Applied Intelligence (2019) 49:3881–3893
https://doi.org/10.1007/s10489-019-01439-y

A recurrent TSK interval type-2 fuzzy neural networks control with online structure and parameter learning for mobile robot trajectory tracking

Aissa Bencherif[1] · Fatima Chouireb[1]

Published online: 8 May 2019

Abstract

This paper focuses on the design of a recurrent Takagi-Sugeno-Kang interval type-2 fuzzy neural network RTSKIT2FNN for mobile robot trajectory tracking problem. The RTSKIT2FNN is incorporating the recurrent frame of internal-feedback loops into interval type-2 fuzzy neural network which uses simple interval type-2 fuzzy sets in the antecedent part and the Takagi-Sugeno-Kang (TSK) type in the consequent part of the fuzzy rule. The antecedent part forms a local internal feedback loop by feeding the membership function of each node in the fuzzification layer to itself. Initially, the rule base in the RTSKIT2FNN is empty, after that, all rules are generated by online structure learning, and all the parameters of the RTSKIT2FNN are updated online using gradient descent algorithm with varied learning rates VLR. Through experimental results, we demonstrate the effectiveness of the proposed RTSKIT2FNN for mobile robot control.

Keywords Mobile robot · Trajectory tracking · Structure and parameter learning · Varied learning rates VLR · Recurrent type-2 fuzzy neural network · RTSKIT2FNN

1 Introduction

In recent years, mobile robots are extensively used in industry and for tasks in dangerous and harsh environments, such as space exploration, military, rescue in the event of natural disasters, and nuclear areas, etc. the ability to track a planned path and perform tasks in these environments autonomously is still among the main challenges facing researchers in the intelligent mobile robot's field. Significant recent studies have been focused on trajectory tracking control. Most of the proposed control strategies are using the kinematic model which produces a reference velocity based on position errors [1–5].

Recently, many studies have been done on the artificial intelligence application in mobile robot control such as fuzzy logic (FLC) and neural network (NN). The Fuzzy control (FLC) uses linguistic information based on human expertise, as a result, the FLC has several advantages such as stability, robustness, no models are required (free model), using expert knowledge and the IF-THEN rules algorithm [6, 7]. Thus, the FLC has attracted more attention in mobile robot control research. In [8], Castillo et al. designed a dynamic fuzzy logic controller for mobile robot trajectory tracking with membership functions optimization using Ant Colony algorithm. In [9], Huq et al. proposed a behaviour-modulation technique which combined the state-based formalism of discrete event system (DES) with the deterministic fuzzy decision making. In [10], Hou et al. proposed adaptive control via fuzzy approach and backstepping for mobile robot control. In [11], Bencherif and Chouireb presented a sensors data fusion based on the fuzzy logic technique for navigation of wheeled mobile robot in an unknown environment. In [7], Hagras. presented a control architecture using type-2 FLC for mobile robot navigation in dynamical unknown environments.

Previous studies achieved the main control objectives and thus have a huge impact on mobile robot control. But on the other hand, the fuzzy logic needs a human expertise knowledge information (there is no learning ability) and Fuzzy rule derivation is often difficult [12]. Recently, much research is interested in the development and applications neural network (NN) for control and system

✉ Aissa Bencherif
a.bencherif@lagh-univ.dz

Fatima Chouireb
f.chouireb@mail.lagh-univ.dz

[1] Department of Electronics, University Amar Telidji, BP 37G, Laghouat, Algeria

identification. In [13], Bencherif and Chouireb proposed a new training method for a neural network which is based on a combination between Levenberg-Marquardt optimisation method and iterated Kalman Filter for mobile robot trajectory tracking control. In [14], Yacine et al. presented the design of a kinematic controller for wheeled mobile robot trajectory tracking based on an adaptive neural network (ADALINE). In [15], Li et al. proposed an incorporating Neural-Dynamic optimisation based on predictive model control approach for mobile robot tracking control. In [16], Shojaei et al. presented mobile robots formation trajectory tracking control of type (m, s) by using neural adaptive feedback.

More recently, there has been much research that discusses incorporating the behaviours of fuzzy logic (FLSs) and neural networks (NNs). The fuzzy neural networks (FNNs) get their learning ability from NNs, and their inference technology from fuzzy systems, FNNs are an effective tool for mobile robot control. FNNs include, the adaptive network based fuzzy inference system (ANFIS) [17, 18] with a fixed structure and all parameters are tuned with a backpropagation or a hybrid learning algorithms and use for the consequent part the Takagi-Sugeno-Kang (TSK) type. The self-constructing neural fuzzy inference network (SONFIN) [19] uses the Mamdani type for the consequent part. The recurrent FNN (RFNN) [20, 21] takes the form of self-feedback connections used as internal memories and are done by returning the output of each membership function (MF) back as an input.

Moreover, all of the aforementioned FNNs and RFNN are based on type-1 fuzzy sets. In recent years, type-2 fuzzy logic systems (FLSs) have drawn much attention in many studies, the type-2 fuzzy logic introduced by Zedah in [22]. The use of type-2 fuzzy sets in nonlinear dynamic systems that are exposed to a variety of internal and external perturbations shows that type-2 methods have more potential than type-1 methods. However, the theory of type-2 FLS is more complicated than type-1 FLS theory. The interval type-2 fuzzy sets have been proposed in [23] to remove complexity and improve the type-2 fuzzy sets theory. The interval type-2 FNN has attracted a great attention in [12, 24–33], and proposed a learning process for an interval type-2 FNN system with uncertain means Gaussian membership functions. In [24], Abiyev and Kaynak described the design of IT2FNN using gradient descent learning algorithm for identification and control of time varying systems. In [25, 26], the authors proposed a self-learning IT2FNN with a fixed structure using sliding mode (SM) theory for parameters learning . In [27], Lin et al. used IT2FNN with adaptive control law for motion control. Castro et al. [28] proposed a hybrid learning algorithm for IT2FNN. In [12, 29], the authors proposed a self-evolving interval type-2 FNN (SEIT2FNN)uses interval type-2 fuzzy sets with type-2 Gaussian MFs uncertain means in the antecedent parts and Takagi-Sugeno-Kang (TSK) fuzzy sets in the consequent parts. Also, it uses the structure learning process to self-structure evolving for the IT2FNN and parameter learning to tuned the antecedent parameters using gradient descent algorithm. The consequent parameters are learned by using a rule-ordered Kalman filter. In [30], Juang et al. proposed a RSEIT2FNN for Dynamic nonlinear Systems, the antecedent part of RSEIT2FNN forms a local internal feedback connections loop by feeding the firing strength of each rule back to itself. In [31], Lin et al. applied a functional-link network (FLN) to the self-evolving IT2FNN for identification and control time-varying plants. In [32], Lin et al. designed a self-organisation T2FNN using gradient descent for parameters learning and the particle swarm optimization (PSO) method to find the optimal learning rates, applied for antilock braking systems. In [33], Lin, Y et al. proposed a simplified interval type-2 fuzzy neural networks with design factors q_r and q_l are learned to set the upper and lower values to avoid using K-M iterative method to find L and R points.

The motivation of this paper is to construct a self evolving recurrent TSK interval type 2 FNN for mobile robot trajectory tracking. A self evolving algorithm gives the ability to achieve the suitable structure of RTSKIT2FNN during the online learning, and the varied learning rates algorithm VLR is used to find the optimal learning rates of the gradient descent algorithm and to guarantee a good stability convergence of the control system which is proved using the discrete Lyapunov function. The major contributions of the proposed control are summarized as:

(1) Successful apply a recurrent frame into interval type-2 FNN structure by using internal-feedback loops in the antecedent part are formed by feeding the interval output of each membership function back to itself.
(2) A type-reduction process in the RTSKIT2FNN is performed by fixing the design factors q_l and q_r to reduce the computational time.
(3) The online structure and parameters learning algorithms are used to ameliorate the learning performances of the RTSKIT2FNN-VLR. Initially, the rule base in the RTSKIT2FNN is empty, and all rules are generated online by the structure learning algorithm. Concurrently to structure learning all the consequent and the antecedent parameters of the RTSKIT2FNN are learned by gradient descent algorithm.
(4) The design of the varied learning rates algorithm VLR to adjust the learning rates of the gradient descent algorithm to guarantee the robustness and the stability convergence of the system.
(5) In addition, experimental results were conducted to verify the proposed control RTSKIT2FNN-VLR

performances by a comprehensive comparison with type-1 FNN and RFNN and other type 2 IFNN's.

The rest of this paper is organized as follows. Section 2 introduces the Kinematics model of the mobile robot. Section 3 presents the RTSKIT2FNN structure. In Section 4, the online structure and parameter learning for RTSKIT2FNN are described in detail. Section 5 presents the experimental results, and Section 6 contains the conclusion.

2 Kinematics model

The unicycle mobile robot kinematics model is the basis of many types of nonholonomic Wheeled Mobile Robots WMRs. For this reason, it has attracted much theoretical attention of researchers.

