This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

A Partition Based Method for Spectrum-Preserving Mesh Simplification

Misranur Yazgan and Yusuf Sahillioğlu

Abstract—The majority of the simplification methods focus on preserving the appearance of the mesh, ignoring the spectral properties of the differential operators derived from the mesh. The spectrum of the Laplace-Beltrami operator is essential for a large subset of applications in geometry processing. Coarsening a mesh without considering its spectral properties might result in incorrect calculations on the simplified mesh. Given a 3D triangular mesh, this paper aims to simplify the mesh using edge collapses, while focusing on preserving the spectral properties of the associated cotangent Laplace-Beltrami operator. Unlike the existing spectrum-preserving coarsening methods, we consider solely the eigenvalues of the operator in order to preserve the spectrum. The presented method is partition based, that is the input mesh is divided into smaller patches which are simplified individually. We evaluate our method on a variety of meshes, by using functional maps and quantitative norms, to measure how well the eigenvalues and eigenvectors of the Laplace-Beltrami operator computed on the input mesh are maintained by the output mesh. We demonstrate that the achieved spectrum preservation is at least as effective as the existing spectral coarsening methods.

Index Terms—Mesh Simplification, Spectrum-Preserving, Laplace-Beltrami Spectrum, Partitioning

1 Introduction

Triangular meshes are frequently used to represent 3D models in geometry processing and computer graphics areas. Most of the applications in these areas prefer high-resolution models containing tremendous amount of details, in order to provide a more realistic experience. However, as the complexity of a mesh increases, the computational cost required to process it also increases. This is where mesh simplification methods come into the picture, in order to create a simpler version of the complex mesh containing fewer details, by reducing the number of vertices, edges and faces existing in the mesh.

Most of these mesh simplification methods focus on appearance preservation, which is the case preferred in areas such as rendering. Unfortunately, appearance-preserving methods fall short of maintaining the spectral properties of differential operators constructed on a mesh (see Figure 1), which are essential for some geometry processing tasks such as shape correspondence and spectral distance computations. When the simplification is performed by ignoring the spectral properties, the related computations carried out on the coarsened mesh become inaccurate.

This paper focuses on the problem of simplifying a 3D triangular mesh, while preserving the spectral properties of the associated Laplace-Beltrami operator. In the recent years, there have been major advancements in spectrum-preserving coarsening methods. However, most of these methods address the problem from a complete algebraic perspective, by relying on directly operating on the matrices corresponding to the differential operators, thus not producing a mesh as output. The others producing an output mesh are based on utilizing the eigenvectors of the differential operator, maintaining the eigenvalues indirectly. Unlike the previous methods, our method considers the eigenvalues of the Laplace-Beltrami operator instead of the eigenvectors, while outputting a simplified triangular mesh. Upon our evaluation on a variety of meshes, it is demonstrated that we preserve the spectrum of the Laplace-Beltrami operator at least as effectively as the existing methods.

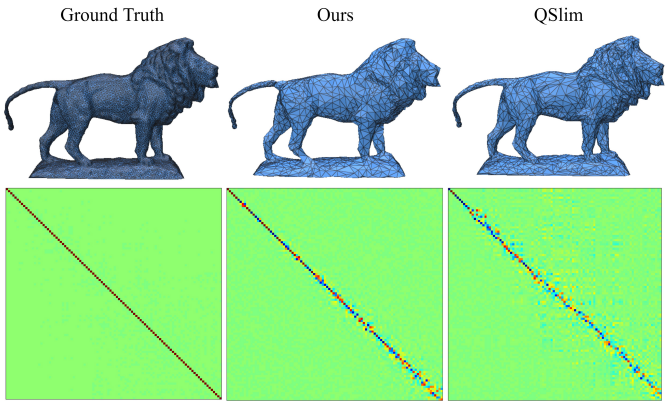


Fig. 1. Lion mesh is reduced from 20212 vertices to 2000 vertices (10% of its initial size). For an effective spectrum preservation, the functional map visualizations should be resembling the identity matrix. Since the main focus of QSlim [1] is to preserve the appearance of the mesh, it falls short of preserving the spectral properties, whereas we perform well despite 10% size.

The contributions of this study can be listed as the following:

- A new partition based mesh simplification method is proposed, whose primary purpose is to preserve the spectrum of the Laplace-Beltrami operator derived from a mesh. The method is capable of preserving the spectrum by considering only the 1D eigenvalues of

- *M. Yazgan and Y. Sahillioğlu are with the Department of Computer Engineering, Middle East Technical University, Ankara, Turkey, 06800. E-mail: see http://ceng.metu.edu.tr/∼ys*

Manuscript received Month Day, 2023

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

the operator, whereas the previous methods are built around the higher-dimensional eigenvectors.
- The proposed method is able to perform the spectrum preservation by considering only a small number of eigenvalues, since it is a partition based method.
- Among the existing spectrum-preserving coarsening methods, our method is one of the two methods which produce a mesh as an output. Please also see Table 1 for the positioning of our paper in literature.

TABLE 1
Classification of fully-automatic spectral simplification methods. Note that only our method is highly parallelizable and hence has the potential to be faster with a GPU implementation (currently CPU).

	Operates On		Mesh Out		Speed		Spect. Preservation		
	Evals	Evecs	Yes	No	Fast	Med	High	Med	Low
Ours	✓		✓			✓	✓		
[2]		✓		✓		✓			✓
[3]		✓	✓		✓			✓	
[4]		✓		✓		✓		✓	

1.1 Preserving Eigenvalues vs. Eigenvectors

Being our main novelty and motivation, it is worth to clarify why preserving the spectrum (the eigenvalues) is more important than eigenvectors. One main reason of using eigenvectors is that if you can preserve those eigenvectors, then you automatically preserve the eigenvalues (see Appendix C in [2]). So conceptually, our competitors [2]–[4] are also aiming for preserving eigenvalues.

We claim that directly comparing eigenvalues could also be a valid approach if one can robustly quantify the difference in eigenvalues, as we do in our paper. Another motivation is rather obvious: dealing with one dimensional scalar eigenvalues instead of multidimensional eigenvectors is theoretically much more efficient. In practice, however, we could not find the proper algebraic update rule that fully demonstrates this efficiency. We, instead, used our partitioning and similar edge elimination heuristics which worked sufficiently well. Please note that this naive way of utilizing the one dimensional eigenvalues may establish a ground for more advanced future works whose main focus is spectrum preservation.

We also note the 'hear the shape of the drum' application, i.e., recovering the geometric shape from pure eigenvalues (no eigenvector needed). While this recovery is theoretically impossible, recent studies such as [5] show practical progress for flat geometries. Hence, a simplification algorithm preserving eigenvalues, such as ours, can 'hear the shape of the drum' at arbitrary resolutions. Finally, while eigenvectors are equivalent up to sign before and after coarsening, there is no such ambiguity with the eigenvalues.

Although we explicitly preserve the first 15 smallest eigenvalues in our algorithm for efficiency purposes, we observe in all our experiments that the remaining eigenvalues are also preserved as a by-product (please see the trend for the first 100 eigenvalues in Figure 13). We can justify our choice of 15 eigenvalues by shape representation [6], retrieval [7], descriptor [8], registration [9], correspondence [10] [11], isospectralization [5], and sampling [12], [13] literature that rely on similar moderate number of eigenvalues. The main reason of success in all these applications as well as ours stems from the fact that a moderate size of small eigenvalues already covers sufficient variation in the spectrum and calculating high-frequency eigenvalues have computational errors, as reported in [14]. Note finally that, our execution time grows linearly with the number of eigenvalues (Figure 20).

2 Related Work

This section addresses the previous work related to mesh simplification, the use of the Laplace-Beltrami operator in spectral geometry processing and spectral coarsening, considering the topics that our method is built on.

Mesh simplification has been a well-studied topic in computer graphics area due to the growing need to be able to represent the meshes at different resolutions corresponding to different level-of-details. For the current context, mesh simplification methods can be categorized as appearance-preserving and spectrum-preserving methods. The majority of the simplification methods are focused on preserving the appearance and the geometric properties of the mesh. The most prominent previous examples in this area include mesh optimization [15] and mesh decimation [16]. One of the most well-known algorithms among the appearance-preserving simplification methods is the quadric error metrics method introduced by Garland and Heckbert [1]. Later, this greedy edge collapse algorithm is extended to preserve a variety of vertex attributes such as textures, colors or normal vectors along with the geometry [17], [18]. For the appearance-preserving methods, [19] provides a comprehensive study, where multiple mesh simplification methods are examined and compared. Even though these methods, that can also benefit further from multiresolution structures for significantly large inputs [20], manage to preserve the appearance of the mesh successfully, they fall short of maintaining the spectral properties of differential operators constructed on a mesh, which lie at the core of some geometry processing tasks such as shape correspondence [21].

The Laplace-Beltrami operator has been utilized for a variety of spectral geometry processing tasks for many years [22]. The use of Laplacian operator for mesh processing was first introduced by Taubin [23], pointing out the analogy between the Laplacian operator and the Fourier analysis. The eigenvalues and eigenvectors of the Laplacian are then exploited in areas such as mesh compression [24], mesh segmentation [25] and shape correspondence [26].

