EMRANN Controller for Aircraft Auto-Landing under Actuator Failure

Sudesh Kumar Kashyap and Girija Gopalarathnam

***Abstract*— Dynamic Radial Basis Function Neural Network (RBFNN) called Extended Minimum Resource Allocation Neural Network or EMRANN is used to aid a Baseline Trajectory Following Controller (BTFC) for achieving better fault tolerance for aircraft auto-landing under severe wind and actuator failures. EMRANN has an inbuilt mechanism to generate or prune the hidden layer neurons based on their contribution to system output, uses on-line training and does not need prior training. The performance of the EMRANN controller is demonstrated using simulated data for single and multiple actuator faults. The results show that the segmentation of BTFC with EMRANN controller improves the ability to handle large faults and also meet the strict touchdown requirements.**

***Index Terms*— BTFC, Radial Basis Function, EMRANN, NDI, Wind Shear, Dryden Model**

I. INTRODUCTION

In any Unmanned Aerial Vehicle (UAV), the most critical part of flight is auto-landing and control system designed using classical technique gives satisfactory performance in terms of meeting a desired flight path for auto-landing and stringent touchdown conditions under healthy control surfaces. The classical controllers coupled with Instrument Landing Systems (ILS) or Global Positioning System (GPS) have limited ability to cope with control surfaces being stuck at certain deflections. It is found that the performance of these controllers can be enhanced to handle faults of large magnitude by using Artificial Intelligence (AI) technique such as Neural Networks (NNs) to augment the classical controllers. This is possible because the NNs have the inherent ability to learn on-line and adapt to sudden changes such as severe wind, control surface failure and sensor failures.

Many examples of application of neural network for control can be found in open literature. Ref. [1] reports the use of an off-line trained neural network used in conjunction with conventional Proportional Integrator Derivative (PID) controller to generate desired trajectories for landing under wind disturbances. In [2], on-line neural network controller is designed for autopilot and Stability Augmentation Systems (SAS) for both the longitudinal and lateral directional dynamics. An Adaptive critic based feed-forward neural network technique has been reported in [3] for aircraft auto-landing in a specified touchdown region in the presence of gust. In [4], neural network controller is trained off-line for auto-landing of aircraft and its performance is found to be similar to that of PID controller. The pseudo control hedging technique proposed in [5] modifies the control command signals when control surfaces are close to their saturation limits. The disadvantage of feed-forward based neural network scheme is that it requires *a prior* training on both normal and faulty operating data and also size in terms of hidden layer neurons of network has to be fixed beforehand. Hence, these networks are not suitable for handling new type of failures since they are not trained to handle them.

Radial Basis Function Neural Network (RBFNN) has a simple network architecture, good local interpolation and global generalization abilities [6] as compared to Multilayer Feed-Forward Neural Networks (MFNN). In terms of time to converge, MFNN with back propagation training algorithm is often too slow whereas in case of RBFNN it is comparatively fast since network can establish its parameters related to hidden neurons directly from the input data.

A variant of RBFNN named MRANN (Minimum Resource Allocation Neural Network) with sequential/on-line learning (not trained *a priori*) was first introduced by Lu [7] with adaptive structure wherein neurons are added or deleted dynamically based on the input data thereby maintaining a compact network. MRANN has been used for several applications varying from function approximation to nonlinear system identification and in flight control. In this paper, an upgraded version of MRANN named Extended Minimum Resource Allocation Neural Network (EMRANN [8]) is used in conjunction with the classical controller (named Baseline Trajectory Following Controller or BTFC) for auto-landing of high performance fighter aircraft subjected to severe wind and ineffective control surfaces due to stuck actuators. EMRANN which has better speed in terms of convergence because only those neurons (called nearest) are updated which have more contribution in the final output.

Figure 1 shows the top view of architecture for aircraft Auto-landing using BTFC and EMRANN controller [9]. Based on aircraft current position in inertial frame, the tracking command generator generates the reference commands such

Sudesh Kumar Kashyap is with the Flight Mechanics and Control Division, National Aerospace Laboratories, Bangalore, India (phone: 91-080-25086536; fax: 91-080-25298293; e-mail: sudesh@nal.res.in)

Girija Gopalarathnam is with Flight Mechanics and Control Division, National Aerospace Laboratories, Bangalore, India (e-mail: ggirija@nal.res.in).

as desired total velocity, altitude, heading angle and tracking error to steer the aircraft to a desired flight path for auto-landing. BTFC in the inner loop is designed using conventional method (root locus) under the no-fault condition but subjected to severe wind. BTFC is not only used to stabilize the overall system but also provides the error signals to train the EMRANN on-line. EMRANN controller aids the BTFC under failure conditions by learning the aircraft inverse by observing total signal to the actuators and comparing it to its output. The details of waypoints, wind simulation and landing requirements are discussed in section II. The details of aircraft model can be found in section III of the paper. Section IV presents the design aspects of BTFC. Section V of the paper covers the basics of radial basis function network and the design EMRANN controller. Conclusion with future work is drawn in section VI of the paper.

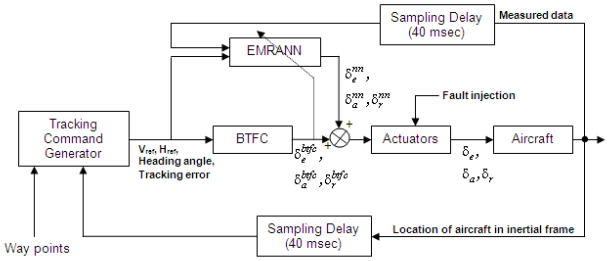

Fig.1: BTFC/EMRANN for Auto-Landing

II. AUTO-LANDING REQUIREMENTS

Figure 2 shows the desired flight path that has to be followed by aircraft for auto-landing.

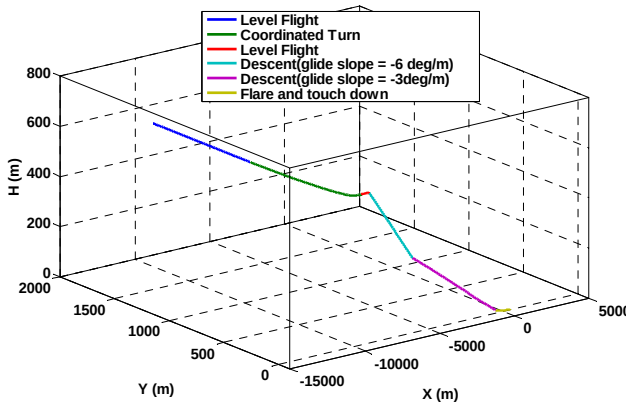

Fig. 2: Different Phases of an Auto-Landing

Table 1 summarizes the way points of flight trajectory. As shown in figure 1 Tracking Command Generator (TCG) computes the reference signals (ψ_{ref} – ***heading***, δ – ***tracking error***, V_{ref} - ***Velocity*** and , H_{ref} – ***Height***) which steers the aircraft to a desired flight path (see figure 2) while tracking error tends to zero. TCG generates the signals during the various phases of flight based on current location of aircraft and way points defined by x, y, z and V.

