Point Cloud Normal Estimation via Low-Rank Subspace Clustering

Jie Zhang[a], Junjie Cao[a], Xiuping Liu[a], Jun Wang[b], Jian Liu[a], Xiquan Shi[c]

[a]*School of Mathematical Sciences, Dalian University of Technology, China*
[b]*College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, China*
[c]*Department of Mathematical Sciences, Delaware State University, USA*

Abstract

In this paper, we present a robust normal estimation algorithm based on the low-rank subspace clustering technique. The main idea is based on the observation that compared with the points *around sharp features*, it is relatively easier to obtain accurate normals for the points *within smooth regions*. The points around sharp features and smooth regions are identified by covariance analysis of their neighborhoods. The neighborhood of a point in a smooth region can be well approximated by a plane. For a point around sharp features, some of its neighbors may be in smooth regions. These neighbor points' normals are estimated by principal component analysis, and used as prior knowledge to carry out neighborhood clustering. An unsupervised learning process is designed to represent the prior knowledge as a guiding matrix. Then we segment the anisotropic neighborhood into several isotropic neighborhoods by low-rank subspace clustering with the guiding matrix, and identify a consistent subneighborhood for the current point. Hence the normal of the current point near sharp features is estimated as the normal of a plane fitting the consistent subneighborhood. Our method is capable of estimating normals accurately even in the presence of noise and anisotropic samplings, while preserving sharp features within the original point data. We demonstrate the effectiveness and robustness of the proposed method on a variety of examples.

Keywords: normal estimation, sharp feature preserving, low-rank representation, subspace clustering

1. Introduction

Estimating surface normals from a noisy point cloud benefits many applications in computer graphics, geometry processing and reverse engineering. Accurate normal estimation allows a better reconstructing and rendering of point-based surfaces [1, 2, 3, 4], anisotropic smoothing [5] to name just a few.

With the assumption that the underlying surface is smooth everywhere, regression-based techniques [6, 7, 8, 9] use the whole neighborhoods to estimate normals. However, when a point is near sharp features, its neighborhood may be anisotropic and sampled from several piecewise surfaces. In such case, all regression-based methods tend to smooth sharp features, since the normal estimation of the point may be influenced by its neighbors lying on some other surfaces. To estimate normals more faithfully for point clouds with sharp features, Li *et al.*[10] (RNE) employed the robust statistics to classify points around sharp features into consistent sub-neighbors represented by different tangent planes. However, this method is sensitive to the density variation, as shown in the bottom row of Fig.1. Boulch *et al.*[11] (HF) introduced a uniform sampling strategy to overcome the influence of density variation. However, the computed normals are unreliable for the points near sharp features, as shown in Fig.1. Moreover, this method tends to blur a sharp feature when the dihedral angel is large.

When a point is extremely near a sharp feature, it is hard to estimate its normal or classify its neighbors faithfully only by using the distance information. To overcome this challenge, we present a novel robust approach to select a consistent neighborhood utilizing the structure of the underlying piecewise surfaces. These surfaces can be approximated by planes and each of them is a 2D subspace, relative to the 3D Euclidean space where the model is embedded. Thus, the segmentation of the neighborhood of a point can be solved as a subspace clustering problem, which has widespread applications in computer vision. The Low-Rank Representation (LRR), emerging in machine learning, is a powerful tool in subspace clustering [12, 13]. It captures the global structure of the data robustly assuming that the subspaces are independent. For our problem, the subspaces near sharp features are dependent, *i.e.* the sum of the dimension of these subspaces is larger than three. The standard low-rank method tends to fail for such case. Hence we present a new low-rank subspace clustering framework with prior knowledge to handle more general subspace clustering. In addition, we observe that the computation of normals which are not near sharp features (i.e. smooth regions) are reliable even with traditional regression-based methods, such as PCA. Based on this observation, we design an unsupervised learning process to estimate the guidance from reliable regions. Although it is relatively time-consuming, the consistent neighbors of a point can be effectively detected even in the presence of noise, which has many applications, including normal estimation and feature extraction, *etc.*. Fig. 2 gives the pipeline of our method. The contributions of our work are summarized as follows:

- The standard LRR model can only work under the assumption that the data are sampled from multiple independent subspaces. We provide a new low-rank subspace cluster-

Figure 1: Reconstructed normals of two plans with shallow angle by Li *et al.* [10] (left), Boulch *et al.* [11] (middle) and our algorithm (right). respectively. The points are sampled uniformly in the top row and non-uniformly in the bottom row.

ing framework with prior knowledge (LRSCPK) for more general subspace clustering. The prior knowledge can be obtained by many ways for different applications.

- For normal estimation, we use LRSCPK to segment and identify consistent subneighborhood and compute accurate normal from the subneighborhood. An unsupervised learning process is designed to estimate the prior knowledge from reliable regions, which are utilized in LRSCPK to classify unreliable regions close, even extremely close to sharp features. The estimated normals preserve sharp features even in the presence of noise and anisotropic samplings.

2. Related work

Given a 3D point cloud, how to accurately estimate normals has been a primary concern in visual computing community. We briefly review it in this section. We do not address the problem of normal orientation, which can be done separately [14, 15, 16, 17, 18].

The classical normal estimation method, proposed by Hoppe *et al.*[6] (PCA), defines the normal of a point as the eigenvector corresponding to the smallest eigenvalue of the covariance matrix of its neighbors. They assume that the local neighborhood of any input point can be approximated by a plane. There are a number of variants of this method which are compared carefully in [19]. Guennebaud *et al.*[7] and Cazals *et al.*[20] used spheres and quadrics to replace planes, respectively. Pauly *et al.*[21] proposed a weighted version of this basic approach by assigning Gaussian weights to its neighbors. By analysing the local noise, curvature and sample density, a method [22] that adaptively chooses the size of neighborhood is proposed. Yoon *et al.*[23] took the ensemble technique from statistics to improve the robustness of PCA. However, PCA and its variants tend to smooth sharp features since they actually are low-pass filters and usually need a large neighborhood to deal with noise.

Another class of methods is based on the improvement of preliminary normal estimation. The adaptive moving least squares method [24] and the robust local kernel regression method [1] estimate normals as the gradient of an implicit surface fitting the local points and their prescribed normals. Bilateral filtering proposed by [25] takes advantage of the difference between preliminary normals to recover sharp features. Though those methods can obtain nearly correct normals for the points close to sharp features, their quality heavily relies on the input normals.

Amenta *et al.*[26] first introduced the Delaunay/Voronoi technique into normal estimation. They used the Voronoi diagram and the furthest vertex of the Voronoi cell to approximate the normals of noise-free point clouds. By finding big Delaunay balls, Dey *et al.*[27] applied this idea to noisy point-clouds. The combination of principal component analysis and Voronoi is proposed by Alliez *et al.*[28] to obtain more stable normals. However, none of them can estimate the normals of points near/on sharp features accurately.

