Computer-Aided Design 145 (2022) 103181

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

3D Shape Segmentation Using Soft Density Peak Clustering and Semi-Supervised Learning

Zhenyu Shu [a,b], Sipeng Yang [c,*], Haoyu Wu [d], Shiqing Xin [e], Chaoyi Pang [a,b], Ladislav Kavan [f], Ligang Liu [g]

[a] School of Computer and Data Engineering, NingboTech University, Ningbo, 315100, PR China
[b] Ningbo Institute, Zhejiang University, Ningbo, 315100, PR China
[c] College of Computer Science and Technology, Zhejiang University, Hangzhou, 310058, PR China
[d] Polytechnic Institute, Zhejiang University, Ningbo, 315100, PR China
[e] School of Computer Science and Technology, ShanDong University, Jinan, 250100, PR China
[f] School of Computing, University of Utah, Salt Lake City, 84112, USA
[g] Graphics & Geometric Computing Laboratory, School of Mathematical Sciences, University of Science and Technology of China, Anhui, 230026, PR China

ARTICLE INFO

Article history:
Received 11 June 2021
Received in revised form 25 November 2021
Accepted 30 November 2021

Keywords:
3D shapes
Segmentation
Semi-supervised
Deep learning

ABSTRACT

Shape segmentation plays a vital role in shape analysis. Recent research uses fully-supervised learning methods to achieve state-of-the-art performance. However, acquiring fully-labeled training data is usually an extremely expensive process. In this paper, we present a novel semi-supervised algorithm for 3D shape segmentation. In our method, users are only required to locate several seed faces with a very simple interaction by using our soft density peak clustering method. Our method can automatically learn the required label information and produce satisfactory segmentation results with our novel optimization model. Various experimental results show that the presented method can achieve superior segmentation performance over previous unsupervised methods and comparable performance to the fully-supervised methods.

1. Introduction

Segmenting 3D shapes into several meaningful parts, i.e., 3D shape segmentation, is a fundamental problem in 3D shape analysis. Numerous geometry processing problems, such as 3D shape retrieval, 3D shape classification, part-based 3D modeling, 3D mesh simplification, and 3D shape skeleton extraction, benefit from fast and robust 3D shape segmentation algorithms. Traditionally, a typical workflow of 3D shape segmentation algorithms consists of three steps. In the first step, each face on 3D shapes is mapped to a corresponding feature vector. As the second step, the feature vectors are clustered into several clusters and given a label in the feature space. Each face is given a label according to its corresponding feature vector's label in the final step.

Recently, learning-based approaches [1–5], especially the deep learning-based ones [6], show their superior performance with more 3D shapes available over non-learning based methods. Moreover, they can be mainly divided into three categories: supervise ones [7], semi-supervised ones [8], and unsupervised ones [9]. Most supervised 3D shape segmentation methods heavily rely on high-quality training data to learn the prior knowledge, which are usually fully-labeled 3D shapes. Although the performance of supervised ones is usually better than the other two categories of methods, fully labeling numerous 3D shape is a very expensive, time-consuming, and extremely tedious task.

In this paper, we propose a novel and effective semi-supervised method for 3D shape segmentation to avoid being faced with the demanding task of preparing expensive fully-labeled training data. Different from existing work, our method labels 3D shapes in the feature space, instead of on the surfaces directly. Our method ensures that the marked parts contain the key features of each segment. Only a simple and efficient user interaction is used in our method to assist in locating several initial labeled seeds. And the label information can be automatically propagated to the unlabeled parts of 3D shapes with our novel optimization model. Fig. 1 shows some examples of segmentation results from our method. The overall workflow of our method is shown in Fig. 2.

The main contribution of this work is our novel semi-supervised 3D shape segmentation method, which significantly reduces the necessity of preparing expensive fully-labeled 3D shapes.

* Corresponding author.
E-mail address: 21825205@zju.edu.cn (S. Yang).

https://doi.org/10.1016/j.cad.2021.103181
0010-4485/

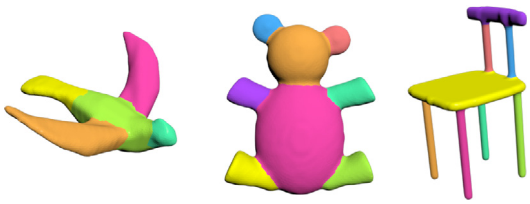

Fig. 1. Example segmentation results of our approach.

The remaining parts of the paper are organized as follows. Section 2 gives a brief review of related work on 3D shape segmentation. Section 3 describes our novel semi-supervised 3D shape segmentation method, our corresponding optimization model, and the overall framework of our algorithm. In Section 4, we show our experimental results, including the comparisons between our method and the state-of-the-art. Finally, we discuss the limitation and future work in Section 5 and conclude the paper in Section 6.

2. Prior work

Shape segmentation is one of the fundamental research topics in computer graphics. It aims to segment a 3D shape, usually represented by a 3D mesh, into semantic parts. Volume-based segmentation [13], surface-based segmentation [14,15] and skeleton-based segmentation [16] are the three main categories of 3D shape segmentation techniques. Early surface-based segmentation methods only process one single shape. Those methods are divided into five classes in [17], including region growing [18], hierarchical clustering [19], boundary segmentation [20], watershed segmentation [21] and iterative clustering [22].

In recent years, with the development of 3D shape repositories, learning-based methods show their superior performance in shape analysis [9,23–25]. Thus learning-based segmentation techniques differ from previous methods in processing a set of shapes instead of a single one with more 3D models available. Learning-based segmentation methods infer more cues from a set of 3D shapes and result in better performance in shape segmentation. There are two main categories of learning-based segmentation methods, including supervised ones and unsupervised ones.

Generally speaking, the training data has both features and labels in supervised learning, which means the connection between features and labels can be learned through a training process. Therefore, the labels can be predicted when facing data with only features but no labels. Different from supervised learning, unsupervised learning is to make the best use of unlabeled samples and discover the structural knowledge in samples.

Supervised segmentation methods. The supervised methods require a lot of labeled 3D shapes as the training dataset. The pioneering work of Kalogerakis et al. [7] proposed a learning-based method for 3D shape segmentation. With the meshes classified and labeled in advance, the method optimized the parameters of an objective function built upon a probabilistic framework and used to label new meshes. Kaick et al. [26] proposed a similar method which learns mesh segmentation features from labeled training shapes. They introduce an objective function consisting of three terms, a face label term, a face–face pair labels binary term, and a probabilistic framework term. In this way, the segmentation results consider not only the geometric features but also the smoothness of the boundaries of different segments.

More recently, two kinds of supervised segmentation approaches have been developed: features-based ones and views-based ones. Features-based methods generate the training data by using 3D shape descriptors, such as Gaussian Curvature (GC) [10], Average Geodesic Distance (AGD) [11], Shape Diameter Function (SDF) [12], Scale-Invariant Heat Kernel Signatures (SIHKS) [27], and Heat Kernel Signatures (HKS) [28]. For example, Xie et al. [1] proposed a fast supervised segmentation algorithm by using multiple geometric feature descriptors and the extreme learning machine algorithm. Guo et al. [6] proposed a segmentation method using deep convolution neural networks.

Instead of using geometric features of 3D shape as the training data, view-based methods try to establish the linkages between the 3D shape and its projections. In [29], a semantic segmentation method is introduced by converting the 3D shapes to their 2D projections. The input training data consists of labeled images of projected shapes. According to the correspondences between the pixels on 2D views and the faces of 3D shapes, 3D shape segmentation can be accomplished as a 2D image segmentation task which has been thoroughly studied. Le et al. [30] proposed a mesh segmentation algorithm which uses both recurrent neural networks and multiple 2D views. By introducing long short term memory, the algorithm can handle the sequential multiple views and obtain satisfactory results on the Princeton Segmentation Benchmark dataset.

