Detection of Additive Sensor Faults in an Unmanned Air Vehicle (UAV) Model using Neural Networks

Ihab Samy*, Ian Postlethwaite and Dawei Gu*****

*Engineering Department, Leicester University, PhD student UK (e-mail:isar1@le.ac.uk).
**Engineering Department, Leicester University, Head of Control and Instrumentation Group, UK (e-mail:ixp@leicester.ac.uk).
***Engineering Department, Leicester University, Professor, Control and Instrumentation Group, UK (e-mail:dag@leicester.ac.uk).

Abstract: Sensor fault detection and isolation (SFDI) is an already well-established field where popular methods, such as unknown input observers, are mostly based on linear time-invariant system equations. New approaches consider neural networks (NNs) as nonlinear system approximators and make use of their online learning capabilities to adapt to time-varying systems. This paper aims to contribute to this field with the application of NNs to a nonlinear unmanned air vehicle (UAV) model. A Radial-Basis Function (RBF) NN trained online with Extended Minimum Resource Allocating Network (EMRAN) algorithms is chosen for modelling purposes due to its good estimation capabilities and compact size. To improve SFDI robustness to unknown inputs we also propose a novel approach which is referred to as residual padding. False alarms and missed faults are found to be avoided with a slight increase in fault detection time in comparison to a conventional residual generator.

Keywords: Sensor fault detection and accommodation, neural networks, unmanned air vehicles, residual generation and evaluation.

1. INTRODUCTION

Fault tolerant flight control systems (FTFCS) can be found in many air-vehicles nowadays. Their general purpose is to detect, isolate, and accommodate for critical failures that occur on-board the vehicle. Studies on the causalities by the US Air Force during the Vietnam War have revealed that upto 70% of aircraft losses could have been avoided if FTFCS were properly designed and implemented (An, 1998). The three recognized classes of faults are actuator, process, and sensor faults (Chen and Patton, 1999). In this paper we focus on sensor faults in an unmanned air vehicle (UAV).

Traditionally, sensor fault detection and isolation (SFDI) is based on physical (hardware) redundancy and simple limit value checking. The former involves a voting scheme and the latter involves checking sensor measurements against set tolerances. It is well accepted that these approaches have several drawbacks such as high costs, the inability to accurately isolate the fault and slow fault detection times. For this reason, model-based SFDI (analytical redundancy) has attracted a wide variety of research interest over the past four decades resulting in several well established techniques, e.g. observers and parity space methods. For a detailed survey of such methods the reader is referred to Chen and Patton (1999).

The majority of model-based SFDI methods rely on linear time-invariant (LTI) models. In highly nonlinear systems, LTI models can sometimes fail to give satisfactory results (Simani *et al.*, 2003). Consequently, there have been attempts to extend existing SFDI methods to nonlinear system theory. An example is the nonlinear observer (Chen and Patton, 1999). Unfortunately these methods can be quite difficult to implement in practice. As an alternative, the use of neural networks (NNs) as nonlinear system approximators has seen an increase in SFDI applications over the past years (Isermann and Balle, 1997). Recent publications in this field include Campa *et al.* (2002), Perla *et al.* (2004), Heredia *et al.* (2005), Capriglione *et al.* (2007).

Few however have been extended to fixed-wing UAVs. Model-based SFDI methods are purely software based and therefore cost less to implement and unlike physical redundancy approaches they do not have any weight implications. This makes them an invaluable SFDI solution for UAVs. Moreover, current trends in UAV design have shown that cheap and low-weight UAVs are more likely to be accepted by the civil aviation industry (Franchi, 2005).

Chow and Willsky (1984) first defined model-based FDI to consist of two main stages; residual generation and residual evaluation. Patton *et al.* (1989) also outlined the criteria for selecting a suitable FDI approach, two of which were low false alarm rates and fewer missed faults. In general, a residual is the difference between the model estimate and the measured signal and is ideally non-zero only when a fault is present. However in real applications, the residual will always be non-zero due to unknown inputs (e.g. measurement noise and disturbances etc.). This can increase the risk of false alarms especially if simple threshold logic is implemented for residual evaluation. Of course one can raise the threshold but

this can also increase the number of missed faults. Ways to improve model-based FDI robustness to unknown inputs is a widely studied topic. Examples include adaptive thresholds, originally proposed in Emami-Naeini *et al.* (1988), or the application of alternative residual evaluation techniques which do not rely on simple threshold logic, e.g. statistical tests on the Kalman filter innovation sequence (Mehra and Peschon, 1971). Another famous approach is the unknown input observer proposed in Watanabe and Himmelblau (1982), which attempts to decouple the effects of unknown inputs on the residual. Alternatively, we suggest a novel approach in section 3.2, which we will refer to as residual generation, padding and evaluation (RGPE).

