ERRATUM

FLEXIBLE STRUCTURE MULTIPLE MODELING USING IRREGULAR SELF-ORGANIZING MAPS NEURAL NETWORK

[International Journal of Neural Systems, Vol. 18, No. 3 (2008) 233–256]

ALIREZA FATEHI*

*Advanced Process Automation & Control (APAC) Research Group,
Faculty of Electrical Engineering,
K. N. Toosi University of Technology, Tehran, Iran
fatehi@kntu.ac.ir*

KENICHI ABE

*Department of Computer Science,
College of Engineering Nihon University,
Koriyama, Japan
abe@cs.ce.nihon-u.ac.jp*

The MMSOM identification method, which had been presented by the authors, is improved to *the multiple modeling by the irregular self-organizing map* (MMISOM) using the *irregular SOM* (ISOM). Inputs to the neural networks are parameters of the instantaneous model computed adaptively at every instant. The neural network learns these models. The reference vectors of its output nodes are estimation of the parameters of the local models. At every instant, the model with closest output to the plant output is selected as the model of the plant. ISOM used in this paper is a graph of all the nodes and some of the weighted links between them to make a *minimum spanning tree* graph. It is shown in this paper that it is possible to add new models if the number of models is initially less than the appropriate one. The MMISOM shows more flexibility to cover the linear model space of the plant when the space is concave.

Keywords: Multiple modeling; irregular self-organizing map; adaptive identification; neural network.

1. Introduction

Most of the controller techniques for plants (dynamic systems) need a model of it to design the controller based on this model. To obtain the model, a model structure is considered and then identification algorithms are used to estimate the parameters of it. One of the main problems in the estimation of the parameters of a model for a plant is how to estimate them when the plant behavior changes during the time. In this situation, the identified linear model can only model part of the plant's dynamical behavior. Therefore, the parameters need to be updated again as the plant situation changes with time. Adaptive

identification has been introduced for this kind of problems, which include different types of estimation methods like recursive least square error (RLSE) estimation and Kalman filter.¹ But the speed of adaptation may result in a large transient error. To overcome this problem and speed up the identification process, *multiple modeling* (MM) method has been introduced.² In multiple modeling there are more than one estimation model for the plant. Each model is valid in a distinct subset of the plant behavioral space. At every instant, according to some defined criteria, effect of the best model is selected or a combination of them is produced to interpret the plant

*Corresponding author.

behavior, instead of adapting the parameter of a single model.

Theory of multiple modeling started from the state estimation of a Gauss-Markov process by Magill³ in 1965. In this type of modeling, which later was called *interactive multiple modeling* (IMM), a bank of models is constructed for the process. The weighted linear combination of the models' outputs estimate the process output. This idea was later developed by other researchers like Sims & Lainiotis,⁴ Ackerson & Fu,⁵ Gholson & Moose⁶ and others. APDAF by Gauvrit⁷ and *partitioning* by Lainiotis⁸ are apparently other variations of Magill's method. Takagi-Sugeno-Kang (TSK) fuzzy system⁹ is also another type of IMM where the output of the system is linear combination of the output of each rule. In the original TSK fuzzy system, output of each rule is a linear combination of its inputs. But it was extended to the local models combined by fuzzy TSK rules.¹⁰ Local Linear Model Tree (LoLiMoT)¹¹ is another type of multiple modeling. In LoLiMoT, models are in a tree structure so that the maximum error when using each model is lower than a threshold.

Multiple modeling is not limited to interactive modeling and estimation. Lainiotis¹² introduced *partitioned adaptive control* (PAC) by adding a controller to each model. Linear combination of the controllers outputs acts as the control signal. Athans *et al.*¹³ used the *multiple model adaptive control* (MMAC) to control an aircraft. Later on, other researchers presented new versions or improved others of what is analogues to IMM, which we may call *interactive multiple control* (IMC), like He *et al.*,¹⁴ Schiller & Maybeck,¹⁵ Gollee & Hunt¹⁶ used an RBF type of coefficient weights to combine the models. Townsend *et al.*¹⁷ applied *Dynamic Matrix Control* (DMC) as the controller and Tan and Chiu¹⁸ used internal model controller as the controller in an interactive multiple controller. Wolpert and Kawato¹⁹ introduced modular learning using the concept of brain decision during control of motion in different contexts. In this method, a set of forward models (predictor) and inverse models (controller) is obtained for different behavioral spaces of the plant. At every instant, forward model determines the contribution of each inverse model's output in final control signal.

Middelton *et al.*²⁰ developed Magill's method from interactive to *switching multiple modeling*

(SMM), in which only one of the models, which better models the behavior of the plant, is selected as the model of the plant, instead of the weighted combination of the models in IMM. Petridis & Kehagias²¹ proposed the switching version of the partition algorithm of Lainiotis⁸ and applied it to nonlinear systems.

In 1986, Martensson²² proposed the concept of switching control. His method soon found its way in multiple modeling.²⁰ Pait & Morse²³ presented *switching multiple control* (SMC) for linear MIMO plant. Narendra & Balakrishnan²⁴ improved this idea and applied it to a linear time-variant plant by model reference adaptive control. They also applied their method to nonlinear plants and studied the effect of different sets of adaptive and fixed models in control systems.²⁵ Banerjee *et al.*²⁶ defined and computed nonlinear model between the linear models to soften switches between the controllers.

Multiple modeling and control has established itself in industrial applications, specially in air traffic^{6,27} control and aviation,^{28–30} bio-engineering^{16,14} and process control.^{26,17,31,32} Some other applications are robotics,²⁵ vehicle control,³³ pattern recognition,³⁴ weather forecasting,³⁵ intrusion detection system (IDS),³⁶ supervisory control³⁷ and power electronics.³⁸

Self-Organizing map (SOM) neural network is a kind of unsupervised learning neural network, which maps and clusters the characteristics of a collection of objects to a number of clusters.³⁹ Most of the applications of this network are in signal processing, pattern recognition and related topics. There are a few works on application of SOM in identification of processes.^{39–42} In these works, SOM is used to cluster the state-space of the plant, therefore, inputs to the network are the state variables of the plant. Fatehi & Abe introduced the *multiple modeling by self-organizing map* (MMSOM),^{43,44} to compute and adapt set of multiple models for a linear time-variant system. In the MMSOM a set of local models (bank of model) are identified for different behavioral situations of the plant using SOM. On the contrary to the previous works, inputs to the SOM neural network are the parameter of the instantaneous (adaptive) models. Therefore, in MMSOM the plant model space is partitioned instead of plant state space. Then, the best model for the present plant situation is selected among them. It is shown

that the statistical distribution of the parameters of local models in MMSOM approximates the statistical distribution of plant parameters for a time varying linear plant.⁴⁵

Although MMSOM algorithm could present multiple model set for a linear time-varying plant, but there are some limitations to use SOM. First, there are a lot of links between its nodes, which degrades flexibility. To increase the flexibility, it is necessary to increase the number of nodes. Second, if the initial number of nodes in SOM is not sufficient to model the plant, we would like to increase the nodes without necessity to start again training of the network from the initial step. Because of the rectangular structure of SOM, increasing the number of nodes means adding a row of nodes. For example, in a $p \times q$ nodes SOM, addition of a node means addition of p or q nodes depending on the direction. To overcome the problem of flexibility, Kangas *et al.*⁴⁶ proposed the *irregular SOM* (ISOM) neural networks. Fortunately, it is possible to add or delete one node to or from this network, using the techniques proposed in Sec. 3.5.

