WATERMARKING 3D SHAPES USING LOCAL MOMENTS

Adrian G. Bors

Dept. of Computer Science, University of York, York YO10 5DD, U.K.
E-mail: adrian.bors@cs.york.ac.uk

ABSTRACT

In this paper a method using local moments is proposed for watermarking 3D shapes modelled by mesh surfaces. The moments are used for separating two regions in selected areas. Two different approaches are adopted for 3D shape watermarking, according to the boundary surface between the two regions: by using parallel planes and bounding ellipsoids, respectively. The proposed algorithms change locations of 3D vertices by placing them in one of the two regions, according to a given code. Both algorithms use a vertex selection procedure followed by the embedding. The paper provides a statistical analysis of distortions caused in shapes by geometrical perturbations. Experimental results are provided for various 3D graphical objects.

1. INTRODUCTION

Most of the research on watermarking has concentrated on audio signals, images, or video sequences [1, 2]. Only recently there is a growing interest in 3D object watermarking [3]-[9]. The nature of shape and graphical data representation is completely different from that of other multimedia data types. A mesh based representation of a 3D object consists of a graph having vertices as its knots, joined by edges and polygonal surfaces. Applying image watermarking algorithms to graphical data is not straightforward and a new methodology should be adopted in the case of 3D shape watermarking.

Authentication of 3D graphical objects has been considered by Yeo and Yeung [3], and by Fornaro and Sanna [4]. Results provided by a copyright protection watermarking algorithm employing modifications in histograms of surface normals were reported by Benedens in [5]. Watermarking of 3D polygonal meshes in spectral domain has been shown to be robust under various attacks by Ohbuchi et al. [6, 7].

Most of the approaches developed for watermarking graphical objects need the knowledge of the original un-watermarked object in the detection stage [4, 5, 7]. In many applications this is not acceptable and there are only a few blind watermarking algorithms [8, 9]. This paper extends the approach from [9], where two different algorithms that embed controlled local geometrical perturbations have been considered. The watermarking algorithm consists of two steps. In the first step a chain of vertices and their neighborhoods are selected and ordered. In the second step locations of selected vertices are changed according to their local neighborhood moments and to the information bit to be embedded. The paper is organised as follows. The procedure for selecting the watermark locations is described in Section 2, the effect of shape perturbations is studied in Section 3, while the watermark embedding and detection algorithm is detailed in Section 4. Experimental results are provided in Section 5 and the conclusions are drawn in Section 6.

2. VERTEX SELECTION

The aim of this work is to embed information in a polygonal mesh based shape representation. The proposed 3D shape watermarking method produces imperceptible changes in the location of certain vertices such that they eventually embed a code. The approach consists of two steps: identifying the most suitable vertices to carry information and the embedding operation.

Firstly, the vertices where to embed information are selected such that the change produced in the 3D shape would not be easily identified by a potential attacker. The set of vertices chosen to carry information is denoted by $\mathcal{C}$, a subset of all vertices that made up the object. A vertex is denoted by V_i and its coordinates are defined by the vector $\mathbf{V}_i$. The neighborhood of a vertex consists of all the vertices from the same object that are connected to it:

$$\mathcal{N}(V_i) = \{V_j \mid |V_jV_i| > 0, j = 1, \ldots, N_i\} \tag{1}$$

where $|V_jV_i|$ denotes the cardinal set between two neighbouring vertices V_i and V_j, while N_i denotes the total number of vertices from $\mathcal{N}(V_i)$. A vertex and its neighborhood are considered as part of a set $\{V_i, \mathcal{N}(V_i)\}$. The most appropriate vertices for watermarking are those from mesh areas consisting of small polygons in the 3D object. Such regions are the equivalent of image areas with texture, details or noise that have been deemed appropriate for image watermarking [1, 2].

