ISSN (Online) 2456-1290

International Journal of Engineering Research in Mechanical and Civil Engineering (IJERMCE)

Vol 6, Issue 5, May 2021

Kinematics Analysis of the Industrial Robotic Arm Mechanism

[1] Mani Sanguri, [2] Saurabh Singh

[1][2] Department of Mechanical Engineering, Motilal Nehru National Institute of Technology Allahabad, India
Email: [1]mani.20183027@mnnit.ac.in,[2] saurabhsingh@mnnit.ac.in

***Abstract*--- Robotic Manipulators are very powerful systems used in almost all industries today to perform different tasks much more precisely and effectively than a human can. One such robotic manipulator is the robotic arm mechanism that is heavily used in all industries, especially the manufacturing industries, due to their greater efficiencies and safer alternatives to humans. For performing the tasks, the most important consideration is getting the gripper of the robotic arm to a required position and orientation. The aim here is to perform the kinematic analysis of the robotic arm mechanism by modeling the forward and inverse kinematics of the arm mechanism. The designing of the movement flow plan and the evaluation of the DH parameter is done for calculating the desired position and orientation of the end effector. Here, the forward kinematics is easy to model, but the inverse kinematics modeling will involve the use of the traditional methods that include the DH notations, transformations, iterations, etc.**

***Keywords*--- Forward Kinematics, Inverse Kinematics, Mechanical Joints and Links, Degree of Freedom**

I. INTRODUCTION

The industry is moving towards complete automation, and the need for Manpower is decreasing day by day [9]. We can consider an Automobile Industry, where many changes have taken place in the last 10-20 years. The industry's current situation requires extensive use of the Robotics arm to automate any work to increase productivity and decrease errors ultimately [10]. These changes have introduced us to the two most popular technologies, called industrial robotics arm and exoskeletons. Our work under this article focuses only on the Industrial robotics arm but can be extended to exoskeletons.

A robotics arm is defined as an arm that is the same as a mechanical arm, which is programmed to control specific Motions or behavior to accomplish a given task [19]. They can be imagined as a human arm with certain constraints or freedom, which perform functions along a given path. During programming a robotics arm, the path is fixed. It means that the initial and final point of the gripper is fixed, and it can do any repetitive task until there is a change in the code. The arm consists of different joints that have the flexibility of Motion in different directions. During our coding and all the analysis, we define the Motion of the arm about joints only. The gripper connected to the end of the arm is called end effectors, and its Motion is also controlled by the code written in software. There are other parts of the arm, which have various functions. It consists of a base on which the whole arm stands, the shoulder, the elbow, the wrist, and finally, the end effector, called a gripper arm.

One of the crucial concepts in Robotics is the degree of freedom, which is the constraint of the motion [18]. If we talk about the mechanical context, the degree of freedom is the number of independent variables required to define the Object [2]. Although we have two dimensions or even three dimensions, the degree of freedom can be more than 3. Consider a robotics arm, which is designed to work as a human arm [1]. The shoulder motion can define either the Yaw motion or the pitch motion, which is left and right or up and down motion, respectively. The wrist can do only pitch or yaw motion, and finally, the elbow can only perform pitch motion. The wrist and shoulder can also make the Rotation motion (Roll). Summing all the motion of different joints gives the value of 5 to 7. Hence, the given robotics arm has a degree of freedom between 5-7. The end effector also has certain degrees of freedom added to the Overall value to define the final degrees of freedom for the complete Industrial Robotics arm.

Robotic arms have joints with different motions, characterized based on the type of motion they produce. It can create five types of Mechanical joint motion [3]: Linear joint, Orthogonal joint, Rotational joint, Twisting joint, and revolving joint. We will be explaining each of them one by one in subsequent paragraphs.

