Article | Received 19 September 2025; Revised 11 December 2025; Accepted 29 January 2026; Published 30 January 2026
https://doi.org/10.55092/sc20260003

Automated BIM modelling from non-digital engineering drawings using improved YOLOv11n-based detection framework

Xinjiang Cai[1,2,3], Jinchang Deng[1], Yong Zhu[1,2,*], Baocheng Zhao[1,2] and Xiaoyong Mao[1,4]

[1] School of Civil Engineering, Suzhou University of Science and Technology, Suzhou 215011, China
[2] Key Laboratory of Multi-Disaster Safety Prevention and Control in Civil Engineering at Provincial Universities, Suzhou University of Science and Technology, Suzhou 215011, China
[3] Advanced Perception and Intelligent Equipment Engineering Research Center of Jiangsu Province, Suzhou City University, Suzhou 215204, China
[4] School of Intelligent Manufacturing and Smart Transportation, Suzhou City University, Suzhou 215204, China

* Correspondence author; E-mail: zhuyong@usts.edu.cn.

Highlights:

- Lightweight DBAL-YOLO for efficient multi-scale detection on scanned drawings.
- Module-based modulus correction improves geometric accuracy of detections.
- Automated 3D BIM generation from detections and segmentations for digital twins.

Abstract: Digital twin city systems are key foundations for intelligent urban management and real-time resilience monitoring, consisting of digital models of buildings, roads and functional units. Efficient 3D modelling technologies for these units—especially existing buildings—are critical for these systems. However, due to a lack of 3D digital models for many old buildings, digital reconstruction relies on scanned paper drawings. Thus, this work develops an efficient structural member detection and rapid 3D reconstruction approach. An improved lightweight YOLOv11n algorithm (DBAL-YOLO) enables efficient and accurate detection of main structural elements (columns and beams) in floor plans, while a parameter-optimised U-Net model achieves pixel-level segmentation of architectural walls in floor plans. By integrating 2D geometric parameter extraction with linear extrusion reconstruction (via architectural modulus check and self-correction), the method enables automated 3D generation of component geometries and their spatial topological relationships. Tests on 3,960 annotated single-story drawings demonstrate that DBAL-YOLO achieves a high precision of 98.8% and an excellent recall of 98.3%, along with notable improvements in computational efficiency. The optimised U-Net yields a Mean Pixel Accuracy (mPA) of 96.4% and a Mean Intersection over Union (mIoU) of 98.8%. Further validation via a five-story building modelling case confirms the proposed approach's capability to efficiently realise rapid 3D modelling of buildings using scanned paper engineering drawings.

Keywords: building information modelling; deep learning; image detection; 3D physical modelling

1. Introduction

For the intelligent urban management and resilience monitoring against various events in modern cities, digital twin city systems are one of the most important information infrastructures [1,2]. These digital twin city systems consist of numerous digital entities, such as buildings, roads and a variety of functional units. The rapid establishment and continuous updating of urban digital twin systems fundamentally depend on the efficient digital reconstruction of these physical entities in large amounts, and, therefore, the efficient 3D digital model generation technology for these units, particularly the existing buildings with 3D features, is determined as the main focus of this work.

Reconstructing Building Information Modelling (BIM) based on 2D drawings serves as a critical approach for the digital management of existing buildings, while simultaneously constituting a primary data source for City Information Modelling (CIM) development [3]. However, due to the absence of digital drawings/documents available for many old buildings, digital reconstruction has to rely on the scanned copies of the archived paper drawings. Traditional manual recognition and modelling modes often require extensive time and large labour investment, and may fail to meet the demands for efficiency, or even accuracy due to human error, in these 3D modelling procedures. Thus, integrated workflow approaches involving the automatic recognition from the non-digital engineering drawings (especially the scanned copies of the drawings), check and self-correction, and 3D modelling of existing drawings are promising solutions for the rapid construction of digital twin city systems.

For target detection and recognition, the You Only Look Once (YOLO) model proposed by Joseph Redmon *et al.* [4] transforms the classification problem into a regression task and achieves end-to-end optimisation. Schönfelder *et al.* [5] developed a robust text detection and recognition pipeline for floor plans by leveraging a domain-specific synthetic data generation strategy and a model comparison study, which identified YOLOv7 and PARSEq as the optimal combination for this task, demonstrating significantly superior performance compared to general-purpose OCR methods. A deep floor plan recognition method based on a multi-task neural network with a room-boundary-guided attention mechanism was proposed by Zeng *et al.* [6]. This approach achieves pixel-wise simultaneous prediction of walls, doors, windows, and multi-category rooms, while balancing training through cross-task and intra-task weighted loss functions. Zhu *et al.* [7] developed an enhanced damage detection module by integrating DyHead and KernelWarehouse into YOLOv8, achieving a 4.48% improvement in mAP with the optimised KernelWarehouse module. Xu *et al.* [8] proposed ArchNetv2, a convolutional neural network (CNN) based architecture, which attained 93.5% mAP in identifying 13 common object categories within architectural floor plans during testing. The enhanced Faster R-CNN framework developed by Zhou *et al.* [9] enables automated extraction of architectural component information from 2D drawings, thereby providing critical data support for subsequent automated BIM generation. This optimised algorithm achieves a mean Average Precision (mAP) of 93.8% in architectural component recognition. Lu *et al.* [10] introduced a semi-automatic recognition method for CAD drawings, which combines Optical Character Recognition (OCR) technology to identify special symbols in floor plans for locating structural components, with a MATLAB-developed algorithm to extract component information. The approach's effectiveness was ultimately validated through case studies.

Despite the effectiveness of existing approaches in detecting architectural components, most of these methods adopt generic convolutional structures and feature fusion strategies that are not specifically optimised for scanned engineering drawings. As a result, they often struggle with slender structural elements and complex multi-scale layouts, while relying on computationally heavy architectures. Such high model complexity limits their practicality for large-scale, on-the-fly processing in real-world BIM workflows, where efficiency and deployability are critical. Moreover, prior works largely focus on detection accuracy and provide limited support for generating geometrically consistent, BIM-ready representations.

Recent Plan-to-BIM research has made significant progress in data enrichment, semantic modelling, and automated reconstruction. Schönfelder *et al.* [11,12] proposed a Drawing Analysis Ontology using RDF/SPARQL to encode semantic relations among plan elements, enabling consistency checks for automated BIM generation. Brauksiepe *et al.* [13] synthesised annotated floor-plan images to alleviate limited training data for deep-learning workflows. Urbieta *et al.* [14] applied Mask R-CNN to segment structural elements from CAD plans and integrated IfcOpenShell scripts to generate multi-story BIM models, while Yang *et al.* [15] proposed a semi-automated Revit/Dynamo workflow for extracting structural components. Bacharidis *et al.* [16] combined Pix2Pix segmentation with single-image depth prediction to reconstruct building façades, and Guo *et al.* [17] enhanced point-cloud-based 3D reconstruction with improved segmentation accuracy.

While these studies make valuable contributions, their limitations are complementary: some rely on complex networks or proprietary tools [14,15], while others focus on specific building aspects or incur high computational costs [16,17]. Few methods provide a lightweight, fully automated pipeline for structural component recognition directly from CAD drawings without external plugins or API interfaces.

It is crucial to clarify that this research focuses on processing two specific types of engineering drawings: structural floor plans and architectural floor plans. Structural plans provide detailed information about load-bearing elements (columns, beams, slabs), while architectural plans contain layout information, including walls, rooms, and spatial organisation. The framework of this work is designed to leverage the complementary information from both drawing types to generate integrated 3D models that contain both structural and architectural components.

To address the above challenges, DBAL-YOLO, a deep-learning–based, lightweight algorithm for CAD-based structural component recognition and rapid 3D model generation was developed. DBAL-YOLO extends the YOLOv11n framework and is specifically designed for multi-scale component detection in architectural drawings. The approach integrates reconstructed multi-scale feature pyramids with dynamic convolution kernels and attention mechanisms to enhance robustness across varying component scales. By combining U-Net segmentation with computer vision–based parametric modelling, the system enables rapid, fully automatic reconstruction of 3D geometric parameters and spatial topologies of structural elements. This plugin-free, self-contained framework supports efficient construction, updating, and management of digital twin systems.

2. Developed DBAL-YOLO model and 3D model reconstruction

An overview of the proposed DBAL-YOLO pipeline is presented in Figure 1. The system accepts scanned structural and architectural floor plans together with other engineering parameter drawings as inputs. After preprocessing (denoising, colour conversion and OCR), three parallel branches extract

complementary information: lightweight multi-scale detection (DBAL-YOLO), pixel-level wall segmentation (U-Net), and OCR-based parameter extraction. The outputs of these branches are fused and corrected by a parameter-fusion and module-based correction stage, which yields engineering-consistent geometric parameters for the subsequent 3D reconstruction (parametric extrusion and mesh generation). Finally, reconstructed models undergo validation and post-processing (mesh refinement and geometry/topology checks), and export in standard formats.

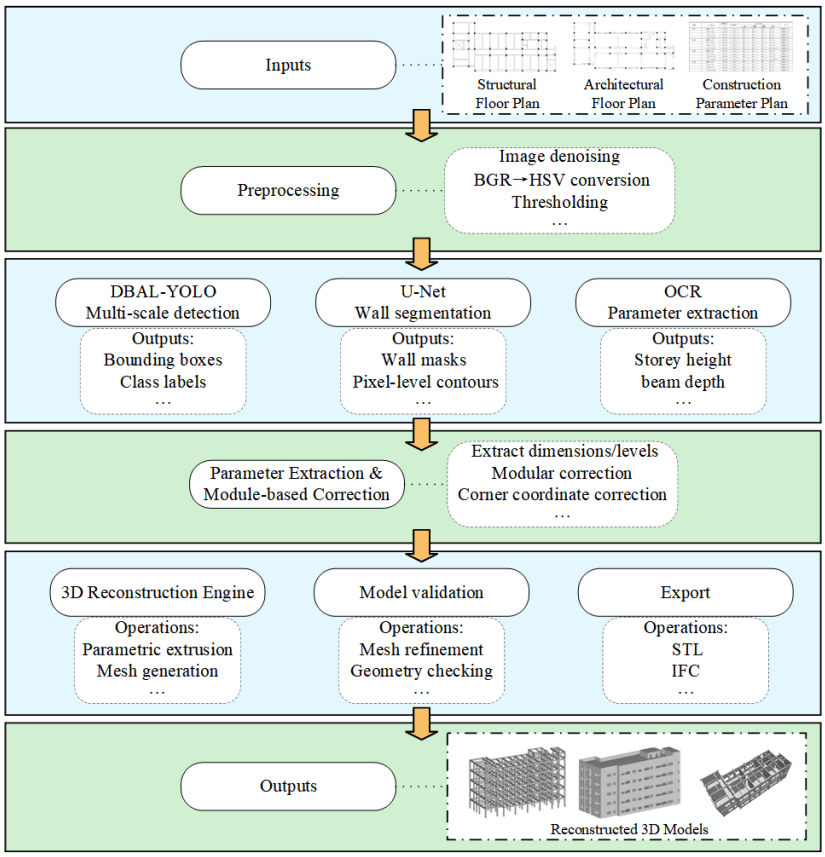


Figure 1. Overall framework of the proposed method.

