Journal of Telecommunication, Electronic and Computer Engineering
ISSN: 2180 – 1843 e-ISSN: 2289-8131 Vol. 14 No. 2
jtec.utem.edu.my

Development of Delta Robot Arm Simulation in ROS2 Foxy Fitzroy Environment

K.M.Saipullah[1], W.H.M.Saad[1], F.N.I.Ramlee[1], M.I.Idris[1] and M.A.F.M.Din[2]
[1]*Faculty of Electronics and Computer Engineering, Universiti Teknikal Malaysia Melaka, 76100 Durian Tunggal, Melaka. Malaysia*
[2]*REKA Inisiatif Sdn Bhd, 26A, Taman City, Off Jalan Kuching, 51200 Kuala Lumpur, Malaysia.*
wira_yugi@utem.edu.my

Article Info

Article history:
Received Mar 14th, 2022
Revised Apr 13th, 2022
Accepted June 15th, 2022

Index Terms:
Delta Robot
ROS2
SDF
URDF

Abstract

This work demonstrates the methodological steps to simulate a three-degree of freedom (DoF) delta robot arm in the Foxy Fitzroy version of Robot Operating System 2 (ROS2). The mechanical design of the delta robot was represented in the form of Simulation Description Format (SDF), translated from the customised Unified Robotic Description Format (URDF) file generated using SolidWorks to URDF Exporter after the designed mechanism had been finalised. It is necessary to use the SDF instead of URDF for delta robot simulation since this type of file format supports closed-loop linkages, one of the crucial features of a delta robot arm mechanism. The simulation of the delta robot motion was conducted in the Gazebo Robot Simulator, where the positioning of the delta robot end effector was observed when the specific force was applied to each of the robot arms. The mechanism of the delta robot behaviour on forward and inverse kinematic was then simulated to observe the positioning of the end effector toward the motion of motor rotational angle and vice-versa.

I. INTRODUCTION

Robots are programmable devices that have been created to carry out many jobs as part of future progress in many fields of technology and development [1]. Robots are becoming more flexible because of constant research and development in robotic software and hardware [2]. In every element of life, robots have become prevalent and have been utilised in medical operations, human aid, factory automation, logistics and delivery, etc. Building up the physical robot to fulfil a specific task, such as a robot to solve an agriculture problem with one particular hardware setting done repeatedly, may result in saturation of the actuator or may cause a harmful condition [3]. Simulation approaches can offer an economic framework to experiment with multiple sensing and operational mechanisms to check the robot's performance in diverse circumstances and quicken the pace of development. It is reliable to overcome the gap between creative ideas and the laboratory [4], [5].

In this study, the ROS2 simulator was employed as part of the experimental platform for building the delta robot simulation. Based on its existing computer-aided design (CAD) of the delta robot model constructed using Solidwork 3D CAD/CAE design software, the URDF script of the delta robot was generated using SolidWorks to URDF Exporter tools. This format is then converted into an SDF robot description format that allows for creating joints and defining parent and child links for close loop robotic systems such as delta robots. The delta robot models can be visualised in RViz, a 3D visualisation tool for ROS applications to give models for a visual demonstration of the motion and robot sensor [4]. The model was then simulated in realistic scenarios using Gazebo Robot Simulator (GAZEBO), a robust physics engine simulator, using appropriate inertia and forces imposed on the robot link and joint.

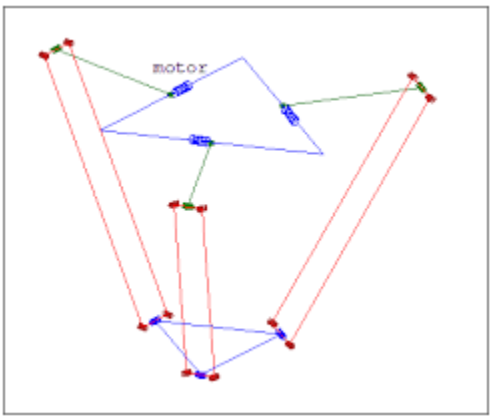

Figure 1. The Basic Skeleton Drawing of The Delta Robot Structure [6].

