Computer Graphics International 1999 (CGI '99), Canmore, Canada, June 4 - June 11, 1999

A Shape-Preserving Data Embedding Algorithm for NURBS Curves and Surfaces

Ryutarou Ohbuchi[1]
ohbuchi@acm.org
Yamanashi University
4-3-11 Takeda, Kofu-shi
Yamanashi, 400-0016, Japan

Hiroshi Masuda
masuda@race.u-tokyo.ac.jp
The University of Tokyo
4-6-1 Komaba, Meguro-ku,
Tokyo, 113-8656, Japan

Masaki Aono
aono@acm.org
IBM Tokyo Research Lab.
1623-14 Shimo-tsuruma
Yamato-shi, Kanagawa-ken
242-8502, Japan

Abstract

Existing data embedding algorithms for polygonal meshes and their attributes can't be applied to the majority of (geometric) computer aided design (CAD) applications, for two major reasons. First, these CAD systems employ parametric curves and surfaces, not polygonal meshes, as their main shape-defining primitives. Second, most CAD applications do not tolerate modifications of model topology and/or geometry that are introduced by existing data embedding algorithms.

This paper proposes a new data embedding algorithm for *non-uniform rational B-spline* (*NURBS*) curves and surfaces, which employs rational linear reparameterization for embedding messages. The algorithm exactly preserves the shape, — that is, the geometry and topology — of its embedding targets. Furthermore, it preserves the data size of the model. We consider these two properties, preservation of shape and preservation of data size, can be significant with regard to the use of data embedding in CAD applications.

In addition to the shape- and data size-preserving data embedding algorithm for NURBS curves and surfaces, this paper outlines additional methods for embedding data in various types of parametric curves and surfaces.

Keywords: computer-aided design (CAD), information security, digital watermark, reparameterization.

1. Introduction

Data embedding, or *(digital) watermarking*, puts structures called *watermarks* into digital contents (e.g., images) in such a way that the structures do not interfere with the intended use (e.g., viewing) of the contents. The watermarks carry information that can be used to manage the contents, in order, for example, to add annotations, to detect tampering, or to authenticate rightful purchasers.

Previous studies have focused on data embedding techniques for "traditional" digital multimedia content data types, such as text, image, video, and audio [22, 26, 1, 3, 15, 21, 4, 8, 14, 25]. Recently, as 3D models gain status as an important multimedia data type [8, 9], data embedding algorithms for embedding data into 3D models have been published.

We have published methods for embedding data into 3D model shape [16, 17, 18]. Out algorithms mainly targeted shapes defined by using 3D polygonal meshes. Some of our algorithms embedded data by modifying the vertex coordinates of meshes. By using geometrical transformation-invariant quantities, such as the ratios of the volumes of tetrahedrons, their watermarks are robust against some of the operations to which 3D models are routinely subjected, such as, affine transformations. Other algorithms embedded data by modifying the topology, — that is, the connectivity — of vertices or triangles, which made the watermarks robust against arbitrary geometrical transformation. We have also proposed methods for embedding data in attributes associated with shape, such as per-vertex texture coordinates [18].

Kanai et al. [10] described an algorithm that employs multiresolution wavelet decomposition of 3D polygonal mesh models for data embedding. Their watermark withstands affine transformation, and is fairly robust against random noise added to vertex coordinates. An example of fragile watermarking (this term will be explained below) for a 3D model is described in [24], which is a 3D version of the approach they have proposed previously for 2D image watermarking. Benedens [2] described a watermarking algorithm that employs a set of normal vectors derived from geometric shape of a 3D model for embedding.

Hartung et al. [7] have developed a method for embedding data in MPEG-4 facial animation parameter (FAP) sequences by using a spread-spectrum technique. Remarkably, their watermarks could be extracted from a rendered movie sequence of 2D images. To extract watermarks, they applied their facial feature tracking system being developed to generate FAP sequences from video sequences of real human faces.

These existing data embedding algorithms are not

[1]This work was conducted while the first author was a research staff at IBM Tokyo Research Laboratory.

suitable, however, for most applications of computer-aided design (CAD). There are several reasons why this is so.

First, the majority of CAD models do not employ polygonal meshes to define shape. They employ parametric curves and surfaces, such as Bezier or non-uniform rational B-spline (NURBS) curves and surfaces. Consequently, previous data embedding algorithms that targeted shapes defined by polygonal meshes cannot be applied to these parametric curves and surfaces without modifications.

Second, CAD models rarely tolerate changes in geometry and/or topology that would be introduced by existing data embedding algorithms. A design for a combustion engine cylinder that is deformed due to modifications of NURBS control point coordinates will not be accepted. Changes in the topology of geometric primitives will become a problem, for example, in the case of finite element analysis, in which preservation of the connectivity of elements is essential.