In this paper we aim to apply a NN-based sensor fault detection and accommodation (SFDA) scheme to a nonlinear UAV model. Only additive sensor faults are considered in the pitch gyro. The NN structure chosen is the Extended-Minimum Resource Allocating Network (EMRAN) Radial Basis Function (RBF) due to its compact size and good estimation characteristics (Li *et al.*, 2000). Comparisons between a typical residual generation and evaluation (RGE) approach and the RGPE are also carried out under different levels of measurement noise and fault types. All tests are carried out in Matlab/Simulink.

The paper is organised as follows; Section 2 introduces the general SFDA structure, section 3 compares the RGE and RGPE methods, section 4 discusses the UAV model and the NN structure, section 5 outlines the tests carried out with a brief explanation of the sensor fault types, sections 6, 7 and 8 will then discuss and conclude the results obtained.

2. GENERAL SFDA OUTLINE

In this paper SFD is divided into two parts (Fig 1):

1. NN modelling
2. Residual generation, padding and evaluation (RGPE)

Measured signals from the real system are used to train (build) the NN model in the first part. These can be any signals except the one being estimated. So for example if SFDA is designed for the pitch gyro then the NN can be trained with any measurements (e.g. speed, throttle settings) except pitch rate. Note that the estimation error feedback (Fig 1) during the NN modelling stage is necessary for NN training.

The second part of SFD is responsible for generating, padding and evaluating the residual (RGPE). Residual evaluation includes a pre-defined threshold which when exceeded triggers a fault alarm. If a fault is detected, NN training is then stopped to avoid learning faulty data which would be seen in the feedback to the NN model.

Once a fault is detected, NN estimates can then replace the faulty measurements where required in the real system (fault accommodation). So for example if the pitch gyro is faulty, the pitch rate NN estimates can instead be used in the flight control system where required. This completes SFDA.

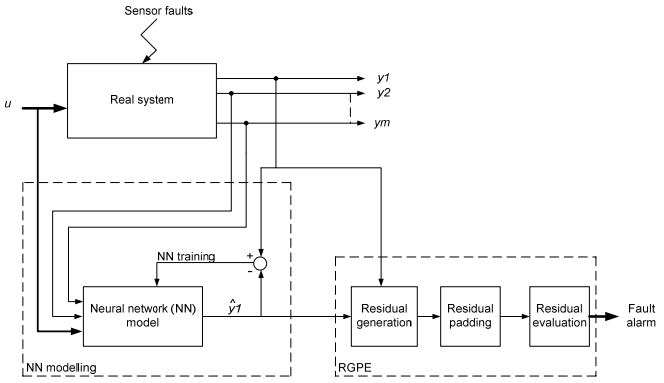

Fig 1 Sensor fault detection structure for faults in *y1*.

3. RESIDUAL

Traditionally a residual is generated and then evaluated to detect a fault in the system. We will refer to this as residual generation and evaluation (RGE). In this paper we suggest an intermediate stage which pads the residual prior to evaluation (Fig 1). We will refer to our approach as RGPE.

3.1 RGE

One residual data point is generally the squared difference between model estimate $\hat{y}$ and real sensor output y (Heredia *et al.*, 2005). To filter the residual, the weighted average of the past M points is taken, generating the following residual:

$$r_i = w \sum_{j=i-(M-1)}^{i} \frac{(y_j - \hat{y}_j)^2}{M} \tag{1}$$

where w is the weight chosen from experience and i is the sampling instant. Ideally, (1) is only non-zero when a fault is introduced. Residual evaluation would then simply involve a low threshold which when exceeded triggers a fault alarm.