In this paper, *multiple modeling by irregular self-organizing map* (MMISOM) is proposed.^{44,47} In the MMISOM, an ISOM neural networks represents the bank of models. ISOM is a graph of all the nodes and some of the weighted links between them such that they present a *minimum spanning tree* (MST) graph, in which integration of the weights is minimum. Using this type of network, it is possible to add a new model. The algorithm does not initially have any information about the proper number of models for the plant and starts with one or few models. New models are added, when the algorithm finds out that more models are required for the desired accuracy. The MMISOM has a good flexibility to cover the plant's linear model space specially when the plant switches between some distinct sub-spaces or when its space is concave.

ISOM has an irregular structure, comparing with other growing self-organizing networks like growing neural gas (GNG) of Fritzke,⁴⁸ incremental self-organizing map (GSOM) of Bauer and Villmann⁴⁹ and dynamic cell structures (DCS) of Bruske and Sommer⁵⁰ which have hyper-rectangular structure. This increases the flexibility of ISOM to cover concave spaces with fewer nodes, which means fewer models and controllers. This reduces the computation time in large problems.

The main contributions of the article are: (1) to propose a method to unsupervisedly construct the bank of model using ISOM, (2) use it for an on-line identification (3) increase the number of necessity models adaptively and (4) it is also shown that the performance of the MMISOM is at least as good as that of an adaptive identification method, and that it improves with time.

The paper is divided into 4 sections. In the next section, an overview of the ISOM is provided. Section 3 describes the MMISOM. Some of its properties are discussed through some illustrative examples. The conclusions are summarized in the last section.

2. ISOM Neural Network

2.1. SOM neural network

SOM is a neural network that learns its inputs by clustering them into some clusters using the Euclidean distance between input vectors and weight vectors of the output nodes, called *reference vector* (RV). It consists of a set of processing units or nodes. Every node in the map has a reference vector w_j . The input vector x is broadcast to all of the nodes. Nearest RV to input vector is called the winner node. After training the network, the winner node represent the cluster that the input vector belongs to. Training algorithm of SOM is as follows³⁹:

- Step (1)** Assign arbitrary initial values to RV's of nodes O_j , $j = 1, \dots, M$, where M is the number of nodes;
- Step (2)** At every iteration t , apply the input vector $x(t)$ to the network;
- Step (3)** Compute distances between RV of each node and the input vector:

$$d_j = \|x(t) - w_j(t)\| \quad (1)$$

where $\|\cdot\|$ denotes the Euclidean norm.

- Step (4)** Define the winner node O^* as:

$$O^* = O_j|_{d_j=d^*, d^*=\min_j d_j} \quad (2)$$

- Step (5)** Update the reference vectors by the learning rule:

$$w_j(t+1) = w_j(t) + \eta(t) [x(t) - w_j(t)] \mathcal{H}(O^*, O_j, t) \quad (3)$$

where $\eta(t)$ is learning rate, and the decreasing function $\mathcal{H}(O^*, O_j, t)$ is neighborhood function for winner node O^* .

Steps 2 to 5 are repeated for every new input to SOM. As a consequence of this algorithm, different RV's tend to become specifically tuned to different domains of the input x , i.e., clusters the input space. Each node of the network represents one of the clusters. RV of this node is the nearest RV to the values of that cluster according to Euclidean distance of Eq. (1).

2.2. MST graph

Consider a network of n nodes with possible *link weights* (LW) L_{ij} between every two nodes O_i and O_j of it. L_{ij} denotes a distance measure between nodes, like Euclidean distance or the cost of direct movement from one node to the other one. It is also assumed that

$$L_{ij} = L_{ji}. \quad (4)$$

Hence, we have

$$l = \binom{n}{2}, \quad (5)$$

possible links for a network of n nodes. For this weighted network, the *MST graph* is defined as a connected graph of all n nodes and some of the links in the network, where the sum of the link weights is minimum. As an example, consider the network shown in Fig. 1(a). Both of the graphs in Fig. 1(a) and (c) are MST. But, the graphs in Fig. 1(d) and (e) are not MST, because the former is not a connected graph and sum of the weights in the latter is not minimum.

Algorithm of obtaining an MST graph for a weighted network has thoroughly been explained in Sedgewich.⁵¹ It is based on this property that if the nodes of the network are divided into two connected parts, among all of the links that connects these two parts, the link with the minimum weight must be in any MST graph of the network. The algorithm starts with one of the nodes as the first part and the others as the second part of the network. Then, it picks the node in the second part which is closer to the node(s) of the first part, and add it and the corresponding link to the nodes of the first part. The algorithm continues with these new parts of the network until all of the nodes join the first part.

2.3. ISOM network

ISOM was proposed as a flexible structure self-organizing map neural networks.⁴⁶ The link weights

Fig. 1. Different possibility of graphs for a weighted link network (a). The graphs (b) and (c) are MST but the graphs (d) and (e) are not.

L_{jk} between the nodes of ISOM is defined as the Euclidean distance between RV's of node O_i and O_j :

$$L_{jk} = \|w_j - w_k\|. \quad (6)$$

Considering the above definition and the definition of the MST graph, the ISOM is defined as an SOM neural network with possible link between every two nodes, where the neighborhood of nodes are such that their links define an MST graph. Figure 2(a) illustrates a 2×2 network with 2 inputs and 4 output nodes. The RV of each node are as mentioned in the figure. Using the definition (6), links have weights shown in the figure. Using these weights between nodes, ISOM network consists of the links $\{L_{ad}, L_{bd}, L_{bc}\}$. This ISOM network is shown in Fig. 2(b). Definition of the neighborhood distance in ISOM is the same as in SOM but it is defined along links emanating from the nodes in the MST graph. Hence, in Fig. 2(b), O_a is a one-step neighbor of O_d and two-steps neighbor of O_b . Therefore,

Fig. 2. Extracting an ISOM: (a) The network with all its possible links, (b) The ISOM network with MST graph.

Neighborhood matrix of ISOM of Fig. 2(b) is:

$$\begin{array}{c|cccc}
 & O_a & O_b & O_c & O_d \\
 \hline
 O_a & 0 & 2 & 3 & 1 \\
 O_b & 2 & 0 & 1 & 1 \\
 O_c & 3 & 1 & 0 & 2 \\
 O_d & 1 & 1 & 2 & 0
 \end{array} \quad (7)$$

This matrix indicates the neighborhood distance between every two nodes. This neighborhood matrix is used during the training of ISOM instead of the regular one of the rectangular SOM.

An ISOM that was MST at one time, may not remain MST, because RV weights change during the training process of ISOM, which result in change in LW's. Therefore, the structure must be examined and possibly modified after each training step. This is a time consuming process. Fortunately, only the RV of the winner node and its neighbor(s) change in an ISOM. On the other hand, this change is small because the learning factor η is small, hence,

the adaptation is temporally smooth. Therefore, the change in each LW is also small. If the graph of ISOM is an MST graph it is unlikely that after a few steps the network is no longer an MST graph. Even if the graph is not an MST one any more, the difference between the sum of its weights and the minimum one is not large at the early steps. Thus, it is not necessary to compute the MST after each successive training. In fact, when the network is in its transition period, it is better to compute the MST frequently and when it is in the steady state period compute it infrequently.

