2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

JLGS-CAD: CAD Reconstruction Based on Joint Learning for Geometry and Sequence

Tianzhou Han[1], Fazhi He[2] (✉)

[1] School of Computer Science, Wuhan University, Wuhan, China
fzhe@whu.edu.cn

Abstract. Achieving both high accuracy and greater similarity to the modeling process of human engineers in CAD reconstruction tasks is a challenging problem. In this paper, we propose JLGS-CAD, a neural network designed based on joint learning, aiming to coordinatively maximize the accuracy of model reconstruction and the quality of the modeling sequence. Based on the characteristics of the CAD modeling process, we divide the reconstruction task between two models: the Extrusion Model, responsible for the geometric accuracy of the reconstructed shape, and the Sketch Model, responsible for the quality of the modeling sequence. We adopt a hybrid supervision approach to enable joint learning of both sequences and geometry in the two models. This method significantly improves the quality of the modeling sequence while maintaining the precision of the reconstructed geometry, allowing the network to produce results more aligned with human modeling workflows. Our training pipeline consists of two stages: a supervised pre-training stage on a large-scale dataset with sequence annotations and a self-supervised fine-tuning stage on a target dataset without sequence labels. This reduces the network's dependency on large annotated CAD modeling datasets. Experiments conducted on the ABC and Fusion 360 datasets demonstrate the effectiveness of our method. JLGS-CAD accurately recovers geometric details and constructs editable and creative modeling workflows, showing clear advantages over state-of-the-art alternatives.

Keywords: CAD Reconstruction, Sequence Generation, Joint Learning.

1 Introduction

Serving as a cornerstone in today's manufacturing processes, Computer-Aided Design (CAD) modeling facilitates the creation of accurate and editable representations of physical items [2]. However, constructing CAD models manually is both time-consuming and skill-intensive, often requiring expert knowledge of design tools and the underlying modeling logic [25]. as 3D scanning technologies become increasingly accessible, the demand for automatic and accurate CAD reconstruction from 3D point clouds has grown significantly.

Automated 3D CAD reconstruction developed from an early time [7]. Traditional approaches to CAD reconstruction typically rely on multi-stage pipelines that convert raw 3D data into polygonal meshes, followed by segmentation, primitive extraction,

and procedural modeling [3, 23]. As deep learning techniques have advanced and found extensive application across various domains, several researchers have suggested employing neural networks to directly predict geometric primitives [9] or Constructive Solid Geometry (CSG) trees [26, 38], and B-Rep CAD models [6, 13]. Recent research has begun to focus on feature-based CAD model reconstruction. A representative class of methods aims to predict sketch-extrusion pairs to simulate the human modeling process [14, 15, 16, 30]. Following the publication of various large-scale CAD datasets [12, 24, 32], some approaches have started to explore CAD reconstruction from a sequence generation perspective. While these methods are promising, they often face a trade-off between geometric fidelity and sequence interpretability, especially when constrained by the availability of sequence-labeled datasets.

To address these challenges, we introduce JLGS-CAD, a novel deep neural network framework with joint learning architecture for geometry and sequence, reconstructing accurate and human-like CAD modeling sequences from point clouds. JLGS-CAD uses a dual-model design: a geometry-focused Extrusion Model predicts extrusion boxes, boolean operations, and sketch occupancy, while a sequence-focused Sketch Model generates sketch drawing commands based on the occupancy map. We treat sequence prediction and geometric reconstruction as two distinct yet complementary objectives and optimize them jointly within a unified neural network. Inspired by multi-task learning methods [1, 29, 35], we design a hybrid supervision framework that fosters an intrinsic connection between geometric and sequential information, enabling outputs that are both accurate and human-like. Moreover, we establish a two-stage training procedure to fully leverage available data and ensure robust generalization. This approach significantly reduces reliance on labeled data while maintaining high accuracy and sequence quality. By explicitly modeling both the geometric fidelity and human-like design logic, JLGS-CAD not only produces high-quality CAD models but also generates modeling sequences that are editable, reusable, and aligned with real-world design workflows. Experiments conducted on popular datasets show that our approach surpasses current baseline methods regarding both the accuracy of reconstruction and the quality of command sequences.

Overall, the principal contributions of this study are as follows:

- We propose a novel neural network designed for CAD reconstruction, JLGS-CAD, which focuses on the optimization of two key goals: geometric accuracy and sequence quality.
- We design a hybrid supervision framework based on the idea of joint learning that simultaneously optimizes geometric reconstruction and sequence prediction. This approach enables the network to establish an intrinsic connection between geometric and sequential information, allowing it to produce command sequences that achieve accurate reconstruction while closely reflecting human modeling style.
- We implement a two-phase training procedure: initially, the model undergoes pretraining on a sequence-labeled dataset, followed by fine-tuning on the target dataset. This approach endows JLGS-CAD with strong generalization ability and low dependency on labeled datasets.

- Comprehensive experiments conducted on widely adopted datasets validate that JLGS-CAD produces more precise and human-like modeling command sequences, surpassing current state-of-the-art techniques for same task.

