Chapter 7

Spectral Watermarking of 3D meshes

7.1 Introduction

In this Chapter, we present a blind and semi-fragile watermarking scheme based on the *spectral decomposition* of 3D meshes. Since the extension of this very common signal processing tool which is the Discrete Fourier Transform (DFT) to 3D meshes is relatively new, we first emphasize on the theoretical properties of this transform. Based on a very classical theorem which states that the basisfunctions of the Fourier transform are the eigenfunctions of the Laplacian operator, Gabriel Taubin [122] has been the first to adapt spectral decomposition to 3D meshes in 1995. Theresince, many improvements have been proposed to use this mesh transform to compression [8, 74, 75], filtering [122], partitioning [76] as well as watermarking [28, 100, 101] purposes.

Towards watermarking purposes, Ohbuchi et al. [101] have introduced a non-blind additive scheme (inspired by spread spectrum techniques) in a copyright protection framework. The watermark detection relies on a registration and resampling step with the original model. This way, this scheme can address connectivity modifications [100]. We propose here a *blind* substitutive watermarking scheme to randomly spread the watermark over the three spectral axis. Our watermarking scheme offers good robustness results against noise addition, smoothing as well as RST transforms. However, it remains weak regarding connectivity attacks such as simplification or remeshing but integrates well into a mesh geometry spec-

tral processing or compression framework, or for a data hiding or augmented content application scenario.

The organization of this Chapter is the following. Section 7.2 presents the theory of the spectral decomposition of 3D meshes. Section 7.3 focuses on methods that enable to reduce the computational cost of the Spectral decomposition. These methods are based on mesh partitioning, on fixed-basis decompostion as well as on the Lapped Orthogonal Transform (LOT) which enables to improve the visual quality of low-frequency representations. Next, Section 7.4 is dedicated to the progressive transmission scheme we propose. Finally, Section 7.5 presents our blind and semi-fragile watermarking scheme.

7.2 Spectral decomposition

In this Section we define how to compute the spectrum of the mesh geometry by using the Laplacian operator. This may be considered as the equivalent of the DFT or Discrete Cosine Transform (DCT) for meshes. The pioneering work of Taubin has been complemented by Ben-Chen et al. [18] and Zhang [142] who have deeply investigated the optimality of the spectral decomposition in the application of 3D mesh compression. We propose here the main results of these works which study the optimality of various discretizations of the Laplacian operator on a 3D mesh.

Here follow some important notations already introduced in Chapter 2. Consider a mesh $M = (A, \boldsymbol{x})$ with n vertices, where A is its adjacency matrix ($A_{ij} = 1$ if points i and j are connected and $A_{ij} = 0$ otherwise) and $\boldsymbol{x}$ is the matrix of geometry coordinate vectors ($\boldsymbol{x}_i$ denotes the 3D (x, y, z) coordinates of point[1] i and $\boldsymbol{x} = [X, Y, Z] \in \mathbb{R}^{n\times 3}$). D is the diagonal matrix of vertex degrees ($D_{ii} = d_i$ and $D_{ij} = 0$ if $i \neq j$). The degree of a vertex is the number of vertices which are connected to it by one edge, i.e. the cardinality of the 1-ring neighborhood.

7.2.1 Discrete Laplacian Operator

In a few words, the most used discretization of the Laplacian operator for 3D meshes is the *discrete uniform Laplacian*, also known as Tutte Lapla-

[1] Like most authors, we use point and vertex as synonyms in our 3D mesh framework.

cian (TL for short and noted T):

$$T = I - D^{-1}A \tag{7.1}$$

where $I \in \mathbb{R}^{n \times n}$ is the identity matrix. This is the operator that Taubin has proposed. As mentioned previously, he has pointed out that the eigenvectors of this operator represent the natural vibration modes of the mesh, while the corresponding eigenvalues capture its natural frequencies. The link with the classical DCT and the DFT are evident. Unfortunately, the eigenvectors of the TL operator possess no analytical form in general and there exist no fast methods, analogous to the Fast Fourier Transform (FFT), to compute the corresponding mesh signal transform. In the particular case of a regular connectivity, it can be shown that the eigenvectors of the TL operator are the same as the basisfunctions of the 2D FFT [75].

In addition to the TL operator, there exist several variants such as the Kirchhoff Laplacian (KL for short and noted K):

$$K = D - A, \tag{7.2}$$

and the normalized Graph Laplacian (GL for short and noted G):

$$G = I - Q, \tag{7.3}$$

where $Q_{ij} = \frac{A_{ij}}{\sqrt{d_i d_j}}$.

The eigen-decomposition of these operators has also been used for mesh filtering. All these operators are *combinatorial*, i.e. they only depend on the connectivity of the mesh. The so-called *geometry-driven Laplacians* account for measures such as edge lengths and triangle angles. Operators of this type include the (edge-length based) scale-dependent Laplacian, or SDL, the mean curvature flow operator, suggested by Desbrun et al. [43] for implicit mesh fairing, as well as operators derived from Floater's shape-preserving weights [50] and mean-value coordinates, designed to generalize Tutte embedding [124] for minimizing parametric distortion. The mean curvature flow operator is generally regarded as the discrete Laplacian-Beltrami operator for triangle meshes.

7.2.2 Discrete Laplacian operators and the Principal Component Analysis

Ben-Chen et al. propose to study the distribution of point positions in the mesh M 9.12. They can prove for the 1D case and the 2D case that for gaussian distributions, the KL matrix K is the pseudo-inverse of the covariance matrix used by Principal Component Analysis (PCA) (a.k.a. Karhunen-Loeve transform). These matrices share the same eigenvectors as well as a zero eigenvalue of multiplicity one. The other non-zero eigenvalues of the TL operator are the inverse of the eigenvalues of the covariance matrix. Since the PCA is optimal for gaussian distributed vectors, they conclude the TL operator is optimal as well. Unfortunately, there is still no extension of valid point distributions for 3D meshes and the study of Ben-Chen et al. only mention that the 3D case should be vey close to the 2D case but they bring no evidence of such statement. The main difficulty is that their model of point distribution needs bounds: in the 1D case, a point cannot be placed outside the positions of its neighbors; in the 2D case, a point cannot be placed outside its 1-ring. For the 3D case, finding such bounds is not evident. If tangential coordinates can be bounded by the 1-ring, there is no such bound for the normal coordinate.

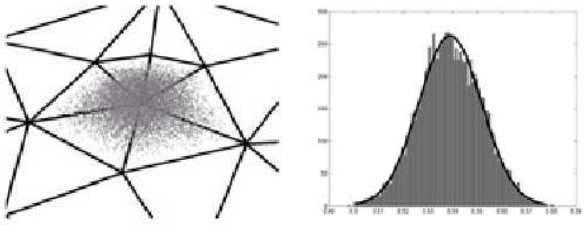


Figure 7.1: *Point distribution example in the 2D case (image courtesy of Ben-Chen et al. [18]). The point position distribution is illustrated on the left. The positions of the point are bounded by its 1-ring and their probability density function is illustrated on the right with respect to the distance to the barycenter of the 1-ring. This distribution is no more valid for the 3D case and the optimality analysis is still not extended from 1D and 2D to the 3D.*

7.2.3 Discrete Laplacian operators as connectivity-preserving linear mesh system

Here, we follow the formalism of Zhang [142] to analyze the properties of the various Laplacian operators with a signal processing approach.

