Al- Khwarizmi Engineering Journal

Al- Khwarizmi Engineering Journal , Vol. 3 , No.1 , pp 1-11 (2007)

The Inverse Solution Of Dexterous Robot By Using Neural Networks

Dr. Bahaa Ibraheem Kazem
Mechatronics Dept Al-kwarizmi College of Engineering
University of Baghdad,, Iraq,

Samer Yahya Hadi
Control and Computers Engineering
University of Technology, Iraq

(Received 1٣ June 200٦; accepted 19 December 2006**)**

Abstract:

The inverse kinematics of redundant manipulators has infinite solutions by using conventional methods, so that, this work presents applicability of intelligent tool (artificial neural network ANN) for finding one desired solution from these solutions. The inverse analysis and trajectory planning of a three link redundant planar robot have been studied in this work using a proposed dual neural networks model (DNNM), which shows a predictable time decreasing in the training session. The effect of the number of the training sets on the DNNM output and the number of NN layers have been studied. Several trajectories have been implemented using point to point trajectory planning algorithm with DNNM and the result shows good accuracy of the end effector position for the desired trajectory.

Keywords: inverse kinematics, robotics, neural network

Introduction:

The kinematics is the science of motion, which deals with motion without consideration of the forces that cause it. Within the science of kinematics one should study the position, velocity, acceleration, and all higher order derivatives of the position variables (with respect to time or any other variable(s)). Hence, the study of the kinematics of manipulators refers to all the geometrical and time-based properties of the motion. The relationships between these motions and the forces and torques, which cause them, are the problem of dynamics [1].

There are two types of kinematics: the forward and inverse kinematics. The forward kinematics problem can be stated as follows: Given the joint variables of the robot, one can determine the position and orientation of the end effecter. The joint variables are the angles between the links in the case of revolute or rotational joints, and the link extension in the case of prismatic or sliding joints. The forward kinematics problem is to be contrasted with the inverse kinematics problem, which can be stated as follows: Given a desired position and orientation for the end-effecter of the robot, one can determine a set of joint variables that achieve the desired position and orientation [2].

Execution of a task by a robot generally requires many movements of its end effecter. The set of situations corresponding to these moves define the geometrical trajectories of the operations space. A trajectory is, therefore defined by the application and the geometrical constraints of the robot and its environment. A distinction is generally made between point-to-point trajectories and continuous path trajectories, depending on the number of geometrical constraints applied to the trajectory. Motion results from a particular parametric description of the trajectory where the parameter is time [3].

If the number of joints of the robotic arm exceeds the dimension of the tool-configuration space, then robotic arm is said to

be kinematically redundant. Robots with more than six axes are always kinematically.
redundant. However it is also possible for robots with fewer axes to be redundant if the dimension of the tool-configuration space is restricted [4].
Dexterity has been interpreted to mean different physical concepts such as a measure of kinematics feature over which a manipulator can kinematically reach all orientations, the operation of manipulators with multiple, actively powered fingers, and a global measure applied over a complete end-effecter trajectory. In one way or another, the aim of considering dexterity is to lead to robots which are similar to the human arm since many desirable features are ingeniously integrated in it and apparently constitutes the ultimate model of a dexterous robot. A dexterous device would have many sensors since a high level understanding of the situations is of the extreme importance to achieve complex tasks. It would have many degrees of freedom (DOF), may be more than a complete human arm [5].

There are many references that deal with robot kinematics and trajectory planning; some of them are mentioned below:
Robot kinematics generally includes forward and inverse kinematics at the position, velocity, and acceleration level. These constructs are essential for the cartesian control of serial manipulators. C. Kapoor and D. Tesar [6] presented the development of a C++ class library that supports the forward and inverse kinematics of all possible geometries of serial manipulators. Object-oriented analysis and design are the primary software development methodology used. Application of this methodology led to the sub-division of the kinematics domain into forward and inverse kinematics. Analysis of these sub-domains resulted in their further sub-division, identification of abstract components, development of classes, interface specifications, and finally implementation and testing.
An approach to solve the inverse kinematics problem for hyper-redundant planar manipulators following any desired path was presented by F. M. Asi [7], his approach is based on defining virtual layers and dividing them into virtual/real three-link or four-link sub-robots. This approach starts by solving for the inverse kinematics problem for the sub-robot located in the lowest virtual layer. The data obtained from the solution of this sub-robot are used to solve the inverse kinematics equations for the sub-robots located in the upper virtual layers. The data obtained by solving the equations of the sub-robots located in each virtual layer affect the solutions of the equations of the sub-robots located in both the upper and lower virtual layers.
By J. Somlo, A. Sokolov, and V. Lukanyin [8] carried out an attempt in order to develop an automatic trajectory planning. It is shown that different approaches to trajectory planning are possible. These are 1. Time optimal 2. Process optimal 3. Force effective 4.Optimal trapezoidal trajectory planning. Realization of all these trajectory-planning principles can be automatized.
S. Yue and D. Henrich [9] focused on the problem of point-to-point trajectory planning for flexible redundant robot manipulators (FRM) in joint space. Compared with irredundant flexible manipulators, a FRM possesses additional possibilities during point-to-point trajectory planning due to its kinematics redundancy. A trajectory planning method to minimize vibration and/or executing time of a point-to-point motion is presented for FRM based on Genetic Algorithms (GAs). Kinematics redundancy is integrated into the presented method as planning variables. Quadrinomial and quintic polynomial are used to describe the segments that connect the initial, intermediate, and final points in joint space. The trajectory planning of FRM is formulated as a problem of optimization with constraints.
A method for solving the inverse kinematics of a humanoid robot based on artificial neural networks is presented by J. De lope, T. Zarraonandia and D. Maravall [10]. The inputs of the network are the desired positions and

