USE OF ARTIFICIAL NEURAL NETWORKS TO FAULT DETECTION AND DIAGNOSIS

Diogo Leite Rebouças, diogolr@dca.ufrn.br
Fábio Meneghetti Ugulino de Araújo, meneghet@dca.ufrn.br
André Laurindo Maitelli, maitelli@dca.ufrn.br
Universidade Federal do Rio Grande do Norte, Technology Center, Departament of Computer Engineering and Automation, 59078-900 – Natal/RN – Brazil

Abstract. *In a real process, all used resources, whether physical or developed in software, are subject to interruptions or operational commitments. However, in situations in which operate critical systems, any kind of problem can bring big consequences. A coupled water tank system was used as a study case model for implementing and testing the proposed methodology. The developed system should generate a set of signals to notify the process operator about the faults that are ocurring, enabling changes in control strategy or control parameters. Due to the damage risks involved with sensors, actuators and amplifiers of the real plant, the data set of the faults are computationally generated and the results will be collected from numerical simulations of the process model. The system will be composed by structures with Artificial Neural Networks.*

Keywords: *Critical Systems, Fault Detection, Fault Diagnosis, Artificial Neural Network.*

1. INTRODUCTION

In the past, the automated supervising process, were mostly composed by some kind of system that had the simple task of checking whether a given variable, such as strength, speed, pressure, level or temperature, exceeds a certain limit or threshold previously specified. If it has ocurred, an alarm was triggered and the operator was warned about the incident, acting in a way to correct the problem. Sometimes the problem could also be corrected in an automatic way for some protection subsystem. This procedure, in many cases, was enough to prevent failures or severe damages to the process, but, the failures or errors were detected only after a certain period of time, which prevents the system from obtaining a detailed diagnosis about what happened (Isermann, 2006).

Considering the methods of Fault Detection and Diagnosis (FDD) that use Artificial Neural Networks (ANN), a series of contributions can be highlighted, such as Gao *et al.* (2000), where a system composed by an Elman neural network was trained to detect faults on engine units, Guo *et al.* (2005), that combines wavelet transform with ANNs to detect faults in rotation machines, Tian *et al.* (2007), where a neuro-fuzzy system was used to detect faults on pipelines and Khaled *et al.* (2010) where a principal component analysis was combined to ANNs to detect faults in manufacturing processes.

Based on this methods, this paper aims to develop a FDD system with ANNs for a dynamic process. The system should be capable to generate alert signals to the process operator in such way that they can be post processed by another system. Thereunto, the system will use a residual error generated by the difference between the real measured variable and the estimated value of this variable obtained from an identification structure of a study case model.

In the following sections the proposed system will be described with more details. The second section should summarize the main concepts of the used ANNs, showing its architecture and model. Following, the third section will show the basic concepts and terminology about FDD, and the proposed system will appear after the study case model description at the fourth section. The last two sections shows the obtained results and conclusions.

2. NEURAL NETWORKS

According to Haykin (2000), the Artificial Neural Networks (ANNs) are parallel structures, massively distributed, composed by simple processing units, named neurons. These structures resemble the human brain due to its ability to acquire knowledge from the environment. This learning occurs through an adjustment of the connection weights, or synaptic weights, which exists between neurons. These connections stores the information acquired by the network.

Among the various neural network architectures, such as radial basis function networks, Kohonen networks, support vector machine and so many others, this work uses a Multilayer Perceptron (MLP), due to its simplicity and applicability. The training algorithm used was the Levenberg-Marquardt (LMA), available in mathematical software Matlab®.

2.1 Process identification with neural networks

As described in Lucena (2005), the model suitable structures for nonlinear system identification are generalizations of linear models. These structures are characterized by their regression vector, which is a vector containing the variables used to estimate the system output. Depending on the choice of the regression vector, different neural model structures

may arise. FIR (Finite Impulse Response), ARX (AutoRegressive eXternal input), ARMAX (AutoRegressive Moving Average eXternal input), OE (Output Error) and SSIF (Innovations State Space Form) are some of the best known linear structures. If the regression vector is selected for ARX models, for example, the structure of the neural model will be called NNARX (Neural Network ARX). Similarly, there will also NNFIR models, NNARMAX, NNOE and NNSSIF.

In this work a NNARX model, based on Nørgaard *et al.* (2000) description, was used in the process identification and in the FDD networks. The regressors of the model, relates the network output with its past values of input and output. Because of that, the use of these regressors are fundamental for system identification. The mathematical expression that describes the nonlinear model used can be viewed in Eq. (1).

$$\hat{y}(t) = f\left(y(t-1), \ldots, y(t-n), u(t-d), \ldots, u(t-d-m)\right) \tag{1}$$

In this equation $\hat{y}$ represents the estimated output, d the transport delay, n the output order, m the input order, $f(\cdot)$ a nonlinear function maped by the neural network, y the output and u the input. In order to ease the implementation, in all trained networks, the input order was the same as the output order ($m = n$).

An estimative generated by NNARX structure is always stable, since it represents purely algebraic relations between the variables and there is no feedback of the estimated output.

3. FAULTS, ERRORS AND FAILURES

The computer systems can be characterized by five fundamental properties: functionality, usability, performance, cost and *dependability* (Kaâniche *et al.*, 2002). According to Laprie *et al.* (1992), the term dependability is related to the system ability to provides a service that can be, justifiably, reliable. Following this reasoning, Avižienis *et al.* (2000) subdivides a dependable system into the three parts, as shown in Fig. 1.