In this paper, we propose data embedding methods for parametric curves and surfaces. The contributions of this paper can be summarized as follows:

- We present an algorithm that embeds data in NURBS curves and surfaces by using reparameterization. The algorithm preserves the exact geometric shapes of given curves and surfaces. In addition, data sizes of the curves and surfaces remain unchanged after data embedding.
- We outlines various alternative approaches for embedding data in parametric curves and surfaces that are not limited to NURBS curves and surfaces. These approaches are classified according to two properties, namely, preservation of exact shapes and preservation of model data sizes.

The rest of this paper is organized as follows. In the remaining part of this section, we briefly review the concept of data embedding in general. In Section II, we describe an algorithm that embeds data in NURBS curves and surfaces while maintaining their exact shapes and data sizes. In Section III, we discuss the implementation of the algorithm and experimental results obtained by using it. In Section IV, we suggest possible alternatives to data embedding into parametric curves and surfaces. We conclude in Section V with a summary and comments on future work.

1.1. Data embedding

In this paper, we call the act of adding a watermark (data) *embedding* or *watermarking*, and that of retrieving the information encoded in the watermark for perusal *extraction*. An object into which information is embedded is a called *cover-*<*datatype*>, an object with an embedded watermark is called a *stego-*<*datatype*>, and the information embedded is called *embedded-*<*datatype*> [22]. The suffix "<datatype>" varies according to the data type, such as image, text, or 3D model. For example, an embedded-text is embedded in a cover-NURBS surface to result in a stego-NURBS surface with an embedded-text.

Traditionally, watermarks have been classified by their *visibility* (or, more generally, *perceptibility*) and *robustness*. A *visible watermark* is made intentionally visible to serve various purposes, such as to deter a third party from unauthorized sales of contents. On the other hand, an *invisible watermark* is imperceptible without mechanical processing. A *robust watermark* should be able to withstand both intentional and unintentional modifications of the watermarked content. A *fragile watermark*, on the other hand, must be altered by intentional (and some unintentional) modifications, so that it can be used to detect tampering with or damage to the content. Here, *unintentional modifications* are the kind that can be expected to affect a content should during the course of its intended use, while *intentional modifications* are the kind that are applied with an intention of modifying or destroying the watermark**.**

The above classification according to the perceptibility of watermarks assumes that the observers of the cover data are humans. In the case of 3D models that are to be processed further by 3D CAD systems, viewing is more indirect. For example, only a milled machine part produced by using the 3D CAD model may actually be visible.

Watermarks can also be classified according to their use of cover data for extraction. If an extraction algorithm requires the original cover data as well as the stego-data, the scheme is called *private watermarking*. Otherwise, the scheme is called *public watermarking*. Embedding schemes by Cox et al [4] or Hartung [7] are examples of private watermarking.

A watermarking scheme may employ a random sequence generator to make an embedded message secure from being read by a third party. For example, in watermarking of an image, the positions of pixels to be modified for watermarks can be scrambled by a pseudo-random sequence generated from a stego-key (or stego-keys) by using a public-key cryptographic method. Scrambling of modulation values can also reduce statistical signatures in order to make watermarking less detectable. At the same time, scrambling and spreading of modifications for data embedding could make the watermark more robust against interference, in a manner similar to spread-spectrum communication. Both public-key cryptography and shared-key (or private-key) cryptographic method [13] can be used for this purpose.

Note that digital watermarking technology alone is not enough to realize applications of the technology. An infrastructure, such as an organization to issue verifiable digital signatures or a trusted place to escrow original models (i.e., models without watermarks) are essential. Furthermore, as in the case with other security and rights related technology, digital watermarking requires social and legal acceptance. The latter is probably more critical to the success of digital watermarking, digital signature and other security-related technology.

2. Data embedding in NURBS curves and surfaces by using reparameterization

This section presents a private data embedding algorithm for NURBS curves and surfaces that preserves the exact geometric shapes as well as the data sizes of models. As a private data embedding scheme, the algorithm requires both the original data and watermarked data for extraction (Figure 1).

The fundamental idea behind this shape-preserving algorithm is that a NURBS curve or surface can be reparameterized without changing its geometric shape. We will describe details of the algorithm with reference to NURBS curves. However, as we will explain later, the algorithms can easily be extended to NURBS tensor product surfaces. This section defines a NURBS curve and its reparameterization, then presents a data embedding algorithm that employs rational linear reparameterization.

A shape-preserving data embedding algorithm is required by most CAD applications, as explained in the previous section. For example, if a model is deformed even slightly as a result of watermarking, a constructive solid geometry operation using the model will yield an erroneous result.

An algorithm that preserves data size as well as shape is generally preferable, to contain communication and storage costs. In the case of NURBS curves and surfaces, the data sizes of models are determined mostly by the number of control points (including weights) and knots. Note that, in our paper, we consider a model's data size to be preserved if the numbers of control points and knots are unchanged. However, the exact number of bits of a compressed model after an ideal entropy coding may change after watermarking by using the data embedding algorithm.