2 Related Work

In recent years, deep learning approaches have been progressively utilized to automatically reconstruct CAD models from raw data. The main approaches can be categorized into three types:

Primitive-Based Reconstruction. Primitive-based reconstruction methods approximate and represent complex CAD models by extracting and combining simple geometric shapes (e.g., planes, cylinders, spheres). Technical fields include parametric surface fitting [27], curve and corner detection [9, 17]. Another related area is reconstruction based on Constructive Solid Geometry (CSG), which combines basic primitives to create complex shapes using Boolean operations [10]. For instance, CSGNet leverages neural networks to parse sequences of Boolean modeling operations [26]. More recent improvements include enhancing representation capability through a three-layer reformulation [21] and introducing more complex geometric primitives (e.g., quadric surfaces) to support richer shape representations [36]. Despite the strong expressive power of primitive-based methods, they tend to lack flexibility and are less editable.

Sketching-Extrusion Based Reconstruction. Sketching and extrusion are commonly used operations in modern CAD modeling workflows, and many studies aim to generate CAD models consistent with this workflow from point clouds [15, 16, 22, 30]. Sketch-and-extrusion methods simulate the real CAD modeling process by predicting 2D sketches and extruding them into 3D models. For example, ExtrudeNet generates 3D shapes by predicting the parameters of closed Bézier curves and performing extrusion operations [22]. SECAD-Net further improves this approach by directly predicting 2D implicit fields using neural networks and training the model through self-supervised learning [16]. The advantage of these methods lies in their alignment with CAD modeling logic, resulting in models with high geometric accuracy. However, since they only support simple sketch and extrusion operations, their representational capability is limited when handling complex shapes.

Sequence Based Reconstruction. Sequence-based reconstruction methods focus on learning the sequential generation process of CAD modeling operations to enable more flexible and editable CAD model generation [8]. DeepCAD is one of the representative studies in this area, generating complete models by learning CAD modeling operation sequences [33]. Similarly, methods like Fusion360 generate editable modeling instruction sequences from point clouds or sketch inputs [32]. SkexGen further expanded on this idea by supporting more complex modeling operations [34]. Recent studies have looked into applying language models for CAD reconstruction. For instance, combining pre-trained language models with diffusion models has been proposed for sequence-based modeling tasks [18]. These methods closely align with real CAD modeling workflows, producing results that are easy to modify. Nevertheless, these approaches continue to encounter difficulties regarding geometric precision and the reconstruction of

intricate shapes. Additionally, their performance heavily relies on datasets with sequence annotations. Building on these advances, our work addresses the trade-off between geometric fidelity and sequence availability, while reducing reliance on large-scale annotated datasets.

3 Problem Statement and Overview

3.1 Sketch-and-Extrude Construction Sequence

By referring to previous works [16, 33, 34] and common dataset formats [12, 32], we represent a CAD model using sketches and extrusions. We define a primitive as the basic shape formed by a sketch along with an extrusion operation. Consequently, the CAD model reconstruction process can be described as follows: starting with an input 3D shape—point cloud in this work—the sketch and extrusion information will be predicted for the n primitives that compose the shape.

3.2 Task Decomposition

For extrusion operations, the key sketch factor affecting reconstruction accuracy is the closed region it encloses. To improve both reconstruction accuracy and sequence quality, we divide sketch prediction into two steps: first, predicting the sketch's occupancy representation to enable precise geometric reconstruction with extrusion information; second, inferring the command sequence forming the sketch based on the occupancy representation. Accordingly, JLGS-CAD comprises two sub-models: the Extrusion Model and the Sketch Model, shown in **Fig. 1** and **Fig. 3**. Extrusion Model receives a point cloud as input and predicts a set of primitives, each defined by its bounding box, Boolean operations, and sketch occupancy representation. The occupancy representation is then input to the Sketch Model, which generates the corresponding sketch command sequence. Combining the sketch sequence with the extrusion information yields the final command sequence representing the full model. Assuming perfect Sketch Model predictions, the Extrusion Model determines the geometric accuracy, while the Sketch Model controls the sequence quality.

3.3 Joint Learning and Hybrid Supervision

We frame the improvement of geometric accuracy and human modeling as a multi-task learning challenge. Inspired by [1, 29], we design the optimization objectives in both the Extrusion Model and the Sketch Model to jointly learn sequence and geometric information using a hybrid supervision framework. In the Extrusion Model, primitive box prediction is supervised by the extrusion command parameters from the dataset, while other supervision signals are derived from geometric comparisons, requiring no additional annotations. The Sketch Model follows the same approach: the sketch command sequence supervises network sequence prediction, while contours and distance fields guide geometric learning. In both models, sequence and geometric learning act

as auxiliary tasks for each other. Geometric learning refines sequence parameter prediction, while sequence learning stabilizes geometric prediction initialization. This multi-task framework enables a deeper understanding of the relationship between sequence learning and geometry.

Given the distinct characteristics of these two objectives, the training process is organized into a pre-training phase followed by a fine-tuning phase. During pre-training, the network undergoes hybrid supervised learning on a sequence-annotated dataset to build general sequence and geometric reasoning capabilities. Next, in the target dataset (evaluation dataset), the network will perform self-supervised fine-tuning based only on geometric information, which will further improve the geometric accuracy of the network prediction with almost no loss in sequence reasoning capabilities.