A *connectivity-preserving linear mesh system* can be written as:

$$\boldsymbol{y} = H\boldsymbol{x} \tag{7.4}$$

with input $\boldsymbol{x}$ and output $\boldsymbol{y}$, where $H \in \mathbb{R}^{n\times n}$. In functional form, we can write $\boldsymbol{y}(k) = H[\boldsymbol{x}(k)] = H \cdot \boldsymbol{x}(k)$, where $\boldsymbol{x}(k)$ and $\boldsymbol{y}(k)$ denote the mesh signals indexed by k . Without loss of generality, let us work with the x-coordinates only, i.e., $X \in \mathbb{R}^n$. The limitation of the approach of Zhang is typically because he considers the three coordinates of a point as completely independent. However, the theoretical results presented in this Section have been successfuly validated through practical experiments.

Zhang proposes to view X_i as an amount of signed potential energy, relative to the origin of the coordinate space and along the x-axis at point i of the mesh. The impulse response of the linear system the mesh can be seen as the result of an energy dispersion. When the input mesh is given by the discrete 1D Kronecker delta function at vertex k, i.e., $\delta(k-l) = 1$, $k, l \in 1, 2, ..., n$, if $k = l$ and 0 otherwise, the output at location k is the impulse response of the mesh linear system:

$$h(k;l) = H[\delta(k-l)] = H \cdot \delta(k-l) = H_{k,l}. \tag{7.5}$$

For a fixed l, $h(k;l)$ models the distribution of the unit amount of energy at vertex k over the mesh grid. This impulse response is completely characterized by the matrix H. Some suitable properties of an impulse response and of the linear system it defines are:

Irreducibility: a matrix A is irreducible if its corresponding graph $G(A)$ is connected [142]. It can be shown that if the mesh is connected, the impulse response is irreducible [26].

Nonnegativity: a matrix H is nonnegative if $\forall i, j \in [0, n], H(i, j) > 0$.

Symmetry: symmetric matrices have nice properties such as real eigenvalues, orthogonality of the eigenvectors and reduced cost of diagonalization by Cholesky decomposition, among others.

Constant stable states - unit row sums: Many mesh processing and analysis tasks are iteratively performed. A stable state represents an equilibrium of an iterative system or a fixed point, i.e., X for which $HX = X$. The constant mesh defined as any mesh X with $X_i = X_j$ for all i and j, is a stable state. A system $Y = HX$ has a constant stable state if and only if the sum of each row of the impulse response H is 1 [142].

Energy conservation - unit column sums: the implication of this property is that the linear system preserves the centroid, or DC value in signal processing terms, of the mesh signal, if and only if each column of H adds up to 1.

Stochasticity and double stochasticity: a real matrix is said to be row-(column)-stochastic if it is nonnegative and has a constant row (column) sum of 1. Double stochasticity requires both row- and column- stochasticity. The spectral radius (the magnitude of the largest eigenvalue) of row-stochastic matrices is equal to 1.

Variance diminishing property: the variance of a mesh X with n vertices is given by $\sigma^2(X) = [\sum_{i=0}^{n}(X_i - \bar{X})^2]/n$, where $\bar{X}$ is the mean of $[X_1, ..., X_n]$. The LMSP system $Y = HX$ has the variance diminishing property if repeated application of H to X cannot increase the variance of X. Using Birkhoff's characterization of doubly stochastic matrices, it can be shown that doubly stochastic matrices do have the variance diminishing property [26]. This however does not hold in general for matrices that are only row- or column-stochastic [26].

Smoothing matrix: if a matrix H has constant unit row sum and if $\lambda = |1|$ is an eigenvalue of H with multiplicity one and if the magnitude of all other eigenvalues of H is inferior to 1, then H is a smoothing matrix or smoothing operator. This implies that applying several times H to a vector smoothes it, i.e. the variations of this vector tend to zero (i.e. to a constant vector). In that sense, the Tutte Laplacian is a smoothing matrix and has indeed been used for low-pass filtering of mesh geometry by Taubin [122].

7.2.4 Properties of discrete combinatorial mesh Laplacian operators

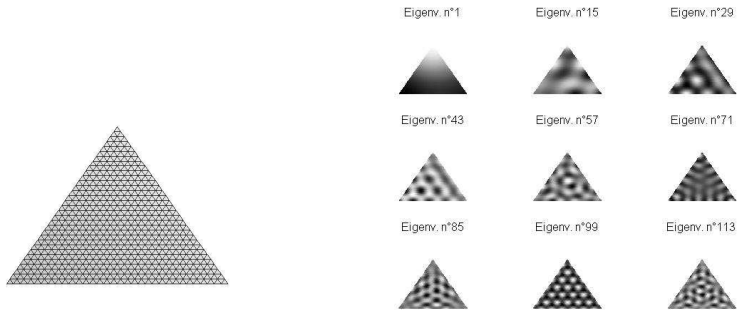
The Kirchhoff operator (KL): we recall that $K_{ij} = d_i$, if $i = j$, $K_{ij} = -1$ if $i = j$ and (i, j) is an edge, and $K_{ij} = 0$ otherwise. Clearly, the KL operator is symmetric and it has constant zero row sum. By the Gerschgorin's Theorem [115], the eigenvalues of K are within $[0, 2d_{max}]$, where d_{max} is the maximum vertex degree. Also, it is well-known [76] that the number of zero eigenvalues of K is precisely the number of components in the mesh graph. Thus for a connected mesh, the smallest eigenvalue of the KL operator is 0 and it is unique. It follows that the KL operator is indeed a smoothing operator.

The normalized graph Laplacian (GL): G is symmetric. It is also known [142] that the smallest eigenvalue of G is 0 and has multiplicity one. However, the GL operator is not a smoothing operator since it does not have constant zero row sum. In general, the eigenvectors of G corresponding to the zero eigenvalue are not constant. Applying this operator does not produce a smoother geometry. Even though the GL operator has proven to be quite useful in analyzing topological properties of graphs, it is unsuitable to use for mesh smoothing or spectral mesh compression.

The Tutte Laplacian (TL): T can be rewritten as $T = RK = I - RA = I - C$, where R is equal to D^{-1} and C is the centroid matrix : $C_{ij} = \frac{1}{d_i}$ if and only if (i, j) is an edge. Although C has the same zero-nonzero structure as the adjacency matrix A, it is not symmetric in general, and it is neither doubly stochastic, as such the variance diminishing property does not always hold [115]. It can be shown however that the eigenvalues of T are all real and lie in the interval $[0, 2]$. Also, T has constant zero row sum. It turns out that the TL and GL operators are similar and share thus the same set of eigenvalues. To see this, note that since $C = R \cdot A$, $R^{-1/2} \cdot C \cdot R^{1/2} = R^{1/2} \cdot A \cdot R^{1/2}$. Note that $Q = R^{1/2} \cdot A \cdot R^{1/2} = I - G$, therefore:
$R^{-1/2} \cdot T \cdot R^{1/2} = R^{-1/2} \cdot (I - C) \cdot R^{1/2} = I - R^{-1/2} \cdot C \cdot R^{1/2} = I - Q = G$.
It follows that the zero eigenvalue of T also has multiplicity one, and T is a smoothing operator. In general however, the eigenvectors of T are not orthogonal, since T is not symmetric.

