Managing CAD Data as a Multimedia Data Type Using Digital Watermarking

[1]Ryutarou Ohbuchi, [2]Hiroshi Masuda
[1]Yamanashi University, Computer Science Department (ohbuchi@acm.org)
[2]Tokyo University, Artifacts Research Center (masuda@race. u-tokyo.ac.jp)

Key words: Computer-Aided Design (CAD), information security, digital watermarking, parametric surfaces, reparameterization, knot insertion

Abstract: Three-dimensional (3D) model, including 3D geometric CAD model, has come to be recognized as a "multimedia" data type, along with such data types as sound, text, still image, and movie. As a data type, each one of these data types has a set of operations associated with it. Representative of operations are input, output, editing, indexing, hyper-linking, compression, similarity-based search, and some form of security, such as authentication, tamper detection, and intellectual property (IP) management.

This paper discusses digital watermarking technology for 3D geometric CAD data, which may offer a solution to such security related issues of authentication, tamper detection, and intellectual property management. This paper first examines 3D CAD data as a multimedia data type, followed by discussion on its watermarking opportunities. The paper then presents multiple approaches to watermark mechanical CAD models defined by using parametric curves and surfaces. In particular, two methods that targets NURBS curves and surfaces are presented. The methods, one based on reparameterization and the other based on knot insertion, preserves exact geometric shape of target NURBS objects.

1. INTRODUCTION

Membership of "multimedia" data type has long been limited to sound, text, still image, and movie data. Recently, however, three-dimensional (3D) object is increasingly regarded as a full-fledged multimedia data type. Ever increasing computational and graphics capability of personal computers, as well as such graphics data exchange standards as the VRML [VRML] and MPEG-4 SNHC [MPEG4] are behind this trend.

A 3D geometric Computer-Aided Design (CAD) model can also be viewed as a multimedia data object, although this may not be a common perception. A 3D CAD model is a multimedia data object, however, if one considers such scenarios as design documentation, design reuse, co-design, integrated design and manufacturing, marketing, and maintenance, for example. These scenarios, all of which can be prefixed with "network-based" or "Internet-based", demand that we treat 3D geometric CAD model as a member of the multimedia media data types.

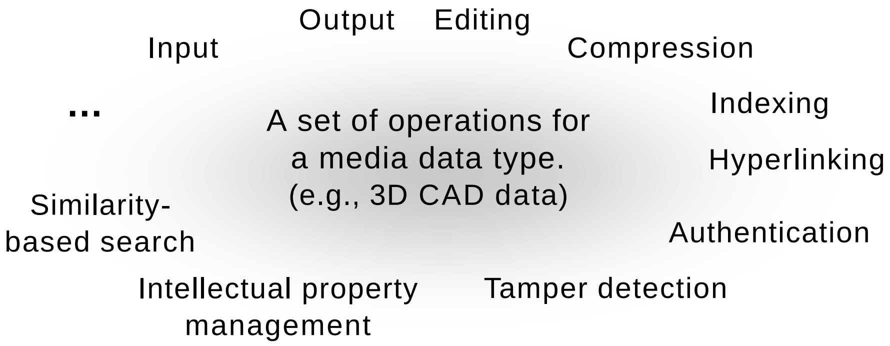


Figure 1. Examples of operations associated with a multimedia data type, including 3D CAD data.

As an abstract data type, a media data type is associated with its own set of operations (Figure 1). A representative set of operations includes *input*, *output*, *editing*, *compression*, *indexing*, *hyperlinking*, *similarity* (or *content*) *-based search*, *authentication*, and *intellectual property (IP) management*. The first two of the operations are the minimal necessary for a media data type, but the third, editing, is often necessary as well. The rest of the operations gains importance when the data object is shared, distributed, stored, or broadcast, e.g., via the Internet.

