Journal of Computers Vol. 28, No. 3, 2017, pp. 235-250
doi:10.3966/199115592017062803019

Application of Two Modified Autonomous Development Algorithms in Robot Obstacle Avoidance

Hong-Ge Ren, Rui Yin*, Tao Shi, and Fu-Jin Li

College of Electrical Engineering, North China University of Science and Technology
No. 21, Bohai Road, Tangshan, 063210, P. R. China
renhg@ncst.edu.cn 384042235@qq.com st99@ncst.edu.cn lfj@ncst.edu.cn

Received 17 June 2016; Revised 9 December 2016; Accepted 20 December 2016

Abstract. In view of high dimension, the difficulty of training, the problem of slow learning speed in the application of BP neural network in mobile robot path planning, an algorithm of reinforcement Q learning based on online sequential extreme learning machine (Q-OSELM algorithm) was proposed in this paper. And then, due to the random selection of weight and threshold parameters, it also proposes an extreme learning machine algorithm optimized by particle swarm (PSO-ELM algorithm) in this paper. Firstly, Q-OSELM algorithm obtains current environment and the status information of the robot through the characteristic of reinforcement learning, which combines dynamic network with supervised learning. After that, the online sequential extreme learning machine is used to approximate the function of the current status to get the rewards and punishments of the current status; Secondly, it is used to solve the problem of slow training speed by the characteristic of less parameter settings and better generalization performance. PSO-ELM algorithm is used to optimize the input weights and the hidden layer bias of the extreme learning machine which have been seen as the particle of particle swarm optimization algorithm to improve the network structure of the extreme learning machine. It could overcome inaccuracy of traditional extreme learning machine through particle swarm optimization algorithm. Finally, the performance of two learning algorithms is verified. The simulation experimental results show that the Q-OSELM learning algorithm improves the initiative of machine learning. And compared with the Q-OSELM algorithm, the PSO-ELM algorithm has better generalization ability and higher training precision. Simulation experiments are carried out to verify the stability and convergence of the two algorithms.

Keywords: autonomous learning, obstacle avoidance, online sequential extreme learning machine, particle swarm optimization algorithm, reinforcement Q learning,

1 Introduction

Nowadays mobile robot has become a hot issue in research field. However, the obstacle avoidance of robot is an important topic in the field of mobile robot learning. This problem has been described as follows: according to a few optimization methods, mobile robot can find an optimal or near optimal path. It can enable the robot to reach the target position from the starting position, and can avoid the obstacles on the path in spatial motion [1-2]. As a matter of fact, it is a problem of mobile robot navigation and avoiding collision. Since 1970s, there are mainly three methods on the path planning of mobile robot which could show as follow. Firstly, it is the mobile robot path planning method based on the environment models, which can deal with the complete consistent environment. The shapes and locations of the obstacles are in a given status, but it can't deal with the unknown environmental information online. The specific methods are grid method, visual map method and topological method. Secondly, it is the mobile robot path planning method based on the sensor information. The specific methods are artificial potential field method, determining the grid method, fuzzy logic method. At last, it is the mobile robot path planning method based on behavior. It decomposed navigation problem into multiple

* Corresponding Author

independent navigation behavior primitives. Such as obstacle avoidance, following, goal guidance, behavior elements, which have the complete motion control unit with sensors and actuators and corresponding navigation function. They coordinate each other, and complete the overall navigation tasks.

In view of the problem of the poor adaptive real-time performance of the traditional robot, slow learning and training speed, firstly a reinforcement learning theory based on online sequential extreme learning machine is proposed and applied to mobile robot system in this paper. Due to the robot's motion environment is complex and unknown. It needs a lot of time to learn, and it is difficult to be programmed for robot in various environments. Therefore, it is important to introduce a learning theory which has a cognitive learning mechanism and robot can learn autonomously online. Reinforcement learning adjusts the strategy from the actual system learning experience [3]. It is a process of gradually approaching the optimal strategy. Learning process doesn't require the supervision of the supervisor. And Q learning is the most basic reinforcement learning algorithm [4-7]. Extreme learning machine is a single hidden layer feed-forward neural network [8-11], which only needs set the number of hidden layer nodes in the network. In the implementation of the algorithm, it isn't required to adjust the input weights and the bias of the network, and produces the unique optimal solution. So it has the advantages of fast learning speed and good generalization performance. The learning algorithm proposed in this paper not only can make the agent improve the learning ability, but also can improve the learning speed of the robot greatly. Secondly, in view of the uncertainty of random input weights and thresholds of extreme learning machine, it also proposes a particle swarm optimization algorithm for extreme learning machine (PSO-ELM algorithm) to optimize the training parameters of extreme learning machine. In 1995, Kennedy and Eberhart were inspired by the phenomenon of flying birds cluster and foraging activities, and a particle swarm optimization algorithm was proposed [12-13]. The utility model has the advantages of small number of individuals and good robustness and great effect on all kinds of optimization problems. And the algorithm uses the particle swarm optimization algorithm to optimize the input weights and the hidden layer bias of the extreme learning machine. Then it optimizes the output weights. Under the premise of the fast training speed, the training accuracy of the algorithm is greatly improved, and the optimal planning path of the robot is obtained.

In Section 1, we introduce some scholars' research results in recent years, illustrate the limitations of these methods and describe the superiority of proposed algorithms. In Section 2, we illustrate some traditional algorithms such as Q learning algorithm, PSO algorithm, extreme learning machine and so on, and design these two kinds of autonomous development algorithms. In Section 3, we design the obstacle avoidance system for mobile robot. And in Section 4, we verify the superiority and precision of the proposed algorithms. Finally, Section 5 concludes this paper and our ideas for future work.

2 Design of Two Kinds of Autonomous Development Algorithms

2.1 Q Learning Algrithm

Reinforcement learning is an important machine learning method. The basic principle of reinforcement learning [14] has been shown in Fig. 1. It gets positive reward through a behavior strategy of agent from the environment, and makes the probability of this behavior strategy increase. The next action will depend on whether the agent can make it to obtain the maximum cumulative reward value of behavior strategy.

