Feature Preserving Mesh Denoising Based on Graph Spectral Processing

Gerasimos Arvanitis, Aris S. Lalos, *Member, IEEE,* Konstantinos Moustakas *Senior Member, IEEE* and Nikos Fakotakis

Abstract—The increasing interest for reliable generation of large scale scenes and objects has facilitated several real-time applications. Although the resolution of the new generation geometry scanners are constantly improving, the output models, are inevitably noisy, requiring sophisticated approaches that remove noise while preserving sharp features. Moreover, we no longer deal exclusively with individual shapes, but with entire scenes resulting in a sequence of 3D surfaces that are affected by noise with different characteristics due to variable environmental factors (e.g., lighting conditions, orientation of the scanning device). In this work, we introduce a novel coarse-to-fine graph spectral processing approach that exploits the fact that the sharp features reside in a low dimensional structure hidden in the noisy 3D dataset. In the coarse step, the mesh is processed in parts, using a model based Bayesian learning method that identifies the noise level in each part and the subspace where the features lie. In the feature-aware fine step, we iteratively smooth face normals and vertices, while preserving geometric features. Extensive evaluation studies carried out under a broad set of complex noise patterns verify the superiority of our approach as compared to the state-of-the-art schemes, in terms of reconstruction quality and computational complexity.

Index Terms—Spectral Smoothing, Orthogonal Iteration, Spectral Denoising Filtering, Feature Extraction, Level Noise Estimation

1 Introduction

In recent years, an increasing demand for acquiring, processing and streaming 3D models representing real world objects is observed. These models usually come as very dense and noisy meshes that stand in need of solutions capable of distinguishing noise from local geometric features. Similarly, the surfaces extracted from volumetric data (e.g., MRI and CT devices) contain topological and geometric noise that needs to be removed before further processing. Although there are several methods in the literature for performing feature preserving mesh denoising, there are still challenges that need to be addressed, especially if we take into account that the scale of acquired data in real time scanning operations is growing very fast. We no longer deal exclusively with individual shapes, but with entire scenes, possibly at the scale of entire cities with many objects defined as structured shapes (e.g., aerial scanning [1], [2], slam scanning [3], underwater scanning [4], scalable multi-object scanning [5], large-scale terrestrial scanning [6], large statues scanning [7]) resulting in a sequence of 3D surfaces that are affected by noise with different characteristics.

Throughout the years, numerous approaches have been proposed improving more or less some of the key feature preserving denoising characteristics [8], [9], [10]. Though, recovering the structure of large scenes is still without a doubt a stimulating challenge. To the best of our knowledge, none of the approaches is capable of identifying accurately (i) the subspace where the features of each surface part lie, (ii) the noise characteristics on the different parts of the surface. Those requirements become more essential in scenarios where large 3D models are scanned in parts, generating a sequence of 3D surfaces that arrive sequentially in time and require fast and accurate feature preserving surface denoising. In those cases, the denoising method should be also capable of mitigating dynamic noise with varying characteristics per surface part. This effect is attributed to the fact that several environmental factors that affect the surface noise pattern of each part may also change in time, e.g., variable lighting conditions and orientation of a hand held scanner, standing in need of solutions that are capable of identifying the noise and the geometry characteristics. To deal efficiently with the aforementioned challenges we suggest applying a coarse-to-fine denoising approach.

We initially decompose the original mesh in a number of registered 3D patches. In the coarse step, each part is treated individually. The coarse denoising approach is capable of identifying both i) the spectral subspace size where the features lie and ii) the level of noise, using a model based Bayesian learning scheme. In order to reduce the required complexity for evaluating the spectral subspace for each part of the surface, we adopt a subspace tracking approach that exploits potential coherences between the spectral subspaces of adjacent parts. In the fine step we eliminate the small amount of remaining noise. To that end we apply a feature aware guided normal filtering (GNF) method. More specifically, the application of the coarse step allows the accurate identification of the geometric features, which are then used to accelerate the execution time of the conventional GNF approaches. To summarize, the main contributions of this work are:

- We provide a novel coarse denoising step that filters out a significant amount of noise without affecting geomet-

- *G. Arvanitis, A. S. Lalos, K. Moustakas and N. Fakotakis are with the Department of Electrical and Computer Engineering, University of Patras, Rio 26504 Patras, Greece. E-mail: arvanitis@ece.upatras.gr*

ric features under non stationary noise scenarios. This approach is essential for denoising large scale meshes representing complex 3D scenes, that are reconstructed in parts, affected by noise with different characteristics. To achieve that, we developed: i) An approach that automatically identifies the subspace size where sharp and small-scaled geometric features lie. Then, the subspace of interest is evaluated using orthogonal iterations, an efficient subspace tracking approach that exploits the spatial coherences between different parts of the surface, and results in very fast execution times. ii) A method which is capable of identifying the pattern of noise, using a model based Bayesian learning scheme. The identified subspace and the evaluated noise patterns are then used to mitigate variable noise patterns that are added in different surface patches that are processed individually.

- The use of the coarse step allow us to accurately identify geometric features, which are then used by the proposed feature aware GNF approach that achieves faster execution times as compared to the conventional GNF approach. More specifically, on contrary to GNF scheme, the iterative process for searching the ideal patches is not applied to the whole mesh but only in these vertices that have been classified as features.
- We prove that the fine step can be also considered as a graph spectral method and we provide extensive simulations carried out using several scanned and CAD models under a broad set of noise configurations showing that: i) The coarse scheme can be also adopted by any SoA denoising approach accelerating its execution time and allowing them to efficiently mitigate noise patterns that vary between different parts of the surface. ii) The combination of the proposed coarse and the fine graph spectral processing steps offers both enhanced robustness and low complexity requirements when dealing with anisotropic and non stationary noise patterns.

An extensive performance assessment using a large collection of different 3D models, including both CAD and scanned models, under a broad set of noise patterns, including: i) non isotropic Gaussian noise [11] ii) real scan noise introduced by a variety of 3D scanning devices [9] iii) staircase effects caused by MRI and CT devices [12] and iv) complex noise patterns that may vary significantly over the different parts of the scanned surface, clearly shows the benefits of our coarse-to-fine method as compared to the state-of-the-art (SoA) approaches in terms of reconstruction accuracy and complexity.

The paper is organized as follows: Section 2 reviews prior art in detail. Section 3 presents the preliminaries necessary for our method. Section 4 presents our contributions and demonstrates how these techniques are capable to improve the final execution time such as the appearance of the final results. Section 5 presents the qualitative, quantitative and comparative results of our method, and discusses the advantages and limitations of the proposed method, while Section 6 draws conclusions and identifies future directions.

2 Related Work

Mesh denoising aims to decrease or eliminate the noise, preserving all useful information such as geometric details of different scale. It is a vital pre-processing tool for improving imperfect meshes obtained by scanning devices and digitization processes. During the last years several methods have been proposed, yet, despite the significant improvement, the main need for a robust and fast algorithm that can manage large meshes with persistent noise still remains a challenge. The existing approaches can be classified in two core groups that treat differently noise and salient features, known as **isotropic** and **anisotropic** respectively.

Isotropic Methods

The most well known isotropic method is the Laplacian smoothing [13] that has the mesh shrinkage as a major disadvantage. In order to solve this limitation, Taubin [14] proposed a two-stage approach where two sequential filters are applied iteratively: the first one performs Laplacian smoothing while the second is used to prevent shrinkage. The accurate reconstruction depends on the shrinkage and inflation parameters, which are selected empirically and in many cases suffer from stability issues. Desbrun et al. extended this approach to irregular meshes by using the mean curvature flow analogy to rescale the mesh preventing shrinkage [15]. Several surveys that cover basic definitions and applications of the graph spectral methods have been introduced by Gotsman [16], Levy [17], Sorkine [18] and more recently by Zhang et al. [19]. All these surveys classify the spectral methods according to several criteria related to the employed operators, the application domains and the dimensionality of the spectral embeddings used. Graph spectral processing of 3D meshes relies on the singular/eigenvectors and/or eigenspace projections derived from appropriately defined mesh operators [20]. A summary of the mesh filtering approaches that can be efficiently carried out in the spatial domain using convolution approaches is given by Taubin in [21]. Computing the truncated singular value decomposition, can be extremely memory-demanding and time-consuming. To overcome this limitations, subspace tracking algorithms have been proposed as fast alternatives relying on the execution of iterative schemes for evaluating the desired eigenvectors per incoming block of floating point data corresponding in our case, to different surface patches [22]. The Orthogonal iterations is the most widely adopted subspace tracking method, due to the fact that it results in very fast solutions when the initial input subspace is close to the subspace of interest [23].

Anisotropic methods

Most feature preserving mesh denoising approaches locally adjust vertex positions while respecting the underlying features and can be classified in four major categories.

Anisotropic Geometric Diffusion: The first one is based on anisotropic geometric diffusion [24], [25]. In [26] the new vertex position is estimated by a nonlinear weighted mean of local neighborhood vertices. The method is simple and can be easily reproduced but its computational complexity is quite high. In [27] sharp features are removed while preserving small-scale features of the geometry of the original mesh.

The main limitation of these approaches is the distortion of the geometric features, that is attributed to the fact that all the vertices are handled in the same manner.

Bilateral Filtering of Vertices and Normals: These iterative methods use normals to estimate the weights of the filters. However, the accurate reconstruction of the geometric features strongly depends on the effect of noise in the weight estimation [28], [29]. The authors in [30] present a robust approach for denoising triangular surface meshes using a combination of bilateral filtering, feature detection, surface fitting and projection techniques. An extended version of bilateral filtering is based on normal filtering and vertex position update [8], [31], [32], [33], [34]. These methods are described as two-stage iterative approaches. Firstly, the face normals are filtered and then the vertex positions are updated based on the filtered face normals. The authors in [35] propose a two stage processing method for mesh denoising. Initially, part of the noise is removed so that the features of the mesh are preserved. A fine denoising step is applied on the points that are classified as features. Although the aforementioned methods preserve most of the sharp features, they fail to preserve features of different scale, such as medium or small scale features.

Sparse Optimization and Tensor Voting: The third category includes surface reconstruction and decimation approaches that regularize both vertices and normals [36]. These type of methods minimize the energy of both vertex position and normal error. The authors in [10], propose an l_0 minimization approach that provides accurate surface reconstruction in Gaussian noise cases, however its performance is significantly deteriorated when considering complex noise patterns (e.g., anisotropic noise, real scan noise). Despite the good surface reconstruction results they cannot always preserve sharp features well [37] while in other cases [38] the execution time is high. The authors in [39] present a cascaded denoising framework composed of a multiscale tensor voting step, a vertex clustering step for detecting sharp features and a piecewise fitting step for preserving the identified features, dealing efficiently with challenging situations, such as irregular surface sampling. In [40] a vertex classification step of noisy meshes takes place before applying the denoising process. In [41] the usage of consistent sub-neighborhoods is used for vertex classification. The method of [42] is an iterative approach combining pre-filtering, feature detection, and l_1-based feature recovery. The main limitation of the presented approaches is the increased required computational complexity and the fact that in many cases irregularly highlight the identified sharp features.

Data-driven Approaches: The authors in [9] suggest a method for mesh denoising which uses training sets of noisy objects, scanned by the same devices, and extract information that are used for denoising meshes with similar noise type. While many geometric features are reconstructed adequately, geometric details are limited to those not included in the training set.

3 Preliminaries on Graph Spectral Processing

In this section we present the basic definitions and well known preliminaries related to graph spectral processing.

