In *Proc. of SIGGRAPH'95*, pp. 369–376.

A General Construction Scheme for Unit Quaternion Curves with Simple High Order Derivatives[1]

Myoung-Jun Kim[2] Myung-Soo Kim[3] and Sung Yong Shin[2]
[2]Korea Advanced Institute of Science and Technology (KAIST)
[3]Pohang University of Science and Technology (POSTECH)

ABSTRACT

This paper proposes a new class of unit quaternion curves in $SO(3)$. A general method is developed that transforms a curve in R^3 (defined as a weighted sum of basis functions) into its unit quaternion analogue in $SO(3)$. Applying the method to well-known spline curves (such as Bézier, Hermite, and B-spline curves), we are able to construct various unit quaternion curves which share many important differential properties with their original curves.

Many of our naive common beliefs in geometry break down even in the simple non-Euclidean space S^3 or $SO(3)$. For example, the de Casteljau type construction of cubic B-spline quaternion curves does not preserve C^2-continuity [10]. Through the use of decomposition into simple primitive quaternion curves, our quaternion curves preserve most of the algebraic and differential properties of the original spline curves.

CR Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling – Curve, surface, solid, and object representation, – Geometric algorithms. **Keywords**: Quaternion, rotation, orientation, $SO(3)$, Bézier, Hermite, B-spline

1 INTRODUCTION

In computer animation, it is very important to have appropriate tools to produce smooth and natural motions of a rigid body. A rigid motion in R^3 can be represented by a Cartesian product of two component curves: one in the Euclidean space R^3 and the other in the rotation group $SO(3)$ [3, 12]. The curve in R^3 represents the center position of the rigid body, and the other curve in $SO(3)$ represents its orientation. Often, techniques for specifying a rigid motion construct the two curves independently. It is relatively easy to produce smooth curves in the Euclidean space. B-spline, Hermite, and Bézier curves exemplify well-known techniques for constructing position curves in R^3. However, smooth orientation curves in $SO(3)$ are much more difficult to construct.

The spline curves in R^3 are usually defined in two different but equivalent ways: i.e., either as an algebraic construction using basis functions or as a geometric construction based on recursive linear interpolations [2]. This paper proposes a general framework which extends the algebraic construction methods to $SO(3)$. Most of the previous methods are based on extending the linear interpolation in R^3 to the *slerp* (spherical linear interpolation) in $SO(3)$ [14, 16, 17, 18]. The two (i.e., algebraic and geometric) construction schemes generate identical curves in R^3; however, this is not the case in the non-Euclidean space $SO(3)$ [10]. Many of our commonplace beliefs in geometry break down in the non-Euclidean space $SO(3)$.

When the recursive curve construction is not based on a simple closed form algebraic equation, it becomes extremely difficult to do any extensive analysis on the constructed curve. For example, consider the problem of computing the first derivative of the cubic Bézier quaternion curve or the squad curve generated by a recursive slerp construction [17, 18]. Though Shoemake [17, 18] postulates correct first derivatives, the quaternion calculus employed there is incorrect (see [9] for more details). Kim, Kim, and Shin [9] developed a correct quaternion calculus for the first derivatives of unit quaternion curves; however, an extension to the second derivatives becomes much more complex (also see [11] for a related quaternion calculus on blending-type quaternion curves).

More seriously, the C^2-continuity of a spline curve in R^3 may not carry over to $SO(3)$. Furthermore, the curve conversion between two different spline curves, e.g., from a cubic Bézier curve to the corresponding cubic B-spline curve, may not work out in $SO(3)$ [10]. Kim, Kim, and Shin [10] show that there is no C^2-continuity guaranteed for the cubic B-spline quaternion curves which are generated by the recursive slerp construction of Schlag [16]. Similarly, the B-spline quaternion curve of Nielson and Heiland [13] also fails to have C^2-continuity (see [10] for more details).

In this paper, we take an important step toward generalizing the algebraic formulation of spline curves in R^3 to similar ones in $SO(3)$. In the new algebraic formulation, the computation of high order derivatives becomes almost as easy as that of the spline curves in R^3. We show that the quaternion curves generated in this way preserve many important differential properties of their original curves in R^3. They are defined in simple closed algebraic forms of quaternion equations, and they have the same degrees of geometric continuity as their counterparts in R^3.

As a demonstration of the feasibility of our proposed method, we first construct a Bézier quaternion curve with n control unit quaternions. Then, the cubic Bézier quaternion curve is used to construct a Hermite quaternion curve. We also construct a k-th order B-spline quaternion curve which is C^{k-2}-continuous and locally controllable. There are many other spline curves which can be defined by weighted sums of control points; our construction

[1]This research was partially supported by the Korean Ministry of Science and Technology under the contract 94-S-05-A-03 of STEP 2000 and TGRC-KOSEF.
[2]Computer Science Department, KAIST, Taejeon 305-701, Korea.
[3]Dept. of Computer Science, POSTECH, Pohang 790-784, Korea.
E-mail: {mjkim, syshin}@jupiter.kaist.ac.kr and mskim@vision.postech.ac.kr.

method is general in that all these spline curves are extendible to their corresponding quaternion curves. Since it is possible to manipulate the position curve as well as the corresponding orientation curve in a unified manner, the modeling and manipulation of rigid motions can be managed more conveniently.

The key to the success of our construction method is that the quaternion curve is formulated as a product of simple primitive quaternion curves: $q_i(t) = \exp(\omega_i f_i(t))$, $i = 1, \ldots, n$, for some fixed angular velocity $\omega_i \in R^3$ and real valued function $f_i(t)$, where $\exp : R^3 \to SO(3)$ is an exponential map [3]. Each primitive curve, $q_i(t)$, represents a rotation about a fixed axis, $\omega_i \in R^3$, for which the derivative formula has an extremely simple form: $q_i'(t) = \exp(\omega_i f_i(t))(\omega_i f_i'(t)) = q_i(t)(\omega_i f_i'(t))$. Since the chain rule can be applied to the quaternion product: $q(t) = q_1(t) \cdots q_n(t)$, the resulting differential formula has the simplest expression that one can expect for unit quaternion curves. Furthermore, a similar technique can be used to obtain high order derivatives of our quaternion curves. There have been many other methods suggested for the construction of quaternion curves, including the CAGD approaches based on constructing rational curves on the 3-sphere S^3 [8, 20]. Nevertheless, no method has provided such a simple derivative formula. The only exception is the method of Pobegailo [15] which constructs a C^k-continuous spherical curve as a product of $(k+1)$ rotation matrices: $R_{k+1} R_k \ldots R_1$. The generated curve is, however, essentially restricted to the Bézier type quaternion curve of our method.

