Miao YW, Feng JQ, Xiao CX *et al.* Differentials-based segmentation and parameterization for point-sampled surfaces. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 22(5): 749∼760 Sept. 2007

Differentials-Based Segmentation and Parameterization for Point-Sampled Surfaces

Yong-Wei Miao[1,2] (缪永伟), Jie-Qing Feng[1] (冯结青), Chun-Xia Xiao[1] (肖春霞), Qun-Sheng Peng[1] (彭群生) and A.R. Forrest[3]

[1]*State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou 310027, China*

[2]*College of Science, Zhejiang University of Technology, Hangzhou 310032, China*

[3]*School of Computing Sciences, University of East Anglia, Norwich, NR4 7TJ, U.K.*

E-mail: {miaoyw, jqfeng, cxxiao, peng}@cad.zju.edu.cn

Received September 30, 2006; revised May 6, 2007.

Abstract Efficient parameterization of point-sampled surfaces is a fundamental problem in the field of digital geometry processing. In order to parameterize a given point-sampled surface for minimal distance distortion, a differentials-based segmentation and parameterization approach is proposed in this paper. Our approach partitions the point-sampled geometry based on two criteria: variation of Euclidean distance between sample points, and angular difference between surface differential directions. According to the analysis of normal curvatures for some specified directions, a new projection approach is adopted to estimate the local surface differentials. Then a k-means clustering (k-MC) algorithm is used for partitioning the model into a set of charts based on the estimated local surface attributes. Finally, each chart is parameterized with a statistical method — multidimensional scaling (MDS) approach, and the parameterization results of all charts form an atlas for compact storage.

Keywords computer graphics, point-sampled surface, segmentation, parameterization, k-means clustering, multidimensional scaling

1 Introduction

When processing meshes or point-sampled surfaces, many applications require a proper parameterization of the surface, such as texture mapping and texture synthesis[1∼3], surface morphing and editing[4∼6], surface remeshing and multiresolution analysis[7∼9], geometry compression[10∼12] etc. The key issue behind a proper parameterization is how globally or locally to flatten the surface as isometrically as possible. A good survey of surface parameterization was given by Floater and Hormann[13].

Naturally, surface parameterization is closely related to the topic of surface development. From knowledge of differential geometry, only certain kinds of surfaces are developable[14], for example, cylinder surface, cone surface, and tangent surface. It is thus difficult to parameterize a general surface globally without distortions (in terms of angle or distance distortions), especially if the surface has areas of high curvature. This is also true for surface meshes and point-sampled geometry. Due to the success of the atlas approach for texture mapping[15], the piecewise parameterization approach received more and more attention in recent years[16∼20]. The piecewise approach consists of three steps in general: first, a surface patch is partitioned into a set of charts, each chart is then parameterized individually as a region of a texture domain, and finally, all parameterization results are combined to form an atlas.

However, this piecewise parameterization scheme is applicable only to polygonal meshes or triangular meshes, which possess globally consistent topological information. The connectivity information facilitates geometry processing and parameterization. Point-based representation of geometry is essentially a discrete sampling of a continuous surface without topological information. Although topological information may be preserved by a graph which connects each sample point to its nearest neighbors[21,22], the high complex connectivity information leads to large distance distortions or time-consuming parameterizations. Moreover, how to select a suitable neighborhood for estimating the local geometric attributes of the surface at each sample point is still an open problem.

In this paper, we propose an efficient segmenta-

Regular Paper

This work is supported by the National Grand Fundamental Research 973 Program of China under Grant No. 2002CB312101, the National Natural Science Foundation of China (NSFC) under Grant Nos. 60503056, 60333010, and the Natural Science Foundation of Zhejiang Province under Grant No. R106449.

tion and parameterization approach for point-sampled surfaces. The proposed approach partitions a given point-sampled surface based on two criteria: variation of Euclidean distance between sample points, and angular difference between surface differential directions which are estimated by a projection method. Subsequently, each chart is parameterized with a statistical method so as to minimize the distance distortions. Our main contributions can be summarized as follows:

- an estimation algorithm of local surface differentials at sample points by analyzing surface normal curvature using the projection method;
- a partition scheme which divides the surface model into a set of charts through k-means clustering (k-MC) according to two criteria;
- a piecewise parameterization approach with distance distortion minimizing — a multidimensional scaling (MDS) solution.

Our motivation for using k-means clustering and multidimensional scaling of the unstructured point-sampled geometry lies in the following observations: there is a high coherence in local point set regarding their intrinsic geometry properties, and the goal of low distance distortion for parameterization can be achieved using statistical methods on a discrete point cloud.

2 Related Work

Estimating Local Surface Differentials. The local shape of point-sampled surfaces can be described in terms of differential geometry concepts such as normal, principal normal, curvature and principal curvature, etc. These geometric parameters are not only required for generating realistic images of point-sampled surfaces, but also play an important role in surface modeling and editing[23∼25].

Pauly *et al.*[24] applied the principal component analysis method to the neighborhoods of sample points to estimate normals and curvatures. Based on Levin's MLS approximation method[26], Alexa and Adamson[27] adopted the gradient of the local implicit surface as an accurate surface normal estimation and presented efficient orthogonal projection operators for sampling theory. Translating from Taubin's integral eigenvalue method (IEM)[28] for estimating the tensor of curvature for polygonal meshes, Lange and Polthier[29] derived a similar method for estimating principal curvatures and principal curvature directions for point set surfaces. Nevertheless, their approach is computationally complex and time-consuming.

Chart Partitioning. There are several methods for partitioning surface meshes into charts. Krishnamurthy and Levoy[30] proposed an interactive approach for partition reconstruction. An automatic segmentation method is proposed by Maillot *et al.*[15], by which face clustering is guided by their normal distribution. Sander *et al.*[16] used a region-growing approach to segment the mesh, then merge charts according to the criteria of both planarity and compactness by a greedy face clustering algorithm. These approaches produce charts with convex boundaries. This restriction may increase distortions in the final parameterization.

The algorithm by Levy *et al.*[1] searches for sharp feature curves which correspond to high curvature of the surface and then grows charts so as to align chart boundaries with these feature curves. Yamauchi *et al.*[31] proposed a mesh segmentation method employing the integrated Gaussian curvature to measure the developability of a chart. They evenly distributed the integrated Gaussian curvature over the charts and automatically ensured a disk-like topology for each chart. Sander *et al.*[18] proposed a chartification approach based on the dual graph of the surface mesh to form compact charts. Their cutting curves are along high-curvature lines, which are traced with the Dijkstra search algorithm. Katz and Tal[32] hierarchically decomposed surface meshes using a fuzzy clustering approach based on geodesic distance and angular distance in the dual graph space, then optimized chart boundaries using graph cutting techniques. All these mesh segmentation approaches utilize the connectivity information for chart generation or face clustering.

Chart Parameterization. There are several schemes for flattening a surface chart as isometrically as possible. Maillot *et al.*[15] unfolded each chart to form a texture atlas by optimizing edge springs of non-zero rest length. In 2001, Sander *et al.*[16] proposed a parameterization approach which minimizes both texture stretch and texture deviation between levels of details. In 2002, they[33] further proposed a mesh parameterization scheme specialized to the geometric signal by minimizing the signal approximation error. Based on a least squares approximation of the Cauchy-Riemann equations for conformal maps, Levy *et al.*[1] proposed a new parameterization method — least squares conformal map. Desbrun *et al.*[34] proposed a similar parameterization method which minimizes the distortion of different intrinsic measures of the original mesh.

