We are IntechOpen, the world’s leading publisher of Open Access books
Built by scientists, for scientists

3,900  
Open access books available

116,000  
International authors and editors

120M  
Downloads

154  
Countries delivered to

TOP 1%  
Our authors are among the
most cited scientists

12.2%  
Contributors from top 500 universities

WEB OF SCIENCE™
Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com
1. Introduction

Fuzzy logic (FL) and artificial neural networks (ANNs), despite their successful use in many challenging control situations, still have drawbacks that limit them to only some applications. Their combined advantages have thus become the subject of much research into ways of overcoming their disadvantages. Neuro-fuzziness is one resulting rapidly emerging field. ANFIS network, proposed by Jang, is one popular neuro-fuzzy system [1-4].

For specific-problem training of an ANFIS network, [1] proposes use of hybrid learning rule, which combines gradient descent technique and least-square estimator (LSE). Being a method of supervised learning, it needs a teaching signal, which can be difficult to provide when the ANFIS network is to be a feedback controller, as the desired control actions that the teaching signal represents are unknown. Literatures have proposed several ANFIS learning methods in which ANFIS is applied as a MIMO controller. Djukanović et al., for example, uses a special ANFIS learning technique called temporal back propagation (TBP); control of a nonlinear MIMO system is by considering both the controller and the plant as a single unit each time step. The method, however, is complex and distinctly computation-heavy [5-9].

Another training approach for ANFIS-controller of nonlinear MIMO systems is inverse learning; the ANFIS network is trained to learn the inverse dynamics of the plant it controls. Its success, however, is crucial on three elements: accurate modeling of the original system (a problem when the system is complex), availability of the system’s inverse dynamics (they do not always exist), and appropriate distribution of the training data (could be impossible, given the constraints of the system’s dynamics). [10-13] has another training approach besides the ones already mentioned.
Present robot navigation systems demand controllers that can solve complex problems under uncertain and dynamic environments. ANFIS garners interest because it offers the benefits of both neural network (NN) and FL, and removes their individual disadvantages by combining them on their common features. ANN is a new motivation for studies into FL. It can be used as a universal learning paradigm in any smooth parameterized models, including fuzzy inference systems [14-15].

Traditional robot control methods rely on strong mathematical modeling, analysis, and synthesis. Existing approaches suit control of mobile robots operating in unknown environments and performing tasks that require movement in dynamic environments. Operational tasks in unstructured environments such as remote planets and hazardous waste sites, however, are more complex, yet the analytical modeling is inadequate. Many researchers and engineers have tried to solve the navigational problems of mobile robot systems [16-18]. Though fuzzy systems can use knowledge expressed in linguistic rules (and thus could implement expert human knowledge and experience), fuzzy controller lacks a systematic design method. Tuning of membership-function parameters takes time. NN learning techniques can automate the process so development can be hastened and performance improved. The combination of NN and FL has produced neuro-fuzzy controllers and created their present popularity. In real-time autonomous navigation, a robot must be able to sense its environment, interpret the sensed information to obtain knowledge of its position and environment, and plan a route that gets it to the target position from an initial position and with obstacle avoidance and control of its direction and velocity. Ng et al. [19-23] propose a neural-integrated fuzzy controller that integrates FL representation of human knowledge with NN learning to solve nonlinear dynamic control problems. Pham et al. focus on developing intelligent multi-agent robot teams capable of both autonomous action and dynamic-environment collaboration in achieving team objectives. They also propose a neuro-fuzzy adaptive action selection architecture that enables a team of robot agents to achieve adaptive cooperative control of cooperative tasks, track dynamic targets, and push boxes. Crestani et al. defines autonomous navigation in mobile robots as a search process within a navigation environment that contains obstacles and targets, and propose a fuzzy-NN controller that considers navigation direction and navigation velocity as controllable. Rutkowski et al. derived a flexible neuro-fuzzy inference system; their approach increases structural and design flexibility in neuro-fuzzy systems. Hui et al. and Rusu et al. discuss neuro-fuzzy controllers for sensor-based mobile robot navigation. Garbi et al. implemented an adaptive neuro-fuzzy inference system in robotic vehicle navigation [24-26].

