BIM Module for Deep Learning-driven parametric IFC reconstruction

Oscar Roman [1,2], Maarten Bassier [3], Sam De Geyter [3], Heinder De Winter[3], Elisa Mariarosaria Farella [2], Fabio Remondino [2]

[1] Department Information Engineering and Computer Science (IECS), University of Trento, Trento, Italy - oscar.roman@unitn.it

[2] 3D Optical Metrology (3DOM) unit, Bruno Kessler Foundation (FBK), Trento, Italy
web: http://3dom.fbk.eu – email: oroman@fbk.eu, elifarella@fbk.eu, remondino@fbk.eu

[3] Dept. of Civil Engineering, TC Construction - Geomatics, KU Leuven - Faculty of Engineering Technology, Ghent, Belgium
maarten.bassier@kuleuven.be, sam.degeyter@kuleuven.be, heinder.dewinter@kuleuven.be

Keywords: Automation in constructions, Deep-Learning, Scan-to-BIM, Semantic segmentation, Computational Geometry

Abstract

The creation of Building Information Models (BIM) is driven by cutting-edge software applications, plug-ins, and APIs that constitute the backbone of BIM authoring tools. While free tools and APIs offer visualization and customization options, geometric modelling remains largely restricted to interactive work and proprietary platforms, which sometimes limits flexibility and efficiency. There are still only a few comprehensive workflows that fully automate the reconstruction of building elements from reality-based surveyed data. This paper introduces an innovative reconstruction pipeline developed for the Scan-to-BIM Challenge at the CVPR 2024 Workshop, where it achieved second place in the competition. A deep learning (DL)-driven BIM Module for parametric IFC reconstruction is designed to accurately reconstruct both primary and secondary building elements within a BIM framework, starting from unstructured point cloud data captured via Terrestrial Laser Scanning (TLS). By leveraging DL techniques, particularly Convolutional Neural Networks (CNNs) and Transformers Networks (PTv3), our approach uses late fusion instance segmentation across both 2D and 3D modalities to accurately identify and reconstruct class-specific elements. The pipeline ultimately generates Industry Foundation Classes (IFC) elements, enhancing modelling accuracy, parameter estimation, and consistency in subsequent stages. Results highlight the pipeline's strong performance on various datasets, underscoring the crucial role of DL in advancing Scan-to-BIM workflows.

1. Introduction

The Scan-to-BIM concept has gained significant attention due to its increasing multi-sector application and interoperability, leading to the development of various BIM reconstruction methods from point cloud data, each utilizing different strategies (Bassier et al., 2016; Rashdi et al., 2022). Traditionally, the BIM industry has been dominated by proprietary authoring tools, which can sometimes hinder workflow efficiency for companies and engineering studios. As the demand for BIM solutions continues to rise, especially in Architecture, Engineering, and Construction (AEC) sector (Rocha and Mateus, 2021), there is a growing need for more efficient and open-source alternatives able to deliver comparable functionality and efficiency. Additionally, the growing focus on automating Scan-to-BIM processes has led to more efficient and advanced methods for reconstructing models directly from point cloud data.

Despite the AEC industry's growing demand for Scan-to-BIM procedures, challenges in automation persist. Key benchmarks on this topic, such as ISPRS and CVPR 2024, focus on complex indoor scenes using single-storey LiDAR or RGBD data, with strict requirements for Industry Foundation Classes (IFC) and evaluations based on completeness, correctness, and accuracy.

While comprehensive BIM benchmarks remain limited, advancements in machine learning (ML) and deep learning (DL) have significantly enhanced automation in Scan-to-BIM processes. Recent achievements in Scan-to-BIM automation leverage and integrate algorithms from computer vision and photogrammetry, such as octree and KD-tree spatial indexing for efficient data management and CNNs tailored for point-based data. These improvements have led to significant progress in semantic segmentation, instance segmentation, and class-specific reconstruction, enhancing both accuracy and automation.

1.1 Objectives

The presented work focuses on developing a comprehensive 3D building reconstruction pipeline from reality-captured data. The final goal is to enable a fully integrated pipeline that can accurately identify and reconstruct complex architectural elements within indoor spaces.

In the initial phases of the pipeline, object detection and semantic segmentation are employed to precisely identify and classify primary structural elements, such as walls, floors, and ceilings, along with secondary components, in particular doors.

The developed code and resources are available in the following GitHub repository: https://github.com/Saiga1105/Scan-to-BIM-CVPR-2024.

2. Background and related works

In recent years, various Scan-to-BIM approaches have been developed, exploring different workflows and techniques. Typically, all the methods presented in the literature consist of the following steps:

- *Semantic segmentation;*
- *Classification* and *instance segmentation*;
- the *final reconstruction phase*.

Each of these stages is essential for converting unstructured point clouds into structured BIM representations. The following sections offer a more technical breakdown of these key steps.

2.1 Segmentation and classification

Recent innovations in semantic segmentation have introduced transformative methods like Segment Anything Model (SAM, Kirillov et al., 2023), which excels in zero-shot segmentation with prompt-based inputs, and DINOv2 (Liu et al, 2023),

leveraging self-supervised learning for dense predictions with robust, transferable features.
K-Net (Zhang et al., 2021) represent a leap forward in object grouping, with transformer-based architectures providing better global context understanding and efficient segmentation. This approach highlights the growing trend of integrating transformers and self-supervised learning into segmentation tasks, significantly enhancing accuracy and adaptability across diverse applications.
While DL methods, including Convolutional Neural Networks (CNNs) and Graph Neural Networks (GNNs), especially those based on Point Transformers, have experienced groundbreaking advancements from (Zhao et al., 2021) to the cutting-edge innovations by (Wu et al., 2024), the Random Forest (RF) algorithm continues to stand out as a powerful tool in ML. As underscored by Grilli and Remondino (2020), the RF algorithm's robustness and effectiveness remain highly relevant, demonstrating its enduring value and versatility in the ever-evolving landscape of ML applications.