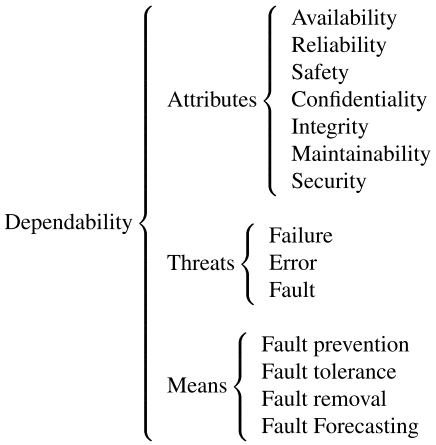


Figure 1. Dependability systematic classification, based on Avižienis *et al.* (2000).

The first group is used to provide an analysis about the quality of a dependable system. The second group brings the terms used to express undesirable threats – but, in principle, not unexpected – that makes the system become not dependable. Finally, the third group shows the means and the techniques by which it becomes possible to offer a dependable service.

About the second group, the term *failure* must be used to indicate when occur a deviation of the system behavior, making it incapable to provide the service for which it was designed. An *error*, however, is related to the system state and can lead to a failure. Briefly, if there is an error, then there is a sequence of actions that can lead to a failure. Last, but not least, the term *fault* is the cause of an error and is related to a defect. Normally, it is said that the term *fault* may be defined as a defect that has the potential to generate errors.

3.1 Fault detection and diagnosis

In order to ensure the success of planned operations and recognize the behaviorial problems in the process, many supervision and monitoring systems are being developed. According to Chiang *et al.* (2001), among other functions, these systems can detect, diagnose and eliminate faults, ensuring that the process operations satisfies the performance specifications.

Additionally, the information provided by a monitoring system should not only inform the system operator about what is going on, but also help him to take corrective actions in order to remedy the problem. As a result, the ineffective time will be reduced, the system protection will be increased and the operational costs will be decreased.

Chiang *et al.* (2001) shows that there are four states involved in the process monitoring: fault detection, fault isolation, fault diagnosis and fault recovery, as shown in Fig. 2. Although arranged as a sequence of actions, all states are not always strictly necessary. Often, automated changes from one state to another is transparent to the operator, displaying only the crucial information to take appropriate action.

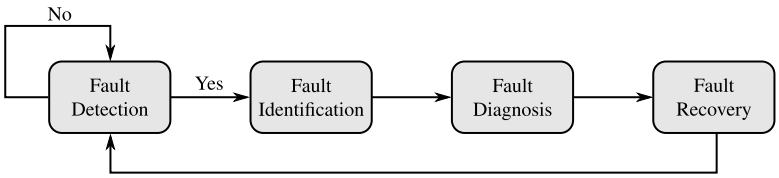
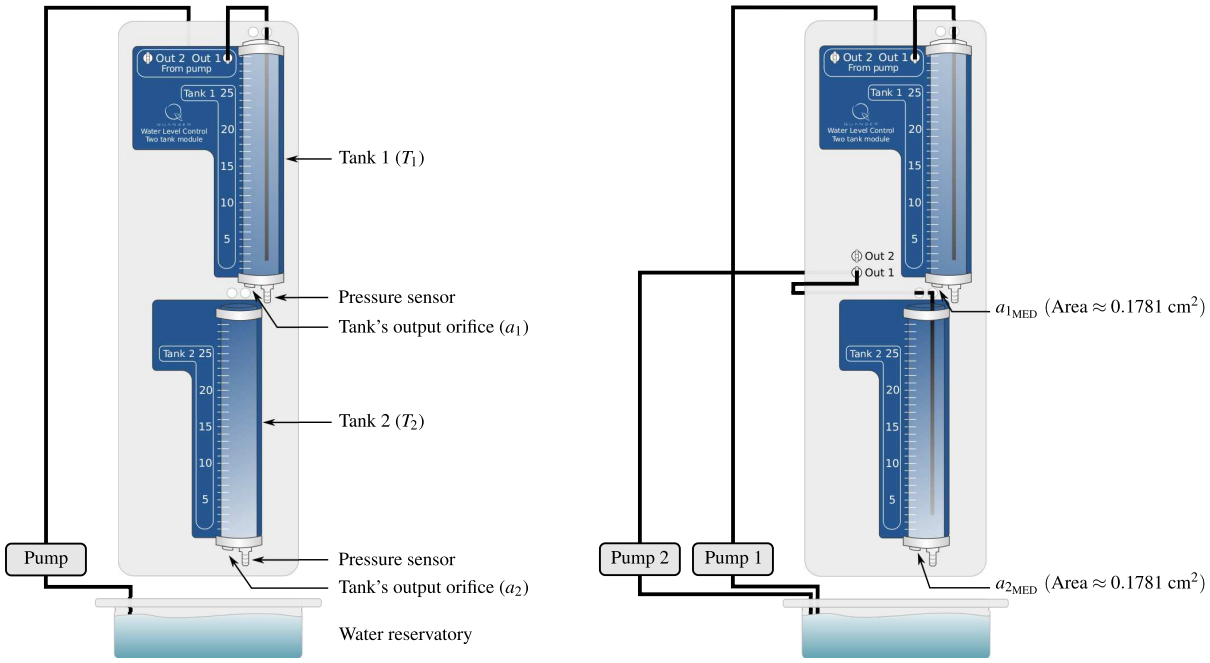


Figure 2. States of fault detection and diagnosis, based on Chiang *et al.* (2001).

4. PROPOSED SYSTEM

This paper proposes a system development for FDD in a dynamic process. The process in question consists of a coupled water tanks system developed by Quanser®, schematically represented in Fig. 3(a).



(a) Original configuration. (b) Proposed configuration.

Figure 3. Case study – Coupled Water Tanks.

