INVERS KINEMATIC MAPPING OF 6 DoF ARTICULATOR USING ANFIS (ADAPTIVE NEURO-FUZZY INFERENCE SYSTEM)

Edwar Yazid and Estiko Rijanto
Pusat Penelitian Tenaga Listrik dan Mekatronik (Puslit Telimek),
LEMBAGA ILMU PENGETAHUAN INDONESIA (LIPI),
Research Center for Electrical Power and Mechatronics,
INDONESIAN INSTITUTE OF SCIENCES,
Jl. Cisitu No.21/154D, Bandung 40135, Indonesia
Email: edwar_yazid@plasa.com

Abstract

Well known methods for the resolution in Inverse Kinematics use a pseudo-inverse's approach. The solution in general can only be obtained numerically which is computationally expensive. Hence, these approaches are oftentimes not suitable for real time applications. Furthermore, these methods operate at the velocity level and not at the position level so that global properties of the workspace cannot easily be derived. Other methods are based on optimization of some motion criteria like minimal joint motion or minimum energy. These methods, like the pseudo-inverse based methods, are numerical, and have the same disadvantageous properties.

The simulation of application of Adaptive Neuro-Fuzzy Inference System (ANFIS) in Inverse Kinematics Mapping is discussed in this paper. The ANFIS is applied to build Inverse Kinematics mapping of a 6 DoF articulator. The articulator is constructed of four links with four DoF and a gripper-cutter mechanism with two DoF. This method does not require any constraints to be imposed on the articulator configuration, nor does it require any functions to be optimized. It works identically for non redundant and for redundant articulator. The solution is obtained in closed-form as opposed to numerical solutions of the well known methods. The simulation results show that most of the time is spent in evaluating the membership functions and deriving an output from the rule-base in the ANFIS. This required approximately 80% of the total Inverse Kinematics processing time, the rest of it for the computation of the elements of the Jacobian.

Keywords : *Articulator, neuro-fuzzy, Jacobian matrix, trajectory, inverse kinematics*

1. Introduction

In robotic motion planning and control, the solutions to the kinematics problems are essential to achieve the goals of a robotic operation. The forward kinematics problem in robotics is concerned with the transformation of position and orientation information in a joint space to a Cartesian space described by a forward kinematics equation as follows.

$$\underline{r}(t) = \underline{f}(\underline{\theta}(t)) \tag{1}$$

where $\underline{\theta}(t)$ is an m-vector of joint variables, $\underline{r}(t)$ is an n–vector of Cartesian variables, and $\underline{f}(\cdot)$ is a continuous nonlinear function whose structure and parameters are known for a given articulator. The inverse kinematics problem is to find the joint variables given the desired positions and orientations of the end-effector through the inverse mapping of the forward kinematics equation (1)

$$\underline{\theta}(t) = \underline{f}^{-1}(\underline{r}(t)) \tag{2}$$

Inverse kinematics solves the problem inverse to the one forward kinematics deals with. It maps an m-dimensional task space vector into an n-dimensional joint vector. The inverse kinematics problem is thus much more difficult to solve than the forward kinematics problem for serial-link articulators. For a redundant manipulator with n > m, the inverse kinematics is a one-to-many mapping. The most direct way to solve (2) is to derive a closed-form solution from (1). Unfortunately, obtaining a closed-form solution is difficult for most

ISBN 979-15577-7-1

articulators due to their nonlinearity of $f(\cdot)$. Moreover, the solution is often not unique for kinematically redundant articulators due to their redundancy. Making use of the relation between joint velocity $\dot{\theta}(t)$ and Cartesian velocity $\dot{r}(t)$ is a common indirect approach to the inverse kinematics problem. The velocity vectors $\dot{\theta}(t)$ and $\dot{r}(t)$ have the following linear relation.

$$\dot{r}(t) = J(\theta(t))\dot{\theta}(t) \tag{3}$$

where $J(\theta(t))$ is the Jacobian matrix and can be rank–deficient sometimes. In a kinematically redundant articulator, $m > n$. This indirect approach begins with the desired velocity of the end-effector $\dot{r}_d(t)$ based on a planned trajectory and required completion time T. As a requirement, $\dot{r}_d(0) = \dot{r}_d(t) = 0$. The corresponding joint position vector $\theta(t)$ is obtained by integration $\dot{\theta}(t)$ of for a given $\theta(0)$. The resulting $\theta(t)$ then is used to calculate articulator pose.

