To appear in Proceedings of International Conference on Shape Modeling and Applications 2004

Robust Watermarking of Point-Sampled Geometry

Daniel Cotting[1] Tim Weyrich[1] Mark Pauly[2] Markus Gross[1]

[1]Computer Graphics Laboratory
Swiss Federal Institute of Technology (ETH)
Zurich, Switzerland
{dcotting, weyrich, grossm}@inf.ethz.ch

[2]Computer Science Department
Stanford University
Stanford, CA, USA
mapauly@graphics.stanford.edu

Abstract

We present a new scheme for digital watermarking of point-sampled geometry based on spectral analysis. By extending existing algorithms designed for polygonal data to unstructured point clouds, our method is particularly suited for scanned models, where the watermark can be directly embedded in the raw data obtained from the 3D acquisition device. To handle large data sets efficiently, we apply a fast hierarchical clustering algorithm that partitions the model into a set of patches. Each patch is mapped into the space of eigenfunctions of an approximate Laplacian operator to obtain a decomposition of the patch surface into discrete frequency bands. The watermark is then embedded into the low frequency components to minimize visual artifacts in the model geometry. During extraction, the target model is resampled at optimal resolution using an MLS projection. After extracting a watermark from this model, the corresponding bit stream is analyzed using statistical methods based on correlation. We have applied our method to a number of point-sampled models of different geometric and topological complexity. These experiments show that our watermarking scheme is robust against numerous attacks, including low-pass filtering, resampling, affine transformations, cropping, additive random noise, and combinations of the above.

1. Introduction

With the recent proliferation of 3D geometry in scientific and multi-media applications, methods for copyright protection and ownership assertion are gaining increasing attention. A common and widely used method for ownership authentication is digital watermarking. The idea is to encode the ownership information directly into the data, thus modifying the original data by embedding a digital signature. Whenever the rightful ownership is in dispute, the embedded watermark can be extracted from the data. Since in a private watermarking scheme this requires possession of the original unmarked object and the secret watermark key, only the true owner can assert his claim.

In this respect any watermarking method faces two competing goals. On the one hand, the watermark should not degrade the data, e.g. significantly alter its visual appearance. On the other hand, the watermark should be as robust as possible, i.e. the extraction of the signature should be stable, even under malicious attacks on the watermarked data. A pirate might modify the original data with the sole intent to destroy the watermark, for example, by applying filtering or resampling operations. A watermarking scheme is considered robust, if the successful removal of a watermark by these attacks leads to a severe degradation of the data, i.e. renders it useless for most applications.

To achieve better robustness against most common attacks, existing watermarking schemes for functional data (sound, images, video) often first apply some sort of transformation on the original dataset, e.g. a spectral decomposition, and embed the watermark in the coefficients of the transformed data.

1.1. Related work

Due to their simplicity in data structure, images and audio files were the first domains, watermarking procedures were developed for. Nowadays, suited algorithms are well explored, and combined systems for multi-media and image data have existed for a long time [38]. A plenitude of different approaches has been developed, image watermarking in frequency domain [13, 6, 32, 25] soon superseding approaches in spatial domain [24, 34, 16] due to its higher robustness. A comparable development has occurred in the field of 3D mesh watermarking: Watermarking systems encoding in frequency domain are generally preferred to approaches working directly on the vertex position information [17, 36, 2, 27, 3].

Adapting the spectral decomposition to discrete geometry is challenging, since neither global parameterization, nor regular sample distribution are given for 3D models in general. Yet spectral transforms, such as the Fourier transform, heavily rely on these prerequisites. Taubin was the first to define a spectral decomposition for 3D models represented by polygonal meshes based on the eigenvector decomposition of a discrete Laplacian matrix [33]. By considering the spectral basis functions as eigenfunctions of the Laplacian operator, generalized geometric frequencies can be defined with respect to different approximations of the Laplacian on manifolds. This can be understood as a generalization of the Fourier transform, where the spectral basis functions can also be defined as eigenfunctions of the Laplace operator.

But contrary to the regular functional setting, where the basis functions are given analytically, for irregularly sampled manifolds the spectral basis functions have to be computed explicitly by solving the eigenproblem of the Laplacian.

Karni and Gotsman applied this scheme to implement a spectral mesh compression algorithm [12]. Soon, first watermarking schemes were developed, which rely on spectral representations of 3D meshes [11, 26]. Ohbuchi et al. [18, 19] present an approach using the Laplacian matrix and achieve robustness against similarity transformations, random noise, low-pass filtering, mesh simplifications and cropping.

1.2. Contributions

We present a watermarking system for 3D models represented by unstructured clouds of point samples. Point sets have become a popular surface representation for large complex geometric objects due to their simplicity and flexibility in modeling arbitrary shapes [22, 10, 39]. Since point clouds are the basic output of most 3D acquisition devices, our method can be directly applied to scanned data. As 3D data acquisition is often expensive and time-consuming, there is particular interest to ensure copyright protection on these raw data sets.

Our system is an extension of previous work by Ohbuchi [18, 19], adapting specific parts to point clouds: A new patching algorithm is introduced, mesh connectivity is replaced by a *k*-nearest neighbor search, and a resampling method based on moving least squares (MLS) has been developed. Furthermore, innovative new features have been added, like an affine registration and a generalization of watermark encoding to arbitrary attributes. Because appearance parameters are directly stored for every point sample, we introduce a unified, integrated treatment of geometry and additional attributes like texture.

Note that our scheme is general in the sense that it can directly be applied not only to point clouds, but also to other discrete surface representations such as triangle meshes or spline-patches by simply taking the set of vertices or control points as input. The additional structural data of these surface representations, e.g. the connectivity graph of a mesh, carries additional information that mostly affects the parameterization, not the geometry of the surface. Since one of our goals is to be robust against resampling (or remeshing) our method does not take parametric (or connectivity) information into account when embedding the watermark. Instead, we build a Riemannian graph of the point samples solely based on Euclidean distance. Thus our method uses only information that is intrinsic to the data and does not rely on a specific connectivity of the samples.

By encoding watermarks into low frequencies, the procedure is robust against a multitude of attacks, including random noise and smoothing. Resampling is effectively undone by an MLS projection, and similarity and affine transformations are reversed by an automatic PCA alignment or our user-guided affine registration. Additionally, patching achieves robustness against cropping of the geometry. Since the watermarks are robust against further embeddings, the watermarking system is suited for hierarchical marking as well.

The system presented in this paper fulfills all the requirements of an effective watermarking procedure and, for the first time, it enables to sign not only meshes, but also point-based 3D geometries to assure ownership rights.

2. Overview of the Method

The watermarking pipeline consists of two separate pipelines: an embedding pipeline and an extraction pipeline, both containing five pipeline stages (cf. Figure 1). The conceptual flow is similar to Ohbuchi's procedure [18, 19].