Unsupervised segmentation methods. As mentioned above, the supervised segmentation methods demand a considerable labeled dataset. However, labeling is a time-consuming, costly, and tedious process. Therefore, unsupervised segmentation methods have become a very active research topic in the last ten years. This kind of method can be divided into two main categories: descriptor-based methods and alignment-based methods. Descriptor-based methods use 3D shape descriptors to label similar faces with a common label to segment mesh models. Sidi et al. [31] introduced an unsupervised segmentation method for non-rigid shapes. They establish the correspondence among faces by using spectral clustering and diffusion maps. Due to the lack of prior knowledge, their method is slightly inferior compared to the supervised methods.

Zhuang et al. [32,33] proposed two semi-automatic methods to segment a mesh into several patches whose boundaries are aligned with ridge and valley lines by introducing a mesh embedding approach or employing correlation clustering methods respectively. Similar to [31], Wu et al. [34] proposed an unsupervised method by using spectral clustering in the descriptor space. To achieve better results, they employ the over-segmentation method to divide meshes into patches. Besides, Zhang et al. [35] presented an unsupervised segmentation method based on face-level descriptors and soft clustering. Unlike clustering-based unsupervised methods, Shu et al. [2] introduced a deep learning-based segmentation method. Several different geometric feature vectors are concatenated into one high-dimensional vector in their method. Stacked auto-encoders are used in their method to learn low dimensional feature vectors that represent geometric features.

Alignment-based methods segment 3D shapes by utilizing the global alignment of shapes. These methods can achieve better performance by finding the correspondences among the models belonging to the same category. For example, Golovinskiy and Funkhouser [36] introduced an unsupervised segmentation method that consistently segments a set of 3D shapes using graph clustering techniques. Their method establishes the correspondences among parts of models and segments 3D shapes simultaneously. Xu et al. [37] proposed an unsupervised method to address global alignment and scaling of shapes in the 3D shape segmentation process. They analyze shapes in the part-level to decompose each shape into several semantic parts. Huang and Koltun et al. [38] introduced a joint shape segmentation method which is also entirely unsupervised. The results show that they improved the segmentation quality by using a joint segmentation technique.

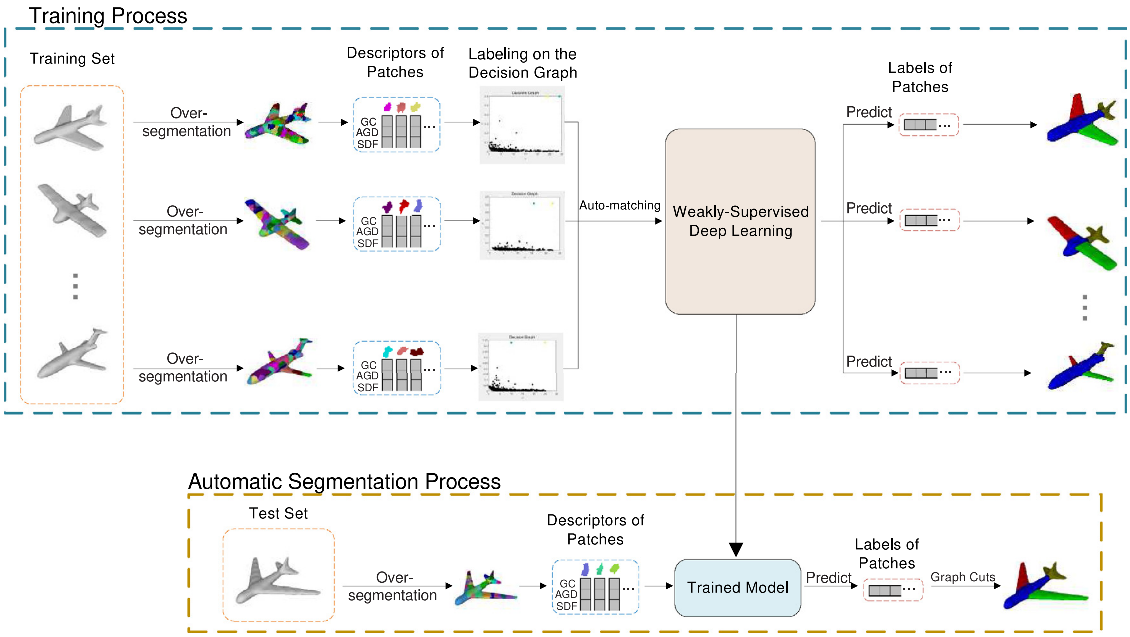

Fig. 2. The workflow of our proposed approach for 3D shape segmentation. Gaussian Curvature (GC) [10], Average Geodesic Distance (AGD) [11], Shape Diameter Function (SDF) [12] are 3D shape feature descriptors.

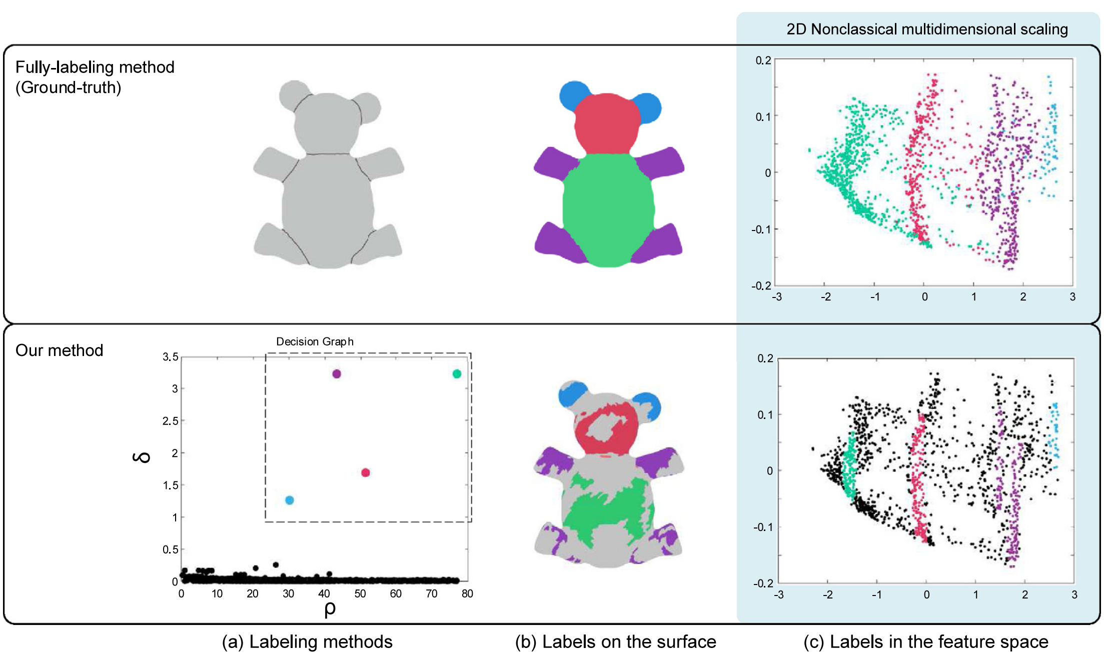

Fig. 3. The results of the fully-labeling method and our labeling method. Column (a) shows the two different methods of labeling a 3D shape. And the labeled results on the surface are showed in column (b), while column (c) shows the labeling results in the feature space. In column (c), all the feature vectors of faces of the 3D shape are embedded into a two-dimensional space. In our approach, we label a 3D shape on the decision graph by selecting appropriate clustering centers, and the labeled result (b) will present in real-time to help users adjust the selecting box.

Generally speaking, fully-supervised segmentation methods usually perform better than unsupervised methods in shape segmentation tasks. However, they do require lots of fully-labeled training data to achieve satisfactory results. Unfortunately, fully labeling numerous 3D shape is a very costly and time-consuming task. To reduce the cost of preparing training data, our method introduces the density peaks clustering algorithm into the labeling process, which significantly reduces the time spent. As shown in Fig. 3, instead of directly labeling faces on the surface of 3D shapes, we do it on the decision graph by adjusting the selection box. The second and the third column respectively show the labeled results on the surface and in the feature space. After that, our method establishes a semi-supervised model to convert the initial labeling shapes into fully-labeled shapes. Different from other methods, our approach can naturally capture the important seed faces, which helps achieve satisfactory performance.

