Digital watermarking of 3D meshes

Mauro Barni[a], Franco Bartolini[b], Vito Cappellini[b], Massimiliano Corsini[b], Andrea Garzelli[a]

[a] Department of Information Engineering, University of Siena, Italy
[b] Department of Electronic and Telecommunications, University of Florence, Italy

ABSTRACT

Despite the increasing interest in digital watermarking of multimedia data, watermarking of 3D geometrical models has received little attention by the research community. One of the main reasons is that geometric data is intrinsically complex to handle, in addition a lot of diverse attacks can be thought of that are not possible in the 2D and 1D cases. So it is very difficult to develop *robust* watermarking algorithms for 3D models. In order to overcome the above problems most of the systems proposed so far exploit the knowledge of the original, non marked, mesh for watermark recovery. The practical usefulness of non-blind schemes, though, is very limited, hence the need to develop new blind schemes for 3D watermarking. In this paper we propose a blind watermarking algorithm for 3D meshes. Watermarking is achieved by perturbing the position of the vertices of the model according to a spherical pseudo-random bumped surface. The pseudo-random position and amplitude of the bumps encode the watermark. In order to gain robustness while keeping the distortion to a minimum, the watermark is embedded into a low resolution version of the mesh. The coarse version of the model is obtained by a MAPS[1] (Multiresolution Adaptive Parameterization of Surface) algorithm. The base domain produced by this algorithm is used to obtain the full resolution watermarked model from the coarse watermarked one. Watermark recovery is accomplished by means of a standard correlation detector. Experimental results show that the system assure high visual quality of the watermarked model, and that a good degree of robustness can be reached for models with a sufficiently high number of faces.

Keywords: Digital watermarking, 3D watermarking, geometric modelling, polygonal meshes, multiresolution 3D analysis.

1. INTRODUCTION

In these last years the applications using and managing 3D geometry data are quickly increasing in number even thanks to the computer graphics power reached by standard personal computers. For these reasons the diffusion of 3D models is in progress. Another key factor in the wide diffusion of a given data type is the existence of efficient algorithms for the processing of this media type. For example, for the image and video case a lot of processing tools to denoise, compress, transmit, enhance, analyse and edit these kind of signals exist. In this framework, computer graphics research community has recently put a lot of effort to provide a new mathematical framework for the so-called Digital Geometry Processing (DGP)[2] .

The creation of new tools to process geometric data is very helpful for 3D watermarking technology. As a matter of fact, 3D watermarking sets a brand new class of problems that were not present in the image and video cases; geometric data has intrinsic curvature, topology and no implicit ordering (with respect to the regular sampling of an image): it is not a simple 2D to 3D extension. Additionally a 3D model may undergo more complex and sophisticated attacks with respect to image and video media type. So it is very difficult to extend the well-consolidated image and video watermarking algorithms to this new type of media. The consequence is that while a lot of techniques and methods to embed copyright information in image and video have been developed and tested with good performances, only few algorithms to hide confidential information (for IPR, authentication and so on) into a 3D model have been developed.

In the following we provide a panoramic of the peculiarities of 3D watermarking, focusing on watermarking of polygonal meshes and we present a novel watermarking algorithm to embed a detectable watermark[3] into a polygonal mesh.

Contact author: M. Barni, Department of Information Engineering, University of Siena, Via Roma 56, 53100 - Siena, Italy, Phone: +39 0577 234621, Fax: +39 0577 263602, e-mail: barni@dii.unisi.it

1.1. 3D models and representation

Different representations are commonly used for 3D models; for example a 3D model can be described as a collection of parametric curves (e.g. Non-Uniform Rational B-Splines, NURBS[4]) or as a set of implicit surfaces[*]. More usually, a 3D model is represented by polygonal meshes.

A mesh can be seen as a $t-$uple (K, V) where $V = \{v_i \in \mathbb{R}^3 | i = 1 \ldots N_v\}$ is the set of the vertices of the model (points in $\mathbb{R}^3$) and K is a set encoding adjacency information for vertices, edges and faces of the mesh. In particular K is formed by subsets of $\{1, \ldots, N_v\}$. These subsets are called *simplices* of three types: vertices $v = \{i\} \in \mathcal{V}$, edges $e = \{i, j\} \in \mathcal{E}$, faces $f = \{i, j, k\} \in \mathcal{F}$. The set K is called *simplicial complex* and is defined as $K = \mathcal{V} \bigcup \mathcal{E} \bigcup \mathcal{F}$. A vertex v_i is a neighbor of another vertex v_j if an edge exists that connects v_i and v_j. The set of all the neighbors of a vertex v_i is called *1-ring* of the vertex and is defined as $v_1(i) = \{j | (i, j) \in \mathcal{E}\}$. The cardinality of $v_1(i)$ is called *degree* or *valence* of the vertex v_i. The *geometric realization* of a simplex $s \in K$, denoted with $\varphi(s)$, is the strictly convex hull of the vertices v_i with $i \in s$. For example the geometric realization of an edge $\{i, j\} \in \mathcal{E}$ is the segment connecting the vertex v_i with the vertex v_j, that of a face $\{i, j, k\} \in \mathcal{F}$ is the triangle defined by the vertices v_i, v_j and v_k, and so on. The 3D model is the geometric realization of the mesh $\varphi(K)$ defined as $\bigcup_{s \in K} \varphi(s)$. In this work we assume that the model is represented through a mesh, since the mesh is the lowest common denominator of surface representation. In fact, it is easy to convert other representations to meshes. Usually the vertices are characterized not only by theirs coordinates but even by other attributes such as texture coordinates, color and so on. Here we are interested only in the geometry of the mesh so we do not take in account these attributes. In the following, when we refer to a mesh we intend a *triangular mesh*, i.e. a mesh composed only by triangles. This assumption implies no loss of generality since every polygon of a non-triangular mesh can be triangulated to obtain a triangular mesh. A mesh is called *irregular* if its vertices can have any valence, *completely-regular* if all vertices have the same valence and *semi-regular* if most of its vertices have the same degree except a small number that can have any valence. This last definition arise because a semi-regular mesh is obtained by repeatedly regularly subdividing[5] an irregular coarse one. During the subdivision process the irregular vertices of the initial mesh remain irregular while most of the newly inserted vertices converge to valence six (for triangular semi-regular mesh). This classification is very important because for semi-regular meshes (and, obviously for completely regular ones) a lot of geometric processing tools exist. For example wavelet decomposition is defined only for semi-regular and completely-regular meshes[6] . An irregular mesh can be converted in a semi-regular one by a remeshing operation[1,7] . Our algorithm works on irregular meshes not to impose any restrictions on the regularity of the to-be-watermarked mesh.

