Engineering Applications of Artificial Intelligence 42 (2015) 36–44

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Genetic-algorithm-based type reduction algorithm for interval type-2 fuzzy logic controllers

Tzyy-Chyang Lu

Advanced Institute of Manufacturing with High-Tech Innovations, National Chung Cheng University, Chiayi, Taiwan

ARTICLE INFO

Article history:
Received 27 October 2014
Received in revised form
6 January 2015
Accepted 23 February 2015
Available online 2 April 2015

Keywords:
Interval type-2 fuzzy logic controllers (IT2-FLCs)
Type reduction
Optimization

ABSTRACT

In interval type-2 fuzzy logic controllers (IT2-FLCs), the output processing includes type reduction and defuzzification. Recently, researchers have proposed many efficient type reduction algorithms, but there are no effective schemes to improve the output of defuzzification. This paper presents a genetic-algorithm-based type reduction algorithm, which reduces the type of an interval type-2 fuzzy set and provides optimal defuzzified output from the type-reduced set. In addition, the proposed type reduction is executed offline (in other words, the controller has been reduced to type-1 in practical applications), which significantly reduces the computational cost and facilitates the design of controllers that operate in real time. To demonstrate the effectiveness of the proposed method, truck backing control problems are utilized. The results show that the proposed method outperforms general IT2-FLCs in terms of speed, computational cost, and robustness.

1. Introduction

Fuzzy logic controllers (FLCs) have been successfully applied to a wide variety of applications. There have been some publications in the design of type-1 fuzzy logic controller (T1-FLC). For example, Bingul and Karahan (2011) and Bouallegue et al. (2012) employ the particle swarm optimization algorithm to tune the FLC structures, which applied to the robot trajectory control and the electrical DC drive speed control, respectively. Cazarez-Castro et al. (2012) use fuzzy Lyapunov synthesis to design the FLCs to solve the output regulation problem of a servomechanism with non-linear backlash. Mendes et al. (2014) use a hierarchical genetic algorithm (GA) to automatically extract all fuzzy parameters of a FLC in order to control nonlinear industrial processes.

Interval type-2 fuzzy logic systems are an extension of traditional type-1 fuzzy logic systems. When the information is so fuzzy that even defining the membership function values in the interval [0,1] is difficult, a type-2 membership function is beneficial (Hosseini et al., 2012). A type-2 membership function can be considered as a collection of different embedded type-1 fuzzy sets, which construct the footprint of uncertainty (FOU) (Hu et al., 2012). Mendel (2001) and Hagras (2004) have shown that using type-2 fuzzy sets will result in the reduction of the rule base compared to that obtained using type-1 fuzzy sets. Furthermore, the extra degrees of freedom provided by the FOU enables an interval type-2 fuzzy logic controller (IT2-FLC) to produce outputs that cannot be achieved by a T1-FLC with the same number of membership functions. Thus, an IT2-FLC is able to model more complex input–output relationships than its type-1 counterpart and, thus, can give better control response (Coupland et al., 2006; Biglarbegian et al., 2009). Recently, there has been a growing interest in using IT2-FLC in many applications as well as in control processes (Biglarbegian et al., 2011; Hosseini et al., 2012; Cara et al., 2013).

The major difference between a T1-FLC and an IT2-FLC is that for the latter, at least one of the membership functions in the rule base is a type-2 membership function (Sepulveda et al., 2007). Hence, the inference engine outputs are type-2 fuzzy sets, and type reduction is needed to convert them into a type-reduced set (Wu, 2006, 2013). A type-reduced set is an interval type-1 set defined by the left and right centroids (Karnik and Mendel, 1999, 2001). The crisp output can be any value from this interval set depending on uncertainties (Ulu et al., 2011). Karnik–Mendel algorithms (KMAs) (Karnik and Mendel, 2001) are the most popular type reduction approach for computing the boundary centroids. However, they suffer from two major shortcomings. First, KMAs are iterative algorithms that use a trial-and-error process of testing various centroids to find the boundary ones for all input values. This is time-consuming and thus cannot be realized in real time. Second, in the defuzzification process, KMAs take the average of these boundary centroids as the defuzzified output, which may not be a good choice.

In recent years, some variants of type reduction algorithms have been proposed. These algorithms can be classified into two categories: enhanced KMAs (EKMAs) (Wu and Mendel, 2009; Yeh

E-mail address: tclu2012@gmail.com

http://dx.doi.org/10.1016/j.engappai.2015.02.012
0952-1976/

et al., 2011; Hu et al., 2010, 2012) and alternative type reduction algorithms (ATRAs) (Gorzalczany, 1987; Liang and Mendel, 2000; Wu and Mendel, 2002; Wu and Tan, 2005; Coupland and John, 2007; Greenfield et al., 2008; Nie and Tan, 2008; Du and Ying, 2010; Tao et al., 2012). EKMAs improve directly over the original KMAs, obtaining the same boundary centroids using fewer iterative computations. However, in the defuzzification process, they still take the average of the left and right centroids as a defuzzified output. In other words, EKMAs give the exact same outputs as those of the original KMAs but faster. Unlike EKMAs, ATRAs have closed-form representations, which give the approximate outputs from KMAs. These algorithms are usually much faster (but not necessarily have better control response) than the original KMAs. Ulu et al. (2011) proposed a dynamic defuzzification method that uses a linear combination of the boundary centroids to enhance the control response. However, since the boundary centroids are still computed using KMAs, the dynamic defuzzification method has the same computational load as that of the general defuzzification method.

Instead of taking the average of the left and right centroids, the present work employs GA to find the optimal defuzzified output from the type-reduced set. In addition, the GA-based type reduction is executed offline (in other words, the controller has been reduced to type-1 in practical applications), which significantly reduces the computational cost and facilitates the design of controllers that operate in real time. Finally, and most importantly, the designed controller is type-1 in practical applications but has the complex control surfaces like type-2. This means that the designed controller possesses more degrees of freedom in design aspects like a type-2 controller.

The rest of this paper is organized as follows. Section 2 briefly reviews IT2-FLCs. Section 3 describes the proposed genetic-algorithm-based type reduction. Section 4 shows the results of the proposed method applied to the truck backing up control problem. Finally, Section 5 gives the conclusions.

2. Interval type-2 fuzzy logic controllers

An interval type-2 fuzzy set (IT2-FS) $\tilde{A}$ is characterized by an interval type-2 membership function as (Wu, 2013)

$$\tilde{A} = \int_{x \in X} \int_{u \in J_x \subseteq [0,1]} 1/(x,u) \tag{1}$$

where x is the primary variable in domain X, $u \in [0,1]$ is the secondary variable in domain J_x at each $x \in X$, and J_x is called the primary membership of x. The secondary grades of $\tilde{A}$ are all equal to 1.

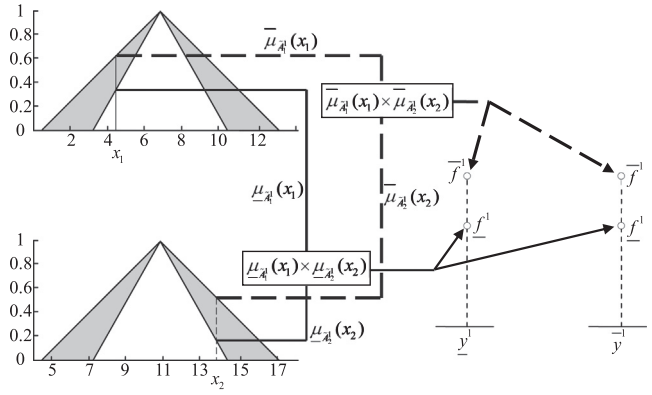

Fig. 1. Two interval type-2 membership functions of antecedent part for rule $\tilde{R}^1$ (left). Weighing interval sets and corresponding fired interval type-2 sets for first output (right).