The development of closed form equations for the second order derivatives provides an important step toward solving the important problem of torque minimization for 3D rotations [1, 6, 7]. However, the torque minimization problem requires much more. The optimal solution is given as a critical path in the problem of calculus of variations among a set of quaternion paths which satisfy the given conditions. For this purpose, we may need to extend the quaternion curve construction scheme of this paper to that of quaternion surfaces and volumes. This is currently a difficult open problem. Therefore, the important problem of torque minimization for 3D rotations has not been solved in this paper. Nevertheless, the basic algebraic approach taken in this paper provides an important conceptual framework for future research toward this goal.

The rest of this paper is organized as follows. In Section 2, we briefly review some useful concepts and definitions related to unit quaternions. Section 3 describes the main motivation of this work. Section 4 outlines our basic idea for constructing unit quaternion curves. Section 5 constructs the Bézier, Hermite, and B-spline quaternion curves and discusses their differential properties. Section 6 shows some experimental results with discussions on possible further extensions. Finally, in Section 7, we conclude this paper.

2 PRELIMINARIES

2.1 Quaternion and Rotation

Given a unit quaternion $q \in S^3$, a 3D rotation $R_q \in SO(3)$ is defined as follows:

$$R_q(p) = qpq^{-1}, \quad \text{for } p \in R^3, \tag{1}$$

where $p = (x, y, z)$ is interpreted as a quaternion $(0, x, y, z)$ and the quaternion multiplication is assumed for the products [3, 17, 18]. Let $q = (\cos\theta, \hat{v}\sin\theta) \in S^3$, for some angle θ and unit vector $\hat{v} \in S^2$, then R_q is the rotation by angle 2θ about the axis $\hat{v}$. The multiplicative constant, 2, in the angle of rotation, 2θ, is due to the fact that q appears twice in Equation (1). Also note that $R_q \equiv R_{-q}$; that is, two antipodal points, q and $-q$ in S^3, represent the same rotation in $SO(3)$.

The two spaces S^3 and $SO(3)$ have the same local topology and geometry. The major difference is in the distance measures of the two spaces $SO(3)$ and S^3. A rotation curve $R_{q(t)} \in SO(3)$ is twice as long as the corresponding unit quaternion curve $q(t) \in S^3$. When a smooth rotation curve $R_{q(t)}$ has an angular velocity $2\omega(t) \in R^3$, the unit quaternion curve $q(t) \in S^3$ satisfies:

$$q'(t) = q(t)\omega(t). \tag{2}$$

In this paper, we construct the unit quaternion curves in S^3, instead of $SO(3)$; therefore, we interpret the velocity component $\omega(t)$ of $q'(t)$ as the angular velocity, instead of $2\omega(t)$.

The unit quaternion multiplication is not commutative; the order of multiplication is thus very important. Let $q_1, q_2, \cdots, q_n$ be a sequence of successive rotations. When each q_i is measured in the global frame, the product $q_n q_{n-1} \cdots q_1$ is the net rotation of successive rotations. However, when each q_i is measured in the local frame, the final product $q_1 q_2 \cdots q_n$ represents the net rotation. Note that the latter is the same as the successive rotations of $q_n, q_{n-1}, \cdots, q_2, q_1$ in the global frame. Here, we assume each rotation is specified in the local frame. This is simply for notational convenience; in the local frame, we can write the multiplications in the same order as the rotations. By reversing the order of quaternion multiplications, the same construction schemes can be applied to the quaternion curves defined in the global frame.

2.2 Exponential and Logarithmic Maps

Given a vector $v = \theta\hat{v} \in R^3$, with $\hat{v} \in S^2$, the exponential:

$$\exp(v) = \sum_{i=0}^{\infty} v^i = (\cos\theta, \hat{v}\sin\theta) \in S^3,$$

becomes the unit quaternion which represent the rotation by angle 2θ about the axis $\hat{v}$, where v^i is computed using the quaternion multiplication [3, 11, 18]. The exponential map exp can be interpreted as a mapping from the angular velocity vector (measured in S^3) into the unit quaternion which represents the rotation. When we limit the domain of exp so that $|\theta| < \pi$, the map exp becomes 1-to-1 and its inverse map log can be defined for unit quaternions. (See [9] for more details on exp and log.) The power of a unit quaternion q with respect to a real valued exponent α is defined to be a unit quaternion: $q^\alpha = \exp(\alpha \log q)$.

Given two unit quaternions q_1 and q_2, the geodesic curve $\gamma_{q_1,q_2} \in S^3$ (which connects q_1 and q_2) has constant tangential velocity $\log(q_1^{-1}q_2)$. The geodesic curve equation is given by:

$$\gamma_{q_1,q_2}(t) = q_1 \exp(t \cdot \log(q_1^{-1}q_2)) = q_1(q_1^{-1}q_2)^t \; . \tag{3}$$

The geodesic γ_{q_1,q_2} is also called the *slerp* (spherical linear interpolation) between q_1 and q_2.

3 MOTIVATION

For a translational motion in R^3, the position curve $p(t)$ is represented by:

$$p(t) = \int v(t)dt + p_0, \tag{4}$$

where $v(t)$ is the velocity and p_0 is the initial position. This relation can also be represented by the following linear differential equation:

$$p'(t) = v(t) \; . \tag{5}$$

However, as shown in Equation (1), the relation between $q(t)$ and $\omega(t)$ is non-linear: $q'(t) = q(t)\omega(t)$.

One of the most important problems in aero-space engineering is how to find a torque optimal rotational path which connects the start and target orientations [7]. Many numerical methods have been suggested to find the optimal path, in which Equation (1) plays an important role as governing equation [6]. Barr et al. [1] also raised torque optimization as a significant problem in computer animation, and they constructed a torque optimal piecewise slerp quaternion path (i.e., as a sequence of discrete unit quaternions) by using a non-linear optimization technique. There still remains the important open problem of how to construct such a torque optimal path with piecewise spline quaternion curves which have simple closed form equations. An immediate open problem concerns the computation of high order derivatives of a unit quaternion curve. In this paper, we resolve the crucial problem of how to formulate the quaternion curves and their high order derivatives as closed form equations.

