Optimising Point Cloud to Wireframe Conversion for 3D City Modeling

Samantha Soo Chin Yee[1], Uznir Ujang[1]*, Suhaibah Azri[1], Miguel González Cuétara[2] and Ismail Rakip Karas[3]

[1] 3D City Modeling Research Lab, Faculty of Built Environment and Surveying, Universiti Teknologi Malaysia (UTM), 81310 Skudai, Johor, Malaysia – samanthascy01@gmail.com, mduznir@utm.my, suhaibah@utm.my

[2] Torre Siglo XXI. Paseo Fluvial nº15 planta 11, 06011 Badajoz, Spain – ceo@ecapturedtech.com

[3] Faculty of Engineering, Computer Engineering Department, Demir Celik Campus, Karabuk University, 78050 Karabuk, Türkiye – ismail.karas@karabuk.edu.tr

Keywords: Point Cloud Processing, Wireframe Modelling, 3D City Modeling, Plane Detection, Clustering.

Abstract

Urban planning increasingly relies on structured 3D city models for analysis and decision-making, but while point clouds capture rich detail, their unstructured nature complicates direct use in CityJSON-based workflows. We address the challenge of translating raw point clouds into lightweight, interoperable models by confronting the lack of point connectivity, the difficulty of extracting stable planes, edges, and boundaries at scale, and the need for CityJSON-compliant outputs. We introduce a four-phase pipeline based on basic primitives (cuboid, pyramid, sphere, torus) that comprises specification and setup, data capture with EyesCloud3D, preprocessing through downsampling, noise removal, and normal estimation, and processing with RANSAC plane detection, DBSCAN clustering, Convex Hull and Alpha Shapes for boundary extraction, Triangle Normal Analysis for sharp-edge detection, volume-based RMSE evaluation, CityJSON conversion, and visualization in CityJSON Ninja. The approach yielded volume RMSEs of 309.14 cm³ for the cuboid, 70.2541 cm³ for the pyramid, 352.0292 cm³ for the sphere, and 58.1517 cm³ for the torus, while downsampling and denoising reduced point counts substantially, for example from 1,554,781 to 5,151 and then to 5,124 for the cuboid, and wireframes rendered efficiently and validated in CityJSON for the cuboid, pyramid, and sphere although the torus inner hole was not represented correctly. These results indicate that a plane and edge driven pipeline works best for sharp-edged, convex objects, with thresholds around 35 degrees for the cuboid and 30 degrees for the pyramid balancing sensitivity and noise, whereas performance degrades on smooth or non-convex shapes because Triangle Normal Analysis over detects tessellation artifacts on spheres and Convex Hull or Alpha Shapes cannot recover the torus topology. We conclude that the method is effective for sharp-edged urban forms and CityJSON-ready, and we suggest extending it with curvature-aware or implicit-surface reconstruction for smooth or non-convex geometries, adding semantic attributes, auto-tuning thresholds, and evaluating on compound building assemblies representative of real urban scenes.

1. Introduction

3D city modelling is the process of creating detailed three-dimensional digital representations of urban environments such as buildings, roads, terrain, vegetation, infrastructure, and landscape elements. There are various terms used for 3D city models such as ‘Virtual City’, ‘Cybertown’, ‘Cybercity’ and ‘Digital City’. As we know, these 3D models are increasingly important in various applications, especially for urban planning, disaster management, real-time monitoring, and smart city applications, helping cities make more accurate decisions. They also support advanced decision-support analyses such as 3D geomarketing segmentation for higher spatial-dimension planning (Azri et al., 2016). A typical 3D city model is derived from various acquisition techniques, such as photogrammetry and laser scanning, extrusion from 2D footprints, synthetic aperture radar (SAR), architectural models and drawings, handheld devices like using multi-view iPhone images or an iPhone video file as an input, procedural modelling, and volunteered geoinformation (Biljecki et al., 2015).

One of the primary sources of 3D data is point cloud data. These datasets, often acquired from technologies like LiDAR (Light Detection and Ranging) and photogrammetry, which can be further processed to create Digital Elevation Models (DEMs) or Triangulated Irregular Networks (TINs) (Behley et al., 2021), offer highly detailed and accurate representations of cities and buildings. The main challenge is that point clouds are unstructured (Xu et al., 2021). Point clouds provide an accurate and detailed representation of urban environments, but their unstructured nature makes them difficult to integrate directly into 3D city models. Basically, point cloud are not arranged in a regular grid (Bello et al., 2020). Each point is scanned independently, causing the distance between neighbouring points to be varied. Furthermore, the acquired 3D points are unorganized and lack a clear data structure that considers their relationships (Xu et al., 2021). Spatial connections or relationships between the points are unknown. To address these limitations and enable analytics such as 3D geomarketing, Azri et al. (2020) introduced a Voronoi classified and clustered data constellation as a novel 3D data structure that organizes and clusters points for more effective segmentation and planning. For example, point clouds consist of individual points with coordinates (X, Y, Z) but without information on how these points connect to form shapes, surfaces, or objects.

