Fast and Robust Edge Extraction in Unorganized Point Clouds

Dena Bazazian*, Josep R. Casas†, Javier Ruiz-Hidalgo‡
Signal Theory and Communications Department, Universitat Politecnica de Catalunya
Barcelona, Spain
Email: *dena.bazazian@tsc.upc.edu, {†josep.ramon.casas, ‡j.ruiz}@upc.edu

***Abstract*—Edges provide important visual information in scene surfaces. The need for fast and robust feature extraction from 3D data is nowadays fostered by the widespread availability of cheap commercial depth sensors and multi-camera setups. This article investigates the challenge of detecting edges in surfaces represented by unorganized point clouds. Generally, edge recognition requires the extraction of geometric features such as normal vectors and curvatures. Since the normals alone do not provide enough information about the geometry of the cloud, further analysis of extracted normals is needed for edge extraction, such as a clustering method. Edge extraction through these techniques consists of several steps with parameters which depend on the density and the scale of the point cloud. In this paper we propose a fast and precise method to detect sharp edge features by analysing the eigenvalues of the covariance matrix that are defined by each point's k-nearest neighbors. Moreover, we evaluate quantitatively, and qualitatively the proposed methods for sharp edge extraction using several dihedral angles and well known examples of unorganized point clouds. Furthermore, we demonstrate the robustness of our approach in the noisier real-world datasets.**

I. Introduction

Edge extraction has attracted a lot of attention in computer vision. Many applications are built around this concept. Examples include object recognition, similarity, registration, matching, down sampling, and visualization. In non-photorealistic rendering, sharp edges are used to enhance the visual perception. Additionally, in the case of segmentation, knowledge about the position of features can be of great help. The computer vision community has drawn their attention to 3D scene analysis in recent years using stereo and multi-camera systems, and specially after the success of commercial depth sensors, such as MS kinect or Asus Xtion.

Most of the existing research on extracting edges in point clouds are considering either statistical and geometrical methods or estimating the normals on sharp edges.

The challenge for estimating normals on the edge feature points is related to the neighbourhood employed for the normal estimation. The neighborhood may enclose points belonging to different surface patches across the edge feature.

In [1]–[3], Weber et al. estimate normals by triangulation. This technique is quite sensitive, particularly for the relevant points located around edge. Furthermore, triangulation for normal estimation is computationally expensive and difficult to implement in real time for large scale point clouds.

In this paper, we first explore the challenges of sharp edge extraction in small dihedral angles of a 3D surface. In general most techniques to extract sharp edges are prone to error when the dihedral angle is small, due to the local analysis neighborhood taking points on both sides of the surface edge. The main contribution of this paper is to evaluate techniques of sharp edge extraction quantitatively for toy examples and qualitatively for some 3D shapes. Our intention is to extract sharp edges through different techniques and then quantitatively compare the accuracy of results and their computational load.

The remainder of the article is organized as follows. Section II presents related work, followed by a description of our approach and architecture in Section III. Section IV reports the experimental results of our approach, and conclusions are drawn in section V.

II. Related Work

Sharp feature extraction is a key issue in many scientific fields, such as computer graphics, medical imaging, computer vision and computational fluid dynamics. Some research efforts focus on extracting sharp features on point clouds (3D data).

A. Edge extraction

There are multiple techniques for the edge and sharp feature extraction in point clouds, which can be categorized into the classes hereafter: In [6]–[8] the authors have employed robust statistics to extract sharp features. In [9], [10] surface segmentation and in [13] line segmentation has been explored to extract sharp features. Alternately, [1], [2], [11], [12] propose a region growing method that segments the point cloud into clusters and identify the regions with sharp features based on the analysis of the normals of the points.

Fleischman et al. [6] use statistical techniques in order to identify sharp features. Neighborhoods of points are segmented into regions corresponding to the same surface part, and the creation of neighborhoods is guided by the moving least squares (MLS) computation. A development of this work by Daniels et al. [7] extracts feature curves on the reconstructed MLS surface. The benefit is that points on the sharp feature can be identified in the case of noisy and rough input data. Following this idea, Oztireli et al. [8] adopted a robust implicit moving least-squares (RIMLS) method to locally approximate the scanned surface and to preserve sharp features. They have employed kernel regression to extend the moving least squares (MLS) surface reconstruction with sharp features. Their method increases the presentation of sharp features by combining the MLS and local kernel regression.