Table 1: Auto-landing waypoints

Way points index	Phases							x(m)	y (m)	z(m)	V (m/s)
1	1							-10766	1730	600	82.88
2		2						-10766	900	600	82.88
3			3					-9866	0	600	82.88
4				4				-9351	0	600	82.88
5					5			-6496	0	300	82.88
6						6		-1000	0	12	82.88
7							7	0	0	0	76.19
8								234	0	0	76.19

It can be seen that there are basically two types of trajectory geometry either straight line (in level and flare phases) or arc of circle (coordinated turn). Figure 3 shows the waypoints and trajectory geometry for straight line and circular arc. The equations used by TCG to generate reference signals can be found from [11].

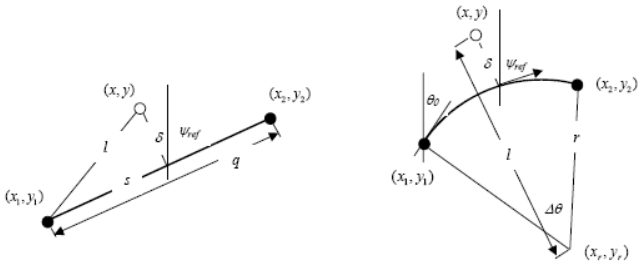

Fig. 3: Way Points and Geometry of Straight Line and Circular Arc

In present study, aircraft is subjected to severe wind disturbances causing auto-landing even more challenging. At low altitude the most commonly encountered wind disturbances are wind shear and turbulence. Wind shear is characterised as a change in wind speed and direction within a short distance. Microburst or downburst is a wind shear in a vertical direction. Turbulence is assumed to exist only in horizontal direction and is modelled using a Dryden spectrum. The vertical component of wind i.e. Microburst is defined as:

$$w_w = -w_0\left(1+\frac{ln\left(\frac{h}{510}\right)}{ln(51)}\right) \tag{1}$$

The above equation is valid only when height h above ground level is less than 150 meter and above that height Microburst vanishes. Below 150 meter, constant w_0 is equals to 12 m/sec when h is above h_{shear} (91 meter) otherwise it is -12 m/sec. The forward component of wind i.e. turbulence is defined by

$$u_w = u_{g1} + u_{gc} \tag{2}$$

$$\dot{u}_{g1} = 0.2\left|u_{gc}\right|\sqrt{2a_u}N_1 - a_u u_{g1} \tag{3}$$

where, u_{g1} is turbulence component, u_{gc} is the mean wind and N_1 is the Gaussian random noise with zero mean and variance of 100. The variables u_{gc} and a_u are computed as

$$a_u = \begin{cases} \dfrac{U_0}{100h^{1.5}} & h > 70m \\ \dfrac{U_0}{600} & h \le 70m \end{cases} \quad (4)$$

$$u_{gc} = \begin{cases} -u_0\left(1 + \dfrac{ln\left(\dfrac{h}{510}\right)}{ln(51)}\right) & h \ge 3m \\ 0 & h < 3m \end{cases} \quad (5)$$

where, U_0 and u_0 are 72 m/sec and 6 m/sec, respectively.

The side component of wind is represented as

$$v_w = \begin{cases} -10 & h < 190m \\ 10 & 190 \le h < 470m \\ 0 & h \ge 470m \end{cases} \quad (6)$$

Figure 4 shows the profile of wind components during various phases of auto-landing under healthy conditions. It can be seen from the figure that most critical part (second descent with glide slope of – 3deg/m and flare) of auto-landing is between altitudes of 0-150 meter at which all the three components of wind disturbance are present at the same time.

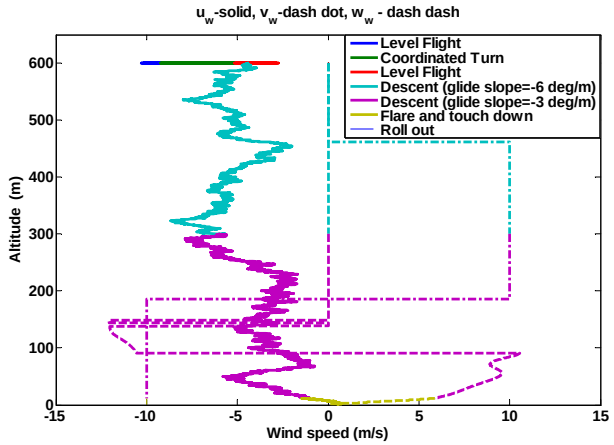

Fig. 4: Wind Components during Auto-landing

A. *Performance Criteria*

The performance criteria is defined in terms of i) vertical and lateral deviations of aircraft from the desired flight path and ii) rise time, settling time, over-shoot and steady state error of aircraft's response such as altitude, heading angle, airspeed and flight path angle. The performance criteria for lateral direction is given by

$$\Delta y = \begin{cases} 5 & h \le 30m \\ 5 + \dfrac{15(h-30)}{120} & 30m < h \le 150\,m \\ 20 & h > 150m \end{cases} \quad (7)$$

The performance criteria for vertical direction is given by

$$\Delta h = \begin{cases} 0.5 & h \le 30m \\ 0.5 + \dfrac{4.5(h-30)}{120} & 30m < h \le 150\,m \\ 6 & h > 150m \end{cases} \quad (8)$$

B. *Safety Criteria*

The desired touchdown point of aircraft under normal condition is x=0, y=0 and z=0. There are some safety criteria that need to be satisfied during final phase of auto-landing under the severe wind and actuator fault. These safety criteria are as follows:

- Flight path angle γ should be less than 0.7 degree
- Absolute value of bank angle ϕ should be less than 10 deg
- Total airspeed V_T should be greater than 60 m/s
- Maximum distance from touchdown point along y-axis should not be more than ± 5 meter

C. *Robustness and Control Activity Criteria*

The system for auto-landing should be robust to the variation of certain parameters such as Centre of Gravity (CoG), mass, moment of inertia and time delay. In present study, robustness evaluation is carried out against the following parameters:

- Lateral variation in CoG from 25% to 45% of mean aerodynamic chord
- Mass variation within $\pm 20\%$ of actual
- $\pm 10\%$ variation in moment of inertia, I_{xx}
- Sampling delay up to 100 milli second (see figure 1)

In terms of control activity criteria, the mean actuator rates for all the control surfaces should be less than 33% of maximum rates and mean throttle rate should not be greater than 15% of the maximum rate.

