Modeling and Inverse Controller Design for an Unmanned Aerial Vehicle Based on the Self-Organizing Map

Jeongho Cho*, Jose C. Principe, *Fellow*, *IEEE*, Deniz Erdogmus, *Member*, *IEEE*, and Mark A. Motter, *Senior Member*, *IEEE*

***Abstract*— The next generation of aircraft will have dynamics that vary considerably over the operating regime. A single controller will have difficulty to meet the design specifications. In this paper, a SOM-based local linear modeling scheme of an unmanned aerial vehicle (UAV) is developed to design a set of inverse controllers. The SOM selects the operating regime depending only on the embedded output space information and avoids normalization of the input data. Each local linear model is associated with a linear controller, which is easy to design. Switching of the controllers is done synchronously with the active local linear model that tracks the different operating conditions. The proposed multiple modeling and control strategy has been successfully tested in a simulator that models the LoFLYTE® UAV.**

***Index Terms*—Multiple model, Inverse Controller, Self-Organizing Map.**

I. Introduction

The identification of unknown nonlinear dynamical systems has received considerable attention in recent years since it is an indispensable step towards controller design of nonlinear systems. Many systems, especially aircraft, have dynamics that vary considerably over the operating regime, effectively bringing the issue of time varying parameters (or nonlinearity) into the design [22]. Controllers for aircraft have been designed predominantly by classical control techniques [39]. While this tradition has produced many highly reliable and effective control systems, recent years have seen a growing interest in the use of robust, nonlinear adaptive control theory for flight control [3],[22],[36]. For instance, the concept of multiple models with switching, according to a change in dynamics, has been an area of interest in control theory in order to simplify both the modeling and the controller design [13,17,21,33].

System identification methods fall into two broad categories: global and local. Global approximations can be made with various function representations, e.g., polynomials, rational approximation, and multi-layer perceptrons (MLPs) [9]. To approximate a function *f*, a model should be capable of representing its many possible variations. If *f* is complicated, there is no guarantee that any given representation will approximate *f* equally well across all space. The dependence on representation can be reduced using local approximation where the domain of *f* is divided into local regions and a separate model is used for each region [32].

Local modeling is based on nearest-neighbors in the operating space where a simple model is constructed using only the neighboring samples. The rationale behind this approach is basically that it is easier to develop local models (or controllers) because the dynamics are simpler locally than globally [33]. For instance, if the system phenomena or behavior changes smoothly with the operating point, then a linear model (or controller) will always be sufficiently accurate locally provided that the operating regime is sufficiently small, even though the system may contain complex nonlinearities when viewed globally. In these methods, the global dynamics is approximated by a preset number of local linear models that need to be specified by the user. The added difficulty in local modeling is the switching among models, but recently the approximation properties of multiple models have been examined in detail [13]. Under mild conditions, it has been shown that multiple models can uniformly approximate any system on a compact subset provided a sufficient number of local models are given [7]. Finally, with this approach, the model/controller structure is easy to understand and interpret [33].

There are many examples in the literature where the local modeling paradigm has been successfully applied for the modeling of nonlinear autonomous and nonautonomous systems. Jacobs et al. [28] have proposed the mixtures of expert model that is composed of several different expert networks and a gating network that localizes the experts. They showed that a simple model can be built by dividing a vowel discrimination task into appropriate subtasks. Bottou and Vapnik [29] have proposed to use local learning algorithms

Manuscript received December 3, 2003; revised March 30, 2005. This work was supported in part by Accurate Automation Corporation under grant #463, and NASA grant NAG-1-02068. *Asterisk indicates corresponding author*.

J. Cho* and J.C. Principe are with the department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: jeongho@cnel.ufl.edu).

D. Erdogmus is with the department of Computer Science and Biomedical Engineering, Oregon Health & Science University, Beaverton, OR 97006 USA.

M.A. Motter is with the Dynamics and Control Branch, NASA Langley Research Center, Hampton, VA 23681 USA.

instead of training a complex system by all data samples and demonstrated that a set of subsystems trained with a subset of data can improve the performance for an optical character recognition problem. Murray-Smith [31] similarly has extended RBF networks where each local model is a linear function of the input and exhibited a great success in control problems. In a similar context of mixture of local experts, Tipping and Bishop [ref] have utilized a combination of local linear PCA projections for clustering and density modeling for the application of handwritten digit recognition. On the contrary, Farmer and Sidorowich [32] have shown that local linear models provide an effective and accurate approximation of chaotic dynamical systems activating only one model at any given time as opposed to the mixture of models. The neural-gas architecture proposed by Martinetz et al. [10] is similar to the SOM in that the competitive network divides the input space in a set of smaller regions and then local linear models are created by a LMS like rule. They showed that the neural-gas network outperforms MLPs and RBF networks for time series prediction. The same group [30] used a Self-Organizing Map (SOM) for the control of a robotic arm. Principe and Wang [16] have successfully modeled a chaotic system with a SOM-based local linear modeling method. Vesanto et al. [34] and Moshou [35] proposed a scheme that essentially followed local linear modeling based on SOM topology for nonautonomous system.

In a number of applications of modeling with switching, a SOM has been utilized to divide the operating region into local regions. The SOM is particularly appropriate for multiple switching because it converts complex, nonlinear statistical relationships of high-dimensional data into simple geometric relationships that preserve the topology in the feature space [8]. Thus the role of the SOM is to discover patterns in high dimensional input space and divide the input space into a set of regions represented by the weights of each processing element (PE). However, most of the work on local linear approaches only addresses the recursive state-dependent prediction. The approximation of nonautonomous systems using a finite set of local models, which involves an explicit dependence on external driving inputs, has not been fully studied.

Global controller design with the aid of multiple models has been extensively reported in the literature. Murray-Smith and Johansen [33] utilized an extended RBF network where each local model is a linear function of the input and they reported great success for control problems. Palizban *et al.* [40] attempted to control nonlinear systems with the linear quadratic optimal control technique using multiple linear models. As another blending approach, the multiple paired forward and inverse models was proposed by Wolpert and Kawato [42] to show that the approach produces effective motor control tackling the problems of biological motor learning and control. In contrast, Narendra and Balakrishnan [13] proposed the multiple model approach in the context of adaptive control with switching where local model performance indices have been used to select the local controller. Motter and Principe [11], and Principe *et al.* [17] have successfully modeled the set point regulation of the NASA Langley 16 Foot Transonic wind tunnel during the aerodynamic testing of a model aircraft. But the control of the wind tunnel is bang-zero-bang (on-off), therefore it is still rather easy to cluster the control input time series into a finite (relatively small) number of clusters. In this case, the next value of the state was completely specified by one of three values of control input based on the current state, which does not happen in general. Consequently, a more general representation of the underlying nonautonomous dynamical system is highly desirable.

Inspired by this approach, a local control strategy with switching by extending the SOM-based local modeling scheme for nonautonomous and nonlinear systems is proposed in this paper. Local linear models are derived through competition using the SOM and they are derived from the data samples corresponding to each of the SOM's PEs. At any time instant, the model representing the plant dynamics is chosen by the SOM depending on the history of the plant output. In addition, a set of an inverse controllers corresponding to each linear model is also derived to show the effectiveness of the proposed scheme. Simulation results using the proposed strategy for identification and control of a missile are presented to demonstrate the versatility of the algorithm. Also, the proposed scheme is tested to identify the highly nonlinear aircraft dynamics of the experimental LoFLYTE® UAV and to design the corresponding inverse controller. Results show that the switching linear models are a promising alternative for system identification when compared with a single global model.

II. Nonlinear System Representation by Multiple Models

The idea of multiple modeling is to approximate a nonlinear system with a set of relatively simple local models valid in certain operating regimes. Because of the complexity, uncertainty and nonlinearity of a large class of systems, we often cannot derive appropriate models from first principles, and are not capable of deriving accurate and complete equations for input-state-output representations of the systems. Hence we need to resort to input-output data in order to derive the unknown nonlinear system model [6]. The technique of multiple model networks is appealing for modeling complex nonlinear systems due to its intrinsic simplicity [33].

We begin with a brief overview of a dynamical systems approach to input-output modeling. When no physical knowledge of the system is available, we have to determine a model from a finite number of measurements of the system's inputs and outputs. An autonomous dynamical system's approach to "black-box" modeling based on Takens Embedding theorem was first suggested by Casdagli [26]. The delay embedding offers the possibility of accessing linear or nonlinear coupling between variables and is a fundamental tool in nonlinear system identification. The use of delay variables in the structure of these dynamical models is similar to that originally studied by Leontaritis and Billings [27], and is common in linear time-series analysis and system identification [23].

