MopCAD: A Multimodal Operation-Oriented CAD Generation Framework with Visual Guidance and Context Control

Anonymous ACL submission

Abstract

Computer-aided design (CAD) modeling is a sequential process that requires precise reasoning over operation parameters and spatial locations. However, existing CAD generation methods mostly adopt a one-shot paradigm and make limited use of visual spatial guidance, leading to simple, imprecise, and less controllable generation. To address these problems, we propose MopCAD, a multimodal framework for controllable natural language-guided CAD modeling. To the best of our knowledge, MopCAD is the first framework to introduce intermediate visual guidance into the CAD modeling process, enabling operation-level localization for step-wise CAD generation. We further propose the first history selector for CAD procedural context control, which retrieves relevant historical operations and addresses the context explosion problem in long-sequence CAD generation. Experiments on Op-CAD and CAD-Editor show that MopCAD achieves stronger execution validity and geometric accuracy than general vision-language models and CAD-specific baselines, demonstrating a more controllable and scalable paradigm for complex CAD generation.

1 Introduction

Computer-aided design (CAD) modeling is essential for industrial design and manufacturing, yet it has a steep learning curve and requires substantial domain expertise. Recent advances in large language models and multimodal models have made natural language-based or image-based CAD modeling increasingly feasible.

CAD generation is fundamentally different from mesh-, point-cloud-, or voxel-based 3D generation. It is not merely about producing a 3D object with a similar overall appearance, but about constructing a procedural model composed of precise modeling operations. Although many studies have explored CAD generation and made substantial progress(Khan et al., 2024; Li et al., 2025b; Yuan et al., 2025), the generated models are still often limited to simple structures, with insufficient controllability and precision. When extended to high-precision and high-complexity CAD generation, existing methods commonly face several important yet unresolved challenges.

Mismatch between one-shot generation and precise CAD modeling. Most existing CAD generation methods follow a one-shot paradigm, where users provide natural language descriptions or images and the model directly generates a complete CAD model. Although this paradigm can produce visually plausible results for simple shapes, it often fails to follow precise CAD instructions, especially for complex models that require accurate local structures, spatial placement, and interpretable operation sequences. This issue is further amplified because complex CAD models are difficult to describe completely and precisely with a single natural language instruction.

Insufficient exploration of visual guidance. Existing methods usually use visual inputs as auxiliary conditions for understanding the overall shape. However, CAD modeling requires strong spatial reasoning and geometric analysis. Visual information can indicate not only the current or final object appearance, but also provide spatial cues for inferring the target region where the next operation should be applied. In current studies, visual inputs are rarely transformed from generic conditions into precise operation-level guidance for CAD instruction following, leaving their value for accurate localization and precise parameter prediction underexplored.

Context explosion in long-sequence CAD modeling. CAD modeling is naturally a sequence-accumulated process. As model complexity increases, the number of modeling steps also grows. Directly feeding the complete history into the model leads to rapidly increasing training and in-

ference costs, while also introducing redundant or irrelevant information. Such context explosion severely limits the ability of existing methods to handle complex CAD models, which is one important reason why most current CAD generation methods remain focused on short sequences and simple structures.

In this work, we propose MopCAD, a controllable multimodal CAD generation framework that decomposes CAD construction into step-wise operations and introduces intermediate visual guidance to localize the target region of each operation. To address context explosion in long-sequence CAD modeling, we further design a history selector that retrieves the most relevant historical operations for the current step. This enables the model to focus on useful procedural context while reducing redundant input, improving both generation efficiency and stability.

The contributions of this work are summarized as follows:

- We introduce a multimodal operation-oriented CAD generation framework that leverages intermediate visual cues as geometric localization guidance, improving the controllability, interpretability, and geometric precision of the modeling process.

- We propose a history selector module for context control in complex CAD model generation. To the best of our knowledge, this is the first solution to address the long-context challenge in complex CAD generation, alleviating the training and inference burdens caused by long procedural contexts.

- We conduct comprehensive experiments on Op-CAD and CAD-Editor benchmarks, comparing our method with general vision-language models and CAD-specific editing models. Experimental results show that our framework achieves better execution validity, geometric accuracy, and long-sequence CAD generation capability.

2 Problem Formulation

Rather than treating CAD generation as a one-shot prediction problem, we focus on the accuracy of step-wise operation generation. Each predicted operation is expected to correctly follow the current instruction and update the existing CAD state. We consider a controllable CAD modeling scenario where a user guides the construction process through a sequence of step-wise natural language instructions $\mathcal{X} = \{x_1, x_2, \ldots, x_T\}$, where x_t denotes the instruction at step t. Correspondingly, the CAD construction process is represented as a sequence of modeling operations or code snippets $\mathcal{C} = \{c_1, c_2, \ldots, c_T\}$, where c_t denotes the CAD operation code generated at step t.

