Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 133 (2018) 660–667

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Robotics and Smart Manufacturing (RoSMa2018)

Kinematic Analysis and Development of Simulation Software for Nex Dexter Robotic Manipulator

Amogh Patwardhan, Aditya Prakash, Rajeevlochana G. Chittawadigi*

Department of Mechanical Engineering, Amrita School of Engineering, Bengaluru, Amrita Vishwa Vidyapeetham, India

Abstract

Industrial robots are used extensively in manufacturing units for various tasks. Most of the robot manufacturers provide software for offline and online programming of robots, but they are usually sold separately. Few robot manufacturers do not have a dedicated simulation software for their robots. In this paper, a methodology to create CAD model of an existing physical robot is described by taking the example of Nex Dexter 5-axis robotic manipulator. Later, the Denavit-Hartenberg (DH) parameters were extracted using a methodology reported elsewhere. By using the DH parameters, few CAD files were exported. A Teach Pendant Application has been developed using Visual C# that can connect to Virtual Robot Module of RoboAnalyzer for visualization of robot motion. The kinematic analysis of the robot was formulated for joint and Cartesian motion of the robot. The application was then integrated with the physical Nex Dexter robotic manipulator. Hence, robot motion intended for the robot is first tested in simulation environment and once it is found suitable, motion of the actual robot takes place. The methodology proposed is generic and can be used to simulate any robotic manipulator.

*Keywords:*robot simulation; industrial robots; Cartesian motion; robot kinematics

1. Introduction

Robots are primarily classified as mobile and fixed base robots. Mobile robots have a base that moves either on land, water or air whereas fixed base robots have one of its parts fixed to ground. Fixed base robots are further classified as serial systems, parallel systems and also as tree type systems, which is a combination of the former two. Parallel robots generally have a moving platform that can have a tool or object. Though parallel robots have better rigidity and accuracy, they have smaller workspaces and are subject to interlocking of the robot links. On the other hand, serial

* Corresponding author. Tel.: +91-80-25183700; fax: +91-80-28440092.
E-mail address: rg_chittawadigi@blr.amrita.edu

10.1016/j.procs.2018.07.101

robots have larger workspaces though they are less accurate. In industries, serial robots are used more extensively and are referred to as industrial manipulators.

Industrial robots or manipulators are generally used to perform tasks such as assembly, grinding, pick and place, painting, welding, etc. The robots are expected to perform the tasks in Cartesian space where as they are controlled in their joint space. Hence, a mathematical model correlating the joint motion and the motion of the end-effector, the last link of the robot, has to be known to achieve desired robot motion. The mathematic model, known as kinematic model, is generally derived using the defacto standard Denavit-Hartenberg (DH) parameters. Hence, for a given robot, one should know its DH parameters before proceeding ahead with its kinematic analysis and motion.

Robots are used in environments with static and dynamic obstacles. Also, if the motion of the robot is checked in a simulation software before being programmed on an actual robot, the efforts and time will be reduced significantly. Also, simulation can avoid any possible damage of the physical robot. Hence, a simulation software for a robot is of great importance. Robot simulation software are broadly classified into offline and online simulation. In the former, the CAD model of robot and its workcell is shown to the user. Any robot motion in the form of joint and Cartesian jogging can be provided as input and the 3D CAD model of the robot arm moves. Once the desired motion is achieved, the robot motion or the program can be sent to actual robot controller. During the simulation, the actual robot is not disturbed and hence robot's productivity is not affected during the simulation. However, any dynamic obstacles in the workcell can not be identified and accordingly alter robot motion, which is done in online simulation. Hence, for environments with dynamic objects such as a human co-worker or for multiple robots sharing common workspace, online simulation should be performed.

Several robot simulation software exist which allow offline and online simulation of robots. Some of the commercial ones include ABB RobotStudio, KUKA Sim Pro, RoboDK library, Visual Components, etc. A few free or opensource alternatives exist such as ROBOMOSP[1], V-REP, etc. Another such software is RoboAnalyzer[2], a 3D model based robotics learning software which is available for free, co-developed by the third author. It has several modules related to forward kinematics, inverse kinematics, inverse and forward dynamics, and Cartesian motion planning of serial robots. Its Virtual Robot Module (VRM) has CAD models of more than 20 industrial robots and can perform joint and Cartesian motion on them. The architecture of VRM is kept such that newer models of robots can be added with ease and simulated readily. RoboAnalyzer software has several advantages and a few disadvantages over other similar software, as reported in [2]. Due to space restrictions, they have not been listed down here.