In case of the text data type, technologies for many of these operations are fairly well understood and available. Social acceptance of and legal infrastructure necessary for some of these operations, e.g., IP protection, have long been established. For other data types, however, not all of the operations are well understood. For example, in the case of still image data, operations other than input, output, compression, and editing are not well developed. Similarity-based (or, content-based) search and copy protection of still images available on the Internet is still in the research and development stage. Time-dependency that exists in such media data types as audio and movie adds a degree of difficulty. Editing, hyperlinking, and indexing time-varying data embody many issues to be solved. Simply hyperlinking “from this to that” can be quite difficult if “this” and “that” must be specified both in time and space. One wants search for a scene in a movie in which “objects of these kind of shapes/colours/etc. move in this manner”. But what is a “scene”, and how can we specify objects and their movements?

1.1 3D model as a media data type

In case of 3D model, even such basic operations as input and output are still the subject of active research and development. Traditionally, editing and synthesis, rather than direct input (for example by capturing) has been the mainstay approach to creating 3D model shape, attributes (e.g., colour, texture coordinates, etc.), and motion. Methods to specify, represent, and modify shapes has been the focus of CAD research. A recent trend is to directly input or capture 3D models using such technologies as laser range-scanner and optical motion tracker. Converting millions of 3D points from the range scanner into useful shape representation leaves many subjects to be studied. In terms of 3D model output, visual display has been the primary method, but such additional visual cues as stereo and head-motion parallax have been added in some cases. Tactile and force display techniques have also been explored. An ultimate form of 3D output is to create actual 3D object by using a rapid prototyping technology or a numerically controlled (NC) milling machine.

Increased incidence of distribution and sharing of 3D models, especially through the Internet, created a whole new set for requirements for 3D models. Operations other than input, output, and editing have suddenly become important. Foremost in the list of newly important operation is the *compression* of 3D geometry and its attributes. Michael Deering started the topic in 1995 [Deering95], whose technology has been incorporated into the Sun Microsystems' Java3D [Java3D]. Compression of vertex connectivity in polygonal meshes is emphasized in the work by Taubin et al. [Taubin97]. Taubin's compression method has been included in the ISO standard MPEG-4 SNHC [MPEG4]. A good set of reference materials on the subject of 3D model compression can be found in [Rossignac99]. Hyperlinking and indexing have found uses in some CAD systems, e.g., to annotate and cross-reference designs and parts for documentation and collaboration.

For 3D models, the other operations, such as authentication, IP management, and similarity-based search have seen little work so far. These operations, however, are vital to net-based almost-everything. Nowadays it is common for several divisions within a company to collaborate via network over physical distances and time zones. Often, more than one company are involved in a design, exchanging design documents. In the future, a mechanical part may be purchased as a "virtual component", i.e., as a design, not as a finished product. Now imagine following scenarios. Is the CAD data you just received from your colleagues in a branch office (or a vendor) via the Internet genuine, or is it subtly tampered with so that your project fails to meet the deadline? If you are a vendor of a part, how do you prevent a thief disguised as a customer copying your design provided for pre-production "evaluation"? How do you detect, prove, and put forth a legal claim regarding such IP-violations?

This paper discusses techniques for digital watermarking 3D CAD models. Digital watermarking could provide (at least partial) solutions to such security-related issues as IP management, tamper-detection and authentication.

1.2 Digital Watermarking

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 uses (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. The act of adding a watermark is called (data) *embedding* or *watermarking*, and that of retrieving the information encoded in the watermark for perusal is called *extraction* (Figure 2). An object to be watermarked is called *cover <datatype>*, the object with watermark is called *stego <datatype>*. The suffix "<datatype>" specifies the data type of the content. For example, a message bits are embedded in a cover-NURBS surface to create a stego NURBS surface.

Arguably the most important property of a watermark is its transparency. Watermarks must be *transparent* to the intended applications. We distinguish two kinds of transparency, *functional* and *perceptual*. For most of the traditional data types, such as image and audio data, transparency of a watermark is to be judged by human beings. If the cover-data and stego-data are indistinguishable to human observers, the watermark is *perceptually transparent*. For other data types, such as 3D geometric CAD data, transparency of the watermark is judged by if *functionality* of the data is altered or not. A perceptually transparent watermark may or may not be functionally transparent. Likewise, a functionally transparent watermark may or may not be perceptually transparent. For example, a perceptually transparent watermark added to CAD data of an engine cylinder may alter the shape of the cylinder enough to interfere with the function of the engine.