For point-sampled geometry, research on efficient surface parameterization is comparatively rare. Floater and Reimers[21] introduced meshless parameterization for a single point-sampled patch. They mapped the sample points into a planar parameter domain, one to one, by solving a sparse linear system. In the area of reverse engineering, Barhak and Fischer[35] proposed respectively partial differential equation parameterization and neural network self organizing maps parameterization for irregular and scattered 3D points. The partial differential equation scheme can avoid self-intersection of the parametric grid, whilst the self organizing maps approach supports a uniform and smooth reconstruction of the surface. Zwicker *et al.*[2] described a parameterization algorithm for point-sampled surfaces which is similar to Levy's method for polygonal meshes[1]. They solved the linear least squares problem efficiently by hierarchical clustering points. Zigelman *et al.*[3] presented a surface flattening method based on a multidimensional scaling approach. These algorithms tend to introduce angle or distance distortions when flattening the entire surface patch onto a planar domain. Moreover, their methods are not efficient enough for large scale models.

k-Means Clustering Algorithm (*k-MC*). Statistical clustering algorithms are applied in many applications, such as pattern classification, pattern recognition, data compression, and vector quantization, etc. Among statistical clustering algorithms, k-means clustering is the most widely used approach. It can be described as follows: given a set of n data points in $\boldsymbol{R}^d$, determine a set of k centers so as to minimize the mean squared distance from each data point to its nearest center. The popular heuristic for solving the k-means clustering problem is based on a simple iterative scheme for finding a locally minimal solution, namely Lloyd's algorithm[36,37].

Multi-dimensional Scaling (*MDS*). Multi Dimensional scaling can be considered as an alternative to factor analysis. It is related to the point representation of objects. MDS maximizes the fit between the proximity measure of each pair of objects and the distances between all of them in the map. It is a statistical technique used to uncover the geometric similarities of datasets.

Tenenbaum *et al.*[38] proposed a nonlinear dimensionality reduction technique — IsoMap based on classical MDS. It preserves the intrinsic geometry of the data, i.e., the geodesic manifold distances between all pairs of data points. Given a set of high-dimensional points, the method computes the geodesic distances between each pair of points on a manifold, then applies MDS to these geodesic distances to find a set of points embedded in low-dimensional space with similar pairwise distances. A similar approach can be extended to mesh parameterization by reducing data dimensionality to two, i.e., 2-dimensional planar data[3].

3 Estimation of Local Surface Differentials at Sample Points

3.1 Analyzing the Local Surface Differentials

For a given direction $\boldsymbol{T}$ in the tangent plane Π at sample point $\boldsymbol{p}$ on a *regular surface* S, a unique normal curve Γ can be defined as the intersection between the surface and the normal plane corresponding to the tangent direction $\boldsymbol{T}$. The normal curvature κ_n of the planar normal curve is completely determined by the tangent direction $\boldsymbol{T}$ at point $\boldsymbol{p}$. The maximum and minimum normal curvatures κ_1 and κ_2 are called the *principal curvatures*. The tangent directions corresponding to the two principal curvatures are called the *principal curvature directions* $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$, respectively.

The local surface differentials $\{\Pi, \boldsymbol{n}, \kappa_1, \kappa_2, \boldsymbol{e}_1, \boldsymbol{e}_2, \kappa_n\}$ for point $\boldsymbol{p}$ on a regular surface S have the following relationships[14]:

- the principal curvature directions $\boldsymbol{e}_1, \boldsymbol{e}_2$, and surface normal $\boldsymbol{n}$ form an orthogonal local frame at point $\boldsymbol{p}$;
- for any tangent $\boldsymbol{T} = t_1\boldsymbol{e}_1 + t_2\boldsymbol{e}_2$, the corresponding normal curvature can be calculated as $\kappa_n(\boldsymbol{T}) = \kappa_1 t_1^2 + \kappa_2 t_2^2$;
- Gaussian curvature K and mean curvature H are defined as: $K = \kappa_1\kappa_2$, $H = (\kappa_1 + \kappa_2)/2$, which are the intrinsic surface geometric invariants. They reflect the surface normal variation at the sample point $\boldsymbol{p}$.

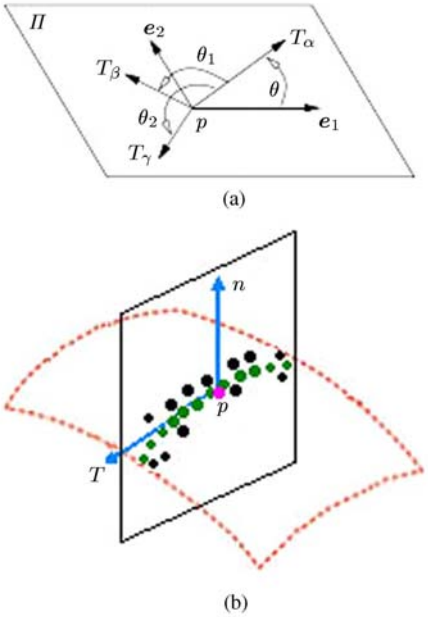

Fig.1. (a) Analysis for principal directions and principal curvatures in the tangent plane. (b) Projecting neighboring points onto the normal plane.

Our projection algorithm of estimating the surface differentials at the regular surface point $\boldsymbol{p}$ are derived from the theory of differential geometry and the following fact[39].

Let the tangent plane at point $\boldsymbol{p}$ be Π, the angle from a chosen tangent direction $\boldsymbol{T}_\alpha$ to the principal direction $\boldsymbol{e}_1$ be θ and the angle from $\boldsymbol{T}_\alpha$ to $\boldsymbol{T}_\beta$ and $\boldsymbol{T}_\gamma$ be θ_1 and θ_2, respectively (see Fig.1(a)).

The normal curvatures for the direction $\boldsymbol{T}_\alpha$, $\boldsymbol{T}_\beta$ and $\boldsymbol{T}_\gamma$ can be expressed in terms of the two principal curvatures, respectively:

$$\begin{cases} \kappa_\alpha = \kappa_1 \cos^2\theta + \kappa_2 \sin^2\theta, \\ \kappa_\beta = \kappa_1 \cos^2(\theta+\theta_1) + \kappa_2 \sin^2(\theta+\theta_1), \\ \kappa_\gamma = \kappa_1 \cos^2(\theta+\theta_2) + \kappa_2 \sin^2(\theta+\theta_2). \end{cases} \quad (1)$$

Thus, the relation between the angle of tangent direction and the corresponding normal curvature can be deduced.

$$\tan(2\theta+\theta_2) = \frac{\sin(\theta_1-\theta_2)}{\dfrac{\kappa_\alpha-\kappa_\beta}{\kappa_\alpha-\kappa_\gamma} \cdot \dfrac{\sin\theta_2}{\sin\theta_1} - \cos(\theta_1-\theta_2)}. \quad (2)$$

So, with the three normal curvatures $\kappa_\alpha, \kappa_\beta, \kappa_\gamma$ regarding to three sampled directions and corresponding θ_1, θ_2, the angle θ and the principal directions can be determined from (2). Furthermore, substituting θ and θ_1, θ_2 into the linear system (1), the principal curvatures κ_1 and κ_2 can be calculated.