II. OBJECTIVE OF ANALYSIS

We are currently in Industry 4.0, and it becomes essential

ISSN (Online) 2456-1290

International Journal of Engineering Research in Mechanical and Civil Engineering (IJERMCE)

Vol 6, Issue 5, May 2021

to have a good understanding of Robotics and exoskeleton, especially for Mechanical Engineering.

The main objective of the analysis is as follows:

1. Understanding the use of Robotics arm and how the motion is implemented
2. Get familiar with different software that is used for the Kinematic analysis, which includes V-REP and MATLAB and SIMULINK
3. Explore different types of Robotics arm available in the V-REP library and develop the motion

III. KINEMATIC MODEL

Kinematic analysis of Robotics arm requires the Input from the motion of the joint to produce the required motion on the arm, or Input from the motion of the arm to give the needed motion to the Joints [14]. In Mechanics, Kinematics deals with velocity and acceleration of the body and has no relation with the force and torque induced because of the motion. The same is the case for the Robotics arm, where we will be focusing on the motion of joints instead of focusing on the force and torque produced due to the motion (which comes under dynamic analysis).

The definition of a Coordinate system becomes very important as the involvement of velocity and acceleration need to define the coordinate system about which the motion is taking place. Here, we use the Right-Hand coordinate system, which is defined with the configuration of our right hand. For example, we know the direction of the X-axis and Y-axis and want to determine the direction of Z-axis. In that case, we have to use our right hand and point the finger so that all the three fingers (defining the three coordinate axes) are perpendicular to each other. We can select any two of them as X-axis and Y-axis, and the third one will be the Z-axis. The defined axes constitute the right-hand coordinate system.

The most fundamental equation used in the kinematic Analysis is the kinematic equation formed by the chain (kinematic chain) that makes the robot. The equation is non-linear and can't be solved directly through analytical methods. We need to use different software(s) to simulate the motion of the robotic arm with the input from the joint angle and joint velocity. Kinematic Analysis is divided into two broad classifications, i.e., Forward kinematics and inverse kinematics. The working of both models is the same, but the input and output differ, along with the governing equation of motion.

The time derivate of the kinematic equation of the robotics arm gives the Jacobian, which establishes a relation between the Joint rate to the angular and linear velocity of the end effector (gripper) [5]. Deriving from the principle of virtual work, it can be shown that the Jacobian also establishes the relationship between the Joint torque, its resultant force, and torque by the end effector [16]. We can also identify the singularity (singular configuration) from the Jacobian. Let q be the column matrix that represents the join velocities, let 'n' is the number of joints. Hence, the size of the matrix is 'n*1' [12], [13].

Let X represents the column matrix for end-effector velocities, let 'm' is a number defined as 6 for spatial robots and 3 for planar robots. Hence the size of the matrix is 'm*1' [11].

Let J defines the Jacobian matrix for the given configuration according to joint velocities and end effector velocities [4], [15].

$$X=J*q \qquad (i)$$

If the order of matrix q is 'n*1' and the order of X is 'm*1', then the order of J is 'm*n [6].

Forward kinematics computes the value of the position of end-effectors from the input value of joint parameters, which includes joint angle and joint velocities [7]. The body Jacobian, which consists of speed Jacobian and rotational velocity Jacobian matrix, gives the relationship between the two [17]. This consists of a non-linear equation and hence, there is no analytical solution available for each type of motion. Inverse kinematics computes the value of Joint parameters, from the input value of end-effectors parameters [8].

In this paper, we will be making a forward kinematic model as well as an Inverse kinematic model to develop the motion of the robotics arm to complete a task. MATLAB and SIMULINK will be used to develop the model from the design data received from either the SOLIDWORKS or the V-REP software [20].

IV. METHODOLOGY

In this paper, the kinematic analysis and simulation of the robotic arm are done using two methodologies. In the first methodology, we use a robotic arm template already present in the V-REP environment for simulation. The first methodology uses the V-REP environment for the simulation and the MATLAB environment for controlling the mechanism and the second methodology uses SOLIDWORKS for designing the CAD model, Simscape Multibody environment for the simulation, and the MATLAB environment for controlling the mechanism.

