

Innovative Approaches for 3D LiDAR Point Cloud Data Compression: Graph Theory meets Wavelet Transform

S. Lakshmi bala¹ and Dr. PL. Chithra^{2,*}

¹ Ph.D. Research Scholar, Department of Computer Science, University of Madras, Chennai – 600 025, India.

² Professor, Department of Computer Science, University of Madras, Chennai – 600 025, India.

lakshmibala.sj@gmail.com , chitrasp2001@yahoo.com

To Cite this Article

S. Lakshmi bala¹ and Dr. PL. Chithra^{2,*}, ” **Innovative Approaches for 3D LiDAR Point Cloud Data Compression: Graph Theory meets Wavelet Transform**”
Musik In Bayern, Vol. 90, Issue 1, Jan 2025, pp43-64

Article Info

Received: 24-12-2024 Revised: 02-01-2025 Accepted: 11-01-2025 Published: 22-01-2025

Abstract

LiDAR (Light Detection and Ranging) point cloud data is essential for 3D spatial representation, because of its complex geometry and volume, it can be difficult to store and transmit. Its rich structural characteristics are frequently not preserved by conventional compression techniques. In order to resolve this, a mathematical framework is proposed for compressing point cloud data called 3DGWT-LAE (Graph Wavelet Transform with Laplacian matrix, Adjacency matrix, and Eigenvalues) is put forth. With the goal to capture the geometric structure, the framework first builds a sparse adjacency matrix using nearest neighbor graphs. The adjacency matrix yields the Laplacian matrix, which encodes smoothness and connectedness. A compact representation of the data is given by the eigenvalues obtained from the spectral analysis of the Laplacian. The data is then efficiently compressed while maintaining geometric integrity by applying multi-resolution wavelet filters, which produce coefficients via the GWT. The proposed 3DGWT-LAE mathematical framework is demonstrated by the qualitative and quantitative results obtained using a variety of 3D LiDAR point cloud datasets (Sydney Urban Datasets), validate the framework effectiveness and show that it can strike a compromise between compression and geometric integrity. The scan270.pcd from Sydney urban dataset, which had an initial size of 1100 KB, was used as input to test the proposed compression technique. The effectiveness of the proposed method was demonstrated by the substantial reduction in file size to 345.98 KB after the compression technique was applied. Furthermore, 492.70 KB was the size of the rebuilt file, guaranteeing accurate data restoration. The file was compressed to about **68.5** of its original size using this method, which yields a compression ratio of about 3.18. Crucially, the original data integrity is maintained because the reconstruction technique is lossless. These outcomes demonstrate the 3DGWT-LAE method capacity to effectively compress point cloud data while preserving its original quality during reconstructed point cloud data. Notably, it offers significant compression efficiency without sacrificing geometric precision, making point cloud data transmission and storage cost-effective.

Keywords: *LiDAR, 3D, Laplacian Matrix, Adjacency Matrix, Eigen values, Graph Wavelet Transform, Lossless, Geometric accuracy.*

I. Introduction

Light Detection and Ranging (LiDAR) is an advanced remote sensing technique that employs laser beams to measure distances with great precision. The device sends laser pulses to a target surface and measures the time it takes for the reflected signals to return. LiDAR systems use the speed of light as a constant to compute the distance to the target. LiDAR generates highly detailed spatial data by generating dozens or millions of pulses per second from many directions. The output of a LiDAR system is known as 3D Point Cloud Data (PCD), which is a dense set of data points in three dimensions. Each point indicates a specific location in the environment, which is commonly stated in Cartesian coordinates (x, y , and z). Depending on the system's capabilities, the PCD may include additional information such as intensity values (reflectance of laser pulses) and RGB color data. LiDAR PCD is a key component of current spatial analysis and modelling. Its superior resolution and precision enable ground-breaking applications in artificial intelligence, machine learning, and robotics. LiDAR PCD is becoming more accessible as compression and processing techniques progress, paving the path for new advancements in 3D data representation and manipulation. A vital component of three-dimensional (3D) spatial representation, LiDAR (Light Detection and Ranging) PCD presents significant storage and transmission issues because of its intricate geometry and vast volume. Conventional compression techniques typically fail short in effectively capturing point cloud data rich structural properties. As a result, 3DGWT-LAE an extensive mathematical framework for compressing the 3D point cloud data using wavelet transform and graph-based strategies is proposed. The exponential expansion of point cloud data necessitates the development of innovative compression methods that preserve geometric integrity while reducing data size. The significance of complex mathematical frameworks in resolving compression challenges is highlighted in the proposed work.

To highlight the main features of the proposed approach, paying particular attention to its impact, benefits, and innovation. This well-organized framework guarantees lucidity and highlights the significance of proposed approach.

- The proposed 3DGWT-LAE technique compresses 3D point cloud data using a mathematical framework based on the Graph Wavelet Transform (GWT).
- It combines wavelet analysis and spectral graph theory to accomplish effective compression while maintaining crucial topological and geometric characteristics.
- The proposed 3DGWT-LAE method guarantees lossless reconstruction with an infinite Peak Signal-to-Noise Ratio (PSNR) and a Mean Squared Error (MSE) of 0.001.
- The proposed technique 3DGWT-LAE preserves the correctness of reconstructed point cloud data, which is essential for applications that need high precision.
- Compression ratios (CR) are noticeably higher than with conventional techniques like WinRAR and 7-Zip.
- The method suitability for real-world 3D data with different levels of complexity was demonstrated by testing on the Sydney Urban Dataset. The ability to maintain fine details in 3D point cloud data, including corners and edges.

The next sections are organized as follows: Section II covers the discussions relevant prior work in the field of LiDAR point cloud data processing. Section III clarify the proposed methodology, detailing the procedures for preprocessing the data, the novel method utilized for processing and compression, and the framework for performance evaluation. The experimental results are shown in Section IV, demonstrating both quantitative and qualitative data to validate the proposed methodology. And lastly, Section V summarizes major findings and recommended paths for future research.

II. Related Works

One major barrier to the broad use of graph neural networks (GNNs) in many applications is their computational complexity. The number of parameters in GNN models increases quickly with the size of the input graph, which results in longer training and inference times, as explained in [1]. Because of their excellent performance on a wide range of tasks described in [2], graph neural networks, or GNNs, have been employed extensively in graph analysis. Modern TE optimisers rely on conventional optimisation methods like constraint programming, local search, or linear programming, which are described in [3]. The latest tensor decomposition approach is used in the compression phase to take use of the patterns and redundancies that are naturally present in the 3D point cloud data, as described in [4]. A 3D point cloud's irregularity, dispersion, and sparseness prevent it from being evenly arranged on a regular grid. Potential information loss and empty operation are two drawbacks of the convolution neural network (CNN) approach for 3D point cloud feature extraction, as stated in [5]. Ideal method for handling irregular point clouds, but their use in large-scale LiDAR point cloud processing is limited due to the enormous calculations required to search for nearby points in the graph, as detailed in [6]. In point cloud data coordinates, as well as displacement and shear to enhance the model's resilience and invariance to the geometric alterations detailed in [7]. The main query is whether sensor measurements, as detailed in [8], may be eliminated from the graph without significantly reducing the amount of information retained.

Data processing, transmission, and storage are severely hampered by the exponential increase in the volume of 3D data created, especially in the form of Light Detection and Ranging (LiDAR) point clouds (PCs). Point cloud (PC) representation of 3D visual information has shown to be a very versatile format with a wide range of applications, from machine vision jobs in the robotics and autonomous driving domains as detailed in [9] to multimedia immersive communication. Colour signals can now be compressed using graph-based coding approaches, fully utilising the geometry data provided in [10]. There is a lot of study being done on picture compression and retrieval. Low level image retrieval descriptors, which are based on statistical features of pixel values, will change as a result of lossy image compression, which reduces the visual quality of images and alters their actual pixel values, as explained in [11]. A more condensed representation of maps is necessary to minimise complexity while maintaining the localisation performance outlined in [12], as loading, communication, and processing of the original dense maps take a lot of time on the onboard computing platform. The sensor's massive data output, which causes a number of problems with transmission, processing, and storage. These issues can now be resolved by applying data compression methods to the point cloud mentioned in [13]. The effectiveness of the model has been confirmed using well-known and proven compression methods as WinRAR and 7-Zip, which are detailed in [14]. The axis outlier identification method outlined in [15] is used to clean and normalise the raw LiDAR data. The k-means algorithm, which is explained in [16], is used to partition 3D point clouds into clusters. The unequal distribution of geometric and spectral information described in [17] makes it difficult to extract local features in multispectral LiDAR point cloud sceneries. Dependable eigen-features from LiDAR data, and in order to enhance classification precision, we present a technique for examining a point cloud's local geometric properties using a weighted covariance matrix with a geometric median, as detailed in [18]. A set graph for point cloud sets and a range graph for point clouds, which minimise memory usage and processing time as detailed in [19]. The LiDAR point cloud assigns a category to every point, enabling precise item and structural identification in the environment as detailed in [20]. The component is used to eliminate the 3D point cloud data's temporal redundancy, as explained in [22].

