The 6th Annual IEEE International Conference on
Cyber Technology in Automation, Control and Intelligent Systems
June 19-22, 2016, Chengdu, China

*Brain-Inspired Strategy for the Motion Planning of Hyper-Redundant Manipulators**

Liangliang Zhao, Jingdong Zhao*, Hong Liu
State Key Laboratory of Robotics and System
Harbin Institute of Technology
Harbin, China
E-mail: zhaojingdong@hit.edu.cn, zhaoliangliang0619@126.com

Abstract—**The main challenge of motion planning for a hyper-redundant manipulator is to implement a modular structure ensure real time and high performance of the control system. In this research, we present a strategy to deal with the motion planning problem of a hyper-redundant manipulator, include uncertain time delay to the control system and obstacle avoidance. Similarly to the principles of motor control in human brain, we extract primitive motions from a batch of motion data of a hyper-redundant manipulator, and reprogram the complex motions by the sequence of combinations of primitive motions. Based on the neural network algorithm, we present the simulation results of training experiment and testing experiment. All the simulations have confirmed that the proposed control strategy provides remarkable efficiency in motion planning of hyper-redundant manipulators.**

Keywords—hyper-redundant manipulators; human brain; motion planning; primitive motion

I. INTRODUCTION

Hyper-redundant manipulators are very useful in the highly constrained environments, which have a large or infinite degree of kinematic redundancy and greater robustness than ordinary manipulators [1-2]. In 1972, Hirose and his team invented a world-first snake-like manipulator, which was named Active Cord Mechanism [3]. To our knowledge, hyper-redundant manipulators have been implemented in a complex and changeable environment, such as earthquake on-site searching work and nuclear plant radiation detecting work.

Hyper-redundant manipulator has much more degrees of freedom (DOFs) than ordinary manipulators. Motion planning is still a great challenge. As the number of DOFs increasing, the delayed effects of the control system become large. We need to introduce an effective and simple algorithm for the motion planning to decrease or eliminate the time delay of the control system.

For motion planning of hyper-redundant manipulators, several strategies have been developed. N. Shvalb presented a novel probabilistic algorithm for real-time motion planning of a hyper-redundant robot in the configuration space [4]. Ka-Wai Kwok introduced a real-time navigation and dimensionality reduction scheme to control an articulated snake robot under dynamic active constraints [5]. Chaohui Gong proposed an approach to infer the tilt angle of the slope, on which a snake robot is moving, based on the estimated state of its virtual chassis. And they found an optimal policy which maximizes the traveling speed of sidewinding based on the inferred slope angle [6].

Brain-inspired strategy for motion planning is that complex motions can be segmented into multiple primitive motions, then the primitive motions can build up complex motions [7-8]. In this way, humans predominate the primitive motions by imitating the actions of others, the primitive motions could be the most fundamental of all human movements in daily life [9]. Humans learn complex motions through combination of primitive motions in a specific sequence. So we can control primitive motions instead of joint angle values when motion planning of hyper-redundant manipulators. Reference [10] proposed a modular structure to control the human like movements of a robot in a way similar to which human brain does to perform the motor control. Reference [11] employed wavelet neural network as a tool for solving direct kinematics of hybrid robot.

This paper presents a strategy for the motion planning of a hyper-redundant manipulator by combination of the primitive motions. In the Section 1, we introduce the structure feature of the hyper-redundant manipulator, and establish the forward kinematic model based on Denavit-Hartenberg (D-H) method. Then, on the basis of forward kinematic model, we indentified of primitive motion modules, include direction module, bend module and location module. In the Section 4 and Section 5, we proposed a motion planning methodology based on the Artificial Neural Network(ANN), and present the simulation results of training experiment and testing experiment. The simulation results confirm that the brain-inspired control strategy provides remarkable efficiency in the motion planning.

