Int j simul model 7 (2008) 1, 5-16
ISSN 1726-4529 Original scientific paper

FAULT DETECTION AND ISOLATION IN ROBOTIC MANIPULATOR VIA HYBRID NEURAL NETWORKS

Dev Anand, M.; Selvaraj, T. & Kumanan, S.
Department of Production Engineering,
National Institute of Technology, Tiruchirappalli-620 015, Tamilnadu, India
E-Mail: anandpmt@yahoo.co.in

Abstract
Fault diagnosis systems are important for industrial robots, especially those operated in remote and hazardous environment. Faults in robotic manipulator can cause economic and serious damages. So the Robots need the ability to independently as well as effectively detect and tolerate internal failures in order to continue performing their tasks without the need for immediate human intervention. This saves time and cost involved in repairing the robot. This type of autonomous fault tolerance is also useful for industrial robots in that it decreases down-time by tolerating failures, identifies faulty components or subsystems to speed up the repair process, and prevents the robot from damaging the products being manufactured. So an attempt is made to develop a robust fault detection system to identify and isolate the faults in robot manipulator. In this paper, two artificial neural networks are employed to identify and isolate the faults. A learning architecture, approximation of dynamic behavior of robot manipulator, is used to generate the residual vector, by comparing with actual measured values. First, A multi layer perceptron feed forward network, whose structure is characterized by layered graph, trained with back propagation algorithm is applied to reproduce the dynamic behavior, then counter propagation network which learns a near optimal look up-table approximation to the mapping being approximated. The counter propagation network has the ability to compress a huge amount of data in a few weights and parameters. Simulations employing a SCORBOT ER 5u plus five links robotic manipulator are showed demonstrating that the system can detect and isolate correctly faults that occur in non-trained trajectories. The main contribution of this work is the first application of fault detection and isolation to robot manipulator with non-additive fault.
(Received in January 2007, accepted in October 2007. This paper was with the authors 3 months for 2 revisions.)

Key Words: Robot Manipulators, Fault Isolation, Fault Detection, Hybrid Neural Networks

1. INTRODUCTION

Robots are often used in inaccessible or hazardous environments in order to alleviate time, cost and risk involved in preparing humans to endure these conditions. In order to perform their expected tasks, the robots are often quite complex, thus increasing their potential for failures. However, if people are frequently sent into these environments to repair every component failure in the robot, the advantages of using the robot are quickly lost. Fault tolerant robots are needed which can effectively detect and adapt to software or hardware failures in order to allow the robots to continue working until repairs can be realistically scheduled. Hence availability and reliability of these components are important for production environment. Hence, automated monitoring of the robotic manipulator for any faults and effective accommodation of such faults plays a crucial role in the use of robotic manipulators as autonomous systems [1]. Fault diagnosis and isolation methods are usually based on the residual generation and analysis concept. A mathematical model is used to reproduce the

dynamic behavior of the fault-free system; the deviation of the output predicted by the model from actual output measurements forms the so-called residuals, which, when properly analyzed, provides valuable information about failures. The *ANN*-based *FDI* methodology is validated in an extensive simulation study with 5-axes SCORBOT 5u plus. This article is organized as follows: in Section 2 we briefly introduce the concepts of *FDI* we will be referring to throughout the article; the material in this section is widely available in the literature. In Section 3 we introduce the *ANN*-based *FDI* methodology for robotic manipulators. Sections 4 simulation results to validate the proposed framework. We conclude the article in Section 5.

2. LITERATURE REVIEW

Many fault tolerant systems have been developed for computer, airplane, and industrial systems. Several of these techniques used in fault tolerant systems have provided models for robotic fault tolerance schemes. However, the trend in robotics is to use only those schemes which rely on physical redundancy of components. Many methods of fault tolerance exist which do not alter the robot physical system.

2.1 Redundancy based robot fault diagnosis

Previous work on fault tolerance for the mechanical aspect of robots has concentrated on algorithms, which rely on duplicated parts for their fault tolerant abilities. These schemes generally deal with faults in one specific part of the robot (mechanical failure in the motor, kinematic joint failure, etc.). In duplicating the motor, the two motors in a joint must be able to work together to provide one output velocity for the joint. When one of the motors breaks, the other one takes over the faulty motor's functions while adjusting to any transients introduced into the system by the failed motor. The fault tolerant advantages of redundancy have also led to adding extra parallel structures, resulting additional investment cost.

2.2 Kinematically redundant fault diagnosis system

Many robots today have the advantage of being kinematically redundant [2]. That is, the robot has more degrees-of-freedom or motions than necessary to position and orient the end effector which allows the robot to choose between multiple joint configurations for a given end effector position in the robot workspace. This natural redundancy can be used to create fault tolerant algorithms, which use the alternate configurations to aid in positioning a robot with failed joints. These algorithms would not require the addition of extra motors, sensors, or other components to the robot but would use the existing structure to provide fault tolerance.

2.3 Analytical redundancy based fault diagnosis

The design and analysis of fault diagnosis architectures for robotic systems using the model-based analytical redundancy approach [3-6], has received considerable attention. In this approach quantitative nominal models of the robotic system together with sensory measurements are used. These approaches are usually based on state estimation, parameter estimation [7], and parity relations [8], yet most of the current techniques developed rely on the assumption that the process to be linear in nature [9-11]. The appeal of model-based approach [12] lies in the fact that the redundancy required for detecting faults is created using powerful information processing techniques without the need of additional physical instrumentation in the system. However, the model-based fault diagonsis approach relies on the key assumption that a mathematical characterization of the manipulator is known a priori.

