Feature-Preserving Surface Reconstruction From Unoriented, Noisy Point Data

J. Wang, 1 Z. Yu, 1 W. Zhu 2 and J. Cao 3

1 Computer Science Department, University of Wisconsin-Milwaukee, WI, USA
davis.wjun@gmail.com; yuz@uwm.edu
2 Department of Mechanical Engineering, Zhejiang University, Hangzhou, China
wdzhu@zju.edu.cn
3 School of Mathematical Sciences, Dalian University of Technology, Dalian, China
jjcao1231@gmail.com

Abstract
We propose a robust method for surface mesh reconstruction from unorganized, unoriented, noisy and outlier-ridden 3D point data. A kernel-based scale estimator is introduced to estimate the scale of inliers of the input data. The best tangent planes are computed for all points based on mean shift clustering and adaptive scale sample consensus, followed by detecting and removing outliers. Subsequently, we estimate the normals for the remaining points and smooth the noise using a surface fitting and projection strategy. As a result, the outliers and noise are removed and filtered, while the original sharp features are well preserved. We then adopt an existing method to reconstruct surface meshes from the processed point data. To preserve sharp features of the generated meshes that are often blurred during reconstruction, we describe a two-step approach to effectively recover original sharp features. A number of examples are presented to demonstrate the effectiveness and robustness of our method.

Keywords: unoriented noisy point data, surface reconstruction, robust statistics, feature-preserving reconstruction

ACM CCS: Computing methodologies, Computer graphics, Shape modeling, Point-based models

1. Introduction
Surface reconstruction from sampled points of real-world objects has a wide variety of applications in computer-aided design, computer vision and computer graphics. The definition of surface reconstruction is formulated as follows: given a point cloud sampled from the surface of an object, recover the original surface from which those points came. With the advance of scanning technology, 3D scanning devices become more affordable and popular to acquire point clouds from real-world objects. Current scanners are capable of producing large amounts of raw dense point data, which inevitably contain some level of noise and outliers. Consequently, the main difficulty in designing a surface reconstruction algorithm is to guarantee that the geometry and topology of the original surface are recovered consistently in the presence of the inherent noise of the acquired data. Furthermore, when the original surface contains sharp features, the necessity of being robust to noise is especially challenging since noise and sharp features are ambiguous, and most techniques are prone to blur important features or even amplify noisy samples [FCOS05].

In this paper, we present a robust surface mesh reconstruction method for noisy and outlier-ridden point data. Inspired by the widespread application of statistical methods in computer vision area, a kernel-based scale estimator is adopted to estimate the scale of inliers of the input point data. A series of best tangent planes are extracted for each point by using the mean shift clustering and adaptive scale sample consensus techniques. Consequently, all points have their corresponding tangent planes, including the outliers. For each point, the inliers are projected onto the corresponding tangent plane and the projected points are parameterized in the plane to generate a bitmap. The largest connected component is extracted in the bitmap. If the number of pixels within the largest connected component is lower than a threshold, the point is considered as an outlier. By performing this operation on all points, the outliers are detected and removed. For the rest of points, we estimate the normals and thereby dichotomize the feature points and the non-feature points. We denoise the feature and non-feature points, respectively, by projecting them onto their corresponding neighbouring piecewise smooth surfaces. Having this pre-processed point data with normals, one of existing surface reconstruction methods is...
employed to generate a surface mesh. Since the sharp features of the generated mesh are blurred frequently, a two-step feature-preserving approach is given to recover original sharp features. The pipeline is illustrated in Figure 1.

2. Related Work

Due to its broad applications, surface reconstruction from point data has been studied extensively over the years [HDD*92, ACK01, ABCO*01, BC00, KSO04, PMG04, OBS05, HK06, ACSTD07, MPS08, SMG10]. To reconstruct a smooth surface, an implicit surface construction technique is comprehensively used to interpolate or approximate the input point data equipped with normals, where the challenge is to choose an implicit surface function. Ohtake et al. [OBA*03] defined the multi-level partition of unity implicit surface as a shape representation, and thereby constructed surface models from point data with normals based on octree-subdivision. This method requires surface samples equipped with normals and is not stable in the presence of noise. Nagai et al. [NOS09] designed a reconstruction method from oriented point data based on smoothing partition of unity implicit surface, which is able to preserve sharp features pleasingly from the point data with a certain level of noise. Kazhdan et al. [Kaz05, KBH06] proposed the Fourier-based and Poisson surface reconstruction techniques, which are computationally efficient and resilient to noise. Similarly, Calakli and Taubin [CT11] introduced a variational formulation to design a smoothed signed distance surface reconstruction method. Generally, implicit function methods have the ability to handle a certain level of noise and outliers. Most of them are working on the oriented point data and rely on the assumption that the point data are sampled with smooth surfaces (i.e., without sharp features). However, if the assumption is not true, the erroneous reconstruction results would be yielded consequently.