Unicycle WMRs have usually two driving wheels, one mounted on each side of their center, and a free rolling wheel for carrying the mechanical structure. These two wheels have the same radius r and separated by 2R and are driven by two electrical actuators for the motion and orientation, and the free rolling wheel is a self-adjusted supporting wheel. It is assumed that this mobile robot is made up of a rigid frame and non-deformable wheels, and they are moving in a horizontal 2D plane with the global coordinate frame (O, X, Y). The configuration is represented by vector coordinates: $q = [x, y, \theta]$, that is, the position coordinates of the point q the centre of the WMR in the fixed coordinate frame OXY, and its orientation angle as shown in Fig. 1. The linear velocity of the wheels represented by v and the angular velocity of the mobile robot is w. The kinematics model (or equation of motion) of the mobile robot is then given by:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} v \ \cos(\theta) \\ v \ \sin(\theta) \\ w \end{pmatrix} \tag{1}$$

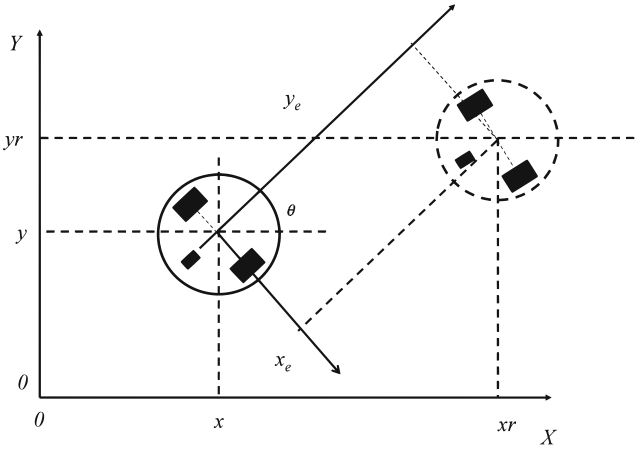


Fig. 1 Model description of a unicycle robot

To consider a trajectory tracking problem, a reference trajectory should be generated. The reference trajectory is:

$$\begin{pmatrix} \dot{x}_r \\ \dot{y}_r \\ \dot{\theta}_r \end{pmatrix} = \begin{pmatrix} v_r \ \cos(\theta_r) \\ v_r \ \sin(\theta_r) \\ w_r \end{pmatrix} \tag{2}$$

The error coordinates represented by the world coordinates are:

$$q_r - q = \begin{pmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{pmatrix} \tag{3}$$

In the view of moving coordinates, the error coordinates are transformed into:

$$\begin{pmatrix} x_e \\ y_e \\ \theta_e \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{pmatrix} \tag{4}$$

3 Recurrent TSK interval type-2 fuzzy neural networks description

This section introduces the structure of a RTSKIT2FNN. This multi-input single-output (MISO) system consists of n inputs and one output. Figure 2 shows the proposed six-layered network which realizes an interval type-2 fuzzy neural network using Takagi-Sugeno-Kang (TSK) type for the consequent part. The basic functions of each layer are described as follows.

Layer 1 **(Input layer)**: Each node in this layer represents an input variable x_j, $j = 1, ..., n$ and there are no weights to be modified in this layer.

Layer 2 **(Fuzzification Layer or membership function layer)**: each node in the membership layer performs a recurrent interval type-2 MF to execute the fuzzification operation.

The input of the membership layer can be represented by:

$$r_j^i(t) = x_j(t) + \mu_{\overline{A}_j^i}(t-1)\alpha_j^i \tag{5}$$

Where t denotes the number of iterations, α_i^j represents the weight of the self-feedback loop,$\mu_{\overline{A}_j^i}(t-1)$ represents the output signal of layer 2 in the time (t-1) and is defined with Gaussian membership functions as

$$net_i(r_j^i) = -\frac{1}{2}\left(\frac{r_j^i - m_j^i}{\sigma_j^i}\right)^2 \tag{6}$$

$$\begin{aligned} \mu_{\overline{A}_j^i} &= \exp(net_i(r_j^i)) \\ &\equiv N(m_j^i, \sigma_j^i, r_j^i), m_j^i \in \left[m_{j1}^i, m_{j2}^i\right] \end{aligned} \tag{7}$$

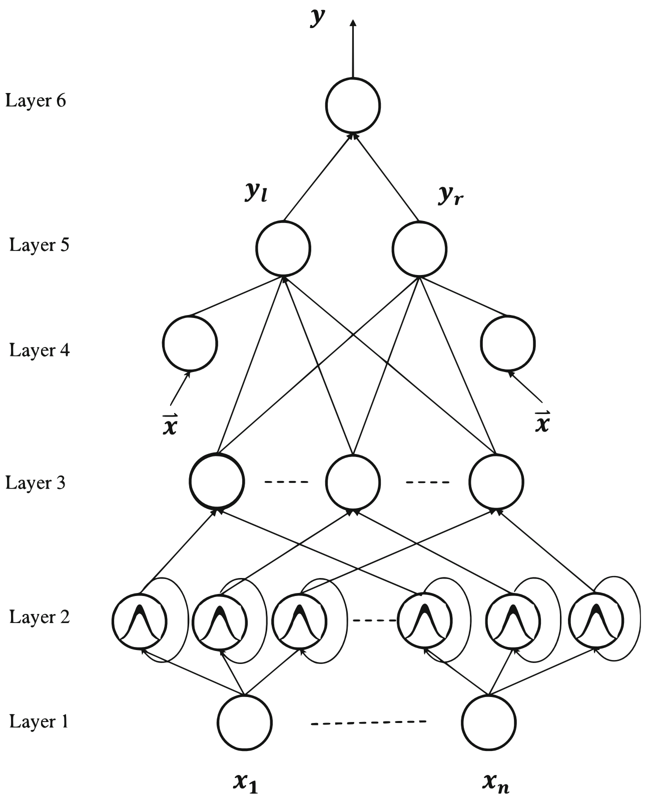


Fig. 2 Structure of the RTSKIT2FNN

Each MF of the premise part can be represented as a bounded interval in terms of upper MF, $\overline{\mu}^i_j$, and a lower MF,$\underline{\mu}^i_j$, where

$$\overline{\mu}^i_j(r^i_j) = \begin{cases} N(m^i_{j1}, \sigma^i_j, r^i_j), r^i_j < m^i_{j1} \\ 1, \quad m^i_{j1} \le r^i_i \le m^i_{j2} \\ N(m^i_{j2}, \sigma^i_j, r^i_j), r^i_j > m^i_{j2} \end{cases} \quad (8)$$

And

$$\underline{\mu}^i_j(r^i_j) = \begin{cases} N(m^i_{j1}, \sigma^i_j, r^i_j), r^i_i \le \frac{m^i_{j1}+m^i_{j2}}{2} \\ N(m^i_{j2}, \sigma^i_j, r^i_j), r^i_j > \frac{m^i_{j1}+m^i_{j2}}{2} \end{cases} \quad (9)$$

And thus, the output of this layer can be represented as an interval $[\overline{\mu}^i_j, \underline{\mu}^i_j]$ as shown in Fig. 3 .

Layer 3 (Firing Layer): In this layer, there are M nodes each node represents one fuzzy rule, and it computes the firing strength using the values that it receives from layer 2 and an algebraic product operator adopted in fuzzy meet operations, to generate the spatial firing strength F^i, and is computed as follows

$$\underline{f}^i = \prod_{j=1}^{n} \underline{\mu}^i_j, \overline{f}^i = \prod_{j=1}^{n} \overline{\mu}^i_j \text{ and } F^i = [\underline{f}^i; \overline{f}^i] \quad (10)$$

Layer 4 (Consequent Layer): In this layer, each node represents a TSK-type consequent node, for each rule node in layer 3 there is a corresponding consequent node in layer 4. The output is an interval type-1 set and can be represented as $[w^i_l, w^i_r]$ where l and r indices represent the left and right limits respectively, the node output can be expressed as

$$[w^i_l, w^i_r] = [c^i_0 - s^i_0, c^i_0 + s^i_0] + \sum_{j=1}^{n} [c^i_j - s^i_j, c^i_j + s^i_j] x_j \quad (11)$$

that is

$$w^i_l = \sum_{j=0}^{n} c^i_j x_j - \sum_{j=0}^{n} |x_j| \, s^i_j \quad (12)$$

and

$$w^i_r = \sum_{j=0}^{n} c^i_j x_j + \sum_{j=0}^{n} |x_j| \, s^i_j \quad (13)$$

where $x_0 \equiv 1$.