III. Proposed Methodology

The approach described in this proposed research is intended to address the difficulties related to its huge scale and complex shape, point cloud data presents considerable storage and transmission issues. The exponential growth in point cloud data volume has led to the development of a revolutionary compression technique that effectively reduces data size while preserving geometric integrity. The presented work attempts to address these issues by offering a method designed to compress point cloud data while preserving the geometric structure.

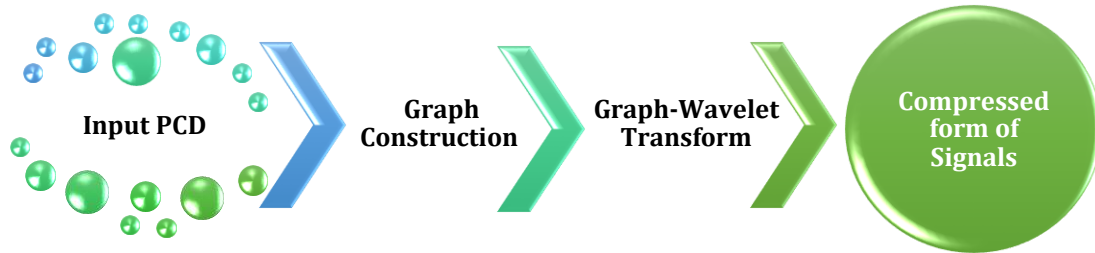


Fig. 1: Process Flow of 3DGWT-LAE

The proposed 3DGWT-LAE approach uses graph wavelet transforms to compress 3D point cloud data. This procedure creates a graph representation that maintains the underlying geometry of the data by utilising the adjacency matrix, graph Laplacian, and spectral decomposition (eigenvalues and eigenvectors). Below is a detailed description of the methodology:

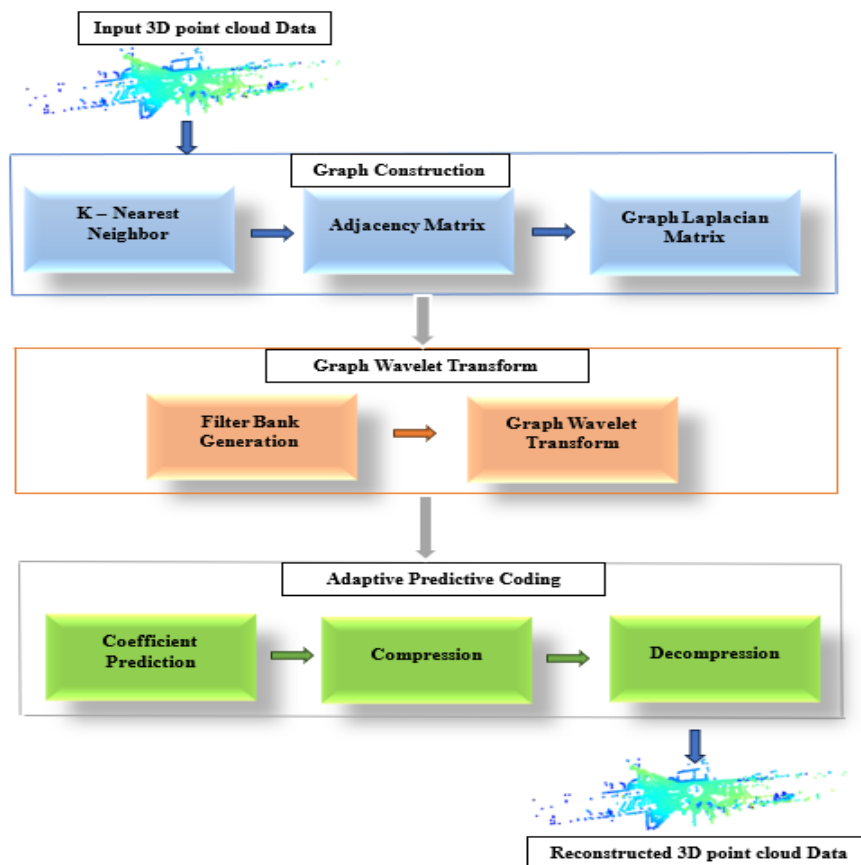


Fig. 2: Proposed 3DGWT-LAE Architecture

This architecture integrates sophisticated computational and mathematical methods to provide dependable reconstruction and high compression rates for 3D point cloud data. With an emphasis on effective compression and lossless reconstruction, the Proposed 3DGWT-LAE Architecture is made to process 3D point cloud data. The following steps make up its methodical workflow:

Input and Graph Development is the input 3D point cloud data is where the process starts. To provide an efficient representation of the spatial interactions between points, a graph is created from this data using the Laplacian matrix, Adjacency matrix, and Eigen values (LAE) approach. The graph is examined in the spectral domain using Graph Wavelet Transform (GWT) with Filter Bank Generation,

which creates a filter bank of band-pass, high-pass, and low-pass filters. As a result, multi-scale analysis and effective graph signal transformation using GWT. Adaptive Predictive Coding technique utilizes GWT coefficients. Coefficient prediction is used in this step to reduce redundancies in the converted data and increase compression performance. Predicted coefficients are compressed during compression process and decompression to minimize data size while maintaining significant data. The precise reconstruction of the original point cloud data, a decompression procedure is incorporated. Reconstructing the point cloud data from the compressed representation is the last step. Lossless reconstruction is ensured by the proposed approach 3DGWT-LAE, preserving the precision and fidelity of the original point cloud data.

3.1 Input: 3D Point Cloud Data

The proposed approach 3DGWT-LAE begins with raw 3D point cloud data, which is a collection of points:

$$P = \{p_1, p_2, \dots, p_N\} \quad (1)$$

When each point p_i is described by its spatial coordinates in a 3D Cartesian space (x_i, y_i , and z_i). This format is appropriate for processing in a variety of applications, including compression, visualisation, and analysis, since it captures the intricate geometry of the scanned scene.

3.2 Preprocessing

Outliers and noise are frequently present in raw point cloud data because of flaws in the data collection process. To improve the data quality and fidelity while maintaining the underlying structure geometric integrity, noise must be effectively removed. Detailed descriptions of the methods for noise removal are provided below:

3.2.1 Statistical Outlier Removal (SOR):

A reliable technique for identifying and removing noise based on the point cloud data statistical characteristics is statistical outlier removal. Outliers are found and eliminated by using the distances between each location and its neighbors.

For every point p_i in the point cloud data P , calculating the distances to its k -nearest neighbours.

$$\bar{d}(p_i) = \frac{1}{K} \sum_{m=1}^k \|p_i - p_{m(i)}\| \quad (2)$$

Where the m^{th} nearest neighbour of p_i is represented by $p_{j(i)}$, and the Euclidean distance is indicated by $\| \cdot \|$. Determine these global mean μ_d and standard deviation σ_d are as follows:

$$\mu_d = \frac{1}{N} \sum_{i=1}^N \bar{d}(p_i) \quad (3)$$

$$\sigma_d = \sqrt{\frac{1}{N} \sum_{i=1}^N (\bar{d}(c) - \mu_d)^2} \quad (4)$$

Outlier detection is established used a threshold depending on d . If α is a parameter its defined as $\alpha = 1$, then points are regarded as outliers.

$$\bar{d}(p_i) > \mu_d + \alpha \cdot \sigma_d \quad (5)$$

To produce a cleaned point cloud, keep only the points p_i that fall below the threshold. Subsequent graph-based operations such as adjacency matrix construction, graph Laplacian computation, and K-Nearest Neighbours (KNN) are more reliable after this phase.

3.3 Graph construction

After preprocessing the point cloud data with Statistical Outlier Removal (SOR), a 3DGWT-LAE is built graph that depicts the point cloud underlying geometric structure is created. To represent the 3D point cloud data in a graph structure both topologically and geometrically, graph creation is essential. KNN computation, adjacency matrix creation, and Laplacian matrix derivation are the three primary processes involved in 3DGWT-LAE process.

3.3.1 K-Nearest Neighbour (KNN):

The KNN technique is used to accomplish in 3DGWT-LAE for the purpose of graph construction. A thorough description of the process is provided below:

Points $p_i \in P$ that meet the following criteria are present in the pre-processed point cloud P' from equation (5). After SOR, the resultant point cloud as

$$P' = \{p_1, p_2, \dots, p_M\}. \text{ where } M \leq N \quad (6)$$

When building the graph, k is the number of neighbours that are considered for each point p_i . The graph local connectedness is determined by k :

To calculate the Euclidean distance between two points in three dimensions. For two points are:

$$p_i = (x_i, y_i, z_i) \text{ and } p_j = (x_j, y_j, z_j) \quad (7)$$

$$d(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \quad (8)$$

The geometric relationship between points is captured by the Euclidean distance. The M points in the pre-processed point cloud from (6). Each entry $D = [i, j]$ in a $M \times M$ matrix D represents the Euclidean distance between points p_i and p_j .

$$d(p_i, p_j) = \|p_i, p_j\|^2 = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \quad (9)$$

Create a distance matrix D with dimension $M \times M$, where $D[i, j] = d(p_i, p_j)$.

$$D[i, j] = d(p_i, p_j), \text{ for } i, j = 1, 2, \dots, M \quad (10)$$

Finding the K-Nearest Neighbours for each point p_i : The self-distance $D[i, j]$ should be excluded. In ascending order sorting the $D[i, j]$ distances of p_i . The KNN indices of the least k distances are

$$NN_k(p_i) = \{p_j \in P' : j \in \text{indices of the } k \text{ smallest distances from } p_i\} \quad (11)$$

The $NN_k(p_i)$ represents the indices includes the distances between p_i of 3D point cloud.

