Title:
Estimating Principal Curvatures in Noisy Point Clouds through Local Quadric Surface Fitting

Authors:
Farbod Khameneifar, farbod.khameneifar@polymtl.ca, École Polytechnique de Montréal
Hamid Ghorbani, hamid.ghorbani@polymtl.ca, École Polytechnique de Montréal

Keywords:
Point Cloud, Curvature Estimation, Quadric Surface Fitting, Local Neighborhood

DOI: 10.14733/cadconfP.2018.54-58

Introduction:
3D scanners sample coordinate data points from an object's surface. The set of data points is called a point cloud. Reliable estimation of curvatures of the underlying surface at a data point of a scanned point cloud is an essential task in several point cloud processing applications including segmentation, feature detection, registration, smoothing, etc. It is well known that the local surface shape in the vicinity of a data point can be well approximated by a quadric surface [6]. Local quadric surface fitting is thus considered as one of the accurate and robust methods for estimating curvatures at discrete data points sampled from a smooth surface [4]. In this approach, a generic quadric surface is fitted, in least-squares sense, to the local neighborhood (i.e., neighboring points) of the point of interest. Then, the surface curvatures are calculated for the fitted local quadric surface and assigned back to the data point [1]. The effectiveness of this approach, however, depends on the quality of the fitted quadric surface, which itself depends on the selected neighboring points to which the surface is fitted. In order to fit a generic quadric surface, at least 10 neighboring points are needed. It should be noted that scanned point clouds typically have non-uniform point distribution and considerable noise. If special attention is not paid to the selection of local neighboring points, an imbalance would be present in the set of neighboring points, which can adversely affect the quality of the fitted quadric surface.

The first author has recently identified the balance requirement of the local neighborhood for quadric surface fitting, and proposed the Territory Claiming (TC) algorithm for establishing a balanced neighborhood for this purpose [2]. The main idea behind the territory claiming algorithm is that to be able to well approximate the underlying geometry in the vicinity of a data point p, the points in the local neighborhood of the point p must sufficiently cover a small local surface patch (topologically equivalent to a disk) around the point p. Therefore, the set of local neighboring points around the point p should meet the following two requirements: directional balance and close proximity. In this paper, the effect of the use of the balanced neighborhood for estimating the surface curvatures at the discrete data points is investigated. This work demonstrates that the utilization of the neighboring points selected based on the territory claiming algorithm results in significant improvement of curvature estimation through local quadric surface fitting.

Related Works:
Most commonly, distance-based method (i.e., k-nearest neighbors) is used for local neighborhood identification, which selects the neighbors based on only their distance from the query point [1,4]. While the distance-based method is simple to implement, it leads to biased local neighborhoods when applied to the non-uniformly distributed points of a noisy point cloud. As an alternative to the distance-based neighborhoods, mesh-based neighbors have been proposed by researchers [3]. The reconstructed mesh explicitly defines the neighboring relationship, as the points connected by edges to the point of interest are considered as the neighbors. However, very often the mesh structure becomes distorted and unreliable when a relatively large amount of noise is present. Park et al. [5]

have proposed elliptic Gabriel graph (EGG), a neighborhood graph for selecting neighboring points while preventing the bias problem of distance-based methods. Their motivation has been driven by the need for avoiding the bias in plane fitting for estimating the normal vectors at discrete data points. Although their method works well for selecting a few points for plane fitting without bias, the expansion of the EGG neighborhood to include more points (required for quadric surface fitting) often results in an ill-proportioned set of points. In particular, points farther from the point of interest would be included, which have a harmful impact on close proximity requirement of the local neighborhood. In this paper, the territory claiming (TC) algorithm [2] is used for establishing a balanced neighborhood around the point of interest in order to fit a reliable local quadric surface. The accuracy of estimated curvature using the fitted surface over the TC neighborhood is analyzed and compared to the estimated curvature through the fitted surface over the other existing neighborhoods, namely distance-based (k-nearest neighbors), mesh-based and EGG neighbors.

