Dual-Quaternion Surfaces and Curves

Ben Kenwright

Abstract— Dual-quaternions offer an elegant and efficient possibility for representing parametric surfaces and curves due to their distinguishing properties. While quaternions are a popular concept for representing rotations, dual-quaternions offer a broader classification (composition of rotation and translation in a unified form). This paper presents a new approach using dual-quaternions for creating customizable parametric curves and surfaces. We explain the fundamental theory behind dual-quaternion algebra and how it is able to be harnessed to describe parametric geometry. The approach leverages popular mathematical concepts behind current parametric techniques. As we show, dual-quaternions are suitable for describing control points for parametric equations. We provide the mathematical details, in addition to experimental results to validate the approach.

Index Terms—surfaces, curves, interpolation, dual-quaternion, control points, design, graphics, geometry, graphics

1 Introduction

The parametric representation of curves and surfaces is an important technique used in many fields to model geometric forms. This is due to the fact, that a parametric equation is a compact and flexible form for representing geometry. While there are a variety of parametric methods for representing curves and surfaces in computing (e.g., B-spline and NURBS), one of the most popular and well known is the Bézier technique. The Bézier approach has been successful, as it offers a tool for interactive design for a broad range of applications, such as, in visualization [25], [5], design [12], [2], and even animation [24]. The Bézier's approach operates similar to other parametric techniques by employing a control net (which is intuitive and flexible for users). The control net is represented by a set of control points, typically, in Euclidean space $\mathbb{R}^3$ or $\mathbb{R}^2$. The Bézier approach makes it effortless to produce coordinated and controlled smooth curves. Control points provide a visually intuitive solution, and for many applications, are mathematically convenient.

The control points are typically represented by positions in Euclidean space. While orientation information is able to be combined with the control points (i.e., rotation and translation), they are calculated and interpolated in parallel (usually orientation in quaternion form due to the interpolation properties). However, both translational and orientational information are independent of one another (kept in separate silos) with no spatial kinematic connection or relationship. Existing research on curve and surface techniques have focused predominately on the interpolation aspects with several studies exploring new spline control techniques [18], [11], [1], [3], [7], [21], [20], [8]. Non of these studies have addressed how to unify the calculation of rotational and positional interpolation of curves and surfaces (based upon existing spline control concepts). While Hermite interpolation in combination with dual-quaternions has been suggested by researchers [22], [13], the focus was on efficient spline motions (i.e., compared to the geometric representation and visualization as discussed in this paper). As we discuss, dual-quaternions have gained considerable attention in the field of robot control and computer animation [16], [19], [17], [15], due to their mathematical properties, which we believe offer potential benefits to parametric curves and surfaces. As far as we know, this paper is the first to consider the geometric representation of curves and surfaces with dual-quaternions (coupled holistic connection between the position and orientation control points). While several properties of Bézier curves are inherited, new features are introduced in this paper through our dual-quaternion approach.

As we show, dual-quaternions are a powerful mathematical form, that can be used to construct free-form curves and surfaces. As Bézier curves and surfaces have become fundamental tools in many challenging and varied applications, ranging from computer-aided geometric design to generic object shape descriptors. The shape is defined by the control data and generated iteratively using the appropriate interpolation algorithm. Interpolation data is typically 'positions' (i.e., surface points which make up the shape mesh). While orientation data is able to be added, this is normally kept separate (detached and independent). Our approach, 'combines' the interpolation data as 'dual-quaternions' - which brings together both orientation and translation information (unified holistic form). The surface control points are represented by dual-quaternions. Our approach extends Bézier surfaces and curves, while keeping the same fundamental underpinnings that have made it so popular (i.e., easy to compute and very stable). What is more, our approach continues to produce very smooth curves with continuity which makes it suitable for design purposes.

