# Optimized 3D Watermarking for Minimal Surface Distortion

Adrian G. Bors, *Senior Member, IEEE*, and Ming Luo

***Abstract*— This paper proposes a new approach to 3D watermarking by ensuring the optimal preservation of mesh surfaces. A new 3D surface preservation function metric is defined consisting of the distance of a vertex displaced by watermarking to the original surface, to the watermarked object surface as well as the actual vertex displacement. The proposed method is statistical, blind, and robust. Minimal surface distortion according to the proposed function metric is enforced during the statistical watermark embedding stage using Levenberg–Marquardt optimization method. A study of the watermark code crypto-security is provided for the proposed methodology. According to the experimental results, the proposed methodology has high robustness against the common mesh attacks while preserving the original object surface during watermarking.**

***Index Terms*— 3D digital watermarking, Levenberg–Marquardt optimization, surface error minimization.**

## I. Introduction

DIGITAL watermarking aims to fulfill simultaneously a set of requirements such as: non-visibility, robustness, high bit capacity and crypto-security. Usually, by enforcing any of these constraints we limit the effectiveness of all the others. The methodology proposed in this paper aims to minimize the distortions introduced in the graphical object by watermarking while being blind and robust as well. Digital watermarking was considered for a variety of media such as: images, video, audio, 3D graphical objects or vectorial data. In the following we consider the meshes of 3D graphical objects as cover media. Depending on whether the original graphical object is needed or not in the decoding stage we have non-blind or blind watermarking. The methods from the first category usually have good robustness [1], [2] but a limited range of applications, and in this paper we only consider blind watermarking. The application of watermarking methodology includes copyright protection, authentication, digital fingerprinting, embedding information relevant to the graphical object for database indexing purposes, etc.

Watermarking methods can be categorized as deterministic [3], [4] or statistical [5]–[7]. The methods from the first category embed a set of constraints [3], [8], while those from the second category perform statistical changes in distributions of measurements from the object's mesh. Usually, deterministic methods allow a higher capacity of information embedding, making them suitable for steganography, but on the other hand they have lower robustness to attacks. Watermarking of graphics has been performed in both spatial and transform domains. Transform domain watermarking considers embedding the message in the mesh spectral domain [2], [9], the wavelet domain [10], or in the spheroidal harmonic coefficients [11]. Spatial domain considers embedding the watermark directly in the geometry of the 3D object [3], [7], [8].

Watermarking, invariably introduces distortions in the mesh surface. A study of the distortion introduced by digital watermarking in mesh surfaces of 3D objects was performed in [3]. Visual masking procedures, used for hiding changes produced by the watermarks in 3D graphics, have been proposed in [12] for the distortions introduced by the methods proposed in [4] and [8]. A surface preservation method using embeddings within distributions of geodesic distances was proposed in [7]. A method minimizing the quadric error metric (QEM) by using geometric constraints with respect to the cover object surface was used in [13].

In this paper we propose a new methodology for 3D watermarking by ensuring a minimal surface distortion. Distributions of distances from vertices to the object center are changed according to the watermark code using the Levenberg–Marquardt optimization in spherical coordinates. The optimization process minimizes the sum of Euclidean distances from the vertex, displaced by watermarking, to the original surface, to the watermarked surface and to its original location. The Levenberg–Marquardt method [14], [15] was used in a variety of applications including surface fitting of graphical objects [16], shape processing [17] and image watermarking [18]. The proposed methodology ensures a minimal distortion in the resulting watermarked 3D mesh and represents a generalization of the 3D watermarking methods from [6] and [13]. Section II describes the statistical approach to 3D watermarking. Section III provides the surface distortion metric used as a watermarking cost function. Section IV describes the watermark embedding approach for the vertex placement using the Levenberg–Marquardt method. A study of watermark robustness in the minimal 3D object while preserving its surface is provided in Section V while the crypto-security for the proposed method is analyzed in Section VI. Section VII provides the experimental results, while Section VIII concludes the paper.

Manuscript received April 27, 2012; revised October 16, 2012; accepted December 11, 2012. Date of publication December 24, 2012; date of current version March 11, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jana Dittmann.

The authors are with the Department of Computer Science, The University of York, York YO10 5GH, U.K. (e-mail: adrian.bors@york.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2012.2236345

## II. Statistical Watermarking of Mesh-Based Representations of 3D Objects

This section describes the processing stages for statistical watermarking: statistical variable representation, histogram mapping function and the watermark decoding algorithm.

### A. Statistical Variable Representation

Let us assume that we want to embed a message (code) of $M$ bits into a 3D object $\mathcal{O}$. We denote the object center by $O$ and a vertex on its surface as $V_j$, while their corresponding coordinates are $\mathbf{o}$ and $\mathbf{v}_j$, respectively. The vertices of the mesh object $\mathcal{O}$ are clustered into $M$ bins such that each bin is used for embedding one bit of message $B_i$, $i = 1, \ldots, M$. In the proposed approach, we cluster the vertices according to their distance from $\mathbf{v}_j$ to the object center $\mathbf{o}$, which is considered as reference:

$$\mathbf{o} = \frac{\sum_{\mathbf{v}_j \in \mathcal{O}} A(\mathbf{v}_j)\mathbf{v}_j}{\sum_{\mathbf{v}_j \in \mathcal{O}} A(\mathbf{v}_j)} \tag{1}$$

where $A(\mathbf{v}_j)$ represents the area of all triangular faces containing the vertex $\mathbf{v}_j$. The object center defined in this way is more robust than by simply taking the average of all its vertices, as it was used in [6], particularly when considering the watermark robustness to various attacks such as remeshing or simplification. For the center calculation we could have used the object volume moments as in [19], as well.

For a given vertex $\mathbf{v}_j \in \mathcal{O}$, let us denote its distance to the reference point given by the center, representing the vertex norm, by $\rho_j = \|\mathbf{v}_j - \mathbf{o}\|$ and consider this as a statistical variable. After ranking the distances from the center to vertices, we evaluate $\rho_{min} = \min_{\mathbf{v}_j}(\rho_j)$ and $\rho_{\max} = \max_{\mathbf{v}_j}(\rho_j)$, where $\mathbf{v}_j \in \mathcal{O}$. Consequently, the vertices are grouped into $M$ sets according to their distances to the object center, as:

$$\begin{aligned}\mathcal{B}_i = \{\mathbf{v}_j \in \mathcal{O} \mid \rho_{min} + \varepsilon(\rho_{\max} - \rho_{min}) + (i-1)\rho_b \le \rho_j, \\ \rho_j < \rho_{\max} - \varepsilon(\rho_{\max} - \rho_{min}) + i\rho_b\}\end{aligned} \tag{2}$$

where $i = 1, \ldots, M$ and each of these sets contains a number of vertices located in an identical range of distances from the object center equal to:

$$\rho_b = \frac{(1-2\varepsilon)(\rho_{\max} - \rho_{min})}{M} \tag{3}$$

where $\varepsilon \in [0, \varepsilon_{\max}]$ represents a trimming ratio, accounting mostly for characteristic object features, which are eliminated from further consideration for watermark embedding. Such features would usually represent outliers in the given statistical representation of vector norms and following watermarking, their change could be visible. The value of $\varepsilon$ is generated according to the secret key. A study of robustness to crypto-attacks which depends on the number of bits embedded $M$ and on $\varepsilon_{\max}$ is provided in Section VI.

### B. Histogram Mapping Function

Histogram mapping was used for statistical watermarking in various methods including those from [5]–[7]. Each bin of vertex norms has a width given by $\rho_b$, calculated according to equation (3) with limits defined as $\rho_{i,min}$ and $\rho_{i,max}$. In the following we consider two methods for statistically embedding a message bit into a bin containing a set of vertex norms.

In the first embedding method, the statistical variables are firstly normalized to the range [0, 1] by using:

$$\tilde{\rho}_{ij} = \frac{\rho_{ij} - \rho_{i,min}}{\rho_{i,max} - \rho_{i,min}}. \tag{4}$$

As shown in [6], the distribution of the statistical variable $\tilde{\rho}_{ij}$ is usually close to a uniform distribution. Thus, the expected mean value of the statistical variable is $1/2$. In order to embed one bit we introduce a bias in the corresponding histogram by changing its mean value according to:

$$\hat{\mu}_i = \begin{cases} \frac{1}{2} + \alpha \text{ if } B_i = 1 \\ \frac{1}{2} - \alpha \text{ if } B_i = 0 \end{cases} \tag{5}$$

where $\alpha$ is the watermark strength factor influencing the visual distortion as well as the robustness, with a higher $\alpha$ providing higher robustness but more watermark visibility, while $B_i$, for $i = 1, \ldots, M$, is the bit to be embedded into the $\mathcal{B}_i$'s bin containing vertices according to (2). The first histogram mapping function changes the normalized distances, in order to enforce (5), as:

$$\tilde{\rho}'_{ij} = \tilde{\rho}^{\beta}_{ij} \tag{6}$$

where $\tilde{\rho}'_{ij}$ is the resulting watermarked normalized vertex norm, and $\beta \in (0, 1)$ when embedding a bit of 1, and $\beta \in (1, \infty)$ for a bit of 0. The watermarked vertex norms are obtained by mapping $\tilde{\rho}'_{ij}$ back to the original interval as:

$$\hat{\rho}_{ij} = \tilde{\rho}'_{ij}(\rho_{i,max} - \rho_{i,min}) + \rho_{i,min}. \tag{7}$$

The second method embeds the message by changing the variance of the vertex norm histogram. In this case, $\rho_{ij}$ is normalized to the range $[-1, 1]$ as:

$$\tilde{\rho}_{ij} = 2\frac{\rho_{ij} - \rho_{i,min}}{\rho_{i,max} - \rho_{i,min}} - 1. \tag{8}$$

The expected variance for a uniform distribution is $1/3$. We embed one bit by modifying the variance of $\tilde{\rho}_{ij}$ according to:

$$\hat{\sigma}_i^2 = \begin{cases} \frac{1}{3} + \alpha \text{ if } B_i = 1 \\ \frac{1}{3} - \alpha \text{ if } B_i = 0. \end{cases} \tag{9}$$

The histogram mapping function for modifying each element from the set $\mathcal{B}_i$ is defined in this case as:

$$\tilde{\rho}''_{ij} = sign(\tilde{\rho}_{ij})|\tilde{\rho}_{ij}|^{\beta} \tag{10}$$

where $\beta \in (0, 1)$ when embedding a bit of 1 and $\beta \in (1, \infty)$ for embedding a bit of 0.

The watermarked vertex norms are obtained by the inverse normalization function:

$$\hat{\rho}_{ij} = \frac{1}{2}(\tilde{\rho}''_{ij} + 1)(\rho_{i,max} - \rho_{i,min}) + \rho_{i,min}. \tag{11}$$

Firstly, a statistical variable from the watermarked distributions is sampled. Secondly, a chosen vertex from the corresponding bin is displaced such that its vertex norm fits the sampled statistical variable while fulfilling a surface preservation constraint criterion as well. In Section III we provide a new mesh surface error criterion in order to achieve a minimal 3D object distortion following 3D watermarking.

