ASME 2016 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC2016
August 21-24, 2016, Charlotte, North Carolina

DRAFT: DETC-60187

3D MESH SEGMENTATION WITH SVM AND BOUNDARY CORRECTION

Zhiqing Zhao
Department of Mechanical and Aerospace Engineering
The State University of New York at Buffalo
Buffalo, NY, USA

Ehsan Tarkesh Esfahani
Department of Mechanical and Aerospace Engineering
The State University of New York at Buffalo
Buffalo, NY, USA

ABSTRACT

3D shape mesh segmentation and labeling is a fundamental problem in shape understanding and analysis. This problem has wide applications in CAD model, additive manufacturing, robotics system, medical imaging etc. We modify SVM, a commonly used supervised learning algorithm, as a multi-class classifier using vote method to reach the goal of simultaneous segmentation and labeling of 3D meshes. Using both unary and pairwise geometric features extracted from the point cloud and triangulated mesh together with the labels that are already manually assigned by human users in PSB dataset as training data to classify the meshes. To improve the performance of boundary edges in the meshes, we conduct a 2nd level of classifier which is a binary classifier. In this process, we just learn the boundary edges with the pairwise features giving information at connections between faces that estimate the pairwise relationship of faces by strengthening the local properties. In addition, for the homogeneous check before re-classification that removes the noisy points mis-classified in the extracted boundaries, we apply K-means which is an unsupervised clustering method to group the boundaries and together with their group centers, then find out the noisy points that have larger distance to its cluster center compared to the average distance. The experimental results show that this method effectively works and has the comparable and even more stable performance to state-of-the-art.

1. INTRODUCTION

3D mesh segmentation is a popular and fundamental shape understanding problem in computer and engineering graphics attracting many researchers working on [1]. It gives the basis for many applications such like object matching and recognition, additive manufacturing, robotics grasping system, medical imaging, CAD application and so on. It researches on how to use quantitative analysis to decompose objects into semantic parts and sometimes we also desire automatically labeling for the parts. It still remains an open problem of labeled-segmentation that the parts of shapes are often manually specified in most researches. There're two aspects on this problem regarding on machine learning. In many applications, we not only need semantic segmentation of a 3D shape which can be done using unsupervised learning method, but sometimes we also require the labels for the decomposed parts. Thus labeled examples are needed to benefit the segmentation rather than solely uses geometric features. This kind of method is called supervised learning. There are many methods existing in this field and stimulate my work on this research.

Kalogerakis et al. [2] presented a data-driven approach to simultaneous segmentation and labeling of parts in 3D meshes. An objective function formulated as Conditional Random Field model is learned using a collection of labeled training meshes. Also, the unary terms assessing the consistency of faces with labels and terms between labels of neighboring faces are both applied to achieve the main goal of segmentation and the smooth on boundaries between different classes. Most of the researches usually need prior knowledge learned from training dataset. The size of training set and the complication of shapes largely affect the performance. *Jiajun Lv et al*. [3] improved this method and proposed a semi-supervised segmentation and labeling algorithm incorporated knowledge of both segmented, labeled meshed, and unsegmented, unlabeled meshes. A Conditional Random Field based objective function is also used to measure both the consistency of labels and faces and labels of neighboring faces. In addition, this paper adds a conditional entropy to the objective function to implant information from the unlabeled meshed. When using supervised learning algorithm, a tedious pre-training process is needed and greatly affected by the size of training set. Moreover, the labels of training set should be pre-assigned manually which is a time-costing process. To address these problems, *Xiuping Liu et al.* [4] developed a low-rank representation model with structure guiding to address the time-consuming and large training set problems. Manually labeling the training meshes is a tedious and subjective process that may cause inevitable mislabeling. This method successfully eliminates the pre-training process and the test mesh can be segmented and labeled just using a few examples which is much more convenient and efficient to conduct. Besides, the cases that multiple categories of objects are also taken into account. The robust $\ell_{2,1}$ norm which overcomes the distractions [5] [6] the guiding from geometric similarity as well as the labeling structure corporately obtain the correct labeling. This method can also address the problem when there are mislabeled meshes. Thus it is a more efficient and stable method compared to other existing learning based methods. Low-rank structure introduced in this paper has proved as an effective method for data analysis [7]. Also, *Candès et al*. [8] proposed the robust principal component analysis (RPCA) which recovers the principal components of a data matrix. Low-rank representation (LRR) [5] [6] further promotes this recovery from a single subspace to multiple independent subspaces. In terms of stability of segmentation, *Oana Sidi et al*. [9] introduced an algorithm for unsupervised co-segmentation of a set of shape to semantic parts. Unsupervised learning method does not use the labels that solely measures the correspondence and similarities of features to achieve the goal of classification. Unlike many researches that handle the shape segmentation problem based on one or a pair of shapes especially for unsupervised methods, they use a set of shapes. In addition, they develop the co-segmentation as a clustering problem based on the geometric shape descriptor in spatial coordinates fixed to the objects themselves so that it has the invariant stability regarding on the different shapes, poses and locations. Their method is superior for the highly varied shape set. Similarly, *Min Meng et al*. [10] also put up with an unsupervised method for shape segmentation. They used the over-segmentation results to cluster the primitive patches to get the initial result and then iteratively established a statistical model to describe the clusters from previous estimation and used the multi-class optimization to improve the performance. This method is leading in unsupervised learning methods and has the comparable results to the supervised methods. *Jiaxi Hu* and *Jing Hua* [11] introduced a method for extracting salient features from surfaces that are represented by triangle meshes. Instead of usual spatial domains, they employed the Laplace-Beltrami spectral domain to extract the feature points. This method gets the salient spectral geometric features invariant to spatial transformations. Besides, it is also scale-invariant no matter how the shape is scaled so that it can achieve the good performance for both global and partial matching. *Guillaume Lavoué et al*. [12] proposed a method that using Markov Random Fields for improving 3D mesh segmentation. It is also a clustering algorithm given the features like curvature, density and roughness. It is based on Markov Random Fields, graphical probabilistic models. Using both attributes and geometry in the clustering and obtain the optimal global solution only according to the local interactions. In addition, all the parameters are estimated and optimized in very short running time.

In this thesis, we develop a supervised learning method to segment the 3D shape meshes and classify them into meaningful parts with labels using multi-class SVM (Support Vector Machine). And in order to improve the boundary edges performance, we establish a second classifier, a binary SVM classifier, only focusing on the boundaries. Extract the geometric features from the 3D meshes consisting of point cloud [13]. After the 1st classifier, we used the result labels to extract the boundaries between different classes and then apply the 2nd classifier to strengthen the boundary edges. Besides, to conduct the homogeneous check, we apply k-means clustering which is an unsupervised learning process to enhance the boundaries and meanwhile correct the mis-assigned labels.

2. MATERIALS AND METHODS

2.1 DATASET

Nowadays, the 3D shape segmentation research has gain more population, however, the enormous methods are conducted using different dataset that lacks a uniform criterion. In this case, it is unreliable to compare all the existing methods if there is not a benchmark available. Princeton Segmentation Benchmark (PSB) [14] is a publicly available database of polygonal models collected from the World Wide Web and a suite of tools for comparing shape matching and classification algorithms [15]. We employ this dataset for all of our work. It provides 3D object meshes, as well as segmentations provided by human users. The labels to each class are also already manually assigned but needs some pre-processing that we will discuss in the later section. For different object categories, the training is separate.

