ISO/IEC JTC 1/SC 29/WG1 M 88038
88th Meeting, Online, 07-10 July 2020

INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION

ISO/IEC JTC 1/SC 29/WG1
(ITU-T SG16)

Coding of Still Pictures

JBIG
Joint Bi-level Image Experts Group

JPEG
Joint Photographic Experts Group

TITLE: Benchmarking of the plane-to-plane metric

SOURCE: Evangelos Alexiou (EPFL), Touradj Ebrahimi (EPFL)

PROJECT: JPEG PLENO

STATUS: Informative

REQUESTED ACTION: For review and information by WG1

DISTRIBUTION: WG1

Convenor Contact:

ISO/IEC JTC 1/SC 29/WG 1 Prof. Touradj Ebrahimi
EPFL/STI/IEL/GR-EB, Station 11, CH-1015 Lausanne, Switzerland
Tel: +41 21 693 2606, Fax: +41 21 693 7600, E-mail: Touradj.Ebrahimi@epfl.ch

1. Introduction

Surface normal vectors attributes are crucial for a number of use-cases in point cloud imaging including rendering, surface reconstruction, segmentation, and feature extraction, among others. In essence, the normal vectors indicate the shape of the underlying 3D model, which is represented through point primitives. Normal vectors are not natively exported during point cloud acquisition in most cases, and when this happens, it is not necessary that they accurately reflect the underlying surfaces. Thus, it is rather common to (re-)estimate them from a point cloud model in an offline, post-processing stage, where various normal estimation algorithms and configurations might be tested.

In principle, the problem of normal estimation on point clouds has been extensively studied from different communities, such as computer graphics, signal processing and mathematics, while lately, deep-learning solutions have also been proposed. In this context, a point cloud is seen as a collection of unorganized points that are sampled from continuous surfaces, and the objective is to infer the underlying shape from the discrete samples. Normal estimation can be considered as an ill-posed problem, in the sense that there is no unique solution for a given point cloud. Moreover, the performance of normal estimation algorithms is highly affected by the density of samples and surface curvature irregularities across a model, as well as the presence of noise that can be injected during acquisition, or other processing stages (e.g., compression) [1]. The accuracy, robustness and efficiency of widely-used normal estimation algorithms has been evaluated and reported by a series of studies in the literature [2, 3].

In this report, the objective is to benchmark the plane-to-plane metric[1] described in [4]; this is an objective quality metric that is based on the angular similarity of tangent planes between two point cloud models. The computation of this metric relies on normal vectors that are carried with associated pairs of points, and quantifies the difference in orientation of their corresponding tangent planes. Thus, the performance of the metric depends by default on the normal vectors and the way they approximate the underlying surfaces. Thus, the selection of the normal estimation algorithm as well as its configuration is crucial in order to achieve high prediction power. In this framework, we choose three widely-used normal estimation algorithms, and we test several neighborhood sizes over which the normal vector of a point is estimated:

- Plane fitting using *k*-nearest neighbors
- Plane fitting using range search
- Quadric fitting using range search

In a first step, we evaluate the normal estimation error of the selected algorithms and configurations against ground truth normal vectors on point clouds that are produced after sub-sampling a set of reference mesh models. This stage is referred to as *verification*. In a second step, we benchmark the plane-to-plane metric, as a function of the selected normal estimation algorithms and configurations, against ground truth subjective scores from two datasets, namely, JPEG-ES2-PCCD [5] and M-PCCD [6]. This stage is referred to as *benchmarking*.

[1] https://github.com/mmspg/point-cloud-angular-similarity-metric

2. Verification

2.1 Models

We pick 4 mesh models with different shapes and curvature distributions from the references of the PointXR dataset [7]. The models can be seen in Figure 1, namely, *guanyin*, *roy*, *rhetorician*, and *vase*, from left to right.

Figure 1: Test mesh models used at *verification*.

2.2 Methodology

For every mesh model, we repeat the following procedure to obtain the ground truth normal vectors:

1. Load a mesh model in CloudCompare [8] and use a build-in functionality to estimate the normal vectors per vertex i.e., the normal at a vertex is obtained as the mean normal vector across all the triangles connected to this vertex[2].
2. Generate a point cloud by sampling the mesh model at different sparsity levels, using a target number of output points[3]. In this report, we set 250K, 500K, 1M and 2M as targets, to simulate realistic use cases of point clouds that are employed in point cloud compression activities. The point cloud is generated after obtaining a constant number of samples at random positions from each triangle of the mesh. The normal vector of a point is obtained after applying spatial interpolation on the normal vectors of the vertices that surround it in the original mesh.
3. Remove neighboring points of the output point cloud with identical normal vectors.
4. Scale geometry of the point cloud in the range [0, 1][3].
5. Apply normal estimation algorithm on the point cloud geometry.

At this point, we have the testing point clouds associated with the ground truth normal vectors. The normal estimation algorithms under evaluation are applied on the coordinates of the testing point clouds (step 5), and evaluated against the ground truth normal vectors using the *average error per point in degrees*. Note that the angular error measurements are obtained considering *unoriented normal vectors*. Note that it is common to obtain flipped normal vectors even at cases of neighboring points; to address this issue, several algorithms have been proposed (e.g., Minimum Spanning Tree), however, their performance and impact is outside of the scope of this report.

