INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

Sculpting Efficiency: Hierarchical Codec Framework for 3D LiDAR Point Cloud Data

P L Chithra[1]*, S Lakshmi Bala[2]

1 Professor, Department of Computer Science, University of Madras, Chennai, Tamil Nadu, India
2 Research Scholar, Department of Computer Science, University of Madras, Chennai, Tamil Nadu, India

OPEN ACCESS

Received: 29-08-2024
Accepted: 15-03-2025
Published: 10-04-2025

Citation: Chithra PL, Bala SL (2025) Sculpting Efficiency: Hierarchical Codec Framework for 3D LiDAR Point Cloud Data. Indian Journal of Science and Technology 18(11): 904-921. https://doi.org/10.17485/IJST/v18i11.1583

* **Corresponding author**.

chitrasp2001@yahoo.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: A novel and effective 3D compression strategy is proposed for Point Cloud Data (PCD) from Light Detection and Ranging (LiDAR) sensors. The proposed methodology called 3DPCD-TPK Codec: t-distributed Stochastic Neighbour Embedding (t-SNE), Density-Based Spatial Clustering of Applications with Noise (DBSCAN), and K-dimensional Tree (K-D Tree) for 3D PCD Compression. 3DPCD-TPK codec achieves effective compression of 3D PCD by combining t-SNE for dimensionality reduction, DBSCAN for clustering, and K-D Tree for effective spatial indexing. To accomplish lossless compression of PCD, LZ4 compression is then applied. **Methods:** The proposed 3DPCD-TPK codec effectively compresses 3D LiDAR data while maintaining local links and structures. High-dimensional PCD is decreased by t-SNE while maintaining crucial linkages. K-D Trees maximize nearest-neighbour searches, while DBSCAN groups related points together while eliminating noise. Lastly, LZ4 reduces storage and transmission expenses without sacrificing important information by compressing the clustered data. The method efficacy in 3D PCD compression was demonstrated through evaluation on the Sydney Urban and Waupaca County Land Information Datasets. **Findings:** The proposed 3DPCD-TDK has a sequence of procedures comprising the compression methodology: loading PCD, dimensionality reduction using t-SNE, clustering with DBSCAN, K-D Tree construction for nearest neighbour search, and compression using LZ4. There were 4,367,881 points in the original (0843.las with 116.64 MB) LAS data, each with three dimensions. After 300 iterations of t-SNE analysis, the KL divergence decreased from 81.33 to 2.85 when 91 nearest neighbours were employed. 3.4 was the ultimate cost function. After t-SNE, the embedded data had 1,01,680 points with 95.64 Mega bytes (MB) LAS file was first created. The LAS file had a 7 cluster, from which 4590 points were eliminated due to noise. Following processing, the LAS file shrank to 3.41 MB by using t-SNE. The number of bytes grew to 2.31 MB after building K-D Trees. The data was then condensed by 1.16 MB from the Proposed 3DPCD-TDK codec, yielding a compression of 100.55 times than original with compression percentage

of 99.96%. The PCD file is compressed to just 1.16 MB using the 3DPCD-TPK technique, which is far less than the LAS files compressed with WinRAR (21.68 MB), LZ4 (73.69 MB), and 7-Zip (32.59 MB). With a minimum Mean Squared Error (MSE) of 0.004, the compressed and rebuilt point cloud data show no discernible changes from the original Peak Signal-to-Noise Ratio (PSNR) hits infinity, and the spatial distribution of points is perfectly conserved since the Earth Mover's Distance (EMD) is 0.0, which greatly lowers storage requirements without sacrificing excellent quality. This illustrates efficiency of proposed 3DPCD-TPK at reducing file sizes. **Novelty:** The complete strategy to compressing 3D PCD while maintaining crucial spatial linkages and facilitating effective data retrieval is what makes this compression technology distinctive. The 3DPCD-TDK codec provides an all-encompassing solution that is tailored to the unique characteristics of 3D PCD by combining methods like t-SNE for dimensionality reduction, DBSCAN for clustering, K-D Trees for nearest neighbour search, and LZ4 compression in Lossless technique.

Keywords: DBSCAN; LZ4 Algorithm; LiDAR; t-SNE; K-D Trees

1 Introduction

An exponential increase in data has resulted from the growing usage of LiDAR and other 3D scanning technologies in applications including industrial automation, urban mapping, and autonomous driving. Large 3D point cloud data files are frequently hundreds of megabytes in size. The rapid growth of LiDAR 3D PCD requires advanced techniques for effective processing and analysis. To address issues in dimensionality reduction, segmentation, compression, and spatial querying, the proposed work provides the techniques integrating local affinity capture, Gaussian t-SNE, DBSCAN clustering, LZ4 algorithm, and K-D tree hierarchical representation called 3DPCD-TDK codec.

Effective compression algorithms are required due to the volume and richness of data provided by LiDAR sensors. Due to the significantly greater effective scanning depth of light detection and ranging (LiDAR), PCD distribution is unequal and the sector becomes sparse in deep space. As a result, the distribution of nearby points surrounding the main points of interest has few peculiarities [(1)]. LZ4 is very fast, yet it has a subpar compression ratio for complex, massive 3D PCD. Larger compressed sizes result from existing approaches such as LZ4 in [(2)], which are not tailored to suit the specific structure of PCD. This is addressed by the proposed 3DPCD-TPK technique, which optimises both compression ratio and speed by integrating t-SNE, DBSCAN, and KD-tree. Although the Statistical Subspace Outlier Detection and Logarithmic Transformation employed in the LPCT technique [(3)] efficiently reduce spatial coefficient ranges and conceal unreliable places, it might not completely exploit the spatial and geometric properties of 3D point cloud data. By combining cutting-edge methods like t-SNE for dimensionality reduction, DBSCAN for clustering, and KD-tree structures to improve compression efficiency and spatial coherence, the proposed 3DPCD-TPK approach overcomes these drawbacks. The complex relationship between PCD geometry and colour variables have, which can result in less ideal compression results that either lose geometric accuracy or visual quality [(4)]. This gap is filled by the proposed 3DPCD-TPK approach, which improves colour and spatial management. It computes a dimensionless quality deterioration metric that enables accurate quantisation parameter tweaking for low distortion and a lower bit rate. Existing approaches [(5)] for Voronoi tessellation create 3D tessellations by uniformly distributing input particles through K-D tree

decomposition; however, they are not as computationally efficient and scalable when dealing with big and complicated 3D datasets.Advanced spatial partitioning techniques such as DBSCAN for clustering and KD-tree for effective indexing are used in the proposed 3DPCD-TPK method to overcome these constraints. In order to characterise a certain percentage of neighbours who have the same class as the central point, it fits the bandwidths of the Gaussian neighbourhoods of the labelled data using the class information [6]. This gap is filled by the proposed 3DPCD-TPK approach, which uses adaptive neighbourhood scaling, dynamic clustering with DBSCAN, and effective spatial indexing with K-D tree. The clustering algorithm DBSCAN clusters adjacent points together treating them as dense regions divided into less areas with high density. A popular density-metrics-based clustering technique called DBSCAN is effective at locating clusters with homogeneous densities [7]. By incorporating adaptive density thresholds and fusing DBSCAN with K-D tree indexing, the suggested 3DPCD-TPK approach bridges this gap. This enables more accurate clustering in point clouds with mixed densities, guaranteeing reliable and accurate cluster recognition even in areas with diverse spatial properties.

Acquiring the statistical parameters required for the model by looking at different point attributes in a large dataset [8]. To bridge this gap, the 3DPCD-TPK approach uses advanced dimensionality reduction t-SNE to simplify the statistical parameter extraction process. The method increases the accuracy and efficiency of parameter acquisition, enabling more effective model training and increased overall performance in processing huge point cloud datasets. This is achieved by optimising the analysis of point attributes and combining adaptive clustering techniques DBSCAN and effective spatial indexing K-D tree. The existing approaches for normalising and cleaning LiDAR data, which rely on axis outlier identification and circular differential cosine transformation, frequently find it difficult to handle high-dimensional data and guarantee reliable signal decomposition [9]. The proposed 3DPCD-TPK approach combines LZ4 for effective compression with t-SNE, KD-trees, and DBSCAN for improved data structuring and clustering. Both the effectiveness of compression and the accuracy of data representation are enhanced by this combined method.