The tanks (T_1 e T_2) are mounted on the front of the support base and positioned in such way that the water flows from the upper tank (T_1) to the bottom tank (T_2) through orifice a_1, and from the T_2 to the water reservatory through orifice a_2. The output water flow varies according to the orifices a_1 and a_2, available in three different diameters.

Since the two tanks have the same cross-sectional area ($A_1 = A_2 = A$), their dynamics are similar. However, find a mathematical model that adequately describes the dynamics of such tanks is not so simple, because the general equations of motion and energy that describe the fluid flow are quite complicated. Therefore, some fundamental assumptions are needed. So, it is assumed that the water in the tank is incompressible and the flow is non-viscous, non-rotational and regular (Dorf and Bishop, 2009). Considering these aspects, after a series of algebraic manipulations using Bernoulli's equation, the equations for a direct feed in T_1, can be described by Eqs. (2) and (3).

$$\dot{L_1} = \frac{K_m}{A} V_p - \left[\frac{a_1}{A}\sqrt{2g}\right]\sqrt{L_1} \tag{2}$$

$$\dot{L_2} = \left[\frac{a_1}{A}\sqrt{2g}\right]\sqrt{L_1} - \left[\frac{a_2}{A}\sqrt{2g}\right]\sqrt{L_2} \tag{3}$$

where K_m represents the pump flow constant, V_p the voltage applied to the pump, a_i the T_i's output orifice, L_i the water level in T_i and g the gravity acceleration.

In order to make the proposed system more generic and possibly make further studies on fault tolerance, the system was modified by introducing another pump with the same characteristics as the first one, as shown in Fig. 3(b). Clearly, in this case, the system has no more only a single input and a single output (SISO). Now, the equations of the multiple input and multiple output (MIMO) system can be described by Eqs. (4) and (5).

$$\dot{L_1} = \frac{K_m}{A} V_{p_1} - \left[\frac{a_1}{A}\sqrt{2g}\right]\sqrt{L_1} \tag{4}$$

$$\dot{L_2} = \frac{K_m}{A} V_{p_2} + \left[\frac{a_1}{A}\sqrt{2g}\right]\sqrt{L_1} - \left[\frac{a_2}{A}\sqrt{2g}\right]\sqrt{L_2} \tag{5}$$

where V_{p_1} is the voltage applied to the first pump and V_{p_2} is the voltage applied to the second pump.

4.1 Simulated faults

Despite the various faults that may exist in a coupled water tanks, only some of these were selected to be simulated, as shown in Tab. 1.

Table 1. Selected faults – Classification.

Fault #	Name	Acronym
Sensors		
1	Uncalibrated Gain	UGSeF
2	Uncalibrated Offset	UOSeF
3	Noise Sensitivity	NSSeF
4	Burned Sensor	BSeF
Actuators		
5	Uncalibrated Gain	UGAF
6	Uncalibrated Offset	UOAF
7	Noise Sensitivity	NSAF
8	K_m variation	K_mAF
9	Burned Actuator	BAF
Structural		
10	Tank's Leak	TLStF
11	Tank's a_i variation	Ta_iVStF
12	Tank's a_i obstruction	Ta_iOStF

Since in these types of simulation the system is normally exposed to adverse conditions, which could cause damage throughout the structure involved, the proposed system was computationally simulated.

4.2 Neural structures

The neural networks for identification and FDD should be carefully chosen, since an inappropriate choice can make the system innefective, not performing the function for which it was assigned.

The neural structure for identification, that must represent the dynamics of the system, has a single neural network, which receives as input the past values of the levels, $L_1(k-1)$ and $L_2(k-1)$, and voltages applied, $V_{p_1}(k-1)$ and $V_{p_2}(k-1)$, generating, on its output, estimated levels, called $\widehat{L}_1(k)$ and $\widehat{L}_2(k)$. The best trained neural network to this purpose was obtained from a second-order model NNARX with eight neurons on hidden layer. The validation mean square error was 3.73×10^{-6}.

The structure of FDD, in turn, was composed by twelve neural networks, in which each of these is associated with a single fault, configuring a set of "specialists". However, it is not a committee machine, since there is no network that performs the decision-making.

The input of each network is composed by the past values of the levels, $L_1(k-1)$ and $L_2(k-1)$, the voltages applied to the pumps, $V_{p_1}(k-1)$ and $V_{p_2}(k-1)$, and the residual errors produced from the difference between the real and estimated output, $e_i(k) = L_i(k) - \widehat{L}_i(k)$. The output of each network is a 2-bit binary word, which indicates whether the fault is being detected in T_1, in T_2 or in T_1 and T_2 simultaneously. A schematic diagram can be viewed in Fig. 4.

Opting for this disarticulated neural networks structure occurs by several factors. An example that can be highlighted is the fact that more than one fault may be happening simultaneously in the system. In this case, if only one neural network was used, beyond the N distinct words (one for each fault), the FDD system should indicate each fault combination in the

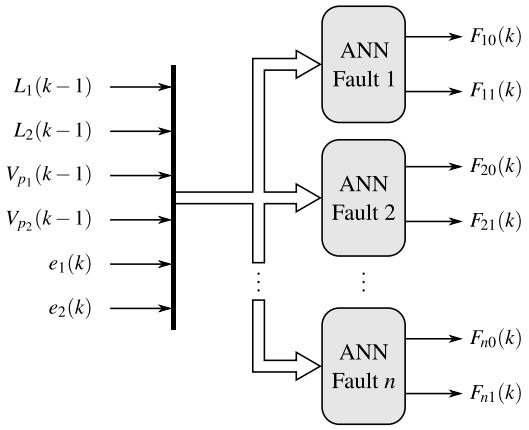


Figure 4. Neural network structure for fault detection and diagnosis.

output. Considering only the twelve selected faults, taken two by two, a total of 66 possible combinations should to be generated. If all combinations were considered this number would grow exorbitantly.