Contribution The key contribution of this paper, is the explanation and demonstration of dual-quaternions for creating parametric curves and surfaces. The paper leverages existing techniques in combination with the mathematics of dual-

- *Ben Kenwright*
  bkenwright@ieee.org

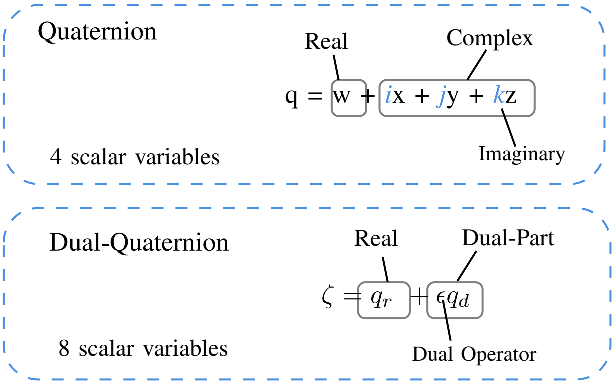


Fig. 1: Visual Overview of Quaternion and Dual-Quaternion Components.

quaternions to create a solution that is both usable and efficient. Our approach, builds upon the strengths of the Bézier curve and surface technique to create a new polynomial class with adjustable shape parameters. While dual-quaternions have been well studied in computer animation for the purpose of optimal blending of rigid transformations SE(3) in character skinning, this paper proposes **utilizing dual-quaternion properties for optimal interpolation of rotational and translational control points** for the geometric design of curves and surfaces.

2 MATHEMATICS

This paper outlines an alternative modification to the Bézier curve and surface method, with an aim to retain the underlying properties. At the heart of our approach is dual-quaternions. To ensure this paper provides sufficient detail on the topic, we provide a compact and concise explanation of the core mathematical concepts behind dual-quaternions and how they fit into our proposed technique for generating surfaces and curves. Dual-quaternions, although not as well known as Quaternions, provide a fundamental and solid base for describing three-dimensional transforms (orientation and translation) of an object or a vector. They are efficient and well suited for solving a variety of problems in computer graphics and animation [15].

2.1 Algebraic Definitions

Quaternion Operations The quaternion was discovered by Hamilton in 1843 as a method of performing 3-D multiplication [16]. A quaternion q is given by Equation 1. Since we are combining quaternions with dual number theory, we give the elementary quaternion arithmetic operations (also see Figure 1).

$$q = [s, \vec{v}], (q_w, q_x, q_y, q_z) \tag{1}$$

where s scalar part is $s = q_w$ and vector part is $\vec{v} = (q_x, q_y, q_z)$. The four-tuple of independent real values assigned to one real axis and three orthonormal imaginary axes: i, j, k.

- **addition**: $q_1 + q_2 = [s_1, \vec{v}_1] + [s_2, \vec{v}_2] = [s_1 + s_2, \vec{v}_1 + \vec{v}_2]$
- **additive identity**: $0 = [0, 0]$
- **scalar multiplication**: $kq = [ks, k\vec{v}]$
- **multiplication**: $q_1 q_2 = [s_1, \vec{v}_1][s_2, \vec{v}_2] = [s_1 s_2 - \vec{v}_1 \cdot \vec{v}_2, s_1 \vec{v}_2 + s_2 \vec{v}_1 + \vec{v}_1 \times \vec{v}_2]$
- **multiplication identity**: $1 = [1, 0]$
- **dot product**: $q_1 \cdot q_2 = (q_{1x} q_{2x} + q_{1y} q_{2y} + q_{1z} q_{2z} + q_{1w} q_{2w})$
- **magnitude**: $||q|| = \sqrt(s^2 + ||\vec{v}||^2)$
- **conjugate** $q^* = [s, -\vec{v}]$

Dual-Quaternion Operations The elementary arithmetic operations necessary for us to use dual-quaternions.

