Automated Point Cloud Filtration Through Minimization of Point Cloud Metrics

Michael Holm, Eliot Winer; Iowa State University; Ames, IA

Abstract

Point clouds generated from 3D scans of part surfaces consist of discrete points, some of which may be outliers. Filtering techniques to remove these outliers from point clouds frequently require a "guess and check" method to determine proper filter parameters. This paper presents two novel approaches to automatically determine proper filter parameters using the relationships between point cloud outlier removal, principal component variance, and the average nearest neighbor distance. Two post-processing workflows were developed that reduce outlier frequency in point clouds using these relationships. These post-processing workflows were applied to point clouds with artificially generated noise and outliers, as well as a real-world point cloud. Analysis of the results showed the approaches effectively reduced outlier frequency while preserving the ground truth surface, without requiring user input.

Introduction

3D scans are often stored as point clouds, which consist of discrete 3D points irregularly sampled from continuous surfaces [1]. While advancements have been made in acquiring and storing point cloud data, there are still several issues that exist with their effective utilization in many applications. One of the most prominent issues is the existence of outliers in a point cloud, resulting from overexposure, reflective surfaces, and user error [2], [3].

Outlier points are false measurements that do not belong to the scanned surface, have geometrical discontinuities, and sharp features [4]. Figure 1 shows a point cloud of an automobile with many outliers, shown in red.

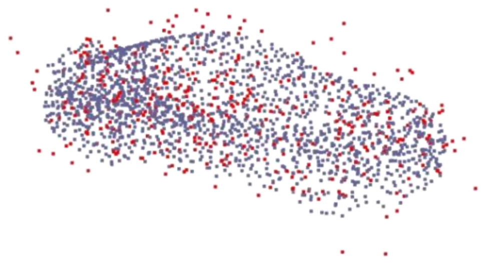

Figure 1: A point cloud with many outliers, shown in red

A scanned point cloud can contain both outliers and noise, caused by different sources. The difference between the two is that noise refers to random variations in the positional data of a scanned point, while outliers deviate significantly from the expected distribution of the scanned data [3].

Outliers create problems during point cloud registration, meshing, and normal estimation [5], [6], [7]. To address the challenges outliers create for point cloud applications, several open-source application programming interfaces (APIs) have been developed from various research projects such as the Point Cloud Library [8], MeshLab [9], Open3D [10], and cilantro [11]. These APIs include filters that are intended to remove outliers without affecting the scanned surface data (i.e. the ground truth surface) in point clouds.

To effectively use a filter provided from one of these APIs, a user must determine appropriate filtration parameters so that the filtered point cloud is an accurate representation of the scanned object. This process requires a user to guess what parameters may create a point cloud that contains fewer outliers while maintaining "good" points from the ground truth surface, and then adjust their guess based on subjective visualization of the resulting filtered point cloud.

Newly developed APIs provide tools to filter point clouds, but they do not address issues related to a user's ability to select the appropriate filtration parameters to remove outliers. Many outlier filtration methods have been researched, traditionally requiring input parameters provided by an expert user. The performance of these filters is heavily dependent on the selection of suitable parameters.

Background

Post-processing Software Advances

Post-processing to improve the quality of the point cloud representation of the scanned object can be very time consuming. Raw point clouds (i.e., point clouds that haven't been post-processed) often require a combination of noise reduction [12], file size reduction [13], and 3D mesh creation [14]. The Point Cloud Library, commonly known as PCL, [8] has made concerted efforts to help users choose point cloud post-processing techniques.

Along with these post-processing techniques, PCL also implements many statistical metrics that can be used to describe aspects of a point cloud. These include metrics like bounding box size and location of a point cloud's centroid. One of these metrics is known as the nearest neighbor average distance. This metric samples one point, measures the Euclidean distance to its nearest neighbor in the point cloud, repeats this process for every point in the point cloud, and averages all these distances. The nearest neighbor average distance is then calculated using Equation 1, where $\overline{NND_P}$ represents the nearest neighbor average distance of a point cloud P and $|P|$ represents the number of points in the point cloud P.

$$\overline{NND_P} = \frac{\sum_{i=1}^{|P|} Distance\ from\ P_i\ to\ P_i's\ nearest\ neighbor}{|P|} \tag{1}$$

Another statistical analysis method implemented in PCL is principal component analysis (PCA). This algorithm, which is well established in data science, can be used on a point cloud to find three principal components, along with the percentage of variance accounted for by each component. PCA is used in point cloud research to align point clouds along their principal component axes [15], [16].

