Research on Multi-agent Robot Behavior Learning Based on Fuzzy Neural Network

Jun-ru Wang

Chongqing Vocational College of Transportation, Chongqing 402247, China
xjenxuejn@163.com

Abstract. Human behavior or motion is diverse and complex. In order to design a robot with better sensitivity and better performance, it is necessary to set up multiple control nodes to facilitate the control robot to imitate the trajectory of human motion. However, due to the traditional robot behavior learning method, the control nodes are relatively single, and there is no clear behavior target node for the reference object, which leads to a large deviation in the robot behavior learning trajectory. Therefore, a fuzzy neural network-based Multi-agent robot behavior learning method. This method is based on the fuzzy neural network to control the behavior of robot. By optimizing the learning parameters of robot behavior, it can enhance the search program of behavior learning. According to the multi-level robot behavior learning model, it can identify the master-slave target of reference object and realize a more accurate multi-agent robot behavior learning method. The test results show that compared with the traditional method, the behavior trajectory of the proposed method is basically consistent with the behavior trajectory of the reference object. It can be seen that the method has better performance and meets the research requirements at the current stage.

Keywords: Keywords fuzzy neural network · Multi-agent robot · Behavior learning

1 Preface

With the continuous improvement of science and technology, multi-agent robots appear. Traditional behavior learning methods, through the robot’s intelligent central control unit, simulate human movement and activity. However, with many tests, it is found that the behavior learning of this method has a large deviation, and the robot’s behavior trajectory does not conform to the actual situation. Therefore, a multi-agent robot behavior learning method based on fuzzy neural network is studied to solve this problem. This method can achieve more accurate behavior learning by strengthening the setting of control node and defining the master-slave motion node of reference target. This method not only enhances the working efficiency of multi-agent robot, but also provides a research idea for the development of science and technology [1, 2].
2 Multi-agent Robot Behavior Learning Method Based on Fuzzy Neural Network

2.1 Robot Behavior Control Based on Fuzzy Neural Network

According to the fuzzy theory of fuzzy neural network, the behavior of robot under the control of fuzzy neural network model is studied. The robot’s fuzzy rule \( i = (1, 2, \ldots, n) \) is of the form: If \( \beta_i(t) \) is \( K_{1i} \) and \( \beta_r(t) \) is reliable data in the \( K_{ir} \) set, then:

\[
\hat{y}(t) = -\beta_i(t)D_iy(t) + (W_i + \Delta W_i)f(y(t)) + (U_i + \Delta U_i)
\]

\[
f(y(t - \tau)) + I + V_i u(t)
\]

(1)

Where: \( \beta_i(t) \) represents the fuzzy set, \( i = 1, 2, \ldots, r \) represents the known variable, and \( r \) is the number of fuzzy rules. \( u(t) \in M^p \) represents the input value of the input robot; matrix \( D_i = \text{diag}\{d_{i1}, d_{i2}, \ldots, d_{in}\} \) is a positive definite diagonal matrix, and \( W_i, U_i \in M^{n \times n} \) and \( V_i \in M^{n \times p} \) are known constant matrices. The matrices \( \Delta W_i \) and \( \Delta U_i \) are parameter uncertainties and satisfy the following conditions:

\[
[\Delta W_i, \Delta U_i] = K_i F(t) [Q_{1i}, Q_{2i}]
\]

(2)

In the formula: \( Q_{1i}, Q_{2i}, K_i \) represent known constant matrices of appropriate dimensions. Gaussian fuzzy neural network is a fuzzy neural network composed of product reasoning rules, single-valued fuzzy generators, Gaussian membership functions, and central average anti-fuzzifier [3]. From this, the output of a Gaussian fuzzy neural network can be given as:

\[
\tilde{y}(t) = \sum_{i=1}^{m} \omega_i[\beta(t)] - D_i y(t) + (W_i + \Delta W_i)f(y(t)) \]

\[
+ (U_i + \Delta U_i)f(y(t - \tau)) + I + V_i u(t) / \sum_{i=1}^{m} \omega_i[\beta(t)]
\]

(3)

According to the formula (1-3), the output results are as follows:

\[
y(t) = \sum_{i=1}^{n} \sigma_i[\beta(t)] \cdot \lambda_i / \sum_{i=1}^{n} [\Delta W_i, \Delta U_i]
\]

(4)

Formula: \( \lambda_i \) represents the membership of state \( \beta(t) \) to fuzzy set \( K_{ir} \) in rule \( i \), and the following conditions are met:

\[
\lambda_i \geq 0, \sum_{i=1}^{m} \sigma_i[\beta(t)] = 1
\]

(5)

In the formula: \( \sigma_i \) represents the network control parameter of rule \( i \). The control program is set up according to the above formula to implement a fuzzy neural network model to control the behavior learning mode of a multi-agent robot [4].
2.2 Optimization of Robot Behavior Learning Parameters

The fuzzy neural network based on behavior learning is a simple and effective network structure for multi-agent robot behavior learning, but the fixed behavior control parameters are difficult to adapt to the changing external environment. Therefore, particle swarm optimization algorithm is used to modify the control nodes in the network through the implicit parallel operation ability and good adaptive ability. At the same time, the control nodes in the network also provide a better population for the particle swarm optimization algorithm, so that the algorithm can carry out more biased control node search.