Curvature Estimation:

Given a scanned point cloud from a smooth surface, we are interested in calculating the principal curvatures (i.e., the maximum and minimum of the normal curvature, k_1 and k_2, respectively) at each point. For an analytical surface $S(x, y, z) = 0$, the principal curvatures at any point on the surface can be calculated using the first and second fundamental forms [1]. If the underlying surface of the point cloud in the vicinity of a data point p can be locally approximated by such an analytical function, the approximated representation can be used to estimate the curvature at point p. The generic quadric surface of Eqn. (3.1) is found to be the most suitable option for this purpose [1].

$$S(x,y,z) = c_1x^2 + c_2y^2 + c_3z^2 + c_4xy + c_5yz + c_6zx + c_7x + c_8y + c_9z = 0 \tag{3.1}$$

The following steps should be taken to estimate the principal curvatures at point p (the details can be found in [1]):

1) Establish a set of at least 10 local neighboring points around the point p.
2) Fit the generic quadric surface S of Eqn. (3.1) to the neighboring points.
3) Find the closest point p_0 on the quadric surface S to the point p.
4) Calculate the principal curvatures of the quadric surface S at p_0.
5) Assign back the calculated curvature to the point p.

While theoretically sound, the effectiveness of the above approach in practice depends on how well-balanced the neighboring points for fitting are selected out of non-uniformly distributed point cloud data points in the first step. We propose the utilization of the Territory Claiming (TC) algorithm for establishing a balanced neighborhood of points that helps more reliable curvature estimation. For completeness, the TC algorithm is briefly outlined below. For more details, readers are referred to [2].

Territory Claiming (TC) Algorithm

First, the algorithm sorts all the points in a subset of the point cloud based on the distance from the point p, and then the nearest point to the point p is picked as the first accepted neighbor in $N(p)$. Based on the already accepted neighbor, the algorithm partitions the space into two divisions: the claimed territory of the accepted neighboring point, and the unclaimed territory in which the next neighboring point can reside. The potential neighboring points are picked one at a time in the order of increasing distance from the point p, and checked against the following criterion: The point is accepted as a neighbor, only if it is not located in the claimed territory of any of the already established neighbors. A positive parameter β is involved in the creation of the claimed territories (more details can be found in [2]). Fig. 1 shows an example of selecting neighboring points around the point of interest p using the TC algorithm for a particular value of β (i.e., $\beta = 2$). 10 nearest points to the point p are shown in the figure. The points are sorted from q_1 to q_{10} in the order of increasing distance from p. The point q_1 is the nearest point to p and considered the first accepted neighbor. Then, the second nearest point q_2 is checked whether it can be a neighboring point. As it is shown in Fig. 1(a), the point q_2 is in the claimed territory of point q_1 (hatched region). Therefore, the point q_2 cannot be accepted as a neighboring point of the point p. However, q_3 can be accepted as a neighbor, since it is not in the claimed territory of q_1. Likewise, the points q_4 and q_5 cannot be accepted, because they are in the claimed territory of q_3; but q_6 is accepted, since it is not in the claimed territory of any of the already established neighbors (i.e., q_1 and q_3). Following this sequence, the algorithm creates the directionally balanced set of neighboring points $\{q_1, q_3, q_6, q_8\}$, shown with its convex hull (in green) in Fig. 1(b).

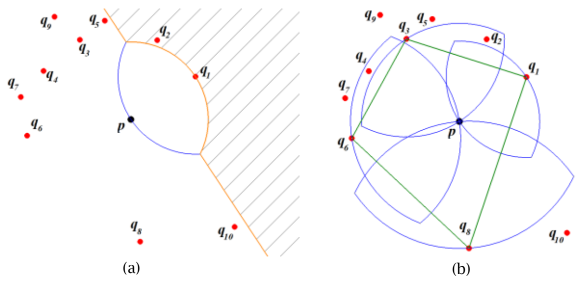

Fig. 1: Selecting a directionally balanced set of neighboring points around the point p: (a) the candidate point q_2 inside the claimed territory of the accepted neighboring point q_1; (b) the selected balanced set of neighboring points $\{q_1, q_3, q_6, q_8\}$.