3.2 RGPE

This method aims to further reduce the effects of unknown inputs on the residual in comparison to the conventional approach described in section 3.1. It involves the same steps as the RGE method. The only difference is that the residuals are now padded with artificial data, i.e. the M points in (1) are extended with padding data before the weighted average is calculated.

Let us think of the residual as n sets (**S1, S2**…**Sn**) with p data points in each set. Each data point is the squared difference between the model estimate and real sensor output (as we described in section 3.1). With padding, we manage to extend each of the n sets with artificial data. Padding maps a length p set to a length $p_{pad}>p$ set, where p_{pad} - p artificial data are added. So for example if the residual sets have 5 data points each (i.e. p=5) and we want to add 2 artificial padding points; then each set will now have 7 data points, i.e. p_{pad} =7.

Each artificial data point added equals the *minimum* magnitude (min{**Sn**}) in the n^{th} set. So for example if set **S1** is {1 0 3 4} then the padding points to be added to **S1** will have

a value of zero. The aim is to reduce the overall average of each residual set so that the effects of unknown inputs are minimised. One could rightly ask whether padding would also dampen the effects of faults. However to try to avoid this, we choose min{**Sn**} instead of 'zeros' because the latter will also result in faults being dampened while min{**Sn**} is more specific to the individual residual sets. An example will illustrate this.

Consider this simple example which demonstrates the benefits of residual padding. The following assumptions are made (no units are used):

- A residual set when no faults or unknown inputs are present is {0 0 0 0}.
- A fault gives a *continuous* residual magnitude of 6.
- An unknown input gives a residual magnitude of 24.
- Only two padding points can be added to the residual set, each with magnitudes equal to min{**set**}.
- The residual threshold has a magnitude of 5.

Let us consider three scenarios; scenario 1 is when faults and unknown inputs are not present, scenario 2 is when only a fault is present and scenario 3 is when only an unknown input is present.

Scenario 1: The residual set is {0 0 0 0} with an average of zero even if padding points are added. Therefore the threshold is not exceeded as desired.

Scenario 2: The residual set is {6 6 6 6} and so the padding points to be added would be of magnitude 6. The padded residual becomes {6 6 6 6 *6 6*} with an average of 6 and so the threshold is exceeded and a fault alarm is triggered.

Scenario 3: The residual set is {24 0 0 0}. If padding is not applied and the simple average of this residual is taken, the average would be 6 and so the threshold is exceeded resulting in a false alarm. However if instead we pad the residual set using the same technique as scenario 2, the residual set becomes {24 0 0 0 *0 0*}. The average is now 4 and therefore the threshold is not exceeded.

This is obviously a simple example with no consideration to scenarios such as drifting faults or adjacent unknown input patterns; however the underlying concepts are the same. Residual padding attempts to damp unknown inputs and not the faults.

Furthermore, if the unknown inputs are sufficiently padded, the residual can be amplified without the risk of increasing the number of false alarms. Amplifying the residual makes the detection of incipient faults easier and therefore reducing the number of missed faults. Incipient faults are slowly developing or small faults which are generally difficult to detect (Patton *et al.*, 1989).

However it is also crucial that excessive padding is avoided as it can eventually damp the faults or increase the fault detection time. This is discussed in section 7.

4. UAV AND NN MODEL

4.1 UAV Model

As the tests carried out are simulation based, the real system in Fig 1 is replaced with a nonlinear UAV model. The UAV is powered by a single jet engine and is controlled with conventional control surfaces; two ailerons, two elevators and a rudder. Coupling between longitudinal and lateral dynamics was assumed to be negligible and only the longitudinal motion of the aircraft was considered. No control laws are implemented (i.e. model is open-loop) and flight data is collected for different elevator demands (with a 3-2-1-1 input pattern) and throttle settings. The model features both system and measurement noise where the system noise is modelled as zero mean, white, Gaussian gust disturbances on the angle of attack and sideslip with a 0.1 deg standard deviation. The measurement (sensor) noise is also assumed to be Gaussian and white.

