Scribble based 3D shape segmentation via weakly-supervised learning

Zhenyu Shu, Xiaoyong Shen, Shiqing Xin*, Qingjun Chang, Jieqing Feng, Ladislav Kavan, and Ligang Liu

Abstract—Shape segmentation is a fundamental problem in shape analysis. Previous research shows that prior knowledge helps to improve the segmentation accuracy and quality. However, completely labeling each 3D shape in a large training data set requires a heavy manual workload. In this paper, we propose a novel weakly-supervised algorithm for segmenting 3D shapes using deep learning. Our method jointly propagates information from scribbles to unlabeled faces and learns deep neural network parameters. Therefore, it does not rely on completely labeled training shapes and only needs a really simple and convenient scribble-based partially labeling process, instead of the extremely time-consuming and tedious fully labeling processes. Various experimental results demonstrate the proposed method's superior segmentation performance over the previous unsupervised approaches and comparable segmentation performance to the state-of-the-art fully supervised methods.

Index Terms—3D shapes, Segmentation, Scribble, Weakly-supervised, Deep learning

1 Introduction

Automatic segmentation of 3D shapes is a fundamental problem in shape analysis [1]. It is also central to many computer graphics problems, including mesh parameterization [2], skeleton extraction [3], resolution modeling [4], shape retrieval [5] and so on. A commonly used idea in traditional segmentation algorithms is to first map all the triangular faces on 3D shapes, which are usually represented by 3D triangular meshes, into feature space using shape signatures, then clustering the faces on a mesh into several clusters in the feature space, and finally transforming the result onto the dual graph of the mesh to obtain the final segmentation result. Earlier segmentation work [1], [6], [7] does not utilize any prior knowledge and focuses on employing unsupervised methods to cluster faces in the feature space to obtain the desired results. Recent work [8], [9], [10] uses prior knowledge to improve performance by classifying faces in the feature space using supervised methods.

There are two trends for improving the performance of 3D shape segmentation. One trend is to use multiple shape signatures simultaneously to improve the ability to describe a shape's geometric features from different perspectives, as a single signature can only characterize a shape from a specific perspective.

- *Zhenyu Shu and Qingjun Chang are with School of Computer and Data Engineering, Ningbo Institute of Technology, Zhejiang University, Ningbo, PR China. E-mail: shuzhenyu@nit.zju.edu.cn (Zhenyu Shu)*
- *Xiaoyong Shen is with Department of Computer Science and Engineering, The Chinese University of Hong Kong, HongKong.*
- *Shiqing Xin is with School of Computer Science and Technology, Shan-Dong University, Jinan, PR China. Corresponding author. E-mail: xinshiqing@163.com (Shiqing Xin)*
- *Jieqing Feng is with State Key Lab of CAD&CG, Zhejiang University, Hangzhou, PR China.*
- *Ladislav Kavan is with School of Computing, University of Utah, Salt Lake City, USA.*
- *Ligang Liu is with Graphics & Geometric Computing Laboratory, School of Mathematical Sciences, University of Science and Technology of China, Anhui, PR China.*

Manuscript received month day, year; revised month day, year.

Fig. 1. Some representative results of our approach.

The other trend is to use and learn from prior knowledge, such as training data, to improve the faces' classification results in feature space. These two techniques work very well and are used in the state-of-the-art segmentation systems [8], [9], [10], [11], [12]. Due to their nature, most existing learning-based 3D shape segmentation methods rely heavily on high-quality training data to gain a significant performance improvement. However, preparing high-quality training data for 3D shape segmentation is quite expensive because manually labeling each face on a 3D shape is a very tedious and time-consuming task, even though some tools [13] have been developed to accelerate the labeling process. As an example, preparing the training data for 380 3D shapes in the Princeton Shape Benchmark dataset, 12 volunteers took about 10 hours to complete the manual labeling.

In this paper, we propose an effective weakly-supervised method for shape segmentation based on deep learning to address the problem of tedious manual labeling. In our method, we use sparse scribble-based labels, rather than the complete high-quality labels to train our 3D shape segmentation model. Our method does not need the tedious and time-consuming labeling process for preparing training data and therefore is much more practical than existing learning-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2892076, IEEE Transactions on Visualization and Computer Graphics

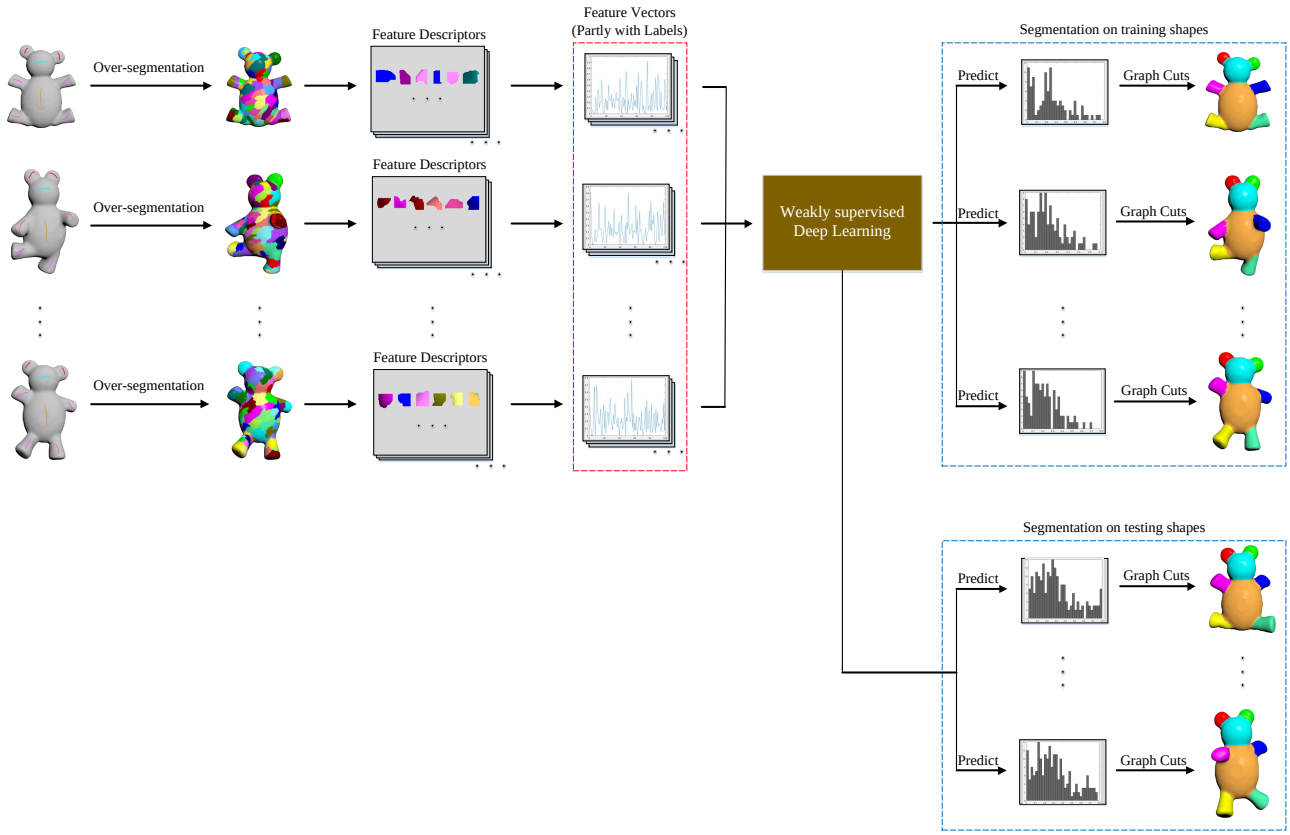


Fig. 2. Workflow of our proposed approach for 3D shape segmentation.

based 3D shape segmentation approaches. Experimental results on the Princeton Shape Benchmark dataset [13] show that our method outperforms other state-of-the-art scribble-based or non-scribble-based segmentation methods, and can achieve comparable or only slightly worse performance when compared with existing fully supervised deep learning-based 3D shape segmentation methods. Figure 1 shows some representative results of our approach, and an overview of our algorithm is shown in Figure 2.

The main contribution of this work is our novel shape segmentation method using weakly-supervised deep learning techniques, eliminating the tedious and time-consuming training data preparation processes.

The remaining parts of the paper are organized as follows. Section 2 reviews the related work on 3D shape segmentation. In Section 3, we give our novel scribble-based deep training model and the overall framework of our algorithm. After that, we show extensive experimental results, as well as comparisons with the state of the art, in Section 4. Finally, we discuss the limitations and future work in Section 5 and conclude this paper in Section 6.