III. AIRCRAFT MODEL

In present paper a high performance aircraft is considered whose characteristic is similar to the F-16 aircraft. The aerodynamic coefficient based 6 DoF model of an aircraft with spilt control surfaces (left and right ailerons and elevators) is realized in MATLAB/SIMULINK® environment. The equation of motion with reference to body-fixed axis is derived by assuming the aircraft is a rigid body. The elevator and aileron aerodynamic data has been split into two parts

corresponding to left and right control surfaces using Computational Fluid Dynamics (CFD) computations. Table 2 contains the control surfaces deflection and actuator rate limits.

Table 2: Deflection and Rate Limits

Surface	Surface deflection limit		Rate limit of actuator
	min	max	
Ailerons	-20 deg	20 deg	60 deg/s
Elevators	-25 deg	25 deg	60 deg/s
Rudder	-30 deg	30 deg	60 deg/s

Details of aircraft dynamics and equation of motions can be found in [11]. Table 3 shows the details of aircraft characteristic.

Table 3: Deflection and Rate Limits

Wing-span, b	9.144 meter (m)
Mean chord length, $\bar{c}$	3.45 m
Wing area, S	27.87 m^2
Mass, m	9295.4128 Kg
Engine angular momentum, H_e	216.9 Kg-m^2/sec
Center of gravity location (from leading edge), x_{cg}	$0.35\bar{c}$ m
Reference center-of-gravity location (from leading edge) for aerodynamic data, x_{cgr}	$0.35\bar{c}$ m
Moment of inertia about body-axis x, I_x	12875 Kg-m^2
Moment of inertia about body-axis y, I_y	75674 Kg-m^2
Moment of inertia about body-axis z, I_z	85552 Kg-m^2
Cross moment of inertia, I_{xy}	0 Kg-m^2
Cross moment of inertia, I_{yz}	0 Kg-m^2
Cross moment of inertia, I_{xz}	1331 Kg-m^2
Maximum deflection angle of speed brake, δ_{sb}	60 deg.
Maximum deflection angle of leading edge flap, δ_{lef}	25 deg.

IV. Design of Baseline Trajectory Following Controller

In case of aircraft auto-landing, it is desirable that the aircraf follows the reference command signals like total airspeed altitude above ground, heading w.r.t. true north, etc. This ca be achieved by designing an appropriate feedback contro system. The most reliable and proven method to desig control system is based on root-locus analysis wherein th feedback gain is selected such that all the poles of a close loo system remain on LHS of s-plane and at same time the desire performance is met in terms of peak response, settling time damping, etc.

The control design based on root locus is applicable to SISO (single-input single-output) linear time-invariant system only. In present case BTFC is designed using root locus based approach and using MATLAB functions such as: 'series', 'feedback', 'rlocus' and 'bode'. Under this approach, a linear model is obtained from aircraft's 6 DoF non-linear equation of motion about the equilibrium points or steady state conditions using small perturbation theory. These equilibrium or operating points are obtained for given flight condition (altitude and Mach number) using MATLAB® based function 'trim'. Trim states and corresponding control deflections for that flight conditions are as follows:

- $\boldsymbol{u = 81.31}$ m/s
- $\boldsymbol{v = 0}$ m/s
- $\boldsymbol{w = 16.08}$ m/s
- $\boldsymbol{p = q = r = 0}$ rad./s
- $\phi = \boldsymbol{0}$ rad.
- $\theta = \boldsymbol{0.19528}$ rad.
- $\psi = \boldsymbol{0}$ rad.
- $\boldsymbol{h = 600}$ meter
- $\delta_e = \boldsymbol{-0.01117}$ rad.
- $\delta_a = \delta_r = \boldsymbol{0}$ rad.
- $\delta_{thr} = \boldsymbol{0.22}$ (between 0 and 1)

Figure 5 shows the responses of non-linear and linear models for doublet excitation given to left elevator. The design of BTFC is carried out for longitudinal and lateral motions separately. The BTFC for longitudinal motion controls the aircraft to follow the reference signals such as altitude and velocity through elevators and throttle, respectively, whereas BTFC for lateral motion controls the aircraft to follow the reference heading through ailerons. The controllers are designed for worst-case delay of 40 millisecond (twice the controller sampling interval), introduced in the loop. It is also assumed that angle of attack and sideslip angles are not available for the feedback.

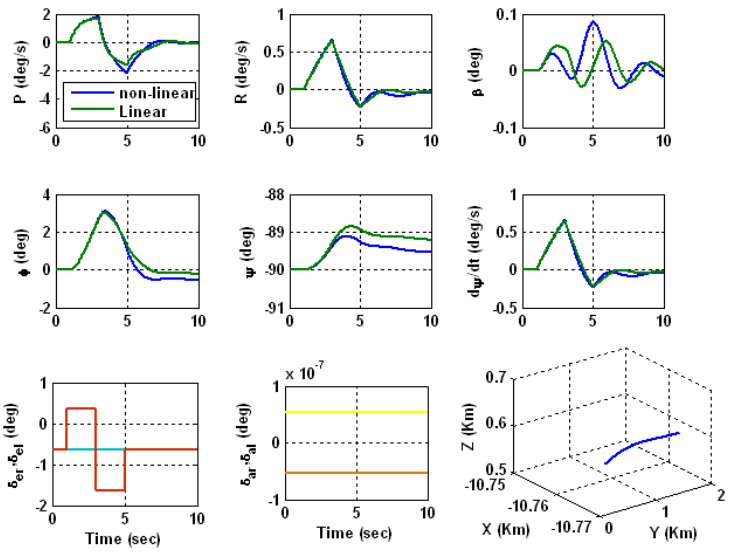

Fig. 5: Responses of non-linear and linear models

Figures 6-7 shows the BTFC for longitudinal and lateral-directional modes respectively. The aircraft model used in present study has primarily five control surfaces. The elevators control the pitch motion by operating in identical fashion. If one control surface fails and moves to hard over position then the other healthy surface is used to control the aircraft. Similarly, ailerons are used to control the roll motion by operating in differential mode. The rudder is used to control the yaw motion of the aircraft. It is observed from the control input matrix (B) that the elevator failure can cause significant roll moment and it is about 60% effective as the

ailerons in creating roll moment. On other hand the ailerons are not that much effective to create significant pitching moment. So it can be concluded that in case of ailerons failure, elevators can be used to control the roll moment of the aircraft in addition to pitch control. In present study the control surface failures are injected just before coordinated or level turn (somewhere between 8-10 second) thereby introducing maximum demand on control system.