3.1 Polygon Models

In this work we focus on triangle meshes, which are the most common polygon models. Triangle meshes $\mathbf{M}$ with k vertices can be represented by two different sets $\mathbf{M} = (\mathcal{V}, \mathcal{F})$ corresponding to the vertices $\mathcal{V}$ and the indexed faces $\mathcal{F}$ of the mesh. Each vertex can be represented as a point $v_i = (x_i, y_i, z_i)\ \forall\ i = 1, k$. In this case we create a vector of vertices $\mathbf{v} = [\mathbf{x},\ \mathbf{y},\ \mathbf{z}]$ in a 3D coordinate space such as $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \Re^{k\times 1}$ and $\mathbf{v} \in \Re^{k\times 3}$. This means that we have a set of k points such that $\mathcal{V} = \{v_1, v_2, \ldots, v_k\}$. Additionally, each face is defined by a set of 3 indices to vertices $f_i = [v_{i1},\ v_{i2},\ v_{i3}]\ \forall\ i = 1, k_f$ where $k_f > k$, so we have k_f faces $\mathcal{F} = \{f_1, f_2, \ldots, f_{k_f}\}$. Each face constitutes a triangle, the simplest surface unit, that can be represented by its centroid point $\mathbf{c}_i$ and its outward unit normal $\mathbf{n}_{\mathbf{c}i}\ \forall\ i = 1, k_f$. The vertices of a noisy mesh $\tilde{\mathbf{M}} = (\tilde{\mathcal{V}}, \mathcal{F})$ satisfy the following identity:

$$\tilde{\mathbf{v}}_i = \mathbf{v}_i + \tilde{\mathbf{z}}_i \quad \forall\ i = 1, \ldots, k \tag{1}$$

where $\mathbf{v}_i$ are the noise free vertices and $\tilde{\mathbf{z}}_i$ represents a 1×3 noise vector.

3.2 Spectrum of a Graph

The spectrum of a graph is defined in terms of the eigenvalues and eigenvectors of the Laplacian matrix. The Laplacian matrix of a graph $\mathcal{G}$, assuming that a set of edges $\mathcal{E}$ can be directly derived from $\mathcal{V}$ and $\mathcal{F}$, which correspond to the connectivity information, is defined as:

$$\mathbf{L} = \mathbf{D} - \mathbf{C} \tag{2}$$

where $\mathbf{C} \in \Re^{k\times k}$ is the connectivity matrix of the mesh with elements:

$$\mathbf{C}_{ij} = \begin{cases} e^{-\frac{\|v_i - v_j\|_2^2}{2\alpha^2}} & if\ i, j \in \mathcal{E} \\ 0 & otherwise \end{cases} \tag{3}$$

$\mathbf{D} = diag\{d_1, d_2, \ldots, d_k\}$ is a diagonal matrix with $d_i = \sum_{j=1}^{k} \mathbf{C}_{ij}$ and α is the variance that sets a threshold to edge existence. The weighted adjacency matrix is ideal for emphasizing the coherence between Laplacian matrices of different submeshes by providing geometric information; on the contrary, the binary provides only connectivity information. The normalized form of the Laplacian is a non-negative matrix given by $\mathcal{L} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ that can be diagonalized as:

$$\mathcal{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T \tag{4}$$

where $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k]$ is an orthonormal matrix with the eigenvectors and $\mathbf{\Lambda} = diag\{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ is a diagonal matrix with the corresponding eigenvalues. The eigenvectors and eigenvalues of the Laplacian matrix $\mathcal{L}$ provide a spectral interpretation of the graph signals. The Graph Fourier Transform (GFT) of a vector $\mathbf{v}$ is defined as its projection onto the eigenvalues of the graph $\hat{\mathbf{v}} = \mathbf{U}^T\mathbf{v}$, and the inverse GFT is given by $\mathbf{v} = \mathbf{U}\hat{\mathbf{v}}$. In the following part of our work, we will apply GFT for both filtering out and attenuating higher frequencies of vertices and face normals.

3.3 Cutting off Higher Frequencies & Attenuating Frequencies

At any noisy mesh $\tilde{\mathbf{M}}$, the vertices and face normals that correspond to the sharp features 'lie' in a low dimensional subspace of size m, while noise usually has a flat spectrum that is easily identifiable at the higher $k - m$ frequencies. Therefore, one way to mitigate the noise effects is to perform

spectral smoothing by filtering out the higher frequencies. In mathematical terms, the subspace decomposition of the normalized Laplacian matrix defined by the vertices of the mesh can be re-written as:

$$\mathcal{L} = \begin{bmatrix} \mathbf{U}_{k-m} & \mathbf{U}_m \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda}_{k-m} & 0 \\ 0 & \mathbf{\Lambda}_m \end{bmatrix} \begin{bmatrix} \mathbf{U}_{k-m} & \mathbf{U}_m \end{bmatrix}^T \quad (5)$$

where $\mathbf{U}_m = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m]$. Then a smoothed version of the noisy vertices can be generated by performing low pass spectral filtering of the Cartesian coordinates $\mathbf{v} = \mathbf{U}_m \mathbf{U}_m^T \tilde{\mathbf{v}}$.

Normals $\mathbf{n}$ can be considered as a signal defined on an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the signal value at each node equals the corresponding normal vector coordinates. In order to average the normals that belong to a common smooth region, one could potentially perform a graph filtering operation by substituting the normal at each node by a weighted average of the normal values of its neighbors. In matrix form, this operation may be written as:

$$\begin{aligned} \mathbf{n}_k &= \mathbf{D}^{-1}\mathbf{C}\mathbf{n} \\ &= \mathbf{D}^{-1/2}\mathbf{D}^{-1/2}\mathbf{C}\mathbf{D}^{-1/2}\mathbf{D}^{1/2}\mathbf{n} \Longrightarrow \\ \mathbf{D}^{1/2}\mathbf{n}_k &= (\mathbf{I} - \mathcal{L})\,\mathbf{D}^{1/2}\mathbf{n} \end{aligned} \quad (6)$$

By denoting the normalized versions $\hat{\mathbf{n}}_k = \mathbf{D}^{1/2}\mathbf{n}_k$ and $\hat{\mathbf{n}} = \mathbf{D}^{1/2}\mathbf{n}$ Eq. (6) can be re-written as:

$$\hat{\mathbf{n}}_k = \underbrace{\mathbf{U}}_{IGFT} \quad \underbrace{(\mathbf{I} - \mathbf{\Lambda})}_{\substack{Spectral \\ Response}} \quad \underbrace{\mathbf{U}^T}_{GFT} \hat{\mathbf{n}} \quad (7)$$

Eq. (7) shows that any averaging filter on the normals is a frequency selective graph transform with a spectral response $h(\lambda_i) = 1 - \lambda_i$ which corresponds to a linear decay and $\mathbf{I}$ represents the identity matrix. An averaging filter tries to preserve the low frequency components and attenuate the high frequency ones.

4 Overview of our Method

The coarse denoising approach suggests decomposing a mesh into smaller submeshes using a fast graph partitioning method, and reducing the amount of artifacts and noise within each patch by cutting off the higher frequencies. To fine tune the smoothing operation and preserve corners and sharp edges, we propose a novel noise level estimation scheme which dynamically evaluates the subspace size where the features 'lie' using a model based Bayesian learning approach. One of the major limitations of graph spectral processing (GSP) is the required computational complexity, since it involves the evaluation of the singular vectors of a matrix (spectral coefficients) with size equal to the number of vertices in the patch. To overcome this issue we propose a fast and efficient way for evaluating the GSP coefficients of the different patches by using an approach that is based on subspace projections.

The proposed coarse denoising approach allows us to accurately cluster the face normals into features (edges, corners) and flat areas and then apply a fine denoising operation based on a graph spectral filter that averages normals that belong to a common smooth region, while preserving sharp changes of normals that are indicated as features. The processed normals are then used to update the corresponding vertex positions. In addition, this process significantly accelerates the convergence of the following process. Finally, for a given feature face we search among a set of candidate patches that contain the feature face and pick the one with the most consistent normal directions (ideal patch selection). The average normal of the chosen patch is then used as the normal of the selected face. Such guidance provides a robust estimation for the true normal in the presence of noise, enabling our denoising algorithm to handle highly noisy meshes. The proposed two stage graph spectral processing architecture (TSGSP) is illustrated in Fig. 1, while in the rest part of this section we provide a detailed description of the different parts that increase the robustness of the graph spectral based denoising.

4.1 Fast Sequential Processing of 3D Patches Using Orthogonal Iterations

Calculating the graph Laplacian eigenvalues of the mesh geometry can become restrictive as the density of the models increases. To overcome this limitation, several approaches suggest processing large meshes into parts [43], [44]. In the proposed scheme, we initially separate the whole mesh to submeshes and then we replace SVD with an approximate iterative method that is much more computational efficient.

Processing of Registered 3D Patches

The noisy 3D mesh is partitioned into s submeshes that are reconstructed individually, using the METIS method described in [44], [45]. Each submesh $\tilde{\mathbf{M}}_i$ consists of k_i points, where $\sum_{i=1}^{s} k_i = k$. The Cartesian coordinates of the k_i points included in the i^{th} submesh are represented as a vector $\mathbf{v}_i \in \Re^{k_i \times 3}$. To avoid the problem of edge effects we process overlapped submeshes and estimate the weighted average of vertex positions for the overlapping nodes. The weights that are assigned to each point are proportional to the degree of the node (e.g., number of neighbors) in the corresponding submesh. For example, in Fig. 2 we present the weights assigned to a point (highlighted in red) that participates in three overlapped submeshes.

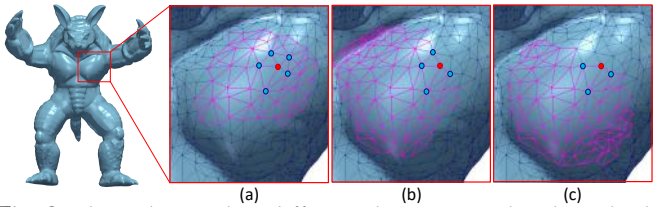

Fig. 2: The red point has different degree in each submesh, the corresponding weights a are: (a) $a = 5$, (b) $a = 4$, (c) $a = 3$.

The ideal number of submeshes depends on the total number of points of a mesh, however, an extensive evaluation study shows that submeshes with 500-800 points provide the best results in terms of both execution time and visual error.

Fast Spectral Smoothing Using Orthogonal Iterations

A direct application of spectral method to the vertices of each submesh $\tilde{\mathbf{M}}_i$ individually would result in a complexity of the order $\mathcal{O}(\sum_{i=1}^{s} k_i^3)$. To minimize this complexity, we suggest exploiting the coherence between the spectral components of the different submeshes using orthogonal iterations (OI).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

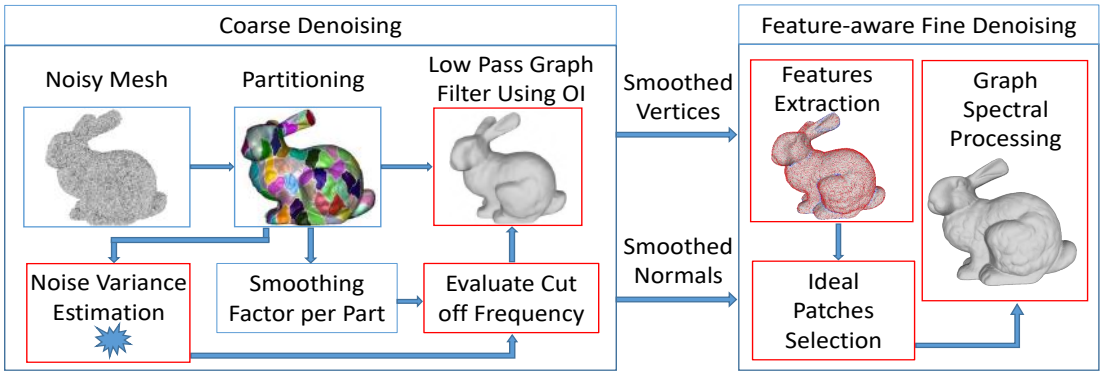

Fig. 1: Sequential diagram of the procedure. The proposed framework is separated into a coarse denoising stage and a fine denoising stage. The proposed methods are represented with red boxes.

Proposition 1. We estimate the $\mathbf{U}_m\,[i]\ \forall\ i=1,s$ according to the next equation:

$$\mathbf{U}_m\,[i] = Orthonorm\left\{\mathbf{R}^{\beta}\,[i]\,\mathbf{U}_m\,[i-1]\right\} \tag{8}$$

where $\mathbf{R}[i]=(\mathbf{L}[i]+\delta\mathbf{I})^{-1}$, δ is a small positive scalar value that ensures positive definiteness of $\mathbf{R}[i]$ and $\mathbf{L}[i]$ is the graph Laplacian of the vertices of the submesh $\tilde{\mathbf{M}}_i$. Depending on the choice of $\mathbf{R}^{\beta}\,[i]$, we obtain alternative iterative algorithms with different convergence properties. The convergence rate of Eq. (8) depends on $|\lambda_{m+1}/\ \lambda_m|^{\beta}$ where λ_{m+1} is the $(m+1)$-st largest eigenvalue of $\mathbf{R}^{\beta}$ [46].