In practice, this assumption is usually not valid since it is difficult to obtain the necessary modeling accuracy required for the construction of reliable analytical redundancy-based *FD* architectures. Unavoidable modeling uncertainties, which arise due to modeling errors, time variations, measurement noise, and external disturbances, deteriorate the performance of *FD* schemes by causing false alarms. This is evidenced in [13]. One way to deal with the absence of a mathematical model is to build a model from input-output data.

A number of studies have been dedicated to the assessment and analysis [14] of robot reliability. Other studies related to enhancing a robot's tolerance to failure include work on layered failure tolerance control [15], failure tolerance by trajectory planning [16], kinematic failure recovery [17] and manipulators specifically designed for fault tolerance [18]. The generalization to more joints being at their limits is obvious. Similar results hold for robots with higher degrees of redundancy, e.g., if two joints are at their upper limits, there must be a vector of the null space of *J* for which the corresponding components are nonzero and of the same sign. Of course, this is easier to determine when there is only a single degree of redundancy. More details on the multiple degree-of-redundancy cases can be found in [19].

Given a reasonably well-understood operational environment, there are two reasons for undesirable behaviours: random errors or systematic (design) errors. Random errors are those due to hardware or component faults, and these are typically analyzed using techniques such as Failure Mode and Effect Analysis (*FMEA*), [20]. The likelihood that random errors cause undesirable behaviours can be reduced, in the first instance, by employing high reliability components. But systems that require high dependability will typically also need to be fault tolerant through redundancy for example. This is an important point since swarm engineered systems should, in this respect, offer very significant advantages over conventional complex systems. This works study of reliability models is perhaps less conclusive. The most interesting conclusion is that a multi-state reliability model is needed to account for the partially failed robots identified by [21] *FMEA*. They have shown that a multi-state reliability model can have interesting implications for optimum swarm size (from a reliability perspective), although this finding comes with a clear health warning. Analysis of systematic errors for the swarm as a whole is much more problematical, particularly if the desired behaviours are emergent. However, in [22] they explore the use of the temporal logic formalism for specification and possibly proof of correctness of emergent behaviours. In addition to that [23], a model-based fault-detection approach was successfully demonstrated experimentally. This approach was based on the generation of residuals through a filtered torque estimate, which does not rely upon the measurement of acceleration quantities.

Overall, one problem with fault detection and isolation methods which rely on the system's mathematical model is that, for some real robots, detailed modelling is difficult. The failure prone complex operating environment of a standard multi-robot application dictates some amount of fault-tolerance to be incorporated into the system. Being able to identify the extent of fault-tolerance in a system would be a useful analysis tool for the designer [24]. Unfortunately, it is difficult to quantify system fault-tolerance on its own. The observations will help us further evaluate refine our approach. Propose an engineering based approach for measuring system intelligence. In this method, learning is used to theoretically measure system intelligence through a formal analysis of system software architecture and hardware configurations.

Other related works include [25] approach for measuring autonomy for intelligent systems and Analytical Hierarchy Process (*AHP*) [26] for measuring system intelligence. Recent techniques involve the use of neural networks or fuzzy systems for this purpose. This necessitates the development of *FD* algorithm, which has the ability to detect manipulator failures in the presence of modeling uncertainties. Such algorithms are referred as robust fault diagnosis schemes. Generally, the *FDI* process is viewed as consisting of two stages: residual

generation and decision making as shown in Fig. 1. Outputs from the sensory are processed and compared with the expected values from quantitative nominal model, resulting value is referred as residual. In the second stage, the decision process, the residuals are examined for the presence of failure signatures. Decision functions or statistics are calculated using the residuals, and a decision rule is then applied to the decision statistics to determine if any failure has occurred. It is argued that a robust *FDI* system can be achieved by designing a robust residual generation process.

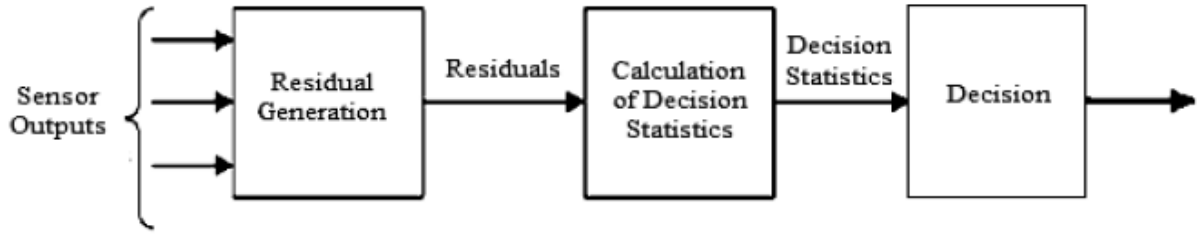


Figure 1: Two Stage Structures of Decision Statistics.

1) Fault *Detection and Isolation* (*FDI*) deals with determining if a malfunction has occurred in the system,
2) *Fault Diagnosis* (*FD*) considers the problem of isolating and/or identifying a fault.