Point-based graphics are very common types of 3D object graphics in research fields of 3D modeling, graphical rendering and engineering design requires geometric processing and related areas. Point cloud data which represents shape surface provides huge advantages for 3D shape understanding [16]. 3D point clouds are easier than ever to collect with the development of laser scanners and positioning systems [17] [18]. In our research on 3D meshes, PSB provides the shapes consisting of point clouds, and then the points set is triangulated to get a topological structure of the surface which is called a 3D mesh as shown in Fig. 1. Thus we can directly use the surfaces which are formed by triangulated meshes or applying the original point cloud to process the later work [19].

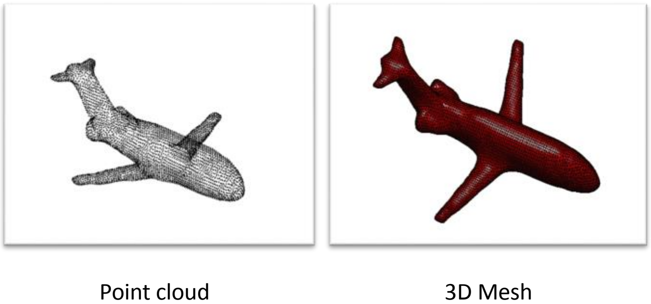


Figure. 1. Point cloud and 3D meshes

2.2 PRE-PROCESSING

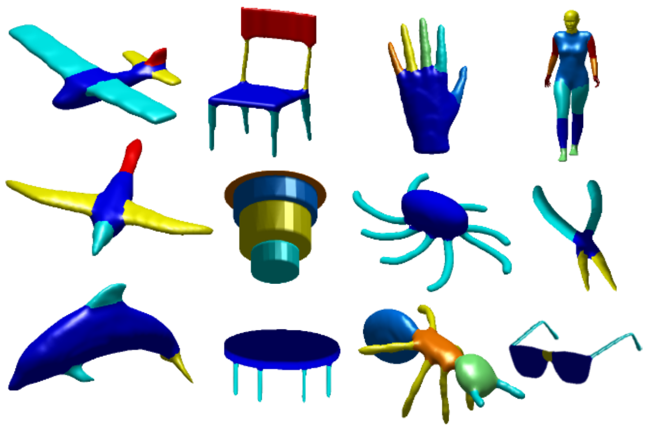


Figure. 2. Labeled shape meshes in PSB

In the PSB dataset, many of the mashes even within single object category have different directions in the coordinate. In this case, the matching and comparison between surface meshes may be somewhat affected by this inconformity. Most of features we use are not view-independent that means the method is not invariant to different poses, locations and directions. Due to the invariant property of shape is not our topic to research, we need conduct the pre-processing to align the meshes into the same direction in order to handle these meshes under a uniformed. We apply the 3D transformation which is a fundamental geometric method to change the object's direction in the 3D coordinates. In this work, only 3D rotation transformation is used since shape centers are all oriented on the origins of the coordinates.

Suppose a 3D point *P(x, y, z)* under the condition that the mesh center is oriented at the origin of coordinates, then rotate P using the rotation matrix regarding to the desired axis that rotates around. The new point after transformation *P'* is defined as *P'* = *M*P*.

Besides the transformation, another pre-processing we need to do is to unify the labels. The labels within a category may not consistent. The label represents the class that means the same part should share the same label. However, some meshes are not labeled in the same way, for example, the wings of one airplane are labeled as 1 while labeled as 2 in another airplane. Thus, we need simply repeat this processing for all the meshes to convenience the labels matching.

2.3 FEATURE EXTRACTION

Shape representations are used to collate the information stored in sensed 3D points so that surfaces can be compared efficiently. Shape can be represented in many different ways [20]. Such diversity of shape representations methods remains it an open research that can hardly reach a common consensus.

Feature is any distinctive aspect, quality or characteristic that may be symbolic (i.e., color) or numeric (i.e., height). The combination of d feature is represented as a D-dimensional column vector called a feature vector. The D-dimensional space defined by the vector is called feature space and the objects are represented as points in feature space [21]. Since the computation of features is time-consuming and adding features may increase the running time, we should try to use as informative features as possible [22].

In this work, we use both unary features that illustrate the similarities of single face and pairwise features that associated with neighboring faces. The unary features are the most important part that solely used can already lead to a meaningful performance, and the pairwise features can give the information at connection between faces so that they can strengthen the local performance. The unary features we use are singular values extracted from Principal Component Analysis of local face centers, average Euclidean distances, histogram and orientation features. In addition, multi-scale surface curvature and multi-scale dihedral angles between faces are used as the pairwise features. All these features are combined to form a basic 303-dimensional features vector x_i per face i.

2.3.1 UNARY FEATURES

Before extracting the features, a normalization is required for the data analysis. We use 30th percentile of distances between all pairs of vertices described as

$$Z = 0.3\frac{\sum_{i,j}^{n} d_{ij}}{n} \qquad (1)$$

Where d_{ij} is the distances between all pairs of vertices and n is the number of vertices.

a) PCA features

Compute the singular values s_1 s_2 s_3 of the covariance of local face centers, for various ratio (5%, 10%, 20%, 30%, 50%), and add the following features for each patch: $s_1/(s_1+s_2+s_3)$, $s_2/(s_1+s_2+s_3)$, $s_3/(s_1+s_2+s_3)$, $(s_1+s_2)/(s_1+s_2+s_3)$, $(s_1+s_3)/(s_1+s_2+s_3)$, $(s_2+s_3)/(s_1+s_2+s_3)$, s_1/s_2, s_1/s_3, s_2/s_3, $s_1/s_2+s_1/s_3$, $s_1/s_2+s_2/s_3$, $s_1/s_3+s_2/s_3$, yielding 75 features total.

As shown in Fig. 4, the process to compute the features is as following: for each face f_i, calculate the center c_i . Given the ratio, find the neighboring points to construct a cloud and compute the center of this cloud. Then calculate the covariance of centers and get the eigenvalue s_1 s_2 s_3 that is the principal component of the face centers [2].

b) Average Euclidean distance

The Average Geodesic Distance (AGD) function has been used for shape matching [22]. It measures how "isolated" each face is from the rest of the surface. Different to the Euclidean distance, the geodesic distance calculates the distance between points along the surface such like the distance on the globe which is not a direct distance between two points in space. It shows the shortest path from one point to another. The most common algorithm to calculate the geodesic distance is Dijkstra. However, since the data size in our work is enormously huge, calculating the geodesic distance is pretty time-consuming. Thus we just use the Euclidean distance since it does not matter when processing the data within the same and uniform way. The average Euclidean distance for each face is computed by averaging the distance from its face center to all the other face centers. In our case, we also consider the squared mean and the 10^{th}, 20^{th}… 90^{th} percentile. Then, we normalize each of these 11 statistical measures by subtracting its minimum over all faces.

c) Orientation features

We also include the x, y, z coordinates of each face. It's a straightforward feature that illustrates the orientation of the training set. For each triangular face with three vertices V_1, V_2, V_3, the center is simply calculated as

$$c_i = \frac{1}{3}\left(\sum_1^3 V_{xi}, \sum_1^3 V_{yi}, \sum_1^3 V_{zi}\right) \quad (2)$$

d) Histogram

Shape representations can span many different classes regarding to diverse aspects. For example, it can be classified by the number of parameters that are used for describing the primitives [20] such as using planar surface patches with many primitives but a few parameters for each, while using generalized cylinders with fewer primitives but each has many parameters though. Besides, the local and global nature of representation also gives a way for shape representation. The Gaussian image is global representations for single objects and surface curvature [23] estimates the local surface properties which can be used in complex scenes. Moreover, another factors that can be taken into account for shape representation is the coordinate system [20]. Viewer-centered coordinate system and object-centered system can be both applied. Viewer-centered representation [24] is easy to obtain but it requires to align the shapes in order to effectively compare the shapes. However, object-centered coordinate system can represent an object in a coordinate system fixed to the object. The description of the surface is view-independent so that surfaces can be directly compared without first aligning the surfaces.