Robots are one way to improve industrial automation productivity. Robotic manipulators have been used in routine and dangerous-environment manufacturing jobs. They are highly nonlinear dynamic systems subject to uncertainties. Obtaining accurate dynamic equations for their control laws is thus difficult. Uncertainties in their dynamic models include unknown grasped payloads and unknown frictional coefficients. Adaptive control or model-free intelligent control has been much proposed as able to compensate for those uncertainties [26-28].
Virtual reality (VR) has become important to applications in engineering, medicine, statistics, and other areas where 3D images can aid understanding of system complexity. The interactability of a virtual system can in many applications be enhanced by touch sensing. Haptic feedback can convey to a human user, virtual environment forces. It has become useful in tele-surgery, where a master manipulator guides robotic surgical tools while providing realistic-force feedback to the surgeon. It is already available in many systems under development but these often still are specifically developed research prototypes (i.e., providing specific-force feedbacks and for specific problems) [29, 30].

Kinematics analysis is key to motion control of humanoid manipulators. Its main problems are forward kinematics and inverse kinematics. The inverse kinematics of a 7-DOF manipulator has multiple solutions; obtaining anthropomorphic solutions is thus a problem. Detailed research in it has yet to be found, though works on design, control, and obstacle avoidance in humanoid manipulator exist. In serial manipulators, between forward kinematics and inverse kinematics, solution of the latter is a lot more difficult. Solution methods of inverse kinematics generally are numerical, analytical, or geometric. Numerical method is most widely used, but it cannot obtain all possible solutions. Analytical method can derive all possible solutions but is much more difficult. Geometric method is simple and easy to understand but suits only a few types of manipulators [31, 32].

A human-manipulator skeleton has 7-DOF mechanism; other degrees of freedom of the arm are performed by the tendon. A 7-DOF anthropomorphic arm has been developed; it has the manipulability criterion and uses 3-DOF planar manipulator theory (i.e., the mechanism of the human limb can be modeled by three moving links). 3-DOF planar mechanism is fundamental to an anthropomorphic arm; its evaluation criteria can be used to analyze the arm’s operational performance. 3-DOF planar manipulator theory is a supposition that the three link lengths are of the upper arm, the forearm, and the hand, and the three joints are 3-DOF shoulder joint, 1-DOF elbow joint, and 3-DOF wrist joint [33].

This chapter presents the design of an ANFIS controller of a VR manipulator model and simulation of the ANFIS-controlled system’s command execution. Simulation results of the 7-DOF human-manipulator show an improved control system. Section II presents the manipulator’s kinematic model whereas Section III the structure of the ANFIS controller and its learning methods. Section IV presents the manipulator’s VR model, whereas Section V, the simulation of the case study, the system design, and the conclusions.

2. Kinematics of the 7-DOF manipulator

Forward kinematics was used to calculate the racket’s posture according to the joint angles. It is quite useful for analysis of the manipulator’s workspace and verification of the inverse kinematics. Denoting the shoulder width as $D$, the upper arm length as $L_1$, and the lower arm length as $L_2$, the position of the shoulder is thus $P_1(0, -D, 0)$ and the position of the elbow is $P_2(x_{p2}, y_{p2}, z_{p2})$. Denoting the joint angles as $q_1 \ldots q_7$, the posture of joint $i$ relative to joint $i-1$ can thus be described by a 4×4 homogeneous matrix $^{i-1}T_i$. [34]:

$$^{i-1}T_i$$
Where: \( \text{sq}_i \equiv \sin(q_i) \); \( \text{cq}_i \equiv \cos(q_i) \).

