Orientation Interpolation in Quaternion Space Using Spherical Biarcs *

Wenping Wang and Barry Joe
Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada T6G 2H1
Phone: 403-492-5757
e-mail address: wenping@cs.ualberta.ca and barry@cs.ualberta.ca

Abstract

We consider the problem of interpolating a smooth curve to a point sequence in the unit quaternion space $\mathcal{U}$. This problem has application to object orientation interpolation in computer animation, sweep surface generation in solid modeling[7, 6]. Since the unit quaternions form the unit sphere S^3 in E^4, a simple curve scheme using *spherical biarcs* is presented to solve this problem. The spherical biarc is a curve on a sphere consisting of two smoothly joining circular arcs. It is shown that for any two given points and two tangents specified at the two points on the unit sphere S^3, there always exist spherical biarcs interpolating these data and these biarcs are easy to construct. This leads to an algorithm for constructing a smooth and locally controllable circular arc spline curve to interpolate a sequence of unit quaternions in $\mathcal{U}$. We also discuss how to compute in-between quaternions efficiently on the resulting spline curve.

Key words: Quaternions, spherical biarcs, rational Bézier curves, orientation, animation.

1 Introduction

The applications of quaternions in computer graphics have been explored since 1985 [7, 8, 6]. Quaternions can be represented as points in E^4 (4-dimensional Euclidean space). The set of all unit quaternions, denoted by $\mathcal{U}$, can be identified with the unit sphere S^3 in E^4. It is well known that a rotation transformation in E^3 can be represented by two diametrically opposite quaternions on S^3; and two orientations of an object in E^3 differ by a rotation about a fixed axis. Therefore unit quaternions can be used to represent object orientations in E^3 with respect to a reference orientation. More properties of quaternions can be found in [1].

In computer animation the following orientation interpolation problem arises. To model a gradual change of object orientations in E^3 over a time interval, first the orientations of an object in a series of keyframes, called *keyframe orientations*, are specified and represented by a sequence of unit quaternions $\{X_i\}$, called *keyframe quaternions*. Then a curve that is C^1 continuous with respect to the time parameter in the unit quaternion space $\mathcal{U}$ is sought to interpolate $\{X_i\}$. Then sequence of unit quaternions on the interpolating curve, which are called *in-between* quaternions and are usually denser than $\{X_i\}$, are computed to represent orientations of the object between keyframes. The C^1 smoothness of the curve is necessary for the angular velocity to change smoothly because a kink on the interpolating curve means an abrupt change of rotation axis and a jump in the magnitude of the tangent vector means a sudden change of angular speed.

Orientation interpolation is also useful in solid modeling for generating sweep surfaces [6]. In this application the cross-section curve is subjected to an orientation interpolation and also to a position interpolation. A sweep surface is produced by connecting successive in-between frames of the cross-section curve.

Since the set $\mathcal{U}$ of unit quaternions is the sphere $S^3 \subset E^4$, the curve interpolation problem in $\mathcal{U}$ resolves itself into a spherical curve interpolation problem. There are several methods in the literature for solving this problem. Shoemake [7] proposes a spherical analogue of the cubic Bézier curve via the de Casteljau recursive construction, replacing the six linear interpolations by six *Slerps*. A *Slerp* is an interpolation along a great arc on the sphere. In [8] Shoemake gives an analogue of Boehm's quadrangle construction of cubic curves, called *Squad*. These two curves are known to have exponential parametric representations. In [6] spherical analogues of the cubic cardinal spline and the tensioned B-spline curve are used, where the curves are defined by subdivision procedures. A different approach

*This work was partially supported by a grant from the Natural Sciences and Engineering Research Council of Canada.

is taken in [3], where the Hermite cubic interpolant is used to interpolate two points and the end tangents on S^3, and then the interpolant is projected onto the sphere S^3 through the center of S^3, as in general the interpolant is not contained in S^3.

In this paper we will use spherical biarcs represented as piecewise rational quadratic Bézier curves to interpolate points on S^3. The spherical biarcs are then stitched together to design a G^1, i.e. unit tangent vector continuous, interpolating spline curve on S^3. The result is a locally controllable circular arc spline curve, or a rational quadratic spline curve on S^3. This curve can be made C^1 using a simple arclength parameterization or other reparameterization methods. Also we consider the computation of two types of in-between quaternions on the resulting spline curve: equally spaced in-between quaternions on the spline curve; and in-between quaternions corresponding to equally spaced values of the time parameter. The main contribution of this work is to propose an interpolating curve that has a simple parametric representation and allows efficient computation of in-between quaternions.

The organization of this paper is as follows. In Section 2 we introduce spherical biarcs on S^3. In Section 3 we give an algorithm for constructing a G^1 circular arc spline to interpolate a sequence of keyframe quaternions in $\mathcal{U}$, and then consider using reparameterization to make the curve C^1 with respect to the time parameter. Section 4 discusses the computation of in-between quaternions on the interpolating curve. Section 5 presents a brief comparison of our method with other methods of curve interpolation on sphere S^3, and some examples of applying spherical biarcs. Section 6 contains concluding remarks.

2 Spherical biarcs

Spherical biarcs are curves on a sphere which consist of two circular arcs joining with G^1 continuity. Given two distinct points X_0, X_1 on $S^3 \subset E^4$ and two tangent directions T_0, T_1 specified at X_0 and X_1 respectively, we define $D = \{X_0, T_0, X_1, T_1\}$ to be a data set for biarc interpolation. It will be shown in this section that for any given data D there exists a family of spherical biarc on S^3 interpolating D. We are mainly concerned with the construction of spherical biarcs instead of the existence theorem, which would require a lengthy proof [10], so we will only list some properties of spherical biarcs.

