ROS 2 Configuration for Delta Robot Arm Kinematic Motion and Stereo Camera Visualization

Khairul Muzzammil Saipullah [1], Wira Hidayat Mohd Saad [2]*, Sook Hui Chong [3], Muhammad Idzdihar Idris [4], Syafeeza Ahmad Radzi [5]
[1,2,3,4,5] Faculty of Electronic and Computer Engineering, Universiti Teknikal Malaysia Melaka, 76100 Durian Tunggal, Melaka, Malaysia
[1]Autonomous Robotics Division, Strukture Robotics Sdn. Bhd., 63000 Cyberjaya, Selangor, Malaysia
Email: [2] wira_yugi@utem.edu.my
* Corresponding Author

***Abstract*—The Delta robot is one of the robot types that is used in agriculture and industrial application. However, before the complex physical development of the robot, a simulation needs to be developed to ensure the perfect functionality of the design. Therefore, this paper presented a development of simulation for a parallel delta robot using a Robot Operating System 2 (ROS 2) environment and stereo camera visualization. The contribution of this research is to present the development details and the proposed solution to solve issues encountered during the development. The development of script in the format of eXtensible Markup Language (XML), Unified Robot Description Format (URDF), and Simulation Description Format (SDF) are presented for describing a robot's physical structure, allowing a robotic system to be depicted in a tree structure, and defining the delta robot arm, which is made up of closed-loop kinematic chain linkage that will be simulated in Gazebo. For the results, several Gazebo plugin libraries are compared and tested for the wheels motion control, stereo camera visualization, and delta robot arm kinematic motion. From the experiment, the best method is inverse kinematic motion the method is selected and used in the simulation. The selected method resulted in an average percentage error of 3.92%, 3.72%, and 2.92%, respectively for each joint.**

Keywords—ROS2; Delta Robot; Stereo Camera; SDF; Gazebo; Rviz2.

I. INTRODUCTION

The term "robot" refers to an electromechanical device with numerous degrees of freedom (DOF) that humans can program to perform various tasks. [1]. In the year of 1979, the Robot Institute of America has defined the word 'robot' as a "re-programmable, multifunctional manipulator that is designed to move material, parts, tools, or specialized devices through variable programmed motions for the performance of a variety of tasks" [2]. A robot arm in a robotic system represents the end-effector that simulates the human arm to carry out tasks such as picking and placing. For years, robotics has been used in the industry. A robotic system can also fulfill its duty in agriculture to complete some acceptable jobs by researching the parallels between agriculture and industries [3][39][40]. The study of robotics is to combine and simulate certain aspects of human body function by implementing mechanisms, sensors, actuators, and computers [4].

Based on research done in [5], agriculture is one of the crucial sectors contributing to the world's economic growth. For instance, the agriculture sector in Malaysia had contributed 7.1 percent to the Gross Domestic Product (GDP) in 2019 [6]. The rise in the number of people working in agriculture has prompted the development of autonomous robotics to boost production and efficiency [4][7]. Throughout the revolution in agriculture, different types of robotics development have been implemented. The types of robotic technology included fruit pickers, weed pullers, drones, and Light Detection and Ranging (LIDAR) [8].

The delta robot is one of the sorts of robots that can be used in agriculture. Due to its speed and product assembly benefits, the delta robot, also known as a parallel robot, is frequently employed in industrial applications [9]. The Delta robot is a parallel robot as the top fixed base is always parallel to the bottom moving end effector in their reference axis. Delta robot can perform the movement in X, Y, and Z directions with a fixed base on top that connects three arms, and all the arms are connected as a triangular form to the interconnected end effector [10][35]. A camera can be added to the end-effector to allow image visualization of the thing to be picked up or followed [36]. In this project, a stereo camera can be mounted to the delta robot's end effector to offer a stereo viewing capability. The stereo camera comprises two or more image sensors that allow it to replicate human binocular vision by adjusting the distance between two lenses, allowing depth information to be reorganized [11] [27].

In this paper, a simulation of a parallel delta robot based on ROS 2 is designed and developed. The main contribution of this research is the study of the development of essential tools for delta robot simulation in the ROS 2 environment such as the XML, URDF, and SDF files. The second contribution is the discussion of the simulation development analysis that includes the proposed solutions taken to solve the issues encountered during the development of the delta robot simulation. This paper is arranged as follows. Section II will be discussing the background of tools and software used in the development. Section III will discuss the detailed development and analysis of the development and the last Section IV will present the conclusion of the research.

II. Method

A. Robot Operating System (ROS)

Robot Operating System (ROS) is a middleware framework that offers an abstraction layer between the hardware and the application layers. ROS allows the developers to perform manipulation on the robot hardware [37][38]. ROS is an open-source platform that depends on the contribution of the developers and the ROS user community in the introduction of new packages or libraries. It is equipped with a system used to manage the packages, which allows users to reuse and reimplement the code easily and hence allows developers to contribute their design and application back as a package to ROS [12]. ROS goes on top of Linux Ubuntu in the service robot sector and is becoming the standard in robotic programming [13].

