Fuzzy Hint Acquisition for the Collision Avoidance Solution of Redundant Manipulators Using Neural Network

Samy F. M. Assal, Keigo Watanabe, and Kiyotaka Izumi

Abstract: A novel inverse kinematics solution based on the back propagation neural network (NN) for redundant manipulators is developed for online obstacles avoidance. A laser transducer at the end-effctor is used for online planning the trajectory. Since the inverse kinematics in the present problem has infinite number of joint angle vectors, a fuzzy reasoning system is designed to generate an approximate value for that vector. This vector is fed into the NN as a hint input vector rather than as a training vector to guide the output of the NN. Simulations are implemented on both three- and four-link redundant planar manipulators to show the effectiveness of the proposed position control system.

Keywords: Fuzzy system, inverse kinematics, neural networks, online collision avoidance, redundant manipulators.

1. INTRODUCTION

Kinematically redundant manipulators are those that have more degrees-of-freedom than that required for a given task. Such manipulators offer several advantages over non-redundant ones in many aspects such as singularity and obstacles avoidance, keeping the joints within their limits and torque optimization. Those objectives are considered to be a second subtask while the end-effector tracking is to be the first.

Collision avoidance problem is the most important investigating subtask for those manipulators. It must be achieved carefully; otherwise, serious damage may occur to the manipulator as well as the surrounding environment. This problem can be solved either off-line or online. Off-line collision avoidance is specified as follows: given the end-effector trajectory, a set of obstacles in space and initial manipulator configuration, then require the joint trajectory for the collision avoidance. This approach needs expensive computations for global path planning before the manipulator starts its motion. On the other hand, online obstacles avoidance enables those manipulators to move in an unstructured environment including both fixed and moving obstacles without collision depending on local path planning which is based on sensor information.

Many techniques for the off-line problem tried to represent the workspace by analytical model in order to measure the distance between the obstacles and the links. In [1], the obstacles and links were represented by spheres and ellipsoids respectively and the collidability measure which was defined as the sum of the inverse of predicted collision distances between links and the obstacles was introduced in the optimization process. The obstacles avoidance in the approach that was applied for the path planning in [2] was expressed in terms of the distance between potentially colliding parts. Since multi-layer neural network (NN) has been used to learn the direct and the inverse kinematics of robot manipulators successfully [3-5], an NN approach was introduced in [6] to solve the inverse kinematics problem in an environment with obstacles. Training patterns for both obstacle free and obstacle avoidance cases have been developed, where the training algorithm was augmented with a distance penalty function for obstacles avoidance case. In this approach, a ball-covering object modeling technique was used efficiently for calculating the distance between the links and obstacles. An adaptive fuzzy logic approach in which the obstacles were modeled by convex bodies was proposed in [7]. A minimum distance between the links and obstacles that was taken to be greater than a preplanned threshold distance was presented as a criterion for the obstacles avoidance.

On the other hand, several strategies have been

Manuscript received October 5, 2004; accepted November 4, 2005. Recommended by Editorial Board member Eun Tai Kim under the direction of past Editor-in-Chief Myung Jin Chung.

Samy F. M. Assal is with the Department of Advanced Systems Control Engineering, Graduate School of Science and Engineering, Saga University, 1-Honjomachi, Saga 840-8502, Japan. His permanent address: Dept. of Production and Mechanical Design, Faculty of Engineering, Tanta University, Egypt (e-mail: sfassal@ieee.org).

Keigo Watanabe and Kiyotaka Izumi are with the Department of Advanced Systems Control Engineering, Graduate School of Science and Engineering, Saga University, 1-Honjomachi, Saga 840-8502, Japan (e-mails: {watanabe, izumi}@me.saga-u.ac.jp).

developed for online problem. Collision avoidance using reflex control was proposed in [8]. It described enhancements to reflex control through the use of a higher level attractive potential function which helps to guide the reflexes for simultaneous goal attraction and obstacle avoidance. For the real-time collision avoidance proposed in [9], the obstacles avoidance requirement was represented as inequality constraints in the workspace and those inequalities were satisfied while the end-effector tracks the desired trajectory. The scheme was based on the implementation of the configuration control [10]. A new approach based on velocity potential function was proposed in [11] for the repulsive function, which has no local minima even for a cluttered environment. A kinematic control algorithm, which is based on the redundancy resolution at the velocity level, was proposed in [12]. The obstacles avoidance strategy is to assign each point on the manipulator, which is close to the obstacle, a velocity component in a direction that is away from the obstacle. An iterative solution of the Inverse Geometric Model (IGM) was used in [13] for the real-time operation of redundant robots. A pseudo-distance that was introduced for modeling various obstacles leaded to the formulation of the anti-collision constraints. Those constraints were introduced in the objective function which was minimized yielding the inverse solution of the IGM algorithm. Recently, NN approach has been applied for the real-time obstacles avoidance of redundant manipulators. In [14], the Lagrangian network [15] was applied for this problem with the incorporation of the obstacles avoidance strategy proposed in [24]. A dual NN was developed in [16] for online solution to collision-free inverse kinematics problem. The collision avoidance requirement was represented by dynamically-updated inequality constraints. Physical constraints such as joint limits were also included in the formulation.

The classical Jacobian-based redundancy resolution techniques for solving the collision avoidance problem are quantitative approaches in nature; that is, they rely on quantitative and precise description of the world and the task for choosing the optimal configuration, which is not often available in the real-world. Also, they are inflexible approaches in the sense that they depend on optimizing a fixed performance criterion during the whole run-time which is not suitable for any change in the environment. Therefore, in this paper, a new concept is developed for the online inverse kinematics solution of redundant manipulators to reach inside an enclosed space with unknown configuration by applying a back propagation NN and a fuzzy logic system. To eliminate the need for object modeling and distance calculation for this sophisticated task, a new method for planning the trajectory using an exploring sensor is first presented. To solve an inverse kinematics problem of redundant manipulators by using the NN, the concept of limiting the searching space is next presented by using a hint generator provided by a fuzzy reasoning system. An illustrative numerical example for that concept is also presented. The construction of each module of the proposed position control system is elaborated. Simulations are conducted on both three- and four-link redundant planar manipulators to prove the capability of the manipulator to reach inside any enclosed space avoiding the collision with its boundaries and to obtain the corresponding joint angle profiles. Some discussions regarding the present approach are presented finally.

2. PROBLEM SETTING AND EXPLORING SENSOR

The task under consideration is to control the redundant manipulator to reach inside an enclosed space, whose configuration is not known in advance. This task is required for many real applications. Situations that can not be accessed or discovered by the human hand are generally spoken; some are working in a cavity, picking and placing objects in inaccessible locations, surface inspection inside a duct or a tunnel and working in a space behind a long opening. The collision avoidance with this space is the most important subtask, except for planning and tracking a trajectory to be the first subtask. Since the global path planning is impossible to realize in this application, the problem needs to use an online planning. Collision avoidance remains a difficult problem because almost all the manipulator links are bounded by obstacles, i.e. many active obstacles at the same time. The approach that depends on measuring the distance between the obstacles and each point on the manipulator links that is close to the obstacle, is not practical and time consuming. In addition, the use of fixed vision system to recognize the entire scene for measuring this distance is not possible in this task. Therefore, the present approach treats the problem from another viewpoint; briefly, it is based on planning a trajectory which permits for all the links to negotiate their ways around this trajectory within permissible limit to avoid the collision, specifically, controlling each angle between the links and the trajectory within its permissible limit. Thus, the need for object modeling and distance calculation can be eliminated.