Let us consider a first and second order moment based description for the V_i's neighborhood. Such a description can be represented geometrically as an ellipsoid which roughly models the local geometry. The center of the ellipsoid is given by :

$$\mu_i = \frac{\sum_{V_j \in \mathcal{N}(V_i)} \mathbf{V}_j}{N_i} \tag{2}$$

where N_i is the number of vertices in the neighborhood. The shape of the ellipsoid is calculated as the second order moment (variance) of the given neighborhood :

$$\mathbf{S}_i = \frac{\sum_{V_j \in \mathcal{N}(V_i)} (\mathbf{V}_j - \mu_i)(\mathbf{V}_j - \mu_i)^T}{N_i} \tag{3}$$

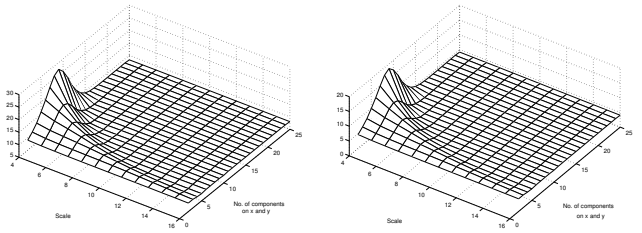
Certain neighborhoods, which are not deemed appropriate for watermarking, provide a singular covariance matrix $\mathbf{S}_i$. Such situations happen when all the vertices, from the neighborhood $\mathcal{N}(V_i)$, are located onto the same plane.

Let us consider $D(V_i)$, the squared distance from a vertex to its neighborhood $\mathcal{N}(V_i)$:

$$D(V_i) = \sum_{V_j \in \mathcal{N}(V_i)} \|\mathbf{V}_j - \mathbf{V}_i\|^2 \tag{4}$$

The polygonal surface of a certain neighborhood can be considered as a measure of the local shape variation. The area of all the polygons (considered here as triangles), connected to a certain vertex,

0-7803-8554-3/04/$20.00 ©2004 IEEE.



(a) Distance to neighborhoods. (b) Surface area.

Fig. 1. Evaluation of average $D(V_i)$ and $A(V_i)$.

is:

$$A(V_i) = \frac{1}{2}\sum_{V_j \in \mathcal{N}(V_i)} \|V_iV_j\| \, \|V_iV_{(j+1)\bmod N_i}\| \sin(\overrightarrow{V_iV_j}, \overrightarrow{V_iV}_{(j+1)\bmod N_i}) \quad (5)$$

where the vertices in the neighborhood are considered ordered, and where mod denotes operation modulo. In order to ensure the unobservability of the watermarks, modifications can be produced only in areas consisting of small polygons. A threshold $T(V_i)$, depending on the local surface $A(V_i)$, or on the distance $D(V_i)$, is considered for the selection of vertices.

The distances $D(V_i)$, for every vertex V_i to its neighborhood, are evaluated according to (4). A head chain vertex, having the smallest $D(V_i)$, is selected. The set $\mathcal{C}$ is made up of vertices and their neighborhoods that fulfil the following conditions:

$$\{V_j \in \mathcal{C} \mid \mathcal{N}(V_j) \notin \mathcal{C} \bigwedge D(\hat{V}_j) < T(V_j)\} \quad (6)$$

where $j = 1, \ldots, M$, M is the total number of vertices that fulfils the above conditions and $\hat{V}_j$ is the watermarked vertex. The selected bit holding vertices are ordered according to their Euclidean distances to the head chain vertex.

