

Laplacian2Mesh: Laplacian-Based Mesh Understanding

Qiujie Dong, Zixiong Wang, Junjie Gao, Shuangmin Chen, Zhenyu Shu, Shiqing Xin*

Abstract—Geometric deep learning has sparked a rising interest in computer graphics to perform shape understanding tasks, such as shape classification and semantic segmentation on three-dimensional (3D) geometric surfaces. Previous works explored the significant direction by defining the operations of convolution and pooling on triangle meshes, but most methods explicitly utilized the graph connection structure of the mesh. Motivated by the geometric spectral surface reconstruction theory, we introduce a novel and flexible convolutional neural network (CNN) model, called Laplacian2Mesh, for 3D triangle mesh, which maps the features of mesh in the Euclidean space to the multi-dimensional Laplacian-Beltrami space, which is similar to the multi-resolution input in 2D CNN. Mesh pooling is applied to expand the receptive field of the network by the multi-space transformation of Laplacian which retains the surface topology, and channel self-attention convolutions are applied in the new space. Since implicitly using the intrinsic geodesic connections of the mesh through the adjacency matrix, we do not consider the number of the neighbors of the vertices, thereby mesh data with different numbers of vertices can be input. Experiments on various learning tasks applied to 3D meshes demonstrate the effectiveness and efficiency of Laplacian2Mesh.

Index Terms—Geometric deep learning, Laplacian-Beltrami space, Implicit geodesic connection, Mesh processing.

1 INTRODUCTION

THE undeniable success of deep learning on 2D computer vision and text has sparked significant attention to its applicability in 3D geometry. As the pioneer of learning a feature extraction method from the 3D geometric data, PointNet [1] directly applies CNNs on the irregular and sparse point cloud representation. Since then many researchers have been inspired to design more effective learning methods [2], [3] to finish the fundamental geometric analysis tasks. Apart from the point cloud representation, or point cloud, for short, various discrete approximations for 3D shapes have been introduced to represent 3D geometric shapes, such as multi-view images [4], voxels [5], meshes [6], etc.

Due to the inherently regular and dense structure of images, the image pixels are sampled by a uniform grid, which reduces the application difficulty of 2D CNNs, resulting in an important role on various vision tasks [7], [8] with large amounts of data. To directly use the advantage of the CNNs in the image domain to work on the 3D geometric data, many efficient methods mapped the 3D shapes to regular forms which include obtaining the multi-view 2D projections [4] by using many cameras to shoot the object in sequence and mapping the 3D shapes to 3D voxel grids [9] which like the pixels in 3D space. Even though the above approaches can directly use the neural networks well-performing on 2D images, the former indirect representation method lacks the global information and the resolution of voxel data is limited due to the curse of

dimensionality.

In this paper, we use the triangle mesh representation, one of the most popular representations for 3D data, to learn a 3D geometric network. Meshes are consist of a set of vertices, edges, and faces, which use a series of triangles in 3D space to approximate the geometric surface. Compared with the point cloud, the mesh has topological structures that can provide more geometric information. Different from the voxels, mesh only represents the surface of the shapes, so it doesn't need large amounts of memory to save wasteful or redundant information from the unoccupied volumetric data. Despite its non-uniform when representing the flat and sharp position in the surface, the mesh representation has a lucrative characteristic which is its native connection property. Because of the irregular structures, we cannot directly define convolution and pooling on 3D geometric data by taking advantage of the mature 2D networks. However, extending the power of CNNs to the 3D domain working on irregular meshes is nontrivial and will provide new windows for us to understand the real world.

To tap into the potential attributes of the mesh data, we propose a novel and fundamentally different deep learning method with supervision, named Laplacian2Mesh. Defining the special neural network, which performs convolution operation and pooling operation on native irregular triangular meshes, is inspired by classic spectral surface reconstruction works [10] which are dominated by the Laplace operator. In general knowledge, the eigenvalues of the Laplace operator are equivalent to the frequency in the signal domain. The arrangement of all eigenvalues together is a characterization of the smoothness of the graph signal, where the eigenvectors corresponding to the smaller eigenvalues are smoother, and vice versa.

Because the neural networks need a consistent input format to work, an issue that cannot be circumvented, some dedicated approaches [6], [11] have been designed to obtain the mesh features with regular forms. Every edge of the mesh is incident to exactly two triangle faces when it is a 2-manifold 3D shape, this

- Q. Dong, Z. Wang, J. Gao, and S. Xin are with the School of Computer Science and Technology, Shandong University, Qingdao, PR China. E-mail: qiujuie.jay.dong@gmail.com, zixiong_wang@outlook.com, gjjsdnu@163.com, xinshiqing@sdu.edu.cn.
- S. Chen is with School of Information and Technology, Qingdao University of Science and Technology, Qingdao, PR China. E-mail: csmqq@163.com.
- Z. Shu is with the School of Computer and Data Engineering, Ningbo Institute of Technology, Zhejiang University, Ningbo PR China. E-mail: shuzhenyu@nit.zju.edu.cn.
- S. Xin is the corresponding author.

Manuscript received April 19, 2005; revised August 26, 2015.



Fig. 1. Reconstructing mesh coordinates using increasing numbers of eigenfunctions. The colors show the error of the reconstructed coordinates to the rightmost original mesh, shades of purple indicating the low error. These numbers below the image indicate the number of eigenvectors used in reconstruction, and as we can see that the reconstructed Homer shape using more than half of the eigenfunctions (over 500 in this image) is very similar to the original mesh with 2000 faces and 1002 vertices.

property can be used to define a natural fixed-sized convolutional neighborhood of four edges. Aiming to deal with the different geometric structures of the vertices of a mesh shape, some researchers proposed to discretize a local tangent plane into angular intervals with fixed numbers. Taking the advantage of the Laplacian spectral clustering method to extend the classic over-segmentation approach to the geometric deep learning domain is also an effective work to handle the above problem.

The most essential reason for considering the number of neighborhoods is that these methods need to utilize the graph connection property of the mesh data. Mentioned above neural networks, however, use the native graph structure explicitly. Instead, we note that the spectral decomposition of the Laplacian-Beltrami operator, simply referred to as the Laplacian operator, is an implicit way to represent the topology characteristic of the triangle meshes. Laplacian2Mesh constructs a novel neural network on the surface mesh, where the channel self-attention convolutions [12] and Laplacian-mesh pooling layers of the architecture operate on the Laplacian space instead of Euclidean space. Benefiting from this spatial transformation by spectral bases of the Laplacian, we do not need to design a module to obtain the mesh elements (faces, edges, or vertices) with an equal number of neighbors, or even we needn't the input meshes have the same data dimension because Laplacian2Mesh uses the Laplacian eigenvectors corresponding to the N smallest eigenvalues to project the features of mesh data in the 3D domain to the N -dimensional Laplacian space, where the eigenvalues are in ascending order and the first is discarded. In the new space, the Laplacian-mesh has the same feature dimensions. As a subsidiary advantage of this operation, we can realize multi-resolution modeling by choosing multiple different N values.

Furthermore, in our Laplacian2Mesh neural network, a key point is the Laplacian-mesh pooling operation which operated on the regular structures in our well-defined space. Pooling, downsampling the features of the shapes, is a significant layer in the CNNs, which focus on the more informative mesh elements. The essence of the pooling layer in the network is dimensionality reduction, which is used to reduce the feature dimension of the output of the convolutional layer. The advantage of the operation is to reduce the number of network parameters and prevent overfitting. Analyzing the pooling from the network's perspective, these layers can increase the receptive field of the network, thereby learning the global features of the shape. We observed and found that the well-known spectral surface reconstruction [10] can show the effect of the mesh simplification, this reconstruction technique removes the high-frequency terms and keeps the topology structure. To take advantage of this property, we construct the mesh pooling layers by the Laplacian multi-space transformation, these layers do not require complicated human design. For the shape understanding

tasks, we prefer the low-frequency signs because the perturbation occasionally occurs in the high-frequency terms, wherefore the mesh pooling in Laplacian2Mesh can retain more effective mesh information while reducing the data dimension.

Although Laplacian2Mesh utilizes the property of mesh connection implicitly, the natural potential of this geometric structure has not been fully tapped. To make full use of this connection structure, we define a new loss function called adjacency-loss using the adjacency matrices which represent the connection relationship of the vertices in the mesh, the loss function urges adjacent points on the Euclidean distance to tend to the same category.

Laplacian2Mesh illustrates its aptitude and can be utilized to address a variety of mesh understanding tasks. We perform experiments on common datasets and highly non-uniform meshes and achieve better performance in two basic applications: mesh classification and mesh semantic segmentation. The code will be open source after the manuscript is accepted.

In summary, the main contributions of our method are as follows:

- Laplacian2Mesh, a general geometric neural network on meshes in Laplacian space, processing on irregular input data with pyramid structure;
- a re-think for taking advantage of the connection structure of meshes, utilizing the property in implicitly way, thereby ignoring the number of neighborhoods of every vertex; simple strategies for mesh pooling in the projection space by the multi-space transformation of Laplacian;
- a new loss–adjacency-loss with clustering effect is defined, along with the dihedral angles on the vertices;
- experiments demonstrating the effectiveness of Laplacian2Mesh in mesh understanding tasks: 3D shape classification and semantic segmentation.

