The Dual Quaternion Algebra and its Implementation in Asymptote Language

Migran N. Gevorkyan[1], Dmitry S. Kulyabov[1,2,*], Anna V. Korolkova[1] and Leonid A. Sevastianov[1,2]

[1]*RUDN University, 6 Miklukho-Maklaya St, Moscow, 117198, Russian Federation*
[2]*Joint Institute for Nuclear Research, 6 Joliot-Curie St, Dubna, 141980, Russian Federation*

Abstract
The algebras of dual quaternions and screws are often opposed to geometric algebra. The purpose of this paper is to describe the algebra of dual quaternions and the algebra of screws, to give a number of examples of the use of dual quaternions to describe the screw motion of points, lines and planes in three-dimensional space. This algebra is very poorly covered in the literature, and the actively used principle of Kotelnikov-Study transfer is apparently forgotten. All calculations were performed using the Asymptote language. Structures were created that implement dual numbers, quaternions, and dual dual quaternions, as well as a set of computational tests to verify these structures.

Keywords
screws, motors, rotations, translations, computer geometry, Asymptote

1. Introduction

In the course of research on the application of analytical projective geometry in the field of computational geometry [2], the authors often came across references to motors, propellers, and dual dual quaternions. All mentions were very brief and basically boiled down to the fact that the mentioned entities are extremely unintuitive and difficult to understand and use. They were often contrasted with geometric algebra methods, which were presented as more understandable and logical [4].

The search for a detailed description of the mathematical apparatus of the algebra of screws and dual quaternions led the authors to works in the field of mechanics of absolutely rigid bodies. It turned out that the theory of screws was developed back in the late 19th and early 20th centuries in the works of R. S. Ball, E. Study [3], A. P. Kotelnikov. The most complete description can be found in the monograph [1]. However, at present this theory is little known and there are practically no software implementations of screw algebras and dual quaternions.

Another methodological problem is the lack of examples of the application of screws and dual quaternions to computer geometry problems. The sources found are mainly focused on the problems of applied mechanics. In this paper, we have tried to at least partially eliminate this shortcoming.

Since all the examples are focused on the application of dual quaternion algebras and screws to geometric problems, the Asymptote language was chosen as the language for implementation. This language allows you to create custom data structures (data types) and overload all the basic

6^{th} *International Conference "Computer Algebra", Moscow, June 23–25, 2025*
10.22363/12585-2025-6-011
EDN: BNYSRN
*Corresponding author.
gevorkyan-mn@rudn.ru (M. N. Gevorkyan); kulyabov-ds@rudn.ru (D. S. Kulyabov); korolkova-av@rudn.ru (A. V. Korolkova); sevastianov-la@rudn.ru (L. A. Sevastianov)
https://yamadharma.github.io/ (D. S. Kulyabov)
0000-0002-4834-4895 (M. N. Gevorkyan); 0000-0002-0877-7063 (D. S. Kulyabov); 0000-0001-7141-7610 (A. V. Korolkova); 0000-0002-1856-4643 (L. A. Sevastianov)

algebraic operators. This made it possible to create data types for dual numbers, quaternions, and dual dual quaternions. You can also create second-order functions that return other functions that implement screw motion using Roderig's screw formulas and sandwich products of dual quaternions.

2. Dual numbers and quaternions

Here we list the basic concepts used in the construction of dual quaternionic algebra.

A dual number is a parabolic complex number $z = a + b\varepsilon$, where the imaginary unit is a parabolic imaginary unit defined by the equality $\varepsilon^2 = 0$. For these numbers, you can define the same operations as for the usual (elliptical) complex numbers.

For two numbers $z_1 = a_1 + b_1\varepsilon$ and $z_2 = a_2 + b_2\varepsilon$, addition, subtraction, and multiplication can be defined. The formula for multiplication will look like this:

$$z_1 z_2 = a_1 a_2 + (a_1 b_2 + b_1 a_2)\varepsilon,$$

A dual conjugation is a number $\overline{z} = \overline{a + b\varepsilon} = a - b\varepsilon$. The square of the module of the number $|z|^2 = z\overline{z} = a^2$, and the module itself as $|z| = |a|$.

Quaternion is a hypercomplex number of the form $q = q_0 + q_1 i + q_2 j + q_3 k = q_0 + \mathbf{q}$, where the imaginary units i, j, k are determined by the equality $i^2 = j^2 = k^2 = ijk = -1$. From this equality, we can obtain a multiplication table of i, j, k among themselves and define quaternion multiplication, which is most easily expressed in terms of the scalar and vector parts of the quaternion as follows:

$$qp = q_0 p_0 - (\mathbf{q}, \mathbf{p}) + q_0\mathbf{p} + p_0\mathbf{q} + \mathbf{q} \times \mathbf{p}.$$

A pure quaternion is a quaternion without a scalar part q_0. A pure quaternion is associated with a vector, an ordinary quaternion with $q_0 = 1$ is associated with an affine point, and a quaternion with $q_0 \neq 0.1$ is associated with a point mass. The q_0 component plays the role of a weight coordinate in this case.

