Automated Recognition of Structural Elements from Point Cloud Scans

Eliott SENTENAC
TELECOM Nancy
54600 Villers-lès-Nancy, France
eliott.sentenac@telecomnancy.eu

Alexandre PADELLINI
TELECOM Nancy
54600 Villers-lès-Nancy, France
alexandre.padellini@telecomnancy.eu

Guillaume HISLEUR
TELECOM Nancy
54600 Villers-lès-Nancy, France
guillaume.hisleur@telecomnancy.eu

Supervised by
Moufida Maimour
TELECOM Nancy
54600 Villers-lès-Nancy, France
Moufida.Maimour@telecomnancy.eu

***Abstract*—**
The automation of the Scan-to-BIM (Building Information Modeling) process has garnered increasing attention in recent years, yet fully autonomous systems remain out of reach. This paper presents a semi-automated 3D Scan-to-BIM pipeline capable of identifying and reconstructing basic structural elements, floors, ceilings, and walls, from raw point cloud data. The proposed method leverages Python and key libraries such as pye57 for reading .e57 files, Open3D for point cloud processing, and the Blender Python API for geometry reconstruction and .obj export. Although the number of elements to detect is currently user-defined, the process produces structurally coherent models with minimal human correction. Limitations include the lack of clustering, which can lead to redundant surfaces, and the absence of semantic classification, preventing direct .ifc generation. Despite these constraints, the system demonstrates reliable performance on the exemple dataset detailed in section IV and forms a strong foundation for further development. Future work aims to incorporate surface clustering, semantic tagging, and generalization beyond the Manhattan-world assumption to enable broader applicability and full automation of the Scan-to-BIM workflow.

I. Introduction

Currently, laser scanning technologies are capable of capturing detailed spatial data in the form of point clouds containing billions of points, commonly used for architectural and structural documentation. However, the raw point cloud data is inherently unstructured and must be converted into a structured 3D model to be useful in practical applications. This process is called Scan-to-BIM (Building Information Modeling), and although commercial software solutions such as AutoCAD exist for this purpose, they rely heavily on manual intervention, making the reconstruction process both time-consuming and repetitive. Consequently, there is a growing interest in automating this workflow, either partially or fully. This study focuses specifically on the automated detection and reconstruction of key structural components, namely, walls, floors, and ceilings. The objective of this research is to explore potential methods for automating the Scan-to-BIM workflow.

The remainder of this paper is organized as follows. Section II presents a review of existing solutions related to the automation of the Scan-to-BIM process. Section III details the methodology adopted in this study. Section IV applies the proposed approach to a case study involving the scan of a simple residential building. Finally, Section V discusses the implications of the results and outlines potential directions for future research.

II. State of the art

The automation of the Scan-to-BIM (Building Information Modeling) workflow has been the subject of extensive research in recent years. Despite notable progress, current methodologies have yet to achieve results that can fully replace manual intervention. Nonetheless, the field has seen substantial advancements. [1],[2]

Automation within this domain may be approached through either two-dimensional (2D) or three-dimensional (3D) methodologies. [3] In the present study, we focus exclusively on the 3D approach; therefore, the details pertaining to traditional 2D workflows are omitted.

The standard 3D Scan-to-BIM pipeline typically consists of several key stages:

Segmentation: This initial stage involves dividing the raw point cloud into smaller, more manageable regions for analysis. A common strategy, utilizes vertical histograms to segment the structure based on elevation levels. This method proves effective for most conventional building geometries.

Classification: Unique to 3D workflows, this stage assigns semantic labels to each segmented region, identifying them as specific structural elements such as floors, walls, or ceilings. This step is often accomplished using rule-based algorithms or machine learning techniques trained on labeled datasets.