### C. Watermark Extraction

The watermark extraction algorithm is blind, *i.e.* it does not require the original object in the decoding stage. In the watermark decoding stage, after evaluating the object center $\mathbf{o}$, we use the watermark key in order to generate the percentage $\varepsilon$ for finding the vertices which had been eliminated from the watermark decoding procedure. Consequently, we calculate $\rho_b$, assuming known the number of embedded bits $M$, and we form bins with the corresponding extracted vertices. Then, the watermark code is extracted bit by bit, where for each bin, the given range of distances from the vertices to the object center are normalized according to either equation (4) or (8), for the mean or variance methods, respectively. For the first watermarking method, the code is extracted by applying a test on the mean value of the bin histogram:

$$\begin{cases} \text{if } \hat{\mu}_i > \frac{1}{2} \text{ then } B_i = 1 \\ \text{if } \hat{\mu}_i < \frac{1}{2} \text{ then } B_i = 0. \end{cases} \tag{12}$$

For the second watermarking method, the variance is calculated, and the bit is extracted following the test:

$$\begin{cases} \text{if } \hat{\sigma}_i^2 > \frac{1}{3} \text{ then } B_i = 1 \\ \text{if } \hat{\sigma}_i^2 < \frac{1}{3} \text{ then } B_i = 0. \end{cases} \tag{13}$$

## III. Distortion Metric Criterion for Surface Preserving Watermarking

Invisibility of the changes caused by watermarking is one of the major requirements for watermarking and in this section we propose a surface error minimization approach for 3D watermarking. Let us consider that we are provided with the statistical variable $\hat{\rho}_j$, sampled from the watermark based distributions as described in the previous section. Watermarking is defined by displacing a certain vertex $V_j$ to a new location $\hat{V}_j$ such that $\|O\hat{V}_j\| = \hat{\rho}_j$. We can observe that this corresponds to a statistical mapping on a set of spheres centered at $O$, while their radii are statistical variables $\hat{\rho}_j$ fulfilling the bit embedding conditions. Simultaneously, a surface distortion minimization condition is imposed such that the changes produced by the watermark would not be identifiable.

The constraints considered in the following consist of the distortion between the watermarked and the original surface, the smoothness of the resulting watermarked surface as well as the original vertex displacement. The cost function $E(\cdot) = \mathbf{f}^T\mathbf{f}$ is defined by a vector function $\mathbf{f} = (\sqrt{k_1}\mathbf{f}_1, \sqrt{k_2}\mathbf{f}_2, \sqrt{k_3}\mathbf{f}_3)^T$ with three components $\mathbf{f}_1$, $\mathbf{f}_2$ and $\mathbf{f}_3$ defined by weighting parameters $k_1, k_2, k_3$ such that $k_1+k_2+k_3 = 1$. The first error metric component $\mathbf{f}_1$ defines the distortion of the watermarked vertex with respect to the original surface as:

$$\mathbf{f}_1 = \begin{pmatrix} < (\hat{\mathbf{v}}_j - \mathbf{v}_j),\ \mathbf{n}_1 > \mathbf{n}_1 \\ \vdots \\ < (\hat{\mathbf{v}}_j - \mathbf{v}_j),\ \mathbf{n}_l > \mathbf{n}_l \end{pmatrix} \tag{14}$$

where $\hat{\mathbf{v}}_j$ is the location of the watermarked vertex $\hat{V}_j$, $< \cdot, \cdot >$ is the dot product and $\mathbf{n}_l$, $l = 1, \ldots, \mathcal{N}_{V_j}$, is the normal vector of a triangle adjacent to the vertex $V_j$ from the original surface with $\mathcal{N}_{V_j}$ representing the number of adjacent triangles to the vertex $V_j$. The vector $< (\hat{\mathbf{v}}_j - \mathbf{v}_j),\ \mathbf{n}_l > \mathbf{n}_l$ represents the projection of the vertex displacement along the orthogonal direction from $\hat{V}_j$ to the polygon $\mathbf{F}_l$, $l = 1, \ldots, \mathcal{N}_{V_j}$, which is adjacent to $\hat{V}_j$. Let us define $D(\hat{\mathbf{v}}_j, \mathbf{F}_l)$ as the distance from vertex $\hat{V}_j$ to the polygon $\mathbf{F}_l$. We can observe that

$$\mathbf{f}_1^T\mathbf{f}_1 = \sum_{l=1}^{\mathcal{N}_{V_j}} D^2(\hat{\mathbf{v}}_j, \mathbf{F}_l). \tag{15}$$

The metric $\mathbf{f}_1^T\mathbf{f}_1$ represents the Quadric Error Metric (QEM), *i.e.* the squared distance from the watermarked vertex to the original surface following the vertex displacement due to watermark embedding [13]. QEM was used for assessing the surface quality for mesh simplification by Garland and Heckbert [20] and was shown to have a lower computational complexity than the Hausdorff distance [21]. In the watermarking approach from [13], the vertex is displaced, onto the plane of a given adjacent triangle from the surface of the object, to the location of the intersection between the sphere centered in $O$, of radius $\hat{\rho}_j$, and the plane perpendicular on this triangle and containing $O$. The resulting displaced vertex minimizes QEM.

The second vector function component $\mathbf{f}_2$, is defined for measuring the distance of the watermarked vertex to the updated surface as:

$$\mathbf{f}_2 = \begin{pmatrix} < (\hat{\mathbf{v}}_j - \mathbf{v}_j),\ \hat{\mathbf{n}}_1 > \hat{\mathbf{n}}_1 \\ \vdots \\ < (\hat{\mathbf{v}}_j - \mathbf{v}_j),\ \hat{\mathbf{n}}_l > \hat{\mathbf{n}}_l \end{pmatrix} \tag{16}$$

where $\hat{\mathbf{n}}_l$, $l = 1, \ldots, \mathcal{N}_{V_j}$ is the normal vector of the polygon contained in the modified surface, neighboring the watermarked vertex $\hat{V}_j$ of location $\hat{\mathbf{v}}_j$, where we consider that the number of triangles and other polygons does not change following the watermark embedding. If there are no changes in the orientation of the planes corresponding to the watermarked object, in the neighborhood of $\hat{\mathbf{v}}_j$, which is unlikely in most cases, we would have:

$$Q_{\mathbf{v}_j}(\hat{\mathbf{v}}_j) = \mathbf{f}_2^T\mathbf{f}_2 = \mathbf{f}_1^T\mathbf{f}_1. \tag{17}$$

We consider $\mathbf{f}_2$ in order to ensure the resulting watermarked object surface smoothness and its consistency in appearance with the cover object surface. When updating the vertex $\mathbf{v}_j$, some of its neighbors may have been previously watermarked and consequently the local surface is displaced from its original position. The surrounding updated polygons are most likely no longer located on the same planes as the original ones. By considering only the component $\mathbf{f}_1$ in $\mathbf{f}$ we can obtain distorted surfaces following watermarking.

The third error function component $\mathbf{f}_3$ corresponds to the vertex displacement, representing the Euclidean distance between the watermarked and original vertex locations:

$$\mathbf{f}_3 = \hat{\mathbf{v}}_j - \mathbf{v}_j \tag{18}$$

such that the squared Euclidean distance between the original vertex $V_j$ and the watermarked vertex $\hat{V}_j$ is:

$$\mathbf{f}_3^T\mathbf{f}_3 = ||\hat{\mathbf{v}}_j - \mathbf{v}_j||^2. \tag{19}$$

The third constraint is added in order to compensate for the surface error which is not accounted for by the previous two error function components $\mathbf{f}_1$ and $\mathbf{f}_2$. For example, by choosing any point on the plane containing the original triangle as the updated vertex, will produce no errors with respect to the first and second error function components if none of their neighbors' locations are changed by watermarking. However, if the new vertex is displaced too far away from its original location on the plane, it will introduce a very large distortion to the object surface, although $\mathbf{f}_1^T\mathbf{f}_1 = \mathbf{f}_2^T\mathbf{f}_2 = 0$ in this case. Therefore, the third error function component is necessary as a dragging force in order to ensure the minimal displacement for the watermarked vertex from its original location.

Fig. 1 displays an example illustrating the three error function components for a single triangle on the surface of the object when displacing the vertex $V_j$ into $V_j'$ following watermark embedding. If only the distance to the original surface would be considered, *i.e.* the plane containing $\triangle V_j AB$, then $V_j''$ introduces the smallest distortion because it is located on the surface of the original object and consequently has $\mathbf{f}_1^T\mathbf{f}_1 = 0$. But obviously in the example from Fig. 1, the resulting $\triangle V_j'' A'B'$ is twisted, where $A'$ and $B'$ correspond to new locations following watermarking and may result into unpleasant visual effects for the object surface when considering the illumination of the graphical scene. This type of distortion will be prevented by considering the second error function component $\mathbf{f}_2$. The location of the new vertex must take into account simultaneously the location of the original surface and that of the updated surface in order to ensure an appropriate surface smoothness after watermarking. The user defined parameters $k_1$, $k_2$ and $k_3$ weigh the error with respect to the original surface, watermarked object surface and the Euclidean distance between the original and updated vertices. We can adjust these parameters in order to emphasize the importance of an error function component over the others according to specific watermarking requirements. When $k_2 = 0$ and $k_3 = 0$ we obtain the criterion used in [13] for watermarking, while when $k_1 = 0$ and $k_2 = 0$ we have the criterion used in [6]. A comprehensive empirical study about how to choose these parameters is provided in Section VII-B from the experimental results.

## IV. Optimal Vertex Placement for Watermark Embedding

In this section we describe how we embed the watermark by displacing a given vertex such that it minimizes the surface distortion criterion function $E(\cdot)$. The constraint that we have to enforce is $||O\hat{V}|| = \hat{\rho}_j$, *i.e.* the watermarked vertex should be on the sphere $\mathcal{S}(O, \hat{\rho}_j)$ centered in the object center $O$, of the radius given by the statistical variable $\hat{\rho}_j$ corresponding to the bit to be embedded and calculated as described in Section II. Simultaneously we aim to produce minimal distortion to the object surface. Due to the embedding symmetry with respect to object's center, in the following we consider the spherical coordinate representation for the vertex

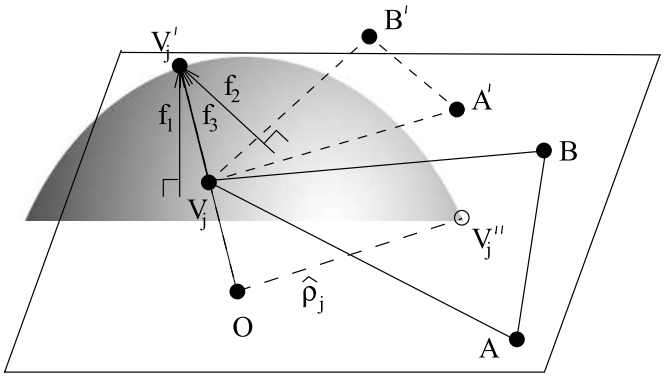



Fig. 1. The three components of the surface preservation criterion when following watermark embedding, a triangle $\triangle V_j AB$ is modified into $\triangle V_j' A'B'$. The statistical watermark variable corresponds to the sphere centered in $O$, with radius $||OV_j''|| = ||OV_j'|| = \hat{\rho}_j$. $A'$ and $B'$ are vertices displaced following watermark embedding from their original locations $A$ and $B$.

location $\hat{\mathbf{v}}$:

$$\hat{\mathbf{v}} = \begin{pmatrix} \hat{\rho}_j \cos\hat{\phi}\sin\hat{\theta} \\ \hat{\rho}_j \sin\hat{\phi}\sin\hat{\theta} \\ \hat{\rho}_j \cos\hat{\theta} \end{pmatrix}. \tag{20}$$

In the following we update the vector $\boldsymbol{\psi} = (\,\hat{\phi}\,,\,\hat{\theta}\,)^T$, while considering the vertex norm $\hat{\rho}_j$ as constant, while minimizing the criterion given by $E(\cdot)$.

