Intelligence Decision Making of Fault Detection and Fault Tolerances Method for Industrial Robotic Manipulators

D. Sivasamy, M. Dev Anand, K. Anitha Sheela

Abstract--- FD 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 failure. Based on the failure an intelligent decision is taken with the help of the neuro fuzzy fault diagnosis system. The main aim of this work is the introduction of a new algorithm for robots fault detection which forms part of a proposed intelligent decision making framework for fault tolerance in robotic manipulator. In developing the model, this work explores the affects of failures in an example robot using a technique called Neuro-Fuzzy Approach. The robot components critical to fault detection are revealed using a Neuro-Fuzzy (NF) approach. To evaluate our NF based fault detection and tolerance method we performed an extensive simulation study with a Scorbot ER 5u plus robot manipulator. In this research work we considered all faults possible to occur in robot manipulator. The Scorbot ER 5u plus model was developing in robotics toolbox for MATLAB using the NF algorithms.

I. FAULT TOLERANCE IN ROBOT

Robots are often used in inaccessible or hazardous environments in order to alleviate some of the 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. This research builds a foundation for fault tolerant robots by developing new intelligent algorithms which detect hardware failures in the robot system and trigger the appropriate fault tolerant actions. Many fault tolerant systems have been developed for computer, airplane, and industrial systems [Collacott (1977), Galler and Slenski (1991), Kieckhafer (1988), Merrill (1988), Wensley and Harclerode (1987b)]. Several of these techniques have provided models for robotic fault tolerance schemes such as those presented in [Valavanis (1991)]. However, the trend in robotics seems to be to use only those schemes which rely on physical redundancy of components. Many methods of fault tolerance exist which do not alter the physical system.

II. COMPUTER FAULT TOLERANCE

A common method used to provide fault tolerance in computer systems is Triple Modular Redundancy (TMR) [Nelson (1990)] in which three processors all work on the same problem and compare their results. If one of the processors is faulty and its result does not agree with the results of the other two processors, the faulty processor is voted out of the final decision and the correct result is passed on to the rest of the system. Only one faulty processor can be tolerated by this system, however. More failures can be detected and isolated by increasing the number of redundant components. To avoid adding a multitude of redundant parts to computer systems, other methods were developed which reconfigure the data or code in the computer among the working parts once one part has failed [Visinsky *et al* (1991)]. The literature discusses time redundancy in which a computational cycle is lengthened so a fault free part (or parts) will have enough time to handle the tasks of a faulty component. Other systems use set-switching or processor-switching schemes [Chean and Fortes (1990)] for reconfiguration. In software, arithmetic codes are used to find and correct errors in matrix computations like those performed in robot kinematics [Han (1990)]. Check bits and error correction codes help monitor data transmissions and allow a reconstruction of the original data if the transmission line is faulty. In [Chow and Willsky (1984)], develop a useful mathematical approach for determining the various redundancies that are relevant to the failures under consideration. Robot diagnosis, generally speaking, includes fault detection, fault isolation and fault identification [Coghill and Shen (2001)]. The most powerful approaches are those using a process model, where quantitative and qualitative knowledge based models, data based models, or combinations thereof are applied [Frank *et al* (2000)]. According to [Schroder (2003)] proposed qualitative approach to fault diagnosis of dynamical systems, mainly process control systems. However, most of current fault diagnosis approaches focus on one of robot fault categories, hardware failure, or faults caused by modelling errors or uncertainty.

Revised Manuscript Received on July 10, 2019.

D. Sivasamy*, External Research Scholar, ECE, Jawaharlal Nehru Technological University, Hyderabad-85, Telangana, India. (e-mail: sivasamy.d@gmail.com)

M. Dev Anand, Professor & Research Director, Department of Mechanical Engineering, Noorul Islam Centre for Higher Education, Kumaracoil, Thuckalay, Kanyakumari District, Tamil Nadu, India.

K. Anitha Sheela, Professor & Head, ECE, Jawaharlal Nehru Technological University, Hyderabad-85, Telangana, India.

III. ROBOT COMPUTER ARCHITECTURE FAULT TOLERANCE

The original focus for the work is an eight joint, kinematically redundant robot with a proposed parallel VLSI architecture [Walker and Cavallaro (1991)] to compute real time control for the robot. The opportunity for parallel implementation of the robotic algorithms has been exploited in the design [Hamilton (1991)] and could provide a valuable foundation for tolerance of processor failures within the controller. A traditional approach to dynamic reconfiguration for arrays, sometimes called set-switching [Chean and Fortes (1990)] removes an entire row, column, or diagonal of the array to isolate the faulty processor. The algorithm is then modified to deal with the new dimensions of the mesh. Only a few errors can be tolerated before the mesh is reduced to an unusable size. This dynamic reconfiguration method isolates just the faulty processor and its communication links and then reassigns the faulty processor's data to the fault free neighbors.