In this paper, a methodology for the development of offline and online simulation software for a custom robot is proposed. These methodology would help a robot manufacturer to use RoboAnalyzer platform to develop a custom robot simulation software. Section 2 explains the steps to be followed in the modeling of 3D CAD assembly of a physical robot, Nex Dexter 5-axis robotic manipulator in this case. In Section 3, the methodology to determine or extract the DH parameters from the CAD model of a robot and preparation of 3D models for simulation software are presented. Thereafter, the formulations for the kinematic analysis of the robot is described in Section 4. A client application, acting as an online simulation software, is presented in Section 5 followed by the conclusions.

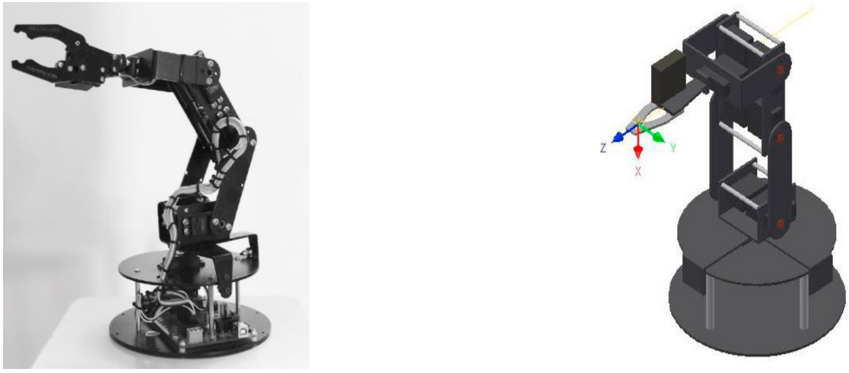
2. Modeling of 3D CAD Assembly of Robot

The CAD assembly files of industrial robots from standard robot manufacturers such as ABB, Fanuc, KUKA, etc., are generally available in the respective manufacturer's website and can be used to import in Virtual Robot Module after making certain modifications. However if the CAD model of a robot is not available, the model has to be developed manually by the end-user. One such robot that has been taken as an example in this paper is Nex Dexter 5-axis robotic manipulator, shown in Fig. 1(a). The methodology proposed in this paper is generic and any physical robot can be modeled and simulated in VRM. The steps followed in modeling of CAD assembly of the robots are below:

- Measure the important dimensions of each robot link
- Model each robot link as a CAD part in Autodesk Inventor software. Any other CAD software can also be used for modeling. However, certain steps followed in this this methodology have to be executed manually in the other software.

- Measure the distance between consecutive joint axes. These dimensions find their way in the DH parameters of the robot, which are determined at a later stage. The measured dimensions should be validated with the DH parameters found.
- Insert the CAD part of each link in Autodesk Inventor CAD assembly. The base part (Link 0) should be placed such that the bottom plane of the component is on the Assembly *XY* plane and the first joint axis is along the Assembly *Z* axis.
- Define revolute joints (Insert Constraint) between the consecutive robot links. While defining the constraints between the robot parts, the first part to be selected should always be the link that is towards the base part and the second selection should be the part towards the last link (end-effector). For example, if Insert Constraint 1 is to be created, the first selection should be base part (Link 0) and the second selection should be Link 1.
- Place a UCS (User Coordinate System), a marker to represent *X*, *Y* and *Z* axes of a coordinate system, inside the end-effector link. The UCS added should be such that its Z axis is parallel to the direction of the last joint axis. Also the origin of the UCS should be located at the center of the gripper or any tool at the end-effector, typically known as TCP (Tool Center Point).

The CAD assembly of the given robot in Autodesk Inventor software is shown in Fig. 1(b). The assembly should have only n+1 parts (n=degree-of-freedom or the number of joint axes) and have only n Insert Constraints. These Insert Constraints should also be defined from the base link to the end-effector. Any change in the order will fail the DH parameters extraction methodology explained in the next section.