Initially, we can select any location on the sphere of radius $\hat{\rho}_j$, centered in the object center, as the initial value. For example, we move the vertex $V$ to $\hat{V}$ along the direction of $\overrightarrow{OV}$ such that $||O\hat{V}|| = \hat{\rho}_j$, as it was considered in [6], as an appropriate initialization. Then, we use the iterative Levenberg–Marquardt optimization method in order to find the optimal vertex location minimizing the surface error $E(\cdot)$. Levenberg–Marquardt method [14], [15] represents an iterative gradient-descend minimization approach which solves nonlinear least square problems, subject to constraints and can be viewed as a Gauss–Newton method using a trust region approach which is robust to the choice of initialization as well. Levenberg–Marquardt firstly linearizes the given nonlinear problem by using a Taylor expansion around the vector $\boldsymbol{\psi} = (\hat{\phi}, \hat{\theta})^T$:

$$\mathbf{f}(\boldsymbol{\psi} + \mathbf{h}) = \mathbf{f}(\boldsymbol{\psi}) + \mathbf{J}\mathbf{h} \tag{21}$$

where $\mathbf{f}(\cdot)$ is the given surface distortion function, defined in the previous section and used here as a constraint criterion, and $\mathbf{h} = (\,\Delta\phi\,,\,\Delta\theta\,)^T$ is the step size. $\mathbf{J}$ is the Jacobian matrix of the vector function $\mathbf{f}$ and is calculated as:

$$\mathbf{J} = \begin{pmatrix} \dfrac{\partial \mathbf{f}^T}{\partial\hat{\phi}} \\ \dfrac{\partial \mathbf{f}^T}{\partial\hat{\theta}} \end{pmatrix}. \tag{22}$$

The optimal step $\mathbf{h}_k$ is calculated at each iteration $k$, in order to update the vector $\boldsymbol{\psi}_k$:

$$\boldsymbol{\psi}_{k+1} = \boldsymbol{\psi}_k + \mathbf{h}_k. \tag{23}$$

Levenberg–Marquardt uses iteratively the following equation in order to calculate $\mathbf{h}_k$:

$$(\mathbf{J}_k^T\mathbf{J}_k + \mu_k\mathbf{I})\mathbf{h}_k = \mathbf{J}_k^T\mathbf{f}_k \tag{24}$$

where $\mu_k > 0$ is the damping factor, $\mathbf{f}_k$ and $\mathbf{J}_k$ represent the surface distortion metric and its Jacobian, calculated at iteration $k$ with respect to $\boldsymbol{\psi}_k$. The initial value of the step $\mu_0$ is chosen as in [17]:

$$\mu_0 = 10^{-6} \max\{\text{diag}(\mathbf{J}_0^T \mathbf{J}_0)\} \tag{25}$$

where diag represents the diagonal of the matrix $\mathbf{J}_0^T \mathbf{J}_0$, which corresponds to the Jacobian matrix of $\mathbf{f}$ calculated for $\boldsymbol{\psi}_0$. $\mu_k$ is updated according to the approach proposed by Nielsen in [22]. The algorithm is not sensitive to the initial choice of $\mu_0$ as this value is continuously optimized by the updating procedure as in [14]. The damping factor serves two main purposes. Firstly, because $\mu_k > 0$, the coefficient matrix is positive definite, and this ensures that $\mathbf{h}_k$ leads to the minimization of the cost function in the descent error direction. Secondly, the damping parameter influences both the step size and the direction of the gradient descent. When $\boldsymbol{\psi}_k$ is close to the optimal value, the convergence rate of the Levenberg–Marquardt is almost quadratic. Levenberg–Marquardt was proved to converge and the optimization process is terminated when either the step size $\mathbf{h}_k$ becomes too small, or the error $E(\cdot)$ is too small, or when the loop exceeds a pre-set number of iterations [14], [17].

## V. Minimal Object Watermarking

Surface distortion produced by watermarking was analyzed in [3] for diverse surface variations. In the following we provide a minimal object watermarking analysis when employing the proposed watermark embedding approach. The sphere is known as the object of maximal compactness for a given volume and is chosen for this study due to its radial symmetry to its center as well. The watermarking methodology proposed in this paper as well as those of the approaches from [6], [13] are based on radial symmetry properties of objects with respect to their center and a sphere represents the simplest cover object for such methods.

In the following we consider the asymptotic analysis when embedding a single bit into a sphere of radius 1, whose vertices are distributed angular uniform on its surface, while assuming $\varepsilon = 0$ and the mean distribution embedding as in (5). When enforcing the minimization of the cost function $\mathbf{f}$, provided in Section III, the embedding of the watermark will produce displacements of the vertices along their radius which results into a series of bumps on the surface.

Let us consider the cross-section through a sphere as the circle of center $O$ and radius $R$ from Fig. 2. We consider a point $C$ on the sphere and assume that the embedded watermark is represented asymptotically as a set of semispheres-like bumps, one of them centered in $C$. By embedding the watermark in the sphere cover object by means of such bumps we ensure the enforcing of maximal radial change while preserving surface smoothness and fulfilling the constraints given in Section III. Let us consider that the watermark is characterized by six bumps with each of them semisphere-like, regularly placed on the surface of the sphere object. When considering placing 6 semisphere-like bumps onto a sphere, their radius is $r = R\sqrt{2-\sqrt{3}}$, where $R$ is the radius of the sphere cover object, for covering symmetrically the area all-round the sphere. Each of these bumps can be seen as located on top of a cone with the apex in the center of the sphere and characterized by an angle of $\pi/3$. When embedding a given watermark, a vertex $D$ on the surface of the sphere is displaced by a distance $d$ along the direction of $\overrightarrow{OD}$ and moved to a new location $A$. Let us consider an angle $\widehat{DOC} = \theta$ between the angular location on the original object for $D$ and that of $C$. After applying the sine and cosine theorems in $\triangle ACD$ and $\triangle OCD$ and by using other trigonometrical properties, we obtain the following distortion equation for the vertices embedding the watermark:

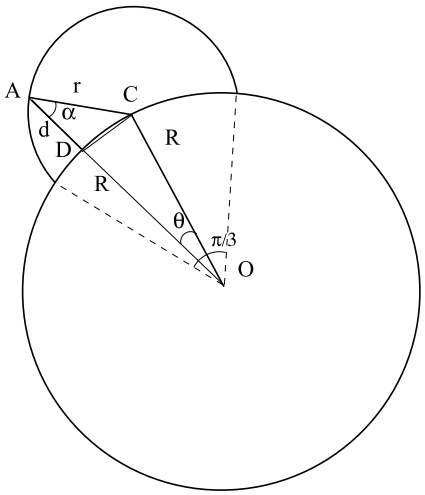



Fig. 2. Cross-section through a sphere cover object, where the watermark is embedded as a set of semisphere-like bumps on its surface.

$$d^2 = R^2\Bigg[4\sin^2(\theta/2)\left(1-\sqrt{1-\sqrt{3}+\cos^2(\theta)}\right) + 2\cos^2(\theta) - \sqrt{3}\Bigg]. \tag{26}$$

The resulting watermarked sphere surface is smooth and the only discontinuities on the watermarked object surface are located at the boundary between the semisphere-like bumps and the sphere cover object. For attenuating the distortion produced by these bumps we consider a scale factor $k > 1$ for replacing $R$ with $R/k$ in equation (26). The usage of $k$ is similar to that of the watermark strength factor $\alpha$ from (5). Examples of watermarked spheres, when embedding a bit of 0 and 1, using this watermarking function, are shown in Figs. 3(a) and (b), respectively, when considering two bumps, symmetrically placed and a scale of $k = 2$. In Figs. 3(c) and (d) we show the watermarked sphere when embedding a bit of 0 and 1, respectively, using ten bumps, placed along the intersection of the sphere with two perpendicular planes, using a scale of $k = 4$. For the sake of exemplification, the distortions on the surface of the sphere are visible in these figures. For a higher value for $k$ such distortions are less visible.

The distortion function is $\mathbf{f}^2 = 2d^2$ for each vertex displaced with $d$ from (26). In the case when considering the watermark embedded by a single semisphere-like bump, after integrating the difference on the entire sphere's surface we obtain a

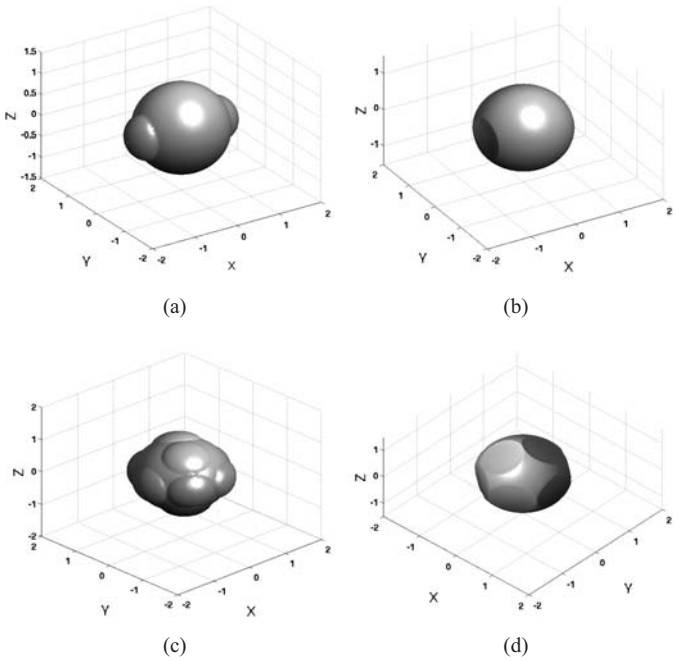



Fig. 3. Semisphere like bumps watermark embeddings on a sphere. (a) Embedding of bit 0 using 2 bumps and $k = 2$. (b) Embedding of bit 1 using 2 bumps and $k = 2$. (c) Embedding of bit 0 using 10 bumps and $k = 4$. (d) Embedding of bit 1 using 10 bumps and $k = 4$.

volume difference of about $0.11\pi R^3$ from the original sphere of radius $R$ and for $k = 1$. When considering several such bumps, the distortion would increase proportionally for the case when they are far away from each other, but less so when they would overlap. The safety detection difference (SDD) represents the normalized difference between the distribution of the distances from the watermarked object surface to the center and that of a uniform distribution, representing no watermark embedding. SDD represents the bias which would result from (12) after assuming the watermark embedding and possible attacks. SDD, when measured at the watermark embedding stage, characterizes the level of distortion that the watermarked object may undergo during an attack, while the watermark can still be appropriately decoded. SDD is given by:

$$SDD = \left| \hat{\mu}_s - \frac{1}{2} \right| \tag{27}$$

where $\hat{\mu}_s$ is the mean of the statistical variable characterizing the watermark. The bin is provided by the distances from vertices located on the surface of the watermarked object to the object center and they correspond to the range $[R, R \pm \frac{R}{k}\sqrt{2-\sqrt{3}}]$, where the sign is positive for a bit of 0 and negative for a bit of 1. A larger SDD would represent a more significant watermarked object distortion as well as a more robust watermark.

