The 5th Joint International Conference on Multibody System Dynamics
June 24–28, 2018, Lisbon, Portugal

Inverse Kinematics for General 6R Manipulators in RoboAnalyzer

Sasanka Sekhar Sinha[1], Rajeevlochana G. Chittawadigi[2] and Subir Kumar Saha[3]

[1, 3] *Department of Mechanical Engineering, Indian Institute of Technology Delhi, New Delhi, India*
[1]*mez168275@mech.iitd.ac.in,* [3]*saha@mech.iitd.ac.in*
[2] *Department of Mechanical Engineering, Amrita School of Engineering, Bengaluru, Amrita Vishwa Vidyapeetham, India*
[2] *rg_chittawadigi@blr.amrita.edu*

ABSTRACT — *Robotics has incessantly pervaded the world of industrial automation. As a subject, it has become indispensable in the academic and research curriculum. The various concepts relating to robot kinematics, dynamics, motion planning, etc., are often incomprehensible to beginners due to the abstruse underlying mathematics. Several robotics learning software have sprung up to reduce the learning curve. RoboAnalyzer is one such 3D model based robotics learning software primarily focussed on serial robot analysis based on DH (Denavit-Hartenberg) convention of robot geometry. A comprehensive description of the software functionalities and features is available online. This paper reports a further development of RoboAnalyzer in the form of addition of inverse kinematics of a generic 6R serial manipulator to the existing Inverse Kinematics module of the software.*

1 Introduction

Inverse kinematics is quintessential to the control of robots. The desired task is often prescribed in terms of the end-effector coordinates in the Cartesian space and needs to be translated to the joint-space coordinates. This is precisely the problem of inverse kinematics. The first documented effort to solve for the inverse kinematics of a 6R manipulator was by Pieper [1]. He gave closed-form solutions for manipulators where three consecutive axes were concurrent. For general 6R manipulators, he proposed iterative numerical techniques and was able to set upper bounds for the number of solutions using an elimination strategy. Roth et al. [2] used synthetic arguments to give a non-constructive proof of an upper bound of 32 on the number of solutions to this problem. Albala and Angeles [3] gave the first constructive solution to this problem as a 12×12 determinant whose entries were quartic polynomials in the half-tangent of one of the joint variables. This method of constructing a monic polynomial in the half-tangent of one of the joint angles is called the *lower dimensional approach*. Duffy and Crane [4] combined spherical trigonometry and dialytic elimination to obtain a 32-degree polynomial in the half-tangent of a joint variable and found that the polynomial always yielded extraneous roots. Later on, Tsai and Morgan [5] took recourse to *higher dimensional approach*. Eight second degree equations were solved numerically using polynomial homotopy continuation to yield a maximum of 16 solutions for various 6R manipulators. This led them to conjecture that the problem has at most 16 solutions. Lee and Liang [6] gave the exact solution in lower dimensions. They obtained a 16-degree polynomial in the half-tangent of a joint variable through dialytic elimination, thus confirming the conjecture of Tsai and Morgan. Later, Raghavan and Roth [7] gave a simpler procedure than the one formulated by Lee and Liang using the properties of the polynomial ideal of the multivariate equations. However, the process of expanding a symbolic determinant to get the 16-degree polynomial is slow and the problem of computing roots of such polynomials can be ill-conditioned. Manocha and Canny [8] reduced the problem of computing the matrix determinant as a monic polynomial to finding the eigenvalues of a matrix. Backward stable algorithms and fast implementations for eigen decomposition greatly enhances the efficiency and numerical stability of the solution procedure.

Till recently, RoboAnalyzer was able to solve for the inverse kinematics of a 6R wrist-partition robot using the methodology described in [9]. Precisely, the robot architecture permitted a decoupling solution strategy- positioning with the articulated arm (first 3 revolute joints) and orientation using the wrist (last 3 intersecting revolute joints) or the spherical joint. However in the presence of wrist offsets, this method fails. Most industrial 6R manipulators have this structure. However, errors in manufacturing and assembly necessitate kinematic identification followed by calibration and compensation. A general 6R inverse kinematic solution procedure will definitely eliminate the need for compensation.