Moreover, a simple modification at one neural network will not affect in any other. Thus, if at any time is noticed that the introduction of a new variable improves the detection for one fault, only one neural network needs to be retrained. In future works, hibrid structures – like neuro-fuzzy networks, Kalman filters, statistical analysers and many more –, can be also utilized as a specialist.

Once known all used subsystems, the proposed FDD system can be viewed in Fig. 5. Attach this system to another, or associate it with a Fault-Tolerant Control System (FTCS), can be made in a simple way, by processing the information available at the output interface, named F_1, F_2, ..., F_n. However, this is not the objective here.

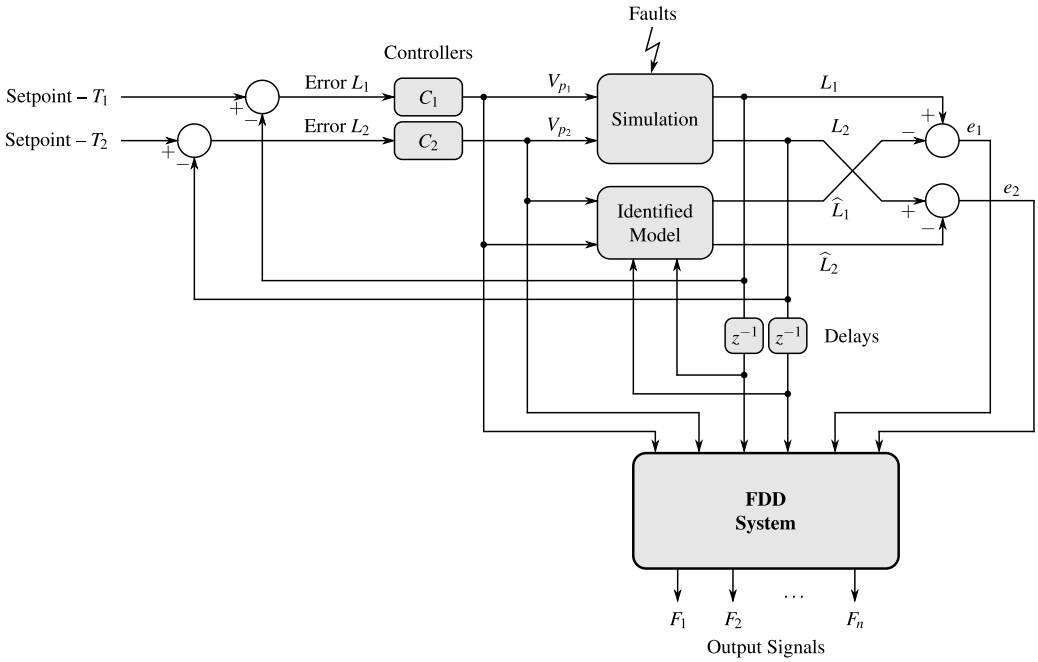


Figure 5. Proposed system schematic diagram.

5. RESULTS

5.1 Data acquisition

The first step to be taken for the identification and detection processes, is to obtain the experimental samples for a supervised neural networks training. So, the data acquisition was done by a stimulation of the system through the application of pseudo random binary signals (PRBSs) in the setpoint of each tank and in the system fault parameters.

The range values applied to the setpoint varies between the minimum (zero) to the maximum (thirty), while for the fault detection, the values were applied as shown in Tab. 2. The values generated in the interval determined by the

minimum and maximum of each parameter were multiplied by the default values and applied to the model.

Table 2. Applied values for training step.

Fault	Default value	Min	Max	Representativeness
UGSeF	0,16*	0,8	1,2	Up to ±6 cm
UOSeF	1,0	-3,0	3,0	Up to ±3 cm
NSSeF	1,0	-0,03	0,03	Up to ±9 cm
BSeF	1,0	0,0	0,0	–
UGAF	1,0	0,8	1,0	Up to -3 Volts
UOAF	1,0	-1,0	0,0	Up to -1 Volts
NSAF	1,0	-0,03	0,03	Up to ±0,45 Volts
K_mAF	K_m	0,7	1,1	–
BAF	1,0	0,0	0,0	–
TLStF	$a_{i_{\text{MED}}}$**	0,25	0,75	25 a 75% of $a_{i_{\text{MED}}}$
Ta_iVStF	$a_{i_{\text{MED}}}$	0,75	1,25	±25% of $a_{i_{\text{MED}}}$
Ta_iOStF	$a_{i_{\text{MED}}}$	0,0	0,5	–

* Established by the manufacturer.
** Cross-sectional area is approximately 0.1781 cm^2.

During the identification process were obtained 6,000 (six thousand) samples, wich is equivalent to 10 (ten) minutes of simulation. About the fault detection process, 12,000 (twelve thousand) samples were obtained, wich is equivalent to 20 (twenty) minutes of simulation. All data were collected with a sampling period of 100 ms, identical to that used in the real process.

In possession of the obtained values, the training of the neural networks was started. All networks were trained in offline mode with the neural networks toolbox of Matlab®, using the LMA algorithm.

5.2 Selected neural structures

As has been seen, the best network used for the model identification was obtained from a second-order NNARX structure with eight neurons on hidden layer. This network was selected among 54 others who had been trained for this same purpose and has a mean square error of 3.73×10^{-6}.

The number of trained neural networks increases significantly for the FDD. For each order of the NNARX structure, the number of the neurons on hidden layer was changed three times. Each time that number was changed, six neural networks were trained, which guarantees that the selected networks would not be compromised by convergence problems due the bad weights initialization or due to local minima. Thus, for a second order structure, for example, were trained $3 \times 6 = 18$ neural networks.

