2018 4th International Conference on Computing Communication and Automation (ICCCA)

Keyboard-Based Control and Simulation of 6-DOF Robotic Arm Using ROS

Rajesh Kannan Megalingam, Nigam Katta, Raviteja Geesala, Prasant Kumar Yadav, Ruthvik Chanda Rangaiah
Dept. of Electronics and communication Engineering,
Amrita Vishwa Vidyapeetham,
Amritapuri, India
rajeshkannan@ieee.org, 123nigam.k@gmail.com, ramsrimanasi@gmail.com, yprasant0@gmail.com, ruthvikchanda1999@gmail.com

***Abstract*— In this paper, we are proposing the design and simulation of a 6 DOF robotic arm to be used for the search and rescue operations in the disaster-hit areas. An algorithm developed for the keyboard based interaction with the 6 DOF robotic arm which makes it more user-friendly is also discussed. The 6- DOF articulated robotic arm was designed and visualized in RVIZ and, Moveit is being used as a control interface using Robot Operating System (ROS). The manipulation of the arm with end effector refrain us from using the joint by the joint control mechanism. The technique presented in this research work is inexpensive and easier to control. In the designed 6 DOF robotic arm, the first three DOF is for the position of the arm and the rest three are used for the manipulation of the gripper.**

Keywords—Inverse Kinematics, URDF, Forward Kinematics, Transfer Matrix, Moveit, 6-DOF, ROS.

I. INTRODUCTION

Basically a robotic arm is a mechanical system which works similar to a human arm. It consists of an end effector usually designed to manipulate and interact with the surroundings. They are mostly used in industrial and service purposes like pick and place, exploration, manufacturing, laboratory research, space exploration etc. The 6 Degrees of freedom is to pivot in 6 different ways which resembles a human arm. The major issues concerned in an industrial robotic arm are its mechanical structure and the control mechanism. So we designed a lightweight robotic arm which makes it easy to mount and carry on any robotic system. The keyboard based control proposed in this research work which makes the designed arm easier to control compared to the other control mechanisms like the joystick control, slider based control etc. This keyboard based interface is compatible with any other robotic arm provided the dimensions and URDF of the robotic arm.

Robot Operating System (ROS) provides an integrated platform to control robotic systems. ROS is a special kind of framework initially developed with the purpose of working on robots in the research fields. In order to understand how the ROS framework works one should be clear about the concept of communication of messages through topic between nodes. Simulation is one of the ways to optimize the design and improve the control of the robotic systems.ROS provides a 3-D visualizer (RVIZ) which helps us to visualize the pose or estimates of the robot. With properly-set URDF file, one can visualize robot model in RVIZ. The Simulation software used for the control of the 6DOF robotic arm using the keyboard is RVIZ.

II. RELATED WORKS

Paper [1], discusses about simulating a robot in the Gazebo with the help of ROS_control packages and how the robot configuration is generated in Gazebo using the URDF file. The paper describes the extensive use of simulation platform for a real-life scenario. In paper [2] the design and inverse kinematics of a 3 DOF robotic arm is described. It discusses the practicality of the arm design in work and the calculations of the joint angles using kinematics while performing some tasks. Paper [3] describes how the robot's motion is controlled using inverse kinematics and the use of depth camera to capture the user's input. It also explains how to control the end effector using some speech and gesture commands. Paper [4] reports the mobility of a 5 DOF articulated arm that has been tested using direct and indirect kinematic model. The robot is simulated in MATLAB - Simulink and the trajectories and working domain is determined. Based on the known gripper position and orientation the desired rotation is calculated. Paper[5] presents the concept of an in-parallel actuated mechanism and also derives the basic kinematic equations to manipulate the 6 DOF robotic arm. They also used Jacobian matrix for simulation. Paper [6] explains about the most commonly used simulations in various robotic fields. The mathematical and analytical approach to solve the problems of forward and inverse kinematics to control the end effector is discussed in [7] and [14]. Underwater workspace and kinematic analysis of a robotic arm is studied in paper [8]. In the paper [9] the equations of forward kinematics are derived using D-H notation. The simulation is done in MATLAB by creating 3D graphics.. The simulation helped them to detect the motion of the robotic arm. Paper [10] uses a simplified simulation to control the movement of a 4 axis arm. The kinematics and inverse kinematics of the robotic arm based on the D-H parameters are calculated in paper [11]. Paper [12] and [13] gives a detailed description about multi-robot simulator and the library for grasp synthesis. Paper [15] details us on tracking and position object using vision systems associated with the robotic arm using laser range method.