Next, extracting meaningful geometric features such as edges, planes, and boundaries from point cloud faces significant challenges. One of the current methods, which are edge-based methods, can detect all edges, but it is hard to distinguish between the edges of the target area and its boundary (Ruan & Liu, 2020). This traditional edge detection technique, like those developed by Lin et al., Huang & Brenner focus on accurately capturing detailed geometric features, such as edges, borders, and surfaces from 3D point clouds, and can be said to be very

* Corresponding author

useful for complex urban structures (Alshawabkeh, 2020). Complementary to these detection approaches, efficient 3D data organization and retrieval methods, such as the spatial access method and 3D vector data clustering technique proposed by Azri et al. (2014) for urban geospatial database management. It can accelerate processing of dense urban datasets and support faster, more scalable feature-extraction pipelines. Normally high precision edge detection algorithms will process large volumes of data, leading to slow performance if dealing with dense point clouds. However, if using wireframe which focusses on simplifying point cloud data by only capturing essential geometric features, it will be faster and more efficient for large-scale 3D city modelling because fewer resources are being used.

In addition, once the wireframe model is generated from point cloud, converting the wireframe models into CityJSON format while maintaining compatibility with various urban modelling and visualisation tools will be a critical challenge. CityJSON serves as a standard model and requires a specific schema that includes both geometric data and semantic information in 3D city modelling data (Akın & Cömert, 2024). Methods for structuring and quantising 3D vector data, such as the Crisp Clustering Algorithm for 3D geospatial vector data quantization by Azri et al. (2015), can support this conversion by organising wireframe primitives into compact, well-defined encodings that are easier to validate against CityJSON and exchange across systems. But CityJSON, like CityGML, aims to streamline data exchange for 3D city models by offering a simpler and more accessible format. Efficient data conversion from wireframe models to CityJSON would help ensure that 3D city models are easily shared and used in various applications, much like CityGML. If a wireframe model generated from point cloud data does not conform to the CityJSON schema, it may not render correctly in visualisation platforms which can lead to inefficient and limitation of usage due to this incompatibility.

In short, without an optimised process for converting point clouds, urban planners and engineers struggle to use point clouds for city modelling, leading to inefficiencies in decision-making, visualisation, and analysis. This will affect the precision of the city models and limit their use, especially in planning and developing smart cities in line with Malaysia's vision for smart city initiatives, which focus on quality and smart living. Moreover, decision-support approaches that integrate Multiple Criteria Evaluation with GIS have proven valuable in planning contexts such as ecotourism, reinforcing the need for structured, interoperable 3D datasets that can feed robust spatial analyses (Mohd & Ujang, 2016). Furthermore, modern alternatives to traditional point cloud-to-mesh techniques are provided by recent AI-driven techniques that allow for high-fidelity 3D reconstruction of complex geometries and topologies, such as neural implicit representations using SDF, occupancy fields, and NeRF (Zhang et al., 2025). Thus, this research aims to optimise a more efficient method for translating raw data into structured models, enabling users to easily create and work with 3D city models.

2. Methodology

2.1 Scope of Work

This research uses basic 3D primitive objects as shown in Figure 1, such as a cuboid, pyramid, sphere, and torus as foundational components for the wireframe modelling process. It provides a structured framework for translating point-cloud into accurate wireframe models. CityJSON is the primary format used to structure and store the optimise 3D wireframe models.

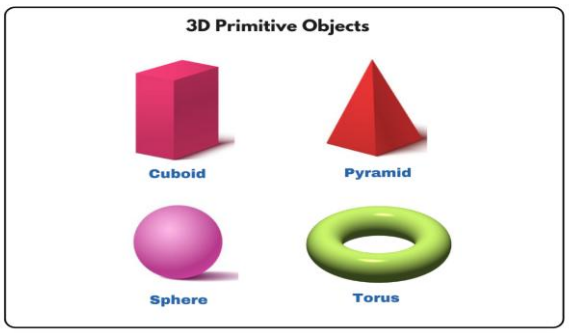


Figure 1. 3D Primitive Objects

2.2 Research Methodology Workflow

The detailed methodology workflow proposed as shown in Figure 2 will document each stage of data handling, from initial preprocessing to final boundary extraction, edge detection, and wireframe generation, providing a clear framework for implementation. Each step will be designed to ensure that the translation process is consistent and efficient. Python will be the primary language due to its wide range of libraries and tools for point cloud processing and 3D modelling. Essential libraries include Open3D, Numpy, Matplotlib, Plotly, Scikit-learn and SciPy.

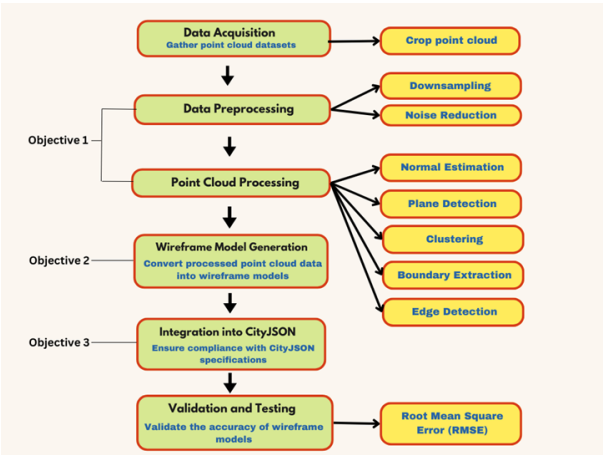


Figure 2. Research Methodology Framework

Downsampling is the process to reduce the number of points in a point cloud while retaining the overall structure and geometry of the object. Noise reduction in point clouds is essential to eliminate outliers. Real-world raw point cloud data often contains noise caused by sensor inaccuracies, data acquisition errors, or environmental factors.