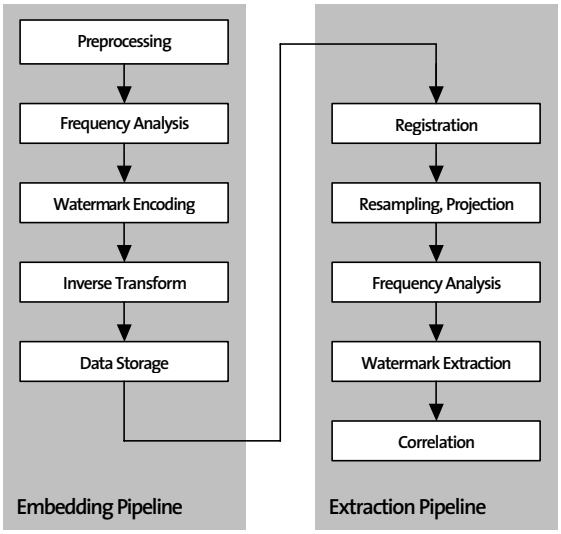

Figure 1: An overview: The embedding pipeline and the extraction pipeline.

The embedding process proceeds as follows: In a preprocessing step the model is decomposed into a set of disjoint patches. Each patch is transformed into the frequency domain, where the watermark is encoded into the spectral coefficients. The final watermarked model is then obtained by applying an inverse transform to each patch.

Our watermarking scheme is private, i.e. the original model is required during the watermark extraction performed on a given point cloud. We believe that this is a reasonable assumption in the context of ownership assertion, which is the main focus of our method.

During extraction the following steps are performed: The original object and the potentially marked test object are aligned using a registration process. Then the surface of the test object is resampled with the resolution of the original object. After a frequency transformation, the watermark is extracted using both the resampled test geometry and the original object. Based on the correlation between extracted and original watermark, ownership can be asserted or disproved.

3. Watermark Embedding

The embedding pipeline of our watermarking system proceeds in multiple steps as described in the next subsections.

3.1. Preprocessing: Patching

The watermark is not applied to the entire 3D object in one step, but rather to subsets of the data.

There are several reasons for performing this partitioning.

- Using multiple patches allows a watermark to be extracted even in case of large deformations and significant attacks on the signed object as long as one single patch is sufficiently preserved and can be registered with the original object. Patching also helps when watermarks have to be extracted from cropped objects.
- A variable patch size can accommodate different security requirements: On the one hand, small patches increase the robustness against cropping attacks. On the other hand, large patches allow higher redundancy in the bit encoding, leading to increased resilience against random noise. A variable, user-defined patch size is therefore desirable.
- Patches ensure the scalability of the watermarking procedure. The time and space efficiency of the algorithm using patches is higher than for the same procedure defined over the entire point cloud. This is due to the fact that memory consumption for the frequency transformation of large models exceeds the capabilities of today's computers.
- Additionally, patching increases the numerical stability of the procedure, because the eigenvalue problem of the frequency transformation can be solved on smaller matrices.

To create the patches, we use a top-down approach, which recursively splits the point set until a suitable patch size is obtained. The principal component analysis (PCA) is used as a core element of our patching, inspired by previous work by Pauly et al. [21] and Shaffer et al. [29]. For every point set of the recursive procedure, the principal component is determined. Space is then subdivided by a plane which is perpendicular to the direction of greatest variation.

To determine the minimum patch size we observe that every watermark bit is embedded c times (chip-rate), and only a fraction $1/u$ of the frequencies contained in a patch is used for embedding bits. Furthermore, for a frequency transformation as many points are needed as resulting frequencies are required. Therefore, for a watermark length w (up to 160 bit), the minimum patch size is $u \cdot c \cdot w$ points.

No additional criteria like boundary conditions are considered. Potential discontinuities at the boundaries of the patches are not visible as shown in Section 6.2. Even distant groups of points may belong to the same patch, without impacting the results in a negative way.

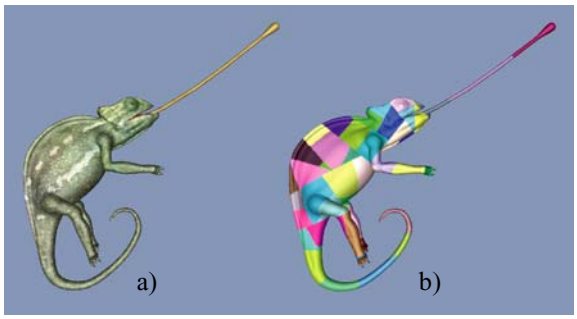

Figure 2: Point cloud model before (a) and after (b) the patching procedure. Each patch is highlighted with a random color.

3.2. Frequency Analysis

Our watermarking scheme embeds the digital signature by modifying the coefficients of a spectral representation of the point-sampled model. Pauly and Gross [20] presented such a representation that splits the model into patches, computes a parameterization and regular sampling grid for each patch, and then applies a standard 2D Fast Fourier Transform on the discrete patch function. We also use a patching method, but we omit the parameterization and resampling steps. This avoids artifacts due to distortions in the parameter mapping and allows for more flexibility in computing the patch layout. Instead of a 2D Fourier transform we use a decomposition of the surface signal into a linear combination of eigenvectors of an approximate Laplacian operator [33].

For their 3D mesh watermarking, Ohbuchi et al. [18, 19] build Laplacian matrices by using connectivity information of the meshes. Diagonal elements are set to the valences of the corresponding vertices, and the off-diagonal elements (i, j) and (j, i) are set to -1 in case the vertices i and j are adjacent. The remaining elements are set to zero.

3D point clouds do not contain explicit connectivity information. A substitute connectivity can be constructed by virtually connecting each vertex to its k nearest neighbors (typically $6 \le k \le 12$). In general, a k-nearest neighbor search does not lead to symmetric relationships. We enforce symmetry by setting correspondent matrix elements (i, j) and (j, i) to -1. Thus the valences of the points can finally become higher than k. The Laplacian constructed this way can be used to calculate a frequency representation of the geometry [33]. The decomposition of the signal into a linear combination of the eigenvectors is still valid in our 3D point cloud setting.

Our algorithm only uses the lower part of the spectrum where the embedding is more robust. This allows for an optimized calculation of the eigenvectors, e.g. using an iterative Arnoldi (e.g. see [28, 30, 8]) or Lanczos [7] procedure, which produces the m smallest eigenvalues of the Laplacian matrix for a patch with a points P_i and the corresponding eigenvectors.

After the eigenproblem has been solved, the eigenvectors are normalized and sorted in increasing size of their corresponding eigenvalues.