Layer 5 (Output Processing Layer): Each node in this layer computes an interval output $[y_l, y_r]$ by using the output from layer 4 and the firing strength F_i from layer 3, and the design factors $[q_l, q_r]$ proposed in [33] to adjust the upper and lower values without using the Karnik-Mendel iterative method. In this paper, q_l and q_r are non-adjustable values and set to 0.5. The outputs of this layer can be computed by

$$y_l = \frac{(1 - q_l) \sum_{i=1}^{M} \underline{f}^i w^i_l + q_l \sum_{i=1}^{M} \overline{f}^i w^i_l}{\sum_{i=1}^{M} \underline{f}^i + \overline{f}^i} \quad (14)$$

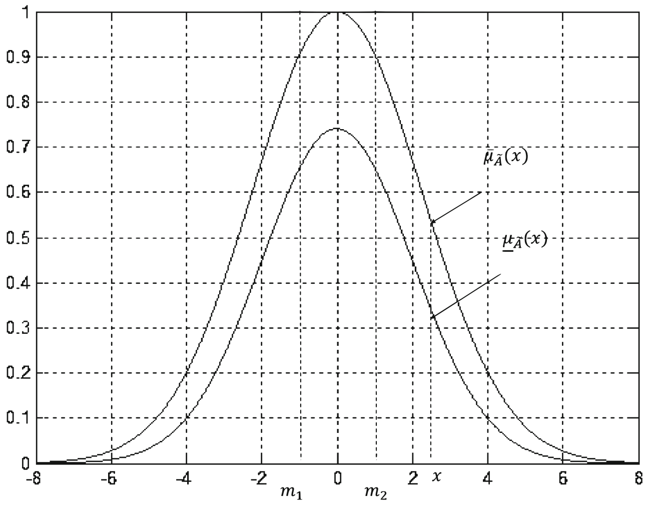


Fig. 3 Interval type-2 fuzzy set of Gaussian shape

And

$$y_r = \frac{(1-q_r)\sum_{i=1}^{M}\overline{f}^i w_r^i + q_r\sum_{i=1}^{M}\underline{f}^i w_r^i}{\sum_{i=1}^{M}\underline{f}^i + \overline{f}^i} \tag{15}$$

Layer 6 (output layer): The node in this layer computes the output variable of the RTSKIT2FNN using a sum operation of the output interval of layer 5 y_l and y_r. In this paper the output of this layer is given by

$$y = \frac{y_l + y_r}{2} \tag{16}$$

4 Online learning algorithm

This section presents the two-phase learning scheme algorithm for RTSKIT2FNN. The objective of the first phase is structure learning that discusses how to construct the RTSKIT2FNN rules. The learning starts with empty rule-base in the RTSKIT2FNN and all fuzzy rules are evolved by an online structure learning scheme. The second phase is the parameter learning phase that uses a gradient descent algorithm with varied learning rates for online parameters learning.

4.1 Structure learning

The structure learning used the rule firing strength variation as a criterion for rule generation, this idea was used for type-1 fuzzy rule generation in [19], and developed to type-2 fuzzy rule generation [12, 33] by using the interval firing strength F^i in (10), the structure learning of the proposed RTSKIT2FNN is introduced as follows.

For the first incoming data point $\overrightarrow{x}$ is used to generate the first fuzzy rule with mean, width, centre and the weight of the self-feedback loop of recurrent interval type 2 fuzzy set as

$$[m_{j1}^1, m_{j2}^1] = [x_j - \Delta x, x_j + \Delta x],$$
$$\sigma_j^1 = \sigma_{fixed} \text{ and } \alpha_j^1 = \alpha_{fixed} \tag{17}$$

where σ_{fixed} is a predefined threshold, Δx indicates the width of the uncertain mean region. If Δx is too small, then the type-2 fuzzy set approximates a type-1 fuzzy set and if Δx is too large, then a type-2 fuzzy set will cover most of the input domain. In this paper Δx and σ_{fixed} are set to 0.1 and 0.3 respectively and α_{fixed} is set to 0.1.

For each subsequent of incoming data $\overrightarrow{x}$, we compute the firing strengths interval in (10) then we compute the mean or centre value of firing strength for each rule i denoted as

$$f_c^i = \frac{\bar{f}^i + \underline{f}^i}{2} \tag{18}$$

f_c^i the spatial firing strength centre is a rule generation criterion to generate a new rule based on a predetermined threshold f_{th} . For each new incoming data,

$$I = \arg \max_{1 \le i \le M(t)} f_c^i \tag{19}$$

With *M(t)* is the number of generated rules at time t. If $f_c^I \le f_{th}$, then new fuzzy rules will be generated $M(t\text{+}1)\text{=}M(t)\text{+}1$; this condition means that the data point does not match well from any existing rules, so the new fuzzy rule evolved. The uncertain means, width and centre of type 2 fuzzy set related to the new fuzzy rule are defined in the same manner as the first rule

$$[m_{j1}^{M(t)+1}, m_{j2}^{M(t)+1}] = [x_j - \Delta x, x_j + \Delta x]\,, \alpha_j^{M(t)+1} = 0.1 \tag{20}$$

and

$$\sigma_j^{M(t)+1} = \beta \left| x_j - \frac{m_{j1}^I + m_{j2}^I}{2} \right| \tag{21}$$

where $\beta > 0$ determines the overlap degree between fuzzy sets. Equation (21) denotes that the width is equal to Euclidean distance between the average centres of the new fuzzy set and fuzzy set *I*. The high overlapping between fuzzy sets is obtained when β is too large, and there is no overlapping if β is too small. In this paper is set to 0.5; this indicates that the width is equal to the half distance between fuzzy sets

The initial consequent parameters c_j^1 with $j = 1, .., n$ in the first rule are set to very small values 0.05, and s_j^1 are defined as the initial output interval range are set to very small values 0.002. For the new generated rules the consequent parameters are assigned to the same values as the first rules

$$c_j^1 = c_j^{M(t+1)} = 0.05 \quad j = 1, \ldots n \tag{22}$$

$$s_j^1 = s_j^{M(t+1)} = 0.002 \tag{23}$$

The structure learning and fuzzy rules generation algorithm as follows:

```
    IF x is the first incoming data then do
    {Assigned the initial uncertain means and width and center of fuzzy set and generate the first fuzzy rule by (17)
    $[m_{j1}^{1}, m_{j2}^{1}] = [x_j - \Delta x, x_j + \Delta x]$ and $\sigma_j^i = \sigma_{fixed}$ $\alpha_j^i = 0.1$ $j = 1, ...., n$
    And sets the initial consequent parameters
    $c_j^1 = 0.05$        $s_j^1 = 0.002$        $j = 1, ...,n$

}
Else
    {Compute (19)
    If $f_c^I(t) \le f_{th}$
    {Generate the new fuzzy rule and assigned the uncertain means and width and center of the new fuzzy set rule (20),(21) M(t+1)=M(t)+1
    $[m_{j1}^{M(t)+1}, m_{j2}^{M(t)+1}] = [x_j - \Delta x, x_j + \Delta x]$     $\alpha_j^{M(t)+1} = 0.1$
    $\sigma_j^{M(t)+1} = \beta \left| x_j - \frac{m_{j1}^I + m_{j2}^I}{2} \right|$
    And also set consequent parameters of fuzzy rules
    $c_j^{M(t)+1} = 0.05$        $s_j^{M(t)+1} = 0.002$        $j = 1, ...,n$

}}
```

Parameters of each evolved rules are modified by the parameter-learning algorithm introduced in the next section.

4.2 Parameter learning

Parameter learning is performed concurrently with the structure learning process. Parameter Learning tunes all parameters for each fuzzy rule, including the current fuzzy rule and that was newly generated. To describe the parameter learning, we define the objective function as

$$E = \frac{1}{2}\left(y_d\left(t+1\right) - y\left(t+1\right)\right)^2 \tag{24}$$

where $y_d(t+1)$ is the desired output and $y(t+1)$ is the output of RTSKIT2FNN. The updated parameters based on gradient descent algorithm are given by the following equations:

$$m_{j1}^i\left(t+1\right) = m_{j1}^i\left(t\right) - \eta_{m_1}\frac{\partial E}{\partial m_{j1}^i} \tag{25}$$

$$m_{j2}^i\left(t+1\right) = m_{j2}^i\left(t\right) - \eta_{m_2}\frac{\partial E}{\partial m_{j2}^i} \tag{26}$$

$$\sigma_j^i\left(t+1\right) = \sigma_j^i\left(t\right) - \eta_\sigma\frac{\partial E}{\partial \sigma_j^i} \tag{27}$$

$$\alpha_j^i\left(t+1\right) = \alpha_j^i\left(t\right) - \eta_\alpha\frac{\partial E}{\partial \alpha_j^i} \tag{28}$$

$$c_j^i\left(t+1\right) = c_j^i\left(t\right) - \eta_c\frac{\partial E}{\partial c_j^i} \tag{29}$$

$$s_j^i\left(t+1\right) = s_j^i\left(t\right) - \eta_s\frac{\partial E}{\partial s_j^i} \tag{30}$$

where $\eta_{m1}, \eta_{m2}, \eta_\sigma, \eta_\alpha, \eta_c, \eta_s$ are the learning rates ,$j = 1, ., n$ and $i = 1, , M$, n and M are the number of inputs and rules respectively . The derivation of the premise part in (25)–(28) can be expressed by the following:

$$\frac{\partial E}{\partial m_{j1}^i} = (y - y_d) \times \left(\left(\frac{\partial y_l}{\partial \bar{f}^i} + \frac{\partial y_r}{\partial \bar{f}^i}\right)\frac{\partial \bar{f}^i}{\partial m_{j1}^i} + \left(\frac{\partial y_l}{\partial \underline{f}^i} + \frac{\partial y_r}{\partial \underline{f}^i}\right)\frac{\partial \underline{f}^i}{\partial m_{j1}^i}\right) \tag{31}$$

$$\frac{\partial E}{\partial m_{j2}^i} = (y - y_d) \times \left(\left(\frac{\partial y_l}{\partial \bar{f}^i} + \frac{\partial y_r}{\partial \bar{f}^i}\right)\frac{\partial \bar{f}^i}{\partial m_{j2}^i} + \left(\frac{\partial y_l}{\partial \underline{f}^i} + \frac{\partial y_r}{\partial \underline{f}^i}\right)\frac{\partial \underline{f}^i}{\partial m_{j2}^i}\right) \tag{32}$$

$$\frac{\partial E}{\partial \sigma_j^i} = (y - y_d) \times \left(\left(\frac{\partial y_l}{\partial \bar{f}^i} + \frac{\partial y_r}{\partial \bar{f}^i}\right)\frac{\partial \bar{f}^i}{\partial \sigma_j^i} + \left(\frac{\partial y_l}{\partial \underline{f}^i} + \frac{\partial y_r}{\partial \underline{f}^i}\right)\frac{\partial \underline{f}^i}{\partial \sigma_j^i}\right) \tag{33}$$

$$\frac{\partial E}{\partial \alpha_j^i} = (y - y_d) \times \left(\left(\frac{\partial y_l}{\partial \bar{f}^i} + \frac{\partial y_r}{\partial \bar{f}^i}\right)\frac{\partial \bar{f}^i}{\partial \alpha_j^i} + \left(\frac{\partial y_l}{\partial \underline{f}^i} + \frac{\partial y_r}{\partial \underline{f}^i}\right)\frac{\partial \underline{f}^i}{\partial \alpha_j^i}\right) \tag{34}$$

The derivation of $\frac{\partial y}{\partial f}$ can be expressed by the following equations :

$$\frac{\partial y_l}{\partial \bar{f}^i} = \frac{(1 - q_l)w_l - y_l}{\sum \bar{f}^i + \underline{f}^i} \tag{35}$$

$$\frac{\partial y_l}{\partial \underline{f}^i} = \frac{q_l w_l - y_l}{\sum \bar{f}^i + \underline{f}^i} \tag{36}$$

$$\frac{\partial y_r}{\partial \bar{f}^i} = \frac{q_r w_r - y_r}{\sum \bar{f}^i + \underline{f}^i} \tag{37}$$

$$\frac{\partial y_r}{\partial \underline{f}^i} = \frac{(1-q_r) w_r - y_r}{\sum \bar{f}^i + \underline{f}^i} \tag{38}$$

Now, we calculate the derivatives of firing functions with respect to the premise parameters

$$\frac{\partial \bar{f}^i}{\partial m^i_{j1}} = \frac{\partial \bar{f}^i}{\partial \overline{\mu}^i_j}\frac{\partial \overline{\mu}^i_j}{\partial m^i_{j1}} = \begin{cases} \bar{f}^i \times \frac{r^i_j - m^i_{j1}}{(\sigma^i_j)^2}, r^i_j < m^i_{j1} \\ 0 \qquad otherwise \end{cases} \tag{39}$$

$$\frac{\partial \underline{f}^i}{\partial m^i_{j1}} = \frac{\partial \underline{f}^i}{\partial \underline{\mu}^i_j}\frac{\partial \underline{\mu}^i_j}{\partial m^i_{j1}} = \begin{cases} \underline{f}^i \times \frac{r^i_j - m^i_{j1}}{(\sigma^i_j)^2}, r^i_j > \frac{m^i_{j1}+m^i_{j2}}{2} \\ 0 \qquad otherwise \end{cases} \tag{40}$$

$$\frac{\partial \bar{f}^i}{\partial m^i_{j2}} = \frac{\partial \bar{f}^i}{\partial \overline{\mu}^i_j}\frac{\partial \overline{\mu}^i_j}{\partial m^i_{j2}} = \begin{cases} \bar{f}^i \times \frac{r^i_j - m^i_{j2}}{(\sigma^i_j)^2}, r^i_j > m^i_{j2} \\ 0 \qquad otherwise \end{cases} \tag{41}$$

$$\frac{\partial \underline{f}^i}{\partial m^i_{j2}} = \frac{\partial \underline{f}^i}{\partial \underline{\mu}^i_j}\frac{\partial \underline{\mu}^i_j}{\partial m^i_{j2}} = \begin{cases} \underline{f}^i \times \frac{r^i_j - m^i_{j2}}{(\sigma^i_j)^2}, r^i_j \le \frac{m^i_{j1}+m^i_{j2}}{2} \\ 0 \qquad otherwise \end{cases} \tag{42}$$

$$\frac{\partial \bar{f}^i}{\partial \sigma^i_j} = \frac{\partial \bar{f}^i}{\partial \overline{\mu}^i_j}\frac{\partial \overline{\mu}^i_j}{\partial \sigma^i_j} = \begin{cases} \bar{f}^i \times \frac{(r^i_j - m^i_{j1})^2}{(\sigma^i_j)^3}, r^i_j < m^i_{j1} \\ \bar{f}^i \times \frac{(r^i_j - m^i_{j2})^2}{(\sigma^i_j)^3}, r^i_j < m^i_{j2} \\ 0 \qquad otherwise \end{cases} \tag{43}$$

$$\frac{\partial \underline{f}^i}{\partial \sigma^i_j} = \frac{\partial \underline{f}^i}{\partial \underline{\mu}^i_j}\frac{\partial \underline{\mu}^i_j}{\partial \sigma^i_j} = \begin{cases} \underline{f}^i \times \frac{(r^i_j - m^i_{j2})^2}{(\sigma^i_j)^3}, r^i_j \le \frac{m^i_{j1}+m^i_{j2}}{2} \\ \underline{f}^i \times \frac{(r^i_j - m^i_{j1})^2}{(\sigma^i_j)^3}, r^i_j > \frac{m^i_{j1}+m^i_{j2}}{2} \end{cases} \tag{44}$$

$$\begin{aligned} \frac{\partial \bar{f}^i}{\partial \alpha^i_j} &= \frac{\partial \bar{f}^i}{\partial \overline{\mu}^i_j}\frac{\partial \overline{\mu}^i_j}{\partial \alpha^i_j} = \frac{\partial \bar{f}^i}{\partial \overline{\mu}^i_j}\left(\frac{\partial \overline{\mu}^i_j}{\partial r^i_j}\frac{\partial r^i_j}{\partial \alpha^i_j}\right) \\ &= \begin{cases} \bar{f}^i \times \frac{r^i_j - m^i_{j1}}{(\sigma^i_j)^2} \times \overline{\mu}^i_j(t-1), r^i_j < m^i_{j1} \\ \bar{f}^i \times \frac{r^i_j - m^i_{j2}}{(\sigma^i_j)^2} \times \overline{\mu}^i_j(t-1), r^i_j < m^i_{j2} \\ 0 \qquad otherwise \end{cases} \end{aligned} \tag{45}$$

$$\begin{aligned} \frac{\partial \underline{f}^i}{\partial \alpha^i_j} &= \frac{\partial \underline{f}^i}{\partial \underline{\mu}^i_j}\frac{\partial \underline{\mu}^i_j}{\partial \alpha^i_j} = \frac{\partial \underline{f}^i}{\partial \underline{\mu}^i_j}\left(\frac{\partial \underline{\mu}^i_j}{\partial r^i_j}\frac{\partial r^i_j}{\partial \alpha^i_j}\right) \\ &= \begin{cases} \underline{f}^i \times \frac{r^i_j - m^i_{j2}}{(\sigma^i_j)^2} \times \underline{\mu}^i_j(t-1), r^i_j \le \frac{m^i_{j1}+m^i_{j2}}{2} \\ \underline{f}^i \times \frac{r^i_j - m^i_{j1}}{(\sigma^i_j)^2} \times \underline{\mu}^i_j(t-1), r^i_j > \frac{m^i_{j1}+m^i_{j2}}{2} \end{cases} \end{aligned} \tag{46}$$