A. Autonomous System

When we are trying to understand an irregular sequence of measurements, an immediate question is what kind of process generates such a series. Under the deterministic assumption, irregularity can be autonomously generated by the nonlinearity of the intrinsic dynamics. Let the possible states of a system be represented by points in a finite dimensional phase space, $\Re^P$. This can be realized by a map of $\Re^P$ onto itself:

$$\begin{bmatrix} x(k+1) \\ \vdots \\ x(k-P+2) \end{bmatrix} = f\left(\begin{bmatrix} x(k) \\ \vdots \\ x(k-P+1) \end{bmatrix}\right) \qquad (1)$$

where $x(k)$ is the system state and f is typically referred as the vector field. The predictive mapping is the centerpiece of modeling since once determined, f can be obtained from the predictive mapping $f_i : \Re^P \rightarrow \Re$ as

$$x(k+1) = f_i(\vec{x}(k)) \qquad (2)$$

where $\vec{x}(k) = [x(k)\ x(k-1) \cdots x(k-P+1)]^T$. In addition, Singer et al. [19] derived the locally linear prediction based on this relationship as

$$f_i(\vec{x}) \approx \vec{a}_i^{\,T} \vec{x} + b_i \qquad (3)$$

The vector and scalar quantities of $\vec{a}$ and b are estimated from the selected pairs $(x(j+1), \vec{x}(j))$ in the least square sense, where j is the index of the data samples in the operating regime, i.e. one model. To obtain a stable solution, more than P pairs must be selected. In general, the above local model fitting is composed of two steps: a set of nearby state searches over the signal history and model parameters which, when pieced together, provide a global modeling of the dynamics in state space. The underlying dynamics f is then approximated as

$$f \approx \bigcup_{i=1,\cdots,N} f_i \qquad (4)$$

where N is the number of operating regimes. Based on this approximation of an autonomous system, local linear models have performed very well in comparative studies on time series prediction problems and in most cases have generated more accurate predictions than global methods [18,20,24]. Moreover, the nonlinear dynamical system can be identified by local framework even in the presence of noise if enough data are available to cover all of the state space since local regions are local averages of the data. To make the local network less sensitive to noise and outliers, more than one neighbor can be utilized in local modeling.

B. Nonautonomous System

The temporal state evolution of an autonomous system is functionally dependent only on the system state, but a nonautonomous system, such as considered in this work, allows for an explicit dependence on an independent variable, the control input, in addition to the system state. For an autonomous system, it is reasonable to assume that the future behavior of the system can be predicted over some finite interval from a finite number of observations of past outputs. In contrast, predictions of the behavior of a nonautonomous system require consideration of not only the "internal" deterministic dynamics (past outputs), but also of the "external" driving term (future input) [23,11,5].

Some common classical approaches for nonlinear nonautonomous system modeling are based on polynomials for the realization of the nonlinear mapping. Other methods that have been developed for nonlinear system identification include Volterra Series, neural networks, etc. [33,14]. Normally, a discrete-time nonlinear dynamic system can be described by a NARX (Nonlinear Auto-Regressive with eXogenous input) model that is an extension of the linear ARX model, and represents the system by a nonlinear mapping of past inputs and output terms to future outputs, that is,

$$y(k+1) = f(y(k), \cdots, y(k-d_y), u(k), \cdots, u(k-d_u)) \qquad (5)$$

Here $y(k) \in Y \subset \Re^p$ is the output vector and $u(k) \in U \subset \Re^q$ is the input vector. For simplicity, we will set $p = q = 1$. Let the $(d_y + d_u + 2)$ - dimensional basis vector be

$$\psi(k) = [\psi_y(k), \psi_u(k)] = [y(k), \cdots, y(k-d_y), u(k), \cdots, u(k-d_u)] \qquad (6)$$

where $\psi(k)$ is in the set $\Psi = Y^{d_y+1} \times U^{d_u+1}$. If the nonlinear function $f(\cdot)$ is invertible *w.r.t.* the input $u(k)$, then a controller may be constructed by training an inverse neural network. Unfortunately, most of these nonlinear functions are not invertible, so the application of this approach is limited. Also, when the environment of a system changes abruptly, the original model (and hence the controller) is no longer valid [5]. In order to solve these difficulties, it is appealing to use a methodology that decomposes the overall modeling problem into a set of simpler local modeling problems, each for a different operating regime. In so doing controllers can be designed *a priori* for each of the local models, and can be easily inverted.

In the interest of modeling the local dynamics of a nonautonomous system in each region, the local approximation method presented for autonomous systems can be extended by letting $\vec{x}(k) = \psi(k)$ in (2), so that $y(k+1) = f_i(\psi(k))$. Provided that necessary smoothness conditions on $f_i : \Psi \rightarrow Y$ are satisfied, a Taylor series expansion can be used around the operating point. The first-order approximation about the system's equilibrium point produces N local predictive ARX models $f_1, \cdots, f_N$ of the plant described by

$$f_i(\psi(k)) \approx \sum_{j=0}^{d_y} a_j^i y(k-j) + \sum_{j=0}^{d_u} b_j^i u(k-j),\ i = 1,\cdots,N \qquad (7)$$

where a_j^i and b_j^i are the parameters of the i^{th} model. Although higher order Taylor approximations would improve accuracy, they are not very useful in practice because the number of parameters in the model increases drastically with the expansion order.

Our proposed methodology is summarized in (7): first, the delayed version of input-output joint space is decomposed into a set of operating regions that are assumed to cover the full operating space[1]. Next, for each operating regime we choose a simple linear ARX model to capture the dynamics of the region. Consequently, a nonlinear nonautonomous system is approximated by a concatenation of local linear models

$$f(\psi) \approx \bigcup_{i=1,\cdots,N} f_i(\psi_y,\psi_u) \qquad (8)$$

Once we identify the plant using multiple models, it is necessary to associate these models with a corresponding controller.

C. *Structure of Controllers*

Now we discuss the control problem for the local linear model using an inverse control framework [13,21]. The central advantage of such a framework is that an inverse model can be used directly to build a feed-forward controller. Thus, for the desired behavior, the controller just asks the model to predict the action needed.

As stated before, our principal objective is to determine a control input, $u(k)$, which will result in the output, $y(k+1)$, of the plant tracking with sufficient accuracy a specified sequence, $y_d(k+1)$. The system identification block has *N* predictive models denoted by $\{f_i\}_{i=1}^N$, in parallel. Corresponding to each model f_i, a controller C_i is designed such that C_i achieves the control objective for f_i. Therefore, at every instant one of the models is selected and the corresponding controller is used to control the actual plant. In order to control a plant, consider the control problem where the dimension of the input is equal to that of the output, that is, $p = q$. From equation (7), because $p = q$, and under the assumption that b_o is invertible, the control law of an inverse controller for the model, f_{i^o}, can be directly calculated as

$$u_{i^o}(k) = b_o^{-1}\left(y_d(k+1) - \sum_{j=0}^{d_y} a_j^{i^o} y(k-j) - \sum_{j=1}^{d_u} b_j^{i^o} u(k-j) \right) \qquad (9)$$

Therefore, at time instance *k*, the control $u_{i^o}(k)$ can be obtained, if the future target of $y(k)$, $y_d(k+1)$, is known. Therefore, the set of local linear models simplifies the control design for a nonlinear plant. So instead of a global neuro-controller as in other adaptive control schemes [12], here we can get by with a group of linear controllers associated with each identified model, thus taking care of the system over the whole operating regime. One advantage of this scheme is its simplicity and fast convergence to get the desired response. Another advantage is that the dynamic space is decomposed in the appropriate switching among very simple linear models, which leads to accurate modeling and controls.

On the other hand, a possible disadvantage of the proposed approach is that the overall stability may not be guaranteed due to the switching among models if the models are quite different each other. The models in the neighborhood, however, are alike using a SOM and the unwanted effect of discontinuities between the models may not be occurred. Moreover, as Narendra and Balakrishnan have shown in [13], the overall system will be globally stable for any arbitrary switching sequence, provided that a sufficient number of local models are given. As another disadvantage of the proposed method, creating a set of models by embedded input and output may possibly cause serious problem in the presence of large noise or outliers since the wrong predictive model due to noise is very likely to cause poor control. Hence, the selection of the right model is as important as creating models and designing controllers.

III. Selection of Operating Regimes with a SOM

Building local mappings in the full operating space is a time and memory consuming process, which led to the natural idea of quantizing the operating regimes and building local mappings in positions given by prototype vectors obtained from running the plant. For quantization of the operating regimes, the k-nearest-neighbor method is effective but it disregards neighborhood relations, which may affect performance [10]. In contrast, the SOM has the characteristic of being a local framework liable to limit the interference phenomenon and to preserve the topology of the data using neighborhood links between PEs. Neighboring PEs in the network compete with each other by means of mutual lateral interactions, and develop adaptively into specific detectors of different signal patterns [8]. The training algorithm is simple, robust to missing values, and it is easy to visualize the map. These properties make SOM a prominent tool in data mining [34].

In most of the papers discussing local linear models for system identification, the SOM has been used with a first order expansion around each PE in the output space. The SOM transforms an incoming signal pattern of arbitrary dimension into a one or two-dimensional discrete map, and performs this transformation adaptively in a topologically ordered fashion [8]. The results obtained so far with this methodology have been quite promising, however, problems that need to be solved remain: first, efficiently partitioning the operating regimes in high dimensional spaces is still a problem due to the curse of dimensionality [5]; second, it may be hard to find a small number of variables to characterize the operating regimes due to the possibly large number of local

[1] This requires a rich training set with a variety of maneuvers.

models; third, all the methods have to be extended for nonautonomous regimes.