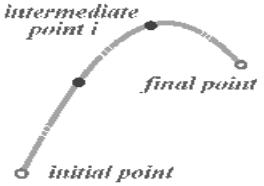


Fig.1. Intermediate point on the point-to-point path.

orientations of one foot with respect to the other foot. The output is the joint coordinates that make it possible to reach the goal configuration of the robot leg.

Trajectory planning

Trajectory refers to a time history of position, velocity and acceleration. Suppose that the point-to-point trajectory is connected by several segments with continuous acceleration at the intermediate via point (as shown in figure 1). The intermediate points can be given as particular points that should be passed through.

This is useful especially when there are obstacles in the working area. If one wishes to be able to specify the position, velocity, and acceleration at the beginning and the end of a path segment, a quadrinomial and a quintic polynomial are required. Let us assume that there are pm intermediate via points between the initial and the final points. Between the initial points to pm intermediate via points, a quadrinomial is used to describe these segments as [9];

$$x_{i,i+1}(t) = a_{i0} + a_{i1}t_i + a_{i2}t_i^2 + a_{i3}t_i^3 + a_{i4} \quad (1)$$

(i=0,…. pm-1) (2)

(ai0,…,ai4) are constants and ti is the time from the last intermediate point to the moment. The constrains are given as:

$$x_i = a_{i0} \quad (3)$$

$$x_{i+1} = a_{i0} + a_{i1}T_i + a_{i2}T_i^2 + a_{i3}T_i^3 + a_{i4}T_i^4 \quad (4)$$

$$\dot{x}_i = a_{i1} \quad (5)$$

$$\dot{x}_{i+1} = a_{i1} + 2\ a_{i2}T_i + 3\ a_{i3}T_i^2 + 4\ a_{i4}T_i^3 \quad (6)$$

$$\ddot{x}_i = 2\ a_{I2} \quad (7)$$

where Ti is the executing time from point i to point i +1. The five unknowns can be solved as:

$$a_{i0} = x_i \quad (8)$$

$$a_{i1} = \dot{x}_i \quad (9)$$

$$a_{i2} = \ddot{x}_i / 2 \quad (10)$$

$$a_{i3} = \begin{pmatrix} 4\ x_{i+1} - \dot{x}_{i+1}T_i - 4\ x_i \\ -3\ \dot{x}_i T_i - \ddot{x}_i T_i^2 \end{pmatrix} / T_i^3 \quad (11)$$

$$a_{i4} = \begin{pmatrix} \dot{x}_{i+1}T_i - 3\ x_{i+1} + 3\ x_i \\ +2\ \dot{x}_i T_i + \ddot{x}_i T_i^2 / 2 \end{pmatrix} / T_i^4 \quad (12)$$

The intermediate point i +1's acceleration can be obtained as:

$$\ddot{x}_{i+1} = 2\ a_{i2} + 6\ a_{i3}T_i + 12\ a_{i4}T_i^2 \quad (13)$$

The segment between the number pm of intermediate points and the final point can be described by quitic polynomial as:

$$x_{i,i+1}(t) = b_{i0} + b_{i1}t_i + b_{i2}t_i^2 + b_{i} \quad (14)$$
$$+ b_{i4}t_i^4 + b_{i5}t_i^5$$

where (i= pm) and the constraints are given as:

$$x_i = b_{i0} \quad (15)$$

$$x_{i+1} = b_{i0} + b_{i1}T_i + b_{i2}T_i^2 + b_{i3}T_i^3 \quad (16)$$
$$+ b_{i4}T_i^4 + b_{i5}T_i^5$$

$$\dot{x}_i = b_{i0} \quad (17)$$

$$\dot{x}_{i+1} = b_{i1} + 2\ b_{i2}T_i + 3\ b_{i3}T_i^2 + \quad (18)$$
$$4\ b_{i4}T_i^3 + 5\ b_{i5}T_i^4$$

$$\ddot{x}_i = 2\ b_{i2} \quad (19)$$

$$\dot{\theta}_{i+1} = 2\ b_{i2} + 6\ b_{i3} T_i + 12\ b_{i4} T_i^2 + 20\ b_{i5} T_i^3 \quad (20)$$

and these constraints specify a linear set of six equations with six unknowns whose solution is:

$$b_{i0} = x_i \quad (21)$$

$$b_{i1} = \dot{x}_i \quad (22)$$

$$b_{i2} = \dot{\theta}_i / 2 \quad (23)$$

$$b_{i3} = \left(\begin{array}{l} 20\ x_{i+1} - 20\ x_i - \\ \left(8\ \dot{x}_{i+1} + 12\ \dot{x}_i\right) T_i - \left(3\ \dot{\theta}_i - \dot{\theta}_{i+1}\right) T_i^2 \end{array} \right) / 2\ T_i^3 \quad (24)$$