In Sections 2 and 3, we will use homogeneous coordinates $X = (x_1, x_2, x_3, x_4, x_5)$ to represent points in E^4. When $x_5 \neq 0$, the point represented by X, denoted by $[X]$, is a *finite point*. The corresponding affine coordinates of $[X]$ are given by $(x_1/x_5, x_2/x_5, x_3/x_5, x_4/x_5)$. Any finite point has the homogeneous representation $(x_1, x_2, x_3, x_4, 1)$, which will be called the *standard form*. When $x_5 = 0$, $[X]$ is a *point at infinity*. The straight line passing through two distinct points X, Y is denoted by XY. The line segment connecting X and Y is denoted by $\overline{XY}$. A direction in E^4 can be represented by a homogeneous representation T of a point at infinity. Note that T and $-T$ represent two opposite directions, although they stand for the same point at infinity. With homogeneous coordinates, the unit sphere S^3 can be expressed as $XAX^T = 0$, where $A = \mathrm{diag}[I_4, -1]$ with I_4 being the 4 by 4 identity matrix. A point Y is inside S^3 if $YAY^T < 0$; it is outside S^3 if $YAY^T > 0$.

We now consider how to represent a circular arc on S^3. A circular arc can be represented as a rational quadratic Bézier curve in homogeneous form [5]

$$P(u) = (1-u)^2 P_0 + 2wu(1-u)P_1 + u^2 P_2, \qquad (1)$$

$0 \le u \le 1$, where the control points P_0 and P_2 are in standard form. When P_1 is a finite point, it is also assumed to be in standard form. $P_0P_1P_2$ is called the *control polygon* of the curve. The scalar w is called the *weight* of the curve. When the arc $P(u)$ is on the sphere S^3: $XAX^T = 0$, from the familiar properties of Bézier curves, we have that $P_0 \in S^3$, $P_2 \in S^3$, and that the line segments $\overline{P_0P_1}$ and $\overline{P_1P_2}$ are tangent to S^3 at P_0 and P_2 respectively. Therefore $P_0AP_0^T = P_0AP_1^T = 0$ and $P_2AP_2^T = P_2AP_1^T = 0$, for P_1 is in the tangent hyperplanes $P_0AX^T = 0$ and $P_2AX^T = 0$ of S^3 at P_0 and P_2 respectively. Substituting $P(u)$ in $P(u)AP(u)^T = 0$, yields

$$2(1-u)^2u^2P_0AP_2^T + w^2[2u(1-u)]^2P_1AP_1^T = 0.$$

Since P_1 in not on S^3, $P_1AP_1^T \neq 0$. So

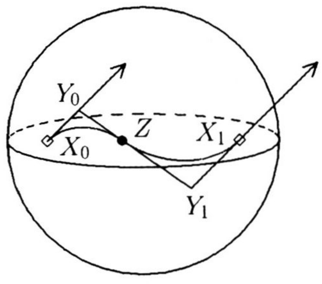
$$w^2 = -\frac{P_0AP_2^T}{2P_1AP_1^T}. \qquad (2)$$

Further, as P_1 is outside the sphere S^3, $P_1AP_1^T > 0$. Because P_0 and P_2 are in standard form and $P_0 \neq P_2$,

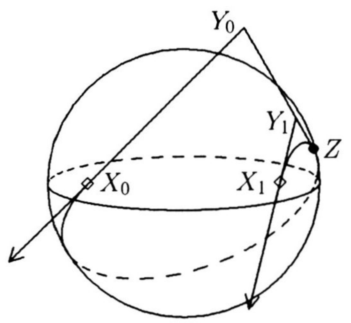
$$\begin{aligned} P_0AP_2^T &= -\frac{1}{2}(P_0 - P_2)A(P_0 - P_2)^T \\ &= -\frac{1}{2}(P_0 - P_2)(P_0 - P_2)^T < 0. \end{aligned}$$

So the right hand side of (2) is always positive. Thus two real values of w can be derived from (2). When P_1 is a finite point, the positive weight and negative weight correspond to, respectively, the minor arc and major arc of a circle on S^3 with control polygon $P_0P_1P_2$. When P_1 is a point at infinity, the two weights w solved from (2) correspond to two complementary semicircles of a circle with control polygon $P_0P_1P_2$ [5].

We now construct a spherical biarc B interpolating the given data $D = \{X_0, T_0, X_1, T_1\}$. Naturally we may assume that the tangent directions T_0 and T_1 are tangent to S^3. Therefore $X_0AX_0^T = X_0AT_0^T = 0$ and $X_1AX_1^T = X_1AT_1^T = 0$. Without loss of generality, we further assume that X_0 and X_1 are in standard form and $T_0AT_0^T = T_1AT_1^T = 1$. Let the biarc



(a)



(b)

Figure 2.1: The notation used in the spherical biarc interpolation problem. Note that a major arc is used in the biarc in (b).

B consist of two circular arcs C_0 and C_1 with control polygons X_0Y_0Z and ZY_1X_1 respectively, where the point Z, called the *joint* of B, is the point where the two arcs C_0 and C_1 join. The above notation is illustrated in Figure 2.1.

In seeking a biarc interpolant we need to distinguish types of biarcs and singular data.