- **dual-quaternion**: $\zeta = q_r + q_d \varepsilon$
- **scalar multiplication**: $s\zeta = sq_r + sq_d \varepsilon$
- **addition**: $\zeta_1 + \zeta_2 = q_{r1} + q_{r2} + (q_{d1} + q_{d2})\varepsilon$
- **multiplication**: $\zeta_1 \zeta_2 = q_{r1} q_{r2} + (q_{r1} q_{d2} + q_{d1} q_{r2})\varepsilon$
- **conjugate**: $\zeta^* = q_r^* + q_d^* \varepsilon$
- **magnitude**: $||\zeta|| = \zeta \zeta_*$

where q_r and q_d indicate the real and dual part of a dual-quaternion.

For a beginners introduction to dual-quaternions and a comparison of alternative methods (e.g., matrices and Euler angles) and how to go about implementing a straightforward library we refer the reader to the paper by Kenwright [16]. Kavan et al [14] also contains an introduction to dual numbers and dual quaternions with discussions on screw parameterization.

2.2 Algebra in Context

Dual-Quaternion Vector Transformation A dual-quaternion is able to transform a 3D vector coordinate as shown in Equation 2. Note that for a unit dual-quaternion the inverse is the same as the conjugate.

$$p' = \hat{\zeta} p \hat{\zeta}^{-1} \tag{2}$$

where $\hat{\zeta}$ is a unit dual-quaternion representing the transform, $\hat{\zeta}^{-1}$ is the inverse of the unit dual-quaternion transform. p and p' are the dual-quaternions holding 3D vector coordinate to before and after the transformation (i.e., $p = (1, 0, 0, 0) + \epsilon(0, v_x, v_y, v_z)$)).

Plücker Coordinates Plücker coordinates are used to create Screw coordinates which are an essential technique of representing lines. We need the Screw coordinates so that we can re-write dual-quaternions in a more elegant form to aid us in **formulating a neater and less complex interpolation method** that is comparable with spherical linear interpolation for classical quaternions.

The Definition of Plučker Coordinates:

- $\vec{p}$ is a point anywhere on a given line
- $\vec{l}$ is the direction vector
- $\vec{m} = \vec{p} \times \vec{l}$ is the moment vector
- $(\vec{l}, \vec{m})$ are the six Plučker coordinate

We can convert the eight dual-quaternion parameters to an equivalent set of eight screw coordinates and vice-versa. The definition of the parameters are given below in Equation 3:

$$\begin{aligned}\text{screw parameters} &= (\theta, d, \vec{l}, \vec{m}) \\ \text{dual-quaternion} &= q_r + \epsilon q_d \\ &= (w_r + \vec{v}_r) + \epsilon(w_d + \vec{v}_d)\end{aligned} \tag{3}$$

where in addition to $\vec{l}$ representing the vector line direction and $\vec{m}$ the line moment, we also have d representing the translation along the axis (i.e., pitch) and the angle of rotation θ.

Convert dual-quaternion to screw-parameters

$$\begin{aligned}\theta &= 2cos^{-1}(w_r) \\ d &= -2w_d \frac{1}{\sqrt{(\vec{v}_r \dot{\vec{v}}_r)}} \\ \vec{l} &= \vec{v}_r \left(\frac{1}{\sqrt{(\vec{v}_r \dot{\vec{v}}_r)}} \right) \\ \vec{m} &= \left(\vec{v}_d - \vec{l}\frac{dw_r}{2} \right) \frac{1}{\sqrt{(\vec{v}_r \dot{\vec{v}}_r)}}\end{aligned} \tag{4}$$

Convert screw-parameters to dual-quaternion

$$\begin{aligned}w_r &= cos\left(\frac{\theta}{2}\right) \\ \vec{v}_r &= \vec{l} sin\left(\frac{\theta}{2}\right) \\ w_d &= -\frac{d}{2} sin\left(\frac{\theta}{2}\right) \\ \vec{v}_d &= sin\left(\frac{\theta}{2}\right)\vec{m} + \frac{d}{2}cos\left(\frac{\theta}{2}\right)\vec{l}\end{aligned} \tag{5}$$

Dual-Quaternion Power We can write the dual-quaternion representation in the form given in Equation 6.

