Watermarking Mesh-Based Representations of 3-D Objects Using Local Moments

Adrian G. Bors, *Senior Member, IEEE*

***Abstract*—A new methodology for fingerprinting and watermarking three-dimensional (3-D) graphical objects is proposed in this paper. The 3-D graphical objects are described by means of polygonal meshes. The information to be embedded is provided as a binary code. A watermarking methodology has two stages: embedding and detecting the information that has been embedded in the given media. The information is embedded by means of local geometrical perturbations while maintaining the local connectivity. A neighborhood localized measure is used for selecting appropriate vertices for watermarking. A study is undertaken in order to verify the suitability of this measure for selecting vertices from regions where geometrical perturbations are less perceptible. Two different watermarking algorithms, that do not require the original 3-D graphical object in the detection stage, are proposed. The two algorithms differ with respect to the type of constraint to be embedded in the local structure: by using parallel planes and bounding ellipsoids, respectively. The information capacity of various 3-D meshes is analyzed when using the proposed 3-D watermarking algorithms. The robustness of the 3-D watermarking algorithms is tested to noise perturbation and to object cropping.**

***Index Terms*—Mesh representation, moments, three-dimensional (3-D) graphical objects, watermarking.**

I. Introduction

DIGITAL watermarking, fingerprinting, digital rights management, and steganography employ various techniques for embedding information into digital media. The information embedded can be used for media copyright protection, authentication, annotation, access control, data hiding, or for data and media manipulation. Usually in the watermarking terminology a "cover media" represents the original object, while the watermarked object is called "stego media." A watermark should fulfill a set of requirements such as imperceptibility by human perception or by software detection, as well as robustness to stego media processing and to potential attacks [1]. Most of the research in watermarking has concentrated on cover media such as audio data, still images (bitmaps), fax data, or video [1]–[3]. Image or audio watermarking has been performed in the spatial domain [4] or in a transform domain (for example, in the discrete cosine transform (DCT) coefficients) [1], [5]. However, due to the nature of the data representing the cover media, these methods cannot be applied to three-dimensional (3-D) graphical objects and models.

A 3-D graphical object can be represented in various ways: using voxels, polygonal meshes, constructive solid geometry (CSG) or as an implicit set of parametrized equations, such as nonuniform rational B-splines (NURBS), or other splines [6]. While voxels provide a volumetric description of 3-D objects [7], meshes, and parametric equations model their surfaces. A polygonal mesh of a 3-D object is represented as a graph consisting of a set of vertices, joined by edges and polygons. Unlike in other visual media representations, in the case of 3-D objects modeled by meshes, we have to deal with a relatively low quantity of data. A set of attributes such as color, texture, shading, can be associated with each vertex [6].

Watermarking 3-D graphical objects has been performed from various perspectives. In [8], an optical-based system employing phase shift interferometry was devised for mixing holograms of 3-D objects, representing the cover media and the hidden data, respectively. Watermarking of texture attributes has been attempted by Garcia and Dugelay [9]. Hartung *et al.* watermarked the stream of MPEG-4 animation parameters, representing information about shape, texture and motion, by using a spread spectrum approach [10].

Attributes of 3-D graphical objects can be easily removed or replaced. This is why most of the 3-D watermarking algorithms are applied on the 3-D graphical object geometry. Authentication is concerned with the protection of the cover media and should indicate when this has been modified. Authentication of 3-D graphical objects by means of fragile watermarking has been considered in [11], [12]. Ohbuchi *et al.* discussed three methods for embedding data into 3-D polygonal models in [13]. Many of the approaches applied onto the 3-D object geometry aim to ensure invariance at geometrical transformations. This can be realized by using ratios of various 2-D or 3-D geometrical measures [13]–[16]. Results provided by a copyright protection watermarking algorithm employing modifications in histograms of surface normals were reported by Benedens in [17]. Local statistics has been used for watermarking 3-D objects in [18], [19]. Multiresolution filters for mesh watermarking have been considered in connection with interpolating surface basis functions [20], and with pyramid-based algorithms [21]. Benedens and Busch introduced three different algorithms, each of them having robustness to certain attacks while being suitable for specific applications [22]. Algorithms that embed data in surfaces described by NURBS use changes in control points [16] or reparameterization [23]. Wavelet decomposition of polygons was used for 3-D watermarking in [24], [25]. Watermarking algorithms that embed information in the mesh spectral domain using graph Laplacian have been proposed in [26]–[28].

Manuscript received July 6, 2004; revised December 5, 2004. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Benoit Macq.

The author is with the Department of Computer Science, University of York, York YO10 5DD, U.K. (e-mail: adrian.bors@cs.york.ac.uk).

Digital Object Identifier 10.1109/TIP.2005.863116

A few characteristics can be outlined for the existing 3-D watermarking approaches. Some of the 3-D watermarking algorithms are based on displacing vertex locations [16], [19], [20] or on changing the local mesh connectivity [21], [23]. Minimization of local norms has been considered in the context of 3-D watermarking in [22], [29]. Localized embedding has been employed in [13], [16], [22]. Arrangements of embedding primitives have been classified according to their locality in: global, local and indexed [13]. Localized and repeatative embedding are used in order to increase the robustness to 3-D object cropping [28].

Preferably, a watermarking system would require in the detection stage only the knowledge of the watermark given by a key and that of the stego object. However, most of the approaches developed for watermarking 3-D graphical objects are nonblind and require the knowledge of the cover media in the detection stage [9], [10], [12], [17], [20], [21], [23]–[27], [29]. Some algorithms require complex registration procedures [20]–[22], [25], [29] or should be provided with additional information about the embedding process in the detection stage [15], [16], [24].

A nonlinear 3-D watermarking methodology that employs perturbations in the 3-D geometrical structure, is described in this paper. The watermark embedding is performed by a processing algorithm in two steps. In the first step, a string of vertices and their neighborhoods are selected from the cover object. The selected vertices are ordered according to a minimal distortion criterion. This criterion relies on the calculation of the sum of Euclidean distances from a vertex to its connected neighbors. A study of the effects of perturbations in the surface structure is provided in this paper. The second step estimates first and second order moments and defines two regions: one for embedding a bit of $\mathbf{1}$ and another one for embedding a bit of $\mathbf{0}$. First- and second-order moments have desirable properties of invariance to various transformations [30], [31] and have been used for shape description [7]. These properties can ensure the detection of the watermark after the 3-D graphical object was transformed by affine transformations. Two different approaches, that produce controlled local geometrical perturbations, are considered for data embedding: using parallel planes and bounding ellipsoids. The detection stage is completely blind in the sense that it does not require the cover object. The paper is organized as follows. The description of general requirements for 3-D watermarking is provided in Section II. The procedure for selecting the watermark locations is described in Section III. The analysis of the effect of perturbations in meshes is provided in Section IV, while the watermark embedding and detection algorithm is detailed in Section V. Experimental results are provided in Section VI, and the conclusions of this study are drawn in Section VII.

II. Digital Rights Management of 3-D Graphical Objects

The representation of graphical data is different from that of other multimedia data types. Images consist of 2-D data arrays arranged on a regular lattice, while video streams and volumetric images such as those acquired by magnetic resonance imaging (MRI), can be seen as data arrays on 3-D lattices, representing sampled information of moving sequences or 3-D graphical objects. In the case of polygonal mesh-based representations, 3-D objects are not a sampled quantity, but a graph description of locations and surfaces. In a graph-based description of a 3-D object, knots represent vertices given by their 3-D coordinates while their interconnectivity is given by a mesh [6]. Vertices are joined by edges and are forming flat, convex polygons. Usually, triangles are used rather than arbitrary polygons because triangles are always both flat and convex. As any polygon may be decomposed into a set of triangles, a triangle mesh representation does not limit the generality of the 3-D object representation. For proper shading and texturing, surface normals should be calculated for each surface patch [6].

Three-dimensional graphical objects can be modeled using computer graphics packages [6], web-design tools such as virtual reality modeling language (VRML), can be acquired using sensors such as 3-D laser scanners, MRI, computer tomography (CT), or they can be estimated from images. Various techniques can be used to model 3-D graphical objects from images, for example stereoscopic vision, shape from shading or texture, shape from motion, etc.

Watermarking has been employed for copyright protection and authentication. In the first case, the watermark code protects the copyright ownership and should subsist any likely changes the data may undergo. In the second case, the watermark is fragile and protects the data. Lately, new applications have been considered for watermarking, including fingerprinting for data queries and retrieval, registration of confidential information, behavioral information for graphical agent systems, etc.

Watermarking of 3-D graphical objects requires a completely new approach when compared to audio, image or video watermarking [1]–[3], [19], [32]. Moreover, testing of 3-D watermarking algorithms should be done in a completely new framework [33]. Requirements for 3-D watermarks include the nonperceptibility of changes brought to the original model by the watermark and the ability to detect the watermark even after the object underwent various transformations or attacks [1]. Such transformations can be inherent for 3-D object manipulation in computer graphics [6] or computer vision [31] or they may be done intentionally with the malefic purpose of removing the watermark. Transformations of 3-D meshes can be classified into geometrical and topological. Geometrical transformations include affine transformations such as rotation, scaling, or their combinations, and can be local or applied to the entire object. Topological transformations consists of changing the order of vertices in the object description file, polygon simplification, remeshing or cropping parts of the object. Other processing algorithms include object compression and encoding, such as MPEG-4 [10], [18], [25]. Smoothing and noise corruption algorithms can be mentioned from the category of intentional attacks. A large variety of attacks can be modeled generically by noise corruption. Noise corruption in 3-D models amounts to a succession of small perturbations in the location of vertices. Section IV provides a study of the effects of noise perturbation in surfaces represented by meshes. Depending on the application, various conditions can be imposed on the maximum level of distortion that a 3-D watermark system may undergo. For example in the case when watermarking 3-D computer aided design (CAD) models, the upper bound of the distortion level is given by the admissible tolerance.

In any watermarking or fingerprinting approach, there is a tradeoff between being able to make the watermark survive a set of transformations and the actual visibility of the watermark. Most of the existing 3-D watermarking algorithms are robust to certain attacks, but not to others. Usually, topology-based watermarking algorithms are not robust to affine transformations [22], while vertex displacement algorithms are not robust to mesh simplifications. It would be preferable to embed the watermark in regions displaying a great amount of variation in the 3-D object. This is similar with the procedure of considering regions of high frequency for image watermarking [1], [5].

An ideal watermarking system has a blind detection algorithm. A nonblind system would need the original cover media in the detection stage. Usually, it is expected that a nonblind approach can provide better robustness to various attacks [1]. However, a nonblind watermarking approach is not suitable for most applications.

III. Vertex Selection

The approach described in this paper consists of imperceptibly changing the location of certain vertices in such a way that eventually a code is embedded. Color, shading, texture, or other attributes may be easily changed or removed. Due to their functionality in graph-based shape representation, vertex coordinates are the most suitable features to be used for watermarking graphical objects [19], [22]. On the other hand, various attributes can be used to mask the changes produced to the local object geometry.