2.1. Brief introduction for the YOLOv11n network

YOLO [4] is a single-stage detector known for real-time efficiency, consisting of a backbone for feature extraction, a neck for multi-scale fusion, and a head for localisation and classification. As the lightweight model of the YOLOv11 series, YOLOv11n reduces parameters (2.6 M) and computation (6.6 GFLOPs) through depth-wise separable convolutions and channel pruning, while incorporating an SPPF module within a streamlined CSPDarknet backbone for efficient edge deployment. The network architecture is shown in Figure 2.

In the backbone, YOLOv11n adopts a lightweight feature-extraction pipeline composed of C3K2, SPPF, and a self-attention-enhanced C2PSA block. Compared with YOLOv8n, the insertion of C2PSA after SPPF strengthens long-range dependency modelling with minimal computational overhead. The C3K2 module replaces the standard C2f structure and introduces a multi-branch design that combines a shallow convolution path with a deeper C3K/Bottleneck path, enabling more effective multi-scale feature capture. Additionally, depth-wise separable convolutions are used in the detection head to further

reduce parameters and computational cost, allowing YOLOv11n to maintain strong detection accuracy while remaining suitable for lightweight deployment. In the detection head's classification branch, inspired by MobileNets [18], certain standard convolutions are swapped for depth-wise separable convolutions, further reducing parameters and FLOPs.

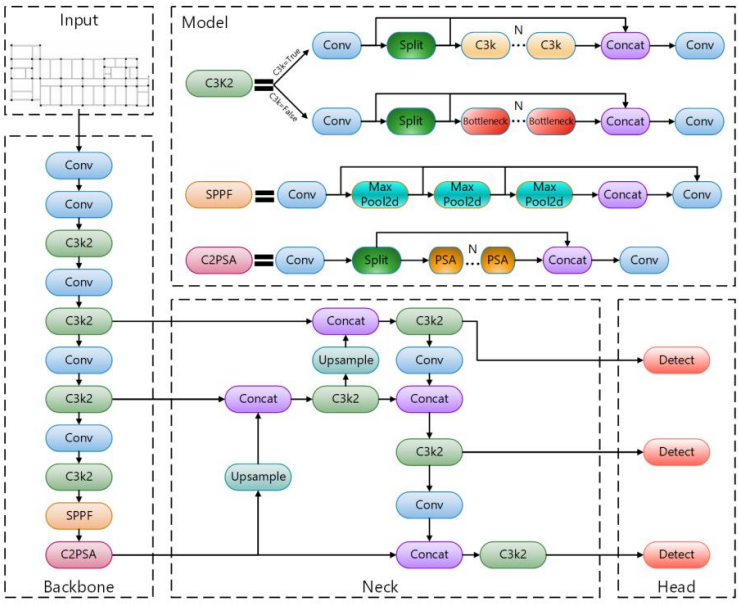


Figure 2. YOLOv11n network structure.

2.2. DBAL-YOLO network

The overall architecture of DBAL-YOLO is shown in Figure 3. The design follows a lightweight, task-oriented principle tailored to scanned CAD-based Plan-to-BIM scenarios, where structural components are typically slender, multi-scale, and embedded in dense linework. Accordingly, the backbone, neck, attention mechanism, and detection head are configured to balance directional feature sensitivity, cross-scale information propagation, and computational efficiency. This architecture provides the structural basis for robust component recognition while maintaining low parameter count and inference cost.

The DBAL-YOLO architecture was designed under practical deployment constraints of scanned CAD-based Plan-to-BIM workflows, where structural elements are typically slender, multi-scale, and embedded in cluttered linework, while computational efficiency remains critical. Accordingly, the selected modules jointly address three core requirements: directional sensitivity to elongated structures, effective multi-scale feature interaction, and lightweight inference.

Dynamic Snake Convolution (DSConv [19]) is selectively embedded in C3K2 blocks to enhance receptive behaviour along dominant geometric directions, which is particularly beneficial for detecting beams, columns, and thin linear components without introducing excessive computational overhead. To propagate such directional features across scales, a bidirectional feature pyramid network (BiFPN [20]) is adopted as the neck, enabling weighted cross-scale fusion with reduced redundancy. To further strengthen contextual discrimination in noisy scanned drawings, the original attention mechanism is replaced with AFGC attention [21], which compactly integrates global and local information critical for resolving ambiguous symbols and dense layouts. In addition, a lightweight asymmetric detection

head (LADH [22]) is employed to reduce head-side parameters and FLOPs while preserving localisation accuracy. Crucially, these modules are adapted and selectively deployed (e.g., localised DSConv insertion and depthwise-separable BiFPN) so they act synergistically within a unified computational budget. This task-oriented integration is anticipated to improve sensitivity to slender and multi-scale structural elements while reducing overall model size and computational cost, which will be analysed and discussed (based on ablation analysis and heatmap comparisons) in the following sections.

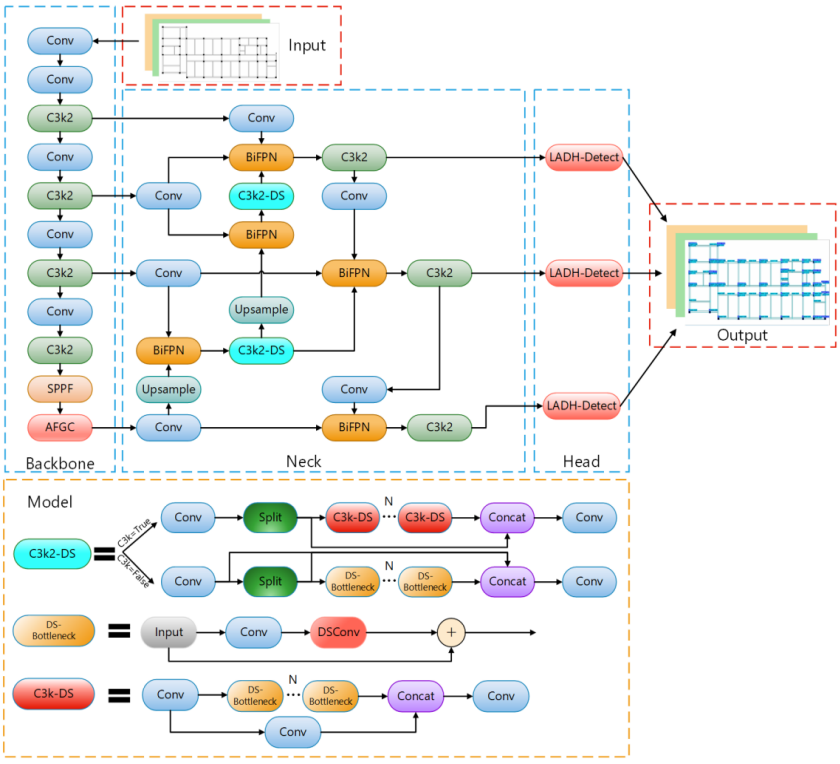


Figure 3. DBAL-YOLO network structure.

2.2.1. Improved C3k2 module

In the YOLOv11n neck network, certain C3K2 modules incorporate Dynamic Snake Convolution (DSConv), as illustrated in Figure 4. DSConv begins by linearising a standard 3 × 3 kernel into a length-9 sequence (Equation (1)–(2)), as given by:

$$K_{i+c} = (x_{i+c}, y_{i+c}) \tag{1}$$

$$K = \{(x-1, y-1), (x-1, y), \cdots, (x+1, y+1)\} \tag{2}$$

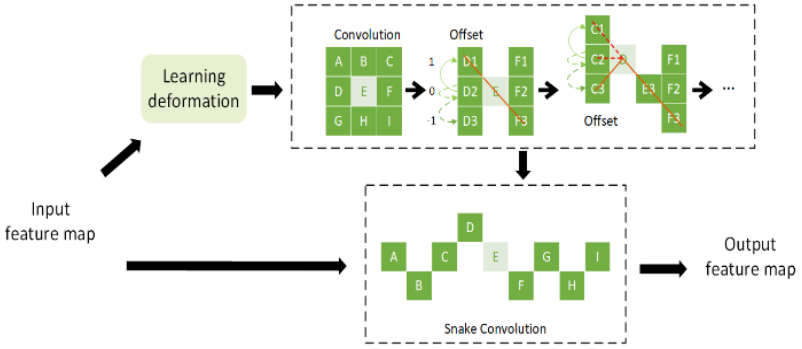


Figure 4. DSConv structure.

It then uses an accumulative offset Δy (Equation (3)–(4))—where each new position depends on the sum of previous offsets-to compute each grid point's precise coordinates:

$$K_{i\pm c} = \begin{cases} (x_{i+c}, y_{i+c}) = (x_i + c, y_i + \sum_{i}^{i+c} \Delta y) \\ (x_{i-c}, y_{i-c}) = (x_i - c, y_i + \sum_{i-c}^{i} \Delta y) \end{cases} \tag{3}$$

Similarly, the offset along the Y axis is described by:

$$K_{j\pm c} = \begin{cases} (x_{j+c}, y_{j+c}) = (x_j + \sum_{j}^{j+c} \Delta x, y_j + c) \\ (x_{j-c}, y_{j-c}) = (x_j + \sum_{j-c}^{j} \Delta x, y_j - c) \end{cases} \tag{4}$$

Through this iterative update mechanism, DSConv markedly enhances the network's capacity to detect fine-grained, elongated structures in engineering drawings.

2.2.2. BiFPN network

In this paper, a Bidirectional Feature Pyramid Network (BiFPN) into YOLOv11n for enhanced feature fusion was integrated. BiFPN, an improved architecture built upon bidirectional cross-scale fusion FPN [23], is depicted in Figure 5. By introducing learnable weights for each incoming feature map, the model's fusion capability is further strengthened.

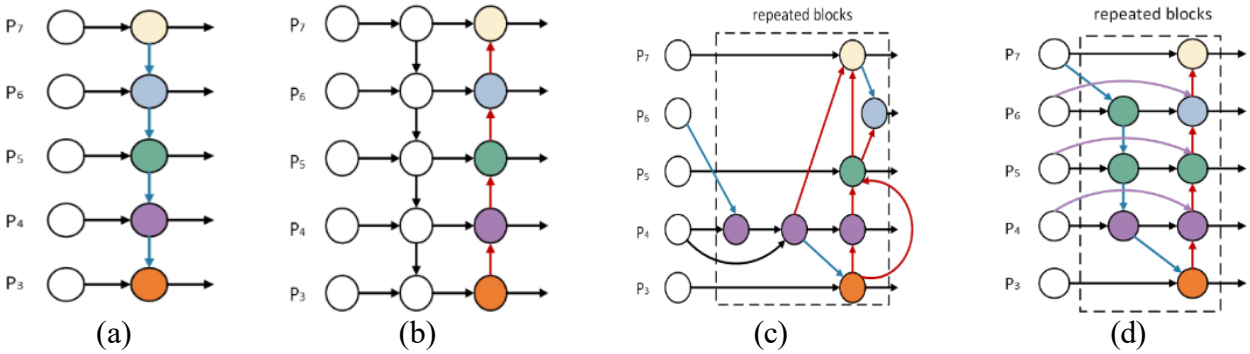


Figure 5. Feature networks. **(a)** FPN; **(b)** PANet; **(c)** NAS-FPN; **(d)** BiFPN.