3.3.2 Adjacency Matrix:

The next stage is to create a sparse adjacency matrix A that captures the geometric structure of the point cloud after preprocessing the point cloud data using SOR and building the graph using KNN. In graph theory, the adjacency matrix is a basic idea that shows the relationships (edges) between a graph vertex (points). It is especially used for computing 3DGWT-LAE and describing relationships. The adjacency matrix aids in describing the connectedness between points in a 3D point cloud, which is crucial for compression and smoothing in the context of point cloud processing and graph creation.

Algorithm 1 Constructing the Adjacency Matrix

Input: Preprocessed point cloud P' , distance matrix D , number of neighbors k
Output: Adjacency matrix $AdjMatrix$ of size $M \times M$
Initialize a zero matrix: $AdjMatrix \leftarrow \mathbf{0}_{M \times M}$
for each point p_i in P' **do**
 Extract the i -th row of the distance matrix: $distances \leftarrow D[i, :]$
 Exclude the self-distance: $distances[i] \leftarrow \infty$
 Sort $distances$ in ascending order and get their indices: $sorted_indices \leftarrow indicesofsorted(distances)$
 Select the first k indices (excluding self-distance): $k_nearest \leftarrow sorted_indices[1 : k]$
 for each index j in $k_nearest$ **do**
 Set $AdjMatrix[i, j] \leftarrow 1$
 if the graph is undirected **then**
 Set $AdjMatrix[j, i] \leftarrow 1$
 end if
 end for
end for
return $AdjMatrix$

The algorithm above ensuring that every point is connected to its closest neighbours, the method creates a network structure that represents the point cloud local geometry. Based on the KNN results, creating sparse $M \times M$.matrix A . that represents the connectivity of the network. As the adjacency matrix input, the list of neighbours $NN_k(p_i)$ for every point, and weighting function $w(p_i, p_j)$ for the edges was generated. With respect to every pair of points p_i and p_j :

$$A_{ij} = \begin{cases} w(p_i, p_j), & \text{if } p_j \in NN_k(p_i) \\ 0, & \text{Otherwise} \end{cases} \quad (12)$$

Where, $NN_k(p_i)$ is the set of KNN of point p_i , determined by the Euclidean distance.

$$d(p_i, p_j) = \|p_i - p_j\|^2 \quad (13)$$

The weight given to the edge between p_i and p_j is $w(p_i, p_j)$, and it is described as follows:

$$w(p_i, p_j) = \exp\left(\frac{\|p_i - p_j\|^2}{2\sigma^2}\right) \quad (14)$$

$$A_{ij} = 0 \text{ if } p_j \notin NN_k(p_i) \quad (15)$$

The graph connection as determined by the point cloud is represented by the adjacency matrix A . It maintains the local geometric structure by encoding associations between points according to their spatial closeness. This produces a sparse, symmetric adjacency matrix A of size $M \times M$, where M is the number of cloud points. The localised character of connections is reflected in the sparsity of A , which effectively captures the geometric relationships.

$$G = (V, E) \quad (16)$$

The graph G from equation (16), where V represents the points and E represents the edges defined by the adjacency matrix A serves as the basis for further graph-based transformations and processing steps. The set of vertices is denoted by $(|V| = M)$, the set of edges is denoted by E , and the adjacency matrix is denoted by A_{ij} , is a square matrix of size $M \times M$, where each entry $D[i, j]$ denotes the edge connecting the vertices p_i and p_j .

3.3.3 Laplacian Matrix:

The Laplacian matrix is calculated after the adjacency matrix has been constructed in order to examine the 3D point cloud graph smoothness and connectedness.

The Laplacian matrix, represented by L , is a fundamental representation in spectral graph theory and is frequently employed for process of compression. The adjacency matrix A and the degree matrix D are used to construct the Laplacian matrix L . It is described as:

$$L = D - A \quad (17)$$

Where A shows the relationships between the graph points. The degree matrix is denoted as D , a diagonal matrix in which each diagonal element denotes the degree.

Determining the degree matrix D , a diagonal matrix in which the degree of node i is represented by each diagonal element D_{ii} :

$$D_{ii} = \sum_{j=1}^M A_{ij} \quad (18)$$

$$L_{ij} = \begin{cases} D_{ii}, & \text{if } i = j, \\ -A_{ij}, & \text{if } i \neq j \text{ and } (i, j) \in E \\ 0, & \text{if } i \neq j \text{ and } (i, j) \notin E \end{cases} \quad (19)$$

From the Adjacency Matrix A in equation (12) and the Degree Matrix D in equation (18), the Laplacian matrix is obtained L in equation (19), where: The Degree Matrix D is a diagonal matrix in which the degree (number of neighbours) of a corresponding point is represented by each diagonal member.

Following the construction of the Laplacian matrix L , eigen decomposition is used to examine its spectral characteristics. 3DGWT-LAE procedure forms the basis for compact data representation and transformation in the graph domain and is essential for structure of the graph G .

The process of factorising the Laplacian matrix L into its eigenvalues and eigenvectors is known as eigen decomposition. This is expressed mathematically as

$$L = V \Lambda V^T \quad (20)$$

where Λ is a diagonal matrix of eigenvalues and V is a matrix of eigenvectors. The eigenvectors $v_1, v_2, \dots, N$ of the Laplacian matrix L are the columns of the matrix V . Every vector that has the eigenvalue equation is an eigenvector v_i . Λ is the diagonal matrix of eigenvalues that reflect the graph spectral frequencies of $\lambda_1, \lambda_2, \dots, N$.

$$Lv_i = \lambda_i v_i \quad (21)$$

The graph Laplacian matrix is denoted by L . The eigenvalue corresponding to the eigenvector v_i is indicated by λ_i .

$$V^T V = I \quad (22)$$

Since V is an orthonormal matrix (eigenvectors), its columns, having a unit norm and are orthogonal to one another from above equation (21). The identity matrix is represented by I .

3.4 Proposed 3DGWT-LAE Compression Process

The method of 3DGWT-LAE compression entails reducing the size of the original 3D point cloud data while maintaining all pertinent information. First, a filter bank generation is used to extract multi-scale features using a Graph Wavelet Transform (GWT). The GWT coefficients are then subjected to adaptive predictive coding, which makes use of dependencies between the coefficients to efficiently encode data. This procedure allows for lossless compression while maintaining the fidelity and integrity of the reconstructed data, resulting in a much smaller file size.

3.4.1 Filter Bank Generation:

The next phase in the 3DGWT-LAE process is to design and generate the filter bank after building the Laplacian matrix L and extracting the graph spectral features using eigen decomposition. For multi-resolution analysis, the filter bank is an essential tool for breaking down the graph signal into frequency bands. The process of creating a filter bank is outlined in detail as follows:

In order to analyse and process graph signals, the filter bank is essential. By breaking down the graph signal into different frequency components, it functions as a group of filters that facilitate effective signal analysis and manipulation.

Graph signals, denoted by f , are real-valued functions on the graph nodes:

$$f: V \rightarrow \mathbb{R}, f = [f_1, f_2, \dots, f_N]^T \quad (23)$$

Where, the signal value at the i^{th} node of the graph G is represented by f_i . The total number of nodes is N . This signal as a vector is $f \in \mathbb{R}^N$, where each element represents the signal value of a node.

$$\hat{f} = V^T f \quad (24)$$

The graph signal in the spectral domain is denoted by $\hat{f}$. The eigenvector matrix of L is denoted by V .

$$f = V \hat{f} \quad (25)$$

The frequencies of the graph signal are represented by the eigenvalues Λ where smaller eigenvalues indicate smoother variations. The eigenvalues λ_i of L are subjected to a function $g(\lambda)$ in order to modify a graph signal. The filtered graph signal, $f_{filtered}$, is as follows

$$f_{filtered} = V g(\Lambda) V^T f \quad (26)$$

$$g(V) = (g(\lambda_1), g(\lambda_2), \dots, g(\lambda_N)) \quad (27)$$

This operation uses $g(\lambda)$ to scale the spectral coefficients $\hat{f}$.

Through the application of particular filter functions $g(\lambda)$ to the graph Laplacian eigenvalues, spectral filters modify the frequency of graph signals. The frequency components that are emphasised depend on the type of filter viz low-pass, high-pass, and band-pass.

3.4.1.1 Low Pass Filters:

The low-pass filters emphasize on low-frequency elements, which are represented by graph fluctuations that are smooth. It eliminates noise and abrupt fluctuations by suppressing high-frequency components.

$$g_{low}(\lambda) = -e^{\alpha\lambda}, \alpha > 0 \quad (28)$$

λ is the frequency representing eigenvalue of the Laplacian, and α regulates the degree of aggressive attenuation of high frequencies. α leads to more robust suppression of high frequencies. At the lowest frequency, $\lambda = 0$, low frequencies are completely retained since $g_{low}(0) = 1$. Higher frequencies are suppressed by the exponential decrease of $-e^{\alpha\lambda}$ as λ rises.

