Proceedings of the ASME 2024
International Design Engineering Technical Conferences and
Computers and Information in Engineering Conference
IDETC/CIE2024
August 25–28, 2024, Washington, DC

DETC2024-143913

A GEOMETRICO-STATISTICAL REPRESENTATION FOR OFF-ROAD ENVIRONMENTS CAPTURED BY LIDAR

Taimoor Daud Khan[1], Shantanu Vyas[1], Vinayak R. Krishnamurthy[1,2,*]

[1]J. Mike Walker '66 Department of Mechanical Engineering, Texas A&M University, College Station, TX
[2]Department of Computer Science and Engineering (affiliated), Texas A&M University, College Station, TX

ABSTRACT

In this paper, we present a hierarchical representation of LiDAR point cloud data in off-road environments with the aim of generating 3D scene abstractions for autonomous vehicles. Different objects in the environment are better represented at different levels of detail based on the type of object, the application context and the information available about the object. Despite this, prior works on point cloud scene processing focus on detailed scene rendering and scene reconstruction which are computationally expensive and often require prior training. In addition, these works are centered on structured on-road environments. Unlike on-road environments, off-road scenes consist of highly unstructured and irregularly shaped objects such as rough terrains, trees, and shrubs. As a result, most algorithms for segmenting, reconstructing, and consequently navigating such environments are inherently challenging. Furthermore, the transmission of off road scenes over networks to other vehicles operating in coordination or to remote sites is expensive when working with high fidelity scene representations. In contrast we present a geometric-statistical algorithm that is computationally inexpensive and provides a multi-level of detail scene abstraction for robot decision making and task execution. The low footprint of the abstraction also allows for lossless scene transmission over the network. We present qualitative results for scene abstractions for various outdoor scenes in the RELLIS-3D dataset and also present quantitative assessment of the abstractions compared to segmented objects in the scenes.

Keywords: Scene Abstraction, Vehicle Perception, Off-Road Environments, Scene Representation

1. INTRODUCTION

1.1 Motivation & Challenges

Technological advancements in remote sensing have led to widespread development of 3D point clouds for object and scene representation. While point clouds have the ability to represent complete three-dimensional information pertaining to a scene, such information is inherently unstructured (i.e. it does not contain topological information) and also partial due to occlusion. As a result, there has been extensive research, especially for autonomous driving, to extract meaningful information through a segmentation [1], [2], [3], reconstruction [4], [5] and object detection techniques [6]. In this work, we specifically focus on the problem of multi-level-of-detail scene representation and present a computational framework for scene abstraction.

Multi-robot ground vehicles in off-road environments present a unique set of challenges that may require a different perspective on how a scene is represented, processed, and transmitted across a network of unmanned robots for navigation tasks. First, off-road environments typically contain a mixture of man-made as well as natural structures. As a result, a complete reconstruction or even semantic segmentation of a given scene from a point cloud is particularly challenging and may come with significant computational overheads, particularly for transmission over lossy networks [7]. Secondly, robot navigation and path-finding tasks are contextual in that planning a path for a given robot may or may not require high-fidelity reasoning about the scene. As a simple example, portions of the path closer to a robot need to be more accurate while the portions farther away can be refined as the robot navigates forward. We argue that a multi-level-of-detail (MLOD) approach is needed to represent the scene at different levels of fidelity, based on how much information is necessary for safe and accurate navigation.

In this work, our aim is to develop such a representation that allows for an effective way to trade-off between the fidelity of representation, the time for computation, and the memory footprint for effective transmission. Toward this aim, we argue for scene representations as abstractions rather than reconstructions without the need for explicit segmentation.

*Corresponding author: vinayak@tamu.edu
Documentation for `asmeconf.cls`: Version 1.37, September 1, 2024.

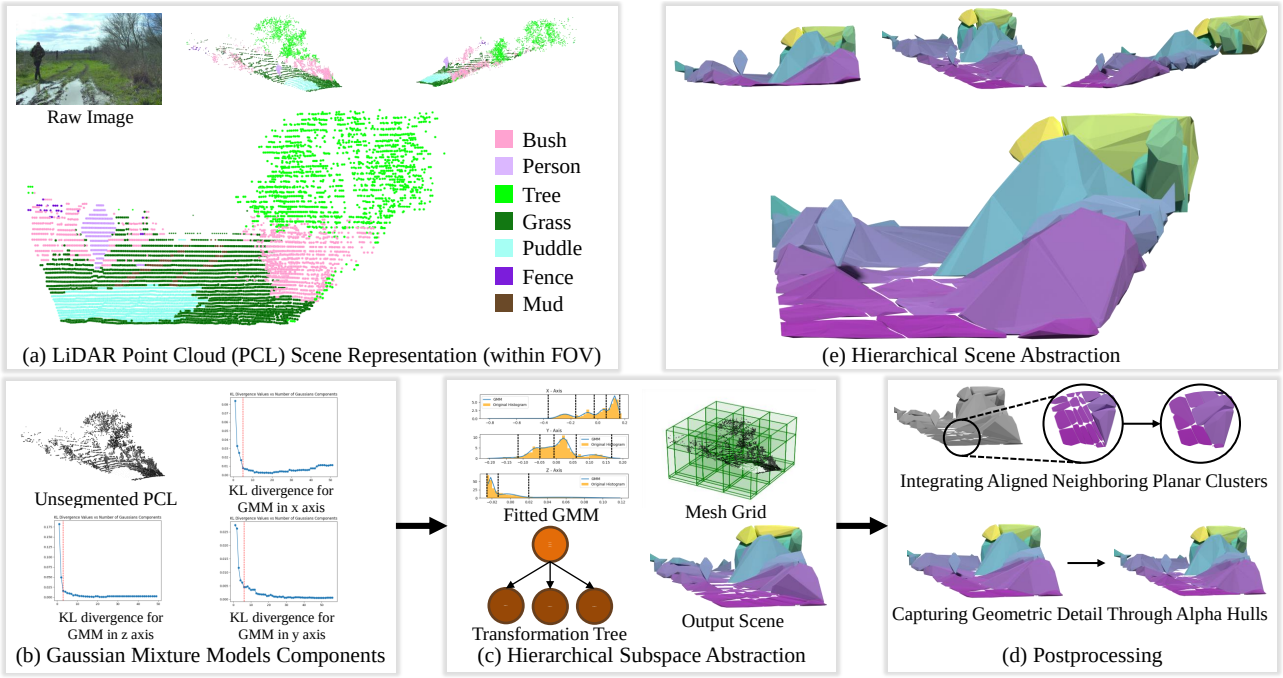


FIGURE 1: Our technical pipeline extracts hierarchical scene abstractions from off-road point clouds as demonstrated using the RELLIS-3D dataset (a). Focused on unsegmented scenes, we exclusively utilize the unsegmented version of the point cloud data. Our fundamental approach focuses on constructing a transformation tree data structure for multilevel of detail scene abstractions. We begin by analyzing point distributions to derive optimal Gaussian mixture components along each principal axis, followed by fitting Gaussian Mixture Models (b, c). Next, we use hierarchical subspace clustering to identify clusters, forming nodes in the transformation tree, which grows hierarchically to increase depth and level of detail in scene abstractions (c). We utilize post-processing techniques for improved geometric representations (d). An example output of our method at transformation tree depth of 3 is shown from different angles (e).