Manocha's code was real-time and met industrial level of performance. It was available open-source in Linux environment. This inverse kinematics routine could enhance the capabilities of RoboAnalyzer software. A detailed description of the existing software functionalities and features is available in [10]. To incorporate in RoboAnalyzer, Manocha's code had to be ported to MS-Windows platform. Section 2 of this paper recapitulates the formulation strategy proposed by Raghavan and Roth [7]. Their solution to the inverse kinematics problem had certain shortcomings which were later addressed by Manocha and Canny [8], which is also discussed in a subsection. Section 3 deals with a numerical example and the results from RoboAnalyzer software demonstrating successful integration of the code with RoboAnalyzer.

2 Inverse Kinematics

The determination of the joint angles required to reach desired end-effector position and orientation is inverse kinematics. For serial robots, inverse kinematics usually has multiple solutions and hence care has to be taken on how to derive it. In this section, an overview of the inverse kinematics formulation, used in this paper, is provided.

2.1 Problem Formulation

The standard Denavit-Hartenberg (DH) convention has been used to model the 6R manipulator (R: Revolute). The links are numbered from 0 to 6 and are connected by 6 joints, as shown in Fig. 1. Joint *i* connects link *i*-1 to link *i*. The *i*th coordinate system is attached to the end of the (*i*-1)th link.

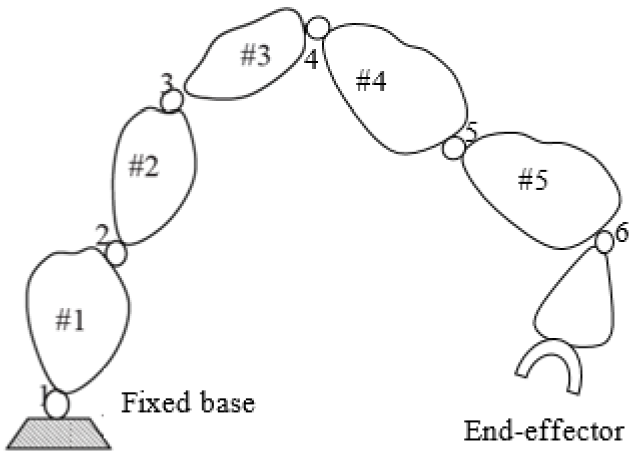


Fig. 1 Serial 6R manipulator

The 4 × 4 homogeneous transformation matrix relating coordinate systems *i*+1 and *i* is:

$$\mathbf{T}_i = \begin{bmatrix} c_i & -s_i\lambda_i & s_i\mu_i & a_ic_i \\ s_i & c_i\lambda_i & -c_i\mu_i & a_is_i \\ 0 & \mu_i & \lambda_i & b_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where

$s_i = sin\theta_i$, $c_i = cos\theta_i$, θ_i is the ith joint angle,

$\mu_i = sin\alpha_i$, $\lambda_i = cos\alpha_i$, α_i is the twist angle between the joint axes i and i+1,

a_i is the link length of link i and b_i is the corresponding joint offset.

For the 6R manipulator with revolute joints, a_i's,b_i's, μ_i's and λ_i's are known. For the inverse kinematics problem, we are given the pose, consisting of both position and orientation, of the end-effector (EE) as:

$$\mathbf{T}_{EE} = \begin{bmatrix} l_x & m_x & n_x & p_x \\ l_y & m_y & n_y & p_y \\ l_z & m_z & n_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

We are required to compute the joint angles $\theta_1,\ \theta_2,\ \theta_3,\ \theta_4, \theta_5$ and θ_6 such that

$$\mathbf{T}_1\mathbf{T}_2\mathbf{T}_3\mathbf{T}_4\mathbf{T}_5\mathbf{T}_6 = \mathbf{T}_{EE} \tag{3}$$

2.2 Raghavan and Roth Solution [7]

Raghavan and Roth rearranged the matrix equation (3) as:

$$\mathbf{T}_3\mathbf{T}_4\mathbf{T}_5 = \mathbf{T}_2^{-1}\mathbf{T}_1^{-1}\mathbf{T}_{EE}\mathbf{T}_6^{-1} \tag{4}$$