For a unit quaternion curve $q(t) \in S^3$, we may reformulate the curve in the following equivalent form:

$$q(t) = \exp(\log(q(t))).$$

By applying the chain rule to this equation, we obtain the first derivative formula:

$$q'(t) = d(\exp)_{\log(q(t))} \left(\frac{d}{dt} \log(q(t)) \right),$$

where $d(\exp)_{\log(q(t))}$ is the differential (i.e., Jacobian matrix) of exp. Kim, Kim, and Shin [9] used this formula to provide simple C^1-continuity proofs for various previous methods [11, 17, 18]. The second and high order derivative formulas, however, become extremely complex even with this representation. Moreover, for general quaternion curves $q(t)$, their logarithmic curves, $\log(q(t))$, also have very complex formulas. In this paper, we exploit simple primitive quaternion curves $q(t)$ which allow simple formulas for both the high order derivatives and the logarithmic curves.

Shoemake [18] used the formula:

$$dq^\alpha = q^\alpha \log(q) d\alpha + \alpha q^{\alpha-1} dq. \qquad (6)$$

In general, this formula is incorrect. For example, Equation (6) claims that: $dq^2 = 2q\ dq$. However, we have:

$$dq^2 = d(qq) = dq\ q + q\ dq \neq 2q\ dq$$

since the quaternion multiplication is not commutative (also see [9] for more details). Nevertheless, there is a special case where this formula works. When the quaternion curve $q(t)$ is restricted to the rotation around a fixed axis $\omega \in R^3$: i.e.,

$$q(t) = \exp(\omega\alpha(t)),$$

for a real-valued function $\alpha(t)$, $q'(t)$ has a simple form:

$$q'(t) = \exp(\omega\alpha(t))(\omega\alpha'(t)) = q(t)(\omega\alpha'(t)),$$

which is equivalent to the formula of Equation (6) in this special case. Higher order derivatives of $q(t)$ are also easy to compute by the chain rule.

To make good use of this simple differential property, all the unit quaternion curves of this paper will be constructed as the products of primitive quaternion curves: $q_i(t) = \exp(\omega_i \alpha_i(t))$, $i = 1, \ldots, n$, for a fixed angular velocity $\omega_i \in R^3$ and a real valued function $\alpha_i(t)$. Since the chain rule can be applied to the quaternion product: $q(t) = q_1(t) \cdots q_n(t)$, the quaternion curve derivative can be obtained in an extremely simple form. Furthermore, applying a similar technique recursively, high order derivatives can also be obtained in simple forms. In this paper, we construct each component quaternion curve $q_i(t)$ to be C^k-continuous by choosing C^k-continuous basis function $\alpha_i(t)$. Therefore, their quaternion product $q(t) = q_1(t) \cdots q_n(t)$ becomes C^k-continuous.

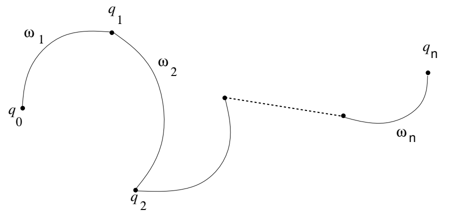

Figure 1: Piecewise Slerp Interpolation of $\{\omega_i\}$.

4 BASIC IDEA

4.1 Cumulative Form

Given a sequence of points $p_0, p_1, \cdots, p_n \in R^3$, the simplest C^0-continuous curve $p(t) \in R^3$, which interpolates each point p_i at $t = i$, is given by the following linear interpolation:

$$\begin{aligned} p(t) &= p_0 + \alpha_1(t)\Delta p_1 + \alpha_2(t)\Delta p_2 + \cdots + \alpha_n(t)\Delta p_n \\ &= p_0 + \textstyle\sum_{i=1}^{n} \alpha_i(t)\Delta p_i, \end{aligned}$$

where

$$\Delta p_i = p_i - p_{i-1}$$

$$\alpha_i(t) = \begin{cases} 0 & \text{if } t < i \\ t - i & \text{if } i \leq t < i+1 \\ 1 & \text{if } t \geq i+1 \end{cases}$$

Similarly, given a sequence of unit quaternions $q_0, \cdots, q_n \in S^3$, we can construct a C^0-continuous unit quaternion curve $q(t) \in S^3$, which interpolates each unit quaternion q_i at $t = i$, as follows:

$$\begin{aligned} q(t) &= q_0 \exp(\omega_1\alpha_1(t)) \exp(\omega_2\alpha_2(t)) \cdots \exp(\omega_n\alpha_n(t)) \\ &= q_0 \textstyle\prod_{i=1}^{n} \exp(\omega_i\alpha_i(t)), \end{aligned}$$

where

$$\omega_i = \log(q_{i-1}^{-1} q_i).$$

This is a piecewise *slerp* (spherical linear interpolation) of $\{q_i\}$ (see Figure 1). In the rotational space, a *slerp* implies a rotation with a constant angular velocity (around a fixed rotation axis). The slerp curve segment joining q_i and q_{i+1} is the geodesic interpolation, which is given by: $q_i \exp(\omega_i\alpha_i(t))$, based on Euler's theorem of principal rotation.

At this point, we may generalize $\{\alpha_i\}$ to other functions, rather than a piecewise linear function, so that the resulting quaternion curve becomes C^k-continuous while interpolating the given sequence of unit quaternions. The two sequences $\{\alpha_i\}$ and $\{\omega_i\}$ can be viewed as basis functions and their coefficients, respectively. We call $q_0 \prod \exp(\omega_i\alpha_i)$ the *cumulative form* of $q(t)$ with their coefficients $\{\omega_i\}$. The cumulative form has several advantages over other quaternion curve representations:

1. It has a simple closed form equation, which simplifies the evaluation of curve points and reduces the numerical errors.
2. It facilitates straight-forward computations of high order derivatives.
3. Using C^k-continuous basis functions $\{\alpha_i\}$, we can easily construct C^k-continuous quaternion curves, which are further controlled by the coefficients $\{\omega_i\}$.
4. A well-chosen set of basis functions makes the constructed quaternion curves locally controllable.
5. Since $\|\exp(\alpha\omega)\| = 1$, the quaternion curves are in S^3.