The approach described in this paper embeds an array of bits in selected vertices without altering the mesh topology. The 3-D watermarking algorithm consists of two distinct steps: identifying the most suitable vertices to carry watermarking information and the embedding operation itself. The identification of a suitable sequence of vertices or polygons for 3-D watermarking was also considered in [11], [13], [15].

A vertex is denoted by $V_i \in \mathcal{O}$, where $\mathcal{O}$ is the 3-D graphical object, and its coordinates are defined by the vector $\mathbf{V}_i$. Moments are invariant at affine transformations. Let us consider the neighborhood of a vertex as all the vertices from the same object that are connected to it by means of an edge

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

where $|V_j V_i|$ represents the cardinal set between two neighboring vertices V_i and V_j, while N_i denotes the number of vertices from $\mathcal{N}(V_i)$. The vertex V_i is not considered as a component of its own neighborhood. A vertex and its neighborhood are considered as part of a set $\{V_i, \mathcal{N}(V_i)\}$. Let us denote the array of all vertices selected to carry information by $\mathcal{B}$, a subset of the polygonal mesh defining the object $\mathcal{B} \subset \mathcal{O}$. Not all the vertices are appropriate to be used for watermarking. Changes in certain vertex locations can be more visible than in others. The most appropriate vertices for watermarking are those from areas characterized by short edges, displaying many small variations in the 3-D object surface. Such regions are the equivalent of image areas with textures or with a great amount of detail that are used for image watermarking [1], [5].

A graphical object is used in the context of graphical scenes, usually after various affine transformations such as scaling, translation and rotation [6]. Let us consider a second-order moment description for each neighborhood. Such a description can be represented geometrically as an ellipsoid which roughly models the geometry of the V_is neighborhood [7]. The center of the ellipsoid corresponds to the neighborhood's mean and is calculated by averaging the coordinates of the vertices from the entire neighborhood

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

where N_i is the number of vertices in the neighborhood $\mathcal{N}(V_i)$. The shape of the ellipsoid is calculated as the second-order moment (covariance matrix) of the set of locations in the given neighborhood

$$\boldsymbol{\Sigma}_i = \frac{\sum_{V_j \in N(V_i)} (\mathbf{V}_j - \boldsymbol{\mu}_i)(\mathbf{V}_j - \boldsymbol{\mu}_i)^T}{N_i}. \tag{3}$$

The ellipsoid which locally describes the neighborhood $\mathcal{N}(V_i)$ geometry is given by

$$(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) = K \tag{4}$$

where $\mathbf{x}$ is a vector located on the ellipsoid described by $(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ characterising $\mathcal{N}(V_i)$, and K represents the extension of the ellipsoid with respect to its center $\boldsymbol{\mu}_i$. Certain neighborhoods may lead to a singular covariance matrix. Such situations correspond to the cases when all the vertices from the given neighborhood are located in the same plane and when a representation as in (4) is not appropriate.

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

$$D(V_i) = \sum_{V_j \in \mathcal{N}(V_i)} \|\mathbf{V}_i - \mathbf{V}_j\|^2 \tag{5}$$

where $\|\cdot\|$ denotes the Euclidean distance. This measure is used in Section IV to study the effects of geometrical perturbations in 3-D graphical objects. 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, is calculated as

$$A(V_i) = \frac{1}{2} \sum_{V_j \in \mathcal{N}(V_i)} \|\mathbf{V}_i - \mathbf{V}_j\| \left\|\mathbf{V}_i - \mathbf{V}_{(j+1)\text{mod} N_i}\right\| \sin(\overrightarrow{V_i V_j}, \overrightarrow{V_i V}_{(j+1)\text{mod} N_i}) \tag{6}$$

where the vertices in the neighborhood are considered ordered, and where mod denotes operation modulo. It can be observed from (5) and (6) that both local measures $D(V_i)$ and $A(V_i)$ depend on the square of edge lengths in the local neighborhood. Usually, polygons with large surfaces have long adjacent edges.

The threshold for selecting a certain vertex as a bit-holder for watermarking is calculated with respect to the local surface

$$T(V_i) = k_1\sqrt{A(V_i)} \simeq k_2\sqrt{D(V_i)} \tag{7}$$

where k_1, k_2 are small constants. The underlying assumption used in (7) and studied in Section IV is that we can produce unobservable modifications in areas consisting of small polygons, while any modification of vertices adjacent to large polygons may become visible. Vertex sites are selected and ordered. Various methods have been proposed for ordering vertices in meshes for watermarking applications [11], [15], [18], [19]. The vertex selection procedure adopted in this paper is introduced in the following. Distances $D(V_i)$ from every vertex $V_i \in \mathcal{O}$ to its neighborhood, are evaluated according to (5). Changes in vertices having small distances $D(V_i)$ to their neighborhoods $\mathcal{N}(V_i)$ are less perceivable than changes performed in other vertices. The first vertex $V_{(1)}$ in the chosen set is selected as the one having the smallest distance to its neighborhood after watermarking

$$V_{(1)} = \min_{V_i \in \mathcal{B}} D(\hat{V}_i) \tag{8}$$

where $D(\hat{V}_i)$ is the distance from the watermarked vertex $\hat{V}_i$ to its neighborhood, calculated according to (5). The array $\mathcal{B}$ is made up by choosing sets of vertices and their neighborhoods in a certain order with the condition that a newly selected vertex V_i is not part of any previously selected vertex neighborhood and does not produce a significant distortion

$$\{V_i \in \mathcal{B} \,|\, \mathcal{N}(V_i) \notin \mathcal{B} \bigwedge D(\hat{V}_i) < T(\hat{V}_i)\} \tag{9}$$

where $D(\hat{V}_i)$ is the distance (5) evaluated after watermarking the vertices, or after predicting the likely distortion produced by watermarking, where $\hat{V}_i$ represents a stego vertex, $i = 1, \ldots, M$ and M is the total number of vertices in the array $\mathcal{B}$ that fulfills the above mentioned conditions. The ordering among the selected vertices is done according to the Euclidean distances between selected vertices and the initial vertex in the set

$$\left\|\mathbf{V}_{(k-1)} - \mathbf{V}_{(1)}\right\|^2 < \left\|\mathbf{V}_{(k)} - \mathbf{V}_{(1)}\right\|^2 \tag{10}$$

where the array of vertices $V_{(k)} \in \mathcal{B}$ is ranked for $k = 1, \ldots, M$. Orderings of discrete data have been considered in algorithms using order statistics [7]. Fig. 1 shows how a set of vertices is selected, from a graph representing a mesh, according to the conditions (7), (9) and ordered according to increasing Euclidean distance to the head vertex (8), (10).

The selection and ordering of vertices in $\mathcal{B}$ according to the above procedure is obviously invariant to translation, rotation and uniform scaling of the 3-D graphical model. Certain 3-D graphical objects are made up from several meshes. Meshes can be watermarked individualy or jointly. In order to increase the robustness to 3-D object cropping, the given code is embedded repeatedly into various compact target areas of the mesh [28], [32] that corresponds to a local arrangement of vertices [13]. The mesh surface modeling the object, can be split into several areas, each defining a separate cover media. Such a split can be done based on the geometrical proximity of vertices, by clustering their 3-D location coordinates [7] and by taking into account their connectivity. Each segmented part of the mesh must embed the given code at least once. A code of B bits will then be redundantly embedded using localized and ordered sub-arrays of $\mathcal{B}$. The array of bit-holder vertices $\mathcal{B}$ is split into groups of B sets $\{V_i, \mathcal{N}(V_i)\}$, each group located in the same geometrical proximity, defined according to (10). The watermark is repeatedly embedded into the object a number of times equal to $\lfloor M/B \rfloor$. The selection and ordering procedure described above depends exclusively onto the geometric locations of vertices. Consequently, the bit-holder selection procedure will resist any change in the vertex order in the polygonal mesh description file. Shape perturbation by noise is studied in the following section.

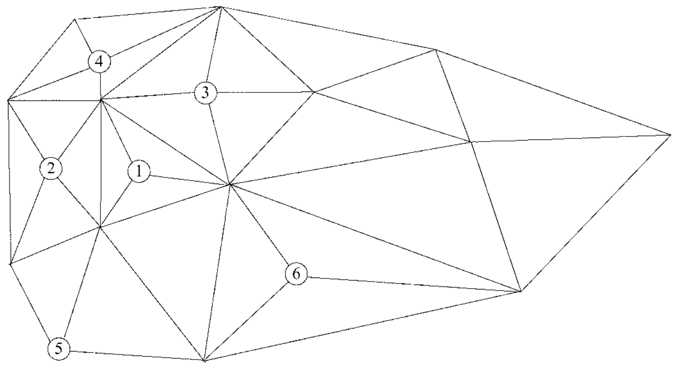


Fig. 1. Exemplification of selecting and ordering a set of vertices from a mesh-based representation of a surface. Numbers indicates the vertex rank in the chosen array for watermarking.

IV. Analyzing the Effect of Perturbations on Meshes

A large variety of mesh surfaces can be modeled by using mixtures of Gaussian functions. Gaussian mixtures have been used for modeling 3-D objects in volumetric images [7]. In this application a mesh is considered parameterized with respect to the 2-D plane. A general surface height $F(V)$ can be represented as the sum of a set of Gaussian functions as follows:

$$F(V) = \lambda \sum_{i=1}^{R\times R} \exp\left[-\frac{(\mathbf{V}-\mathbf{c}_i)^T(\mathbf{V}-\mathbf{c}_i)}{S^2}\right] \tag{11}$$

where λ is a normalization constant, $\mathbf{c}_i$ is the Gaussian function center, S^2 is the variance considered identical for all Gaussian components and $\mathbf{V}$ is the vertex parameter vector, chosen equally spaced on a plane in the following range:

$$\mathbf{V} = \{1, \ldots, R\lfloor 100/R \rfloor\} \times \{1, \ldots, R\lfloor 100/R \rfloor\}. \tag{12}$$

The function $F(V)$ is represented as a surface made up of bumps, each bump representing a Gaussian component. The surface is more or less smoother depending on the number of Gaussian functions, on their parameters and on the number of vertices from the mesh. The normalization parameter λ is chosen such that the surface height $F(V)$ is bounded to the interval $[0, 20]$. An equal number of Gaussian components R is considered along each axis with their centers equally spaced. A large variety of different surfaces is obtained for $S = [4, 16]$

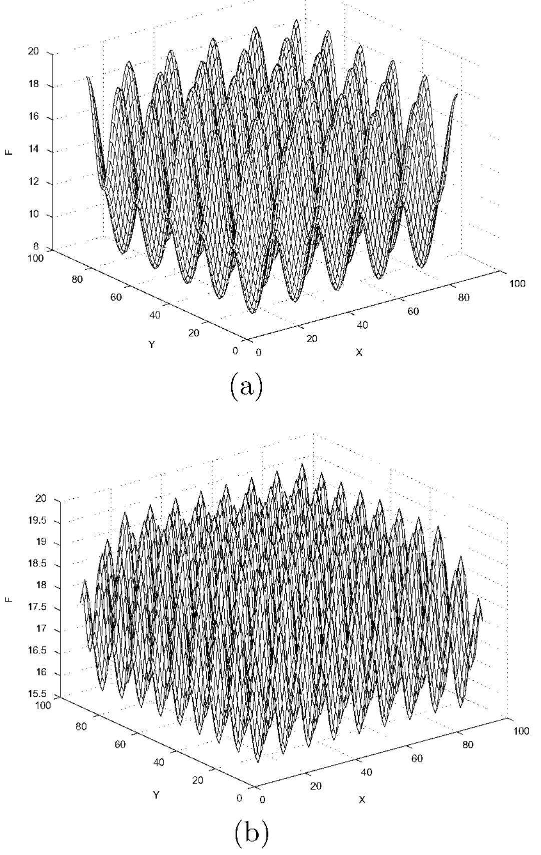


Fig. 2. Examples of surfaces modeled as mixtures of Gaussians. (a) Mesh surface for $R = 6$ and $S = 8$. (b) Mesh surface for $R = 10$ and $S = 6$.