A. V-REP AND MATLAB SIMULINK INTEGRATION

In this methodology, the KUKA-KR16 robotic arm model template is used that is already present in the V-REP environment. The Virtual Robot Experimentation Platform

ISSN (Online) 2456-1290

International Journal of Engineering Research in Mechanical and Civil Engineering (IJERMCE)

Vol 6, Issue 5, May 2021

(V-REP), is an environment that is used for simulating the 3D robotic systems that help you in modeling, editing, programming, and simulating any of the robotic systems. It can help in the modeling and simulation of the sensors, mechanisms, robots, and whole systems in a variety of ways. V-REP can be used for a variety of purposes ranging from remote monitoring, hardware control, fast prototyping and verification, fast algorithm development/parameter adjustment, safety double-checking, robotics-related education, or factory automation simulations. The three central elements here include: Scene Objects, Calculation Modules, and Control Mechanisms.

Here the control mechanisms consist of the Embedded Scripts, Plugins, and Add-ons that are used for a local interface where we have the same process or hardware; and the Remote API clients and ROS nodes are used as external interfaces where we do not have the same processor hardware. Remote API clients consist of over 100 API functions (Extendable), it has a client program (your program), a server relationship (V-REP plugin), and the programming language that can be used are C/C++, Python, Java, MATLAB, Octave, Lua & Urbi.

In this methodology, MATLAB is used as a Remote API client which is used as an external interface to control the working of the robotic system. In this model, there is a proximity sensor placed near the conveyor belt that will be used to sense whether the block has reached the position, where the robotic arm will be able to lift it from the conveyor belt and place it on the table. As soon as it senses that block has reached that position, the robotic arm will move such that the gripper will be just above the block and will be able to pick it from the belt. The steps involved in controlling the KUKA KR-16 robotic arm include the following:

1. Setting up the simulation environment in V-REP as shown in Figure 1.
2. Making required changes to the LUA scripts of the ROBOTIQ_85 (gripper), the Customizable Table, and the Customizable Conveyor. The changes need to be done in Non-Threaded Child Script for ROBOTIQ_85 in the initialization function, Non-Threaded Child Script for the Customizable Table, and Non-Threaded Child Script for the Customizable Conveyor.
3. The file named remApi.m, remoteApiProto.m, simpleTest.m and remoteApi.dll should be extracted from the: C:\ProgramFiles\V-REP3\VREP_PRO_EDU\programming\remoteApiBindings\matlab\matlab.
into the folder that contains your simulation files.

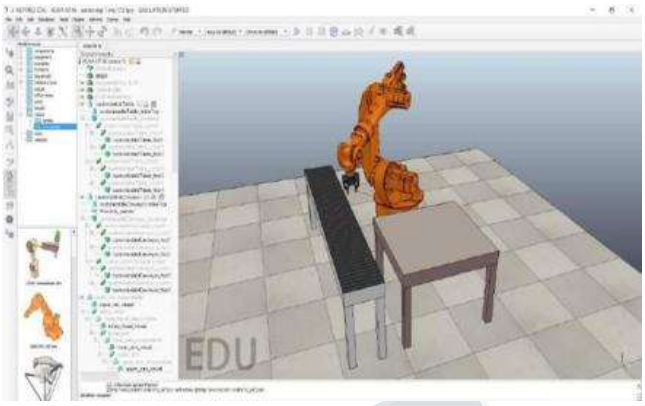


Figure 1: Simulation environment in the V-REP Software

4. Writing the MATLAB function for the picking and placing of the blocks, moving the arm, and gripping the block and the MATLAB code for integrating these functions and getting the task done.
5. Run the V-REP Simulation and the simpleTest MATLAB code to check for any faults.
6. Run the V-REP Simulation and the MATLAB code for carrying out the task.

