Mesh Discriminative Features for 3D Steganalysis

Ying Yang, Yale University, USA
Ioannis Ivrissimtzis, Durham University, UK

We propose a steganalytic algorithm for triangle meshes, based on the supervised training of a classifier by discriminative feature vectors. After a normalization step, the triangle mesh is calibrated by one step of Laplacian smoothing and then a feature vector is computed, encoding geometric information corresponding to vertices, edges and faces. For a given steganographic or watermarking algorithm, we create a training set containing unmarked meshes and meshes marked by that algorithm, and train a classifier using Quadratic Discriminant Analysis. The performance of the proposed method was evaluated on six well-known watermarking/steganographic schemes with satisfactory accuracy rates.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Hierarchy and geometric transformations*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Authentication*

General Terms: Security

Additional Key Words and Phrases: Triangle meshes, discriminative feature vector, data embedding, steganalysis

1. INTRODUCTION

Watermarking/steganography refers to the process of embedding a *watermark* over cover signals, such as digital images, video or 3D models. Typical applications include copyright protection, integrity authentication and covert communications. As a counterpart to watermarking/steganography, *steganalysis* aims to detect the existence of watermarks in a given signal.

While there is no generally accepted distinction between watermarking and steganography, in the image processing literature, the term steganalysis is most commonly used when the undetectability of the message under probing by steganalytic algorithms is a concern. On the other hand, as there are no steganalytic algorithms for 3D models, in the geometric modelling literature, the term watermarking is most commonly used for algorithms aiming at embedding small payloads that are resistant to removal by malicious attacks, while steganography usually refers to high capacity algorithms.

In this paper, we propose a steganalytic method for triangle meshes, which is the ubiquitous standard in digital surface representation, adopting a framework which has been successfully developed for image steganalysis. The main idea is that even though the presence of the watermark is often imperceptible to the human eye, it may nevertheless disturb the natural statistics of the 3D signal and thus become detectable.

Y. Yang is with the Department of Computer Science, Yale University, USA. I. Ivrissimtzis is with the School of Engineering and Computing Sciences, Durham University, UK. E-mail: ying.yang.yy368@yale.edu and ioannis.ivrissimtzis@durham.ac.uk.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

We implement this idea by computing for each mesh a characteristic feature vector capturing geometric information extracted from its Cartesian and Laplacian coordinates, its dihedral angles and face normals. For the extracted feature vector to have sufficient for our purposes discriminative power, we do not compute it directly on each mesh, but on the difference between the mesh and a filtered copy of it called the *reference* mesh. This technique is known in the literature of image steganalysis as *calibration*. Here, the reference mesh is produced by applying one iteration of Laplacian smoothing.

In [Fridrich et al. 2002], where calibration was introduced, the reference image was deemed to serve as an approximation of the original image before the embedding of the watermark. However, as [Kodovský and Fridrich 2009] points out, this approximation does not need to hold for the calibration to work. Instead, what we expect from the calibration is to erase a part of the changes introduced by watermarking, without altering the cover significantly. Then, as a result, the difference between a mesh and its reference will be distinctively larger for watermarked than for clean models. From this point of view, what our experiments demonstrate is that when one of the six test watermarking/steganographic algorithms is used, the difference between a mesh and its smoothed version is distinctively larger for marked meshes, when a certain well-chosen statistical measure of that difference is used.

After the extraction of the feature vector, a supervised learning algorithm based on Quadratic Discriminate Analysis is applied to a training set of feature vectors from unmarked meshes and meshes marked by a given watermarking/steganographic algorithm, yielding a steganalytic classifier for that particular method. We also obtained a universal steganalyzer by training the classifier on a set of meshes marked by all six watermarking/steganographic test algorithms.

The adopted machine learning approach, which has already been successfully employed in image steganalysis, seems to be even more suitable for triangle meshes. Indeed, given the irregularity of triangle mesh representations of surfaces, the alternative approach of applying signal decomposition for analysis or processing is expected to be extremely challenging and often gives poor results.

Contribution: We propose a steganalytic technique for triangle meshes, which, to the best of our knowledge, is the first 3D steganalytic method in the literature. Experiments on six well-known steganographic/watermarking algorithms show that the method can be successfully used as a benchmark for the anti-steganalysis performance of existing and future 3D steganographic/watermarking algorithms.

Limitations: Given that there are no other steganalytic methods to compare our results against, it is difficult to judge the significance of the obtained accuracy rates. However, we may assume that as it is the case with image steganalysis too, one may achieve higher accuracy rates by devising algorithms targeting specific steganographic/watermarking methods.

A second limitation is that the proposed steganalytic method only targets algorithms that hide information into the mesh geometry. Even though such algorithms are the mainstream of 3D steganography/watermarking, it should be noted that there are also algorithms hiding information into the mesh connectivity, or into the data redundancy of polygonal list files encoding triangle meshes. In its current form, our method cannot detect watermarks inserted by such algorithms.

As a third limitation, we notice that, in its current form, the proposed algorithm can detect but not remove a watermark. Depending on the steganographic algorithm, watermarks are usually removed with standard mesh processing attacks, such as smoothing, mesh simplification, or noise addition. In practice, a steganalytic algorithm can facilitate such attacks by confirming the removal of the watermark. We also notice that, in its current form, the proposed algorithm does not estimate the parameters of the steganographic algorithm used for watermark insertion.

Even though the proposed steganalytic algorithm has been tested against six popular 3D steganography/watermarking methods, we note that there are several other existing methods we have not tested

the algorithm against, as for example the recently proposed multi-channel watermarking algorithms based on the manifold harmonic transform.