In order to handle sharp features (e.g., edges and corners) from the point data, Fleishman et al. [FCOS05] extended the moving least-squares (MLS) surface to detect and reconstruct sharp features by partitioning the point set into multiple smooth regions. The surface is defined as the intersection of these smooth regions, which results in a piecewise smooth MLS surface capable of describing sharp features. However, when the ratio of signal to noise is reasonably low, the method would not be able to recover sharp features. In addition, it is inclined towards producing jagged sharp features. Oztireli et al. [OGG09] extended the MLS reconstruction to surfaces with sharp features using the kernel regression technique. The advantages of MLS and local kernel regression make this method robust in the presence of a certain level of noise. Salman et al. [SYM10] used the covariance matrices of Voronoi cells to extract a set of sharp features from the point data. They applied a feature-preserving variant of a Delaunay refinement process to construct a surface mesh containing a good representation of the extracted sharp features. This method works on the premise that the protecting ball can cover the gap between the extracted features and the reconstructed smooth surface. Avron et al. [ASGC010] introduced an $\ell_1$-spare approach for reconstruction of point set surfaces with sharp features. With these methods, sharp features of the objects are generally well preserved in face of a low level of noise and outliers; however, they have difficulties in obtaining satisfactory results in the presence of a reasonably high level of noise and outliers.

3. Robust Tangent Plane Detection

Prior to estimating normals, we detect the best local tangent planes for the input point data. Generally, the linear regression based on least squares is the most common method to fit a model from point data. However, for a point of real scanning data, its neighbourhood may be corrupted by some noise or gross outliers, or may contain more than one structure (i.e., pseudo-outliers). Consequently, it is fairly challenging to obtain the surrounding inliers of the point, from which the local tangent plane is faithfully approximated. As we know, a single outlier can be sufficient to force the least-squares estimator to produce an arbitrarily large value. To tackle this
problem, we adopt a robust statistical method to find the best tangent plane for each point, in which the non-parametric density and density gradient estimation techniques are exploited to estimate the scale of inliers, and thereby a modified random sample consensus strategy is used to obtain the best tangent plane. The noise scale estimation does not get affected by sharp features, and is able to achieve favourable results even in the presence of more than 80% outliers.

3.1. Noise scale estimation

In this section, we adopt a robust noise scale estimator [WS04], derived from kernel density estimation and mean shift techniques. Following the main idea of the statistical estimator, we take advantage of random sampling to obtain a local plane for each point so as to estimate the noise scale, respectively. Given an arbitrary point \( p \), its neighbourhood is searched, denoted by \( \text{Nbr}(p) \). After randomly choosing non-collinear 3-subset from \( \text{Nbr}(p) \), the three points determine a plane \( \theta \). We calculate the residuals (i.e. geometric distances) of all points in \( \text{Nbr}(p) \). Given a set of residuals \( R = \{r_1, \ldots, r_n\} \) (\( n \) is the number of points in \( \text{Nbr} \)), the kernel density estimator is defined as:

\[
\hat{f}(r) = \frac{1}{nh} \sum_{i=1}^{n} K \left( \frac{r-r_i}{h} \right),
\]

where \( K(\cdot) \) is a window or kernel function of width \( h \). We select the Epanechnikov kernel that yields the minimum mean integrated square error. The kernel is defined as:

\[
K(X) = \begin{cases} 
\frac{2}{3}c_d^{-1}(d+2)(1-X'X) & \text{if } (X'X \leq 1) \\
0 & \text{otherwise}, 
\end{cases}
\]

where \( c_d \) is the volume of a \( d \)-dimensional hypersphere of unit radius (i.e. \( c_1 = 2, c_2 = \pi \)). In our case, the kernel is applied on the residual, so we have \( d = 1 \).

To estimate the extrema of the kernel density \( \hat{f}(r) \), we compute the gradient of this density. Given the Epanechnikov kernel, the density gradient estimate is written as:

\[
\nabla \hat{f}(r) = \frac{n}{nh^2c_d} \frac{d+2}{h} \left( \frac{1}{n_t} \sum_{i \in \text{Nbr}(r)} (r-r_i) \right),
\]

where \( S_0(r) \) is a hypersphere of the radius \( h \), having the volume \( h^2c_d \), centred at \( r \), and containing \( n_t \) data points. Consequently, we can define the mean shift vector \( M_k(r) \) [CM02] as:

\[
M_k(r) = \frac{1}{n_t} \sum_{i \in \text{Nbr}(r)} (r-r_i).
\]

It has been observed that the mean shift vector points towards the direction of the maximum increase in the density [WS04]. Therefore, the optimal mode can be obtained using an iterative procedure:

\[
r_{k+1} = r_k + \omega \cdot M_k(r_k),
\]

where \( 0 < \omega \leq 1 \). The iteration terminates until convergence, that is,

\[
\frac{||M_k(r_{k+1})|| - ||M_k(r_k)||}{||M_k(r_{k+1})||} < \varepsilon,
\]

or the maximal iteration \( \tau \) is reached. By testing a number of models, \( \varepsilon = 0.012 \) and \( \tau = 250 \) generally produce good results.