Consider the rule base of an IT2-FLC composed of M rules with the following form:

$$\tilde{R}^i: \text{ IF } x_1 \text{ is } \tilde{A}_1^i \text{ and } \ldots \text{ and } x_n \text{ is } \tilde{A}_n^i, \text{ THEN } y \text{ is } Y^i$$

where $i = 1, 2, \ldots, M$, $\tilde{A}_j^i$, $j = 1, \ldots, n$, is an IT2-FS, and $Y^i = [\underline{y}^i, \overline{y}^i]$ is an interval, which can be understood as the centroid of a consequent IT2-FS, or the simplest Takagi–Sugeno–Kang (TSK) model (Wu, 2013).

For an input vector $\boldsymbol{x} = x_j$, $j = 1, \ldots, n$, the membership degree $\mu_{\tilde{A}_j^i}(x_j)$ is an interval set, denoted by

$$\mu_{\tilde{A}_j^i}(x_j) = \left[\underline{\mu}_{\tilde{A}_j^i}(x_j), \overline{\mu}_{\tilde{A}_j^i}(x_j)\right] \tag{2}$$

where $i = 1, \ldots, M$, $j = 1, \ldots, n$. As only interval type-2 sets are used and the *meet* operation is implemented with the *product* t-norm, the firing set is the following type-1 interval set (Wu and Tan, 2006):

$$F^i(\boldsymbol{x}) = \left[\prod_{j=1}^{n} \underline{\mu}_{\tilde{A}_j^i}(x_j), \prod_{j=1}^{n} \overline{\mu}_{\tilde{A}_j^i}(x_j)\right] = \left[\underline{f}^i, \overline{f}^i\right] \tag{3}$$

where $i = 1, \ldots, M$. An example of two interval type-2 membership functions in the antecedent part for rule $\tilde{R}^1$ and the corresponding fired interval type-2 sets for the first output are shown in Fig. 1.

Type reduction is performed to combine $F^i(\boldsymbol{x})$ and the corresponding rule consequents. The center-of-sets type reduction (Karnik and Mendel, 2001; Shill et al., 2012) is used in this paper. Then, $C(\boldsymbol{x})$ is all possible combinations of the centroids:

$$C(\boldsymbol{x}) = \bigcup_{\substack{f^i \in F^i(\boldsymbol{x}) \\ y^i \in Y^i}} \frac{\sum_{i=1}^{M} f^i y^i}{\sum_{i=1}^{M} f^i} = [c_{\min}, \ c_{\max}] \tag{4}$$

where

$$c_{\min} = \min_{L \in [1, M-1]} \frac{\sum_{i=1}^{L} \overline{f}^i \underline{y}^i + \sum_{i=L+1}^{M} \underline{f}^i \underline{y}^i}{\sum_{i=1}^{L} \overline{f}^i + \sum_{i=L+1}^{M} \underline{f}^i} \tag{5}$$

$$c_{\max} = \max_{R \in [1, M-1]} \frac{\sum_{i=1}^{R} \underline{f}^i \overline{y}^i + \sum_{i=R+1}^{M} \overline{f}^i \overline{y}^i}{\sum_{i=1}^{R} \underline{f}^i + \sum_{i=R+1}^{M} \overline{f}^i} \tag{6}$$

L and R are switch points, and $\left\{\underline{f}^i\right\}$ and $\left\{\overline{f}^i\right\}$ have been sorted in ascending order (Mendel, 2001; Mendel and Wu, 2010). L and R can be computed using KMAs or their variants. Take $c_{\min}$ of Fig. 2(a) as an example. An iterative procedure finds the switch point L satisfying:

$$\underline{y}^L \le c_{\min} < \underline{y}^{L+1} \tag{7}$$

For $i \le L$, the upper bounds of the firing intervals are used to calculate $c_{\min}$; for $i > L$, the lower bounds are used. This ensures that $c_{\min}$ is the minimum. A similar procedure can be used to find the switch point R to ensure that $c_{\max}$ is the maximum (see Fig. 2(b)). Finally, the crisp output can be obtained by taking the average of $c_{\min}$ and $c_{\max}$:

$$y = \frac{c_{\min} + c_{\max}}{2} \tag{8}$$

3. Genetic-algorithm-based type reduction

According to the above analysis, the defuzzified output y is determined by the boundary centroids in (4). However, taking the average of these centroids as a defuzzified output may not be a good choice. In this work, the boundary centroids are replaced by the

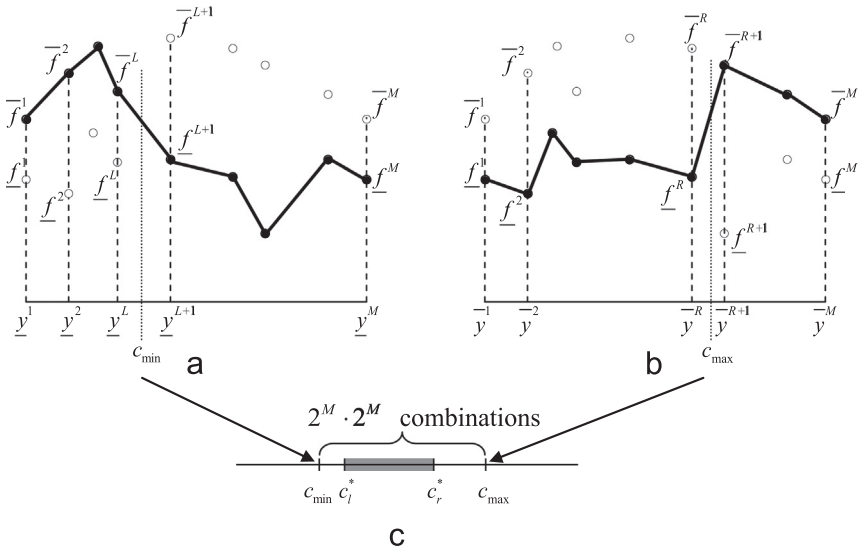

Fig. 2. Type reduction. (a) Compution of minimal centroid $c_{\min}$. (b) Compution of maximal centroid $c_{\max}$. (c) Type-reduced set (interval type-1 set).

optimal centroids c_l^* and c_r^* (see Fig. 2(c)), which are respectively obtained as

$$c_l^* = \bigcup_{f^i \in \{\underline{f}^i, \overline{f}^i\}} \frac{\sum_{i=1}^{M} f^i \underline{y}^i}{\sum_{i=1}^{M} f^i} \tag{9}$$

and

$$c_r^* = \bigcup_{f^i \in \{\underline{f}^i, \overline{f}^i\}} \frac{\sum_{i=1}^{M} f^i \overline{y}^i}{\sum_{i=1}^{M} f^i} \tag{10}$$