In the following we assess the robustness of the watermark, when watermarking a sphere using the distortion distance (26) and assuming a number of semisphere-like bumps $N = \{2, 4, 6, 8, 10\}$, symmetrically located on the surface of the sphere cover object and watermark attenuation $R/k$ for $k = \{1, \ldots, 8\}$. In all these cases it can be observed

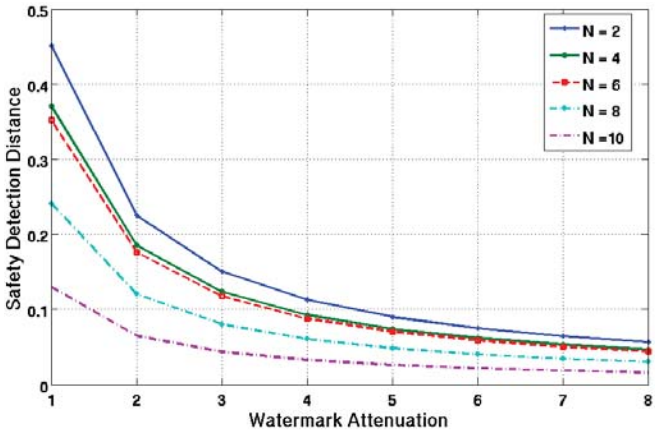



Fig. 4. SDD from (27) when watermarking a sphere using $N$ semisphere-like bumps, symmetrically located while assuming various attenuation factors $k$.

that the center of the watermarked object does not change due to the symmetry of the watermark embeddings. In Fig. 4 we provide the evaluation of SDD and we can observe that when embedding fewer bumps actually we have a larger SDD, while having a smaller distortion. This denotes a higher robustness of such watermarks to that of random changes to the object surface. However, fewer semisphere-like bumps watermarks can be easier identified by a crypto-attacker who may decode the watermark code.

## VI. Watermark Security

In this section we analyze the security of the proposed watermarking approach. We assume the Kerckhoffs's principle, according to which the watermark system is entirely known to a crypto-attacker aiming to break the watermark code. We have only one security parameter, $\varepsilon$, representing the trimming ratio used for removing the vertices that are either close to the object center or they are from the other extremity, according to (2).

A crypto-attacker would need to guess the value of $\varepsilon$ using exhaustive search, which given the nature of the watermarking process would actually be limited to a set of quantized values. The decoding by the attacker depends on the overlap between corresponding bins and on the actual sequence of bits. While overlaps of bins would not affect a set of identical bits because they are all characterized by the same vertex radii distributions in their bins, the worst case scenario for an attacker would be when the watermark code is a sequence of consecutive bits of 1 and 0. Let us assume in the following that the expectations of the mean for a bin from the region close to the object center as well as to the one that is far away, is uniform and consequently $E[\hat{\mu}] = \frac{1}{2}$. We assume $E[\hat{\mu}] = \frac{3}{4}$ for $B_i = 1$ and $E[\hat{\mu}] = \frac{1}{4}$ for $B_i = 0$, resulting into $SDD = \frac{1}{2}$ for both cases, according to (27). In Fig. 5 we evaluate the bias in the estimation of the mean of the vertices from bins estimated by a crypto-attacker attempting to break the watermark code. We consider that the watermark has 8 bits and we assume $\varepsilon = 0.15$. In Fig. 5, we assume that the attacker tries $\varepsilon \in \{0.04, 0.08, 0.12, 0.2\}$. A bias of $\frac{1}{4}$ or larger, corresponds to a uniform distribution or to a bit flip and would result into a wrongly estimated bit for

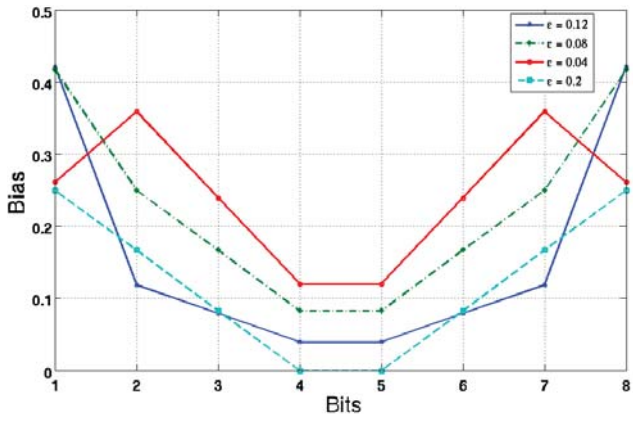



Fig. 5. Bias in the estimation of the means for each bin, corresponding to the attempts by a crypto-attacker to find a watermark code when considering that the embedded code has $M = 8$ alternating bits of 1 and 0, considering $\varepsilon = 0.15$.

the crypto-attacker. It can be observed from Fig. 5, that due to the symmetry of the trimming, according to equation (2), the middle bits are more likely to be identified during a crypto-attack.

It can be observed that following a crypto-attack, the resulting bins containing an overlap of half of bin between the actual distributions corresponding to two consecutive bins of 1 and 0, represents a failure for the attacker. The number of trials $\tau$, that are required for a cryto-attack on the watermark to succeed, depends on how well the attacker can guess the location of half bins and the number of embedded bits $M$, as follows:

$$\tau = \frac{\varepsilon(\rho_{\max} - \rho_{min})}{\frac{\rho_b}{2}} = \frac{2\varepsilon M}{(1 - 2\varepsilon)} \tag{28}$$

where we have used (3). This expression would result for $\varepsilon = 0.15$ into a number of 28 and 55 trials for a crypto-attacker, when assuming a number of 64 and 128 embedded bits, respectively. The number of required trials would double when using an asymmetric trimming for the vertices located at the extremities of the range of distances from vertices to the object center. In order to increase the security of the watermark, the solution would be to introduce additional key-generated parameters. For example we can consider a reference point, which is different from the center of the object. The location of such a reference point with respect to the center of the object would be represented by a shift vector, adding three new parameters to be generated by the security key. Another additional security parameter can be to introduce key-generated widths for mesh bands $\rho_b$ in equation (3), each band being used for embedding a single bit. A crypto-attacker would have to guess all these parameters and the total number of trials that he/she will have to perform would increase significantly to several thousands. Nevertheless, the location of the center would be limited to a specific region inside the object and the band width would have a minimal value in order to achieve a good watermark robustness as well.

## VII. Experimental Results

In the following we provide the experimental results when using the proposed methodology for watermarking 3D graphical objects represented as surface meshes. The experiments address both the watermark change visibility as well as the bit capacity, choice of parameters, robustness to various attacks, etc. In all experiments, excepting for the bit embedding capacity study, we embed $M = 64$ bits in each object. The watermark strength is set as $\alpha = 0.1$ in both equations (5) and (9), except for when studying the influence of the $\alpha$ parameter. The trimming ratio $\varepsilon$ used in equation (3) is considered bounded by $\varepsilon_{\max} = 0.15$ in order to ensure an appropriate balance between robustness, object distortion, bit-capacity and security. For each result we consider the embedding of one hundred different watermark codes into each graphical object and the final figure represents the averaging of all the corresponding results. In the experiments we compare the results provided by the watermarking method ensuring surface preservation based on the Levenberg–Marquardt optimization method, as described in Section IV, (denoted as L–M) with the methods proposed by Cho *et al.* [6], and the method using quadric error metric (QSP), which was proposed in [13]. All these methods rely on using the object center as the reference. For each method we employ two different statistical approaches, corresponding to modifying the mean or the variance of the histogram of distances from vertices to the object center, according to either equation (5) or (9) and these are identified by appending either "Mean" or "Var" to the name of the watermarking method.

### A. Experimental Models and Their Surface Preservation Assessment

The proposed methodology was applied on several graphical objects and in this paper, we provide the results when hiding digital information into six 3D graphical models shown in Fig. 6: *Bunny*, *Fish*, *Gear*, *Dragon*, *Buddha* and *Head*. The number of vertices and polygons for each of these graphical objects are provided in Table I. The meshes of these objects display a large variation of surfaces and shapes. *Bunny* is a small mesh object which is widely used in computer graphics experiments. *Gear* is a CAD object which contains large flat regions and sharp corners as well as circular holes. Such objects may have regions which can be modified only up to the admissible tolerance or even can not be modified due to their functionality, and watermarking such objects would require additional constraints. *Dragon* is considered as a large 3D model, considering its numbers of vertices and polygons, which has high variance in certain regions. *Buddha* has a complex surface variation and contains irregular holes. *Head* is an ellipsoidal like object with many smooth surfaces which has human features.

Surface distortion assessment is essential when evaluating 3D object watermarking methodology. We measure the surface quality of the watermarked objects using the method proposed by Cignoni *et al.* in [23] which represents an approximation of the Hausdorff distance for 3D mesh surfaces. This method employs a criterion which evaluates the forward $E_f(\mathcal{O}, \hat{\mathcal{O}})$

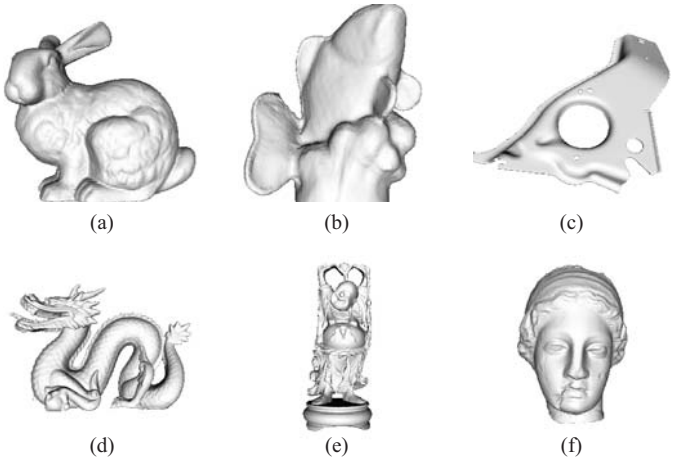


Fig. 6. 3D Models used in the experiments. (a) *Bunny.* (b) *Fish.* (c) *Gear.* (d) *Dragon.* (e) *Buddha.* (f) *Head.*

TABLE I
CHARACTERISTICS OF THE 3D MODELS USED IN EXPERIMENTS

| Models | No. of Vertices | No. of Polygons |
|---|---|---|
| *Bunny* | 34 833 | 69 449 |
| *Fish* | 64 982 | 129 664 |
| *Gear* | 231 703 | 463 430 |
| *Dragon* | 422 335 | 844 886 |
| *Buddha* | 89 544 | 179 222 |
| *Head* | 100 759 | 201 514 |

and the backward $E_b(\hat{\mathcal{O}}, \mathcal{O})$ root mean square (RMS) errors between the surfaces of the original graphical object $\mathcal{O}$ and the watermarked graphical object $\hat{\mathcal{O}}$. The maximum root mean square (MRMS) value is used as the measure of distortion:

$$E(\mathcal{O}, \hat{\mathcal{O}}) = \max\{E_f(\mathcal{O}, \hat{\mathcal{O}}), E_b(\hat{\mathcal{O}}, \mathcal{O})\} \tag{29}$$

where:

$$E_f(\mathcal{O}, \hat{\mathcal{O}}) = \frac{\sum_{\mathbf{v}\in\mathcal{O}} \min_{\hat{\mathbf{v}}\in\hat{\mathcal{O}}} \|\mathbf{v} - \hat{\mathbf{v}}\|}{|\mathcal{O}|} \tag{30}$$

$$E_b(\hat{\mathcal{O}}, \mathcal{O}) = \frac{\sum_{\hat{\mathbf{v}}\in\hat{\mathcal{O}}} \min_{\mathbf{v}\in\mathcal{O}} \|\hat{\mathbf{v}} - \mathbf{v}\|}{|\hat{\mathcal{O}}|} \tag{31}$$

where $\mathbf{v} \in \mathcal{O}$ and $\hat{\mathbf{v}} \in \hat{\mathcal{O}}$ represent vertices from the original object and from the watermarked object, respectively, while $|\mathcal{O}|$ represents the number of vertices from the graphical object $\mathcal{O}$. We can observe from these formulae that this error measure is a good indicator of the local distortions such as spikes that may emerge following the 3D surface watermarking. Such significant changes to the object surface are undesirable to be produced by watermarking. However, the Hausdorff distance does not evaluate how spread are the surface errors on a specific object and the visual surface assessment is necessary to be undertaken for a full surface evaluation.

