International Journal of Advanced Computer Research (ISSN (print): 2249-7277 ISSN (online): 2277-7970)
Volume-4 Number-1 Issue-14 March-2014

Implementation and Validation of Artificial Intelligence Techniques for Robotic Surgery

Aarshay Jain[1], Deepansh Jagotra[2], Vijayant Agarwal[3]

Abstract

The primary focus of this study is implementation of Artificial Intelligence (AI) technique for developing an inverse kinematics solution for the Raven-II^{TM} surgical research robot [1]. First, the kinematic model of the Raven-II^{TM} robot was analysed along with the proposed analytical solution [2] for inverse kinematics problem. Next, The Artificial Neural Network (ANN) techniques was implemented. The training data for the same was careful selected by keeping manipulability constraints in mind. Finally, the results were verified using elliptical trajectories. The originally proposed analytical solution was found to be computationally inefficient, gave multiple solutions and its existence necessitates the use of the Standard Raven-II^{TM} Tool [2]. The solution devised using ANN technique gave a single solution which was thirteen times faster than the original solution. Moreover, it is generic in nature and can be used for any type of tool. Thus, a novel solution for solving the inverse kinematics problem of the Raven-II surgical robot was formulated and confirmed.

Keywords

Surgical Robotics, Artificial Intelligence, Cognitive Robotics, Artificial Neural Network, Computer Assisted Surgery.

1. Introduction

In this ever evolving age of robotics, smarter and more innovative technology has made inroads into almost every field of modern human civilization.

Manuscript received January 09, 2014.

Aarshay Jain, Manufacturing Processes and Automation Engineering, Netaji Subhas Institute of Technology, New Delhi, India.

Deepansh Jagotra, Manufacturing Processes and Automation Engineering, Netaji Subhas Institute of Technology, New Delhi, India.

Vijayant Agarwal, Manufacturing Processes and Automation Engineering, Netaji Subhas Institute of Technology, New Delhi, India.

Tele-surgery is one such technology where surgical operations can be performed precisely and remotely by medical practitioners. This technology is employed to conduct Minimally Invasive Surgical (MIS) procedures which involve making small incisions for surgery. Advantage of robotic surgery is that the surgical instruments located at the tip of the manipulator may provide a full range of motion and ability, allowing the instruments to rotate more than 360° through tiny incisions with high level of dexterity and degree of freedom. Another important advantage is the maximizing of the movable range of the surgery tool.

Most existing surgical robotic systems employ analytical techniques for tool tracking, which is more accurate as compared to other techniques like numerical solutions or artificial intelligence. But broadly there are two disadvantage of the analytical technique. First, validity of an analytical solution requires certain specific design constraints like intersection of at least three consecutive axes [3]. Second, the analytical technique gives multiple solutions and in most of the cases we have to use constraints like limitation of joint movements which is not desirable for the application like robotic surgery. Another disadvantage of analytical techniques is that it requires knowledge of multi-body dynamic system concepts with which a lot of which medical professionals may not be comfortable with. The dynamics or mechanics of a manipulator is a Multi Input Multi Output (MIMO) system which is uncertain, tightly coupled, non-linear, singular and transcendental. The conventional methods are also not computationally efficient and not applicable to real time applications due to the presence of non-linear parameters like coupling, flexibility, friction, backlash, etc. Accurate mathematical modelling of these is highly unlikely. Therefore, in this work, we will investigate and propose algorithm based on AI techniques or soft computing which can also be used by medical researchers in this domain as very little knowledge of system dynamics is required since it is based upon input–output data pairs only or on the rule based systems [4] [5] [6] [7]. The artificial intelligence technique can be automated by use of sensors so that a surgeon can easily make modifications in the system. This would in turn assist the robotic surgeon to feel more comfortable with the system after having some background knowledge of the same. Moreover, concerted efforts have been made to devise an inverse kinematics solution using Artificial Neural Networks (ANN), which has considerable advantages over the traditional analytical or numerical solutions. One prominent advantage is that ANN provides a unique solution and this can be limited to areas where we train the network by selecting data points such that only a specific region of the workspace is covered. The network output will automatically lie in that specific region. Another important advantage is the speed of solution. An analytical solution involves calling of trigonometric functions which takes more computational time as compared to a neural network which uses simple additions, multiplications and exponential functions. In this study, the ANN solution developed was found to be thirteen times faster than analytical solution. This study utilizes MATLABTM as the medium of analysis along with compatible toolboxes like Peter Corke Robotics Toolbox [8] and Neural Networks Toolbox MATLABTM, which aided in the enhancement of research process. This would allow easy optimization of results which can then be implemented directly on the Robot Operating Software platform [9] which is currently being used for research on Raven-IITM robot.

