Inverse Control of Cable-driven Parallel Mechanism Using Type-2 Fuzzy Neural Network

LI Cheng-Dong[1] YI Jian-Qiang[1] YU Yi[1]
ZHAO Dong-Bin[1]

Abstract The level adjustment of cable-driven parallel mechanism is challenging due to the difficulty in obtaining an accurate mathematical model and the fact that different sources of uncertainties exist in the adjustment process. This paper presents a type-2 fuzzy neural network (T2FNN) based inverse control scheme for the level adjustment process. The T2FNN, whose consequent interval weights are trained by the iterative least squares estimation method, is used to approximate the inverse dynamics of the process and to handle uncertainties. Finally, the proposed control scheme and its counterpart — type-1 fuzzy neural network based inverse control scheme — are compared and implemented for leveling the cable-driven parallel mechanism.

Key words Inverse control, type-2 fuzzy logic, neural network, parallel mechanism

DOI 10.3724/SP.J.1004.2010.00459

When loading and unloading some payloads such as aircrafts, satellites, and special containers, which are valuable, fragile, and unfortunately eccentric, we must level their bottom surface as these payloads cannot endure a point-to-point touch with the ground or the assembly platform. Based on the analysis of advantages and disadvantages of existing techniques and mechanisms[1−2], we have designed and constructed a new cable-driven parallel mechanism[3−4]. The structure and configuration of the cable-driven parallel mechanism is shown in Fig. 1. The parallel mechanism mainly consists of a rectangular worktable, linear motion units diagonally fixed on the rectangular worktable, two angle sensors fixed in two diagonals of the payload, cables passing through the axial line of each linear motion unit, and a computer control platform. In our experiment, the center of gravity of the payload can be changed through adding or taking off heavy weights; as a result, the payload will become slantwise. The control objective is to make the bottom surface of the eccentric and slantwise payload be leveled by regulating the positions of the linear motion units to change the lengths of the cables.

For the cable-driven parallel mechanism, a two-dimensional model is established in [4], but for the three-dimensional payload, the accurate mathematical model is quite difficult to obtain, as the centroid position is uncertain and the mass of the payload is assumed to be unknown to meet the actual engineering requirements. Therefore, design of conventional controllers for this cable-driven parallel mechanism becomes very difficult. Inverse control scheme[5−8] offers a method to solve this problem. The concept of inverse control has been a preferred approach to the design of control systems with complex and uncertain dynamics. It utilizes an inverse controller to approximate the inverse dynamics of the process. This control strategy has been proven effective in many applications, such as control of a partially simulated exothermic reactor[5], control of a steel pickling process[6], temperature system control[7].

Moreover, we note that high levels of uncertainties exist in this real-world application, which are listed as follows: 1) uncertainties from the angle sensors, which arise from inevitable measurement errors and the resolution limits of the angle sensors; 2) uncertainties existing in the adjustment process. The payload will sway in the adjustment process, and this will further affect the precision of the angle sensors; 3) uncertainties associated with the experimental data, which are used to tune the controller for level adjustment. The experimental data are obtained using manual adjustment; hence, noise inevitably exists in the data.

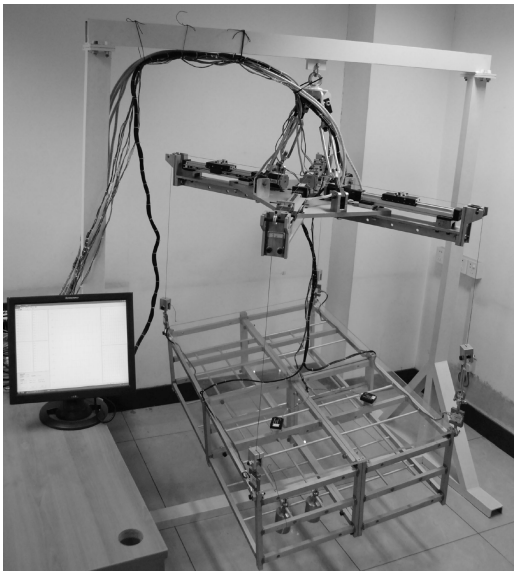


Fig. 1 Cable-driven parallel mechanism

Proponents of type-2 fuzzy logic[9−16] have argued that type-2 fuzzy logic can provide the capability to model high levels of uncertainties that conventional fuzzy logic (type-1 fuzzy logic) has some limitations to handle. Applications of type-2 fuzzy logic[14−16] have also demonstrated that it has the potential to produce more complex input-output map and outperform type-1 fuzzy logic. Furthermore, to combine the advantages of type-2 fuzzy logic and neural networks; type-2 fuzzy neural network (T2FNN)[17−20] was studied by researchers and applied in system identification[18], control problems[17, 20], etc. To the authors′ knowledge, most researches of T2FNN are about function approximation, whereas in the control domain, only few papers use T2FNN as direct controllers where BP algorithms are adopted. Till now, no paper has used T2FNN as an inverse controller.