In order to develop residual generation procedures, it is important to identify the redundancy relations of a system and to characterize them according to how they are affected by model errors and uncertainties. In this work, a systematic approach was developed to that consider uncertainties directly. Further to develop the concept of neural network based redundancy model as a basis for determining redundancy relations for residual generation. This residual generation is least sensitive to model errors. Numerous advantages characterize this method:

i. For diagnosis no additional sensors are required. Only the signals of motor armature current, motor angular velocity and axis position are necessary.
ii. The overall procedure works in real time during normal operation
iii. They lead to early and reliable fault detection

This paper presents a learning methodology, based on nonlinear modeling techniques. The robotic manipulator dynamics prior to the occurrence of a fault are assumed to be known exactly. The main idea behind this approach is to use neural networks to monitor the robotic system for any changes in the dynamics of the system due to faults. By using the approximation capabilities of neural network, the network can be used to generate the residual based on the fault.

3. ROBOT MANIPULATOR FDI WITH ANN

In this section we present the FDI method based on artificial neural networks. The dynamic nonlinear equations of an *n* degree-of-freedom robot manipulator in the continuous time in closed form can be written as equation (1):

$$M(q(\ddot{q})) + C(q,\dot{q}) + F(\dot{q}) + G(q) = \tau + \tau_d \qquad (1)$$

where q and τ are the *(n x 1)* vectors of joint variables and driving joint torques respectively, M is the *(n x n)* symmetric positive-definite inertia matrix, C is the vector of Coriolis and centrifugal forces, G is the vector of gravitational forces, F is the vector of friction torques and τ_d is a quantity including un-modeled disturbances. We begin with a presentation of residual generation for a generic dynamic system, and then specialize it for the case of a

robotic manipulator. In the sequence we discuss the fault isolation criterion we use and the failures that the system is able to cope.

3.1 Artificial neural networks

The most frequently applied neural models are the feed forward perceptron used in multilayer networks, i.e., the Multi Layer Perceptron (*MLP*), and the Counter Propagation Network (CPN). Both networks are capable of approximating any nonlinear unique static function to arbitrary desired accuracy. This form of mapping is well suited for pattern recognition applications, where both the input vector and the output one represent spatial patterns that are independent of time. The introduction of explicit dynamics into these *ANN*s requires the spatial representation of time. The MLP trained with the back propagation algorithm is a very popular model in neural network and can be used as a practical system for performing a nonlinear input/output mapping of a general nature. For a *p*-dimensional input vector and a *q*-dimensional output vector, the MLP input/output relationship defines a mapping from a *p*-dimensional Euclidean space to a *q*-dimensional Euclidean output space. Using only one hidden layer, presenting in the n^{th} sample (where $n = 1, 2...$ p), the input vector $I(n) = [X_1(n) X_2(n) ... X_p(n)]^T$, the activation of the output neuron k (where $k = 1, 2... q$) is:

$$O_k(n) = \varphi_k\left[\sum_{j=-0}^{n} W_{jk}(n)\varphi_j\left[\sum_{i=0}^{p} W_{j_i}(n)X_i(v)\right]\right] \tag{2}$$

where m is the number of neurons in the hidden layer, w_{kj} is the weight between the j^{th} neuron of the hidden layer and the k^{th} neuron of the output layer, w_{ji} is the weight between the i^{th} neuron of the input layer and the j^{th} neuron of the hidden layer, w_k is the nonlinear activation function of the output layer and w_j is the nonlinear activation function of the hidden layer. Typically, the network consists of a set of input parameters that constitute the input layer, one or more hidden layers of computation nodes and an output layer of computation nodes. The input signal propagates through the network in a forward direction on a layer-by-layer basis. MLPs have been applied to solve some difficult and diverse problems by training them in a supervised manner with a highly popular algorithm known as the error back propagation algorithm.

3.2 Residual generation

In discrete time, the state equation of a fault-free nonlinear dynamic system is given by:

$$x(t + \Delta t) = f(x(t), u(t)) \tag{3}$$

where $x(t)$ is the state vector at time t, $u(t)$ is the applied control vector, Δt is the sample rate and $f(.)$ is the vector-valued nonlinear function of the fault-free system. Considering now that a fault i occurs, the dynamics of the system are modified to:

$$x(t + \Delta t) = g_i(x(t), u(t)) \tag{4}$$

where $g_i(.)$ is the vector-valued nonlinear function of the system affected by fault i. The faults may or may not be additive inputs (i.e., dependent only on the time variable). The i^{th} fault vector can be defined as the difference between the faulty system dynamics (Eq. 3) and the fault-free system dynamics (Eq. 2):

$$\psi_i = g_i(x(t), u(t)) - f(x(t), u(t)) \tag{5}$$

Obviously, for the fault-free system $\Psi_i(t + \Delta t)=0$, generally, for each possible fault i, the fault vector Ψ_i has a particular behavior, called the fault signature. For the identification of the fault type, the fault vector must be computed and analyzed. Therefore, the dynamic behavior of the fault-free system (Eq. 2) must be estimated (for example, by the mathematical model or by an ANN). Then, when fault i occur, the residual vector can be computed as:

$$\dot{\psi}(t+\Delta t) = x(t+\Delta t) - \hat{x}(t+\Delta t) = g_i(x(t),u(t)) - \hat{f}(x(t),u(t)) = \Psi_i(t) + e_i(x(t),u(t)) \qquad (6)$$

where $\hat{f}(.)$ is a vector that represents the input-output mapping of the estimated fault-free dynamic behavior of the system and e_i is the error between the actual fault-free behavior and the estimated one. In real systems and fault free case, the error is due to external disturbances, un-modeled system uncertainty, mapping errors (or modeling errors in model-based systems), and measurement noise.