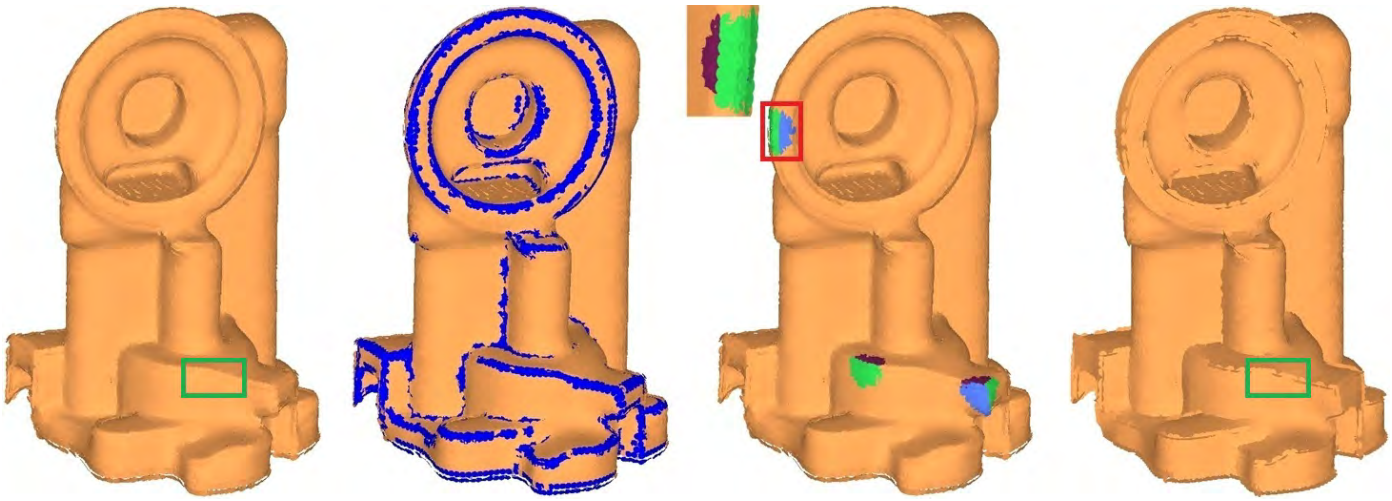
More recently, by combining a robust noise-scale estimator and a kernel density estimation, Li *et al.*[10] (RNE) proposed a robust normal estimation method which can handle noise and sharp features well. However, it does not address variation of density at edges, for the kernel density estimation favors the sides with high density. Boulch *et al.*[11] (HF) used a stop criterion inherited from robust statistics to speed the Randomized Hough Transform and a uniform sampling strategy to overcome the density anisotropy. However, when the dihedral angle between the two planes forming the sharp feature is large, the difference is small between normals produced by triples sampled from different sides, so these normals are likely to vote for the same bin. Consequently, it tends to generate smooth effect near sharp features.

3. Low-rank subspace clustering with prior knowledge

In this section, we present our low-rank subspace clustering framework with prior knowledge, based on which our normal estimation algorithm is designed (Section 4). First, we give a short review of subspace clustering. Subspace clustering is a fundamental problem which has numerous applications in image processing and computer vision. In practice, the data points are always drawn from a union of subspaces with lower intrinsic dimensions than the dimension of the ambient space. The task of subspace clustering is to segment the data into multiple low-dimensional linear subspaces.

3.1. Sparse subspace clustering (SSC)

The recently proposed sparse representation, low-rank representation achieves competitive results in subspace clustering [12, 13, 29]. These methods are all based on the observation that a data vector can be represented by the data vectors



(a) (b) (c) (d)

Figure 2: Method overview. (a) The oil pump module with the normal computed by PCA. (b) Initial detected candidate feature points. (c) The classified subneighborhoods. The neighborhood within the red box contains three subneighborhoods rendered in blue, green and brown and the zoomed view is from left. (d) Estimated normals. Comparing (a) with (d), our method preserves the sharp features better, and the RMS_τ introduced in section 5 is 0.6716 and 0.1308, respectively.

from the same subspace and their models can be summarized as follows

$$min f(Z) \quad s.t. \quad X = XZ, \tag{1}$$

where $f(Z)$ is a matrix function, each column of the matrix X represents a sample. The affinity matrix S can be defined as

$$S = |Z| + |Z'|. \tag{2}$$

Then the NCut method [30] is applied to get the clustering result. The key of these methods is to define an appropriate function f.

Sparse subspace clustering (SSC) proposed by E. Elhamifar *et al.* [29] uses the 1D sparsity. It is an effective method for the independent subspace clustering and the model can be written as

$$min\|Z\|_1 \quad s.t. \quad X = XZ, diag(Z) = \mathbf{0}, \tag{3}$$

where $\|\cdot\|_1$ represents l_1-norm. However, [31] gives failure examples in dimensions higher than three. Moreover, SSC finds the sparsest representation of each sample individually without global constraints.

3.2. Low-rank subspace clustring (LRSC)

To capture the global structure of the whole data, low-rank representation [12, 13] considering the 2D sparsity, *i.e.* the rank of a matrix, is presented for subspace clustering. The model can be written as

$$min\|Z\|_* \quad s.t. \quad X = XZ, \tag{4}$$

where $\|\cdot\|_*$ represents the sum of singular value. It is an effective subspace clustering algorithm that outperforms the state-of-the-art algorithms in handling corrupted data. However, when the subspaces are not independent, this method may fail.

3.3. Low-rank subspace clustering with prior knowledge (LRSCPK)

To segment two or more intersection planes, an affinity matrix which is dense between same class and sparse between different classes is preferred. Thus we propose a low-rank subspace clustering framework (LRSCPK) with prior knowledge:

$$min\|Z\|_* + \beta\|\mathcal{P}_\Omega(Z)\|_1 \quad s.t. \quad X = XZ. \tag{5}$$

where Ω is a guiding matrix and $0 \le \Omega(i,j) \le 1$, $\mathcal{P}_\Omega$ can be seen as a mapping satisfying $\mathcal{P}_\Omega(Z)_{i,j} = \Omega(i,j) \times Z(i,j)$. The guiding matrix Ω can be seen as prior knowledge. Ω has the property that the samples from the same intra class contribute to a smaller weight while the samples from interclass contribute to a larger weight. We note that the subspace can be clustered correctly even when Ω contains partial prior and gentle noise in our experiments.

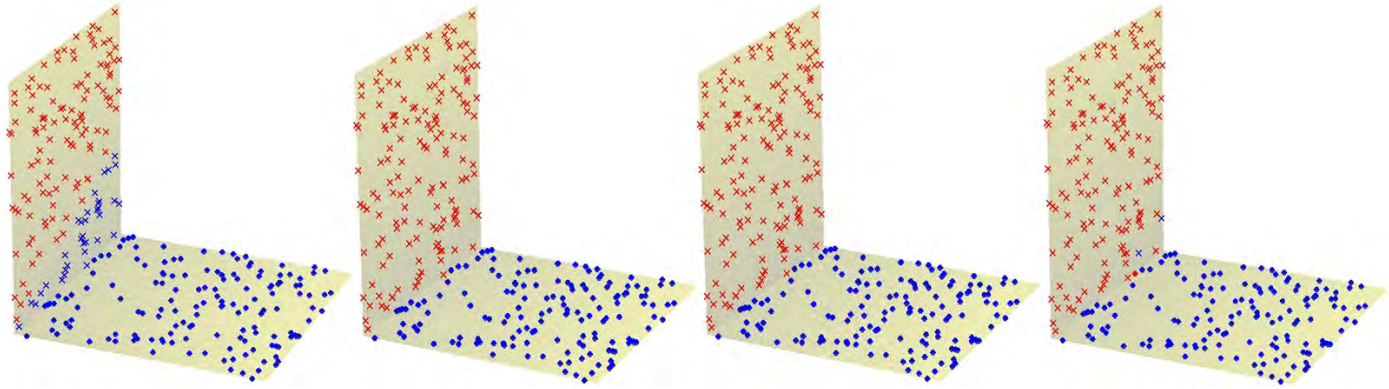
In order to improve the robustness of this method, we replace the equality constraint of problem (5) with a soft constraint

$$min\|Z\|_* + \beta\|\mathcal{P}_\Omega(Z)\|_1 + \gamma\|E\|_{2,1} \quad s.t. \quad X = XZ + E, \tag{6}$$

where β and γ are two parameters and we set them as 1 in this work, $\|\cdot\|_{2,1}$ is the $l_{2,1}$ norm and defined as the sum of l_2 norms of the columns of a matrix.