The present approach depends on using an exploring sensor, a laser transducer at the end-effector for online planning the trajectory inside the enclosed space. This sensor rotates in a two-axis gyroscopic motion, as shown in Fig. 1, to scan the enclosed space for the maximum distance. Specifically, it rotates, in

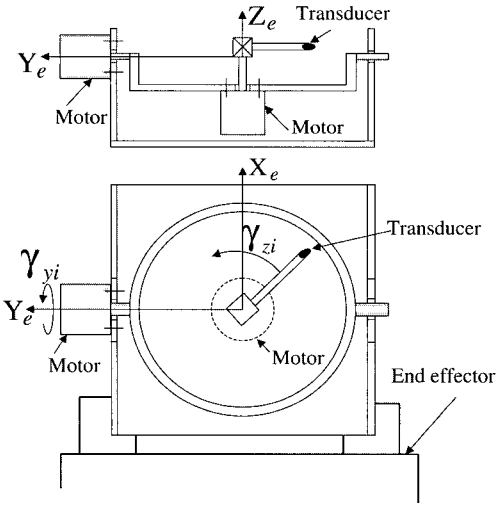


Fig. 1. Construction of the exploring sensor.

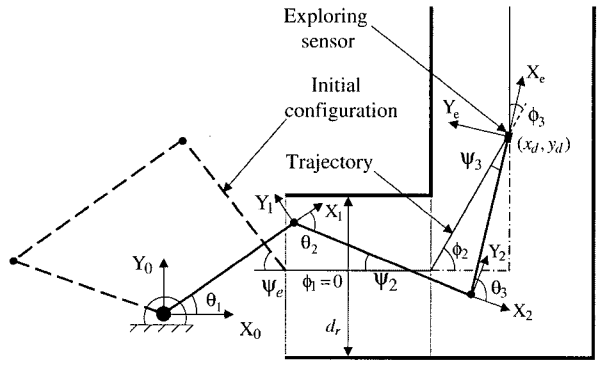


Fig. 2. Task and parameters description.

rotational steps, γ_{yi} around Y_e-axis, and then at each rotational step of γ_{yi}, it rotates γ_{zi} around Z_e-axis. Thus, it records the angle γ_{yi} for the plane that has the maximum distance recorded and then within this plane it records the angle γ_{zi} that corresponds to the bisector of the angular range between the equal maximum distances recorded in this plane. Those two angles $(\gamma_{yi}, \gamma_{zi})$ define the orientation of each segment of the planned trajectory with respect to the end-effector coordinate system (X_e, Y_e, Z_e), because the resulting planned trajectory will be a multi-segment line due to the repeated scanning of the enclosed space after tracking each planned segment. Using this exploring sensor, the problem can be reduced to a kind of inverse kinematics problem of redundant manipulators: given the Cartesian coordinate values of the end-effector, search the joint angles for the collision avoidance.

In Fig. 2, a coordinate frame (X_i, Y_i, Z_i) for $i = 0,1,\cdots,n-1$, is assigned to the joint of each link of the n serial links redundant manipulator to describe the relative position and orientation between links according to the Denavit-Hartenberg (D-H) representation [17]. It is assumed that the coordinate frame of the end-effector is the same as that of the exploring sensor. The joint variable θ_i, which is rotating about Z_{i-1}-axis, is the angle between the X_{i-1}-axis and X_i-axis. The following two variables are required to be defined in this approach.

Definition 1 (ϕ_i angles): They are the angles between every two consequent intersected segments of the planned trajectory.

To calculate those angles, let $\boldsymbol{u}_{si}$ represent a unit vector along each segment of the planned trajectory with respect to the end-effector coordinate system, and $\boldsymbol{u}_t$ be a unit vector along X_e-axis representing the initial position of the transducer. Then, $\boldsymbol{u}_{si}$ can be calculated by rotating $\boldsymbol{u}_t$ an angle γ_{yi} around Y_e-axis then an angle γ_{zi} around Z_e-axis, such that

$$\begin{aligned}\boldsymbol{u}_{si} &= R_z(\gamma_{zi})R_y(\gamma_{yi})\boldsymbol{u}_t \\ &= \begin{bmatrix}\cos\gamma_{zi} & -\sin\gamma_{zi} & 0\\ \sin\gamma_{zi} & \cos\gamma_{zi} & 0\\ 0 & 0 & 1\end{bmatrix}\times\begin{bmatrix}\cos\gamma_{yi} & 0 & \sin\gamma_{yi}\\ 0 & 1 & 0\\ -\sin\gamma_{yi} & 0 & \cos\gamma_{yi}\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix} \\ &= \begin{bmatrix}\cos\gamma_{zi}\cos\gamma_{yi}\\ \sin\gamma_{zi}\cos\gamma_{yi}\\ -\sin\gamma_{yi}\end{bmatrix},\end{aligned} \tag{1}$$

where i denotes the index for the number of the segment.

Thus, ϕ_i can be calculated by using the dot product of two unit vectors. Each vector is along each segment. Considering the first segment of the trajectory as a reference and has $\phi_1 = 0$, the following equation holds

$$\phi_{i+1} = \cos^{-1}\frac{\boldsymbol{u}^b_{si}\cdot\boldsymbol{u}^b_{s(i+1)}}{\left\|\boldsymbol{u}^b_{si}\right\|\left\|\boldsymbol{u}^b_{s(i+1)}\right\|} = \cos^{-1}(\boldsymbol{u}^T_{si}\,\boldsymbol{u}_{s(i+1)}), \tag{2}$$

where

$$\boldsymbol{u}^b_{si} = R^e_0\boldsymbol{u}_{si}$$

in which R^e_0 is the rotational matrix relating the base coordinate system and the end-effector coordinate system.

Definition 2 (Cone angles, ψ_j): They are the angles between each link being inside the enclosed space and the corresponding intersected segment of the planned trajectory as shown in Fig. 2.

They are defined as a measure for both the most dexterous configuration before going inside (ψ_e angle) and the criterion of the collision avoidance while going inside (ψ_j angles). The maximum permissible limit of the cone angles (ψ_j) is

$$\psi_{j\max} = \sin^{-1}\left(\frac{d_r}{2a_j}\right), \tag{3}$$

where d_r is the diameter of the enclosed space and a_j is the length of link j. The cone angle ψ_e at the entrance can be calculated by using the dot product of two unit vectors: the first is $\boldsymbol{u}_{s1}$ while the second is a unit vector along X_e-axis of the end link $\boldsymbol{u}_e = [1 \quad 0 \quad 0]^T$ such that

$$\begin{aligned}\psi_e &= \cos^{-1}\frac{\boldsymbol{u}_{s1}\cdot\boldsymbol{u}_e}{\|\boldsymbol{u}_{s1}\|\|\boldsymbol{u}_e\|} \\ &= \cos^{-1}(\boldsymbol{u}_{s1}^T\boldsymbol{u}_e) \\ &= \cos^{-1}\left(\cos\gamma_{z1}\cos\gamma_{y1}\right).\end{aligned} \tag{4}$$

3. CONCEPT FORMULATION

3.1. Hint generator

The proposed control system for this problem is shown in Fig. 3. The fuzzy system works as a hint generator. A hint is an approximate solution of the inverse kinematics problem. Specifically, the fuzzy system generates approximate vector $\boldsymbol{\theta}_h$ for the joint angle vector of the manipulator $\boldsymbol{\theta} \in \Re^n$. All elements of $\boldsymbol{\theta}_h$ are within the values required to avoid the collision with the enclosed space. They are fed to the NN, which works in the inverse of the kinematics problem as a second input vector rather than as

