Leader-Following Formation Control with an Adaptive Linear and Terminal Sliding Mode Combined Controller Using Auto-Structuring Fuzzy Neural Network

Masanao Obayashi, Kohei Ishikawa, Takashi Kuremoto, Shingo Mabu
Graduate School of Science and Engineering, Yamaguchi University, Ube, Yamaguchi, Japan
email: {m.obayas, s003vk, wu, mabu}@yamaguchi-u.ac.jp

Kunikazu Kobayashi
School of Information Science and Technology, Aichi Prefectural University, Nagakute, Aichi, Japan
email: kobayashi@ist.aichi-pu.ac.jp

Abstract—This paper proposes an intelligent formation control method of the leader and follower agents with nonlinear dynamics. In the proposed method, agents can exchange only information of positions of available agents and the follower agents follow the leader, taking a predefined formation. In the real world, the method that doesn’t need a lot of information to make cooperative behaviors is very useful for the case of environments existing communication delay and weak communication. In addition, each agent will be controlled by the linear and terminal combined sliding mode control method. Furthermore, to adapt the change of the environment, the Auto-Structuring Fuzzy Neural Control System (ASFNCs) is introduced to provide appropriate control inputs while coping well with disturbance and nonlinear dynamics. In the simulation, it is verified that the proposed method is useful in the point of performance of the leader-following formation control.

Keywords—formation control; linear sliding mode control; terminal sliding mode control; auto-structuring fuzzy neural network

I. INTRODUCTION

Recently, formation problems have attracted more researchers as their importance and necessity are known widely. For example, Hou et al. [1] proposed a robust adaptive control approach to solve a consensus problem of Multi Agent Systems (MAS), Song et al. [2] and Cheng et al. [3] dealt with leader-follower consensus problem. Defoort et al. [4], Morbidi et al. [5] proposed a formation following control using sliding mode control, a kind of representative robust control methods. Cui et al. [6] dealt with leader-follower formation control of underactuated autonomous underwater vehicles. Yu et al. [7] dealt with time-varying velocity cases of a distributed leader-follower flocking control for multi-agent dynamical systems. In the view point of controllers using soft computing tools, Chang et al. [8] used fuzzy, Lin [9] used fuzzy basis function network, Chen et al. [10] used neural networks, Lin et al. [11] used reinforcement Q learning method, for formation control. However, designed controller structures for these formation or consensus problems are fixed, therefore, their methods are no flexible for changes of environments and they have possibility of useless computation load. Many of the control methods mentioned above are using linear sliding mode control method. Recently, terminal sliding mode control method has been used for constructing the following controller because of its superiority on fast finite time convergence and less steady state errors to objective states of the agents, Yu et al. [12], Zou et al. [13], Chang et al. [14].

We have already proposed the leader-following formation control method, Obayashi et al. [15] that had the following three features; use of linear sliding mode control, and the controller had the optimal structure, named ASFNCs, using self-structuring algorithm adapting to change of environments making use of the concept of Cheng et al. [16] and then introducing of making only use of the positions of agents without velocities of them.

In this paper, we consider to apply the superior point of the terminal sliding mode control to our previous work, Obayashi et al. [15], that is, to switch the linear sliding mode control and terminal one appropriately.

The rest of the paper is as follows: in Section II preparation to be required to follow the paper. The structure of the proposed system are described in Section III. The controller design is described in Section IV. In Sections V, VI, and VII, learning algorithm, following control problem and computer simulation are described, respectively. We have a conclusion in Section VIII.