B. SOLIDWORKS AND MATLAB-SIMULINK INTEGRATION

There will not be templates available for all the robotic arm mechanisms that are being used in the industry because of which there will be a need to design a 3-D model of the robotic arm mechanism using any of the available CAD software and then convert it into a Simulink model using the Simscape Multibody Link which is used for the kinematic analysis. Here the Simulink acts as the simulation platform and MATLAB acts as the interface used to control the movements of the mechanism. The CAD software used is SOLIDWORKS. Here, both forward and inverse kinematic analysis are performed on the model by developing suitable Simulink model and MATLAB codes. A detailed explanation of the steps is given below:

1. Designing a 3-D model of the robotic arm mechanism using the CAD software as shown in Figure 2.

ISSN (Online) 2456-1290

International Journal of Engineering Research in Mechanical and Civil Engineering (IJERMCE)

Vol 6, Issue 5, May 2021

Figure 2: 3-D Model of the Robotic Arm Mechanism

2. Exporting it as an .xml file from SolidWorks to the Simscape Multibody Link Platform.
3. Using the MATLAB software, importing it from .xml to .slx format to the Simulink platform with the command smimport('Name of the file here').
4. In the Simulink platform, take the revolute joints you need to control and form a subsystem of the rest of the joints that are not required and name it 'Extra System' as in Figure 3.
5. To the revolute joints that are required, add suitable blocks for controlling the joints and getting the desired output from them. Fig. 4(a) and 4(b) shows how a model will be derived for the kinematic analysis

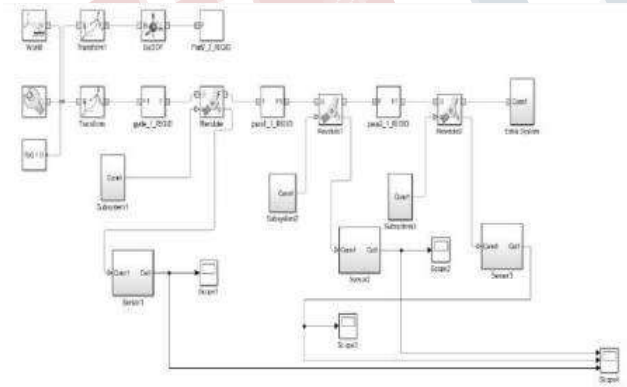
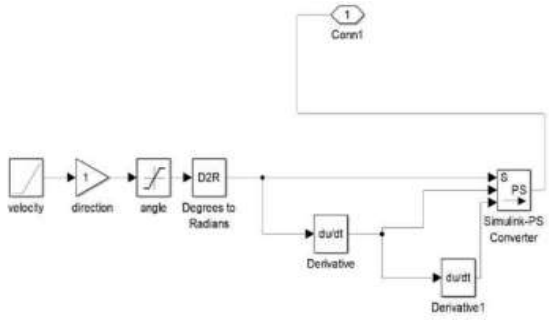
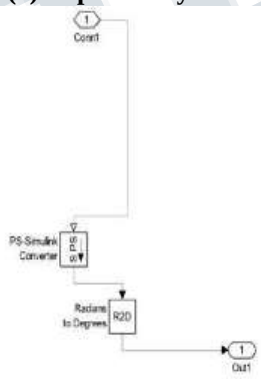


Figure 3: Simulink Model for the Robotic Arm

6. For the Forward Kinematic Model, the MATLAB code is developed that takes the input of the joint angle and the joint velocities and then computes the transformation matrix, and hence the tait-bryan angle is computed using this transformation matrix.
7. For the Inverse Kinematic Model, the MATLAB code is developed that takes the input of the required rotation of the total robotic arm (tait-bryan angle) along the X, Y, Z axes and the joint velocities and computes the rotation matrix which is then used to compute the joint angles.
8. Finally, the simulation of the robotic arm is done by using the velocity and angle of each joint, and the output is viewed in the Simulink Platform.



