MVA2013 IAPR International Conference on Machine Vision Applications, May 20-23, 2013, Kyoto, JAPAN

Automatic Sharp Feature Extraction from Point Clouds with Optimal Neighborhood Size

Trung-Thien Tran, Sarah Ali and Denis Laurendeau
Computer Vision and System Laboratory, Laval University
1065, avenue de la Medecine, Quebec, G1V-0A6, Canada
trung-thien.tran.1@ulaval.ca, denis.laurendeau@gel.ulaval.ca

Abstract

A novel algorithm is proposed for extracting sharp features automatically and at optimal scale from point clouds. First, the vector between a given point and the centroid of its neighborhood at a given scale is projected on the normal at this point. This projection is called the 'projected distance' at this point. The projected distance and surface normal vector are recalculated at several scales for each point. In a second stage, the projected distance at different scales is analyzed in order to choose the optimal neighborhood size and update the final projected distance value for the point. Finally, Otsu's method is applied to the histogram of the final projected distances on the cloud in order to select the optimal threshold value which determines whether points are on a sharp feature or not. The method has many advantages such as automatic selection of threshold, optimal neighborhood selection, accurate and robust detection of sharp features on a wide variety of objects. To demonstrate the robustness of the method, it is applied on both synthetic and complex point clouds with different noise levels.

1 Introduction

Digital scanning devices have been used for various applications. Due to the rapid development of scanning technologies, very large sets of accurate 3D points can be collected with such devices. Therefore, more and more applications use these sensors, especially in industrial manufacturing. Among emerging problems, sharp feature extraction from scanned data has recently received much attention from the research community because this operation is very important for many operations such as segmentation [1] and surface reconstruction [2]. In fact, most manufactured objects consist of the combination of common geometric primitives and the intersections between these primitives can be considered as sharp features.

The existing methods can be classified into two categories: mesh-based and point-based methods. Point-cloud processing algorithms [3, 4] have recently achieved some promising results, and this data type preserves the original structure of the underlying surface. Generally, sharp features are found at the points that have a large variation in curvature [5] or discontinuities in surface normal orientation [6]. In addition, many other techniques preserve sharp features during surface reconstruction. The approach proposed by Daniels et al. [7] used Robust Moving Least Square extended from MLS to preserve sharp features extracted from noisy data. Nevertheless, these methods for local surface reconstruction are computationally expensive. PCA-based feature extraction methods [8, 9] are popular approaches. In [9], Pauly et al. calculate surface variation using the PCA technique. However, the method requires that threshold values be selected for weighted estimation and global threshold value for feature extraction. Therefore, this paper proposes a new algorithm for extracting sharp features automatically from point clouds with optimal neighborhood size selection and automatic threshold selection.

The rest of this paper is organized as follows. The sharp feature extraction algorithm is described in Section 2. Results and discussion are presented in Section 3, while Section 4 draws some conclusions on the proposed method.

2 Proposed Sharp Feature Extraction Algorithm

A flow diagram of our method is shown in Figure 1. The 3D data is scanned from the object surface and yields an unstructured point cloud $p_i \in R^3$, i=1...N; N is the number of points. Each block is described with more details in the following.

2.1 Projected distance

Projected Distance: This value expresses the structure of the underlying surface supported by the point cloud. The projected distance $DIS(\sigma, i)$ given by Equation 1, is calculated by projecting the distance, from point p_i to its local data centroid $\overline{p_i}$ (the left of Figure 2), along the normal orientation obtained by [10].

$$DIS(\sigma, i) = abs(\overrightarrow{p_i - \overline{p_i}}.\overrightarrow{n_i}) = abs(\|\overrightarrow{p_i - \overline{p_i}}\|.\cos\theta_i) \quad (1)$$

where θ_i is the angle between vector $\overrightarrow{p_i - \overline{p_i}}$ and vector $\overrightarrow{n(\sigma, i)}$, which is the surface normal at point p_i. σ is the neighborhood size and i is the index of the point. Angle θ_i between vector $\overrightarrow{p_i - \overline{p_i}}$ and vector $\overrightarrow{n_i}$ can be greater or smaller than 90 degrees, but the projected distance is always a positive number. That is why the direction of the normal vector is not considered here as it is in [10], only the orientation is used in our work.

The basic principle of our method is that the projected distance is almost zero for points lying on a smooth point of the surface. However, the projected distance has a large value if the point is on or close to a sharp feature. Our method uses this projected distance to assess whether the point is located on a sharp feature or not.