2 Related Work

Shape segmentation [14], [15] aims at dividing a 3D shape into meaningful parts and plays an important role in shape analysis and shape understanding. Until now, many methods have been proposed to address this problem. A common practice is to first extract geometric feature vectors and then apply some decomposition techniques to obtain a satisfactory segmentation result: extreme learning machine [9], approximate convexity analysis [16], concavity-aware fields [17], spectral clustering [18], K-Means [19], core extraction [20], randomized cuts [21], normalized cuts [21], graph cuts [21], [22], random walks [23]. The existing shape segmentation methods can be mainly divided into two categories, including unsupervised 3D shape segmentation and supervised 3D shape segmentation.

Unsupervised 3D shape segmentation. Earlier work focuses on segmenting one single 3D shape without prior knowledge. For example, Sidi and van Kaick et al. [7] proposed an unsupervised method that treats segmentation as a clustering problem in a descriptor space. Golovinskiy and Funkhouser [24] use graph clustering techniques to segment a set of 3D shapes simultaneously. Their assumption is that a global rigid alignment exists between the input shapes, which facilitates iteratively establishing correspondence between respective parts. Huang and Koltun et al. [25] suggested formulating the joint segmentation problem as an integer quadratic programming problem. Experimental results show that segmenting a set of similar 3D shapes simultaneously significantly outperforms traditional segmentation that targets a single shape. Hu and Fan et.al [26] also presented an unsupervised approach for segmenting a set of 3D shapes by over-segmenting the input shapes into primitive patches and then grouping similar patches via a subspace clustering scheme. Recently, Meng and Xia et al. [6] proposed to improve the segmentation results by a multi-label optimization process.

Compared with segmenting a single 3D shape, segment-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2892076, IEEE Transactions on Visualization and Computer Graphics

ing a family of 3D shapes in the same class can achieve better results, assuming that their underlying relevance is effectively utilized. Therefore, more work employing this idea was proposed by being combining with various techniques. Wu and Wang et al. [1] suggested a spectral clustering method that generates consistent segmentation by performing spectral clustering in a fused space of shape descriptors. Xu and Li et al. [27] used anisotropic part scales to part correspondence and their algorithm performs well on a diverse set of shapes. Later, Zheng and Cohen-Or et al. [28] suggested an indirect top-down approach to deal with large shape variations.

A significant advantage of unsupervised 3D shape segmentation is that it does not need any training data, which can be expensive to prepare. However, due to the absence of prior knowledge, their segmentation performance may not be as good as supervised ones.

Supervised 3D shape segmentation. Because 3D shape segmentation can actually be regarded as a face clustering process in the feature space, a data-driven method may be helpful to obtain better results. Therefore, more recent work tries to further improve performance by utilizing prior knowledge during segmentation. For example, Wang and Asafi et al. [29] presented a semi-supervised learning method where the user actively assists in the co-analysis by iteratively providing inputs. Their method requires only a sparse set of constraints to quickly converge toward a consistent and error-free semantic labeling of the set. Kalogerakis and Hertzmann et al. [8] proposed a supervised approach to do labeling and segmentation simultaneously. They showed that the segmentation performance can be dramatically improved by learning techniques. Van Kaick and Tagliasacchi et al.[30] introduced an approach to part correspondence which incorporates prior knowledge imparted by a training set of pre-segmented, labeled models and combines that knowledge with content-driven analysis based on the geometric similarity between the matched shapes. Recently, Guo and Zou et al. [10] train convolutional neural networks to effectively classify faces in feature space to segment 3D shapes. Supervised segmentation methods usually outperform unsupervised ones. However, they often require a huge set of manually labeled segmentation results, which greatly limits their applicability.

Existing learning-based segmentation methods [8], [9], [10] need a fully labeled training shapes to train a learning model. Due to the limited generalizability of learning-based methods, the prediction models must be learned by using 3D training shapes similar to the testing 3D shapes, to achieve satisfactory segmentation results. For example, to segment a chair shape, one has to construct a prediction model by learning from training shapes of chairs. If we want to segment a variety of 3D shapes using learning-based methods, we have to manually prepare many different training 3D shapes, which usually require a different set of training data for different types of 3D shapes, where each face on the 3D shapes needs to be labeled. Therefore, expanding the capabilities of the system to different types of shapes becomes prohibitively expensive in terms of the manual labor required for producing the training data. Although a tool [13] for manually segmenting 3D shapes has been provided by the computer graphics research community, the labeling process is still quite tedious and time-consuming.

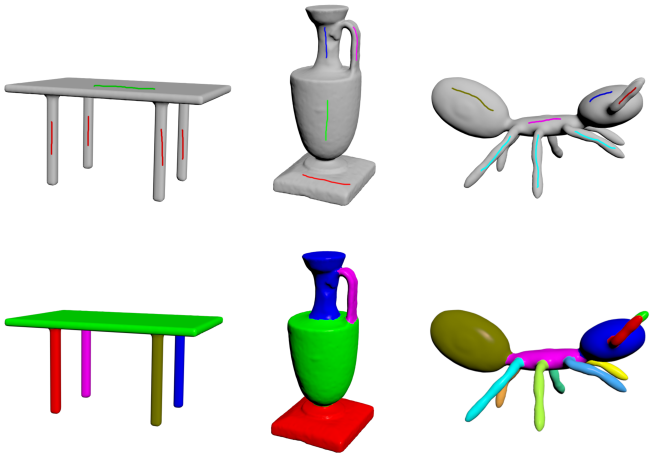


Fig. 3. Some examples of weakly labeled shapes with scribbles (top). The labeling information is automatically propagated to unlabeled parts on the 3D shapes by our algorithm (bottom). Note that we use different colors to show disconnected segments even if they have the same label.

To reduce the time of training data preparation, we propose a novel optimization model to remove the necessity of fully labeled training 3D shapes. With our new optimization model, only some sparsely labeled training 3D shapes are required for the learning process. The labeling information will be automatically propagated to the unlabeled parts of the training 3D shapes and given as a part of the optimization results in our algorithm. Figure 3 shows the weakly labeled training 3D shapes used as the input of our algorithm and the corresponding propagation results.

To our best knowledge, we are the first to introduce weakly-supervised deep learning techniques to 3D shape segmentation. It is worth pointing out that the distinguishing feature of our method is that it does not need fully labeled training shapes for the learning process, which can greatly reduce the expenses spent on training data preparation in existing learning-based segmentation method. For existing learning-based segmentation methods, a large number of similar fully labeled training shapes must be prepared. That means the users must fully label a lot of training shapes, which is quite expensive. Also, when segmenting different target 3D shapes, different groups of corresponding training 3D shapes are required for training the learning model, which is a significant drawback of the existing learning-based segmentation methods. Our method removes the necessity of the tedious and time-consuming training 3D shapes preparation process, and directly learns from scribble-based weakly labeled training 3D shapes. Therefore, our method is much more practical when compared with the existing learning-based methods, because it does not require fully labeled training 3D shapes.

3 Scribble-based Learning Model

3.1 Objective Function

We use a graphical model to propagate information from scribbles to unknown faces. For acceleration, we oversegment 3D shapes first, where each shape is divided into

primitive patches, such as the ones achieved in [21]. We then build a dual graph of the primitive patches. Each primitive patch is regarded as a vertex in the dual graph and an edge in the graph represents the similarity between two patches. Given a training 3D shape M, the scribble labels of M are $S = \{s_k, l_k\}$ where s_k are the faces of a scribble k and l_k is that scribble's category label. For a primitive patch x_i, we want to find its category label y_i.

In order to merge the initial small patches on the surface of a 3D shape into resulting segments, we have three considerations.

First, the segmentation results should respect the scribble labels provided by users and regard them as the ground truth. Therefore, we use the following unary energy term $E^i_{scr}(y_i)$ to ensure the optimization results comply with the training labels provided by scribbles.

$$E^i_{scr}(y_i) = \begin{cases} 0 & \text{if } y_i = l_k \text{ and } x_i \cap s_k \neq \emptyset \\ -\log\left(\frac{1}{|\{l_k\}|}\right) & \text{if } y_i \in \{l_k\} \text{ and } x_i \cap S = \emptyset \\ \infty & \text{otherwise} \end{cases} \tag{1}$$

The meaning of $E^i_{scr}(y_i)$ is the following: If a primitive patch x_i overlaps with a scribble s_k, then its corresponding penalty is zero when assigned l_k or infinity when the assigned label is not l_k. Otherwise, x_i can be assigned any labels with equal probability if it does not overlap with any scribble. However, it can never be assigned any labels that do not occur in this 3D shape.