Next, we need to determine the bandwidth \( h \). The kernel density estimation and mean shift method relate to the smoothness of the sample distribution. In 1D space, a simple over-smoothed bandwidth selector is recommended [TS85]:

\[
h = \left[ \frac{243R(K)}{35u_2(K)n} \right]^{0.2} s_k,
\]

where \( R(K) = \int_{-1}^{1} K^2(\xi) d\xi \), \( u_2(K) = \int_{-1}^{1} \xi^2 K(\xi) d\xi \) and \( K(\xi) \) is the Epanechnikov kernel function; \( s_k \) is a pre-estimation of the standard deviation. It is suggested using a generalization of the median estimator based on percentiles smaller than 50%:

\[
s_k = \frac{d_k}{\Phi^{-1}\left(\frac{1-\alpha}{2}\right)},
\]

where \( d_k \) is the half-width of the shortest window including at least \( k \) residuals; \( \Phi^{-1}[\cdot] \) is the argument of the normal cumulative density function; \( n \) is the total number of residuals. In our implementation, \( k \) is set 20% of \( n \), which is analogous to dependence on the 20% of data points with minimum error.

Next, we need to determine the bandwidth \( h \), the mean shift based procedure is run to find the optimal \( r^* \) such that all the points whose residuals fall in \([0, r^*] \) comprise the largest cluster of inliers. Subsequently, a least median squares estimator [RL87] is applied to the cluster in order to estimate the final inliers scale, which is given by:

\[
\sigma(\theta) = 1.4826 \left( 1 + \frac{5}{n-p} \right) \sqrt{\text{med}_p(R^2)},
\]

where \( n \) is the number of the sample points, \( p \) is the dimension of the parameter space (e.g. 3 for a plane) and \( \langle R \rangle \) is the set of residuals of sampling points related to the plane \( \theta \). The least median squares estimator is probably the most extended robust estimator due to its simplicity and robustness when the ratio of inliers is higher than 0.5. In this case, the least median squares converges to a solution since the cluster only includes inliers.

3.2. Tangent plane detection

According to random sample consensus theory, let \( P \) be the probability to get a clean subset, the number of samples to draw in order to find the clean subset is:

\[
m = \frac{\log(1 - P)}{\log(1 - (1 - \eta)^p)}.
\]

where \( \eta \) is the ratio of outliers contained in the whole point set; \( p \) is the size of a subset to compute a minimal parametric model (e.g. 3 for a plane).

We assume that when the best tangent plane is correctly detected, two conditions should be satisfied:
Figure 2: Bitmap generation and largest component extraction. (a) Point cloud with the point of interest (in red); (b) the neighbouring points; (c) one of the tangent planes; (d) point projection onto the tangent plane; (e) the generated bitmap. The number of each pixel in the bitmap stands for the number of points projected onto the corresponding grid. There are seven connected components, and the largest connected component contains 131 points.

1. There are as many as possible geometrically connected data points near or on the tangent plane. This condition is based on the observation that point data which are the inliers of a ‘real’ plane cluster are contiguous in space.

2. The residuals of inliers should be as small as possible. That is, the scale of inliers should be as small as possible.

Then, the score function of a tangent plane \( \theta \) around \( p \) is defined as:

\[
S_{\text{Nbr}}(\theta) = \frac{|CC_{\text{max}}(\theta)|}{\sigma(\theta)},
\]

where \( |CC_{\text{max}}(\theta)| \) is the largest connected component of inliers with 8-neighbour topology, and \( \sigma(\theta) \) is the scale of inliers.

The largest connected component is extracted as follows. Given a point \( p \), its neighbouring points \( \text{Nbr} \) and a tangent plane \( \theta \), we project the inliers of \( \theta \) onto \( \theta \) and parameterize the projected points to generate a bitmap. A pixel in the bitmap is set if a point is projected onto it. Meanwhile, each pixel is associated with a number, which indicates the number of projected points in this pixel. Ideally, the size of the pixels in the bitmap corresponds to the distance between neighbouring points in \( \text{Nbr} \), that is, the sampling resolution. However, if the data are sampled non-uniformly, we choose the average value of the distances between neighbouring points in \( \text{Nbr} \). Based on the bitmap, connected components are computed and the largest one is extracted. Figure 2 gives an example of bitmap generation and largest connected component extraction.