3. The MMISOM Algorithm

3.1. An overview to the algorithm

Consider the dynamic plant $\mathcal{P}$ described by

$$y(t) = \varphi_{\mathcal{P}}^T(t)\theta_{\mathcal{P}}(t) + \omega(t) \quad (8)$$

where $\varphi_{\mathcal{P}} \in \mathbb{R}^{n_{\mathcal{P}}}$ is the regression vector of the plant, $\theta_{\mathcal{P}} \in D_{\mathcal{P}} \subset \mathbb{R}^{n_{\mathcal{P}}}$ denotes the time-variant

parameters vector of the plant, n_p is the number of regressor variables, ω is the white noise with mean zero and variance σ_ω^2 , and y is the output. $D_{\mathcal{P}}$ is the space of plant parameters which called plant behavioral space. Different behavior can be expressed by different values of $\theta_{\mathcal{P}}$. Plant linear model space $D_{\mathcal{M}} \subset \mathbb{R}^{n_{\mathcal{M}}}$ is the family of all the linear models $\theta_{\mathcal{M}}$ for the plant (8) with definition:

$$y(t) = \varphi_{\mathcal{M}}^T(t)\theta_{\mathcal{M}}(t), \quad (9)$$

where $\theta_{\mathcal{M}} \in D_{\mathcal{M}} \subset \mathbb{R}^{n_{\mathcal{M}}}$ and $\varphi_{\mathcal{M}} \in \mathbb{R}^{n_{\mathcal{M}}}$ are respectively the model parameter vector and the regressors of the model and $n_{\mathcal{M}}$ denote the number of the parameters of the model. For simplicity $\varphi_{\mathcal{M}}(t)$ is written as $\varphi_{\mathcal{M}}$. The problem is to determine parameters of M linear local models:

$$\Theta_{\mathcal{L}} = \{\theta^i | \theta^i \in \mathbb{R}^{n_{\mathcal{M}}}, i = 1, \dots, M\}, \quad (10)$$

in such a way that each θ^i is a good approximation of a region $d^i_{\mathcal{M}} \subset D_{\mathcal{M}}$ around θ^i , where $\cup_{i=1}^M d^i_{\mathcal{M}} = D_{\mathcal{M}}$. $\Theta_{\mathcal{L}}$ is called *bank of models*. The term *good approximation* will precisely be defined later in Sec. 3.4. Each $d^i_{\mathcal{M}}$ approximates some subset $d^i_{\mathcal{P}}$ of the plant behavioral space. At every instant, a model θ^i in $\Theta_{\mathcal{L}}$ is selected as the model of the plant. This model is supposed to be the best model for the plant according to some defined criteria.

In this paper, above multiple modeling problem is solved by using the irregular self-organizing map neural networks. The MMISOM algorithm is composed of two general steps: (1) Construction of the bank of local models, and (2) selection of the best model. The input and output values of the plant is used to identify an *instantaneous model* (IM) of the plant at every instant. This means sweeping $D_{\mathcal{M}}$ by producing and adapting θ^i 's. Inputs to the ISOM are IM's. ISOM clusters IM's, which are produced at every instant. Each cluster is represented by a neuron of the ISOM network. The reference vector of each node of ISOM represents a local model ($\theta^i, i = 1, \dots, M$) for the plant. As a result, the bank of local models is produced by and embedded in ISOM neural network.

The performance of all the local models are evaluated in parallel at every instant. The model with the best performance, i.e., closer response to the plant one, is selected as the best model for the plant at that time. Although the IM is also a model of the plant at that time, but the response of selected best

model is better than IM adapted during the transient time, as is illustrated in examples of Sec. 3.6. New models are added to the bank of local models in ISOM, in addition to adaptation of the present local models. In the next subsections details of the algorithm are presented.

3.2. Identification of the instantaneous model

IM is a linear model with defined structure $\mathcal{M}$. It should be identified by an adaptive identification method. Any kind of on-line identification methods¹ may be used to estimate the parameters of IM. We use recursive least squares estimation (RLS), since it is the basic on-line identification method which other techniques compared with it. The model

$$y = \varphi_{\mathcal{M}}^T(t)\theta^o(t) \quad (11)$$

is to be estimated with criteria

$$V(\theta^o, t) = \frac{1}{2} \sum_{i=1}^t \lambda^{(t-i)} (y(i) - \varphi_{\mathcal{M}}^T(i)\theta^o(i))^2 \quad (12)$$

Assume that the matrix $\Phi(t) = [\varphi(1), \varphi(2), \dots, \varphi(t)]^T$ has full rank, that is, $\Phi^T(t)\Phi(t)$ is non-singular, for all $t \geq t_0$. Given $\hat{\theta}^o(t_0)$ and $P(t_0) = (\Phi^T(t_0)\Phi(t_0))^{-1}$, the least-square estimate $\hat{\theta}(t)$ then satisfies the recursive equations¹:

$$\hat{\theta}^o(t) = \hat{\theta}^o(t-1) + K(t)(y(t) - \varphi_{\mathcal{M}}^T(t)\hat{\theta}^o(t-1)), \quad (13)$$

$$K(t) = P(t)\varphi_{\mathcal{M}}(t) = K(t-1)P(t-1)\varphi_{\mathcal{M}}(t) \times (\lambda I + \varphi_{\mathcal{M}}^T(t)P(t-1)\varphi_{\mathcal{M}}(t))^{-1}, \quad (14)$$

$$P(t) = (I - K(t)\varphi_{\mathcal{M}}^T(t))P(t-1)/\lambda \quad (15)$$

where λ is the forgetting factor, P is the covariance matrix and K is the adaptation gain. If output dimension is one identity matrix I in (14) is replaced by a scalar 1. The adaptive IM model obtained above follows the plant behavior and model it. When the plant behavior changes, the IM parameters also change. If the plant returns to the first situation later, the estimator (13) to (15) must again compute the IM model parameters for this situation. These values for the parameters had already been computed but have been lost later, because of the changes in the plant behavior. This drawback of adaptive estimation is specifically harmful when the plant changes abruptly. In this situation, IM parameters converges

to the new values slowly. So, its response cannot follow the plant fast enough. But in multiple modeling, the previous estimation parameters are stored in the form of local models. Every time that the behavior of the plant changes, the proper model is selected from the bank of local models. This takes a smaller amount of time compared with the time required for convergence of IM parameters.

3.3. Producing the ISOM bank of models

ISOM maps the input space $\mathcal{J}$ to the output space $\mathcal{O}$. $\mathcal{O}$ represents the quantization space of $\mathcal{J}$. In the MMISOM, the number of inputs to the ISOM is equal to the number of the estimation parameters in IM, i.e. $n_{\mathcal{M}}$. It has M output nodes, where M is the desired number of local models. Therefore, IM parameters obtained by Eq. (13) are given to the network at every instant. ISOM quantizes $D_{\mathcal{M}}$, the parameter space of IM, in the form of embedded local models.

If the plant behavior abruptly changes, the value of IM is not valid during the transition period until it adapts to the new behavior of the plant. Training the ISOM in this period must be ceased, otherwise the network learns some wrong data (invalid models). Therefore,

Training condition: if the cost of IM (Eq. (16)) is more than a threshold c , the training stops. It will start again when the cost become less than $b(\leq c)$.

3.4. Selection of the best model

As the bank of models is constructed, the next step is selection of the best model from it for the present behavioral situation of the plant. Selection of the best model needs dealing with two subject; what to compare, and how to compare.

Difference between the estimated outputs of the models and the output of the plant are compared to find the best model. For this purpose, models are operated in parallel, as shown in Fig. 3. The model with the lowest error is selected as the best model of the plant. In this stage the IM is also treated as one of the local models and take part in competition. The reason is that, training of the neural network can be on-line or off-line. The algorithm can introduce the model of the plant in the situations that they have been experienced for a few times. Therefore,

none of the models are suitable at the early steps before sufficient training. Fortunately, the algorithm can take advantage of the adaptive IM identifier, as the best model during that learning period. Hence, the quality of identification is at least as the single-model adaptive IM one but improves during the time.