4.2 NN Model

The NN structure chosen is the EMRAN RBF proposed in Li *et al.* (2000) which consists of a group of input signals, a hidden layer and an output layer (Fig 2). It starts with zero hidden units (neurons) and only adds hidden units if *all* of the following three criteria are met (for a single output NN):

$$e_i = | y_i - \hat{y}_i | \qquad > E1 \tag{2}$$

$$e_{iRMS} = \sqrt{\sum_{j=i-(M-1)}^{i} \frac{e_j^2}{M}} \qquad > E2 \tag{3}$$

$$d_i = \| \mathbf{x}_i - \mu_{ir} \| \qquad > E3 \tag{4}$$

where y_i and $\hat{y}_i$ are the real sensor output and NN estimation at sample instant i respectively, and μ_{ir} is the centre of the hidden unit *closest* to the current NN input vector $\mathbf{x}_i$. *E1* ensures that the NN estimation error is below a set threshold (i.e. good enough), *E2* checks if the root mean square (RMS) of the past M errors is low enough and *E3* checks if the minimum distance between the current input vector and the centres of the hidden units is significantly small.

If less than three of the criteria shown in (2)-(4) are met, then the NN training algorithm updates (i.e. tunes) the network parameters (centres, widths and weights) of *only* the most active (so called 'winner') neuron. This reduces the number of parameters to be updated which speeds up the training process. An additional reason why the EMRAN RBF NN is superior to many other NN designs is that it can automatically remove hidden units which contribute the least to the NN estimates. This maintains a minimum structure size (Li *et al.*, 2000).

4.3 NN Inputs

Sensor faults are only considered in the pitch gyro and so the NN is only required to estimate UAV pitch rate. The NN

inputs must therefore include the longitudinal flight parameters. These consist of:

- Pitch rate (q)
- Angle of attack (α)
- Normal-acceleration (a_z)
- Airspeed (V)
- Altitude (H)
- Elevator angle (δ_e)
- Throttle (τ)

The input a_z was replaced by a_z / V^2 to smoothen the non-linear estimations (Campa *et al.*, 2002).

Including all the longitudinal flight parameters outlined above in the NN input set ensures accurate pitch rate estimations but can also result in a large network size. To avoid this we carried out a separate test to find out which flight parameters have the least effect on the NN output. The resulting NN can be described by the following input/output relation:

$$q_i = \mathrm{NN}\left[\ \alpha_i, \delta_{ei}, V_i, a_{zi} / V_i^2\ \right] \tag{5}$$

where NN is the NN model and i is the sampling instant. Note also that pitch rate is not included in the NN input set as explained in section 2 and Fig 1.

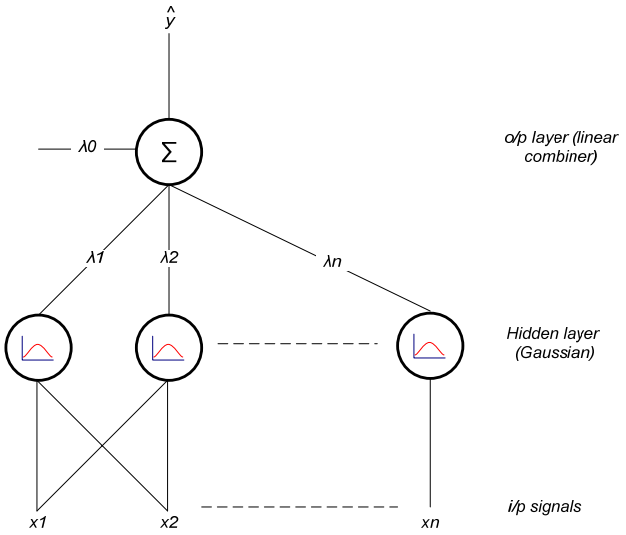

Fig 2 Fully connected RBF NN. x, λ, $\hat{y}$ are inputs, weights and output respectively.

5. SFDA APPLICATION TO THE UAV MODEL

5.1 Sensor Fault Scenarios

Flight data sensors can fail in several ways. Due to limited space, we consider *additive* faults in the pitch gyro. These faults influence the sensor outputs by an addition of a term. They are described by ramp functions and can be of soft or hard nature depending on the duration of the ramp T_R. Typically T_R is 1s for hard additive faults, and 4s for soft additive faults (An, 1998).

5.2 NN Offline Training

Prior to any SFDA tests, the NN structure must be initialised. This is done through offline training where one data set from the UAV model is used to build the NN structure. However one must be careful not to over-train the NN or else it will perform poorly when exposed to novel data.