Then the derivation of consequent parameter in (29–30) can be expressed by the following:

$$\begin{aligned} \frac{\partial E}{\partial c^i_j} &= (y - y_d)\times\left(\left(\frac{\partial y}{\partial y_l}\frac{\partial y_l}{\partial w^i_l}\frac{\partial w^i_l}{\partial c^i_j}\right) + \left(\frac{\partial y}{\partial y_r}\frac{\partial y_r}{\partial w^i_r}\frac{\partial w^i_r}{\partial c^i_j}\right)\right) \\ &= (y - y_d)\times\left(\frac{(1-q_l)\underline{f}^i + q_l\overline{f}^i}{2\sum_{i=1}^M \underline{f}^i + \overline{f}^i}x_j \right. \\ &\qquad \left. + \frac{(1-q_r)\overline{f}^i + q_r\underline{f}^i}{2\sum_{i=1}^M \underline{f}^i + \overline{f}^i}x_j\right) \\ &= (y - y_d)\times\left(\frac{x_j}{2\sum_{i=1}^M \underline{f}^i + \overline{f}^i}\right)(1-q_l + q_r)\underline{f}^i \\ &\quad +(1-q_r + q_l)\overline{f}^i \end{aligned} \tag{47}$$

$$\begin{aligned} \frac{\partial E}{\partial s^i_j} &= (y - y_d)\times\left(\left(\frac{\partial y}{\partial y_l}\frac{\partial y_l}{\partial w^i_l}\frac{\partial w^i_l}{\partial s^i_j}\right) + \left(\frac{\partial y}{\partial y_r}\frac{\partial y_r}{\partial w^i_r}\frac{\partial w^i_r}{\partial s^i_j}\right)\right) \\ &= (y - y_d)\times\left(\frac{(1-q_r)\overline{f}^i + q_r\underline{f}^i}{2\sum_{i=1}^M \underline{f}^i + \overline{f}^i}|x_j| \right. \\ &\qquad \left. - \frac{(1-q_l)\underline{f}^i + q_l\overline{f}^i}{2\sum_{i=1}^M \underline{f}^i + \overline{f}^i}|x_j|\right) \\ &= (y - y_d)\times\left(\frac{|x_j|}{2\sum_{i=1}^M \underline{f}^i + \overline{f}^i}\right)(1-q_r - q_l)\overline{f}^i \\ &\quad -(1-q_l - q_r)\underline{f}^i \end{aligned} \tag{48}$$

The structure of the global online learning process is presented in the flowchart Fig. 4.

4.3 Convergence analysis

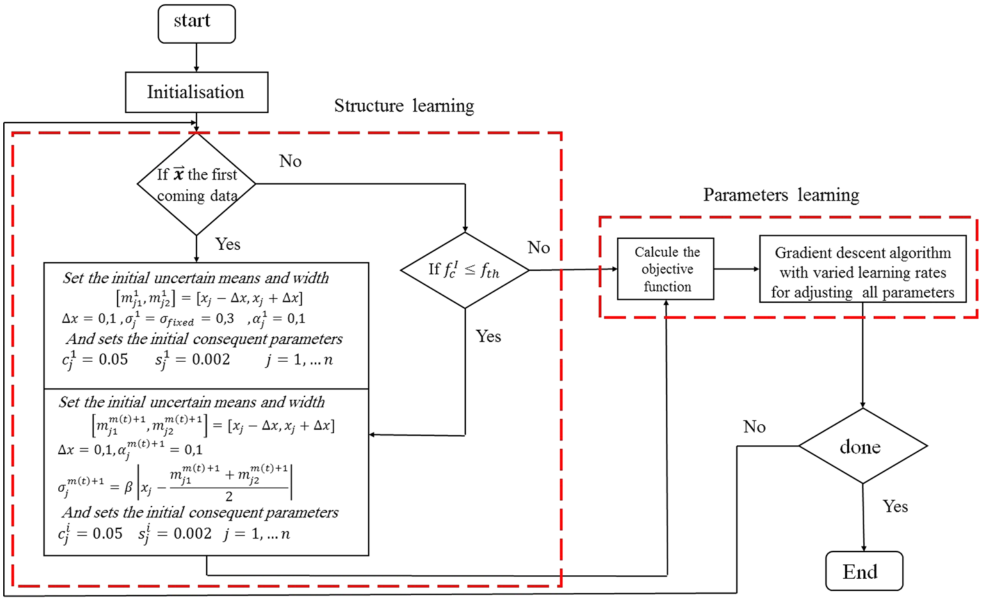
In this section, we propose the varied learning rates of parameter learning to assure the convergence of the tracking errors based on the analysis of a discrete-type Lyapunov function. Consider the objective function in (24) as the discrete Lyapunov function.

The derivative of the proposed discrete Lyapunov function can be written as

$$\Delta E(t) = E(t+1) - E(t) \tag{49}$$

The linearized model of (49) can be calculated via (25)–(30) as follow

Fig. 4 Flowchart of the online learning of the RTSKIT2FNN


start
Initialisation
Structure learning
If $\vec{\boldsymbol{x}}$ the first coming data
No
Yes
Set the initial uncertain means and width
$[m_{j1}^{1}, m_{j2}^{1}] = [x_j - \Delta x, x_j + \Delta x]$
$\Delta x = 0{,}1$, $\sigma_j^{1} = \sigma_{fixed} = 0{,}3$, $\alpha_j^{1} = 0{,}1$
And sets the initial consequent parameters
$c_j^{1} = 0.05$ $s_j^{1} = 0.002$ $j = 1, \dots n$
Set the initial uncertain means and width
$\left[m_{j1}^{m(t)+1}, m_{j2}^{m(t)+1}\right] = [x_j - \Delta x, x_j + \Delta x]$
$\Delta x = 0{,}1, \alpha_j^{m(t)+1} = 0{,}1$
$\sigma_j^{m(t)+1} = \beta \left| x_j - \frac{m_{j1}^{m(t)+1} + m_{j2}^{m(t)+1}}{2} \right|$
And sets the initial consequent parameters
$c_j^{i} = 0.05$ $s_j^{i} = 0.002$ $j = 1, \dots n$
If $f_c^{I} \leq f_{th}$
No
Yes
Parameters learning
Calcule the objective function
Gradient descent algorithm with varied learning rates for adjusting all parameters
done
No
Yes
End

$$
\begin{aligned}
E\left(t+1\right) &= E\left(t\right) + \Delta E\left(t\right) \\
&\approx E\left(t\right) - \sum_{j=1}^{n} \eta_{m_1} \left(\frac{\partial E}{\partial m_{j1}^{i}}\right)^2 + \eta_{m_2} \left(\frac{\partial E}{\partial m_{j2}^{i}}\right)^2 \\
&\quad + \eta_{\sigma} \left(\frac{\partial E}{\partial \sigma_j^{i}}\right)^2 + \eta_{c} \left(\frac{\partial E}{\partial c_j^{i}}\right)^2 + \eta_{s} \left(\frac{\partial E}{\partial s_j^{i}}\right)^2 \\
&\quad + \eta_{\alpha} \left(\frac{\partial E}{\partial \alpha_j^{i}}\right)^2
\end{aligned}
\tag{50}
$$

Then $E(t+1)$ can be expressed as

$$
\begin{aligned}
E\left(t+1\right) = & \left(\frac{1}{6} E\left(t\right) - \eta_{m_1} \sum_{j=1}^{n} \left(\frac{\partial E}{\partial m_{j1}^{i}}\right)^2\right) \\
& + \left(\frac{1}{6} E\left(t\right) - \eta_{m_2} \sum_{j=1}^{n} \left(\frac{\partial E}{\partial m_{j2}^{i}}\right)^2\right) \\
& + \left(\frac{1}{6} E\left(t\right) - \eta_{\sigma} \sum_{j=1}^{n} \left(\frac{\partial E}{\partial \sigma_j^{i}}\right)^2\right) \\
& + \left(\frac{1}{6} E\left(t\right) - \eta_{c} \sum_{j=1}^{n} \left(\frac{\partial E}{\partial c_j^{i}}\right)^2\right) \\
& + \left(\frac{1}{6} E\left(t\right) - \eta_{s} \sum_{j=1}^{n} \left(\frac{\partial E}{\partial s_j^{i}}\right)^2\right) \\
& + \left(\frac{1}{6} E\left(t\right) - \eta_{\alpha} \sum_{j=1}^{n} \left(\frac{\partial E}{\partial \alpha_j^{i}}\right)^2\right)
\end{aligned}
\tag{51}
$$

where η_{m1}, η_{m2}, η_{σ}, η_{α}, η_{c}, η_{s} are the learning rates parameters. The derivatives of the objective function in (51) are calculated in parameter learning Section 4.2. The learning-rate parameters of the RTSKIT2FNN are designed as

$$
\eta_{m_1} = \frac{E\left(t\right)}{6\left[\sum_{j=1}^{n} \left(\frac{\partial E}{\partial m_{j1}^{i}}\right)^2 + \varepsilon\right]}
\tag{52}
$$

$$
\eta_{m_2} = \frac{E(t)}{6\left[\sum_{j=1}^{n} \left(\frac{\partial E}{\partial m_{j2}^{i}}\right)^2 + \varepsilon\right]}
\tag{53}
$$

$$
\eta_{\sigma} = \frac{E\left(t\right)}{6\left[\sum_{j=1}^{n} \left(\frac{\partial E}{\partial \sigma_j^{i}}\right)^2 + \varepsilon\right]}
\tag{54}
$$

$$
\eta_{\alpha} = \frac{E\left(t\right)}{6\left[\sum_{j=1}^{n} \left(\frac{\partial E}{\partial \alpha_j^{i}}\right)^2 + \varepsilon\right]}
\tag{55}
$$

$$
\eta_{c} = \frac{E(t)}{6\left[\sum_{j=1}^{n} \left(\frac{\partial E}{\partial c_j^{i}}\right)^2 + \varepsilon\right]}
\tag{56}
$$

$$
\eta_{s} = \frac{E(t)}{6\left[\sum_{j=1}^{n} \left(\frac{\partial E}{\partial s_j^{i}}\right)^2 + \varepsilon\right]}
\tag{57}
$$

where ε is positive constant, then (51) can be rewritten as

$$
E\left(t+1\right) \approx \varepsilon \left(\eta_{m_1} + \eta_{m_2} + \eta_{\sigma} + \eta_{c} + \eta_{s} + \eta_{\alpha}\right)
\tag{58}
$$