BiFPN employs both top-down and bottom-up connections alongside a fast normalisation fusion mechanism. The dual fusion at level 6 can be described using Equation (5), as given by:

$$\begin{cases} P_6^{td} = \mathrm{Conv}\left(\dfrac{w_1 \times P_6^{in} + w_2 \times \mathrm{Resize}(P_7^{in})}{w_1 + w_2 + \varepsilon}\right) \\ P_6^{out} = \mathrm{Conv}\left(\dfrac{w_1' \times P_6^{in} + w_2' \times P_6^{td} + w_3' \times \mathrm{Resize}(P_5^{out})}{w_1' + w_2' + w_3' + \varepsilon}\right) \end{cases} \tag{5}$$

where P_6^{in} and P_7^{in} are the inputs at levels 6 and 7 with weights w_1 and w_2; then, w_1', w_2', and w_3' weight the three inputs in the bottom-up fusion. Conv denotes depthwise separable convolution followed by batch normalisation and activation; ε ensures numerical stability. Other levels follow the same construction.

2.2.3. Feature refinement and efficient detection

To achieve enhanced feature representation and a lightweight design, the AFGCAttention module replaces C2PSA to fuse global and local contexts, while the Lightweight Asymmetric Detection Head (LADH-Head) reduces parameters and FLOPs through its asymmetric structure, improving overall efficiency without compromising detection accuracy. The asymmetric design of LADH-Head is particularly advantageous for processing structural engineering drawings, where components such as beams and columns often exhibit slender, elongated geometries and are arranged in multi-scale, densely annotated layouts. Unlike symmetric detection heads that apply uniform processing across all feature dimensions, the asymmetric architecture allocates computational resources more flexibly, emphasising feature refinement along the dominant spatial directions of structural elements. This directional adaptability enhances the model's ability to localise slender components accurately while suppressing false positives caused by cluttered linework or symbolic annotations. Furthermore, the asymmetric reduction in network parameters aligns with the need for efficient inference in practical BIM workflows, enabling robust detection performance even on hardware-constrained platforms.

2.3. Semantic segmentation algorithms (for components)

The U-Net segmentation algorithm [24] is well-suited for segmenting geometrically irregular building walls due to its strength in preserving fine details. Its symmetric encoder-decoder architecture, complemented by skip connections, effectively captures global semantic context while recovering high-resolution local features during upsampling. Compared to alternatives like FCN and the DeepLab series, U-Net offers a more streamlined structure and demonstrates high pixel-level accuracy even with limited training data, enabling precise wall segmentation in engineering drawings. The network architecture is illustrated in Figure 6.

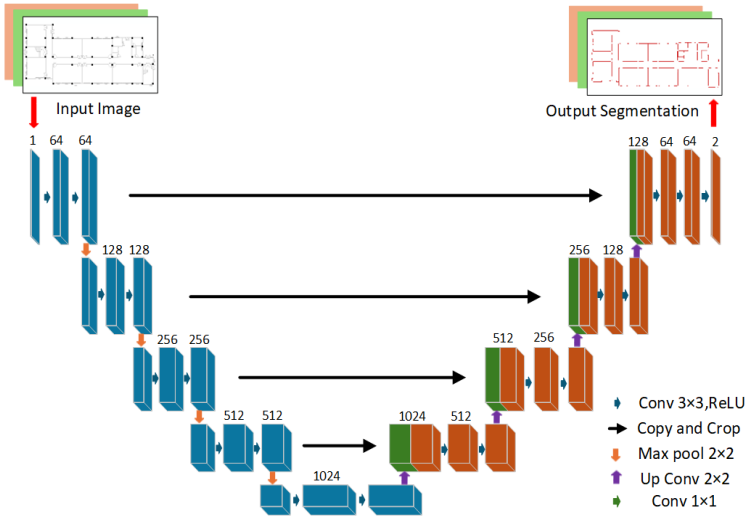


Figure 6. U-Net network structure**.**

2.4. 3D reconstruction approaches

2.4.1. 3D reconstruction methods for beams, columns, and slabs

To reconstruct columns, beams, and slabs, the following workflow based on object detection outputs was proposed:

(1) Parameter Extraction and Data Preprocessing: The workflow begins by automatically extracting precise dimensional parameters from the engineering drawings using Optical Character Recognition (OCR) technology. This process captures both component-level dimensions (e.g., lengths, heights, thicknesses) and level heights. The OCR module identifies and interprets the relevant textual annotations, converting them into machine-readable numerical values. Concurrently, detection bounding boxes for structural components are summarised. To correct minor localisation errors caused by detection and measurement inaccuracies, a modularisation-based error-correction scheme is applied (Figure 7), which yields precise 2D corner coordinates $p_i(x_i, y_i)$.

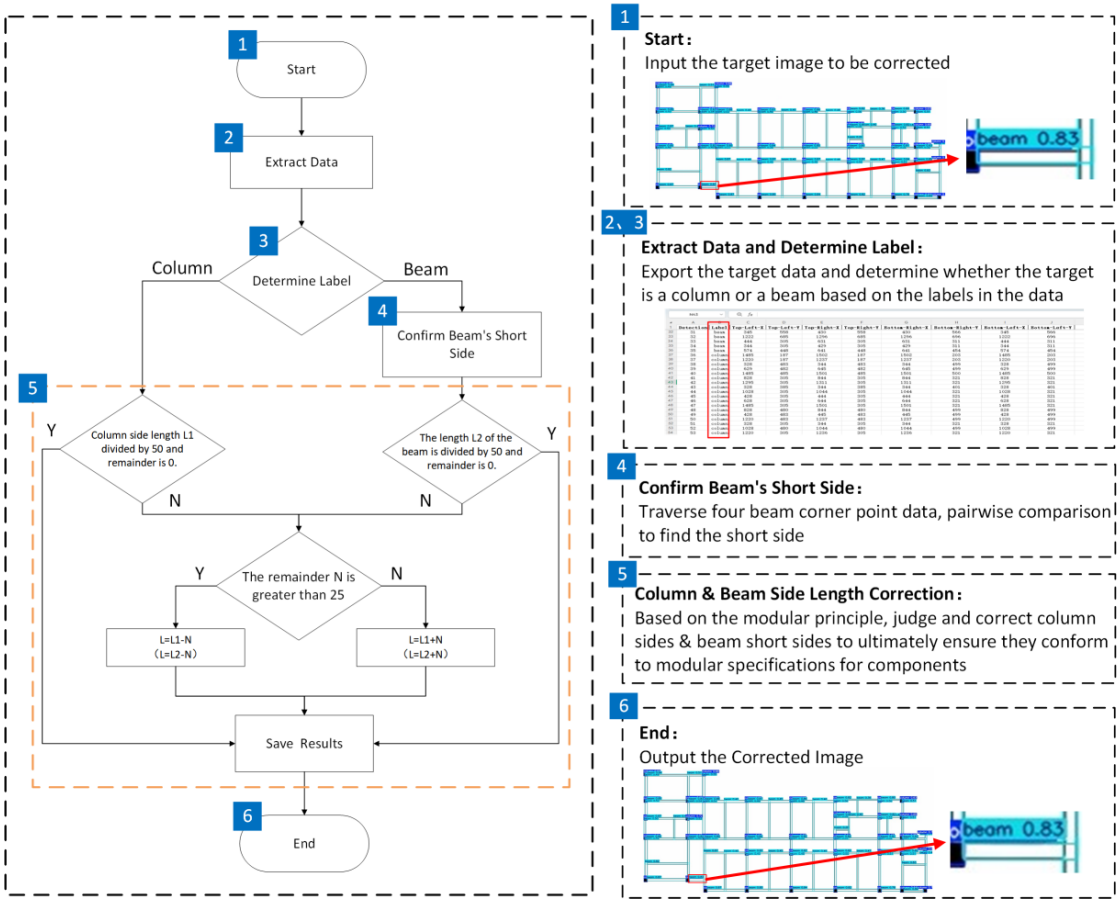


Figure 7. Process of component error check and correction.

(2) 3D Plane Construction: Using the corrected 2D points, a linear extrusion method generates 3D solids with thickness. The bottom and top vertex coordinates are defined as:

$$v_i^b = (x_i, y_i, h_0) \tag{6}$$

$$v_i^t = (x_i, y_i, h_0 + h) \tag{7}$$

where v_i^b and v_i^t denote the coordinates of the bottom surface and top surface vertices of the 3D component, respectively, h_0 is the elevation of the bottom surface, and h is the component height.

(3) Mesh Partitioning and Model Generation: To faithfully represent the geometry, the surfaces of each component are triangulated into facets. Vertex coordinates are stored in arrays conforming to the numpy-stl schema, and standard STL files are generated via library functions for efficient storage and downstream use.

2.4.2. 3D reconstruction methods for walls

To enable automated 3D reconstruction of building walls, a background mask via a U-Net segmentation model was generated, then extract wall contours through image processing, and finally construct 3D models from these contours. The principal steps include:

(1) Image Preprocessing and Region Segmentation: Load architectural drawings with OpenCV, convert from BGR to HSV to suppress noise, extract pixels within predefined mask thresholds, and merge multiple threshold ranges to obtain a robust wall-region mask.

(2) Contour Extraction and Polygon Approximation: Use OpenCV to extract contours from the mask, capturing geometric outlines. Raw contours are often complex due to noise and thus apply approxPolyDP with tolerance, as given in Equation (8), in order to approximate them as simpler polygons, reducing the vertex count while preserving the essential shape.

$$\epsilon = 0.005 \times \mathrm{arcLength}(\mathrm{contour}, \mathrm{True}) \tag{8}$$

(3) Extrusion Reconstruction and Model Generation: Closed polygons are defined in the XY-plane and linearly extruded along the Z-axis to the prescribed wall height, yielding 3D solids. These individual solids are then joined and merged, and the resulting unified model is exported as a standard STL file using appropriate library functions.

2.4.3. Export and BIM interoperability

In the current implementation, reconstructed component geometries are exported as triangulated meshes in STL format for efficient storage, visualisation, and result verification (Sections 2.4.1–2.4.2). Although STL provides a lightweight and widely compatible mesh-based representation, it does not encode semantic, parametric, or spatial hierarchy information, which limits its direct applicability to BIM-oriented downstream tasks such as structural analysis, facility management, and quantity take-off. To enhance interoperability with BIM standards, the proposed approach can be extended by leveraging the semantic labels and geometric parameters extracted during the detection and reconstruction stages. Specifically, detected components (e.g., walls, columns, beams, and slabs) can be mapped to corresponding IFC entity types (such as IfcWallStandardCase, IfcColumn, IfcBeam, and IfcSlab) and represented using parametric geometry, for example, through profile-based extrusion (IfcExtrudedAreaSolid), with associated property sets (IfcPropertySet) and spatial containment relationships (IfcRelContainedInSpatialStructure). The resulting IFC models therefore contain explicit engineering semantics and parametric characteristics, enabling direct use in structural analysis and facility management systems. Practical implementation can be realised using open-source libraries (e.g., IfcOpenShell, xBIM) or commercial BIM platform APIs (e.g., Revit API).

This study adopts STL as the output format to ensure clarity of the reconstruction pipeline and convenience for visual validation, while clearly outlining a feasible extension toward IFC-based BIM representations, thereby laying the foundation for subsequent semantic enrichment and system-level integration in engineering applications.

3. Experimental results and analysis

3.1. Dataset preparation

The dataset used in this study was collected from multiple real-world sources to ensure diversity and representativeness of non-digital engineering drawings. The drawings were primarily obtained from local architectural design institutes and archival management departments, covering a range of building types (e.g., residential, educational, and commercial structures) and spanning several design periods to include variations in drafting standards, graphical styles, and annotation conventions. The collected drawings vary in terms of drawing scale, line density, symbol usage, and physical condition (e.g., varying line clarity, scanning artefacts, and handwritten annotations), which collectively represent the typical range of document states encountered in scanned archival materials.