In this case, the problem becomes how to select one particular solution from multiple solutions. The most popular approach to this problem is pseudo-inverse, Newton's method which is computationally expensive. In recent years, soft computing such as fuzzy logic, neural network, etc. have been developed for solving various matrix algebra and optimization problems. In robotics research, much effort has been devoted to solve the inverse kinematics problem; e.g., [1], [2], [4], [5]-[9] by implementing soft computing separately and generally for redundant articulator. In this paper, fuzzy logic is combined with neural network, it is called Adaptive Neuro Fuzzy Inference System (ANFIS). This hybrid system is applied for the inverse kinematics mapping of a 6 DoF articulator which is designed and made by Research Center for Electrical Power and Mechatronics, LIPI. ANFIS attempts to learn inverse kinematics through the training data and then uses this to solve the inverse kinematics problem. After the learning, a direct mapping either from task space vector $r(t)$ to joint vector $\theta(t)$ or from $\dot{r}_d(t)$ to $\dot{\theta}(t)$ is constructed. Such an approach is usually complicated by the fact that inverse kinematics is a one-to-many mapping, but the method does not require any constraints to be imposed on the articulator configuration, nor does it require any functions to be optimized. It works identically for non redundant and for redundant articulator.

2. Inverse Kinematics Mapping

In order to determine $\theta(t)$ for given $r_d(t)$ through the join velocity vector, $\dot{\theta}(t)$ needs to be computed. One way to determine $\dot{\theta}(t)$ is using Fuzzy Inference System, that deduces the inverse kinematics if the forward kinematics of the problem is known. Variables such as $\dot{r}(t)$, $J(\theta(t)$ and $\dot{\theta}(t)$ are defined in fuzzy value. Inverse kinematics mapping is approached with a simple rule base. The ideology of this rule-base can be illustrated by the following example, using the fuzzy value : pm = "positive medium", nl = "negative large", ns = "negative small", a sample rule for Inverse Kinematics might be formulated as follows.

IF $\dot{r}(t)$ is *pm* *AND* $J(\theta(t)$ is *nl* THEN $\dot{\theta}(t)$ is *ns*

which is intuitively clear: when the link is pointing more or less in the direction of the x-y-z-axis, which is equivalent to saying $J(\theta(t)$ is nl and we try to move the end-effector a small amount in the direction of the respective-axis ($\dot{r}(t)$ is pm), we should decrease $\theta(t)$ a little ($\dot{\theta}(t)$ is ns). This concept can now be explored for different combinations of $\dot{r}(t)$ and $J(\theta(t)$, and the fuzzy rules can be constructed. Furthermore, since the rule-base has been derived using a linearized model, the principle of superposition can be applied and is used to combine the contributions of the rule-base of each joint for the given $\dot{r}(t)$. This is the basis for implementation of the Inverse Kinematics Mapping. The rule-base we actually used for the inverse kinematics mapping is defined over slightly different fuzzy variables.

The coefficient $J_{ij}(\theta(t))$ is the element of the Jacobian matrix in the i-th row and j-th column. In other words, it relates the i-th component of the vector $\dot{r}(t)$ with the j-th joint angle $\theta_j(t)$. The way to obtained $\dot{\theta}(t)$ are combined. Each combination of $\dot{r}(t)$ and $J(\theta(t)$, $\dot{\theta}(t)$ is determined through fuzzy mapping using

Jurusan Teknik Mesin dan Industri FT UGM
ISBN 979-15577-7-1

membership functions. The result of the inverse kinematics mapping can be denoted [Alois Schacherbauer et.al, 1992] as follows.

$$\dot{\theta}_{ij}^{+}(t) = \text{Fuzzy Mapping}(J(\theta(t)), \dot{r}_i(t)) \qquad (4)$$