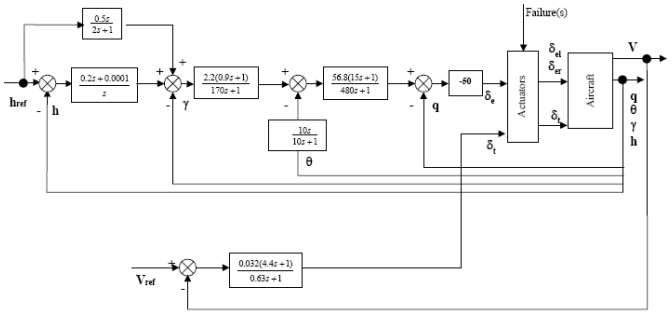

Fig. 6: BTFC for longitudinal motion[11]

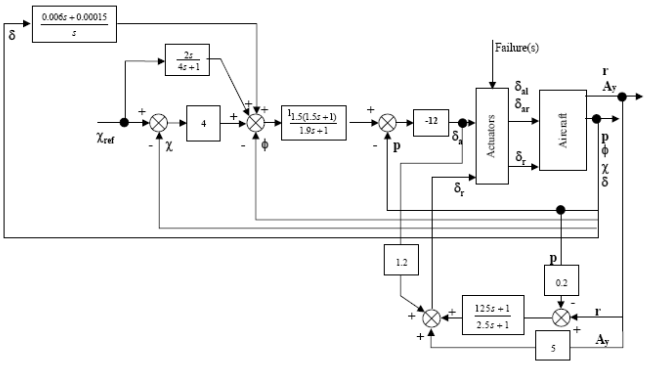

Fig. 7: BTFC for lateral motion[11]

It is seen from the simulation that BTFC is not able to handle the single elevator alone failure for not more than ± 3 deg. Figures 8-9 show the simulated response of BTFC subjected to left alone elevator stuck to -3 deg at 10th second. It can be seen from figure 9 that the lateral deviation is within desired limit (non-shaded portion) for most of the time, thereby, meeting one of the performance criteria.

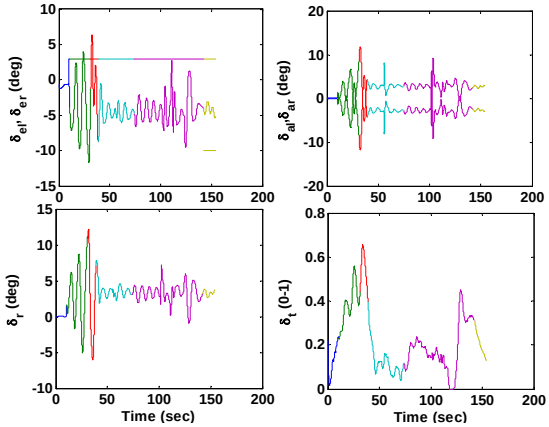

Fig. 8: Control inputs – left elevator stuck at -3 deg at 10th second

As a part of robust and control activity criteria, BTFC is also evaluated with respect to different seed number used in wind simulation, mass variation (-20% to 20%) and CG shift (30% to 40% of Mean Aerodynamic Chord). Under the healthy condition BTFC is robust to variations of above parameters, whereas under the failure cases it is observed that the success rate of touchdown reduces to 70% in case of seed number whereas in case of mass and CG variations the success rate is very less.

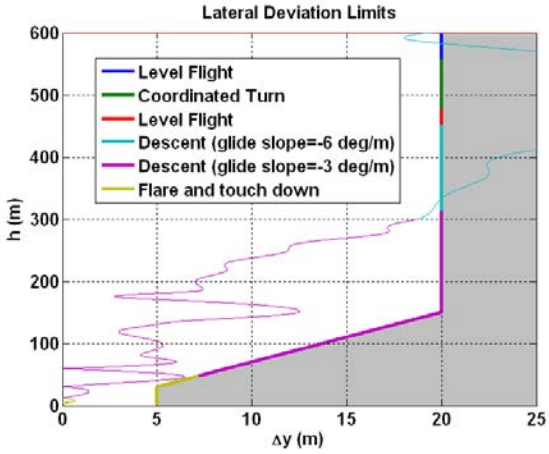

Fig. 9: Lateral Deviation–left elevator stuck at -3 deg at 10th second

V. RBFNN AND EMRANN CONTROLLER

Figure 10 shows the structure of RBFNN with Gaussian function as one type of radial basis function. The RBFNN response at each of its output node is given by

$$y_k = b_k + \sum_{j=1}^{n} \alpha_{kj} \phi_j(U); k = 1,...,p \tag{9}$$

$$\phi_j(U) = exp\left(\frac{-\|U - \mu_j\|^2}{\sigma_j^2} \right) \tag{10}$$

where, $U = [u_1,...,u_m]$ is the input vector to network and $\|.\|$ denotes Euclidean norm.

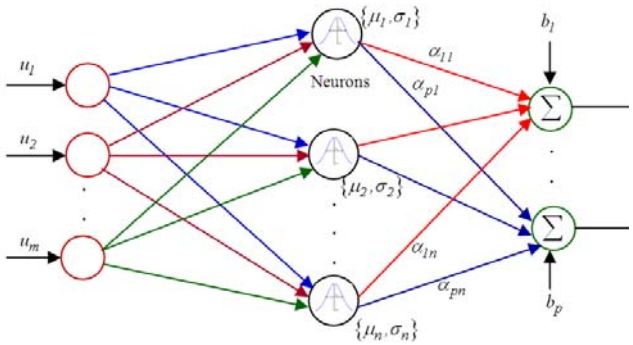

Fig. 10: RBFNN with Gaussian Function

m - inputs, n-hidden neurons, p-outputs, α -weights, b – bias, μ / σ - center/width of Gaussian function

MRANN is RBFNN type of network. The beauty of this type of network is that it starts with no hidden neurons i.e. $n = 0$ and $y_k = b_k$ with $k = 1,...,p$, where $\boldsymbol{p}$ is the number of outputs (nodes). The network output vector y_k is compared with corresponding measured response vector $\boldsymbol{y_k^m}$ of aircraft and the difference (error vector) is used in various criteria to decide whether to add or delete.

Criteria to add new neurons: new hidden neuron is created if the all the following conditions are satisfied simultaneously:

i) $\boldsymbol{e^i} = \left\| \left(\boldsymbol{y_k^m} - \boldsymbol{y_k} \right)_{\boldsymbol{i}} \right\| \geq \boldsymbol{E_1}$, where ||.|| is Euclidian norm and $\boldsymbol{E_1}$ is the threshold

ii) $\boldsymbol{e^i_{rms}} = \sqrt{\dfrac{\boldsymbol{1}}{\boldsymbol{s_w}} \sum_{\boldsymbol{j=i-s_w+1}}^{\boldsymbol{i}} \left\{ \left(\boldsymbol{y_k^m} - \boldsymbol{y_k} \right)^{\boldsymbol{T}} \left(\boldsymbol{y_k^m} - \boldsymbol{y_k} \right) \right\}_{\boldsymbol{j}}} \geq \boldsymbol{E_2}$, where '$\boldsymbol{i}$' is the current observation index or time, $\boldsymbol{s_w}$ is the window length over which root mean square (RMS) error is computed and $\boldsymbol{E_2}$ is the threshold

iii) $\boldsymbol{e^i_{nearest}} = \left\| \left(\boldsymbol{U} - \mu_{\boldsymbol{nearest}} \right)_{\boldsymbol{i}} \right\| \geq \boldsymbol{max}\left(\boldsymbol{e_{max}} . \gamma^{\boldsymbol{i}}, \boldsymbol{e_{min}} \right)$, where, $\mu_{\boldsymbol{nearest}}$ is the vector consisting centre of those Gaussian functions (from any of existing hidden neurons) closest to current input vector $U = [u_1,...,u_m]$, $\boldsymbol{0} < \gamma < \boldsymbol{1}$ is the decay factor and $\boldsymbol{e_{min}}, \boldsymbol{e_{max}}$ are the minimum and maximum threshold, respectively.

