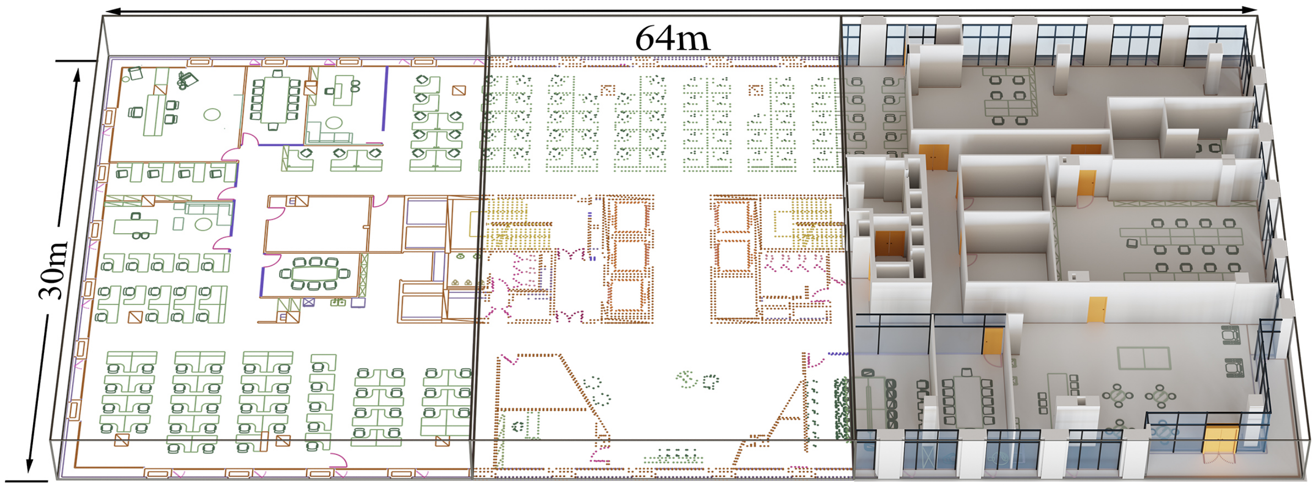
CADSpotting: Robust Panoptic Symbol Spotting on Large-Scale CAD Drawings

Jiazuo Mu[1,2†] Fuyi Yang[1,2†] Yanshun Zhang[2] Junxiong Zhang[1,2] Yongjian Luo[1]
Lan Xu[1] Yujiao Shi[1‡] Yingliang Zhang[2‡]
[1] ShanghaiTech University [2] DGene Digital Technology



CAD Drawing Input **Dense Feature Learning** **Spotting & 3D Interior Modeling**

Figure 1. Our CADSpotting method accurately identifies and segments symbols in CAD drawings, enhancing tasks like 3D interior modeling. It uses dense point sampling within a unified point cloud model for robust primitive feature representation and integrates Sliding Window Aggregation during inference for efficient panoptic symbol spotting in large-scale drawings. Finally, our method automates parametric 3D interior reconstruction by assembling architectural 3D objects guided by semantic information from CADSpotting.

Abstract

We introduce CADSpotting, an effective method for panoptic symbol spotting in large-scale architectural CAD drawings. Existing approaches struggle with symbol diversity, scale variations, and overlapping elements in CAD designs. CADSpotting overcomes these challenges by representing primitives through densely sampled points with attributes like coordinates and colors, using a unified 3D point cloud model for robust feature learning. To enable accurate segmentation in large, complex drawings, we further propose a novel Sliding Window Aggregation (SWA) technique, combining weighted voting and Non-Maximum Suppression (NMS). Moreover, we introduce LS-CAD, a new large-scale CAD dataset to support our experiments, with each floorplan covering around 1,000 m^2, significantly larger than previous benchmarks. Experiments on FloorPlanCAD and LS-CAD datasets show that CADSpotting significantly outperforms existing methods. We also demonstrate its practical value through automating parametric 3D reconstruction, enabling interior modeling directly from raw CAD inputs.

† Equal contributions.
‡ Corresponding author.

1. Introduction

Architectural Computer-Aided Design (CAD) drawings, especially those used for interior design, serve as detailed digital representations that convey essential information about a building's structure, layout, and intricate details. These drawings include critical elements such as furniture placement, electrical layouts, and spatial organization, ensuring consistency and precision throughout the construction process. For instance, the Shanghai Mercedes-Benz Arena utilize over 3,000 CAD drawings to guide its structural design, optimize spatial arrangements, and streamline the construction workflow. As CAD designs continue to grow in complexity, automating the detection and interpolation of symbols within these floorplans becomes increasingly important in various downstream tasks, such as code compliance checking and 3D interior modeling [20].

Despite the progress made in symbol recognition, the task of panoptic symbol spotting in CAD drawings remains under-explored. The challenges are multifaceted, arising from the vast diversity of symbol types, the need to differentiate between visually similar symbols, and the presence of overlapping elements. Additionally, variations in scale, orientation, and stylistic representation further complicate accurate identification.

Traditional methods approach symbol recognition in a query-by-example manner [29], but these techniques struggle to handle the vast diversity of graphical symbols in real-world datasets. More recent learning-based approaches [7, 25] have shown significant progress in addressing the symbol spotting task. Early solutions [5, 27, 40] typically leverage convolutional neural networks (CNNs) or graph-based models to spot symbols. More advanced methods leverage Transformer architectures and attention mechanisms [6, 40], improving global relationship reasoning among symbols. These methods use rasterized pixel images derived from vector CAD drawings as input. Such conversion, however, introduces discrepancies between the original vector graphics and the rasterized images, resulting in errors due to the loss of precise geometric information.

The representative work SymPoint [16] treats CAD drawings as sets of primary points, leveraging techniques from point cloud analysis for effective CAD symbol spotting. Such a representation simplifies model complexity while achieving strong performance on panoptic symbol spotting tasks in CAD drawings. Building on this foundation, SymPoint-V2 [17] introduces Layer feature-enhanced encoding to incorporate graphical layer information and a position-guided Training method to enhance model learning and accelerate convergence. Yet both approaches rely on manually defined primitive feature representations, i.e., four fixed primitive types. As a result, they struggle to capture the full diversity of CAD graphical primitives. In addition, these methods focus on processing small to medium-scale datasets due to the absence of effective strategies for preserving connectivity during symbol spotting across expansive drawings. By far, there is still a lack of feasible methods to address the panoptic symbol spotting task in large-scale CAD drawings, as dense symbol clutter and significant variations in scale present substantial challenges for accurate segmentation and symbol recognition.

In this paper, we present CADSpotting, a simple yet highly effective panoptic symbol spotting method tailored for handling CAD drawings at a much larger scale. Specifically, CADSpotting does not rely on fixed graphic primitive types. Instead, it densely samples points along CAD graphic primitives to construct comprehensive point cloud representations. Each point is represented by basic attributes, such as coordinates and color, forming a lightweight but expressive feature set. We utilize a unified 3D point cloud processing model to learn robust features from the sampled points. CADSpotting then employs a streamlined Transformer decoder for panoptic symbol spotting, effectively leveraging the learned features to achieve precise segmentation. To handle large-scale, real-world CAD drawings, we further propose a novel Sliding Window Aggregation (SWA) technique, combining a weighted voting strategy with Non-Maximum Suppression (NMS) to enable accurate and effective panoptic segmentation.

On the data front, we further introduce LS-CAD, a new large-scale CAD dataset containing 50 annotated floorplans from a variety of building types, including campuses and office complexes. Each drawing follows the same fine-grained annotation standards as the FloorPlanCAD dataset, with an average coverage of over 1,000 square meters per floorplan. LS-CAD is the first dataset of its scale in this domain, part of which will be released to the research community under a license waiver, to stimulate further developments. Experimental results on both FloorPlanCAD and LS-CAD demonstrate that CADSpotting outperforms the state-of-the-art methods by a margin, demonstrating its robustness and effectiveness in handling the complexities of large-scale, real-world CAD drawings.

We also demonstrate the effectiveness of CADSpotting by automating the generation of 3D interior models through parametric modeling. Using precise instance-level data obtained from CADSpotting, we compute spatial parameters for architectural elements, including wall contours, door and window positions, door orientations, and pivot points. With these spatial parameters clearly defined, we leverage parametric reconstruction techniques within Blender to efficiently build 3D interior models directly from raw CAD data. This approach significantly streamlines the modeling process, enhancing productivity and accuracy. To the best of our knowledge, our proposed pipeline is the first comprehensive solution that seamlessly integrates CAD segmentation with automated 3D interior model reconstruction.