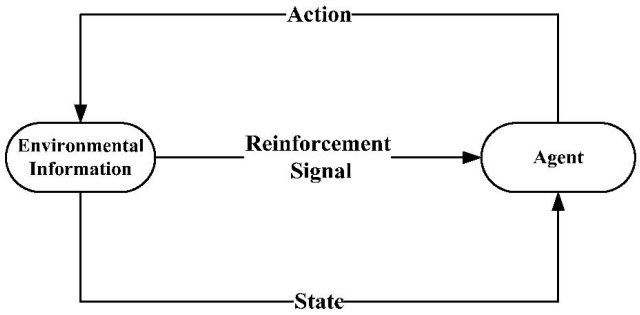

Fig. 1. Reinforcement learning framework

Q learning algorithm is the most basic learning algorithm in reinforcement learning, which combines the dynamic programming with the knowledge of animal psychology. So that it can realize the online learning of the machine with reward. The algorithm is modeled by the Markov decision process, and the optimal solution is obtained by the iterative calculation. The formula for iteration is as (1).

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \kappa\left[R(s_t, a_t) + \gamma_m Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right] \quad \textbf{(1)}$$

Among them, $Q(s_t, a_t)$ indicates that the size of value function after executing the action of a_t at the status of s_t; γ is the discount factor; κ is the learning factor, and $0 < \kappa < 1$; $R(s_t, a_t)$ is the reward value obtained after executing the action of a_t at the status of s_t.

Q learning algorithm theory is realized according to the following steps:

Step 1: Initialize $Q(s_t, a_t)$ randomly;

Step 2: Observe the current status s_t and select the action a_t;

Step 3: Get the next status s_{t+1} and output the reward value R at the same time;

Step 4: Adjust the Q value according to the formula (1).

2.2 Particle Swarm Optimization Algorithm (PSO Algorithm)

In the PSO model, the essence is that each optimization problem can be regarded as a "particle" in the search space [15]. Each particle has a "Fitness value", which depends on the fitness function $F(X_j)$. All particles are given the memory function. So it can accurately remember the best location of them in the search process. Each particle runs randomly in the solution space. Their motion distance and direction are determined by the same particle velocity. The particles adjust timely according to their own flight experience and their companion's flight experience, and update their position and speed according to the two "extremum" and "Fitness value". One of "extremum" is the optimal solution of the particle named individual best position p_{best}; the other "extremun" is the optimal solution for the whole population named global best position g_{best}.

In D-dimensional space, the population $X = (X_1, X_2, \cdots, X_m)$ is composed of m particles. Among them, $X_j = (x_{j1}, x_{j2}, \cdots, x_{jD})^T$ indicates the position of the jth particle. $V_j = (v_{j1}, v_{j2}, \cdots, v_{jD})^T$ indicates the velocity of the jth particle. $P_j = (p_{j1}, p_{j2}, \cdots, p_{jD})^T$ indicates the individual optimal position of the jth particle. $P_g = (p_{g1}, p_{g2}, \cdots, p_{gD})^T$ indicates global optimal position of population. In the training process of PSO algorithm, the particle velocity and position are updated by the formula (2) and (3).

$$V_{id}^{u+1} = wV_{id}^{u} + c_1 R_1 \left(P_{id}^{u} - X_{id}^{u}\right) + c_2 R_2 \left(P_{gd}^{u} - X_{id}^{u}\right) \quad \textbf{(2)}$$

$$X_{id}^{u+1} = X_{id}^{u} + V_{id}^{u+1} \quad \textbf{(3)}$$

In the formulas, u represents evolutionary iteration number. c_1, c_2 are the acceleration constants, and they are nonnegative number. We usually take c_1=c_2=2. R_1, R_2 are two constants which obtain into the interval of (0,1). w represents the inertia factor which is used to adjust the scope of the regulation of solution space. In formula (2), wV_{id}^{u} is called the "momentum" section which represents current velocity of particles. $c_1 R_1 \left(P_{id}^{u} - X_{id}^{u}\right)$ is called the "cognitive" section which represents current behavior in thinking and memory of particles. $c_2 R_2 \left(P_{gd}^{u} - X_{id}^{u}\right)$ is called the "society" section which represents information sharing and cooperation among particles.

Algorithm process of PSO is as follows:

Step 1: Initialize the position and velocity of the particle swarm. Take the acceleration constants c_1=c_2=2 and set the maximum number U_{max} of iterations.

Step 2: The fitness value f of each particle is calculated according to the fitness function $F(X_j)$.

Step 3: Compare the calculated fitness value f with the individual optimal position p_{best}. If the performance of calculated fitness value is better than p_{best}, we replace the previous individual optimal position with X_j. On the contrary, the current optimal position is unconverted.

Step 4: Compare the calculated fitness value *f* with the global optimal position g_{best} of the individual. If the performance of calculated fitness value is better than g_{best}, we replace the global optimal position with X_j. On the contrary, the current global optimal position is unconverted.
Step 5: Update position and velocity of each particle according to formula (2), (3), and update the next particle in accordance with **Step 2~Step 4** again.
Step 6: Detect whether the adaptation value has reached the specified accuracy. If the specified accuracy is reached the maximum number of iterations U_{max}, and the loop iteration is over. Otherwise it returns **Step2**.

2.3 Extreme Learning Machine

Extreme learning machine is a kind of single-hidden layer feed-forward networks. The ELM algorithm need not iteration. The input weights and hidden layer nodes obtain randomly. The purpose is to achieve the training error minimization, and to determine the output weights through this algorithm. The network structure diagram is shown in Fig. 2.

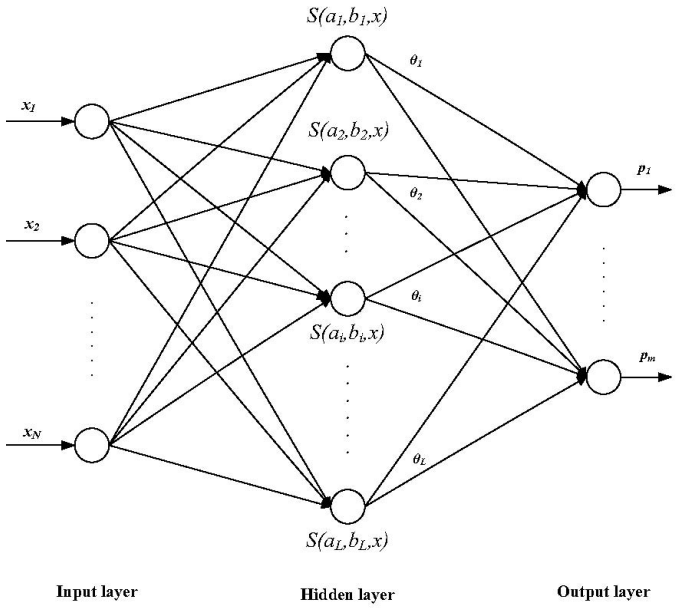

Fig. 2. Network model of extreme learning machine

For N arbitrary and different samples of (x_i, q_i), Among them, $x_i = [x_{i1}, x_{i2}, \cdots, x_{in}]^T \in R^n$, $q_i = [q_{i1}, q_{i2}, \cdots, q_{im}]^T \in R^m$, the model of the extreme learning machine with N hidden layer nodes by mapping function of S can be expressed as formula (4).