Parameters for new neuron: whenever, the above conditions are satisfied, a new hidden neuron is added to the network. The following changes are required in the network to adapt a newly added neuron:

- Total number of neurons now: n=n+1
- Centre vector: $\mu_{n+1} = U_i$
- Width: $\sigma_{\boldsymbol{n+1}} = \kappa \left\| \left(\boldsymbol{U} - \mu_{\boldsymbol{nearest}} \right)_{\boldsymbol{i}} \right\|$, where κ is an overlap factor
- Weight vector: $\alpha_{\boldsymbol{n+1}} = \left[\boldsymbol{y_1^m} - \boldsymbol{y_1}, ..., \boldsymbol{y_p^m} - \boldsymbol{y_p} \right]$

Criteria to delete neurons: Since every hidden neuron is connected to each of output nodes, deletion of hidden neuron is only possible if its contribution is consistently minimum (below some threshold) among the other hidden neurons to the outputs for a long time, say over the window length of $\boldsymbol{n_w}$. Let us assume the neuron under the observation is $\boldsymbol{j^{th}}$ among total of $\boldsymbol{n}$ neurons so its contribution to $\boldsymbol{k^{th}}$ output (node) at $\boldsymbol{i^{th}}$ instant is given by

$$o_{kj}^i = \alpha_{kj} \phi_j \left(U_i \right) = \alpha_{kj} \, exp\left(\frac{-\left\| U - \mu_j \right\|^2}{\sigma_j^2} \right) \tag{11}$$

For network with multiple outputs, o_{kj}^i can be represented in vector form:

$$o_j^i = \alpha_j \phi_j \left(U_i \right) = \alpha_j \, exp\left(\frac{-\left\| U - \mu_j \right\|^2}{\sigma_j^2} \right) \tag{12}$$

Each element of o_j^i is examined continuously over sample of $\boldsymbol{n_w}$ and $\boldsymbol{j^{th}}$ neuron deleted only when it is found that all the elements o_j^i have insignificant contribution to overall value of the network outputs. In order to avoid any inconsistency possible due to absolute value of the outputs, element o_{kj}^i is normalised by the following factor:

$$o_{k,max}^i = max\left(o_{k,1}^i, ..., o_{k,n}^i \right) \tag{13}$$

Thus the normalised output vector is given by

$$n_j^i = \frac{o_j^i}{o_{k,max}^i} \tag{14}$$

$\boldsymbol{j^{th}}$ neuron deleted if each element of n_j^i is less than threshold δ over a window length $\boldsymbol{n_w}$.

Updating existing neurons

If the criteria to add new hidden neuron are not satisfied at current instant $\boldsymbol{i}$, then the parameters of existing neurons are updated using following formula of Extended Kalman Filter (EKF):

$$\boldsymbol{X_i} = \boldsymbol{X_{i-1}} + \boldsymbol{K_i e^i} \tag{15}$$

where, the current value of parameters of neurons is given by vector $\boldsymbol{X_i} = \left[\alpha_{\boldsymbol{0}}, \{\alpha_{\boldsymbol{1}}, \mu_{\boldsymbol{1}}, \sigma_{\boldsymbol{1}}\}, ..., \{\alpha_{\boldsymbol{n}}, \mu_{\boldsymbol{n}}, \sigma_{\boldsymbol{n}}\} \right]_{\boldsymbol{i}}$ with total number of elements equal to $\boldsymbol{s} = \boldsymbol{p} + \left(\boldsymbol{p} + \boldsymbol{m} + \boldsymbol{1} \right) \boldsymbol{n}$ and $\boldsymbol{K_i}$ is the Kalman gain given by

$$\boldsymbol{K_i} = \boldsymbol{P_{i-1}} \boldsymbol{H_i^T} \left[\boldsymbol{H_i} \, \boldsymbol{P_{i-1}} \boldsymbol{H_i^T} + \boldsymbol{R} \right]^{-1} \tag{16}$$

where, variable $\boldsymbol{R}$ is the variance of measurement noise of size $\boldsymbol{p}$ x $\boldsymbol{p}$ with zero off-diagonal elements and $\boldsymbol{P}$ a positive definite symmetric state error covariance matrix of size $\boldsymbol{s}$ x $\boldsymbol{s}$ is updated using:

$$\boldsymbol{P_i} = \left[\boldsymbol{I_{s \times s}} - \boldsymbol{K_i H_i} \right] \boldsymbol{P_{i-1}} + \boldsymbol{q I_{s \times s}} \tag{17}$$

where, $\boldsymbol{H_i} = \nabla_{\boldsymbol{X}} \boldsymbol{y_k^i}$ is the gradient vector/matrix (or observation vector/matrix with size $\boldsymbol{p} \times \boldsymbol{s}$) of the function $\boldsymbol{y_k^i}$ (eq. (9)) with respect to parameters $\boldsymbol{X}$ evaluated at

$\boldsymbol{X_{i-1}}$ and $\boldsymbol{q}$ is a scalar that determines the allowed random step in the direction of the gradient vector.

$$\boldsymbol{H_i} = \begin{bmatrix} \boldsymbol{I_{p\times p}}, \phi_1(\boldsymbol{U_i})\boldsymbol{I_{p\times p}}, \phi_1(\boldsymbol{U_i})\dfrac{2\alpha_1^i}{(\sigma_1^i)^2}(\boldsymbol{U_i}-\mu_1^i)^T, \phi_1(\boldsymbol{U_i})\dfrac{2\alpha_1^i}{(\sigma_1^i)^3}\|\boldsymbol{U_i}-\mu_1^i\|^2 \\ \ldots, \phi_n(\boldsymbol{U_i})\boldsymbol{I_{p\times p}}, \phi_n(\boldsymbol{U_i})\dfrac{2\alpha_n^i}{(\sigma_n^i)^2}(\boldsymbol{U_i}-\mu_n^i)^T, \phi_n(\boldsymbol{U_i})\dfrac{2\alpha_n^i}{(\sigma_n^i)^3}\|\boldsymbol{U_i}-\mu_n^i\|^2 \end{bmatrix} \quad (18)$$