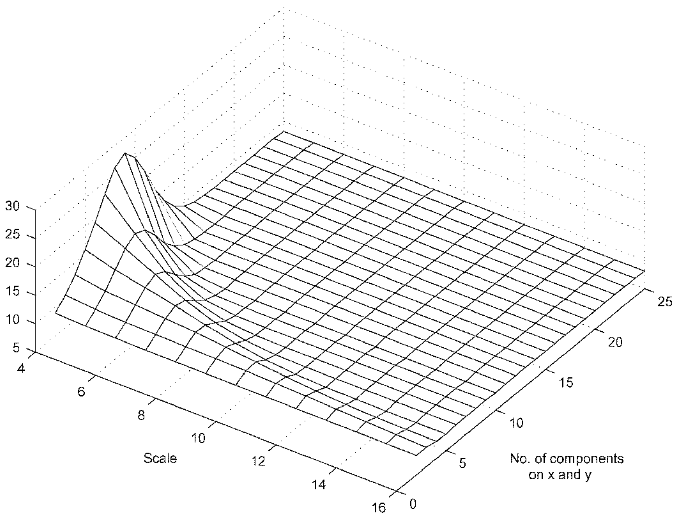


Fig. 3. Evaluation of average $D(V_p)$ for various S and R.

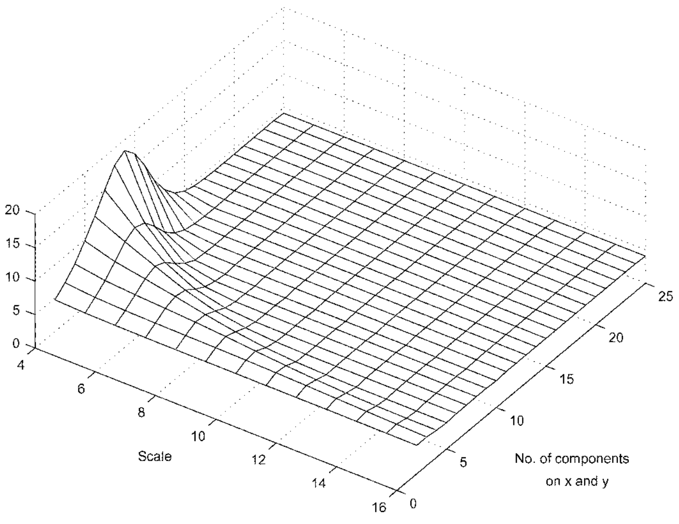


Fig. 4. Evaluation of the local surface average $A(V_i)$ for various S and R.

and $R = \{2, \ldots, 25\}$. Examples of surfaces, generated according to (11) are shown in Figs. 2(a) and (b) and 5(a), for three different pairs of R and S. A set of vertices denoted by V_p is chosen through subsampling the mesh along each axis by two. The neighborhood size is $N_i = 8$ for all selected vertices. Distances $D(V_p)$ are calculated from each selected vertex to its 8-vertex neighborhood, for selected vertices, by using (5).

The average distance $D(V_p)$, for all selected vertices to their neighborhoods, for the surfaces defined by a large variety of S and R is plotted in Fig. 3. In Fig. 4, an approximation of the average surface $A(V_i)$ is displayed. The similarity, up to a multiplication constant, between the two plots from Figs. 3 and 4 is evident. The average surface connected to a selected vertex is larger for a small scale parameter S, when given a specific number of Gaussian components $R \times R$. Larger variations in the mesh surface are obtained when having few Gaussians of small variance.

Surface perturbations are produced onto the surface modeled by $F(V)$ from (11) and represented by the mesh defined by (12), along mesh height direction, in selected vertices. The perturbation induced in the given surface is represented as a fraction from the local distances provided by (5)

$$F(\hat{V}_p) = F(V_p) + \eta D(V_p) \tag{13}$$

where $\hat{V}_p$ is the perturbed vertex, M is the total number of perturbed vertices, and $\eta \in (0, 1)$ is the ratio of the distance from the chosen site to its neighborhood $D(V_p)$. In Fig. 5(a), the surface modeled by $R \times R = 3 \times 3$ Gaussian components is shown when $S = 23$. The surface from Fig. 5(a) is shown after being perturbed by (13) in Fig. 5(b) and (c), when considering $\eta = 0.02$ and $\eta = 0.2$, respectively. From Fig. 5(c), it can be observed that the perturbation produced by $\eta = 0.2$ causes significant distortions to the original surface.

In order to assess shape distortions produced by (13), a large set of surfaces is considered, by varying S and R. The distortion is calculated using

$$E = \frac{1}{M} \sum_{p=1}^{M} |D(\hat{V}_p) - D(V_p)| \tag{14}$$

where $D(\hat{V}_p)$ is the distance from the perturbed vertex to its neighborhood, and M is the total number of perturbed vertices. The distortion plots for $R \times R = \{3 \times 3, 6 \times 6, 10 \times 10\}$ components, when assuming various values for S, and for a perturbation range of $\eta = [0.02, 0.2]$ are represented in Fig. 6(a)–(c), respectively. From these plots, we can observe that distortions are higher for smaller S and for a larger number of components

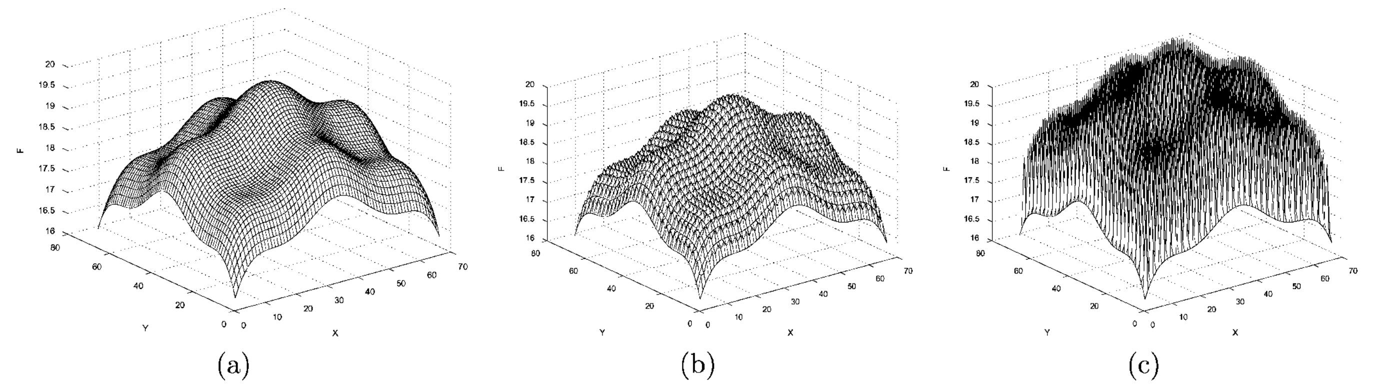


Fig. 5. Effect of perturbations in a mesh surface modeled by a mixture of Gaussians with $R = 3, S = 23$. (a) Original mesh surface. (b) Perturbed mesh surface for $\eta = 0.02$. (c) Perturbed mesh surface for $\eta = 0.2$.

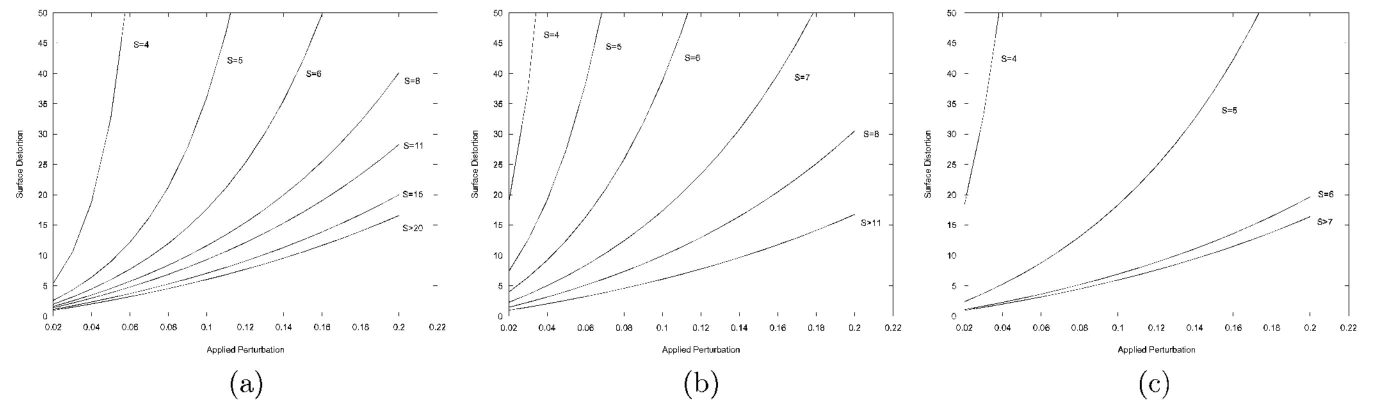


Fig. 6. Distortions produced by noise in various mesh surfaces. (a) Surface modeled by 3 × 3 components. (b) Surface modeled by 6 × 6 components. (c) Surface modeled by 10 × 10 components.

R. Also, we can remark that distortions caused by perturbations in mesh surfaces are predictable. The variation of such distortions with the applied perturbation is almost linear for certain perturbation intervals and for surfaces that tend to be smooth.