The algorithm searches for all the distinct directionally balanced sets of neighboring points in a range of β values, and selects the closest neighbor set to the point of interest as the best neighborhood. That is the set of neighboring points, $N_\beta(p)$, for which the average Euclidean distance of the neighboring points from p is minimal.

Once the closest directionally balanced set of neighbors around the point p is established, the algorithm checks the number of points in the established neighborhood. If the number of points is less than 10, the established neighborhood can be expanded to obtain the needed number of points while maintaining the balance of the overall neighborhood. The algorithm removes the already established set of neighboring points from point cloud, and repeats the presented process once more to select another closest directionally balanced neighbor set around the point p from the remaining points of the point cloud. The union of the two sets of neighbors is the overall neighborhood. The algorithm iterates this remove-select-combine procedure as long as the overall number of neighboring points reaches at least 10 points.

Results and Discussion:

Case studies have been conducted to examine the effect of local neighborhood on the accuracy of curvature estimation. The results of the utilization of the proposed local neighborhood (established by TC algorithm) is compared to the results from the neighborhoods established by other existing methods, namely k-nearest neighbors (k-NN), mesh neighbors, and elliptic Gabriel graph (EGG) neighbors. The accuracy of the estimated principal curvatures, k_1 and k_2, is evaluated using both simulated and scanned noisy point cloud data sets.

The reason for using the simulated point cloud data is that for ideal theoretical surfaces the true principal curvatures are known at each point that can be used as a reference for comparison. The approximation error of the estimated curvatures is quantified as the difference between the estimated value and the true value. Two different geometric shapes were employed to generate the simulated point clouds: the surface of a torus and a bicubic Bézier patch. On each of the two surfaces, 10,000 points were sampled with random distribution. Then, different levels of noise (Gaussian deviates) in random directions were superimposed onto the sampled points to simulate noisy point clouds. The term "noise level" in this work refers to the standard deviation of the noise, which was specified as a percentage of the diagonal length of the bounding box (BBD%) of the point cloud. Fig. 2 shows noisy point clouds for torus and bicubic Bézier surface patch with 0.5 BBD% superimposed noise.

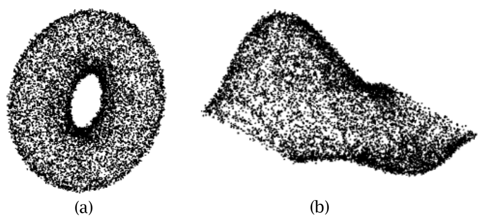

Fig. 2: Simulated noisy point clouds of: (a) torus, and (b) bicubic Bézier surface patch.

Fig. 3 and Fig. 4 show the comparison results of the average error and standard deviation (σ) for the estimated k_1 and k_2 at the 10,000 points of the torus and Bézier patch point clouds (with the superimposed noise of different levels up to 0.5 BBD%), respectively. Among the compared neighborhoods, 10-NN and 24-NN are the k-nearest neighbors (k-NN) with k = 10 and k = 24. As the plots suggest, when the proposed neighborhood (established by TC algorithm) is utilized, more accurate and more consistent curvature estimation results are obtained in comparison with the cases in which other existing neighborhoods are used. In general, the superiority of the proposed approach is more pronounced as the noise level increases.