II. PREPARATION

The dynamics of each agent consisting of MAS is described as following the $n$th order nonlinear differential equation,

$$x^{(n)} = f(x, u_{AFNS}) + \Delta f(x, u_{AFNS}) + d$$

$$= f(x, u_{AFNS}) - h \cdot u_{AFNS} + h \cdot u_{AFNS} + \delta_1,$$  \hspace{1cm} (1)

$$= f_1(x, u_{AFNS}) + h \cdot u_{AFNS} + \delta_1,$$

where $x = [x \hspace{0.2cm} \dot{x} \hspace{0.2cm} \cdots \hspace{0.2cm} x^{(n-1)}]^T \in \mathbb{R}^n$: observable states of the agent, $f(x)$ unknown continuous function, $h$ a predefined constant, $u_{AFNS}$ control input, $\delta_1$ term including the indefinite term $\Delta f(x, u_{AFNS})$ of the agent and the disturbance. $\delta_1$ is assumed that $|\delta_1| < D_1$. Here, $D_1$ is a predefined positive constant.
III. STRUCTURE OF THE PROPOSED SYSTEM

The proposed system in this paper is shown in Figure 1.

![Figure 1. Structure of the proposed multi agent control system.](image)

The objective of the control: The derivation of the control law which makes the orbit \( x \) of the states of the agent follow the reference orbit \( x_v \) accurately. We define the following error vector \( e = [e, e', \ldots, e^{(n-1)}] \) as

\[
e = x_v - x,
\]

where \( x_v \) : the reference vector, that is, their virtual agent state vector followed by its own agent. The virtual agent has its position and velocity decided by observing the position of the leader (see Section VI).

IV. CONTROLLER DESIGN

Figure 2 shows the ASFNCS. The ASFNCS consists of the fuzzy neural network controller with function of node adding/pruning (Auto-structuring FNNC) and the robust controller.

A. Fuzzy neural network controller (FNNC)

Figure 3 shows the structure of the FNNC. \( \Gamma_k \), output of \( k \) th node of the hidden layer, means the fitness of rule \( k \).

\[
\Gamma_k(e) = \prod_{j=1}^{R} \exp \left\{ -\frac{(e^{(i-1)} - m_{ij})^2}{\sigma_{ij}} \right\},
\]

\( (k = 1, \ldots, R), \)

\[
u_{asfnn} = \sum_{k=1}^{R} \xi_k \cdot \Gamma_k,
\]

where \( m_i = [m_{i1}, m_{i2}, \ldots, m_{in}] \), \( \sigma_i = [\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{in}] \) are center and width vectors of the \( k \)th rule output function, respectively. \( R \) is the number of the rule nodes. \( u_{asfnn} \) is the output of the FNNC. \( \xi_k \) is the weight between the \( k \)th rule node and the output of the FNNC.

The output \( u^* \) of ideal controller is like that,

\[
u^* = u_{asfnn} + \phi_1 = \sum_{k=1}^{R} \xi_k \cdot \Gamma_k + \phi_1,
\]

\( \phi_1 \) : difference between ideal controller and approximated optimal FNNC.

However, \( u^* \) would not be used for controller design actually (see Section IV B).

The estimated value \( \hat{u}_{asfnn} \) of the optimal FNNC \( u^*_{asfnn} \) is as follows,

\[
\hat{u}_{asfnn} = \sum_{k=1}^{R} \hat{\xi}_k \cdot \Gamma_k,
\]
where $\hat{\xi}_k$ is the estimate value of the optimal $\xi^*$. 

**B. Auto structuring mechanism of FNNC**

Node adding: Nodes adding process is as follows,

$$\Gamma_{\text{max}} = \max(\Gamma_k), \quad k = 1, 2, \cdots, R(t),$$

(7)

where $R(t)$ is the number of the node at time $t$. When the next relation exits,

$$\Gamma_{\text{max}} \leq \Gamma_{ih},$$

(8)

where $\Gamma_{ih} \in [0, 1]$ is a pre-defined threshold and given by trial and error, a new node $\sigma_i$ is added and the center and width of the node are like this,

$$m_{R_{ij}} = e^{i-1}, \quad \sigma_{R_{ij}} = \sigma_i, \quad (i = 1, \cdots, n),$$

(9)

where $\sigma_i$ is a predefined positive constant and the weight $\xi_{k_{ij}}$ between the new node and the output is set as

$$\xi_{R_{ij}} = \xi_i,$$

(10)

where $R_{ij}$ is the number of the added node.