IV. REDUNDANCY BASED ROBOTIC FAULT TOLERANCE

Previous work on fault tolerance for the mechanical aspect of robots has concentrated on those 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.) with only token thought going to the more critical, system wide effects of the failures. 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. If the robot is performing a time critical or delicate task, fault tolerance must allow the robot to get a run-away motor under control quickly before any damage to the environment or the robot occurs. The fault tolerant advantages of redundancy have also led to adding extra parallel structures, such as a backup arm or leg [Tesar (1990)], in order to allow many different reconfiguration possibilities in the presence of a failure.

Redundant components offer an obvious solution to the reconfiguration problem by providing a backup if one of the components fail. As in Triple Modular Redundancy (TMR) with computers, redundancy may also give the robot system multiple components to check and vote among, thus improving fault detection

V. KINEMATIC REDUNDANCY FAULT TOLERANCE

Many robots today have the advantage of being kinematically redundant. 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 [Walker and Cavallaro (1991)]. In [Maciejewski (1998)] has quantified the effect of joint failure on the remaining dexterity of a kinematically redundant manipulator. Robot controllers may further attempt to ease the transition through singular configurations for the robot [Deo (1992)]. A configuration is considered singular if the robot is fully extended or folded in on itself in such a way as to hinder motion in one direction without rapid changes in one or more joint positions. Fault detection routines might interpret these jumps in the joint velocities as failures in the robot and erroneously shut down a fault free system. The optimal damped least squares technique used in the Singularity Robust Inverse (SRI) algorithm described in [Deo (1991)] ensures feasible joint velocities with minimum end effector deviation from the specified trajectory. This new inverse kinematics scheme enables the manipulator to avoid drastic joint motions at or near singular configurations and helps eliminate false alarms in the fault detection algorithms. By using Deo's SRI algorithm in the robot controller, the velocities of the robot during singular configurations are moderated eliminating possible false alarms in the detection routines.

In addition to possessing a number of other important properties, kinematically redundant manipulators are inherently more tolerant to locked-joint failures than non-redundant manipulators. However, a joint failure can still render a kinematically redundant manipulator useless if the manipulator is poorly designed or controlled. This work presents a method for identifying a region of the workspace of a redundant manipulator for which task completion is guaranteed in the event of a locked-joint failure. The existence of such a region, called a *failure-tolerant workspace*, will be guaranteed by [Rodney (2007)] imposing a suitable set of artificial joint limits prior to a failure.

VI. ANALYTICAL REDUNDANCY BASED FAULT DIAGNOSIS & RESULTS

Analytical redundancy is another concept for failure detection and isolation which uses only the available sensor components in a system to generate residuals from which failures can be identified. In [Stengel (1991)] give thorough reviews of the various methods of analytical redundancy. By comparing the histories of sensor outputs versus the actuator inputs, results from dissimilar sensors can be compared at different times in order to check for failures. The design and analysis of fault diagnosis architectures for robotic systems using the model based analytical redundancy approach [Edward and Willsky (1984), Fabrizi and Walker (1997), Michael *et al* (1998) and Frank *et al* (2000)] have 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 [Frank (1990)], parameter estimation [Isermann 1991)] and parity relations [Gertler (1988)], yet most of the current techniques developed rely on the

assumption that the process is linear in nature [Willsky (1976), Patton and Chen (1991), Patton (1994)]. The appeal of the model based approach [Visinsky *et al* 1994] lies in the fact that the redundancy required for detecting faults is created using powerful information processing techniques without the need for additional physical instrumentation in the system. A number of studies have been dedicated to the assessment and analysis [Carreras and Walker (2001)] of robot reliability. Other studies related to enhancing a robot's tolerance to failure include work on layered failure tolerance control [Ting (1993)], failure tolerance by trajectory planning [Ralph and Pai (1999)], kinematic failure recovery [Park *et al* (1996)] and manipulators specifically designed for fault tolerance [Yi *et al* (2006)]. 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 case can be found in [Roberts (2001)].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), [Dailey (2004)]. The likelihood that random errors cause undesirable behaviours can be reduced, in the first instance, by employing high reliability components. The most interesting conclusion is that a multi-state reliability model is needed to account for the partially failed robots identified by [Winfield and Nembrini (2006)] 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 [Winfield *et al* (2005b)] they explore the use of the temporal logic formalism for specification and possibly proof of correctness of emergent behaviours. In addition to that [Dixon (2000)], 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.