ROS2 is a complete re-design of the framework of ROS to tackle the shortcomings of ROS, which allow ROS2 to satisfy the industry needs and standards effectively [14]. The most significant difference between ROS and ROS2 is that ROS2 selects the Data Distribution Service (DDS) middleware for the communication layer, whereas ROS is not [15]. DDS middleware is known as a software layer. It allows the abstraction of the application from the information of the operating system, low-level data formats, and network transport. The DDS provides the Dynamic Discovery of publishers and subscribers in ROS2. With Dynamic Discovery, the endpoints for the communications do not require any configuration as they will be automatically discovered by DDS [16]. A topic is essential in acting as a bus for the messages exchanged between nodes. A node may simultaneously publish and subscribe data to several topics. When a Publisher node wants to send a message to a Subscriber node, the Publisher node will need to publish the message to the topic so that the Subscriber node will subscribe to that message via the topic. The distro of ROS2 selected for this project is ROS2, Foxy Fitzroy, with the codename of 'foxy'. Foxy Fitzroy is the sixth release of ROS 2 on June 5th, 2020, that will meet its End of Life (EOL) date on May of 2023 [17].

Rviz stands for ROS Visualization, is a ROS 3D visualization tool that allows users to visualize the robot model through the transform (tf) published by a ROS package such as *robot_state_publisher*. Besides that, Rviz can display the data captured by a camera or a laser scan in terms of images or point clouds. A Gazebo can provide a close substitute for how a robot would behave in a real-world physical environment. It may replicate forces such as gravity and torque in the simulation environment. The main difference between Rviz and Gazebo can be summed up in the excerpt from Morgan Quiley, who is one of the original developers of ROS, in his book [18] "Rviz visualize what the robot think is happening while Gazebo shows what is happening in a real-world environment.".

Simulation Description Format, abbreviated as SDFormat or SDF, is another XML format file that describes the robot physical's structural description and the environments for the robot simulators, visualization, and control. Other than kinematic and dynamic attributes, SDF also allows the sensor, surface properties, texture, and joint friction of a robot to be defined in detail [19]. SDF is the standard format of the Gazebo simulation modeling, where the sensors and actuators can be implemented in Gazebo by using a plugin library [20].

B. Delta Robot

Delta Robot, also known as a parallel robot, is widely used in industry due to its high-speed performance as it has a lightweight structure. It is a successful invention from Reymon Clavel in 1985. The Delta robot is often used in industrial picking and packaging processes [21]. The Delta robot is composed of several kinematic chains that connect between the end-effector and the top base to perform movement through the rotation of the servo motor and the joints [22].

Delta robot is available in different Degree-of-freedom (DOF) such as 3-DOF or 4-DOF. It indicates the number of movable joints available in the Delta Robot. There are 3-DOF for X-Y-Z translation for a 3-DOF Delta Robot, while 4-DOF Delta Robot has an extra fourth inner leg which controls the rotation freedom of the platform of the end effector [23]. By analyzing the kinematic links of a Delta Robot, the total number of degrees of freedom can be computed through Equation (1) [24].

$$F = b(n - g - 1) + \sum_{i=1}^{g} fi - fid + s \tag{1}$$

Where b is number of DOF in space, n is number of elements, g is number of joints, fi is DOF of i-th element, is fid = number of identical DOF and s is number of passive joints

Kinematics refers to the motion performed by the body of a robotic mechanism without consideration of the forces and torques that cause the motion. The connection between several rigid bodies forms a robotic mechanism through the joint where the orientation and position of the rigid body are known as "pose". Hence, the kinematics of a robotic mechanism describes the velocity, acceleration, pose, and all higher-order derivatives of the pose of the body that comprises a mechanism [25].

There are two types of kinematic motion: forward kinematic and inverse kinematic. The forward kinematic motion of a delta robot indicates the condition where the joint variable, such as the servo motor's angular rotation angle, is known and is used to compute the position and orientation of the end-effector of the delta robot arm. Inverse Kinematic motion is performed when the position and orientation of the end-effector are known. The angle of angular rotation of the servo motor and the joints will be calculated and identified based on the orientation of the end-effector [26].

C. Delta Robot Simulation

Some related projects have been referred to obtain the idea for this project. In the paper "Delta robot controlled by robotic operating system" [28], the implementation of wireless control system on a parallel robot based on the RS-232 interface is carried out. The simulation environment chosen in this project is ROS for implementing the controlling of the robot joint actuators. Program scripts are built to apply the inverse kinematics of the delta robot.

Besides, Python-based algorithms supported by OpenCV libraries are also developed for vectorizing and rasterizing images. The developed robot undergoes an experiment in drawing applications. The inverse kinematics principle is used to determine the angle of each actuator by knowing the end-effector's position. This project carried out the robot movement simulation process in Rviz, and Markers were used to represent the simple shapes of the delta robot.