Whenever a new neuron is added to hidden layer of the network, following changes are required in EKF to accommodate the new neuron:

- Dimension of state $\boldsymbol{X}$ should be increased by $(\boldsymbol{p}+\boldsymbol{m}+\boldsymbol{1})$ i.e.

$$\boldsymbol{s} = \boldsymbol{p}+(\boldsymbol{p}+\boldsymbol{m}+\boldsymbol{1})\boldsymbol{n}+(\boldsymbol{p}+\boldsymbol{m}+\boldsymbol{1}) = \boldsymbol{p}+(\boldsymbol{p}+\boldsymbol{m}+\boldsymbol{1})(\boldsymbol{n}+\boldsymbol{1})$$

- State error covariance should be $\boldsymbol{P_{new}} = \begin{bmatrix} \boldsymbol{P_{old}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{p_0 I_{(p+m+1)\times(p+m+1)}} \end{bmatrix}$, where $\boldsymbol{p_0}$ is an estimate of the uncertainty in the initial values assigned to the parameters $\boldsymbol{X}$. The difference between MRANN and EMRANN is that all the parameters ($\boldsymbol{X}$) are updated in case of MRANN whereas in EMRANN parameters corresponding to nearest neurons are updated.

The idea of having neural network based controller is to expand the fault-tolerant envelope of BTFC for auto-landing. These neural network based controllers approximate the Non-Linear Dynamic Inversion (NDI) controller online by neural learning schemes. There are the two versions of neural controller (EMRANN) used in present study:

- The feedback error learning strategy – EMRANN 1
- Estimation before control strategy – EMRANN 2

Robust NDI controllers are designed based on integrator back stepping method. Consider the following equations representing a second order short period aircraft dynamics:

$$\dot{\boldsymbol{q}} = \boldsymbol{f_1}(\boldsymbol{q}, \alpha, \delta_e) \quad (19)$$

$$\dot{\alpha} = \boldsymbol{f_2}(\boldsymbol{q}, \alpha) \quad (20)$$

Here it is assumed that functions $\boldsymbol{f_1}, \boldsymbol{f_2} \in \boldsymbol{R}$ are smooth and their first derivative is bounded in the neighbourhood of the trajectory. In back steeping algorithm, first thing is to compute the desired pitch rate $\boldsymbol{q_d}$ computed by inverting eq. (20) for given α and desired angle of attack rate $\dot{\alpha}_d$, i.e.

$$\boldsymbol{q_d} = \boldsymbol{f_2^{-1}}(\dot{\alpha}_d, \alpha) \quad (21)$$

where, $\dot{\alpha}_d = \boldsymbol{G}_\alpha(\alpha_d - \alpha)$ and $\boldsymbol{G}_\alpha$ is the feedback gain and α_d is the desired angle of attack.

For given $\boldsymbol{q}$ and α, the desired elevator deflection is computed by inverting eq. (19)

$$\delta_e = \boldsymbol{f_1^{-1}}(\boldsymbol{q_d}, \alpha, \dot{\boldsymbol{q}}_d) \quad (22)$$

where, $\dot{\boldsymbol{q}}_d = \boldsymbol{G_q}(\boldsymbol{q_d} - \boldsymbol{q})$ and $\boldsymbol{G_q}$ is the feedback gain and $\boldsymbol{q_d}$ is the desired pitch rate computed using eq.(21).

EMRANN based scheme learns the inverse of plant and thereby driving the output of BTFC to zero. A properly tuned neural network basically reconstructs the inverse functions in the region in which BTFC is stable. Based on the learned information, it can enhance the stability region by a process of extrapolation. In EMRANN 1 only those parameters (such as neurons centres, widths and weights) are updated which are within a given radius of the current input [11]. Figures 11-12 show the schematic of EMRANN 1 controller. The various tuning parameters of the networks are obtained using optimization technique such as Genetic Algorithm under the variety of hard over failure positions of the control surface with a condition of successful auto-landing to be met [11].

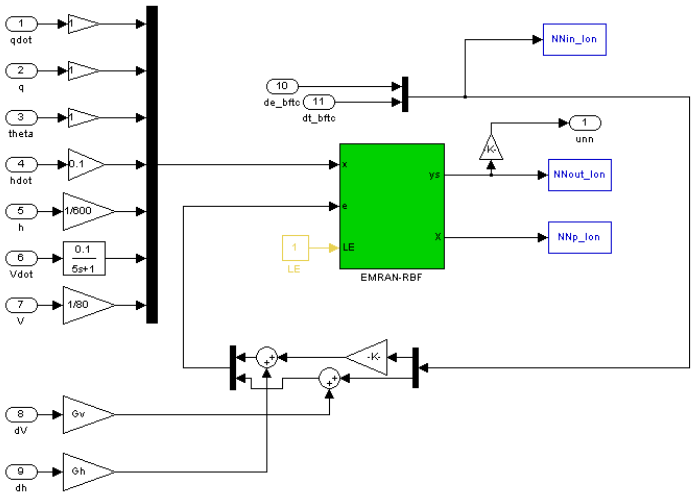

Fig. 11: EMRANN 1 for Longitudinal Motion

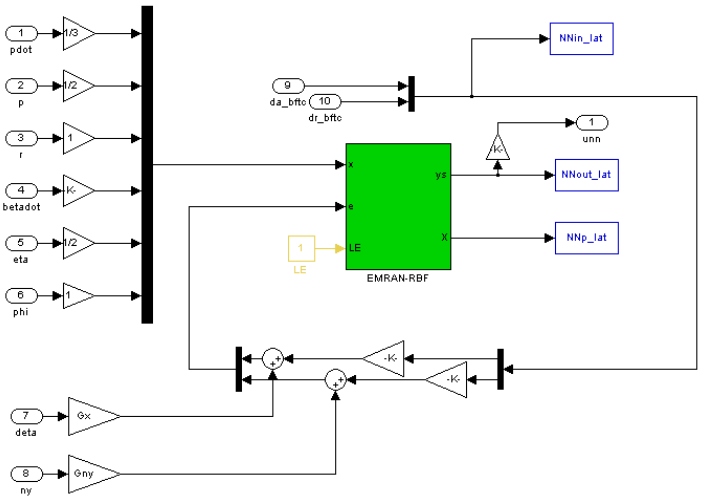

Fig. 12: EMRANN 1 for Lateral-directional Motion

Table 4 shows the final tuned parameters of these networks.