V. Embedding and Detecting the Watermark

The information to be embedded is represented as a sequence of bits generated according to a key. This binary sequence is embedded in an ordered array of B vertices and their neighborhoods, $\{V_i, \mathcal{N}(V_i)\}$, extracted from the 3-D graphical object as described in Section III. A relationship that is invariant to translation, rotation and scaling can be defined between each selected vertex V_i and its neighborhood $\mathcal{N}(V_i)$, as it was observed in [13] for various geometrical primitives. The volume associated with each selected site $\{V_i, \mathcal{N}(V_i)\}$ is split into two regions: one for embedding a bit of $\mathbf{0}$, and the other for embedding a bit of $\mathbf{1}$. This corresponds to a local requantization when watermarking. The information provided by the watermark is embedded in the graphical object structure by performing a succession of small perturbations at the locations of chosen bit-holding vertices $V_i \in \mathcal{B}$. In the following it is assumed that not all the vertices from the set $\{V_i, \mathcal{N}(V_i)\}$ are onto the same plane. However, if this is the case, small perturbations are induced in the vertices $V_j \in \mathcal{N}(V_i)$, such that the neighborhood $\mathcal{N}(V_i)$ can be described by a relationship as that from (4). The study of the perturbation effects in mesh surfaces is provided in the previous Section. The embedding algorithms have been designed such that they result in a minimal distortion in the 3-D graphical object structure and provide invariance to geometrical transformations. The boundary separating the two regions, embedding a bit of $\mathbf{0}$, and of $\mathbf{1}$, respectively, depends only on the neighborhood's $\mathcal{N}(V_i)$ vertices location. This procedure ensures robustness to scaling. Two different 3-D mesh embedding approaches are described in the following. Both apply perturbations in the 3-D mesh structure but they use different local geometry modeling. They are called parallel planes and bounding ellipsoids, depending on the way how the local neighborhood is modeled and how the embedding takes place.

A. Parallel Planes Embedding

The first approach defines two parallel planes for the region associated to a cover vertex V_i, using the geometry of its neighborhood $\mathcal{N}(V_i)$. In the first step, the normal $\overrightarrow{\mathbf{N}}_j$ at each vertex from the neighborhood, $V_j \in \mathcal{N}(V_i)$, is calculated. A surface normal is calculated for each polygon as the vector product of the orientations of two of its edges divided by the vector length. The surface normal $\overrightarrow{\mathbf{N}}_i$ 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 parallel planes is denoted by $\overrightarrow{\mathbf{Q}}(V_i)$ and is given by averaging the orientations of all surface normals from that neighborhood

$$\overrightarrow{\mathbf{Q}}(V_i) = \frac{\sum\limits_{V_j \in \mathcal{N}(V_i)} \overrightarrow{\mathbf{N}}_j}{N_i} \tag{15}$$

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 $\boldsymbol{\mu}_i$, derived in (2), 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 $\overrightarrow{\mathbf{Q}}(V_i)$

$$\Psi^2(V_i) = \frac{\sum_{V_j \in \mathcal{N}(V_i)} [(\mathbf{V}_j - \boldsymbol{\mu}_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)]^2}{N_i} \tag{16}$$

where $|\cdot|$ denotes the scalar product. This measure ensures the robustness to scaling for the information embedding regions. In the case when embedding a $\mathbf{1}$ bit, the vertex V_i is projected along the direction of $\overrightarrow{\mathbf{Q}}(V_i)$, inside the volume defined by the parallel planes such that

$$|(\hat{\mathbf{V}}_i - \boldsymbol{\mu}_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)| < \Psi(V_i) - \epsilon \tag{17}$$

where $\hat{\mathbf{V}}_i$ is the new location of the watermarked vertex (stego vertex) and ϵ is a small distance ensuring the robustness to various potential attacks. When embedding a $\mathbf{0}$ bit, the vertex is projected outside the volume defined by the parallel planes

$$|(\hat{\mathbf{V}}_i - \boldsymbol{\mu}_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)| > \Psi(V_i) + \epsilon. \tag{18}$$

If relationships (17) and (18) are already fulfilled according to the selected string of neighborhoods and their corresponding bits, no updating is necessary.

The minimal distortion embedding condition is ensured by using the local gradient to the surface that models the embedding region, as the direction of change. The embedding rule in the case of bounding planes consists of moving the chosen vertex along the parallel plane normal. When embedding a bit of $\mathbf{1}$, in the case when the vertex initially fulfills (18), the updating rule is given by

$$|\hat{\mathbf{V}}_i - \mathbf{V}_i| = [-\Psi(V_i) + \epsilon + (\mathbf{V}_i - \mu_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)] \frac{\overrightarrow{\mathbf{Q}}(V_i)}{||\overrightarrow{\mathbf{Q}}(V_i)||} \tag{19}$$

while for embedding a bit of $\mathbf{0}$, when the vertex initially fulfills (17), the updating rule is

$$|\hat{\mathbf{V}}_i - \mathbf{V}_i| = [\Psi(V_i) + \epsilon - (\mathbf{V}_i - \mu_i) \cdot \overrightarrow{\mathbf{Q}}(V_i)] \frac{\overrightarrow{\mathbf{Q}}(V_i)}{||\overrightarrow{\mathbf{Q}}(V_i)||}. \tag{20}$$

The embedding rules of (19) and (20) ensure a minimal local distortion in the graphical object by using a direction of change that is parallel with the dual planes normal $\overrightarrow{\mathbf{Q}}(V_i)$. It can be observed that the robustness of the watermark is improved by choosing a larger ϵ. On the other hand, a larger ϵ may lead to visible artifacts in the 3-D graphical object.

B. Bounding Ellipsoids Embedding

The second embedding approach consists of defining bounding ellipsoids similar to those describing the neighborhoods in (4), for each selected vertex $V_i \in \mathcal{B}$. Ellipsoids model second-order moment representations and have suitable invariance properties to affine transformations [30], [31]. Ellipsoids have been used for segmenting 3-D objects in volumetric images [7], while ellipses have been considered for watermarking in the DCT domain [5] or in the image graylevel domain [4]. When embedding a $\mathbf{1}$ bit, the vertex V_i should be located inside the bounding ellipsoid such that it fulfills

$$(\hat{\mathbf{V}}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\hat{\mathbf{V}}_i - \boldsymbol{\mu}_i) < K - \varepsilon \tag{21}$$

where $\boldsymbol{\mu}_i$ is the center of the ellipsoid as obtained in (2), $\boldsymbol{\Sigma}_i$ defines its shape, consistent with the second-order moment for the local neighborhood, according to (3), and K provides the extent of the bounding region and ε is a small distance, ensuring the robustness to noise. K depends on a normalization constraint in order to ensure the robustness to scaling. A bit $\mathbf{0}$ is embedded when the vertex V_i is located just outside its corresponding bounding ellipsoid, by fulfilling

$$(\hat{\mathbf{V}}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\hat{\mathbf{V}}_i - \boldsymbol{\mu}_i) > K + \varepsilon. \tag{22}$$

If relationships (21) and (22) are already fulfilled by the neighborhoods $\mathcal{N}(V_i)$, according to their corresponding bits, no change will be performed. Otherwise, the cover vertices $\mathbf{V}_i$ will be moved into stego vertices $\hat{\mathbf{V}}_i$ according to their neighborhood geometry and the given information to be embedded.

In the following, the updating equations are provided for the bounding ellipsoids method. The minimal distortion vertex updating direction for watermarking is perpendicular onto the bounding surface. The normal to the bounding ellipsoid at a location $\mathbf{x}$, denoted by $\overrightarrow{\Gamma}(\mathbf{x})$, is obtained after differentiating equation (4)

$$\overrightarrow{\Gamma}(\mathbf{x}) = 2(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}. \tag{23}$$

The updating equation for the vertex location when embedding a bit of $\mathbf{1}$, when the cover vertex V_i is located according to (22), is given by

$$\hat{\mathbf{V}}_i = \mathbf{V}_i - \frac{\overrightarrow{\Gamma}(V_i)}{||\overrightarrow{\Gamma}(V_i)||} [(\mathbf{V}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{V}_i - \mu_i) - K + \varepsilon] \tag{24}$$

while for embedding a bit of $\mathbf{0}$, when the cover vertex V_i is located according to (21), has the following updating direction:

$$\hat{\mathbf{V}}_i = \mathbf{V}_i - \frac{\overrightarrow{\Gamma}(V_i)}{||\overrightarrow{\Gamma}(V_i)||} [(\mathbf{V}_i - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{V}_i - \boldsymbol{\mu}_i) - K - \varepsilon]. \tag{25}$$

No change is performed if the vertices already fulfill (21) or (22), according to their corresponding bits. In Fig. 7(a), a mesh is shown with three selected cover vertices highlighted. In Fig. 7(b), the final location of stego vertices is shown when

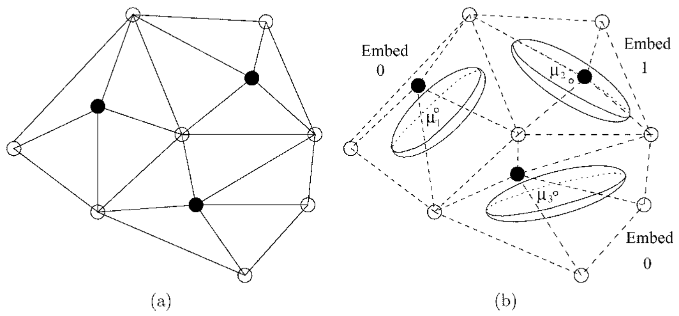


Fig. 7. Embedding information in polygonal meshes using bounding ellipsoids. (a) Initial mesh. (b) Embedding the code $\mathbf{010}$.

embedding the code $\mathbf{010}$ by using the bounding ellipsoids algorithm.

C. Three-Dimensional Watermarking System

The local mesh topology does not change when embedding digital information using either of the two 3-D graphics watermarking algorithms. However, the location of certain vertices is changed according to the embedding rules outlined above. This may influence the synchronization between the constraints embedded in the local vertex geometry $\mathcal{B}$ and the given bit array B. The change in vertex location is predictable according to the study provided in Section IV. The vertex order, that has been calculated in the cover vertices according to (10), may be changed by the perturbations caused in stego vertices by watermarking. In this situation, additional perturbations have to be applied onto the vertices from the neighborhoods $V_j \in \mathcal{N}(V_i)$. These changes are performed such that the ordering of stego vertices is maintained after watermarking, while the entire information is eventually contained in the stego mesh surface. The threshold $T(V_i)$ from (7) is necessary for selecting the right number of watermarked vertices according to (9). It may be necessary to recalculate this threshold in the detection stage, according to (7). The vertex neighborhood ordering done according to the procedure explained in Section III could be affected only by relatively large noise perturbations. The main processing blocks for watermarking/fingerprinting 3-D mesh surfaces are displayed in Fig. 8. A feedback loop is used in this flowchart, similar with that employed in the 3-D watermarking algorithm from [11], in order to ensure that the ordering of stego-vertices remains unchanged after the embedding is done.

