Implementation of Dual Quaternion-based Robot Forward Kinematics Algorithm in ROS

Conference Paper · June 2022

DOI: 10.1109/MECO55406.2022.9797160

4 authors:

Nikola Živković
Lola Institut
5 PUBLICATIONS 4 CITATIONS
SEE PROFILE

Jelena Vidakovic
Lola Institut
36 PUBLICATIONS 143 CITATIONS
SEE PROFILE

Stefan Mitrovic
Lola Institut
15 PUBLICATIONS 64 CITATIONS
SEE PROFILE

Mihailo P. Lazarević
University of Belgrade
150 PUBLICATIONS 1,956 CITATIONS
SEE PROFILE

Implementation of Dual Quaternion-based Robot Forward Kinematics Algorithm in ROS

Nikola Zivkovic
Robotics department
Lola Institute
Belgrade, Serbia
ORCID: 0000-0002-2276-2933

Jelena Vidakovic
Robotics department
Lola Institute
Belgrade, Serbia
ORCID: 0000-0002-3363-8807

Stefan Mitrovic
Robotics department
Lola Institute
Belgrade, Serbia
ORCID: 0000-0001-9857-3082

Mihailo Lazarevic
Faculty of Mechanical Engineering
University of Belgrade
Belgrade, Serbia
ORCID: 0000-0002-3326-6636

Abstract—**This paper presents a method for the implementation of a robot forward kinematics algorithm that complies with the Denavit-Hartenberg (DH) convention in Robot Operating System (ROS). The integration of the algorithm in ROS is based on the representation of DH parameters in dual quaternion space. The main motivation for the presented research is to make use of ROS powerful visualization tools available in tasks that require robot forward kinematics calculation while keeping the principles of DH robot modeling convention. Implementation of the dual quaternion-based robot forward kinematics algorithm in ROS is demonstrated using a serial 6DoFs robot as an example.**

Keywords—robot, forward kinematics, ROS, dual quaternions, Denavit-Hartenberg

I. INTRODUCTION

Robot Operating System (ROS) is a set of open-source software libraries and tools intended for building robot applications [1]. ROS is a popular tool in the robotic community due to its extensive tools, libraries, and conventions which enable simplified design, simulation, and control of robots.

Robot forward kinematics algorithm is used in various tasks such as robot applicative programming, prediction of potential collisions with obstacles, sensor signal processing algorithms, design of path planner, path optimization, robot control strategies, etc. Regarding verification of programmed robot trajectories, 3D visualization tools and simulation systems may significantly facilitate a robot programming process. ROS provides free and open-source tools that enable the visualization of 3D robots' models and their movement.

Denavit-Hartenberg (DH) convention is a convention for selecting frames of reference and corresponding DH parameters that describe an open kinematic chain of a robot manipulator. DH convention is widely used in the robotics community due to the ability to describe a pose/movement in terms of four variables only, for every link/joint (minimal set of parameters). Conventionally, DH convention is integrated into a solution of robot forward kinematic problem by using homogeneous transformation matrices (HTM) and matrix multiplication [2].

To represent the robot link's pose/movement in ROS, unit quaternions are used to describe attitude/rotation and a 3D vector is used to describe the position/translation of the attached coordinate frame. In order to make the best use of the visualization tools available with ROS in tasks that include forward kinematic calculation for a robot model that has been developed according to rules of the DH convention, an elegant solution based on the application of unit dual quaternions and the use of third–party graphic modeler is proposed herein. The solution is founded on the fact that the representation of the DH parameters in unit dual quaternion space is straightforward [3]. Consequently, a rigid transformation between adjacent joints is defined by dual quaternion multiplication of the four unit dual quaternions corresponding to four defined DH parameters. Furthermore, complete rigid transformation is described by a single mathematical object, which has advantages in developing various robotic algorithms [3].

The paper is organized as follows: Section II gives a brief introduction to the unit dual quaternions operators and their role in representing rigid transformations. Description of the unit dual quaternion-based robot forward kinematics algorithm which complies with the rules of DH convention is presented herein. In Section III, the implementation of the previously defined algorithm in ROS is described, including creating a 3D model, URDF file and the executable dual quaternion transform publisher node file that handles computation of the forward kinematics and communication with Rviz [4]. In Section IV, concluding remarks are given.