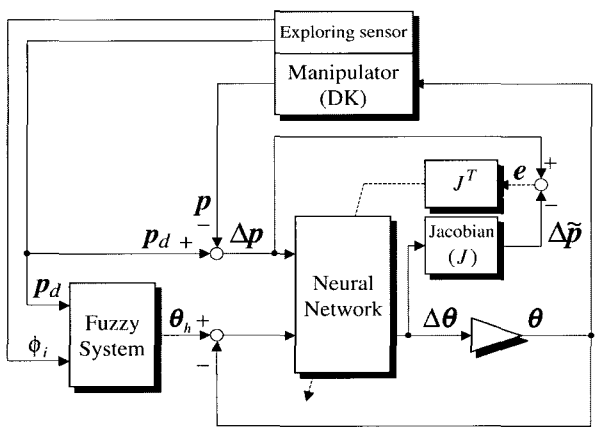


Fig. 3. The proposed control system.

training examples [18]. In [18], a systematic method for incorporating hints (prior information about the function) in the usual learning from example process was developed. Since the inverse kinematics of redundant manipulator has infinite number of solutions, the hints generated from the fuzzy system are fed into the NN to limit the searching space, to accelerate and to improve the quality of the solution. Specifically, the inverse kinematics solution by the NN converges rapidly to the required joint angle vector $\boldsymbol{\theta}$ that is very close to $\boldsymbol{\theta}_h$ among all possible vectors to achieve the desired end-effector position at each time step along the planned trajectory. Since there is no reference output for the NN, the output $\Delta\boldsymbol{\theta}$ is mapped to make a measurable error space and then the error $\boldsymbol{e}$ is backpropagated through the transpose of the Jacobian to update the weights as proven in Subsection 4.2. The homogeneous transformation matrices of the simulated manipulator are used to obtain the actual end-effector position vector $\boldsymbol{p} \in \Re^m$ in m-dimensional Cartesian coordinate, while the exploring sensor is used to feedback both the desired Cartesian vector of the end-effector position $\boldsymbol{p}_d = [x_d \quad y_d]^T$ each time step along each segment of the planned trajectory and the corresponding ϕ_i angle.

The fuzzy system is designed based on controlling ψ_j angles within their permissible limits around the planned trajectory to generate the hint vector $\boldsymbol{\theta}_h$. Unfortunately, this fuzzy system can generate values only for each link being inside the enclosed space; therefore, we substitute for the rest of the joint angle values by their previous values then train the NN to obtain the correct values.

Concisely, whenever the explorer sensor plans different trajectory, the fuzzy system accordingly online generates different hint vectors to the NN to give the appropriate solution.

Moreover, from the mathematical point of view, the problem can be visualized as a solution of the following forward kinematics problem:

$$\boldsymbol{p} = \boldsymbol{f}(\boldsymbol{\theta}), \tag{5}$$

where $\boldsymbol{f}(\cdot): \Re^n \mapsto \Re^m$. Linearizing this equation by considering small displacement about the current configuration, we obtain

$$\Delta\boldsymbol{p} = J\Delta\boldsymbol{\theta}, \tag{6}$$

where $J = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\theta}}$ is the $m \times n$ Jacobian matrix. For redundant manipulator, $m < n$ and J is of full rank, so that the system has infinite number of solutions.

The general solution of equation (6) is given by

$$\Delta\theta = J^{+}\Delta p + (I - J^{+}J)y, \quad (7)$$

where y is an arbitrary vector, J^{+} is the pesudoinverse of the Jacobian matrix and I is the $n \times n$ identity matrix.

If the utilization of redundancy will not be considered, the solution can be given by the first term only as follows:

$$\Delta\theta = J^{+}\Delta p, \quad (8)$$

which is the least square solution with minimum norm for the joint angle. This solution can be replaced by an NN with Δp as input vector and $\Delta\theta$ as output vector.

On the other hand, if the utilization of redundancy will be considered, the solution can be given by equation (7) where the second term represents the set of all vectors in the null space of the Jacobian, i.e., does not produce any change in the end-effector. This equation has two input vectors Δp and y for sharing the output vector $\Delta\theta$. Therefore, an NN with two input vectors and one output vector can replace this equation. The first input vector is the differential change of the Cartesian coordinate values of the end-effector while the second, the arbitrary one can be chosen to represent the desirable joint angle vector to avoid the collision with the enclosed space; it can be obtained from the fuzzy system and presented on-line as a hint to the NN to search for the suitable solution.

3.2. Illustrative example

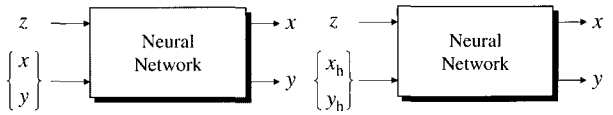
In a non-unique problem that has many solutions for an input, this example is introduced to illustrate the concept of guiding the output of an NN, feeding the hint values for the solution to the NN as an additional input vector rather than the training data.

Consider the following equation,

$$z = x^2 + y^2. \quad (9)$$

The inverse problem of this equation has many solutions, i.e., many pairs (x, y) for a certain value of z. Therefore, in constructing an NN to generate this solution, two input vectors will be presented. The first vector is $i_1 = [z]^T$ while the second is $i_2 = [x \quad y]^T$ which is considered to be the hints in the training mode.

A three-layered resilient backpropagation NN [19] with 45 neurons in each hidden layer was constructed as shown in Fig. 4, where the input vector is $i = [i_1^T \quad i_2^T]^T = [z \quad x_h \quad y_h]^T$ while the output vector is $o = [x \quad y]^T$ in the testing mode. Two numerical cases, each has the same value for z in its input vector but different values for both x_h and y_h,



(a) Training mode. (b) Testing mode.

Fig. 4. NN represents a relation between hint input and correct output.

were obtained as follows:

Case 1:

$i = [31.25 \quad 5 \quad 2]^T \quad o = [5.2031 \quad 2.0727]^T$

Case 2:

$i = [31.25 \quad -5 \quad -2]^T \quad o = [-5.1696 \quad -2.1231]^T$

It is found that, for the same input and different hints, the NN responds with appropriate outputs in the sequel.

4. CONSTRUCTION OF PROPOSED CONTROL SYSTEM

4.1. Fuzzy systems

A zero-order Sugeno fuzzy model system [20], Fuzzy Model (1) in Fig. 5, is designed to generate the collision avoidance criterion ψ_j angles within their permissible limits ($\psi_j \in [0, \psi_{j\max}]$) for each ϕ_i angle, where $j = n, n-1, \cdots, r+1$ and r is the number of links outside the enclosed space. It has the following form:

$$R_k : \text{If } q \text{ is } A_k \text{ then } z_{nk} = C_{nk} \text{ and} \cdots \text{ and } z_{[r+1]k} = C_{[r+1]k},$$

where k is the index of the rule, $k = 1, 2, \cdots, 7$; C_{nk} and $C_{[r+1]k}$ are constants; q, z_{nk} and $z_{[r+1]k}$ are considered to be ϕ_i, ψ_n and ψ_{r+1} angles respectively.

Seven Gaussian membership functions are constructed for ϕ_i angle in its universe ($\phi_i \in [-120, 120]$) with the following form

$$\mu_k(q) = e^{-\left(\frac{q - c_k}{\sigma_k}\right)^2}, \quad (10)$$

where c_k and σ_k are the center and width parameters of the membership function respectively.

Seven rules are designed for this system as shown in Table 1, where the firing strength of every rule is $h_k = \mu_k$. Here, NB, NM, NS, ZE, PS, PM and PB denote the linguistic labels, negative big, negative medium, negative small, zero, positive small, positive medium and positive big respectively. The consequent

Table 1. Rules for fuzzy model (1).