1.2. 3D watermarking issues

In digital watermarking, a digital code, or watermark, is embedded into the 3D model, called the host or cover model, so that a given piece of information is indissolubly tied to it. This information can later be used to prove ownership, identify a misappropriating person, trace the model dissemination through the network, or simply inform users about the rights-holder or the permitted uses.

The way watermarking algorithm recover the watermark from the model has a strong impact on practical applications, it is then common to classify digital watermarking techniques by their decoding processes.

- **Blind vs. non blind**. A watermarking algorithm is blind if it does not need to compare the marked and unmarked documents to recover the watermark. Conversely, a watermarking algorithm is not blind if it needs the original data to extract the information from the marked document. Blind techniques are sometimes referred to as oblivious or public.
- **Readable vs detectable**. In this case we distinguish between algorithms that embed a code that can be read without knowing it in advance, and those that insert a mark that can only be detected, that is, a user can only verify whether a given code is contained in the document. Detectable watermarking is sometimes referred to as 1-bit watermarking because the detector output is just yes or no.

[*]The implicit method uses an equation depending by axis variables to describe a shape. For example the equation $x^2 + y^2 + z^2 = 1$ represents the sphere of radius 1.

In this paper we focus on blind, detectable watermarking of 3D meshes. As to the requirements a watermarking system must satisfy, the most important ones are robustness, i.e. the ability to survive manipulations, unobtrusiveness, and capacity. Of course, the exact meaning of the above requirements depend on the type of media under consideration, thus in the following sections we specialize them to the 3D case.

1.2.1. Requirements of 3D watermarking

Watermark capacity

Although in general the watermark capacity does not depend on the particular algorithm, but it is related to the characteristics of the host signal, of the allowed embedding distortion and of the attack strength, we will refer to the capacity of a given technique as the amount of information bits that the watermark is able to convey. In this sense, capacity is a fundamental property of any watermarking algorithm, which very often determines whether a technique can be profitably used in a given context or not. Generally speaking, capacity requirements always struggle against two other important requirements, that is imperceptibility and robustness.

Having said this, it is obvious that the capacity of any 3D watermarking system is in relation with the complexity of the given mesh, where by *mesh complexity* we intend the number of faces and vertices it contains. Thus, a mesh with millions of faces will convey more bits than a simple mesh with a few faces.

Imperceptibility

The watermarked model must maintain the same visual quality of the original one. The importance and the meaning of this property depends on the intended use of the model. Usually the intended use is viewing; so the watermarked model and the original one must appear identical to visual inspection. This is a crucial point because often a user sees a 3D model in an interactive way. On the contrary, images and video do not allow such an extensive user-interaction, so it is by far simpler to hide the watermark using appropriate perceptual masks[8, 9] . It is important to underline that for some applications the imperceptibility of the watermark may not be a sufficient requirement. This is the case, for example, when we want to analyze the deformations of cultural heritage goods by periodic 3D acquisition of their surfaces.

Robustness

Every watermarking algorithm to be used in IPR (Intellectual Property Rights) applications has to be robust against manipulations, usually called attacks, of the watermarked media. The problem with 3D watermarking is that a lot of attacks are possible. In section 1.2.3 we give more details on 3D watermarking attacks.

1.2.2. Embedding domain

The first step towards the definition of a watermarking algorithms, consists in the choice of the host features, i.e. the selection of a set of properties of the cover 3D model that will bear the watermark information. Of course, many possibilities exist here, however, as we already said, we are interested in the geometry of the model, so we focus only on geometric and topological features.

- **Geometric Features.** The main geometric features of a mesh are its vertices. One possible way to embed the watermark is to modify the position or the normals of vertices (vertex normals are related to the curvature of the mesh). Both these entities are altered by perturbing the coordinates of mesh vertices.
- **Topological Features.** These features are related to the connectivity of the mesh vertices. Usually, a set of connected vertices is selected by using geometric features. Then, the topology of these vertices is redefined to encode one or more bits.

In the proposed approach we have decided to use as embedding features the vertices position because vertices contain most of the information of the mesh and the approaches based on topological features suffer the re-triangulation attack (see next section), that is straightforward to implement.

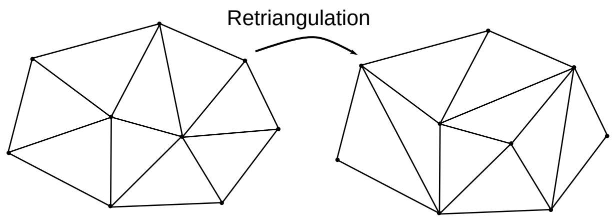

Figure 1. Re-triangulation attack.