3. Soft density peaks clustering and semi-supervised model

Shape segmentation can be seen as a face clustering task of 3D shapes. In our method, we propose an improved density peak clustering algorithm and combine it with our novel objective function. Based on that, we present our novel semi-supervised 3D shape segmentation method, which greatly simplifies the labeling process by users and ensures high-quality labeling at the same time.

3.1. Labeling based on density peaks clustering

The density peak clustering (DPC, [39]) algorithm provides us with a simple way of clustering by decision graphs. In the DPC algorithm, two parameters are needed to calculate for each data point, including the local density ρ_i and the distance δ_i. In a data points set $S = \{s_1, s_2, \ldots, s_n\}$, the local density ρ_i represents the number of points near the data point s_i, and ρ_i is defined as:

$$\rho_i = \sum_j \chi\left(d_{ij} - d_c\right), \tag{1}$$

where $\chi(x) = 1$ if $x < 0$, otherwise $\chi(x) = 0$, d_{ij} denotes the distance between point s_i and s_j, d_c denotes a cutoff distance. Higher local density ρ_i means more data points are near the point s_i. The distance δ_i is defined as the Euclidean distance between point s_i and the nearest point s_j which has a higher local density:

$$\delta_i = \min_{j:\rho_j > \rho_i}\left(d_{ij}\right), \tag{2}$$

where d_{ij} denotes the Euclidean distance between point s_i and s_j. One can plot all the data points on a two-dimensional decision graph. The vertical axis and the horizontal axis represent the distance δ_i and the local density ρ_i respectively. In the graph, a small set of points at the upper right corner of the decision graph, which have big ρ_i and big δ_i, are deemed as the cluster centers according to the DPC algorithm. When cluster centers are fixed, each point can be assigned to the corresponding class by finding its nearest cluster center of higher density.

Based on a set of selected feature descriptors, we map each triangular face into a point in the feature space. It seems that the DPC algorithm can be used for clustering faces (mapped to points in the feature space) into different groups while each group defines a meaningful part of the 3D shape. However, there are two difficulties:

1. Since the original DPC algorithm runs in $O(n^2)$ time, it becomes impractical for our problem where n may amount to tens of thousands.
2. The DPC algorithm does not provide the confidence to assign a face with a label. Any samples assigned with the same label are considered equal for the results of the DPC algorithm.

A detailed 3D shape usually consists of 10K or even more faces. As mentioned above, too many faces will make the DPC algorithm unpractical. To overcome this problem, we employ the over-segmentation method [20] to divide the 3D mesh model into some small patches, where each one contains dozens of faces. We calculate the feature vectors of all the faces in each patch, and we choose the face closest to the mean level to represent the patch. In this way, the number of input data points for the DPC algorithm is greatly reduced, so that the results can be obtained in reasonable computation time.

To address the second issue, we extend the DPC algorithm in this paper to produce probability distributions instead of exact classification labels. Provided cluster center k_t and its corresponding label l_t, we define the probability that k_i belongs to category l_t as:

$$P_{i,t} = \exp\left(-\frac{d_{i,t}^2}{2\sigma^2}\right), \tag{3}$$

where $\sigma = \tan^{-1}(\rho_i)$ and $d_{i,t}$ denotes the distance between the two feature vectors of patches k_i and k_t. As a post-processing step, all $P_{i,t}$ are normalized and regarded as the corresponding probability distribution of k_i. In this paper, we name our extended DPC as soft density peaks clustering (SDPC) algorithm.

The training data of our method is obtained by employing a fast marking method based on the SDPC. As shown in Fig. 4, we label 3D shapes on the decision graph of the SDPC algorithm. Users adjust the selection box from the top right to choose appropriate clustering centers on the decision graph. The visually labeled 3D shape will be updated in real-time on the right side, assisting users during the selection process. In Fig. 4(a) and (b), if the user selects insufficient clustering centers, the visually labeled 3D shape immediately indicates inadequate labeling. In this case, the user should continue to expand the selection box. However, if too many clustering centers are selected, as shown in (d), we can find the shape is over labeled. When a satisfactory label is marked, as shown in (c), we regard it as a part of the training data. In this way, massive 3D shapes can be fast labeled with relatively high-quality labels.

In our method, three well-known 3D shape feature descriptors, including Average Geodesic Distance (AGD) [11], Shape Diameter Function (SDF) [12], and Gaussian Curvature (GC) [10], are selected and concatenated into 3-dimensional feature vectors. For each patch getting from the over-segmentation step, we regard the feature vector on the face selected by the policy mentioned above as the feature vector of the patch. The experimental results show that the three chosen feature descriptors perform well.

3.2. Objective function

With our extended soft density peaks clustering, we can get the preliminary result of 3D shape segmentation according to the probability distribution of each face belonging to each class. Inspired by the previous work proposed in [40], to get the exact label of each face, we further propose a novel objective function, which considers the adjacency relationship and geometric features. By minimizing the objective function, each face can be assigned an appropriate label.

There are four energy terms in our objective function:

$$E = E_{dpc}(L) + E_{dnn}(L, W) + \lambda E_{fea}(L) + \gamma E_{smooth}(L), \tag{4}$$

where

$$\begin{cases} E_{dpc}(L) = \sum_i E_{dpc}^i(l_i), \\ E_{dnn}(L, W) = \sum_i E_{dnn}^i(l_i, W), \\ E_{fea}(L) = \sum_{i,j} E_{fea}^{i,j}\left(l_i, l_j\right), \\ E_{smooth}(L) = \sum_{i,j} E_{smooth}^{i,j}\left(l_i, l_j\right). \end{cases} \tag{5}$$

L represents all the labels of patches, W denotes the weights of deep neural networks, and parameters λ and γ are empirically set to 1.0 and 0.13 respectively. We minimize the objective function to obtain our trained model.

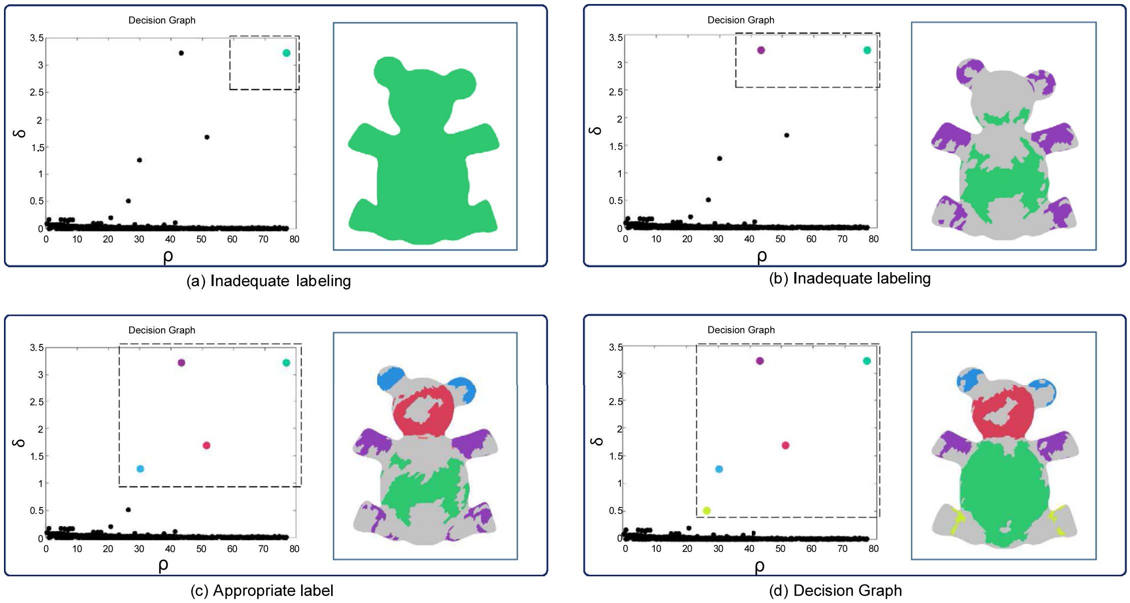

Fig. 4. The labeling process of our method. We label a 3D shape on the decision graph, inspired by the density peaks clustering algorithm [39]. Starting from the top right of the decision graph, users choose the appropriate clustering centers according to the visual labeled 3D shape on the right side. The labeled model will be updated in real-time as the different clustering centers are selected.