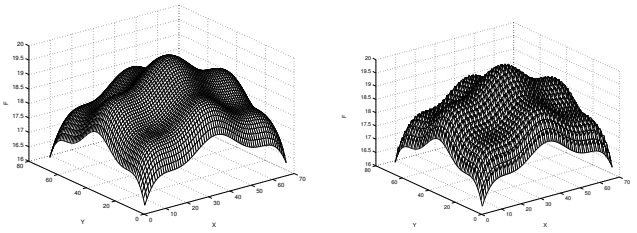
Each selected vertex and its neighborhood will hold a bit after watermarking. In order to increase the robustness to cropping, the set $\mathcal{C}$ is split into localized subsets. Considering a code with B bits to be embedded into the given shape, these bits will be redundantly embedded using localized vertex chains of $\mathcal{C}$. The set of bit-holder vertices $\mathcal{C}$ is split into groups of B sets $\{V_i, \mathcal{N}(V_i)\}$, each group to be embedded in the same geometrical proximity. The watermark is repeatedly embedded a number of times equal to $\lfloor M/B \rfloor$.

3. PRODUCING PERTURBATIONS ON SURFACES

A large variety of surfaces can be modelled by using Gaussian mixtures :

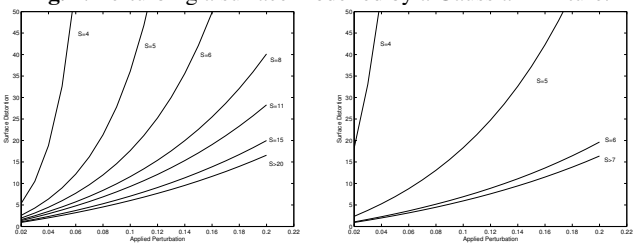
$$F(\mathbf{x}) = \lambda \sum_{i}^{R\times R} \exp\left[-\frac{(\mathbf{x}-\mathbf{c}_i)^T(\mathbf{x}-\mathbf{c}_i)}{S^2}\right] \quad (7)$$

where λ is a constant representing the contribution of each Gaussian function and the normalization, $\mathbf{c}$ is the Gaussian function center, S is the scale parameter and $\mathbf{x}$ is the vertex vector, chosen equally spaced on a plane in the following range: $\mathbf{x} = \{1, \ldots, R\lfloor 100/R \rfloor\} \times \{1, \ldots, R\lfloor 100/R \rfloor\}$. An equal number of Gaussian components is considered along each axis, respectively R, while their centers are equally spaced. Vertices $\mathbf{x}_p$ are selected for modification by subsampling the mesh with 2 on each axis, while $N_i = 8$ for all selected vertices. Many different surfaces are considered for $S = [4, 16]$ and $N = \{2, \ldots, 25\}$. Distances $D(\mathbf{x}_p)$ are calculated from each selected vertex $\mathbf{x}_p$ to its 8-vertex neighborhood, by using (4). The average distance $D(\mathbf{x}_p)$ for all selected vertices, to their neighborhoods, for a certain surface, characterized by S and R is shown in Fig. 1a. In Fig. 1b an approximation



(a) Initial surface. (b) Distorted, $\eta = 0.02$.

Fig. 2. Perturbing a surface modelled by a Gaussian mixture.



(a) Distortions when $R = 3$. (b) Distortions when $R = 10$.

Fig. 3. Distortions produced in the surface by local perturbations.

of the average surface $A(V_i)$ is shown. One can observe the similarity of the two plots from Figure 1. The average surface connected to a selected vertex is larger for a small scale parameter when given a certain number of components.

Surface perturbations are produced onto the surface modelled by $F(\mathbf{x})$ from (7), along z direction, in selected vertices. The perturbation induced in the given surface is represented as a fraction from the local distances provided by (4):

$$F(\hat{\mathbf{x}}_p) = F(\mathbf{x}_p) + \eta D(\mathbf{x}_p) \quad (8)$$

where $\hat{\mathbf{x}}_p$ is the perturbed vertex, $\eta \in (0, 1)$ is the perturbation ratio. In Fig. 2 the resulting surface for $N = 3$ Gaussian components with $S = 23$ is represented before and after inducing perturbations, when considering $\eta = 0.02$.