Demarsin et al. [9] also searched for sharp features in point

cloud data; they are interested in closed sharp features. They use segmentation to identify the regions of sharp features. The output is a set of points with many points representing the feature line. Therefore, they use a graph approach with a minimum spanning tree for closed feature lines.
Xu et al. [10] proposed a method to segment surface and extract edge feature lines of irregular fractured fragments, an accurate surface segmentation is implemented by merging faces based on face normal vector and roughness, and edge feature lines are extracted based on the surface segmentation.
Gumhold et al. [11] present a method that uses the Riemannian tree to build the connectivity information in the point cloud. Then, they analyze the neighborhood of each point via principal component analysis (PCA). The Eigen values of the correlation matrix are used to determine a probability of a point belonging to a feature, and the kind of feature. This method can differentiate between line-type features, border and corner points. The result is a quite dense set of points covering the feature, independently of whether the feature is sharp or not, since points with high curvature values are detected.
Weber et al. [1], [2] present a method for detecting sharp features on an unstructured point cloud; this method computes a Gauss map clustering on local neighborhoods in order to discard all points that are unlikely to belong to a sharp feature.
Feng et al. [12] have presented an algorithm for reliably detecting multiple planes in real time from point clouds. They have constructed a graph whose node and edge represent a group of points and their neighborhood respectively. An agglomerative hierarchical clustering is performed on that graph to systematically merge nodes belonging to the same plane.
Noise reduction algorithms such as jump edge filtering may be suitable, especially for finding better boundaries [1] for each region.
Lin et al. [13] proposed a method that is capable of accurately extracting plane intersection line segments from large-scale raw scan points. The 3D line-support region, namely, a point set near a straight linear structure, is extracted simultaneously. The 3D line-support region is fitted by a Line-Segment-Half-Planes (LSHP) structure, which provides a geometric constraint for a line segment, making the line segment more reliable and accurate.
In addition Wang et al. [24] employ the majority voting scheme, in order to detect distinct geometric features such as sharp edges and outliers in a scanned point cloud.
To obtain sharp edge features precisely it is essential to estimate normals accurately using a convenient neighbourhood points. In the following section we will summarize the normal estimation techniques on the sharp edge features.

B. Normal estimation

Reliable estimation of normal vectors at each point in a scanned point cloud has become a fundamental step in point cloud data processing. Extracting sharp edge features from a 3D point cloud requires accurate normals as input in order to generate high quality surfaces.
The performance of common point based rendering techniques is much dependent on the accuracy of the input normals. In this section, we review some research on the computation of normals; specifically, some efforts on normal estimation to extract sharp features from point clouds.
Park et al. [14] proposed EGG (Elliptic Gabriel Graph) which is an intuitive extension of the Gabriel graph (GG), using an elliptic influence region. EGG provides balanced neighbors by considering both distance and directional spread and can be used for normal vector estimation.
Holzer et al. [15] have presented two methods for fast estimation of surface normals from organized point cloud data using integral images. The use of integral images makes it possible to adapt the considered neighborhood size according to depth and object borders without any additional cost in terms of processing speed.
Demarsin et al. [9] extract closed sharp feature lines to create a closed curve network. They used a first order segmentation to extract candidate feature points and process them as a graph to recover the sharp feature lines. They considered the Delaunay triangulation to estimate normal and the normal of each sample point is estimated as the normal of the least squares plane through the neighbors.
Grim et al. [16] have concentrated on non-uniformly sampled and noisy point data. The output of the algorithm they propose is a surface normal for each data point, a local surface approximation in the form of a one-ring, the local shape (flat, ridge, bowl, saddle, sharp edge, corner, boundary), the feature size, and a confidence value that can be used to determine areas where the sampling is poor or not surface-like.
Li et al. [17] estimate normals on unorganized point clouds by employing statistics methods to detect the local tangent plane for each point. Their proposed algorithm is capable of dealing with points located in high curvature regions or near/on complex sharp features.
Zhang et al. [18] use neighbor points normals as prior knowledge to carry out neighborhood clustering. Afterwards they design an unsupervised learning process to represent the prior knowledge as a guiding matrix. Thereupon by low-rank subspace clustering with the guiding matrix, they segment the anisotropic neighborhood into several isotropic neighborhoods. Hence, the normal of the points near sharp features is estimated as the normal of a plane fitting the consistent sub-neighborhood. Their method is capable of estimating normals in noisy and anisotropic samplings, while preserving sharp features within the original point data.
Wang et al. [19] have developed a normal estimation method in order to establish effectively a proper neighborhood for each point in the scanned point cloud. In particular, for a point near sharp features, an anisotropic neighborhood is formed to only enclose neighboring points located on the same surface patch as the point. Neighboring points on the other surface patches are discarded.
The challenge for estimating normals on the sharp feature points is that the neighborhood employed for the normal estimation would enclose points belonging to different surface patches across the sharp feature. In particular, for a point near sharp features, an anisotropic neighborhood is formed to only enclose neighboring points located on the same surface patch as the point. Neighboring points on the other surface patches are discarded.