1.2 Background

Most methods involving scene analysis for multi-robot ground vehicles aim for high fidelity scene reconstruction. For example, the use of volumetric representation (voxels) to model vegetation from LiDAR point clouds is demonstrated in one of the early works by Deschaud et al. [8]. However, volumetric representations are inherently expensive both in terms of memory and time in modeling scene feature germane for proper navigation. Recently, splat-based scene reconstruction is used in simulation to model the foreground using splatting techniques after removing dynamic objects (pedestrians, cars, cyclists, etc.) from the scene [9]. Dynamic objects are then added to the scene as synthetic CAD models. While promising, a real-time application of such techniques outside the simulator is still to be investigated. Surface mesh reconstruction for open scenes have also been studied [10]. These methods are sensitive to occlusion, have scalability issues and large memory overheads depending on the size of the point cloud. We observe that based on the application context, task execution can be performed with only relevant partial scene information and full scene semantics and scene reconstruction are meaningless endeavors with high computational and memory overheads.

Most of the techniques developed primarily work with urban structured environments and do not fare well with off road LiDAR data. Data for autonomous off-road driving is marred with unstructured, irregular, and variable density features which prevent the use of models developed for driving in urban centers to be applied to autonomous off-road driving. As mentioned earlier, another challenge with autonomous driving applications that involve data transmission over network is the loss of data in transmission [7]. Transmission of high-fidelity scene representations for cloud based learning or other control and coordination applications over low latency networks exudes a considerable effort. Even though the raw data contains dense information, such information may not necessarily be usable or useful for real-time control and coordination of ground autonomous vehicles.

1.3 Approach and Contribution

Our approach is to develop a methodology to capture a multi-level-of-detail (MLOD) geometric description of the off-road scene to enable meaningful physical assessment of the environment with low computational and memory cost while also maintaining the speed of data transmission. Our premise is not to reconstruct the scene but rather do a quick and transmittable abstraction with a reasonable level of detail for decision making in autonomous off-road driving. In this paper we present a new data structure for scene abstraction using a novel combination of statistical and geometric tools (see Figure 1). The structure also encodes a tree hierarchy to represent the scene at multiple levels of detail. We hierarchically apply probabilistic models of Gaussian Mixtures to analyze point distribution and find groupings of points that abstract certain features and encode them in a transformation tree. In addition, we use an integration and alpha hull module in our algorithmic pipeline to account for abstraction

of variable density regions (trees, shrubs, etc.) as well as planar artifacts (ground). We present the results of our algorithm on the off-road RELLIS-3D point cloud dataset [11] and systematically evaluate the different modules within the algorithm.

2. RELATED WORKS

2.1 Primitives for Scene Representation

Geometric primitives have been widely used in point cloud scene understanding and scene reconstruction. Xia et al. [12] categorized these primitives into distinct groups; as shape (lines, surface, volumetric shapes) and structural primitives (skeletons, edges). Geometric shapes such as circles have been used for tree identification by fitting circular line segments to point cloud data sets containing tree trunks [13]. In another example were used for indoor scene reconstruction [14]; linear primitives were fitted to walls projected on a 2d plane to segment the indoor environment. In an urban setting, volumetric primitives were used to partition point cloud data and obtain a candidate of boxes to approximate building structures [15] and in another work, plane fitting was used to get polygonal meshes for buildings [16]. These primitives work well in structured environments but are difficult to adapt to off road environments with irregular data.

Structural primitives such as skeletons have been used to model tree structures [17], but the proposed method is specific for object detection in scenes and not generalizable to entire scene reconstruction and abstraction.

Scene reconstruction using LiDAR point clouds has also been studied as surface reconstruction problems. 3D octrees have been used to in conjunction with optimization on energy functions to directly obtain watertight meshes of a scene [18], however, such an approach is costly and well suited to high fidelity scene reconstruction. While simple geometric shapes have been well studies in scene reconstruction their application is limited to structured point clouds scenes and are not generalize to offroad environments.

2.2 Clustering Techniques for Point-Cloud Representations

Clustering techniques have often been used in segmenting point clouds, particularly for tasks like object detection and semantic classification. Previous research has predominantly concentrated on applying these methods within indoor or urban settings [19] or for analyzing structured objects [20]. For instance, Zhou et al. [21] devised a two-step approach for segmenting point clouds in large-scale scenes, initially identifying ground points and then employing a self-adaptive Euclidean clustering algorithm to cluster non-ground objects. Ioannou et al. [19] introduced a method utilizing the difference of normals to segment unorganized 3D point clouds from urban scenes like KITTI and TITAN, aiding in object detection and annotation. Nakagawa [22] utilized region-based clustering, leveraging normal vectors to extract groups sharing directional similarity for range image generation. Additionally, previous works have explored hierarchical clustering variations, incorporating bipartite graph matching to cluster and merge regions, particularly for delineating buildings [23]. Subspace clustering has also been employed for classifying point clouds in construction sites [24]. Nonetheless, these methodologies primarily target organized urban environments and structured data points, often incurring significant computational costs [23, 24]. There's been a lack of focus on utilizing clustering techniques for generating scene abstractions for off-road environments, which constitutes the primary focus of our work.

3. METHODOLOGY

In our work we focus on a multi-level of detail (MLOD) scene representation of LiDAR point clouds for autonomous off road driving using a combinatorial statistical and geometric approach. We realize this MLOD abstraction using a novel transformation tree structure such that all leaf nodes of the tree at a certain tree depth contains clusters of the points that represents our abstraction. Each tree node encodes a transformation based on Principal Component Analysis to account for the different orientations of objects within the scene. While other works focus on high fidelity of scene representation our approach focuses on abstract representation that has low a footprint and does not require prior segmentation of the point cloud data. In this section we describe the transformation tree structure and our methodology of abstraction. We also define the metrics for experimental evaluation of our approach.