Normal estimation is one of the fundamental preprocessing processes in preparing a 3D point cloud for further 3D geometry processing. A normal vector is used to describe the orientation of a surface at a specific point, and these vectors that provide valuable geometric details are useful for understanding the shape and structure of 3D surfaces represented by point clouds.

Starting from plane detection, each object will be tested with one algorithm, as shown in Figure 3. Plane detection is the

process to identify the largest flat surfaces (planes) in 3D point cloud data. RANSAC can find planes iteratively by fitting models to random point subsets. Next, the clustering step is crucial to organise the non-plane points into meaningful clusters for further analysis based on their spatial relationships. DBSCAN is good at groups dense points into arbitrary-shaped clusters and identifies noise.

Boundary extraction is used to define the shape, extent, and geometry of these features by providing edges or boundaries, making them easier to analyse and visualise. Convex Hull is useful in finding the outer convex boundary, and Alpha Shapes will extract a tighter and on-convex outline. The previous boundary extraction step will only provide an outline, but the next edge detection step is used to identify the sharp edges in the boundaries. Triangle Normal Analysis is good in detecting edges where adjacent triangle normal shows large angular differences.

After testing, the volume root mean square error (RMSE) is calculated by measuring the volume of each object in SketchUP and the visual quality of each model will be evaluated. The final output is converted to cityjson format and visualised using CityJSON Ninja.

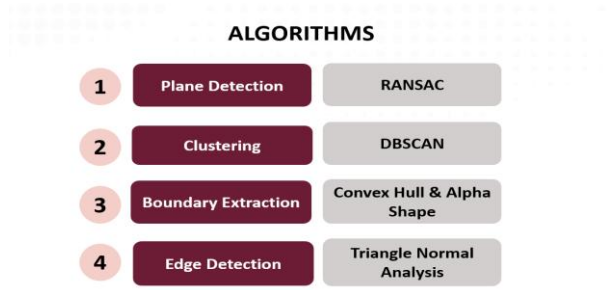


Figure 3. The algorithms used for each step

2.3 Data Collection

Data is collected by capturing images or videos of targeted objects, generating point clouds, cropping the point cloud and preparing for the wireframe model conversion by downloading the data in compatible format using EyesCloud3D. The targeted objects include cuboid, pyramid, sphere and torus. The step-by-step processes are summarised in Figure 4 below.

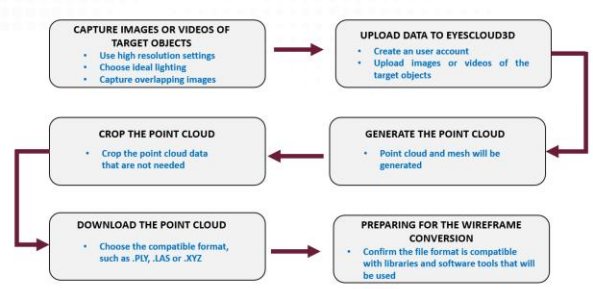


Figure 4. The overall steps of generating point cloud data using eyesCloud3D

The results of data collection for each object are shown in Figure 5. When the data collection processing is complete, download the generated point cloud in a compatible format, which is .PLY to be used in other applications or software like Open3D for further analysis and model refinement.

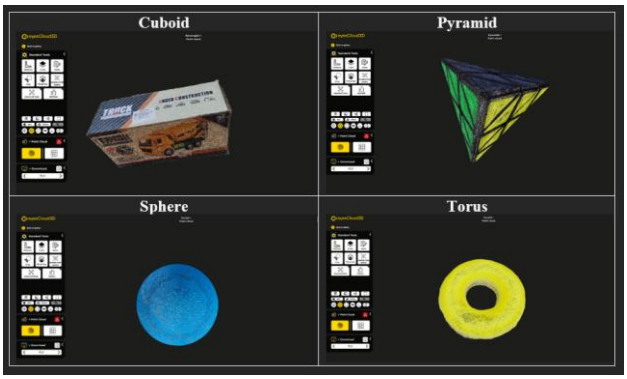


Figure 5. The data collected for each object after cropping

3. Results and Discussion

The result of this study is divided into six parts, which are pre-processing results, the results of wireframe generation from point cloud for each object such as cuboid, pyramid, sphere, and torus and evaluation of visual quality of the wireframe models.

3.1 Pre-processing Results

There are three steps of preprocessing, which are downsampling, noise reduction, and normal estimation. The obtained pre-processing results have improved the overall data quality, emphasising that a good initial processing results is important for further processing steps.

The results showing the point count before and after downsampling are shown in Table 1.

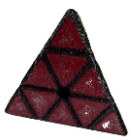
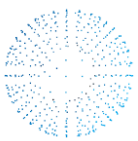
Objects	Original Point Count	Downsampled Point Count
Cuboid	 1554781	 5151
Pyramid	 1288963	 262
Sphere	 1489853	 477
Torus	 604692	 11877

Table 1. The point count of the original and downsampled point count

Removing noise is an essential step to ensure that the data set is clean and suitable for further processing. The results showing the point count before and after noise reduction are shown in Table 2.

