ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
DOI: 10.17559/TV-20160108193022

SKINNER OPERANT CONDITIONING MODEL AND ROBOT BIONIC SELF-LEARNING CONTROL

Jianxian Cai, Li Hong, Lina Cheng, RuihongYu

Original scientific paper
A Fuzzy Skinner Operant Conditioning Automaton (FSOCA) is constructed based on Operant Conditioning Mechanism with Fuzzy Set theory. The main character of FSOCA automaton is: the fuzzed results of state by Gaussian function are used as fuzzy state sets; the fuzzy mapping rules of fuzzy-conditioning-operation replace the stochastic "conditioning-operant" mapping sets. So the FSOCA automaton can be used to describe, simulate and design various self-organization actions of a fuzzy uncertain system. The FSOCA automaton firstly adopts online clustering algorithm to divide the input space and uses the excitation intensity of mapping rule to decide whether a new mapping rule needs to be generated in order to ensure that the number of mapping rules is economical. The designed FSOCA automaton is applied to motion balanced control of two-wheeled robot. With the learning proceeding, the selected probability of the optimal consequent fuzzy operant will gradually increase, the fuzzy operant action entropy will gradually decrease and the fuzzy mapping rules will automatically be generated and deleted. After about seventeen rounds of training, the selected probabilities of fuzzy consequent optimal operant gradually tend to one, the fuzzy operant action entropy gradually tends to minimum and the number of fuzzy mapping rules is optimum. So the robot gradually learns the motion balance skill.

Keywords: *balanced control; Fuzzy Set; mapping rules; Skinner Operant Conditioning Mechanism*

Model Skinner Operant Conditioning automata i bionički naučeno upravljanje robota

Izvorni znanstveni članak
Fuzzy Skinner Operant Conditioning Automaton (FSOCA) sastavljen je na temelju Operant Conditioning mehanizma primjenom teorije neizrazitih skupova. Osnovno obilježje automata FSOCA je sljedeće: neizraziti rezultati stanja pomoću Gausove funkcije koriste se kao skupovi neizrazitog stanja; neizrazita pravila preslikavanja (fuzzy mapping rules) kod fuzzy-conditioning-operacije zamjenjuju stohastičke "conditioning-operant" skupove preslikavanja. Stoga se automat FSOCA može koristiti za opisivanje, simuliranje i dizajniranje raznih samo-organizirajućih radnji fuzzy nesigurnog sustava. Automat FSOCA najprije usvaja online algoritam grupiranja (clustering) u svrhu podjele ulaznog prostora (input space) te koristi intenzitet pobude pravila preslikavanja kako bi odlučio treba li generirati novo pravilo preslikavanja da bi broj pravila preslikavanja bio ekonomičan. Dizajnirani FSOCA automat primijenjen je za reguliranje balansiranja gibanja robota s dva kotača. Kako se učenje nastavlja, odabrana vjerojatnoća fuzzy operanta koji optimalno slijedi postepeno će se povećavati, entropijsko djelovanje fuzzy operanta će se postepeno smanjivati pa će se automatski generirati i izbrisati neizrazita pravila preslikavanja. Nakon otprilike sedamnaest krugova obuke, odabrane vjerojatnosti neizrazitog posljedičnog optimalnog operanta postupno teže prema jednoj, entropija djelovanja neizrazitog operanta postupno se smanjuje i broj neizrazitih pravila preslikavanja postaje optimalan. Tako robot postupno uči vještinu balansiranja gibanja.

Ključne riječi: *neizraziti skup; pravila preslikavanja; Skinner Operant Conditioning Mechanism; uravnoteženo upravljanje*

1 Introduction

The combination of the disciplines of Psychology of Learning, Biology and Machine Learning leads to the development of Bionic Self-learning theory and practice. The main research objective of intelligent control and artificial intelligence has been enabling robots to obtain the bionic self-learning ability and gradually acquire new knowledge in the process of operation and gaining similar skills of motion control possessed by animals and human beings. A great number of reports and literature has been dedicated to robot control, an area in bionic self-learning, which is under great interest in the study of neural networks [1÷7]. Although the study on robots based on artificial neural networks has connected robot motion control with neural physiology and cognitive science, the connection is still weak and the motion control skills of robots still rely on descriptive control rules, which involve excessive elements of design and less bionic self-learning and organizing skills in a biological system. This has impeded the development of bionic self-learning. The theory of Operant Conditioning, as an important guide to the study on the learning mechanism in human and animal neural networks, has brought the research on bionic self-learning to a new stage [8].

Since the mid-1990s, Carnegie Mellon University in the US has been focusing on the computing theory and model of Skinner Operant Conditioning and applied this model on the autonomous robots [9]. Under the influence of their study, several relevant research areas to Skinner Operant Conditioning including ALC (Autonomous Learning Control) have received wide study interest.

Professor Zalama of the Department of Automatic Control and Systems Engineering in the University of Valladolid, has conducted many in-depth researches on the learning and control behaviours of robots and with his team, developed a computing method for obstacle avoidance Operate Conditioning based on the theory of Operant Conditioning. Through enabling the robot to move at different angular velocities in an environment of disordered obstacles, and activating nodes in the angular velocity mapping, a system of weights was developed in this model. The robot gradually obtained the skill of obstacle avoidance in a surveillance-free environment by reinforcing negative signals generated by collision. In 2002, a neural networks response to stimulation was developed in response to the navigation problem to correct navigation errors and realize learning reinforcement in Operant Conditioning [10]. In 2006, they designed learning and computing model of first and second order conditioning for a robot named Arisco with audio-visual sensation. This model was created using competition artificial neural networks and enabled Arisco to have some self-organizing functions [11]. In 1997,

Gaudiano and Chang of the laboratory of Neurobotics in Boston University in the US conducted a similar research to build a neural computing model on the combination of Pavlov theory and Skinner Operant Conditioning theory in response to a navigation problem in a wheeled robot named Khepera. Khepera can learn obstacle avoidance through navigation without any empirical knowledge and instructing signals [12]. In 2005, a research team on robots in Mechanical Engineering, Waseda University reported their study results. Itoh et al. believe that robots in the future should be more humanized, expressive, emotional and individualistic. Therefore, they designed a new behaviour model for humanized robots, based on Skinner Operant Conditioning theory, and realized the model in WE-4RII robots. The experiment showed that based on OC model, WE-4RII could select a proper behaviour model [13] in accordance with a particular setting autonomously within given behaviour list and learned an interaction skill---shaking hands with human.

A problem laying in the above researches is the following: how should the Skinner OC on machines and robots be realized? Among them, a majority of solutions relies on descriptive language while some adopt conventional artificial neural networks. However, purely descriptive language is not formalized and therefore does not have the ability of generalization; Conventional artificial neural networks cannot reflect the real structure and function of a biological neural system. In response to this question, Professor Ruan Xiao-gang has conducted an in-depth research since 2009, and has been working on building a OC computing model [14] with probabilistic automaton and put forward the concept of Skinner Operant Conditioning Automata (SOCA) though simulating Skinner's pigeons experiment and applying it to the self-learning of two-wheeled self-balancing robots. This method showed good self-learning abilities [15÷17] by enabling the robots to master self-balancing through learning. In 2010, the research team used cerebellar model to build an OC computing method based on the OC automaton, and conducted a bionic experiment on two-wheeled self-balancing robots [18÷20].