Delta robot is a parallel robot of over-constraint introduced in the early 90s [7]. The design basis for the Delta-Leg then became a standard delta robot design, and the characteristics of the typical delta robot are analysed in [8]. The delta robot parallel includes three parallel connections between the upper arm and strength arm connecting the engine or rotating joint with a base at the top end and ball joint connection at the bottom of the revolving plate or a two-dimensional rotor connection [9]. Figure 1 shows the basic structure of the delta robot. Due to parallelogram methods, the base and the end effector are parallel to achieve a pure translational motion. As

shown in Figure 2, the integrated central rotating unit is on the fixed base frame, and all these driving engines are mounted to provide an alternative degree of freedom (DoF) [10].

Figure 2. 4-DoF Delta Robot [11]

Clavel et al. recommended a mechanical structure where the robot actuators are mounted on a fixed frame [12]. With this, the delta robot can attain high speeds and motion acceleration. The main advantage of the delta robot is its speed. The difference between conventional and delta robots is that the standard robot arm must move the payload and all the servos in each joint, whereas the delta robot moves its frame only.

The position, velocity, acceleration, and all higher-order derivatives of position variables concerning time or some other vector or variable are studied in kinematics physics [1]. The delta robot's parallel mechanism ensures the replication of the same kinematic chains [1]. Three additional reference frames have been defined to simplify the calculations by turning the global reference frames along the z-axis for each active rotating joint on the fixed platform plane. The end-effector is always parallel to the base, and the direction around the vertical axle is always zero relative to the base. Therefore, the robot's kinematics action never varies according to the parallel nature of the joints (lower arms) [13].

It is always essential to map joint coordinates to the end-effector coordinates in robotics. This map or method used to derive the final effector coordinates from the joint coordinates is called direct kinematics or forward kinematics [14]. Forward kinematics allows the position and orientation of the final effector to be represented as a function of the joint variables of the mechanical system to the reference frame. Delta robot or parallel robot forward kinematics calculation is different from the serial kinematic calculation since three servo motors are simultaneous [15]. On the other hand, the inverse kinematics problem is complicated since the equations to be solved usually non-linear. Thus, several solutions exist, but it is not always possible to find a solution, or there are no solutions allowed given the manipulator structure of the robot.

Simulation is the method of constructing a model of a real or imaginary physical structure, running a model, and evaluating the performance of the implementation [16]. In mobile robotics, simulation has established itself as an effective tool. In a virtual environment, the ability to test and construct robotic systems is commonly used in science and education, enabling the rapid production of robotic software. [17]. The simulator helps developers efficiently program and test robots in an optimal environment where defined conditions can be easily replicated. It is also challenging to realise the simulator, regardless of its utility, because the expense of introducing the simulator can be an unforeseen cost in developing the robot, and this issue can be solved by developing an open robot simulation environment [18].

Robot Operating System (ROS) behaves as a middleware for better communication of different distributed processes frameworks. It contains programs that allow users to monitor robot operations rather than computer applications [3]. ROS allows code to be managed in a general way with more abstracted language and can therefore be easily transferred to another microprocessor that drives it. It would also be advantageous for the robot engineer to lose less time due to incompatibility problems [3]. ROS offers hardware abstraction in robotics and widely used features are available as ROS packages. The package is described as software that encapsulates essential features that are easy to reuse [19]. The latest version of ROS is called ROS2, and its file system is shown in Figure 3.