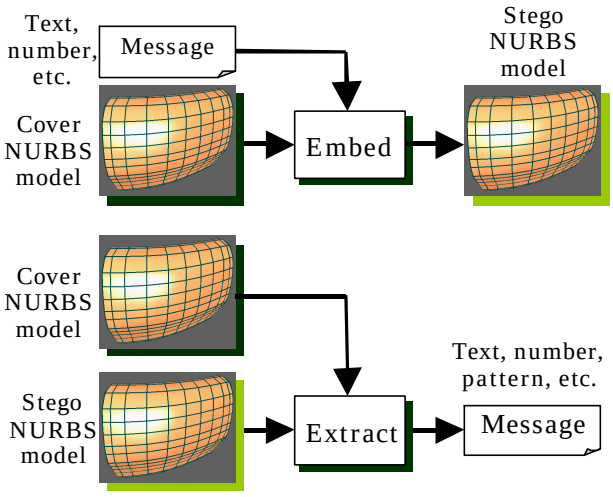


Figure 1. Flow of data in our private data embedding algorithm for NURBS curves and surfaces, which requires original cover-NURBS models for extracting embedded messages.

2.1. NURBS curve

In preparation for the explanation of our data embedding algorithm, this section defines a NURBS curve. Detailed explanations of NURBS and other parametric curves and surfaces can be found in such books as Farin [5] or Piegl and Tiller [19]. This paper follows the notations of Piegl and Tiller.

A pth-degree NURBS curve $\mathbf{C}(u)$ defines a point that traces a trajectory in 3D space as the scalar parameter value u varies within the range $[a,b]$.

$$\mathbf{C}(u)=\frac{\sum_{i=0}^{n} N_{i,p}(u)w_i\mathbf{P}_i}{\sum_{i=0}^{n} N_{i,p}(u)w_i} \quad u\in[a,b], \tag{1}$$

where a set of control points $\{\mathbf{P}_i\}$ forms a control polygon and $\{w_i\}$ are the weights. An increase in the weight w_i pulls the line closer to the control point $\mathbf{P}_i$. $N_{i,p}(u)$ is the ith B-spline basis function of degree p (order $p+1$), defined recursively as

$$N_{i,o}(u)=\begin{cases}1 & \text{if } u_i\le u<u_{i+1}\\ 0 & \text{otherwise}\end{cases}$$

$$N_{i,p}(u)=\frac{u-u_i}{u_{i+p}-u_i}N_{i,p-1}(u)+\frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}}N_{i+1,p-1}(u). \tag{2}$$

A non-periodic and non-uniform knot vector, which is a nondecreasing sequence of real numbers, is defined as

$$U=\{\underbrace{a,...,a}_{p+1},u_{p+1},...,u_{m-p-1},\underbrace{b,...,b}_{p+1}\} \tag{3}$$

where $a\le u_i\le u_{i+1}\le b$ and $i=0,...,m-1$. Knots in a NURBS curve are the points (in parameter space) where rational polynomial curves are grafted together to form a multi-segment curve.

2.2. Reparameterization of NURBS curve

A NURBS curve $\mathbf{C}(u)=\{x(u),y(u),z(u)\}$ defined on $u\in[a,b]$ is reparameterized by a function $u=f(s)$ so that the curve is computed as a function of a new parameter s instead of the original parameter u. We require that the function $f(s)$ be increasing ($f'(s)>0$ for all $s\in[c,d]$ in which $a=f(c)$ and $b=f(d)$) so that the same point x is not traced more than once. Detailes on the reparameterization of NURBS curves and surfaces can be found, for example, in [23] and [19].

Reparameterization of a parametric curve can be performed by using a large class of scalar function that satisfies the condition stated above. Obviously, a NURBS curve reparameterized by using an arbitrary function is in general not a NURBS. If a reparameterization function $u=f(s)$ is a polynomial of degree greater than one, the

resulting function $\mathbf{C}(s)$ is a NURBS but it will have a raised degree, requiring larger numbers of knots and control points. Data size is not preserved if such reparameterization is employed for data embedding, although model's geometric shape can be preserved.

We chose a rational linear function to reparameterize NUBRS curves for data embedding. Reparameterization by using a rational linear function maintains not only the exact geometric shape of the curve but also its data size, since the degree of the NURBS curve is not altered. Reparameterization by using a rational-linear (alias linear fractional, bilinear, or Mobius) function has been studied by Lee and Lucian in [11]. We use their results, which are summarized below.