Finally, the optimal defuzzified output is expressed as

$$y^* = \frac{c_l^* + c_r^*}{2} \tag{11}$$

3.1. Type reduction model and encoding scheme

For a given input vector $\boldsymbol{x}$, both (9) and (10) have 2^M combinations. These combinations can be respectively represented by M binary bits b_i, $i = 1, 2, \ldots, M$, where "0" indicates that the upper bound $\overline{f}^i$ is selected and "1" indicates that the lower bound $\underline{f}^i$ is selected. Take $M = 2$ in (9) as an example. When an input vector $\boldsymbol{x}$ is given (so that $\left\{\underline{f}^i\right\}_{i=1,2}$ and $\left\{\overline{f}^i\right\}_{i=1,2}$ are determined by (3)), binary strings [01], [00], [11], and [10] can respectively represent:

$$c_{l_1} = \frac{\overline{f}^1 \underline{y}^1 + \underline{f}^2 y^2}{\overline{f}^1 + \underline{f}^2}, \quad c_{l_2} = \overline{f}^1 \underline{y}^1 + \frac{\overline{f}^2 y^2}{\overline{f}^1 + \overline{f}^2},$$

$$c_{l_3} = \frac{\underline{f}^1 y^1 + \underline{f}^2 y^2}{\underline{f}^1 + \underline{f}^2}, \quad c_{l_4} = \underline{f}^1 \underline{y}^1 + \frac{\overline{f}^2 \underline{y}^2}{\underline{f}^1 + \overline{f}^2}$$

as shown in Fig. 3.

To find the optimal centroids c_l^* and c_r^*, the binary string:

$$s = [b_1 \ldots b_M b_{M+1} \ldots b_{2M}] \tag{12}$$

is used to represent a type reduction model, where $b_1 \ldots b_M$ represents the combination in (9) and $b_{M+1} \ldots b_{2M}$ represents the combination in (10).

3.2. Partitioning strategy

In an IT2-FLC, a different input value may fire different membership functions and rule consequents, and thus there are different optimal combinations of upper and lower bounds of the firing intervals (type-reduced set). For this reason, each input interval is divided into regions, and various type reduction models are created depending on these regions.

To find the global optimal type reduction models, the binary solution S can be expressed as

$$S = [s^1 \quad \ldots \quad s^D] \tag{13}$$

where $s^j = \left[b_1^j \ldots b_M^j b_{M+1}^j \ldots b_{2M}^j\right]$ is a type reduction model, as in (12), where $j = 1, \ldots, D$, and D is the total number of type reduction models. For an n-input IT2-based T1-FLC, if the input intervals are divided respectively into d_1, d_2, $\ldots d_n$ regions, the total number of type reduction models is

$$D = \prod_{j=1}^{n} d_j \tag{14}$$

Take a two-input IT2-FLC with seven rules as an example. If the inputs are both divided into three regions, there will be $3^2 = 9$ type reduction models. In this case, the binary solution can be expressed as

$$S = [s^1 \quad \ldots \quad s^9] \tag{15}$$

where $s^j = \left[b_1^j \ldots b_{14}^j\right]$ and $j = 1, \ldots, 9$. Fig. 4 shows the 1st and 9th type reduction models for this example.

3.3. Simple genetic algorithm

Genetic algorithms (GAs) are powerful search and optimization algorithms based on the principals of natural evolution. They maintain a population of chromosomes (individuals), which represent potential solutions to the problem. Here, a chromosome is a binary string, as in (13).

Although there are many variations of GAs, a simple GA is applied to the optimization of type reduction models. The procedure of the GA can be summarized as follows:

Step 1: Randomly generate an initial population $G(t)$.
Step 2: Evaluate the fitness function of each individual of the current population $G(t)$.

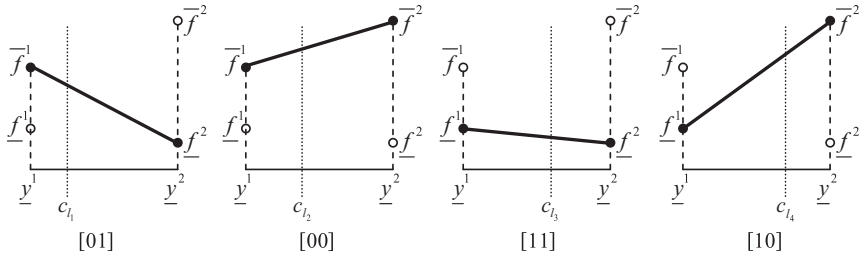

Fig. 3. Example of $M=2$ in (9).

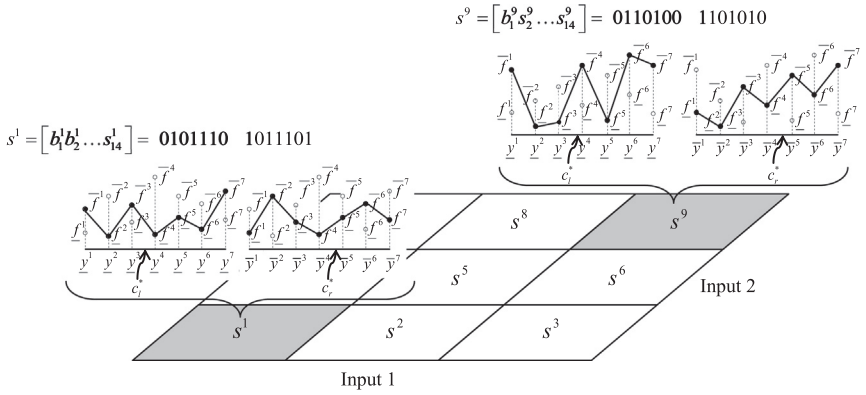

Fig. 4. Example of two-input IT2-based T1-FLC with seven rules. Each input interval is divided into three regions and nine type reduction models are created. In s^j, $j=1, 2,\ldots, 9$, "0" indicates that the upper bound $\overline{f}^i$, $i=1, 2,\ldots, 7$, is selected and "1" indicates that the lower bound $\underline{f}^i$, $i=1, 2,\ldots, 7$, is selected.

Step 3: Create an intermediate population $I(t+1)$ by applying the reproduction operators to $G(t)$.
Step 4: Generate a new population $G(t+1)$ by applying the genetic operators of crossover and mutation to this intermediate population $I(t+1)$.
Step 5: $t=t+1$; If (NOT_END), go to step 2.

4. Simulations and results

The truck backing problem is a well-known control problem that is generally used as a benchmark for the evaluation of new control algorithms (Wang et al., 2004; Shill et al., 2012). The path planning of the truck is determined by x, y, and ϕ, where x and y are the horizontal and vertical positions, respectively, and ϕ is the angle of the truck with respect to the horizontal axis. The truck is controlled by a steering angle θ. In the simulation, enough clearance is assumed so that y does not have to be considered as an input. The input ranges considered in this simulation are $x\in[-10,\ 10]$ and $\phi\in[-150,\ 150]$, and the output range is $\theta\in[-8,\ 8]$.