In our work, we use the T2FNN to approximate and model the inverse dynamics of the level adjustment process of the cable-driven parallel mechanism. As the fuzzy rule base and the antecedent part parameters (centers and widths of type-2 fuzzy sets) of the T2FNN can be obtained from our experimental experience, only the consequent part parameters are tuned to reduce the training time. In parameter learning, the iterative least squares estimation (ILSE) method[21−22], which has fast convergence

Manuscript received December 15, 2008; accepted February 25, 2009
Supported by National High Technology Research and Development Program of China (863 Program) (2007AA04Z239) and National Natural Science Foundation of China (60621001, 60975060)
1. Institute of Automation, Chinese Academy of Sciences, Beijing 100190, P. R. China

speed, is adopted. After being trained, the T2FNN inverse controller is applied to level the cable-driven parallel mechanism. Simultaneously, comparisons with type-1 fuzzy neural network (T1FNN) based inverse controller are also made. Experimental results and comparisons show that both inverse controllers can level the cable-driven parallel mechanism, but, the T2FNN inverse controller performs more consistently than the T1FNN inverse controller, in other words, the T2FNN inverse controller performs better than the T1FNN inverse controller when facing high levels of uncertainties.

This paper is organized as follows: In Section 1, the cable-driven parallel mechanism is introduced. In Section 2, the structure and learning of the T2FNN inverse controller for the cable-driven parallel mechanism are studied. In Section 3, experimental results and comparisons are made. Finally, the conclusions are drawn in Section 4.

1 Cable-driven parallel mechanism

In this section, we present the configuration of the cable-driven parallel mechanism and specify the requirement of the controller for this mechanism.

The configuration of the cable-driven parallel mechanism as shown in Fig. 1 mainly consists of the following major components:

1) Rectangular worktable: On the rectangular worktable, the actuators — linear motion units are fixed in the diagonals. Also, there is one cable passing through the axial line of each linear motion unit. Therefore, we can regulate the cable in each diagonal through changing the positions of the linear motion units, denoted as u_x and u_y, to realize our control objective — leveling the bottom surface of the inclined payload.

2) Angle sensors: Two angle sensors are used to measure the inclination of the payload. As the top surface and the bottom surface of the payload shown in Fig. 1 are parallel, hence, for convenience, the two angle sensors are fixed on the top surface of the payload in each diagonal. If the two diagonal inclination angles, denoted as θ_x and θ_y, are both zero, then the bottom surface of the payload will be leveled.

3) Computer control platform: The control platform is used to realize the function of the controller designed for this level adjustment problem.

In this control problem, the payload is the controlled plant whose mass is supposed to be unknown with uncertain centroid position. This assumption leads to meet the actual engineering requirements. The payload′s center of gravity can be changed through adding or removing some heavy weights. The bottom surface of the payload needs to be adjusted to be level ($\theta_x < 0.2^\circ$ and $\theta_y < 0.2^\circ$) in no more than 20 s in the experiments.

As discussed above, it is difficult to obtain an accurate mathematical model and design a conventional controller. And, there are several sources of uncertainties in this real-world application. Considering that T2FNN has the capability to model high levels of uncertainties and can approximate any given nonlinear systems, in the following, we will use the T2FNN to construct the inverse model of this cable-driven parallel mechanism and develop a T2FNN inverse controller for the level adjustment of this cable-driven parallel mechanism.

2 Type-2 fuzzy neural network based inverse controller

In this section, considering the characteristics of the mechanism, we first discuss the architecture of the inverse control system for the cable-driven parallel mechanism, and then, clarify the structure and learning algorithm of the T2FNN inverse controller.

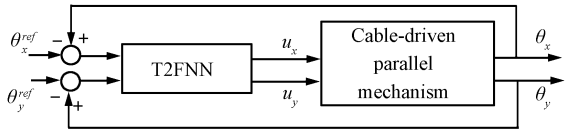
2.1 Architecture of the inverse control system

Fig. 2 (a) shows us the structure of the T2FNN based inverse control system. This control architecture mainly consists of the T2FNN inverse model that acts as the controller placed in series with the cable-driven parallel mechanism under control. As shown in Fig. 2 (a), the inputs of this T2FNN inverse model are the two diagonal inclination angles θ_x and θ_y, and the outputs of the T2FNN inverse model are the position changes u_x, u_y of the two linear motion units.

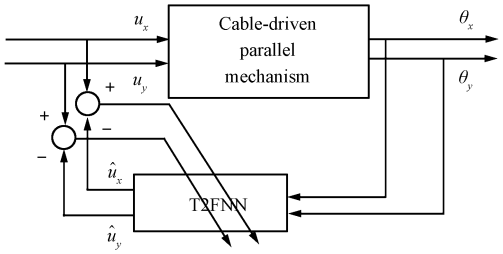
To obtain the T2FNN inverse model of the cable-driven parallel mechanism, the training process is depicted in Fig. 2 (b). For the training, the training data are obtained from real experiments to reflect input-output characteristics of the cable-driven parallel mechanism. And, the inputs and outputs of the training data are the values of θ_x, θ_y and the corresponding control signals u_x, u_y, respectively. Using the training data set $\{(\theta_x^1, \theta_y^1, u_x^1, u_y^1), (\theta_x^2, \theta_y^2, u_x^2, u_y^2), \cdots, (\theta_x^N, \theta_y^N, u_x^N, u_y^N)\}$, the T2FNN can be trained by ILSE method to minimize the error function E defined by

$$E = \sum_{i=1}^{N} \left([u_x^i - \hat{u}_x(\theta_x^i, \theta_y^i)]^2 + [u_y^i - \hat{u}_y(\theta_x^i, \theta_y^i)]^2 \right) \quad (1)$$

where N is the number of training data, $\hat{u}_x(\theta_x^i, \theta_y^i)$ and $\hat{u}_y(\theta_x^i, \theta_y^i)$ are the actual outputs of the T2FNN for the input (θ_x^i, θ_y^i).