The previous work by Principe et al. [17] provided the starting point for the proposed modeling architecture. Their objective was to construct a neural architecture, capable of capturing locally the underlying dynamics of a chaotic time series (based on a SOM). For an autonomous system, it is reasonable to assume that the future behavior of the system can be predicted over some finite interval from a finite number of observations of past outputs. In contrast, for predictions of the behavior of a nonautonomous system, we have to consider two different dynamics; one for the state space and the other for the control input space. Consequently, the most important difference is how to capture the dynamics in the input-output joint space, which is fundamental for identifying the unknown nonautonomous system. Several options are possible, and we have been investigating them:

Firstly, we tried to find the local models by quantizing the input-output joint space by embedding not only the outputs but also the control inputs using one SOM. This modification is essential because the purpose is to characterize the system dynamics that exist in the input-output joint space. However, we encountered some difficulties such as normalization of the joint space and large dimensionality of the space involved (many degrees of freedom and large dynamic range of parameters) [2].

Secondly, in order to reduce the approximation error with local models based on a SOM, we utilized a counter-propagation network, which are hybrid networks that combine supervised and unsupervised learning to create a self-organizing look-up table that can be used for function approximation. The advantage of this network over conventional look-up tables is that the Kohonen map provides for a statistically optimal coverage of the input space even if the mathematical form of the underlying function is completely unknown. As input feature vectors from a training set are presented to the network, unsupervised learning is used to create a topology-preserving (Kohonen) map of the input data while, at the same time, supervised learning is used to associate an appropriate output feature vector with each PE on the map. Since the output at each PE is just the average output for all of the feature vectors that map to that point local models might be created for better approximation using the quantization error in the input space and the average output. This is achieved by coupling each PE with a linear mapping in such a way that a functional relationship can be established between each Voronoi region in the input space (of the SOM) and the desired signal [37]. However, this method required a much larger map to make the estimation error in the desired output space smaller. Additionally, when noise is added in the input of the SOM, the quantization error in the input Voronoi region may be magnified by the local models.

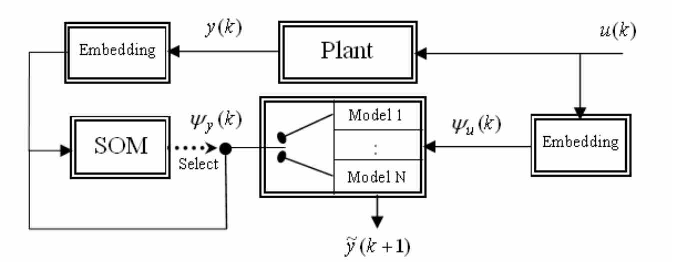


Fig. 1. Block diagram of SOM-based modeling for nonautonomous system.

As the number of dependent variables is increased, the process becomes increasingly difficult to model accurately. This led us to think that a model that uses only a few of the observed variables will be more accurate than a model that uses all the observed variables. In this scheme, therefore, we let the SOM look at only the current output and its past values to decide the winner, and create the models with the control inputs.

Here we will pursue the last option for the following reasons. The competitive learning rule works best for normalized inputs. The SOM algorithm uses the Euclidean metric to measure distances between vectors. For example, if one variable has values in the range of [-100,…,100] and another in the range of [-1,…,1] the former almost completely dominates the map organization because of its greater impact on the measured distances. Either, the measure of distance is weighted by the inverse of the scales or the data must be normalized such that each component of the input vectors have unit variance and zero means [1]. However, normalization looses information (the mean or the scale can be important) and it can become meaningless if the data dynamic range (or mean) changes over time. Therefore we cannot normalize the data (nor create the weighted Euclidean metric) in this way since it is not always guaranteed that the mean and the dynamics range of the data are available. In addition, SOM modeling becomes increasingly difficult with increasing number of dependent variables because it is basically a memory based approach that does not scale up well with the input dimension. Accordingly a model that uses only a few of the observed variables will be more accurate than a model that uses all of the observed variables. When the SOM modeling is done in the output space, we let the SOM look at only the current output and its past values to decide the winner which represent the operating regime, and create the models with the control inputs as shown in Fig. 1. By doing this, normalization of the input space is not necessary since the clustering is performed solely by the history of the output.

IV. Model Development Procedure

The SOM is employed as a modeling infrastructure to construct the local models. It provides a codebook representation of the plant dynamics and organizes the different dynamic regimes in topological neighborhoods. Thus we can create a set of models that are local to the data in the Voronoi tessellation created by the SOM. In this architecture of local linear modeling, the SOM is trained to position the local models in the embedded output space and the development of local models is done by directly fitting the quantized embedded output samples obtained from the SOM and corresponding embedded control input samples.

A. *Training the SOM*

The first learning phase (self-organizing phase) decides the proper partition of the local models and reduces the high dimensional embedded output space. This step is accomplished via the Kohonen learning process. Let $\psi_y(k) = [y(k), \cdots, y(k-d_y)]$ denote the input vector for the SOM and w_i denote the weight vector of PE i. With each vector, $\psi_y(k)$, presented as the input to the network, the Kohonen learning algorithm [8] adaptively discretizes the continuous input space into a set of *N* disjoint Voronoi cells. The response of a SOM to input $\psi_y(k)$ is determined by the reference vector w_{i^o} of the PE that produces the best match to the input.

$$i^o = \arg\min_i \{\| \psi_y(k) - w_i(k) \|\} \tag{10}$$

Then the *k*th adaptation of the weights is done in the following manner.

$$w_i(k+1) = w_i(k) + \eta(k)\Lambda_i(k)(\psi_y(k) - w_i(k)), \quad i = i^o \tag{11}$$

where $\eta(k)$ and $\Lambda(k)$ designate respectively a learning rate and a neighborhood function, and both of them gradually shrinks with time k. Each new feature vector presented to the network will trigger a response that is the average for those feature vectors closest to it in the input data space.

B. *Estimation of the Local Models*

The SOM preserves topological relationships in the input space in such a way that neighboring inputs are mapped to neighboring PEs in the map space. Then, when each PE is extended with a local model it can actually learn the mapping $\tilde{y}(k+1) = f(\psi_y(k), \psi_u(k))$ in a supervised way. In this step, the embedded control input space, $\psi_u(k)$, corresponding to each local embedded output space, $\psi_y(k)$, is utilized to find the models that cover the full range of operation we want our models to cover. Each PE has an associated local model $\{\vec{a}_i, \vec{b}_i\}$ in equation (7) that represents the approximation of the local dynamics.

The local model weights $\{\vec{a}_i, \vec{b}_i\}$ are computed directly from the desired signal samples y_i^j and the input-output samples by a least square fit within a Voronoi region centered at the current winning PE chosen from $\psi_y(k)$. The size of the data samples in the region must be at least equal to the ($d_y + d_u + 2$)-dimensional basis vector. The design procedure for this local model is as follows:

1. Apply training data to the SOM and find the winning PE corresponding to the input $\psi_y(k)$ such that we have winner-input pairs.
2. Use the least square fit to find the local linear model coefficients for the winning PE, i^o, where desired output vector $y_i^j \in \Re^M$ as

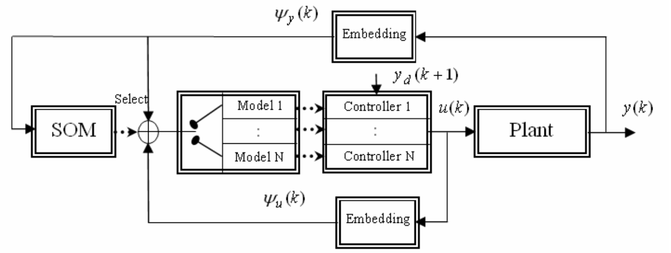


Fig. 2. Proposed SOM-based inverse control scheme.

$$y_{i^o}^j = [\vec{a}_{i^o}^T \ \vec{b}_{i^o}^T] \begin{bmatrix} \psi_{yi^o}^j \\ \psi_{ui^o}^j \end{bmatrix} \quad for\ \forall j \in M \tag{12}$$

where $\vec{\theta} = [\vec{a}_{i^o}^T \ \vec{b}_{i^o}^T]$ is the sought linear model coefficients, *M* is the size of inputs involved in the winning PE i^o. Specifically, the least-squares problem

$$Y = \vec{\theta}X \tag{13}$$

is solved for $\vec{\theta}$, where $X \in \Re^{(d_y+d_u+2)\times M}$ is defined as a matrix that contains each input vector associated with the winning PE, and $Y \in \Re^M$ is defined as a vector that contains the target outputs. It is well-known that although the least-squares solution obtained from (13) is reasonably good when the noise level is low, the estimates tend to be biased for higher levels of noise [17]. Addition of a single sample to a cluster can radically change the distances. Besides, the models will perform very well for that particular training set with very low error because it has memorized the training examples but they may not perform well with new data sets. Thus we make use of data samples from the winner as well as the neighbors to create the local models in order to make them more robust as well as to improve the generalization for the network. Also we take the data samples from the neighbors in case less data than the dimension of the input are assigned in some Voronoi region. The performance difference between these two approaches is demonstrated with simulations.