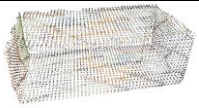
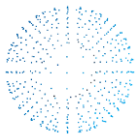
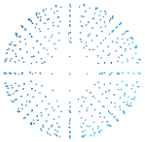
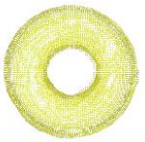
Objects	Downsampled Point Count	Cleaned Point Count
Cuboid	 5151	 5124
Pyramid	 262	 247
Sphere	 477	 459
Torus	 11877	 11450

Table 2. The point count of the downsampled and cleaned point count

The results showing the point cloud after normal estimation are shown in Table 3.

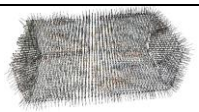
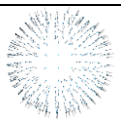
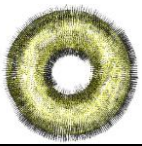
Objects	Point Cloud with Normals	Objects	Point Cloud with Normals
Cuboid		Sphere	
Pyramid		Torus	

Table 3. Estimated normal for the point cloud of each object

Accuracy testing is crucial for assessing the quality of the generated models. The RMSE is calculated on the volume difference between the original objects and the wireframe model output using the formula shown in Table 4, which involves comparing the volume, edge length, height, radius between the original and wireframe models.

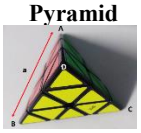
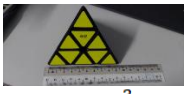
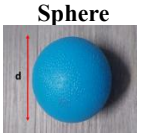
Objects	Formula	Volume of real object
Cuboid 	$V = l \times w \times h$ where: l = length, w = width, h = height	 V = (23 x 12 x 8) cm = 2208 cm³
Pyramid 	$V = \frac{a^3}{6\sqrt{2}}$ where: a = length of an edge $Average\ edge = \frac{e1+e2+e3+e4+e5+e6}{6}$ The average length for each edge is calculated for measured model	 $V = \frac{a^3}{6\sqrt{2}}$ = 85.91 cm³
Sphere 	Radius, r : $r = \frac{d}{2}$ where : d= diameter Volume : $V = \frac{4}{3}\pi r^3$	 $r = \frac{4.738\ cm}{2}$ =2.369 cm $V = \frac{4}{3}\pi(2.369)^3$ =55.69 cm³
Torus 	$V = 2\pi^2 Rr^2$ where: R = major radius (distance from the centre of the tube to centre of the torus) r = minor radius (radius of the tube)	 $V = 2\pi^2(2.6)(1.1)^2$ = 62.10 cm³

Table 4. The formula for four objects and the calculated volume of each real object.

3.2 Results of wireframe generation from point cloud for each object

These results in Table 5 show the point cloud processing outputs of each object for plane detection, clustering, and boundary extraction.

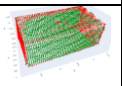
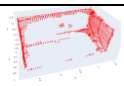
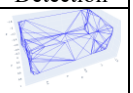
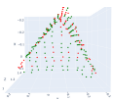
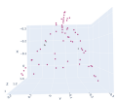
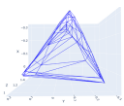
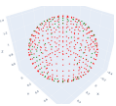
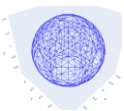
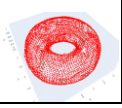
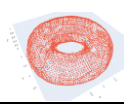
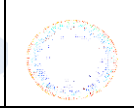
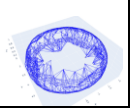
Objects, ID	Plane Detection	Clustering	Boundary Extraction	Edge Detection
Cuboid, C1				
Pyramid ,P1				
Sphere, S1				
Torus, T1				

Table 5. The results of point cloud processing for each object

For each configuration, cuboid, pyramid and torus will be tested with 30 °, 35 ° and 40 ° of the sharp angle thresholds, while sphere will be tested with 5, 10 and 15 of the sharp angle thresholds using Triangle Normal Analysis, resulting in outputs like C1_30, C1_35 and others. The wireframe models are then being evaluated by measuring their volume, the results are shown in Table 6 below.

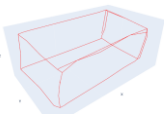
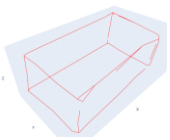
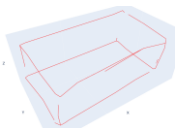
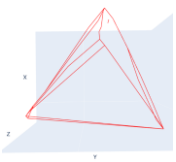
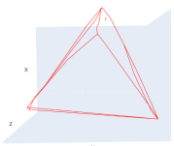
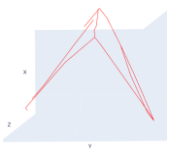
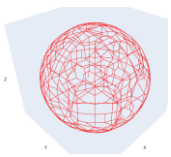
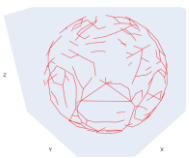
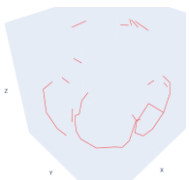
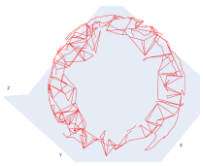
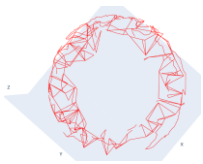
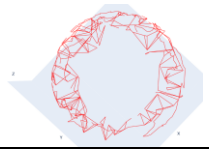
ID	Wireframe Models	Volume of Wireframes (cm^3)
C1_30		**1832.54**
C1_35		**1415.23**
C1_40		**1221.70**
P1_30		**11.35**
P1_35		**10.27**
P1_40		**5.58**
S1_5		**434.89**
S1_10		**434.89**
S1_15		**407.72**
T1_30		**3.95**
T1_35		**2.40**
T1_40		**2.04**