Alongside these studies, since the Laplacian operator is invariant under isometric transformations, it also served as a robust foundation for deformation-invariant shape descriptors. With this purpose, the eigenvalues of the Laplacian operator are utilized for extracting fingerprints called Shape-DNA, to represent a surface or a mesh [8], [27]. These works have proved that the spectrum of the Laplacian has a discriminative power that is capable of capturing the global properties of a shape. However, it should also be noted that the spectrum of the Laplacian does not provide a complete identification for the shape, since there are some rare cases, where two non-isometric shapes have the same spectrum. The use of eigenvalues alone as shape descriptors is one of our source of inspirations for depending this simplification method on the eigenvalues of the Laplacian operator.

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

Following these works, shape signatures called global point signature (GPS) [6] and heat kernel signature (HKS) [28] are introduced, including the eigenvectors into the scenario as well. These shape descriptors are utilized in works such as detecting global intrinsic symmetries of the shapes by Ovsjanikov et al. [29].

In the recent years, there has been significant developments in simplification methods which are focused on preserving the spectral properties of a mesh rather than just the appearance. Öztireli et al. [30] resampled points on a manifold surface by preserving the spectrum of the Laplacian operator. Similarly, Liu et al. [2] presented a spectrum-preserving coarsening method, which is also built on sampling points from the original mesh. Their method can be directly applied to the discrete geometric operators derived from a mesh including the Laplacian operator. They also introduced a metric build upon functional maps [31] to measure how well the spectrum of an operator is preserved after the coarsening. The mentioned metric is utilized in this work, as well as in [3] and [4]. However, both of the proposed methods do not produce a mesh. Later, Lescoat et al. [3] proposed a spectral mesh simplification method built upon the formulation presented in [2], which produces a mesh as output. They altered the greedy edge collapse algorithm introduced by Garland and Heckbert [1] with a spectral cost metric. Their spectral cost relies on the eigenvectors of the Laplacian while preserving the spectrum, whereas our method focuses directly on preserving the eigenvalues of the Laplacian operator. Following the work of Liu et al. [2], Chen et al. [4] proposed an operator coarsening scheme using chordal decomposition, enabling the optimization of an operator separately from the mesh. The main difference that separates our method from the existing spectral simplification methods is that our method solely relies on the eigenvalues of the Laplacian operator, whereas the others utilize the eigenvectors. Given a mesh, a list of Laplacians defined in different dimensionalities, and associated spectral bands, recent [32] coarsens the mesh by deciding the order of contractions greedily. Another recent study [33] moves the focus from appearance to solving equations on surface while performing mesh simplification.

3 METHOD

Our spectrum-preserving mesh simplification algorithm is based on edge collapse operations. The main goal of the method is to preserve the spectral properties of the input mesh at the low frequencies as much as possible, while reducing the number of elements used to represent the mesh. For this, we only consider the smallest k eigenvalues, since the high-frequency components will not be present on the simplified mesh.

3.1 Algorithm

The input to the algorithm is a manifold triangular mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$, which can possibly contain boundaries. After the simplification process, it outputs a coarser mesh $\widetilde{\mathcal{M}} = (\widetilde{\mathcal{V}}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{F}})$ with spectral properties as close as possible to that of $\mathcal{M}$. In addition, the algorithm also requires that the following inputs are provided:

- m: the desired number of vertices in the simplified mesh
- k: the number of eigenvalues to preserve
- p: the number of partitions that the mesh will be divided into
- x: the number of similar edge collapses
- n: the number of edges to consider for the similar edge collapses

Algorithm 1 Spectrum-Preserving Simplification

Input: $\mathcal{M} = (\mathcal{V}, \mathcal{E}, \mathcal{F}), m, k, p, x, n$
Output: $\widetilde{\mathcal{M}} = (\widetilde{\mathcal{V}}, \widetilde{\mathcal{E}}, \widetilde{\mathcal{F}})$

$\widetilde{\mathcal{V}} \leftarrow \mathcal{V}; \widetilde{\mathcal{E}} \leftarrow \mathcal{E}; \widetilde{\mathcal{F}} \leftarrow \mathcal{F}$
Divide the mesh $\widetilde{\mathcal{M}}$ into smaller partitions $\widetilde{\mathcal{M}}_1, \widetilde{\mathcal{M}}_2, \ldots, \widetilde{\mathcal{M}}_p$ where $\widetilde{\mathcal{M}}_i = (\widetilde{\mathcal{V}}_i, \widetilde{\mathcal{E}}_i, \widetilde{\mathcal{F}}_i)$
Assign partitions $\widetilde{\mathcal{M}}_1, \widetilde{\mathcal{M}}_2, \ldots, \widetilde{\mathcal{M}}_p$ to threads $T_1, T_2, \ldots, T_p$
Calculate the number of edges that will be collapsed in each partition $nE_1, nE_2, \ldots, nE_p$
Assign threads $T_1, T_2, \ldots, T_p$ to thread batches $B_1, B_2, \ldots, B_b$
for batch B_i in $B_1, B_2, \ldots, B_b$ **do**
  for thread T_j in batch B_i **do**
    $\Lambda_{in} = \{\lambda_{in_1}, \ldots, \lambda_{in_k}\} \leftarrow$ the first k eigenvalues of the Laplacian for partition $\widetilde{\mathcal{M}}_j$
    while $nE_j > 0$ **do**
      $allCosts \leftarrow FindEdgeCosts(\widetilde{\mathcal{M}}_j, \widetilde{\mathcal{E}}_j, k, \Lambda_{in})$
      $bestEdges \leftarrow$ edgeIds of $allCosts[0:n]$
      Collapse $bestEdges[0]$; $nE_j \leftarrow nE_j - 1$
      for i in $1, \ldots, x$ **do**
        $costs \leftarrow FindEdgeCosts(\widetilde{\mathcal{M}}_j, bestEdges, k, \Lambda_{in})$
        Collapse $costs[0]$; $nE_j \leftarrow nE_j - 1$
      end for
    end while
  end for
end for

Algorithm 2 FindEdgeCosts

Input: $\widetilde{\mathcal{M}}_j = (\widetilde{\mathcal{V}}_j, \widetilde{\mathcal{E}}_j, \widetilde{\mathcal{F}}_j), \mathcal{E}, k, \Lambda_{in}$
Output: $costs$

$costs \leftarrow \{\}$
for edge $e \in \mathcal{E}$ **do**
  $result \leftarrow$ Collapse edge e
  if $result$ is $successful$ **then**
    $\Lambda_{out} = \{\lambda_{out_1}, \ldots, \lambda_{out_k}\} \leftarrow$ the first k eigenvalues of the Laplacian for partition $\widetilde{\mathcal{M}}_j$
    $cost_e \leftarrow$ the difference between Λ_{in} and Λ_{out} wrt L_{evd} norm
    Reverse the edge collapse
  end if
  Add $(e, cost_e)$ into $costs$
end for
Sort $costs$ wrt increasing cost
return $costs$

Algorithm 1 summarizes the overall method, while Algorithm 2 presents the main idea lying at the core of the method, which is to compare the distance between the

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

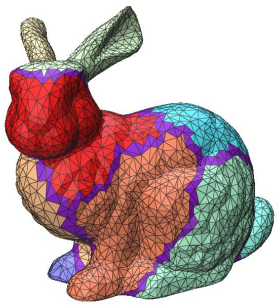


Fig. 2. Bunny mesh with 3485 vertices is divided into 10 partitions. The triangles in the edge-cut region are marked with purple for better understanding.

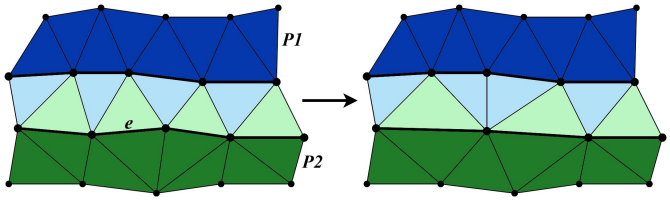


Fig. 3. Indirect simplification of the edge-cut region: If edge e lying on the boundary of partition P_2 is collapsed, it results with the removal of a triangle inside the edge-cut region. Here, light blue and light green triangles represent the triangles which will be removed indirectly with the collapse of the edges lying on the boundary of P_1 and P_2 respectively.

eigenvalues of the Laplacian computed on the original mesh and the eigenvalues obtained after an edge collapse, for measuring spectrum preservation. It is based on collapsing the edges given in the edge set $\mathcal{E}$ one by one and associating each edge with a cost measuring how much the spectrum of the Laplacian operator is affected from the current collapse. These costs are calculated by comparing the set of first k eigenvalues with respect to the norm provided in Equation 1 [7]. Here, $\lambda_i^{\mathcal{M}}$ and $\lambda_i^{\widetilde{\mathcal{M}}}$ stand for the i^{th} eigenvalues of the Laplacian operators derived from the input and output meshes respectively.

$$L_{\text{evd}}(\mathcal{M}, \widetilde{\mathcal{M}}) = \frac{1}{2}\sum_{i=1}^{k} \frac{\left[|\lambda_i^{\mathcal{M}}|^{\frac{1}{2}} - |\lambda_i^{\widetilde{\mathcal{M}}}|^{\frac{1}{2}}\right]^2}{|\lambda_i^{\mathcal{M}}|^{\frac{1}{2}} + |\lambda_i^{\widetilde{\mathcal{M}}}|^{\frac{1}{2}}} \tag{1}$$

Since the eigen-decomposition of the Laplacian operator takes very long time, performing this cost association for every edge in the mesh at each step is impractical. To avoid this, we introduced several approaches on top of the core idea, which are explained one by one in the following sections.