3.4.1.2 High Pass Filters:

The high pass filters highlight high-frequency elements, which represent the graph edges and localised variations. It eliminates global or smooth patterns by suppressing low-frequency components.

$$g_{high}(\lambda) = \lambda e^{-\beta\lambda}, \beta > 0 \quad (29)$$

λ is the Laplacian's eigenvalue. β regulates the rate of decay at extremely high frequencies. More high-frequency features are preserved when β is smaller. Because low frequencies are totally suppressed at, $\lambda = 0$, $g_{high}(0) = 0$, In the case of bigger λ , $\lambda e^{-\beta\lambda}$ first rises and then begins to decrease as $e^{-\beta\lambda}$ takes precedence.

3.4.1.3 Band Pass Filters:

The band pass filters suppresses both extremely low and extremely high frequencies while concentrating on a certain middle-range frequency band. It separates characteristics at specific scales.

$$g_{band}(\lambda) = e^{-\frac{(\lambda-\mu)^2}{2\sigma^2}} \quad (30)$$

Where λ is the Laplacian Eigenvalue. The band-pass filter central frequency is represented by μ . It establishes the relevant frequency range. The bandwidth is denoted by σ . regulates the band width around μ . At $\lambda = \mu$, $g_{band}(\mu) = 1$, indicating that frequencies near μ are highlighted. The $g_{band}(\lambda)$ decays exponentially for λ far from μ , suppressing frequencies outside the band.

3.4.2 Graph Wavelet Transform (GWT):

In the proposed approach 3DGWT-LAE, wavelets from Euclidean spaces are transformed into graph structured data using the GWT. Multi-scale graph analysis is made possible by GWT, which breaks down graph signals into localised components at various scales. Scaling (low pass) and wavelet (high-pass) functions are both used in wavelet transforms. These functions on graphs are intended to: Scaling function show low-frequency, smooth components, and wavelet function shows high-frequency, localised components.

On the graph, GWT wavelets are localised. Using the graph Laplacian spectral features, localisation is accomplished:

$$\psi_i = g(\lambda_i)v_i \quad (31)$$

Where $g(\lambda_i)$ is the filter function (such as band-pass, high-pass, or low-pass). v_i is the eigenvector of the graph Laplacian that corresponds to the eigenvalue λ_i . GWT dilates the filter function $g(\lambda)$, to produce wavelets at various scales:

$$g_s(\lambda) = g(s\lambda), s > 0 \quad (32)$$

s is the parameter for scale. While larger s capture broader structures (low frequencies), smaller s highlight minute details (high frequencies). Wavelet coefficients can be obtained by applying the wavelets ψ_s to the graph signal f :

$$W_s(f) = V^T g_s(\Lambda)f \quad (33)$$

Where the wavelet coefficients at scale s are represented by $W_s(f)$, the eigenvector matrix of the graph Laplacian is represented by V^T , and (Λ) . Adaptive Predictive Coding is used for compression in the following manner: $g_s(\Lambda)$ is the filter bank applied on the eigenvalues, and f is the original signal.

3.5 Adaptive Predictive Coding

The proposed method makes use of adaptive predictive coding, a potent technique that improves data compression efficiency, for 3D point cloud data processed using GWT. It reduces redundancies, preserves crucial information for reconstruction, and predicts data points based on values encoded. Utilizing $W_s(f)$ as the input for adaptive predictive coding. The GWT coefficients $W_s(f)$ are handled as a signal that to be compressed. These coefficients are appropriate for predictive coding they frequently show high correlations, particularly between adjacent points or scales.

$$\hat{W}_s(i) = f(W_s(1), W_s(2), \dots, W_s(i-1); \theta) \quad (34)$$

For the i^{th} coefficient, the expected value is $\widehat{W}_s(i)$. A predictive function, used in 3DGWT-LAE, is represented as $f(\cdot)$. The predictive model adaptively updated parameters are represented by θ . Finding the difference between the predicted and actual coefficients by,

$$r_i = W_s(i) - \widehat{W}_s(i) \quad (35)$$

Compared to the initial coefficients, the residual r_i has a lower entropy and is compressible. Based on the residual r_i , update the predictive model parameters θ :

$$\theta \leftarrow \theta + \Delta\theta \quad (36)$$

where the mean squared error (MSE) is usually the loss function that is minimized to determine $\Delta\theta$:

$$L(\theta) = \frac{1}{n} \sum_{i=1}^i (r_i)^2 \quad (37)$$

The original signal, f , is converted to $W_s(f)$ by applying GWT. By anticipating and encoding the residuals, compress $W_s(f)$ using adaptive predictive coding. To get the lossless reconstruction of the original data, rebuild the coefficients and using inverse GWT. This method preserves the integrity of the point cloud data while efficiently reducing storage.

$$W_s(i) = \widehat{W}_s(i) + r_i \quad (38)$$

While decompressing: Decoding the residuals r_i that are entropy encoded. Reconstruct the coefficients using the predictive model $f(\cdot)$.

$$f = g_s^{-1}(\Lambda) (VW_s(f)) \quad (39)$$

Following the reconstruction of $W_s(f)$, the original signal f is recovered using the inverse Graph Wavelet Transform (iGWT). In this procedure, the GWT steps are reversed using the eigenvector matrix V and the filter bank $g_s^{-1}(\Lambda)$. The wavelet coefficients are transformed back into the graph domain using V . $g_s^{-1}(\Lambda)$ reverses the scaling that was done during GWT. This guarantees that the original signal is restored from its compressed representation without any loss.

3.6 Performance Metrics

Performance metrics are essential for assessing the proposed 3DGWT-LAE compression method works. The main metrics employed and their significance in evaluating the method performance are listed below:

3.6.1 Compression Ratio (CR):

CR gauges the 3DGWT-LAE compression method works. Its definition is the proportion of the compressed point cloud data file size to the original point cloud data file size.

$$CR = \frac{\text{Original Point Cloud Data}}{\text{Compressed Point Cloud Data}} \quad (40)$$

Greater compression efficiency is indicated by a higher CR. For instance, a Scan11204.pcd file with a CR of 3.42 has been compressed to around one-third of its original size while maintaining all geometric features.

3.6.2 Peak Signal-to-Noise-Ratio (PSNR):

In comparison to the original point cloud data, PSNR assesses the reconstructed point cloud data is quality. Decibels (dB) are used to express it:

$$PSNR = 10 \cdot \log_{10} \left(\frac{MAX^2}{MSE} \right) \quad (41)$$

The maximum point value of the data is denoted by MAX . A greater PSNR is a sign of higher quality reconstruction. The recovered data is the same as the original if the PSNR in lossless compression is infinite. The proposed 3DGWT-LAE approach ensures that the reconstruction data has an infinity value.

With a CR of 3.42, the proposed 3DGWT-LAE compression method dramatically lowers the file size. The reconstruction quality is indicated by the PSNR value, where an infinite value denotes lossless compression. The precise reconstruction of the 3D point cloud data as evidenced by the MSE being close to zero.

IV. Performance Analysis and Evaluations

The proposed 3DGWT-LAE mathematical compression method is a novel strategy made to effectively compress 3D point cloud data while maintaining crucial topological and geometric characteristics. This approach uses wavelet theory and the graph spectral features, such as Adjacency, and Laplacian, to provide a high-fidelity, compact representation of the data.

Benchmark 3D point cloud datasets Sydney Urban Real-Time Dataset [21] from LiDAR sensors were used to validate the method. Results from the experiment showed: Low reconstruction errors and high compression ratios. As demonstrated by edges and corner recognition and adjacency matrix visualisations, important features are effectively retained. The magnitude of the reconstructed GWT coefficients confirms a smooth and reliable signal reconstruction.