Both instantaneous and long-term measures are needed to reliably and accurately estimate models. Integrating error in a period of time decreases the effect of noise. On the other hand, using the present error causes the algorithm to follow rapid changes in the plant. But, it degrades stability of the switches between models. A performance index incorporating these features has the from;

$$J^i(t) = \beta |e_i(t)| + \gamma \left(\sum_{\tau=0}^{t-1} \alpha^\tau e_i^2(t-\tau) \right)^{1/2} \quad (16)$$

where $e_i(t)$ is the error of the i th model output at time t , α and β and γ are design parameters and J^i is defined as the cost function of the model i . α and β and γ can be chosen to yield a desired combination of instantaneous and long-term accuracy measure. The forgetting factor α determines the memory of the index in rapidly switching environments and ensures boundedness of J^i for bounded e_i .²⁴

3.5. Increasing the number of models in the MMISOM

As mentioned in Sec. 1, considering appropriate number of models is important in multiple modeling. Insufficient models causes inaccuracy in the modeling. While using few models causes inaccuracy, using a very large number of models is computationally inefficient. Both of them decrease the efficiency of accurate modeling in a reasonable time. SOM is not suitable for increasing the number of neurons. But ISOM is perfect for this purpose. If the node is properly added, ISOM keeps its structure and remains MST. Using the method introduced below, it is possible to add just one neuron (model) while keep it MST.

Suppose the node O_* , with RV equal to w_* , is going to be added to an n -nodes ISOM. The Euclidian differences between w_* and all of the $w_i, i = 1, \dots, n$, in the network are computed using Eq. (6). The new neuron connects to its nearest neuron in the present network, Fig. 4. So, the graph of the new ISOM is still MST.

Fig. 3. The structure of the plant and the models in the MMISOM. Identification starts with instantaneous modeling and then producing the bank of models as prototype models in ISOM. Best model at each time is selected among these models including to the instantaneous model.

Assume that there are m models in ISOM which they can model the plant in some subset $D_{\mathcal{D}}^m$ of the plant behavioral space $D_{\mathcal{D}}$. If the behavior of the plant changes to a new situation that does not belong to $D_{\mathcal{D}}^m$, the present models cannot properly represent the plant. Training ISOM for this situation may change the network such that it cannot anymore model the plant for some subset of the original $D_{\mathcal{D}}^m$ with desired accuracy. To avoid it, it is better to add a new neuron for the new behavioral situation of the plant. So, a new model is added to the MMISOM if none of the present models are suitable for the plant

and training the present network decrease its performance to properly model the plant:

Condition 1: If the cost of all of the models are more than the threshold a , i.e.:

$$\min_i J^i > a, \quad (17)$$

a new model is added to the MMISOM.

Each model represents some subset of the plant behavior. If the plant is in a situation which does not represent by the models, the only way to get a suitable model for that, is online adaptive identification.

Fig. 4. New node O_* is added to the ISOM. It connects to the nearest neuron O_j . L_{j*} is the smallest one among all links from O_* .

Therefore, IM which is an online adaptive identified mode is the best choice for the new local model. It is also reasonable to add the new model when IM is suitable based on the training condition of Sec. 3.3:

Condition 2: The new model is added to the bank of models if the **Training condition** (of Sec. 3.3) is satisfied.

This condition means, if the cost of IM becomes more than c its parameters are not used as the parameters of the new model. They are used when the cost returns to b . If this condition is not met,

addition of the new model is postponed. If none of the models in ISOM are suitable for the plant, the costs of every of them increase and stay high. But the cost of the IM decreases after initial increase. Figure 5 can be considered as a typical cost function of IM and BM. It will be stated more later during explanation of Example (1). Every 1000 steps the parameters of the plant are changed and as it is depicted in Fig. 5 the cost of all the models are increased rapidly. IM does not initially appropriate to model the new situation, and therefore, its performance index is initially increased. Later, as it adapts to the plant, its performance index is decreased. So, Conditions 1 and 2 properly add a new model to ISOM.

On the other hand, suppose that the plant is in the domain of model i . Since other models are not suitable for the plant, their outputs are different from the plant one. Now suppose that the plant's parameters change at time t_o , such that it is now in the domain of model j . The output of this model (*i.e.* y^j) is initially different from the plant one but converges to y after some steps. Due to this condition and the existence of integral of error in the performance index J^i . Therefore, it decreases after an initial transition. The length of this transition depends on the natural frequency of the plant at that stage, and also to the related value of both the integral coefficient γ and the forgetting factor α to the error coefficient β in J . J^j is initially high. Since model i is not any more valid after t_o , it is likely that Condition 1 is met for a while when J^i increases to a value which is larger than a and before J^j decreases to a value below it. Condition 2 will also inevitably be satisfied after some steps. Therefore, Conditions

Fig. 5. Cost of IM and BM in Example (1). It is clear that after training the best model introduced by the bank of model is better than the adaptive model during the transient time after every abrupt change of the plant parameters.

1 and 2 cause addition of one new node, although there was the suitable model j for the plant.

To overcome the above problem, let's have a close look to what has happened to the costs of i and j local models and IM. IM and the model i have been suitable for the plant before the plant changes. After that, J^o (the cost for IM) increases. But it decrease later with a transition time that depends on the plant natural frequency, the adaptation gain K in Eq. (14) and the coefficients α, β and γ . So, K also increases the delay in reduction of J^o . But K does not affect J^j . As a result, J^o decreases slower than (see Fig. 6, column 3). J^j decreases to a value less than b before J^o does if b is properly selected. This could not happen if there was not any suitable model in the bank of models (see Fig. 6, column 1). Therefore:

Condition 3: After Condition 1 is satisfied; Condition 2 adds one node to the MMISOM if still

$$\min J^i > b. \quad (18)$$

The values of a , b and c affect the possibility of adding a node and consequently affect the structure of ISOM. Increasing the value of a means acceptance of more error in the plant estimation. On the other hand, small a means more accuracy in the expense of more models. The MMISOM assigns models to subsets of the plant linear model space. After selection of the best model, it can be adapted for more accuracy. If b and c are small, the respected node of ISOM are trained by IM for a small subspace around it. Hence, the model is fitted to that small subspace. It means that more models are needed to appropriately model the plant. On the other hand, with large b and c the network is trained in the situations that IM is not accurate. Therefore, *determination of b and c* is a trade off between over fitness to the plant and accuracy of data used for training the network. The effect of noise is also changed by a , b and c . This effect is studied on the Example (2) of the next sub-section.

3.6. Illustrative examples

Two examples are presented in the following to illustrate learning in the MMISOM. In the first example, the plant is noise free. The effect of noise is studied in the second one. Some of the properties of the algorithm are discussed through these two examples.

Example (1). The MMISOM algorithm is used to model a linear time-variant plant

$$y(t) = -a_1y(t-1) - a_2y(t-2) + b_0u(t) + b_1u(t-1), \quad (19)$$

where u is a random signal with uniform distribution

$$u \sim U(-1, 1) \quad (20)$$

Four parameters of the plant (19) get any of the 5 following sets

$$\begin{aligned} (a_1, a_2, b_1, b_0) \in \text{prmtrset} \\ = ((-1.1, 0.24, 0.05, 0.005), \\ (-0.9, 0.14, 0.65, 0.455), \\ (-0.7, 0.06, 0.45, 0.0225), \\ (-1.3, 0.36, 0.25, 0.075), \\ (-1.48, 0.49, 0.85, 0.765)). \end{aligned}$$

The parameters remaining constant in each situation for 1000 steps. After that, another set of parameters out of $D_{\mathcal{P}}$ is randomly selected. The problem is simulated for 60,000 steps. Selections of the plant parameters out of $D_{\mathcal{P}}$ are repeated after every 20,000 steps to illustrate improvement in the identification of plant. For example, plant parameters in the period [1001, 2000] is the same as the parameters in the periods [21001, 22000] and [41001, 42000]. Therefore, it is possible to find out whether any problem, which was occurred in the earlier steps of simulation, has been removed later. 20 consequent selected sets of parameters have indices:

$$\begin{aligned} \text{prmtrserial} = (5, 3, 4, 2, 5, 4, 1, 3, 4, 1, 3, \\ 4, 1, 4, 2, 5, 3, 5, 4, 3) \end{aligned} \quad (21)$$

IM is computed by RLSE adaptive identification of Sec. 3.2 with forgetting factor $\lambda = 0.98$.