Table 6. The wireframe models of each object and their volume

After that, the RMSE of volume, length, width and height are calculated and they are ranked based on the lowest volume RMSE. The ranking of volume RMSE for wireframe models of cuboid, including length RMSE, width RMSE and height RMSE are shown in Table 7.

Rank	ID	Volume RMSE (cm^3)	Length RMSE (cm)	Width RMSE (cm)	Height RMSE (cm)
1	C1_35	309.1400	0.7	0.8	0.4
2	C1_40	324.8600	1.1	0.8	0.4
3	C1_30	394.4600	0.7	1.0	0.7

Table 7. The ranking of volume RMSE for wireframe models of cuboid

The ranking of volume RMSE for wireframe models of pyramid, including edge RMSE and height RMSE are shown in Table 8.

Rank	ID	Volume RMSE (cm^3)	Edge RMSE (cm)	Height RMSE (cm)
1	P1_30	70.2541	4.0833	3.3260
2	P1_40	72.3557	4.4667	3.6132
3	P1_35	72.8312	4.2667	3.5254

Table 8. The ranking of volume RMSE for wireframe models of pyramid

The ranking of volume RMSE for wireframe models of sphere, including diameter RMSE and radius RMSE are shown in Table 9.

Rank	ID	Volume RMSE (cm^3)	Diameter RMSE (cm)	Radius RMSE (cm)
1	S1_15	352.0292	4.4620	2.2310
2	S1_5	379.2019	4.6620	2.3310
3	S1_10	379.2019	4.6620	2.3310

Table 9. The ranking of volume RMSE for wireframe models of sphere

Lastly, the ranking of volume RMSE for wireframe models of torus, including major radius RMSE and minor radius RMSE are shown in Table 10.

Rank	ID	Volume RMSE (cm³)	Major Radius RMSE (cm)	Minor Radius RMSE (cm)
1	T1_30	58.1517	4.10	1.40
2	T1_35	59.7012	4.10	1.60
3	T1_40	60.0565	4.50	1.60

Table 10. The ranking of volume RMSE for wireframe models of torus

4. Discussion

This section discusses the analysis of the generated wireframe models for the cuboid, pyramid, sphere, and torus. It includes an evaluation of their structural accuracy and visual quality to assess the effectiveness of the applied methods.

4.1 Analysis of wireframe models for each object

The wireframe models for each object with their volume RMSE and their ranking are shown in Figure 6.

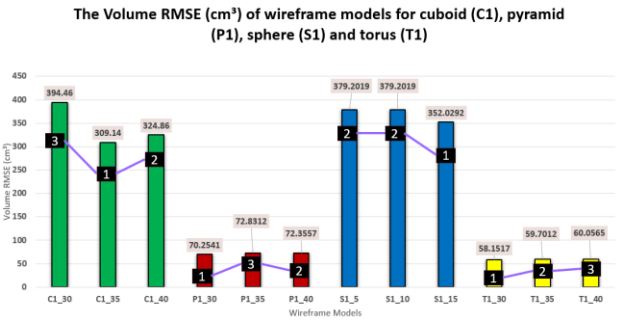


Figure 6. The ranking of wireframe models based on their volume RMSE

A cuboid is an ideal object for many 3D reconstruction algorithms due to its perfectly planar faces, sharp edges at 90-degree angles, and it is a convex object. Based on Figure 6, C1_35 and C1_40 ranked 1 and 2 for volume RMSE, which gives 309.14 cm³ and 324.86 cm³ respectively. For C1, RANSAC provides stable plane detection, DBSCAN removes outliers effectively, and Convex Hull & Alpha Shape fits cuboid shape well. The rank for C1 dropped when the sharp edge threshold increased from C1_35 (rank 1) to C1_40 (rank 2) due to worsened length RMSE (0.7 to 1.1) and increased volume RMSE. C1_30 has the worst performance (Rank 3) for volume RMSE among C1 than C1_35 and C1_40. This suggests that for this combination, a threshold of 40 ° was too strict, causing the Triangle Normal Analysis to miss actual, slightly less than 40 ° edges on the cuboid, a threshold of 30 ° is also too sensitive, while the 35 ° threshold probably provided a better balance between detecting true edges and avoiding noise. Although Convex Hull & Alpha Shape are powerful, their effectiveness depends heavily on the specific alpha parameter chosen.