A rational-linear function $g(u)$ is defined as follows.

$$s = g(u) = \frac{au + b}{gu + d} \tag{4}$$

$$u = f(s) = \frac{-ds + b}{gs - a} \quad s \in [c, d] \tag{5}$$

where $f(s)$ is the inverse of $g(u)$.
We let

$$m(u) = gu + d, \quad l(s) = gs - a \tag{6}$$

To ensure that $g(u)$ and $f(s)$ are well-behaved, we assume that

$$\begin{aligned} &ad - gb > 0, \\ &m(u) \neq 0 \quad \text{for all } u \in [a, b] \text{ , and} \\ &l(s) \neq 0 \quad \text{for all } s \in [c, d] \end{aligned} \tag{7}$$

Then, the reparameterized curve $\mathbf{C}(s)$ is obtained as follows:

- The control points $\{\mathbf{P}_i\}$ remain the same.
- The new knots S_i are the image under $g(u)$ of the original knots u_i, $s_i = g(u_i)$.
- The new weights $\{\overline{w}_i\}$ (modulo a common nonzero factor) are obtained from either Eq. (8) or Eq. (9):

$$\overline{w}_i = w_i \prod_{j=1}^{p} l(s_{i+j}) \tag{8}$$

$$\overline{w}_i = \frac{w_i}{\prod_{j=1}^{p} m(u_{i+j})}. \tag{9}$$

where s_{i+j} and u_{i+j} are the new and old knots, respectively.

2.3. A NURBS curve as an embedding primitive

In this paper, we call a minimum unit of modification for data embedding an *embedding primitive*. A rational linear reparameterization can be applied to a NURBS curve in order to use the latter as an embedding primitive.

Data can be embedded in the rational linear reparameterization function $g(u)$ by manipulating the coefficients a, b, g, and d. Note, however, that the numbers of degrees of freedom of $g(u)$ is actually three, not four, which is obvious when the function is rewritten as follows:

$$s = g(u) = \frac{au + b}{gu + d} = \frac{\frac{a}{g}u + \frac{b}{g}}{u + \frac{d}{g}} = \frac{k_1 u + k_2}{u + k_3}, \tag{10}$$

where $k_1 = a/g$, $k_2 = b/g$ and $k_3 = d/g$.

We find coefficients k_1, k_2, and k_3 for encoding data, such as numbers. Coefficients k_1, k_2, and k_3 are computed by specifying three points (u_1, s_1), (u_2, s_2), and (u_3, s_3) through which the function $g(u)$ must pass (Figure 2). Substituting the three points in (4) and solving for k_1, k_2, and k_3 yields the following formula:

$$\begin{aligned} k_1 &= \frac{(u_1 s_1 - u_2 s_2)(s_1 - s_3) - (u_1 s_1 - u_3 s_3)(s_1 - s_2)}{(u_1 - u_2)(s_1 - s_3) - (u_1 - u_3)(s_1 - s_2)} \\ k_2 &= u_1 s_1 + k_3 s_1 - k_1 u_1 \\ k_3 &= \frac{(u_1 s_1 - u_3 s_3)(u_1 - u_2) - (u_1 s_1 - u_2 s_2)(u_1 - u_3)}{(u_1 - u_3)(s_1 - s_2) - (u_1 - u_2)(s_1 - s_3)} \end{aligned} \tag{11}$$

While all three degrees of freedom may be manipulated to encode data, here, we constrain the two endpoints so that $u_1 = s_1 = a$ and $u_3 = s_3 = b$. This way, the range of parameters remains unchanged after the reparameterization. We encode data in the remaining degree of freedom, by setting the offset $D = s_2 - u_2$. The larger the magnitude of D, the more the curve of the function $s = g(u)$ deviates from the straight line $s = u$.

We have to pay attention to the amount of this offset, for several reasons. With a large value of D, parameterization can be skewed, or "bad." (If the parameterization of a curve is "good," a set of points on a curve is geometrically evenly spaced, given a set of uniformly spaced parameter values.) We should keep *D* relatively small so that the curves before and after data embedding have similar parameterization. With such small *D*, a curve with a good parameterization should also have a good parameterization after data embedding. On the other hand, if *D* is too small, it will be prone to numerical errors. We control the magnitude of D as a parameter of embedding.

Our experimental implementation employed a simple method of encoding a number into the offset D. Given the message data *d* of size *L* bits and the predefined range of the offset $[D_{\min}, D_{\max}]$, the offset *D* is computed by using the following formula:

$$D = \frac{(D_{\max} - D_{\min})(d + 0.5)}{2^L} + D_{\min} \tag{12}$$

This *D* is used to offset an *interior* knot of the NURBS

curve. A NURBS curve with not interior knot is a rational Bezier curve. For the embedding method to work, the curve must be a NURBS curve, that is, it must have at least one interior knot. We used the knot in the middle of the knot vector whose index i in the knot vector is computed from $i = \lfloor m/2 \rfloor$.