2. RAVEN-II™ Kinematic Model

The Raven-II™ system has two spherical positioning mechanisms with 3-DOF supporting interchangeable 4-DOF instruments [1]. The two positioning mechanisms form the Gold (left) and the Green (right) arm. This study focuses on providing an inverse kinematics solution for the Green Arm. The DH parameters [9], i.e. the physical parameters, of the right arm are specified in Table 1. The Standard Raven-II™ tool length has been taken into account to get the actual tool position of the tool in the inverse kinematics solution.

Table 1: DH Parameters of Right Spherical Mechanism [1]

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	180°	0	0	θ_1
2	75°	0	0	θ_2
3	52°	0	d_3	-90°

The task of defining the kinematic Model in MATLAB was done by utilizing the open source Peter Corke Robotics Toolbox [8]. Two properties have to be taken into consideration for accurate definition of the kinematic model. First, a tool transform of '0.47 along negative z-axis' was chosen to cater to the length of the Standard Raven-II™ tool [2]. Secondly, the option of 'modifiedDH' parameters has to be selected because the inverse kinematics solutions derived in [2] are based on forward kinematics transform as stated in [11]. But, the Peter Corke Toolbox uses a default transform which is opposite to this screw transform as in [8].
Theoretically a revolute joint is capable of rotating 360 degrees and a prismatic joint can have as long a linear motion as desired. But physically, every joint has some restraint. Joint limits are provided to overcome the possibility of mechanical interactions between various parts of a robot. Another important use of joint limits is to narrow down the range of inverse kinematics solutions obtained from analytical analysis. Also, joint limits help in preventing singularity poses of the robotic manipulator. But at the same time, the joint limits should not hamper a robot's manoeuvring capabilities and its ability to perform the desired task. The joint limits are given in Table 2.

Table 2: Joint Limits

JOINT	LIMITS
q1	0 – 90 degrees
q3	45 – 135 degrees
d3	0.24 – 0.46 mm

A 3D CAD model of the Raven-II™ robot was developed and then imported into MATLAB™ with SimMechanics™ Link toolbox that links a CAD assembly to MATLAB environment [8]. The model was then simulated using the AI techniques

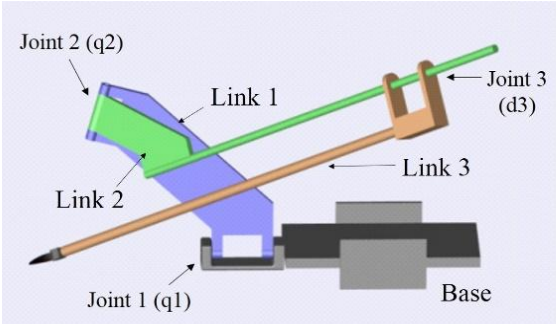

Figure 1: Raven-IITM 3D CAD model imported into SimMechanics

The 3D CAD model lead to a better understanding of the Raven-II™ mechanisms, DH parameters [10], manoeuvrability and robot workspace. The CAD model was made incorporating the tool design to keep the considerations of the tool in mind while deciding the workspace.To be imported into MATLAB™, the CAD model was converted into an Extensible Markup Language (XML) format through the SimMechanics Link utility in SimMechanics™. Alongside the XML file, individual parts were converted into STL (STereoLithography) format for the purpose of actual visualization. The XML file was imported into SimMechanics environment using the 'mech_import' command. The imported assembly has been shown in figure 1. This is a kinematic model which fully complies with the DH parameters of the robot as defined in [1].