III. PROPOSED APPROACH

Our idea is partly motivated by the sharp feature detection method proposed by Weber et al. [1]. They propose a normal

clustering approach based on Gauss map clustering to detect sharp feature points. This algorithm aims to classify points into sharp feature and non-sharp feature points by clustering the normals of potential triangles in the neighborhood.
In Weber's method, each point p has k nearest neighbors. Hence, there are $k(k-1)$ possible triangles [1] built with p and two neighborhood points as vertices. Hereon there are $k(k-1)$ normal vectors of these triangles. Feature detection is performed by analysing the clustering behaviour of all these normals in a given neighborhood of point p.
In this technique, the calculated normals are quite sensitive to measurement noise. Moreover, if a sample point and two neighbour points are aligned in a row, then these three points will not form a triangle. The normal vector does not exist in this case, as illustrated in Fig. 1. Furthermore, any neighborhood formed by points in the two sides of a sharp edge may provide normals which are neither reliable nor accurate. Moreover, for large scale point clouds, this method of normal estimation tends to present a high computational load in order to process $k(k-1)$ normals in the neighborhood of each point.
In order to extract edges, Weber proposes to map the set of normals computed in the neighborhood of each point to the Gaussian sphere (Gauss map). Then these normals are clustered with a hierarchical agglomerative (bottom-up) clustering method [21]. Initially, each point of the Gauss map is considered as a separate cluster. Afterwards, clusters are merged step by step into larger clusters. Weber defines the criterion for the merging process as a distance measure computed from the angle between normals in the Gauss map, as defined in (1).

$$D_c(S_1, S_2) = \frac{1}{|S_1||S_2|} \sum_{x \in S_1} \sum_{y \in S_2} d(x, y), \tag{1}$$

where S_1, S_2 are two clusters to be compared, $|S|$ is the number of elements in a cluster and d is the distance measure on the Gauss map. Each merging increases the distance between the clusters. The clustering algorithm can stop when the distance between the existing clusters exceeds a certain threshold according to the angle between the normals in the Gauss map, as it is presented in Algorithm 1. When the agglomeration process results in a single cluster, the point is considered to lay in a flat plane. In the case of two to four clusters, Weber proposes to classify the point as a feature point.
We propose a method in III-A to estimate normals using the PCA (Principal Component Analysis) and then clustering normals according to Weber's method [1] to extract sharp edge features.
However, these techniques of extracting sharp features by classifying normals are prone to error when the dihedral angle of the sharp edge is small. Hereupon, we propose a faster method in III-B to extract sharp features merely by analysing eigenvalues of covariance matrix that are defined by each point's k-nearest neighbors.

A. PCA and agglomerative clustering

We propose to replace Weber's normal estimation method (triangulation) with PCA. For each point of the cloud, a least squares local plane is fitted to its k nearest neighbors. The normal of each point is the eigenvector corresponding to the smallest eigenvalue of the covariance matrix. After estimating the normal of each point, then we consider k nearest neighbors for the sample point and afterwards cluster the normals of those k nearest neighbors with the agglomerative technique.
We have estimated normals, and implemented it as proposed in [20]. An open source implementation of normal estimation is made available in the Point Cloud Library (PCL) [23] [2].

Algorithm 1 Agglomerative Clustering algorithm

```
procedure AGGLOMERATIVE–CLUSTERING
    Consider each element as an individual Cluster at the first step
    for all the Clusters do
        Find the angle between each Clusters
    end for
    Find the pair of clusters with minimum angle
    Merge two clusters with the minimum angle as in (1)
While MinAngle < Threshold
    Compute number of clusters
end procedure
```

B. Eigenvalue analysis

To avoid the complexity of the process of edge extraction, due to the two step of estimating normals and clustering, in this section we propose a method to extract sharp edges without clustering. Since estimating normals by PCA is based on the eigenvalues of the covariance matrix, we extract edge features merely with the variation of the eigenvalues for each point. This statistical approach eliminates the sensitivity of the normal estimation of sharp edges, furthermore, it removes the clustering step simplifying the edge estimation process.
Covariance is a measure of how much each of the dimensions varies from the mean with respect to each other. For a 3-dimensional data set (X, Y, Z), the 3×3 Covariance matrix C for a sample point $p(x, y, z)$ is given by:

$$C = \begin{bmatrix} Cov(x,x) & Cov(x,y) & Cov(x,z) \\ Cov(y,x) & Cov(y,y) & Cov(y,z) \\ Cov(z,x) & Cov(z,y) & Cov(z,z) \end{bmatrix} \tag{2}$$

where, for instance $Cov(x, y)$ is the Covariance of x, y computed as:

$$Cov(x,y) = \frac{\sum_{i=1}^{k}(x_i - \bar{x})(y_i - \bar{y})}{n-1} \tag{3}$$

Afterwards we explore the Eigenvalues of C: $\lambda_0 \leq \lambda_1 \leq \lambda_2$.

In [4], [5] Pauly et. al. introduce the following concept of surface variation $\sigma_k(p)$:

$$\sigma_k(p) = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \tag{4}$$

The surface variation, $\sigma_k(p)$, for each sample point with k neighbors allows us to distinguish whether the point belongs

[1] Note that Weber computes normals for the discrete Gauss map from the triangle formed by the sample point p and two neighbors p_i, p_j as $n_{ij} = pp_i \times pp_j$, considering the two possible results $n_{ij} = -n_{ij}$ per triangle.

[2] Available as a feature in the PCL trunk: http://www.pointclouds.org.