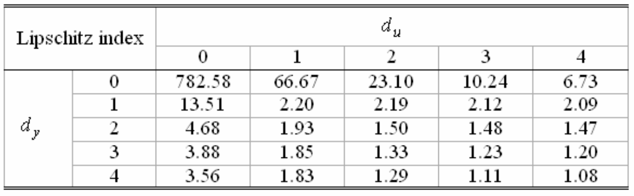
3. In testing, once the winning PE is determined we select the appropriate local model from the list of associated models. Apply the local model to obtain the estimated output

$$\tilde{y}(k+1) = \vec{a}_{i^o}^T \psi_y(k) + \vec{b}_{i^o} \psi_u(k) \tag{14}$$

C. *Design of the Controllers*

We developed in advance a set of local linear models for the plant and switch them according to the measured output history. Thus, once the right local linear model is determined, the corresponding controller is designed using (9). A schematic diagram of the proposed SOM based inverse control system is shown in Fig. 2 where the inverse control seeks to model the inverse of the plant. A set of controllers appears in series with the plant. The command input, $y_d(k+1)$, is fed to the controller and provides also the desired response. Hence, when the error is small the controller transfer function is the inverse of the plant.

TABLE I
LIPSCHITZ INDEX FOR ESTIMATING THE EMBEDDING DIMENSION FOR THE MISSILE SYSTEM.



Lipschitz index		d_u				
		0	1	2	3	4
d_y	0	782.58	66.67	23.10	10.24	6.73
	1	13.51	2.20	2.19	2.12	2.09
	2	4.68	1.93	1.50	1.48	1.47
	3	3.88	1.85	1.33	1.23	1.20
	4	3.56	1.83	1.29	1.11	1.08

Generally, an adaptive controller that meets the specifications is slow to adapt. However, our approach models all the operating regimes and automatically divides the operating regimes by the number of PEs. So once the current operating region is determined by the SOM the corresponding controller is triggered so that the plant tracks the desired signal. Moreover, even if wrong PE is assigned in the winning PE due to noise a similar dynamic model can be activated since neighboring SOM PEs represent neighboring regions in the dynamic space. Thus, the proposed control system can reach the set point fast, and even if the dynamic model is not the most appropriate, there is an extra flexibility to match the set point with small error.

V. SIMULATION RESULTS

In this section, the proposed SOM-based local modeling algorithm is applied to two examples. The first example is a realistic nonlinear missile system. The second example describes the application of the proposed method to the LoFLYTE® UAV model.

A. Example1: Missile Dynamics

We consider a simplified nonlinear missile dynamic equation [15]. This model is obtained using the following assumptions; constant velocity, stabilized roll angle, no change of mass.

$$\dot{x}_1 = x_2 - 0.1\cos(x_1)(5x_1 - 4x_1^3 + x_1^5) - 0.5\cos(x_1)u$$
$$\dot{x}_2 = -65x_1 + 50x_1^3 - 15x_1^5 - x_2 - 100u \qquad (15)$$
$$y = x_1$$

where x_1 is sideslip angle, x_2 is yaw-rate, and u is rudder deflection. In (15), we consider a SISO model, assuming that only the state x_1 is available for measurement. The signal is obtained by integrating the equation (15) with the forth-order Runge-Kutta method with a sampling time of $0.05s$. In order to cover the whole control envelope, the input signal is uniformly distributed between –0.5 to 0.5. We choose the model as

$$\tilde{y}(k+1) = f\big(y(k), y(k-1), u(k), u(k-1)\big) \qquad (16)$$

by Lipschitz index (see [6]) shown in Table I. The theory states that the best embedding dimension is obtained when the index stops decreasing. A SOM is trained with $\big(y(k), y(k-1)\big)$ over 6000 samples. After training, local linear models are constructed from the embedded output (used for training) and the embedded control input corresponding each PEs. The created models are tested by a new sequence of 2000 samples. The network dimension was chosen as 10×10 based on prediction performance shown in Fig. 3.

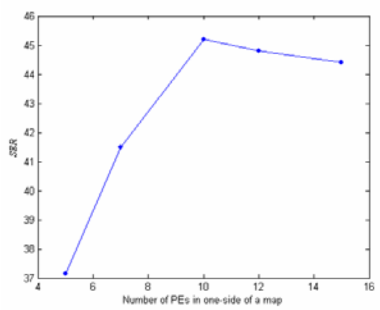
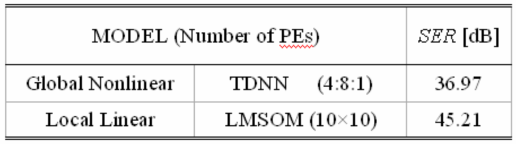


Fig. 3. One-step prediction performance *v.s.* network dimension on independently generated test data.

TABLE II
COMPARISON OF THE PREDICTION RESULTS OBTAINED BY TDNN AND LMSOM ON TEST SET.



MODEL (Number of PEs)		*SER* [dB]
Global Nonlinear	TDNN (4:8:1)	36.97
Local Linear	LMSOM (10×10)	45.21

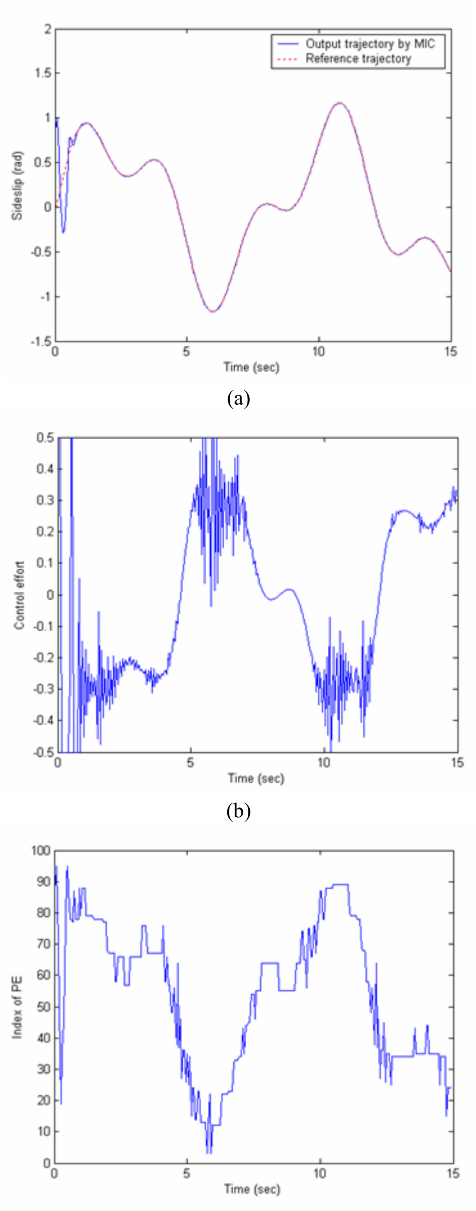
Table II shows the comparison of the one-step prediction performance for different models evaluated through Signal-to-Error Ratio (*SER*) defined in

$$SER\,[dB] = 10\log_{10}\left\{ \sum_k y(k+1)^2 \Big/ \sum_k (y(k+1) - \tilde{y}(k+1))^2 \right\}.$$

The best result with LMSOM (Local Modeling based on a SOM) was a *SER* of around 45.21 dB while with the Time Delay Neural Network (TDNN) one obtained a *SER* of about 36.97 dB. The number of PEs in the hidden layer of the TDNN is chosen as 8 by 20 Monte-Carlo simulations varying the size of the hidden layer. This result shows that the SOM local modeling scheme is better than the global modeling.

After creating a set of local models the design of the controllers is carried out for each of the local models. The multiple controller-plant pair is tested for tracking a desired trajectory $y_d(k+1) = 0.8\sin(3\pi k/250) + 0.5\sin(3\pi k/100)$. Fig. 4(a) shows the desired trajectory and the system output. Fig. 4(b) is the output from the designed Multiple Inverse Controller (MIC) and Fig. 4(c) is the history of winner of the SOM. Note that the system output perfectly tracks the trajectory except in a transient time of a few seconds. The switching is done automatically by the SOM. The winner trajectory clearly indicates that the controllers are switching accurately so that the overall system with the controller tracks the desired signal very well.

Alternatively, a TDNN inverse controller may be utilized to control a complex plant in the presence of uncertainty. Thus we compared the performance of the proposed system with that of a TDNN inverse controller trained through the TDNN model. The controller parameters in the fixed control structure



(a)

(b)

(c)

Fig. 4. Tracking performance result by SOM-based inverse controller. (a) Side-slip angle: desired and actual output (b) Control input (c) Winner index.

are adapted by an algorithm that ensures that the desired performance level is maintained and the parameters are updated by back propagating the error through the model as shown in Fig. 5. The results using a TDNN inverse controller and using the MIC for trajectory tracking are compared in Table III where the experiment performed with multiple controllers demonstrated superiority to one global inverse controller. In addition, we performed experiments for set point tracking by a TDNN controller and MIC. Fig. 6 compares the result of controlling the side-slip of the missile to several different set points. As can be seen in the figure, SOM-based multiple inverse controllers are doing a good job in tracking the desired signal, since they know the precise dynamics at each point. On the other hand, a global nonlinear controller does not follow the desired signal as accurately, though it is able to track the changes. Additionally, the proposed multiple-controller scheme exhibits both reduced overshoot and settling time compared with the global inverse controller.