II. DESCRIPTION OF THE HYPER-REDUNDANT MANIPULATORS

A. *Structure Feature of Hyper-redundant Manipulators*

The hyper-redundant manipulator generally constitutes of a number of the same or similar unit modules, and has one fixed base like industrial robots. In this article, it consists of seven same unit modules, each of the unit module includes a pitch joint and a roll joint. The 3-D view of the hyper-redundant

*Corresponding author: Jingdong Zhao, E-mail: zhaojingdong@hit.edu.cn manipulator and the details of the unit module are shown in Fig. 1 and Fig. 2.

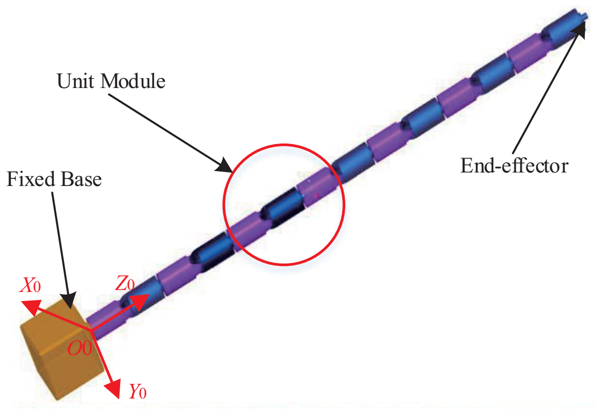

Fig. 1. The 3-D view of hyper-redundant manipulator

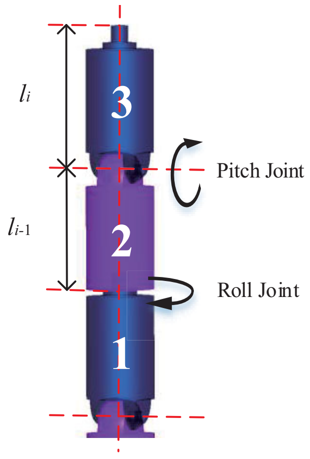

Fig. 2. The 3-D view of unit module

From Fig.1, the hyper-redundant manipulator consists of one fixed base, 7 unit modules, one end-effector. And it has 14 DOFs.

From Fig. 2, as the most basic unit of the hyper-redundant manipulator, the unit module consists of one pitch joint and one roll joint. The pitch joint can be flexible to rotate in any direction. The volume of the pitch joint workspace is a half of spherical surface. And it can be parametrized in the general form:

$$\begin{cases} x = l_i \sin\theta_{12} \sin\theta_{23} \\ y = l_i (1+\cos\theta_{23}) \\ z = l_i \cos\theta_{12} \sin\theta_{23} \end{cases} \tag{1}$$

Where l_i is the connecting rod length, θ_{12} is the rotation angle between the first link and the second link, θ_{23} is the rotation angle between the second link and the third link, x, y and z are the end-effector coordinate values of the connecting rod 3(Fig. 3), which relative to the coordinates of pitch joint. The movement range of θ_{12} is -180° ~ +180°, θ_{23} is -90° ~ +90°, l_1=l_3=…=l_{13}=35mm, l_2=l_4=…=l_{12}=37mm and l_{14} =42mm, the dimensions for the manipulator are 509mm.

B. *Kinematic Model of Hyper-redundant Manipulator*

The structure of the hyper-redundant manipulator is very complicated, and its kinematic model should not be affected by the state change of the joint diameters. Therefore, it can be simplified into a linkage mechanism. The most common way of modelling forward kinematic is D-H method [12]. In our article, we modify the D-H modelling from one-unit module into the whole model of the hyper-redundant manipulator. The D-H coordinate system of the hyper-redundant manipulator is shown in Fig. 3.