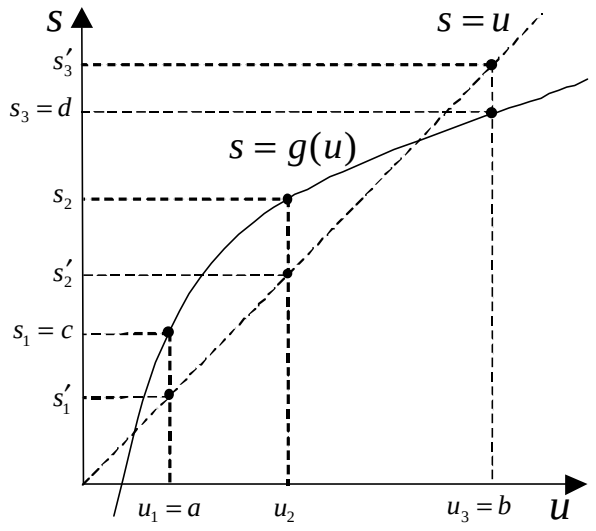


Figure 2. Determining a rational-linear function, that has three degrees of freedom, for reparameterization.

Extraction of a watermark starts with a comparison of the value of the *i*th knots in the knot vectors of the cover-model (original model) and the stego-model (watermarked model) to find the offset $D = s_2 - u_2$. From this value, the original data d can be recovered by using the above formula (12).

It would be preferable to randomly scramble a message as it is embedded, so that the embedded bits do not show telltale patterns in the stego-data. In addition, all the curves and surfaces in the model should be reparameterized by using a random sequence so that curves and surface patches modified for embedding can not be distinguished easily from those without embedding. However, neither of these features was implemented in our proof-of-concept system.

2.4. A tensor-product NURBS surface as an embedding primitive

Up to this point, our explanation of the embedding algorithm has assumed that NURBS curves are used as its embedding primitives. It is trivial to extend the method to include a NURBS surface created as the tensor product of two NURBS curves as another kind of embedding primitive.

Data embedding due to reparameterization of a NURBS surface parameterized by u and v can be accomplished by applying reparameterization as described in the previous section twice. Two offset values D_u and D_v computed from two sets of data to be embedded are used to reparameterize each of the two NURBS curves used to form the surface. The offsets D_u and D_v shifts interior knots at know positions in the two knot vectors for the u and v parameters. Other knots in the knot vectors and weights of the control points are adjusted accordingly so that the shape of the surface remains unchanged.

If a NURBS curve is able to store L bit, a tensor product NURBS surface is able to store twice that amount.

2.5. Ordering embedding primitives

In order to embed a significant amount of information, e.g., tens to hundreds of bytes, multiple embedding primitives — that is, NURBS curves and surfaces — must be ordered. Such ordering can be achieved by using the topology of objects in CAD models for example, by creating a spanning tree of surface patches and traversing the tree in a depth-first order. Furthermore, objects in a CAD model are often numbered sequentially so that various properties (e.g., color and material) can be associated with them. Such sequential identification numbers can also be employed to order embedding primitives.

Upon embedding, a mapping from a bit string in a message to an ordered set of embedding primitives can be scrambled by using a pseudo-random number sequence as mentioned before. Such scrambling could make the message secure from third parties, and if combined with repeated embedding, could make the embedding more robust against various disturbances.

3. Experiments and Results

We have implemented the shape- and size-preserving data embedding algorithm for the NURBS curves and surfaces described in the previous section. Our current implementation of the algorithm embedded L bits per NURBS curve and $2L$ bits per NURBS surface by using a rational linear reparameterization. Ordering of multiple NURBS curves and surfaces is done by assuming that each curve or surface is explicitly and uniquely numbered.

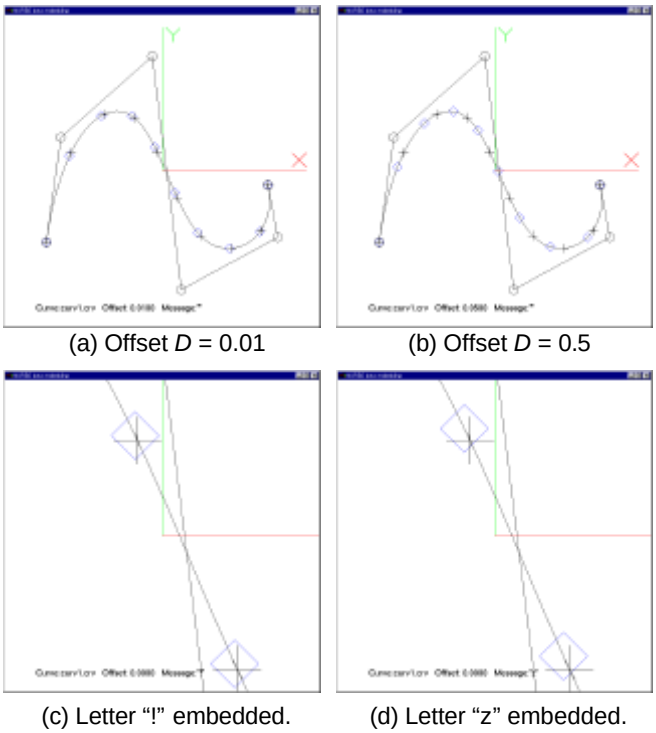
The code was written in C++, using the OpenGL graphics API. It was written as a window-system independent code by using the GLUT toolkit and the GLUI user-interface toolkit, so that it runs on both X-window/UNIX based systems and on Windows NT/95 systems without modification.