The truck kinematics model is described by the following dynamic equations (Wang et al., 2004).

$$x(t+1)=x(t)+\cos[\phi(t)+\theta(t)]+\sin[\theta(t)]\sin[\phi(t)] \tag{16}$$

$$y(t+1)=y(t)+\sin[\phi(t)+\theta(t)]-\sin[\theta(t)]\sin[\phi(t)] \tag{17}$$

$$\phi(t+1)=\phi(t)-\sin^{-1}\left[\frac{2\sin(\theta(t))}{b}\right] \tag{18}$$

where $b=4$ is the length of the truck. The control objective is to back up the truck to a desired position ($x_{desired}=0$) and angle ($\phi_{desired}=90^\circ$). The simulation sample time step is 0.1 s. Since y is not considered, (17) is not used. In the GA, population size is 50, the maximum number of generations is 20, crossover probability is 0.9, and mutation probability is 0.2. The fitness function is defined as

$$\text{fitness}=\frac{1}{1+\text{err}} \tag{19}$$

$$\text{err}=\sum_{t=1}^{N}\left(|x(t)-x_{desired}|+|\phi(t)-\phi_{desired}|\right) \tag{20}$$

where N is the total number of simulation steps.

In the fuzzy controller, each input and output domain consists of seven interval type-2 membership functions, which are optimized, as shown in Fig. 5. The rule base and the corresponding consequents are shown in Table 1. Both Δx and $\Delta\phi$ are divided into five uniform intervals, for a total of 25 type reduction models.

In this study, root mean square error (RMSE) is used as a performance criterion, which is defined as follows:

$$\text{RMSE}_{(position)}=\sqrt{\frac{\sum_{i=1}^{M}\sum_{t=1}^{N_i}\left(x^i(t)-x_{desired}\right)^2}{\sum_{i=1}^{M}N_i}} \tag{21}$$

$$\text{RMSE}_{(angle)}=\sqrt{\frac{\sum_{i=1}^{M}\sum_{t=1}^{N_i}\left(\phi^i(t)-\phi_{desired}\right)^2}{\sum_{i=1}^{M}N_i}} \tag{22}$$

where M is the total number of initial positions and N_i is the total number of simulation steps for ith, $i=1, 2, \ldots, M$, initial position.

4.1. Clean environment

Two initial positions, $(x_0, y_0, \phi_0)=(10, 0, 140)$ and $(-10, 600, 70)$, are used to generate a series of training data pairs as desired input–output pairs. Fig. 6 shows the trajectories under the control of KMA, the Nie-Tan (NT) method (Nie and Tan, 2008), and the proposed method. Since EKMAs give the same outputs as those of KMA, they are not considered separately in this section. The NT method is a

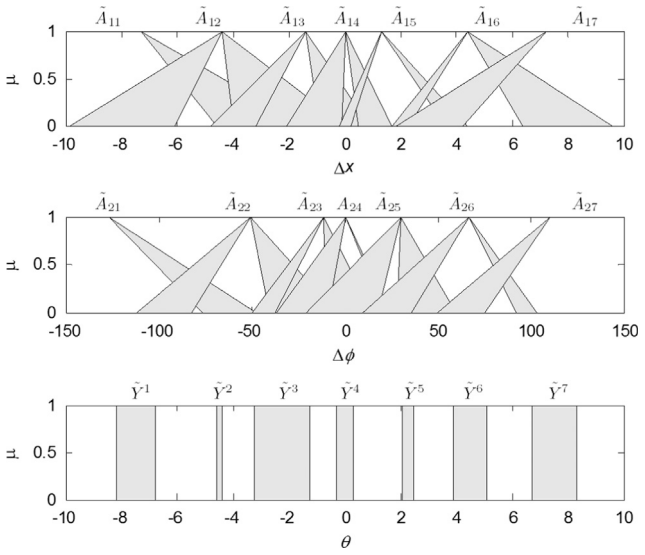

Fig. 5. Optimized membership functions and singletons for input and output variables of IT2-FLC.

Table 1
Rule base and consequents of IT2-FLC.

	$\tilde{A}_{21}$	$\tilde{A}_{22}$	$\tilde{A}_{23}$	$\tilde{A}_{24}$	$\tilde{A}_{25}$	$\tilde{A}_{26}$	$\tilde{A}_{27}$
$\tilde{A}_{11}$	$\tilde{Y}^4$	$\tilde{Y}^3$	$\tilde{Y}^2$	$\tilde{Y}^2$	$\tilde{Y}^1$	$\tilde{Y}^1$	$\tilde{Y}^1$
$\tilde{A}_{12}$	$\tilde{Y}^5$	$\tilde{Y}^4$	$\tilde{Y}^3$	$\tilde{Y}^2$	$\tilde{Y}^2$	$\tilde{Y}^7$	$\tilde{Y}^7$
$\tilde{A}_{13}$	$\tilde{Y}^6$	$\tilde{Y}^5$	$\tilde{Y}^4$	$\tilde{Y}^3$	$\tilde{Y}^2$	$\tilde{Y}^2$	$\tilde{Y}^1$
$\tilde{A}_{14}$	$\tilde{Y}^6$	$\tilde{Y}^6$	$\tilde{Y}^4$	$\tilde{Y}^4$	$\tilde{Y}^3$	$\tilde{Y}^2$	$\tilde{Y}^2$
$\tilde{A}_{15}$	$\tilde{Y}^7$	$\tilde{Y}^6$	$\tilde{Y}^6$	$\tilde{Y}^5$	$\tilde{Y}^4$	$\tilde{Y}^3$	$\tilde{Y}^2$
$\tilde{A}_{16}$	$\tilde{Y}^7$	$\tilde{Y}^7$	$\tilde{Y}^6$	$\tilde{Y}^6$	$\tilde{Y}^5$	$\tilde{Y}^4$	$\tilde{Y}^3$
$\tilde{A}_{17}$	$\tilde{Y}^1$	$\tilde{Y}^1$	$\tilde{Y}^1$	$\tilde{Y}^6$	$\tilde{Y}^6$	$\tilde{Y}^5$	$\tilde{Y}^4$

special case of the Wu-Tan (WT) method, with the weighting h_i^n in Wu and Tan (2005) of 0.5; they are the fastest ATRAs (Wu, 2013). It can be seen from Fig. 6 that the NT (or WT) method does not significantly outperform KMA (and thus EKMAs). Take the following two cases as examples. KMA and the NT method respectively need 94 steps and 110 steps to move the truck to a satisfactory position ($|\Delta x|<0.2$, $|\Delta\phi|<1$) from $(x_0, y_0, \phi_0)=(10, 0, 140)$, and respectively need 78 steps and 94 steps from $(x_0, y_0, \phi_0)=(-10, 600, 70)$. The proposed method requires only 60 steps and 46 steps for these tasks, respectively. In addition, the proposed method is energy-efficient. As shown in Fig. 6(b), the proposed method reduces the energy nearly to zero, whereas KMA creates chattering output. This chattering phenomenon is caused by the non-smooth control surfaces, which will be discussed in Section 4.3. Table 2 shows the RMSE of the training data. As can be seen, both the position and angle errors of the proposed method are smaller than that of KMA and the NT method.

Different input values may fire different membership functions and rule consequents, and thus there are different optimal type reduction models. It is believed that a fine partitioning of input intervals will enhance the performance of the designed controller. Cases with 1, 25, and 49 type reduction models, whose input intervals are respectively divided into 1 (i.e., no partitioning is used), 5, and 7 regions, were compared. Fig. 7 shows the training results from two initial positions, $(x_0, y_0, \phi_0)=(10, 0, 140)$ and $(-10, 600, 70)$. Clearly, an increase in the number of divisions leads to a finer type reduction model resolution, which decreases training error.