To avoid overtraining the NN, two sets of flight data of 225s each were collected from the UAV model:

1. Train set: This is used to train the NN
2. Test set: This is used to query the NN, i.e. with learning switched off.

Offline training continues until the point where the NN RMS estimation error from the test set reaches a minimum (Fig 4).

5.3 SFDA Test Procedure

Once the NN structure is initialised (offline training), it can be used in the SFDA tests with online learning, i.e. run in parallel with the UAV model with continuous training (Fig 1).

The SFDA tests are as follows:

1. Configuration 1 (Fig 3): Divided into two main groups. The first considered hard additive sensor faults, and the second was for soft additive sensor faults. Each group is further divided into two subgroups where fault magnitudes of 20% and 5% are considered. For each fault magnitude the fault was randomly introduced at different times (*t1, t2,…tn*). Fault detection times for each test are then recorded. Note that the fault magnitudes are a % of the pitch gyro sensor range. All flight data was corrupted with additive zero mean, white, Gaussian measurement noise.
2. Configuration 2: Same as configuration 1, but with 2x standard deviation of the measurement noise.
3. Configuration 3: Same as configuration 1, but with 10x standard deviation of the measurement noise.

The mean detection times (MT), number of undetected faults (UD) and mean false alarm (FA) percentage were recorded for each configuration. The false alarm percentage is calculated by recording the total time the residual remains above the set threshold prior to any faults. The three configurations are implemented for both the RGE and RGPE methods (Table 1).

All flight data was normalised between 0-1 prior to use in the NN, and *E1*, *E2, E3* were set at 0.001, 0.01 and 0.3 respectively (2)-(4). A NN offline learning rate of 0.04 and an online learning rate of 0.0007 were found most suitable.

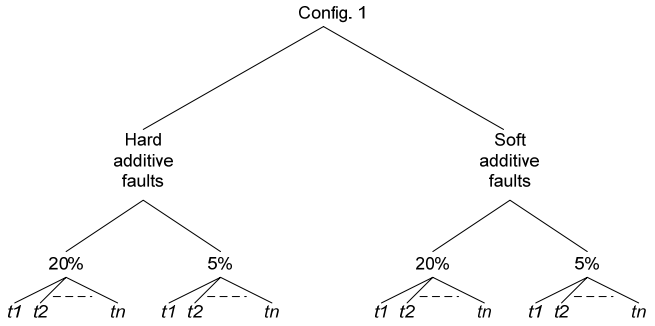

Fig 3 Test outline for configuration 1.

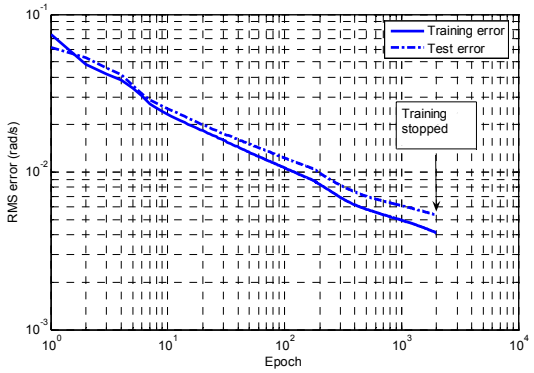

Fig 4 Offline training error history.

6. RESULTS

6.1 Offline Training

Fig 4 shows the NN RMS estimation error history during offline training where 1 epoch represents one pass through the entire data set. After 1981 epochs (47 mins CPU time) offline training was stopped to avoid overtraining the NN structure.

6.2 SFDA

Table 1 shows that in general the conventional RGE approach has a faster fault detection time than the RGPE approach. This is expected, as pointed out in section 3.2, because residual padding inevitably dampens the faults and not just the unknown inputs. On average the RGPE method caused a 55% increase in the MT.

On the other hand, the RGPE has a 0% FA and no undetected faults (Table 1). To demonstrate this, consider Fig 5 where an example of the residual for one of the tests is shown for RGE and RGPE. In this example the fault was introduced at 615s. It can be seen that the RGE method has several false alarms while the RGPE method has none. Therefore the unknown inputs are successfully dampened.