3.3.1. A toy example

A toy example is provided to verify the effectiveness and robustness of LRSCPK. We sample 300 points from two intersecting planes in $\mathbb{R}^3$, 150 for each plane. Forty percent prior knowledge with 20%, 30%, 40% errors are presented to LRSCPK. Specifically, the ideal full guiding matrix $G \in \mathbb{R}^{300*300}$ has the property that if two samples p_j and p_k are in the same plane, $G(j,k) = 0$, otherwise $G(j,k) = 1$. The guiding matrix Ω is generated by choosing 40% elements of G randomly and the other elements of Ω are all zeros. The errors are added by randomly selecting 20%, 30%, 40% elements from the 40% elements



(a) (b) (c) (d)

Figure 3: Segmentation results of LRR and LRSCPK on two intersecting planes. (a) The result of LRR with 11% clustering error. From (b) to (d) are the results of LRSCPK with 40% prior knowledge which are corrupted by 20%, 30% and 40% errors, respectively. The clustering error is 0, 0 and 1%.

and switching their values, *i.e.* from ones to zeros if they are ones originally (resp. zero to one). As illustrated by Fig. 3 (a), (b) and (c), LRSC fails to segment two dependent subspaces, while LRSCPK generates faithful segmentation using partial prior knowledge even with considerable levels of errors. Moreover, when the prior knowledge is seriously corrupted by errors, LRSCPK still provides a more qualified segmentation than LRSC, as shown in Fig. 3 (d).

4. LRSCPK for normal estimation

4.1. Overview of our algorithm

With a noisy point cloud $\mathcal{P} = \{p_i\}_{i=1}^N$ as input, our algorithm takes three steps to estimate the normals, while preserving sharp features. First, we detect the points which are near sharp features. Then, for each of these points, the neighborhood of it is segmented by LRSCPK with prior knowledge estimated from its local structure. Finally, using the segmentation result we select a consistent subneighborhood to estimate the normal. The overall procedure of our method is shown in Fig. 2.

4.2. Candidate feature points selection

Given a point p_i, we select a neighborhood $\mathcal{N}_i$ of size S which depends on the noise scale. Then we compute a weight w_i and an estimated normal n_i by covariance analysis of the local neighborhood. Denote $\lambda_0 \le \lambda_1 \le \lambda_2$ as the singular values of the covariance matrix defined in [21]. The weight w_i is computed as

$$w_i = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2}.$$

The normal n_i is defined as the eigenvector v_0 corresponding to the smallest eigenvalue λ_0.

The weight w_i measures the confidence of p_i belonging to a feature. If w_i is larger than a given threshold w_t then p_i is regarded as a candidate feature point. Denote the distribution of $\{w_i\}_{i=1}^N$ as f_w and smooth it by

$$min_{\hat{f}_w} \quad \|\hat{f}_w - f_w\|_F + \|Df_w\|_1,$$

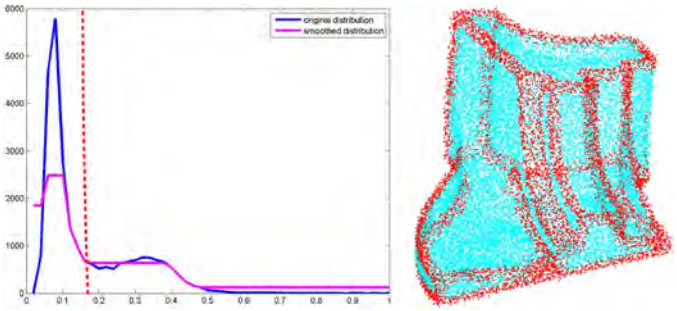


Figure 4: The choice of w_t (left) and the candidate feature points (right). The red dashed line represent where the threshold is selected

where D is the second difference matrix, $\|\cdot\|_F$ and $\|\cdot\|_1$ represent l_2 norm and l_1 norm, respectively. We choose the w_t after the first peak of the smoothed distribution with gently decrease, as shown in Fig.4.

4.3. Subneighborhood segmentation

4.3.1. Subneighborhood segmentation by subspace clustering

For a candidate feature point p_i, we first select a larger neighborhood $\mathcal{N}_i^*$ of size S^* than $\mathcal{N}_i$ used for candidate feature points selection, *i.e.* $S^* > S$. In the local coordinates with p_i as the origin, the neighbor point p_{ij} of p_i is represented as $p_{ij} = [x_j, y_j, z_j, n_{xj}, n_{yj}, n_{zj}]', j = 1, \ldots, S^*$, where $[x_j, y_j, z_j]$ is the coordinate of $p_{ij} \in \mathcal{N}_i^*$ and $[n_{xj}, n_{yj}, n_{zj}]$ is its normal computed by PCA. The sampling matrix is defined as $X = [p_{i1}, p_{i2}, \ldots, p_{iS^*}]$. The optimal coefficient matrix $Z \in \mathbb{R}^{S^* \times S^*}$ can be computed by solving *Equation* (6). After obtaining the affinity matrix from *Equation* (2), we segment $\mathcal{N}_i^*$ by NCut. Since the number of subspaces is unknown, we segment these neighbor points into two classes first. If any one of them is not a subspace, it is segmented into two classes further. The iterative progress is terminated until each class is a subspace.

Determining whether or not a class is a subspace is performed by fitting a plane to the class and measuring the average

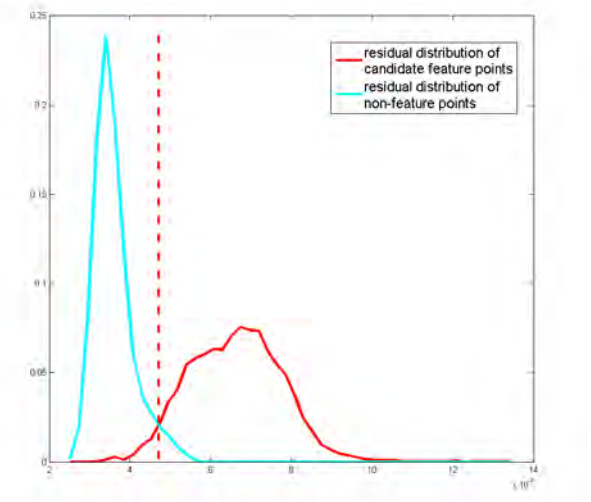


Figure 5: The choice of τ_f. The red dashed line represent where the threshold is selected

residual. If the average residual is smaller than a threshold τ_f, the class is determined as a subspace. To automatically compute an appropriate τ_f, we compute normalized average residual histograms of candidate feature points and non-candidate points respectively and set τ_f as the horizontal ordinate where the two histograms cross, as illustrated in Fig.5.

4.3.2. Construction of the guiding matrix

The guiding matrix Ω can be built by various ways. We estimate it according to the observation that although normals estimated by PCA are error-prone for points around sharp features, they are reliable for points away from sharp features. Thus the reliable regions are used to guide the segmentation of the neighborhood. For any two points p_{ij} and p_{ik} in the neighborhood of p_i, we compute the normals of p_{ij} and p_{ik} using their K-nearest neighbors respectively and compute the distance $D(j,k)$ between the two planes specified by the normals. If p_{ij} or p_{ij} is adjacent to sharp features, *i.e.* with larger w_{ij} or w_{ik}, no plane approximates its neighborhood well. We randomly choose r points from its K-nearest neighbors to estimate its normal. The construction of the distance matrix D is shown in **algorithm** 1.

$D(j,k)$ can be seen as the probability of points j and k belonging to different planes, *i.e.* subspaces. We set $\Omega(j,k) = 1$ if $D(j,k)$ is larger than a threshold τ_{max}, and set $\Omega(j,k) = 0$ otherwise. Denoting D_{max} as the set of the largest 40% elements in the matrix D, we set $\tau_{max} = min(D_{max}, 1 - cos(45°))$. This setting of τ_{max} means that when the angle between the two estimated planes of p_{ij} and p_{ik} is larger than 45°, they are predicted to belong to different subspaces.