Second, the segmentation results should conform to the predicted results of our trained model, i.e., a deep neural network. We use a unary term $E^i_{neu}(y_i, \Theta)$ to respect the output of the deep neural network and define it as:

$$E^i_{neu}(y_i, \Theta) = -\log P(y_i | X, \Theta). \tag{2}$$

Here Θ represents the parameters of the deep neural network. $\log P(y_i | X, \Theta)$ denotes the log probability of predicting x_i to have the label y_i. It can be computed by summing the log probabilities of all faces in the patch x_i.

Finally, faces with similar geometric features should be merged into one common segment. To measure the similarity between two primitive patches in the feature space, we define a pairwise term $E^{i,j}_{fea}(y_i, y_j)$ in our optimization problem as follows:

$$E^{i,j}_{fea}(y_i, y_j) = \begin{cases} \exp\left\{-\frac{\|f(x_i)-f(x_j)\|_2^2}{\delta^2}\right\} & \text{if } y_i \neq y_j \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $f(x_i)$ and $f(x_j)$ represent the corresponding feature vectors of primitive patches x_i and x_j, respectively. The purpose of this term is to ensure that two primitive patches should be very different in the feature space if they have different labels.

With the above considerations, we give the following objective function to minimize by our scribble-based learning model:

$$E = E_{scr}(Y) + E_{neu}(Y, \Theta) + \lambda E_{fea}(Y), \tag{4}$$

where Y denotes the set of $\{y_i\}$, $E_{scr}(Y) = \sum_i E^i_{scr}(y_i)$, $E_{neu}(Y, \Theta) = \sum_i E^i_{neu}(y_i, \Theta)$ and $E_{fea}(Y) = \sum_{i,j} E^{i,j}_{fea}(y_i, y_j)$.

There are two sets of variables to be optimized, which are primitive patches' category labels Y and the deep neural network's parameters Θ, such that the above objective function is minimized to get the trained deep learning models. In this paper, λ is empirically set to be 1.0. Equation (4) is only for one 3D shape. Our final energy function sums Equation (4) over all 3D shapes in the same category.

3.2 Optimization

The optimization problem proposed in the previous section can be effectively solved by alternating optimization (also known as block coordinate descent). We fix Θ to solve for Y and then fix Y to solve for Θ.

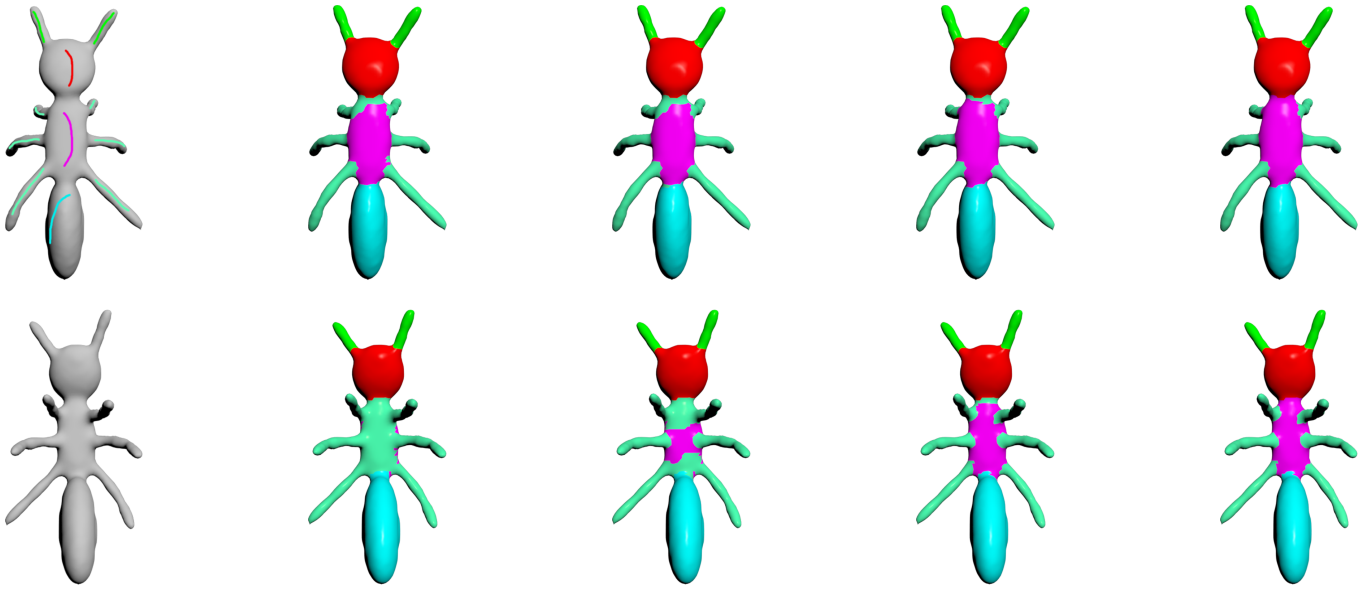
Propagating users' scribble information to unlabeled faces. When Θ is fixed, the solver propagates labels to unmarked faces based on users' scribble, geometric features, and deep neural network predictions. In this step, the objective function can be effectively minimized by using graph cuts algorithm [31] because the objective function in Equation (4) can be seen as a linear combination of the unary term $(\sum_i E^i_{scr}(y_i) + \sum_i E^i_{neu}(y_i, \Theta))$ and the pairwise term $(\sum_{i,j} E^{i,j}_{fea}(y_i, y_j))$. This formulation is widely used for 3D shape segmentation [6], [9], [10].

Optimizing deep neural network parameters. When Y is fixed, minimizing the objective function turns into training a deep neural network and the solver learns a deep neural network for face-wise 3D shape segmentation. Given the labels Y for all faces on 3D shapes, the deep neural network can be trained by regarding feature vectors and label information of all faces as input vectors and output probability vectors respectively. The last layer of the deep neural network outputs the face-wise log probabilities, which are used to update the unary term in our energy function.

Our solver is initialized from the graph cuts step without using network prediction and then alternates between the two steps. In our experiments, the neural network contains 4 fully connected layers and the numbers of neurons in the two hidden layers are experimentally tuned to best fit the training set. The first layer and the last layer contains d and c neurons respectively, where d is the dimension of input feature vectors and c is the number of label categories provided by scribbles. The rectified linear unit [32] is employed as the activation function for each layer except the last layer. The last layer of our neural network is a softmax classifier producing the probabilities of each faces belonging to different categories. Our solver stops iterations when the total energy does not decrease anymore.

Note that here the graph cuts step is applied on the patch level, while the deep neural network is directly trained on the face level. Theoretically, we can change our objective function so that our method can apply the graph cuts step and the training step on the face level. However, applying graph cuts on the patch level instead of the face level can further reduce the computation time. Therefore, we apply graph cuts algorithm on the patch level (usually 50 patches) instead of the face level in our method.

Figure 4 shows the optimization process of our algorithm for an ant model. In this figure, we can see that the intermediate segmentations improve with each successive iteration.



(a) Scribble (b) Iteration 1 (c) Iteration 2 (d) Iteration 3 (e) Final result

Fig. 4. The optimization process for an ant model. With the decreasing value of the objective function, the intermediate segmentation result of each iteration becomes better and better. The solver converges after 4 iterations. The top row and the bottom row show the ant model from the top view and bottom view respectively. Average geodesic distance, shape diameter function and Gaussian curvature are used as feature descriptors.

In each iteration, the labeling information is learned from the result of the graph cut by the deep neural network, and then used to more accurately predict the labeling information. As the deep neural network gets updated, the labeling information from the network becomes more reliable, improving the accuracy propagated labeling information from the graph cuts. This propagated labeling information in turn improve the deep neural network learning. In our experiments, we empirically find that the optimization process converges after four iterations and the satisfactory result is obtained.

3.3 Shape signatures selection

To extract feature vectors for faces and primitive patches, four widely known feature descriptors, including average geodesic distance (AGD) [33], shape diameter function (SDF) [3], Gaussian curvature (GC) [34] and scale-invariant heat kernel signatures (SIHKS) [35], are selected in this paper. These feature descriptors are deemed to have the capability of characterizing the geometric properties well from different perspectives. For each face, we concatenate the values (for AGD, SDF, and GC) and vectors (for SIHKS, the dimension is experimentally set to be 19) together and obtain a corresponding feature vector, whose dimension is 22, to be used in Equation (2). For each primitive patch, a different strategy is adopted for extracting feature vectors. For scalar fields defined on each face, such as AGD, SDF, and GC, we use histograms capturing the feature distribution to extract the feature vectors for each primitive patch. For SIHKS, which is defined as a vector field on each face, we extract feature vectors by employing the famous bag-of-feature (BoF) technique for each primitive patch. We note that the number of bins in the histograms and that of the bags for the bag-of-feature representation are both set to be $B = 100$. Like the face-level situation, the feature vectors from different shape signatures are also concatenated into a single feature vector to be used in Equation (1) and (3). In this way, any feature descriptor can be adapted into our algorithm framework.