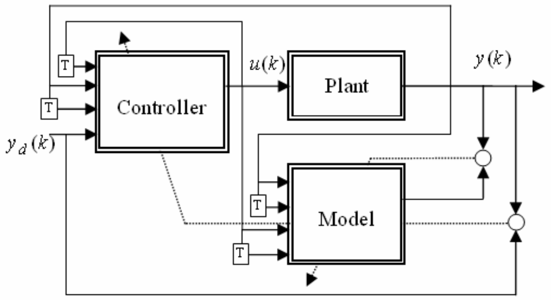


Fig. 5. TDNN inverse controller via Backpropagation through (Plant) Model.

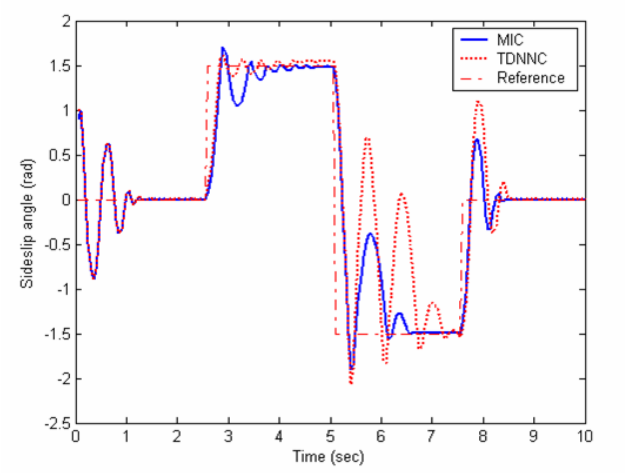


Fig. 6. Comparison of the performance for set point tracking by the SOM-based inverse controller and a TDNN inverse controller.

TABLE III
COMPARISON OF THE OVERALL RESULTS ON ARBITRARY TRAJECTORY TRACKING OBTAINED BY A TDNN INVERSE CONTROLLER AND SOM-BASED INVERSE CONTROLLER FOR THE MISSILE SYSTEM.

CONTROLLER		SER [dB]
Global Nonlinear	TDNN inverse controller	28.2
Local Linear	Multiple inverse controllers	34.9

B. *Example2*: *LoFLYTE*® *UAV*

The second example of the nonlinear dynamical system considered for this study is the LoFLYTE® UAV designed by Accurate Automation Corporation (AAC), shown in Fig. 7. The LoFLYTE® program is an active flight test program at the Air Force Flight Test Center at Edwards Air Force Base, with the objective of developing the technologies necessary to design, fabricate, and flight test a Mach 5 waverider aircraft [3,38]. In addition, the LoFLYTE® UAV has been used to understand the low speed characteristics of a hypersonic shape and to demonstrate several innovative flight control technologies. Our task is to develop an algorithm for modeling and controlling the LoFLYTE® system solely based on input-output data. In classical notation, longitudinal motion consists

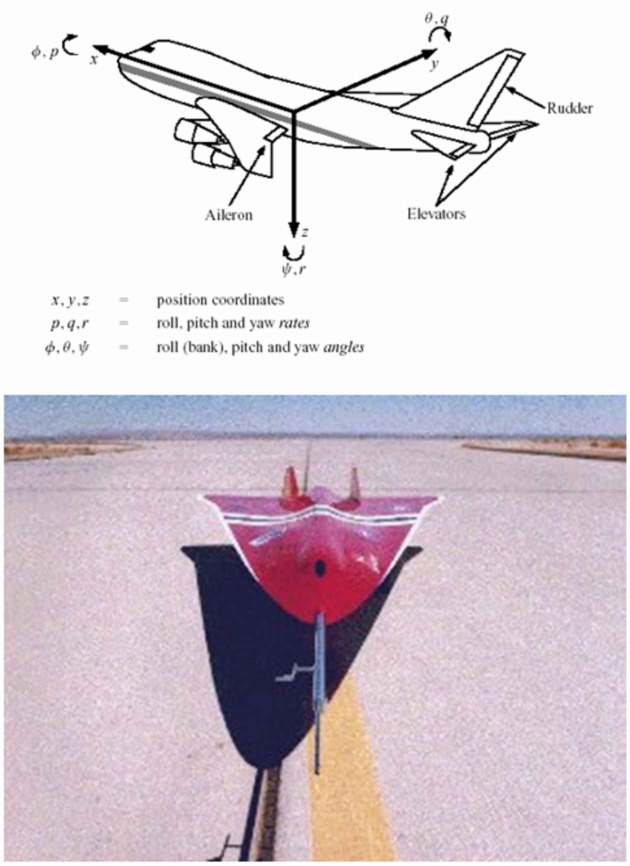

ϕ, p
x
θ, q
y
Rudder
Aileron
Elevators
z
ψ, r
x, y, z = position coordinates
p, q, r = roll, pitch and yaw *rates*
ϕ, θ, ψ = roll (bank), pitch and yaw *angles*

Fig. 7. General description of aircraft (top) and LoFLYTE® testbed aircraft (bottom).

of pitching (θ, q) motion, while the lateral motion consists of rolling (ϕ, p) and yawing (ψ, r) movement. The elevator (δ_e) and the throttle (δ_{th}) control the longitudinal motion, while the aileron (δ_a) and rudder (δ_r) primarily affect lateral motion. The general dynamics of the system are described by

$$\begin{aligned}
\dot{u} &= -(wq - vr) - g\sin\theta + F_x / M_a \\
\dot{v} &= -(ur - wp) + g\cos\theta\sin\phi + F_y / M_a \\
\dot{w} &= -(vp - uq) + g\cos\theta\cos\phi + F_z / M_a \\
\dot{p} &= \left((I_{yy} - I_{zz})qr + I_{xz}(\dot{r} + pq) + L\right)/ I_{xx} \\
\dot{q} &= \left((I_{zz} - I_{xx})rp + I_{xz}(r^2 - p^2) + M\right)/ I_{yy} \\
\dot{r} &= \left((I_{xx} - I_{yy})pq + I_{xz}(\dot{p} - qr) + N\right)/ I_{zz} \\
\dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\
\dot{\theta} &= q\cos\phi - r\sin\phi \\
\dot{\psi} &= q\sin\phi\sec\theta + r\cos\phi\sec\theta
\end{aligned} \tag{17}$$

where M_a is the aircraft mass and F_x, F_y, F_z, L, M, N are actuator-induced forces and moments. The forces and moments are nonlinear functions of aircraft's states and control inputs [25]. This model was programmed in a software simulator by AAC and was used here as the source of the data. In this study, we wish to estimate the aircraft's outputs under the assumption that we can only access part of the states which are p, q, r, u, v, w while the goal is to track the desired trajectory considering the case of an aircraft moving with a constant throttle.

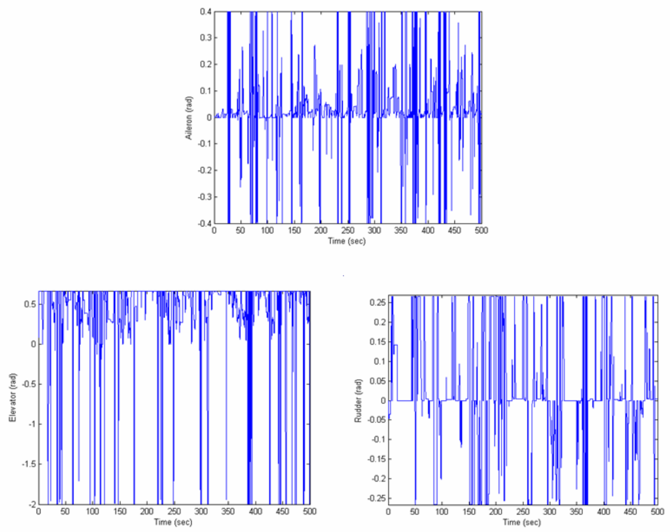
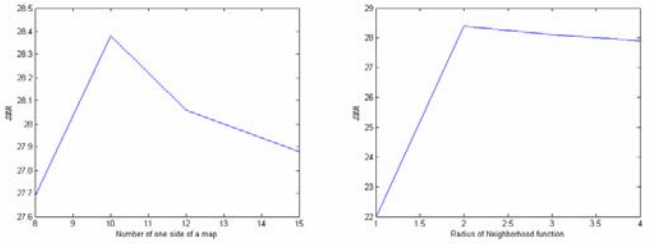


Fig. 8. Training set: 3 control inputs ($\delta_a, \delta_e, \delta_r$).



(a) (b)

Fig. 9. Performance of estimation of p *v.s.* (a) network dimension (b) radius of neighborhood function.