Table 4: Optimised EMRANN 1 Parameters [11]

Parameters	Initial Value		Converged Value	
	Longitudinal	Lat-directional	Longitudinal	Lat-directional
E_1	0.21069	0.06357	0.21069	0.06356
E_2	0.08734	0.09024	0.08734	0.09024
ϵ_{min}	0.05310	0.09359	0.05310	0.09358
ϵ_{max}	0.20668	0.40775	0.20668	0.40778
r	0.89074	0.65236	0.89086	0.65236
r_χ	1.39137	0.22171	1.39137	0.22171
p	0.12503	**0.0000**	0.12503	**0.0076**
R	**2.1444**	10.000	**12.3137**	10.000
η	0.003536	0.002585	0.003536	0.002585
δ	1e-6	1e-6	1e-6	1e-6
G_h	0.0050	-	0.0050	-
G_V	0.0896	-	0.0896	-
G_χ	-	0.9994	-	0.9994
G_{ny}	-	**0.0608**	-	**0.0931**
Cost Function	338		229	

The performance of EMRANN 1 and BTFC is compared for stuck left elevator at –10 deg at 10th second of flight.

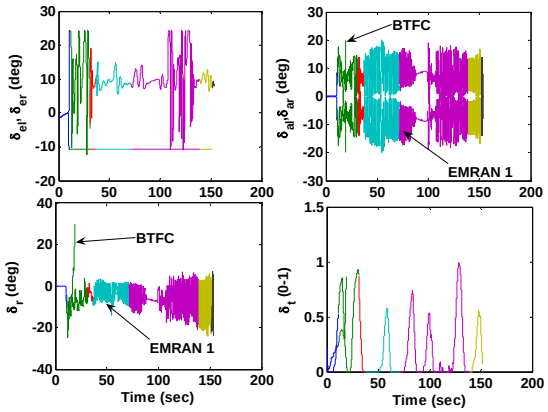

Fig. 13: Control inputs – left elevator stuck at -10 deg at 10th second

Figure 13 shows that in case of EMRANN 1, how the deflections of control surfaces are adjusted once the failure is detected for auto-landing. Figure 14 shows the comparison of simulated response of BTFC and EMRANN 1 which indicate that BTFC is clearly not able to handle the failure and aircraft crash whereas EMRANN 1 successfully handles the failure.

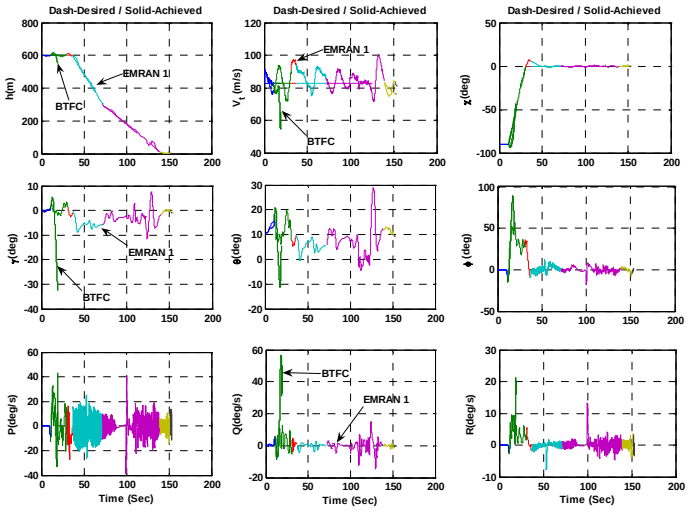

Fig. 14: Simulated Response of BTFC and EMRANN 1 – left elevator stuck at -10 deg at 10th second

In case of EMRANN 1, failure tolerance was improved by adding the scaled trajectory error (i.e. $\Delta h, \Delta V, \Delta\chi$) to their respective scaled control input from BTFC. In case of EMRANN 2, a more rational approach is used to enhance the failure tolerance of BTFC. This approach is known as estimation before control strategy wherein trajectory simultaneously asymptotically converges to the desired trajectory. The other difference of EMRANN 2 from EMRANN 1 is that former performs its estimation calculation using different output compared to its control calculation. The estimated control deflection is used to compute the error (e.g. $\boldsymbol{error} = \delta_e^{btfc} - \hat{\delta}_e^{nn}$ as compared to $\delta_e^{btfc} - \delta_e^{nn}$) which drives the network towards the convergence asymptotically. The other output of EMRANN 2 for e.g. δ_e^{nn} is added to BTFC output and resultant signal ($\delta_e^{btfc} + \delta_e^{nn}$) controls the surface deflections. Figures 15-16 show the schematic of EMRANN 2.

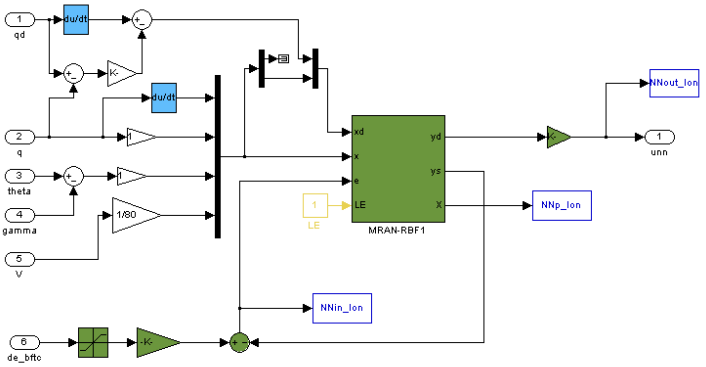

Fig. 15: EMRANN 2 for Longitudinal Motion

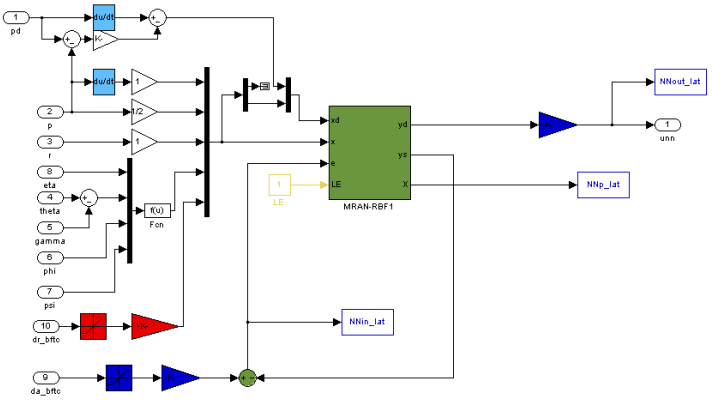

Fig. 16: EMRANN 2 for Lateral-directional Motion

In case of longitudinal motion, reference signal q_d used in EMRANN 2 is nothing but the reference input to the inner most loop of the BTFC. Similar is the case of reference signal roll rate p_d. In case of lateral-directional EMRANN 2, one of

the inputs is sideslip angle $\hat{\beta}$ computed using $\hat{\beta} = \chi + \hat{\alpha}\phi - \psi$, where $\hat{\alpha} \approx \theta - \gamma$.