Since $\dot{\theta}_{ij}^{+}(t)$ yields approximately $\dot{r}_i(t)$, scaling the contributions of each $\dot{\theta}_{ij}^{+}(t)$ to $\dot{r}_i(t)$ by some factor is needed such that the sum of the contributions is close to $\dot{r}_i(t)$. The first scaling is in each row. The effective joint angle change is accordingly given by the following equation [Alois Schacherbauer et.al, 1992].

$$\hat{\dot{\theta}}_{ij} = a_{ij} \cdot \dot{\theta}_{ij}^{+}, \text{ where } a_{ij} = \frac{|J_{ij}|}{\sum_{j=1}^{n} |J_{ij}|} \qquad (5)$$

Consider for one particular joint j, $\hat{\dot{\theta}}_{ij}$ then scaled in each column. The following effective joint angle change is obtained [Alois Schacherbauer et.al, 1992].

$$\dot{\theta}_j = \sum_{i=1}^{n} b_{ij} \cdot \hat{\dot{\theta}}_{ij}, \text{ where } b_{ij} = \frac{|J_{ij}|}{\sum_{i=1}^{m} |J_{ij}|} \qquad (6)$$

Variables b_{ij} are chosen under assumption: starting from a particular $\dot{\theta}_j$ and considering $\dot{\theta}_j + \delta$, the change in $\dot{r}_i(t)$ where $|J_{ij}|$ (i=1...m) is the largest. To keep errors small, the averaged $\dot{\theta}_j$ should be closest to these $\hat{\dot{\theta}}_{ij}$, whose corresponding $|J_{ij}|$ is large. Joint angle $\theta(t)$ and pose $r(t)$ are saved to be used as training data to train ANFIS (Adaptive Neuro-Fuzzy Inference System) network. During training, the ANFIS network learns to map the pose $r(t)$ to the joint angle $\theta(t)$. The same procedure can be conducted to map $\dot{r}(t)$ and $\dot{\theta}(t)$.

An artificial neural network can perform parallel processing and self-training, it can also construct relationship between input and output by learning from sample. The neural network used is a regular error back-propagation. In this paper, as shown in fig. 1, the network contains three layers : the first layer is an input layer, containing three nodes , the second layer is a hidden layer, containing seven nodes and the third layer has only one node, representing the joint angle. The elements of the input layer is denoted by h, hidden layer is denoted by i and j denotes the output layer, the respective input and output are x_{kh} and d_{kj}.

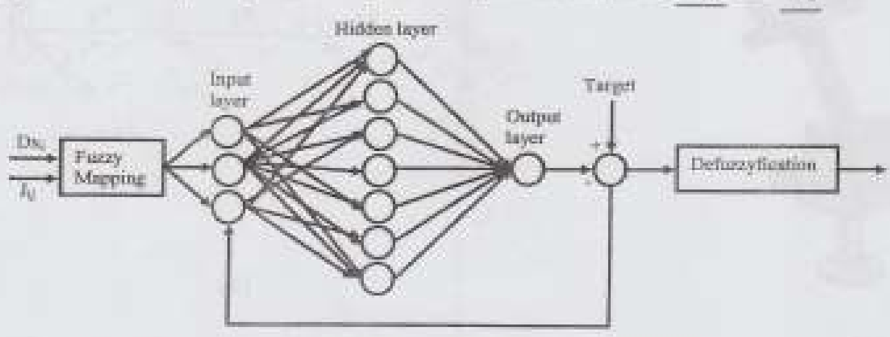


Figure 1. Diagram block of ANFIS

The transfer function for each node in the network is expressed with the sigmoid function. Error function is represented with $E_k = \frac{1}{2}\Sigma(y_j - d_{kj})^2$. The output y_i from each noc : of the hidden layer is [Zhang Jinlong et.al, 2005] :

$$x_i = \sum_{h=1}^{1} w_{hi} \cdot x_{kh} + b_i \text{ and } y_i = \frac{1}{1+\exp(-x_i)}, \quad i = 1,2,\ldots,7 \qquad (7)$$

ISBN 979-15577-7-1

The output y_j from each node of the output layer is :

$$x_j = \sum_{i=1}^{7} w_{ij} \cdot y_i + b_j \quad \text{and} \quad y_j = \frac{1}{1+\exp(-x_j)}, \quad j = 1 \tag{8}$$

