Reconstructing 3D Wireframes from Point Clouds by Decoding Three Orthographic Blueprints

Anonymous cvm submission

Paper ID 634

Abstract

Reconstructing compact and editable 3D CAD wireframes from raw point clouds remains a formidable challenge in computer vision and graphics. Existing methods that directly operate in 3D space often struggle with issues like noise sensitivity, intricate post-processing, and a neglect of vital geometric constraints inherent to CAD design. To overcome these limitations, we introduce a novel framework that leverages the classical principle of orthographic projections to reformulate the 3D problem into a series of 2D analysis tasks. Our approach begins by extracting edge points from the 3D point cloud and projecting them onto three principal planes to generate front, top, and left views. A key contribution is PC-NET, an end-to-end network that takes segmented 2D point sets from these views as input. PC-NET simultaneously performs parametric primitive fitting through affine transformations of template geometries and predicts the geometric constraints between primitives, such as coincident, parallel, and perpendicular relationships. This process yields three high-quality, constraint-rich 2D CAD sketches. Finally, a robust merging algorithm integrates these three views to reconstruct a precise, compact, and well-constrained 3D CAD wireframe. Extensive qualitative and quantitative experiments demonstrate that our method surpasses current state-of-the-art techniques in both accuracy and robustness, effectively bridging the gap between unorganized point clouds and structured CAD models.

Keywords: ***3D Reconstruction, Point Cloud, CAD Wireframe, Three-View Projection, Primitive Fitting, Geometric Constraints***

1. Introduction

The proliferation of 3D sensing technologies such as LiDAR and structured-light scanners has made point cloud acquisition faster and more accessible than ever. However, the reverse engineering of high-quality, editable 3D CAD models from such unorganized point data remains a significant and unsolved challenge in computer vision and digital geometry processing. The transition from raw point sets to structured CAD representations is crucial for applications in industrial design, quality inspection, and additive manufacturing, yet it continues to elude fully automated and robust solutions.

Existing approaches to this problem can be broadly classified into two categories based on their use of edge guidance. Methods such as HP-Net [29], ParSeNet [30], and Point2CAD [31] belong to the first category, which directly segments point clouds and fits surfaces to each region. While conceptually straightforward, these techniques are highly sensitive to point cloud quality, often failing in the presence of noise, outliers, or missing data. Moreover, they rely heavily on the notoriously difficult task of instance-level point cloud segmentation, which frequently leads to inaccurate boundary definitions and imperfect surface fitting.

To circumvent these issues, a second category of methods has emerged, focusing on reconstructing the 3D wireframe—the structural skeleton of the CAD model—as an intermediate representation. Techniques like PIE-Net [6], WireframeNet [27], and SED-Net [26] first identify edge points in the input point cloud and then construct a wireframe based on these points. This wireframe serves as a structural prior to guide subsequent surface reconstruction, significantly improving robustness. Despite these advances, such methods still face considerable challenges in post-processing, including complex surface trimming and error-prone wireframe correction. More critically, they largely ignore the rich set of geometric constraints—such as *Parallel*, *Perpendicular*, and *Coincident* relationships—that are fundamental to CAD design and essential for generating valid, editable models.

In this paper, we introduce a novel and efficient framework that bridges the gap between raw point clouds and high-quality 3D CAD wireframes by explicitly incorporating both geometric primitives and their constraints. Confronted with the difficulty of directly reasoning about 3D wireframes and the scarcity of annotated 3D constraint data, we propose an innovative strategy: reformulating the 3D

reconstruction task as a problem of *multi-view 2D sketch analysis*. By projecting 3D edge points onto three orthogonal planes, we transform the problem into analyzing three standard 2D CAD sketches, for which strong datasets and methods exist.

A critical prerequisite for accurate sketch analysis is robust segmentation of the projected 2D edge points. To address this, we leverage the powerful Segment Anything Model (SAM) [35] and fine-tune it for the specialized task of instance-level segmentation in CAD sketch projections. Recognizing the lack of appropriate training data for this specific domain, we constructed a large-scale dataset of annotated 2D CAD sketch projections, containing over 50,000 instances with meticulously labeled geometric primitives. This carefully curated dataset enables the fine-tuned SAM to achieve superior segmentation performance on orthographic projections of 3D edge points, effectively distinguishing individual geometric primitives in complex sketch layouts.

Recent years have witnessed notable progress in 2D CAD sketch understanding. Vitruvion [13], for instance, employs an autoregressive model to sequentially generate primitives and constraints. While powerful, it is prone to error accumulation and is computationally intensive. PPI-Net [5] frames primitive detection as a set prediction problem using the DETR architecture [25], but its reliance on Hungarian matching for label assignment can lead to training instability and suboptimal performance, particularly on small or complex primitives. Very recently, the work of [28] demonstrates the promising application of Vision Transformers in parsing parametric primitives from CAD sketches, achieving improved sequence modeling capability; however, it still follows an autoregressive generation paradigm and does not explicitly model geometric constraints.

To overcome these limitations, we present *PC-NET*—a unified network architecture for joint **P**rimitive parameterization and **C**onstraint estimation. PC-NET directly consumes the segmented 2D point sets from our SAM-based segmentation network and simultaneously infers parametric primitives (lines, arcs, circles) and their geometric relationships. This holistic analysis produces clean, well-constrained 2D sketches from each view. These are then fused through a novel merging algorithm to recover a consistent, compact, and fully constrained 3D CAD wireframe, all without manual intervention.

Our main contributions are summarized as follows:

- A novel, three-view-based reconstruction framework that effectively transforms 3D point cloud wireframe recovery into a sequence of 2D sketch analysis tasks, simplifying the problem and leveraging rich 2D annotation resources.
- A specialized SAM-based segmentation network, fine-tuned on our newly constructed dataset of 2D CAD sketch projections, for accurate instance-level segmentation of orthographically projected edge points.
- An end-to-end network, PC-NET, capable of jointly performing accurate primitive fitting and robust constraint estimation from 2D point sets, overcoming limitations of existing autoregressive and detection-based methods.
- A practical and efficient method for integrating the analyzed three-view sketches into a high-quality 3D parametric wireframe, demonstrating significant improvements in accuracy, completeness, and structural validity over current state-of-the-art techniques.

1.1. CAD Wireframe Reconstruction

The problem of reconstructing CAD wireframes from 3D data has evolved significantly with advances in deep learning and computer vision. Early approaches like PIE-Net [6] pioneered the parametric extraction of point cloud edges by leveraging single curve parameterization knowledge, establishing a new direction for learning-based wireframe reconstruction. This work inspired several subsequent developments: PC2WF [20] improved upon this foundation by operating in an end-to-end manner, encoding complete point clouds into latent representations and predicting connectivity between detected corner points. However, these methods often struggle with complex topological structures and require careful handling of corner detection accuracy.

More recent approaches have explored alternative paradigms. NerVE [17] bypasses explicit point cloud classification by directly predicting edges through a unified free-form curve representation, offering flexibility in handling multiple primitive types but sometimes sacrificing parametric precision. DEF [16] takes a different approach by regressing distances from points to feature curves within local blocks, enabling sharp curve prediction but requiring careful patch selection and integration. The neural rendering revolution has also influenced this domain, with NEF [18] building upon NeRF [19] to predict CAD feature curves from multiple calibrated images, though this introduces dependency on multi-view setups.