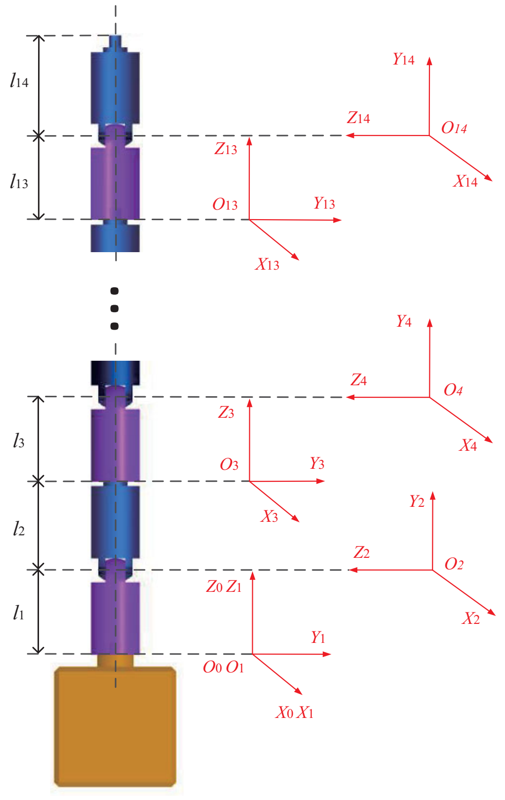

Fig. 3. D-H coordinate system of the hyper-redundant manipulator

Base on the D-H coordinate system, we can get the D-H parameters as shown in Table 1.

TABLE I. D-H PARAMETERS OF EACH MODULE

Module Number	D–H Parameter			
	α_i	a_i	d_i	θ_i
1	0	0	0	θ_1
2	π/2	0	l_1	θ_2
3	−π/2	0	l_2	θ_3
4	π/2	0	l_3	θ_4
⋮	⋮			
13	−π/2	0	l_{12}	θ_{13}
14	π/2	0	l_{13}	θ_{14}

Where α_i is the angle of rotation from the Z_i axis to the Z_{i+1} axis, a_i is the distance from the Z_i axis to the Z_{i+1} axis, d_i is the distance from the X_{i-1} axis to the X_i axis, θ_i is the angle of rotation from the X_{i-1} axis to the X_i axis.

According to the rule of D-H transform coordinates, we can establish the transformation matrix from frame i-1 to i, and it can be given as:

$$ {}^{i-1}_{i}T = \mathrm{Rot}(x,\alpha_{i-1})\mathrm{Trans}(\alpha_{i-1},0,0)\mathrm{Rot}(z,\theta_i)\mathrm{Trans}(0,0,d_i) \quad (2)$$

Where *Rot* represents the rotation coordinate transformations, *Trans* represents the translation coordinate transformations. The motion between two frames can be obtained based on the Eqs. (2) and the D–H parameters in table 1, the transformation matrix of ${}^{i-1}_{i}T$ can be expressed as:

$$ {}^{0}_{1}T = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, i=1 \quad (3)$$

$$ {}^{i-1}_{i}T = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ 0 & 0 & -\sin\alpha_{i-1} & -l_i\sin\alpha_{i-1} \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$, i = 2,3,...,14$

III. Identification of Primitive Motion Modules

On the basis of the kinematic model of hyper-redundant manipulator, we introduce a strategy which can be used to identification of primitive motion modules. The motion of a hyper-redundant manipulator can be divided into the motion of one or more pitch joints or roll joints. Due to the manipulator has many more DOFs, there are a lot of possible primitive motion modules can be selected.

When we identified of a primitive motion module, make sure that no interference will be caused by it. In consideration of the important function of the real time and high performance of the control system, there is a limit to the number of modules that we introduced. According to the D-H coordinate system, we can define the motion direction by rotating the first roll joint which connected to the fixed base. By controlling the movement of other roll joints, the hyper-redundant manipulator can adjust its position along the Z_0 direction. And it can avoid the different types of cylindrical obstacles by adjusting the angle of pitch joints.