The OC automaton that has been built has quick convergence speed but its accuracy of learning is relatively low, which limits the application of OC automata. There are two major reasons leading to the poor learning performance of OC automata: 1) The output of OC automata is a limited and discrete behaviour set. The operant behaviour of the automata is not continuous, resulting in failure in smooth control output and oscillations in output; in addition, in terms of OC self-learning model, the self-learning and adaptation abilities of the learning model are constrained and subject to failure by the limited number of operant behaviours available when the control effects are poor and the change of outside conditions resulting in new behaviour models whose optimal operant behaviours are not in the behaviours set. Therefore, due to the discrete output and limited number of operant behaviour, the OC automata cannot ensure that its amount of control learning is optimal in a nonlinear, time-varying and continuous system. The accuracy of learning and self- adaptation cannot be guaranteed. 2) The number of inward mappings in OC automata is fixed. Among them, there are redundant mapping rules, which reduce the speed of learning. In fact, the human control behaviour is generated by revising a small number of rules to create complex control behaviour instead of large set of rules. Therefore, it is necessary to economize the number of mapping rules to improve the learning performance and self-adaptation abilities in OC learning model.

Increasing the number of operant behaviours can alleviate the problem of lack of output control smoothness but will reduce the learning speed. The solution to this question, other bionic mechanisms such as reinforcement learning, mostly is the adoption of the neural-fuzzy networks [21, 22], which have low convergence speed and instantaneity.

Literature [23] has proposed the Q learning method, which realized the automatic increase and decrease of the number of fuzzy rules, and solved the problem of fixed mapping rules. However, this method selects consequent behaviours in the fuzzy inference system from a fixed set of behaviours, thus resulting in the lack of smoothness in the output control; Literature [24] designed a fuzzy logic system based on reinforcement learning using genetic algorithms based on Q value and online clustering method. The fuzzy logic system can adopt learning rules online and automatically generate fuzzy rules from zero. However, as it adopts genetic algorithms, it is complicated with large amount of computation.

Although the above studies on reinforcement learning cannot solve the problems in OC automata fundamentally, they indicate that fuzzy logic is an effective solution. Fuzzy logic has strong self-learning and adaptation skills, receiving wide interest among researchers. Fuzzy logic system has features including high accuracy, wide application, strong generalization ability and ease of building. It can use a limited size of fuzzy set to describe status and operant behaviour space, adapt to fuzzy descriptions and uncertain knowledge, in line with human thinking model. Therefore, fuzzy logic systems are more apt to combine with bionic learning which stresses the initiative. It is now widely used in bionic self-learning models to tackle the problems in complicated continuous systems. The advantages of adopting fuzzy logic system are summarized as follows: 1) It is capable of smooth output of continuous control; 2) Several successful solutions to the automatic increase and decrease of fuzzy rules; 3) Fuzzy inference system’s fuzzification process is equivalent to the discretization process of the OC automata. Therefore, we can select suitable membership functions to avoid the discrete errors in the discretization of the OC automata; 4) OC automata’s learning process is similar to the fuzzy inference process under the fuzzy control. Therefore, using fuzzy language and fuzzy inference to describe OC automata not only makes the structure clear but also clearly demonstrates the OC automata learning results in the form of a list of rules, showing the accumulation of learning experience of the self-learning model in a direct way.

2 Design of Fuzzy Skinner Operant Conditioning Automaton (FSOCA)

2.1 Mechanism of operant conditioning

The core content of Skinner operant conditioning theory is that by way of learning or training, animals will find their nervous tissue changed. The change results in the connection between certain percept sequence and action sequence, namely the continuous recursive process from "percept" to "action" and again to "percept". The operant conditioning mechanism is shown in Fig. 1.

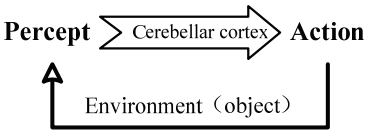


Figure 1 Sketch map of operant conditioning mechanism

The learning control on the basis of operant conditioning mechanism principles mainly consists of three elements: behaviour selection mechanism (choice behaviour based on probability), evaluation mechanism and orientation mechanism. As the core part of learning, the orientation mechanism is used to update behaviour selection strategies. Fig. 2 is the sketch map of learning control mechanism on the basis of operant conditioning principles.

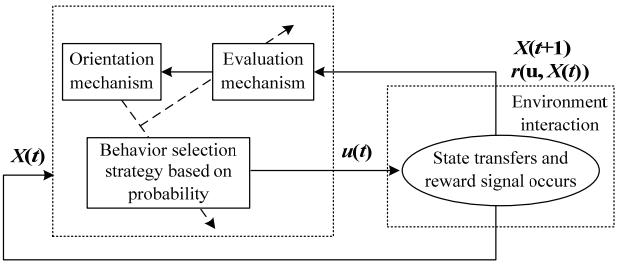


Figure 2 Learning mechanism on the basis of operant conditioning principles

2.2 Structure of FSOCA

The most salient feature of fuzzy control is that it expresses experts' control experience and knowledge as language control rules and then controls the system through these rules. Thus, fuzzy control theory has become a significant branch of intelligent control theory. As both the antecedent and consequent of fuzzy logic system are depicted by natural language variables, which makes it unnecessary to establish precise math models and easy to transform expert knowledge into control signals directly, it has become a significant method in robot control [25]. The fuzzy conditional statement being made up of several linguistic variables, fuzzy inference reflects a certain way of thinking of humans. If fuzzy inference is viewed as the mapping relationship between state space and action space, we can establish the Fuzzy Skinner Operant Conditioning Automata based on fuzzy set theory. FSOCA uses limited fuzzy set to describe conditions and operation behaviour space.

Designed structure of FSOCA is shown in Fig. 3. In the learning model displayed in Fig. 3, the antecedent of each mapping rule corresponds to a fuzzy subset F_{ij} of input space and the consequent is a certain operation behavior $a_j^*(t)$ of the corresponding operation behavior set $A=\{a_k \mid k=1,2,\cdots,r\}$. Therefore, in essence, the learning problem of FSOCA is to seek the optimal decision vector for each mapping rule.

The definition of FSOCA that can be formalized is as follows:

Definition 1 FSOCA is a nine-tuple calculation model: FSOCA $=< x, F, A, \Gamma, f, \varphi, L, H, \Psi >$. Each part is illustrated as follows:

(1) Internal continuous state of FSOCA: $x_i (i=1,2...,n)$, the actual state value of detected control systems. n represents the number of internal continuous state in learning models. By employing the online clustering algorithm on $x(t)$, we can construct the antecedent of FSOCA automatically.

(2) Internal fuzzy state set of FSOCA: $F=\{F_{ij} \mid i=1,...,n; j=1,2,\cdots,N\}$ .As the state antecedent of FSOCA, F emerges as the fuzzy subset after x fuzzification. With respect to the fuzzification or discretization of $x(t)$, the Gaussian function is adopted:

$$F_{ij}(x)=\exp\left\{-\left(\frac{x_i-c_{ij}}{b_{ij}}\right)^2\right\} \tag{1}$$

In this formula, c_{ij} and b_{ij} stand for the center and width of the Gaussian function respectively. j = 1,..., L is the number of clusters, namely the number of mapping rules.

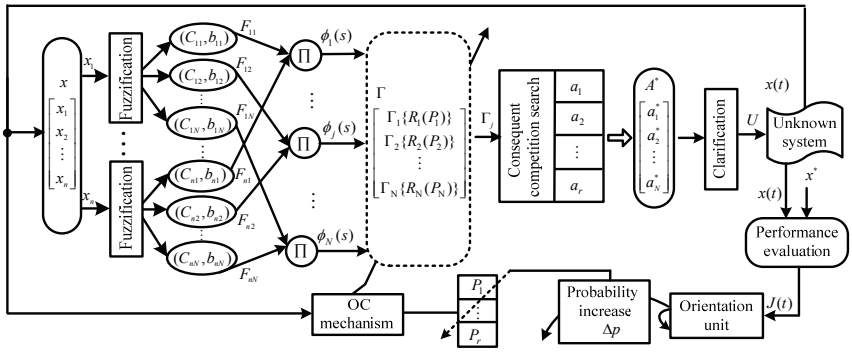


Figure3 Structure of FSOCA

Thus, we obtain the excitation intensity of the jth mapping rule, which is:

$$\varphi_j(x)=\prod_{i=1}^{n}F_{ij}=\exp\left\{-\sum_{i=1}^{n}\left(\frac{x_i-c_{ij}}{b_{ij}}\right)^2\right\} \quad (2)$$

(3) The consequent operation behavior set of FSOCA: $A=\{a_k \mid k=1,2,\cdots,r\}$. a_k stands for the k^{th} available operation behavior and r the number of available consequent operation behavior. The goal of learning is to search for the optimal consequent $a_j^* \in A^*=[a_1^*,a_2^*,...,a_N^*]$ among the consequent operation behavior set A.