To make it clear, we assume that there is a 3D shape S, which is divided into k patches $s_1, s_2, \ldots, s_k$ by the over-segmentation method [20] in advance, and we want to segment S into n parts. Let matrix $P^{k\times n}$ denote the probability distribution obtained from the soft density peaks clustering, where $P_{i,t}$ represents the probability of the ith patch belonging to the tth part. For the objective function, we have the following considerations.

First of all, the segmentation results should respect the probability matrix $P^{k\times n}$. We build an energy term E^i_{dpc} to represent the prior result of SDPC, which is a column vector defined as follows. And the predicted results of our learned model should be in agreement with the final segmentation results. Thus, we train a deep neural network to learn the mapping between geometric features of patches and their labels, which can be represented by a term $E^i_{dnn}(l_i, W)$. And patches with similar feature vectors should be classified into the same segment [40], which is represented by a pairwise term $E^{i,j}_{fea}(l_i, l_j)$ in our optimization problem. In the end, we build an energy term $E^{i,j}_{smooth}(l_i, l_j)$ for the smoothness of the segmentation results.

$$E^i_{dpc}(l_i) = -\log P_{i,l_i} \tag{6}$$

where $i = 1, 2, \ldots, k$, l_i is the label of the ith patch, and P_{i,l_i} is the value of ith row and l_ith column of the probability matrix $P^{k\times n}$. The smaller E^i_{dpc} means that the label of the ith patch is more consistent with the clustering results of our SDPC algorithm.

$$E^i_{dnn}(l_i, W) = -\log Pre(l_i, W), \tag{7}$$

where $i = 1, 2, \ldots, k$, W represents the weights of the deep neural network, and $Pre(l_i, W)$ denotes the predicted probability of the ith patch belonging to the l_i-th part.

$$E^{i,j}_{fea}(l_i, l_j) = \begin{cases} \exp\left(-\frac{\|g(s_i)-g(s_j)\|_2^2}{\delta^2}\right) & \text{if } l_i \neq l_j \\ 0 & \text{otherwise,} \end{cases} \tag{8}$$

where $g(s_i)$ and $g(s_j)$ denote the feature vectors of patches s_i and s_j. The energy term $E^{i,j}_{fea}$ represents that there should be wide gaps between the feature vectors with different category labels.

$$E^{i,j}_{smooth}(l_i, l_j) = \begin{cases} -\log\left(\frac{\theta(f_i, f_j)}{\pi}\right) & \text{if } l_i \neq l_j \\ 0 & \text{otherwise,} \end{cases} \tag{9}$$

where $\theta(f_i, f_j)$ denotes the dihedral angle between neighboring facets f_i and f_j. Note that concave angles and convex angles are weighted by 1 and 0.1 respectively. The energy term $E^{i,j}_{smooth}$ represents that the boundary between any two segments should be aligned with the concave edges from people's subjective cognition.

It is worth pointing out that the training 3D shapes are naturally segmented during the optimization process. Meanwhile, the trained neural network can be used to predict and segment new 3D shapes. Note that our objective function contains two groups of variables to optimize, L and W. The optimization process is described in Section 3.3 in detail.

3.3. Optimization

Note that the objective function E can be divided into two parts, where the first part $\{E_{dpc}(L)+E_{fea}(L)+E_{smooth}(L)\}$ is only related to the labels L, and the second part $\{E_{dnn}(L, W)\}$ couples the weights W and the labels L. We use the alternating optimization technique to minimize the objective function E, which iteratively fixes W to solve for L and then fixes L to solve for W.

Fixing L to solve for W is actually a training process of neural networks. Neural networks try to learn nonlinear mapping functions between the patches' feature vectors and their labels. In this paper, a deep neural network with four layers is used. The inputs are feature vectors $F = \{f_1, f_2, \ldots, f_k\}$, and the output are the labels L.

Fixing W to solve for L is a process of adjusting the labels of the patches. The first part $\{E_{dpc}(L) + E_{fea}(L) + E_{smooth}(L)\}$ is a simple function depending on the labels L. We employ the graph cuts algorithm [41] to efficiently solve this problem. The alternating process repeats several times until the labels of patches do not change anymore, or getting a satisfactory result after reaching

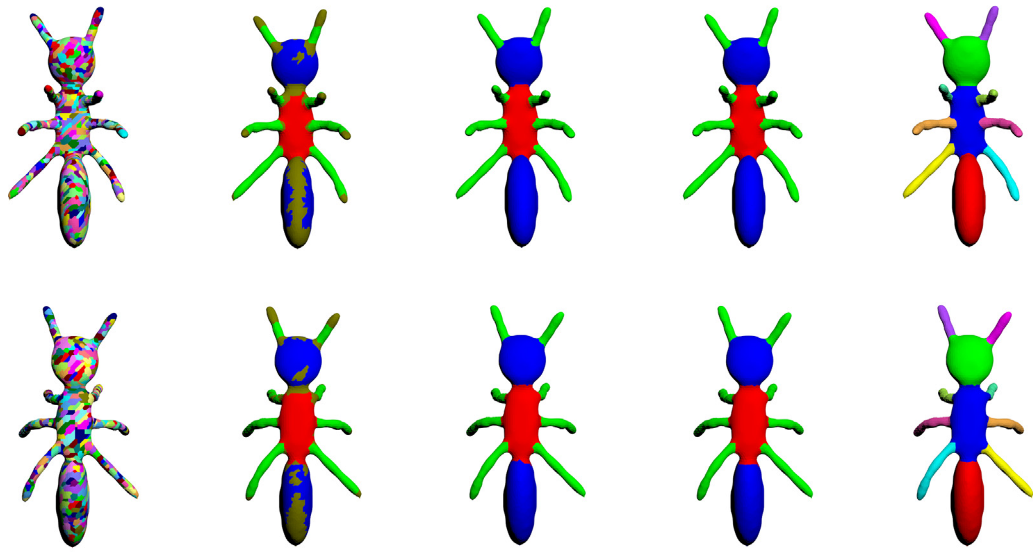

Fig. 5. The iterative optimization process for an Ant model. The top row shows the top view of the Ant model, while the second row shows the bottom view. We start by over-segmenting it into patches. After the interaction with the user, we iteratively minimize the objective function (corresponding to Step4 and Step5 of our algorithm), with segmentation results progressively becoming better. Finally, the satisfactory result is obtained when our algorithm finishes.

the user-given iteration count. An example of our optimization process is shown in Fig. 5.

3.4. Algorithm

Our algorithm can be summarized as follows.

Semi-supervised segmentation algorithm

Inputs:
3D shapes
Outputs:
Labels of every face
Method:
Step1: Divide the model into small patches with the over-segmentation algorithm.
Step2: Extract the feature vectors of patches.
Step3: Label the training shapes on the decision graph of the Soft Density Peaks Clustering.
Step4: Optimize energy term E_{dnn} by training the deep neural network.
Step5: Optimize energy term $E_{dpc} + E_{fea} + E_{smooth}$ by employing the graph cuts algorithm.
Step6: Repeat Step4 and Step5 until labels do not change or a user-given maximum iteration count is reached.

Our method provides a novel semi-supervised way to segment 3D shapes by combining soft density peaks clustering and optimization of the objective function. The users only need to choose the prominent points from the decision graph of density peaks clustering throughout the whole process. Then, our method can automatically obtain the desired segmentation results.