Plant parameters are shown in Fig. 6 together with adaptive IM. Convergence speed of the IM parameters depends on λ and the natural frequencies of the plant, as mentioned in Sec. 3.5. Poles of the plants of $D_{\mathcal{P}}$ are:

$$\begin{aligned} \text{poleset} = ((-0.3, -0.8), (-0.2, -0.7), (-0.1, -0.6), \\ (-0.4, -0.9), (-0.5, -0.98)) \end{aligned} \quad (22)$$

On the average, there are around 100 to 500 steps delay in identification of IM parameters, depending on the natural frequencies in *poleset*. This delay remain the same and does not reduce even after many iterations of identification as seen in Fig. 6.

Fig. 6. Parameters of the plant, IM and the best model in some portions of time in Example (1). In the early stages of time when ISOM has not yet trained sufficiently the best model of the bank of model is worse than the adaptive model. But after enough training it introduce a better model than the adaptive model during the transient time.

The MMISOM is started with a single model. Later, when the plant parameters change, it increases the number of nodes to ISOM network using the algorithm of Sec. 3.5. Since the training is on-line, the learning factor of the network is set to $\eta = 0.01$.

All through the simulation, neighborhood function does not change, since the training is an on-line one. The neighborhood function $\mathcal{H}$ is 1 for the winner node, 0.5 for one step neighborhood (those nodes which are directly connected to the winner node) and 0 elsewhere. The neighborhoods in network is checked every 30 steps to keep ISOM an MST graph. The performance index of Eq. (16) with

$$(\alpha, \beta, \gamma) = (0.95, 1, 0.2). \quad (23)$$

is used to measure the cost function of models. The value of a , b and c are empirically obtain as:

$$(a, b, c) = (1, 0.2, 0.2). \quad (24)$$

Figure 6 illustrates the parameters of BM in addition to the plant and IM ones. In early stages of modeling, when the ISOM has not been sufficiently trained, the best model of the network at that stage can not model the plant. Therefore a new model (the same as IM) is added to bank of models. This is the time when cost of BM jumps to that of IM in Fig. 5. But gradually, the best model approaches to the plant faster than IM, as is depicted by comparing the columns of Fig. 6. Parameters change of the plant are the same for all 3 columns. IM also convergence more or less the same in every columns. But there is a significant improvement in BM convergence from left to right columns. Figure 7 depicts the structure of the neural network after 60,000 steps training. The star points stands for the 5 sets of plant parameters $D_{\mathcal{P}}$. The nodes and links depict the ISOM network. There are some nodes between the plant parameters sets which stabilize the modeling as explained below.

Convergence of BM after switch of step 26,001 is not satisfactory and therefore a new model is added after that. Let's follow it to understand what happens in the MMISOM. $D_{\mathcal{P}}(1)$ is selected at this step as the parameters of the plant. Although the same set was previously selected during the periods [6001, 7000], [9001, 10000] and [12001, 13000] (which is clear from *prmrtrserial* of Eq. (21)), but there is not any suitable model for it at step 26001, which is in contrary to what is expected. Node O_1 was suitable for $D_{\mathcal{P}}(1)$ at the periods started at steps 9001

and 12001. Later, during period [22001, 23000] that the plant parameters are $D_{\mathcal{P}}(4)$, node O_7 was the winner node for a while, see Fig. 8b. This node and the node O_1 are neighbor at that time. Therefore, O_1 changes toward $D_{\mathcal{P}}(4)$. When the plant again chooses $D_{\mathcal{P}}(1)$ as its parameters on step 26001, no any suitable node exists for it, Fig. 10. Although two sets $D_{\mathcal{P}}(1)$ and $D_{\mathcal{P}}(4)$ are far from each other, training the network for one $D_{\mathcal{P}}$ changes the best node in the other one. This means that there were not enough nodes in ISOM to cover the space between these two sets. The algorithm has fixed this problem by adding node O_{12} at step 26215. Later, when the plant experiences the same situation on step 42001 as at step 22001, there is O_{12} between O_1 and O_7 . This time O_{12} is changed toward $D_{\mathcal{P}}(4)$ instead of O_1 . Hence, there is the suitable node O_1 for $D_{\mathcal{P}}(1)$ when the change at step 46001 occurs. This node is introduced by the MMISOM as the best one in just 10 steps, *i.e.* at step 46011, see Fig. 8a, much faster than adaptation of IM model which occurs at step 46215.

Figure 5 illustrates the cost function of IM and BM. The cost of BM is higher than IM during the transient time of the early changes in the plant parameters. But it improves as the network learns the plant parameters sets. The cost function of BM is lower than that of IM and reduces faster than that during the transient time after all of the changes from step 15001, except the change at step 26001.

The effect of Conditions 1 to 3 in Sec. 3.5, can be illustrated by studying addition of node O_{12} . At step 26000, just before change of the parameters of the plant, the node O_9 is the BM, as is shown in Fig. 8a. The cost of O_9 is 0.0003 at this step and it is also low for its two neighbors O_7 and O_{10} , see Fig. 11. This node is not any more suitable for the plant after the step 26001 and its cost function starts to increase. At step 26002, training of ISOM stops because the cost function of IM is higher than $c = 0.2$. O_7 is winner node from step 26001 to 26006. If the training was continued, the node O_7 and its two neighbors at these steps, that is O_1 and O_9 , would converge to invalid values of IM. It would also happen to node O_{10} , O_9 and O_7 , when node O_{10} was the winner node after step 26007, see Fig. 8b. As IM adapts to the plant, its cost function decreases. But cost of O_1 , which is currently the best model for the plant, cannot decrease because it is not that suitable for the plant. At step 26009, the minimum value for the

Fig. 7. The structure of SOM in Example (1) in the step 60,000. There are some nodes around the plant parameters locations. The intermediate nodes 4, 5, 9 are produced by the algorithm to stabilize the main nodes around the plants parameters.

cost functions among local models is 1.1655, which is more than $a = 1$. Therefore Condition 1 for adding a new neuron is satisfied. Simultaneously, cost function of IM is also increased as a result of mismatch between its parameters and the plant's one. When cost function of IM reduces to lower than $b = 0.2$ at step 26215, cost of node O_1 is 0.5973, which is more than b . Therefore, Conditions 2 and 3 are satisfied and the new O_{12} is added to the ISOM. Notice that, IM converge the plant on 215 steps after changes at every steps 6000, 26000, 46000, which is very long in compare with the selection of O_{12} on 10 steps after the change at step 46000.

As mentioned above, in the MMISOM the neighboring neurons of the winner one also learn the situation, but with smaller learning factor than the winner neuron. This may damage current fine models. However, this is a specification of the MMISOM which is essential for online learning of the local models. To emphasize on the merit of this specification let's consider the cases where learning is just applied to the winner model. Suppose that plant behavioral space and 30 current local models are as shown in Fig. 12. Moreover, suppose that no new node adds to the network. Using the above consideration, every time the plant is somewhere in left-down subspace, there is

Fig. 8. Selected best model and the winner node in Example (1). Best model selects according to Eq. (16) and used as the model of the plant. Winner node is selected according to (2) and has the closer parameters to the IM model. During the transient period some of the intermediate nodes become the best node due to the effect of the plant states before and after the changes in the plant parameters.

only one neuron which is the best, and therefore this neuron will adapt for fine modeling. But if the MMI-SOM is used, the local models neighboring to the initial single model in the left-down subspace, which are currently in the right-top subspace, are tuned for (pulled toward) the left-down subspace. This will end up to some kind of distribution of the local models as shown in Fig. 13.