TABLE II
EVALUATING THE MRMS ERROR, ACCORDING TO (29), WHEN CHANGING $k_1$, $k_2$, AND $k_3$, WHERE THE RESULTS SHOULD BE MULTIPLIED WITH $10^{-4}$

| Parameters | | | *Bunny* | | *Gear* | |
|---|---|---|---|---|---|---|
| $k_1$ | $k_2$ | $k_3$ | L–MMean | L–MVar | L–MMean | L–MVar |
| 1.0 | 0.0 | 0.0 | 0.44 | 0.17 | 949.09 | 428.09 |
| 0.0 | 1.0 | 0.0 | 0.46 | 0.18 | 1025.21 | 355.11 |
| 0.0 | 0.0 | 1.0 | 1.18 | 0.62 | 1861.15 | 1024.07 |
| 0.5 | 0.5 | 0.0 | 0.41 | 0.18 | 654.57 | 303.15 |
| 0.5 | 0.0 | 0.5 | 1.10 | 0.59 | 1546.87 | 862.52 |
| 0.0 | 0.5 | 0.5 | 1.17 | 0.62 | 1817.69 | 998.94 |
| 0.45 | 0.45 | 0.1 | 0.69 | 0.36 | 621.97 | 313.10 |
| 0.4 | 0.4 | 0.2 | 0.88 | 0.46 | 1007.14 | 537.90 |
| 0.49 | 0.49 | 0.02 | 0.40 | 0.21 | 410.57 | 148.12 |

### B. Evaluating the Parameter Setting

In this section we analyze the effect when varying the weighting parameters for the error function components used in the optimization process outlined in Section IV when applied on the cost function described in Section III. The parameters $k_1$, $k_2$ and $k_3$ weigh the error function components $\mathbf{f}_1$, $\mathbf{f}_2$ and $\mathbf{f}_3$ characterizing the errors of the updated vertex with respect to the original surface, watermarked surface and the Euclidean distance to the original vertex location, respectively. Various parameter combinations ($k_1$, $k_2$, $k_3$), emphasizing one or another of the error components, are listed in the first column of Table II. L–MMean and L–MVar denote the watermarking methods using the error cost function minimization by Levenberg–Marquardt, as described in Section IV, when employing either equation (5) for mean change, or equation (9) for variance change. The results are evaluated according to the distortion produced to the 3D object surface, measured using the maximum mean root squared error (MRMS) provided in equation (29). Table II provides the MRMS results for two of the graphical objects: *Bunny* and *Gear*. The first three sets of parameters consider the effects of each cost function component individually, while equating the contribution of the other two to zero. When considering only the Euclidean distance with respect to the original vertex, *i.e.* for $k_3 = 1$ and $k_1 = k_2 = 0$, the resulting surface distortion is much larger. For this parameter combination it can be observed that the watermarking method becomes identical with the method of Cho *et al.* described in [6]. For the parameter setting of $k_1 = 1$ and $k_2 = k_3 = 0$, we have the QSP approach from [13]. The results show that if the errors with respect to the original surface and the watermarked surface are considered simultaneously and equally weighted, the resulting distortion is much smaller than the other two cases when using two error components. This indicates that the significance of the first two error function components $\mathbf{f}_1$ and $\mathbf{f}_2$ should outweigh the third error function component, respectively $\mathbf{f}_3$. According to Section III, the error function components with respect to both the original surface and the updated surface must be considered. The fourth parameter selection provides less surface distortion than the first and the

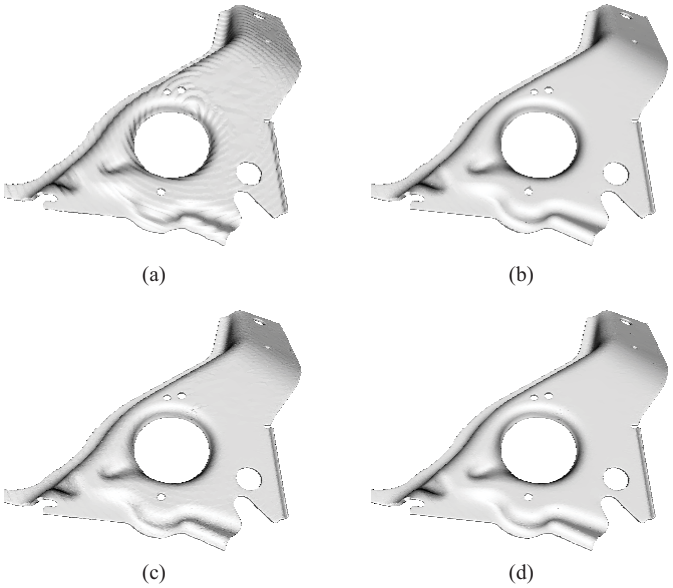


Fig. 7. Visual effects for various $k_1$, $k_2$, and $k_3$ settings. (a) $k_1 = 0.0$, $k_2 = 0.0$, $k_3 = 1.0$. (b) $k_1 = 0.5$, $k_2 = 0.5$, $k_3 = 0.0$. (c) $k_1 = 0.45$, $k_2 = 0.45$, $k_3 = 0.1$. (d) $k_1 = 0.49$, $k_2 = 0.49$, $k_3 = 0.02$.

second cases, according to these experiments. We found that the last case, where $k_1 = k_2 = 0.49$ and $k_3 = 0.02$, provides the minimum surface distortion according to the MRMS criterion. This result verifies our error function construction, as described in Section III, while underlying the importance of minimizing the error with respect to the original object surface together with enforcing the smoothness of the resulting watermarked object surface, nevertheless without neglecting the error with respect to the original vertex location. Fig. 7 illustrates various visual effects on the *Gear* object when using different configurations of $k_1$, $k_2$ and $k_3$. In the rest of the paper, we consider the configuration of $(k_1, k_2, k_3) = (0.49, 0.49, 0.02)$ for the weighting parameters because this provides the best watermarked object surface quality.

### C. Evaluation of Surface Distortion

A very important requirement for hiding digital information into graphical objects consists of achieving a minimal surface distortion such that it is not visible to an outsider. Table III compares the distortions introduced by the watermarking methods proposed in this paper and the Cho's methods under the same experimental settings. For each method we use both statistical approaches corresponding to equation (5) for mean change and equation (9) for variance change. We use MRMS, proposed in [23] and provided in equation (29), as the numerical distortion measure for comparing various watermarking methods. As it can be observed from these results, L–MMean and L–MVar provide better surface preservation results than the other methods. There was no object surface distortion variation observed when changing the watermark code embedded.

Fig. 8 shows the visual differences among the watermarked *Bunny* object when employing the methods based on mean histogram updating according to equation (5). The surface

TABLE III
WATERMARKED OBJECT DISTORTION MEASURED BY MRMS, WHERE ALL THE FIGURES SHOULD BE MULTIPLIED WITH $10^{-4}$

| Object | L–MMean | L–MVar | QSPMean | QSPVar | ChoMean | ChoVar |
|---|---|---|---|---|---|---|
| *Bunny* | 0.40 | 0.21 | 0.43 | 0.25 | 1.18 | 0.62 |
| *Fish* | 0.12 | 0.06 | 0.15 | 0.07 | 0.48 | 0.24 |
| *Gear* | 409.59 | 148.16 | 679.46 | 212.11 | 1860.67 | 1023.67 |
| *Dragon* | 0.29 | 0.13 | 0.36 | 0.15 | 1.09 | 0.57 |
| *Buddha* | 0.29 | 0.16 | 0.27 | 0.16 | 0.91 | 0.47 |
| *Head* | 0.12 | 0.06 | 0.13 | 0.06 | 0.32 | 0.16 |

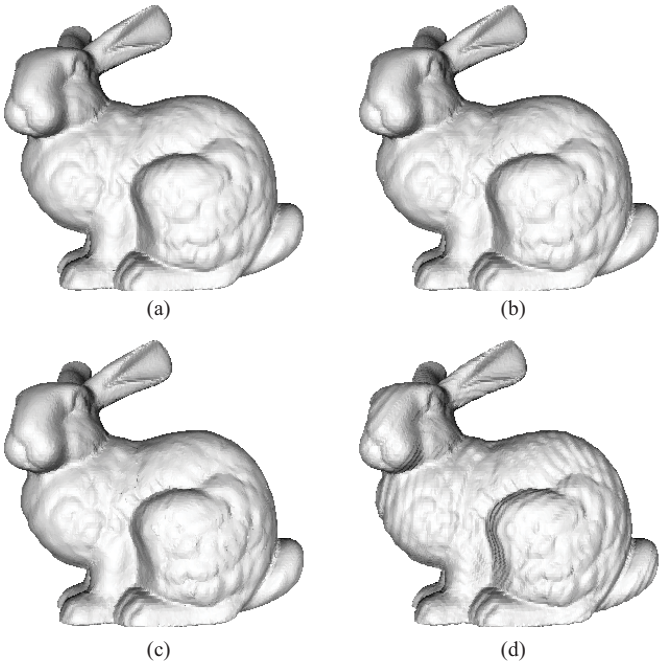


Fig. 8. Distortions produced when watermarking the *Bunny* object. (a) Original object. (b) L–MMean. (c) QSPMean. (d) ChoMean.

distortion minimization methods proposed in this paper produce much smaller distortion than Cho's method. As it can be observed from Fig. 8(d), Cho *et al.* [6] method introduces ripple like distortions to the surface of graphical objects. Fig. 9 compares close details of the distortions produced by L-M and QSP methods. From these images of details we can observe that the surface error minimization methods using Levenberg–Marquardt produce better results than the methods based on the QSP approach, proposed in [13].