Right value and threshold value are modified by the following equations below :

$$\omega_{ij}(t+1) = \omega_{ij}(t) - \eta \frac{\partial E_k}{\partial x_j} y_i + \alpha[\omega_{ij}(t) - \omega_{ij}(t-1)] \tag{9}$$

$$\theta_j(t+1) = \theta_j(t) - \eta \frac{\partial E_k}{\partial x_j} + \alpha[\theta_j(t) - \theta_j(t-1)] \tag{10}$$

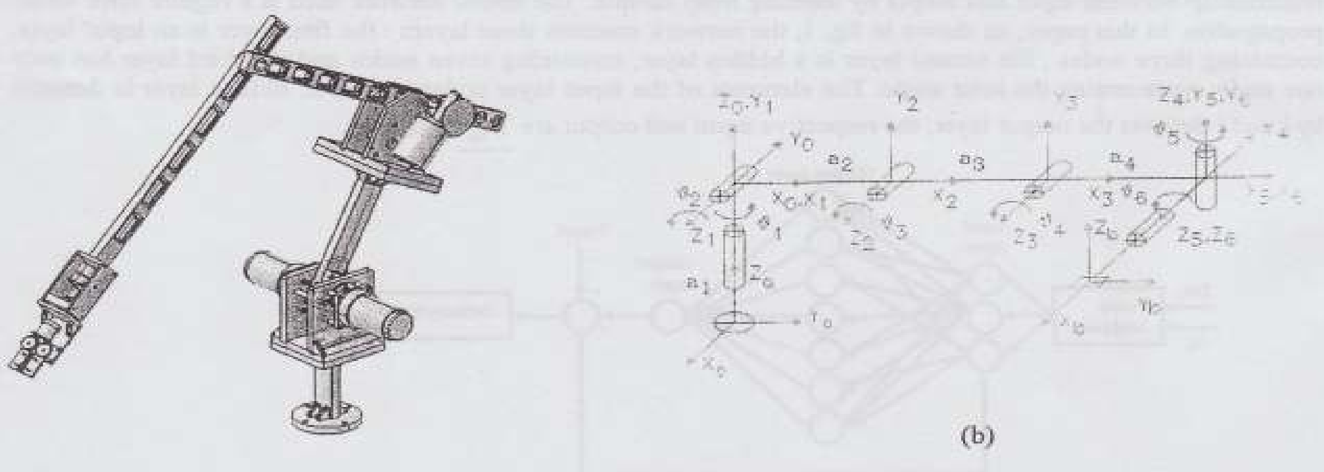
$$\omega_{hi}(t+1) = \omega_{hi}(t) - \eta \frac{\partial E_k}{\partial x_i} y_h + \alpha[\omega_{hi}(t) - \omega_{hi}(t-1)] \tag{11}$$

$$\theta_i(t+1) = \theta_i(t) - \eta \frac{\partial E_k}{\partial x_i} + \alpha[\theta_i(t) - \theta_i(t-1)] \tag{12}$$

Where the momentum term $\alpha \in (0,1)$, η is learning rate and t is the modifying quantity.

3. Coordinate frames for the 6 DoF MoroLIPI Articulator

The 6 DoF MoroLIPI articulator is designed by Research Center for Electrical Power and Mechatronics. (Puslit Telimek), LIPI. The MoroLIPI articulator has six degrees-of freedom (four links with four DoF and a gripper-cutter mechanism with two DoF) as illustrated in fig. 2 (a) and the coordinate system in fig. 2(b). The all joints are revolute. The maximum travel for these joints is 663.55 mm for the lateral (X) axis, 663.55 mm for the vertical (Y) axis, and 663.6795 mm for the longitudinal (Z) axis. All of the joints are actuated by servo motors. The connecting members and supporting structures of the MoroLIPI are made out of aluminium. The articulator's "end-effector" is gripper-cutter mechanism with two DoF.