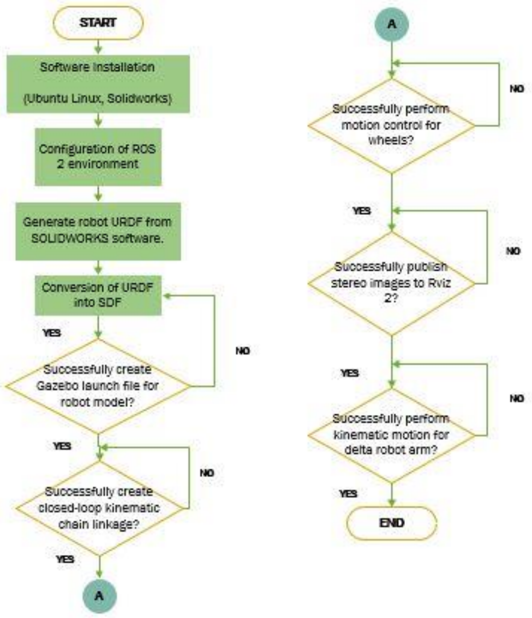


Fig. 1. The flow chart of the simulation development

In work done by R. Barth et al. in [29], it reported the experience of using ROS middleware for the development of agricultural robots. In that project, ROS is implemented to construct the software for several subsystems. The subsystems developed included perception, sensing, manipulator control, system framework, and mission control. The perception framework of the project includes physical sensors, virtual sensors, and sensor fusion. The dull implementation of the modular framework was conducted in ROS and MATLAB. The development and evaluation of algorithms are implemented in MATLAB and transferred over to ROS once they are finalized.

The work done by F. Okoli et al. in [30] focussed on the cable-driven parallel robot by developing a simulator using Gazebo and ROS. The properties such as mass, inertia, pose, link visual, and collision of the robot are described in an SDF file. A dynamic controller is developed to illustrate the proposed simulator by detailing the tension distribution, and various trajectories are performed. A trajectory controller is developed to compute the desired wrench to be applied to the platform, and a Tension Distribution Algorithm is used to map the desired wrench.

D. *Proposed Development*

The overall development flow is summarized and shown in Fig. 1. The flow involves the following tasks:

a. Installation and setup of the necessary software.
b. 3D design and export of the delta robot into ROS environments.
c. Development of essential codes and files for the simulation
d. Troubleshooting for issues encountered during the development

In this project, Gazebo will be used as the 3D simulator to simulate the operation of the robotic system, and Rviz 2 will be used to visualize the stereo images captured by the virtual stereo camera attached to the delta robot arm. The research gap between this study and the work in [28] is that this project aims to simulate the kinematic motion of a delta robot with 3-DOF in Gazebo simulator by describing the robot's physical structure in an SDF file. SDF can describe the closed-loop kinematic chain linkage of the delta robot arm to be simulated in Gazebo. Besides, the research gap between this work and the study proposed in [29] is that the simulation of this project will be carried out without the implementation of algorithm evaluation in MATLAB, where the currently available plugin library resources in ROS2 will be utilized without involving the development of a customize algorithm or controller.

III. RESULT AND ANALYSIS

A. *Spawning Delta Robot*

ROS Visualization (RVIZ) is a visualization platform that allows users to view the robot state and the sensor output, such as image or point cloud. The *robot_state_publisher* node needs to be run to enable Rviz 2 to parse the robot's structure for viewing the robot state in Rviz 2 application. *robot_state_publisher* uses the Unified Robot Description Format (URDF) file specified by the *robot_description* parameter. The URDF file of the root is generated from SOLIDWORKS software. The *robot_state_publisher* and *joint_state_publisher_gui* are declared in a launch file to allow the launching of two nodes simultaneously. *joint_state_publisher* is a ROS package that reads the *robot_description* parameter to find all the non-fixed joints and publishes a JointState message. The *joint_state_publisher*_node will then be run to show the Graphical User Interface (GUI) of Joint State Publisher, as shown in Fig. 2, that can be used to adjust the angular rotational degree of the non-fixed joints.

However, as URDF only supports the robot kinematic chain structure as a tree structure, the closed-loop kinematic chain linkage of a delta robot cannot be achieved by using URDF. When the sliders in *joint_state_publisher*_gui are varied to change the angular rotational degree of the joints, the arm's connection can be seen as broken, as shown in Fig. 3.

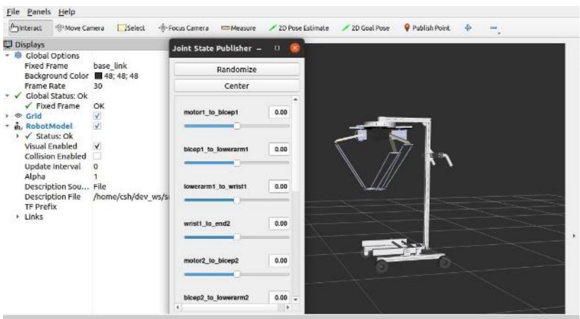


Fig. 2. Joint State Publisher GUI.

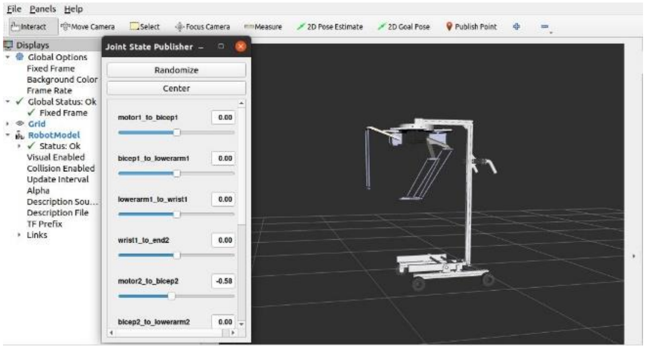


Fig. 3. Broken connection of delta robot arm.