D. Three-Dimensional Watermark Detection

The watermark detection stage aims to recover the information that has been embedded in the 3-D shape. The proposed watermarking method is blind and does not require the original cover media for the detection proposes. The detection of the information embedded in 3-D graphical objects is done using stego vertex ordering and local geometric constraints. Two different watermark detection approaches can be considered: one where each individual bit is distinctly retrieved from the shape, and the second one consisting of the statistical modeling of the decoded bits distributions. In the first method, the embedding steps are traced backward. First, the array of marked vertices and their neighborhoods is selected and ordered in the same way as it was explained in Section III. For the chosen vertices and their

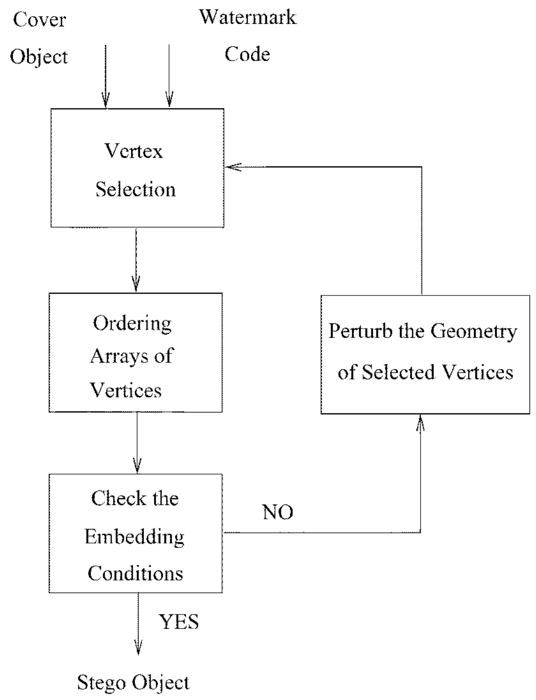

Cover Object
Watermark Code
Vertex Selection
Ordering Arrays of Vertices
Perturb the Geometry of Selected Vertices
Check the Embedding Conditions
NO
YES
Stego Object

Fig. 8. Main processing blocks for the 3-D watermarking scheme.

neighborhoods, when the parallel planes embedding has been used, the relationship (17) is verified for detecting a $\mathbf{1}$ bit and (18) for a $\mathbf{0}$ bit. On the other hand, when embedding was done using ellipsoidal bounding volumes, relationships (21) and (22) are considered for detecting a $\mathbf{1}$ bit and a $\mathbf{0}$ bit, respectively. Thus, a sequence of bits is retrieved from the shape. In order to improve the robustness to errors in the detected bit-sequence from the given watermarked 3-D shape, error correction codes can be used. Error correction codes are widely used in communication systems [34]. An example of error correction code is the Hamming code which checks for the parity of bits [35]. Such a watermark retrieval approach can be adopted for steganography or fingerprinting applications. In the statistical modeling of decoded bits approach, a distribution of all the distances from the stego vertex locations to their neighborhood representations, is performed. The resulting distribution can be approximated with two Gaussian probability density functions, with the means offset with ϵ and $-\epsilon$, or with ε and $-\varepsilon$, depending on the embedding method and corresponding to the embedding of a $\mathbf{1}$ bit and a $\mathbf{0}$ bit, respectively. The likelihood of the presence of a certain code can be statistically estimated from these distributions [1]. Two types of errors may occur: false rejection error when the embedded watermark is not detected, and false detection error, when a watermark is detected without ever being embedded in the given 3-D shape. It is expected that the false detection error is negligible.

VI. Experimental Results

The watermarking methodology described in this paper can be applied to a large variety of 3-D shapes and graphical objects described by meshes. Such objects can be created by using a graphical package, they can be extracted from 2-D images (using shape-from-shading, shape-from-texture), or from 3-D

TABLE I
CHARACTERISTICS OF THE GRAPHICAL OBJECTS USED IN EXPERIMENTS

3-D Model	No. of Vertices	No. of Polygons	No. polygons connected to a vertex	Bounding Ellipsoids		Parallel Planes	
				No. of Stego Vertices	No. of Embeddings	No. of Stego Vertices	No. of Embeddings
Dog	654	1286	2.0	183	3	113	2
Fan	1532	2634	1.7	275	5	199	3.5
Guillotine	2723	4578	1.7	451	8	307	5
Screwdriver	2073	4076	2.0	280	5	203	3.5
Sink	674	1068	1.6	98	2	66	1

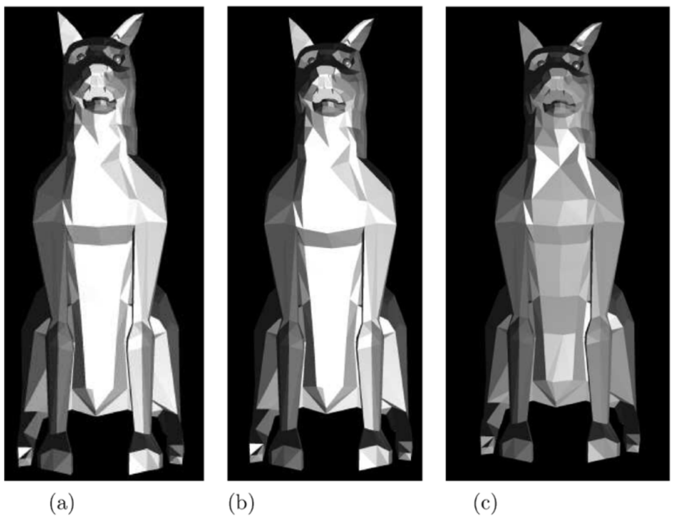


Fig. 9. Graphical object representing a dog. (a) Original. (b) Watermarked using bounding ellipsoids. (c) Watermarked using parallel planes.

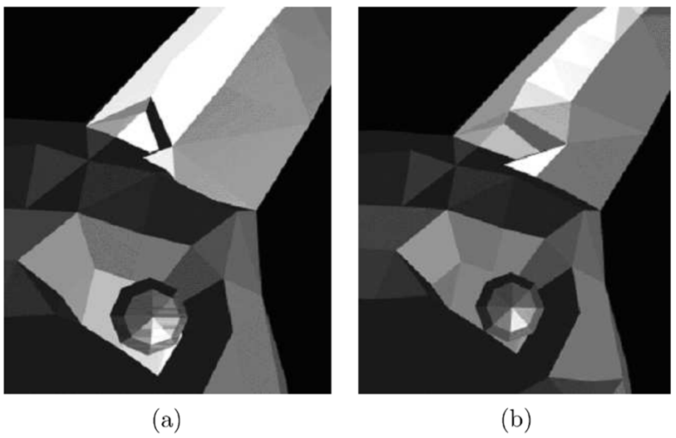


Fig. 10. Detail of graphical object representing a dog. (a) Original. (b) Watermarked using parallel planes.

images (volumetric images such as those acquired by MRI or computer tomography). In the following, the proposed 3-D watermarking techniques are applied on five shapes. The graphical objects are in 3D-Studio format providing a mesh-based representation, by giving locations of vertices and their connectivity, listing the polygons and their vertices. The 3-D objects under consideration represent both animation characters and computer aided design (CAD) objects and some of them are made up from more than one mesh. The five objects, listed in Table I, are called: "Dog," "Fan," "Guillotine," "Screwdriver," and "Sink." In the first four columns of Table I, the 3-D graphical object names are provided together with the characteristics of their meshes: number of vertices, polygons, and the average number of polygons connected to a vertex. The original graphical objects are shown as follows: "Dog" in Fig. 9(a), "Fan" in Fig. 11(a), "Guillotine" in Fig. 12(a), "Screwdriver" in Fig. 14(a), and "Sink" in Fig. 15(a).

A watermark code of 32 bits is generated using cyclic redundancy check of a particular bit-string and is represented as a sequence of eight chains, 4 bits each. Cyclic redundancy checks have the property of mapping evenly across the space of possible values. The information to be embedded for each chain is seven bits long, where four bits represent the watermark code and the rest are used as error correction bits. Hamming correction codes have been used [35]. One bit-error can be detected and corrected in each chain using three error correction bits. The given code is embedded in the 3-D objects according to the methodology described in Sections III and V. The watermarking algorithms proposed in this paper are blind, in the sense that in the detection stage they require only the stego mesh for recovering the embedded code. In the detection stage every embedded bit is retrieved from the stego object and, for testing purposes, a XOR operation is eventually performed with the watermark code [5]. This shows the number of bit errors. A watermark code detection decision is taken based on an acceptable minimal bit error [32].

In Figs. 9(b), 11(b), 12(b), 14(b), and 15(b), the given five objects are shown after being watermarked using bounding ellipsoids algorithm, while in Figs. 9(c), 11(c), 12(c), 14(c), and 15(c), they are represented after being watermarked using the parallel planes algorithm. For the sake of visualizing the distortions caused by watermarking, details of four of the original graphical objects are enlarged and displayed in Figs. 10(a), 13(a), 16(a), and 17(a), respectively. Details of the "Dog" and "Sink" shapes watermarked by the parallel planes algorithm are displayed in Figs. 10(b) and 17(b). The details of the watermarked "Fan" and "Screwdriver" by using bounding ellipsoids

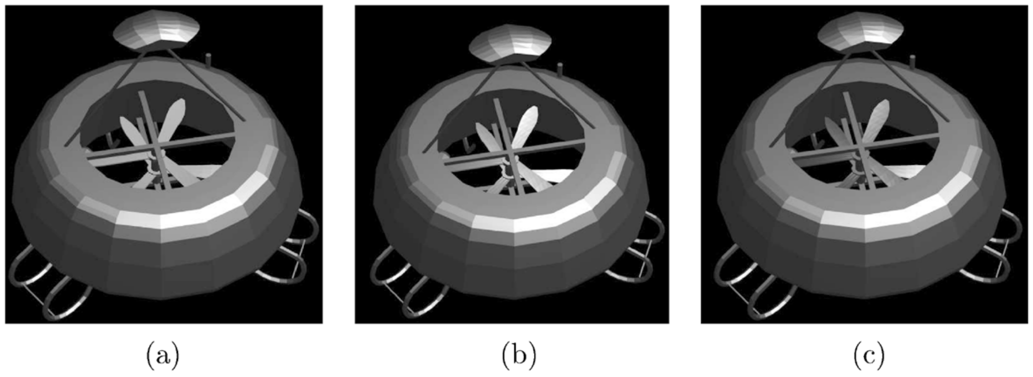


Fig. 11. Graphical object representing a fan. (a) Original. (b) Watermarked using bounding ellipsoids. (c) Watermarked using parallel planes.