In the problem of using particle swarm optimization algorithm to optimize the control parameters of robot behavior learning, each particle is regarded as an independent individual in the search space, and the position vector of each particle corresponds to the solution of an optimization problem. Each particle has a speed that determines the direction and speed of the robot’s behavior. The performance of all particles is evaluated by the fitness value of an optimized function [5].

Position of the $i$th particle can be expressed as a vector $w_i = (w_{i1}, w_{i2}, \ldots, w_{iD})$, and the speed of the behavior can be expressed as a vector $s_i = (s_{i1}, s_{i2}, \ldots, s_{iD})$. In the process of controlling the behavior of the robot, the optimal position found by the $i$th particle is $p_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, and the optimal position of all the control nodes obtained by the particle is $p_k = (p_{k1}, p_{k2}, \ldots, p_{kD})$. Each particle updates its speed and position according to the following formula:

$$s_i(n + 1) = s_i(n) + x_1 b_1 [p_i - w_i(n)]$$
$$+ x_2 b_2 [p_k - w_i(n)] w_i(n + 1)$$
$$w_i(n) + s_i(n)$$ (6)

In the formula: $i = 1, 2, \ldots, m$ represents different particles; $n$ represents the number of iterations, that is, the number of positioning steps of the particles; $x_1$ and $x_2$ represent acceleration coefficients or learning factors, respectively; $b_1$ and $b_2$ represent random numbers with a variation range of $[0, 1]$. According to the update result of the above formula, set the optimization process of the robot behavior learning control parameters, as shown in Fig. 1 below.

Given the appropriate optimization weight, it can balance the local and global search ability, so as to reduce the number of iterations, so as to control the robot behavior learning faster. Generally, the smaller one can improve the local search ability; the larger one can speed up the convergence speed, and the dynamic one can obtain better optimization results than a fixed value. Therefore, adjusting the size can achieve the goal of balancing the local search control ability and convergence speed [6]. There are two ways to achieve dynamic changes: one is to change linearly in the search process of the algorithm; the other is to dynamically change according to a measure function corresponding to the algorithm performance. This optimization uses the first method, and its expression is as follows:

$$\hat{\omega}_i = \omega_{\text{max}} - n \frac{\omega_{\text{max}} - \omega_{\text{min}}}{n_{\text{max}} s_i(n + 1)}$$ (7)

In the formula: $\omega_{\text{max}}$ and $\omega_{\text{min}}$ represent the maximum and minimum inertia weights; $n$ represents the current number of iteration steps; $n_{\text{max}}$ represents the total number
of iterations. The above formula is used to optimize the control parameters of robot behavior learning and enhance the standard and reliability of behavior learning.

2.3 Establishing a Multi-level Robot Behavior Learning Model

According to the optimized parameters, a multi-level robot behavior learning model is established to control the multi-agent robot behavior learning trajectory.

The hierarchy of learning model refers to the fact that the actual learning program can be divided into several subprograms, which can be subdivided into several lower level subprograms. Subprograms of the same level often have similar functions. Complex programs are characterized by their emergence, which reflects the non-additive and holistic nature of the program from low-level features to high-level features, which are unique to high-level but not at low-level. Different levels of programs reflect the emergence of different properties. High level subprograms have more complex forms of motion and program characteristics. Complex programs are hierarchical, which is an important theoretical basis for establishing behavioral models. Analytic Hierarchy Process provides a mathematical method for quantitative analysis for program analysis and design. When modeling a complex program, the program is first decomposed into several layers, and then the organization is integrated from the lower layer to the higher layer to form the whole program, and a program hierarchy model is established [7].
For general decision-making problems, it can usually be described by a three-level structure. The highest level is the overall goal of the program, and the coarse-grained quantization unit is used to describe the system characteristics. The quantization unit describes the model of each subroutine; the lowest layer is the various solutions used to solve the problem, and the fine-grained quantization unit is used to describe the lowest-level subroutine model. The layers are organized through association variables to form a hierarchical model of complex programs, as shown in Fig. 2.