Unavoidable modelling uncertainties, which arise owing to modelling errors, time variations, measurement noise and external disturbances cause deterioration in the performance of fault detection schemes by [Wunnenberg (1990)] causing false alarms. One way to deal with the absence of a mathematical model is to build a model from input output data. Recent techniques involve the use of neural networks or fuzzy systems for this purpose. In [Chen and Lee (2002)], for instance, Radial Basis Function (RBF) and perception neural networks are used for process modelling. In [Vemuri and Polycarpou (1997)] use neural networks for fault detection and isolation, which utilize learning methodology that is based on a nonlinear nominal model of the manipulator and nonlinear faults. In [Shin and Lee (1999)], a robust tracking controller/fault detection scheme was proposed that utilized the full dynamic model of the robot manipulator. Unfortunately, the fault detection residuals are based on conservative thresholds, which are obtained by taking the norm of user defined upper bounds for the position and velocity tracking errors. As regards position and velocity, the control of robots in the neural network approach to manipulator fault detection was adopted in [Chen (1999)]; however, the fault detection algorithms are based on user defined bounds in the modelling uncertainty. Many efficient control concepts have been developed and put into practice within the last 20 years [Sajidman *et al* (1995), Kuntze (1988), De Luca (2000); Mbede *et al* (2000)]. Both model based approaches (e.g., inverse system technique, adaptive algorithm, predictive control) and heuristic fuzzy or adaptive fuzzy approaches applied to rigid and elastic robot structures have been proposed. The state of the art for robot control concepts with external sensory is quite different. While for some special industrial applications with force/torque and visual sensors [Kuntze and Lubbert (1995), Yoshikawa (2000)], considerable results have been achieved, there remains a lack of generic multi sensor based surveillance and control concepts. Obviously, a Point-to-Point (PTP) motion has to be controlled by a different algorithm rather than a force controlled de-burring or hole fitting operation. Being able to identify the extent of fault-tolerance in a system would be a useful analysis tool for the designer [Balajee and Lynne (2007)]. Unfortunately, it is difficult to quantify system of fault-tolerance. Other related works include [Yavnai's (2000)] approach for measuring autonomy for intelligent systems and Analytical Hierarchy Process (AHP) [Finkelstein's (2000)] for measuring system intelligenceThis necessitates the development of the fault diagnosis algorithm, which has the ability to detect manipulator failures in the presence of modeling uncertainties. Such algorithms are referred to as robust fault diagnosis schemes. Generally, the fault detection and isolation process is viewed as [Frey (2004)] consisting of two stages: residual generation and decision making, as shown in Figure 1. Outputs from the sensory are processed and compared with the expected values from the quantitative nominal model; the resulting value is referred to 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 fault detection and isolation system can be achieved by designing a robust residual generation process.

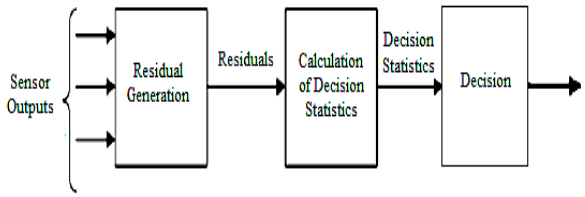


Figure 1: Two Stage Structure of Decision Statistics

Numerous advantages characterize this method:

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

Neuro Fuzzy Systems

Recently, the combination of neural networks and fuzzy logic has received attention. The idea is to lose the disadvantages of the two and gain the advantages of both. Neural networks bring into this union the ability to learn. Fuzzy logic brings into this union a model of the system based on membership functions and a rule base. This field of study is still in its infancy. Determining the fuzzy membership functions from sample data using a neural network is the most obvious method of using the two together. The definition of the membership function has a huge impact on the system response. Often, the programmer must use trial and error to find acceptable values. Assuming a certain shape and finding the beginning and endpoints for the fuzzy values in a fuzzy set is a neural network optimization problem [Nauck *et al* (1993)]. Figure 3. is a diagram of such a system.

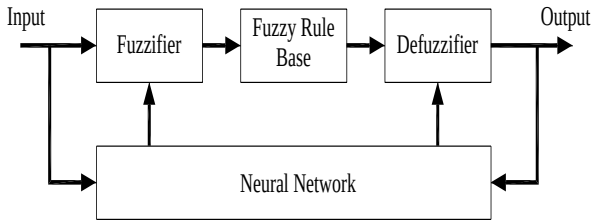


Figure 2: A Fuzzy System Whose Membership Functions are Adjusted by a Neural Network

Figure 3. Shows a more complex integration - the use of neural networks to determine both the fuzzy membership functions and the rule base. The nonlinearity of the membership functions is unique to membership functions derived by neural networks. They help minimize the number of rules.