In this study, the hyper-redundant manipulator is divided into three type of primitive motion modules, include direction module, bend module and location module as shown in Fig. 4. By defining the primitive motion module in this way, the efficiency of control operation is improved. And this method can make hyper-redundant manipulator effectively to avoid the cylindrical obstacle.

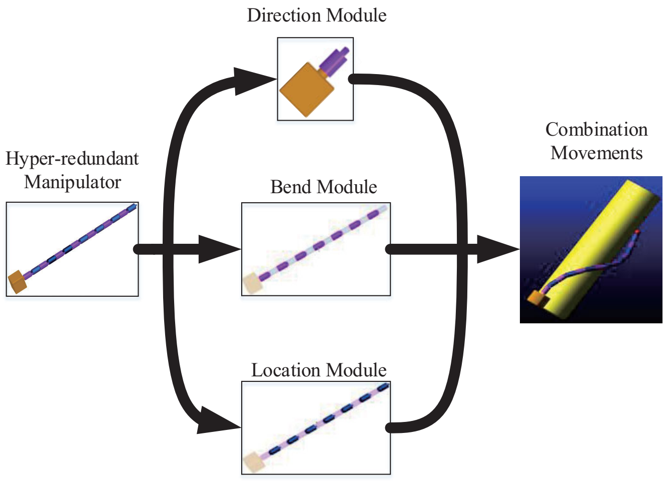

Fig. 4. Three type of primitive motion modules

A. Direction module

The direction module consists of the fixed base and one roll joint. The major function of it is to determine the movement orientation of the hyper-redundant manipulator.

To execute this motion, the manipulator need to rotation the θ_1 degrees (Table 1). When completed this motion not with other modules, the end-effector rotation θ_1 degrees about Z_0 axis (Fig. 1).

B. Bend module

The bend module consists of seven pitch joints. The major function of it is to determine the deformation degree of the hyper-redundant manipulator by proper setting angles of each pitch joint. In a real environment, the curving motion is mainly used for avoiding obstacles or arriving at the desired position.

Based on the D-H coordinate system of the manipulator (Fig. 3), we need to set angle value of the pitch joints to execute this motion, i.e. the angle value of $\theta_2,\theta_4,\theta_6,\theta_8,\theta_{10},\theta_{12},\theta_{14}$ (Table 1).

C. Location module

The location module consists of six roll joints. By setting angles of the roll joints, we can adjust the position of the end-effector about along the Z_0 direction (Fig. 1).

To execute this motion, we need to set angle value of the roll joints based on the D-H coordinate system of the manipulator (Fig. 3), i.e. the angle value of $\theta_3,\theta_5,\theta_7,\theta_9,\theta_{11},\theta_{13}$ (Table 1).

IV. Motion Planning Methodology

The autonomous learning ability of the human brain rise from it has many learning subsystems, and every single subsystem has one or more bio-functions [13]. Through simulation of the neural frame of human brains on the physical plane, an Artificial Neural Network(ANN) can have a number of the basic bio-functions [14].

In this study, the ANN has three layers, including input layer *a*, hidden layer *b* and output layer *c*. The architecture model of the ANN is illustrated in Fig. 5.