Note also that the threshold setting for the RGPE method is higher (0.006 rad^2/s^2). This is because residual amplification was made possible without the risk of increasing the number of false alarms. Consequently, incipient faults can be detected unlike the RGE method which had 2 undetected faults per configuration (Table 1).

Once the fault is detected, NN learning is switched off and the NN estimates are compared to the ideal (non-faulty) measurements. This assesses the fault accommodation properties. Fig 6 shows an example where the fault is introduced at 615s and detected 1.17s later. We can see how the NN estimates closely follow the ideal measurements after the fault is detected. The average of the RMS NN pitch rate estimation error was found to be 0.39 deg/s, 0.42 deg/s and 0.47 deg/s for sensor noise configurations 1, 2 and 3 respectively.

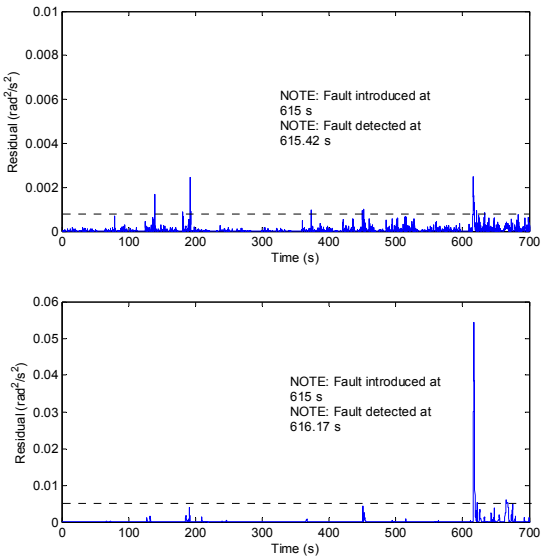

Fig 5 Residual plots. Top: RGE approach, Bottom: RGPE approach. Fault type is hard-additive with 20% magnitude. Configuration 1 settings.

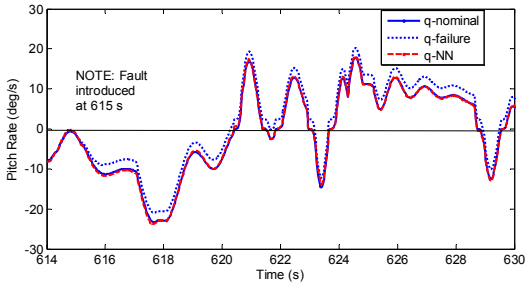

Fig 6 Fault accommodation. (q-nominal: non-faulty pitch gyro, q-failure: faulty pitch gyro, q-NN: NN estimate of pitch rate). Configuration 1 settings.

Table 1. Fault detection results

	RGE			RGPE		
	MT(s)	UD	FA(%)	MT(s)	UD	FA(%)
Config. 1	0.72	2	0.10	1.30	0	0
Config. 2	0.77	2	0.09	1.32	0	0
Config. 3	0.79	2	0.14	1.53	0	0

7. DISCUSSION

In order to dampen (pad) the unknown inputs effectively, more padding points must be added to the residual. Unfortunately by doing so, we run the risk of increasing the fault detection times or even completely damping the faults. A trade-off is therefore necessary. In our tests each residual set was padded with the same number of residual points. So a residual set with p data points would have p padding (artificial) points added to it. As a result, the average fault detection times (MT) were slightly increased in comparison to the conventional RGE approach, but we also managed to remove any false alarms and avoid any missed faults (Table 1).

The RGE approach on average failed to detect two faults (Table 1). These were found to be soft additive faults with 5% sensor range. It is generally accepted that such sensor fault types are more difficult to detect (Patton *et al.*, 1989). This is because they result in small residuals which do not exceed the set threshold. Simply lowering the threshold or amplifying the residual as a solution would also increase the false alarm rate. However with the RGPE approach the residual was sufficiently amplified without causing any false alarms (Fig 5). Therefore in general the RGPE approach was more robust to unknown inputs and sensitive to incipient faults in comparison to the conventional RGE approach. The main drawback is the increase in fault detection time. However the amount of padding can be tuned so that this increase is kept within acceptable limits (generally dictated by the user). This is seen by the fact that the MTs in Table 1 are still reasonably low.