Assuming the position of Joint-7 in the fixed coordinate is \( P_3(x_p^3, y_p^3, z_p^3) \) and its pose as described by RPY (Roll Pitch Yaw) angles is \( (\phi, \theta, \psi) \), the posture of Joint-7 can thus be described by a homogenous matrix \( ^0T_7 \):

\[
^0T_7 = \begin{bmatrix}
\text{cq}_0 & \text{cq}_1 & \text{cq}_2 & \text{cq}_3 & \text{cq}_4 & \text{cq}_5 & \text{cq}_6 & x_p^3 \\
\text{sq}_0 & \text{sq}_1 & \text{sq}_2 & \text{sq}_3 & \text{sq}_4 & \text{sq}_5 & \text{sq}_6 & y_p^3 \\
0 & 0 & 0 & 1 & z_p^3
\end{bmatrix}
\]

From (1) and (2), the racket’s posture can be calculated as:

\[
\varphi = \tan(2(n_y n_x)) & \quad x_{p3} = p_x \\
\theta = \tan(-n_x \text{cp} \psi + \text{sn} \psi) & \quad y_{p3} = p_y \\
\psi = \tan(2(\text{sq} n_x - \text{sn} n_y)) & \quad z_{p3} = p_z
\]

(4)

with \( \tan() \) being the four-quadrant inverse tangent function and (3) the manipulator’s forward kinematics. All the commands (e.g., the racket’s hitting position and hitting speed) are given by the visual system, in the operation space. They must be transformed into joint space values. Jacobian matrix is used to calculate the joint space speed according to the operation space speed. The mapping relationship between them is [35]:

\[
V = J(q) \dot{q}
\]

(5)

with \( V \) being the racket’s speed in the operation space and \( \dot{q} \) the joint space speed. \( J \) is the 6x7 Jacobian matrix and can be derived by differential transformation method. The \( i \)th item of \( J \) is:
Design and Simulation of Anfis Controller for Virtual-Reality-Built Manipulator

\[
J_i = \begin{bmatrix}
(p_i \times n_i)_k \\
(p_i \times o_i)_k \\
(p_i \times a_i)_k \\
n_{ik} \\
o_{ik} \\
a_{ik}
\end{bmatrix}
\]

(6)

Where \(n(n_x, n_y, n_z), o(o_x, o_y, o_z), a(a_x, a_y, a_z)\) and \(p(p_x, p_y, p_z)\) are the items of matrix \(i^{-1}T_i\).

\[
i^{-1}T_j = i^{-1}T_i \cdots i^{-1}T_j = \begin{bmatrix}
n_x & o_x & a_x & p_x \\
o_y & o_y & a_y & p_y \\
n_z & o_z & a_z & p_z \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

(7)

The parameter \(k\) stands for the rotational axis of joint \(i\). For example, if the joint rotates around the \(x\)-axis, then \(k\) is \(x\). Racket speed \(V\) can be calculated by (4) whereas joint space speed \(q\) cannot be uniquely determined because \(J\) is not a square matrix. This problem can be solved by Moore-Penrose method:

\[
q = J^+V
\]

(8)

with \(J^+ = (J^TJ)^{-1}J^T\) being the Moore-Penrose pseudo inverse matrix of \(J\).