2.2 Instance segmentation

The most significant advancements have occurred in instance segmentation. Traditionally, this process relied on heuristic methods such as region growing (Yang et al., 2023b), connected components (Zhang et al., 2023), or scene graphs (Yang and al., 2023a). Recent progresses, however, leverage sophisticated algorithms like Mask R-CNN (Sujatha et al., 2023), which integrates region-based convolutional neural networks with object detection and segmentation.
In addition, end-to-end instance segmentation is achievable with some solutions like OneFormer3D, which learns instance queries directly from the data and achieves 70-80% of mean Average Precision (mAP), depending on the benchmark. This progress parallels advancements in computer vision, especially in 2D object detection, where techniques like Grounding DINO (Liu et al, 2023a) have significantly surpassed the state-of-the-art in 3D.

2.3 Reconstruction

Finally, reconstruction strategies can differ, exploring a wide range of approaches. This step incorporates RANSAC fitting (Hemmer, 2024), cell decompositions (Liu et al., 2023b), adjacency graphs (Damien et al., 2023), shape grammar (Stouffs, 2022, Yang et al., 2023c). End-to-end deep learning methods, such as Scan2BIM-NET (Perez-Perez et al., 2021), focus primarily on improving semantic segmentation. Recent methodologies, like NeRF-to-BIM (Hachisuka et al., 2023), aim mainly to reduce object occlusions, while BIM-Net++ (Campagnolo et al., 2023) specifically concentrates on enhancing object detection within the Scan-to-BIM process.
The most comprehensive and efficient reconstruction procedures have been presented by Bassier et al. (2020a), with a particular focus on walls reconstruction (Bassier et al., 2020b). These methods also incorporate graph-based pipelines, as detailed in (Bassier et al., 2024). Other methods employ statistical math pipelines based on Reversible Jump Markov Chain Monte Carlo (Tran and Khoshelham, 2020), or applying computational geometry through Topologic Maps (Roman et al., 2024), starting from edge extraction (Li et al., 2024).
Models derived with these techniques are primarily used for structural monitoring (Jiang et al., 2022), evaluating structural features of existing buildings (Özkan et al., 2024), and tracking progress on construction sites (Kim et al., 2020).

3. Methodology

This work introduces a BIM module for 3D reconstruction, designed to integrate with DL networks and create a fully automated, end-to-end BIM reconstruction pipeline. The module streamlines the process from raw data input to BIM model output, adhering to IFC standards to ensure interoperability and structured data organization. The current focus is on achieving consistent and detailed BIM reconstructions through a late fusion detection approach, which combines outputs from separate models at the final stage to enhance detection performance by leveraging diverse information sources, ultimately improving the subsequent reconstruction phase.

3.1 Datasets

The dataset used in this work is sourced from the IEEE/CVF CVPR 2024 Scan-to-BIM challenge (https://cv4aec.github.io/), comprising 3D building models from 16 floors across 8 buildings. Data are provided as point clouds in LAZ format files. Due to the high density and complexity of these point clouds (Table 1), the processing requires substantial computational power, including high-performance CPUs, large memory capacity, beyond the capabilities of a standard computer. For this reason, some pre-processing phases are required.

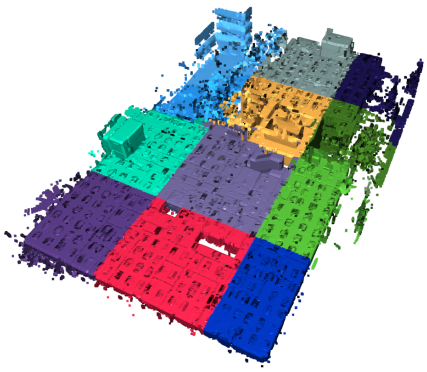


Figure 1. Partitioned point cloud.

3.2 Data pre-processing

The pre-processing steps start with the subsampling of the point cloud data to a resolution of 1 cm. As illustrated in Figure 1, the dataset has been partitioned into smaller subsets according to a pre-defined grid, ensuring more efficient processing while maintaining the integrity of the data.
Additionally, a training dataset for the instance segmentation task needs to be created by manually segmenting class elements within the point clouds. Finally, RDF graphs are generated, incorporating metric metadata for each point and linking these to corresponding BIM objects. This approach facilitates object tracking throughout the reconstruction process, following the Geomapi toolbox guidelines (Bassier et al., 2024).

3.3 Detection

In the detection phase, instance segmentation is performed for both primary structural classes (walls, ceilings, floors, columns) and secondary classes (doors).
The semantic segmentation is conducted using PTv3 (Wu et al., 2024; Zhao et al., 2021) and Pointcept (Section 2.1). Two scalar fields are assigned to the unstructured point clouds: first, a class label is assigned to each point from a total of seven categories (floors, ceilings, walls, columns, doors, and unassigned), while the second scalar field associates each ID to each labelled object.

The combination of PTv3 and Pointcept is highly effective for segmenting unstructured point clouds, accurately identifying, in particular, walls, ceilings, and floors.
For these categories, as evidenced by validation with training data, they yield high mean Intersection over Union (mIoU) (Table 2), while the results in other classes have been less reliable.

File name	Million points	Weight (MB)	RGB
05_MedOffice_01_F2	26,30	63.60	✗
19_MedOffice_07_F4	35,19	153.70	✓
32_ShortOffice_05_F1	26,75	118.85	✗
32_ShortOffice_05_F2	24,67	111.05	✓
35_Lab_02_F1	75,97	344.30	✗
35_Lab_02_F2	59,21	258.13	✗

Table 1. Characteristic of the partitioned input point cloud.