Transformation Tree: Given a point cloud dataset $P = \{(x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_n, y_n, z_n)\}$ with n points in a global reference frame $\mathcal{G}$, we define the transformation tree $\mathcal{T}$ as a rooted connected acyclic directed graph. The root node of the tree contains the point cloud $\tilde{P}$ which is P expressed in its principal reference frame $\mathcal{B}_0$. In addition to the point set $\tilde{P}$, the root node also contains the transformation T_0 that maps $\mathcal{G}$ to $\mathcal{B}_0$. We then run a clustering algorithm on the point distribution in the root node and express each cluster in its own principal frame $\mathcal{B}_{1,i}$ where the first subscript represents the tree depth and the second subscript i represents the i_{th} child of the parent node. The transformed clusters $\tilde{P_{1,i}}$ and the associated transformations $T_{1,i}$ that map $\mathcal{G}$ to $\mathcal{B}_{1,i}$ are added as child nodes to the root node. Note the transformation stored in each node maps the global frame to the principal axis of the point cluster stored in that node. We hierarchical cluster point distributions in each node at a certain tree depth and add these clusters as child nodes to their parent node. This process is repeated until the desired tree depth d is reached. The final point clusters $\tilde{P_{d,i}}$ contained in the leaf nodes of the tree of depth d represent the abstract representation of the scene. Applying the inverse of the corresponding transformations $T_{d,i}$ on these clusters gives the points clusters in the global reference frame $\mathcal{G}$.

3.1 Method Overview

Given the point cloud dataset P in $\mathcal{G}$, we compute the abstract representation via the tree structure using the following steps:

- **Step 1**: LiDAR point clouds often exhibit multi modal behavior, meaning they contain distinct sub-populations or clusters of points representative of the different artefacts in the scene. Relying on the Central Limit Theorem it can be argued that these sub-populations are normally distributed and that they can be represented as Gaussian functions. Therefore we use a Gaussian Mixture Model to model the given

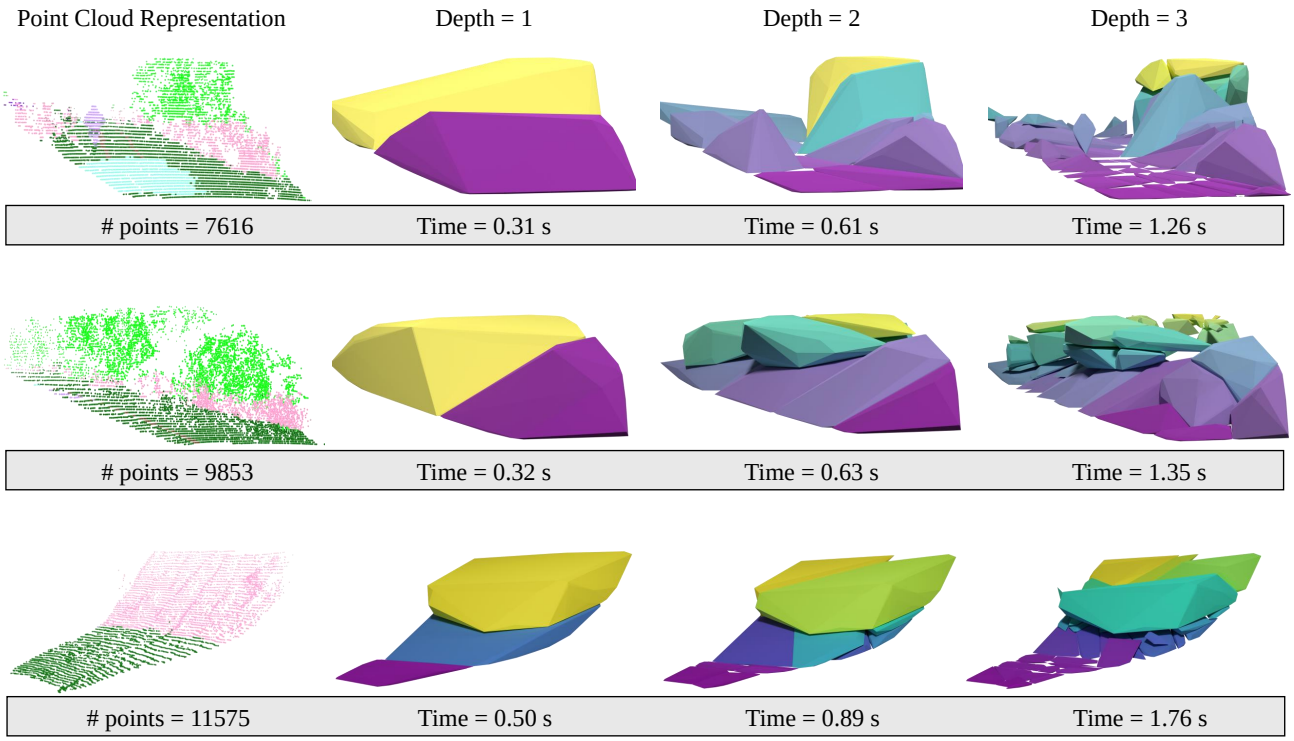


FIGURE 2: Results of Experiment 1 for multiple scenes at three different transformation tree depths show an increase in the quality of abstraction as tree depth increases. As tree depth increases objects such as humans and ground becomes more clearer, however, higher depths can lead to topological discontinuities as seen in the ground abstractions in scene 1 and 3.

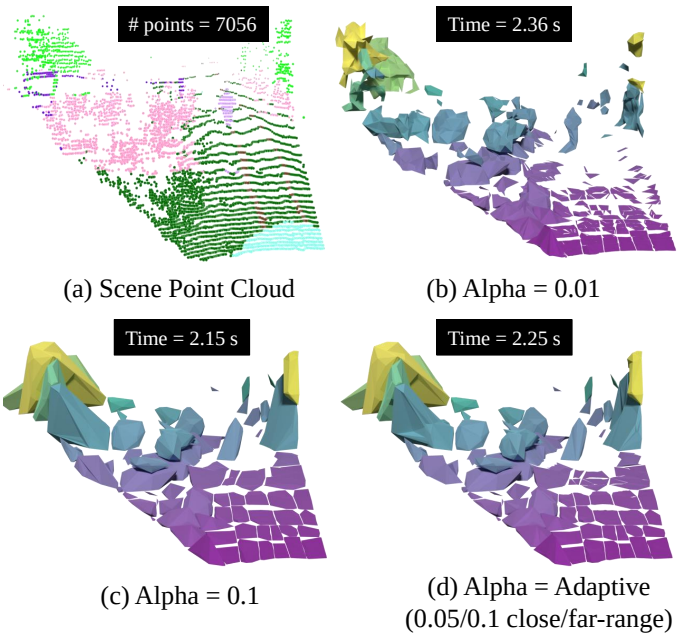
point cloud distribution. In the first step we determine the ideal number of Gaussian components for a Gaussian Mixture Model (GMM) used for approximating the point cloud distribution.