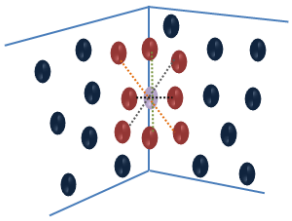

Fig. 1: Illustration of a case with a sample point (in brown) and 8 neighbors (in red) used for triangulation in normal estimation

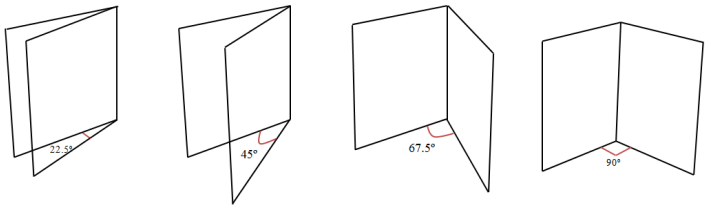

Fig. 2: Sharp edge features for different dihedral angles

to a flat plane or to a salient point (edge) in the point cloud. Since the smallest eigenvalue of covariance matrix for the flat surfaces is zero then the value of the surface variation for the flat surfaces would be zero.

IV. Experimental results

In this section we explain the experimental data set used in our research and the experimental results of edge extraction.

A. Experimental data

We propose to first study the behavior of edge detectors with simple synthetic (artificial) point clouds, for which we can easily label the ground truth. The existence of ground truth also allows to derive quantitative measures for the evaluation of different edge extraction techniques.

1) Geometric shapes: The artificial point clouds in this paper are based on geometric concepts. Since the aim of this research is to extract sharp edges, we determine edges as the junction points of two planes. In order to analyse the effect of edge sharpness on the effectiveness of edge detection, we sample a surface made of two planar rectangular patches joined at various dihedral angles. As shown in Fig. 2.
In addition, in order to explore our approach at different scales, we have designed on a synthetic curved wedge, which has sharp and curve features. This shape is defined as a parametric synthetic shape for objective evaluation of 3D feature descriptors. The generated shape has the form of a curved wedge represented by a point cloud, with varying sharpness and scale at the geometrical edge. We have extended the sharp edge of the symmetric curved wedge to a cylindrical section, by considering a scale, which can be defined as the radius of the cylindrical section. Part of the plane surface is changed into a cylindrical section at a certain point, keeping the continuity at the surface transition both in position and in gradient. Fig. 3 shows a synthetic curved wedge for which the radius of the cylindrical section is 0.2 cm and the angle between the planes is 90°.
The point distribution is uniform both for the curved and for the sharp edge objects.

2) Ground truth estimate: The ground truth of geometric shapes is defined at the synthesis stage by labelling the points located at the proximity of the sharp edges. The width of the edge line for the ground truth, is set the equivalent to the average distance between the points for each neighborhood size.

Fig. 3: Synthetic curved wedge with an angle of 90° and a radios of the cylindrical section of 0.2cm

B. Experimental results

In order to quantitatively compare the results of the different techniques, we propose to use the F1-Score, as metrics, defined as:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{5}$$

where precision is defined as the proportion of points correctly detected by the edge extraction technique and recall is defined as the proportion of points labeled as edges in the ground truth. Precision and recall are defined as:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN} \tag{6}$$

where TP stands for True Positives representing the number of correctly detected points, FP stands for False Positives representing the number of wrongly detected points, FN stands for False Negatives, representing the number of false rejections, i.e. points that belong to the ground truth but are not detected by the edge extraction technique.

1) Edge extraction for a small dihedral angle: Our first variation with respect to Weber's strategy is to implement PCA to estimate normals instead of triangulation, according to the explanation in III-A. A second variation implementation is the proposed strategy for 3D edge detection from the analysis of the eigenvalues according to the explanation in III-B. We compare the detection of edge points on several dihedral angles with the reference method [1] and the two proposed variations. The values of precision, recall and F1-score for this experiment are given in Table I.
When comparing triangulation and PCA methods in the two first column groups of Table I, F1-Scores are almost the same for the largest angles (90° and 67.5°) whereas triangulation is slighly more precise for smaller angles (45° and 22.5°). The computational load of the triangulation method makes it 60 times slower than PCA, as will be discussed below in

	Triangulation Method and Clustering			PCA and Clustering			Eigenvalues Analysis		
Dihedral Angle	Precision	Recall	F1-Score	Precision	Recall	F1-Score	Precision	Recall	F1-Score
90°	0.886	0.956	0.919	0.933	0.896	0.914	0.902	0.909	0.906
67.5°	0.890	0.935	0.912	0.925	0.912	0.918	0.914	0.888	0.901
45°	0.755	0.943	0.839	0.671	0.745	0.706	0.907	0.873	0.890
22.5°	0.589	0.936	0.723	0.602	0.795	0.685	0.795	0.962	0.870

TABLE I: Quantitative evaluation of methods for edge extraction for several dihedral angles