Recent advancements have further accelerated this domain by embracing explicit 3D representations. Most notably, EdgeGaussians [41] leverages 3D Gaussian Splatting for highly efficient edge field reconstruction from images, significantly improving speed over NeRF-based methods. However, it remains a two-stage pipeline, first reconstructing a dense edge point cloud before fitting curves, thus inheriting the limitations of stage-separation. Concur-

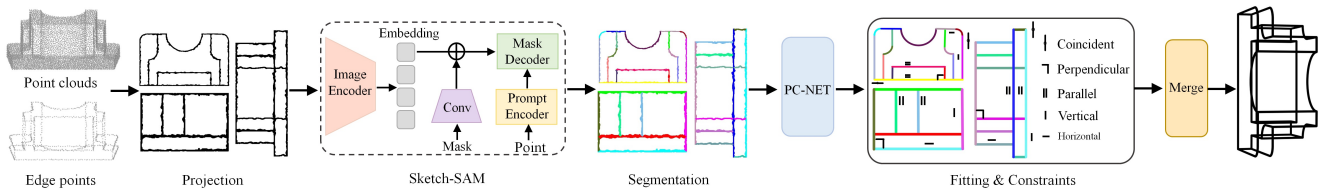

Figure 1. Overall pipeline of the proposed framework. Starting from an input point cloud, edge points are first classified and then projected onto three orthogonal views. Our fine-tuned SAM-based segmentation network processes these projections to extract instance-level segments, and the resulting segments are fed into our PC-NET for joint primitive and constraint parsing. Finally, a merging algorithm integrates the three-view analyses to reconstruct a high-quality 3D CAD wireframe.

rently, EMAP [42] introduces a "cluster-then-fitting" strategy within a neural implicit framework, enhancing edge localization through Unsigned Distance Function (UDF) modeling and unbiased equation rendering. While these methods represent significant progress, they remain fundamentally constrained by the two-stage paradigm where curve fitting quality depends heavily on the initial edge reconstruction.

Unlike these methods that operate directly in 3D space, our approach introduces a paradigm shift by leveraging 2D sketch analysis through orthogonal projections. This allows us to benefit from well-established 2D computer vision techniques while maintaining the structural information necessary for accurate 3D reconstruction. Furthermore, our method uniquely incorporates geometric constraint estimation as an integral part of the reconstruction process, addressing a critical aspect often overlooked in prior work.

1.2. CAD Model Reconstruction

CAD model reconstruction encompasses broader challenges beyond wireframe extraction. Traditional methods like those employing alpha shapes [21] provided early solutions for mesh reconstruction from point clouds but often lacked the structural awareness required for CAD applications. The introduction of neural networks revolutionized this field, enabling more sophisticated reconstruction strategies.

Primitive-based approaches have shown particular promise. Point2Cyl [34] reconstructs CAD models by detecting and stretching cylindrical components, while methods like [36, 37] utilize CSG operations to assemble complex models from simple primitives. These approaches excel at capturing regular geometric structures but may struggle with free-form surfaces and complex constraints.

The sketch-extrusion paradigm represents another significant direction. SECAD-Net [22] employs self-supervised learning with sketch-based extrusion operations, demonstrating the power of leveraging CAD modeling principles in reconstruction. Similarly, Free2CAD [39] parses input strokes into command sequences, bridging the gap between raw input and parametric CAD operations. These methods acknowledge the procedural nature of CAD design but often require sequential processing that can accumulate errors.

Recent trends in neural representation have opened new possibilities for direct CAD reconstruction. Methods like NEF [18] demonstrate that neural radiance fields can be repurposed for precise curve extraction, while EMAP [42] shows improved edge localization through hybrid implicit-explicit representations. The introduction of 3D Gaussian Splatting [45] has enabled unprecedented rendering speeds, with EdgeGaussians [41] adapting this framework specifically for edge reconstruction. However, these approaches still rely on intermediate representations rather than direct parametric curve optimization.

Recent advances in boundary representation (B-Rep) recovery, such as ComplexGen [23], aim to reconstruct editable CAD models directly from point clouds using CNN-based architectures. Meanwhile, methods like [40] explore signed distance function approximations for editable model reconstruction. While these approaches move closer to practical CAD applications, they often face challenges in recovering precise geometric constraints and maintaining topological consistency.

Our work positions itself within this landscape by focusing specifically on high-quality wireframe reconstruction as a fundamental step toward complete CAD model recovery. By ensuring accurate geometric relationships and parametric precision at the wireframe level, we provide a solid foundation for subsequent surface modeling and solid operations.

1.3. Sketch Analysis and Understanding

The analysis of 2D sketches has emerged as a critical component in bridging visual information and geometric understanding. Large-scale datasets like SketchGraphs [12] have enabled data-driven approaches to parametric CAD sketch analysis, providing the foundation for numerous subsequent works.

Generative approaches have shown remarkable capabilities in sketch understanding. Vitruvion [13] employs autoregressive models to extract parametric primitives and

their constraints from hand-drawn sketches, demonstrating the power of sequence generation for structured output. However, the sequential nature of these models can lead to error accumulation and computational inefficiency. Detection-based methods like PPI-Net [5] adapt object detection frameworks to primitive extraction but may struggle with small primitives and precise constraint relationships due to their reliance on Hungarian matching and set prediction.

The application of foundation models to sketch analysis has opened new possibilities. Works like [10] demonstrate zero-shot sketch-based retrieval using CLIP [11], while Sketch-Primitive [1] explores parametric primitive matching for classification tasks. These approaches highlight the potential of large-scale pre-training for geometric understanding.

Contemporary developments in differentiable rendering have created new pathways for sketch-based reconstruction. DiffVG [46] pioneered differentiable rasterization of vector graphics, enabling gradient-based optimization of 2D parametric curves. This paradigm has been extended to 3D through works like 3Doodle [43] and Diff3DS [44], which project 3D Bézier curves to 2D for differentiable rendering. However, these methods typically focus on sketch generation rather than reconstruction from observed imagery, and their projection-based approach can limit 3D consistency.

Concurrent research has also explored various aspects of sketch parsing, including constraint discovery [15], language modeling for CAD sequences [32], and constrained sketch generation [33]. DeepCAD [14] takes a particularly relevant approach by converting point clouds into parametric sequences through sketch-based operations, though it focuses on generation rather than reconstruction.

Our work advances this field by developing a unified framework that combines the strengths of template-based primitive fitting with learned constraint estimation. Unlike generative approaches that may suffer from error propagation, or detection-based methods that can miss fine details, our method provides direct, precise primitive parameterization while explicitly modeling the geometric relationships essential for CAD applications. The integration of SAM-based segmentation further demonstrates how foundation models can be effectively adapted for specialized domains like engineering sketch analysis.

2. Method

Our framework follows a structured five-stage pipeline to reconstruct high-quality 3D CAD wireframes from point clouds, as illustrated in Figure 1. First, edge points are extracted from the input point cloud. Second, these 3D edge points are projected onto three orthogonal planes to generate standard engineering views. Third, we employ a fine-tuned segmentation network based on SAM to perform instance-level segmentation of the projected 2D points. Fourth, the segmented point sets are processed by our novel PC-NET to generate parameterized primitives and their geometric constraints, forming high-quality 2D CAD sketches. Finally, the three-view sketches are integrated and optimized to recover a complete and well-constrained 3D wireframe.

2.1. Edge Point Extraction

The initial stage of our pipeline identifies the structurally significant edge points from the input point cloud. We leverage established point cloud classification networks that have demonstrated exceptional proficiency in edge point identification. Specifically, we build upon the capabilities of PIE-Net [6], WireframeNet [27], and SED-Net [26], which employ advanced deep learning architectures to classify each point in the cloud as either belonging to an edge or a surface region. This edge classification provides the fundamental geometric elements for subsequent processing stages, focusing our analysis on the most semantically meaningful portions of the point cloud.