At each step, the model is required to generate the next CAD operation according to the current instruction and the existing modeling state. In our task, the current modeling state is represented by two complementary sources: the historical modeling code and the multi-view renderings of the current CAD model. Specifically, the historical modeling code before step t is denoted as $H_{<t} = \{c_1, c_2, \ldots, c_{t-1}\}$, which provides precise parametric and procedural information. The visual state of the current CAD model is represented by multi-view renderings $V_{t-1} = I_{t-1}$.

Therefore, the step-wise CAD generation task is formulated as predicting the current operation code c_t conditioned on the current instruction, historical modeling code, and current visual state:

$$c_t = G(x_t, H_{<t}, V_{t-1}). \tag{1}$$

When $t = 1$, the historical code $H_{<t}$ is empty, and V_0 corresponds to a blank CAD canvas. Therefore, the modeling process starts only from the user's initial instruction.

After generating and executing c_t, the CAD state is updated accordingly, providing the modeling state for subsequent steps. In this way, the final CAD model can be progressively constructed through a sequence of step-wise operations.

3 Framework Overview

As shown in Figure 1, our framework consists of two main components: a two-stage visual-guided operation generation module and a history selector module for context control. The first component introduces intermediate visual geometric guidance into the generation process. Specifically, given I_t and V_{t-1}, we predict an operation-level visual representation $\hat{z}_t$, which is supervised by the latent representation z_t of the ground-truth operation highlight image $\mathcal{G}_t$. This representation serves as an implicit localization clue, helping the model identify where the next CAD operation should be applied.

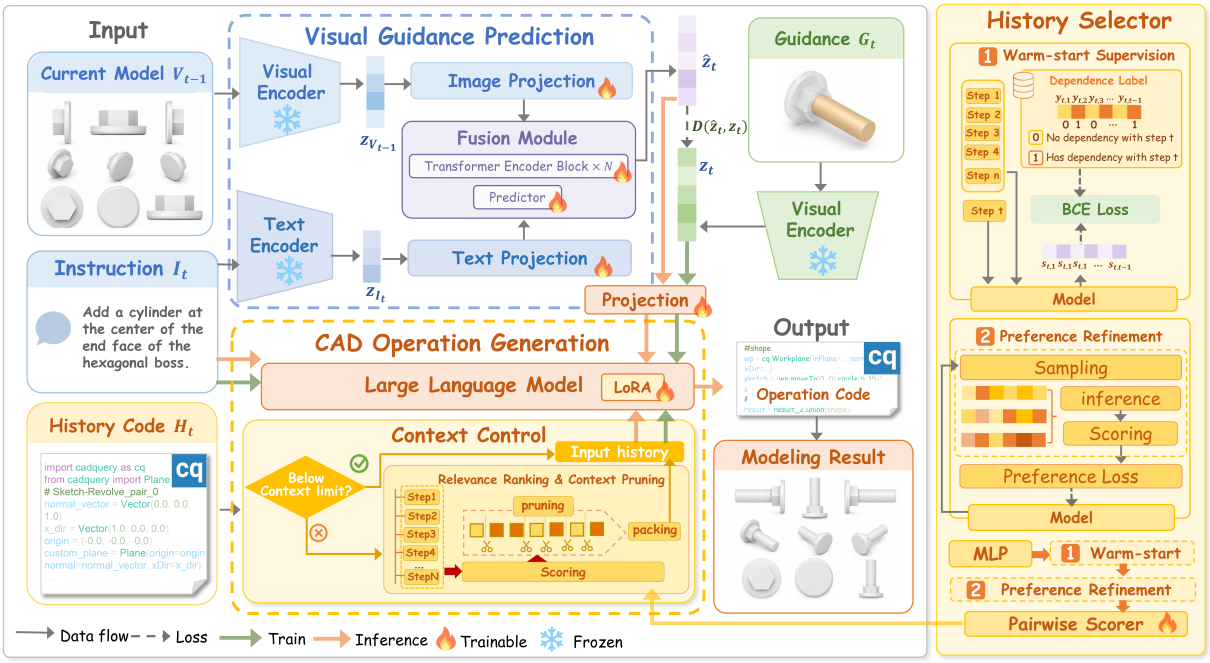


Figure 1: Overview of MopCAD. A user request is parsed into step-level CAD instructions. For each step, the visual action module predicts a visual edit embedding from the previous render and the instruction. The CAD operation generator injects the predicted embedding as soft tokens and generates executable CadQuery code conditioned on selected procedural history.