$$f_L(x) = \sum_{i=1}^{L} \theta_i S(a_i, b_i, x_j) = p_j, j = 1,2,\cdots,N \quad \textbf{(4)}$$

Among them, $a_i = [a_1, a_2, \cdots, a_n]^T$ represents the weight vector of the hidden layer nodes and the input nodes in the *i*th layer. b_i represents the bias of the *i*th hidden layer node. $\theta_i = [\theta_1, \theta_2, \cdots, \theta_n]^T$ represents the weight vector of the hidden layer nodes and the output nodes in the *i*th layer. L represents the number of hidden layer nodes. Thus, the formula (4) can be simplified as a formula (5).

$$H\theta = P \quad \textbf{(5)}$$

Among them, H represents hidden layer output matrix. The *i*th column of H correspond to the input $x_1, x_2, \cdots, x_n$ of the *i*th hidden layer output vector, and $\theta = \left[\theta_1^T, \theta_2^T, \cdots, \theta_L^T\right]_{L\times m}$, $P = \left[p_1^T, p_2^T, \cdots, p_N^T\right]_{N\times m}$.

In the ELM algorithm, input weights and thresholds select randomly. So the hidden layer output matrix is a known matrix. Thus, the extreme learning machine is transformed into solve the least square solution of the input weights to achieve the training of the network. Output weight matrix $\hat{\theta}$ is as formula (6).

$$\hat{\theta} = H^* P = \left(H^T H\right)^{-1} H^T P \tag{6}$$

In the formula, H^* represents the Moore-Penrose generalized inverse of the hidden layer output matrix.

2.4 Online Sequential Extreme Learning Machine

In view of the traditional extreme learning machine in the industrial production process can not deal with the dynamic model and generate a sequence of batch training data and other issues. A simple and effective learning model of data learning machine, which broadens the application range of ELM is proposed. It realizes single or learns sample data together. The sample data is only in the learning phase. After the completion of the study ELM network throws away the data immediately. Generally speaking, online sequential extreme learning machine learn sample data which obtained in the present moment. It doesn't require learning the sample data which has been trained before, and reduces the time of learning and training greatly.

Online sequential extreme learning machine [16~18] is an online incremental learning method based on the traditional ELM algorithm. The steps of the algorithm are given by reference [19]:

Step 1: Initialization phase

Select the partial data set $D_0 = \{(x_i, p_i), i = 1,2,\cdots, N_0\}$ from the given sample set $D = \{(x_i, p_i), i = 1,2,\cdots, N\}$, among them $N_0 \geq L_0$.

Step 1.1: Obtain the input weights a_i and $b_i (i = 1,2,\cdots, L)$. Calculate the hidden layer output matrix H_0;

Step 1.2: Calculate the initial output weights $\theta^0 = Y_0 H_0{}^T P_0$. Among them, $Y_0 = \left(H_0{}^T H_0\right)^{-1}$, $P_0 = \left[p_1, p_2, \cdots, p_{N_0}\right]^T$;

Step 1.3: Set k =0.

Step 2: Serialization stage

Step 2.1: Learn the k +1th data that is $d_{k+1} = \left(x_{N_0+k+1}, p_{N_0+k+1}\right)$;

Step 2.2: Set $P_{k+1} = \left[t_{N_0+k+1}\right]^T$ and calculate the implicit layer output matrix of new learning data:

$$H_{k+1} = \left[S\left(a_1 \cdot x_{N_0+k+1} + b_1\right) \;\; S\left(a_2 \cdot x_{N_0+k+1} + b_2\right) \;\; \cdots \;\; S\left(a_L \cdot x_{N_0+k+1} + b_L\right) \right]$$

Step 3: Calculate output weights θ^{k+1}

$$\begin{cases} Y_{k+1} = Y_k - Y_k H_{k+1}{}^T \left(I + H_{k+1} Y_k H_{k+1}{}^T\right)^{-1} H_{k+1} Y_k \\ \theta^{k+1} = \theta^k + Y_{k+1} H_{k+1}{}^T \left(P_{k+1} - H_{k+1} \theta^k\right) \end{cases}$$

Step 4: Set k=k+1, if k>N, algorithm is end. Otherwise it returns Step 2.1.

Definition 1. The output value of the online sequence extreme learning machine network is θ^{k+1} after the kth iteration. Intermediate parameter matrix is Y_{k+1}. Due to OSELM is learning online. We can calculate the parameters of the k+1th through the parameters of the kth. Iterative formula is

$$\begin{cases} Y_{k+1} = Y_k - Y_k H_{k+1}{}^T \left(I + H_{k+1} Y_k H_{k+1}{}^T\right)^{-1} H_{k+1} Y_k \\ \theta^{k+1} = \theta^k + Y_{k+1} H_{k+1}{}^T \left(P_{k+1} - H_{k+1} \theta^k\right) \end{cases}$$

Proving

First of all, ELM algorithm is used to complete the training of the batch data, such as formula (7).

$$\theta^0 = \left(H_0{}^T H_0\right)^{-1} H_0{}^T P_0 \tag{7}$$

After that, the new data which has M training samples are added in the training model. The new hidden layer output matrix of the network can be expressed as formula (8).

$$H(a_1,\cdots,a_L,b_{1,}\cdots,b_L,x_1,\cdots,x_N,\cdots,x_{N+M}) = \begin{bmatrix} S(a_1,b_1,x_1) & \cdots & S(a_L,b_L,x_1) \\ \vdots & \ddots & \vdots \\ S(a_1,b_1,x_N) & \cdots & S(a_L,b_L,x_N) \\ S(a_1,b_1,x_{N+1}) & \cdots & S(a_L,b_L,x_{N+1}) \\ \vdots & \ddots & \vdots \\ S(a_1,b_1,x_{N+M}) & \cdots & S(a_L,b_L,x_{N+M}) \end{bmatrix} \tag{8}$$

Formula (8) can be simplified as a formula (9).

$$H = \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \tag{9}$$

Network output matrix can be rewritten as the formula (10).

$$P = \begin{bmatrix} P_0 \\ P_1 \end{bmatrix} \tag{10}$$

So the new output value is updated to the formula (11).