Another very important classification of watermark is by *robustness*. A *robust watermark* should withstand both intentional and unintentional modifications of the stego-data. On the other hand, a *fragile watermark* must be affected by intentional (and

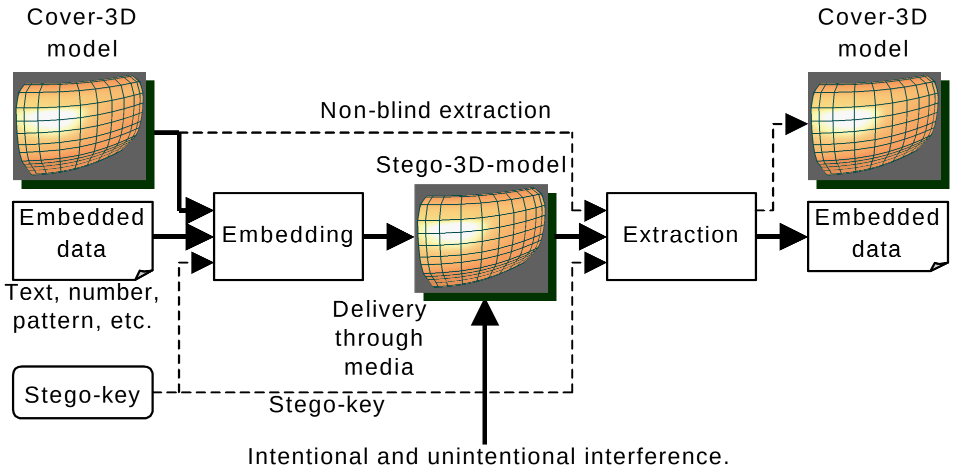


Figure 2. A representative flow of watermarking.

some unintentional) modifications so that tampering and other damage to the data can be detected. Here, *unintentional modifications* are applied to a data object during the course of its normal use, while *intentional modifications* are applied to the data object with an intention of modifying or destroying the watermark. A robust watermark can be used to claim copyright of illegally duplicated data, while a fragile watermark finds application in tamper detection, for example. Robustness and transparency of the watermark is often at odds; making a watermark more robust tent to make it less transparent.

Watermarks can also be classified according to their use of cover data for extraction. A *public* watermarking scheme extract message using stego-data only. Such an extraction is called *blind*-extraction or blind-detection. A *private* watermarking scheme requires original cover-data as well as the watermarked stego-data for its *non-blind* extraction of embedded message. While a private watermarking scheme with non-blind extraction enables more robust and accurate extraction (e.g., subtraction of the cover-data from the stego-data reveals the signal, that is, the watermark), a public scheme usually is easier to be adopted an application scenario.

A watermarking scheme may employ cryptographic approach to make embedded messages secure from a third party. For example, messages to be embedded can be ciphered prior to embedding by using a public-key cryptography [Menezes96]. A cryptographic function may be embedded into the watermarking process itself, for example, to scramble mapping from a message bit and corresponding watermark structures.

Note that, for the watermark to be effective, legal infrastructure, social acceptance, as well as technical infrastructures such as verifiable digital signature and trusted places to escrow cover 3D models are necessary.

1.3 Previous Work on Watermarking 3D Models

While there has been numerous studies on watermarking text, still images, audio, and movie data, relatively small number of published work exist on watermarking 3D models. We are the first to publish on the subject [Ohbuchi97, Ohbuchi98a, Ohbuchi98b]. Our first set of 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, these watermarks are robust against some of the operations to which 3D models are routinely subjected. For example, watermarks based on ratios of volumes of pairs of tetrahedrons are robust against affine transformation. Other algorithms of ours embedded data by modifying the topology, — that is, connectivity — of vertices or triangles, which made these watermarks robust against arbitrary geometrical transformation. We have also proposed methods to embed data into attributes associated with geometric shapes such as per-vertex texture coordinates [Ohbuch98b].