(a). Input Subsystem



(b). Output Subsystem

Figure 4: Subsystem of Simulink Multibody

C. ALGORITHM DEVELOPED FOR THE ANALYSIS

The forward kinematic model involves taking the joint angles and the joint velocities as user input and computing the tait-bryan angles, the transformation and, the rotation matrix from the obtained data. The following is the algorithm used for developing the MATLAB code that is used for performing the forward kinematic analysis on the robotic arm mechanism:

1. Set both the rotation and the matrices as an identity matrix.
2. Enter the joint angles as a user input
3. Calculate the transformation matrix using the joint angles
4. Enter the joint velocities and the order of rotation of the x, y, and z-axes as a user input
5. Calculate the tait-bryan angle using the transformation matrix
6. Calculate the rotation matrix using the tait-bryan angles and the order of rotation
7. Feed the joint angle and the joint velocity thus obtained

ISSN (Online) 2456-1290

International Journal of Engineering Research in Mechanical and Civil Engineering (IJERMCE)

Vol 6, Issue 5, May 2021

into the simulation environments to get the desired movement of the arm.

The inverse kinematic model involves taking the tait-bryan angles and the joint velocities from the user and using these to compute the joint angles, the rotation matrix, and the transformation matrix. The following is the algorithm used for developing the MATLAB code that is used for performing the inverse kinematic analysis on the robotic arm mechanism:

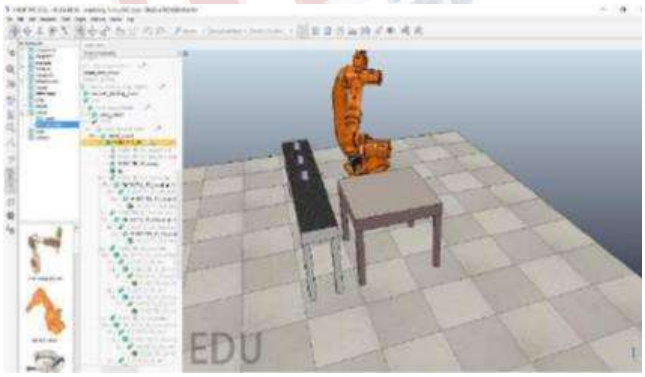
1. Set both the rotation and the matrices as an identity matrix.
2. Enter the joint velocities and the tait-bryan angle as a user input
3. Enter the order of rotation of the co-ordinate axes
4. Calculate the rotation matrix from the data obtained from the user
5. Calculate the joint angles using the rotation matrix obtained
6. Calculate the transformation matrix from the joint angles
7. Feed the data thus obtained into the simulation environments to get the desired movement of the arm.

V. ANALYSIS RESULTS

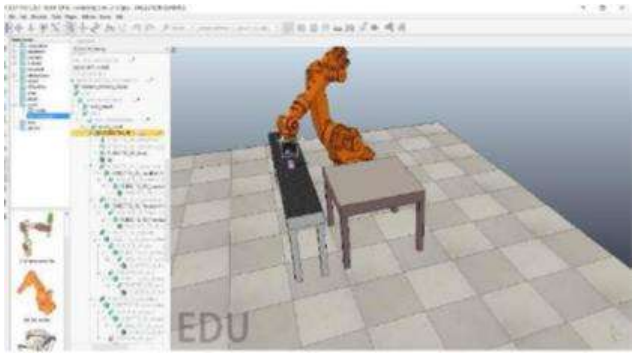
The Analysis carried out in MATLAB and SIMULINK through this methodology results in the simulation of the robotics arm and it performs the motion according to the given input.

A. RESULTS FROM THE FIRST METHODOLOGY

The following is the image of the simulation of the KUKA KR-16 Robotic Arm model on the V-REP environment where the proximity sensor has sensed the block and the gripper is picking up the block.