A total of 1,320 original drawings were collected and processed (Figure 8), each manually annotated with bounding boxes for structural components (columns and beams) and pixel-level masks for walls using the LabelMe tool. To enhance dataset diversity and model robustness, the dataset was expanded to 3,960 samples through data augmentation, including 90° and 180° rotations. The drawings and corresponding annotations were then randomly shuffled and divided into training, validation, and test sets with a ratio of 7:2:1, ensuring balanced distribution across subsets for reliable model evaluation.

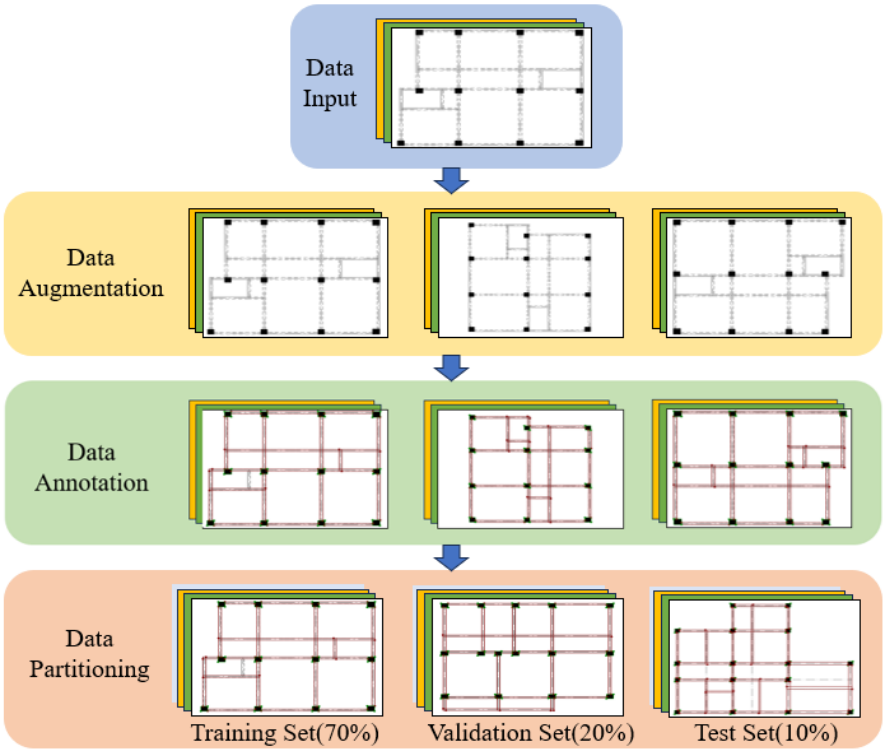


Figure 8. Process of dataset preparation.

3.2. Experimental platform and environment

Experiments were conducted on a Windows 10 (64-bit) operating system, with a deep learning environment comprising Python 3.10, CUDA 11.8, and PyTorch 2.0.0. Model training was performed on an NVIDIA RTX A5500 GPU with 24 GB of memory. More details of the environment configurations are summarised in Table 1.

Table 1. Configuration of experimental environment.

Experimental environment	Environmental configuration
Operating environment	Window10 (64-bit)
Programming environment	Python 3.10
Deep learning framework	PyTorch 2.0.0
CUDA	11.8
CPU	13th Intel(R) Core(TM) i9-13900 K 3.0 GHz
GPU	NVIDIA RTX A5500-24 GB
Memory	64 GB

3.3. Evaluate metrics

For object detection, model performance is quantified using five metrics: recall (R), precision (P), mAP, number of parameters (Params), and computational complexity (GFLOPs). Precision is defined as the proportion of true positives among all positive predictions, and recall is the proportion of true positives among all actual positives, as given by:

$$P=\frac{TP}{TP+FP} \tag{9}$$

$$R=\frac{TP}{TP+FN} \tag{10}$$

where TP, FP and FN are true positives, false positives and false negatives, respectively. Precisions and recalls are computed for the detected classes to produce the P-R relationship curves; the area under each P-R curve yields the corresponding average precision (AP), as given by:

$$AP=\int_0^1 p(r)dr \tag{11}$$

Then, the mean value over all the n classes is represented by mAP, as given by:

$$mAP=\frac{1}{n}\sum_{i=1}^{n} AP_i \tag{12}$$

where p(r) is the precision at recall r, and AP_i is the AP for class i.

For semantic segmentation, the primary evaluation indicators are the mPA and mIoU. mPA quantifies the overall pixel-level classification accuracy, and mIoU is the average of the IoU values across all classes, as given by:

$$mPA=\frac{1}{n+1}\sum_{i=0}^{n}\frac{P_{ii}}{\sum_{i=0}^{n}\sum_{j=0}^{n}P_{ij}}\times 100\% \tag{13}$$

$$mIoU=\frac{1}{n+1}\sum_{j=0}^{n}\frac{P_{ii}}{\sum_{j=0}^{n}P_{ij}+\sum_{j=0}^{n}P_{ji}-P_{ii}}\times 100\% \tag{14}$$

where n is the number of classes, P_{ij} represents the number of pixels erroneously sorted into class j while the true class is i, and P_{ii} is the number of pixels correctly predicted for class i.

3.4. Comparison of object detection models

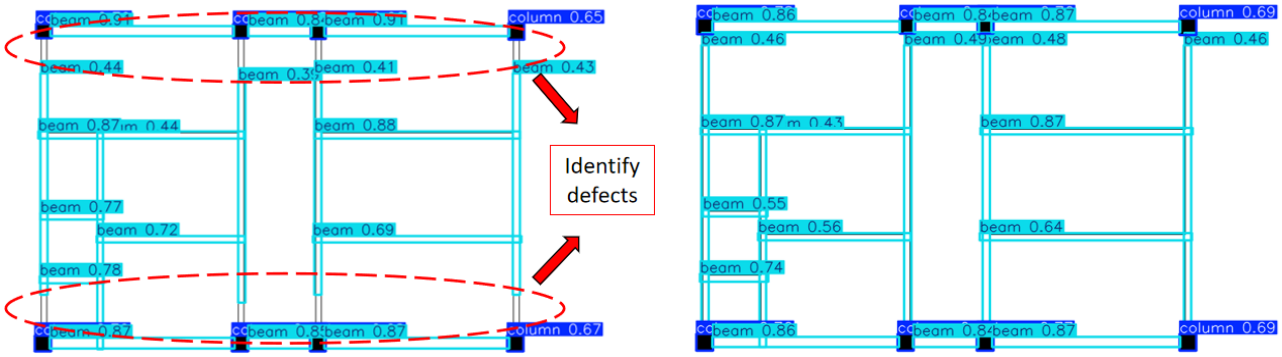
3.4.1. DSConv performance comparison

To quantify DSConv's contribution to detecting slender structural elements, a controlled ablation study was performed, comprising the following configurations: YOLOv11n (Baseline); full replacement of all C3K2 blocks with DSConv; DSConv applied only to the backbone; DSConv applied only to the neck; DSConv applied to the anterior (front) half of the neck; and DSConv applied to the posterior (back) half of the neck (ours). Table 2 summarises quantitative results (Precision, Recall, mAP, Params, and GFLOPs).

Table 2. Ablation results of DSConv applied to different C3K2 blocks.

Model	Precision (%)	Recall (%)	mAP (%)	Params (M)	GFLOPs (G)
YOLOv11n (Baseline)	97.2	97.9	82.1	2.58	6.6
Full C3K2	96.5	97.6	76.2	3.00	7.0
Backbone C3K2	94.4	90.2	68.5	2.77	6.7
Neck C3K2	97.5	98.2	83.5	2.81	6.7
Neck (back) C3K2	96.8	97.5	82.5	2.76	6.5
Neck (front) C3K2 (Ours)	98.1	98.8	83.6	2.63	6.5

To provide practical intuition for the DSConv ablation results, a qualitative comparison of detection outputs with and without DSConv on representative slender structural elements in floor plans is shown in Figure 9. The DSConv-enabled model yields tighter, more complete bounding boxes and reduces fragmentation of thin beams and columns, while the baseline frequently produces partial detections or bounding boxes that fail to fully enclose elongated elements. These visual observations are consistent with the quantitative ablation results reported in Table 2.



(a) (b)

Figure 9. Comparison of structural component detection results with and without DSConv. **(a)** Baseline detection without DSConv; **(b)** Detection with DSConv integrated.

3.4.2. BiFPN performance comparison

To evaluate the effectiveness of the BiFPN design for multi-scale feature fusion in architectural floor plan detection, an ablation comparison was conducted against several standard pyramid architectures. The baseline neck of YOLOv11n (Baseline) was replaced with conventional FPN, PANet, and NAS-FPN, and these variants were compared with the BiFPN under identical training and evaluation settings. Table 3 reports Precision, Recall, mAP, Params, and GFLOPs.

As shown in Table 3, the BiFPN variant achieves superior overall detection accuracy while maintaining comparable model complexity and inference cost, indicating that the proposed BiFPN provides more effective multi-scale feature aggregation for architectural drawings.

Table 3. Ablation results of different feature pyramid structures.

Model	Precision (%)	Recall (%)	mAP (%)	Params (M)	GFLOPs (G)
YOLOv11n (Baseline)	97.2	97.9	82.1	2.58	6.6
FPN	95.5	95.4	79.8	1.89	6.2
PANet	96.8	95.9	80.6	2.35	6.6
NAS-FPN	97.3	98.0	81.8	2.64	6.8
BiFPN (Ours)	98.0	98.2	83.2	1.92	6.3

3.4.3. Experimental comparison among different improvement modules

To isolate the contribution of the proposed AFGCAttention module, an ablation study was carried out using a common detection backbone augmented with different attention mechanisms. The evaluated variants include the baseline network without attention, models equipped with Squeeze-and-Excitation (SE [25]) and Convolutional Block Attention Module (CBAM [26]), as well as the proposed AFGC-Attention. All models were trained and tested under identical experimental settings. Quantitative comparisons in terms of detection accuracy and computational complexity are reported in Table 4.

Table 4. Ablation results of different feature pyramid structures.

Model	Precision (%)	Recall (%)	mAP (%)	Params (M)	GFLOPs (G)
YOLOv11n (Baseline)	97.2	97.9	82.1	2.58	6.6
+SE	97.5	85.6	78.9	2.34	6.1
+CBAM	96.9	84.0	76.6	2.41	6.1
+AFGCAttention (ours)	97.5	97.8	83.4	2.40	6.1

As shown in Table 4, the proposed AFGCAttention yields the best trade-off between detection accuracy and computational overhead among the tested attention mechanisms. It achieves the highest mAP while incurring only marginal additional parameters and latency compared with lightweight alternatives. Accordingly, AFGCAttention was adopted in the final model and used in all subsequent experiments.

3.4.4. Experimental comparison among different improvement modules

To further validate the efficacy of the proposed algorithmic enhancements, ablation experiments were conducted with various module combinations, as summarised in Table 2, with ticks (√) indicating the inclusion of the corresponding module. All experiments were performed for 350 epochs with a batch size of 32, using stochastic gradient descent (SGD) as the optimiser, and an initial learning rate of 0.01. The comparative results are summarised in Table 5.

Table 5. Comparison of various improvement modules.