In order to assess distortions produced in the shape, a large set of surfaces is considered by varying S and R. The distortion in the distance from each perturbed vertex to its neighborhood is:

$$E = \frac{1}{M}\sum_{i=1}^{M} |D(\hat{\mathbf{x}}_i) - D(\mathbf{x}_i)| \quad (9)$$

where $D(\hat{\mathbf{x}}_i)$ is the distance from the perturbed site to its neighborhood, and M is the total number of perturbed vertices. The distortions plots, calculating E for $R = 3$, and $R = 10$ components, when assuming various values for S, are represented in Figures 3a and 3b, respectively. From these plots we can observe that distortions are higher for smaller S, while they are linearly predictable for surfaces that tend to be smooth.

4. EMBEDDING AND RETRIEVING THE WATERMARK

The watermark code is embedded in a set of B vertices and their neighborhoods, modelled as ellipsoids. Two separate geometric areas are considered in the space defined by the set $\{V_i, \mathcal{N}(V_i)\}$, one for embedding a bit of $\mathbf{0}$, and the other for embedding a bit of $\mathbf{1}$. The vertex to be watermarked is moved to one of the two regions, according to its corresponding bit.

The first approach defines two parallel planes for a bit-holder site V_i, using the geometry of its neighborhood $\mathcal{N}(V_i)$. Firstly, the normal $\overrightarrow{\mathbf{T}}_j$ at each vertex from the neighborhood, $V_j \in \mathcal{N}(V_i)$, is evaluated. The surface normal $\overrightarrow{\mathbf{T}}_i$ at the vertex V_i is taken as the average of surface normal directions corresponding to all its

adjacent polygons. The orientation of the two bounding planes is denoted by $\overrightarrow{\mathbf{Q}}(V_i)$ and is given by averaging the orientation of all surface normals from that neighborhood :

$$\overrightarrow{\mathbf{Q}}(V_i) = \frac{\sum_{V_j \in \mathcal{T}(V_i)} \overrightarrow{\mathbf{T}}_j}{N_i} \quad (10)$$

where N_i is the number of vertices in the neighborhood $\mathcal{N}(V_i)$. The two planes are located, at equal distance from the neighborhood's center μ_i, on both sides, calculated in the direction of the average neighborhood normal, $\overrightarrow{\mathbf{Q}}(V_i)$. The distance from the planes to the neighborhood's center is derived as the variance of distances in the local neighborhood, projected along the direction $\overrightarrow{\mathbf{Q}}(V_i)$:

$$\Psi(V_i) = \frac{\sum_{V_j \in \mathcal{N}(V_i)} [(\mathbf{V}_j - \mu_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)]^2}{N_i} \quad (11)$$

where $|\cdot|$ denotes the scalar product. In the case when embedding a $\mathbf{1}$ bit, the vertex $\mathbf{V}_i$ is projected along the direction of $\overrightarrow{\mathbf{Q}}(V_i)$ inside the volume defined by the parallel planes such that :

$$|(\hat{\mathbf{V}}_i - \mu_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)| < \Psi(V_i) - \epsilon \quad (12)$$

where $\hat{\mathbf{V}}_i$ is the new location of the watermarked vertex and ϵ is a small distance. When embedding a $\mathbf{0}$ bit, the vertex is projected outside the volume defined by the parallel planes. The projection defined by (12) ensures a minimal local distortion in the 3D shape. The updating rule for embedding a bit of $\mathbf{1}$ is given by:

$$|\hat{\mathbf{V}}_i - \mathbf{V}_i| = [(\mathbf{V}_i - \mu_i) \cdot \overrightarrow{\mathbf{Q}}(V_i) - \Psi(V_i) + \epsilon] \frac{\overrightarrow{\mathbf{Q}}(V_i)}{\|\overrightarrow{\mathbf{Q}}(V_i)\|} \quad (13)$$

while for embedding a bit of $\mathbf{0}$ is similar but with opposite direction of modification. The updating equation (13) is used only when the relationship (12) is not already fulfilled by the local geometry according to the bits to be embedded in the 3D shape structure.