The inverse kinematics is used to calculate the joint angles according to the racket’s posture. The manipulator is redundant, so the elbow’s position is not uniquely determined when the racket’s posture is given. The motion characteristics of a human arm show the mapping relationships between elbow position and racket posture. In a general configuration of the manipulator (see Fig. 2), any three points on the manipulator’s neck, shoulder, elbow, and wrist are not collinear. The position of elbow \(P_2\), whose axis is \(P_1P_3\), is in circle \(O\). The position of the center point \(O = (x_O, y_O, z_O)\) and the radius \(r\) of the circle are determined by positions \(P_1\) and \(P_3\), and arm length parameters \(L_1\) and \(L_2\). \(\Omega_1\) denotes the plane constructed by points \(P_1, P_2, P_3\), and \(\Omega_2\) the plane constructed by points \(O, P_1, P_2\). \(\alpha\) denotes the separation angle between \(\Omega_1\) and \(\Omega_2\). Once the position of \(P_3\) is known, \(\alpha\) can uniquely determine \(P_2\) elbow position. If the points \(O, P_1, P_3\) are collinear whereas \(P_1, P_2, P_3\) are not, the plane \(\Omega_2\) does not exist. Angle \(\alpha\) can be defined as the separation angle between \(\Omega_1\) and the horizontal plane. Also, if the points \(P_1, P_2, P_3\) are collinear, the plane \(\Omega_1\) does not exist and \(P_2\) position can be calculated according to positions \(P_1\) and \(P_3\) and arm length parameters \(L_1\) and \(L_2\). Considering the general configuration of the manipulator (see Fig. 2), the inverse kinematics can be solved if angle \(\alpha\) and the racket’s posture are given. According to the motion characteristics of human arms, the mapping relationship between \(\alpha\) and the racket posture can be built offline. Denoting with \(i(i_x, i_z, i_i)\) the unit vector pointing from \(P_3\)
to P₁, and assuming the angle α to have been calculated by a well-trained ANN model, the circle O' can thus be expressed as [34-36]:

\[
\begin{align*}
&i_1(x - x_o) + i_2(y - y_o) + i_3(z - z_o) = 0 \\
&(x - x_o)^2 + (y - y_o)^2 + (z - z_o)^2 = r^2
\end{align*}
\]  

(9)

Figure 1. General configuration of the manipulator

The plane Ω₂ and the circle O' intersect at two points. One point near the neck is P₂ (x₂, y₂, z₂) and angle P₂O'P₂ is the separation angle α between Ω₁ and Ω₂. Denoting Ω₂ norm vector as \( \mathbf{n} \) and position as P₂ should satisfy this equation:

\[
\mathbf{OP}_w \cdot \mathbf{n} = 0
\]  

(10)

P₂ position can be calculated from (9) and (10). P₂ position (x₂p, y₂p, z₂p) satisfies this equation:

\[
\sqrt{(x_{p2} - x_m)^2 + (y_{p2} - y_m)^2 + (z_{p2} - z_m)^2} = 2r \sin \frac{\alpha}{2}
\]  

(11)

P₂ position can be calculated from (9) and (11). With P₁, P₂, P₃ positions and the cosine theorem, Joint-4 angle can be calculated as:

\[
q_4 = \pi - \alpha \cos \left( \frac{L_1^2 + L_2^2 - \|P_1P_3\|^2}{2L_1L_2} \right)
\]  

(12)
With P2 position, this can be obtained:

\[
\begin{bmatrix}
 x_{p2} & y_{p2} & z_{p2} & 1
\end{bmatrix}^T = T_1^1 T_2^2 T_3^3 \begin{bmatrix} 0 & 0 & -L_1 & 1 \end{bmatrix}^T
\]  

(13)

The angles of Joints 1 and 2 can thus be derived as:

\[
\begin{align*}
q_1 &= a \tan 2(\frac{-x_{p2}}{c_{q_2} L_1}, \frac{-z_{p2}}{c_{q_2} L_1}) + k\pi \\
q_2 &= a \sin((y_{p2} + D) / L_1)
\end{align*}
\]  

(14)

With P3 position, this can be obtained:

\[
\begin{bmatrix}
 x_{p3} & y_{p3} & z_{p3} & 1
\end{bmatrix}^T = 0 T_1^1 T_2^2 T_3^3 T_4^4 \begin{bmatrix} 0 & 0 & -L_2 & 1 \end{bmatrix}^T
\]  

(15)

Joint-3 angle can thus be derived as:

\[
\begin{align*}
q_3 &= a \tan 2(\frac{y_3 + D - sq_2 L_1 - sq_2 cq_4 L_2 - sq_2 sq_4 L_2}{c_{q_2} sq_2 sq_4 L_2}, \frac{x_3 + sq_4 cq_2 (1 + cq_4)L_1}{sq_2 cq_1 L_1}) \\
&= 0 \tan 2((y_{p3} + D) / L_2)
\end{align*}
\]  