Model	DSConv	BiFPN	AFGC	LADH	Precision (%)	Recall (%)	mAP (%)	Params (M)	GFLOPs (G)
YOLOv11n					97.2	97.9	82.1	2.58	6.6
	√				98.1	98.8	83.6	2.63	6.5
	√	√			98.2	98.9	83.5	1.97	6.5
	√	√	√		98.4	98.4	84.1	1.79	6.3
DBAL-YOLO	√	√	√	√	98.8	98.3	85.2	1.62	5.4

The ablation studies in Table 5 systematically evaluate the contributions of each proposed module. Beginning with the YOLOv11n baseline, components were added incrementally to assess their impact on performance. Replacing the standard C3k2 module with the C3k2-DS structure, which incorporates DSConv, increased both Precision and Recall by approximately 0.9%, indicating its utility in detecting elongated structural components. The integration of BiFPN for bidirectional multi-scale feature fusion reduced the number of parameters by 25.1%, confirming its role in constructing a more efficient network architecture. Replacing the C2PSA layer with the AFGCAttention mechanism improved mAP by 0.6% while lowering computational complexity, supporting its function in refining feature representation. The use of the lightweight LADH detection head further decreased parameter count and computational load, with no reduction in accuracy. The complete DBAL-YOLO model, which integrates all four modules, achieved a mAP of 85.2% with 1.62 M parameters and 5.4 GFLOPs, demonstrating that the modules contribute collectively to enhancing performance while maintaining efficiency.

In object detection tasks, Box_loss and mAP0.50:0.95 are two critical performance indices. Box_loss serves as a loss function to quantify the discrepancy between predicted bounding boxes and ground-truth bounding boxes. Its primary objective is to minimise coordinate errors (including centre point offsets, width, and height) to improve localisation accuracy. A lower Box_loss value indicates smaller geometric deviations between predicted and ground-truth boxes, indicating enhanced localisation capability. As a comprehensive evaluation metric originally established for the Microsoft COCO (Common Objects in Context) dataset, mAP0.5:0.95 computes the AP across ten discrete IoU thresholds ranging from 0.50 to 0.95 (step size 0.05) and averages these values. This multi-threshold evaluation framework, pioneered by COCO, systematically assesses detection robustness under varying localisation precision requirements. When applied to this study's custom dataset, the same computational protocol was adopted to ensure standardised performance benchmarking. Higher mAP0.5:0.95 scores reflect superior generalisation capabilities under balanced precision-recall trade-offs. Figure 10 presents quantitative comparisons of the Box_loss and mAP0.50:0.95 indices before and after architectural refinements.

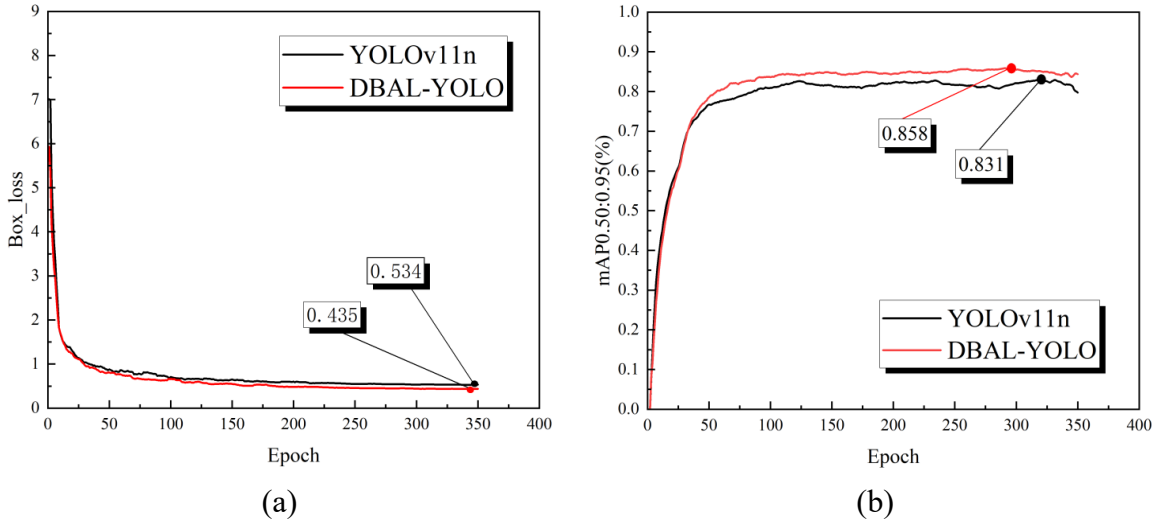


Figure 10. Comparison of key parameters of the model before and after improvements. **(a)** Comparison in bounding box loss; **(b)** Comparison in mAP0.50:0.95.

Heatmap comparisons achieve a visual interpretation of models' decision-making focus, revealing inherent disparities in feature localisation accuracy, noise robustness, and multi-object discrimination capability across target detection architectures. This analysis establishes an interpretable foundation for precise model optimisation, with exemplar heatmaps from high-level feature layers demonstrating enhanced feature activation patterns. To validate this optimisation paradigm, another three incrementally modified models were also engineered: (a) Improved Model 1: Baseline YOLOv11n neck integrated with C3K2-DS module, (b) Improved Model 2: Baseline YOLOv11n neck with C3K2-DS and BiFPN incorporation, and (c) Improved Model 3: Baseline YOLOv11n neck with C3K2-DS, BiFPN and AFGCAttention incorporation. The heatmap comparison is shown in Figure 11.

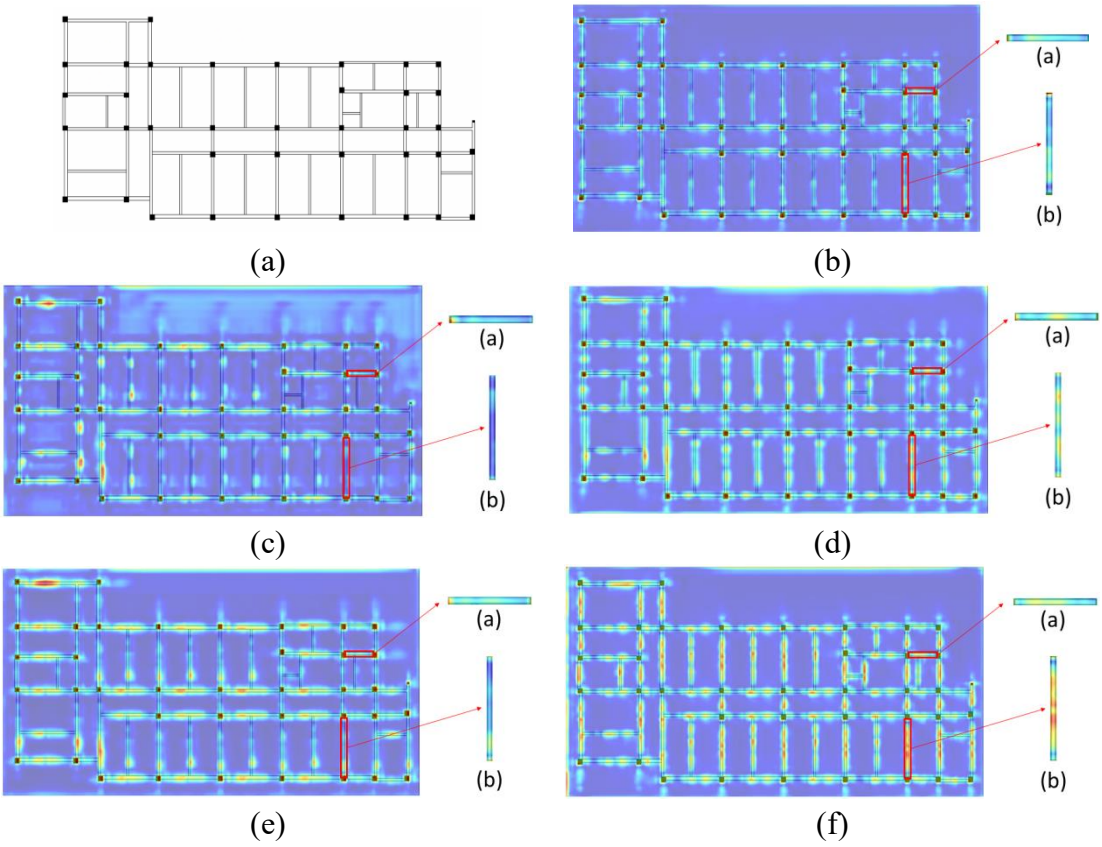


Figure 11. Comparison of heatmaps of the models. **(a)** Original; **(b)** YOLOv11n; **(c)** Improved model 1; **(d)** Improved model 2; **(e)** Improved model 3; **(f)** DBAL-YOLO.

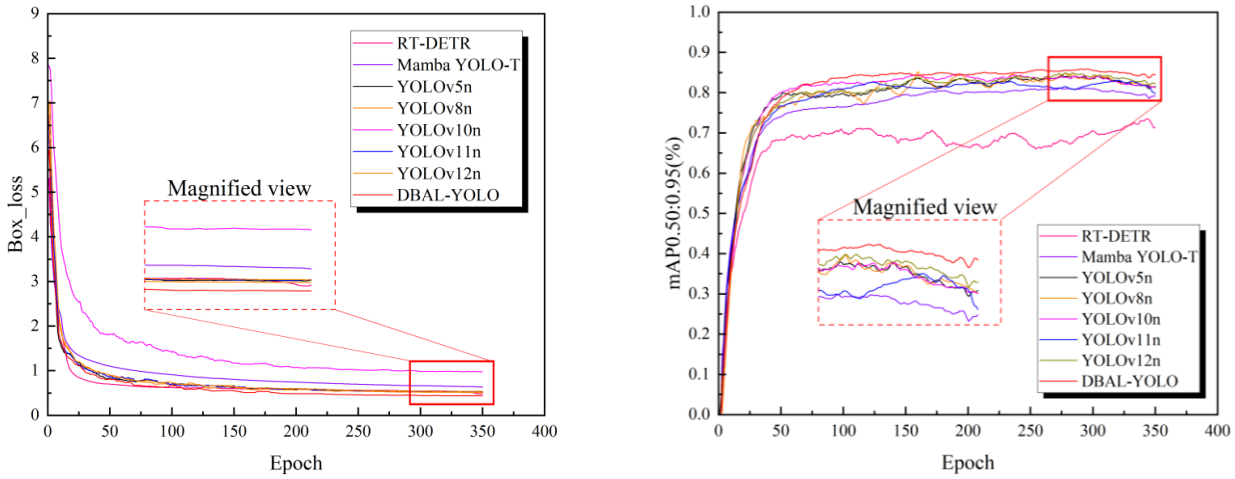
As shown in Figure 11, the target detection attention of YOLOv11n, Improved Model 1, and Improved Model 2 on beam and column components is significantly weaker than that of DBAL-YOLO. The activation intensity of target regions in their heatmaps is insufficient, making it difficult to reflect the complete characteristics of structural components. Although the Improved Model 3 exhibits better attention on components along the horizontal direction of the target image, there are still deficiencies in attention along the vertical direction of the image. Moreover, its heatmap focus is mainly concentrated in the middle part of the beam, while the attention in the regions at both ends of the beam is significantly lower than that of DBAL-YOLO. A further comparison of the heatmap distributions in the local slender target regions (a) and (b) reveals that DBAL-YOLO demonstrates significant advantages in the feature perception ability of slender components. Its attention distribution is more balanced and covers the entire component region, and the heatmap intensity in non-target regions is significantly weaker than that of

other comparative models. This result validates the robustness and precision advantages of DBAL-YOLO in detecting slender targets in complex scenarios.