A pyramid is a convex object with planar faces, sharp and defined edges where faces meet. For P1 combinations, P1_30 with a volume RMSE of 70.2541 cm³ is ranked 1, followed by P1_40 (rank 2) and P1_35 (rank 3) with volume RMSE of 72.3557 cm³ and 72.8312 cm³ respectively. It cannot be denied that RANSAC and DBSCAN are strong, which gives accurate results. For boundary extraction, P1 is using an effective algorithm called Convex Hull & Alpha Shape that can perfectly extract the boundary of the pyramid and is ideal for convex objects such as cuboids and pyramids. In this combination, the 30 ° sharp angle threshold is also excellent for detecting most sharp edges. Similar to other models, in this case, a threshold of 40 ° is slightly too strict that might cause tiny imperfections along the edges that a 30 ° threshold would capture, while the 35 ° threshold is less aggressive than 30 ° which might result in missing details. This highlights that the choice of sharp angle threshold is significant to improve overall accuracy. The volume RMSE for P1_40 (rank 2) is higher than P1_35 (rank 3), although P1_40 is good at overall volume capture, but its edge RMSE (4.4667) and height RMSE (3.6132) are relatively worse than P1_35 with 4.2667 and 3.5254 of edge and height RMSEs, respectively. This is because 40 ° threshold will cause a less precise edge localisation, leading to the higher edge and height RMSE.

A sphere is a continuously curved, convex object with no planar faces and no sharp edges. For sphere, S1_15 ranked 1 with volume RMSE, diameter RMSE and Radius RMSE which are 352.0292 cm³, 4.462 cm³ and 2.231 cm³ respectively. The remaining models, S1_5 and S1_10 have the same ranking (2) due to their same 379.2019 cm³ of volume RMSE, 2.331 cm³ of radius RMSE and 4.662 cm³ of diameter RMSE. This result proves that Triangle Normal Analysis used by S1 is not suitable and effective in detecting meaningful ‘edges’ for spheres that are without sharp edges, it includes a significant amount of tessellation artefacts and noise as ‘edges’, causing a highly jagged and distorted wireframe model that leads to high RMSE. Since spheres do not have real sharp edges, the use of different sharp angle thresholds does not impact the effect of ‘edges’ detection differently because it does not extract relevant boundary information. The volume RMSE is considered high which is far away from the real volume, showing that the initial processing steps and the limitations of edge detection methods for a sphere will result wireframe models become oversized (mostly 9cm+ of diameter) and poorly defined the representation of the sphere.

A torus is a complex, smooth, and non-convex 3D shape that lacks sharp edges, corners and planar faces. T1_30 is ranked 1 with 58.1517 cm³ of volume RMSE. Obviously, by using Convex Hull & Alpha Shape, it fails to represent the inner hole or circle of the torus, producing a solid ring, so this limitation will limit the model accuracy and affect the visual quality of the wireframe model. At 30 ° of threshold, more ‘edges’ will be detected but false ‘edges’ have no useful boundary information for the major and minor radii due to the limited approximation from Convex Hull. The same goes for T1_35, which ranked 2 with 59.7012 cm³, only a few unuseful ‘edges’ are detected. In addition, T1_40, which ranked 3 have 60.0565 cm³ of volume RMSE, in which it uses the strictest 40 ° threshold that provides the least useful ‘edges’ information. The failure of Convex Hull to capture the inner circle of the torus and Triangle Normal Analysis to provide useful ‘edges’ caused the formation of a significantly oversized and incorrect wireframe model.

4.2 Analysis of visual quality and wireframe integrity for each object

In addition to using RMSEs to evaluate the accuracy of wireframe models, the evaluation of visual quality and wireframe integrity is also significant to ensure consistency and correctness compared to the original objects. The evaluation of visual quality for the wireframe model of each object with lowest volume RMSE is shown in Table 11 below.

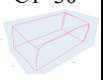
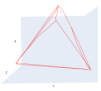
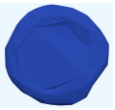
Objects	Best Visualised Model	Visual Quality Summary	CityJSON Valid?	Visualised correctly in CityJSON Ninja?
Cuboid	C1_30 	Complete planes, slight warping at corners and duplicated lines	Yes 	Yes
Pyramid	P1_35 	Complete faces, minor lines with minimal line duplication	Yes 	Yes
Sphere	S1_5 	Good spherical shape, fully enclosed, too dense	Yes 	Yes
Torus	T1_30 	Partial ring, fewer break lines and line duplication	Yes 	No

Table 11. The visualised wireframe model for each object with lowest volume RMSE

The cuboid models are highly compatible with edge-based wireframe extraction methods in which they produce strong visualisation results. This success is due to the geometric properties of cuboid, with flat planar structures and well-defined sharp edges that strongly aligns with the algorithms for boundary extraction and sharp angle identification. According to the results, most of the models are captured with their high accuracy of six faces and sharp edges, creating wireframes that are both structurally complete and visually clean. Its suitability also can be reflected from the well-preserved rectangular shape of cuboid, with continuous, straight and closed lines that enhance both structural and visual clarity. The consistency of cuboid's wireframe demonstrates how well the edge detection technique represents objects with sharp edges like cuboids, making it suitable in applications for 3D modelling. However, there are still minor issues such as disconnected corners, slight gaps between edge segment and duplicating lines, which is likely due to sharp angle threshold sensitivity or noise from plane detection, in which they would not largely affect the overall visual quality of wireframe. For example, C1_30 has complete planes, but it has slightly duplicated lines at the corner. Such limitations can be addressed by using post-processing techniques such as vertex merging and continuity filtering to eliminate redundancies and ensure full edge closure. All the models are compatible with CityJSON format and can be visualised as solid in CityJSON Ninja.