[2] https://www.cloudcompare.org/doc/wiki/index.php?title=Normals%5CCompute
[3] https://www.cloudcompare.org/doc/wiki/index.php?title=Mesh%5CSample_points

2.3 Normal estimation algorithms

2.3.1 Plane fitting using *k*-nearest neighbors

For this algorithm, the classic approach described in [9] is employed. In particular, the *k*-nearest neighborhood (*k*-nn) is used to identify a local region over which a planar surface is fitted. For every model and each target number of points, neighborhoods of 8 up to 512 points are set, by doubling the previous number.

The Meshlab [10] build-in function is used to implement this algorithm. It is noteworthy that throughout our experimentation, we have confirmed that we get identical results with the Matlab implementation, and a custom script running the Principal Component Analysis followed by the selection of the eigenvector with the smallest eigenvalue, for these models. Moreover, it should be mentioned that different results are obtained when using the same algorithm as implemented in PCL [11].

2.3.2 Plane fitting using range search

This algorithm is identical to the one described above (section 2.3.1), with the only difference that a range search with a fixed radius is employed to identify the neighborhood, instead of the *k*-nearest points. For every model and each target number of points, neighborhoods are defined using a radius equal to 0.001 up to 0.01, with a step of 0.001 (each model's geometry is limited to a bounding box of size 1).

The CloudCompare build-in function is used to implement this algorithm. Benefiting from the octree structure that is used to identify the nearest points, the implementation is fast. Another important advantage, which is a typical bottleneck in the range search algorithm, is the fact that in case of zero neighboring points at a specific radius, the search range is progressively increasing up until a minimum number of points are reached. Thus, normal vectors are estimated for all points.

In the plane fitting algorithm described in section 2.3.1, for every point, its normal vector is estimated across a fixed number of points with arbitrary range, while in this case, an arbitrary number of points is used which span at a fixed range. There are no indications in the literature which algorithm provides more accurate results. In previous studies, usually, one is selected over the other, and different configurations of the chosen algorithm are evaluated [2, 3].

2.3.3 Quadric fitting using range search

This algorithm is based on fitting a quadratic polynomial in a neighborhood that is identified by using the range search algorithm. For every model and each target number of points, neighborhoods are defined using a radius equal to 0.001 up to 0.01, with a step of 0.001 (each model's geometry is limited to a bounding box of size 1).

The CloudCompare build-in function is used to implement this algorithm. As mentioned in section 2.3.2, there are several advantages that are offered.

2.4 Results

In Figure 2, we present the normal estimation error in terms of angles expressed in degrees between the *unoriented* estimated and ground truth normal vectors. Note that in each row a different model is depicted, while in each column a different normal estimation algorithm is provided. In every figure, each curve reports the angular error at different neighborhood sizes, and corresponds to a certain variation of a model (i.e., target number of points, or sparsity level).