(a) Structure



(b) Training process

Fig. 2 The structure and training process of the T2FNN based inverse control system for the cable-driven parallel mechanism

In the following, we will discuss the structure and learning of the T2FNN for the cable-driven parallel mechanism in detail.

2.2 Structure of type-2 fuzzy neural network

In [17−20], the structure and training of T2FNNs have been discussed. A T2FNN is very similar to a conventional (type-1) fuzzy neural network (T1FNN). Both of them have four layers, but some differences exist between them: 1) In the membership function layer (Layer 2), the fuzzy sets of T1FNN are type-1, but the fuzzy sets of T2FNN are interval type-2; 2) The consequent weights between Layer 3 and

Layer 4 are crisp values for T1FNN, while the consequent weights between Layer 3 and Layer 4 are interval values for T2FNN; 3) Layer 4 is only used to carry out the defuzzification process for a T1FNN, but, for a T2FNN, Layer 4 needs to accomplish the type-reduction and defuzzification processes.

The structure of the T2FNN for the cable-driven parallel mechanism is shown in Fig. 3. This is a two-input-two-output network. In order to ensure the safety of the mechanism, the two diagonal inclination angles should be bounded in $[-6^\circ, 6^\circ]$. By assigning each input variable (θ_x or θ_y) five Gaussian interval type-2 fuzzy sets (IT2FSs) – NB, NS, ZR, PS, PB in the range $[-6^\circ, 6^\circ]$, we can obtain M ($M = 25$) fuzzy rules, each of which has the following form.

$$\begin{aligned}\text{Rule } k: \text{IF } \theta_x \text{ is } \widetilde{A}_x^k \text{ and } \theta_y \text{ is } \widetilde{A}_y^k, \text{ THEN}\\ u_x \text{ is } [\underline{w}_x^k, \overline{w}_x^k] \text{ and } u_y \text{ is } [\underline{w}_y^k, \overline{w}_y^k]\end{aligned} \tag{2}$$

where $k = 1, 2, \cdots, M$, $[\underline{w}_x^k, \overline{w}_x^k]$, $[\underline{w}_y^k, \overline{w}_y^k]$ are interval weights of the consequent part, $\widetilde{A}_z^k (z = x, y)$ are IT2FSs NB, NS, ZR, PS or PB, the membership functions of which are shown in Fig. 4, and can be expressed as

$$\underline{\mu}_{\widetilde{A}_z^k}(\theta_z) = \exp\left[-\frac{1}{2}\left(\frac{\theta_z - m_z^k}{\underline{\sigma}_z^k}\right)^2\right] \tag{3}$$

$$\overline{\mu}_{\widetilde{A}_z^k}(\theta_z) = \exp\left[-\frac{1}{2}\left(\frac{\theta_z - m_z^k}{\overline{\sigma}_z^k}\right)^2\right] \tag{4}$$

where $\underline{\mu}_{\widetilde{A}_z^k}$ and $\overline{\mu}_{\widetilde{A}_z^k}$ denote the grades of the lower and upper membership functions of the IT2FS $\widetilde{A}_z^k$, $[\underline{\sigma}_z^k, \overline{\sigma}_z^k]$ is the uncertain standard deviation of $\widetilde{A}_z^k$.

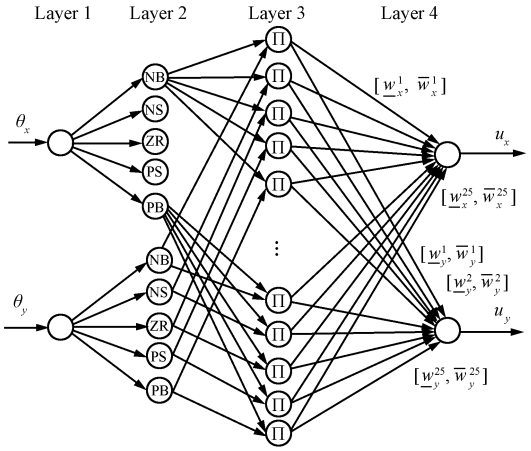


Fig. 3 The structure of type-2 fuzzy neural network

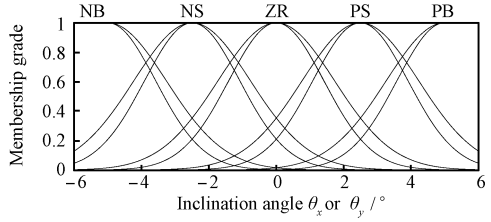


Fig. 4 Membership functions for θ_x and θ_y

Once a crisp input $\boldsymbol{\theta} = (\theta_x, \theta_y)$ is applied to the T2FNN, through the singleton fuzzifier (Layer 1) and the type-2 inference process (Layer 2), the firing strength of the k-th rule (k-th node), which is an interval type-1 set, can be obtained as

$$F^k = [\underline{f}^k, \overline{f}^k] \tag{5}$$

where

$$\underline{f}^k = \underline{\mu}_{\widetilde{A}_x^k}(\theta_x) \star \underline{\mu}_{\widetilde{A}_y^k}(\theta_y) \tag{6}$$

$$\overline{f}^k = \overline{\mu}_{\widetilde{A}_x^k}(\theta_x) \star \overline{\mu}_{\widetilde{A}_y^k}(\theta_y) \tag{7}$$

in which $\star$ denotes minimum or product t-norm. In our study, product t-norm is adopted.