The second embedding approach consists of defining bounding ellipsoids, for each chosen bit-holding site $V_i \in \mathcal{C}$. When embedding a $\mathbf{1}$ bit, the vertex V_i is projected along the shortest path inside the bounding ellipsoid:

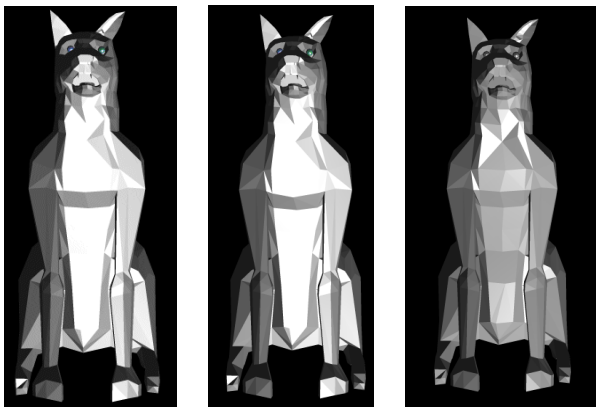
$$(\hat{\mathbf{V}}_i - \mu_i)^T \mathbf{S}_i^{-1} (\hat{\mathbf{V}}_i - \mu_i) < K - \epsilon \quad (14)$$

where K provides the extension of the bit-embedding areas and ϵ is a small quantity. A bit $\mathbf{0}$ is embedded by moving the vertex V_i just outside its corresponding bounding ellipsoid. The shortest updating path is given by the normal to the bounding ellipsoid surface which is obtained after differentiating its surface equation. The updating equation for the vertex location when embedding a bit of $\mathbf{1}$ is given by:

$$\begin{aligned} \hat{\mathbf{V}}_i \quad = \quad & \mathbf{V}_i - \frac{(\mathbf{V}_i - \mu_i)^T \mathbf{S}_i^{-1}}{\|(\mathbf{V}_i - \mu_i)^T \mathbf{S}_i^{-1}\|} \cdot \\ & \cdot [(\mathbf{V}_i - \mu_i)^T \mathbf{S}_i^{-1} (\mathbf{V}_i - \mu_i) - K + \epsilon] \end{aligned} \quad (15)$$

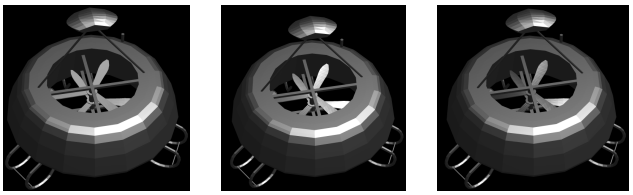
while for a bit $\mathbf{0}$ is similar but with opposite updating direction.

The watermark detection stage aims to recover the information that has been stored in the shape content. In this approach the original 3D object is not required in the detection stage. Firstly, the set of marked vertices and their neighborhoods is selected in the same way as it is described in Section 2. A sequence of bits is retrieved from the 3D shape by checking the local shape constraints characterizing the bits $\mathbf{1}$ or $\mathbf{0}$. Eventually, a XOR operation can be evaluated between the resulting set of bits and the watermark code [2]. A decision is taken based on an acceptable minimal bit error. Error correction codes, that are widely used in communication systems [10], are employed in order to improve the robustness to errors in the detected bit-sequence. However, error correction codes lead to the inclusion of additional bits, increasing the length of the code that must be embedded.



(a) (b) (c)

Fig. 4. Graphical object representing a dog. (a) original; watermarked using: (b) bounding ellipsoids; (c) parallel planes.



(a) (b) (c)

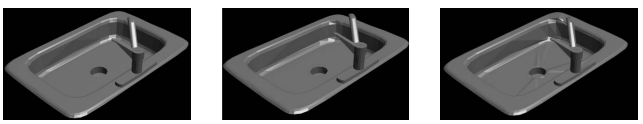
Fig. 5. Graphical object representing a fan. (a) original; watermarked using: (b) bounding ellipsoids; (c) parallel planes.