3.4 Algorithm

Our algorithm works on a set of similar 3D shapes and it consists of five stages, as illustrated in Figure 2. First, similar to the super-pixel based image segmentation [36], [37], we divide each shape into primitive patches for speed-up, like in [21]. Second, we calculate feature vectors for each patch, as described in Section 3.3. Then in the next stage, we manually scribble the shapes and get the weakly-labeled training data for our segmentation method. After that, with the scribble-based labels and the extracted feature vectors for each face and primitive patch, a deep learning model is constructed and trained to predict the label of each primitive patch in the shapes. Finally, the learned model is applied to the shapes to obtain the segmentation results. We apply graph cuts algorithm [6], [31] to further improve the quality of the segmentation.

Our algorithm can be summarized as follows. Given a set of manually scribbled training 3D shapes,

1) For each training 3D shape, divide it into several primitive patches.
2) For each primitive patch, calculate corresponding feature vectors.
3) Minimize the objective function (4) using alternating optimization.
    a) Apply graph cuts technique to get an initial guess of Y.
    b) Fix Y and train deep neural network parameters Θ to minimize the objective function (4).

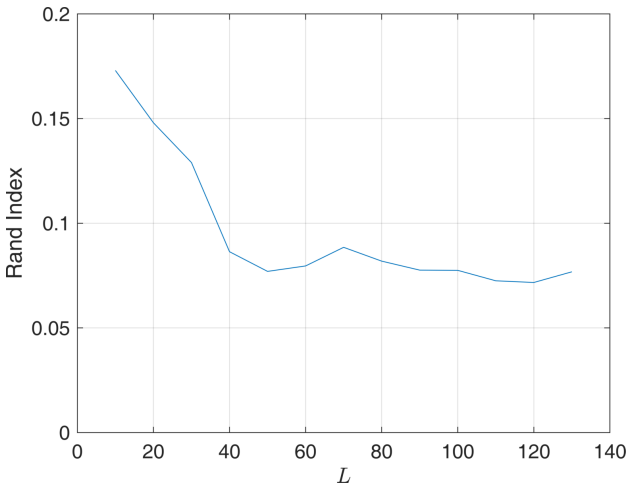


Fig. 5. The average Rand Index scores with regard to L on the PSB dataset, where lower values indicate better results.

c) Fix Θ and optimize Y to minimize the objective function (4).
d) Repeat step (3b) and step (3c) until convergence.

The results of our algorithm include not only segmented training 3D shapes but also a neural network which can be used to predict and segment test 3D shapes.

4 Evaluation

Experimental dataset. We test our segmentation algorithm on the Princeton Segmentation Benchmark (PSB) dataset [13] and ShapeNetCore dataset [12], [38], which are both open datasets for 3D shape segmentation. The PSB dataset contains 19 different object categories and each category contains 20 3D shapes. For PSB dataset, we use the ground-truth provided by [8] for evaluation. For ShapeNetCore dataset, we use the ground-truth provided by [38], which are converted from point-based segmentation results provided by [12]. It is worth pointing out that most of the 3D shapes in the ShapeNetCore dataset are non-manifold while our algorithm needs a manifold shape as the input. Therefore, we resample and reconstruct all the shapes in the dataset so that all 3D shapes become manifold. We developed a simple software tool to facilitate the labeling process and employed 3 volunteers to scribble the training 3D shapes. Some examples of scribble results are shown in Appendix.

Evaluation metrics. We use four metrics, Rand Index, Cut Discrepancy, Hamming Distance and Consistency Error, which are defined by [13] to evaluate the segmentation results. Rand Index is used to compute the similarity between two segmentations. Cut Discrepancy measures the similarity between different cuts by comparing the boundaries. As opposed to Cut Discrepancy, Hamming Distance compares the regions and evaluates the minimum number of substitutions required when changing one region into another. Consistency Errors, including the global one (GCE) and the local one (LCE), measure the hierarchical differences between two segmentations of the same shape.

Parameter settings. In this paper, the weight coefficient λ in the optimization problem (4) is set to 1.0. In our experiments, we tried various choices of L. We show the average Rand Index scores with different values of L on PSB in Figure 5. It can be seen that the Rand Index score gets worse when L is far less than 50 because small parts cannot be captured very well by our method in this case. However, if L is far larger than 50, too many patches will lead our method to be too inefficient. Based on the above observation, the number of patches for over-segmenting is experimentally set to $L = 50$.

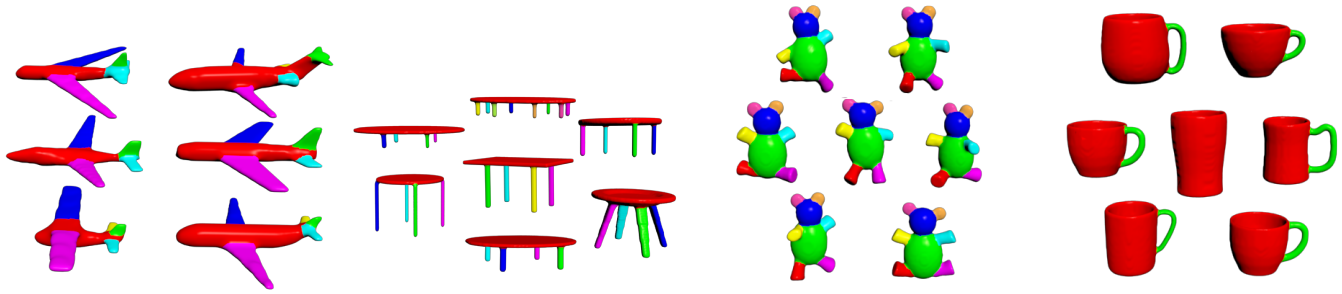
4.1 Results

Our method is a weakly-supervised one and the training dataset only requires partially annotated 3D shapes. It is worth pointing out that, although the 3D shapes in the training set only contain partially annotated information (obtained from scribbles) instead of fully annotated ones, all the 3D shapes in the training set are automatically fully annotated by our learning process as a by-product of our algorithm. Therefore, our algorithm can produce two sets of segmentation results. One set of results are the segmentation results on the testing set, obtained by applying our learned model to the testing set. The other set of results are the segmentation results on the training set, obtained from the learning process of our method.

Figure 6 shows the segmentation results on the training shapes by using our method for some representative categories of 3D shapes in PSB. Figure 7 shows the segmentation results on the testing shapes by using our method on the same dataset. Although there are large shape variations, a large majority of the segmentation results are desirable and consistent with our perception. The Rand Index score statistics of our segmentation for training shapes on the PSB dataset, as well as those of other methods, are detailed in Table 1. These were obtained by first computing the average Rand Index score for each category respectively, and then averaging over all categories in the corresponding dataset. From Table 1 we can see that our algorithm obtains an average Rand Index of 0.076, which outperforms related algorithms.

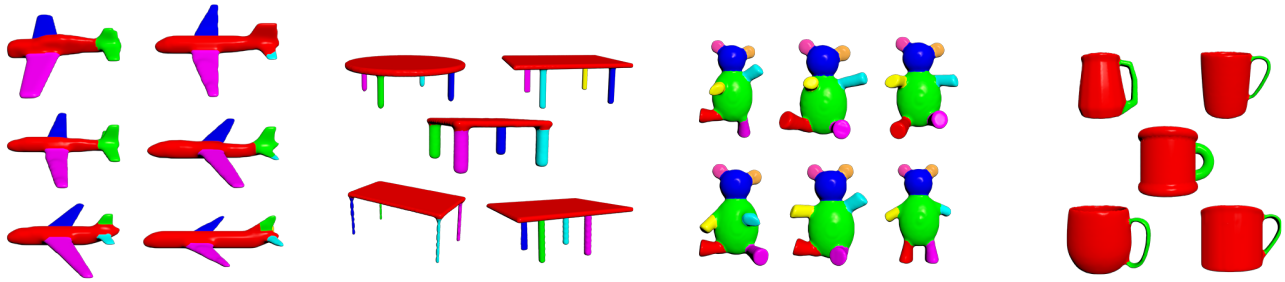
4.2 Sensitivities to scribble quality

The performance of our algorithm relies on the scribbles provided by users. To test the sensitivity of our algorithm to scribble quality, we compute the Rand Index scores of our algorithm for each 3D shape in PSB, while varying the percentage of the surface covered by the scribbles.. For a 3D shape, let S be the set of faces covered by the scribbles provided by our volunteers. We gradually remove faces from S to reduce the coverage of scribbles. For each step, we randomly remove faces from S and get the corresponding Rand Index scores from our algorithm. Figure 8 shows the corresponding Rand Index scores after averaging over the whole PSB. We can see that although the average Rand Index score decreases rapidly when more scribble information provided by users, the score is already very small even when the scribbles provided by the user only cover 1.0% area of the whole surfaces of 3D shapes.