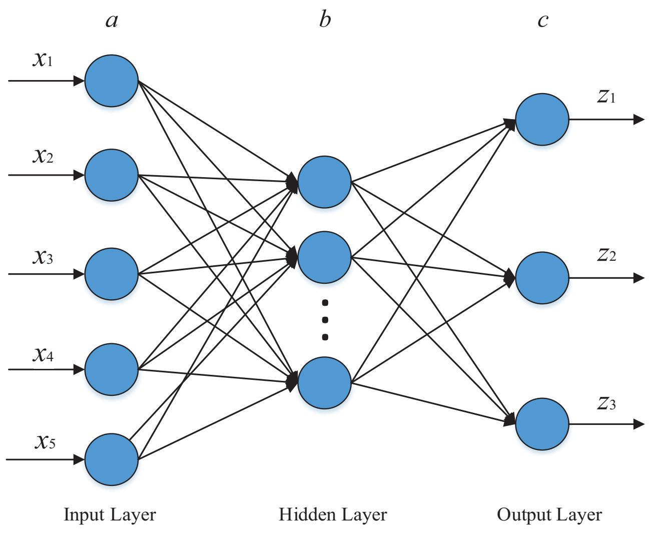

Fig. 5. The architectural model of the ANN

From the Fig.5, the input layer contains five elements, X= [x_1, x_2, x_3, x_4, x_5]. Where x_1, x_2 and x_3 are the coordinate values relative to the base coordinates which determined by the desired position of the end effector, and the unit is mm. The element x_4 is the distance of the obstacles relative to the base coordinates, and the unit is mm. The element x_5 is the diameter of the obstacles, and the unit is mm. The output layer contains three elements Z= [z_1, z_2, z_3], which denotes the rotation angles of the direction module, the bend module and the location module, i.e. $\theta_1=z_1$, $\theta_{2,4\ldots14}=z_2$ and $\theta_{3,5\ldots13}=z_3$. And the unit of them are degree. In this research, an output layer element z_i is computed from the inputs:

$$z_i = f\left(\sum_{i=1}^{5} x_i w_i + b\right) \tag{5}$$

Where *b* denotes the threshold value, *f*(*x*) denotes the activation function, and w_i denotes weights between the input and output layers. The transfer functions between nodes are squashing function and liner function, and they can be given as:

$$f(x) = \frac{2}{1+e^{-2x}} - 1, -1 \le f(x) \le 1 \tag{6}$$

$$f(x) = x, f(x) \in (-\infty, +\infty) \tag{7}$$

In this study, *net* represents the input of any node, *O* represents the output of any node, the input and output of the hidden layer and the output layer can be given as:

$$net_b = \sum_{i=1}^{5} w_{ab} O_a,\ net_c = \sum_{i=1}^{n} w_{bc} O_b \tag{8}$$

$$O_b = f(net_b), O_c = f(net_c) \tag{9}$$

The algorithm for the Mean Squared Error (MSE) is defined by:

$$E = \frac{1}{2}\sum_{i=1}^{3}(z_i - \hat{z}_i)^2 \tag{10}$$

Where $\hat{z}_i$ denotes the actual output of the network, and $\hat{z}_i = O_c$. The w_i is adjusted by negative gradient, and it can be given as:

$$w_{t+1} = w_t + \Delta w_t = w_t - \eta \frac{\partial E}{\partial w}\bigg| w = w_t \tag{11}$$

where *t* denotes the number of iterations, η denotes the learning rate i.e. the time steps, and $\eta \in [0,1]$. By adjusting the weight of the ANN and adding new simply datasets can reduce the square error. This is the learning process of a neural network. The learning algorithm of ANN consist of the following steps [15]:

- Initialization:

  Setting the value of learning rate η and expected error $\hat{E}$, and the w_i is initialized randomly.
- Import the input vector X= [x_1, x_2, x_3, x_4, x_5] and the output vector Z= [z_1, z_2, z_3].
- Calculate the actual output: $\hat{Z} = [\hat{z}_1\ \ \hat{z}_2\ \ \hat{z}_3]$, and the MSE: *E*.
- If $E \le \hat{E}$, end the program. Where $\hat{E}$ is the expected MSE.
- Calculate learning errors of each layer.
- Modify the value of w_i, go to STEP 3, and recalculate the MSE: *E* and the actual output: $\hat{Z} = [\hat{z}_1\ \ \hat{z}_2\ \ \hat{z}_3]$.
- Program ends.

V. Simulation

Hyper-redundant manipulators are ideal for obstacle avoidance because of their ability to assume diverse and complicated geometries [16]. Based on the motion planning methodology which is proposed in the above Section, we present the simulation results of the control for a hyper-redundant manipulator. In this Section, we present three types of the simulation experiments based on different diameter and position of the cylindrical obstacles as shown in Fig. 6.