2 RELATED WORK

In recent years, geometric deep learning [13] is becoming a high-profile subfield of machine learning and graphics. This has inspired researchers to find more powerful methods in this discipline. Nevertheless, we only summarize works directly related to our task of learning from 3D shapes in this section, and readers are referred to recent surveys [14], [15] for a comprehensive review of 3D geometric deep learning.

2.1 Geometric Deep Learning on Meshes

The main challenge of constructing a neural network on 3D shapes is how to describe the 3D objects such that these have regular structures adapting to the deep learning frameworks. Hereafter

we provide a brief introduction to the main approaches which are committed to solving the problem of ill-defined fundamental neural network operations in 3D meshes.

Volumetric Convolution. Just like the pixels are the element of 2D images, one idea to utilize the 2D CNNs on 3D shapes is to represent a 3D shape using the binary voxel that its value is 1 if it is inside a shape, 0 otherwise. As such, it is straightforward to extend 2D grids to voxels, thus the common image-based CNNs can be transferred to the 3D domain. Wu et al. [9] first proposed a network processing the volumetric data, which focuses on the 3D shape classification and retrieval tasks. Following that, many papers are proposed for a variety of geometric tasks, such as shape reconstruction [16], semantic shape segmentation [17], and shape alignment [18].

The primary drawback of volumetric grids, however, is their complexity increases cubically with the number of voxels, which limit the high-resolution of 3D shapes. To alleviate the above problem, many approaches [19], [20] with acceleration strategies utilize the sparsity of voxel occupancy to save the computation cost, which spatial hierarchy is close to the multi-resolution input in our approach. In addition, using this representation like stair will cause it to be less appropriate to extract some important features, such as curvature, in the 3D surfaces.

Global Parameterization. Surface parameterization is a concurring direction for using the 2D CNNs on 3D geometric data directly; some methods turn the irregular meshes data into regular images, wherefore the inputs with high-resolution can be easily handled. Sinha et al. [21] define the technique of surface parameterization, and then Maron et al. [22] and Haim et al. [23] extend this approach to general scenarios. Unlike our approach, this representation introduces some Unexpected scenes, such as orientations, angle distortion, and cut seams, such that the prediction of the network depends on the quality of the global parameterization.

Rendering-based techniques are a simple parameterization onto the image domain. One of the main approaches in this representation is the multi-view 2D projections [4] which is a bridge to process the 3D geometric data utilizing existing techniques and architectures from mature 2D neural networks, each of the methods using the representation renders 3D shapes into multiple 2D projections from an ordered set of viewpoints. Furthermore, it enables to fine-tune model that has been trained on existing large image datasets to process the rendered images. Su et al. [4] pioneered this approach under the CNN framework for the task of shape classification, however, multi-view CNNs lose some geometric properties makes them hard to perform some tasks that need local features of shapes. Then Le et al. [24] tackled semantic segmentation using a multi-view recurrent neural network (RNN) with conditional random field (CRF), which unlock possibilities in using 2D networks for local surface processing tasks. Even though these approaches provide a convenient way to process the 3D shapes, they do not tackle the issue of self-occlusions.

Local Parameterization. When the pending shape contains high-frequency information, global parameterization cannot avoid either high distortion or cut seams. Remarkably, many mesh-based learning methods construct the local relationship between the elements of meshes and their neighborhoods such as vertices and their k-ring neighborhood by local parameterization which can lead to much less distortion compared to global parameterization. Masci et al. [25] were the first to apply the local parameterization to define a geodesic convolutional neural network on non-Euclidean

manifolds, which flattens each of the patches with small curves on the surface to the 2D plane. He et al. [11] define a directional curvature along a tangent vector to obtain neighbor information with a consistent structure.

An active challenge to date, however, is how to orient the convolution kernel for this approach. To overcome the issue, researchers provided some choices, such as learning pseudo-coordinate functions described with gaussian mixture models [26], dynamically computing the weights from neighborhood features or weighting based on normal vector similarity [27]. Since considering the local information by adjacency-loss and using the mesh structure in an implicitly way, our approach is free of the above issues.

In addition, defining the localized convolution kernels is necessary for this representation; strategies include localizing spectral filters [28], utilizing the classic geometric approaches (e.g., B-splines [29], wavelets [30], and extrinsic Euclidean convolution [31]). The methods with local parameterization also introduce a drawback that only the small receptive field is applied to aggregate the information from the surface meshes, wherefore the performances of these approaches depend on the mesh resolution of the 3D objects.

Graph Convolution. We know that the intrinsic geodesic connections of triangle meshes can be seen as a graph with a specific structure, which inspired some workers to engineer the learning algorithms for triangle meshes by borrowing from the graph neural networks (GNNs) which are always used to deal with the geometric information. Compared to our using implicitly graph connection, most approaches consider the regular graph structure one of which can use graph convolution to process the surface meshes. For example, MeshWalker [32] breaks the custom of constructing the regular neighborhood structures, which uses random walks [33] along edges to get the information of 3D shapes. While a surface mesh can be viewed as a specific graph and graph convolutions can directly be worked on the meshes, this neglects the natural potential that the meshes have more structured information than the graph in other domains.

Triangular Meshes-Specific Methods. To leverage the structure information of the surface mesh, some learning methods design mesh-specific operations for convolution layers and pooling layers of the neural networks, these directly define the special convolutions on the surface meshes. MeshCNN [6] learns an order invariant convolution on edge of meshes and proposes an edge collapse-based mesh pooling operation. Using a primal graph and dual graph to capture the adjacency relationship of triangle meshed, PD-MeshNet [34] performs convolution operation using graph attention network and defines a pooling operation based on the mesh simplification technique.

In addition, each face of meshes also has 3 faces as its neighborhood when the meshes have 2-manifold structures. Similar to the methods using the features from edges, face-based approaches focus on aggregating information between neighboring faces efficiently and effectively. SubdivNet [35] offers a more general and standard convolution directly defined on meshes and provides uniform downsampling inspired by loop subdivision operation.

2.2 Laplacian-based Approaches.

The Laplacian operator plays a central role in geometry processing and shapes analysis, applied to tasks including symmetry detection [36], correspondence [37], shape recognition [38], and shape

retrieval [39]—among countless others. It is interesting to note the fact that small perturbations to the low-frequency components of a geometric signal are less perceptible than corresponding perturbations to high-frequency components. Due to this solution to the Laplacian system being robust to noises and sampling bias, this operator is ubiquitous in spectral geometry processing related to our work such as the Heat Kernel Signature (HKS) [40]. A comprehensive survey [41] for spectral shape understanding was provided by Wang and Solomin, which highlighted key theoretical properties as well as their numerical discretizations.

Graph Laplacian The important concept must be mentioned in this representation is a spectral geometry – graph Laplacian, which acts on the per-vertex and its neighboring vertices in the meshes. Curvature Laplacian [42] proposed by Liu and Zhang is the earliest work applying graph Laplacian to define a spectral approach to the task of shape understanding. Bruna et al. [43] define the spectral networks that project the per-vertex features onto a spectral basis by a learned linear model. Qiao et al. [44] define the Laplacian pooling layers using Laplacian spectral clustering. Despite graph Laplacians being favored for their simplicity, the drawbacks of failing to converge to a continuous operator that captures the geometry of the surface can lead to undesirable behavior in practical applications. Compared to the graph Laplacian, geometry-aware Laplacian behaves more consistently and is free of the above issue, such that the graph Laplacian-based methods have been replaced by those using the cotangent Laplacian, which also is used in this paper.

Cotangent Laplacian For triangle meshes, a common choice of discrete Laplacian is cotangent Laplacian that forms the basis for a theory of discrete holomorphic functions and discrete Riemann surfaces and arises naturally via linear finite elements, discrete exterior calculus, electrical networks, and minimal surfaces. This operator is very sparse, easy to build, and generally works well for unstructured meshes with irregular vertex distributions, which also can be used on non-manifold meshes by accumulating per-triangle matrices.

Spectral Surface Reconstruction Our Laplacian-mesh pooling operation is built on the works of spectral surface reconstruction, which is different from some convolution theorem-based spectral convolutions [14] each of which projects the spatial signals of the surfaces to the spectral domain by Fourier transform before performing the point-wise multiplication in the spectral domain. The latter methods are mainly used when these shapes do not change the spectral bases because these algorithms are Fourier bases dependent, however, the feature space in our Laplacian2Mesh does not transform to the frequency domain, but the Laplacian space constructed by low-frequency Laplacian eigenbasis.