The SimMechanics Toolbox of MATLAB™ used here provides various advantages. It can directly make a model corresponding to the mates applied in the CAD model assembly. But, the mates have to be carefully chosen such that it can identify the model using those mates. Another important factor to be considered is the initial pose. SimMechanics takes the pose in which the model is imported as the zero pose of the robot. Thus, the CAD assembly has to be imported in a state corresponding to its zero pose. A novel Simulink block, as depicted in figure 2, was developed to create a joint space trajectory with the customized ANN. The Base, Link 1, Link 2, Link 3 blocks depict the different kinematic links as recognized by SimMechanics from the imported CAD assembly. These contain kinematic information including DH parameters. The q1, q2 and d3 blocks represent the joints as shown in figure 1, where B and F refer to base and follower respectively for a particular joint. For instance, for the joint q2 (figure 1) represented by block q2 (figure 2), B refers to link 1 and F refers to link 2. The third connection in each joint block us the Joint Actuator, which takes joint angle, velocity and acceleration as input and acts as an interface between ANN and the joint block. Here known parameters are different positions of tool's tips and joints angles were calculated using ANN block.

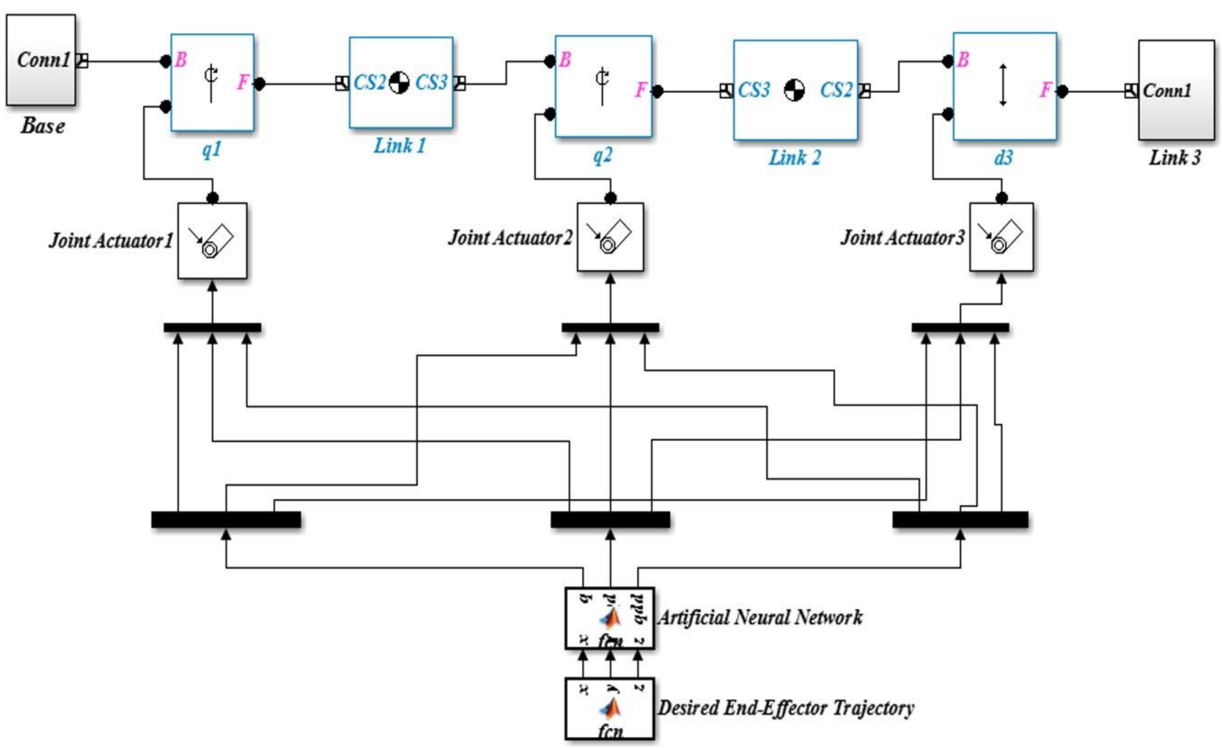

Figure 2: SimMechanics Simulink Block

3. Inverse Kinematics

The inverse kinematics problem is used to determine the actual joint angle required to attain a specific pose or trajectory of the robot end-effector [12]. Various methods are used for solving the inverse kinematics problem. These include the analytical solution, numerical solution and Artificial Intelligence.

