Quaternions: Applications in Computer Graphics

Ahmed Walid

Abstract- Quaternions are some of the most essential topics in computer graphics. The purpose of this article is to show some of the uses and advantages of using quaternions. It uses them to represent three-dimensional rotations in the image processing field. It starts by discussing what quaternions are and their mathematical properties. After that, it goes over their applications and uses.

Index Terms- Quaternions, Image processing, 3D Rotation, Vectors

I. INTRODUCTION

Quaternions are 4 dimensional numbers that were created by mathematician, William Rowan Hamilton. He has spent years trying to figure out how to multiply triplets, in the hope of inventing a three dimensional number system. It was only at some point when he realized that making it four dimensional. There has been a lot of debate on quaternions, where some mathematicians believed it is how three dimensional space should be taught. On the other hand, many mathematicians believed that quaternions were very complex to be the primary way of teaching it.

They, however, were found to be useful for three-dimensional rotation and motion graphics. Therefore, it is widely used by computer scientists in those fields. It is also used in many other areas, such as game development, 3D modelling and animations. This is because they make an excellent replacement for using matrices. That is because they describe three-dimensional space and how they represent different mathematical operations is less memory intensive than matrices.

II. QUATERNION ALGEBRA

Quaternions are hyper-complex numbers of 4 dimensions. They consist of 3 complex (vector) parts and one real (scalar) part. They can be represented in a few different notations as shown in **figures 1, and 2:**

$$q = [w\ v] \text{ or } q = [w\ (x\ y\ z)]$$

$$q = a + bi + cj + dk$$

Figure 1 shows quaternion notations

$$\begin{aligned} 1 &= (1, 0, 0, 0) \\ \mathbf{i} &= (0, 1, 0, 0) \\ \mathbf{j} &= (0, 0, 1, 0) \\ \mathbf{k} &= (0, 0, 0, 1) \end{aligned}$$

Figure 2 shows each part of a quaternion

Where in the first notation the w is the scalar part, while the v stands for the vector part of the quaternion, or the v is split into its three components, x, y, and z. In the second notation a is the real part of the quaternion, while i, j, and k are the complex parts. Please note that the two notations are going to be used throughout this paper. [3], [7]

The components $i, j\ and\ k$ as defined above satisfy the rules defined in **Figure 3**.

$$i^2 = j^2 = k^2 = ijk = -1$$
$$ij = -ji = k$$
$$jk = -kj = i$$
$$ki = ik = j$$

Figure 3 shows the definition of a quaternion

It was implied that the product of any two terms in this expression is a quaternion product, which is

$$p = (a_p, v_p) \text{ and } q = (a_q, v_q)$$

$$pq = (a_p a_q - v_p . v_q,\ a_p v_q + a_q v_p + v_p \times v_q)$$

Figure 4 shows quaternion product

defined for the two quaternions p and q as in **Figure 4,** where:

With a being the scalar part of the two quaternions and v the vector part. In addition to that, $v_q . v_p$ and $v_p \times v_q$ denote the dot and cross products respectively. From that we conclude that multiplication in quaternions is not a commutative – since the vector cross product is not – process.

Quaternion addition on the other hand is a very simple process, where the corresponding components of the quaternion are added together. This could be illustrated as in **Figure 5,** where p and q are two quaternions. [1], [3], [7]

$$p + q = (a_p + a_q, v_p + v_q)$$

Figure 5 shows quaternion addition

The identity element in Quaternions is, therefore, (0, 0, 0, 0).

Although Quaternion multiplication is not commutative, it has the associative and distributive properties over addition. Therefore, if we have p, q and r. This property could be shown as in **Figure 6**. [3], [7]

$$(pq)r = p(qr)$$
$$(p + q)r = pr + qr$$
$$p(q + r) = pq + pr$$

Figure 6 shows some arithmetic properties of quaternions

The magnitude of a quaternion can be defined as shown in **Figure 7**. [1], [2], [3]

$$\begin{aligned}\|\mathbf{q}\| &= \left\| \begin{bmatrix} w & (x \quad y \quad z) \end{bmatrix} \right\| = \sqrt{w^2 + x^2 + y^2 + z^2} \\ &= \left\| \begin{bmatrix} w & \mathbf{v} \end{bmatrix} \right\| = \sqrt{w^2 + \|\mathbf{v}\|^2}.\end{aligned}$$

$$|p| = \sqrt{\|p\|} = \sqrt{a^2 + b^2 + c^2 + d^2}$$

Figure 7 shows the magnitude of a quaternion.