As a geometric shape can be seen as a collection of various signals [45], many spectral methods attempt to use this property to process the geometry. In particular, reconstructing geometric signals has achieved remarkable success and popularity for most, if not all, of these tasks. Figure 1 shows an example of the spectral reconstructed Homer – Homer Jay Simpson – shape with increasing numbers of coefficients of Laplacian eigenfunctions, and the reconstructed shape using more than half of the eigenfunctions is very similar to the original mesh. The phenomenon produced by this figure shows that non-shrinking filtering can be easily captured, by simply selecting the eigenvectors that correspond to low frequencies. Unlike the nature function space defined by spherical harmonics, the eigenfunction basis is well adapted to the geometry of the 3D shape and can be defined for shapes of

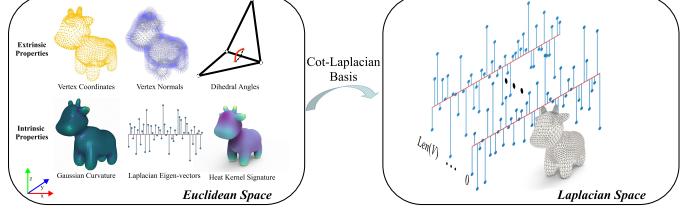


Fig. 2. Schematic diagram of spatial transformations performed in our work. We obtain our extrinsic properties (vertex coordinates, vertex normals, and dihedral angles of vertices) and intrinsic properties (gaussian curvature, Laplacian eigenvectors, and HKS) in Euclidean space, which are converted into Laplacian signal features by cot-Laplacian basis.

arbitrary topology.

3 METHODOLOGY

3.1 Preliminaries

We first briefly explain the background knowledge that provides the theoretical base for our algorithm design in this paper before we introduce the details of Laplacian2Mesh.

The triangle mesh $M = (V, F)$ consists of the vertices $V = \{v_i \mid v_i \in \mathbb{R}^3\}$ and the triangular faces $F = \{f_i \mid f_i \in \{1, \dots, |V|\}^3\}$ that are defined by triplets of vertices for triangular meshes.

Notably, for mesh M , the connectivity is explicitly defined using a set of pairs of vertices, which are called edges, in many previous methods, but in our defined Laplacian space, this connection relationship is a natural property. In the new space, we do not need to spend a lot of time deliberately designing a regular connection structure, which also makes the Laplacian pooling/unpooling operations in our network simpler and more efficient.

All the elements vertices V , faces F and edges of meshes have some natural extrinsic properties, such as the coordinates and normals. In this work, the i -th vertex v_i holds an input feature vector G_{v_i} including extrinsic features $G_{v_i}^{extrinsic}$ and intrinsic features $G_{v_i}^{intrinsic}$, which is to be processed by Laplacian2Mesh, more details in subsection 3.5. As mentioned above, the feature vector in the euclidean space is transformed to the Laplacian space by cot-Laplacian basis, wherefore the Laplacian signal features, notated \tilde{G}_{v_i} , instead of the obtained features in the 3D space, run through the neural network. The new feature vector in the Laplacian space can be shaped to any size by selecting eigenvectors E corresponding to the low-frequency laplacian eigenvalues, which will be utilized as the multi-resolution information.

The Laplacian spectral reconstruction, a critical property for our work, prioritize reconstruction of low-frequency information of the meshes (see Figure 1). For the cot-Laplacian matrix, we compute the k smallest eigenvectors $E_k = [e_1, e_2, \dots, e_k]$, then the new vertices V_{new} of the reconstructed mesh are given by:

$$V_{new} = E_k @ (E_k^T @ V), \quad (1)$$

where $@$ and T denote matrix multiplication and matrix transpose, respectively. Here, the reconstructed mesh differs very little from the original mesh when $k \geq \text{len}(V)/2$ is chosen.

In our work, hence, we utilize this property to map the feature vector G_{v_i} to the Laplacian signal features \tilde{G}_{v_i} , which be expressed as

$$\tilde{G}_v = E_k^T @ G_v. \quad (2)$$

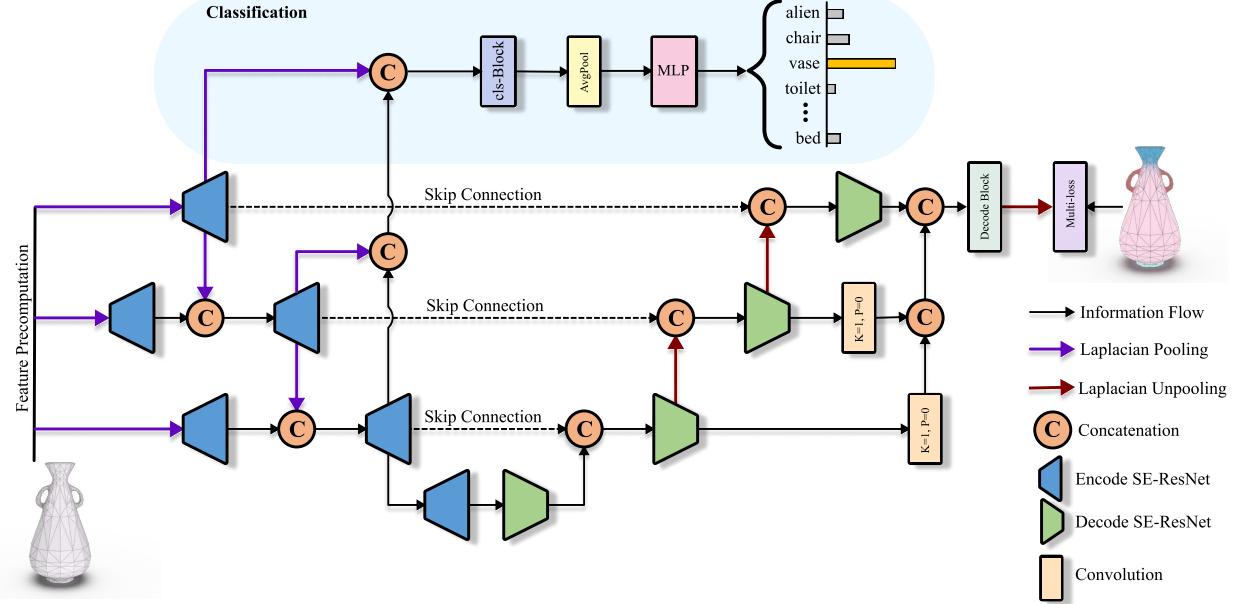


Fig. 3. The pipeline of our Laplacian2Mesh for the tasks of mesh classification and semantic segmentation. Given a 3D mesh as input, we build a hierarchical Laplacian-transform structure without explicitly considering the connection property of the mesh. We illustrate that the structure leads to re-think notions of pooling and unpooling operations on the meshes, and the encode/decode SE-ResNet blocks can make us not pay attention to the importance of each feature. The yellow blocks are the 1×1 convolution, out of which the $K = 1, P = 0$ means kernel=1 and padding=0 in this convolution block. The Multi-loss consists of the cross-entropy loss and adjacency-loss, where the latter is a new loss function we define in this paper.

Here, the dimension of \tilde{G}_v is $k \times \text{len}(G_{v_i})$, where $\text{len}(G_{v_i})$ is the dimension of features in the 3D space, and $\text{len}(G_{v_i}) = \text{len}(G_{v_i}^{\text{extrinsic}}) + \text{len}(G_{v_i}^{\text{intrinsic}})$. Figure 2 gives an overview of this operation for obtaining the Laplacian signal features. Since the k and $\text{len}(G_{v_i})$ are known values to us, using this property, our model can handle mesh shapes with any number of vertices without some operations like padding or truncation, wherefore we can circumvent the complex and irregular topology of mesh data.

3.2 Network Architecture

The architecture of our Laplacian2Mesh is illustrated in Figure 3, which includes two neural networks of 3D shape classification and semantic segmentation.

Given a triangle mesh, our Laplacian2Mesh takes the precomputed features matrix in the 3D space as the input of the neural network, where each row of the matrix is the feature vector of a vertex. We choose three values of k to achieve multi-resolution input, and we make sure that the first k value, notated k_0 , is greater than or equal to half the number of vertices of the mesh, that is, $k_0 \geq \text{len}(V)/2$, so as to obtain more mesh shape information.

Several Laplacian pooling operations are then used for spatial transformation and downsampling operation. At the same time, Laplacian unpooling plays an important role at the upsampling stage. Details about Laplacian pooling and unpooling operations will be further discussed in the next subsection.

In Laplacian2Mesh, many SE-ResNet blocks that are applied as the basic module of our network, which blocks can bring out the adaptive channel self-attention convolutions. These blocks free us from the task of distinguishing feature importance, while the ResNet part ensures that we can use a deep enough network. The architecture of the SE-ResNet block in our design is shown in Figure 4.

Moreover, we have designed cls-block (see Figure 5) and decode-block (see Figure 6) for the tasks of shape classification

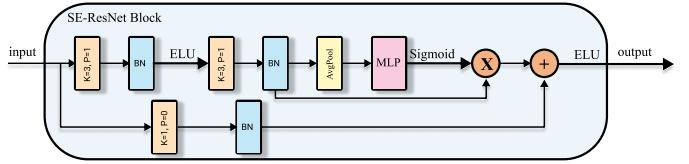


Fig. 4. The building blocks of our network. It can realize channel self-attention.