Figures 3 (a)–(d) show the effects of reparameterization of a third-degree NURBS curve with one interior knot by using a rational linear function with varying offset D. The straight line segments are the control polygon, in which the control points are marked with circles. The curve is marked with 10 markers, which correspond to 10 uniformly spaced parameter values in the range $[a, b]$. For both the original and the reparameterized curves, parameterizations are indicated, respectively, by cross and rectangular markers.

A curve reparameterized by using a smaller offset value $D = 0.01$ showed parameterization similar to the original curve. However, a curve reparameterized with a larger

offset, such as $D = 0.5$, showed a clearly skewed parameterization. The range of the offset D used for embedding needs be controlled so that the resulting reparameterization is not too skewed.

Figure 4 shows the NURBS curve with a $L = 8$-bit message embedded. The 8-bit message was embedded in a double precision (64 bit) floating-point number that is used to represent the knot value internally. While 8 bits are not very useful on their own, a set of such embedding primitives can be ordered by using an ordering method to embed a larger, useful amount of data. Figure 5 shows an example of reparameterizing a NURBS tensor product surface, which has degree 3 in both u and v. The offsets used for the reparameterization are $D_u = 0.001$ and $D_v = 0.02$ for the parameters u and v, respectively. Figure 6 shows an example of 16-bit data in the form of two characters "aZ" embedded into the same NURBS suaface. The amplitude of the offset D is set intentionally high, at $[D_{min}, D_{max}]=[0.001, 0.01]$, to make the close-up reveal the effects of the reparameterization. With the $L = 8$, D_{max} can be set much lower without compromising the stability of data embedding.



(a) Offset D = 0.01 (b) Offset D = 0.5

(c) Letter "!" embedded. (d) Letter "z" embedded.

Figure 3. Examples of reparameterization of a NURBS curve with offsets (a) D = 0.01, (b) D = 0. 5. In these figures, crosses mark the original and the rectangles mark the reparameterized curve at ten equally spaced parameter values. In figures (c) and (d), letters "!" and "z" in ASCII code are embedded.

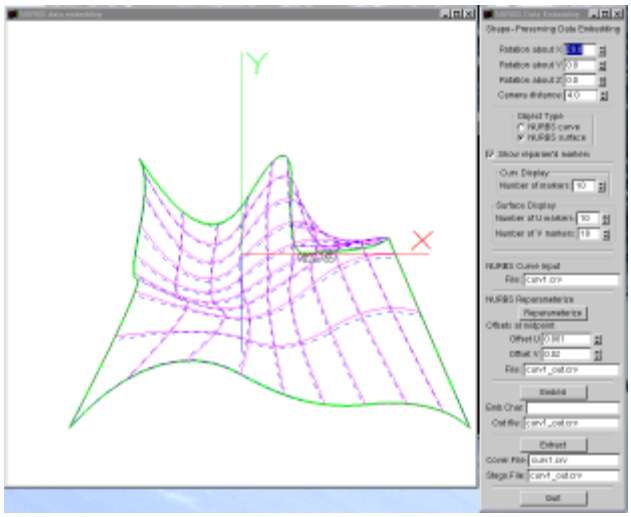
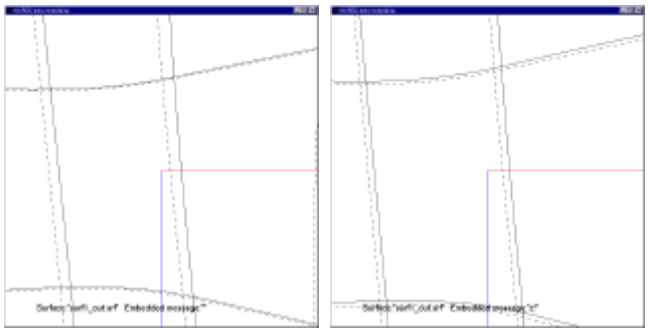


Figure 4. A NURBS surface of degree 3 in both u and v is reparameterized with $D_u = 0.001$, $D_v = 0.02$, shown with a simple user interface.



(a) Reaprameterized with $D_u = 0.01$, $D_v = 0.001$. (b) A 2-byte string "aZ" was embedded in the surface.

Figure 5. Close-up of the surface of Figure 4. In (a), the surface is reparameterized with $D_u = 0.01$, $D_v = 0.001$. In figure (b), a 2-byte string "aZ" was embedded in the surface.