2. Related Work

Panoptic Image Segmentation. Image segmentation is a fundamental task in computer vision, traditionally divided into two main categories: semantic segmentation [1, 2, 18, 34, 36] and instance segmentation [8, 9, 13, 28]. Semantic segmentation assigns a class label to every pixel in an image, while instance segmentation distinguishes individual objects within the same class. Recently, SAM [13] introduces a large model trained on over

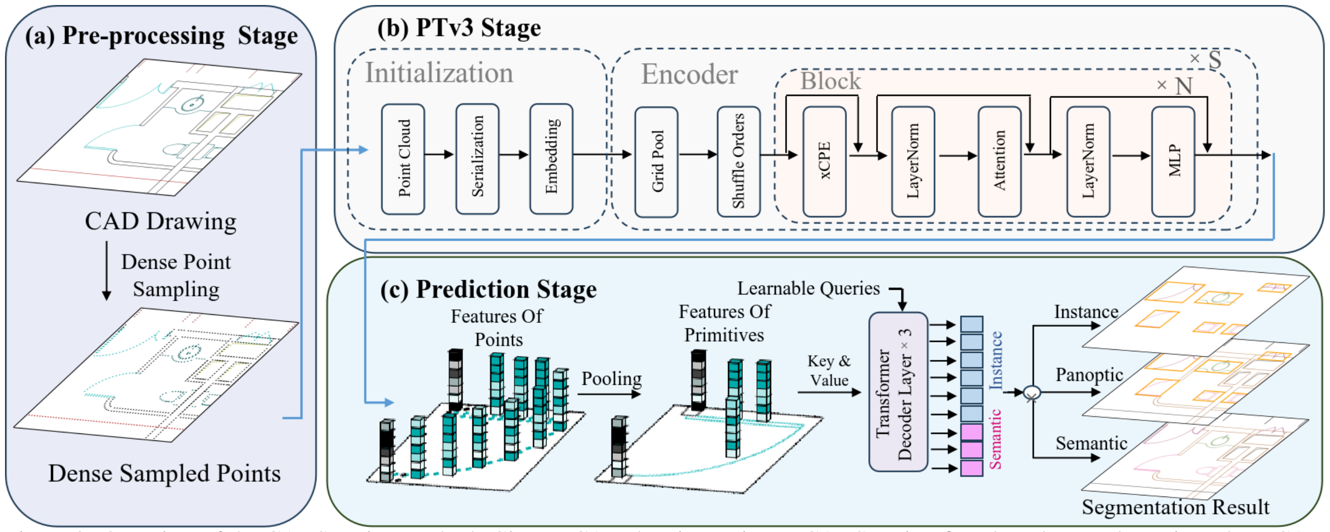


Figure 2. Overview of the CADSpotting method: Given a CAD drawing as input, CADSpotting first densely samples points along CAD graphic primitives to build a comprehensive point cloud representation, with each point defined by its coordinates and color. Next, Point Transformer V3 (PTv3) serves as the backbone, extracting robust features from the sampled points. We then apply mixed pooling to obtain the primitive-level features. Finally, a streamlined Transformer decoder is used for effective panoptic symbol spotting.

one billion masks, leveraging prompt-based interactions for enhanced interactive segmentation. SAM2 [24] extends this approach to video segmentation. However, semantic and instance segmentation methods struggle to accurately represent uncountable background elements, such as the sky or terrain. To address this challenge, Kirillov et al. [12] propose Panoptic Segmentation, which unifies semantic and instance segmentation, effectively handling both countable and uncountable background components. Early approaches [3, 11, 15, 31] primarily employ CNN-bases architectures, while recent advancements, such as Mask2Former [4], SegFormer [36], and OneFormer [9], leverage Transformer models, incorporating a mask attention mechanism to boost segmentation performance. Following the large-model paradigm of SAM, SEEM [41] further refines segmentation using a prompt-driven design. Despite these advances, most existing segmentation models remain pixel-centric, limiting their effectiveness in processing vector graphics like CAD drawings. Consequently, they often fail to capture critical vector-based information, such as overlapping relationships and precise geometric attributes.

CAD Symbol Spotting. Early efforts [10, 22, 23, 26, 37] in symbol spotting on CAD drawings rely on handcrafted feature descriptors, such as shape and structure, coupled with techniques like sliding window searches or graph matching for symbol retrieval. The introduction of deep learning techniques marks a significant shift, with models based on Faster R-CNN [25] and YOLO v3 [7] greatly enhancing the accuracy of symbol recognition tasks [27]. However, traditional symbol spotting methods are primarily designed for countable objects and do not effectively handle uncountable symbols, which are common in CAD environments. To address this limitation, Fan et al. [5] introduce the FloorPlanCAD dataset and CNN-GCN method, enabling the recognition of both countable and uncountable symbols for more comprehensive CAD parsing. Countable object instances (e.g., windows, doors) can be clearly identified and individually counted whereas uncountable stuff (e.g., wall, railing) refers to elements that cannot be individually distinguished. Building on this, Zheng et al. [40] propose GAT-CADNet, a model that represents CAD drawings as graph structures and performs segmentation by predicting adjacency matrices. Further advancements include CADTransformer [6], which utilizes the Vision Transformer to extract features from scalar maps, leading to improved predictions. Meanwhile, significant progress has been made in the field of 3D point cloud segmentation [14, 30, 35, 39]. The seminal work SymPoint [16] proposes a novel method that converts symbols into primary point representations, thereby enhancing feature extraction through point cloud analysis and upsampling techniques. Concurrently, Liu et al. [17] extend SymPoint by incorporating LFE and PGT modules, further improving feature representation and model performance. Despite these advances, the manually defined feature representation is often constrained by a limited set of graphical primitive types, which hinders their ability to capture the full diversity of CAD symbols.

3. Method

An overview of our CADSpotting method is illustrated in Fig.2. By incorporating a dense point sampling strategy

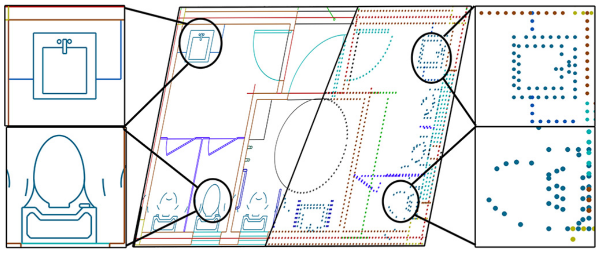


Figure 3. Dense point sampling. Left: A CAD drawing containing various primitives. Right: Dense point sampling converts each primitive symbol to a dense set of points.

within a unified point cloud processing model, our method effectively extracts features for CAD primitives, yielding a robust representation that supports various primitive types. We describe each module in detail in the following sections.

3.1. Primitive Feature Learning

SymPoint [16] introduces a new concept that represents CAD primitives as a set of 2D points. It constructs a handcrafted feature for each graphical primitive, capturing details such as type (e.g., line, arc, circle, ellipse), length, etc. SymPoint then applies point cloud analysis to learn primitive representations, further enhancing connectivity through an Attention with Connection Module. A masked-attention transformer decoder is used to complete the panoptic symbol spotting task. Although SymPoint shows strong performance in CAD symbol spotting, its feature representation is manually defined and limited to four fixed primitive types, which may constrain its capacity to capture the diverse range of CAD symbols. Real-world CAD drawings often contain various complex elements, such as Bézier curves, rather than simple predefined line types. Therefore, methods that rely on handcrafted features lack sufficient robustness. To address these constraints, we propose a novel graphical primitive feature representation for CAD drawings that incorporates dense point sampling within a unified point cloud analysis model to improve generalizability of primitive-wise representations.

Dense Point Sampling. In CAD drawings, graphical primitives are typically categorized into line types and shape types. Line primitives encompass straight segments, curves, and complex contours such as Bézier curves, while shape primitives include basic geometric forms like rectangles, circles, and ellipses. Inspired by SymPoint [16], we propose a dense point sampling strategy that performs equidistant sampling on each primitive to generate dense point data for feature extraction. Rather than assigning fixed types to CAD graphical primitives and using handcrafted features to represent them, our dense point sampling method offers greater feasibility and adaptability, making it suitable for a wide range of CAD primitives. Fig.3 illustrates the process of our dense point sampling method.

Given a CAD drawing containing N primitives, we perform dense sampling to generate a 2D point cloud P from these primitives at a fixed distance d. Each 2D point $p \in P$ is represented by a six-dimensional vector: $(x, y, z, r, g, b) \in \mathbb{R}^6$,where x, y represent the 2D coordinates of the point, z is set to 0, and r, g, b represent the color information (red, green, blue channels) of the point. We then employ Point Transformer V3 [35] (PTv3) as the backbone for feature extraction, leveraging its design optimized for the unordered nature of point clouds, which provides robust performance in feature extraction.

Feature Pooling. To obtain the final feature representation for CAD primitives, we apply Primitive Mixed Pooling (PMP) to transform point-wise features into primitive-wise features. Let $\mathcal{PMP} : \mathbb{R}^{M\times C} \rightarrow \mathbb{R}^{N\times C}$ denote our pooling operator, which aggregates the point features $\mathbf{f} \in \mathbb{R}^{M\times C}$ into primitive-wise feature vector $\mathbf{g}_i \in \mathbb{R}^{N\times C}$:

$$\mathbf{g}_i = \mathcal{PMP}_i(\mathbf{f}) \quad \forall i \in \{1, ..., N\} \tag{1}$$

where M denotes the total number of sampled points, and N represents the number of primitives. Let A_i represent the i^{th} single primitive with its corresponding dense point cloud P_i. The c-th dimension feature of $\mathbf{g}_i$ is determined by the sum of the maximum and average values of the corresponding c-th dimension features across all points:

$$\mathbf{g}_i(c) = \mathcal{PMP}_i^c(\mathbf{f}) = \underbrace{\max_{p\in P_i} \mathbf{f}_p(c)}_{\text{Max pooling}} + \underbrace{\frac{1}{|P_i|}\sum_{p\in P_i} \mathbf{f}_p(c)}_{\text{Average pooling}} \tag{2}$$

This pooling method condenses a large set of dense point features into a more compact set of primitive-level features, effectively reducing computational complexity and storage requirements. At the same time, primitive-level pooling retains the essential information of the contained points, preserving global consistency in the primitive representation and minimizing information loss. Loss calculations are also conducted at the primitive-level, aligning with our final goal of performing CAD spotting tasks.

3.2. Panoptic Symbol Spotting

We utilize a simple transformer decoder to extract semantic and instance symbol spotting information from the learned primitive features. In our model, the pooled primitive features serve as key and value inputs to three consecutive layers of Transformer decoders, where the self-attention mechanism effectively captures dependencies among the input features. The queries are initialized randomly and optimized during training to compute accurate self-attention scores with other features.

Following OneFormer3D [14], we partition the decoder output into two parts: the first K_{ins} represent instance proposals, and the subsequent K_{sem} outputs correspond to semantic symbol spotting. The output dimension of the K_{ins} + K_{sem} vector is $N \times (\text{Num}_{\text{ins}} + \text{Num}_{\text{sem}})$, integrating both instance and semantic information. The first Num_{ins} dimensions are used to generate instance masks, while the remaining Num_{sem} dimensions determine the semantic category of each primitive. This configuration enables panoptic symbol spotting by integrating instance and semantic information for comprehensive scene understanding.