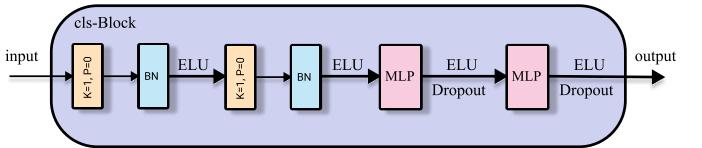


Fig. 5. The cls-Block is part of the pipeline that performs the task of mesh classification.

and segmentation, respectively. In these blocks, $ELU(\cdot)$ is chosen as the activation function in our Laplacian2Mesh, which eliminates the influence of bias shift and improves robustness to noise. Many convolution blocks with $\text{kernel_size} = 1$ and $\text{padding} = 0$ are used in our network to connect blocks with different dimensions, which can reduce the number of parameters and improve the generalization of this model when compared with multilayer perceptron (MLP). To reduce the information loss caused by downsampling, skip connections originating from the U-Net network [46] are utilized to directly let the encoding part and the decoding part with the same dimension hand in hand, followed by the operation of concatenation $\text{Cat}(\cdot)$ which can keep as much information as possible.

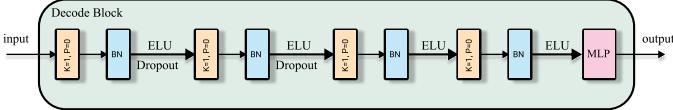


Fig. 6. The decode block is part of the pipeline that performs the task of semantic segmentation.

3.3 Laplacian Pooling & Unpooling

We introduce the Laplacian pooling and Laplacian unpooling to achieve the downampling and upsampling of our Laplacian2Net, which extend conventional pooling to the irregular mesh data. Figure 7 gives a brief illustration of these operations.

Laplacian Pooling. As shown in the lower part of the Figure 7, Laplacian pooling relies on the eigenvectors for conversion between different network layers, which makes the pooling operation on irregular triangle meshes as easy as on regular image grids. Unlike the method in MeshCNN [6] that needs to consider the connection of the edges and defines the mesh pooling via dynamic edge collapse [47], Laplacian2Mesh simplify this issue by only processing vertices V of the mesh M , which is simple but effective. Nevertheless, the connection property of mesh M is not ignored, but instead implicitly encoded into eigenvectors and adjacency-loss as described in the next subsection.

Since the total number of vertices in a mesh can vary significantly from model to model, an ideal network architecture should be able to deal with meshes with different numbers of vertices. From the previous subsection, we have known that our work free of this issue by choosing different values of k . To trade off the accuracy and time complexity of our work, three values of k are selected as a good decision, that is, $k = \{k_0, k_1, k_2\}$ in this paper.

For the certain layer i and j in the network, Laplacian pooling needs to transform the information flow I_i at layer i to the information flow I_{ij} , a new information flow that utilizes the $\text{len}(k_j)$ -dimension Laplacian space to represent the information flow of the $\text{len}(k_i)$ -dimension, via the eigenvectors k_i and k_j . The operation can be expressed as

$$I_{ij} = (k_j^T @ k_i) @ I_i. \quad (3)$$

Then a concatenation $\text{Cat}(\cdot)$ followed and merge the two tensors I_j and I_{ij} . The new information flow I_j^{new} at layer j is written as

$$I_j^{\text{new}} = \text{Cat}(I_j, I_{ij}). \quad (4)$$

Among them, the other $\text{Cat}(\cdot)$ operations in Figure 3 have the same function, which will not be elaborated.

With the pyramid of the values of k , the shape information of the 3D dragon model becomes more and more abstract (see Figure 7), that is to say, Laplacian pooling is used for both increasing the receptive field and eliminating the perturbation which occasionally occurs in high-frequency terms of the Laplacian sign.

Laplacian Unpooling. Laplacian unpooling, the upper path of the Figure 7, is the reverse process of Laplacian pooling using inverse-Laplacian transform, which can restore the information of different spatial dimensions instead of obtaining high-dimensional information through interpolation approach like other algorithms (i.e. SubdivNet [35]). Thus, the Laplacian unpooling does not learnable parameters, and it also relies on the eigenvectors from the cotangent Laplacian.

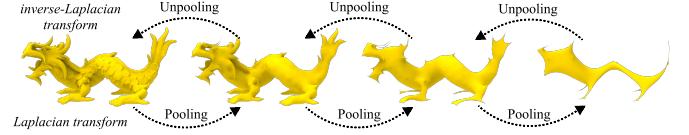


Fig. 7. Pooling and unpooling. The pooling operation based Laplacian transform can eliminate the high-frequency information of the 3D dragon shape and expand the network receptive field. The inverse-Laplacian transform-based unpooling operation is the inverse of pooling.

The Laplacian unpooling is also defined with the help of the mesh pyramid, and each of the layers is paired with a mesh Laplacian pooling layer in the task of semantic segmentation. Compared with the method [6] that stores the connectivity of the graphs at the input of each pooling layer, our approach is more simple and uniform.

3.4 Loss Function

Although the eigenvectors implicitly consider the edges of the mesh M , it is far from enough. In our design, to take advantage of the natural property of edge connection, we seek a way that can concisely describe the relationship among vertices. Instead of directly processing the edges, we achieve this by designing a new loss function, adjacency-loss function, with a clustering effect.

Our Laplacian2Mesh predicts the softmax values of the vertices V is P , however, the ground truth of the V is y . Then, we define the predictive value Y_i of vertex v_i that is

$$Y_i = P_{i,y_i}, \quad (5)$$

thus the predictive value Y of vertices V is $Y = [Y_0, Y_1, \dots, Y_{n-1}]^T$, where n is the number of vertices V . Since the adjacency-loss function is defined in the Euclidean space, we utilize the Euclidean distance $\Psi(v_i, v_j)$ to measure the distance between the vertex v_i and vertex v_j , which is computed as follows:

$$\Psi(v_i, v_j) = \sqrt{(v_i - v_j)^2}. \quad (6)$$

For the vertices V , the matrix of Euclidean distance is $\Psi(V) = \{\Psi(v_i, v_j) \mid i, j \in \{0, 1, \dots, n-1\}\}$, where $\Psi(V)$ is a oblique symmetric matrix.

To pay more attention to the neighbors of the V , the weighted Euclidean distance Θ of the Euclidean distance $\Psi(Y)$ of the predictive value Y is

$$\Theta = \Omega \odot \Psi(Y), \quad (7)$$

where \odot is the Hadamard product and the weight $\Omega = \exp(-\Psi(V)/2\sigma)$ is a Gaussian filter with the filter bandwidth σ . The Gaussian filter is considered the ideal time-domain filter in electronics and signal processing.

We introduce the adjacency matrix A to the new loss function, which can further refine the weighted Euclidean distance Θ in that we only focus on the 1-ring neighbors of the vertices V . For the vertex v_i , its weighted Euclidean distance from all 1-ring vertices is written as Θ_i^{adj} , that is,

$$\Theta_i^{\text{adj}} = \sum_{j=0}^{n-1} (\Theta_{ij} \cdot A_{ij}), \quad (8)$$

where Θ_i^{adj} is a scalar value.

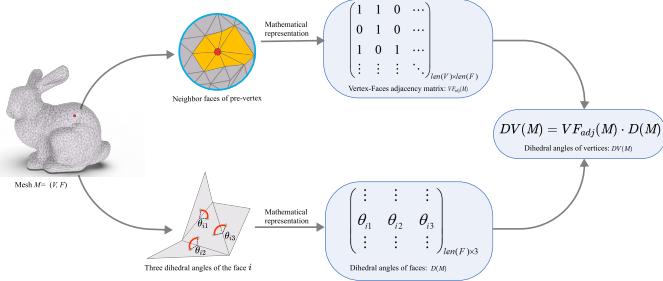


Fig. 8. Data flow for dihedral angles of vertices. In this figure, the vertex v_i (red point) has eight neighbor faces (yellow triangular faces), and the face f_i has three dihedral angles (red arcs).

Then, Laplacian2Mesh proposes the adjacency-loss function that is computed as follows:

$$L^{adj} = \frac{\sum_{i=0}^{n-1} (\Theta_i^{adj})}{len(y)}, \quad (9)$$

where $\Phi_i = \sum_{j=0}^{n-1} A_{ij}$ is the number of the 1-ring neighbors of the vertex v_i . Here, we can see that L^{adj} simultaneously considers the connection property of the edges and the spatial clustering of vertices V .

In our Laplacian2Mesh, we utilize the multi-loss functions for the task of semantic segmentation, where the primal loss function is cross entropy loss L^{CE} . The loss function is defined as:

$$L = L^{CE} + \omega \cdot L^{adj}, \quad (10)$$

where the ω is a hyperparameter that makes a trade-off between L^{CE} and L^{adj} . For the task of shape classification, We only use the cross-entropy loss L^{CE} .