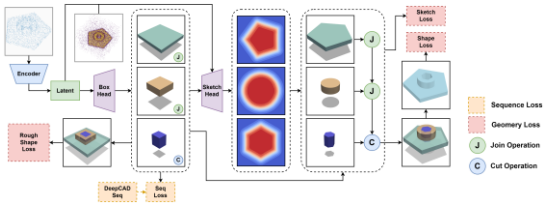

Fig. 1. Network architecture for Extrude Model. The input point cloud is encoded into latent first, then Box Head predicts primitive box parameters.For each primitive, the Sketch Head predicts a 2D sketch SDF in the local coordinate frame. Primitives are sequentially merged into a global SDF via Boolean operations, then converted into an occupancy map for supervision.

4 Method

We now present the detailed implementation of our JLGS-CAD framework. JLGS-CAD consists of two main components, which we refer to as the Extrusion Model and the Sketch Model. A target point cloud is first input to the Extrusion Model, which predicts the basic parameters of the primitive components, the signed distance field (SDF) of the sketch, and the sketch's occupancy representation. The occupancy representation is then passed to the Sketch Model, which predicts a sequence of sketch drawing commands. The final CAD reconstruction sequence is obtained by combining the outputs of both models. Both modules are designed with a multi-task learning strategy that jointly optimizes for geometry recovery and sequence prediction, and support hybrid-supervised pretraining and self-supervised fine-tuning.

4.1 Extrusion Model

In JLGS-CAD, the Extrusion Model is crafted to precisely match the shape of the CAD model. Inspired by [16, 19], we adopt an implicit representation using Signed Distance Fields (SDF) to describe the reconstruction result, which allows for resolution-independent, high-fidelity shape modeling while keeping parameter size under control. **Fig. 1** shows the architecture of the Extrusion Model. A set of query points $Q = \{q_i\}_{i=1}^{N}$ is input into the network, which predicts the signed distance S_i from each query point q_i

to the surface of the object. This distance is then converted into an occupancy prediction O_{pred}, which is matched against the ground truth occupancy label O_{gt} to learn the shape representation.

Point Cloud Encoder. Given an input point cloud P, the first stage is a point cloud encoder based on PointNet++ [20]. Through feature abstraction, input P is transformed into a feature vector $\mathbf{z} \in \mathbb{R}^{256}$, later employed for primitive parameter prediction and SDF regression.

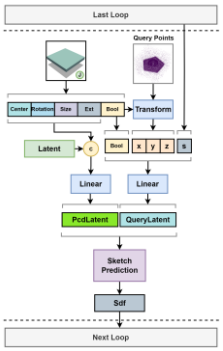

Fig. 2. Architecture for Sketch Head. Sketch Head iteratively predicts the2D SDF sketch of each primitive by combining point cloud features with query points, Boolean operations, and the current global SDF, progressivelyrefining the geometry and updating the overall shape.

Primitive Box Prediction. After obtaining the latentrepresentation $\mathbf{z}$, we first decode it into a set of primitive boxparameters, which define the coarse shape and composition of the target model. This is achieved using a decoder wecall the Box Head, which outputs n primitives:

$$\text{BoxHead}(z) = \{\mathbf{B}_i\}_{i=1}^{n}, \quad \mathbf{B}_i = (c_i, r_i, s_i, e_i, b_i) \tag{1}$$

where $c_i = (x, y, z)$ is the reference point of the box, also the origin of the sketch plane, $r_i = (\theta, \phi, \gamma)$ is the rotation angles, $s_i = (l, w)$ is the estimated sketch size (length and width), $e_i = (e_{\text{min}}, e_{\text{max}})$ is the extrusion height range, $b_i \in \mathbb{R}^3$ is the Boolean operation type (join, cut, intersect), following DeepCAD and Fusion360.

Sketch SDF Prediction. Once the box parameters are predicted, the next step is to reconstruct the fine geometry of each primitive by predicting its corresponding sketch.We refer to this component as Sketch Head. Inspired by[16, 30], we use an implicit representation via 2D SDF to describe each sketch. Query points are first transformed intothe local coordinate system defined by each box. Then their signed distances are predicted and projected onto the sketch plane to obtain 2D sketch SDFs. Given that primitives are added sequentially via Boolean operations and the orderaffects the resulting geometry, the Sketch Head is designed as an iterative module. It predicts sketches in sequence and updates the global SDF after each prediction to minimize interference from subsequent primitives.

For the i-th primitive, the latent point cloud z and its box parameters B_i are passed through MLPs to produce the latent code L_{Pcd}. Simultaneously, transformed query points Q_t in the sketch coordinate system are processed along with the Boolean operation b_i and the current global SDF S_{last} to produce the latent query features L_{Query}. These are concatenated and passed to the sketch prediction module to produce the 2D SDF of the sketch:

$$S_i^{2D} = f_{\text{sketch}}(\text{MLP}_{pcd}([z, B_i]), \text{MLP}_{query}([Q_t, b_i, S_{\text{last}}])) \tag{2}$$

The resulting 2D sketch SDF is lifted into 3D and combined with the global SDF using Boolean operations.