(a) AirPlane (b) Table (c) Teddy (d) Cup

Fig. 6. Representative segmentation results produced by our approach (on training shapes) on the Princeton segmentation benchmark dataset.



(a) AirPlane (b) Table (c) Teddy (d) Cup

Fig. 7. Representative segmentation results produced by our approach (on testing shapes) on the Princeton segmentation benchmark dataset.

TABLE 1
The Rand Index scores of segmentation for each category of 3D shapes in PSB dataset with different methods.

Object Categories	Ours	PMC	WcSeg	RandCuts	NormCuts	RandWalks	Kmeans	SDF	FitPrim	CoreExtra
Human	0.064	**0.059**	0.084	0.117	0.126	0.162	0.136	0.133	0.139	0.199
Cup	**0.021**	0.089	0.125	0.268	0.432	0.443	0.559	0.384	0.488	0.309
Glasses	**0.048**	0.124	0.167	0.100	0.137	0.246	0.195	0.197	0.223	0.260
Airplane	0.081	0.108	**0.070**	0.129	0.200	0.249	0.236	0.083	0.179	0.272
Ant	**0.007**	0.057	0.009	0.041	0.083	0.076	0.109	0.009	0.100	0.056
Chair	**0.029**	0.116	0.066	0.172	0.081	0.143	0.193	0.075	0.198	0.166
Octopus	**0.022**	0.066	0.026	0.093	0.101	0.103	0.142	0.041	0.132	0.048
Table	0.014	0.025	0.056	0.413	0.260	0.274	0.465	0.174	0.309	0.210
Teddy	0.068	0.294	**0.040**	0.050	0.122	0.113	0.177	0.043	0.130	0.103
Hand	**0.078**	0.220	0.103	0.132	0.180	0.194	0.218	0.199	0.234	0.166
Plier	**0.050**	0.110	0.078	0.119	0.175	0.206	0.179	0.368	0.170	0.067
Fish	**0.100**	0.471	0.144	0.307	0.450	0.429	0.476	0.263	0.477	0.303
Bird	**0.092**	0.212	0.094	0.105	0.183	0.220	0.182	0.104	0.191	0.111
Armadillo	**0.065**	0.195	0.076	0.093	0.126	0.117	0.108	0.076	0.088	0.165
Bust	0.222	**0.189**	0.256	0.227	0.327	0.330	0.352	0.308	0.315	0.281
Mech	0.128	**0.064**	0.132	0.348	0.367	0.388	0.505	0.242	0.417	0.335
Bearing	0.106	0.159	**0.087**	0.160	0.250	0.303	0.322	0.105	0.202	0.380
Vase	**0.101**	0.351	0.109	0.115	0.285	0.301	0.389	0.212	0.271	0.182
Fourleg	0.148	0.175	0.139	0.182	0.201	0.232	0.197	**0.127**	0.189	0.164
Average	**0.076**	0.162	0.098	0.167	0.215	0.238	0.270	0.165	0.234	0.199

In order to obtain a desirable segmentation result, there are two key aspects. On the one hand, users' input scribbles should be representative, i.e., a scribble should be able to reflect the key geometric features or the overall geometric shape of the target area dominated by the scribble. On the other hand, the segmentation algorithm should be sufficiently intelligent to enable different scribbles to snatch their own regions while maintaining a regional balance. Generally speaking, the two aspects are closely related and inseparable. If the geometric shapes of different patches are much different from each other, then it decouples the requirements of input scribbles. For example, for the desk model in Figure 9, whether long scribbles or short scribbles, the segmentation results are identical. However, when the geometric features of different patches are quite similar, scribbles that are more representative have to be required. If the input scribbles are very short and not representative (see the top row of Figure 10), our algorithm may report a low-quality segmentation result. But generally speaking, our algorithm is able to yield a high-quality segmentation result as long as the input scribbles are not intentionally bad. See the bottom row of Figure 10.

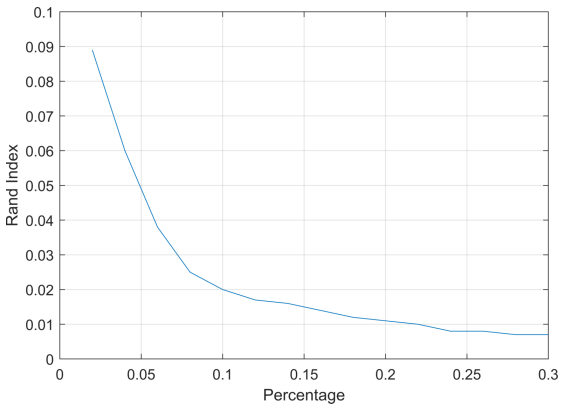


Fig. 8. The average Rand Index scores of our segmentation results on training shapes when using different percent of coverage with scribbles.

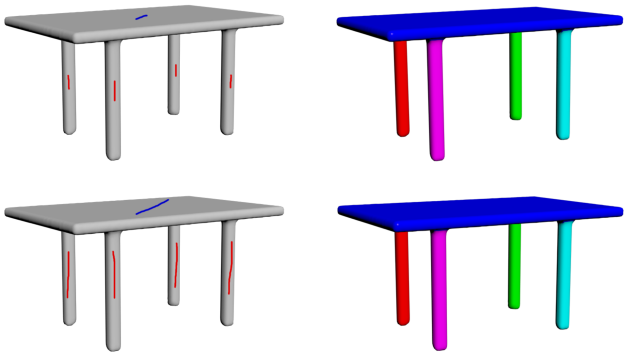


Fig. 9. Segmentation results produced by using short and long scribbles for a Table model. The top line shows short scribbles and the corresponding segmentation results. The bottom line shows long scribbles and the corresponding segmentation results.

4.3 Comparison

We compare our method with 12 other segmentation algorithms including Paint mesh cutting (PMC, [39]), approximate convexity analysis (WcSeg, [16]), Randomized cuts [21], Normalized cuts [21], Random walks [23], K-Means [19], SDF [3], Fitting primitive [40], Core extraction [20], FSDL [10], the method of Kalogerakis et al. (SB19, [8]) and Extreme Learning Machine (ELM, [9]) on the PSB database. All the related segmentation algorithms are compared against human-generated segmentations (provided by [8]). Table 1 shows the comparison of results from our method on training shapes and other algorithms using Rand Index. Additionally, we also use the other four metrics for evaluation and the detailed statistics plots are shown in Figure 11. It can be seen that our segmentation method on training shapes outperforms other segmentation approaches no matter what kind of evaluation metric is used, except the WcSeg method. When comparing with the WcSeg method, our algorithm is also better for all metrics except Consistency Error. No matter what kind of metric is used, lower values mean closer similarity to the human-generated ground truth. Figure 12 provides a qualitative comparison on the Teddy model, the Cup model, and the Armadillo model.

Comparison with another scribble-based method. Table 1 shows the comparison between our method and Paint mesh cutting (PMC, [39]), which is also a 3D shape segmentation algorithm based on users' scribbles, with Rand Index score. From the results, we can see that our method outperforms PMC for 16 out of 19 categories. Furthermore, the average Rand Index score of our method for all categories is lower than PMCs'. Additionally, Figure 11 shows the superior performance of our method compared to PMC.