1.2.3. Attacks

One of the main problems with 3D watermarking is that a lot of complex attacks can be carried out on a polygonal mesh. Possible attacks range from simple attacks such as rotation, translation and uniform scaling to complex attacks such as remeshing that involves geometric resampling with topology changes of the surface. It is difficult to imagine all the possible attacks on a 3D model. Some of the most important ones are:

- **Translation/Rotation/Uniform Scaling.** These geometric transformations are very used in computer graphics to position a 3D model within a scene. In addition, geometric transformations such as affine and projective transformations may be used, even if they are less common that plain translations, rotations and isotropic scaling.
- **Noise.** For noise attack we intend the addition of random vectors to mesh vertices. The modulus of these vectors have to be small compared with the mesh dimension to preserve the overall shape of the model.
- **Re-triangulation of vertices.** This attack concerns the changes between the connections of the mesh vertices (figure 1).
- **Mesh smoothing.** A smoothing of the surface represented by a polygonal mesh can be obtained by mesh filtering such as Taubin filtering.[10] This kind of filters act on the mesh as a low-pass filter attenuating the roughness of the surface.
- **Polygonal simplification.** Polygonal simplification is often used to transmit a low-level version of the model or to optimize a model eliminating most of the non-salient faces.
- **Cropping.** Cropping concerns the disjunction of a part of the model. Users can discard the pieces of the model that they do not need (e.g. the hand of a statue).
- **Remeshing** Remeshing is used to regularize a mesh converting an irregular mesh into a semi-regular[1,7] or a completely-regular[11] one. This operation can be seen as a geometric resampling of the shape of the model followed by a re-definition of the vertices connections (a re-triangulation) in order to give the mesh vertices the desired valence.

1.3. Previous works on 3D watermarking of polygonal meshes

In these last years some algorithms to embed data into 3D polygonal models have been developed. Most of them have been proposed by Ohbuchi et al.[12,13] and by Olivier Benedens[14,15] .

More recently, 3D watermarking algorithms that use a multiresolution approach, such the algorithm proposed in this paper, have been developed[16,17] . The main limitations of these algorithms is that they require the original model to recover the watermark. On the contrary, the fundamental feature of our multiresolution-based technique is blindness, i.e. our algorithm does not need the original model to extract the inserted watermark. Additionally, the proposed method provides a high visual quality of the watermarked model.

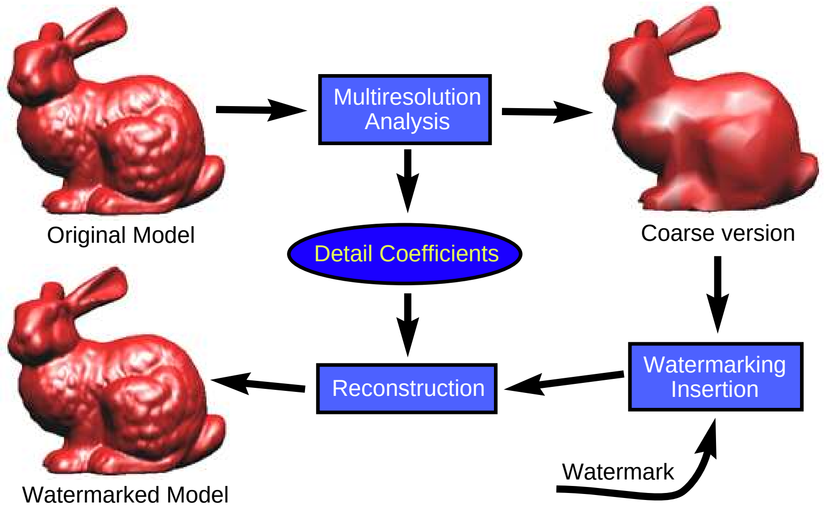

Figure 2. Sketch of the proposed approach.

2. A MULTIRESOLUTION 3D WATERMARKING FRAMEWORK

As sketched in figure 2, our approach to the 3D watermarking problem is a multiresolution one. We start by performing a multiresolution analysis of the mesh, since we assume that the rough shape of the mesh is already contained in a low resolution version of the model, and that such a shape can not be modified by the attacker. Then we perturb the vertices of this coarse version and extend these perturbations to the higher resolution versions of the mesh so to obtain the final marked-mesh. In the watermark extraction process, the coarse version of the model is computed again by replicating the analysis performed by the embedder. The detector, then, looks for the watermark at the coarse level.

The described approach gives a *class of blind watermarking* algorithms. In fact, the multiresolution analysis can be performed by using a large number of well-consolidated different methods and, consequently, the extension of the watermark from the coarse version to the original one can be done in many different ways. So, a lot of variations can be investigated to find the most robust and promising ones. The main idea behind the multi-resolution approach is that the insertion of the watermark in a low-resolution version of the original model has two benefic effects. The first one is that the visual quality of the watermarked model is not impaired, the second one is that some attacks, hopefully the most common ones, leave the coarse version of the model virtually intact. The latter effect can be motivated by noting that since the low resolution version *captures* the very content of the model, it can not be modified without degrading the model quality. Stated in another way, we are following the old watermarking paradigm requiring that the watermark be injected into perceptually significant components of the host data.

As to the algorithm used to derive the coarse version of the model, we used a variant of the Multiresolution Adaptive Parameterization of Surface (MAPS)[1] algorithm to obtain a coarse parameterized version that enable us to extend the watermark from the low-resolution mesh to the original one. Before describing the insertion and extraction process we now give a brief explanation of the MAPS algorithm.

2.1. MAPS algorithm

An important element in the design of algorithms which manipulate mesh approximations of 2-manifold is the construction of a parameterization of the mesh. Usually, the manifold is parameterized over a *base domain* consisting of a set of planar regions that piecewise map the original mesh by a function defined from this base domain to $\mathbb{R}^3$. The main idea of MAPS is to use mesh simplification to induce a parameterization of the original mesh over a base domain consisting of a coarse version of the input mesh where each triangle parameterizes a region of the starting mesh (figure 4). In a few words, during the simplification process a mapping is constructed.

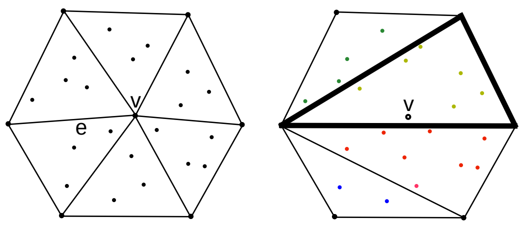

Figure 3. A example mesh with previously assigned vertices. The reassignment after a simplification step (vertex v is removed by collapsing the edge e, then it is assigned to the evidenced triangle).

Each point of the original mesh is mapped onto the surface of a triangle of the base domain, by paying attention to record the information (details) that permit to recover the original position of the point on the original mesh.