After obtaining $\hat{z}_t$, we project it into the language model embedding space as e_t. The projected visual guidance e_t is then combined with the instruction I_t and the historical CAD code tokens from $\mathcal{H}_{<t}$, and fed into the LLM-based operation generator G_θ. In this way, the generator predicts $\hat{o}_t$ conditioned not only on textual instructions and code history, but also on operation-level visual geometric cues.

To address the long-context burden caused by long CAD construction sequences, we further introduce a history selector. Instead of directly feeding the entire history $\mathcal{H}_{<t}$ into the generator, the selector evaluates the relevance between the current $stept$ and each previous operation $step0, step1, ..., step(t-1)$. It then selects a compact subset of relevant historical operations, denoted as $\mathcal{S}_t$. The generator finally uses I_t, e_t, and $\mathcal{S}_t$ to generate the next operation $\hat{o}_t$.

The selector is trained with a two-stage strategy. We first warm-start it using operation dependencies parsed from Op-CAD, so that it can learn explicit construction relationships between CAD operations. Then, we refine it with preference pairs constructed from downstream generation results, encouraging the selector to choose historical contexts that lead to better execution validity and geometric quality.

4 Visual-Guided CAD Generation

4.1 Visual Guidance Prediction

The goal of this stage is to predict a visual geometric guidance representation for the next CAD operation. Given the current instruction x_t and the previous CAD visual state V_{t-1}, the predictor F_ϕ predicts an operation-level visual representation:

$$\hat{z}_t = F_\phi(x_t, V_{t-1}). \tag{2}$$

We use the ground-truth operation highlight image $\mathcal{G}_t$ as supervision. $\mathcal{G}_t$ marks the geometric region changed by the current operation, providing a direct localization signal. It is encoded by a frozen visual encoder E_v to obtain the target representation:

$$z_t = E_v(\mathcal{G}_t). \tag{3}$$

The predictor is optimized by aligning the predicted representation with the target highlight representation:

$$\mathcal{L}_{vg} = \mathcal{D}(\hat{z}_t, z_t), \tag{4}$$

where $\mathcal{D}(\cdot)$ denotes the visual representation alignment objective, such as embedding distance or contrastive alignment loss.

During inference, the ground-truth highlight image is unavailable. Therefore, the predicted representation $\hat{z}_t$ is used as an implicit geometric localization cue and passed to the operation generation stage after being projected into the language model embedding space. This enables the generator to use not only textual instructions and historical CAD code, but also visual guidance indicating where the current operation should be applied.

4.2 CAD Operation Generation

This stage aims to generate the CAD operation code for the current step. We first use a visual-to-language projection module P_ω to map the highlight embedding into continuous visual soft tokens:

$$E_t^v = P_\omega(z_t), \quad z_t = E_{\text{hl}}(G_t), \tag{5}$$

where G_t is the ground-truth highlight image during training and E_{hl} is the frozen visual encoder.

The textual input consists of the current instruction I_t and the selected historical operations $H_{<t}$. After token embedding, we prepend the visual soft tokens to the text embeddings:

$$E_t = [E_t^v; E_t^{\text{text}}]. \tag{6}$$

The CAD operation generator G_θ then autoregressively predicts the current operation code:

$$\hat{o}_t = G_\theta(E_t). \tag{7}$$

The injected visual soft tokens provide operation-level geometric guidance, helping the model localize the target region and generate CAD code consistent with the current modeling state. During inference, the ground-truth highlight embedding is replaced by the predicted visual action embedding.

5 History Selector Module

In long-sequence CAD modeling, directly feeding the full operation history $\mathcal{H}_{<t}$ into the generator introduces high context cost and may include many irrelevant historical steps. To address this issue, we design a history selector to construct a compact and relevant historical context for each generation step.

5.1 Selector Training

We train the selector with a two-stage strategy. The first stage provides warm-start supervision from parsed operation dependencies. The second stage further refines the selector using downstream generation feedback.

Warm-start Supervision. In this stage, we use operation dependency labels parsed from Op-CAD to construct supervision signals. For each current step t and historical step $i < t$, we assign a relevance label $y_{t,i}$ indicating whether o_i is useful for generating o_t. The selector computes a relevance score $s_{t,i} = R_\psi(q_t, h_i)$ for each current-history pair, where q_t and h_i denote the current-step and historical-step representations, respectively. The selector is then trained to predict these relevance labels:

$$\mathcal{L}_{\text{sup}} = \sum_{i<t} \text{BCE}\big(\sigma(s_{t,i}), y_{t,i}\big), \tag{8}$$

where $\sigma(\cdot)$ is the sigmoid function.

Preference Refinement. After warm-start training, we refine the selector using downstream generation feedback. For each current step, the warm-start selector samples multiple candidate historical contexts. We feed each context into the LLM-based CAD generator, execute the generated CAD code, and score the result using execution validity and geometric metrics. We then construct preference pairs, where the context leading to a valid result or lower geometric error is treated as S_t^+, and the worse one as S_t^-.