The spin image is an example of object-centered coordinate system. In 3D problem, the spin image is generally created by oriented 3D points and associated directions. The definition of an oriented point includes the surface mesh vertex's 3D position and the normal [13] at the vertex. For surface matching, spin images are constructed for every vertex in the surface mesh. It projects the 3D points into a 2D dimension by fitting a planar on each point using the normal at this point. For each point, a 2D accumulator is then defined as the Fig.3 below shows. The resulting accumulator can be considered as an image. The dark parts suggest the denser bin that means large number of projected points drop into this bin.

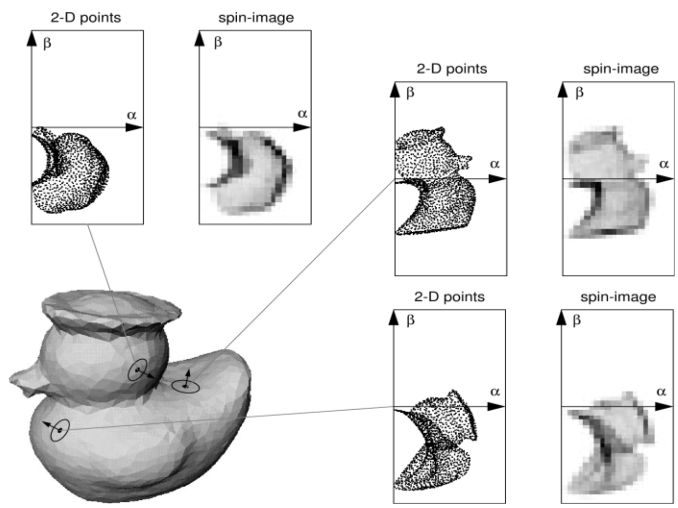


Figure. 3. Spin images [20]

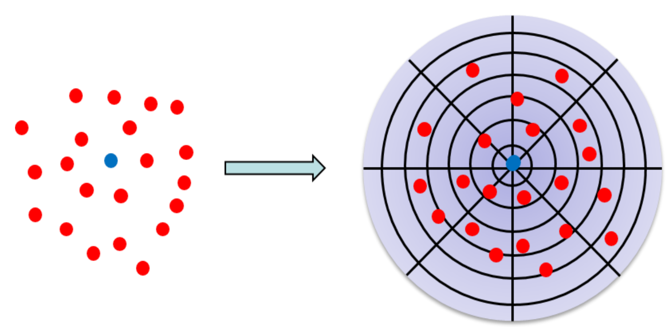


Figure. 4. Histogram

Spin image are 2D features transformed from 3D surface mesh. Similar to this idea of view-independent system, we extend it to 3D feature that can be call histogram. Instead of fit a plane on one point with the normal, we construct a sphere around each face center that the radius has the longest distance between this face center and other points on the entire surface mesh. We evenly separated the sphere with 8*8 bin resolution that divided it into 8 pieces in terms of radius and 360 degree angle respectively. Then we calculate the number of points that drop into each bin, totally yielding 64 features.

2.3.2 PAIRWISE FEATURES

Actually, solely using the unary features can already obtain reasonable performance that the surface meshes can be classified into parts effectively. However, the boundaries between different classes are classified roughly which badly influence the performance of classification. Thus, we consider using the pairwise features that give the information at the connection areas. The main role of the pairwise term is to improve boundaries between segments and to prevent incompatible segments from being adjacent. In our work, we use curvatures and multi-scale dihedral angles as the pairwise features.

e) Curvature features

Computing curvatures of surface is a fundamental important problem that is widely used for computer graphics and engineering applications. To compute the curvature of surface mesh, the surface fitting and approximation are often required. Surface fitting and matching is a common technology and has been commonly applied in engineering, statistics and computer graphics. Generally, we can get a set of discrete points by measures and extraction. Unfortunately, since these points are extracted by natural or human acts so that they may be not pretty precisely organized, and the function is hardly predicted, it is necessary to fit a surface using these points. Due to the error and noise of the rare point data we cannot require the fitting surface passes through all the points. The goal is to minimize the error of fitting surface.

The main research of surface matching is to describe, display, design and analyze surfaces in computer graphics systems. *Coons, Bézier et al.* made the fundamental basis of its theory in 1960s. By now, over more than 50 years' development, it has been a geometric theoretic system that is mainly including interpolation, fitting and approximation. There are wide ranges of surface approximation methods like Bézier, B-spline, Hermite, Least Square and so on. They are used in various practical situations regarding to their characteristics. For example, Bézier patch is defined by a rectangular array of control points whose dimensions determine the degrees of the curves comprising the isoperimetric curve net. It is not only an easy-going method but also has the convex hull property and advantage of global shape guarantee. However, there are also some disadvantages like lack of local flexibility, not suitable for complicated surface and problems of surface patches connection. Further, *de Boor* and *Cox et al.* presented B-spline surface fitting method remaining the merits of Bézier like perceptual intuition and convex hull property and also recovering the shortcomings of Bézier. It solves the local control and patches connection problems thus improves the preciseness interpolation. But B-spline is still having its evident and significant weakness that we will not extend here.

Considering the measure and experimental errors of discrete points, it is not a good way to use interpolation to approximate the surfaces. In this case, it would be better to use fitting method. One of the most common method is Least Square that is a universally applied idea of fitting and approximation. Minimizing the leas-Square leads to a set of linear equations and by solving this we can get the fitting surface. However, in practical situations such like huger size of discrete points and more complicated shapes, this method is not convenient and effective enough to be conducted, for example, sometimes needs divided into sections to do the smoothing The moving least-squares (MLS) was presented by *Levin* [44] in 1960s and its variants have been successfully used to define point-set surfaces in a variety of point cloud data based modeling and rendering applications [25][26][27][28][29].

Moving Lest-Squares (MLS) provides a method for surface fitting and matching [30]. It largely improves the traditional Least Squares to achieve more accurate and smoother surface. *Lancaster* and *Salkauskas* [31] firstly applied MLS for fitting surface and *Belyschko* [32] promoted it into non-mesh method. Compared to Least-Square method, it mainly has three aspects of improvements: 1) There is no need to subdivide the fitting and approximation areas, only discrete point cloud is needed; 2) Due the introduced compact supporting weights, it possesses the local flexibility; 3) Changing the parametric order of base function can conveniently control surface fitting and approximation preciseness and the changes of weight function can control the surface smoothness.

Instead of using the surface that is already given in the dataset, we approximate MLS surface by point cloud and then calculate the curvatures including Gaussian curvature, average curvature and principal curvatures on the surface. Now we give a brief introduction of MLS surface and detail can be referred in [29]. Approximate MLS surface by point cloud and then calculate Gaussian curvature, average curvature and principal curvatures on the surface, with various distance ratio (1%, 2%, 5%, 10% relative to the median of all-pairs distances). Let k_1 and k_2 be the principal curvatures of a patch. Also include the following features: k_1, $|k_1|$, k_2, $|k_2|$, k_1k_2, $|k_1k_2|$, $(k_1+k_2)/2$, $|(k_1+k_2)/2|$, k_1-k_2, yielding 36 features total.