(a)

Figure 2. The MoroLIPI articulator. (a). Mechanical configuration. (b). Coordinate system

Coordinate frames for 6 DoF MoroLIPI Articulator are assigned as shown fig. 2 (b). They are established using principles of the Denavit-Hartenberg (D-H) convention. The base frame is Xa, Ya, Za which does not move with respect to X0, Y0, Z0. The gripper frame Xb, Yb, Zb does not move with respect to X6, Y6, Z6. The reference frames from X1, Y1, Z1 to X6, Y6, Z6 attached to the links are positioned following the Denavit-Hartenberg convention and so at the end of the links.

4. Simulation and Results

Jurusan Teknik Mesin dan Industri FT UGM
ISBN 979-15577-7-1

Actual workspace in the three-dimensional (3-D) space and its orthographic projections can be seen in fig. 3 which is virtually identical to the desired path to the X-Z, Y-Z and X-Y planes. This workspace is obtained through forward kinematics model. They show the approximate extent of the actual reachable space. Part of the boundary of the reachable space is determined by the joint limits of the arm. Each of them corresponds to a randomly generated joint vector. The ranges of motion for the six joint angle are: $\theta_1 \in [-360^0, 360^0]$, $\theta_2 \in [-53^0, 53^0]$, $\theta_3 \in [-136^0, 170^0]$, $\theta_4 \in [180^0, 180^0]$, $\theta_5 \in [360^0, 360^0]$ and $\theta_6 \in [-45^0, 45^0]$. These figures show the approximate extent of the actual reachable space. The inverse kinematics mapping is started by generating a reference trajectory between initial and final pose, the initial joint vector is $\theta(0) = (0,0,0,0,0,0)^T$ and $\theta(t) = (-90^0, -60^0, -60^0, -45^0, -45^0, -30^0)^T$, in the work volume of articulator as shown in fig. 3 by using cubic trajectory planning. Output of this generated trajectory is position and orientation. They become input for inverse kinematic mapping simulation.

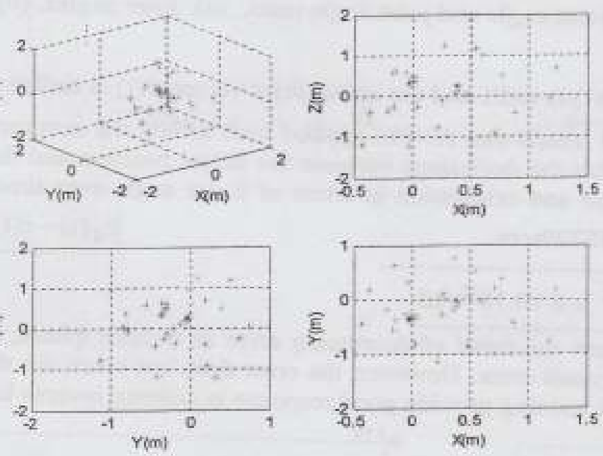


Figure 3. The plots of a reachable task space positions in 3D and its projected

Figures 4(a) and (b) show actual coordinate $r(t) = (x, y, z, x^{\omega}, y^{\omega}, z^{\omega})$ and $\dot{r}(t) = (\dot{x}, \dot{y}, \dot{z}, \dot{x}^{\omega}, \dot{y}^{\omega}, \dot{z}^{\omega})$ where (x,y,z) denotes the Cartesian coordinates of the end-effector in meters, $(x^{\omega}, y^{\omega}, z^{\omega})$ denotes the orientation variables in radians, $(\dot{x}, \dot{y}, \dot{z})$ denotes the translational velocity variables in m/s and $(\dot{x}^{\omega}, \dot{y}^{\omega}, \dot{z}^{\omega})$ denotes the angular velocity variables in rad/s, respectively. The orientation of the end-effector is required to keep unchanged when the articulator moves, $(x^{\omega}, y^{\omega}, z^{\omega})$ are set to zero throughout simulation. It can be seen that actual coordinate for $r(t)$ and $\dot{r}(t)$ can pursue desired coordinate.

