Comparative Analysis of the Accuracy of LiDAR Data Conversion Algorithms to CityJSON Models: Case of Open Source and FME Commercial Software.

Upetha K G B[*],Piyasena N M P M[**]

Department of Surveying and Geodesy, Faculty of Geomatics, Sabaragamuwa University of Sri Lanka.

Abstract- This study conducts a comparative analysis of the accuracy and efficiency of converting LiDAR data into CityJSON models using Python-based tools and FME commercial software, focusing particularly on model similarity, accuracy variability, data utilisation, and the dependency on LiDAR precision. Findings indicate that Python-based models achieve a 63.9% similarity rate in the generated 3D models, suggesting significant room for refinement to enhance detail and accuracy, particularly as model size increases larger models tend to face scaling challenges affecting their accuracy. Unlike FME, which employs a simplified approach by using fewer points from LiDAR point clouds as vertices, Python utilises all available points, allowing for detailed but computationally intensive modelling. This comprehensive utilisation of data points can lead to higher fidelity in the representation of physical spaces, although it also requires substantial computational resources, potentially slowing down the processing time. The accuracy of both modelling approaches heavily depends on the precision of the underlying LiDAR data. The simplification strategy adopted by FME may help increase overall system efficiency and accuracy by reducing noise and computational demands. However, Python's detailed use of all LiDAR points as vertices offers potential for greater model detail at the expense of increased system load. The study suggests that Python models could significantly benefit from simplifying point selection processes, which would align their accuracy and efficiency closer to FME's outcomes. This research enhances the understanding of the practical implications of software choice in 3D city modelling, providing valuable insights that assist geospatial professionals in selecting the most appropriate tools based on specific project needs, thereby influencing future innovations and standard practices in the field.

Keywords – CityJSON, LiDAR to CityJSON, Open Source 3D Models, Python Based 3D models.

1. INTRODUCTION

Urban modelling has been transformed by digital technologies, especially through formats like CityJSON, which efficiently represents 3D urban environments. Originating from CityGML, CityJSON simplifies 3D model complexities while maintaining rich detail and compatibility with geospatial standards. It aims for accessibility and developer-friendliness, facilitating tool and API development for urban analysis, crucial as urban complexities increase (Ledoux *et al.*, 2019; Wang *et al.*, 2020).

Complementing CityJSON, the Feature Manipulation Engine (FME) is pivotal in converting diverse geospatial data into CityJSON, enhancing the management of expanding urban environments. FME automates data conversion tasks, allowing professionals to focus on analysis rather than data format intricacies, highlighting its role in urban and digital twin technologies development(*Documentation | FME Community*, no date).

The integration of Python in urban modelling illustrates a shift towards interoperability and open-source solutions in geospatial technologies. Python's appeal lies in its simplicity and the extensive library ecosystem, supporting a range of applications from data manipulation to spatial analyses and machine learning. Python-based platforms are integral for processing and analyzing CityJSON data, addressing urban development challenges like infrastructure planning and disaster risk assessment (*3D Geoinformation*, no date).

This research tackles the challenge of converting LiDAR data into CityJSON format, a task that intersects geospatial analysis, urban planning, and software engineering. LiDAR data, despite its richness, presents significant processing and conversion challenges, necessitating advanced software like Python and FME. Python offers flexibility and extensive library support for point cloud processing, crucial for segmentation algorithms. Conversely, FME excels in data integration and conversion, ideal for managing complex spatial relationships (Pfeifer *et al.*, 2014; Shahidinejad, Kalantari and Rajabifard, 2024)

The objective is to conduct a comparative analysis of Python and FME in processing LiDAR data for CityJSON models, assessing their effectiveness, efficiency, and accuracy in creating urban digital twins. This evaluation will inform recommendations for their application in professional practice, aiming to enhance the geospatial and urban planning knowledge base.