The proposed method 3DPCD-TDK codec seeks to address the difficulties involved in processing and interpreting the LiDAR 3D PCD by combining these innovative techniques into a cohesive framework. To overcome the aforementioned difficulties, provide a thorough framework that incorporates revolutionary techniques into various LiDAR data processing stages. The ability of the proposed PCD compression model to produce high compression ratios, low storage needs, and quick processing times without sacrificing structural integrity serves as justification for its effectiveness. The following are the main contributions made by proposed 3DPCD-TDK method:

- Gaussian-based t-SNE (t-Distributed Stochastic Neighbor Embedding) is utilized for dimensionality reduction. It provides an efficient visualization and analysis in High-dimensional 3D LiDAR data.
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise) technique is utilized for adaptive segmentation of LiDAR 3D PCD into coherent clusters in a robust and adaptable.
- The proposed method utilizes K-D tree hierarchical representation to index and organize compressed LiDAR data, enabling expedient and scalable spatial querying.
- Effective Compression using LZ4 technique: The proposed 3DPCD-TDK framework effectively provides lossless compression of LiDAR 3D PCD by utilizing the LZ4 technique, guaranteeing high compression ratios with low computational overhead.

The rest of this paper are arranged as follows: A methodology for 3D PCD compression is described in Section 2, along with the compression procedure and suggested block diagram. Section 3 discusses the findings presenting the experimental results and performance assessments on the benchmark dataset. The work is finally brought to a close in Section 4, which highlights the conclusion and potential future directions of this research.

2 Methods

Effectively compressing the 3D LiDAR PCD has become essential for preserving storage capacity and facilitating instant transmission in areas with limited resources in real-time LiDAR applications like environmental monitoring and autonomous vehicles. The performance of the proposed 3DPCD-TPK method was evaluated using two different datasets. The Sydney Urban (PCD) Datasets [10], and The Waupaca County Land Information (LAS) Laser formats [11], both data focusses on topographical and geographic aspects and contains detailed land information in LAS format. The datasets are made up of 3D point cloud data with colour attributes and x, y, and z coordinates. The proposed codec was implemented according to its specifications on a dataset. Python Notebook Version of Jupyter of 3.10.

The proposed method utilized the LZ4 algorithm from low-latency compression. In order to leverage tensor-based operations efficiently and enable networks to learn hierarchical representations of point cloud data is described in [12]. By combining the effective compression of LZ4 with hierarchical learning approaches, the proposed 3DPCD-TPK approach closes this gap. The approach makes sure that tensor operations are optimised for high-performance learning by utilising DBSCAN for clustering,

KD-trees for spatial indexing, and t-SNE for dimensionality reduction.

In the framework of OT-based point cloud compression, binary-tree (BT) partitions are presented for the first time as alternative geometry partition types. Higher coding efficiency and variable three-dimensional (3D) spatial representations are made possible by the adaptive geometry partition technique[13]. efficient compression method for voxelized 3D point cloud properties. An input voxelized 3D point cloud is first split into equal-sized blocks. Next, a geometry-guided sparse representation (GSR) is suggested to deal with the uneven structure of 3D point clouds by removing the redundancy inside each block[14]. A method for point cloud resampling that guarantees uniform point cloud density within and outside of feature curves are discussed[15]. 3D point cloud data require a significant amount of storage capacity, which poses a significant obstacle to effective communication. Nevertheless, compressing sparse, disordered, non-uniform, and high dimensional data efficiently remains a challenge[16]. The proposed model effectively compresses PCD while permitting fast decompression with minimal data loss by fusing statistical, clustering, and entropy-based approaches. This makes it appropriate for practical uses including LiDAR storage, urban mapping, and 3D reconstruction.

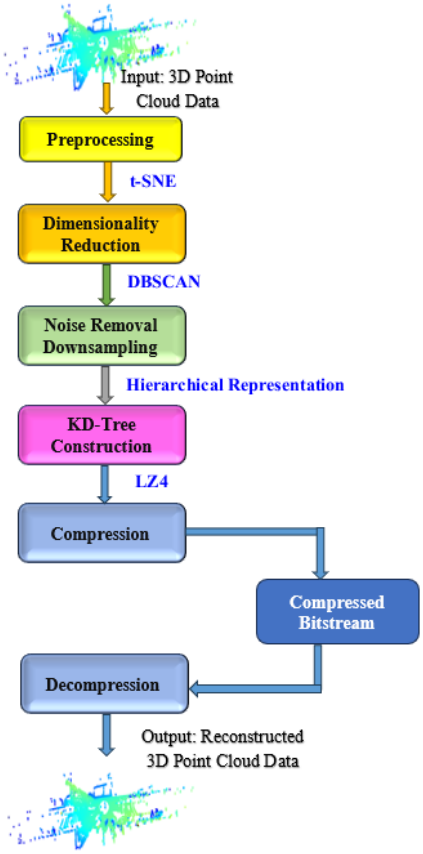


Fig 1. Process Flow of Proposed 3DPCD-TDK Work

The Figure 1 illustrates the proposed codec process flow for compressing 3D PCD. The proposed methodology delineates the sequential stages of the t-SNE algorithm, spanning from the initialization phase with input data to the acquisition of final embeddings in a lower-dimensional space. Applying DBSCAN clustering and segmentation to the 3D PCD is the next stage. This procedure involves dividing the data into density-based clusters, which separates the PCD into different regions. The PCD is represented more effectively and compactly. Using K-D trees is one method for hierarchical representation after the 3D PCD. Lastly, the decomposed 3D PCD is compressed using the LZ4 technique. Massive files like PCD are processed with LZ4 algorithm, a quick compression method that offers effective data compression and decompression. The PCD can decrease the amount of storage space required for preserving the PCD while keeping all of its significant information by using LZ4 compression.

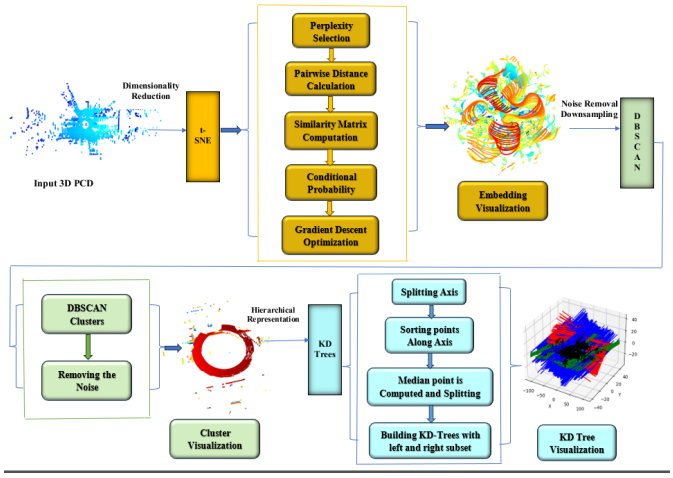


Fig 2. Proposed 3DPCD-TDK Architecture

The above Figure 2 describes the architecture of 3DPCD-TDK with the results is to effectively preprocess and compress 3D PCD while maintaining key characteristics. Emphasizing how t-SNE lowers computing cost and makes future grouping easier. Based on density, DBSCAN locates point clusters in the reduced space. Explain how it partitions the data into meaningful clusters by combining points that are densely packed together. K-D trees make it possible to efficiently retrieve nearby points, which is necessary for compression.