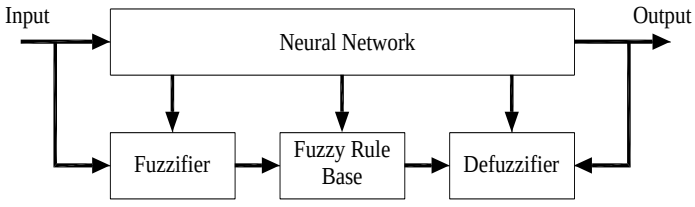


Figure 3: A fuzzy System Defined by a Neural Network

Another approach is to incorporate fuzzy logic into the neurons of the neural networks. This approach developed because of the original neuron model proposed by [McCulloch and Pitts (1943)]. The McCulloch and Pitts cell produced an all-or-none output. It was quickly realized that neurons with output in the range of [0, 1] produced much better results. The concept of a fuzzy neuron, however, has advanced beyond simply expanding the range of outputs on a crisp neuron. Some researchers have incorporated membership functions and rule bases into the individual neurons, as shown in Figure. 4.

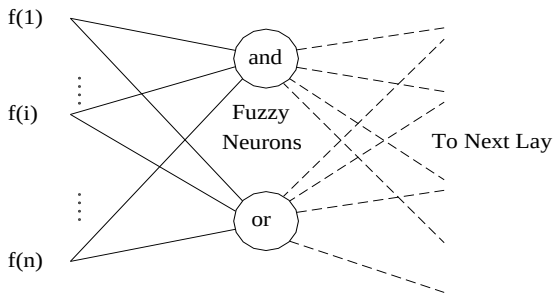


Figure 4: A Neural Network of Fuzzy Neurons

The idea of fuzzification of control variables into degrees of membership in fuzzy sets has been integrated into neural networks as shown in Figure 5. If the inputs and outputs of a neural network are fuzzified and defuzzified, significant improvements in the training time, in the ability to generalize, and in the ability to find minimizing weights can be realized. Also, the membership function definition gives the designer more control over the neural network inputs and outputs.

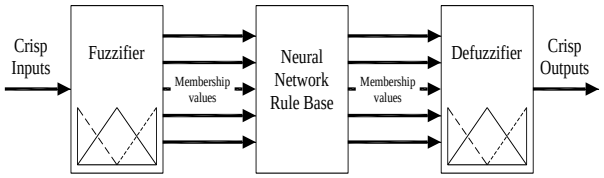


Figure 5: A Fuzzy System with Neural Network Rule Base

The implementation of a fuzzy module for residual evaluation can be very difficult with an increasing number of residuals taken into account. The problem of finding appropriate membership functions and rules is often a tiring process of trail and error. The model of neuro fuzzy system for feature evaluation is shown in Figure .6. Just like linear classifiers fuzzy systems require in contrast to ANNs manual tuning to obtain good classification results. In order to automate the design phase of the entire system in the scheme of neuro fuzzy, approaches are used for designing residual evaluation modules

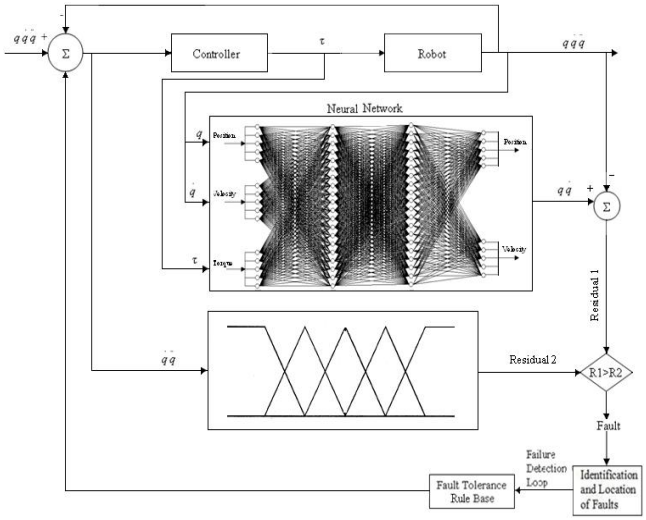


Figure 6: Neuro Fuzzy System for Feature Evaluation

Robot Model

The dynamic nonlinear equations of an *n* degree-of-freedom robot manipulator in the continuous time

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

$$\Delta T = T_{Model} - T_{Measured}$$

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 or un-modeled dynamics.