Clustering: Following classification, the next step involves aggregating segmented parts that belong to the same structural component. Clustering aims to group geometrically and

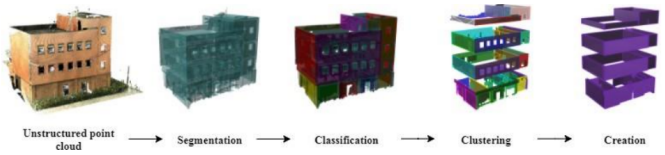

Fig. 1. Illustration of the four steps of the workflow.

semantically similar elements into unified objects. Numerous clustering techniques are available, each offering specific advantages and trade-offs in terms of accuracy, computational complexity, and robustness to noise. This step is particularly critical, as improper clustering may result in misrepresentation of the building's actual structure, propagating errors into the subsequent modeling phase.

Model Creation: In the final stage, a structured 3D model is generated from the clustered objects. This process often involves geometric simplification and parameterization, where certain object attributes, such as wall thickness, height, or orientation, are assumed to be constant to reduce computational complexity and improve robustness. These simplifications enable the construction of consistent and computationally tractable building models while preserving essential structural characteristics.

These four steps are illustrated in Fig. 1, which presents an example of the proposed automated workflow applied to a simple building scan. [3]

III. Methodology

This section presents the problem in detail and outlines the approach used to solve it. The objective is to create a 3D representation of the scanned structure without requiring human intervention, using the computational tool. We opted to the 3D scan-2-BIM method instead of the 2D approach as we believe the 3D method is more accurate and has more potential, even if it requires data acquired with more specific tools.

A. Analysis of the problem

The core challenge is to reconstruct a 3D model of a scanned structure without human input. This entails developing a tool capable of loading the point cloud representing the structure, identifying its components, such as walls and floors, and segmenting them prior to reconstruction.

Loading the point cloud is a non-trivial task. Most file formats used in this field are proprietary, making them difficult to process using external programming tools. Instead, field experts often rely on standard software packages, which offer limited flexibility for developing custom solutions. Additionally, the size of point cloud data, often amounting to billions of points, poses performance and memory challenges, necessitating efficient file handling and analysis methods.

The fundamental problem lies in producing a reconstruction that matches the quality of one created manually, without exceeding the time a human would take. The tool must be capable of recognizing and classifying architectural elements, grouping them into semantic components such as walls or floors, and reconstructing these into an accurate 3D model.

B. Proposed solution

Due to the inherent complexity of the problem domain, the proposed solution is limited to the reconstruction of structural elements such as floors, ceilings, and walls. Furthermore, this approach assumes that the buildings conform to the Manhattan world assumption, wherein most structural elements are aligned with three main coordinate axes.

To mitigate the computational and memory overhead associated with processing billions of 3D points, a uniform sampling strategy is employed. This approach allows for significant data reduction while preserving the structural integrity necessary for subsequent reconstruction steps. The input point clouds are provided in the E57 format, an open and extensible standard that facilitates interoperability and supports future research endeavors. Additionally, readily available Python libraries simplify the parsing and manipulation of E57 files.

Upon loading the point cloud, a voxel grid filtering is applied with a voxel size of 0.05 meters to further downsample the data. A Statistical Outlier Removal (SOR) algorithm is then used to eliminate points that deviate significantly from the local neighborhood distribution (parameters: nb_neighbors=20, std_ratio=2.0). Subsequently, surface normals are estimated for each point to facilitate the identification of planar regions with similar orientations, a computationally intensive step.

For floor detection, points with a surface normal whose z-component exceeds 0.999 are isolated. A RANdom SAmple Consensus (RANSAC) algorithm is then applied to extract candidate planes. Planes containing fewer than 200 points are discarded to avoid noise artifacts. For valid planes, a second SOR is applied before they are added to the list of detected floors. These points are then excluded from the dataset, and the process is repeated until no viable candidates remain. The final set of floor planes is sorted by point count, and the top n entries are retained. At present, the number of floors cannot be determined automatically (see Fig. 2). A bar chart representing the point density distribution along the Z-axis was generated to assess the feasibility of floor and ceiling detection based solely on vertical point concentration. As illustrated in the chart, the presence of significant noise and non-structural elements introduces substantial variability in point density across height levels. Consequently, it becomes impractical to reliably infer the number of floors or ceilings from this distribution alone. The lack of a clearly distinguishable threshold complicates the task of preserving all genuine structural layers while simultaneously eliminating false positives.