2. RELATED WORK

In this section, we briefly review 3D watermarking/steganography. For more details on the relevant literature we refer the reader to the survey in [Wang et al. 2008]. In addition, the image steganalysis methods relevant to our approach are also reviewed.

3D Watermarking: Similarly to image watermarking, 3D watermarking approaches work either in the spatial domain [Yeo and Yeung 1999] [Lin et al. 2005] [Cho et al. 2007] [Yang and Ivrissimtzis 2010], or in the frequency domain [Ohbuchi et al. 2002] [Praun et al. 1999] [Uccheddu et al. 2004] [Yin et al. 2001]. For 3D model verification, Yeo et al. [Yeo and Yeung 1999] propose a watermarking method which perturbs each vertex to ensure that two predefined hash functions have the same value on it. Due to the heavy dependence on the order of traversal of the vertices, this method has the causality problem. Lin et al. [Lin et al. 2005] address this issue by taking advantage of vertex-order-independent hash functions. Based on modifying the mean value and variance of the distribution of vertex norms, Cho et al. propose two blind robust watermarking algorithms in [Cho et al. 2007]. Instead of modifying the vertex norms, Luo et al. [Luo and Bors 2011] propose two surface-preserving robust 3D watermarking approaches through changing the mean and the variance of geodesic distance distributions. Based on the fact that human eyes are less sensitive to changes at a rough surface patch than those at a smooth surface, Kim et al. [Kim et al. 2010] present an adaptive 3D watermarking algorithm that varies the strength of watermark depending on the masking effect of surface roughness. Recently, Yang et al. [Yang and Ivrissimtzis 2010] presented a watermarking algorithm based on the modification of the lengths of Laplacian coordinate vectors. Their method is able to resist common mesh editing attacks to some extent, due to the good behaviour of the Laplacian coordinate under such operations.

Another category of robust 3D mesh watermarking methods is based on frequency domain analysis. Using the mesh spectral analysis of Karni et al. [Karni and Gotsman 2000], Ohbuchi et al. [Ohbuchi et al. 2002] insert the watermark into the low frequencies. Praun et al. [Praun et al. 1999] employ Hoppe's multiresolution decomposition [Hoppe 1996] to find the perceptually significant features of a given 3D model and insert the watermark into these features. The watermarking algorithms by Uccheddu et al. [Uccheddu et al. 2004] and Yin et al. [Yin et al. 2001] use the multiresolution analysis in [Lounsbery et al. 1997] and [Guskov et al. 1999], respectively. Konstantinides et al. [Konstantinides et al. 2009] propose a robust, blind 3D watermarking algorithm by mapping regions of the mesh onto oblate spheroids and embedding the watermark into their spheroidal harmonic coefficients. Liu et al. [Liu et al. 2008] propose a spectral watermarking algorithm based on the Manifold Harmonic Transform (MHT) [Vallet and Lévy 2008]. In [Liu et al. 2012] the same authors present a spectral watermarking algorithm based on Dirichlet Manifold Harmonic Transform (D-MHT) [Liu et al. 2010]. Due to the properties of the harmonic transform, the two watermarking approaches are robust against various attacks, including connectivity modification, noise addition and smoothing.

3D Steganography: Motivated by the idea in [Cayre and Macq 2003], Wang et al. [Wang and Cheng 2005] increase the embedding capacity and reduce the distortion by employing a multi-level embedding procedure. Bogomjakov et al. [Bogomjakov et al. 2008] propose a distortion-free steganographic method, using the redundancy in the indexed representation of a mesh by permuting the order in which faces and vertices are stored. Tu et al. [Tu et al. 2010] improves the efficiency in the original mapping of [Bogomjakov et al. 2008], further increasing its capacity. Chao et al. [Chao et al. 2009] propose a high-capacity, low-distortion method based on vertex projection onto the principal axis. Given a

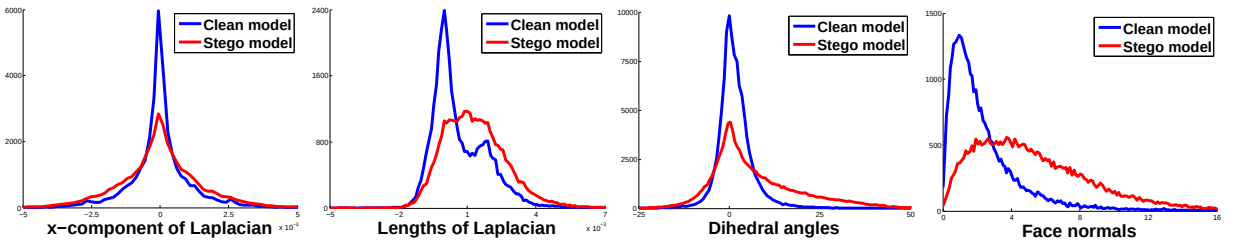

Fig. 1. Comparison of the histograms before and after embedding with the variance-based watermarking in [Cho et al. 2007] on the Stanford Bunny. The y axis shows the frequency. The histograms were constructed from the x component of Laplacian coordinates **(Left)**, the differences of the lengths of Laplacian coordinate vectors **(Second from Left)**, the differences of the dihedral angles **(Second from Right)** and the differences of the angles between face normals **(Right)**.

tolerance for the normal distortion, Yang et al. [Yang et al. 2013] compute an appropriate quantization level for the mesh vertices and replace the unused Least Significant Bits (LSB) with watermark bits.