From the result obtained by using URDF to describe the delta robot and visualize it in Rviz 2, it is proved that URDF is not suitable for this project to define the robot structure of the delta robot. Hence, the Simulation Description Format (SDF) file will describe the physical robot structure of the delta robot in this project. The connection for the delta robot arm has been modified to form a closed-loop kinematic chain linkage, and a launch file is written by referring to the tutorial published by the work in [31] to launch the delta robot in Gazebo. The result of spawning the robot in Gazebo is shown in Fig. 4.

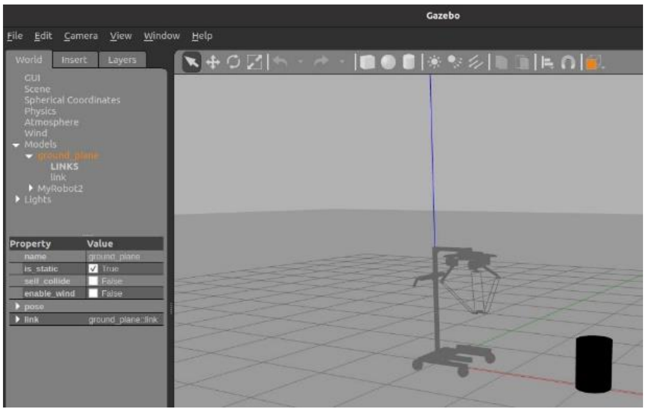


Fig. 4. Robot spawned in Gazebo.

B. *Wheel Motion Control*

The wheel motion control of the delta robot is performed by implementing the Gazebo plugin in the SDF file of the delta robot. The gazebo plugin used for the wheel motion control in this project is the *skid_steer_drive* plugin, which can control four wheels. In the SDF file, the plugin is added with the tag <plugin>, and the .so file, which is the plugin library's object file, is specified to allow the system to run the function in the library code. The *skid_steer_drive* plugin will publish a topic named */demo/cmd_demo*. The /demo is indicated in the <namespace> tag, and the *cmd_demo* is the remapping of the *cmd_vel*, a topic that will be used to control the velocity of a joint. Several approaches can be implemented to send the velocity command to the topic published by the plugin. For example, the velocity command can be sent through the geometry_msgs/Twist message by running the command line shown in Table 1.

TABLE I. COMMAND FOR VELOCITY

ros2 topic pub /demo/cmd_demo geometry_msgs/Twist '{linear: {x: 1.0}}' -1

This command line will allow the robot to move towards the x-axis with a velocity of 1 meter per second. Besides, the wheel motion of the robot can be controlled through the robot steering function equipped in the rqt. The Graphical User Interface (GUI) for the robot steering, as shown in Fig. 5, can be found in rqt>Plugins>Robot Tools>Robot Steer. By moving the slider, the velocity of the robot can be varied. This approach also sends the geometry_msgs/Twist message to the topic published by the plugin.

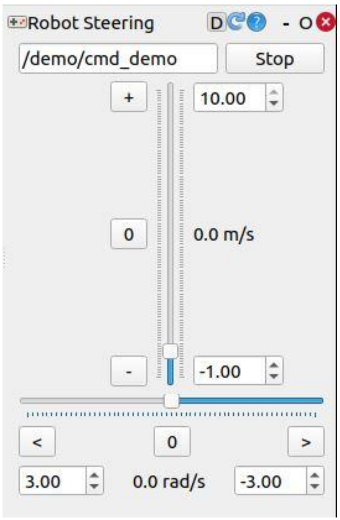


Fig. 5. Robot steering GUI.

In this project, the approach chosen to control the wheel motion is through the *teleop_twist_keyboard* package. The node of *teleop_twist_keyboard* can be run by running the command line shown in Table 2.

TABLE II. COMMAND FOR TELEOP NODE

ros2 run teleop_twist_keyoboard teleop_twist_keyboard - - ros-args --remap cmd_vel:=/demo/cmd_demo

Using the Gazebo Plotting Utility feature, the Velocity vs Sim Time graph is plotted in Fig. 6. Some overshoots can be observed at the starting time of the graph as the robot starts to change its state from a static state to a moving state. However, when the velocity is varied after the starting time, the overshoot is reduced for every time of variation.

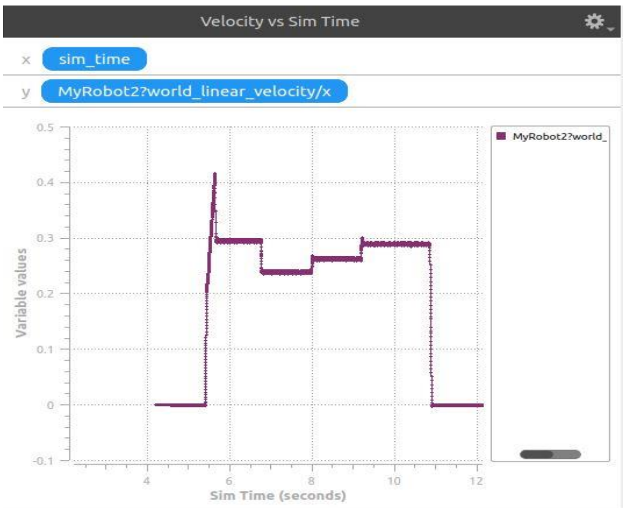


Fig. 6. Velocity vs Sim Time graph.