(a) Physical model

(b) CAD Assembly in Autodesk Inventor

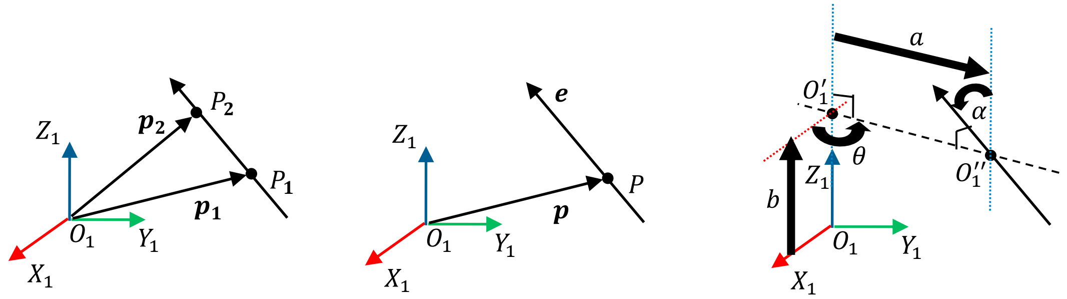
Fig. 1. Nex Dexter 5-axis robot arm

3. DH Parameters of Robot

The Denavit-Hartenberg (DH) parameters are generally used to represent a robot architecture. For the sake of completeness of the paper, a very brief explanation of the DH parameters is given below.

A line can be represented in a 3D coordinate system in various ways. One way is to represent it as position vectors ($\boldsymbol{p_1}$and $\boldsymbol{p_2}$) of two points (P_1 and P_2) on the line, requiring a total of 6 coordinates, as illustrated in Fig. 2(a). Another way is to represent using position vector ($\boldsymbol{p}$) of any point (P) on the line and another unit vector ($\boldsymbol{e}$) to represent the direction of the line which also requires a total of 6 parameters, as shown in Fig. 2(b). Denavit and Hartenberg [3] proposed that a line can be represented using just four parameters.

If a common normal is drawn between the *Z* axis of the coordinate system ($X_1Y_1Z_1$) and the line, it would intersect at two points O_1' and O_1'', as shown in Fig. 2(c). They proposed four parameters required to reach the line from coordinate system. Actually, the proposal was for open-loop and closed-loop mechanisms. If a coordinate system is known on a link and the axis of the next joint is represented as a line, four parameters are required to go from the coordinate system to the line. Once the line is reached, a new coordinate system is formed on the next link and the subsequent joint axis is reached using four more parameters. The new coordinate frame will have its *Z* axis along the joint axis, its X axis along the common normal and Y axis obtained using cross-product rule. The origin of the new coordinate system is at O_1''. The four DH parameters are described briefly in Table 1. More details on the assignment of DH frames, with regard to serial robots, can be found in [4].



(a) Two position vectors (b) A position vector and direction (c) Four DH parameters

Fig. 2. Line representation with respect to a Coordinate frame

Table 1. Denavit-Hartenberg (DH) parameters

Parameter	**Description**
Joint Offset (b)	Linear distance from O_1 to O_1' along Z_1
Joint Angle (θ)	Angle from $X_1$to common normal about Z_1
Link length (a)	Linear distance from O_1' to O_1'' along common normal
Twist Angle (θ)	Angle from Z_1 (a line parallel to it) and the line about common normal

3.1. DH parameters from technical drawings

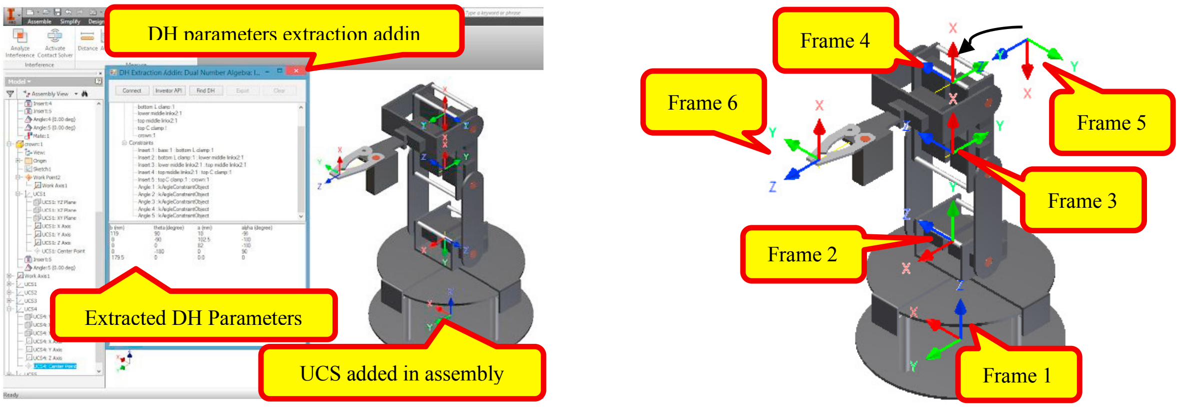
Robot manufacturers generally provide technical drawings of the robot arm in which key dimensions such as distance between joint axes and other information are provided. By following the steps mentioned in [4], one can determine the DH parameters of the robot at hand. An analytical methodology to automatically extract the DH parameters from a CAD model of the robot was proposed in [5], which will be followed in this paper.