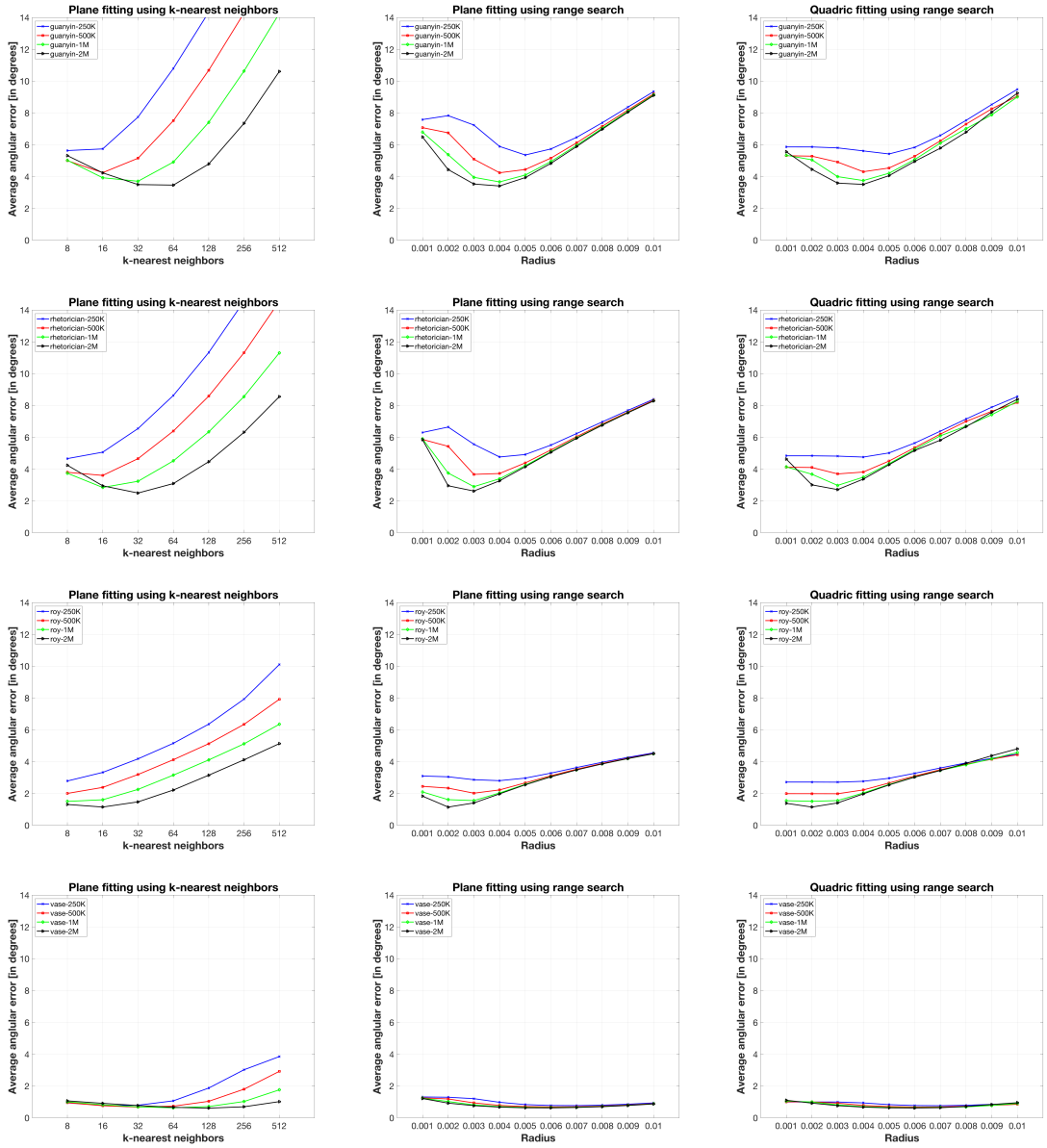

Figure 2: Normal estimation error for normal estimation algorithms and configurations per model.

As we can see, for all algorithms, the accuracy of the estimated normal vectors strongly depends on the testing model (geometric characteristics and sparsity levels) and the configuration of the algorithm [2, 3]. It is evident that plane and quadric fitting using range search are comparable, while the error trend for the plane fitting using *k*-nn behaves differently. Based on our results, when using *k*-nn, the minimum error is achieved at larger *k*'s for denser models, whereas for sparser models, the minimum is achieved for smaller neighborhoods. In principle, the normal estimation error is constantly lower at denser models, for all *k*'s. When using range search, similar performance is observed between plane and quadric fitting, with the latter showing minor accuracy improvements. In both cases, for sparse models, the error is stable or even decreasing as the radius is increasing up to a threshold. In principle, the normal estimation error is lower at denser models at low- up to mid-radii, and then the same error is observed independently of the sparsity level. Considering that larger radii might lead to accounting points that don't belong to the same local region (e.g., "occluded" surfaces), the error will increase in a

similar way independently of the sparsity level of a model, given that all of them are scaled identically. The same applies across different fitting surfaces, where we observe very similar error trends after the points where the minimum error is achieved.

Figure 3 indicates the mean and the standard deviation of the number of points for different radii, for all variations of the *guanyin* model. As expected, the number of points deviates depending on the model's sparsity. In principle, we can say that there is a sweet spot for the identification of a region that would lead to more accurate results, per sparsity level.

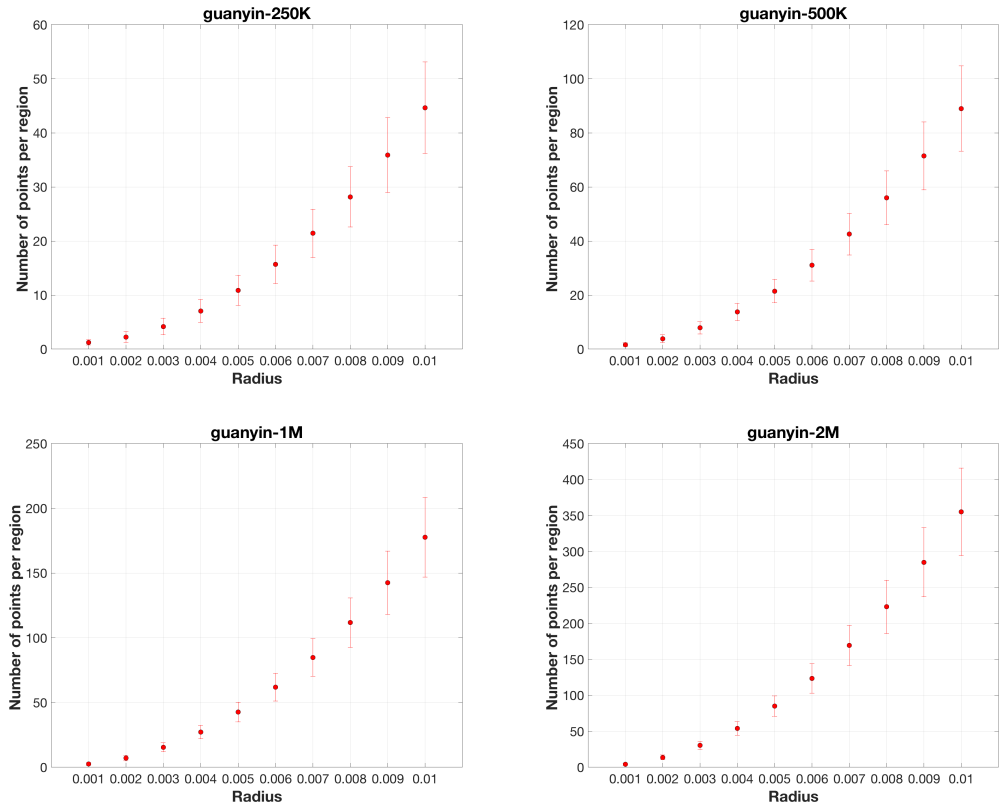

Figure 3: Mean and standard deviation of the number of points that consist of local neighborhoods when using range search, for different variations of the *guanyin* model.