More specifically, let $\varphi(K^N)$ be the geometric realization of the original mesh (level N), $\varphi(K^0)$ the geometric realization of the base domain (level 0) and Π a bijection from $\varphi(K^N)$ to $\varphi(K^0)$ that maps the point $p \in \varphi(K^N)$ on the point $p^0 \in \varphi(K^0)$ in the following way:

$$p^0 = \alpha p_i + \beta p_j + \gamma p_k, \tag{1}$$

where (p_i, p_j, p_k) defines a triangle of the base domain and α, β and γ are barycentric coordinates ($\alpha + \beta + \gamma = 1$).

The mapping Π is constructed during the simplification process computing, for each simplification step, the piecewise linear bijections Π^{N-1} between $\varphi(K^N)$ and $\varphi(K^{N-1})$ starting with Π^N, which is the identity, and ending with $\Pi^0 = \Pi$. In particular, for each vertex (call it p_i) removed by a simplification step the flattening 1-ring around p_i is considered. After the flattening, the simplification is performed and p_i is assigned to the face of the new triangulation where it lies (see vertex v in figure 3). So, let (α, β, γ) the barycentric coordinates of p_i with respect to the flattened face where it lies, i.e. $p_i = \alpha\mu(p_j) + \beta\mu(p_k) + \gamma\mu(p_m)$, the map results $\Pi^{N-1}(p_i) = \alpha p_j + \beta p_k + \gamma p_m$. The operator $\mu(.)$ performs a proper (conformal map[18]) flattening of $p_1(i)$. It is important to underline that not only the removed vertex needs to be assigned to a face, but even all the previously-assigned vertices involved in the simplification step must be reassigned (figure 3).

Figure 4 shows the parameterization of a particular of the bunny model (69,451 faces) over a base domain composed by 500 faces. After the parameterization it is possible to determine the position on the mesh of each point of the base domain, through Π^{-1}. In particular, given a point p on the base domain and the triangle which contains it, i.e. $p = \alpha\Pi(p_i) + \beta\Pi(p_j) + \gamma\Pi(p_k)$, the inverse mapping is:

$$\Pi^{-1}(p) = \alpha p_i + \beta p_j + \gamma p_k \in \varphi(K^L), \tag{2}$$

where p_i, p_j and p_k are the original mesh vertices and α, β and γ are the barycentric coordinates of p on the base domain with respect to the triangle $(\Pi(p_i), \Pi(p_j), \Pi(p_k))$.

We decided to use the MAPS algorithm because it allows to explicit control the number of triangles of the base domain, it avoids some problems typical of mesh parameterizations and the constructed base domain coincides with a coarse version of the original model. In our implementation we use Garland's simplification[19] algorithm to simplify the mesh. This simplification is based on a quadric error metric that provides a coarse version that is a good geometric approximation of the original model.

2.2. Insertion

The watermark embedding process can be described by the following steps:

1. The MAPS algorithms is applied to the original mesh. The output of this step is a coarse parameterized version of the original model. Thanks to this we can associate each vertex of the original mesh to a point on a base triangle. In the next we see how we have used this vertices association to compute details that allow us to reconstruct the model from the deformed (watermarked) coarse version.

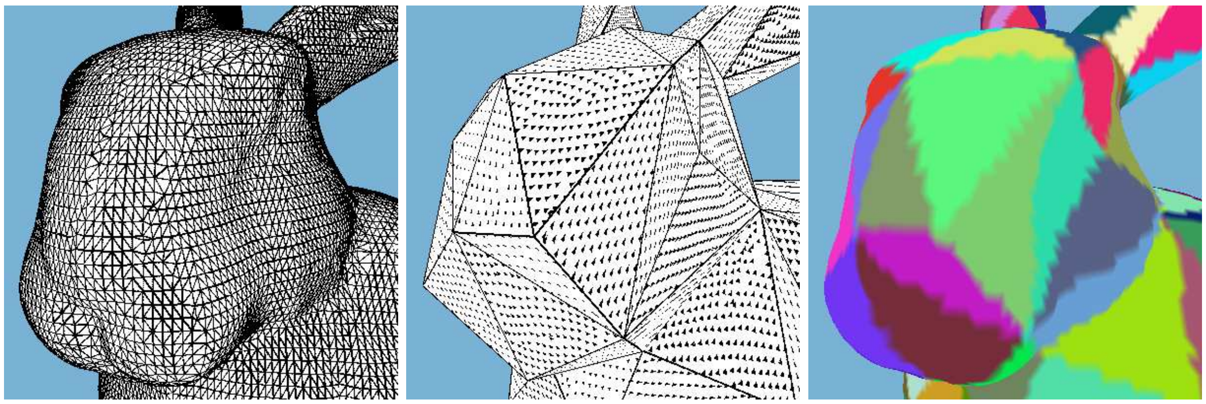

Figure 4. Left: Input bunny model (detail). Center: Computed base domain; each points on base triangle correspond to a vertex on the input mesh. Right: Regions of the original model colored according to their assigned base triangle.

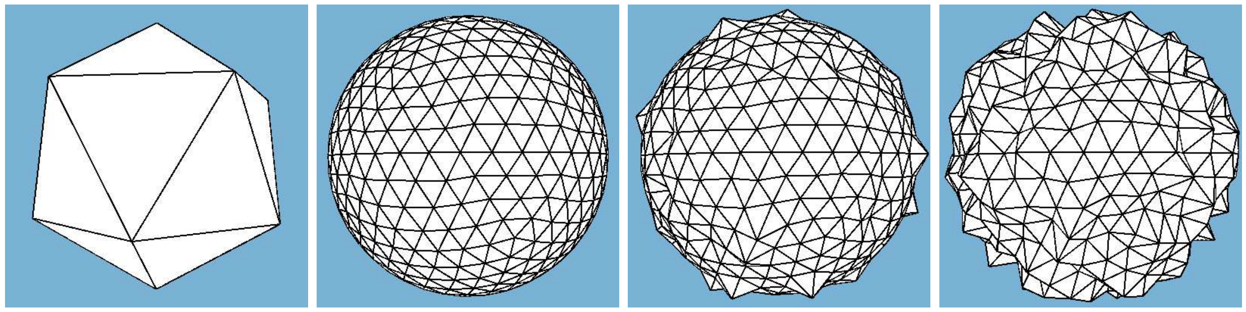

Figure 5. Watermarking sphere generation. From right to left: the starting icohesaedron, the icohesaedron subdivided 3 times, the sphere during the "bumps adding" process and the final watermarking sphere.