$$\begin{aligned}\hat{\zeta} &= cos\left(\frac{\theta + \epsilon d}{2}\right) + (\vec{l} + \epsilon \vec{m}) sin\left(\frac{\theta + \epsilon d}{2}\right) \\ &= cos\left(\frac{\hat{\theta}}{2}\right) + \hat{v} sin\left(\frac{\hat{\theta}}{2}\right)\end{aligned} \tag{6}$$

where $\hat{\zeta}$ is a unit dual-quaternion, $\hat{v}$ is a unit dual-vector ($\hat{v} = \vec{l} + \epsilon \vec{m}$), and $\hat{\theta}$ is a dual-angle ($\hat{\theta} = \theta + \epsilon d$).

The dual-quaternion in this form is exceptionally interesting and valuable as it allows us to calculate a dual-quaternion to a power. Calculating a dual-quaternion to a power is essential for us to be able to easily calculate spherical linear interpolation. However, **instead of purely rotation as with classical quaternions, we are instead now able to interpolate full 6-dimensional degrees of freedom** (i.e., rotation and translation) by using dual-quaternions.

$$\hat{\zeta}^t = cos\left(t\frac{\hat{\theta}}{2}\right) + \hat{v} sin\left(t\frac{\hat{\theta}}{2}\right) \tag{7}$$

Dual-Quaternion Screw Linear Interpolation (ScLERP) ScLERP is an extension of the quaternion SLERP technique, and allows us to create constant smooth interpolation between dual-quaternions. Similar to quaternion SLERP, we use the power function to calculate the interpolation values for ScLERP shown in Equation 8.

$$\text{ScLERP}(\hat{\zeta}_A, \hat{\zeta}_B : t) = \hat{\zeta}_A(\hat{\zeta}_A^{-1}\hat{\zeta}_B)^t \tag{8}$$

where $\hat{\zeta}_A$ and $\hat{\zeta}_B$ are the start and end unit dual-quaternion and t is the interpolation amount from 0.0 to 1.0.

Alternatively, a **fast approximate alternative to ScLERP** was presented by Kavan et al. [14] called Dual-Quaternion Linear Blending (DLB). Furthermore, dual-quaternions have gained a great deal of attention in the area of character-based skinning. Since, a skinned surface approximation using a weighted dual-quaternion approach produces less kinking and reduced visual anomalies compared to linear methods by ensuring the surface keeps its volume.

3 METHOD

The mathematical method for curves can be extended to surfaces. The most important element in this paper, for the creation of the curves and surfaces is the use of dual-quaternions (control points) and the interpolation technique. As with other approaches, the shape is defined by a set of points and the surface is created by interpolating between points. The approach is flexible in its nature (e.g., different dual-quaternion interpolation methods). The basic steps we use in this paper are:

- Define the dual-quaternion control points
- Set the interpolation method (LERP, DLB or ScLERP)
- Iteratively construct the curve (or surface)

The control points define the geometric shape. We apply an uncomplicated De-Casteljau's algorithm to calculate the interpolation points (since the algorithm is computationally more suited and slightly more numerically stable than other approaches as reported in the literature [10]). We tested both a DQ-SCLERP and DQ-LERP interpolation method. While our approach is applicable to other parametric curve (or surface) models, we focus on Bézier approach, as this is a popular choice that is numerically more stable than other forms [6].

Converting between Euclidean coordinates and Euler/Quaternion Angles to a dual-quaternion format (see the Mathematical section on converting to and from the different coordinate systems).