$$b_{i4} = \left(\begin{array}{l} 30\ x_i - 30\ x_{i+1} + \\ \left(14\ \dot{x}_{i+1} + 16\ \dot{x}_i\right) T_i + \left(3\ \dot{\theta}_i - 2\ \dot{\theta}_{i+1}\right) T_i^2 \end{array} \right) / 2\ T_i^4 \quad (25)$$

$$b_{i5} = \left(\begin{array}{l} 12\ x_{i+1} - 12\ x_i \\ -\left(6\ \dot{x}_{i+1} + 6\ \dot{x}_i\right) T_i - \left(\dot{\theta}_i - \dot{\theta}_{i+1}\right) T_i^2 \end{array} \right) / 2\ T_i^5 \quad (26)$$

Using neural networks in robot kinematics

From above, it is clear that in order to perform end effecter position control of a robotic manipulator, two problems need to be solved [11]:

1. Inverse kinematics problem: Given the Cartesian coordinates of the end effecter, specified either as a single point or as a set of points on a trajectory, joint angles need to be found.
2. Target position control: Given the final end effecter position, a joint angles sequence suitable for achieving the final position needs to be found.

In this work a dual NN model has been designed in order to find the inverse solution for three degrees of freedom redundant planner robot. Each architecture of these NN exist of three layers backpropagation networks. The first network is used to find the joint angles to reach the desired point when it located in the circular work area with a maximum radius equal to (rmax/2) or (maximum reach/2). The second network is used to find the joint angles associated with remaining points in the robot work area, that means

(maximum reach/2) $\leq$ r $\leq$ (maximum reach).

Each of these two networks has three layers. There are nineteen input vectors with two elements, and there are ten neurons in the first hidden layer, twenty five neurons in the second hidden layer and three neurons in the third (output) layer. The transfer functions in the input and output layers are (purelin) and the hidden layer transfer functions are (logsig).

Case studies

A planar articulated robot with three links is used in the following case studies. The robot has one redundant freedom in terms of positioning. The first link has the length of (L1=4 unit), the second link has the length of (L3=3 unit), and the third link has the length of (L2=2 unit). All the cases are simulated in the horizontal plane.

Inverse Kinematics

If the global angle (q_3) which is equal to the summation of the three local angles (θ_1,θ_2,θ_3) takes any random value then the x and y coordinates of the end of second link can be easily found by using the conventional inverse solution[1], where

$$q_2 = \tan^{-1} \frac{r}{\pm \sqrt{1 - r^2}} \quad (27)$$

Where r can be found from:

$$r^2 = \frac{dx^2 + dy^2 - L_1^2 - L_2^2}{2\ L_1^2 L_2^2} \quad (28)$$

$$q_1 = \tan^{-1} \frac{dy}{dx} - \tan^{-1} \frac{L_2 s_2}{L_1 + L_2 c_2} \quad (29)$$

$$q_3 = \theta_1 + \theta_2 + \theta_3 \quad (30)$$

then

$\theta_3 = q_3 - \theta_1 - \theta_2$

Where dx & dy are the x and y coordinates of the end of the second link. It is clear from the previous equations that there are two solutions for any value of q_3, but there are infinite possibilities for the value of q_3, hence there are infinite solutions to the three links arm to reach any point inside the work space, therefore the neural network is used to find one desired solution.

To find the joint angles of the articulated robot in order to reach the point of (x = -2.2 unit) and (y = 4.6 unit), figure (2) shows the solution of this case.

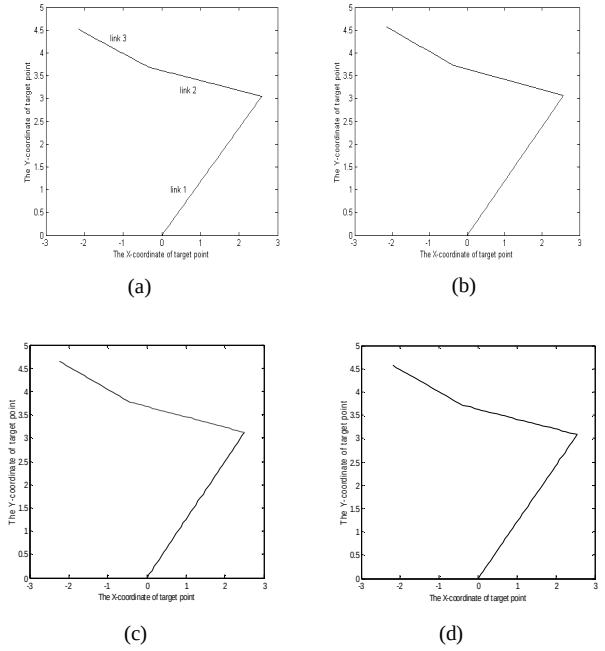


Fig. 2 .The end effecter at point (x=-2.2 unit) and (y=4.6 unit)

(a) two hidden layers N.N. and (19) training sets
(b) two hidden layers N.N. and (37) training sets
(c) single layer N.N. and (19) training sets
(d) single layer N.N. and (37) training sets

The tables (1) & (2) show the value of error by using the N.N. technique:

Where the error was found by using the following equation:

$$e = \sqrt{\left((xd - xnn)^2 + (yd - ynn)^2\right)}$$

where xd : the desired x-coordinate of target point.

xNN : the x-coordinate of the target point from the neural networks.

yd : the desired y-coordinate of target point.

yNN : the y-coordinate of the target point from the neural networks.