We translate decoder outputs to semantic proposals and instance proposals. A semantic proposal includes the log-likelihood values for each primitive being classified into different semantic labels and can be further processed through thresholding to generate a binary mask. Similarly, an instance proposal also contains log-likelihood values, but these indicate the likelihood of each primitive being classified as a specific instance. Additionally, we employ a bipartite matching strategy based on the Hungarian algorithm to find the optimal correspondence between predicted results and ground truth labels while training. During inference, we select the top-k instance proposals of the highest confidence scores and apply matrix-NMS [33] to suppress non-maximal instance predictions.

Loss Function. The loss function L is composed of classification loss and primitive mask loss. The classification loss L_{cls} is defined as a multi-class cross-entropy loss. The primitive mask loss combines binary cross-entropy loss, L_{bce}, with the Dice loss [21], L_{dice}, to evaluate the correspondence between predicted instances and ground truth.

$$L = \lambda_{\text{cls}} L_{\text{cls}} + \lambda_{\text{bce}} L_{\text{bce}} + \lambda_{\text{dice}} L_{\text{dice}} \tag{3}$$

3.3. Sliding Window Aggregation

To address the challenge of panoptic symbol segmentation at large scales, a key issue is ensuring the generalization of instance symbol segmentation across varying scales. During inference on larger drawings than those encountered during training, the network must produce significantly more instance proposals. However, this process is limited by the fixed number of instance queries inherent in Mask Transformer-based methods, a constraint that also affects our approach. Nonetheless, CAD instance segmentation offers certain advantages over general image instance segmentation. Specifically, in CAD floorplans, instances of the same category maintain relatively consistent sizes across different scenes when a uniform scale is applied. Consequently, we can reasonably assume that the size of instances in inference will not exceed that of the same category in the training images.

Therefore, we propose using a fixed-size window based on the training image dimensions and apply 2D sliding window inference on the large input drawings. For each sliding window, once the primitives intersects with or are encompassed by the window, we utilize the features of these primitives along with the window's semantic and instance proposals, thereby increasing the likelihood of fully capturing potential instances. After gathering proposals across all windows, we apply distinct aggregation strategies for semantic and instance proposals.

For semantic proposals, we use a voting scheme to predict each primitive's semantic label, treating each primitive as an independent unit. Primitives that extend across larger spatial areas may be captured by multiple windows. In such cases, we prioritize proposals that capture the primitive as completely as possible by using weighted voting, where each window's voting weight is determined by the proportion of the primitive's dense points observed relative to the total dense points of that primitive.

For instance proposals, we utilize Sparse Non-Maximum Suppression(Sparse-NMS) on the proposals generated from all windows, which increases processing efficiency (detailed analysis is provided in the supplementary material). A potential instance may be partially or fully observed across multiple windows. By ensuring a sufficiently small sliding step size, each instance is fully captured within at least one window. Incomplete proposals from other windows are subsequently filtered out during the NMS process.

3.4. Automated 3D Interior Reconstruction

CADSpotting delivers precise panoptic segmentation results, serving as the basis for our parametric 3D interior modeling algorithm. Utilizing detailed instance information and primitive positions from segmentation, we apply instance-level processing techniques to compute spatial parameters for key architectural elements, specifically walls, doors, and windows. For doors, we determine their positions, orientations, and pivot points by analyzing the relationships between arcs and lines within each segmented instance. Window positions, on the other hand, are directly extracted from their corresponding instances. Complex wall structures require a specialized approach: we first traverse and merge endpoints of adjacent lines within wall instances, forming closed polygons. These polygons are then rasterized into binary images, from which final wall contours are identified using the method from [32]. With these spatial parameters, we reconstruct a complete 3D interior model using a parametric reconstruction pipeline in Blender. The necessary 3D primitives for walls, doors, and windows are manually prepared in Blender, providing reusable templates for streamlined parametric interior mod-

Building	Area(m^2)	#Primitive	#Instance
Office 1	1,799	28,193	1,185
Campus 1	1,949	12,571	603
Hotel 1	1,193	26,709	335

Table 1. Summary statistics for five LS-CAD samples.

eling. This approach significantly enhances efficiency, automating the conversion of raw CAD data into structurally accurate 3D interior models within minutes through procedural generation. An example result from our automated 3D interior reconstruction method is illustrated in Fig. 4, with additional details provided in the supplementary material.

4. Experiments

4.1. The Proposed LS-CAD Dataset

We introduce LS-CAD, a new large-scale CAD dataset consisting of 50 floorplans from extensive buildings, such as campuses and office buildings. Each CAD drawing in LS-CAD covers an area of at least 1,000 square meters, with the number of primitives ranging from approximately 2,900 to over ten thousand. Table.1 showcases 3 large-scale CAD drawings from our dataset, providing detailed information such as area, number of primitives and instance count, which underscore the large-scale nature of LS-CAD.

Additionally, the floorplans in LS-CAD include a wide variety of complex symbols, such as doors, windows, walls, elevators and parking spaces. We provide fine-grained annotations consistent with the standards of the FloorPlanCAD [5] dataset. A portion of LS-CAD will be publicly available under a license waiver to support further research in panoptic symbol spotting for large-scale CAD drawings.

4.2. Experimental Settings

Implementation Details. The FloorPlanCAD dataset comprises approximately 15,000 drawings, annotated across 35 symbol categories, including 30 things and 5 stuff classes. To ensure fair comparison, we utilize the same training subset from the FloorPlanCAD dataset as used by SymPoint. We train our model on a machine with 8 NVIDIA A100 GPUs, using a batch size of 2 per GPU for 512 epochs. For dense point sampling, we set a fixed distance $d = 0.14$. We use the AdamW [19] optimizer with a learning rate of 10^{-4} and a weight decay of 0.05. Data augmentation techniques include random horizontal flipping with a probability of 0.5, global rotation and scaling transformations, translation along the x and y axes and color normalization to $[-1, 1]$. We set loss weight as $\lambda_{\text{cls}} : \lambda_{\text{bce}} : \lambda_{\text{dice}} = 0.5 : 1 : 1$. Additionally, we set $K_{\text{sem}} = 36$ as the number of semantic labels, and both the top-k value and K_{ins} are set to 220. We use $\Delta = 70$ as the step size of SWA on LS-CAD dataset.

Metrics. Following the approach of [6], we assess the performance of our model using multiple metrics. For semantic symbol spotting, we use F1 score, weighted F1 score (wF1), and mean Intersection over Union (mIoU). For instance symbol spotting, we employ AP50, AP75, and mean Average Precision (mAP). For panoptic symbol spotting, we utilize Panoptic Quality (PQ), Segmentation Quality (SQ), and Recognition Quality (RQ). PQ is a comprehensive metric combining SQ and RQ to evaluate model performance in semantic and instance segmentation tasks. Further details on metric formulation are available in [6].

Methods	F1	wF1	mIoU
PanCADNet [5]	80.6	79.8	-
CADTransformer [6]	82.2	90.1	-
GAT-CADNet [40]	85.0	82.3	-
SymPoint [16]	86.8	85.5	69.7
CADSpotting (ours)	**93.5**	**93.9**	**83.3**

Table 2. Quantitative comparison of semantic symbol spotting on FloorPlanCAD dataset.

Methods	AP50	AP75	mAP
DINO [38]	64.0	54.9	47.5
SymPoint [16]	66.3	55.7	52.8
CADSpotting (ours)	**72.2**	**69.1**	**69.0**

Table 3. Quantitative comparison of instance symbol spotting on FloorPlanCAD dataset.

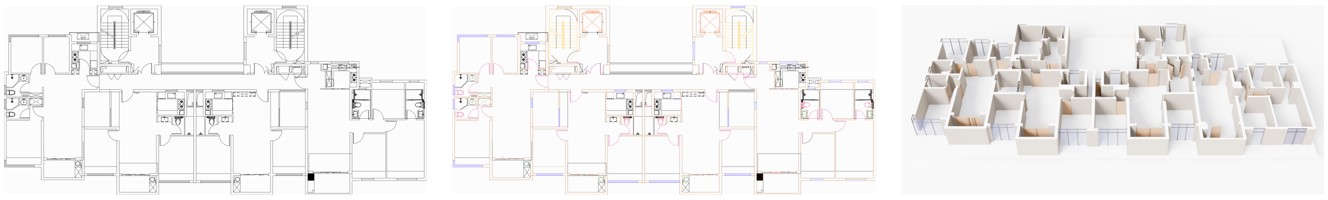
4.3. Quantitative Evaluation

To demonstrate the effectiveness of our approach, we compare it against SOTA methods on the FloorPlanCAD [5] and LS-CAD datasets. A detailed comparison of semantic, instance, and panoptic symbol spotting tasks is provided in the following paragraphs.

Semantic Symbol Spotting. As shown in Table.2, we compare our method with PanCADNet [5], CADTransformer [6], GAT-CADNet [40], and SymPoint [16]. Our CADSpotting approach achieves the highest performance in both the F1 and wF1 metrics, outperforming SymPoint by 13.6% in mIoU.

Instance Symbol Spotting. We also evaluate our method against DINO [38] and SymPoint for instance symbol spotting, using the bounding box maximization of predicted mask to compute the AP metric, consistent with SymPoint. As shown in Table.3, our method achieves superior results in the AP50, AP75 and mAP metrics, with improvements of 5.9% AP50, 8.4% AP75 and 8.9% mAP over SymPoint.

Panoptic Symbol Spotting. We then compare our method with the same methods used in the semantic symbol spotting comparison. The results, presented in Table.4, show that our approach surpasses SymPoint and other prior methods across all metrics, including Total/Thing/Stuff PQ, SQ and RQ. Additionally, our approach offers superior generalization across diverse CAD primitives, thanks to its



(a) CAD drawing as raw input (b) Prediction from CADspotting (c) Automated reconstruction

Figure 4. Our CADSpotting takes a CAD drawing as input (a) and outputs predicted semantic and instance segmentation results (b). After that, we design a automated reconstruction pipeline to generate 3D interior models (c).