The networks were selected from a second, third and fourth-order structures. So, the number of trained networks would amount to $18 \times 3 = 54$ for each fault. However, there were twelve faults to be trained. Thus, the total was $54 \times 12 = 648$ distinct neural networks.

Because of this large number, all the networks went through a validation process with three simulations. In each simulation were recorded Type I and Type II errors, composing an average error for each structure. The average values obtained during the selection process can be viewed in Tab. 3.

The system was composed after a selection of the best structures and submited to the final simulation of one minute and forty five seconds, divided into intervals of fifteen seconds as shown in Fig. 6.

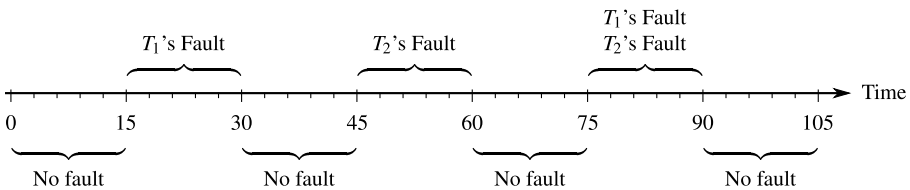


Figure 6. Final simulation – Intervals.

In this final test, the values of each fault parameter were kept fixed during the interval in which that fault was acting on the system. The values of these parameters are shown in Tab. 4.

Table 3. Best ANNs for FDD.

Fault	Order	HLN*	Train. #	Correct Answers	Type I Errors	Type II Errors	Total Errors (%)
UGSeF	4	28	2	23491,33	203,33	305,33	2,12%
UOSeF	4	28	5	23890,33	8,66	101	0,46%
NSSeF	4	20	3	23317	324,66	358,33	2,84%
BSeF	4	20	4	23994	0,66	5,33	0,02%
UGAF	2	8	3	20710,33	1626,66	1663	13,7%
UOAF	4	28	3	23075,33	635,66	289	3,85%
NSAF	2	8	6	14153,33	3407	6439,66	41,03%
K_mAF	2	8	5	20764,66	1551,33	1684	13,48%
BAF	4	28	6	23980	2,33	17,66	0,083%
TLStF	4	24	1	23774,33	74	151,66	0,94%
Ta_iVStF	2	8	3	22465,33	437	1097,66	6,39%
Ta_iOStF	2	12	4	23995,66	1	3,33	0,018%

* Hidden layer neurons.

Table 4. Used parameter values.

Sim. #	Fault	Modified value
1	UGSeF	Gain $= 0,128$
2	UOSeF	-2,0 cm
3	NSSeF	$\pm$2%
4	BSeF	Gain $= 0,0$
5	UGAF	Gain $= 0,8$
6	UOAF	-0,5 Volts
7	NSAF	$\pm$2%
8	K_mAF	$K_m = 3,45$
9	BAF	Gain $= 0,0$
10	TLStF	$a_{\text{VZ}} = a_{\text{MED}}/2$
11	Ta_iVStF	$a_i{}' = a_{\text{MED}}/2$
12	Ta_iOStF	$a_i{}' = a_{\text{MED}}/4$

5.3 Obtained results

The obtained results can be seen in Figs. 7 to 18. In these images, the red hatched areas represents the intervals in which the fault was detected in T_1, while the blue hatched areas represents the intervals in which the fault was detected in T_2.

The first fault to be simulated was UGSeF, whose the results can be seen in Fig. 7. In this simulation the value of the sensor gain was reduced to 80% of the default value. In this figure, the system identified the presence of the fault only when the parameter value was modified. After this period, the fault was "compensated" by the controllers, who sent more voltage to the pump, causing the return of the output to the setpoint. However, that "compensation" was made in a improperly way, since the sensor's reading had an error of 20%.

Thus, when the value read by the sensors is 24 cm, the tank is about to overflow, reaching, in fact, the upper limit of 30 cm. In an academic application this may not represent any risk to equipments beyond those that the water could cause. But, in critical applications, that "compensation" may bring several damages.

The system behaved in a similar manner to that in UOSeF and TLStF, as observed in Figs. 8 and 9. Especially for the TLStF simulation, another output orifice, named a_L, was considered. This orifice has the same characteristics of the tank output orifice a_i, but has a different diameter.

The results for these faults are not consistent with the Tab. 3. This situation may be ocurring because the networks has identified the rapid dynamic changes of the PRBS, failing to identify continuous abrupt changes.

A possible alternative to solve this problem would be to use binary flags, activated at the time that the first variation was detected and deactivated in the next detection. These flags indicate that the faults are acting during the time interval in which they were active.

Another simulation shows that the NSSeF was easily identified by the system, as shown in Fig. 10. However, due to the noise with uniform distribution ($\pm$2%), the system can not detect the fault at some points. At these points, the value

generated by the `rand` function keeps the signal next to the setpoint.

Unlike the NSSeF, the simulation performed to the NSAF was not so easily identified, as shown in Fig. 11. The results obtained for T_1 can even be considered reasonable, while the results for T_2 are clearly unacceptable, since none of the points in which the fault should have been identified were recognized by the network. Nevertheless, the results are consistent with the Tab. 3, where the total error exceeds 40%.

As well as NNSeF, all other remaining faults were also easily identified by the system, as shown in Figs. 12 to 18.

6. CONCLUSIONS

This work was developed in order to provide a FDD system for a coupled water tanks. Thereunto, the system uses a neural structure to process the available values and inform the user about the faults that are ocurring.

Since this structure is completely disjointed, another techniques can be mixed to compose a hibrid fault detection and diagnosis system. The used techniques can replace those networks whose the performance were below the expectations.