2.2. Three-View Orthographic Projection

To transform the 3D edge analysis problem into more manageable 2D tasks, we project the identified 3D edge points onto three orthogonal coordinate planes. This generates standard engineering views: front view (X-Y plane), top view (X-Z plane), and left view (Y-Z plane). The projection process follows conventional orthographic projection principles, where each 3D point (x, y, z) is mapped to 2D coordinates by discarding one coordinate dimension: (x, y) for front view, (x, z) for top view, and (y, z) for left view. This multi-view representation effectively captures the complete geometric information of the 3D structure while enabling us to leverage powerful 2D computer vision techniques for sketch analysis.

2.3. SAM-based Edge Segmentation

A critical challenge in processing the projected 2D points is robust instance segmentation to distinguish individual geometric primitives. To address this, we fine-tune the Segment Anything Model (SAM) [35] specifically for CAD sketch segmentation. We constructed a specialized dataset containing over 50,000 annotated 2D CAD sketch projections with meticulously labeled geometric primitives. This dataset enables SAM to learn the distinctive characteristics of engineering sketches, including line segments, circular arcs, and complete circles. The fine-tuned model achieves superior performance in segmenting the orthographic projections of 3D edge points into coherent primitive instances, providing clean input for subsequent parametric analysis.

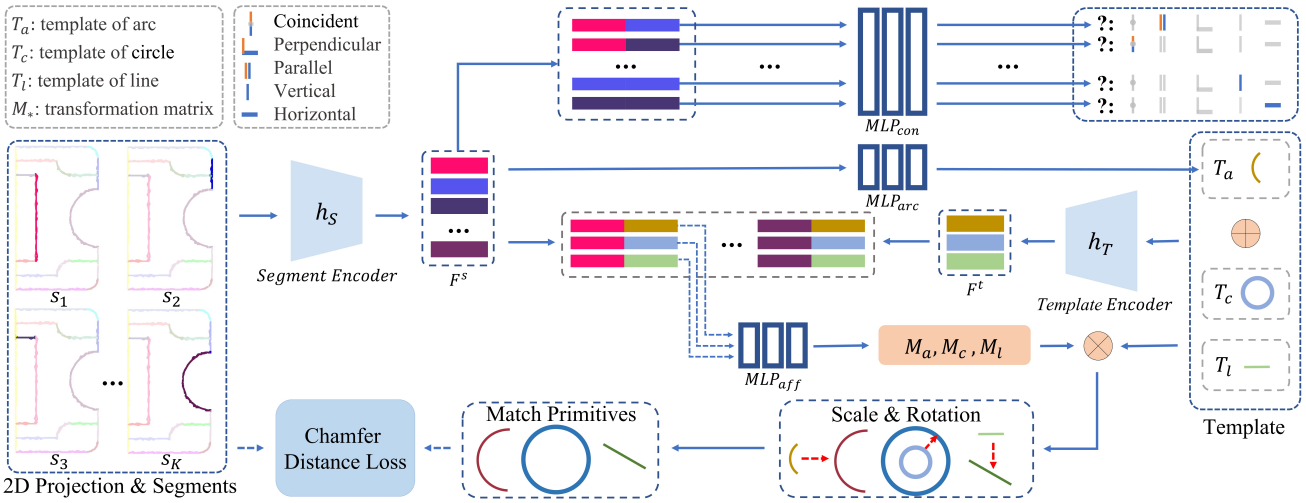

Figure 2. **PC-NET Architecture.** Our network takes segmented 2D projections as input and addresses the primitive fitting problem through affine transformations of template primitives. Simultaneously, PC-NET predicts geometric constraints between primitives, including Coincident, Parallel, Perpendicular, Horizontal, and Vertical relationships.

2.4. PC-NET: Primitive and Constraint Estimation

The core of our 2D sketch analysis is PC-NET, an end-to-end network that jointly performs parametric primitive fitting and geometric constraint estimation. As shown in Figure 2, PC-NET takes the segmented point sets $S = \{s_i\}_{i=1}^{K}$ as input, where s_i represents a segmented primitive instance and K denotes the total number of segments.

2.4.1 Segment and Template Encoding

Each segmented point set s_i is processed by a Segment Encoder h_s composed of 6 Transformer layers with 8 self-attention heads each and a feedforward dimension of 512. This encoder extracts rich feature representations:

$$F^s = h_s(S), \tag{1}$$

where F^s denotes the encoded features of all segments.

Simultaneously, we maintain parameterized templates for fundamental geometric primitives: circle template T_c, line template T_l, and arc template T_a. The arc template addresses the challenge of variable arc lengths through a dedicated Arc Prediction Module MLP_{arc}:

$$T_a = MLP_{arc}(F^s), \tag{2}$$

where T_a represents an arc template with unit radius centered at the origin.

These templates are encoded through a Template Encoder h_T with identical architecture to h_s:

$$F^t = h_t(T_c, T_l, T_a), \tag{3}$$

where F^t represents the template features.

2.4.2 Primitive Fitting via Affine Transformation

To align the fixed templates with variously oriented and scaled segments, we predict affine transformation matrices using an Affine Transformation Module MLP_{aff}:

$$M = MLP_{aff}(F^s, F^t), \tag{4}$$

where $M = \{M_c, M_l, M_a\}$ contains transformation matrices for each template type.

We then compute the optimal primitive match by evaluating the Chamfer Distance between transformed templates and input segments:

$$\mathcal{L}_{fit} = CD(M_* \otimes T_*, s), \tag{5}$$

where $\otimes$ denotes the application of affine transformation.

2.4.3 Geometric Constraint Analysis

Beyond primitive fitting, PC-NET explicitly models geometric relationships between primitives. We design a Constraint Prediction Module MLP_{con} that estimates five constraint types: Coincident, Perpendicular, Parallel, Vertical, and Horizontal, plus a null type indicating no constraint:

$$\hat{C} = \prod_{i=1}^{K} \prod_{j=1}^{K} MLP_{con}(F_i^s, F_j^s), \tag{6}$$

$$\mathcal{L}_{con} = CE(\hat{C}, C), \tag{7}$$

where CE denotes cross-entropy loss between predicted constraints $\hat{C}$ and ground truth C.

2.4.4 Multi-Task Learning

PC-NET jointly optimizes both objectives through a weighted combination:

$$\mathcal{L} = w_{fit} \times \mathcal{L}_{fit} + w_{con} \times \mathcal{L}_{con}, \tag{8}$$

where w_{fit} and w_{con} balance the contribution of each loss term.

2.5. Three-View to 3D Wireframe Reconstruction

The final stage integrates the analyzed three-view sketches into a coherent 3D wireframe. Let $F = \{f_i\}_{i=1}^{K_f}$, $T = \{t_i\}_{i=1}^{K_t}$, and $L = \{l_i\}_{i=1}^{K_l}$ represent the primitive sets for front, top, and left views respectively, with corresponding 2D point sets P^f, P^t, and P^l.

2.5.1 3D Point Recovery

We reconstruct 3D points by exploiting the complementary nature of orthographic projections. For each 2D point in one view, we identify corresponding points in the other two views based on shared coordinate values. As shown in Figure 3, a top view point p_i^t and left view point p_j^l with matching x-coordinates can be combined to form a 3D point $p_i^w = (x_1, y_1, z_1)$. For circular primitives (Figure 4), we sample four representative points to ensure robust reconstruction despite their degenerate projections in alternate views.

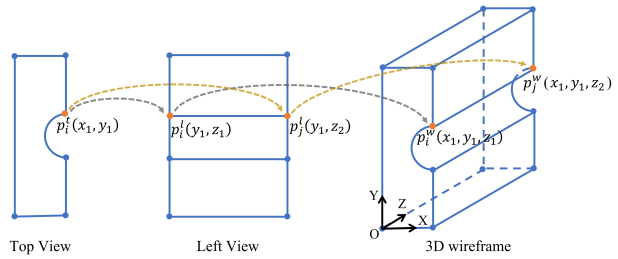

Figure 3. 3D vertex reconstruction through correspondence establishment across orthogonal views.