- **Step 2**: We construct our transformation tree using a hierarchical subspace clustering algorithm such that each node of the tree represents a sub-cluster of the point cloud with statistically similar features. Representing these point clusters using geometric primitives such as a convex hull for all the nodes at a tree depth gives an abstract representation for the entire scene.
- **Step 3**: We post process the scene representation by integrating the tree nodes using a planarity measure to improve topological consistency and contiguity. We then use geometric shapes such as convex hulls and alpha shape for the final scene representation.

The methods used in the aforementioned steps are discussed in more detail in the following sections.

3.1.1 Gaussian Component Determination for the Gaussian Mixture Model. Fitting a GMM to a point distribution involves learning the parameters such as means, covariances and mixing coefficients of the Gaussian components. For simplicity we model the data in the three dimensions independently and fit uni-variate Gaussian's to the data in each of these dimensions. To determine the number of Gaussian components to fit to the data we began by computing the data histogram for each dimension. We chose the number of bins for the histogram as the maximum of the sturges and Freedman Diaconis Estimator. Next, we fit Gaussian Mixture models to the data in the three dimensions separately with Gaussian components varying from 1 to 5, if the corresponding bin size of the data histogram is less than 5 we use the bin size as the maximum number of Gaussian components in the GMM. The Gaussian components are capped at 5 to limit computational expense. The GMM is learnt using the expectation-maximization (EM) algorithm [25] and for time efficiency we initialize the weights of the GMM using the k-means++ algorithm [26]. Once we have learnt the GMMs with these varying Gaussian components for each dimensions, we use KL divergence as a metric to measure the relative entropy of the fitted distributions and the corresponding data histogram. We then use the Kneedle algorithm [27] to find the elbow point i.e. is the number of ideal Gaussian components.The corresponding GMM is then selected as the best probabilistic approximation of the given data.

3.1.2 Hierarchical Subspace Clustering. For a LiDAR point distribution P, we first apply the technique of principal component analysis (PCA) to find the principal axis $\mathcal{B}_0$ of the data. We then add the transformation of the principal frame $\mathcal{B}_0$ from the global frame $\mathcal{G}$ along with the transformed point distribution ($\tilde{P}$) to the root node of our transformation tree. Next, we analyze this data distribution in the root node along each of its principal axis and determine the ideal number of Gaussian components for a GMM that best approximates it using the procedure described in step 1. Using the fitted probabilistic function from step 1 we find local minima of the distribution along each axis and



(a) Scene Point Cloud (b) Alpha = 0.01

(c) Alpha = 0.1 (d) Alpha = Adaptive (0.05/0.1 close/far-range)

FIGURE 3: Effect of alpha values on the alpha-hull representation of scenes at transformation tree depth = 3. Lower α values (0.01) provide accurate representations of objects such as the human and trees, but are computationally expensive and cause increased topological discontinuities (b). Higher α values (0.1) are quicker to compute but may not capture geometric intricacies (c). An adaptive approach based on varying α values depending on distance from vehicle can provide balanced results (d).

use these minima along with the axis bounds of the point clusters to create a 3D mesh grid. We discard all of the empty boxes within a grid and for each non empty cell, we repeat the above process i.e. we again do the PCA on the points within the cell and add the corresponding transformation and the transformed point subsets as child nodes to the current node and fit a GMM to it as described in step 1. We hierarchically apply this method for a specified number of iterations to all the tree nodes at a given depth, the leaf nodes of the final tree represent the abstracted component clusters defined in their principal component frames. Using the inverse of the encoded transformation in the leaf nodes we transform and get the point clusters in the global reference frame $\mathcal{G}$.

3.1.3 Integration of Planar Features and Scene Represenation using Geometric Shapes. Given the point clusters in the global reference frame, geometric shapes such as convex hulls can be fitted to abstract the scene. However, at higher levels of tree depths, the point clusters tend to diminish in size and exhibit a quasi-planar structure associated with the sparse LiDAR point clouds. This characteristic may result in topologically disconnected regions within the scene abstractions, which is particularly undesirable for semantic classes such as terrains. Moreover, convex hulls may fail to accurately capture the geometric intricacies associated with terrains and foliage. To address these challenges, we propose two post-processing steps: (a) integration of planar point clusters to 'patch' topologically disconnected regions, and (b) utilization of alpha/concave hulls for improved geometric representations [28]

Integration of Planar Point Clusters: Our initial step towards integration involves the identification of planar point clusters. To accomplish this, we employ (PCA) on each point cluster derived from the hierarchical subspace abstraction procedure. The determination of planarity is based on the analysis of the smallest eigenvalue associated with the covariance matrix, compared against a predefined threshold (e.g., 0.1 in our experiments). A point cluster is deemed planar if its smallest eigenvalue falls below this threshold, and the corresponding eigenvector is designated as its normal vector. This signifies that the variance of point distribution is minimal along this axis, indicating planarity [29, 30]. Subsequently, we identify the k-nearest neighbors (kNN) for each planar cluster. If any of these neighbors are also planar, we assess the similarity in the direction of their normal vectors using cosine similarity. A cosine similarity exceeding a specified threshold (e.g., >0.9) indicates alignment in direction. In such cases, the neighboring clusters are integrated. Our approach is grounded in the assumption that planar clusters with similar normal vectors likely belong to the same semantic class (e.g., terrain). In our experimental setup, we set k to 2, as larger values may lead to the merging of distant clusters, resulting in elongated and distorted hulls that may not accurately represent the scene.

Alpha Hull Representations: Our second post-processing step involves the fitting of alpha hulls to the point clusters, aiming for improved geometric representations compared to simple convex hulls. To compute these alpha hulls, we adopt an adaptive strategy. Employing a small α value can yield a detailed representation of the scene, but it comes at a higher computational cost and may result in the presence of holes, particularly in sparse regions, such as those distant from the LiDAR sensor/vehicle where point density is low (Figure. 3 (b)). Conversely, larger α values do not offer significant benefits over convex hulls, although they can be computationally efficient (Figure. 3 (c)). Our adaptive approach initially identifies point clusters whose centroids lie within a threshold radius of the LiDAR sensor/vehicle, utilizing a smaller α value to represent these clusters. For clusters lying beyond this radius, alpha hulls with larger α values are employed. This methodology helps us strike a balance between computational efficiency and the fidelity of representations (Figure. 3 (d)).