Among the twelve selected faults, eight were easily identified and three had a satisfactory performance, with a small detection problem that can be solved with binary flags. The other fault was not correctly identified for T_2, but can be considered reasonable for detection on T_1.

The results may improve when the real values are used, since they vary within the range of values in which the networks were trained. This situation can correct the problem that occurs for detecting UGSeF, UOSeF and TLStF, avoiding the use of binary flags.

Thus, the system had a satisfactory performance and could be capable to identify about 92% of the proposed faults, proving that MLP networks are efficient structures for identification and for the fault detection and diagnosis.

Once tested with various excitation signals, the system could be attached to a FTCS. In this case, the signals generated by the FDD system will serve as an “alarm”. The FTCS, in turn, may perform the controllers reconfiguration by modifying their parameters, or even their structures, making that the system keep functioning properly, until the fault is corrected.

7. ACKNOWLEDGEMENTS

The authors thank CAPES for financial support and also the colleagues in ITA and UFPA.

8. REFERENCES

Avižienis, A., Laprie, J. and Randell, B., 2000. “Fundamental concepts of dependability”. In *Proceedings of the 3rd Information Survivability Workshop*. pp. 7–12.

Chiang, L.H., Russel, E.L. and Braatz, R.D., 2001. *Fault detection and diagnosis in industrial systems*. Springer.

Dorf, R.C. and Bishop, R.H., 2009. *sistemas de controle modernos*. Rio de Janeiro: Editora LTC, 11th edition.

Gao, X.Z., Ovaska, S.J. and Dote, Y., 2000. “Motor fault detection using elman neural network with genetic algorithm-aided training”. In *IEEE International Conference on Systems, Man, and Cybernetics*. Vol. 4, pp. 2386–2392.

Guo, Q., bin Yu, H. and dong Xu, A., 2005. “Modified morlet wavelet neural networks for fault detection”. In *International Conference on Control and Automation*. Vol. 2, pp. 1209–1214.

Haykin, S., 2000. *Redes Neurais: princípios e prática*. Bookman.

Isermann, R., 2006. *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer-Verlag.

Kaâniche, M., Laprie, J. and Blanquart, J.P., 2002. “A framework for dependability engineering of critical computing systems”. *Safety Science*, Vol. 40, No. 9, pp. 731–752.

Khaled, O., Hedi, D., Lotfi, N., Messaoud, H. and S-abazi, Z., 2010. “Fault detection and localization with neural principal component analysis”. In *18th Mediterranean Conference on Control Automation (MED)*. pp. 880–885.

Laprie, J., Avizienis, A. and Kopetz, H., 1992. *Dependability: Basic Concepts and Terminology*. Springer-Verlag New York, Inc., New York.

Lucena, P.B., 2005. *Análise de um Controlador baseado no Jacobiano Estimado da Planta Através de uma Rede Neural*. Master’s thesis, Universidade Federal do Rio Grande do Norte.

Nørgaard, M., Ravn, O., Poulsen, N.K. and Hansen, L.K., 2000. *Neural Networks for Modelling and Control of Dynamic Systems*. Springer-Verlag.

Tian, J., Gao, M., Cao, L. and Li, K., 2007. “Fault detection of oil pump based on fuzzy neural network”. In *Third International Conference on Natural Computation*. Vol. 2, pp. 636–640.

9. Responsibility notice

The author(s) is (are) the only responsible for the printed material included in this paper.

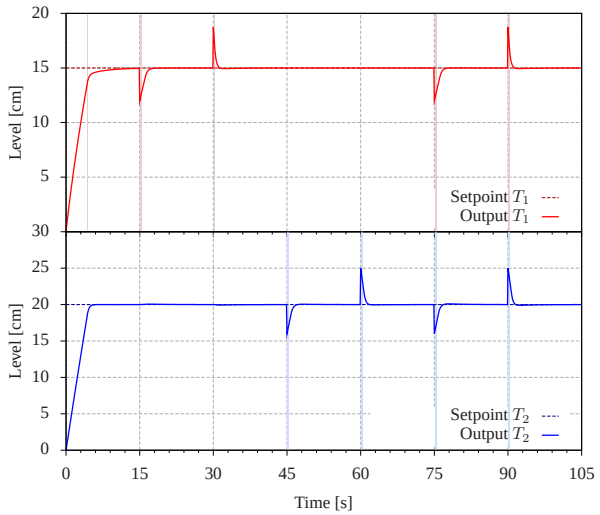


Figure 7. UGSeF simulation – Sensor's gain reduced to 80% from the default value.

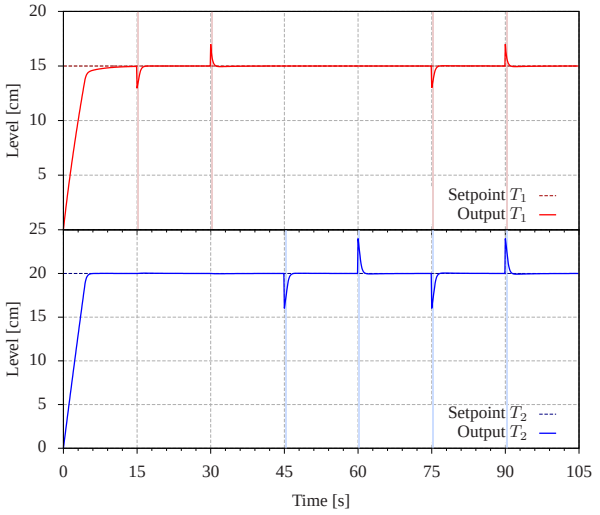


Figure 8. UOSeF simulation – Sensor's offset configured to -2 cm.