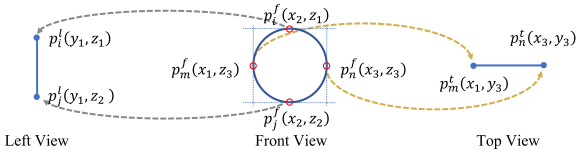

Figure 4. Recovery process for circular elements in 3D wireframe construction.

2.5.2 Initial Wireframe Construction

The transformation from recovered 3D points to a structured wireframe represents a critical phase in our reconstruction pipeline. The foundation of this process lies in establishing precise 3D vertex correspondences across orthogonal views. As illustrated in Figure 3, we employ a robust matching algorithm that identifies corresponding 2D points across different views based on coordinate consistency. For instance, a point in the top view and its corresponding point in the left view share identical x-coordinates, enabling accurate 3D coordinate computation through multi-view geometric constraints.

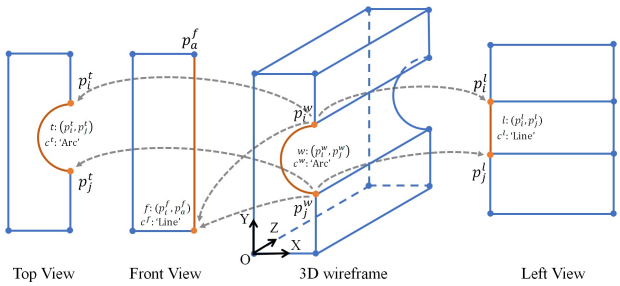

Figure 5. Primitive type determination for 3D wireframe segments through multi-view analysis.

We establish connectivity between 3D points through a multi-view consistency criterion: two 3D points are considered connected if and only if they are associated with the same 2D primitive instance across all three orthogonal views. This strict requirement ensures that only geometrically valid connections are established, effectively leveraging the complementary nature of orthographic projections to resolve ambiguities that would be challenging in single-view analysis.

The determination of primitive types follows a carefully designed hierarchical strategy that reflects the projective geometry of CAD models:

1. **Circle Identification**: A 3D primitive is classified as a circle if any of its associated 2D projections manifests as a complete circle. This prioritization acknowledges the distinctive projective behavior of circular features, which when viewed along their axis, project as lines in two views but maintain their circular character in the perpendicular view. As demonstrated in Figure 4, circular elements require special handling due to their degenerate projections in non-perpendicular views, where they may be obscured by other linear primitives.

2. **Arc Identification**: If no circular projections are present but at least one view displays an arc segment, the 3D primitive is classified as an arc. This accounts for situations where circular features are partially occluded or represent non-full circular segments. The arc classification preserves the curved nature while acknowledging the partial coverage.

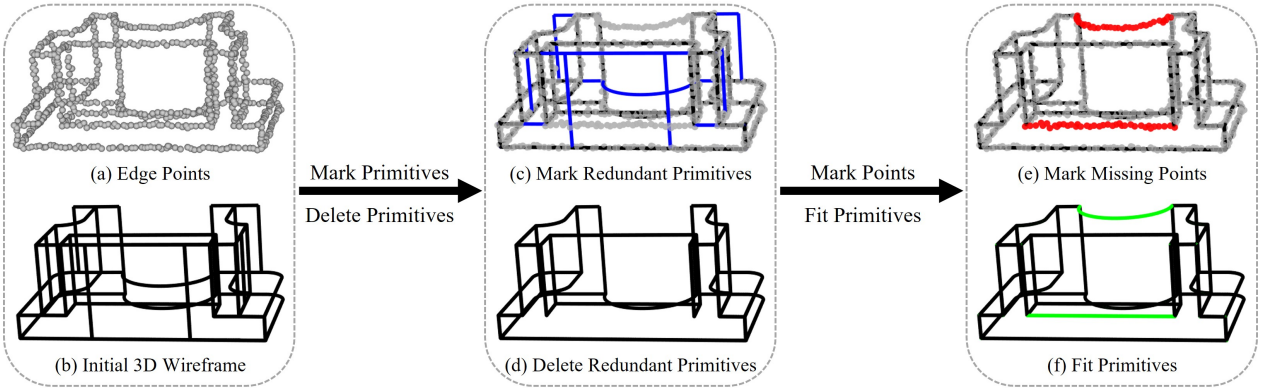

Figure 6. Wireframe optimization workflow: (a-b) alignment, (c) redundancy identification, (d) redundancy removal, (e) incompleteness detection, (f) completeness enhancement.

3. **Line Identification**: Only when all three views consistently represent the primitive as straight line segments do we classify it as a linear element. This conservative approach ensures that curved elements are not mistakenly simplified into linear approximations.

As illustrated in Figure 5, this hierarchical reasoning effectively captures the projective characteristics inherent in engineering drawings, where 3D geometric intent is encoded through multiple 2D representations. The figure demonstrates how a single 3D primitive can manifest differently across views, and how our method leverages these variations for accurate type determination. The methodology is particularly robust for handling complex cases such as helical curves, oblique circles, and partial cylindrical surfaces that frequently appear in mechanical components.

Following the initial connectivity establishment, we implement a graph-based refinement process to eliminate spurious elements. Considering the wireframe as a 3D graph $G = (V, E)$ where vertices V represent 3D points and edges E represent primitive segments, we analyze the vertex degrees and primitive associations. Vertices with degree less than 3 that are exclusively associated with line primitives are identified as redundant artifacts, typically arising from over-segmentation or noise in the projection analysis. These vertices and their associated edges are systematically removed, significantly improving the structural clarity of the wireframe while preserving genuine geometric features.

2.5.3 Wireframe Optimization

The initial wireframe construction, while structurally sound, often requires refinement to achieve CAD-quality results. As comprehensively demonstrated in Figure 6, we employ an iterative optimization framework that treats the original 3D edge points as valuable prior knowledge for geometric validation and enhancement.

The optimization begins with spatial alignment (Figure 6a-b) to address potential discrepancies in scale, position, and orientation between the reconstructed wireframe and the input point cloud. We compute an optimal affine transformation that minimizes the Hausdorff distance between the wireframe and the edge points, ensuring proper registration for subsequent analysis. This alignment step is crucial for handling cases where projection uncertainties or sampling variations introduce systematic offsets.

We then establish a bidirectional consistency measure between the wireframe primitives and the edge points, with the identification phase visualized in Figure 6c:

- **Point-to-Primitive Association**: For each edge point p_i, we compute its minimum distance to all wireframe primitives. Points within a threshold distance t (typically set to 0.5-1.0% of the model's bounding box diagonal) are marked as supporting evidence for the nearest primitive.
- **Primitive-to-Point Support**: Each primitive is evaluated based on the number of supporting edge points within its proximity. Primitives with at least n_{min} supporting points (empirically set to 5-10 points depending on point density) are considered well-supported and retained in the final model.

Redundancy Removal: The consistency analysis identifies redundant elements. The redundancy comprises primitives with insufficient evidential support (unmarked primitives shown as blue lines in Figure 6c), which are typically artifacts from over-connection or false primitive detection. Redundant elements are then systematically removed (Figure 6d) to produce a minimal yet complete representation.

Completeness Enhancement: Regions with dense concentrations of unmarked edge points (visualized as red spheres in Figure 6e) indicate structural elements missing from the initial wireframe. We address these gaps through a sophisticated completion pipeline that culminates in the enhanced structure shown in Figure 6f:

1. **Point Cloud Segmentation**: Unmarked points are grouped into coherent clusters using an adaptive depth-first search that considers both spatial proximity and local curvature characteristics.