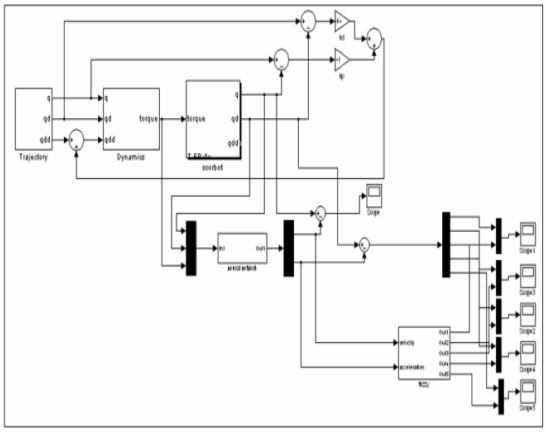


Figure 7: Neuro Fuzzy Based Fault Diagnosis Decision Making Model Using Simulink

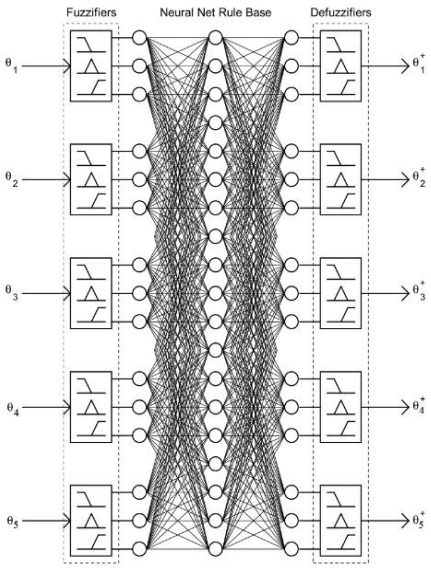
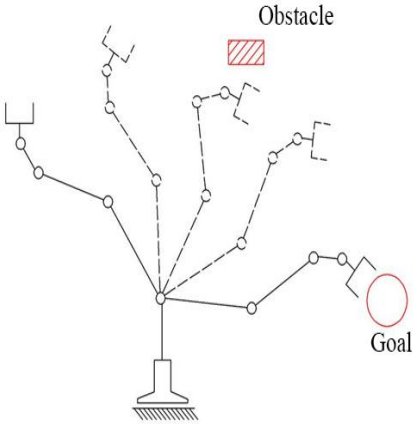