While PCL provides useful point cloud post-processing tools, competitors have also worked to address the need to help users choose appropriate post-processing tool(s). In 2018, one of these competitors, Open3D [17], was published to make point cloud post-processing tools easier to use.

Statistical Outlier Removal Filter

Open3D includes a statistical outlier removal filter that requires two input arguments called "number of neighbors" and "standard ratio". The filter removes all points whose average distance to the nearest "number of neighbors" lies more than "standard ratio" standard deviations outside the average for the entire point cloud. This process can be defined as shown in Equation 2. In this equation, P_n represents the raw, unfiltered cloud, P_i represents any single point in P_n and P_f represents the resulting filtered point cloud. In addition, *x* represents the value given for the "number of neighbors" argument, and *y* represents the value given for the "standard ratio" argument.

$$P_f = \{P_n \mid \text{The average distance of } P_i \text{ to its nearest x neighbors is within y standard deviations of the overall distribution}\} \tag{2}$$

This filter has been shown to reduce the number of outliers in a point cloud, but only when used with the correct parameters [18]. Finding these correct parameters requires an iterative approach, in which a user must repeatedly guess filter parameters until satisfactory filtration results are achieved.

Point Cloud Databases

Parallel to the development of these advances in post-processing, several publicly available point cloud databases were created [19], [20]. One of these databases, the MVP Point Cloud Database, was created to provide a benchmark for the point cloud community to use for various applications, including point cloud completion, point cloud registration, and other standard processes [21]. This database consists of 2,400 point clouds of varying geometry, each with 2,048 uniformly sampled points across the point cloud. An example of a point cloud from this database is shown in Figure 2.

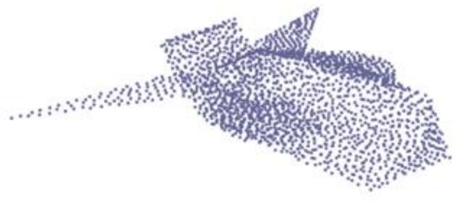

Figure 2: A point cloud of a ship from the MVP point cloud database

In addition to the MVP database, a database of real-world point clouds was introduced by Wolff et al. [22]. This database contains several real-world point clouds collected using photogrammetry. These point clouds include significant noise and outliers. Both the MVP database and Wolff's database were used to evaluate the proposed filtration methods.

Point Cloud Comparison Metrics

Research groups have created several metrics to quantitatively measure the visual difference between two point clouds [23], [24]. Chamfer distance is a numerical measurement for the visual difference between two points clouds (e.g., two point clouds "A" and "B") [25], [21]. For each point in point cloud A, a corresponding point in point cloud B is found, such that the Euclidean distance between the two points is a minimum. This distance is then squared and added together, for every point in point cloud A. This process is repeated for point cloud B, and finally, the two sums are added together.

Even with all the improvements in point cloud processing, there are still areas of potential improvement that need to be addressed, including point cloud filters whose filtration performance does not rely on user input and standardized databases for evaluation of new methods.

Methodology

This section describes the approaches used to filter point clouds without requiring user input and the methods used to evaluate the filter performance. For each artificial point cloud that was filtered using these automatic approaches, a percentage error was found between the automatically filtered point cloud and an estimate of the best filtration results possible. The approaches used to automatically find acceptable point cloud filtration parameters developed by the authors used the Open3D statistical outlier removal filter.

Automatic Filter Parameter Estimation

The approaches to determine acceptable parameters for Open3D's statistical outlier removal filter plotted the statistical outlier removal filter's parameters against two metrics of the resulting filtered point cloud. One of these metrics was the percentage of the filtered point cloud's variance accounted for by the principal component with the least variance. When PCA is performed on a dataset, the output is an uncorrelated set of principal components. After finding these components, the one that accounted for the smallest percentage of variance in the cloud was isolated. This percentage, called the "principal component least variance ratio" was then plotted against the two parameters of the statistical outlier removal filter, using matplotlib's 3D plot feature [26].

For the surface created, the minimum was found, and the "number of neighbors" and "standard ratio" associated with that minimum were used as parameters for Open3D's statistical outlier removal filter. This filter was used to produce a final, filtered point cloud by using these filtration parameters at this minimum location to clean the original, unfiltered point cloud.