4. Alternative data embedding approaches for parametric curves and surfaces

In this section, we suggest various possible approaches to embedding data in parametric curves and surfaces that include Bezier, rational Bezier, B-spline, and other curves and surfaces, as well as NURBS curves and surfaces.

We classify the data embedding approaches outlined in this section by using two properties of embedding methods: preservation of model shape and preservation of model data size. A *shape-preserving* method preserves the exact shape of an original model, while a *shape-altering* method alters it. A *size-preserving* method retains the original data size after embedding, while a *size-altering* method changes it. These classifications are orthogonal.

As we discussed previously, shape-preserving data embedding is desired for most CAD applications. Size-preserving methods preserve the amounts of data occupied by parameters that define curves or surfaces, such as control points and knot vector elements. By "data size preserving," we mean that the numbers of these parameters remain unchanged. Their exact values, and hence the exact number of bits of the model after an ideal entropy coding, may change. The size-preservation property is preferable, for example, for communication and/or storage efficiency.

Please note, in the following, that a discussion relating to a type of curve can also be applied to a surface generated as a tensor product of curves of that type.

4.1. Shape-preserving, size-altering methods

Data embedding that preserves the shape, but alters a model's data size can be realized by injecting redundancy into representations of parametric curves and surfaces. Such redundancy injection can be achieved by using such techniques as knot insertion, degree elevation, and reparameterization which involves degree elevation.

- **Knot Insertion:** For those parametric curves with multiple spans, such as nonrational B-spline curves and NURBS curves, new knots can be inserted into the curve. The values of the inserted knot, the mere presence of the new knot, or the location of the new knot in the knots vector could encode information to be embedded.
- **Degree Elevation:** For nonrational curves, such as Bezier and B-spline curves, elevating the degree of a curve introduces new control points. For example, the amount of the increase in degree or the locations of new control points could encode information.
- **Reparameterization Involving Degree Elevation:** Reparameterization by using a polynomial or rational polynomial of a degree greater than one can be applied to rational parametric curves, such as rational Bezier and NURBS curves. Such reparameterization will raise the degree of the curve, introducing new control points and knots, and increasing the data size.

If we assume shape preservation, finding and removing knots and/or control points inserted for data embedding from those that are in the original model can be quite difficult. Thus, data embedding methods suggested in this subsection could be more robust than the method we have described in Section 2 that employs reparameterization. Note, however, that the increase in data size due to embedding can be a significant disadvantage for some of the applications.

4.2. Shape-altering, size-preserving methods

If exact shape preservation is not an issue, additional approaches exist for embedding information into parametric curves and surfaces.

- **Modulation of Control Points and/or Weights:** The control points for a curve (tensor-product surface) form a one-dimensional (two-dimensional) ordered set of 3D or 4D points. These values can be modulated, as if they were simply regularly ordered sets of floating point values. Modulation of control points is applicable to a large class of parametric curves and surfaces, including Bezier, rational Bezier, B-spline, and NURBS curves and surfaces.
- **Knot Vector Modulation:** The values of knots in a knot vector, which is an ordered 1D sequence of scalar values, can be modified to encode information. This method is applicable to parametric curves with knots, such as B-spline and NURBS curves and surfaces.

Note that the modulation of values, such as control point coordinates, weights, and knot values may be performed either in their original domain or in a transformed domain. For example, a watermarking algorithm that modifies control point coordinates for the embedding can be combined with a lossy compression algorithm for curved surfaces that employs discrete cosine transformation for its compression (e.g., [12]).

If shape alterations are tolerated, approaches suggested in this section could be used as basis for robust data embedding algorithms.

5. Summary and future work

This paper has presented an algorithm that embeds data in rational parametric curves and surfaces, particularly, NURBS curves and surfaces, by using reparameterization. We employed rational linear reparameterization, since it enables data embedding that preserves the exact geometric shapes of the NURBS curves and surfaces as well as their data sizes.

This paper also suggested alternative approaches for embedding data in parametric curves and surfaces that are not limited to NURBS. These approaches are classified according to their shape-preservation and size-preservation properties. Knot insertion and degree elevation are among possible alternative data-embedding methods that preserve shape but alter data size.

In the future, we would like to incorporate linear rational B-spline reparameterization function into our implementation so that the data density is improved. We also would like to implement the alternative data-embedding approaches outlined in outlined in Section IV.

References

[1] W. Bender, D. Gruhl, and N. Morimoto, Techniques for Data Embedding, IBM Systems Journal, Vol. 35, Nos. 3 & 4, 1996.

[2] O. Benedens, Geometry-Based Watermarking of 3D Models, *IEEE CG&A*, pp. 46-55, January/February 1999.

[3] G. Braudway, K. Magerlein, and F. Mintzer, Protecting Publicly-Available Images with a Visible Image Watermark,

IBM Research Report, TC-20336 (89918), January 15, 1996.