Definition 2.1: *A biarc is* degenerate *if one of its arcs reduces to a single point. A biarc that is not degenerate is called* proper.

In practice, only proper biarcs are useful because in a degenerate biarc the tangent direction at one endpoint of the biarc is not defined, so it is meaningless to talk about interpolating end tangent directions.

Definition 2.2: *Data $D = \{X_0, T_0, X_1, T_1\}$ is called* singular *if $X_0 + \rho T_0 = X_1 + \rho T_1$ for some ρ or if $T_0 = T_1$. Data that is not singular is called* regular.

By definition regular data are generic. It will be seen that singular data and regular data have essentially different biarc solutions. We will first establish the properties of proper spherical biarcs interpolating regular data. Specifically, we have to find the unknowns Y_0, Y_1 and Z so that the arcs C_0 and C_1 joining at Z form a proper biarc. Let

$$\begin{aligned} Y_0 &= X_0 + k_0 T_0, \\ Y_1 &= X_1 - k_1 T_1, \end{aligned} \tag{3}$$

where k_0, k_1 are parameters to be determined. We assume that $k_0 k_1 \neq 0$. If $k_0 = 0$ or $k_1 = 0$, one of the triangles $\Delta X_0 Y_0 Z$ and $\Delta Z Y_1 X_1$ degenerates into a point, so the corresponding biarc would be degenerate.

The signs of k_0 and k_1 are also significant. When k_i is finite and $k_i > 0$, $i = 0, 1$, in order to interpolate the end tangent T_i, the weight of the arc C_i with control polygon ZY_iX_i must have the positive weight, i.e. the minor arc is used. When $k_i < 0$, $i = 0, 1$, C_i should have the negative weight, i.e. the major arc is used. When k_i is infinite, $i = 0, 1$, C_i is a semicircle since Y_i is a point at infinity.

For regular data we have the following existence theorem. The proof is given in [10].

Theorem 2.1: *For regular data D, a point Z on S^3 is the joint of a proper spherical biarc on S^3 interpolating regular data D iff Z is given by*

$$Z = k_1 Y_0 + k_0 Y_1, \tag{4}$$

where k_0 and k_1, with $k_0 k_1 \neq 0$, satisfy

$$X_0 A X_1^T + k_0 X_1 A T_0^T - k_1 X_0 A T_1^T + k_0 k_1 (1 - T_0 A T_1^T) = 0. \tag{5}$$

Moreover, Eqn. (5) has infinitely many solutions (k_0, k_1) with $k_0 k_1 \neq 0$.

More properties of spherical biarc interpolating regular data are listed below. Their proofs are given in [10].

Property 1: *For any nonzero solutions (k_0, k_1) of Eqn. (5), the control points Y_0, Y_1, and Z of the two arcs of the corresponding biarc are given, in standard form, by*

$$Y_0 = X_0 + k_0 T_0, \quad Y_1 = X_1 - k_1 T_1, \quad Z = \frac{k_0 Y_1 + k_1 Y_0}{k_0 + k_1},$$

where $k_0 + k_1$ never vanishes for regular data D. The weights w_0 and w_1 of the arcs C_0 and C_1 are determined from

$$w_0 = \frac{(-2 X_0 A Z^T)^{1/2}}{2k_0}, \quad w_1 = \frac{(-2 X_1 A Z^T)^{1/2}}{2k_1}.$$

Property 2: *The locus of the joints of all spherical biarcs interpolating regular data D is a circle J on S^3 passing through X_0 and X_1. The parametric equation of the circle J in parameter k_0 is*

$$Z(k_0) = k_0^2[(T_0 A T_1^T - 1) X_1 + (X_1 A T_0^T)(T_0 - T_1)]$$

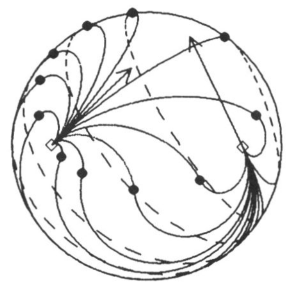


Figure 2.2: A family of proper spherical biarcs on S^2 interpolating the indicated regular data. Joints marked with • are on the same circle.

$$+ \quad k_0[(X_0AT_1^T)X_1 + X_0AX_1^T(T_0 - T_1) + (X_1AT_0^T)X_0] \\ + \quad (X_0AX_1^T)X_0. \qquad (6)$$

Here $Z(k_0)$ is not necessarily in standard form. To use proper biarcs only, the points X_0 and X_1 on J cannot be used as joints.

Because it is difficult to visualize curves in E^4, we use the unit sphere S^2 in E^3 to illustrate the above properties (see Figure 2.2). In Figure 2.2 there is a family of proper spherical biarcs on S^2 interpolating the indicated regular data, with all the joints lying on the same circle. If we intersect $S^3 \subset E^4$ with the 3-dimensional affine manifold in E^4 determined by the points X_0, X_1 and directions T_0, T_1, then the intersection would look like what is shown in Figure 2.2.

The following theorem is about the existence and construction of spherical biarcs interpolating singular data D on S^3.