To prove this property, three points are chosen from the fandisk model, shown in Figure 2. These points are located on a smooth, near-sharp and sharp area, respectively. Projected distance values for the three

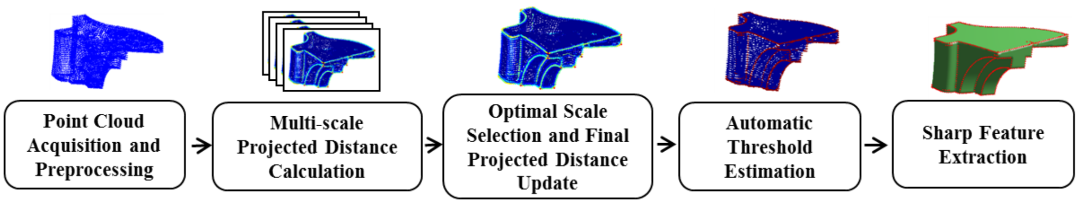

Figure 1. Overview of proposed method.

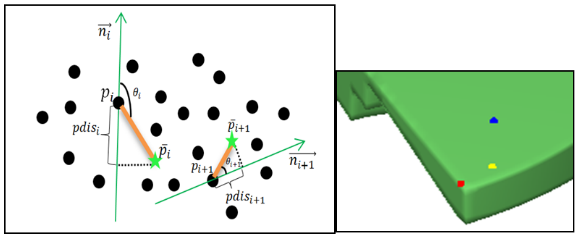

Figure 2. Projected distance estimation procedure.

Table 1. Multi-scale projected distances at three points.

Scale Size	Smooth	Near-sharp	Sharp
8	1.75E-19	1.75E-19	0.518
12	2.78E-19	2.78E-19	0.683
16	2.17E-19	0.028	0.554
20	9.62E-20	0.023	0.621
24	8.92E-20	0.038	0.622
28	2.46E-19	0.076	0.676
32	3.08E-19	0.089	0.668

points are calculated at multiple scales k=(8,12,...32) and shown in Table 1. From the table, we observe that all the projected distances of the smooth point (blue) have a small value close to zero. The values at a near-sharp point (yellow) are small and less than 0.1, while they are much larger at a sharp point (red).

2.2 Multi-scale normal vector and projected distance calculation

If the normal vector and projected distance were computed for a fixed neighborhood size, the result would not be accurate. For a small σ, the normal vector estimation is not stable due to noise. As the neighborhood size increases, the normal vector and the projected distance of points near an edge will change greatly because of the smoothing that occurs near discontinuities. Therefore, we have implemented a multi-scale approach for computing the surface normal vector and the projected distance, which is shown in Table 1.

2.3 Optimal neighborhood size selection

In this section, projected distance values at multi-scales σ for each point are analyzed in order to select optimal neighborhood size. Three points are chosen at the surface of the fandisk shown on the right in Figure 2.

The "rate of change" value, which is obtained by dividing the projected distance at the larger σ by the one at the smaller σ, is defined and shown in Figure 3.

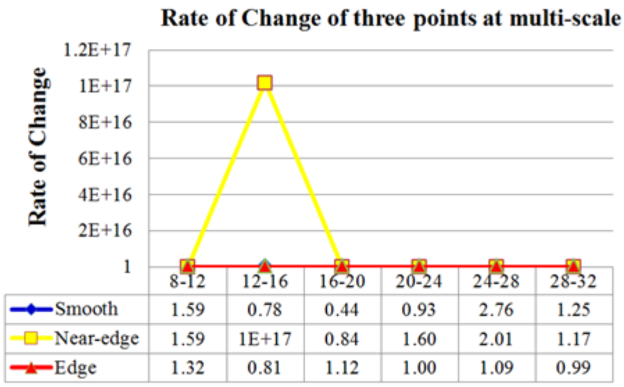

Figure 3. Rate of change of three points at multi-scale.

On smooth regions and edge regions on the surface, the rate of change does not change much with increasing neighborhood size and stays close to 1 (red line and blue line hidden under red one in Figure 3). In this case, any scale can be considered as being optimal for this point, but $\sigma = 16$ is chosen as the best and is used in some papers [11].

At the point close to an edge, the rate of change varies greatly and rises up to 10^{17} (yellow line in Figure 3); it jumps significantly between neighborhood sizes 12 and 16. In this case, $\sigma = 12$ is considered as the optimal neighborhood size for this point. From this analysis, we propose an algorithm (Figure 4) for choosing the optimal scale.

In the Figure 4, s is the number of rate of change. ε expresses the sudden change in distance between two adjacent scales and does not affect the performance

```
for i=1:N                  % each point data
    rate=zeros(s,1);       % initial rate of change
    for j=1:s              % multi-scale
          rate(j,1)=DIS(σ_{j+1},i)/DIS(σ_j,i);
    end
    peak = find (rate > ε);
    if (peak == empty)
          optimal_σ =16;
    else  optimal_σ = σ_peak
end
```

Figure 4. Pseudo code for optimal neighborhood selection.