(a). Initial Position before the Simulation

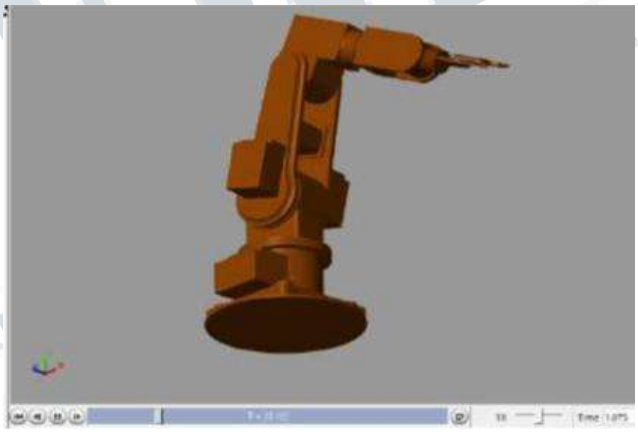


(b). Final Position after the Simulation

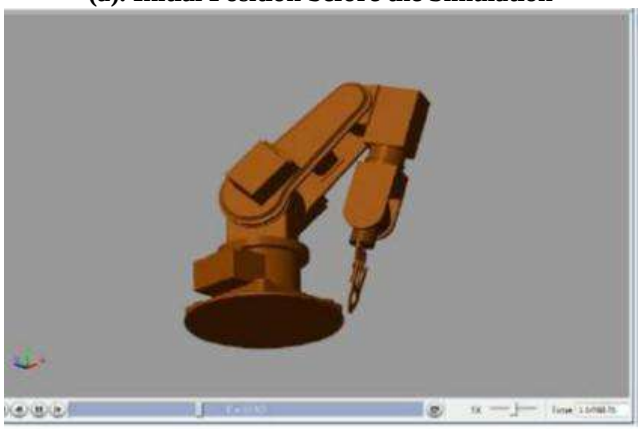
Figure 5: Position of the KUKA KR-16 Robotic Arm

B. RESULTS FROM SECOND METHODOLOGY

The following is the image of the simulation of the robotic arm mechanism on the SimMechanics window of the MATLAB Simulink platform where the angles and velocities of each joint will be used to simulate the mechanism.



(a). Initial Position before the Simulation



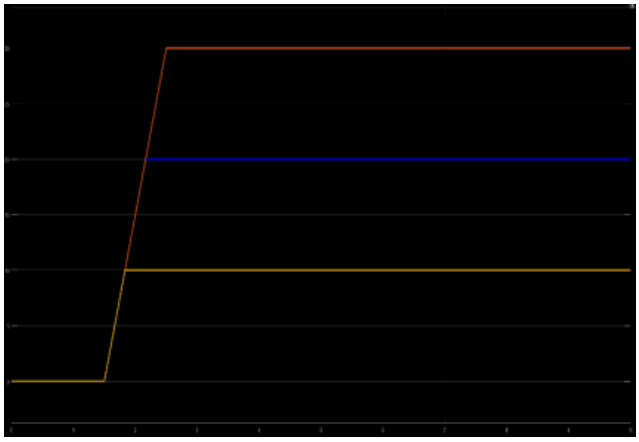
(b). Final Position after the Simulation

Figure 6: Position of the Robotic Arm Mechanism

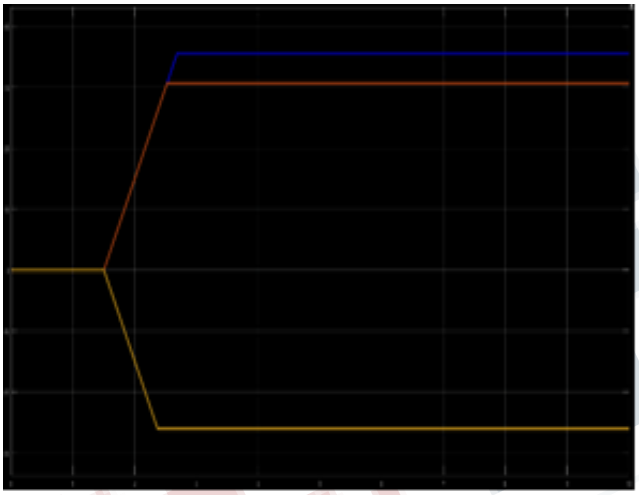
ISSN (Online) 2456-1290

International Journal of Engineering Research in Mechanical and Civil Engineering (IJERMCE)