SDF and Occupancy Representation. We use SDF to represent both the entire shape and each primitive. Each new primitive's SDF S_i is merged into the global SDF S_{global} using the following Boolean operations:

$$\begin{aligned} \text{Join}(S_1, S_2) &= \min(S_1, S_2) \\ \text{Cut}(S_1, S_2) &= \max(S_1, -S_2) \\ \text{Intersect}(S_1, S_2) &= \max(S_1, S_2) \end{aligned} \tag{3}$$

$$O = \text{Sigmoid}(-\eta S) \tag{4}$$

In the specific implementation, we use softmax to avoid gradient vanishing [21]. Through Boolean operations, we gradually merge primitives to obtain the final signed distance field of the entire model. Then we convert the signed distance field into an occupancy representation, where points inside the shape are denoted as 1 and points outside are marked as 0. Referring to [16], the Sigmoid function is used to perform operations that enable differentiable transformations.

4.2 Sketch Model

Statistics of sequence annotation information in the DeepCAD dataset show that more than 75% of the commands in the sequence that constitutes the CAD model are sketch-related commands. Therefore, improving the quality of the sketch command sequence is essential for elevating the overall sequence quality produced by the model. Many previous works on reconstructing CAD models from point clouds can output occupancy representations or similar representations of sketches [15, 16, 30], but they use heuristic methods or contour fitting to construct sketch sequences, which results in inaccurate restoration of details and reduced editability. In JLGS-CAD, we design the Sketch Model to be a high-quality imitation of the drawing process of human engineers. The network architecture of the Sketch Model is depicted in **Fig. 3**. The network receives the occupancy representation of a sketch as input and predicts a sequence of sketch commands to precisely capture the sketching process. The following are the modules of the Sketch Model:

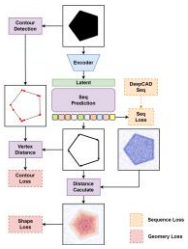

Fig. 3. The Sketch Model takes a sketch's occupancy representationas input and predicts a sequence of sketch commands that accurately imitates the human drawing process. It uses a convolutional encoder and anautoregressive decoder for sequence prediction, with auxiliary supervision from contour fitting and distance field loss to enhance geometric accuracy.

Sequence Prediction. When the occupancy representation of the sketch is input into the Sketch Model, we use a 2D convolutional image encoder with multi-scale feature fusion to transform the occupancy data into latent L, then predict the drawing sequence command by command through a transformer decoder with an autoregressive structure. Referring to the sequence format of [33], when predicting the parameters of a line or arc, we predict only their end points and make the end point of the final curve in the loop to serve as the start of the first, so that our curve sequence can always be closed.

Contour Fitting. In our Sketch Model, we use contour fitting as an auxiliary supervisory information to improve the contour accuracy. In the data processing stage, the Teh-Chin chain approximation [37] is employed to derive the sketch's contour. Then we minimize the distance from all vertices in the contour to each curve in the sequence prediction to make the prediction result fit the contour more closely.

Distance Field. To improve the geometric accuracy of sketch reconstruction, the predicted query point signed distance field from the Extrusion Model is utilized as supervision. We compute the shortest distance from the query point P to the closed curve enclosed by the curve corresponding to the command, then reduce the discrepancy between the predicted field and the actual field. The detailed formula is outlined below:

$$D(P, Line) = \begin{cases} \| P - A \|, & t < 0 \\ \| P - B \|, & t > 1 \\ \| P - (A + \frac{(AP \cdot AB)}{\|AB\|^2} AB) \|, & otherwise \end{cases}, t = \frac{AP \cdot AB}{|AB|^2} \tag{5}$$

where A,B are two endpoints of Line.

$$D(P, Arc) = \begin{cases} |\, \| P - O \| - r|, & \theta_1 \le \theta_P \le \theta_2 \\ min(\| P - A \|, \| P - B \|), & otherwise \end{cases} \tag{6}$$

where A,B are two endpoints of Arc, O is the center, r is radius, θ_1, θ_P, θ_2 are angles of start, P and end.

$$D(P, Circle) = |\, \| P - O \| - r| \tag{7}$$

with O representing the center of the Circle, r is radius.

$$D(P, C) = min\{d|d = D(p, c), c \in C\} \tag{8}$$

where C is the set of curves.

Following [22], we can convert the distance field into a signed distance field by determining whether the query points lie inside the shape. However, this approach introduces more complex gradient variations. Since our sequences are always closed, our primary focus is on the accuracy of the shape's boundary. Therefore, we use an unsigned distance field for the predicted sequence results and minimize the absolute difference between this field and the signed distance field of the query points.

4.3 Training

The Extrusion Model as well as the Sketch Model are trained independently. The training strategies of the two models are similar. Both are pre-trained with hybrid supervision based on a sequence-annotated dataset and fine-tuned with self-supervision on the evaluation dataset.