II. FORWARD KINEMATICS ALGORITHM BASED ON DH CONVENTION IN UNIT DUAL QUATERNIONS SPACE

A. Unit dual quaternions

Dual quaternions are a combination of dual numbers and quaternions. Rigid transformations can be represented by unit dual quaternions in the compact form of the screw motion [5-

7]. A dual quaternion consists of a primary and a dual part which are both unit quaternions. The primary part of the unit dual quaternion can be used to describe a rigid rotational transformation, while the unit dual part is equivalently used to describe a rigid translational transformation:

$$\mathbf{h}=\mathbf{h}^{\text{primary}}+\varepsilon\mathbf{h}^{\text{dual}}, \qquad (1)$$

where ε is the dual operator. According to [8-9], the primary and dual part of the unit dual quaternion can be computed using the following equations:

$$\mathbf{h}^{\text{primary}}=\mathbf{q}_{\mathbf{u}}^{\text{rot}}, \qquad (2)$$

$$\mathbf{h}^{\text{dual}}=0.5\left(\mathbf{q}_{\mathbf{u}}^{\text{rot}}\odot\mathbf{q}_{\mathbf{t}}^{\text{trans}}\right), \qquad (3)$$

$$\mathbf{q}_{\mathbf{u}}^{\text{rot}}=\left[\cos\left(\frac{\theta}{2}\right),\ \sin\left(\frac{\theta}{2}\right)u_x, \sin\left(\frac{\theta}{2}\right)u_y, \sin\left(\frac{\theta}{2}\right)u_z\right], \qquad (4)$$

$$\mathbf{q}_{\mathbf{t}}^{\text{trans}}=\left[0,\ mu_x,\ mu_y,\ mu_z\right], \qquad (5)$$

where the symbol $\odot$ represents the unit quaternion multiplication operator. Angle θ is an angle of rotation about an axis in the direction of an unit vector $\mathbf{u}=[u_x, u_y, u_z]$, and m is a scalar representing translation along the axis in the direction of an unit vector $\mathbf{t}=[t_x, t_y, t_z]$.

Dual quaternion multiplication is the basis for the forward kinematics algorithm development. Multiplication operation with unit dual quaternions as operands allows for a composition of rigid transformations that results in a simple computation of all links' poses, including the end-effector [5]. Dual quaternion multiplication can be written in the following manner:

$$\mathbf{h}_1\otimes\mathbf{h}_2=\mathbf{h}_1^{\text{p}}\odot\mathbf{h}_2^{\text{p}}+\varepsilon\left(\mathbf{h}_1^{\text{p}}\odot\mathbf{h}_2^{\text{d}}+\mathbf{h}_1^{\text{d}}\odot\mathbf{h}_2^{\text{p}}\right), \qquad (6)$$

where $\mathbf{h}_1, \mathbf{h}_2$ are dual quaternions defined as:

$$\mathbf{h}_1= \mathbf{h}_1^{\text{p}}+ \varepsilon\mathbf{h}_1^{\text{d}}, \qquad (7)$$

$$\mathbf{h}_2=\mathbf{h}_2^{\text{p}}+\varepsilon\mathbf{h}_2^{\text{d}}. \qquad (8)$$

In (6), (7) and (8), superscripts "p" and "d" denote primary and dual parts, respectively. The symbol $\otimes$ in (6) represents unit dual quaternion multiplication.

B. Forward kinematics algorithm based on DH convention

In modeling the kinematics of a serial-link robot, Denavit-Hartenberg notation is still the most widely used due to its intuitive and concise geometric interpretation of a robot [10]. Minimal parameter requirement for describing rigid transformation between adjacent robot frames is the most significant advantage of DH convention, which is traditionally applied by using HTM [2]. Considering that a single dual quaternion can represent a translation followed by a rotation or vice versa, the application of DH convention is straightforward in dual quaternion space. According to DH convention, a rigid transformation from (i−1)-st to i-th frame is defined by the following matrix equation [11], as illustrated in Fig. 1:

$$\mathbf{T}_{i\text{-}1}^{i}=\mathbf{T}_{\text{rot}}(\text{z}_{i\text{-}1},\theta_i)\mathbf{T}_{\text{trans}}(\text{z}_{i\text{-}1},\text{d}_i)\mathbf{T}_{\text{trans}}(\text{x}_i,\text{a}_i)\mathbf{T}_{\text{rot}}(\text{x}_i,\alpha_i), \qquad (9)$$