3.5 Input Features

The input feature for each vertex is a 39-dimensional vector, which is divided into a 30-dimensional intrinsic shape descriptor and a 9-dimensional extrinsic shape descriptor (see Figure 2). The intrinsic shape descriptor includes 1-dimensional gaussian curvature, 9-dimensional HKS, and the other 20 dimensions are the absolute values of the Laplacian eigenvectors corresponding to the 20 lowest frequencies excluding 0. The components of the extrinsic shape descriptor are the 3-dimensional vertex coordinates, the 3-dimensional vertex normal, and the 3-dimensional dihedral angles on the vertex.

Dihedral Angles of Vertices. Extensive work has demonstrated that graph-cut technology plays a pivotal role in 3D shape semantic segmentation tasks, which was often attached to the information processing architecture as post-processing in previous work. In recent years, as an implicit graph-cut, dihedral angles of the 3D surface mesh have been directly used as a feature to train the neural network, and its effect is remarkable. Instead of directly processing the dihedral angles of the faces, we also utilize the mapping transformation to convert the dihedral angles feature on the faces to those on the vertices.

As illustrated in Figure 8, the 3D bunny has $len(V)$ vertices V and $len(F)$ faces F . For the mesh M , $VF_{adj}(M)$ and $D(M)$ express the vertex-faces adjacency matrix and dihedral angles of faces, respectively. Then, the dihedral angles of vertices matrix $DV(M)$

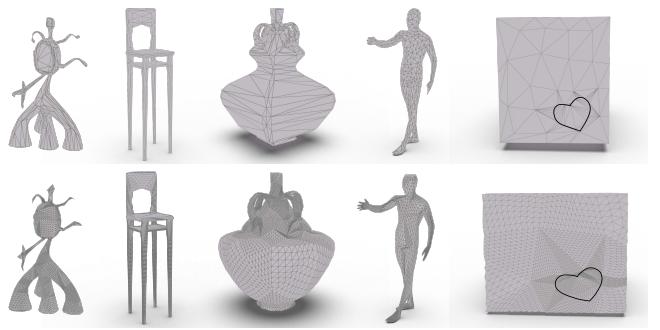


Fig. 9. Two evolutionary versions of the original dataset: the upper part is the data-meshcnn from [6], and the lower part is the data-subdivnet from [35].

with $len(V) \times 3$ dimension is computed by the equation on the far right of the Figure 8.

A face f_i in the manifold and watertight 3D mesh has 3 neighbor faces, however, Laplacian2Mesh does not require the mesh to be manifold or watertight. This relaxed condition results in that the number of neighbor faces of the face f_i is not 3. If the edge is a non-manifold edge, our solution is to discard all dihedral angles corresponding to this edge; if the face f_i is a boundary face, we will pad the missing dihedral angle value with the value 1, which mean that this location has gentle terrain.

4 EXPERIMENTS

Laplacian2Mesh is a general approach, which may be applied to a wide range of 3D shape understanding tasks. We demonstrate its performance for the two fundamental geometric analysis tasks of mesh classification and mesh semantic segmentation, comparing it to the reported results for recently-used datasets. We have also conducted many ablation experiments to demonstrate the effectiveness of the key components of our Laplacian2Mesh.

4.1 Data Processing

For all tasks, there are currently two versions evolved from original datasets, data-meshcnn from MeshCNN [6] and data-subdivnet from SubdivNet [35]. The former simplified each mesh in the original dataset to roughly the same number of edges, and the latter performs remesh processing on the meshes in order to add the Loop subdivision sequence connectivity to the data. As shown in Figure 9, data-subdivnet has a denser mesh structure than data-meshcnn, which generates 10 to 20 remeshed meshes for each mesh in the training dataset and test dataset. Since our work relies on the Laplacian eigendecomposition, it prefers the simple data-meshcnn, thus, unless otherwise specified, the dataset we use is data-meshcnn in the following experiments.

Augmentation. To make the network converge faster, we first process the input meshes to fit inside a unit cube. For the training set, we apply random anisotropic scaling with a normal distribution mean $\mu = 1$ and standard deviation $\delta = 0.1$ on the vertex locations, following MeshCNN [6]. In order to increase the diversity of some datasets, we are inspired by SubdivNet [35] to randomly select three Euler angles (randomly sampled from $0, \pi/2, \pi$, or $3\pi/2$) to rotate the data.

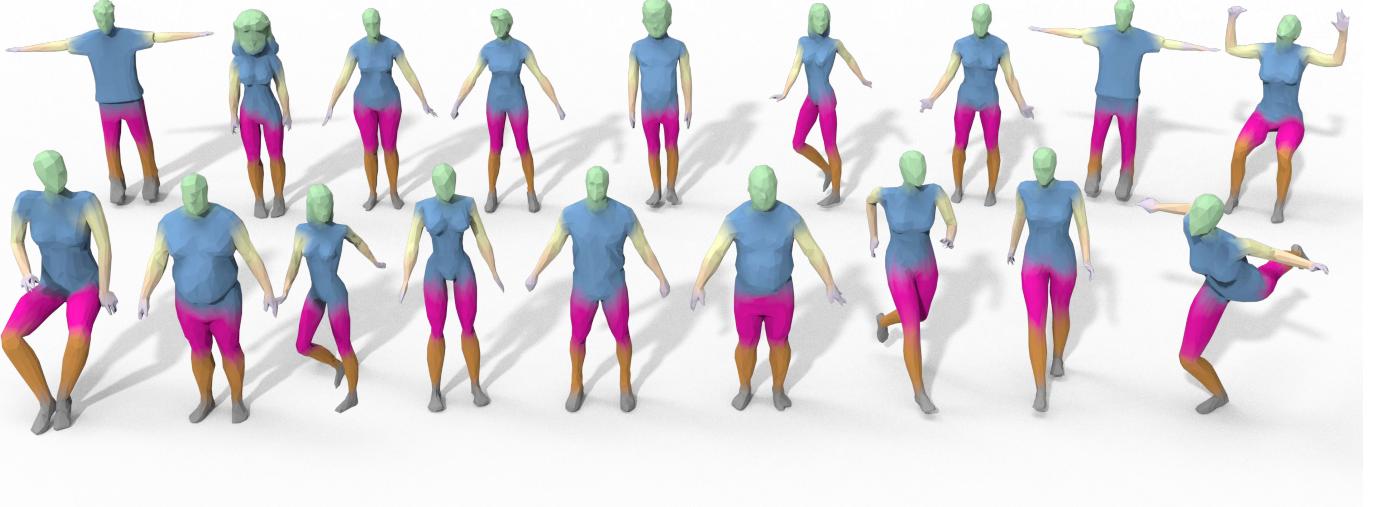


Fig. 10. Mesh segmentation results on the entire Human Body test set. The qualitative results of the co-segmentation demonstrate the effectiveness of our Laplacian2Mesh on the task of human body segmentation.

4.2 Mesh Classification

We demonstrate the power of Laplacian2Mesh on the mesh classification task with two recently-used datasets: SHREC11 [48] and Manifold40 [35]. The goal is to classify each mesh from the test set into one of the pre-defined classes, and we define that the accuracy is the ratio of correctly predicted meshes.

SHREC11. The SHREC11 dataset consists of 30 classes, with 20 examples per class. Following the setting in [49], our Laplacian2Mesh is evaluated on two protocols out of which the dataset is split into 16 or 10 training samples in each subject.

Table 1 reports the average accuracy on 3 random splits into training and test set. Surprisingly, Laplacian2Mesh can correctly all meshes in the test set, no matter how the dataset is split, which means that we achieve 100% accuracy on this dataset. When the split is 16 examples for training and 4 for test, Laplacian2Mesh with large learning rate has reached 100 % accuracy in the fourth epoch. When training and test have the same number of the objects, we also get the 100% accuracy at about the 400th epoch. These suggest that Laplacian2Mesh is completely competent for the mesh classification task on the SHREC11 dataset.

TABLE 1
Classification accuracy on the SHREC11 dataset [48].

Method	Split 16	Split 10
GWCNN	96.6%	90.3%
MeshCNN	98.6%	91.0%
PD-MeshNet	99.7%	99.1%
MeshWalker	98.6%	97.1%
SubdivNet	99.9%	99.5%
HodgeNet	99.2%	94.7%
Laplacian2Mesh (ours)	100%	100%

Manifold40. Compared with ModelNet40 [9] that is a commonly-used dataset, Manifold40 dataset has a more compact data structure, which is helpful to speed up the training process of our network. Manifold40 is the reconstructed dataset of ModelNet40, nevertheless, Manifold40 is more challenging since the

TABLE 2
Classification accuracy on Manifold40 [35]. The first two rows are the point cloud-based methods, and other methods input meshes.