Extrusion Model training. As shown in **Fig. 1**, we designed four supervisory signals for the Extrusion Model. First, we calculate the gap between the prediction result of the primitive box and the sequence annotation in the dataset. We define this loss as:

$$L_{box}^{seq} = CrossEntropy(B_{pred}, B_{gt}) \tag{9}$$

Next, we introduce two reconstruction losses aimed at reducing the geometric discrepancy between the network output and the ground truth. To establish the correspondence between the model and the supervised information of the sequence, we utilize the box parameters to compute the occupancy representation of the coarse model assembled from cubes, subsequently comparing it with the ground truth. We compute the mean squared error (MSE) as the box shape loss:

$$L_{box}^{shape} = MSE(O_{pred}^{box}, O_{gt}) \tag{10}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2 \tag{11}$$

For the final result of the network prediction, we also calculate MSE loss between the predicted value of occupancy, which comes from the structure after the sketch 2D signed distance field is stretched and merged, and the ground truth:

$$L_{shape} = MSE(O_{pred}, O_{gt}) \tag{12}$$

Inspired by [16], we introduce a 2D sketch loss to promote learning accurate sketch contours. For each primitive i, we project the query points within its height range onto the plane of sketch, followed by a comparison between the predicted and ground-truth 2D occupancy maps:

$$L_{sketch}^{i} = MSE(O_{pred}^{i}, O_{gt,proj}^{i}) \tag{13}$$

To mitigate the influence of later primitives on earlier sketches, we exclude affected query points from the loss computation as **Fig. 4** shows:

$$Q_i = Q - \{q \in Q | O_j(q) > threshhold, j > i\} \tag{14}$$

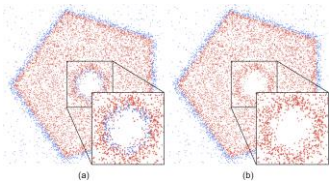

Fig. 4. The filtering ensures that the sketch loss at each step accurately reflects only the contribution of the current primitive. (a)Occupancy before filtering (b)Occupancy after filtering.

Extrusion Model uses all the above losses for calculation during pre-training, while the box sequence loss is excluded during fine-tuning. The total loss is as follows:

$$L_{total} = \lambda_1 L_{box}^{seq} + \lambda_2 L_{box}^{shape} + \lambda_3 L_{shape} + \lambda_4 L_{sketch} \tag{15}$$

In order to adapt the network to the difference in loss between the two training stages, we set the weight λ_1 of the box sequence loss to decrease with the training process during pre-training. The hyperbolic tangent function is employed to achieve this attenuation and activated when the training progress reaches 20%:

$$W(t) = \frac{1-\tanh(\alpha(t-0.5))}{2}, t = \frac{\text{epoch}}{\text{max\_epoch}} \tag{16}$$

Sketch Model training. As shown in **Fig. 3**, we designed three supervisory signals for the Sketch Model. First, we compute the gap between the sketch sequence prediction and the ground truth:

$$L_{sketch}^{seq} = CrossEntropy(C_{pred}, C_{gt}) \tag{17}$$

Next, we calculate the distance between the contour vertices obtained by Teh-Chin chain approximation and the curve connected by the sketch sequence, and use it as the contour loss:

$$L_{contour} = MSE(D_{contour}, 0) \tag{18}$$

To enhance the geometric precision of the Sketch Model further, we calculate a set of query points' distance field of the curve formed the sketch sequence, compare it with the distance field ground truth of the query points, and calculate the shape loss:

$$L_{shape} = MSE(D_{pred}, D_{gt}) \tag{19}$$

Sketch Model will use all the above losses for calculation during pre-training, while omitting the sketch sequence loss in fine-tuning. The total loss function is as follows:

$$L_{total} = \lambda_1 L_{sketch}^{seq} + \lambda_2 L_{contour} + \lambda_3 L_{shape} \tag{20}$$

We set same sequence loss decay strategy in Sketch Model as in Extrusion Model.

5 Experiments

In the following part, we evaluate the efficacy of JLGS-CAD on DeepCAD [12, 33] dataset and Fusion 360 Gallery [32] dataset. By conducting comprehensive comparisons and ablation experiments, we report the efficacy of our method, showing that it outperforms current leading baseline techniques in CAD reconstruction.

5.1 Experimental Setups

Datasets. We train JLGS-CAD using the DeepCAD dataset and evaluate it on both DeepCAD dataset [12, 33] and Fusion 360 Gallery [32]. To unify the annotation formats, we converted the labels from Fusion 360 Gallery to match the sequential format used in DeepCAD. We performed a more comprehensive parsing of the DeepCAD dataset, extracting data like extrusion boxes, SDF, occupancy representations to satisfy the requirements of JLGS-CAD. We generate surface point cloud samples from the mesh models in the datasets as input to the network. Following [4, 5], we randomly sample query points from the vicinity space of the models and compute their occupancy representations. During the self-supervised fine-tuning, we employ the pretrained model released by [19] to generate occupancy fields for the shapes.