A real quaternion is a quaternion where the vector part of it is equal to zero. On the other hand, a pure quaternion is the opposite, as its real part is equal to zero.

As mentioned before, Quaternion multiplication is associative and not commutative. It should also be known that the magnitude of a quaternion product is equal to the product of the magnitudes, as shown in **Figure 8**. [3], [5]

$$\|pq\| = \|p\|\|q\|$$

III. Quaternion Applications

Quaternions in Image Processing:

Quaternions are great for representing colors in memory compared to using matrices. Therefore, pure quaternions are usually used for representing RGB color images. Three channels are used for the Red, Blue, and Green values with the fourth channel used for transparency or padding. [1], [2]

For example, let $f(x, y)$ be an RGB image function as in **Figure 9**. Every pixel of the image can be represented as a pure quaternion. [1], [13]

$$f(x,y) = f_R(x,y)\boldsymbol{i} + f_G(x,y)\boldsymbol{j} + f_B(x,y)\boldsymbol{k}$$

Figure 8 shows a function that represents an RGB image as a pure quaternion

It could also be illustrated in **Figure 9**, how colors look on the quaternion space.

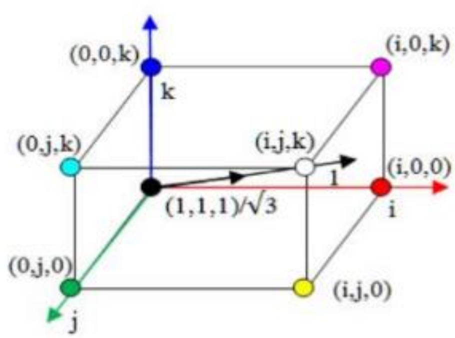

Figure 9 shows RGB colors in a quaternion space

The use of quaternions in color images, together with Fourier Transform – which will be explained later in this section – made it a lot easier for processing and studying color images. This is as they contribute to ideas such as edge detection and color sensitive filtering. [1], [4]

Fourier transform is a mathematical concept that transforms color images from the spatial domain to the frequency domain when applied. The reason this transformation happens is because it is a lot easier for computers to deal with images in the frequency domain. This is as frequency domain shows the “details” of an image by letting only certain types of frequencies to pass. [6]

EULER ANGLES:

One of the main reasons quaternions are used is due to their great advantage when it comes to describing rotations. The reason they are good with that is due to their ability to describe Orientation. [3]

To understand what orientation is, imagine a vector in 3D space as shown in **Figure 10**. Imagine that the vector is twisting around itself. Since vectors have a direction and a magnitude, but no orientation. [3], [8]

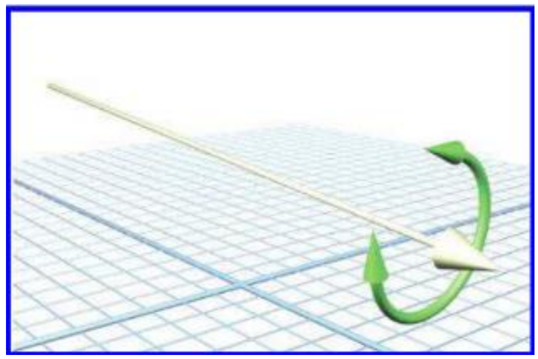

Figure 10 shows a vector in 3D space showing that vectors have no orientation

The vector is going to stay the same. On the other hand, imagine that instead of this vector we have a jet as shown in **Figure 11**. If the jet is twisting its orientation is going to change, unlike the vector.