4. Evaluation

Experimental dataset. We evaluate our method on the Princeton Segmentation Benchmark (PSB) dataset [42] and the Shape COSEG Dataset [43]. The PSB dataset provides 19 different categories of shapes, and each category contains 20 shapes. The PSB dataset [42] also contains ground-truth segmentation results provided by human users. The Shape COSEG Dataset consists of two main parts, including a small set and a large set. The small one contains 8 different categories and 190 shapes in total. While, the large one includes 400 Chair shapes, 300 Vase shapes, and 200 Alien shapes. The corresponding ground-truth is also provided by [43].

Evaluation metric. The main evaluation metric used in this paper is the Rand Index (RI) defined by [42], which is a comprehensive evaluation metric. The Rand Index counts for the differences between the two segmentation results. The Rand index value is usually computed by comparing segmenting results with hand-labeled segmentation (ground truth). The lower the Rand Index is, the more similar the two segmentations are, i.e., better performance.

In this section, we exhibit the average Rand Index value of each category of models and compare it with other supervised and unsupervised methods. Besides, we also use Accuracy to measure the segmentation performance. The Accuracy is defined as the ratio of the correctly predicted area on the surface of 3D shapes over the whole area. Note that the labels of some segments are reassigned appropriately in order to compute the accuracies of predictions conveniently.

Parameters. The deep neural network used in this paper contains 4 layers. The input layer is set to match the dimensions of the input data automatically. And the output layer is set to the number of clustering centers.

4.1. Results

As mentioned in Section 3, our method is a semi-supervised one. In the first stage, we partly label the training dataset through a very simple way which picks clustering centers on the decision graph and trains a neural network model based on the clustering results. In the process, the segmentation results of training shapes have been naturally produced. In the second stage, testing 3D shapes are segmented automatically by applying the learned neural network model. Thus, our method produces two types of segmentation results, one for the training 3D shapes and one for the testing 3D shapes.

Figs. 6 and 7 show some segmentation results of training 3D shapes and testing 3D shapes, respectively. Note that each category in the PSB dataset contains 20 different shapes. We

Fig. 6. Representative segmentation results of our training process on the PSB dataset.

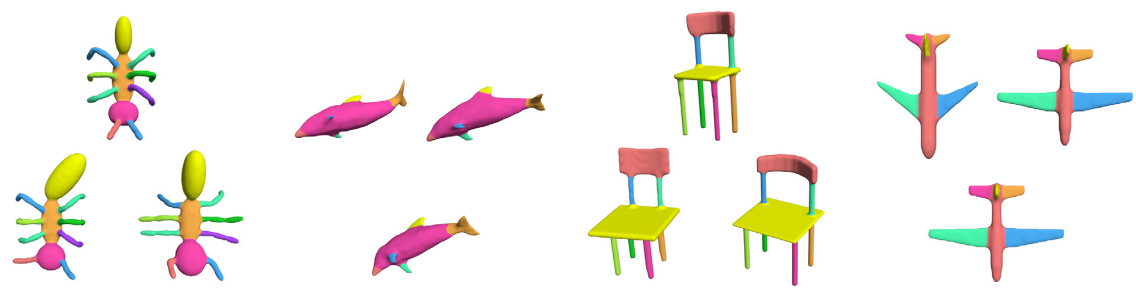

Fig. 7. Representative segmentation results of our testing process on the PSB dataset.

Table 1
Comparison of our method and 6 other unsupervised methods on the PSB dataset with the Rand Index scores.

Object categories	Ours	WcSeg	CoreExtra	FitPrim	NormCuts	RandCuts	RandWalks
Human	**0.119**	0.128	0.189	0.159	0.163	0.132	0.363
Cup	**0.036**	0.165	0.291	0.436	0.392	0.277	0.311
Glasses	0.150	0.175	0.287	0.225	0.147	**0.105**	0.395
Airplane	**0.070**	0.089	0.243	0.172	0.185	0.130	0.283
Ant	**0.016**	0.021	0.062	0.128	0.097	0.044	0.092
Chair	**0.059**	0.105	0.185	0.214	0.113	0.189	0.202
Octopus	**0.019**	0.029	0.052	0.128	0.106	0.088	0.135
Table	**0.035**	0.089	0.222	0.318	0.240	0.402	0.141
Teddy	**0.050**	0.057	0.114	0.138	0.122	0.067	0.116
Hand	**0.054**	0.112	0.155	0.196	0.165	0.116	0.236
Plier	0.087	**0.086**	0.094	0.194	0.208	0.126	0.277
Fish	**0.118**	0.203	0.268	0.433	0.387	0.299	0.381
Bird	**0.050**	0.103	0.124	0.204	0.192	0.124	0.250
Armadillo	0.098	**0.080**	0.133	0.102	0.199	0.086	0.200
Bust	**0.157**	0.264	0.314	0.313	0.340	0.247	0.338
Mech	**0.106**	0.173	0.327	0.343	0.337	0.327	0.241
Bearing	**0.078**	0.105	0.367	0.162	0.166	0.139	0.275
Vase	**0.141**	0.162	0.231	0.281	0.326	0.162	0.253
Fourleg	**0.150**	0.153	0.184	0.198	0.220	0.183	0.431
Average	**0.084**	0.121	0.202	0.229	0.216	0.171	0.259

randomly select 15 of 20 as the training shapes, and the remaining shapes are regarded as the testing shapes. Compared with the ground-truth segmentation results, satisfactory segmentation results are achieved on both the training and testing shapes. And on the testing set, a comparison of our method with and without the graph cuts is shown in Fig. 8.

4.2. Comparison

To further evaluate the performance of our method, we compare it with other unsupervised methods and supervised methods on the PSB dataset and the Shape COSEG dataset.

Comparison with unsupervised methods. As shown in Table 1, we compare our method with other 6 unsupervised methods, including concavity analysis (WcSeg, [48]), Core extraction [49], Fitting primitive [50], Normalized cuts [20], Randomized cuts [20], and Random walks [51]. We calculate the average Rand Index score of each category of shapes in the PSB dataset and list the results of our method and 6 other unsupervised segmentation methods. The detailed statistics of the Rand Index score are shown in Table 1. We can see that most of our results achieve the best performance with the lowest Rand Index score compared with unsupervised methods. Our semi-supervised method obtains an average Rand Index score of 0.084, which performs better than the other 6 unsupervised algorithms. Fig. 9 shows a

Table 2
The comparison of Rand Index scores among our method, the fully supervised learning method proposed by Kalogerakis et al. (SB19, [7]), the two semi-automatic mesh segmentation methods proposed by Zhuang et al. (SEMI-1, [32] and SEMI-2, [33]), and fully supervised deep learning method (FSDL, [6]) on the PSB dataset. Note that the segmentation results for the training 3D shapes are also generated by our method in the first stage.

Algorithms	Human	Cup	Glasses	Airplane	Ant	Chair	Octopus	Table	Teddy	Hand
Our method (training models)	**0.120**	0.097	0.154	0.084	**0.017**	0.062	**0.018**	0.062	0.036	0.096
Our method (testing models)	0.128	0.112	0.165	0.123	0.019	0.103	0.031	0.096	0.045	0.111
SB19	**0.120**	**0.090**	**0.135**	**0.079**	0.018	**0.055**	0.019	0.062	0.033	0.101
FSDL	0.135	0.093	0.150	0.085	0.020	0.059	0.021	**0.048**	**0.030**	**0.095**
SEMI-1	0.185	0.103	0.181	0.121	0.050	0.094	0.051	0.074	0.056	0.118
SEMI-2	0.162	0.099	0.176	0.103	0.034	0.079	0.039	0.070	0.045	0.104

Algorithms	Plier	Fish	Bird	Armadillo	Bust	Mech	Bearing	Vase	Fourleg	Average
Our method (training models)	0.084	0.160	**0.077**	0.098	**0.223**	0.110	**0.070**	0.141	0.148	0.095
Our method (testing models)	0.106	0.166	0.079	0.106	0.233	0.116	0.103	0.163	0.154	0.114
SB19	**0.054**	**0.128**	0.078	**0.069**	0.241	**0.109**	**0.070**	0.150	**0.136**	**0.092**
FSDL	0.070	0.135	0.082	0.075	0.235	0.112	0.072	**0.131**	0.142	0.094
SEMI-1	0.103	0.196	0.098	0.132	0.341	0.178	0.101	0.184	0.167	0.133
SEMI-2	0.094	0.178	0.094	0.118	0.301	0.151	0.089	0.165	0.134	0.118

Table 3
The comparison of average segmentation accuracy (%) between our method and other methods, including the unsupervised segmentation methods proposed by Sidi et al. [31], Hu et al. [44], Wu et al. [34], and weakly supervised segmentation method proposed by Kim et al. [45] on the small Shape COSEG Dataset.