C. Stereo Camera Visualization

The stereo visualization of the delta robot is performed by implementing the stereo camera sensor and stereo camera plugin in the SDF file. In the plugin code, the baseline is set to 0.07, while the distortion is set to 0 to generate a pair of clear images without distortion. The baseline of the stereo camera represents the distance between the centre of the left and right lens. By referring to the Intel Realsense D400 Series Product Family Datasheet [32], the baseline value for the stereo depth module is suggested to be 20 mm to 70 mm. Hence, the baseline for the stereo camera in this project is set to 70 mm, which is 0.07 m.

The stereo camera visualization of the robot is successfully generated in Rviz 2, where the output of the left and right images captured by the stereo camera attached to the delta robot arm is shown in Fig. 7. The top image represents the left view of the camera, and the bottom image represents the right view.

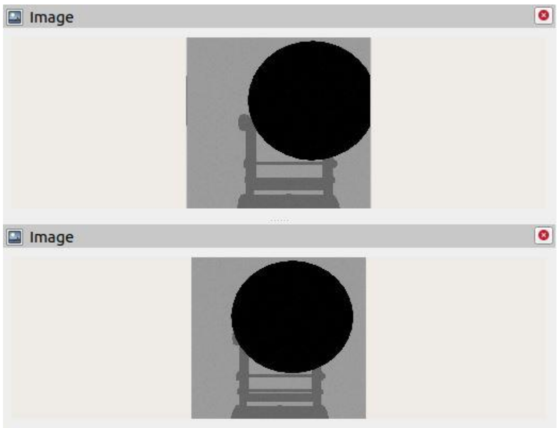


Fig. 7. Left and right images generated in Rviz 2.

D. Delta Robot Arm Kinematic Motion

In this project, the kinematic motion performed by the delta robot arm is implemented through the approach of implementing a Gazebo plugin named *tricycle_drive* that will control the rotational velocity of three motors.

The *teleop_twist_keyboard* will be used to send the velocity command to the motors connecting to three of the kinematic chains. The topic published by the *tricycle_drive* plugin is */demo/cmd_arm,* which is the remapping of *cmd_vel*. The command line to run the *teleop_twist_keyboard* to control the motors is by specifying the topic published by the plugin shown in Table 3.

TABLE III. COMMAND FOR TELEOP FOR SPECIFIC MOTOR

ros2 run teleop_twist_keyoboard teleop_twist_keyboard - - ros-args - -remap cmd_vel:=/demo/cmd_arm

The concept of inverse kinematic is implemented where a cylindrical object is placed at the coordinate of (2 0 0) in the Gazebo world as the target object that the end-effector will reach. After the delta robot is driven to the location where the end-effector is located right above the object, the kinematic motion of the delta robot arm is performed. By varying the height of the object to indicate the end-effector's position along the Z-axis, the positions of the joints of three motors in terms of angular rotational degree (radian) are recorded.

Ideally, the angular rotational degree of three motors will be 0 degrees when the biceps are parallel to the floor. From Fig. 8, three biceps are not parallel to the floor. As the SDF file is converted from the URDF file exported from SOLIDWORKS software, the robot state in the software will become the initial position of the delta robot, where the current position of the biceps with offset will be indicated as 0 degrees during the simulation. The offset angle of each biceps is identified by using the measuring tool in SOLIDWORKS software. The offset angle is computed through the trigonometric formed at the motors, as shown in Fig. 9.

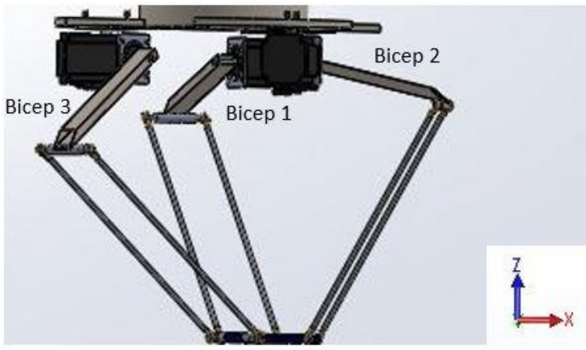


Fig. 8. Labelled bicep of a delta robot arm.

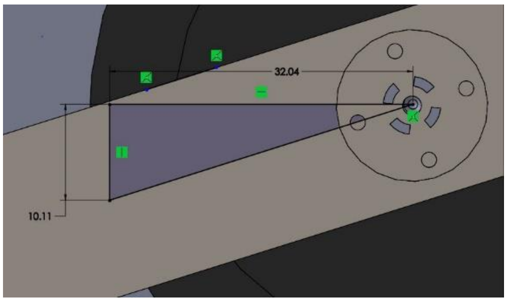


Fig. 9. A right-angled triangle formed at Bicep 1.

By applying the trigonometric theorem, the offset angles of the three biceps are computed and recorded in Table 4.

TABLE IV. OFFSET ANGLE OF BICEPS.

Bicep	Offset angle (degree)	Offset angle (radian)
Bicep 1	17.513	0.3057
Bicep 2	14.448	0.2522
Bicep 3	19.238	0.3358

The height of the cylindrical object is varied for five different values to analyze the angular rotational degree of the motor when the end-effector is located at a different coordinate. A gazebo plugin named "*gazebo_ros_ joint_state_publisher*" is added to the SDF file of the delta robot to read the joint state of the joints specified in the code.