The method of MLS can obtain the surface with C^∞ continuity from the point cloud. Given a point set $P = \{p_i\}$, $p_i \in R^3$, $i \in \{1,\dots, N\}$, MLS surface S_p defined by P is obtained through a projection. Around a point r_i , in the local coordinates fixed on it projection q_i , the polynomial patch is defined as

$$G_i = q_i + u\tilde{o}x_i + v_i\tilde{o}y_i + g_i(u,v)\tilde{o}n_i,\ (u,v) \in [-h, h^2] \quad (3)$$

When h is small, we can assume that the polynomial surface patch $G(u, v)$ is a good approximation of the original surface so that we can use it to calculate the curvature around r_i. Assume $g_i(u, v)$ is quadratic polynomial as $g_i(u, v) = a + bu + cv + duv + eu^2 + fv^2$. Thus

$$G_u = (1,0, b + d\ \ v + 2eu) \quad (4)$$

$$G_{uu} = (0,0,2e) \quad (5)$$

$$G_v = (0,1,c + d\ \ u + 2fv) \quad (6)$$

$$G_{vv} = (0,0,2f) \quad (7)$$

$$\mathrm{n} = \frac{1}{S}(-\mathrm{b} - \mathrm{dv} - 2\mathrm{eu}, -\mathrm{c} - \mathrm{du} - 2\mathrm{fv}, 1) \quad (8)$$

$$G_{uv} = (0,0,d) \quad (9)$$

Where $\mathrm{S} = \sqrt{(\mathrm{b} + \mathrm{d}\ \ \mathrm{v} + 2\mathrm{eu})^2 + (\mathrm{c} + \mathrm{d}\ \ \mathrm{u} + 2\mathrm{fv})^2 + 1}$, then

$$\mathrm{L} = 2\mathrm{e}/\mathrm{S} \qquad \mathrm{E} = 1 + (b + d\ \ v + 2eu)^2 \quad (10)$$

$$\mathrm{M} = \mathrm{d}/\mathrm{S} \qquad \mathrm{F} = (b + d\ \ v + 2eu)\ \ (c + d\ \ u + 2fv) \quad (11)$$

$$\mathrm{N} = 2\mathrm{f}/\mathrm{S} \qquad \mathrm{G} = 1 + (c + d\ \ u + 2fv)^2 \quad (12)$$

Finally, the Gaussian curvature on the surface is

$$k_G = \frac{LN - M^2}{EG - F^2} = \frac{4EF - d^2}{S^4} \quad (13)$$

The average curvature is

$$k_H = \frac{NE - 2MF + LG}{2(EG - F^2)} \quad (14)$$

The principal curvatures can be calculated by Gaussian curvature and average curvature as

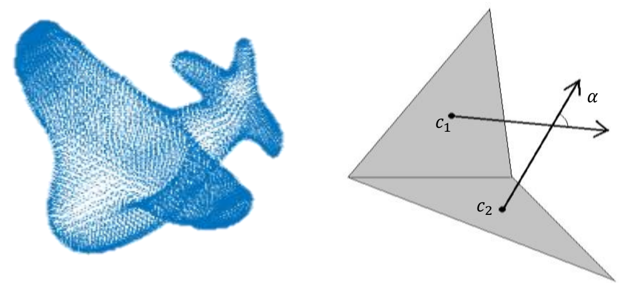
$$k_1 = k_H + \sqrt{{k_H}^2 - k_G} \quad (15)$$

$$k_2 = k_H - \sqrt{{k_H}^2 - k_G} \quad (16)$$

f) Dihedral angles

Let ω_{ij} be the exterior dihedral angle between faces i and j. The basic feature is given as *min* (ω_{ij}/π). We also compute the average of the dihedral angles around each edge at ratio of 0.5%, 1%, 2%, 4% of the median of all-pairs distances in the mesh. We then exponentiate each of the above features with each exponent in the range 1 to 10 such like $f^1, f^2, f^3, \ldots, f^{10}$, yielding 50 dihedral angle features totally.

Firstly, we need to compute the normal on the surface. For each triangular face, calculate the two tangent vectors on the face center and the normal should be perpendicular to these two vectors so that by repeating this process we can obtain all the normal on the surface. Afterwards, the dihedral angle can be represented as the angle between the normal vectors of two faces as shown in Fig.5.



Normal of the surface

Dihedral angles between two faces

Figure. 5. Normal and dihedral angle

From Fig.6, we can see the improvement when adding the pairwise features. The accuracy of classification is only 46.41% while it reaches 83.14% after applying the pairwise features which is a significant improvement.

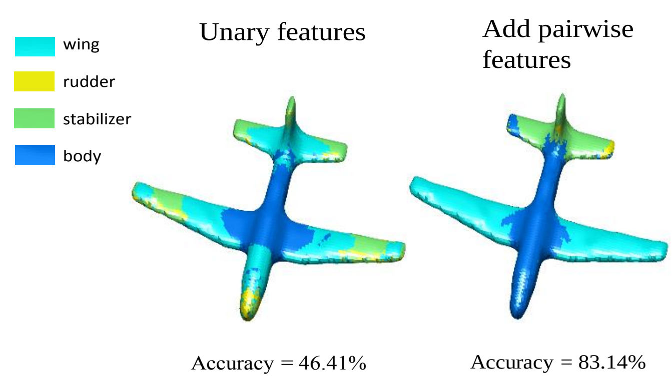


Figure. 6. Improvement by applying pairwise features

2.4 SVM LEARNING

In the research of machine learning [35], Support Vector Machine is a supervised learning model that is widely used in pattern recognition, classification and regression analysis. Pattern is a composite of traits or features characteristic of an individual. In classification, a pattern is representative as a pair of variables $\{x, \omega\}$ where x is a collection of observations or features called feature vector and ω is the concept behind the observation called label. The goal of a classifier is to partition feature space into class-labeled decision regions and the borders between decision regions are called decision boundaries [36].

Typically, a pattern recognition system contains:

1) Sensor that collect the rare data
2) Preprocessing mechanism
3) Feature extraction mechanism (manual or automated)

4) Classification or description algorithm
5) A set of examples (training set) already classified or described

2.4.1 LINEAR CLASSIFICATION

Generally, SVM is a binary classifier model [37] defined as the linear classifier that has the maximum marginal in feature space. It takes the advantages of flexibility and convenience since it can be easily applied into many classification situations such as linear and non-linear, binary and multi-class. It is also a confident and reliable classifier that it is less possible to have the overfitting problem than other methods.