4.2 Cumulative Basis

In the Euclidean space R^3, there are many well-known spline curve construction schemes such as Bézier and B-spline curves. Most of the spline curves are represented as the sums of basis functions with their control points as the coefficients. Let $\{B_i\}$ be the basis functions and $\{p_i\}$ be the control points. Then, the spline curve $p(t)$, determined by $\{p_i\}$, is given by:

$$p(t) = \sum_{i=0}^{n} p_i B_i(t) ,$$

which is called the *basis form* of $p(t)$. We present a general scheme that converts the basis form to the cumulative form of the quaternion curve. Using this method, we can easily construct various unit quaternion curves from their analogues in the Euclidean space R^3. The basis form can be first converted into the following form:

$$p(t) = p_0 \widetilde{B}_0(t) + \sum_{i=1}^{n} \Delta p_i \widetilde{B}_i(t) ,$$

where

$$\begin{aligned} \Delta p_i &= p_i - p_{i-1} , \\ \widetilde{B}_i(t) &= \textstyle\sum_{j=i}^{n} B_i(t) . \end{aligned}$$

Then, the corresponding quaternion curve is obtained as follows:

$$q(t) = q_0^{\widetilde{B}_0(t)} \prod_{i=1}^{n} \exp(\omega_i \widetilde{B}_i(t)) , \tag{7}$$

by converting $p(t)$ to $q(t)$, p_0 to q_0, Δp_i to $\omega_i = \log(q_{i-1}^{-1} q_i)$, and vector addition to quaternion multiplication. Equation (7) is given in a cumulative form. One needs to be extremely careful in the order of multiplication: $q_{i-1}^{-1} q_i$. If the angular velocities are given in the global frame, the quaternion multiplication: $q_i q_{i-1}^{-1}$ should be used, instead of $q_{i-1}^{-1} q_i$. The new basis $\{\widetilde{B}_i\}$ is called the *cumulative basis* of $\{B_i\}$. The cumulative form is the basic tool for our quaternion curve construction in $SO(3)$. Simply by deriving a cumulative basis $\{\widetilde{B}_i\}$, we can easily obtain a quaternion curve which shares many important differential properties with its counterpart in the Euclidean space R^3.

5 A NEW CLASS OF QUATERNION CURVES

5.1 Bézier Quaternion Curve

We can represent an n-th order Bézier curve with Bernstein basis $\beta_{i,n}(t) = \binom{n}{i}(1-t)^{n-i} t^i$ as follows:

$$p(t) = \sum_{i=0}^{n} p_i \beta_{i,n}(t) , \tag{8}$$

where p_i's are the control points. For the Bézier curve given in a basis form, we can apply our quaternion curve construction method. We first reformulate Equation (8) as follows:

$$p(t) = p_0 \tilde{\beta}_{0,n}(t) + \sum_{i=1}^{n} \Delta p_i \tilde{\beta}_{i,n}(t) ,$$

where the cumulative basis functions are given by:

$$\tilde{\beta}_{i,n}(t) = \sum_{j=i}^{n} \beta_{i,n}(t) . \tag{9}$$

Then, by converting it to the cumulative form, we can obtain the n-th order Bézier quaternion curve with control points $\{q_i\}$ as follows:

$$q(t) = q_0 \prod_{i=1}^{n} \exp(\omega_i \tilde{\beta}_{i,n}(t)) , \tag{10}$$

where

$$\omega_i = \log(q_{i-1}^{-1} q_i) .$$

Note that $\tilde{\beta}_{0,n}(t) = 1$. This Bézier quaternion curve has a different shape from the Bézier quaternion curve of Shoemake [17].

5.2 Hermite Quaternion Curve

A cubic Hermite curve is defined by two end points, p_a and p_b, and two end velocities, v_a and v_b. Alternatively, the Hermite curve can be represented by a cubic Bézier curve:

$$p(t) = \sum_{i=0}^{3} p_i \beta_i(t) , \tag{11}$$

with the condition:

$$p_0 = p_a,\ p_1 = p_a + v_a/3,\ p_2 = p_b - v_b/3,\ p_3 = p_b, \tag{12}$$

where $\beta_i(t) = \beta_{i,3}(t)$ for $i = 0, 1, 2, 3$.

Similarly, a cubic Bézier quaternion curve can be used to define a Hermite quaternion curve which interpolates two end unit quaternions, q_a and q_b, and two end angular velocities, ω_a and ω_b. From Equation (10), the cubic Bézier quaternion curve is given by:

$$q(t) = q_0 \prod_{i=1}^{3} \exp(\omega_i \tilde{\beta}_i(t)) , \tag{13}$$

where $\tilde{\beta}_i(t) = \tilde{\beta}_{i,3}(t)$ for $i = 1, 2, 3$. The quaternion counterpart of Equations (12) is given by:

$$q_0 = q_a,\ q_1 = q_a \exp(\omega_a/3),\ q_2 = q_b \exp(\omega_b/3)^{-1},\ q_3 = q_b.$$

These four identities determine the three coefficients ω_i of the cubic Bézier quaternion curve in Equation (13) as follows:

$$\begin{aligned} \omega_1 &= \log(q_0^{-1} q_1) = \log(q_a^{-1} q_a \exp(\omega_a/3)) = \omega_a/3 \\ \omega_2 &= \log(q_1^{-1} q_2) = \log(\exp(\omega_a/3)^{-1} q_a^{-1} q_b \exp(\omega_b/3)^{-1}) \\ \omega_3 &= \log(q_2^{-1} q_3) = \log(\exp(\omega_b/3) q_b^{-1} q_b) = \omega_b/3. \end{aligned}$$

Using these three angular velocities, we can construct a cubic Hermite quaternion curve from Equation (13). Note that we can assign arbitrarily large angular velocities at the curve end points. The angular velocity ω_2 provides an extra degree of freedom in choosing the number n of revolutions while not losing the end point interpolation property. That is, instead of ω_2, we may use

$$\omega_2 + \widehat{\omega}_2 \cdot n\pi \quad \text{for an integer } n,$$

where $\widehat{\omega}_2 = \omega_2/\|\omega_2\|$. The curve shape changes, depending on the number of revolutions. (Note that the angular velocity is measured here in S^3; therefore, the magnitude is half the rotation in the physical space.) Figure 2 shows the graphs of basis functions $\tilde{\beta}_i$'s. Figure 3 shows Hermite quaternion curves with the same control points, but with different angular velocities ω_b's. Since it is difficult to visualize the quaternion curves in S^3, they are projected onto a unit sphere in R^3.