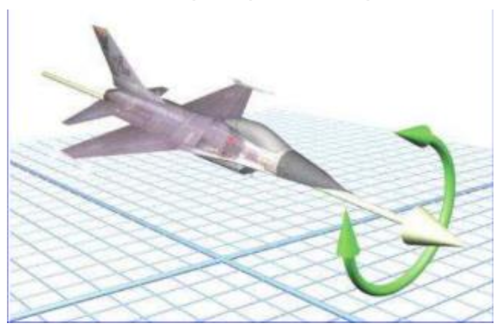

Figure 11 shows a jet in 3D space to show its orientation

This gives us an idea of what orientation is, we can think of it as how the rigid body of the jet exists on the 3D space as a result of rotation.

3D rotation can be expressed in different ways. The most known ways are Matrices, Euler angles and Quaternions. We are going to discuss each of them, and the differences between each way, with its advantages and disadvantages. [3]

The most common method of representing rotation in 3D space is called “Euler Angles”, it is named after the mathematician who invented them called, Leonhard Euler. Euler angles are composed of three torus circles that are aligned on top of each other on the x, y, and z axes **as shown in Figure 12.** [3], [8], [10]

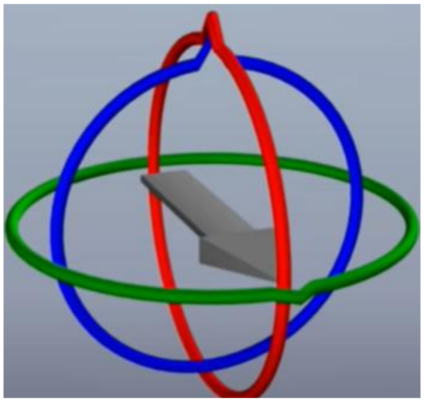

Figure 12 shows x, y, and z axes that make up Euler angles

Euler angles use three numbers to describe an orientation. These three numbers are all degrees, which is usually how people think of representing orientation. They are also easy to be stored in small quantities of data. That is why they are widespread when handling animations and three-dimensional rotations.

There is no such thing as an invalid set of angles in Euler angles. This is as if any set of angles is valid and could be used. Euler angles are used widely as they are really easy to implement and use. However, they have a problem that results when two circles are aligned on the same axis, **as shown in Figure 13**. This problem is called Gimbal lock, and it prevents the torus circles from moving as they are locked together. This problem is very popular in game development and computer graphics fields. They have another problem: you can represent the same orientation with a few different sets of angles. [3], [8], [11]

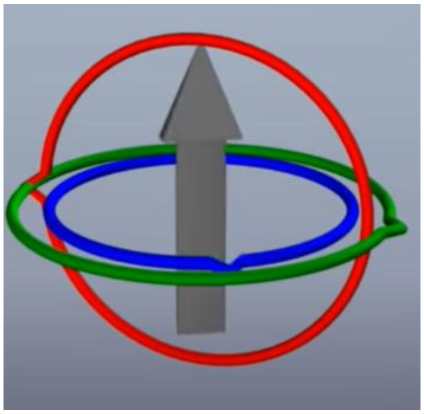

Figure 13 shows Gimbal lock of Euler angles

ROTATION VIA QUATERNIONS:

Quaternions are most commonly used for representing 3D rotation, since they were invented by William Hamilton. The reason quaternions are used so much, especially in game development and modern

graphics is a process called SLERP. SLERP is short for Spherical Linear Interpolation. This process lets us smoothly interpolate between two orientations using quaternions. It also avoids all the problems and issues that plagued Euler angles.