3. Benchmarking

3.1 Data sets

We evaluate the performance of the plane-to-plane metric as a function of the normal estimation algorithms that were analysed in section 2. For this purpose, we use two subjectively annotated datasets, namely, JPEG-ES2-PCCD [5] and M-PCCD [6].

- The JPEG-ES2-PCCD contains 6 colored static point clouds that represent human figures of different characteristics (e.g., resolution, acquisition noise) that are degraded using V-PCC and two G-PCC variants after combining the Octree and TriSoup geometry codecs with the Lifting color encoder, following the MPEG Common Test Conditions (CTC) document [12]. The exact degradation levels per model can be found in [5]. The compressed point clouds were subjectively evaluated using point-based rendering in four independent laboratories in a passive

evaluation protocol. This dataset is obtained from the recent JPEG activities in the context of the Exploratory Study 2.

- The M-PCCD contains 8 colored static point clouds that represent both human figures and inanimate objects that are degraded using V-PCC and the four G-PCC variants (i.e., Octree+Lifting, Octree+RHAT, TriSoup+Lifting and TriSoup+RAHT), following the MPEG CTC document [12]. The compressed point clouds were subjectively evaluated using point-based rendering in two independent laboratories in an interactive evaluation protocol without imposing any time limitation.

3.2 Normal estimation algorithms

3.2.1 Plane fitting using *k*-nearest neighbors

For this algorithm, a *k*-nn is used to identify a local region to fit a planar surface. For every model, neighborhoods of 8 up to 1024 points are set, by doubling the previous number.

The Meshlab build-in function is used to implement this algorithm.

3.2.2 Plane fitting using range search

For this algorithm, a range search is performed setting a radius to identify a local region to fit a planar surface. For every model, neighborhoods are formed using a radius of 5, 10, 20, 30, 40 and 50.

The CloudCompare build-in function is used to implement this algorithm.

3.2.3 Quadric fitting using range search

For this algorithm, a range search is performed setting a radius to identify a local region to fit a quadric surface. For every model, neighborhoods are formed using a radius of 5, 10, 20, 30, 40 and 50.

The CloudCompare build-in function is used to implement this algorithm.

3.3 Plane-to-plane metric

The plane-to-plane metric [4] is based on the angular similarity of tangent planes that correspond to associated points between a stimulus under evaluation, B, and a reference point cloud, A. In particular, for every point of the stimulus under evaluation, b_i, a point that belongs to the reference point cloud, a_j, is identified, using the nearest neighbor algorithm. Then, the angular similarity of tangent planes is computed using the corresponding normal vectors and based on the angle θ, which denotes the minimum out of the two angles that are formed by the intersecting tangent planes (or, equivalently, the unoriented normal vectors).

This error value is associated to every b_i and provides an approximation of the dissimilarity between the underlying local surfaces. A total distortion measurement for the stimulus under evaluation is obtained as the mean squared error across all individual error values.

The plane-to-plane angular similarity is defined as the symmetric value, obtained after setting both point clouds A and B as the reference, and keeping the maximum error (i.e., minimum similarity).

3.4 Benchmarking

The Pearson Linear Correlation Coefficient (PLCC), the Spearman Rank Order Correlation Coefficient (SROCC), the Root-Mean Square Error (RMSE) and the Outlier Ratio (OR) performance indexes are computed between pairs of MOS and predicted MOS, to measure the performance of the plane-to-plane

metric against the subjective ground truth. The predicted MOS is obtained after applying the logistic fitting function on the objective quality scores, essentially, repeating the analysis reported in [5, 6], in order to be able to compare the results.

3.5 Results

It should be noted that as a first step, and before applying any normal estimation algorithm, duplicated coordinates were discarded for all models from all datasets that are employed in the sections below. For most configurations, the number of points that form a local neighborhood is large, thus, the impact of this processing step should not be critical; however, it is stated for the shake of completeness.

3.5.1 JPEG-ES2-PCCD

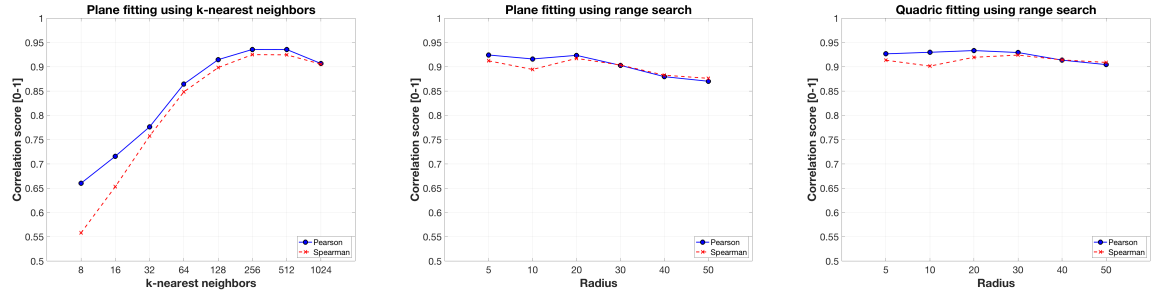

Figure 4: Pearson and Spearman correlation coefficients from benchmarking of the plane-to-plane metric using JPEG-ES2-PCCD, as a function of the region size defined to estimate the normal vectors.