The second metric investigated in this manner was the nearest neighbor average distance. A function was created to plot the statistical outlier removal filter parameters against this average distance. This surface was then plotted using the same tools and techniques to plot the principal component least variance ratio surface described above. An example of this surface is shown in Figure 3. In this figure, the surface minimum is indicated by a red star. Again, the parameters associated with the minimum of this surface were used to filter the original, unfiltered point cloud.

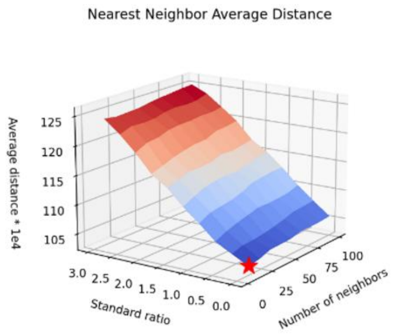

Figure 3: An example of the nearest neighbor average distance surface created in this process, with the surface minimum indicated by a red star

Synthetic Noise and Outlier Generation

To evaluate the automated point cloud filtration approaches proposed using the MVP point cloud database, a point cloud from the MVP Point Cloud Database was corrupted using an algorithm similar to past research [27]. This corruption algorithm randomly selected a percentage of points in the MVP point cloud, equal to the corruption percent parameter, and moved these points in a random direction. The extent of this movement was determined using a normal distribution, whose mean was zero and whose standard deviation, σ, was found by multiplying the length of the MVP point cloud's 3D diagonal by some predetermined magnitude multiplier parameter.

Multiple values for the magnitude multiplier and corruption percentage parameters were used and presented in the results section. An image of a point cloud with noise and outliers generated through this process, (referred to as a "corrupted cloud"), can be seen in Figure 4. For this figure, a magnitude multiplier and corruption percentage argument of 0.05, and 0.2 respectively were used to generate outliers.

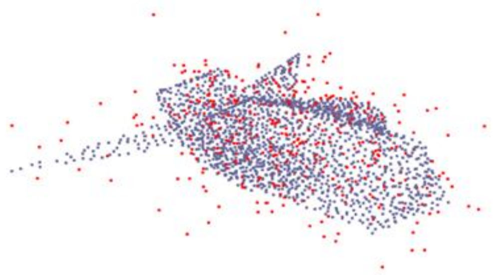

Figure 4: A corrupted point cloud generated from the MVP database with generated noise and outliers shown in red

Synthetic Filter Performance Measurement

To quantitatively measure the performance of these automated approaches when used on synthetic point clouds, a metric was created to assess how well these approaches filtered a corrupted MVP point cloud. This metric, called the summary statistic, was defined using Equations 3-7.

$$\text{RMSE} = \frac{\sqrt{\frac{\Sigma_{p_i' \in P_f p_i \in P_o} (\text{distance between } p_i \text{ and } p_i')^2}{|P_f|}}}{\text{nearest neighbor average distance in original cloud}} \quad (3)$$

$$\text{Precision} = \frac{|\text{noise and outlier points removed by filter}|}{|\text{points removed by filter}|} * 100 \quad (4)$$

$$\text{Recall} = \frac{|\text{noise and outlier points removed by filter}|}{|\text{corrupted points in original corrupted point cloud}|} * 100 \quad (5)$$

$$\text{Accuracy} = \frac{|\text{corrupted points removed by filter}| + |\text{correct points left}|}{|\text{points in original point cloud}|} * 100 \quad (6)$$

$$\text{Summary statistic} = \frac{\text{avg(precision, recall, accuracy)}}{\text{RMSE}} \quad (7)$$

The definition of RMSE, precision, recall, and accuracy in the above equations are derived from past research [28], [29]. In the RMSE equation (3), P_f and $|P_f|$ represent the filtered MVP point cloud and the number of points in the point cloud, respectively. Additionally, P_o represents the ground truth MVP point cloud. If a point $p_i' \notin P_f$, but $p_i \in p_o$, because of p_i being filtered out, that index i is not used in the RMSE calculation.

To assess filter performance on synthetically corrupted point clouds, the summary statistic was computed twice. Once for the automatically filtered MVP point cloud and the second for a point cloud using the best possible parameters for the statistical outlier removal filter. Then, the percentage error between these two summary statistic values was computed. These best possible parameters were found by selecting the filtration parameters that resulted in a maximum for the summary statistic.

The chamfer distance between the ground truth point cloud and a filtered point cloud was also used as a performance measurement. In this case, the best possible parameters were found using the minimum of the chamfer distance.

Evaluation of Automatic Parameter Selection

The proposed approaches were evaluated by first corrupting a random MVP point cloud using the process described in the Synthetic Noise and Outlier Generation section. This corrupted point cloud was then filtered using the minimum nearest neighbor average distance approach or the minimum principal component least variance ratio approach.