$$\begin{aligned} \theta &= \left(H^T H\right)^{-1} H^T P \\ &= \left\{ \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \right\}^{-1} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} P_0 \\ P_1 \end{bmatrix} \\ &= \left(H_0{}^T H_0 + H_1{}^T H_1\right)^{-1} \left(H_0{}^T P_0 + H_1{}^T P_1\right) \end{aligned} \tag{11}$$

To set $Y_0 = \left(H_0{}^T H_0\right)^{-1}$ **(12)**

$$Y_1 = \left(H_0{}^T H_0 + H_1{}^T H_1\right)^{-1} \tag{13}$$

Then we have formula (14).

$$Y_1 = \left(Y_0{}^{-1} + H_1{}^T H_1\right)^{-1} \tag{14}$$

According to the matrix inversion theorem of Sherman-Morrison, the formula (14) can be simplified as the formula (15).

$$Y_1 = Y_0 - \frac{Y_0 H_1{}^T H_1 Y_0}{I + H_1 Y_0 H_1{}^T} \tag{15}$$

By formula (15), we can be seen that Y_1 can be deduced by Y_0. So we can get the formula (16).

$$Y_{k+1} = Y_k - \frac{Y_k H_{k+1}{}^T H_{k+1} Y_k}{I + H_{k+1} Y_k H_{k+1}{}^T} \tag{16}$$

Calculate the inverse of both sides in formula (14) by formula (17).

$$Y_1^{-1} = Y_0^{-1} + H_1{}^T H_1 \Rightarrow Y_0^{-1} = Y_1^{-1} - H_1{}^T H_1 \tag{17}$$

So we get the formula (18).

$$\theta^0 = \left(H_0{}^T H_0\right)^{-1} H_0{}^T P_0 = Y_0 H_0{}^T P_0$$

$$\Rightarrow H_0{}^T P_0 = Y_0^{-1}\theta^0 \tag{18}$$

We derive the formula (11) to the formula (19).

$$\begin{aligned}\theta^1 &= \left(H_0{}^T H_0 + H_1{}^T H_1\right)^{-1}\left(H_0{}^T P_0 + H_1{}^T P_1\right)\\ &= Y_1\left(H_0{}^T P_0 + H_1{}^T P_1\right)\\ &= Y_1\left(Y_0^{-1}\theta^0 + H_1{}^T P_1\right)\\ &= Y_1\left[\left(Y_0^{-1} - H_1^T H_1\right)\theta^0 + H_1^T P_1\right]\\ &= \theta^0 + Y_1 H_1{}^T\left(P_1 - H_1\theta^0\right)\end{aligned} \tag{19}$$

In the formula, θ^0, H_1, P_1 are all known matrix.

It can be seen that the new output weight matrix can be updated in the old model and need not train again. Update formula can be expressed as formula (20).

$$\theta^{k+1} = \theta^k + Y_{k+1} H_{k+1}{}^T\left(P_{k+1} - H_{k+1}\theta^k\right) \tag{20}$$

Proof finished.

Compared with the other network models, the model of the online sequential extreme learning machine has the advantages of less parameter, shorter learning time, simple and easy to implement. And it is a good model of the data learning machine.

2.5 Q Learning Algorithm Based on Online Sequential Extreme Learning Machine (Q-OSELM)

In order to get the biggest cumulative reward value of the learning process, the reinforcement learning algorithm is that agent communicates with environment timely through the action, and obtains the reward and punishment from the environment. Its disadvantage is that convergence speed of the simple agent interact with environment is very slow. Based on this kind of problem, this paper proposes a Q algorithm based OSELM (Q-OSELM), and overcome these defects.

The input of OSELM is the external environment information that the mobile robot can perceive. The output is the corresponding reward and punishment for each action decision, named Q value. The framework of the network structure is shown in Fig. 3.