Node pruning: Nodes pruning process is as follows, the cost function for measuring the important index $a_i$ of the node $k$ is defined as

$$E_{i} = \frac{1}{2}(u^* - \sum_{k=0}^{R} \hat{\xi}_k \Gamma_k)^2,$$

(11)

using the Taylor series expansion, where the parameter falls into the local minimum, the sensitivity of $E_i$ for $\hat{\xi}_k$ becomes next equation,

$$E_i(\hat{\xi}_k) \approx \frac{1}{2}(\frac{\partial E_i}{\partial \xi_k})^2, \quad k = 1, 2, \cdots, R.$$

(12)

Here, we define the important index of the node as

$$a_i(t+1) = a_i(t) \times \exp \left\{ -\beta \times \frac{\partial E_i}{\partial \xi_k}(\hat{\xi}_k) \right\},$$

(13)

and $\delta[*]$ function to decide whether make $a_i$ decrease is defined as

$$\delta[E_i(\hat{\xi}_k)] = \begin{cases} 1 & \text{if } E_i(\hat{\xi}_k) \geq 0 \\ 0 & \text{if } E_i(\hat{\xi}_k) < 0 \end{cases}$$

(14)

This means that if the rule $k$ is not activated enough, that is, $E_i(\hat{\xi}_k) \geq 0$, then, $a_k$ decreases. If $a_k \leq a_{ih}$, the node $k$ is considered as unnecessary node and it is deleted. Here $a_{ih}$ is a predefined positive constant and given by trial and error. The number of deleted nodes is presented as $R_i$. The output $u_{afsn}$ of the FNNC introducing the auto-structuring mechanism is as follows,

$$u_{afsn} = \sum_{k=1}^{\hat{R}} \hat{\xi}_k \Gamma_k - \sum_{p=1}^{\hat{R}} \hat{\xi}_p \Gamma_p,$$

(15)

where $u_k$ is the output after adding new nodes and, $u_p$ is the output after pruning nodes. With (15), we get

$$\tilde{u}_{afsn} = u^* - \hat{u}_{afsn} = \sum_{k=1}^{\hat{R}} \hat{\xi}_k \Gamma_k + \xi_*$$

(16)

where $\hat{\xi} = \xi^* - \hat{\xi}$. 

**C. Stability analysis**

The structure of the auto-structuring fuzzy neural control system (ASFNS) is shown in Figure 2. The output $u_{ASFNCS}$ of the ASFNS is as follows, Cheng et al. [16],

$$u_{ASFNCS} = u_{afsn} + u_*,$$

(17)

$u_{asfnn}$ is described by (15), and $u_*$ is the output of the robust controller. To derive an adaptive controller, we define the sliding variable $s$ as

$$s = e^{(n-1)} + k_1 e^{(n-2)} + \cdots + k_{n-1} e^{(n-1)}$$

(18)

+ $\gamma k_{n1} e^{(n)} + (1-\gamma) k_{n2} e^{(n)}$, where each of all the coefficients $k_i$ is pre-defined positive constant, $\gamma (0 \leq \gamma \leq 1)$ is a parameter. When $\gamma = 0.0$, 1.0 , sliding variable (18) can be regarded as LSM, TSM, respectively. In this paper, $\gamma$ changes nonlinearly from 1.0 to 0.0 during the controller working. The $p$ and $q$ are positive odd integers, which satisfy the following condition:

$$p > q.$$  

(19)

Then, a sliding mode controller can be designed as follows:

$$u_{SMC} = u_E + u_H.$$  

(20)

