Discrete-Time kinematic Control Based on MIFRENs for a Redundant Robot Manipulator of 7-DOF.

E. Torres, H. Gutiérrez, C. Treesatayapun, V. Parra
Department of Robotic and Advanced Manufacturing
CINVESTAV-IPN, Unidad Saltillo
Ramos Arizpe, 25900, México
omarelio.torres@cinvestav.edu.mx, gufohu@gmail.com,
chidentree@cinvestav.edu.mx, vicente.parra@cinvestav.edu.mx
Teléfono: (844)-4389600 ext. 8600

***Resume*— This article describes an adaptive discrete-time kinematic controller based on velocity mode for a 7-DOF robot manipulator using a class of adjustable networks called Muplti-Input Fuzzy Rules Emulated Networks(MIFRENs). The proposed adaptation algorithm guarantees closed loop performance and stabilization analysis. The experiment set-up with Mitsubish PA-10 performs the controller validation. According to the comparison results with a proportional controller, the superior performance and tracking trajectory can be obtained by the adaptive controller.**

Keywords: redundant manipulators, adaptive control, neural control, discrete-time.

I. INTRODUCTION

Since its appearance, the robot manipulators have been studied in almost all aspects, such as modeling, control and programming; Many nonlinear controllers that have been applied to industrial robot manipulators need the exact knowledge of the robot's dynamic model and its parameters (Yang & Huang, 1990). Unfortunately, nonlinearities such as friction, backlash and death zones are difficult to model. Moreover, its parameters may not be accurately specified due to measurement errors (Wang & C.S.G, 2003). These factors make the controller design complicated and not robust; The combination of fuzzy logic systems and neural networks known as neuro-fuzzy systems (Li, 1997), (Wai, 2010) have been utilized as intelligent controllers to cope with such nonlinearities problems. These intelligent controllers have the advantage of human knowledge and reasoning processes of fuzzy logic with the learning ability of neural networks without the need of the mathematical model of the plant. However, most of these systems have complicated structures, high computational complexity and compatible design in continues-time domain. Neuro-fuzzy systems called Multi-Input Fuzzy Rules Emulated Networks (MIFRENs) (Treesatayapun and Uatrongjit, 2005) have the advantage of low computational complexity due to its simple structure. This work presents a discrete-time controller based on MIFRENs with some modifications suited for robotic systems; For the experimental results, MIFRENs controller is compared with a proportional controller using a Mitsubishi Heavy Industries PA-10 robot in velocity mode. The MIFRENs performance is presented for regulation and tracking trajectory. The rest of the paper is organized as follows: section II describes the redundacy robot problem for redundant robots, afterwards, in III section describes the general structure of MIFRENs, the parameter adaptation and the stability analysis. Section IV, shows a brief description of the robot manipulator PA-10 from Heavy Industries, the controller design and experimental results. Finally, section V describes the conclusions, in the appendix A describes the analytical inverse kinematics to be used in this article.

II. REDUNDANCY ROBOT PROBLEM

For a robot of 6-DOF, only 3-DOF for position and 3-DOF for orientation is necessary to perform 3D space tasks. The direct kinematic is obtained with the following formula

$$\boldsymbol{x}_7 = f(\boldsymbol{q}) \in \mathbb{R}^6 \tag{1}$$

being $\boldsymbol{x}_7 \in \mathbb{R}^3 = \begin{bmatrix} \mathrm{o}x & \mathrm{o}y & \mathrm{o}z \end{bmatrix}^T$ tip position of the end-effector of the robot manipulator. The direct kinematic of velocity is

$$\dot{\boldsymbol{x}_7} = J(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{2}$$

where $J(\boldsymbol{q}) = \frac{\partial f(\boldsymbol{q})}{\partial \boldsymbol{q}}$ is the jacobian of the robot.

For the robot PA-10, a redudant robot manipulator of 7-DOF, the inverse kinematic of velocity is given by

$$\dot{\boldsymbol{q}} = J(\boldsymbol{q})^+ + [I - J(\boldsymbol{q})^+ J(\boldsymbol{q})]\boldsymbol{z} \in \mathbb{R}^m \tag{3}$$

where $J(\boldsymbol{q})^+ = J(\boldsymbol{q})^T [J(\boldsymbol{q}) J(\boldsymbol{q})^T]^{-1}$ is the pseudo-inverse from the left. For redundant manipulators, there are more degrees of freedom (DOF) than the ones needed to perform their end-effector positioning tasks. In the equation (3); part $[I - J(\boldsymbol{q})^+ J(\boldsymbol{q})]\boldsymbol{z}$ is due to redundacy of the robot manipulator. Inverse kinematic has been solved by several methods, (Ding & Tso, 1998), (Assal & Watanabe, 2006), (Zhang & Wang, 2003); For these article, the methodology in (Shimizu and Kakuya, 2008) is utilized and such inverse kinematics is described in the appendix.