The results from semantic segmentation (PTv3) and instance segmentation (Pointcept) are stored in a JSON graph file. Each element is represented as a node, recording its geometric features and positions.

Class	Mean (%)	Std Dev (%)	Count parts	mIoU (%)	mAcc (%)
floors	12.5	4.8	41	92.5	95.0
ceilings	17.7	6.9	40	92.7	94.1
walls	30.9	8.3	40	82.9	85.8
columns	0.7	0.4	12	38.6	40.1
doors	0.9	0.5	39	42.9	58.5
unassigned	40.6	17.7	42	91.3	100.0

Table 2. Summary statistics and performance metrics for various classes.

This structure allows for precise oriented bounding boxes and spatial relationships (Figure 2), which can be enriched with additional data for further applications.

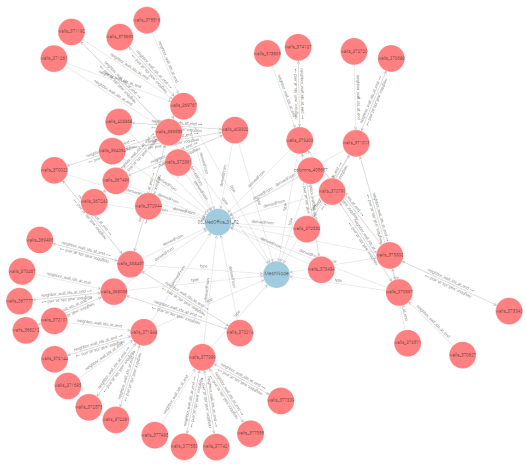


Figure 2. Metadata structure for building elements in graph node file.

3.3.1 Columns detection

Columns are often occluded and represent only 0.7 % of the scene (Table 2), leading to a stagnation in training results at 40 %. To address this, we applied a vision-based approach (Figure 3) for columns that capitalizes on the grid pattern of structural elements, as explained in Equation 1:

$$C_f = C_g + (C_v \cap G) \quad (1)$$

Where:

- C_f is the final cluster of columns;
- C_g is the columns cluster inferred by geometric features;
- C_v is the columns cluster inferred by visual-based approach;
- G is the grid defined by visual-based network.

We trained a YOLO v8 (Varghese and Sambath, 2024) model using eight distinct training datasets, optimizing for maximum recall. Subsequently, candidate columns were analysed in 3D, where they were evaluated based on detection scores, grid compliance, and point cloud signatures.

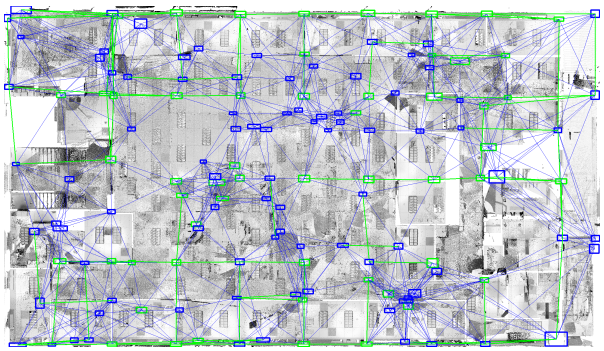


Figure 3. Visual-based detection approach for columns identification.

3.3.2 Openings detection

The door detection process (Figure 4) is applied after the wall reconstruction phase and leverages the GroundingDINO pre-trained model to identify doors with remarkable accuracy. The detection utilizes both the reconstructed meshes of walls, and the wall point cloud as inputs. The first input is used to compute the main axis of each wall, defined by the vector formed between the start and end points of the walls in the *xy* plane. From this axis, a 1-meter offset is applied to both sides, ensuring an accurate orthographic projection of the wall in 3D space. This projection, incorporating the point cloud as second input, is transformed into a raster image, where each unit is finely sampled at 0.01 pixels in both length and height. Rays are cast across the scene to detect visible surfaces, avoiding self-intersections, and generating orthographic images of the walls. Using these images, the GroundingDINO pre-trained model identifies potential door locations through predicted oriented bounding boxes. These detections are refined by applying thresholds based on real-world dimensions. In particular, let w represent the opening width of a door, and h represent the opening height of a door. The dimensional requirements for a valid door are specified in Equation 2:

$$t_{min\_width} = 0.50\ m \leq w \leq t_{max\_width} = 3.00\ m \quad (2)$$
$$t_{min\_height} = 1.50\ m \leq h \leq t_{max\_height} = 2.30\ m$$

If $\mathcal{L}_{\text{ref}}$ represents the reference level, defined as the distance between the floor and the opening, the condition for the topology and positioning of a door can be expressed as following (Eq. 3):

$$\mathrm{L}_{ref} < 0.50\ m \tag{3}$$

This ensures that the object is positioned close enough to the ground, satisfying the criteria for being identified as a door.

Figure 4. GroundingDINO Zero-shot object detection for openings recognition.

These additional filters help eliminate false positives, ensuring that only valid door candidates are retained. Once identified, the doors are reconstructed as 3D point clouds, allowing for precise extraction of their geometry and spatial positioning. The result is a comprehensive geometric representation of potential doors, ready for use in architectural restitution and analysis.

3.4 Multi-modal fusion results

After completing various object detection tasks, we fuse the results and update the RDF graph files to report the new data. This multi-modal approach optimizes detection accuracy across different data environments and scales. The detected objects are grouped into point clusters, which are enclosed within oriented bounding boxes, featured by metadata, including geometric properties, object IDs, locations, and orientations, and finally stored in a JSON graph file.

IfcBuildingElement	Parameters
IfcWallStandardCase	IfcLocalPlacement (p1, p2) Wall Thickness (m) base constraint (*URI*) base offset (m) top constraint (*URI*) top offset (m)
IfcColumn	IfcRectangleProfileDef.XDim (m) IfcRectangleProfileDef.YDim (m) IfcLocalPlacement (c) base constraint (*URI*) top constraint (*URI*)
IfcDoor	IfcLocalPlacement (p1, p2) IfcRectangleProfileDef.XDim (m) IfcRectangleProfileDef.YDim (m) base constraint (*URI*) top constraint (*URI*) top offset (m)

Table 3. IFC reconstruction parameters.

