Proceedings of the 8th IEEE Mediterranean Conference
on Control and Automation (MED 2000)
Rio, Patras, GREECE 17-19 July, 2000

ADAPTIVE MODE TRANSITION CONTROL OF NONLINEAR SYSTEMS USING FUZZY NEURAL NETWORKS

FREEMAN RUFUS†, GEORGE VACHTSEVANOS†, BONNIE HECK†

† Georgia Institute of Technology, School of Electrical and Computer Engineering, Atlanta, Georgia, USA, george.vachtsevanos@ee.gatech.edu

Abstract. An adaptation scheme is proposed for the online customization of mode transition controllers designed off-line via the method of blending local mode controllers. It consists of the desired transition trajectory model, the active plant model and the active controller model, which is the mode transition controller. The latter two models are initially off-line trained and online adapted via structure/parameter learning. The control sensitivity matrix and the one-step-ahead predictive output of the active plant model are used to adapt the parameters of the mode transition controller such that the desired transition trajectory is tracked. The proposed adaptation scheme is illustrated for a hover to forward flight mode transition control of a helicopter encountering parametric changes and wind disturbances.

Key Words. Adaptive tracking control, fuzzy neural networks, mode transition control, kaczmarz's algorithm.

1. INTRODUCTION

Complex large-scale systems such as unmanned aerial vehicles and industrial processes are demanded to possess the intelligence required to behave in an autonomous manner under uncertain environmental conditions. Typically, these systems are required to operate in a finite number of operational modes that require robust, stable and smooth transitions between them. A (local) operational mode (or mode of operation) is considered to be a region in the system's state space in which the system exhibits quasi steady-state behavior. And a mode transition (or mode to mode) controller refers to a controller that transitions a system from a start mode of operation to the goal mode. The problem of transitioning between two operational modes can be solved by non-adaptive techniques such as gain scheduling, sliding mode control and the method of blending local mode controllers. However, when the system to be controlled differs significantly from the nominal system used in the design methods above, degraded tracking performance of the desired transition trajectory is to be expected.

The basic objective of adaptive control is to maintain consistent performance of a system in the presence of uncertainty or unknown variation in plant parameters. Typically, adaptive control is developed for MIMO linear systems, SISO nonlinear systems and for certain classes of MIMO nonlinear systems. In [2,10], adaptive controllers were developed for a class of feedback linearizable nonlinear systems. In [5], an adaptive output feedback tracking control was proposed for a class of single-input single-output nonlinear systems with uncertain differentiable time-varying parameters. Recently, adaptive techniques based upon the one-step-ahead control strategy have been developed for more general nonlinear systems. In [3,4], neural network based one-step-ahead control strategies were proposed for a class of nonlinear SISO systems. In [8], a nonlinear one-step-ahead control scheme based upon a recurrent neural network model was proposed

for nonlinear SISO processes. The neural network model was trained via a recursive least squares (RLS) algorithm and the gradient descent update rate for the control law was determined by stability consideration. Finally, for a general class of MISO nonlinear systems, an adaptive quasi-one-step-ahead control law was proposed in [6]. The control law was derived using the sensitivity between the controlled system input and output and the quasi-one-step-ahead predictive output. The sensitivity of the plant was estimated using RLS and the predictive output was obtained by a recurrent neural network.

In this paper, an adaptation scheme is proposed for the real-time adaptation of mode transition controllers designed via the method of blending local mode controllers (BLMC) [7]. The control objective is to adapt the blending gains portion of the mode transition controllers such that the nonlinear plant state vector tracks the desired transition trajectory from a start mode of operation to a goal mode. The Adaptation scheme is composed of a desired transition model, an active plant model and an active controller model which is the mode transition controller. The desired transition model, the active plant model and the blending gains portion of the active controller model are represented via a fuzzy neural network construct discussed in [9]. All three fuzzy neural models are trained off-line while the latter two models are adapted online. The active plant model which incorporates local model information is initially trained off-line to capture the desired transition trajectory and controls. Afterwards, the active plant model is online adapted via structure and parameter learning to capture the input/output behavior of the nonlinear system to be controlled. Likewise, the blending gains portion of the mode transition controller is determined off-line and is adapted online via structure and parameter learning to track the desired transition trajectory. The new blending gains to be developed by the mode transition controller is determined from the control sensitivity matrix and the one-step-ahead predictive output of the active plant model. The parameter learning for the active plant and active controller models is based upon Kaczmarz's algorithm [1].