III. ROBOTIC ARM KINEMATICS AND DYNAMICS

Kinematics deals with the spatial displacement of the robot as a function of time. In general, we face two main problems

978-1-5386-6947-1/18/$31.00 ©2018 IEEE

in controlling the robotic arm: first is forward/ direct kinematics and the second is inverse kinematics. We majorly control the robotic arm using inverse kinematics because it makes our job simpler and we also have IK solvers to achieve our goal. Denavit and Hartenberg (DH parameters) [1955] proposed a generalized and simpler approach of using the matrix algebra to describe the spatial geometry of the links of a robot arm with respect to a fixed reference frame. This method deals with homogeneous (4x4) matrices. In any robotic system, to understand its dynamics we must be aware of transformation matrices. Even inverse kinematics deals with transform matrices. These kinematics and inverse kinematics equations help us to understand the motion of the joints. We calculate the position and orientation of joints with respect to a reference joint. The general equations of transform matrices in the 3D-coordinate system are explained here. As shown in Fig. 1 consider two frames 'A' and 'B' and let P be a given point in frame B which is represented by the matrix □□ in equation (1). In order to understand the position and orientation of point 'P' with respect to frame 'A', we apply 3D- transformation by multiplying the matrix with the transformation matrix ${}^{A}T_{B}$ in equation (2). In the transformation matrix, we deal with both rotational and translational matrices. Rotational matrix is a 3x3 matrix and must be orthonormal whereas translational matrix is 3x1. The rotational matrices in describe the orientation of the frame.

$$P_B = \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} \tag{1}$$

$${}^{A}T_{B} = \begin{bmatrix} {}^{A}R_{B} & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \tag{2}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

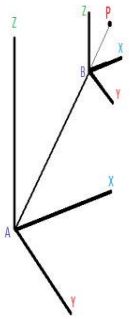

Fig. 1. Frame A and B

Steps to move the end-effector from the actual position to the new Cartesian goal position:

1. We know our actual position (x, y, z - coordinates and orientation) with forward kinematics.
2. We define our new goal position (x, y, z - coordinates and orientation).
3. Solve the inverse kinematics to get the angles for the new goal position.
4. Send angles to the motors/motor controllers to define speed and acceleration and implement a control loop for that.
5. Robot arm should move to a new position.

IV. DESIGN AND IMPLEMENTATION

A 6 DOF articulated robotic arm is designed in Solidworks with dimensions mentioned in Table 1. It is converted it into URDF using the same software with maximum reach length of 120cm. Fig. 2 represents the design of the model 6 DOF robotic arm in Solidworks. URDF (Unified Robot Description Format) is the basic/conventional format for describing robots in ROS. We have used this format to build our virtual arm. The joint positions, link lengths and global origin are defined according to our need. This is further used in RVIZ to visualize the motion of the arm accordingly. Any robot with revolute joints between all its base members falls in the category of the articulated robot. In an articulated arm numbers of joints are from two to ten. The axes of the revolute joints can be parallel or orthogonal to each other with some pairs of joints parallel and others orthogonal to each other. Arm base called as waist which is vertical to the ground and the upper body of the robot base is connected to the waist through a revolute joint which rotates along the axis of the waist and to which we have our second DOF (shoulder). Third DOF which resembles the elbow of a human arm. The fourth DOF gives yaw movement, fifth gives the pitch and the last DOF gives the roll movement. The reason behind our choice for this kind of arm is it has more work envelope. This kind of design can be used in industrial and research purpose as well. The first 3 DOF's give positional (i.e x, y, z) coordinates to the end-effector and last 3 DOF's gives the orientation of the end-effector.

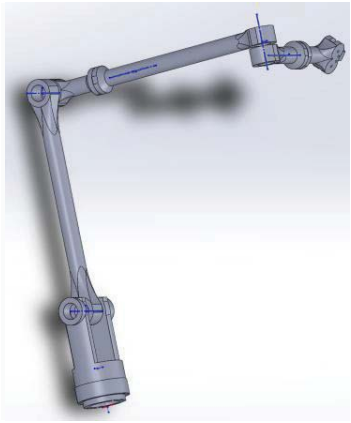

Fig. 2. Design of the 6 DOF robotic arm in Solidworks.