Figure 8: A Sample Neuro-Fuzzy System



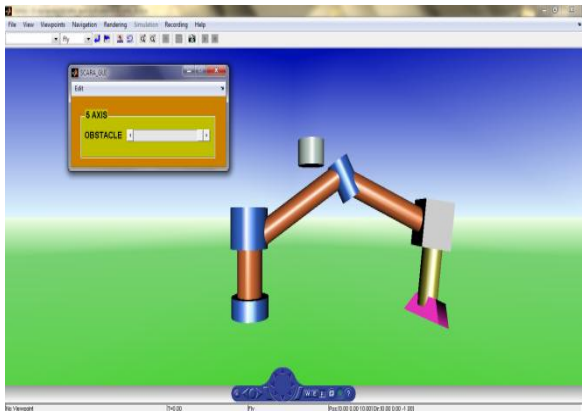


Figure 9: A Sample Training Trajectory Obtained from the Simulator

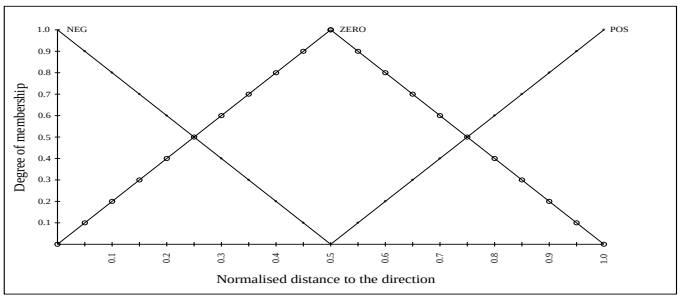


Figure 10: The Fuzzy Membership Function Definition NF1

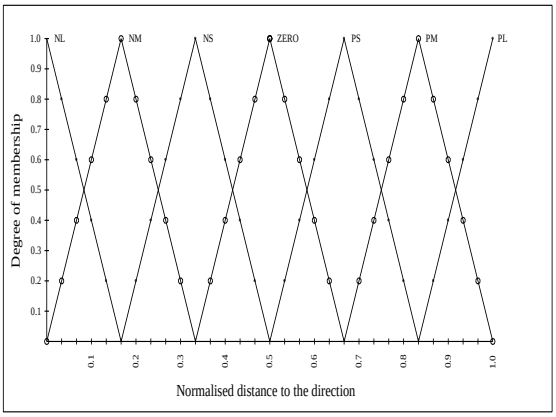


Figure 11: The Fuzzy Membership Function Definition NF2

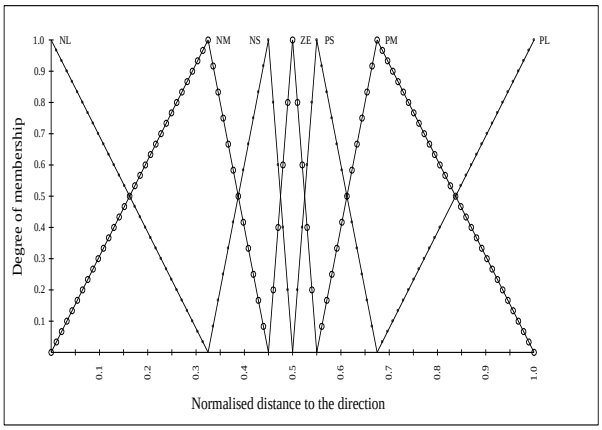


Figure 12: The Fuzzy Membership Function Definition NF3

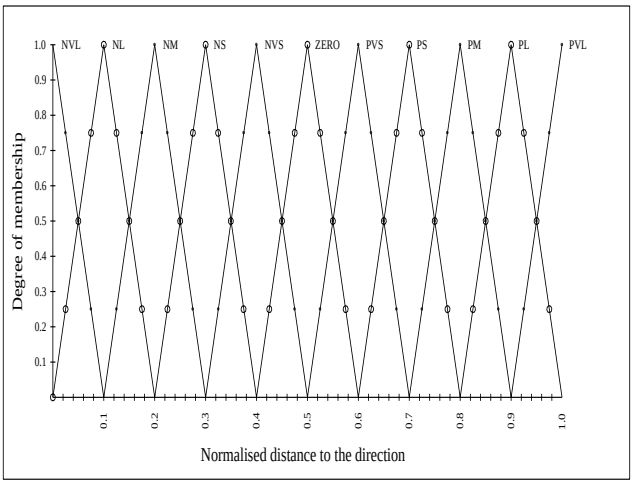


Figure 13: The Fuzzy Membership Function Definition NF4

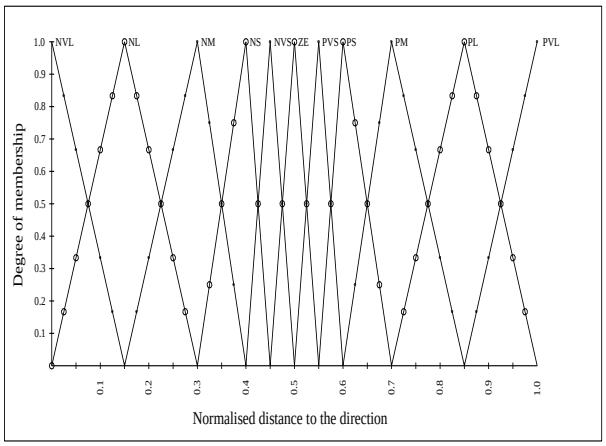


Figure 14: The Fuzzy Membership Function Definition NF5

3-D Surface Plots Obtained for All Joint Angles of 5-DOF Industrial Manipulator

The following Figures: 16-20 shows surface plot of nine neuro fuzzy relating inputs with joint angles of 5-DOF Redundant manipulator. Figure. 15. Indicates the surface plot between nine input versus θ_1. It shows that when the values of y and z moving in a positive direction, there is a marginal increase followed by a decrease in surface plot of θ_1 is shown in Figure 16. The Figure depicts that the value of θ_2 values. The inputs-output θ_2 increases linearly when moving in the positive direction of y coordinate to some values of y and then there is a sudden increase of $\theta_2$values. No significant change in the value of is observed with change in values of z coordinate. By moving from negative direction to the positive direction of x and y coordinates, the $\theta_3$value decreases first then followed by slightly θ_2 increase, can be easily conclude from Figure 17. Similarly the surface plot of θ_4 with input variables x and z coordinate is depicted in Figure 18. It shows that the value of inputs has significant effect in determining the value of. It concludes from the surface plot that the contribution of interdependent parameters toward obtaining the output can easily provide θ_5 through the neuro fuzzy programming and can be hardly obtained otherwise without employing massive computations. All the surface viewer plots show that the total surface is covered by the rule base.