An analytical solution for inverse kinematics problem has been derived in [2]. But, it offers eight solutions for each end-effecter position. In practice, limiting joint angles is a method used to reduce the number of solutions. But, it is not a concrete method as it does not guarantee single solution. Also, an analytical solution has been derived only for the Standard Raven-IITM tool which assumes the link length of the third link (a3) as zero. But, for a different tool the analytical solution may not exist and then we will have to resort to alternative forms of solution as proposed later. Additionally, the analytical solution does not consider the quality of the solution in terms of robot manipulability [11], which is an important parameter that should be considered while deciding the appropriate solution. On further analysis, the analytical solution was found to have another drawback. The function involves calling trigonometric functions which is high on computational requirements and thus is difficult to implement on real time systems.

Another method for determining inverse kinematics is the numerical solution wherein the solution for joint angles is determined iteratively. The error in the pose from desired pose is updated iteratively till the desired error is achieved. One advantage that this method harbours is that it is high on accuracy. But it has various disadvantages. First, it requires an initial estimate of the solution. Second, the computational time required is very high as compared to AI and analytical method.

Artificial Neural Network is an efficient AI tool used for identifying highly non-linear systems. One important application of the same is solving inverse kinematics problem in robotics. It is both accurate and gives a unique solution and thus is best suited to solve the problem. The Levenberg-Marquardt backpropagation algorithm [14] was customized for the same. Developing an ANN involves deciding an appropriate network architecture and then training the network with some training data. A custom neural network was made having architecture as shown.

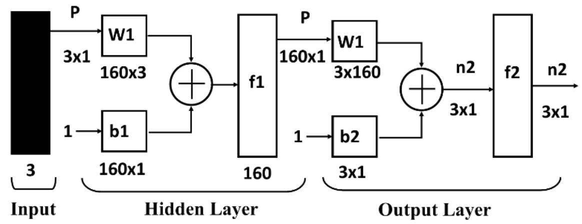

Figure 3: Neural Network Architecture

This is a 2-Layer Neural Network with the 3 inputs as the end effecter pose, 1 hidden layer consisting of 160 neurons with 'tangent sigmoid' activation function (f1) and 1 output layer consisting of 3 outputs as the inverse kinematics solution in terms of joint variables with 'linear' activation function (f2). The block diagram satisfies the notations used in [14]. Here, 'P' is the input vector, 'W' denotes the weight vector matrix, 'b' denotes the bias vector, 'n' is the net input to the neuron activation function and 'a' is the final neuron output. Other important parameters used in the custom network are enlisted below:

Table 3: Neural Network Parameters as per MATLAB Neural Networks Toolbox

Property	Value
inputs{1}.processFcn	mapminmax
layers{1}.processFcn	tansig
layers{2}.processFcn	purelin
outputs{1}.processFcn	mapminmax
divideFcn	dividetrain
trainFcn	trainlm
trainParam.mu	0.013
trainParam.mu_inc	3
trainParam.mu_dec	0.075

Another important aspect in creating the neural network is selection of appropriate training data which should cover the entire workspace and at the same time should be concise to allow fast training of the neural network. Also, our training data selection algorithm is designed so as to select only the high manipulability data. Here, manipulability refers to the ability of the robotic manipulator to achieve a specific point in the workspace. The manipulability measure [11], [13] is given as. Lower the value of w, poorer is the manipulability or in other words, nearer is the robot to the condition of singularity.