Method	Total			Thing			Stuff		
	PQ	SQ	RQ	PQ	SQ	RQ	PQ	SQ	RQ
PanCADNet [5]	59.5	82.6	66.9	65.6	86.1	76.1	58.7	81.3	72.2
CADTransformer [6]	68.9	88.3	73.3	78.5	94.0	83.5	58.6	81.9	71.5
GAT-CADNet [40]	73.7	91.4	80.7	-	-	-	-	-	-
SymPoint [16]	83.3	91.4	91.1	84.1	94.7	88.8	48.2	69.5	69.4
CADSpotting (Ours)	**88.9**	**95.6**	**93.0**	**89.7**	**96.2**	**93.2**	**80.6**	**89.7**	**89.8**

Table 4. Quantitative comparison of panoptic symbol spotting on FloorPlanCAD dataset. Our method achieves the best performance among the comparison algorithms.

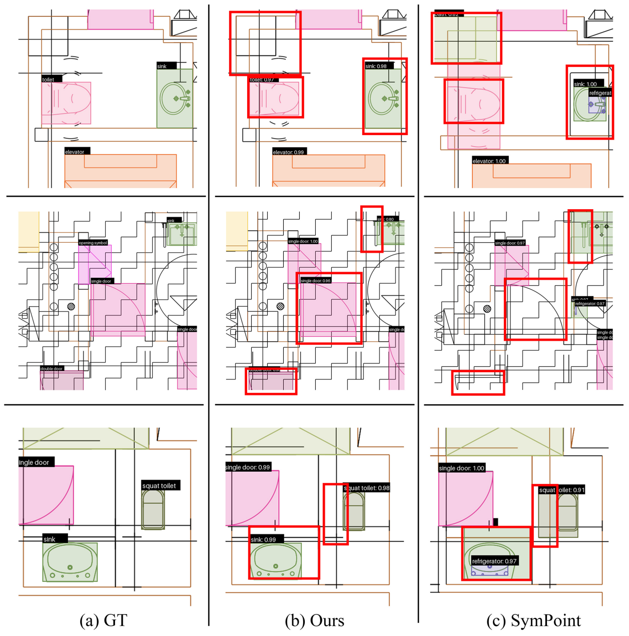


Figure 5. Qualitative comparison of panoptic symbol spotting. Our method accurately detects symbol instances, even in situations where symbols are overlapped by other elements.

dense point sampling based feature representation.

We also compare the effectiveness of Sliding Window Aggregation (SWA) and Block Partitioning (BP) methods when applied to the CADSpotting model on the LS-CAD dataset. We apply BP method using the same size as drawings the FloorPlanCAD dataset without overlapping between blocks, to divide large-scale CAD drawings into smaller blocks. When employing the BP method, we ensure that each primitive is uniquely represented by retaining only those whose starting points fall within the block. The results in Table 5 indicate that, compared to the BP method, SWA achieves the highest performance on PQ, with an absolute improvement of 10.4%, although there is a slight decrease of 1.0% on SQ. The result shows that our CADSpotting method with the SWA technique can accurately recognize instances of CAD symbols in large-scale CAD drawings.

Method	PQ	SQ	RQ
Ours + BP	42.8	**94.8**	45.2
Ours + SWA	**53.2**	93.8	**56.6**

Table 5. Quantitative comparison of semantic and panoptic symbol spotting on LS-CAD dataset.

4.4. Qualitative Comparison

We perform qualitative comparisons with SymPoint on the FloorPlanCAD dataset. Fig.5 shows that our method achieves accurate and robust panoptic symbol spotting, even in challenging scenarios involving walls, complex or overlapping symbols, and uncommon furniture symbols. This demonstrates the effectiveness of our proposed dense point sampling based primitive features. Additionally, Fig.6 presents our semantic and panoptic symbol spotting results on a large-scale CAD drawing of an office building with 28,193 primitives and 1,185 instances from our LS-CAD dataset. These results demonstrate that our CADSpotting enhanced by SWA technique, accurately identifies semantic and instance information in large-scale CAD drawings.

To evaluate model generalizability and demonstrate the utility of LS-CAD dataset, we also train models on a combined dataset consisting of FloorPlanCAD and LS-CAD. Comprehensive experimental results and analyses are pro-

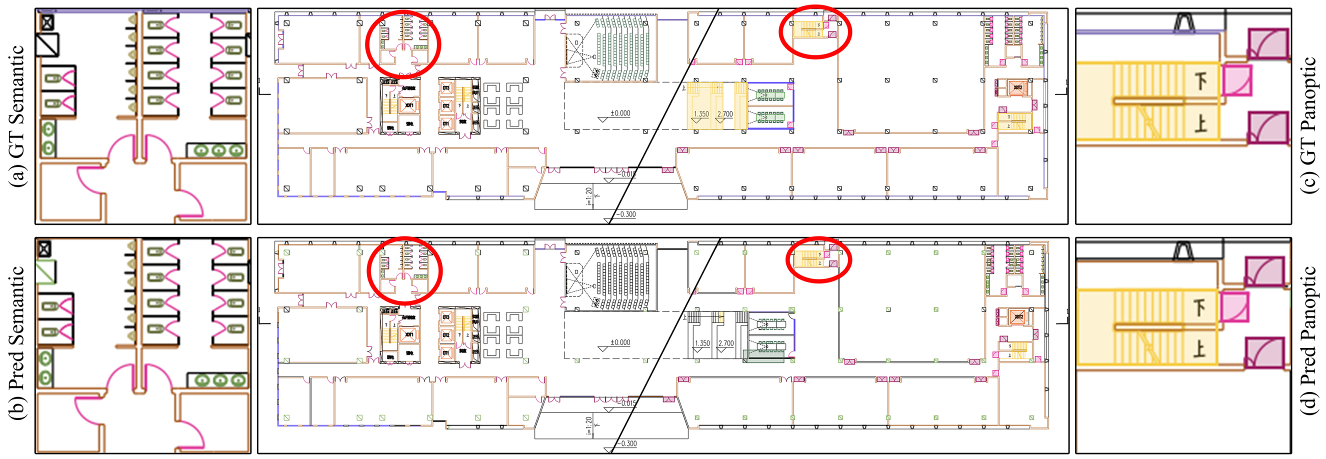


Figure 6. Semantic and panoptic symbol spotting results of CADSpotting with the SWA technique on a large-scale office building CAD drawing from our LS-CAD dataset. The figure shows full CAD drawings with GT and predicted results in the center, with closeup views of GT in (a, c) and predicted results in (b, d).

Methods	Backbone	PQ	SQ	RQ
Handcrafted Features	PTv1	78.4	88.5	88.6
Handcrafted Features	PTv3	80.2	90.2	89.0
Ours	PTv3	**88.9**	**95.6**	**93.0**

Table 6. Ablation study on feature learning methods on FloorPlanCAD dataset. We compare our dense point sampling based primitive features with the handcrafted feature representations in SymPoint. All methods utilize the same decoder architecture to learn intermediate features.

Method	PQ	SQ	RQ
w/o Color Feature	85.8	92.5	92.7
Color Feature	**88.9**	**95.6**	**93.0**

Table 7. Ablation study on RGB color information on FloorPlanCAD dataset. We compare the performance of CADSpotting with and without the inclusion of RGB color information.

vided in the supplementary material.

4.5. Ablation Studies

In this section, we present ablation studies examining the effects of feature learning and RGB information. Additional ablation studies, which explore various pooling strategies and evaluate performance across all classes, are provided in the supplementary material.

Feature Learning. We compare our dense point sampling based primitive feature learning method with the approach in SymPoint, which uses handcrafted features to represent each individual primitive. SymPoint employs Point Transformer(PTv1) [39] as its backbone network and incorporates hierarchical multi-resolution primitive features to leverage intermediate features, enhancing the decoder's performance. To evaluate the impact of different point cloud analysis methods, we first replace SymPoint's backbone with PTv3 [35], maintaining the same decoder architecture as ours to ensure a fair comparison of feature learning techniques in panoptic symbol spotting. As shown in Table.6, with the same decoder configuration, simply switching to PTv3 results in a 1.8% improvement in PQ. When we apply our proposed primitive feature learning method, performance further increases by 8.7% in PQ, demonstrating the significant performance gains achieved by our dense point sampling based feature learning approach.

RGB Color Information. We compare the performance of CADSpotting through the inclusion versus exclusion of color information in the FloorPlanCAD dataset. Table.7 shows that adding RGB information improves the PQ metric by 3.1%.

5. Conclusion

We have introduced CADSpotting, a novel method for panoptic symbol spotting in large-scale CAD drawings. Our approach uses dense point sampling within a unified point cloud framework to represent CAD primitives, combined with a Sliding Window Aggregation technique that integrates weighted voting and Non-Maximum Suppression for precise segmentation. We also present LS-CAD, a dataset of 50 annotated floorplans from diverse building types. Experiments on LS-CAD and FloorPlanCAD datasets have confirmed the effectiveness and scalability of our method, with successful applications in automated 3D reconstruction demonstrating its practical value.

Our approach currently faces limitations due to the dataset's primary emphasis on residential and commercial buildings, which restricts its applicability to industrial and cultural contexts. Additionally,it may struggle to capture significant stylistic variations among CAD symbols. Addressing these challenges through the integration of in-context learning may help enhance annotation efficiency and extend applicability. This is left as our future work.