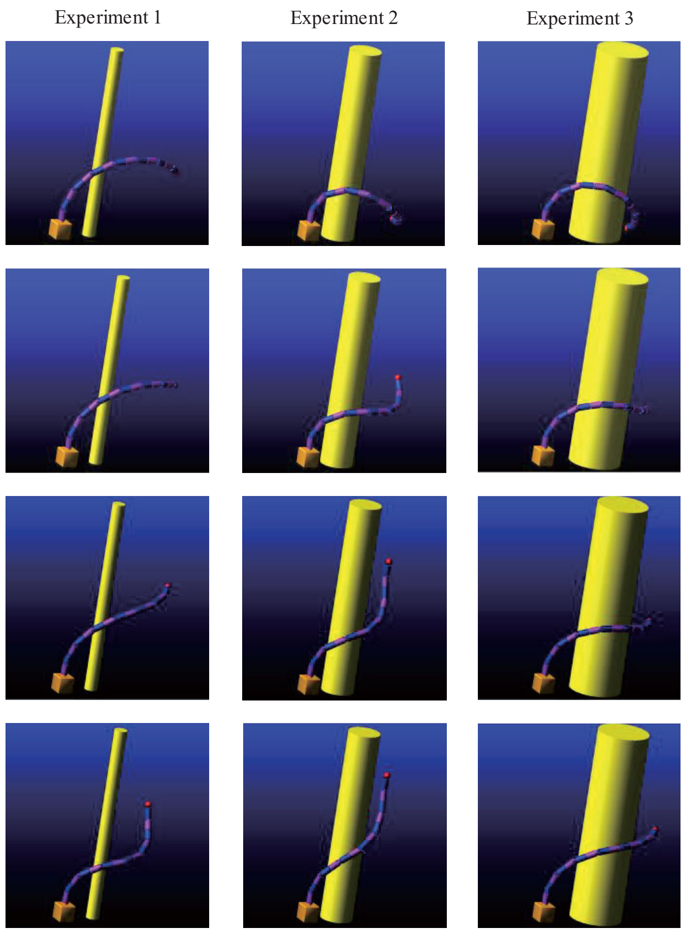

Fig. 6. Three groups of simulation experiments

By controlling the rotation angles of each primitive motion module, the hyper-redundant manipulator can achieve the motion of avoid the cylindrical obstacles. And then we collect the data from the three groups simulation experiments respecttively, generate the database of neural network.

There are 49 groups of the experiment data, and the first simulation experiment include 20 groups of data, the second simulation experiment include 18 groups of data, and the third simulation experiment include 11 groups of data.

The correlated parameters settings of the neural network are as follow: the number of iterations are 100, the expected MSE is 0.01. And the neural network is a two-layer feedforward network, with a sigmoid transfer function in the hidden layer and a linear transfer function in the output layer. The error change curve of neural network training as shown in Fig. 7. The linear regression between the network outputs and the corresponding targets as shown in Fig. 8.