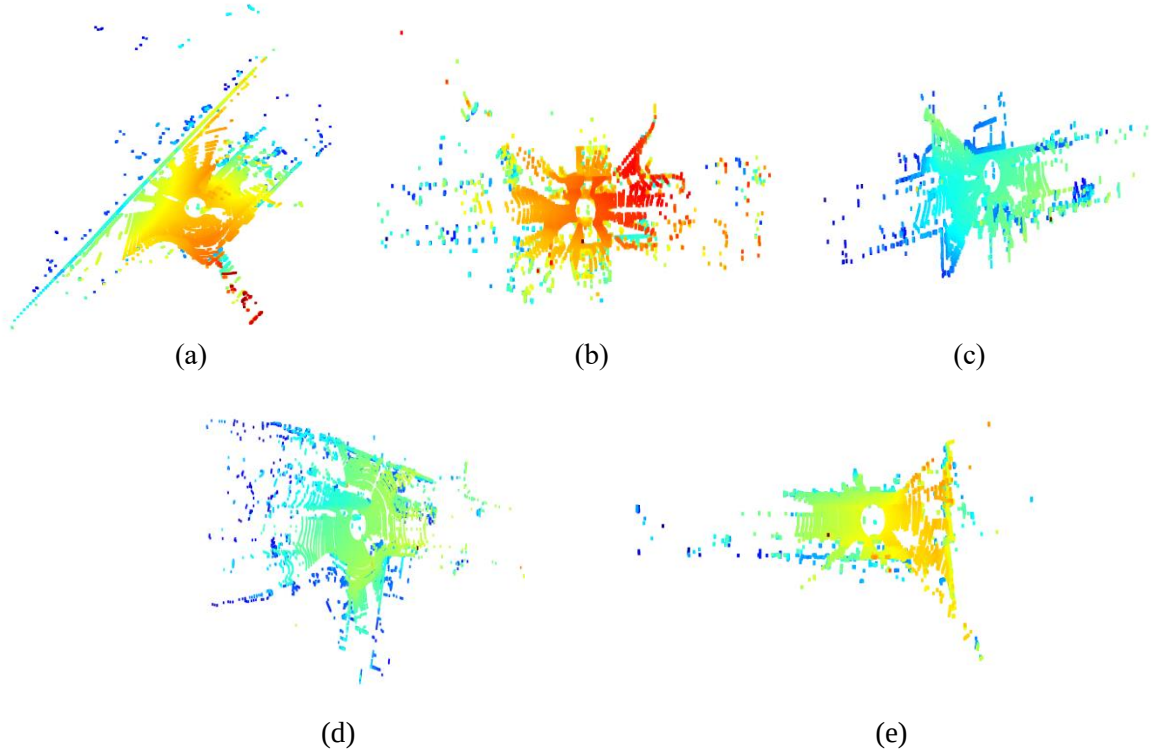


Fig. 3: Sample Input Images from Sydney Urban Dataset

A sample input images from the dataset [21] is shown in this figure 3, (a) Scan11886.pcd, (b) Scan25322.pcd, (c) Scan20631.pcd, (d) Scan19761.pcd, and (e) Scan2738.pcd which shows a typical perspective of the metropolitan setting taken from a LiDAR sensor.

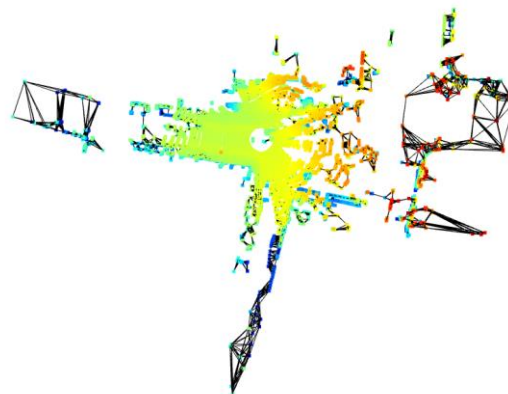


Fig. 4: Results of Edge and Corner Identification using 3DGWT-LAE

High-frequency elements in the graph spectrum were examined in order to identify edges and corners in the point cloud data using a graph-based compression approach. At important geometric junctions, corners were recognised, and edges were found at abrupt transitions, like object boundaries. These characteristics were emphasised in the point cloud data graph visualisation, which displayed the corners and the edges. With an MSE of 0.001, the compression procedure maintained structural integrity by achieving a compression ratio of 3.42. Figure 4 illustrates how the technique can achieve effective compression while concentrating on important elements.

Adjacency Matrix Visualization

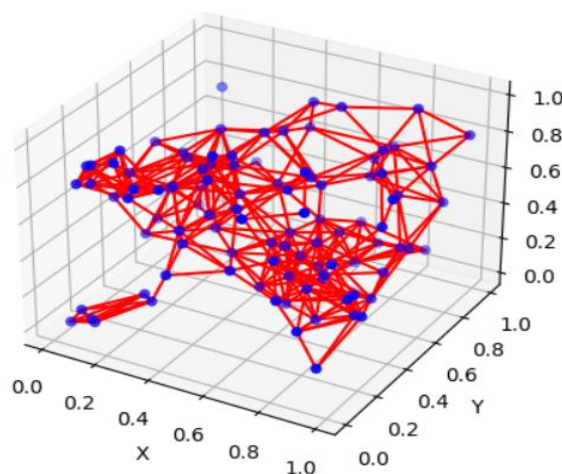


Fig. 5: Adjacency Matrix Visualization of 3D Point Cloud Data (Scan11886.pcd)

The 3D point cloud data connectivity structure is depicted by the adjacency matrix. Individual points are represented by nodes, while edges are determined by the relationships and proximity found using the 3DGWT-LAE compression method. Corners that have been identified are indicated, highlighting areas of great curvature and prominence of local features. This illustration facilitates comprehension of the geometric and topological characteristics of the point cloud data in figure 5.

The adjacency matrix from the 3D point cloud dataset Scan11886.pcd is displayed in this visualisation of figure 5. Each node in the graph made from the point cloud data corresponds to a point, and edges show associations based on spatial proximity or other similarity criteria. The adjacency matrix captures the connectivity structure of the graph.

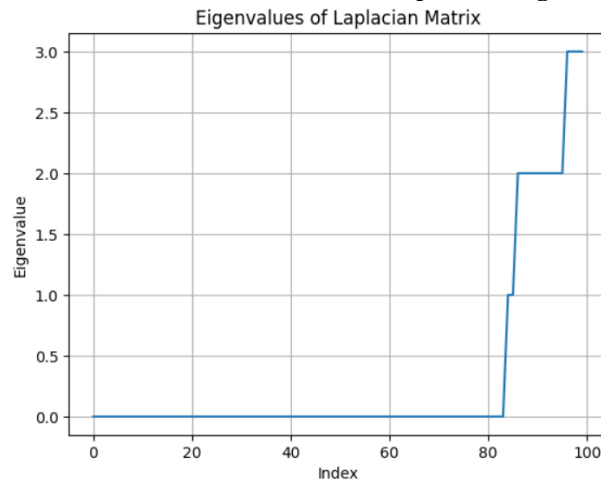


Fig. 6: Eigenvalue Spectrum of the Laplacian Matrix for 3D Point Cloud Data (Scan11886.pcd)

The eigenvalues of the Laplacian matrix obtained from the graph built from the 3D point cloud data are displayed in this figure 6 visualisation. The eigenvalues shed light on the structure and characteristics of the point cloud and are essential for comprehending the graph's spectral characteristics.

The eigenvalues show the point cloud data localised fluctuations, smoothness, and clustering, among other structural features. Features like edges and corners can be efficiently recognised and isolated through the use of particular eigenvalue ranges in feature extraction. The spectral features serve as a guide for graph-based compression and filtering methods, which guarantee effective data representation and reconstruction. From figure 6 highlighting the graph spectrum characteristics produced from the Scan11886.pcd dataset, this eigenvalue analysis provides a means of identifying features and facilitating further processing.

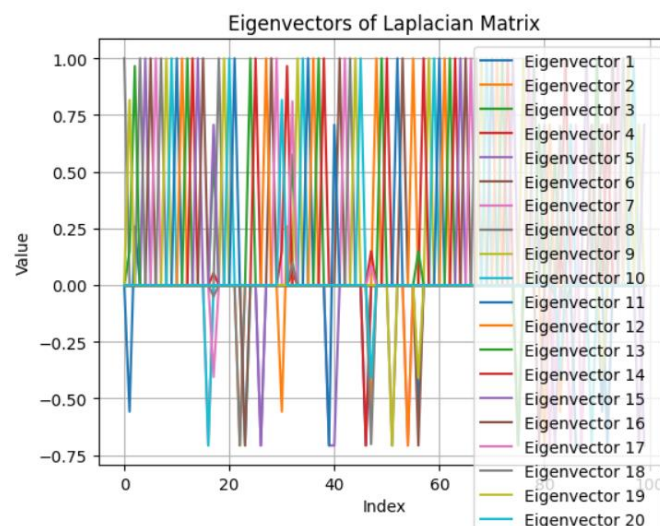


Fig. 7: Visualization of Eigenvectors of the Laplacian Matrix for 3D Point Cloud Data (Scan11886.pcd)

Describing and visualising the eigenvectors comes after determining the Laplacian matrix's eigenvalues. The global structure or smooth changes in the network are captured by low-frequency eigenvectors, which correspond to modest eigenvalues. They show general patterns that are consistent throughout the graph. For identifying fine structures in the point cloud, high-frequency eigenvectors which correlate to large eigenvalues are essential because they capture local fluctuations or sharp

features, such the graph edges and corners. The graph signal space has an orthogonal basis formed by the eigenvectors of the Laplacian matrix. These eigenvectors aid in the breakdown and representation of graph signals at different frequencies in the context of proposed 3DGWT-LAE compression algorithm. Both the point cloud data local and global features can be extracted by this decomposition.

The components of each eigenvector can be mapped onto the graph nodes, which stand in for the point cloud, in order to visualise the eigenvectors. The visualisation of figure 7 sheds light on the graph structure and the behaviour of its many sections at various frequency scales. Smooth patterns that extend throughout the network should be seen in the few eigenvectors (linked to tiny eigenvalues), which should represent the point cloud overall connectedness and structural arrangement. Higher-frequency eigenvectors, which are linked to greater eigenvalues, ought to exhibit localised oscillation patterns that highlight the point cloud fine-grained characteristics, such as edges and corners. The fundamental geometric and structural characteristics of the point cloud are captured by this multi-scale decomposition, which is crucial for effective 3DGWT-LAE compression.