References

[1] Liang-Chieh Chen. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 2

[3] Liang-Chieh Chen, Huiyu Wang, and Siyuan Qiao. Scaling wide residual networks for panoptic segmentation. *arXiv preprint arXiv:2011.11675*, 2020. 3

[4] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022. 3

[5] Zhiwen Fan, Lingjie Zhu, Honghua Li, Xiaohao Chen, Siyu Zhu, and Ping Tan. Floorplancad: A large-scale cad drawing dataset for panoptic. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10128–10137, 2021. 2, 3, 6, 7

[6] Zhiwen Fan, Tianlong Chen, Peihao Wang, and Zhangyang Wang. Cadtransformer: Panoptic symbol spotting transformer for cad drawings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10986–10996, 2022. 2, 3, 6, 7

[7] Ali Farhadi and Joseph Redmon. Yolov3: An incremental improvement. In *Computer vision and pattern recognition*, pages 1–6. Springer Berlin/Heidelberg, Germany, 2018. 2, 3

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2

[9] Jitesh Jain, Jiachen Li, Mang Tik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. Oneformer: One transformer to rule universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2989–2998, 2023. 2, 3

[10] Xinyang Jiang, Lu Liu, Caihua Shan, Yifei Shen, Xuanyi Dong, and Dongsheng Li. Recognizing vector graphics without rasterization. *Advances in Neural Information Processing Systems*, 34:24569–24580, 2021. 3

[11] Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6399–6408, 2019. 3

[12] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9404–9413, 2019. 3

[13] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 2

[14] Maxim Kolodiazhnyi, Anna Vorontsova, Anton Konushin, and Danila Rukhovich. Oneformer3d: One transformer for unified point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20943–20953, 2024. 3, 5

[15] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7026–7035, 2019. 3

[16] WENLONG LIU, Tianyu Yang, Yuhan Wang, Qizhi Yu, and Lei Zhang. Symbol as points: Panoptic symbol spotting via point-based representation. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 3, 4, 6, 7

[17] Wenlong Liu, Tianyu Yang, Qizhi Yu, and Lei Zhang. Sympoint revolutionized: Boosting panoptic symbol spotting with layer feature enhancement. *arXiv preprint arXiv:2407.01928*, 2024. 2, 3

[18] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 2

[19] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[20] MarsBIM. Cad to bim conversion. `https://www.marsbim.com/services/bim/cad-to-bim-conversion/`. Accessed: 2024-11-10. 2

[21] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. Ieee, 2016. 5

[22] Thi-Oanh Nguyen, Salvatore Tabbone, and O Ramos Terrades. Symbol descriptor based on shape context and vector model of information retrieval. In *2008 The Eighth IAPR International Workshop on Document Analysis Systems*, pages 191–197. IEEE, 2008. 3

[23] Thi-Oanh Nguyen, Salvatore Tabbone, and Alain Boucher. A symbol spotting approach based on the vector model and a visual vocabulary. In *2009 10th International Conference on Document Analysis and Recognition*, pages 708–712. IEEE, 2009. 3

[24] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 3

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016. 2, 3

[26] Alireza Rezvanifar, Melissa Cote, and Alexandra Branzan Albu. Symbol spotting for architectural drawings: state-of-the-art and new industry-driven developments. *IPSJ Transactions on Computer Vision and Applications*, 11:1–22, 2019. 3

[27] Alireza Rezvanifar, Melissa Cote, and Alexandra Branzan Albu. Symbol spotting on digital architectural floor plans using a deep learning-based framework. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2419–2428, 2020. 2, 3

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015. 2

[29] Marçal Rusiñol, Josep Lladós, and Gemma Sánchez. Symbol spotting in vectorized technical drawings through a lookup table of region strings. *Pattern Analysis and Applications*, 13:321–331, 2010. 2

[30] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8216–8223. IEEE, 2023. 3

[31] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019. 3

[32] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985. 5

[33] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *Advances in Neural information processing systems*, 33:17721–17732, 2020. 5, 1

[34] Huikai Wu, Junge Zhang, Kaiqi Huang, Kongming Liang, and Yizhou Yu. Fastfcn: Rethinking dilated convolution in the backbone for semantic segmentation. *arXiv preprint arXiv:1903.11816*, 2019. 2

[35] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4840–4851, 2024. 3, 4, 8

[36] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in neural information processing systems*, 34: 12077–12090, 2021. 2, 3

[37] Bingchen Yang, Haiyong Jiang, Hao Pan, and Jun Xiao. Vectorfloorseg: Two-stream graph attention network for vectorized roughcast floorplan segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1358–1367, 2023. 3

[38] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In *The Eleventh International Conference on Learning Representations*, 2022. 6

[39] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 3, 8

[40] Zhaohua Zheng, Jianfang Li, Lingjie Zhu, Honghua Li, Frank Petzold, and Ping Tan. Gat-cadnet: Graph attention network for panoptic symbol spotting in cad drawings. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11747–11756, 2022. 2, 3, 6, 7

[41] Xueyan Zou, Jianwei Yang, Hao Zhang, Feng Li, Linjie Li, Jianfeng Wang, Lijuan Wang, Jianfeng Gao, and Yong Jae Lee. Segment everything everywhere all at once. *Advances in Neural Information Processing Systems*, 36, 2024. 3

CADSpotting: Robust Panoptic Symbol Spotting on Large-Scale CAD Drawings

Supplementary Material

6. More Details on the LS-CAD Dataset

In Fig.8, we present visualizations of several samples from the LS-CAD dataset to highlight the diversity of the proposed dataset. The top panel illustrates Office 3, which spans a floor area of 2,994m^2, including 16,803 primitives and 559 instances. The bottom panel depicts Hotel 1, which spans a floor area of 1,193m^2, consisting of 26,709 primitives and 335 instances.

To demonstrate the generalization capability of our model and highlight the value of the LS-CAD dataset, we implement a cross-dataset joint training paradigm. We randomly partition complete large-scale CAD drawings into three subsets: 80% for training, 10% for validation, and 10% for testing. The division of the LS-CAD dataset strictly adheres to the standardized data splitting protocol established in FloorPlanCAD. The training and validation sets from both datasets are subsequently integrated to construct a cross-dataset collaborative training environment. As shown in Table.8, the cross-dataset joint training strategy significantly improves model performanc on the LS-CAD test set, although with slight performance degradation on the original FloorPlanCAD test set. This demonstrates the necessity of the proposed LS-CAD dataset for provding more diverse CAD representations. Notably, CADSpotting+SWA method reaches the best performance of 75.5 on the PQ metric. After cross-dataset joint training, the PQ score for CADSpotting decreases by only 1.2% on FloorPlanCAD, whereas it drops by 4.3% on SymPoint. This indicates that our model exhibits stronger generalization capabilities.

7. Quantitative Evaluation

Ablation Study on Feature Learning. We present a comprehensive evaluation of panoptic quality (PQ), segmentation quality (SQ), and recognition quality (RQ) for each class, comparing different feature learning methods. Our final approach achieves superior performance across the majority of classes, particularly excelling in commonly used categories such as door classes, window classes, and wall classes. The detailed results are shown in Table.10.

Ablation Study on Pooling. We further evaluate the necessity and impact of different pooling methods on our approach, using a baseline configuration without a pooling layer. As shown in Table.9, max pooling improves performance by 11.5%, average pooling by 13.0%, and mixed pooling by 14.3% in PQ. Based on these results, we select mixed pooling for our CADSpotting approach.

8. Qualitative Evaluation

We provide comparisons of prediction results on different datasets. For FloorPlanCAD, we compare the ground truth and prediction using their primitive colors to intuitively evaluate prediction accuracy, as shown in Fig.9, Fig.10 and Fig.13. For LS-CAD, we present the results of semantic and panoptic predictions, highlighting the performance of the method in practical scenarios, as shown in Fig.11 and Fig.12.

9. Sliding Window Implementation Details

The Algorithm.1 details the implementation process of our SWA. In the matrix-NMS [33] process for sliding window aggregation, computing the Intersection over Union (IoU) between all pairs of instance proposal masks is essential for constructing the IoU matrix. However, the number of instance proposals grows substantially with the size of the input drawing. If a dense matrix representation is used, both the time and memory complexities of this operation become prohibitively high.

To mitigate this issue, we observe that only a small fraction of proposal pairs exhibit nonzero overlap, as most proposals belong to different inference windows. Fig. 7 visualizes the sparsity structure of the IoU matrix during sliding window aggregation. By leveraging this sparsity, we reimplemented the IoU computation using sparse matrix routines, significantly reducing computational cost and memory usage without sacrificing accuracy.

10. Automated 3D Interior Reconstruction

Based on the spotting results generated by CADSpotting, we present more 3D reconstruction renderings in Fig.14. By systematically optimizing the parameters of predefined instance components using instance segmentation data and primitive spatial coordinates, we achieve accurate 3D model reconstruction through component-level geometric adaptation

For wall polygon extraction, we employ an innovative SVG-PNG conversion strategy to differentiate floor areas from walls: 1) Merge adjacent endpoints and remove duplicate segments to simplify SVG paths. 2) Convert vectors to 8K resolution binarized PNG while maintaining coordinate alignment. 3) Identify the largest connected component as the floor through connected component analysis, with secondary components as candidate walls. 4) Re-vectorize boundaries through coordinate mapping to generate closed wall polygons.

Method	Training Dataset	Test on FloorPlanCAD			Test on Cross-dataset		
		PQ	SQ	RQ	PQ	SQ	RQ
SymPoint [16]	FloorPlanCAD	83.3	91.4	91.1	33.2	83.0	39.9
SymPoint	Cross-dataset	79.0	89.2	88.6	57.0	84.0	67.8
CADSpotting	FloorPlanCAD	**88.9**	**95.6**	**93.0**	42.8	**94.8**	45.2
CADSpotting	Cross-dataset	**87.3**	**95.4**	**91.4**	**69.9**	91.6	**76.3**
CADSpotting(SWA)	FloorPlanCAD		/		53.2	**93.8**	56.6
CADSpotting(SWA)	Cross-dataset		/		**75.5**	84.9	**88.9**

Table 8. Cross-dataset joint training performance comparison between SymPoint and CADSpotting.