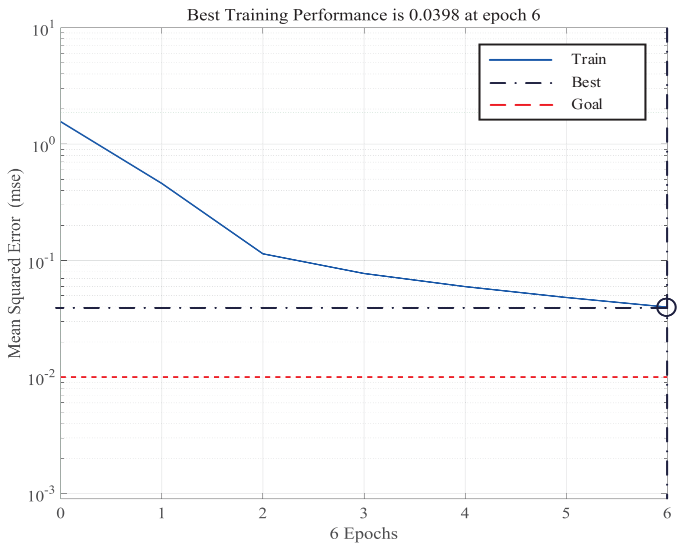

Fig. 7. The error change curve of neural network training

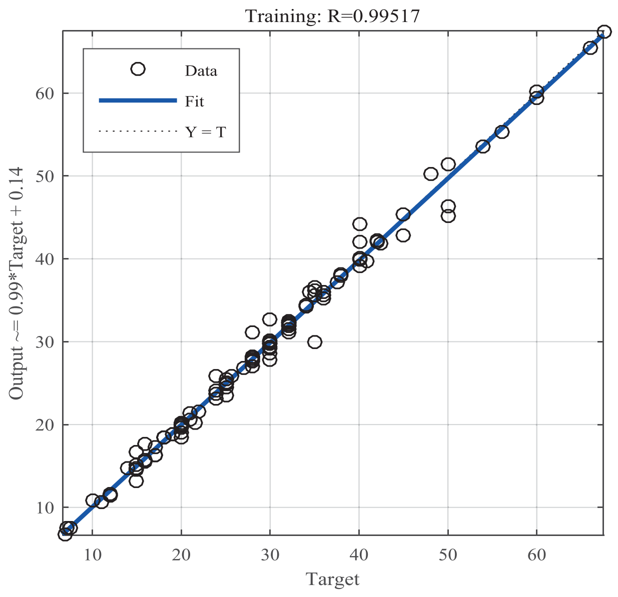

Fig. 8. The network outputs with respect to targets for training

This training stopped when the MSE is 0.0398, which occurred at epoch 6. The speed of the method and its training was less than 1s. According to the Fig. 7 and Fig. 8, we can arrive at the conclusion that the best training performance of the MSE is 0.0398. The output tracks the targets very well for training, and the R-value is over 0.99 for the total response. The experimental data of neural network training verify that the actual output of the network are performed without considerable errors.

In this case, the network response is satisfactory, and we can now teste the trained neural network model with three input vector: X_{test1}= [-14 274 -15 100 50], X_{test2}= [-27 304 -320 180 80] and X_{test3}= [26 260 48 100 20], the test results as shown in Table 2.

TABLE II. TEST RESULTS OF HYPER-REDUNDANT MANIPULATOR

Name	**Target Data**			**Actual Output Value**		
Symbol	z_1	z_2	z_3	$\hat{z}_1$	$\hat{z}_2$	$\hat{z}_3$
Numerical Value	32	32	16	31.65	31.99	17.52
	28	18.9	28	28.06	19.03	27.74
	67.5	30	7.5	67.39	29.76	7.61

According to the Table 2, we can get the errors of two testing samples: $error_{\text{test1}}$= [0.35 0.01 -1.52], $error_{\text{test2}}$= [-0.06 -0.13 0.26] and $error_{\text{test3}}$= [0.11 0.24 0.11]. These results demonstrated that motion planning methodology which described in Section 4 can meet requirement of the hyper-redundant manipulator control system. Due to the complex construction of hyper-redundant manipulator and the small size of training database, some errors of testing experiment are distinct.

VI. CONCLUSION

In this work, we suggest and analyze a method to deal with the motion planning problems of hyper-redundant manipulator which has 14 DOFs. The main contributions of this paper as follows:

a) Modified the D-H modelling of hyper-redundant manipulator from one-unit module into the whole model, and introduced a strategy which can be used to identification of primitive motion modules.

b) Based on the learning algorithm of ANN, we proposed a motion planning methodology.

c) Presented three types of the simulation experiments based on different diameter and position of the cylindrical obstacles, and tested the trained neural network model with three input vectors.