Rule no.	IF	THEN
1	ϕ_i is PS	ψ_n is PS and ⋯ and ψ_{r+1} is PS
2	ϕ_i is NS	ψ_n is PS and ⋯ and ψ_{r+1} is PS
3	ϕ_i is PM	ψ_n is PM and ⋯ and ψ_{r+1} is PM
4	ϕ_i is NM	ψ_n is PM and ⋯ and ψ_{r+1} is PM
5	ϕ_i is PB	ψ_n is PB and ⋯ and ψ_{r+1} is PB
6	ϕ_i is NB	ψ_n is PB and ⋯ and ψ_{r+1} is PB
7	ϕ_i is ZE	ψ_n is PS and ⋯ and ψ_{r+1} is PS

part of the rules has a crisp value and the overall outputs are taken as a weighted average of each rule's output:

$$z_j = \frac{\sum_k h_k z_{jk}}{\sum_k h_k}; \qquad j = n, \cdots, r+1. \tag{11}$$

To generate the hint values for the joint angle of each link being inside the enclosed space, $n-r$ fuzzy models (2) to (4) are designed as shown in Fig. 5, one for each angle as a Sugeno fuzzy model. Each has the following form:

$$R_k : \text{If } q_1 \text{ is } A_k \text{ and } q_2 \text{ is } B_k \text{ then } z_k = z(q_1, q_2),$$

where k is the index of the rules for each fuzzy system, $k = 1, 2, \cdots, 7$; and q_1, q_2 and z_k are considered for ϕ_i, ψ_j and θ_{hj} respectively, for $j = n, n-1, \cdots, r+1$. It is worthy to mention that, for each θ_{hj}, the values of both ϕ_i and the corresponding ψ_j that are generated from the fuzzy model (1), are chosen according to the relative position of joint $j+1$ coordinates, $\boldsymbol{p}_{j+1}$, with respect to the planned trajectory; except for $j = n$ since the generation of θ_{hn} depends on the position of the end-effector $\boldsymbol{p}_d$. Specifically, to generate θ_{hj}, the segment number i of the planned trajectory at which the intersection with link j will occur, must be determined first. For example, in case of three-link manipulator, ϕ_1 and the corresponding ψ_3 are used to generate θ_{h3} for link 3 while the end-effector is tracking the first segment of the trajectory. After tracking the first segment, the end-effector is going to track the second segment of the trajectory; so, ϕ_2 and the corresponding ψ_3 are used to generate θ_{h3} for that link. To generate θ_{h2} for link 2, we should determine at which segment the intersection with that link will occur; it can be done by using the relative position of joint 3, $\boldsymbol{p}_3$ with respect to the planned trajectory. For instance, let this link intersect with the first segment; so, ϕ_1 and the corresponding ψ_2 are used to generate θ_{h2}.

The design of each fuzzy system is as follows: Gaussian membership functions are assigned for the processed sensor data ϕ_i angle in its universe and for each collision avoidance criterion ψ_j angle in its universe ($\psi_j \in [0, \psi_{j\max}]$) with

$$\mu_{kl}(q_l) = e^{-\left(\frac{q_l - c_{kl}}{\sigma_{kl}}\right)^2}, \tag{12}$$

where l is the index of the input $l = 1$, 2, and c_{kl} and σ_{kl} are the center and width parameters of the membership function respectively. Here, seven and

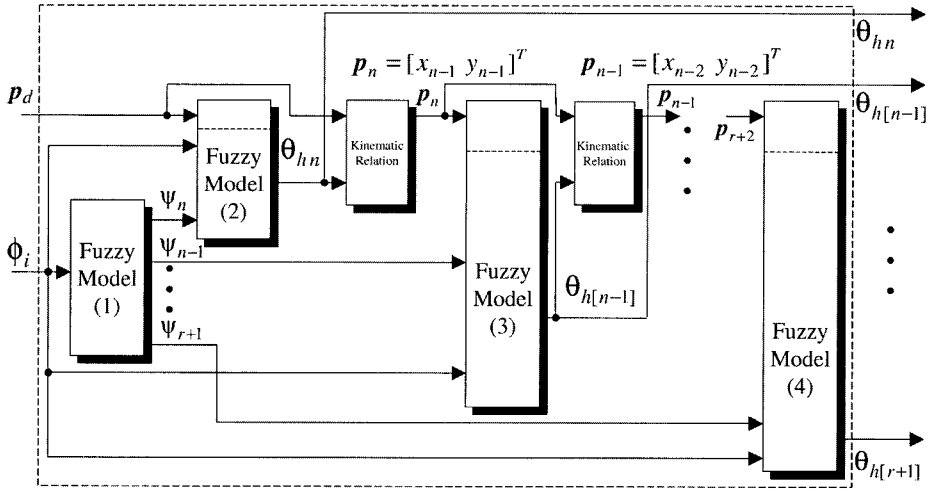


Fig. 5. Construction of the fuzzy systems.

Table 2. Rules for fuzzy model (2).

Rule no.	IF	THEN
1	ϕ_i is ZE and ψ_3 is PS	$\theta_{h3} = -\psi_3$
IF $0 < x' \le 1/2$		
2	ϕ_i is PS and ψ_3 is PS	$\theta_{h3} = f_1(\phi_i, \psi_3, x')$
3	ϕ_i is NS and ψ_3 is PS	$\theta_{h3} = f_2(\phi_i, \psi_3, x')$
4	ϕ_i is PM and ψ_3 is PM	$\theta_{h3} = f_1(\phi_i, \psi_3, x')$
5	ϕ_i is NM and ψ_3 is NM	$\theta_{h3} = f_2(\phi_i, \psi_3, x')$
6	ϕ_i is PB and ψ_3 is PB	$\theta_{h3} = f_1(\phi_i, \psi_3, x')$
7	ϕ_i is NB and ψ_3 is PB	$\theta_{h3} = f_2(\phi_i, \psi_3, x')$
IF $1/2 < x' \le 1$		
2	ϕ_i is PS and ψ_3 is PS	$\theta_{h3} = f'_1(\phi_i, \psi_3, x')$
3	ϕ_i is NS and ψ_3 is PS	$\theta_{h3} = f'_2(\phi_i, \psi_3, x')$
4	ϕ_i is PM and ψ_3 is PM	$\theta_{h3} = f'_1(\phi_i, \psi_3, x')$
5	ϕ_i is NM and ψ_3 is PM	$\theta_{h3} = f'_2(\phi_i, \psi_3, x')$
6	ϕ_i is PB and ψ_3 is PB	$\theta_{h3} = f'_1(\phi_i, \psi_3, x')$
7	ϕ_i is NB and ψ_3 is PB	$\theta_{h3} = f'_2(\phi_i, \psi_3, x')$

Table 3. Rules for fuzzy model (3).

Rule no.	IF	THEN
1	ϕ_i is ZE and ψ_2 is PS	$\theta_{h2} = \psi_2$
IF y_2 is above the planned trajectory		
2	ϕ_i is PS and ψ_2 is PS	$\theta_{h2} = f_3(\psi_2, x')$
3	ϕ_i is NS and ψ_2 is PS	$\theta_{h2} = \psi_2$
4	ϕ_i is PM and ψ_2 is PM	$\theta_{h2} = f_3(\psi_2, x')$
5	ϕ_i is NM and ψ_2 is PM	$\theta_{h2} = \psi_2$
6	ϕ_i is PB and ψ_2 is PB	$\theta_{h2} = f_3(\psi_2, x')$
7	ϕ_i is NB and ψ_2 is PB	$\theta_{h2} = \psi_2$
IF y_2 is under the planned trajectory		
2	ϕ_i is PS and ψ_2 is PS	$\theta_{h2} = f'_3(\psi_2, x')$
3	ϕ_i is NS and ψ_2 is PS	$\theta_{h2} = -\psi_2$
4	ϕ_i is PM and ψ_2 is PM	$\theta_{h2} = f'_3(\psi_2, x')$
5	ϕ_i is NM and ψ_2 is PM	$\theta_{h2} = -\psi_2$
6	ϕ_i is PB and ψ_2 is PB	$\theta_{h2} = f'_3(\psi_2, x')$
7	ϕ_i is NB and ψ_2 is PB	$\theta_{h2} = -\psi_2$

three membership functions are used for the first and second inputs.