2.1 T-SNE (t-Distributed Stochastic Neighbor Embedding) for Geometric Redundancy reduction

In the process of condensing 3D PCD using t-SNE, the two most significant steps are the computation of the similarity matrix and the selection of the cost function, that is predominantly portrayed by the Kullback-Leibler (KL) divergence. This technique minimises distortions in the distances between neighbouring data points while locating lower-dimensional embeddings of the data points[17]. Detailed overview of the employing of KL divergence and Gaussian kernel similarity matrix in t-SNE compression for 3D PCD:

2.1.1 Similarity matrix with Gaussian kernel

A. Implementing of Similarity Matrix:. The first stage in t-SNE is building a similarity matrix which tracks pairwise similarities between high-dimensional space locations. In 3D PCD, a Gaussian kernel function is a technique used for estimating the similarity between two points. Leveraging the Euclidean distances between the points in the high-dimensional space, the Gaussian kernel determines how similar the points are to their neighbouring point.

Each point is represented by its XYZ coordinates in the context of 3D PCD.

$$P_i = (x_i, y_i, z_i) \tag{1}$$

In 3D space, the Euclidean distance between two points $P_i \; and \; P_j$ is determined using

$$Euclidean\ Distance = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \tag{2}$$

The linear distance between the two points, which indicates their geometric separation in three dimensions is computed using the above equation. Pairwise similarities between points in the similarity matrix for Gaussian kernel computation is determined using their Euclidean distances.

In the similarity matrix, each element P_{ij} denotes the similarity between points P_i, P_j. A Gaussian kernel function, which takes into account the Euclidean distance between points, is frequently used to calculate the similarity P_{ij}:

$$P_{ij} = \frac{exp\left(-\frac{\|P_i - P_j\|}{2\sigma^2}\right)}{\sum_{k \neq l} exp\left(-\frac{\|P_k - P_l\|}{2\sigma^2}\right)} \tag{3}$$

Where, i and j denotes the indices of the two points for which we are calculating the similarity in the similarity matrix.

k and l are the normalisation term is determined by aggregating over all other point pairs in the dataset using the indices pursuant to their Euclidean distance, pairs of points similarity are determined by the Gaussian kernel function.

P_{ij} is the similarity between points P_i and P_j in the similarity matrix.

$\Sigma_{k\neq l}$ represents the total of all point pairings $P_k$and P_l, where k and l are indices that vary over all points in the dataset with the exception of i and j. This guarantees the summation of all other points in the collection, with the exception of P_i and P_j points.

$exp\left(-\frac{\|p_i-p_j\|}{2\sigma^2}\right)$ represents the Gaussian kernel function, which determines how similar two points are in relation to adjacent ones with $P_k$and P_l. The $-$ is the Euclidean distance between points $P_k$and P_l and σ is the bandwidth parameter regulating the kernel's width. According to their respective distance from Euclidean.

$exp\left(-\frac{\|p_k-p_I\|}{2\sigma^2}\right)$ represents the tailored pair of points P_i and P_j for which to intend constructing the similarity matrix entry of P_{ij}, this term computes the similarity between them.

The degree of precision of the similarity matrix in preserving the geometric relationships between points is ensured by the Euclidean distance metric. In order to create the similarity matrix using the Gaussian kernel, it serves as a framework for identifying the pairwise similarities. Through a combination of a Gaussian kernel and the Euclidean distance metric, the similarity matrix precisely captures the global and local structure of the 3D PCD.

2.1.2 Kullback-Leibler (KL) Divergence

The metric known as Kullback-Leibler (KL) divergence is used to measure the disparity between two probability distributions. KL divergence functions as a cost function in the context of t-SNE (t-Distributed Stochastic Neighbour Embedding) aims to minimise the difference between the primeval high-dimensional data similarity structure and its representation in a lower-dimensional space.

Based on pairwise similarities between data points, a similarity matrix P is constructed in the original high-dimensional space. For mapping high-dimensional data points to a lower-dimensional space while maintaining pairwise similarities is the intent of t-SNE. Depending on the embedded points a further similarity matrix Q is created in the low-dimensional space.

The KL divergence $(KL(P(Q))$ measures the difference between the distributions which are represented by P and Q. To compute the KL Divergence

$$(KL(P(Q)) = \Sigma_i \Sigma_j \, p_{ij} log \frac{p_{ij}}{q_{ij}} \tag{4}$$

Where, $p_{ij}\, and\, q_{ij}$ are the elements of the Similarity matrices P and Q. t-SNE determines a lower-dimensional embedding that accurately replicates the underlying structure of the high-dimensional data by minimising the KL divergence. KL Divergence value for the sample input data Scan2299.pcd t-SNE dimensionality reduction and embedded data shape of (52980, 3) as displayed results in Table 1.

2.2 Segmentation and Clustering using DBSCAN

Following the geometric redundancy of 3D PCD has been compressed using t-SNE, the compressed points conceivably grouped relative to their spatial density using DBSCAN. A density-based spatial clustering of applications with noise system (DBSCAN). DBSCAN uses the Euclidean distance to determine precisely how far distant points are in 3D space.

$$Q = (q_1, q_2, \ldots, q_n) \tag{5}$$

Where, the 3D coordinates in 3D PCD the reduced dimensional space is denoted by Q. DBSCAN is used to find clusters in this space based on the reduced-dimensional points in t-SNE Q. For each point q_i is to determine the number of points inside a given radius ε, which is an estimate of the local density surrounding each point is provided by this,

$$N\varepsilon(q_i) \tag{6}$$

The method builds from core points to identify clusters. For every significant basis q_i it locates every point in the ε neighbourhood indicating as Equation (6) . These points are appended to the identical cluster as q_i and any of these points are core points, recursively adding points to the cluster by exploring their ε neighbourhood.

$$N\varepsilon(q_i) \geq minPts \tag{7}$$

Where minPts is the minimal number of points needed to create a cluster, then a point is regarded as a core point. These centres are to form cluster centres.

$$N\varepsilon(q_i) = (q_i\varepsilon \mid distance\,(q_i, q_j) \leq \varepsilon\} \tag{8}$$

Where q_i is the specific point in the 3D PCD that is frequently referred to as the "centre" point and q_i is the other point in PCD. The $N\varepsilon(q_i)$ indicates the ε neighbourhood of the point q_i. This neighbourhood is made up of all the points q_j in the PCD Q such that the Euclidean distance between $q_i$and $q_j$is less than or equal to ε.

The method builds from core points to identify clusters. For every significant basis q_i it locates every point in the ε neighbourhood indicating as in Equation (6). These points are appended to the identical cluster as q_i and any of these points are core points, recursively adding points to the cluster by exploring there ε neighbourhood. Noise is defined as points that are neither core points nor part of any cluster. Each point is either labelled as noise or assigned to a cluster once it has been processed. It generates core points and expands clusters around them according to density as it iterates through each point in PCD. Since point cloud detection is density-based, it is possible to precisely identify and reject noise and false targets that have low-density distributions[18].

Within PCD, the clustering is often leveraged to locate unique items or areas of interest. Points with dense regions are grouped together, Outliers are identified as Noise. In the input data Scan2299.pcd process of DBSCAN Number of clusters: 13 and Removed 930 points labeled as noise as the results depicted inTable 3.

2.3 Hierarchical Representation for LiDAR 3D point cloud data using K-D Trees

The points are represented in PCD as Equation (1). A binary tree with an axis-aligned hyper-rectangle represented by each node is called a K-D tree. Splitting the hyper-rectangle along a selected dimension $(x,\ y, or\ z)$ at each level of the tree. Every node has pointers to its left and right child nodes along with a reference to a point in the dataset. The relationship between PCD and local neighbourhood surface variations during the neighbourhood recovery process and finds the dynamically scaled ideal point cloud neighbourhoods have been articulated[19].

Then partitioning the 3D space recursively by selecting the dimension with the greatest point spread in order to build the K-D tree. As moving down, the tree needs to switch between the $(x,\ y, and\ z)$ dimensions for 3D PCD. The splitting dimension at $level\ l$ and represented as d_l.