2. **Multi-view Primitive Fitting**: Each point cluster is projected onto the three principal planes, and we perform robust primitive fitting in each 2D domain.

3. **3D Primitive Recovery**: The 2D fitting results from all views are consolidated using our hierarchical type reasoning (Circle $\rightarrow$ Arc $\rightarrow$ Line). For each candidate primitive, we compute a consensus score based on fitting quality across views and geometric consistency with adjacent elements.

4. **Structural Integration**: Successfully recovered primitives are integrated into the existing wireframe through vertex merging and topological validation. The integration process ensures watertight connectivity and maintains the manifold properties of the final wireframe.

3. Experiments

3.1. Experimental Setup

3.1.1 Datasets and Implementation Details

We conducted comprehensive experiments across three key components: SAM-based segmentation, 2D sketch analysis, and 3D wireframe reconstruction. For each component, we constructed specialized datasets to ensure optimal performance.

SAM Fine-tuning Dataset: We constructed a specialized dataset derived from the ABC dataset [24] for fine-tuning the Segment Anything Model (SAM). The dataset construction process involved several key steps: First, we processed thousands of 3D CAD models by extracting sharp edges to generate wireframes. These wireframes were then projected onto three orthogonal planes (front, top, and left views) to create standard engineering views. Each projection was stored as parametric primitives (lines, arcs, circles) in structured JSON files. We generated dense 2D point sets through numerical sampling from these parametric equations, with random perturbations added to each point to simulate real-world scanning noise. The point sets were converted to 1024×1024 images through a visualization pipeline that included line dilation, canvas centering, and instance differentiation. This process yielded over 50,000 training images with instance-level annotations.

2D Sketch Analysis Dataset: For 2D sketch analysis, we utilized the dataset provided by Vitruvion [13], containing 200,000 standard 2D CAD sketches for training and 10,000 for testing. Each primitive was sampled with $k = 25$ points based on its parametric equations, with random noise within $[-\sigma, \sigma]$ ($\sigma = 0.02$) added to simulate real-world conditions.

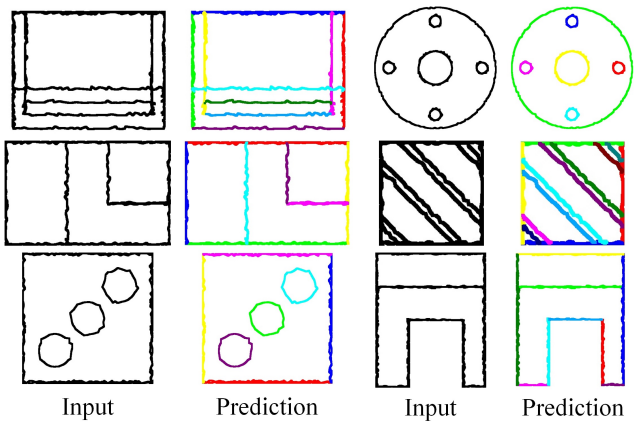

Figure 7. Qualitative results of SAM-based segmentation on CAD sketches

3D Wireframe Reconstruction Dataset: For 3D wireframe reconstruction, we utilized the ABC dataset [24] to create a comprehensive benchmark. We selected 5,000 diverse CAD models covering various mechanical parts and engineering components. From these models, we extracted standard 3D wireframes by computing sharp feature edges and structural curves. The dataset includes complex geometric features such as holes, fillets, chamfers, and intricate surface intersections. Each 3D wireframe serves as ground truth for evaluating the reconstruction accuracy from point clouds. To generate input point clouds, we sampled points from the original CAD surfaces and applied realistic noise patterns simulating laser scanner artifacts. The dataset is divided into 4,000 models for training and 1,000 for testing, ensuring adequate coverage of different geometric complexities.

Implementation Details: All experiments were conducted on a server with RTX 3090 GPU. We employed a transfer learning strategy for SAM fine-tuning, freezing the Image Encoder and Prompt Encoder while training only the Mask Decoder. The model was trained using a combined Binary Cross-Entropy and Dice loss, with prompts generated by randomly sampling up to 16 points from ground-truth primitives. For PC-NET, we trained for 100 epochs using Adam optimizer with learning rate 0.00001, batch size 64, and loss weights $w_{fit} = 10.0$, $w_{con} = 1.0$. For the 3D reconstruction pipeline, we used the fine-tuned SAM for initial segmentation and PC-NET for primitive extraction, followed by our proposed three-view integration algorithm.

3.1.2 Metrics

We employed comprehensive evaluation metrics for each component. For SAM segmentation, we used mean Intersection-over-Union (mIoU) and boundary F-score. For

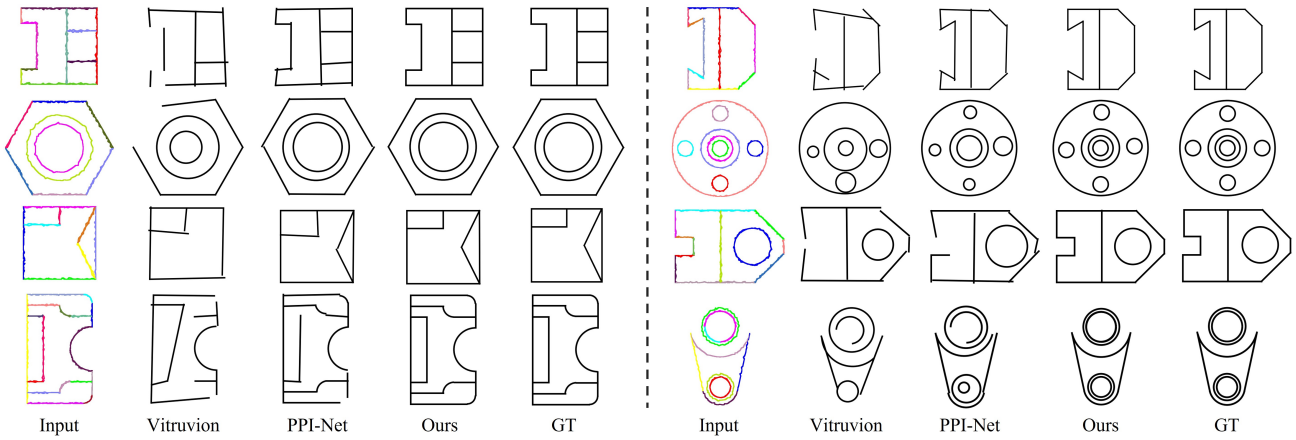

Figure 8. PC-NET is compared with Vitruvion [13] and PPI-Net [5] on 2D CAD sketches.

Table 1. SAM fine-tuning results on CAD sketch segmentation

Method	mIoU (%)	Boundary F-score (%)
Vanilla SAM	46.6	35.5
Fine-tuned SAM	**92.3**	**89.7**

2D sketch analysis, we adopted Primitive Type Accuracy (Acc_{type} ↑) and Chamfer Distance (CD ↓) following PPI-Net [5]. For 3D wireframe reconstruction, we used Chamfer Distance (CD ↓) and Hausdorff Distance (HD ↓) as in NerVE [17].

3.2. Comparison with State-of-the-Art Methods

3.2.1 SAM-based Segmentation Performance

Our fine-tuned SAM model demonstrates remarkable performance on CAD sketch segmentation. As shown in Table 1, the model achieves 92.3% mIoU and 89.7% boundary F-score, significantly outperforming the vanilla SAM model on engineering drawings. The qualitative results in Figure 7 show that our fine-tuned model accurately segments individual geometric primitives even in complex sketch layouts, maintaining clear boundaries between adjacent primitives.