Seven rules are designed for each fuzzy system. The algebraic product is considered for the T-norm of the antecedent part of the rule; therefore the firing strength of every rule is given by

$$h_k = \mu_{k1}\mu_{k2}. \tag{13}$$

The consequent part of the rules consists of a crisp function z_k. The overall output of each system is taken as the weighted average of each rule's output:

$$z = \frac{\sum_k h_k z_k}{\sum_k h_k}. \tag{14}$$

A method for constructing the antecedent part of the fuzzy rules for the $n-r$ models is based on both smoothly changing the hint value of each joint variable by using trigonometric functions $Q(x')$ and skillfully negotiating the link's way around the planned trajectory by using ψ_j angles in positive and negative directions. According to this concept, the general form of the z_k is as follows:

$$z_k = \phi_i \times Q(x') \pm \psi_j \times Q(x'), \tag{15}$$

where x' is the ratio of displacement of the end-effector along each trajectory segment to the trajectory segment length t_s.

A construction example of the fuzzy rules for fuzzy models (2) and (3) is illustrated in Tables 2 and 3, where $n=3$ and $r=1$ and the consequent functions are designed as follows:

$$f_1(\phi_i, \psi_3, x') = \phi_i \sin(\pi x') - \psi_3, \tag{16}$$

$$f'_1(\phi_i, \psi_3, x') = \phi_i - \psi_3 \cos(\pi(2x'-1)), \tag{17}$$

$$f_2(\phi_i, \psi_3, x') = \phi_i \tan\left(\frac{\pi}{2}x'\right) - \psi_3 \cos(2\pi x'), \tag{18}$$

$$f'_2(\phi_i, \psi_3, x') = \phi_i + \psi_3 \cos(\pi(2x'-1)), \tag{19}$$

$$f_3(\psi_2, x') = \psi_2 \cos\left(\frac{3}{2}\pi x'\right), \tag{20}$$

$$f'_3(\psi_2, x') = -\psi_2 \sin\left(\frac{\pi}{2}x'\right). \tag{21}$$

The rules of the fuzzy model 3, as shown in Table 3 is based mainly on the relative position of the joint 3 coordinates, $\boldsymbol{p}_3 = [x_2 \quad y_2]^T$, with respect to the planned trajectory. The position of joint $n-1$ coordinates, $\boldsymbol{p}_{n-1}$ can be calculated from the position of joint n, $\boldsymbol{p}_n = [x_{n-1} \quad y_{n-1}]^T$, by using simple kinematics relation.

4.2. The back propagation neural network

It has been reported in [21] that an NN with two-hidden layers enables the learning process to be more successful because of its ability to extract the local features and the global features from the input space by the first and the second hidden layers respectively, particularly when the size of the input layer is large. Also, because of the nonlinear mapping and

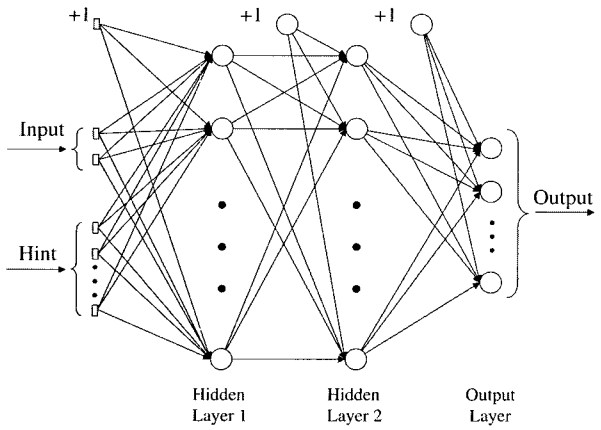


Fig. 6. The architecture of the neural network.

adaptation capability of such NNs, a back propagation NN with two hidden layers as shown in Fig. 6 is used for the inverse kinematics problem. Two input vectors are presented to that NN. The first is considered for the differential change of the Cartesian variable vector, $\Delta \boldsymbol{p} = [\Delta x \quad \Delta y]^T$ while the second is considered for the hint values, $\Delta \boldsymbol{\theta}_h = [\Delta\theta_{h1} \quad \Delta\theta_{h2} \quad \cdots \quad \Delta\theta_{hn}]^T$ where $\Delta\theta_{hi} = \theta_{hi} - \theta_i$ for $i = 1, \cdots, n$. The output vector is considered for the differential change of the joint variable $\Delta \boldsymbol{\theta} = [\Delta\theta_1 \quad \Delta\theta_2 \quad \cdots \quad \Delta\theta_n]^T$.

The forward calculation through the NN to compute the output based on the matrix calculus approach is as follows:

Letting X represent the input vector $X = \left[1 \quad \Delta \boldsymbol{p}^T \quad \Delta \boldsymbol{\theta}_h^T\right]^T$, the output of the first and second hidden layers and of the output layer are as follows:

$$\tilde{Z}_1 = \boldsymbol{\sigma}(U^T X), \tag{22}$$

$$\tilde{Z}_2 = \boldsymbol{\sigma}(V^T Z_1), \tag{23}$$

$$Z_3 = \boldsymbol{\sigma}(W^T Z_2), \tag{24}$$

where for example $\boldsymbol{\sigma}(U^T X) = [\sigma(\{U^T X\}_1), \ldots, \sigma(\{U^T X\}_{l_1})]^T$, in which $\sigma(\cdot)$ is the hyperbolic tangent activation function $\sigma(x) = \frac{2}{1+e^{-2x}} - 1$. U, V and W consist of connection weights and thresholds matrices for the first and second hidden layers and the output layer, respectively, which are given by

$$U = \left[U_0^T \quad \tilde{U}^T\right]^T, \tag{25}$$

$$V = \left[V_0^T \quad \tilde{V}^T\right]^T, \tag{26}$$

$$W = \left[W_0^T \quad \tilde{W}^T\right]^T, \tag{27}$$

where U_0, V_0 and W_0 are the threshold vectors for the first, second and output layers given by

$$U_0 = \left[U_{01} \quad U_{02} \quad \cdots \quad U_{0l_1}\right],$$

$$V_0 = \left[V_{01} \quad V_{02} \quad \cdots \quad V_{0l_2}\right],$$

$$W_0 = \left[W_{01} \quad W_{02} \quad \cdots \quad W_{0m_2}\right],$$

and

$$\tilde{U} = \left[U_{ij}\right] \quad \text{for } i = 1, 2, \cdots, m_1,\ j = 1, 2, \cdots, l_1,$$

$$\tilde{V} = \left[V_{ij}\right] \quad \text{for } i = 1, 2, \cdots, l_1,\ j = 1, 2, \cdots, l_2,$$

$$\tilde{W} = \left[W_{ij}\right] \quad \text{for } i = 1, 2, \cdots, l_2,\ j = 1, 2, \cdots, m_2,$$

in which l_1, l_2 and m_2 are the numbers of the first and second hidden units and of the output units respectively; m_1 is the number of the input units; and

$$Z_1 = \left[1 \quad \tilde{Z}_1^T\right]^T, \tag{28}$$

$$Z_2 = \left[1 \quad \tilde{Z}_2^T\right]^T. \tag{29}$$

Since the reference output (i.e., the desired differential change of the joint variable) is not known at the output layer, the output vector must be mapped to make a measurable error space, which is the differential change of the Cartesian variable. Thus, the output vector Z_3 is mapped through the Jacobian matrix J to

$$Y = JZ_3. \tag{30}$$

Letting Y' represent the desired differential change of the Cartesian variable vector, the error vector at iteration k can be calculated as $\boldsymbol{e}(k) = Y' - Y(k)$. Then, the least squares cost of the output error is given by

$$E(k) = \frac{1}{2}\mathrm{Tr}\left\{\boldsymbol{e}(k)\boldsymbol{e}^T(k)\right\}, \tag{31}$$

where $\mathrm{Tr}\{\cdot\}$ is the trace operator. Assuming linear activation function at the output layer, the cost function can be rewritten as

$$E(k) = \frac{1}{2}\mathrm{Tr}\left\{\left(Y' - JW^T(k)Z_2\right)\left(Y' - JW^T(k)Z_2\right)^T\right\}. \tag{32}$$