Once the interval firing strengths in Layer 3 are computed, to generate a crisp output from Layer 4, the outputs of the third layer should be type-reduced and then defuzzified[9]. Using the center-of-sets (COS) type-reduction method[9−11], the type-reduced interval $[\underline{u}_z, \overline{u}_z]$ can be computed as

$$U_z = [\underline{u}_z, \overline{u}_z] = \int_{w_z^1 \in [\underline{w}_z^1, \overline{w}_z^1]} \cdots \int_{w_z^M \in [\underline{w}_z^M, \overline{w}_z^M]} \int_{f^1 \in [\underline{f}^1, \overline{f}^1]} \cdots \int_{f^M \in [\underline{f}^M, \overline{f}^M]} \frac{1}{\dfrac{\sum_{k=1}^M f^k w_z^k}{\sum_{k=1}^M f^k}} \tag{8}$$

where $z = x, y$; the left end point $\underline{u}_z$ and the right end point $\overline{u}_z$ can be computed by

$$\underline{u}_z = \frac{\sum_{k=1}^M g^k \underline{w}_z^k}{\sum_{k=1}^M g^k}, \quad g^k = \underline{\delta}^k \overline{f}^k + (1 - \underline{\delta}^k)\underline{f}^k \tag{9}$$

$$\overline{u}_z = \frac{\sum_{k=1}^M h^k \overline{w}_z^k}{\sum_{k=1}^M h^k}, \quad h^k = \overline{\delta}^k \underline{f}^k + (1 - \overline{\delta}^k)\overline{f}^k \tag{10}$$

where $\underline{\delta}^k$ and $\overline{\delta}^k$ can be determined in $\{0, 1\}$ by Karnik-Mendel algorithms[9−12]. The algorithms for determining $\underline{\delta}^k$ and $\overline{\delta}^k$ are given in detail in the Appendix.

The defuzzified output of Layer 4 can be computed as the average of $\underline{u}_z$ and $\overline{u}_z$, i.e.,

$$u_z = \frac{1}{2}(\underline{u}_z + \overline{u}_z) = \frac{1}{2}\left[\sum_{k=1}^M \widetilde{g}^k \underline{w}_z^k + \sum_{k=1}^M \widetilde{h}^k \overline{w}_z^k\right] \tag{11}$$

where $z = x$ or y, $\widetilde{g}^k = \frac{g^k}{\sum_{k=1}^M g^k}$, $\widetilde{h}^k = \frac{h^k}{\sum_{k=1}^M h^k}$.

2.3 Learning of type-2 fuzzy neural network

In our application of leveling the cable-driven parallel mechanism, the antecedent IT2FSs of the two diagonal inclination angles θ_x, θ_y and their parameters (e.g., fixed means and uncertain standard deviations) can be easily determined. Here, we just tune the consequent parameters (interval weights) of the T2FNN inverse model to approximate the characteristics of the cable-driven parallel mechanism.

Suppose that the input-output training data are $(\theta_x^1, \theta_y^1, u_x^1, u_y^1)$, $(\theta_x^2, \theta_y^2, u_x^2, u_y^2)$, $\cdots$, $(\theta_x^N, \theta_y^N, u_x^N, u_y^N)$. Our goal

is to adjust the interval weights to minimize the following squared error function:

$$E = E_x + E_y \tag{12}$$

where

$$E_x = \sum_{i=1}^{N}(u_x^i - \hat{u}_x(\boldsymbol{\theta}^i))^2 \tag{13}$$

$$E_y = \sum_{i=1}^{N}(u_y^i - \hat{u}_y(\boldsymbol{\theta}^i))^2 \tag{14}$$

in which $\hat{u}_x(\boldsymbol{\theta}^i), \hat{u}_y(\boldsymbol{\theta}^i)$ are the outputs of the T2FNN for the input $\boldsymbol{\theta}^i = (\theta_x^i, \theta_y^i)$.

To the authors' knowledge, all the earlier studies using T2FNNs as controllers train the T2FNNs by BP algorithms (steepest descent algorithm)[17, 20]. But inherent shortcomings are encountered in the use of the steepest descent algorithm, e.g., slow convergence speed in training, easy to be trapped in a local minima, etc. From (11), the output $u_z(\boldsymbol{\theta})$ of the T2FNN is linear with the consequent weights $\underline{w}_z^k$ and $\overline{w}_z^k$. For this reason, to improve the learning performance, the ILSE method[21−22] is adopted to determine the consequent interval weights.

With the ILSE method[21−22], the consequent parameter vector $\boldsymbol{w}_z$ is updated by

$$\boldsymbol{w}_z^{i+1} = \boldsymbol{w}_z^i + \gamma_{i+1} P^i \boldsymbol{q}_z^{i+1}[u_z^{i+1} - (\boldsymbol{q}_z^{i+1})^{\mathrm{T}} \boldsymbol{w}_z^i] \tag{15}$$

$$P^{i+1} = P^i - \gamma_{i+1} P^i \boldsymbol{q}_z^{i+1} (\boldsymbol{q}_z^{i+1})^{\mathrm{T}} P^i \tag{16}$$

$$\gamma_{i+1} = \frac{1}{\lambda + (\boldsymbol{q}_z{}^{i+1})^{\mathrm{T}} P^i \boldsymbol{q}_z^{i+1}} \tag{17}$$

where

$$\boldsymbol{w}_z = [\underline{w}_z^1, \cdots, \underline{w}_z^M, \overline{w}_z^1, \cdots, \overline{w}_z^M]^{\mathrm{T}},$$

$$\boldsymbol{q}_z^i = \frac{1}{2}[\widetilde{g}^1(\boldsymbol{\theta}^i), \cdots, \widetilde{g}^M(\boldsymbol{\theta}^i), \widetilde{h}^1(\boldsymbol{\theta}^i), \cdots, \widetilde{h}^M(\boldsymbol{\theta}^i)]^{\mathrm{T}}$$

$z = x$ or y, $0 < \lambda \leq 1$ is the forgetting factor and P^i is the covariance matrix. The initial conditions are $\boldsymbol{w}_z^0 = \boldsymbol{0}$ and $P^0 = 100I$, where I is the identity matrix of dimension $2M \times 2M$.