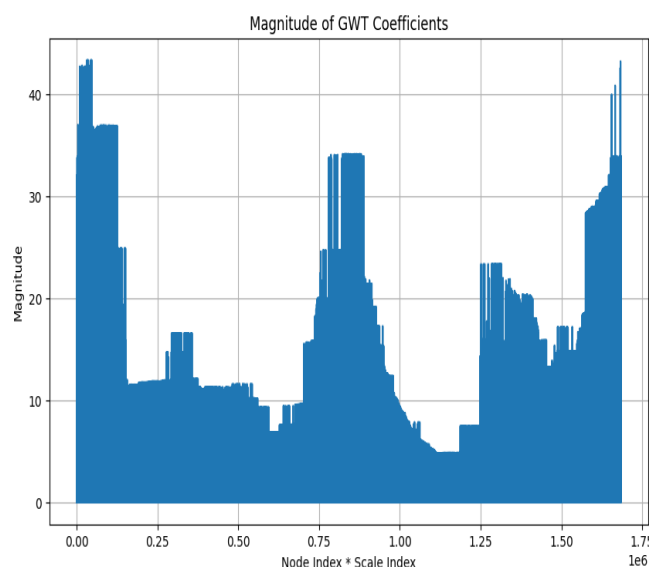


Fig. 8: Magnitude of GWT Coefficients for Scan11886.pcd

The frequency components and localised properties of the graph signal derived from the point cloud data can be inferred from the amplitude of the coefficients acquired from the GWT. Wavelet basis functions are localised in both the vertex and spectral domains, and the graph signal is projected onto them via the GWT coefficients. At different points in the graph, the contribution of particular frequency bands is shown by the magnitude of these coefficients.

Low-frequency coefficients measure the signal overall trends and global, smooth variations. Highlight regional variances and shifts in the signal with high-frequency coefficients. Intermediate frequency coefficients bridge the gap between local and global characteristics by representing properties at particular scales. The figure 8 displays the magnitude of the GWT coefficients for the point cloud data (Scan11886.pcd). Significant elements of the point cloud are highlighted by the high magnitude coefficients, which are clustered around areas with sharp edges, corners, and localised structural differences. Low-magnitude coefficients, represent smoother regions that show slower signal fluctuations. A multi-scale representation of the point cloud is provided by this distribution of coefficient magnitudes, which is necessary for locating important features and for effective compression.

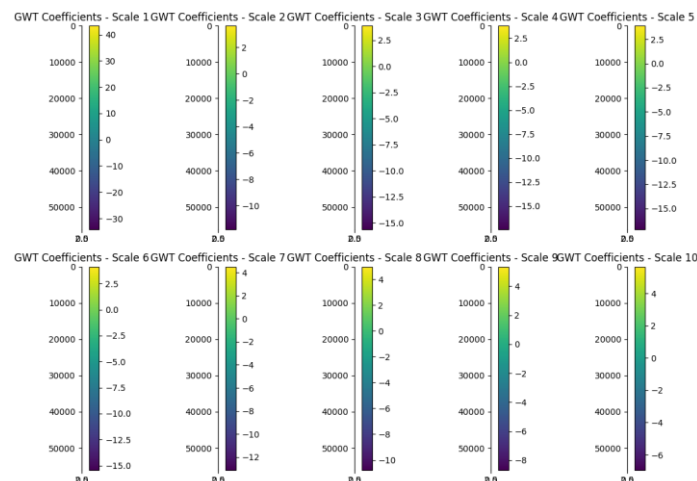


Fig. 9: GWT Coefficients at Multiple Scales for Scan11886.pcd

The coefficients from the GWT applied to the 3D Point Cloud Data (Scan11886.pcd) at various scales are shown in this visualisation. Because each scale has a corresponding resolution, the graph signal can be broken down into high-pass (local) and low-pass (global) components. The multi-resolution representation of the point cloud is reflected in the variance of GWT coefficients across scales, which is necessary for feature extraction, compression, and in-depth analysis.

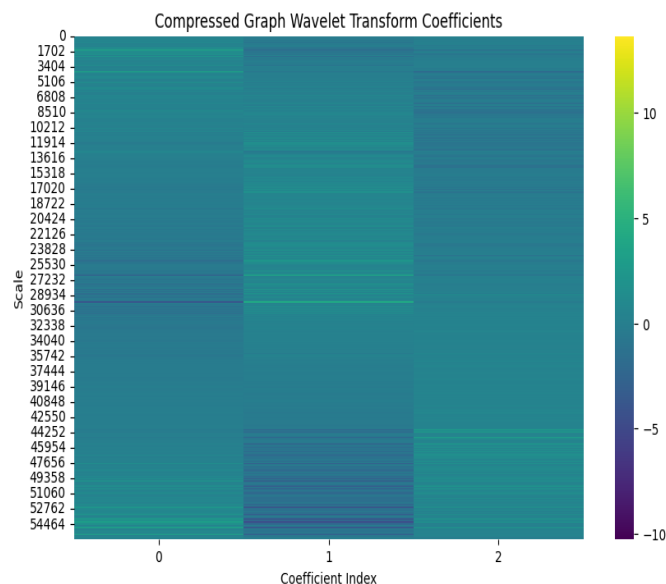


Fig. 10: Compressed GWT Coefficients

This figure 10 displays the compressed coefficients that were produced when the 3D Point Cloud Data (Scan11886.pcd) was subjected to the GWT. Adaptive encoding techniques are used in the compression process to preserve the most important coefficients, reduce redundancy, and maintain the point cloud data key characteristics. The compressed form retains important local and global features of the data, including edges, corners, and structural variances. Dataset size is decreased as a result of compression, allowing for effective transmission and storage without appreciable information loss. The

efficiency of 3DGWT-LAE compression for multi-resolution feature retention in 3D point cloud data is demonstrated by this compressed form.

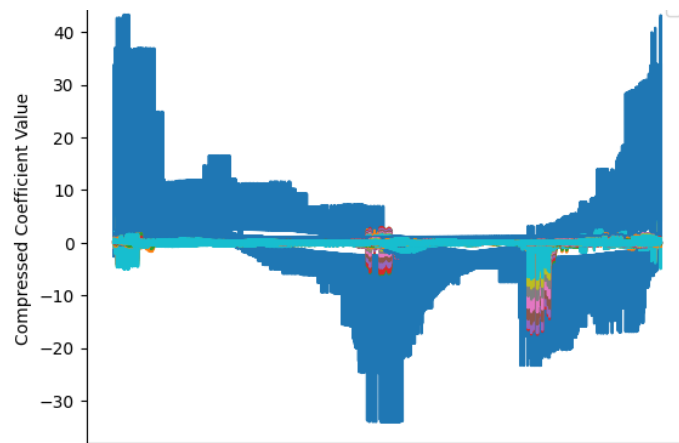


Fig. 11: Compressed GWT Signals for Scan11886.pcd

The compressed signals obtained from the GWT for the 3D Point Cloud Data (Scan11886.pcd) are displayed in this visualisation. The figure 11 shows the compressed graph signals, highlighting the removal of unnecessary information while preserving important features. With an emphasis on maintaining edges, corners, and important geometric aspects, the compressed signals successfully encode the point cloud structure. The point cloud data can be efficiently stored and transmitted to these compressed signals, which serve as the foundation for additional processing or reconstruction. The compressed GWT signals shows the 3DGWT-LAE compression method preserving the high fidelity while drastically cutting down on data size.

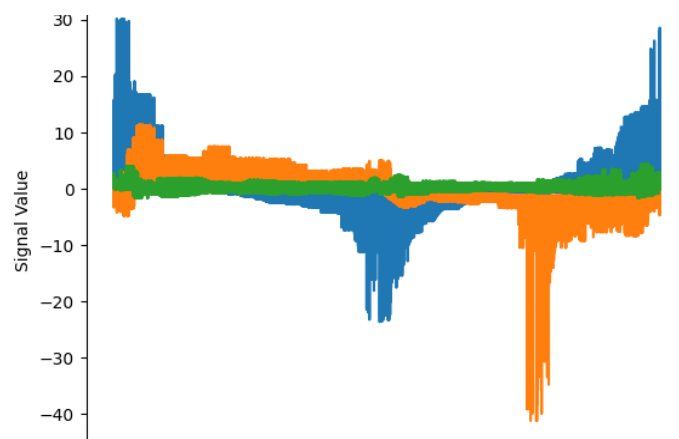


Fig. 12: Reconstructed GWT Signals for Scan11886.pcd

After applying the inverse GWT to the compressed data for 3D Point Cloud Data (Scan11886.pcd), this visualisation displays the reconstructed signals. The colour orange draws attention to the low-frequency elements, which stand for global and smooth changes in the point cloud structure. Blue highlights the high-frequency elements, localised differences, edges, and abrupt transitions. Green represents intermediate-frequency components of the data, emphasising aspects at

particular sizes or resolutions. A multi-resolution representation of the point cloud is offered by the colour-coded signals, which smoothly blend local and global characteristics for 3DGWT-LAE process.