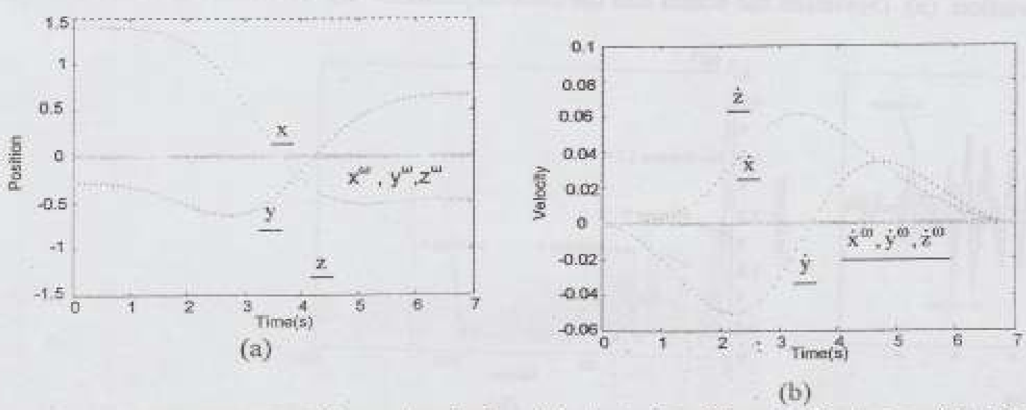


Figure 4. Position and velocity. (a). Actual positions in Cartesian. (b). Linear velocity

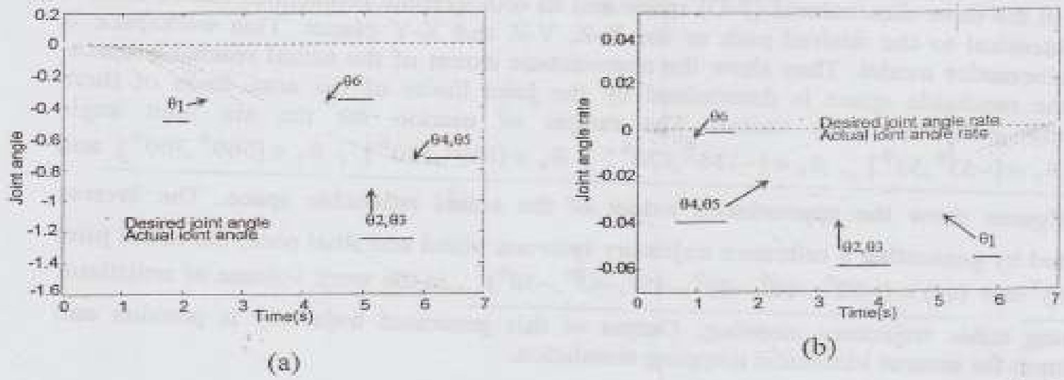


Figure 5. Desired actual joint angle and joint angle rates. (a). Joint angles. (b). Joint angle rates.

The actual joint velocity vector $\dot{\theta}(t)$ in rad/s and the actual joint vector $\underline{\theta(t)}$ in radian generated from the ANFIS are shown in fig. 5(a) and (b). It means that ANFIS applied here satisfy the accuracy in position and velocity. Figure 6(a), 7(a) and 8(b) show that the deviations between the actual position and the desired one, between the actual velocity and the desired one and orientation in terms of Euler angle over time are very small, precisely, max $\|r_d(t) - r(t)\| = 0.38220\text{mm}$, $\|\dot{r}_d(t) - \dot{r}(t)\| = 5.3.10^{-2}\text{ mm}$,

$$\max \sqrt{(X_{err}^{\omega})^2 + (Y_{err}^{\omega})^2 + (Z_{err}^{\omega})^2} = 6.93.10^{-2}\text{ rad}$$

Figure 6(b) and 7(b) show the trend of decreasing error in training phase, both position and velocity. They converge towards the goal small error. However, the error does not reach the desired cut-off. Despite this, the weight trained as result of this training provide good response in solving inverse kinematics problem.