3.2 Calculating Normal Curvature by Curve Fitting Scheme

To calculate the discrete curvature of unorganized point clouds, we can adopt the traditional curve fitting scheme. By taking the sample point $\boldsymbol{p}$ as the origin of the local coordinate system, and the normal $\boldsymbol{n}$ as the y axis, the projection points can be represented as some $(x; y)$ pairs. We then fit an n-degree polynomial to the resulting $(x; y)$ pairs. For efficiency, we choose quadratic polynomial as the fitting curve due to the fact that quadrics are the lowest degree polynomials that can locally approximate all cases of normal curvatures, such as positive curvature for convex curves, negative curvature for concave curves, etc.

To satisfy the constraint that the polynomial curve passes through the origin, we take the constant term of the polynomial as zero, and the fitting polynomial can be represented in the form

$$y = ax + bx^2.$$

Then, a simple linear system can be obtained by least squares the fitting error and solved efficiently. The relative curvature at the origin of the planar quadratic curve is finally estimated as

$$\kappa_r = \frac{-2b}{(1+a^2)^{\frac{3}{2}}}$$

which approximated the normal curvature for the given tangent direction.

3.3 Estimating Local Surface Differentials by Projection Method

Our algorithm takes as input a set of unstructured *surfels* rather than a discrete point cloud. Normals are often available, and the input *surfels* can be represented as clouds of point-direction pairs $\{(\boldsymbol{p}_i, \boldsymbol{n}_i)\}$. Based on the classical differential geometry and the analysis for principal directions and principal curvatures in the tangent plane, our projection algorithm of estimating local surface differentials at a regular point $\boldsymbol{p}_i$ consists of the following four steps:

- For each regular point $\boldsymbol{p}_i$, its neighborhood is determined adaptively so as to keep the local sampling density $\rho = k/r^2$ as a constant, where r is the radius of the enclosing sphere of k-nearest neighboring sample points.
- According to the normal direction $\boldsymbol{n}_i$ for sample point $\boldsymbol{p}_i$, the tangent plane Π can easily be obtained. On the tangent plane Π, three different tangent directions $\boldsymbol{T}_\alpha, \boldsymbol{T}_\beta, \boldsymbol{T}_\gamma$ are sampled. For each sampled tangent direction, a normal plane is defined by the surface normal and the tangent direction.
- All adaptively selected neighboring points to $\boldsymbol{p}_i$ are then projected onto this normal plane (see Fig.1(b)). The normal curvature can be approximated as the discrete curvature of these planar discrete projection points on the normal plane, and can be calculated through curve fitting scheme (see Subsection 3.2).
- Based on the three estimated normal curvatures $\kappa_\alpha, \kappa_\beta, \kappa_\gamma$ for three tangent directions $\boldsymbol{T}_\alpha, \boldsymbol{T}_\beta, \boldsymbol{T}_\gamma$, respectively, the principal directions and corresponding principal curvatures can be determined as the above closed form (see Subsection 3.1).

Finally, Gaussian curvature and mean curvature at each sample point are straightforward results with the above estimations.

Table 1. Comparison of Estimation Times for Our Projection Method and the Integral Eigenvalue Method (IEM) for Different Models

Models	#Points	Neighbors	Timings for Differential Estimations (s)	
			Our Method	IEM Method
Bunny	35 283	7/27	0.53	0.87
Horse	48 484	3/99	1.27	3.03
Rabbit	67 038	11/40	1.15	2.38
Fandisk	103 570	6/84	1.57	3.02
Venus	134 345	9/50	2.25	4.58

Notice: The timing is measured on a PC with a Pentium IV 2.0GHz CPU, 512MB memory.

The surface normal gives us first order information of the surface configuration around the sample point $\boldsymbol{p}_i$, while the principal curvature directions and the principal curvatures give us second order information. Unlike other schemes, our projection algorithm can estimate local surface differentials according to the normal curvatures for only three tangent directions, which makes it an efficient method for estimating differentials of large-scale point-sampled surfaces. The above algorithm was implemented and tested on a PC with a Pentium IV 2.0GHz CPU, 512MB memory and Windows XP. Table 1 compares the estimation timings for surface differentials by our projection method and the integral eigenvalue method (IEM), which demonstrates the efficiency of our scheme. For example, for the horse model (it has totally 48 484 sample points and the size of neighborhood is about 3/99, i.e., ranged adaptively from 3 to 99 (not including itself)), the timing for estimating surface differentials is about 1.27s for our projection method while the timing for the integral eigenvalue method is about 3.03s, etc. Examples of local differential estimations for the fandisk model and the horse model are illustrated in Figs.2 and 3, respectively. For the sake of visualization, all figures for differential estimations are colored by the magnitude of the corresponding curvature tensors (Hermosa pink means high curvature zone and black means low curvature zone, and so on). For the horse model, we compare the performance for curvature estimations by two different methods, which demonstrate the robust and accuracy of our projection scheme. Furthermore, the surface principal directions at sample points should be obtained. Now they can be determined easily from our projection approach (see Fig.4). They are fundamental in the next section for segmenting the point-sampled model with the criterion of the angular difference between surface differential directions.

4 Model Segmentation Using k-Means Clustering Algorithm

Given a point-sampled surface S, the goal of segmentation is to decompose the surface model S into k disjoint parts $S_1, S_2, \ldots, S_k$, whose union is S.

In this section, a new model segmentation approach based on statistical analysis is proposed. It provides an efficient k-means clustering solution for clustering a set of discrete point clouds[36].

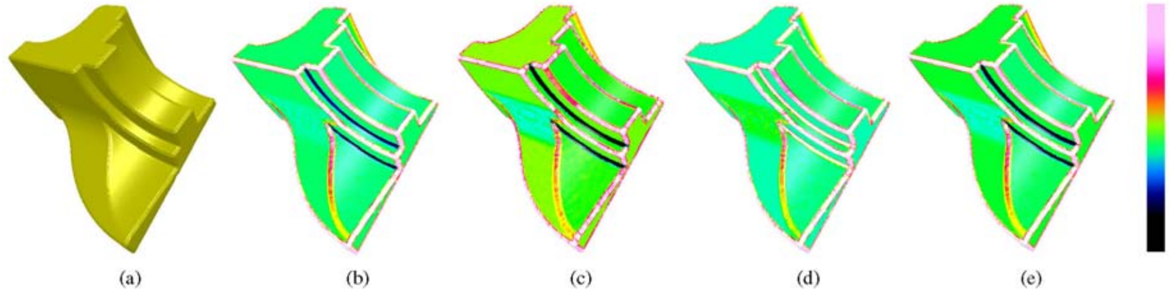

Fig.2. Local surface differential estimations for the fandisk model by our projection method. (a) Original fandisk model. (b) Color-coded κ_1-curvature estimation. (c) κ_2-curvature estimation. (d) Color-coded Gaussian curvature estimation. (e) Mean curvature estimation.