The projection of the raw geometry data in the domain defined by the graph Laplacian eigenvalues $\mathbf{U}_m$ allow efficient representations that can be exploited by a number of applications such as, in our case, denoising. However, in order to avoid smoothing any potential feature point we need to accurately identify the size m of the subspace where the edges and corner points lie. The following subsection provides a dynamic scheme that can successfully identifies the optimal subspace size m.

Dynamic Identification of $\mathbf{m}$ *and* $\boldsymbol{\sigma}_{\mathbf{z}}^{\mathbf{2}}$

We start by assuming that the observed noisy vertices can be separated into two quantities, namely the clean vertices and the external noise, as follows:

$$\mathbf{v}_j = \mathbf{v}_{\mathbf{c}\,j} + \tilde{\mathbf{z}}_j \tag{9}$$

$$= \mathbf{U}_{\mathbf{m}_j^*}\hat{\mathbf{v}}_j + \tilde{\mathbf{z}}_j\ \ j=\{x,y,z\} \tag{10}$$

where $\mathbf{m}_j^*$ represents the optimal length of features subspace size, $\hat{\mathbf{v}}_j = \mathbf{U}_{\mathbf{m}_j^*}^T\mathbf{v}_{\mathbf{c}\,j}$ is projected clean vertices and $\tilde{\mathbf{z}}_j$ is the vector of noise coordinate $j=\{x,y,z\}$. If we assume that $\mathbf{m}_j^*$ is known, then the $\hat{\mathbf{v}}_j$ can be modeled as a parametrized multivariate Gaussian distribution:

$$p(\hat{\mathbf{v}}_j) \sim \mathcal{N}(0,\boldsymbol{\Pi}_{0_j}),\ \ \boldsymbol{\Pi}_{0_j} = \boldsymbol{\gamma}_j\boldsymbol{\Sigma}_j\ \ j=\{x,y,z\} \tag{11}$$

where $\boldsymbol{\gamma}_j$ is a scalar parameter and $\boldsymbol{\Sigma}_j \in \Re^{\mathbf{m}_j^*\times\mathbf{m}_j^*}$ is a positive definite matrix. By using the Bayes rule and assuming that the noise vector $\tilde{\mathbf{z}}_j$ has distribution with zero mean value then $\tilde{\mathbf{z}}_j \sim \mathcal{N}(0,\boldsymbol{\sigma}_{\mathbf{z}_j}\mathbf{I})$. However, in the general case, the optimal feature subspace size value of $\mathbf{m}_j^*$ is not known. A non optimal choice of $\mathbf{m}_j$, would lead either to the exclusion of some feature components or the inclusion of noisy components. In this case, the l_2 norm of the vector $\|\mathbf{v}_j-\mathbf{U}_{\mathbf{m}_j}\hat{\mathbf{v}}_j\|/k$ will be a value outside the region $[\boldsymbol{\sigma}_{\mathbf{z}_j}^{\mathbf{2}}-\epsilon,\boldsymbol{\sigma}_{\mathbf{z}_j}^{\mathbf{2}}+\epsilon]$, where ϵ is a negligible threshold. Motivated by this observation we propose an iterative scheme that either increases or decrease the feature subspace size, so that the aforementioned termination condition is satisfied. For each choice of m the following Bayesian learning scheme is executed in order to identify the projected clean coefficients and the level of noise. In particular, the following Proposition 2 presents the Bayesian learning steps for the identification the of projected clean coefficients per submesh $i=1,s$ and per coordinate $j=\{x,y,z\}$.

Proposition 2. Assuming that the posterior density of $\hat{\mathbf{v}}_{ij}$, is also Gaussian $p(\hat{\mathbf{v}}_{ij}|\mathbf{v}_{ij};\boldsymbol{\sigma}_{\mathbf{z}_{ij}},\boldsymbol{\gamma}_{ij}\boldsymbol{\Sigma}_{ij}) \sim \mathcal{N}(\mathbf{H}_{ij},\boldsymbol{\Pi}_{ij})$ we can estimate its mean and covariance matrix as follows:

$$\mathbf{H}_{ij} = \boldsymbol{\Pi}_{0ij}\mathbf{U}_{\mathbf{m}_{ij}}^T\left(\mathbf{U}_{\mathbf{m}_{ij}}\boldsymbol{\Pi}_{0ij}\mathbf{U}_{\mathbf{m}_{ij}}^T+\boldsymbol{\sigma}_{\mathbf{z}_{ij}}\mathbf{I}_{k_i}\right)^{-1}\mathbf{v}_{ij} \tag{12}$$

$$\boldsymbol{\Pi}_{ij} = \boldsymbol{\Pi}_{0ij}-\boldsymbol{\Pi}_{0ij}\mathbf{U}_{\mathbf{m}_{ij}}^T\left(\mathbf{U}_{\mathbf{m}_{ij}}\boldsymbol{\Pi}_{0ij}\mathbf{U}_{\mathbf{m}_{ij}}^T+\boldsymbol{\sigma}_{\mathbf{z}_{ij}}\mathbf{I}_{k_i}\right)^{-1}\mathbf{U}_{\mathbf{m}_{ij}}\boldsymbol{\Pi}_{0ij} \tag{13}$$

For finding the parameters $\boldsymbol{\sigma}_{\mathbf{z}_{ij}},\boldsymbol{\gamma}_{ij},\boldsymbol{\Sigma}_{ij}$ we employ the expectation-maximization (EM) algorithm to maximize the $p(\mathbf{v}_{ij};\boldsymbol{\sigma}_{\mathbf{z}_{ij}},\boldsymbol{\gamma}_{ij}\boldsymbol{\Sigma}_{ij})$ per coordinate, meaning that $j=\{x,y,z\}$. For the estimation of the model parameters that maximize the aforementioned likelihood we use iteratively the following learning rules:

$$\boldsymbol{\sigma}_{\mathbf{z}_{ij}} = \frac{\|\mathbf{v}_{ij}-\mathbf{U}_{\mathbf{m}_{ij}}\hat{\mathbf{v}}_{ij}\|_2^2+\boldsymbol{\sigma}_{\mathbf{z}_{ij}}[\mathbf{m}_{ij}-Tr(\boldsymbol{\Pi}_{ij}\boldsymbol{\Pi}_{0ij}^{-1})]}{k_i} \tag{14}$$

$$\boldsymbol{\gamma}_{ij} = \frac{Tr(\boldsymbol{\Sigma}_{ij}^{-1}(\boldsymbol{\Pi}_{ij}+\mathbf{H}_{ij}\mathbf{H}_{ij}^T))}{\mathbf{m}_{ij}} \tag{15}$$

$$\boldsymbol{\Sigma}_{ij} = \frac{\boldsymbol{\Pi}_{ij}+\mathbf{H}_{ij}\mathbf{H}_{ij}^T}{\boldsymbol{\gamma}_{ij}} \tag{16}$$

The performance of the proposed algorithm can be further improved by constraining the matrix $\boldsymbol{\Sigma}_{ij}$ in order to have a Toeplitz symmetric structure with elements $(\boldsymbol{\Sigma}_{ij})_{(q,w)} = \mathbf{r}_{ij}^{|q-w|},\ \forall\ q,w\in[1,\cdots,\mathbf{m}_{ij}]$. This form is equivalent to modeling the elements in the non-zero block as a first order auto-regressive process, where $\boldsymbol{r}_{ij}$ can be estimated by:

$$\boldsymbol{r}_{ij} = sign(q1/q0)min\{|q1/q0|,0.99\} \tag{17}$$

where $q0$ is the average of the elements along the main diagonal, $q1$ is the average of elements along the main subdiagonal of $\boldsymbol{\Sigma}_{ij}$ and the value 0.99 is a bound selected by the user. The proposed scheme suggests executing the aforementioned methods iteratively, initializing the $\mathbf{m}_{ij}$ and $\boldsymbol{\sigma}_{\mathbf{z}_{ij}}^{\mathbf{2}}$

to any arbitrary value, until the following stopping criterion is satisfied $\boldsymbol{\sigma}^2_{\mathbf{z}_{ij}} - \epsilon < \|\mathbf{v}_{ij} - \mathbf{U}_{\mathbf{m}_{ij}}\hat{\mathbf{v}}_{ij}\|_2^2/k_i < \boldsymbol{\sigma}^2_{\mathbf{z}_{ij}} + \epsilon$. In each iteration t, if $\|\mathbf{v}_{ij} - \mathbf{U}_{\mathbf{m}_{ij}}\hat{\mathbf{v}}_{ij}\|_2^2/k_i < \boldsymbol{\sigma}^2_{\mathbf{z}_{ij}} - \epsilon$ then the length of $\mathbf{m}_{ij}[l] = \mathbf{m}_{ij}[l-1] - 1$ is decreased by one, otherwise the length of $\mathbf{m}_{ij}[l] = \mathbf{m}_{ij}[l-1] + 1$ is increased by one. For the initialization, arbitrary values of $\mathbf{m}_{ij}[0]$ and $\boldsymbol{\sigma}^2_{\mathbf{z}_{ij}}[0]$ could be used. However, to avoid exhaustive search, we suggest an optimal initialization strategy presented in the following subsection. In Algorithm 1 we briefly describe the steps of this iterative procedure.

Algorithm 1: Identification of the optimal Subspace Size $\mathbf{m}^*_{ij}$ per submesh and per coordinate, using Model based Bayesian Learning (MBL).

```
Initialize the values of m_ij[0] and σ²_{z_ij}[0], according to Eqs. (18) and (22) ;
for i = 1, s do
    for j ∈ {x, y, z} do
        while |‖v_ij − U_{m_ij} v̂_ij‖₂²/k_i − σ²_{z_ij}| < ε do
            for g = 1, b do
                Estimate the non zero values of H_ij via Eq. (12) ;
                Estimate the corresponding variances Π_ij via Eq. (13) ;
                Update the values of σ_{z_ij}, γ_ij, Σ_ij, r_ij according to Eqs. (14) - (17) ;
            end
            if ‖v_ij − U_{m_ij} v̂_ij‖₂²/k_i > σ²_{z_ij} + ε then
                m_ij[l] = m_ij[l − 1] + 1 ;
            else
                vm_ij[l] = m_ij[l − 1] − 1 ;
                l = l + 1 ;
            end
        end
    end
end
```

Initialization of Parameters $\mathbf{m}$ *and* $\boldsymbol{\sigma}^2_{\mathbf{z}}$