Implementation Details. JLGS-CAD is developed with PyTorch and trained on an Nvidia RTX 3090 GPU. For the Extrusion Model, the training process is carried out using the Adam optimizer [11], set with a learning rate of 1×10^{-4}. The model is first pretrained using the training set for 200 epochs with 24 samples per batch. Afterward, the model undergoes fine-tuning using the test set, where shapes are randomly organized into batches of 512 and each batch is further trained over 50 epochs, producing a fine-tuned model for evaluation on that group. Following SECAD-Net[16], the number of predicted primitives is defined as 4. Throughout the training process, we use a mask to block the influence of subsequent primitives, which allows the network to predict all primitive sketches simultaneously without a significant increase in training time. Our approach maintains a parameter scale comparable to SECAD-Net, with about 30% increase in GPU memory usage due to multi-step primitive feature encoding. During inference, the iterative structure leads to approximately a 50% increase in inference time. The Adam optimizer is utilized in the Sketch Model, configured with a learning rate of 1×10^{-4}, pretraining it for 300 epochs with 64 samples per batch. We then fine-tune it in the same manner as the Extrusion Model—grouping 512 shapes per batch and training for 50 epochs per group. During both pretraining and fine-tuning stages of the Sketch Model, we employ data augmentation such as random rotations, scaling of sketches, and injecting noise into the signed distance fields of query points.

Evaluation Metrics. For quantitative evaluation, we adopt commonly used metrics from prior work [33], including Chamfer Distance (CD), Intersection-over-Union (IoU), Invalid Rate (IR), Command Accuracy (ACC_{cmd}) and Parameter Accuracy (ACC_{param}).

Table 1. Quantitative assessment of reconstruction outcomes on DeepCAD dataset. Method with * means data that we cite from [18], for the reconstruction code is not publicly available.

Method	MedCD↓	IoU↑	IR↓ (%)	ACCcmd↑ (%)	ACCparam↑ (%)
DeepCAD	1.036	0.439	13.68	76.49	68.72
Point2cyl	0.694	0.697	5.33	34.15	27.56
SECAD-Net	0.438	0.728	6.62	36.89	28.35
ExtrudeNet*	0.337	0.403	25.34	28.17	24.73
HNC-CAD*	0.864	0.653	5.62	82.69	74.58
CAD-Diffuser*	0.302	0.743	**1.48**	88.55	82.92
Ours	**0.271**	**0.783**	2.35	**89.86**	**84.11**

5.2 Quantitative results

We compare our method with several similar CAD reconstruction approaches capable of outputting editable sequences. The quantitative results for both DeepCAD and Fusion 360 Gallery datasets are respectively presented in **Table 1** and **Table 2**. For DeepCAD [33] and Point2Cyl [30], we retrained and tested their original implementations on the same datasets. For SECAD-Net [16], we adapted it by substituting the voxel encoder with a point cloud encoder, and then retrained and evaluated it. For the remaining methods, we cite the results reported in CAD-Diffuser [18].

Table 2. Quantitative assessment of reconstruction outcomes on Fusion 360 dataset. Method with * means data that we cite from [18], for the reconstruction code is not publicly available.

Method	MedCD↓	IoU↑	IR↓ (%)	ACCcmd↑ (%)	ACCparam↑ (%)
DeepCAD	4.427	0.37	24.14	70.68	62.91
Point2cyl	0.631	0.638	5.89	35.33	27.94
SECAD-Net	0.529	0.704	6.98	34.26	25.89
ExtrudeNet*	0.495	0.373	24.97	27.43	23.75
HNC-CAD*	3.682	0.635	7.27	75.46	64.52
CAD-Diffuser*	0.385	0.632	**1.65**	85.56	**80.48**
Ours	**0.348**	**0.744**	2.94	**85.81**	79.54

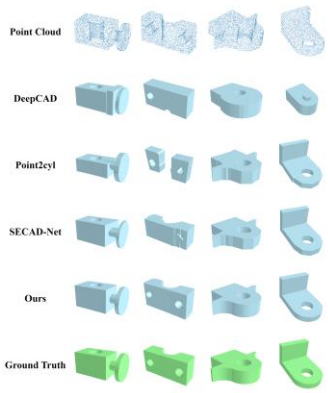

Fig. 5. Visual comparison between reconstruction results.

As shown, our proposed JLGS-CAD outperforms all other methods across all evaluation metrics. In particular, the superior MedCD and IoU values demonstrate our method's high geometric fidelity. The best performance in command and parameter accuracy shows that our model effectively recovers human-like modeling sequences. We also show some visual comparison between reconstruction results in **Fig. 5**.

5.3 Ablation Study

Ablation studies are conducted to evaluate the efficacy of our architectural design within JLGS-CAD. **Table 3** reports the quantitative results on the DeepCAD dataset.

Table 3. Ablation Study results on DeepCAD dataset. E means ExtrusionModel. S means Sketch Model.

Settings	MedCD↓	IoU↑	IR↓ (%)	ACCcmd↑ (%)	ACCparam↑ (%)
Baseline	0.438	0.728	6.62	36.89	28.35
E	0.329	0.769	3.11	43.76	33.62
E+S	0.307	0.773	2.76	87.52	79.73
E Finetune	0.276	0.781	2.49	87.93	82.46
E+S Finetune	0.271	0.783	2.35	89.86	84.11
Baseline	0.438	0.728	6.62	36.89	28.35
E	0.329	0.769	3.11	43.76	33.62

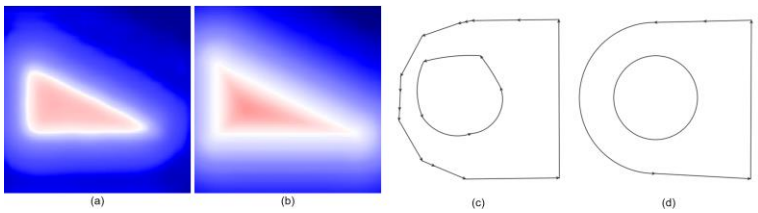

Fig. 6. (a)Sketch SDF before Extrusion Model finetune. (b)Sketch SDF after Extrusion Model finetune. (c)Sketch sequence before Sketch Model finetune. (d)Sketch sequence before Sketch Model finetune.