Thus, the best tangent plane \( \hat{\theta} \) is obtained by solving this objective function:

\[
\hat{\theta} = \theta \arg\max(S_p).
\]

Based on the aforementioned techniques, we first apply the mean shift clustering to find the bounds of an inlier cluster. Then, we adopt the least median squares estimator to estimate the scale of the inlier cluster. Finally, the score function is computed to search the best tangent plane. Specifically, for each point \( p \), the best tangent plane is extracted by the following robust algorithm:

---

Algorithm 1. Detect Best Tangent Plane

1. \( \text{Nbr} \leftarrow \text{neighborhood of } \ p \)
2. for \( i = 0 \) to \( m \)
   1. \( \theta \leftarrow \text{plane fitted from 3-subset } \in \text{Nbr} \)
   2. \( r \leftarrow \text{residual}(\text{Nbr}, \theta) \)
   3. Compute the bandwidth \( h \)
   4. Initialize \( r_0 \) with 0
   5. repeat
      1. \( r_{k+1} \leftarrow r_k + \theta \cdot M_h(r_k) \)
      2. \( \sigma(\theta) \leftarrow 1.4826 \left( \frac{1 + \frac{5}{|\text{Nbr}| - 3}}{\text{med} R^2} \right) \sqrt{\text{med} R^2} \)
      3. \( S_{\text{Nbr}}(\theta) \leftarrow \frac{|CC_{\text{max}}(\theta)|}{\sigma(\theta)} \)
      4. \( \hat{\theta} \leftarrow \arg\max S_{\text{Nbr}}(\theta) \)
   6. until convergence \( \Rightarrow r^* \leftarrow r_{k+1} \)
   7. \( R \leftarrow \bigcup_{0}^{\infty} r \)

---

This detection procedure, referred to as Adaptive Scale SAmple Consensus, is similar to RANSAC. Because no prior knowledge concerning the scale of inliers is necessary (the scale estimation is data driven), the Adaptive Scale SAmple Consensus is an important improvement over RANSAC. It is highly robust to heavily corrupted data with multiple structures and discontinuities. Empirically, it can tolerate more than 80% outliers.

4. Outlier Removal and Noise Smoothing

4.1. Outlier removal

By running the above-mentioned algorithm, all points have corresponding tangent planes, even for the outliers, which should be removed eventually. We check the highest score function for each point. If the score is lower than a threshold, the point is regarded as an outlier. Accordingly, outliers are detected and removed.
4.2. Normal orientation

After removing outliers, we assign as its normal the unit normal of its best tangent plane for each remaining point. However, we should notice that the the normals are well estimated, while they are not oriented for the lack of inside and outside information in the input point data. Therefore, the normals need to be oriented prior to the following surface reconstruction. Here, we adopt a globally geometric method [CHL*11] to orient the normals. Specifically, the constrained Laplacian smoothing is performed first on the point cloud, resulting in a contracted point cloud. Then the visibility confidence of the original and contracted point clouds are estimated by voting from multiple viewpoints. Based on the observation that the confidence is increased if the original point is visible, the normals initially estimated from the best tangent plane are flipped based on the contracted vectors and estimated confidence. Next, a Laplacian smoothing is applied twice to further adjust the orientation of points with zero or lower confidence. As a result, the normals are oriented consistently for the point clouds with a reasonable high level of noise.

4.3. Noise smoothing

We attempt to filter out noise. If a point is located at an edge or corner of the point cloud, it is defined as a feature point; otherwise it is a non-feature point. Then, two different surface fitting and projection strategies are applied to those two types of points, respectively, to smooth noise, while preserving genuine features in the original point cloud.

4.3.1. Feature point detection

In Algorithm 1, the best tangent plane is extracted. Here, we keep a list of the best M tangent planes instead. Specifically, for each point p, we keep the best 30 random planes (according to the score function in Equation 11) and sort them with the decreasing scores, denoted by PlnList(p) = [θ1, θ2, ..., θ30], and use them as candidates for the best M tangent planes. In our implementation, M = 4. Let BestPlnList(p) be the list of the best M tangent planes, initialized with ø, we use the following method to detect if p is a feature or non-feature point:

**Algorithm 2.** Is Feature Point

BestPlnList(p) ← θ1
for each θi ∈ PlnList(p)
    if \( \frac{SSim(θ_i)}{SSim(θ_j)} > 0.75 \)
        then do
            for each θj ∈ BestPlnList(p)
                if n0, n0 < T(e.g.:0.4)
                    then BestPlnList(p) ← θi
        if |BestPlnList(p)| > 1
            then return (true)
        else return (false)

4.3.2. Surface fitting and projection

According to the surface theory, the local geometry around a surface point can be regarded as a height field over its tangent plane. To faithfully characterize the local geometric properties in this field, surface patches with at least second-order are required. Therefore, we choose quadratic surfaces to approximate the local geometry around a surface point.