3.3 Residual generation in mechanical manipulators

The dynamic of a fault-free robotic manipulator with actuators in each joint is given by:

$$\dot{x} = \begin{bmatrix} \dot{q}(t) \\ q(t) \end{bmatrix} = \begin{bmatrix} \dot{q}(t) \\ M(q(t))^{-1}\left[\tau(t) + \tau_d(t) - C(q(t),\dot{q}(t)) - F(q(t),\dot{q}(t),t) - G(q(t))\right] \end{bmatrix} \qquad (7)$$

where q is the vector of joint angular positions, τ is the vector of joint torques, M is the inertia matrix, t is the time index, C is the vector of Coriolis and centrifugal forces, G is the vector of gravitational torques, F is the vector of frictional torques and other nonlinearities, and τ_d is the vector of external uncorrelated disturbances. As the robot joint accelerations are not usually measured in robotic manipulators.

3.4 Residual analysis

In this work residual analysis for fault isolation purposes is also performed with *ANN*s. Two network architectures are employed. A general schema for architectures is shown in Fig. 2. Outputs 1 through *q-1* correspond to the *q-1* possible failure modes, while output *q* corresponds to fault-free operation. The *ANN*'s output *i (i = 1... q-1)* is trained to present a '1' in case fault *i* occurs and '0' otherwise. The output *q* is trained to present a '1' in case fault-free operation and '0' otherwise.

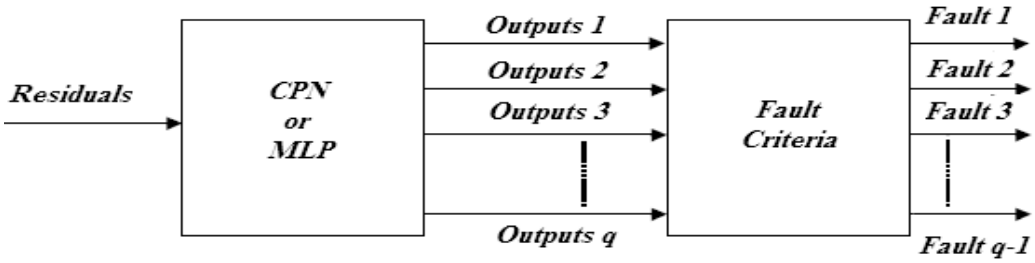


Figure 2: Residual Analysis Employing Architecture.

The *FDI* procedure can be summarized as follows:

1. Joint positions and velocities are measured at time step *t*;
2. The normalized positions and velocities and the torques applied at *t* are presented to the *MLP*, which estimates the positions and velocities at $t+\Delta t$;
3. The *MLP* outputs are compared with the normalized positions and velocities measured at $t+\Delta t$ to generate the residuals;
4. The velocity residuals are presented to the second *ANN* (*CPN* architecture);
5. The *CPN* (or *MLP*) classifies the residuals and generates a vector that, when analyzed under the fault criterion, indicates the operation status of the system (fault-free or faulty, along with the indication of the fault);
6. The time step *t* is incremented; return to step 1.

3.5 Fault indication criterion

Generally, the fault is signaled when a threshold is crossed. However, as the faults and the mapping errors are generally correlated with the system dynamics, a small threshold can be a source of false alarms, while a large one can hide the fault effects. The fault isolation scheme presented above will indicate the fault information in the *ANN*s outputs. It is known, however, that false alarms may occur, and one cannot rely on isolated information to decide on whether a failure actually occurred. In this work we adopt the following fault criterion: a fault is said to have occurred whenever one of the *ANN* outputs is greater than the other ones for g consecutive time steps; where g is achieved by trial and error (searching for a good compromise between false alarm rate and detection delays).

3.6 Failure modes

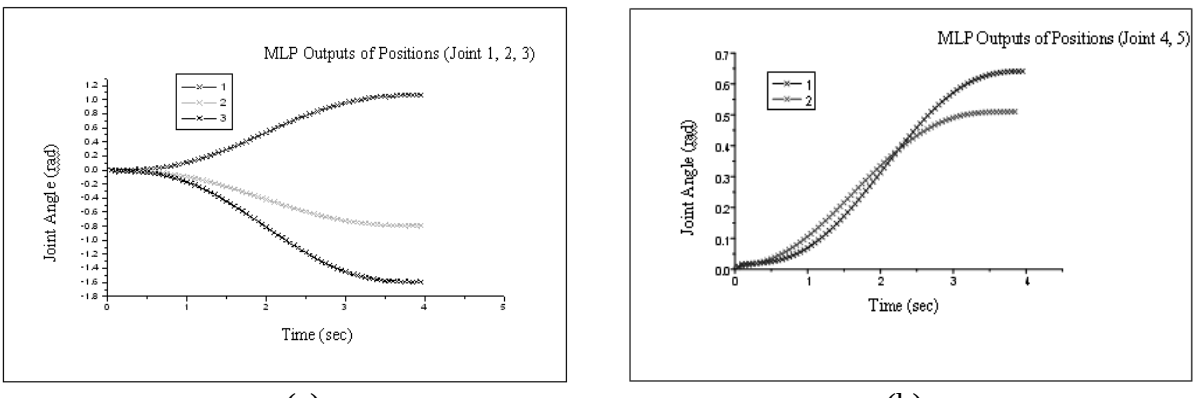
In this work we consider one locked joint fault for each manipulator joint. In free-swinging joint faults, a loss of torque occurs on a joint. This fault can be caused, for example, by a rupture seal on a hydraulic actuator, by a loss of electric power on an electric actuator, or by a mechanical fault in a drive system.