3.5 Reconstruction

Building on the advanced 2D and 3D reconstruction techniques outlined in recent studies (Bassier et al., 2020b; Bassier et al., 2024), this approach concurrently generates parametric BIM geometries along with their topological structures, ensuring full compliance with IFC standards. IFC objects are computed for each cluster C_i (see Table 3 for parameter details), detecting each identified object.

The BIM models are reconstructed in a hierarchical order, starting with the primary elements such as *IfcWallStandardCase, IfcColumn*, and *IfcSlab*, and then secondary elements, in particular *IfcDoor* elements. Secondary elements are closely linked to primary ones, as they are often embedded within them (e.g., doors within walls).

3.5.1 Walls reconstruction

The *IfcWallStandardCase* script initializes a set of nodes for walls ($\mathcal{W}$), ceilings ($\mathcal{C}$), and floors ($\mathcal{F}$) from the graph file. These nodes are fundamental to define the minimum (z_{base}) and maximum (z_{top}) z-coordinate value of each wall's vertical extent. The algorithm finds and links the nearest reference levels L_{top} and L_{base} for the top and base of the wall, in particular (Eq. 4):

$$L_{\text{top}} = \min_{L_i} \left| z_{\text{top}} - z_{L_i} \right| \quad \text{and} \quad L_{\text{base}} = \min_{L_i} \left| z_{\text{base}} - z_{L_i} \right| \tag{4}$$

Where z_{Li} denotes the height of the *i-th* reference level.

The topology of walls is then defined by performing a plane segmentation using RANSAC (Hemmer, M., 2024) to detect the dominant plane of the wall ($p_{//}$), as shown in Equation 5, and the wall's normal vector ($p_{\perp}$):

$$\mathrm{p} \parallel : ax + by + cz + d = 0 \tag{5}$$

The algorithm fits a plane to the set of 3D points $\boldsymbol{P}_i = (x_i, y_i, z_i)$, minimizing the error function (E), as in (Eq. 6):

$$\mathrm{E} = \sum_{i=1}^{N} \frac{(ax_i + by_i + cz_i + d)^2}{a^2 + b^2 + c^2} \tag{6}$$

Walls thinner than a specified threshold (t_{th} = 0.12 m) are not reconstructed, and the remaining are grouped into clusters *Ci* based on two criteria: distance and orientation. In particular, for distance (Eq. 7):

$$d(C_i, C_o) = \min_{P_i \in C_i, P_o \in C_o} |P_i - P_o| \leq t_d \tag{7}$$

And for orientation (Eq. 8):

$$\theta(C_i, C_o) = \arccos\left(\frac{|\overrightarrow{n_{C_i}}||\overrightarrow{n_{C_o}}|}{\overrightarrow{n_{C_i}} \cdot \overrightarrow{n_{C_o}}} \right) \leq t_o \tag{8}$$

Finally, analysing both equation we will merge clusters (Eq. 9):

$$\text{if } \mathrm{d}(C_i, C_o) \leq t_d \text{ and } \theta(C_i, C_o) \leq t_o. \tag{9}$$

Where:

- t_d is a distance threshold;
- t_o is a threshold for orientation similarity;
- C_i is the *i-th* cluster of a wal;
- C_o is the i-*th* cluster of a wall.

Thickness and orientation of walls are conditioned by both distance threshold t_d and orientation threshold t_o. All walls geometric features are computed using maximum orthogonal distance, normal vectors and recreate bounding box points, analysing also intersections between wall axes (Bassier and Vergauwen, 2020b). Finally, the script creates a visual representation of the wall using a *TriangleMesh* (Figure 7).

To compute the geometric features of walls, such as the curves representing the wall's start and end points, consider a wall node characterized by a normal vector n and a wall thickness t.
The orthogonal start point of the wall is determined by calculating $\boldsymbol{p_{ortho,\,start}} = p_{start} + \frac{t}{2}\boldsymbol{n}$ and $\boldsymbol{p_{ortho,\,end}} = p_{end} + \frac{t}{2}\boldsymbol{n}$.
Furthermore, for each wall node n, we collect the start and end points: $p_{axis,points} = \{p_{start}, p_{end}\}$, as well the corresponding orthogonal points, defined as $p_{otho,axis} = \{p_{ortho,start}, p_{ortho,end}\}$. These points are derived from the wall's geometry and serve as the basis for determining the relationships between walls. The nearest neighbour (NN) search is employed to find the closest walls within a threshold distance $t_{intersection}$. In particular, given two segment lines:

- Wall n, defined by p^n_{start}, p^n_{end},
- Wall w, defined by p^w_{start}, p^w_{end}.

the intersection point $p_{intersection}$ is found by solving the following system of linear equations, both in 2D or 3D (Eq. 10a, 10b):

$$p^n(t) = p^n_{\text{start}} + \text{t}(p^n_{\text{end}} - p^n_{\text{start}}) \quad (10a)$$
$$p^w(s) = p^w_{\text{start}} + \text{s}(p^w_{\text{end}} - p^w_{\text{start}}) \quad (10b)$$

The final intersection point is determined by solving the equation presented in Equation 11.

$$p^n_{\text{start}} + \text{t}(p^n_{\text{end}} - p^n_{\text{start}}) = p^w_{\text{start}} + \text{s}(p^w_{\text{end}} - p^w_{\text{start}}) \quad (11)$$

where t and s are determined from these equations.
The subsequent process verifies the accuracy of wall connections by identifying neighbour walls at each node's start and end points and determining which walls intersect or connect based on proximity and alignment. These connections are cross-referenced with ground truth (GT) data to evaluate overlap consistency, where overlap ratios measure the alignment between computed and GT neighbour IDs. To validate the accuracy of these computed connections, the algorithm calculates an overlap ratio O_n by comparing each node's computed neighbours $N_{c,n}$ with the ground truth (GT) neighbours $N_{gt,n}$. This ratio is defined by the Equation 12:

$$O_n = \frac{|N_{c,n} \cap N_{gt,n}|}{|N_{gt,n}|} \quad (12)$$

A high overlap ratio, close to 1, indicates that the computed wall connections closely reflect the GT structure, confirming accurate connectivity and structural continuity within the model. This validation step ensures that the model accurately represents real-world wall connections, which is essential for reconstruction fidelity.