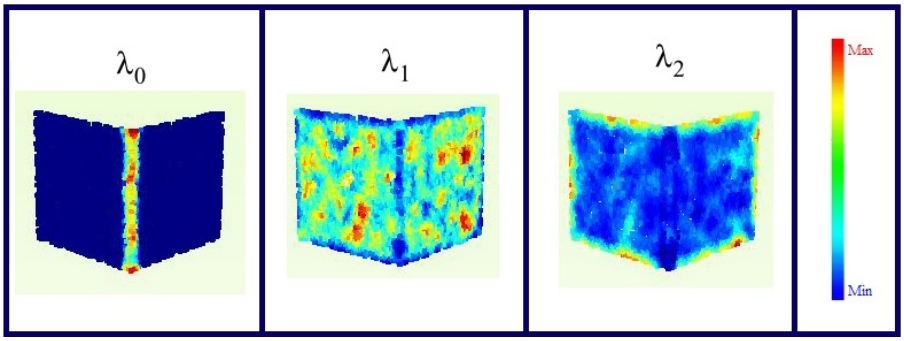

Fig. 4: Analysis of the eigenvalues of the covariance matrix. Behaviour of the eigenvalues $\lambda_0 \leq \lambda_1 \leq \lambda_2$

paragraph IV-B2. The last column group in Table I allows comparing these results with the technique of surface variation [4], [5] for the analysis of eigenvalues according to (4). As shown in the Table, the results for the 45° and 22.5° dihedral angles are more accurate compared to the two previous techniques.
The surface variation method does not employ normal vectors explicitly, whereas for the two previous techniques we have employed agglomerative clustering to group the normal vectors in each neighborhood. The common challenge of the agglomerative clustering method is to find a suitable angle threshold for reliable edge extraction results.
The analysis of the eigenvalues of the covariance matrix in the surface variation method has a certain advantage over the two previous methods to face this challenge, both in terms of precision and simplicity, and also in terms of computational load, as we will show later in IV-B2. Since the covariance is a measure of how much each of the dimensions varies from the mean with respect to each other, the eigenvalues of the covariance matrix measure the variation of the corresponding point along the direction of the eigenvectors. Hence the largest and smallest eigenvalues of the covariance matrix correspond to the dimensions that have the strongest and smallest correlation [22]. There are three eigenvalues for the covariance matrix of the three dimensional data sets, and the eigenvector of smallest eigenvalue is associated to the normal vector. Thus, for a flat surface, the amount of the smallest eigenvalue will be zero and if any curvature is present in the surface defined by the neighborhood of the sample point, then the amount of the corresponding smallest eigenvalue will be larger. Fig. 4 shows the amount of the eigenvalues of the covariance matrix in a pseudocolor representation (red is high, yellow is mid-high and green and blue represent low values).
For qualitative evaluation of the eigenvalue analysis technique, we provide edge detection results on other 3D shapes, as shown in Fig. 5. The Bunny and Dragon are from the Stanford 3D Scanning Repository [3] and the Trim-Star is from the Aim @ SHAPE Shape Repository [4].

2) Fast edge extraction: To evaluate the eigenvalue analysis technique's performance in terms of time, the total computation time for the three techniques is given in Table II. We implemented, the three techniques in C++. All experiments were conducted on a system with an Intel Core i5-3470 CPU of 3.20GHz and 8GB of RAM. No multi-threading or any other parallelism such as OpenMP or GPU was used in our implementation. As shown in Table II, normal estimation with PCA is about 60 times faster than triangulation. Moreover, the eigenvalue analysis is two times faster than clustering in the second step of the reference method to extract edges.

3) Robustness to noise: In order to test the robustness of the eigenvalue analysis method to noise, we show the results for noisy models perturbed with additive 10% and 20% of Gaussian noise. Fig. 6 shows the edge extraction of the surface by eigenvalue analysis in a noisy point cloud. In this case we have computed the surface variation as in (4). Edges are detected and even salient points in the corners are clearly detected in both cases with 10% and 20% added noise. In addition the F1-Scores for edge detection with Gaussian noise in this figure are given in Table III. As shown, the F1-Score is over 0.5 even for the point cloud with 20% Gaussian noise. Increasing the percentage of the Gaussian noise, increases recall and decreases precision, due to the scattered noisy points around the surface, which are counted as the (false) edge points.

4) Multi-scale analysis: In order to extend the proposed eigenvalue analysis method to other scales, we propose a multi-scale approach. In this last experiment, we work both with