These filtered point clouds were compared to an optimally filtered point cloud, following the summary statistic and chamfer distance metrics described in the Automatic Filter Performance Measurement section. This process was repeated ten times. The percentage errors were averaged, and the process was repeated for 49 additional MVP point clouds. These 50 average errors were again averaged together. Analysis of these point clouds was repeated five separate times with different amounts and magnitudes of corruption.

Finally, when evaluating filtration performance on the real-world point cloud that was filtered using these methods, the summary statistic and the chamfer distance evaluation methods could not be used. These methods require knowledge of a point cloud's ground truth surface, which was not available for the real-world point cloud. As a result, the filtration of the real-world point cloud was evaluated from a visual perspective as well as quantitatively, by comparing the percentage of points removed by the minimum nearest neighbor average distance approach, compared to the percentage of points removed by the filter presented by Wolff et al. [22].

Results

Table 1 shows results from the MVP point cloud evaluation process described in the Evaluation of Automatic Parameter Selection section. Each row shows the average results of 500 experimental runs, using 50 point clouds. In total the results from 2,500 experimental runs are shown, using the minimum nearest neighbor average distance approach.

Table 1: Filtration results from 50 randomly sampled MVP Point Cloud

Corruption Percentage	Magnitude Multiplier	Average Summary Statistic Error	Summary Statistic Standard Deviation	Average Chamfer Distance Error	Chamfer Distance Standard Deviation
30%	.05	4.17%	2.06%	30.86%	25.32%
20%	.1	6.12%	3.00%	22.22%	40.64%
20%	.05	8.17%	3.65%	70.96%	42.79%
20%	.01	7.66%	3.76%	214.17%	115.74%
10%	.05	15.07%	5.21%	249.37%	154.88%

This automatic approach had high error when used on point clouds with a small amount and magnitude of outliers. As the magnitude and number of outliers increased in a corrupted MVP point cloud, the average percentage error dropped. Additionally, the error percentage in the summary statistic generally followed the direction of the error in the chamfer distance, but the amount of change between different levels of noise was greater for the chamfer distance error percentage.

When performing this experimental process using the minimum principal component least variance ratio approach, the same trends appeared. That is, higher corruption magnitudes and percentages resulted in better performance. However, the percentage errors and standard deviations were much higher when using the minimum principal component least variance ratio approach.

Figures 5 and 6 show a point cloud being filtered using the minimum nearest neighbor average distance approach. When filtering this point cloud, a chamfer distance error of 80% was recorded. This demonstrates that, although high chamfer distance and summary statistic errors were recorded when using these automated approaches, the filtered point clouds still contained fewer outliers, while maintaining the ground truth surface.

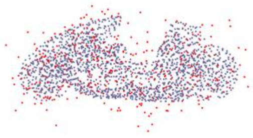
Figure 5: Corrupted MVP point cloud to be filtered using the minimum nearest neighbor average distance approach

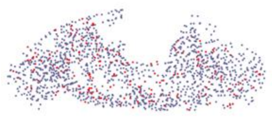
Figure 6: Filtered MVP point cloud, filtered using the minimum nearest neighbor average distance approach

These automatic outlier filtration approaches were also tested on a real world point cloud presented by Wolff et al. [22]. Figures 7 and 8 show the real-world point cloud before and after filtration using the proposed minimum nearest neighbor average distance approach.

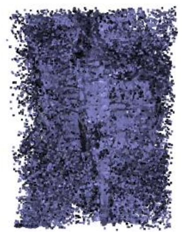
Figure 7: Original real-world point cloud

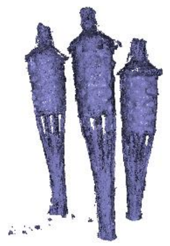
Figure 8: Real-world point cloud filtered with minimum nearest neighbor average distance approach

From a qualitative perspective, the results of this automatic filtration using the minimum nearest neighbor average distance are comparable to the filter presented by Wolff et al. [22]. Conversely, when using the minimum principal component least variance ratio approach, many outliers were left in the filtered point cloud. Quantitatively, the minimum nearest neighbor average distance approach removed 28.08% of points in the original point cloud, compared to the 69.56% of points removed by the filter presented by Wolff et al. [22].