Now, the point coordinates P_i or additional attributes like color values can be projected onto the m eigenvectors v_j to get the spectral coefficients Z_j $(j = 1...m)$:

$$Z_{jk} = \sum_{i=1}^{a} P_{ik} \cdot v_{ji}, \quad k \in \{x, y, z\}. \tag{1}$$

Thus, for each eigenvector of the decomposition we obtain three spectral coefficients. The same bit will be encoded in each of these three values to improve robustness.

To perform the inverse transform at a later time we need to determine the contribution of the spectral values which have not explicitly been calculated. For this purpose we compute an offset R, which has to be added to the inverse transform of the calculated spectral coefficients to get the original coordinates $(i = 1...a)$:

$$R_{ik} = P_{ik} - \sum_{j=1}^{m} Z_{jk} \cdot v_{ji}, \quad k \in \{x, y, z\}. \tag{2}$$

3.3. Watermark Encoding

The watermark can now be set up and encoded. We use a bit string derived from a 160-bit SHA-1 hash value [15], which is dependent on a user-defined key K and specific features of the original object. Analogously to [27], we use the eigenvectors v_1, v_2 and v_3 of the principal component analysis and their corresponding eigenvalues e_1, e_2 and e_3 as features. Therefore, the hash value H is defined as

$$f_e = e_1 \| e_2 \| e_3, \tag{3}$$

$$f_v = v_{1x} \| v_{1y} \| v_{1z} \| v_{2x} \| v_{2y} \| v_{2z} \| v_{3x} \| v_{3y} \| v_{3z}, \tag{4}$$

$$H = \text{SHA-1}(K \| f_e \| f_v), \tag{5}$$

where the ||-operator represents the string concatenation.

The watermarking procedure allows the user to define the number w of bits to embed, so that the watermark W can be shorter than 160 bits:

$$W_k = H_k, \text{ for } k = 1 \ldots w. \tag{6}$$

This w-bit watermark W is now embedded into the low frequency part of the spectrum of every single patch. To increase the robustness against random noise, each bit of W is embedded multiple times into the spectrum. When decoding, a majority vote determines the most probable value to restore. The chip-rate c determines how many times a single bit is embedded and influences the capacity of a patch: $w = a/(uc)$ independent bits can be encoded into the $m = a/u$ lowest frequencies. So before embedding, the watermark's length is increased, transforming the w-bit watermark W into an m-bit watermark M:

$$M_j = W_{[(j-1) \bmod w]+1}, \text{ for } 1 \le j \le m. \tag{7}$$

For security reasons, the embedding procedure uses a randomization process, where $p_j \in \{-1, 1\}$ is the j-th value of a pseudo random sequence uniquely defined by the hash value H and therefore indirectly determined by the secret key K. We use the following randomization:

$$r_j = \text{PRNG}_j(\text{CRC-32}(H)). \tag{8}$$

The CRC-32 function [9] maps the 160-bit SHA-1 value to a 32-bit number of type *long*, which can be used as a seed for the pseudo random number generator PRNG (e.g. the RAN3 function [35]). Resulting numbers smaller than 0.5 are mapped to -1, whereas larger values are set to 1:

$$p_j = \begin{cases} -1, & \text{if } r_j < 0.5. \\ 1, & \text{else.} \end{cases} \tag{9}$$

The encoding and modulation of the m frequencies can now proceed as follows ($j = 0 \ldots m$):

$$Z_{jk}' = \begin{cases} Z_{jk} + p_j \cdot \alpha \cdot l, & \text{if } M_j = 1, \\ Z_{jk} - p_j \cdot \alpha \cdot l, & \text{else.} \end{cases} \qquad k \in \{x, y, z\}. \tag{10}$$

Where l is the length of the diagonal of the object bounding box. The positive number α regulates the amplitude of the modulation: Larger values lead to increased robustness at the cost of degradation in visual quality. Useful values for the strength of embedding α are in the range from 0.001 to 0.01.

3.4. Inverse Transform and Data Storage

To obtain the final watermarked model, we transform the modified spectral coefficients back into the spatial domain ($i = 1 \ldots a$):

$$P_{ik}' = R_{ik} + \sum_{j=1}^{m} Z_{jk}' \cdot v_{ji}, \quad k \in \{x, y, z\}. \tag{11}$$

For later extraction of the watermark from a test object, we need to store:

- The original object O.
- The secret key K.
- The parameters of the watermark W.

4. Watermark Extraction

The goal of watermark extraction is to decide, whether a test object O^+ contains a certain watermark in order to prove the ownership. Prerequisite for a correct watermark extraction is a registration of the test object with the original object.

4.1. Registration

In order to be able to extract the watermark successfully, the original object and the potentially marked object have to be aligned by a registration process.

Simple similarity transformations, i.e. rotations, translations and scaling, can be treated automatically using principal component analysis. However, one has to be careful when the object contains symmetries, as the alignment might not be unique. In this case, or when the object has been cropped or deformed substantially, we use an interactive affine registration method.

The user defines a set of corresponding points $\boldsymbol{p}_i^+ = (x_i^+, y_i^+, z_i^+)^T$ and $\boldsymbol{p}_i = (x_i, y_i, z_i)^T$ on the test object O^+ and the original object O with the help of 3D markers (cf. Figure 3).

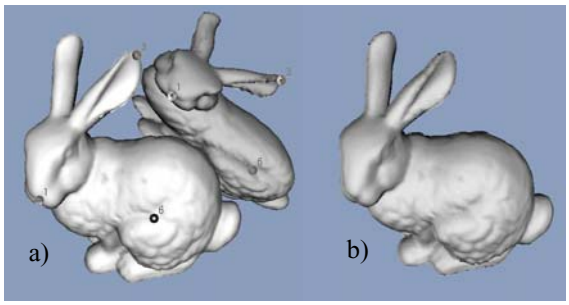

Figure 3: a) For the affine registration a total of 7 pairs of 3D markers is defined by the user. From the chosen camera position not all the markers are visible. b) The test object is transformed by the affine registration in such a way that it is aligned with the original object.

The goal of the registration is to find a transformation matrix A, such that

$$\begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix} = \boldsymbol{A} \cdot \begin{bmatrix} x_i^+ \\ y_i^+ \\ z_i^+ \\ 1 \end{bmatrix}, \text{ where } \boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (12)$$

The transformation of the homogeneous coordinates of a marked point on the test object with the matrix A should result in the homogeneous coordinates of the corresponding point on the original object.