4. SIMULATION RESULTS

To evaluate our ANN-based FDI method we performed an extensive simulation study with a Scorbot *ER* 5u plus manipulator. The Scorbot ER 5u plus was simulated in *MATLAB* using the Kohonen Self-Organizing Map (*KSOM*) Algorithm [27].

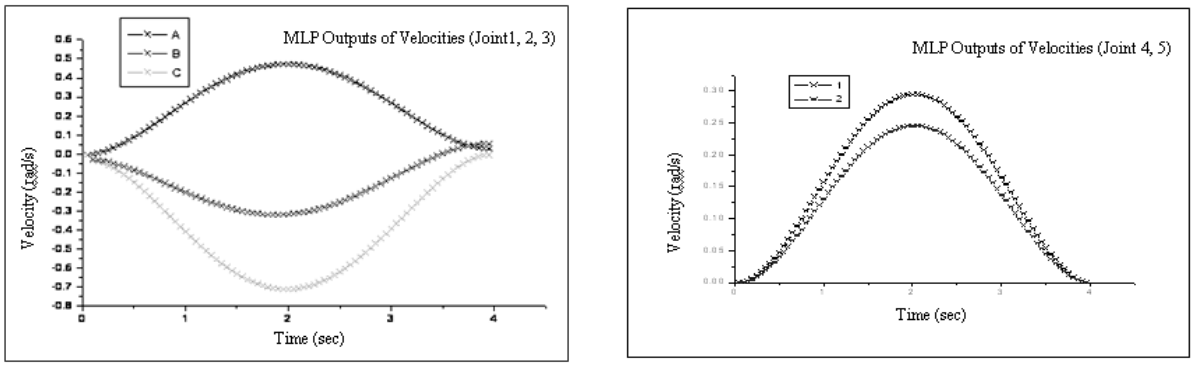
4.1 Residual generation results

The MLP used to reproduce the manipulator dynamic behavior has one hidden layer with 49 neurons. The input layer has 15 neurons (5 joint positions, 5 joint velocities, and 5 joint torques measured at t) and the output layer has 10 neurons (5 joint positions and 5 joint velocities at $(t+\Delta t)$). The training set is formed from 4000 patterns obtained by simulating more than 150 different trajectories with 54 samples each (at a sample rate of 0.071 s). The trajectories of the training set were selected randomly. Figs. 3(a), (b) and 4(a), (b) present the normalized joint positions and velocities and the respective *MLP* outputs in the simulation of an untrained fault-free trajectory. Figs. 5(a) and (b) presents the residuals for the same trajectory. Two test sets are used to validate the residual generation. The first set has 7000 patterns obtained in the simulation of 300 random trajectories. The second set has 7000 patterns with measurement noise (with variance = 0.01) added to the positions and velocities.



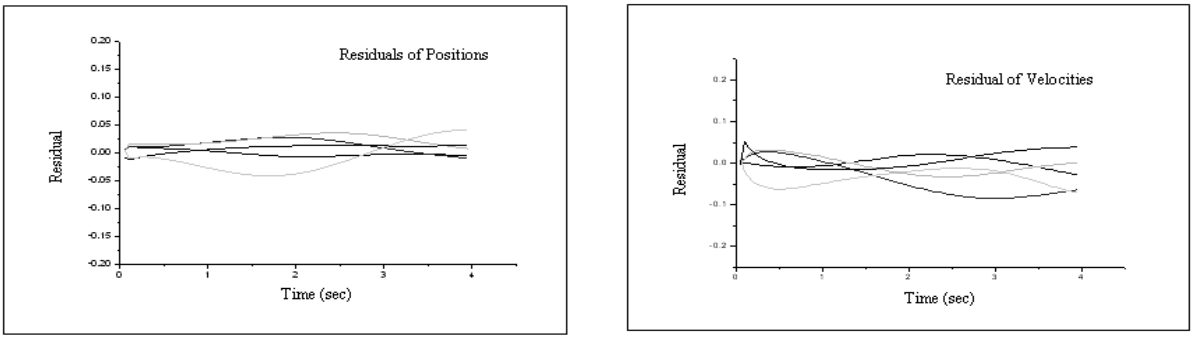
(a) (b)

Figure 3: Normalized Positions and Respective *MLP* Outputs for a Fault-Free Trajectory.



(a) (b)

Figure 4: Normalized Velocities and Respective MLP Output for a Fault Free Trajectory.



(a) (b)

Figure 5: Normalized Residuals of Positions and Velocities for Fault Free Conditions.

4.2 Residual analysis results: non-additive fault

The CPN is trained with 1100 patterns, obtained with the simulation of 120 trajectories with 20 samples each as shown in Table I.

Table I: Trajectories of the CPN Training Set (Non-Additive Faults).