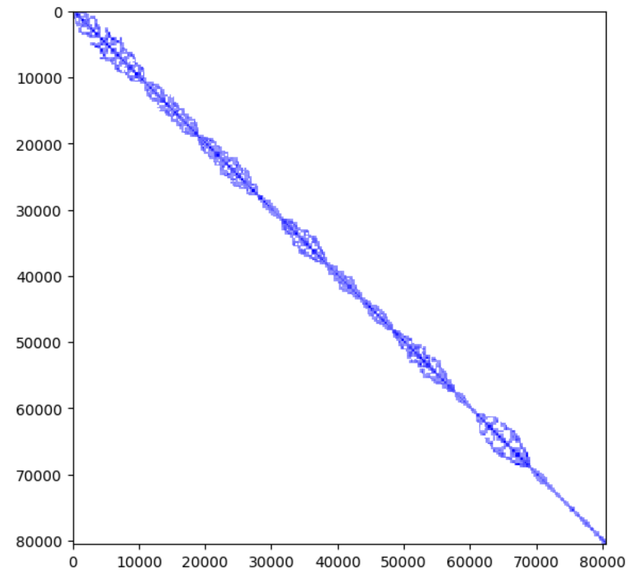


Figure 7. Visualization of the sparsity structure of the IoU matrix during sliding window aggregation for a sample drawing from the LS-CAD dataset. The Reverse Cuthill-McKee (RCM) algorithm was applied to reorder the matrix rows and columns, reducing its bandwidth and improving the visualization.

The proposed method exhibits notable robustness against recognition inaccuracies – incomplete walls are automatically reconciled during stage 1, while small noise particles are filtered in stage 3 using area thresholds.

For doors and windows, we employ category-specific parameterization strategies. Doors are systematically categorized into four distinct subtypes (single/double/sliding/folding) based on semantic labels, with positions, orientations, and pivot points determined through linear-arc spatial relationships.

Windows are grouped via union-find algorithms, calculating average lines of co-grouped segments as representative lines, then selecting the segment closest to the representative line within each group as the window instance position.

The final reconstruction leverages Blender's geometry nodes for procedural modeling: 1) Extrude wall polygons to 3D volumes using floorplan height attributes. 2) Instance-place door/window assemblies at infered positions. 3) Apply floor based on origin CAD drawing.

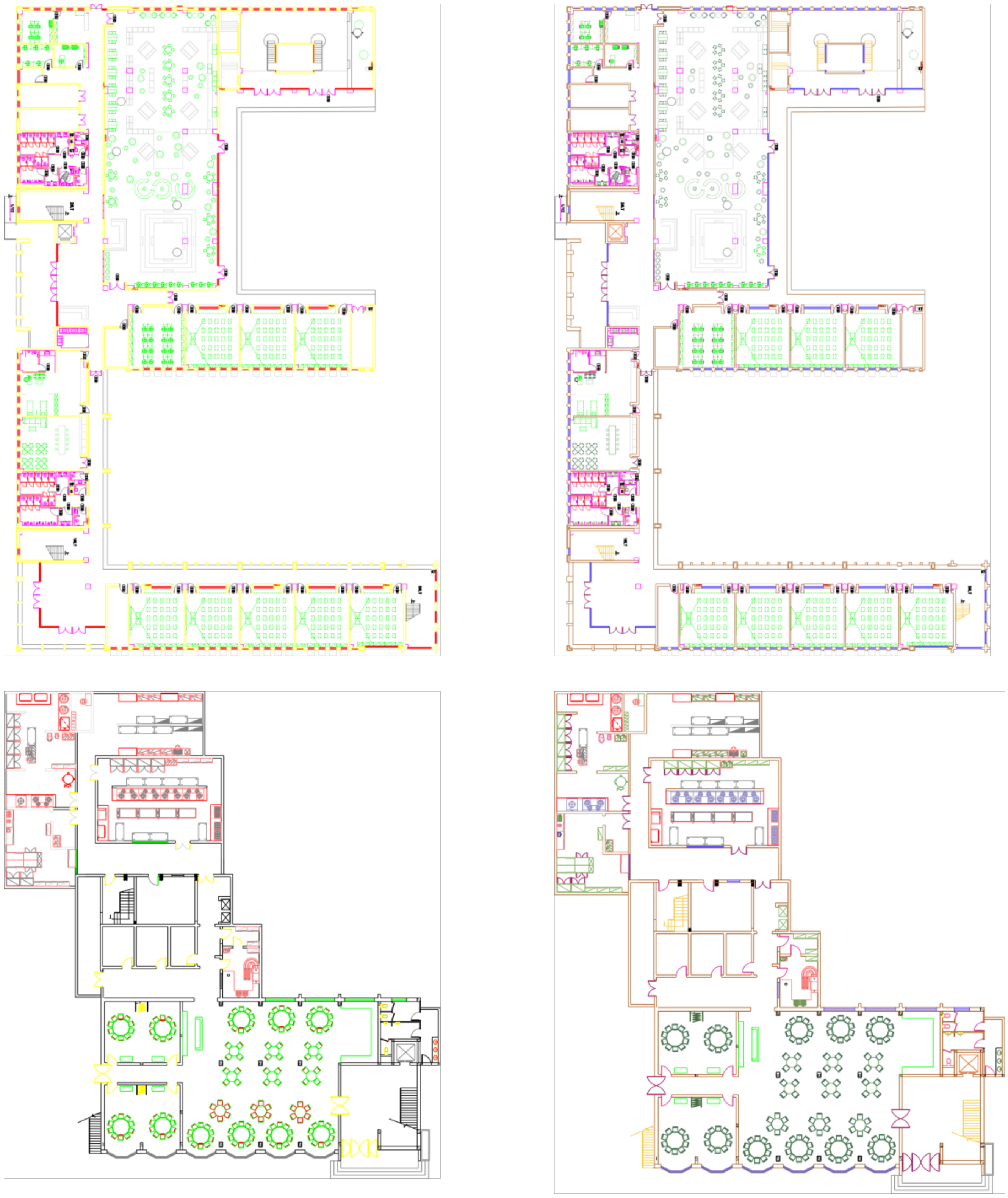
Primitive Pooling Type	PQ	SQ	RQ
w/o Pooling	74.6	89.3	83.6
Max	86.1	94.1	91.5
Average	87.6	94.5	92.7
Mixed (Max+Average)	**88.9**	**95.6**	**93.0**

Table 9. Ablation study on different pooling methods test on FloorPlanCAD dataset.

This integrated approach achieves automated 3D reconstruction with exceptional efficiency, transforming raw CAD data into structurally complete 3D interior models within minutes through streamlined procedural generation.

Algorithm 1: The pseudocode of SWA

```
Input: sliding steps, CAD drawing, proposals $p$
Output: final_sem, final_inst
1  Initialize tot_sem $\leftarrow [N, C]$
2  Initialize tot_inst $\leftarrow$ [win_num $\times p, N$]
3  foreach sliding window $w_i$ do
4      sem_i $\leftarrow$ PRED_SEMANTIC($w_i$)
5      inst_i $\leftarrow$ PRED_INSTANCE($w_i$)
6      foreach primitive Id $pId$ do
7          sem_i $\leftarrow$ ONE_HOT(sem_i[$pId$])
8          tot_sem[$pId$] $\leftarrow$ tot_sem[$pId$] + sem_i
9      end
10     tot_inst[(i-1)*$p$ : i*$p$] $\leftarrow$ inst_i
11 end
12 final_sem $\leftarrow$ ARGMAX(tot_sem)
13 final_inst $\leftarrow$ NMS(tot_inst)
```



(a) Raw Input (b) GT

Figure 8. Examples from the LS-CAD Dataset. The figure shows the raw input and ground truth of two samples in the dataset. The raw input represents the unprocessed original svg image, while the ground truth is the svg image generated based on manually annotated labels, which facilitates the visualization and comparison of results. This dataset is designed to assess the performance of CAD image processing algorithms.

class	**A**			**B**			**C**			**D**		
	PQ	SQ	RQ	PQ	SQ	RQ	PQ	SQ	RQ	PQ	SQ	RQ
single door	**91.59**	97.08	94.34	75.67	90.68	83.44	87.23	92.68	94.12	87.46	92.85	94.20
double door	**93.51**	96.93	96.47	75.14	88.76	84.65	87.30	91.85	95.04	88.27	92.73	95.18
sliding door	**96.54**	98.78	97.73	83.55	50.31	83.55	94.51	96.15	98.29	91.32	95.65	95.46
folding door	57.67	87.88	65.62	39.66	95.83	41.38	**70.88**	83.29	85.11	52.03	82.64	62.96
revolving door	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
rolling door	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
window	**89.60**	97.24	92.14	75.49	91.87	82.17	77.13	85.24	90.49	74.53	85.06	87.62
bay window	**48.15**	95.26	50.55	41.92	88.51	47.37	22.25	77.38	28.75	12.00	68.83	17.44
blind window	**90.57**	98.17	92.25	77.71	96.14	80.83	79.56	84.85	93.76	77.28	83.91	92.10
opening symbol	**46.64**	76.82	60.71	37.97	72.60	52.30	36.47	79.11	46.10	20.10	72.22	27.84
sofa	**88.53**	96.29	91.93	71.82	82.05	87.53	76.98	90.28	85.28	73.74	87.40	84.37
bed	**89.79**	92.47	97.11	78.30	83.76	93.47	76.87	86.26	89.11	76.51	84.44	90.61
chair	83.58	94.04	88.88	70.05	85.54	81.89	**84.42**	92.26	91.51	80.85	91.71	88.16
table	**75.39**	88.46	85.22	53.18	78.36	67.87	68.03	86.46	78.68	61.99	86.43	71.72
TV cabinet	**95.23**	97.57	97.60	82.85	86.79	95.46	89.99	93.41	96.33	79.94	85.50	93.49
Wardrobe	**93.76**	97.23	96.43	84.53	89.58	94.36	85.37	88.77	96.17	82.50	85.94	95.99
cabinet	**81.40**	91.16	89.28	65.83	81.45	80.82	71.37	82.66	86.33	69.15	83.42	82.90
gas stove	**97.34**	99.17	98.16	90.08	96.15	93.69	97.10	97.89	99.19	96.66	97.61	99.03
sink	**87.59**	95.21	92.00	76.13	91.10	83.57	84.88	90.98	93.30	83.42	90.03	92.65
refrigerator	**91.96**	96.24	95.55	71.86	81.87	87.78	81.70	87.44	93.44	80.28	85.58	93.80
air conditioner	**86.53**	99.02	87.39	78.66	96.36	81.64	80.59	92.04	87.56	75.45	91.70	82.27
bath	**79.30**	93.31	84.98	63.84	83.21	76.73	66.62	79.71	83.58	62.06	77.36	80.22
bath tub	**85.00**	91.92	92.47	70.96	82.90	85.61	69.94	80.14	87.27	64.82	78.13	82.96
washing machine	**89.17**	97.74	91.23	78.29	90.75	86.27	86.40	91.21	94.73	77.97	85.26	91.45
squat toilet	**95.35**	98.25	97.05	78.70	94.34	83.42	92.58	94.79	97.67	89.99	92.68	97.09
urinal	**94.32**	98.76	95.51	80.33	92.08	87.24	91.80	94.84	96.79	91.31	94.79	96.32
toilet	**95.21**	98.08	97.07	81.39	91.92	88.55	90.81	93.01	97.63	90.48	93.53	96.74
stairs	**86.47**	94.28	91.72	74.68	89.68	83.27	68.56	82.02	83.59	65.50	79.52	82.36
elevator	**96.15**	97.53	98.58	79.08	94.76	83.44	84.44	90.52	93.28	80.64	88.59	91.03
escalator	**73.73**	88.37	83.44	67.98	79.15	85.88	29.80	72.72	40.98	44.64	73.70	60.56
row chairs	85.16	94.94	89.71	82.45	92.43	89.21	**85.69**	94.53	90.65	84.45	93.99	89.86
parking spot	**89.68**	95.20	94.20	84.34	92.67	91.01	72.22	80.84	89.34	72.64	82.77	87.76
wall	**82.37**	88.83	92.73	75.08	81.61	66.81	81.86	65.29	68.74	44.48	65.44	67.97
curtain wall	**64.80**	89.88	72.09	64.17	84.45	75.98	60.25	87.36	73.90	37.62	73.21	51.39
railing	**75.67**	92.22	82.06	62.87	85.69	73.37	37.82	76.42	49.49	28.53	70.73	40.34
total	**88.91**	95.65	92.96	74.59	89.26	83.56	80.24	90.17	88.98	78.36	88.59	88.46