Sequence of computing the guiding matrix. In order to improve the reliability of the guiding matrix, we first segment neighborhoods of points with smaller w_i, store the segmentation results of each neighborhood and use them to update the guiding matrices of points around sharp features which are computed later. We denote the number of p_{ij} and p_{ik} in the same subspace and in different subspaces as R_{jk} and N_{jk}, respectively. If $R_{jk} > N_{jk}$, we define $\Omega(j,k)$ as $\Omega(j,k) = min\left(\Omega(j,k), 1 - \frac{R_{jk}}{R_{jk}+N_{jk}} \times exp(-1/R_{jk})\right)$. Otherwise we define $\Omega(j,k)$ as $\Omega(j,k) = max\left(\Omega(j,k), \frac{N_{jk}}{R_{jk}+N_{jk}} \times exp(-1/N_{jk})\right)$.

Algorithm 1: Construct The Distance Matrix

Input: *CandidateFeaturePointSet A*, *Non-FeaturePointSet B*;
Output: *DistanceMatrixD*;

```
DistanceMatrix D = 0;
for each point p_ij ∈ A ∪ B do
    for each point p_ik ∈ A ∪ B do
        if p_ij ∉ A then
            Normal n1 = normal(NearestNeighbors);
        else
            Normal n1 = normal(RandomNearestNeighbors);
        if p_ik ∉ A then
            Normal n2 = normal(NearestNeighbors);
        else
            Normal n2 = normal(RandomNearestNeighbors);
        D(j,k) = 1 − | < n1, n2 > |
```

Improvement for points near sharp features. When p_{ij} or p_{ik} is near sharp features, the normal estimated using r random points from a point's K-nearest neighbors is not reliable. Therefore the reliability of the probabilities $D(j,:)$ or $D(k,:)$ is lower. We reduce the corresponding element in the guiding matrix Ω. Specifically, if only one of p_{ij} or p_{ik} is around sharp features we set $\Omega(j,k) = \alpha\Omega(j,k)$, $0 < \alpha < 1$. If both of them are near sharp features we set $\Omega(j,k) = \beta\Omega(j,k)$, $0 < \beta < \alpha$. We take $\alpha = 0.6$ and $\beta = 0.2$ in our experiments.

4.4. Normal estimation

For the non-candidate points, their neighborhood is isotropic and normals estimated by PCA is used. For any candidate feature point p_i, its neighborhood is anisotropic. We segment its neighborhood into several isotropic subneighborhoods by the method described in section 4.3. For each subneighborhood, a plane is fitted to p_i and the subneighborhood. Thus each subneighborhood has a fitting residual. The subneighborhood with the minimum residual is identified as a consistent subneighborhood of p_i. Then an accurate normal can be easily estimated using the consistent subneighborhood.

4.5. Time complexity of our algorithm

Our algorithm takes three steps: candidate feature points selection, subneighborhood segmentation and normal estimation. The time complexity of them are $O(N\log N)$, $O(N_f \times (\log N + (S^*)^3))$ and $O(N_f \log N)$, respectively, where N is the number of points and N_f is the number of the candidate feature points. We use Kd-tree for neighborhood search (the $\log N$ factor) and repeat some local operations for each point (the N factor) in the

first step and for each candidate feature point (the N_f factor) in the remaining steps. For each candidate feature point, the most time-consuming operation is to solve *Equation* (6) when segmenting subneighborhood of size S^*. A modified alternating direction method (ADM) is designed with the same complexity of the standard one [32] (the $(S^*)^3$ factor) and presented in Appendix 1. Thus the time complexity of our algorithm is $O(N\log N + N_f \times (S^*)^3)$.

5. Results and applications

A variety of noisy point clouds with sharp features are tested to evaluate the performance of our approach. We also compare our method with some classic and state-of-the-art algorithms: PCA [6], robust normal estimation (RNE) [10] and hough transform (HF) [11].

To evaluate the estimation accuracy, the Root Mean Square measure with threshold (RMS_τ) [11] is used as follows:

$$RMS\_\tau = \sqrt{\frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}}(f(\widehat{n_{p,ref}n_{p,est}}))^2}, \tag{7}$$

where

$$f(\widehat{n_{p,ref}n_{p,est}}) = \begin{cases} \widehat{n_{p,ref}n_{p,est}}, & if \widehat{n_{p,ref}n_{p,est}} < \tau \\ \pi/2, & otherwise \end{cases},$$

$n_{p,ref}$ is the reference normal at p and $n_{p,est}$ is the estimated one. We take $\tau = 10$ degrees, as proposed by [11]. In the following, the points with the measure greater than 10 degrees are considered as bad points.

The parameters S, S^*, K and r of our algorithm are chosen as $S = 70$, $S^* = 120$, $K = 30$ and $r = 10$ for all models in this section. We use the same neighborhood size S^* for all the three methods. All other parameters, if any, are set to default. We evaluate the estimation accuracy on the data with synthetic noise: a centered Gaussian noise with deviation defined as a percentage of average distance between points.

5.1. Comparisons with other methods

Sharp features with shallow angles. In Fig. 6, we compare the normal estimation results of the Octahedron model. It contains sharp features generated by two intersection planes with shallow angles. Normals estimated by PCA are overly smoothed on/near sharp features. Those of HF are also a bit smooth, since normals produced by triples sampled from different sides are likely to vote for the same bin when the angle is shallow. The bottom row of Fig. 6 demonstrates that our algorithm preserves sharp features better. We can see that the number of bad points and RMS_τ are both smaller from our method, as shown in the middle row of Fig. 6 and Tab.1.

Neighborhood with multiple features. The normal estimation results of Fandisk model are compared in Fig. 7. The neighborhood of a point may contain multiple feature lines as the marked narrow-band regions in Fig. 7. The complex neighborhood structure brings more challenges to the existing normal estimation methods. As Fig. 7 shown, PCA, RNE and HF

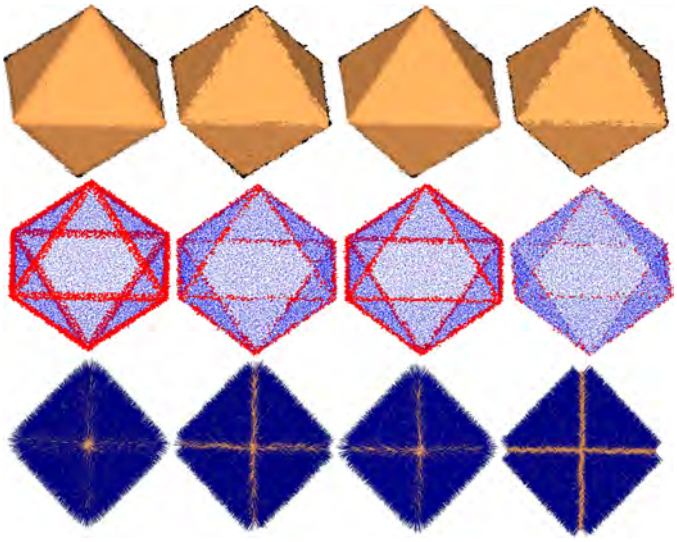


Figure 6: Comparison for Octahedron model with 50% noise. From the top to bottom row are the results rendering using surfels, the visualization of bad points and top view of the generated normals, respectively. Left to right are the results of PCA, RNE, HF and our algorithm, respectively.

generate more bad points around the narrow-band regions. Our method utilizing the neighborhood's structure information does not have this flaw. In addition, the bottom row of Fig. 7 shows that our method preserves shape features comparatively better in presence of noise, while the others all blur sharp features.

The bad point distributions of PCA, RNE, HF and our method are presented in Fig. 8 for the Octahedron and Fandisk models. We notice that the number of bad points from our method are much less than other methods in all normal deviation regions from 10 to 90 degree nearly. For the Fandisk model, PCA obtains the smallest number of bad points with normal deviation between 80 to 90 degrees among all compared methods. It is because that the normals near sharp features by PCA are excessively smoothed and the largest deviations are almost 40 to 60 degrees. The higher deviation between the above regions from RNE, HF and our method is mainly because that heavy noise makes the intersection of two planes becoming a ribbon from a line, where the points are supposed to have two directions.