where $\mathbf{T}_{i\text{-}1}^{i}$ is HTM which maps homogeneous vector described in coordinate frame i to its representation in coordinate frame (i−1). Matrix $\mathbf{T}_{\text{rot}}(\text{z}_{i\text{-}1},\theta_i)$ describes rotation about $\text{z}_{i\text{-}1}$ axis for an angle θ_i , $\mathbf{T}_{\text{trans}}(\text{z}_{i\text{-}1},\text{d}_i)$ stands for translation along $\text{z}_{i\text{-}1}$ axis by the distance d_i, $\mathbf{T}_{\text{trans}}(\text{x}_i,\text{a}_i)$ is a translation along x_i axis by the distance a_i and $\mathbf{T}_{\text{rot}}(\text{x}_i,\alpha_i)$ is a rotation about x_i axis for an angle α_i. In (9), it is assumed that the robot joint is revolute. Herein, to use DH convention in dual quaternion space, matrices on the right side of (9) that account for rotation about and a translation along $\text{z}_{i\text{-}1}$ are represented by a single unit dual quaternion $\mathbf{h}_{\text{z}_{i\text{-}1}}$, and similarly rotation about and translation along x_i axis are represented with another single unit dual quaternion $\mathbf{h}_{\text{x}_i}$ in the following manner:

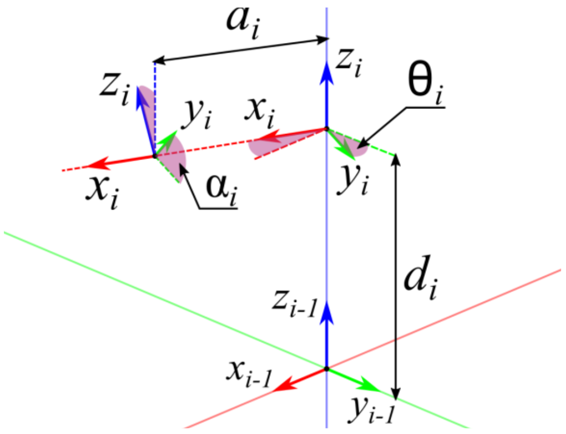

Figure 1. Coordinate frame transformation according to the DH convention.

$$\mathbf{h}_{\text{z}_{i\text{-}1}}=\mathbf{h}_{\text{rot}}(\text{z}_{i\text{-}1},\theta_i)\mathbf{h}_{\text{trans}}(\text{z}_{i\text{-}1},\text{d}_i)=$$
$$=\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{rot}}+0.5\varepsilon\left(\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{rot}}\odot\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{trans}}\right), \qquad (10)$$

$$\mathbf{h}_{\text{x}_i}= \mathbf{h}_{\text{trans}}(\text{x}_i,\text{a}_\text{i})\mathbf{h}_{\text{rot}}(\text{x}_i,\alpha_i)=$$
$$=\mathbf{h}_{\text{x}_i}^{\text{rot}}+ 0.5\varepsilon\left(\mathbf{h}_{\text{x}_i}^{\text{rot}}\odot\mathbf{h}_{\text{x}_i}^{\text{trans}}\right), \qquad (12)$$

where $\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{rot}}$, $\mathbf{h}_{\text{x}_i}^{\text{rot}}$, $\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{trans}}$ and $\mathbf{h}_{\text{x}_i}^{\text{trans}}$ are defined as:

$$\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{rot}}=\left[\cos\left(\frac{\theta_i}{2}\right),0,0,\sin\left(\frac{\theta_i}{2}\right)\right], \qquad (13)$$

$$\mathbf{h}_{\text{x}_i}^{\text{rot}}= \left[\cos\left(\frac{\alpha_i}{2}\right),\sin\left(\frac{\alpha_i}{2}\right),0,0\right], \qquad (14)$$

$$\mathbf{h}_{\text{z}_{i\text{-}1}}^{\text{trans}}=[0,\ 0,\ 0,\ \text{d}_i], \qquad (15)$$

$$\mathbf{h}_{\text{x}_i}^{\text{trans}}=[0,\text{a}_i,\ 0,\ 0]. \qquad (16)$$

Consequently, to represents a rigid transform from (i−1)-st to i-th frame, a unit dual quaternion obtained by the dual quaternion multiplication of the two unit dual quaternions $\mathbf{h}_{\text{z}_{i\text{-}1}}$ and $\mathbf{h}_{\text{x}_i}$ can be used:

$$\mathbf{h}_{i\text{-}1}^{i}=\mathbf{h}_{\text{z}_{i\text{-}1}}\otimes\mathbf{h}_{\text{x}_i}. \qquad (17)$$