### D. Watermark Robustness Assessment

The proposed watermarking methodology is robust against attacks that do not distort the graphical object surface including the affine transformations, vertex reordering, etc. However, the proposed information hiding methodology is not robust to changes in the object center. Such object center changes can be produced by object cropping, when applying significant asymmetric scaling or by adding localized noise of high amplitudes. However, such large changes may result into the destruction of

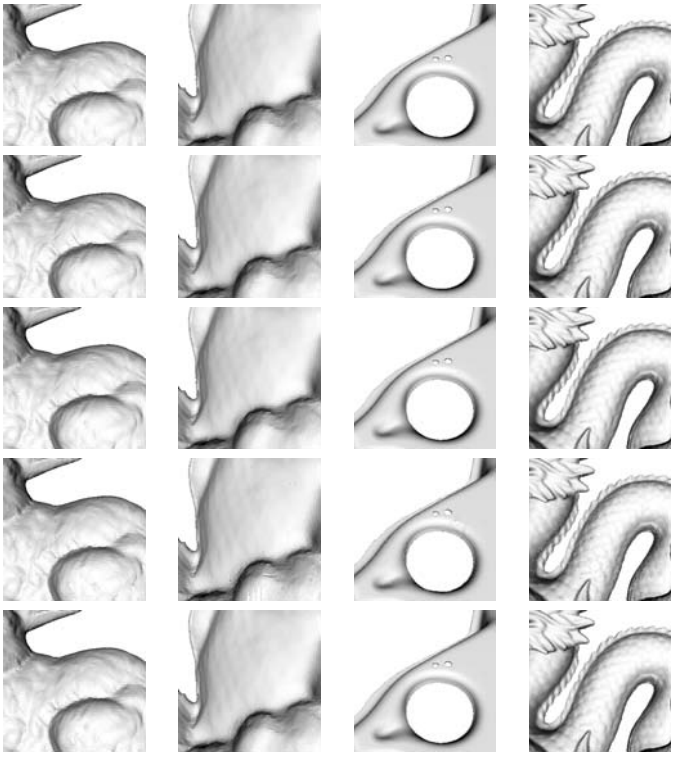


Fig. 9. Visual comparison of graphical object details following watermarking. From the first row to the bottom, the following are represented on each row: original objects and watermarked by L–MMean, L–MVar, QSPMean and QSPVar, respectively. (a) *Bunny*. (b) *Fish*. (c) *Gear*. (d) *Dragon*.

the graphical object. In the following, we assess the robustness against additive noise, Laplacian smoothing, mesh simplification, quantization and uniform resampling. We compare the results provided by the proposed methodology with those of Cho's method [6] for all six graphical objects. Each test result represents the average of bit error ratios (BER). BER is represented as a number in the range $[0, 1]$ representing the ratio of bits lost after each attack from the total number of embedded bits considering the embedding of one hundred different watermark codes into each object. Fig. 10 displays the effects on the *Bunny* model after certain attacks.

Additive noise is a common attack, which can be used to model other more complex mesh changes. In the following we consider additive random noise according to the following equation:

$$\tilde{\mathbf{v}}_i = \hat{\mathbf{v}}_i + \epsilon \|\hat{\mathbf{v}}_{\max}\| \overrightarrow{\mathbf{p}} \tag{32}$$

where $\tilde{\mathbf{v}}_i$ represents the noisy watermarked vertex $\hat{\mathbf{v}}_i$, $\epsilon \in [0, 1]$ is the percentage of $\|\hat{\mathbf{v}}_{\max}\|$, which corresponds to the largest Euclidean distance measured from the object center to all object vertices, and $\overrightarrow{\mathbf{p}}$ is a unitary vector of random direction. Fig. 10(a) shows the watermarked *Bunny* graphical object after noise addition with $\epsilon = 0.5\%$. The plots from Fig. 11 show the robustness against noise when varying $\epsilon = [0.1, 1]\%$ for all four methods (two using Levenberg–Marquardt optimization and two based on the Cho *et al.* approach from [6]) and for all six graphical objects under consideration. From these plots we can observe that the robustness of L–MMean is better in *Fish*

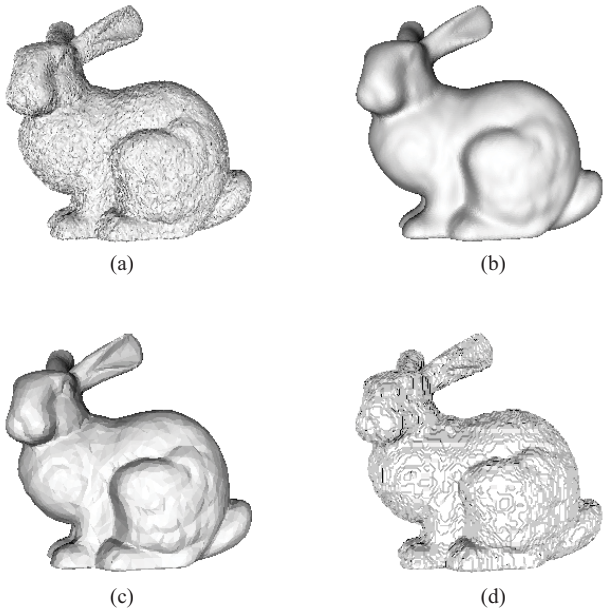


Fig. 10. Watermarked *Bunny* model after various attacks. (a) Noise $\epsilon = 0.5\%$. (b) Laplacian smoothing $\lambda = 0.5$, 10 iterations. (c) 90% mesh simplification. (d) 7 b quantization.

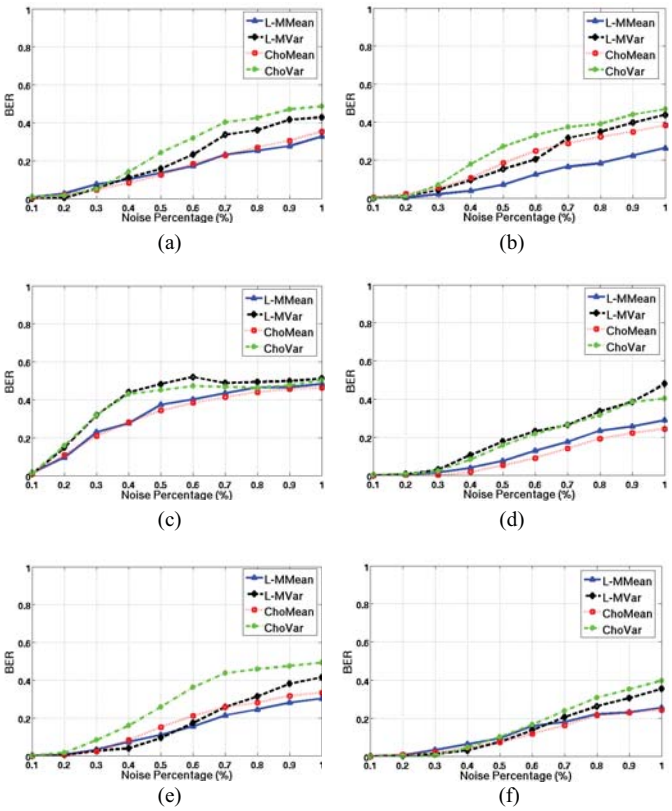



Fig. 11. Robustness against additive noise. (a) *Bunny*. (b) *Fish*. (c) *Gear*. (d) *Dragon*. (e) *Buddha*. (f) *Head*.

and *Buddha*, ChoMean is better in the *Dragon* object while the noise robustness results are similar in the other objects. The results on the *Gear* object are not that good as those achieved when attacking the other five objects because this is a CAD

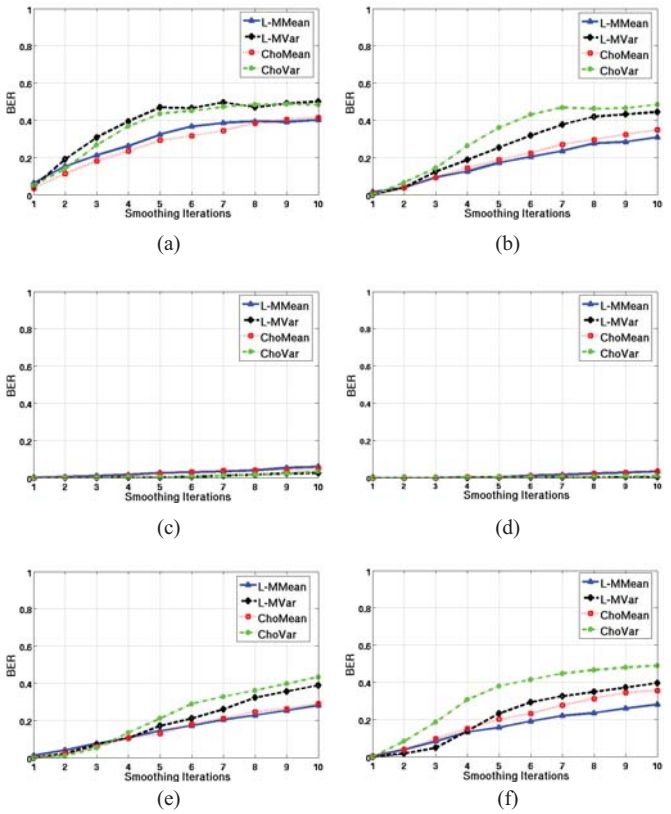



Fig. 12. Robustness against Laplacian smoothing. (a) *Bunny*. (b) *Fish*. (c) *Gear*. (d) *Dragon*. (e) *Buddha*. (f) *Head*.

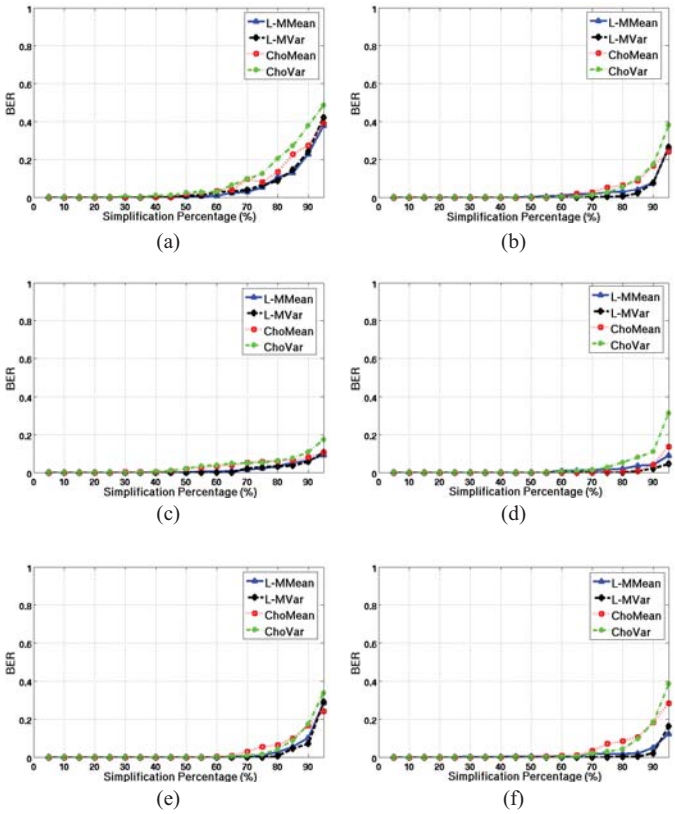



Fig. 13. Robustness against mesh simplification. (a) *Bunny*. (b) *Fish*. (c) *Gear*. (d) *Dragon*. (e) *Buddha*. (f) *Head*.

object containing large flat areas and sharp angles. Additive noise can easily destroy such distinctive features.