2. A bumped spherical surface is generated starting from a pseudorandom sequence W representing the to-be-embedded watermark. In the following we refer to this surface with the name *watermarking sphere*.

3. The vertices of the base domain are perturbed according to the watermarking sphere.

4. The details of the model, that were stored previously, are re-add to the watermarked coarse version of the mesh (reconstruction phase) to produce the watermarked model.

2.2.1. Watermarking sphere generation

The watermarking sphere is generated by a pseudo-random sequence of linear "bumps"on a spherical mesh of original radius R. First of all the spherical mesh is created by recursively subdividing an icosahedron. Then this flat sphere is bumped according to the watermark sequence $W = \{w_1, w_2 \ldots w_{N_V}\}$. The watermark W is nothing but a pseudo random sequence uniformly distributed in the range $[-R_{max}, R_{max}]$. The sequence length N_V coincides with the number of vertices of the sphere. The radius of each vertex v_i of the sphere is altered by adding to it the quantity w_i.

Two parameters control the generation of the watermarking sphere: R_{max}, determining the strength of the watermark and the number of times that the icosahedron is subdivided. For example subdividing the icosahedron five times yields a watermarking sphere composed by $20 \times 4^5 = 20,480$ faces. The number of faces of the watermarking sphere (N_f) is an important parameter to take in account because it influences, with R_{max}, the recovery process and the robustness of the algorithm. More specifically, it determines the degree of correlation of the bumps on watermarking sphere: if N_f is much larger than the number of vertices of the model, then the

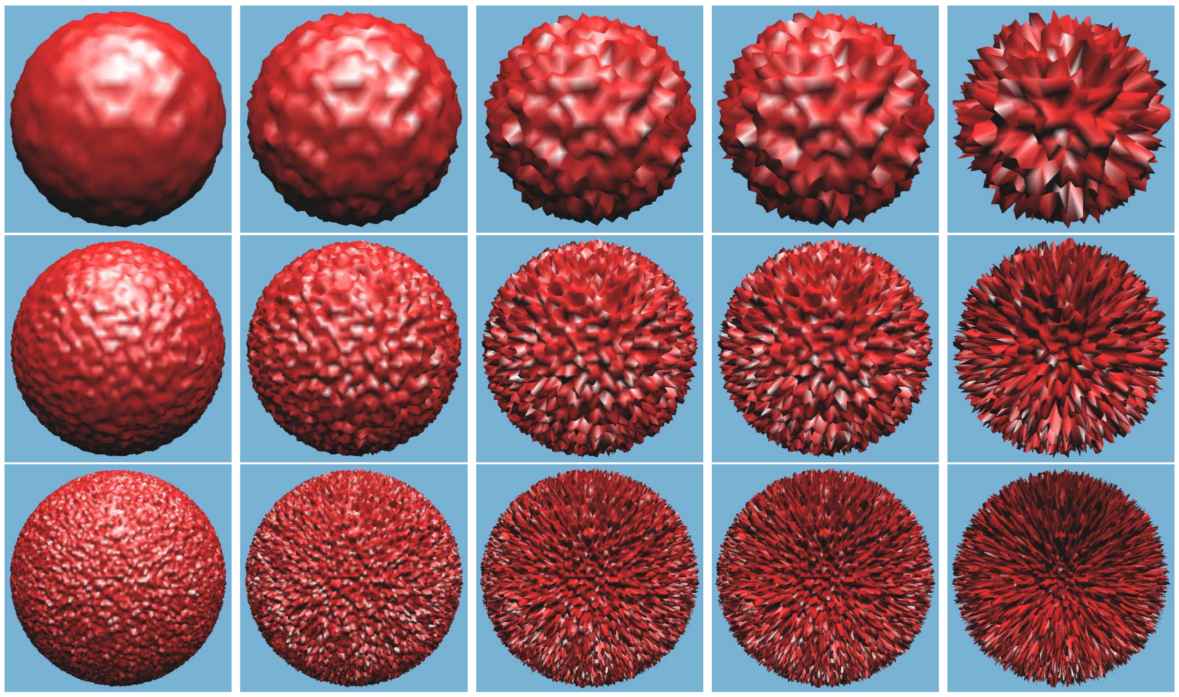

Figure 6. Watermarking spheres. The parameter R_{Max} increases from left to right (ranging from 0.2 through 2.0). The spheres of the first row have $N_f = 5,120$ faces, those of the second rows and third $N_f = 20,480$, and $N_f = 81,920$ respectively.

watermark may be assumed to be uncorrelated, otherwise the non-null correlation between the points of the sphere must be taken into account.

The watermarking sphere generation process is depicted in figure 5, whereas some watermarking spheres obtained with different parameters of N_f and R_{max} are showed in figure 6. As it may be seen R_{max} controls the "height"of the bumps while N_f controls the density of bumps on the sphere.

2.2.2. Watermarking process

Watermark embedding corresponds to step 3 of the insertion algorithm.

All the watermarking process is better described in polar coordinates, so when we refer to a vertex position we have to think to it in spherical coordinates (ϕ, θ, r). Imagine to place the watermarking sphere in the baricenter of the coarse version of the to-be-marked model; by extending the lines connecting a vertex v_i to the baricenter of the mesh we identify a point of intersection P_i on the watermarking sphere characterized by spherical coordinates $(\Phi_{P_i}, \Theta_{P_i}, R_{P_i})$. Watermarking is achieved by modifying the position of each vertex of the coarse parameterized version according to the radius of P_i. More specifically, by letting r_i be the distance of the i-th vertex from the baricenter of the 3D model, the vertex is moved to a new position defined by the same angular coordinates and a new radius given by:

$$r_{i,w} = r_i + \gamma(R_{P_i} - R) = r_i + \gamma\Delta_i, \tag{3}$$

where γ is a parameter controlling the watermark strength, R_{P_i} is the radius of the sample value on the watermarking sphere associated to the vertex v_i, R is the radius of the non-marked watermarking sphere and where we let $\Delta_i = R_{P_i} - R$.