Figure 14 shows the output response of the plant to a normalized random Gaussian input during the last 20000 steps of simulation. It illustrates that the out of the plant is rader changeable when its parameters change. Figure 15 shows the prediction error of the adaptive model and the multiple model. When the parameter of the plant changes the prediction errors increase since the current models (both adaptive and multiple ones) cannot interpret the plant new behavior. But, the output of the multiple model approaches the plant one much faster

than the output of the adaptive model. Table 1 compares two models. The output of the multiple model approaches the output of the plant nearly 3 times faster than the output of the adaptive one. To compute the average settling time, an error treshold of 0.01 is used. It has better performance using both the integral of absolute error and mean square error criteria.

The above result of output error may be discussed from a different point of view. Let's compare the result of 4 different modeling methods; (1) Single Adaptive Modeling (SAM), (2) All Fixed Model (AFM),⁵² (3) Fixed Model and One Adaptive Model (FMOAM)⁵² and (4) MMISOM. The bank of local multiple models of step 26000, where there are 11 local models are considered for all multiple models. These models can correctly model the plant when its parameters are among $D_{\mathcal{D}}(2, 3, 4, 5)$, but neither can model the plant when its parameters are

Fig. 9. Selected best model and the winner node during the transient time after change at steps 26000 and 46000 in Example (1).

$D_{\mathcal{P}}(1)$. At this step, the structure of AFM, FMOAM and MMISOM are set to be the same. Figures 16 and 17 respectively illustrate the output and error of the above 4 different types of modeling methods for the plant (19) for around 200 steps after step 26000. At step 26216 a new local model is added to MMISOM. The response of this new MMISOM (called MMIOSM(46000)) for around 200 steps after step 46000 is also illustrated in these figures. SAM is an adaptive model. Therefore, after a few hundred steps its output approaches the plant output. AFM has not the capability to follow the new situation. Therefore, its output error remains high as far as the plant parameters is $D_{\mathcal{P}}(1)$. FMOAM and MMISOM has an embedded adaptive model, the same as SAM. Therefore, after around 125 steps when their embedded adaptive model become better than all other 11 local models (according to performance index (16)), their performance starts to improve. The performance of the best local model is better than the adaptive one up to step 100. From step 100 to step 125 either of them alternatively become better than the other one. However, MMISOM has the

ability to add new local model(s) to its bank of model for this new situation. This cause a big difference between MMISOM and other 3 techniques in step 46000. At this step, SAM, AFM and FMOAM has the same structure and parameters as they have in step 26000. Therefore their outputs and output errors are as shown in Figs. 16 and 17 after step 26000. But MMISOM had learned $D_{\mathcal{P}}(1)$ by adding a new model. As a result, its output error decreases much faster than the other ones.

Two other cases are also discussed in Ref. 52; (1) All Adaptive Model (AAL) and (2) Fixed Model with one Free-Running and one Reinitialized Adaptive Model (FMFRRAM). All of the models are adaptive in AAL. This kind of modeling is suitable for the plants that do not remain in any situation for a long time. Otherwise, all of the local models will be the same due to adaptation of them. In the case that the plant enters to a new situation which none of the local adaptive models are good, one of them which are closer to the plant right after the change of the plant parameters will become the best one and adapt to the plant more

Fig. 10. The structure of SOM in Example (1) in the step 26,000. Node 12 will be added in step 26215 as an intermediate between nodes 1 and 7 to stabilize the network. Node 9 is connected to nodes 7 and 10 and node 7 is connected to node 1 in addition to node 9.

rapidly. Therefore, the output of this model will be nearly the same as the output of FMOAM in Fig. 16.

In FMFRRAM, the reinitialized adaptive model reinitialized to the new best model after any change in the plant parameters. However, if there are not any good model for the plant (as it happened in steps after 26000), this re-initialization cannot improve the performance of modeling. It would be the Free-Running model which should adapt to the new situation. That is, its response again would be as FMOAM.

Above discussion shows that the performance of MMISOM is superior to all the structures discussed

in Ref. 52, when the plant enters a new situation. This is because:

- (1) MMISOM has the ability to learn new behavior.
- (2) MMISOM has an embedded adaptive modeling.
- (3) If the training of ISOM neural network of MMISOM stops, it will be as AFM.
- (4) Local models in MMISOM are reference vectors of an ISOM neural network. RV of the winner neuron (the best model) and some of its neighbor adapt to the new data. This gives the ability of fine tuning, the same as FMFRRAM.
- (5) Although all of the models in MMISOM have the ability of adaptation, but since just the winner

Fig. 11. Performance of the IM and models of SOM in some steps ends to the addition of the node O_{12} . It shows that in step 26215 when the performance of the adaptive instantaneous model is lower than the threshold, none of the models in the bank of model are below the threshold, therefore a new model (model 12) must be added.

Fig. 12. Two behavioral subspace of a plant and initial local models for it before the training starts. Black stars indicate the local models and grey dots indicate the input data.

and its neighbors are trained (generally one-step neighbors), even if the plant remains for a long time in a situation, the parameters of other models will not change that much.

Example (2). In the previous example the plant was noise free. In this example, we are going to

Fig. 13. Final local models (typical) using one step neighborhood. Black stars indicate the local models and grey dots indicate the input data. Without neighborhood definition of ISOM the left bottom subspace could become modeled just by the original single model shown in Fig. 12. Defining a one step neighborhood, some of the models which initially were in the right-top subspace, tend to the left-bottom subspace.

Fig. 14. Output response of the plant to a normalized random Gaussian input. It shows that the response of the plant when its parameters change is very different.

study effects of noise on the MMISOM. Consider the plant:

$$y(t) = -a_1 y(t-1) - a_2 y(t-2) + b_0 u(t) + b_1 u(t-1) + \omega(t), \quad (25)$$

Fig. 15. Output prediction error of the adaptive model and the multiple model. Output of the multiple model approaches the plant output much faster than the adaptive one. Decreasing the transient time with small error is the main contribution of the multiple model approach.

Table 1. Comparison of the performance of the multiple model and adaptive model in prediction of the output of the plant in Example (1).

	$\int e dt$	$\int e^2 dt$	Average settling time (steps)
Adaptive	0.1321	0.2943	262
Multiple	0.0648	0.2329	89

where ω is zero mean Gaussian noise with standard deviation $\sigma_\omega = 0.1$. On one hand, addition of this noise to the plant reduces accuracy of the estimation parameter of IM and, on the other hand, increases output error for all of the local models. Therefore, the average costs of all the models, including BM one, are increased compared to their noise free counterpart models. As a result, the threshold levels should be increased. Empirically, they are set to:

$$(a, b, c) = (1, 0.5, 0.5). \quad (26)$$

The MMISOM algorithm is applied to the plant Eq. (25) with *prmrserial* as following for 80,000 steps:

$$\begin{aligned} \text{prmrserial} = & (1, 2, 3, 4, 5, 1, 3, 5, 2, 4, 1, 4, \\ & 2, 5, 3, 1, 5, 4, 3, 2). \end{aligned} \quad (27)$$

In this case, two problems arise regarding to the identification of local models and selection of BM.

The first problem is that θ^o perturbs around its nominal value, instead of converging to it. Learning these IM's means that there are some models with parameters around θ_φ , see Fig. 19. The cost of all of them, as shown in Fig. 20, are small when the plant parameters are θ_φ . Noise cause either of them to be selected as BM. This is the reason for existence of too many switches between two or more models in each period in Fig. 21. These switches do not pose problem for identification because the models have close parameters and performance. The perturbation may also reduce if hysteresis is defined for the switches.