An equivalent controller $u_E$ is expressed as

$$u_E = h^{-1}( -f_{n1} u + x^{(n)} + k_1 e^{(n-1)} + \cdots + k_{n-1} e^{(n-1)} + \gamma k_{n1} e^{(n)} + (1-\gamma) k_{n2} e^{(n)}$$

(21)

$$u_H = D_h \text{sgn}(s).$$

(22)

Using (1), (2), (18) and (21), we get

$$\dot{s} = e^{(n)} + k_1 e^{(n-1)} + \cdots + k_{n-1} e^{(n-1)}$$

+ $\gamma k_{n1} e^{(n)} + (1-\gamma) k_{n2} e^{(n)}$

(23)

Using Eq. (16), Eq. (23) can be rewritten as
\[ \dot{s} = h(u^* - u_{asfns} - u_i) = h(\sum_{k \in \mathbb{N}} z_k \Gamma_k + \varepsilon - u_i). \]  

In this paper, the robust controller is used to eliminate the effect of error \( \varepsilon \). Consider the Lyapunov function candidate in the following form:

\[ V = \frac{1}{2} \dot{x}^2 + \frac{h}{2 \eta_\rho} \sum_{k = 1}^m \bar{z}_k^2 + \frac{h}{2 \eta_\rho} \bar{\rho}^2. \]  

Take the derivative of Eq.(25) and using Eq.(24), it is concluded that

\[ \dot{V} = s \dot{s} + \frac{h}{\eta_\rho} \sum_{k = 1}^m \bar{z}_k \dot{z}_k + \frac{h}{\eta_\rho} \bar{\rho} \dot{\bar{\rho}}. \]

For achieving \( \dot{V} \leq 0 \), the adaptation law and the robust controller are chosen as

\[ \frac{\dot{z}_k}{\eta_\xi} = -\frac{\dot{s}}{\eta_\xi}, \quad k = 1, 2, \ldots, \hat{K}, \]

\[ u_s = \hat{\rho} \text{sgn}(s), \]

\[ \hat{\rho} = -\dot{\rho} = -\eta_\rho |\hat{\rho}| \]

where \( \eta_\rho \) and \( \eta_\xi \) are positive constants. Then Eq. (26) can be rewritten as

\[ \dot{V} = h(s \varepsilon - \hat{\rho}|\dot{s}| - (\rho - \hat{\rho})|s|) \leq h|\varepsilon| - |\rho||s| \]

\[ = -h(\rho - |\varepsilon||s|) \leq 0 \]  

If \( |\varepsilon| \leq \rho \), \( \dot{V} \leq 0 \) holds. By Barbalat’s Lemma, \( s \to 0 \) as \( t \to \infty \), then the stability is guaranteed.

V. LEARNING ALGORITHM

We introduce the online learning algorithm to adjust the parameters of ASFNS, Cheng et al. [16]. We derive the algorithm the gradient-decent method. First, the adaptive law shown in (27) can be written as

\[ \frac{\dot{z}_k}{\eta_\xi} = \frac{\dot{s}}{\eta_\xi}. \]

According to the gradient-decent algorithm, the adaptive law of \( \dot{z}_k \) also can be represented as

\[ \frac{\dot{z}_k}{\eta_\xi} = \frac{\partial E_2}{\partial z_k} = \frac{\partial E_2}{\partial z_k} \frac{\partial z_k}{\partial \xi_0} \frac{\partial \xi_0}{\partial \xi_{o_3}} \frac{\partial \xi_{o_3}}{\partial \hat{\rho}} = \frac{\partial E_2}{\partial z_k} \frac{\partial z_k}{\partial \xi_0} \frac{\partial \xi_0}{\partial \eta_{o_3}} \frac{\partial \eta_{o_3}}{\partial \eta_\rho} \frac{\partial \eta_\rho}{\partial \eta_\xi} \]

where \( E_2 = 1/2(x_k - \chi)^2 \) is defined as the cost function. From Figure 3, (3), and (4), \( net^{[2]} = \Sigma_{-1} \left[ (e^{t-1} - \hat{m}_b)^2 / (\hat{\sigma}_v)^2 \right], \)

\[ y_{i_{o_3}}^{[2]} = \exp(\text{net}^{[2]}), \quad y_{i_{o_3}}^{[3]} = \sum \xi_i \left[ y_{i_{o_3}}^{[2]} = \sum \xi_i \xi_0 \Gamma_k \right]. \]  