![Fig. 2. Hierarchy model](image)

Some of the environments that multi-robot programs face are often unknown, dynamic, and unstructured. At the same time, due to the limitations of the robot’s own sensors and the presence of environmental noise, each robot in the group will encounter each other during the task to some unpredictable triggers. When a robot program performs a learning task, the group behavior of the robot through local interaction will be very complicated, and the evolution process of the robot group behavior is a random process. Therefore, according to the robot group behavior, the multi-level robot behavior learning model is established, mostly using the Markov property and random probability calculation method to derive the group behavior learning model, the fractal modeling method, as shown in Fig. 3 [8].

Because the robot population behavior has Markov characteristics or approximate Markov characteristics, the state of the program at any time depends only on the state of the previous moment, independent of the historical state, and its dynamic characteristics can be represented by the following probability distribution:

\[
P_{aa'} = P(A_{t+1} = \hat{\omega}_{i}a' | \hat{\omega}_{i}a_{t})
\]  

(8)

In the formula: \(P_{aa'}\) is the transition probability from state \(a\) to state \(a'\); \(a_{t}\) is the state at time \(t\); \(A_{t+1}\) is the state at time \(t + 1\).

Both fractal theory and program hierarchy reveal the multi-level and multi-dimensional connection between the whole and the part of complex program, while fractal emphasizes the similarity between the whole and the part. Because complex programs have both diversity and self-similarity, fractal theory can be used to model
complex programs. Fractals provide a new idea and method for modeling complex programs. So far, the establishment of a multi-agent robot behavior learning model has been realized.

2.4 Identify Objectives and Achieve Behavioral Learning

Use the behavioral learning model above to identify learning targets, extract target features through local feature point detection and descriptor analysis. This recognition method describes the local features of the relationship between the master and slave feature points, and constructs feature descriptors by using different feature point gradients, color attributes and location relationships. The proposed feature point extraction algorithms have different performances in terms of complexity, repetition, and lighting robustness, and can be divided into strong feature points and weak feature points. Among them, the strong feature point is more stable and robust than the weak feature point, but its algorithm complexity is also higher than the weak feature point. In addition, in the same learning data, the number of strong feature points is usually less than that of weak feature points [9].

The strong feature point $P_1$ (main point) and the weak feature point $P_2$ (auxiliary point) of the learning object are extracted separately. If there are auxiliary point sets in the neighborhood of the main point $P_1^i$ that exceed the threshold value, this area is a feature detection area. Figure 4 is a schematic diagram of the effect of extracting the feature points from the main points. In the figure, the blue squares are Harris main feature points and the red circles are fast auxiliary points. It can be seen that the number of main points is less than the number of auxiliary points, and there are auxiliary points exceeding the threshold number in this area, so this area is a feature detection domain.

The local feature descriptor of the gesture is constructed using the master-slave points. According to the analysis of Fig. 4, there are many attributes of the primary and secondary points in the detection area, including the color information of the primary and secondary points, the gradient, and the relationship between the secondary points and the position of the primary points. Target recognition is carried out by using the
local descriptor method for discrete sampling, and the selected discrete sampling mode is shown in Fig. 5 below.

The detection area sampling mode $PS = \{P_i\}$ is given, where $PS$ is the local area discrete coordinate point set with the origin $(0, 0)$ as the center. The above image is a sampling mode with a radius of 18 from the center and an interval of 3. The sampling coordinates of each detection area are generated by taking the coordinates of feature points as offsets. In order to achieve rotation invariance and give the main direction of the detection area, the sampling mode point $P_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$ can be mapped to the corresponding main direction position according to the following formula.

$$P_i = \begin{bmatrix} \cos \alpha_P & -\sin \alpha_P \\ \sin \alpha_P & \cos \alpha_P \end{bmatrix}$$  \hspace{1cm} (9)
In the formula: \( \alpha_p \) represents the main direction of the current region for all strong and weak feature points. The pixel values corresponding to the sampling points in the main direction mode constitute \( M_{PS} \) of the region, that is, the number-dimensional feature descriptor \( N_{PS} = [q_{PS}] \) of the sampling mode point set. To transform the current feature into a binary feature \( K_{PS} = [k_{PS}] \) for accelerated matching:

\[
\begin{aligned}
   k_{PS}^i &= h_k \left( q_{PS}^i, \bar{q}_{PS} \right) \\
   h_k(x_i, y_i) &= \begin{cases} 
      1, & (x \geq y) \\
      0, & (x < y) 
   \end{cases}
\end{aligned}
\]  

(10)

In the formula: \( \bar{q}_{PS} \) is the average value of \( N_{PS} \); \( x_i \) and \( y_i \) are the horizontal and vertical coordinates of the mapping node; \( h_k(*) \) is the identification function. So far, based on the above, a multi-agent robot behavior learning based on fuzzy neural network has been realized [10].