Image Steganalysis: The calibration method was introduced in [Fridrich et al. 2002] for attacking F5 steganography [Westfeld and Pfitzmann 2001]. The idea of analyzing histograms of characteristic functions has been implemented among others by Ker [Ker 2005] for LSB steganalysis of grayscale images.

A major challenge in developing universal steganalytic algorithms is the identification of features that are modified by watermark embedding. Farid [Farid 2002] proposes a universal approach that uses a wavelet-like decomposition to build higher-order statistical models of natural images. Rather than using empirical probability density function (PDF) moments, Xuan et al. [Xuan et al. 2005] and Harmsen et al. [Harmsen and Pearlman 2003] employ empirical characteristic function (CF) moments for blind steganalysis. Wang et al. [Wang and Moulin 2007] decompose images into groups of data samples and use the informative features from empirical PDF and CF moments of subband images for universal steganalysis.

3. TRIANGLE MESH STEGANALYSIS

Given a target watermarking/steganographic algorithm, the steganalytic algorithm extracts N-dimensional feature vectors $\mathbf{F}_i$ from a training set of clean and watermarked models

$$\mathbf{F}_i = (f_{i,1}, f_{i,2}, \cdots, f_{i,N}) \tag{1}$$

where i is the index of the model in the training set and $N = 208$ in the paper. These feature vectors are the input of a supervised learning algorithm, which produces a classifier associated with the target watermarking/steganographic algorithm.

3.1 Normalization

In a pre-processing step, the model is normalized by a coordinate system change before feature extraction. We apply Principal Component Analysis (PCA) to the vertex coordinates, and align the xyz axes of the coordinate system with the three principal directions $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$, respectively, assuming, without loss of generality that $\lambda_1 \geq \lambda_2 \geq \lambda_3$ for the corresponding eigenvalues. After this coordinate system transformation, we uniformly scale the model into the unit cube centered at $(0.5, 0.5, 0.5)$.

The normalization ensures that the feature vector $\mathbf{F}_i$ of each model is invariant under affine transformations. Also, normalization restricts the values of each component $f_{i,j}$ of $\mathbf{F}_i$ to a limited range and thus, prevents the large feature values dominating the smaller values. Notice that because of the use

of PCA, we are not able to specify the orientation of the axes of the new coordinate system. That means that all extracted features should be invariant under an orientation change of any of the axes.

3.2 Calibration

The reference mesh M' we use for calibration is produced by applying one iteration of Laplacian smoothing on the original mesh M. We use a standard Laplacian operator corresponding to the *Kirchhoff* matrix [Bollobás 1998] with entries

$$\mathbf{R}_{i,j} = \begin{cases} \mathrm{val}(v_i) & \text{if } i = j \\ -1 & \text{if } v_j \in \mathcal{N}(i) \quad 1 \le i, j \le V \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where v_i is the vertex indexed by i, $\mathrm{val}(v_i)$ and $\mathcal{N}(i)$ denote the valence and the 1-ring neighborhood of v_i, and V is the number of mesh vertices.

We have found that this simple calibration process works well against all watermarking algorithms we have tested the proposed method on. Moreover, its simplicity gives a reasonable expectation that the experimental results will generalize well against other watermarking algorithms that have been proposed, or will be proposed in the future.

3.3 Feature Extraction

The computation of the feature vector $\mathbf{F}_i$ is a two-step process: *extracting features* and *computing components of* $\mathbf{F}_i$. More specifically, we first compare the original and the reference models and compute vectors with components corresponding to mesh vertices, edges or faces. Next, these vectors are processed to produce the components of the feature vector $\mathbf{F}_i$.

Regarding the vertex vectors, we compute the absolute values of the differences of the x, y and z coordinates of M and M', as well as the length of the vector of Cartesian coordinate differences. Essentially, these components are the absolute values of the Laplacian coordinates and the length of the vector of Laplacian coordinates of M. Next, we obtain four more vectors by computing the same absolute differences but on the Laplacian rather than the Cartesian coordinates of M and M'. The computations are done separately on vertices with valence less than, equal, or greater than six, excluding all boundary vertices. That is, we treat separately vertices that are topologically convex, planar, or concave in the combinatorial Gaussian curvature sense [Higuchi 2001]. The total number of vectors obtained from the mesh vertices is 24.

Regarding the edges of the mesh, we compute the vector of the absolute values of the differences of the dihedral angles between M and M'. Finally, regarding mesh faces, we compute the vector of the angles between the normals of M and M', obtaining a total of 26 vectors from vertices, edges, and faces.

Fig. 1 shows the changes in the histograms of the extracted features induced by the variance-based watermarking in [Cho et al. 2007] when applied on the Stanford *Bunny*. The observed large differences in the shape of the histogram between the clean and the marked *Bunny* model illustrate in a nutshell why the proposed method works.

3.4 The Feature Vector $\mathbf{F}_i$

From each of the 26 vectors computed in the previous section, we compute eight components of the feature vector $\mathbf{F}_i$, creating a vector of dimension $N = 208$. Let $\mathbf{f}$ be one of these 26 vectors. The first four components of $\mathbf{F}_i$ corresponding to $\mathbf{f}$ are the mean, variance, skewness and kurtosis of $\log(|\mathbf{f}|)$. Notice that these four statistical characteristics are commonly used in image staganalysis [Farid 2002]

as vector shape descriptors. The purpose of the log transform is the reduction of the range of the values and also an increase in the weight of small positive values.