In Section 2, a brief description of the fuzzy neural network construct used in the paper is given. The online adaptation of mode transition controllers is described in Section 3. Finally, in Section 4, the adaptation scheme proposed in this paper is illustrated for the hover to forward flight transition of a helicopter encountering parametric changes and wind disturbances.

2. FUZZY NEURAL NETWORKS

The fuzzy neural structure proposed in [9] will be considered in this section. This FNN structure consists of a fuzzy rule base of Takagi-Sugeno fuzzy rules with the rule consequents being linear polynomials of the input premise variables. Both structure learning and parameter learning is used to adaptively develop the FNN construct in [9]. The structure learning inserts new membership functions, create new fuzzy rules and select initial parameters of the new rules on the basis of the desired output data. The parameter learning updates the consequent weights via Kaczmarz's algorithm.

2.1. Structure

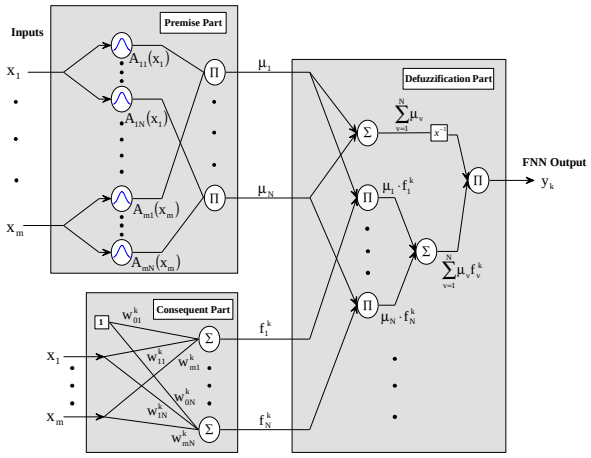

Fig. 1. General configuration of the FNN architecture

The fuzzy neural architecture proposed in [9] is divided into the premise part, the consequent part and the defuzzification part as shown in Fig. 1. The premise module partitions the premise space, assigns membership functions to each premise cell and develops the rule base of fuzzy rules. The consequent module consists of the rule consequents being linear polynomials of the input premise variables. Finally, the defuzzification module combines the firing strengths of the rules and the rule output functions to provide the final system output. Therefore, this FNN construct realizes the fuzzification, fuzzy reasoning, and defuzzification functionalities of a connectional fuzzy inference mechanism.

Let $\boldsymbol{x} = [x_1, \ldots, x_m]^T$ and $\boldsymbol{y} = [y_1, \ldots, y_p]^T$ denote the input and output vectors of the FNN, respectively. The fuzzy rule base of the FNN consists of a collection of N fuzzy rules of the form:

$$R^{(j)}: \quad IF\ x_1\ is\ A_{1j}\ AND\ x_2\ is\ A_{2j}\ AND \cdots AND\ x_m\ is\ A_{mj}$$
$$THEN\ f_j^1 = w_{0j}^1 + w_{1j}^1 x_1 + \cdots + w_{mj}^1 x_m\ AND \cdots$$
$$AND\ f_j^p = w_{0j}^p + w_{1j}^p x_1 + \cdots + w_{mj}^p x_m \quad (1)$$

where f_j^ℓ denotes the j[th] rule output associated with the ℓ^{th} output component y_ℓ. $W_{0j}^\ell,\ldots,w_{ij}^\ell,\ldots,w_{mj}^\ell$ are the polynomial coefficients connecting linearly the input variables to the f_j^ℓ consequent function. Finally, $A_{1j},\ldots,A_{ij},\ldots,A_{mj}$ are labels of the fuzzy sets in the premise space associated with the j[th] rule $R^{(j)}$. Each linguistic label A_{ij} is associated with a gaussian membership function, $m_{A_{ij}}(x_i)$ which specifies the degree to which a given x_i satisfies the quantifier A_{ij}:

$$m_{A_{ij}}(x_i) = \exp\left[-\frac{1}{2}\frac{(x_i - m_{ij})^2}{s_{ij}^2}\right] \quad (2)$$

where m_{ij} and s_{ij} denote the mean and standard deviation of the gaussian membership function. The degree of fulfillment (or the firing strength) of each rule $R^{(j)}$ is taken as:

$$m_j(x_1,\ldots,x_m) = m_{A_{1j}}(x_1)\times\cdots\times m_{A_{mj}}(x_m) \quad (3)$$

Given an input vector $\boldsymbol{x}$, the ℓ^{th} output component y_ℓ, of the fuzzy system is inferred as follows:

$$y_\ell = \frac{\sum_{j=1}^N m_j \cdot f_j^\ell}{\sum_{j=1}^N m_j}, \ell = 1,\ldots,p \quad (4)$$

The inferred outputs of Equation (4) result from the application of the weighted-average defuzzification method.

2.2. Structure Learning with Local Model Information

Structure learning using local model information is exactly same as the structure learning scheme described in [9] except that the new rules' consequent weights are initialized in the following way:

$$\left[\begin{array}{c|ccc} w_{0,N+1}^1 & w_{1,N+1}^1 & \cdots & w_{m,N+1}^1 \\ \vdots & \vdots & \vdots & \vdots \\ w_{0,N+1}^p & w_{1,N+1}^p & \cdots & w_{m,N+1}^p \end{array}\right] = \left[\begin{array}{c|c} y - \frac{\partial y}{\partial x}x & \frac{\partial y}{\partial x} \end{array}\right] \quad (5)$$

where x is the model input, y is the desired output corresponding to x and $\partial y/\partial x$ is the jacobian matrix defined at x.

2.3. Parameter Learning via Kaczmarz's Algorithm

Consider the current input/output pair of the form: $\{x^d, y^d\}$, where x^d and y^d denote the desired input and output vectors of the FNN model, respectively. Let y denote the current output of the FNN model given the input x^d. The consequent parameters for the j[th] rule are updated using the following equation:

$$W_j(t+1) = W_j(t) + \Delta W_j(t) \quad (6)$$

where $\Delta W_j = [\Delta W_j^a \mid \Delta W_j^b]$, $\Delta W_j^a = \Delta\bar{y}^j - \Delta W_j^b \bar{x}^j$, $\Delta W_j^b(t) = \frac{\{y^d - y\}}{j^T j} r_j [x^d - \bar{x}^j]^T$, $\Delta\bar{y}^j = \frac{\{y^d - y\}}{j^T j} r_j$, $j = [f_1^T \quad f_2^T \quad \cdots \quad f_N^T]^T$, $f_j^T = r_j [1 \quad (x^d - \bar{x}^j)^T]$, $r_j = m_j \Big/ \sum_{v=1}^N m_v$ and $\bar{x}^j$ is the center of the j[th] rule such that $m_j(\bar{x}^j) = 1$.

2.4. FNN Linear Incremental Model

Let $\boldsymbol{x} = [x_1, x_2, \ldots, x_m]^T$ and $\boldsymbol{y} = [y_1, y_2, \ldots, y_p]^T$ denote the input and output vectors of the FNN model depicted in Figure 1, respectively. Suppose that the FNN model has N fuzzy rules. Then, the rs[th] element of the FNN linear incremental model is:

$$\left[\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right]_{rs} = \frac{\sum_{v=1}^N m_v \left\{ (y_r - f_v^r)\frac{(x_s - m_{sv})}{s_{sv}^2} + w_{sv}^r \right\}}{\sum_\ell^N m_\ell} \quad (7)$$

3. ONLINE ADAPTATION OF MODE TRANSITION CONTROLLERS

In this section, an adaptation scheme is proposed for the online customization of mode transition controllers designed off-line via the method of blending local mode controllers. The control objective is to adapt the blending matrices such that plant output vector tracks the output vector of a desired transition model. In order to apply the discrete-time adaptation scheme to the continuous-time system, it is assumed that the sample rate has been appropriately selected. Fig. 2 shows the configuration for indirect adaptive mode transition control. The adaptation scheme is composed of five components: the *desired transition model*, the *active*

plant model, the *plant adaptation mechanism*, the *active controller model* and the *controller adaptation mechanism*.