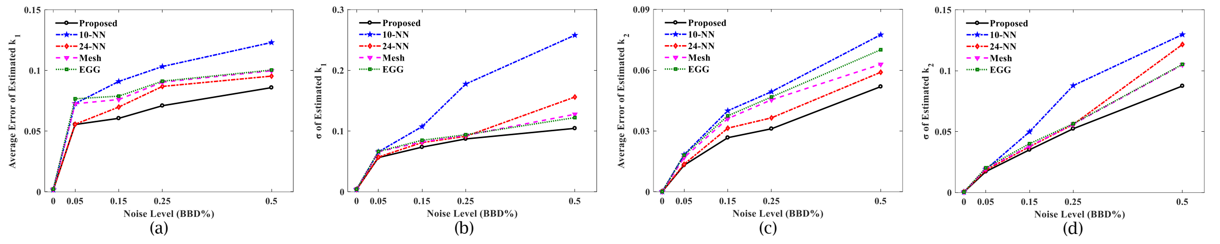

Fig. 3: Comparison of principal curvatures approximation errors for torus point clouds: (a) average error, and (b) standard deviation for k_1; (c) average error, and (d) standard deviation for k_2.

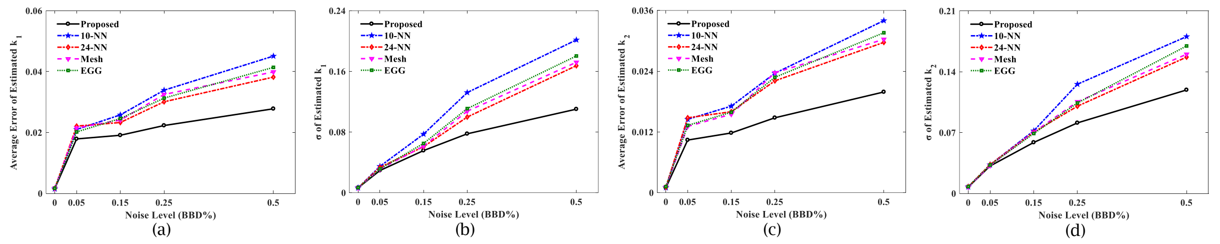

Fig. 4: Comparison of principal curvatures approximation errors for bicubic Bézier patch point clouds: (a) average error, and (b) standard deviation for k_1; (c) average error, and (d) standard deviation for k_2.

The second case study evaluates the performance of the proposed approach on a scanned point cloud. The scanned point cloud (shown in Fig. 5(a)) containing 99,925 points has been obtained by an LDI Surveyor WS3040 3D laser scanner. In contrast to simulated point clouds, the true principal curvatures for real scanned data points are unknown. In this work, comparison of the methods was made according to the consistency in curvature estimation at reduced point densities due to the fact that the bias issue becomes more significant as the point cloud gets sparser. The principal curvatures, k_1 and k_2, at each point of the original point cloud were estimated from the quadric surface fitted to the local neighborhood established through each of the five neighborhood selection methods. The obtained curvatures for each method were regarded as the reference principal curvatures for the corresponding method. Then, by randomly deleting 20%, 40%, 60%, and 80% of points from the original data set, the point density was reduced, and new point sets (shown in Fig. 5(b-e)) were obtained.

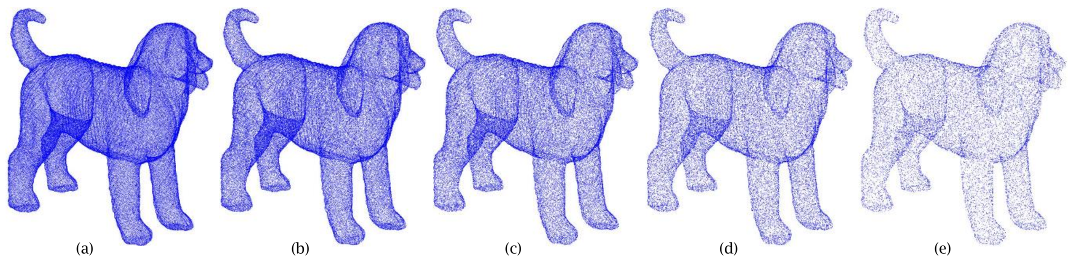

Fig. 5: Scanned point clouds: (a) original set, and the reduced sets with a point density of (b) 80%, (c) 60%, (d) 40%, and (e) 20% of the original set.