In Figure 4, the Pearson and the Spearman correlation coefficients are depicted as a function of the neighborhood size per normal estimation algorithm, against the JPEG-ES2-PCCD subjective scores. It can be observed that when using plane fitting with *k*-nn, the performance is improving as the number of neighbors is increasing of up to $k = 256$, remains almost stable until $k = 512$, where it starts to decay until $k = 1024$. When using plane fitting with range search, the performance is almost stable and high for radii up to 20, while it progressively decreases for larger values. A similar performance is observed when using quadric fitting, with slightly better performance and less deviations across the tested radii.

In Table 1 we report the performance indexes for the best performing configurations of all normal estimation algorithms that were tested, along with the D1 PSNR and D2 PSNR metrics [13] for comparison purposes.

Normal estimation algorithm	**PLCC**	**SROCC**	**RMSE**	**OR**
Plane-fitting, *k*-nn, $k = 256$	0.936	0.925	0.403	0.589
Plane-fitting, range, R = 5	0.924	0.912	0.436	0.644
Quadric-fitting, range, R = 20	0.934	0.920	0.410	0.611
D1 PSNR [5]	0.868	0.855	0.540	0.752
D2 PSNR [5]	0.913	0.910	0.443	0.588

Table 1: Performance evaluation of the plane-to-plane metric in JPEG-ES2-PCCD, using the best performing configurations for all normal estimation algorithms that were selected.

In Figure 5, scatterplots indicating the prediction performance of the metric are illustrated for plane fitting using *k*-nn, and plane and quadric fitting using range search. It is easy to conclude that using plane and quadric fitting with range search, we obtain very similar results. Using the *k*-nn algorithm to

form neighborhoods, though, might lead to poor generalization capabilities (i.e., across different contents), especially when a small k is used. As the neighborhood size is increasing, such deviations are reduced. Moreover, with larger radii, the similarity scores spanning-range is decreasing, which is reasonable if we consider that the estimated surfaces are getting smoother as the regions over which they are estimated is increasing. Thus, less differences and higher similarity scores are obtained.

Figure 5: Subjective against objective quality scores from the plane-to-plane metric. In the first row, plane fitting with k-nn is used to estimate normal vectors, with k = 8, 64 and 512. In the second row and third rows, plane and quadric fitting are used with radius equal to 5, 20 and 50 from left to right, respectively.

3.5.2 M-PCCD

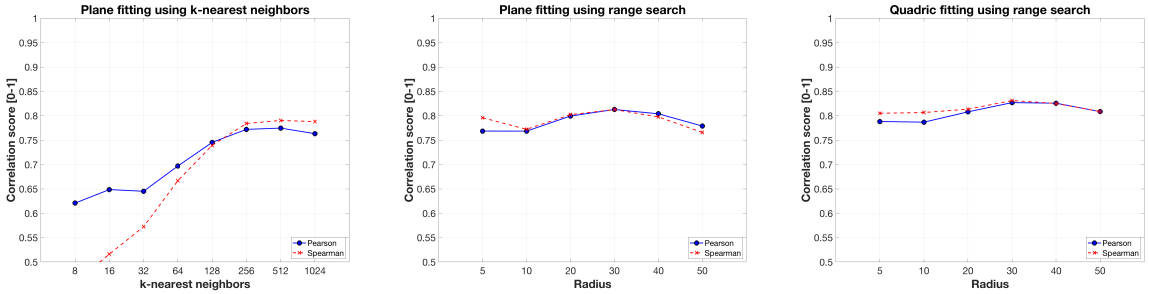

Figure 6: Pearson and Spearman correlation coefficients from benchmarking of the plane-to-plane metric using M-PCCD, as a function of the region size defined to estimate the normal vectors.

In Figure 6, the Pearson and the Spearman correlation coefficients are depicted as a function of the neighborhood size per normal estimation algorithm, against the M-PCCD subjective scores. Similar

performance trends are observed when compared to the results obtained using the JPEG-ES2-PCCD. The prediction power of the metric is lower in this case, however, this is a general observation for all metrics, since the types of contents, as well as the number of models and codecs that are evaluated in this dataset is larger, constituting a more challenging benchmarking setup.

In Table 2 we report the performance indexes for the best performing configurations of all normal estimation algorithms that were tested, along with the D1 PSNR and D2 PSNR metrics for comparison purposes.

Normal estimation algorithm	**PLCC**	**SROCC**	**RMSE**	**OR**
Plane-fitting, *k*-nn, *k* = 512	0.775	0.791	0.862	0.802
Plane-fitting, range, R = 30	0.813	0.813	0.794	0.832
Quadric-fitting, range, R = 30	0.827	0.831	0.766	0.819
D1 PSNR [6]	0.720	0.759	0.885	0.819
D2 PSNR [6]	0.756	0.807	0.834	0.852

Table 2: Performance evaluation of the plane-to-plane metric in M-PCCD, using the best performing configurations for all normal estimation algorithms that were selected.