[3] http://graphics.stanford.edu/data/3Dscanrep/
[4] http://www.aimatshape.net/

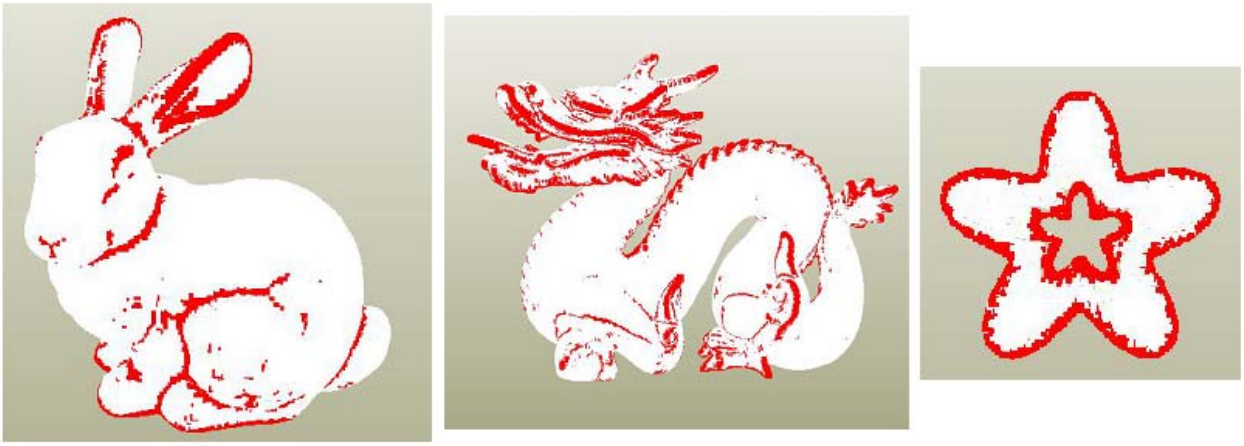

Fig. 5: Eigenvalue analysis for the Bunny, Dragon and Trim-Star in order to extract sharp features

	Total Time		
Dihedral Angle	Triangulation Method and Clustering	PCA and Clustering	Eigenvalues Analysis
90°	2m58.618s	0m2.596s	0m1.260s
67.5°	2m46.539s	0m2.432s	0m1.540s
45°	2m19.363s	0m2.328s	0m1.552s
22.5°	2m20.025s	0m2.480s	0m1.380s

TABLE II: Computation time of different methods of edge extraction

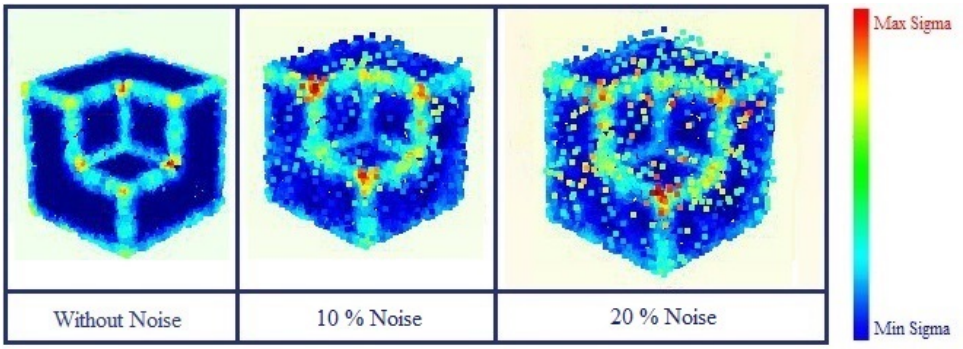

Fig. 6: Edge extraction with the eigenvalues of the covariance matrix with different levels of Gaussian noise in the values of the coordinates of the points in the cloud

the sharp edge between two planes in a 90° dihedral angle, and with the curved wedge presented in IV-A1. We have considered different number of local neighbors and computed the surface variation according to (4). As shown in Fig. 7, we have analysed, for different neighborhood sizes, the evolution of the surface variation parameter $\sigma_k(p)$ at each surface point according to the distance of the point to the edge.

We consider K neighbors as a discrete scale parameter. As explained in IV-B1 when the smallest eigenvalue is zero, means that the sample point lies in a plane. For small neighborhood sizes (K=10, 22), the smallest eigenvalue of the covariance matrix is not zero only when the sample point is placed at a very short distance to the edge, as some points in its neighborhood may be placed after the edge.

For the case of two planes, there is a sharp change in the value of the surface variation at a certain distance to the edge. This turning point in each corresponds to the equivalent radius of the neighborhood for the K nearest neighbors in a uniformly sampled surface. Increasing the number of neighbors, the distance for this turning point increases, which denotes that more points in the neighborhood are located after the edge.

For the case of the curved wedge, the distance for this turning point is constant because, according to IV-A1, it depends on the radius of cylindrical section. In our experiment the radius is 0.2 cm. As shown in Fig. 7, the turning point is placed at 0.2 for all the scales. When the number of neighbors

	Without Noise	10% Noise	20% Noise
Precision	0.875	0.531	0.410
Recall	0.881	0.901	0.922
F1-Score	0.878	0.668	0.568

TABLE III: F1-Score for the eigenvalue analysis in the point cloud without and with Gaussian noise

is small the cylindrical section surface can be considered approximately flat, and when increasing the number of neighbors, the variation of the $\sigma_k(p)$ in the cylindrical section increases. Interestingly enough, a certain spread in the values of surface variation can be observed, according to the number of neighbors, for points placed closer than 0.2 cm to the edge.

V. Conclusion

In this paper we have investigated the challenges of edge extraction techniques in unorganized point clouds. We focused on the quantitative results for several synthetic dihedral angles and the total computation time for several edge detection strategies.

We have proven that normal estimation for edge extraction shows definitely smaller computation times when using PCA than when normal estimation is done by triangulation in the neighborhood, with similar detection efficiency in both cases. We have also proven that edge extraction by the analysis of the eigenvalues the covariance matrix (via the surface variation parameter) is faster and more accurate in small dihedral angles. In addition, it reduces user dependency by eliminating any parameter (threshold) in the edge extraction method.