To model the aircraft dynamics (*p, q, r, u, v, w*), a total of 6 SOMs are used for quantization of each embedded output space as predictors, for instance, one SOM is trained with $\psi_p(k) = [p(k), \cdots, p(k - d_y)]^T$ for modeling roll-rate dynamic, p. Thus the linear coupling between p, q, r, u, v, w is only implicitly modeled[2]. In this way, each output of the aircraft can be described by a dynamic model that takes into account the control input variables such as $\delta_a, \delta_e, \delta_r$:

$$\tilde{y}(k+1) = f_i(\psi_u(k), \psi_y(k)), \quad i = 1, \cdots, N \tag{18}$$

where

$$\psi_u(k) = \{\delta_a(k), \cdots, \delta_a(k - d_u), \delta_e(k), \cdots, \delta_e(k - d_u), \delta_r(k), \cdots, \delta_r(k - d_u)\}^T.$$

By doing this, the complexity can be reduced and it helps to understand the raw data.

Again, we selected an embedding dimension based on the Lipschitz index. The results are shown in Table IV. Since the

[2] As we mentioned in the introduction, due to difficulties related with dynamic range normalization, local linear models that take state coupling into account are not as accurate as this approach. Instead, we utilize the delayed outputs in order to compensate for the disregarded information due to the coupling.

TABLE IV.
LIPSCHITZ INDEX FOR ESTIMATING THE EMBEDDING DIMENSION FOR LOFLYTE® UAV.

p		d_u					q		d_u				
		0	1	2	3	4			0	1	2	3	4
d_y	0	14627	8208	8208	8208	4704	d_y	0	17503	17503	17503	17503	2188
	1	53	53	53	36	17		1	181	90	90	90	90
	2	21	13	9	9	9		2	19	19	19	19	19
	3	12	9	6	6	5		3	13	6	6	5	5
	4	10	7	6	6	5		4	12	3	3	3	3
r		d_u					u		d_u				
		0	1	2	3	4			0	1	2	3	4
d_y	0	2159	1073	855	855	855	d_y	0	18092	18092	18092	18092	18092
	1	26.9	26.9	26.9	26.9	26.9		1	1008	476	476	238	238
	2	2.6	2.6	2.6	2.6	2.6		2	279	228	228	228	228
	3	2.2	2.2	2.2	2.2	2.2		3	110	67	53	43	41
	4	1.6	0.9	0.8	0.8	0.8		4	93	63	43	41	41
v		d_u					w		d_u				
		0	1	2	3	4			0	1	2	3	4
d_y	0	81884	81884	26872	26872	26872	d_y	0	11719	10080	4138	3812	3812
	1	610	74	74	74	74		1	281	281	281	281	120
	2	16	6	6	6	6		2	29	16	11	11	11
	3	16	5	5	4	4		3	21	10	5	5	5
	4	16	5	5	4	4		4	15	10	4	4	4

TABLE V.
COMPARISON OF THE IDENTIFICATION PERFORMANCE ON 10 SETS OF TEST DATA.

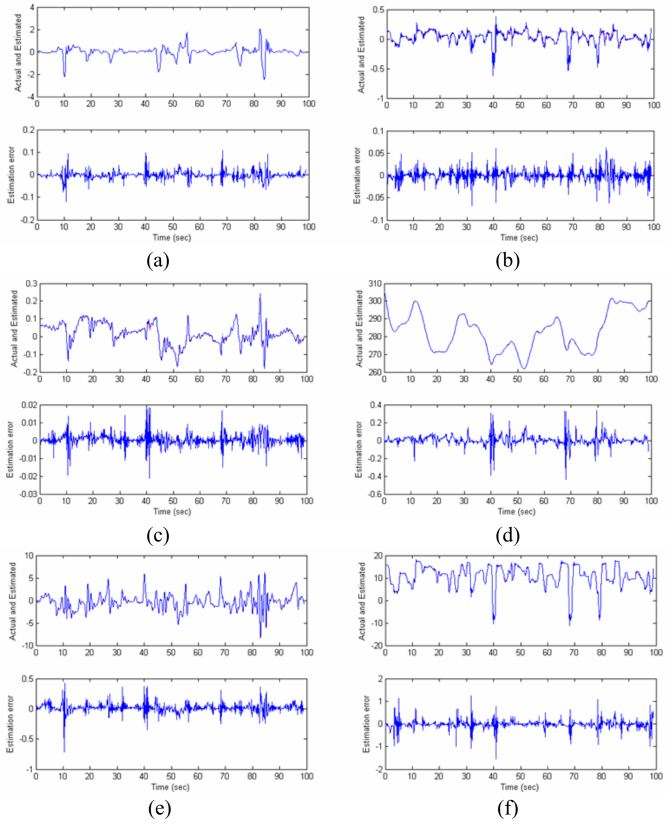
Model	*SER* [dB] (*NRMSE*)					
	p	q	r	u	v	w
LMSOM	28.12±1.2 (0.0093± 0.0025)	20.21±1.9 (0.0266± 0.0070)	28.56±3.3 (0.0149± 0.0053)	74.73±3.2 (0.0001)	27.83±1.9 (0.0105± 0.0027)	37.98±3.7 (0.0094± 0.0047)
TDNN	27.63±1.3 (0.0101± 0.0021)	19.57±1.0 (0.0278± 0.0077)	27.34±2.5 (0.0169± 0.0040)	47.56±4.7 (0.0046± 0.0035)	27.24±1.7 (0.0169± 0.0023)	36.57±3.7 (0.0111± 0.0055)

optimal values of embedding dimension for each output dynamic model are different, we chose the largest number among the embedding dimensions selected for each output estimation as $d_y = 3$ for each output and $d_u = 3$ for 3-D control inputs (aileron, elevator and rudder). The linearized input/output relationship then is

$$\begin{gathered}\tilde{p}(k+1) = \sum\nolimits_{i=1}^{4} a_i^p\, p(k-i+1) + \\ \sum\nolimits_{i=1}^{4} \left\{ b_i^p \delta_a(k-i+1) + c_i^p \delta_e(k-i+1) + d_i^p \delta_r(k-i+1) \right\} \\ \vdots \\ \tilde{w}(k+1) = \sum\nolimits_{i=1}^{4} a_i^w\, w(k-i+1) + \\ \sum\nolimits_{i=1}^{4} \left\{ b_i^w \delta_a(k-i+1) + c_i^w \delta_e(k-i+1) + d_i^w \delta_r(k-i+1) \right\}\end{gathered} \tag{19}$$

The training samples are obtained by exciting the aircraft dynamics (LoFLYTE® Simulator) using the control inputs shown in Fig. 8. Each SOM is trained with the embedded output, ψ_y whose dimension is 4, over 5000 samples and the best result was obtained with a 10×10 grid map ($N = 100$) as shown in Fig. 9(a). It is noted that with an increasing size of the map less data are assigned to each PE, and so proper training requires larger data sets.

After training the SOMs, 100 local linear models are constructed from ψ_y (used for training) as well as the embedded control inputs ψ_u corresponding to each PEs in the least square sense. In this phase, we utilized the winner and the neighbor PE's to create the local models in order to improve the generalization for the network.



(a) (b)

(c) (d)

(e) (f)

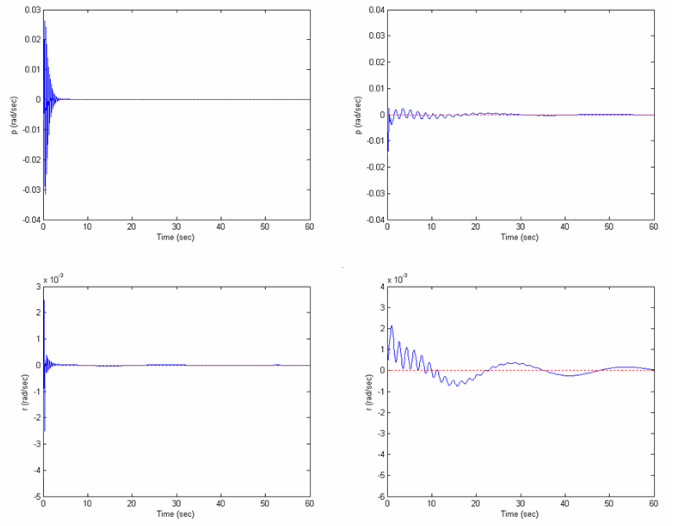
Fig. 10. Identification of 6 outputs and corresponding error: (a) p (b) q (c) r (d) u (e) v (f) w

As shown in Fig. 9(b), better performance in the testing set was obtained when we take the data samples from the winning PE as well as neighbor PEs in local modeling phase rather than from the winning PE alone[3]. The created models were tested by new sequences of 10 sets with 1,000 samples. Table V shows the identification performance of different outputs of the system using LMSOM by the approximation accuracy, which is evaluated through Signal-to-Error Ratio (*SER*) and Normalized Root Mean Square Error (*NRMSE*) $= 1/\max(y)\sqrt{1/L\sum_k (y(k+1) - \tilde{y}(k+1))^2}$.