\( m_{ki} \) means the number of layer in Figure 3. Thus the Jacobian term \( \partial E_2 / \partial \xi_{o_3} \) is obtained through observation of (31) and (32). Thus, the adaptive law for the estimation terms of means \( \hat{m}_{ki} \), and variance \( \hat{\sigma}_{ki} \) can be derived as

\[ \hat{m}_{ki} = \eta_\omega \frac{\partial E_2}{\partial \hat{m}_{ki}} = \eta_\omega \frac{\partial E_2}{\partial \eta_\omega} \frac{\partial \eta_\omega}{\partial \xi_{o_3}} \frac{\partial \xi_{o_3}}{\partial \eta_{o_3}} \frac{\partial \eta_{o_3}}{\partial \eta_\rho} \frac{\partial \eta_\rho}{\partial \eta_\xi} \]

\[ \hat{\sigma}_{ki} = \eta_\sigma \frac{\partial E_2}{\partial \hat{\sigma}_{ki}} = \eta_\sigma \frac{\partial E_2}{\partial \eta_\sigma} \frac{\partial \eta_\sigma}{\partial \xi_{o_3}} \frac{\partial \xi_{o_3}}{\partial \eta_{o_3}} \frac{\partial \eta_{o_3}}{\partial \eta_\rho} \frac{\partial \eta_\rho}{\partial \eta_\xi} \]

\[ \eta_\omega, \eta_\sigma \] are positive constants.

A. The algorithm of ASFNS

Step 1 Initialize the parameters of ASFNS.
Step 2 Calculate \( s \) using (18).
Step 3 Calculate \( u_{asfns} \) using (15) and update \( \hat{\rho}, \hat{m}, \hat{\sigma} \) using (32)-(34).
Step 4 Calculate \( u_s \) using (28) and adjust \( \hat{\rho} \) using (29).
Step 5 Calculate \( u_{asfns} \) using (17) and input it to the agent.
Step 6 If the controlling time is over, the simulation ends, else go to Step 2.

VI. LEADER-FOLLOWING FORMATION CONTROL PROBLEM

The dynamics (1) of the \( i \)th agent in multi agents is rewritten as

\[ \dot{x}_i(t) = f(t, x_i(t), \dot{x}_i(t)) + d_i + u_{i, \text{ASFNC}}. \]

where \( t \) : time, \( x_i = (x_{i_x}, x_{i_y})^T, \dot{x}_i = (\dot{x}_{i_x}, \dot{x}_{i_y})^T \) : the position, velocity of the agent, respectively. \( f(t, x_i(t), \dot{x}_i(t)) \in \mathbb{R}^2 \) : a term including the nonlinearity \( u_{asfns} = (u_{i_x}, u_{i_y})^T \in \mathbb{R}^2 \), \( u_{i, \text{ASFNC}} = (u_{i_x}, u_{i_y})^T \in \mathbb{R}^2 \) : control input to the \( i \)th agent, \( d_i \in \mathbb{R}^2 \) : disturbance. In this paper, we consider the following control method that make the agents follow and catch the leader keeping the formation by the group. Cui et al. [6]. We assume that agents could exchange only positions information of the agents each other. Observing only the position of the leader, each agent constructs the virtual orbit to the leader. Realizing this virtual orbit by the virtual agent, and the agent follows its own agent. The following describes the method of following the leader by the virtual agent.
The following error of the $i$th virtual agent is expressed as

$$e_{vi} = ((z_i + \omega_i) - x_i) + \sum_{j \in N_i} ((z_i + \omega_i) - z_j),$$  

where $e_{vi}$: consensus error, $z_i = x_i$, $\omega_i = (x_i - x_j)$, $x_i \in \mathbb{R}^2$ : the position of the $i$th agent in a 2-dimensional space.