3.2 Evaluation Metrics

To evaluate our proposed abstraction scheme we consider the following three metrics:

- **Visual Qualitative Assessment:** The abstraction aims to find a digital representation of the given point cloud scene. We visually assess the quality of abstraction in terms of the features that are abstracted; we compare abstractions of planar structures such as the ground or humans in the scene with features with variable densities such as the bush and trees. We also visually assess if the abstracted feature correctly depicts a derivable terrain segment or not.
- **Computational Time:** We quantify the time taken by each key module in the algorithm and also compare the time taken

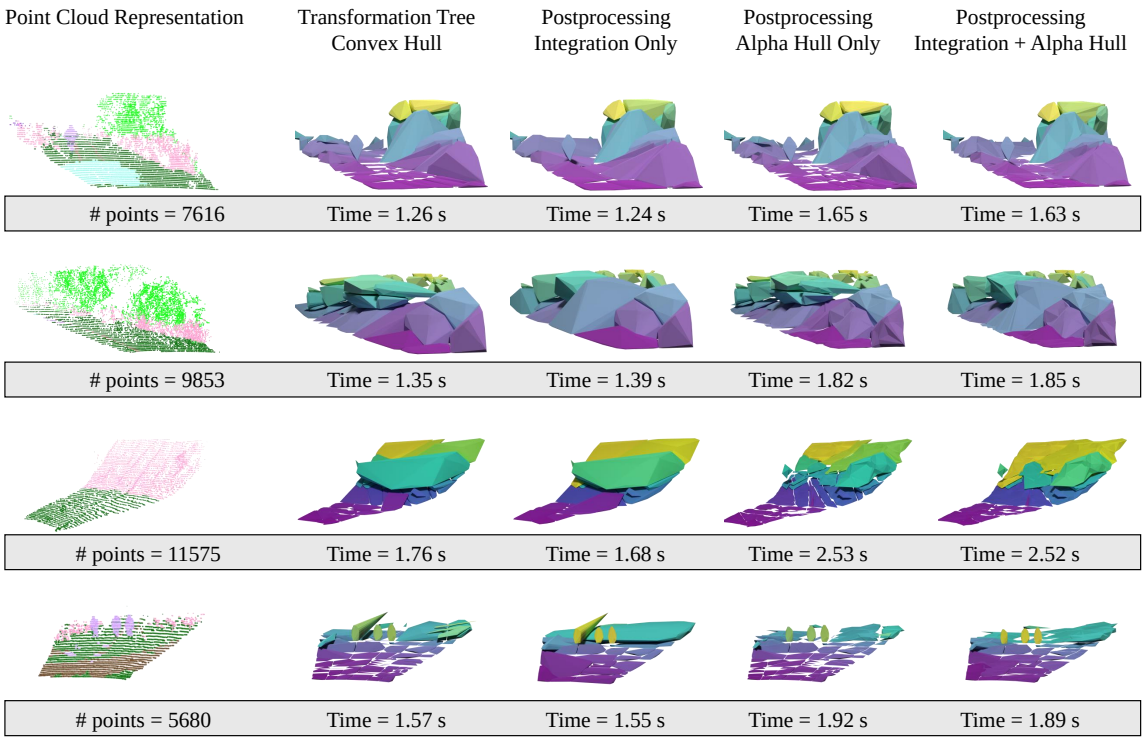


FIGURE 4: Comparison between results obtained from experiments 1 (2nd column), 2 (3rd column), 3 (4th column) and 4 (5th column) for transformation tree depth of 3. The integration step (3rd column) helps reduce topological inconsistencies between point clusters. The alpha hulls represent the geometric features of the scene better while removing unwanted artifacts (third row and fourth row). Combination of post-processing steps, while relatively more computationally expensive, results in overall better visual representations.

for the abstractions at different levels of detail.

- **Abstraction Accuracy:** We use the segmentation labels in the RELLIS-3D dataset to assess the accuracy of the abstracted components. A perfect abstraction of the scene will have points with only one label in an abstracted component. We use this as ground truth and for each abstracted component we determine the label with the most number of points in it. This label is considered the true label for the abstracted component and the corresponding points are considered to be correctly classified Points with a different label in the abstracted component are considered as a mislabeled points. We sum up the correctly labeled and incorrectly labeled points for the entire scene and report them as a percentage of the total number of points in the scene.

4. EXPERIMENTS & FINDINGS

We conducted four experiments to evaluate the performance of our abstraction methodology. We analyzed the qualitative and computational performance of the algorithms at multilevel of detail in isolation and in conjunction with the post-processing steps. Our algorithms were written in Python and the experiments were conducted on a Windows Laptop equipped with an AMD Ryzen 9 5900HS CPU, 16GB RAM and an NVIDIA 3070RTX GPU. The dataset for our experiments consisted of 100 randomly sampled scenes from the RELLIS-3D dataset [11] for which we extracted the point clouds within the field of view of the vehicle. The four experiments and their corresponding findings are explained in the following sections.

4.1 Experiment 1: Multi Level of Detail Abstraction: Varying Tree depth

For our first experiment, we wanted to observe the effects of varying transformation tree depths on the level of detail extracted in the scene abstractions. The experiment was carried out in the absence of post-processing steps, as it helped us understand the results of the hierarchical subspace clustering technique. The experiment was carried out for tree depths = 1, 2, and 3.

Findings: The qualitative results of our first experiment show an increase in abstraction quality with increasing tree depths (Figure. 2). At level 1 we have obscure details about the scene and as the depth grows we see an abstraction that encodes the finer features in the scenes such as the human in scene 1 and the ground in scenes 1, 2 and 3. This is expected because the number of sub-clusters of points increases as the tree depth increases, leading to more features being abstracted within the scene. However, we observe that planar features, such as the ground, display disconnected hulls as observed at depth 3 for scenes 1 and 3. The presence of the disconnected regions may suggest that increasing depths of the tree may lead to topological inconsistencies within certain semantic classes of the scene.

We also observe that for regions that represent trees and bushes our method gives elongated and skewed hulls as seen in the three scenes in Figure 2. The elongated hull can also be observed in the fourth scene in column 2 of Figure 4. This phenomena can be explained by outlier points within specific

planes of the sub-clusters that are captured by the LiDAR but are far from the regular distribution. This can be expected in off-road environments where some LiDAR rays may be blocked by vegetation while others may penetrate the foliage in the vegetation and pick up points from distant objects. Considering convex hulls are sensitive to outliers, the presence of these distant points may result in the elongated and skewed hulls.

In regards to the time analysis, we observe that increasing the tree depths results in increasing computational times. The increased times can be associated to computing increased number of subclusters at higher tree depths. We get split second performance for abstraction at tree depth 1 and 2 and observe a computational time of more than a second for the three scenes at tree depth 3.

4.2 Experiment 2: Integration of Planar and Aligned Point Clusters

In our second experiment, we exclusively tested the efficacy of our first post-processing step, i.e., integration of neighboring planar and aligned point clusters, where the final set of point clusters were abstracted through convex hulls. We conducted the experiment for a transformation tree depth = 3, as we noticed the topological discontinuities starting at that tree depth. The quality and computational expense of the integration techniques were analyzed.