To define the relationship between the end-effectors frame and base frame following equation is used:

$$\mathbf{h}_{\text{end}}=\mathbf{h}_0^1\otimes\ \mathbf{h}_1^2\otimes\ \ldots\otimes\ \mathbf{h}_{n\text{-}1}^{n}, \qquad (18)$$

where n is the degree of freedom of the robot manipulator. The position and the orientation of the end-effector can be extracted from $\mathbf{h}_{\text{end}}$ via the following equations:

$$\mathbf{v}_{\text{position}}=2\left(\mathbf{h}_{\text{end}}^{\text{d}}\odot\overline{\mathbf{h}}_{\text{end}}^{\text{p}}\right), \qquad (19)$$

$$\mathbf{v}_{\text{orientation}}=\mathbf{h}_{\text{end}}^{\text{p}}, \qquad (20)$$

where $\overline{\mathbf{h}}_{\text{end}}^{\text{p}}$ represents a conjugate of the dual quaternion $\mathbf{h}_{\text{end}}^{\text{p}}$ [8].

III. Integration of the Algorithm in ROS

In this section, the integration of the presented robot forward kinematics algorithm in ROS is described. Implementation of the algorithm in ROS is demonstrated using a serial 6-DoFs robot LIRL15 (Lola15 IRL) as an example.

A. ROS, RViz and URDF file

ROS is a distributed framework of processes (aka nodes) that enables executables to be individually designed and loosely coupled at runtime [1]. ROS operates using nodes, messages and topics. RViz [4] is the 3D visualization tool for ROS applications. ROS uses Unified Robot Description Format (URDF) file to generate the robot's graphical representation within the dedicated visualizer (RViz) and to assign robot joints and links with specific mechanical characteristics. URDF is a type of XML format used to describe a robot's main features, including kinematic (links and joints), inertial, visual and other properties.

URDF can be used to "manually" build a relatively simple representation of a robot, but such an approach is tedious and error-prone, best suited for rapid testing. The alternative to making a robot model with URDF elements from scratch is to import already modeled meshes into RViz via the URDF file, using the element "<mesh>" which specifies the path where the mesh file is stored. Dedicated robot models developed in third-party graphic modelers (Blender, Maya, SolidWorks, CATIA, etc.) may be imported as meshes, thus simplifying the workflow and providing better precision in the model development. URDF file can also be generated via end-user developed converters using the DH parameters as input [12]. However, aforementioned converters don't have an option to import mesh models.

Herein, it is opted to model meshes in Blender [13], an open-source 3D design software that best suits our needs during the present development phase. Each link is modeled as a separate file and exported in the Digital Asset Exchange (DAE) file format, which ROS supports. The coordinate frame for each link is positioned and orientated in Blender following the rules of the DH convention, thus bypassing the need to define DH parameters within the URDF file. Examples are shown in Fig. 2. Assigned coordinate systems are rigidly attached to the respective links.

B. Dual quaternion transform publisher node

The dual quaternion transform publisher node's main purpose is to, based on the robot description in the URDF file and joint input angles, calculate the poses of all links via the forward kinematics algorithm and parse that information to RViz. The dual quaternion transform publisher developed within this study is a node similar to the native ROS robot state publisher node, except that it uses dual quaternions to represent a rigid transformation instead of 3×3 rotation matrices. The advantage of dual quaternions over rotation matrices is used less space to store information and dual quaternions encapsulate translation and rotation into single object.

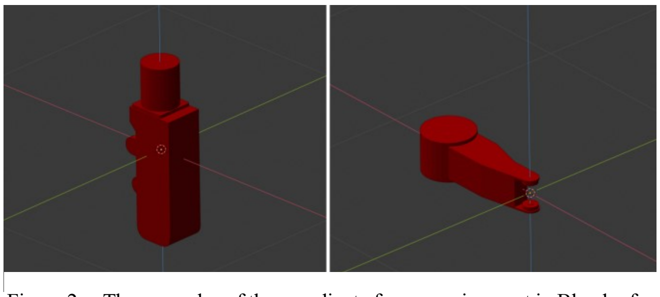
Figure 2. The examples of the coordinate frames assignment in Blender for LIRL15 robot.