III. MIFRENs ADAPTIVE CONTROLLER

A. MIFRENs Structure

The network structure of MiFREN can be divided into 5 layers shown in figure (1).

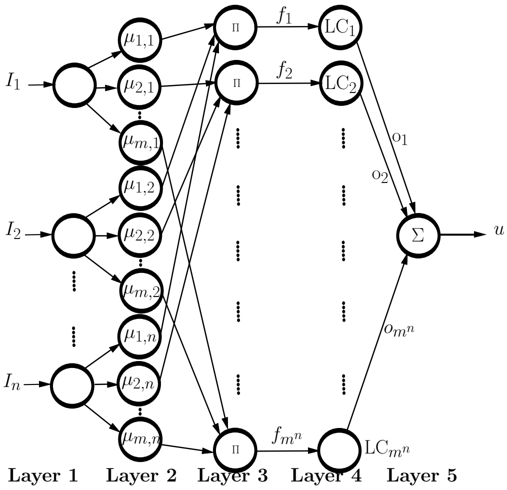

Figure 1. MIFRENs structure

Layer 1: In this layer, we acquire the input to pass it to the next layer.

Layer 2: This layer can be considered as the fuzzification stage. Here we have m^n nodes corresponding to the number of rules. Each node in this layer contains a membership function that belongs to the linguistic level (fuzzy set) j of the input i . The output of each node is the membership grade (value between 0 and 1) of the respective input and it specifies the degree as the input satisfies the linguistic level associated with the node. This output is denoted by $\mu_{j,i}$ with $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$. We use Gaussian functions as input membership functions defined by the following expression,

$$Mf_i = \exp\left(-\frac{1}{2}\frac{(e - X_{0_i})^2}{(\sigma_i)^2}\right), \qquad (4)$$

where e is the input variable, X_{0_i} is the center position and σ_i is the standard deviation. For the membership functions in the extremes(Positive Big and Negative Big), we use the following equation

$$Mfe_{p,n} = \frac{1}{exp(-\xi_{p,n}(e - \rho_{p,n})) + 1}, \qquad (5)$$

where $\xi_{p,n}$ is used to modify the slope of the curve and $\rho_{p,n}$ is the point over the horizontal axis corresponding to the value 0.5 of the function. The parameters in this layer are referred as nonlinear parameters.

Layer 3: This layer corresponds to the fuzzy inference. The number of nodes in this layer is m^n corresponding with the number of rules. The output signal at each node gives the firing strength of each rule and it can be calculated by

$$f_k = \prod_{i=1}^{n} \mu_{k,i}, \qquad (6)$$

where $k = 1, 2, ..., m^n$.

Layer 4: This layer can be considered as the defuzzification stage. Here, there are m^n nodes wich contain a linear function called linear consequence (Lc). The node takes the corresponding firing strength value of the rule and gives the corresponding consequent value. The output of this layer is given by

$$O_k = \alpha_k \mu_{f_k} + \beta_k, \qquad (7)$$

where α_k is the slope of the line and β_k is the bias. These parameters are referred as consequent parameters or linear parameters.

Layer 5: The single node in this layer computes the overall output as the summation of all incoming signals. The output of this layer is applied directly to the plant. It is obtained by

$$U = \sum_{k=1}^{m^n} O_k. \qquad (8)$$

B. MIFRENs controller and parameters adaptation

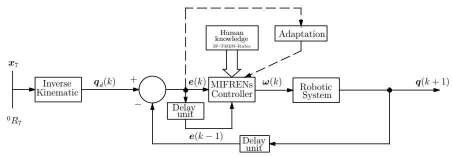

Figure 2. scheme of the MIFRENs controller using analytical inverse kinematics

Let consider the controlled system illustrated in Figure 2, these system represents the scheme of the MIFRENs controller using the analytical inverse kinematic, where $\boldsymbol{q}_d(k) \in \mathbb{R}^7$ represents the analytical inverse kinematic equations (27)-(33), $\boldsymbol{e}(k) \in \mathbb{R}^7$ represents the error of each joint, $\boldsymbol{e}(k-1) \in \mathbb{R}^7$ represent the error in past time $k-1$ of each joint. Generally, define the measured position $\boldsymbol{q}(k) \in \mathbb{R}^7$ as the plant output y_k and velocity command $\boldsymbol{w}(k) \in \mathbb{R}^7$ as the control effort $u(k)$. MIFRENs uses backpropagation learning rule which is in essence the simple steepest decent method to update the node's parameters of the adaptive network (Jang and Sun, 1997). Firstly, consider the cost function of squared error as

$$V(k) = \frac{1}{2}e^2(k) = \frac{1}{2}[y_d(k) - y(k)]^2, \qquad (9)$$

where e is the error of the system, y_d is the target position and y is the current position of the robot in the time index k.