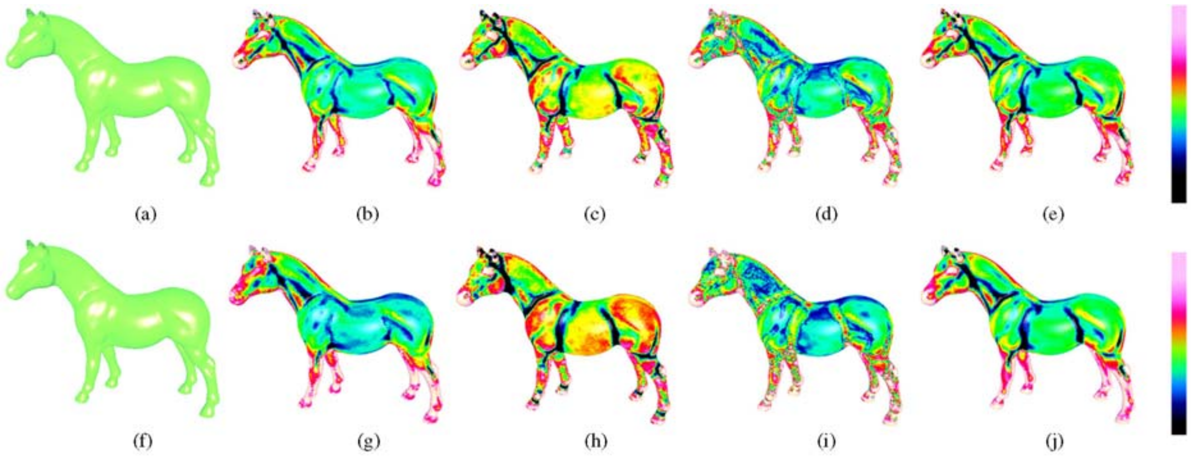

Fig.3. Comparison of different methods for local surface differential estimations. Upper row is differential estimations by our projection method: (a) Original horse model. (b) Color-coded κ_1-curvature estimation. (c) κ_2-curvature estimation. (d) Color-coded Gaussian curvature estimation. (e) Mean curvature estimation. Lower row is the corresponding differential estimations by the integral eigenvalue method.

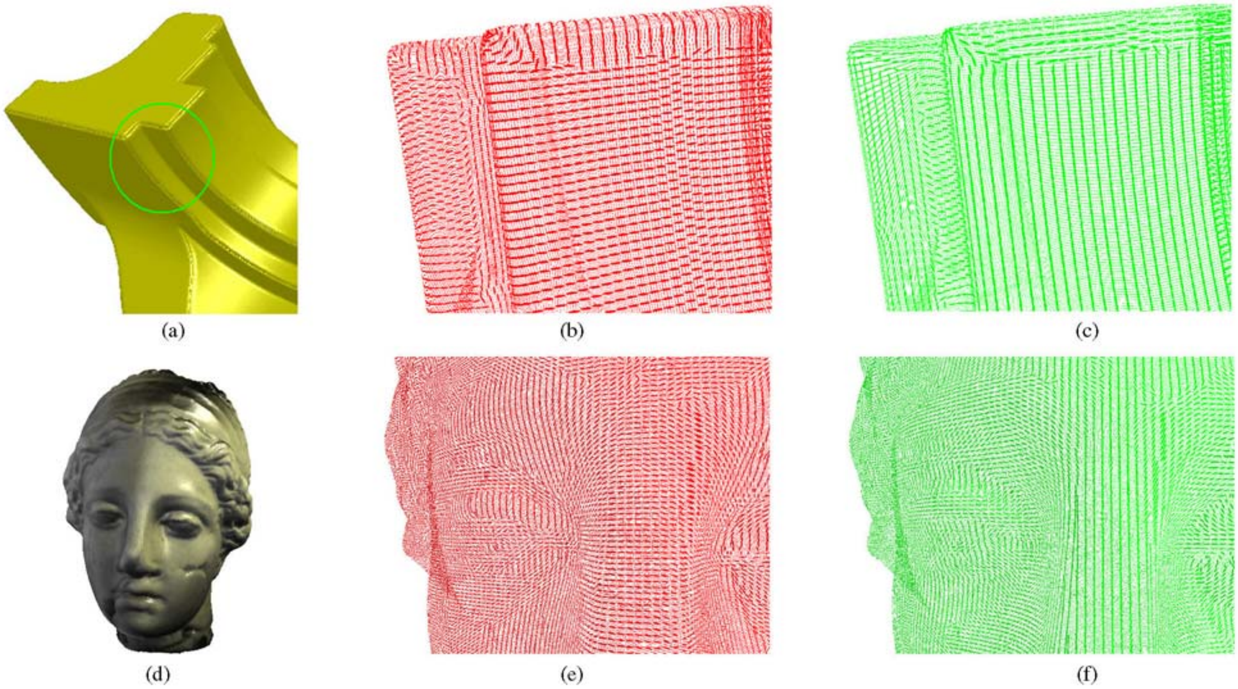

Fig.4. Principal curvature direction estimations by our projection method. (a) Original fandisk model. (b) κ_1-principal directions for a corner of fandisk model. (c) κ_2-principal directions for a corner of fandisk model. (d) Original venus model. (e) κ_1-principal directions for face of venus model. (f) κ_2-principal directions for face of venus model.

Our clustering algorithm is based on the underlying assumption that two sample points, distant in terms of both Euclidean distance and angular difference between local differential directions, are less likely to be in the same cluster than points which are close. This assumption can create compact and planar surface charts, so that we can achieve low distortions in the subsequent parameterization procedure.

We define the angular difference between two sample points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ as follows:

$$ang\_Diff(\boldsymbol{p}_i, \boldsymbol{p}_j) = 1 - \cos^2 \alpha_{ij}$$

where α_{ij} is the angle between local differential directions at $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$. The local differential direction can be the normal or principal curvature directions. A large *ang_Diff* means a big angular deviation between local differential directions and vice versa.

The proposed segmentation algorithm aims at partitioning point-sampled surfaces into k disjoint sub-patches. Each sub-patch S_i is represented by its center $\boldsymbol{C}_i$. The local differential directions for center $\boldsymbol{C}_i$ are calculated as:

$$direct(\boldsymbol{C}_i) = \sum_{\boldsymbol{p}_j \in S_i} \theta(\boldsymbol{p}_j, \boldsymbol{C}_i) direct(\boldsymbol{p}_j)$$

where $\theta(\cdot,\cdot)$ denotes a Gaussian weighting or a normalized Gaussian weighting function. The objective for clustering is to minimize the distance from each sample point to its closest center, such as:

$$\min F$$

where:

$$F = \lambda \sum_{\boldsymbol{C}_i} \sum_{\boldsymbol{p}_j \in S_i} Euclid\_Dist^2(\boldsymbol{p}_j, \boldsymbol{C}_i)/num + (1-\lambda) \sum_{\boldsymbol{C}_i} \sum_{\boldsymbol{p}_j \in S_i} ang\_Diff(\boldsymbol{p}_j, \boldsymbol{C}_i)/num.$$

The first term measures the average squared Euclidean distance between all sample points $\boldsymbol{p}_i$ over all sub-patches and their corresponding centers. Minimizing the Euclidean distance ensures that the physical distances between each pair of sample points on the same chart are small, thus making each chart compact. However, taking account of only the first term would include sample points with big angular differences but close in physical distances in the same chart. Thus, the planarity of each chart cannot be guaranteed.

The second term measures the average angular difference between all $\boldsymbol{p}_i$ over all sub-patches and their corresponding centers. Minimizing the average angular difference ensures that the angular deviation between all differential directions of sample points on the same chart are small, thus making each chart planar.

We trade off the compactness and planarity of each chart by using a weighting parameter λ. $\lambda = 1$ means clustering the model by Euclidean distance only, whilst $\lambda = 0$ means by angular difference only. Varying the weight will affect the clustering result, see Figs.5, 6 and 7.