T**able (1**) error by using two hidden layers N.N.

no.of t.s.	θ_1(deg)	θ_2(deg)	θ_3(deg)	
19	49.6488	118.0402	-12.1608	
37	50.0103	117.2505	-12.4518	
x_d (unit)	yd(unit)	x_{NN} (unit)	y_{NN}(unit)	Error (unit)
-2.2	4.6	-2.1608	4.5180	0.0909
-2.2	4.6	-2.1653	4.5770	0.0414

Table (2) error by using single layer N.N.

no.of t.s.	θ_1(deg)	θ_2(deg)	θ_3(deg)	
19	51.2902	116.3237	-13.6137	
37	50.4623	117.4577	-13.3115	
x_d (unit)	y_d(unit)	x_{NN} (unit)	y_{NN}(unit)	Error (unit)
-2.2	4.6	-2.2263	4.6415	0.0491
-2.2	4.6	-2.1940	4.5703	0.0303

Point To Point Trajectory Planning

In first case study the training sets are (19) and the network used is a two hidden layers N.N., where the end effector is moved from the start point (x=-3, y=-5, vx=1, vy=2, ax=1, ay=0) to the target point (x=5, y=2, vx=0, vy=0, ax=0, ay=0) but the end effector is passing through the intermediate point (x=-1, y=-2, vx=2, vy=1) and the desired time for moving the end effector from the first point to the intermediate point is (2 second), and the time for moving the end effector from the intermediate point to the target point is (3 second).By using the equations of the trajectory planning, the (x) and (y) coordinates, and their first and second derivatives with time can be found.

The desired path of the end effecter from the first point to the target point passing

through the intermediate point is shown in figure (3)

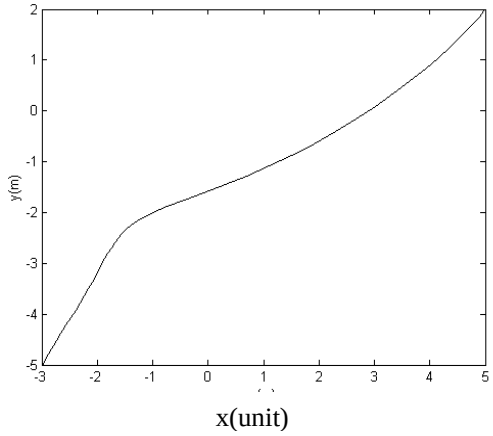


Fig. 3. The desired point-to-point path

The x & y coordinates of the desired path with time are shown in figure (4):

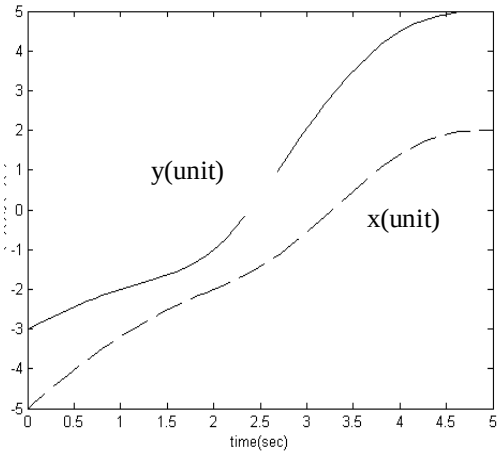


Fig. 4 .The x & y coordinates of desired path with respect to time

The first time derivative of the x & y coordinates, i.e. (vx)&(vy) with respect to time is shown in figure (5):

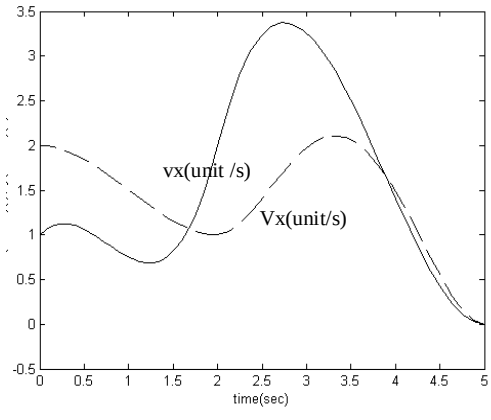


Fig. 5 .vx & vy versus time

The second time derivative of the x & y coordinates, i.e. (ax) & (ay) with respect to time is shown in figure (6):

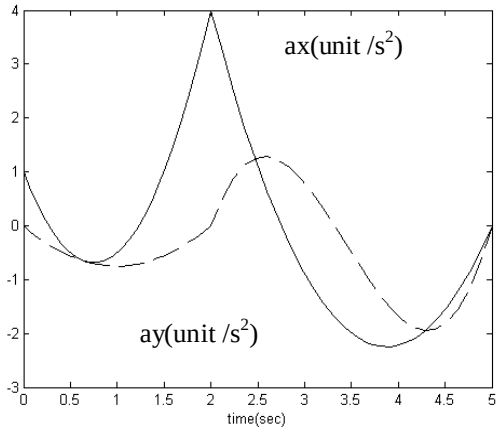


Fig. 6. ax & ay versus time

The first, second & third joints angle with respect to time are shown in figure (7):