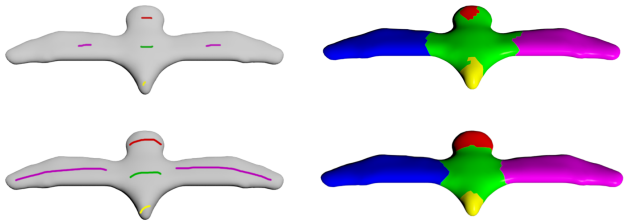
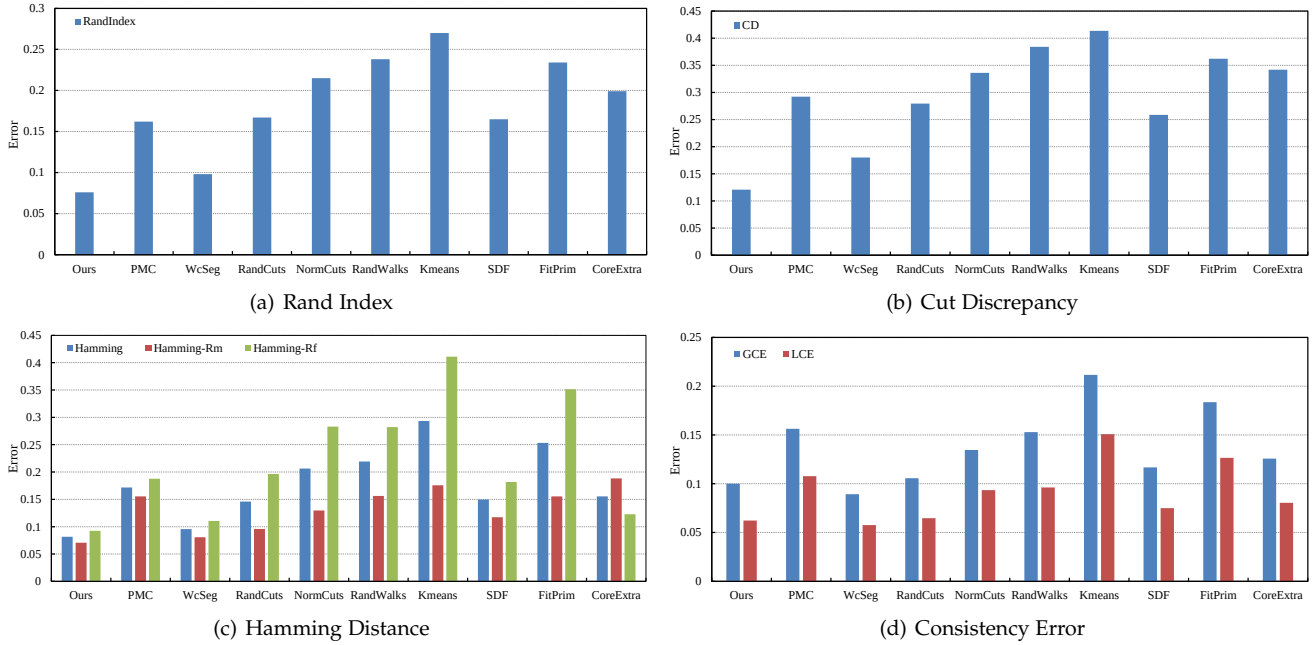


Fig. 10. Segmentation results produced by using short and long scribbles on a Bird model. The top line shows short scribbles and the corresponding segmentation results. The bottom line shows long scribbles and the corresponding segmentation results.

Comparison with fully supervised deep learning method. We compare our method with three other 3D shape segmentation methods (FSDL, [10], SB19, [8], ELM, [9]), which employs fully supervised deep learning techniques to segment 3D shapes. Table 2 shows the Rand Index scores for each category of 3D shapes in PSB using these four methods. Note that the results from our method on testing shapes and other methods are obtained by regarding 19 3D shapes in each category (there are 20 3D shapes in each category in total) as training set and the one left out 3D shape is used as testing set. From the results, we can see that Rand Index scores of our method on training shapes for 7 categories are better than FSDLs' and worse for the other 12 categories. The Rand Index scores of our method on testing shapes for 6 categories is better than FSDLs' and worse for the other categories. The average performance of our method is worse than FSDL. Similarly, SB19 also achieve slightly better performance than our method. We believe the main reason is that our method is weakly-supervised and utilizes much fewer human labeled training samples than FSDL. Figure 13 shows a qualitative comparison between our method and FSDL on the Fish model, the Bird model, and the Armadillo model.

Additionally, we compare our method with the state-of-the-art method PFCN proposed in [38] on ShapeNetCore dataset ([12]), which is also fully supervised. The method combines image-based fully convolutional networks and surface-based conditional random fields to obtain coherent segmentations of 3D shapes. The ShapeNetCore dataset contains 4916 3D shapes totally, where the most 3D shapes in the dataset are non-manifold. Because our method can only be applied to manifold 3D shapes, we remesh all the non-manifold shapes in ShapeNetCore and convert them to manifold version. The training and testing setting used in the two methods follows the one provided by PFCN. Table 3 shows the Rand Index scores of two methods for 14 categories of 3D shapes in the manifold ShapeNetCore dataset. We can see that Rand Index scores of our method on training shapes for 5 categories are better than PFCNs' and worse for the other 9 categories. The Rand Index scores of

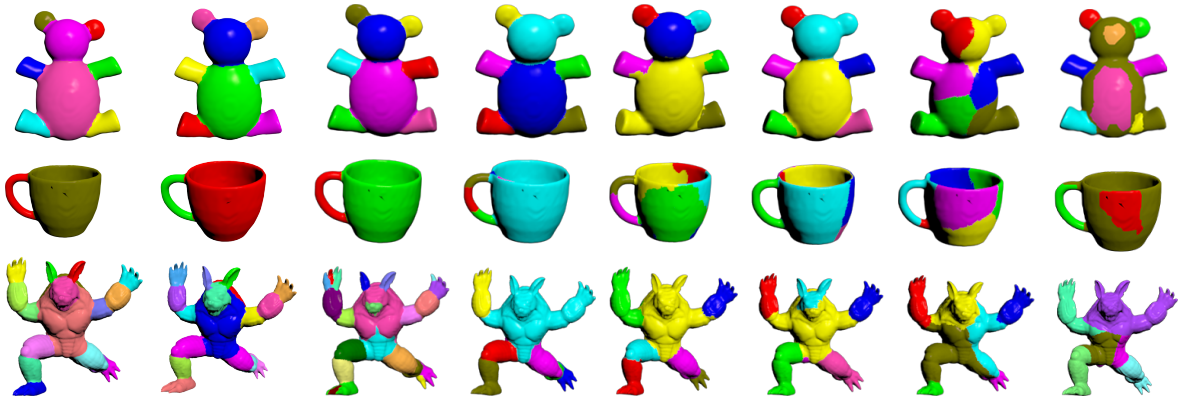
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2019.2892076, IEEE Transactions on Visualization and Computer Graphics



(a) Rand Index (b) Cut Discrepancy

(c) Hamming Distance (d) Consistency Error

Fig. 11. Performance plots of different segmentation algorithms with respect to four evaluation metrics. Lower values indicate closer similarity to the human-generated ground truth.



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 12. Comparison with other six segmentation algorithms. The approaches used here include (a)Ground Truth; (b) Ours (training shapes); (c) WcSeg; (d) Randomized cuts; (e) Normalized cuts; (f) Random walks; (g) K-Means; (h) PMC.

TABLE 2
The comparison of Rand Index scores from our method and fully supervised deep learning method (FSDL, [10]), the method of Kalogerakis et al. (SB19, [8]), and Extreme Learning Machine (ELM, [9]) on the PSB dataset. The results from FSDL, SB19, and ELM are got by regarding 19 3D shapes in each category as the training set and the left 1 3D shape as the testing set.

Algorithms	Human	Cup	Glasses	Airplane	Ant	Chair	Octopus	Table	Teddy	Hand
Our method (training shapes)	0.064	0.021	0.048	0.081	0.007	0.029	0.022	0.014	0.068	0.078
Our method (testing shapes)	0.072	0.025	0.051	0.081	0.013	0.026	0.030	0.021	0.079	0.082
FSDL	0.082	0.012	**0.006**	0.036	0.030	0.033	0.033	0.017	0.101	0.051
SB19	**0.038**	**0.008**	0.095	**0.031**	**0.005**	**0.009**	**0.011**	**0.010**	**0.012**	**0.039**
ELM	0.098	0.103	0.088	0.089	0.042	0.071	0.018	0.059	0.036	0.114
Algorithms	Plier	Fish	Bird	Armadillo	Bust	Mech	Bearing	Vase	Fourleg	Average
Our method (training shapes)	0.050	0.100	0.092	0.065	0.222	0.128	0.106	0.101	0.148	0.076
Our method (testing shapes)	0.060	0.100	0.094	0.065	0.234	0.134	0.111	0.106	0.153	0.081
FSDL	0.051	**0.004**	**0.007**	0.196	**0.026**	0.045	**0.028**	**0.005**	**0.011**	0.041
SB19	**0.019**	0.049	0.069	**0.027**	0.151	**0.039**	0.056	0.087	0.069	**0.031**
ELM	0.054	0.132	0.166	0.094	0.221	0.159	0.154	0.156	0.091	0.102