Second-order symmetric Tutte Laplacian (SSTL): Zhang has proposed a new symmetric version of the Tutte Laplacian operator, SSTL, given by $J = T^T T$, where T is the TL operator. As a direct consequence of the zero row sum property of T, the SSTL also has constant zero row sum. It is positive semi-definite: for any vector v, $v^T J v = v^T T^T T v = (Tv)^T (Tv) = \|Tv\|^2 \geq 0$. It can be shown that the SSTL is a smoothing matrix, simply by prooving that the zero eigenvalue of J has multiplicity one. Let e be an eigenvector of J corresponding to the zero eigenvalue. Then $T^T T e = 0$, but $e = 0$. It follows that $e^T T^T T e = 0$, so $\|Te\|^2 = 0$. Therefore, $Te = 0$ and e is an eigenvector of T corresponding to 0, so it has to be the constant one, as long as the mesh is connected. In this case, the multiplicity of the zero eigenvalue of J is one, and J is a smoothing matrix. Unlike the TL and KL operators, the SSTL operator extends weights to the second-order neighbors of a vertex. One may view it as a symmetric approximation of the second order TLT^2. It has been shown [43] that second-order fairing operators tend to achieve a good balance between smoothing and having less shape distortion.

In order to emphasize on the links between the eigendecomposition of these operators and the well-known DFT or DCT, we have represented the some eigenmodes for the TL, KL and GL operators for a regular connectivity triangle in Fig. 7.2 and Fig. 7.4. Furthermore, the first non-constant eigenvector of the KL operator, known as the Fiedler vector, has interesting properties. Following Taubin, this vector represents the first (non DC) natural vibration mode of the mesh and it enables to bisect the mesh in two well-balanced parts [76]. Furthermore, by coloring the isovalues of this vector on a 3D shape, we can see a propagating front from one extremity of the shape to another one. This vector as well as other modes are illustrated on figure 7.4. The fact that the connectivity, without knowledge of the geometry, can give information on the shape of the 3D model has also been investigated by Isenburg et al. [63]. They have shown that, given a spherical parameterization (see Chapter 2), if all triangles given by the connectivity are iteratively embedded in the 3D space with the same area, the resulting shape is similar to the true shape of the mesh descrbed by the geometry (see Fig. 7.5). Of course, the results are far better if the shape is sampled with regular connectivity and a constant point sampling density. But it may give an insight on why combinatorial Laplacians capture the shape of 3D models.



(a) **(b)**

Figure 7.2: *Vibration modes for several Laplacian operators. The values of the laplacian eigenvectors are mapped on corresponding points in the mesh (color code from black to white). (a) a regular connectivity mesh with* 500 *points, (b) the eigenmodes of this mesh for the KL operator.*

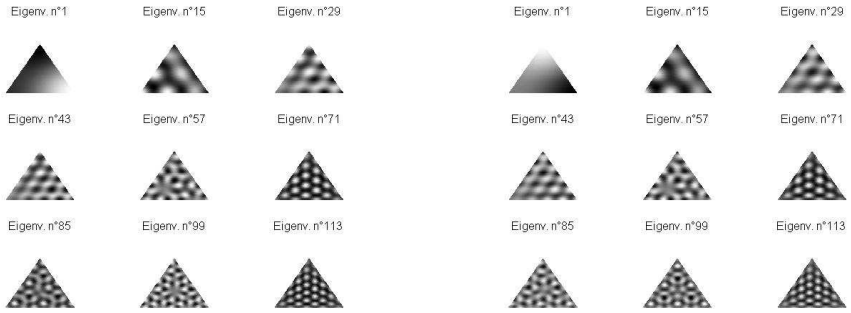
7.2.5 Mesh spectra

As seen before, the eigenvectors of a laplacian operator form an orthogonal basis of $\mathbb{R}^n$. The eigenvalues of K (resp. T), λ_i , $0 \leq i \leq n-1$, are bounded by 0 and $2d_{max}$ (resp. by 0 and 2), where d_{max} is the maximum vertex degree. The transformed vectors of the geometry $\boldsymbol{x} = X, Y, Z$ are obtained by separately projecting the X, Y, Z vectors over the basis functions. The eigenvalues are ordered by decreasing magnitude. We thus obtain the following projection system:

$$\Lambda = diag(\lambda_0, ..., \lambda_{n-1}) = B^{-1}LB, \tag{7.6}$$

where L stands for a laplacian operator. In the sequel, we focus on KL and TL operators. The columns of B are the basis functions (eigenvectors of L) and the columns of B^{-1} are the dual basis functions. If L is the KL operator, $L = K$ is symmetric, $B^{-1} = B^T$ and dual basis functions are equal to the basis functions (this is not true for the TL operator). As a result of the transform applied to the geometry, we obtain three vectors of dimension n called spectra or pseudo-frequencies (P, Q, R):

$$\begin{cases} P = BX \\ Q = BY \\ R = BZ \end{cases} \tag{7.7}$$



(a) **(b)**

Figure 7.3: *Vibration modes for several Laplacian operators. The values of the laplacian eigenvectors are mapped on corresponding points in the mesh (color code from black to white). (a) the eigenmodes for the TL operator (b) the eigenmodes for the GL operator.*

An exact reconstruction is performed by:

$$\begin{cases} X = B^{-1}P = B^T P \\ Y = B^{-1}Q = B^T Q \\ Z = B^{-1}R = B^T R \end{cases} \tag{7.8}$$

The sum of the powers of the signal over the three pseudo-frequency axis defines the power spectrum of the Laplacian of M (coefficients sorted with respect to eigenvalues):

$$S_i = \|P_i\|^2 + \|Q_i\|^2 + \|R_i\|^2 \ , \ 0 \leq i \leq n-1 \tag{7.9}$$

This spectrum is illustrated in Fig. 7.6 for two patches extracted from the models *Bunny* and *Fandisk*.

7.3 Fixed basis decomposition, partitioning and overlapping

For large meshes (a typical limit today is about 500 vertices), the computation of the eigenvalues can become unstable, and very time-consuming. This computation is generally done in $O(n^3)$, but it may be reduced to

Figure 7.4: *Vibration modes on 3D meshed models. On the left, the Fiedler vector isovalues have been color coded. Isenburg et al. propose to use it to stream meshes from an extremity to another (image courtesy of Isenburg et al. [64]). On the right, the first non constant vibration modes have been represented with colors. These are the four first non-constant eigenvectors of the KL operator for two different models. It can be seen that these similar shapes also have similar eigenvectors.*

$O(n)$ in the case of sparse matrices. For these reasons, meshes have to be partitioned into several sub-meshes (patches). It is the case in Fig. 7.6 where the only patches in blue are considered to compute the spectra. They are limited to 500 points. But, since the eigenvalues depend on the mesh connectivity, the matrix diagonalization must be performed for every patch. In order to improve the efficiency of the method, it may be useful to perform the decomposition on a fixed basis for all the patches. We use the same idea as in an earlier work [75]: a Tutte projection [124] is performed to map the original geometry on a fixed basis for every patch. In the following, we first describe Tutte projection and vertex mapping, which are the two stages for translating from arbitrary to fixed connectivity basis functions. Finally we complement this approach with the presentation of our new patch augmentation scheme by spatial overlapping. Using a fixed basis, we divided by 3 the overall processing time. This comes from avoiding diagonalizing too many 500×500 matrices. In the sequel, we refer to the eigenvectors of the combinatorial Laplacian operator based on the original connectivity as *optimal basis functions*.