The selector is optimized to assign higher scores to preferred contexts:

$$\mathcal{L}_{\text{pref}} = -\log \sigma\left(\text{Score}(S_t^+) - \text{Score}(S_t^-)\right). \tag{9}$$

This refinement stage directly aligns history selection with downstream CAD generation quality.

Through this two-stage training strategy, the selector learns not only structural dependencies from parsed CAD operations, but also which historical contexts are more beneficial for downstream CAD generation.

5.2 Context Control

The selector is used in both training and inference. During training, it constructs compact contexts for supervised operation generation, allowing the generator to learn from relevant historical dependencies instead of the full operation history. During inference, after each generated operation is executed and appended to the history, the selector re-ranks the accumulated history for the next step. This enables long-sequence CAD generation under a controlled context length.

For the current step t, the selector first embeds the current instruction and visual guidance into

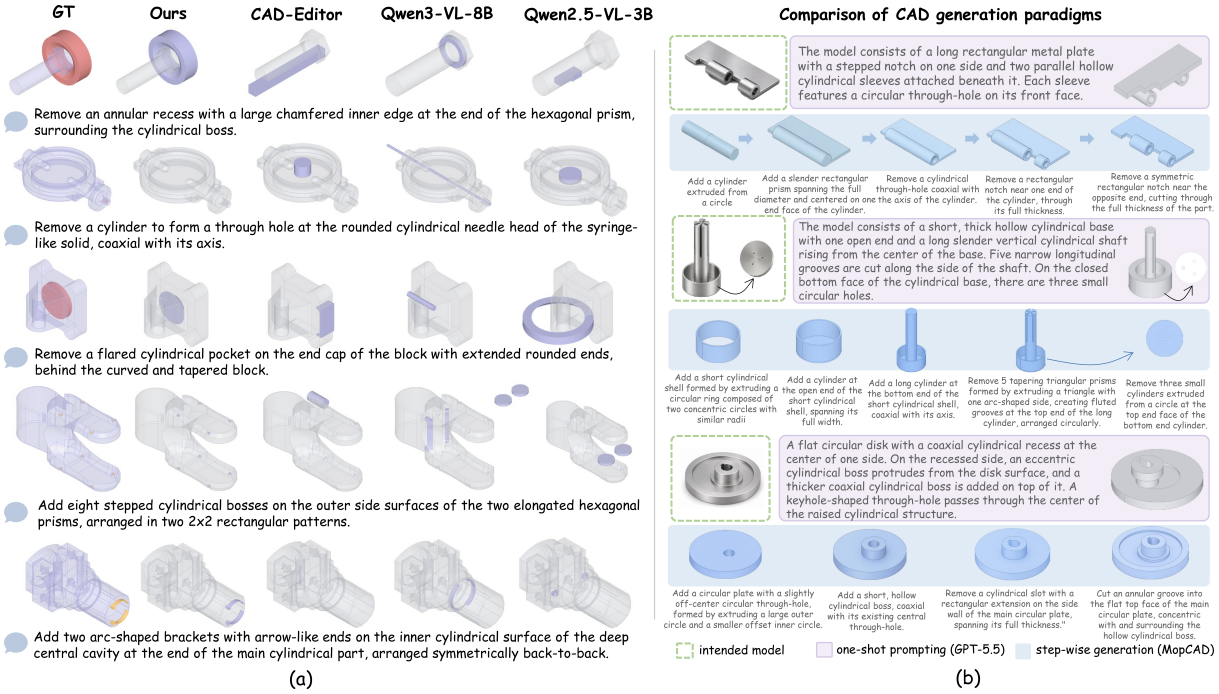


Figure 2: Qualitative Evaluation. The generated models usually preserve the coarse target structure and execute as CAD programs, but fine geometric differences remain visible in local features and proportions.

a current-step representation q_t. For each previous operation o_i with $i < t$, it also builds a historical-step representation h_i. A lightweight pairwise MLP R_ψ then estimates the relevance score $s_{t,i} = R_\psi(q_t, h_i)$, where $s_{t,i}$ measures how useful the historical operation o_i is for generating the current operation.

Based on the relevance scores, the selector ranks all previous operations and selects the most relevant ones under a fixed context budget:

$$S_t = \mathrm{Select}(\mathcal{H}_{<t}, s_{t,i}, B), \tag{10}$$

where B denotes the maximum context budget. After selection, the chosen steps are reordered chronologically to preserve the procedural order of CAD construction. The operation generator then predicts the current CAD code based on the current instruction, visual guidance, and selected history:

$$\hat{o}_t = G_\theta(x_t, E_t^v, S_t). \tag{11}$$

6 Experiments

6.1 Experimental Setup

Datasets. We use the Op-CAD dataset (Bai et al., 2026), which contains around 128K history-instruction-operation pairs for operation-oriented CAD generation. Each sample provides the previous modeling history, the current natural language instruction, and the target CAD operation. We filter the dataset to retain extrusion and revolution operations, and split it into 80% training, 10% validation, and 10% testing.