Now, we will show that the cubic Hermite quaternion curve interpolates the two orientations, q_a and q_b, and the two angular

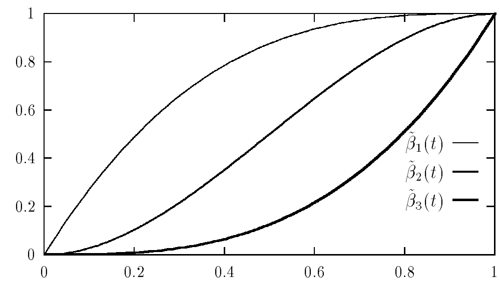

Figure 2: Graphs of $\tilde{\beta}_i$.

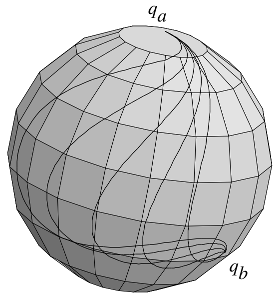

Figure 3: Examples of Hermite Quaternion Curves.

velocities, ω_a and ω_b, at the curve end points. It is easy to show that:

$$\begin{aligned} q(0) &= q_0 \exp(0)\exp(0)\exp(0) \\ &= q_0 = q_a, \\ q(1) &= q_0 \exp(\omega_1)\exp(\omega_2)\exp(\omega_3) \\ &= q_0(q_0^{-1}q_1)(q_1^{-1}q_2)(q_2^{-1}q_3) \\ &= q_3 = q_b \end{aligned}$$

The first derivative of $q(t)$ is given by:

$$\begin{aligned} q'(t) &= q_0 \exp(\omega_1\tilde{\beta}_1(t))(\omega_1\dot{\tilde{\beta}}_1(t))\exp(\omega_2\tilde{\beta}_2(t))\exp(\omega_3\tilde{\beta}_3(t)) \\ &\quad + q_0 \exp(\omega_1\tilde{\beta}_1(t))\exp(\omega_2\tilde{\beta}_2(t))(\omega_2\dot{\tilde{\beta}}_2(t))\exp(\omega_3\tilde{\beta}_3(t)) \\ &\quad + q_0 \exp(\omega_1\tilde{\beta}_1(t))\exp(\omega_2\tilde{\beta}_2(t))\exp(\omega_3\tilde{\beta}_3(t))(\omega_3\dot{\tilde{\beta}}_3(t)) \end{aligned}$$

where

$$\begin{aligned} &\tilde{\beta}_1(t) = 1-(1-t)^3,\ \tilde{\beta}_2(t) = 3t^2-2t^3,\ \tilde{\beta}_3(t) = t^3, \\ &\dot{\tilde{\beta}}_1(t) = \ 3(1-t)^2, \dot{\tilde{\beta}}_2(t) = \ 6t(1-t), \dot{\tilde{\beta}}_3(t) = 3t^2. \end{aligned}$$

Thus, we have:

$$\begin{aligned} q'(0) &= q_0 \exp(0)(\omega_1 3)\exp(0)\exp(0) \\ &\quad + q_0 \exp(\omega_1\tilde{\beta}_1(t))\exp(0)(\omega_2 0)\exp(\omega_3\tilde{\beta}_3(t)) \\ &\quad + q_0 \exp(\omega_1\tilde{\beta}_1(t))\exp(\omega_2\tilde{\beta}_2(t))\exp(0)(\omega_3 0) \\ &= q_0\omega_a = q_a\omega_a \end{aligned}$$

$$\begin{aligned} q'(1) &= q_0 \exp(\omega_1)(\omega_1 0)\exp(\omega_2)\exp(\omega_3) \\ &\quad + q_0 \exp(\omega_1)\exp(\omega_2)(\omega_2 0)\exp(\omega_3) \\ &\quad + q_0 \exp(\omega_1)\exp(\omega_2)\exp(\omega_3)(\omega_3 3) \\ &= q_0 q_0^{-1} q_1 q_1^{-1} q_2 q_2^{-1} q_3(3\omega_3) \\ &= q_3\omega_b = q_b\omega_b, \end{aligned}$$

which means that the quaternion curve $q(t)$ has its angular velocities $\omega_a = q_a^{-1}q'(0) = q(0)^{-1}q'(0)$ and $\omega_b = q_b^{-1}q'(1) = q(1)^{-1}q'(1)$ at the curve end points.

5.3 B-spline Quaternion Curve

The B-spline curve is very popular in computer graphics because of its extreme smoothness and local controllability. By moving a control point, we can selectively modify the B-spline curve without losing its geometric continuity. In this section, we consider how to convert a B-spline curve in R^3 into its quaternion analogue with a cumulative form. Then, we investigate the properties such as C^k-continuity and local controllability.

A B-spline curve is defined by a weighted sum of B-spline basis functions $B_{i,k}(t)$, which are C^{k-2}-continuous $(k-1)$-th order piecewise polynomials. Given a set of control points $\{p_i\}$, a B-spline curve $p(t)$ is given by:

$$p(t) = \sum_{i=0}^{n} p_i B_{i,k}(t).$$

The B-spline basis functions $B_{i,k}(t)$ are defined by the following recurrence relation [4]:

$$B_{i,1}(t) = \begin{cases} 1 & \text{if } t_i < t < t_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

and

$$B_{i,k}(t) = \frac{t-t_i}{t_{i+k-1}-t_i} B_{i,k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}} B_{i+1,k-1}(t)\ .$$

It is easy to show that $B_{i,k}$'s are C^{k-2}-continuous piecewise polynomials of degree $(k-1)$. They are C^{k-2}-continuous everywhere, and may not be C^{k-1}-continuous only at the knots $\{t_i\}$. Each $B_{i,k}(t)$ has a non-zero support on the interval $[t_i, t_{i+k}]$, i.e., $B_{i,k}(t) = 0$ for $t < t_i$ or $t > t_{i+k}$.