This difference in percentages is due to these proposed approaches not effectively removing surface level noise, unlike Wolff's filter. However, Wolff's filter additionally requires information regarding the color of all points in a point cloud. A user does not always have this information, in which case, these proposed approaches would provide a good alternative method to effectively remove outliers from the point cloud. Additionally, it is possible that Wolff's filter was overly aggressive, removing more points than necessary.

Conclusions & Future Work

This paper presented automatic methods for determining appropriate parameters for Open3D's statistical outlier removal filter. The proposed approaches use relationships between a point cloud's nearest neighbor average distance and principal component least variance ratio. The effectiveness of these approaches was investigated using the MVP point cloud database and a real-world point cloud with noise and outliers.

Through this investigation, it was concluded that these approaches tend to work better as the quantity and magnitude of outliers present in a corrupted point cloud increase. These findings were supported by a wide array of tests performed on these filtration techniques. Additionally, it was concluded that these approaches do not effectively remove surface level noise.

This research presents potential for future work. Plotting a full grid of points every time a point cloud needs to be filtered is a computationally intensive task. These approaches could instead be redesigned to use a gradient descent technique.

References

[1] S. Luo and W. Hu, "Score-Based Point Cloud Denoising," 2021.

[2] Y. Wang and H. Y. Feng, "Effects of scanning orientation on outlier formation in 3D laser scanning of reflective surfaces," *Opt. Lasers Eng.*, vol. 81, pp. 35–45, Jun. 2016, doi: 10.1016/J.OPTLASENG.2016.01.003.

[3] N. Charron, S. Phillips, and S. L. Waslander, "De-noising of Lidar Point Clouds Corrupted by Snowfall," in *2018 15th Conference on Computer and Robot Vision (CRV)*, May 2018, pp. 254–261. doi: 10.1109/CRV.2018.00043.

[4] X. Ning, F. Li, G. Tian, and Y. Wang, "An efficient outlier removal method for scattered point cloud data," *PLOS One*, Aug. 2018, doi: 10.1371/journal.pone.0201280.

[5] C. Stucker, A. Richard, J. D. Wegner, and K. Schindler, "SUPERVISED OUTLIER DETECTION IN LARGE-SCALE MVS POINT CLOUDS FOR 3D CITY MODELING APPLICATIONS," *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.*, vol. IV–2, pp. 263–270, May 2018, doi: 10.5194/isprs-annals-IV-2-263-2018.

[6] L. Yan, P. Wei, H. Xie, J. Dai, H. Wu, and M. Huang, "A New Outlier Removal Strategy Based on Reliability of Correspondence Graph for Fast Point Cloud Registration," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–17, 2022, doi: 10.1109/TPAMI.2022.3226498.

[7] X. Yuan, H. Chen, and B. Liu, "Point cloud clustering and outlier detection based on spatial neighbor connected region labeling," *Meas. Control*, vol. 54, no. 5–6, pp. 835–844, May 2021, doi: 10.1177/0020294020919869.

[8] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," *IEEE Int. Conf. Robot. Autom.*, 2011, [Online]. Available: http://pointclouds.org

[9] P. Cignoni *et al.*, "MeshLab: an Open-Source Mesh Processing Tool. Automatic 3D scanning View project Virtual reconstruction of the entrance of Ripoll's Monastery View project MeshLab: an Open-Source Mesh Processing Tool," *Eurographics Ital. Chapter Conf.*, 2008, doi: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[10] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A Modern Library for 3D Data Processing," Jan. 2018.

[11] K. Zampogiannis, C. Fermüller, and Y. Aloimonos, "cilantro: A Lean, Versatile, and Efficient Library for Point Cloud Data Processing," *ACM Int. Conf. Multimed.*, 2018, doi: 10.1145/3240508.3243655.

[12] X. F. Han, J. S. Jin, M. J. Wang, W. Jiang, L. Gao, and L. Xiao, "A review of algorithms for filtering the 3D point cloud," *Signal Process. Image Commun.*, vol. 57, pp. 103–112, Sep. 2017, doi: 10.1016/J.IMAGE.2017.05.009.

[13] C. Suchocki and W. Błaszczak-Bąk, "Down-Sampling of Point Clouds for the Technical Diagnostics of Buildings and Structures," *Geosciences*, vol. 9, no. 2, Art. no. 2, Feb. 2019, doi: 10.3390/geosciences9020070.

[14] M. Zwicker and C. Gotsman, "Meshing Point Clouds Using Spherical Parameterization," *Eurographics Symp. Point-Based Graph.*, 2004.