The following error $x_{ei}$ of the $i$th virtual agent considering the integral term to reduce the steady state error is expressed as

$$x_{ei} = e_{vi} + \phi_i + \int_0^t e_{vi} \, dt,$$  

where $\phi_i = [\phi_{1i}, \phi_{2i}]^T$, the derivative of $\phi$ with respect to time is defined as

$$\dot{\phi}_i = -\beta_i(\phi_i) - Kx_{ei},$$  

where $\beta_i(\phi) = [\lambda_i \tanh(\phi_i / \lambda_i), \lambda_2 \tanh(\phi_i / \lambda_2)]^T$, $K = \text{diag} \{ k_{1i}, k_{2i} \}$, $\phi_i(0) = 0$.

The velocity $\dot{x}_{vi}$ of the $i$th virtual agent is defined as

$$\dot{x}_{vi} = \beta_i(\phi_i) + \beta_2(\phi_i),$$  

where $\beta_2(\phi_i) = [k_{1i} \tanh(\phi_i / k_{1i}), k_{2i} \tanh(\phi_i / k_{2i})]^T$, $k_{1i}, \lambda_i$ are positive constants.

### VII. COMPUTER SIMULATION

We demonstrate the effectiveness of our proposed method by trying to follow the leader (target) by the multi agent constructing by four agents.

The horizontal and vertical direction of velocity of each agent are $\dot{x}_{i1} = 0.02 \cos(0.05t)$, $\dot{x}_{i2} = 0.015 \sin(0.03t)$, respectively, and the initial position of the leader is given as $x = [0, 0]^T$. The initial horizontal and vertical positions of the agents are both given randomly in the range of $[-5, 5]$, and the initial velocities of all the four agents are equal to 0. The parameters used in the simulation are shown in Tables I and II. The dynamics and disturbances of the each agent are given as

$$f(t, x_i(t), \dot{x}_i(t)) = \begin{bmatrix} \sin(x_{i1}) + \dot{x}_{i1} \cos(\dot{x}_{i2}) \\ \cos(x_{i2}) + \dot{x}_{i2} \sin(x_{i1}) \end{bmatrix},$$  

$$d_i = \begin{bmatrix} -\sin(2t) \\ \sin(t) \end{bmatrix}.$$  

The information exchanges among agents each other are carried out according to Figure 4. Each agent behaves dispersive and cooperatively through information exchanges and follows the leader, taking the predefined formation. In this simulation, from Figure 4, only agent 1 and 2 can observe the position of the leader and the agent 4 can observe the positions of the agents 1 and 3, and so on. Sampling time is 0.01$[s]$ and total controlling time is 50 $[sec]$.

Each of Figures 5-11 has 5 orbits: red is for target (leader), other 4 orbits are for follower agents. The orbits of the target and follower agents in the case of the conventional method, LSM, are shown in Figure 5. Figure 6 is the enlarged figure of the transient (initial part of) positions of the leader and follower agents. The orbits of the leader and agents in the case of the conventional method, LSM, are shown in Figure 7. Figure 8 is the enlarged figure of the transient (initial part of) positions of the agents and the leader. Comparing these four figures, TSM method is superior to the LSM method in the point of consensus error of the positions of transient states of the leader and follower agents. However, the consensus error of them in the point of the steady state by LSM method is smaller than those of TSM method.