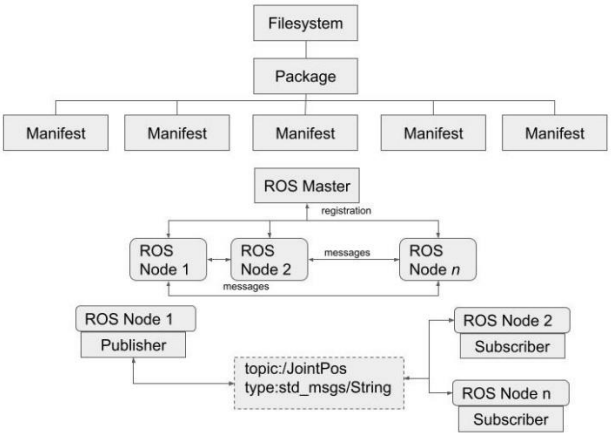

Figure 3. ROS1 and ROS2 Filesystem

Nowadays, millions of SMEs worldwide use CAD technology to design and model their robotics products [20]. Using 3D CAD kits, such as the SolidWorks API and the Autodesk Inventor, CAD-based offline robot simulation systems have been researched to simulate various industrial applications [21]. SolidWorks is a powerful program that describes geometrical structural specifics and offers a high degree of definition for functional modelling, especially in creating parameter design features. The software has a powerful assembly mechanism in that the modelled components are moulded and visually designed [22]. Using SolidWorks software to create a delta parallel robot 3D model will prevent repeating modelling work to assemble the part and make it much more straightforward than any other softwares. The sample of delta robot modelling using SolidWorks is shown in Figure 4.

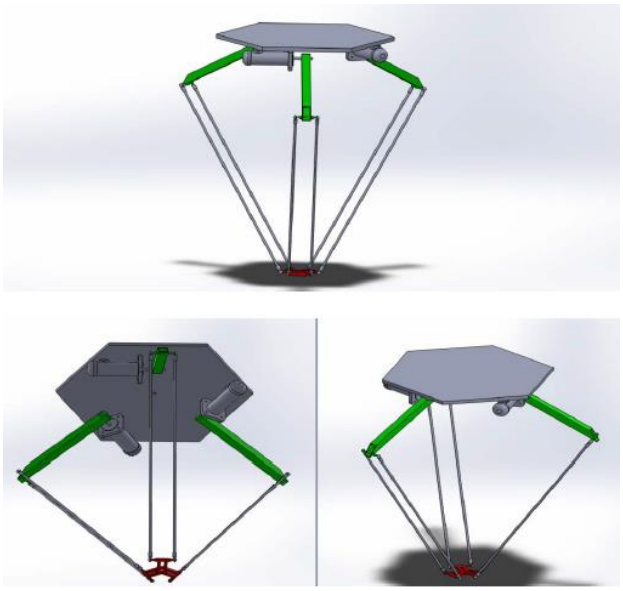

Figure 4. Delta Robot in SolidWorks

GAZEBO is a robotic simulation environment with a solid and precise physics engine, high-quality graphics and programmable user and graphical interfaces [23]. The integration between ROS and GAZEBO is assisted by a series of GAZEBO plugins supporting a wide range of existing robots and sensors [24]. A robot model can be visualised in ROS using URDF or SDF formats. By utilising this type of file, it is possible to model the different characteristics of robots, such as form, colour, joints, etc. The way to build and simulate the SDF robot model is to write and compile the SDF code script. When writing a script, it should be specified that the links and joints can be separated and that the relationship between the sections is defined as parent and child as in Table 1.

Table 1
Example Of the Robot's Description In URDF

Symbol	Quantity
Name of the robot	robot name = "robot"
Define the part	link name = "base_link"/
Define the connection node	joint name =" arm_1_joint" type = "revolute"
Define the connection relationship	parent link = "base_link"/ child_link= "arm_1"/

II. DEVELOPMENT OF DELTA ROBOT IN ROS2

A. System Environment

The summary of the environment used in this work is shown in Table 2. The primary system environment used are the Ubuntu 20.04 and the ROS2 Foxy Fitzroy.

Table 2
Summary of System Environment

Tools	Concept
VirtualBox	Oracle VirtualBox is a frameworkfor cross-platform virtualisation
Ubuntu 20.04	The key operating system (OS), with a high degree of configuration, Ubuntu is the OS of choice, facilitating the installation of all constituent components to build the simulation environment.
ROS 2 Foxy Fitzroy	The Ubuntu 20.04 (Focal) releaseis the primary target for ROS2 Foxy Fitzroy. However, other operations are controlled to varying degrees.
URDF and SDF	The URDF or SDF files are XML-based and meant to achieve a genuine robot in 3D.
GAZEBO	GAZEBO has been selected for its ease of model portability and ease of connectivity between ROS and virtualisation software modules, resources, and libraries. It can be very time-consuming and costly to test on physical hardware.