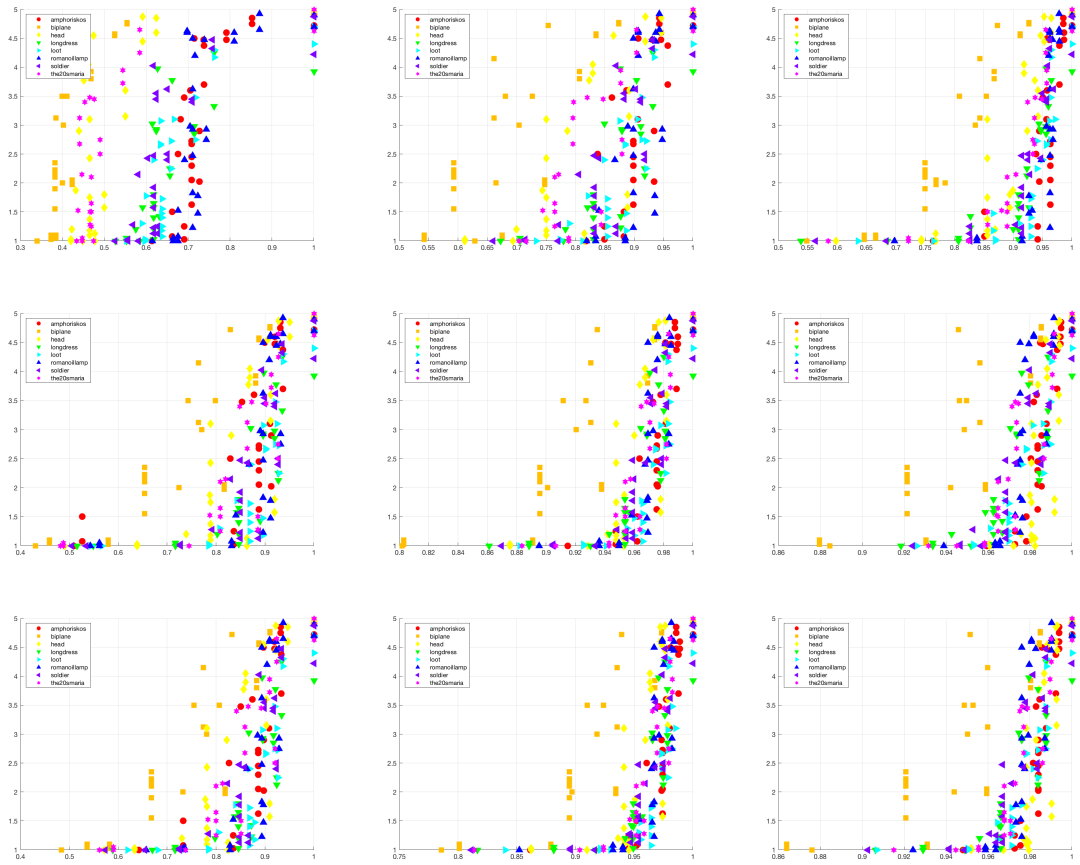

Figure 7: Subjective against objective quality scores from the plane-to-plane metric. In the first row, plane fitting with *k*-nn is used to estimate normal vectors, with *k* = 8, 64 and 512. In the second row and third rows, plane and quadric fitting are used with radius equal to 5, 20 and 50 from left to right, respectively.

In Figure 7, scatterplots indicating the prediction performance of the metric are illustrated for plane fitting using *k*-nn, and plane and quadric fitting using range search. Similar results are observed with respect to the analysis conducted for the JPEG-ES2-PCCD. Regarding the *head* and especially the *biplane* models, it is noteworthy that their geometry is highly irregular and noisy. Thus, these contents behave as outliers in our results, and have a negative impact in the overall performance.

3.6 Discussions

To have a better understanding about the differences in neighborhood formulation between the two approaches (i.e., *k*-nn and range search), in Figure 8 we indicatively present the average and the standard deviation of the number of points of local neighborhoods for the *longdress* model, and its encoded version with Octree at R02, following the MPEG CTC [12]. It can be seen that the number of points using range search exceeds the number of points employed with the *k*-nn for a radius higher than 10 for the reference model, while an order of magnitude deviations are observed when the neighborhood populations are compared to the encoded model. Note that this encoded stimulus is the second sparsest in the dataset.

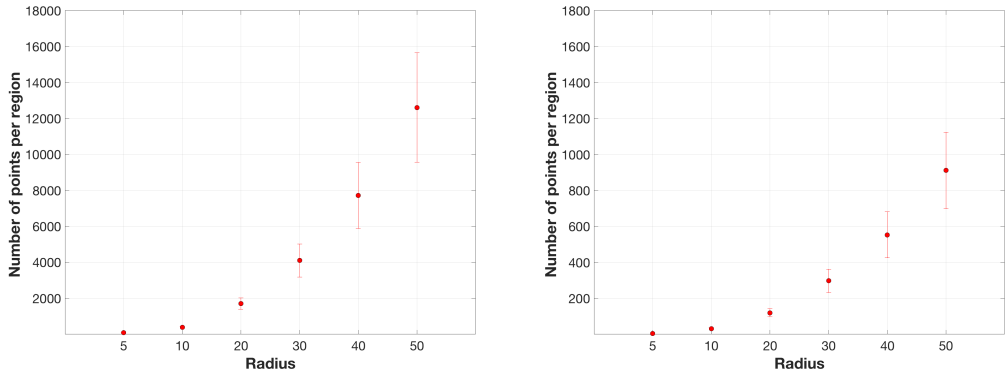

Figure 8: Mean and standard deviation of the point number of local neighborhoods when using range search, for different variations of the *longdress* model: reference and encoded using Octree at R02.