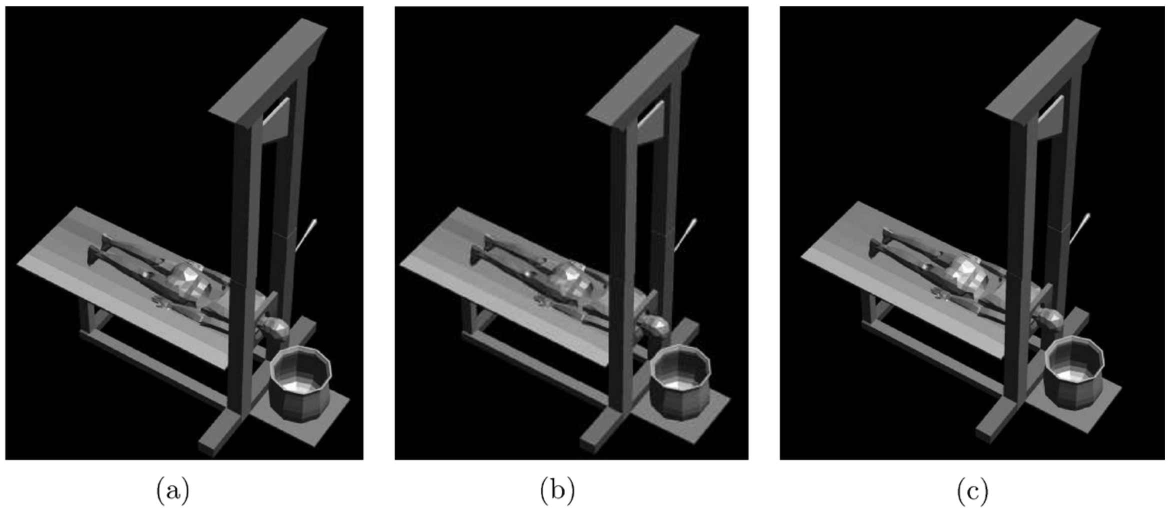


Fig. 12. Graphical object representing a guillotine. (a) Original. (b) Watermarked using bounding ellipsoids. (c) Watermarked using parallel planes.

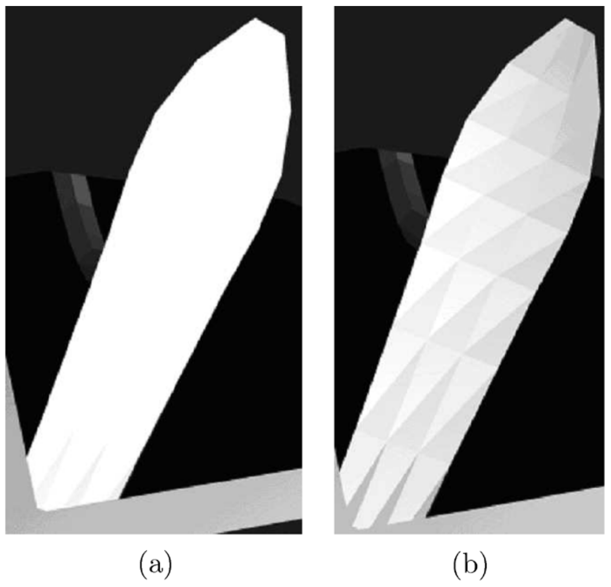


Fig. 13. Detail of graphical object representing a fan. (a) Original. (b) Watermarked using bounding ellipsoids.

embedding are shown in Figs. 13(b) and 16(b), respectively. The stego objects are not highly distorted by watermarking when compared to the original cover objects. Due to the fact that many polygons are rather small and by using the proposed criteria for selecting vertices for watermarking, according to the procedure from Section III, most modifications caused by watermarking can be observed only in the detail images of 3-D meshes. Distortions in the upper torso, right ear and neck of the "Dog" caused by the parallel planes embedding algorithm can be observed in Figs. 9(c) and 10(b). The parallel planes algorithm causes distortions in the large polygons from the bottom of the "Sink" as it can be observed in Fig. 15(c). Modifications caused by the bounding ellipsoids watermarking algorithm can be observed in the detail showing the top edge of the "Screwdriver" handler from Fig. 16(b). From all these examples, it can be observed that distortions caused by the parallel planes watermarking algorithm are larger than those caused by the bounding ellipsoids algorithm. All the graphical objects are displayed under general visibility conditions, using simple illumination models and flat shading [6]. Flat shading, unlike interpolative shading algorithms, provides an easy identification tool for any change in the mesh geometry, highlighting artifacts caused by watermarking [33]. When applying textures, colors, more sophisticated illumination models or shading algorithms such as Gouraud or Phong [6], the geometrical distortions become less visible. Such techniques that are used in computer graphics for scene representation, can contribute to the watermark masking.

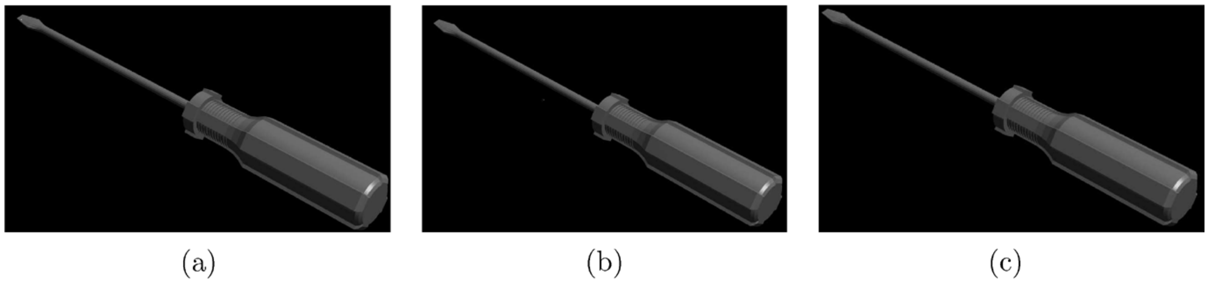


Fig. 14. Graphical object representing a screwdriver. (a) Original. (b) Watermarked using bounding ellipsoids. (c) Watermarked using parallel planes.

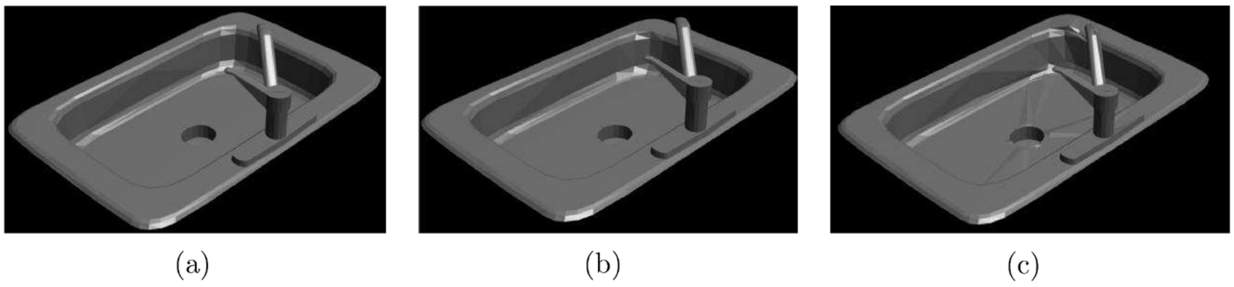


Fig. 15. Graphical object representing a sink. (a) Original. (b) Watermarked using bounding ellipsoids. (c) Watermarked using parallel planes.

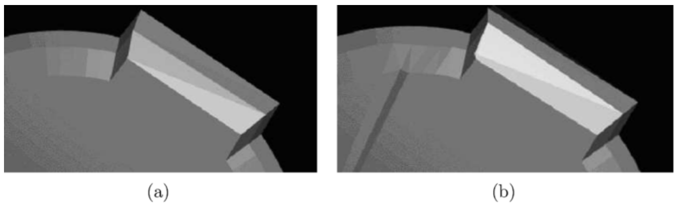


Fig. 16. Detail of graphical object representing a screwdriver. (a) Original. (b) Watermarked using bounding ellipsoids.

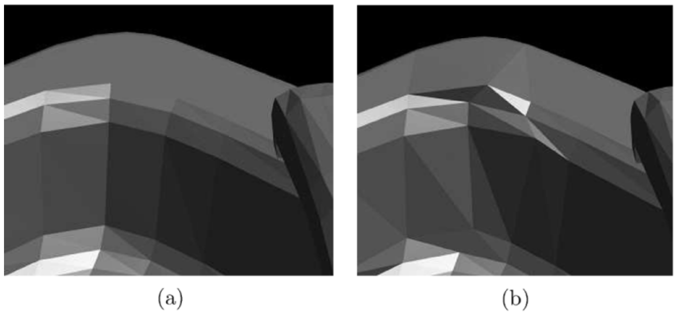


Fig. 17. Detail of graphical object representing a sink. (a) Original. (b) Watermarked using parallel planes.

A very important issue in steganography, fingerprinting and in general in information hiding is the capacity of embedding, representing how many bits can be embedded in a given graphical object. This information is provided in Table I for the five objects when using each of the two 3-D watermarking algorithms. From Table I, it can be observed that more bit holder vertices can be found in graphical objects when using the bounding ellipsoid algorithm than the parallel planes algorithm. As it can be observed from the figures of 3-D stego graphical objects, the parallel planes watermarking method causes larger distortions in the mesh structure when compared to the bounding ellipsoids. This leads to an increase in the number of vertices that cannot be watermarked due to the high likelihood of large distortion after watermarking the 3-D graphical object. Alterations of the vertex ordering (10) are less likely to happen in the case of bounding ellipsoids. Generally, shapes of manufactured objects, such as "Fan," "Screwdriver," and "Sink" have a lower capacity to store information when compared with other objects. Such objects, used in CAD, tend to have large flat surfaces that do not fulfill the bit holder selection conditions outlined in Section III. The number of repetitions for the 32-bit codes that can be embedded in each object is provided in Table I for each watermarking method. It can be observed that the watermark can be repeatedly embedded at least a few times in the structure of each 3-D graphical object. This provides an increased robustness to cropping. If no error correction bits would have been used, the watermark could have been embedded for a larger number of times. However, the watermark robustness to other distortions than cropping would have decreased in that case.

It is trivial to show that the proposed watermarking methodology is unaffected by rotation, translation and scaling of the objects. Both, the selection and ordering of cover vertices described in Section III and the local embedding rules outlined in Section V are invariant to these transformations or their combination. Experiments have found no false detection probabilities in the given set of objects, when considering a large set of different 32-bit codes.

In the following, the results when testing the watermark robustness at two different attacks are described. The first attack is topological and consists of extracting meshes or cropping specific regions from the stego graphical objects. In this experiment only 3-D graphical objects watermarked by bounding ellipsoids have been considered. Two distinct meshes are extracted from the "Guillotine" stego graphical object, displayed in Fig. 12(b). The two meshes, called "Basket" and "Person," are shown in Fig. 19(a) and (b), respectively. The "Dog" stego graphical object, displayed in Fig. 9(b), has been sliced in the region of the neck.