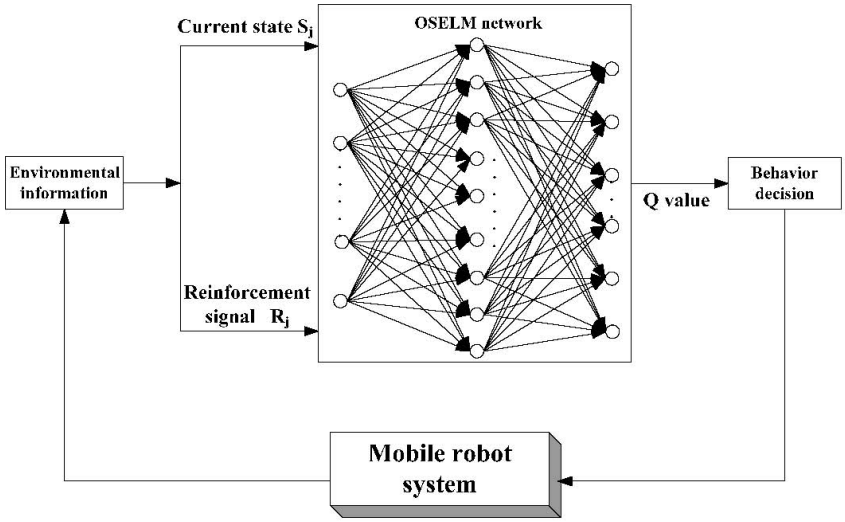

Fig. 3. Framework of Q learning network based on OSELM

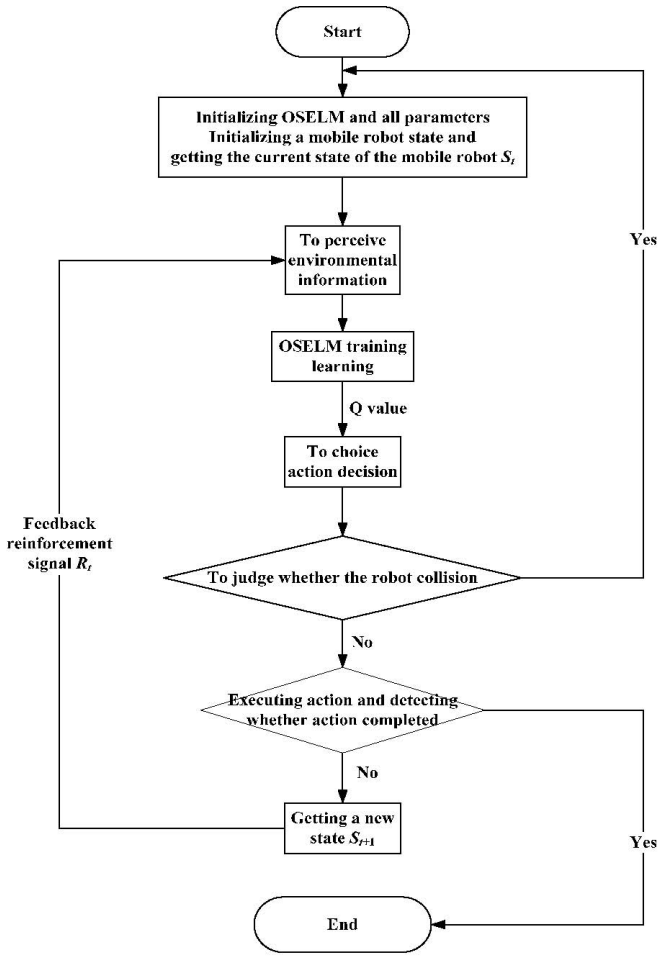

Fig. 4. The flow chart of Q-OSELM algorithm

The algorithm is realized according to the following steps:
Step 1: Initialize the various parameters which used in network training and OSELM neural network;
Step 2: Initialize the status of the mobile robot;
Step 3: Obtain the status information for the current mobile robot system;
Step 4: Input current status information to the OSELM network for training, and select action decision according to the output of the Q value;
Step 5: Perform action decisions and make the mobile robot achieve a new state s_{t+1}. At the same time, a feedback signal R_t is obtained. If the robot has a collision, the robot will return the initial status and starts again;
Step 6: Complete the network training by the feedback signal to the ELM through the environmental;
Step 7: Repeat **Step 3~Step 6** until the training completed, and complete autonomous learning.
The flow chart of the Q-OSELM algorithm is shown as Fig. 4.

Many studies show that many of the advantages of ELM. It can meet the needs of Q learning [20]. Firstly, ELM is a single-hidden layer feed-forward neural network. It takes an adaptive way to imitate the human brain to learn and train. Network generalization ability is very strong, so that the intelligent system of agent can adapt to environmental changes better.

Secondly, ELM has the characteristics of self-learning, self-organization ability and so on. These not only ensure the convergence of the Q learning algorithm, but also enhance the ability of the agent to recognize the unknown environment greatly.

Finally, ELM has a strong fault tolerance capability which can be identified according to the characteristics of the controlled object.

2.6 Extreme Learning Machine Algorithm Optimized by Particle Swarm (PSO-ELM Algorithm)

From the 2.3 section, we find that the extreme learning machine obtains the input weights and the bias of hidden layer randomly. Due to the output weights of the network are calculated by the input weights and the bias of hidden layer. The main problems are the training accuracy is low, partial hidden layer node failure and so on. In view of the above mentioned problems, this paper proposes an extreme learning machine algorithm optimized by particle swarm (PSO-ELM algorithm). It uses particle swarm optimization algorithm to optimize the network input weights and hidden layer bias of the extreme learning machine to obtain an optimal network with a relatively high learning accuracy.
The flow chart of the PSO-ELM algorithm is shown as Fig. 5.

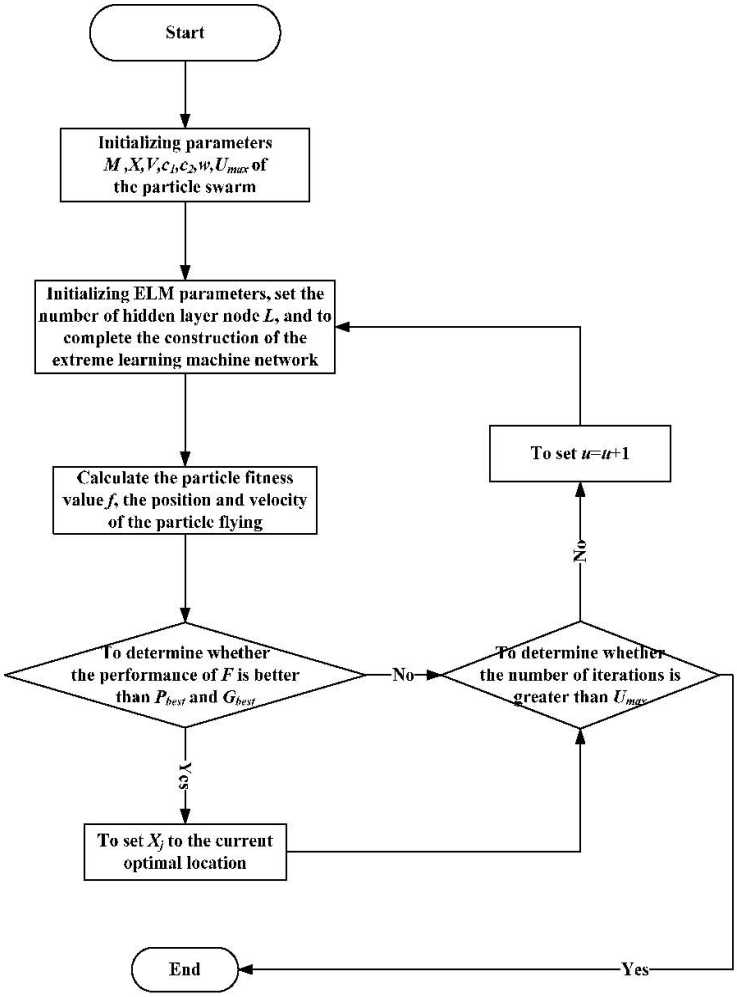

Fig. 5. The flow chart of the PSO-ELM algorithm

The steps of PSO-ELM algorithm are as follows:
Step 1: Initialize the number of particle swarm m, position X and velocity V, maximum number of evolutionary iteration U_{max}, acceleration factor $c_1=c_2=2$, and inertia factor w;
Step 2: Construct the extreme learning machine network. Initialize each parameter of extreme learning machine, and set extreme learning machine network hidden layer node number L;
Step 3: Code particle swarm. According to the input layer and the hidden layer weights ω which are obtained by coding, the weights between the hidden layer and the output layer β, and hidden layer node bias, we can calculate the output matrix of the training samples $\hat{P}$. The fitness value of the particle position has been calculated.
Step 4: According to formula (2) and (3), the velocity and position of the particle swarm optimization have been calculated.