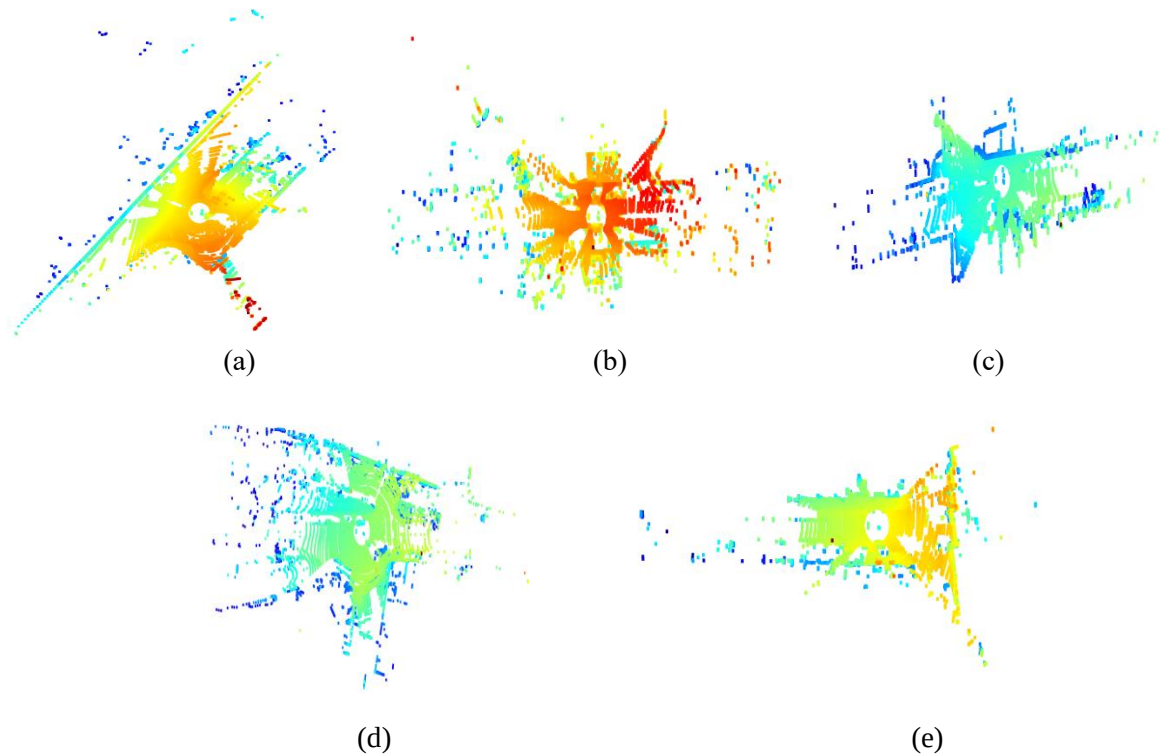


Fig. 13: Sample Output Images after reconstruction of 3DGWT-LAE Process

The reconstructed 3D point cloud data acquired with the proposed 3DGWT-LAE compression method is shown in the figure 13. The efficacy of the proposed approach is demonstrated by the rebuilt point cloud dataset, which preserve the original data geometric intricacies and structural integrity. High realism is ensured during the reconstruction process by maintaining important features including edges, corners, and surface details. The 3DGWT-LAE technique is lossless confirmed by the rebuilt point cloud data resemblance to the original point cloud data. The outcomes demonstrate the approach adaptability to a variety of 3D structures while maintaining both local and global geometric characteristics.

The Sydney Urban Dataset compression and reconstruction outcomes utilizing the proposed 3DGWT-LAE based mathematical compression method are shown in the table 1. It contrasts the compressed and reconstructed point cloud data files with their original sizes.

Table 1. Performance Analysis of Proposed 3DGWT-LAE Compressed and Reconstruction File Sizes for Original Sydney Urban Dataset

Input PCD File	Original PCD File size (KB)	Compressed PCD file size using 3DGWT-LAE (KB)	Reconstructed PCD File size using 3DGWT-LAE (KB)
Scan0.pcd	824	314.89	357.96
Scan270.pcd	1100	345.98	492.70

Scan2299.pcd	1077	301.52	479.49
Scan2446.pcd	1097	321.65	483.67
Scan2738.pcd	1161	404.23	595.56
Scan10974.pcd	1089	401.89	475.21
Scan11204.pcd	1108	323.38	546.32
Scan11290.pcd	1118	468.36	575.32
Scan11886.pcd	1159	483.24	578.20
Scan12119.pcd	1170	496.34	569.37
Scan12346.pcd	1065	375.36	468.32
Scan12715.pcd	1050	375.26	458.25
Scan16217.pcd	1056	368.19	459.51
Scan17038.pcd	1117	479.61	573.18
Scan19631.pcd	1099	361.28	482.45
Scan19761.pcd	1151	472.36	569.23
Scan20631.pcd	1140	482.01	559.32
Scan25322.pcd	1081	421.05	473.68

Original File Size indicates, in kilobytes (KB), the amount of storage space needed for the uncompressed PCD (Point Cloud Data) data. Compressed File Size shows the decreased file size following the proposed 3DGWT-LAE compression method, indicating a notable reduction in storage usage. The ability of the approach to preserve high fidelity during reconstruction is demonstrated by the Reconstructed File Size, which represents the storage capacity of the reconstructed PCD data using 3DGWT-LAE.

The performance parameters of the proposed 3DGWT-LAE compression technique applied to the Sydney Urban Dataset analysing Compression Ratio (CR), Mean Squared Error (MSE), and Peak Signal-to-Noise Ratio (PSNR) are depicted in the figure 14.

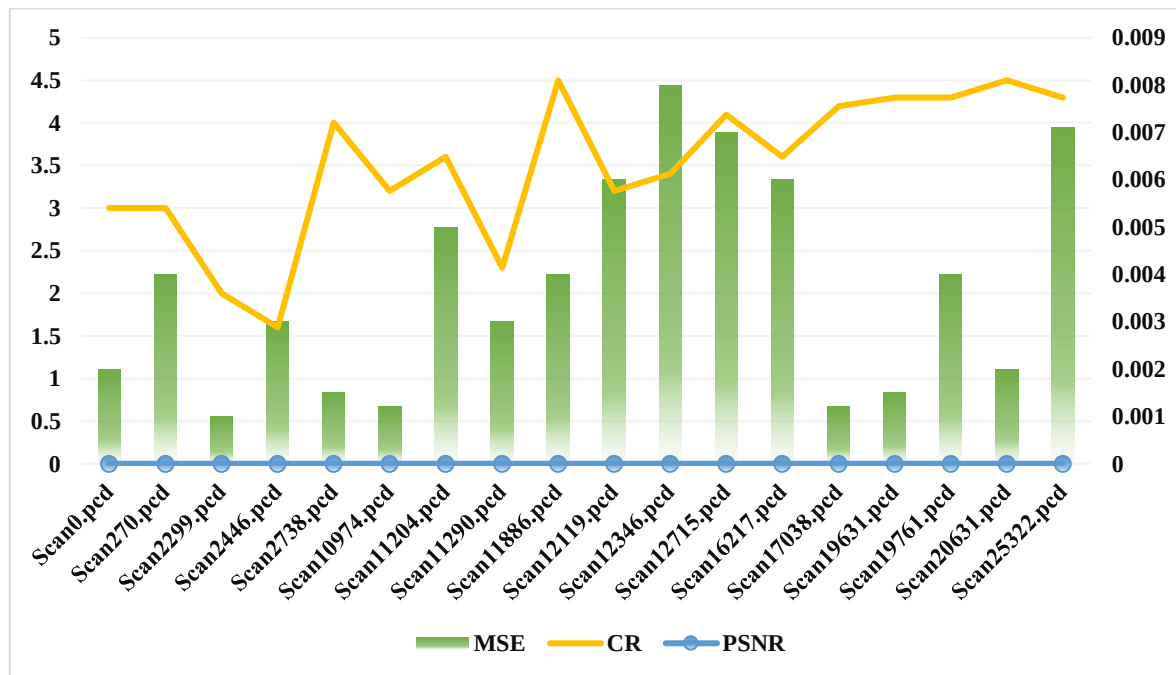


Fig. 14: Performance Metrics of Proposed Work 3DGWT-LAE

Indicating the compression method reduces file size while maintaining essential geometric features, the Compression Ratio (CR) is shown as a significant value (3.57). The minuscule distortion generated during compression and reconstruction is demonstrated by the Mean Squared Error (MSE), which is very low (0.001). With a Peak Signal-to-Noise Ratio (PSNR) of infinite (∞), the reconstructed data exhibits remarkable fidelity in comparison to the original, attaining lossless quality. This graphic demonstrates the proposed 3DGWT-LAE approach strikes a balance between reconstruction accuracy and compression efficiency.