Theorem 2.2: *Let D be singular data on S^3. Let $Y = X_0+\rho T_0$ if $X_0 + \rho T_0 = X_1 + \rho T_1$ for some ρ or let $Y = T_0$ if $T_0 = T_1$. Let J be the 2-dimensional sphere which is the intersection of S^3 with the polar hyperplane of Y with respect to S^3, i.e. $J = \{X | XAX^T = 0,\ YAX^T = 0\}$. Then a point Z is the joint of a proper spherical biarc interpolating D iff $Z \in J - \{X_0, X_1\}$. The control points Y_0 and Y_1 of all these biarcs are the same and given by $Y_0 = Y_1 = Y$.*

The above theorem also gives the control polygons of the spherical biarcs interpolating singular data D. The weights of the two arcs in a biarc can then be solved for from the control points through (2) with appropriate signs.

By Theorem 2.2, the family of spherical biarcs interpolating singular data has entirely different structure from that of regular data. The former has two degrees of freedom, whereas the latter has only one.

In order to design a short and smooth rotation path between keyframe orientations, it is desirable for the spherical biarc interpolant to have a small amount of winding and no sharp twist. Once the endpoints and end tangent directions are fixed, the shape of the biarc is determined by the joint. Through extensive experiments [10], we feel that a reasonable choice of the joint for achieving small winding and no sharp twist is to use an *equal chord biarc*. In an *equal chord biarc* the chords of its two arcs have equal length.

We consider first how to find an equal chord biarc interpolating regular data $D = \{X_0, T_0, X_1, T_1\}$. The joints of the equal chord biarcs are the points of intersection of the bisecting perpendicular hyperplane of the line segment $\overline{X_0X_1}$ with the circle J, the locus of the joints defined by (6). Obviously there are two such points of intersection. It can be verified easily that the bisecting perpendicular hyperplane of the line segment $\overline{X_0X_1}$ is

$$(X_0 - X_1)AX^T = 0.$$

By substituting the parametric equation (6) of the circle J in the above equation, the parameters (k_0, k_1) of the two points of intersection can be solved:

$$k_0^{(1)} = \frac{-X_0AX_1^T}{X_1AT_0^T - \sqrt{\Delta}}, \quad k_1^{(1)} = \frac{-X_0AX_1^T}{-X_0AT_1^T - \sqrt{\Delta}},$$

and

$$k_0^{(2)} = \frac{-X_0AX_1^T}{X_1AT_0^T + \sqrt{\Delta}}, \quad k_1^{(2)} = \frac{-X_0AX_1^T}{-X_0AT_1^T + \sqrt{\Delta}},$$

where

$$\Delta = (X_0AX_1^T)(T_0AT_1^T - 1) - (X_0AT_1^T)(X_1AT_0^T).$$

It is shown in [10] that $\Delta > 0$ for regular data, and that $(k_0^{(2)}, k_1^{(2)})$ is a preferable choice in the sense that it gives a biarc consisting of only minor arcs for more regular data than $(k_0^{(1)}, k_1^{(1)})$. In the following we will use the biarc given by

$$k_0 = k_0^{(2)}, \quad k_1 = k_1^{(2)}. \qquad (7)$$

When D is singular, by Theorem 2.2, the locus of the joints is the 2-dimensional sphere J, excluding X_0 and X_1. Let the bisecting perpendicular hyperplane of the line segment be $\overline{X_0X_1}$ L, defined by $L = \{X | (X_0 - X_1)AX^T = 0\}$. Then every point on the circle $J \bigcap L$ is a joint that gives an equal chord biarc interpolating the singular data D.

3 Interpolation of keyframe quaternions

In this section an algorithm is presented for interpolating a sequence of keyframe quaternions in the unit quaternion space

$\mathcal{U}$, or equivalently, a point sequence on S^3, using spherical biarcs. The algorithm is capable of preserving the shape of the data points, which can be regarded as the shape of the spherical polygon obtained by connecting consecutive data points X_i and X_{i+1} by the minor arc on the great circle of S^3 that passes through X_i and X_{i+1} (the arclength of this arc is the shortest spherical distance between X_i and X_{i+1} [7]).

To apply the spherical biarcs for interpolation, we first need to provide tangent directions at the data points. The following algorithm is essentially a scheme for defining these tangent directions.

Algorithm 3.1: *The input is a point sequence $\{X_i\}_{i=1}^n$ in standard form on S^3, $n \geq 3$. It is assumed that no two consecutive points are antipodal on S^3. Let C_i be the minor arc connecting X_i and X_{i+1} on the great circle of S^3 passing through X_i and X_{i+1}, with the associated direction being from X_i to X_{i+1}. Let T_i^1 be the tangent direction of C_{i-1} at X_i. Let T_i^2 be the tangent direction of C_i at X_i. Assume that $T_i^1 A (T_i^1)^T = T_i^2 A (T_i^2)^T = 1$. Let $d_i = \arccos(X_i X_{i+1}^T - 1)$, i.e. the arclength of C_i, $i = 1, 2, \ldots, n-1$.*

1. *The tangent direction T_i at X_i is defined to be*

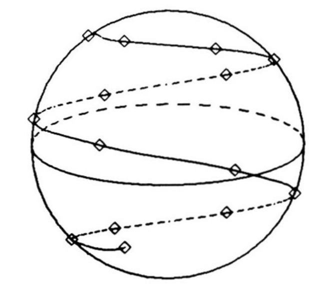
$$\begin{aligned} T_1 &\leftarrow T_1^2; \\ T_i &\leftarrow \frac{d_{i-1}T_i^1 + d_i T_i^2}{\|d_{i-1}T_i^1 + d_i T_i^2\|}, \quad i = 2, \ldots, n-1; \\ T_n &\leftarrow T_n^1. \end{aligned}$$