3. Dual quaternions

A dual quaternion is a dual number with coefficients in the form of quaternions (Cayley-Dickson doubling procedure):

$$Q = q + q^o\varepsilon,$$

where the quaternion q is the main part, and q^o is the moment part. Q can be written as a number with eight components

$$Q = q_0 + q_1 i + q_2 j + q_3 k + q_0^o\varepsilon + q_1^o i\varepsilon + q_2^o j\varepsilon + q_3^o k\varepsilon.$$

Axiomatically, it is assumed that the parabolic imaginary unit ε commutes with elliptical imaginary units i, j, k, that is, $i\varepsilon = \varepsilon i$, $j\varepsilon = \varepsilon j$, $k\varepsilon = \varepsilon k$.

Simplifying somewhat, we will call a screw a dual quaternion, both parts of which are pure quaternions. We will denote the screw in bold: $\mathbf{Q} = \mathbf{q} + \mathbf{q}^o\varepsilon$.

Three different conjugation operations are defined for a dual quaternion

- $Q^* = (q + q^o\varepsilon) = q^* + q^{o*}\varepsilon$ — quaternionic (complex) conjugation;
- $\overline{Q} = \overline{q + q^o\varepsilon} = q - q^o\varepsilon$ — dual conjugation;
- $Q^\dagger = (\overline{q + q^o\varepsilon})^* = q^* - q^{o*}\varepsilon$ is a quaternion dual conjugation.

dual quaternion multiplication can be defined for dual quaternions, and scalar and screw (vector) multiplications for screws.

4. The principle of Kotelnikov–Study transference

The principle of transference in the form in which A. P. Kotelnikov formulated it states. All formulas of the theory of finite rotations and kinematics of motion of a rigid body with one fixed point, when replacing real quantities in them with dual analogues, turn into formulas for finite displacements and kinematics of motion of a free rigid body.

For example, consider the Rodrigues formula for rotating a point P with a radius vector $\mathbf{p}$ around an axis passing through the origin with a guide vector $\mathbf{a}$ by an angle θ:

$$\mathbf{p}' = \cos\theta\mathbf{p} + \sin\theta\mathbf{a}\times\mathbf{p} + (1-\cos\theta)(\mathbf{a},\mathbf{p})\mathbf{a}.$$

According to the principle of transfer, the angle θ should be replaced in this formula by the dual angle $\Theta = \theta + \theta^o\varepsilon$, the radius vector $\mathbf{p}$ by the screw $\mathbf{L} = \mathbf{v} + \mathbf{m}\varepsilon$, guiding vector $\mathbf{a}$ onto screw $\mathbf{A} = \mathbf{a} + \mathbf{a}^o\varepsilon$. The scalar and vector product of the vectors will then be replaced by the scalar and screw product of the screws.

5. Dual quaternion formulas for screw motion

If in the unit quaternion λ for the rotation of the vector (pure quaternion) $\mathbf{p}$ around the axis $\mathbf{a}$ is given by the formula $\lambda = \cos\dfrac{\theta}{2} + \sin\dfrac{\theta}{2}\mathbf{a}$, then replacing the angle θ and the vector $\mathbf{a}$ with dual analogues using the principle of transference, then we get a dual quaternion: $\Lambda = \cos\dfrac{\Theta}{2} + \sin\dfrac{\Theta}{2}\mathbf{A}$ which implements the screw movement of points, straight lines and planes.

The sandwich formula for a straight line represented by a screw $\mathbf{L} = \mathbf{v} + \mathbf{m}\varepsilon$ looks like $\mathbf{L}' = \Lambda\mathbf{L}\Lambda^*$. An affine point is represented using a dual quaternion of the following form: $P = 1 + \mathbf{p}\varepsilon$. Planes can also be written as a dual quaternion $\Pi = \mathbf{n} + d\varepsilon$, where $\mathbf{n}$ is the normal vector of the plane, and d is the distance from the plane to the origin. The same formulas work for the screw motion of a point and a plane: $P' = \Lambda P\Lambda^\dagger$ and $\Pi' = \Lambda\Pi\Lambda^\dagger$.

6. Conclusion

The Kotelnikov-Study transference principle is naturally implemented programmatically if the types of dual numbers, quaternions, and dual quaternions are defined, as well as arithmetic and algebraic operators are overloaded, and scalar and vector multiplications are defined. In this case, the calculation of the screw motion is reduced to a compact program code, since all the computational complexity is already implemented in the created data types. And since the implementation of dual quaternions uses ready-made types of dual numbers and quaternions, part of the complexity is transferred to the implementation of these types, thus distributing the overall complexity at different levels. The specific details of the implementation are planned to be outlined in the presentation of the report.

References

[1] F. M. Dimentberg, The screw calculus and its applications in mechanics, 1968.

[2] M. N. Gevorkyan, A. V. Korolkova, D. S. Kulyabov, L. A. Sevastyanov, Implementation of analytic projective geometry for computer graphics, Programming and Computer Software 50 (2024) 153–165. doi:10.1134/S0361768824020075.

[3] F. Klein, Vorlesungen Über Höhere Geometrie, Springer Berlin Heidelberg, 1926. doi:10.1007/978-3-642-49848-0.

[4] E. Lengyel, Mathematics, volume 1, Terathon Software LLC, 2016. URL: http://foundationsofgameenginedev.com.