3.5.2 Columns reconstructions

The *IfcColumn*, based on C_i column clusters, is reconstructed using a multi-step process to minimize noise and accurately restore rectangular columns. First, an oriented bounding box (OBB) is calculated, including noise points. Then, points in the C_i cluster are projected onto a plane at their average $\bar{z}$, effectively flattening them onto a plane. Furthermore, the bounding box is aligned along the x and y axes, through rotation matrix computation and projected at the average height $\bar{z}$ (Eq. 13):

$$P_i^{\text{proj}} = (x_i, y_i, \bar{z}) \quad (13)$$

In this 2D space, we applied a ConvexHull polygon (Eq. 14):

$$\mathcal{H}(\boldsymbol{P}) = \{p \mid p = \textstyle\sum_{i=1}^{N} \lambda_i p_i,\ \sum_{i=1}^{N} \lambda_i = 1,\quad \lambda_i \geq 0\} = \frac{|N_{c,n} \cap N_{gt,n}|}{|N_{gt,n}|} \quad (14)$$

where λ_i are the convex coefficients used to define the convex hull of the set of points $\boldsymbol{P}$.
To refine the bounding box, each boundary of the previous OBB is divided into intervals, and a Gaussian distribution is computed based on the distribution of points within each interval. The peak of these distributions determines the correct x and y coordinate for the minimum bounding box (mOBB).

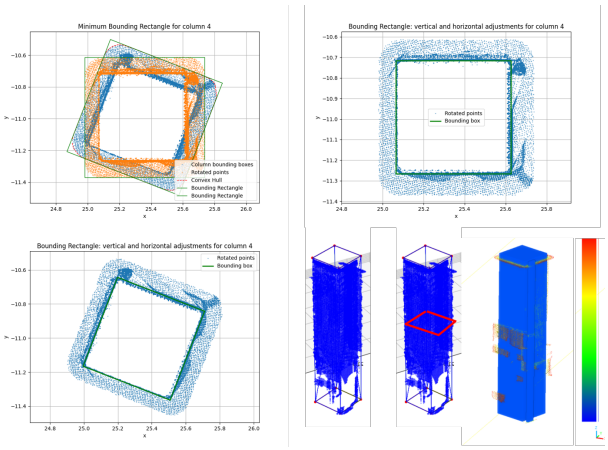


Figure 5. Operational steps for column reconstruction and generation of the mOBB.

This process (Figure 5) yields a more accurate, high-precision minimum oriented bounding box. The box is then returned to its original position (Figure 6) using the rotation matrix and translation (roto-translation). To determine the height of a column, the script follow the same one defined for the wall class. Additionally, to ensure structural integrity, columns with minimal size smaller than 0.15 m are excluded from reconstruction.
So shortly, *IfcColumn* reconstruction uses mOBB to derive rectangular columns, with parameters like width, length and height, and the centre is placed at the base centre.

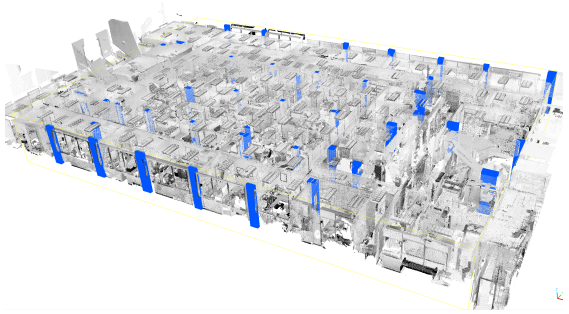


Figure 6. Positioning of columns in the geo-referenced point cloud.

3.5.3 Doors reconstruction

As previously mentioned, door reconstruction is derived from the reconstructed walls integrating the door. The door reconstruction process begins with the detection of potential door candidates using a combination of geometric features and image analysis based on walls nodes and walls point clouds (Figure 9). Typically, doors in a point cloud are represented by clusters of points outlining their structure; however, in some cases, a door

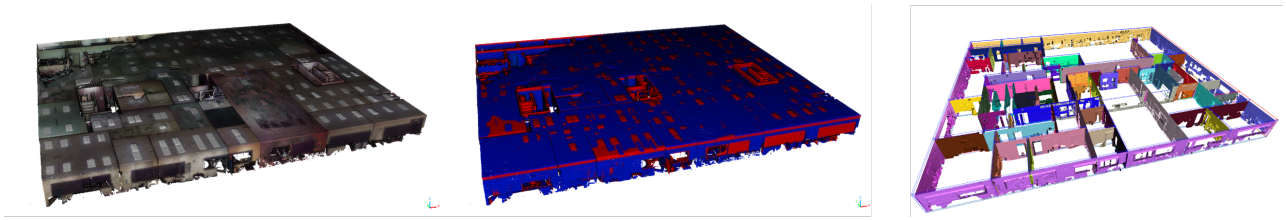


Figure 7. Reconstruction phases for the 35_Lab_02_F2 Dataset wall nodes.