The success of our SAM fine-tuning can be attributed to several factors: First, the specialized dataset provides sufficient and relevant training examples. Second, the partial fine-tuning strategy preserves SAM's powerful visual representations while adapting to the specific characteristics of engineering sketches. Third, the use of ground-truth points as prompts during training effectively teaches the model to associate prompt patterns with complete primitive instances.

3.2.2 2D Sketch Analysis

We compare our PC-NET with Vitruvion [13] and PPI-Net [5] on 2D sketch analysis. Quantitative results in Table 2 show that our method achieves superior performance with 96.78% primitive type accuracy and 0.011 Chamfer distance.

Table 2. We compare our approach with PPI-Net [5] and Vitruvion [13] on the 2D sketches obtained through the projection of the edge point cloud.

	Metrics	
Model	Acc_{type} ↑	CD ↓
Vitruvion [13]	86.51%	0.045
PPI-Net [5]	94.33%	0.037
Ours	**96.78%**	**0.011**

The qualitative comparison in Figure 8 reveals distinct advantages of our approach. Vitruvion's autoregressive generation suffers from error accumulation, as visible in columns 2 and 7, where errors in early steps propagate to affect subsequent primitive extraction. PPI-Net, based on DETR detection framework, shows limitations in handling small primitives and establishing stable primitive relationships, leading to omissions and weak connections (columns 3 and 8).

In contrast, our PC-NET processes 2D point sets through simultaneous primitive fitting and constraint estimation. This integrated approach avoids error propagation while maintaining strong connections between primitives. The results in columns 4 and 9 demonstrate complete primitive extraction with proper geometric relationships.

3.2.3 3D Wireframe Reconstruction

For 3D wireframe reconstruction, we compare with SED-Net [26] and NerVE [17]. As shown in Table 3, our method achieves significantly better metrics (CD: 0.013, HD: 0.023), demonstrating the advantage of our three-view projection approach.

The visual comparison in Figure 9 provides insights into

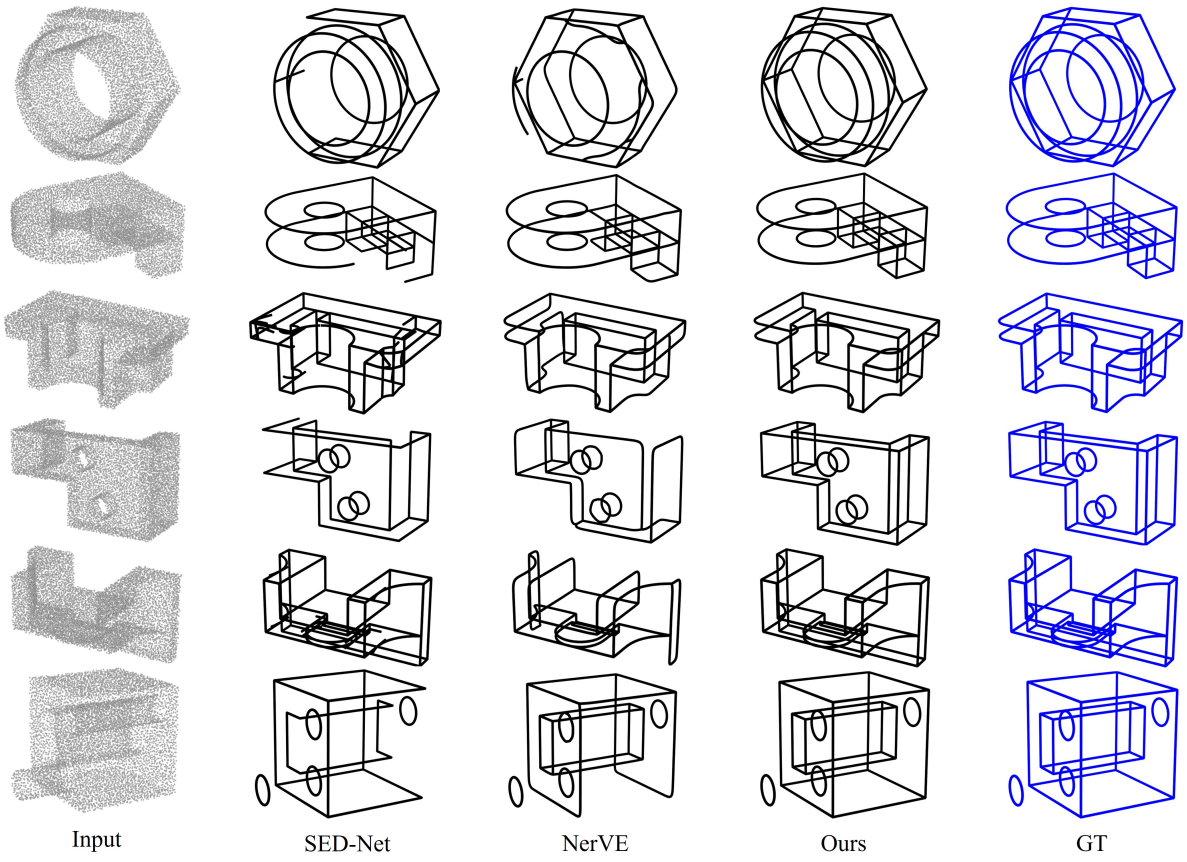

Figure 9. Compare with state-of-art methods to achieve wireframe reconstruction from point clouds.

Table 3. Quantitative comparisons on parametric curve extraction.

	Metrics	
Model	$CD \downarrow$	$HD \downarrow$
SED-Net [26]	0.049	0.157
NerVE [17]	0.024	0.075
Ours	**0.013**	**0.023**

the performance differences. SED-Net (Column 2) suffers from incomplete wireframes due to imperfect instance segmentation. NerVE (Column 3) produces overly smooth transitions and misses sharp features due to its volumetric representation. Our method (Column 4) preserves sharp features and complete structures by leveraging the complementary information from three views and incorporating geometric constraints.

3.3. Ablation Studies

3.3.1 Effect of SAM Fine-tuning

We ablate the importance of SAM fine-tuning by comparing segmentation performance before and after adaptation. As shown in Table 1, the fine-tuned model improves mIoU by 45.7% and boundary F-score by 54.2% over the vanilla SAM. This significant improvement demonstrates that domain adaptation is crucial for handling the unique characteristics of engineering sketches, such as precise boundaries and structured geometric patterns.

Table 4. We compare the impact of the presence or absence of constraint branch on our fitting-based network.

Metrics	$Acc_{type} \uparrow$	$CD \downarrow$
$loss_{fit}$	96.23%	0.017
$loss_{fit} + loss_{con}$	**96.78%**	**0.015**

3.3.2 Effect of Constraint Branch

The constraint branch in PC-NET plays a vital role in maintaining geometric relationships. Table 4 shows that incorporating constraints improves both primitive type accuracy (96.23% to 96.78%) and Chamfer distance (0.017 to 0.015). The additional ablation in Table 5 demonstrates that applying constraints to fitted primitives further improves Chamfer distance to 0.011, confirming that constraints enhance parametric accuracy without affecting type classification.

Table 5. We compared the impact on the metrics before and after applying constraint relationships to the primitives.