In each interaction we update the parameters using the gradient descent method according with the next equation

$$P_i(k+1) = P_i(k) - \eta \frac{\partial V(k)}{\partial P_i(k)}, \tag{10}$$

where P_i is whichever parameter of input membership functions or linear consequences and η is the learning rate, which indicates the step size in direction of the vector $\frac{\partial V(k)}{\partial P_i(k)}$. In the developed algorithm for this work, we use two different learning rates, one for the premise parameters and another for the consequent parameters. By applying the chain rule to (9), we have

$$\frac{\partial V(k)}{\partial P_i(k)} = \frac{\partial V(k)}{\partial e(k)} \frac{\partial e(k)}{\partial y(k)} \frac{\partial y(k)}{\partial U_{real}(k)} \cdots$$

$$\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)}, \tag{11}$$

where $\frac{\partial V(k)}{\partial e(k)} = e(k)$ and $\frac{\partial e(k)}{\partial y(k)} = -1$ the term $\frac{\partial y(k)}{\partial U_{real}(k)}$ is called plant information and for simplicity is denoted by Yp. It can be approximated by

$$Y_p \simeq \frac{y(k) - y(k-1)}{U_{real}(k) - U_{real}(k-1)}. \tag{12}$$

Finally, the equation (10) is transformed into

$$P_i(k+1) = P_i(k) + \eta e(k) Y_p \frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)}. \tag{13}$$

C. Closed Loop Stability

The use of the gradient descent technique for optimizing the cost function has an effect caused by the learning rate over the system stability. Too large values of learning rate can cause instability and too small values reduce the learning performance and provoke slow convergence. To overcome such drawback, we need to consider the stability according to Lyapunov method.

Consider the following Lyapunov function

$$V(k) = \frac{1}{2} e^2(k) = \frac{1}{2} [y_d(k) - y(k)]^2. \tag{14}$$

The change of this function is

$$\Delta V(k) = \frac{1}{2} [e^2(k+1) - e^2(k)], \tag{15}$$

where $e(k+1) = e(k) + \Delta e(k)$. Replacing this in (15), we obtain

$$\begin{aligned}
\Delta V(k) &= \tfrac{1}{2}[(e(k) + \Delta e(k))^2 - e^2(k)], \\
&= \tfrac{1}{2}[e^2(k) + 2e(k)\Delta e(k) + \Delta e^2(k) - e^2(k)], \\
&= \tfrac{1}{2}[2e(k)\Delta e(k) + \Delta e^2(k)], \\
&= e(k)\Delta e(k) + \tfrac{1}{2}\Delta e^2(k), \\
&= \Delta e(k)[e(k) + \tfrac{1}{2}\Delta e(k)].
\end{aligned}$$

The change of the error $\Delta e(k)$ can be approximated by

$$\Delta e(k) = \frac{\Delta e(k)}{\Delta P_i(k)} \Delta P_i(k) \simeq \frac{\partial e(k)}{\partial P_i(k)} \Delta P_i(k). \tag{16}$$

We have from (13)

$$\Delta P_i(k) = \eta e(k) Y_p \frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \tag{17}$$

and

$$\frac{\partial e(k)}{\partial P_i(k)} = \frac{\partial e(k)}{\partial y(k)} \frac{\partial y(k)}{\partial U_{real}} \frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)}. \tag{18}$$

According to section III-B, $\frac{\partial e(k)}{\partial y(k)} = -1$ and $\frac{\partial y(k)}{\partial U(k)} = Y_p$. Changing this in (18), we obtain

$$\frac{\partial e(k)}{\partial P_i(k)} = -Y_p \frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)}. \tag{19}$$

Replacing (17) and (19) into (16) we have

$$\Delta e(k) = -\eta_i e(k) Y_p^2 \left(\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \right)^2. \tag{20}$$

Therefore, replacing this last equation into (15), the change of Lyapunov function can be calculated by

$$\Delta V(k) = -\eta_i e(k) Y_p^2 \left(\frac{\partial U(k)}{\partial P_i(k)} \right)^2 \cdots$$

$$\left[e(k) - \frac{1}{2} \eta_i e(k) Y_p^2 \left(\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \right)^2 \right]. \tag{21}$$