According to the generalized delta rule [21], the updating algorithm of connection weights is given by

$$W(k+1) = W(k) + \alpha\Delta W(k-1) - \eta\frac{\partial E(k)}{\partial W(k)}, \tag{33}$$

where α is the momentum factor, η is the learning

rate and $\partial E(k)/\partial W(k)$ can be calculated as follows:

$$\frac{\partial E(k)}{\partial W(k)} = \frac{1}{2}\frac{\partial}{\partial W(k)}\mathrm{Tr}\left\{Y'Y'^T - 2\left(Y'Z_2^T\right)W(k)J^T + W(k)\left(J^TJ\right)W^T(k)\left(Z_2Z_2^T\right)\right\}. \quad (34)$$

Using the following Trace identities

$$\frac{\partial \mathrm{Tr}\{BAC\}}{\partial A} = B^TC^T,$$

$$\frac{\partial \mathrm{Tr}\{ABA^TC\}}{\partial A} = C^TAB^T + CAB,$$

we obtain:

$$\begin{aligned}\frac{\partial E(k)}{\partial W(k)} &= -\left(Z_2Y'^T\right)J + \left(Z_2Z_2^T\right)W(k)\left(J^TJ\right)\\ &= -Z_2\left(Y'^T - Z_2^TW(k)J^T\right)J\\ &= -Z_2\boldsymbol{e}^T(k)J\\ &= -Z_2\boldsymbol{\delta}_3^T(k),\end{aligned} \quad (35)$$

where $\boldsymbol{\delta}_3(k) = J^T\boldsymbol{e}(k)$ is the $m_2 \times 1$ local gradient vector of the output layer. The local gradient vectors of the second and first layers are as the original algorithms in [22], which are given by

$$\boldsymbol{\delta}_2(k) = (I + \mathrm{diag}(\tilde{Z}_2))(I - \mathrm{diag}(\tilde{Z}_2))\tilde{W}(k)\boldsymbol{\delta}_3(k), \quad (36)$$

$$\boldsymbol{\delta}_1(k) = (I + \mathrm{diag}(\tilde{Z}_1))(I - \mathrm{diag}(\tilde{Z}_1))\tilde{V}(k)\boldsymbol{\delta}_2(k), \quad (37)$$

where $\mathrm{diag}(X)$ is the diagonal matrix having the elements of the vector on the diagonal. Therefore, the updating algorithms of the weights and thresholds are reduced to

$$W(k+1) = W(k) + \alpha\Delta W(k-1) + \eta Z_2\boldsymbol{\delta}_3^T(k), \quad (38)$$

$$V(k+1) = V(k) + \alpha\Delta V(k-1) + \eta Z_1\boldsymbol{\delta}_2^T(k), \quad (39)$$

$$U(k+1) = U(k) + \alpha\Delta U(k-1) + \eta X\boldsymbol{\delta}_1^T(k). \quad (40)$$

5. SIMULATION RESULTS

Simulations have been performed mainly based on three-link planar manipulator with three revolute joints. However, another simulation has been also conducted based on four-link planar manipulator with two degrees-of-redundancy to study the effectiveness of the proposed control system for manipulators with multiple degrees-of-redundancy. Each link has a length of 1 [m]. The initial configuration of the three-link manipulator is set to $\theta = [165^\circ \ -153^\circ \ -11^\circ]^T$, while that for the four-link manipulator is set to $\theta = [120^\circ \ -172^\circ \ 105^\circ \ -81^\circ]^T$. The simulation results are presented to demonstrate the performance of the proposed position control system for redundant manipulator to track the planned trajectory inside the unknown enclosed space. The objective is to simulate the redundant manipulator while going inside an enclosed space specified as a bended tube of a diameter of 1 [m]. Each portion of the tube is simulated by two straight lines representing its boundary $(y_{bj} = m_{bj}x_b + c_{bj})$ while the rotating sensor is simulated by a line rotating an angle γ_{zi} around Z_e-axis. At each rotational step of γ_{zi} angle, the length of this rotating line t_s that is a function of γ_{zi} can be calculated from the intersection of this rotating line with each line representing the tube boundaries. That is, the absolute position of any point s along this rotating line at any step of γ_{zi} can be calculated from the homogeneous transformation matrix ${}^s_0T$, which represents the absolute position and orientation of this line, given by

$${}^s_0T = {}^n_0T\ {}^s_nT, \quad (41)$$

where

$${}^n_0T = \prod_{k=1}^{n} {}^k_{k-1}T$$

describes the absolute position and orientation of the link n, and ${}^k_{k-1}T$ is the 4×4 homogeneous transformation matrix relating link $k-1$ coordinates and link k coordinates according to the D-H representation, which has the following form:

$${}^k_{k-1}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ {}^k_{k-1}\boldsymbol{p} & & {}^k_{k-1}R & \end{bmatrix},$$

where

$${}^k_{k-1}\boldsymbol{p} = \begin{bmatrix} a_k\cos\theta_k \\ a_k\sin\theta_k \\ \rho_k \end{bmatrix},$$

$${}^k_{k-1}R = \begin{bmatrix} \cos\theta_k & -\sin\theta_k\cos\beta_k & \sin\theta_k\sin\beta_k \\ \sin\theta_k & \cos\theta_k\cos\beta_k & -\cos\theta_k\sin\beta_k \\ 0 & \sin\beta_k & \cos\beta_k \end{bmatrix}.$$

Here, β_k is the twist angle between Z_k and Z_{k-1} axes in a plane perpendicular to a_k and ρ_k is the link offset between X_k and X_{k-1} axes along the Z_{k-1} axis. Using the above homogeneous transformation matrix, ${}^s_nT$ can be obtained by substituting t_s for a_k, and setting $\beta_k = 0$, $\theta_k = \gamma_{zi}$

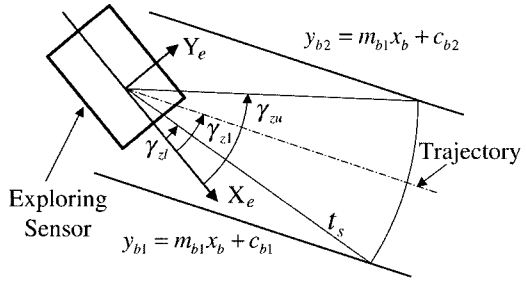


Fig. 7. Top-view of exploring sensor in simulation, which represents a situation just after going into an entrance.

and $\rho_k = 0$, as follows:

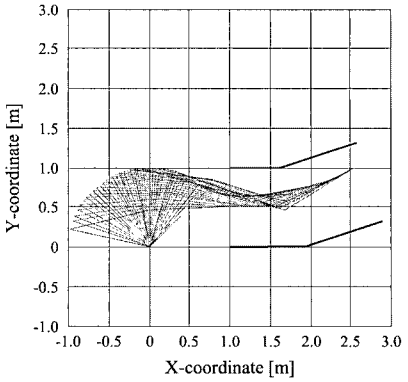
$$ {}^{s}_{n}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ t_s \cos\gamma_{zi} & \cos\gamma_{zi} & -\sin\gamma_{zi} & 0 \\ t_s \sin\gamma_{zi} & \sin\gamma_{zi} & \cos\gamma_{zi} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