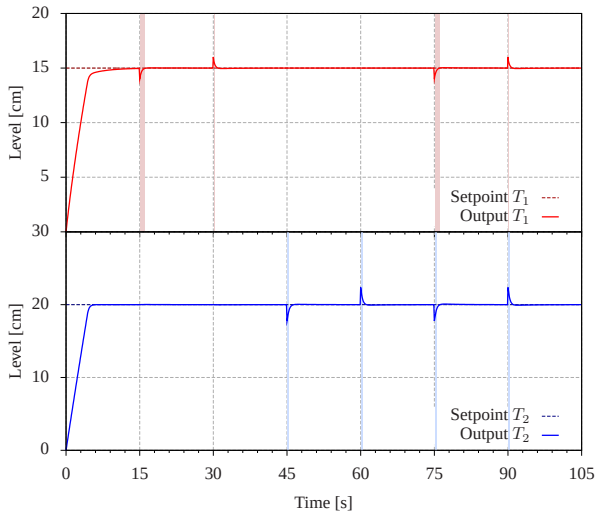


Figure 9. TLStF simulation – Where $a_L = a_{\text{MED}}/2$ and $a_{\text{MED}} \approx 0.1781$ cm^2.

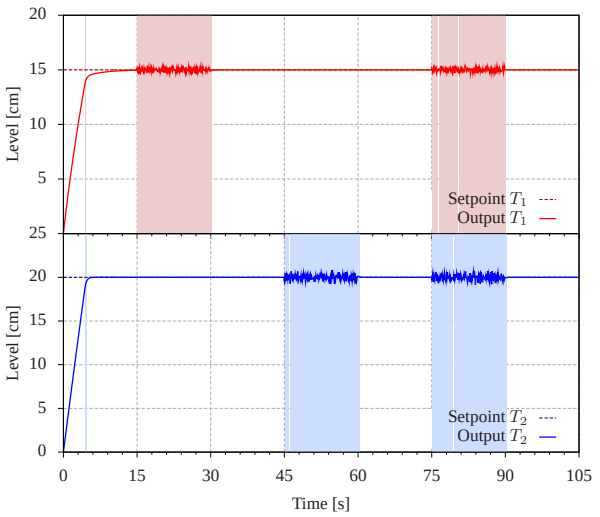


Figure 10. NSSeF simulation – Assumming a uniform distribution noise from $\pm 2\%$.

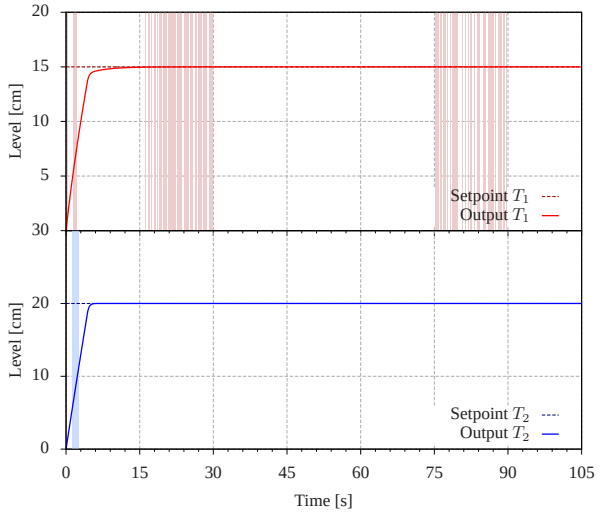


Figure 11. NSAF simulation – Assumming a uniform distribution noise from $\pm 2\%$.

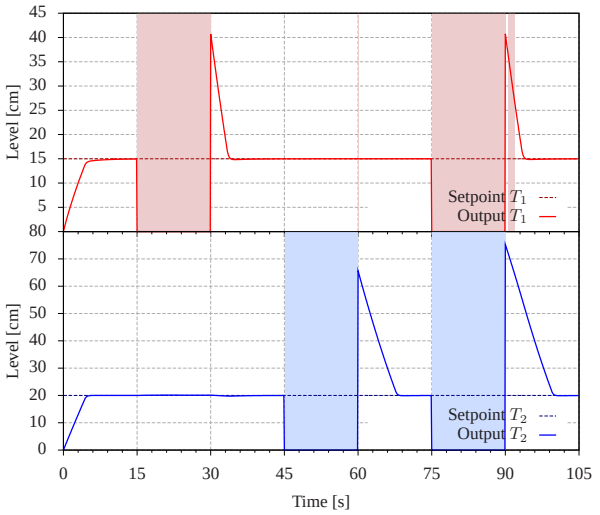


Figure 12. BSeF simulation – Sensor's gain reduced to zero.

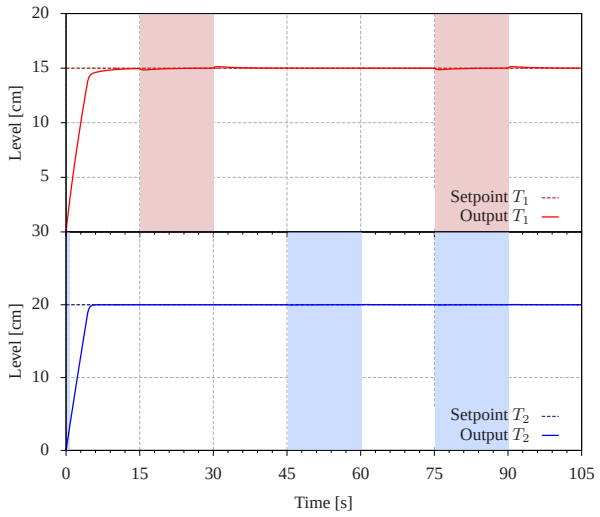


Figure 13. UGAF simulation – Actuator's gain reduced to 80% from the default value.