The stability according Lyapunov says that this last function must be negative defined (Khalil, 2002), then multiplying the terms we have

$$\Delta V(k) = -\eta_i e^2(k) Y_p^2 \left(\frac{\partial U(k)}{\partial P_i(k)} \right)^2$$

$$+\frac{1}{2} \eta_i^2 e^2(k) Y_p^4 \left(\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \right)^4, \tag{22}$$

where the second term will be always positive. Therefore, η_i needs to be greater than 0 and

$$\frac{1}{2} \eta_i^2 e^2(k) Y_p^4 \left(\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \right)^4$$

$$< \eta_i e^2(k) Y_p^2 \left(\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \right)^2. \tag{23}$$

Cancelling terms and clearing η_i

$$\eta_i < \frac{2}{Y_p^2 \left(\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)} \right)^2}. \tag{24}$$

Finally, we have that

$$0 \le \eta_i < \frac{2}{Y_p^2 (\frac{\partial U_{real}(k)}{\partial U_{MIFRENs}(k)} \frac{\partial U_{MIFRENs}(k)}{\partial P_i(k)})^2}. \tag{25}$$

The learning rate η_i must be in this range to assure stability of the system; To obtain the learning rate for linear and nonlinear parameters respectively, we compute the upper bounds to every parameter at each sampling time and we take the minimum multipling by two constansts γ_l and γ_{nl}, with values between $0 < \gamma_{l,nl} < 1$.

IV. Experimental Setup and Results

The Mitsubishi PA-10 robot is an industrial manipulator comprised by trifasic brushless CD servomotors (Campa & Torres, 2005) supported by harmonic drives in each joint. The acronym means *"Portable General-Purpose Intelligent Arm"* and the system is established in four levels, (see Figure 3) which forms a hierarchical structure.

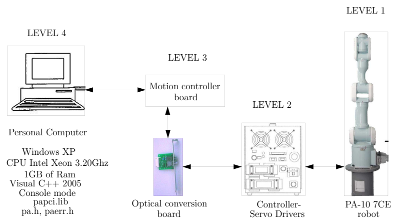

Figure 3. Experimental set-up

- Level 4: Control operation section; comprised by the personal computer
- Level 3: Motion control section; comprised by the motion controller board PCI MHI-D7281 from Mitsubishi Heavy Industries and the optical conversion board MHI-D7210, the transmision of data to the servo drivers is established using a communication protocol ARCNET by fiber optic cable
- Level 2: Servo actuators
- Level 1: Robot manipulator

The personal computer(Level 4) uses windows xp as operative system, with two processors CPU Intel Xeon 3.20Ghz, 1Gb of Ram. The c/c++ programs run in visual c++ 2005 console mode, using papci.lib as the PA-10 robot manipulator library and pa.h, paerr.h headers necessary to manipulate the robot.

A. Design of the MIFRENs Controller

As controller, MIFRENs is a discrete time domain control, which can emulate fuzzy rules with n inputs in the familiar (If...Then) format(see figure 1) and adapt the parameters of its fuzzy sets in each sampling time. The number of fuzzy sets inference linguistic levels is denoted by m; therefore, the number of rules will be m^n. The general structure of a rule of MIFRENs according with the structure presented in section III is

If *I_1 is $\mu_{m,1}$ and I_2 is $\mu_{m,2}$ ... and I_n is $\mu_{m,n}$* **Then** *O_{m^n} is Lc_{m^n}.*

For this design, the inputs of the controller for every link at the time index k will be the error defined like $e_i(k) = q_{d_i}(k) - q_i(k)$ and the previous error $e_i(k-1) = q_{d_i}(k-1) - q_i(k-1)$, where q_i is the angular position of the link and q_{d_i} is the desired position.

Fuzzification (second layer): The fuzzy sets for both inputs were chosen as ζ_{e_i}={NE, ZE, PE}, where the label NE, ZE, and PE stand for the linguistic levels "Negative", "Zero" and "Positive" respectively, for the $e_i(k)$ input. ζ_{ep_i}={NEP, ZEP, PEP}, where the label NEP, ZEP, and PEP stand for "Negative", "Zero" and "Positive" respectively, for the $e_i(k-1)$ input.