The algorithm for determining the training data can be described stepwise as follows:

- The joint angles were varied in small steps (one degree for joint 1 and 2 and one mm for joint 3) and the forward kinematics transforms were found. These small steps ensured that we cover the entire workspace of operation.
- The result gave more than 1.8 million values containing the robot pose (position of end effecter with respect to Remote Motion Centre [2]) and corresponding joint angles. Now, this data needs to be narrowed down such that we can cover the entire workspace in least possible values. For this, the resultant poses were rounded off to the nearest centimetre. This resulted in a high repetition in the end effecter poses.
- Then a sorting algorithm was developed using which all the repeating sets of poses were identified and separated. This resulted in just 8491 unique sets.
- Now, a selection of the best value among the recurring values was done using the concept of manipulability. The joint angles corresponding to the highest manipulability among the repeated end effecter poses was chosen.
- Finally, a set of 8491 values was collected from original 1.8 million values. These were then restored to the original values, i.e. the ones before rounding off, and then used for training.

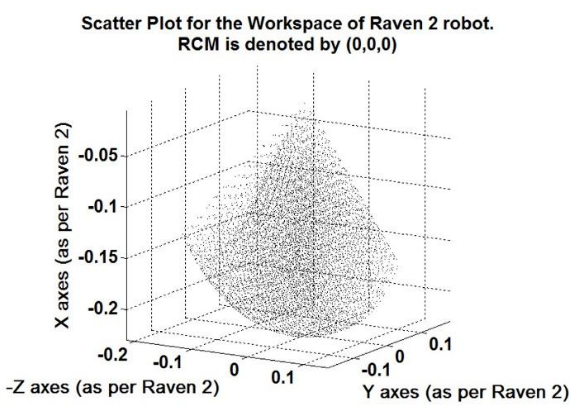

Figure 4: Training Data Points

4. Solution Testing

The neural network parameters were optimized through testing. Testing was done using two sets of data points. First, the 1.8 million values calculated using the entire joint angle range. But physically, the robot operates in a smaller region that covered by the entire joint range. Thus, a smaller joint range was chosen which covers the region of normal operation of robot. These two test data spaces are chosen to give a clear idea of validity of the solution formed. The results obtained from the network testing on the selected workspace are illustrated in Table 4. The corresponding errors separately in the X, Y, and Z dimensions are shown in figure 7, figure 8 and figure 9 respectively. For robotic surgery, this error range is quite satisfactory.

Table 4: Neural Network Output Analysis

	X-axes	Y-axes	Z-axes
RMS error (meters)	4.92 e-05	3.59 e-05	3.61 e-05
MAX error (meters)	6.41 e-04	3.22 e-04	4.51 e-04
MIN error (meters)	4.06 e-11	3.52 e-11	9.44 e-12