As the control signal of systems, the final output of FSOCA is expressed as:

$$U=\frac{\sum_{j=1}^{N}\varphi_j(x)a_j^*(t)}{\sum_{j=1}^{N}\varphi_j(x)} \quad (3)$$

(4) The fuzzy "condition-operation" mapping rule set of FSOCA:

$\Gamma=\{R_j(P_j) \mid P_j \in P=\{P_1,...,P_N\}; R_j \in R=\{R_1,...,R_N\}\}$

which replaces the random "condition-operation set" in FSOCA. In this formula, N is the total number of mapping rules and $R_j(P_j)$ is the j^{th} mapping rule. $p_{jk} \in P_j=(p_{j1},p_{j2},...,p_{jr})$ is the probability value of implementing operation behavior a_k by learning models under the fuzzy state F_{ij}, meeting the requirement of $0<p_{jk}<1$ and $\sum_{k=1}^{r}p_{jk}=1$. Probability distribution $P=\{P_1,\ldots,P_N\}$ plays a role in controlling the random degree in the process of competition and selection by consequent behavior. The mapping rules of FSOCA with competing consequents are as follows:

$R_j(P_j)$:

If $F_{ij}(t)$ Then a is $a_1(t)$ with p_{j1}

or a is $a_2(t)$ with p_{j2}

……

or a is $a_r(t)$ with p_{jr}

It can be seen that the fuzzy "condition-operation" operation rule set of FSOCA resembles the definition of fuzzy rule table in fuzzy inference system. The main difference between the two is that the mapping rule of the former is random and each mapping rule is connected with a certain probability while the fuzzy rule of the latter is definite.

(5) State transition function of FSOCA: $f: F_{ij}(t)\times a_k(t)\rightarrow F_{ij}(t+1)$. The fuzzy state $F_{ij}(t+1)$ at the time of *t+1* is determined by the state $F_{ij}(t)$ at the time of t and the consequent operation behavior $a_k(t) \in A$ and has nothing to do with the state and operation behavior before the time of *t*.

(6) Orientation function of FSOCA: φ. $\varphi_{ik} \in \varphi$ is the orientation value corresponding to the mapping rule of "fuzzy state F_{ij} - consequent operation a_k", meeting the requirement of $\varphi_{ik} \in [0,1]$.

(7) Learning mechanism of FSOCA: $L: \Gamma_j(t)\rightarrow\Gamma_j(t+1)$. Its function is to achieve the optimal selection of consequent behavior.

As learning proceeds, if FSOCA is able to increase the selection probability $p_{jk}(t)$ of large-orientation consequent behavior $a_k(t)$, it means that the consequent of mapping rules tends to select the operation behavior that makes the orientation function minimal. In other words, the learning model has already understood and adapted to the environment and acquired "learning of random mapping rules". Thus, the learning mechanism of FSOCA is as follows:

If the implementation of operation behavior$a$ $k(t)$ results in $F_{ij}(t)\rightarrow F_{ij}(t+1)$ and $\varphi\left(F_{ij}(t+1)\middle|a_k(t)\right)<\varphi\left(F_{ij}(t)\middle|a_k(t)\right)$, then the probability value $p_{jk}\left(a_k(t)\mid F_{ij}(t)\right)$ of implementing operation behavior $a_k(t)$ tends to increase under the premise of fuzzy condition $F_{ij}(t)$ or vice versa.

(8) Operation behavior entropy of FSOCA: $H_j \in H=\{H_1,H_2,...,H_N\}$. $H_j(t)$ is the operation behavior entropy of the *jth* mapping rule in FSOCA, which is

$$\begin{aligned}H_j\left(A(t)\mid F_{ij}(t)\right)&=-\sum_{k=1}^{r}p_{jk}\log_2 p_{jk}\\&=-\sum_{k=1}^{r}p_{jk}\left(a_k\mid F_{ij}\right)\log_2 p_{jk}\left(a_k\mid F_{ij}\right)\end{aligned} \quad (4)$$

When the consequents $a_k(t)$ of all mapping rules have equal possible probability, operation behavior entropy becomes the largest. Operation behavior entropy is used to measure the uncertainty degree of mapping rules which could further measure the amount of information acquired in learning models. In other words, the learning goal of FSOCA is to transform the uncertain consequent of mapping rules to a certain one, enabling mapping rule set to evolve from the unorganized to the organized instinctively or spontaneously under the domination of FSOCA.

(9) Internal parameter vector of FSOCA: $\Psi=[\zeta,\gamma,\eta_1,\eta_2,\phi^*,\varpi,\Delta b_{\min},\Delta c_{\min}]$. ζ,γ is the coefficient related to orientation functions, η_1,η_2 the learning parameter in updated probability formula, c_{ij},b_{ij} the width value and center value of the Gaussian function, φ^* the excitation intensity threshold of mapping rules, ϖ the degree of overlap between clusters and $\Delta b_{\min},\Delta c_{\min}$ the similarity threshold of cluster width and cluster center. These parameters are collectively referred to as internal parameters of FSOCA. The selection of these values has

not only significant impacts on the learning speed and accuracy of learning models but also direct influence on the success of learning.

The basic learning process of FSOCA can be summarized as follows: suppose that the state detected by control systems is x(t) at the time of t, firstly the Gaussian function is employed to conduct the fuzzy processing on x(t) and online clustering algorithm is used to automatically construct the mapping rule antecedent of FSOCA; next, fuzzy subset F_{ij} activates the mapping relationship Γ_j as an activation signal and employs the learning mechanism of operant conditioning to obtain a certain operation behavior $a_k(t)$ in consequent behavior set A of mapping rules. Each operation behavior has a corresponding probability value $p_{jk}(t)$ used to evaluate alternative consequent behavior. Consequent behavior with higher probability value indicates a better learning result and that the frequency of being selected increases in subsequent learning. Then, implement the selected consequent behavior, which contributes to the state transition $F(t) \to F(t+1)$, and calculate new orientation value $\varphi_{jk}(t+1)$. Thus, the variation amount of orientation value $\bar{\varphi}_{jk} = \varphi_{jk}(t+1) - \varphi_{jk}(t)$ is obtained. Ultimately, according to the trend of variation amount of orientation value, the learning mechanism of operant conditioning $L(\bullet)$ is employed to adjust and update the probability vector P_j and reward probability of consequent operation behavior. When new state is activated, repeat this process until the optimal consequent behavior set A^* is learned. Therefore, the essence of FSOCA is to achieve the optimal mapping from fuzzy antecedent state F_j to fuzzy consequent behavior $a_k(t)$.

FSOCA is the result of the fuzzification of Skinner Operant Conditioning Automata. Comparing the two, we can find that their differences are mainly in three ways. Firstly, with regard to the discretization of continuous input state, FSOCA utilizes the fuzzification method which is mature and suitable for actual systems. Secondly, FSOCA outputs continuous smooth control variable. Thirdly, the number of FSOCA mapping rules can be deleted automatically.

3 Design of learning algorithm

3.1 Design of orientation function

Let the orientation function of FSOCA be $\varphi = \{\varphi_{ik} \,|\, i \in (1,2,\cdots,n), k \in (1,2,\cdots,r)\}$. According to the definition of orientation function and given the actual situation of control systems, the design of orientation function should satisfy the following conditions:

① $\varphi_{ik}(t) \in [0,1]$;

② Suppose that the desired system output is x^*, define the output error as $e = x - x^*$. At the time of t and under the premise of discrete state $s_i(t)$, if the selection of operation behavior a_k causes the state transfer $s_j(t+1)$ and reduces error, namely e(t+1)-e(t)<0, it shows that the system has a greater orientation towards the mapping of "state $s_i(t)$ - operation a_k "; otherwise, the orientation is small.