When defining k pairs of points ($k \geq 4$), we get a system of equations for the elements of every row of A. For the unknown elements a_{11}, a_{12}, a_{13} and a_{14}, we get the following system:

$$x_i = x_i^+ \cdot a_{11} + y_i^+ \cdot a_{12} + z_i^+ \cdot a_{13} + a_{14}, \text{ for } i = 1 \ldots k. \quad (13)$$

For $k > 4$ the system is overdetermined, and we calculate a_{11}, a_{12}, a_{13} and a_{14} by using the least squares method, minimizing the error

$$E = \sum_{i=1}^{k} (x_i - (x_i^+ \cdot a_{11} + y_i^+ \cdot a_{12} + z_i^+ \cdot a_{13} + a_{14}))^2. \quad (14)$$

The remaining elements of the matrix can be determined analogously.

By transforming all the points of the test object, we get a reoriented verification object, which is aligned with the original object.

4.2. Resampling, Projection

When extracting a watermark, we need to establish a one-to-one correspondence between samples in the given point cloud and the original data set, which is patched in the same way as during embedding. For this purpose we use a resampling operator based on the moving least squares (MLS) approximation presented by Levin [14]. This method defines a surface from a collection of point samples as the invariant set of a projection operator. A local polynomial approximation of the surface is computed using weighted least squares fits and points are projected onto the polynomial. If a point projects onto itself, it belongs to the surface. The conceptual steps are illustrated in Figure 4. For more details we refer to [21, 1, 23].

The resampling procedure leads to robustness against resampling attacks and permits the application of the same Laplacian that is used for the embedding.

4.3. Frequency Analysis

Watermark extraction relies on the same frequency analysis that is used for embedding. Given a patch of the original object O, the point coordinates of the corresponding patch of the test object O^+ are projected onto the eigenvectors of the Laplacian computed on the original patch. This yields in addition to the original spectral coefficients Z_j a set of corresponding coefficients Z_j^+ for the test patch. Note that no Laplacian is computed on the test object.

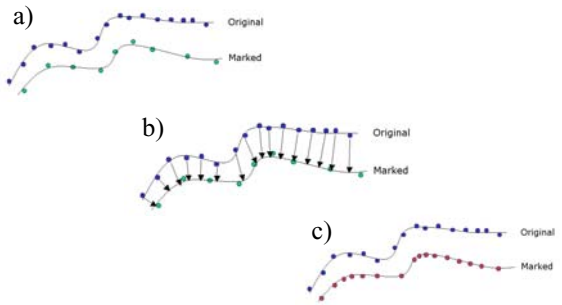

Figure 4: a) A cut through the original object, the marked object and their MLS surfaces. b) The points of the original object are projected onto the MLS surface of the marked object. c) The following steps in the watermark extraction process use the newly created points of the marked object.

4.4. Watermark Extraction

The extraction proceeds in multiple steps by summing differences of test and original spectral values. A counter S_i is updated for each of the w bits of the watermark W. The counters are all initially set to zero ($i = 1 \ldots w$): $S_i = 0$.
For all the m spectral values, the differences are now calculated and added to the counter of the corresponding watermark bit, where p_j is the j-th number of the sequence of pseudo random numbers with the same initialization value as the one used in the embedding process ($j = 1 \ldots m$):

$$s_j = (Z_{jx}^+ - Z_{jx}) + (Z_{jy}^+ - Z_{jy}) + (Z_{jz}^+ - Z_{jz}), \quad (15)$$

$$S_{[(j-1) \bmod w]+1} = S_{[(j-1) \bmod w]+1} + p_j \cdot s_j. \quad (16)$$

In the ideal case, i.e. when the change of the point coordinates (e.g. due to random noise) is small, the sums should have the following values ($i = 1 \ldots w$):

$$S_i = -(-1)^{W_i} \cdot 3 \cdot c \cdot \alpha \cdot l. \quad (17)$$

Because the embedding strength α, the chip-rate c and the object extent l are always positive, it is sufficient to consider the sign of the counter S_i to extract the embedded watermark bit. The bits W_i^+ of the extracted watermark W^+ of a patch can be determined as follows ($i = 1 \ldots w$):

$$W_i^+ = \begin{cases} 1, & \text{if } S_i > 0, \\ 0, & \text{else.} \end{cases} \quad (18)$$

This extraction is performed for each patch.

4.5. Correlation

In general, the watermarks W^+ extracted from the patches do not exactly match the watermark W which has been embedded. In order to estimate whether the watermarks have originally been embedded as the watermark W, we perform a bitwise comparison. For every test patch and its extracted watermark W^+ we calculate the correlation with the original watermark:

$$C = \frac{1}{w} \cdot \sum_{i=1}^{w} (W_i^+ = W_i). \quad (19)$$

We then choose the maximum correlation value of all the patches, because this allows the identification of the copyright owner even when there is only one single patch left which has not been manipulated sufficiently by an attacker. Given p patches with their correlation values C_k, the maximum correlation is

$$\hat{C} = \max(C_1, \ldots, C_p). \quad (20)$$

In case this value exceeds a certain threshold (e.g. 0.75), we assume that the test object contains the watermark W.

5. Implementation

Our watermarking system has been developed in C++ as a plug-in for the Pointshop3D system [39] and includes the watermarking component, a separate affine registration tool and various sample attack plug-ins (cf. Figure 5).

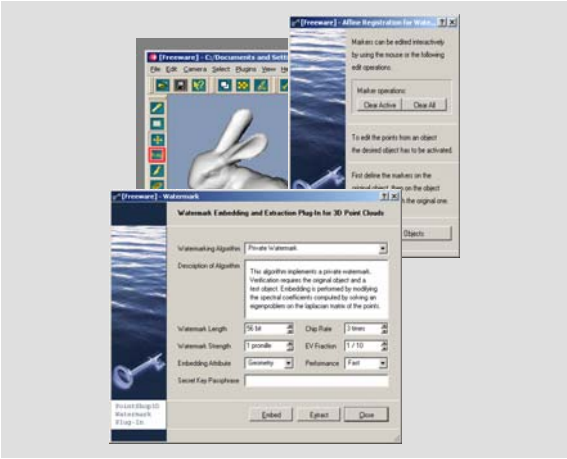

Figure 5: Pointshop3D, the watermarking component and the affine registration tool.

Our frequency transformation uses two different implementations of eigenproblem solvers based on the Arnoldi method. The faster version uses the Meschach library [31] and does not exactly return the smallest eigenvalues, but rather eigenvalues in the lower half of the spectrum, with an increasing concentration of eigenvalues towards the lower end of the spectrum. The more accurate version is based on ARPACK [8] and returns the m smallest eigenvalues, but requires a longer computation time.

6. Results

This section shows some results achieved by our implementation of the presented watermarking procedure. For the frequency transformation, the faster version of the Arnoldi method based on the Meschach library [31] is used. Besides analyzing the robustness against unintentional modifications and deliberate attacks, we present timing measurements for the watermark embedding.