We further evaluate the generalization ability of our method on 500 selected sequence modeling samples from the CAD-Editor benchmark (Yuan et al., 2025).

Baselines. We compare our method with two types of baselines. First, we use open-source vision-language models, including Qwen2.5-VL-3B (Bai et al., 2025b) and Qwen3-VL-8B (Bai et al., 2025a), to evaluate whether general VLMs can directly use visual inputs for CAD operation generation. Second, we use CAD-Editor as a CAD-domain editing baseline, which allows us to compare our method with a specialized model designed for text-based CAD operation localization.

Metrics. We report Success Rate (SR), Chamfer Distance (CD), and Hausdorff Distance (HD). SR measures whether the generated result is valid and executable. CD evaluates the overall geometric consistency between the generated model and the ground truth, while HD measures the worst-case geometric deviation.

Implementation Details. We use frozen DINOv2-with-registers giant (Oquab et al., 2023)

Model	Op-CAD			CAD-Editor		
	SR ↑	Mean CD ↓	Mean HD ↓	SR ↑	Mean CD ↓	Mean HD ↓
Qwen2.5-VL-3B	29.41%	77.128	381.200	36.46%	66.033	402.359
Qwen3-VL-8B	74.10%	<u>29.874</u>	226.264	85.06%	45.952	274.100
CAD-Editor (8B)	42.15%	210.414	542.152	<u>95.31%</u>	41.880	295.779
MopCAD (3B)	<u>78.92%</u>	32.028	<u>218.575</u>	95.11%	<u>21.803</u>	<u>230.122</u>
MopCAD (7B)	**84.38%**	**20.769**	**214.677**	**96.36%**	**19.436**	**163.657**

Table 1: Main results on Op-CAD and CAD-Editor. SR measures generation validity. Mean CD and Mean HD measure 3D similarity and outlier discrepancy, respectively. Bold and underlined values denote the best and second-best results within each benchmark group.

Setting	SR ↑	Mean CD ↓	Mean HD ↓
MopCAD w/o Visual	69.64%↓11.76%	72.516↑126.41%	340.510↑55.79%
MopCAD w/ Shuffled Visual	68.70%↓12.95%	55.460↑73.16%	333.710↑52.68%
MopCAD w/o Selector	60.51%↓23.33%	33.796↑5.52%	240.830↑10.18%

Table 2: Ablation results. The red values indicate the relative performance degradation compared with the full MopCAD (3B) model.

and CLIP ViT-B/16 (Radford et al., 2021) as the visual and text encoders, respectively. Input images are resized to 518×518. During training, only the projection layers, fusion module, and prediction head are updated. For CAD operation generation, we adopt Qwen2.5-Coder-3B-Instruct (Hui et al., 2024) and Qwen2.5-Coder-7B-Instruct as the backbone models and fine-tune them with LoRA, resulting in MopCAD (3B) and MopCAD (7B). The visual action representation is injected into the language model as 16 continuous soft tokens. The history selector is implemented as a lightweight pairwise MLP. During inference, it retrieves relevant historical steps under a fixed 4096-token context budget and reorders the selected steps chronologically before generation.

6.2 Main Results

Quantitative Evaluation. Table 1 presents the main quantitative results on Op-CAD and CAD-Editor. MopCAD achieves consistently strong performance across both benchmarks. Compared with general VLM baselines and the CAD-specific CAD-Editor baseline, MopCAD improves generation validity while reducing geometric errors in most settings. In particular, MopCAD (7B) achieves the best results on all reported metrics, reaching 84.38% SR on Op-CAD and 96.36% SR on CAD-Editor, with the lowest Mean CD and Mean HD on both benchmarks. MopCAD(3B) also remains highly competitive, outperforming the CAD-specific CAD-Editor baseline in overall performance despite using a smaller backbone, and ranking second across the evaluated benchmarks. This suggests that the proposed visual-guided framework is effective across different backbone scales.

Qualitative Evaluation. Figure 2 presents representative qualitative comparisons between MopCAD (7B) and other models. In long-context single-step CAD generation, MopCAD better captures the key geometric parameters from previous modeling steps and accurately applies the required operation to the target region. Compared with other models, MopCAD can more reliably localize the operation position and generate the intended geometry, including additive structures such as cylindrical sleeves, vertical shafts, and coaxial bosses, as well as subtractive structures such as through-holes, grooves, annular recesses, and symmetric notches. The generated results also preserve important geometric relations, such as coaxial alignment, eccentric placement, parallel structures, and circular symmetry. These observations show that our framework helps the model better determine where and how the next CAD operation should be performed, leading to more stable and faithful long-sequence CAD generation.