Expected outcomes include a detailed analysis of each platform's methodologies, an evaluation of their accuracy, and the development of best practices for using LiDAR data in CityJSON models. These insights will guide urban planners, architects, and geospatial professionals in choosing suitable tools based on specific project needs. The study focuses on publicly available LiDAR datasets and current software versions, which may limit the generalizability of findings to other contexts or software updates. Nonetheless, the research seeks to influence current practices and encourage further exploration in urban digital modelling.

2. LITERATURE REVIEW

The literature review delves into the significant roles of LiDAR technology, CityJSON, and Python in enhancing urban modelling and geospatial analysis. It underscores their contributions to improving the accuracy and efficiency of 3D city models. LiDAR technology is crucial as it provides detailed three-dimensional data that is vital for creating precise models used in urban planning and environmental management. These models assist in developing smart city initiatives and are essential for applications like flood risk assessments (Hernandez-Garcia *et al.*, 2011; Nys, Poux and Billen, 2020).

CityJSON, developed as an advancement from CityGML, presents a streamlined format that simplifies the representation of 3D urban environments while ensuring compatibility with urban and geospatial standards. Its efficiency and user-friendly design promote broader use in urban and geospatial modelling, enhancing data handling and visualisation across various projects (Wang *et al.*, 2020; Nys and Billen, 2021).

Python, integrated with libraries such as GDAL, Pyproj, and Shapely, facilitates efficient data processing and manipulation, positioning it as a fundamental tool in geospatial analysis and urban modelling. Python's role extends to managing LiDAR data using libraries like PDAL and laspy, and it integrates with 3D modelling tools such as Blender, crucial for producing realistic urban models (Khayyal, Zeidan and Beshr, 2022; Cortés, 2023).

The geometry of these models relies on the use of vertices and faces. Vertices are points in 3D space that define the corners of shapes, while faces are polygons formed by these points. This structure is key to precise modelling and rendering, enabling the creation of accurate digital representations of urban environments (Wang *et al.*, 2020).

Accuracy assessment is a critical aspect, focusing on precise surface area measurements to check model completeness and identify areas needing improvement. The similarity percentage, which compares the model's surface area with the actual object, is vital for ensuring accuracy in heritage conservation projects. Additionally, understanding the relationship between a model’s geometry and its area is crucial for confirming that the model accurately reflects the real object’s spatial characteristics, a key factor for reliability (Elberink and Vosselman, 2007; Akca *et al.*, 2010; Borkowski and Jóźków, 2012).

Overall, these technologies and methods significantly enhance urban modelling, contributing to more sustainable and effective urban planning and development.

3. METHODOLOGY

The methodology employs a detailed 3D model of Nijmegen, Netherlands, sourced from Kadaster's national 3D Basisvoorziening. This model integrates high-resolution LiDAR data for accurate topography, supplemented by BGT and BAG registrations for geographic and structural details, and enhanced by aerial imagery to enrich visual context.

In the analysis, the comparison of roof surface areas between standard and Python-enhanced CityJSON models is assessed using Mean Absolute Percentage Error (MAPE) and similarity percentages. MAPE provides a quantifiable measure of the average deviation between model estimations in percentage terms, facilitating accuracy evaluation. Concurrently, similarity percentages gauge the alignment between the model outputs, serving as an efficacy metric. This analytical approach is designed to illuminate discrepancies and potential enhancements in the Python-enhanced model relative to the standard configuration.