The B-spline curve may be reformulated in the following cumulative form:

$$p(t) = p_0 \widetilde{B}_{0,k}(t) + \sum_{i=1}^{n} \Delta p_i \widetilde{B}_{i,k}(t)\ ,$$

where

$$\Delta p_i = p_i - p_{i-1}$$

and

$$\begin{aligned} \widetilde{B}_{i,k}(t) &= \textstyle\sum_{j=i}^{n} B_{j,k}(t) \\ &= \begin{cases} \sum_{j=i}^{i+k} B_{j,k}(t) & \text{if } t_i < t < t_{i+k-1} \\ 1 & \text{if } t \geq t_{i+k-1} \\ 0 & \text{if } t \leq t_i \end{cases} \end{aligned}$$

By converting $p(t)$ to $q(t)$, p_0 to q_0, Δp_i to ω_i, and summation to quaternion multiplication, the corresponding quaternion curve is obtained in a cumulative form as follows:

$$q(t) = q_0^{\widetilde{B}_{0,k}(t)} \prod_{i=1}^{n} \exp(\omega_i \widetilde{B}_{i,k}(t))\ ,$$

where $\omega_i = \log(q_{i-1}^{-1}q_i)$. This gives our B-spline quaternion curve, which is C^{k-2}-continuous and locally controllable with the control points $\{q_i\}$ and the angular velocities $\{\omega_i\}$. The B-spline quaternion curve also allows arbitrarily large angular velocities between two consecutive control points $\{q_i\}$. Figures 4 and 5 show the graphs of basis functions $B_{i,4}(t)$ and $\widetilde{B}_{i,4}(t)$, respectively. Note that $\widetilde{B}_{i,k}(t)$ is non-constant only in the interval $[t_i, t_{i+k-1}]$, whereas $B_{i,k}(t)$ is non-constant in $[t_i, t_{i+k}]$. Figure 6 shows examples of B-spline quaternion curves. The local shape controllability is shown with dashed curves.

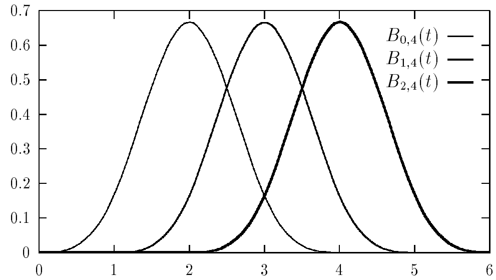

Figure 4: B-spline Basis Functions.

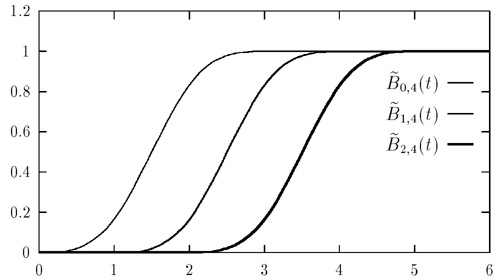

Figure 5: Cumulative B-spline Basis Functions.

Now, we can investigate some interesting differential and geometric properties of the B-spline quaternion curve: i.e., C^k-continuity and local controllability. It is easy to show that $\widetilde{B}_{i,k}(t)$ is C^{k-2}-continuous since it is a weighted sum of C^{k-2}-continuous functions $\{B_{i,k}\}$. Therefore, the k-th order B-spline quaternion curve is C^{k-2}-continuous as well. Since the cumulative basis function $\widetilde{B}_{i,k}(t)$ is non-constant in the interval $[t_i, t_{i+k-1}]$, the quaternion curve $q(t)$ on the interval $t_l \leq t < t_{l+1}$ can be represented by:

$$\begin{aligned} q(t) &= q_0 \left(\prod_{i=1}^{l-k+1} \exp(\omega_i \cdot 1)\right) \left(\prod_{i=l-k+2}^{l} \exp(\omega_i \widetilde{B}_{i,k}(t))\right) \\ &\quad \left(\prod_{i=l+1}^{n} \exp(\omega_i \cdot 0)\right) \\ &= q_{l-k+1} \prod_{i=l-k+2}^{l} \exp(\omega_i \widetilde{B}_{i,k}(t)). \end{aligned}$$

This equation shows that the quaternion curve $q(t)$ depends only on the quantities: $q_{l-k+1}, \omega_{l-k+1}, \cdots, \omega_l$, that is, only on the k control points: $q_{l-k+1}, q_{l-k}, \cdots, q_l$. In other words, by moving the control point q_l, the curve shape is influenced only on the interval $[t_l, t_{l+k}]$.

Furthermore, the derivation of $q'(t)$ is extremely simple due to the B-spline differentiation formula [4]:

$$\frac{d}{dt} \sum \alpha_i B_{i,k}(t) = \sum (k-1) \frac{\alpha_i - \alpha_{i-1}}{t_{i+k-1} - t_i} B_{i,k-1}(t),$$

where α_i's are constants. Therefore, we have:

$$\widetilde{B}'_{i,k}(t) = \frac{k-1}{t_{i+k-1} - t_i} B_{i,k-1}(t).$$

In the case of a uniform B-spline, there is a further reduction to: $\widetilde{B}'_{i,k}(t) = B_{i,k-1}(t)$.

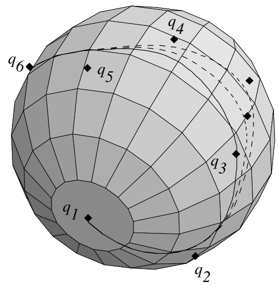

Figure 6: B-spline Quaternion Curve with Six Control Points.

6 EXPERIMENTAL RESULTS

6.1 Torque Computation

The spline interpolation in R^3 produces a path p which minimizes the energy:

$$\int \|\ddot{p}\|^2 dt,$$

while satisfying given constraints. Thus, the spline interpolation path does not generate unnecessary force (or acceleration). The energy minimization property makes the spline interpolation very useful. As far as rotational motion is concerned, it is important to minimize torque (or angular acceleration). Most of the previous results on quaternion interpolation have concentrated on improving the computational efficiency rather than attacking the more challenging problem of energy minimization.

Barr et al. [1] took an important step toward the energy minimization. The quaternion path is approximated by discrete unit quaternions and a time-consuming non-linear optimization is employed in the algorithm. There still remains an important open problem concerning how to construct an energy minimization quaternion path, ideally with a closed form equation and without using a time-consuming optimization technique. In this paper, we have provided a useful step toward the resolution of this problem by introducing a new class of spline quaternion curves for which high order derivatives can be obtained in simple closed form equations. Although the problem of how to deal with these closed form equations is still unsolved, our quaternion curves exhibit promising behaviors by generating relatively small torque compared with those generated by the previous methods.