Trajectories	Operation
1-20	Fault in joint 1 (fault 1)
20-40	Fault in joint 2 (fault 2)
40-60	Fault in joint 3 (fault 3)
60-80	Fault in joint 4 (fault 4)
80-100	Fault in joint 5 (fault 5)
100-120	No Fault

The architecture described in Section 3.4 was employed for residual analysis. The training parameters can be shown in the Table II.

Table II: Parameters Used for Residual Analysis (Non-Additive Faults).

Number of inputs	15
Number of neuron in the hidden layer	15
Number of outputs	10
Learning rate	0.1
Momentum	0.7
Number of patterns in the training	1100
Training time (MLP)	576 min
Training time (CPN)	95 min

The faults are set to occur in the beginning of the trajectories, before the manipulator reaches its set-point. Two test sets were used to evaluate the architecture. The first set has 7000 patterns obtained by simulating of 350 random trajectories with 20 samples each. The second set has 7000 patterns with measurement noise (with variance = 0.01) added to the positions and velocities. The Table III presents the mean squared errors for the architecture.

Table III: Mean Squared Errors (MSE) of the ANN's Trained in Classifying the Residual.

Set	CPN Architecture	MLP Architecture
Training (1100 patterns)	2.7×10^{-2}	1.5×10^{-2}
Test 1 (7000 patterns – without noise)	3.9×10^{-2}	3.2×10^{-2}
Test 2 (7000 patterns – with noise)	4.1×10^{-2}	3.8×10^{-2}

Applying the fault criterion described in Section 3.5 with $g = 5$, the results of the classification of the non-additive faults can be seen in Table IV for the test set 1 (7000 patterns without noise) and in Table V for the test set 2 (7000 patterns with noise).

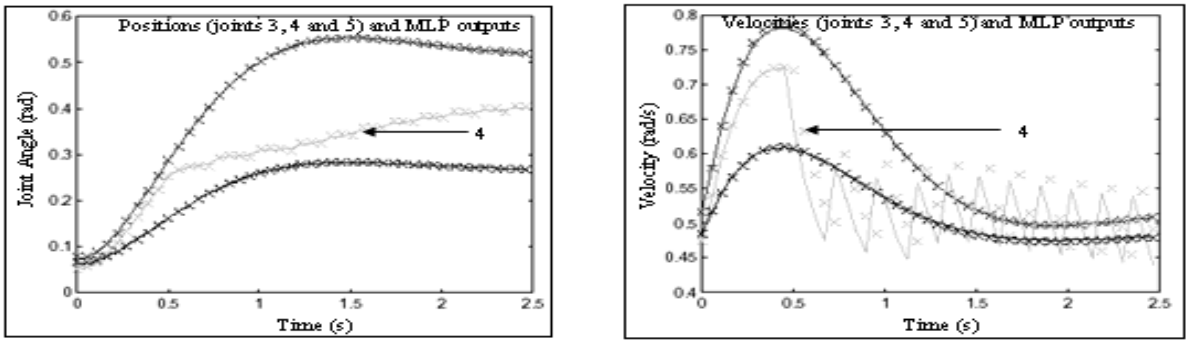
Table IV: Results of the FDI System for Test Set 1.

Set	CPN Architecture	MLP Architecture
Number of false alarms	3	4
Number of faults not detected	0	1
Trajectories without isolation errors	97.54 %	95.00 %

Table V: Results of the FDI System for Test Set 2.

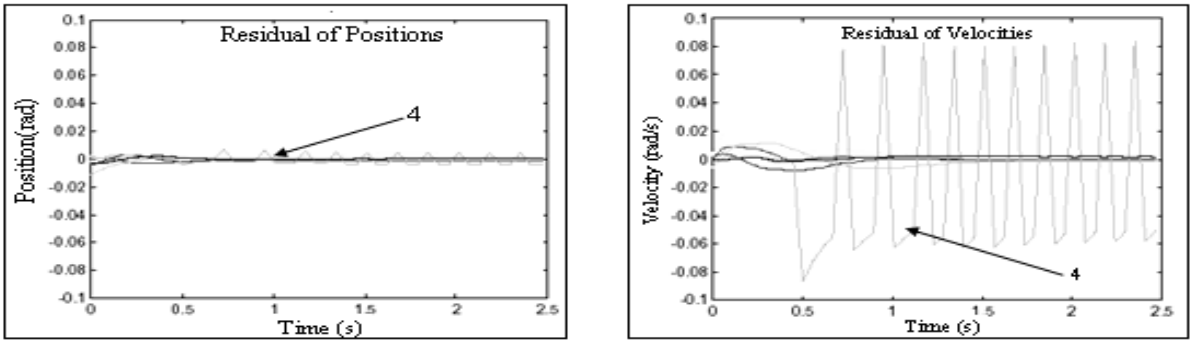
Set	CPN Architecture	MLP Architecture
Number of false alarms	9	4
Number of faults not detected	0	1
Trajectories without isolation errors	93.19 %	96.74 %

Fig. 6(a) and (b) shows the normalized joint positions and velocities and the respective MLP outputs of the 3 first joints in a simulation with fault 1 occurring at $t = 0.524$ s. Fig. 7(a) and (b) shows the residuals for the same trajectory. The reader will note that the position residuals are not as significant as the velocity residuals. Fig. 8 shows the CPN outputs for the same trajectory and Fig. 9 shows the fault isolation after the fault criterion is applied.