On the basis of these two conditions mentioned above, the expression of the designed orientation function is as follows:

$$\varphi_{ik}(t) = \frac{e^{\gamma J_{ik}(t)} - e^{-\gamma J_{ik}(t)}}{e^{\gamma J_{ik}(t)} + e^{-\gamma J_{ik}(t)}} \tag{5}$$

$J_{ik}(t) = \dot{e}_{ik}^2(t) + \zeta e_{ik}^2(t)$ is equivalent to the system real-time performance indicator under the influence of operation behavior a_k. The error is represented as $e_{ik}(t) = x_i(t) - x^*$, in which x^* represents the desired state value. $\zeta > 0$ is the weight coefficient of error and $\gamma > 0$ the coefficient of orientation function. The orientation function $\phi(t)$ amounts to the transformation of original error measurement value $e(t)$ which is made to range from 0 to 1. According to the orientation function expression designed by formula (5), the relationship between orientation value and orientation quality is that when orientation value approaches 0, the performance of learning models proves to be the best and the orientation reaches the maximum; when the value approaches 1, the performance of learning models is the worst and the corresponding orientation reaches the minimum; when the value lies between 0 and 1, the smaller the value, the better the performance of corresponding model. Therefore, the goal of learning is to make the performance index function approach the minimal.

Note: The orientation function designed here is mainly prepared for the control system. As far as control systems are concerned, error serves as the most direct indicator of reflecting the quality of system performance. So we design the orientation function based on the system error. Also, given that the closer the system error approaches 0, the better the system performance, we define the relationship between the orientation value and the orientation quality as the one mentioned above.

3.2 Design of learning mechanism

Learning mechanism serves to achieve the random mapping $L: \Gamma(t) \mapsto \Gamma(t+1)$. Assuming that $s_i(t)$ is the state at the time of t and the orientation value is $\phi_i(t)$, implement operation behavior $a_k(t) \in A$ according to mapping set Γ of random "condition-operation", after which we observe that $s_j(t+1)$ is the state at the time of t+1 and the orientation value is $\phi_j(t+1)$. Since after implementing operation $a_k(t)$ variation amount $\varphi_j(t+1) - \varphi_i(t)$ of orientation function value can be used to judge the performance of the operation, the design of learning mechanism is as follows according to the Skinner operant conditioning theory:

$$\begin{cases} IF\ \varphi_j(t+1) < \varphi_i(t) \\ THEN \begin{cases} p_{ik}(t+1) = p_{ik}(t) + \Delta_1 \quad a(k) = a_k \\ p_{ik'}(t+1) = p_{ik'}(t) - \Delta_1' \ a(k) \neq a_k \end{cases} \end{cases} \tag{6}$$

The increase part is designed as: $\begin{cases} \Delta_1 = \alpha(\phi)[1 - p_{ik}(t)] \\ \Delta_1' = \alpha(\phi)\, p_{ik'}(t) \end{cases}$

In the formula, $\alpha(\varphi) = \dfrac{1}{1 + \exp\left[-\eta_1 \dfrac{\varphi_i(t) - \varphi_j(t+1)}{\varphi_i(t)}\right]}$;

$$\begin{cases} IF\ \varphi_j(t+1) > \varphi_i(t) \\ THEN \begin{cases} p_{ik}(t+1) = p_{ik}(t) - \Delta_2 \quad a(k) = a_k \\ p_{ik'}(t+1) = p_{ik'}(t) + \Delta_2' \ a(k) \neq a_k \end{cases} \end{cases} \tag{7}$$

The increase part is designed as:

$$\begin{cases} \Delta_2' = \beta(\varphi)\, p_{ik'}(t) \\ \Delta_2 = \beta(\varphi)[\dfrac{1}{r-1} - p_{ik}(t)] \end{cases}$$

In the formula, $\beta(\varphi) = \dfrac{1}{1 + \exp\left[-\eta_2 \dfrac{\varphi_j(t+1) - \varphi_i(t)}{\varphi_i(t)}\right]}$

In the OC learning mechanism formula, $\eta_1 > 0$ and $\eta_2 > 0$ are learning parameters and $\alpha(\phi)$ and $\beta(\phi)$ are learning rate functions which meet the requirement of $0 < \alpha(\phi) < 1$ and $0 < \beta(\phi) < 1$. Adding orientation function ϕ into learning rate functions $\alpha(\phi)$ and $\beta(\phi)$ not only play a role in influencing learning speed but also enable learning models to reflect the orientation characteristics more similar to animals.

From formula (6) and (7), we can see that the excitation probability of random mapping is mainly determined by the variation amount $\varphi_j(t+1) - \varphi_i(t)$ of orientation value. In specific, under the condition of $\varphi_j(t+1) - \varphi_i(t) > 0$, the probability $p(a_k(t) \mid s_i(t))$ of implementing operation behavior $a_k(t)$ in the state of $s_i(t)$ tends to decrease; on the contrary, under the condition of $\varphi_j(t+1) - \varphi_i(t) < 0$, the probability $p(a_k(t) \mid s_i(t))$ of implementing operation behavior $a_k(t)$ in the state of $s_i(t)$ tends to increase. And the greater the variation amount of orientation value, the greater the values of $\alpha(\phi)$ and $\beta(\phi)$, the faster the increase speed of corresponding "good" operation behavior and the decrease speed of "bad" operation behavior; on the contrary, the smaller the variation amount of orientation value, the slower the update speed of probability.

3.3 Design of clustering algorithm

Fuzzy antecedents of FSOCA are based on online clustering algorithm. It is because data is generated during online bionic learning process, so clustering algorithm that automatically generates a certain number of mapping rules is needed; one mapping rule corresponds to one clustering in state space and excitation intensity can be used to examine the extent of how state belongs to corresponding clustering, namely, state $x(t)$ of high excitation intensity is close to the clustering center in geometric space. Therefore, excitation intensity of mapping rules is used in this article as a standard on whether new mapping rules are generated.

Suppose t=0, and excitation state is $x_i(0)$, a new mapping rule is then generated, with that the center and width of its corresponding gaussian function are: $c_{1j} = x_i(0), b_{1j} = b^*$, of which b^* is given in advance. When t=1, the maximum excitation intensity is $\varphi_J = \max\limits_{1 \le j \le L(t)} \varphi_j(x)$ (that is to calculate the extent of how the new input state belongs to every clustering), of which $L(t)$ is the value when t automatically generates mapping rules. If $\varphi_J \le \varphi^*$, a new mapping rule is generated, in which $\varphi^* \in (0,1)$ is the threshold value of excitation intensity of the mapping rule. The center and width of gaussian function of the new mapping rule are designed as follows:

$$b_{\{L(t)+1\}i} = x_i(t) \tag{8}$$

$$c_{\{L(t)+1\}i} = \varpi \cdot \sum_{i=1}^{n} \frac{(x_i - b_{Ji})^2}{c_{Ji}^2} \tag{9}$$

of which, ϖ is the degree of overlapping between two clustering.

As the learning goes on, the number of clustering increases, so does that of mapping rules. In order to reduce the number of mapping rules and save resources for the system, clustering that is highly similar to one another, should be merged. Clustering merging is confirmed through judging old and new subordinate function of input variables, which is based on:

$$\begin{cases} \left\| b_{ji} - b_{j'i} \right\| < \Delta b_{\min} \\ \left\| c_{ji} - c_{j'i} \right\| < \Delta c_{\min} \end{cases} \tag{10}$$

In this formula, $\Delta b_{\min}$ and $\Delta c_{\min}$ are similarity threshold values of the given similar clustering, with j referring to the ordinal number of the clustering j and j' the ordinal number of the clustering j'. If the centers and width of two clusterings are close, then the above inequation is met, so the two clusterings are similar and can be merged, of which the center and width of the clustering are distributed as below: $b_{ji} = \dfrac{b_{ji} + b_{j'i}}{2}$, $c_{ji} = \dfrac{c_{ji} + c_{j'i}}{2}$; otherwise clustering cannot be merged.