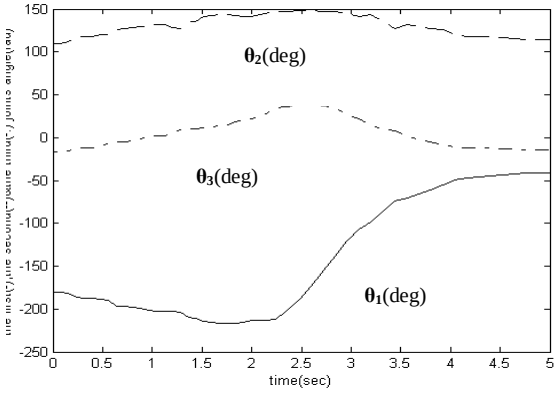


Fig. 7. θ_1, θ_2 & θ_3 versus time

Figure (8) shows both of the real and desired paths of the arm.

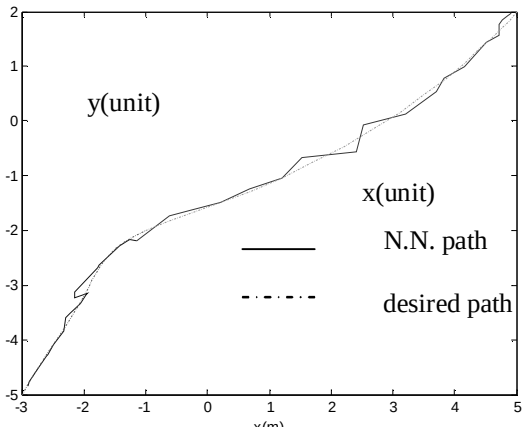


Fig. 8 . The real and desired paths of the arm

The configuration of the three links planar articulated robot is shown in figure (9).

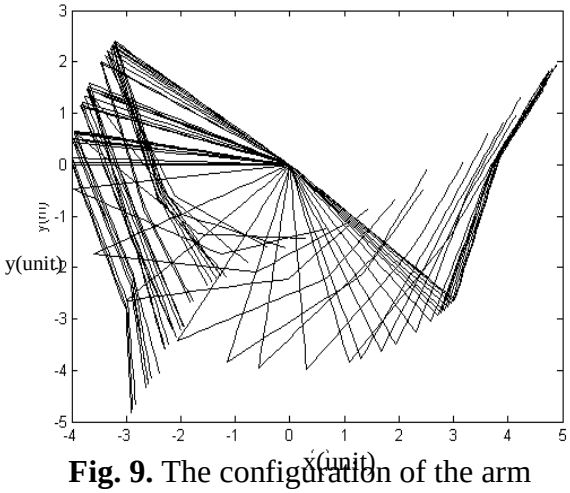


Fig. 9. The configuration of the arm

To find the error between the desired trajectory and neural network trajectory, one can calculate the error between some points along them. It is the best method to find the accuracy of the neural network method; there are forty one points along the desired trajectory and neural network trajectory which are used to find the error, where the error in each point is found by using the equation (31).Figure (10) shows the error versus the distance of the end effector from the base.

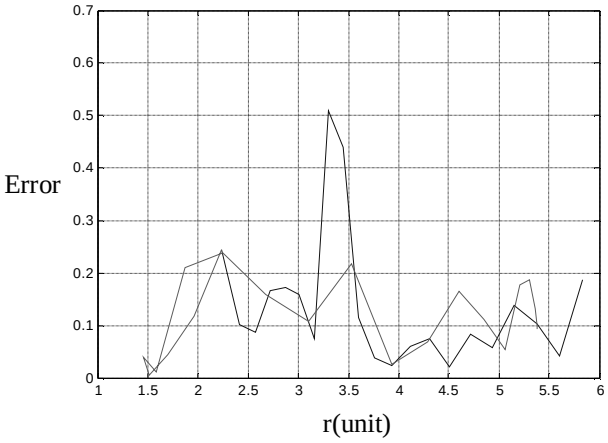


Fig.10 . The error versus the distance from the base

The second case is similar to the previous case but the training sets are (37) and the network used has a two hidden layers N.N., where the end effector is moved from the start point (x=-3, y=-5, vx=1, vy=2, ax=1, ay=0) to the target point (x=5, y=2, vx=0, vy=0, ax=0, ay=0) but the end effector is passing through the intermediate point (x=-1, y=-2, vx=2, vy=1) and the desired time for moving the end effector from the first point to the intermediate

point is (2 second), and the time for moving the end effector from the intermediate point to the target point is (3 second).
The first, second & third joints angle with respect to time are shown in figure (11):