To further test the robustness of the proposed method, various initial positions were used for the well-trained controllers. Table 3

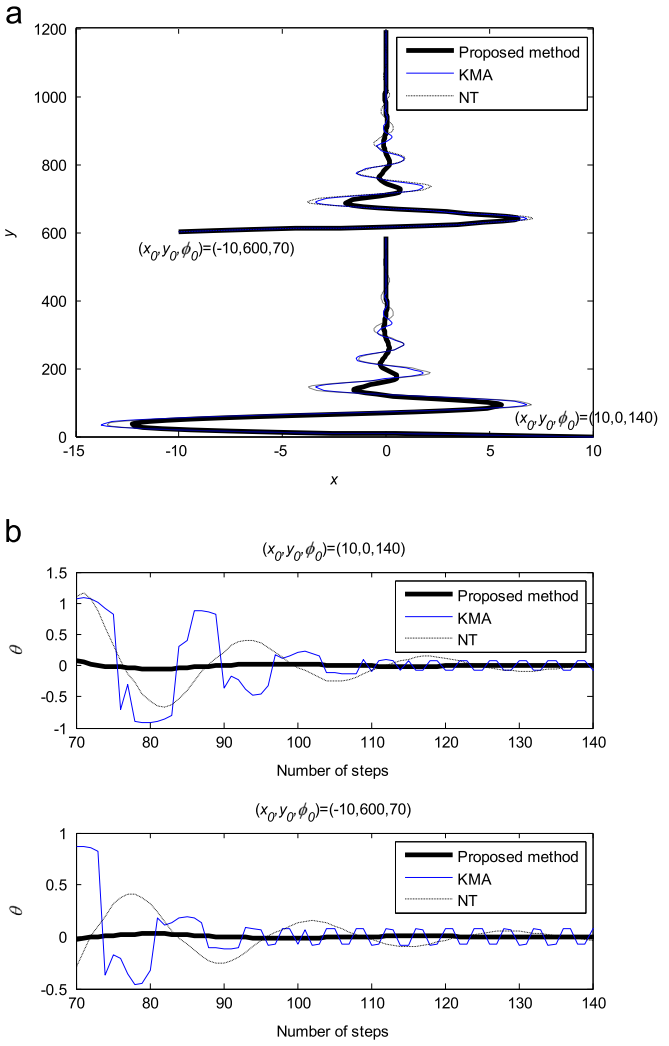

Fig. 6. Training results of KMA, NT, and the proposed method with initial positions $(x_0, y_0, \phi_0)=(10, 0, 140)$ and $(-10, 600, 70)$. (a) Control trajectories. (b) Control output from 70 to 140 steps.

Table 2
Training RMSE of KMA, NT, and the proposed method with initial positions $(x_0, y_0, \phi_0)=(10, 0, 140)$ and $(-10, 600, 70)$.

	KMA	NT	Proposed method
RMSE (position)	0.117	0.0937	0.0130
RMSE (angle)	0.885	0.3368	0.0324

shows the steps required for the truck to arrive at a satisfactory position ($|\Delta x|<0.2$, $|\Delta\phi|<1$) for 14 initial positions. Some of the trajectories are plotted in Fig. 8. The results were compared with those obtained by KMA and the NT method. Clearly, the proposed method requires less number of steps to drive the truck to the assigned position for all initial positions. The higher the number of divisions, the less number of steps is needed to complete the task. The partitioning strategy thus enhances the performance of the controller. Table 4 shows the RMSE of the test data.

To compare the computational cost of the proposed method with that of KMA and ATRAs,[1] the computation time of KMA, the NT method (fastest ATRA), and the proposed method was recorded

[1] Wu (2013) demonstrated that all ATRAs are faster than EKMAs. Therefore, EKMAs are not considered in this section.

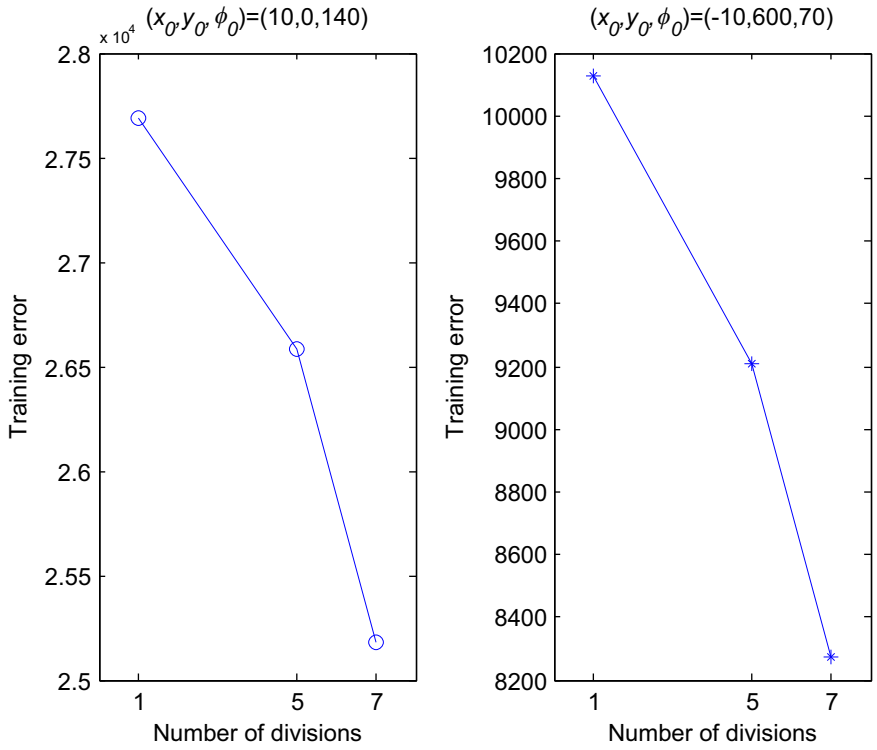

Fig. 7. Training error for various numbers of divisions. *X* axis is the number of divisions for two inputs (Δx and $\Delta\phi$), where 1, 5, and 7 represent 1, 25, and 49 type reduction models, respectively.

Table 3
Steps required for truck to arrive at satisfactory position ($|\Delta x| < 0.2, |\Delta\phi| < 1$) for 14 initial positions (best result among the methods is highlighted in bold for each initial position).

Initial position (x_0, y_0, θ_0)	KMA	NT	Proposed method		
			$D=1$	$D=5$	$D=7$
$(-9, 60, -5)$	139	137	103	89	**86**
$(-9, 60, 85)$	82	84	60	46	**45**
$(-9, 60, 175)$	119	135	102	97	**94**
$(-3, 70, -5)$	120	136	101	86	**84**
$(-3, 70, 85)$	64	68	39	30	**29**
$(-3, 70, 175)$	130	132	99	95	**90**
$(-3, 70, -95)$	151	152	141	125	**123**
$(3, 80, -5)$	120	136	101	86	**84**
$(3, 80, 85)$	51	70	39	34	**33**
$(3, 80, 175)$	137	131	99	94	**86**
$(3, 80, 265)$	147	152	130	125	**122**
$(9, 90, 95)$	82	84	60	52	**48**
$(9, 90, 185)$	139	137	104	99	**98**
$(9, 90, 265)$	152	152	139	134	**131**

for 150 steps. All simulations are performed on an Intel 3.0 GHz dual CPU, and the programs are written in MATLAB. The results are shown in Table 5. Clearly, owing to its iterative nature, KMA has a high computational cost. The computational cost of the NT method is significantly lower than that of KMA because the former uses a closed-form type reduction method without iterative computation. The proposed method, whose results are similar to those for the NT method, performs the type reduction offline and makes use of the equivalent type-1 membership grades. This leads to lower computational time, making the proposed method suitable for real-time control. In addition, the proposed method provides better output than those of the NT method and KMA.