Here, we do not adjust the antecedent parameters for two reasons. The first one is that the antecedent parameters can be easily determined in our application; the second is that tuning with only the consequent parameters can significantly reduce the training time and is more practicable.

3 Experimental results

In our experiment, we use 492 experiential data pairs to train the T2FNN inverse model. The initial consequent parameter vectors $\boldsymbol{w}_x$ and $\boldsymbol{w}_y$ are set to be 0. The forgetting factor λ is chosen to be 0.9. The root mean squared error (RMSE) curve in the training process is demonstrated in Fig. 5, where RMSE $=\sqrt{E/N} = \sqrt{(E_x + E_y)/N}$. Fig. 6 demonstrates an example of the training process of the consequent interval weights. After being trained, the rule tables for u_x and u_y are shown in Tables 1 and 2, respectively.

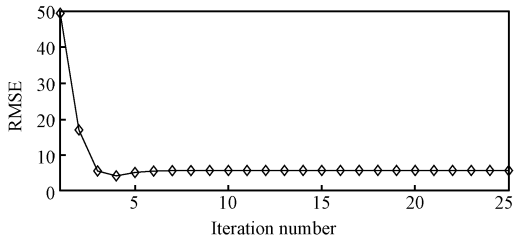


Fig. 5 RMSE curve in the training process for $\boldsymbol{w}_x$ and $\boldsymbol{w}_y$

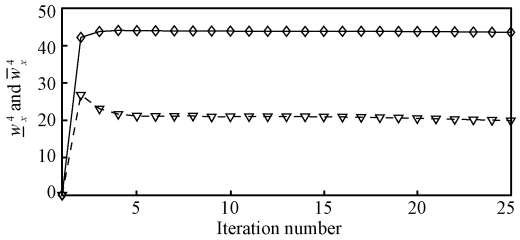


Fig. 6 Training process of $\underline{w}_x^4$ ($\bigtriangledown$) and $\overline{w}_x^4$ ($\Diamond$)

Table 1 Rule table for u_x

		θ_x				
u_x(mm)		NB	NS	ZR	PS	PB
	NB	[34.6384, 104.0283]	[17.6200, 41.8753]	[19.0126, 21.3735]	[19.9231, 43.5774]	[41.7403, 99.4294]
	NS	[17.6320, 40.9025]	[44.1999, 48.6525]	[19.1361, 23.1038]	[44.4670, 47.1031]	[19.6939, 34.5232]
θ_y	ZR	[−1.4157, −0.8939]	[−1.5490, −0.3534]	[−7.2618, 4.3008]	[−1.0464, 0.8134]	[−0.3499, 0.1310]
	PS	[−43.4602, −19.3997]	[−47.1519, −40.5727]	[−18.6420, −18.6374]	[−48.6317, −43.9666]	[−40.6944, −18.9890]
	PB	[−98.0743, −43.8629]	[−37.6772, −19.2509]	[−21.9452, −18.7983]	[−41.9587, −18.7532]	[−100.4452, −41.0768]

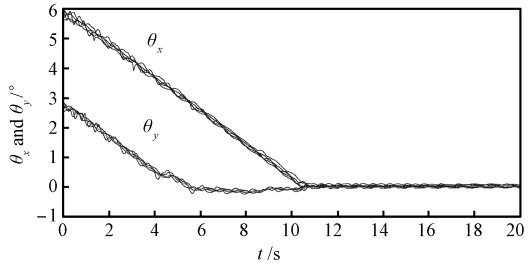
Table 2 Rule table for u_y

		θ_x				
u_y(mm)		NB	NS	ZR	PS	PB
	NB	[33.6385, 103.2333]	[17.5543, 39.1302]	[2.5712, 2.8789]	[−39.6072, −20.0114]	[−97.6042, −42.4389]
	NS	[17.6363, 39.4192]	[40.7539, 47.2926]	[1.7397, 3.0943]	[−49.3998, −47.8134]	[−42.5363, −21.1289]
θ_y	ZR	[19.9065, 23.1437]	[19.9813, 19.9894]	[−5.1374, 6.1661]	[−22.0722, −21.0073]	[−23.1810, −20.6070]
	PS	[20.1400, 41.2859]	[47.6908, 49.3785]	[−2.5672, −1.4376]	[−45.8555, −42.1699]	[−40.1494, −18.7404]
	PB	[42.2360, 98.5407]	[19.7849, 42.9565]	[−2.8188, −2.4565]	[−40.0838, −18.5723]	[−100.3569, −38.8833]

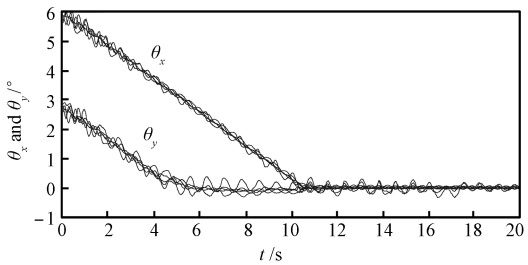
For comparison, a T1FNN inverse controller is also designed and experimented. The antecedent type-1 fuzzy sets – NB, NS, ZR, PS, and PB are still Gaussian membership functions with the same centers m_z^k as in the T2FNN, but the widths of them are the average of $\underline{\sigma}_z^k$ and $\overline{\sigma}_z^k$. In the same way, the crisp weights of the T1FNN are set as the average of the interval weights of the T2FNN.