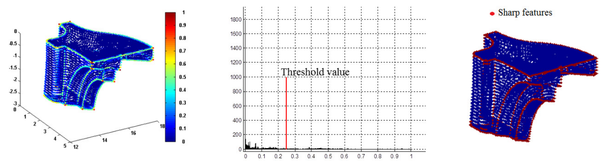

Figure 5. Sharp feature extraction process.

much as it changes little in value. *peak* is the position at which a sudden change occurs.

The above algorithm provides the optimal neighborhood size for each point, so the normal vector and the projected distance are computed reliably at each point. The projected distance at that optimal size is taken as the final value for this parameter.

2.4 Automatic threshold selection

Once the final value of the projected distance is obtained, we need to determine the optimal threshold value to extract sharp features from the point cloud. In our paper, the well-known Otsu's method [12] is applied.

The final projected distance is normalized in the interval [0, 1] shown on the left of Figure 5. Blue dots correspond to smooth regions on the underlying surface, while green and red dots correspond to potential edges and corners on the underlying surface. Final projected distance values are rearranged in order of increasing value between 0 and 1, and the histogram is created as the input for Otsu's method (middle of Figure 5). The threshold value is located at the red line (middle of Figure 5). The results are shown on the right of Figure 5.

3 Experimental Results

To illustrate the efficiency of our method, we have tested it on complex models corrupted by different noise levels. Moreover, the results provided by our method are compared with some other methods such as mean curvature, normal and Pauly's method [9].

3.1 Results obtained on different objects with complex shapes

As mentioned previously, differential geometry information (ie. curvature and normal) is often used in

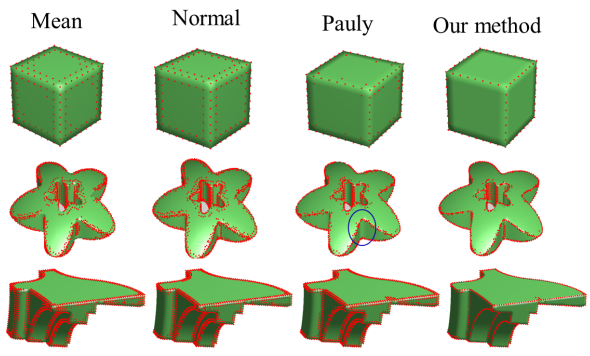

Figure 6. Results for various point cloud datasets.

feature extraction. Therefore, mean curvature and normal difference between points are calculated. Pauly et al. [9] used surface variation, which is calculated as the division of the smallest eigenvalue by the sum of all eigenvalues of its neighborhood as the criteria for feature extraction. To show the advantage of using the projected distance proposed here, our automatic threshold selection approach is applied for Mean Curvature, Normal and Pauly's methods because these methods require that this threshold value be chosen by the user. The output uses graphics rendering and sharp features are displayed as red dots.

Observing Figure 6 clearly indicates that the method proposed can detect edge features very sharply. Mean curvature, normal and Pauly's methods rather detect thick features including many potential feature points. Even Pauly's method fails to detect the feature at the bottom of the trim-star (blue circle in Figure 6).

Thus, our method provides better results due to optimal neighborhood selection, so that reliable normal vectors are computed. The projected distance for smooth and near-edge points is much smaller than that of sharp features. Hence, the feature lines are thinner and more accurate.

Figure 7 shows the results obtained by manually tuning different threshold values for Pauly's method. With small threshold values (0.15 or 0.2), the results consist of thick sharp feature lines. With large threshold values (0.25 or 0.3), the approach misses some part of the sharp features (red ellipse in Figure 7) and also contains wrong feature points (blue circle in Figure 7). However, the results from our proposed method are accurate and clear even without tunning the threshold values. More results for complex models are shown in Figure 9.

3.2 Results for different noise levels

To illustrate the robustness of our method to noise, some results for a noisy fandisk are presented. Gaussian noise with zero mean and standard deviations of 1%, 2% and 3% of the average distance between points was added to data point. Figure 8 shows that good results are obtained for 1% , 2% and 3%

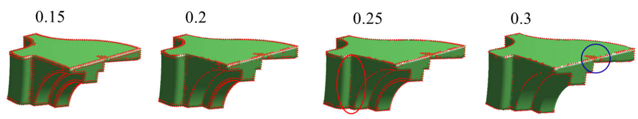

Figure 7. Results with tuning different threshold values for Pauly's method.