The left hand side matrix has entries which are functions of $\theta_3,\ \theta_4$ and θ_5. Similarly, the right hand side matrix has entries which are functions of $\theta_1,\ \theta_2$ and θ_6. Moreover, the columns 3 and 4 of the right hand side are devoid of θ_6. Comparing the entries of column 3 on both sides yields 3 scalar equations, namely $\mathbf{p} = [p_1,\ p_2,\ p_3]^T$. Similarly, comparison of column 4 entries gave $\mathbf{l} = [l_1,\ l_2,\ l_3]^T$. Raghavan and Roth studied the properties of the Ideal generated by **p** and **l** and found that the left and right hand sides of the following equations have the same power products as the left and right sides of p_i and l_i:

$$\mathbf{p}.\mathbf{p}, \mathbf{p}.\mathbf{l}, \mathbf{p}\times\mathbf{l}, (\mathbf{p}.\mathbf{p})\mathbf{l} - 2(\mathbf{p}.\mathbf{l})\mathbf{p} \tag{5}$$

The above combinations along with **p** and **l** yield 14 scalar equations which can be arranged in matrix form as:

$$\mathbf{Q}\begin{bmatrix} s_1s_2 \\ s_1c_2 \\ c_1s_2 \\ c_1c_2 \\ s_1 \\ c_1 \\ s_2 \\ c_2 \end{bmatrix} = \mathbf{P}\begin{bmatrix} s_4s_5 \\ s_4c_5 \\ c_4s_5 \\ c_4c_5 \\ s_4 \\ c_4 \\ s_5 \\ c_5 \\ 1 \end{bmatrix} \tag{6}$$

where **Q** is a 14×8 matrix whose entries are all constants. **P** is a 14×9 matrix whose entries are linear functions of s_3 and c_3. Raghavan and Roth used 8 of the 14 equations in Eq. (6) to eliminate the left hand side power product terms in terms of the right hand side terms. Consequently, they obtained the following relation:

$$\mathbf{\Sigma}\begin{bmatrix} s_4 s_5 \\ s_4 c_5 \\ c_4 s_5 \\ c_4 c_5 \\ s_4 \\ c_4 \\ s_5 \\ c_5 \\ 1 \end{bmatrix} = \mathbf{0} \tag{7}$$

where $\mathbf{\Sigma}$ is a 6×9 matrix whose entries are linear combinations of s_3, c_3 and 1. Further simplification was done by substituting

$$s_3 = \frac{2x_3}{1+x_3^2},\ c_3 = \frac{1-x_3^2}{1+x_3^2},\ s_4 = \frac{2x_4}{1+x_4^2},\ c_3 = \frac{1-x_4^2}{1+x_4^2},\ s_5 = \frac{2x_5}{1+x_5^2},\ c_5 = \frac{1-x_5^2}{1+x_5^2}$$

where $x_3 = tan\left({}^{\theta_3}/_2\right)$, $x_4 = tan\left({}^{\theta_4}/_2\right)$ and $x_5 = tan\left({}^{\theta_5}/_2\right)$. Subsequently, Eq. (7) was multiplied by $(1 + x_3)^2$, $(1 + x_4)^2$ and $(1 + x_5)^2$ to clear out the denominators and resulted in the following equation:

$$\mathbf{\Sigma}'\begin{bmatrix} x_4^2 x_5^2 \\ x_4^2 x_5 \\ x_4^2 \\ x_4 x_5^2 \\ x_4 x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{bmatrix} = \mathbf{0} \tag{8}$$

where $\mathbf{\Sigma}'$ is a 6×9 matrix whose entries are quadratic polynomials in x_3. Equation (8) is not a square system. It is converted in to one by using dialytic elimination to yield

$$\begin{bmatrix} \mathbf{\Sigma}' & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}' \end{bmatrix}\begin{bmatrix} x_4^3 x_5^2 \\ x_4^3 x_5 \\ x_4^3 \\ x_4^2 x_5^2 \\ x_4^2 x_5 \\ x_4^2 \\ x_4 x_5^2 \\ x_4 x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{bmatrix} = \mathbf{0} \tag{9}$$

The above equation is overconstrained and the coefficient matrix must be singular for the system to have non-trivial solution. The determinant of the coeficient matrix is a polynomial of degree 24 in x_3. However, Raghavan and Roth proved that $(1 + x_3^2)^4$ exactly divides the determinant. Factoring out the term gives residual polynomial of degree 16 in x_3. The roots of this polynomial give the values of x_3 corrresponding to the 16 solutions of the inverse kinematics problem.

2.3 Problems in Raghavan and Roth Solution Procedure

Many properties of the ideal generated by $\mathbf{p}$ and $\mathbf{l}$ may be violated due to floating point computatition. The expanded form of the determinant in Eq. (9) may not be exactly divisible by $(1 + x_3^2)^4$. Finally, root computation of polynomials of degree 16 can be ill-conditioned. Needless to mention that symbolic expansión of the determinant of the coefficient matrix in Eq. (9) is not congenial for real time performance.