The convergence speed of the aforementioned scheme strongly depends on the initial value of $\mathbf{m}_{ij}[0]$. An extensive simulation study using different models with geometric features of different scale, showed that the subspace size where the feature lie is strongly dependent to the level of noise and the values of the geometric Laplacian. Motivated by this observations we decided to use the following initialization strategy. An initial value that significantly accelerates this approach is the following one:

$$\mathbf{m}_{ij}[0] = k_i\left(1 - \frac{e^{-|\,\widetilde{e}_{ij}\,|}}{\hat{\boldsymbol{\sigma}}_z[0]}\right) \;\forall\, i = 1, s \;\;,\;\; \forall\, j \;\in \{x, y, z\} \tag{18}$$

where $\hat{\boldsymbol{\sigma}}^2_z[0]$ is the noise variance vector and $\widetilde{e}_{ij}$ is the normalized value of smoothing factor e_{ij} [47], [48], expressed in mathematical terms as:

$$\mathbf{e}_i = \left\|\mathbf{L}^{1/2}[i]\mathbf{v}_i\right\|_2^2 = \mathbf{v}_i^T\mathbf{L}[i]\mathbf{v}_i, \quad \forall\, i = 1, \ldots, s \tag{19}$$

where $\mathbf{e}_i = [e_{i_x}\;\; e_{i_y}\;\; e_{i_z}]$. Small values of $\widetilde{e}_{ij}$ represent smooth areas while large values represent rough areas. For the estimation of the initial noise level $\hat{\sigma}^2_z[0]$, we start by assuming that the geometry data and the noise are uncorrelated, so the variance of the projected vertices of mesh can be expressed as [49]:

$$\sigma(\tilde{\mathbf{v}}\mathbf{u}) = \sigma(\mathbf{v}\mathbf{u}) + \sigma_z^2 \tag{20}$$

where $\sigma(\mathbf{v}\mathbf{u})$ represents the variance of vertices $\mathbf{v}$ in the $\mathbf{u} \in \Re^{3\times 1}$ direction, $\tilde{\mathbf{v}}$ are the affected by the noise vertices and σ_z is the standard deviation of the noise. The direction of minimum variance can be calculated by applying the PCA on different equalized patches of the mesh. For this reason we start by separating the mesh vertices $\tilde{\mathbf{v}}$ into ϕ overlapped patches. Each patch consists of k_ϕ vertices based on their relative distance which are estimated using k-nearest neighbor (k-nn) algorithm, so that the i^{th} patch can be represented by the $\tilde{\mathbf{v}}_i = [\tilde{\mathbf{v}}_{i_1}, \tilde{\mathbf{v}}_{i_2}, \ldots, \tilde{\mathbf{v}}_{i_{k_\phi}}]^T \in \Re^{k_\phi\times 3}$. The minimum variance direction can be estimated by the eigenvector associated to the minimum eigenvalue of the covariance matrix $\mathbf{R}_{\tilde{\mathbf{v}}_\phi} = \frac{1}{\phi}\sum_{i=1}^{\phi}\tilde{\mathbf{v}}_i\tilde{\mathbf{v}}_i^T$. The variance of the vertices projected onto the minimum variance direction equals the minimum eigenvalue of the covariance matrix $\mathbf{R}_{\tilde{\mathbf{v}}_\phi}$, i.e.:

$$\lambda_{min}(\mathbf{R}_{\tilde{\mathbf{v}}_\phi}) = \lambda_{min}(\mathbf{R}_{\mathbf{v}}) + \sigma_z^2 \tag{21}$$

where $\lambda_{min}(\mathbf{R})$ represents the minimum eigenvalue of the matrix $\mathbf{R}$. Vertices of smoothed patches, span a subspace whose dimension is smaller than $k_\phi \times 3$. These patches are low-rank and they can be used for the estimation of $\hat{\sigma}^2_z$ taking advantage of the fact that the minimum eigenvalue of their covariance matrix $\lambda_{min}(\mathbf{R}_{\mathbf{v}})$ can be assumed as zero. Since Gaussian noise has the same power in every direction and all eigenvalues are the same, we can estimate the noise variance from the subspace spanned by the eigenvectors of the covariance matrix $\mathbf{R}_{\tilde{\mathbf{v}}_\phi}$ with zero eigenvalues, i.e.:

$$\hat{\sigma}_z^2 = \lambda_{min}(\mathbf{R}_{\tilde{\mathbf{v}}_\phi}). \tag{22}$$

Therefore by substituting Eq. (22) in Eq. (18) we can evaluate the subspace size $\mathbf{m}_i$ that can be used for smoothing the noisy mesh without affecting noticeably the feature points.

4.2 Feature Preserving Surface Reconstruction

The coarse denoising step presented above, generates a smooth version of the normal vectors without deteriorating features. Although, the smoothed normals of the surface provide a more accurate representation of the original normal vectors; further processing is still required in order to: i) better highlight specific local characteristics, like edges and corners and ii) provide a more smooth version of the flat areas. To that end, the main goal is to accurately classify face normals into features and flat areas. Then we focus on finding the most representative normals for each face, based on the identified features across different neighborhoods. After identifying the ideal neighbors and weights, we apply a two stage graph spectral processing scheme that iteratively reconstructs the normals (e.g., each face normal is replaced by an average face normal across the ideal neighborhood) and then the vertices are updated accordingly.

Identification of Features

For each face i of the mesh we create a patch $\mathcal{P}_i = \{f_{i_1}, f_{i_2}, \ldots, f_{i_\rho}\}$ that consists of the closest faces selected by the k-nn algorithm. We consider different neighborhoods by generating k_f sets of ρ face normals:

$$\mathcal{N}_{c_i} = [\hat{\mathbf{n}}_{ci_1},\; \hat{\mathbf{n}}_{ci_2}, \ldots, \hat{\mathbf{n}}_{ci_\rho}]^T \;\forall\, i = 1, \ldots, k_f \tag{23}$$

After defining the covariance matrix for each vector $\mathcal{N}_{c_i} \in \Re^{\rho\times 3}$ as:

$$\mathbf{R}_{N_{c_i}} = N_{c_i}^T N_{c_i} \in \Re^{3\times 3} \tag{24}$$

we perform the following eigenvalue decomposition: $\mathbf{R}_{N_{c_i}} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$, where the columns of $\mathbf{U}$ are the eigenvectors of $\mathbf{R}_{N_{c_i}} \in \Re^{3\times 3}$ and $\boldsymbol{\Lambda} = diag(\lambda_{i1}, \lambda_{i2}, \lambda_{i3})$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

is a diagonal matrix with the corresponding eigenvalues $\lambda_{ij}\ \forall\ j = 1, 3$. Building on the same line of thought with the authors in [50], we can distinguish the following three cases: (a) corner ($\lambda_{i1} \cong \lambda_{i2} \cong \lambda_{i3}$), (b) edge ($\lambda_{i1} \cong \lambda_{i2} < \lambda_{i3}$) and (c) flat area ($\lambda_{i1} < \lambda_{i2} \cong \lambda_{i3}$). As a next step we use a k-means ($k = 3$) clustering to the eigenvalues in order to classify each face as a corner, edge or flat area. Since our focus is only on faces that represent potential features, we merge the other two cases (edge, corner) to one class (features) and another one, characterized as faces of flat areas (non-features). The described scheme is easily adaptable and can be used for the estimation of both small and large-scale features. In Fig. 3 we present a 3D model with geometric features of different scale, highlighted with red dots. By inspecting this figure it can be easily seen that our algorithm is capable of identifying both small and large scale features by simply modifying the patch size.

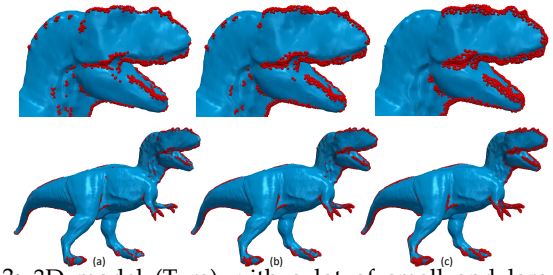

Fig. 3: 3D model (Tyra) with a lot of small and large-scale features. Feature classification using patch area of (a) 10, (b) 20, (c) 40 neighbors.

Selecting the Best Neighborhoods and Weights

In this subsection, we present a spectral method, similar to the one presented in Sec. 3.4, for estimating the representative face normal vectors, using an adjacency matrix of the face normals that can be considered as neighbors. To build this adjacency matrix it is important to identify the best neighborhood of each normal and then assign the appropriate bilateral weights. Our goal is to construct neighborhoods by identifying the faces that have common geometric properties. We initially execute the k-nn for each face of the mesh. The neighborhood of a non feature face i is formed by the k nearest neighbors $\mathcal{P}_i$. The k_p candidate neighborhoods of the a i feature face are all the patches, where the face i is participating as a nearest neighbor, defined as $\mathcal{B}_i = \{\mathcal{P}_{i1}, \mathcal{P}_{i2}, \ldots, \mathcal{P}_{ik_p}\}$, where the g^{th} candidate area of i^{th} face is represented as $\mathcal{P}_{ig}$. Our purpose is to find which of the candidate patches $\mathcal{P}_{ig}$ is the ideal representative area for i^{th} feature face.

Three parameters are investigated in order to identify the desirable connected areas: (a) the normalized difference τ_g between λ_{g3} and λ_{g1} eigenvalues (defined in the previous subsection) (b) the inner product between the face normal and the average normal of the patch and (c) the maximum distance between the face normal and the other face normals belonging to the same patch. The best neighborhood of a feature face i is the area that maximizes Eq. (25):

$$\mathcal{A}_i = \begin{cases} (\mathcal{P}_{ig} \mid max(\frac{\tau_g \cdot \xi_{ig}}{\omega_{ig}})) & f_i = feature \\ \mathcal{P}_i & otherwise \end{cases} \quad \begin{matrix} \forall\ i = 1, k_f \\ \forall\ g = 1, k_p \end{matrix} \tag{25}$$

$$where \quad \tau_g = \frac{|\lambda_{g1} - \lambda_{g3}|}{\lambda_{g3}} \tag{26}$$

$$\xi_{ig} = \langle \hat{\mathbf{n}}_{ci}, \sum_{j \in \mathcal{P}_{ig}} \frac{\hat{\mathbf{n}}_{cj}}{|\mathcal{P}_{ig}|} \rangle \tag{27}$$

$$\omega_{ig} = max(|\hat{\mathbf{n}}_{ci} - \hat{\mathbf{n}}_{cj}|_2) \quad \forall\ j \in \mathcal{P}_{ig} \tag{28}$$

$\mathcal{A}_i$ represents the area where each faces have similar geometric characteristics with the face i. In Fig. 4 we illustrate different candidate areas of the selected face, which is highlighted with blue color. After identifying the most

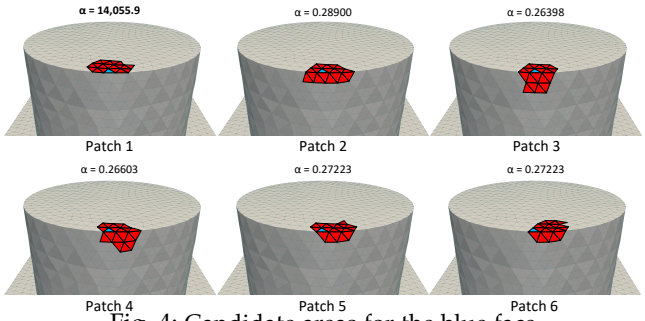

Fig. 4: Candidate areas for the blue face.

representative area we focus on the estimation of the ideal values for the weighted adjacency matrix. Each connected face has different weights, similar to the bilateral weights, which are evaluated using the following equation:

$$\mathbf{C}_w = \mathbf{W}_c \circ \mathbf{W}_s \circ \mathbf{C}_a \tag{29}$$

where $\circ$ denotes the Hadamard product, $\mathbf{C}_a$ is the binary adjacency matrix constructed using the k-nearest neighbors and $\mathbf{W}_c, \mathbf{W}_s \in \Re^{k_f \times k_f}$ are estimated according to:

$$\mathbf{W}_{c_{ij}} = \begin{cases} exp(-\left\|\mathbf{c}_i - \mathbf{c}_j\right\|^2 / 2\sigma_c^2) & if\ \mathbf{C}_{a_{ij}} = 1 \\ 0 & otherwise \end{cases} \tag{30}$$

$$\mathbf{W}_{s_{ij}} = \begin{cases} exp(-\left\|\hat{\mathbf{n}}_{ci} - \hat{\mathbf{n}}_{cj}\right\|^2 / 2\sigma_s^2) & if\ \mathbf{C}_{a_{ij}} = 1 \\ 0 & otherwise \end{cases} \tag{31}$$

Two Stage Processing of Normals and Vertices

The ideal neighborhoods and weights are then used to fine tune the normals according to:

$$\bar{\mathbf{n}}_c = (\mathbf{D}^{-1}\mathbf{C}_\mathbf{w})^\zeta \hat{\mathbf{n}}_c \tag{32}$$

where $\hat{\mathbf{n}}_c$ represents the smoothed normals, ζ is an integer that represents a denoising factor. The fine tuned normals are then used to update the vertices according to [31]:

$$\mathbf{v}_i^{(t+1)} = \mathbf{v}_i^{(t)} + \frac{\sum_{j \in \mathbf{\Psi}_i} \bar{\mathbf{n}}_{cj}(\langle \bar{\mathbf{n}}_{cj}, (\mathbf{c}_j^{(t)} - \mathbf{v}_i^{(t)}) \rangle)}{|\mathbf{\Psi}_i|} \tag{33}$$

$$\mathbf{c}_j^{(t+1)} = (\mathbf{v}_{j1}^{(t+1)} + \mathbf{v}_{j2}^{(t+1)} + \mathbf{v}_{j3}^{(t+1)})/3 \ \ \forall\ j \in \mathbf{\Psi}_i \tag{34}$$

where $\langle \mathbf{a}, \mathbf{b} \rangle$ represents the inner product of $\mathbf{a}$ and $\mathbf{b}$, (t) represents the number of iteration and the matrix $\mathbf{\Psi}_i$ is the cell of vertices that are directly connected with the vertex $\mathbf{v}_i$. This iterative process can be considered as a gradient descent process that is executed for minimizing the energy term $\sum_{j \in \mathbf{\Psi}_i} \|\bar{\mathbf{n}}_{cj}(\mathbf{c}_j^{(t)} - \mathbf{v}_i^{(t)})\|^2$ across all faces. An overview of the proposed method is presented in Algorithm 2.