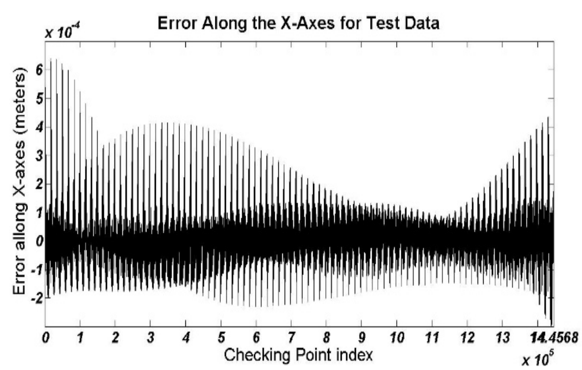

Figure 5: Error along X-Axis

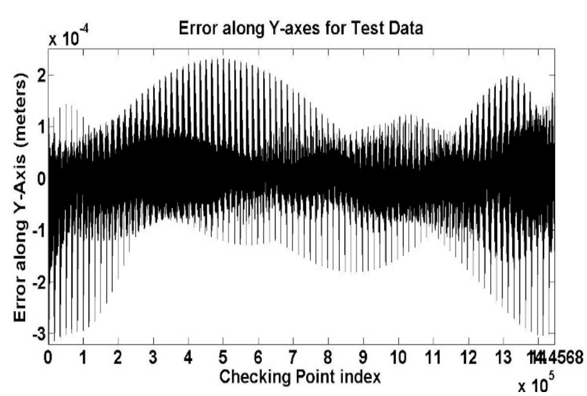

Figure 6: Error along Y-Axis

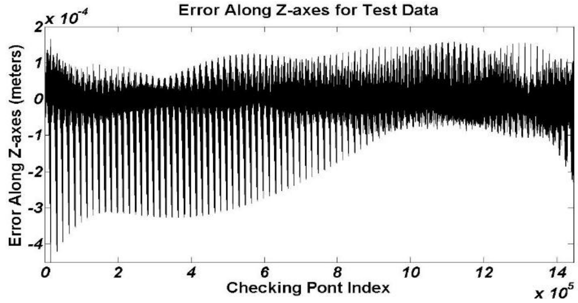

Figure 7: Error along Z-Axis

As evident from figures, a vast majority of points lie within very small error range. Also, when tested with the entire workspace, the error values are almost same with slight variations due to poor performance at the near singularity points. These points will be rarely required in actual practice and can be ignored for testing. We can increase the number of layers in the network and hence improve the performance at these points also. But, this would result in higher computation requirements for solution and also much higher training time.

The solution obtained was also tested on elliptical end effector trajectories on planes parallel to the XY, XZ and YZ planes as shown in figures 8, figure 9 and figure 10. The figures contains actual points marked as '+' in red color and solution obtained by application of ANN overlaps them and marked as 'o' in blue color. The high accuracy reflected in these figures corroborate the validity of AI solution.

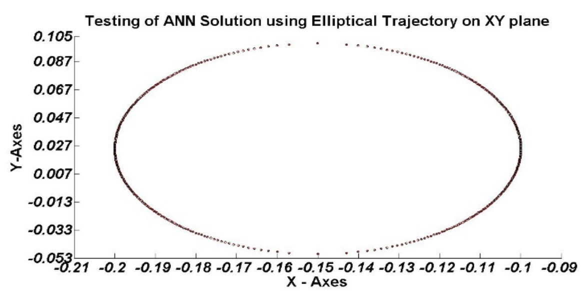

Figure 8: Trajectory Testing on XY Plane

The artificial neural network solution shows a significant improvement in computational time over analytical solution. Both the solution codes were tested for the testing data set as mentioned above constituting of 1445679 values. The results obtained are shown in table 5.

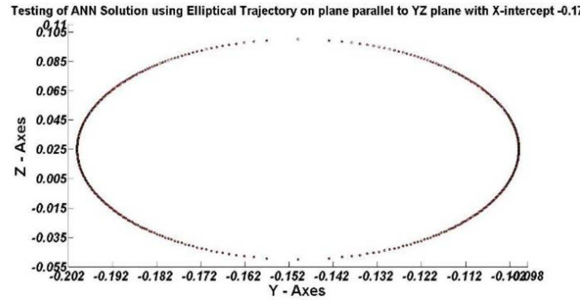

Figure 9: Trajectory Testing on YZ Plane

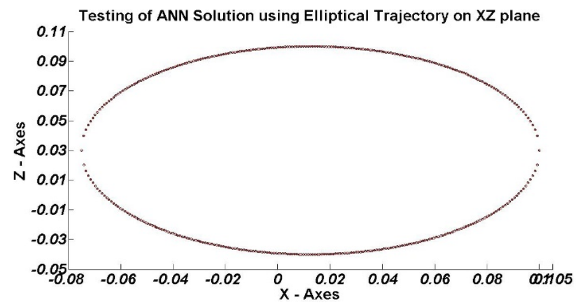

Figure 10: Trajectory Testing on YZ Plane Comparison of Computation Times

Table 5: Comparison of Computational Time

	No of Test Inputs	**Total Time taken (sec)**	**Time for one solution (μs)**
1.	1445679	65.063	45
2.	1445679	4.693	3.25

This clearly portrays the superiority of inverse kinematic solution based on neural networks as it is almost 13 times faster than the analytical solution. The neural network solution will thus improve the dynamic response of the robot. Additionally, both the solutions were executed on same platform. Thus, the enhancement in computational efficiency is not platform dependent.

Acknowledgment

We would like to thank Dr. Blake Hannaford, University of Washington and his team for developing the open-source research platform to enable advances in surgical robotics.