For the smoothing attack test we use the Laplacian algorithm proposed in [24]. A watermarked and smoothed *Bunny* when considering a smoothing parameter $\lambda = 0.5$ after 10 iterations is shown in Fig. 10(b). The robustness of the watermarking methods against the Laplacian smoothing when applied for 1 to 20 iterations with $\lambda = 0.5$ are provided in Fig. 12 for the six objects. The proposed methods show in general similar results to each other, but it can be observed that L–MVar is better than ChoVar in the *Fish*, *Buddha* and *Head* objects. It can be observed that the robustness to smoothing is higher for larger objects. The watermarked *Bunny*, which has fewer vertices than the other objects, is the least robust to the Laplacian smoothing attack among the six objects.

Mesh simplification is commonly used in computer graphics, particularly for graphics compression, and represents a challenging attack to watermarked 3D objects [3]. The quadric metric simplification software, described in [20], was used for testing the robustness at mesh simplification. Fig. 10(c) shows the watermarked *Bunny* object after 90% simplification. Fig. 13 provides the robustness to the simplification attack for the four methods and for the given six graphical objects. In these experiments, we test the robustness of the watermarking methods when varying the mesh simplification ratio between 5% and 95%. The proposed methodology performs excellently in the case of the mesh simplification attack. According to the results from Fig. 13, L–MMean method provides the best results for the *Fish*, *Gear*, *Dragon* and *Head* objects while L–MVar is better than ChoVar in most cases. This is because the L–M methods represent the watermarked object surface as being consistent with the original 3D object surface.

Fig. 10(d) shows the *Bunny* object attacked after 7 bits quantization. As shown in the plots from Fig. 14 all four algorithms are fairly robust up to applying 8 bits quantization attacks, while the histogram mean changing methods when using (5) perform usually better than methods that change the histogram variance when using (9), which is a result similar to that provided by the noise attack tests.

We compare the robustness of all four methods against the remeshing attack after uniformly sampling a percentage of 100%, 80%, 60%, 40% and 20% from all vertices of the original objects, using the method proposed in [25]. The new points are sampled from the tangent plane to the object surface and the mesh surface is reconstructed by remeshing. From the robustness plots shown in Fig. 15 we can see that L–MMean provides similar results to ChoMean for the *Bunny* object while its robustness results are better than the other methods for the other five objects. In these five objects L–MVar provided again better results than the ChoVar approach.

### *E. Computational Requirements*

Table IV provides the comparison of the timing required for embedding 64 bits when using all the watermarking methods considered, for all six graphical objects. The second and third columns of Table IV provide the average number of

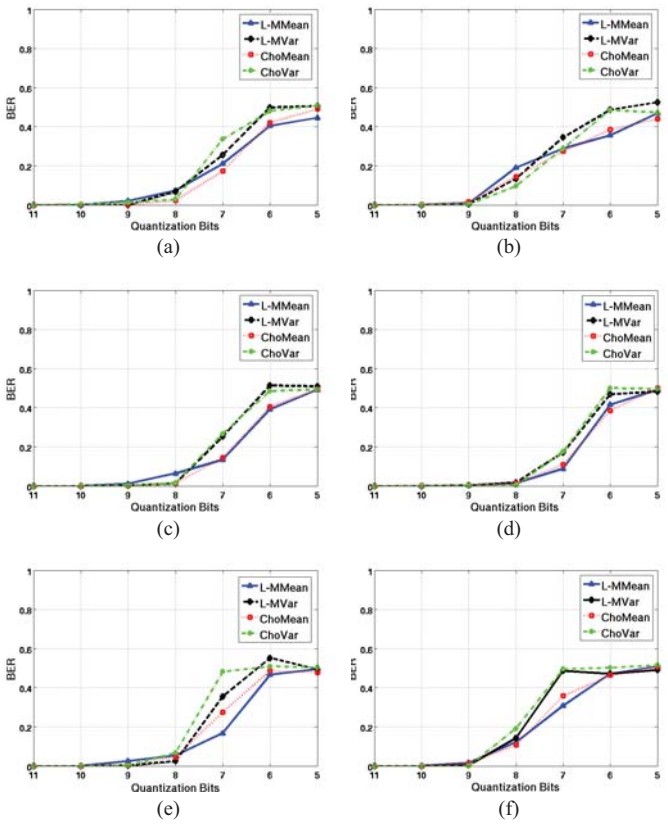



Fig. 14. Robustness against bit quantization. (a) *Bunny*. (b) *Fish*. (c) *Gear*. (d) *Dragon*. (e) *Buddha*. (f) *Head*.

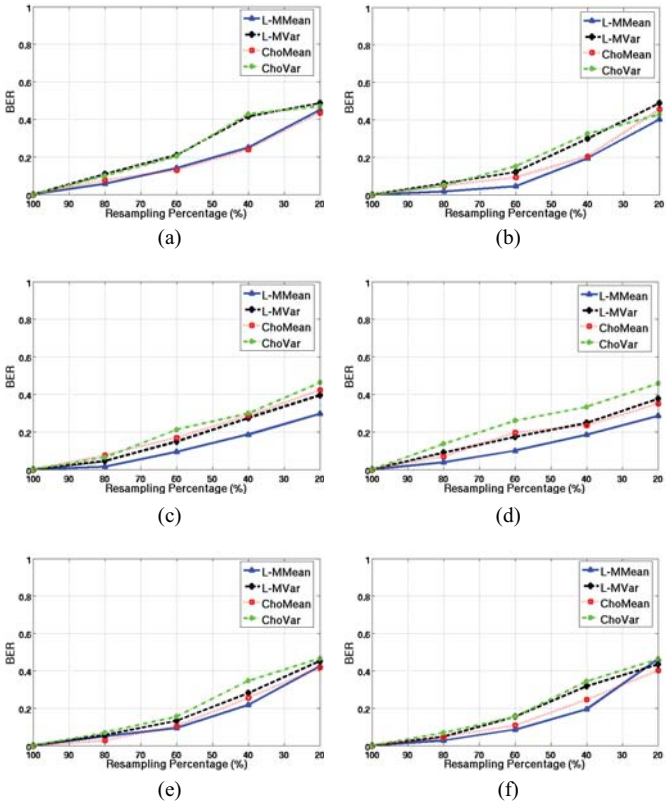



Fig. 15. Robustness against uniform resampling and remeshing where the horizontal axes show the percentages of original vertices. (a) *Bunny*. (b) *Fish*. (c) *Gear*. (d) *Dragon*. (e) *Buddha*. (f) *Head*.

iterations required by the Levenberg–Marquardt method for each model. In Fig. 16(a) we provide the convergence of the step function $\|\mathbf{h}_k\|$, where $k$ is the iteration, as well as that of the minimization of the infinity norm $||\mathbf{f}_k^T \boldsymbol{\psi}_k||_\infty$, which represents the entry with the largest absolute value from the matrix $\mathbf{f}_k^T \boldsymbol{\psi}_k$, evaluated for a typical vertex which is watermarked by using the L–M method at iteration $k$. From these results it can be observed that L–M method converges in just a few iterations. Fig. 16(b) shows the variation of the error function $E(\cdot)$ for the same vertex considered for Fig. 16(a). The Levenberg–Marquardt method automatically adjusts the step size and the step direction of $\mathbf{h}_k$, while testing whether the step size is appropriate. In this example, although nine iterations are required for convergence, the vertex is actually only moved once at step 5 where it can be observed that both $||\mathbf{f}_k^T \boldsymbol{\psi}_k||_\infty$ and the error $E(\cdot)$ are decreasing.

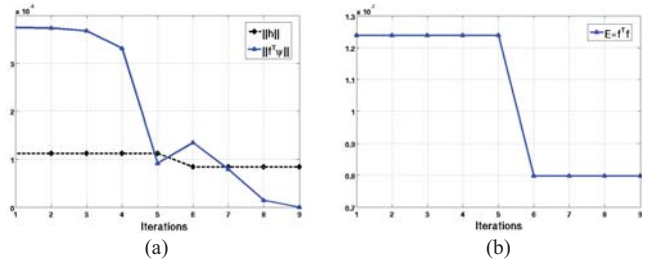



Fig. 16. Typical convergence of Levenberg–Marquardt for a vertex. (a) Variation of the step size $||\mathbf{h}||$ and $||f_k^T \psi_k||_\infty$. (b) Error function.

### *F. Evaluation of the Bit-Capacity and of the Embedding Strength Factor α*

In the following tests we consider the *Bunny* object whose characteristics are provided in Table I. Fig. 17(a) illustrates the relations between the 3D object surface distortion and the watermark strength factor $\alpha$. Fig. 17(b) shows the distortion variation with respect to the watermarking capacity. It can be observed that the distortion is increased when either $\alpha$ or the embedding capacity is increased. Figs. 18(a) and (b) provide the robustness plots when considering additive noise for L–MMean and QSPMean methods, respectively, while

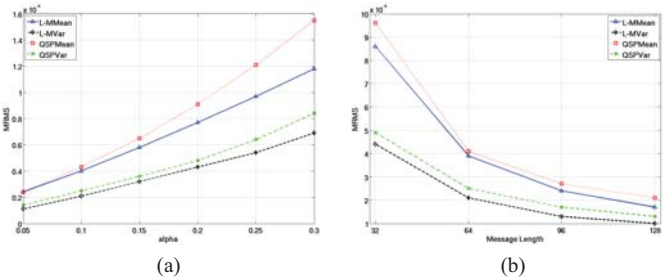


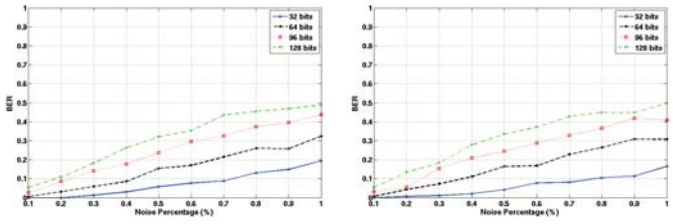

Fig. 17. Distortion with respect to the watermark strength factor $\alpha$ and bit-capacity. (a) Distortion when varying $\alpha$. (b) Distortion when increasing capacity.

increasing the embedded bit capacity. Figs. 19(a) and (b) provide the robustness plots when increasing the watermark strength factor $\alpha$ while considering additive noise for L–MMean and QSPMean methods, respectively.