3.4 The learning process

Step 1. Initialization: iterative learning steps $t = 0$; sampling time t_s = 0,01 s. Orientation information of operating behavior at the beginning is unknown, so the rate of initial operating behavior is:

$p_{jk}(0) = \frac{1}{r}$ $(j = 1,2\cdots N), (k = 1,2\cdots r)$ of which, N is the number of "condition-operation" mapping rules and r is the number of behavior within collection.

Step 2. Perceive state of the two-wheeled robot and fuzzily process it with gaussian function, and then mapping rules antecedences of FSOCA can be automatically generated through online clustering algorithm.

Step 3. According to probability vector P_j of random mapping Γ_j, output one operating behavior $a_k(t)$ which is randomly chosen from the alternative operating behavior collection A.

Step 4. Receive and analyze response of the two-wheeled robot system to ak(t) and get increment of orientation value $\vec{\varphi}(t+1)$ with the rewarding rate.

Step 5. Conditioned reflex of operation: with $P(t+1) = \Gamma(a(t), \phi(t), P(t))$ and $d(t+1) = \Gamma(w(t+1), z(t+1))$, update the selection rate and the rewarding rate of operating behavior.

Step 6. Recursive transfer: if $\lim_{t\to\infty} p_{jk}(t+1) \approx 1 \,\&\, \lim_{\substack{t\to\infty \\ k\neq k'}} p_{jk'}(t+1) \approx 0$ is met, or certain learning steps are finished, move to **Step 7**. Otherwise, update time variable $t = t+1$, and choose randomly a new fuzzy consequent behavior $a_{k'}(t+1)$ according to updated probability vector $P_j(t+1)$ and repeat "**Step 2 ÷ Step 5**" until the optimal fuzzy consequent collection A^* is achieved.

Step 7. The end.

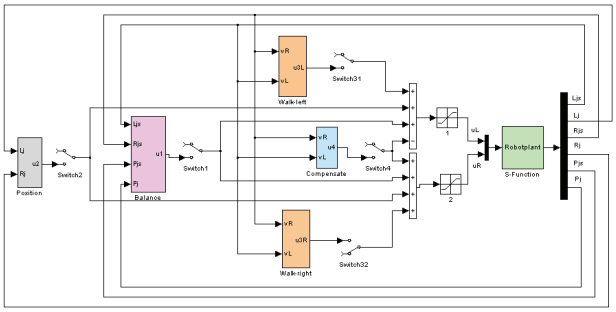


Figure 4 Simulation model of the two-wheeled self-balance robot

5 Result of simulation experiment and its analysis

Build the "exact model" for simulation in the environment of Simulink. As Fig. 4 shows, uL and uR are motor voltage of the left and right wheels of the robot; Ljs, Lj, Rjs, Rjs, Pjs and Pj are angular velocity of the left wheel, the left corner, angular velocity of the right wheel, the right corner, angle velocity of the robot and angle of inclination of the robot. The result of the first four variables multiplied by the wheel radius R is: forward speed of the left wheel, displacement of the left wheel, forward speed of the right wheel, and displacement of the right wheel. No-linear model of the robot is compiled by S-Function. There is a switch respectively connected to attitude balance sub-controller u1, sentinel balance sub-controller u2, walking motion sub-controller u3L, u3R, and compensation controller u4, through which different exercise modes of the two-wheeled robot are switched, of which free self-balance control module is controlled by FSOCA, while others by PID.

5.1 Free self-balance exercise experiment

Fig. 5 shows how the two-wheeled robot achieves free self-balance control. State collection $F = \{(F_i, F_j) | i = 1..., n; j = 1, ..., n\}$ of inclination and angular velocity after fuzzy processing are used as conditions activation signal of FSOCA; under the free self-balance control mode, U that is gotten after clarification of optimal fuzzy operating behavior a^*, is used as the voltage control signal of the two wheels: $u_l = u_r = U$.

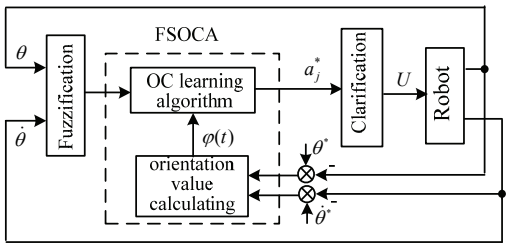


Figure 5 Structure of Free Self-balance Control Based On FSOCA

(1) Setting of Simulation Parameter

Iterative learning steps t = 0; sampling time t_s = 0,01 s; when the robot is in off-line learning, parameter in orientation function is ζ = 0,6, γ = 0,03 and learning coefficient in the updated rate formula is η_1 = 0,01, η_2 = 0,001; when the robot is in online learning, ζ = 0,5, γ = 0,01, η_1 = 0,05, η_2 = 0,005; when the robot is learning, the mapping field can contract to correspond to lower bound value ε = 0,0005 of learning error and excitation intensity threshold value φ^* = 0,0006 of mapping rules. Parameter setting involved in the clustering algorithm is as follows: width of gaussian function is $b^* = 5$; excitation intensity threshold value is φ^* = 0,0006; degree of overlapping between clustering is $\varpi = 0{,}4$; respective similarity threshold value of width and the center of clustering are $\Delta b_{\min}$ = 0,02 and $\Delta c_{\min}$ = 0,02. The initial state of the robot is θ = 0,2 rad, otherwise the value is 0; all the initial operating behavior collection is A={−24, −5, −1, 0, 1, 5, 24}, of which the initial rate of every behavior is $p_{ik}(0) = \frac{1}{7}$, with its corresponding initial operating behavior entropy $H(0) = -\sum_{k=1}^{7} p_{ik} \times \log_2 p_{ik} \Big|_{p_{ik}=\frac{1}{7}} \approx 2{,}81$, and then operating behavior entropy achieves its maximum.

(2) Simulation Result and Its Analysis

Start off-line training first. According to learning steps, train the robot for 30 times with 18 seconds every time. If the robot keeps still in 18 seconds, the training is

successful and training experience this time can be counted and we can move to the next training; if the robot falls, a failure is counted and we move to the next training on the basis of the previous successful training. When the training is finished, there are 28 times' successful training and 2 failures, with 93 % success rate, a bit higher than SOCA.

Fig.6 shows the curve of operating behavior entropy corresponding to stable state (0,0) in 30 times' training. It shows as learning goes on, operating behavior entropy begins to reduce, and after 17 times' training, this value keeps stable and achieves its minimum. On the one hand, the changing condition of operating behavior entropy examines convergence of clustering in this article; on the other hand, at initial state, random mapping control rules of FSOCA are in disorder, but after self-learning, random mapping control rules are in order and can be self-organized and form a positive and orderly rules collection. Compared with SOCA, the change of FSOCA is more smooth, and operating behavior entropy continues to reduce. The reason is that in the process of learning of FSOCA, even though the robot chooses a "bad" operation in the later learning, output can be relatively stable, because control signal of the robot is a weighted sum of fuzzy operating behavior and weight of "bad" behavior is low. That is also the reason why the failure rate of FSOCA is low.