Vol 6, Issue 5, May 2021



(a). For the Forward Kinematics



(b). For the Inverse Kinematics

Figure 7: Graphs of the Joint Angle vs Time

C. RESULTS FROM MATLAB CODE

The following are the results obtained from the codes used for the calculations involved in performing the forward and inverse kinematic analysis of the robotic arm mechanism.

```
>> Inverse_Kinematics
Enter the order of rotation
 1. X-Y-Z 2.X-Z-Y
 3.Y-X-Z 4.Y-Z-X
 5.Z-X-Y 6.Z-Y-X
1
Enter the rotation in the x-direction 20
Enter the rotation in the y-direction 30
Enter the rotation in the z-direction 10

rotmatrix =

    0.8529    0.0052    0.5221
    0.1504    0.9551   -0.2552
   -0.5000    0.2962    0.8138


Rotation matrix created!

transmatrix =

    0.8529    0.0052    0.5221
    0.1504    0.9551   -0.2552
   -0.5000    0.2962    0.8138

Enter the joint velocity 1 30
Enter the joint velocity 2 30
Enter the joint velocity 3 30
The joint angles are computed!
The angle of joint 1 is:-2.605239e+01
The angle of joint 2 is:3.553135e+01
The angle of joint 3 is:3.064234e+01
```

(a). Forward Kinematics

```
>> Forward_Kinematics
Enter the joint angle 1 in degrees 10
Enter the joint angle 2 in degrees 20
Enter the joint angle 3 in degrees 30

transmatrix =

    0.7146   -0.6131    0.3368
    0.6337    0.7713    0.0594
   -0.2962    0.1710    0.9397

Enter the joint velocity 1 30
Enter the joint velocity 2 30
Enter the joint velocity 3 30
Enter the order of rotation
 1. X-Y-Z
 2.X-Z-Y
 3.Y-X-Z
 4.Y-Z-X
 5.Z-X-Y
 6.Z-Y-X
1
The Tait-Bryan angles are calculated!
The tait bryan angles thus computed are:
1.031410e+01, 1.722940e+01, 4.156670e+01
```

(b). Inverse Kinematics

Figure 8: Calculations involved in the Kinematic Analysis

VI. CONCLUSION

The robotic kinematics involves the movement of the gripper to a required position through the movement of the links and joints present in the mechanism. Robotic Kinematics can be divided into two types include the forward and the inverse kinematics. Here, the forward and the inverse kinematics are analyzed for a robotic arm mechanism. Forward kinematics is the solving of the forward transformation equation for finding the hand location from the angles and velocities of the joints and

Inverse kinematics is the solving of inverse transformation equation for finding the joint angles from the gripper location in 3-D space. For this we have used two methodologies, first involves the V-REP and MATLAB Simulink integration and the second involves the SOLIDWORKS and MATLAB Simulink integration. In the first methodology, then we take the desired user input and perform the calculations as per the kinematic model using MATLAB, an interface is formed between MATLAB and V-REP through which the output is fed to V-REP and the desired motion is obtained by running the simulation. In the second methodology, the CAD model of the robotic arm mechanism is built in SolidWorks and exported to the Simulink platform and a basic model for the mechanism is developed in which we make the required modification for running the forward and inverse kinematic analysis. We take the desired user input and perform the calculations as per the kinematic model using MATLAB. The desired motion of the robotic arm mechanism is obtained by feeding this input to the Simulink block system and running the simulation.