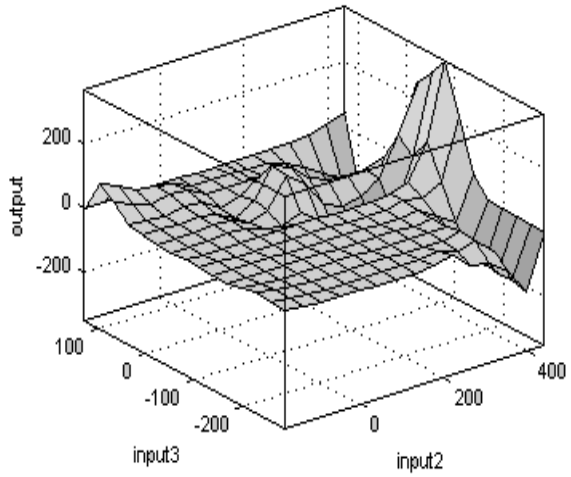


Figure 15: Surface Plots for θ_1

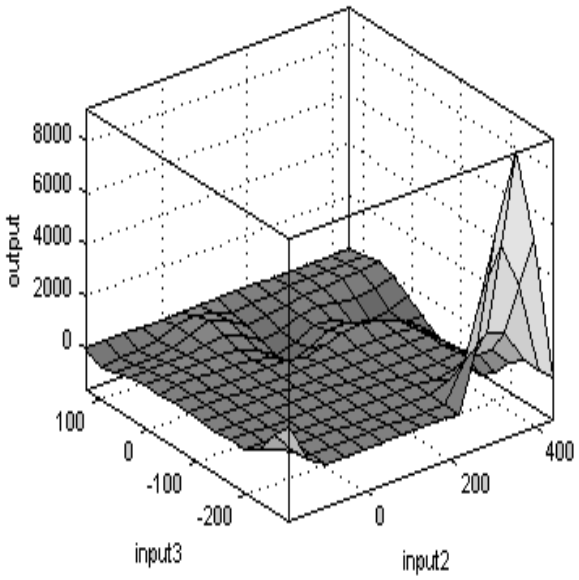


Figure 16: Surface Plots for θ_2

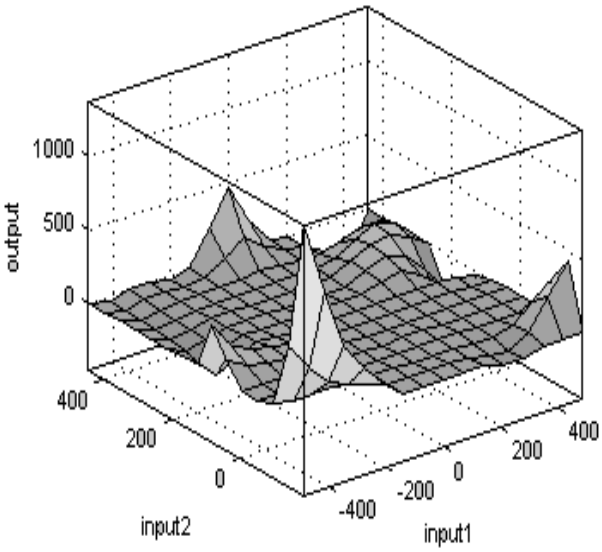


Figure 17: Surface Plot for θ_3

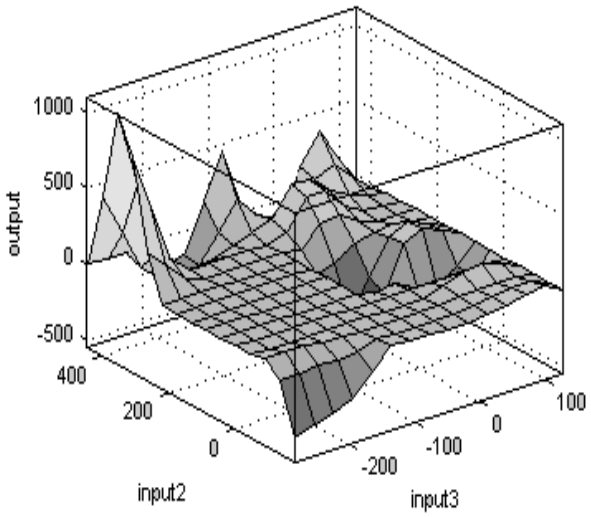


Figure 18: Surface Plots for θ_4

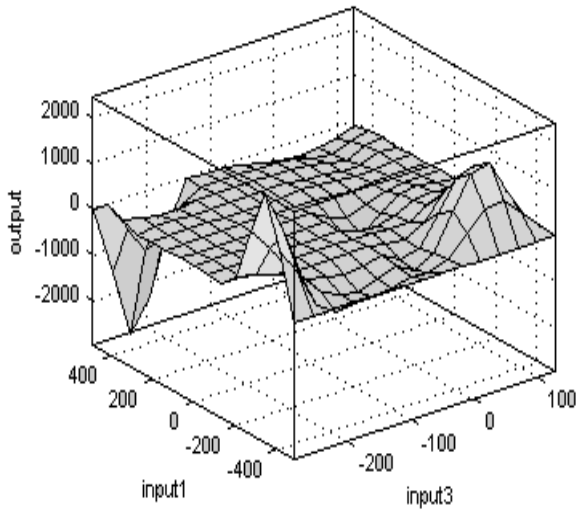


Figure 19: Surface Plots for θ_5

VII. CONCLUSION

In this study, the inverse kinematics solution using neuro fuzzy for a 5-DOF industrial manipulator is presented. The field of neuro-fuzzy technology will become an important part of intelligent control. The ability to learn how to control a process from sample data is its biggest asset. In this report, nine neuro-fuzzy controllers were trained to emulate a human's example control of a robotic arm.The difference in joint angle deduced and predicted with neuro fuzzy model for a 5-DOF industrial manipulator clearly depicts that the proposed method results with an acceptable error.The modelling efficiency of this technique was obtained by taking three end-effector coordinates as input parameters and five joint positions for a 5-DOF industrial manipulator respectively as output parameters in training and testing data of NF models. Also, the neuro fuzzy model used with a smaller number of iteration steps with the hybrid learning algorithm. Hence, the trained neuro fuzzy model can be utilized to solve complex, nonlinear and discontinuous kinematics equation complex robot manipulator; thereby, making neuro fuzzy an alternative approach to deal with inverse kinematics. The analytical inverse kinematics model derived always provide correct joint angles for moving the arm end-effector to any given reachable positions and orientations. As the neuro fuzzy approach provides a general frame work for combination of NN and fuzzy logic. The efficiency of neuro fuzzy for predicting the inverse kinematics of industrial manipulator can be concluded by observing the 3-D surface viewer, residual and normal probability graphs. First, the membership function definitions are an important part of the neuro-fuzzy system. Second, the fuzzification of a neural network's inputs and outputs allows neural networks to learn more complex functions than ever before. The performance of the neuro-fuzzy controllers in this specific application, however, is less than perfect. A trained neuro-fuzzy system is only as good as the training data used to train it. The use of neuro-fuzzy systems for control has been examined. It is the opinion of this researcher that fuzzification of a neural network's inputs and outputs will become standard procedure in neural network applications.

REFERENCES

1. SCORBOT-ER VII User's Manual, 3rd Edition, Intelitek Inc., Catalog # 100016 Rev. C, February 1996.