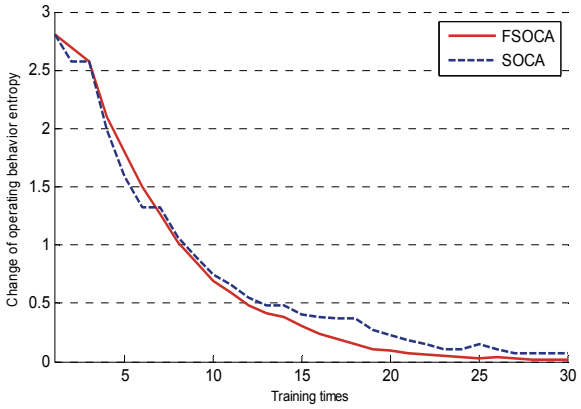


Figure 6 Curve of Information Entropy

Fig. 7 shows the number of fuzzy mapping rules in 30 times' training.

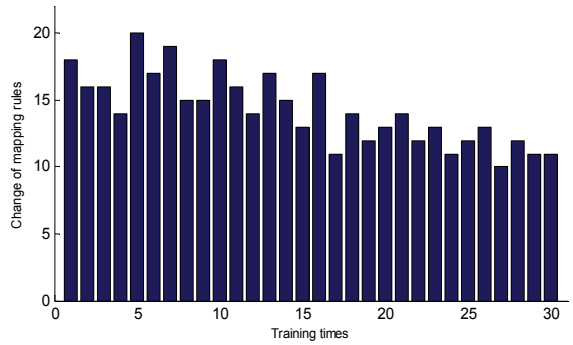
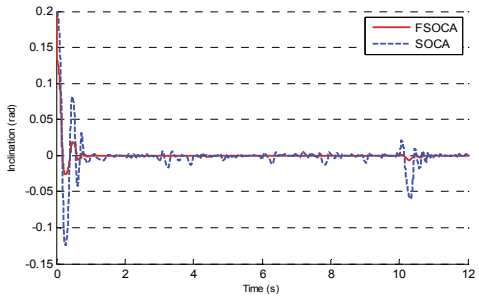


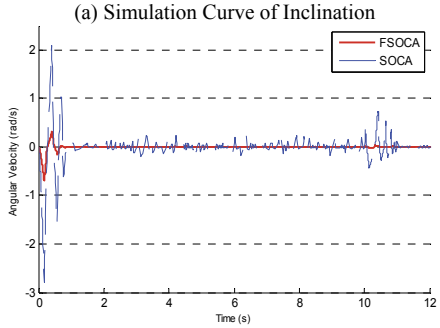
Figure 7 Number of mapping rules generated in every training

The result shows: in all the training, the average number of fuzzy mapping rules in the previous training is 20; while the number becomes 13 after 18 times' training; and after 25 times' training, the number is stable at 11. Compared with $7 \cdot 7 = 49$ mapping rules of SOCA, those of FSOCA are dramatically reduced due to clustering algorithm, which saves for storage space of the computer and improves the speed of learning convergence.

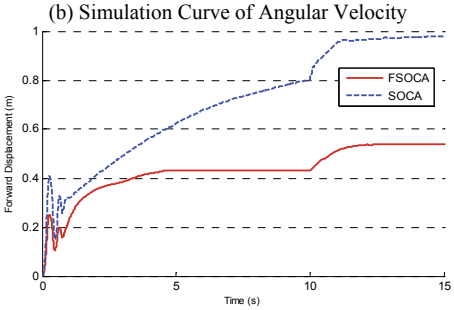
Secondly, move to online learning training. Fig.8 shows simulation curves of inclination, angular velocity, displacement, forward velocity and motor control voltage of the robot.



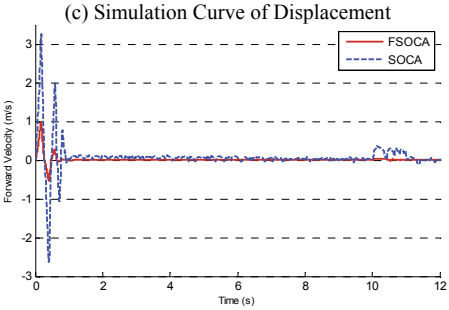
(a) Simulation Curve of Inclination



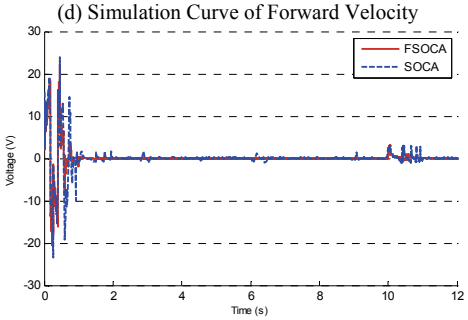
(b) Simulation Curve of Angular Velocity



(c) Simulation Curve of Displacement

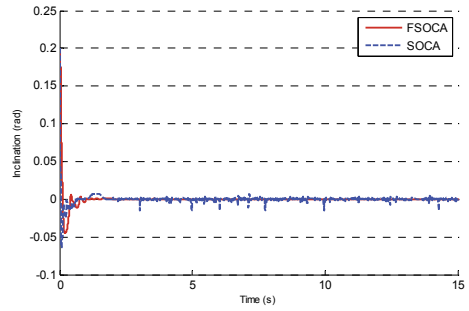


(d) Simulation Curve of Forward Velocity

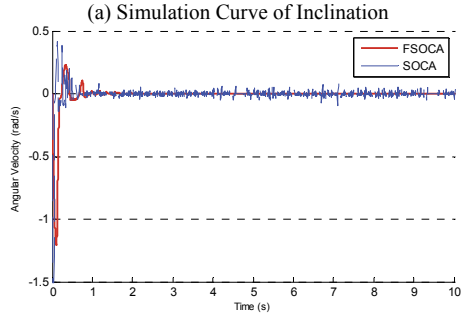


(e) Simulation Curve of Motor Control Voltage

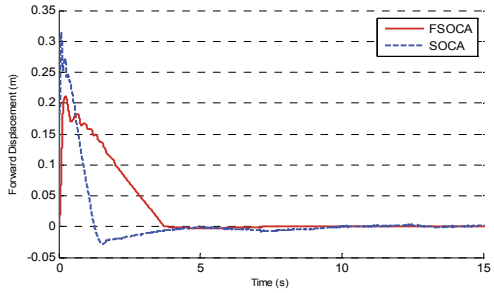
Figure 8 Simulation Result of Free self-balance Control



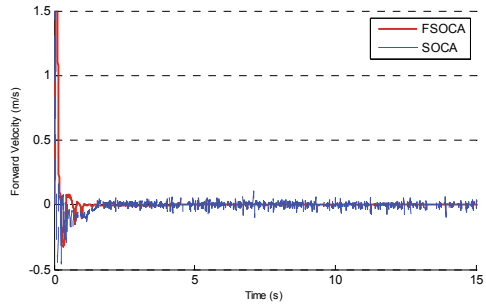
(a) Simulation Curve of Inclination



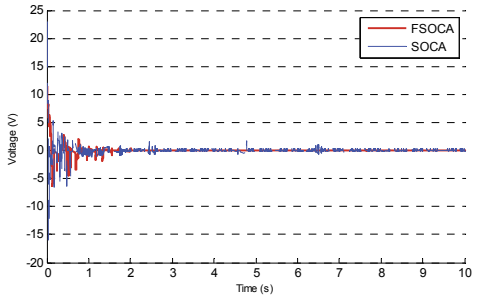
(b) Simulation Curve of Angular Velocity



(c) Simulation Curve of Displacement



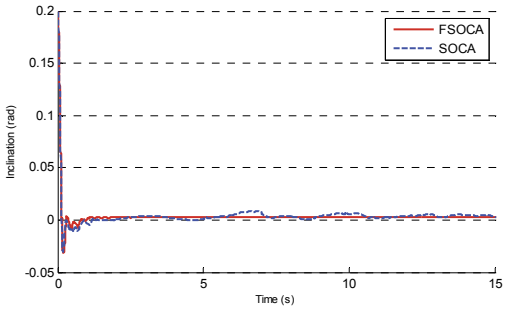
(d) Simulation Curve of Forward Velocity