Algorithms	Candelabra	Chairs	Fourleg	Goblets	Guitars	Lamps	Vases	Irons	Average
Our method	**97.9**	**98.7**	87.0	95.6	94.2	**98.7**	**94.2**	**96.5**	**95.4**
Sidi	84.4	84.8	77.3	98.2	87.2	94.3	87.4	N/A	87.7
Hu	93.9	89.6	**88.7**	**99.2**	**98.0**	90.7	80.2	N/A	91.5
Wu	97.2	87.6	87.7	98.7	97.4	98.4	83.4	79.4	92.7
Kim	87.9	98.4	87.1	98.1	96.5	97.6	83.2	N/A	92.9

Table 4
The comparison of average segmentation accuracy (%) between our method and other two fully supervised segmentation methods proposed by Hanocka et al. [46], Lahav et al. [47] for each category model on the large Shape COSEG Dataset.

Algorithms	Vases	Chairs	Telealiens	Mean
Our method	95.5	98.2	97.3	97.0
MeshCNN	97.3	**99.6**	97.6	98.2
MeshWalker	**98.7**	**99.6**	**99.1**	**99.1**

(a) (b) (c)

Fig. 8. Comparison of our method on the testing set with and without the graph cuts. (a) shows the results without the graph cuts. (b) shows the results with the graph cuts. (c) shows the ground truth.

qualitative comparison of our method and the other methods on the Airplane, Cup, and Bird models.

Comparison with supervised methods and semi-automatic mesh segmentation methods. We make comparisons to two other fully supervised 3D shape segmentation methods, including SB19 [7] and FSDL [6], and two semi-automatic methods, including SEMI-1 [32] and SEMI-2 [33]. As shown in Table 2, the Rand Index scores of our method on training shapes for 4 categories are better than the results of FSDL and SB19. Meanwhile, the results on the training shapes getting from our method are better than those from SB19 for 6 categories and those from FSDL for 8 categories. Also, the results on the testing shapes from our method are worse than those from SB19 and FSDL for most categories. Note that our method achieves slightly better performance than the two semi-automatic mesh segmentation methods. We think the main reason is that those two methods are more time-consuming than our method, and it is not so easy for users to get satisfactory segmentation results on the whole dataset in a limited time with them.

Fig. 10 shows a qualitative comparison among our method and the other two supervised methods on 6 models in the PSB dataset. Our method obtains segmentation results comparable to the two algorithms mentioned above, although our semi-supervised method can utilize only very limited input from the users.

Results on the Shape COSEG Dataset. To further evaluate our method, we test our algorithm on the Shape COSEG Dataset. Fig. 11 shows the segmentation result of our method on the Candelabra models, Fourleg models, Guitars models, Vases models, and Irons models. We can see that satisfactory results can be obtained with our segmentation algorithm. Additionally, we compare our method with other methods, including the unsupervised segmentation methods proposed by Sidi et al. [31], Hu et al. [44], Wu et al. [34], the weakly supervised segmentation method proposed by Kim et al. [45] and the fully supervised segmentation methods proposed by Hanocka et al. [46], Lahav et al. [47]. On the Shape COSEG Dataset, researchers prefer to use an accuracy score for evaluation. Therefore, we calculate the average segmentation accuracy score for each category of models, including the training set and the testing set on the small Shape COSEG Dataset. The comparison results on the small Shape COSEG Dataset are shown in Table 3. We can see that our method's segmentation accuracies are all over 85% for every category of shapes. The accuracy of our method for 5 in 8 categories is better than other methods, and the other 3 in 8 categories are close to the best results. From Table 4, we can see that although our

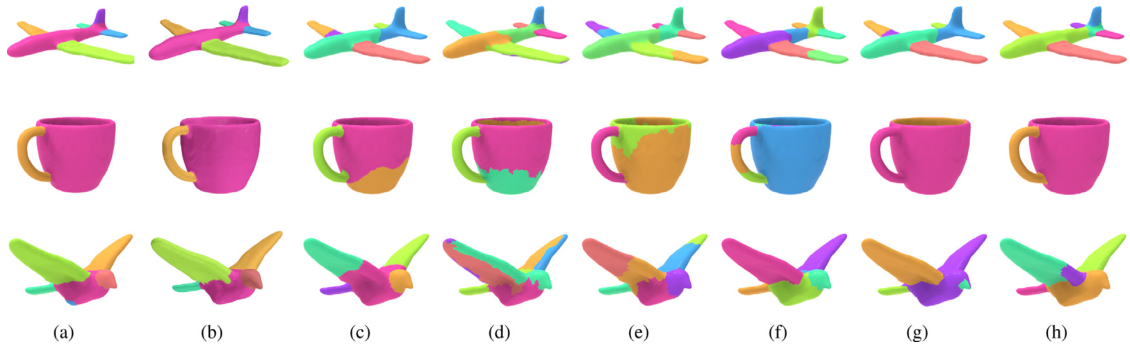

Fig. 9. Comparison with other six segmentation algorithms. The approaches exhibited here include (a) Ground Truth; (b) Ours; (c) CoreExtra; (d) FitPrim; (e) Normalized cuts; (f) Randomized cuts; (g) Random walks; (h) WcSeg.

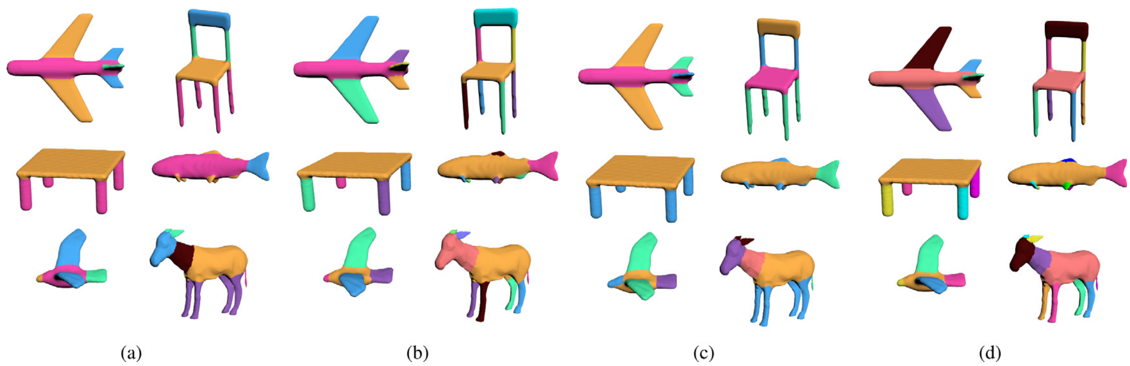

Fig. 10. Qualitative comparison of our method, fully supervised learning method FSDL [6], and SB19 [7]. (a) shows the ground truth; (b) The results of our method; (c) The results of SB19; (d) The results of FSDL.

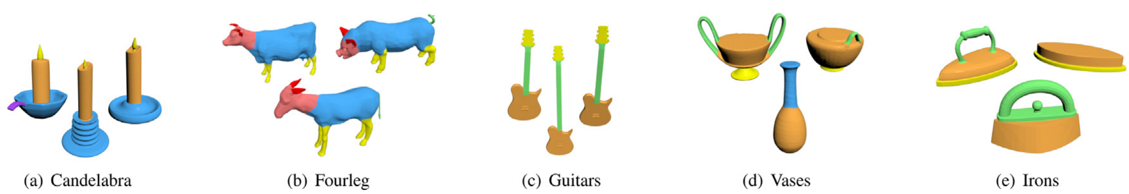

Fig. 11. Representative segmentation results of our approach on the Shape COSEG Dataset.