A natural question is, how the performance is improving as the local neighborhoods are increasing, when in the *verification* section 2 we showed that by enlarging the local regions, the normal estimation error is rapidly increasing. In fact, by increasing the neighbourhood size, the local surface approximations are smoother, which can be interpreted as applying a low pass filter; note, that this doesn't necessarily mean more accurate. However, it is evident that by estimating the local surfaces from small neighborhoods of 10 or 20 points, high frequency geometric components are reflected on the estimated normal vectors, which do not represent the shape, or how we perceive the shape of the models.

A visualization example is provided in Figure 9. The model is rendered using point shading assisted by the estimated normal vectors under ambient light. In this illustration, a plane fitting with the *k*-nn approach is used, and results from all the selected *k* values are presented. As we can see, by increasing the size of the local neighborhoods over which the normal vectors are estimated, the underlying surfaces are getting smoother.

In Figure 10, different versions of the *longdress* model along with corresponding quality scores are also provided as examples.

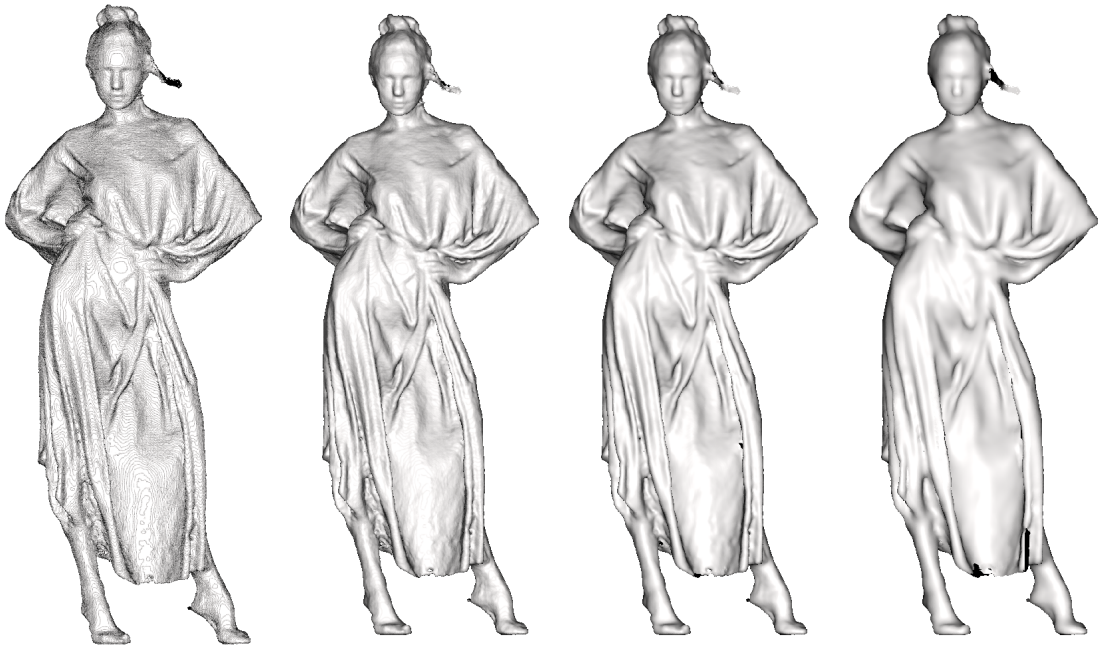

Figure 9: Visualization of the reference *longdress* with point shading. The normal vectors are estimated with plane fitting and k = 8, 32, 128 and 512 from left to right.