Wall detection proceeds by filtering points with surface normals whose z-component is less than 0.001. The algorithm then identifies directions aligned with potential wall orientations by evaluating the density of normals approximating the form $(\cos(\alpha), \sin(\alpha), 0)$, allowing for a tolerance of 0.1 degrees on α (see Fig. 3). This chart illustrates the distribution of surface normals with respect to their alignment to vectors

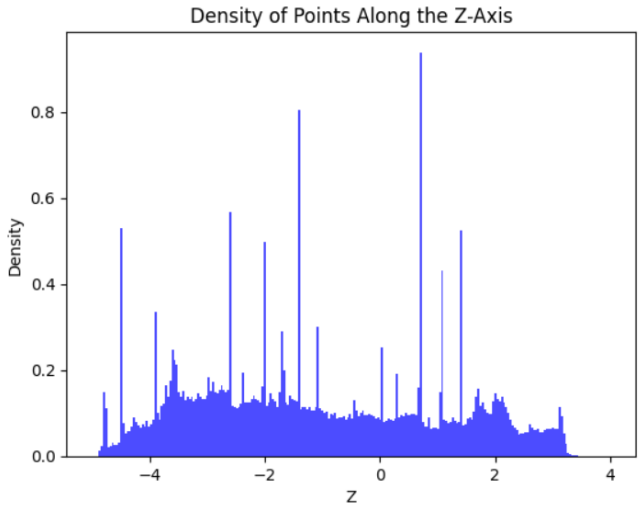

Fig. 2. Bar chart representing the point density distribution along the Z-axis.

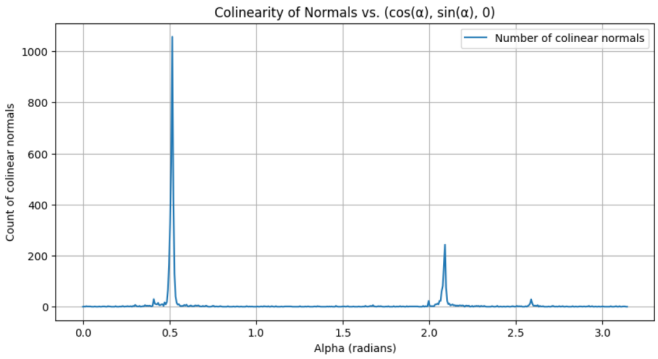

Fig. 3. Chart representing the distribution of surface normals in each direction.

across various angular directions (measured in radians). Despite the presence of noise, two prominent peaks are clearly distinguishable, indicating dominant orientations within the point cloud. These peaks correspond to the principal orthogonal directions of the building's structure and are subsequently selected as reference axes for the following wall detection and reconstruction procedures. The two dominant directions are selected, and the floor detection procedure is adapted to extract vertical surfaces corresponding to these orientations.

For 3D reconstruction, the Blender Python API (bpy) is utilized to export the generated geometry as .obj files. While Industry Foundation Classes (IFC) files would offer richer semantic information, their generation necessitates metadata beyond mere geometric shape, which was not available in this context. The reconstruction process involves computing the minimum-volume rectangular cuboid that encloses each identified point cluster using Open3D. The corner coordinates of each cuboid are then used with bpy to construct the corresponding 3D mesh, iteratively applied to all relevant clusters.

Starting from the initial volume computed for each plane, neither the extents nor the center of the volume are modified. Instead, the volume is rotated to align its widest face's normal vector with the Z-axis in the local reference frame defined by the two major vectors and the Z-axis (mv1, mv2, z). For horizontal elements such as floors, the best alignment is determined by comparing the similarity between the point cloud of the detected plane and the computed volume rotated by either 0° or 90° around the Z-axis. The same approach is applied for vertical elements such as walls, using the two major vectors, with appropriate volume rotations for alignment.

C. Implementation

The proposed pipeline was entirely implemented in Python. The pye57 library was utilized to read and parse .E57 point cloud files. For the majority of geometric and spatial computations performed on the point cloud data, the Open3D library was employed due to its efficient data structures and comprehensive support for point cloud processing. The final reconstruction phase leveraged the Python API of Blender (bpy) to generate and export 3D models. It is important to note that this implementation does not rely on scripting within the Blender application itself; instead, it uses the standalone Python API, thereby eliminating the need to install or launch Blender during execution.