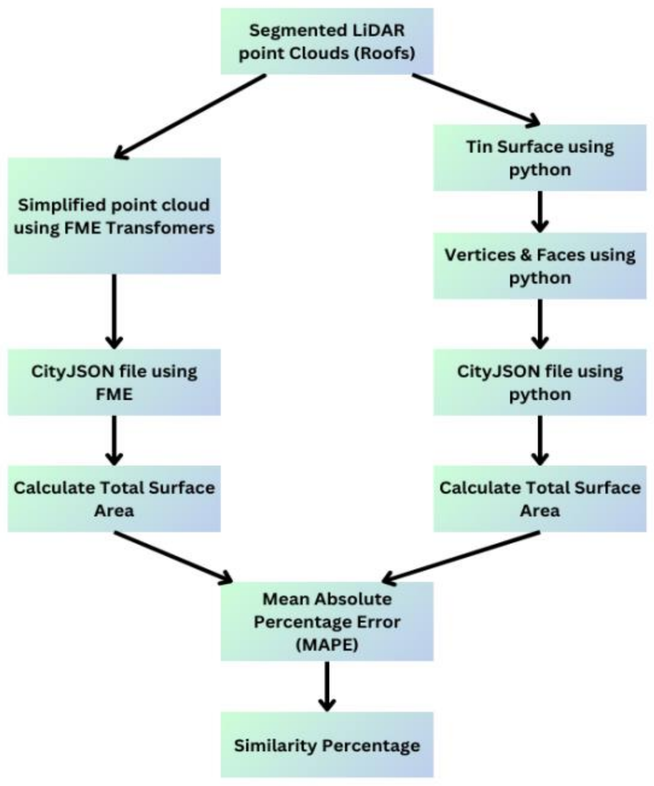
Segmented LiDAR point Clouds (Roofs)
Simplified point cloud using FME Transfomers
Tin Surface using python
Vertices & Faces using python
CityJSON file using FME
CityJSON file using python
Calculate Total Surface Area
Calculate Total Surface Area
Mean Absolute Percentage Error (MAPE)
Similarity Percentage

Figure 1: Methodology

Equation 1

$$Similarity\ Percentage = 100\,\% - MAPE$$

Equation 2

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Ai - Pi}{Ai}\right| \times 100\,\%$$

Ai - Area of the FME model.

Pi - Area of the CityJSON model.

n - Number of roofs (samples).

4. RESULTS ANALYSIS AND DISCUSSION

The graph will illustrate the comparison of roof surface areas between the standard and Python-enhanced CityJSON models, highlighting how the Python-enhanced version generally predicts larger areas, underscoring the enhancements' impact on model accuracy.

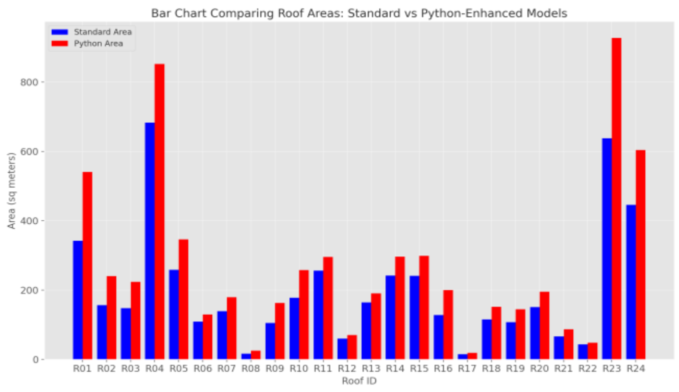

Figure 2: Roof Surface Areas

The comparative bar chart illustrates significant variations in roof areas between standard and Python-enhanced CityJSON models. This visualisation highlights the Python model's capacity to substantially increase roof area measurements, suggesting its effectiveness for more detailed urban modelling and analysis, potentially improving accuracy in urban planning simulations.

This scatter plot compares roof areas between the standard and Python-enhanced CityJSON models. The "100% Similarity Line" illustrates where areas are equal, serving as a baseline for assessing the impact of Python enhancements on urban modelling accuracy.