3.2 Partitioning

The input mesh is divided into smaller partitions and each partition is simplified separately. Thus, the Laplacian matrices are constructed within the partitions, instead of constructing them for the whole mesh. This effectively decreases the size of the Laplacian operator, leading to much faster eigenvalue computations. In addition, when each partition is treated separately, there are fewer edges to consider while selecting the best edge to collapse. A sample partitioning is provided in Figure 2. Although such a local approximation of the spectrum through partitions is not theoretically justified, we experimentally demonstrate that the eigenvalues are preserved consistently better than the state-of-the-arts (Figure 13). Besides, spectral compression [24] and shape matching [34] studies also justify local approximations similar to ours for their own purposes.

Alongside its benefits, partitioning the vertices may introduce some problems to the visual quality of the output mesh. When each vertex is assigned to a partition, the triangles left between the partition boundaries constitute the edge-cut regions. Since the simplification is performed within the partitions, the edges lying inside these regions are not collapsed directly, but they get simplified indirectly when the edges lying on the partition boundaries are collapsed. This indirect simplification is illustrated in Figure 3. In this way, the triangles inside the edge-cut regions are also included in the simplification process and these regions are not left with the original resolution. However, it is still best to minimize these regions as much as possible. For this, we utilized a partitioning tool called METIS [35], which provides an option to prioritize minimizing the edge-cut regions, specifically the number of edges in the edge-cut regions, while keeping the partition sizes balanced. It also guarantees that every edge belongs to only one partition, so that the partitions do not affect each other.

As experimentally demonstrated in the previous spectrum-preserving coarsening methods [2], [3], for proper spectrum preservation, the number of vertices left in each partition on the coarsened mesh should be at least $3 \times k$. Hence, this situation bounds the number of vertices that can be removed from a partition, allowing the simplification only up to a point. To overcome this downfall of partitioning, if the desired output size m could not be achieved with the given number of partitions p, we apply multi-step simplification with hierarchical partitioning. With this, the simplification is performed in multiple steps and at each step we simplify the mesh only until an intermediate resolution. A sample hierarchical partitioning can be found in Figure 4.

3.3 Similar Edge Elimination

In this context, similar edges are defined as the edges, whose removal affect the spectrum with an amount as close as possible to each other. We essentially cache a batch of edges without recomputing the Laplacian, and simplify this cache first.

In the algorithm, at each step, the costs are calculated for all the edges in the partition according to how much their removal affect the original spectrum. Then, the edges are ordered with respect to this cost. At this point, instead of only collapsing the edge with the lowest cost, we can consider the best n edges. However, we do not collapse these best n edges successively, because collapsing them without any checks results with a worse spectrum preservation. Instead, for the upcoming x steps, we select the best edge among these best n edges found in the previous step, instead of considering all the edges in the partition. In this way, we consider all edges in a partition once every x steps, and for the remaining steps, we select the edge from the best n edges found in the previous step. It should be noted that

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

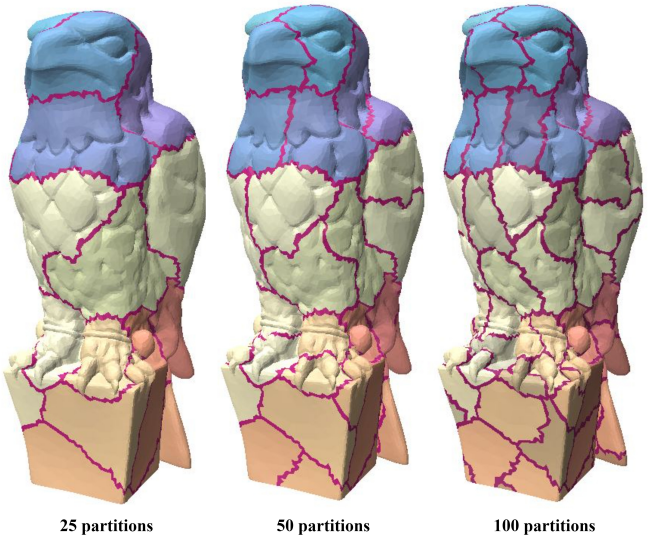


Fig. 4. Considering the partitionings from left to right, each partition is divided into 2 smaller partitions, resembling a tree structure. The simplification is performed from right to left, by simplifying the mesh until an intermediate resolution at each step.

n must be greater than $3 \times x$, since some of these n edges will get removed indirectly through other collapses in this subset. For our purposes, x is selected as 4 and n is selected as 20.

3.4 Threading

Since the mesh is already divided into partitions and they are all treated separately, we can run the partitions in parallel by assigning each partition to a thread. However, if all of the partitions are run at the same time, the neighboring partitions affect each other causing race conditions. In order to eliminate the potential race conditions without utilizing concurrency control mechanisms and to avoid making the method more complicated, threads are run batch by batch, in a way that no two neighboring partitions are run at the same time. The problem of assigning the threads into batches is analogous with the well-known graph coloring problem, for which finding the optimal solution is NP-complete [36]. However, using one of the greedy solutions providing an upper bound on the number of batches is sufficient for our purposes.

The greedy solution utilized here orders the vertices with respect to their degrees decreasingly and assigns the colors to vertices in that order [37]. In our case, vertices correspond to partitions and colors correspond to batches. A partition's degree is defined as the number of its neighboring partitions. By employing this greedy algorithm, the number of batches required is guaranteed to be at most one more than the maximum number of neighbors a partition has. In this way, significant time improvements and deterministic results are achieved through threading, while ensuring that no partition affect each other.

4 Results

In order to highlight our main focus and contribution, which is the direct preservation of eigenvalues, we first provide

Fig. 5. The partitions running at the same time are marked with the same color, where each color represents a different thread batch.

spectrum preservation results and then end with our execution times, which is slower than the fastest competitor [3].

Our method is evaluated by considering a variety of metrics. The results are obtained on a machine with a 4-core Intel i7-6700HQ 2.60GHz CPU and 16 GB of RAM. We focus on preserving the first 15 eigenvalues ($k = 15$). However, the functional maps are of size 100×100, so that we can also demonstrate the spectrum preservation beyond the first 15 components. As dataset, we used models common to our competitors as well as the first 5 models from each of the 10 categories of [38]. We have a CPU implementation in C++ that uses Spectra library [39]. See author's website for code.

4.1 Evaluation Metrics

We used functional maps and the quantitative metrics which are previously introduced by Liu et al. [2] and Lescoat et al. [3], in order to measure how well the spectrum of a mesh is preserved after the simplification. This same test protocol enables fair comparisons with our competitors [2]–[4].

Functional maps [31] describe a way to transfer functions from one shape to another. In spectral mesh simplification context, they are utilized as a measure to evaluate how well the eigenvectors of the Laplacian operator $\widetilde{L} \in \mathbb{R}^{m \times m}$ computed on the output mesh $\widetilde{\mathcal{M}}$ represents the eigenvectors of the Laplacian operator $L \in \mathbb{R}^{n \times n}$ computed on the input mesh $\mathcal{M}$. In order to take only the low frequency components into account, the functional map C will be considered for the first k eigenvectors. Therefore, the functional map $C \in \mathbb{R}^{k \times k}$ between the input and output meshes can be defined as the following:

$$C = \widetilde{\Phi}^T \widetilde{M} P \Phi \tag{2}$$

Here, $\Phi \in \mathbb{R}^{n \times k}$ and $\widetilde{\Phi} \in \mathbb{R}^{m \times k}$ are the set of eigenvectors corresponding to the first k eigenvalues of the Laplacian operator obtained on the input and output meshes respectively. $\widetilde{M} \in \mathbb{R}^{m \times m}$ stands for the mass matrix constructed on the output mesh, and $P \in \mathbb{R}^{m \times n}$ represents the projection matrix. The optimal functional map is a diagonal matrix filled with entries 1 and -1. Thus, the closer the functional map to a diagonal matrix, the better the spectrum is preserved.

Projection matrix is a fine-to-coarse operator allowing to transfer functions across meshes at different resolutions.

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

Since the Laplacian operators computed on the input and output meshes have different sizes, their corresponding eigenvectors are of different lengths as well. In order to be able to compare them in a meaningful way, projection matrix P is utilized in the functional map. We computed P during the simplification process as in [3].

Projection matrix can be defined in terms of a series of Q matrices, such that $P = Q_n Q_{n-1} \dots Q_2 Q_1$. Here, each edge collapse operation is associated with a restriction matrix $Q \in \mathbb{R}^{n-1 \times n}$, where n denotes the number of vertices existing in the mesh before the collapse. Q matrix summarizes the changes in the vertex set of the mesh caused by an edge collapse, such that $V^{'} = QV$.