Assume there are two classes, we need to develop a classifier to separate them which is a line in this case. Actually, there can be infinite such lines that can separate these two classes. We assume that one class is marked as *-1* while another is *+1*, the line is $f(x) = \omega \cdot x + b$ where x and ω are support vectors. When the dimension of x is *2*, $f(x)$ defines a line, when x is *3* and now $f(x)$ represents a plane in 3D space. Moreover, if the dimension is more than *3*, it defines a hyperplane with the dimension of *n-1* in *n-D* space. As we discuss above, we already mark the two classes of points that are *+1* and *-1* so that we can just use $sgn(f(x))$ to predict a new point where *sgn* means the sign function. $sgn(f(x)) = +1$ when $f(x) > 0$ while $sgn(f(x)) = -1$ when $f(x) < 0$. However, the key question is how to choose the optimal line $f(x)$ that can classify these two classes. In a word, we need make the nearest point to this line has the longest distance that is to have the maximum margin width. The margin width is defined as

$$M = \frac{2}{\sqrt{\omega \cdot \omega}} \tag{17}$$

Besides, the support vectors are on the lines $\omega \cdot x + b = +1$ and $\omega \cdot x + b = -1$. Now we add y that represents the class ($y = +1$ or -1) so that we have obtain the support vector function $y(\omega \cdot x + b) = 1$. By now, we get the classification function that we need optimize:

$$\max \frac{1}{||\omega||} \tag{18}$$

Then, we can transform this function into the form as below which is a least square problem:

$$\min \frac{1}{2}||\omega||^2 \tag{19}$$

$$\text{s.t.}, y_i(\omega^T x_i + b) \geq 1, i = 1, \ldots, n$$

It is a quadratic programming problem with constrain and the problem is convex that means we can definitely obtain the global optimal results. This optimization problem can be transformed and solved by using Lagrange multiplier that we do not extend this in detail here.

2.4.2 MULTI-CLASS CLASSIFICATION

In the previous sections, we discuss the binary classification problem. Actually, SVM is simply born as a binary classifier. However, it still presents several ways to solve the multi-class problem [37]. One is *1* vs *1* and the other one is *1* vs *(n-1)*. For the first method, *n(n-1)* classifiers are needed and the classifier (*i*, j) can tell if the point belongs to class *i* or class *j*. For the latter method, we need *n* classifier that *i*th classifier can tell the sample belongs to class i or the complementary set of class i. From the research of *Dr. Lin* (the developer of libsvm), the method *1* vs *1* is better that *1* vs *(n-1)* and has been approved to have high accuracy [38].

In this work there may be more than two classes for the meshes parts so that we need establish a process for the multi-classification. In our work, we apply the *1* vs *1* method. Firstly, establish classifier for arbitrary two classes, we have *n(n-1)/2* or C_n^2 classifiers. Then use vote method to calculate the total times of which class each face belongs to. The class having maximum vote is the final class of x_i.

2.4.3 APPLY SVM LEARNING

The shapes are provided of the point cloud and the surface mesh. Generally, we extract features on all faces and form the training set. However, both the features extraction and training process are time-consuming and also require the memory of computer. Due to the huge data, it is not efficient and practical to do with all of the faces. Moreover, since the samples are the faces that there is already huge amount even in one mesh, there is no worry that the training set is too small and it even does not matter whether the training samples belong to the same mesh. Thus we do not use all the faces in each mesh to reduce the workload. Under this condition, we propose a selection of faces as training samples and meanwhile we can test on the rest faces.

Assume we have a few meshes from same object category. However, we do not use all the faces in them thus we need select the training samples from them. Now we randomly select the training samples. But if we choose them completely in a random way, we may miss some classes which may lead to serious mis-classification. Thus, we set a ratio to randomly select the samples within each class to make sure including all the classes in the training set. This method is not only good at reduce the time and memory cost, but also keeps the randomness and global property of the training set. Moreover, it is also convenient do obtain the testing set are the rest of samples.

In addition to the training and testing, we also need to do the validation check for classification performance. Even though there is no overlap between training and testing set, if they come from the same mesh group, they still maintain the conformity and similarity between the training and testing sets. Thus, we also need to do the validation check that test on another mesh that are exactly different to the training and testing meshes. Generally, it is reasonable that the performance of validation check is worse than testing. In addition, the ratio should be selected carefully. If it is too small that the training set is so small that may lose the global property and cause overfitting while too many samples with larger ratio will greatly increase the time and memory cost.

In practical operation, as Fig.7 shows, we have four meshes of human, take one of them as the evaluation samples, then select the training and testing samples in the other three meshes with a certain ratio and also test on the mesh outside. At the second time, take another mesh out and do the same thing. By repeating this process, we can report the average results for testing and evaluation.

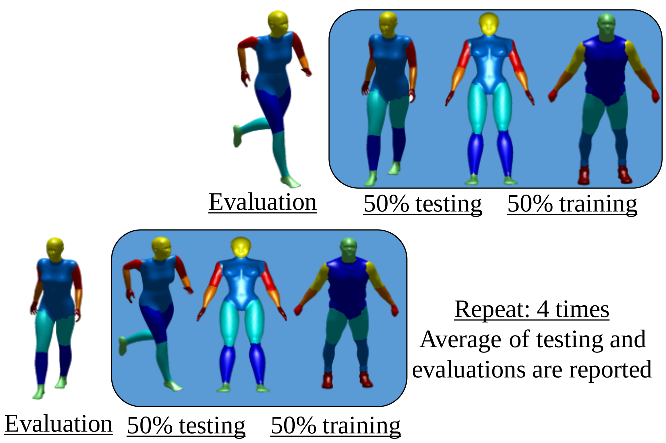


Figure. 7. Training, testing and evaluation

3. BOUNDARY IMPROVEMENT

By now, from the process above, we can already have the reliable performance that the shape meshes can be classified into meaningful parts successfully. But the boundary parts between different classes are still not neatly classified. So we consider to do some modifications for the boundary parts.

Benhabiles H, LavouéG et al. [39] presented a method of boundary edges learning for 3D mesh segmentation. An objective boundary edges function is learned from a set of segmented training meshed and use this function to segment an input meshes. The function is learned using Adaboost classifier [40] that automatically selects from a set of geometric features the most relevant ones to detect the boundary edges.

Adaboost is a non-linear classifier that is also widely applied in machine learning. However, it largely depends on the features and the size of training set. Since SVM can be also used for non-linear problems, for example use the kernel function or add a penalty function to linear classification function, we still develop a method using SVM to learn the boundary edges. We make two layers of classification. For the 1st classifier, only use the unary features. After this process, we get the output labels. And then we use these labels to extract the boundaries of the mesh. Since there may be misclassified faces, the extracted boundaries are not exactly correct so that the homogeneous check is needed in order to improve the boundaries. In this process, we use K-means to find the centers of clusters to remove the mis-extracted or noisy points to get the boundaries that make more sense. Then, we conduct classifier 2 just focused on boundary edges to train them that is a binary classification process to improve the performance on boundaries. The result shows that by the boundary processing, the performance can be improved.

3.1 WORKFLOW

After talking about the work process, it is necessary to outline the work entirely. As the details in Fig.8, firstly, we conduct some pre-processing on the raw data which are the point cloud and the triangulated mashes in PSB, aligning the meshes into same direction in coordinates, unifying the labels to make sure they have the same criteria. Secondly, extract the unary and pairwise geometric features of the data as the training set. To improve the boundary classification performance, we establish two layers of classification. We use all the features for the first level of classifier. After this, we can get a set of labels showing the classification results. There may be many parts that mislabeled. Then, we extract the boundary areas of the mesh. Due to the mis-classifications and the error caused by the ratio when extracting boundaries, some points may be mis-extracted as boundaries, thus we need take the homogeneity check to find out these points and then correct their labels. To reach the goal, we apply K-means, which is a clustering method that we will discuss later in detail, to capture the noisy points to boundaries. After that, we will have the more accurate boundaries. Then apply the pairwise features on the boundary edges and conduct the second classifier to obtain the final labels.