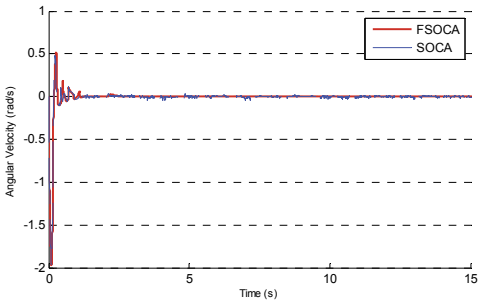
(e) Simulation Curve of Motor Control Voltage

Figure 9 Simulation Result of Point Balance Control

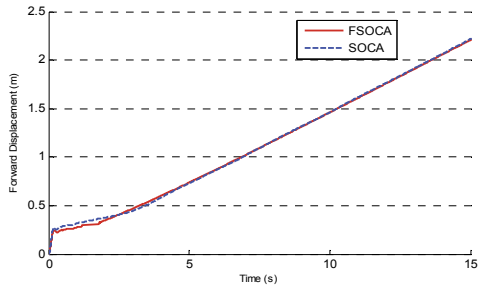
In order to examine anti-jamming capability of FSOCA, the robot is given an impulse interference of 10 in the tenth second. When comparing FSOCA and SOCA, it shows that output of the former is more smooth. That is because FSOCA is fuzzily processed, so the range of output voltage is between [−24, 24], which means continuous voltage can reduce strong jitter of the system. Further comparison shows that in initial learning stage of the former, the balance state can be recovered in one second and overshoot is smaller; in the later stage, learning error is close to 0. Therefore, FSOCA has bigger convergence rate and higher learning accuracy. After interfered, FSOCA can recover to the balance state in 0.5 second after a short jitter. Therefore, compared to SOCA, anti-jamming capability of FSOCA is stronger.



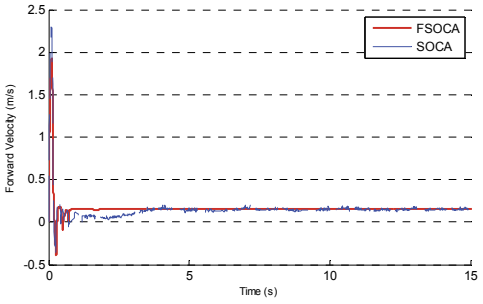
(a) Simulation Curve of Inclination



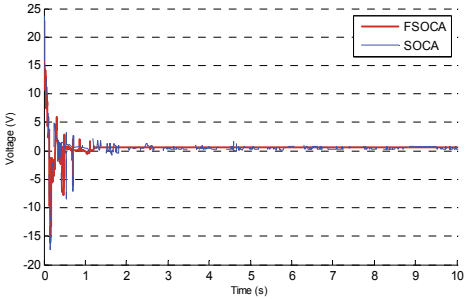
(b) Simulation Curve of Angular Velocity



(c) Simulation Curve of Displacement

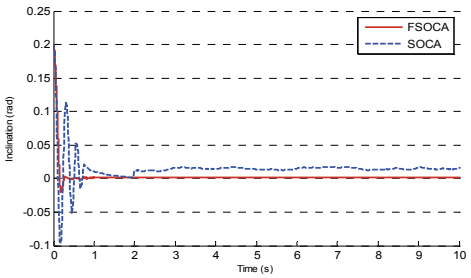


(d) Simulation Curve of Forward Velocity

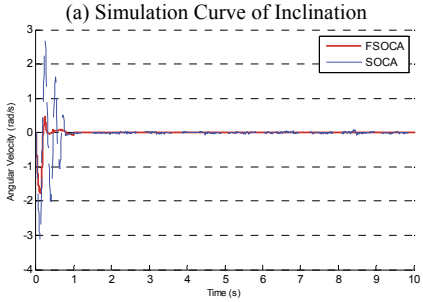


(e) Simulation Curve of Motor Control Voltage

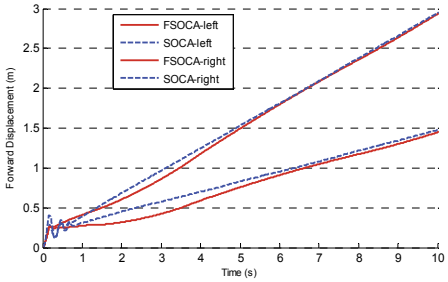
Figure 10 Simulation Result of Straight Moving Balance Control



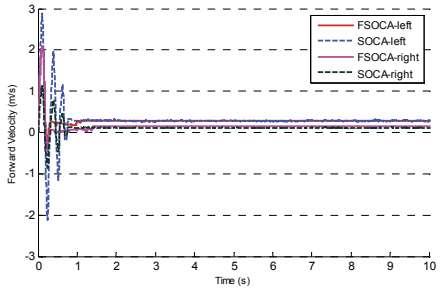
(a) Simulation Curve of Inclination



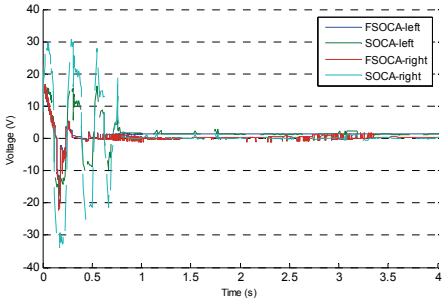
(b) Simulation Curve of Angular Velocity



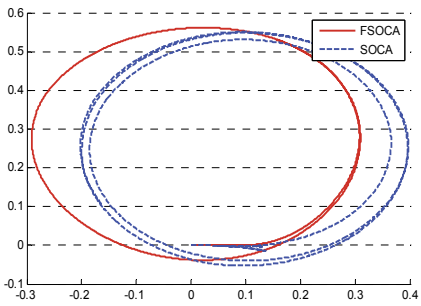
(c) Simulation Curve of Displacement



(d) Simulation Curve of Forward Velocity



(e) Simulation Curve of Motor Control Voltage



(f) Track in The x-y Plane

Figure 11 Simulation Result of Steering Move Balance Control

5.2 Point balance exercise experiment

On the basis of the above learning result, point balance exercise can be achieved by superposition of point balance control modules. Fig. 9 shows the simulation curve of inclination, angular velocity, displacement, forward velocity and motor control voltage of the robot.

Fig. 9 shows that simulation result of point balance control is similar to that of free self-balance control. The robot with FSOCA can recover balance in 1,2 s and stop at target location x = 0 m; compared with SOCA, FSOCA has more smooth curves, bigger convergence rate and higher control accuracy.

5.3 Straight move balance exercise experiment

On the basis of the above learning results, move balance exercise can be achieved by superposition of move exercise balance control modules. Suppose desired speed of the left and right wheel is respectively $v_l = v_r$ = 0,15 m/s, Fig. 10 shows the simulation curve of inclination, angular velocity, displacement, forward velocity and motor control voltage of the robot.

Fig. 10 shows the robot with FSOCA begins to move uniformly after 1 second at the speed of $v_l = v_r$ = 0,15 m/s; inclination does not recover to 0 but keeps at a small angle range θ; motor voltage also keeps at a constant value to ensure the robot can move uniformly. Compared to simulation result of SOCA, output of FSOCA is more smooth and learning speed and accuracy are much higher. Compared to the previous exercise modes, improvement of learning accuracy becomes more obvious.

5.4 Steering move balance exercise experiment

On the basis of straight move balance exercise control, steering move balance exercise can be achieved by setting desired speed of the two wheels respectively at v_{dl} = 0,3 m/s, v_{dr}= 0,15 m/s. Fig. 11 shows the simulation curve of inclination, angular velocity, displacement, forward velocity and motor control voltage of the robot.

Fig. 11 shows steering move is similar to straight move, except that the track is a circle. Compared to SOCA, the curve is more smooth and learning speed and accuracy is higher; the track in less than 1 s becomes a desired circle with radius of 0,3 m.