REFERENCES

[1] Manoel Carlos Ramon, Assembling and Controlling a Robotics Arm, SpringerLink, CA, USA
[2] Madhiha farman, Muneera Al-Shaibah, Zoha aoaiath, Firas S. Jarrar, Design of three degrees of freedom Robotics Arm, Internation Journal of Computer applications
[3] Groover M.P (2008), Automation, Production Systems, and Computer Integrated Manufacturing, 3rd edition
[4] Kevin M. Lynch, and Frank C. Park, Modern Robotics and control, Cambridge University press
[5] Professor Wei Zhang, Velocity kinematics and Jacobian, Ohio state University, Coloumbus, Ohio, USA
[6] Robotics Dynamics Lecture notes, Robotics Lab, ETH Zurich, HS 2017
[7] Jean-pierre Merlet, Fifth International Conference on Advanced Robotics 'Robots in Unstructured Environments 1991
[8] Somchart chokchaitam, International Association of Computer Science and Information technology, 2009
[9] Zhang Ruishu, Zhang Chang, Zheng Weigang. The status and development of industrial robots, IOP Conf. Series: Materials Science and Engineering 423 (2018) 012051
[10] Linn D. Evjemo1 & Tone Gjerstad & Esten I. Grøtli & Gabor Sziebig. Trends in Smart Manufacturing: Role of Humans and Industrial Robots in Smart Factories. Collection of Robotics in Manufacturing. Current Robotics Reports (2020) 1:35–41
[11] ĽUBICA MIKOVÁ, RÓBERT SUROVEC, ERIK PRADA, MICHAL KELEMEN. Mathematical Model of Mobile Robot for the Path tracking of the Robot, JOURNAL OF INTERDISC IPLINARY RESEARCH. 146-147.
[12] F. Solc, B. Honzik. Modelling and control of Soccer Robots. 7th International Workshop on Advanced Motion Control. Proceedings (Cat. No.02TH8623), IEEE Xplore
[13] Randall Bisha, SanjayJoshia, JeffreySchank, J asonWexlera. Mathematical modeling and computer simulation of a robotic rat pup. Elsevier. Volume 45, Issues 7–8, April 2007, Pages 981-1000.
[14] Fikrul Akbar Alamsyah. The Kinematics Analysis of Robotic Arm manipulators Cylindrical Robot RPP Type for FFF 3D Print using Scilab. IOP Conf. Series: Materials Science and Engineering 494 (2019) 012100.
[15] Reza Yazdanpanah A. Geometric Jacobians Derivation and Kinematic Singularity Analysis for Smokie Robot Manipulator & the Barrett WAM. 5th International Conference on Robotics and Mechatronics (ICROM), Tehran, Iran, 2017.
[16] Javad Sovizi , Aliakbar Alamdari , Venkat N. Krovi. A Random Matrix Approach to Manipulator Jacobian. ASME Digital Collections, Paper No: DSCC2013-3950, V003T39A005; 10 pages.
[17] Xin Yaxian, Hong Zhen, Li Bin, Li Yibin. A comparative study of four Jacobian matrix derivation methods for quadruped robot. 2015 34th Chinese Control Conference (CCC), IEEE Xplore, 14 September 2015.
[18] Adelhard Beni Rehiara. Kinematics of AdeptThree Robot Arm. Submitted: October 22nd 2010, DOI: 10.5772/17732.
[19] MIT Open Courseware : Introduction to Robots, Lecture notes. Retrieved from : https://ocw.mit.edu/courses/mechanical-engineering/2-12-introduction-to-robotics-fall-2005/lecture-notes/
[20] R. Venkata, Neeraj Kumar and R. Sreenivasulu. Inverse Kinematics (IK) Solution of a Robotic Manipulator Using PYTHON. Journal of Mechatronics and Robotics.