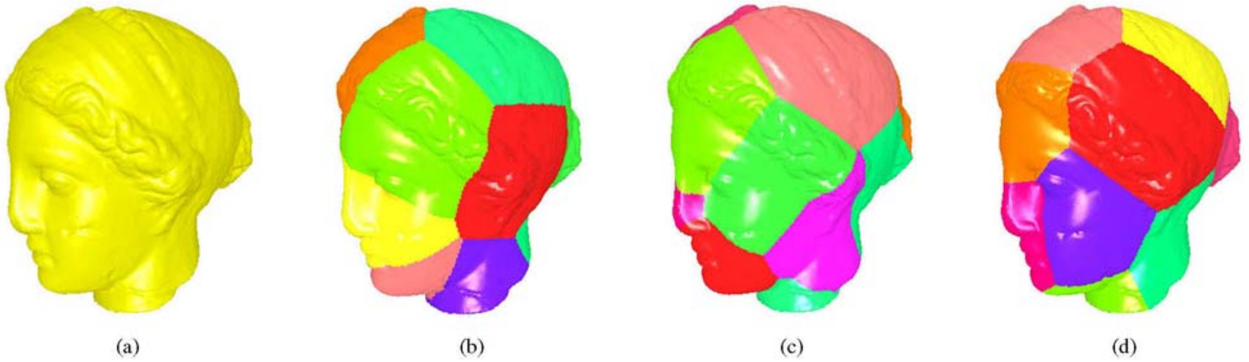

Fig.5. Segmentation examples using the k-means clustering algorithm by weighting Euclidean distance and normal direction. (a) Original venus model. (b) Model segmentation by Euclidean distance only ($\lambda = 1.0$). (c) Model segmentation by $\lambda = 0.3$. (d) Model segmentation by normal direction only ($\lambda = 0.0$).

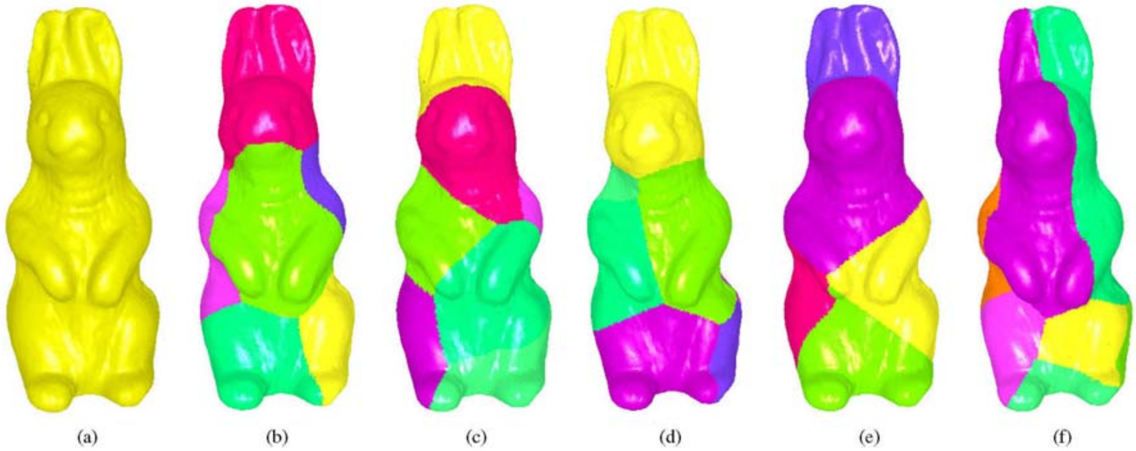

Fig.6. Segmentation examples using the k-means clustering algorithm by weighting Euclidean distance and first principal curvature direction. (a) Original rabbit model. (b) Model segmentation by Euclidean distance only ($\lambda = 1.0$). (c), (d), (e) Model segmentation by $\lambda = 0.8$, $\lambda = 0.5$ and $\lambda = 0.3$, respectively. (f) Model segmentation by first principal curvature direction only ($\lambda = 0.0$).

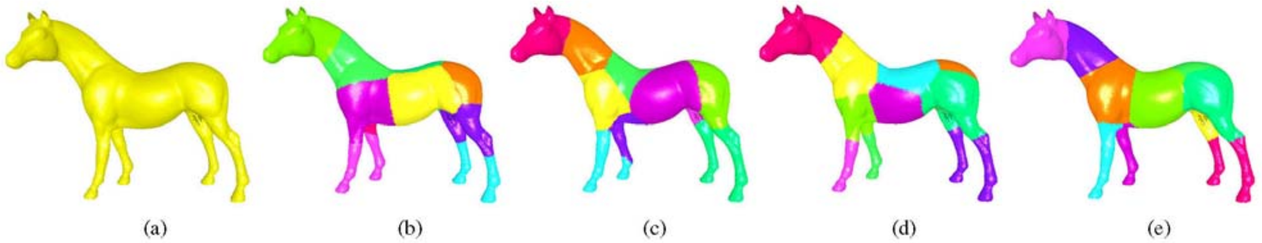

Fig.7. Segmentation examples using the k-means clustering algorithm by weighting Euclidean distance and second principal curvature direction. (a) Original horse model. (b) Model segmentation by Euclidean distance only ($\lambda = 1.0$). (c), (d), (e) Model segmentation by $\lambda = 0.8$ and $\lambda = 0.4$, respectively. (f) Model segmentation by second principal curvature direction only ($\lambda = 0.0$).

Unfortunately, there is no efficient solution known to this optimal problem because it is an NP-hard problem[37]. One of the popular heuristics for solving the problem is based on an iterative scheme for finding a local minimal solution. We refer to this as the k-means clustering algorithm or generalized Lloyd's algorithm.

The idea of the k-means clustering algorithm for point-sampled geometry segmentation is as follows: firstly, k initial centers are selected randomly. For the sake of efficiency, we choose k centers so as to maximize the Euclidean distances between them. Secondly, for each sample point, the distances F to all centers are calculated. Using these distances, all of the discrete points are partitioned into k clusters by assigning each point to its nearest center. Thirdly, the algorithm updates the centers to be the centroids of their associated clusters before starting a new partition with these new centers. The second and third steps are iterated until convergence or until some predefined number of iterations is performed. Finally, the whole model is decomposed into k disjoint charts according to these final clusters.

However, due to the expensive cost of determining the nearest centers, a naive implementation of the generalized Lloyd's algorithm will be time-consuming. To improve the efficiency of the proposed clustering algorithm, we organize the discrete point clouds in a Kd-tree and adopt a filtering algorithm[37] for computing the nearest center to each sample point in each stage of Lloyd's algorithm. We associate each node of the Kd-tree with a subset of candidate centers. The candidate centers are selected which might serve as the nearest neighbors for some sample points lying within the associated bounding box. For the root of tree, the candidate centers consist of all k centers. While traversing to the node's children, the special center $\boldsymbol{C}^*$ closest to the midpoint of the bounding box is found, and if any sample points in the bounding box are closer to the $\boldsymbol{C}^*$ than they are to candidate center $\boldsymbol{C}$, the candidate center $\boldsymbol{C}$ for the children node is pruned or filtered. For each leaf node of the Kd-tree, we assign each sample point to its single candidate or to its nearest candidate center.