For this, we have quantitatively compared the accuracy of results and the total computation time in the analysis of synthetic objects for which we do have ground truth.

Furthermore, by adding Gaussian noise to the artificial point clouds, we have demonstrated that this approach how much will be robust in the noisier real-world datasets.

The outcomes of our study favors the proposed strategy of eigenvalues analysis, and can be summarized in three aspects. First, it works for edges in very small dihedral angles. Second, it is a fast procedure, and third, it reduces the user dependency.

We plan to exploit the algorithm proposed in [24] in order to develop a normal estimation method that rejects neighborhood outliers. Furthermore, our future work aims at the design of an adaptive threshold algorithm for multi-scale analysis. We plan to complete semi-automatic edge extraction for 3D point-clouds. Our main aim is a new classification framework that allows discrete surface analysis at multiple scales. Moreover, with the grouping of edge points, we plan to extract feature lines, for analysis or visualization to enhance the the rendering of 3D objects.

Acknowledgment

This work has been developed in the framework of the project TEC2013-43935-R, financed by the Spanish Ministerio de Economia y Competitividad and the European Regional Development Fund (ERDF).

References

[1] Weber, C., Hahmann, S., Hagen, H. *Sharp feature detection in point clouds.* Shape modelling international conference, pp. 175-186, 2010.

[2] Weber, C., Hahmann, S., Hagen, H. *Methods for feature detection in point clouds.* Visualization of Large and Unstructured Data Sets -IRTG Workshop, pp. 90-99, 2010.

[3] Weber, C., Hahmann, S., Hagen, H., Bonneau, G. *Sharp feature preserving MLS surface reconstruction based on local feature line approximations.* Graphical Models, 74 (6), pp. 335345, 2012.

[4] Pauly, M., Gross, M., Kobbelt, L. *Efficient Simplification of Point-Sampled Surfaces.* Visualization, VIS. IEEE, pp. 163-170, 2002.

[5] Pauly, M., Keiser, R., Gross, M. *Multi-scale feature extraction on point-sampled surfaces.* Computer Graphics Forum, 22(3), pp. 281-289, 2003.

[6] Fleischman, S., Cohenor, D., Silva, T. *Robust moving least-squares fitting with sharp features.* ACM Trans Graph, pp. 37-49, 2005.

[7] Daniels, J., Ochotta, T., Ha, L. K. *Spline-based feature curves from point-sampled geometry.* Vis. Comput, 24(6), pp. 449-462, 2008.

[8] Oztireli, C., Guennebaud, G., Gross, M. *Feature preserving point set surfaces based on non-linear kernel regression.* Computer Graphics Forum, 28(2), 2009.

[9] Demarsin, K., Vanderstraeten, D., Volodine, T., Roose, D. *Detection of closed sharp edges in point clouds using normal estimation and graph theory.* Computer-Aided Design, 39(4), pp. 276-283, 2007.

[10] Xu, J., Zhou, M., Wu, Z., Shui, W., Ali, S. *Robust surface segmentation and edge feature lines extraction from fractured fragments of relics .* Journal of Computational Design and Engineering, 2(2), pp. 79-87, 2015.

[11] Gumhold, S., Wang, X., Mcleod, R. *Feature extraction from point clouds.* Proceedings of 10th International Meshing Roundtable, 2001.

[12] Feng, C., Taguchi, Y., Kamat, V. *Fast plane extraction in organized point clouds using agglomerative hierarchical clustering.* IEEE International Conference on Robotics and Automation (ICRA), pp. 6218-6225, 2014.

[13] Lin, Y., Wang, C., Cheng, J., Chen, B., Jia, C., Chen, Z., Li, J. *Line segment extraction for large scale unorganized point clouds.* ISPRS Journal of Photogrammetry and Remote Sensing, 102, pp. 172-183, 2015.

[14] Park, J., Shin, H., Choi, B. *Elliptic gabriel graph for finding neighbors in a point set and its application to normal vector estimation.* Computer Aided Design, 38(6), pp. 619-626, 2006.

[15] Holzer, S., Rusu, R.B., Dixon, M., Gedikli, S., Navab, N. *Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images.* IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2684-2689, 2012.

[16] Grim, C., Smart, W. *Shape classification and normal estimation for non-uniformly sampled, noisy point data.* Computers Graphics, 35(4), pp. 904-915, 2011.

[17] Li, B., Schnabel, R., Klein, R., Cheng, Z., Dang, G., Jin, S. *Robust normal estimation for point clouds with sharp features.* Computers Graphics, 34(2), pp. 94-106, 2010.

[18] Zhang, J., Cao, J., Liu, X., Wang, J., Liu, J., Shi, X. *Point cloud normal estimation via low-rank subspace clustering.* Shape Modeling International (SMI) Conference, 37(6), pp. 697-706, 2013.