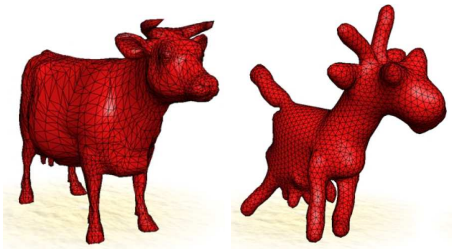


Figure 7.5: *Connectivity shapes (data courtsey of Isenburg et al. [63]). On the left, the cow model represented with combinatorial and geometric information. On the right, the same model represented only with the connectivity. The two shapes are interestingly similar.*

7.3.1 Tutte projection

Changing the connectivity of a mesh into a fixed one is generally a difficult problem. A Tutte projection allows to assign 2D coordinates to vertices with respect to their connectivity. This is one of the most simple parameterizations (see Chapter 2). The Tutte projection is an iterative process: the vertices of the patch border are fixed on a circle, and 2D coordinates of the inner vertices are updated until every vertex is at a fixed distance from the center of mass of its neighborhood. This process converges to an equilibrium: the vertices are mapped on the disk while respecting their connectivity. Finding a solution in the Tutte disk will translate into a solution in the original connectivity representation. Such a projection is also known as harmonic projection (a.k.a. Tutte parameterization or barycentric parameterization).

7.3.2 Vertex mapping

The problem amounts to construct a one-to-one correspondence between vertices of the Tutte projection and of a regular grid. To assign every arbitrary connectivity to a fixed one is an assignment problem. This assignment problem is known as *bipartite matching*, which can be formulated into a unit capacity flow problem. The costs of the bipartite graph are the distances between a vertex in the Tutte projection and every vertex in the regular grid. Several methods already exist to solve this problem [33]. We proposed the augmenting path algorithm [69], which provides a solution within a couple of seconds for a 500-vertices patch. The fixed basis is sim-

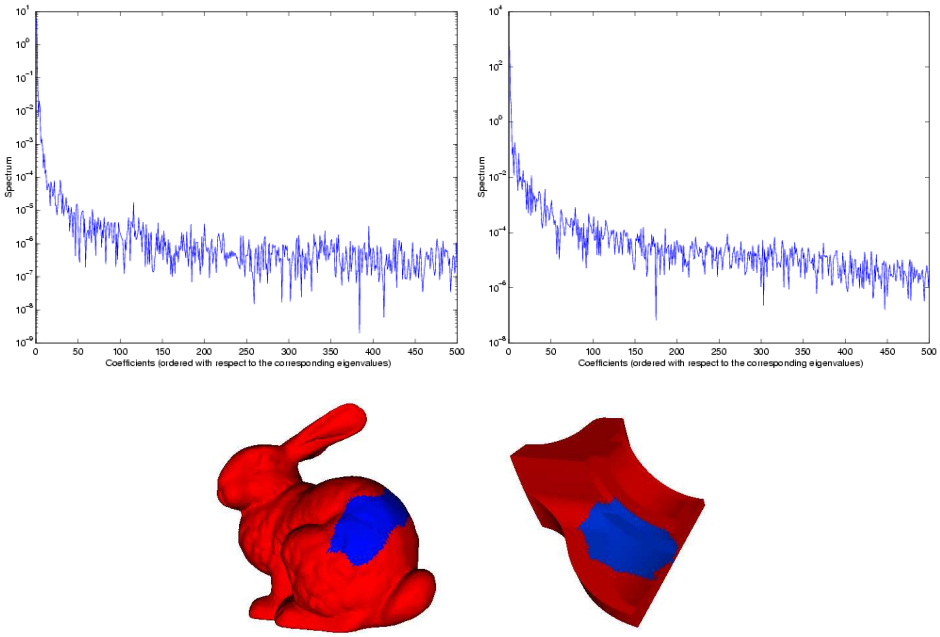


Figure 7.6: *Spectra (log scale) of the dark blue patch for models Bunny and Fandisk (optimal basis).*

ply obtained by adding a multiplication by a permutation matrix in the projective system (Eq. 7.6). Indeed, vertex mapping only renumbers the vertices. This step sets the overall decomposition algorithm complexity to $O(n^2)$ (computation of distances between vertices of the Tutte projection and those of the regular grid). The proposed heuristic has shown to perform within less than $O(n^2)$.

The main advantage of this method is the use of only one fixed basis for every patch, thus saving both memory and processing time. However, the major drawback is that decorrelation is of lower quality than if performed on the optimal basis. Since vertex mapping changes the connectivity, the geometry is still well reconstructed, but on an associated 6-regular connectivity. There is still decorrelation, but it is not performed on the mesh original connectivity. This drawback is well illustrated within the transmission problem, where the quality of the progressive reconstructed mesh is still significally improved up to the end of the bitstream.

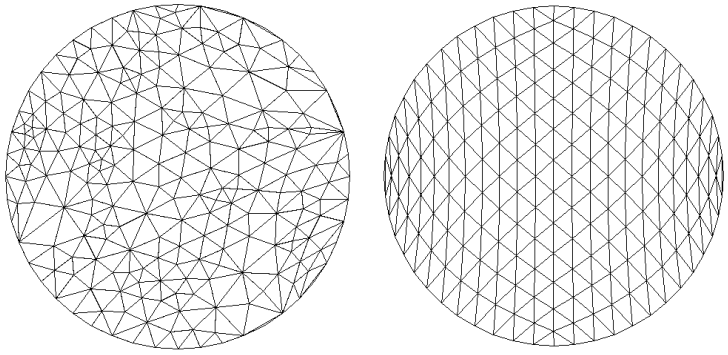


Figure 7.7: *A patch after Tutte projection. The vertices are located at the center of their neighbors. The vertices still have original connectivity but they now have 2D-only coordinates. The grid contains 196 vertices.*

7.3.3 Partitioning and patch augmentation by overlapping

For partitioning the mesh into patches, we have followed Karni et al. [74] by using the MeTiS software [76]. This general purpose approach takes the connectivity into account regardless of the geometry. The algorithm used to partition the mesh consists in iterative spectral bisections of the mesh (which minimizes the number of edges cut by the partitioning) as well as an heuristic which enables to retrieve well-balanced patches. This partitioning method provides patches with approximately 400 vertices. A regular mesh (as the one presented in Fig. 7.7) has an amount of vertices given by the square of an integer which for small size patches leads to values like $14^2 = 196$, $15^2 = 225$, $16^2 = 256$, etc. For computational reasons this number has been fixed to $23^2 = 529$. Therefore we have to increase the amount of vertices of any patch to 400 or 529. In their original work [75], Karni and Gotsman introduce some new vertices located in the center of existing triangles. This has no effect on the original geometry.