The second problem occurs in the periods that $D_\varphi(1)$ is selected. In this time, there is a significant perturbation on estimation parameters of a_1 and a_2 . This happens because b_0 is 0.05 and b_1 is close to zero, approximately. Therefore, the steady state gain of the plant is low. In this situation, level of the plant output is low and the impact of noise at output is high. Table 2 depicts the value of $S/N = \sum |y| / \sum |\omega|$ and $\text{var } y / \text{var } \omega$ in some periods when each set of D_φ is select where *var* denotes variance. S/N for $D_\varphi(1)$ is approximately 1/7 of $D_\varphi(3)$ and 1/143 of $D_\varphi(5)$ ones, which are respectively the next lowest and the highest ones. Small b_0 and b_1 results weak IM identification. There are 5 models $\theta^2, \theta^7, \theta^{10}, \theta^{11}, \theta^{12}$ with low b_0 and b_1 but different a_1 and a_2 in Fig. 19. These models have small outputs and their cost functions are close to each other when $D_\varphi(1)$ is selected. Therefore, noise determine BM among them.

As explained above, adding of noise causes some fluctuation and rapid switches on model selection. But, multiple modeling is still superior to adaptive one in prediction of the plant output. Table 3 shows the performance of the adaptive and multiple models in output prediction during the last 20000 steps of simulation, when a normalized random Gaussian input is applied to the plant. Average settling time is around 2 times shorter when multiple modeling is used for prediction. Due to the standard deviation of the noise, the average settling time is computed using a treshold of 0.1 on error. The average prediction error is around 25% lower and the mean square error is around 10% lower than the adaptive one.

The number of models in this example is 20 which is comparable to 24 models for the same plants structure but with 3 sets of parameters for it in Narendra *et al.*⁵² The main reason to have fewer model is the possibility of clustering IM's into some fewer local models by ISOM.

Fig. 16. Output of plant, MMISOM, AFM, FMOAM⁵² and single adaptive model (SAM) of Example (1). MMISOM (46000) is for period [45990, 46200].

Fig. 17. Output error of different models of the plant of Example (1). MMISOM model is compared with AFM, FMOAM⁵² and single adaptive model (SAM). MMISOM (46000) is for period [45990, 46200].

Fig. 18. Parameters of the plant, IM and BM in the last 20,000 steps of Example (2). As explained in the text, chattering of parameters a_1 and a_2 in some situations is due to the low signal to noise ratio.

(a)

(b)

Fig. 19. Structure of SOM in the step 80,000 of Example (2). Since the parameters b_0 and b_1 are around zero, models 2, 7, 11, 12 and 10 has nearly the same output. Level of noise affects selection among them.

Fig. 20. Cost of the instantaneous (IM) and the best (BM) model in Example (2). Noise increases the cost level compare to Example (1), Fig. 5. However, the algorithm still can introduce the best model faster than the adaptive method.

Fig. 21. Index of the best model selected in Example (2). The selected best model changes rapidly. But it switches to some neighbor model by means of ISOM neighborhood which are close-by models in terms of parameters. Indices of close-by models may be far from each other.

Table 2. Relation between the output of the plant and the noise for various plant parameters set. When the signal to noise ratio is low, the effect of noise on best model selection increases.

	$D_{\mathcal{P}}(1)$	$D_{\mathcal{P}}(2)$	$D_{\mathcal{P}}(3)$	$D_{\mathcal{P}}(4)$	$D_{\mathcal{P}}(5)$
Step time	[10200, 11000]	[12200, 13000]	[14200, 15000]	[11200, 12000]	[13200, 14000]
$\Sigma y /\Sigma w $	7	99	50	63	1000
$\text{var } y/\text{var } \omega$	43	9000	3400	3830	10^6

Table 3. Comparison of the performance of the multiple model and adaptive model in prediction of the output of the plant in Example (2).

	$\int e dt$	$\int e^2dt$	Average settling time (steps)
Adaptive	0.2136	0.9546	182
Multiple	0.1645	0.8698	98

4. Conclusion

In this paper, multiple modeling by irregular self-organizing map neural networks (MMISOM) has been presented, which is a method to model a plant with multiple models using the irregular self-organizing map neural network. At every instants, instantaneous models are obtained by an adaptive identification method. These models are clustered by ISOM to make a bank of local models which are the reference vectors of ISOM. ISOM has the same number of inputs as the number of the parameters of the model and has the same number of output nodes equal to the number of the required local models. Reference vectors of ISOM are adapted to various situations of the plant. When the plant is in a specific situation, the best model for the plant is selected

from bank of models. As a result, after adequate training, identification of the plant model is faster specially when the plant behavior changes abruptly.

A major concern in multiple modeling is the suitable number of models. In the MMISOM, it is not necessary to consider the number of models at the beginning of the algorithm. It may start with only one or few nodes. A model is added when it is required, using the algorithm presented in this paper. The cost function of modeling is used to determine necessity to add a new model. Parameters of the algorithm should properly be assigned to avoid creating large number of models but at the same time achievement of accuracy.

The MMISOM has improvement on identification of time-varying plants comparing with conventional single-model adaptive identification. Major improvement is in the transient period, where the single-model adaptive identifier adapts to new situation of the plant by some delay. Simulation results indicate that the best model on the MMISOM is introduced for the plant much faster than the adaptive IM, after the neural network has been sufficiently trained.

Training of the neural network can be on-line or off-line. The algorithm can introduce the model of the plant in the situations that they have been

experienced for a few times. Therefore, none of the models are suitable at the early steps before sufficient training. Fortunately, the algorithm can take advantage of the adaptive IM identifier, as the best model during that learning period. Hence, the quality of identification is at least as good as the single-model adaptive IM one but improves during the time.

Although the algorithm introduces the best model for the plant faster than the IM identifier, but the quality of the local models and consequently the selected best model are directly depend on the quality of IM identification.