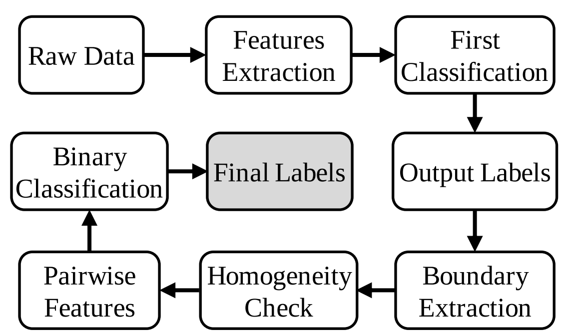


Figure. 8. Workflow of our method

3.2 BOUNDARY EXTRACTION

As we discuss in the last section, we need extract the boundaries and do the homogeneity check. Boundary extraction is a straightforward process. For each point, we find the neighboring points set associated to a certain ratio, if there are points within this set that the labels are different with this point, they can be described as boundaries. Repeat this process, we can obtain the boundaries all over the surface mesh as shown in Fig.9.

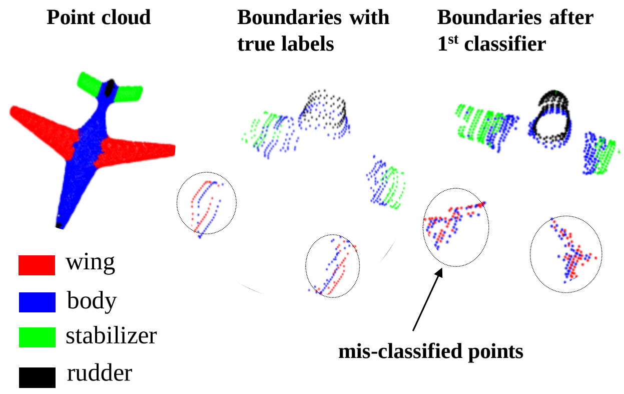
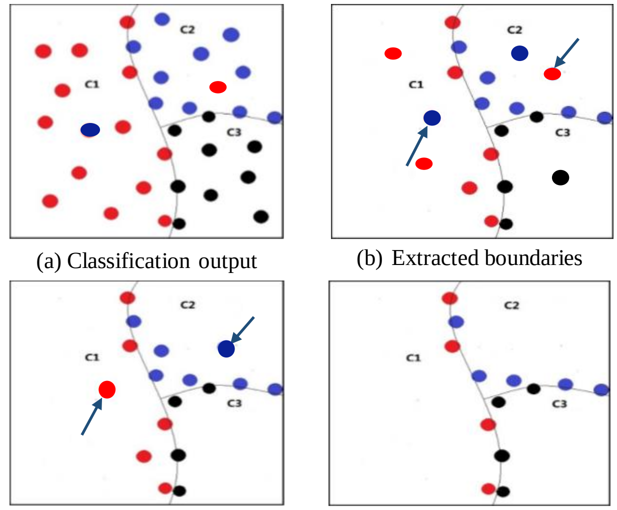


Figure. 9. Boundary extraction

3.3 HOMOGENEOUS CHECK

After extracting the boundary areas, however, the Fig.10(b) shows there are some mis-extracted points that are not supposed to be boundaries and some of them are even mis-classified due to the mis-classifications resulted from the 1st level classifier or sometimes error by distance ratio. To solve this problem, before continue the 2nd level of classifier, we need to remove the noisy points. One method we can do is to modify the ratio for finding the neighbors to change the boundaries range. However, this method is too experiential and artificial that are largely dependent on the human manual action. There is another way we can take into account. The mis-extracted points are mostly isolated from the boundaries seen as noise so that we can propose a way to filter these noises. We apply K-means which is a commonly applied unsupervised clustering algorithm.



(a) Classification output
(b) Extracted boundaries
(c) Homogeneous check
(d) Neat boundaries

Figure. 10. Homogeneous check

Clustering is a technology that analyzes the static data that are widely used in machine learning, pattern recognition, image processing and so on. It is a method that classifies the data into groups or subsets by the static classification method to make the components within the same class have the similar properties. Generally, we use clustering analysis as the unsupervised learning algorithm that there is no label in the process and the classification is only based on the similarities of sample points.

K-means is an interactive algorithm based on the distance. It classifies n observation samples in to k clusters in order to make each sample to its corresponding center have the minimum distance compared to other centers. The entire process is demonstrated below:

1) Randomly select k samples as the initial clusters centers from all the observation samples. In our work we set k as the numbers of clusters of boundaries regarding on different categories. And assign all the observation samples into the k clusters based on the shorter distance. Thus, we have the initial cluster decision which is the first iteration.

2) We now have k clusters and calculate the new center of each cluster which is mean value of cluster. Then for each center find the nearest samples and bring them into the new cluster.

3) Repeat the step 2) until the centers do not change anymore.

By K-means processing, we obtain the cluster centers of boundaries. Then make a decision criterion that if the distance of a point to its center is large, based on a certain distance ratio, then describe the point as an isolated point. There is an assumption that the distance of a point to its center is shorter than to other clusters so that if this distance is still large it can be decided as the noisy point.

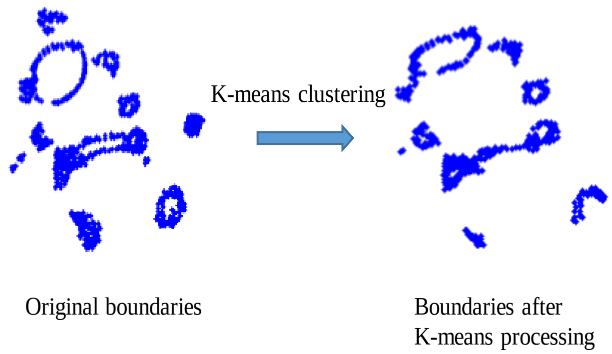


Figure. 11. K-means for homogeneous check

Now we obtain the new boundaries that are more neat and clearer than before because we remove the isolated points to the boundary areas as shown in Fig.11. And we also correct the labels of mis-classified points within the noisy points. Afterwards, we can use this boundary to continue the 2nd level classification.

3.4 RE-CLASSIFY BOUNDARY

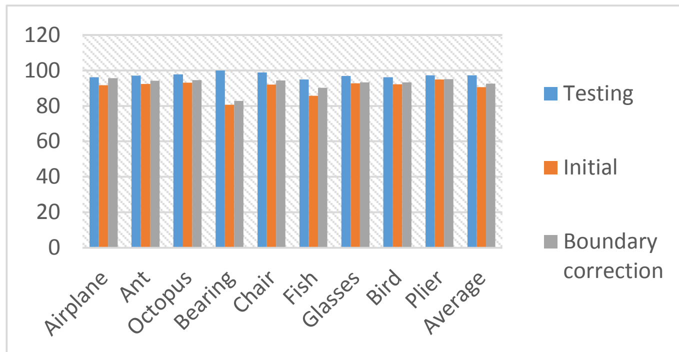
In our work, as illustrated in the workflow, in order to improve the boundaries performance, we come up with two levels classification. In the first classification process. By the 1st level of classifier, we should get the meaningful result but not work very well on boundaries. Now we want to make some improvement for the edges between classes. After extracting the boundary areas according to the labels resulting from 1st level classifier, it is necessary to processing the boundary data since there may be some mis-labeled points as we discussed in the previous section. Now we only apply the pairwise features on the boundary points and learn the 2nd classifier that only focuses on edges. After that we get the new labels for the boundaries and substitute them into the original result labels to achieve the final result.

In addition, the 2nd classifier should be a binary classifier because around the boundaries we should have only two classes and then we re-classify them to strength them. However, since there are errors in the 1st level of classifier, together with the error resulting from boundaries extraction and homogenous check, or sometimes the boundary is an intersection with three or more classed, we may have more than two classes around edges. In this case, we can simply calculate the number of selected boundary samples for each class respectively and extract the most frequent two classes to form the boundaries.