### 3 Simulation Test Experiments

In order to test the reliability of the proposed robot behavior learning method, a simulation test experiment is proposed to analyze the behavior learning trajectory of the multi-agent robot under the proposed method. In order to make the experimental results have obvious differences, and take the traditional robot behavior learning method as the control target, according to the experimental results, the learning differences between the two methods are analyzed.

Based on the simulation of the differential motion of the end of the robotic arm as the basic motion operation, and the combination of linear motion and spiral curve motion as experimental examples, the trajectory generation effect of multi-agent robot behavior learning under the control of the two methods was tested. The combined linear differential motion simulation effect diagram is shown in Fig. 6 below.
Two methods are used to control the simulation manipulator in the figure above and start to perform the behavior learning task. The Fig. 7 below is the test results of the learning trajectory of the combined linear differential motion under the two learning methods.

(A) Test results of the proposed method

(B) Test results of traditional methods

Fig. 7. Comparison results of combined linear motion learning trajectories
According to the above two experimental results, we can see that the behavior learning trajectory of the proposed robot learning method is very similar to that of the reference object; However, there is a large deviation between the behavior learning trajectory of the traditional method and the motion trajectory of the reference object. In order to ensure the universality of the experimental results, the curve motion of the reference object is studied. Figure 8 is a schematic diagram of the curve motion simulation effect.

Fig. 8. Effect diagram of curve motion simulation

Based on the above conditions, Fig. 9 below shows the test results of the learning trajectory of curve movement under the two learning methods.

It can be seen from the motion trajectories of the above two groups of pictures that the control effect of the proposed method is better than that of the traditional method. Based on two experimental tests, it can be seen that the behavior learning method of the multi-agent robot studied in this study has better control effect.

In order to further verify the effectiveness of this method, the behavior learning trajectory control accuracy of this method and the traditional method is compared, and the comparison results are shown in Fig. 10.

According to Fig. 10, the behavior learning trajectory control accuracy of this method is higher than that of traditional behavior learning trajectory control, which shows that the behavior learning trajectory control effect of this method is better.
(a) Test results of the proposed method

(b) Test results of traditional methods

Fig. 9. Comparison results of curve motion learning trajectories

Fig. 10. Comparison of control accuracy of behavior learning trajectory
4 Concluding Remarks

The robot behavior learning method proposed in this paper solves the problem of large deviation of learning trajectory of traditional methods by strengthening the control of robot nodes and improving the accuracy of robot operation behavior. However, there is still a certain degree of error in the behavior learning method proposed this time. In future research, the error should be further controlled to a smaller range.

References

1. Jin, J.C., Ma, X.L.: A multi-objective agent-based control approach with application in intelligent traffic signal system. IEEE Trans. Intell. Transp. Syst. 20(10), 3900–3912 (2019)
2. Zhang, F., Li, N., Yuan, R., et al.: Robot path planning algorithm based on reinforcement learning. Huazhong Kexue Ban (Ziran Kexue Ban)/J. Huazhong Univ. Sci. Technol. (Nat. Sci. Ed.) 46(12), 65–70 (2018)
3. Baklouti, N., Abraham, A., Alimi, A.M.: A beta basis function interval type-2 fuzzy neural network for time series applications. Eng. Appl. Artif. Intell. 71, 259–274 (2018)
4. Xu, B., Zhang, X.P., Liu, L.Q.: The failure detection method of WSN based on PCA-BDA and fuzzy neural network. Wirel. Pers. Commun. 102(4), 1–11 (2018)
5. Zhang, A., Ma, S., Li, B., et al.: Tracking control of tangential velocity of EEL robot based on iterative learning control. Jiqiren/Robot 40(6), 769–778 (2018)
6. Mizanoor Rahman, S.M.: Cyber-physical-social system between a humanoid robot and a virtual human through a shared platform for adaptive agent ecology. IEEE/CAA J. Automatica Sinica 5(1), 190–203 (2018)
7. Kim, T., Kim, J.-W., Lee, K.: Detection of sleep disordered breathing severity using acoustic biomarker and machine learning techniques. Biomed. Eng. Online 17(1), 16 (2018)
8. Chi, W.Q., Liu, J.D., Hedeyeh, R.-T., et al.: Learning-based endovascular navigation through the use of non-rigid registration for collaborative robotic catheterization. Int. J. Comput. Assist. Radiol. Surg. 13(6), 855–864 (2018)
9. Goodfellow, Ian, McDaniel, Patrick, Papernot, Nicolas: Making machine learning robust against adversarial inputs. Commun. ACM 61(7), 56–66 (2018)
10. Krempel, R., Kulkarni, P., Yim, A., et al.: Integrative analysis and machine learning on cancer genomics data using the Cancer Systems Biology Database (CancerSysDB). BMC Bioinform. 19(1), 156 (2018)