For each non-feature point, we seek the underlying surface faithfully representing its local geometry, and project the point onto the surface to carry out smoothing. Prior to fitting the underlying surface, the associated neighbourhood of the point needs to be searched, which is implemented via the normal-based region growing technique. Starting from a non-feature point p with the normal n, its neighbouring points NP(p) are searched within a bounding sphere. Within NP(p), we search the point q which satisfies two conditions: the normal \( \mathbf{n}_q \) of q is closest to \( \mathbf{n} \) and the angle between \( \mathbf{n}_q \) and \( \mathbf{n} \) is lower than a threshold, and then add it into the neighbourhood Nbr(p) of p, initialized with p. The neighbourhood keeps growing until convergence, that is, the number of the points in Nbr(p) reaches a threshold (typically set as 15), or no more point can be added. The points within the neighbourhood Nbr(p) are used to fit a quadratic surface S. Then, we project p onto S and update it with the projection point.

For a feature point, it is usually defined by multiple surface patches. Thus, to preserve features requires fitting a number of surface locally. This is challenging because it needs to identify discontinuities or the locus of the intersection of many local smooth surfaces in the presence of a certain level of noise. To address this issue, we partition the neighbourhood of the point into a certain number of segments and fit them with quadratic surface patches, followed by projecting the point onto the intersection of patches to update it. Here, we take advantage of the number of BestPlnList(p) (i.e. |BestPlnList(p)|) to determine the number of segments. Each plane in BestPlnList(p) has an associated largest connected component, that is, a list of points. Among those points, we search the one whose normal vector is closest to the plane normal. Using this point as a seed, the region growing process is performed to search its neighbourhood, which is then fitted with a quadratic surface. As a result, the feature point has |BestPlnList(p)| associated quadratic patches.
surfaces. Having those quadrilateral surfaces, we project the point onto those surfaces to get the projection points, and use the average of all projection points to update the point. Figure 3 gives neighbouring surface patches of non-feature and feature points in a block model. Figure 4 shows the result of outlier removal, noise smoothing and normal estimation from the noisy point data of \textit{octa}.

To demonstrate the effectiveness of normal estimation, we segment the point data after processing according to the normal dissimilarity criterion. The region growing process is performed to carry out the segmentation. Specifically, an arbitrary, un-segmented point is chosen as the first seed, and the region is initialized with this seed point. The region is growing by adding one of the neighbouring points of the seed such that the normal vector angle between the added point and the seed point is smallest, and less than a predefined threshold. Then, the seed is updated with the latest added point and the region growing is iterated until no more points could be added into the current region. This region is referred to as a segment. By repeating the above growing process until all points are segmented, the point cloud segmentation is completed. Each segment is assigned with a different colour.

5. Feature-Preserving Mesh Reconstruction

After removing noise and outliers, several surface reconstruction approaches can be exploited to generate a triangular mesh as introduced in Section 1. With those methods, a common issue is that sharp features within the original model are frequently blurred. To recover those sharp features, an iterative, two-step approach is presented here, in which the vertex normals of the mesh are filtered bilaterally and the vertex positions are thereby updated iteratively. First, we design an effective bilateral filter to smooth vertex normals of the mesh. Given a triangular face \( f \) with a normal \( \mathbf{n} \) and a centroid \( \mathbf{c} \), the bilateral filtered normal \( \bar{n} \) of \( f \) is defined by:

\[
\bar{n} = \frac{\sum_{j \in N(i)} W_f(|\mathbf{c}_j - \mathbf{c}_i|) W_f(\mathbf{n}_j, \mathbf{n}_i) \mathbf{n}_j}{\sum_{j \in N(i)} W_f(|\mathbf{c}_j - \mathbf{c}_i|) W_f(\mathbf{n}_j, \mathbf{n}_i)},
\]

where \( N(i) = \{ j : ||\mathbf{c}_j - \mathbf{c}_i|| < \rho \} \) is the neighbouring face set of \( f \), and \( \mathbf{n}_j \) is the normal of the face \( f_j \) within \( N(i) \). \( W_f(\mathbf{n}_j, \mathbf{n}_i) \) is represented by:

\[
W_f(\mathbf{n}_j, \mathbf{n}_i) = \begin{cases} 
0 & \text{if } (\mathbf{n}_i \cdot \mathbf{n}_j) \leq \lambda \\
[(\mathbf{n}_i \cdot \mathbf{n}_j) - \lambda]^4 & \text{otherwise,}
\end{cases}
\]

where \( \lambda \) is a threshold set within \([0, 1]\). Essentially, the normals are truncated if they diverge from \( \mathbf{n} \) to a certain degree. That is, this filter ignores heavy noise and thus is insensitive to a high level of noise.