B. Delta Robot Kinematics

The top view of a delta robot arm is shown in Figure 5 where F_0: = (O_0, X_0, Y_0, Z_0), as illustrated in the figure, is defined as the robot's inertial reference frame centre at O_0. α_i are the angles related to X_0 and the following values are associated with each robot leg α_i.

$$\alpha_i = \left[\frac{\pi}{6}, \frac{5\pi}{6}, \frac{3\pi}{2}\right] \quad (1)$$

The subscript i= (1,2,3) indicates the three delta robot arm legs

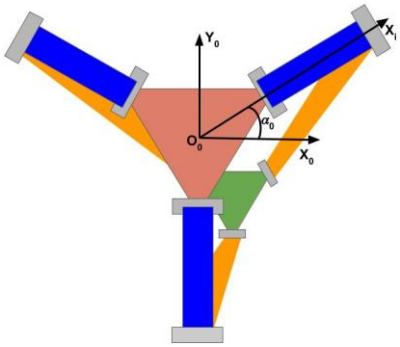

Figure 5. Top View of Delta robot Arm

A side and front view of a delta robot arm are shown in Figure 6 and Figure 7. In this diagram, R_i represents the radius of the fixed platform, or also known as a base platform or upper platform, r_i represents the radius of the mobile platform also known as end effector or lower platform, $L1_i$ represents the length of the actuated link or bicep, $L2_i$represents the length of the parallelogram or forearm, θ_{i1} represents the actuated angle (associated with the active joint A_i), and θ_{i2} and θ_{i3} represent the passive angles (associated with the passive joint B_i).

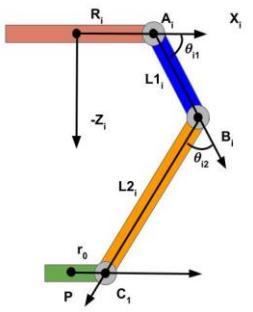

Figure 6. Side view of Delta robot arm

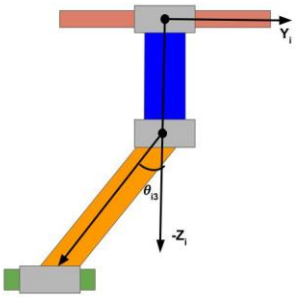

Figure 7. Front View of Delta Robot Arm

The delta robot's direct kinematics or forward kinematic, which addresses the unknown positions are $P = [X_p, Y_p, Z_p]$, for a set of angles θ_{i1} for each arm *i*=(1,2,3) of the motor, whereas for inverse kinematic, the desired P to the specific position of the end effector determines the angle of motor θ_{i1} for each arm *i*=(1,2,3) at the base platform. The conversion calculator for inverse and forward kinematic can be easily found in [25].

III. ROBOT SIMULATION AND ROS2 CONFIGURATION

A. Delta Robot CAD Design

SolidWorks is used to design the mechanical part of the robot that will then be exported to the ROS2 environment. One of the advantages of using SolidWorks is that it employs a three-dimensional design technique with a simple GUI. The 3D designs of the delta robot in SolidWorks are shown in Figure 8. After designing the component or part and verifying its motion using the URDF SolidWorks plugin, the plan is exported to URDF files.

Ideally, the URDF is used to achieve 3D models of the robots for the simulation in ROS2 and GAZEBO. The customisation form and colour features of the model are enabled by developing a 3D model using URDF. One of the drawbacks of URDF is that the closed kinematic chains cannot be simulated. The tree structure needs to be observed when building a model as the parent link will have many children, but the child link cannot have many parents. Thus, closed kinematics chains cannot be generated. The tree structure of the joints is shown in Figure 9.