Variational density near sharp features. Fig. 9 shows the normals estimated on a Cube sampled with face-specific variations of density. PCA and RNE are severely affected by the non-uniform sampling. Since HF has designed a sampling strategy to deal with density variation, it performs better than PCA and RNE. However, there are still some errors near sharp features. Our method can effectively overcome the sampling anisotropy around sharp features.

Computational statistics. To evaluate our method more precisely, models with different noise scales are computed, and RMS_τ and the number of bad points are presented in Tab. 1. The statistics show that our method achieves the lowest RMS_τ and NBP. More accurate normals are estimated under different noise scales.

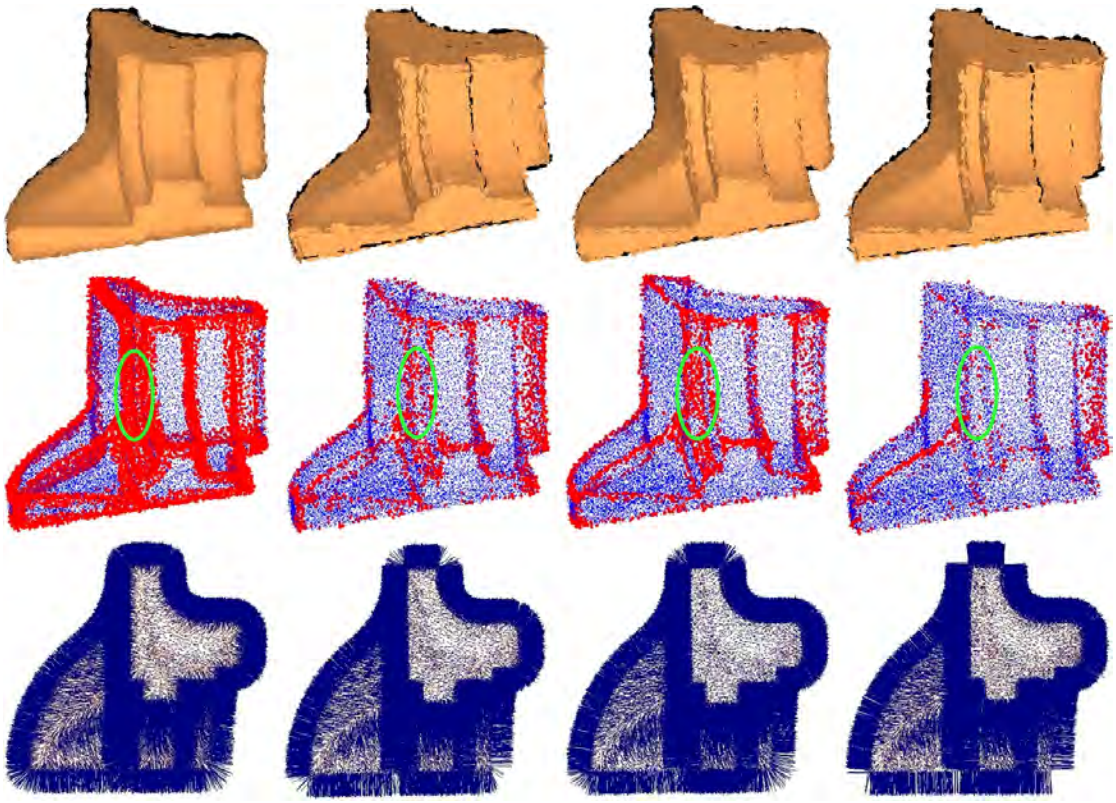


Figure 7: Comparison for Fandisk model with 50% noise. From top to bottom row are the results rendering using surfels, the visualization of bad points and top view of the generated normals, respectively. Left to right are the results of PCA, RNE, HF and our algorithm, respectively.

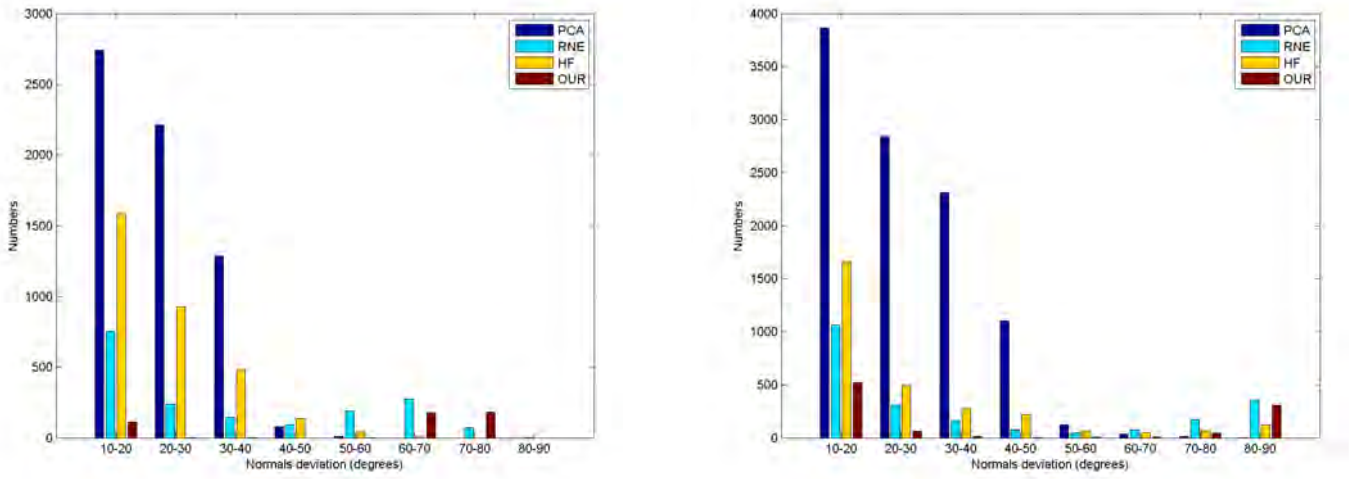


Figure 8: The histograms of the normal deviation on Octahedron model (left) and Fandisk model (right) shown in Fig.6 and Fig.7, respectively.

Table 1: RMS_τ and the number of bad points (NBP) on Fandisk, Octahedron and Cube model with different noise scales.

Model	Noise scale	PCA		RNE		HF		OUR	
		RMS_τ	NBP	RMS_τ	NBP	RMS_τ	NBP	RMS_τ	NBP
Octahedron	40%	0.7730	6338	0.3115	1005	0.4399	2034	**0.1424**	**200**
Octahedron	50%	0.7722	6324	0.4123	1772	0.5496	3183	**0.2174**	**478**
Octahedron	60%	0.7773	6406	0.4910	2518	0.6161	4002	**0.2986**	**914**
Fandisk	40%	0.9874	10206	0.3708	1469	0.4022	1668	**0.2601**	**688**
Fandisk	50%	0.9921	10303	0.4694	2272	0.5341	2957	**0.3088**	**970**
Fandisk	60%	1.0010	10487	0.5578	3214	0.6568	4485	**0.3809**	**1483**
Cube	40%	0.5889	4510	0.2606	879	0.1043	135	**0.0667**	**51**
Cube	50%	0.5865	4473	0.2537	829	0.1063	138	**0.0674**	**50**
Cube	60%	0.5864	4471	0.2577	851	0.1147	159	**0.0851**	**82**

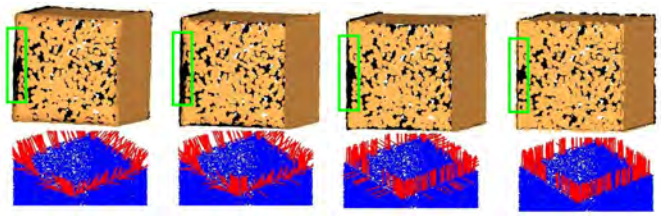


Figure 9: Comparison for Cube model with 50% noise. Density is uniform on each face; if density is 1 for the front face, then it is 5 for the upper face and 10 for the right face. The top row contains the results rendering using surfels, and the bottom row contains the top view of computed normals of the front face of the Cube model. Left to right are the results of PCA, RNE, HF and our algorithm, respectively.