After filtering the face normals, we update the vertex positions based on these normals. Taubin [Tau02] applied orthogonality between the normal and edge vectors of the face to generate the linear equation system for updating vertex position:

\[
\begin{align*}
\mathbf{n}_f \cdot (\mathbf{v}_i - \mathbf{v}_j) &= 0, \\
\mathbf{n}_f \cdot (\mathbf{v}_j - \mathbf{v}_k) &= 0, \\
\mathbf{n}_f \cdot (\mathbf{v}_k - \mathbf{v}_i) &= 0,
\end{align*}
\]

(15)

Since there is a non-trivial solution to this equation system, Taubin [Tau02] proposed to find its least-squares solution, that is, to minimize the following error function defined on mesh:

\[
E(V) = \sum_{f \in \partial f} \sum_{i,j \in f} ||\mathbf{n}_f \cdot (\mathbf{v}_i - \mathbf{v}_j)||^2,
\]

(16)

where \( \partial f \) is the set of edges of face \( f \), and \( \mathbf{n}_f \) is the updated normal of \( \mathbf{n}_f \). Solving the optimization problem gives rise to the updated position \( \mathbf{v}_i' \) of \( \mathbf{v}_i \):

\[
\mathbf{v}_i' = \mathbf{v}_i + \lambda \sum_{j \in NV(i), f \in NF(i,j)} \frac{[\mathbf{n}_f'(\mathbf{n}_f' \cdot (\mathbf{v}_i - \mathbf{v}_j))]}{||\mathbf{e}_f - \mathbf{v}_i||}
\]

(17)

where \( NV(i) \) is the set of vertices connected with \( \mathbf{v}_i \); \( NF(i, j) \) is the set of faces sharing the edge \( \mathbf{e}_{ij} \); and \( \lambda > 0 \) is the iteration step size. To choose \( \lambda \) is non-trivial. We introduce a more effective error function which takes into consideration the centroids of neighbouring faces. Minimizing the error function yields the updated vertex position as:

\[
\mathbf{v}_i' = \mathbf{v}_i + \sum_{j \in NV(i), f \in NF(i,j)} \frac{||\mathbf{e}_f - \mathbf{v}_i||}{\sum_{j \in NV(i), f \in NF(i,j)}} \cdot [\mathbf{n}_f' \cdot (\mathbf{n}_f' - (\mathbf{v}_i - \mathbf{v}_j))].
\]

(18)

Applying this two-step smoothing method to the mesh, the sharp features in the original mesh (i.e. the input point cloud) can be well recovered. In particular, if the mesh is not close, the boundaries are considered features automatically. Meanwhile, the bumpy faces

\[\text{Figure 4: The processing result of the noisy octa point data. (a) Original point data; (b) noisy point data (noise: 2%); (c) extracted feature points; (d) point data after noise smoothing and normal estimation; (e) point cloud segmentation. From the results, the noise is well handled and the normals are reliably calculated.}\]
Figure 5: Reconstruction results of the oil pump model. (a) Noise point data (noise: 3.5%); and the reconstruction results from (b) SMPU \text{[NOS09]}, (c) scattered \text{[OBS05]}, (d) Fourier \text{[Kaz05]}, (e) SSD \text{[CT11]}, (f) Poisson \text{[KBH06]} and (g) our method. The second row gives the processed point data and the corresponding reconstruction results. The third row shows the mean curvature distributions of the results of the second row. From the corresponding reconstruction results between the first and second rows, our point processing approach can produce favourable inputs to general surface reconstruction methods. In addition, SMPU \text{[NOS09]} and Scattered \text{[OBS05]} can also preserve some sharp features; however, some artifacts are generated around sharp edges. Comparatively, our feature-preserving method achieves better results.

Figure 6: Reconstruction results of the daratech model. (a) Noisy point data (noise: 3%); and the reconstruction results from (b) RIMLS \text{[OGG09]}, (c) scattered \text{[OBS05]}, (d) SMPU \text{[NOS09]}, (e) Poisson \text{[KBH06]} and (f) our method. The second row gives the processed point data and the corresponding reconstruction results. The mean curvature distributions of the results from the second row are presented in the last row. Our pre-processing approach yields good inputs for those reconstruction methods. Meanwhile, sharp features are better preserved with our feature-preserving method.

around the boundaries can still be smoothed. If the mesh is non-manifold, the topological relationship is corrupted around the non-manifold area. Consequently, the features around the non-manifold area may not be smoothed and sharpened.