Findings: The results in column two and three of Figure 4 show the abstractions of scene with and without the integration post processing step. We observe that the integration step mitigates the discontinuities in the abstracted features specifically the ground terrain to a certain degree. We observe merged convex hulls for the ground features in column two in contrast to the abstractions in column 1. However, the method does not completely resolve the issues of topological discontinuities as we observe holes present in the results. While integration step does well in terms of connecting planar components next to each other, it can inadvertently lead to the integration of different features that exhibit the same planarity characteristics. We observe in scene 1 that after integration we get a single abstraction for the ground and the puddle. Similarly in scene 4 we observe an abstraction that encodes the ground with the mud trails.

From the experiments it is also observed that the computational time improves in some instances when we use integration as a post processing step. This marginal increase can be attributed to the fact that integration reduces the number of abstracted components that are rendered with convex hulls.

4.3 Experiment 3: Effect of rendering with Alpha shape

For our third experiment, we exclusively tested the efficacy of utilizing alpha hulls to represent the point clusters obtained from the hierarchical subspace clustering method, and as a result did not employ the integration step. We conducted the experiment with different values of alpha = 0.01 and 0.1. Additionally, we tested our adaptive methodology where close-range clusters were represented with alpha values = 0.05, and far range clusters were represented with alpha values = 0.1.

Findings: The results show that using a smaller alpha hull value gives a detailed abstraction for the scene but it also leads to discontinuities in the abstraction in regions where the point density is sparse. It also mitigates elongated and skewed hulls as observed in experiment 1. Column 4 in Figure 4 shows how alpha hulls cater to the skewed hulls in scene 2, 3 and 4. The geometric representation breaks up the elongated hull to give a better abstraction of the scene. In contrast a large value does not give a detailed geometric representation and may not be effective in containing the skewed and elongated hulls, however they do well in regions where point cloud density is low. We observe a balance in the advantages that are offered by the small and large alpha hull values when using an adaptive approach in which we change alpha hull value based on the distance from the vehicle. For autonomous driving higher fidelity of abstraction of the immediate surroundings is preferred and hence for points close to the lidar we use a smaller alpha hull value. We observe a detailed representation of the scenes in column 5 of Figure 4 in regions that are close to the vehicle and for point clusters that are farther and sparse in nature we see lesser number of discontinuities with a large alpha hull value.

For a computational perspective using a smaller alpha hull value is computationally expensive than using a large alpha hull value. When using the adaptive approach for alpha hull rendering we strike a balance in computational expense and quality of the rendered scene. In general the post processing step of alpha hulls is computationally expensive and takes more.

4.4 Experiment 4: Hybrid Approach

For our fourth experiment, we tested the algorithm in its entirety, with all of the the modules being employed. We reported the final scene abstractions at depth 3 with the post processing step of integration and alpha hull module. We tested and compared the total computational time vs. the time taken by individual modules. Additionally, we reported the accuracy of the abstracted scenes across the 100-scene dataset.

Findings: We observe that the quality of the abstractions improve when using integration with adaptive alpha hull values. Integration of abstracted components prior to alpha hull rendering improve continuity of abstraction as they mitigate discontinuities in the abstracted scenes, however they do exhibit a tendency of merging regions with different semantics but having the same planarity. Integration also reduces the number of abstracted components which reduces the time computation for the alpha hull module and leads to a reduction in total time as compared to when working with alpha hulls without any integration. Alpha hulls also remove elongated and skewed hulls observed for scenes in column 1 and 2 of Figure 4 and using an adaptive value for alpha hull caters to the sparse point densities in regions far away from the LiDAR.

A detailed time computational analysis of the approach is provided in Figure 5. The box plot shows that time increase as tree depth increases which is evident as tree growth is done hierarchically and each hierarchical iteration adds to the computation time. We observe that it takes approximately 2 seconds to generate an abstraction with depth 3 of the transformation tree. Also we observe that of the total time that it takes to abstract a scene the highest computational expense is made in the tree construction step. We performed a systematic analysis and noted

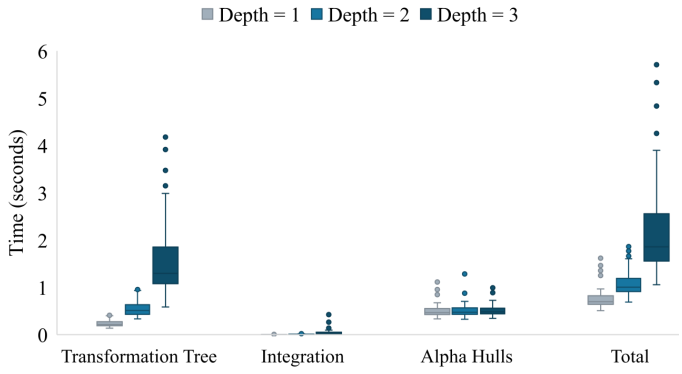


FIGURE 5: Computational times for our MLOD abstraction and post processing steps for 100 RELLIS-3D Scenes. The results show computational time increases as tree depth increases. In addition, it can also be observed that the transformation tree module contributes the most to the total time taken for scene abstraction. The post processing modules in comparison are not computationally expensive.

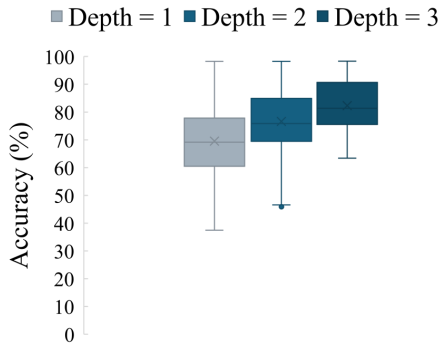


FIGURE 6: Accuracy assessment of abstraction on 100 RELLIS-3D scenes. The number of correctly classified points increase as tree depth increases, we observe that more then 80% points are correctly classified at tree depth 3 as opposed to 70% at tree depth 1.

that for the total time it takes to abstract 100 scenes, 95.6% of the time is expended in the determination of components for Gaussian mixture models using the KL divergence, hence we identify step 1 of our methodology as the bottleneck operation in our abstraction scheme. This can be mitigated by using parallel computing in future works. Of the two post processing steps alpha hull is computationally more expensive than integration of planar features.

The accuracy assessment for the abstraction at different tree depths is shown in Figure 6. We observe that the accuracy increases as tree depth increases; we see an increase from approximately 70% to 85% correctly classified points in the abstraction as we vary tree depth from 1 to 3.