Instead of adding virtual vertices, we propose a patch augmentation by adding vertices located in adjacent patches, thus yielding redundancy in the representation because of the overlapping between patches. Vertices are added in a spiral way as described in Fig. 7.8 . In practice, it turns out that the Laplacian operator tends to smooth the geometry, preferably pushing the vertices in the interior of the patches. With this added re-

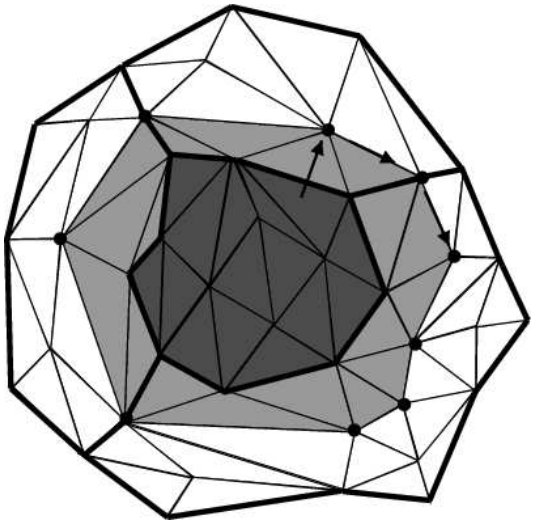


Figure 7.8: *Patch augmentation by overlapping : in this exemple the mesh is partitioned in* 4 *patches. The patch being augmented is represented in dark grey. The vertices selected for the overlapping on the neighboring patches are found in a spiral way as shown by the arrows. The patch resulting from the augmentation is represented in dark and light gray.*

dundancy, acting like a *Lapped Orthogonal Transform* (LOT) for meshes, we improve the quality of the reconstructed mesh, at a reduced extra cost. Vertices belonging to more than one patch are reconstructed by computing the mean of the geometrical values from the patches they belong to. Fig. 7.9 and Fig. 7.10 illustrate the artifacts caused by partitioning and the visual improvement due to patch augmentation. The artifacts appear on the triangles which link vertices belonging to different patches. As the Laplacian operator is a smoothing operator which tends to collapse each patch to its center of gravity, these triangles are stretched and become larger and larger when fewer low-frequency coefficients are used for reconstruction. At the limit when only the DC component is used for reconstruction, each patch is smoothed to its center of gravity. Indeed, the eigenvector corresponding to the eigenvalue whith maximal amplitude is constant, and the projection of the patch geometry results in computing its mean.

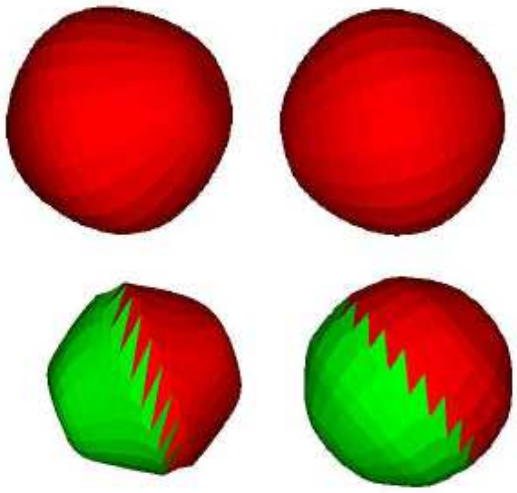


Figure 7.9: *Artifacts caused by partitioning. A sphere has been reconstructed, from left to right, with* 7 *and* 15% *of low frequency coefficients. On the first row, the sphere has been processed without partitioning, on the second row, two patches (green and red) have been created before spectral decomposition. The border of the patches shows large triangles. The stretching of these common triangles is due to the fact that each patch is smoothed (the Laplacian operator is a smoothing operator) towards its center of gravity.*

7.4 Mesh geometry compression and progressive transmission

The previous Section has presented how to decompose geometry over the mesh connectivity and how to reconstruct it. From a compression or transmission point of view, we have to make sure that connectivity has already been decoded before beginning decoding any spectral coefficient. If connectivity is sent first, the original patches yielding the basis functions may be easily retrieved. Then, one sends the geometrical information either with respect to the geometrical relevance of the coefficients or patch by patch. Compression is achieved by quantization of the spectral coefficients followed by entropy coding.

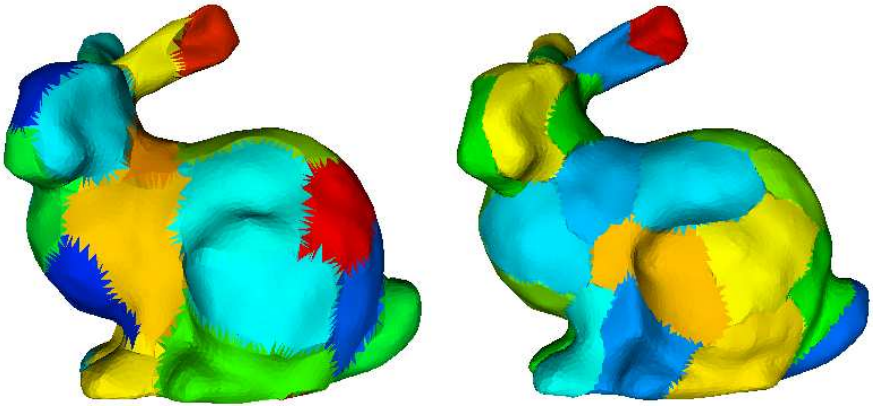


Figure 7.10: *Left: Bunny model with* 500*-vertices patches.* 5% *of the spectral coefficients are used for reconstruction. No patch overlapping.* $D_{vis} = 7.84 \times 10^{-6}$. *Right: bunny with* 400*-vertices patches, augmented to* 480 *vertices (20% overlapping).* 5% *of the spectral coefficients are used for reconstruction. The quality of the reconstruction is improved, particularly on the border of the patches. The basis functions are optimal (i.e: no vertex mapping), no fixed basis have been used to show the raw effect of overlapping.* $D_{vis} = 7.01 \times 10^{-6}$. *See Eq. 7.11 for visual distance definition.*

With a correct bitstream structure (see Fig. 7.11), it becomes easy to progressively send the spectral coefficients in the spectra order. However, in the case of on-the-fly decoding of the geometry, the first approximations of the models are often of poor quality. We address this issue with our patch augmentation strategy: the geometry keeps relatively smooth because overlapping has been introduced in the decomposition. The low frequencies are also better captured. In order to compare our results with those of Karni and Gotsman [74], we use their *visual metric*.