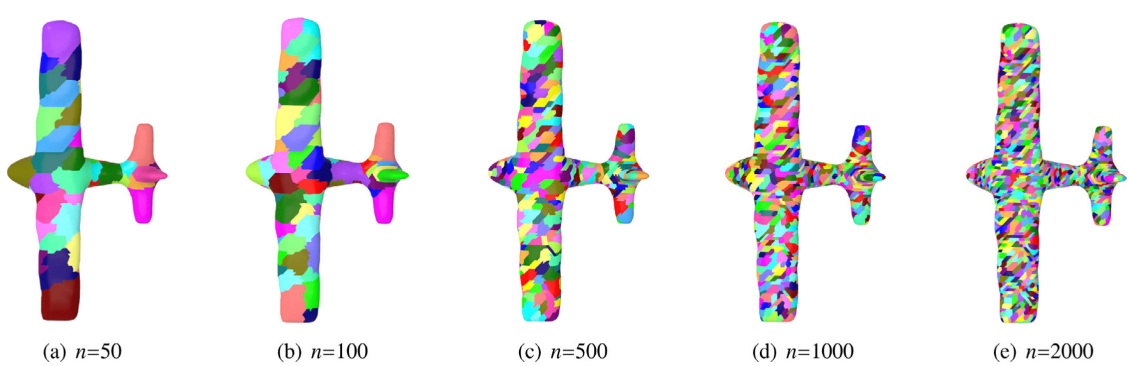

Fig. 12. Over-segmentation results for the different number of patches n.

method performs better on the Shape COSEG Dataset than on the PSB dataset, the other two supervised methods perform better than us on the large Shape COSEG Dataset. We think the main reason is that more training data are provided in the Shape COSEG Dataset than in the PSB dataset.

4.3. Sensitivity to over-segmentation

In our method, all input 3D shapes are divided into n patches by the over-segmentation algorithm in advance. To measure how the segmentation results are affected by the different choices of

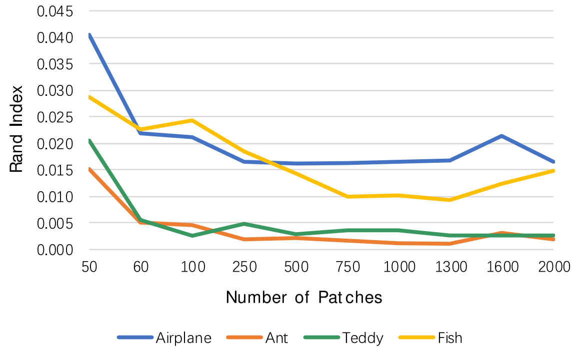

Fig. 13. Our method's average Rand Index scores by the different number of patches n on four categories of testing models.

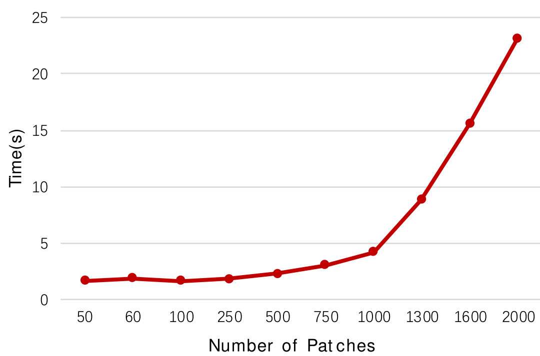

Fig. 14. The average time that our SDPC algorithm consumed with the different number of patches n.

the number of patches n, we test our algorithm with different n for 4 categories of 3D shapes and compute the corresponding average Rand Index scores. Additionally, we measure the average amount of time SPDC takes for different n because increasing n will lead to longer computation time.

Fig. 12 shows the over-segmentation results for varying n. We test 10 different n varying from 50 to 2000. In order to obtain an impartial result, we try our best to make the same choices on the decision graphs. Fig. 13 shows the correspondence between the average Rand Index score and n. Fig. 14 shows the correspondence between the compute time and n. In Fig. 13, with the increase of n, the average Rand Index score of all four kinds of shapes decreases rapidly at first. However, when n is greater than 500, the downward trend becomes not so obvious. In Fig. 14, the time that the SDPC algorithm consumes significantly increases when n exceeds 1000. To balance the segmentation performance and the time consumption, we set $n = 750$ in our method.

4.4. Performance

We implement the proposed algorithm in Matlab and C++. The performance of our approach is measured on a PC with a 3.20 GHz Intel Core™ 12 Quad processor, 16 GB of RAM, and NVIDIA GeForce GTX 2080 Ti GPU. Our semi-supervised method consists of two main phases, the training one and the testing one. For the training phase, the computation time is mainly spent on density peaks clustering, neural network training, and iterative optimization. On average, our algorithm takes about 3 min to train a model. Segmenting an unlabeled model takes about 30 s in the testing phase. It takes about 50 min to go through the pipeline for a category model (20 shapes total, 15 training shapes and 5 testing shapes) on the PSB dataset.

5. Limitation and future work

Our algorithm assumes that all similar shapes are classified into one category in advance. We train and test our method on one category of models because our method relies on the correspondences between 3D shapes to learn neural networks.

Second, our method employs various 3D shape descriptors and graph cuts algorithm to obtain segmentation results. Thus, the input 3D shapes of our algorithm must be manifold. Besides, our method still depends on user interactions, which needs to be improved in the future. A possible way to strengthen our work is to find the clustering centers automatically, which is one of our future works.

6. Conclusion

In this paper, we propose a semi-supervised method for 3D shape segmentation. It simplifies the full manual labeling process that previous learning-based approaches employ. With a simple selecting process on a decision graph of SDPC, our method dramatically reduces the time spent on manual labeling. We validate our method on the Princeton Shape Benchmark and the Shape COSEG Dataset and compare it with other segmentation methods. The experimental results show that our method performs significantly better than the existing unsupervised algorithms and achieves performance comparable to fully supervised algorithms and other semi-supervised methods.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. We would also like to thank Oliver van Kaick, Zizhao Wu, Yunhai Wang, and Ruizhen Hu for their kind help. This work is supported by the National Natural Science Foundation of China (61872321, 62172356, 62025207, 61772016, 61972350), Natural Science Foundation of Zhejiang Province, China (LY22F021018), National Science Foundation (IIS-1617172, IIS-1622360, IIS-1764071), Open Research Projects of Zhejiang Lab, China (2019NB0AB03), and Ningbo Major Special Projects of the "Science and Technology Innovation 2025, China" (2020Z005, 2020Z007, 2021Z012).

References

[1] Xie Z, Xu K, Liu L, Xiong Y. 3D Shape segmentation and labeling via extreme learning machine. Comput Graph Forum 2014;33(5):85–95.
[2] Shu Z, Qi C, Xin S, Hu C, Wang L, Zhang Y, et al. Unsupervised 3D shape segmentation and co-segmentation via deep learning. Comput-Aided Geom Des 2016;43:39–52.
[3] Yi L, Kim VG, Ceylan D, Shen I-C, Yan M, Su H, et al. A scalable active framework for region annotation in 3D shape collections. ACM Trans Graph 2016;35(6):1–12.
[4] Zhu C, Xu K, Chaudhuri S, Yi L, Guibas LJ, Zhang H. AdaCoSeg: Adaptive shape co-segmentation with group consistency loss. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020, pp. 8543–52.
[5] Zhang F, Sun Z, Song M, Lang X. Progressive 3D shape segmentation using online learning. Comput Aided Des 2015;58:2–12.
[6] Guo K, Zou D, Chen X. 3D Mesh labeling via deep convolutional neural networks. ACM Trans Graph 2015;35(1):1–12.
[7] Kalogerakis E, Hertzmann A, Singh K. Learning 3D mesh segmentation and labeling. ACM Trans Graph 2010;29(4):1–12.