The information such as the joint name, position, and velocity published by *the gazebo_ros_ joint_state_publisher* plugin can be viewed by echoing the topic published by the plugin as shown in Fig. 10. The angular rotational degrees (radian) of each joint are recorded, and the theoretical value of the angular rotational degrees is determined by using the online calculator in [33] by referring to the paper done by Y. Kadam et al. [34]. The results are tabulated in Table 5, where the error percentages of each result are also identified.

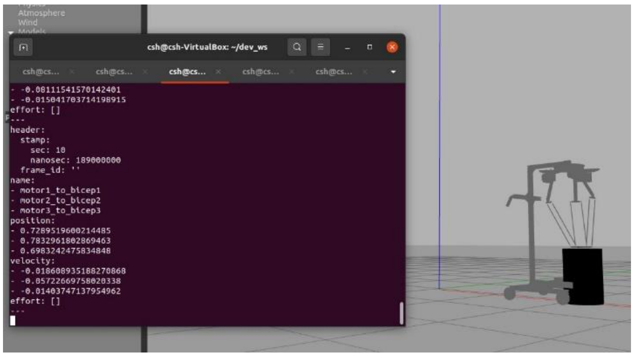
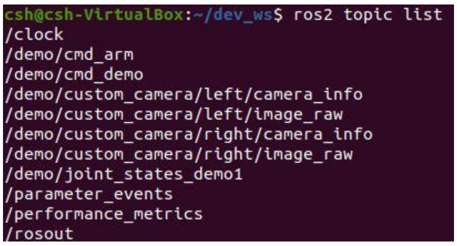


Fig. 10. Joints position when end-effector at Z=550mm.



```
csh@csh-VirtualBox:~/dev_ws$ ros2 topic list
/clock
/demo/cmd_arm
/demo/cmd_demo
/demo/custom_camera/left/camera_info
/demo/custom_camera/left/image_raw
/demo/custom_camera/right/camera_info
/demo/custom_camera/right/image_raw
/demo/joint_states_demo1
/parameter_events
/performance_metrics
/rosout
```

Fig. 11. ROS 2 topic list.

From the result obtained, the average percentage error of Θ1 is 3.92 %, Θ2 is 3.72 %, and Θ3 is 2.92 %. In this project, several topics have been published by different Gazebo plugins implementation. All the ROS 2 topics published are shown in Fig. 11 by running the command line: *ros2 topic list*The relationship between each topic and node can be determined through the rqt_graph, as shown in Fig. 12.

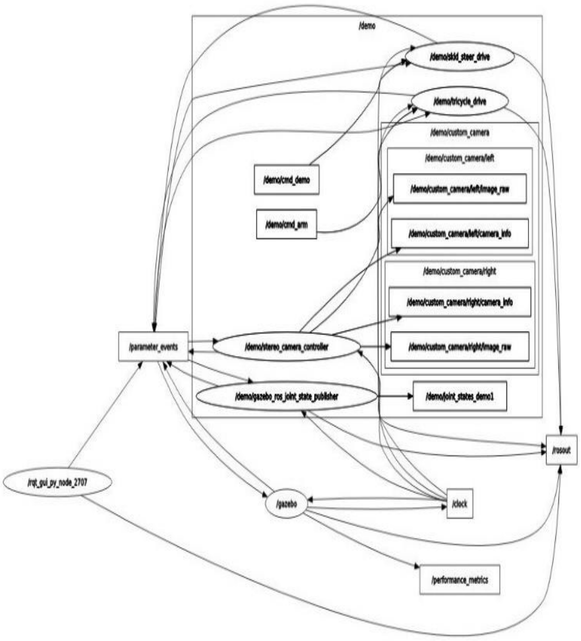


Fig. 12. The rqt graph.

TABLE V. COMPARISON BETWEEN THEORETICAL VALUE AND MEASURED VALUE.

End-effector position along Z-axis (mm)	Theoretical value (rad)	Measured value + offset angle (rad)	Percentage error (%)
550	Θ1= 0.9752 Θ2= 0.9752 Θ3= 0.9752	Θ1= 1.0042 Θ2= 1.0066 Θ3= 0.9941	Θ1= 2.97 Θ2= 3.22 Θ3= 1.94
600	Θ1= 0.8140 Θ2= 0.8140 Θ3= 0.8140	Θ1= 0.8566 Θ2= 0.8440 Θ3= 0.8358	Θ1= 5.23 Θ2= 3.69 Θ3= 2.68
650	Θ1= 0.6713 Θ2= 0.6713 Θ3= 0.6713	Θ1= 0.6977 Θ2= 0.6711 Θ3= 0.6799	Θ1= 3.93 Θ2= 0.03 Θ3= 1.28
700	Θ1= 0.5373 Θ2= 0.5373 Θ3= 0.5373	Θ1= 0.5474 Θ2= 0.5086 Θ3= 0.5417	Θ1= 1.88 Θ2= 5.34 Θ3= 0.82
750	Θ1= 0.4065 Θ2= 0.4065 Θ3= 0.4065	Θ1= 0.4293 Θ2= 0.3809 Θ3= 0.4386	Θ1= 5.61 Θ2= 6.30 Θ3= 7.90