Method	Input	Accuracy
PointNet++	point cloud	87.9%
PCT	point cloud	92.4%
MeshNet	mesh	88.4%
MeshWalker	mesh	90.5%
SubdivNet	mesh	90.9%
Laplacian2Mesh (ours)	mesh	91.2%

reconstruction error and simplification distortion. This dataset contains 12311 CAD models from 40 categories, where 9843 models are used for training and 1468 models are used for test.

Our method obtains results comparable to the state-of-the-art for this dataset; results are shown in Table 2. We report both the point cloud-based results and the mesh-based results, and Laplacian2Mesh stands out among the mesh-based methods on the Manifold40 dataset. Inspired by MeshWalker [32], the accuracy of our Laplacian2Mesh can reach 92.8% when the 5 cross-labeled classes (desk/table and flower-pot/plant/vase) are discarded. The reason is that these models in the 5 categories (divided into two groups) produce very similar results in the same group when using fewer eigenvectors for spectral reconstruction, which are equally ambiguous for humans to distinguish.

4.3 Mesh Semantic Segmentation

Mesh semantic segmentation is the basis for many applications in shape understanding. To test our algorithm, we applied our method to two commonly-used datasets: Human Body Segmentation [22] and COSEG [50]. We evaluate the accuracy of the predicted every face, which is an agreed evaluation metric.

Human Body Segmentation. The human body dataset, labeled by [22], consists of 370 training shapes from SCAPE [51], FAUST [52], MIT [53], and Adobe Fuse [54]. The 18 models in the test set are from SHREC07 [55] (humans) dataset. According to the

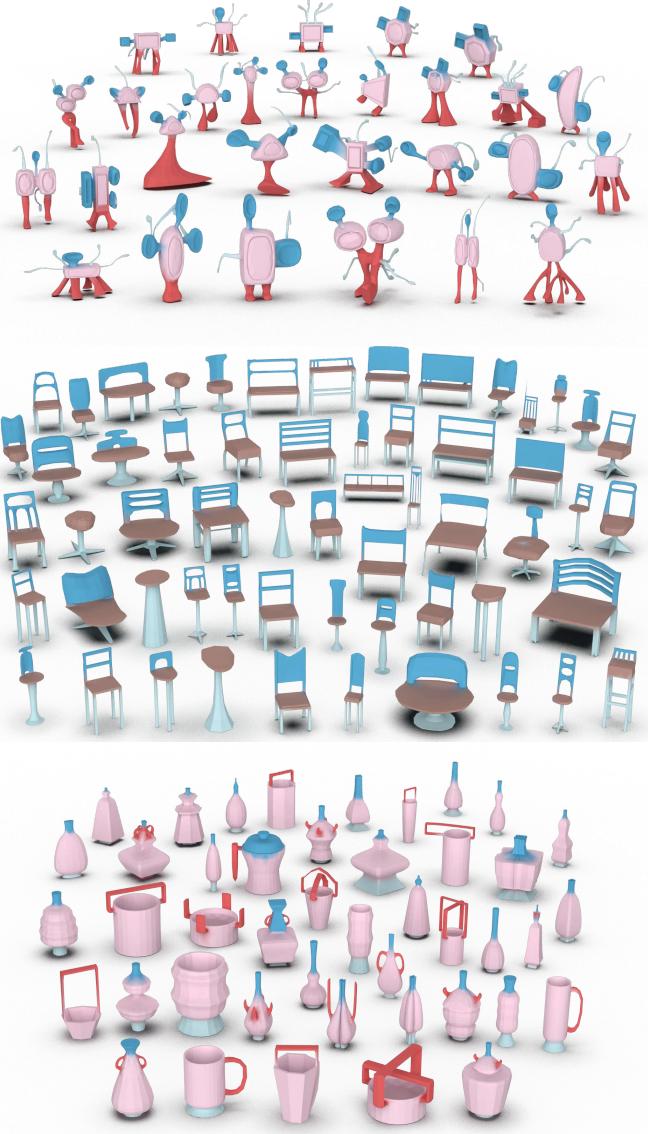


Fig. 11. Gallery of segmentation results for the COSEG dataset. From top to bottom, they are Tele-aliens sub-dataset, Chairs sub-dataset, and Vases sub-dataset.

TABLE 3
Mesh segmentation accuracy on the human body dataset [22].

Method	Input	Accuracy
PointNet	point cloud	74.70%
PointNet++	point cloud	82.30%
MeshCNN	mesh	85.39%
PD-MeshNet	mesh	85.61%
HodgeNet	mesh	85.03%
Laplacian2Mesh (ours)	mesh	88.58%

labels in [56], the meshes are manually segmented into 8 parts. We use the same version of human body dataset as in MeshCNN [6], which is downsampled to 1500 faces per mesh.

For a fair comparison, as in HodgeNet [57], the accuracy we got based on "hard" ground-truth face labels rather than the result after "soft" or smooth processing [32]. We report the segmentation accuracy of this dataset in Table 3, which shows that our method generates more accurate part segmentation. We also show our

learned segmentations on the entire Human Body test set in Figure 10 where the boundaries of the shape segmentation are relatively consistent.

COSEG. We also evaluate the performance of Laplacian2Mesh on the three largest sets from the COSEG shape dataset: Vases, Chairs, and Tele-aliens containing 300, 400, and 200 models in each, respectively. The meshes are labeled into 3 parts (Chairs set) or 4 parts (Vases set & Tele-aliens set). We generate a random 85%-15% train-test split for each set, as in MeshCNN [6].

The quantitative results are provided in Table 4, and the qualitative results for all models in the test set are displayed in Figure 11. Our method achieves somewhat competitive results compared to the state-of-the-art methods. Since many large triangular faces exist at the junctions of the parts in the Tele-aliens set, this will cause our model to present incorrect segmentation labels. Nevertheless, the visual shapes segmentation results at the top part of Figure 11 are still surprising.

To further test the effectiveness of our method, we also use the evolved COSEG dataset from SubdivNet [35]. In contrast to the original dataset, both training and test sets in the dataset are augmented by a factor of ten, each mesh has over 8000 vertices and over 16000 faces. The dataset is split into training and test sets with a ratio of 4:1. Laplacian2Mesh uses the same dataset without additional data augmentation. Table 5 shows that our method is evenly matched with SubdivNet [35].

TABLE 4
Mesh segmentation accuracy on the COSEG dataset [50].

Method	Vases	Chairs	Tele-aliens
DCN	90.90%	95.70%	-
MeshCNN	92.36%	92.99%	96.26%
HodgeNet	90.30%	95.68%	96.03%
Laplacian2Mesh (ours)	94.58%	96.58%	95.01%

TABLE 5
Mesh segmentation accuracy on the COSEG dataset processed by [35].

Method	Vases	Chairs	Tele-aliens
SubdivNet	95.99%	95.10%	98.55%
Laplacian2Mesh (ours)	95.53%	97.31%	96.05%

4.4 Non-watertight & Non-manifold Mesh Segmentation

Many current excellent mesh segmentation methods (like MeshCNN [6]) are trained on the premise that the input mesh must be watertight or manifold or both. This puts forward higher requirements for the limited mesh datasets, which limits the development of 3D geometric deep learning. Laplacian2Mesh does not require such perfect mesh data, and a small number of unclosed regions on the mesh or non-manifold elements (non-manifold vertices or non-manifold edges) will not affect our predictions, as shown in Figure 12. This is due to the fact that our method deals with features on each vertex, and the definition of special mesh connections allows us to avoid this issue.

4.5 Ablation Studies

We perform three ablation studies (on the input features or on the adjacency-loss or the input size) to justify some of the design and parameter choices in our architecture.

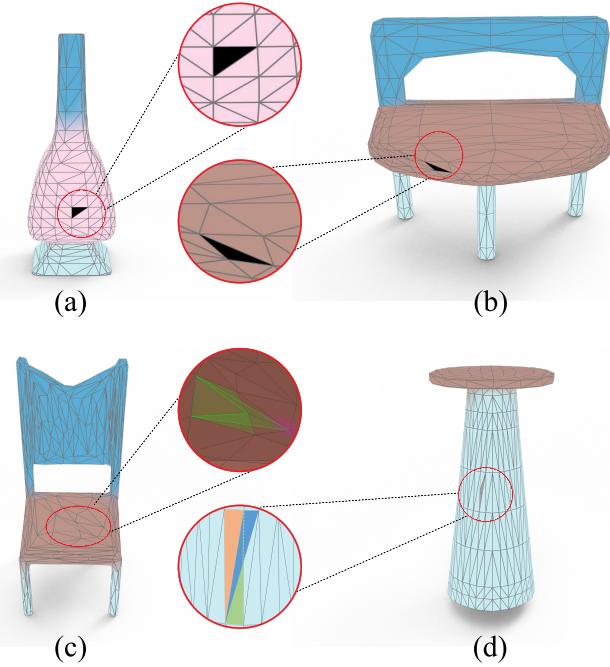


Fig. 12. Segmentation results on non-watertight and non-manifold data. (a) and (b) are non-watertight vase and chair, respectively; (c) has a non-manifold vertex, and (d) has a non-manifold edge.