Assume that edge e, joining vertices u and v is collapsed, where u is kept while v is deleted from the mesh. Let $u^{'}$ be the index of u after collapse, also the index of the merged vertex, and $x^{'}$ be the index of vertex x after collapse. Considering this notation, the only non-zero elements in the Q matrix will be $Q_{u^{'}u} = 0.5$, $Q_{u^{'}v} = 0.5$ and $\forall x \in V - \{u, v\}$, $Q_{x^{'}x} = 1$, since the merged vertex position is selected as the midpoint of the edge.

The quantitative norms previously introduced by Lescoat et al. [3] for not only relying on visual inspections are provided in Equation 3.

$$\begin{aligned} \text{Laplacian commutativity: } \|C\|_L^2 &= \frac{\|C\Lambda - \widetilde{\Lambda}C\|^2}{\|C\|^2} \\ \text{Orthonormality: } \|C\|_D^2 &= \|C^T C - Id\|^2 \end{aligned} \tag{3}$$

where $\Lambda, \widetilde{\Lambda} \in \mathbb{R}^{k \times k}$ are diagonal matrices storing the first k eigenvalues calculated on the input and output meshes respectively. Here, the Laplacian commutativity norm is originated from the idea that the eigenvalues of the Laplacian should not change between the original and the simplified meshes. On the other hand, the orthonormality norm is used as a quantitative measure to tell how close the functional map to the identity matrix. Therefore, for these norms, the closer the values are to zero, the better the spectrum is preserved throughout the simplification.

4.2 Partition Size

In Figure 6, going from left to right, the number of partitions is decreased, so the average partition size is increased. As the partition size is increased, the partitions become more and more capable of representing the overall neighborhood. Thus, it can be observed that both of the norms become smaller, leading to a better spectrum preservation. For each mesh, partition sizes should be adjusted so that the timing improvements are sufficient to compensate the loss in the spectrum preservation.

4.3 Number of Eigenvalues

The spectrum preservation results obtained by employing different number of eigenvalues to preserve are presented in Figure 7. Both the Laplacian commutativity and the orthonormality norms tend to decrease, as the number of eigenvalues to preserve is increased. Because, increasing the number of eigenvalues results with a better spectrum preservation for the higher frequency components. However, this only holds up to a certain point. After that, it only introduces additional computational cost to the method.

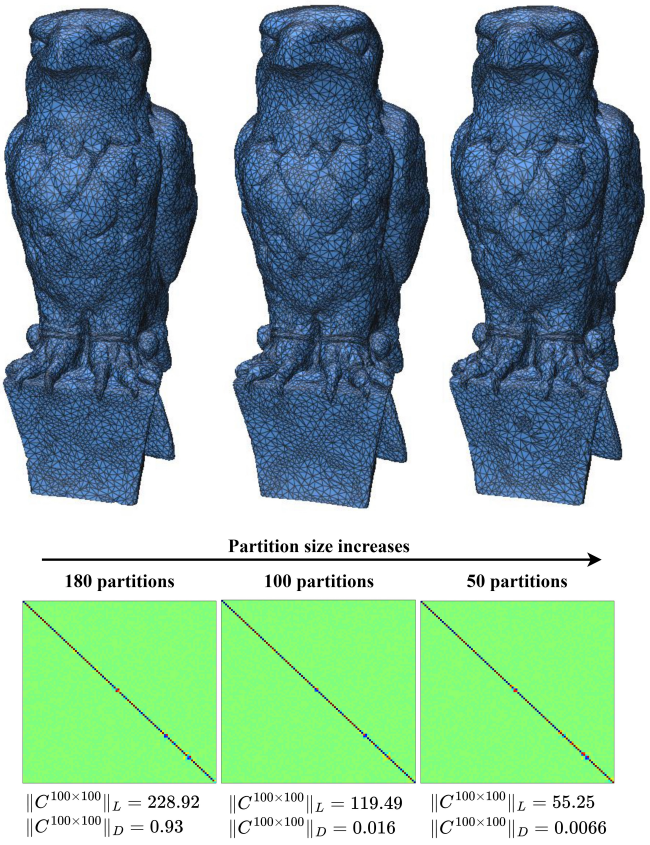


Fig. 6. As the partition size is increased (the number of partitions is decreased), our spectrum preservation becomes more effective.

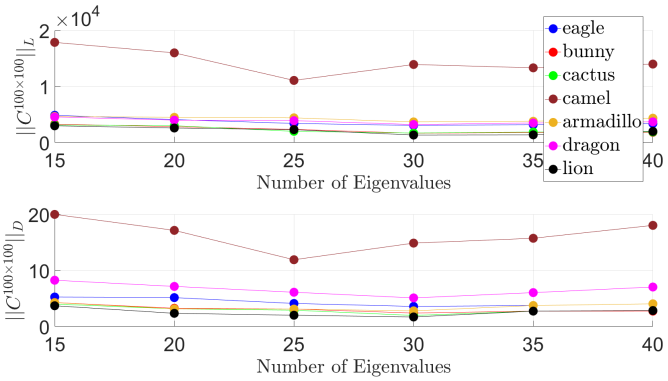


Fig. 7. As the number of eigenvalues preserved increases, the spectrum of the Laplacian operator is better preserved, but up to a point. Plots are shown for seven different models using two spectral norms.

4.4 Number of Similar Edge Collapses

In Figure 8, the spectrum preservation results obtained by employing various numbers of similar edge collapses are presented. As the number of similar edge collapses is increased, both of the norms also increase. It can also be seen that the bottom right parts of the functional maps which correspond to the high-frequency eigenvalues start to scatter, indicating that the capability to preserve the spectrum at the higher frequencies decrease. This is because as the number of similar edge collapses increases, more edges are chosen among the same restricted subset of edges.

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

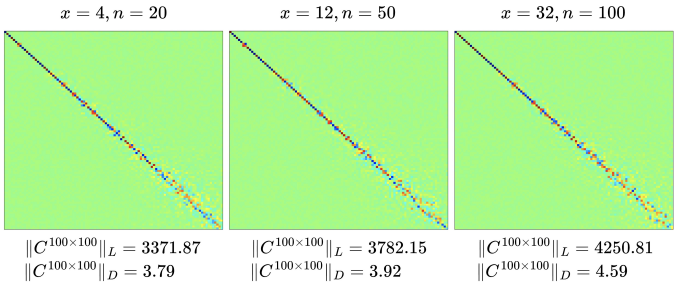


Fig. 8. x is the number of similar edges to collapse, while n is the size of the edge set for choosing similar edges. Here, from left to right, the number of similar edge collapses is increased, leading to a worse spectrum preservation between the input and output meshes.

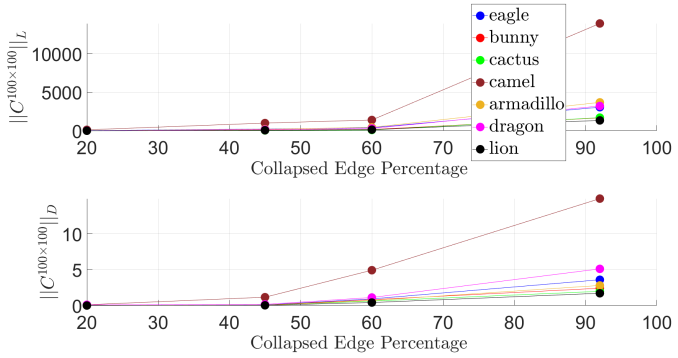


Fig. 9. Relative errors with different percentages of edge collapses.

The mesh, however, gets modified with each edge collapse, while the edges in the restricted subset were selected as the best choices for an older state of the mesh. At some point, the edges outside that subset may be preserving the spectrum more than the edges in the subset, however they are left as not considered. Therefore, it is a better practice not to keep the number of similar edge collapses too large.

We also show the way commutativity and orthonormality errors behave as number of collapses changes in Figure 9, where we observe significant increments after around 60% of collapses.

4.5 Comparisons

The proposed method is evaluated by comparing against some of the existing spectrum-preserving methods such as spectral coarsening [2], spectral simplification [3] and chordal coarsening [4]. For fair comparison, all of the methods are configured to preserve the first 15 components of the spectrum along with ours. It should be noted that our method and the spectral mesh simplification method proposed by Lescoat et al. [3] produce a coarsened mesh as output. However, the methods proposed by Liu et al. [2] and Chen et al. [4] do not produce an output mesh. Therefore, for their methods, the vertices that are selected to exist on the coarsened domain are visualized by marking the vertices over the surface of the original mesh.

To make the quantitative results comparable, before obtaining the results, all the meshes are scaled so that their surface areas correspond to unit area. For the mass matrices M and $\widetilde{M}$, it is ensured that $tr(M) = tr(\widetilde{M}) = 1$.