We further compare MopCAD with a strong one-shot generation setting using GPT-5.5 for CAD modeling from scratch. As shown in Figure 2(b), one-shot generation tends to miss local structures or produce less controllable results when the target model is difficult to fully specify in a single instruction. In contrast, MopCAD decomposes the task into step-by-step operations, leading to more accurate final models and a more controllable generation process. This step-wise, human-centered workflow makes intermediate results easier to inspect, guide, and revise, reducing the cost of rework in complex CAD model creation.

6.3 Ablation Study

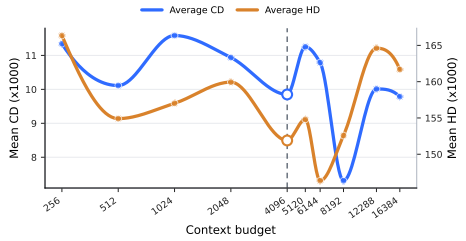
We conduct ablation studies based on MopCAD (3B) to evaluate the contribution of visual guidance and history selection. Specifically, we consider three settings: removing visual guidance (*w/o Visual*) to examine the importance of visual action information, shuffling the visual input (*w/ Shuffled Visual*) to test whether the model relies on step-aligned visual guidance, and removing the history selector (*w/o Selector*) to evaluate the importance of context selection in long-sequence CAD generation.

As shown in Table 2, all ablated settings lead to performance degradation, confirming the contribution of both visual guidance and history selection. Removing visual guidance reduces SR from 78.92% to 69.64% and significantly increases geometric errors, while shuffling the visual input also hurts performance despite preserving the visual tokens. This shows that MopCAD relies on step-aligned visual action information rather than simply benefiting from additional visual inputs. Removing the history selector further causes a large SR drop, indicating that selecting and organizing relevant historical steps is important for reliable long-context CAD generation. Overall, visual guidance improves operation localization and geometric fidelity, while the selector helps maintain generation validity under a limited context budget.

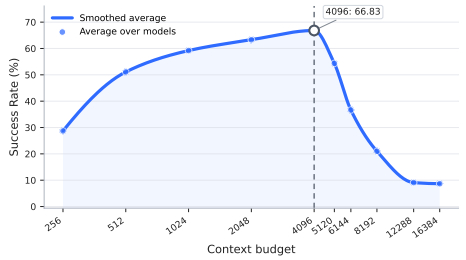
6.4 Effect of Historical Context Length

We further analyze how historical context length affects long-sequence CAD generation. Based on the ranked historical steps produced by the selector module, we sample 1,832 long-sequence operations and construct 10 context settings with token budgets ranging from 256 to 16,384. We evaluate MopCAD, Qwen2.5-VL-3B, and Qwen3-VL-8B under these settings, and report the average success rate and 3D geometric metrics.

As shown in Figure 3, model performance improves as the context budget increases from short to medium lengths, with the average success rate peaking around 4096 tokens. However, further increasing the context length does not bring consistent gains. Instead, the success rate drops sharply after 4096 tokens, while CD and HD fluctuate rather than improve monotonically. This suggests that longer histories do not necessarily provide more useful information; excessive context may introduce irrelevant or weakly related operations that distract the model from the current target step. These results support our selector-guided context construction strategy, which preserves useful modeling history while reducing interference from redundant long-context information.



(a) Average CD and HD across context budgets. Geometry quality does not improve monotonically with longer context, supporting the need for history selector.



(b) Average success rate across context budgets. Performance rises up to the mid-context range and then drops when very long contexts introduce irrelevant history.

Figure 3: Comparison of model performance under different context settings.

6.5 Failure Analysis

During inference, we identify four major failure modes of MopCAD, as shown in Figure 4. First, MopCAD still struggles with non-standard plane reasoning, where inaccurate plane parameters may lead to spatially misaligned operations. Second, cutout operations are more difficult than additive operations because subtraction requires predicting both the removed shape and its interaction with the existing geometry, resulting in occasional cutout shape errors. Third, highly symmetric objects can introduce position ambiguity, causing the model to apply the correct operation to an incorrect symmetric region. Finally, when one instruction involves multiple operations across different planes, MopCAD may confuse the target directions or reference planes. These cases suggest that future work should improve explicit plane reasoning, subtractive operation modeling, and spatial disambiguation for complex CAD structures.