The same data set was used in training a TDNN to compare with the performance of the proposed method for system identification. Training conditions, such as the embedding dimension and the number of network were kept the same in this comparison between the local modeling and the global modeling. Table V shows the best identification performance with TDNN which was trained using the Conjugate-Gradient algorithm. The best result with TDNN was obtained from 20 Monte-Carlo simulations with 12 PEs in the hidden layer[4]. From Table V, we can conclude that the constructed SOM-based network is a good model of the underlying dynamics because it provides larger *SER* and smaller *NRMSE* for all

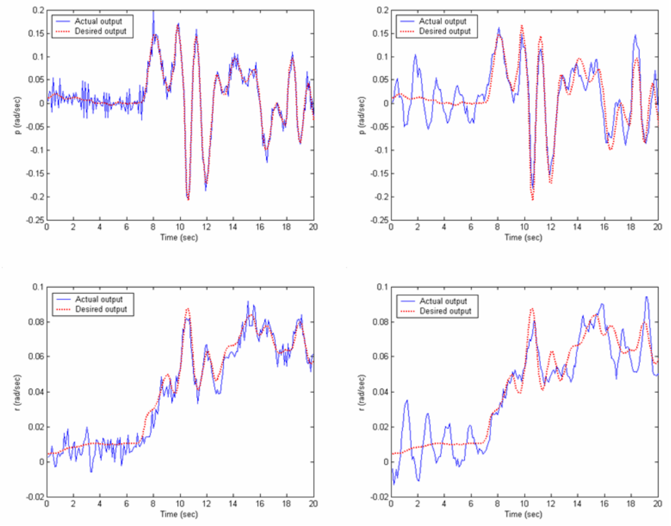
[3] Taking neighbor PEs means that the radius of neighborhood function is greater than 1.

[4] We tried various sizes of TDNN's and found that the one with 12 PE's in the hidden layer performs best in system identification.



(a) (b)

Fig. 11. Comparison for controlling roll-rate and yaw-rate to track the set point (a) by the proposed inverse controller and (b) by a global inverse controller.



(a) (b)

Fig. 12. Performance of controlling roll-rate and yaw-rate to track an arbitrary trajectory with measurement noise (SNR = 20) (a) by the proposed inverse controller and (b) by a global inverse controller.

outputs than the TDNN. Fig. 10 shows the superposition of the desired signal and the model output and corresponding errors by the SOM-based local models, and it can be seen that the estimated output closely follows the original signal. Consequently, it turned out that the proposed strategy of finding proper location of fixed models depending on the prior information available to the designer for finding aircraft dynamics is superior to those using a single global nonlinear model. In addition, it should be noted that the proposed modeling scheme makes identification of the plant very compact and computationally efficient since the aircraft dynamics are captured in a compact lookup table of linear models.

We now consider the control problem with the SOM-based local models created. When we design controllers we usually assume that the coupling between lateral and longitudinal motion is minimal. Here, we performed a simulation to control the roll-rate (p) and yaw-rate (r) of the aircraft by aileron (δ_a) and rudder (δ_r), setting elevator to zero and throttle to constant. Thus, once we have the linear models for the roll-rate and yaw-rate, and the desired values, $p_d(k+1)$ and $r_d(k+1)$, the inverse controller (inversion-based predictive model), $\delta_a(k)$ and $\delta_r(k)$, for the aircraft's roll-rate and yaw-rate tracking is obtained by

$$\begin{bmatrix}\delta_a(k)\\ \delta_r(k)\end{bmatrix} = f_i^{-1}\left(\begin{bmatrix} p_d(k+1), p(k),\cdots,p(k-3),\delta_a(k-1),\cdots,\\ \delta_a(k-3),\delta_r(k-1),\cdots,\delta_r(k-3)\\ r_d(k+1),r(k),\cdots,r(k-3),\delta_a(k-1),\cdots,\\ \delta_a(k-3),\delta_r(k-1),\cdots,\delta_r(k-3)\end{bmatrix}\right) \quad (20)\ \ (20)$$

For performance comparisons, we applied a global inverse controller for the same control problems. A TDNN as a global inverse controller is trained by back-propagating error through the model.

Fig. 11 compares the set point tracking performance of a TDNN controller and the multiple inverse controller in the absence of sensor noise. From the responses it can be seen that the multiple controller approach is obviously very good except the first few seconds. However, it shows poor transient response when the TDNN controller is utilized. Another performance test is to enforce the tracking of the roll-rate and yaw-rate to signals $p_d(k+1)$ and $r_d(k+1)$ which are given in real time during the course of the flight, while being subjected to unmeasured sensor disturbances. All 6 output measurements are corrupted by zero-mean random sequences with 20dB of signal-to-noise ratio (SNR). The results of a flight test with the proposed method are shown in Fig. 12(a) and in Fig. 12(b) we show the same with the TDNN controller. It can be seen that the roll-rate and the yaw-rate track their command signals quite well even under the existence of measurement noise by the multiple controllers. The simulated flight test demonstrates that the proposed controller is capable of closely approximating perfect inversion by only looking at the past information. Also, it proves that the multiple controller framework indeed provides exceptional tracking.

VI. Concluding Remarks

Local models have been a source of much interest because they have the ability to adhere to the local shape of an arbitrary surface, which is difficult especially in cases when the dynamical system characteristics vary considerably throughout the state space. Therefore, in a number of cases, the local modeling approach, despite its simplicity, has provided more effective and accurate approximation than global modeling. Local linear modeling is the simplest implementation of the local modeling method. However, most of the work on local linear approaches only addresses the recursive state-dependent prediction. Approximation of higher-order nonautonomous systems using a finite set of local linear models has not been fully studied.

In this paper, we have attempted to solve the problem of nonlinear system identification and control using the local linear modeling methodology. Classical control approaches involving linearization and gain scheduling are seldom applicable to complex nonlinear systems. However, an obvious way to analyze complex systems is to adopt the divide and conquer strategy. We have taken the concept of self-organization in embedded output space extended with multiple models by the SOM. The significance of the proposed scheme is that the operating regime is selected by the embedded output and that local models are built by the embedded output as well as the embedded control input data samples which are spaced in the local regime, which marks the fundamental difference between the work of this paper and others. We have shown the effectiveness of the proposed approach to identify complex aircraft dynamics. Its comparison with other neural network-based alternatives showed clear advantages of local modeling in terms of performance. In addition, we have designed an inverse controller for every linear model and the switching is astutely handled by the SOM depending on the delayed version of output of the nonlinear system. The proposed multiple modeling and control strategy has been successfully tested in modeling and control of the LoFLYTE® UAV.

References

[1] G. Chakraborty and B. Chakraborty, “A Novel Normalization Technique for Unsupervised Learning in ANN,” *IEEE Trans. Neural Networks*, vol. 11, no. 1, pp. 253-257, 2000.

[2] J. Cho, J. Lan, G. Thampi, J. C. Principe and M. A. Motter, “Identification of Aircraft Dynamics Using a SOM and Local Linear Models,” in *Proc. IEEE Midwest Sym. Circuits and Systems*, vol. 2, 2002, pp. 148-151.

[3] C. Cox, J. Neidhoefer, R. Saeks and G. Lendaris, “Neural Adaptive Control of LoFLYTE,” in *Proc. American Control Conf.*, vol. 4, 2001, pp. 2913-2917.

[4] M. O. Efe and O. Kaynak, “Identification and Control of a Nonlinear Bioreactor Plant Using Classical and Dynamical Neural Networks,” in *Proc. IEEE Int. Sym. Industrial Electronics*, vol. 3, 1997, pp. 1211-1215.

[5] M. Ge, M. Chiu and Q. Wang, “An Extended Self-Organizing Map for Nonlinear System Identification, in *Proc. Decision and Control*, 1999, pp. 1065-1070.

[6] X. He and H. Asada, “A New Method for Identifying Orders of Input-Output Models for Nonlinear Dynamic Systems,” in *Proc. American Control Conf.*, 1993, pp. 2520-2523.

[7] T. A. Johansen and B. A. Foss, “Constructing NARMAX models using ARMAX models,” *Int. J. Control*, vol. 58, no. 5, pp. 1125-1153, 1993.

[8] T. Kohonen, *Self-Organizing Maps*, NewYork: Springer-Verlag, 1995.

[9] S. Lawrence, A. C. Tsoi and A. D. Back, “Function Approximation with Neural Networks and Local Methods: Bias, Variance and Smoothness,” in *Proc. Australian Conf. Neural Networks*, 1996, pp. 16-21.

[10] T. M. Martinetz, S. G. Berkovich, and K. J. Schulten, ““Neural-Gas” Network for Vector Quantization and its Application to Time-Series Prediction,” *IEEE Trans. Neural Networks*, vol. 4, no. 4, pp. 558-569, 1993.

[11] M. A. Motter and J. C. Principe, “Classification and Prediction of Wind Tunnel Mach Number Responses using both Competitive and Gamma Neural Networks,” in *Proc. World Cong. Neural Networks*, vol. 2, 1995, pp. 25-29.

[12] K. S. Narendra and K. Parthasarathy, “Identification and Control of Dynamical Systems using Neural Networks,” *IEEE Trans. Neural Networks*, vol. 1, no. 1, pp. 4-27, 1990.

[13] K. S. Narendra and J. Balakrishnan, “Adaptive Control Using Multiple Models,” *IEEE Trans. Automatic Control*, vol. 42, no. 2, pp. 171-187, 1997.

[14] O. Nelles, *Nonlinear System Identification*, NewYork: Springer-Verlag, 2001.

[15] X. Ni, M. Verhaegen, A. J. Krijgsman and H. B. Verbruggen, “A New Method for Identification and Control of Nonlinear Dynamic Systems,” *Eng. Appl. Artif. Intell.*, vol. 9, no. 3, pp. 231-243, 1996.