In our experiments, the user can adjust three issues to control the segmentation results. The first parameter is the chart number k for every point-sampled model. The chart number k is adaptively chosen depending on the number of sample points and the shape complexity of the sampled geometry. For each chart, the number of sample points should not exceed a given threshold so as to ensure the efficiency of succeeding chart parameterization, and the angular difference of sample points should not be large so as to minimize the distance distortion of chart parameterization. The second issue is the criterion selection of the clustering algorithm, for example, clustering by weighting Euclidean distance and normal direction or by weighting Euclidean distance and principal curvature directions. By weighting Euclidean distance and normal direction, we segment the venus model into 12 charts (Fig.5). For rabbit and horse model, by weighting Euclidean distance and principal curvature directions, we segment the rabbit model into 8 charts (Fig.6) and the horse model into 10 charts (Fig.7), respectively. The final parameter is the weighting parameter λ in the objective function F, which trade off the compactness and planarity of each chart. Both Figs. 6 and 7 show that a large value of λ emphasizes the compactness of the model chart, whilst a small λ emphasizes the planarity and segments the model according to principal curvature directions. We can choose these proper parameters for model segmentation according to the specific application and the underlying geometric model.

5 Chart Parameterization Using Multidimensional Scaling

Given a chart composed of discrete unorganized point clouds $S_k = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N\}$, we want to flatten it so as to establish a chart parameterization. As a preprocess, the geodesic distance between each point pair is approximated by the shortest path distance, which can be calculated based on the following graph-based approach: first, a neighborhood graph is constructed by connecting a given point to its nearest neighbors, with link weights equal to the Euclidean distances between the points; then the shortest path distances are computed between all point pairs in the constructed graph using Dijkstra's or Floyd's algorithm.

Our algorithm for chart parameterization is based upon the IsoMap (isometric feature mapping) dimensionality reduction technique[38], which is termed multidimensional scaling (MDS) in the context of statistics. The approach first builds a squared geodesic distance matrix $\boldsymbol{M}$. Elements of matrix $\boldsymbol{M}$ are the squared geodesic distances between sample points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ $(i, j = 1, 2, \ldots, N)$. The chart parameterization is equivalent to constructing an embedding of the parameterized coordinates in a 2-dimensional plane, such that for each pair of sample points the error between corresponding squared Euclidean distance in the plane and squared geodesic distance in the underlying surface chart is as small as possible. The MDS method can optimally preserve the intrinsic geometry of the original surface chart[3,38]. Our idea of MDS is to approximate the matrix $\boldsymbol{M}$ by a squared Euclidean distance matrix $\boldsymbol{E}$, which is constructed as follows:

- determine the $N \times N$ symmetric matrix $\boldsymbol{M}$ by evaluating the squared geodesic distances between each pair of sample points;
- apply double centering and normalization to $\boldsymbol{M}$, i.e., $\boldsymbol{B} = -\dfrac{1}{2}\boldsymbol{JMJ}$, where $\boldsymbol{J}$ is an $N \times N$ centering matrix defined by $\boldsymbol{J} = \boldsymbol{I} - \dfrac{1}{N}\boldsymbol{1}\boldsymbol{1}^T$, $\boldsymbol{I}$ is the $N \times N$ identity matrix, and $\boldsymbol{1}$ is a vector of ones of length N. This is used to restrict the barycenter of the set of pairwise distances to lie at the origin;
- compute the eigenvalues $\{\lambda_i\}$ (in decreasing order) and their corresponding eigenvectors $\{\nu_i\}$ of $\boldsymbol{B}$ $(i = 1, 2, \ldots, N)$, let ν_i^j be the j-th component of the i-th eigenvector;
- finally, the first and the second eigenvectors represent the parameterization information of correspond-

ing sample points. So, for each sample point $\boldsymbol{p}_j$ of the original surface, let $\bar{\boldsymbol{p}}_j$ be its corresponding point in the parameter plane. Then its k-th coordinate is:

$$\bar{\boldsymbol{p}}_j^k = \sqrt{\lambda_k}\nu_k^j, \quad (k = 1, 2; j = 1, 2, \ldots, N).$$

For the spectral decomposition of the matrix $\boldsymbol{B}$, i.e., in our case, to find the two largest eigenvalues and their corresponding eigenvectors, we use the power method[40] to accelerate this procedure due to its $O(2N^2)$ computational complexity. However, in order to efficiently parameterize each chart, the chart size should not exceed a given threshold, which can be achieved by adaptively choosing the chart number k depending on the sampled geometry as mentioned in Section 4. In our experiments, the threshold of the chart size is 3000 sample points.

The distance distortion regarding the parameterization of one chart can be measured as a weighted average of the distance distortion ($wAvDist\_D$) over all sample points:

$$wAvDist\_D = \frac{1}{\sum\limits_k N_k} \sum_i N_i Dist\_D(i)$$

where N_i denotes the number of selected neighboring points for sample point $\boldsymbol{p}_i$, and $Dist\_D(i)$ denotes distance distortion of sample point $\boldsymbol{p}_i$ under parameterization, which can be calculated as the root-mean-square stretch of neighboring points $\boldsymbol{p}_j$ ($j = 1, 2, \ldots, N_i$) for sample point $\boldsymbol{p}_i$:

$$\sqrt{\frac{\max\left(\dfrac{d_{\text{geo}}(\boldsymbol{p}_i, \boldsymbol{p}_j)}{d_{\text{param}}(\bar{\boldsymbol{p}}_i, \bar{\boldsymbol{p}}_j)}\right)^2 + \min\left(\dfrac{d_{\text{geo}}(\boldsymbol{p}_i, \boldsymbol{p}_j)}{d_{\text{param}}(\bar{\boldsymbol{p}}_i, \bar{\boldsymbol{p}}_j)}\right)^2}{2}}$$

where $d_{\text{param}}(\bar{\boldsymbol{p}}_i, \bar{\boldsymbol{p}}_j)$ represents the Euclidean distance between the parametric coordinates $\bar{\boldsymbol{p}}_i$ and $\bar{\boldsymbol{p}}_j$, and $d_{\text{geo}}(\boldsymbol{p}_i, \boldsymbol{p}_j)$ represents the geodesic distance between points $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$.

The examples of parameterizing one chart are shown in Figs. 8 and 9. In Floater's meshless parameterization, we use a natural ordered boundary[21] for minimizing the distance distortion. Some comparisons between our MDS and meshless parameterization algorithm are given, which indicate that the distance distortion of our MDS approach is always small.

Finally, we pack all parameterized charts into a rectangular texture domain by adopting a rectangle packing algorithm[16]. The example of the bunny model parameterization is shown in Fig.10.

6 Conclusions

In this paper, a differentials-based segmentation and parameterization approach for point-sampled surfaces is proposed, which consists of three parts. Based on the analysis of surface normal curvature and estimated by curve fitting scheme, a new projection scheme is employed to estimate local surface differentials, including principal curvatures and principal curvature directions. An efficient k-means clustering algorithm is then proposed for partitioning the model into a set of charts. Finally, the multidimensional scaling method is used for chart parameterization, a special nonlinear dimensionality reduction technique which minimizes distortion of distances between all pairs of data points. Compared with other parameterization approaches, our method partitions the point-sampled surface based on its local geometric features, hence producing better parameterization results.