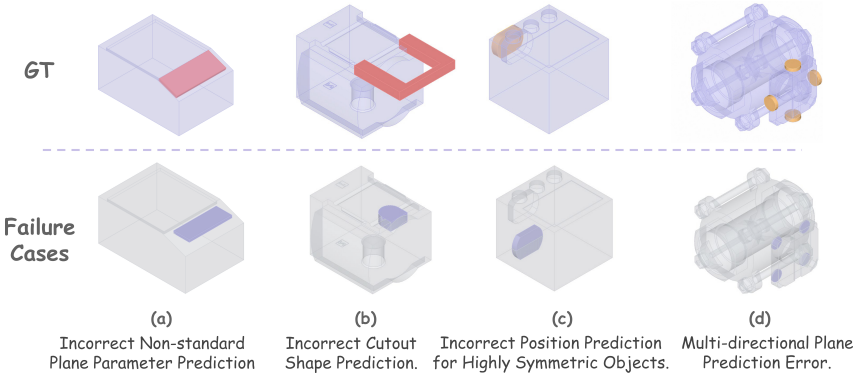


Figure 4: Failure cases of MopCAD. The top row shows the ground-truth operations, while the bottom row shows the corresponding failed predictions.

7 Conclusion

We introduced MopCAD, a controllable framework for step-wise CAD code generation with visual action guidance and history-aware context selection. MopCAD predicts visual action representations from rendered CAD states, injects them into code language models as continuous soft tokens, and retrieves relevant historical operations for long-sequence generation.

Experiments on Op-CAD and CAD-Editor show that MopCAD achieves stronger execution validity and geometric accuracy than general vision-language models and CAD-specific editing baselines. Overall, our work points toward a more fine-grained paradigm for CAD generation, making the modeling process more human-centered, interpretable, and controllable.

8 Related Works

Multimodal CAD Generation. A large body of work has explored language- and vision-conditioned CAD generation, covering text descriptions (Xie and Ju, 2025; Alam and Ahmed, 2024), single images (You et al., 2024; Chen et al., 2025) and point clouds (Uy et al., 2022; Liu et al., 2024; Rukhovich et al., 2025). Text2CAD (Khan et al., 2024) introduces natural language as a condition for generating sequential parametric CAD models, while CAD-Llama (Li et al., 2025b) adapts large language models to text-based parametric CAD generation. CAD-Coder (He et al., 2025) further reformulates text-to-CAD generation as executable CAD code synthesis, and STEP-LLM (Shi et al., 2026) directly generates CAD STEP models from natural language descriptions. Beyond text-only conditioning, CADDreamer (Li et al., 2025c) reconstructs CAD models from single-view images, CAD-MLLM (Xu et al., 2024) unifies text-, image-, and point-cloud-conditioned CAD generation, and CAD-GPT (Wang et al., 2025b) enhances multimodal CAD synthesis with spatial localization tokens. Recent VLM-based methods such as ReCAD (Li et al., 2025a) and `cadrille` (Kolodiazhnyi et al., 2025) further explore reinforcement learning for image-based or multimodal CAD code generation. However, their reliance on the one-shot paradigm limits their ability to handle long-history, localized, and step-wise industrial CAD modeling. Our work addresses this limitation by introducing visual action-guided operation generation and controllable procedural context selection.

Controllable and editing-based CAD generation. Several recent systems move beyond one-shot generation toward controllability. FlexCAD (Zhang et al., 2024) unifies controllable CAD generation across construction hierarchies by converting CAD models into structured text and masking hierarchy-aware fields. CAD-Editor (Yuan et al., 2025) introduces a locate-then-infill pipeline for text-based CAD editing. GeoCAD (Zhang et al., 2025) targets local geometry-controllable CAD generation with local-part captions for user-specified modifications. CADMorph (Ma et al., 2025) studies geometry-driven parametric CAD editing through a plan-generate-verify loop. CADKnitter (Le et al., 2025) explores compositional CAD generation from text and geometry guidance, while CADFusion (Wang et al., 2025a) introduces visual feedback into LLM-based text-to-CAD generation. Our method is closest in spirit to these controllable CAD generation and editing approaches, but focuses on operation-oriented generation with visual action guidance and history-aware context control.

Limitations

Despite the effectiveness of MopCAD, it also reveals several limitations that call for further efforts in future research.

First, MopCAD reduces the ambiguity of one-shot CAD generation by decomposing complex modeling tasks into step-wise operations. This makes the generation process more controllable and easier to inspect. However, natural language ambiguity cannot be fully eliminated. Each step still benefits from a reasonably clear description of the target region, operation type, and key parameters. When the instruction is highly vague or underspecified, the model may still struggle to infer the intended operation. This suggests that future CAD generation systems could further combine step-wise generation with stronger interactive control or auxiliary signals to support more precise instruction following.