2. *Apply the spherical biarc to interpolate $D_i = \{X_i, T_i, X_{i+1}, T_{i+1}\}$, $i = 1, \ldots, n-1$. When D_i is regular, use the parameters k_0 and k_1 given by (7). When D_i is singular, Z can be chosen according to the last paragraph of Section 2.*

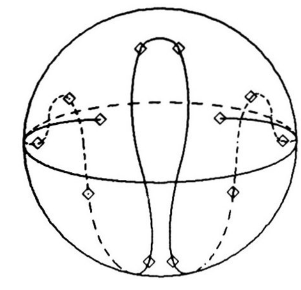
Note that at an internal point X_i the assigned tangent direction is a weighted average of T_i^1 and T_i^2. Figure 3.1 shows two examples of interpolating a point sequence by Algorithm 3.1. Again the point sequence is given on S^2 for easy visualization. Other schemes can be used to choose the tangent direction in step 1 of the above algorithm; the given scheme produces better interpolations than others we have tried so far.

The rational quadratic spline curves derived above are G^1 continuous. One way to obtain C^1 parameterizations of these curves is to use the arclength parameterization. This parameterization is easy to derive since every piece of the spline curve is a circular arc that is known to have a simple arclength parametric representation.

In some animation applications, different prescribed time instants have to be assigned to keyframes. In this case the arclength parameterization cannot be used, for in general arclengths of the curve segments between keyframe quaternions are not proportional to the corresponding time intervals between consecutive keyframes. Therefore a reparameterization is needed to make the interpolating curve C^1 continuous with respect to the time parameter.



(a)



(b)

Figure 3.1: Two circular arc spline curves constructed using Algorithm 3.1 interpolating data points on S^2.

We use a cubic Hermite polynomial to reparameterize each piece of the spline curve to make it C^1 continuous. This reparameterization method can correct any G^1 piecewise spline curve into a C^1 curve with respect to the time parameter. The idea is as follows. First, by linear reparameterization, all curve segments in a G^1 spline curve $P(u)$, which is assumed to be in nonhomogeneous coordinates, can be defined over consecutive time intervals. Let V be the meeting point of two arcs $P_0(u)$, $t_0 \leq u \leq t_1$, and $P_1(u)$, $t_1 \leq u \leq t_2$, of the curve $P(u)$, i.e. $V = P_0(t_1) = P_1(t_1)$. The tangent vector of $P(u)$ may be discontinuous at V since it can be that $\|P_0'(t_1)\| \neq \|P_1'(t_1)\|$. We just have to change the length of one of the tangent vectors to make the curve C^1 at V. For the simplicity of discussion, we assume that $0 < \|P_0'(t_1)\| < \|P_1'(t_1)\|$. Let $P_1(u(v))$, $t_1 \leq v \leq t_2$, be a reparameterization of $P_1(u)$, where $u(v)$ is a C^1 monotonically increasing function mapping $[t_1, t_2]$ onto itself, i.e. $u(t_1) = t_1$ and $u(t_2) = t_2$. If $u(v)$ is chosen in such a way that $u'(t_1) = \|P_0'(t_1)\| / \|P_1'(t_1)\| < 1$, then

$$\|\mathrm{d}P_1(u(v))/\mathrm{d}v\|_{v=t_1} = u'(t_1)\|P_1'(u(t_1))\|$$

$$= \quad ||P_0'(t_1)||.$$

That is, the curve $P(u)$ becomes C^1 at the point V if the new parameterization $P_1(u(v))$ is used instead of $P_1(u)$ over $[t_1, t_2]$.

The reparameterization required by each piece of the curve can be accomplished by a cubic Hermite polynomial. As a result, the interpolating spline curve becomes C^1 everywhere with respect to the time parameter, and it passes through key quaternions at the prescribed times. Note that the monotonicity of the above reparameterization function is crucial to the animation application. It can be shown that the cubic Hermite polynomial always gives a monotonically increasing reparameterization function due to the fact that the two end derivatives are chosen to be ≤ 1.

4 Computation of in-between quaternions

In this section we consider how to compute a sequence of in-between quaternions $\{Q_i\}$ on the circular arc spline curve interpolating $\{X_i\} \subset \mathcal{U}$, as constructed in Section 3. We will consider this problem in two cases: (1) The in-between quaternions are equally spaced points on the interpolating curve. (2) The in-between quaternions correspond to equally spaced values of the time parameter.

4.1 Equally spaced in-between quaternions

We start with the trigonometric parameterization of a circular arc, which is closely related to the arclength parameterization. For notational convenience, in this section we use affine coordinates to represent points in E^4. Let $\bar{C}$ be a circular arc in E^4 with its center at the origin. Let V_s and V_e be the two endpoints of the arc $\bar{C}$ in affine coordinates. Assume that $\bar{C}$ is not a semicircle. According to [7], $\bar{C}$ may be represented as

$$V(t) = \frac{\sin[(1-t)\psi]}{\sin\psi} V_s + \frac{\sin(t\psi)}{\sin\psi} V_e, \tag{8}$$

$0 \leq t \leq 1$, where ψ is the central angle subtended by $\bar{C}$, satisfying $0 < \psi < 2\pi$ and $\psi \neq \pi$.