Using the above-designed URDF we configured MoveIt required files using MoveIt setup assistant. In this section, we first loaded the URDF file and created the collision matrix. Then the virtual joints are defined which are used to attach the arm base to the environment. Moveit divides the arm joints and manipulator into groups. This helps to have a safer control of the robotic arm by resolving errors due to singularity issues.

TABLE I. LINKS LENGTHS OF THE ARM.

S.NO	JOINTS	LENGTHS (mm)
1.	WAIST (1st DOF)	100
2.	SHOULDER (2nd DOF)	400
3.	ELBOW (3rd DOF)	100
4.	YAW ORIENTATION (4th DOF)	350
5.	PITCH ORIENTATION (5th DOF)	100
6.	ROLL ORIENTATION (6th DOF)	150

V. ARCHITECTURE

The authors described the detailed description of how to use Robotic operating system (ROS) as a platform to control and visualize a robotic arm in RVIZ. Fig. 3 represents the detailed architectural diagram of the Robotic operating system.

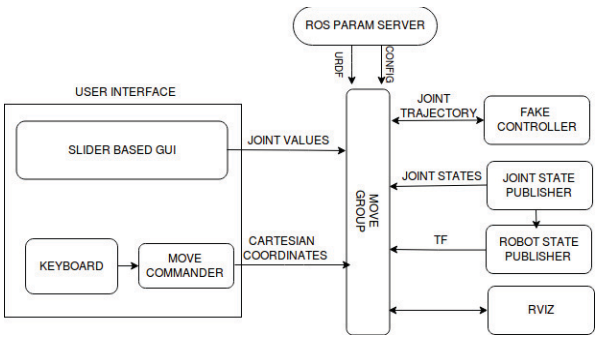

Fig. 3. Architectural diagram.

The User interface block has two different interfaces associated in controlling of 6 DOF robotic arm. Through slider based GUI user can simulate the model robotic arm using forward kinematics. Fig. 4 represents the slider based GUI. This GUI consists of 6 sliders which represents the current joint states of each DOF. Every joint can be controlled manually by changing the joint values in GUI. The inputted joint values are published to Move group. Move group monitors the current transform information and joint states to know the current pose of the model arm. With reference to the current pose, it produces the corresponding joint trajectory path to the required goal pose. Once the path is produced, RVIZ will simulate the model arm.

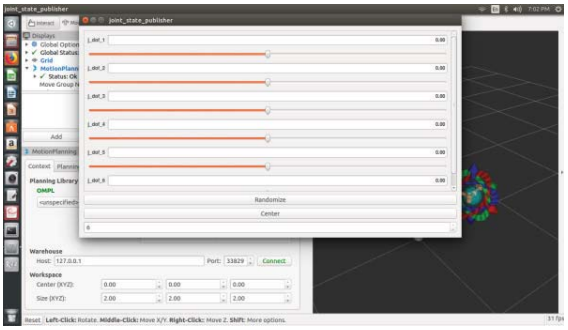

Fig. 4. Slider-Based GUI

Keyboard is another interface through which the user can simulate the model robotic arm. In this interface, the end effector of the model robotic arm is manipulated by using inverse kinematics plugin in Moveit. This plugin calculates the required joint values of the corresponding DOF when a coordinate goal position is given to the end effector. Through a custom python node, the user can input the desired x, y, z coordinates to manipulate the model arm manipulator. When the keys a, s, and d on the keyboard are pressed, accordingly the coordinate variables x, y, z increases respectively. When the keys z, x, and c are pressed the coordinate variables x, y, z decreases respectively. These x, y, z variable values are published to the move group through Move commander. Move commander acts as a bridge between keyboard and the Move group. Move group subscribes the x, y, z variable values. IK plugin in Moveit returns the corresponding joint values for the given x, y, z values. Move group gets the transforms and current joint states through the robot state publisher and joint state publisher. With reference to current pose, the Open Motion Planning library (O+MPL) plans the collision-free path for the manipulator to reach the required goal pose. The produced plan is executed by the Move group. RVIZ visualizes the entire execution of the model robotic arm.