6.1. Test Settings

To test the watermarking procedure, signatures are embedded into three different models. Every model has its own peculiarities and properties (see Figure 6).

- The chameleon consists of 71'180 points and has a fairly smooth surface with fine extremities.
- The bunny consists of 34'834 points and has a structured surface without fine extremities.
- The dragon head consists of 51'002 points and has a structured surface with fine extremities.

6.2. Basic Requirements

Watermarks have to satisfy basic requirements to be useful. On the one hand, they should impact the visual quality of the geometries as little as possible, and on the other hand, watermarks should always be extractable if no modifications have been performed on the signed objects.

The *completeness* property requires that an embedded watermark can be shown to be contained in a signed object. The correlation value has to be higher than a predefined threshold (0.75 in our case).

In addition, the probability that an impostor with an invalid watermark can "prove" that his watermark is contained in the geometry should be negligible. To satisfy this *soundness* requirement, the correlation value of a watermark not contained in a signed or unsigned object should be around 0.5 and not exceed the threshold value. Figure 9 presents graphs containing correlation values of multiple patches. On the x-axis all the patches of the object are contained. The y-axis represents the correlation and ranges from 0 at the bottom to 1 at the top, with the blue line indicating the threshold value at 0.75.

Figure 7 shows the impact of the watermark embedding on the visual appearance of the model for different modulation parameters α. The 56-bit watermark is embedded with chip-rate 3 and frequency usage 1/10. Higher values of α improve the robustness of the watermark, but eventually lead to visible distortions of the model geometry.

However, as the examples in the next section illustrate, sufficiently high robustness can be achieved even for low values of α, i.e. without causing any perceivable artifacts. This also holds at the patch boundaries as illustrated in Figure 8, where a watermark with strength 0.005 has been embedded.

6.3. Robustness Against Attacks

This section presents the results of watermark extractions after attacks have been performed on the objects.

Repetitive embedding of watermarks leaves the original watermark mostly intact as can be seen in Figure 10, which shows the correlation of the first watermark after one, five and ten additional watermarks. The 16-bit watermarks are embedded with chip-rate 3, frequency usage 1/20 and strength 0.001. Due to this property, our scheme is secure against attacks trying to overwrite a signature with another watermark. Moreover, our approach is suited for hierarchical marking of geometries (e.g. along a distribution channel).

The effect of *additive random noise* on the correlation values is shown in Figure 11. The noise is added with intensity 0.01. This intensity is multiplied with the maximum extent of the geometry bounding box to get the maximum coordinate displacement of the points. The 16-bit watermark is embedded with chip-rate 3, frequency usage 1/60, and strength 0.005.

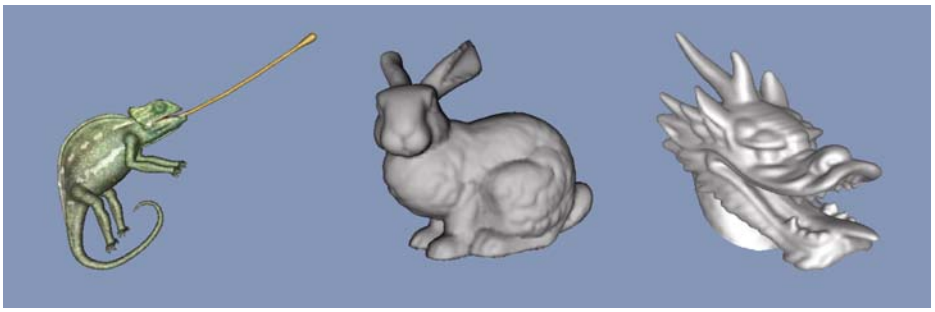

Figure 6: The three test models: the chameleon, the bunny and the dragon head.

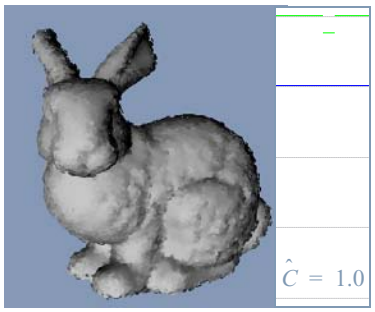

Figure 11: Random noise.

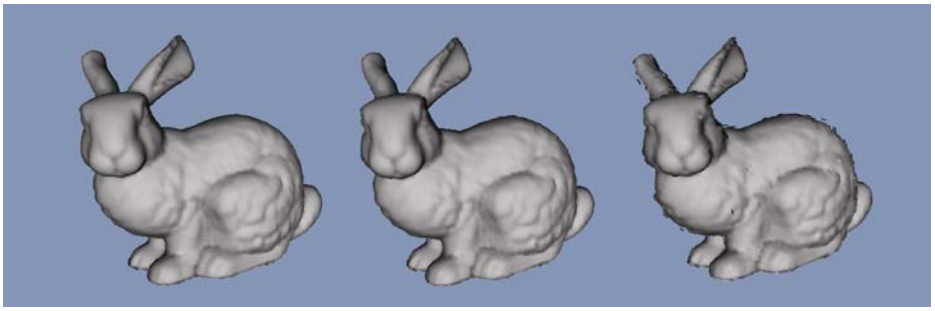

Figure 7: A 56-bit watermark embedded with different modulations (0.001, 0.005, and 0.01).

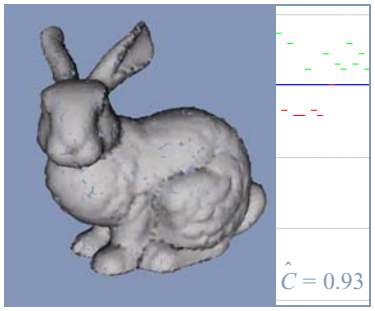

Figure 12: Point reduction (50%).

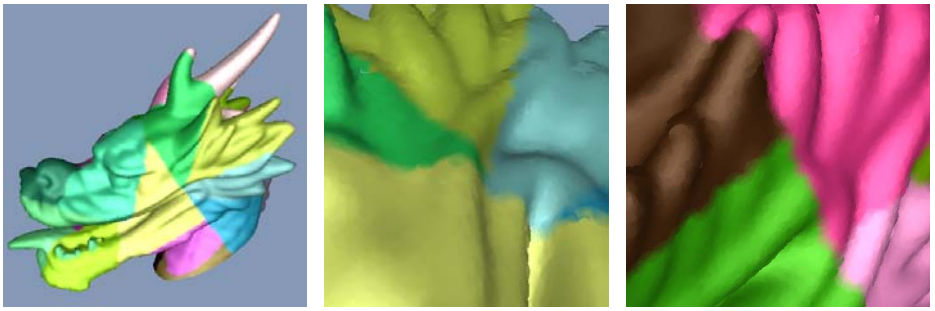

Figure 8: Patching of a watermarked dragon head: no artifacts at patch boundaries are visible.