Kanai et al. [Kanai98] describes an algorithm that employs multiresolution wavelet decomposition of 3D polygonal meshes for watermarking. Their watermark withstands affine transformation, and is fairly robust against random noise added to vertex coordinates. Their method requires that the mesh to have 1-to-4 subdivision connectivity, however. Praun and Hoppe [Praun99] developed a robust watermarking algorithm based on a transformed-domain technique. Their watermark withstands similarity transformation and fairly robust against random noise added to vertex coordinates. However, their target mesh must be rather complex in terms of number of vertices.

Yeo, et al [Yeo99] describes a fragile watermarking method for 3D models, which is a 3D version of the approach they have proposed previously for 2D image watermarking. Benedens [Benedens99] described a watermarking algorithm that employs a set of normal vectors derived from geometric shape of a 3D model for embedding.

Hartung et al. [Hartung98] 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.

Existing watermarking algorithms that targets 3D *shapes* are not suitable, however, for watermark shapes found in most geometric CAD models. There are two majors reasons why this is so. First, all previous watermarking algorithms for 3D shapes targets shapes defined by using polygonal meshes. Most current CAD models use parametric curves and surfaces such as non-uniform rational B-spline (NURBS) curves and surfaces for shape definition. Thus, previous watermarking algorithms that targeted polygonal meshes cannot be applied readily to most of the CAD models. Second, all previous watermarking algorithms for 3D shapes alter shape geometry and/or topology to watermark, by assuming perceptual, not functional, transparency. However, CAD models rarely tolerate changes in geometry and/or topology. Watermarking algorithms that exactly preserves shape are required for most CAD data.

This paper discusses digital watermarking technology for 3D geometric CAD models consisting of parametric curves and surfaces with the objective of providing authentication, tamper-detection, IP protection and other security related operations to these CAD models. In particular, we present details of algorithms that employ knot insertion and reparameterization to watermark NURBS curves and surfaces. These methods preserve exact geometric shape of NURBS curves and surfaces.

2. WATERMARKING 3D GEOMETRIC CAD MODELS

Watermarking in general exploits redundancy in the watermarking target to encode extra information that is the watermark. There are three classes of redundancies that can be exploited to add structures to 3D geometric CAD data for watermarking.

- Innate redundancy: A shape itself may have redundancy. Information may be encoded in the shape by altering parts of the shape without interfering with the function of the shape. Locations and manners of shape modifications must be controlled carefully so that the watermark is functionally transparent.

In geometric CAD, a shape has function. However, shape in most 3D geometric CAD data has certain amount of arbitrariness that can be exploited to encode watermarks without affecting intended function of the shape. An obvious method that has been in use for years is to engrave manufacturer name and part number somewhere on a mechanical part.

- Representation redundancy: Representation of a shape may have redundancy. Information may be encoded for watermarking by modifying the representation while exactly preserving the shape.

  For example, knots can be inserted into a NURBS surface without changing its geometric shape. Once inserted, knots are very difficult to remove if exact preservation of the model's geometric shape is enforced.

- Encoding redundancy: Encoding of a shape representation has redundancy so that watermark information can be encoded by modifying the encoding. Geometric shape and its representation do not change.

  As a trivial example, assume that each control point coordinate value of a CAD model has accuracy of at most 6 decimal digits, while its data format has space for 10 fixed digits. Of 10 digits, 4 digits may be allocated to encode information for watermark.

This paper focuses on the second class of approach, which exploits redundancy that exists in shape representations. Of various representations, this paper focuses on parametric curves and surfaces as targets since they are the dominant shape representation in current geometric CAD systems.

3. WATERMARKING PARAMETRIC CURVES AND SURFACES

Alternative approaches for watermarking parametric curves and surfaces can be classified by their two properties: 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 does not. A *size-preserving* method preserves the original data size while a *size-altering* method does not. By "data size preserving," we mean that the numbers of shape-defining parameters such as control points and knot vector to 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 that most of the discussions below on a curve can also be applied to a surface generated as a tensor product of a pair of such curves.

3.1 A Shape-preserving, size-preserving method

Rational-linear reparameterization is the only one method we came up so far in this category, which preserves both shape and data size. A watermarking method based on this property was first presented in [Ohbuchi99].