VI. EXPERIMENT AND RESULTS

The Moveit configured files were used to launch the demo version of our URDF in RVIZ. We added Axes plugin to visualize a coordinate axis at the base of the model robotic arm. Fig. 5 represents the initial pose and a coordinate axis at the base of the model arm in RVIZ. Through Moveit commander node we built a connection between keyboard and

move group. We published different x, y, and z coordinate values to the Move group to manipulate the pose of the model arm. In figures 6, 7, 8 and 9 the transparent orange colored arm represents the previous pose of the model arm. The silver colored arm represents the goal pose of the arm for the user inputted values. The blue line indicates the z-axis, the red line indicates the x-axis and green line indicates the y-axis.

The algorithm is tested by simulating robotic arm for different user inputs. Figures 6, 7, 8 and 9 are some of the sample results of the robotic arm visualization of different coordinates given by the user.

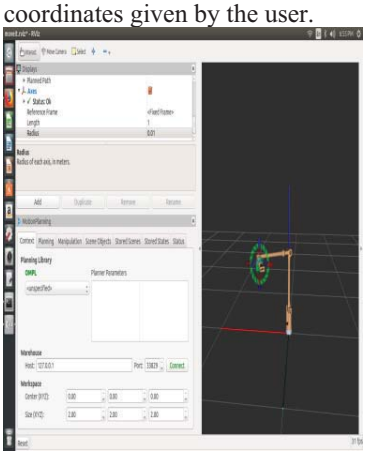

Fig. 5. Simulation of a model arm in RVIZ.

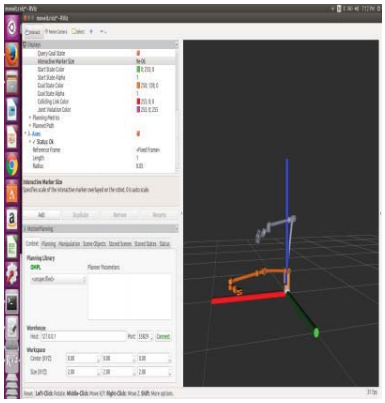

Fig. 6. Model arm at goal pose X = 0.347021457694, Y =0.201272445463, Z = 0.478438453617.

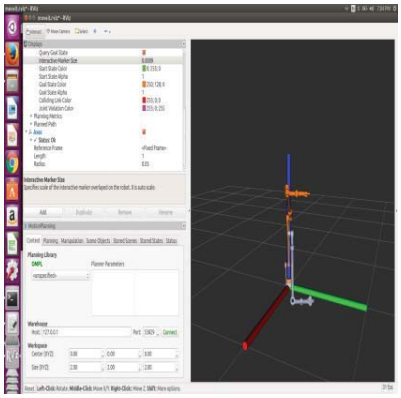

Fig. 7. Model arm at goal pose X = 0.374935093033, Y = 0.315734815186, Z = 0.0813328702545.

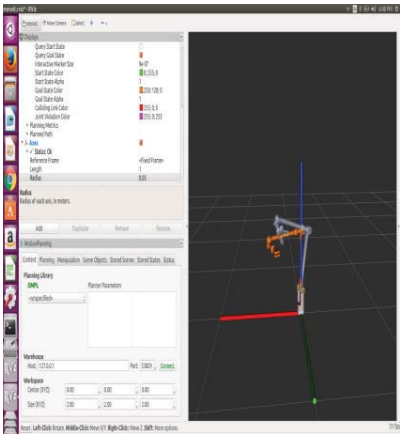

Fig. 8. Model arm at goal pose X = 0.29668090586, Y= 0.29668090586, Z = 0.451957497039.

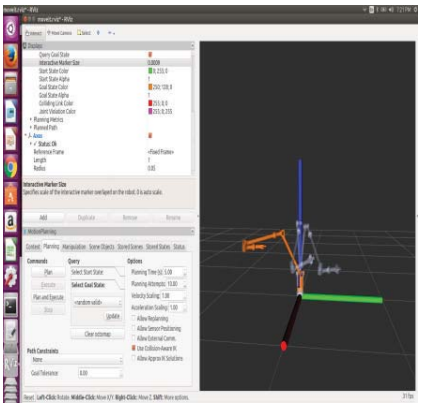

Fig. 9. Model arm at goal pose X = 0.48730810225, Y = -0.0859955474569, Z = 0.251662956214.