Applying the learning rates values from (52–57) into the (58) we have

$$
E(t{+}1) < \frac{E(t)}{6} + \frac{E(t)}{6} + \frac{E(t)}{6} + \frac{E(t)}{6} + \frac{E(t)}{6} + \frac{E(t)}{6} = E(t)
\tag{59}
$$

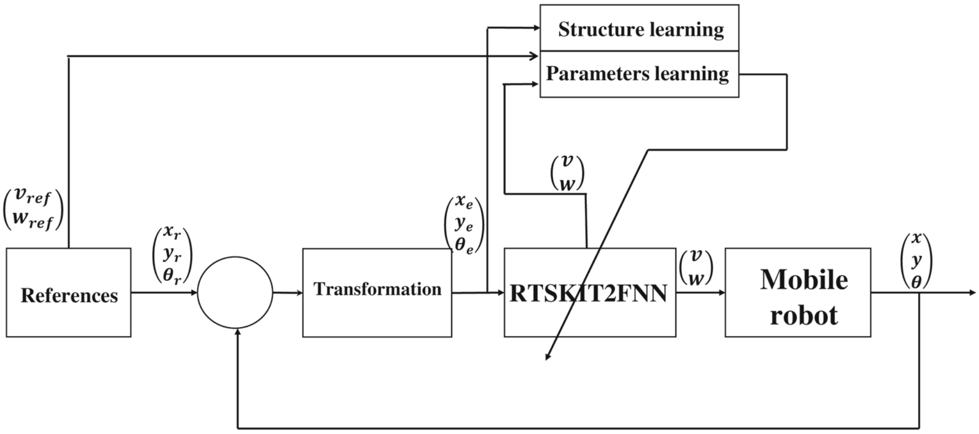

Structure learning
Parameters learning
$\begin{pmatrix} v \\ w \end{pmatrix}$
$\begin{pmatrix} v_{ref} \\ w_{ref} \end{pmatrix}$
$\begin{pmatrix} x_r \\ y_r \\ \theta_r \end{pmatrix}$
$\begin{pmatrix} x_e \\ y_e \\ \theta_e \end{pmatrix}$
References
Transformation
RTSKIT2FNN
$\begin{pmatrix} v \\ w \end{pmatrix}$
Mobile robot
$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix}$

Fig. 5 RTSKIT2FNN control scheme for a unicycle wheeled mobile robot

From (59), we can guarantee that the error represented in (24) will converge to zero with varied learning rates designed in (52)–(57). Moreover, the mobile robot with the proposed RTSKIT2FNN control system has the ability to track the reference path stably.

5 Experimental results

5.1 Robot description and control architecture

Herein, we present experimental results to validate the control performance of the proposed RTSKIT2FNN with online structure and parameter learning for stable path tracking of mobile robots. In this paper, a Pioneer 3-DX mobile robot is used as an example to verify the effectiveness of the proposed trajectory tracking control scheme shown in Fig. 5.

Pioneer 3-DX mobile robot is shown in Fig. 6. The platform is assembled with motors featuring 500-tick encoders, 19 cm wheels, tough aluminum body, 8 forward-facing ultrasonic (sonar) sensors, 1, 2 or 3 hot-swappable batteries, and a complete software development kit. The base Pioneer 3-DX platform can reach speeds of 1.6 meters per second. The Pioneer 3-DX robot is an all-purpose base, used for research and applications involving mapping, teleportation, localization, monitoring, reconnaissance, and other behaviors.

5.2 Results

For RTSKIT2FNN's, the used parameters are chosen as follows. The RTSKIT2FNN for linear velocity has two inputs $(x_e, \dot{x}_e)$ and one output v, the spatial firing strength threshold is set to $f_{th} = 0.6$. The RTSKIT2FNN for angular

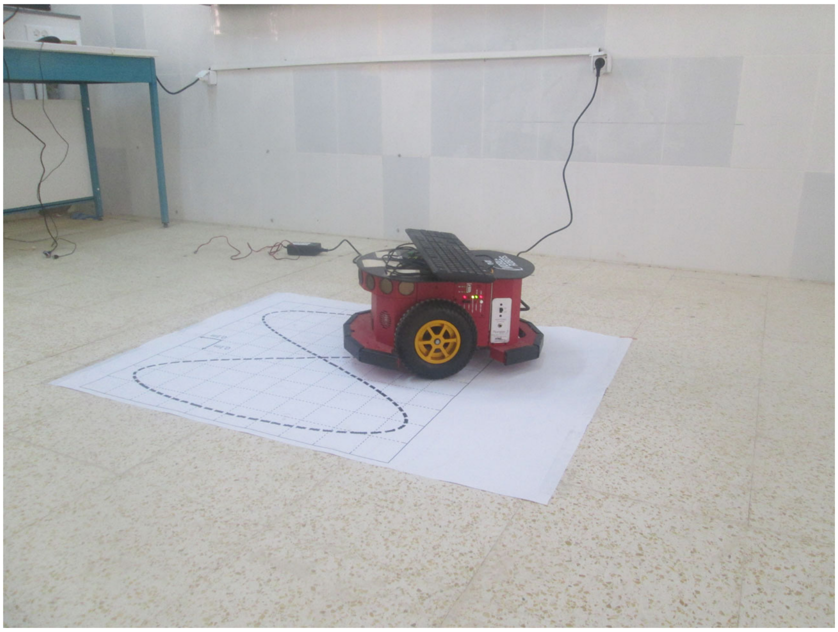


Fig. 6 Experimental test with Pioneer 3-DX mobile robot

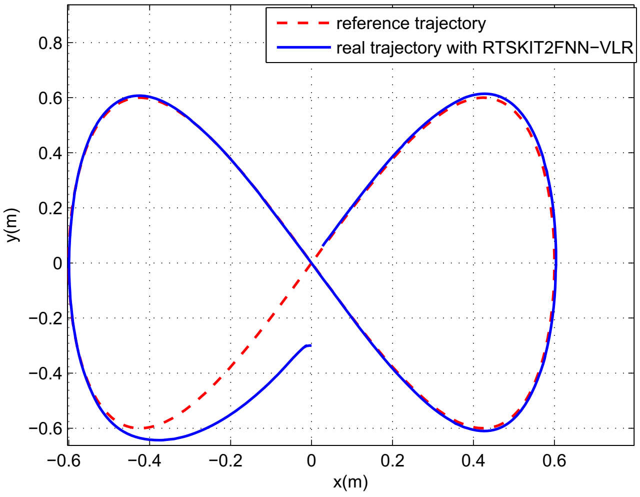


Fig. 7 Tracking response of a mobile robot with RTSKIT2FNN controllers for Lemniscate trajectory with initial position $X_0 = [0; -0.3; \pi]$

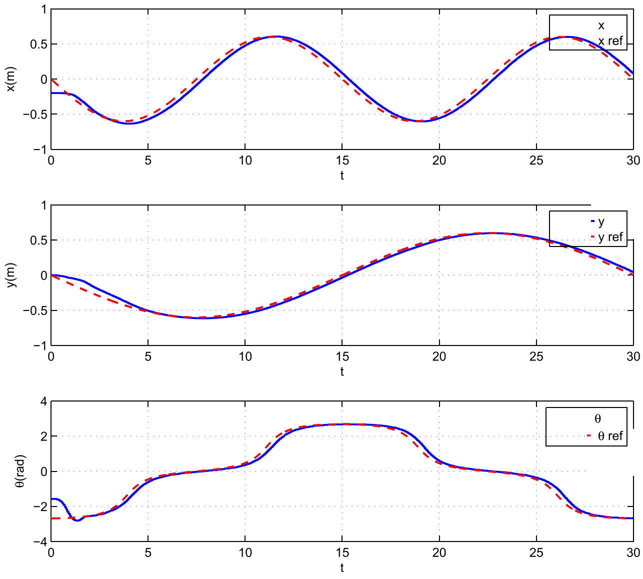


Fig. 9 The position coordinates for lemniscate trajectory

velocity has two inputs (y_e, θ_e) and one output w, the spatial firing strength threshold is $f_{th} = 0.3$. For the fixed learning rates FLR are set to $\eta_v = 0.5$ for linear velocity and $\eta_w = 0.01$ for the angular velocity. For the varied learning rates VLR the positive constant ε is set to 0.01 and 0.00001 for the linear and angular velocities respectively. The objective functions for parameters learning of the two RTSKIT2FNN's are chosen as follows

$$E_v = \frac{1}{2}\left(v_r\left(t+1\right) - v\left(t+1\right)\right)^2 \tag{60}$$

$$E_w = \frac{1}{2}\left(w_r\left(t+1\right) - w\left(t+1\right)\right)^2 \tag{61}$$