2.2.3. Model reconstruction

After the base domain is produced by the MAPS algorithm we consider, for each point of the base triangle associated to a vertex of the original mesh, the vector connecting it to its original position. These vectors represent the "details" to reconstruct the mesh from the coarse version. In fact, to reconstruct the mesh we simply adds these vectors to theirs associated points over the base domain. These vectors are stored before watermark insertion. To obtain the watermarked model from its watermarked coarse version we re-add to each point of the base domain its associated detail vector.

Let $\vec{t} = (t_x, t_y, t_z)$ be the detail vector associated to the point $p = (p_x, p_y, p_z)$ liying on the base triangle $T(v_1, v_2, v_3)$. After watermark insertion the position of p changes. To find the new position of p, say p_w, we use the baricentric coordinates (α, β, γ) of p with respect to triangle T. So, if p is expressed as $p = \alpha v_1 + \beta v_2 + \gamma v_3$ its new position becomes $p_w = \alpha v_{1,w} + \beta v_{2,w} + \gamma v_{3,w}$. To obtain the position of p_w on the original mesh (P_w) we simply add $\vec{t}$ to p_w; $P_w = \vec{t} + p_w$, while the original position was $P = \vec{t} + p$.

2.3. Recovery

Given the watermark W and a mesh, the detector must decide whether W is contained in the mesh or not. This problem reduces to a standard hypothesis testing problem. Under the hypothesis that r_i and Δ_i are uncorrelated i.i.d. normal random variables, the optimum watermark detector reduces to a correlation-based detector[20] . As previously noticed, it is important to underline that r_i and Δ_i are uncorrelated if and only if N_f is sufficiently high with respect to the number of vertices of the base domain.

To be specific, let us assume the to-be-inspected model has already been simplified according to Garland's algorithm. The simplification stops when the coarse version reaches a prefixed number of vertices (n). Then, by relying on W the watermarking sphere is generated and the correlation between the radial components of each vertex and the corresponding points on the bumped sphere is computed:

$$\rho = \frac{1}{n} \sum_{i=1}^{n} r_i \Delta_i, \tag{4}$$

where r_i and Δ_i are the same as in equation (3), and n is the number of vertices of the base domain. Of course the assumptions under which the correlation-based detector is optimum may not hold in practice, however for the sake of simplicity, and for the difficulty to define a proper model fitting the statistics of r_i and Δ_i, we decided to adopt a correlation based detector. Such a choice is validated a posteriori by means of experimental results. To summarize, let H_0 and H_1 denote the following hypothesis:

- H_0 : the watermark given by W is not present or another watermark is present
- H_1 : the watermark is present

We accept as true the hypothesis H_1 if $\rho > T_\rho$, otherwise we accept as true H_0. The probability to accept H_1 when the watermark is not present is called *false detection* probability, $P_f = P\{\rho > T_\rho | H_0\}$, while the probability to accept H_0 when the watermark is present is called *missed detection* probability and is given by $P_m = P\{\rho <= T_\rho | H_1\}$. By fixing P_f and by invoking the central limit theorem, it is possible to compute P_m[20]:

$$P_m = \frac{1}{2} \text{erfc} \left(\frac{\gamma \sqrt{n \overline{\Delta^2}} - \sqrt{2} \text{erfc}^{-1}(2P_f) \sigma_r}{\sqrt{2} \sigma_r} \right), \tag{5}$$

where $\overline{\Delta^2} = \frac{1}{n} \sum_{i=1}^{n} \Delta_i^2$, erfc() is the error function and σ_r^2 is the variance of the radial component of the mesh vertices. The threshold T_ρ can be expressed as a function of P_f:

$$T_\rho = \sqrt{\frac{2\sigma_r^2 \overline{\Delta^2}}{n}} \text{erfc}^{-1}(2P_f) + \mu_r \overline{\Delta}, \tag{6}$$

where $\overline{\Delta} = \frac{1}{n} \sum_{i=1}^{n} \Delta_i$ and μ_r is the expected value of the radial component of the mesh vertices.

It is worth noting that from equation (5) and (6) it is possible to see that the number of vertices of the watermarked coarse version plays an important role in the robustness of the watermarking algorithm, since the higher the n the lower the P_m for a given P_f.