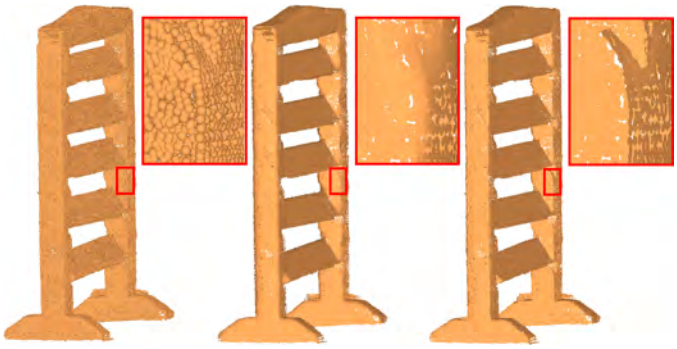


Figure 11: Normal estimation for the Shutter model. Left to right are the input raw scan model with 291.2k points, the results of PCA and our algorithm, respectively.

5.2. More results for raw scans of real objects

We also apply our method to real scanned point clouds, see Fig. 10 and Fig. 11, in which the typical imperfections, such as noise, outliers and non-uniform distribution are ubiquitous. Although the edges of these models are corrupted with noise, especially for the Genus2 and Box models in Fig. 10, our algorithm recovers them faithfully. The Genus2 model also contains close-by surface sheets in the marked regions. The normal estimation approaches based on distance, such as PCA and its variants, tend to be affected greatly, while our structure based method not. Both the Armadillon and Taichi models are non-uniform sampled. The Taichi model has complex structures in the marked region of Fig. 10. Our algorithm preserves the sharp feature well even in such complicated case. Tiny structures challenge normal estimation. The Shutter in Fig. 11 contains many such tiny structures. When some of them are sampled adequately as the marked region, faithful normals are estimated by our method and the tiny structures can be visualized clearer.

5.3. Implementation details

The proposed approach has been implemented in Matlab and is not optimized for efficiency. All experiments have been performed with 1 CPU Intel(R) Xeon(R) 2.53GHz. A typical example such as the Armadillon model with 99.4k points and 8.4k candidate feature points in Fig. 10, takes a total of 4618 seconds. Of that time, candidate feature points selection, subneighborhood segmentation and normal estimation take 21, 4595 and 2 seconds respectively. As stated in section 4.5, the most time-consuming operation is to solve *Equation* (6) when segmenting subneighborhood of size S^* for each candidate feature point. A parallel C++ implementation with the linearized alternating direction approach [33] could increase the performance remarkably.

5.4. Sharp feature detection

We further demonstrate the effectiveness of our neighborhood segmentation algorithm on sharp feature detection, which is a fundamental problem in geometry analysis. The sharp features are usually detected as the intersection of multiple piecewise smooth surfaces. Rejecting the pseudo features from real features challenges existing feature extraction methods, since they both are close to the intersection with similar local information, and the difference between the closeness is hard to distinguish. Previous algorithms usually use multi-scale scheme or anisotropic neighborhood selection to extract feature points [34, 35, 36, 37]. However, these methods still could not effectively distinguish them by only using distance information. Taking use of the neighborhood segmentation based on subspace structure, we design the following feature extraction method.

For each candidate feature point p_i, we segment its neighborhood into some subneighborhoods and fit a plane for each.

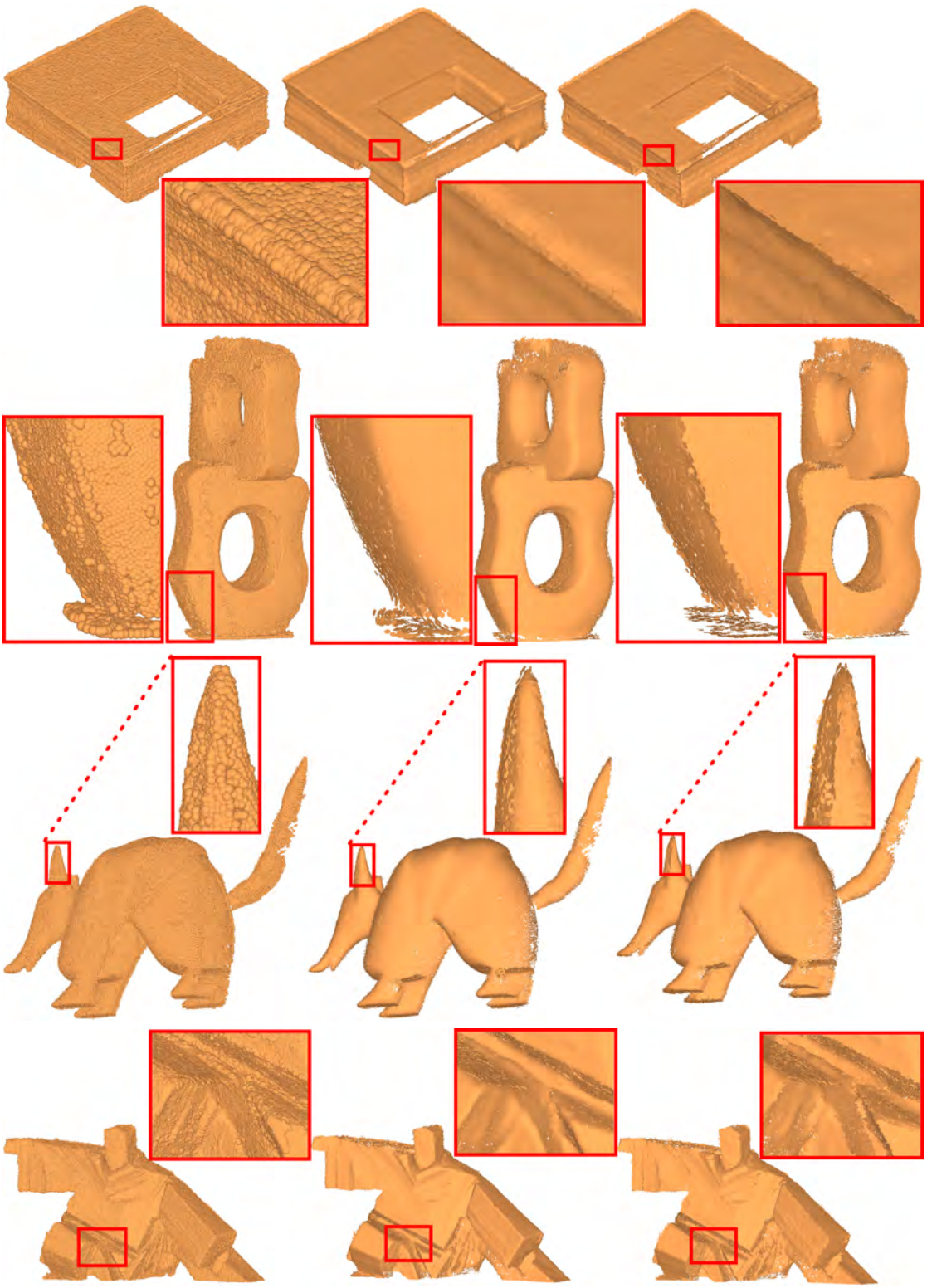


Figure 10: Normal estimation for raw scans of real objects: Box (222.5k points), Genus2 (50k points), Armadillon (99.4k points) and Taichi (537.6k points). Left to right are the input model, the results of PCA and our algorithm, respectively.

Then we compute the residuals between these fitting planes and p_i. If there is more than one residual which is less than the threshold $\tau_{feature}$, we consider p_i as a feature point. It is worth noting that because we use average residual to define the τ_f, we set $\tau_{feature} = 2 \times \tau_f$.