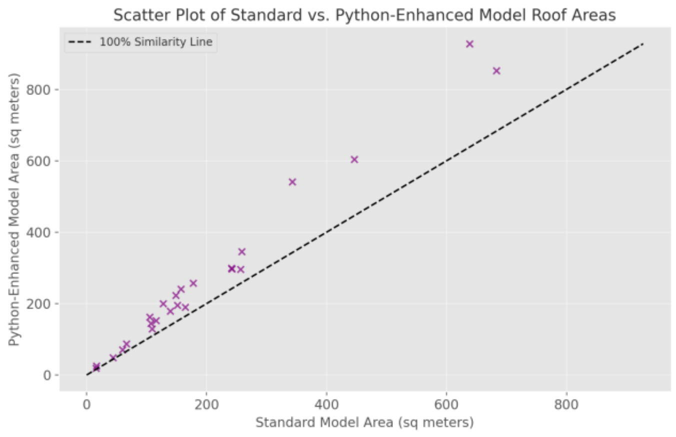

Figure 3 : Scattered Plot of the Roof Areas

Points above the "100% Similarity Line" indicate greater roof areas in the Python-enhanced model, highlighting its potential for more detailed and expansive urban analysis. This trend suggests improved data handling and increased model precision facilitated by Python scripting.

To ascertain the fidelity of CityJSON model augmentations, roof areas were systematically segregated into four discrete classes, and both the Mean Absolute Percentage Error (MAPE) and similarity percentages were meticulously computed for each category. The ensuing data, delineated in the table below, suggests a notable homogeneity in MAPE and similarity percentages across the varying classes of roof areas.

Table 1 : Class wise Similarity Percentage

Roof Area Class	MAPE (%)	Similarity Percentage(%)
0 - 200	36.60	63.40
200 - 400	35.23	64.77
400 - 600	35.39	64.61
600 - 800	35.06	64.94

The results evince a marginal fluctuation in MAPE values, ranging from 35.06% to 36.60%, elucidating that the enhancements implemented in the Python-enhanced models engender comparable error rates and accuracy levels across diverse roof dimensions. This uniformity underscores the robustness of the model's enhancements, which are instrumental for urban planners and architects who require precise data to guide urban layout and infrastructural designs. The consistent performance across various spatial dimensions is critical for ensuring reliable urban modelling and simulation.

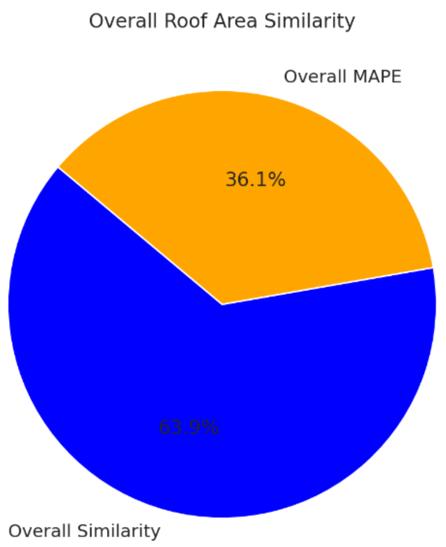

Figure 4 : Overall Similarity Percentage

The Overall Similarity and Error chart provides a visual representation of the comparative accuracy between the standard and Python-enhanced CityJSON models. The chart delineates an overall similarity of approximately 63.9%, indicating that, on average, the enhanced model maintains a substantial degree of fidelity to the standard model.

5. CONCLUSION

The exploration of Python-based and FME models for 3D urban modeling reveals nuanced distinctions in their approach to utilizing LiDAR data, impacting overall model accuracy and efficiency. Our analysis indicates that

Python-based models achieve a 63.9% similarity with standard 3D models. While this demonstrates a respectable degree of accuracy, it also highlights considerable scope for refinement, particularly in enhancing detail and precision. This is especially evident in larger models, where Python-based approaches face scaling challenges that could undermine the utility and application of the models in real-world scenarios.

A significant factor influencing model accuracy is the methodology employed in processing LiDAR data. Python-based models incorporate all points from LiDAR point clouds as vertices, aiming for a high level of detail. In contrast, FME models utilize a simplified vertex selection, which, while potentially reducing the richness of the model, streamlines processing and may enhance computational efficiency. This simplification aids in mitigating noise and lessening computational burdens, suggesting a strategic trade-off between model complexity and operational pragmatism.