[8] Li S, Zhang C, He X. Shape-aware semi-supervised 3D semantic segmentation for medical images. In: Medical image computing and computer assisted intervention – MICCAI 2020. Cham: Springer International Publishing; 2020, p. 552–61.
[9] Meng M, Xia J, Luo J, He Y. Unsupervised co-segmentation for 3D shapes using iterative multi-label optimization. Comput Aided Des 2013;45:312–20.
[10] Gal R, Cohen-Or D. Salient geometric features for partial shape matching and similarity. ACM Trans Graph 2006;25(1):130–50.
[11] Shapira L, Shalom S, Shamir A, Cohen-Or D, Zhang H. Contextual part analogies in 3D objects. Int J Comput Vis 2010;89(2):309–26.
[12] Shapira L, Shamir A, Cohen-Or D. Consistent mesh partitioning and skeletonisation using the shape diameter function. Vis Comput 2008;24(4):249–59.
[13] Chazelle BM. Convex decompositions of polyhedra. In: Proceedings of the thirteenth annual ACM symposium on theory of computing. ACM; 1981, p. 70–9.
[14] Bergamasco F, Albarelli A, Torsello A. Semi-supervised segmentation of 3D surfaces using a weighted graph representation. In: International workshop on graph-based representations in pattern recognition. Springer; 2011, p. 225–34.
[15] Fang Y, Sun M, Kim M, Ramani K. Heat-mapping: A robust approach toward perceptually consistent mesh segmentation. In: Proceedings of IEEE computer vision and pattern recognition. IEEE; 2011, p. 2145–52.
[16] Tierny J, Vandeborre J-P, Daoudi M. Topology driven 3D mesh hierarchical segmentation. In: IEEE international conference on shape modeling and applications. IEEE; 2007, p. 215–20.
[17] Rodrigues RS, Morgado JF, Gomes AJ. Part-based mesh segmentation: A survey. Comput Graph Forum 2018;37(6):235–74.
[18] Chazelle B, Dobkin DP, Shouraboura N, Tal A. Strategies for polyhedral surface decomposition: an experimental study. Comput Geom 1997;7(5–6):327–42.
[19] Katz S, Tal A. Hierarchical mesh decomposition using fuzzy clustering and cuts. ACM Trans Graph 2003;22(3):954–61.
[20] Golovinskiy A, Funkhouser T. Randomized cuts for 3D mesh analysis. ACM Trans Graph 2008;27(5):1–12.
[21] Mangan AP, Whitaker RT. Surface segmentation using morphological watersheds. In: Proc. IEEE visualization. 1998.
[22] Shlafman S, Tal A, Katz S. Metamorphosis of polyhedral surfaces using decomposition. Comput Graph Forum 2002;21(3):219–28.
[23] Bu S, Liu Z, Han J, Wu J, Ji R. Learning high-level feature by deep belief networks for 3-D model retrieval and recognition. IEEE Trans Multimed 2014;16(8):2154–67.
[24] Xie J, Dai G, Fang Y. Deep multimetric learning for shape-based 3D model retrieval. IEEE Trans Multimed 2017;19(11):2463–74.
[25] Shu Z, Xin S, Xu H, Kavan L, Wang P, Liu L. 3D Model classification via principal thickness images. Comput Aided Des 2016;78:199–208.
[26] van Kaick O, Tagliasacchi A, Sidi O, Zhang H, Cohen-Or D, Wolf L, et al. Prior knowledge for part correspondence. Comput Graph Forum 2011;30(2):553–62.
[27] Bronstein MM, Kokkinos I. Scale-invariant heat kernel signatures for non-rigid shape recognition. In: IEEE conference on computer vision and pattern recognition. 2010, p. 1704–11.
[28] Raviv D, Bronstein MM, Bronstein AM, Kimmel R. Volumetric heat kernel signatures. In: Proceedings of the ACM workshop on 3D object retrieval. ACM; 2010, p. 39–44.
[29] Wang Y, Gong M, Wang T, Cohen-Or D, Zhang H, Chen B. Projective analysis for 3D shape segmentation. ACM Trans Graph 2013;32(6):1–12.
[30] Le T, Bui G, Duan Y. A multi-view recurrent neural network for 3D mesh segmentation. Comput Graph 2017;66:103–12.
[31] Sidi O, van Kaick O, Kleiman Y, Zhang H, Cohen-Or D. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. ACM Trans Graph 2011;30(6):1–10.
[32] Zhuang Y, Zou M, Carr N, Ju T. Anisotropic geodesics for live-wire mesh segmentation. Comput Graph Forum 2014;33(7):111–20.
[33] Zhuang Y, Dou H, Carr N, Ju T. Feature-aligned segmentation using correlation clustering. Comput Vis Media 2017;3(2):147–60.
[34] Wu Z, Wang Y, Shou R, Chen B, Liu X. Unsupervised co-segmentation of 3D shapes via affinity aggregation spectral clustering. Comput Graph 2013;37(6):628–37.
[35] Zhang F, Sun Z, Song M, Lang X, Yan H. 3D Shapes co-segmentation by combining fuzzy c-means with random walks. In: 2013 international conference on computer-aided design and computer graphics. IEEE; 2013, p. 16–23.
[36] Golovinskiy A, Funkhouser T. Consistent segmentation of 3D models. Comput Graph 2009;33(3):262–9.
[37] Xu K, Li H, Zhang H, Cohen-Or D, Xiong Y, Cheng Z-Q. Style-content separation by anisotropic part scales. ACM Trans Graph 2010;29(6):1–10.
[38] Huang Q, Koltun V, Guibas L. Joint shape segmentation with linear programming. ACM Trans Graph 2011;30(6):1–12.
[39] Rodriguez A, Laio A. Clustering by fast search and find of density peaks. Science 2014;344(6191):1492–6.
[40] Shu Z, Shen X, Xin S, Chang Q, Feng J, Kavan L, et al. Scribble based 3D shape segmentation via weakly-supervised learning. IEEE Trans Vis Comput Graphics 2020;26(8):2671–82.
[41] Boykov Y, Veksler O, Zabih R. Fast approximate energy minimization via graph cuts. IEEE Trans Pattern Anal Mach Intell 2001;23(11):1222–39.
[42] Chen X, Golovinskiy A, Funkhouser T. A benchmark for 3D mesh segmentation. ACM Trans Graph 2009;28(3):1–12.
[43] Wang Y, Asafi S, van Kaick O, Zhang H, Cohen-Or D, Chen B. Active co-analysis of a set of shapes. ACM Trans Graph 2012;31(6):1–10.
[44] Hu R, Fan L, Liu L. Co-segmentation of 3D shapes via subspace clustering. Comput Graph Forum 2012;31(5):1703–13.
[45] Kim VG, Li W, Mitra NJ, Chaudhuri S, DiVerdi S, Funkhouser T. Learning part-based templates from large collections of 3D shapes. ACM Trans Graph 2013;32(4):1–12.
[46] Hanocka R, Hertz A, Fish N, Giryes R, Fleishman S, Cohen-Or D. MeshCNN: A network with an edge. ACM Trans Graph 2019;38(4):1–12.
[47] Lahav A, Tal A. MeshWalker: Deep mesh understanding by random walks. ACM Trans Graph 2020;39(6):1–13.
[48] Kaick OV, Fish N, Kleiman Y, Asafi S, Cohen-OR D. Shape segmentation by approximate convexity analysis. ACM Trans Graph 2014;34(1):1–11.
[49] Katz S, Leifman G, Tal A. Mesh segmentation using feature point and core extraction. Vis Comput 2005;21(8):649–58.
[50] Attene M, Falcidieno B, Spagnuolo M. Hierarchical mesh segmentation based on fitting primitives. Vis Comput 2006;22(3):181–93.
[51] Lai YK, Hu SM, Martin RR, Rosin PL. Fast mesh segmentation using random walks. In: Proceedings of ACM symposium on solid and physical modeling. 2008, p. 183–91.