Let $\{V_i\}$ be a point sequence on $\bar{C}$, with $V_i = V(i\Delta t)$, $\Delta t \neq 0$, $i = 0, 1, \ldots$, and let $\alpha = \psi \Delta t$. By (8),

$$V_i = \frac{\sin(\psi - i\alpha)}{\sin\psi} V_s + \frac{\sin(i\alpha)}{\sin\psi} V_e,$$

so V_i can be expressed as a linear combination of $\sin(i\alpha)$ and $\cos(i\alpha)$. Obviously $\{V_i\}$ is an equidistant point sequence. It is easy to see that $\sin(i\alpha)$ and $\cos(i\alpha)$ satisfy the difference equation

$$x_{i+2} - 2kx_{i+1} + x_i = 0, \quad i = 0, 1, \ldots,$$

where $k = \cos\alpha$. So $\{V_i\}$ also satisfies this equation, i.e.

$$V_{i+2} = 2kV_{i+1} - V_i, \quad i = 0, 1, \ldots. \tag{9}$$

Therefore, by the uniqueness of the solution to the initial value problem of the linear difference equation, we have shown that given the first two distinct points V_0 and V_1 on $\bar{C}$, the difference equation (9) generates an equidistant point sequence $\{V_i\}$ on $\bar{C}$. The distance between consecutive points of the sequence and the direction of the sequence are determined by V_0 and V_1.

We consider now applying Eqn. (9) to generate in-between quaternions $\{Q_i\}$. As a result of the spherical biarc interpolation, we may assume that a circular arc C is given as a rational quadratic Bézier curve in affine coordinates,

$$P(u) = \frac{(1-u)^2 P_0 + 2wu(1-u)P_1 + u^2 P_2}{(1-u)^2 + 2wu(1-u) + u^2}, \tag{10}$$

$0 \leq u \leq 1$. Here we assume that C is not a semicircle, i.e. P_1 is a finite point. In case that C is a semicircle, it can be subdivided into two quarters of a circle. Let K be the center of C. Then it can be shown [5] that K is given by

$$K = P_1 + \frac{1}{2(1-w^2)}(P_0 + P_2 - 2P_1).$$

Since $w = \cos(\psi/2)$, where ψ is the central angle subtended by C, $1 - w^2 \neq 0$ as long as $0 < \psi < 2\pi$. Let $\bar{C}$ denote the translation $U(u) = P(u) - K$ of the arc C. Then $\bar{C}$ has its center at the origin. So by the preceding discussion, $\bar{C}$ can be expressed as

$$V(t) = \frac{\sin[(1-t)\psi]}{\sin\psi}(P_0 - K) + \frac{\sin(t\psi)}{\sin\psi}(P_2 - K), \tag{11}$$

$0 \leq t \leq 1$, where $\psi = 2\arccos(w)$. The arclength of $\bar{C}$ is $\psi||P_0 - K||$.

The arclength increment of in-between quaternions can be determined from the total arclength of the circular arc spline curve and the number of in-between quaternions needed. Therefore the increment Δt on each arc can be computed. So we just need to compute the two initial points V_0 and V_1 to start the iterative computation on each arc using (9). When C is the first arc of the spline curve, $V_0 = V(0) = P_0 - K$ can be used. The second point V_1 is given by $V_1 = V(\Delta t)$. Then using (9), a point sequence $\{V_i\}$ on $\bar{C}$ is generated. Finally, the in-between quaternions $\{Q_i\}$ on $C: \; P(u)$ are obtained by $Q_i = V_i + K$, $i = 0, 1, \ldots$.

When C is not the first arc of the spline curve, due to the fact that in general the total arclength of the preceding arcs is not an integer multiple of the arclength increment, the two initial points must be computed differently to maintain a constant arclength increment between the first initial point on C and the last point of the preceding arc. To do this, the accumulated arclength up to the present in-between quaternion should be recorded, and this information can then be used to compute the first two initial points on C through (11). Then the iterative computation proceeds similarly.

The cost of the above method for computing in-between quaternions can be analyzed as follows. In the computation of (9), $k = \cos(\psi\Delta t)$ is a constant, and $2k$ can be precomputed. Obviously, 4 multiplications and 4 subtractions are required to compute a V_i. The subsequent translation $Q_i = V_i + K$ needs another 4 additions. So, besides the preprocessing, on average 4 multiplications and 8 additions/subtractions are needed to compute one in-between quaternion.

4.2 In-between quaternions with equal time intervals

It has been explained that through a reparameterization using a cubic Hermite polynomial $u(v)$ for each piece of the interpolating circular arc arc spline curve, the curve can be made C^1 with respect to the time parameter. To compute in-between quaternions with equal time intervals, we first compute values $\{u_i\}$ of the polynomial $u(v)$ at equally spaced time parameter values $\{v_i\}$, and then compute in-between quaternions $\{Q_i\}$ using the rational quadratic parametric representation of the spline curve at the parameter values $\{u_i\}$.

The values $\{u_i\}$ can be computed using forward differencing, using on average three additions per u_i. Then the computation of an in-between quaternion Q_i requires evaluating four rational quadratic polynomials sharing the same denominator, and thus using another 10 multiplications, 4 divisions and 10 additions per Q_i. Hence it needs 10 multiplications, 4 divisions, and 13 additions on average for each in-between quaternion.

5 Comparisons and experiments

It has been shown that spherical biarcs can be used to design a C^1 spline curve interpolating a sequence of points $\{X_i\}$ in the unit quaternion space. Consequently, the difference method for generating circular arcs based on the trigonometric parameterization can be used to efficiently compute equally spaced in-between quaternions $\{Q_i\}$ on the interpolating curve.