In order to divide a set of points $P^{'}$ at a specific node into two subsets and estimating the median along the splitting dimension d_l . Let m is the median of the coordinates of the points along dimension d_l. After that segregate $P^{'}$ into two sets:

$$P^{'}{}_L = \Big((x,\ y,\ z) \in P^{'} \Big| d_l\ (x,y,z) < m\} \tag{9}$$

$$P^{'}{}_R = \Big((x,\ y,\ z) \in P^{'} \Big| d_l\ (x,y,z) \geq m\} \tag{10}$$

In construction of a node that represents the hyper-rectangle that is defined by the points in $P^{'}$. The left and right subtrees are then recursively constructed through passing the $P^{'}{}_L$ and $P^{'}{}_R$ to the child nodes respectively, and selecting the subsequent splitting dimension.

The buildKDTree is a node that stores a point and the splitting dimension in the K-D tree.

Input: *3D Point Cloud Data (points P, depth d)*

Output: *KD Tree construction*

function buildKDTree(points P, depth d):
if P is empty:
return null
else:
dimension dim = d % 3 // alternate between x, y, z dimensions
sort points P along dim
median_index = |P| / 2
median_point = P[median_index]

node = new KDTreeNode(median_point, dim)
node.left_child = buildKDTree(P[0:median_index-1], d+1)
node.right_child = buildKDTree(P[median_index+1:], d+1)
return node

2.4 Compression of LiDAR 3D point cloud data using LZ4

Following K-D Tree construction, utilizing (Lempel-Ziv 4) LZ4 compression to further reduce the file size while maintaining lossless reconstruction, which makes the proposed 3DPCD-TDK codec effective for 3D PCD compression.

As a quick and lossless compression technique, LZ4 is used. By detecting recurring patterns and substituting them with shorter references, LZ4 effectively encodes spatially connected points while drastically lowering storage needs. LZ4 places a higher priority on speed than entropy-based compression methods, allowing for high fidelity decompression in real-time. The K-D Tree hierarchical structure increases pattern detection and redundancy removal, which boosts LZ4 efficiency. The 3DPCD-TDK Codec is very effective for 3D PCD compression because of its integration, which guarantees that the final compressed representation stays compact while maintaining the integrity of the original PCD.

2.5 Decompression of 3DPCD-TPK Approach

Decompression of the 3D PCD while preserving structural integrity and reducing reconstruction loss is the goal of the decompression procedure. Following the proposed 3DPCD-TDK compression pipeline with t-SNE, DBSCAN, KD-Tree, and LZ4, the following procedures describe how decompression is carried out:

In order to recreate the original PCD while maintaining its structural integrity, the decompression procedure flips the compression phases. Initially, LZ4 decompression reverses the entropy reduction used during compression in order to recover the encoded spatial information. The hierarchical spatial arrangement of points is then recovered by traversing the KD-Tree structure in reverse. In order to resemble the original distribution, DBSCAN then reconstructs the density-based reduced points, hence expanding the previously detected clusters. The reduced-dimensional representation is then mapped back to the original feature space by the t-SNE inversion, which uses a learnt or approximated mapping for reconstruction. This sequential process ensures that the PCD is reconstructed with minimal deviation from the original, evaluated through metrics such as EMD, MSE, and PSNR, demonstrating the effectiveness of the proposed 3DPCD-TDK method in achieving high-fidelity decompression.

3 Results and Discussion

A real-time dataset for 3D PCD processing and urban planning tasks is the Sydney Urban Dataset[10] in .pcd (Point Cloud Data) file format. In total, it consists of 18 sub-datasets, each of which depicts a range of urban environments and actual circumstances. The dataset records detailed 3D PCD with attributes including timestamps, position, color, and spatial coordinates $(x,\ y,\ and\ z)$. Other characteristics, parameters of surface reflectance-reflecting intensity. Waupaca County's high-resolution LiDAR PCD is available in the Waupaca County Land Information resource dataset[11], a geospatial resource. Environmental planning, land-use analysis, and topographic mapping are the main applications for the dataset. It comes in the .las (Laser) file format, which is used to store LiDAR PCD. This dataset contains elevation values for terrain modelling in addition to 3D spatial coordinates $(x,\ y,\ and\ z)$, which reflect exact places in space. It also includes color and intensity measurements that show surface reflectance, which aids in differentiating between various land cover types.

3.1 Proposed Work 3DPCD-TPK Step-wise Results

3.1.1 T-SNE Geometric Redundancy Reduction Analysis

The 3D PCD was compressed using t-SNE, and the following outcomes were attained:

A. Nearest Neighbor Computation. Every data point in the dataset was linked to its 91 closest neighbours in the high-dimensional space, according to the t-SNE computation of the 91 nearest neighbours for each sample in (Scan2299.pcd). The local correlations and structures found in the dataset are capable of being captured and preserved by t-SNE through to this computation.

B. Conditional Probability Computation. For each of the 100 samples in the dataset, conditional probabilities were calculated to show how likely it was that any given point would be its neighbour given its distance from another.

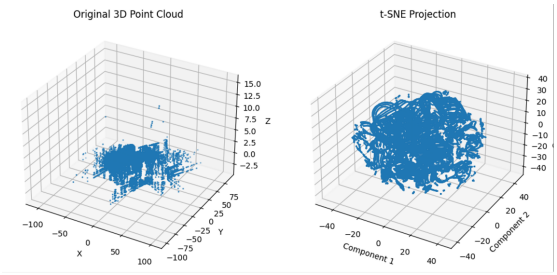
C. Mean Sigma Calculation. Based on the computation of conditional probabilities, the mean sigma value for the average of 0.311118 for Scan2299.pcd was found to represent the average scale of the Gaussian kernels.

D. Embeddings. The reduced-dimensional coordinates of the original dataset points are represented by the final embeddings. The three-dimensional embeddings enable the compressed data to be visualized and analyzed. The conventional of "KDE" is "Kernel Density Estimation". It is a method for calculating a random variable probability density function. Estimating the underlying distribution of data points in a PCD which is a common challenge in analysis and visualisation tasks. The t-SNE technique incorporates kernel density estimation of the points in the embedded space.

E. KL Divergence Evaluation. To evaluate the fidelity of compression, the KL divergence between the original and compressed data distributions is calculated. Whereas a bigger KL divergence can suggest information loss during compression, a low KL divergence shows that the local structure and relationships seen in the original data are preserved in the compressed representation. As an illustration:

With early articulation, the KL divergence was measured to be 73.31 after 200 iterations for the input Scan2299.pcd results displayed in Table 1. The difference between the original high-dimensional space and the lower-dimensional embedding generated by t-SNE is indicated by this divergence value. The KL divergence significantly decreased to 2.235654 after 300 iterations suggesting that the t-SNE algorithm has successfully converged to a lower-dimensional representation with minimized divergence. The original and compressed data distributions KL divergence was calculated. The local structure and linkages in the original 3D PCD are efficiently preserved by t-SNE compression, as indicated by a low KL divergence for PCD in following Figure 3(a) symbolizes the original PCD, and Figure 3(b) projects the t-SNE embeddings for input Scan2299.pcd.

An important parameter in t-SNE is perplexity modifies the embedding by dividing attention between local and global data structures. A balance between capturing local and global structures in the PCD is sought after with a perplexity of 30. The KL divergence at different points following 200 and 300 iterations is used to gauge how well the optimisation process is performing. The mismatch in pairwise similarities between data points in the original high-dimensional space and their corresponding data points in the lower-dimensional embedding is measured by the cost function in t-SNE. Reduced cost function values are better in the context of t-SNE since they show a more successful embedding of the high-dimensional data into a lower-dimensional space, better maintaining the local structure and relationships within the data.



(a) (b)

Fig 3. Illustrates the t-SNE Embeddings for Input 3D PCD

The Table 1 displays the t-SNE performance analysis description Embedded values give the data in a clear that the original PCD data shape is embedded and succinct overview by presenting the results for the Sydney Urban Dataset[10].

The below Table 2 displays the t-SNE performance analysis description Embedded values give the data in a clear that the original PCD data shape is embedded and succinct overview by presenting the results for the Waupaca County Land Information Dataset[11].