2.4 Reduction to Eigenvalue Problem by Manocha and Canny

Manocha and Canny reduced the problem of root finding to an eigenvalue problem. The 12×12 coefficient matrix in Eq. (9) was expressed as

$$\mathbf{\Sigma}'' \equiv \begin{bmatrix} \mathbf{\Sigma}' & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}' \end{bmatrix} = \mathbf{A}x_3^2 + \mathbf{B}x_3 + \mathbf{C} \tag{10}$$

where **A**, **B** and **C** are 12×12 matrices consisting of constant entries. When the matrix **A** is well-conditioned, Eq. (10) can be premultiplied by $\mathbf{A}^{-1}$ to yield

$$\bar{\mathbf{\Sigma}}'' = \mathbf{I}x_3^2 + \mathbf{A}^{-1}\mathbf{B}x_3 + \mathbf{A}^{-1}\mathbf{C} \tag{11}$$

where **I** is a 12×12 identity matrix. $\mathbf{A}^{-1}\mathbf{B}$ and $\mathbf{A}^{-1}\mathbf{C}$ are computed by linear equation solvers. From the rules of linear algebra, Manocha and Canny created a matrix **M** defined as:

$$\mathbf{M} \equiv \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{A}^{-1}\mathbf{C} & -\mathbf{A}^{-1}\mathbf{B} \end{bmatrix} \tag{12}$$

such that the eigenvalues of **M** correspond exactly to the roots of $\det(\mathbf{\Sigma}'') = 0$. The eigenvectors of **M** corresponding to eigenvalue x_3 have the structure

$$\mathbf{V} = \begin{bmatrix} \mathbf{v} \\ x_3\mathbf{v} \end{bmatrix} \tag{13}$$

where **v** is the vector corresponding to the variables appearing in Eq. (9). In case, the matrix **A** is ill-conditioned, the problem is reduced to a generalized eigenvalue problem by constructing two matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ defined as

$$\mathbf{M}_1 \equiv \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{A} \end{bmatrix}, \mathbf{M}_2 \equiv \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{C} & -\mathbf{B} \end{bmatrix} \tag{14}$$

Furthermore, the roots of $\det(\mathbf{\Sigma}'') = 0$ correspond to the eigenvalues of the generalized eigenvalue problem $\mathbf{M}_1 - x_3\mathbf{M}_2$. The eigenvectors remain the same as in Eq. (13). The details of the process are omitted here to avoid obscuring the reader.

3 Integration in RoboAnalyzer

The algorithm proposed by Manocha was in Linux and RoboAnalyzer software is developed for Microsoft Windows. Hence, there was a need to port his code to an environment that could be integrated with RoboAnalyzer software. The following libraries or activities were required to develop the ported program as Generic 6R.exe.

- EISPACK and LAPACK routines for matrix operations were interlaced with C programs which were based on Manocha's implementation.
- MinGW (Minimalist GNU for Windows) was used to get rid of dependencies [11]. MinGW provides a complete open source programming tool set- a port of the GNU Compiler Collection (GCC), including C, C++, ADA and Fortran compilers.
- GNU Binutils for Windows (assembler, linker, archive manager)- which is suitable for the development of native MS-Windows applications, and does not depend on any 3rd-party C-Runtime DLLs. It depends on a number of Microsoft DLLs provided as components of the operating system.

RoboAnalyzer already had an Inverse Kinematics (IKin) module which could solve for standard robot architecture for which closed-form solution exists. These are reported in [12]. To integrate the new generic IKin 6R program, a new entry was added in the list of robots. Upon selecting it, an user interface for input of the DH parameters of the generic 6R robot along with the desired end-effector pose is shown to the user. Once the user provides the input values, RoboAnalyzer writes these data in a pre-determined file structure for the 6R generic program to read. The program (Generic6R.exe) then executes the ported version of formulation explained in Section 2, and writes the output in a file. The output file is then read by RoboAnalyzer IKin module and displays the possible solutions. These steps are illustrated in Fig. 2.