7.4.1 Visual metric

The visual metric of Karni et al. has been presented in Chapter 3. In a few words, this metric is characterized by a geometrical Laplacian for every point. This Laplacian is used to penalize the increase of the distance of a vertex to the center of its neighbors (i.e: the barycentric position). Let l_{ij} be the distance of point p_i to point p_j. The geometrical local Laplacian

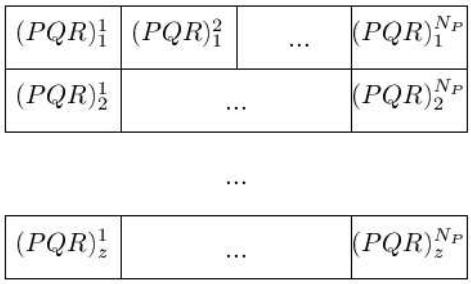


Figure 7.11: *Bitstream organization for the transmission of mesh spectra.*

operator GeL is defined as:

$$GeL(p_i) = p_i - \frac{\sum_{j \in \{i^*\}} l_{ij}^{-1} p_j}{\sum_{j \in \{i^*\}} l_{ij}^{-1}} \tag{7.10}$$

The local Laplacian represents the local smoothness of the mesh at point p_i. The visual distance between a mesh M and another mesh M' with the same connectivity must combine global distances between vertices and distances between local Laplacians. Such a visual distance D_{vis} then takes into account raw PSNR-like information and local smoothness difference:

$$D_{vis}(M, M') = \frac{1}{2n} \left(\sum_{i=0}^{n-1} \|p_i - p'_i\| + \sum_{i=0}^{n-1} \|GeL(p_i) - GeL(p'_i)\| \right) \tag{7.11}$$

This metric has the advantage to differentiate between random noise addition on the vertices and poor reconstruction quality: the eye better accommodates on a smooth object rather than on a noisy mesh. The metric increases in case of local disturbances which generally results in a poor quality to the human eye (see Fig. 7.10). Unfortunately, the order of magnitude of the visual metric depends on the mesh. The perceptibility threshold is not the same for *Bunny* and *Fandisk* models (resp. 10^{-6} and 10^{-5}).

7.4.2 Geometric compression

Partitioning has been performed with the MeTiS [76] algorithm, and matrix diagonalization with LAPACK [13]. The vertex mapping heuristic makes use of the augmenting path method [69]. The P, Q, R vectors are

uniformly quantized and entropy coded. A typical range of quantization for geometrical information is from 10 to 16 bits per coordinate. Actually, it turns out that spectral coefficients need more precision in their representation than spatial coefficients. At this stage we have used a uniform scalar

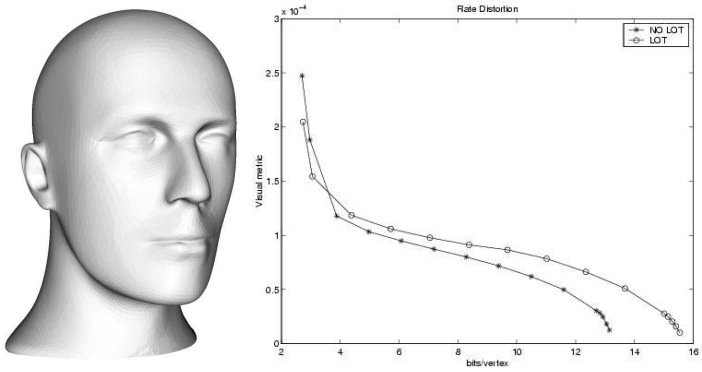


Figure 7.12: *Rate-distortion curves : Head model for compression (optimal basis functions). Uniform quantization at* 16 *bits. The perceptibility threshold is* $D_{vis} = 1.4 \times 10^{-4}$. *Overlapping results approximately in a* 1 *bit/vertex overhead, but provides much better quality at the beginning of the reconstruction.*

quantizer and a Huffman coder. Fig. 7.12 presents the rate/distortion curve for the *Head* model. Compression has been computed with optimal basis functions. It turns out that our algorithm better performs when using 14 bits for uniform quantization.

7.4.3 Progressive transmission

We have simulated on-the-fly decoding during progressive transmission. In order to better represent the geometrical artifacts of this transform, we have chosen a CAD object which has flat areas and orthogonal adjacent surfaces (see Fig. 7.13). In the case of such a CAD model like *Fandisk* model, one must wait for the whole data to get rid of spurious waves (similar to Gibbs effects) on the flat areas. This demonstrates that spectral decomposition used towards compression of CAD data is a bad choice since these models are locally not differentiable. However, the smoothing approximations for models coming from natural signals, like animals, faces and so on, are very good.

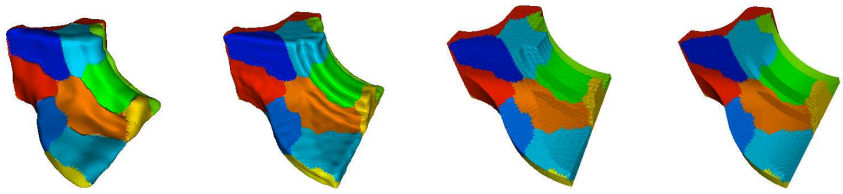


Figure 7.13: *Various stages of on-the-fly mesh geometry decoding. A* 20% *patch overlapping has been selected (*480 *vertices per patch after augmentation). On CAD objects, many coefficients are needed to better reconstruct the geometry. From left to right,* 5, 15, 60 *and* 80 *percent of the spectral coefficients have been used. The visual distortion, from left to right, is respectively of* $D_{vis}^{5\%} = 7.7 \times 10^{-5}$, $D_{vis}^{15\%} = 4.56 \times 10^{-5}$, $D_{vis}^{75\%} = 1.3 \times 10^{-5}$, $D_{vis}^{90\%} = 0.62 \times 10^{-5}$. *The model is well reconstructed from* $D_{vis} = 5 \times 10^{-6}$ *on.*

7.5 Watermarking

In this section we describe how to watermark the geometry of the mesh in the spectral domain. The watermarking scheme is substitutive. We aim at inserting a signature of 64 bits, as often required in many copyright protection applications. We use repetition as a basic channel coding scheme. This scheme is made private by the use of a secrete key, scrambling the watermark bits all over the mesh spectra.

7.5.1 Method

We randomly modify the relationship between the vectors P, Q, R to insert the watermark. The strength of the watermarking process is defined as the index i_0 at which we begin to insert the bits in the spectra. We do not watermark the low frequencies to ensure better imperceptibility. Every sum $S_i \ , \ i \geq i_0$ (see Eq. 7.9), potentially carries one bit of the watermark. We repeat the watermark as much as possible within the $(n - i_0)$ remaining coefficients. Furthermore, in order to avoid giving more importance to the first bits of the watermark during the retrieval process, we scramble it with a secrete key. Focusing on one set of three coefficients (P_i, Q_i, R_i), we sort

them:

$$(P_i, Q_i, R_i) \rightarrow (C_{min}, C_{inter}, C_{max}) \text{ with } \begin{cases} C_{min} = \min(P_i, Q_i, R_i) \\ C_{max} = \max(P_i, Q_i, R_i) \\ C_{inter} = \text{mid}(P_i, Q_i, R_i) \end{cases} \tag{7.12}$$

The interval $[C_{min}; C_{max}]$ of length $\Delta = C_{max} - C_{min}$ is divided into two equal intervals: $W_0 = [C_{min}; C_{min} + \Delta/2]$ and $W_1 =]C_{min} + \Delta/2; C_{max}]$.