Then, the trained T2FNN inverse controller and the T1FNN inverse controller are used for the level adjustment of the cable-driven parallel mechanism from different initial angles conditions, e.g., (5.8°, 2.7°), (−5.6°, −2.6°), (5.0°, 5.0°), and (−5.0°, −5.0°). And, for each initial condition, five experiments are done respectively for the T2FNN inverse controller and the T1FNN inverse controller. Here, only for the initial angles condition (5.8°, 2.7°), the control results of the two inverse controllers in five experiments are shown in Figs. 7 (a) and (b). The control results for the other three initial angles conditions (−5.6°, −2.6°), (5.0°, 5.0°), and (−5.0°, −5.0°) are omitted, as the experimental results are similar for the four initial angles conditions. To show the superiority of the T2FNN inverse controller compared with the T1FNN inverse controller when facing uncertainties, the standard deviations (STD) of the dynamic process $\theta_x(t)$ and $\theta_y(t)$ in the 5 experiments are shown in Figs. 8 (a) and (b).

From the experimental results (e.g. Fig. 7), we can see that both the T2FNN inverse controller and the T1FNN inverse controller can adjust the angle between the bottom surface of the payload and the horizontal plane to be less than 0.2° in about 12 s, hence, the performance requirements can be achieved. Moreover, from Figs. 7 and 8, we can also see that the T2FNN inverse controller performs obviously better and more consistently than the T1FNN inverse controller when facing uncertainties such as the measurement errors of the angle sensors and the swing of the pay-load in the adjustment process, because the response curves of θ_x and θ_y with the T2FNN inverse controller are more tightly packed than that with the T1FNN inverse controller.

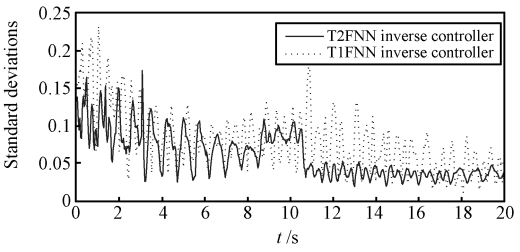


(a) Control results of the T2FNN inverse controller

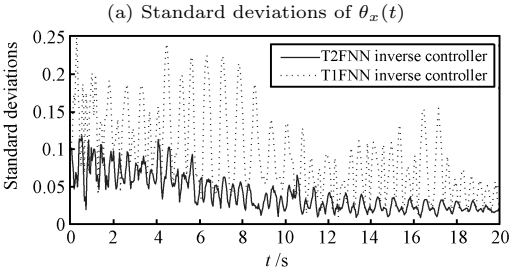


(b) Control results of the T1FNN inverse controller

Fig. 7 Control results of the T2FNN inverse controller and the T1FNN inverse controller for θ_x and θ_y in five experiments with the same initial angles (5.8°, 2.7°)



(a) Standard deviations of $\theta_x(t)$



(b) Standard deviations of $\theta_y(t)$

Fig. 8 Standard deviations of $\theta_x(t)$ and $\theta_y(t)$ in the five experiments

4 Conclusion

In our application of leveling the cable-driven parallel mechanism, it is quite difficult to obtain a suitable mathematical model and design conventional controllers for the level adjustment process. A promising way to overcome these difficulties is to use inverse model to approximate the inverse dynamic behavior of the process. In this work, considering that different sources of uncertainties exist in the level adjustment process and type-2 fuzzy logic has the ability to handle high levels of uncertainties and approximate any nonlinear functions, a T2FNN inverse control strategy is proposed and implemented for leveling the cable-driven parallel mechanism. To enhance the performance of the T2FNN inverse controller, the ILSE method was adopted to determine the consequent interval weights. Experimental result demonstrated the effectiveness of the proposed control scheme. Moreover, experimental comparisons with the T1FNN inverse controller indicated that the T2FNN inverse controller performed more consistently.

Appendix

In this appendix, we give two algorithms for computing $\underline{\delta}^k$, $\overline{\delta}^k$ in (9), (10), respectively. These algorithms are based on Karnik-Mendel algorithms[9−12].

To begin, for simplicity, let us denote

$$\begin{aligned}\underline{\boldsymbol{f}} &= (\underline{f}^1, \underline{f}^2, \cdots, \underline{f}^M)^{\mathrm{T}}\\ \overline{\boldsymbol{f}} &= (\overline{f}^1, \overline{f}^2, \cdots, \overline{f}^M)^{\mathrm{T}}\\ \underline{\boldsymbol{w}}_z &= (\underline{w}_z^1, \underline{w}_z^2, \cdots, \underline{w}_z^M)^{\mathrm{T}}\\ \overline{\boldsymbol{w}}_z &= (\overline{w}_z^1, \overline{w}_z^2, \cdots, \overline{w}_z^M)^{\mathrm{T}}\end{aligned}$$

where $z = x$ or y.