[19] Wang, Y., Feng, H., Yung, D., Felix, E., Engin, S. *An adaptive normal estimation method for scanned point clouds with sharp features.* Computer-Aided Design, 45 (11), pp. 1333-1348, 2013.

[20] Rusu, R.B. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*, PhD dessertation, Institut fr Informatik der Technischen Universitt Mnchen, 2009.

[21] Hastie, T., Tibshirani, R., Friedman. J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, Boston, MA, USA, 2nd edition, 2009.

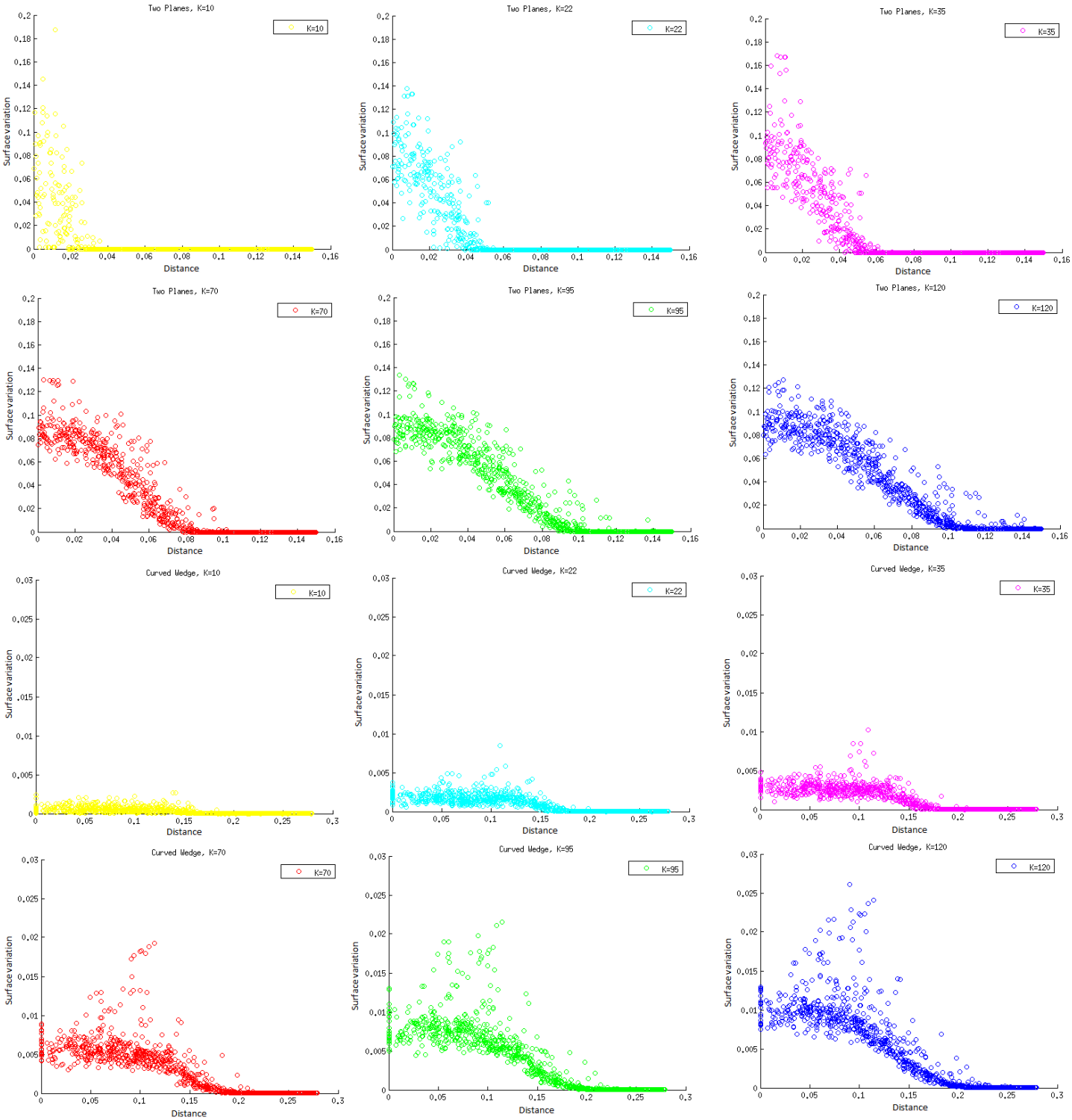

Fig. 7: Scatter-plot of the evolution of surface variation parameter for different neighborhood sizes, over the distance of the each point to the edge for the intersection of two planes (top) and the curved wedge surface (bottom)

[22] Garland, M. *Quadric-Based Polygonal Surface Simplification.* PhD thesis, Carnegie Mellon University, CS Dept., 1999. Tech. Rept. CMU-CS-99-105.

[23] Rusu, R.B., Cousins, S. *3D is here: Point Cloud Library (PCL).* In IEEE International Conference on Robotics and Automation (ICRA), pp. 1-4, 2011.

[24] Wang, Y., Feng, H. *Outlier detection for scanned point clouds using majority voting.* Computer-Aided Design, 62, pp. 31-43, 2015.