3.1.2 DBSCAN Clustered Analysis

Clustering of DBSCAN utilising t-SNE reduced PCD. The reduced PCD was subjected to DBSCAN clustering after t-SNE dimensionality reduction in order to find clusters within the data. DBSCAN works effectively with real-world LiDAR data because it can identify groups of irregular forms. DBSCAN efficiently separates sparse (noise) from dense (cluster) regions. In LiDAR-based point cloud compression, where raw data frequently contains scattered points, outliers, and sensor noise, this is relevant for compression. DBSCAN is an efficient clustering technique in the proposed 3DPCD-TPK codec strategy because of its autonomous cluster recognition, noise resilience, and capacity to manage complicated structures. The proposed approach

Table 1. Performance Analysis of t-SNE Embedded values for Input Sydney Urban Dataset

Input PCD File	Original Data shape	Nearest Neighbour	KL Divergence after 200 Iterations	KL Divergence after 300 Iterations	Cost Function	Perplexity used for PCD	Embedded Data shape
Scan0.pcd	42165,3	91	96.26	2.24	2.2800	30	36862, 3
Scan270.pcd	56283,3	91	73.39	2.85	2.8419	30	54159, 3
Scan2299.pcd	55107,3	91	73.31	2.83	2.8249	30	52980, 3
Scan2446.pcd	56145,3	91	73.63	2.85	2.8473	30	54019, 3
Scan2738.pcd	59395,3	91	73.84	2.89	2.8896	30	56274, 3
Scan10974.pcd	55714,3	91	73.16	2.83	2.8243	30	53205, 3
Scan11204.pcd	56677,3	91	72.22	2.81	2.8014	30	51444, 3
Scan11290.pcd	57188,3	91	72.67	2.83	2.8264	30	48897, 3
Scan11886.pcd	59297,3	91	74.31	2.91	2.9068	30	57265, 3
Scan12119.pcd	59894,3	91	74.13	2.91	2.9096	30	56858, 3
Scan12346.pcd	54484,3	91	73.76	2.85	2.8520	30	51220, 3
Scan12715.pcd	53731,3	91	73.08	2.81	2.8051	30	51744, 3
Scan16217.pcd	54030,3	91	73.47	2.83	2.8309	30	52005, 3
Scan17038.pcd	57148,3	91	73.80	2.86	2.8605	30	54611, 3
Scan19631.pcd	56217,3	91	73.98	2.86	2.8646	30	53709, 3
Scan19761.pcd	58883,3	91	73.88	2.89	2.8901	30	54595, 3
Scan20631.pcd	58356,3	91	74.07	2.89	2.8945	30	54602, 3
Scan25322.pcd	55302,3	91	72.09	2.78	2.7849	30	51126, 3

Table 2. Performance Analysis of t-SNE Embedded values for Input Waupaca County Land Information Datasets

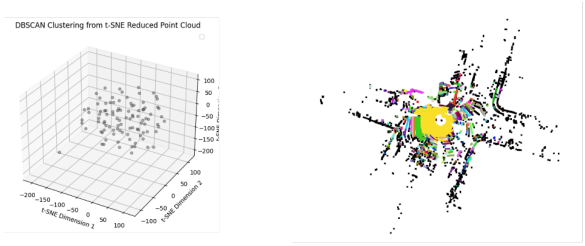
Input LAS File	Original Data shape	Nearest Neighbour	KL Divergence after 200 Iterations	KL Divergence after 300 Iterations	Cost Function	Perplexity used for PCD	Embedded Data shape
0043.las	3910870, 3	91	80.90	3.37	3.3713	30	96780, 3
0088.las	4684060, 3	91	80.94	3.40	3.3961	30	99780,3
0099.las	4422580, 3	91	80.45	3.38	3.3751	30	98900,3
0329.las	3378400, 3	91	77.75	3.06	3.0604	30	65540,3
0426.las	4033675,3	91	80.37	3.35	3.3551	30	96890,3
0427.las	3682674, 3	91	81.14	3.29	3.2917	30	87900, 3
0486.las	3826996, 3	91	79.33	3.22	3.2156	30	79700, 3
0487.las	4164541, 3	91	80.21	3.30	3.3034	30	88700, 3
0515.las	3787304, 3	91	81.01	3.36	3.3670	30	95000, 3
0843.las	4367881, 3	91	81.33	3.41	3.4108	30	101680, 3

guarantees that clusters are produced based on the intrinsic structure of the 3D PCD by utilizing DBSCAN's adaptive clustering capability, which results in an efficient compression procedure. The following outcomes were obtained from the clustering process:

A. Number of Clusters. Within the reduced PCD, the DBSCAN algorithm found a total number of clusters. A cluster is a collection of data points that are closely spaced apart and have similar properties.

B. Noise Removal. The points in PCD which were recognised and classified as noise during the clustering phase. These points were regarded as outliers or data noise as they did not fit into any of the clusters that had been found. The structure and distribution of the data inside the reduced are clarified by the DBSCAN clustering results. The points arranged using the data clustering technique to facilitate effective querying. To achieve continuous levels, it should be mentioned that further

multi-dimensional indexing and clustering algorithms could be used [20]. Based on density, DBSCAN automatically calculates the number of clusters. This is especially helpful for PCD, as the quantity of natural clusters varies from various dataset and is frequently unknown. The DBSCAN method assists in finding significant patterns and correlations in the data by locating clusters and eliminating noise points, which makes subsequent analysis and interpretation attainable is shown in below Figure 4.



(a) (b)

Fig 4. Illustrates the DBSCAN Clustering from t-SNE Embeddings for Sample 3D PCD

Above Figure 4(a) displays the DBSCAN algorithm found 13 clusters, which are represented by different colours in the result. In the 3D PCD of Figure 4(b) shows the (Scan2299.pcd) discern between several groups of points by assigning a distinct colour to each cluster as shown in Figure 4. A unique colour is assigned to every cluster, setting it apart from other clusters. This makes it possible for distinguish and visually recognise several spatially related groupings of points inside the PCD. Noise-classified points are shown in black. Based on the supplied eps and min_samples parameters, these are the points that are not part of any cluster that has been discovered and displayed the detailed results in the beneath Table 3 describes the outcomes of the DBSCAN on 3D PCD. The structured overview of the clusters that were produced after the clustering process for Sydney Urban Dataset [10].

Table 3. Performance Analysis of Clustering and removing the Noise values for Input Sydney Urban Dataset

Input PCD File	Original size of PCD in KB	Number of Points in PCD	Number of Clusters	Noise (Removing points) in PCD
Scan0.pcd	824	42165	9	957
Scan270.pcd	1100	56283	5	977
Scan2299.pcd	1077	55107	13	930
Scan2446.pcd	1097	56145	11	943
Scan2738.pcd	1161	59395	14	927
Scan10974.pcd	1089	55714	13	935
Scan11204.pcd	1108	56677	22	880
Scan11290.pcd	1118	57188	12	938
Scan11886.pcd	1159	59297	8	995
Scan12119.pcd	1170	59894	5	980
Scan12346.pcd	1065	54484	7	990
Scan12715.pcd	1050	53731	8	965
Scan16217.pcd	1056	54030	13	936
Scan17038.pcd	1117	57148	8	963
Scan19631.pcd	1099	56217	9	977
Scan19761.pcd	1151	58883	17	918
Scan20631.pcd	1140	58356	11	944
Scan25322.pcd	1081	55302	31	830

The following Table 4 describes the outcomes of the DBSCAN on 3D PCD. The structured overview of the clusters that were produced after the clustering process for Waupaca County Land Information Dataset.