Table 10. Quantitative results of the ablation study for panoptic symbol spotting across different classes.
A: Dense point sampling + PTv3 + Pooling(Ours). B: Dense point sampling + PTv3. C: Handcrafted features + PTv3. D: Handcrafted features + PTv1

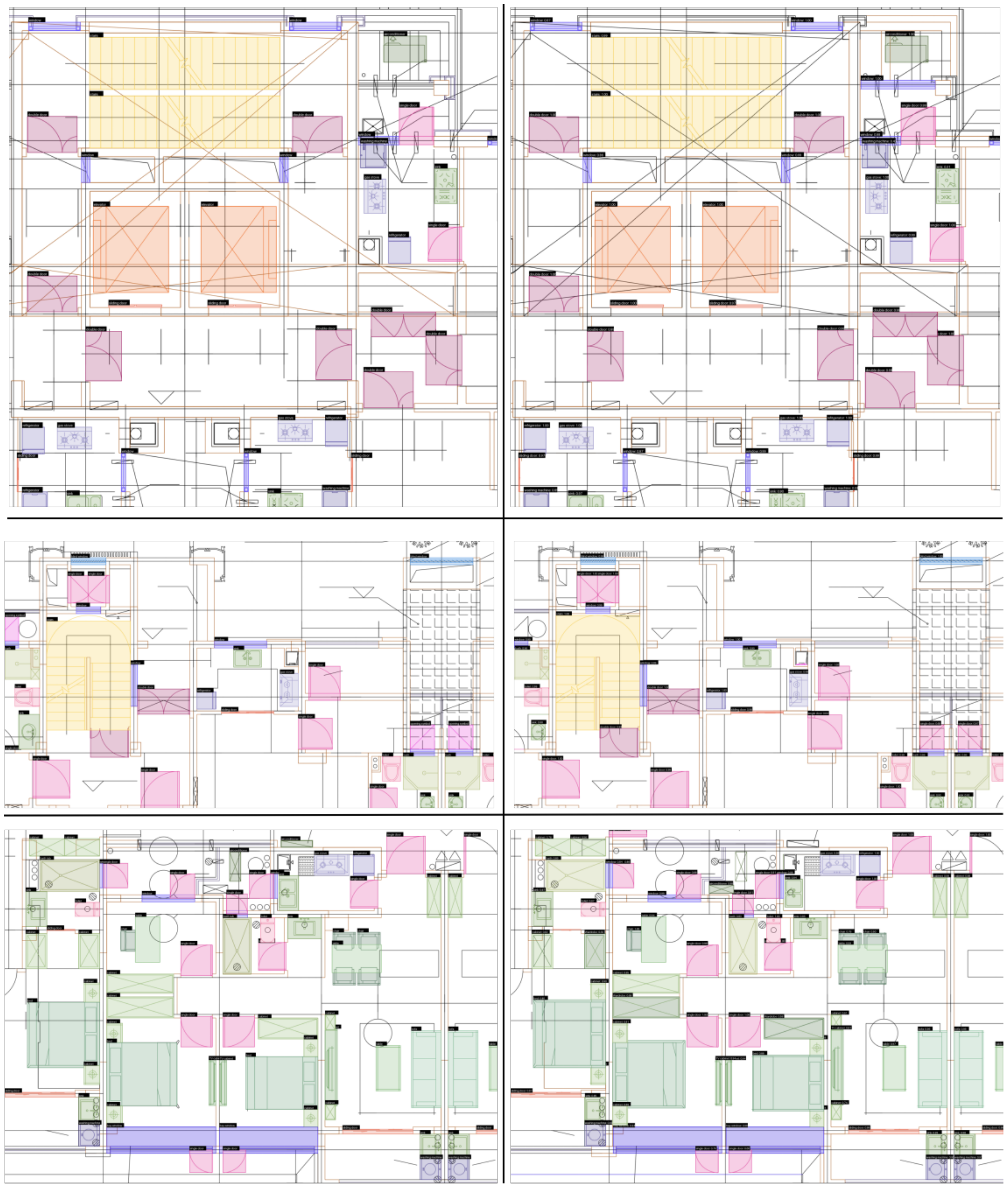


Figure 9. Qualitative comparison of panoptic symbol spotting. Our method accurately detects symbol instances, even in situations where symbols are overlapped by other elements.