<table>
<thead>
<tr>
<th>TABLE I. PARAMETERS OF THE VIRTUAL AGENT.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>$\lambda_1, \lambda_2$</td>
</tr>
<tr>
<td>$k_{1i}, k_{2i}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TABLE II. PARAMETERS OF THE ASFNCS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
</tr>
<tr>
<td>$k_1$</td>
</tr>
<tr>
<td>$k_2$</td>
</tr>
<tr>
<td>$\Gamma_{th}$</td>
</tr>
<tr>
<td>$\sigma_c$</td>
</tr>
<tr>
<td>$\zeta$</td>
</tr>
<tr>
<td>$\xi_c$</td>
</tr>
<tr>
<td>$E_{th}$</td>
</tr>
<tr>
<td>$\beta$</td>
</tr>
<tr>
<td>$\alpha_{th}$</td>
</tr>
<tr>
<td>$\eta_{\zeta}$</td>
</tr>
<tr>
<td>$\eta_\rho$</td>
</tr>
<tr>
<td>$\eta_{\alpha_{th}} \cdot \eta_{\sigma}$</td>
</tr>
</tbody>
</table>
Figure 5. Trajectories of all the agents using LSM ($\gamma = 0$ in (18)).

Figure 6. Transient trajectories of all the agents using LSM ($\gamma = 0$).

Figure 7. Trajectories of all the agents using TSM ($\gamma = 1$ in (18)).

Figure 8. Transient trajectories of all the agents using TSM ($\gamma = 1$).

Figure 9. Trajectories of all the agents using TLSM.

Figure 10. Transient trajectories of all the agents using TLSM.

Figure 11. Trends in distances error between a leader and all follower agents.

Figures 9-10 show the results using our proposed LSM and TSM combined control, changing $\gamma$ in (18) from 1.0 to 0.0 non-linearly. Figure 11 shows trends in the consensus error between a leader and all follower agents for LSM and TLSM. From Figure 11, it can be found that consensus error by our proposed method is smaller than that of the conventional LSM method.

In order to confirm that our method is superior to the LSM method, we adopted the next performance index $J$, average error area;

$$J = \frac{1}{T} \sum_{t=0}^{T} e(t), \quad e = \sum_{i=1}^{N} \left( \| x^i - x_L^i \|^2 + \| y^i - y_L^i \|^2 \right),$$

(42)
where $T$ means controlling time, $x_L = [x_l, \dot{x}_l]$ , $y_L = [y_l, \dot{y}_l]$ mean position and velocity vectors in $x$ direction and $y$ direction of the leader, respectively. $x_{f_i}$ and $\dot{x}_{f_i}$ mean those of the $i$th follower agent. $e_{j}$ mean sum of consensus errors of between the leader and all the follower agents at time $t$, average consensus error area, respectively. Table 3 shows comparison of the control performances, that is, transient consensus error and average error area. Table 3 shows our proposed method is superior to the conventional LSM method.

VIII. CONCLUSION

In this paper, we proposed the simple and useful method, that is, an adaptive LSM and TSM combined formation controller design method with auto-structuring fuzzy neural network. Additionally, feature of the proposed method is that all the agents can exchange only information of positions according to the network structure of multi agents like Figure 4, making velocities of their virtual agents, and it has the variable structure to adapt for changes of the environment.

| TABLE III. PERFORMANCE COMPARISONS OF BOTH METHODS. |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
|                 | Transient consensus error |                 |                 |                 |                 |                 |                 |
| agent           | a_1             | a_2             | a_3             | a_4             | Total           |                 |                 |
| LSM             | 0.148           | 0.161           | 0.234           | 0.24            | 0.78            |                 |                 |
| TLSM (Proposed) | 0.14            | 0.145           | 0.226           | 0.237           | 0.748           |                 |                 |
| Average consensus error area |                 |                 |                 |                 |                 |                 |                 |
| agent           | a_1             | a_2             | a_3             | a_4             | Total           |                 |                 |
| LSM             | 0.356           | 0.368           | 0.535           | 0.541           | 1.8             |                 |                 |
| TLSM (Proposed) | 0.316           | 0.321           | 0.478           | 0.489           | 1.604           |                 |                 |

REFERENCES