3.1.1 Rational-Linear Reparameterization Based Algorithm

Reparameterization of NURBS curves and surfaces by using *rational linear function* exactly preserves their geometric shapes. It also preserves their data size, in terms of number knots, control points, and weights. We exploited these properties for a watermarking method for NURBS curves and surfaces that preserves exact shape and data size. The method employs non-blind extraction scheme so that the extraction requires original cover-NURBS curves or surfaces. The watermark embedded using this method is not robust; another reparameterization will destroy the watermark. Thus, this method will be useful primarily as a fragile watermark.

Let us first define a NURBS curve. A *p*th-degree NURBS curve $\mathbf{C}(u)$ defines a point that traces a trajectory in 3D space as the scalar parameter value u varies in the interval $[a,b]$. (For the details on NURBS curves and surfaces, see, for example, Piegl and Tiller [Piegl97].) A set of control points $\{\mathbf{P}_i\}$ forms a control polygon, and $\{w_i\}$ are the weights of the control points. Recursively defined function $N_{i,p}(u)$ is the *i*th B-spline basis function of degree p (order $p+1$). U is a non-periodic and non-uniform knot vector, which is a nondecreasing sequence of real numbers, that is, $a \le u_i \le u_{i+1} \le b$ for $i = 0,...,m-1$.

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

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

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

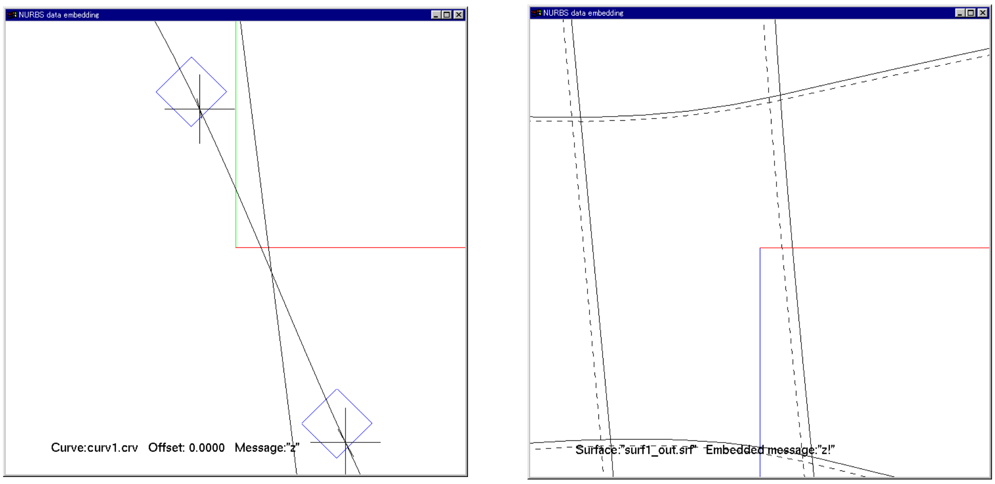
$$U = \{\underbrace{a,...,a}_{p+1}, u_{p+1},...,u_{m-p-1}, \underbrace{b,...,b}_{p+1}\} \tag{3}$$

A NURBS curve $\mathbf{C}(u) = \{x(u), y(u), z(u)\}$ defined on u can be reparameterized by a function $u = f(s)$ so that the curve is computed as a function of a new parameter s. 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. We reparameterize using rational linear function for its shape and data size preservation property. Such reparameterization has been studied by Lee and Lucian [Lee91].

There are four coefficients a , b , g , and d in $g(u)$. However, degrees-of-freedom (DOF) of $g(u)$ is actually three, which is clear by rewriting $k_1 = a/g$, $k_2 = b/g$ and $k_3 = d/g$.

$$s = g(u) = \frac{au+b}{gu+d} = \frac{k_1 u + k_2}{u + k_3} \tag{4}$$

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



Curve:curv1.crv Offset: 0.0000 Message:"z"

Surface:"surf1_out.srf" Embedded message:"z!"

(a) A letter "z" is embedded in a NURBS curve. Cross and rectangular markers are at the same parameter values but plotted before and after the watermarking.

(b) A 2-byte string "aZ" was embedded in a Cartesian product NURBS surface. Solid and dotted lines are iso-parametric lines before and after the watermarking.