TABLE IV

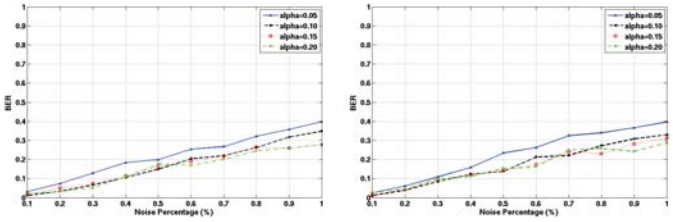
COMPARISON OF THE WATERMARK COMPUTATIONAL REQUIREMENTS. THE SECOND AND THIRD COLUMNS PROVIDE THE AVERAGE NUMBER OF ITERATIONS REQUIRED BY THE LEVENBERG–MARQUARDT METHOD UNTIL CONVERGENCE FOR THE MEAN AND VAR METHODS. THE COLUMNS 4–9 REPRESENT AVERAGE PROCESSING TIMES IN SECONDS

| Object | L–MMean Iter. (No) | L–MVar Iter. (No) | L–MMean (s) | L–MVar (s) | QSPMean (s) | QSPVar (s) | ChoMean (s) | ChoVar (s) |
|---|---|---|---|---|---|---|---|---|
| *Bunny* | 6.20 | 5.74 | 5.75 | 5.3 | 1.7 | 1.79 | 0.39 | 0.4 |
| *Fish* | 5.69 | 5.03 | 10.28 | 9.36 | 3.29 | 3.25 | 0.85 | 0.88 |
| *Gear* | 5.87 | 5.24 | 42.55 | 38.52 | 12.96 | 12.69 | 3.98 | 3.87 |
| *Dragon* | 5.88 | 5.35 | 77.55 | 69.78 | 22.82 | 22.4 | 6.09 | 6.67 |
| *Buddha* | 5.56 | 5.03 | 13.52 | 12.45 | 4.87 | 4.93 | 1.56 | 4.83 |
| *Head* | 5.96 | 5.29 | 16.21 | 14.44 | 4.88 | 4.91 | 1.09 | 4.9 |




(a) (b)

Fig. 18. Robustness to additive noise when increasing the bit capacity. (a) Robustness of L–MMean when varying capacity. (b) Robustness of QSPMean when varying capacity.

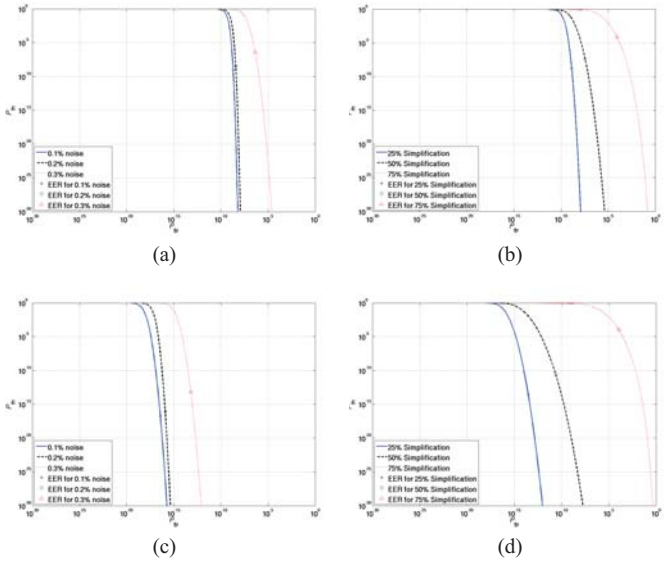



(a) (b)

Fig. 19. Robustness when increasing the watermark strength factor $\alpha$. (a) Robustness of L–MMean when varying $\alpha$. (b) Robustness of QSPMean when varying $\alpha$.

### G. Receiver Operating Characteristic Analysis

Receiver Operating Characteristic (ROC) curves represent the relation between the probability of false positives $P_{fp}$ and the probability of false negatives $P_{fn}$. The probabilities $P_{fp}$ and $P_{fn}$ are evaluated by varying the decision threshold in the correlations for the bits extracted from each of the objects when deciding whether the watermark is present or not, as in [1], [5]–[7]. The tests are performed for 100 correct and 100 wrong keys and the correlation results are approximated by Gaussian distributions. Figs. 20(a) and (c) show the ROC curves for the *Bunny* object for L–MMean and L–MVar when considering additive noise levels of $\{0.1\%, 0.2\%, 0.3\%\}$. Figs. 20(b) and (d) show the ROC curves for the *Bunny* object for L–MMean and L–MVar when considering mesh simplifications of $\{25\%, 50\%, 75\%\}$. The corresponding equal error rates (EER) are indicated in each plot from these figures. As it can be observed from Fig. 20, L–MMean provides better results than L–MVar and better than the ROC curve results from [6] and [7].




(a) (b)

(c) (d)

Fig. 20. ROC analysis. (a) and (b) L–MMean with noise and simplification attacks. (c) and (d) L–MVar with noise and simplification attacks.

## VIII. CONCLUSION

In this paper, we propose a new 3D watermarking methodology which minimizes the object surface distortions. The watermark is embedded into histograms of distances from the object center to vertices on its surface by slightly displacing the location of the vertices. The proposed method considers a novel surface error function consisting of three components measuring the distortion with respect to the original surface, the watermarked surface and the Euclidean distance from the original vertex location. This error function ensures both a minimal distortion with respect to the original surface as well as it enforces smoothness in the object surface resulting after watermarking. The 3D object surface distortion is minimized by using the Levenberg–Marquardt optimization method for vertices represented in spherical coordinates. We provide a study of minimal object watermark embedding and a study of the watermark security for the proposed methodology. The security of the watermark can be improved if additional key-generated parameters would be added. As shown in the experiments, the proposed watermarking methodology has high robustness against common mesh attacks while ensuring

a minimal surface distortion. The proposed methodology can be adapted for usage on other 3D object representations such as voxel-based or on parametric models after using quantization. The watermarks can be used for information hiding, copyright protection, for embedding information relevant to database organization as well as for embedding object-specific information which would be used during graphics animation.

## REFERENCES


[1] E. Praun, H. Hoppe, and A. Frinkelstein, "Robust mesh watermarking," in *Proc. SIGGGRAPH Conf.*, 1999, pp. 69–76.

[2] R. Ohbuchi, A. Mukaiyama, and S. Takahashi, "A frequency-domain approach to watermarking 3-D shapes," in *Proc. Eurograph., Comput. Graph. Forum*, 2002, pp. 373–382.

[3] A. G. Bors, "Watermarking mesh-based representations of 3-D objects using local moments," *IEEE Trans. Image Process.*, vol. 15, no. 3, pp. 687–701, Mar. 2006.

[4] 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.

[5] S. Zafeiriou, A. Tefas, and I. Pitas, "Blind robust watermarking schemes for copyright protection of 3-D mesh objects," *IEEE Trans. Visualizat. Comput. Graph.*, vol. 11, no. 5, pp. 496–607, Oct. 2005.

[6] J. W. Cho, R. Prost, and H. Y. Jung, "An oblivious watermarking for 3-D polygonal meshes using distribution of vertex norms," *IEEE Trans. Signal Process.*, vol. 55, no. 1, pp. 142–155, Jan. 2007.

[7] M. Luo and A. G. Bors, "Surface-preserving watermarking of 3-D shapes," *IEEE Trans. Image Process.*, vol. 20, no. 10, pp. 2813–2826, Oct. 2011.

[8] O. Benedens, "Geometry-based watermarking of 3-D models," *IEEE Comput. Graph. Appl.*, vol. 19, no. 1, pp. 46–55, Feb. 1999.

[9] M. Luo and A. G. Bors, "Principal component analysis of spectral coefficients for mesh watermarking," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2008, pp. 441–444.

[10] M. Kim, S. Valette, H. Jung, and R. Prost, "Watermarking of 3-D irregular meshes based on wavelet multiresolution analysis," in *Proc. Int. Workshop Digit. Watermark.*, 2005, pp. 313–324.

[11] J. M. Konstantinides, A. Mademlis, P. Daras, P. Mitkas, and M. G. Strintzis, "Blind robust 3-D mesh watermarking based on oblate spheroidal harmonics," *IEEE Trans. Multimed.*, vol. 11, no. 1, pp. 23–38, Jan. 2009.

[12] K. Kim, M. Barni, and H. Z. Tan, "Roughness-adaptive 3-D watermarking based on masking effect of surface roughness," *IEEE Trans. Inform. Forensics Security*, vol. 5, no. 4, pp. 721–733, Dec. 2010.

[13] M. Luo and A. G. Bors, "Shape watermarking based on minimizing the quadric error metric," in *Proc. IEEE Int. Conf. Shape Model. Appl.*, Jun. 2009, pp. 103–110.

[14] K. Levenberg, "An algorithm for least-squares estimation of nonlinear parameters," *Quart. Appl. Math.*, vol. 11, no. 2, pp. 164–168, 1944.

[15] D. Marquardt, "A method for the solution of certain non-linear problems in least squares," *SIAM J. Appl. Math.*, vol. 11, no. 2, pp. 431–441, 1963.

[16] K. Cheng, W. Wang, H. Qin, K. Wong, H. Yang, and Y. Liu, "Design and analysis of optimization methods for subdivision surface fitting," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 5, pp. 878–890, Oct. 2007.

[17] M. Eigensatz, R. Sumner, and M. Pauly, "Curvature-domain shape processing," in *Proc. Eurograph., Comput. Graph. Forum*, vol. 27, no. 2, pp. 241–250, 2008.

[18] M. Alvarez-Rodriguez and F. Perez-Gonzalez, "Analysis of pilot-based synchronization algorithms for watermarking of still images," *Signal Process.-Image Commun.*, vol. 17, no. 8, pp. 611–633, 2002.

[19] A. Tuzikov, S. Sheynin, and P. V. Vasiliev, "Computation of volume and surface body moments," *Pattern Recognit.*, vol. 36, no. 11, pp. 2521–2529, 2003.

[20] M. Garland and P. Heckbert, "Surface simplification using quadric error metrics," *SIGGRAPH, Graph. Models*, vol. 66, no. 6, pp. 370–397, 1997.

[21] P. Heckbert and M. Garland, "Optimal triangulation and quadric-based surface simplification," *J. Computat. Geometry*, vol. 14, nos. 1–3, pp. 49–65, 1999.

[22] H. B. Nielsen, "Damping parameter in Marquardt's method," Dept. Math. Modelling, Tech. Univ. Denmark, Lyngby, Denmark, Tech. Rep. IMM-REP-1999-05, 1999.

[23] P. Cignoni, C. Rocchini, and R. Scopigno, "Metro: Measuring error on simplified surfaces," *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, 1998.

[24] G. Taubin, "Geometric signal processing on polygonal meshes," in *Proc. Eurograph. State Art Rep.*, 2000, pp. 1–11.

[25] M. Attene and B. Falcidieno, "Remesh: An interactive environment to edit and repair triangle meshes," in *Proc. IEEE Int. Conf. Shape Model. Appl.*, Jun. 2006, pp. 271–276.


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

He was a Research Scientist with the Signal Processing Laboratory, Tampere University of Technology, Tampere, Finland, from 1992 to 1993. From 1993 to 1999, he was a Research Associate with the University of Thessaloniki. In 1999, he joined the Department of Computer Science, University of York, York, U.K., where he is currently a Lecturer. In 2006, he was a Visiting Scholar with the University of California at San Diego, and an Invited Professor with the University of Montpellier II, France. He has authored or co-authored more than 90 papers in international journals and conference proceedings. His current research interests include image processing, digital watermarking, pattern recognition, computational intelligence, computer vision, and nonlinear digital signal processing.

Dr. Bors has been an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING since 2010 and was an Associate Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS from 2001 to 2009. He is a member on the Organization Committee of the International Conference on Computer Analysis of Images and Patterns (CAIP), in 2013, and on the technical committees of several other international conferences.

**Ming Luo** was born in Qingdao, China, in 1983. He received the B.Sc. degree (Hons.) from Swansea University, Swansea, U.K., and the Ph.D. degree from the University of York, York, U.K., in 2006 and 2010, respectively, both in computer science.

He was a Research Scientist with the Department of Computer Science, the University of York, from 2006 to 2010. His current research interests include 3D digital watermarking, object recognition, and object tracking.