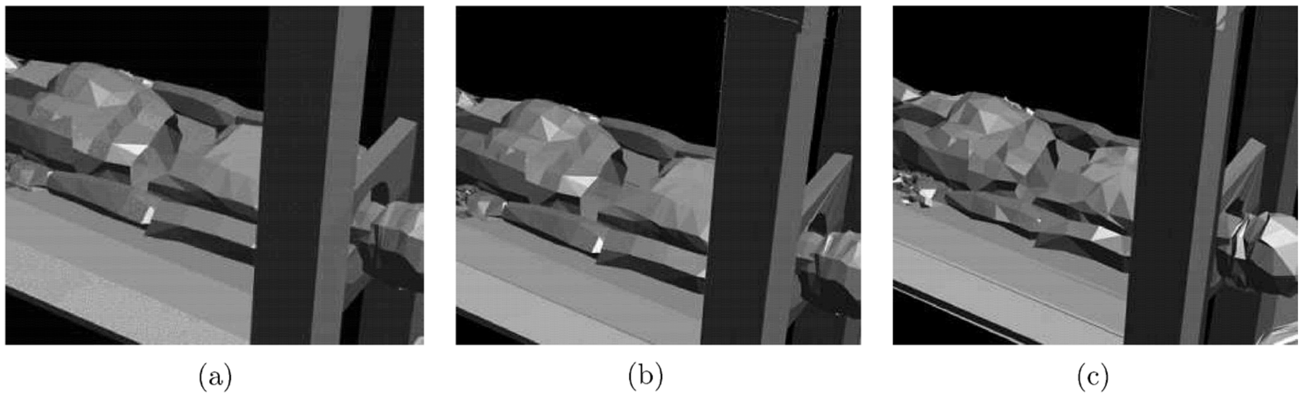
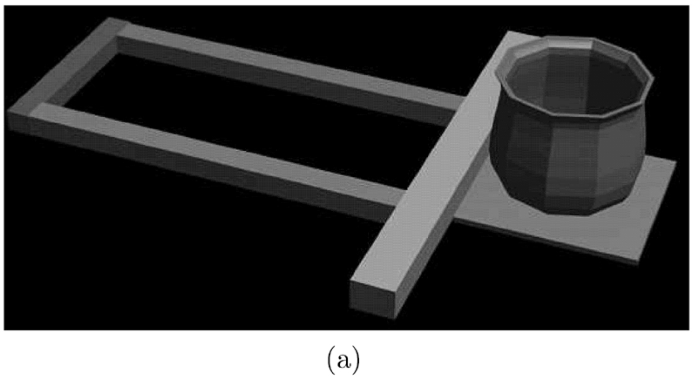


Fig. 18. Perturbations produced by Gaussian noise shown on a detail from the“Guillotine” graphical object. (a) Original detail. (b) Watermarked object corrupted by Gaussian noise corresponding to $E = 0.15$. (c) Watermarked object corrupted by Gaussian noise corresponding to $E = 0.75$.



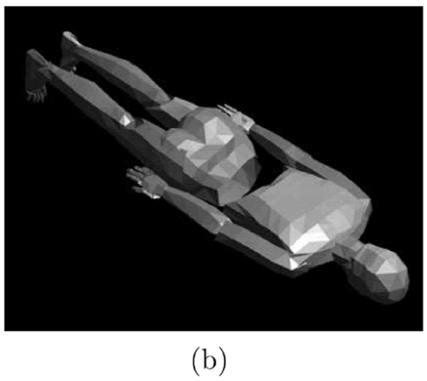


Fig. 19. Meshes extracted from the “Guillotine” graphical object. (a) “Basket.” (b) “Person.”

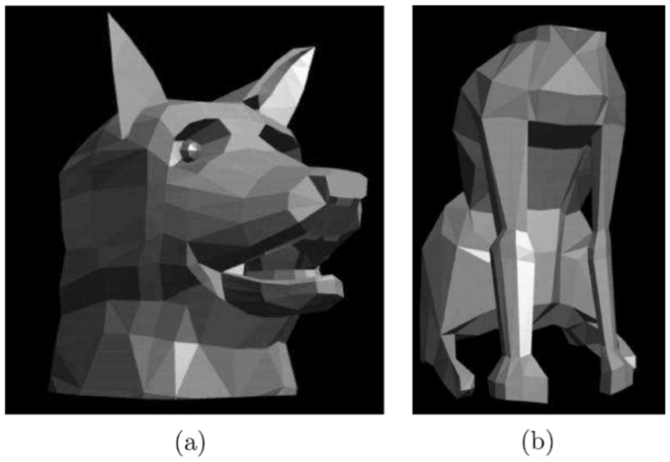


Fig. 20. Cropping of the “dog” model after being watermarked by the bounding ellipsoids algorithm. (a) “Dog-head.” (b) “Dog-body.”

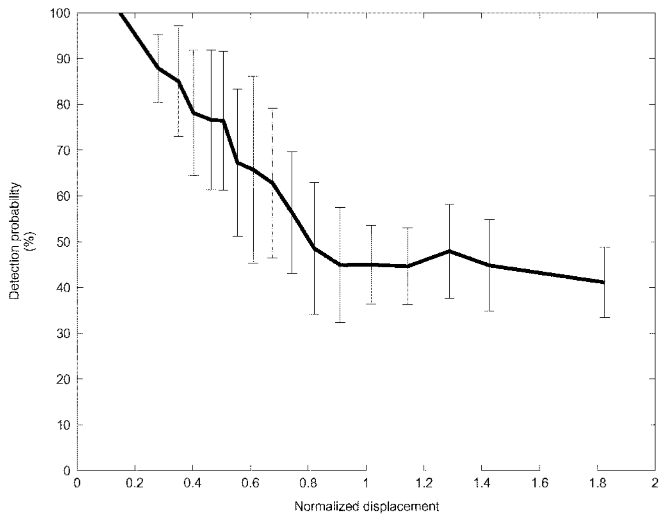
TABLE II
WATERMARK DETECTION RESULTS FROM CROPPED 3-D GRAPHICAL OBJECTS

Cropped Segment	Original 3-D Model	No. of Vertices	No. of Polygons	No. of Detected Stego Vertices	Code Detection Rate (%)
Basket	Guillotine	531	1060	98	87.5
Person	Guillotine	1092	3006	322	100
Dog-head	Dog	389	772	112	100
Dog-body	Dog	265	512	63	50

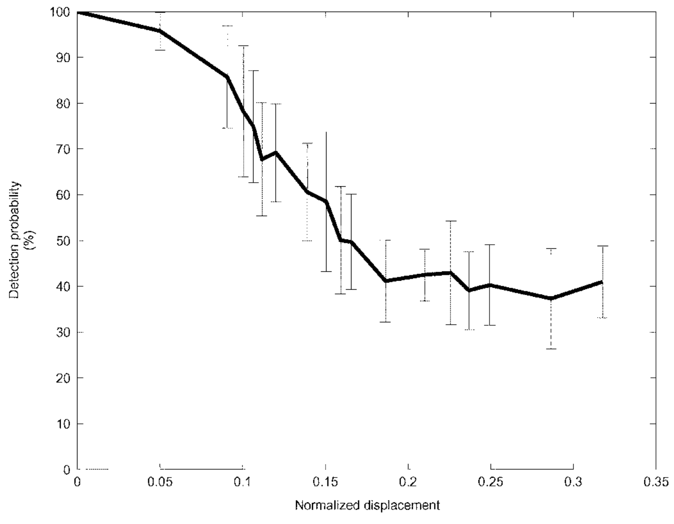
22 new vertices are generated in the region of separation. The new graphical objects “Dog-head” and “Dog-body” are shown in Fig. 20(a) and (b), respectively. The geometrical properties of the meshes extracted or cropped from the 3-D graphical objects are provided in Table II. The number of detected stego vertices and the percentage of the recovered watermark are provided in Table II, as well. Due to the repetition embedding, the watermark was fully recovered from both “Person” and “Dog-head” graphical objects. In the cases of “Basket” and “Dog-body,” the number of information carrying bits recovered exceeded that of the message including the error correction codes. However, the bit error rate is caused in both cases by the loss of identical bits in all the repeatative codes that are recovered.

In another attack, random perturbations have been applied onto the 3-D graphical objects structure. The effect of perturbations in artificially generated meshes was analyzed in Section IV. Dislocation of vertices consists of a general perturbation model that can represent the effect of a large range of possible attacks. The perturbations are modeled according to Gaussian noise $||\check{\mathbf{V}}_i - \mathbf{V}_i|| \sim \aleph(0, \sigma^2)$, where $\check{\mathbf{V}}_i$ represents the location of a distorted vertex by noise. The following measure, called normalized displacement, is used to test the level of distortion:

$$E(\sigma) = \frac{1}{N} \sum_{i=1}^{N} \frac{\sigma^2}{D(V_i)} \tag{26}$$



(a)



(b)

Fig. 21. Robustness results for 3-D watermarking after applying Gaussian noise. (a) Bounding ellipsoids. (b) Parallel planes.

where N represents all the sites in the graphical object and $D(V_i)$ is provided in (5). One hundred noisy versions of each 3-D graphical object have been created after embedding various watermark codes and for different noise variance σ^2, in the case of each watermarking algorithm. The average bit detection rates are shown in the plots from Fig. 21(a) and (b), when calculating the watermark detection from the noisy 3-D watermarked graphical objects by bounding ellipsoids and parallel planes algorithms. Error bars show the standard deviation from the bit detection rate average for the perturbation caused for a certain noise variance. A detail from "Guillotine" object is shown in Fig. 18(a). The same detail is displayed in Fig. 18(b) and (c) after applying noise corresponding to a normalized displacement of $\mathrm{E} = 0.15$ and of $\mathrm{E} = 0.75$, respectively. The distortion levels of the graphical objects, shown in Fig. 18(b) and (c), correspond to the mesh perturbation breaking points for the parallel planes and bounding ellipsoids watermarking algorithms, according to the plots from Fig. 21(b) and (a), respectively. From these figures, it is evident that bounding ellipsoids provide a higher robustness to noise than parallel planes watermarking. Certain 3-D shapes

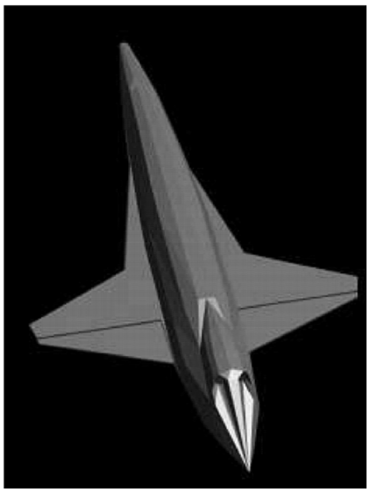


Fig. 22. Three-dimensional graphical model of an aircraft.

cannot be watermarked due to their geometrical properties that render them unsuitable for watermarking. Such a shape, representing an aircraft, is displayed in Fig. 22. This shape is rather simple and contains 265 vertices and 295 polygons. The 53-bit holders that have been found in its structure are not enough to store the given watermark under the conditions set above, including redundant check bits. For perfect decoding, a mesh surface either original or resulted from cropping should contain a certain minimum number of vertices. This number depends on how many vertices can carry information, their neighborhood size, and the error correction code.