[4] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, Secure Spread Spectrum Watermarking for Multimedia, *IEEE Trans. on Image Processing*, Vol. 6, No. 12, pp. 1673-1678, 1997.

[5] G. E. Farin, *Curves and Surfaces for Computer-Aided Geometric Design, A Practical Guide*, Fourth Edition, Academic Press, San Diego, CA, 1997.

[6] F. Hartung and B. Girod, Copyright Protection in Video Delivery Networks by Watermarking of Pre-compressed Video, *Lecture Notes in Computer Science*, Vol. 1242, pp. 423-436, Springer, 1997.

[7] F. Hartung, P. Eisert, and B. Girod, Digital Watermarking of MPEG-4 Facial Animation Parameters, *Computer and Graphics*, Vol. 22, No. 4, pp. 425-435, Elsevier, 1998.

[8] ISO/IEC 14772-1 Virtual Reality Model Language (VRML).

[9] ISO/IEC JTC1/SC29/WG11 *MPEG-4 Visual* and *MPEG 4 SNHC*.

[10] S. Kanai, H. Date, and T. Kishinami, Digital Watermarking for 3D Polygons using Multiresolution Wavelet Decomposition, Proc. of the *Sixth IFIP WG 5.2 International Workshop on Geometric Modeling: Fundamentals and Applications (GEO-6)*, pp. 296-307, Tokyo, Japan, December 1998.

[11] E. T. Y. Lee, and M. L. Lucian, Mobius Reparameterizations of Rational B-splines, Computer Aided Geometric Design, Vol. 8, pp. 213-215, Elsevier, 1991.

[12] H. Masuda, R. Ohbuchi, and M. Aono, Compression and Progressive Transmission of Parametric Surfaces, *Transactions of the Information Processing Society of Japan*, Vol. 40, No. 7, pp. 1188-1195, 1999 (in Japanese).

[13] A. J. Menezes, P. C. van Oorshot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[14] F. Mintzer, G. W. Braudway, and M. M. Yeung, Effective and Ineffective Digital Watermarks, Proceedings of the *IEEE International Conference on Image Processing (ICIP) '97*, Vol. 3, pp. 9-12, 1997.

[15] J. J. K. O'Ruanaidh, W. J. Dowling and F. M. Boland, Watermarking Digital Images for Copyright Protection, *IEE Proc.-Vis. Image Signal Process.*, Vol. 143, No. 4, pp. 250-256, August 1996.

[16] R. Ohbuchi, H. Masuda, and M. Aono, Watermarking Three-Dimensional Polygonal Models, *Proceedings of the ACM Multimedia '97*, Seattle, Washington, USA, November 1997, pp. 261-272.

[17] R. Ohbuchi, H. Masuda, and M. Aono, Watermarking Three-Dimensional Polygonal Models Through Geometric and Topological Modifications, pp. 551-560, *IEEE Journal on Selected Areas in Communications*, May 1998.

[18] R. Ohbuchi, H. Masuda, and M. Aono, Geometrical and Non-geometrical Targets for Data Embedding in Three-Dimensional Polygonal Models, *Computer Communications*, Vol. 21, pp. 1344-1354, Elsevier (1998).

[19] L. Piegl, W. Tiller, *The NURBS Book*, 2nd Edition, Springer, Berlin, 1997.

[20] B. Pfitzmann, Information Hiding Terminology, in R. Anderson, Ed., *Lecture Notes in Computer Science* No. 1174, pp. 347-350, Springer, 1996.

[21] J. R. Smith and B. O. Comiskey, Modulation and Information Hiding in Images, in R. Anderson, Ed., *Lecture Notes in Computer Science* No. 1174, pp. 207-296, Springer, 1996.

[22] K. Tanaka, Y. Nakamura, and K. Matsui, Embedding Secret Information into a Dithered Multilevel Image, *Proc. 1990 IEEE Military Communications Conference*, pp. 216-220, 1990.

[23] W. Tiller, Rational B-splines for curve and surface representation, *IEEE Computer Graphics and Applications*, Vol. 3, No.6, pp. 61-69, 1983.

[24] B-L. Yeo and M. M. Yeung, Watermarking 3D Objects for Verification, *IEEE CG&A*, pp. 36-45, January/February 1999.

[25] M. M. Yeung, F. C. Mintzer, G. Braudway, and A. R. Rao, Digital Watermarking for High-Quality Imaging, Proceedings of the *First IEEE Workshop on Multimedia Signal Processing*, Princeton, NJ, USA, June 1997, pp. 357-362.

[26] J. Zhao and E. Koch, Embedding Robust Labels into Images for Copyright Protection, Proc. of *the Int'l. Congress on Intellectual Property Rights for Specialized Information, Knowledge, and New Technologies*, Vienna, August 1995.