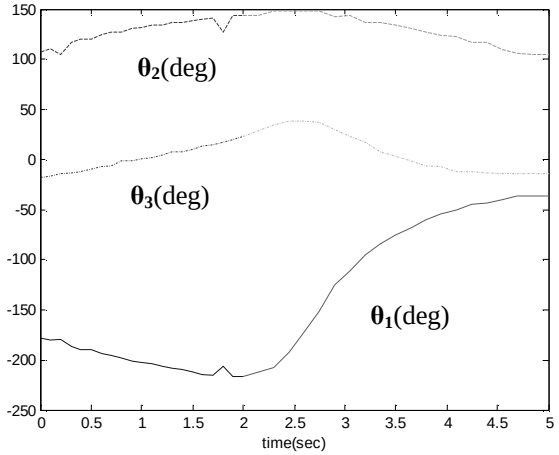


Fig. 11. θ_1, θ_2 & θ_3 versus time

Figure (12) shows both of the real and desired paths of the arm**.**

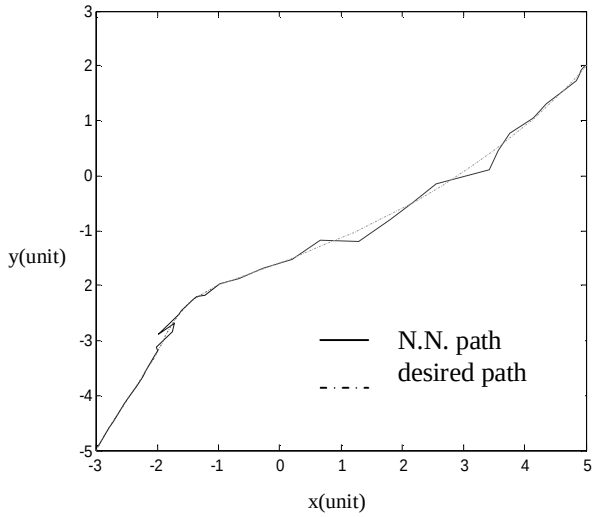


Fig. 12. The real and desired paths of the arm

The configuration of the robot is shown in figure (13); Figure (14) shows the error versus the distance of the end effector from the base.

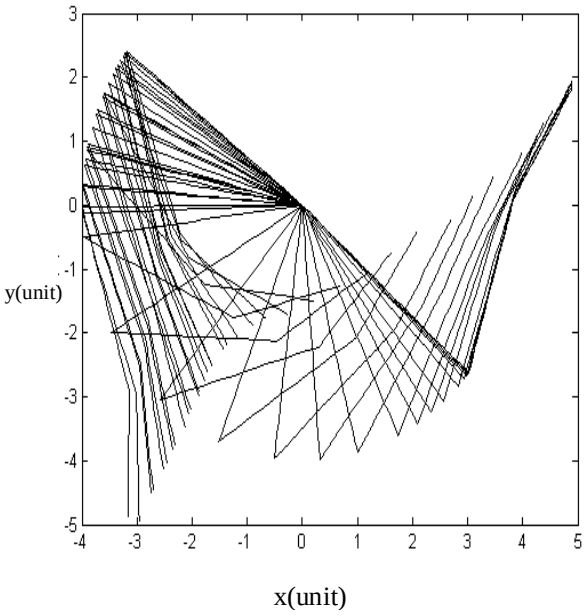


Fig. 13 .The configuration of the arm

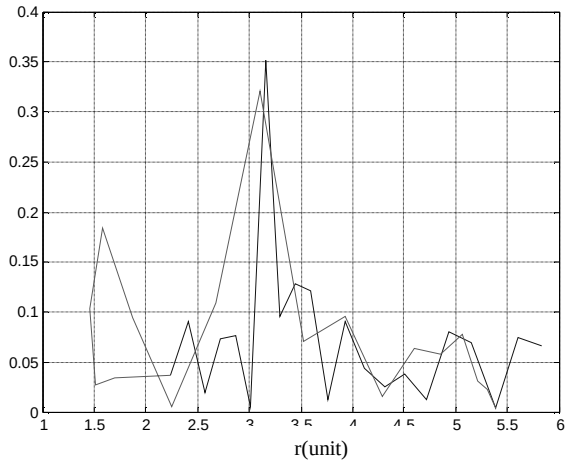


Fig 14. The error versus the distance from the base

The same case is studied by using the single layer NN instead of two hidden layers NN, when the training sets are (15) we got the results shown in the following figures. And when the training sets are (37) we got the results shown in the following figures.