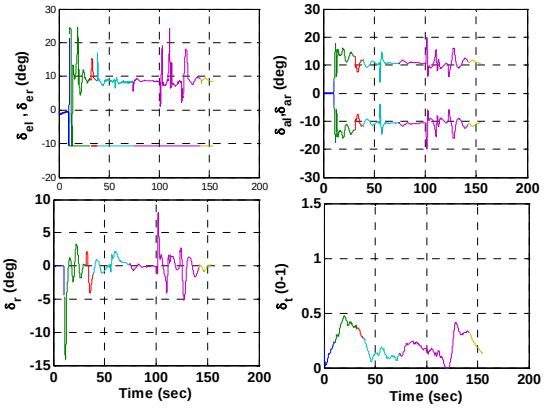

Fig. 17: Control inputs – left elevator stuck at -10 deg at 10th second

It can be seen from figure 17 that additional rolling moment introduced due to stuck left elevator is compensated through ailerons (left +ve / right –ve). In additional to that it can also be seen that the control surface deflections are smooth as compared to those in case of EMRANN 1. As compared to EMRANN 1 the lateral deviation from the desired trajectory is less in case EMRANN 2. Same is the case of body rates response and total velocity (fig. 18).

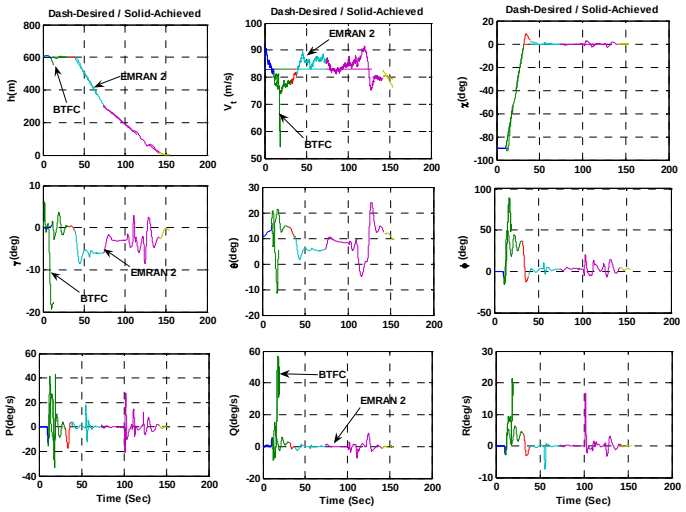

Fig. 18: Simulated Response of BTFC and EMRANN 2 – left elevator stuck at -10 deg at 10th second

VI. Conclusion

BTFC, EMRANN 1 and EMRANN 2 are used for Auto-landing under severe wind and actuator fault. The neural network schemes enhance the stability region of BTFC by learning inverse of plant. The more rational approach is EMRANN 2 which guarantees the asymptotic convergence of trajectories. Also the success rate of EMRANN 2 is higher as compared to EMRANN 1.

Acknowledgment

Authors acknowledge Prof. N. Sundarajan, School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore for the mathematical model and Dr. Abhay Pashilkar, Senior Scientist, FMCD, NAL, Bangalore for providing SIMULINK model of the aircraft, BTFC and EMRAN and technical support.

References

[1] C. C. Jorgensen and C. Scheley, “Neural Network Basedline Problem for Control of Aircraft Flare and Touchdown”, Neural Network for Control, Cambridge, MA: MIT Press, pp.402-425, 1990

[2] M.R. Napolitano and M. Kincheloe, “On-line Learning Neural-Network Controllers for Autopilot Systems”, AIAA Journal of Guidance, Control and Dynamics, Vol. 33, No.6, pp. 1008-1015, 1995

[3] G. Saini, and S. N. Balakrishnan, “Adaptive Critic Based Neurocontroller for Autolanding of Aircrafts with Varying Glideslopes ”, Proceeding on International Conference on Neural Networks, Vol. 4, pp. 2288-2293, 1997

[4] Y. Iiguni, H. Akiyoshi and N. Adachi, “An Intelligent Landing System Based on a Human Skill Model”, IEEE Transaction on Aerospace and Electronic Systems, Vol. 34, No.3, pp. 877-882, 1998

[5] N. Kim, A. J. Calise, N. Hovakimyan and J. V. R. Prasad, “Adaptive output Feedback for High-Bandwidth Flight Control”, AIAA Journal of Guidance, Control and Dynamics, Vol. 25, No.6, pp. 993-1002, 2002

[6] N. Sundararajan, P. Saratchandram and Lu Ying Wei, “Radial Basis Function Neural networks with Sequential Learning: MRAN and its Applications”, Progress in Neural Processing, Vol. 11, ISBN 981-02-3771-5, 1999

[7] Y. Lu, N. Sundararajan and P. Saratchandran, “A Sequential Learning Scheme for Function Approximation using Minimal Radial Basis Function Neural Networks”, Neural Computation, Vol. 9, pp. 461-478, 1997

[8] Y. Li, N. Sundararajan and P. Saratchandran, “Analysis of Minimum Radial Basis Function Network Algorithm for Real-time Identification of Non-Linear Dynamic Systems”, IEE Proceeding on Control Theory Applications, Vol. 147, No. 4, pp. 478-484, 2000

[9] H. Gomi, M. Kawato, “Neural network control for a closed-loop system using feedback-error-learning”, Neural Networks, Vol. 6, No.7, pp. 933-946,1993

[10] Laut T. Nguyen, Marilyn E. Ogburn, William P. Gilbert, Kemper S. Kibler, Phillip W. Brown and Perry L. Deal, Simulator study of Stall/Post Characteristics of a Fighter Airplane With Relaxed Longitudinal Static Stability, NASA Technical Paper 1538, 1979.

[11] Nikhil S. Naikal, Rohit Panikkar, Abhay A. Pashilkar and R. Nagaraj, "Improved Fault Tolerance for Autolanding Using Adaptive Backstepping Neural Controller", IEEE Conference on Control Applications - CCA2007, Suntec City Convention Centre Singapore, October 1-3, 2007

[12] Richard S. Russel, Non-Linear F-16 Simulation using Simulink and MATLAB, University of Minnesota, Version 1.0 June 22, 2003

Dr. Sudesh K. Kashyap received his PhD degree titled “Decision fusion using fuzzy logic” from University of Mysore and M.E. degree in Electrical engineering with specialization in Automatic Control and Robotics from M.S. University of Baroda. Currently, he is working at National Aerospace Laboratories, Bangalore as scientist since 2002. His areas of interest are Kalman filtering, sensor fusion, Fuzzy logic, Bayesian theory and neural network.

Dr. (Mrs.) Girija G, received her Ph.D. from Bangalore University in 1996. She is presently the Deputy Head of the Flight Mechanics and Control Division at National Aerospace Laboratories (NAL), Bangalore. Her areas of research are: modeling, parameter estimation of aerospace vehicles, multi sensor data fusion and Enhanced and Synthetic Vision systems