3.1 De-Casteljau's Algorithm for Interpolation Points

De-Casteljau's algorithm is a recursive method for evaluating the Bézier curve (or surface) at any point between the control points. The method allows the properties of Bézier curve (or surface) to be derived swiftly and efficiently without

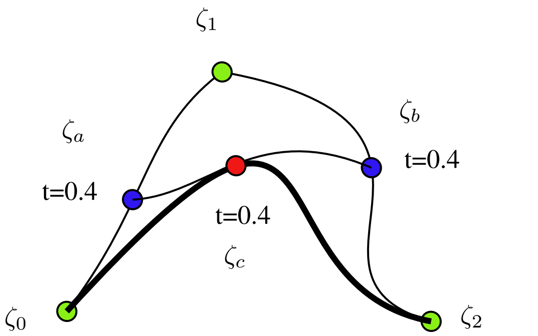


Fig. 2: De-Casteljau's Algorithm for a Curve (using dual-quaternions).

any reference to the Bernstein polynomials and essentially with only geometric argument (i.e., dual-quaternion control points). De-Casteljau algorithm is numerically more stable way (compared to using the parametric form directly) of evaluating the position of a point on the curve for any given interpolation value (only requiring a series interpolations). The geometric interpretation of De-Casteljau's algorithm is straightforward, and allows us to subdivide each segment between dual-quaternion control points.

For example, the interpolation process for a curve (see Figure 2):

$$\begin{aligned} \zeta_a &= INTERP(t, \zeta_0, \zeta_1) \\ \zeta_b &= INTERP(t, \zeta_1, \zeta_2) \\ \zeta_c &= INTERP(t, \zeta_a, \zeta_b) \end{aligned} \tag{9}$$

where $INTERP$ is the interpolation calculation, discussed in the mathematics section (e.g., DLB or ScLERP) and ζ are the dual-quaternion control points.

For a surface, instead of a curve being parameterized by a single variable t, we use two variables s and t (with a range from 0 to 1). De-Casteljau's algorithm can be extended to handle Bézier surfaces. That is, the De-Casteljau's algorithm can be applied several times to find the corresponding point on a Bézier surface $p(s,t)$ given (s,t).

4 Experimental Results

We tested our approach using a number of simple test cases (e.g., 2D plots and 3D surfaces). The results show that our approach is able to successfully generate curves (and surfaces) using dual-quaterions for the control points (see Figure 3 and 5). For curves, 3 to 4 control points provides sufficient control and detail (however, due to the nature of the algorithm, this is flexible and could easily be increased).

While the traditional Bézier approach uses control points defined by vector positions and is contained within the convex hull (made up of the control points), this is not so for the dual-quaternion model. That is, dual-quaternion control points are able to deviate outside of this convex hull (worth noting as this could be a disadvantage if the technique requires a pre-defined containment geometry).