Figure 4. NURBS curves and surfaces are watermarked by using reparameterization, which exactly preserves their geometric shapes.

$$\begin{aligned} & ad - gb > 0, \\ & m(u) = gu + d \neq 0 \quad \text{for all } u \in [a,b], \text{ and} \\ & l(s) = gs - a \neq 0 \quad \text{for all } s \in [c,d] \end{aligned} \tag{5}$$

The reparameterized curve $\mathbf{C}(s)$ has the same control points $\{\mathbf{P}_i\}$. The new knots s_i are the image $s_i = g(u_i)$ of the original knots u_i. The new weights $\{\overline{w}_i\}$ (modulo a common nonzero factor) are obtained from either one of the equations (6):

$$\overline{w}_i = w_i \prod_{j=1}^{p} l(s_{i+j}) \quad \text{or} \quad \overline{w}_i = \frac{w_i}{\prod_{j=1}^{p} m(u_{i+j})}. \tag{6}$$

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

The function $g(u)$, which has 3-DOF, can be specified by three points (u_1, s_1), (u_2, s_2), and (u_3, s_3) through which the function $g(u)$ must pass using the following formula.

$$\begin{aligned} k_1 &= \frac{(u_1 s_1 - u_2 s_2)(s_1 - s_3) - (u_1 s_1 - u_3 s_3)(s_1 - s_2)}{(u_1 - u_2)(s_1 - s_3) - (u_1 - u_3)(s_1 - s_2)} \\ k_2 &= u_1 s_1 + k_3 s_1 - k_1 u_1 \\ k_3 &= \frac{(u_1 s_1 - u_3 s_3)(u_1 - u_2) - (u_1 s_1 - u_2 s_2)(u_1 - u_3)}{(u_1 - u_3)(s_1 - s_2) - (u_1 - u_2)(s_1 - s_3)} \end{aligned} \tag{7}$$

We fix the two endpoints, that is, $u_1 = s_1 = a$ and $u_3 = s_3 = b$, so that the parameter domain is the same before and after the reparameterization. We encode data in the remaining DOF by choosing 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 keep the magnitude of D small so that the curve after reparameterization has a "good" parameterisation. We experimented with a simple amplitude modulation to encode 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:

$$D = \frac{(D_{\max} - D_{\min})(d + 0.5)}{2^L} + D_{\min} \tag{8}$$

This D is used to offset an *interior* knot of the NURBS curve. We used the knot in the middle of the knot vector whose index i in the knot vector is computed from $i = \lfloor m/2 \rfloor$.

Extraction of a watermark starts with a comparison of the value of the ith 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 formula (8).

Figure 4 shows examples of embedding data into a NURBS curve and a surface using the method described above.

3.2 Shape-preserving, size-altering methods

A watermarking algorithm that exactly preserves the model’s shape but alters its 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 that 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.

3.2.1 Knot Insertion Based Algorithm

Knot insertion into NURBS curves and surfaces can be used for a robust watermarking algorithm that exactly preserves shape. Watermarks added by the method are quite robust against attempted removal assuming that the intended application of the watermark demands exact shape preservation. This is because knots removal applied in an attempt to remove watermark will inevitably involves minute shape change, which is not tolerated. In addition to shape preservation and robustness, this method offer significantly larger data capacity than the reparameterization based algorithm described in Section 3.1.1, since multiple knots may be inserted in a NURBS curve.

Given a NURBS curve

$$\mathbf{C}^w(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P}_i^w \tag{9}$$

defined on a not vector $U = \{u_0,...,u_m\}$, a knot insertion adds a new knot to the knot vector so that geometric shape of the curve remains unchanged. Let a new knot value that is added to carry watermark message be $\bar{u} \in [u_k, u_{k+1})$. We insert the knot $\bar{u}$ into U to form the new knot vector $\bar{U}$;

$$\bar{U} = \{\bar{u}_0 = u_0,...,\bar{u}_k = u_k, \bar{u}_{k+1} = \bar{u}, \bar{u}_{k+2} = u_{k+1},...,\bar{u}_{m+1} = u_m\}. \tag{10}$$

After the knot insertion, $\mathbf{C}^w(u)$ has the representation of the form;

$$\mathbf{C}^w(u) = \sum_{i=0}^{n+1} \bar{N}_{i,p}(u)\mathbf{Q}_i^w \tag{11}$$

where $\{\bar{N}_{i,p}(u)\}$ are the pth-degree basis function on $\bar{U}$ and $\{\mathbf{Q}_i^w\}$ is the new set of control points. The curve is the same both geometrically and parametrically before and after the knot insertion. While the new set of control point $\{\mathbf{Q}_i^w\}$ is different from $\{\mathbf{P}_i^w\}$, the new knot vector $\bar{U}$ includes U.