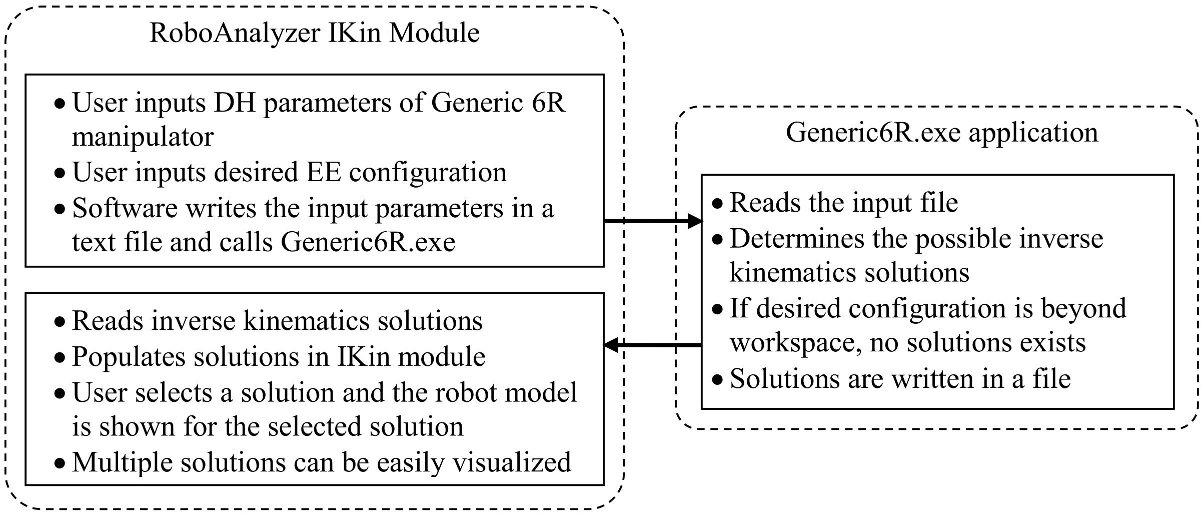

RoboAnalyzer IKin Module
• User inputs DH parameters of Generic 6R manipulator
• User inputs desired EE configuration
• Software writes the input parameters in a text file and calls Generic6R.exe
• Reads inverse kinematics solutions
• Populates solutions in IKin module
• User selects a solution and the robot model is shown for the selected solution
• Multiple solutions can be easily visualized
Generic6R.exe application
• Reads the input file
• Determines the possible inverse kinematics solutions
• If desired configuration is beyond workspace, no solutions exists
• Solutions are written in a file

Fig. 2 Integration of RoboAnalyzer IKin module and Generic6R application

4 Numerical Example

The implementation proposed in this paper has been tested for various robot configurations which do not have a wrist-partitioned architecture. An example is shown here for illustration. The matrix defining the pose of the end-effector, used as input is

$$\mathbf{T}_{EE} = \begin{bmatrix} -0.357279 & -0.85 & 0.387106 & 0.798811 \\ 0.915644 & -0.237 & 0.324694 & -0.000331 \\ -0.184246 & 0.470458 & 0.862973 & 1.200658 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