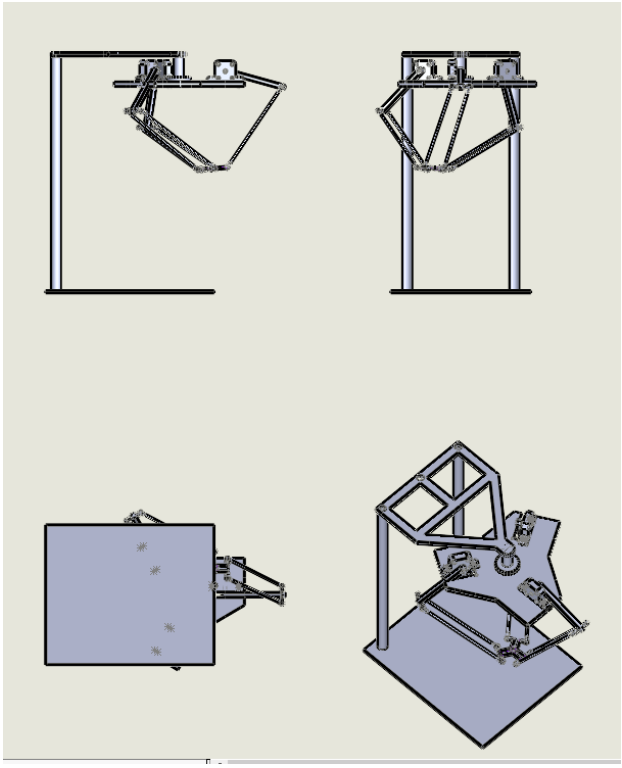

Figure 8. Sample 3D Design Delta Robot Arm in SolidWorks

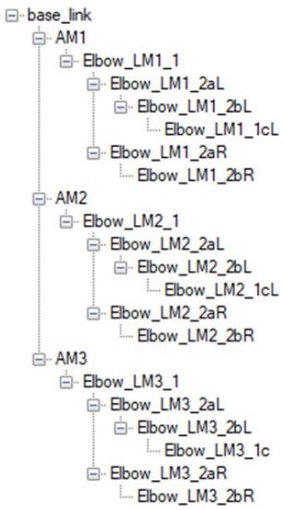

Figure 9. The Joint Tree Structures

Parallel robots like delta robots need to have a closed kinematic chain. So, a modification should be made to the URDF using a simple text editor to allow the model to be created and saved into an SDF file format since both files are in the form of XML markup language. The joint properties window in the exporter needs to be configured to adjust the coordinate reference settings for the joints and apply the joint limitations. The most crucial aspect of this is to ensure that the link has an appropriate inertia tensor in all its biceps attached to the motor. Changing the mesh from coarse to fine often results in a much more complex model with little visual difference. Coordinate systems and axes are the basic setups for the mechanical pieces of the mechanical structure of the delta robot. The origin of the base link needs to be in the correct place, and a proper coordinate system for each joint and link needs to be selected. The simple coordinate is the Cartesian with *x*-axes, *y*-axes and *z*-axes. Not having an origin point will affect the overall motion of the joints. The

visualisation of the axes used in the delta robot is shown in Figure 10. We can see that all the moving parts' axes are registered with a coordinate system.

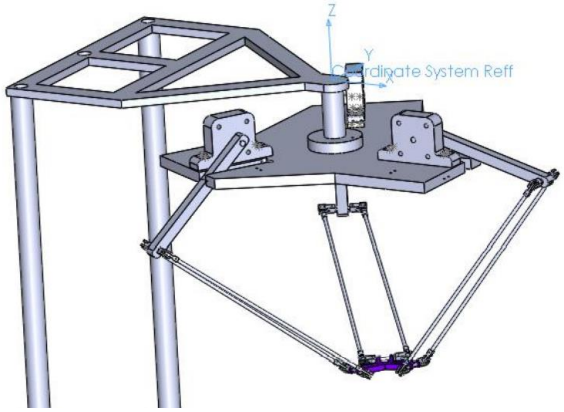

Figure 10. Axes of The Delta Robot Arm

The graph of links connected through connections between links and others can symbolise the kinematic chain or tree, as shown in Figure 11. Looking into one of the arms at i=1 where the relationship between base_link and $L1_1$, there is Elbow_LM1_1. Since the forearm of the delta robot, $L2_1$ is in a pair, from $L1_1$. It is connected to $L2_1$ left and right linking bar through Elbow_LM1_2aL and Elbow_LM1_2aR. Each link is connected to a certain number of joints that moves other links by a single connection until the last part ends movable effector platform is connected to each arm link using Elbow_LM1_1cL. Multiple joints can be connected to one connection in URDF nomenclature, although the connection can only be a child in one of its joints and must be a parent to all the other connected joints. Later, the open connection setup done in URDF is edited to make it in the close loop form inside the SDF file formatting. The Base Link is the first and the most special connection in the tree. From the Odometry Frame (/odom) and Map, the origin of the axis system for World x, y, and z codes is established. It should be mentioned that the URDF_to_graphiz command in the terminal can generate a graph of the kinematic tree or chain.