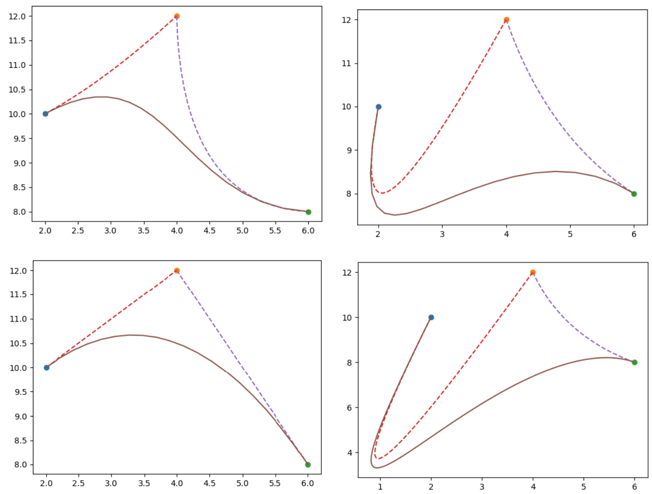


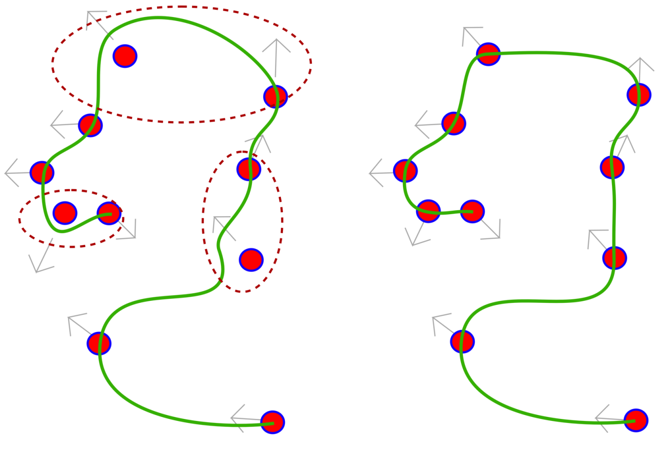
Fig. 3: **Dual-Quaternion Control Points (Curves)** - Three control points with a plot of the intermediate points for the surface. While the example illustrates a quadratic Bézier (i.e., 3 points), a cubic Bézier curve with 4 points (or more) is possible. Dotted line shows the interpolated path between the control points. (bottom-left) orientation for all the control points is the same.

4.1 Example Application

An example applications that shows the potential benefits of dual-quaternions for numerical optimization. Smooth interpolation between transforms (position and rotation/direction) **Our method allows us to create an 'overshoot' effect with no snapping while producing a a smooth curve.** Due to the coupled nature of the transforms between way-points, it allows us to have way-points influence the splines in many ways (pass through points and directions) - stretch and deform the curve while keeping the underlying properties.

For instance, the proposed technique can utilize both manual design methods or automated tuning (fitting functions). Tuning methods such as those presented by Tanaka et al. [23] and Gálvez [9] use automated algorithms to reconstruct non-uniform rational B-spline surfaces of a certain order from a given set data.

Given a specific set of sample points (training data) and constraints (function/predefined surface). The dual-quaternion approach allows the solution greater flexibility compared to keeping the 'positional' and 'orientational' data separate. This is due to the 'coupled nature' of our dual-quaternion approach (not confined to the bounding convex of the positional data). To achieve this solution using a conventional Bézier solution would require adding additional data points to the sample data. This sort of situation is common in dynamic surfaces which may bulge and skew surface for different conditions (as shown in Figure 4).



(a) (b)

Fig. 4: **Comparison** - Given the same number of points the unified method allows the creation of splines or surfaces that deviate from the constrained (positional) way-points while also representing orientationl details to meet the fitness (tuning) criteria. (a) Shows the desired path given the positional points and orientations using the unified dual-quaternion spline. (b) Separate position and orientation spline is unable to match the path without adding additional position way-points.

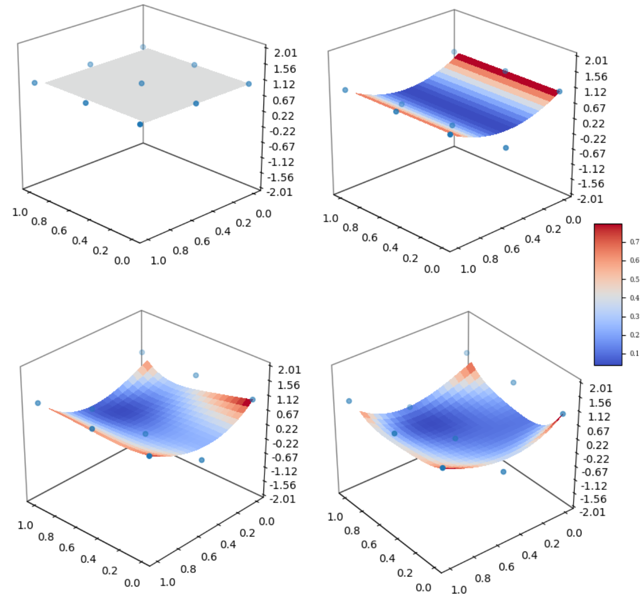


Fig. 5: **Dual-Quaternion Control Points (Surface - 3x3 Control Net)** - (Top left) flat surface with identical orientations), (Top right) control point positions are moved around (orientations kept the same), (Bottom Left & Right) show the control point positions are kept the same while the rotational information is modified.