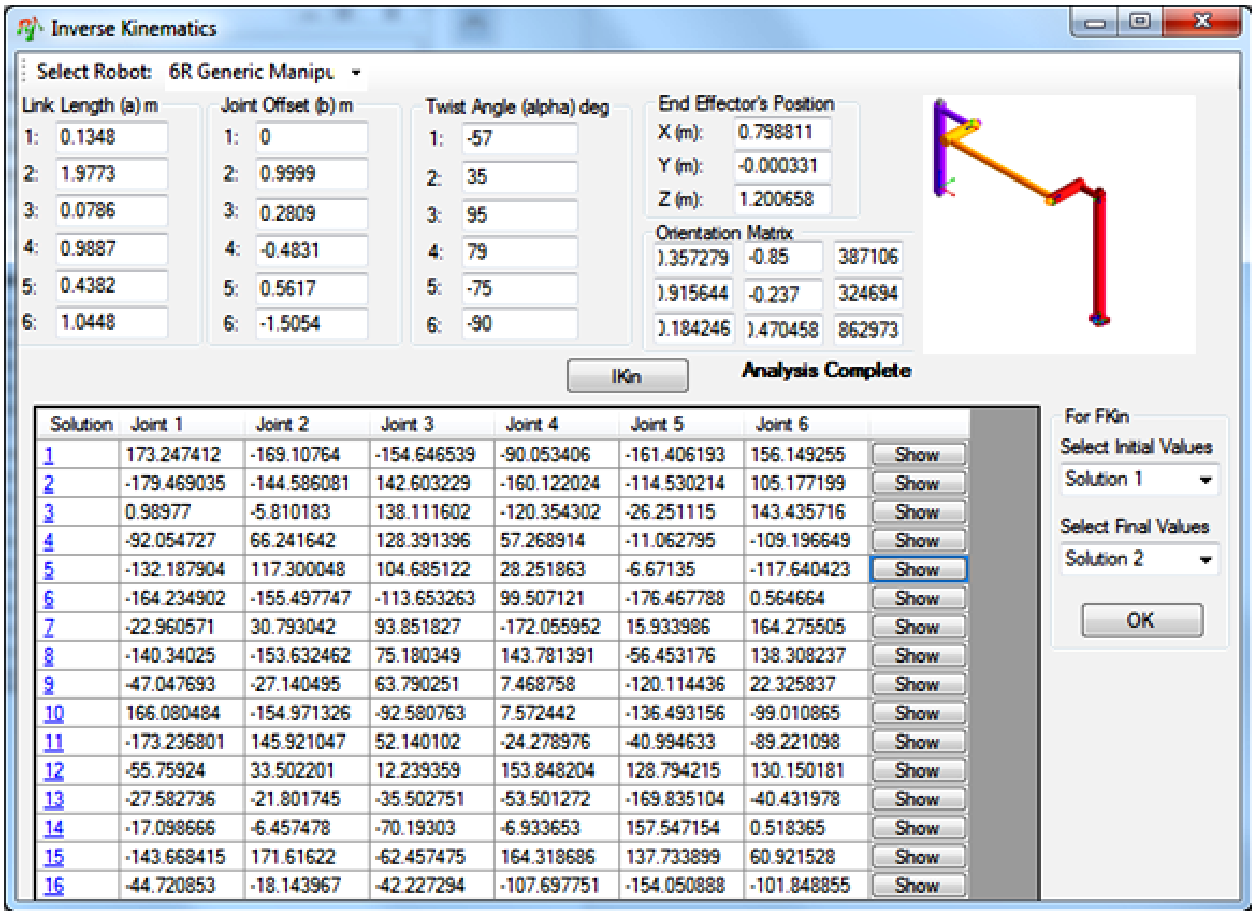
The DH parameters of the 6R manipulator for the example is

Sl. no.	*Link Length*	*Offset Distance*	*Twist Angle*
i	a_i	b_i	α_i
1	*0.1348*	*0*	*-57*
2	*1.9773*	*0.9999*	*35*
3	*0.0786*	*0.2809*	*95*
4	*0.9887*	*-0.4831*	*79*
5	*0.4382*	*0.5617*	*-75*
6	*1.0448*	*-1.5054*	*-90*

Tab. 1: DH parameters of a generic 6R manipulator

These data were passed on to Generic6R.exe application which determined 16 possible solutions. The IKin module of RoboAnalyzer with 16 possible solutions is shown in Fig. 3. The solution selected by the user will be shown to the end-user in the main RoboAnalyzer application.

The main RoboAnalyzer application also has a forward kinematics implementation that shows the homogeneous transformation matrix of the end-effector for the given DH parameters and joint angles. This matrix can be used to verify whether the input for IKin was obtained correctly or not. Thus, integration with RoboAnalyzer provides a faster and easier way to verify the solutions obtained in the 6RGeneric solver implemented in this paper. Figure 4 shows all the 16 solutions as viewed in the main RoboAnalyzer application. Note that these solutions can be shown one at a time and for the sake of comparison, they have been put as a single figure.



Select Robot: 6R Generic Manipu

	Link Length (a) m	Joint Offset (b) m	Twist Angle (alpha) deg
1:	0.1348	0	-57
2:	1.9773	0.9999	35
3:	0.0786	0.2809	95
4:	0.9887	-0.4831	79
5:	0.4382	0.5617	-75
6:	1.0448	-1.5054	-90