Rule base (third layer): As we have two inputs and each one has three linguistic levels, thus we have nine rules, which are shown next

Rule 1: If $e_i(k)$ is **PE** and $e_i(k-1)$ is **PEP** Then $O_1 = \alpha_1 f_1$
Rule 2: If $e_i(k)$ is **PE** and $e_i(k-1)$ is **ZEP** Then $O_2 = \alpha_2 f_2$
Rule 3: If $e_i(k)$ is **PE** and $e_i(k-1)$ is **NEP** Then $O_3 = \alpha_3 f_3$
Rule 4: If $e_i(k)$ is **ZE** and $e_i(k-1)$ is **PEP** Then $O_4 = \alpha_4 f_4$
Rule 5: If $e_i(k)$ is **ZE** and $e_i(k-1)$ is **ZEP** Then $O_5 = \alpha_5 f_5$
Rule 6: If $e_i(k)$ is **ZE** and $e_i(k-1)$ is **NEP** Then $O_6 = \alpha_6 f_6$
Rule 7: If $e_i(k)$ is **NE** and $e_i(k-1)$ is **PEP** Then $O_7 = \alpha_7 f_7$
Rule 8: If $e_i(k)$ is **NE** and $e_i(k-1)$ is **ZEP** Then $O_8 = \alpha_8 f_8$
Rule 9: If $e_i(k)$ is **NE** and $e_i(k-1)$ is **NEP** Then $O_9 = \alpha_9 f_9$

Inference engine (third layer): Here, we use the and operation to do the aggregation between inputs. The output of the third layer of the MIFRENs network is calculated using the T-norm: product. With this, we obtain the firing strength(f_k) of each rule.

Defuzzification (fourth layer): To make the defuzzification step, we have nine linear consequences since we have nine rules. The fuzzy set corresponding is ζ_{O_k}={PBO, PMO, PSO, PNZO, PZO, NNZO, NSO, NMO, NBO}, where the labels, from left to right, mean "Positive Big", "Positive Medium", "Positive Small", "Nearly Zero Positive", "Zero", "Nearly Zero Negative", "Negative Small", "Negative Medium" and "Negative Big". In this case, we choose for each line a bias $\beta_i = 0$. Therefore, only the slope α_i is going to be adapted and we can save computation time.

B. Results

In order to evaluate the system performance of MIFRENs controller, a proportional controller compares with the MIFRENs controller, (see figures 4, 5, 6); In both controllers α=180 degrees, θ=180 degrees, and ϕ=90.

V. Conclusion

The direct adaptive controller has been introduced for 7-DOF robotic system based on velocity mode for class of discrete time systems. The dynamic model of the robotic system is not necessary in order to apply this controller. The working cartesian coordinate including pitch, yaw and roll has been transformed to angular positions by using inverse kinematic. In this application, we can assign the desired position of trajectory with in cartesian coordinates directly. The closed loop system stability has been assured by the proposed adaptive algorithm. The suitable learning rate has been determined by assistance information related on the robotic systems. The experimental results with Mitusibishi PA-10 have represented the superior performance and tracking ability in comparison with the proportional controller.

VI. Acknowledgment

The authors would like to thank CONACYT(Project # 84791) for the financial support through this work.

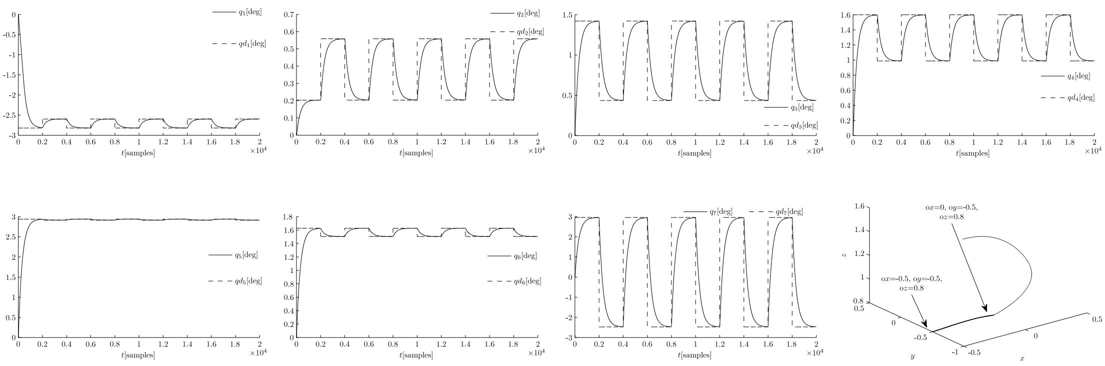

Figure 4. Tracking position regulation of each joint using a proportional controller; Last figure- tracking trayectory by the end-effector in two points.