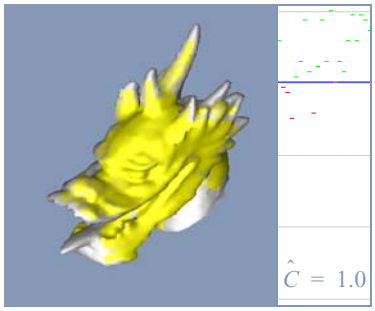

Figure 13: Resampling.

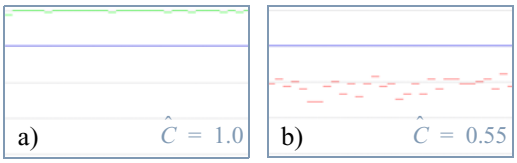

Figure 9: a) Completeness. b) Soundness.

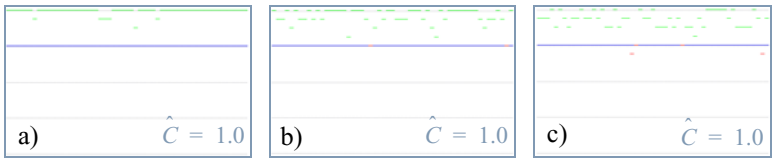

Figure 10: Correlation after n additional marks: a) $n = 1$. b) $n = 5$. c) $n = 10$.

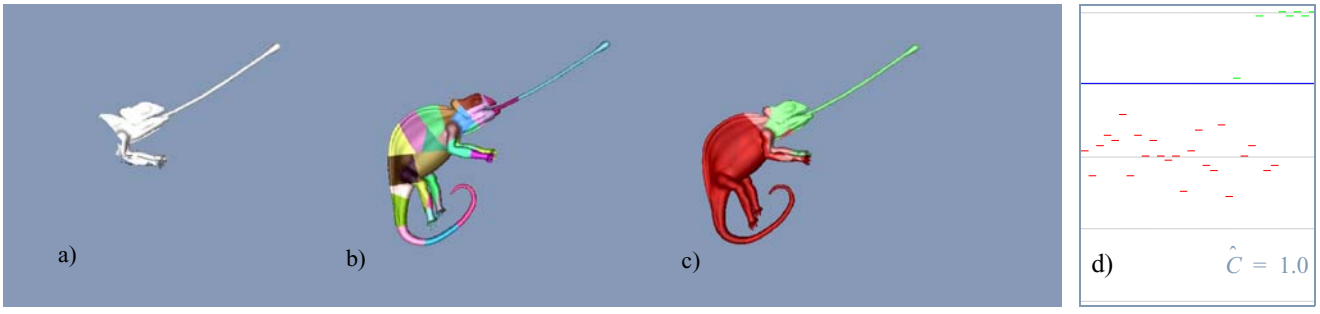

Figure 14: a) Cropped marked model. b) Patching structure. c) Success of extraction of each patch. d) Corresponding correlation values.

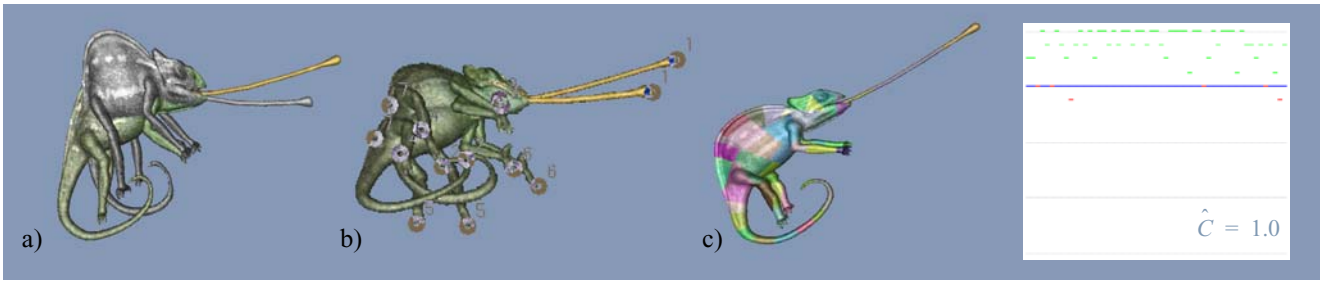

Figure 15: Affine registration. a) The original object and the marked, affinely deformed test object (gray). b) Corresponding points on both geometries are defined. c) After registration, the original object (here with patching) and the test object overlap.

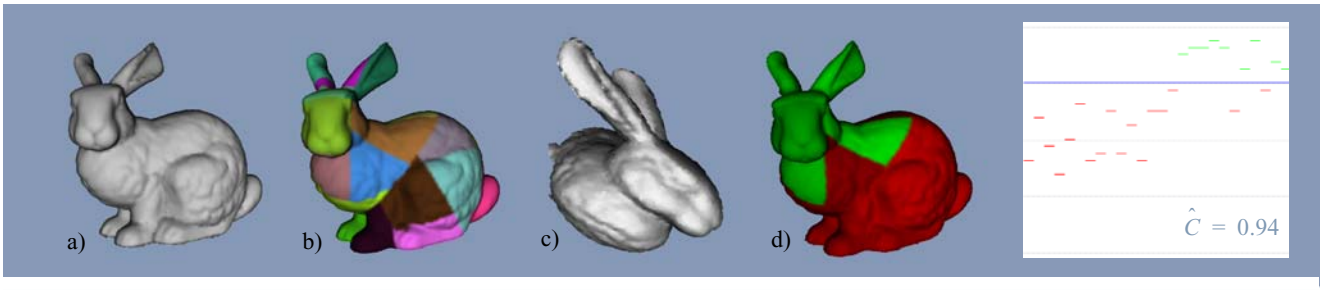

Figure 16: Combined attack. a) Original object. b) Patching. c) Marked bunny test object. d) Extraction success of the patches.

The resulting geometries and correlations after *object simplifications* and *resampling* are shown in Figure 12 and Figure 13. In Figure 12, the bunny is reduced to 50% of the points. The 56-bit watermark is embedded with chip-rate 3, frequency usage 1/10 and strength 0.004. In Figure 13, the surface in the yellow area of the dragon is resampled using the brush tool in Pointshop3D [39]. The watermark is encoded with the same properties as in Figure 12.

By partitioning the object, watermarks can be embedded in a robust manner against *cropping* of the geometry (cf. Figure 14). The 56-bit watermark is embedded with chip-rate 3 and frequency usage 1/10. Note that no patch is eliminated before extraction, which leads to values for patches that cannot be projected onto the test object in a meaningful way.