5. EXPERIMENTAL RESULTS

The selected objects for watermarking have a low number of vertices and polygons and they represent both animation characters and technical objects. A watermark code of 32 bits is generated using cyclic redundancy error checks of a particular bit-string. The 32 bit code is split into 8 chains, of 4 bits each. Cyclic redundancy checks have the property of mapping quite evenly across the space of possible values. The information to be embedded is seven bits long, where four bits represent the watermark code and the rest are used as error correction bits. Experiments found no false alarms when checking a large set of graphical objects for the detection of randomly generated codes.

Experimentally it was found that a watermark can be safely detected in a 3D graphical object if 75 % of the code bits from at least half of all the bit-chains, forming the watermark code, can be recovered. Graphical characteristics of the objects under consideration together with their capacity to store information are provided in Table 1. From Table 1 it can be observed that more bit holder vertices can be found when using bounding ellipsoids when compared with using parallel planes for defining the embedding volume. Lower distortions in the case of bounding ellipsoids are less likely to affect the embedding rules, according to (6).

The results for three objects are displayed: "dog" in Fig. 4a, "fan" in Fig. 5a and "sink" in Fig. 6a. In Figs. 4b, 5b and 6b, the same objects are shown after being watermarked using bound-

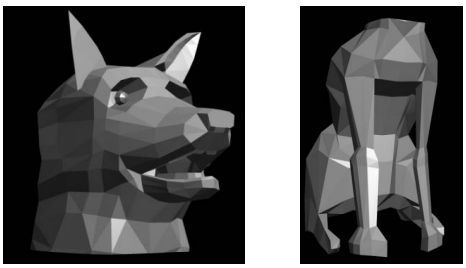


(a) (b) (c)

Fig. 6. Graphical object representing a sink. (a) original; watermarked using: (b) bounding ellipsoids; (c) parallel planes.

Graphical Model	No. of Vertices/ Polygons	Bit holders bounding ellipsoids	Bit holders parallel planes
"Dog"	654 / 1286	28.0%	17.3%
"Fan"	1532 / 2634	18.0%	13.0%
"Guillotine"	2723 / 4578	16.6%	11.3%
"Screwdriver"	2073 / 4076	13.5%	9.8%
"Sink"	674 / 1068	14.5%	9.8%

Table 1. Characteristics of various graphical objects.



(a) Head of the "dog" (b) Body of the "dog"

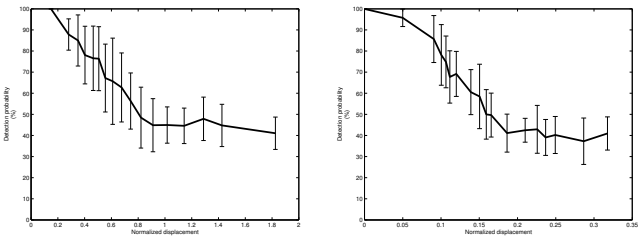
Fig. 7. Cropping results of the watermarked "dog" model.

ing ellipsoids, while in Figs. 4c, 5c and 6c are represented after being watermarked using parallel planes. It can be observed that the distortions caused by the parallel planes algorithm have higher visibility than the distortions caused by the bounding ellipsoids algorithm. Such distortions are more evident in the planar structure of the "dog" model and on the bottom of the "sink", when using parallel planes for defining the embedding volumes.