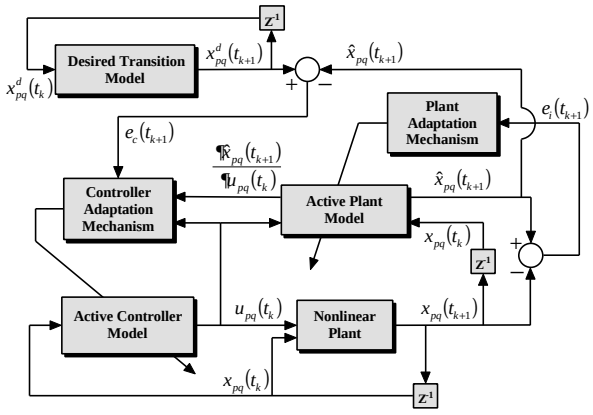

Fig. 2. Configuration for indirect adaptive mode transition control

3.1. Desired transition model

A fuzzy neural model representation of the desired transition trajectory is determined off-line.

3.2. Active plant model

A fuzzy neural model of the input/output relationship of the nonlinear plant along the desired transition is determined by off-line training. Also, the linear model information along the desired trajectory is incorporated into the consequent part of the fuzzy neural model. Afterwards, the *active plant model* is adapted online via the plant adaptation mechanism to account for plant variations on a real-time basis.

3.3. Plant adaptation mechanism

The active plant is adapted online to account for plant variations on a real-time basis. At time instant t_k, the adaptation of the active plant model is accomplished by performing structure/parameter learning on the basis of the current input/output data of the system to be controlled.

3.4. Active controller model

The active controller model is the mode controller, which is determined off-line via the BLMC approach [7]. The blending gains of the active controller model are adapted online using the *controller adaptation mechanism*.

3.5. Controller adaptation mechanism

Let ACM and APM denote the active controller model and the active plant model, respectively. Let $u_{pq}(t_k)$ be the currently developed control input by the ACM which corresponds to $x_{pq}(t_k)$. Suppose that $x^d_{pq}(t_{k+1})$ represents the desired trajectory at t_{k+1} provided by the desired transition model. The steps of the controller adaptation mechanism algorithm are:

1. Apply ACM to $x_{pq}(t_k)$ and produce the current initial estimate of the control input $u_{pq}(t_k)$.
2. Input $u_{pq}(t_k)$ and $x_{pq}(t_k)$ to APM and produce $\hat{x}_{pq}(t_{k+1})$. Calculate $\tilde{x}^d_{pq}(t_{k+1}) = x^d_{pq}(t_{k+1}) - x_{pq}(t_{k+1})$ using the predictive one-step-ahead output $\hat{x}_{pq}(t_{k+1})$ in place of the unavailable output $x_{pq}(t_{k+1})$.
3. The true control sensitivity matrix $D(x_{pq}(t_k), u_{pq}(t_k))$ is approximated via the APM's incremental control matrix $\hat{D}$, which can be calculated from Equation (7). When the APM is not sufficiently activated by the input $(x_{pq}(t_k), u_{pq}(t_k))$, the control sensitivity information contained in the strongest fired rule's consequent parameters is used to determine $\hat{D}$.
4. Compute the weighted least squares optimal control law,

$$u'_{pq}(t_k) = u_{pq}(t_k) + \left[\hat{D}^T \cdot Q \cdot \hat{D}\right]^{-1} \hat{D}^T \cdot Q \cdot \tilde{x}^d_{pq}(t_{k+1}). \qquad (8)$$

Afterwards, calculate the desired blending weights $k'_{pq}(t_k)$.