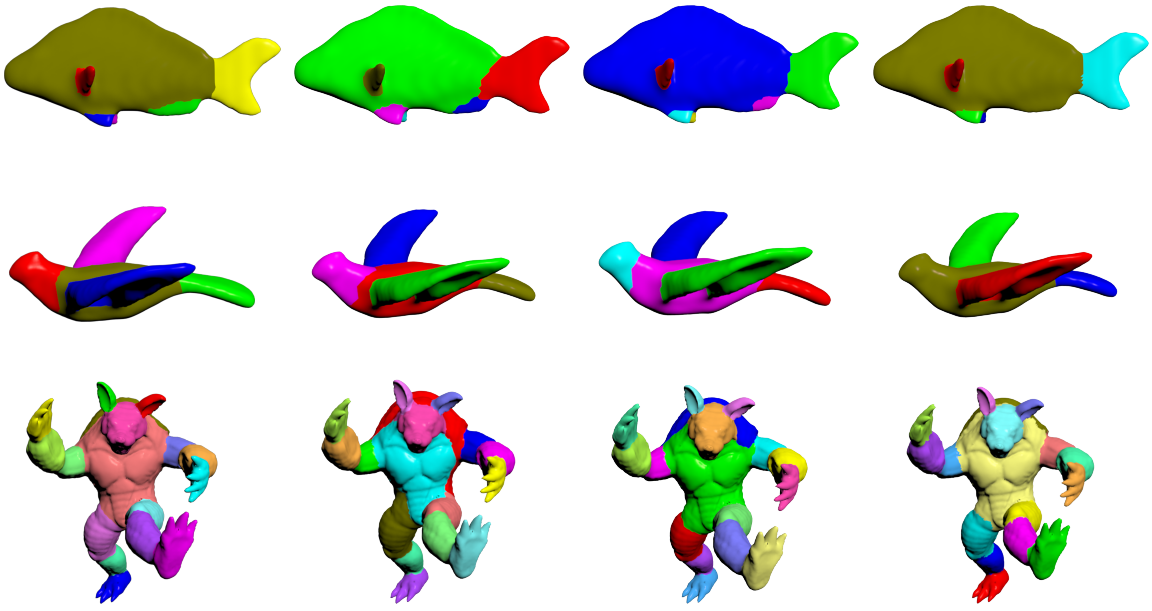


Fig. 13. The Comparison of our method on training shapes, testing shapes, and fully supervised deep learning method (FSDL, [10]). (a) Ground Truth; (b) The results of our method on training shapes; (c) The results of our method on testing shapes; (d) The results of FSDL.

TABLE 3
The comparison of Rand Index scores from our method and Projective Fully Convolutional Networks (PFCN, [38]) on the ShapeNet dataset. The results from PFCN are provided by the authors.

Algorithms	Skateboard	Rocket	Pistol	Mug	Motorbike	Laptop	Lamp	
Our method (training shapes)	0.038	0.161	0.126	**0.020**	0.193	0.104	0.094	
Our method (testing shapes)	0.046	0.180	0.137	0.032	0.211	0.116	0.098	
PFCN	**0.033**	**0.123**	**0.055**	0.062	**0.019**	**0.042**	**0.079**	
Algorithms	Knife	Guitar	Earphone	Chair	Bag	Airplane	Table	Average
Our method (training shapes)	**0.087**	0.066	**0.025**	0.230	**0.024**	0.086	**0.050**	0.093
Our method (testing shapes)	0.128	0.080	0.044	0.295	0.053	0.103	0.062	0.113
PFCN	0.149	**0.029**	0.046	**0.070**	0.098	**0.023**	0.105	**0.067**

TABLE 4
The average computation time of different algorithms ('+': measured on a PC with 2.66GHz CPU and NVIDIA GeForce 1080 Ti GPU, '△': measured on a PC with Xeon E5-2670 2.60GHz CPU, '#': measured on a PC with 2GHz CPU).

Algorithms	Ours	PMC	FSDL	WcSeg	RandCuts	NormCuts	RandWalks	Kmeans	SDF	FitPrim	CoreExtra
Time(s)	132.3^{+}	8.6^{+}	$14400.0^{\triangle}$	93.7^{+}	$83.8^{\#}$	$49.4^{\#}$	$1.4^{\#}$	$2.5^{\#}$	$8.9^{\#}$	$4.6^{\#}$	$19.5^{\#}$

our method on testing shapes for 5 categories is better than PFCNs' and worse for the other 9 categories. The average performance of our method is worse than PFCN.

Our method is weakly supervised and its performance is not as good as fully supervised methods. However, generating the training data (scribbles) for our algorithm is much simpler than generating training data for fully supervised methods.

4.4 Performance

We implemented the proposed algorithm in Matlab and C++. The computation time of the algorithm is shown in Table 4. On average, our algorithm takes more than 2 minutes to process a shape, where the time cost for computing features is about 11 seconds and the rest time is spent solving the optimization problem. However, our current unoptimized implementation could be further sped up by parallelization techniques.

5 Limitation and Future Work

First, designing a desirable structure of the deep neural network is non-trivial. We will conduct more experiments to obtain a general configuration such that the algorithm framework can work well across various model libraries and various basic shape signature combinations.

Second, our algorithm relies on the assumption that the shape category is known. In order to make the algorithm fully automatic, we need to add a separate classification network to predict which category the test model belongs to in the future.

Third, the input shapes of our method must be manifold. Extending our method to handle non-manifold shapes is our future work.

Finally, the required computation time, especially the time spent in training the deep neural network, is still too long. In the future, we will seek a parallelized implementation, which will greatly reduce the time costs and facilitate its use in practice.

6 Conclusion

In this paper, we propose a novel weakly-supervised method for segmenting 3D shapes. Different from previous learning-based approaches, completely labeled training shapes are not necessary for our method. With a scribble-based labeling process, which is much cheaper compared to the traditional one of fully labeling the training data, our method can achieve a comparable performance. We validate our method on an open dataset Princeton Shape Benchmark, and make extensive comparisons with the state-of-the-art approaches on this problem. The experimental results demonstrate that our method can achieve better performance than the existing unsupervised approaches and comparable performance than the fully supervised method.

Acknowledgments

Many thanks to the anonymous reviewers for their valuable comments and suggestions. We would also like to thank Dimitar Dinev, Oliver van Kaick, Zizhao Wu, Yunhai Wang and Ruizhen Hu for their kind help. This work is supported by National Natural Science Foundation of China (61872321, 61672482, 11626253, 61732015, 61772016, 61572022), National Science Foundation (IIS-1617172, IIS-1622360), Natural Science Foundation of Zhejiang Province (LY17F020018), Natural Science Foundation of Ningbo City (2018A610161), Ningbo Leader and Top-notch Talent Training Project (NBLJ201801010), Ningbo Innovative Team: The intelligent big data engineering application for life and health (2016C11024), the Fundamental Research Funds of Shandong University, and the Open Project Program of the State Key Lab of CAD & CG (Grant No. A1702), Zhejiang University.

References

[1] Z. Wu, Y. Wang, R. Shou, B. Chen, and X. Liu, "Unsupervised co-segmentation of 3D shapes via affinity aggregation spectral clustering," *Computers & Graphics*, vol. 37, no. 6, pp. 628–637, 2013.

[2] O. Sorkine, D. Cohen-Or, R. Goldenthal, and D. Lischinski, "Bounded-distortion piecewise mesh parameterization," in *Proceedings of the Conference on Visualization '02*. IEEE Computer Society, 2002, pp. 355–362.

[3] L. Shapira, A. Shamir, and D. Cohen-Or, "Consistent mesh partitioning and skeletonisation using the shape diameter function," *The Visual Computer*, vol. 24, no. 4, pp. 249–259, 2008.

[4] P. Simari, D. Nowrouzezahrai, E. Kalogerakis, and K. Singh, "Multi-objective shape segmentation and labeling," *Computer Graphics Forum*, vol. 28, no. 5, pp. 1415–1425, 2009.

[5] S. Shalom, L. Shapira, A. Shamir, and D. Cohen-Or, "Part analogies in sets of objects," in *Proceedings of the 1st Eurographics Conference on 3D Object Retrieval*, 2008, pp. 33–40.

[6] M. Meng, J. Xia, J. Luo, and Y. He, "Unsupervised co-segmentation for 3D shapes using iterative multi-label optimization," *Computer-Aided Design*, vol. 45, no. 2, pp. 312–320, 2013.