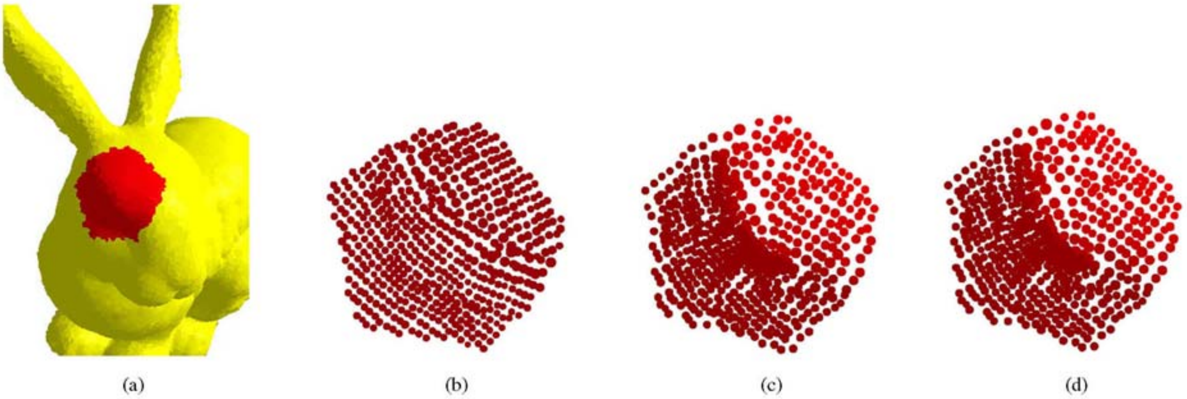

Fig.8. Parameterization one chart of the bunny model. (a) Selected chart on bunny model. (b) Parameterization using our parameterization approach (distance distortion is 1.1112). (c) Parameterization by Floater's uniform parameterization approach (distance distortion is 1.2156). (d) Parameterization by Floater's reciprocal distance parameterization approach (distance distortion is 1.1652).

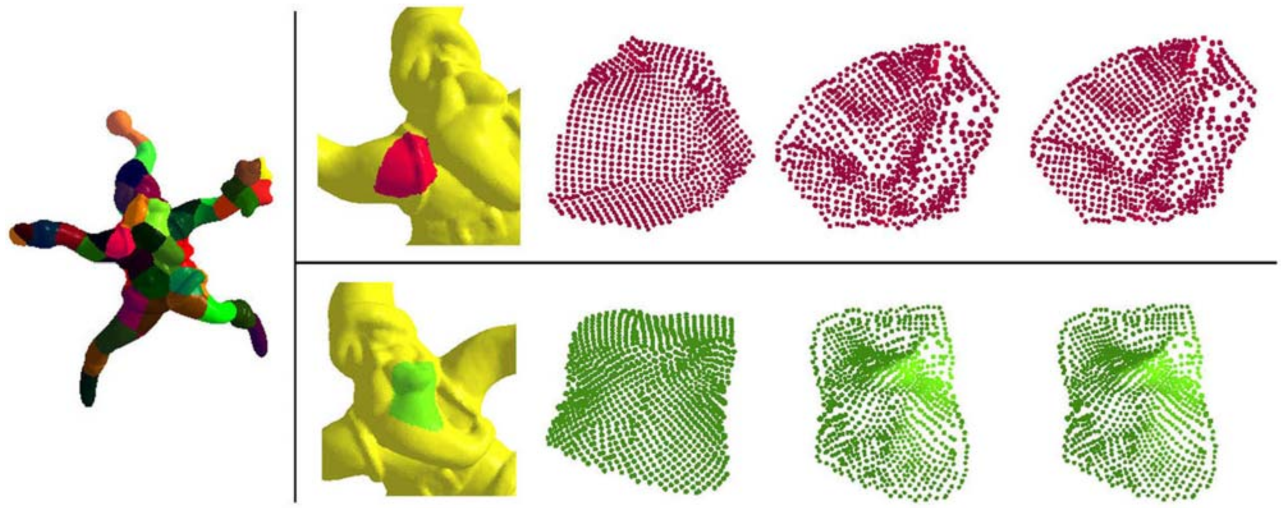

Fig.9. Parameterization of two charts of the Santa model. Left: model segmentation using the clustering algorithm. Top Right: first chart on the model, and parameterization using our parameterization approach (distance distortion is 1.0723), Floater's uniform parameterization approach (distance distortion is 1.1674) and Floater's reciprocal distance parameterization approach (distance distortion is 1.1170), respectively. Down Right: second chart on the model, and parameterization using our parameterization approach (distance distortion is 1.1263), Floater's uniform parameterization approach (distance distortion is 1.2395) and Floater's reciprocal distance parameterization approach (distance distortion is 1.1809), respectively.

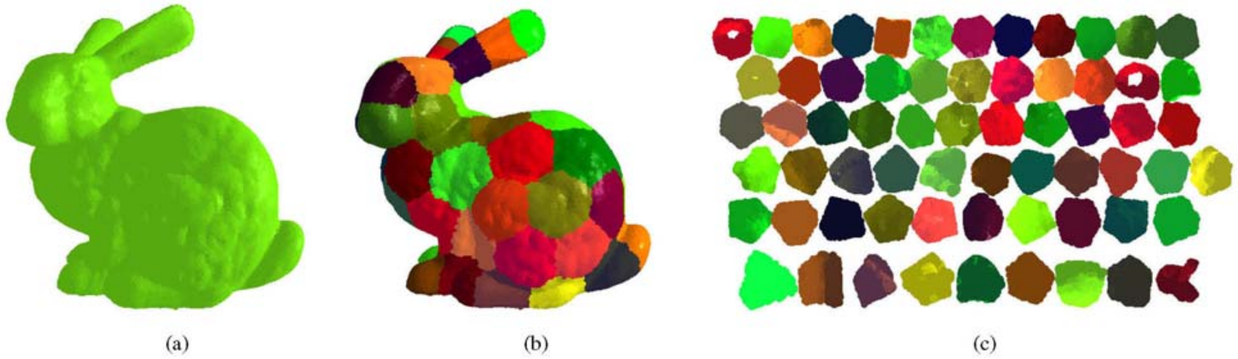

Fig.10. Forming a texture atlas for the bunny model. (a) Original bunny model. (b) Model segmentation based on first principal curvature direction. (c) Chart parameterization to form the texture atlas.

With the proposed piecewise parameterization method, future research should focus on geometry processing of point-sampled objects, such as morphing, resampling, editing, compression, etc.

References

[1] Levy B, Petitjean S, Ray N, Maillot J. Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics*, 2002, 21(3): 362∼371.

[2] Zwicker M, Pauly M, Knoll O, Gross M. Pointshop 3D: An interactive system for point-based surface editing. *ACM Transactions on Graphics*, 2002, 21(3): 322∼329.

[3] Zigelman G, Kimmel R, Kiryati N. Texture mapping using surface flattening via multidimensional scaling. *IEEE Transactions on Visualization and Computer Graphics*, 2002, 8(2): 198∼207.

[4] Biermann H, Martin I, Bernardini F, Zorin D. Cut-and-paste editing of multiresolution surfaces. *ACM Transactions on Graphics*, 2002, 21(3): 312∼321.

[5] Pauly M, Keiser R, Kobbelt L, Gross M. Shape modeling with point-sampled geometry. *ACM Transactions on Graphics*, 2003, 22(3): 641∼650.

[6] Liu X G, Bao H J, Peng Q S. Digital differential geometry processing. *Journal of Computer Science and Technology*, 2006, 21(5): 847∼860.

[7] Eck M, DeRose T, Duchamp T *et al.* Multiresolution analysis of arbitrary meshes. In *Proc. ACM SIGGRAPH'95*, Los Angeles, California, 1995, pp.173∼182.

[8] Lee A, Sweldens W, Schröder P *et al.* MAPS: Multiresolution adaptive parametrization of surfaces. In *Proc. ACM SIGGRAPH'98*, Orlando, Florida, 1998, pp. 95∼104.

[9] Liu Y J, Tang K, Yuen M M. Multiresolution free form object modeling with point sampled geometry. *Journal of Computer Science and Technology*, 2004, 19(5): 607∼617.