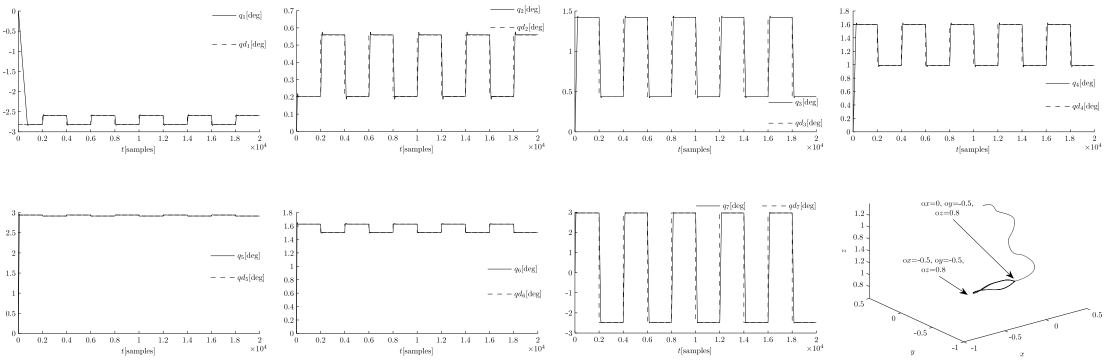

Figure 5. Tracking position regulation of each joint using MIFRENs controller; Last figure- tracking trayectory by the end-effector in two points.

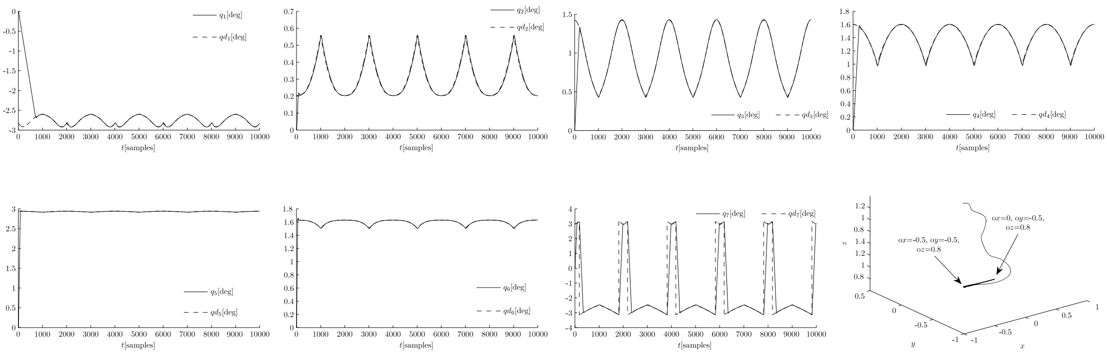

Figure 6. Tracking position regulation of each joint using MIFRENs controller; Last figure- tracking trayectory of a line by the end-effector.

APPENDIX

A. Analytical Inverse kinematics

Extented and deep information regarding the analytical inverse kinematic of a 7-DOF can be found in (Shimizu and Kakuya, 2008). The arm angle ψ is equivalent to the redundancy and it can arbitrarily be chosen for any pose of the manipulator's tip. Let $\boldsymbol{x}_7 \in \mathbb{R}^3 = \begin{bmatrix} \mathrm{o}x & \mathrm{o}y & \mathrm{o}z \end{bmatrix}^T$ and ${}^0R_7 \in \mathbb{R}^{3\times 3}$ the tip position and orientation, respectively, and d_{bs} =0.317mts, d_{se} =0.45mts, d_{ew} =0.48mts , d_{wt} =0.07mts the link lengths of the Mitsubishi robot manipulator PA-10; The subscripts bs, se, ew, wt mean base-shoulder (bs), shoulder-elbow (se), elbow-wrist(ew) and wrist-tip (wt).

The orientation matrix is represented by

$$ {}^0R_7 = \begin{bmatrix} \mathrm{c}_\alpha \mathrm{c}_\theta & -\mathrm{s}_\alpha \mathrm{c}_\phi + \mathrm{c}_\alpha \mathrm{s}_\theta \mathrm{s}_\phi & \mathrm{s}_\alpha \mathrm{s}_\phi + \mathrm{c}_\alpha \mathrm{s}_\theta \mathrm{c}_\phi \\ \mathrm{s}_\alpha \mathrm{c}_\theta & \mathrm{c}_\alpha \mathrm{c}_\phi + \mathrm{s}_\alpha \mathrm{s}_\theta \mathrm{s}_\phi & -\mathrm{c}_\alpha \mathrm{s}_\phi + \mathrm{s}_\alpha \mathrm{s}_\theta \mathrm{c}_\phi \\ -\mathrm{s}_\theta & \mathrm{c}_\theta \mathrm{s}_\phi & \mathrm{c}_\theta \mathrm{c}_\phi \end{bmatrix} \quad (26) $$