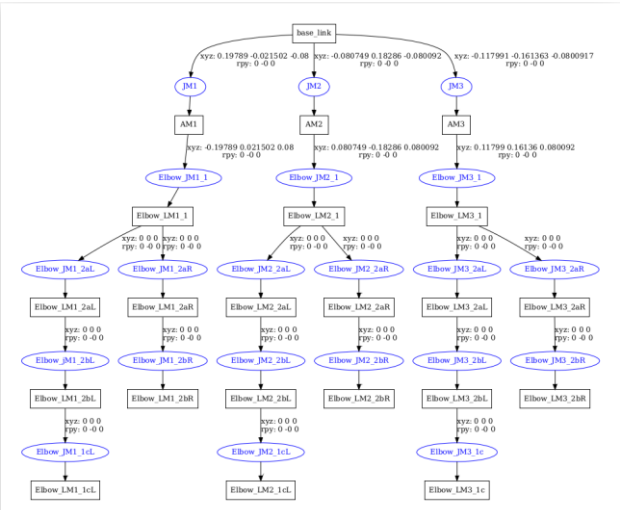

Figure 11. Graph of Links of The Delta Robot Arm Joints in URDF

The ROS2 control package provides excellent capabilities for controlling robots. Similarly, the joint state publisher and robot state publisher packages are essential for controlling the robots. The joint displacement is confined to one axis due to the physical properties of revolute joints. As a result, the motion plan in this section is restricted to moving the robot model with the motor along the YZ-axis. The ROS node is responsible for calculating the desired angles and sending them to the joints in the 3D model to complete the motion operation. While constructing the ROS2 package using the colcon build instruction, the SDF file is included along with other files and folders such as package.xml, launch file and folder, and setup.cfg. The 3D model robot in RViz with the joint_state_publisher to control the movement can be viewed in Figure 12.

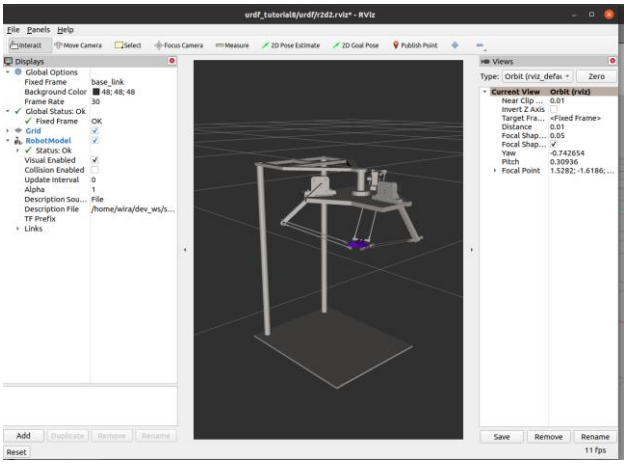

Figure 12. The 3D Views of the Delta Robot in RViz

B. GAZEBO Model

Parallel robots have a closed kinematic chain; hence a change should be made to allow model generation. As a result, the robot was designed using a URDF file, which had been converted to an SDF file (simulation description format) with the command used

```
gz sdf -p ./my_urdf.urdf > ./my_sdf.sdf
```

First, to build a delta robot arm, it is necessary to create a folder for the model, open a terminal and use this command:

```
mkdir -p ~/.gazebo/models/delta_robot_arm.
```

Next, a configuration model file that contains the model description is generated. This file provides the robot's name, version, creator, email address and robot description. This command is used to modify this model.config file:

```
gedit ~/.gazebo/models/ delta_robot_arm
    /model.config.
```