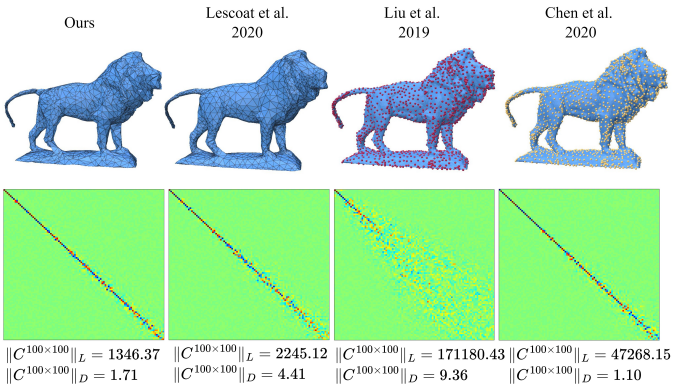


Fig. 10. With each method, lion mesh is simplified from 20212 to 2000 vertices (10% of its initial size).

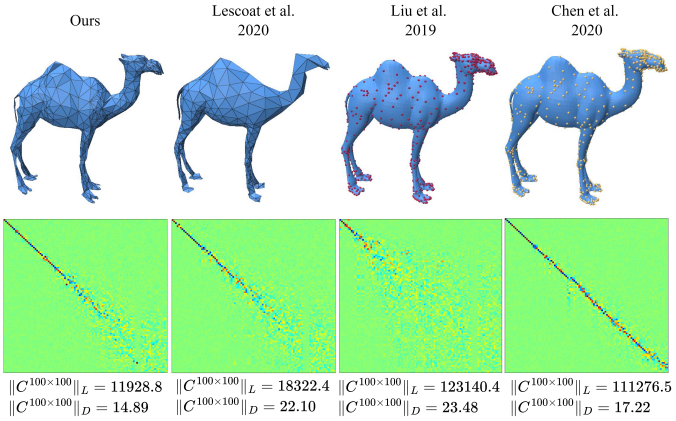


Fig. 11. With each method, camel mesh is simplified from 9757 to 800 vertices (8% of its initial size).

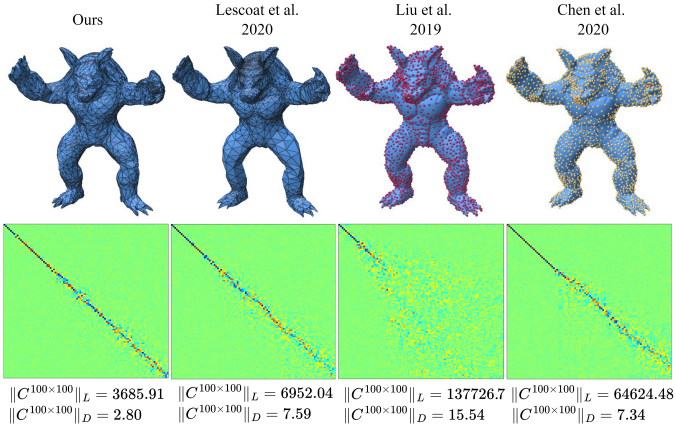


Fig. 12. With each method, armadillo mesh is simplified from 49990 to 3000 vertices (6% of its initial size).

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

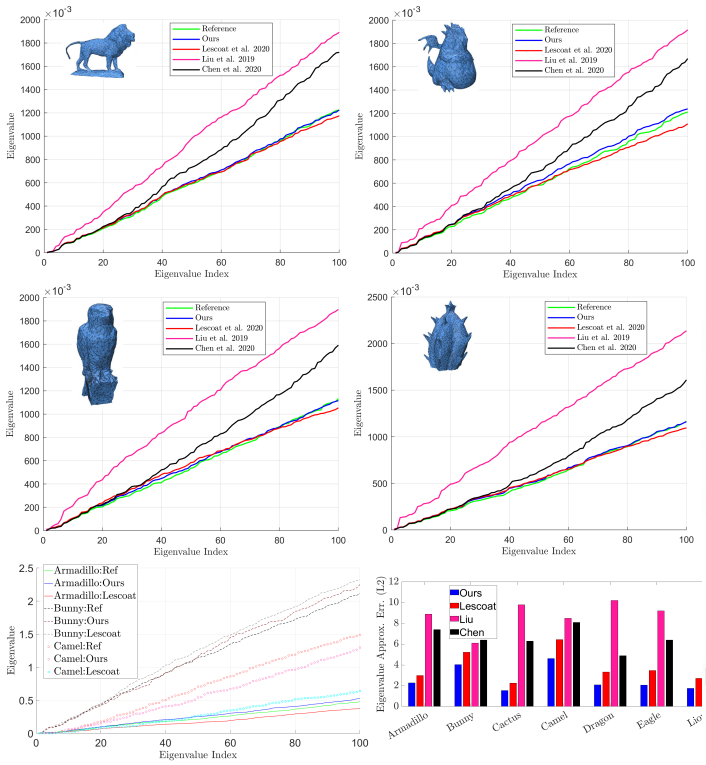


Fig. 13. The first 100 eigenvalues of the Laplacian operator obtained on the original reference mesh and its coarsened versions (top two rows). Similar plotting for the rest of the models using the best-performing competitor only (last row left). 100 eigenvalues are compared under L2 norm to obtain the bar chart for each model and method (last row right).

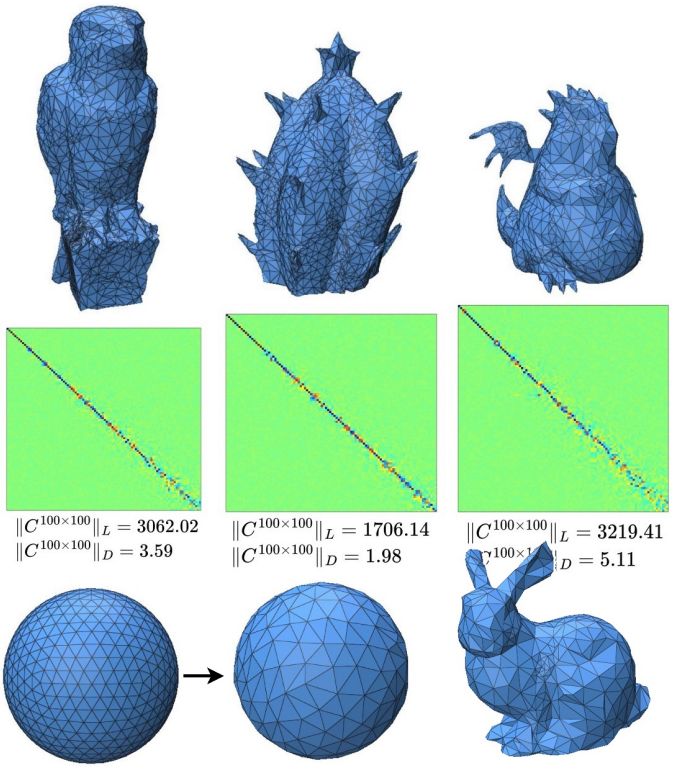
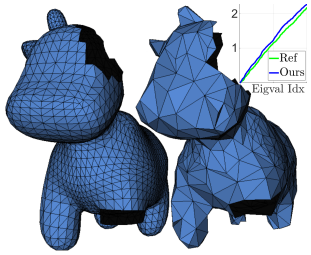


Fig. 14. Meshes simplified by our method to 8% of initial sizes.

Figures 10, 11 and 12 show that our method usually outperforms the method of Liu et al. [2], while considering a small number of eigenvalues. It performs somewhere between the method of Lescoat et al. [3] and Chen et al. [4] in terms of functional map visualizations. We outperform all of the competitors in terms of direct eigenvalue comparisons as we produce plots that are closest to the reference plots (Figure 13). When only the functional map visualizations are considered, for the majority of the cases, the method of Chen et al. [4] preserves the spectrum better than all other methods including ours, since they resemble the identity matrix more. However, it should be noted that the functional map visualizations represent only the eigenvector preservation, they do not stand for the eigenvalues. When the norms are examined, it can be seen that our Laplacian commutativity norm is much smaller. This is because although the method of Chen et al. [4] manages to preserve the first 15 eigenvalues, it fails to preserve the eigenvalues beyond that point, as can be seen in Figure 13.

When we investigate Figures 10, 11 and 12 further, we observe that our appearance visualizations are already better than [3] as we preserve head and toe features but [3] cannot. Other competitors [2], [4] do not produce a mesh at all. Our matrix visualization, on the other hand, is worse than [4] only and for the Camel model only. We still have a better L norm here though because L norm includes eigenvalues that we preserve much better than [4].



The main advantage of our method over the existing ones is to be able to preserve the spectrum by focusing on a small number of eigenvalues such as 15, while outputting a simplified mesh. Our method focuses on preserving the eigenvalues directly, whereas other existing methods carry out this implicitly by preserving the eigenvectors. We provide further results in Figure 14. Note also that our algorithm is insensitive to sharp creases such as the ones in the base of eagle (Figures 6 and 14) and the cuboid (Figure 15). We also maintain our stability on non-watertight meshes with boundaries as depicted in the inset.

4.6 Validation through Applications

We validate the spectrum preservation capability of our method further by comparing our spectrum with the original over various applications that use eigenvalues and/or eigenvectors.