6. Results and Discussions

All algorithms described have been implemented and run on a PC with 1.8 GHz CPU and 2 GB RAM. Our implementation does not take advantage of the very parallelizable nature of some of the stages,
and doing so should increase the efficiency. In our implementation, we adopt the Poisson surface reconstruction method [KBH06] to generate an initial triangular mesh but any surface reconstruction method may be used. We have tested our algorithm on a variety of unorganized, unoriented point cloud data with either raw or synthetic noise; outliers for analysis of the effectiveness of our method. The synthetic noise is made by a zero-mean Gaussian function with standard deviation proportional to the diagonal length of the bounding box of the input point cloud. The synthetic outliers are generated randomly in the bounding box of the input point cloud.

6.1. Parameters

In our algorithm, there are several parameters: (1) the neighbourhood size $N$ for normal estimation; (2) the normal difference threshold $\lambda$; (3) the iterations of normal smoothing $n_1$; and (4) the iterations of vertex updating $n_2$. Among the parameter, the neighbourhood size is relevant to the density of the input point cloud. If the point cloud is dense, the size could be a big value; otherwise, it should be small. The normal difference threshold depends on the sharpness of features in the original models. While smaller normal difference
Figure 10: Reconstruction results of the buddha, cat, terra-cotta shiisa and statue models.

Figure 11: Reconstruction results of the octa model in Figure 4. (a) The original mesh; (b) the noisy point data (noise: 2%); (c) our reconstruction result; (d) the histogram of normal deviations (in radians), where the horizontal axis is the normal deviation between the reconstructed mesh and the original mesh, and the vertical axis is the percentage to each deviation value; (e) the histogram of Hausdorff distance, where the horizontal axis is the error between the reconstructed mesh and the original mesh, and the vertical axis is the percentage to each error value. Note that the original and reconstructed meshes are scaled within a unit bounding box. The maximal normal deviation is 0.6532 and there are more than 90% points whose normal deviations are lower than 0.1876. The maximal position error is 0.0074 and there are more than 90% points whose position errors are less than 0.0018.

threshold yields lower normal errors for models with smooth surfaces, for models with sharp edges, too small a value of the threshold results in a big error. Therefore, it should be relatively bigger for surfaces with sharp edges, while smaller for smooth surfaces. The iterations of normal smoothing and vertex updating rely on the ‘feature-blurring’ degree of results from typical surface reconstruction methods. If the sharp features are extensively blurred, the iterations should be set with a big value; otherwise, they should be small. Based on a number of experiments, we have the following typical settings in our implementation: \( N \in [150 - 200] \); \( \lambda \in [0.2 - 0.6] \); \( n_1 \in [5 - 25] \) and \( n_2 \in [5 - 20] \).

6.2. Comparisons with previous methods

To demonstrate the effectiveness of our method, we compare it with eight related methods, that is, Fourier [Kaz05], RMLS [FCOS05], Poisson [KBH06], APSS [GG07], Scattered [OBS05], Smoothed MPU (SMPU) [NOS09], DelaunayRef [SYM10], \( \ell_1 \)-Sparse [ASGCIO10] and SSD [CT11]. We implement \( \ell_1 \)-Sparse [ASGCIO10] and RMLS [FCOS05] ourselves while the executables of all other methods are from the corresponding authors. First, we run all those methods on the input data with noise and outliers, where normals of point cloud data are estimated by the combined approach [HDD+92]. Then, our algorithm is adopted to remove outliers, estimate normals and smooth noise for the point data. Based on the processed point data, those methods are executed again to obtain the corresponding results.

Figure 5 shows the reconstruction results of the oil pump model from different methods. The noisy point data are processed by removing outliers, smoothing noise and estimating normals with our point processing method. Apparently, all those methods achieve better results from the processed point data than those from the original noisy data. It suggests that our point processing approach can produce favourable inputs to general surface reconstruction methods. Fourier [Kaz05] and Poisson [KBH06] are able to deal with a certain level of noise thanks to the characteristic of implicit surface approximation, where the iso-surface extraction is performed similar to the Marching Cubes, which essentially is incapable of handling sharp features. SMPU [NOS09] fits small subsets of the point data separately and blends them together by using partitions of unity. Since the blending tends to preserve the fitted geometry in each cell, sharp features can be detected and retained without sacrificing smoothness in other regions. However, it is fairly sensitive to noise due to its local feature. We notice that most of sharp features are preserved well from the results of SMPU [NOS09] and Scattered [OBS05], while some jagged triangular faces are yielded around sharp edges. Comparatively, our output is more satisfactory. In Figure 6, we present the mean curvature distributions of reconstruction
Figure 12: Robustness to noise and outliers. (a) Original noise-free point data; (b) point data with noise (noise: 3%); point data with outliers: (c) 10%, (d) 40% and (e) 75%. The second row shows the corresponding reconstruction results from our method. The third row gives the mean curvature distributions of reconstruction results. The Hausdorff distance distributions are presented in the last row, where all models are scaled within a bounding box of \([0, 0, 0] \sim [5, 5, 5]\). As noise is added, the output degrades gracefully, still capturing all of sharp features; even after adding 10%, 40% or even 75% of outliers, the reconstruction remains of high quality in terms of the recovery of sharp features and reconstruction accuracy.