For the other four components, we first build the histogram of $\mathbf{f}$ with

$$H = \left\lceil \frac{\max(\mathbf{f}) - \min(\mathbf{f})}{h} \right\rceil \tag{3}$$

bins, where the size of the bin is given by the Freedman–Diaconis rule [Freedman and Diaconis 1981]

$$h = 2\,\frac{\mathrm{IQR}(\mathbf{f})}{n^{1/3}}, \tag{4}$$

where $\mathrm{IQR}(\cdot)$ denotes interquartile range and n is the number of components in $\mathbf{f}$. By counting the number of elements in each bin, we obtain the frequency vector $\mathbf{n} = (n_1, n_2, \cdots, n_H)$ and its difference vector $\mathbf{n}' = (n_2 - n_1, n_3 - n_2, \cdots, n_H - n_{H-1})$. Then, the four remaining components of $\mathbf{F}_i$ obtained from $\mathbf{f}$ are the mean, variance, skewness and kurtosis of $\log(|\mathbf{n}'|)$.

We notice that the dimension of the vector $\mathbf{F}_i$ is relatively low, compared for example with the PEV-274 [Pevný and Fridrich 2007], which has dimension 274 and is used for image steganalysis. Indeed, given that images are structurally simpler than triangle meshes, one would expect that a good discriminative feature vector for meshes would have a higher dimension. Nevertheless, the proposed feature vector works well in practice.

3.5 The Classifiers

We train the classifiers using the feature vectors extracted from a training set of 3D models with and without watermarks. Before training the classifiers, the features are scaled into comparable dynamic ranges. More specifically, for any component f of the feature vector, we find its minimum $f_{\min}$ and maximum $f_{\max}$ values on the set of all training models. For any training or test model, we scale the feature component f by

$$\bar{f} = \frac{f - f_{\min}}{f_{\max} - f_{\min}} \tag{5}$$

Notice that $\bar{f} \in [0, 1]$ for all training models, while it is expected that $\bar{f}$ will also fall in [0, 1] for most test models. This scaling process prevents the features with large numerical ranges from dominating those with small numerical ranges and thus greatly improve classification accuracy [Wang and Moulin 2007].

Finally, the classifier is obtained via quadratic discrimination that fits multivariate normal densities with covariance estimates stratified by group [Krzanowski 2000]. Notice that the simpler Fisher Linear Discriminant has been successfully employed in prior steganalysis work [Farid 2002]; however, we have empirically found that in our case it is slightly outperformed by the quadratic discrimination. We have also tested Adaboost classifiers based on Linear and Quadratic Discriminants, but there was no improvement on the results.

When the classifiers for each target embedding method have been computed, the steganalysis process is straightforward. Given a test 3D model, we just compute its feature data, apply the classifier for a particular steganographic method to the feature data and assign it to one of the two categories: *unmarked* by that method or *marked* by that method.

4. EXPERIMENTAL RESULTS

In this section, we experimentally validate the proposed steganalytic algorithm against six well-known embedding techniques. They include LSB modification [Yang et al. 2013], the high-capacity stegano-

graphic scheme [Chao et al. 2009], two watermarking methods proposed in [Cho et al. 2007], the Laplacian coordinate-based watermarking method in [Yang and Ivrissimtzis 2010] and the frequency-based watermarking in [Ohbuchi et al. 2002]. Before presenting the validation results, we shall briefly describe these six embedding approaches in the next sub-section.

The experimental dataset consists of the 360 models in Princeton Shape Benchmark [Shilane et al. 2004] and four models from the Stanford 3D Scanning Repository. We assumed that all these 364 downloadable models were unmarked. Notice that the 3D models in the dataset vary greatly in overall shape and size. Fig. 2 illustrates a part of the experimental dataset.

All algorithms have been implemented in MATLAB and C++ and tested on a PC running on an Intel Core 2 Duo T6570 2.1 GHz processor with 2 GB memory. In our current non-optimized implementation, it takes about 40 minutes to compute the feature vectors in the training dataset and less than one second to construct a classifier. Notice that this one-off training process is the computational bottleneck of the algorithm. After the classifiers have been computed, it takes about six seconds to analyze a test model with 10,000 vertices.

4.1 Description of Implemented Methods and Parameter Settings

LSB Modification Steganography [Yang et al. 2013]: This simplest form of steganographic algorithm works by first computing an appropriate quantization level for each mesh vertex and for a given tolerance ϵ of face normal degradation, and then by simply repalcing the less significant bits with message bits. For the extraction of the embedded message we read the less significant bits on each vertex coordinate, after some coordination problems related to the computation of the appropriate quantization level in the clean and the marked models have been resolved. The method is adaptive, embedding different numbers of bits into distinct vertices, depending on the shapes of their neighborhoods. The method appears to outperform any previously proposed steganographic methods in terms of embedding capacity and distortion control.

In the experiments, we vary the value of ϵ of the tolerance for the expected face normal degradation, simulating various amounts of embedding distortion. In particular, we use $\epsilon = 1^\circ$, $\epsilon = 2^\circ$ and $\epsilon = 5^\circ$.

High-capacity Steganography [Chao et al. 2009]: This method starts with a coordinate system transformation to the three principal axes of the 3D model, then uniformly divides the line segment whose two end points are the furthest projections of the mesh vertices on each principal axis into a set of intervals, and finally assigns to each segment a two-state object (0 or 1) in an interleaved manner. Next, each segment is further divided into a *change region* and an *un-change region*. Finally, the message embedding takes place by either keeping a vertex intact, or moving it into the change region of an interval, depending on both the message bit to be inserted and the state of the interval into which the vertex is projected. This single-layered embedding can be easily extended to a multi-layered version, achieving this way high embedding capacity and low embedding distortion.