5. Train ACM to capture desired blending weights $k'_{pq}(t_k)$ given current input $x_{pq}(t_k)$. Note that parameter learning with local model information is used to train the ACM.
6. Put $t_k \leftarrow t_{k+1}$ and perform the same procedure at the next time t_{k+1}.

4. HOVER TO FORWARD FLIGHT EXAMPLE

4.1. Model of Helicopter's Forward Dynamics

The proposed adaptation scheme will be illustrated on the following model representing the longitudinal channel dynamics of an Apache helicopter constrained to have no vertical motion; only longitudinal and pitch rotation motions are allowed [7]:

$$X = X_{trim} + X_{\dot{x}}(\dot{x} - \dot{x}_{trim}) + X_q(\dot{q} - \dot{q}_{trim}) + X_{d_e}(d_e - d_{e,trim})$$

$$M = M_{trim} + M_{\dot{x}}(\dot{x} - \dot{x}_{trim}) + M_q(\dot{q} - \dot{q}_{trim}) + M_{d_e}(d_e - d_{e,trim})$$

$$\ddot{x} = \frac{X}{m \cdot \cos(q)} - g \cdot \tan(q)$$

$$\ddot{q} = \frac{M}{I_Y}$$

where $\ddot{x}$, $\ddot{q}$ and d_e represent the forward acceleration (ft/s^2), pitch angle acceleration (rad/s^2) and longitudinal cyclic input (deg), respectively. X represent the aerodynamic force along the "X axis" and M represent the pitching moment about the "Y axis". The parameters X_{trim}, $X_{\dot{x}}$, $X_{\dot{q}}$, X_{d_e}, M_{trim}, $M_{\dot{x}}$, $M_{\dot{q}}$, M_{d_e}, $\dot{x}_{trim}$, $\dot{q}_{trim}$, $d_{e,trim}$ are functions of $\dot{x}$. X_{trim} and M_{trim} are the trim values of the aerodynamic force X and the pitching moment M, respectively. The variables $X_{\dot{x}}$, $X_{\dot{q}}$, X_{d_e}, M_{trim}, $M_{\dot{x}}$ and $M_{\dot{q}}$ are the partial derivatives of X and M with respect to $\dot{x}$, $\dot{q}$ and d_e, respectively. The physical constants m and I_Y are the mass of the helicopter and the moment of inertia along the Y axis. The state vector of the helicopter model is $[x_1 \quad x_2 \quad x_3 \quad x_4]^T = [\dot{x} \quad \ddot{x} \quad q \quad \dot{q}]^T$. It is assumed that the output vector of the model is the same as the state vector.

4.2. Hover to Forward Flight Mode Controller

In [7], a hover to forward flight mode controller was designed via the BLMC approach. The controller was designed such that the closed-loop system transitions from $[0.0000 \quad 0.0000 \quad 0.1008 \quad 0.0000]^T$ to $[92.8278 \quad 0.0000 \quad 0.0402 \quad 0.0000]^T$ in minimum time with the following constraints:

$-1.0000 \le \dot{x} \le 94.0000$, $-2.5000 \le \ddot{x} \le 20.0000$

$-0.7000 \le q \le 0.7000$, $-0.6000 \le \dot{q} \le 0.6000$

$-6.5000 \le d_e \le 4.5000$.

The adaptation scheme proposed in Section 3 will be applied to the mode transition controller mentioned above. A sample time of $T_S = 0.05s$ is chosen for the adaptation scheme. The desired minimum time trajectory and control are used to train the *desired transition model* and the *active plant model*. Also, the local model information along the desired trajectory is incorporated into the consequent part of *active plant model*. The structure learning parameters used by the *controller adaptation mechanism* and the *plant adaptation mechanism* was $d = 0.2$ and $b = 0.5$ where d is the lower threshold for membership value and b is the desired overlap degree. The upper limit of the width of each membership function was $s^U = [0.20 \quad 0.20 \quad 0.05 \quad 0.05 \quad 0.20]$ for the *active plant model* and $s^U = [0.20 \quad 0.20 \quad 0.05 \quad 0.05]$ for the *active controller model.*