Expanding the right-hand side of equation (41) to obtain the elements ${}^{s}_{0}T_{21}$ and ${}^{s}_{0}T_{31}$, substituting them for both $x_b \equiv {}^{s}_{0}T_{21}$ and $y_{bj} \equiv {}^{s}_{0}T_{31}$ in each equation representing the tube boundaries respectively, and arranging it, it follows that

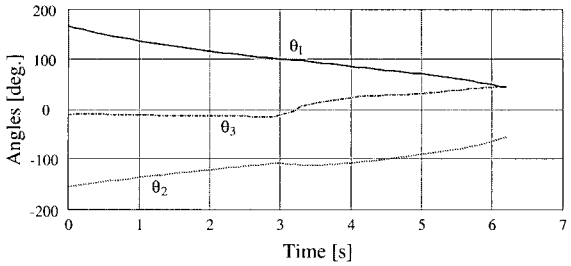
$$ t_s = \frac{m_{bj}\,{}^{n}_{0}T_{21} - {}^{n}_{0}T_{31} + c_{bj}}{A - m_{bj}B}, \qquad (42) $$

where $A = {}^{n}_{0}T_{32}\cos\gamma_{zi} + {}^{n}_{0}T_{33}\sin\gamma_{zi}$, $B = {}^{n}_{0}T_{22}\cos\gamma_{zi} + {}^{n}_{0}T_{23}\sin\gamma_{zi}$ and ${}^{n}_{0}T_{ij}$ are the elements of the homogeneous transformation matrix ${}^{n}_{0}T$.

The scenario of the simulation is as follows: record the two angles γ_{zu} and γ_{zl} which dominate the length t_s which is equal to the maximum range of the sensor, as shown in Fig. 7. Thus, γ_{z1} can be calculated as $\gamma_{z1} = \gamma_{zl} + \frac{\gamma_{zu} - \gamma_{zl}}{2}$. Therefore, the exploring sensor plans the trajectory by using γ_{z1} for equation (1), while $\gamma_{y1} = 0$ for planar application; then calculates ψ_e angle by using equation (4), at the entrance to modify the manipulator configuration towards the most dexterous configuration before going inside by using the hint values at that point; after that tracks the planned trajectory. Whenever it plans different segment of the trajectory i.e. calculates new value for ϕ_i angle using equation (2), the fuzzy system online generates different hints to avoid the collision while tracking the trajectory. That requires online training for the NN,

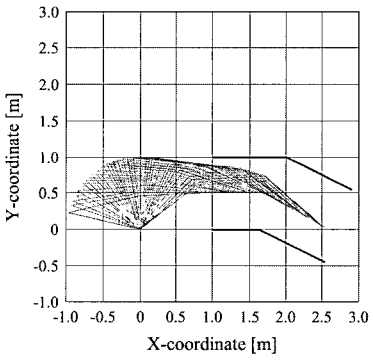


(a) Manipulator configurations.

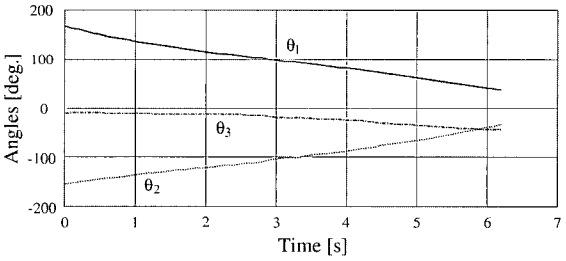


(b) Joint angle profiles.

Fig. 8. Simulation result for the enclosed space I.



(a) Manipulator configurations.

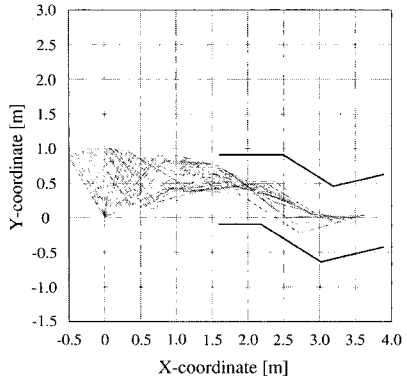


(b) Joint angle profiles.

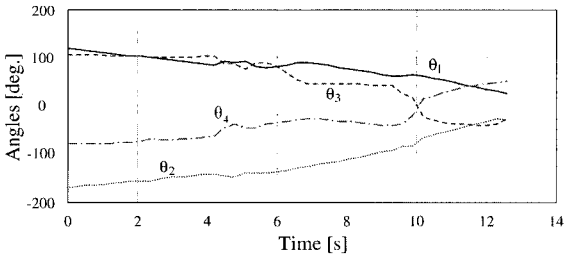
Fig. 9. Simulation result for the enclosed space II.

which can be achieved by using the error at the Cartesian space to update the weights.

A two-hidden layered NN with 30 neurons per each is used in simulations. For each simulation, the initial values for the weights are selected as random values



(a) Manipulator configurations.



(b) Joint angle profiles.

Fig. 10. Simulation result for the enclosed space III.

on $[-1,1]$. The learning rate η and the momentum factor α are chosen to be 0.1 and 0.9 respectively for the best results.

Figs. 8 and 9 represent the simulation results for different cases I and II of the enclosed space for the three-link case, while Fig. 10 represents the results for the four-link case. The manipulator successive configurations throughout the planned trajectory are represented in Figs. 8(a), 9(a) and 10(a) while the joint angle profiles are represented in Figs. 8(b), 9(b) and 10(b). The results demonstrate the efficacy of the proposed control system to track the trajectory while avoiding the collision with the environment. This performance is due to the ability gained by the NN to follow the hint generated from the fuzzy system. Neither discontinuity nor highly abrupt change is appeared in the joint angle profiles even in the case of two degrees-of-redundancy manipulator, which verifies good performance of the proposed control system.

6. DISCUSSION

The classical Jacobian-based approach for solving the inverse kinematics problem of redundant manipulator is to require the solution to optimize a certain performance criterion; a commonly used criterion [23] is defined as:

$$g(\boldsymbol{\theta}) = (\boldsymbol{\theta} - \boldsymbol{\theta}_r)^T H (\boldsymbol{\theta} - \boldsymbol{\theta}_r), \qquad (43)$$

where $\boldsymbol{\theta}_r$ is desirable constant arm posture for avoiding collision and H is a diagonal constant matrix.

In the task under consideration in this paper, there is no certain value for $\boldsymbol{\theta}_r$ throughout the trajectory. In other words, within the whole range of each joint variable, there is sub range at which the collision may occur. Furthermore, this sub range is different from point to point along the whole trajectory because it depends on the enclosed space configuration. Therefore, the fuzzy system in the proposed approach is designed for continuous online generation of an appropriate approximate value for the joint angle vector throughout the whole trajectory. Exploiting the NN as an advanced nonlinear controller to solve the inverse kinematics problem, the approximate joint vector is fed into the NN as a second input vector. This feeding process is proven to be a proper integration method to guide the output of the NN because the inverse kinematics problem has infinite number of solutions.

Moreover, the proposed technique avoids the time consuming in measuring the distance between the obstacles and the links, as an alternative Jacobian-based approach [12,24], to avoid the collision especially in an environment with many potential collisions with obstacles, by simply controlling ψ_j angles around the planned trajectory. It also avoids other problems, related to this approach, such as bouncing the links between obstacles in case of improper choice of the critical point velocity.

7. CONCLUSIONS

The problem of online inverse kinematics solution for redundant manipulator to reach inside an unknown enclosed space while avoiding the collision with the environment has been addressed in this paper. A novel sensor based position control system for that manipulator was designed by using the back propagation NN. The principle of this control system was based on generating approximate value for the joint angles vector using fuzzy reasoning system, which was designed based on controlling the cone angle around the planned trajectory as a criterion for the collision avoidance. This vector was fed into the NN, which functioned as online inverse kinematics solution, as a second input vector to limit the searching space and guide the output of the NN. For the online trajectory planning in this paper, a laser transducer at the end-effector rotated in a two-axis gyroscopic motion was used to eliminate the need for the object modeling and distance calculation. The proposed control system can be regarded as a general method for the redundancy resolution by the NN, because it can be applied to many other problems such

as joint limits avoidance by generating an appropriate hint joint angles vector for every problem then feeding it to the NN. Simulation results verified good performance of the proposed control system.