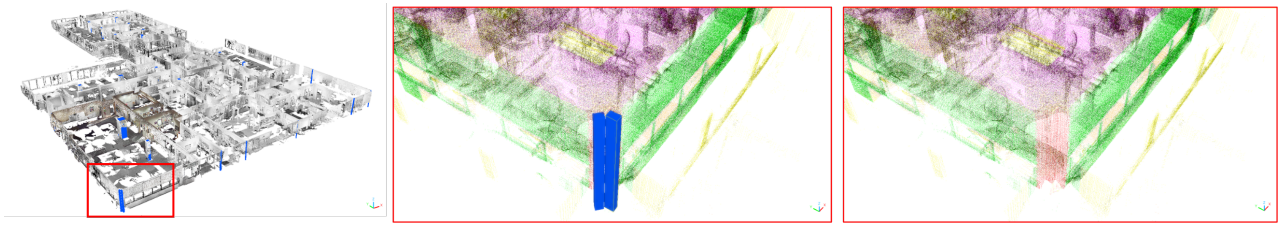


Figure 8. Reconstruction error: column inaccurately mapped within the model due to detection inaccuracies.

may instead be indicated by a gap or absence of points, where the empty space implies the presence of a doorway.

Once a door is identified, its 3D point cloud is generated by sampling the surface between its start point $P_{start} = (x_1, y_1, z_1)$ and end point $P_{end} = (x_2, y_2, z_2)$. This point cloud accurately captures the door's 3D geometry, assuming that the door thickness matches the thickness of the wall. Additionally, the axis of the door's bounding box is aligned with the wall's axis, ensuring that the door's orientation corresponds with the structural elements.

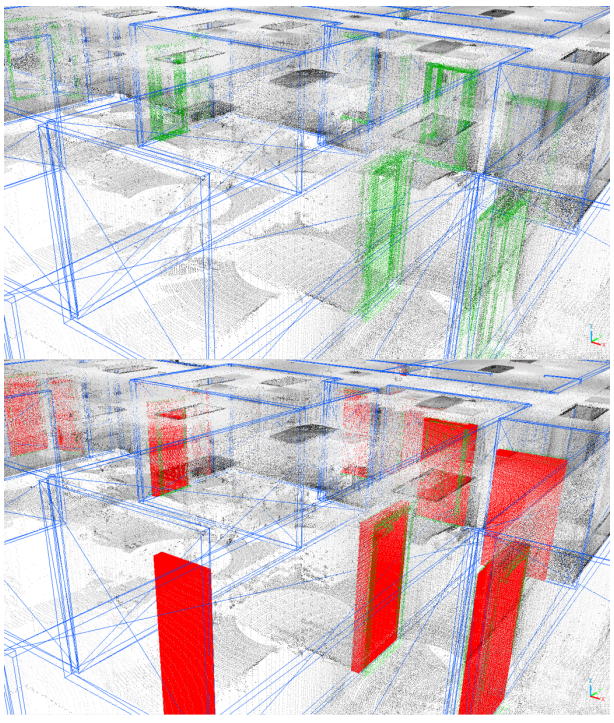


Figure 9. Door detection in the point cloud *(top)* and 3D door reconstruction *(bottom)*.

A comparison is made between the computed doors and the ground truth doors, focusing on geometric properties. The Euclidean distance between the center of the computed door and the ground truth door is calculated to measure positional accuracy. In addition to positional accuracy, the surface area of the door is also evaluated. The bounding box of the door is defined by its width w and height h, and the area A of the door is calculated as in Equation 15:

$$A = (w \times h) \tag{15}$$

Where:

- $w = |x_2 - x_1|$ is the width of the door (horizontal distance between start and end points),
- $h = |z_2 - z_1|$ is the height of the door (vertical distance).

By comparing the surface area and the positional distance d between the computed and ground truth doors, this process ensures the accurate reconstruction of doors in 3D space, validating both the topology and spatial positioning. These comparisons ensure that the detected doors conform to real-world dimensions and are correctly positioned within the reconstructed architectural model.

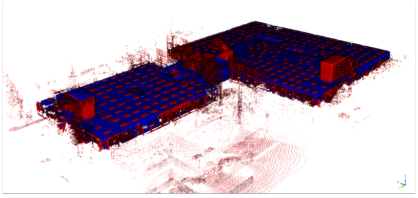
4. Results

In the detection phase, the combination of PTv3 and Pointcept in this multi-modal data processing strategy significantly enhanced performance, achieving a notable 86.1% F1-score at a 5 cm detection range and a mean Intersection over Union (mIoU) of 79.6%. Table 2 showed the results for the detection results.

The reconstruction process of the CVPR challenge datasets follows a hierarchical approach, starting from primary elements like walls, floors and columns, to secondary elements. The evaluation, based on a IoU parameter, has been focused on these elements, and concretely, walls are represented as centrelines with thickness to ensure clarity at intersections, while columns and doors are defined by their centre points, and extensions.

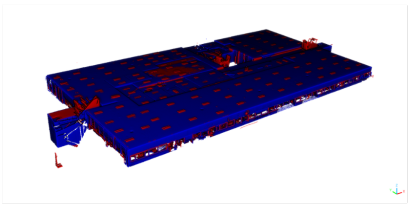
Table 4 summarizes the average reconstruction accuracy across all datasets, evaluated at 5 cm, 10 cm, and 20 cm thresholds, which represent the maximum allowable distances from ground truth points. Results (Figure 10) could be significantly improved, as PTv3 may perform better with additional training data, particularly for secondary building elements. As a result, the subsequent elements reconstruction may be more detailed and accurate.

Moreover, the results show that *32_ShortOffice* datasets, precisely the dataset *32_ShortOffice _05_F1* and the dataset *32_ShortOffice _05_F2,* consistently achieve the highest Precision, Recall, and F1 Scores, particularly at larger distances (20 cm), demonstrate the best overall performance.