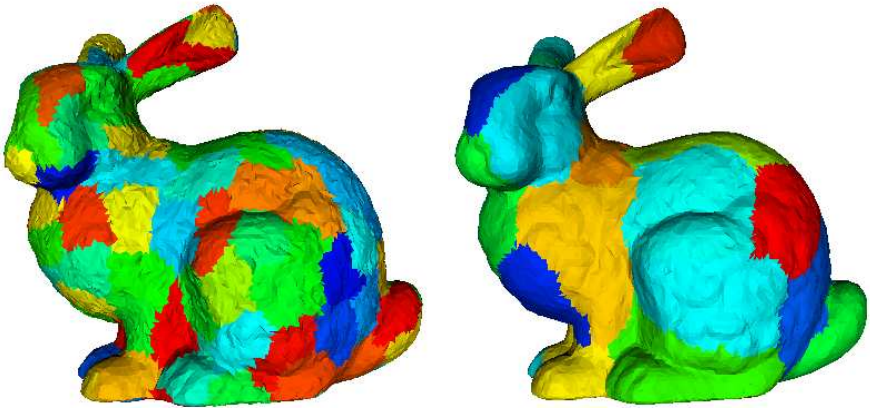


Figure 7.14: *Watermark insertion with the same strength. On the left, Bunny model with* 100*-vertices patches. The watermark is very noticeable. On the right, Bunny model with* 500*-vertices patches. The* 64*-bits watermark may not be distinguished. The* 500*-vertices size of the patches for compression is also well adapted to watermarking purposes.*

Embedding

If the bit to be embedded into the i^{th} set of coefficients is a 0 and C_{inter} is located in the W_0 subset, we do not change the triple. On the contrary, if C_{inter} lies in the W_1 subset, we flip it with respect to the center of the $[C_{min}, C_{max}]$ interval (see Fig. 7.15). The flipping distance is divided by a factor $k = 10$ chosen in order to ensure visual imperceptibility. The procedure is exactly symetrical in case of a 1 embedding. This way, the geometrical modification (if any) is randomly spread amongst the three sets of pseudo-frequencies. In the case of the patch overlapping, the watermark

is computed on the augmented patch but embedded only in inner points of the patch without overlapping.

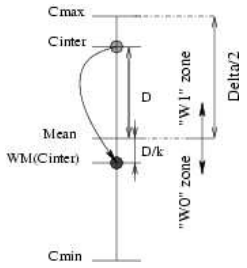


Figure 7.15: *Watermarking embedding process in the case of a* 0 *commitment. The* C_{inter} *value has to be flipped under the border. The scheme is the same in case of a* 1 *commitment. This is very similar to QIM watermarking [31].*

Retrieval

To retrieve the watermark, we simply read whether C_{inter} lies in W_0 or W_1. Descrambling is only possible for the owner of the secrete key. As said previously, the watermark is repeated several times to improve robustness. We adopt a simple majority voting strategy to determine the value of the embedded bit from the sets of all its embedded replications. Repetition has been shown to be a simple yet efficient channel coding strategy for watermarking applications [17], like the present one and watermark scrambling has been shown to drastically increase the recovery stage performance. Without scrambling, only the first bits are recovered, whereas all bits have equal probability to be recovered in the case of scrambling.

7.5.2 Results and discussion

We present here the results of our watermarking algorithm. We show the visual distortion as a function of the watermarking strength. For the *Bunny* model, the watermark is unnoticeable at a $D_{vis} = 2.5 \times 10^{-6}$ degradation level. The distortion threshold is not the same as for compression since the embedding process does not disturb the mesh in the same way. Secondly, we have tested the recovery of the watermark when the model is compressed, in order to demonstrate the robustness of our watermarking technique against quantization of the spectral coefficients.

Effect of overlapping on watermarking strength

We are now interested in evaluating the influence of overlapping on the watermarking process. As illustrated in Fig. 7.16, using overlapping enables to insert the watermark into lower frequencies coefficients. Further watermarking results will then feature fixed basis geometric transform with overlapping.

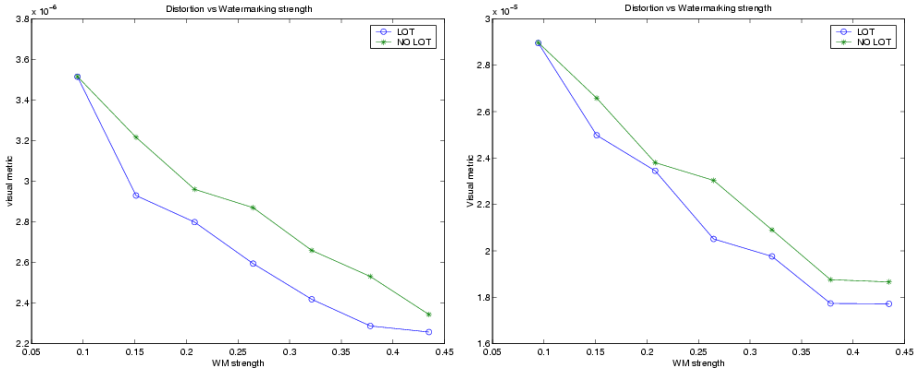


Figure 7.16: *Left: Distortion introduced by watermark embedding for decreasing insertion strength (Bunny model). Right: Distortion introduced by watermark embedding for decreasing insertion strength (Fandisk model). The overlapped transform embeds the watermark with less distortion. However, CAD models remain difficult to watermark with this technique.*

Perception of the spectral watermark

Figure 7.17 illustrates the spectral watermark perception for the *Bunny* model partitioned into 30 patches. The perceptive metric we have used is the one proposed in Chapter 4. It turns out that the watermark is perceived for watermark embedding forces superior to 70 percents of the spectrum (starting from high to low frequencies). Three representative distortions corresponding to this figure are illustrated in figure 7.18.

Partition and watermark embedding

Figure 7.19 represents the effect of the first Laplacian eigenvector. The center of gravity of each patch is moved into one canonical direction of

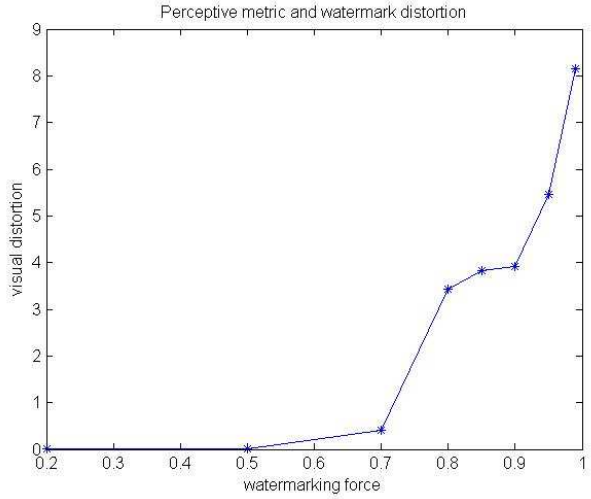


Figure 7.17: *Perceptive metric based on 2D views comparison with the Mutual Information criterion and watermark strength expressed in percentage of the total number of spectral components (starting from high to low frequencies i.e. for increasing embedding force). The values of the metric have been normalized between* 0 *and* 10 *following the experiment proposed in Chapter 4 on the* Bunny *model partitioned into* 30 *patches. The watermark turns visible only when more than* 70% *of the spectrum is watermarked.*

the 3D space. All the spectrum has been used for watermark embedding but the most perceptible distortion is due to the first eigenvector under partitioning. Such distortion is not present for small meshes which can be dealt with without partitioning.