5. DISCUSSION

5.1 Limitations

In our experiments with higher tree depths (Depth > 2), we observed a significant increase in computation time for creating the transformation tree and fitting the Gaussian Mixture Model (GMM) compared to lower tree depths (Figure. 5). This rise in computation time can be attributed to the challenge of identifying optimal Gaussian components for a growing number of point clusters present at higher tree depths. Presently, our algorithm operates sequentially on all nodes of the transformation tree, addressing one leaf node at a time. This approach was chosen initially to assess the effectiveness of the method in providing satisfactory scene abstractions. However, for future work, we can utilize parallel computing techniques to execute our steps concurrently on each leaf node, thereby improving computational efficiency. Utilizing GPU-based libraries can further improve computation times, as our transformation tree data structure lends itself well to parallel computing, with each leaf node amenable to independent processing.

Another limitation in our study is the occurrence of topological incontiguities in classes such as terrains, which are evident at higher tree depths due to the characteristics of the lidar point cloud distributions (Figures. 3, 4). While our post-processing integration step helps in 'patching' similar planar aligned point clusters, it does not entirely resolve the issue of incontiguous regions (Figure. 4 Third Column). One potential strategy to address this challenge is to recursively apply the post-processing integration step until all similar planar point clusters are accurately integrated into a single cluster. This iterative approach may help in minimizing the observed 'holes' in our scene abstractions.

5.2 Adaptive Approaches Towards Scene Abstractions

Our current method employs a hierarchical subspace abstraction technique over a fixed number of iterations, wherein all tree nodes are extracted to the same depth. While this approach ensures appropriate representation and division of each point cluster, an adaptive strategy may enhance computational efficiency and better representations. Specifically, this involves dynamically growing the tree only for nodes requiring further division, while preserving nodes that satisfy specific termination criteria. Several techniques can be used to define a termination criteria. For example, a version of our post-processing integration step can assess whether the leaf nodes of a parent node are generating similarly aligned, planar clusters. If so, further division of the parent node is halted, terminating the hierarchical process for that node. Additionally, integration based on other features such as similarity across PCA components and proximity of leaf nodes can be measured for integration, defining a termination step for the parent node. Volume and centroid movements of clusters offer another approach to terminating certain nodes. For instance, if a parent node is being split into clusters with significantly smaller volumes compared to others in the scene, it may be terminated, as sparse LiDAR point clouds often lead to such small volume clusters, potentially diminishing scene representation. Similarly, evaluating centroid movement can aid termination decisions. If leaf node centroids exhibit minimal movement compared to the parent node centroid (e.g. in the case of foliage clusters), the parent node can be terminated.

The termination of certain nodes, especially at higher tree depths, may significantly improve computational efficiency by reducing the application of the Gaussian Mixture Model (GMM) fitting process to fewer leaf nodes at each hierarchical itera-

tion. Furthermore, adaptive approaches can be extended to post-processing steps as well. For instance, our current implementation of the integration step may merge planar and aligned distributions of humans standing close to each other into a single cluster, resulting in a unified convex/alpha hull instead of distinct ones. To mitigate this, evaluating clusters based on the directions of their normal vectors can be employed, integrating clusters facing upwards/downwards (e.g., those representing terrain classes) while preserving those facing towards/away from sensors (e.g., upright objects such as humans, fences, poles). These adaptive approaches may help improve computational efficiency as well as address issues such as topological inconsistencies and discontiguities.

6. CONCLUSION

In this work, we introduced a novel approach for hierarchical representation of LiDAR point cloud data tailored towards off-road environments. We took a geometric-statistical approach towards generating 3D scene abstractions by developing a transformation tree data structure, that encoded clusters of point clouds in a tree hierarchy. The depth of the trees corresponded to the level of details abstracted from the scene. In order to obtain the clusters, we leveraged a generalizable method for fitting GMMs to identify and segment point distributions. The abstracted point clusters were subsequently subjected to post-processing techniques such as integration of planar aligned clusters, and fitting geometric shapes such as convex and alpha hulls, to further refine the abstraction process. We evaluated our approach on the RELLIS-3D dataset, by conducting four experiments to study the effects of varying tree depths, integration of planar aligned clusters, effect of alpha hulls and the final results of integrating each step. Through the experiments, we showed that our approach generated low footprint abstraction at multilevel of details and was able to successfully represent various scene features like humans, ground and vegetation with reasonable detail.

ACKNOWLEDGMENTS

This work was supported by the Distributed Autonomous Robotic Experiments and Simulations (DARES) research project funded by the Army Research Laboratory (ARL). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the ARL.

REFERENCES

[1] Maturana, Daniel, Chou, Po-Wei, Uenoyama, Masashi and Scherer, Sebastian. "Real-time Semantic Mapping for Autonomous Off-Road Navigation." *Proceedings of 11th International Conference on Field and Service Robotics (FSR '17)*: pp. 335 – 350. 2017.

[2] Piewak, Florian, Pinggera, Peter, Schäfer, Manuel, Peter, David, Schwarz, Beate, Schneider, Nick, Pfeiffer, David, Enzweiler, Markus and Zöllner, Marius. "Boosting LiDAR-based Semantic Labeling by Cross-Modal Training Data Generation." (2018). URL 1804.09915.

[3] Chen, Zhe, Zhang, Jing and Tao, Dacheng. "Progressive LiDAR adaptation for road detection." *IEEE/CAA Journal of Automatica Sinica* Vol. 6 No. 3 (2019): p. 693–702. DOI 10.1109/jas.2019.1911459. URL http://dx.doi.org/10.1109/JAS.2019.1911459.

[4] Kühner, Tilman and Kümmerle, Julius. "Large-Scale Volumetric Scene Reconstruction using LiDAR." *2020 IEEE International Conference on Robotics and Automation (ICRA)*: pp. 6261–6267. 2020. DOI 10.1109/ICRA40945.2020.9197388.

[5] Ouyang, Zhenchao, Liu, Yu, Zhang, Changjie and Niu, Jianwei. "A cGANs-Based Scene Reconstruction Model Using Lidar Point Cloud." *2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC)*: pp. 1107–1114. 2017. DOI 10.1109/ISPA/IUCC.2017.00167.

[6] Alaba, Simegnew Yihunie and Ball, John E. "A Survey on Deep-Learning-Based LiDAR 3D Object Detection for Autonomous Driving." *Sensors* Vol. 22 No. 24 (2022). DOI 10.3390/s22249577. URL https://www.mdpi.com/1424-8220/22/24/9577.

[7] Liu, Shaoshan, Yu, Bo, Tang, Jie, Zhu, Yuhao and Liu, Xue. "Communication Challenges in Infrastructure-Vehicle Cooperative Autonomous Driving: A Field Deployment Perspective." *IEEE Wireless Communications* Vol. 29 No. 4 (2022): pp. 126–131. DOI 10.1109/MWC.005.2100539.