(16)

From (2),

\[
T_5^4 T_6^5 T_7^6 = [T_1^1 T_2^2 T_3^3 T_4^4]^{-1} T_7^6 = T^*
\]  

(17)

The angles of Joints 5-7 can thus be derived as:

\[
\begin{align*}
q_5 &= a \tan 2(-T_{12}^*, T_{23}^*) + k\pi \\
q_6 &= a \tan 2((sq_5 T_{13}^* - sq_5 T_{23}^*), T_{33}^*) \\
q_7 &= a \tan 2((-cq_3 T_{12}^* - sq_3 T_{22}^*, cq_3 T_{13}^* + sq_3 T_{23}^*))
\end{align*}
\]  

with $T_{ij}^*$ being the $i^{th}$ row and $j^{th}$ column item of $T$.

3. General structures of ANFIS

ANFIS integrates ANN with FIS. The ANFIS analyzed here was a first-order Takagi Sugeno Fuzzy Model. The analysis has four inputs: front obstacle distance ($x_1$), right obstacle distance ($x_2$), left obstacle distance ($x_3$), and target angle ($x_4$). The output is steering angle. The ‘if-then’ rules are [26, 37]:

\[
\begin{align*}
\text{Rule: IF } & \quad x_1 \quad \text{is} \quad A_1; \quad x_2 \quad \text{is} \quad B_1; \quad x_3 \quad \text{is} \quad C_m \quad \text{and} \quad x_4 \quad \text{is} \quad D_n \quad \text{THEN} \quad F_j = p_j x_1 + r_j x_2 + s_j x_3 + t_j x_4
\end{align*}
\]  

(19)
Where

\[ F_i = p_i x_i + r_i x_2 + s_i x_3 + t_i x_4 + u_i \quad \text{for steering angle} \]

\[ j = 1 \text{ to } q_j; \quad k = 1 \text{ to } q_2; \quad m = 1 \text{ to } q_3; \quad n = 1 \text{ to } q_4; \quad \text{and} \quad i = 1 \text{ to } q_1 q_2 q_3 q_4 \quad (20) \]

A, B, C, and D are the fuzzy membership sets defined for input variables \( x_1, x_2, x_3, \) and \( x_4 \).

\( q_1, q_2, q_3, \) and \( q_4 \) are the number of membership functions, respectively for the fuzzy systems of inputs \( x_1, x_2, x_3 \) and \( x_4 \). \( f_i \) is the linear consequent functions defined in terms of inputs \( x_1, x_2, x_3, \) and \( x_4 \). \( q, r, s, t, \) and \( u \) are consequent parameters of an ANFIS fuzzy model. Same-layer nodes of an ANFIS model have similar functions. Output signals from the nodes of a preceding layer are input signals to a present layer. The output obtained through the node function will be input signals to the next layer (see Figure 2) [38, 39].

Layer 1: This is the input layer, which defines obstacles as either static or moving and also the tracker robot’s target position. It receives signals from \( x_1, x_2, x_3, \) and \( x_4 \).

Layer 2: Every node in this layer is an adaptive node (square node) with a particular fuzzy membership function (node function) specifying the degrees to which the inputs satisfy the quantifier. For four inputs, the node outputs are:

\[ L_{2g} = U_{sg}(x) \quad \text{for} \quad g = 1, \ldots, q_1 \quad (\text{for input } x_1) \]

\[ L_{2g} = U_{bg}(x) \quad \text{for} \quad g = q_1 + 1, \ldots, q_1 + q_2 \quad (\text{for input } x_1) \]

\[ L_{2g} = U_{cg}(x) \quad \text{for} \quad g = q_1 + q_2 + 1, \ldots, q_1 + q_2 + q_3 \quad (\text{for input } x_1) \]

\[ L_{2g} = U_{dg}(x) \quad \text{for} \quad g = q_1 + q_2 + q_3 + 1, \ldots, q_1 + q_2 + q_3 + q_4 \quad (\text{for input } x_1) \quad (21) \]