5 Results

The effectiveness of our approach in terms of both reconstruction quality and computation complexity is highlighted by a thorough experimental study, presented in this section.

Algorithm 2: Two Stage Graph Spectral Processing.

Function: 3D Mesh denoising
Input : Triangle noisy mesh $\tilde{\mathbf{M}}$ corrupted by noise.
Output : Triangle denoising mesh.
`/* Coarse reconstruction: cutting off higher frequencies */`
1 Divide $\tilde{\mathbf{M}}$ into s overlapped submeshes $\tilde{\mathbf{M}}_s$ using METIS method described in [44] [45] ;
2 Use steps of **Algorithm** 1 ;
3 Recreate the whole smoothing mesh;
`/* Fine reconstruction: attenuating higher frequencies */`
4 Classify each face as feature or non-feature via Eq. (24) ;
5 Create weighted adjacency matrix $\mathcal{A}$ via Eqs. (25)-(31) ;
6 Fine denoising of guided normals $\bar{\mathbf{n}}_c$ via Eq. (32) ;
7 Update vertices via Eqs. (33)-(34) ;
8 **return** Triangle denoising Mesh ;

5.1 Experimental Setup

To provide an objective comparison between the tested solutions, we follow the general pipeline of our Two Stage Graph Spectral Processing (TSGSP) method described in Section 4. In all the experiments we have used a PC Intel core i7-2600 CPU @ 3.40GHz 3.70GHz, 10 GB RAM. The main core of the algorithms is written in C++ and Matlab.

Datasets & Metrics

We have used a wide range of CAD and scanned 3D models. More specifically, the experiments are carried out with four different types of datasets. (a) Scanned models [51] form Stanford dataset e.g. Stanford Bunny, (b) CAD (computer-aided design) models [52] e.g. Fandisk model which is provided courtesy of MPII by the AIM@SHAPE shape Repository, (c) Real scanned noisy models from 3 different devices (Kinect v1, Kinect v2, Kinect Fusion) and synthetic noise [9] and (d) Real scanned lungs affected by staircase effect using MRI device [12].

The quality of the denoising results is evaluated using a variety of different metrics that are shortly presented below:

- θ: represents the angle between the normal of the ground truth face and the resulting face normals, averaged over all faces.
- NMSVE (Normalized Mean Square Visual Error): has been shown to correlate well with perceived distortion by measuring the average error in the Laplacian and Cartesian domains [53].

5.2 Coarse Denoising Step Evaluation

In this section we evaluate the benefits of the coarse denoising step in terms of both reconstruction quality and execution time. In particular, we explicitly show that it could be efficiently used by many SoA approaches for improving their ability to deal with varying noise patterns.

Benefits of Orthogonal Iterations

In order to apply the method in the most efficient way, we have exhaustively investigated a wide range of parameters that appear in Eq. (8) which are: i) the number of iterations η, ii) the multiplication factor β, where $\mathbf{R}^{\beta} = (\mathbf{L} \cdot \mathbf{L}^T)^{\beta}$ and iii) the initialization $\mathbf{U}_m[0]$ for each submesh. Table 1 illustrates the impact of iterations and matrix multiplications in both reconstruction accuracy and execution time of the proposed OI approach. As we expected, $OI(\beta)$ is faster than $OI(\eta)$ when $\eta = \beta > 1$, since a matrix multiplication requires less computational time than performing one OI. The multiplication factor significantly affects the iterations required for convergence, while at the same time is much more computational efficient and comes at a lower cost than performing an orthonormalization step, which is required in each OI.

	Armadillo				**Hand**			
	SC1	**SC2**	**SC3**	**SC4**	**SC1**	**SC2**	**SC3**	**SC4**
η	0	1	2	3	0	1	2	3
β	8	4	3	2	6	3	2	2
t_1	3.02	2.24	2.45	1.35	17.61	11.65	6.69	6.69
t_2	2.71	5.17	7.37	10.99	23.41	42.04	59.20	73.30
t_{tot}	5.73	7.41	9.82	12.34	41.02	53.69	65.89	79.99

TABLE 1: Different configurations **SC1-SC4** of η and β values in order to attain an almost identical reconstruction accuracy using the SVD and OI method. t_1: required time for evaluating $\mathbf{R}^{\beta}$, t_2: required time for executing η iterations and $t_{tot} = t_1 + t_2$ (in seconds).

Table 2 shows how the initialization for each submesh affects the convergence rate of the OI approach. More specifically, we consider three individual initialization scenarios: (a) Semi-random Initialization using the sequence annotated by the METIS segmentation, (b) Consistent initialization: using always $\mathbf{U}_m[0]$ and (c) Sequential initialization: The subspace of each submesh is initialized to the subspace of a previously processed adjacent submesh.

	Sphere		**Fandisk**		**Tyra**	
$\mathbf{U}_m[i-1]$	t	min θ	t	min θ	t	min θ
Random	0.163	13.75	0.135	10.91	0.707	12.82
$\mathbf{U}_m[0]$	0.162	13.81	0.134	10.91	0.664	12.72
Adj.	0.165	13.8	0.149	11	0.720	12.83

TABLE 2: Execution time for different approach of $\mathbf{U}_m[i-1]$ initialization.

It can be easily noticed that the results are similar in terms of both reconstruction quality and execution time. This is attributed to the spatial coherence that arises when processing small patches of dense 3D meshes, resulting in similar connectivities, which is also evident from the subspaces of the matrices $\mathbf{R}[i]$. Due to this effect the OI based algorithm converges efficiently regardless of the initialization scheme. This is a very useful observation which can be effectively exploited in a parallel implementation set up.

Benefits of Dynamic Scheme for the Optimal m Identification

Fig. 5 presents (i) the ideal number $k - m$ of eigenvectors that need to be removed for minimizing the angle θ, (ii) the estimated subspace size m using the empirical rule of Eq. (18) and (iii) the estimated subspace size m using MBL algorithm. In Fig. 6 we illustrate the accuracy of the noise level estimation sub-module used for the MBL initialization. For the experimental results we adopted the following process. Firstly, a known Gaussian noise $\mathcal{N}(0, \sigma_z)$, using a wide range of noise variances $\sigma_z = [0.1, \cdots, 1]$ with 0.1 step, is added to each one of the vertex coordinated individually.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

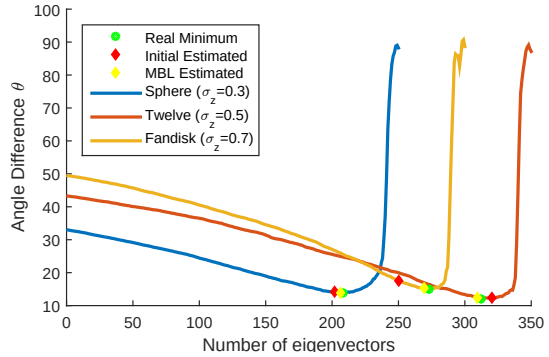

Fig. 5: The real and estimated subspace size for the minimization of θ.

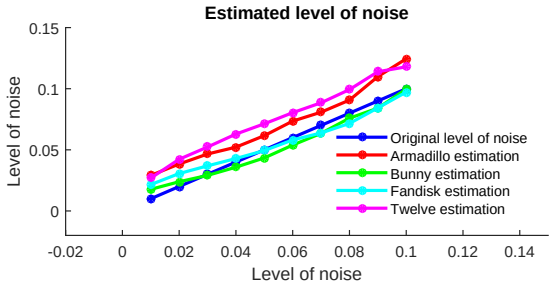

Fig. 6: Level of noise estimation for different noisy 3D meshes.

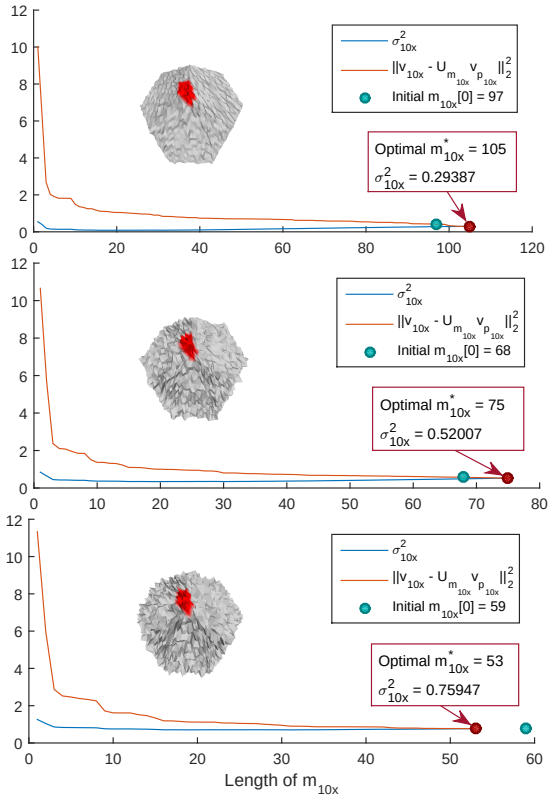

Fig. 7: Optimal values of m_{10_x} and σ_{z10_x} using MBL.

Although, there is a small deviation depending on the input model, it constitutes an ideal initial condition for the Bayesian learning step. In Fig. 7 we present an example of applying MBL algorithm in a specific submesh $i = 10$ for identifying the optimal values of m_x and σ_{z_x} using different level of noise in each case ($\sigma_z = 0.2, \sigma_z = 0.5, \sigma_z = 0.7$ correspondingly).

Benefits of Coarse Denoising as a Pre-Processing Step

We would like to emphasize that the coarse denoising method can be considered as an out-of-core method that could be employed by any other SoA method as a pre-processing step, optimizing both its reconstruction quality and its computational complexity. The reconstruction benefits can be easily identified by inspecting Fig. 8 which presents denoising results of SoA methods (first row) and the corresponding results after using the proposed coarse denoising approach as a pre-processing step (second row). Similarly, Fig. 8 shows the effects of the coarse denoising step in the execution time of the Guided normal approach [8]. By inspecting the number of iterations required for achieving a given reconstruction quality, the execution times of the coarse denoising step and of fine denoising iterations, it can be noted that the coarse denoising step accelerates the convergence rate of the fine denoising approach, thus significantly improving the total execution time of the whole denoising process.

5.3 Feature-Aware Fine Denoising Evaluation

For the evaluation we use a wide range of alternative approaches for selecting the best neighborhoods and the corresponding adjacency weights. More specifically, in Fig. 9 we present the original noisy mesh, the fine denoising output after using the k-nn patch directly without searching for the ideal area as described in Section 4, the results after using Gaussian weights and finally the results after using our proposed approach.

5.4 Comparisons Between TSGSP and SoA

In this section we present the results of our TSGSP method as compared to other SoA approaches. More specifically, we present an objective comparison between the tested solutions and the pipeline described in Section 4.

Computational Complexity Benefits

To highlight the superior performance of the proposed scheme in terms of computational complexity, we provide in Table 3 the execution times of the TSGSP approach as compared to the Guided normal algorithm [8], which is the dominant among the SoA competitors. Note that while both approaches, are based on a sequential update of the face normals and vertices, TSGSP results in lower execution times (68x-87x). This reduction is attributed to the application of the coarse denoising approach that filters out the high frequency components, accelerating the convergence speed of the necessary corrections/adjustments of the vertex positions. Moreover, it should be also noted that the selection of the ideal neighborhood for each face normal correction is also a crucial parameter that contributes to the reduction of the execution time required for readjusting the mesh faces/vertices during a fine processing step.