[8] Deschaud, Jean-Emmanuel, Prasser, David, Dias, M. Freddie, Browning, Brett and Rander, Peter. "Automatic data driven vegetation modeling for lidar simulation." *2012 IEEE International Conference on Robotics and Automation*: pp. 5030–5036. 2012. DOI 10.1109/ICRA.2012.6225269.

[9] Fang, Jin, Zhou, Dingfu, Yan, Feilong, Zhao, Tongtong, Zhang, Feihu, Ma, Yu, Wang, Liang and Yang, Ruigang. "Augmented LiDAR Simulator for Autonomous Driving." *IEEE Robotics and Automation Letters* Vol. 5 No. 2 (2020): p. 1931–1938. DOI 10.1109/lra.2020.2969927. URL http://dx.doi.org/10.1109/LRA.2020.2969927.

[10] Marchand, Y., Vallet, B. and Caraffa, L. "EVALUATING SURFACE MESH RECONSTRUCTION OF OPEN SCENES." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* Vol. XLIII-B2-2021 (2021): pp. 369–376. DOI 10.5194/isprs-archives-XLIII-B2-2021-369-2021. URL https://isprs-archives.copernicus.org/articles/XLIII-B2-2021/369/2021/.

[11] Jiang, Peng, Osteen, Philip, Wigness, Maggie and Saripalli, Srikanth. "RELLIS-3D Dataset: Data, Benchmarks and Analysis." (2022). URL 2011.12954.

[12] Xia, Shaobo, Chen, Dong, Wang, Ruisheng, Li, Jonathan and Zhang, Xinchang. "Geometric Primitives in LiDAR Point Clouds: A Review." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* Vol. 13 (2020): pp. 685–707. DOI 10.1109/JSTARS.2020.2969119.

[13] H.-G. Maas, S. Scheller, A. Bienert and Keane, E. "Automatic forest inventory parameter determination from terrestrial laser scanner data." *International Journal of Remote Sensing* Vol. 29 No. 5 (2008): pp. 1579–1593. DOI

10.1080/01431160701736406. URL https://doi.org/10.1080/01431160701736406, URL https://doi.org/10.1080/01431160701736406.

[14] Cui, Yang, Li, Qingquan, Yang, Bisheng, Xiao, Wen, Chen, Chi and Dong, Zhen. "Automatic 3-D Reconstruction of Indoor Environment With Mobile Laser Scanning Point Clouds." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* Vol. 12 No. 8 (2019): pp. 3117–3130. DOI 10.1109/JSTARS.2019.2918937.

[15] Li, Minglei, Wonka, Peter and Nan, Liangliang. "Manhattan-world Urban Reconstruction from Point Clouds." *ECCV*. 2016.

[16] Nan, Liangliang and Wonka, Peter. "PolyFit: Polygonal Surface Reconstruction from Point Clouds." *2017 IEEE International Conference on Computer Vision (ICCV)*: pp. 2372–2380. 2017. DOI 10.1109/ICCV.2017.258.

[17] Mei, Jie, Zhang, Liqiang, Wu, Shihao, Wang, Zhen and Zhang, Liang. "3D tree modeling from incomplete point clouds via optimization and L1-MST." *International Journal of Geographical Information Science* Vol. 31 (2017): pp. 1021 – 999. URL https://api.semanticscholar.org/CorpusID:205794840.

[18] Caraffa, L., Brédif, M. and Vallet, B. "3D OCTREE BASED WATERTIGHT MESH GENERATION FROM UBIQUITOUS DATA." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* Vol. XL-3/W3 (2015): pp. 613–617. DOI 10.5194/isprsarchives-XL-3-W3-613-2015. URL https://isprs-archives.copernicus.org/articles/XL-3-W3/613/2015/.

[19] Ioannou, Yani, Taati, Babak, Harrap, Robin and Greenspan, Michael. "Difference of normals as a multi-scale operator in unorganized point clouds." *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission*: pp. 501–508. 2012. IEEE.

[20] Mahdaoui, Abdelaaziz and Sbai, El Hassan. "3D point cloud simplification based on k-nearest neighbor and clustering." *Advances in Multimedia* Vol. 2020 (2020): pp. 1–10.

[21] Zhou, Ying, Wang, Dan, Xie, Xiang, Ren, Yiyi, Li, Guolin, Deng, Yangdong and Wang, Zhihua. "A fast and accurate segmentation method for ordered LiDAR point cloud of large-scale scenes." *IEEE Geoscience and Remote Sensing Letters* Vol. 11 No. 11 (2014): pp. 1981–1985.

[22] Nakagawa, Masafumi. "Point cloud clustering using panoramic layered range image." *Recent applications in data clustering* (2018): pp. 75–92.

[23] Xu, Sheng, Wang, Ruisheng, Wang, Hao and Zheng, Han. "An optimal hierarchical clustering approach to mobile LiDAR point clouds." *IEEE Transactions on Intelligent Transportation Systems* Vol. 21 No. 7 (2019): pp. 2765–2776.

[24] Xu, Yusheng, Tuttas, Sebastian, Heogner, Ludwig and Stilla, Uwe. "Classification of photogrammetric point clouds of scaffolds for construction site monitoring using subspace clustering and PCA." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* Vol. 41 (2016): pp. 725–732.

[25] Moon, T.K. "The expectation-maximization algorithm." *IEEE Signal Processing Magazine* Vol. 13 No. 6 (1996): pp. 47–60. DOI 10.1109/79.543975.

[26] Arthur, David and Vassilvitskii, Sergei. "k-means++: the advantages of careful seeding." *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*: p. 1027–1035. 2007. Society for Industrial and Applied Mathematics, USA.

[27] Satopaa, Ville, Albrecht, Jeannie, Irwin, David and Raghavan, Barath. "Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior." *2011 31st International Conference on Distributed Computing Systems Workshops*: pp. 166–171. 2011. DOI 10.1109/ICDCSW.2011.20.

[28] Edelsbrunner, Herbert and Mücke, Ernst P. "Three-dimensional alpha shapes." *ACM Transactions On Graphics (TOG)* Vol. 13 No. 1 (1994): pp. 43–72.

[29] El-Sayed, Emad, Abdel-Kader, Rehab F, Nashaat, Heba and Marei, Mahmoud. "Plane detection in 3D point cloud using octree-balanced density down-sampling and iterative adaptive plane extraction." *IET Image Processing* Vol. 12 No. 9 (2018): pp. 1595–1605.

[30] Yeon, Suyong, Jun, ChangHyun, Choi, Hyunga, Kang, Jaehyeon, Yun, Youngmok and Lett Doh, Nakju. "Robust-PCA-based hierarchical plane extraction for application to geometric 3D indoor mapping." *Industrial Robot: An International Journal* Vol. 41 No. 2 (2014): pp. 203–212.