being $\mathrm{c}_\alpha = \cos\alpha$, $\mathrm{c}_\theta = \cos\theta$, $\mathrm{c}_\phi = \cos\phi$, $\mathrm{s}_\alpha = \sin\alpha$, $\mathrm{s}_\theta = \sin\theta$, $\mathrm{s}_\phi = \sin\phi$. The symbols α, θ, and ϕ represents yaw, pitch and roll of the end-effector. The analytical inverse kinematic of a 7-DOF robot which involves the arm angle ψ is denoted by

$$o_1 = -a_{s22}\sin(\psi) - b_{s22}\cos(\psi) - c_{s22}$$
$$o_2 = -a_{s12}\sin(\psi) - b_{s12}\cos(\psi) - c_{s12}$$
$$qd_1 = \arctan2(o_1, o_2) \quad (27)$$
$$qd_2 = \arccos(-a_{s32}\sin(\psi) - b_{s32}\cos(\psi) - c_{s32}) \quad (28)$$
$$o_3 = a_{s33}\sin(\psi) - b_{s33}\cos(\psi) - c_{s33}$$
$$o_4 = -a_{s31}\sin(\psi) - b_{s31}\cos(\psi) - c_{s31}$$
$$qd_3 = \arctan2(o_3, o_4) \quad (29)$$
$$qd_4 = \arccos\left(\frac{\|\boldsymbol{x}_{sw}\|^2 - d_{se}^2 - d_{ew}^2}{2d_{se}d_{ew}}\right) \quad (30)$$
$$o_5 = a_{w23}\sin(\psi) + b_{w23}\cos(\psi) + c_{w23}$$
$$o_6 = a_{w13}\sin(\psi) + b_{w13}\cos(\psi) + c_{w13}$$
$$qd_5 = \arctan2(o_5, o_6) \quad (31)$$
$$qd_6 = \arccos(a_{w33}\sin(\psi) + b_{w33}\cos(\psi) + c_{w33}) \quad (32)$$
$$o_7 = a_{w32}\sin(\psi) + b_{w32}\cos(\psi) + c_{w32}$$
$$o_8 = -a_{w31}\sin(\psi) - b_{w31}\cos(\psi) - c_{w31}$$
$$qd_7 = \arctan2(o_7, o_8) \quad (33)$$

where (27)-(33) are the articulated variables of a 7-DOF robot denoted in [rad].

a_{sij}, b_{sij}, and c_{sij} are the (i,j) elements of the matrices $A_s \in \mathbb{R}^{3\times3}$, $B_s \in \mathbb{R}^{3\times3}$, $C_s \in \mathbb{R}^{3\times3}$ and are constant matrices denoted $A_s = U_{swX}\ {}^0R_3$, $B_s = -U_{swX}^2\ {}^0R_3$, $C_s = \boldsymbol{u}_{sw}\ \boldsymbol{u}_{sw}^T\ {}^0R_3$.
The rotation matrix ${}^0R_3$ for the inverse kinematic solution is given by

$$ {}^0R_3 = \begin{bmatrix} \mathrm{c}_1\mathrm{c}_2 & -\mathrm{c}_1\mathrm{s}_2 & -\mathrm{s}_1 \\ \mathrm{s}_1\mathrm{c}_2 & -\mathrm{s}_1\mathrm{s}_2 & \mathrm{c}_1 \\ -\mathrm{s}_2 & -\mathrm{c}_2 & 0 \end{bmatrix} \quad (34)$$

being $\mathrm{s}_1 = \sin(p_1)$, $\mathrm{s}_2 = \sin(p_2)$, $\mathrm{c}_1 = \cos(p_1)$, $\mathrm{c}_2 = \cos(p_2)$.

$$p_1 = \arctan2(x_{bw2}, x_{bw1}) \quad (35)$$
$$p_4 = \arccos\left(\frac{\|\boldsymbol{x}_{sw}\|^2 - d_{se}^2 - d_{ew}^2}{2d_{se}d_{ew}}\right) \quad (36)$$
$$o_9 = x_{bw3} - d_{bs}$$
$$o_{10} = \sqrt{x_{bw1}^2 + x_{bw2}^2}$$
$$o_{11} = d_{ew}\sin(p_4)$$
$$o_{12} = d_{se} + d_{ew}\cos(p4)$$
$$p_2 = -\arctan2(o_9, o_{10}) - \arctan2(o_{11}, o_{12}) + \frac{\pi}{2} \quad (37)$$

where x_{bwi} represents the i element of the vector from base to the wrist $\boldsymbol{x}_{bw}$ and is represented by $\boldsymbol{x}_{bw} = \boldsymbol{x}_7 -^0 R_7\boldsymbol{l}_{wt}$
The kew-symmetric matrix of vector U_{swX} is denoted by