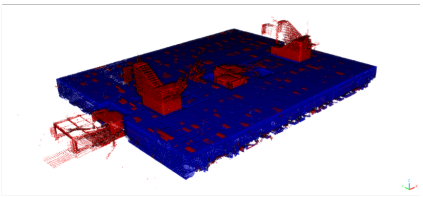
05_MedOffice_01_F2

Metric	5 cm	10 cm	20 cm
Precision Average	0.69	0.84	0.90
Recall Average	0.67	0.83	0.89
F1 Average	0.68	0.83	0.89
IoU Average	0.61	0.61	0.61



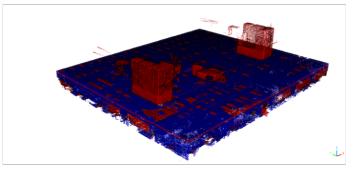
19_MedOffice_07_F4

Metric	5 cm	10 cm	20 cm
Precision Average	0.71	0.76	0.79
Recall Average	0.67	0.72	0.75
F1 Average	0.69	0.75	0.77
IoU Average	0.67	0.67	0.67



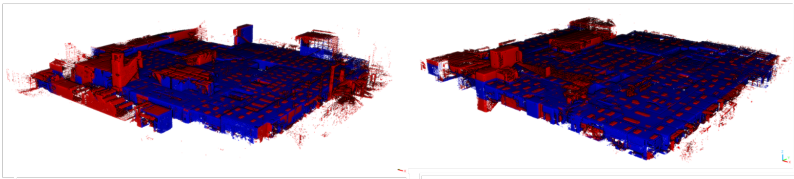
32_ShortOffice_05_F1

Metric	5 cm	10 cm	20 cm
Precision Average	0.65	0.76	0.80
Recall Average	0.72	0.84	0.88
F1 Average	0.68	0.79	0.83
IoU Average	0.66	0.66	0.66



32_ShortOffice_05_F2

Metric	5 cm	10 cm	20 cm
Precision Average	0.65	0.76	0.88
Recall Average	0.72	0.84	0.88
F1 Average	0.68	0.79	0.83
IoU Average	0.66	0.66	0.66



35_Lab_02_F1

Metric	5 cm	10 cm	20 cm
Precision Average	0.66	0.73	0.78
Recall Average	0.69	0.77	0.83
F1 Average	0.67	0.74	0.80
IoU Average	0.58	0.58	0.58

35_Lab_02_F2

Metric	5 cm	10 cm	20 cm
Precision Average	0.61	0.64	0.66
Recall Average	0.48	0.51	0.53
F1 Average	0.47	0.50	0.52
IoU Average	0.43	0.43	0.43

Figure 10. Complete results and metrics for all the datasets of the CVPR challenge and for the three thresholds (5 cm, 10 cm, 20 cm).

The *35_Lab_02_F1* also performs better than *35_Lab_02_F2*, with higher Precision and F1 Scores, while *MedOffice* datasets show improvement in metrics with increased distance. However, all datasets maintain quite consistent IoU values, indicating stable overlap performance but with varying effectiveness in other metrics.

Despite achieving high F1-scores for walls and columns, challenges remain with small variations impacting metrics such as mIoU, especially for walls due to their single-faced nature and scanning limitations, and for columns, which are often occluded during the surveying phase.

Threshold (cm)	Columns (%)		Doors (%)		Walls (%)	
0.05	mIoU:	80.6	mIoU:	39.6	mIoU:	84.3
	F1:	81.4	F1:	41.5	F1:	81.3
0.10	mIoU:	88.7	mIoU:	55.4	mIoU:	91.5
	F1:	89.5	F1:	58.0	F1:	88.3
0.20	mIoU:	91.7	mIoU:	60.1	mIoU:	97.2
	F1:	92.5	F1:	63.1	F1:	93.9
Overall Average	**mIoU:**	**72.9**	**mIoU:**	**53.2**	**mIoU:**	**67.8**
	F1:	**87.8**	**F1:**	**54.2**	**F1:**	**87.8**

Table 4. Results for columns, walls and doors of the CVPR datasets, for three different thresholds (5 cm, 10 cm and 20 cm).

5. Conclusions

This research presents an advanced BIM reconstruction framework using DL techniques, achieving significant gains in segmentation accuracy for point cloud data with an F1-score of 86.1% at a 5 cm detection range and a mIoU of 79.6%. Employing PTv3 and Pointcept within a late fusion framework, the multi-modal pipeline accurately reconstructs primary elements like walls and floors and secondary elements like doors, aligning with IFC standards. Wall detection accuracy reached 82.9%, while occluded elements such as columns and doors posed more significant challenges, with mIoU of 38.6% and 58.5% and a lower recall for doors (37%).

These results highlight areas for improvement, particularly in occlusion handling, though the pipeline maintained an average mIoU of 67.8% across all elements.

Future work will focus on improving segmentation accuracy, as it directly impacts the reconstruction phase (see Figure 8), and can sometimes lead to incorrect reconstructions. Furthermore, the workflow will incorporate additional secondary features such as windows and enhance algorithms to more effectively mitigate noise and occlusions in point cloud data. By leveraging DL approaches, this will advance Scan-to-BIM automation within the AEC industry, in accordance with industry standards.

References

Bassier, M., Hadjidemetriou, G., Vergauwen, M., Van Roy, N., Verstrynge, E., 2016. Implementation of Scan-to-BIM and FEM for the Documentation and Analysis of Heritage Timber Roof Structures. *Proc. Int. Euro-Mediterranean Conf.*

Bassier, M., Vergauwen, M., Poux, F., 2020a. Point Cloud vs. Mesh Features for Building Interior Classification. *Remote Sens., 12, 2224.*

Bassier, M., Vergauwen, M., 2020b. Unsupervised re-construction of building information modeling wall objects from point cloud data. *Automation in Construction, 120, 1–20.*

Bassier, M., Vermandere, J., De Geyter, S., De Winter, H., 2024. GEOMAPI: Processing close-range sensing data of construction scenes with semantic web technologies, *Automation in construction, 164, 105454.*

Campagnolo, D., Camuffo, E., Michieli, U., Borin, P., Milani, S., Giordano, A., 2023. Fully Automated Scan-to-BIM via Point Cloud Instance Segmentation. *Proc. IEEE Int. Conf. Image Process. (ICIP)*, 291–295.