New set of control points $\{\mathbf{Q}_i^w\}$can be computed by the following formula [Piegl97].

$$\mathbf{Q}_i^w = a_i\mathbf{P}_i^w + (1 - a_i)\mathbf{P}_{i-1}^w \tag{12}$$

where

$$a_i = \begin{cases} 1 & i \le k - p \\ \dfrac{\bar{u} - u_i}{u_{i+p} - u_i} & k - p + 1 \le i \le k \\ 0 & i \ge k + 1 \end{cases} \tag{13}$$

Information is encoded into this watermark by choosing the value of the inserted knot $\bar{u}$. We call a knot inserted for watermarking a *watermark knot*. Since a NURBS curve can be inserted with more than one knot, which multiplies data capacity of the watermark per curve.

The value of watermark knot $\bar{u}$ can take any value in the range $u_0 < \bar{u} < u_m$. In our proof-of-concept implementation, we arbitrarily decided to insert the watermark knot near the middle of a pair of adjacent knots. Deviation from the midpoint is determined by the data value and an amplitude scale parameter g. Assuming the message data d of size L bits, we compute the watermark knot $\bar{u}$ as

$$\bar{u} = \frac{u_{k+1} - u_k}{2}\left(1 + g\,\frac{d - 2^{L-1}}{2^L}\right) \tag{14}$$

The amplitude scale parameter g is chosen small enough (e.g., $g \le 0.01$) so that $\bar{u}$ stays close to the middle.

Extraction of the watermark is non-blind, so that the original cover-NURBS curve is required for extraction. Comparing the knot vector of the original curve with that of the watermarked curve finds inserted knot(s) and their value(s). Embedded message can be recovered easily from the knot values.

If multiple watermark knots are added to a NURBS curve, they must be ordered somehow to correctly embed and extract information. (Imagine a 16 bit message, in which each of the four 4 bit fragments is encoded into one of four watermark knots in a knot vector. Without ordering among the fragments, correct extraction of the 16 bit message is not possible.) Ordering of a set of knots on a NURBS curve can be introduced, for example, by sorting the watermark knots by their value. To embed a larger chunk of data, multiple NURBS curves must be ordered. Such ordering can be introduced, for example, by using topological connection the curves have in a CAD model. Extending this watermarking method to a tensor product NURBS surface is trivial.

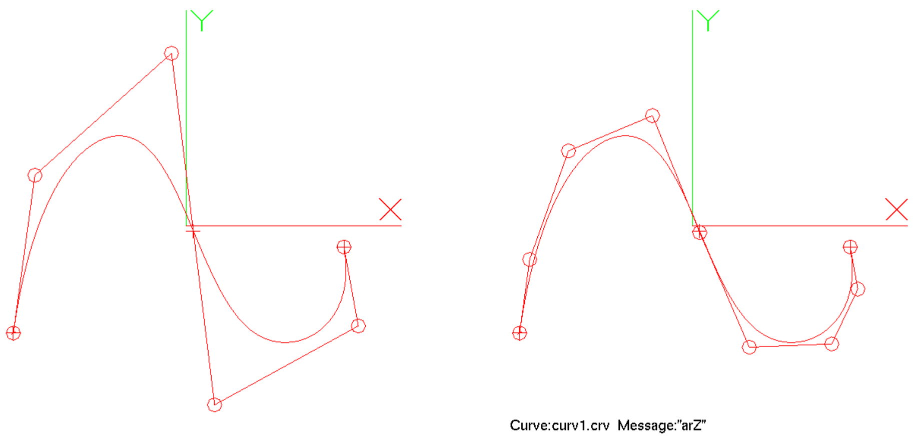
A possible attack on this watermarking scheme would be to insert one or more knot to confuse identification of watermark-knots. This attack can be accomplished without compromising the requirement of exact geometric shape preservation. A possible method to weaken such attack is to employ a pseudo-random number sequence generated by a seed, called *stego-key*, to uniquely specify locations of inserted knot in an ordered sequence of multiple NURBS curves. The extraction stego-key can be sent separately from the CAD data by using an established public-key infrastructure.