Step 5: All particles should be compared with their previous best position and the whole particle swarm. If it is better than the previous best position, and updates the current position to be optimal. Otherwise the optimal position is unconverted.
Step 6: Update current iteration number $u=u+1$. If the current iteration number is less than the maximum number of iterations U_{max}, then returns to **Step 3** for further iteration.
Step 7: According to the position of the optimal particle, the network parameters of extreme learning machine ω, β, b are obtained. And then the optimal extreme learning machine network outputs are obtained. And the optimal path planning and control system are established.

3 Obstacle Avoidance System for Mobile Robot

In order to realize autonomous obstacle avoidance of mobile robot, a kind of intelligent control system based on Q-OSELM algorithm is designed. It is as shown in Fig. 6. Based on the idea of control structure, in the process of work, there are many kinds of basic action of robot. This paper aims to research path planning. And the algorithm can improve the learning speed and self-learning ability of robot. Robots can be evaluated according to the action. And then adjust the behavior strategy according to the evaluation. It also could implement the best action for path planning. Finally, the function of the executor is to achieve the action when the robot completed the task. And the intelligent control structure model of extreme learning machine algorithm optimized by particle swarm needs to be replaced the learning framework of Q-OSELM with PSO-ELM.

Overall, a robot obtains status information from the environment and generates a corresponding action. With the increase in the number of operations, the mobile robot can learn the best combination of status and action. It can choose the right action according to the environment, and can achieve the optimal path planning.

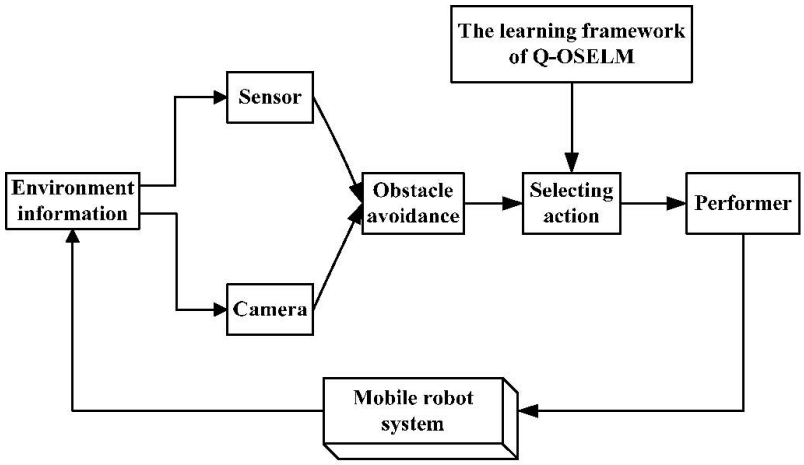

Fig. 6. Mobile robot intelligent control structure model

4 Experiment and Simulation Results Analysis

In order to verify the feasibility of the algorithm, experiments were carried out by Matlab 7.0. The experimental environment is Windows7, 64bit operating system, Core i5-3470 3.2GHz 4G Intel memory.

4.1 Simulation Experiment Design and Result Analysis for Q-OSELM Algorithm

The experiment set up a small square obstacle and three large square obstacles, a starting position and a target position. And their positions were shown in Fig. 7.

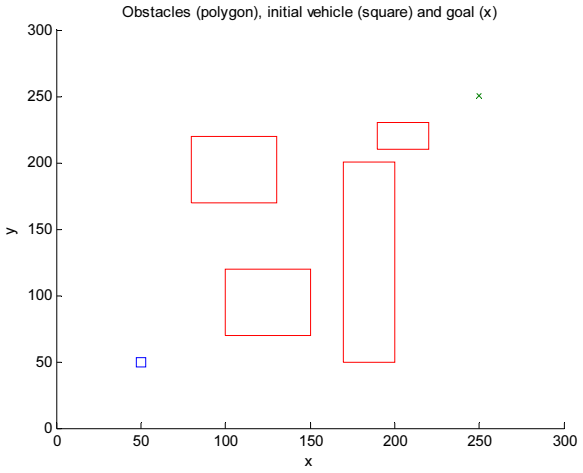

Fig. 7. Square obstacles, starting point and target location

The key point of the experiment was how to get the reinforcement return of Q learning. Therefore, selected the appropriate reinforcement learning which could improve the convergence rate of training network. According to the task of the robot in this paper, we selected the discount factor $\gamma = 0.95$, learning factor $\kappa = 0.7$, and separated the reinforcement return into two parts:

Reinforcement return R_{r_obs} was generated by the relative movement direction of the mobile robot and the obstacles. Reinforcement return R_{r_goal} was generated by the relative movement direction of the mobile robot and the target position.

The finally return was the sum of the two parts of the reinforcement returns.

$$R_{r\_obs} = \begin{cases} -0.4 & \text{Movement of the obstacles} \\ +0.6 & \text{Far From the obstacles} \\ -1.0 & \text{Collision with obstacles} \\ 0 & \text{Another movement states} \end{cases} \qquad R_{r\_goal} = \begin{cases} +0.5 & \text{Close to the target point} \\ -0.5 & \text{Stay away from the target point} \\ 0 & \text{Another movement states} \end{cases}$$

In the course of training, the proposed Q-OSELM algorithm was applied in this paper. Because the obstacles were distributed randomly in the environment, the robot will select the initial stage randomly. And led to a collision. Robots got the corresponding reinforcement returns, and carried out the storage. After then, we approximated the function through ELM network in order to complete training. After a period of training, the robot could learn to avoid obstacles and achieve the target point smoothly. Simulation results were shown in Fig. 8.

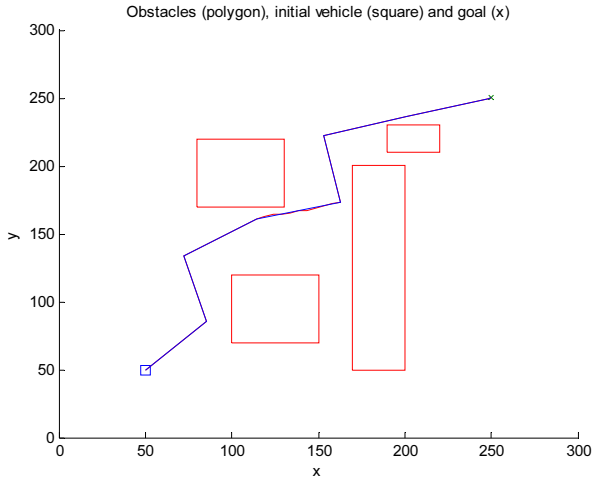

Fig. 8. Obstacle avoidance path

Table 1. Comparison of convergence speed and error of algorithm

	Iteration number	Error
TD algorithm	100	3.2%
Q-OSELM algorithm	25	5.7%

In two figures, the blue square is the starting position of the robot, the green "×" is the target point that robot want to arrive, red polygons are the obstacles' positions. The red line indicates the robot walking route, while the blue line indicates the better path in this task.