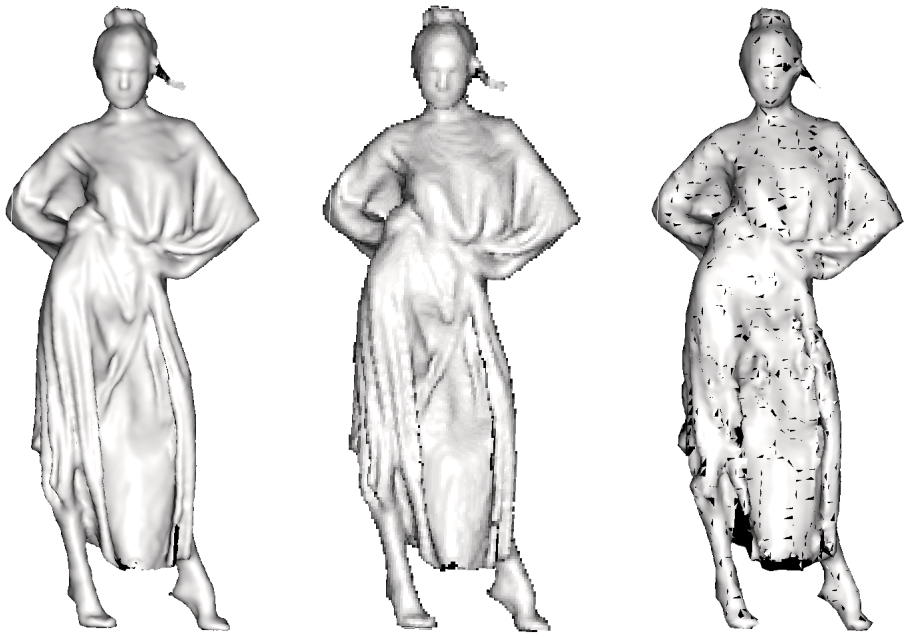

Figure 10: Three versions of the *longdress* model: reference, stimulus after encoding with Octree at R02, and stimulus after encoding with TriSoup at R01 following the MPEG CTC [12], from left to right. The normal vectors are estimated using plane fitting with range search and radius 10. The obtained plane-to-plane similarity scores are 0.8830 and 0.7958, for the second and third stimulus, respectively.

4. Conclusion

Based on our results, the performance of the plane-to-plane metric is crucially affected by the region size over which the normal vectors are estimated. In fact, we found that by enlarging the neighborhoods (up to a certain size), the performance is improving for the employed datasets. This result is counterintuitive, considering that larger neighborhoods typically lead to larger normal estimation errors, as shown in section 2. However, larger neighborhoods lead to reconstruction of smoother surfaces and act as low-pass filters, which eliminate high-frequency geometric components in local scales that do not reflect how we perceive the shape of a model. Notably, optimizations to achieve lower normal estimation errors do not necessarily lead to better performance for the plane-to-plane metric.

Following our results, the use of the quadric fitting model with range search and a radius equal to 20 is found to be the best option, under the employed datasets. In particular, the performance of this fitting model is quite robust for different radii, and it shows consistency in quality predictions across different. Despite the fact that the range search doesn't guarantee the same neighborhood population, it grands that the samples falling in "the same" spatial sub-space will be used to estimate the normal vectors for both the reference and the distorted models. In our context, it is reasonable to favor the approximation of underlying surfaces under comparison in local regions that best-match.

References

[1] Mitra, Niloy J., and An Nguyen. "Estimating surface normals in noisy point cloud data." In *Proceedings of the nineteenth annual symposium on Computational geometry*, pp. 322-328. 2003.

[2] Klasing, Klaas, Daniel Althoff, Dirk Wollherr, and Martin Buss. "Comparison of surface normal estimation methods for range sensing applications." In 2009 IEEE international conference on robotics and automation, pp. 3206-3211. IEEE, 2009.

[3] Jordan, Krzysztof, and Philippos Mordohai. "A quantitative evaluation of surface normal estimation in point clouds." In 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4220-4226.

[4] Alexiou, Evangelos, and Touradj Ebrahimi. "Point cloud quality assessment metric based on angular similarity." In 2018 IEEE International Conference on Multimedia and Expo (ICME), pp. 1-6.

[5] Perry, Stuart, Huy P. Cong, Luís A. da Silva Cruz, João Prazeres, Manuela Pereira, Antonio Pinheiro, Emil Dumic, Evangelos Alexiou, Touradj Ebrahimi. "Quality evaluation of static point clouds encoded using MPEG codecs", In 2019 IEEE International Conference on Image Processing (ICIP), pp. 4325-4329.

[6] Alexiou, Evangelos, Irene Viola, Tomás M. Borges, Tiago A. Fonseca, Ricardo L. de Queiroz, and Touradj Ebrahimi. "A comprehensive study of the rate-distortion performance in MPEG point cloud compression." APSIPA Transactions on Signal and Information Processing 8 (2019).

[7] Alexiou, Evangelos, Nanyang Yang, and Touradj Ebrahimi. "PointXR: A toolbox for visualization and subjective evaluation of point clouds in virtual reality." In 2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX), Athlone, Ireland, 2020, pp. 1-6.

[8] CloudCompare (version 2.10.1) [GPL software]. (2020). Retrieved from [URL].

[9] Hoppe, Hugues, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. "Surface reconstruction from unorganized points." In Proceedings of the 19th annual conference on Computer graphics and interactive techniques, pp. 71-78. 1992.

[10] Cignoni, Paolo, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. "Meshlab: an open-source mesh processing tool." In Eurographics Italian chapter conference, vol. 2008, pp. 129-136. 2008.

[11] Rusu, Radu Bogdan, and Steve Cousins. "3d is here: Point cloud library (pcl)." In 2011 IEEE international conference on robotics and automation, pp. 1-4. IEEE, 2011.

[12] MPEG 3DG, "Common test conditions for point cloud compression," ISO/IEC JTC1/SC29/WG11 Doc. N18474, Geneva, Switzerland, Mar 2019.

[13] Tian, Dong, Hideaki Ochimizu, Chen Feng, Robert Cohen, and Anthony Vetro. "Geometric distortion metrics for point cloud compression." In 2017 IEEE International Conference on Image Processing (ICIP), pp. 3460-3464. IEEE, 2017.