References

[1] Blake Hannaford, Jacob Rosen, Diana W. Friedman, Hawkeye King, Phillip Roan, Lei Cheng, Daniel Glozman, Ji Ma, Sina Nia Kosar,and Lee White, "Raven-II: An Open Platform for Surgical Robotics Research", IEEE Transactions on Biomedical Engineering, 60(4), 2013.
[2] Hawkeye King, Sina Nia Kosari, Blake Hannaford, Ji Ma, "Kinematic Analysis of the Raven-IITM Research Surgical Robot Platform.", UWEE (University of Washington Electrical Engineering) Technical Report Number UWEETR, 2012.
[3] D. Pieper, "The Kinematics of Manipulators Under Computer Control. Desertion", Stanford University, 1968.
[4] Ren´e V. Mayorga∗, Pronnapa Sanongboon, "Inverse kinematics and geometrically bounded singularities prevention of redundant manipulators: An Artificial Neural Network approach", Elsevier Robotics and Autonomous Systems 53, pp. 164–176, 2005.
[5] Oscar Castillo, Patricia Melin, "Intelligent adaptive model-based control of robotic dynamic systems with a hybrid fuzzy-neural approach", Elsevier Applied Soft Computing 3(4), pp. 363–378, 2003.
[6] Young H. Kim and Frank L. Lewis, "Neural Network Output Feedback Control of Robot Manipulators", IEEE Transactions on Robotics And Automation 15(2), 1999.
[7] Amar Khoukhi, Luc Baron, Marek Balazinski, Kudret Demirli, "A hierarchical neuro-fuzzy system to near optimal-time trajectory planning of redundant manipulators", Elsevier Engineering Applications of Artificial Intelligence 21(7), pp. 974-984, 2008.
[8] P.I. Corke, "Robotics, Vision & Control", Springer 2011, ISBN 978-3-642-20143-1, 2011.
[9] ROS (Robot Operating Software), Available: http://wiki.ros.org/.
[10] J. Denavit and R.S. Hartenberg, "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices", Journal of Applied Mechanics, pp 215-221, 1955.
[11] John J. Craig, "Introduction to Robotics – Mechanics and Control", Pearson Education, ISBN 0201543613, 9780201543612, 2005.
[12] D. Baker and C. Wampler, "On the Inverse Kinematics of Redundant Manipulators", International Journal of Robotics Research 7(2), 1988.
[13] T. Yoshikawa, "Manipulability of Robotic Mechanisms" The International Journal of Robotics Research, MIT Press, Cambridge, MA 4(2), 1985.
[14] M.T. Hagan, H.B. Demuth, M Beale, "Neural Network Design", International Thomson Publishing Inc, ISBN 981-240-376-0, 1996.
[15] Vijyant Agarwal, "Trajectory Planning of Redundant Manipulator using Fuzzy Clustering method" International Journal of Advanced Manufacturing Technology 61(5-8): pp. 727-744, 2012.
[16] Loi-Wah Sun, Frederick Van Meer, Yan Bailly, and Chung Kwong Yeung, "Design and Development of a Da Vinci Surgical System Simulator" In: Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation, 2007, Harbin, China, 2007.

Aarshay Jain, born in Delhi on 6th May 1992, is a Bachelor of Engineering student at Netaji Subhas Institute of Technology pursuing a major in Manufacturing Processes and Automation Engineering. Specific research interests include Machine Learning, Artificial Intelligence and Robotics.

Deepansh Jagotra, born in Delhi on 5th December 1991, is currently pursuing Bachelors in Manufacturing Processes and Automation Engineering from the Netaji Subhas Institute of Technology, Delhi. He has a profound interest in the field of Robotics, Automation and Artificial Intelligence.

Dr. Vijayant Agarwal, did his B.Tech in Mechanical Engineering from MMEC Gorakhpur, India in 1993 and M.Tech from IIT Delhi, India in 1995, Ph.D in 2010 for University of Delhi. At present he is Associate Professor in NSIT Delhi, India. He has authored more than 25 articles in International Journals and Conferences.