[7] O. Sidi, O. van Kaick, Y. Kleiman, H. Zhang, and D. Cohen-Or, "Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering," *ACM Transactions on Graphics*, vol. 30, no. 6, pp. 1–10, 2011.

[8] E. Kalogerakis, A. Hertzmann, and K. Singh, "Learning 3D mesh segmentation and labeling," *ACM Transactions on Graphics*, vol. 29, no. 4, pp. 1–12, 2010.

[9] Z. Xie, K. Xu, L. Liu, and Y. Xiong, "3D shape segmentation and labeling via extreme learning machine," *Computer Graphics Forum*, vol. 33, no. 5, pp. 85–95, 2014.

[10] K. Guo, D. Zou, and X. Chen, "3D mesh labeling via deep convolutional neural networks," *ACM Transactions on Graphics*, vol. 35, no. 1, pp. 1–12, 2015.

[11] Z. Shu, C. Qi, S. Xin, C. Hu, L. Wang, Y. Zhang, and L. Liu, "Unsupervised 3d shape segmentation and co-segmentation via deep learning," *Computer Aided Geometric Design*, vol. 43, pp. 39–52, 2016.

[12] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3D shape collections," *ACM Transactions on Graphics*, vol. 35, no. 6, pp. 1–12, 2016.

[13] X. Chen, A. Golovinskiy, and T. Funkhouser, "A benchmark for 3D mesh segmentation," *ACM Transactions on Graphics*, vol. 28, no. 3, pp. 1–12, 2009.

[14] M. Attene, S. Katz, M. Mortara, G. Patane, M. Spagnuolo, and A. Tal, "Mesh segmentation - a comparative study," in *IEEE International Conference on Shape Modeling and Applications, 2006*, Matsushima, 2006, p. 7.

[15] A. Shamir, "A survey on mesh segmentation techniques," *Computer Graphics Forum*, vol. 27, no. 6, pp. 1539–1556, 2008.

[16] O. V. Kaick, N. Fish, Y. Kleiman, S. Asafi, and D. Cohen-OR, "Shape segmentation by approximate convexity analysis," *ACM Transactions on Graphics*, vol. 34, no. 1, pp. 1–11, 2014.

[17] O.-C. Au, Y. Zheng, M. Chen, P. Xu, and C.-L. Tai, "Mesh segmentation with concavity-aware fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 7, pp. 1125–1134, 2012.

[18] L. Rong and Z. Hao, "Segmentation of 3D meshes through spectral clustering," in *Proceedings of the 12th Pacific Conference on Computer Graphics and Applications*, 2004, pp. 298–305.

[19] S. Shlafman, A. Tal, and S. Katz, "Metamorphosis of polyhedral surfaces using decomposition," *Computer Graphics Forum*, vol. 21, no. 3, pp. 219–228, 2002.

[20] S. Katz, G. Leifman, and A. Tal, "Mesh segmentation using feature point and core extraction," *The Visual Computer*, vol. 21, no. 8, pp. 649–658, 2005.

[21] A. Golovinskiy and T. Funkhouser, "Randomized cuts for 3D mesh analysis," *ACM Transactions on Graphics*, vol. 27, no. 5, pp. 1–12, 2008.

[22] S. Katz and A. Tal, "Hierarchical mesh decomposition using fuzzy clustering and cuts," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 954–961, 2003.

[23] Y. K. Lai, S. M. Hu, R. R. Martin, and P. L. Rosin, "Fast mesh segmentation using random walks," in *Proceedings of ACM Symposium on Solid and Physical Modeling*, 2008, pp. 183–191.

[24] A. Golovinskiy and T. Funkhouser, "Consistent segmentation of 3D models," *Computers & Graphics*, vol. 33, no. 3, pp. 262–269, 2009.

[25] Q. Huang, V. Koltun, and L. Guibas, "Joint shape segmentation with linear programming," *ACM Transactions on Graphics*, vol. 30, no. 6, pp. 1–12, 2011.

[26] R. Hu, L. Fan, and L. Liu, "Co-segmentation of 3D shapes via subspace clustering," *Computer Graphics Forum*, vol. 31, no. 5, pp. 1703–1713, 2012.

[27] K. Xu, H. Li, H. Zhang, D. Cohen-Or, Y. Xiong, and Z.-Q. Cheng, "Style-content separation by anisotropic part scales," *ACM Transactions on Graphics*, vol. 29, no. 6, pp. 1–10, 2010.

[28] Y. Zheng, D. Cohen-Or, M. Averkiou, and N. J. Mitra, "Recurring part arrangements in shape collections," *Computer Graphics Forum*, vol. 33, no. 2, pp. 115–124, 2014.

[29] Y. Wang, S. Asafi, O. van Kaick, H. Zhang, D. Cohen-Or, and B. Chen, "Active co-analysis of a set of shapes," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 1–10, 2012.

[30] O. van Kaick, A. Tagliasacchi, O. Sidi, H. Zhang, D. Cohen-Or, L. Wolf, and G. Hamarneh, "Prior knowledge for part correspondence," *Computer Graphics Forum*, vol. 30, no. 2, pp. 553–562, 2011.

[31] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.
[32] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning*, 2010, pp. 807–814.
[33] L. Shapira, S. Shalom, A. Shamir, D. Cohen-Or, and H. Zhang, "Contextual part analogies in 3D objects," *International Journal of Computer Vision*, vol. 89, no. 2, pp. 309–326, 2010.
[34] R. Gal and D. Cohen-Or, "Salient geometric features for partial shape matching and similarity," *ACM Transactions on Graphics*, vol. 25, no. 1, pp. 130–150, 2006.
[35] M. M. Bronstein and I. Kokkinos, "Scale-invariant heat kernel signatures for non-rigid shape recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1704–1711.
[36] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proceedings of the 9th IEEE International Conference on Computer Vision*, 2003, pp. 10–17.
[37] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.
[38] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3D shape segmentation with projective convolutional networks," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2017.
[39] L. Fan, L. Liu, and K. Liu, "Paint mesh cutting," *Computer Graphics Forum*, vol. 30, no. 2, pp. 603–612, 2011.
[40] M. Attene, B. Falcidieno, and M. Spagnuolo, "Hierarchical mesh segmentation based on fitting primitives," *The Visual Computer*, vol. 22, no. 3, pp. 181–193, 2006.

Zhenyu Shu got his Ph.D. degree in 2010 at the Zhejiang University, China. He is now working as an associate professor at Ningbo Institute of Technology, Zhejiang University. His research interests include computer graphics, digital geometry processing and machine learning. He has published over 30 papers in international conferences or journals.

Xiaoyong Shen got his Ph.D. degree in Computer Science and Engineering Department in the Chinese University of Hong Kong in 2016. He is now a Postdoctoral Fellow at Computer Science and Engineering Department in the Chinese University of Hong Kong. His research interests include computer graphics and computer vision.

Shiqing Xin is an associate professor at the Faculty of School of Computer Science and Technology in Shandong University. He received his Ph.D. degree in applied mathematics at Zhejiang University in 2009. His research interests include computer graphics, computational geometry and 3D printing.

Qingjun Chang is currently an undergraduate in School of Computer and Data Engineering at Ningbo Institute of Technology, Zhejiang University. His research interests include digital geometry processing and machine learning.

Jieqing Feng received his bachelor and Ph.D. both in Applied Mathematics from the National University of Defense Technology and Zhejiang University in 1992 and 1997, respectively. He is currently a full professor of computer science at the State Key Laboratory of Computer Aided Design and Computer Graphics in Zhejiang University. His research interests include space deformation and its applications, digital geometry processing, human body modeling via stereo vision, real-time rendering and fast simulation in solar thermal power generation.

Ladislav Kavan received his Mgr. degree from Charles University in Prague in 2003 and his Ph.D. degree in 2007 from Czech Technical University in Prague. He is currently an assistant professor at the University of Utah, USA. His research interests include computer animation, physics-based simulation and geometry processing.

Ligang Liu received the BSc degree in 1996 and the Ph.D. degree in 2001 from Zhejiang University, China. He is a professor at the University of Science and Technology of China. Between 2001 and 2004, he was at Microsoft Research Asia. Then he was at Zhejiang University during 2004 and 2012. He paid an academic visit to Harvard University during 2009 and 2011. His research interests include geometric processing and image processing. He serves as the associated editors for journals of IEEE Transactions on Visualization and Computer Graphics, IEEE Computer Graphics and Applications, Computer Graphics Forum, Computer Aided Geometric Design, and The Visual Computer. His research works could be found at his research website: http://staff.ustc.edu.cn/lgliu