Our cubic quaternion curves generate a similar amount of torque (in the range of $\pm 5\%$) to that of Shoemake [17] when the control unit quaternions have small angular variations. However, our curves perform much better when the variations are large. The methods based on recursive slerp constructions may generate twisted curves when the spherical control polygon on S^3 has edges of relatively large lengths. This effect can be explained as follows. The slerp-based methods generate a sequence of intermediate quaternion curves $q_{i,k}(t)$'s, and blend each pair of two consecutive curves $q_{i,k}(t)$ and $q_{i+1,k}(t)$ to generate another sequence of quaternion curves $q_{i,k+1}(t)$'s:

$$q_{i,k+1}(t) = q_{i,k}(t) \exp(f_i(t) \log(q_{i,k}(t)^{-1} q_{i+1,k}(t))),$$

for some blending function $f_i(t)$. When the variations of control quaternions are large, the two quaternion curves $q_{i,k}(t)$ and $q_{i+1,k}(t)$ have large curve lengths. They may have complex winding shapes in the compact space S^3. The difference quaternion curve

$q_{i,k}(t)^{-1}q_{i+1,k}(t)$ would also wind many times. However, the measure, $\log(q_{i,k}(t)^{-1}q_{i+1,k}(t))$, is always bounded by π, which totally ignores the large amount of winding. As a result, the blending curve $q_{i,k+1}(t)$ experiences large bending, which produces large torque. As the intermediate curves with bending shapes are blended recursively, the resulting quaternion curve has a twisting shape.

Our quaternion curves do not suffer from such a degeneracy. This is because our primitive quaternion curve $q_i(t) = \exp(\omega_i \alpha_i(t))$ may accommodate arbitrarily large angular velocity ω_i. The resulting curve has a large number of winding; however, the curve does not produce extraordinary bending and/or twisting. Therefore, our curves perform much better when the angular variations are large. Furthermore, our curves have C^2-continuity, which means that there is no torque jump at the curve joint. The piecewise cubic quaternion curves (based on the de Casteljau type construction) are not C^2-continuous; they have torque jump at each curve joint. High degree rational spherical curves have C^2-continuity [19]; however, their speeds are less uniform than the curves based on the de Casteljau type construction. Therefore, the rational curves generate redundant tangential accelerations, which has undesirable effect.

6.2 Animation Examples

We present some examples to demonstrate the feasibility of our quaternion curves. Figure 7 shows an animation of a flying boomerang. In this example, the motion path is composed of a Hermite curve for the translation and a Hermite quaternion curve for the rotation. Using the Hermite quaternion curve, we can specify arbitrary orientations and angular velocities of the boomerang at the start and end of the animated motion. Note that the boomerang experiences many revolutions. This effect is obtained by assigning large angular velocities at both ends.

Our cubic B-spline quaternion curve produces extremely smooth motions. Figure 8 shows a motion path, which is specified by a B-spline curve with six control points. A rigid motion path can be used to specify the sweeping of a 2D cross section. Figure 9 shows a sweep object generated from the same motion path given in Figure 8. Figure 10 shows an example of B-spline quaternion interpolation for a rigid body, where the position and orientation interpolation curves are constructed by the B-spline interpolation curves in R^3 and $SO(3)$, respectively. Six keyframes are used in this example, and they are shown in dark tints.

7 CONCLUSIONS

A general construction method is proposed for unit quaternion curves. Given a spline curve in R^3, the spline curve is reformulated in a cumulative basis form and the corresponding quaternion curve is constructed by converting each vector addition into the quaternion multiplication. The quaternion curve is formulated as a finite product of simple quaternion curves, which makes the evaluation of high order derivatives quite straightforward. The constructed quaternion curves preserve many important differential properties of their counterparts in R^3. Furthermore, the quaternion curves and their high order derivatives are given by simple closed form equations.

Experimental results are quite promising in that our quaternion curves use small torque compared with the previous quaternion curves, especially when the control quaternions have large variations. Although the important torque minimization problem has not been solved in this paper, our approach provides an important initial step toward this goal.

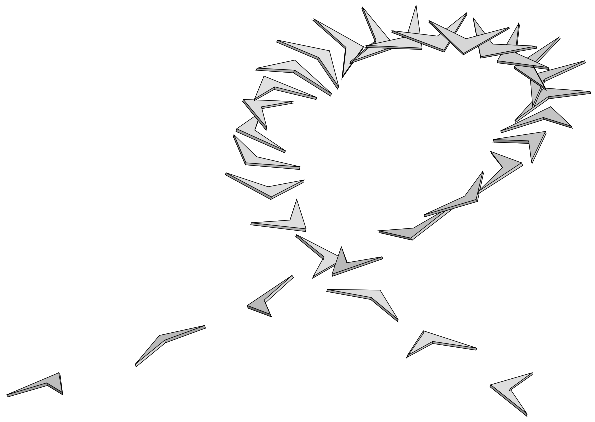

Figure 7: Boomerang Animation using one Hermite Quaternion Curve.

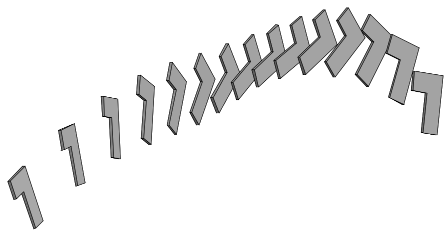

Figure 8: Rigid Motion by a Cubic B-spline Quaternion Curve.

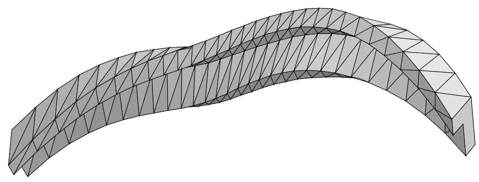

Figure 9: Sweeping with Cubic B-spline Quaternion Curves.

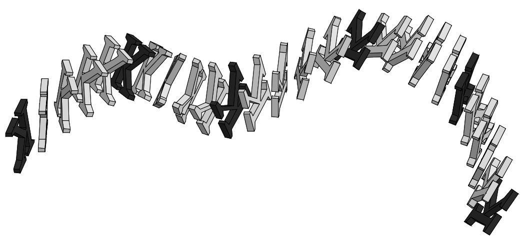

Figure 10: Example of B-spline Quaternion Interpolation.

REFERENCES

[1] BARR, A., CURRIN, B., GABRIEL, S., AND HUGHES, J. Smooth interpolation of orientations with angular velocity constraints using quaternions. *Computer Graphics (Proc. of SIGGRAPH '92)* (1992), pp. 313–320.