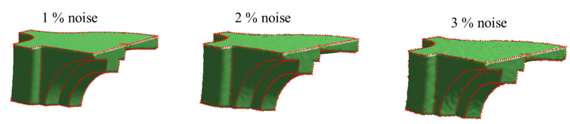

Figure 8. Results for different noise level of fandisk dataset.

Table 2. Number of detected sharp features on the cube model by 4 methods.

Normal	Curvature	Pauly	Our method
236	247	134	91

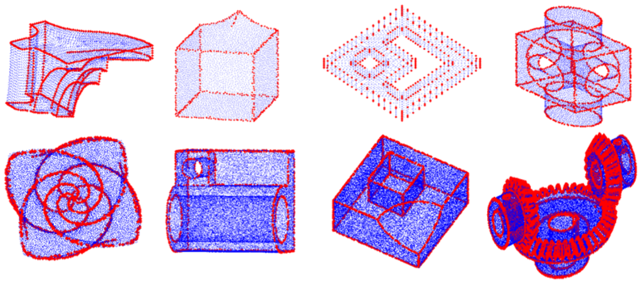

Figure 9. Sharp features extracted from point cloud models

Table 3. Timing of our proposed method for point cloud models.

Point data	Input points	Normal estimation	Projected distance	Total time (s)
Cube	386	0.022	0.017	0.055
Trim-star	5192	0.312	0.247	0.687
Fandisk	6475	0.382	0.301	0.847
Joint	50000	3.037	2.35	6.885

noise and only a small part is contaminated by wrong feature points as in Figure 8.

Moreover, the accuracy of the proposed method is shown in Table 2. A synthetic cube point cloud has 386 points with 91 sharp features. Mean, normal and Pauly's method detect many wrong sharp features while our method extracts 91 sharp features correctly.

The computation time using MATLAB in a 3.2 GHz Intel Core i7 platform for every step of the proposed method is given in Table 3. Our method is remarkably fast and can be applied to large point clouds.

4 Conclusion

A new algorithm is proposed for extracting sharp features automatically on point clouds with optimal neighborhood size. First, the projected distance is calculated at multiple neighborhood sizes. Then, the projected distance values are analyzed to find the optimal neighborhood size and update the final projected distance for each point. Finally, an optimal threshold value is determined automatically using Ostu's method to decide whether the point is on a sharp feature or not. Many experiments have been conducted in order to show that the method works well for various objects with complex shapes.

Acknowledgements

The authors thank the AIM@SHAPE Shape Repository for making models available. This research project was supported by the NSERC / Creaform Industrial Research Chair on 3-D Scanning.

References

[1] Stylianou G, Farin G. Crest lines for surface segmentation and flattening. IEEE Transactions on Visualization and Computer Graphics 2004;10(5):536–44.

[2] Kuo CC, Yau HT. A new combinatorial approach to surface reconstruction with sharp features. IEEE Transactions on Visualization and Computer Graphics 2006;12(1):73–82.

[3] Alexa M, Behr J, Cohen-Or D, Fleishman S, Levin D, Silva CT. Point set surfaces. In: Proceedings of the conference on Visualization '01. VIS '01; Washington, DC, USA: IEEE Computer Society; 2001, p. 21–8.

[4] Amenta N, Kil YJ. Defining point-set surfaces. In: SIGGRAPH 2004. New York, NY, USA: ACM; 2004, p. 264–70.

[5] Demarsin K, Vanderstraeten D, Volodine T, Roose D. Detection of closed sharp edges in point clouds using normal estimation and graph theory. Comput Aided Des 2007;39(4):276–83.

[6] Merigot Q, Ovsjanikov M, Guibas L. Voronoi-based curvature and feature estimation from point clouds. IEEE Transactions on Visualization and Computer Graphics 2011;17(6):743–56.

[7] Daniels II J, Ochotta T, Ha LK, Silva CT. Spline-based feature curves from point-sampled geometry. Vis Comput 2008;24(6):449–62.

[8] Gumhold S, Wang X, Macleod R. Feature extraction from point clouds. In: Proceedings of the 10 th International Meshing Roundtable. 2001, p. 293–305.

[9] Pauly M, Keiser R, Gross M. Multi-scale feature extraction on point-sampled surfaces. Computer Graphics Forum 2003;22(3):281–9.

[10] Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W. Surface reconstruction from unorganized points. In: Proceedings of the 19th annual conference on Computer graphics and interactive techniques. SIGGRAPH 1992; New York, NY, USA: ACM; 1992, p. 71–8.

[11] Weber C, Hahmann S, Hagen H. Sharp feature detection in point clouds. In: Shape Modeling International Conference (SMI '10). 2010, p. 175–86.

[12] Otsu N. A Threshold Selection Method from Gray-Level Histograms. IEEE Transactions on Systems, Man, and Cybernetics 1979;9:62–6.