The membership functions considered here for A, B, C, and D are bell-shaped functions and defined as:
\[\mu_a(x) = \frac{1}{1 + \left(\frac{x - c_a}{a_a}\right)^2}; \quad g = 1 \text{ to } q_1\]

\[\mu_{b_g}(x) = \frac{1}{1 + \left(\frac{x - c_{b_g}}{a_{b_g}}\right)^2}; \quad g = q_1 + 1 \text{ to } q_1 + q_2\]

\[\mu_{b_g}(x) = \frac{1}{1 + \left(\frac{x - c_{b_g}}{a_{b_g}}\right)^2}; \quad g = q_1 + q_2 + 1 \text{ to } q_1 + q_2 + q_3\]

\[\mu_{b_g}(x) = \frac{1}{1 + \left(\frac{x - c_{b_g}}{a_{b_g}}\right)^2}; \quad g = q_1 + q_2 + q_3 + 1 \text{ to } q_1 + q_2 + q_3 + q_4\]

(22)

with \(a_a, b_g, \) and \(c_g\) being the parameters for fuzzy membership function. The bell-shaped function changes its pattern with changes to the parameters. This change will give various contours of the bell-shaped function, as needed and in accordance with the data set for the problem considered.

**Layer 3:** Every node in this layer is a fixed node (circular) labeled ‘\(\pi\).’ \(L_3\) output is the product of all incoming signals.

\[L_{3i} = W_i = \mu_{a_g}(x), \mu_{b_g}(x), \mu_{c_g}(x), \mu_{c_g}(x); \quad \text{For } i = 1, \ldots, q_1 + q_2 + q_3 + q_4 \text{ and } g = 1, \ldots, q_1 + q_2 + q_3 + q_4\]

(23)

Each of the second layer’s node output represents the firing strength (degree of fulfillment) of the associated rule. The T-norm operator algebraic product \([T_{\text{ap}}(a,b) = ab]\) was used to obtain the firing strength \((W_i)\).

**Layer 4:** Every node in this layer is a fixed node (circular) labeled “\(N\)” The output of the \(i^{th}\) node is the ratio of the firing strength of the \(i^{th}\) rule \((W_i)\) to the sum of the firing strength of all the rules [40].

\[L_{4i} = \overline{W_{ij}} = \frac{W_i}{\sum_{r=1}^{4} W_r}\]

(24)

This output gives a normalized firing strength.
Layer 5: Every node in this layer is an adaptive node (square node) with a node function.

\[ L_{ij} = \overline{W}_j f_i = \overline{W}_j (p_i x_1 + r_i x_1 + s_i x_3 + t_i x_4 + u_i) \quad (25) \]

with \( \overline{W}_j \) being the normalized firing strength form (output) from Layer-3 and \([p_i, r_i, s_i, t_i, u_i]\) the steering-angle parameter set. Parameters in this layer are consequent.

Layer 6: The single node in this layer is a fixed node (circular) labeled “Σ”. It computes the overall output as the summation of all incoming signals.

\[ L_{Gj} = \sum_{r=1}^{r=q_1, q_2, q_3, q_4} \sum_{i=1}^{i=q_1, q_2, q_3, q_4} \overline{W}_j f_i \quad (26) \]

This work’s ANFIS development has six-dimensional space partitions and \(q_1, q_2, q_3, \text{and } q_4\) regions. Each region is governed by a fuzzy if-then rule. The first layer is the input layer. The second contains premise or antecedent parameters of the ANFIS and is dedicated to fuzzy sub-space. Consequent parameters of the fifth layer were used to optimize the network. During the forward pass of the hybrid learning algorithm, node outputs go forward until Layer-5 and the consequent parameters are identified by least-square method. In the backward pass, error signals propagate backwards and the premise parameters are updated by gradient descent method [40, 41].