4.2. Simulation Results

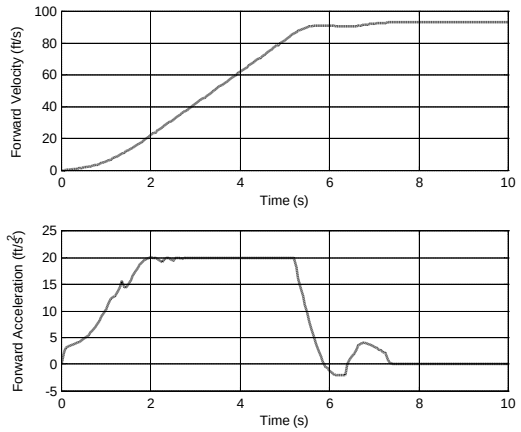

Fig. 3. Plots of desired $\dot{x}$ and $\ddot{x}$ nominal trajectories

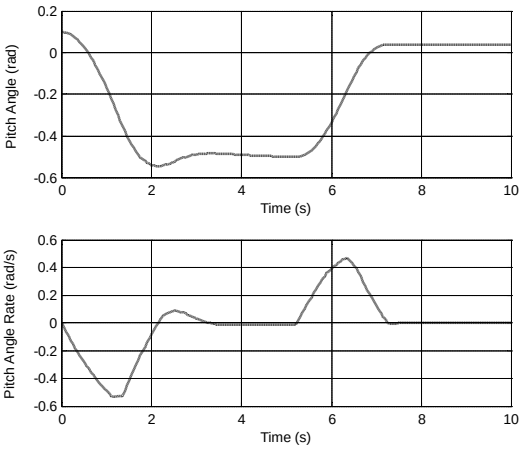

Fig. 4. Plots of $\dot{?}$ and $\ddot{?}$ nominal trajectories

Fig. 3 and 4 show the desired $\dot{x}$, $\ddot{x}$, q and $\dot{q}$ trajectories. For the nominal mode transition controller, Fig. 6 show the mean squared error from the desired transition trajectory for wind disturbances and parametric changes of X_{d_e}. For small parametric changes and wind disturbances, the controller exhibits good tracking performance of the desired transition trajectory. However, as the magnitude of the parametric changes and wind disturbances increase the tracking performance of the controller degrades. Fig. 7 show mean squared error from the desired transition trajectory for wind disturbances and parametric changes of X_{d_e}, for the least squares adaptation of the nominal mode transition controller. As expected, if the approximate plant accurately captures the local model information and the input/output behavior of the system to be controlled, the adapted controller exhibits

excellent tracking performance when encountering parametric changes and wind disturbances.