End Effector's Position
X (m): 0.798811
Y (m): -0.000331
Z (m): 1.200658

Orientation Matrix

).357279	-0.85	387106
).915644	-0.237	324694
).184246	).470458	862973

Analysis Complete

Solution	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
1	173.247412	-169.10764	-154.646539	-90.053406	-161.406193	156.149255
2	-179.469035	-144.586081	142.603229	-160.122024	-114.530214	105.177199
3	0.98977	-5.810183	138.111602	-120.354302	-26.251115	143.435716
4	-92.054727	66.241642	128.391396	57.268914	-11.062795	-109.196649
5	-132.187904	117.300048	104.685122	28.251863	-6.67135	-117.640423
6	-164.234902	-155.497747	-113.653263	99.507121	-176.467788	0.564664
7	-22.960571	30.793042	93.851827	-172.055952	15.933986	164.275505
8	-140.34025	-153.632462	75.180349	143.781391	-56.453176	138.308237
9	-47.047693	-27.140495	63.790251	7.468758	-120.114436	22.325837
10	166.080484	-154.971326	-92.580763	7.572442	-136.493156	-99.010865
11	-173.236801	145.921047	52.140102	-24.278976	-40.994633	-89.221098
12	-55.75924	33.502201	12.239359	153.848204	128.794215	130.150181
13	-27.582736	-21.801745	-35.502751	-53.501272	-169.835104	-40.431978
14	-17.098666	-6.457478	-70.19303	-6.933653	157.547154	0.518365
15	-143.668415	171.61622	-62.457475	164.318686	137.733899	60.921528
16	-44.720853	-18.143967	-42.227294	-107.697751	-154.050888	-101.848855

For FKin
Select Initial Values: Solution 1
Select Final Values: Solution 2

Fig. 3 Inverse Kinematics (IKin) module in RoboAnalyzer for general 6R manipulator

5 Conclusions

Inverse kinematics of serial robots has always been a complicated topic due to various challenges. One such is that equations should exist in closed form to solve it analytically. A methodology proposed by Manoha and Canny for the solution of inverse kinematics of a generic 6R manipulator was ported to Windows as an application. It was then integrated with RoboAnalyzer, a 3D model based robotics learning software, developed by the authors. The integration has resulted in an easy way to visualize and validate the 16 solutions possible for a generic 6R manipulator. To widen the use of the software, it has been made available free through http://www.roboanalyzer.com.

Acknowledgements

The authors are thankful to Prof. Dinesh Manocha for allowing the usage of parts of his inverse kinematics code and for his discussions regarding the implementation of the new features in RoboAnalyzer software.