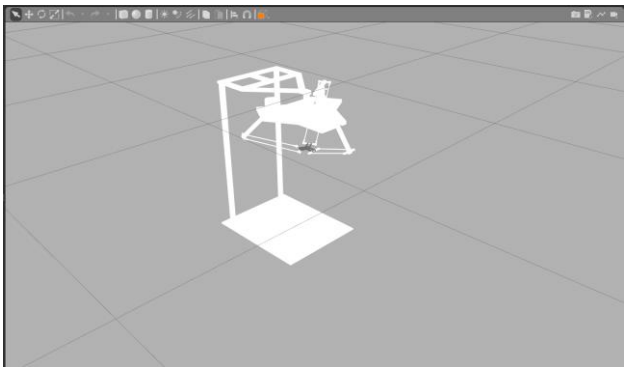

Figure 13. The Outcome of the Delta Robot in GAZEBO

After that, from URDF exporter, a file containing meshes it to provide us with .STL file that needs to be converted to .DAE. by using a mesh converter. Then, the .dae file is moved into a folder for the model, which is delta_robot_arm. Thus, creating an SDF file (simulation format description), which has been converted from the commands previously. Before running the robot model into GAZEBO, all the necessary files, model.config, meshes in .dae and model.sdf. need to be checked Then, GAZEBO is run to view a robot model. The robot model can be viewed in Figure 13. As a result, users can place it wherever they choose by clicking inside the environment.

IV. CONCLUSION AND FUTURE WORKS

In conclusion, the capacity to develop a delta robot arm simulation for the ROS2 Environment has been proved in this project. ROS2 was chosen because it is the best middleware for its current requirements. It is vital to note that the delta robot arm simulation is still in the stage of work in progress. This progress indicates that using the library in a real-world operation is not advisable. Despite this flaw, ROS's power lies in the development community. Once the ROS library has reached the practical level, it may be implemented in the actual process. Another option is to create this library and submit it to the ROS Industrial repository branch, where it will be reviewed and implemented in the repository. The type of joints provided in the URDF packages was the critical constraint during the project's development. A few alternatives are given to tackle this problem. The first is to convert the URDF data into SDF files and the meshes STL files to .DAE files, as it is possible to contribute to GAZEBO modelling. The second option is to operate the robot in RViz using a joint state publisher.

ACKNOWLEDGEMENT

Authors would like to thank the Machine Learning & Signal Processing (MLSP) research group under Fakulti Kejuruteraan Elektronik dan Kejuruteraan Komputer (FKEKK), of Universiti Teknikal Malaysia Melaka (UTeM) for sponsoring this work under project PJP/2020/FKEKK/PP/S01788 and providing the use of the existing facilities to complete this project.

REFERENCES

[1] R. T. Arrazate, "Development of a URDF file for simulation and programming of a delta robot using ROS," no. February, 2017.

[2] A. M. Romanov, "A review on control systems hardware and software for robots of various scale and purpose. part 1. industrial robotics," *Russian Technological Journal*, vol. 7, no. 5, pp. 30–46, 2019.

[3] J. Kerr and K. Nickels, "Robot operating systems: Bridging the gap between human and robot," *Proceedings of the Annual Southeastern Symposium on System Theory*, pp. 99–104, 2012, doi: 10.1109/SSST.2012.6195127.

[4] A. Domel, S. Kriegel, M. Brucker, and M. Suppa, "Autonomous pick and place operations in industrial production," vol. 39, no. 2, pp. 356–356, 2015, doi: 10.1109/urai.2015.7358978.

[5] S. Gunasagaran, K. Kamarudin, A. S. A. Yeon, R. Visvanathan, S. M. Mamduh, and A. Zakaria, "Implementation of Mobile Robot Localisation and Path Planning for Navigation in Known Map," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 10, no. 1–15, Art. no. 1–15, May 2018, Accessed: Feb. 07, 2022. [Online]. Available: https://jtec.utem.edu.my/jtec/article/view/4049

[6] "Delta robot," *Wikipedia*. Jan. 26, 2022. Accessed: Feb. 07, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Delta_robot&oldid=1067967400

[7] F. Azad, M. R. H. Yazdi, and M. T. Masouleh, "Kinematic and Dynamic Analysis of 3-DOF Delta Parallel Robot Based on the Screw Theory and Principle of Virtual Work," *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*, 2019, doi: 10.1109/KBEI.2019.8734994.