REFERENCES

[1] S. I. Choi and B. K. Kim, "Obstacle avoidance control for redundant manipulators using collidability measure," *Robotica*, vol. 18, pp. 143-151, 2000.

[2] E. G. Gilbert and D. W. Johnson, "Distance functions and their application to robot path planning in the presence of obstacles," *IEEE J. of Robot. and Automat.*, vol. RA-1, no. 1, pp. 21-30, March 1985.

[3] A. Guez and Z. Ahmad, "Solution to the inverse kinematics problem in robotics by neural networks," *Proc. IEEE Int. Conf. on Neural Networks*, vol. II, pp. 617-624, 1988.

[4] J. A. Francisco, "Multilayer back-propagation neural network for learning the forward and inverse kinematics equations," *Proc. Int. Joint Conf. on Neural Networks*, pp. II-319-321, 1990.

[5] E. S. H. Hou and W. Utama, "An artificial neural network for redundant manipulator inverse kinematics computation," *Proc. of the SPIE- The International Society for Optical Engineering*, vol. 1607, pp. 668-677, 1992.

[6] M. Ziqiang and T. C. Hsia, "Obstacle avoidance inverse kinematics solution of redundant robots by neural networks," *Robotica*, vol. 15, pp. 3-10, Jan-Feb 1997.

[7] M. T. H. Beheshti and A. K. Tehranl, "Obstacle avoidance for kinematically redundant robots using an adaptive fuzzy logic algorithm," *Proc. the American Control Conf.*, San Diego, California, pp. 1371-1375, June 1999.

[8] T. S. Wikman and W. S. Newman, "A fast, on-line collision avoidance method for a kinematically redundant manipulator based on reflex control," *Proc. IEEE Int. Conf. Robot. and Automat.*, pp. 261-266, 1992.

[9] K. Glass, R. Colbaugh, D. Lim, and H. Seraji, "Real-time collision avoidance for redundant manipulators," *IEEE Trans. on Robot. and Automat.*, vol. 11, no. 3, pp. 448-257, June 1995.

[10] H. Seraji, M. K. Long, and T. S. Lee, "Motion control of 7-DOF arms: The configuration control approach," *IEEE Trans. on Robot. and Automat.*, vol. 9, no. 2, pp. 125-139, April 1993.

[11] W. J. Cho and D. S. Kwon, "A sensor-based obstacle avoidance for a redundant manipulator using a velocity potential function," *Proc. IEEE Int. Workshop on Robot and Human Communication*, pp. 306-310, 1996.

[12] L. Zlajpah and B. Nemec, "Kinematic control algorithms for on-line obstacle avoidance for redundant manipulators," *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Lausanne, pp. 1898-1903, 2002.

[13] V. Perdereau, C. Passi, and M. Drouin, "Real-time control of redundant robotic manipulators for mobile obstacle avoidance," *Robotics and Autonomous Systems*, vol. 41, pp. 41-59, 2002.

[14] W. S. Tang, C. M. L. Lam, and J. Wang, "Kinematic control and obstacle avoidance for redundant manipulators using a recurrent neural network," *Proc. Int. Conf. on Artificial Neural Networks*, pp. 922-929, 2001.

[15] J. Wang, Q. Hu, and D. Jiang, "A Lagrangian network for kinematic control of redundant robot manipulators," *IEEE Trans. on Neural Networks*, vol. 10, no. 1, pp. 1123-1132, 1999.

[16] Y. Zhang and J. Wang, "Obstacle avoidance for kinematically redundant manipulators using a dual neural network," *IEEE Trans. on Syst., Man and Cyber.*, vol. 34, no. 1, pp. 752-759, Feb. 2004.

[17] R. P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*, MIT Press, Cambridge, MA, 1981.

[18] Y. S. Abu-Mostafa, "A method for learning from hints," *Advances in Neural Information Processing Systems 5*, pp. 73-80, 1993.

[19] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," *Proc. IEEE Int. Conf. on Neural Networks*, San Francisco, pp. 586-591, 1993.

[20] J. S. R. Jang and C. T. Sun, "Neuro-fuzzy modeling and control," *Proc. of the IEEE*, vol. 83, no. 3, pp. 378-406, March 1995.

[21] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan, New York, 1994.

[22] F. L. Lewis, S. Jagannathan, and A. Yesidirek, *Neural Network Control of Robot Manipulators and Nonlinear Systems*, Taylor & Francis, 1999.

[23] T. Yoshikawa, "Analysis and control of robot manipulators with redundancy," in Robotics Research: *The First International Symposium* M. Brady and R. Paul, Eds., MIT Press, Cambridge, MA, pp. 735-748, 1984.

[24] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *Int. J. of Robotic Research*, vol. 4, no. 3, pp. 109-117, 1985.

Samy F. M. Assal received the B.Sc. and M.Sc. degrees in Mechanical Power Engineering from Alexandria University, Egypt, in 1989 and 1997 respectively. He is currently pursuing the Ph.D. degree at the Department of Advanced Systems Control Engineering, Saga University, Japan. His research interests include dynamics and control of nonlinear systems, intelligent control systems, neural network control, fuzzy control and their applications to redundant manipulators.

Keigo Watanabe received the B.S. and the M.S. degrees in Mechanical Engineering from University of Tokushima, Tokushima, in 1976 and 1978, respectively and D.E. degree in Aeronautical Engineering from Kyushu University in 1984. From 1980 to March in 1985, he was a research associate in Kyushu University. From 1985 to March in 1990, he was an Associate Professor in the College of Engineering, Shizuoka University. From April in 1990 to March 1993 he was an associate Professor at Saga University. From April in 1993, he is being a full professor in the Department of Mechanical Engineering, Saga University. Especially, from April 1998, he is also a full professor in the Department of Advanced Systems Control Engineering, Graduate School of Science and Engineering. He has published more than 480 technical papers in transactions, journals, and international conference proceedings, and is the author or editor of 22 books, including Adaptive Estimation and Control (Englewood Cliffs, NJ: Prentice-Hall, 1991), Stochastic Large-Scale Engineering Systems (New York: Marcel Dekker, 1992), Intelligent Control Based on Flexible Neural Networks (Norwell, MA: Kluwer, 1999) and Evolutionary Computations: New Algorithms and Their Applications to Evolutionary Robots (Heidelberg, Germany: Springer-Verlag, 2004). He is an Active Reviewer of many journals and transactions, an Editor-in-Chief of Machine Intelligence and Robotic Control, an Associate Editor of IEEE Trans. on Industrial Informatics, and an editorial board member of the Journal of Intelligent and Robotic Systems. His research interests are in stochastic adaptive estimation and control, robust control, neural network control, fuzzy control, and genetic algorithms and their applications to machine intelligence and robotic control.

Kiyotaka Izumi received a B.E. degree in Electronic Engineering from the Nagasaki Institute of Applied Science in 1991, a M.E. degree in Electrical Engineering from the Saga University in 1993, and a D.E. degree in Faculty of Engineering Systems and Technology from the Saga University in 1996. From April in 1996 to March in 2001, he was a Research Associate in the Department of Mechanical Engineering at Saga University. From April in 2001, he was with the Department of Advanced Systems Control Engineering, Graduate School of Science and Engineering, Saga University. From August in 2004, he is an Associate Professor in the Department of Advanced Systems Control Engineering, Graduate School of Science and Engineering, Saga University. His research interests are in intelligent control, fuzzy control, evolutionary computation, and their applications to the robot control.