In summary, the proposed DBAL-YOLO algorithm reduces parameter volume by 38.2% and computational load by 14.3% *versus* YOLOv11n. Performance metrics on the validation set confirm significant enhancements: precision (+ 1.3%), recall (+ 0.4%), and mAP (+ 3.1%). These improvements, coupled with its refined attentional focus and elevated localisation accuracy during structural element detection, substantiate the framework's efficacy.

3.4.5. Experimental comparison of object detection classification algorithms

To further validate the performance of the proposed improved model, a comparative experiment was conducted on the same dataset using several representative lightweight mainstream models. The experimental settings were kept consistent with those used in the ablation study. The results are presented in Table 6, and the comparison of key parameters is illustrated in Figure 12.



(a) (b)

Figure 12. Comparison of key parameters across the tested models. **(a)** Comparison in bounding box loss; **(b)** Comparison in mAP0.50:0.95.

As shown in Table 6, YOLOv5n, YOLOv8n and YOLOv12n attain commendable precision but require substantially greater parameter counts and computational resources. In contrast, DBAL-YOLO preserves a lightweight profile with faster inference while delivering higher mAP, enabling more accurate and more complete detection of structural elements.

To provide an intuitive comparison beyond the numerical results in Table 6, Figures 13 and 14 visualise the detection outputs of four models (YOLOv5n, YOLOv8n, YOLOv12n and DBAL-YOLO) under different layout complexities. Figure 13 presents representative cases with relatively simple floor-plan structures and well-separated components, serving as a baseline scenario. Figure 14 further examines a single representative complex floor plan dominated by dense component distributions and slender elements, which better reflects challenging conditions commonly encountered in practical engineering drawings.

Table 6. Comparison of different object detection models.

Models	Precision (%)	Recall (%)	mAP (%)	Params (M)	GFLOPs (G)
RT-DETR [27]	95.4	86.2	78.0	31.9	103.4
Mamba YOLO-T [28]	97.4	98.4	82.1	5.78	13.2
YOLOv5n	98.4	98.6	83.6	2.18	5.8
YOLOv8n	98.5	98.8	84.0	2.68	6.8
YOLOv10n	93.0	90.3	85.6	2.27	6.5
YOLOv11n	97.2	97.9	82.1	2.58	6.6
YOLOv12n [29]	98.9	98.7	84.6	2.51	5.8
DBAL-YOLO	98.8	98.3	85.2	1.62	5.4

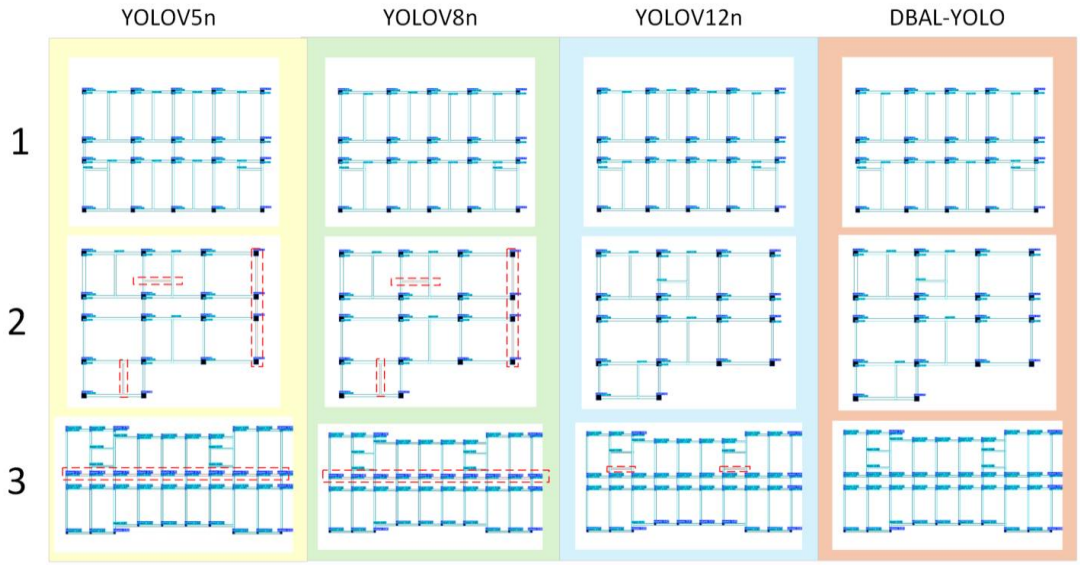


Figure 13. Comparison of different models on simple structural drawings.

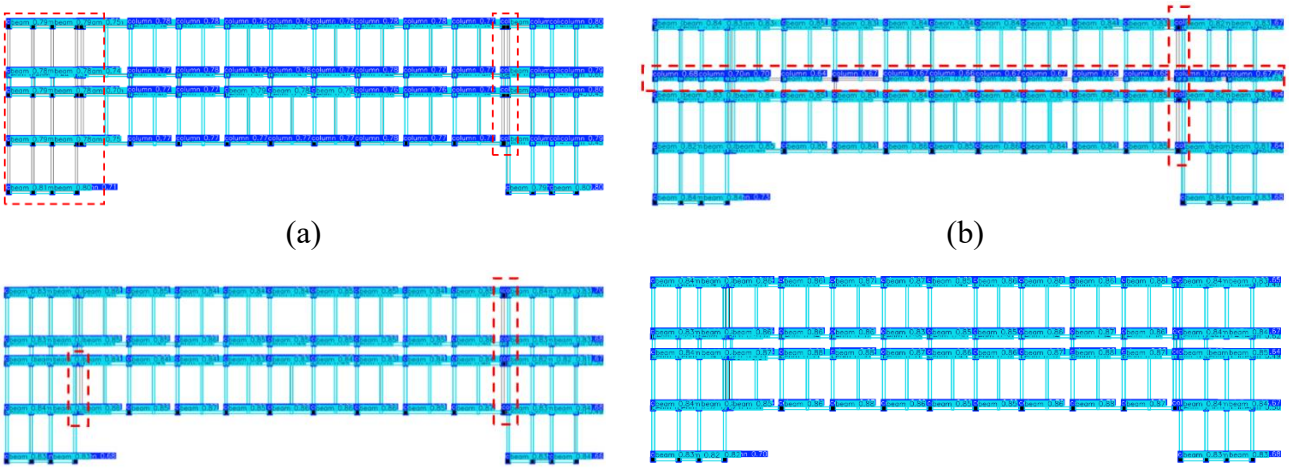


Figure 14. Comparison of different models on complex structural drawings. **(a)** YOLOv5n; **(b)** YOLOv8n; **(c)** YOLOv12n; **(d)** DBAL-YOLO.

The comparison indicates that, although the other models perform reasonably well on the simple layouts shown in Figure 13, some structural components remain undetected, whereas DBAL-YOLO

achieves consistently more complete detections even under low scene complexity. In the more challenging case shown in Figure 14, DBAL-YOLO produces notably more stable and complete results in densely packed regions, while the other models exhibit fragmented predictions or miss closely spaced, slender elements. These observations are consistent with the quantitative results in Table 6 and demonstrate the superior robustness of DBAL-YOLO for engineering drawings with dense structural layouts.

3.5. Segmentation model integration and performance

3.5.1. Role of U-Net in the YOLO-based framework

The U-Net semantic segmentation module is strategically incorporated into the predominantly YOLO-based framework to address a specific limitation of object detection in the context of certain architectural elements. While the enhanced DBAL-YOLO detector excels at localising and classifying discrete structural components characterised by well-defined rectangular geometries such as columns and beams, it faces inherent challenges when detecting building walls. In architectural drawings, walls typically exhibit complex, irregular, and non-rectilinear morphologies that cannot be accurately delineated by bounding boxes. Furthermore, walls often manifest as continuous regions rather than discrete objects, rendering them incompatible with conventional object detection paradigms.

The U-Net model is specifically employed to perform pixel-level segmentation of these irregular wall geometries. Its encoder-decoder architecture with skip connections enables precise delineation of wall boundaries, preserving the exact geometric contours regardless of shape complexity. This capability is crucial for generating accurate 3D wall models through extrusion operations in the subsequent reconstruction phase. In the integrated pipeline, DBAL-YOLO and U-Net operate in a complementary manner: DBAL-YOLO handles the detection of regular structural members, while U-Net specialises in segmenting irregular wall regions. The outputs from both models are subsequently fused during the 3D reconstruction process to generate a complete building model.

3.5.2. Comparative performance analysis

To evaluate segmentation model performance, a comparative study was conducted under identical experimental conditions using mainstream architectures: PSPNet [30], DeepLabv3+ [31], and the U-Net model adopted in this work. All experiments were run for 200 training epochs with a batch size of 4, employing both SGD and Adam optimisers. For optimal convergence, the learning rate was set to 1×10^{-4} when using SGD and to 1×10^{-2} when using Adam. Results are summarised in Table 7.

Table 7. Summary of experimental results for semantic segmentation algorithms.

Models	Optimizer	mPA (%)	mIoU (%)
PSPNet	SGD	91.2	96.3
Deeplabv3+	SGD	95.7	97.4
U-Net	SGD	94.1	97.8
PSPNet	Adam	96.2	98.3
Deeplabv3+	Adam	96.4	98.4
U-Net	Adam	96.4	98.8

As shown in Table 4, with the Adam optimiser, the U-Net model achieved an mPA of 96.4% and a mIoU of 98.8% on the wall segmentation task, outperforming the other mainstream algorithms. This model is primarily employed for segmenting building walls in the engineering drawings, thereby enhancing modelling accuracy and precision.

4. Integrated validation (case study)

The validity of the proposed procedure was checked using both structural and architectural floor plans of a five-storey building. This case study demonstrates the framework's capability to process and integrate information from both drawing types to generate a complete 3D building model.

4.1. Object detection of columns and beams

The structural plan of the building's first floor was extracted and processed with the DBAL-YOLO object detection model, yielding geometric components for 3D modelling and extracting their associated geometric parameters. The detection results are shown in Figure 15 and Figure 16.

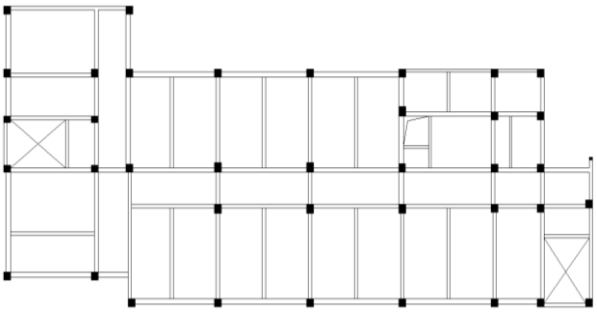


Figure 15. Layout of the structural floor plan.

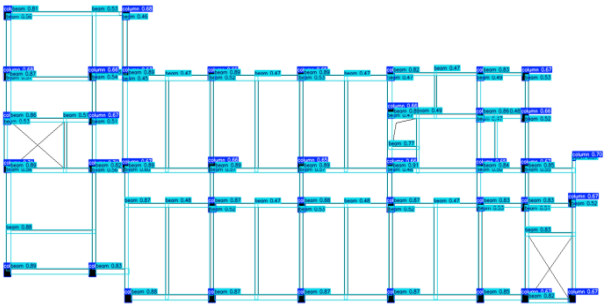


Figure 16. Detection results of columns and beams.