Watermarks embedded using the proposed methodology do not affect the mesh topology or the vertex interconnectivity. However, certain changes in the mesh topology can affect the watermark. As it can be observed from Section III, the selection and ordering of vertices depend on local neighborhoods, that rely on the mesh connectivity as defined in (1). Significant changes to the mesh connectivity may affect the vertex selection and ordering. Such changes consist in removing certain vertices and edges, or replacing edges with others by creating new polygons and neighborhoods that are different from those used for embedding. All the shapes used in experiments have a relatively low number of vertices and polygons. Such shapes are usually used in computer graphics and can be obtained from objects represented with a larger density of vertices and polygons after using a polygon reduction algorithm. It is evident that the proposed methodology can be applied in graphical objects with a larger number of vertices and polygons. The number of vertices found suitable for watermarking would be higher in such cases.

In all the experimental results, we have observed robustness with respect to geometrical perturbation caused by noise, cropping and other attacks in the case of both 3-D watermarking algorithms. The watermarking results obtained for art-like 3-D graphical objects are better than those obtained with industrial graphical objects. Industrial objects have fewer and larger surfaces as it can be observed from Table I, and Figs. 9, 11, 12, 14, and 15. Consequently, such objects contain fewer vertex sites suitable for watermarking. The bounding ellipsoid watermarking approach is able to find more bit-holding vertices, produces less visible mesh perturbations and is also more robust to attacks than the parallel planes embedding algorithm in all the experiments that have been performed. The bounding ellipsoids algorithm either collapses vertices toward the center of the neighborhood or moves them slightly away from that center. This creates a rather compact and integrated representation of the local stego geometry that includes the stego vertex and its neighborhood $\{\hat{V}_i, \mathcal{N}(\hat{V}_i)\}$, when compared to the parallel planes 3-D watermarking algorithm.

VII. Conclusion

This paper introduces a new methodology for embedding information in the 3-D shape content. The 3-D shapes are represented as meshes formed by vertices joined by edges and polygons. Two different algorithms are introduced, both embedding controlled nonlinear perturbations in the geometrical structure of the 3-D mesh, without affecting the mesh topology. The proposed watermarking algorithms do not require the original object in the detection stage. The algorithms have two stages. In the first stage, a set of vertices and their neighborhoods are selected and ordered according to a minimal distortion visibility criterion. The embedding procedure consists of localized geometrical changes for selected vertices. A study of the effects of geometrical perturbations in mesh-based representations of surfaces is provided in this paper. Repeatative embedding of the watermark in various 3-D object regions ensures robustness to cropping. Two different regions are created in the local geometry of selected vertices, for embedding distinctly a bit of 1 or of 0, respectively. Two techniques are considered for watermarking, depending on the way how the local neighborhood is modeled: using bounding ellipsoids and parallel planes. The bounding ellipsoids watermarking algorithm uses the first and second-order moments. Both methods are invariant to affine transformations and to a range of other attacks including changing the vertex order in the 3-D object description file. The proposed watermarking algorithms have been applied to both art-like graphical objects and to 3-D models of mechanical objects such as those used in CAD. The capacity of information embedding has been evaluated for the 3-D graphical objects under study. Watermark robustness was tested to vertex perturbation as well as to 3-D graphical object cropping. The watermarks embedded using the bounding ellipsoids algorithm produce a lower level of distortion in the 3-D stego meshes than the parallel planes algorithm. The proposed watermarking methodology has many potential applications including copyright protection, authentication, steganography, object database management, encoding of behavioral patterns for graphical characters, digital cinematography, etc.

References

[1] I. Cox, M. Miller, and J. Bloom, *Digital Watermarking*. New York: Morgan Kaufmann, 2001.
[2] J. Eggers and B. Girod, *Informed Watermarking*. Norwell, MA: Kluwer, 2002.
[3] M. Wu and B. Liu, “Data hiding in image and video: Part I—Fundamental issues and solutions,” *IEEE Trans. Image Process.*, vol. 12, no. 6, pp. 685–695, Jun. 2003.
[4] A. Nikolaidis and I. Pitas, “Region-based image watermarking,” *IEEE Trans. Image Process.*, vol. 10, no. 11, pp. 1726–1740, Nov. 2001.
[5] A. G. Bors and I. Pitas, “Image watermarking using bock site selection and DCT domain constraints,” *Opt. Exp.*, vol. 3, no. 12, pp. 512–522, 1998.
[6] A. Watt, *3-D Computer Graphics*, 3rd ed. Reading, MA: Addison-Wesley, 2000.
[7] A. G. Bors and I. Pitas, “Object classification in 3-D images using alpha-trimmed mean radial basis function network,” *IEEE Trans. Image Process.*, vol. 8, no. 12, pp. 1730–1744, Dec. 1999.
[8] S. Kishk and B. Javidi, “3D object watermarking by 3-D hidden object,” *Opt. Exp.*, vol. 11, no. 8, pp. 874–888, 2003.
[9] E. Garcia and J.-L. Dugelay, “Texture-based watermarking of 3-D video objects,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 853–866, Aug. 2003.
[10] F. Hartung, P. Eisert, and B. Girod, “Digital watermarking of MPEG-4 facial animation parameters,” *Comput. Graph.*, vol. 22, no. 4, pp. 425–435, 1998.
[11] B.-L. Yeo and M. M. Yeung, “Watermarking 3-D objects for verification,” *IEEE Comput. Graph. Appl.*, vol. 19, no. 1, pp. 36–45, Jan. 1999.
[12] C. Fornaro and A. Sanna, “Private key watermarking for authentication of CSG models,” *Comput.-Aided Design*, vol. 32, no. 12, pp. 727–735, 2000.
[13] R. Ohbuchi, H. Masuda, and M. Aono, “Watermarking three-dimensional polygonal models through geometric and topological modifications,” *IEEE J. Sel. Areas Commun.*, vol. 16, no. 4, pp. 551–560, Apr. 1998.
[14] M. G. Wagner, “Robust watermarking of polygonal meshes,” in *Proc. Geometric Modeling and Processing*, Hong Kong, 2000, pp. 201–208.
[15] F. Cayre and B. Macq, “Data hiding on 3-D triangle meshes,” *IEEE Trans. Signal Process.*, vol. 51, no. 4, pp. 939–949, Apr. 2003.
[16] O. Benedens, “Affine invariant watermarks for 3-D polygonal and NURBS based models,” in *Proc. Int. Workshop Information Security*, Wollongong, Australia, 2000, pp. 15–29. *LNCS 1975*.
[17] ——, “Geometry based watermarking of 3-D models,” *IEEE Comput. Graph. Appl.*, vol. 19, no. 1, pp. 46–55, Jan. 1999.
[18] B. Koh and T. Chen, “Progressive browsing of 3-D models,” in *Proc. IEEE Workshop Multimedia Signal Processing*, Copenhagen, Denmark, 1999, pp. 71–76.
[19] T. Harte and A. G. Bors, “Watermarking 3-D Models,” in *Proc. IEEE Int. Conf. Image Processing*, vol. III, Rochester, NY, 2002, pp. 661–664.
[20] E. Praun, H. Hoppe, and A. Finkelstein, “Robust mesh watermarking,” in *Proc. Int. Conf. Computer Graphics and Interactive Techniques*, vol. 6, Los Angeles, CA, 1999, pp. 69–76. (*Computer Graphics*).
[21] K. Yin, Z. Pan, J. Shi, and D. Zhang, “Robust mesh watermarking based on multiresolution processing,” *Comput. Graph.*, vol. 25, no. 3, pp. 409–420, 2001.
[22] O. Benedens and C. Busch, “Toward blind detection of robust watermarks in polygonal models,” in *Proc. EUROGRAPHICS*, vol. 19, Interlaken, Switzerland, 2000, pp. C199–C208. *(Computer Graphics Forum)*.
[23] R. Ohbuchi, H. Masuda, and M. Aono, “A shape-preserving data embedding algorithm for NURBS curves and surfaces,” in *Proc. Computer Graphics Int. Conf.*, Canmore, AB, Canada, 1999, pp. 180–187.
[24] S. Kanai, H. Date, and T. Kishinami, “Digital watermarking for 3-D polygons using multiresolution wavelet decomposition,” in *Proc. Int. Workshop Geometric Modeling: Fundamentals and Applications*, Tokyo, Japan, 1998, pp. 296–307.
[25] S.-H. Yang, C.-Y. Liao, and C.-Y. Hsieh, “Watermarking MPEG-4 2-D mesh animation in multiresolution analysis,” in *Proc. Advances Multimedia Information Processing*, Hsinchu, Taiwan, R.O.C., 2002, pp. 66–73. *LNCS 2532*.
[26] R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama, “Watermarking 3-D polygonal meshes in the mesh spectral domain,” in *Proc. Graphics Interface*, Ottawa, ON, Canada, 2001, pp. 9–17.
[27] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, “A frequency-domain approach to watermarking 3-D shapes,” in *Proc. EUROGRAPHICS*, vol. 21, Saarbrucken, Germany, 2002, pp. 373–382. *(Computer Graphics Forum)*.
[28] F. Cayre, P. Rondao-Alface, F. Schmitt, B. Macq, and H. Maitre, “Application of spectral decomposition to compression and watermarking of 3-D triangle mesh geometry,” *Signal Process.: Image Commun.*, vol. 18, no. 4, pp. 309–319, 2003.
[29] O. Benedens, “Robust watermarking and affine registration of 3-D meshes,” in *Proc. Information Hiding*, Noorwijkerhout, The Netherlands, 2003, pp. 177–195. *LNCS 2578*.

[30] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*. New York: McGraw-Hill, 1965.

[31] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision, Vol. I*. Reading, MA: Addison-Wesley, 1992.

[32] A. G. Bors, "Watermarking 3-D shapes using local moments," in *Proc. IEEE Int. Conf. Image Processing*, vol. I, Singapore, Oct. 24–27, 2004, pp. 729–732.

[33] O. Benedens, J. Dittmann, and F. A. P. Petitcolas, "3D watermarking design evaluation," *Proc. SPIE*, vol. 5020, pp. 337–348, 2003.

[34] J. Baylis, *Error-Correcting Codes*, London, U.K.: Chapman-Hall, 1998.

[35] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, Amsterdam, The Netherlands: North-Holland, 1977.

Adrian G. Bors (M'00–SM'04) received the M.S. degree in electronics engineering from the Polytechnic University of Bucharest, Bucharest, Romania, in 1992, and the Ph.D. degree in informatics from the University of Thessaloniki, Thessaloniki, Greece, in 1999.

From September 1992 to August 1993, he was a Research Scientist with the Signal Processing Laboratory, Tampere University of Technology, Tampere, Finland. From 1993 to 1999, he was a Research Associate, first with the Department of Electrical and Computer Engineering and then with the Department of Informatics at the University of Thessaloniki. In March 1999, he joined the Department of Computer Science, University of York, U.K., where he is currently a Lecturer. He has authored and coauthored 13 journal papers and more than 50 papers in edited books and international conference proceedings. His research interests include digital watermarking, computational intelligence, computer vision, image processing, pattern recognition, and nonlinear digital signal processing.

Dr. Bors has been an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS since 2001 and a member of the technical committee for the IEEE International Conference on Image Processing every year since 2001.