6 Conclusion

Combing the fuzzy set theory, this paper establishes the FSOCA, the main characteristic of which is that it can be used to depict, simulate and design various self-organizing behaviors of fuzzy and uncertain systems. By integrating fuzzy inference, FSOCA enables learning models to output continuous operation behavior and achieves smooth control. Online clustering method realizes the automatic deletion of fuzzy mapping rules and ensures that the number of fuzzy mapping rules is the most economical. The simulation result in the balance control of two-wheeled robots indicates that as learning proceeds, the selection probability of optimal fuzzy consequent operation behavior gradually approaches 1,

entropy of fuzzy operation behavior tends to be minimal, the number of mapping rules is close to the optimal and relative to SOCA, learning performance is significantly improved. Through imposing pulse interference on robots, FSOCA's ability of anti-interference and fast recovery is verified.

Acknowledgments

The work was supported by Scientific Research Plan Projects for Higher Schools in Hebei Province (No. QN2014313) and Spark Program of Earthquake Sciences Project (No. XH14072).

7 References

[1] Tahriri F.; Mousavi M.; Yap H. J.; Siti Zawiah M. D.; Taha Z. Optimizing the Robot Arm Movement Time Using Virtual Reality Robotic Teaching System. // International Journal of Simulation Modelling. 14, 1 (2015), pp. 28-38. DOI: 10.2507/IJSIMM14(1)3.273

[2] Capi, G.; Nasu, Y.; Barolli, L. et al. Application of genetic algorithms for biped robot gait synthesis optimization during walking and going up-stairs. // Advanced Robotics, 15, 6 (2011), pp. 675-694. DOI: 10.1163/156855301317035197

[3] Karabegović, I.; Karabegović, E.; Mahmić, M.; Husak, E. The application of service robots for logistics in manufacturing processes. // Advances in Production Engineering & Management. 10, 4(2015), pp. 185-194. DOI: 10.14743/apem2015.4.201.

[4] Vilasís-Cardona, X.; Luengo, S.; Solsona, J. and et al. Guiding a Mobile Robot with Cellular Neural Networks. // International Journal of Circuit Theory and Applications. 30, 6(2002), pp. 611-624. DOI: 10.1002/cta.212

[5] Jerbic, B.; Nikolic, G.; Chudy, D.; Svaco, M.; Sekoranja B. Robotic Application in Neurosurgery Using Intelligent Visual and Haptic Interaction. // International Journal of Simulation Modelling. 14, 1(2015), pp. 71-84. DOI: 10.2507/IJSIMM14(1)7.290

[6] Chatterjee, P.; Mondal, S.; Chakraborty, S. A comparative study of preference dominance-based approaches for selection of industrial robots. // Advances in Production Engineering & Management. 9, 1(2014), pp. 5-20. DOI: 10.14743/apem2014.1.172.

[7] Zhang Z.-H.; Hu, C. Multi-Model Stability Control Method of Underactuated Biped Robots Based on Imbalance Degrees. // International Journal of Simulation Modelling. 14, 4(2015), pp. 647-657. DOI: 10.2507/IJSIMM14(4)7.318

[8] Veloso, M. M.; Rybski, P. E.; Chernova, S.; Vail, D. CMRoboBits: Creating an Intelligent AIBO Robot. // AI Magazine, 27, 1(2006), pp. 67-82.

[9] Touretzky, D. S.; Saksida, L. M. Operant Conditioning in Skinnerbots. // Adaptive Behavior. 5, 3/4(1997), pp. 219-247. DOI: 10.1177/105971239700500302

[10] Zalama, E.; Gomez, J.; Paul, M.; Peran, J. R. Adaptive Behavior Navigation of a Mobile Robot. // IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans. 32, 1(2002), pp. 160-169. DOI: 10.1109/3468.995537

[11] Dominguez, S.; Zalama, E.; García-Bermejo, J. G.; Pulido, J. Robot Learning in a Social Robot. // Lecture Notes in Computer Science. 4095(2006), pp.691-702. DOI: 10.1007/11840541_57

[12] Gaudiano, P.; Chang, C. Adaptive Obstacle Avoidance with a Neural Network for Operant Conditioning: Experiments with Real Robots. // IEEE International Symposium on Computational Intelligence in Robotics and Automation / Monterey, CA, 1997, pp. 13-18. DOI: 10.1109/cira.1997.613832

[13] Itoh, K.; Miwa, H.; Matsumoto, M.; Zecca, M. et al. Behavior Model of Humanoid Robots Based on Operant Conditioning. // 5th IEEE-RAS International Conference on Humanoid Robots / Tsukuba, 2005, pp.220-225. DOI: 10.1109/ichr.2005.1573571

[14] Xiaogang, R.; Jianxian, C. Skinner-Pigeon Experiment Simulated Based on Probabilistic Automata. // Global Congress on Intelligent Systems, Xiamen, 2009, pp. 578-581.

[15] Jianxian, C.; Xiaogang, R. OCPA bionic autonomous learning system and application on robot poster balance control. // Pattern Recognition and Artificial Intelligence, 24, 1(2011), pp. 138-146.

[16] Yuanyuan, G.; Xiaogang, R.; Hongjun, S. Operant conditioning learning automatic and its application on robot balance control. // Control and Decision. 28, 6(2013), pp. 930-939.

[17] Xiaogang, R.; Yuanyuan, G.; Hongjun, S. Bionic autonomous learning method based on operant conditioning automata. // Journal of Beijing University of Technology. 37, 11(2011), pp. 1631-1637.

[18] Xiaogang, R.; Jing, Ch.; Naigong Y. Thalamic Cooperation between Cerebellum and Basal Ganglia Based on a New Tropism-Based Action-Dependent Heuristic Programming Method. // Neurocomputing. 93, 15(2012), pp. 27-30.

[19] Hongge, R.; Xiaogang, R. A bionic learning algorithm based on Skinner's operant conditioning and control of robot. // Robot. 32, 1(2010), pp. 132-137. DOI: 10.3724/SP.J.1218.2010.00132

[20] Xiaogang, R.; Jing Ch. On-line NNAC for a Balancing Two-Wheeled Robot Using Feedback-Error-Learning on the Neurophysiological Mechanism. // Journal of Computers. 6, 3(2011), pp. 489-496.

[21] Ning, J.; Jianqiang, Y.; Dongbin, Zh. Reinforcement learning based online neural-fuzzy control system. // Journal of the Graduate School of the Chinese Academy of Sciences. 22, 5(2005), pp. 631-638.

[22] Ma, X. L.; Likharev, K. Global Reinforcement Learning in Neural Networks. // IEEE Trans. Neural network. 18, 2(2007), pp. 573-577. DOI: 10.1109/TNN.2006.888376

[23] Jouffe, L. Fuzzy Inference System Learning by Reinforcement Methods. // IEEE Trans. Syst., Man, Cybern. C. 28, 3(1998), pp. 338-355. DOI: 10.1109/5326.704563

[24] Juang, C. F. Combination of Online Clustering and Q-Value Based GA for Reinforcement Fuzzy System Design. // IEEE Trans on Fuzzy System. 13, 3(2005), pp. 289-302. DOI: 10.1109/TFUZZ.2004.841726

[25] Chen, Y.; Lu, X. F. Fuzzy Control for Radiative Reagent Diluting and Dividing Robot Arms. // Chinese Journal of Scientific Instrument. 30, 2(2009), pp. 330-334.

Authors' address

Jianxian Cai, grades and associate professor
Li Hong, grades and professor
Lina Cheng, grades and lecturer
Ruihong Yu, grades and ranks
Institute of Disaster Prevention,
Department of Disaster Prevention Instrument,
Sanhe Hebei 065201, China
E-mail: cjxlaq@163.com
E-mail: hongli@cidp.edu.cn
E-mail:cln_83@163.com
E-mail: yuruihong97@sina.com