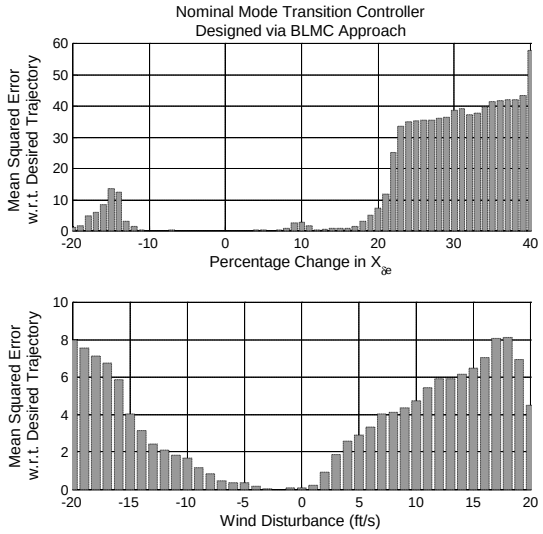

Fig. 5. Plots of mean squared errors for controller designed via BLMC approach

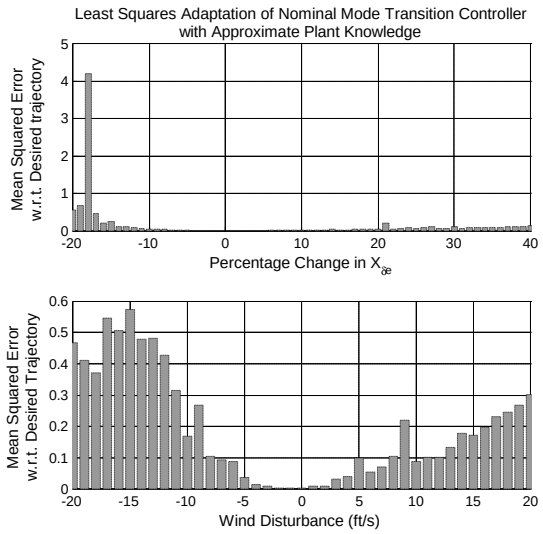

Fig. 6. Plots of mean squared errors for adaptation with approximate plant information

5. CONCLUSIONS

An adaptation scheme is proposed for the online adaptation of mode transition controllers designed via the blending local mode controllers approach. When the active plant model of the adaptation scheme is a good approximation of the system to be controlled, then it is expected that adapted controller will track the desired trajectory very well in the presence of parametric changes and disturbances. However, if the active plant model does not capture the local model and the input/output behavior of the systems to be controlled, poor tracking performance to unstable tracking can result. Further investigation is needed to add robustness capability to the adaptation scheme so that it can avoid controller faults due to poor approximation of the nonlinear plant. One way of preventing controller faults will be to decrease the aggressiveness of the scheme when approximation errors are significant and increase the aggressiveness when the errors are small.

6. REFERENCES

[1] Hunt K.J., Irwin G.R., Warwick K. Neural Network Engineering in Dynamic Control Systems, Springer-Verlag, New York, 1991, pp. 42-45.

[2] Jagannathan S. Adaptive Fuzzy Logic Control of Feedback Linearizable Discrete-Time Dynamical Systems Under Persistence of Excitation, Automatica, Vol. 34, No. 11, pp. 1295-1310, 1998.

[3] Ma X., Loh N.K. One-Step-Ahead Controller Design Using Neural Networks, Proc. American Control Conference, Vol. 2, pp. 958-962, 1992.

[4] Ma X., Loh R.N.K. Neural Network-Based Successive One-Step-Ahead Control of Nonlinear Systems, Proc. American Control Conference, Vol. 4, pp. 2129-2133, 1994.

[5] Marino R., Tomei P. Adaptive Output Feedback Tracking Control for Nonlinear Systems With Time-Varying Parameters, Proceedings of the IEEE Conference on Decision and Control, Vol. 3, pp. 2483-2488, 1997.

[6] Mingzhong L., Fuli W. Adaptive Control of Black-Box Nonlinear Systems Using Recurrent Neural Networks, Proceedings of the IEEE Conference on Decision and Control, San Diego, California, pp. 4165-4170, 1997.

[7] Rufus F., Clements S., Sander S., Heck B., Wills L., Vachtsevanos G. Software-Enabled Control Technologies For Autonomous Aerial Vehicles, 18th Digital Avionics Systems Conference, Vol. 2, pp. 6.A.5-1 - 6.A.5-8, 1999.

[8] Tan Y., Cauwenberghe A.V. Nonlinear One-Step-Ahead Control Using Neural Networks: Control Strategy and Stability Design, Automatica, Vol. 32, No. 12, pp. 1701-1706, 1996.

[9] Theocharis J., Vachtsevanos G. Adaptive Fuzzy Neural Networks as Identifiers of Discrete-Time Nonlinear Dynamic Systems, Journal of Intelligent and Robotic Systems, Vol. 17, pp. 119-168, 1996.

[10] Zhang T., Ge S.S., Hang C.C. Neural-Based Direct Adaptive Control for a Class of General Nonlinear Systems, International Journal of Systems Science, Vol. 28, No. 10, pp. 1011-1020, 1997.