4.6.1 *Only Eigenvalues*

The set of Laplacian eigenvalues has been proved useful as a discriminative global shape descriptor called Shape-DNA [8]. Using Multidimensional Scaling (MDS), we convert the Shape-DNA descriptor distances into Euclidean distances in order to visualize where each model lands in a 2D coordinate system (Figure 15). Specifically, we use the first two principal components of the autocorrelation matrix AA^T where $A_{ij} = e^{-d_{ij}^2}$ is the Gaussian affinity matrix of Shape-DNA distances.

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

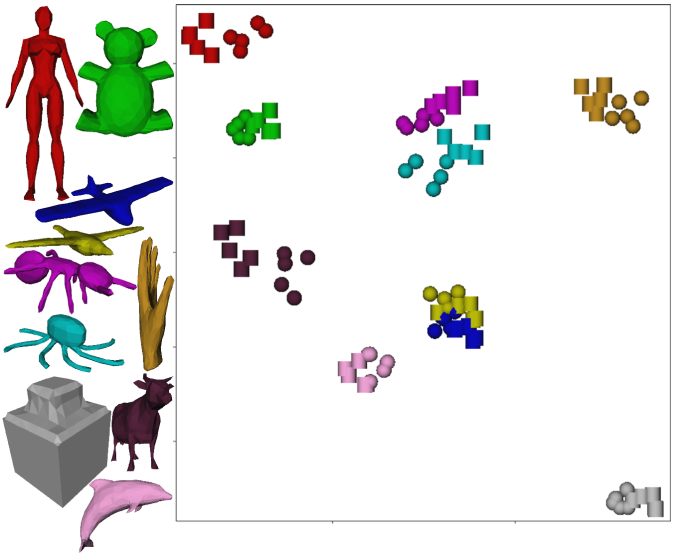


Fig. 15. 2D MDS embeddings based on our eigenvalues from the simplified models (spheres) and ground-truth eigenvalues from the original models (cylinders). Five models from each class are in use. Representative models simplified to about 1K vertices by our method are shown at left with matching colors.

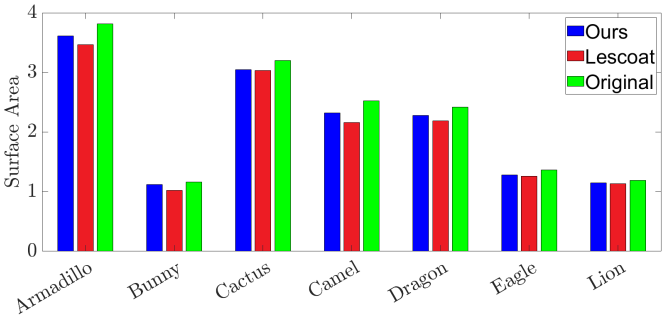


Fig. 16. Surface areas of our simplification results (blue) are always closer to the original surface areas (green) than that of [3] (red).

Although extraction of surface areas from the Laplacian eigenvalues is possible, e.g., through Eq. 41 in [8], this operation requires extremely high-resolution discretization, that defeats our purpose, and very accurate eigenvalues close to the real analytical ones, possibly computed with a cubature integration approach over triangles in parameter space. We, therefore, instead merely compared the total surface areas of the original models and its simplified versions by our method and [3], as eigenvalues are provably related to surface area (Figure 16). Note that other competitors [2] and [4] are not able to produce a surface mesh and therefore not eligible for this comparison. Also, area discrepancy between the original and simplified [38] models has never exceeded 6%, implying satisfactory area preservation by our method.

4.6.2 Only Eigenvectors

Eigenvalues are provable related to min-cuts as well, e.g., the second smallest Laplacian eigenvalue reflects how well connected the overall mesh is, aka algebraic connectivity, and subsequently leads to a robust binary spectral partitioning. Namely, negative entries in the corresponding eigenvector, aka Fiedler vector, gives one partition and positives the other. Obtaining a partitioning similar to the original one is an evidence of our spectrum preservation at this frequency during simplification (Figure 17-top left).

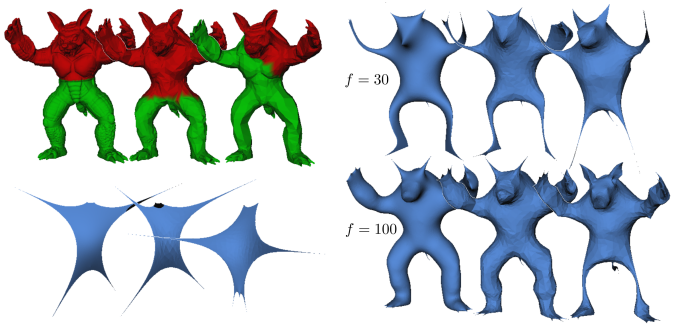


Fig. 17. Eigenvectors from the original spectrum (left), our spectrum (middle), and [3] spectrum (right) are used for clustering (top left), Laplacian Eigenmap (bottom left), and Laplacian embedding (right). Note that our results are always more similar to the originals than [3].

One can also transform the mesh to a different domain by directly using the second, third, and fourth smallest eigenvectors' entries as coordinates, i.e. Laplacian Eigenmap (Figure 17-bottom left), or by projecting the mesh coordinates onto the first f Laplacian eigenvectors (Figure 17-right). The latter can be seen as applying a low-pass filter to the mesh coordinate signal X, i.e., $X' = \Phi_{1..f}\Phi_{1..f}^T X$ where X' is the vector of transformed coordinates.

4.6.3 Both Eigenvalues and Eigenvectors

Laplacian eigenvalues λ_i are also coupled with their corresponding eigenvectors ϕ_i to reveal mesh properties that are not apparent from the edge structure. To this end, isometry-invariant heat kernel descriptor [28] $k_\tau(p,r) = \sum_{i=1}^{f} e^{-\tau\lambda_i}\phi_i(p)\phi_i(r)$ and biharmonic distance [40] $d(p,r) = \sum_{i=1}^{f}(\phi_i(p) - \phi_i(r))^2/\lambda_i^2$ are employed frequently in isometric shape matching. Obtaining values similar to the ones computed on the original models is yet another indication of our valid spectrum preservation (Figure 18-a and c). Similarly, local extrema of the function $g(p) = \sum_{i=1}^{f} \frac{1}{\sqrt{\lambda_i}} \frac{|\phi_i(p)|}{||\phi_i||_{L^\infty}}$ is known to provide consistent sampling between pairs of shapes to be matched [12], [13] (Figure 18-b).

Bunny mesh in Figure 18 is simplified by our method to 17% of its initial size, while the others are simplified to 8%. We observe that even with a high reduction ratio, the descriptor [28], distance [40], and sampling [13] almost stay the same, which implies the spectrum preservation between the original and the simplified meshes.

4.7 Timing

4.7.1 Partition Size

As the mesh is divided into more partitions, the size of each partition gets smaller. Consequently, with each partition having fewer vertices, the size of the Laplacian operators constructed for each partition are reduced, leading to faster eigenvalue computations. Besides, since the partitions are run concurrently via threading, as the number of partitions increases, the time it takes to simplify the overall mesh

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

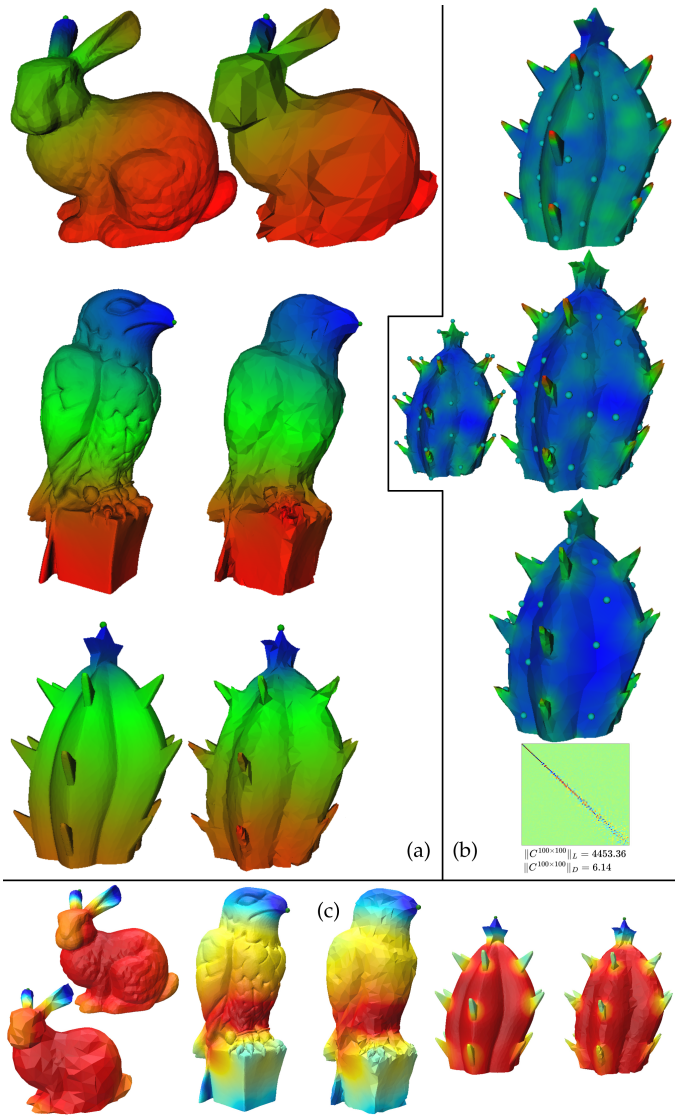


Fig. 18. Heat kernel descriptor (c) and biharmonic distance (a) are visualized with respect to a reference vertex (green) on the original meshes and our simplified versions. In (b), sampling in the original model (row 1) remains more consistent after our simplification (row 2) than that of [3] (row 3 along with their matrix). Local maxima and minima sampling detects the tips (downscaled cactus) and centers, respectively.

decreases. Figure 19 shows how the simplification time changes with respect to the average partition size employed.