(a) (b)

Figure 6: Trajectory with a non-additive fault in joint 4 at 0.524 s
a) Normalized positions and respective *MLP* outputs (x);
b) Normalized velocities and respective MLP outputs (x).
The indicated lines (No. 4) are associated with joint 4.



(a) (b)

Figure 7: Residuals of the trajectory with fault.

The indicated line (No. 4) represents the residuals of joint 4.

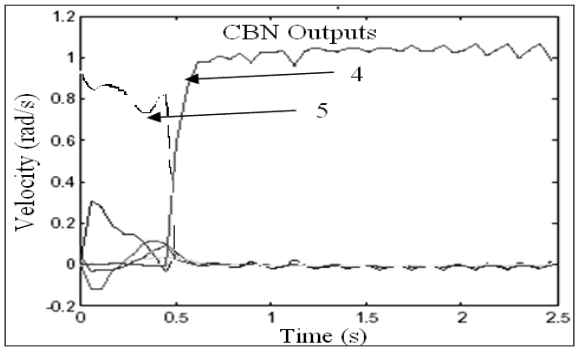


Figure 8: CPN outputs for the velocity residuals presented in Figure 7.

The line No. 4 is associated with fault in joint 4 and the line No. 5 with fault-free operation.

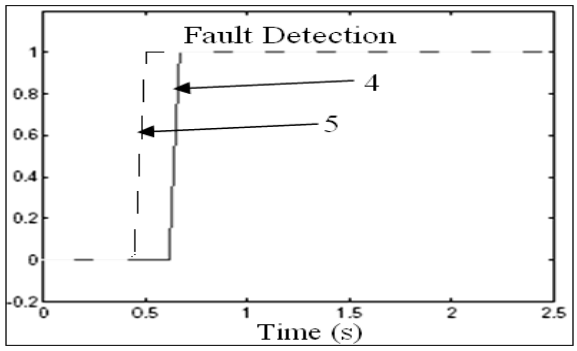


Figure 9: Detection of fault for the CPN outputs presented in Figure 8.

The line No. 4 and the line No. 5 are actual fault.

4.3 Analysis of the results

The *ANN*-based *FDI* system presents very satisfactory results when applied in a simulation of the Scorbot *ER* 5u plus five links robotic manipulator for consider non-additive faults (faults which are due to mechanical failure of the robot such as decoder failure, motor faults, etc.). It can detect and isolate faults that occur in trajectories that do not belong to the training set. The *MLP* reproduces the dynamic behavior of the fault-free robot with a small residual signal. It is important to remember that the quality of the *MLP* training (residual generation) is very important for the performance of the *FDI* system. Residual analysis is better performed by *CPN* using only velocity residuals when compared to *CPN* using the position and velocity residuals because the position residuals is not as significant as the velocity residuals for the Scorbot ER 5u plus manipulator.

Additionally, using only the velocity residual requires a smaller *CPN*. This is interesting in the sense that a smaller number of counter units, necessary to cope with a smaller input dimension, leads to shorter training and running times. The *CPN* architecture also outperforms the *MLP* architecture for the task of residual analysis. This occurs because the classification made by the *CPN* is more robust than that by the *MLP*. Recall that the *CPN* classifies the patterns according to the distance between them and the counter unit centers, while the *MLP* classifies the patterns according to decision surfaces that are placed according to the training patterns. The *MLP* decision surfaces are placed too close to the "edges" of the classes, while they could be moved to more conservative regions for more robust classification [28].

Another advantage of the *CPN* is that the time spent for training it is smaller than the time spent for training the *MLP*. One disadvantage is that the time for running the *CPN* is greater than the time for running the *MLP* because the number of counter units (and the time of processing) in the former are generally greater than the number of hidden units in the latter. It is important to emphasize that the results presented are somewhat dependent on the choice of the parameters in each training procedure. Also, fault isolation is difficult to perform when different faults occupy the same region in the residual space.

An attractive of the *FDI* system via *ANN* is that new faults can be detected and isolated training only the second *ANN* (for residual analysis). As the *CPN* presents a fast training procedure, this can be interesting.

5. CONCLUSION

A basic requirement for fault diagnosis is a reliable detection of even small faults. In this article, fault detection and isolation for mechanical manipulators has been proposed based on the theory of artificial neural networks. Simulation examples are used to illustrate the effectiveness of the algorithm in detection and isolation of faults in a five axis Scorbot 5u plus robotic manipulator. Training of neural network is more important in the functioning of system. Non-additive faults with different trajectory have been considered with varying the position and velocity profile. With an extensive simulation study results, we were able to conclude that *ANN*s are a powerful means for *FDI* tasks, robustly performing both residual generation and residual analysis. The comparison of different *ANN* architectures for residual analysis led us to conclude that feeding position residuals into the *ANN* does not bring about performance enhancements over pure velocity residuals. The acceptance of random trajectories would be desirable, training time and computational effort that would be required to train an appropriate network off line decreases as the task becomes simpler.