4. EXPERIMENTAL RESULTS AND CONCLUSION

We now evaluate the quality of the labels produced by our method. As discussed before, by selection of samples, we randomly select the samples by a certain ratio and conduct the testing using the rest samples. From Tab.1, we report the testing result and the average of testing is 97.22%. Besides, we also take the validation check. We evaluate our method using leave-one-out cross-validation. We report Recognition Rate and Average over all categories, our method obtains approximately 90.59% accuracy. Then we also report the results after boundary correction that can reach 92.62% which is improved by 1.8%.

Table 1. Classification results of our method



We also report the result including three experimental stages of the estimation of our method. Firstly, we apply only the unary features over all faces that is the first stage. In this stage, there may be many mis-classifications, in Fig.12 for example, the accuracy of this stage is only 46.41%. In order to improve them, we also add the pairwise features that is the second stage. We can find the performance of airplane in this stage is much better that the accuracy reaches 83.14%. However, the edges between classes are still not neat and clear enough so that then we conduct the boundaries improvement discussed above. From the third Fig. of airplane, it is evident to see the significant improvements and shows the best result.

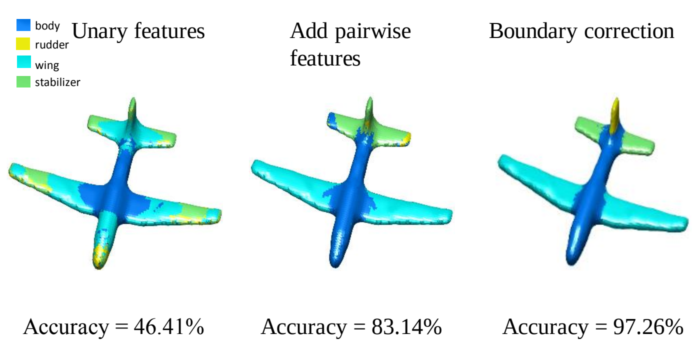


Figure. 12. Improvements of experimental stages

For the detail of boundary improvement, in Fig.13 we compare the performances of initial output and after boundary correction and we also mark the most significant improved parts with circle to emphasize the improvement.

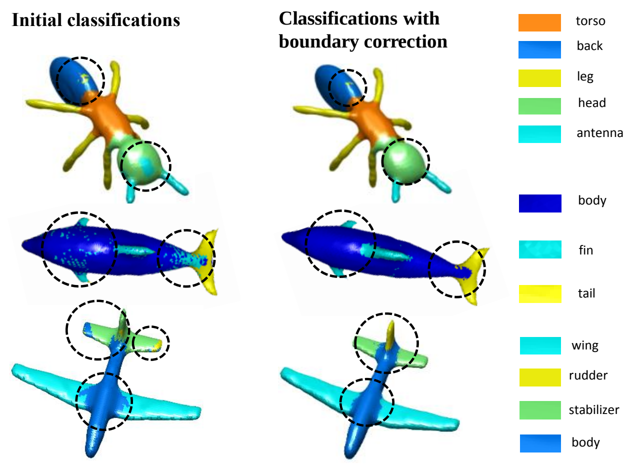


Figure. 13. Boundary improvements

Finally, we report the final output figures in Fig.14 shows our method works well in classifying the meshes and has stable performance over different categories.

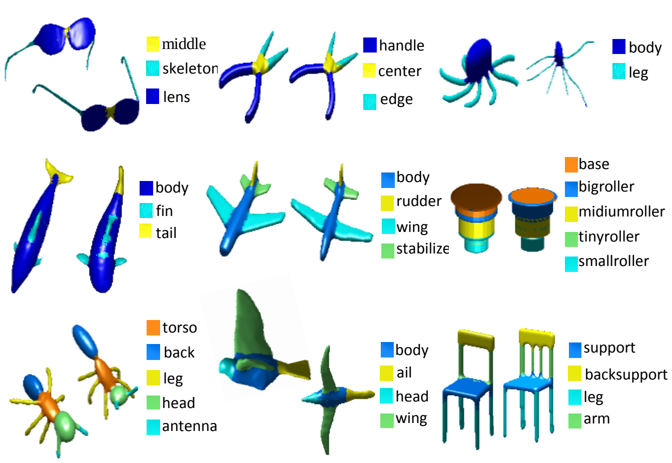
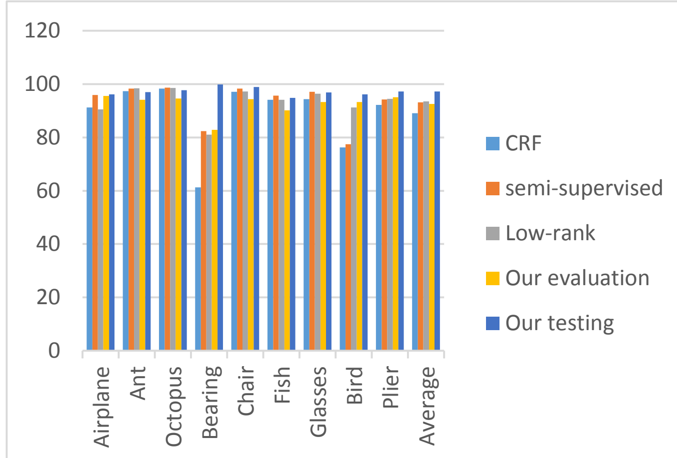


Fig. 3. Classification results

From Tab.2, we report the comparison with other methods in state-of-the-art. We cited three main methods CRF [2], semi-supervised mesh segmentation [3] and low-rank 3D segmentation. From the comparison of averages, our method is comparable to others. Especially compared with CRF since it is also a supervised method, our method is superior to it with less features and less training samples. Moreover, when analyze into detail over all the categories, some of other methods may have pretty low accuracies, for example it is only 61.3% for Bearing with CRF and around 77% for Fish in CRF and Semi-supervised method, while they are all above 80% in our methods over all the categories suggesting that our method has more stable performance.

Table 2. Comparison with other methods



	M1	M2	M3	Our method	Testing
Airplane	91.2	95.9	90.5	95.6	96.18
Ant	97.4	98.3	98.4	94.18	97.02
Octopus	98.3	98.7	98.5	94.62	97.74
Bearing	61.3	82.4	81.1	82.88	99.87
Chair	97.1	98.3	97.2	94.38	98.89
Fish	94.1	95.7	94.1	90.2	94.89
Glasses	94.4	97.1	96.4	93.34	96.93
Bird	76.3	77.5	91.3	93.3	96.22
Plier	92.2	94.2	94.5	95.1	97.27
Average	89.14	93.12	93.56	92.62	97.22

In sum, our method with SVM successfully achieves the research goal of 3D mesh segmentation and labeling simultaneously. By developing the boundary improvement processing, the classification accuracy can improve by 1.8%. In addition, the evaluation of our method is comparable to other researches in the state-of-the-art while the testing performance is superior. Furthermore, the stable performances for all the object categories show that our method is reliable for the 3D mesh segmentation and labeling task.