IV. CONCLUSION AND FUTURE WORKS

In conclusion, the delta robot arm was successfully emulated in the ROS 2 Foxy Fitzroy environment, created using the VirtualBox software. To facilitate the formation of closed-loop kinematic chain linkage of delta robot, the primary robot structure description file, Unified Robot Description Format (URDF), was developed using SOLIDWORKS software and converted to Simulation Description Format (SDF). Adding a stereo camera link to the delta robot arm's end-effector and integrating the stereo camera Gazebo plugin allows the delta robot to visualize stereo images. The stereo camera's left and right images are displayed using Rviz 2. The simulation of the delta robot arm's kinematic motion is performed by implementing the *tricycle_drive* Gazebo plugin that controls three motors. The kinematic motion is controlled through the velocity command sent to the topic published by the plugin to control the rotation of joints connected between motors and biceps. By implementing the *tricycle_drive* Gazebo plugin, Θ1 has an average error percentage of 3.92 %, Θ2 has 3.72 %, while Θ3 has 2.92 % during the performance of inverse kinematic motion.

For future works, a customized gazebo plugin can be built for better improvements specially to automate full control of each of the delta robot arm's three motors. When the motors can be controlled individually, the accuracy of the kinematic motion performed by the delta robot arm may be increased by communicating the necessary angular rotational degree to each of the motors and allowing them to rotate accordingly. The customized gazebo plugin can then be contributed to the ROS2 community via GitHub, allowing other developers to perform a similar function in their project using the same code. Other than that, the left and right images captured by the stereo camera can be further processed by using *cv_bridge* to convert the ROS images to OpenCV images to extract the information of the images, thus allowing the user to determine the distance between the stereo camera and the target object. The real-time distance identified can be inserted into the customized Gazebo plugin code to allow automatic kinematic motion of the delta robot arm based on the object's coordinate.

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] B. William, "An Introduction to Robotics," March 2021. [Online]. Available:https://www.ohio.edu/mechanical-faculty/williams/html/PDF/IntroRob.pdf

[2] R. A. Beasley, "Medical Robots: Current Systems and Research Directions," *Journal of Robotics*, vol. 2012, 2012.

[3] P. Gonzalez-de Santos, R. Fernández, D. Sepúlveda, E. Navas, L. Emmi, and M. Armada, “Field Robots for Intelligent Farms-Inhering Features from Industry,” *Agronomy*, vol. 10, no. 11, p. 1638, 2020.

[4] K. D. Sowjanya, R. Sindhu, M. Parijatham, K. Srikanth, and P. Bhargav, "Multipurpose autonomous agricultural robot," *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, 2017, pp. 696–699.

[5] T. W. Bank, "Agriculture and Food," April 2021. [Online]. Available: https://www.worldbank.org/en/topic/agriculture/overview

[6] "Selected Agricultural Indicators, Malaysia, 2020," Department of Statistics Malaysia Official Portal, May 2021. [Online]. Available: https://www.dosm.gov.my/v1/index.php?r=column/cthemeByCat&cat=72&bul_id=RXVKUVJ5TitHM0cwYWxlOHcxU3dKdz09&menu_id=Z0VTZGU1UHBUT1VJMFlpaXRRR0xpdz09

[7] R. Rahmadian and M. Widyartono, "Autonomous Robotic in Agriculture: A Review," *2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE)*, pp. 1–6, 2020.

[8] R. Lallensack, "Five Roles Robots Will Play in the Future of Farming." May 2021. [Online]. Available: https://www.smithsonianmag.com/innovation/five-roles-robots-will-play-future-farming-180973242/

[9] T. H. T. Tran, D. S. Nguyen, N. T. Vo, and H. N. Le, "Design of Delta Robot Arm based on Topology optimization and Generative Design Method," *2020 5th International Conference on Green Technology and Sustainable Development (GTSD)*, 2020, pp. 157–161.

[10] C. Joochim, A. Kunapinum, S. Kaewkorn, and P. Keeratiwintakorn, "Development of Pick and Place Delta Robot," *International Conference on Interactive Collaborative Learning*, pp. 475–486, 2020.

[11] "What is a stereo vision camera?" May 2021. [Online]. Available: https://www.e-consystems.com/blog/camera/what-is-a-stereo-vision-camera/

[12] P. Estefó, R. Robbes, J. Simmonds, and J. Fabry, "The Robot Operating System: Package Reuse and Community Dynamics," *Journal of Systems and Software*, vol. 151, pp. 226-242, 2018.

[13] R. Tellez, "What is Robot Operating System (ROS)," April 2021. [Online]. Available: https://www.theconstructsim.com/what-is-ros/

[14] H. Kutluca, "Robot Operating Systems 2 (ROS 2) Architecture," 2021. [Online]. Available: https://medium.com/software-architecture-foundations/robot-operating-system-2-ros-2-architecture-731ef1867776

[15] "What is ROS," September 2021. [Online]. Available: https://ubuntu.com/robotics/what-is-ros

[16] D. Foundation, "What is DDS?" September 2021. [Online]. Available: https://www.dds-foundation.org/what-is-dds-3/

[17] "ROS 2 Documentation: Foxy," September 2021. [Online]. Available: https://docs.ros.org/en/foxy/Releases/Release-Foxy-Fitzroy.html