4.7.2 Number of Eigenvalues

Timing results obtained by focusing on preserving different number of eigenvalues are provided in Figure 20. As the number of eigenvalues to preserve is increased, the eigenvalue decompositions performed for the Laplacian operator at each step of the algorithm takes longer time than before. Consequently, the overall simplification time increases.

4.7.3 Number of Similar Edge Collapses

It is observed that enabling similar edge collapses by selecting 4 edges out of 20 candidates, reduces the simplification time to approximately one fifth of the previous timing, while affecting the spectrum as little as possible.

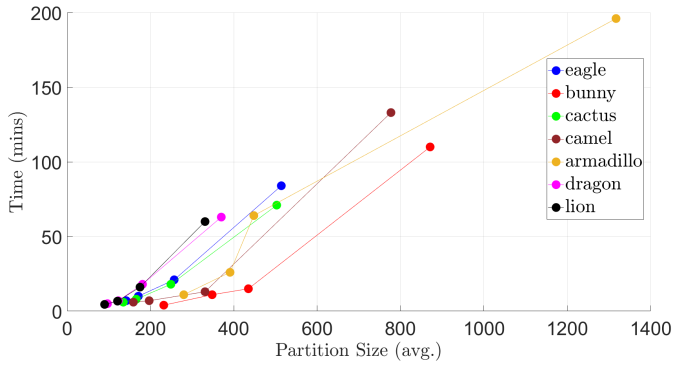


Fig. 19. The overall simplification time increases, as the average partition size is increased (the number of partitions is decreased), as shown on seven different models whose sizes are given in Table 2.

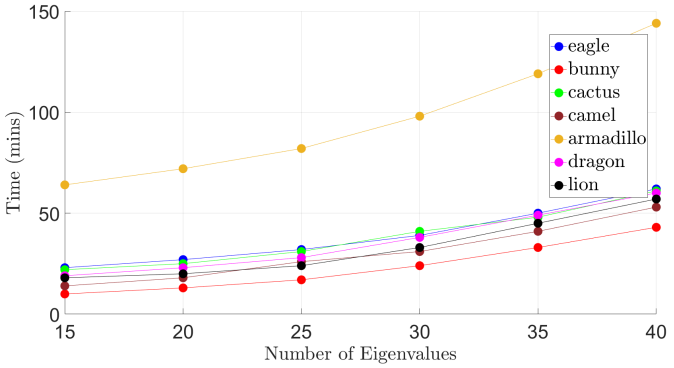


Fig. 20. As the number of eigenvalues to preserve is increased, the overall simplification time also increases, as shown on seven different models.

4.7.4 Threading

Timing results obtained by enabling and disabling threading are provided in Table 2. Here, the number of threads equals the number of partitions that the mesh is divided into and approximately 92% of the vertices are removed from each mesh. It can be observed that threading provides a significant reduction in simplification times. However, since not all of the threads are run at the same time, the acceleration amount is not directly the same as the number of threads (partitions).

TABLE 2
Timings obtained with and without threading on 7 models. Corresponding minutes for the fastest competitor [3] are 0.06, 0.17, 0.68, 0.71, 3.08, 0.32, and 0.29, from top to bottom.

Mesh	Number of Vertices	Number of Threads	Time with Threading (mins)	Time without Threading (mins)
bunny	3485	15	9	18
camel	9757	48	13	40
cactus	25131	180	17	75
eagle	25727	180	18	76
armadillo	49990	200	63	311
dragon	20741	160	15	46
lion	20212	160	14	43

Although there is no linear relationship between the total number of partitions and the amount of acceleration,

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

contributions of threading become more noticeable for the meshes with more partitions. The reason behind this can be visualized with a plot as in Figure 21. Green, blue and red lines respectively indicate the minimum, average and maximum number of threads existing in a batch, for the given number of partitions. It can be observed that when there are more partitions in a mesh, the average number of threads per batch gets increased as well, leading to a higher amount of parallelization. For instance, if there are 8 partitions in a mesh, the number of threads in each batch will be between 2 and 3. On the other hand, if there are 100 partitions in a mesh, the number of threads per batch will range from 11 to 24, which results with a more remarkable timing reduction.

To demonstrate our high parallelizability further, we distribute the independent partition simplification tasks within a batch onto the 6-node cluster of our department[1] where each node has two Intel Xeon 10-core 2.40GHz CPUs. Each node simplifies the part of the original mesh that corresponds to its assigned partitions in a physically parallel fashion. We achieve about seven times speedup (Table 3) compared to Table 2 where we used our original 4-core 2.60GHz Intel PC. The main reason of this acceleration is the elimination of context switches and interleaving of about 30 ($= 4 \times 7 + 2$, requiring about 7 passes) independent threads over the four cores of a single PC. With a cluster of computer nodes, we are able to run all of these threads simultaneously, without any interleaving that gives the illusion of concurrency. Note that, Bunny model does not improve at all as our original 4-core PC is already capable of allocating 1 core to each of the 4 concurrent threads. Finally note that, although the workload within a thread batch is embarrassingly parallel, batches must still be processed sequentially in our current architecture. Further speedup is therefore possible by parallelizing the thread batches with the appropriate synchronization control strategies.

TABLE 3
Timings obtained with a cluster of computers. NumCThreads is short for the average number of concurrently running threads.

	Bun	Cam	Cac	Eag	Arm	Dra	Lio
NumCThreads	4	10	30	30	32	30	30
Time (mins)	9.1	6.7	2.5	2.7	7.9	2.2	2.1

5 Conclusion and Future Work

In this paper, we presented a mesh simplification method, which focuses on preserving the spectral properties of the input mesh as much as possible. Our method is the first to address direct eigenvalue preservation during simplification. The main idea lying at the core of the algorithm is comparing the eigenvalues of the Laplacian operators constructed on the mesh at different steps, to decide which edges to collapse. In order to reduce the computational cost caused by the eigenvalue decompositions carried out in the algorithm, we introduced partitioning, threading and an approach based on eliminating similar edges. We evaluated our algorithm on a variety of meshes and demonstrated that our method is capable of maintaining the spectrum at least as effectively as the existing methods, by only considering a small number of eigenvalues. We also demonstrated that the coarsened meshes produced by the algorithm are still able to represent various properties similar to the way they were represented on the original mesh. Namely, we showed our performance on heat kernel and Shape-DNA descriptors, biharmonic distances, consistent sampling, spectral partitioning, Laplacian Eigenmap, and Laplacian embedding.

1. https://ceng.metu.edu.tr/bigdatalab

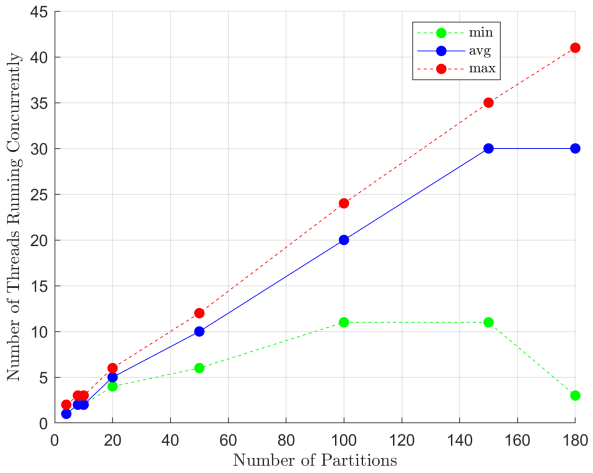


Fig. 21. The number of threads (partitions) running at the same time for various number of partitions.

As stated in Chen et al. [4], simplification methods can either preserve the appearance [1] or the spectrum [2], [3], but not both. In this regard, our spectrum-preserving method is at least as effective as our recent state-of-the-art competitors [2], [3], yet the third state-of-the-art competitor [4] outperforms every other method in terms of eigenvector preservation at the expense of first obtaining a coarsened mesh using a third-party appearance-preserving simplification method. Focusing and relying solely on eigenvalues of cotangent Laplacians, our method clearly outperforms the others in terms of eigenvalue preservation as shown by the eigenvalue approximation plots and errors (Figure 13).