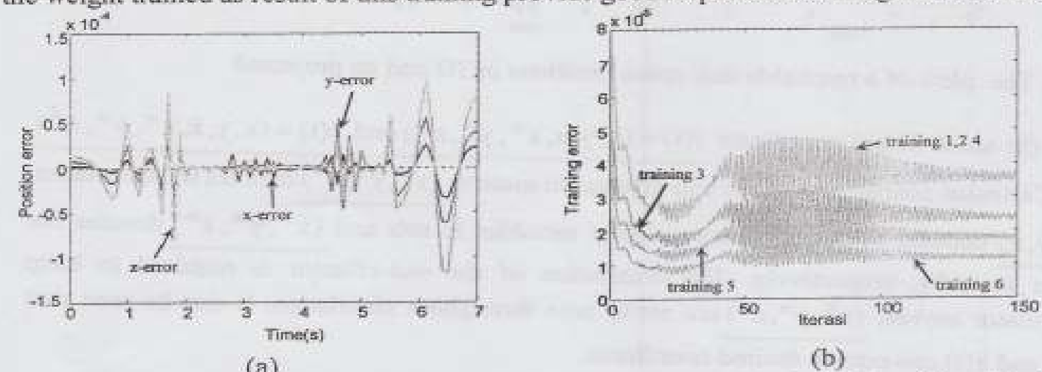


Figure 6. Occurred deviation. (a). Deviation the actual and the desired position. b). Error during training.

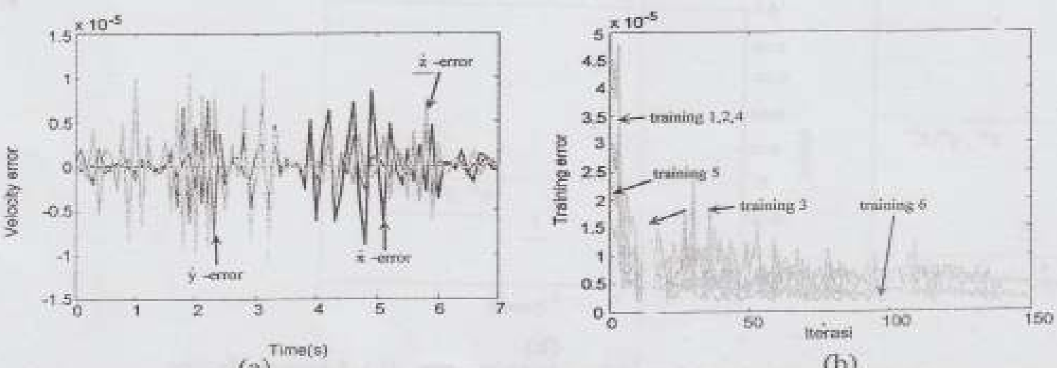
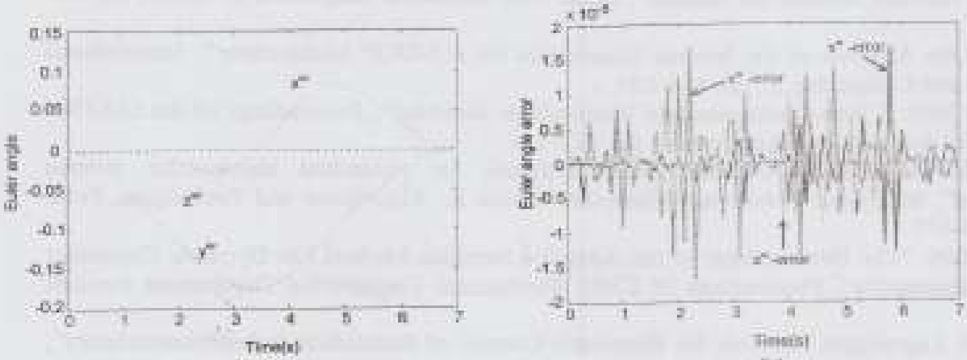


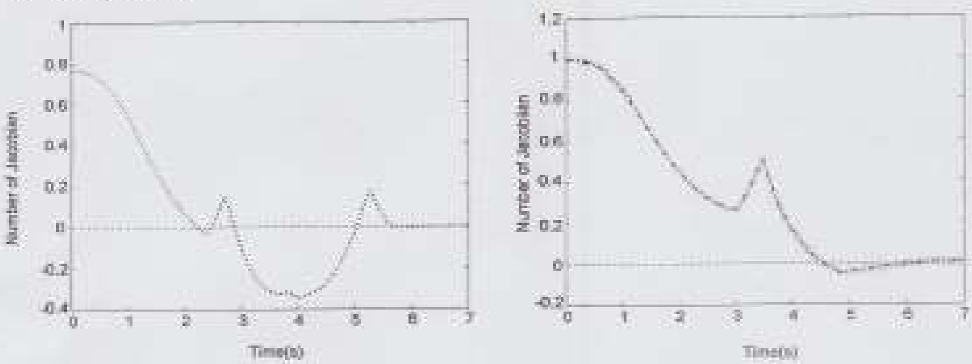
Figure 7. Occurred deviation. (a). Deviation the actual and the desired velocity. (b). Error during training.