Further work on this is to extend the *FDI* scheme to robotic manipulators with a larger number of degrees of freedom. Other different types of faults can be detected and isolated. Fault detection and isolating of robotic manipulator via fuzzy logic, neuro-fuzzy control, hybrid system and expert system. The work presented here can be expanded to include post failure control of the robotic manipulator in a hybrid system frame work**.**

REFERENCES

[1] Pdrs, M. (1988). *Payload deployment and retrieval system malfunction workbook,* Technical Manual, NASA Johnson Space Center, Houston

[2] English, J. D.; Maciejewski, A. A. (1998). Fault tolerance for kinematically redundant manipulators: anticipating free-swinging joint failures, *IEEE Transaction Robot Automation,* Vol. 14, 566-572

[3] Edward, Y. C.; Willsky, A. S. (1984). Analytical redundancy and the design of robust failure detection systems, *IEEE Transactions on Automatic Control,* Vol. Ra. 29, No. 7

[4] Fabrizio, C.; Walker, D. I. (1997). Observer-based fault detection for robot manipulators, *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 4, No. 3
[5] Frank, P. M. (1990). Fault diagnosis in dynamic system using analytical and knowledge based redundancy-a survey and some new results, *Automatica,* Vol. 26, No. 3, 459-474
[6] Michael, A. D.; Marios, M. P. (1998). Incipient fault diagnosis of dynamical systems using online approximations, *IEEE Transactions on Automation Control,* Vol. 43, No. 11
[7] Isermann, R. (1984). Process fault detection based on modeling and estimation methods-a survey, *Automatica,* Vol. 20, No. 4, 387-404
[8] Gertler, J. J. (1988). A Survey of model-based failure detection and isolation in complex plants, *IEEE Control Systems Magazine,* Vol. 8, No. 6, 3-11
[9] Willsky, A. S. (1976). A survey of design methods for failure detection in dynamic systems, *Automatica,* Vol. 12, No. 6, 601-11
[10] Patton, R.; Chen, J. (1991). Robust fault detection using eigen structure assignment: a tutorial consideration and some new results, *Proceedings of the 30th Conference on Decision and Control*, 2242-2247
[11] Patton, R. (1994). Robust model-based fault diagnosis: the state of the art, *Proceedings of SAFEPRO*-CESS'94, Vol. 1, 1-10
[12] Visinsky, M. L.; Walker Cavallaro, I. D. (1994). New dynamic model-based fault detection thresholds for robot manipulators, *IEEE International Conference Robotics and Automation,* Vol. 2, No. 8-13, 1388-1395
[13] Wunnenberg, J.; Frank, P. M. (1990). Dynamic model based incipient fault detection concept for robots, *Proceedings of the 11th IEAC World Congress on Automatic Contro*l, Tallin, Estonia, 61-66
[14] Carreras, C.; Walker, I. D. (2001). Interval methods for fault-tree analysis in robotics, *IEEE Transactions Robotics and Automation*, Vol. 50, No. 1, 3-11
[15] Ting, Y.; Tosunoglu, S.; Tesar, D. (1999). A control structure for fault tolerant operation of robotic manipulators, *Proceedings IEEE International Conference on Robotics Automation,* Atlanta, GA, 684-690
[16] Ralph, S. K.; Pai, D. K. (1999). Computing fault tolerant motions for a robot manipulator, *Proceedings IEEE International Conference on Robotics and Automation*, Detroit, MI, 486-493
[17] Park, J.; Chung, W. K.; Youm, Y. (1996). Failure recovery by exploiting kinematic redundancy, *Proceedings 5th International Workshop Robot Human Communication,* Tsukuba, Japan, 298-305
[18] Yi, Y.; Mcinroy, J. E.; Chen, Y. (2006). Fault tolerance of parallel manipulators using task space and kinematic redundancy, *IEEE Transactions and Robotics,* Vol. 22, No. 5, 1017-1021
[19] Roberts, R. G. (2001). The dexterity and singularities of an under actuated robot, *Journal of Robotic System*, Vol. 18, No. 4, 159-169
[20] Dailey, K. W. (2004). *The FMEA Handbook,* DW Publishing, New York
[21] Winfield, A. F. T.; Nembrini, J. (2006). Safety in numbers: fault tolerance in robot swarms, *International Journal of Modeling, Identification and Control*, Vol. 1, No. 1, 30-37
[22] Winfield, A. F. T.; Sa, J.; Gago, M. C.; Fisher, M. (2005). Using temporal logic to specify emergent behaviours in swarm robotic systems, *Proceedings of Towards Autonomous Robotic Systems (TAROS)*, London
[23] Dixon, W. E.; Walker, I. D.; Dawson, D. M.; Hartranft, J. P. (2000). Fault detection for robot manipulators with parametric uncertainty: a prediction- error-based approach, *In IEEE Transactions Robotics. Automation*, Vol. 16, No. 6, 689-699
[24] Balajee, K.; Lynne, E. P. (2007). *Fault tolerance based metrics for evaluating system performance in multi robot teams,* Distributed Intelligence Laboratory, The University of Tennessee, Knoxville, 54-61
[25] Yavnai, A. (2000). Metrics for system autonomy. Part I: metrics definition, *Proceedings of Performance Metrics for Intelligent Systems (PerMIS),* Vol. Part II.
[26] Finkelstein, R. (2000). A Method for evaluating iq of intelligent systems, *Proceedings of Performance Metrics for Intelligent Systems (PerMIS),* Vol. Part II.
[27] Kohonen, T. (1995). *Self-Organizing Maps*, Springer-Verlag, Berlin
[28] Leonard, J. A.; Kramer, M. A. (1991). Radial basis function networks for classifying process faults, *IEEE Control Systems*, Vol. 11, No. 3, 31-38