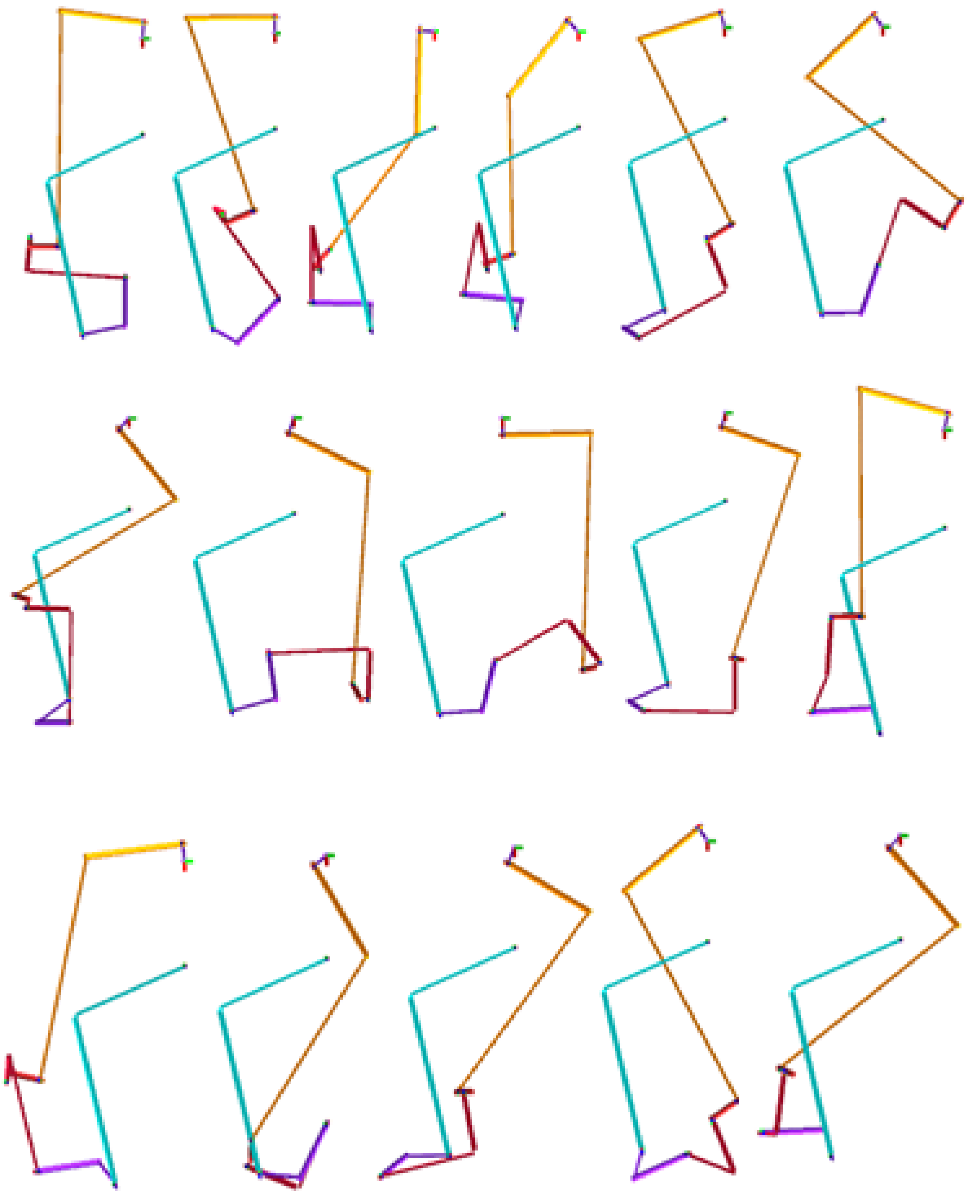


Fig. 4 3D representation of the 16 distinct inverse kinematic solutions

References

[1] D. Pieper, *The Kinematics of Manipulators Under Computer Control*, Ph.D. Thesis, Stanford University, 1968.

[2] B. Roth, J. Rastegar and V. Scheinman, *On the Design of Computer Controlled Manipulators*, In: On Theory and Practice of Robots and Manipulators. International Centre for Mechanical Sciences (Courses and Lectures), vol 201, Springer, Vienna, pp 93-113, 1974.

[3] H. Albala, and J. Angeles, “Numerical Solution to the Input-Output Displacement Equation of the General 7R Spatial Mechanism,” in *Proceedings of the Fifth World Congress on Theory of Machines and Mechanisms*, pp. 1008-1011, New Castle, United Kingdom, 1979.

[4] J. Duffy, and C. Crane, “A Displacement Analysis of the General Spatial 7R Mechanism,” *Mechanisms and Machine Theory*, vol. 15, no. 3, pp.153-169, 1980.

[5] L-W. Tsai and A. Morgan, “Solving the Kinematics of the Most General Six- and Five-Degree-of-Freedom Manipulators by Continuation Methods,” *Transactions of the ASME, Journal of Mechanisms, Transmissions, and Automation in Design*, vol. 107, no. 2, pp. 189-200, 1985.

[6] H-Y. Lee H-Y and C-G. Liang, “Displacement Analysis of the General Spatial 7-Link 7R Mechanism,” *Mechanism and Machine Theory*, vol. 23. no. 3, pp. 219-226, 1988.

[7] M. Raghavan, and B. Roth, “Inverse Kinematics of the General 6R Manipulator and Related Linkages,” *ASME Journal of Mechanical Design*, vol. 115, no. 3, pp. 502–508, 1993.

[8] D. Manocha, and J. F. Canny, “Efficient Inverse Kinematics for General 6R Manipulators,” *IEEE Transactions on Robotics and Autom*ation, vol. 10, no. 5, pp. 648–657, 1994.

[9] S. K. Saha, *Introduction to Robotics*, Second Edition, Tata McGraw Hill, New Delhi, 2014.

[10] R. S. Othayoth, R. G. Chittawadigi, R. P. Joshi and S. K. Saha, “Robot kinematics made easy using RoboAnalyzer software,” *Computer Applications in Engineering Education*, vol. 25, no. 5, pp. 669-680, 2017.

[11] http://www.mingw.org/

[12] J. Bahuguna, R. G. Chittawadigi, and S. K. Saha. “Teaching and learning of robot kinematics using RoboAnalyzer software,” In *Proceedings of Conference on Advances In Robotics*, 2013.