Spatial displacement in 3-dimensional space of the X-Y-Z position and Yaw-Pitch-Role angles independently can be *counter-intuitive*. While quaternions are a convenient notation for capture the orientation from R^3 to R^4 (angular displacement and address limitations such as Gimbal lock). Extending the advantageous benefits of quaternion algebra for representing only rotational displacement operations to dual-quaternions allows us to **capture the positions and orientation** (R^3 to R^8).

4.2 Limitations

One limitation that our approach suffers from, is the fact that the control points are dual-quaternions, which makes it difficult for people to visualize. Hence, it can make it hard for people to get a grip on the shape of the curve while designing (and adjusting the control points). As with the standard Bézier approach, it does not allow local control, so moving one control point affects the whole curve (or surface), mostly around the point that is moved.

4.3 When to use Dual-Quaternions to Create Curves and Surfaces?

There are a variety of strengths and weaknesses of using dual-quaternions for surfaces and curves. Dual-quaternions offer an optimal interpolation solution for rotational and translational control points in geometric design of curves/-surfaces. At the heart of the dual-quaternion form is the Bézier curve basis (with notable similarities), however, dual-quaternions possess extra rotational information. Current parametric representations consider positional and orientation components separately, and all the observations with different levels of precision are given the same weight. The dual-quaternion approach presents a unified positioning and orientational model based on the robust de-Casteljau's algorithm. The question often follows, if we want to design or implement a curve or surface when would we use the dual-quaternion approach. Do we want to interpolate splines that pass through data points, or do we want to design the curves and surfaces by manipulating bounding regions and orientations. This means the question concerning the curves and surfaces focuses on design, rather than representation. Importantly, the dual-quaternion approach has both interpolation and interactive manipulation characteristics. Since iterative and automated approaches may be favored over hand-based (visual) methods for generation. Subsequent modifications taking place without display control (for example the design might only specify limitations and fitness criteria). The system would then be required to search for an optimal solution given the specification (i.e., the user would then not care about the underlying algorithmic mechanics). Of course, software developers, do not often care much about the beauty of a geometric concept - instead the priority is numerical stability, complexity and performance considerations. In terms of performance, given modern hardware advancements, things are not always so clear cut (e.g., bespoke hardware structure), however, the dual-quaternion algorithm is well suited to parallel architectures while being numerically stable and robust.

5 General Discussion and Conclusion

This paper has introduced a method for creating parametric surfaces and curves using dual-quaterions for the control points. Classical Bézier algorithms are restricted to the representation of positional data, compared to our approach, which adds a new aspect. We explained the mathematical underpinning, while showing experimental results to demonstrate the validity of our proposed approach. Our dual-quaternion approach is able to be exploited further to produce a wide range of curves and surfaces. The link between our approach and the Bézier algorithm is important (both theoretically and from a practical aspect). Our approach offers useful modifications and insights to the popular Bézier technique with applications in graphical design and visualization (artistic opportunities). The de-Casteljau algorithm was combined with dual-quaternions, however, the current approach could also be generalized to other cases of B-spline (i.e., deBoor algorithm). Possible future avenues for exploration, would be the combination of our approach with other interactive shape modeling tools, such as those presented by Cui and Sourin [4], to investigate visualization capabilities.

References

[1] Brian Andrew Barsky. The Beta-Spline: A Local Representation Based on Shape Parameters and Fundamental Geometric Measures. 1982. 1

[2] Adesh Bhil, VB Manjunath, Ayub Ashwak, and Mallesh Jakanr. A design of a gear tooth profile using curve. *International Research Journal of Automotive Technology*, 1(6):163–169, 2018. 1

[3] KW Brodlie and Sohail Butt. Preserving convexity using piecewise cubic interpolation. *Computers & Graphics*, 15(1):15–23, 1991. 1

[4] Jian Cui and Alexei Sourin. Interactive shape modeling using leap motion controller. In *SIGGRAPH Asia 2017 Technical Briefs*, page 3. ACM, 2017. 6