4.2. Noisy environment

In practical applications, controllers are designed using a simulation model without measurement noise. However, actual sensors exhibit noise. To examine the noise resistance abilities of the proposed method, the simulations in this section assume that the inputs are noise-free during the design process and noisy in practice. The noise in x and ϕ is uniformly distributed in the intervals $[-0.2, 0.2]$ and $[-1^o, 1^o]$, respectively. The steps required for the truck to arrive at a satisfactory position ($|\Delta x| < 0.2$, $|\Delta\phi| < 1$) for 14 initial positions is shown in Table 6. The results obtained using KMA and the NT method under the same conditions are also shown in the table, where "-" means that the truck did not arrive at a satisfactory position in 150 steps (15.0 s). The best result among the methods is highlighted in bold for each initial position. Clearly, the proposed method is more robust to noise than are KMA and the NT method. Some trajectories under the control of KMA and the NT method are outside the restricted range interval $[-0.2, 0.2]$ (see Fig. 9).

As shown in the table, the results of the proposed method under no partitioning (i.e., $D=1$) are superior to those of KMA and the NT method, except that for the initial position $(x_0, y_0, \phi_0) = (9, 90, 265)$, in terms of required steps. This shows that the proposed optimization strategy not only improves the performance of IT2-FLCs, but also increases robustness to noise. Additionally, the required steps for the proposed method decreases with increasing number of partitions. This indicates that the partitioning strategy leads to higher noise resistance. Table 7 shows the RMSE of the noisy test data.

4.3. Control surface

The control surface, which graphically represents the unknown function articulated by the fuzzy logic controller, can be used to

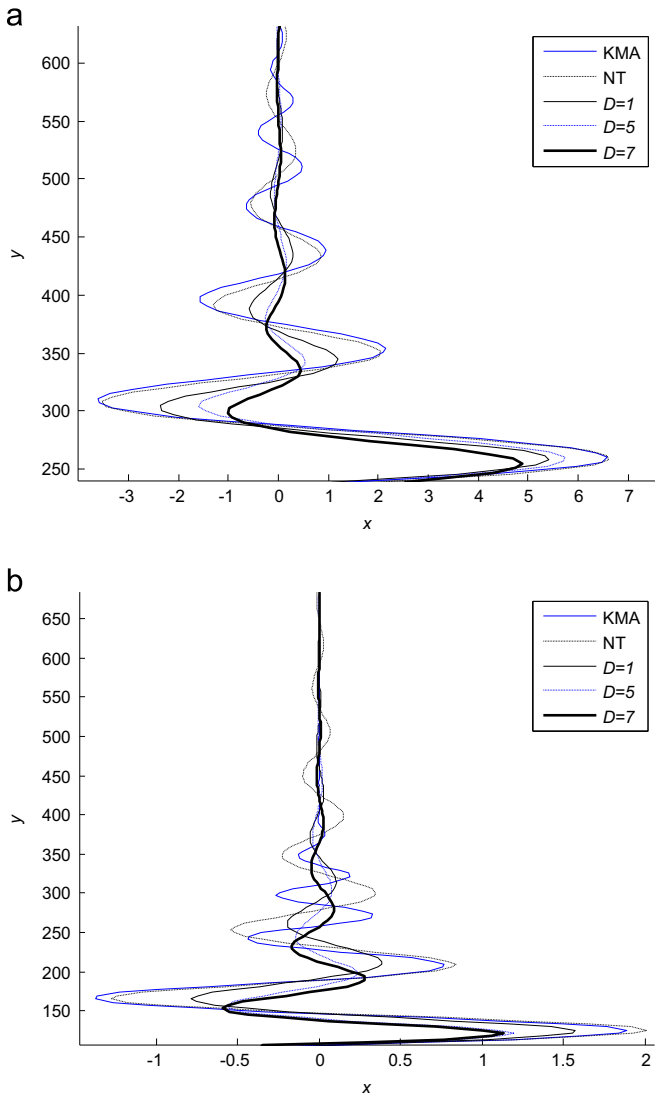

Fig. 8. Truck backing trajectories under control of KMA, NT method, and proposed method (D is the number of divisions) for (a)$(x_0, y_0, \phi_0) = (-3, 70, -95)$ and (b)$(x_0, y_0, \phi_0) = (-9, 60, -5)$.

Table 4
Test RMSE of KMA, NT, and the proposed method for 14 initial positions.

	KMA	NT	Proposed method		
			$D=1$	$D=5$	$D=7$
RMSE (position)	0.6091	0.6216	0.1664	0.1428	0.1091
RMSE (angle)	2.6125	2.5492	1.3100	1.2000	1.0533

Table 5
Computational time of KMA, NT method, and proposed method for 150 Steps.

KMA (s)	NT (s)	Proposed method		
		$D=1$ (s)	$D=5$ (s)	$D=7$ (s)
0.34	0.15	0.13	0.14	0.15

determine the response of each controller. Since an IT2-FLC aggregates the outputs of a large number of embedded type-1 fuzzy sets, its control surface (see Fig. 10(a)) is generally more complex than that of type-1 controllers (see Fig. 10(b)). A detailed

Table 6
Steps required for truck to arrive at satisfactory position ($|\Delta x| < 0.2, |\Delta\phi| < 1$) with noisy test data (best result among the methods is highlighted in bold for each initial position).

Initial position (x_0, y_0, θ_0)	KMA	NT	Proposed method		
			$D=1$	$D=5$	$D=7$
$(-9, 60, -5)$	–	141	106	93	**89**
$(-9, 60, 85)$	–	95	71	52	**48**
$(-9, 60, 175)$	–	–	123	106	**104**
$(-3, 70, -5)$	134	137	104	**89**	**89**
$(-3, 70, 85)$	148	79	50	48	**36**
$(-3, 70, 175)$	143	144	123	104	**97**
$(-3, 70, -95)$	–	–	145	**134**	**134**
$(3, 80, -5)$	–	–	123	100	**99**
$(3, 80, 85)$	–	80	54	42	**39**
$(3, 80, 175)$	142	–	109	97	**88**
$(3, 80, 265)$	–	–	139	130	**125**
$(9, 90, 95)$	–	114	73	61	**56**
$(9, 90, 185)$	–	–	124	111	**107**
$(9, 90, 265)$	–	–	–	148	**140**