Table 4. Performance Analysis of Clustering and removing the Noise values for Input Waupaca County Land Information Dataset

Input LAS File	Original size of LAS in MB	Number of Points in LAS	Number of Clusters	Noise (Removing points) in LAS
0043.las	104.43	3910870	2	1600
0088.las	125.08	4684060	3	1800
0099.las	118.10	4422580	1	1000
0329.las	90.22	3378400	3	2800
0426.las	107.71	4033675	1	1400
0427.las	98.34	3682674	1	900
0486.las	102.19	3826996	2	1450
0487.las	111.21	4164541	1	1660
0515.las	101.13	3787304	1	1320
0843.las	116.64	4367881	7	4590

3.1.3 Hierarchical Representation of PCD

A. Nearest Point Identification. The determination of the PCD closest point to a specific query point. The coordinates of the nearest point are discovered and used to represent it, and then the Euclidean distance from the query point is computed.

B. Building K-D Tree. The output shows how to build a K-D tree data structure to make closest neighbour searches more effective. K-D trees facilitate quick neighbour searches by arranging the dataset into a hierarchical binary tree.

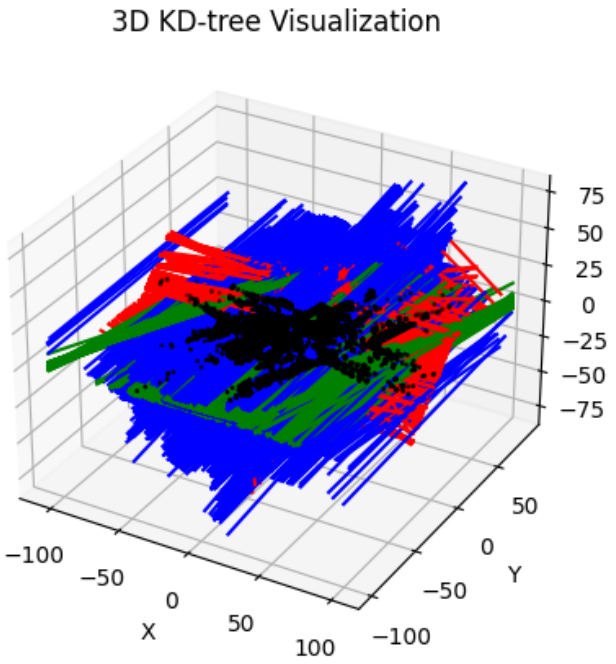


Fig 5. Illustrates the K-D Tree representation of PCD

The above Figure 5 shows the result of (Scan2299.pcd) that searches for another query point using its nearest neighbour after constructing the K-D tree. The nearest point that was found is shown, along with its coordinates and separation from the query location. The nearest neighbouring locations located in the dataset are listed in the output along with their indices and distances.

3.2 Combined Technique Results

Outlining the findings in a proposed 3DPCD-TDK which included t-SNE reduction, K-D trees creation, and PCD compression using LZ4 for Original Input Sydney Urban Dataset presented in below Table 5 .

Table 5. Results of Proposed 3DPCD-TDK Codec for Original Sydney Urban Dataset

Input PCD File	Original PCD File size in KB	Original PCD File size in Bytes	After t-SNE redundancy reduction in KB	After K-D Trees Construction in KB	Proposed 3DPCD-TDK Codec in Bytes	Compression Percentage of using Proposed 3DPCD-TDK Codec
Scan0.pcd	824	843776	663.79	193.41	335	99.96%
Scan270.pcd	1100	1125859	675.56	196.55	631	99.94%
Scan2299.pcd	1077	1102339	661.45	192.79	823	99.92%
Scan2446.pcd	1097	1123099	673.90	196.11	643	99.94%
Scan2738.pcd	1161	1188099	712.90	206.51	547	99.95%
Scan10974.pcd	1089	1114479	668.73	194.73	463	99.95%
Scan11204.pcd	1108	1133739	680.29	197.81	999	99.96%
Scan11290.pcd	1118	1143959	686.42	199.45	631	99.91%
Scan11886.pcd	1159	1186139	711.73	206.19	687	99.95%
Scan12119.pcd	1170	1198079	718.90	208.10	663	99.94%
Scan12346.pcd	1065	1089879	653.97	190.79	511	99.95%
Scan12715.pcd	1050	1074819	644.93	188.38	621	99.94%
Scan16217.pcd	1056	1080799	648.52	189.34	571	99.95%
Scan17038.pcd	1117	1143159	685.94	199.32	679	99.94%
Scan19631.pcd	1099	1124539	674.77	196.34	543	99.95%
Scan19761.pcd	1151	1177859	706.76	204.87	535	99.95%
Scan20631.pcd	1140	1167319	700.43	203.18	683	99.94%
Scan25322.pcd	1081	1106239	663.79	193.41	839	99.90%

Outlining the findings in a proposed 3DPCD-TDK which included t-SNE reduction, K-D trees creation, and PCD compression using LZ4 for Original Input Waupaca County Land Information Dataset presented in below Table 6.

Table 6. Results of Proposed 3DPCD-TDK Codec for Original Waupaca County Land Information Dataset

Input LAS File	Original LAS File size in MB	Original LAS File size in Bytes	After t-SNE redundancy reduction in MB	After K-D Trees Construction in MB	Proposed 3DPCD-TDK Codec in MB	Compression Percentage of using Proposed 3DPCD-TDK Codec
0043.las	104.43	3910870	3.36	2.27	1.14	98.91%
0088.las	125.08	4684060	3.27	2.21	1.11	99.11%
0099.las	118.10	4422580	3.33	2.25	1.13	99.04%
0329.las	90.22	3378400	2.16	1.45	0.75	99.17%
0426.las	107.71	4033675	3.27	2.21	1.11	98.97%
0427.las	98.34	3682674	2.99	2.02	1.01	98.97%
0486.las	102.19	3826996	2.59	1.75	0.91	99.11%
0487.las	111.21	4164541	3.02	2.03	1.01	99.09%
0515.las	101.13	3787304	3.21	2.17	1.09	98.92%
0843.las	116.64	4367881	3.41	2.31	1.16	99.01%

The following Figure 6 depicts the 3D PCD that makes up the input data (a), (b), (c) from [10] and (d), (e), (f) from [11].

PCD is shown in the below output Figure 7 shows size was reduced without a noticeable loss of information.PCDproposed method PCDmay be efficiently preserved and transferred to this compression.

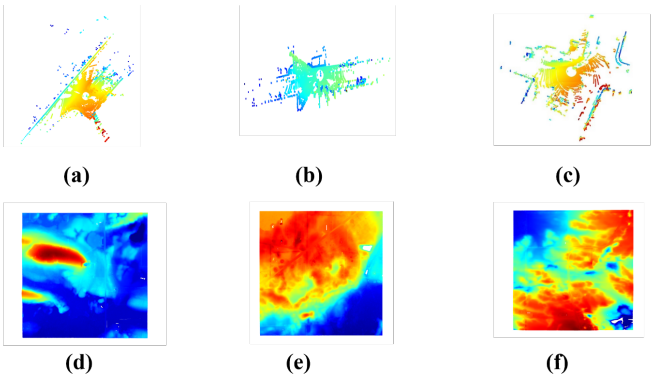


Fig 6. Sample Input data from two Different Datasets (a) Scan11886.pcd, (b) Scan19761.pcd, (c) Scan2299.pcd, (d) 0043.las and (e) 0843.las, (f) 0329.las

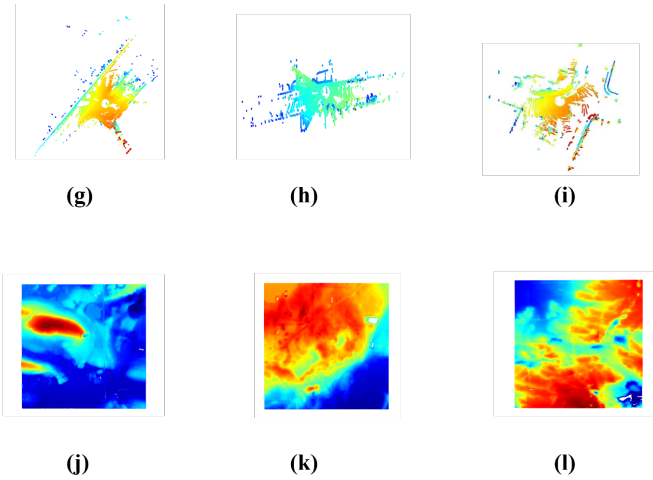


Fig 7. Sample Output data from 3DPCD-TDK of two Different Datasets (g) Scan11886.pcd, (h) Scan19761.pcd, (i) Scan2299.pcd, (j) 0043.las, (k) 0843.las, and (l) 0329.las

3.3 Compression Performance Metrics

3.3.1 Mean Squared Error

With 3D point cloud data (PCD), the Mean Squared Error (MSE) is determined with the following equation:

$$MSE = \frac{1}{N}\sum_{i=0}^{N}(P^{(i)}original - P^{(i)}compressed)^2 \tag{11}$$

Where, N represents the total number of points. The coordinates of the ith point in the original PCD are represented by $P^{(i)}$ (original). In the compressed (reconstructed) PCD, $P^{(i)}$ (compressed) denotes the coordinates of the ith point. The MSE value of 41.91 indicates that, however very slightly, some distortion was introduced during compression.