[5] Eric A De Kemp. Visualization of complex geological structures using 3-d bezier construction tools. *Computers & Geosciences*, 25(5):581–597, 1999. 1

[6] Gerald Farin. A history of curves and surfaces. *Handbook of computer aided geometric design*, 1, 2002. 3

[7] Thomas A Foley. Local control of interval tension using weighted splines. *Computer Aided Geometric Design*, 3(4):281–294, 1986. 1

[8] Frederick N Fritsch and Ralph E Carlson. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246, 1980. 1

[9] Akemi Gálvez and Andrés Iglesias. Particle swarm optimization for non-uniform rational b-spline surface reconstruction from clouds of 3d data points. *Information Sciences*, 192:174–192, 2012. 4

[10] Shang Gao, Zaiyue Zhang, and Cungen Cao. Particle swarm algorithm for the shortest bézier curve. In *2009 International Workshop on Intelligent Systems and Applications*, pages 1–4. IEEE, 2009. 3

[11] Xi-An Han, Yichen Ma, and Xili Huang. A novel generalization of bézier curve and surface. *Journal of Computational and Applied Mathematics*, 217(1):180–193, 2008. 1

[12] Hans-Jürgen Hochfeld and Martin Ahlers. Role of bezier curves and surfaces in the volkswagen cad approach from 1967 to today. *Computer-Aided Design*, 22(9):598–607, 1990. 1

[13] Gašper Jaklič, Bert Jüttler, Marjeta Krajnc, Vito Vitrih, and Emil Žagar. Hermite interpolation by rational G^k motions of low degree. *Journal of Computational and Applied Mathematics*, 240:20–30, 2013. 1

[14] Ladislav Kavan, Steven Collins, Jiří Žára, and Carol O'Sullivan. Geometric skinning with approximate dual quaternion blending. *ACM Transactions on Graphics (TOG)*, 27(4):1–23, 2008. 2, 3

[15] Kenwright. Dual-quaternions and computer graphics. In *Dual Quaternions and Computer Graphics*. ISBN: 978-1979193962, 2017. 1, 2

[16] Ben Kenwright. A beginners guide to dual-quaternions: what they are, how they work, and how to use them for 3d character hierarchies. 2012. 1, 2

[17] B Kenwrigth. Inverse kinematics with dual-quaternions, exponential-maps, and joint limits. *International Journal on Advances in Intelligent Systems*, 6(1), 2013. 1

[18] István Kovács and Tamás Várady. P-curves and surfaces: Parametric design with global fullness control. *Computer-Aided Design*, 90:113–122, 2017. 1

[19] Su Jin Lim and Kwang Ho Shon. Dual quaternion functions and its applications. *Journal of Applied Mathematics*, 2013, 2013. 1

[20] Gregory M Nielson. CAGD's top ten: what to watch. *IEEE Computer Graphics and Automation*, 13(1):35–37, 1993. 1

[21] Larry I Schumaker. On shape preserving quadratic spline interpolation. *SIAM Journal on Numerical Analysis*, 20(4):854–864, 1983. 1

[22] JM Selig. Rational interpolation of rigid-body motions. In *Advances in the Theory of Control, Signals and Systems with Physical Modeling*, pages 213–224. Springer, 2010. 1

[23] Teruo Tanaka, Takahiro Katagiri, and Toshitsugu Yuba. d-spline based incremental parameter estimation in automatic performance tuning. In *International Workshop on Applied Parallel Computing*, pages 986–995. Springer, 2006. 4

[24] Hai Tao and Thomas S Huang. Bezier volume deformation model for facial animation and video tracking. In *International Workshop on Capture Techniques for Virtual Environments*, pages 242–253. Springer, 1998. 1

[25] Ferhat Tas and Selçuk Topal. Bezier curve modeling for neutrosophic data problem. *Neutrosophic Sets and Systems*, page 3, 2017. 1