The sampling time (ts) is 0.01s. In order to exhibit the superiority of the proposed RTSKIT2FNN control system, seven different network structures including ANFIS [17], RFNN [34], IT2FNN [24], IT2FNN-SM [26], SOT2FNN-PSO [32], SEFT2FNN [31] and SIT2FNN in [33] are considered. The SIT2FNN, SEFT2FNN and SOT2FNN-PSO are also examined with the same structure and learning parameters in RTSKIT2FNN with FLR. As a performance criterion, we evaluate the root mean square error (RMSE)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(X\left(i\right) - X_r\left(i\right)\right)^2} \tag{62}$$

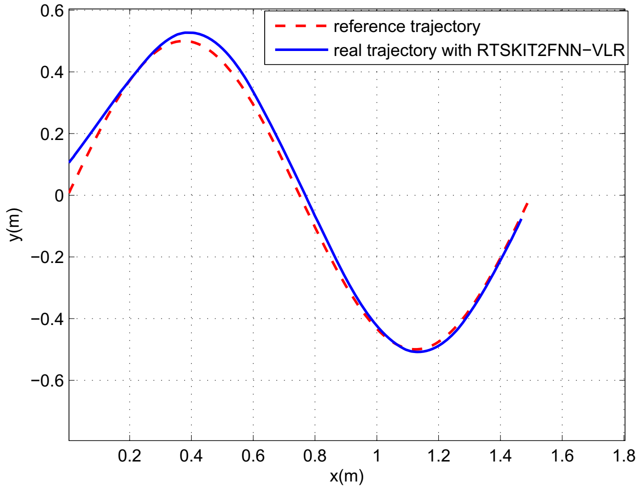


Fig. 8 Tracking response of a mobile robot with RTSKIT2FNN controllers for Sinusoidal trajectory with initial position $X_0 = [0; 0.1; \pi/4]$

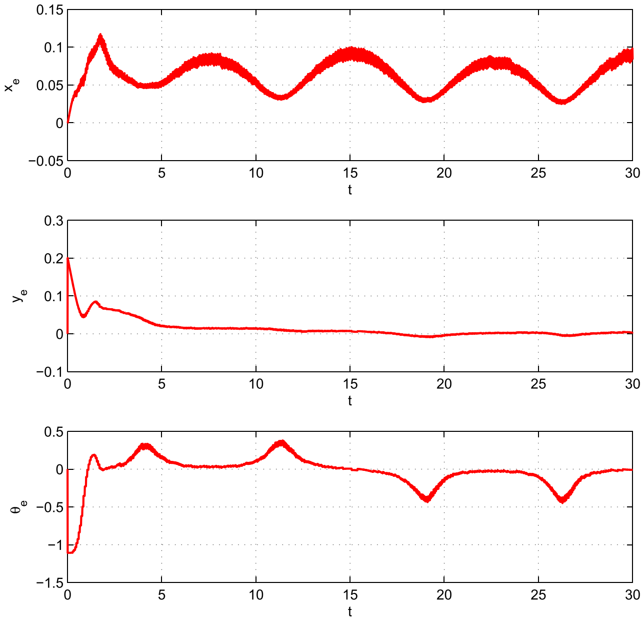


Fig. 10 Tracking errors

Table 1 Comparison results between FLR and VLR

		RMSE x	RMSE y	RMSE θ
Fixed learning	Fig. 7	0.0638	0.062386	0.24733
Rates (FLR)	Fig. 8	0.067332	0.049423	0.10764
Varied learning	Fig. 7	0.040172	0.030544	0.15698
Rates (VLR)	Fig. 8	0.04199	0.043071	0.10414

Experimental results are shown in Figs. 7, 8, 9, 10, and Tables 1, 2, 3 and 4. Figures 7 and 8 show the reference paths Lemniscate and sinusoidal respectively with the real tracked path of the mobile robot using RTSKIT2FNN with varied learning rates (VLR), the dashed red line is the reference trajectory; the solid blue line is the trajectory followed by the mobile robot with RTSKIT2FNN-VLR. Figure 9 presents the position coordinates x, y and θ of the mobile robot with reference coordinates for the Lemniscate trajectory, the tracking errors x_e, y_e and θ_e are shown in the Fig. 10. From these figures, we can see that, the effectiveness of the proposed control scheme using RTSKIT2FNN -VLR for mobile robot trajectory tracking with stable speed and high accuracy.

Table 1 studies the RTSKIT2FNN performances with varied and fixed learning rates for two cases of trajectories. The performances are evaluated using the RMSE of the position (x, y) and the orientation θ. The obtained results show that the RTSKIT2FNN with VLR has better performances than RTSKIT2FNN with FLR. The table shows the difficulty to ensure the good convergence and tracking stability during the online learning by using fixed learning rates (FLR) because the choice of fixed learning rate values is not necessarily appropriate to all of the cases and conditions of tracking control in real time and it may cause in some cases the divergence of the system. On the other hand, the varied learning rates VLR guaranteed the stability, robustness and the good parameters adaptation of the RTSKIT2FNN during the online learning, this VLR property changes the learning rates according to the objective function represented in (60) and (61) to guarantee the convergence of tracking errors.

Table 2 Influence of f_{thv} on the performance of RTSKIT2FNN with $f_{thw} = 0.3$

f_{thv}		0.4	0.5	0.6	0.7
Rule	v	1	7	7	8
Number	w	14	16	14	14
	RMSE x	0.059431	0.059272	0.040172	0.059509
RMSE	RMSE y	0.038592	0.037613	0.030544	0.0390385
	RMSE θ	0.22255	0.23153	0.15698	0.22031

Table 3 Influence of f_{thw} on the performance of RTSKIT2FNN with $f_{thv} = 0.6$

f_{thw}		0.1	0.2	0.3	0.4
Rule	v	7	9	7	7
Number	w	4	8	14	19
	RMSE x	0.059337	0.059193	0.040172	0.059762
RMSE	RMSE y	0.03968	0.040383	0.030544	0.034122
	RMSE θ	0.21419	0.21594	0.15698	0.23828

Additionally, Tables 2 and 3 show the influence of firing strength thresholds f_{thv} and f_{thw} for linear and angular velocity respectively on the performance of RTSKIT2FNN. Table 2 represents the influence of f_{thv} with a fixed f_{thw} and the opposite in Table 3. We notice that larger values of f_{thv} or f_{thw} generate larger numbers of rules and improve the performance of RTSKIT2FNN in general. However, when the value of f_{thv} or f_{thw} is too large, the performance saturates for reason that the online learning process achieved the local minimum, and also the number of rules change according to the learning algorithm types VLR and FLR as shown in Table 4 for the same network. From the two tables the small RMS errors are obtained when $f_{thv} = 0.6$ and $f_{thw} = 0.3$.

Table 4 compares the RMSE values and generated rules number for other related networks ; we can note that the T2FNN networks have the smaller RMSE compared to T1FNN which is represented by two structures ANFIS and RFNN. This note shows the high superiority of T2FNN structure for the mobile robot trajectory tracking. The T2FNN is divided into two types in terms of learning principle: static and evolving. For the static type we have two algorithms IT2FNN-SM that uses sliding mode for learning and IT2FNN which uses the gradient descent. For the evolving learning type we have SOT2FNN-PSO, SEFT2FNN, SIT2FNN, RTSKIT2FNN-FLR, and RTSKIT2FNN-VLR. We can note that the IT2FNN-SM has the smaller RMSE than IT2FNN with gradient descent algorithm, and we can note also that the evolving algorithms have lower RMSE than the static algorithms; this demonstrates the superiority of the evolving algorithms. For the evolving algorithms, the proposed RTSKIT2FNN-VLR has the smaller RMSE compared to the other evolving algorithms. This result shows the high superiority and the effectiveness of the proposed algorithm, we can also note that RTSKIT2FNN-VLR and SEFT2FNN have the largest number of generated rules and the lower RMSE this demonstrates, the relationship between the generated rule and the effectiveness of the evolving algorithms.