[8] S. Giewont and F. Sahin, "Delta-Quad: An omnidirectional quadruped implementation using parallel jointed leg architecture," in *2017 12th System of Systems Engineering Conference (SoSE)*, 2017, pp. 1–6. doi: 10.1109/SYSOSE.2017.7994964.

[9] S. Staporchaisit, C. Mitsantisuk, N. Chayopitak, and Y. Koike, "Bilateral control in delta robot by using Jacobian matrix," *2015 6th International Conference on Information and Communication Technology for Embedded Systems, IC-ICTES 2015*, 2015, doi: 10.1109/ICTEmSys.2015.7110816.

[10] H. Cheng, Z. Zhang, and W. Li, "Dynamic error modeling and compensation in high speed delta robot pick-and-place process," *2015 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2015*, pp. 36–41, 2015, doi: 10.1109/CYBER.2015.7287906.

[11] Humanrobo, *English: TOSY Industrial Robot: Parallel robot*. 2009. Accessed: Feb. 07, 2022. [Online]. Available: https://commons.wikimedia.org/wiki/File:TI_P408-01.jpg

[12] L. Angel and J. Viola, "Parametric identification of a delta type parallel robot," *2016 IEEE Colombian Conference on Robotics and Automation, CCRA 2016 - Conference Proceedings*, 2017, doi: 10.1109/CCRA.2016.7811415.

[13] S. Ahangar, M. V. Mehrabani, A. P. Shorijeh, and M. T. Masouleh, "Design a 3-DOF Delta Parallel Robot by One Degree Redundancy along the Conveyor Axis, A Novel Automation Approach," *2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI)*, pp. 413–418, 2019.

[14] "Why ROS 2?" https://design.ros2.org/articles/why_ros2.html (accessed Jan. 25, 2022).

[15] M. Mustafa, R. Misuari, and H. Daniyal, "Forward Kinematics of 3 Degree of Freedom Delta Robot," *2007 5th Student Conference on Research and Development*, pp. 1–4, 2007.

[16] L. lajpah, "Simulation in Robotics," *Math. Comput. Simul.*, vol. 79, no. 4, pp. 879–897, Dec. 2008, doi: 10.1016/j.matcom.2008.02.017.

[17] P. G. Costa, J. Gonçalves, J. Lima, and P. Malheiros, "SimTwo Realistic Simulator: A Tool for the Development and Validation of Robot Software," *Theory and Applications of Mathematics & Computer Science*, vol. 1, pp. 17–33, 2011.

[18] T. Ishimura, T. Kato, K. Oda, and T. Ohashi, "An Open Robot Simulator Environment."

[19] P. Estefo, J. Simmonds, R. Robbes, and J. Fabry, "The Robot Operating System: Package reuse and community dynamics," *J. Syst. Softw.*, vol. 151, pp. 226–242, 2019.

[20] P. Neto and N. Mendes, "Direct off-line robot programming via a common CAD package," *ArXiv*, vol. abs/1309.2078, 2013.

[21] A. K. Bedaka and C.-Y. Lin, "CAD-based robot path planning and simulation using OPEN CASCADE," *Procedia Computer Science*, vol. 133, pp. 779–785, 2018.

[22] Z. Chang, R. A. Ali, P. Ren, G. Zhang, and P. Wu, "Dynamics and Vibration Analysis of Delta Robot," 2015.

[23] A. Dömel, S. Kriegel, M. Brucker, and M. Suppa, "Autonomous pick and place operations in industrial production," in *2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2015, pp. 356–356. doi: 10.1109/URAI.2015.7358978.

[24] S. Raje, *Evaluation of ROS and Gazebo Simulation Environment using TurtleBot3 robot*. 2020.

[25] "Delta Robot Forward/Inverse Kinematics Calculations." https://www.marginallyclever.com/other/samples/fk-ik-test.html (accessed Feb. 06, 2022).