VII. Future Works

The 6 DOF robotic arm can be simulated in Gazebo using the ROS package controller and explore a more realistic situation to deploy the functionality of the controls. The end effector design can be improved for performing multiple divergent tasks in disaster-hit areas. The singularity issues can be reduced for better performance and enhancement of the task.

VIII. Conclusion

This paper proposed a method to control the 6 DOF robotic arm using simulation software i.e RVIZ using keyboard and sliders based GUI in ROS. The simulation helped us to know the robot parameters like the orientation and position of the robot. The main control used the concepts and algorithms of forward kinematics and inverse kinematics which helped us to control it in an easier way. The design of the 6 DOF arm is done using Solidworks CAD software and simulation is done in the RVIZ. The testing of the control system is done in RVIZ and successfully obtained the results which are required.

ACKNOWLEDGMENT

We thank Amrita University and Humanitarian Technology Lab of ECE Dept for providing us a wonderful opportunity to test and complete our project successfully.

REFERENCES

[1] Wei Qian, Zeyang Xia, Jing Xiong, Yangzhou Gan, Yangchao Guo, Shaokui Weng, Hao Deng, Ying Hu, Jianwei Zhang, "Manipulation Task Simulation using ROS and Gazebo," Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference, pp. 2594-2598, 5 December 2014.

[2] Ramish, S. B. Hussain and F. Kanwal, "Design of a 3 DoF robotic arm," 2016 Sixth International Conference on Innovative Computing Technology (INTECH), Dublin, 2016, pp. 145-149.

[3] Akhilesh Kumar Mishra, Oscar Meruvia-Pastor, " Robot arm manipulation using depth-sensing cameras and inverse kinematics," 2014 Oceans-St John's, 14-19 September, 2014 IEEE International Conference, St. John's, NL, Canada.

[4] Leba, Monica & Pop, Emil, "Articulated Robotic Arm Simulation And Control", 2018.

[5] Annamareddy Srikanth, Y. Ravitej, V.Sivaraviteja, V.Sreechand. "Kinematic Analysis and Simulation of 6 D.O.F. Of Robot for Industrial Application", International Journal Of Engineering And Science Vol.3, Issue 8 (September 2013), PP 01-04.

[6] Žlajpah Leon, "Simulation in robotics," Mathematics and Computers in Simulation, 79 (4), 15 December 2008, pp. 879-897.

[7] Alireza Khatamian, "Solving Kinematics Problems of a 6-DOF Robot Manipulator," Int'l Conf. Scientific Computing CSC'15.

[8] Asghar Khan, Wang Li Quan, "Structure design and workspace calculation of 6-DOF underwater manipulator," 2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 10-14 January 2014.

[9] Alla N Barakat, Khaled A. Gouda, Kenza.Bozed, "Kinematics analysis and simulation of a robotic arm using MATLAB," 2016 4th International Conference on Control Engineering & Information Technology (CEIT), 16-18 December 2016, Hammamet, Tunisia.

[10] Teerawat Thepmanee, Jettiya Sripituk, Prapart Ukakimapurn, "A simple technique to modeling and simulation four-axe robot-arm control," 17-20 October 2007, Coex, Seoul, Korea.

[11] Weimin Shen, Jason Gu, Yide Ma, "3D Kinematic Simulation for PA10-7C Robot Arm Based on VRML, 18-21 August 2007, Jinan, China.

[12] Stefano Carpin, Mike Lewis, Jijun Wang, "USARSim: a robot simulator for research and education" Proceedings 2007 IEEE International Conference on Robotics and Automation, 10-14 April 2007, Roma Italy.

[13] Jongwoo Park, Chan-Hun Park, Dong-li Park, "The library for grasp synthesis and robot simulation," 2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI), 28 June - 1 July 2017, Jeju South Korea.

[14] Megalingam R.K, Sivanantham V, Kumar K.S, Ghanta S, Teja P.S, Gangireddy R, Sakthiprasand K.M, Gedela V.V, "Design and development of inverse kinematic based 6 dof robotic arm using ROS," International Journal of Pure and Applied Mathematics, 118, pp. 2597-2603.

[15] Megalingam R.K, Rajesh Gangireddy, Gone Sriteja, Ashwin Kashyap, Apuroop Sai Ganesh, "Adding intelligence to the robotic coconut tree climber," 2017 International Conference on Inventive Computing and Informatics (ICICI), 23-24 November 2017, Coimbatore India.