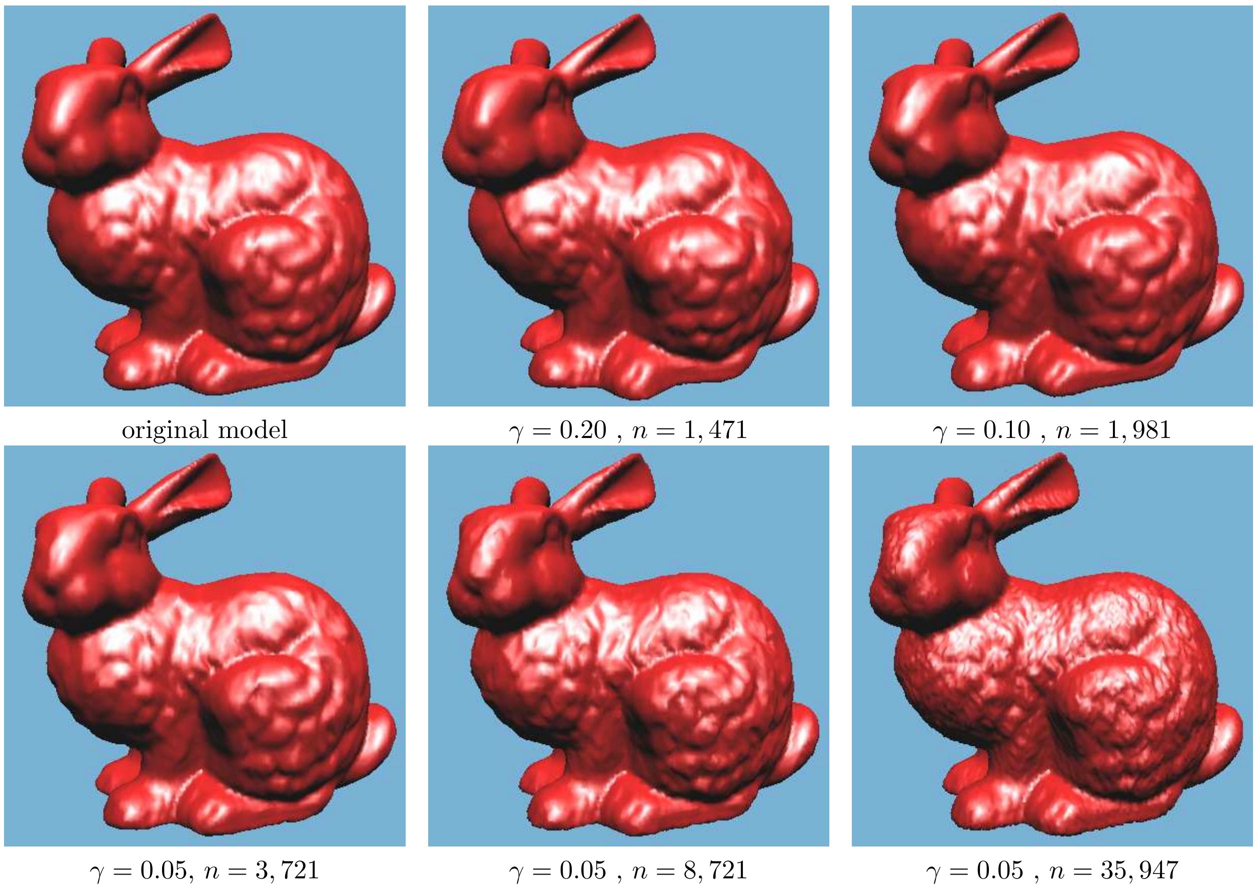
original model
$\gamma = 0.20$, $n = 1,471$
$\gamma = 0.10$, $n = 1,981$
$\gamma = 0.05$, $n = 3,721$
$\gamma = 0.05$, $n = 8,721$
$\gamma = 0.05$, $n = 35,947$

Figure 7. Visual perception of the watermark model for different values of γ and n (defining the coarseness of the base domain). In all the cases Δ belongs to the [-0.1,0.1] interval. Note that the original Bunny model has 69,451 faces.

3. EXPERIMENTAL RESULTS

In this section we evaluated the effectiveness of the proposed watermarking technique from the points of view of obtrusiveness and robustness. With regard to visibility we obtained excellent results, whereas those related to robustness are still preliminary and show that an acceptable degree of robustness can only be achieved for meshes containing a sufficiently large number of vertices.

3.1. Watermark visibility

We evaluated the visibility of the watermark as a function of the watermark strength and the resolution of the base domain the watermark is embedded in. The results we obtained confirm the validity of the multiresolution approach, in that the visibility of the watermark diminishes when the hidden information is embedded at a lower resolution level. An example of this behavior is given in figure 7, where the bunny model is watermarked at different resolution levels and with different watermark strengths. In all the cases we let Δ be uniformly distributed in the [-0.1,0.1] interval. As it can be seen, when the watermark is embedded at a very low resolution, a higher γ can be used without compromising invisibility. Whereas, at high resolution levels, even a very small γ results in a visible watermark.

3.2. Watermark robustenss

A rough measure of watermark robustness is given by ROC curves, in which the missed detection probability is plotted as a function of P_f. Actually, ROC curves are plotted by assuming that no attack is present, ROC

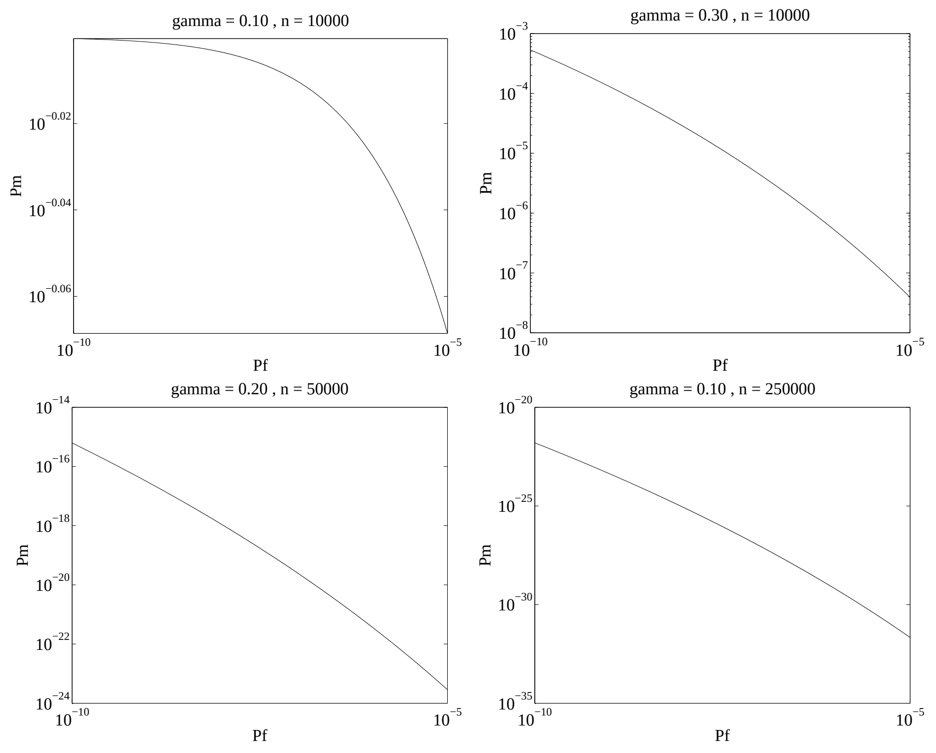

Figure 8. Receiving Operation Characteristic (ROC) Curves for different values of γ and n. The other parameters appearing in equation (5) have been set by looking at the values they assume for the Bunny model, i.e $\sigma_r^2 = 0.032$, $\Delta \in [-0.1, 0.1]$.