[18] A. Addison, "What is the difference between rviz and Gazebo?" September 2021. [Online]. Available: https://automaticaddison.com/what-is-the-difference-between-rviz-and-gazebo/

[19] "What is SDFormat," September 2021. [Online]. Available: http://sdformat.org/

[20] "Macro, URDF, SDF, Gazebo," September 2021. [Online]. Available: https://nu-msr.github.io/me495_site/lecture10_sdf_gazebo.html

[21] École polytechnique fédérale de Lausanne (EPLF) School of Engineering, “Reymond Clavel, creator of the delta robot, reflects on his career,” April 2021. [Online]. Available: https://sti.epfl.ch/reymond-clavel-creator-of-the-delta-robot-reflects-on-his-career/

[22] E. Coronado, M. Maya, A. Cardenas, O. Guarneros, and D. Piovesan, “Vision-based Control of a Delta Parallel Robot via Linear Camera-Space Manipulation,” *Journal of Intelligent & Robotic Systems*, vol. 85, no. 1, pp. 93–106, 2017.

[23] R. L. Williams II, "The Delta Parallel Robot: Kinematics Solutions," January 2016. [Online]. Available: https://www.ohio.edu/mechanical-faculty/williams/html/PDF/DeltaKin.pdf

[24] M. Opl, M. Houlb, J. Pavlik, F. Bradáˇc, P. Blecha, J. Kozubik, and J. Coufal, “DELTA - Robot with Parallel Kinematics,” *Mechatronics*, pp. 445-452, 2011.

[25] K.Waldron and J. Schmiedeler, "Handbook of Robotics Chapter 1: Kinematics," 2007.

[26] S. Kucuk and Z. Bingul, "Robot Kinematics: Forward and Inverse Kinematics," *Industrial Robotics: Theory, Modelling, and Control*, 2006.

[27] "Stereo Vision for 3D Machine Vision Applications," September 2021. [Online]. Available: https://www.clearview-imaging.com/en/blog/stereo-vision-for-3d-machine-vision-applications

[28] D. Rivas-Lalaleo, D. Tumbaco-Mendoza, E. Galarza-Zambrano, and W. Quimbita-Zapata, “Delta robot controlled by robotic operating system,” *ITECKNE Innovación e Investigación en Ingeniería*, vol. 12, no. 1, pp. 54-59, 2015.

[29] R. Barth, J. Baur, T. Buschmann, Y. Edan, T. Hellström, T.-T. Nguyen, O. Ring-dahl, W. Saeys, C. Salinas, and E. Vitzrabin, "Using ROS for agricultural robotics- design considerations and experiences," *Proceedings of the Second International Conference on Robotics and associated High-technologies and equipment for agriculture and forestry*, pp. 509-518, 2014.

[30] F. Okoli, Y. Lang, O. Kermorgant, and S. Caro, “Cable-Driven Parallel Robot Simulation Using Gazebo and ROS,” *ROMANSY 22– Robot Design, Dynamics and Control*, pp. 288-295, 2019.

[31] A. Sears-Collins, "How to Simulate a robot using Gazebo and ROS 2," 2021. [Online]. Available: https://automaticaddison.com/how-to-simulate-a-robot-using-gazebo-and-ros-2/

[32] "Intel Realsense D400 Series Product Family Datasheet," November 2021. [Online]. Available: https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf

[33] "Rotary Delta Robot Forward/Inverse Kinematics Calculations," 2021. [Online]. Available: https://www.marginallyclever.com/other/samples/fk-ik-test.html

[34] Y. Kadam, R. Ajabe, T. More, M. Jangam, and R. Patil, "Development of Delta Robot for Pick and Place on Moving Conveyor," *IJSRD (International Journal for Scientific Research & Development)*, vol. 7, no. 3, pp. 1387-1392, 2019.

[35] F. N. I. Ramlee and W. H. M. Saad, "Delta Robot Arm Simulation for Pick and Place the Object," *INOTEK 2021*, vol. 1, pp. 233–234, 2021.

[36] S. D. Perkasa, P. Megantoro, and H. A. Winarno, "Implementation of a Camera Sensor Pixy 2 CMUcam5 to A Two-Wheeled Robot to Follow Colored Object," *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, pp. 469–501, 2021.

[37] R. Giubilato, S. Chiodini, M. Pertile, and S. Debei, “An evaluation of ROS-compatible stereo visual SLAM methods on a nVidia Jetson TX2,” *Measurement*, vol. 140, pp. 161–170, Jul. 2019, doi: 10.1016/j.measurement.2019.03.038.

[38] E. Chebotareva and L. Gavrilova, “Educational Mobile Robotics Project ‘ROS-Controlled Balancing Robot’ Based on Arduino and Raspberry Pi,” *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, 2019, pp. 209–214, doi: 10.1109/DeSE.2019.00047.

[39] V. Marinoudi, C. G. Sørensen, S. Pearson, and D. Bochtis, “Robotics and labour in agriculture. A context consideration,” *Biosystems*

Engineering, vol. 184, pp. 111–121, Aug. 2019, doi: 10.1016/j.biosystemseng.2019.06.013.

[40] T. Duckett et al., "Agricultural Robotics: The Future of Robotic Agriculture," arXiv:1806.06762 [cs], Aug. 2018, Accessed: Apr. 07, 2022. [Online]. Available: http://arxiv.org/abs/1806.06762.