The watermark robustness to rotation, scaling and other affine transformations of the watermarked 3D shape is evident. The watermark is also resistant to changes in the vertex order in the file description data. The first robustness test consists of cropping specific regions of the graphical object. The region representing the head from the "dog" object is severed from the region of the "dog" body. New vertices are generated in the region of separation. The resulting two shapes are shown in Fig. 7a for the dog's head and in Fig. 7b for the dog's body. Due to the repetition embedding, the watermark was fully recovered from the "dog-head" shape, while it was 50 % recovered from the "dog-body" shape. In a statistical attack, random perturbations are applied to the 3D structure of the graphical object. The perturbations are modelled according to Gaussian noise $\|\breve{\mathbf{V}}_i - \mathbf{V}_i\| \sim \mathcal{N}(0, \sigma)$, where $\breve{\mathbf{V}}_i$ represents the location of a distorted vertex by noise. The normalized dispersion is used to test the level of distortion:

$$E(\sigma) = \frac{1}{L} \sum_{i=1}^{L} \frac{\sigma^2}{D(V_i)} \tag{16}$$

where L represents the total number of vertices in the graphical object and $D(V_i)$ is provided in (4). The plots from Figures 8a and 8b show the watermark detection results from the 3D noisy objects, when watermarked by bounding ellipsoids, and by using parallel planes, respectively. Error bars are showing the absolute deviation from the local average, for a certain normalized displacement $E(\sigma)$. All these experimental results show the watermark resilience and robustness in the case of both graphical watermarking algorithms. However, the bounding ellipsoid watermarking approach produces less visible modifications and has a higher robustness to attacks.



(a) Bounding ellipsoids (b) Parallel planes

Fig. 8. Robustness results when adding Gaussian noise

6. CONCLUSIONS

This paper develops a new digital watermarking methodology for 3D shapes and graphical objects. The proposed watermarking algorithms have two stages and do not require the original object for the watermark detection. In the first stage, a set of vertices and their neighborhoods are selected and ordered. The embedding consists of localized geometrical changes of selected vertex locations. Two different techniques are considered for information embedding. These techniques are different in the way how a local neighborhood is represented geometrically: by using bounding ellipsoids and parallel planes. The analysis of the effects when inducing surface perturbations in 3D surfaces is provided. The watermark robustness was tested to various levels of noise perturbations in the 3D shape structure as well as to 3D object cropping.

7. REFERENCES

[1] I. Cox, M. Miller, and J. Bloom. *Digital watermarking*. Morgan Kaufmann, 2001.

[2] A.G. Bors, I. Pitas, "Image watermarking using block site selection and DCT domain constraints," *Optics Express,* vol. 3, no. 12, pp. 512-522, 1998.

[3] B.-L. Yeo and M.M. Yeung, "Watermarking 3D objects for verification," *IEEE Computer Graphics and Applications*, vol. 19, no. 1, pp. 36–45, 1999.

[4] C. Fornaro and A. Sanna, "Private key watermarking for authentication of CSG models," *Computer-Aided Design*, vol. 32, pp. 727-735, 2000.

[5] O. Benedens, "Geometry based watermarking of 3D models," *IEEE Computer Graphics and Applications*, vol. 19, no. 1, pp. 46-55, 1999.

[6] R. Ohbuchi, H. Masuda, and M. Aono, "Watermarking three polygonal models through geometric and topological modifications," *IEEE Journal on Selected Areas in Communication*, vol. 16, no. 4, pp. 551-560, 1998.

[7] R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama. "Watermarking 3D polygonal meshes in the mesh spectral domain," *Proc. of Graphics Interface*, Ottawa, Canada, pp. 9–17, 2001.

[8] O. Benedens, C. Busch, "Towards blind detection of robust watermarks in polygonal models," *Proc. EUROGRAPHICS, Comp. Graphics Forum,* vol. 19, no. 3, pp. C199-C208, 2000.

[9] T. Harte, A.G. Bors, "Watermarking 3D models," *Proc. IEEE Intern. Conf. on Image Processing,* Rochester, N.Y., vol. II, 2002, pp. 661-664.

[10] J. Baylis, Error-correcting codes. Chapman-Hall, London, 1998.