Now we compare all the methods we have mentioned so far. They are the spherical analogue of the cubic Bézier curve [7], the Squad curve [8], the spherical analogues of the cubic cardinal spline and tensioned B-spline [6], the normalized cubic Hermite interpolant [3], and the spherical biarc. In all these methods, except for the spherical analogues of the cardinal spline and tensioned B-spline, the tangent directions at the data points $\{X_i\}$ can be computed from a few data points. No tangent directions are specified for the cardinal spline and tensioned B-spline. All these methods lead to G^1 interpolation splines on S^3 (except the tensioned B-spline which is a G^2 approximation spline), allow one or two degrees of freedom in choosing the curve segment connecting two consecutive data points (this is usually the magnitude of the end tangent vectors, but is a scalar parameter in the case of the cardinal spline and tensioned B-spline), and provide local controllability (changing one data point affects up to 4 curve segments).

So far there has been no comparison of the quality of the above interpolation methods, because of the difficulty of visualization in E^4. Similarly, it is difficult to see how the spherical biarc compares with the spherical analogue of the cubic polynomial. The main criterion for comparing the various methods has been the efficiency of computing in-between quaternions. Since in their original presentations [7, 8, 6] only the computation of approximately equally spaced in-between quaternions is discussed, we will only compare these methods under the assumption that equally spaced or nearly equally spaced in-between quaternions are computed.

In the spherical analogue of the cubic Bézier curve an in-between quaternion is computed by evaluating six Slerps. According to [8], three of these six Slerps are *static*, the other three are *dynamic*, where in a static Slerp the two endpoints are fixed and in a dynamic Slerp the two endpoints vary; and 8 multiplications are needed to compute a static Slerp and 15 multiplications to compute a dynamic Slerp. So the total cost for computing one in-between quaternion in this scheme is about 69 multiplications and several table look-ups for trigonometric functions. The Squad curve requires two static Slerps and one dynamic Slerp to evaluate a point on the curve, thus reducing the cost of computing one in-between quaternion to about 31 multiplications and several table look-ups. Observe that by using the difference method discussed in Section 4, the computation of static Slerps in a Squad can be reduced to 4 multiplications per point. Therefore one in-between quaternion on a Squad can be computed using 23 multiplications and two table look-ups for the dynamic Slerp.

In the subdivision scheme for the spherical analogues of the cubic cardinal spline and tensioned B-spline, two *Smids* and a dynamic Slerp are needed to compute one in-between quaternion, where the Smid is a special form of the Slerp which can be computed using 8 multiplications/divisions and a square root. So the cost of computing one in-between quaternion in this scheme is at least 31 multiplications/divisions and 2 square roots, which is about the same as the Squad. For the normalized cubic Hermite interpolant, if forward differencing is used to generate points on the unnormalized cubic curve then an average of 12 additions are needed for each point, and the subsequent normalization step for each point requires a square root, 8 multiplications/divisions, and 3 additions.

The spherical biarc scheme proposed in this paper needs only an average of 4 multiplications and 8 additions/subtractions to compute each in-between quaternion. So if only C^1 continuity with respect to the time parameter is demanded the spherical biarc is an obvious choice in view of its efficiency. In addition, the spherical biarc is the only one of the above methods in which an equidistant sequence of in-between quaternions can be computed efficiently.

The above comparisons do not include the conversion time between rotation matrices and quaternions. Also we need to point out that in computer graphics applications, the time spent on other graphics operations, like transformation and shading of primitives in an object, is usually dominating in the total time cost, and almost always swamps the time saved in orientation interpolation by using spherical biarcs over other methods.

We have used the spherical biarc interpolation to create sweep surfaces and object orientation interpolation. Figure 5.1 and 5.2 show two sweep surfaces constructed from in-between frames as interpolations of some keyframes using spherical biarcs in the unit quaternion space. In Figure 5.1 there are 391 in-between frames interpolated from 30 keyframes obtained from a helix. In Figure 5.2 there are 42 in-between frames interpolated from 10 keyframes obtained from a cubic curve. In both examples, the cross-section curves are ellipses, and the keyframes are Frenet frames at the sampled points on each curve. Translations of the keyframes are interpolated by a cubic B-spline curve.

Figure 5.3 shows 13 in-between orientations of a cube interpolated from three keyframe orientations.

6 Concluding remarks

It is natural to ask whether there are other simple C^1 interpolating curves in the unit quaternion space. One open problem is how to use higher degree rational curves or other simple curve schemes for interpolation in the unit quaternion space while allowing efficient evaluation of in-between quaternions. One related work is the construction of rational curves of arbitrarily high degree on a sphere in E^3 in [4]. It would be interesting to generalize this approach to S^3.

Further research is needed on the shape of spherical biarcs. According to [2], a curve in the unit quaternion space with sharp twists corresponds to undesirable large *tangential acceleration* (see definition in [2]). So it would be useful to get a curve with a small average curvature over the whole curve. Specifically, the first problem is how the tangent directions at keyframe quaternions can be specified appropriately. The second problem is, once the tangents have been determined, how to choose the joints of spherical biarcs so that both arcs of each biarc have fairly large radii. Based on our experiments, we have chosen equal chord biarcs. But this choice is yet to be justified theoretically.

References

[1] Altmann, S.L. (1986), *Rotations, Quaternions, and Double Groups*, Clarendon Press, Oxford.

[2] Barr, A.H., Currin, B., Gabriel, S., and Hughes, J. F. (1992), Smooth Interpolation of Orientations with Angular Velocity Constraints Using Quaternions, *Computer Graphics*, vol. 26, no. 2, pp. 313-320.