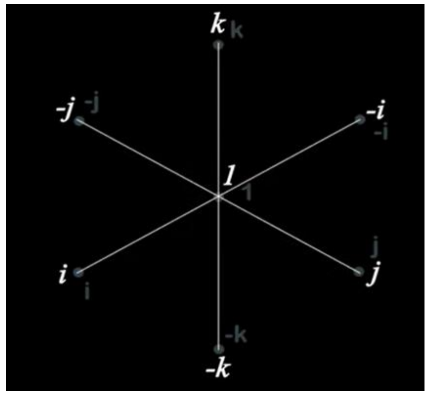

Figure 14 shows quaternion axis to describe rotation

Rotation can be done with the definition of quaternions by doing multiplication. For example, on the quaternion space shown in **Figure 14**, multiplying by $-i$ will rotate the points of a sphere (which is projected as a 2D circle on the axis) to the right. This is where $j(-i)$ will be mapped to k and the point at 1 is mapped to $-i$. For rotation to the left, the points need to be multiplied by i. [1], [3], [6]

The concept of multiplying by i could be used to describe rotation on the i axis. This can be done by multiplying a unit vector v by $i\ or - i$ which describes the direction. When using $-i(v)i$ this creates double rotation in one direction on the i axis as the rotation on the other axis will cancel out. This is **as shown in Figure 15**. [3], [6]

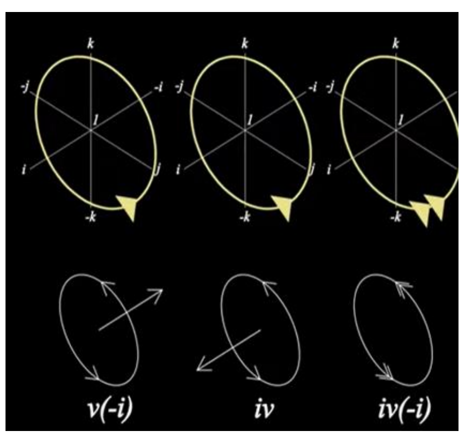

Figure 15 shows quaternion rotation by multiplication

For applying that in a 3D space, there it has to be in the form $qv\bar{q}$. This is where v is a pure quaternion, q has the angle of rotation and axis, and $\bar{q}$ is its conjugate. [3], [6]

QUATERNION INTERPOLATION:

SLERP is a ternary operation. This means that it needs three parameters to work, the first two are the orientations and the third is the interpolation parameter. The two orientations are referenced to as q_0 and q_1 variables. The third operator is t with a value from 0 to 1. At the end, the function looks like this: $slerp(q_0, q_1, t)$. For interpolation between two scalar values a_0 and a_1. The standard linear interpolation (lerp) could be used. Its formula is as shown in **Figure 16**. [3], [8]

$$\Delta a_1 - a_0$$
$$lerp(a_0, a_1, t) = \ a_0 + t\Delta \mathrm{a}$$

Figure 16 shows function that represents interpolation between two points

This formula starts at a_0 and adds the fraction t of the difference between a_0 and a_1. This requires 3 basic steps which are computing the difference between the two values, taking a fraction of this difference and adjusting the original value by this fraction of difference.
The difference between the two values is given by getting the angular displacement from q_0 and q_1 which is given by

$$\Delta q = q_1 q_0^{-1}$$

The second step which is taking a fraction of this difference, for doing that, quaternion exponentiation is used. The fraction of the difference is given by

$$(\Delta q)^t.$$

The third step is taking the original value and adjusting it by the fraction of the difference. The initial value is adjusted by composing angular displacements via quaternion multiplication:

$$(\Delta q)^t q_0$$

Thus, the equation of slerp is given as shown in **Figure 17.**

$$\mathrm{slerp}(\mathbf{q}_0, \mathbf{q}_1, t) = (\mathbf{q}_1 {\mathbf{q}_0}^{-1})^t \mathbf{q}_0.$$

Figure 17 shows the equation of the slerp function

Interpolation could be visualized as the movement between two points in a 4D hypersphere creating an arc. For simplification purposes, this can be shown in a 2D plane, where the two vectors v_0 and v_1 are both of unit length. To find v_t, the result of the interpolation between the two vectors by a fraction t of the distance from v_0 to v_1. At this point, let w be the angle intercepted by the arc from v_0 to v_1. Therefore, v_t is the result of rotating v_0 around the arc by an angle of tw. That is as shown in **Figure 18**. [3], [8], [10]