Figure 5 shows an example of watermarking a NURBS curve by inserting three knots. While the shape of the control polygon changes, that of the NURBS curve itself remains unchanged.

3.3 Shape-altering, size-preserving methods

While tolerating shape change may sound odd for a CAD model, applications exists that do tolerate such shape changes. For example, interference checks performed to evaluate manufacturability, a maintenance manual, or an on-line parts catalogue do not require exact shape preservation. If exact shape preservation is not an issue, additional approaches exist for watermarking parametric curves and surfaces.

- **Control Points and/or Weights Modulation:** 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.



(a) Before watermarking. *(b) After watermarking by inserting three knots.*

Figure 5. A NURBS curve before (a) and after (b) watermarking by using knot insertion. The value of the inserted knot encodes information to be embedded into the watermark.

- **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 numerical values, such as control point coordinates, weights, and knot values may be performed either in their original domain or in a transformed domain. Watermarking in transformed domain is one of the most popular approaches to watermarking for image data type. As the transformation, Fourier, wavelet, discrete-cosine and other transformations are employed. The principle of the transformed domain technique is borrowed from spread spectrum communication. Transformed-domain watermarking techniques can be applied to any kind of data that consists of an ordered set of numbers. For example, algorithms described in Section 3.1.1 and Section 3.2.1 could benefit from this spread-spectrum approach.

4. CONCLUSION

This paper presented digital watermarking as a possible tool to provide security to 3D geometric CAD data. The paper first examined 3D geometric CAD data as a multimedia data type, exploring a set of operations that are necessary for 3D geometric CAD data to become one of the full-fledged multimedia data types. The paper then focused on digital watermarking as a possible means to provide intellectual property management, authentication, and tamer detection operations for 3D geometric CAD models.

In particular, this paper focused on the methods to watermark parametric curves and surfaces, both of which are mainstay data objects for 3D geometric CAD systems. Examples of two watermarking algorithms for NURBS curves and surfaces, one based on rational linear reparameterization and the other based on knot insertion have been presented. Both methods exactly preserve geometric shape of NURBS curves and surfaces. The former, the one based on reparameterization, also preserves data size but the watermark embedded by using this method is not robust. The latter, the one based on knot insertion, is quite robust against attempted removal assuming that the exact preservation of geometric shape is demanded by the application.

REFERENCES

[Benedens99] O. Benedens, Geometry-Based Watermarking of 3D Models, *IEEE CG&A*, pp. 46-55, January/February 1999.

[Hartung98] 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.

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

[MPEG4] ISO/IEC JTC1/SC29/WG11 *MPEG-4 Visual* and *MPEG 4 SNHC.*

[Kanai98] 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 Modelling: Fundamentals and Applications (GEO-6)*, pp. 296-307, Tokyo, Japan, December 1998.

[Menezes96] A. J. Menezes, P. C. van Oorshot, and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

[Ohbuchi97] 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.

[Ohbuchi98a] 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.

[Ohbuchi98b] 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).

[Ohbuchi99] R. Ohbuchi, H. Masuda, and M. Aono, A Shape-Preserving Data Embedding Algorithm for NURBS Curves and Surfaces, In Proceedings of *Computer Graphics International ’99 (CGI ’99)*, pp. 180-187, Canmore, Canada, June 4-June 11, 1999.

[Piegl97] L. Piegl, W. Tiller, *The NURBS Book*, 2nd Edition, Springer, Berlin, 1997.

[Yeo99] B-L. Yeo and M. M. Yeung, Watermarking 3D Objects for Verification, *IEEE CG&A*, pp. 36-45, January/February 1999.