A comparison of Figures 15 and 16 shows that, for the specific floor plan illustrated in Figure 15, the proposed DBAL-YOLO model detected all column and beam elements—yielding a recognition accuracy of 100% on that drawing. This outcome not only verifies the model's robustness in accurately identifying structural components within planar drawings but also highlights its distinct advantage in preserving complete data integrity throughout the feature extraction process. The comprehensive and error-free recognition of structural elements further demonstrates that DBAL-YOLO can effectively extract critical features, thereby facilitating rapid 3D reconstruction while guaranteeing zero information loss in architectural modelling.

4.2. Segmentation for walls

In the floor plans, wall elements frequently exhibit irregular, non-rectilinear geometries that cannot be faithfully captured by bounding-box detection alone. To address this limitation, a UNet-based semantic segmentation module was integrated into the pipeline, specifically designed for pixel-level delineation of wall regions. Leveraging the U-Net's encoder-decoder architecture with skip connections, this module precisely extracts wall contours and reconstructs their geometric features, thereby enabling the generation of complete, high-fidelity 3D building models.

The architectural floor plan of the first story is shown in Figure 17. The optimised U-Net segmentation model is applied to extract the wall regions from the plan, and the resulting segmentation mask is presented in Figure 18 and Figure 19.

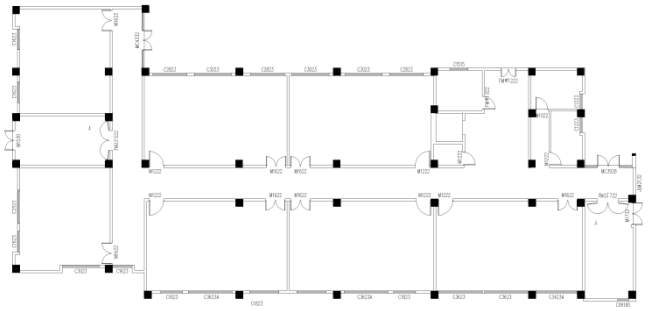


Figure 17. Floor plan of the example building.

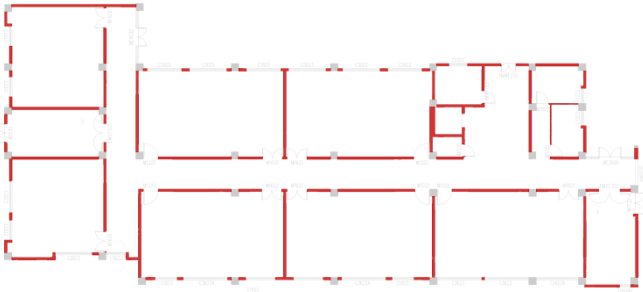


Figure 18. Segmentation results of walls.

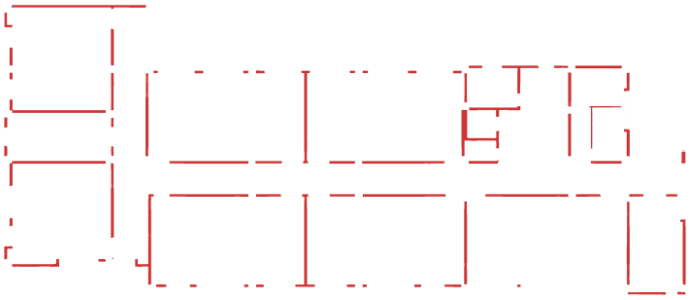


Figure 19. Segmentation results of walls without background.

Comparison between the results in Figure 17 and Figure 19 demonstrates that the parameter-tuned U-Net model effectively suppresses interference from irrelevant elements-doors, windows, staircases, and annotations-and directly produces rapid, high-precision, pixel-level segmentation of all wall regions in the drawings. This accurate delineation establishes a robust foundation for subsequent wall geometry

reconstruction, 3D model generation, and structural analysis, markedly improving both the efficiency and fidelity of the automated modelling workflow.

4.3. 3D reconstruction of the building

Building 2D structural images are first recognised, followed by refinement and extraction of positioning and dimensional information for columns and beams. The 3D structural framework of the building is then generated through triangulated facet partitioning, with its 3D transformation workflow illustrated in Figure 20.

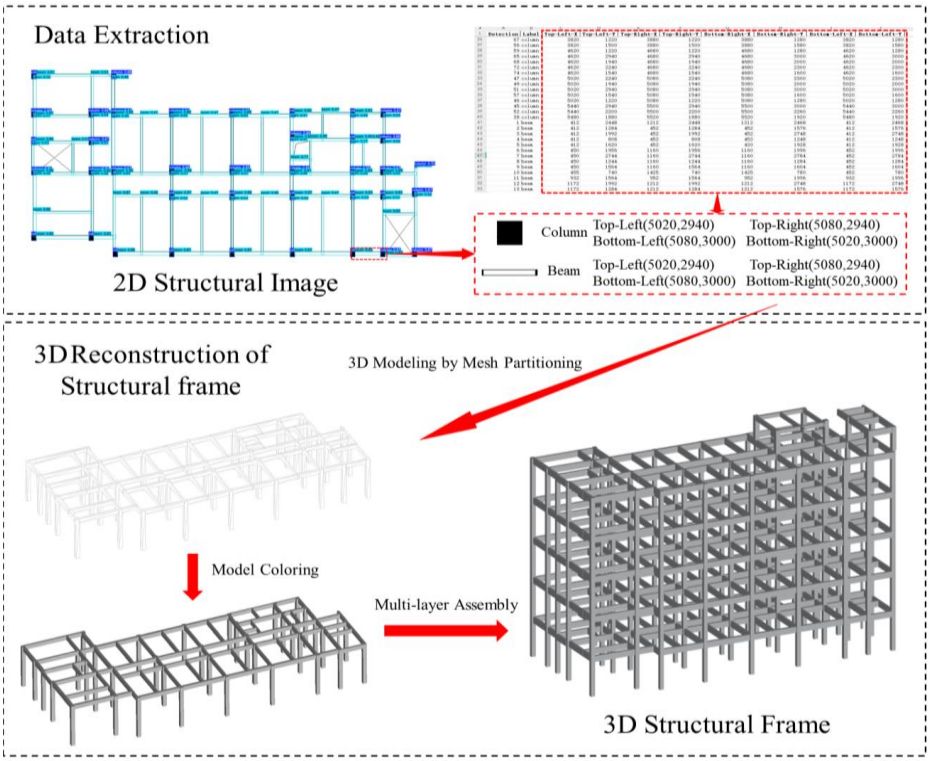
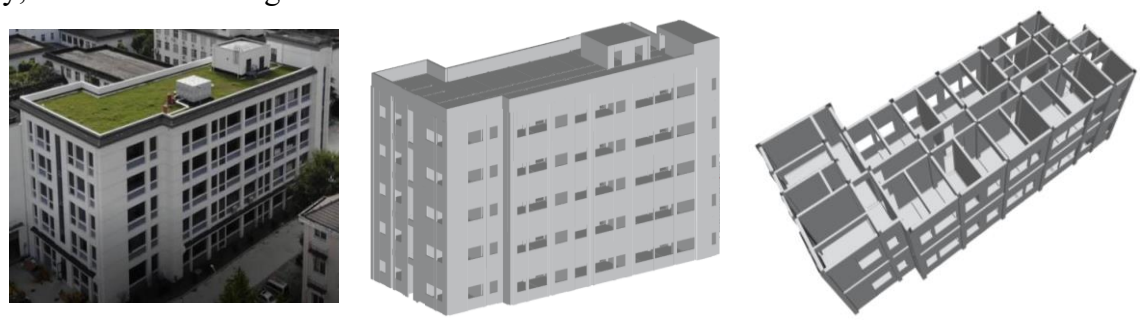


Figure 20. 3D structural frame reconstruction workflow.

Based on the 3D structural frame, 3D wall components are generated via linear extrusion from semantic segmentation masks. The complete 3D building model is subsequently assembled by integrating these components, and its geometric fidelity is validated against UAV-captured real-world imagery, as illustrated in Figure 21.



(a) (b) (c)

Figure 21. 3D model validation against UAV ground truth. **(a)** UAV-Captured Ground Truth Imagery; **(b)** Reconstructed 3D Building Model; **(c)** Cross-Sectional View of the Reconstructed Model.

By integrating the dimensional parameters derived from DBAL-YOLO-based column and beam detection with wall masks generated through U-Net semantic segmentation, as shown in Figure 20 and Figure 21, the proposed workflow enables rapid and precise 3D reconstruction of the structural frame. The reconstructed renderings of both individual columns and beams, as well as the overall model, fully comply with relevant modelling standards, thereby demonstrating high geometric fidelity and conformity with industry specifications.

To further validate the generalizability of the proposed workflow, an additional case study is provided in Figure 22. The figure presents a high-resolution original drawing, the corresponding detection and segmentation results, the automatically generated 3D model, and a real-world photograph for visual comparison. The successful reconstruction from this additional case confirms the method's consistent performance across varied drawing layouts.

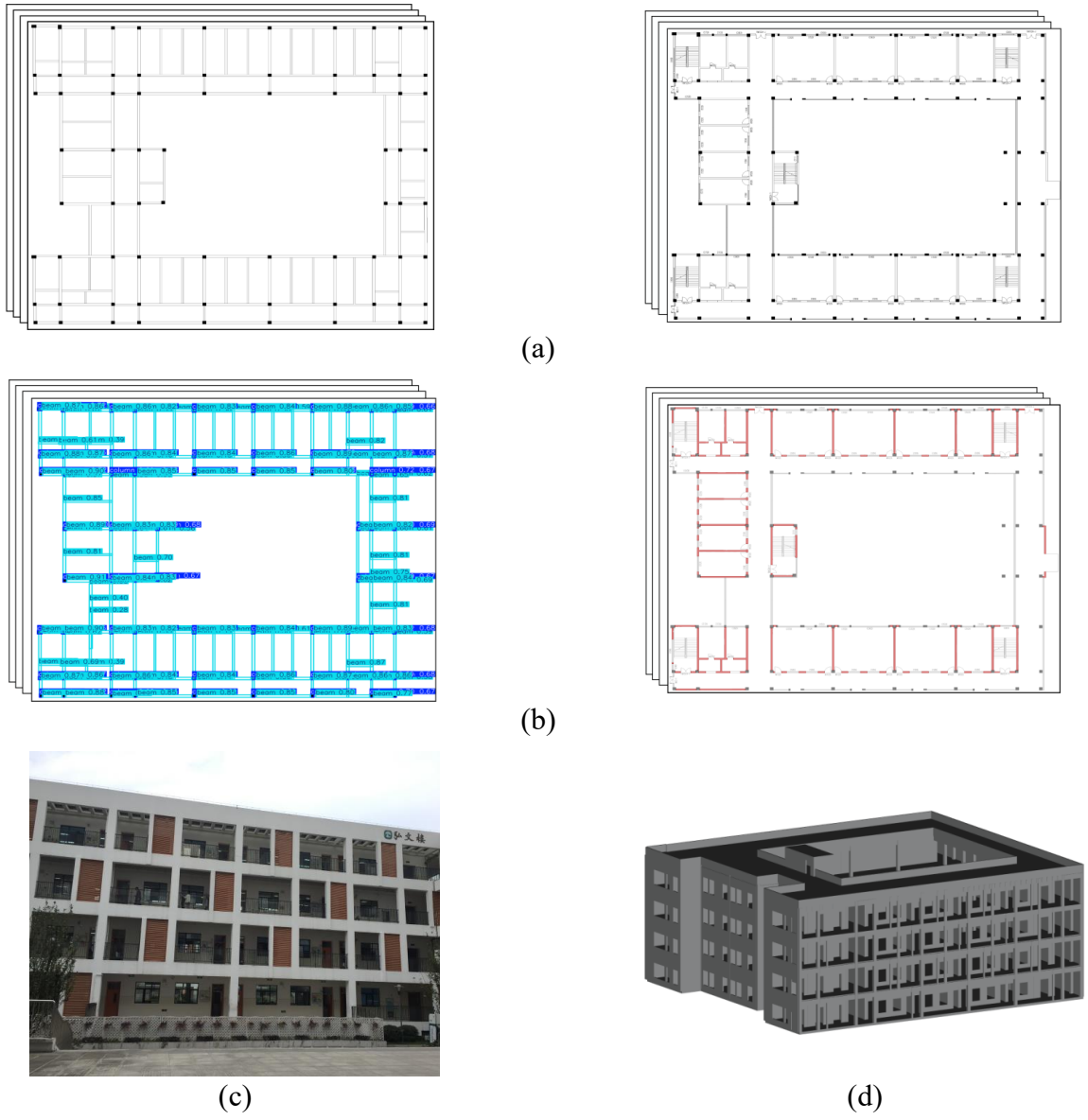


Figure 22. Additional validation case. **(a)** Original floor plan; **(b)** Detection and Segmentation results of the floor plan; **(c)** Real-world photograph for visual comparison; **(d)** Reconstructed 3D Building Model.