[16] J. C. Principe and L. Wang, “Nonlinear time series modeling with self-organizing feature maps,” in *Proc. Neural Networks for Signal Processing*, 1995, pp. 11-20.

[17] J. C. Principe, L. Wang and M. A. Motter, “Local Dynamic Modeling with Self-Organizing Maps and Applications to Nonlinear System Identification and Control,” *Proc. of IEEE*, vol. 86, no. 11, pp. 2240-2258, 1998.

[18] J. J. Sidorowich, “Modeling of Chaotic Time Series for Prediction, Interpolation, and Smoothing,” in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, 1992, pp. 121-124.

[19] A. C. Singer, G. W. Wornell and A. V. Oppenheim, “Codebook Prediction: A Nonlinear Signal Modeling Paradigm,” in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, 1992, pp. 325-328.

[20] J. Stark, D. S. Broomhead, M. E. Davies, and J. Huke, “Takens embedding theorems for forced and stochastic systems,” *Nonlinear Analy.,Theo., Metho. & Applic.*, vol. 30, no. 8, pp. 5303-5314, 1997.

[21] G. Thampi, J. C. Principe, J. Cho and M. A. Motter, “Adaptive Inverse Control Using SOM based Multiple Models,” in *Proc. Portuguese Conf. Automatic Control*, 2002, pp. 278-282.

[22] G. Thampi, J. C. Principe, M. A. Motter, J. Cho and J. Lan, “Multiple Model Based Flight Control Design,” in *Proc. IEEE Midwest Sym. Circuits and Systems*, vol. 3, 2002, pp. 133-136.

[23] D. M. Walker, N. B. Tufillaro and P. Gross, “Radial-Basis Models for Feedback Systems with Fading Memory,” *IEEE Trans. Circuits and Systems*, vol. 48, no. 9, pp. 1147-1151, 2001.

[24] J. Walter, H. Ritter and K. Schulten, “Nonlinear Prediction with Self-organizing Maps,” in *Proc. Int. Joint Conf. Neural Networks*, vol. 3, 1990, pp. 589-594.

[25] E. A. Wan and A. A. Bogdanov, “Model predictive neural control with applications to a 6 DoF helicopter model,” in *Proc. American Control Conf.*, vol. 1, 2001, pp. 488-493.

[26] M. Casdagli, “Nonlinear prediction of chaotic time series,” *Physica. D*, vol. 35, no. 3, pp. 35-356, 1989.

[27] I. J. Leontaritis and S. A. Billings, “Input-Output parametric models for nonlinear systems part I: Deterministic nonlinear systems,” *Int. J. Control*, vol. 41, no. 2, pp.303-328, 1985.

[28] R. A. Jacobs, M. I. Jordan, S. J. Nowlan and G. E. Hinton, “Adaptive mixtures of local experts,” *Neural Computation*, vol. 3, no. 1, pp. 79-87, 1991.

[29] L. Bottou and V. Vapnik, “Local learning algorithms,” *Neural Computation*, vol. 4, no. 6, pp. 888-900, 1992.

[30] H. Ritter, T. Martinetz and K. Schulten, “Topology-conserving maps for learning visuomotor coordination,” *Neural Networks*, vol. 2, pp. 159-168, 1988.

[31] R. Murray-Smith, “Local Model Networks and Local Learning,” in *Fuzzy-Duisburg*, pp. 404-409, 1994.

[32] J. D. Farmer and J. J. Sidorowich, “Predicting Chaotic Time Series,” *Phys. Rev. Let.*, vol. 59, no. 8, pp. 845-848, 1987.

[33] R. Murray-Smith and T.A. Johansen, *Multiple Model Approaches to Modeling and Control*, Taylor & Francis Inc., 1997.

[34] J. Vesanto, “Using the SOM and Local Models in Time-Series Prediction,” in *Proc. Workshop on Self-Organizing Maps*, 1997, pp. 209-214.

[35] D. Moshou and H. Ramon, “Extended Self-Organizing Maps with Local Linear Mappings for Function Approximation and System Identification,” in *Proc. Workshop on Self-Organizing Maps*, 1997, pp. 181-186.

[36] B. S. Kim and A. J. Calise, “Nonlinear Flight Control Using Neural Networks,” *J. Guidance, Control, and Dynamics*, vol. 20, no. 1, pp. 26-33, 1997.

[37] J. Cho, J. C. Principe and M. A. Motter, “A Local Linear Modeling Paradigm with a Modified Counterpropagation Network,” in *Proc. Int. Joint Conf. Neural Networks*, vol. 1, 2003, pp. 34-38.

[38] C. Cox, K. Mathia and R. Saeks, “Learning Flight Control and LoFLYTE,” in *Proc. IEEE-WESCON Conf.*, 1995, pp. 720-723.

[39] C. H. Lee and M. J. Chung, "Gain-Scheduled State Feedback Control Design Technique for Flight Vehicles," *IEEE Trans. Aerospace and Electronic Systems*, vol. 37, no. 1, pp. 173-182, 2001.

[40] H. A. Palizban, A. A. Safavi and J. A. Romagnoli, "A Nonlinear Control Design Approach Based on Multi-Linear Models," in *Proc. American Control Conf.*, 1997, pp. 3490-3494.

[41] M. E. Tipping and C. M. Bishop, "Mixtures of Probabilistic Principal Component Analyzers," *Neural Computation*, vol. 11, no. 2, pp. 443-482, 1999.

[42] D. M. Wolpert and M. Kawato, "Multiple Paired Forward and Inverse Models for Motor Control," Neural Networks, vol. 11, pp. 1317-1329, 1998.

Jeongho Cho received the B.S. degree in control and instrumentation engineering from the Soonchunhyang University, Korea, in 1995 and the M.S. degree in electrical engineering from Dongguk University, Korea, in 1997 and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Florida, Gainesville, in 2001 and 2004, respectively.

Since 2005, he has been a postdoctoral research associate in the department of biomedical engineering at the University of Florida, Gainesville. His research interests include time-series prediction and nonlinear system identification, with applications to navigation and control. Specifically, he is currently working with biomedical signal modeling and on EEG signal analysis for the control of epileptic seizure.

Jose C. Principe (M'83–SM'90–F'00) is currently a Distinguished Professor of electrical and biomedical engineering at the University of Florida, Gainesville, where he teaches advanced signal processing and artificial neural networks (ANNs) modeling. He is also a BellSouth Professor and Founder and Director of the University of Florida Computational NeuroEngineering Laboratory (CNEL). He has been involved in biomedical signal processing, in particular the electroencephalogram (EEG), and the modeling and applications of adaptive systems. He has more than 90 publications in refereed journals, ten book chapters, and over 190 conference papers. He has directed 40 Ph.D. degree dissertations and 57 master degree theses.

Dr. Principe is Editor-in-Chief IEEE Transactions on Biomedical Engineering, President Elect of the International Neural Network Society, and former Secretary of the Technical Committee on Neural Networks of the IEEE Signal Processing Society. He is also a member of the Scientific Board of the Food and Drug Administration, and a member of the Advisory Board of the University of Florida Brain Institute.

Deniz Erdogmus (S'95–M'02) received the B.S. degrees in electrical engineering and mathematics, and the M.S. degree in electrical engineering, with emphasis on systems and control, from the Middle East Technical University, Ankara, Turkey, in 1997 and 1999, respectively, and the Ph.D. degree in electrical engineering from the University of Florida, Gainesville, in 2002. From 1999 to 2004, he has been with the Computational NeuroEngineering Laboratory, the University of Florida, Gainesville.

Currently, he is an Assistant Professor of computer science and biomedical engineering at the Oregon Graduate Institute, Oregon Health and Science University, Beaverton. He has coauthored three book chapters, more than 25 papers in refereed journals, and 70 papers in international conferences. His current research interests include information theoretic aspects of adaptive signal processing and machine learning, with applications to problems in biomedical signal processing, communications, and controls.

Dr. Erdogmus is the recipient of the IEEE-SPS 2003Young Author Best Paper Award and the INNS 2004 Young Investigator Award. He is a member of Tau Beta Pi and Eta Kappa Nu.

Mark A. Motter (M'00–SM'00) was born in Columbia, Pennsylvania, on September 3, 1955. He served in the United States Navy from 1973 until 1979, and was honorably discharged at the rank of Electronics Technician First Class. He then began his formal engineering education at Old Dominion University in Norfolk, Virginia, receiving his BSEE, magna cum laude, and MSEE in 1983 and 1985, respectively. Dr. Motter received his Ph.D. in Electrical and Computer Engineering from the University of Florida in 1998.

Since 1985 he has been employed at NASA Langley Research Center, initially involved in the modeling and control of wind tunnels and associated experimental equipment. Currently, he is a controls research engineer in the Electronics Systems Branch. His current research project is investigating the implementation of self-organizing controllers as well as other biologically inspired control approaches, using a fully autonomous unmanned aerial vehicle, which has successfully completed over 500 test flights. He is a senior member of the IEEE, a member of the Academy of Model Aeronautics, and a registered Professional Engineer.