Regarding the parameter setting, in our tests we use $n_{\text{layers}} = 10$ for the number of layers and $n_{\text{intervals}} = 10000$ for the number of intervals.

Two variants of Robust Watermarking [Cho et al. 2007]: Cho et al. propose two watermarking algorithms based on modifying the radial coordinate of the mesh vertices in a spherical coordinate system. The main ideas of two methods are very similar, the difference being that one alters the mean of the radial coordinates of a set of vertices, while the other changes the variance. More specifically, they first build the histogram of the radial coordinates and normalize the radiuses in each bin into the range [0, 1]. The embedding process is based on the assumption that the mean of the normalized radiuses in each bin is eexpected to be equal to 1/2 and the variance 1/3. By increasing or decreasing the normalized radiuses in each group depending on the value (0 or 1) of the watermark bit to be

Table I. Clean and marked models in the training and test datasets.

Method	Training Dataset		Test Dataset	
	#Clean	#Marked	#Clean	#Marked
Cho's Mean	260	267	104	92
Cho's Variance	260	255	104	98
Laplacian-based	260	263	104	101
Chao's	260	262	104	100
LSB	260	262	104	100
Ohbuchi's	260	250	104	99
All Methods	260	410	104	146

embedded, the make the mean (or variance) of the marked normalized radiuses smaller or greater than 1/2 (or 1/3). The watermark extraction can be carried out blindly by comparing the mean (or variance) of the normalized marked radiuses in each bin with 1/2 (or 1/3).

Regarding the parameter setting of our test, the incremental step size is fixed at $\Delta k = 0.001$ and the number of bins at 64, while we vary the value of the strength factor α for different models. In particular, α is a random number within the ranges $[0.02, 0.06]$ and $[0.12, 0.20]$ for mean-based and variance-based watermarking, respectively. Notice that the purpose of varying α is to simulate the situation of incurring different amounts of embedding distortion to different 3D models.

Laplacian-based Watermarking [Yang and Ivrissimtzis 2010]: Yang et al. propose a watermarking algorithm robust against mesh editing attacks changing the global shape of a mesh. The basic idea behind this method is to embed the watermark into the histogram of the lengths of the Laplacian coordinate vectors. More specifically, it encodes a watermark bit by changing the heights of two adjacent histogram bins via transferring some elements from one bin to another. To decode a watermark bit, it constructs the histogram of the Laplacian lengths on the marked model and then compares the heights of two neighboring histogram bins. Again, the watermark extraction can be executed blindly, with no reference to the original mesh.

In our tests, the embedding threshold is set at $n_{\text{thr}} = 46$, the robustness threshold at $n_{\text{robust}} = 5$, while the number of bins is equal to $\lceil V/70 \rceil$, where V is the number of mesh vertices.

Frequency-based Watermarking [Ohbuchi et al. 2002]: The robust 3D watermarking method by Ohbuchi et al. is considered a classic in frequency-based watermarking. Before watermark embedding, the original cover mesh is divided into a set of patches. The purpose of this segmentation is to reduce the computational time of the subsequent spectral computations, which can be very time-consuming on large matrices. Next, the method computes the low frequency spectrum of the *Kirchhoff* matrix of each patch, computing the m smallest eigenvalues and their corresponding eigenvectors using the *Arnoldi* method [Golub and Van Loan 1996]. Each vertex of the patch is projected onto these m low frequency normalized eigenvectors and the watermark is embedded via additive modulation of the spectral coefficients of the patches. To further improve robustness, each watermark bit is repeatedly inserted with a chip rate c. It is a non-blind method and thus requires the original mesh at the stage of watermark extraction.

In our tests, we fix the modulation ratio at $\beta = 0.001$, the watermark chip rate at $c = 15$ and the number of eigenvectors used for embedding at $m = 160$.

4.2 Specific Steganalyzer

To construct a steganalyzer specific for each one of the six embedding methods, training and test datasets were produced separately. Table I shows the number of clean and marked models in both datasets for each watermarking method.

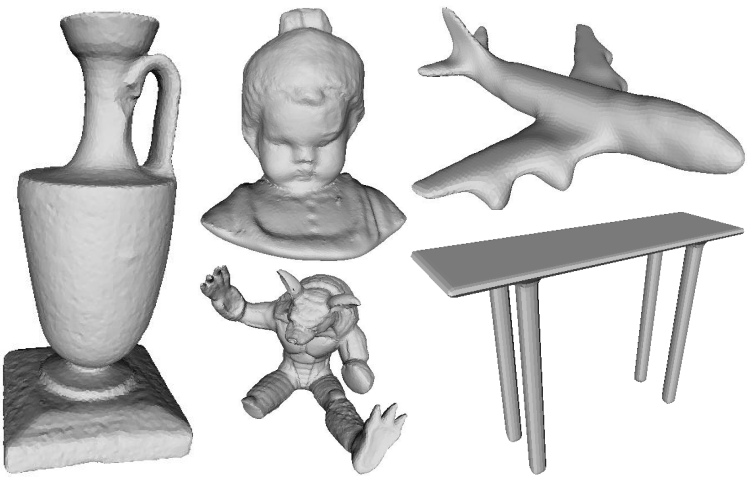

Fig. 2. Some of the models of the experimental dataset.