(a) (b)

Figure 8. Orientation in terms of Euler angle. (a). Orientation between actual and desired directions (b). Orientation errors in terms of Euler angle.

Since the singularity of articulator is defined as the rank deficiency of the Jacobian, the condition number of the Jacobian is a good indicator of singularity. Figure 9(a) shows the transient behavior of the condition number of the Jacobian in non-singularity condition . there will be no lock, since all the joint angles are not zero. Fig. 9 (b) is in singularity condition, where any of θ_2 and θ_3 is zero, then two columns in the Jacobian are equal, DoF is reduced from six to five and the Jacobian becomes ill-conditioned (rank deficient). There will be lock in elbow or shoulder of articulator (the rotation axes is equal to zero). As shown fig. 9(b), the condition number of the jacobian is being changed near zero. The norm of joint velocity is changed quickly from acceleration to deceleration or vice versa. Physically, the articulator moves out of the singular condition, either from elbow lock or shoulder lock.



(a) (b)

Figure 9. The condition number of the Jacobian. (a). Non-singularity condition. (b). Singularity condition.

5. Conclusions

An Inverse Kinematics Mapping based on Adaptive Neuro-Fuzzy Inference System has been developed. This method does not require any constraints to be imposed on the articulator configuration, nor does it require any functions to be optimized. It works identically for non-redundant and for redundant articulator. The solution is obtained in closed-form, as opposed to numerical solutions of popular schemes. The usability of the method has been shown by employing it on a six-DoF articulator in the six-dimensional Cartesian space.

6. Acknowledgement

The author would like thanks you to Mr. M. Fathul Hikmawan in providing the technical data a d drawings of MoroLIPI

References

[1] Alois Schacherbauer and Yangsheng Xu,1992," Fuzzy Control and Fuzzy Kinematic Mapping for a Redundant Space Robot", The Robotics Institute, Carnegie Mellon University.

[2] A.S. Morris and A. Mansor, 1997," Finding the inverse kinematics of manipulator arm using artificial neural network with lookup table ", Cambridge Journals, pp. 617-625

[3] Corke, P. I., 1996. "A robotics toolbox for matlab.". *IEEE Rob. and Auto. Magazine*, 3, March, pp. 24–32.

[4] D. Xu. Et,al., 2005, " An Analysis of the Inverse Kinematics for a 5-DOF Manipulator", International Journal of Automation and Computing 2 , pp. 114-124.

[5] E. M. Schwartz et.al., 2003, " Non-Commensurate Manipulator Jacobian", Proceedings of the IASTED International Conference Robotics and Application, pp. 112-115.

[6] Hou,Edwin ;Utama,Wibawa,1992, " Artificial neural network for redundant manipulator inverse kinematics computation", Intelligent Robots and Computer Vision X: Algorithms and Techniques, Proc. SPIE Vol. 1607, p. 668-677.

[7] Jason C. Geist et. al, 2006, "The Development of An Adaptive Jacobian Method For Dynamic Constraint Handling in Inverse Kinematics", Proceedings of CMU Mechanical Engineering Department Bennett Technical Conference.

[8] J. Wang et.al, 1999, "A Lagrangian Network for Kinematic Control of Redundant Robot Manipulators", IEEE Trans. Neural Networks, vol. 10, pp. 1123–1132.

[9] K.N. Tarchanidis and J.N. Lygouras, 2000, " A Hybrid Neural Network/Newton-Raphson Algorithm Solving the Inverse Kinematics Problem",International Association of Science and Technology for Development.

[10] Zhang Jinlong et.al, 2005, "Ultra Positioning Control Technique Based on Neural Network", Journal of Southeast University: English Edition, pp. 16-19.