3.3.2 Peak Signal Noise Ratio

The Peak Signal-to-Noise Ratio (PSNR) for 3D PCD is calculated using the following equation:

$$PSNR = 10.log_{10}\left(\frac{MAX^2}{MSE}\right) \tag{12}$$

In a 3D PCD, max is the highest value of the PCD in Mean Squared Error (MSE). The quality of the compressed 3D PCD is determined by the PSNR. Since greater PSNR values represent a higher ratio of the maximum potential pixel value to the mean

squared error, they are indicative of superior de quality. The compressed 3D LiDAR PCD is lossless and nearly similar to the original 3D PCD, as indicated by the PSNR of infinity decibels (dB).

3.3.3 Compression Ratio

The ratio of the size of the original PCD to the size of the compressed PCD is known as the Compression Ratio (CR) for 3D PCD. The following is the equation:

$$Compression\ Ratio = \frac{Size\ of\ the\ Original\ Point\ Cloud\ Data}{Size\ of\ the\ Compressed\ Point\ Cloud\ Data} \tag{13}$$

$$Compression\ Percentage = \left(1 - \frac{Compressed\ Size}{Original\ Size}\right) \times 100 \tag{14}$$

The size of the original, uncompressed PCD, expressed in terms of memory or storage, is known as the Size of Original PCD. The amount of memory or storage needed to hold the compressed 3D point cloud is known as the size of the compressed 3D PCD. The compression ratio indicates how much the 3D PCD is compressed using the proposed method 3DPCD-TDK. There has been a reduction in the size of the data, as shown by the superior compression ratio on the average of 100.55 times without compromising the fidelity of data.

3.3.4 Earth Mover's Distance (EMD)

Earth Mover's Distance (EMD) is used to calculate how different two probability distributions between groups of points are from one another. EMD calculates the minimal cost needed to convert one point cloud into another in order to quantify the geometric distortion in the context of 3D point cloud compression.

$$EMD\ (P,Q) = \min_{\emptyset:P\to Q} \frac{1}{N} \Sigma_{p\epsilon P} \| p - \emptyset(p) \|\ d\left(p, \emptyset(p)\right) \tag{14}$$

Where: P is the original PCD, Q is the reconstructed PCD, the Euclidean distance between corresponding points is $d\left(p,\ \emptyset\left(p\right)\right)$, and the best mapping from one set to another is represented as $\emptyset$. For the Sydney Urban [10] and Waupaca County [11] datasets, the proposed approach yields an EMD of 0, meaning that the reconstructed PCD geographic distribution is exactly the same as the original. This outcome demonstrates that 3DPCD-TDK Codec achieves lossless compression by maintaining the geometric structure of the point cloud data without generating distortion.

3.4 Performance Analysis

A performance analysis evaluating the proposed 3DPCD-TPK approach with existing compression algorithms viz 7-Zip, WinRAR, and LZ4 is shown in the accompanying Table 7 and Table 8. It contains the LAS and PCD file compressed sizes made with these techniques.

The performance efficiency of a compression technique is demonstrated in Figure 8 and Figure 9, which show the relationships between the PSNR, compression ratio, and loss function for the Sydney Urban Dataset (PCD files) and Waupaca County Land Information (LAS Files).

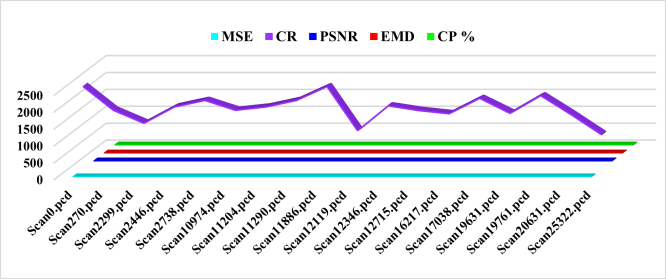


Fig 8. Analysis of Compression Performance: MSE, PSNR, EMD, CR, and CP in the Sydney Urban Dataset

Table 7. Performance Analysis of Proposed 3DPCD-TDK Codec for Original Sydney Urban Dataset (PCD Files) with existing compression techniques

Input PCD File	Original size of PCD in KB	Compressed size of PCD with 7-Zip in KB	Compressed size of PCD with LZ4 in KB	Compressed size of PCD with WinRAR in KB	Compressed size of PCD using 3DPCD-TDK in Bytes
Scan0.pcd	824	439	606	456	**335**
Scan270.pcd	1100	587	842	619	**631**
Scan2299.pcd	1077	582	821	608	**823**
Scan2446.pcd	1097	587	889	618	**643**
Scan2738.pcd	1161	622	958	645	**547**
Scan10974.pcd	1089	590	878	620	**463**
Scan11204.pcd	1108	618	926	630	**999**
Scan11290.pcd	1118	604	922	625	**631**
Scan11886.pcd	1159	608	948	732	**687**
Scan12119.pcd	1170	608	916	612	**663**
Scan12346.pcd	1065	612	805	560	**511**
Scan12715.pcd	1050	550	834	579	**621**
Scan16217.pcd	1056	575	859	575	**571**
Scan17038.pcd	1117	593	834	605	**679**
Scan19631.pcd	1099	583	888	583	**543**
Scan19761.pcd	1151	612	944	616	**535**
Scan20631.pcd	1140	603	941	604	**683**
Scan25322.pcd	1081	591	907	603	**839**

Table 8. Performance Analysis of Proposed 3DPCD-TDK Codec for Original Waupaca County Land Information Dataset (LAS Files) with existing compression techniques

Input LAS File	Original size of LAS in MB	Compressed size of LAS with 7-Zip in MB	Compressed size of LAS with LZ4 in MB	Compressed size of LAS with WinRAR in MB	Compressed size of PCD using 3DPCD-TDK in MB
0043.las	104.43	29.84	67.43	21.14	**1.14**
0088.las	125.08	32.98	77.57	19.60	**1.11**
0099.las	118.10	31.45	73.57	19.47	**1.13**
0329.las	90.22	25.23	57.06	15.52	**0.75**
0426.las	107.71	30.13	65.93	22.55	**1.11**
0427.las	98.34	28.34	61.44	21.81	**1.01**
0486.las	102.19	29.66	65.57	21.76	**0.91**
0487.las	111.21	29.75	68.22	19.77	**1.01**
0515.las	101.13	28.15	63.34	19.22	**1.09**
0843.las	116.64	32.59	73.69	21.68	**1.16**