3.2. DH parameters extraction from CAD model

To extract the DH parameters of a CAD model of robot arm, the steps followed are below:

- Model the CAD assembly as described in Section 2
- Start the Inventor addin
- The addin reads the joint axes data (point on line and unit vector along the line) and determines the DH parameters using the methodology proposed in [5], using line geometry.
- One UCS (User Coordinate System) is programmatically added to the CAD assembly where the DH frames are to be located.
- These UCS from assembly have to be moved inside each Inventor part file, which is a required step to export the CAD file.

The addin and the Nex Dexter CAD assembly are shown in Fig. 3(a). The DH parameters and the UCS added to the assembly can be seen in the figure as well. Note that the origins of Frame 4 and 5 actually coincide. For the sake of clarity, Frame 5 is displaced. The DH parameters extracted automatically had Z axes of Frame 2 (on Link 1) and Frame 3 (on Link 2) are in opposite direction. Similarly for Frame 3 and Frame 4. Manually, the UCS corresponding to Frame 2 and Frame 4 are flipped to have opposite Z axis as compared to their earlier direction. The modified UCS frames are shown in Fig. 3(b). This is done to get DH parameters which are simpler and the kinematic formulations become easy. The DH parameters determined by the addin after some minor modifications are given in Table 2. These parameters are required for the kinematic analysis and simulation of robot. Same methodology of DH parameter extraction can be followed for 3D CAD model of any serial robot.



(a) Inventor addin and CAD assembly

(b) Modified UCS for simpler DH parameters

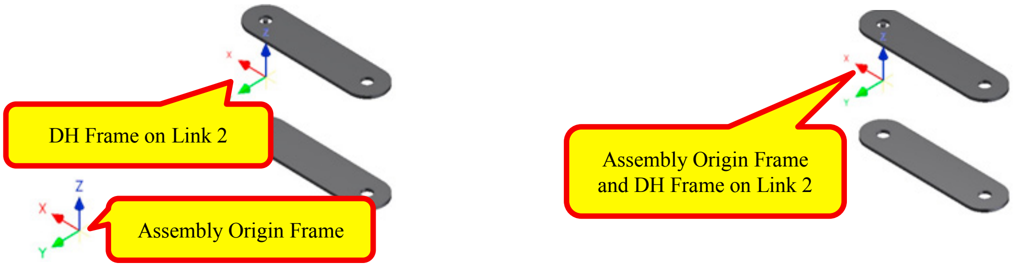
Fig. 3. Automatic extraction of DH parameters using [5]

Table 2. DH parameters of Nex Dexter Robotic Arm [JV: Joint Variable]

Joint	Joint Offset (b_i)	Joint Angle (θ_i)	Link length (a_1)	Twist Angle (α_1)
	(mm)	(°)	(mm)	(°)
Joint 1	119	90 (JV)	10	90
Joint 2	0	90 (JV)	102.5	0
Joint 3	0	0 (JV)	82	0
Joint 4	0	180 (JV)	0	-90
Joint 5	179.5	0 (JV)	0	0

3.3. Preparation of 3D Model for Simulation

The CAD assembly with the UCS is made *n*+1 copies, corresponding to the *n*+1 robot links. Each CAD assembly is separately opened in Autodesk Inventor and all except one part are deleted in the assembly. For example, for Link 2's case, all links except Link 2 are deleted as shown in Fig. 4(a). Note that the UCS attached inside the part of Link 2 is intact and distinct from the Assembly Origin coordinate frame. Using Inventor's assembly constraints, the UCS of Link 2 is constrained with Assembly Origin coordinate frame so that they coincide. This is shown in Fig. 4(b).