Although several optimization strategies were incorporated, the overall pipeline remains computationally intensive. The full process, ranging from the point cloud initialization to the exportation of the reconstructed 3D geometry, typically requires about a minute to complete, not to mention reading the e57 input files, which can take several minutes on its own, depending mainly on the number of scans. Nevertheless, the system demonstrates robust performance in terms of memory management, with no observed issues related to dynamic memory consumption.

IV. Application to a simple building.

A. Simple building dataset

The proposed workflow was evaluated using a dataset consisting of a 3D scan of a simple residential house, comprising over 270 million points (see Fig. 4). It is important to note that, as illustrated in Fig. 4, the point cloud includes not only the structural elements of interest, but also various non-structural features such as vegetation, furniture, and miscellaneous objects. In the context of the current methodology, these elements act as noise and may negatively impact the accuracy of the detection process.

Moreover, the displayed point cloud does not encompass the entirety of the original dataset. The initial scan included an exterior garden area, which was excluded during preprocessing. This decision was made due to the irregularity of the terrain and the presence of numerous noise-inducing features, which would have introduced unnecessary complexity and degraded the quality of the structural analysis.

B. Results

At the current stage of development, the system is capable of detecting multiple floors and ceilings, as well as multiple

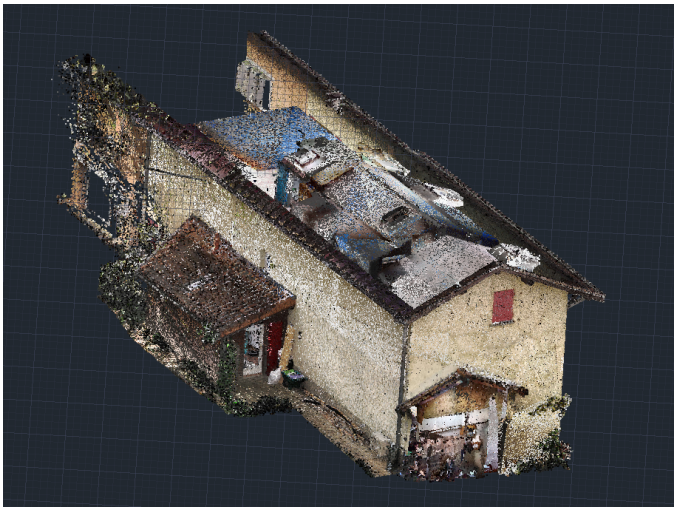

Fig. 4. Visualisation of the point cloud used.

Fig. 5. Display of the detected floors and ceilings

Fig. 6. Display of the detected walls in the direction of one of the two major vectors.

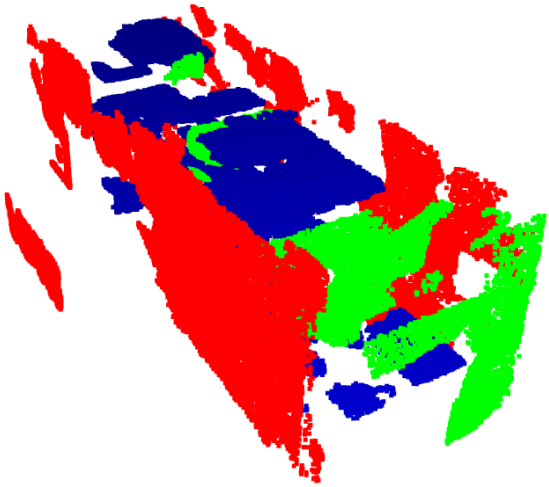

Fig. 7. Display of the detected planes.

walls aligned with the two dominant directions identified (see Fig. 5 and Fig. 6). Fig. 5 displays the detected floors and ceilings, where each colored point cloud represents one of the N selected surface segments. It is evident that certain point clouds, although assigned distinct colors, correspond to the same structural element. This redundancy arises from the absence of a clustering mechanism capable of merging geometrically similar segments that represent the same surface. As a result, single structural elements may remain split in the current implementation.