$$U_{swX} = \begin{bmatrix} 0 & -u_{sw3} & u_{sw2} \\ u_{sw3} & 0 & -u_{sw1} \\ -u_{sw2} & u_{sw1} & 0 \end{bmatrix} \quad (38)$$

where the unit vector u_{swi} represents the i element of the vector $\boldsymbol{u}_{sw}$, $\boldsymbol{u}_{sw} = \frac{\boldsymbol{x}_{sw}}{\|\boldsymbol{x}_{sw}\|}$, $\boldsymbol{x}_{sw} = \boldsymbol{x}_7 - \boldsymbol{l}_{bs} -^0 R_7\boldsymbol{l}_{wt}$.
$\boldsymbol{x}_{sw} \in \mathbb{R}^3$, $\boldsymbol{l}_{bs} = \begin{bmatrix}0 & 0 & d_{bs}\end{bmatrix}^T$, $\boldsymbol{l}_{wt} = \begin{bmatrix}0 & 0 & d_{wt}\end{bmatrix}^T$.
a_{wij}, b_{wij}, and c_{wij} are the (i,j) elements of the matrices $A_w \in \mathbb{R}^{3\times3}$, $B_w \in \mathbb{R}^{3\times3}$, $C_w \in \mathbb{R}^{3\times3}$ and are constant matrices denoted $A_w =^3 R_4^T\ A_s^T\ {}^0R_7$, $B_w =^3 R_4^T\ B_s^T\ {}^0R_7$, $C_w =^3 R_4^T\ C_s^T\ {}^0R_7$.
where the rotation matrix ${}^3R_4$ is represented by

$$ {}^3R_4 = \begin{bmatrix} \cos(qd_4) & 0 & \sin(qd_4) \\ \sin(qd_4) & 0 & -\cos(qd_4) \\ 0 & 1 & 0 \end{bmatrix} \quad (39)$$

REFERENCIAS

Yang, C. & Q. Huang, J. He, H. Jiang and J. Han (1990). Model-based control for 6-DOF Parallel Manipulator. *International Asia Conference on Informatics in Control, Automation and Robotics*.

Wang, J. & L. C.S.G. (2003). Self-Adaptive Recurrent Neuro-Fuzzy Control of an Autonomous Underwater Vehicle. *IEEE Transactions on Robotics and Automation*. Vol. 19, Issue:2.

Li, W. (1997). A method of design of a hybrid neuro-fuzzy control system based on behavior modeling. *IEEE Transactions on Fuzzy Systems*, Vol. 5, Issue:1.

Wai, R. (2010). Adaptive Fuzzy-Neural-Network velocity sensorless control for robot manipulator position tracking, *Control Theory and Applications, IET*, Vol. 4, No. 6 pp. 1079-1093.

Treesatayapun, C. & S. Uatrongjit (2005). Adaptive Controller with Fuzzy Rules Emulated Structure and Its Applications. *Engineering Applications of Artificial Intelligence*, Elsevier, Vol.18, pp.603-615.

Shimizu, M & H. Kakuya, W. Yoon, K. Kitagaki, K. Kosuge (2008). Analytical Inverse Kinematic Computation for 7-DOF Redundant Manipulators With Joint Limits and Its Application to Redundancy Resolution. *IEEE Transaction On Robotics*, Vol.24, No. 5.

Jang, J & C. Sun, E. Mitzutani (1997) *Neuro-Fuzzy and Soft Computing*, Prentice Hall.

Khalil, H. (2002). *Nonlinear Systems*, Prentice Hall.

Campa, R.; E. Torres; V. Santibáez & R. Vargas (2005). Electromechanical Dynamics Characterization of a Brushless Direct-Drive Servomotors. *Memorias VII Congreso Mexicano de Robótica* .

Ding, H.; S.K.Tso (1998). Minimum Infinity-Norm Kinematic Solution for Redundant Robots Using Neural Networks . *International Conference on Robotics and Automation* .

Assal, S.; K. Watanabe; K. Izumi (2006). Neural Network-Based Kinematic Inversion of Industrial Redundant Robots Using Cooperative Fuzzy Hint for the Joint Limits Avoidance . *ASME Transactions on Mechatronics* , Vol.11, No. 5.

Zhang, Y.; J. Wang; J. Xia (2003). A Dual Neural Network for Redundancy Resolution of Kinematically Redundant Manipulators Subject to Joint Limits and Joint Velocity Limits . *IEEE Transactions on Neural Networks*, Vol.14, No. 3.