[2] BARRY, P., AND GOLDMANN, R. A recursive evaluation algorithm for a class of Catmull-Rom splines. *Computer Graphics (Proc. of SIGGRAPH '88)* (1988), pp. 199–204.

[3] CURTIS, M. *Matrix Groups*, Springer-Verlag, 1972.

[4] DE BOOR, C. *A Practical Guide to Splines*, Springer-Verlag, 1978.

[5] HAMILTON, W. *Elements of Quaternions (Volume I, II)*, Chelsea Publishing Company, 1969.

[6] JUNKINS, J., AND TURNER, J. Optimal continuous torque attitude maneuvers. *J. Guidance and Control 3*, 3 (1980), pp. 210–217.

[7] JUNKINS, J., AND TURNER, J. *Optimal Spacecraft Rotational Maneuvers*, Elsevier, 1986.

[8] JUTTLER, B. Visualization of moving objects using dual quaternion curves. *Computers & Graphics 18*, 3 (1994), pp. 315–326.

[9] KIM, M.-J., KIM, M.-S., AND SHIN, S. A compact differential formula for the first derivative of a unit quaternion curve. To appear in *J. of Visualization and Computer Animation* (1995).

[10] KIM, M.-J., KIM, M.-S., AND SHIN, S. A C^2-continuous B-spline quaternion curve interpolating a given sequence of solid orientations. *Proc. of Computer Animation '95* (1995), pp. 72–81.

[11] KIM, M.-S., AND NAM, K.-W. Interpolating solid orientations with circular blending quaternion curves. To appear in *Computer-Aided Design* (1995).

[12] NIELSON, G. Smooth interpolation of orientation. *Models and Techniques in Computer Animation (Proc. of Computer Animation '93)* (1993), N. Thalmann and D. T. (Eds.), Eds., Springer-Verlag, pp. 75–93.

[13] NIELSON, G., AND HEILAND, R. Animated rotations using quaternions and splines on a 4D sphere. *Programmirovanie(Russia)* (July-August 1992), Springer-Verlag, pp. 17–27. English edition, *Programming and Computer Software*, Plenum Pub., New York.

[14] PLETINCKS, D. Quaternion calculus as a basic tool in computer graphics. *The Visual Computer 5*, 1 (1989), pp. 2–13.

[15] POBEGAILO, A. Modeling of C^n spherical and orientation splines. To appear in *Proc. of Pacific Graphics '95* (1995).

[16] SCHLAG, J. Using geometric constructions to interpolate orientation with quaternions. *Graphics GEMS II*, Academic Press, 1992, pp. 377–380.

[17] SHOEMAKE, K. Animating rotation with quaternion curves. *Computer Graphics (Proc. of SIGGRAPH '85)* (1985), pp. 245–254.

[18] SHOEMAKE, K. Quaternion calculus for animation. *Math for SIGGRAPH (ACM SIGGRAPH '91 Course Notes #2)* (1991).

[19] WANG, W. Rational spherical curves. Presented at Int'l. Conf. on CAGD, Penang, Malaysia (July 4-8, 1994).

[20] WANG, W., AND JOE, B. Orientation interpolation in quaternion space using spherical biarcs. *Proc. of Graphics Interface '93* (1993), pp. 24–32.

APPENDIX

In this appendix, we provide the pseudo codes for the construction of quaternion curves presented in Section 5.

Bézier Quaternion Curve

// $q_0, \ldots, q_n$: control points

double $\tilde{\beta}_{i,n}(t) = \sum_{j=i}^{n} \binom{n}{i}(1-t)^{n-i}t^i$

quaternion Bezier$(q_0, q_1, \ldots q_n)(t)$

  return $(q_0 \prod_{i=1}^{n} \exp(\log(q_{i-1}^{-1}q_i)\tilde{\beta}_{i,n}(t)))$;

end

Hermite Quaternion Curve

// q_a, q_b: the start and end orientations
// ω_a, ω_b: the start and end angular velocities

double $\tilde{\beta}_1(t) = 1-(1-t)^3$, $\tilde{\beta}_2(t) = 3t^2 - 2t^3$, $\tilde{\beta}_3(t) = t^3$

quaternion Hermite$(q_a, q_b, \omega_a, \omega_b)(t)$

  $q_0 = q_a$;
  $\omega_1 = \omega_a/3$;
  $\omega_2 = \log(\exp(\omega_1)^{-1}q_a^{-1}q_b\exp(\omega_3))$;
  $\omega_3 = \omega_b/3$;
  return $(q_0 \exp(\omega_1\tilde{\beta}_1(t))\exp(\omega_2\tilde{\beta}_2(t))\exp(\omega_3\tilde{\beta}_3(t)))$;

end

B-spline Quaternion Curve

// t_i : knot sequence

double $B_{i,k}(t)$ = B-spline basis of order k
double $\widetilde{B}_{i,k}(t) = \sum_{j=i}^{i+k} B_{j,k}(t)$

quaternion B-spline$(q_0, q_1, \ldots q_n)(t)$

  $l = \max\{\, i \,|t_{i+k-1} \le t\}$;
  if $(l < 0)$ **then**
    $\{l = 0;\ q = q_0^{\widetilde{B}_{0,k}(t)}\}$
  else
    $q = q_l$;
  for $(i = l+1;\ i \le n\ \&\&\ t_i < t;\ i{+}{+})$
    $q = q\exp(\log(q_{i-1}^{-1}q_i)\widetilde{B}_{i,k}(t))$;
  return (q);

end

Uniform B-spline Quaternion Curve

// $t_i = i - 2$: uniform knots
// $0 \le t \le n$
// uniform-B-spline$(0) = q_0$
// uniform-B-spline$(n) = q_n$
// uniform-B-spline$(i) \approx q_i$

quaternion uniform-B-spline$(q_0, q_1, \ldots q_n)(t)$

  $q_{-1} = q_0 q_1^{-1} q_0$;   //phantom
  $q_{n+1} = q_n q_{n-1}^{-1} q_n$;// control points
  $l = \lfloor t-1 \rfloor$;
  $q = q_l$;
  for $(i = l+1;\ i < t+2;\ i{+}{+})$
    $q = q\exp(\log(q_{i-1}^{-1}q_i)\widetilde{B}_{i,k}(t))$;
  return (q);

end