Similarly, Fig. 6 illustrates the detected walls aligned with one of the two principal directions identified through normal orientation analysis. The same detection process is applied along the second principal axis, enabling the identification of all major vertical structures under the Manhattan-world assumption. As with the floor and ceiling detection, the absence of a clustering step results in potential redundancy, where multiple segments representing parts of the same structural element are treated as distinct entities.

However, the number of structural elements to be extracted must be manually specified by the user, as there is no automated mechanism to infer the expected number of floors or walls.

Fig. 7 presents all the detected planar surfaces prior to the reconstruction phase, aggregated in a single view. While the general structure and outline of the house can be visually identified, several limitations persist at this stage. In particular, certain structural elements may remain difficult to interpret due to the presence of clutter introduced by small non-structural objects. Additionally, architectural features such as stairways and windows introduce discontinuities, manifested as holes, in the point clouds associated with structural surfaces, which further complicate visual interpretation and reconstruction.

The system is also capable of reconstructing the detected structural elements as 3D volumetric representations and exporting them in the standard .obj file format (see Fig. 8 and

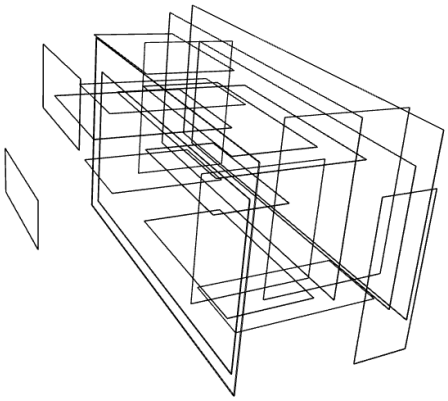

Fig. 8. Display of reconstructed planes.

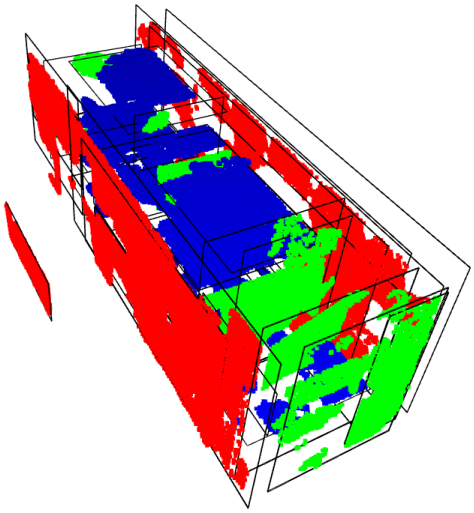

Fig. 9. Display of reconstructed planes alongside the corresponding point cloud.

Fig. 9).

Fig. 8 displays the reconstructed planar elements. Although the overall structure of the building may be difficult to discern due to the overlapping of multiple surfaces, certain positive indicators are evident. Notably, the reconstructed planes are correctly oriented, either perpendicular or parallel to one another, indicating that the resulting model adheres to the Manhattan-world assumption.

Fig. 9 provides a more informative visualization by overlaying the reconstructed planes with the corresponding point clouds for each structural element. This composite view facilitates a clearer understanding of the building's structure. The alignment between the point clouds and their associated reconstructed planes further validates the accuracy of the reconstruction process.

C. Current limits

Several limitations persist within the current implementation. Notably, certain surfaces are erroneously detected as distinct entities despite representing opposite sides of the same physical structure. This issue arises due to the bidirectional scanning of volumes, which leads to duplicate surface detection for geometrically continuous elements.

A potential solution to this problem would involve the integration of a clustering phase, whereby adjacent or parallel planar surfaces within a defined proximity are aggregated into a single volumetric representation. This would help mitigate redundancy and more accurately model the underlying building geometry.

Moreover, the reconstruction process remains relatively simplistic. Structural openings, such as doors, staircases or windows, are not identified and thus not reflected in the generated 3D models. Additionally, the presence of structural openings may influence the estimation of the minimum-volume bounding cuboid, resulting in slight deviations in the orientation or alignment of the reconstructed surfaces.