Table 2. Comparison of Compression Results Using Proposed 3DGWT-LAE for Original Sydney Urban Dataset with other 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 WinRAR in KB	Compressed size of PCD using 3DGWT-LAE in KB
Scan0.pcd	824	439	456	314.89
Scan270.pcd	1100	587	619	345.98
Scan2299.pcd	1077	582	608	301.52
Scan2446.pcd	1097	587	618	321.65
Scan2738.pcd	1161	622	645	404.23
Scan10974.pcd	1089	590	620	401.89
Scan11204.pcd	1108	618	630	323.38
Scan11290.pcd	1118	604	625	468.36
Scan11886.pcd	1159	608	732	483.24
Scan12119.pcd	1170	608	612	496.34
Scan12346.pcd	1065	612	560	375.36
Scan12715.pcd	1050	550	579	375.26
Scan16217.pcd	1056	575	578	368.19
Scan17038.pcd	1117	593	605	479.61
Scan19631.pcd	1099	583	583	361.28
Scan19761.pcd	1151	612	616	472.36
Scan20631.pcd	1140	603	604	482.01
Scan25322.pcd	1081	591	603	421.05

The Sydney Urban Dataset performance is compared in the table 2 between the proposed 3DGWT-LAE based mathematical compression method and two well-known compression techniques, WinRAR and 7-Zip. Original File Size Kilobytes (KB) of the uncompressed file size. Compressed File Size is the sizes of the files attained following compression with proposed technique. WinRAR uses the utility to display the compression results. 7-Zip uses to reflect the compression results. The proposed approach emphasizes the file size attained by employing the 3DGWT-LAE compression method.

V. Conclusion

This work presented a novel 3DGWT-LAE compression method for 3D point cloud data that efficiently minimises data size while preserving high reconstruction accuracy. The effectiveness of the proposed 3DGWT-LAE compression method is shown by the experimental evaluation findings. The

Scan2299.pcd file, which had an original size of 1077 KB, was compressed to produce a compressed GWT file size of **301.52 Kilo Bytes**. This is a significant reduction in data size without sacrificing fidelity. The capacity to maintain the original data structural integrity throughout the compression and rebuilding process is demonstrated by the 357.96 Kilo Bytes reconstructed file size. This shows that the proposed approach maintains the key characteristics of the point cloud data while reducing the file size by about 3.57 times. This compression efficiency demonstrates the method works to reduce the amount of storage and transmission for large scale 3D datasets. The compression ratio shows a notable decrease in file size, which makes it very effective at managing big datasets. The reconstructed data shows little distortion during the compression process and is quite similar to the original, with a mean squared error (MSE) of only 0.001. Furthermore, the lossless reconstruction is confirmed by the infinite peak signal-to-noise ratio (PSNR), which guarantees that the original data is retained with minimal error. These results highlight the proposed method ability to achieve high-quality compression, which makes it a reliable alternative for processing 3D point cloud data in a variety of applications.

The proposed 3DGWT-LAE based compression method shows remarkable effectiveness in minimising the volume of 3D point cloud data while maintaining crucial characteristics. Utilising adaptive coding techniques based on neural networks to further maximise compression ratios without sacrificing reconstruction quality. Extending the process to deal with large 3D datasets, like those produced by LiDAR systems in applications for autonomous driving or real-time urban mapping.

References

1. Alawad, Mohammed, and Mohammad Munzurul Islam. 2024. "Sparsifying Graph Neural Networks with Compressive Sensing." *Proceedings of the Great Lakes Symposium on VLSI 2022*, June. <https://doi.org/10.1145/3649476.3658780>
2. Huang, Linyong, Zhe Zhang, Zhaoyang Du, Shuangchen Li, Hongzhong Zheng, Yuan Xie, and Nianxiong Tan. 2022. "EPQuant: A Graph Neural Network Compression Approach Based on Product Quantization." *Neurocomputing* 503 (June): 49–61. <https://doi.org/10.1016/j.neucom.2022.06.097>.
3. Bernárdez, Guillermo, José Suárez-Varela, Albert López, Xiang Shi, Shihan Xiao, Xiangli Cheng, Pere Barlet-Ros, and Albert Cabellos-Aparicio. 2023. "MAGNNETO: A Graph Neural Network-Based Multi-Agent System for Traffic Engineering." *IEEE Transactions on Cognitive Communications and Networking* 9 (2): 494–506. <https://doi.org/10.1109/tccn.2023.3235719>.
4. P.L. Chithra, and S. Lakshmi bala. 2023. "Deep Learning Model for 3D LiDAR Point Cloud Codec Based on Tensor," November. <https://doi.org/10.1109/icccis60361.2023.10425292>.
5. Liu, Xinyi, Baofeng Zhang, and Na Liu. 2023. "The Graph Neural Network Detector Based on Neighbor Feature Alignment Mechanism in LIDAR Point Clouds." *Machines* 11 (1): 116–16. <https://doi.org/10.3390/machines11010116>.
6. Liang, Zhenming, Yingping Huang, and Yanbiao Bai. 2024. "Pre-Segmented Down-Sampling Accelerates Graph Neural Network-Based 3D Object Detection in Autonomous Driving." *Sensors* 24 (5): 1458–58. <https://doi.org/10.3390/s24051458>.
7. Chithra PL, and Lakshmi S. 2024. "Convolutional Neural Network Model for Stratification of LiDAR 3D Point Cloud Data," January. <https://doi.org/10.4108/eai.23-11-2023.2343218>.
8. Cyrill Stachniss, and Henrik Kretschmar. 2016. "Pose Graph Compression for Laser-Based SLAM." *Springer Tracts in Advanced Robotics*, August, 271–87. https://doi.org/10.1007/978-3-319-29363-9_16.
9. Nuno, Silva Cruz, and Fernando Lopes. 2024. "Impact of LiDAR Point Cloud Compression on 3D Object Detection Evaluated on the KITTI Dataset." *EURASIP Journal on Image and Video Processing* 2024 (1). <https://doi.org/10.1186/s13640-024-00633-4>.
10. Maughey, Thomas, Mira Rizkallah, Navid Mahmoudian Bidgoli, Aline Roumy, and Christine Guillemot. 2021. "Graph Spectral 3D Image Compression," August, 105–32. <https://doi.org/10.1002/9781119850830.ch5>.

11. Schaefer, Gerald. 2008. "Does Compression Affect Image Retrieval Performance?" *International Journal of Imaging Systems and Technology* 18 (2-3): 101–12. <https://doi.org/10.1002/ima.20152>.
12. Yin, Huan, Yue Wang, Li Tang, Xiaqing Ding, Shoudong Huang, and Rong Xiong. 2021. "3D LiDAR Map Compression for Efficient Localization on Resource Constrained Vehicles." *IEEE Transactions on Intelligent Transportation Systems* 22 (2): 837–52. <https://doi.org/10.1109/tits.2019.2961120>.
13. Roriz, Ricardo, Heitor Silva, Francisco Dias, and Tiago Gomes. 2024. "A Survey on Data Compression Techniques for Automotive LiDAR Point Clouds." *Sensors* 24 (10): 3185. <https://doi.org/10.3390/s24103185>.
14. Tamilmathi, A. Christoper, and P. L. Chithra. 2021. "Deep Learned Quantization-Based Codec for 3D Airborne LiDAR Point Cloud Images." *Frontiers in Robotics and AI* 8 (May). <https://doi.org/10.3389/frobt.2021.606770>.
15. Tamilmathi, A. Christoper, and P. L. Chithra. 2022. "Tensor Block-Wise Singular Value Decomposition for 3D Point Cloud Compression." *Multimedia Tools and Applications* 81 (26): 37917–38. <https://doi.org/10.1007/s11042-021-11738-7>.
16. Mahdaoui, Abdelaaziz, and El Hassan Sbai. 2020. "3D Point Cloud Simplification Based on K-Nearest Neighbor and Clustering." *Advances in Multimedia* 2020 (July): 1–10. <https://doi.org/10.1155/2020/8825205>.
17. Yang, Jian, Binhan Luo, Ruilin Gan, Ao Wang, Shuo Shi, and Lin Du. 2024. "Multiscale Adjacency Matrix CNN: Learning on Multispectral LiDAR Point Cloud via Multiscale Local Graph Convolution." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 17 (January): 855–70. <https://doi.org/10.1109/jstars.2023.3335300>.
18. Ching Hsuan Lin, Jyun Yuan Chen, Po Lin Su, and Chung Hao Chen. 2014. "Eigen-Feature Analysis of Weighted Covariance Matrices for LiDAR Point Cloud Classification." *Isprs Journal of Photogrammetry and Remote Sensing* 94 (August): 70–79. <https://doi.org/10.1016/j.isprsjprs.2014.04.016>.
19. Yang, Haozhe, Zhiling Wang, Linglong Lin, Huawei Liang, Weixin Huang, and Fengyu Xu. 2020. "Two-Layer-Graph Clustering for Real-Time 3D LiDAR Point Cloud Segmentation." *Applied Sciences* 10 (23): 8534. <https://doi.org/10.3390/app10238534>.
20. Wu, Rui, and Zhen Liu. 2023. "DGMiniNet: Dynamic Graph Convolution Network for LiDAR Point Cloud Semantic Segmentation on Spherical Coordinates," August, 57–63. <https://doi.org/10.1145/3617695.3617731>.
21. Robotics, A. C. for F. (n.d.). *Sydney Urban Objects Dataset*. www.acfr.usyd.edu.au. <https://www.acfr.usyd.edu.au/papers/SydneyUrbanObjectsDataset.shtml> - Accessed on January 2024.
22. Wang, Zun-Ran, Chen-Guang Yang, and Shi-Lu Dai. 2020. "A Fast Compression Framework Based on 3D Point Cloud Data for Telepresence." *International Journal of Automation and Computing*, July. <https://doi.org/10.1007/s11633-020-1240-5>.