To assess the quality of sharp features detected by the method, we test it on synthetic models with random noise and compare it with the recent work [37] (MSTV). The parameters S, S^*, K and r of our algorithm are choose as $S = 20$, $S^* = 60$, $K = 10$ and $r = 4$ for all tests in this section.

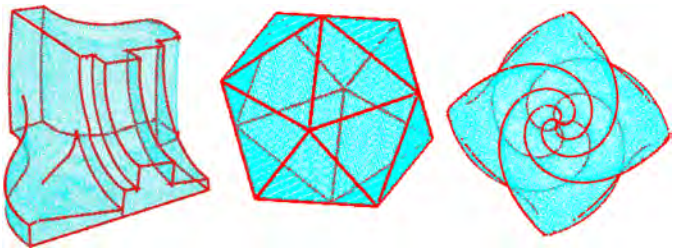


Figure 12: Sharp feature detection results.

Detection results of three different models are show in Fig. 12. For these noise-free models, our method achieves almost perfect detection results, although the shapes are complex. There are Fandisk model, Icosahedron model which has corners with five adjoining planes, and Flower model where sharp features are jaggy. We can see that our method can detect the weak features well, especially in the Flower model.

In order to demonstrate the robustness of our method to noise, the results of Fandisk and Smooth-feature models are shown in Fig.13 and Fig.14, which are perturbed by centered Gaussian noise with the standard deviation of 10% and 20% average distance between points, respectively. As illustrated in Fig.13, our method is able to detect the sharp features in different noise scale for the Fandisk model. Moreover, the weak features are recovered perfectly. The visual comparison between the recent method MSTV and our method is given in Fig.14. MSTV can not restore weak features perfectly and some weak feature points at the top of the model are missing, as shown on the top of Fig.14. While evidence from the bottom of Fig. 14 demonstrates that our method is capable of recovering more weak feature points to a large extent.

6. Conclusion

Motivated by the observation that the segmentation of the neighborhood of the point near sharp features can be treated as subspace clustering, we propose a novel normal estimation method for point clouds, which is able to preserve sharp features even in the presence of heavy noise. In order to obtain more stable segmentation, a more general LRR framework for subspace clustering with guidance from prior knowledge is presented. We also design an unsupervised learning process to estimate the guidance from reliable regions for the purpose of normal estimation. The experiments exhibit that our method is

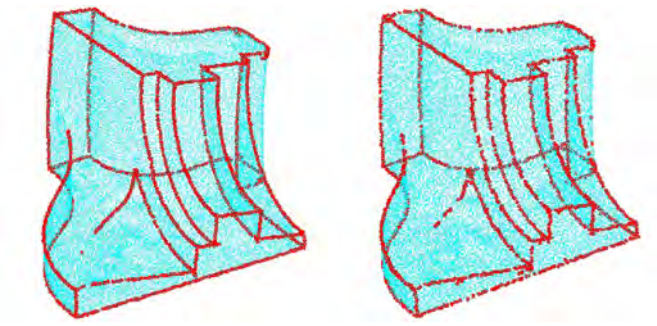


Figure 13: Sharp feature detection results for the Fandisk model with 10% (left) and 20% (right) Gaussian noise.

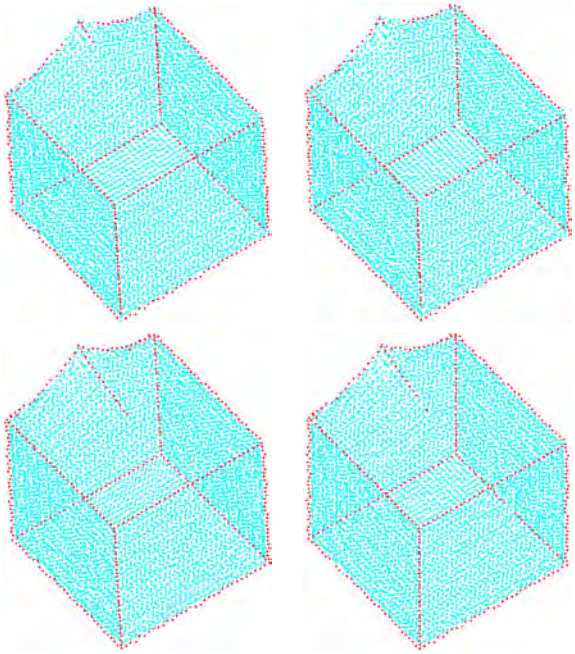


Figure 14: The sharp feature detect results by MSTV (top) and our method (bottom). Left: Smooth-feature model with 10% Gaussian noise. Right: Smooth-feature model with 20% Gaussian noise.

more resilient to noise than state-of-the-art methods. Moreover, it is robust to non-uniform sampling that is a more challenging issue in normal estimation. It would also be interesting to apply our guided low-rank subspace clustering framework to more computer vision and computer graphics applications.

As described in Section 4, our method requires a substantial amount of computation time to solve *Equation* (6) at each candidate feature point. In the future, we try to find a more efficient algorithm to solve *Equation* (6) and design a parallel implementation for our normal estimation method. Another future work is to choose the four parameters S, S^*, K and r adaptively according to noise level and sampling density.

Acknowledgement

The authors would like to thank all the reviewers for their valuable comments. This work is partially supported by the

National Natural Science Foundation of China (No. 61173102, U0935004, 61173103, 91230103) and the Research Funds for the Central Universities (No. DUT13RC206, DUT13JS04). Thanks to Bao Li and Alexandre Boulch for providing the code used for comparison. The Armadillon model is courtesy of Andrei Sharf. The models of Box and Shutter are provided by Hui Huang.