The reconstructed structural elements are currently limited to planar representations. While human input is presently required to assign standardized attributes, such as wall thickness, to each element type, this remains a practical and viable approach. Ideally, the integration of a clustering mechanism would enable the automatic estimation of such parameters. However, this enhancement introduces several challenges, including the risk of incomplete data within the scan and an increased likelihood of computational errors. Accurate clustering would necessitate the detection of a greater number of complementary planes per structural unit, thereby raising the complexity of the process and the probability of failure in the clustering phase.

Finally, while the system can successfully identify and reconstruct major structural surfaces, it does not perform semantic classification. As a result, the generation of IFC files is currently infeasible, given that IFC export requires not only geometric information but also semantic metadata describing the nature and role of each architectural element.

V. Conclusion

A. Summary

Overall, the designed process is capable of recognizing basic structural elements, such as floors, ceilings, and walls, from a point cloud representing a building. These identified elements can then be reconstructed into a 3D model. While the system is not yet perfect, it demonstrates a promising potential to significantly reduce the time required in the Scan-to-BIM workflow. Even though it currently focuses on simple structural features, it achieves a satisfactory level of accuracy, allowing for minimal human intervention in the form of fine adjustments. As such, while it does not yet fully automate the workflow, it constitutes a meaningful step toward complete automation.

Although the developed program may not be directly suitable for end users in its current form, it remains entirely

feasible to integrate it into a higher-level application. Such an interface could streamline the necessary human interactions and make the workflow more accessible and efficient for practical use.

B. Future prospects

If this work were to be further developed, several promising avenues of improvement could be explored.

A primary objective would be the classification of detected structural elements. Assigning semantic labels to each identified component would not only improve interpretability but also enable export in .IFC format, one of the most widely adopted standards in the field. Additionally, classification could aid other steps in the workflow, such as clustering, by helping distinguish between structurally and semantically distinct elements.

Another key enhancement would be the integration of a clustering step prior to reconstruction. Currently, some planes are treated as separate surfaces even though they belong to the same structural unit (e.g., opposite faces of a wall). Introducing clustering would allow for the aggregation of such planes, facilitating a more coherent model and enabling the automatic estimation of structural thickness.

Finally, while the proposed methods yield satisfactory results on the current dataset, they rely on the Manhattan-world assumption and struggle with noise introduced by non-structural elements. Therefore, exploring alternative or complementary methods, especially those that can generalize to more complex or irregular environments, would be a valuable direction for future research, potentially improving both robustness and applicability.

C. Discussion

Several alternative approaches were considered during the development process but ultimately not pursued due to practical limitations.

One such approach involved generating multiple segments between pairs of randomly selected points, then analyzing segment density within the spatial domain to infer the presence of planar surfaces. The underlying assumption was that higher segment density would correlate with the existence of structural elements aligned with specific planes. However, this method proved to be computationally intensive and was therefore deemed impractical for large-scale point clouds.

Another explored option was neighborhood-based surface reconstruction, in which local surface continuity is inferred by analyzing the spatial relationship between neighboring points. Despite its theoretical soundness, this method was also excluded due to excessive computational requirements that rendered it infeasible within the current system constraints.

Finally, a more straightforward method based on point density accumulation along predefined planesn, such as fixed z-levels for floors and ceilings, or axis-aligned vertical planes for walls-was evaluated. This technique was ultimately rejected due to its low accuracy; the presence of non-structural elements in the point cloud introduced significant noise, making it unreliable for accurate surface detection.

REFERENCES

[1] Maarten Bassier, Maarten Vergauwen *Topology Reconstruction of BIM Wall Objects from Point Cloud Data* , 2020

[2] Volk, R.; Stengel, J.; Schultmann, F. *Building Information Modeling (BIM) for existing buildings—Literature review and future needs.*, Autom. Constr., 38, 109–127, 2014.

[3] Maarten Bassier, Meisam Yousefzadeh, Maarten Vergauwen *COMPARISON OF 2D AND 3D WALL RECONSTRUCTION ALGORITHMS FROM POINT CLOUD DATA FOR AS-BUILT BIM*, Turk Ž, 2020