[10] Khodakovsky A, Schröder P, Sweldens W. Progressive geometry compression. In *Proc. ACM SIGGRAPH'00*, New

Orleans, Louisiana, 2000, pp.271∼278.

[11] Waschbüsch M, Würmlin S, Lamboray E *et al.* Progressive compression of point-sampled models. In *Proc. the Eurographics Symposium on Point-Based Graphics'04*, Zurich, Switzerland, 2004, pp.95∼102.

[12] Zhou K, Bao H J, Shi J Y, Peng Q S. Geometric signal compression. *Journal of Computer Science and Technology*, 2004, 19(5): 596∼606.

[13] Floater M, Hormann K. Surface Parameterization: A Tutorial and Survey. Advances in Multiresolution for Geometric Modelling, Dodgson N, Floater M, Sabin M (eds.), Heidelberg: Springer-Verlag, 2005, pp.157∼186.

[14] doCarmo M P. Differential Geometry of Curves and Surfaces. Englewood Cliffs: Prentice-Hall, New Jersey, 1976.

[15] Maillot J, Yahia H, Verroust A. Interactive Texture Mapping. In *Proc. ACM SIGGRAPH'93*, Anaheim, California, 1993, pp.27∼34.

[16] Sander P, Snyder J, Gortler S, Hoppe H. Texture mapping progressive meshes. In *Proc. ACM SIGGRAPH'01*, Los Angeles, California, 2001, pp.409∼416.

[17] Sorkine O, Cohen-or D, Goldenthal R, Lischinski D. Bounded-distortion piecewise mesh parameterization. In *Proc. IEEE Visualization'02*, Boston, Massachusetts, 2002, pp.355∼362.

[18] Sander P, Wood Z, Gortler S *et al.* Multi-chart geometry images. In *Proc. the Eurographics Symposium on Geometry Processing'03*, Aachen, Germany, 2003, pp.146∼155.

[19] Zhou K, Snyder J, Guo B, Shum H-Y. Iso-charts: Stretch-driven mesh parameterization using spectral analysis. In *Proc. the Eurographics Symposium on Geometry Processing'04*, Nice, France, 2004, pp.47∼56.

[20] Zhou K, Wang X, Tong Y *et al.* TextureMontage: seamless texturing of arbitrary surfaces from multiple images. *ACM Transactions on Graphics*, 2005, 24(3): 1148∼1155.

[21] Floater M, Reimers M. Meshless parameterization and surface reconstruction. *Computer Aided Geometric Design*, 2001, 18(2): 77∼92.

[22] Zwicker M, Gotsman C. Meshing point clouds using spherical parameterization. In *Proc. the Eurographics Symposium on Point-Based Graphics'04*, Zurich, Switzerland, 2004, pp.173∼180.

[23] Kalaiah A, Varshney A. Differential point rendering. In *Proc. 12th Eurographics Workshop on Rendering*, London, UK, 2001, pp.139∼150.

[24] Pauly M, Gross M, Kobbelt L. Efficient simplification of point-sampled surfaces. In *Proc. IEEE Visualization'02*, Boston, Massachusetts, 2002, pp.163∼170.

[25] Adamson A, Alexa M. Anisotropic point set surfaces. In *Proc. Afrigraph'06*, Cape Town, South Africa, 2006, pp.7∼13.

[26] Levin D. Mesh-Independent Surface Interpolation. *Geometric Modeling for Scientific Visualization*, Brunnett G, Hamann B, Mueller K, Linsen L (eds.), Heidelberg: Springer-Verlag, 2003, pp.37∼49.

[27] Alexa M, Adamson A. On normals and projection operators for surfaces defined by point sets. In *Proc. the Eurographics Symposium on Point-Based Graphics'04*, Zurich, Switzerland, 2004, pp.149∼155.

[28] Taubin G. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *Proc. 5th International Conference on Computer Vision'95*, Boston, Massachusetts, 1995, pp.902∼907.

[29] Lange C, Polthier K. Anisotropic smoothing of point sets. *Computer Aided Geometric Design*, 2005, 22(7): 680∼692.

[30] Krishnamurthy V, Levoy M. Fitting smooth surfaces to dense polygon meshes. In *Proc. ACM SIGGRAPH'96*, New Orleans, Louisiana, 1996, pp.313∼324.

[31] Yamauchi H, Gumhold S, Zayer R, Seidel H-P. Mesh segmentation driven by Gaussian curvature. *The Visual Computer*, 2005, 21(8∼10): 659∼668.

[32] Katz S, Tal A. Hierachical mesh decomposition using fuzzy clustering and cuts. *ACM Transactions on Graphics*, 2003, 22(3): 954∼961.

[33] Sander P, Gortler S, Snyder J, Hoppe H. Signal-specialized parameterization. In *Proc. 13th Eurographics Workshop on Rendering*, Pisa, Italy, 2002, pp.87∼100.

[34] Desbrun M, Meyer M, Alliez P. Intrinsic parameterizations of surface meshes. *Computer Graphics Forum*, 2002, 21(3): 209∼218.

[35] Barhak J, Fischer A. Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques. *IEEE Transactions on Visualization and Computer Graphics*, 2001, 7(1): 1∼16.

[36] Lloyd S P. Least squares quantization in PCM. *IEEE Trans. Information Theory*, 1982, 28(2): 129∼137.

[37] Kanungo T, Mount D M, Netanyahu N S *et al.* An efficient *k*-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2002, 24(7): 881∼892.

[38] Tenenbaum J, Silva V, Langford J. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000, 290(12): 2319∼2323.

[39] Jia Y-B, Mi L C, Tian J. Surface patch reconstruction via curve sampling. In *Proc. the IEEE International Conference on Robotics and Automation*, Orlando, Florida, 2006, pp.1371∼1377.

[40] Kruskal J B, Wish M. Multidimensional scaling. Beverly Hills, California: Sage Publications, 1978.

Yong-Wei Miao received his M.Sc. degree in differential geometry from the Chinese Academy of Sciences in 1996 and his Ph.D. degree in computer graphics from State Key Lab. of CAD & CG, Zhejiang University in 2007. He is currently an associated professor in Zhejiang University of Technology. His research focuses on virtual reality, digital geometry processing and computer-aided geometric design.

Jie-Qing Feng received his B.Sc. degree in applied mathematics from the National University of Defense Technology in 1992 and his Ph.D. degree in computer graphics from Zhejiang University in 1997. He is a professor in the State Key Lab of CAD&CG, Zhejiang University. His research focuses on space deformation, computer-aided geometric design, and computer animation.

Chun-Xia Xiao received his M.Sc. degree in computational mathematics from Hunan Normal University in 2002 and his Ph.D. degree in computer graphics from State Key Lab. of CAD & CG, Zhejiang University in 2006. He is currently an assistant professor in Wuhan University. His research focuses on virtual reality, digital geometry processing, point-based computer graphics, image and video editing.

Qun-Sheng Peng graduated from Beijing Mechanical College in 1970 and received his Ph.D. degree from the Department of Computing Studies, University of East Anglia, in 1983. He is a professor of computer graphics in Zhejiang University. His research focuses on realistic image synthesis, computer animation, scientific data visualization, virtual reality, feature modeling, and parametric design.

A.R. Forrest graduated in mechanical engineering from the University of Edinburgh in 1965 and received his Ph.D. degree at the University of Cambridge in 1968. He is a professorial fellow in the School of Computing Sciences, University of East Anglia, U.K. His research focuses on point-based geometric modeling, information visualization, and computational geometry.