Table 4 Comparison results of networks RMSE

Networks	Learning principle	Rule number		RMSE x	RMSE y	RMSE θ
		v	w			
ANFIS [17]	static	–	–	0.1313	0.0760	0.323
RFNN [34]	static	–	–	0.1270	0.0697	0.337
IT2FNN-SM [26]	static	–	–	0.0805	0.0713	0.2853
IT2FNN [24]	static	–	–	0.0878	0.0811	0.2866
SOT2FNN-PSO [32]	evolving	4	2	0.0743	0.0639	0.265
SEFT2FNN [31]	evolving	6	5	0.0618	0.0607	0.228
SIT2FNN [33]	evolving	4	3	0.0789	0.0662	0.273
RTSKIT2FNN-FLR	evolving	5	4	0.0638	0.0623	0.2473
RTSKIT2FNN-VLR	evolving	7	14	0.0401	0.0305	0.1569

6 Conclusion

This paper proposed a recurrent TSK Interval Type-2 Fuzzy Neural Networks with online structure and parameters learning control scheme for path tracking of mobile robots. The proposed RTSKIT2FNN has successfully incorporated the recurrent frame of internal-feedback loops into the interval Type-2 Fuzzy Neural Networks. The online learning of RTSKIT2FNN, there is no need to determine an initial network structure for the reason that the structure learning algorithm enables the network to evolve online and generate the required network structure and rules based on the rule firing strength. The structure learning is performed concurrently with parameter learning using the proposed gradient descent algorithm with varied learning rates (VLR) to tune all parameters of the RTSKIT2FNN, the learning convergence and the stability of the control system are proved using the discrete-type Lyapunov function. The experimental results have shown the obvious superiority of the proposed control method for mobile robot trajectory tracking compared to the other control methods.

Acknowledgments The authors would like to thank Telecommunications Signals and Systems Laboratory (TSS) for the support given to this research project

References

1. Matraji I, Al Durra A, Haryono A, Al Wahedi K (2018) Trajectory tracking control of skid-steered mobile robot based on adaptive second order sliding mode control. Control Eng Practice 72:167–176
2. Saradagi A, Muralidharan V, Krishnan V, Menta S, Mahindrakar AD (2017) Formation control and trajectory tracking of nonholonomic mobile robots. IEEE Transactions on Control Systems Technology
3. Hao Y, Wang J, Chepinskiy SA, Krasnov AJ, Liu S (2017) Backstepping based trajectory tracking control for a four-wheel mobile robot with differential-drive steering. In: 2017 36th Chinese on control conference (CCC). IEEE, pp 4918–4923
4. Rubagotti M, Della Vedova ML, Ferrara A (2011) Time-optimal sliding-mode control of a mobile robot in a dynamic environment. IET Control Theory Appl 5(16):1916–1924
5. Xin L, Wang Q, She J, Li Y (2016) Robust adaptive tracking control of wheeled mobile robot. Robot Auton Syst 78:36–48
6. Wai RJ, Liu CM (2009) Design of dynamic petri recurrent fuzzy neural network and its application to path-tracking control of nonholonomic mobile robot. IEEE Trans Indus Electron 56(7):2667–2683
7. Hagras HA (2004) A hierarchical type-2 fuzzy logic control architecture for autonomous mobile robots. IEEE Trans Fuzzy Syst 12(4):524–539
8. Castillo O, Neyoy H, Soria J, Melin P, Valdez F (2015) A new approach for dynamic fuzzy logic parameter tuning in ant colony optimization and its application in fuzzy control of a mobile robot. Appl Soft Comput 28:150–159
9. Huq R, Mann GK, Gosine RG (2006) Behavior-modulation technique in mobile robotics using fuzzy discrete event system. IEEE Trans Robot 22(5):903–916
10. Hou ZG, Zou AM, Cheng L, Tan M (2009) Adaptive control of an electrically driven nonholonomic mobile robot via backstepping and fuzzy approach. IEEE Trans Control Syst Technol 17(4):803–815
11. Aissa B, Fatima C, Yassine A (2017) Data fusion strategy for the navigation of a mobile robot in an unknown environment using fuzzy logic control. In: 2017 5th International conference on electrical engineering-boumerdes (ICEE-B). IEEE, pp 1–6
12. Juang CF, Tsao YW (2008) A self-evolving interval type-2 fuzzy neural network with online structure and parameter learning. IEEE Trans Fuzzy Syst 16(6):1411–1424
13. Aissa BC, Fatima C (2017) Neural networks trained with levenberg-marquardtiterated extended Kalman filter for mobile robot trajectory tracking. J Eng Sci Technol Rev, 10(4)
14. Yacine A, Fatima C, Aissa B (2015) Trajectory tracking control of a wheeled mobile robot using an adaline neural network. In: 2015 4th International conference on electrical engineering (ICEE). IEEE, pp 1–5
15. Li Z, Deng J, Lu R, Xu Y, Bai J, Su CY (2016) Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach. IEEE Trans Syst Man Cybern Syst 46(6):740–749
16. Shojaei K (2016) Neural adaptive output feedback formation control of type (m, 0s) wheeled mobile robots. IET Control Theory Appl 11(4):504–515

17. Aissa BC, Fatima C (2015) Adaptive neuro-fuzzy control for trajectory tracking of a wheeled mobile robot. In: 2015 4th International conference on electrical engineering (ICEE). IEEE, pp 1–4
18. Jang JO (2011) Adaptive neuro-fuzzy network control for a mobile robot. J Intell Robot Sys 62(3–4):567–586
19. Juang CF, Lin CT (1998) An online self-constructing neural fuzzy inference network and its applications. IEEE Trans Fuzzy Syst 6(1):12–32
20. Lee CH, Chiu MH (2009) Recurrent neuro fuzzy control design for tracking of mobile robots via hybrid algorithm. Expert Syst Appl 36(5):8993–8999
21. Wai RJ, Lin YW (2013) Adaptive moving-target tracking control of a vision-based mobile robot via a dynamic petri recurrent fuzzy neural network. IEEE Trans Fuzzy Syst 21(4):688–701
22. Zadeh LA (1975) The concept of a linguistic variable and its application to approximate reasoning-I. Inf Sci 8(3):199–249
23. Mendel JM, John RB (2002) Type-2 fuzzy sets made simple. IEEE Trans Fuzzy Syst 10(2):117–127
24. Abiyev RH, Kaynak O (2010) Type 2 fuzzy neural structure for identification and control of time-varying plants. IEEE Trans Ind Electron 57(12):4147–4159
25. Lu X, Zhao Y, Liu M (2018) Self-learning interval type-2 fuzzy neural network controllers for trajectory control of a delta parallel robot. Neurocomputing 283:107–119
26. Kayacan E, Maslim R (2017) Type-2 fuzzy logic trajectory tracking control of quadrotor vtol aircraft with elliptic membership functions. IEEE/ASME Trans Mechatron 22(1):339–348
27. Lin FJ, Chou PH et al (2009) Adaptive control of two-axis motion control system using interval type-2 fuzzy neural network. IEEE Trans Ind Electron 56(1):178–193
28. Castro JR, Castillo O, Melin P, Rodríguez Díaz A (2009) A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks. Inform Sci 179(13):2175–2193
29. Lin YY, Chang JY, Lin CT (2014) A tsk-type-based self-evolving compensatory interval type-2 fuzzy neural network (tscit2fnn) and its applications. IEEE Trans Ind Electron 61(1):447–459
30. Juang CF, Huang RB, Lin YY (2009) A recurrent self-evolving interval type-2 fuzzy neural network for dynamic system processing. IEEE Trans Fuzzy Syst 17(5):1092–1105
31. Lin CM, Le TL, Huynh TT (2018) Self-evolving function-link interval type-2 fuzzy neural network for nonlinear system identification and control. Neurocomputing 275:2239–2250
32. Lin CM, Le TL (2017) Pso-self-organizing interval type-2 fuzzy neural network for antilock braking systems. Int J Fuzzy Syst 19(5):1362–1374
33. Lin YY, Liao SH, Chang JY, Lin CT et al (2014) Simplified interval type-2 fuzzy neural networks. IEEE Trans Neural Netw Learning Syst 25(5):959–969
34. Mon YJ, Lin CM, Leng CH (2008) Recurrent fuzzy neural network control for mimo nonlinear systems. Intell Autom Soft Comput 14(4):395–415

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Aissa Bencherif was born in Laghouat (Algeria), in January 1989. He received master degrees in automatic in 2013 from the University of Amar Thelidji, Faculty of Technology in Laghouat (Algeria). He is currently a PhD student in robotics system from the laboratory of signals and systems and telecommunication, His research interests concern: mobile robot localisation and control using artificial intelligence and optimisation algorithms.

Correspondence address: laboratory of signals and systems and telecommunication, University of Amar Thelidji Laghouat (Algeria). Algeria: BP 37G Laghouat, Algeria

Fatima Chouireb was born in Laghouat, Algeria, in 1969. She received her Electrical Engineering diploma in 1992, and her Magister and PhD degrees from Saad Dahlab University of Blida, Algeria in 1996 and 2007 respectively, in 1997 she joined the Electrical Engineering Department of Laghouat University, Algeria, as Assistant Lecturer. Since Decembre 2007, Dr Chouireb is an Assistant Professor at the same Department. She is also a Team Leader of “Signal, Image and Speech” research group of LTSS Laboratory, Laghouat University, Algeria. She is currently a Full Professor at the same Department. Her current research interests include: Signal, Image and Speech processing, computer vision, Localisation/Mapping/SLAM and mobile robotics problems.

Correspondence address: laboratory of signals and systems and telecommunication, University of Amar Thelidji Laghouat (Algeria). Algeria: BP 37G Laghouat, Algeria