The robot moves in a random environment with obstacles. At beginning, due to the robot is highly unfamiliar to the unknown environment to lead to collision. Then, after 25 times of training, according to different reward value, robot completes the task of autonomous learning obstacle avoidance and reaches the designated target location by location. Simulation experimental results show that the algorithm has fast training speed. And it is proved that the algorithm has fast convergence. But from Table 1, it can be seen that the training error of Q-OSELM algorithm proposed in this paper is larger than TD algorithm.

In Fig. 8, we can see that the path is so disorderly in the middle position where obstacles are very dense, and lead to the position of the robot walking around here. It shows that the judgment of the algorithm for dense obstacles accuracy has certain defects. And the path is not accurate enough. In this paper, an extreme learning machine algorithm optimized by particle swarm is proposed to improve the accuracy of the path of the robot, and the limitation of the algorithm is improved.

4.2 Simulation Experiment Design and Results Analysis for PSO-ELM Algorithm

In the early stage of the experiment, 10 obstacles and 1 target points were set up in the simulation environment randomly. We selected the number of particles is 70. The algorithm ran 10 times. And experiments showed that each experiment could achieve the optimal path. The second phase of the experiment, we changed the coordinates of the obstacle and target position, and increased the number of obstacles to 14. The number of particles we selected is 80. The algorithm ran 20 times. And experiments showed that 1 failure, 4 times near optimal paths, and 15 times optimal paths. The average iteration number of the algorithm is 6. The simulation results were shown in Fig. 9, 10, 11, 12. The walking path of the simulated robot was observed and analyzed.

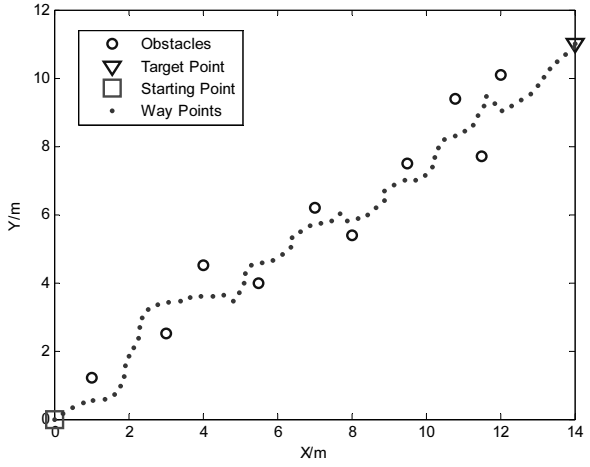

Fig. 9. The robot's trajectory in a simple and unknown environment

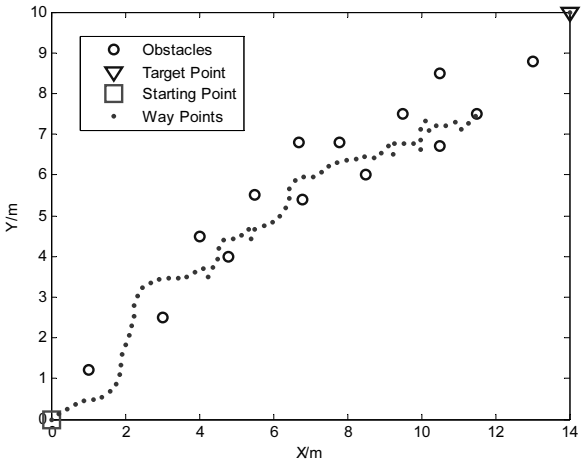

Fig. 10. The movement trajectory after changing the obstacles and the target robot for the first time

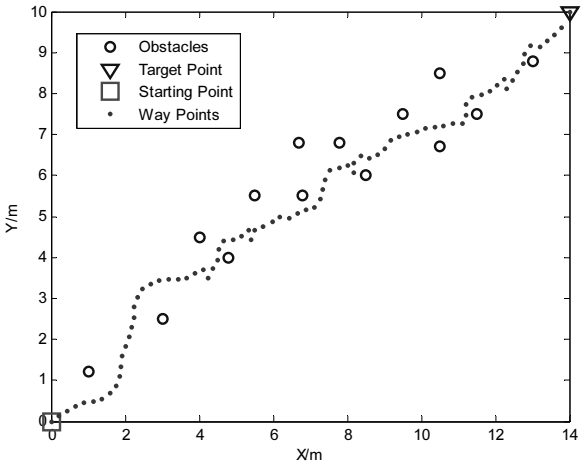

Fig. 11. The movement trajectory after learning three times

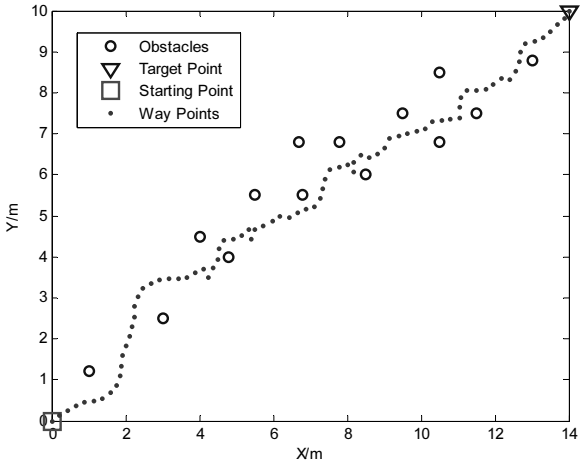

Fig. 12. The movement trajectory after learning six times

In these figures, "∘" represents the coordinate positions of the obstacles. "·" means the position of the robot and the moving trajectory of the robot. And the position coordinate of the target point is expressed in "Δ", "□" represents the starting point.

Obviously, in Fig. 9, we could see that the mobile robot could find the optimal path quickly to avoid obstacles and reach the target point by autonomous learning according to PSO-ELM algorithm.
Fig. 10 showed a simulation result of the first training following the change of the obstacles' and the target's position. It's failing. As we can see, from the change of environment, the robot couldn't adapt the environment rapidly and bump with the obstacles.

Fig. 11 was the movement trajectory of the robot after 3 times of learning. It could be seen that the robot could get the environment information and could avoid the obstacles and reach the target position. But the robot path wasn't smooth, resulting in many redundant points. It is near optimal path.
Fig. 12 showed the movement track of the robot after 6 times. It could be seen that the robot could avoid obstacles successfully and could reach the target location. The path was smooth. It completed the global optimal position. The robot might achieve autonomous learning.