Metrics	$Acc_{type} \uparrow$	$CD \downarrow$
Without constraint	96.78%	0.015
With constraint	**96.78%**	**0.011**

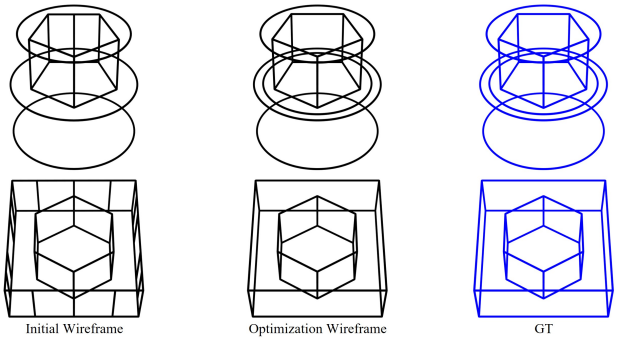

Figure 10. The qualitative analysis of the impact of wireframe optimization algorithms on the reconstruction of wireframe quality.

Table 6. Quantitative analysis of wireframe reconstruction results using optimization algorithms.

Metrics	$CD \downarrow$	$HD \downarrow$
Before Optimization	0.039	0.082
After Optimization	**0.013**	**0.023**

Table 7. Quantitative comparison under different point cloud noise conditions: $X = \{0.05, 0.01, 0.015\}$.

	Metrics	
Noise	$CD \downarrow$	$HD \downarrow$
X=0.015	0.033	0.057
X=0.010	0.025	0.035
X=0.005	0.017	0.028

3.3.3 Effect of Wireframe Optimization

Our wireframe optimization algorithm significantly refines the reconstruction results. Table 6 shows improvements from CD: 0.039 to 0.013 and HD: 0.082 to 0.023 after optimization. Figure 10 visually demonstrates how the optimization removes redundant elements while completing missing structures, resulting in cleaner and more accurate wireframes.

3.4. Robustness Analysis

We evaluate robustness under challenging conditions including noise and sparsity. Table 7 and Figure 11 show that our method maintains reasonable accuracy (CD: 0.033, HD: 0.057) even with significant noise ($X = 0.015$). Under point sparsity (Table 8, Figure 12), with only 50% of edge points, our method still achieves CD of 0.031 and HD of 0.047, demonstrating strong robustness.

The robustness stems from multiple factors: the noise-augmented training data prepares the model for real-world conditions; the three-view approach provides redundant information that compensates for missing data; and the geometric constraints help maintain structural integrity even with incomplete inputs.

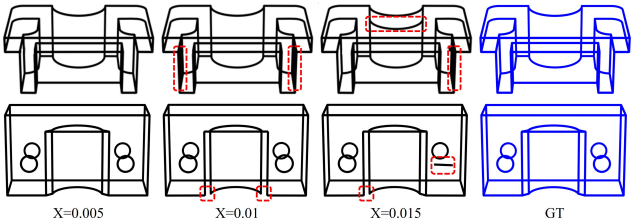

Figure 11. Qualitative analysis results under different point cloud noise conditions: $X = \{0.05, 0.01, 0.015\}$.

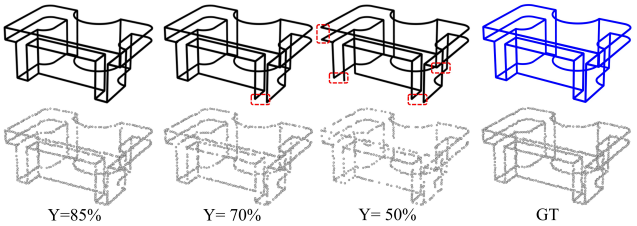

Figure 12. Qualitative analysis results under different edge points sparse conditions: $Y = \{85\%, 70\%, 50\%\}$.

Table 8. Quantitative analysis results under different edge points sparse conditions: $Y = \{85\%, 70\%, 50\%\}$.

	Metrics	
Sparse	$CD \downarrow$	$HD \downarrow$
Y=85%	**0.021**	**0.025**
Y=70%	0.027	0.031
Y=50%	0.031	0.047

3.5. Computational Efficiency

Our complete pipeline demonstrates practical efficiency for real-world applications. The SAM-based segmentation processes 1024×1024 images in near real-time (0.5s per image). PC-NET processes segmented point sets efficiently (0.1s per view), and the three-view integration completes within 2-3 seconds for typical CAD models. This efficiency, combined with the accuracy improvements, makes our approach suitable for industrial applications in reverse engineering and digital manufacturing.

4. Conclusion

In this paper, we have presented a novel framework for high-quality 3D CAD wireframe recovery from point clouds. Our approach introduces several key innovations: a three-view projection paradigm that transforms complex 3D reconstruction into manageable 2D analysis tasks; PC-NET, an end-to-end network that simultaneously performs parametric primitive fitting and geometric constraint estimation;

and an effective strategy for adapting foundation models to engineering domains through targeted fine-tuning. This integrated framework enables precise reconstruction of CAD wireframes that faithfully represent real-world objects without manual intervention.

Extensive experimental validation demonstrates our method's superiority over state-of-the-art approaches in both 2D sketch analysis and 3D wireframe reconstruction. The method maintains robust performance under challenging conditions including noise and point sparsity, while achieving computational efficiency suitable for practical industrial applications in reverse engineering and digital manufacturing.

Limitations and Future Work. While our method shows compelling results, several limitations warrant future investigation. The current implementation focuses on five common constraint relationships; extending this to include more complex constraints such as tangent and concentric relationships would enhance applicability. Additionally, though circles account for most closed segments in typical CAD datasets, supporting other primitives like ellipses and splines would improve generality. Future work will also explore integrating wireframe reconstruction with surface modeling for complete B-Rep generation, and extending the framework to handle dynamic scenes for manufacturing process monitoring.

References

[1] S. Alaniz, M. Mancini, A. Dutta, D. Marcos, and Z. Akata, "Abstracting sketches through simple primitives," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022, pp. 533–549. 4

[2] W. Zhao, Y. Yan, C. Yang, J. Ye, X. Yang, and K. Huang, "Divide and conquer: 3d point cloud instance segmentation with point-wise binarization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 562–571.

[3] T. D. Ngo, B.-S. Hua, and K. Nguyen, "ISBNet: a 3D Point Cloud Instance Segmentation Network with Instance-aware Sampling and Box-aware Dynamic Convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13550–13559.

[4] F. Long, T. Yao, Z. Qiu, L. Li, and T. Mei, "PointClustering: Unsupervised Point Cloud Pre-Training Using Transformation Invariance in Clustering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21824–21834.

[5] L. Wang and X. Wang, "PPI-NET: End-to-End Parametric Primitive Inference," in *Computer Graphics International Conference*, Springer, 2023, pp. 67–78. 2, 4, 9

[6] X. Wang, Y. Xu, K. Xu, A. Tagliasacchi, B. Zhou, A. Mahdavi-Amiri, and H. Zhang, "Pie-net: Parametric inference of point cloud edges," *Advances in Neural Information Processing Systems*, vol. 33, pp. 20167–20178, 2020. 1, 2, 4

[7] B. Pang, H. Xia, and C. Lu, "Unsupervised 3D Point Cloud Representation Learning by Triangle Constrained Contrast for Autonomous Driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5229–5239.

[8] Y. Zeng, C. Jiang, J. Mao, J. Han, C. Ye, Q. Huang, D.-Y. Yeung, Z. Yang, X. Liang, and H. Xu, "CLIP2: Contrastive Language-Image-Point Pretraining from Real-World Point Cloud Data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 15244–15253.

[9] A. Chen, K. Zhang, R. Zhang, Z. Wang, Y. Lu, Y. Guo, and S. Zhang, "Pimae: Point cloud and image interactive masked autoencoders for 3D object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5291–5301.