Input Features. In contrast to the pixel shapes of the 2D images are consistent everywhere, our Laplacian2Mesh requires different input features (see Figure 2) to represent the local geometry and global geometry of the mesh triangle faces, respectively. Thus, both the intrinsic features and extrinsic features we choose in our work are essential to the capability of Laplacian2Mesh. Table 6 indicates that these features are necessary for our model regardless of the shape understanding tasks. As expected, Laplacian eigenvectors has the greatest impact in our Laplacian2Mesh. The phenomenon can be explained by the fact that the Laplacian eigenvectors not only represent the frequency information of 3D meshes, but also implicitly encode the connection information of mesh.

TABLE 6

The results of ablation experiments on the input features. no-Gaussian Curvature means lack of Gaussian Curvature feature compared to Laplacian2Mesh, the rest of these cases are similar.

	Vases	Manifold40
no-Gaussian Curvature	92.22%	90.28%
no-Laplacian EigenVectors	86.30%	85.32%
no-HKS	92.03%	89.75%
no-Dihedral Angles	90.48%	90.14%
Laplacian2Mesh (ours)	94.58%	91.21%

Adjacency Loss. Although Laplacian eigenvectors implicitly encode the connection property of the mesh is effective for our tasks, in the mesh segmentation task, there are still a few vertex features with high similarity, resulting in wrong segmentation results. We use the t-SNE technology to project the multi-dimensional results to the 2D images, as shown in Figure 13. The figure has shown that our designed adjacency loss function as described in subsection 3.4 has the clustering effect, which can make isolated vertices assimilated by their surrounding neighbors. The

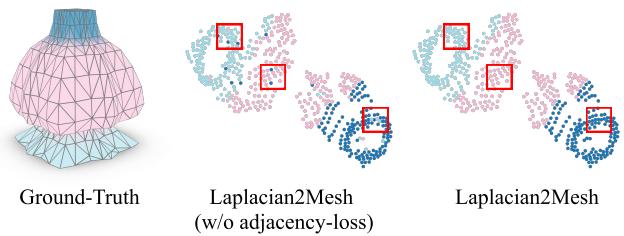


Fig. 13. t-SNE analysis for our adjacency-loss. The figure shows the hierarchy of the segmentation result. Our method cluster the meshes that belong to the same class (indicated by color) without breaking the segmentation boundaries.

TABLE 7
The results of ablation experiments on the adjacency-loss.
no-adjacency-loss means Laplacian2Mesh does not use the adjacency-loss.

	Vases	Chairs
no-adjacency-loss	93.28%	96.32%
Laplacian2Mesh (ours)	94.58%	96.58%

quantitative results listed in Table 7 also illustrate that adjacency loss can improve the accuracy of network prediction.

Input Size. Figure 1 has indicated that the spectral reconstruction loss of Homer mesh is inversely proportional to the number of Laplacian eigenvectors. For the multi-resolution input of Laplacian2Mesh, is it better to choose larger value group of $k = \{k_0, k_1, k_2\}$? As expected, we can know from the results in Table 8 that the answer to the above question is NO. Tele-aliens dataset achieves the best result with a larger k group chosen, but not so for the human-body dataset. From Figure 14, we can find that the difference between the mesh shapes of spectral reconstruction is proportional to the difference between the selected k . From Figure 14, we can find that the difference between the mesh shapes of spectral reconstruction is proportional to the difference between the selected k . Therefore, a larger value group of k should be chosen on the premise of ensuring the difference.

TABLE 8
The results of ablation experiments on the input sizes.

	input sizes	Accuracy
Tele-aliens	512-128-16	94.23%
	512-128-32	94.46%
	512-256-64	95.01%
Human body	512-64-16	85.39%
	512-128-32	88.58%
	512-256-64	86.26%

5 LIMITATIONS

The Cube Engraving dataset [58] contains 4600 objects, out of which 3910 models are used for training and 690 models are used for testing. Each model in the dataset is a cube "engraved" with a shape at a random face in a random location, as demonstrated in the lower part of Figure 15. The mesh shape belongs to a dataset with 23 categories, each of which contains approximately 20 models.

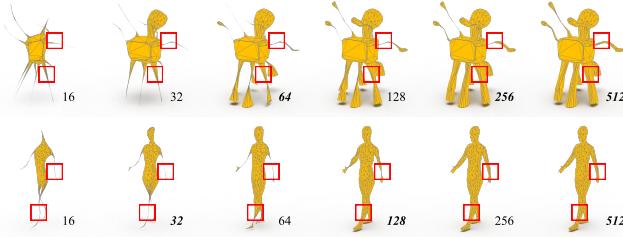


Fig. 14. The spectral reconstruction results of Tele-aliens and human body under different k choices. The number below the shape represents the number of eigenvectors used to reconstruct this shape, where the 3 bold italic values are the k value we use in experiments. The red boxes represent the parts with obvious shape changes during the reconstruction process.

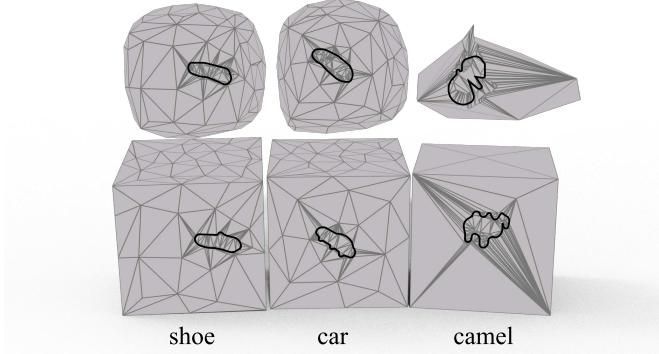


Fig. 15. Limitation. The three cubes in the lower part of this figure are from Cube Engraving dataset, and the upper part is the corresponding spectral reconstruction result. From these shapes, it is difficult for our Laplacian2Mesh to extract discriminative information from them.

Since our method cannot achieve target-driven mesh simplification similar to MeshCNN [6] when performing pooling operations, and the relatively small shape area ratio makes it easier to be ignored in the Laplacian pooling process, as shown in the upper part of Figure 15. Table 9 shows the accuracy of our model for the Cube dataset, which includes two dataset versions we evaluated. However, even when using the data-subdivnet for training, the problem has not been completely solved either.

TABLE 9
Classification accuracy on the Cube Engraving dataset [58].

Method	Accuracy
PointNet++	64.3%
MeshCNN	92.2%
PD-MeshNet	94.4%
MeshWalker	98.6%
SubdivNet	98.9%
Laplacian2Mesh (ours)	91.5%
Laplacian2Mesh (subdivision data)	93.7%

6 CONCLUSION AND FUTURE WORK

We have presented Laplacian2Mesh, a general geometric neural network on irregular triangular meshes. The key idea is to perform spatial transformation-based Laplacian pooling/unpooling via Laplacian eigenvectors. and use the adjacency loss function we

proposed to implement clustering of the vertices in the Euclidean space.

Our approach is general, yet simple. It has several additional benefits. We also did a design to get the dihedral angle on the vertices, which is an implicit graph-cut representation. In contrast to designs where the elements of the meshes being processed must have a regular structure, our method does not require a uniform number of neighbors of vertices in each mesh. Our Laplacian2Mesh achieves competitive results in two mesh understanding tasks—mesh classification and mesh semantic segmentation, and can even handle non-watertight and non-manifold mesh shape.

Apart from the applications in the paper, Several future extensions can be explored. Our ideas can provide a worthwhile exploration direction for other mesh-based research areas, such as shape correspondence, 3D shape retrieval, registration of multiple meshes, etc. They could also be applied to other relate areas, such as GNNs and social networks.

Finally, we believe our approach does not drain the lucrative characteristics of Laplacian. The eigenvalues of laplacian contain important information about the number of mesh segmentations, which will be important prior knowledge for our future work. Another area of future work is to design an unsupervised robust network for mesh semantic segmentation. Our idea is to utilize differentiable rendering to bridge 2D images and 3D shapes [59], using well-established 2D pre-trained networks to supervise the output of the 3D network.

REFERENCES

- [1] C. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 77–85, 2017.
- [2] C. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” in *NIPS*, 2017.
- [3] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, “Pointcnn: Convolution on x-transformed points,” in *NeurIPS*, 2018.
- [4] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller, “Multi-view convolutional neural networks for 3d shape recognition,” *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 945–953, 2015.
- [5] R. Klokov and V. S. Lempitsky, “Escape from cells: Deep kd-networks for the recognition of 3d point cloud models,” *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 863–872, 2017.
- [6] R. Hanocka, A. Hertz, N. Fish, R. Giryes, S. Fleishman, and D. Cohen-Or, “Meshcnn: a network with an edge,” *ACM Transactions on Graphics (TOG)*, vol. 38, pp. 1 – 12, 2019.
- [7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, “Large-scale video classification with convolutional neural networks,” *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.
- [8] M. Chen, G. Ding, S. Zhao, H. Chen, Q. Liu, and J. Han, “Reference based lstm for image captioning,” in *AAAI*, 2017.
- [9] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3d shapenets: A deep representation for volumetric shapes,” *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1912–1920, 2015.
- [10] B. Lévy, “Laplace-beltrami eigenfunctions towards an algorithm that “understands” geometry,” *IEEE International Conference on Shape Modeling and Applications 2006 (SMI’06)*, pp. 13–13, 2006.
- [11] W. He, Z. Jiang, C. Zhang, and A. M. Sainju, “Curvanet: Geometric deep learning based on directional curvature for 3d shape analysis,” *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.
- [12] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, “Squeeze-and-excitation networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 2011–2023, 2020.
- [13] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velicković, “Geometric deep learning: Grids, groups, graphs, geodesics, and gauges,” *ArXiv*, vol. abs/2104.13478, 2021.