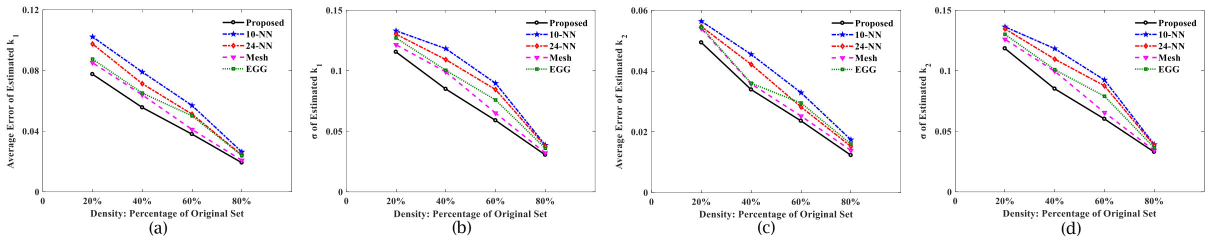

Fig. 6: Comparison of principal curvatures approximation errors for the scanned point clouds of Fig. 5 with reduced point densities: (a) average error, and (b) standard deviation for k_1; (c) average error, and (d) standard deviation for k_2.

For any of the reduced point sets, the principal curvatures were estimated at each point by each of the five methods and compared with the corresponding reference principal curvatures. The estimation error at each point is the difference between the estimated values and the reference values of principal curvatures. The average and standard deviation of estimation errors for all the points of the reduced point sets are shown in Fig. 6. It is observed that the proposed approach outperforms the other competitors in giving more consistent results for the reduced point sets.

Conclusions:

In order to improve the performance of curvature estimation for noisy point clouds via local quadric surface fitting, this paper proposed the utilization of the territory claiming (TC) algorithm for identifying a balanced set of neighboring points around the point of interest. The implementation results from simulated and scanned point cloud data sets substantiated the effect of local neighborhood on the accuracy and consistency of curvature estimation outcome. The results demonstrated that the proposed approach is more effective than the other existing methods in handling the non-uniform distribution and noise in point cloud data, and is more robust towards point density variation.

Acknowledgements:

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under the Discovery Grants program, and in part by the start-up fund of École Polytechnique de Montréal. The scanned point cloud in this paper (the Dog model) has been sampled at CAD/CAM/CAI research laboratory directed by Prof. Hsi-Yung (Steve) Feng at the University of British Columbia. The authors are grateful to Prof. Feng for providing the point cloud for the tests of this paper.

References:

[1] Douros, I.; Buxton, B. F.: Three-dimensional surface curvature estimation using quadric surface patches, Scanning 2002 Proceedings, Paris, May 2002.

[2] Khameneifar, F.; Feng, H.-Y.: Establishing a balanced neighborhood of discrete points for local quadric surface fitting, Computer-Aided Design, 84, 2017, 25-38. https://doi.org/10.1016/j.cad.2016.12.001

[3] Lange, C.; Polthier, K.: Anisotropic smoothing of point sets, Computer Aided Geometric Design, 22(7), 2005, 680-692. https://doi.org/10.1016/j.cagd.2005.06.010

[4] Meek, D. S.; Walton, D. J.: On surface normal and Gaussian curvature approximations given data sampled from a smooth surface, Computer Aided Geometric Design, 17(6), 2000, 521-543. https://doi.org/10.1016/S0167-8396(00)00006-6

[5] Park, J. C.; Shin, H.; Choi, B. K.: Elliptic Gabriel graph for finding neighbors in a point set and its application to normal vector estimation, Computer-Aided Design, 38(6), 2006, 619-626. https://doi.org/10.1016/j.cad.2006.02.008

[6] Struik, D.J.: Lectures on Classical Differential Geometry: Second Edition, Dover Publications, New York, NY, 1988.