References

[1] A. C. Öztireli, G. Guennebaud, M. H. Gross, Feature preserving point set surfaces based on non-linear kernel regression, Comput. Graph. Forum 28 (2) (2009) 493–501.
[2] S. Rusinkiewicz, M. Levoy, Qsplat: a multiresolution point rendering system for large meshes, in: SIGGRAPH, 2000, pp. 343–352.
[3] J. Wang, D. xiao Gu, Z. Yu, C. Tan, L. Zhou, A framework for 3d model reconstruction in reverse engineering, Computers & Industrial Engineering 63 (4) (2012) 1189–1200.
[4] J. Wang, Z. Yu, W. Zhu, J. Cao, Feature-preserving surface reconstruction from unoriented,noisy point data, Computer Graphics Forum.
[5] C. Lange, K. Polthier, Anisotropic smoothing of point sets, , Computer Aided Geometric Design 22 (7) (2005) 680–692.
[6] H. Hoppe, T. DeRose, T. Duchamp, J. A. McDonald, W. Stuetzle, Surface reconstruction from unorganized points, in: SIGGRAPH, 1992, pp. 71–78.
[7] G. Guennebaud, M. H. Gross, Algebraic point set surfaces, ACM Trans. Graph. 26 (3) (2007) 23.
[8] F. Cazals, M. Pouget, Estimating differential quantities using polynomial fitting of osculating jets, Computer Aided Geometric Design 22 (2005) 121–146.
[9] N. J. Mitra, A. Nguyen, L. J. Guibas, Estimating surface normals in noisy point cloud data, Int. J. Comput. Geometry Appl. 14 (4-5) (2004) 261–276.
[10] B. Li, R. Schnabel, R. Klein, Z.-Q. Cheng, G. Dang, S. Jin, Robust normal estimation for point clouds with sharp features, Computers & Graphics 34 (2) (2010) 94–106.
[11] A. Boulch, R. Marlet, Fast and robust normal estimation for point clouds with sharp features, Comput. Graph. Forum 31 (5) (2012) 1765–1774.
[12] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, Y. Ma, Robust recovery of subspace structures by low-rank representation, CoRR abs/1010.2955.
[13] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: ICML, 2010, pp. 663–670.
[14] J. Cao, Y. He, Z. Li, X. Liu, Z. Su, Orienting raw point sets by global contraction and visibility voting, Computers & Graphics 35 (3) (2011) 733–740.
[15] J. Wang, Z. Yang, F. Chen, A variational model for normal computation of point clouds, Visual Computer (2012) 163–174.
[16] L. M. Seversky, M. S. Berger, L. Yin, Harmonic point cloud orientation, Computers & Graphics (2011) 492–499.
[17] S. Liu, C. C. L. Wang, Orienting unorganized points for surface reconstruction, Computers & Graphics 34 (3).
[18] H. Huang, D. Li, H. Zhang, U. Ascher, D. Cohen-Or, Consolidation of unorganized point clouds for surface reconstruction, TOG 28 (5).
[19] K. Klasing, D. Althoff, D. Wollherr, M. Buss, Comparison of surface normal estimation methods for range sensing applications, in: Proc. IEEE Conf. on Robotics and Automation, 2009, pp. 3206–3211.
[20] F. Cazals, M. Pouget, Estimating differential quantities using polynomial fitting of osculating jets, in: Symposium on Geometry Processing, 2003, pp. 177–187.
[21] M. Pauly, R. Keiser, L. Kobbelt, M. H. Gross, Shape modeling with point-sampled geometry, ACM Trans. Graph. 22 (3) (2003) 641–650.
[22] N. J. Mitra, A. Nguyen, Estimating surface normals in noisy point cloud data, in: Symposium on Computational Geometry, 2003, pp. 322–328.
[23] M. Yoon, Y. Lee, S. Lee, I. P. Ivrissimtzis, H.-P. Seidel, Surface and normal ensembles for surface reconstruction, Computer-Aided Design 39 (5) (2007) 408–420.
[24] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C. T. Silva, Point set surfaces, in: IEEE Visualization, 2001.
[25] T. R. Jones, F. Durand, M. Zwicker, Normal improvement for point rendering, IEEE Computer Graphics and Applications 24 (4) (2004) 53–56.
[26] N. Amenta, M. W. Bern, Surface reconstruction by voronoi filtering, Discrete & Computational Geometry 22 (4) (1999) 481–504.
[27] T. K. Dey, S. Goswami, Provable surface reconstruction from noisy samples, Comput. Geom. 35 (1-2) (2006) 124–141.
[28] P. Alliez, D. Cohen-Steiner, Y. Tong, M. Desbrun, Voronoi-based variational reconstruction of unoriented point sets, in: Symposium on Geometry Processing, 2007, pp. 39–48.
[29] E. Elhamifar, R. Vidal, Sparse subspace clustering, in: CVPR, 2009, pp. 2790–2797.
[30] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[31] B. Nasihatkon, R. Hartley, Graph connectivity in sparse subspace clustering, in: CVPR, 2011, pp. 2137–2144.
[32] Z. L. M. C. L. W. Y. Ma, The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, UIUC Technical Report UILU-ENG-09-2215.
[33] Z. Lin, R. Liu, Z. Su, Linearized alternating direction method with adaptive penalty for low-rank representation, in: NIPS, 2011, pp. 612–620.
[34] M. Pauly, R. Keiser, M. H. Gross, Multi-scale feature extraction on point-sampled surfaces, Comput. Graph. Forum 22 (3) (2003) 281–290.
[35] J. D. II, L. K. Ha, T. Ochotta, C. T. Silva, Robust smooth feature extraction from point clouds, in: Shape Modeling International, 2007, pp. 123–136.
[36] C. Weber, S. Hahmann, H. Hagen, Sharp feature detection in point clouds, in: Shape Modeling International, 2010, pp. 175–186.
[37] M. K. Park, S. J. Lee, K. H. Lee, Multi-scale tensor voting for feature extraction from unstructured point clouds, Graphical Models 74 (4) (2012) 197–208.
[38] J.-F. Cai, E. J. Candès, Z. Shen, A singular value thresholding algorithm for matrix completion, SIAM Journal on Optimization 20 (4) (2010) 1956–1982.

Appendix A. Solving of LRSCPK

Alternating Direction Method (ADM) [32] is used to solve *Equation* (6). First, by introducing two auxiliary variables, *Equation* (6) can be written as

$$\begin{aligned} &min_{Z,E}\|J\|_* + \beta\|\mathcal{P}_\Omega(L)\|_1 + \gamma\|E\|_{2,1}, \\ &s.t. X = XZ + E, Z = J, Z = L. \end{aligned} \tag{A.1}$$

Then the problem can be solved by minimizing the following augmented lagrange function

$$\begin{aligned} &\|J\|_* + \beta\|\mathcal{P}_\Omega(L)\|_1 + \gamma\|E\|_{2,1} + \langle Y^B, Z - L\rangle \\ &+\langle Y^A, X - XZ - E\rangle + \langle Y^C, Z - J\rangle + \tfrac{\mu}{2}(\\ &\|Z - J\|_F^2 + \|X - XZ - E\|_F^2 + \|Z - L\|_F^2), \end{aligned} \tag{A.2}$$

where Y^A, Y^B and Y^C are lagrange multipliers. We outline the procedure of the algorithm in Algorithm 2. The closed-form solutions of step 1 and step 4 can be solved via Soft Value Thresholding (SVT) operator [38] and the Lemma 3.2 in [13], respectively.

Algorithm 2: Solving *Equation* (6) by ADM

Input: matrix $\mathbf{X}, \mathbf{\Omega}$, parameter β and γ

Initialize: $\mathbf{Z} = \mathbf{J} = \mathbf{L} = \mathbf{0}$, $\mathbf{E} = \mathbf{0}$, $\mathbf{Y}^A = \mathbf{Y}^B = \mathbf{Y}^C = 0$, $\mu = 10^{-6}$, $max_\mu = 10^6$, $\rho = 1.1$, $\varepsilon = 10^{-8}$.

while not converged **do**

1. Fix the others and update $\mathbf{J}$

$\mathbf{J} = argmin\frac{1}{\mu}\|\mathbf{J}\|_* + \frac{1}{2}\|\mathbf{J} - (\mathbf{Z} + \frac{\mathbf{Y}^C}{\mu})\|_F^2$.

2. Fix the others and update $\mathbf{L}$

$\mathbf{L} = argmin\frac{\beta}{\mu}\|\mathcal{P}_\Omega(\mathbf{L})\|_1 + \frac{1}{2}\|\mathbf{L} - (\mathbf{Z} + \frac{\mathbf{Y}^B}{\mu})\|_F^2$.

3. Fix the others and update $\mathbf{Z}$

$\mathbf{Z} = (2\mathbf{I} + \mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T(\mathbf{X} - \mathbf{E} + \frac{\mathbf{Y}^A}{\mu}) + \mathbf{J} + \mathbf{L} - \frac{\mathbf{Y}^C}{\mu} - \frac{\mathbf{Y}^B}{\mu})$.

4. Fix the others and update $\mathbf{E}$

$\mathbf{E} = argmin\frac{\gamma}{\mu}\|\mathbf{E}\|_{2,1} + \frac{1}{2}\|\mathbf{E} - (\mathbf{X} - \mathbf{XZ} + \frac{\mathbf{Y}^A}{\mu})\|_F^2$.

5. Update the multipliers

$\mathbf{Y}^A = \mathbf{Y}^A + \mu(\mathbf{X} - \mathbf{XZ} - \mathbf{E})$, $\mathbf{Y}^B = \mathbf{Y}^B + \mu(\mathbf{Z} - \mathbf{L})$,

$\mathbf{Y}^C = \mathbf{Y}^C + \mu(\mathbf{Z} - \mathbf{J})$.

6. Update the parameter μ

$\mu = min(\rho\mu, max_\mu)$.

7. Check the convergence conditions

$\|\mathbf{X} - \mathbf{XZ} - \mathbf{E}\|_\infty < \varepsilon$, $\|\mathbf{Z} - \mathbf{J}\|_\infty < \varepsilon$, $\|\mathbf{Z} - \mathbf{L}\|_\infty < \varepsilon$.

end while