(a) Assembly for Link 2

(b) Modified assembly file for Link 2

Fig. 4. Modifications in CAD assembly file for export STL files for each part file

Thereafter, the assembly file is exported as STL file (ASCII format), which can then be read by RoboAnalyzer software. The same steps are repeated for all the robot links. Further, RoboAnlayzer software expects custom robot CAD data and DH parameters data in the form of a XML file (sample can be taken from any existing robot model). The XML file once created using the DH parameters obtained in the previous steps can be used to load Nex Dexter.

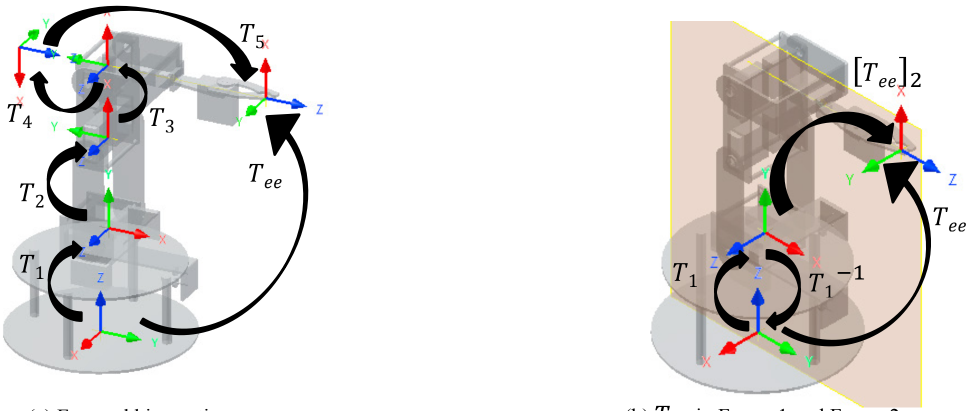
4. Kinematic Analysis

Kinematic analysis of a serial robot can be classified into forward and inverse kinematics. For a given set of joint angles, determination of the configuration (position and orientation) of the end-effector is known as forward kinematics. On the other hand, for a given end-effector configuration, finding the required joint angles is known as inverse kinematics. For serial robots, forward kinematics is straight forward and has unique solutions but the inverse kinematics formulations are only possible for standard robot architecture. Also, inverse kinematics may provide multiple solutions and hence care is required to choose the correct solution.

4.1. Forward kinematics

Referring to Fig. 5(a), the forward kinematic analysis is the solution of the following equation, where T_i and T_{ee} correspond to Homogeneous Transformation Matrix (HTM) corresponding to i^{th} Joint's DH parameters, and that of the End-effector with respect to the base frame (Frame 1 on Link 0), respectively,

$$T_{ee} = T_1 * T_2 * T_3 * T_4 * T_5 \tag{1}$$



(a) Forward kinematics

(b) T_{ee} in Frame 1 and Frame 2

Fig. 5. Homogeneous Transformation Matrices (HTM) for forward and inverse kinematics

4.2. Inverse kinematics

To obtain close form solution for inverse kinematics of the Nex Dexter robot arm, the following steps are followed. Equation 1 is rewritten as

$$T_{ee} = T_1 * [T_{ee}]_2; \qquad \text{where } [T_{ee}]_2 = T_2 * T_3 * T_4 * T_5 \tag{2}$$

$[T_{ee}]_2$ in Equation 2 corresponds to the HTM of the EE with respect to the Frame 2, as shown in Fig. 5(b). Note that this expression will not have any θ_1 terms in it. Also, whenever Joint 1 rotates, a plane containing the End-effector point always lies in the *XY* plane of Frame 2. This is due to the architecture of the robot. Hence, determination of angles of Joint 2, Joint 3 and Joint 4 becomes a planar problem, thus simplifying the calculations. $[T_{ee}]_2$ can be determined as

$$[T_{ee}]_2 = {T_1}^{-1} * T_{ee} \tag{3}$$

where ${T_1}^{-1}$ corresponds to the HTM of Frame 1 with respect to Frame 2. Note that T_1 was earlier obtained by multiplying four individual HTM corresponding to each of the DH parameters for Joint 1. Similarly, ${T_1}^{-1}$ can be obtained by the multiplication of HTMs corresponding to the opposite sign value of each of the DH parameters and the order of multiplication being the opposite. The expression for ${T_1}^{-1}$ is obtained for α_1= 90° as