Algorithm 1. Computation of $\underline{\delta}^k$ in (9)

Step 1. Implement a permutation transformation to $\underline{\boldsymbol{w}}_z$, such that $\widetilde{\underline{\boldsymbol{w}}}_z = \underline{Q}_z \underline{\boldsymbol{w}}_z = (\widetilde{\underline{w}}_z^1, \widetilde{\underline{w}}_z^2, \cdots, \widetilde{\underline{w}}_z^M)^{\mathrm{T}}$, and $\widetilde{\underline{w}}_z^1 \le \widetilde{\underline{w}}_z^2 \le \cdots \le \widetilde{\underline{w}}_z^M$, where $\underline{Q}_z$ is the permutation matrix.

Step 2. Implement the same permutation transformation as in Step 1 to $\overline{\boldsymbol{f}}$ and $\underline{\boldsymbol{f}}$; and, denote $\overline{\boldsymbol{g}} = (\overline{g}^1, \overline{g}^2, \cdots, \overline{g}^M)^{\mathrm{T}} = \underline{Q}_z \overline{\boldsymbol{f}}$, $\underline{\boldsymbol{g}} = (\underline{g}^1, \underline{g}^2, \cdots, \underline{g}^M)^{\mathrm{T}} = \underline{Q}_z \underline{\boldsymbol{f}}$.

Step 3. Compute the switch point L through Karnik-Mendel algorithm[9−12]:

1) Initially, set $g^i = \frac{1}{2}(\underline{g}^i + \overline{g}^i),\ i = 1, 2, \cdots, M$, then, compute $o_l = \frac{\sum_{i=1}^{M} g^i \widetilde{w}_z^i}{\sum_{i=1}^{M} g^i}$, and let $o_l^{'} = o_l$;

2) Find $\widetilde{L}$ $(1 \le \widetilde{L} \le M-1)$, such that $\widetilde{w}_z^{\widetilde{L}} \le o_l^{'} \le \widetilde{w}_z^{\widetilde{L}+1}$;

3) Compute $o_l = \frac{\sum_{i=1}^{\widetilde{L}} \overline{g}^i \widetilde{w}_z^i + \sum_{i=\widetilde{L}+1}^{M} \underline{g}^i \widetilde{w}_z^i}{\sum_{i=1}^{\widetilde{L}} \overline{g}^i + \sum_{i=\widetilde{L}+1}^{M} \underline{g}^i}$ and let $o_l^{''} = o_l$;

4) If $o_l^{''} = o_l^{'}$, then stop and return $L = \widetilde{L}$; else if $o_l^{''} \neq o_l^{'}$, then let $o_l^{'} = o_l^{''}$ and return to 2).

Step 4. Let $\underline{\boldsymbol{r}}_z = (\underbrace{1, 1, \cdots, 1}_{L}, \underbrace{0, \cdots, 0}_{M-L})^{\mathrm{T}}$, then $\underline{\delta}^k$ $(k = 1, 2, \cdots, M)$ can be computed by

$$(\underline{\delta}^1, \underline{\delta}^2, \cdots, \underline{\delta}^M)^{\mathrm{T}} = \underline{Q}_z \underline{\boldsymbol{r}}_z$$

Algorithm 2. Computation of $\overline{\boldsymbol{\delta}}^{\boldsymbol{k}}$ in (10)

Step 1. Implement a permutation transformation to $\overline{\boldsymbol{w}}_z$, such that $\widetilde{\overline{\boldsymbol{w}}}_z = \overline{Q}_z \overline{\boldsymbol{w}}_z = (\widetilde{\overline{w}}_z^1, \widetilde{\overline{w}}_z^2, \cdots, \widetilde{\overline{w}}_z^M)^{\mathrm{T}}$, and $\widetilde{\overline{w}}_z^1 \le \widetilde{\overline{w}}_z^2 \le \cdots \le \widetilde{\overline{w}}_z^M$, where $\overline{Q}_z$ is the permutation matrix.

Step 2. Implement the same permutation transformation as in Step 1 to $\overline{\boldsymbol{f}}$ and $\underline{\boldsymbol{f}}$; and, denote $\overline{\boldsymbol{g}} = (\overline{g}^1, \overline{g}^2, \cdots, \overline{g}^M)^{\mathrm{T}} = \overline{Q}_z \overline{\boldsymbol{f}}$, $\underline{\boldsymbol{g}} = (\underline{g}^1, \underline{g}^2, \cdots, \underline{g}^M)^{\mathrm{T}} = \overline{Q}_z \underline{\boldsymbol{f}}$.

Step 3. Compute the switch point R through Karnik-Mendel algorithm[9−12]:

1) Initially, set $g^i = \frac{1}{2}(\underline{g}^i + \overline{g}^i), i = 1, 2, \cdots, M$, then, compute $o_r = \frac{\sum_{i=1}^{M} g^i \widetilde{\overline{w}}_z^i}{\sum_{i=1}^{M} g^i}$, and let $o_r^{'} = o_r$;

2) Find $\widetilde{R}$ $(1 \le \widetilde{R} \le M-1)$, such that $\widetilde{\overline{w}}_z^{\widetilde{R}} \le o_r^{'} \le \widetilde{\overline{w}}_z^{\widetilde{R}+1}$;

3) Compute $o_r = \frac{\sum_{i=1}^{\widetilde{R}} \underline{g}^i \widetilde{\overline{w}}_z^i + \sum_{i=\widetilde{R}+1}^{M} \overline{g}^i \widetilde{\overline{w}}_z^i}{\sum_{i=1}^{\widetilde{R}} \underline{g}^i + \sum_{i=\widetilde{R}+1}^{M} \overline{g}^i}$ and let $o_r^{''} = o_r$;

4) If $o_r^{''} = o_r^{'}$, then stop and return $R = \widetilde{R}$; else if $o_r^{''} \neq o_r^{'}$, then let $o_r^{'} = o_r^{''}$ and return to 2).