Similarity transformations like rotations, translations and uniform scaling are entirely undone by the registration process based on the principal component analysis (PCA).

Affine transformations are handled by our user-guided registration process as shown in Figure 15. A 16-bit watermark with chip-rate 3, frequency usage 1/20 and strength 0.001 is extracted.

Combined attacks can be handled as well. Figure 16 shows the result of a reduction to 75% of the points, followed by a cropping attack, random noise with intensity 0.005 and an affine transformation. The 32-bit watermark is embedded with chip-rate 3, frequency usage 1/10 and strength 0.005. Note that the extracted maximum correlation value is well above the threshold.

We also tested robustness against *smoothing*. As expected, our approach achieves good results because watermarks are embedded into the lower part of the spectrum.

6.4. Timing Measurements

By far the most time-consuming step in the whole watermarking pipeline is the frequency transformation that computes the low-frequency eigenvectors of the Laplacian.

As discussed above we have implemented two variants of this algorithm, a fast method giving approximate results and a less efficient algorithm that computes the exact lower fraction of the eigenvectors. Figures 17 and 18 compare the running times for both methods, measured on a Pentium III 1 GHz computer running Windows 2000 with 512 MB RAM.

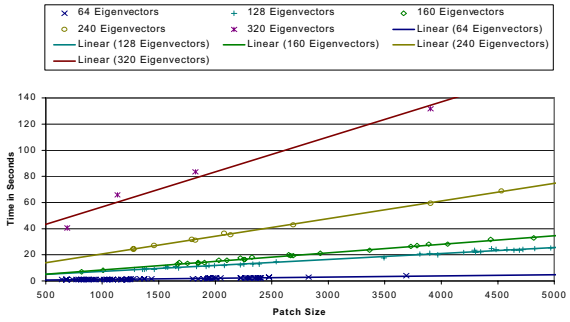

Figure 17: Time for the frequency transformation depending on the patch size *a* and the number *m* of eigenvectors to determine (fast implementation).

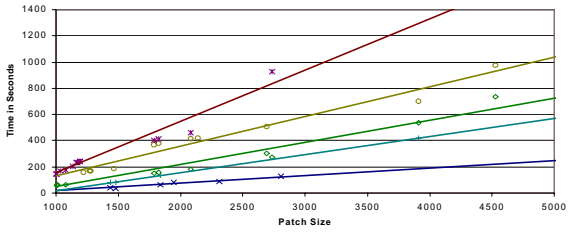

Figure 18: Time for the frequency transformation depending on the patch size *a* and the number *m* of eigenvectors to determine (accurate implementation).

6.5. Discussion

The results clearly show that the presented watermarking procedure can tag objects with watermarks in a robust way.

- Even when embedding a second watermark, which significantly alters the visual appearance of an object, the original watermark can still be extracted. This robustness against the embedding of further watermarks results in a watermarking procedure which is suited for *hierarchical marking*.
- *Similarity transformations* are effectively undone by the automatic reorientation of the object based on a principal component analysis. *Affine transformations* are handled by our user-guided affine registration.
- *Random noise* and *smoothing* do not affect the extraction of the watermark up to a certain extent. Additionally, the impacts of *object simplifications* and *resampling* are effectively minimized by the MLS projection step.
- The presented patching scheme achieves robustness against *cropping* of the object and allows for successful watermark extraction as long as one patch still contains the encoding.

Using the more accurate, slower calculation of the eigenvectors of the Laplacian matrix even better results can be achieved. The watermarks also stay invisible at increased strengths and the robustness is improved, because the bits are embedded exactly into the m lowest frequencies (cf. Figure 19).

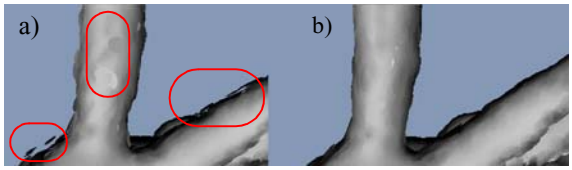

Figure 19: a) Watermarks embedded with the faster version of the frequency transformation and a strength of 0.015 result in visible modifications of the marked object (see areas marked with red). b) When using the more accurate version of the frequency transformation no significant deformations are visible, thus allowing stronger and more robust watermarks to be used.

Although the test cases use shorter watermarks, bit strings of up to 160 bit can be embedded in a robust manner as well. If necessary, longer watermarks can be split by distributing the watermark bits over multiple patches.

Therefore, our watermarking scheme satisfies all the requirements with respect to robustness and credibility of a watermark proof.

One slight disadvantage of the system is that the affine registration cannot be automated and requires user intervention. Also, as described in Section 6.4, running times are fairly high due to the costly computation of the eigenvector decomposition. However, this computation only needs to be done at the embedding stage. If performance is a critical issue, the computed eigenvectors can be stored and reused during watermark extraction.

7. Conclusion and Future Work

In this paper we have presented a robust private watermarking scheme for unstructured 3D point clouds. Watermarks are embedded effectively and with minimal visible impact into the geometry or appearance attributes by using a spectral analysis based on previous work for 3D meshes by Ohbuchi [18, 19]. In contrast to systems only allowing watermarks to be embedded into position values, our plug-in developed for Pointshop3D [39] also enables the embedding in color values of the objects. Our method fulfills all requirements and criteria for an effective watermarking procedure. It is the first system that allows to tag point sampled geometries for protection of ownership rights.

Even though our approach already delivers good results, some further improvements are conceivable. Instead of the relatively simple, automatic PCA registration the system could use a better registration procedure like an iterated closest point approach [4, 5, 37]. Furthermore, the error correction based on majority voting used to extract the watermark could be replaced by a more efficient approach like a syndrome decoder [27]. Additionally, the eigenvalue problem of the frequency transformation could potentially be sped up by replacing the Arnoldi method [28, 30, 8] by an optimized implementation of the Lanczos procedure for symmetric matrices [7]. Future research has to further explore and analyze the relationship between watermark length, chip-rate, patch size, frequency band used, and the resulting robustness. Finally, thanks to its flexibility and its capabilities, the presented system offers an ideal platform for the development of a future steganography system for 3D point geometries, thus allowing to embed secret information invisibly into point clouds.

8. Acknowledgments

The bunny and the dragon head model are courtesy of the Stanford University Computer Graphics Laboratory.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. “Point set surfaces.” In IEEE Visualization 2001, pages 21–28, October 2001.

[2] O. Benedens. “Geometry-Based Watermarking of 3D Models.” IEEE CG&A, 4:46–55, January/February 1999.

[3] O. Benedens and C. Busch. “Towards Blind Detection of Robust Watermarks in Polygonal Models.” Computer Graphics Forum, 19(3), 2000.