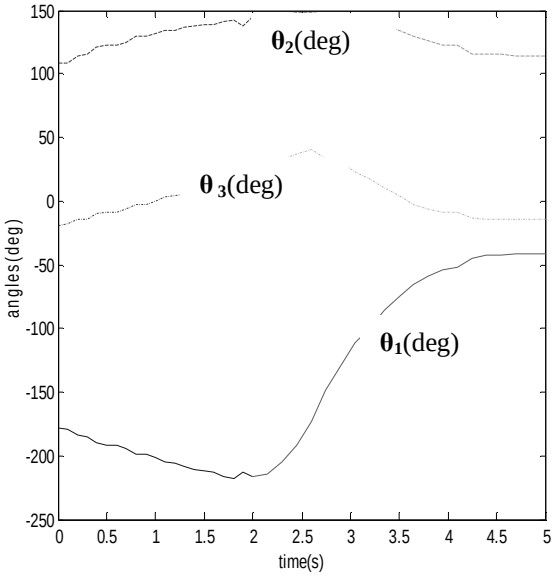


Fig.15. θ_1, θ_2 & θ_3 versus time

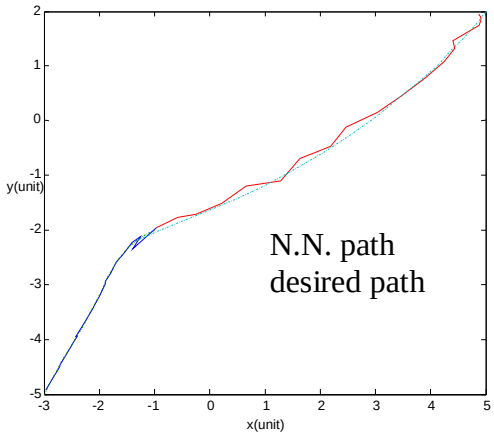
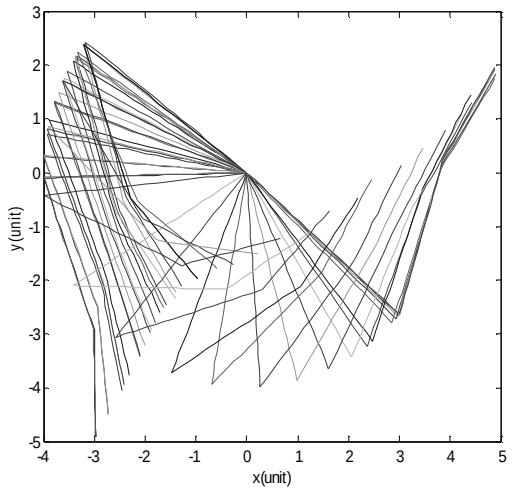


Fig. 16. The real and desired paths of the arm



F

ig. 17. The configuration of the arm

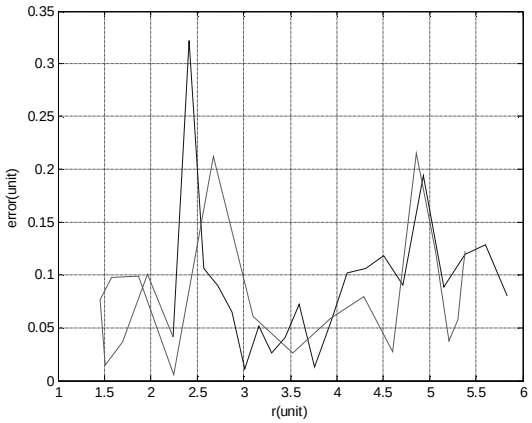


Fig. 18 .The error versus the distance from the base

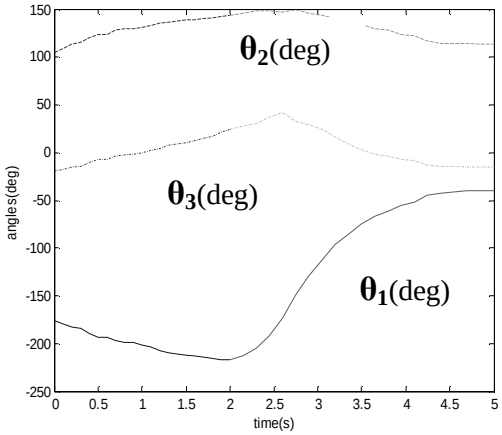


Figure 19 θ_1, θ_2 & θ_3 versus time

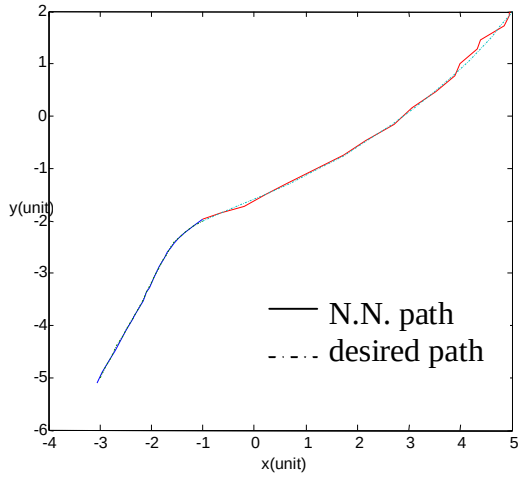


Fig.20 . The real and desired paths of the arm

Conclusions

The major observations of this work can be summarized as follows:

§ For a redundant robot the inverse kinematics has infinite number of solutions, but

§ when using the neural network method there is only one desired solution from these infinite

§ solutions, since the training session is done using the non-singular set of solution.