4. VR Modeling of the 7-DOF manipulator

Design requirements for Virtual Reality Modeling Language (VRML) are described in finite processing allocations, autonomy, consistent self-registration, and calculability. VRML design procedure will be presented. Design in VRML depends on the information available to the designer and his imaging of the object. There are two choices for VR design: one is standard configuration such as sphere, cone, cylinder, etc., another is free design by selecting indexed face set button to get many configurations with free rearrangement of points; every real-form design is thus considered to be the latter [42-45], which starts with building parts one by one and comparing the shape’s similarity against that of the real manipulator part. Manipulator parts cannot be simulated in VR when the VR library’s standard shapes (they are not uniform) are used. Designing thus uses indexed face set in VR. The second choice to be made in design work is very important as it is about connecting all parts to get the final object, and limiting the object’s original point. This is the starting point of the design work. The first shape (e.g., the shoulder) is first set, and then the next shape (forearm) is connected to the “children” button. The same procedure is repeated for other parts. Figure 3 presents the full design of the 7-DOF human arm manipulator [46-50].
The design work considered many parameters related to the system’s real values (see Table 1 for real limits of the joints of a human arm). The joints motor is considered a real DC-motor with a transfer function similar with that in Equation (1).

\[ \frac{T_F(s)}{s} = \frac{1}{s^2 + 6s + 8} \]  

(27)

Figure 4 is a block diagram of the control system. The inputs to the system design have two sets of targets: orientation (\(\theta_1, \theta_2, \theta_3\)) and position (\(T_x, T_y, T_z\)). In the proposed technique, each joint has its own controller, so in all, seven identical (same structure, same training algorithm) ANFIS controllers were used. Inputs of the training algorithm were the desired joint angle and the actual angle. The desired values were calculated by using analytical solution of the IKP algorithm shown previously. Actual values of the joint angle were obtained by feedback from the virtual model.
The structure of the ANFIS controller was built in Matlab software Ver.2011b, with two inputs, an error signal, and change in the error. The fuzzy inference method used was Mamdani's, because it is intuitive, widely accepted, and well-suited to human input, and, for the proposed control structure, it gives better results than does Sugeno inference method. In designing the controller, types of membership functions were tried before selecting the best: triangular built-in membership function (trimf). The trial-and-error design approach for the ANFIS controller, i.e., selecting the interference type, the membership function type, and the number of this membership function in the hidden layer, gave optimal results: minimum number of rules and simple simulation. The design uses seven parallel-connected ANFIS to compute the optimal deflection of the joints and get the desired angle. Figures (5-7) show the procedure of the Matlab-fuzzy-toolbox-based design.

Figure 4. ANFIS control of the 7-DOF human-arm manipulator

Figure 5. Mamdani’s fuzzy inference and the set of rules for training of the ANFIS controller
Figure 6. The surface error and the set of rules for training of the ANFIS controller.

Figure 7. The triangular built-in memberships function with error signal and change in error signal.
Figure 8 shows in Matlab Simulink GUI window the internal structure of the ANFIS controller and the training output for the input signal, with the various steps used in instructing the manipulator’s movement. The ANFIS training used hybrid training algorithm, with the input nodes (3, 3) to the membership functions each having nine rules (see Figure 5). Epoch length was used in training eighty iterations for each sample, with 0.01s Simulink sampling time.

Figure 8. The ANFIS structure in Matlab-Simulink GUI and its training response
6. Simulation results

Figure 9(a) is a block diagram of the simulation done for the ANFIS-controlled manipulator. Seven ANFIS controllers were used, one each for each degree of freedom of the three joints (3 for the shoulder, 1 for the elbow, and 3 for the wrist). The seven were effective in tracking the trajectory desired for the 7-DOF manipulator. The controller’s rules base has 9 rules, each determined by fuzzy neural network (FNN). The desired position and orientation were, in simulation entered as input signal, whereas actual positions for the joints were given as feedback from the output signal. Figure 9(b) shows the results of using both ANFIS and PID controllers on the joints and for the ANFIS controller’s implementation into the virtual model. Performance of the ANFIS-controlled joint was better than that of the PID-controlled one. The ANFIS controller effected fast response in the manipulator and reduced errors, for various complex trajectories of the manipulator.