[15] C. Yuan, X. Yu, and Z. Luo, "3D point cloud matching based on principal component analysis and iterative closest point algorithm," in *2016 International Conference on Audio, Language and Image Processing (ICALIP)*, Jul. 2016, pp. 404–408. doi: 10.1109/ICALIP.2016.7846655.

[16] X. Chen and C.-H. Chen, "Model-based point cloud alignment with principle component analysis for robot welding," in *2017 International Conference on Advanced Robotics and Intelligent Systems (ARIS)*, Sep. 2017, pp. 83–87. doi: 10.1109/ARIS.2017.8297194.

[17] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A Modern Library for 3D Data Processing," Jan. 2018, [Online]. Available: http://arxiv.org/abs/1801.09847

[18] "Point cloud outlier removal - Open3D primary (0cf605f) documentation." Accessed: Feb. 27, 2024. [Online]. Available: https://www.open3d.org/docs/latest/tutorial/geometry/pointcloud_outlier_removal.html

[19] D. Griffiths and J. Boehm, “A review on deep learning techniques for 3D sensed data classification,” arXiv.org. Accessed: Feb. 20, 2024. [Online]. Available: https://arxiv.org/abs/1907.04444v1

[20] S. Qiu, S. Anwar, and N. Barnes, “Geometric Back-projection Network for Point Cloud Classification,” arXiv.org. Accessed: Feb. 20, 2024. [Online]. Available: https://arxiv.org/abs/1911.12885v5

[21] L. Pan *et al.*, “Variational Relational Point Completion Network,” 2021. doi: 10.1109/CVPR46437.2021.00842.

[22] K. Wolff *et al.*, “Point Cloud Noise and Outlier Removal for Image-Based 3D Reconstruction,” 2016.

[23] F. Mémoli and G. Sapiro, “Comparing point clouds,” in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, Nice France: ACM, Jul. 2004, pp. 32–40. doi: 10.1145/1057432.1057436.

[24] D. Urbach, Y. Ben-Shabat, and M. Lindenbaum, “DPDist : Comparing Point Clouds Using Deep Point Cloud Distance,” arXiv.org. Accessed: Feb. 20, 2024. [Online]. Available: https://arxiv.org/abs/2004.11784v2

[25] M. J. Rakotosaona, V. La Barbera, P. Guerrero, N. J. Mitra, and M. Ovsjanikov, “PointCleanNet: Learning to Denoise and Remove Outliers from Dense Point Clouds,” *Comput. Graph. Forum*, vol. 39, no. 1, pp. 185–203, Feb. 2020, doi: 10.1111/CGF.13753.

[26] J. D. Hunter, “Matplotlib: A 2D graphics environment,” Computing in Science & Engineering. Accessed: May 18, 2022. [Online]. Available: https://matplotlib.org/

[27] M.-J. Rakotosaona, V. L. Barbera, P. Guerrero, N. J. Mitra, and M. Ovsjanikov, “PointCleanNet: Learning to Denoise and Remove Outliers from Dense Point Clouds,” *Comput. Graph. Forum*, vol. xx, pp. 1–17, 2020.

[28] H. Daghigh, D. D. Tannant, V. Daghigh, D. D. Lichti, and R. Lindenbergh, “A critical review of discontinuity plane extraction from 3D point cloud data of rock mass surfaces,” *Comput. Geosci.*, vol. 169, p. 105241, Dec. 2022, doi: 10.1016/j.cageo.2022.105241.

[29] N. Diaz, O. Gallo, J. Caceres, and H. Porras, “Real-time ground filtering algorithm of cloud points acquired using Terrestrial Laser Scanner (TLS),” *Int. J. Appl. Earth Obs. Geoinformation*, vol. 105, p. 102629, Dec. 2021, doi: 10.1016/j.jag.2021.102629.

Author Biography

Michael Holm received his BS in Mechanical Engineering from Iowa State University (2022) and his MS in Mechanical and Computer Engineering from Iowa State University (2024). He is now attending Purdue University, researching the application of machine learning towards the efficient analysis of extreme-scale experimental data of plasma facing materials in a nuclear reactor.

Eliot Winer, Ph.D., is the director of the VRAC and professor of Mechanical Engineering, Electrical and Computer Engineering, and Aerospace Engineering at Iowa State University. Dr. Winer has over 25 years of experience working in extended reality and 3D computer graphics technologies on sponsored projects for the Department of Defense, Air Force Office of Scientific Research, National Science Foundation, Department of Agriculture, Boeing, John Deere, and the Federal Highway Administration.