In view of the accuracy of the algorithm, we compared the results of the Q-OSELM algorithm with PSO-ELM algorithm. The comparison results were shown in Fig. 13.

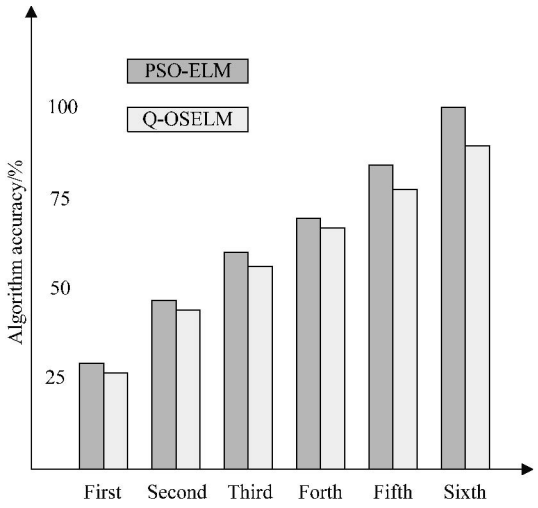

Fig. 13. Change of the algorithms' accuracy with the training times

We could be seen in the figure. At the beginning of the training, the accuracies of the two algorithms were not much more different. However, with the increase of the number of training, after six times of training, accuracy of PSO-ELM algorithm could be reached 100%, while the accuracy of the Q-OSELM algorithm was only 90%. Therefore, PSO-ELM algorithm could greatly improve the convergence accuracy of the algorithm, which could obtain the optimal path.

5 Conclusion

Firstly, this paper presents a Q learning algorithm based on OSELM, named Q-OSELM algorithm. It is applied to the obstacle avoidance of mobile robot. And the corresponding relationship between the behavior and the reward signal is learned gradually, and learns the surrounding unknown environment independently. So as to make the appropriate action to adapt to the environment, the stability and convergence of the algorithm are proved by simulation experiments. Extreme learning machine has a fast train speed, which can accelerate the implementation of Q learning in the agent and can save a lot of time. Secondly, it is mentioned an extreme learning machine algorithm optimized by particle swarm, named PSO-ELM algorithm. In order to get the global optimal obstacle avoidance path, this algorithm optimizes the weights and hidden layer bias of extreme learning machine network to optimize the relative position of the robot relative to the obstacle.

These two algorithms not only can be used in robot's obstacle avoidance, but also can be used in robot's navigation, robot tracking and so on. It can realize the intelligence and autonomy of the mobile robot.

However, the precision and robustness of extreme learning machine are quite advantageous. It can also be applied to the fields of image processing, temperature detection and so on. It has larger research space.

Acknowledgement

Rui Yin was supported by the National Nature Science Foundation (NNSF) under Grant of China 61203343. The author would also like to acknowledge the contributions of Hongge Ren to the simulation experiment and technical support from Tao Shi and Fujin Li.

References

[1] H.Y. Shi, C.Z. Sun, Motion planning of mobile robot in unstructured environment, Robot 26(1)(2004) 27-31.

[2] X.P. Fan, S.Y. Li, T.F. Chen, Robot dynamic obstacle avoidance planning based on new artificial potential field function, Control Theory and Application 22(5)(2005) 703-707.

[3] L.P. Kaelbling, M.L. Littman, A.W. Moore, Reinforcement learning: a survey, Journal of Artificial Intelligence Research 4 (3)(1996) 237-285.

[4] C. Watkins, P. Dayan, Q-Learning, Machine Learning 8(1992) 279-292.

[5] J.N. Tsitsiklis, Asynchronous stochastic approximation and Q-learning, Machine Learning 16(3)(1994) 185-202.

[6] M. Schembri, M. Mirolli, G. Baldassarre, Evolving internal reinforcers for an intrinsically motivated reinforcement learning robot, in: Proc. 6th IEEE International Conference on Development and Learning (ICDL2007), 2007.

[7] A. Johnson, A. Barto, Causal graph based decomposition of factored MDPs, Journal of Machine Learning Research 7(7)(2006) 2259-2301.

[8] G.B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine, a new learning scheme of feed-forward neural networks, in: Proc. IEEE International Joint Conference on Neural Networks, IEEE Budapest, 2004.

[9] B.Y. Wang, S. Zhao, S.M. Zhang, A distributed load forecasting algorithm based on cloud computing and extreme learning machine, Power System Technology 38(2)(2014) 526-531.

[10] W.W. Zong, G.B. Huang, Y.Q. Chen, Weighted extreme machine for imbalance learning, Neurocomputing 101(2013) 229-242.

[11] G.B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multi-class classification, IEEE Trans Pattern Anal Mach Intell 42(2)(2012) 513-529.

[12] X.L. Tang, H. Zhang, Y.Q. Cui, L. Gu, Y.Y. Deng, A novel reactive power optimization solution using improved chaos PSO based on multi-agent architecture, International Transactions on Electrical Energy System 24(5)(2014) 609-622.

[13] B. Fu, Passive target localisation algorithm based on optimizing extreme learning machine with PSO, Computers Applications and Software 32(11)(2015) 325-328.

[14] Y. Gao, S.F. Chen, X. Lu, Survey of reinforcement learning, Automation Journal 30(1)(2004) 86-100.

[15] Y. Zhai, X.B. Guo, The path optimization method on underwater robots getting rid of obstacles, Microelectronics & Computer 33(2)(2016) 100-103.

[16] Q.Z. Zhang, Y.L. Zhou, Active noise control using a feed-forward network with online sequential extreme learning machine, in: Proc. International Symposium on Neural Networks, 2008.

[17] L. Yang, R. Zhang, Online sequential ELM algorithm and its improvement, Journal of Northwest University: Natural Science Edition 42(6)(2012) 885-896.

[18] Y. Lan, Y.C. Soh, G.B. Huang, Ensemble of online sequential extreme learning machine, Neurocomputing 72(13)(2009) 3391-3395.

[19] J.W. Wang, W.T. Mao, L. He, L.Y. Wang, Weighted online sequential extreme learning machine based on imbalanced sample-reconstruction, Journal of Computer Applications 35(6)(2015) 1605-1610.

[20] S.H. You, Y.C. Zhou, H. Wang, An overview of reinforcement learning based on neural networks, Computer Knowledge and Technology 8(28)(2012) 6782-6786.