- [14] M. M. Bronstein, J. Bruna, Y. LeCun, A. D. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, pp. 18–42, 2017.
- [15] Y. Xiao, Y.-K. Lai, F.-L. Zhang, C. Li, and L. Gao, "A survey on deep geometry learning: From a representation perspective," *Computational Visual Media*, vol. 6, pp. 113–133, 2020.
- [16] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and discriminative voxel modeling with convolutional neural networks," *ArXiv*, vol. abs/1608.04236, 2016.
- [17] L. P. Tchapmi, C. B. Choy, I. Armeni, J. Gwak, and S. Savarese, "Segcloud: Semantic segmentation of 3d point clouds," *2017 International Conference on 3D Vision (3DV)*, pp. 537–547, 2017.
- [18] R. Hanocka, N. Fish, Z. Wang, R. Giryes, S. Fleishman, and D. Cohen-Or, "Alignet: Partial-shape agnostic alignment via unsupervised learning," *ACM Trans. Graph.*, vol. 38, pp. 1:1–1:14, 2019.
- [19] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong, "O-cnn," *ACM Transactions on Graphics (TOG)*, vol. 36, pp. 1 – 11, 2017.
- [20] G. Riegler, A. O. Ulusoy, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6620–6629, 2017.
- [21] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3d shape surfaces using geometry images," in *ECCV*, 2016.
- [22] H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V. G. Kim, and Y. Lipman, "Convolutional neural networks on surfaces via seamless toric covers," *ACM Transactions on Graphics (TOG)*, vol. 36, pp. 1 – 10, 2017.
- [23] N. Haim, N. Segol, H. Ben-Hamu, H. Maron, and Y. Lipman, "Surface networks via general covers," *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 632–641, 2019.
- [24] T. Le, G. Bui, and Y. Duan, "A multi-view recurrent neural network for 3d mesh segmentation," *Comput. Graph.*, vol. 66, pp. 103–112, 2017.
- [25] J. Masci, D. Boscaini, M. M. Bronstein, and P. Vandergheynst, "Geodesic convolutional neural networks on riemannian manifolds," *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pp. 832–840, 2015.
- [26] F. Monti, D. Boscaini, J. Masci, E. Rodolà, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5425–5434, 2017.
- [27] N. Verma, E. Boyer, and J. Verbeek, "Feastnet: Feature-steered graph convolutions for 3d shape analysis," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2598–2606, 2018.
- [28] D. Boscaini, J. Masci, S. Melzi, M. M. Bronstein, U. Castellani, and P. Vandergheynst, "Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks," *Computer Graphics Forum*, vol. 34, 2015.
- [29] M. Fey, J. E. Lenssen, F. Weichert, and H. Müller, "Splinecnn: Fast geometric deep learning with continuous b-spline kernels," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 869–877, 2018.
- [30] S. C. Schonsheck, B. Dong, and R. Lai, "Parallel transport convolution: A new tool for convolutional neural networks on manifolds," *ArXiv*, vol. abs/1805.07857, 2018.
- [31] J. Schult, F. Engelmann, T. Kontogianni, and B. Leibe, "Dualconvmeshnet: Joint geodesic and euclidean convolutions on 3d meshes," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8609–8619, 2020.
- [32] A. Lahav and A. Tal, "Meshwalker: Deep mesh understanding by random walks," *ACM Trans. Graph.*, vol. 39, pp. 263:1–263:13, 2020.
- [33] J. Atwood and D. F. Towsley, "Diffusion-convolutional neural networks," in *NIPS*, 2016.
- [34] F. Milano, A. Loquercio, A. Rosinol, D. Scaramuzza, and L. Carlone, "Primal-dual mesh convolutional neural networks," *ArXiv*, vol. abs/2010.12455, 2020.
- [35] S. Hu, Z.-N. Liu, M.-H. Guo, J. Cai, J. Huang, T.-J. Mu, and R. R. Martin, "Subdivision-based mesh convolution networks," *ArXiv*, vol. abs/2106.02285, 2021.
- [36] M. Ovsjanikov, J. Sun, and L. J. Guibas, "Global intrinsic symmetries of shapes," *Computer Graphics Forum*, vol. 27, 2008.
- [37] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas, "Functional maps: A flexible representation of maps between shapes," *ACM Trans. Graph.*, vol. 31, no. 4, jul 2012. [Online]. Available: <https://doi.org/10.1145/2185520.2185526>
- [38] M. M. Bronstein and A. M. Bronstein, "Shape recognition with spectral distances," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, pp. 1065–1071, 2011.
- [39] A. M. Bronstein, M. M. Bronstein, L. J. Guibas, and M. Ovsjanikov, "Shape google: Geometric words and expressions for invariant shape retrieval," *ACM Trans. Graph.*, vol. 30, pp. 1:1–1:20, 2011.
- [40] J. Sun, M. Ovsjanikov, and L. J. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," *Computer Graphics Forum*, vol. 28, 2009.
- [41] Y. Wang and J. M. Solomon, "Intrinsic and extrinsic operators for shape analysis," *Handbook of Numerical Analysis*, 2019.
- [42] R. Liu and H. Zhang, "Mesh segmentation via spectral embedding and contour analysis," *Computer Graphics Forum*, vol. 26, 2007.
- [43] J. Bruna, W. Zaremba, A. D. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," *CoRR*, vol. abs/1312.6203, 2014.
- [44] Y.-L. Qiao, L. Gao, J. Yang, P. L. Rosin, Y.-K. Lai, and X. Chen, "Learning on 3d meshes with laplacian encoding and pooling," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, pp. 1317–1327, 2022.
- [45] M. Botsch and O. Sorkine-Hornung, "On linear variational surface deformation methods," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, pp. 213–230, 2008.
- [46] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.
- [47] H. Hoppe, "View-dependent refinement of progressive meshes," *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997.
- [48] Z. Lian, A. Godil, B. Bustos, M. Daoudi, J. Hermans, S. Kawamura, Y. Kurita, G. Lavoué, H. V. Nguyen, R. Ohbuchi, Y. Ohkita, Y. Ohishi, F. M. Porikli, M. Reuter, I. Sipiran, D. Smeets, P. Suetens, H. Tabia, and D. Vandermeulen, "Shrec '11 track: Shape retrieval on non-rigid 3d watertight meshes," in *3DOR@Eurographics*, 2011.
- [49] D. Ezuz, J. M. Solomon, V. G. Kim, and M. Ben-Chen, "Gwcnn: A metric alignment layer for deep shape analysis," *Computer Graphics Forum*, vol. 36, 2017.
- [50] Y. Wang, S. Asafi, O. M. van Kaick, H. Zhang, D. Cohen-Or, and B. Chen, "Active co-analysis of a set of shapes," *ACM Transactions on Graphics (TOG)*, vol. 31, pp. 1 – 10, 2012.
- [51] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "Scape: shape completion and animation of people," *ACM Trans. Graph.*, vol. 24, pp. 408–416, 2005.
- [52] F. Bogo, J. Romero, M. Loper, and M. J. Black, "Faust: Dataset and evaluation for 3d mesh registration," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3794–3801, 2014.
- [53] D. Vlasic, I. Baran, W. Matusik, and J. Popović, "Articulated mesh animation from multi-view silhouettes," *ACM SIGGRAPH 2008 papers*, 2008.
- [54] Adobe, "Adobe fuse 3d characters," <https://www.mixamo.com>, 2016.
- [55] D. Giorgi, S. Biasotti, and L. Paraboschi, "Shape retrieval contest 2007: Watertight models track," 2007.
- [56] E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3d mesh segmentation and labeling," in *SIGGRAPH 2010*, 2010.
- [57] D. Smirnov and J. M. Solomon, "Hodgenet: Learning spectral geometry on triangle meshes," *ACM Trans. Graph.*, vol. 40, pp. 166:1–166:11, 2021.
- [58] L. J. Latecki and R. Lakämper, "Shape similarity measure based on correspondence of visual parts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1185–1190, 2000.
- [59] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and Z. Cui, "Dist: Rendering deep implicit signed distance function with differentiable sphere tracing," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2016–2025, 2020.