Pyramids with well-defined sharp edges show a higher level of visual success in which most of the models have well-formed faces and the overall geometry is structurally complete. The pyramid's flat triangular faces and sharp, intersecting lines at the base and apex make the pyramid fit for angle-based and boundary extraction methods. The overall pyramid shapes are well captured, and the edges are mostly represented by continuous lines with minimal distortion like duplicated and unconnected lines, including P1_35. The sharp edge transitions particularly at the base corners and apex points are precisely identified and visually presented, resulting in models that can preserve the characteristic geometry of a pyramid. These results clearly indicate that the edge-based detection method performs well for objects with angular or sharp edge structures like pyramids. However, several models give minor structural weaknesses such as slightly jagged lines, disconnected edges or gaps at edge intersections, but they are secondary in nature in which they can be further improved during post-processing steps like edge joining and angular smoothing to maintain the accuracy of pyramid structure and enhance continuity. Like cuboids, all the models are compatible with CityJSON format and can be visualised correctly as solid in CityJSON Ninja.

Based on the wireframe results, sphere presented significant challenges in wireframe visualisation, even though S1_5 and S1_10 generate structurally complete models. The sphere's smooth and curved surface without any sharp edges is the primary issue that leads to the inaccurate representation. It is because edge detection is better suited for objects with well-defined edges like cuboid and pyramid, it cannot detect continuous boundaries, resulting in fragmented, broken and disconnected lines from all sides. The S1_15 model that give the worse performance output which only present partial disconnected lines, causing them being far different from how a complete spherical representation should be looks like. Even when the models with clearly visible spherical structure, the lines are often jagged and duplicated, which reduces the visual clarity as an accurate sphere wireframe. These results are due to the inability and unsuitability of edge-detection methods to handle continuous curvatures without sharp angular transitions like spheres. For a proper and precise wireframe model of a sphere, a smooth and evenly spaced longitudinal and latitudinal arcs forming a grid-like mesh are expected to be developed. To achieve such a perfect wireframe of a sphere, curvature-aware segmentation techniques of mesh approximation methods are required, to generate a more structured and cohesive wireframe output for spheres. All the models are compatible with CityJSON format and can be visualised correctly in CityJSON Ninja.

Torus models had serious limitations in terms of the quality of wireframe visualisation, even though some of them have low RMSE values. Like spheres, the torus also has a smooth surface with no sharp edges, which makes it challenging and difficult to use such edge detection-based methods. All torus models have partially complete outer and inner circles of torus, but the lines are broken, discontinuous, unconnected or misaligned. This fragmented representation of torus significantly reduces the completeness of the model because the inner holes and outer loops of torus do not even form closed and smooth circular structures, leading to an open space and big gaps along the

rings. The wireframe's clarity also degraded due to its overlapping and duplicated line segments that made the models less recognisable as a torus. Unlike objects with sharp edges like cuboid and pyramid, the torus lacks strong curvature changes, making its donut-like topology difficult to be captured using angle-based edge criteria. An ideal wireframe representation of a torus should be having clean and concentric rings with smooth transitions using continuous lines along the tube surface and the major circular path. The current methods used are not suitable for torus due to its limitations in edge detection and curve connectivity. To create an accurate representation of torus, the use of algorithms that can detect the object without sharp edges should be adopted for torus. Increasing the sharp angle threshold to eliminate noise and post-processing steps like edge merging and connecting can be carried out to further improve these models. All the models are compatible with CityJSON format, but they cannot be visualised correctly in CityJSON Ninja because the inner circle of torus is not being visualised.

5. Conclusion & Recommendations

In conclusion, this research evaluated the pipeline of optimising point cloud to wireframe across objects such as cuboids, pyramids, spheres, and tori. For cuboids and pyramids, which have flat faces and sharp edges, they perform well and highly suitable for the method. This is proved by their low volume RMSE and at the same time showing structurally complete and precise in terms of visual quality of wireframe, with only minimal distortion like disconnected and duplicated lines. In contrast, spheres and tori which have smooth surfaces and lack of edges of tori and spheres present significant challenges and limitations. The Triangle Normal Analysis used for edge detection failed to detect meaningful features from these curved surfaces; it only found artefacts instead of edges. For torus which are non-convex pose a major challenge due to boundary extraction using Convex Hull & Alpha Shape methods producing topologically incorrect wireframe models; adopting topologically robust representations such as an abstract topological data structure could help preserve valid adjacency and incidence relationships during conversion (Ujang, Castro, & Azri, 2019). This pipeline is effective for sharp-edged objects like cuboids and pyramids, but it is unsuitable for smooth, edgeless, and complex objects such as spheres and tori, indicating the need for alternative reconstruction techniques or further post-processing steps for improvement.