[4] P. Besl and N. McKay. “A method for registration of 3D shapes.” IEEE Trans. on Pattern Analysis and Machine Intelligence, 18(14):239–256, 1992.

[5] Y. Chen and G. Medioni. “Object modelling by registration of multiple range images.” Image and Vision Computing, 10(3):145–155, April 1992.

[6] I. Cox, J. Kilian, T. Leighton, and T. Shamoon. “Secure Spread Spectrum Watermarking for Multimedia.” IEEE Transactions on Image Processing, 6(12):1673–1687, 1997.

[7] J. K. Cullum and R. A. Willoughby. Lanczos Algorithms for Large Symmetric Eigenvalue Computations, volume 1. Birkhauser, Boston, 1985.

[8] F. M. Gomes and D. C. Sorensen. ARPACK++: A C++ implementation of ARPACK eigenvalue package.

[9] International Organization for Standardization. “ISO Information Processing Systems - Data Communication High-

Level Data Link Control Procedure - Frame Structure." IS 3309, October 1984.

[10] A. Kalahia and A. Varshney. "Statistical Point Geometry." Symposium on Geometry Processing, pages 107-115, 2003.

[11] S. Kanai, H. Date, and T. Kishinami. "Digital Watermarking for 3D Polygons using Multiresolution Wavelet Decomposition." In Proc. of the Sixth IFIP WG 5.2 International Workshop on Geometric Modeling: Fundamentals and Applications (GEO-6), pages 296–307, Tokyo, Japan, December 1998.

[12] Z. Karni and C. Gotsman. "Spectral Compression of Mesh Geometry." In Siggraph 2000, Computer Graphics Proceedings, pages 279–286. ACM Press / ACM SIGGRAPH / Addison Wesley Longman, 2000.

[13] E. Koch and J. Zhao. "Towards Robust and Hidden Image Copyright Labeling." In Proc. of 1995 IEEE Workshop on Nonlinear Signal and Image Processing, pages 452–455, Halkidiki, Greece, June 1995.

[14] D. Levin. "Mesh-independent Surface Interpolation." Advances in Computational Mathematics, 2001.

[15] National Institute of Standards and Technology. "Secure Hash Standard." FIPS publication 180-1, April 1994.

[16] N. Nikolaidis and I. Pitas. "Copyright protection of images using robust digital signatures." In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '96), volume 4, pages 2168–2171, May 1996.

[17] R. Ohbuchi, H. Masuda, and M. Aono. "Watermarking Three-Dimensional Polygonal Models." In ACM Multimedia 97, Seattle, USA, 1997.

[18] R. Ohbuchi, A. Mukaiyama, and S. Takahashi. "A Frequency-Domain Approach to Watermarking 3D Shapes." In Computer Graphics Forum, Proc. EUROGRAPHICS 2002, volume 1(3), 2002.

[19] R. Ohbuchi, S. Takahashi, T. Miyazawa, and A. Mukaiyama. "Watermarking 3D Polygonal Meshes in the Mesh Spectral Domain." In Proceedings of Graphics Interface 2001, pages 9–18, 2001.

[20] M. Pauly and M. Gross. "Spectral Processing of Point-Sampled Geometry." In SIGGRAPH 2001, Computer Graphics Proceedings, pages 379–386. ACM Press / ACM SIGGRAPH, 2001.

[21] M. Pauly, M. Gross, and L. P. Kobbelt. "Efficient Simplification of Point-Sampled Surfaces." In Proc. IEEE Visualization 02, 2002.

[22] M. Pauly, R. Keiser, L. Kobbelt, and M. Gross. "Shape Modeling with Point-sampled Geometry." In SIGGRAPH 2003, Computer Graphics Proceedings, pp. 641-650, 2003.

[23] M. Pauly, L. Kobbelt, and M. Gross. "Multiresolution Modeling of Point-Sampled Geometry." In CS Technical Report 378, ETH Zürich, September 2002.

[24] I. Pitas. "A Method for Signature Casting on Digital Images." In IEEE International Conference on Image Processing (ICIP'96), volume 3, pages 215–218, Lausanne, Switzerland, September 1996.

[25] A. Piva, M. Barni, F. Bartolini, and V. Cappellini. "DCT-Based Watermark Recovering Without Resorting to the Uncorrupted Original Image." In IEEE Signal Processing Society 1997 International Conference on Image Processing (ICIP'97), Santa Barbara, California, October 1997.

[26] E. Praun, H. Hoppe, and A. Finkelstein. "Robust Mesh Watermarking." In Siggraph 1999, Computer Graphics Proceedings, pages 49–56, Los Angeles, 1999. Addison Wesley Longman.

[27] M. Schüpbach. "Geometrie-Watermarking." Diploma Thesis, ETH Zürich, 1999.

[28] J. A. Scott. "An Arnoldi Code for Computing Selected Eigenvalues of Sparse, Real, Unsymmetric Matrices." ACM Transactions on Mathematical Software, 21(4):432–475, 1995.

[29] E. Shaffer and M. Garland, "Efficient Adaptive Simplification of Massive Meshes." IEEE Visualization 01, 2001.

[30] D. C. Sorensen. "Implicitly Restarted Arnoldi/Lanzcos Methods For Large Scale Eigenvalue Calculations." Technical Report TR-96-40, 1996.

[31] D. E. Stewart and Z. Leyk. Meschach Library, Australian National University, Canberra, Australia.

[32] M. D. Swanson, B. Zu, and A. H. Tewfik. "Robust Data Hiding for Images." In 7th Digital Signal Processing Workshop (DSP 96), pages 37–40, Loen, Norway, 1996.

[33] G. Taubin. "A Signal Processing Approach to Fair Surface Design." Computer Graphics, 29(Annual Conference Series):351–358, 1995.

[34] R. van Schnydel, A. Z. Tirkel, and C. F. Osborne. "A Digital Watermark." In IEEE International Conference on Image Processing (ICIP'94), volume 2, pages 86–90, 1994.

[35] W. H. Press et al. Numeric Recipes in C: The Art of Scientific Computing. Cambridge University Press, 2nd edition, 1992.

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

[37] Z. Zhang. "Iterative point matching for registration of free-form curves and surfaces." International Journal of Computer Vision, 13(2):119–152, 1994.

[38] J. Zhao and E. Koch. "A Digital Watermarking System for Multimedia Copyright Protection." In ACM Multimedia 96, Boston, MA, USA, 1996.

[39] M. Zwicker, M. Pauly, O. Knoll, and M. Gross. "Pointshop 3D: An interactive system for point-based surface editing." In SIGGRAPH 2002, Computer Graphics Proceedings, pp. 322-329, 2002.