The manipulator’s movements used the link between Matlab-Simulink and VR environment. Computation for the order of movement was done in Matlab-Simulink. The order was then sent to the VR model to implement, with considerations for the axes between
Matlab’s and VR’s to get the real movement (i.e., for general motion: elbow flexion/extension, elbow rotation (supination/pronation), shoulder adduction/abduction, shoulder flexion/extension, shoulder interior/exterior rotation, wrist flexion/extension, wrist ulnar/radial deviation, shoulder horizontal flexion/extension; for special motion: arm reaching towards the head, arm reaching to the right and the head, arm reaching to the left and the head, and arm reaching to the right) (see Figure 10 for examples).

Figure 10. Postures of the ANFIS-controlled 7-DOF manipulator

7. Conclusion

This work aimed to design an ANFIS-based controller that overcomes the general problems of FL and NN in a dynamic system. To obtain excellent manipulator postures, training of the ANFIS controller for all the elements (fuzzy interference, membership function, number of neurons, number of rules) was by trial and error. The controller was compared with classical PID controller, in tracking the speed and the joint angle’s accuracy. The simulation results allowed these conclusions to be drawn:
For any speed, the ANFIS controller has the better transient response and the steadier state response than does a PID controller; even the best-tuned PID controller is unable to perform well in both slow-speed and high-speed ranges.

With only 9 neurons, 4 layers, and 9 rules, the proposed ANFIS controller is simpler than other adaptive neuro-fuzzy controllers reported by many other researches.

The ANFIS controller does not require an accurate model of the plant. Its relative simplicity makes it fairly easy to construct and implement. High-level knowledge of system is not needed to build a set of rules for a fuzzy controller or for the identification needed in an NN controller.

The ANFIS controller was used in both simulation and experiment in trajectory tracking of multiple manipulators. Kinematic analysis of the 7-DOF manipulator produced the calculation for the envelope of similar manipulators.

VR implementation of the manipulator was able to show the latter’s accuracy. The manipulator can be depended on for many high-accuracy applications such as collecting of minute/fragile items.

Author details
Yousif I. Al Mashhadany
Electrical Dept., Al Anbar University, Engineering College, Baghdad, Iraq

Acknowledgement
Special thanks to Ms. Wirani M. Munawir of Verdana Inc. (verdana.inc@live.com) for her editing of the text.

8. References


[18] Chitra V, (2010), “ANFIS Based Field Oriented Control for Matrix Converter fed Induction Motor”, IEEE international conference on power and energy (PECon2010), Malaysia, 74-78


[31] Jacob R, Joel C. P, Nathan M, Stephen B, Blake H, (2005), “The Human Arm Kinematics and Dynamics During Daily Activities – Toward a 7 DOF Upper Limb Powered Exoskeleton”, This work is supported by NSF Grant #0208468 entitled “Neural Control of an Upper Limb Exoskeleton System” - Jacob Rosen (PI) 0-7803-9177-2/05/ IEEE, 532-539


[40] Mukhtiar S, Chandra A, (2012), “Real Time Implementation of ANFIS Control for Renewable Interfacing Inverter in 3P4W Distribution Network”, This article has been accepted for publication in a future issue of IEEE journal


[45] Michal J, Michal P, Michal C , Peter Nand O, (2012), “Towards Incremental Development of Human-Agent-Robot Applications using Mixed-Reality Testbeds”, This article has been accepted for publication in IEEE Intelligent Systems but has not yet been fully edited. Some content may change prior to final publication.