Future research should evaluate the performance using more complex geometries that resemble real-world scenarios, such as 3D city buildings made up of several combinations of geometric primitives such as cuboids, pyramids, cylinders, etc. This would make it possible to have a better understanding of how this approach performs under various structural constraints that are frequently encountered in urban environments, as well as how it scales with different object complexity. In parallel, adopting efficient 3D urban data management strategies, such as the classified and clustered data constellation proposed by Azri et al. (2016), could organise heterogeneous primitives and attributes more effectively, enabling scalable evaluation, faster querying, and smoother integration into downstream urban-analysis workflows.

Since the current pipeline uses edge-based detection methods, which are made for features that are flat and sharply edges, it is not suitable for smooth and continuous curved objects like tori and spheres. It is recommended that future research should explore the reconstruction methods that are better suited for non-linear topologies and continuous curvature, such as implementing an implicit surface fitting or other surface reconstruction methods, which are better in persevering the geometry of smooth surfaces. Related work on 3D volumetric soft geo-objects demonstrates how volumetric representations can handle continuous phenomena and support robust modelling and analysis (Yusoff et al., 2011).

Acknowledgements

This research was supported by UTM Research University Grant, Vot Q.J130000.3852.23H58 and partially funded by UTM Research University Grant, Vot Q.J130000.3852.23H72.

References

Akın, A. T., & Cömert, Ç. (2024). "CITYJSON2RDF" A converter for producing 3D city knowledge graphs. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, *XLVIII-4/W9-2024*, 15-19. https://doi.org/10.5194/isprs-archives-XLVIII-4-W9-2024-15-2024

Alshawabkeh, Y. (2020). Linear feature extraction from point cloud using color information. *Heritage Science*, *8*(1), 28. https://doi.org/10.1186/s40494-020-00371-6

Azri, S., Ujang, U., Rahman, A. A., Anton, F., & Mioc, D. (2016). 3D Geomarketing Segmentation: A Higher Spatial Dimension Planning Perspective International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives, http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-4-W1/1/2016/

Azri, S., Ujang, U., & Abdul Rahman, A. (2020, 2020/04/01/). Voronoi classified and clustered data constellation: A new 3D data structure for geomarketing strategies. ISPRS Journal of Photogrammetry and Remote Sensing, 162, 1-16. https://doi.org/https://doi.org/10.1016/j.isprsjprs.2020.01.022

Azri, S., Ujang, U., Anton, F., Mioc, D., & Rahman, A. A. (2014, 29 September 2014 - 1 October 2014). Spatial Access Method for Urban Geospatial Database Management: An Efficient Approach of 3D Vector Data Clustering Technique 9th International Conference on Digital Information Management (ICDIM), Bangkok, Thailand.

Azri, S., Ujang, U., Castro, F. A., Abdul Rahman, A., & Mioc, D. (2016, 3//). Classified and clustered data constellation: An efficient approach of 3D urban data management. ISPRS Journal of Photogrammetry and Remote Sensing, 113, 30-42. https://doi.org/http://dx.doi.org/10.1016/j.isprsjprs.2015.12.008

Azri, S., Anton, F., Ujang, U., Mioc, D., & Rahman, A. A. (2015). Crisp Clustering Algorithm for 3D Geospatial Vector Data Quantization. In Lecture Notes in Geoinformation and Cartography (pp. 71-85). Springer Verlag. http://link.springer.com/chapter/10.1007%2F978-3-319-12181-9_5

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Gall, J., & Stachniss, C. (2021). Towards 3D LiDAR-based semantic scene understanding of 3D point cloud sequences: The SemanticKITTI Dataset. *The International Journal of Robotics Research*, *40*(8-9), 959-967.

Bello, S. A., Yu, S., Wang, C., Adam, J. M., & Li, J. (2020). Review: Deep Learning on 3D Point Clouds. *Remote Sensing*, *12*(11).

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., & Çöltekin, A. (2015). Applications of 3D City Models: State of the Art Review. *ISPRS International Journal of Geo-Information*, *4*(4), 2842-2889.

Mohd, Z. H., & Ujang, U. (2016). Integrating Multiple Criteria Evaluation and GIS In Ecotourism: A Review Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., http://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-4-W1/351/2016/

Ruan, X., & Liu, B. (2020). Review of 3D Point Cloud Data Segmentation Methods. *International Journal of Advanced Network, Monitoring and Controls*, *5*(1), 66-71. https://doi.org/doi:10.21307/ijanmc-2020-010

Ujang, U., Castro, F. A., & Azri, S. (2019). Abstract Topological Data Structure for 3D Spatial Objects. ISPRS International Journal of Geo-Information, 8(3). https://doi.org/10.3390/ijgi8030102

Yusoff, I. M., Ujang, U., Rahman, A. A., Katimon, A., & Ismail, W. R. (2011, 4//). Influence of georeference for saturated excess overland flow modelling using 3D volumetric soft geo-objects. Computers & Geosciences, 37(4), 598-609. https://doi.org/http://dx.doi.org/10.1016/j.cageo.2010.05.013

Xu, Y., Tong, X., & Stilla, U. (2021). Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry. *Automation in Construction*, *126*, 103675. https://doi.org/https://doi.org/10.1016/j.autcon.2021.103675

Zhang, X., Yu, R., & Ren, S. (2025). Neural Implicit Representations for Multi-View Surface Reconstruction: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, *31*(10), 9444-9463. https://doi.org/10.1109/TVCG.2025.3582627