Damien, R., Hugo, R., Loic, L., 2023. Efficient 3D Semantic Segmentation with Superpoint Transformer. *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 17195–17204.

Grilli, E., Remondino, F., 2020.Machine Learning Generalisation across Different 3D Architectural Heritage. *ISPRS Int. J. Geo-Inf., 9(6), 379.*

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W., Doll, P., Girshick, R.B., 2023. Segment Anything. *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 3992–4003.*

Hachisuka, S., Tono, A., Fisher, M., 2023. Harbingers of NeRF-to-BIM: A Case Study of Semantic Segmentation on Building Structure with Neural Radiance Fields. *Proc. Eur. Conf. Comput. Constr.*

Hemmer, M., 2024. Algebraic Foundations. *CGAL User and Reference Manual.* CGAL Editorial Board, 5.6.1 edition.

Jiang, Z., Shen, X., Ibrahimkhil, M. H., Barati, K., Linke, J., 2022. Scan-vs-BIM for real-time progress monitoring of bridge construction project. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci., X-4/W3-2022.*

Kim, S., Kim, S., Lee, D.-E, 2020. Sustainable Application of Hybrid Point Cloud and BIM Method for Tracking Construction Progress. *Sustainability, 12, 4106.*

Li Q., Xu, Z., Bai, S., Nie, W., Liu, A., 2024. Multi-modal fusion network guided by prior knowledge for 3D CAD model recognition. *Neurocomputing, 590.*

Liu, S., Zeng, Z., Ren, T., Li, F., Zhang, H., Yang, J., Li, C., Yang, J., Su, H., Zhu, J. and Zhang, L., 2023a. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499.*

Liu, L., Wang, X., Yang, X., Liu, H., Li, J., Wang, P., 2023b. Path planning techniques for mobile robots: Review and prospect. *Expert Syst. Appl.*

Özkan T., Pfeifer N. and Hochreiner G., 2024. Automatic completion of geometric models from point clouds for analyzing historic timber roof structures. *Front. Built Environ., Sec. Construction Management, 10.*

Perez-Perez, Y., Golparvar-Fard, M., El-Rayes, K., 2021. Scan2BIM-NET: Deep Learning Method for Segmentation of Point Clouds for Scan-to-BIM. *J. Constr. Eng. Manage., 147(9), 04021107.*

Rashdi, R., Martínez-Sánchez, J., Arias, P., Qiu, Z., 2022. Scanning Technologies to Building Information Modelling: A Review. *Infrastructures, 7, 49.*

Rocha, G., Mateus, L., 2021. A Survey of Scan-to-BIM Practices in the AEC Industry—A Quantitative Analysis. *ISPRS Int. J. Geo-Inf., 10, 564.*

Roman, O., Mazzacca, G., Farella, E.M., Remondino, F., Bassier, M., Agugiaro, G., 2024. Towards Automated BIM and BEM Model Generation Using a B-Rep-Based Method with Topological Map. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*

Stouffs, R., 2022. A Multi-Formalism Shape Grammar Interpreter. *Proc. CAAD Futures 2021: Commun. Comput. Inf. Sci.*, 1465. Springer, Singapore.

Sujatha, K., Amrutha, K., Veeranjaneyulu, N., 2023. Enhancing Object Detection with Mask R-CNN: A Deep Learning Perspective. *Proc. Int. Conf. Netw. Multimedia Inf. Technol. (NMITCON), 1–6.*

Tran, H., Khoshelham, K., 2020. Procedural Reconstruction of 3D Indoor Models from LiDAR Data Using Reversible Jump Markov Chain Monte Carlo. *Remote Sens.*, 12, 838.

Varghese, R., Sambath, M., 2024. YOLOv8: A Novel Object Detection Algorithm with Enhanced Performance and Robustness. *Proc. Int. Conf. Adv. Data Eng. Intell. Comput. Syst. (ADICS).*

Wu, X., Jiang, L., Wang, P.-S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., Zhao, H., 2024. Point Transformer V3: Simpler, Faster, Stronger. *arXiv preprint*, arXiv:2312.10035.

Yang, J., Wang, C., Liu, Z., Wu, J., Wang, D., Yang, L., Cao, X., 2023a. Focusing on flexible masks: A novel framework for panoptic scene graph generation with relation constraints. *ACM Multimed., 4209–4215.*

Yang, S., Hou, M., Li, S., 2023b. Three-Dimensional Point Cloud Semantic Segmentation for Cultural Heritage: A Comprehensive Review. *Remote Sensing.*

Yang, L., Li, J., Chang, H.-T., Zhao, Z., Ma, H., Zhou, L., 2023c. A Generative Urban Space Design Method Based on Shape Grammar and Urban Induction Patterns. *Land, 12, 1167.*

Zhang, W., Pang, J., Chen, K., Change, C., 2021. K-Net: Towards Unified Image Segmentation. *arXiv preprint, arXiv:2106.14855.*

Zhang, W., Joseph, J., Yin, Y., Xie, L., Furuhata, T., Yamakawa, S., Shimada, K., Burak Kara, L., 2023. Component segmentation of engineering drawings using Graph Convolutional Networks. *Comput. Ind., 147.*

Zhao, H., Jiang, L., Jia, J., Torr, P., Koltun, V., 2021. Point Transformer. *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), 16259–16268.*