Results Assuming Complex Noise Patterns & Geometries

In Figs. 10-13, we present the denoised results of our method in comparison with other SoA methods. For the experiments

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

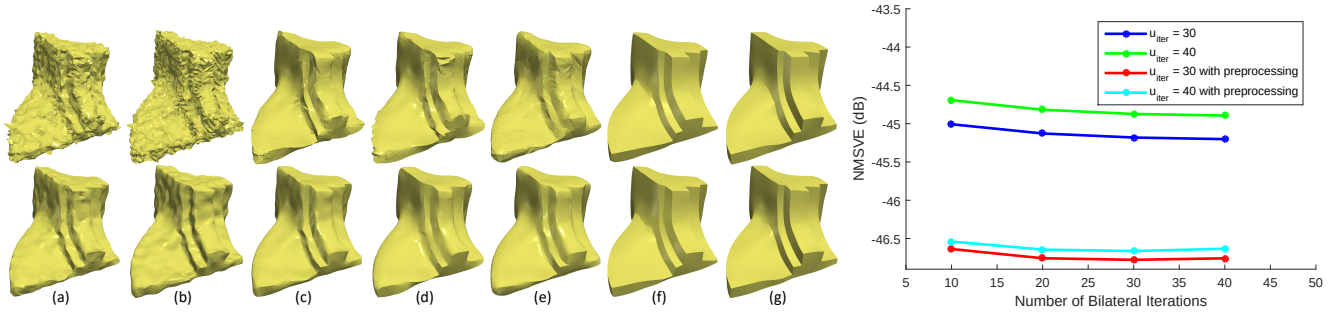

Fig. 8: [Left] Reconstruction results with (2nd row) and without (1st row) preprocessing step using the following approaches: (a) [28], (b) [29], (c) [31], (d) [32], (e) [32], (f) [10], (g) [8]. [Right] Coarse denoising step increases the reconstruction quality. In other words, less iterations are required in order to achieve the same quality (Fandisk).

	Guided mesh normal filtering							TSGSP						
Model	$\mathbf{k_{iter}}$	$\mathbf{t_{k_{iter}}}$	$\mathbf{u_{iter}}$	$\mathbf{t_{u_{iter}}}$	**N**	$\mathbf{t_N}$	t_{tot}	$\mathbf{t_{cd}}$	$\mathbf{t_{fd}}$	$\mathbf{u_{iter}}$	$\mathbf{t_{u_{iter}}}$	**N**	$\mathbf{t_N}$	$\mathbf{t_{tot}}$
Block σ=0.4	40	200.31	30	21.88	17550	10.63	232.82	10.17	32.59	20	14.78	4457	2.68	**60.32** -74%
Twelve σ=0.5	75	273.07	20	7.64	9216	8.11	289.55	4.72	18.67	30	11.54	2297	2.31	**37.24** -87%
Sphere σ=0.3	30	106.91	20	17.60	20882	6.53	131.05	12.27	18.10	12	10.44	3635	1.22	**42.03** -68%
Fandisk σ=0.7	50	257.13	20	10.61	12946	12.19	279.94	10.75	44.82	20	10.82	4464	4.34	**70.73** -74%

TABLE 3: Time performance for different models. k_{iter} presents the number of iterations for bilateral filtering normal executed in $t_{k_{iter}}$, u_{iter} is the number of iterations for vertex update executed in $t_{u_{iter}}$ and N is the number of points searched for finding ideal patches executed in t_N, t_{cd} corresponds to execution time for coarse denoising, feature extraction and level noise estimation, t_{fn} represents the execution time for fine denoising and t_{tot} is the total time expressed in seconds. The evaluation has been conducted by using the following models (V/F): Block (8771/17550), Twelve (4610/9216), Sphere (10443/20882), Fandisk (6475/12946).

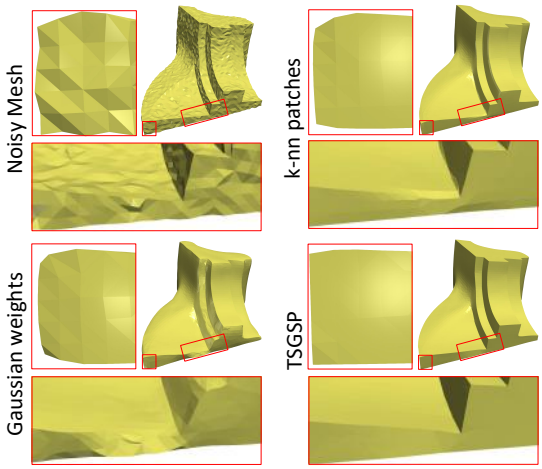

Fig. 9: Results after using a wide range of alternative approaches for selecting the best neighborhoods and the corresponding adjacency weights.

two different models are used which are captured by three different scanning devices (Kinect v1, Kinect-fusion, Kinect v2) as well as synthetic noise. Additionally, in each figure the average angular difference (in degrees) is presented. The results shows that the proposed scheme outperforms the other SoA with respect to reconstruction quality.

Heatmap visualizations Fig. 14 also highlight the distortion alleviation when both components (coarse and fine) are used. The staircase effect, caused by MRI and CT devices [12], is a type of noise that requires more sophisticated schemes that differentiate staircase noise effects from surface features. In contrast to other methods, which often mistakenly consider the staircase effect as a mesh feature, our method effectively handles this case. The features of the mesh are successfully identified due to the application of the coarse-denoising step. By inspecting Fig. 15 we see that most of the SoA approaches mistakenly consider the staircase effect as geometric features that need to be preserved. The superiority of our method as compared to the presented SoA approaches is attributed to the application of the coarse step that smooths the staircase effects and allows a more accurate identification of the true features. The heatmap visualization of the reconstruction error shows that the specialized method of [12] outperforms our approach, however, it should be noted that it creates irrational smoothness in the whole surface as well as a shrinking of the object, destroying also true features on the surface.

Data Driven Initialization Strategy

One major advantage of data-driven methods, e.g., [9] is their computational efficiency in online denoising cases. This attribute makes them capable of handling problems with tight timing restrictions in real time. However, it should be noted that for the case of varying or unknown noise patterns that are not included in the trainning dataset re-trainning is required that is essentially a very slow and computationally costly process. More specifically, in the approach of [9] while the execution time is only 1 second, the training process, using a dataset of 63 models, takes more than 720 seconds.

In this section, we study the performance of the proposed scheme when using some fixed pre-identified initialization parameters. Those parameters have been estimated

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

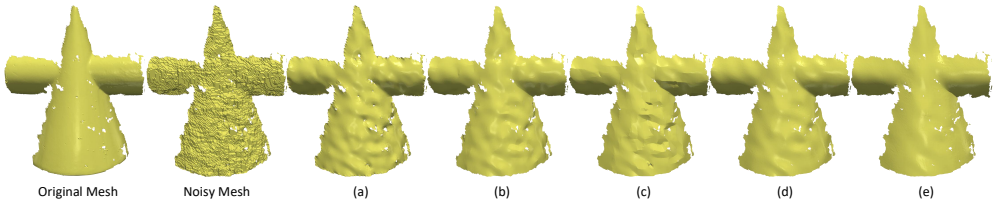

Fig. 10: Kinect v1. The θ metric is: (a) [32] 12.98°, (b) [8] 10.38°, (c) [10] 10.34°, (d) [9] 8.91°, (e) TSGSP $\mathbf{7.98}^\circ$ (cone).

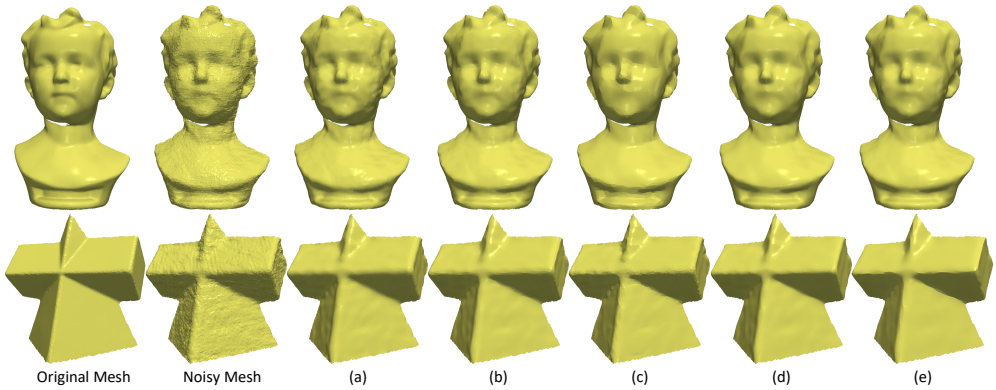

Fig. 11: Kinect-fusion. The θ metric is: (a) [32] 9.39°, (b) [8] 7.90°, (c) [10] 7.65°, (d) [9] 7.79°, (e) TSGSP $\mathbf{7.59}^\circ$ (boy). (a) 9.13°, (b) 7.84°, (c) 7.69°, (d) 7.59°, (e) $\mathbf{7.40}^\circ$ (pyramid).

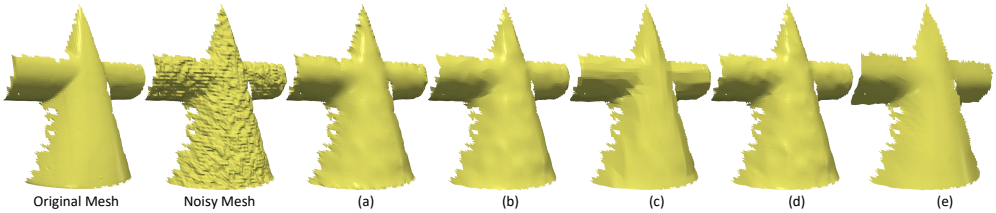

Fig. 12: Kinect v2. The θ metric is: (a) [32] 8.50°, (b) [8] 7.62°, (c) [10] 7.74°, (d) [9] 7.68°, (e) TSGSP $\mathbf{7.41}^\circ$ (cone).

using a training set of known models and they were kept fixed for a testing set of 3D objects. This extension is based on the observation that noisy objects, created by the same scanner devices or affected by the same type of synthetic noise, require similar initialization strategies. For the experiments we used the training set, affected by synthetic noise, provided by the benchmark of [9]. The learning process showed that besides the level of noise, geometric characteristics of the 3D object is also another factor which affects the values of the parameters. For this reason we divided the training set into two new sets (fine geometric & CAD models). This separation comes from the observation that fine geometric models need less iterations and higher value of σ_s since they contain small scale geometric features. On the other hand, the noise in CAD models is more easily distinguishable in flat areas stressing the need for more vertex update iterations with lower σ_s. The identified parameters are then used for evaluating the accuracy of the proposed scheme on two independent test sets consisting again by scanned and CAD models individually. The reconstruction accuracy is evaluated using the average angle difference between the normals of the original and the reconstructed object and the results are presented in Table 4. By inspecting this table it can be observed that the proposed scheme outperforms the method in [9] in 50% of the cases. Therefore, we conclude that our approach can be trained to work as a parameter free method. However, a training set of objects classified in different categories (e.g., according to the geometric details or the scanning device) is required in order to identify a fixed set of parameters for each class of objects.

Time Varying Noise Effects

The proposed scheme is capable of identifying both the level of noise and the spectral subspace where the features lie using a coarse step, while in the fine it identifies features and flat areas dealing efficiently with the aforementioned challenges and achieving at the same time, fast execution times. To investigate the performance of the proposed approach as compared to the SoA methods, we provide a study, where different parts of a 3D model have been affected by different noise patterns. For this experiment we used the dodecahedron model. We initially partitioned the 3D object in twelve different segments (corresponding to each side) and then we apply twelve different noise patterns generated assuming variable noise levels in each segment. The results of the proposed coarse to fine approach together with the other SoA approaches are presented in Fig. 16. The superiority of our method is more obvious when higher level of noise appears in each area. In this case, all the other methods fail to mitigate the noise effects, creating deformations and destroying geometric features.