The main limitation of our algorithm is about efficiency, since it requires the computation of the eigenvalues from scratch for each edge collapse that is considered. One way to overcome this problem would be adapting the classic priority queue method introduced by Garland and Heckbert [1] with the presented cost metric, just as Lescoat et al. [3] did. However, to do this, an algebraic update rule should be found such that an edge collapse can be simulated without requiring the computation of the eigenvalues from scratch. An alternative future work is to investigate ways to approximate eigenvalues (vibration frequencies) [41], which is all one needs to guide our method or any other future simplification method that addresses direct eigenvalue preservation. Note that, it is always possible to first reduce the input size via QSLIM software [1], whose impact on the spectrum would be limited, as stated by [3]. Note also that, we did not take advantage of GPUs in the unoptimized implementation of our parallel algorithm.

As a final note, our method is a greedy algorithm, so it does not guarantee that the set of edge collapses performed leads to the optimal preservation of the spectrum. Yet, we believe that our naïve way of utilizing the 1D eigenvalues

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Visualization and Computer Graphics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2023.3341610

of the Laplacian may establish a ground for more advanced future works, whose main focus is spectrum preservation.

ACKNOWLEDGMENTS

Supported by TUBITAK under the project EEEAG-119E572.

REFERENCES

[1] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proc. of the 24th conference on Computer graphics and interactive techniques*, 1997, pp. 209–216.

[2] H.-T. D. Liu, A. Jacobson, and M. Ovsjanikov, "Spectral coarsening of geometric operators," *ACM Transactions on Graphics*, vol. 38, no. 4, pp. 1–13, 2019.

[3] T. Lescoat, H.-T. D. Liu, J.-M. Thiery, A. Jacobson, T. Boubekeur, and M. Ovsjanikov, "Spectral mesh simplification," in *Computer Graphics Forum*, vol. 39, no. 2, 2020, pp. 315–324.

[4] H. Chen, H.-T. D. Liu, A. Jacobson, and D. Levin, "Chordal decomposition for spectral coarsening," *ACM Transactions on Graphics*, vol. 39, no. 6, pp. 1–16, 2020.

[5] L. Cosmo, M. Panine, A. Rampini, M. Ovsjanikov, M. M. Bronstein, and E. Rodolà, "Isospectralization, or how to hear shape, style, and correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7529–7538.

[6] R. Rustamov, "Laplace-beltrami eigenfunctions for deformation invariant shape representation," in *Symposium on geometry processing*, vol. 257, 2007, pp. 225–233.

[7] V. Jain and H. Zhang, "A spectral approach to shape-based retrieval of articulated 3d models," *Computer-Aided Design*, vol. 39, no. 5, pp. 398–407, 2007.

[8] M. Reuter, F.-E. Wolter, and N. Peinecke, "Laplace–beltrami spectra as 'shape-dna'of surfaces and solids," *Computer-Aided Design*, vol. 38, no. 4, pp. 342–366, 2006.

[9] J. Hu, H. Hamidian, Z. Zhong, and J. Hua, "Visualizing shape deformations with variation of geometric spectrum," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 721–730, 2016.

[10] Y. Sahillioğlu and D. Horsman, "Augmented paths and reodesics for topologically-stable matching," *ACM Transactions on Graphics*, vol. 42, no. 2, pp. 1–15, 2022.

[11] A. Dubrovina and R. Kimmel, "Approximately isometric shape correspondence by matching pointwise spectral features and global geodesic structures," *Advances in Adaptive Data Analysis*, vol. 3, no. 01n02, pp. 203–228, 2011.

[12] M. Edelstein, D. Ezuz, and M. Ben-Chen, "Enigma: Evolutionary non-isometric geometry matching," *ACM Transactions on Graphics*, vol. 39, no. 4, 2020.

[13] X. Cheng, G. Mishne, and S. Steinerberger, "The geometry of nodal sets and outlier detection," *Journal of Number Theory*, vol. 185, pp. 48–64, 2018.

[14] S. C. Schonsheck, M. M. Bronstein, and R. Lai, "Nonisometric surface registration via conformal laplace–beltrami basis pursuit," *Journal of Scientific Computing*, vol. 86, pp. 1–24, 2021.

[15] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle, "Mesh optimization," in *Proc. of the 20th conference on Computer graphics and interactive techniques*, 1993, pp. 19–26.

[16] W. J. Schroeder, J. A. Zarge, and W. E. Lorensen, "Decimation of triangle meshes," in *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, 1992, pp. 65–70.

[17] M. Garland and P. S. Heckbert, "Simplifying surfaces with color and texture using quadric error metrics," in *Proceedings Visualization'98 (Cat. No. 98CB36276)*. IEEE, 1998, pp. 263–269.

[18] J. Van, P. Shi, and D. Zhang, "Mesh simplification with hierarchical shape analysis and iterative edge contraction," *IEEE Trans. on visualization and computer graphics*, vol. 10, no. 2, pp. 142–151, 2004.

[19] P. Cignoni, C. Montani, and R. Scopigno, "A comparison of mesh simplification algorithms," *Computers & Graphics*, vol. 22, no. 1, pp. 37–54, 1998.

[20] F. Ponchio, "Multiresolution structures for interactive visualization of very large 3d datasets," Ph.D. dissertation, Zugl.: Clausthal, Techn. Univ., Diss., 2008, 2009.

[21] Y. Sahillioğlu, "Recent advances in shape correspondence," *The Visual Computer*, vol. 36, no. 8, pp. 1705–1721, 2020.

[22] 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.

[23] G. Taubin, "A signal processing approach to fair surface design," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995, pp. 351–358.

[24] Z. Karni and C. Gotsman, "Spectral compression of mesh geometry," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 279–286.

[25] R. Liu and H. Zhang, "Mesh segmentation via spectral embedding and contour analysis," in *Computer Graphics Forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 385–394.

[26] J. Ren, S. Melzi, P. Wonka, and M. Ovsjanikov, "Discrete optimization for shape matching," in *Computer Graphics Forum*, vol. 40, no. 5. Wiley Online Library, 2021, pp. 81–96.

[27] M. Reuter, F.-E. Wolter, and N. Peinecke, "Laplace-spectra as fingerprints for shape matching," in *Proceedings of the 2005 ACM symposium on Solid and physical modeling*, 2005, pp. 101–106.

[28] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," in *Computer Graphics Forum*, vol. 28, no. 5, 2009, pp. 1383–1392.

[29] M. Ovsjanikov, J. Sun, and L. Guibas, "Global intrinsic symmetries of shapes," in *Computer graphics forum*, vol. 27, no. 5. Wiley Online Library, 2008, pp. 1341–1348.

[30] A. C. Öztireli, M. Alexa, and M. Gross, "Spectral sampling of manifolds," *ACM Trans. on Graphics*, vol. 29, no. 6, pp. 1–8, 2010.

[31] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas, "Functional maps: a flexible representation of maps between shapes," *Trans. on Graphics*, vol. 31, no. 4, pp. 1–11, 2012.

[32] A. Keros and K. Subr, "Spectral coarsening with hodge laplacians," in *ACM SIGGRAPH 2023 Conference Proceedings*, 2023, pp. 1–11.

[33] H.-T. D. Liu, M. Gillespie, B. Chislett, N. Sharp, A. Jacobson, and K. Crane, "Surface simplification using intrinsic error metrics," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 1–17, 2023.

[34] Y. Sahillioğlu and Y. Yemez, "Coarse-to-fine combinatorial matching for dense isometric shape correspondence," in *Computer graphics forum*, vol. 30, no. 5. Wiley Online Library, 2011, pp. 1461–1470.

[35] G. Karypis and V. Kumar, "A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices," *University of Minnesota, Department of Computer Science and Engineering, Army HPC Research Center, Minneapolis, MN*, vol. 38, 1998.

[36] D. S. Johnson and M. R. Garey, *Computers and intractability: A guide to the theory of NP-completeness*. WH Freeman, 1979.

[37] D. J. Welsh and M. B. Powell, "An upper bound for the chromatic number of a graph and its application to timetabling problems," *The Computer Journal*, vol. 10, no. 1, pp. 85–86, 1967.

[38] X. Chen, A. Golovinskiy, and T. Funkhouser, "A benchmark for 3D mesh segmentation," *ACM Transactions on Graphics (Proc. SIGGRAPH)*, vol. 28, no. 3, Aug. 2009.

[39] Y. Qiu, "Spectra. a library for large scale eigenvalue problems," https://spectralib.org/, 2018.

[40] Y. Lipman, R. M. Rustamov, and T. A. Funkhouser, "Biharmonic distance," *ACM Trans. on Grap.*, vol. 29, no. 3, pp. 1–11, 2010.

[41] A. Nasikun, C. Brandt, and K. Hildebrandt, "Fast approximation of laplace-beltrami eigenproblems," in *Computer Graphics Forum*, vol. 37, no. 5. Wiley Online Library, 2018, pp. 121–134.

Misranur Yazgan received her BS and MS degrees from the Computer Engineering department of Middle East Technical University, Turkey. She is now working as a software engineer at a startup software company based in Brooklyn, New York.

Yusuf Sahillioğlu is a Professor in the Department of Computer Engineering at Middle East Technical University, Turkey. He is also an associate editor of The Visual Computer journal. His research interests include digital geometry processing and computer graphics. He has a PhD in computer science from Koç University, Turkey. Contact him at ys@ceng.metu.edu.tr or visit http://www.ceng. metu.edu.tr/∼ys.

© 2023 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.