The choice of the NN learning rate is crucial as a high learning rate guarantees learning of new flight operating conditions but it can also cause poor global approximation capabilities which are necessary for fault accommodation. An offline and online learning rate of 0.04 and 0.0007 respectively, were found most suitable.

The EMRAN RBF NN was run on a 1.6GHz Pentium processor and had a 4-9-1 structure. In real-time implementation, it is important that the NN learning time (t_s) for one sample of data is always less than the flight data sampling time to avoid any delays. It was found that on average t_s was 0.2 ms which was considerably lower than the sampling time of 20ms.

8. CONCLUSIONS

An EMRAN RBF NN, with a 4-9-1 structure, is developed to detect and accommodate sensor faults appearing in the pitch gyro of a nonlinear UAV model. Two fault detection methods were implemented. One considered the conventional approach where the residual is generated and evaluated (RGE) against a threshold. The other considering damping (padding) the residual prior to residual evaluation (RGPE). The RGPE outperformed the RGE with lower false alarm rates and no missed faults. Further work must investigate the robustness of the RGPE approach to a wider class of faults especially intermittent faults.

Acknowledgements

The work presented here is part of project FLAVIIR which is funded by British Aerospace (BAE) Systems. Support for the 1st author was given by the Overseas Research Scheme (ORS).

REFERENCES

An, Y. (1998). A design of fault tolerant flight control systems for sensor and actuator failures using on-line learning neural networks. Ph.D. thesis, Dept. of Mech. and Aerospace Eng., Morgantown, West Virginia.

Campa, G., Fravolini, M. L., Napolitano, M. and B. Seanor (2002). Neural networks based sensor validation for the flight control system of a B777 research model. *American Contr. Conf.,* **Vol. 1**, pp. 412-417.

Capriglione, D., Liguori, C. and A. Pietrosanta (2007). Real time implementation of IFDIA scheme in automotive systems. *IEEE Trans. Instrum. Meas.*, **Vol. 56**, pp. 824-830.

Chow, E. Y. and A. S. Willsky (1984). Analytical redundancy and the design of robust detection systems. *IEEE Trans. Automat. Contr.*, **Vol. AC-29**, pp. 603-614.

Chen, J. and R.J. Patton (1999). Robust model-based fault diagnosis for dynamic systems. Kluwer Academic.

Emami-Naeini, A. E., Akhter, M. M., and S. M. Rock (1988). Effect of model uncertainty on failure detection: the threshold selector. *IEEE Trans. Automat. Contr.*, **Vol. AC-33**, pp. 1106-1115.

Franchi, P. L. (2005). Grand designs: An EC-funded research project has unveiled its proposals for a new generation of aircraft that are intended to give Europe the edge in the civil UAV sector. *Flight International,* pp. 109-114.

Heredia, G., Ollero, A., Mahtani, R., Bejar, M., Remuss, V. and M. Musial (2005). Detection of sensor faults in autonomous helicopters. *IEEE Int. Conf. on Robotics and Automation*, pp. 2229-2234.

Isermann, R. and P. Balle (1997). Trends in the application of model-based fault detection and diagnosis of technical processes. *Contr. Eng. Practice,* **Vol. 5,** pp. 709-719.

Li, Y., Sundararajan, N. and P. Saratchandran (2000). Analysis of minimal radial basis function network algorithm for real time identification of nonlinear dynamic systems. *IEEE Contr. Theory and Application*., **Vol. 4**, pp 476-484.

Mehra, R. K. and J. Peschon (1971). An innovations approach to fault detection and diagnosis in dynamic systems. *Automatica*, **Vol. 7**, pp. 637-640.

Patton, R. J., Frank, P.M. and R. N. Clark (1989). Fault diagnosis in dynamic systems, theory and application. Control Engineering Series. Prentice Hall, London.

Perla, R., Mukhopadhyay, S. and A. N. Samanta (2004). Sensor fault detection and isolation using artificial neural networks. *IEEE Region Conf. TENCON,* **Vol. 4**, pp. 676-679.

Simani, S., Fantuzzi, C. and R. J. Patton (2003). Model-based fault diagnosis in dynamic systems using identification techniques. Springer-Verlag, London.

Watanabe, K. and D. M. Himmelblau (1982). Instrument fault detection in systems with uncertainties. *Int. J. Sys. Sci.*,**Vol 13**, pp. 137-158.