Publishing in the case of the ROS implies making an information available to another node via different topics and messages. To make frame transformations calculated via the above-mentioned forward kinematics algorithm available in RViz, a message of the type geometry_msgs/TransformStamped that defines rigid transformation between two successive coordinate frames is used. The message geometry_msgs/TransformStamped consists of the following items [14]: 1) time stamp, which is incremented and updated through infinite loop; 2) string which stores the name of the parent coordinate frame; 3) string for the name of the child coordinate frame; 4) message that describes the pose of the child coordinate frame with respect to the parent frame, as a vector-quaternion pair. Information for the items 2) and 3) is obtained from the URDF file, time stamp is assigned with the built-in ROS time and finally pose information is calculated via the dual quaternion-based forward kinematics algorithm as shown in the diagram in Fig. 3. In the dual quaternion transform publisher node, the array of objects of the geometry_msgs/TransformStamped type is declared, where each element of the array corresponds to a robot link. The names of the robot links are parsed from the URDF file to the array of geometry_msgs/TransformStamped messages so that child coordinate frame represents *i*-th

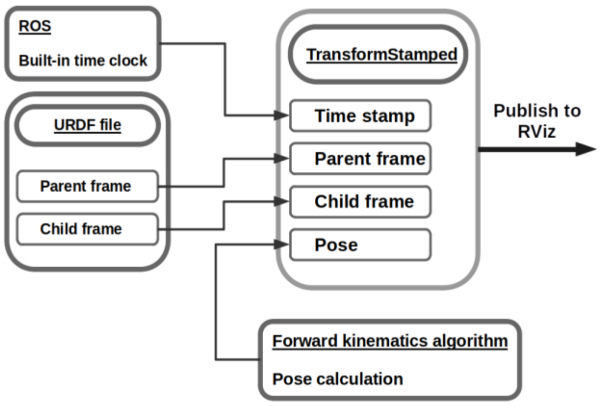

Figure 3. Data flowchart in dual quaternion transform publisher node

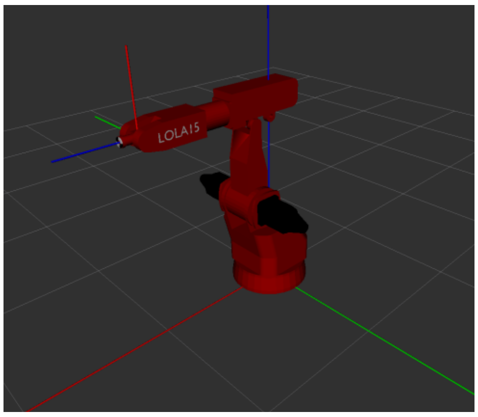

Figure 4. LIRL15 model at the initial configuration in RViz.

coordinate frame and parent frame represents the (i-1)-st frame, according to the DH convention. Essentially child frame is the frame that is rigidly attached to the corresponding link, while the parent frame is the previous frame in the serial robot chain. This method of coordinate frame names assignment allows RViz to use information from the URDF file to visualize 3D models of the robot. The assignment of the coordinate frame names in the above described manner sets the robot LIRL15 at the initial configuration as presented in Fig. 4.

The part of the geometry_msgs/TransformStamped object that stores the information about pose is assigned with the values from the result of the previously described dual-quaternion based forward kinematics algorithm. The forward kinematics is computed in the loop of the dual quaternion transform publisher node in real-time, for each time instance. The updated array of the geometry_msgs/TransformStamped messages is sent to RViz to visualize robot's movement in each loop instance. Information of the end-effector pose is printed in real-time via Command Line Interface (CLI).

The robot trajectory is defined and parsed in the joint space, meaning each joint is commanded to rotate or translate depending on the joint type. RViz can render the motion path of the coordinate frame of choice, so choosing the end-effector's frame, its trajectory in the task space is visualized for the given input joint angles as shown in Fig. 5.

The library is created for computing of robot forward kinematics via unit dual quaternions, and it support necessary data structures and operations required for dual quaternion algebra. For example, the multiplication operator is overloaded to represent the dot product if the operands are of the vectors. Similarly, if operands are unit quaternions, the operator executes unit quaternion multiplication, and the same analogy applies when the operands are unit dual quaternions. The described TransformDualQuaternion library enables dual quaternion transform publisher node to compute forward kinematics and simultaneously move the robot joints for the given input joint angles in RViz.

