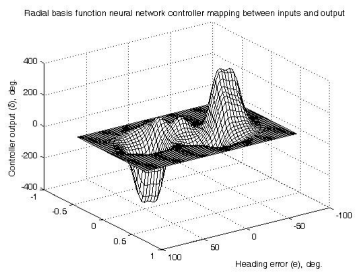
In this experiment, we consider the effect of speed change on the tanker ship steering control. For the first 9000 seconds, the ship operates at a speed of $u = 5\,m/s$. The speed is changed abruptly at $t = 9000$ seconds. The close-loop response and the controller response surface is depicted in Fig 7. The same response as for nominal conditions are observed for the first 9000 seconds. Then, it adapts to the speed change after $t = 9000$ seconds. A slightly degraded transient response is shown but adaptively improved later. The final tuned controller response surface in Fig. 7 is different from Fig. 5 that works under nominal conditions.

Effects of Weight Changes

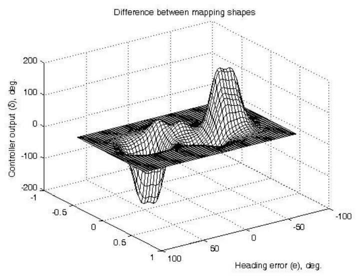
We consider the case of how the ship steers when at $t = 9000$ seconds, we switch from ballast to full conditions. Fig. shows the close-loop response. The neural controller is able to compensate for the



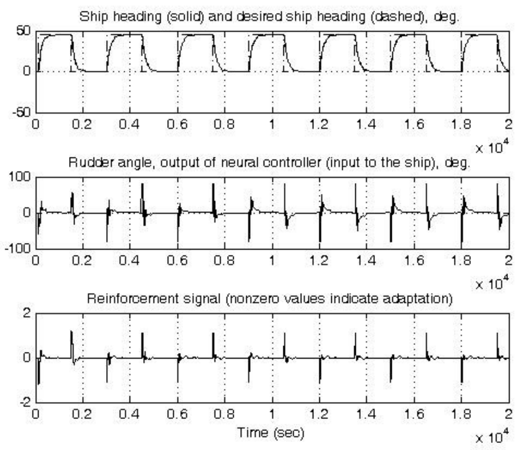
(a)



(b)



(c)



(d)

Figure 8. (a) Controller response surface under ballast condition (b) Controller response surface under full condition (c) Changes of controller response surface after switching and (d) Closed-loop response resulting from using the optimized neural controller for tanker ship.

weight change, and the final tuned controller response surface under full conditions (Fig. 8) is different from the response surface under ballast condition. The difference in the controller mapping is shown in Fig. 8c. The input-output map changes when the controller adapts from a ballast condition to a full one.

Effects of Sensor Noise

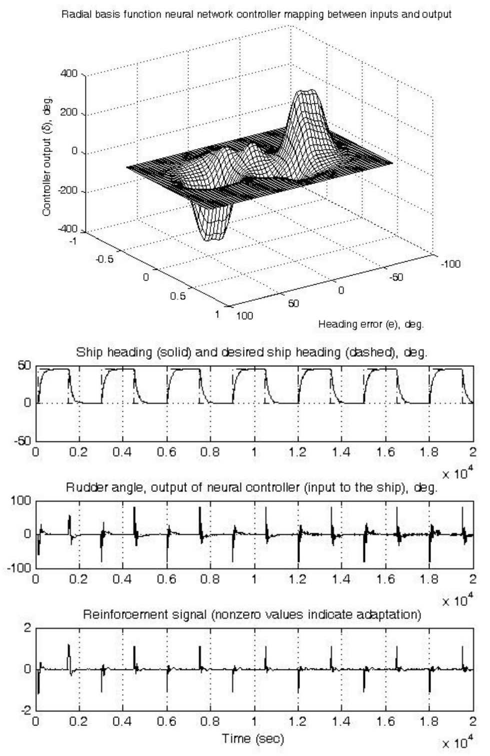


Figure 9. Controller response surface and closed-loop response resulting from using the optimized neural controller for tanker ship steering under sensor noise.

The close-loop response shown in Fig. 9 indicates the additive sensor noise uniformly distributed on $[-0.01, 0.01]$ has little effect on the response. The neural controller is robust to the sensor noise effects.

7 Conclusion

This paper described the study of PSO optimization of Radial Basis Function neural controller for the tanker ship steering system. Despite the highly nonlinear characteristics of tanker ship dynamics, the optimized neural controller has shown its robustness and ability to regulate steering angle under wind disturbance, speed change, sensor noise effect and sudden change of ship load. The neural controller exhibits its fast and adaptive changes of controller response to compensate the external and internal disturbances. The ship heading error is effectively minimized. The promising results in this paper clearly indicate that PSO can be an effective tool to optimize the adaptive neural controller steering control of tanker ship and other naval engineering applications.

References:

[1] C. G. Kallstrom, K. K. Astrom, N. E. Thocell, J. Eriksson, and L. Sten, "Adaptive Autopilots for Tanker," Automatica, p. 241-254, May 1979.

[2] M. Clerc and J. Kennedy, "The particle swarm – Explosion, stability, and convergence in a multidimensional complex space," IEEE Trans. Evol. Comput., vol. 6, no. 1, pp. 58-73, Feb 2002.

[3] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," Inf. Process. Lett., vol. 85, no. 6, pp. 317-325, 2003.

[4] K. E. Parsopoulos and M. N. Vrahatis, "Particle swarm optimization method for constrained optimization problems," in Intelligent Technologies – Theory and Applications: New Trends in Intelligent Technologies, vol. 76, P. Sincak, J. Vascak, V. Kvasnicka, and J. Pospicha, Eds. Amsterdam, The Netherlands: IOS Press, 2002, pp. 214-220.

[5] G. Toscano-Pulido and C. a. Coello, "A constraint handling mechanism for particle swarm optimization," in Proc. IEEE CEC, Portland, OR, Jun. 2004, vol. 2, pp. 1396-1403.

[6] Richard, Mark and Ventura, Dan, "Choosing a starting configuration for particle swarm optimization," Proceedings of the Joint Conference on Neural Networks, pages 68-75, 2005.

[7] Q. Du, M. Faber, M. Gunzburger, "Centroidal Voronoi Tessellations: Applications and algorithms," SIAM Review, vol. 41, 1999, pp. 637-676.

[8] K. Astrom and B. Wittenmark. Apdative Control. Reading, MA:Addison-Wesley Pub. Co., 1989.

[9] M. Bech and L. Smitt, "Analogue simulation of ship maneuver," Tech Rep., Hydro-og Aerodymisk Laboratorium, Lyngby, Denmark, 1969.

[10] B. Birge, "PSOt – a particle swarm optimization toolbox for use with Matlab," in Proceedings of the IEEE Swarm Intelligence Symposium, IN, pp. 182-186, 2003.

[11] Passino, K M. Biomimicry for Optimization, Control and Automation. Springer-Verlag London 2005.