$${T_1}^{-1} = \begin{bmatrix} cos\,\theta_1 & sin\,\theta_1 & 0 & -a_1 \\ -cos\,\alpha_1\,sin\,\theta_1 & cos\,\alpha_1\,cos\,\theta_1 & sin\,\alpha_1 & -b_1\,sin\,\alpha_1 \\ sin\,\alpha_1\,sin\,\theta_1 & -sin\,\alpha_1\,cos\,\theta_1 & cos\,\alpha_1 & b_1\ cos\,\alpha_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} cos\,\theta_1 & sin\,\theta_1 & 0 & -a_1 \\ 0 & 0 & 1 & -b_1 \\ sin\,\theta_1 & -cos\,\theta_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Note that the last column of the HTM in Equation 4 does not have a *Z* component, which also justifies why the end-effector point will always remain on the XY plane of Frame 2. Let $\mathbf{p} = \begin{bmatrix} P_x & P_y & P_z \end{bmatrix}^T$ correspond to the end-effector point with respect to Frame 1 and $\mathbf{p'} = \begin{bmatrix} P_x' & P_y' & P_z' \end{bmatrix}^T$. The inverse kinematics of 5-axis Nex Dexter robot arm can be simplified in the following manner:

- θ_1 has only effect on P_x and P_y and does not influence P_z. By looking at the geometry, θ_1 can be determined as

$$\theta_1 = atan2(P_y, P_z) \tag{5}$$

- Origins of Frame 2, Frame 3, Frame 4, Frame 5 and Frame 6 always line on a plane (*XY* plane of Frame 2). This plane rotates with respect to *YZ* plane of Frame 1. Hence, kinematics of end-effector point in Frame 2 can be simplified as a planar problem. Also, the rotation of last joint (Joint 5) does not have any effect on the position of the end-effector in Frame 2. The expression for the end-effector point in Frame 2 ($\mathbf{p'}$) is given by

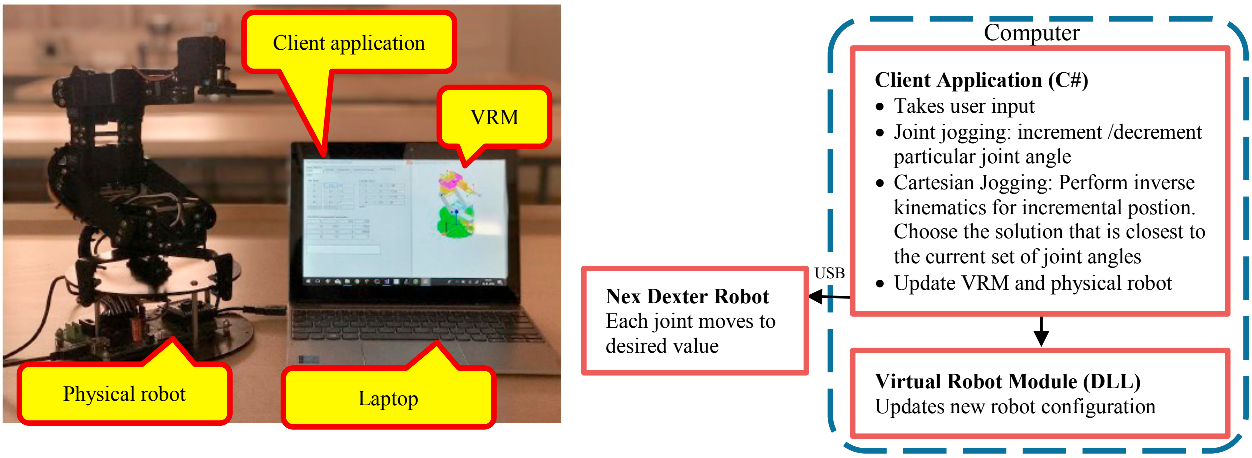
$$\boldsymbol{p'} = {T_1}^{-1} * \boldsymbol{p} => \begin{bmatrix} P_x' \\ P_y' \\ P_z' \\ 1 \end{bmatrix} = \begin{bmatrix} cos\,\theta_1 & sin\,\theta_1 & 0 & -a_1 \\ 0 & 0 & 1 & -b_1 \\ sin\,\theta_1 & -cos\,\theta_1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \tag{6}$$

- Once $\mathbf{p'}$ is determined from $\mathbf{p}$ (which is the actual input for inverse kinematics), the joint angles calculations for θ_2, θ_3 and θ_4 can be performed similar to a planar 3R (R: Revolute) serial robot, as derived in [4]. θ_5 would correspond to the rotation of the last joint and can be one of the Euler angles for the end-effector configuration. To simplify the formulation, θ_5 can be assumed some constant value. Due to space restrictions, the complete inverse kinematics formulations are not included in this paper. The inverse kinematics equations were initially developed as a MATLAB/Octave program. The results obtained were compared with those obtained from RoboAnalyzer software, for the same DH parameters and input values of end-effector configuration (position and orientation), thus validating the formulations derived from scratch.