Figure 13: Data processing results of 2D point data with different levels of sub-sampling, noise and outliers. S%, N% and O% are the percentages of sub-sampling, noise and outliers, respectively. Originally, there are 2000 points, which distribute non-uniformly in 5 lines, i.e. 148, 353, 401, 536 and 562 points, respectively. The numbers of inliers and outliers are (a) (2000, 2445); (b) (2000, 10000); (c) (2000, 13334); (d) (1000, 1223); (e) (1000, 5000); (f) (200, 245); (h) (200, 1000); (i) (200, 1334). Note that we replace plane with line during processing. From the results, our method is able to tolerate 80% outliers, even though the point data is sub-sampled non-uniformly.
Figure 14: Data processing results of 3D point data with different levels of sub-sampling, noise and outliers. S%, N% and O% are the percentages of sub-sampling, noise and outliers, respectively. From the result, we notice that the method may fail, if the point data is quite sparse and the outliers increase to 90%.

Table 1: Timing comparison of surface reconstruction.

<table>
<thead>
<tr>
<th>Model (Input points)</th>
<th>Method (Parameters)</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Normal estimation</td>
<td>Mesh generation</td>
</tr>
<tr>
<td>oil pump p: 31031</td>
<td>SMPU (0.01,9,7,250)</td>
<td>8.847</td>
</tr>
<tr>
<td></td>
<td>Scattered (1e-005,3)</td>
<td>6.524</td>
</tr>
<tr>
<td></td>
<td>Fourier (250)</td>
<td>3.619</td>
</tr>
<tr>
<td></td>
<td>SSD (9)</td>
<td>66.411</td>
</tr>
<tr>
<td></td>
<td>Poisson (9)</td>
<td>7.745</td>
</tr>
<tr>
<td></td>
<td>Our (120,0.5,15,15)</td>
<td>34.444</td>
</tr>
<tr>
<td>daratech p: 171198</td>
<td>APSS (4,0,0001,350)</td>
<td>37.857</td>
</tr>
<tr>
<td></td>
<td>Scattered (1e-005,3)</td>
<td>12.541</td>
</tr>
<tr>
<td></td>
<td>SMPU (0.01,9,7,300)</td>
<td>15.586</td>
</tr>
<tr>
<td></td>
<td>Poisson (9)</td>
<td>10.849</td>
</tr>
<tr>
<td></td>
<td>Our (150,0,55,10,15)</td>
<td>155.790</td>
</tr>
<tr>
<td>part p: 101168</td>
<td>Our (200,0,55,10,10)</td>
<td>114.547</td>
</tr>
<tr>
<td>soldier p: 145151</td>
<td>131.589</td>
<td></td>
</tr>
<tr>
<td>shiisa p: 338010</td>
<td>Our (200,0,35,15,10)</td>
<td>304.547</td>
</tr>
<tr>
<td>statue p: 497419</td>
<td>Our (200,0,3,10,10)</td>
<td>424.795</td>
</tr>
</tbody>
</table>

7. Conclusions

We present an effective method to reconstruct surface mesh from unoriented, noisy and outlier-ridden point data based on the robust statistical approaches. The adaptive scale sample consensus estimator is exploited to estimate the scale of inliers. This estimator is very robust to outliers and point data with multiple structures, being able to tolerate more than 80% outliers. Accordingly, our normal estimation method is robust to noise and outliers. The point processing and normal estimation algorithms produce favourable results that can be
readily used as inputs of the existing surface reconstruction methods. To further preserve sharp features, we give an efficient, two-step method to recover sharp features based on bilateral filtering, while smoothing bumpy mesh.

As shown in Section 3, the score function is defined on the largest connected component of the point to detect the tangent planes. If the neighbouring points belonging to the same underlying surface are too coarse, it is likely that we are unable to find the largest connect component, that is, the maximal number of neighbouring points sufficiently close to the same underlying surface, which, consequently, leads to a ‘false-positive’ tangent plane. Therefore, how to find the best tangent plane in the presence of very coarse data is worthy to be studied in the future work.

Acknowledgement

We would first like to thank the anonymous reviewers for their valuable feedback. We are grateful to Yutaka Ohtake, Michael Kazhdan, Yukie Nagai and Fatih Calakli for providing the executable files. The testing results of APSS and RIMLS are generated from MeshLab. We thank Matt Berger, Yutaka Ohtake and Yukie Nagai for offering point data. Additionally, some of the point data are from the AIM@shape repository.

References