Step 4. Let $\overline{\boldsymbol{r}}_z = (\underbrace{1, 1, \cdots, 1}_{R}, \underbrace{0, \cdots, 0}_{M-R})^{\mathrm{T}}$, then $\overline{\delta}^k$ $(k = 1, 2, \cdots, M)$ can be computed by

$$(\overline{\delta}^1, \overline{\delta}^2, \cdots, \overline{\delta}^M)^{\mathrm{T}} = \overline{Q}_z \overline{\boldsymbol{r}}_z$$

References

1 Etemadi-Zanganeh K, Angeles J. Real-time direct kinematics of general six-degree of freedom parallel manipulators with minimum-sensor data. *Journal of Robotic Systems*, 1995, **12**(12): 833−844

2 Arai T, Yuasa K, Mae Y, Inoue K, Miyawaki K, Koyachi N. A hybrid drive parallel arm for heavy material handling. *IEEE Robotics and Automation Magazine*, 2002, **9**(1): 45−54

3 Zhang X C, Yi J Q, Zhao D B, Yang G S. Modelling and control of a self-levelling crane. In: Proceedings of International Conference on Mechatronics and Automation. Harbin, China: IEEE, 2007. 2922−2927

4 Zhang J H, Zhao D B, Yi J Q, Yu Y. Modeling of a rope-driven self-levelling crane. In: Proceedings of the 3rd International Conference on Innovative Computing Information and Control. Dalian, China: IEEE, 2008. 489−492

5 Hussain M A, Kershenbaum L S. Implementation of an inverse-model-based control strategy using neural networks on a partially simulated exothermic reactor. *Chemical Engineering Research and Design*, 2000, **78**(2): 299−311

6 Daosud W, Thitiyasook P, Arpornwichanop A, Kittisupakorn P, Hussain M A. Neural network inverse model-based controller for the control of a steel pickling process. *Computers and Chemical Engineering*, 2005, **29**(10): 2110−2119

7 Juang C F, Chen J S. A recurrent fuzzy-network-based inverse modeling method for a temperature system control. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2007, **37**(3): 410−417

8 Widrow B, Walach E. *Adaptive Inverse Control: A Signal Processing Approach, Reissue Edition*. New York: Wiley-IEEE Press, 2007. 111−159

9 Mendel J M. *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*. New Jersey: Prentice-Hall, 2001. 287−350

10 Karnik N N, Mendel J M, Liang Q L. Type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems*, 1999, **7**(6): 643−658

11 Liang Q L, Mendel J M. Interval type-2 fuzzy logic systems: theory and design. *IEEE Transactions on Fuzzy Systems*, 2000, **8**(5): 535−550

12 Mendel J M. Advances in type-2 fuzzy sets and systems. *Information Sciences*, 2007, **177**(1): 84−110

13 Mendel J M. Computing derivatives in interval type-2 fuzzy logic systems. *IEEE Transactions on Fuzzy Systems*, 2004, **12**(1): 84−98

14 Hagras H A. A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. *IEEE Transactions on Fuzzy Systems*, 2004, **12**(4): 524−539

15 Liu Z, Zhang Y, Wang Y. A type-2 fuzzy switching control system for biped robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 2007, **37**(6): 1202−1213

16 Melin P, Castillo O. An intelligent hybrid approach for industrial quality control combining neural networks, fuzzy logic and fractal theory. *Information Sciences*, 2007, **177**(7): 1543−1557

17 Lee C H, Lin Y C. An adaptive type-2 fuzzy neural controller for nonlinear uncertain systems. *Control and Intelligent Systems*, 2005, **33**(1): 13−25

18 Wang C H, Cheng C S, Lee T T. Dynamical optimal training for interval type-2 fuzzy neural network (T2FNN). *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2004, **34**(3): 1462−1477

19 Hagras H. Comments on "dynamical optimal training for interval type-2 fuzzy neural network (T2FNN)". *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2006, **36**(5): 1206−1209

20 Li C D, Yi J Q, Zhao D B. Interval type-2 fuzzy neural network controller (IT2FNNC) and its application to a coupled-tank liquid-level control system. In: Proceedings of the 3rd International Conference on Innovative Computing Information and Control. Dalian, China: IEEE, 2008. 508−511

21 Nelles O. *Nonlinear System Identification*. Berlin: Springer-Verlag, 2001. 35−67

22 Jang J S R. Anfis: adaptive-network-based fuzzy inference system. *IEEE Transactions on Systems, Man, and Cybernetics*, 1993, **23**(3): 665−685

LI Cheng-Dong Ph. D. candidate at the Institute of Automation, Chinese Academy of Sciences. His research interest covers computational intelligence and type-2 fuzzy logic. Corresponding author of this paper. E-mail: chengdong.li@ia.ac.cn

YI Jian-Qiang Professor at the Institute of Automation, Chinese Academy of Sciences. His research interest covers intelligent control and computational intelligence.
E-mail: jianqiang.yi@ia.ac.cn

YU Yi Ph. D. candidate at the Institute of Automation, Chinese Academy of Sciences. His main research interest is intelligent control. E-mail: yi.yu@ia.ac.cn

ZHAO Dong-Bin Associate professor at the Institute of Automation, Chinese Academy of Sciences. His research interest covers computational intelligence, intelligent transportation, and robotics. E-mail: dongbin.zhao@ia.ac.cn