5. Simulation Software for Nex Dexter

The CAD model of Nex Dexter robot arm developed as described in Sections 2 and 3 was imported in Virtual Robot Module (VRM) of RoboAnalyzer. The VRM has a feature that it can integrated with other software or applications such as MATLAB or MS Excel and act as a visualizer for robot motion [6]. Similarly, Nex Dexter robot in VRM can be simulated by sending joint angles from another client application developed in C#. The client application has the inverse kinematics formulations derived in the previous section and has capability of performing joint and Cartesian jogging of the robot in the VRM. The client application also been integrated with actual Nex Dexter robot arm using a USB (serial communication) cable. The motion given to the VRM is also sent to the physical robot. The client application and VRM running on a laptop its integration with physical robot is illustrated in Fig. 6.

It was observed that joint jogging of physical robot worked as expected but the Cartesian jogging of physical robot was not satisfactory. This was because the motors of the robot have a resolution of 1° and any decimal value sent to them are rounded off. However, development of the client application was an experience that can be used to develop similar software for other robots with better angle resolution.


Client application
VRM
Physical robot
Laptop
Computer
Client Application (C#)
• Takes user input
• Joint jogging: increment /decrement particular joint angle
• Cartesian Jogging: Perform inverse kinematics for incremental postion. Choose the solution that is closest to the current set of joint angles
• Update VRM and physical robot
USB
Nex Dexter Robot
Each joint moves to desired value
Virtual Robot Module (DLL)
Updates new robot configuration

(a) Implementation (b) Overview of the system architecture

Fig. 6. Online simulation of Nex Dexter robot arm/manipulator

6. Conclusions

A methodology to develop kinematic model and simulation software for a physical robot is proposed in this paper. First, the CAD model is developed in Autodesk Inventor followed by automatic extraction of its DH parameters. An example of Nex Dexter robotic arm is used to demonstrate the proposed methodology. Using the CAD files of each part, extracted DH parameters and kinematics formulation, a simulation software has been developed and reported. The software has capability to perform joint and Cartesian jogging in the simulation environment as well as on the real robot. The same procedure can be followed to develop online and offline simulation software for any physical robot.

Acknowledgements

The authors would like to acknowledge Dr. Anant Malewar of Nex Robotics Pvt. Ltd., Mumbai, India for providing the robot arm and also for helping in understanding how it can be controlled from a computer.

References

[1] Jaramillo-Botero, Andres, Antonio Matta-Gomez, Juan Fernando Correa-Caicedo, and Wilber Perea-Castro. (2006) "Robomosp." *IEEE Robotics & Automation Magazine* **13(4)**: 62-73.

[2] Othayoth, Ratan S., Chittawadigi, Rajeevlochana G., Joshi, Ravi P., and Saha, Subir K. (2017) "Robot kinematics made easy using RoboAnalyzer software." *Computer Applications in Engineering Education* **25(5)**: 669-680.

[3] Denavit, Jacques., Hartenberg, Richard S., (1955) "A Kinematic Notation for Lower-pair Mechanisms Based on Matrices." *ASME Journal of Applied Mechanisms* **22(2)**: 215-221.

[4] Saha, Subir Kumar. (2014) "Introduction to Robotics." Second edition, Tata McGraw Hill Publications, New Delhi, India.

[5] Rajeevlochana, C. G., Saha, Subir K, Kumar, Shivesh. (2012) "Automatic extraction of DH parameters of serial manipulators using line geometry." In Proceedings of the 2nd International Conference on Multibody System Dynamics.

[6] Sadanand, Ratan., Chittawadigi, Rajeevlochana G., Joshi, Ravi P., and Saha, Subir K. (2015) "Virtual robots module: an effective visualization tool for robotics toolbox." In Proceedings of the 2nd International Conference on Advances in Robotics.