6 Discussion and Conclusions

In this paper, we have introduced a novel approach to support fast and efficient mesh denoising of both CAD and fine geometric 3D objects assuming a wide range of noise power densities. Our main goal was to simultaneously maintain

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

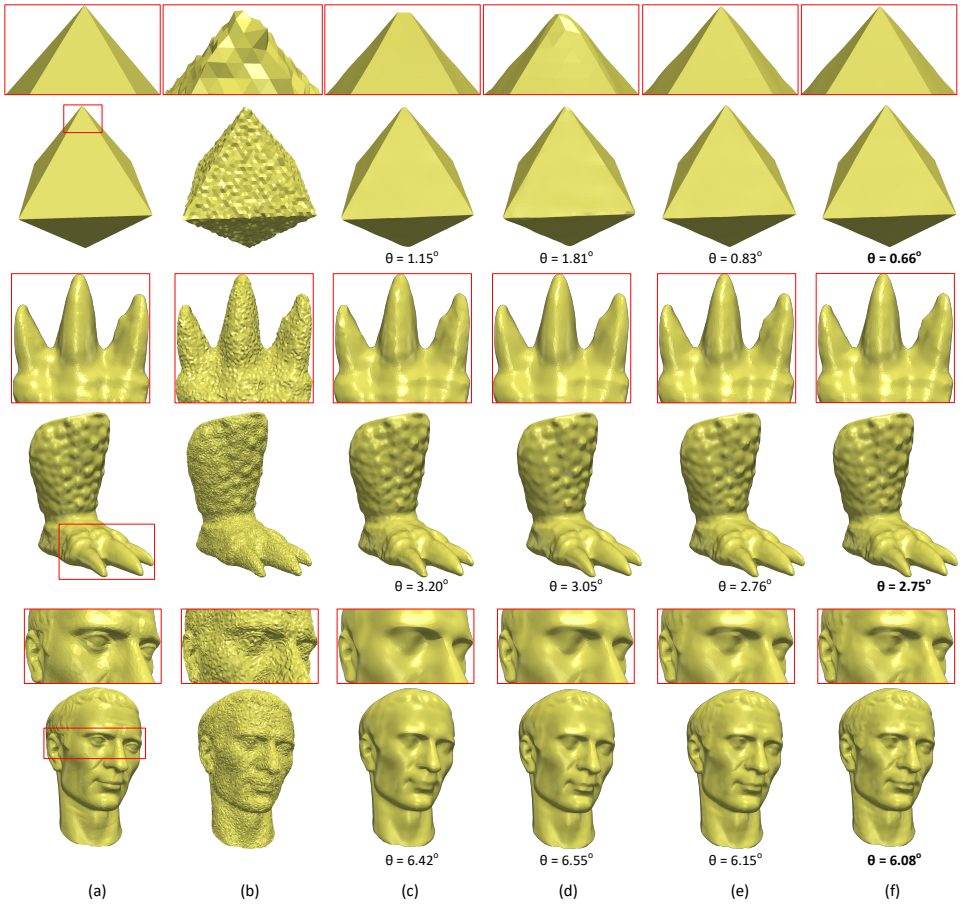

Fig. 13: (a) Original 3D model, (b) Noisy model, (c) Guided Normals Bilateral [8], (d) Cascaded Normal Regression [9], (e) parameter-free TSGSP, (f) TSGSP using ideal parameters.

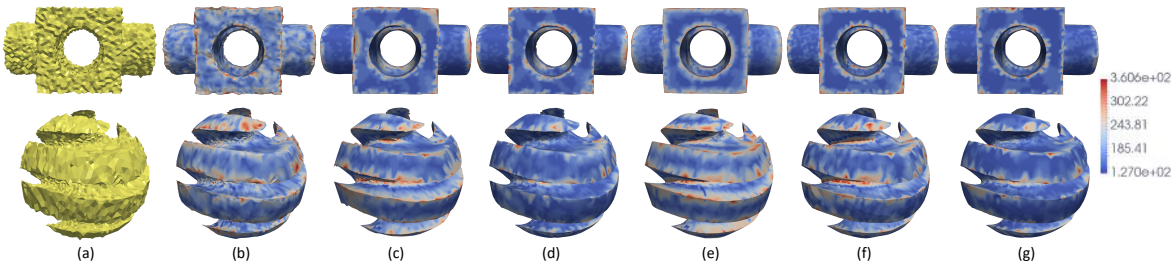

Fig. 14: (a) Noisy models and Heatmap visualization of (b) Non Iterative [29], (c) Fast & Effective [31], (d) Bilateral Normal [32], (e) L_0 min [10], (f) Guided Normal Bilateral [8], (g) TSGSP.

three, commonly conflicting, criteria: fast processing times, plausible reconstruction results and out-of-core behavior. The proposed two stage graph spectral processing (TSGSP) approach is composed of the coarse and fine denoising modules. However, despite the effectiveness demonstrated by a variety of presented experiments and metrics there are still some open issues awaiting investigation:

i) Recovering the structure of large scale scenes is without a doubt a stimulating scientific challenge. Several applications like tele-immersion, real-time surface mapping and tracking, aero-reconstruction for disaster management, have tight timing restrictions making an online reconstruction system indispensable. Therefore, we foresee a need to extend the presented approaches into an online-setting, in order to support such challenging problems.

ii) The fine denoising step requires the initialization of some parameters, such as the variance σ_s, the factor ζ and the number of iteration u_{iter}. However, these parameters can be identified accurately using a data driven approach. In this case, we follow similar initialization strategies for group of objects with similar characteristics, which are determined by the scanning device and the existence of sharp or small scale geometric features.

iii) The presented approaches cannot be used for denoising 3D models where poorly-shaped triangles are formed by vertices mostly lying on sharp edges, as presented in Fig. 17. However, all the SoA approaches also fail, thus rendering this problem as a challenge for the future.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2018.2802926, IEEE Transactions on Visualization and Computer Graphics

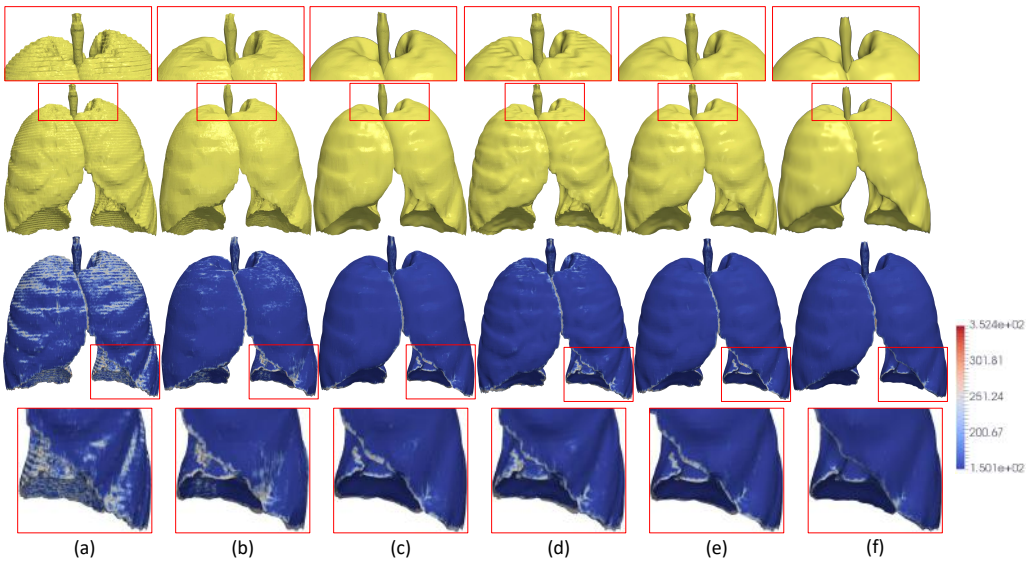

Fig. 15: (a) Original scanned model affected by staircase effect, (b) bilateral [28], (c) fast and effective [31], (d) bilateral normal [32], (e) TSGSP, (f) Staircase-Aware Smoothing [12], (Lungs model).

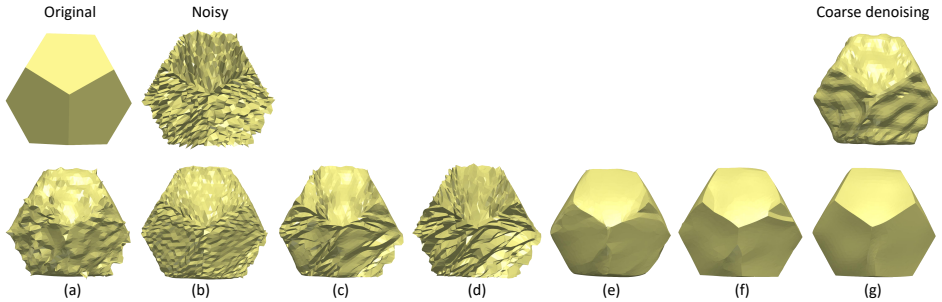

Fig. 16: Denosing results of Twelve model which have been affected by different noise patterns in each of its twelve sides. (a) Bilateral [28], (b) Non Iterative [29], (c) Fast & Effective [31], (d) Bilateral Normal [32], (e) L_0 min [10], (f) Guided Normal Bilateral [8], (g) TSGSP. Sides affected by higher levels of noise.

Name of Model	Our Method	Cascaded Normal Regression [9]	Set of Parameters
Block	**2.3414°**	2.3436°	Set 2
Bumpy torus	**3.9688°**	4.0464°	Set 1
Bunny hi	5.3268°	**5.1152°**	Set 1
Carter100K	**6.8722°**	7.8415°	Set 1
Child	**6.2145°**	6.9801°	Set 1
Chinese lion	**7.1285°**	7.6701°	Set 1
Cube	**0.7747°**	0.8656°	Set 2
Eight	6.0995°	**5.8017°**	Set 1
Eros100K	**8.0866°**	8.3486°	Set 1
Fertility	3.8456°	**3.6379°**	Set 1
Genus3	**2.4576°**	2.5751°	Set 1
Joint	**1.6837°**	1.697°	Set 2
Kitten	2.8386°	**2.8195°**	Set 1
Nicolo	4.5729°	**4.4868°**	Set 1
Part Lp	**2.5046°**	2.5422°	Set 2
Plane sphere	1.3816°	**1.2606°**	Set 2
Pulley	4.763°	**4.5899°**	Set 1
Pyramid	**0.9446°**	0.9912°	Set 2
Rolling stage	4.5187°	**4.1767°**	Set 1
Screwdriver	3.7645°	**2.9652°**	Set 1
Smooth feature	**0.9847°**	1.0085°	Set 2
Sphere	2.5285°	**2.3076°**	Set 2
Star	**1.5895°**	1.6502°	Set 2
Trim star	6.4181°	**4.1866°**	Set 1
Turbine Lp	3.7025°	**2.7707°**	Set 2

TABLE 4: Average angle θ using fixed pre-identified parameters on the different models included in the test set.

iv) Finally, threshold ρ of Eq. (23) should be carefully selected taking into account large scale features only, when the geometry of the processed model consist of both large and small scale features.

Fig. 17: TSGSP fails to perfectly denoise models where poorly-shaped triangles are formed by vertices mostly lying on sharp edges (Crank model).

ACKNOWLEDGMENTS

This work has been supported by the H2020-PHC-2014 RIA project MyAirCoach (grant no. 643607).

REFERENCES

[1] M. Roberts, D. Dey, A. Truong, S. N. Sinha, S. Shah, A. Kapoor, P. Hanrahan, and N. Joshi, "Submodular trajectory optimization for aerial 3d scanning," *CoRR*, vol. abs/1705.00703, 2017.

[2] B. Hepp, M. Nießner, and O. Hilliges, "Plan3d: Viewpoint and trajectory optimization for aerial multi-view stereo reconstruction," *CoRR*, vol. abs/1705.09314, 2017.

[3] M. Klingensmith, S. S. Sirinivasa, and M. Kaess, "Articulated robot motion for simultaneous localization and mapping (arm-slam)," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1156–1163, July 2016.