We take SECAD-Net [16] as the baseline, as our Extrusion Model is built upon its design. First, we evaluate the performance gain from using only the improved Extrusion Model. The results show a significant improvement in geometric reconstruction accuracy, and a modest increase in sequence recovery rate. This demonstrates that incorporating sequence-level supervision enhances the similarity of the predicted primitives to those used in human modeling, providing a strong initialization for geometry reconstruction. With the addition of the Sketch Model, we observe a substantial increase in sequence recovery rate, indicating its effectiveness in predicting sketch sequences. Next, we evaluate the impact of fine-tuning both models individually. Fine-tuning the Extrusion Model further improves geometric accuracy, validating the effectiveness of our self-supervised training strategy in capturing geometric details from input data. Fine-tuning the Sketch Model boosts the sequence recovery rate even further, showing that the Sketch Model, having established an intrinsic link between geometry and sequence during pretraining, can leverage geometric supervision to further refine its sequence predictions. **Fig. 6** visualizes the improvements achieved through finetuning Extrusion Model and Sketch Model.

6 Conclusion

This paper presents JLGS-CAD, a novel deep learning framework developed to reconstruct editable, human-like CAD modeling sequences from point clouds. Unlike prior approaches that treat geometry and sequence prediction separately or simplify modeling operations, JLGS-CAD embraces a joint learning paradigm that simultaneously optimizes geometric reconstruction and sequence generation. By decomposing the task into two synergistic sub-models, an Extrusion Model for geometric accuracy and a Sketch Model for sequence interpretability, we enable the system to align closely with real-world CAD design logic. Furthermore, our hybrid supervision framework leverages both command annotations and geometric signals, facilitating mutual enhancement between geometry and sequence learning. The proposed two-stage training strategy effectively balances labeled and unlabeled data, ensuring scalability and robust generalization across datasets. Experimental results demonstrate that JLGS-CAD not only achieves superior reconstruction accuracy but also generates modeling sequences that are intuitive, reusable, and closely aligned with human design behavior.

Our future research will focus on exploring the combination of the Extrusion and Sketch Models into a cohesive end-to-end network. Meanwhile, we aim to extend the range of command types the network can handle, thereby enhancing the complexity and usability of the reconstructed results. Furthermore, replacing the prediction modules with large language models may unlock new potentials and greater flexibility to the framework.

Acknowledgements. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

References

1. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: A holistic approach to semi-supervised learning. Advances in neural information processing systems 32 (2019)
2. Brière-Côté, A., Rivest, L., Maranzana, R.: Comparing 3d cad models: uses, methods, tools and perspectives. Computer-Aided Design and Applications 9(6), 771– 794 (2012)
3. Buonamici, F., Carfagni, M., Furferi, R., Governi, L., Lapini, A., Volpe, Y.: Reverse engineering modeling methods and tools: a survey. Computer-Aided Design and Applications 15(3), 443–464 (2018)
4. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
5. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (jun 2020)
6. Dupont, E., Cherenkova, K., Kacem, A., Ali, S.A., Arzhannikov, I., Gusev, G., Aouada, D.: Cadops-net: Jointly learning cad operation types and steps from boundary-representations. In: 2022 International Conference on 3D Vision (3DV). pp. 114–123. IEEE (2022)
7. Eggert, D.W., Fitzgibbon, A.W., Fisher, R.B.: Simultaneous registration of multiple range views for use in reverse engineering of cad models. Computer Vision and Image Understanding 69(3), 253–272 (1998)
8. Ganin, Y., Bartunov, S., Li, Y., Keller, E., Saliceti, S.: Computer-aided design as language. Advances in Neural Information Processing Systems 34, 5885–5897 (2021)
9. Guo, H., Liu, S., Pan, H., Liu, Y., Tong, X., Guo, B.: Complexgen: Cad reconstruction by b-rep chain complex generation. ACM Transactions on Graphics (TOG) 41(4), 1–18 (2022)
10. Kania, K., Zieba, M., Kajdanowicz, T.: Ucsg-net unsupervised discovering of constructive solid geometry tree. Advances in neural information processing systems 33, 8776–8786 (2020)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., Panozzo, D.: Abc: A big cad model dataset for geometric deep learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9601– 9611 (2019)
13. Lambourne, J.G., Willis, K.D., Jayaraman, P.K., Sanghi, A., Meltzer, P., Shayani, H.: Brepnet: A topological message passing system for solid models. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12773–12782 (2021)