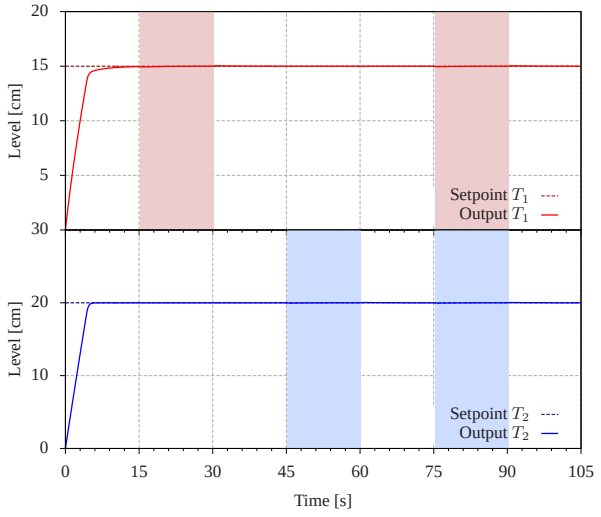


Figure 14. UOAF simulation – Actuator's offset configured to -0.5 Volts.

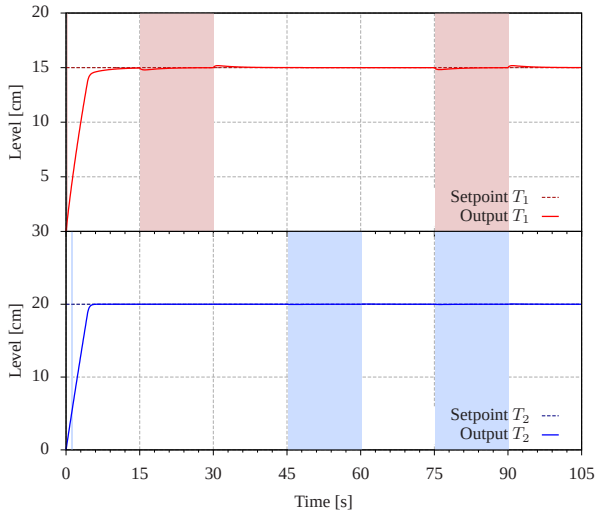


Figure 15. K_mAF simulation – K_m reduced to 75% from the default value.

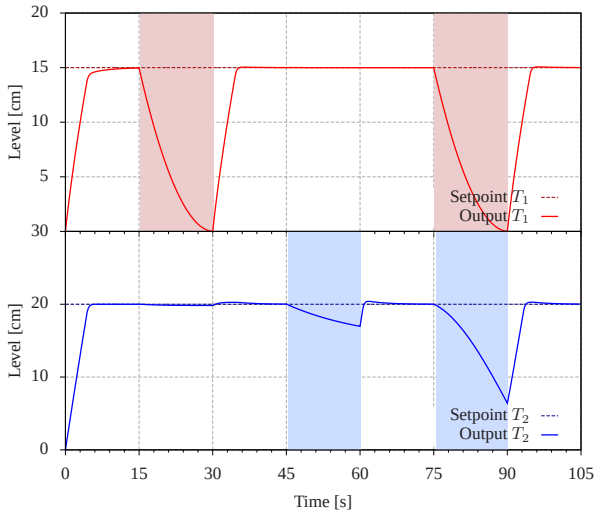


Figure 16. BAF simulation – Actuator's gain reduced to zero.

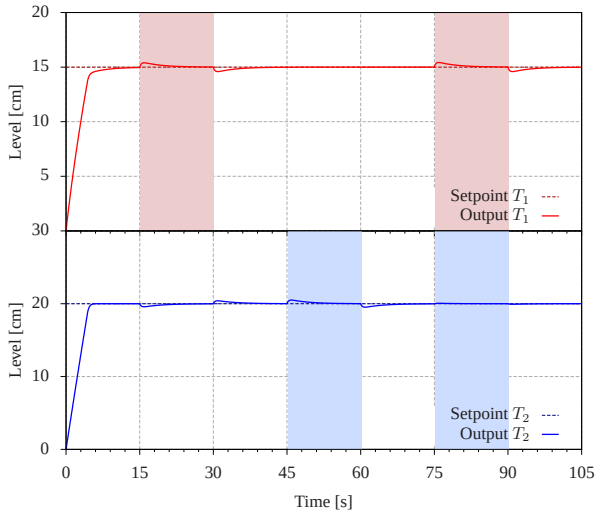


Figure 17. Ta_iOStF simulation – Where $a_i = a_{\text{MED}}/4$.

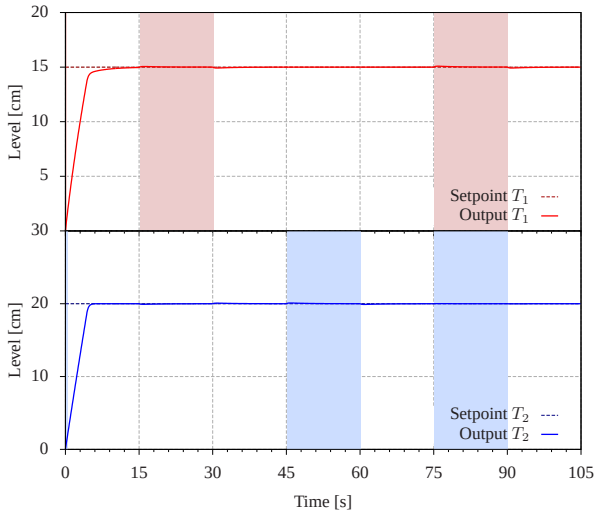


Figure 18. Ta_iVStF simulation – Where $a_i = a_{\text{MED}}/2$.