[10] A. Sain, A. K. Bhunia, P. N. Chowdhury, S. Koley, T. Xiang, and Y.-Z. Song, "CLIP for All Things Zero-Shot Sketch-Based Image Retrieval, Fine-Grained or Not," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 2765–2775. 4

[11] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., "Learning Transferable Visual Models from Natural Language Supervision," in *International Conference on Machine Learning*, PMLR, 2021, pp. 8748–8763. 4

[12] A. Seff, Y. Ovadia, W. Zhou, and R. P. Adams, "Sketchgraphs: A Large-Scale Dataset for Modeling Relational Geometry in Computer-Aided Design," *arXiv preprint arXiv:2007.08506*, 2020. 3

[13] A. Seff, W. Zhou, N. Richardson, and R. P. Adams, "Vitruvion: A Generative Model of Parametric CAD Sketches," *arXiv preprint arXiv:2109.14124*, 2021. 2, 3, 8, 9

[14] R. Wu, C. Xiao, and C. Zheng, "DeepCAD: A Deep Generative Network for Computer-Aided Design Models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6772–6782. 4

[15] Y. Yang and H. Pan, "Discovering Design Concepts for CAD Sketches," in *Advances in Neural Information Processing Systems*, vol. 35, pp. 28803–28814, 2022. 4

[16] A. Matveev, R. Rakhimov, A. Artemov, G. Bobrovskikh, V. Egiazarian, E. Bogomolov, D. Panozzo, D. Zorin, and E. Burnaev, "DEF: Deep Estimation of Sharp Geometric Features in 3D Shapes," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, 2022. 2

[17] X. Zhu, D. Du, W. Chen, Z. Zhao, Y. Nie, and X. Han, "NerVE: Neural Volumetric Edges for Parametric Curve Extraction from Point Cloud," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13601–13610. 2, 9, 10

[18] Y. Ye, R. Yi, Z. Gao, C. Zhu, Z. Cai, and K. Xu, "NEF: Neural Edge Fields for 3D Parametric Curve Reconstruction from Multi-view Images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 8486–8495. 2, 3

[19] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021. 2

[20] Y. Liu, S. D'Aronco, K. Schindler, and J. D. Wegner, "PC2WF: 3D Wireframe Reconstruction from Raw Point Clouds," *arXiv preprint arXiv:2103.02766*, 2021. 2

[21] F. Bernardini, C. L. Bajaj, J. Chen, and D. R. Schikore, "Automatic Reconstruction of 3D CAD Models from Digital Scans," *International Journal of Computational Geometry & Applications*, vol. 9, no. 04n05, pp. 327–369, 1999. 3

[22] P. Li, J. Guo, X. Zhang, and D.-M. Yan, "SECAD-Net: Self-Supervised CAD Reconstruction by Learning Sketch-Extrude Operations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16816–16826. 3

[23] H. Guo, S. Liu, H. Pan, Y. Liu, X. Tong, and B. Guo, "ComplexGen: CAD Reconstruction by B-Rep Chain Complex Generation," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–18, 2022. 3

[24] S. Koch, A. Matveev, Z. Jiang, F. Williams, A. Artemov, E. Burnaev, M. Alexa, D. Zorin, and D. Panozzo, "ABC: A Big CAD Model Dataset for Geometric Deep Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9601–9611. 8

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," in *European Conference on Computer Vision*, Springer, 2020, pp. 213–229. 2

[26] Y. Li, S. Liu, X. Yang, J. Guo, J. Guo, and Y. Guo, "Surface and Edge Detection for Primitive Fitting of Point Clouds," in *ACM SIGGRAPH 2023 Conference Proceedings*, 2023, pp. 1–10. 1, 4, 9, 10

[27] L. Cao, Y. Xu, J. Guo, and X. Liu, "WireframeNet: A Novel Method for Wireframe Generation from Point Cloud," *Computers & Graphics*, vol. 115, pp. 226–235, 2023. 1, 4

[28] X. Wang, L. Wang, H. Wu, G. Xiao, and K. Xu, "Parametric Primitive Analysis of CAD Sketches with Vision Transformer," *IEEE Transactions on Industrial Informatics*, vol. 20, no. 10, pp. 12041–12050, 2024. 2

[29] S. Yan, Z. Yang, C. Ma, H. Huang, E. Vouga, and Q. Huang, "HPNet: Deep Primitive Segmentation Using Hybrid Representations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2753–2762. 1

[30] G. Sharma, D. Liu, S. Maji, E. Kalogerakis, S. Chaudhuri, and R. Měch, "Parsenet: A Parametric Surface Fitting Network for 3D Point Clouds," in *Computer Vision–ECCV 2020*, Springer, 2020, pp. 261–276. 1

[31] Y. Liu, A. Obukhov, J. D. Wegner, and K. Schindler, "Point2CAD: Reverse Engineering CAD Models from 3D Point Clouds," *arXiv preprint arXiv:2312.04962*, 2023. 1

[32] Y. Ganin, S. Bartunov, Y. Li, E. Keller, and S. Saliceti, "Computer-Aided Design as Language," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 5885–5897, 2021. 4

[33] W. Para, S. Bhat, P. Guerrero, T. Kelly, N. Mitra, L. J. Guibas, and P. Wonka, "SketchGen: Generating Constrained CAD Sketches," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 5077–5088, 2021. 4

[34] M. A. Uy, Y.-Y. Chang, M. Sung, P. Goel, J. G. Lambourne, T. Birdal, and L. J. Guibas, "Point2Cyl: Reverse Engineering 3D Objects from Point Clouds to Extrusion Cylinders," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11850–11860. 3

[35] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment Anything,"

in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026. 2, 4

[36] F. Yu, Z. Chen, M. Li, A. Sanghi, H. Shayani, A. Mahdavi-Amiri, and H. Zhang, "CAPRI-Net: Learning Compact CAD Shapes with Adaptive Primitive Assembly," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11768–11778. 3

[37] Z. Chen, A. Tagliasacchi, and H. Zhang, "BSP-Net: Generating Compact Meshes via Binary Space Partitioning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 45–54. 3

[38] G. Sharma, R. Goyal, D. Liu, E. Kalogerakis, and S. Maji, "CSGNet: Neural Shape Parser for Constructive Solid Geometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5515–5523.

[39] C. Li, H. Pan, A. Bousseau, and N. J. Mitra, "Free2CAD: Parsing Freehand Drawings into CAD Commands," *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–16, 2022. 3

[40] J. G. Lambourne, K. Willis, P. K. Jayaraman, L. Zhang, A. Sanghi, and K. R. Malekshan, "Reconstructing Editable Prismatic CAD from Rounded Voxel Models," in *SIGGRAPH Asia 2022 Conference Papers*, 2022, pp. 1–9. 3

[41] K. Chelani, A. Benbihi, T. Sattler, and F. Kahl, "Edge-Gaussians: 3D Edge Mapping via Gaussian Splatting," *arXiv preprint arXiv:2409.12886*, 2024. 2, 3

[42] L. Li, S. Peng, Z. Yu, S. Liu, R. Pautrat, X. Yin, and M. Pollefeys, "3D Neural Edge Reconstruction," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 3

[43] C. Choi, J. Lee, J. Park, and Y. M. Kim, "3Doodle: Compact Abstraction of Objects with 3D Strokes," *ACM Transactions on Graphics*, vol. 43, no. 4, 2024. 4

[44] Y. Zhang, L. Wang, C. Zou, T. Wu, and R. Ma, "Diff3DS: Generating View-Consistent 3D Sketch via Differentiable Curve Rendering," *arXiv preprint arXiv:2405.15305*, 2024. 4

[45] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian Splatting for Real-Time Radiance Field Rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, 2023. 3

[46] T.-M. Li, M. Lukáč, M. Gharbi, and J. Ragan-Kelley, "Differentiable Vector Graphics Rasterization for Editing and Learning," *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, vol. 39, no. 6, 2020. 4