References

1. L. Ljung, *Theory and Practice of Recursive Identification* (Cambridge, MA: MIT Press, 1983).
2. R. Murray-Smith and T.A. Johansen (eds.), *Multiple Model Approaches to Modeling and Control* (Taylor & Francis: London, 1997).
3. D. T. Magill, Optimal adaptive estimation of sampled stochastic process, *IEEE Transactions on Automatic Control* **AC-10**(4) (1965) 434–439.
4. F. L. Sims and D. G. Lainiotis, Recursive algorithm for the calculation of the adaptive Kalman filter weighting coefficients, *IEEE Transaction on Automatic Control* **AC-14** (1969) 215–217.
5. G. A. Ackerson and K. S. Fu, On state estimation in switching environments, *IEEE Transaction on Automatic Control* **AC-15**(1) (1970) 10–17.
6. N. H. Gholson and R. L. Moose, Maneuvering target tracking using adaptive state estimation, *IEEE Transaction on Aerospace and Electronic Systems* **AES-13**(3) (1977) 310–317.
7. M. Gaivrit, Bayesian adaptive filter for tracking with measurements of uncertain origin, *Automatica* **20**(2) (1984) 217–224.
8. D. G. Lainiotis, Partitioning: A unifying framework for adaptive systems, I: Estimation, *Proceedings of the IEEE* **64**(8) (1976) 1126–1143.
9. T. Takagi and M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, *IEEE Transaction on Systems, Man, and Cybernetics* **15**(1) (1985) 116–132.
10. N. Li, Sh. Li and Y. Xi, Multi-model predictive control based on the Takagi–Sugeno fuzzy models: A case study, *Information Sciences* **165** (2004) 247–263.
11. O. Nelles, Local linear model tree for on-line identification of time-variant nonlinear dynamic systems, *International Conference on Artificial Neural Networks* (Bochum, Germany) (1996) 115–120.
12. D. G. Lainiotis, Partitioning: A unifying framework for adaptive systems, II: Control, *Proceedings of the IEEE* **64**(8) (1976) 1182–1198.
13. M. Athans, D. A. Castanon, K.-P. Dunn, C. S. Greene, W. H. Lee, N. R. Sandell and A. S. Willsky, The stochastic control of the F-8C aircraft using a multiple model adaptive control (MMAC) method — part 1: equilibrium flight, *IEEE Transaction on Automatic Control* **AC-22**(5) (1977) 768–780.
14. W. G. He, H. Kaufman and R. Roy, Multiple model adaptive control procedure for blood pressure control, *IEEE Transaction on Biomedical Engineering* **BME-33**(1) (1986) 10–19.
15. G. J. Schiller and P. S. Maybeck, Control of a large space structure using MMAE/MMAC techniques, *IEEE Transactions on Aerospace and Electronic Systems* **33**(4) (1997) 1122–1131.
16. H. Gollee and K. J. Hunt, Nonlinear modeling and control of electrically stimulated muscle: A local model network approach, *International Journal of Control* **68**(6) (1997) 1259–1288.
17. S. Townsend, G. Lightbody, M. D. Brown and G. W. Irwin, Nonlinear dynamic matrix control using local models, *Transaction of the Institute of Measurement and Control* **20**(1) (1998) 47–56.
18. G. T. Tan and M. S. Chiu, A multiple-model approach to decentralized internal model control design, *Chemical Engineering Science* **56** (2001) 6651–6660.
19. D. M. Wolpert and M. Kawato, Multiple paired forward and inverse models for motor control, *Neural Networks* **11** (1998) 1317–1329.
20. R. Middleton, G. Goodwin, D. Hill and D. Mayne, Design issue in adaptive control, *IEEE Transactions on Automatic Control* **33**(1) (1988) 50–58.
21. V. Petridis and A. Kehagias, A multi-model algorithm for parameter estimation of time-varying nonlinear systems, *Automatica* **34**(4) (1998) 469–475.
22. B. Martensson, Adaptive stabilization, PhD dissertation, Lund Institute of Technology, Lund, Sweden (1986).
23. F. M. Pait and A. S. Morse, A cyclic switching strategy for parameter-adaptive control, *IEEE Transaction on Automatic Control* **39**(6) (1994) 1172–1183.
24. K. S. Narendra and J. Balakrishnan, Improving transient response of adaptive control systems using multiple models and switching, *IEEE Transaction on Automatic Control* **39**(9) (1994) 1861–1866.
25. K. Narendra, J. Balakrishnan and M. K. Ciliz, Adaptation and learning using multiple models, switching, and tuning, *IEEE Control Systems Magazine* **15**(3) (1995) 37–51.
26. A. Banerjee, Y. Arkun, B. A. Ogunnaike and R. K. Pearson, Estimation of nonlinear systems using linear multiple models, *AIChE Journal* **43**(5) (1997) 1204–1226.
27. E. Daeipour and Y. Bar-Shalom, IMM tracking of maneuvering targets in the presence of glint, *IEEE Transaction on Aerospace and Electronic Systems* **34**(3) (1998) 996–1003.

28. D. W. Kyger and P. S. Maybeck, Reducing lag in virtual displays using multiple model adaptive estimation, *IEEE Transactions on Aerospace and Electronic Systems* **34**(4) (1998) 1237–1248.
29. V. Nazarzehi, A. Fatehi and M. Zamanian, Control of a laboratory helicopter by multiple modeling adaptive control, *Proceedings of the Iranian Conference on Electrical Engineering*, Zanzan, Iran (in Persian).
30. H. Azimian, A. Fatehi and B. N. Arabi, Closed-Loop Global Fuzzy TSK Modeling: A Case Study, *Proc. 15th Mediterranean Conf. on Control and Automation*, Athens, Greece, 2007.
31. R. H. Nystrom, K. V. Sandström, T. K. Gustafsson and H. T. Toivonen, Multimodel robust control of nonlinear plants: A case study, *Journal of Process Control* **9** (1999) 135–150.
32. R. N. Silva, P. O. Shirley, J. M. Lemos and A. C. Goncalves, Adaptive regulation of superheated steam temperature: A case study in an industrial boiler, *Control Engineering Practice* **8** (2000) 1405–1415.
33. A. Fatehi and K. Abe, Multilinear modeling of nonlinear continuously variable transmission (CVT) gear-box plant by SOM neural networks, *Proceedings of the International Workshop on Soft Computing in Industry (IWSCI'99)*, Muroran, Japan (1999), 31–35.
34. J. C. Nascimento and J. S. Marques, Improving the robustness of parametric shape tracking with switched multiple models, *Pattern Recognition* **35** (2002) 2711–2718.
35. F. Soufivand, A. Fatehi and M. Zamanian, Weather forecasting by multiple modeling, *Proceedings of the Iranian Conference on Electrical Engineering*, Zanzan, Iran (2005).
36. S. J. Han and S. B. Cho, Detecting intrusion with rule-based integration of multiple models, *Computers & Security* **22**(7) (2003) 613–623.
37. J. A. Rodriguez, J. A. Romagnolia and G. C. Goodwin, Supervisory multiple regime control, *Journal of Process Control* **13** (2003) 177–191.
38. D. Alejo, P. Maussion and J. Faucher, Multiple model control of a Buck dc/dc converter, *Mathematics and Computers in Simulation* **63** (2003) 249–260.
39. T. Kohonen, *Self-Organizing Maps* (Springer-Verlag: London, 1997).
40. A. Ultsch, Self organized feature maps for monitoring and knowledge acquisition of a chemical process, *Proceedings of International Conference on Artificial Neural Networks* (Amsterdam, 1993), 864–867.
41. J. C. Principe, L. Wang and M. Motter, Local dynamic modeling with self-organizing maps and applications to nonlinear system identification and control, *Proceedings of the IEEE* **86**(11) (1998) 2240–2258.
42. G. A. Barreto and A. F. Araujo, Nonlinear modeling of dynamic systems with the self-organizing map, *Proceedings of International Conference on Artificial Neural Networks*, (2002) 975–980.
43. A. Fatehi and K. Abe, Plant identification by SOM neural networks, *Proceedings of the European Control Conference (ECC'99)* (Karlsruhe, Germany, 1999), F190.
44. A. Fatehi, Multiple modeling of time-varying systems by self-organizing map neural network, PhD Dissertation, Tohoku University, Sendai, Japan (2001).
45. A. Fatehi and K. Abe, Statistical properties of multiple modeling by self-organizing map (MMSOM), to be published in *The Mediterranean Journal of Measurement and Control* **3**(4) (2007) 151–156.
46. J. Kangas, T. Kohonen and J. T. Laaksonen, Variants of self-organizing maps, *IEEE Transactions on Neural Networks* **1**(1) (1990) 93–99.
47. A. Fatehi and K. Abe, Self-organizing map neural network as a multiple model identifier for time-varying systems, *Sixth International Symposium on Artificial Life and Robotics (AROB 6th'01)* (Tokyo, Japan) (2001) 528–531.
48. B. Fritzke, A growing neural gas network learns topologies, in *Advances in Neural Information Processing Systems 7*, G. Teasuro, D. S. Touretzky and T. K. Leen (eds.) (Cambridge MA: MIT Press) (1995) 625–632.
49. H. U. Bauer and Th. Villmann, Growing a hypercubical output space in a self-organizing feature map, *IEEE Transactions on Neural Networks* **8**(2) (1995) 218–226.
50. J. Bruske and G. Sommer, Dynamic cell structure learns perfectly topology preserving map, *Neural Computation* **7**(4) (1995) 846–865.
51. R. Sedgewich, *Algorithms*, 2nd ed. (Reading, MA: Addison-Wesley) (1988) 451–469.
52. K. Narendra and J. Balakrishnan, Adaptive control using multiple models, *IEEE Transactions on Automatic Control* **42**(2) (1997) 171–187.