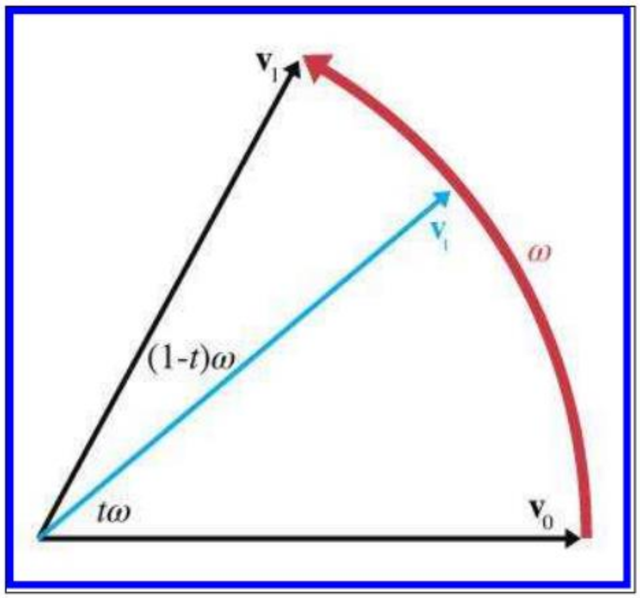

Figure 18 shows interpolation between two points in a 2D plane

IV. CONCLUSION

Quaternions have been a topic of debate for a while throughout history. They were, however, found to be very useful in various fields of computer science. They are used in areas such as Astronomy, Mathematics, Game development and Computer graphics. In astronomy, they were found very useful as image processing was very effective in preparing data before it was studied and analyzed. In game development, they were used instead of Euler angles as they effectively eliminate the problem of Gimbal Lock. In the end, Quaternions have had a lot of uses in modern computing, making them effective.

REFERENCES

[1] B. Chen, H. Shu, G. Coatrieux, G. Chen, X. Sun, and J. L. Coatrieux, "Color Image Analysis by Quaternion-Type Moments," J Math Imaging Vis, vol. 51, no. 1, pp. 124–144, Jan. 2015, doi: 10.1007/s10851- 014-0511-6.

[2] D. Misra, S. Mishra, and B. Appasani, "Advanced Image Processing for Astronomical Images," p.

[3] D. Fletcher, and P. Ian, “3D Math Primer for Graphics and Game Development”. CRC Press.

[4] J. Jenkinson et al., “Machine learning and image processing in astronomy with sparse data sets,” in 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), San Diego, CA, USA, Oct. 2014, pp. 200–203. doi: 10.1109/SMC.2014.6973907.

[5] A. M. Grigoryan and S. S. Agaian, “RETOOLING OF COLOR IMAGING IN THE QUATERNION ALGEBRA,” vol. 1, no. 3, p. 18, 2014

[6] A. M. Grigoryan, A. John, and S. S. Agaian, “Alpha-Rooting Color Image Enhancement Method by Two-Side 2-D Quaternion Discrete Fourier Transform Followed by Spatial Transformation,” IJACEEE, vol. 6, no. 1, p. 01–21, Feb. 2018, doi: 10.5121/ijaceee.2018.6101.

[7] R. Mukundan. “Quarterions: From classical mechanics to computer graphics, and beyond,” University of Canterbury, 2022.

[8] L. Martin, K. Martin, D. B. Erik, “Quaternions, Interoplation and Animation” University of Copenhagen, Jul. 1998

[9] J. Yan-Bin, “Quaternions” Sep. 2020

[10] K. Shoemake, “Animating rotation with quaternion curves”, Computer Graphics, vol. 19, no. 3. p. 245–254, 1985.

[11] P. J. Besl, N. D. McKay. “A method for registration of 3-D shapes”, vol. 14, no. 2, p. 239–256.

[12] A. M. Grigoryan, A. John, and S. S. Agaian, “Color image enhancement of medical images using alpha-rooting and zonal alpha-rooting methods on 2D QDFT,” Orlando, Florida, United States, Mar. 2017, p. 1013618. doi: 10.1117/12.2254889

[13] K. I. Kit, X. Yong-Hui, “Linear Quaternion Differential Equations: Basic Theory and Fundamental Results”, Zhejiang Normal University, Jinhua , Sep. 2017.