4 Conclusion

The evaluation demonstrated that applying t-SNE dimensionality reduction and a proposed 3DPCD-TPK compression codec significantly increased data processing and storage efficiency for the given point cloud datasets. The KL divergence for Scan2299.pcd, which contained 55,107 points and a size of 1077 KB at first, decreased from 73.31 after 200 iterations to 2.83 after 300 iterations. Similarly, for 0329.las, which had 3,378,400 points and origin size of 90.22 MB, during the same number of iterations, the KL divergence dropped from 77.75 to 3.06, with a perplexity of 30 being applied for both datasets. With Scan2299.pcd reducing from (55,107, 3) to (52,980, 3) and 0329.las reducing from (3,378,400, 3) to (65,540, 3), post-t-SNE processing further decreased the data geometries. Scan2299.pcd size was decreased from 1077 KB to 661.45 KB following t-SNE, 192.79 KB with K-D Trees construction, and 823 bytes using the 3DPCD-TPK codec, resulting in increased storage efficiency. Similarly, the size of 0329.las shrank from 3,378,400 bytes to 2.16 MB following t-SNE, 1.45 MB using K-D Trees,

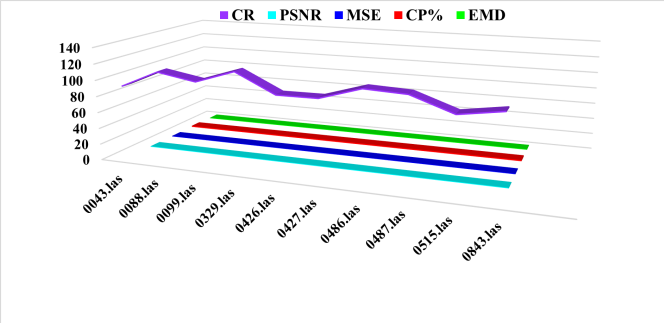


Fig 9. Performance Efficiency for Loss Function, PSNR and Compression Analysis for Waupaca County Land Dataset

and finally 0.75 MB with the proposed codec. With the MSE remaining near zero, there is little loss in reconstruction. High values for the PSNR attest to the compression's ability to preserve fidelity with very little distortion. With compression ratio of 1749.4 and the compression percentage is 99.96%. Additionally, the Sydney Urban and Waupaca County datasets have zero EMD, ensuring that the spatial distribution of points is unaltered following decompression. Comparing the standard methods viz WinRAR, 7-Zip, and LZ4 with proposed 3DPCD-TPK codec and revealed better performance than the state-of-the-art methods.

The 3DPCD-TPK codec exceptional ability to compress large point cloud datasets while preserving crucial data properties is demonstrated by these outcomes. It also accelerates the further data processing activities and makes storage and transmission more efficient. The approach has the potential for wider uses in a number of fields, including medical imaging and geographic data analysis, that are needed for the effective management of massive datasets. Future research could further optimize the codec and explore its application across different types of data to maximize its utility.

References

1) Li Y, Ou Z, Liu G, Yang Z, Chen Y, Shang R, et al. Three-dimensional point cloud object detection based on feature fusion and enhancement. *Remote Sensing*. 2024;16(6). Available from: https://www.mdpi.com/2072-4292/16/6/1045.
2) Liu W, Mei F, Wang C, Neill MO, Swartzlander EE. Data compression device based on modified LZ4 algorithm. *IEEE Transactions on Consumer Electronics*. 2018;64(1):110–117. Available from: https://ieeexplore.ieee.org/document/8306366.
3) Chithra PL, Tamilmathi AC. 3D LiDAR point cloud image codec based on Tensor. *Imaging Science Journal*. 2020;68(1):1–10. Available from: https://www.tandfonline.com/doi/full/10.1080/13682199.2020.1719747.
4) Zhang X, Gao W. Adaptive Geometry Partition for Point Cloud Compression. *IEEE Transactions on Circuits and Systems for Video Technology*. 2021;31(12):4561–4574. Available from: https://ieeexplore.ieee.org/document/9503374.
5) Wu G, Tian H, Lu G, Wang W. ParVoro++: A scalable parallel algorithm for constructing 3D Voronoi tessellations based on kd-tree decomposition. *Parallel Computing*. 2023;115. Available from: https://www.sciencedirect.com/science/article/abs/pii/S0167819123000017.
6) Serna-Serna W, De Bodt C, Alvarez-Meza AM, Lee JA, Verleysen M, Orozco-Gutierrez AA. Semi-supervised t-SNE with multi-scale neighborhood preservation. *Neurocomputing*. 2023;550. Available from: https://www.sciencedirect.com/science/article/abs/pii/S0925231223006197.
7) Qian J, Zhou Y, Han X, Wang Y. MDBSCAN: A multi-density DBSCAN based on relative density. *Neurocomputing*. 2024;576. Available from: https://doi.org/10.1016/j.neucom.2024.127329.
8) l Chithra P, bala SL. Deep learning model for 3D LiDAR point cloud codec based on tensor. *2023 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*. 2023;p. 687–693. Available from: https://www.researchgate.net/publication/378251734_Deep_Learning_Model_for_3D_LiDAR_Point_Cloud_Codec_Based_on_Tensor.
9) Tamilmathi AC, Chithra PL. Tensor block-wise singular value decomposition for 3D point cloud compression. *Multimedia Tools Applications*. 2022;81:37917–37938. Available from: https://link.springer.com/article/10.1007/s11042-021-11738-7.
10) Sydney Urban Objects Dataset. . Available from: https://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml.
11) Download LiDAR Tiles (.dwg & .las). 2020. Available from: https://hub.arcgis.com/documents/5b5c8bee456a4f71a3d9a4c1b15267e2.
12) Guo C, Shi Y, Wu H, Xiang Y, Li W, Zhang C, et al. A combination of library search and Levenberg-Marquardt algorithm in optical scatterometry. *Thin Solid Films*. 2023;767. Available from: https://www.sciencedirect.com/science/article/abs/pii/S0040609023000019.
13) Zhang J, Chen T, Ding D, Ma Z. G-PCC++: Enhanced Geometry-based Point Cloud Compression. In: Proceedings of the 31st ACM International Conference on Multimedia. ACM. 2023;p. 1352–1363. Available from: https://dl.acm.org/doi/10.1145/3581783.3613827.
14) Gu S, Hou J, Zeng H, Yuan H, Ma KK. 3D point cloud attribute compression using geometry-guided sparse representation. *IEEE Transactions on Image Processing*. 2020;29:796–808. Available from: https://ieeexplore.ieee.org/document/8816692.
15) Xue J, Men C, Liu Y, Xiong S. Adaptive neighborhood recovery method for machine learning based 3D point cloud classification. *International Journal of Remote Sensing*. 2023;44(1):311–340. Available from: https://www.tandfonline.com/doi/abs/10.1080/01431161.2022.2162354.
16) Yu J, Wang J, Sun L, Wu ME, Zhu Q. Point cloud geometry compression based on multi-layer residual structure. *Entropy*. 2022;24(11). Available from: https://www.mdpi.com/1099-4300/24/11/1677.

17) Zhou Y, Sharpee TO. Using Global t-SNE to Preserve Intercluster Data Structure. *Neural Computation*. 2022;34(8):1637–1651. Available from: https://pubmed.ncbi.nlm.nih.gov/35798323/.
18) Gao P, Luo S, Paul M. Rate-Distortion Modeling for Bit Rate Constrained Point Cloud Compression. *IEEE Transactions on Circuits and Systems for Video Technology*. 2022;33:2424–2438. Available from: https://ieeexplore.ieee.org/document/9957096.
19) Guo Z, Liu H, Pang L, Fang L, Dou W. DBSCAN-based point cloud extraction for Tomographic synthetic aperture radar (TomoSAR) three-dimensional (3D) building reconstruction. *International Journal of Remote Sensing*. 2021;42(6):2327–2349. Available from: https://www.researchgate.net/publication/348086369_DBSCAN-based_point_cloud_extraction_for_Tomographic_synthetic_aperture_radar_TomoSAR_three-dimensional_3D_building_reconstruction.
20) Van Oosterom P, Van Oosterom S, Liu H, Thompson R, Meijers M, Verbree E. Organizing and visualizing point clouds with continuous levels of detail. *ISPRS Journal of Photogrammetry and Remote Sensing*. 2022;194:119–131. Available from: https://www.sciencedirect.com/science/article/pii/S0924271622002647.