These results demonstrated that the method can meet requirement of the control system of a hyper-redundant manipulator. Although this method can improve the performance of motion planning, some errors of testing experiment may not be sufficient for accuracy controlling. However, we argue that the reasons are the complex construction of the hyper-redundant manipulator and the small size of training database.

REFERENCES

[1] G. S. Chirikjian, "Conformational Modeling of Continuum Structures in Robotics and Structural Biology: A Review," Advanced Robotics, vol. 29, pp. 817-829, 2015.

[2] P. Liljebäck, K. Y. Pettersen, Ø. Stavdahl, and J. T. Gravdahl, "A review on modelling, implementation, and control of snake robots," Robotics and Autonomous Systems, vol. 60, pp. 29-40, 2012.

[3] S. Hirose, "Biologically inspired robots-snake-like locomotors and manipulators," Oxford: Oxford university press, 1993.

[4] N. Shvalb, B. B. Moshe, and O. Medina, "A real-time motion planning algorithm for a hyper-redundant set of mechanisms," Robotica, vol. 31, pp. 1327 - 1335, December 2013.

[5] K. Ka-Wai, T. Kuen Hung, V. Vitiello, J. Clark, G. C. T. Chow, W. Luk, et al., "Dimensionality Reduction in Controlling Articulated Snake Robot for Endoscopy Under Dynamic Active Constraints," Robotics, IEEE Transactions on, vol. 29, pp. 15-31, 2013.

[6] G. Chaohui, M. Tesch, D. Rollinson, and H. Choset, "Snakes on an inclined plane: Learning an adaptive sidewinding motion for changing slopes," 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014), pp. 1114-1119, 2014.

[7] V. B. Penhune and C. J. Steele, "Parallel contributions of cerebellar, striatal and M1 mechanisms to motor sequence learning," Behavioural Brain Research, vol. 226, pp. 579-591, 2012.

[8] Y. Yamashita and J. Tani, "Correction: Emergence of Functional Hierarchy in a Multiple Timescale Neural Network Model: A Humanoid Robot Experiment," PLoS computational biology, vol. 6, 2010.

[9] S. Schaal, A. Ijspeert and A .Billard, "Computational approaches to motor learning by imitation," Philosophical transactions of the Royal Society of London. Series B, Biological sciences, vol. 358, pp. 537-547, 2003.

[10] H. Haghighi, F. Abdollahi, and S. Gharibzadeh, "Brain-inspired self-organizing modular structure to control human-like movements based on primitive motion identification," Neurocomputing, vol. 173, Part 3, pp. 1436-1442, January 2016.

[11] A. Rahmani and A. Ghanbari, "Application of neural network training in forward kinematics simulation for a novel modular hybrid manipulator with experimental validation," Intelligent Service Robotics, vol. 9, pp. 79-91, 2015.

[12] J. Denavit, R. S. Hartenberg, "A Kinematic Notation for Lower-Pair Mechanics based on Matrices," Journal of Applied Mechanics, vol. 22, pp. 215-221, 1955.

[13] M. Naili, A. Boubetra, A. Tari, Y. Bouguezza and A. Achroufene, "Brain-inspired method for solving fuzzy multi-criteria decision making problems (BIFMCDM)," Expert Systems with Applications, vol. 42, pp. 2173-2183, 2015.

[14] G. E. Hinton, S. Osindero and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," Neural Computation, vol. 18, pp. 1527-1554, 2006.

[15] Rumelhart D E and McCMland J L, "Parallel Distributed Processing," Cambridge, MA: MIT press, 1986.

[16] G. S. Chirikjian and J. W. Burdick, "An obstacle avoidance algorithm for hyper-redundant manipulators," in Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on, pp. 625-631 vol.1, 1990.