Watermark robustness against spectral compression

As an attack of the embedded watermark, the spectral data are compressed with the method we have previously described. The strength of the compression is pictured as the number of quantization bits. As it can be seen in Fig. 7.20, the watermark embedded in model *Bunny* is fully recovered from 14 bits of quantization which is the lower bound for compression purposes. In the case of the *Fandisk* model, the two plots are exactly the same. Overlapping has shown to be of specific relevance during watermark embedding, though it has no influence on recovery.

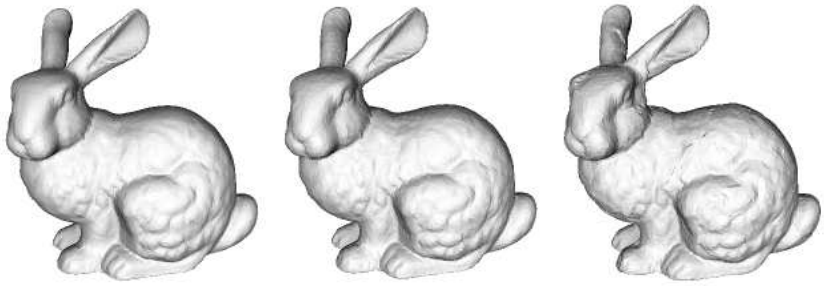


Figure 7.18: *From left to right, three views of the* Bunny *model with watermarking forces of* 50%, 80% *and* 99% *respectively. The corresponding measured distortions are represented on figure 7.17.*

Watermark robustness against geometrical attacks

Finally, we have evaluated our watermarking algorithm against common geometrical attacks. We have only considered connectivity-preserving manipulations. Indeed, without the availability of the original model, it is impossible to retrieve the original connectivity, partition and spectral basis after non connectivity-preserving attacks. We have studied additive random noise on the vertices coordinates, Laplacian smoothing of the mesh and RST transforms. For every attack, we give the maximum parameter value that still enables 64-bit recovery. Considering noise addition, we have focused on the noise power as the attack parameter. The parameter for smoothing is the number of iterations of the smoothing algorithm. The watermark only survives rotations up to 15°. This sensitivity to rotation can be easily overcome by using a preprocessing self-registration with principal axes obtained by PCA (of each patch). However, it is interesting to see that rotation perturbations inferior to 15° do not affect the watermark retrieval. This is important since the PCA axis usually present some orientation instabilities of a few degrees after smoothing or noise addition [23]. This robustness can be explained by the fact that the embedding for a given frequency proceeds by flipping the median spectral component. A rotation modifies, for each frequency, the amplitude of each spectral component but preserves the sum of their squares. It follows that for large rotations, the relative order of P_i, Q_i, R_i can change and the retrieval of the watermark is then impossible. In the case of small amplitude rotations, the relative order of the spectral components is not significantly modified,

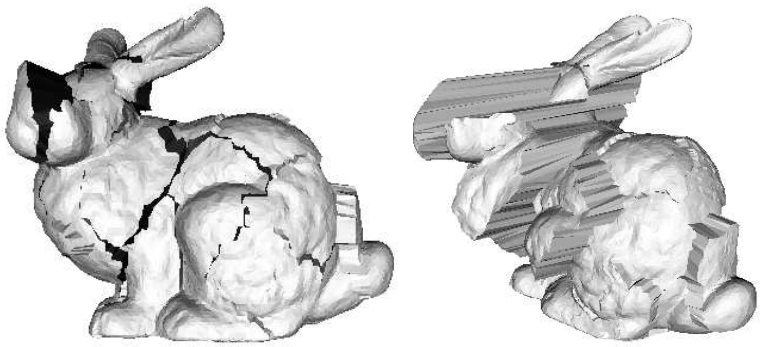


Figure 7.19: *Two different views of the* Bunny *model partitioned into* 30 *patches. All the spectrum has been used for watermark embedding. Watermarking the first Laplacian eigenvector translates the center of gravity of each patch into the watermarked direction.*

thus enabling retrieval. Scaling and translation attacks have no impact in watermark retrieval.

7.6 Conclusion, limitations and future work

Our goal has been to validate the spectral decomposition of mesh geometry as a powerful tool to both compress and watermark the geometry. The principal limitations of this work are related to the partitioning algorithm as well as to the computational cost of the spectral decomposition.

The partitioning often leads to patches that have not a planar topology (e.g. cylindrical patches). It is then impossible to perform a Tutte parameterization and even with optimal basis functions, the smoothing leads to poor results. We propose in the next Chapters some methods (e.g. geodesic Voronoi Diagrams) that enable to get rid of this problem.

For the computational cost, some new methods have recently been proposed to compute the spectral basis functions with RBF functions defined in uniformly distributed seeds [135]. But it is unclear whether these methods can be extended to blind watermarking because a registration step is needed to exactly retrieve the positions of these seeds.

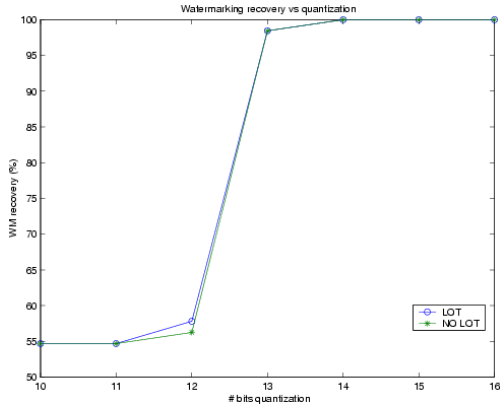


Figure 7.20: *Watermark decoding results for compression attack on* Bunny *model. Our scheme is resistant to classical quantization for spectral coefficients (from* 14 *bits for quantization). Overlapping has no effect on the watermark recovery process. The watermark was embedded at the limit of perceptibility using fixed basis.*

Concerning the proposed watermarking scheme, we have pointed out the watermark cannot be retrieved after connectivity modifications. For a copyright protection application scheme, these attacks are very common and should therefore be taken into account. This is what we propose in the next Chapter.

This work, made in collaboration with François Cayre, has also led to a progressive geometry compression scheme and two publications [8, 28].

Models	Attacks	noise	smoothing	translation	scaling	rotation
bunny		0.44%	20	OK	OK	$< 15°$
horse		0.37%	20	OK	OK	$< 15°$
venus		0.25%	20	OK	OK	$< 15°$
head		0.25%	20	OK	OK	$< 15°$

Figure 7.21: *Various robustness results of our watermarking algorithm against some geometrical attacks that preserve connectivity.*