REFERENCES

[1] Shamir A. A survey on mesh segmentation techniques. Compute Graph Forum 2008;27(6):1539–56.
[2] KalogerakisE, HertzmannA, SinghK. Learning 3D mesh segmentation and labeling. ACMTransGraph2010;29(4):102.
[3] Lv J,ChenX, HuangJ, BaoH. Semi-supervised mesh segmentation and labeling. ComputGraphForum2012;31(7):2241–8.
[4] Xiuping Liu, JieZhang, RishengLiu, BoLi, JunWang, JunjieCao Low-rank 3D mesh segmentation and labeling with structure guiding Computers &Graphics46(2015)99–109

[5] Liu G,LinZ, YanS,SunJ, YuY, MaY. Robust recovery of subspace structures by low-rankrepresentation.IEEETransPatternAnalMachIntell2013;35 (1):171–84.
[6] Risheng Liu, Zhouchen Lin , De la Torre, F., Zhixun Su, Fixed-rank representation for unsupervised visual learning, Computer Vision and Pattern Recognition (CVPR), 2012, 598-605.
[7] TangK, LiuR, SuZ, ZhangJ. Structure-constrained low-rank representation. IEEE TransNeuralNetwLearnSyst2014:1.
[8] Candès EJ, Li X, Ma Y, Wright J. Robust principal component analysis? J ACM 2011;58(3):11.
[9] Oana Sidi, Oliver van Kaick, Yanir Kleiman, Hao Zhang, Daniel Cohen-Or. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. ACM TransGraph2011;30(6):126.
[10] Meng M, XiaJ, LuoJ, HeY. Unsupervised co-segmentation for3D shapes using iterative multi-label optimization. Comput-AidDes2013;45(2):312–20.
[11] GAL, R., AND COHEN-OR, D. 2006. Salient Geometric Features for Partial Shape Matching and Similarity. ACM Trans. Graph. 25, 1.
[12] LAVOU´E, G., AND WOLF, C. 2008. Markov Random Fields for Improving 3D Mesh Analysis and segmentation. In Eurographics workshop on 3D object retrieval.
[13] Zhang J, CaoJ,LiuX, WangJ, LiuJ, ShiX. Point cloud normal estimation via low-rank subspace clustering.ComputGraph2013;37(6):697–706.
[14] http://shape.cs.princeton.edu/benchmark/
[15] CHEN, X., GOLOVINSKIY, A., AND FUNKHOUSER, T. 2009. A Benchmark for 3D Mesh Segmentation. ACM Trans. Graphics 28, 3.
[16] KATZ, S., LEIFMAN, G., AND TAL, A. 2005. Mesh segmentation using feature point and core extraction. Visual Computer 21, 8.
[17] MUNOZ, D., VANDAPEL, N., AND HEBERT, M. 2008. Directional Associative Markov Network for 3-D Point Cloud Classification. In Proc. 3DPVT.
[18] Pinghai Yang and Xiaoping Qian Direct Computing of Surface Curvatures for Point-Set Surfaces Eurographics Symposium on Point-Based Graphics 2007.
[19] Vikas Sindhwani Partha Niyogi niyogi Mikhail Belkin, Beyond the Point Cloud: from Transductive to Semi-supervised Learning ICML ’05 Proceedings of the 22nd international conference on Machine Learning ACM 2005 p.824-831.
[20] Johnson AE, Hebert M. Using spin images for efficient object recognition in cluttered3Dscenes.IEEETransPatternAnalMachIntell1999;21(5):433–49.
[21 https://www.mendeley.com/research/lecture-9-introduction-pattern-analysis-features-patterns-classifiers/?utm_source=desktop&utm_medium=1.14&utm_campaign=open_catalog&userDocumentId=%7Be4c57b2f-0827-4f61-8e52-d2aa1d73cf2d%7D
[22] HILAGA, M., SHINAGAWA, Y., KOHMURA, T., AND KUNII, T. L. 2001. Topology Matching for Fully Automatic Similarity Estimation of 3d Shapes. In SIGGRAPH.
[23] Chin Seng Chua, Ray Jarvis 3D free-form surface registration and object recognition International Journal of Computer Vision, January 1996, Volume 17, Issue 1, pp 77-99.
[24] Chitra Dorai, Anil K. Jain, COSMOS-A Representation Scheme for 3D Free-Form Objects, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 19 Issue 10, October 1997, Page 1115-1130
[25] AMENTA N., KIL Y. J.: Defining point-set surfaces. ACM Trans. Graph. (2004), vol. 23(3), pp. 264-270.
[26] AMENTA N., KIL Y. J.: The domain of a point set surface. Eurographics Workshop on Pointbased Graphics (2004), pp. 139-147.
[27] [DGS05] DEY T. K., GOSWAMI S., SUN, J.: Extremal surface based projections converge and reconstruct with isotopy. Technical Report OSU-CISRC-4-05- TR25 (April 2005).
[28] [DS05] DEY T. K., SUN J.: Adaptive mls surfaces for reconstruction with guarantees. Proc. Eurographics Symposium on Geometry Processing (2005), pp. 43-52.
[29] Pinghai Yang and Xiaoping Qian, Direct Computing of Surface Curvatures for Point-Set Surfaces, Eurographics Symposium on Point-Based Graphics (2007)
[30] Qinghong Zeng and Detang lu Curve and Surface Fitting based on Moving Least Square Journal of Engineering Graphics N0.1 2004.
[31] Lancaster P, Salkauskas K. Surfaces generated by moving least squares methods[J]. Mathematics of Computation, 1981, 37 (155): 141~158.
[32] Belytschko T, LuY Y, Gu L. Element-free galerkin methods[J]. Int.J.Numer.Meth.Engng., 1994, (37): 229~256.
[33] Marcano-Cedeño, A, Quintanilla-Dominguez, J, Cortina-Januchs, M.G. and Andina, D., Feature selection using Sequential Forward Selection and classification applying Artificial Metaplasticity Neural Network, IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society, 2845 – 2850. , 7-10 Nov. 2010.
[34] Ming Li, , Baozong Yuan, 2D-LDA: A statistical linear discriminant analysis for image matrix, Pattern Recognition Letters Volume 26, Issue 5, April 2005, Pages 527–532.
[35]http://www.cnblogs.com/LeftNotEasy/archive/2011/05/02/basic-of-svm.html
[36] VAPNIK C V, Support Vector Network [J], Machine Learning, 1995, 20 (2):273-297.
[37] Liang Sengzhuo Xie Wenxiu and Li Mang Improved 1-vs-1SVM Multi-class Classification Algorithm Journal of Nanchang University (Natural Science) Vol.37 No.3 Jun, 2013.
[38] Zhigang Liu Deren Li and Qianqing Qin etc. The Promotion of Support Vector Machine on Multi-classification [J], Computer Engineering and Application, 2004, 40 (7): 10-13, 65.
[39] Benhabiles H, Lavoué G, VandeborreJ-P, DaoudiM. Learning boundary edges for 3D-meshsegmentation.ComputGraphForum2011;30(8):2170–82.

[40] Freund Y., Schapire R. E., A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences, 1997.
[41] SHAPIRA, L., SHALOM, S., SHAMIR, A., ZHANG, R. H., AND COHEN-OR, D. In Press. Contextual Part Analogies in 3D Objects. International Journal of Computer Vision.
[42] BELONGIE, S., MALIK, J., AND PUZICHA, J. 2002. Shape Matching and Object Recognition Using Shape Contexts. IEEE Trans. Pattern Anal. Mach. Intell. 24, 4.
[43] Belongie S,MalikJ, PuzichaJ. Shape matching and object recognition using shape contexts.IEEETransPatternAnalMachIntell2002;24(4):509–22.
[44] Levin D. Mesh-independent surface interpolation [C]. Advances in Computational Mathematics, 2001.