5. Discussions and future work

While the proposed framework demonstrates promising results in automated BIM reconstruction from 2D drawings, several limitations remain to be addressed in future research, including the underutilization of colour information in CAD layers and incomplete automation in the modelling workflow. Future studies will focus on the following directions:

(1) Colour-aware Processing Enhancement. Future research will focus on developing mechanisms to interpret and utilise colour codes in CAD drawings, which typically distinguish different architectural and structural layers. This would empower the model to better disentangle complex layouts and improve overall detection accuracy.

(2) The method performs robustly on typical datasets but may degrade under severe occlusion, overlapping components, or poor-quality scans (low contrast, dense annotations). Future work will explore occlusion-aware and multi-scale reasoning, synthetic occlusion data augmentation, multi-modal fusion, improved preprocessing, and uncertainty estimation with human verification to enhance robustness and clarify practical limits.

(3) Planned IFC export and semantic enrichment. Future work will implement a downstream exporter that converts the extracted detection/segmentation results into semantically rich IFC models. The conversion will (i) map detected classes to IFC entity types, (ii) create parametric geometry (extruded profiles) rather than faceted meshes where possible, (iii) generate property sets (dimensions, material, level) and spatial containment (building→storey→element), and (iv) validate IFC output with IfcOpenShell and commercial BIM platforms (e.g., Revit). This extension is expected to enable direct use of the reconstructed models for structural analysis, quantity take-offs, and facility management workflows.

6. Conclusions

An efficient structural member detection and rapid 3D reconstruction approach for the whole structure based on non-digital engineering drawings was developed in this work to achieve fast 3D digital model generation for city digital twin systems. Within the research work above, the following conclusions can be drawn:

(1) To improve multi-scale component recognition capability based on non-digital engineering drawings (scanned images), an improved YOLOv11n model, DBAL-YOLO, is developed. This model incorporates DSConv to enhance the neck's C3K2 module, integrates a BiFPN bidirectional feature pyramid network for efficient multi-scale feature fusion, replaces the original C2PSA layer with AFGCAttention to improve key region detection, and employs a lightweight LADH detection head. The proposed improvements reduce model parameter count and computational load while improving or maintaining detection accuracy, yielding a lightweight and effective solution for Plan-to-BIM tasks.

(2) To consolidate the precision in 2D component detection from non-digital engineering drawings, a module-based correction method is developed. According to the building module employed in conventional structural elements, the detection errors can be rectified, thereby significantly improving the geometric modelling accuracy.

(3) Based on the structural element recognition and correction methods above, a fast and automated 3D modelling method implemented in Python is then developed. Without relying on traditional modelling software or additional modelling engines, the proposed procedure generates 3D solids of building

components by traversing detection and segmentation results, applying linear extrusion to component contours, and ultimately achieving efficient, precise, and automated reconstruction of buildings.

Data availability statement

The data or datasets that support the findings of this study are available from the corresponding author upon reasonable request.

Acknowledgments

This work was supported by the Suzhou Science and Technology Plan (Basic Research) Project (Project No. SJC2023002) and the Advanced Perception and Intelligent Equipment Engineering Research Center of Jiangsu Province Open Projects Fund (Project No. 2025ZBYJ01).

Authors' contribution

Conceptualization, Xinjiang Cai, Baochegn Zhao and Yong Zhu; methodology, Xinjiang Cai and Jinchang Deng; software, Xinjiang Cai and Jinchang Deng; validation, Xinjiang Cai, Jinchang Deng and Yong Zhu; formal analysis, Jinchang Deng, Xinjiang Cai and Yong Zhu; investigation, Xinjiang Cai, Jinchang Deng and Yong Zhu; resources, Baocheng Zhao and Xiaoyong Mao; data curation, Xinjiang Cai and Jinchang Deng; writing—original draft preparation, Jinchang Deng and Xinjiang Cai; writing—review and editing, Yong Zhu; visualization, Jinchang Deng, Xinjiang Cai and Yong Zhu; supervision, Xinjiang Cai, Baocheng Zhao and Xiaoyong Mao; project administration, Xinjiang Cai and Yong Zhu; funding acquisition, Xinjiang Cai and Baocheng Zhao. All authors have read and agreed to the published version of the manuscript.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Weil C, Bibri SE, Longchamp R, Golay F, Alahi A, *et al.* Urban digital twin challenges: a systematic review and perspectives for sustainable smart cities. *Sustainable Cities Soc.* 2023, 99:104862.

[2] Lehtola VV, Koeva M, Elberink SO, Raposo P, Virtanen JP, *et al.* Digital twin of a city: review of technology serving city needs. *Int. J. Appl. Earth Obs. Geoinf.* 2022, 114:102915.

[3] LU QC, Parlikad AK, Woodall P, Ranasinghe GD, Xie X, *et al.* Developing a digital twin at building and city levels: a case study of west cambridge campus. *J. Manage. Eng.* 2020, 36(3):05020004.

[4] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: unified, real-time object detection. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, USA, June 26–July 1, 2016, pp. 779–788.

[5] Schönfelder P, Stebel F, Andreou N, König M. Deep learning-based text detection and recognition on architectural floor plans. *Autom. Constr.* 2024, 157:105156.

[6] Zeng Z, Li X, Yu Y, Fu C. Deep floor plan recognition using a multi-task network with room-boundary-guided attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Republic of Korea, October 27–November 2, 2019, pp. 9096–9104.

[7] Zhu X, Zhu Q, Zhang Q, Du Y. Deep learning-based 3D reconstruction of ancient buildings with surface damage identification and localization. *Structure* 2025, 73:108383.

[8] Xu Z, Jha N, Mehadi S, Mandal M. Multiscale object detection on complex architectural floor plans. *Autom. Constr.* 2024, 165:105486.

[9] Zhou Q, Zhao Y, Deng X. Recognition approach of building components in 2D drawings based on improved faster R-CNN (In Chinese). *J. Civ. Eng. Manage.* 2021, 38(5):110–117.

[10] Lu Q, Lee S. A semi-automatic approach to detect structural components from CAD drawings for constructing as-is BIM objects. In *Proceedings of the Computing in Civil Engineering 2017: Information Modeling and Data Analytics*, Seattle, USA, June 25–27, 2017, pp. 84–91.

[11] Schönfelder P, Aziz A, Faltin B, König M. Automating the retrospective generation of as-is BIM models using machine learning. *Autom. Constr.* 2023, 152:104937.

[12] Schönfelder P, König M. Ontology-based reasoning in automatic floor plan analysis. *Adv. Eng. Inform.* 2025, 68:103761.

[13] Brauksiepe M, Dollendorf M, Santehanser T, Wilkop S, Schönfelder P. Towards the rule-based synthesis of realistic floor plan images: a detailed guide. In *Proceedings of the 34. Forum Bauinformatik*, Bochum, Germany, September 27–29, 2023, pp. 317–322.

[14] Urbieta M, Urbieta M, Laborde T, Villarreal G, Rossi G. Generating BIM model from structural and architectural plans using artificial intelligence. *J. Build. Eng.* 2023, 78:107672.

[15] Yang B, Liu B, Zhu D, Zhang B, Wang Z, *et al.* Semiautomatic structural BIM-model generation methodology using CAD construction drawings. *J. Comput. Civ. Eng.* 2020, 34(3):04020006.

[16] Bacharidis K, Sarri F, Ragia L. 3D building façade reconstruction using deep learning. *ISPRS Int. J. Geo-Inf.* 2020, 9(5):322.

[17] Guo B, Yao Y, Li C, Wang Y, Sun N, *et al.* Application of improved 3D-BoNet to segmentation and 3D reconstruction of point cloud instances (In Chinese). *Bull. Surv. Mapp.* 2024, 6:30–35.

[18] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, *et al.* MobileNets: efficient convolutional neural networks for mobile vision applications. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, USA, July 21–26, 2017, pp. 2704–2713.

[19] Qi Y, He Y, Qi X, Zhang Y, Yang G, *et al.* Dynamic snake convolution based on topological geometric constraints for tubular structure segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France, October 2–6, 2023, pp. 6070–6079.

[20] Tan M, Pang R, Le QV. EfficientDet: scalable and efficient object detection. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, June 13–19, 2020, pp. 10778–10787.

[21] Sun H, Wen Y, Feng H, Zheng Y, Mei Q, *et al.* Unsupervised bidirectional contrastive reconstruction and adaptive fine-grained channel attention networks for image dehazing. *Neural Networks* 2024, 176:106314.

[22] Zhang J, Chen Z, Yan G, Wang Y, Hu B. Faster and lightweight: an improved YOLOv5 object detector for remote sensing images. *Remote Sens.* 2023, 15(20):4974.

[23] Lin T, Dollár P, Girshick R, He K, Hariharan B, *et al.* Feature pyramid networks for object detection. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, USA, July 21–26, 2017, pp. 936–944.

[24] Ronneberger O, Fischer P, Brox T. U-Net: convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Munich, Germany, October 5–9, 2015, pp. 234–241.

[25] Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, USA, June 18–22, 2018, pp. 7132–7141.

[26] Woo S, Park J, Lee JY, Kweon I. CBAM: convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, September 8–14, 2018, pp. 3–19.

[27] Zhao Y, Lv W, Xu S, Wei J, Wang G, *et al.* DETRs beat YOLOs on real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, USA, June 16–22, 2024, pp. 16965–16974.

[28] Wang Z, Li C, Xu H, Zhu X, Li H. Mamba YOLO: a simple baseline for object detection with state space model. *arXiv* 2024, arXiv:2406.05835.

[29] Tian Y, Ye Q, Doermann D. YOLOv12: attention-centric real-time object detectors. *arXiv* 2025, arXiv:2502.12524.

[30] Zhao H, Shi J, Qi X, Wang X, Jia J. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, USA, July 21–26, 2017, pp. 2881–2890.

[31] Chen L, Zhu Y, Papandreou G, Schroff F, Adam H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, September 8–14, 2018, pp. 833–851.