Both modeling techniques are heavily dependent on the precision of LiDAR data. The dependency underscores the need for high-quality input data to achieve high fidelity in the resulting models. Improvements in Python-based models could be realized by integrating strategies similar to those used in FME, particularly concerning point selection. Simplifying the point selection process could not only align Python models more closely with the efficient processing characteristics of FME but also enhance their accuracy and applicability in complex urban planning tasks.

Moving forward, it is imperative to balance detail with efficiency in 3D modeling practices. Adopting a hybrid approach that incorporates the strengths of both Python and FME could potentially elevate the standard of urban modeling, making it more precise, reliable, and adaptable to varying scales and complexities.

6. REFERENCES

- *3D Geoinformation* (no date). Available at: https://www.tudelft.nl/en/architecture-and-the-built-environment/research/research-at-bk-bouwkunde/urbanism/3d-geoinformation (Accessed: 4 April 2024).
- Akca, D. *et al.* (2010) ‘Quality assessment of 3D building data’, *Photogrammetric Record*, 25(132), pp. 339–355. doi: 10.1111/j.1477-9730.2010.00598.x.
- Borkowski, A. and Jóźków, G. (2012) ‘Accuracy Assessment of Building Models Created From Laser Scanning Data’, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIX-B3(August 2012), pp. 253–258. doi: 10.5194/isprsarchives-xxxix-b3-253-2012.
- Cortés, I. M. (2023) ‘Open-source software for geospatial analysis’, *Nature Reviews Earth and Environment*, 4(3), p. 143. doi: 10.1038/s43017-023-00401-4.
- *Documentation | FME Community* (no date). Available at: https://support.safe.com/s/documentation (Accessed: 4 April 2024).
- Elberink, S. J. O. and Vosselman, G. (2007) ‘Quality analysis of 3d road reconstruction’, *ISPRS Workshop on Laser Scanning*, XXXVI(May), pp. 305–310.
- Hernandez-Garcia, D. E. *et al.* (2011) ‘3D city models: Mapping approach using LIDAR technology’, *CONIELECOMP 2011 - 21st International Conference on Electronics Communications and Computers, Proceedings*, pp. 206–211. doi: 10.1109/CONIELECOMP.2011.5749361.
- Khayyal, H. K., Zeidan, Z. M. and Beshr, A. A. A. (2022) ‘Creation and Spatial Analysis of 3D City Modeling based on GIS Data’, *Civil Engineering Journal (Iran)*, 8(1), pp. 105–123. doi: 10.28991/CEJ-2022-08-01-08.
- Ledoux, H. *et al.* (2019) ‘CityJSON: a compact and easy-to-use encoding of the CityGML data model’, *Open Geospatial Data, Software and Standards*, 4(1), pp. 1–19. doi: 10.1186/s40965-019-0064-0.
- Nys, G. A. and Billen, R. (2021) ‘From consistency to flexibility: A simplified database schema for the management of CityJSON 3D city models’, *Transactions in GIS*, 25(6), pp. 3048–3066. doi: 10.1111/tgis.12807.
- Nys, G. A., Poux, F. and Billen, R. (2020) ‘City json building generation from airborne LiDAR 3D point clouds’, *ISPRS International Journal of Geo-Information*, 9(9). doi: 10.3390/ijgi9090521.

- Pfeifer, N. *et al.* (2014) 'OPALS – A framework for Airborne Laser Scanning data analysis', *Computers, Environment and Urban Systems*, 45, pp. 125–136. doi: 10.1016/J.COMPENVURBSYS.2013.11.002.
- Shahidinejad, J., Kalantari, M. and Rajabifard, A. (2024) '3D Cadastral Database Systems—A Systematic Literature Review', *ISPRS International Journal of Geo-Information*, 13(1). doi: 10.3390/ijgi13010030.
- Wang, R. *et al.* (2020) 'CityJSON: a compact and easy-to-use encoding of the CityGML data model', *Transactions in GIS*, 11(2), pp. 1147–1164. doi: 10.1186/s40965-019-0064-0.