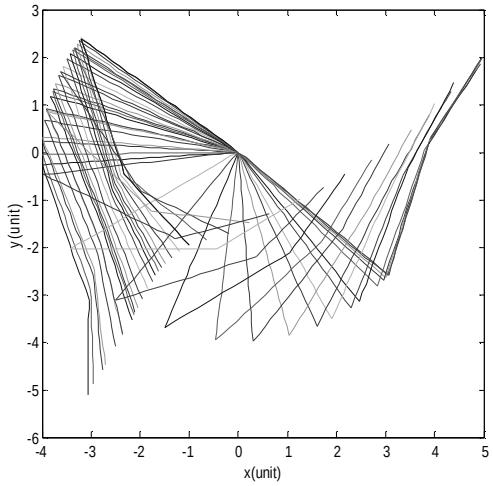


Fig. 21. The configuration of the arm

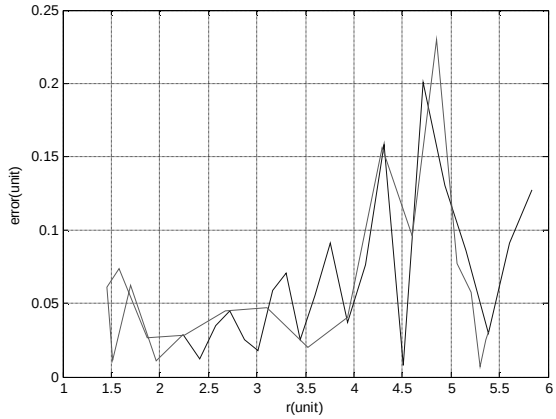


Fig. 22. The error versus the distance from the base

§ In conventional methods for the inverse kinematics, one must find the joint space singularity in order to avoid it, but when using the neural network method there is no need to find the joint space singularity because this method gives one desired solution only.

§ Since the neural network method is an approximation method, the results show that the obtained solution from the proposed neural network model is depends on the initial weights in training session. i.e. the use of a neural network decreases the repeatability for kinematics solution of the manipulator arm.

REFERENCES

[1] John J. Craig "Introduction to Robotics Mechanics and Control",1986.

[2] Mark W. Spong and M. Vidyasagar, "Robot Dynamic and Control", John Wiley and Sons, Inc., 1989.

[3] Alain Liegeois "Robot Technology: Performance and computer-Aided Design", Volume 7, Prentice-Hall, 1985 .

[4] Jacobsen, S.C., Iversen, E.K., Knutti, D.F., Johnson, R.T., and Biggers, K.B., "Design of the Utah/MIT Dexterous Hand," in Proc. IEEE Int. Conf. Robotics and Automation, San Francisco, pp. 1520-1532, April 7-10, 1986.

[5] E. S. Conkur "Real Time Path Planning and Obstacle Avoidance Algorithms for Redundant Manipulators", Ph.D. Thesis, Department of Mechanical Engineering, University of Bristol, September 1997.

[6] Chetan Kapoor and Delbert Tesar "Kinematics Abstraction for General Manipulator Control", Robotics Research Group, The University of Texas at Austin, JJPRC/MERB 1.206, mail Code R9925, Austin, Texas 78712-1100, 1990.

[7] Farshid Magami Asi "Analysis of Hyper Redundant Manipulators", M.Sc. Thesis, Mechanical Engineering Department, Villanova University, August 1998.

[8] J.Somlo, A.Sokolov, V.Lukanyin "Intelligent Robot Control. The Automation Trajectory Planning", Department of Manufacturing Engineering, Budabist, University of Technology and Economics, 2002.

[9] Shigang Yue and Dominik Henrich "Point-to-Point Trajectory Planning of Flexible Redundant Robot Manipulators Using Genetic Algorithms", Embedded Systems and Robotics (RESY), Informatics Faculty, University of Kaiserslautern, D-67653 Kaiserslautern, Germany, 2001.

[10] Javier de Lope, Telmo Zarraonandia, Rafaela Gonzalez-Careaga, and Dario Maravall "Solving the Inverse Kinematics in Humanoid Robots: A Neural Approach", Department of artificial Intelligence, Faculty of Computer Science, University of

Politecnica de Madrid, Campus de Montegancedo, 28660 Madrid, Spain, 2004.
[11] Jack M. Zurada "Introduction to Artificial Neural systems", Jaico Publishing house, 1996.

الحل العكسي لذراع انسان الي متعدد زوايا الوصول بأستخدام الشبكات العصبية

الدكتور بهاء ابراهيم كاظم
قسم هندسة الميكاترونكس
كلية هندسة الخوارزمي – جامعة بغداد

الباحث سامر يحيى هادي
قسم هندسة السيطرة والنظم
الجامعة التكنولوجية

الخلاصة:

هنالك عدد غير منته (infinity) من الحلول العكسية (inverse kinematics solutions) لذراع الية مطولة (redundant arm) باستخدام الطرق التقليديه (conventional methods) , لذلك استخدمت طريقة الشبكه العصبيه (neural network technique) لإيجاد حل واحد مرغوب به (one desired solution) من هذه الحلول.

حيث استخدم في هذه البحث شبكه عصبيه مزدوجة (DNNM) لإيجاد الحلول العكسية (inverse kinematics solutions) لذراع تتكون من ثلاثة قطع تتحرك جميعها في سطح واحد (three links redundant planar robot). إن ألشبكه العصبية المزدوجة (DNNM) المستخدمة في هذا العمل أظهرت كفاءة عالية في تقليل الوقت اللازم لمرحلة التدريب (training).

إن تأثير عدد مجموعات التدريب (training sets) على مخارج (outputs) ألشبكه العصبية المزدوجة (DNNM) قد تم دراستها في هذا العمل, و أظهرت النتائج بأن الدقة في إيجاد موقع نهاية الذراع (end effector) قد تزيد عندما يضاعف عدد مجموعات التدريب (training sets) .

إن طريقة إيجاد المسار بين نقطتين (point-to-point trajectory planning method) قد تم دراستها في هذا العمل كذلك تم دراسة تأثير تغير عدد مجموعات التدريب (training sets) على الدقة.