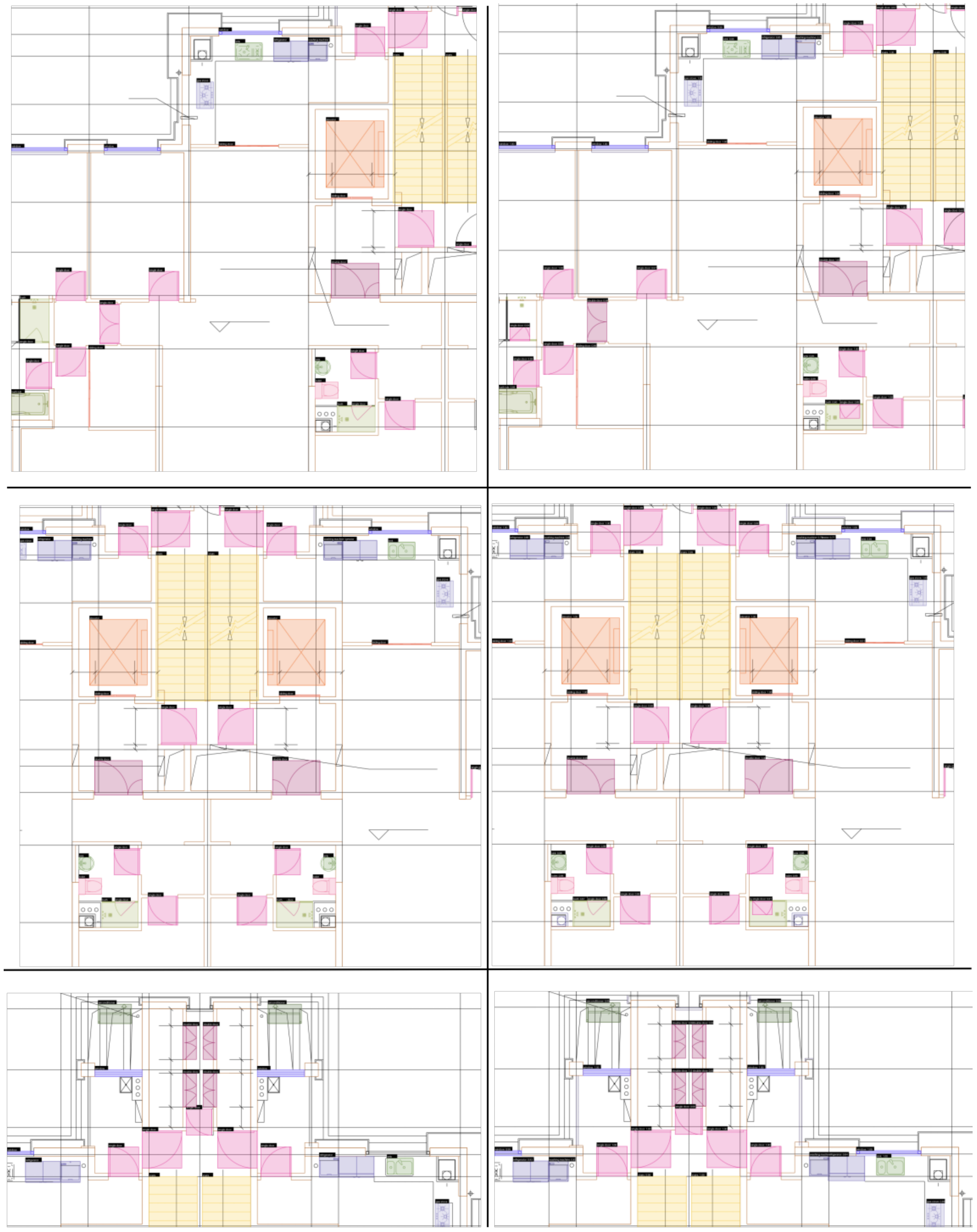
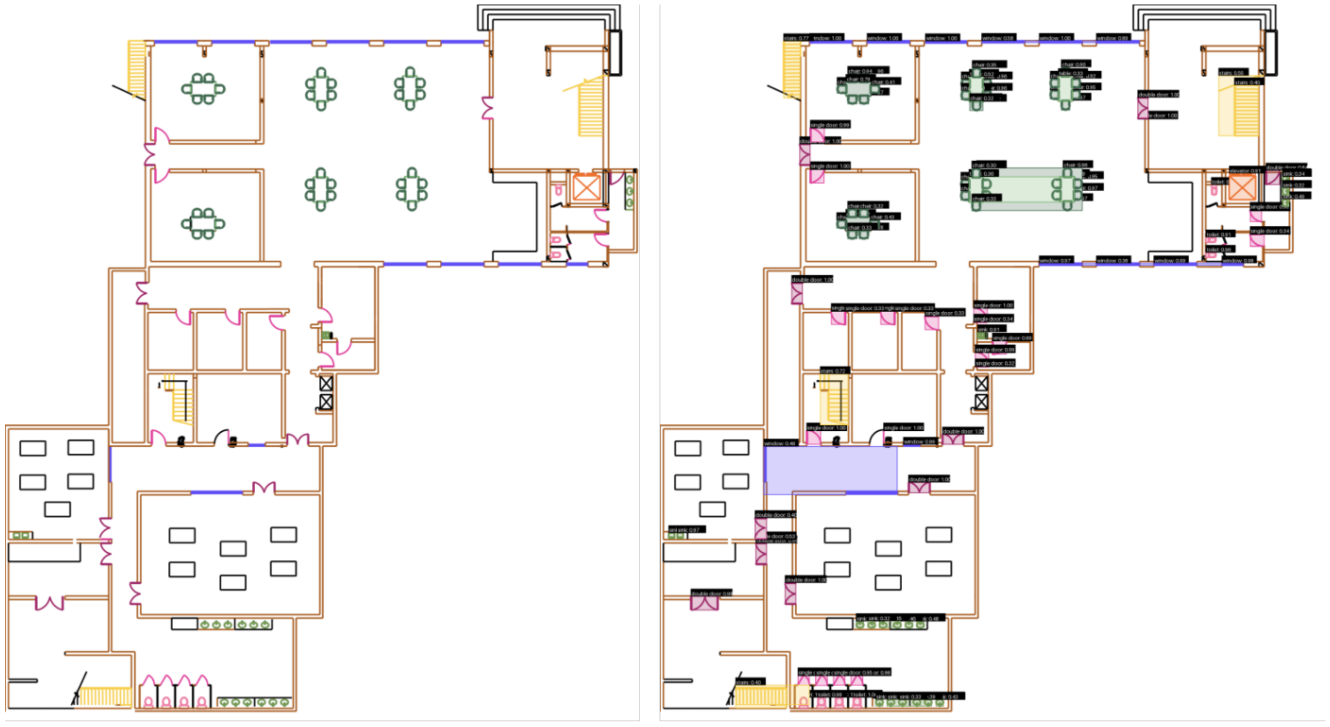
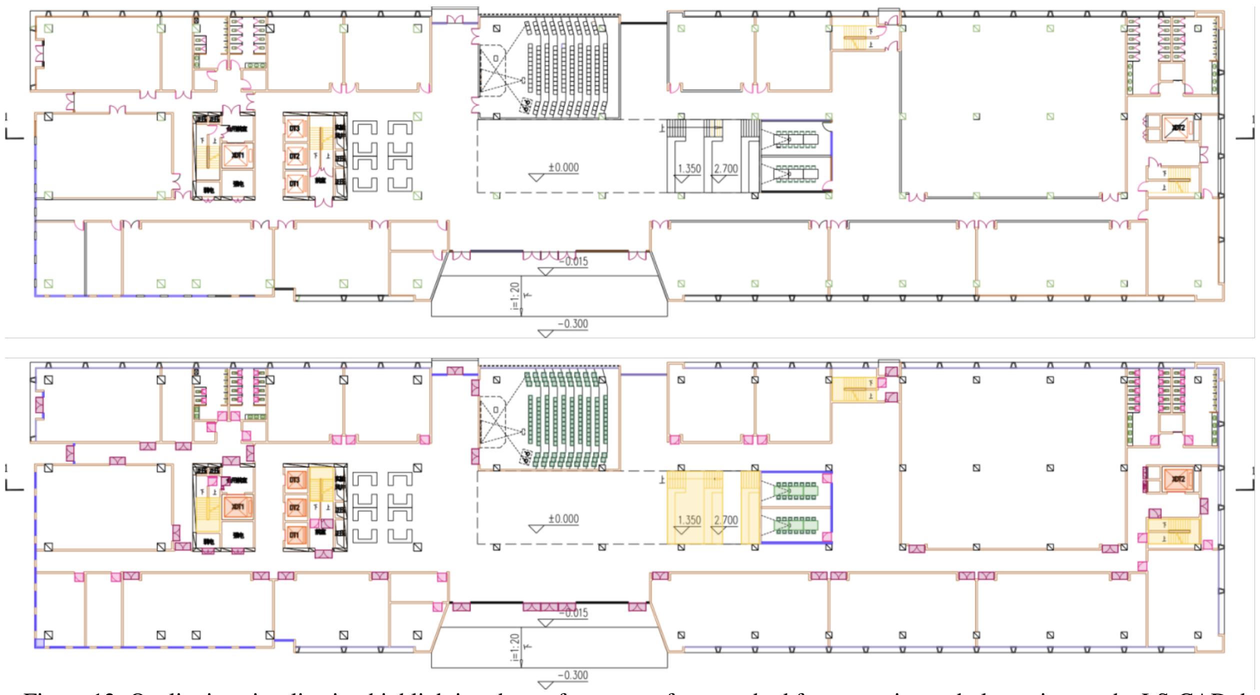


Figure 10. Qualitative comparison of panoptic symbol spotting. Our method accurately detects symbol instances, even in situations where symbols are overlapped by other elements.



(a) Semantic Pred (b) Panoptic Pred

Figure 11. Qualitative visualization highlighting the performance of our method for panoptic symbol spotting on the LS-CAD dataset.



(a) Semantic Pred

(b) Panoptic Pred

Figure 12. Qualitative visualization highlighting the performance of our method for panoptic symbol spotting on the LS-CAD dataset.

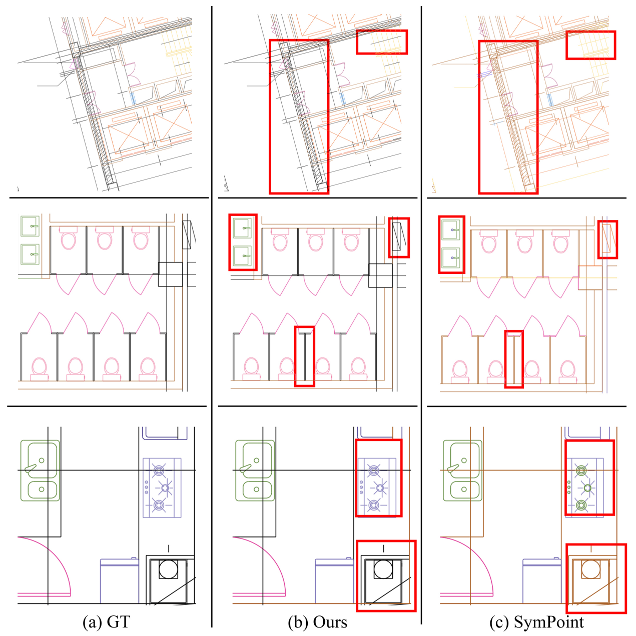
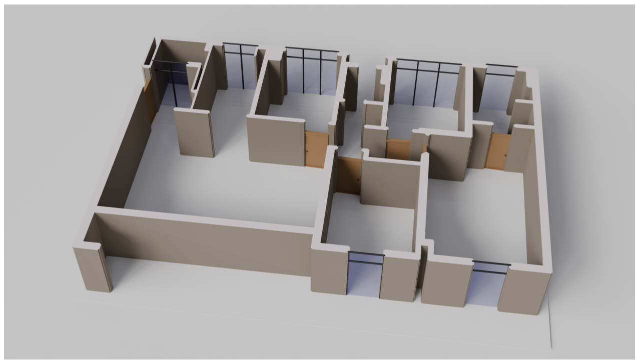

(a) GT (b) Ours (c) SymPoint

Figure 13. Qualitative comparison of semantic symbol spotting. Our dense point sampling based primitive feature learning enables our approach to achieve precise semantic segmentation, particularly on walls and complex, overlapping symbols such as faucets in sinks and uncommon furniture symbols.


(a) Automated 3D reconstruction of small-scale architectural building.

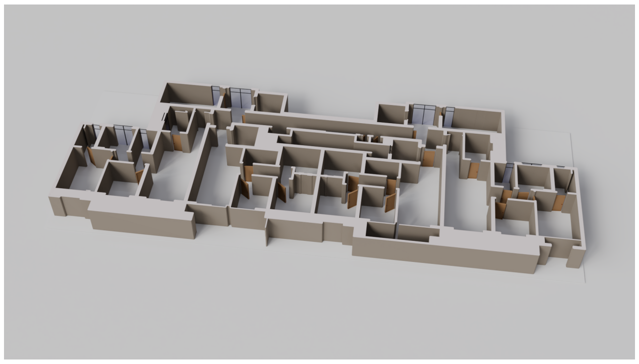

(b) Automated 3D reconstruction of large-scale architectural building.

Figure 14. Automated 3D reconstruction results.