14. Lambourne, J.G., Willis, K., Jayaraman, P.K., Zhang, L., Sanghi, A., Malekshan, K.R.: Reconstructing editable prismatic cad from rounded voxel models. In: SIGGRAPH Asia 2022 Conference Papers. pp. 1–9 (2022)
15. Li, P., Guo, J., Li, H., Benes, B., Yan, D.M.: Sfmcad: Unsupervised cad reconstruction by learning sketch-based feature modeling operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4671–4680 (2024)
16. Li, P., Guo, J., Zhang, X., Yan, D.M.: Secad-net: Self-supervised cad reconstruction by learning sketch-extrude operations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16816–16826 (2023)
17. Li, Y., Liu, S., Yang, X., Guo, J., Guo, J., Guo, Y.: Surface and edge detection for primitive fitting of point clouds. In: ACM SIGGRAPH 2023 conference proceedings. pp. 1–10 (2023)
18. Ma, W., Chen, S., Lou, Y., Li, X., Zhou, X.: Draw step by step: Reconstructing cad construction sequences from point clouds via multimodal diffusion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 27154–27163 (2024)
19. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4460–4470 (2019)
20. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems 30 (2017)
21. Ren, D., Zheng, J., Cai, J., Li, J., Jiang, H., Cai, Z., Zhang, J., Pan, L., Zhang, M., Zhao, H., et al.: Csg-stump: A learning friendly csg-like representation for interpretable shape parsing. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 12478–12487 (2021)
22. Ren, D., Zheng, J., Cai, J., Li, J., Zhang, J.: Extrudenet: Unsupervised inverse sketch-and-extrude for shape parsing. In: European Conference on Computer Vision. pp. 482–498. Springer (2022)
23. Requicha, A.A., Rossignac, J.R.: Solid modeling and beyond. IEEE computer graphics and applications 12(5), 31–44 (1992)
24. Seff, A., Ovadia, Y., Zhou, W., Adams, R.P.: Sketchgraphs: A large-scale dataset for modeling relational geometry in computer-aided design. arXiv preprint arXiv:2007.08506 (2020)
25. Shapiro, V.: Solid modeling. Handbook of computer aided geometric design 20, 473–518 (2002)
26. Sharma, G., Goyal, R., Liu, D., Kalogerakis, E., Maji, S.: Csgnet: Neural shape parser for constructive solid geometry. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (Jun 2018)
27. Sharma, G., Liu, D., Maji, S., Kalogerakis, E., Chaudhuri, S., Mech, R.: Parsenet: A parametric surface fitting network for 3d point clouds. arXiv: Computer Vision and Pattern Recognition,arXiv: Computer Vision and Pattern Recognition (Mar 2020)
28. Sobh, T.M., Owen, J., Jaynes, C., Dekhil, M., Henderson, T.C.: Industrial inspection and reverse engineering. In: Proceedings of 1994 IEEE 2nd CAD-Based Vision Workshop. pp. 228–235. IEEE (1994)
29. Sun, Y., Chen, Y., Wang, X., Tang, X.: Deep learning face representation by joint identification-verification. Advances in neural information processing systems 27 (2014)
30. Uy, M.A., Chang, Y.Y., Sung, M., Goel, P., Lambourne, J.G., Birdal, T., Guibas, L.J.: Point2cyl: Reverse engineering 3d objects from point clouds to extrusion cylinders. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11850–11860 (2022)

31. Wang, X., Zheng, J., Hu, Y., Zhu, H., Yu, Q., Zhou, Z.: From 2d cad drawings to 3d parametric models: A vision-language approach. arXiv preprint arXiv:2412.11892 (2024)
32. Willis, K.D., Pu, Y., Luo, J., Chu, H., Du, T., Lambourne, J.G., Solar-Lezama, A., Matusik, W.: Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences. ACM Transactions on Graphics (TOG) 40(4), 1–24 (2021)
33. Wu, R., Xiao, C., Zheng, C.: Deepcad: A deep generative network for computer-aided design models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6772–6782 (2021)
34. Xu, X., Willis, K.D., Lambourne, J.G., Cheng, C.Y., Jayaraman, P.K., Furukawa, Y.: Skexgen: Autoregressive generation of cad construction sequences with disentangled codebooks. In: International Conference on Machine Learning. pp. 24698–24724. PMLR (2022)
35. Yin, J., Zhou, J., Shan, Y., Peng, J., Liu, H., Zhou, X., Wang, X.: Multi-task learning for hyper-relational knowledge graph completion. In: International Conference on Intelligent Computing. pp. 115–126. Springer (2024)
36. Yu, F., Chen, Q., Tanveer, M., Mahdavi Amiri, A., Zhang, H.: D2csg: Unsupervised learning of compact csg trees with dual complements and dropouts. Advances in Neural Information Processing Systems 36, 22807–22819 (2023)
37. Teh, C.H., Chin, R.T.: On the detection of dominant points on digital curves. IEEE Transactions on pattern analysis and machine intelligence 11(8), 859–872 (1989)
38. Zhang, S., Guan, Z., Jiang, H., Ning, T., Wang, X., Tan, P.: Brep2seq: a dataset and hierarchical deep learning network for reconstruction and generation of computeraided design models. Journal of Computational Design and Engineering 11(1), 110–134 (2024)