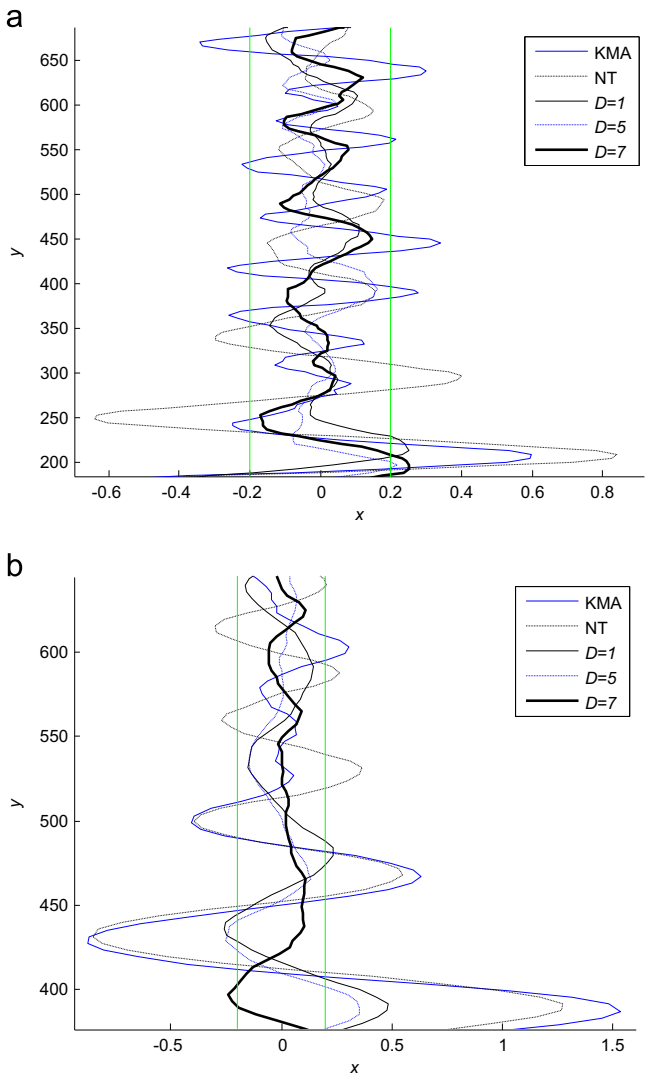

Fig. 9. Truck backing trajectories under control of KMA, NT method, and proposed method with noisy test data (D is the number of divisions) for (a)$(x_0, y_0, \phi_0) = (3, 80, 175)$ and (b)$(x_0, y_0, \phi_0) = (9, 90, 95)$.

response provides good control performance that can handle uncertainties and disturbances. To analyze the response of the proposed controller, the control surfaces for the proposed method

Table 7
Noisy test RMSE of KMA, NT, and the proposed method for 14 initial positions.

	KMA	NT	Proposed method		
			$D=1$	$D=5$	$D=7$
RMSE (position)	1.2358	1.1428	0.8642	0.6741	0.5514
RMSE (angle)	3.4621	3.1492	1.8101	1.6020	1.1512

with $D=1$, $D=5$, and $D=7$ are illustrated in Fig. 10(c)–(e). From these figures, two conclusions can be drawn. First, the partitioning strategy provides extra degrees of freedom (see Fig. 10(d) and (e)), and thus the control surface approaches that of the IT2-FLC, as shown in Fig. 10(a). Without partitioning (see Fig. 10(c)), the control surface is similar to that of a T1-FLC, as shown in Fig. 10(b). Second, the optimization strategy gives a smooth control surface (see Fig. 10(c)–(e)), which translates into a smooth control

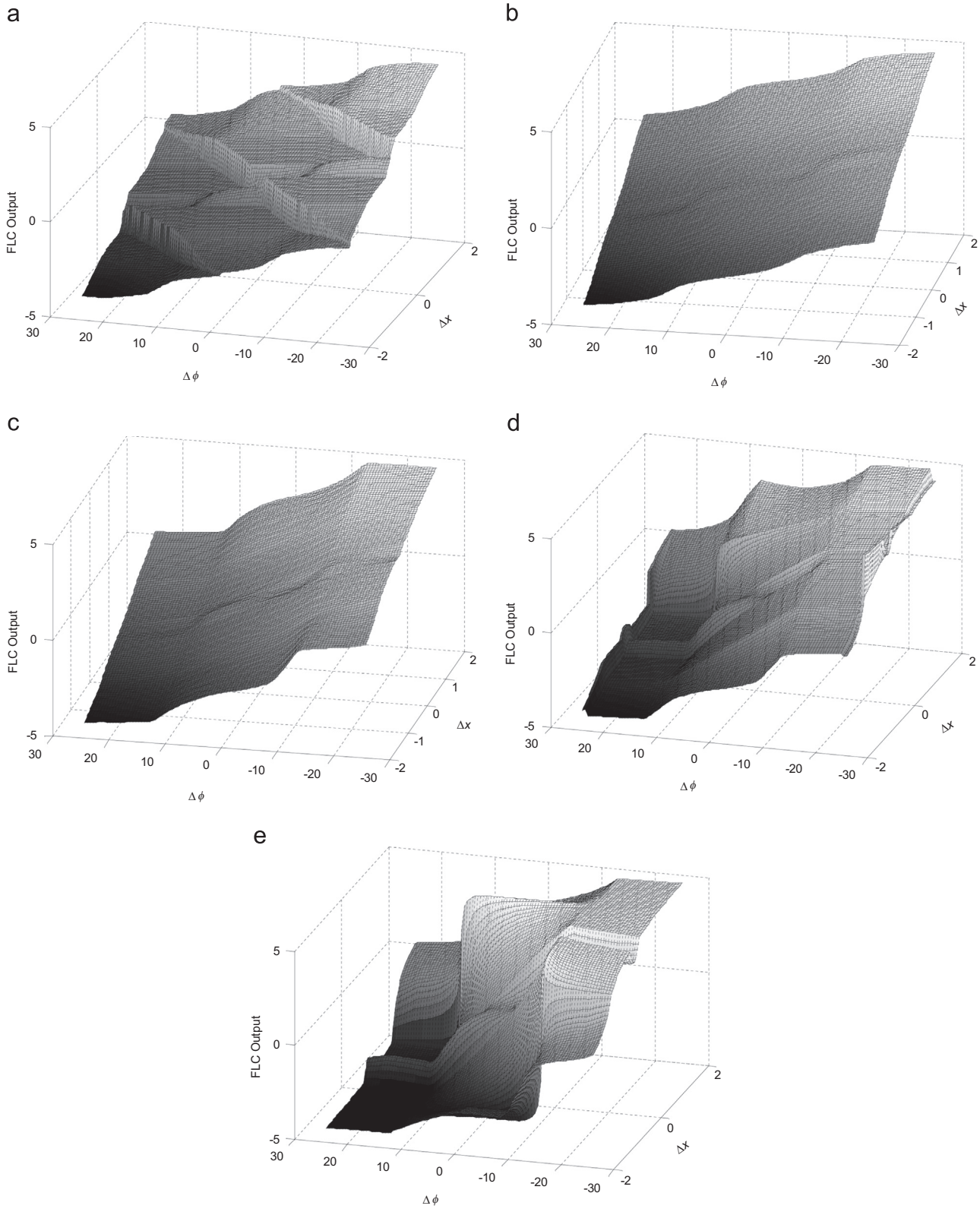

Fig. 10. FLC control surfaces for (a) IT2-FLC, (b) T1-FLC, and proposed method with (c)$D=1$, (d)$D=5$, and (e)$D=7$.

response. The non-smooth control surfaces of the IT2-FLC (see Fig. 10(a)) result in chattering output, as shown in Fig. 6(b).