[4] A. Anwer, S. S. A. Ali, A. Khan, and F. Mériaudeau, "Underwater 3-d scene reconstruction using kinect v2 based on physical models for refraction and time of flight correction," *IEEE Access*, vol. 5, pp. 15 960–15 970, 2017.

[5] X. Fan, L. Zhang, B. Brown, and S. Rusinkiewicz, "Automated view and path planning for scalable multi-object 3D scanning," *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, vol. 35, no. 6, Nov. 2016.

[6] M. Giorgini, F. Barbieri, and J. Aleotti, "Ground segmentation from large-scale terrestrial laser scanner data of industrial environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 1948–1955, Oct 2017.

[7] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk, "The Digital Michelangelo Project: 3D scanning of large statues," in *Proceedings of ACM SIGGRAPH 2000*, Jul. 2000, pp. 131–144.

[8] W. Zhang, B. Deng, J. Zhang, S. Bouaziz, and L. Liu, "Guided mesh normal filtering," *Pacific Graphics*, vol. 34, no. 7, 2015.

[9] P.-S. Wang, Y. Liu, and X. Tong, "Mesh denoising via cascaded normal regression," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 232:1–232:12, Nov. 2016.

[10] L. He and S. Schaefer, "Mesh denoising via l_0 minimization," *ACM Trans. Graph*, vol. 32, no. 4, p. 64:164:8, 2013.

[11] "Noise analysis and synthesis for 3D laser depth scanners," *Graphical Models*, vol. 71, no. 2, pp. 34 – 48, 2009, iEEE International Conference on Shape Modeling and Applications 2008.

[12] T. Moench, S. Adler, and B. Preim, "Staircase-Aware Smoothing of Medical Surface Meshes," in *Eurographics Workshop on Visual Computing for Biology and Medicine*. The Eurographics, 2010.

[13] D. A. Field, "Laplacian smoothing and Delaunay triangulations," *Commun. Appl. Numer. Methods*, vol. 4, no. 6, p. 709712, 1988.

[14] G. Taubin, "A signal processing approach to fair surface design," *ACM SIGGRAPH 95*, pp. 351–358, 1995.

[15] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr, "Implicit fairing of irregular meshes using diffusion and curvature flow," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '99, 1999, pp. 317–324.

[16] C. Gotsman, "On graph partitioning, spectral analysis, and digital mesh processing," in *Shape Modeling International, 2003*. IEEE, 2003, pp. 165–171.

[17] B. Lévy, "Laplace-beltrami eigenfunctions towards an algorithm that" understands" geometry," in *Shape Modeling and Applications, 2006. SMI 2006. IEEE International Conference on*. IEEE, 2006, pp. 13–13.

[18] O. Sorkine, "Laplacian Mesh Processing," in *Eurographics 2005 - State of the Art Reports*, Y. Chrysanthou and M. Magnor, Eds. The Eurographics Association, 2005.

[19] H. Zhang, O. Van Kaick, and R. Dyer, "Spectral mesh processing," in *Computer graphics forum*, vol. 29, no. 6. Wiley Online Library, 2010, pp. 1865–1894.

[20] B. Vallet and B. Levy, "Spectral Geometry Processing with Manifold Harmonics," *Computer Graphics Forum*, 2008.

[21] G. Taubin, "Geometric Signal Processing on Polygonal Meshes," in *Eurographics 2000 - STARs*. Eurographics Association, 2000.

[22] P. Comon and G. H. Golub, "Tracking a few extreme singular values and vectors in signal processing," *Proceedings of the IEEE*, vol. 78, no. 8, pp. 1327–1343, Aug 1990.

[23] Y. Saad, "Analysis of subspace iteration for eigenvalue problems with evolving matrices," *SIAM Journal on Matrix Analysis and Applications*, vol. 37, no. 1, pp. 103–122, 2016.

[24] K. Hildebrandt and K. Polthier, "Anisotropic Filtering of Non-Linear Surface Features," *Computer Graphics Forum*, 2004.

[25] C. L. Bajaj and G. Xu, "Anisotropic diffusion of surfaces and functions on surfaces," *ACM Trans. Graph.*, vol. 22, no. 1, pp. 4–32, Jan. 2003.

[26] S. Yoshizawa, A. Belyaev, and H.-P. Seidel, "Smoothing by example: Mesh denoising by averaging with similarity-based weights," *Proc. IEEE Conf. Shape Model. Appl*, p. 9, 2006.

[27] P.-S. Wang, X.-M. Fu, Y. Liu, X. Tong, S.-L. Liu, and B. Guo, "Rolling guidance normal filter for geometric processing," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 173:1–173:9, Oct. 2015.

[28] S. Fleishman, I. Dror, and D. Cohen-Ori, "Bilateral mesh denoising," *COMPUTER GRAPHICS*.

[29] T. R. Jones, F. Durand, and M. Desbrum, "Non-iterative, feature-preserving mesh smoothing," *ACM Trans. Graph*, vol. 22, no. 3, p. 943949, 2003.

[30] J. Wang, X. Zhang, and Z. Yu, "A cascaded approach for feature-preserving surface mesh denoising," *Comput. Aided Des.*, vol. 44, no. 7, pp. 597–610, Jul. 2012.

[31] X. Sun, P. L. Rosin, R. R. Martin, and F. C. Langbein, "Fast and effective feature-preserving mesh denoising," *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*, vol. 13, no. 5, p. 925938, 2007.

[32] Y. Zheng, H. Fu, O. K.-C. Au, and C.-L. Tai, "Bilateral normal filtering for mesh denoising," *IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS*.

[33] Y. Shen and K. E. Barner, "Fuzzy vector median-based surface smoothing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 3, pp. 252–265, May 2004.

[34] M. Wei, J. Yu, W.-M. Pang, J. Wang, J. Qin, L. Liu, and P.-A. Heng, "Bi-normal filtering for mesh denoising." *IEEE Trans. Vis. Comput. Graph.*, vol. 21, no. 1, pp. 43–55, 2015.

[35] L. Zhu, M. Wei, J. Yu, W. Wang, J. Qin, and P.-A. Heng, "Coarse-to-fine normal filtering for feature-preserving mesh denoising based on isotropic subneighborhoods." *Comput. Graph. Forum*, vol. 32, no. 7, pp. 371–380, 2013.

[36] H. Zhang, C. Wu, J. Zhang, and J. Deng, "Variational mesh denoising using total variation and piecewise constant function space," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 7, pp. 873–886, July 2015.

[37] D. Nehab, S. Rusinkiewicz, J. Davis, and R. Ramamoorthi, "Efficiently combining positions and normals for precise 3D geometry," *ACM Trans. Graph*, vol. 24, no. 3, pp. 536–543, 2005.

[38] J. R. DIEBEL and S. THRUN, "A bayesian method for probable surface reconstruction and decimation," *ACM Trans. Graph*, vol. 25, no. 1, pp. 39–59, 2006.

[39] M. Wei, L. Liang, W. M. Pang, J. Wang, W. Li, and H. Wu, "Tensor voting guided mesh denoising," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 931–945, April 2017.

[40] X. Lu, Z. Deng, and W. Chen, "A robust scheme for feature-preserving mesh denoising," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 3, pp. 1181–1194, 2016.

[41] H. Fan, Y. Yu, and Q. Peng, "Robust feature-preserving mesh denoising based on consistent subneighborhoods," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 312–324, 2010.

[42] R. Wang, Z. Yang, L. Liu, J. Deng, and F. Chen, "Decoupling noises and features via weighted l1-analysis compressed sensing," *ACM Transactions on Graphics*, vol. 33, no. 2, pp. Article 18: 1–12, 2014.

[43] F. Cayre, P. Rondao-Alface, F. Schmitt, B. Macq, and H. Matre, "Application of spectral decomposition to compression and watermarking of 3D triangle meshgeometry," *Signal Processing: Image Communication*, vol. 18, no. 4, pp. 309–319, 2003.

[44] A. Lalos, I. Nikolas, E. Vlachos, and K. Moustakas, "Compressed sensing for efficient encoding of dense 3D meshes using model based bayesian learning," *IEEE Transactions on Multimedia*, vol. PP, no. 99, pp. 1–1, 2016.

[45] A. S. Lalos, I. Nikolas, and K. Moustakas, "Sparse coding of dense 3D meshes in mobile cloud applications," in *2015 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, 2015, pp. 403–408.

[46] P. Zhang, "Iterative methods for computing eigenvalues and exponentials of large matrices," 2009.

[47] Z. Karni and C. Gotsman, "Spectral compression of mesh geometry," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00, 2000, pp. 279–286.

[48] G. Lavoué, "A local roughness measure for 3D meshes and its application to visual masking," *ACM Trans. Appl. Percept.*, vol. 5, no. 4, pp. 21:1–21:23, Feb. 2009.

[49] X. Liu, M. Tanakai, and M. Okutomi, "Single-image noise level estimation for blind denoising," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 22, no. 12, 2013.

[50] M. Svub, P. Krsek, M. Spanel, V. Stancl, R. Barton, and J. Vadura, "Feature preserving mesh smoothing algorithm based on local normal covariance," p. FULL Papers, 2010.

[51] "The Stanford 3D Scanning Repository," http://graphics.stanford.edu/data/3Dscanrep/.

[52] " MPII by the AIM@SHAPE shape Repository," http://visionair.ge.imati.cnr.it/ontologies/shapes/.
[53] Z. Karni and C. Gotsman, "Compression of soft-body animation sequences," *Computers Graphics*, vol. 28, pp. 25–34, 2004.

Gerasimos Arvanitis is a Ph.D. candidate at the Department of Electrical and Computer Engineering Department, University of Patras, Patras, Greece. His main research interest is signal processing and especially image and 3D mesh processing.

Aris S. Lalos Electrical and Computer Engineering Department, University of Patras, Patras, Greece. Aris S. Lalos received the Ph.D. degree in signal processing for wireless communications from the Computer Engineering and Informatics Department (CEID), School of Engineering (SE), University of Patras (UoP), Rio-Patras, Greece, in 2010. He was a Research Fellow at the Signal Processing and Communications Laboratory, CEID, SE, UoP, Rio-Patras, Greece, from 2005 to 2010, with the Signal Theory and Communications (TSC) Department, the Technical University of Catalonia (UPC), from 2012 to 2015 and with the Visualization and Virtual Reality Group, University of Patras from 2015 until today. He is an author of 47 research papers in international journals (20), conferences (24) and edited books (3).

Konstantinos Moustakas Electrical and Computer Engineering Department, University of Patras, Patras, Greece. Konstantinos Moustakas (M07) received the Diploma and Ph.D. degrees in electrical and computer engineering from the Aristotle University of Thessaloniki, Thessaloniki, Greece, in 2003 and 2007, respectively.From 2007 to 2011, he served as a Post-Doctoral Research Fellow with the Information Technologies Institute, Centre for Research and Technology Hellas, Hellas, Greece. He is currently an Associate Professor with the Electrical and Computer Engineering Department, University of Patras, Patras, Greece, and Head of the Visualization and Virtual Reality Group, University of Patras. He has authored or coauthored more than 120 papers in refereed journals, edited books, and international conferences.Prof. Moustakas is a Member of the IEEE Computer Society and Eurographics.

Nikos Fakotakis Prof. Nikos Fakotakis received the B.Sc. degree from the University of London (UK) in Electronics in 1978, the M.Sc. degree in Electronics from the University of Wales (UK), and the Ph.D. degree in Speech Processing from the University of Patras, (Greece) in 1986. From 1986 to 1992 he was lecturer in the Electrical and Computer Engineering Department of the University of Patras, from 1992 to 1999 Assistant Professor, from 2000 to 2003 he has been Associate Professor and since 2003 he is Professor in the area of Speech and Natural Language Processing. Prof. Fakotakis is currently the Director of the Communication and Information Technology Division of the Electrical and Computer Engineering Dept. (since 2005), Director of the Wire Communications Laboratory (WCL) (since 2004), and Head of the Artificial Intelligence Group. The results of the scientific work conducted by Prof. Fakotakis or under his supervision has resulted in over 400 scientific publications in internationally recognized journals and conferences, which have been cited more than 3,000 times.