Regarding the output of the experiments, apart from measuring the accuracy of the constructed steganalyzers, we would also want to be able to detect any possible large information redundancy in the feature vector $\mathbf{F}_i$, which could indicate that our selection of features was not optimal. For that reason we use PCA to reduce the dimension of $\mathbf{F}_i$. Figs. 3 and 4 plot the detection accuracy of the steganalyzers, computed on the test sets of the six implemented embedding techniques, against the number of principal components retained from the feature vector. We notice that the right choice of number of principal components boosts in all cases the accuracy to above 80%. However, the figures also show near optimal rates when all components of $\mathbf{F}_i$ are retained. We also notice that, in all cases, the specificity increases monotonically with the number of principal components, thus, there is no compelling evidence suggesting the use of PCA for reducing the dimension of the feature vector. We interpret this result as showing that there is no considerable redundancy in the feature vector $\mathbf{F}_i$, justifying this way the choice of its components since each one of them is likely to contribute positively to the recognition process.

From Fig. 4 we deduce that, as expected, the detection accuracy increases monotonously with the amount of embedding-induced distortion. That is, we obtain better results on marked models with large distortion. The comparison of Figs. 3 and 4 indicates that, in descending order, the anti-steganalysis performance of the six tested methods ranks as follows: Yang's watermarking, Chao's steganography, Cho's mean-based watermarking, LSB modification, Cho's variance-based watermarking and Ohbuchi's method. The method works particularly well against Ohbuchi's frequency-based watermarking with a detection rate of about 99%.

4.3 Universal Steganalyzer

In addition to the six specific steganalyzers, we also constructed a universal steganalyzer. Our main motivation is that a universal steganalyzer can be used as a benchmark for measuring the antisteganalysis performance of other existing and most importantly future watermarking/steganographic algorithms. The training set was created by randomly selecting 260 clean models from the dataset, creating 410 marked models by randomly selecting clean models from the training set and marking them with a randomly chosen embedding algorithm, and nally mixing them all together into a training set of 670 models.

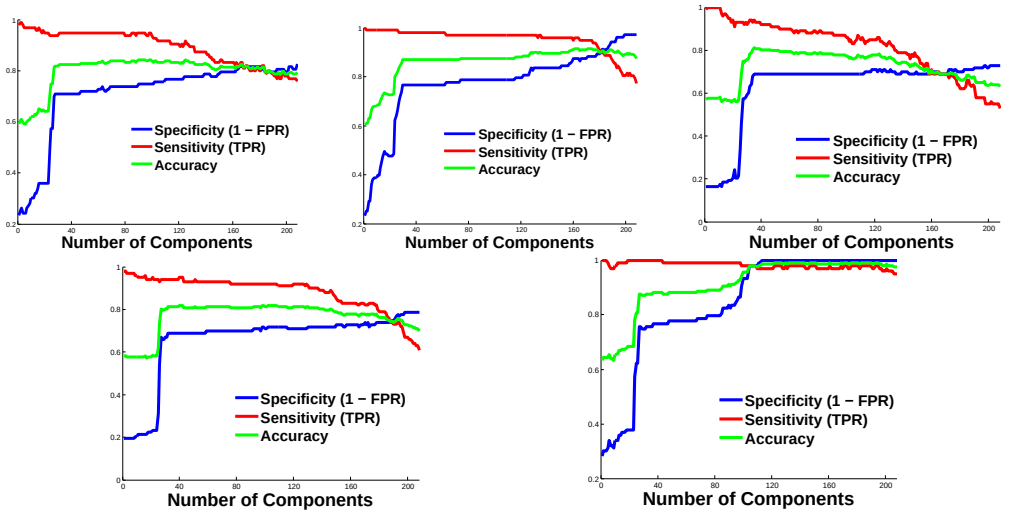

Fig. 3. Detection accuracy plotted against the number of principal components for Cho's mean and variance-based watermarking methods [Cho et al. 2007], Yang's Laplacian coordinates based watermarking [Yang and Ivrissimtzis 2010], Chao's high-capacity steganography [Chao et al. 2009] and Ohbuchi's fequency-based watermarking [Ohbuchi et al. 2002] (from top to bottom and left to right). Here, TPR and FPR indicate true positive rate and false positive rate, respectively.

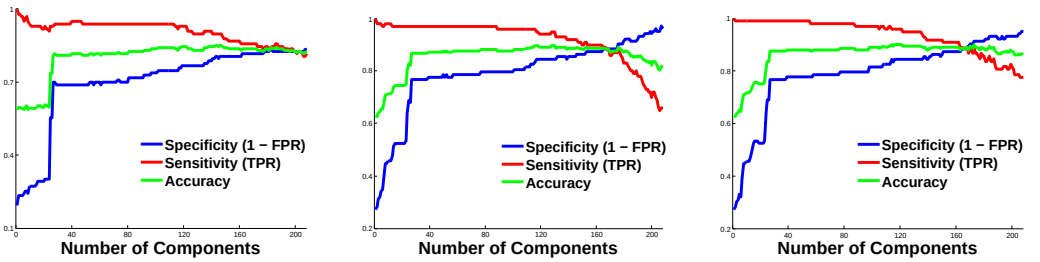

Fig. 4. Detection accuracy plotted against the number of principal components for LSB steganography [Yang et al. 2013] with expected normal distortion of $\epsilon = 1^\circ$, $\epsilon = 2^\circ$ and $\epsilon = 5^\circ$ (from left to right). Here, TPR and FPR indicate true positive rate and false positive rate, respectively.