2. Paul, R. P: Robot Manipulators: Mathematics, Programming and Control, Cambridge, ITS Press, 1981.
3. Luke Cole, Adam Ferenc Nagy-Sochacki, Jonathan Symonds: Drawing Using the Scorbot - ER VII Manipulator Arm, October 29, 2007.
4. D. Constantinescu; E.A. Croft: Smooth and Time Optimal Trajectory Planning for Industrial Manipulators along Specified patHs, 'Journal of Robotic Systems', Vol. 17, No. 5, 2000, 33-249.
5. H. Karagulle; L. Malgaca: Analysis of End Point Vibrations of a Two-Link Manipulator by Integrated CAD/CAE Procedures, 'Elsevier Finite Elements in Analysis and Design', Vol. 40, 2004, 2049–2061.
6. Dr. Anurag Verma; Vivek, A; Deshpande: End-effector position analysis of Scorbot-Er Vu Plus Robot, 'International Journal of Smart Home', Vol. 5, No. 1, January, 2011.
7. John, Q; Gan; Eimei Oyama; Eric, M; Rosales and Huosheng Hu: A Complete Analytical Solution to the Inverse Kinematics of the Pioneer 2 Robotic Arm, 'International Journal of Robotica', Vol. 23, 2005, 123–129.
8. Khaled fawaz; Rochdi Merzouki; Belkacemould-Bouamama: Model Based real time monitoring for collision detection of an industrial robot, 'Elsevier Mechatronics', Vol.19, 2009, 695–704.
9. Lee, H.S; S.L. Chang: Development of a CAD/CAE/CAM System for a Robot Manipulator, 'Journal of Materials Processing Technology', Vol. 140, 2003, 100-104.
10. Lee; Eric; Constantinos Mavroidis: Geometric Design of Spatial PRR Manipulators, 'Mechanism and Machine Theory', Vol. 39, 2004, 395-408.
11. Su; Hai-Jun; J. Michael McCarthy: The synthesis of an RPS Serial Chain to Reach a Given Set of Task Positions, 2003.
12. Colbaugh, R; K. Glass: Adaptive Tracking Control of Rigid Manipulators Using Only Position Measurements, 'Journal of Robotic Systems', Vol. 14, No.1, 1997, 99-26.

13. Farrington, P.A; Nembhard, H.B; Sturrock, D.T; and Evans, G.W: Increasing the Power and Value of Manufacturing Simulation via Collaboration with othEr Analytical Tools, A Panel Discussion, 'Proceedings of the Winter Simulation Conference', 1999.
14. MATLAB and Simulink for Technical Computing, The MathWorks Inc., USA. [Online]: http://www.mathworks.com/.
15. P.I. Corke: A Robotics Toolbox for MATLAB, '*IEEE Robotics* Automation Mag.', Vol. 3, No.1, 1996, 24–32.
16. R K Mittal; J Nagrath: Robotics and Control, Tata McGraw-Hill, 2005.
17. Tenreiro Machado, J.A., Martins de Carvalho, J.L. and Alexandra M.S.F. Galhano, Analysis of Robot Dynamics and Compensation Using Classical and Computed Torque Techniques, IEEE Transactions on Education, 36, (4), 1993.
18. Domenico Prattichizzo and Antonio Bicchi, Dynamic Analysis of Mobility and Graspability of General Manipulation Systems, IEEE Transactions on Robotics and Automation, 14, (2), 1998.