curves characterized by extremely low values of P_f and P_m are a good indication that the watermark is a robust one (at least against very simple attacks such as noise addition). In figure 3.2 the ROC curves stemming from equations (5) are given. As it can be seen, in order to achieve a satisfactorily degree of robustness either a high value of γ must be sued or a large number of vertices must be marked. By comparing the values used to plot the curves in the figure, and those used to obtain the visibility results shown in figure 7, we can conclude that the proposed technique is not suitable to watermark rather simple meshes as that describing the Bunny model. On the contrary, very good results have to be expected when very complex meshes consisting of millions of faces are considered. This is the case, for example, of the 3D digital models acquired in the field of Cultural Heritage (for instance, in the framework of the Digital Michelangelo Project of Stanford University 3D models with about 300 millions faces. have been produced).

4. CONCLUSIONS AND FUTURE WORK

In this work we addressed the problem of blind watermarking of 3D meshes. To this aim, we proposed a multiresolution framework, and presented a practical implementation based on the MAPS simplification algorithm. Watermark insertion and retrieval are kept very simple, since a simple additive embedding rule, and a correlation detector have been used - the whole embedding algorithm runs in less then 10s on a PC with 512 Mbyte of RAM. In spite of this, the results we obtained are encouraging. More specifically, the multiresolution framework permitted us to perfectly hide the watermark within the host 3D mesh, while contemporarily giving promising indications with regard to robustness, at least if a sufficiently large number of vertices is available. Of course a much deeper investigation about robustness is needed, especially with regard to typical 3D attacks, such as re-triangulation, mesh simplification and re-sampling. Future work will also address the possibility of retrieving the watermark after that the mesh has been rotated or scaled, e.g by inserting a re-synchronization step based on hierarchical principal component analysis before watermark detection.

REFERENCES

1. W. F. Lee, W. Sweldens, P. Schroeder, L. Cowsar, and D. Dobkin, “MAPS: Multiresolution adaptive parameterization of surfaces,” in *Proc. SIGGRAPH 1998*, pp. 95–104, 1998.
2. P. Schroder and W. Sweldens, “Digital geometry processing,” *In Sixth Annual Symposium on Frontiers of Engineering* , pp. 41–44, 2001.
3. F. Bartolini, A. Piva, and M. Barni, “Managing copyright in open networks,” *IEEE Internet Computing* **6**, pp. 18–26, May/June 2002.
4. P. Lavoi, “An introduction to nurbs,” *on-line tutorial http://libnurbs.sourceforge.net/nurbsintro.pdf* , January 1999.
5. P. Schroder and D. Zonin, “Course notes: Subdivision for modeling and animation,” *ACM SIGGRAPH 1998* , 1998.
6. M. Eck, T. D. T. DeRose, H. Hoppe, M. Lounsbery, and W. Sweldens, “Multiresolution analysis of arbitrary meshes,” in *Computer Graphics (SIGGRAPH ’96 Proceedings)*, pp. 325–334, 1995.
7. M. Lounsbery, T. DeRose, and J. Warren, “Multiresolution analysis for surfaces of arbitrary topological type,” in *Transaction on Graphics (SIGGRAPH ’95 Proceedings) Annual Conference Series*, 1995.
8. C. I. Podilchuk and W. Zeng, “Image-adaptive watermarking using visual models,” *IEEE Journal Sel. Areas Commun.* **16**, pp. 525–539, May 1998.
9. R. B. Wolfgang, C. I. Podilchuk, and E. J. Delp, “Perceptual watermark for digital images and video,” *Proc. IEEE* **87**, pp. 1108–1126, July 1999.
10. G. Taubin, “A signal processing approach to fair surface design,” *ACM SIGGRAPH 95* , pp. 351–358, August 1995.
11. X. Gu, S. J. Gortler, and H. Hoppe, “Geometry images,” *ACM Siggraph 2002* , pp. 355–361.
12. R. Ohbuchi, H. Masuda, and M. Aono, “Watermarking three-dimensional polygonal models,” in *ACM Multimedia ’97*, pp. 261–272, 1997.
13. R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama, “Watermarking 3D polygonal meshes in the mesh spectral domain,” in *Proceedings of Graphics Interface 2001*, B. Watson and J. W. Buchanan, eds., pp. 9–18, 2001.
14. O. Benedens, “Two high capacity methods for embedding public watermarks into 3D polygonal models,” in *Proc. Multimedia and Security*, pp. 95–99, (Orlando, Florida, USA), 1999.
15. O. Benedens, “Watermarking of 3D polygonal based models with robustness against mesh simplification,” in *Proc. SPIE Security and Watermarking of Multimedia*, pp. 329–340, 1999.
16. E. Praun, H. Hoppe, and A. Finkelstein, “Robust mesh watermarking,” in *Proc. SIGGRAPH 1999*, pp. 49–56, August 1999.
17. S. Kanai, H. Date, and T. Kishinami, “Digital watermarking for 3D polygonals using multiresolution wavelet decomposition,” in *Proc. of the 6th IFIP WG5.2 International Workshop on Geometric Modeling Fundamentals and Applications (GEO-6)*, pp. 296–307, (Tokyo, Japan), December 1998.
18. T. D. T. Duchamp, A. Certain and W. Stuetzle, “Hierarchical computation of PL harmonic embeddings.,” *Tech. Report,* University of Washington , July 1997.
19. M. Garland and P. Heckbert., “Surface simplification using quadric error metrics,” *In SIGGRAPH 97 Proc.* , pp. 209–216, August 1997.
20. J. R. Hernandez, M. Amado, and F. Perez-Gonzales, “DCT-domain watermarking techniques for still images: detector performance analysis and a new structure,” **9**, pp. 55–68, Jan. 2000.