Fig. 5 shows the average detection rate for the six embedding methods, against the number of principal components retained from the feature vector. The accuracy is approximately 80%. A comparison with Figs. 3, 4 shows that the universal steganalyzer achieves lower detection rates than the specific steganalyzers. This is an expected behaviour; universal steganalysis is a more challenging classification problem since there is a choice of steganographic algorithm for inserting the watermark, and thus, the classification space of marked meshes is more complex.

Furthermore, Fig. 5 shows that the optimal detection rate of the universal steganalyzer is achieved when all components of $\mathbf{F}_i$ are used, meaning not only that each component of $\mathbf{F}_i$ is likely to have played a useful role in the steganalysis process, but also that one may expect to obtain a better universal steganalyzer by increasing the dimension of the feature vector. The latter is a reasonable expectation since a more complex classification space requires a higher dimensional feature vector to describe it. However, we note that such an increase of the dimension of the feature vector is likely to reduce the detection rates of the specific steganalyzers, which work on smaller classification spaces.

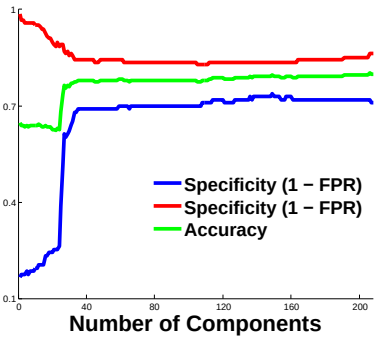

Fig. 5. Average detection rate for all six embedding methods, plotted against the number of retained principal components. Here, TPR and FPR indicate true positive rate and false positive rate, respectively.

5. CONCLUSION

We have presented a steganalytic algorithm for the detection of hidden messages in triangle meshes. The algorithm has been evaluated on six state-of-the-art 3D watermarking/steganographic methods with satisfactory accuracy rates.

We think that the high success rate of the steganalytic algorithm may be partly due to a certain lack of sophistication in the current state-of-the-art of 3D watermarking/steganography and that algorithms with better anti-steganalytic performance should be developed. In the future, we plan to work towards a 3D watermarking algorithm that will be robust against the proposed steganalytic algorithm, or any variants of it that would employ different feature vectors, different calibration methods, or different classifiers. As the main challenge in designing such an undetectable watermarking algorithm, we note our limited understanding of how several components of feature vector, such as the skewness and kurtosis of the graph of the norms of the Laplacian coordinates, are affected by vertex perturbations. Moreover, the high dimensionality of the feature vector means that machine learning algorithms may be able to detect patterns that the designers of the algorithm did not intuitively expect. Thus, some high level guesswork combined with error and trial might be the easiest way forward.

REFERENCES

BOGOMJAKOV, A., GOTSMAN, C., AND ISENBURG, M. 2008. Distortion-free steganography for polygonal meshes. *Computer Graphics Forum (Proceedings of EUROGRAPHICS 2008) 27,* 2, 637–642.

BOLLOBÁS, B. 1998. *Modern graph theory*. Springer Verlag.

CAYRE, F. AND MACQ, B. 2003. Data hiding on 3-d triangle meshes. *IEEE Transactions on Signal Processing 51,* 4, 939–949.

CHAO, M.-W., LIN, C.-H., YU, C.-W., AND LEE, T.-Y. 2009. A high capacity 3D steganography algorithm. *IEEE Transactions on Visualization and Computer Graphics 15,* 2, 274–284.

CHO, J.-W., PROST, R., AND JUNG, H.-Y. 2007. An oblivious watermarking for 3-D polygonal meshes using distribution of vertex norms. *IEEE Transactions on Signal Processing 55,* 1, 142–155.

FARID, H. 2002. Detecting hidden messages using higher-order statistical models. In *Proc. IEEE Int. Conf. Image Process.* Vol. 2. II–905–II–908.

FREEDMAN, D. AND DIACONIS, P. 1981. On the histogram as a density estimator: L_2 theory. *Probability Theory and Related Fields 57,* 4, 453–476.

FRIDRICH, J. J., GOLJAN, M., AND HOGEA, D. 2002. Steganalysis of jpeg images: Breaking the f5 algorithm. In *Information Hiding*. 310–323.

GOLUB, G. AND VAN LOAN, C. 1996. *Matrix computations*. Vol. 3. Johns Hopkins University Press.

GUSKOV, I., SWELDENS, W., AND SCHRÖDER, P. 1999. Multiresolution signal processing for meshes. In *SIGGRAPH'99*. ACM Press, 325–334.

HARMSEN, J. J. AND PEARLMAN, W. A. 2003. Steganalysis based on multiple features formed by statistical moments of wavelet characteristic functions. In *Proceedings of SPIE, Security, Steganography, and Watermarking of Multimedia Contents VI,*. Springer, 131–142.

HIGUCHI, Y. 2001. Combinatorial curvature for planar graphs. *Journal of Graph Theory 38,* 4, 220–229.

HOPPE, H. 1996. Progressive mesh. In *SIGGRAPH'96*. Vol. 96. 99–108.

KARNI, Z. AND GOTSMAN, C. 2000. Spectral compression of mesh geometry. In *SIGGRAPH'00*. ACM Press, 279–286.

KER, A. 2005. Steganalysis of lsb matching in grayscale images. *IEEE Signal Processing Letters 12,* 6, 441–444.