5. Conclusion

In this paper, an IT2-based T1-FLC was proposed. Different from general IT2-FLCs, the proposed controller executes the type reduction offline and makes use of the equivalent type-1 membership grades in practical applications. To ensure smooth and accurate performance, partitioning and optimization strategies were proposed. Truck backing control problems were utilized to demonstrate the effectiveness of the proposed controller. The simulation results support the following conclusions:

(1) GA provides optimal defuzzified output, so the proposed controller requires less time to drive the truck to the assigned position.
(2) The partitioning strategy leads to a finer type reduction model resolution, which decreases training error and leads to higher noise resistance. As shown in the simulation results, the higher the number of divisions, the less time is needed to drive the truck to the assigned position.
(3) GA performs the type reduction offline and makes use of the equivalent type-1 membership grades in practical applications. This leads to lower computational time than general IT2-FLCs and facilitates the design of controllers that operate in real time.
(4) From the viewpoint of control surface, the partitioning strategy provides extra degrees of freedom, and thus the control surface approaches that of the IT2-FLC (without partitioning, the control surface is similar to that of a T1-FLC).
(5) The optimization strategy gives a smooth control surface, which translates into a smooth control response and eliminates the chattering phenomenon.

In the future, the proposed method will be implemented on the actual two-wheeled mobile robots. In addition, we hope to extend our work on other applications such as maglev control, disease-diagnosis system, and health monitoring system.

References

Biglarbegian, M., Melek, W.W., Mendel, J.M., 2009, A practical approach for design of PD and PI like interval type-2 TSK fuzzy controllers. In: IEEE Int. Conf. on Systems, Man and Cybernetics SMC'09, pp. 255–261.

Biglarbegian, M., Melek, W.W., Mendel, J.M., 2011. Design of novel interval type-2 fuzzy controllers for modular and reconfigurable robots: theory and experiments. IEEE Trans. Ind. Electron. 58, 1371–1384.

Bingul, Z., Karahan, O., 2011. A fuzzy logic controller tuned with PSO for 2 DOF robot trajectory control. Expert Syst. Appl. 38, 1017–1031.

Bouallegue, S., Haggege, J., Ayadi, M., Benrejeb, M., 2012. PID-type fuzzy logic controller tuning based on particle swarm optimization. Eng. Appl. Artif. Intell. 25, 484–493.

Cara, A.B., Wagner, C., Hagras, H., Pomares, H., Rojas, I., 2013. Multiobjective optimization and comparison of nonsingleton type-1 and singleton interval type-2 fuzzy logic systems. Fuzzy Sets Syst. 21, 459–476.

Cazarez-Castro, N.R., Aguilar, L.T., Castillo, O., 2012. Designing type-1 and type-2 fuzzy logic controllers via fuzzy Lyapunov synthesis for nonsmooth mechanical systems. Eng. Appl. Artif. Intell. 25, 971–979.

Coupland, S., John, R.I., 2007. Geometric type-1 and type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. 15, 3–15.

Coupland, S., Gongora, M., John, R.I., Wills, K. (2006), A comparative study of fuzzy logic controllers for autonomous robots. In: Proc. IPMU'06, pp. 1332–1339.

Du, X., Ying, H., 2010. Derivation and analysis of the analytical structures of the interval type-2 fuzzy-PI and PD controllers. IEEE Trans. Fuzzy Syst. 18, 802–814.

Gorzalczany, M., 1987. Decision making in signal transmission problems with interval-valued fuzzy sets. Fuzzy Sets Syst. 23, 191–203.

Greenfield, S., Chiclana, F., Coupland, S., John, R., 2008. The collapsing method of defuzzification for discretised interval type-2 fuzzy sets. Inf. Sci. 179, 2055–2069.

Hagras, H., 2004. A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. IEEE Trans. Fuzzy Syst. 12, 524–539.

Hosseini, R., Qanadli, S.D., Barman, S., Mazinani, M., Ellis, T., Dehmeshki, J., 2012. An automatic approach for learning and tuning Gaussian interval type-2 fuzzy membership functions applied to lung CAD classification system. IEEE Trans. Fuzzy Syst. 20, 224–234.

Hu, H., Wang, Y., Gai, Y., 2012. Advantages of the enhanced opposite direction searching algorithm for computing the centroid of an interval type-2 fuzzy set. Asian J. Control 14, 1–9.

Hu, H.Z., Zhao, G., Yang, H.N., 2010. Fast algorithm to calculate generalized centroid of interval type-2 fuzzy set. Control Decis. 25, 637–640.

Karnik, N.N., Mendel, J.M., 1999. Type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. 7, 643–658.

Karnik, N.N., Mendel, J.M., 2001. Centroid of type-2 fuzzy set. Inf. Sci. 132, 195–220.

Liang, Q., Mendel, J.M., 2000. Equalization of nonlinear time-varying channels using type-2 fuzzy adaptive filters. IEEE Trans. Fuzzy Syst. 8, 551–563.

Mendel, J., 2001. Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall, Upper Saddle River, NJ.

Mendel, J.M., Wu, D., 2010. Perceptual Computing: Aiding People in Making Subjective Judgments. Wiley-IEEE Press, Hoboken, NJ.

Mendes, J., Araujo, R., Matias, T., Seco, R., Belchior, C., 2014. Automatic extraction of the fuzzy control system by a hierarchical genetic algorithm. Eng. Appl. Artif. Intell. 29, 70–78.

Nie, M., Tan, W.W. (2008), Towards an efficient type-reduction method for interval type-2 fuzzy logic systems. In: Proc. IEEE Int. Conf. Fuzzy Syst., pp. 1425–1432.

Sepulveda, R., Castillo, O., Melin, P., Rodriguez-Diaz, A., Montiel, O., 2007. Experimental study of intelligent controllers under uncertainty using type-I and type-2 fuzzy logic. Inf. Sci. 177, 2023–2048.

Shill, P.C., Amin, M.F., Akhand, A.H., Murase, K. (2012), Optimization of interval type-2 fuzzy logic controller using quantum genetic algorithms. In: Proc. IEEE Int. Conf. Fuzzy Syst., 1–8.

Tao, C.W., Taur, J.S., Chang, C.W., Chang, Y.H., 2012. Simplified type-2 fuzzy sliding controller for wing rock system. Fuzzy Sets Syst. 207, 111–129.

Ulu, C., Guzelkaya, M., Eksin, T. (2011), A dynamic defuzzification method for interval type-2 fuzzy logic controllers. In: Proc. IEEE Int. Conf. Mechatronics, pp. 318–323.

Wang, C.H., Cheng, C.S., Lee, T.T., 2004. Dynamical optimal training for interval type-2 fuzzy neural network (T2FNN). IEEE Trans. Syst. Man Cybern. Part B Cybern. 24, 1462–1477.

Wu, D., 2013. Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: overview and comparisons. IEEE Trans. Fuzzy Syst. 21, 80–99.

Wu, D., Mendel, J.M., 2009. Enhanced Karnik–Mendel algorithms. IEEE Trans. Fuzzy Syst. 17, 923–934.

Wu, D., Tan, W.W., 2005. Computationally efficient type-reduction strategies for a type-2 fuzzy logic controller. In: Proc. IEEE Int. Conf. Fuzzy Syst., 353–358.

Wu, D., Tan, W.W., 2006. Genetic learning and performance evaluation of interval type-2 fuzzy logic controllers. Eng. Appl. Artif. Intell. 19, 829–841.

Wu, H., Mendel, J.M., 2002. Uncertainty bounds and their use in the design of interval type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. 10, 622–639.

Yeh, C.Y., Jeng, W.H., Lee, S.J., 2011. An enhanced type-reduction algorithm for type-2 fuzzy sets. IEEE Trans. Fuzzy Syst. 19, 227–240.