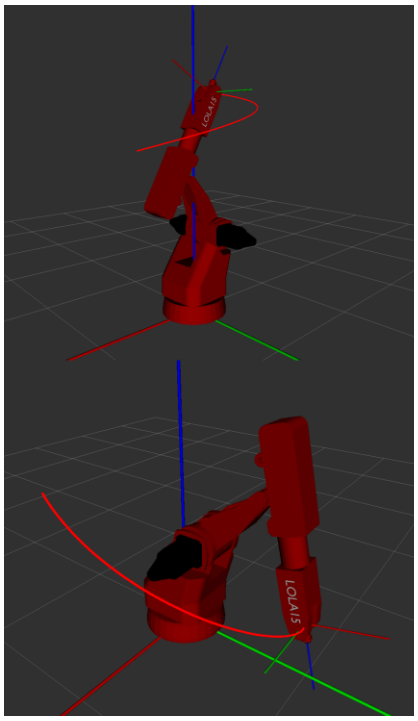

Figure 5. The examples of LIRL15 end-effector trajectory for given joint angles input

IV. Conclusion

This paper presents an integration of robot forward kinematics algorithm based on the dual quaternion representation of Denavit-Hartenberg convention in ROS. The goal of the presented research is to enable the full advantage of ROS associated visualization tools regarding 3D robot model development and movement simulation in robot tasks that include robot forward kinematics calculation, while keeping the principles of DH convention model development. Implementation of dual-based robot direct kinematics algorithm in ROS is demonstrated using a serial 6-DoFs robot LIRL15 as an example.

Acknowledgment

This research has been supported by the research grants of the Serbian Ministry of Education, Science and Technological Development, grant No. 451-03-68/2020-14/200105 and grant No. 451-03-68/2022-14/200066.

References

[1] Stanford Artificial Intelligence Laboratory (2018) Robotic Operating System. Available at: https://www.ros.org [Accessed 2 February 2021].

[2] J. Denavit and R. S. Hartenberg, "A kinematic notation for the lowerpair mechanism based on matrices," ASME J. Appl. Mech., pp. 215–221, June 1955.

[3] B. V. Adorno, "Two-arm manipulation: From manipulators to enhanced human-robot collaboration", Ph.D. dissertation, Université Montpellier II-Sciences et Techniques du Languedoc, 2011.

[4] *RViz*. (2022). [Online]. Available: https://github.com/ros-visualization/rviz.git

[5] B. V. Adorno, "Robot Kinematic Modeling and Control Based on Dual Quaternion Algebra - Part I: Fundamentals.," 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01478225v1

[6] J. Funda, "A Computational Analysis of Line-Oriented Screw Transformations in Robotics", Technical Reports (CIS), 1988, 665.

[7] J. Funda, R. H. Taylor and R. P. Paul, "On homogeneous transforms, quaternions, and computational efficiency," in IEEE Transactions on Robotics and Automation, vol. 6, no. 3, pp. 382-388, June 1990, doi: 10.1109/70.56658.

[8] J. Vidakovic, M. Lazarevic, V. Kvrgic, Z. Dancuo, G. Ferenc, "Advanced Quaternion Forward Kinematics Algorithm Including Overview of Different Methods for Robot Kinematics", FME Transactions, 2014, vol. 42, no. 3, pp. 189-199, doi: 10.5937/fmet1403189v

[9] B. Kenwright, "A Beginners Guide to Dual-Quaternions: What They Are, How They Work, and How to Use Them for 3D Character Hierarchies", *The 20th International Conference on Computer Graphics, Visualization and Computer Vision*, 2012, pp. 1-13.

[10] Wu, Liao, Ross Crawford, and Jonathan Roberts. "Geometric interpretation of the general POE model for a serial-link robot via conversion into DH parameterization." 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019.

[11] M. W. Spong, S. Hutchinson, and M. Vidyasagar, "Forward Kinematics" in Robot Modeling and Control 2nd ed, US:Wiley, 2020, ch. 3, sec. 3.2, pp. 79.

[12] Y. Kang, D. Kim and K. Kim, "URDF Generator for Manipulator Robot," 2019 Third IEEE International Conference on Robotic Computing (IRC), 2019, pp. 483-487, doi: 10.1109/IRC.2019.00101.

[13] *Blender*. (2022). [Online]. Available: https://www.blender.org/

[14] Anon, Geometry_msgs/transformstamped message. geometry_msgs/TransformStamped Documentation. Available at: http://docs.ros.org/en/lunar/api/geometry_msgs/html/msg/TransformStamped.html [Accessed March 13, 2022].