KIM, K., BARNI, M., AND TAN, H. 2010. Roughness-adaptive 3-d watermarking based on masking effect of surface roughness. *IEEE Transactions on Information Forensics and Security 5,* 4, 721–733.

KODOVSKÝ, J. AND FRIDRICH, J. 2009. Calibration revisited. In *Proceedings of the 11th ACM workshop on Multimedia and security*. ACM, New York, NY, USA, 63–74.

KONSTANTINIDES, J., MADEMLIS, A., DARAS, P., MITKAS, P., AND STRINTZIS, M. 2009. Blind robust 3-d mesh watermarking based on oblate spheroidal harmonics. *IEEE Transactions on Multimedia 11,* 1, 23–38.

KRZANOWSKI, W. 2000. *Principles of multivariate analysis*. Oxford University Press.

LIN, H.-Y., LIAO, H.-Y., LU, C.-S., AND LIN, J.-C. 2005. Fragile watermarking for authenticating 3-d polygonal meshes. *IEEE Transactions on Multimedia 7,* 6, 997–1006.

LIU, Y., PRABHAKARAN, B., AND GUO, X. 2008. A robust spectral approach for blind watermarking of manifold surfaces. In *Proceedings of the 10th ACM workshop on Multimedia and security*. ACM, 43–52.

LIU, Y., PRABHAKARAN, B., AND GUO, X. 2010. Dirichlet harmonic shape compression with feature preservation for parameterized surfaces. *Computer Graphics Forum (Proceedings of Pacific Graphics 2010) 29,* 7, 2039–2048.

LIU, Y., PRABHAKARAN, B., AND GUO, X. 2012. Spectral watermarking for parameterized surfaces. *Information Forensics and Security, IEEE Transactions on 7,* 5, 1459 – 1471.

LOUNSBERY, M., DEROSE, T. D., AND WARREN, J. 1997. Multiresolution analysis for surfaces of arbitrary topological type. *ACM Trans. Graph. 16,* 1, 34–73.

LUO, M. AND BORS, A. 2011. Surface preserving robust watermarking of 3-d shapes. *IEEE Transactions on Image Processing 20,* 10, 2813–2826.

OHBUCHI, R., MUKAIYAMA, A., AND TAKAHASHI, S. 2002. A frequency-domain approach to watermarking 3d shapes. *Computer Graphics Forum 21,* 3, 373–382.

PEVNÝ, T. AND FRIDRICH, J. 2007. Merging Markov and DCT features for multi-class JPEG steganalysis. In *Proceedings SPIE, Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents IX, San Jose, CA, January 29–February 1*, E. Delp and P. Wong, Eds. Vol. 6505. 3–14.

PRAUN, E., HOPPE, H., AND FINKELSTEIN, A. 1999. Robust mesh watermarking. In *SIGGRAPH'99*. ACM Press, 49–56.

SHILANE, P., MIN, P., KAZHDAN, M., AND FUNKHOUSER, T. 2004. The princeton shape benchmark. In *Proceedings of Shape Modeling Applications*. IEEE, 167–178.

TU, S.-C., TAI, W.-K., ISENBURG, M., AND CHANG, C.-C. 2010. An improved data hiding approach for polygon meshes. *Vis. Comput. 26*, 1177–1181.

UCCHEDDU, F., CORSINI, M., AND BARNI, M. 2004. Wavelet-based blind watermarking of 3d models. In *Workshop on multimedia and security*. ACM Press, Magdeburg, Germany, 143–154.

VALLET, B. AND LÉVY, B. 2008. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum 27,* 2, 251–260.

WANG, C.-M. AND CHENG, Y.-M. 2005. An efficient information hiding algorithm for polygon models. *Computer Graphics Forum 24*, 591–600.

WANG, K., LAVOUÁ, G., DENIS, F., AND BASKURT, A. 2008. A comprehensive survey on three-dimensional mesh watermarking. *IEEE Transactions on Multimedia 10,* 8, 1513–1527.

WANG, Y. AND MOULIN, P. 2007. Optimized feature extraction for learning-based image steganalysis. *IEEE Transactions on Information Forensics and Security 2,* 1, 31–45.

WESTFELD, A. AND PFITZMANN, A. 2001. High capacity despite better steganalysis (f5–a steganographic algorithm). In *Proceeding of 4th International Workshop on Information Hiding. Lecture Notes in Computer Science*. Vol. 2137. 289–302.

XUAN, G., SHI, Y., GAO, J., ZOU, D., YANG, C., ZHANG, Z., CHAI, P., CHEN, C., AND CHEN, W. 2005. Steganalysis based on multiple features formed by statistical moments of wavelet characteristic functions. In *Proceedings of Information Hiding Workshop*. Springer, 262–277.

YANG, Y. AND IVRISSIMTZIS, I. 2010. Polygonal mesh watermarking using laplacian coordinates. *Computer Graphics Forum (Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing) 29,* 5, 1585–1593.

YANG, Y., PEYERIMHOFF, N., AND IVRISSIMTZIS, I. 2013. Linear correlations between spatial and normal noise in triangle meshes. *IEEE Transactions on Visualization and Computer Graphics 19,* 1, 45 – 55.

YEO, B. L. AND YEUNG, M. M. 1999. Watermarking 3d objects for verification. *Computer Graphics and Applications 19,* 1, 36–45.

YIN, K., PAN, Z., SHI, J., AND ZHANG, D. 2001. Robust mesh watermarking based on multiresolution processing. *Computers & Graphics 25,* 3, 409–420.