[3] Ge, Q.J. and Ravani, B. (1991), Computational Geometry and Mechanical Design Synthesis, *13th IMAC World Congress on Computation and Applied Math.*, Dublin, Ireland, pp. 1013-1015.

[4] Hoschek, J. and Seeman, G. (1991), Spherical Splines, Preprint.

[5] Lee, E.T.Y. (1987), The Rational Bézier Representation for Conics, *Geometric Modeling: Algorithms and New Trends*, ed. G.Farin, SIAM, Philadelphia, pp. 3-19.

[6] Pletinckx, D. (1989), Quaternion Calculus as a Basic Tool in Computer Graphics, *The Visual Computer*, vol. 5, pp. 2-13.

[7] Shoemake, K. (1985), Animating Rotation with Quaternion Curves, *Proceedings of SIGGRAPH 85*, pp. 245-254.

[8] Shoemake, K. (1987), Quaternion Calculus and Fast Animation, *SIGGRAPH 87 Course Notes #10: Computer Animation: 3D Motion Specification and Control*, pp. 101-121.

[9] Wang, W. and Wang, C.Y. (1989), Difference Method for Generation of Circular Arcs and Ellipses, *Computer Aided Design*, vol. 21, no. 1, pp. 33-36.

[10] Wang, W. (1992), *Results on Conics and Quadrics in CAGD*, Ph.D. Thesis, Department of Computing Science, University of Alberta.

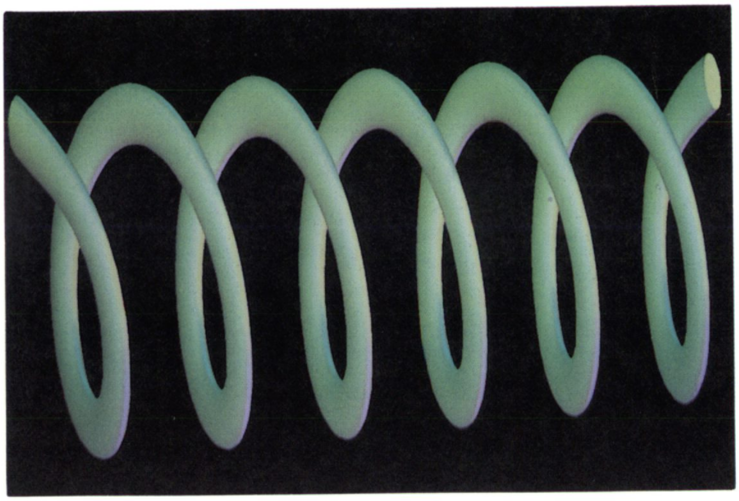


Figure 5.1: Sweep surface along a helix.

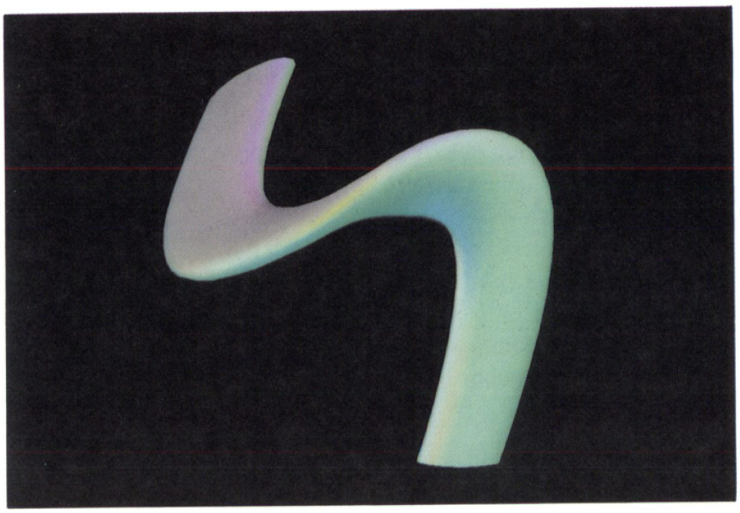


Figure 5.2: Sweep surface along a cubic curve.

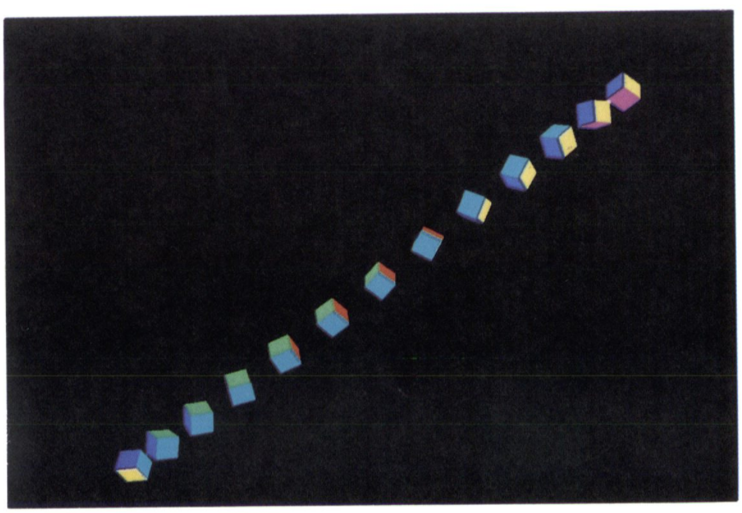


Figure 5.3: Orientations of a cube from 3 key orientations.