Second, the visual guidance in MopCAD provides useful operation-level localization cues and improves geometric fidelity. Nevertheless, it is currently represented as an implicit embedding, which may not always capture fine-grained geometric constraints in difficult cases such as non-standard planes, small local edits, highly symmetric structures, or multi-plane operations. A promising future direction is to incorporate more explicit geometric information, such as reference planes, sketch constraints, user-specified points, or editable intermediate annotations, to further strengthen spatial reasoning.

Third, the evaluation of MopCAD is limited by the current lack of large-scale, diverse, and operation-level CAD datasets. To the best of our knowledge, our experiments are conducted on the most challenging available CAD data that provides step-wise modeling procedures, intermediate visual states, and operation-level annotations. This enables a realistic evaluation of controllable CAD generation, but the available data still falls short of covering the full complexity of real-world CAD modeling. In particular, existing datasets contain limited long-history examples with dense procedural dependencies, and they underrepresent many advanced CAD operations such as fillet, chamfer, shell, mirror, pattern, loft, and sweep. Therefore, our current evaluation focuses mainly on extrude and revolve operations, which are the most reliably available and representative construction operations in existing operation-level CAD datasets. Future progress in CAD generation will require larger and more diverse datasets with richer operation types, more complex modeling histories, and more realistic design workflows.

References

Md Ferdous Alam and Faez Ahmed. 2024. Gencad: Image-conditioned computer-aided design generation with transformer-based contrastive representation and diffusion priors. *arXiv preprint arXiv:2409.16294*.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025a. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Shuai Bai, Ke qin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025b. Qwen2.5-vl technical report. *ArXiv*, abs/2502.13923.

Yixue Bai, Yufei Gu, and Zeke Xie. 2026. Op-cad: Benchmarking and investigating operation-oriented cad generation. In *Proceedings of the 43rd International Conference on Machine Learning*. Accepted to ICML 2026.

Tianrun Chen, Chunan Yu, Yuanqi Hu, Jing Li, Tao Xu, Runlong Cao, Lanyun Zhu, Ying Zang, Yong Zhang, Zejian Li, and 1 others. 2025. Img2cad: Conditioned 3-d cad model generation from single image with structured visual geometry. *IEEE Transactions on Industrial Informatics*.

Changqi He, Shuhan Zhang, Liguo Zhang, and Jiajun Miao. 2025. Cad-coder: Text-guided cad files code generation. *arXiv preprint arXiv:2505.08686*.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. 2024. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. *Advances in Neural Information Processing Systems*, 37:7552–7579.

Maksim Kolodiazhnyi, Denis Tarasov, Dmitrii Zhemchuzhnikov, Alexander Nikulin, Ilya Zisman, Anna Vorontsova, Anton Konushin, Vladislav Kurenkov, and Danila Rukhovich. 2025. cadrille: Multi-modal cad reconstruction with online reinforcement learning. *arXiv preprint arXiv:2505.22914*.

Tri Le, Khang Nguyen, Baoru Huang, Tung D Ta, and Anh Nguyen. 2025. Cadknitter: Compositional cad generation from text and geometry guidance. *arXiv preprint arXiv:2512.11199*.

Jiahao Li, Yusheng Luo, Yunzhong Lou, and Xiangdong Zhou. 2025a. Recad: Reinforcement learning enhanced parametric cad model generation with vision-language models. *arXiv preprint arXiv:2512.06328*.

Jiahao Li, Weijian Ma, Xueyang Li, Yunzhong Lou, Guichun Zhou, and Xiangdong Zhou. 2025b. Cad-llama: leveraging large language models for computer-aided design parametric 3d model generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18563–18573.

Yuan Li, Cheng Lin, Yuan Liu, Xiaoxiao Long, Chenxu Zhang, Ningna Wang, Xin Li, Wenping Wang, and Xiaohu Guo. 2025c. Caddreamer: Cad object generation from single-view images. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21448–21457.

Yujia Liu, Anton Obukhov, Jan Dirk Wegner, and Konrad Schindler. 2024. Point2cad: Reverse engineering cad models from 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3763–3772.

Weijian Ma, Shizhao Sun, Ruiyu Wang, and Jiang Bian. 2025. Cadmorph: Geometry-driven parametric cad editing via a plan-generate-verify loop. *arXiv preprint arXiv:2512.11480*.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, and 1 others. 2023. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Danila Rukhovich, Elona Dupont, Dimitrios Mallis, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. 2025. Cad-recode: Reverse engineering cad code from point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9801–9811.

Xiangyu Shi, Junyang Ding, Xu Zhao, Sinong Zhan, Payal Mohapatra, Daniel Quispe, Kojo Welbeck, Jian Cao, Wei Chen, Ping Guo, and 1 others. 2026. Step-llm: Generating cad step models from natural language with large language models. *arXiv preprint arXiv:2601.12641*.

Mikaela Angelina Uy, Yen-Yu Chang, Minhyuk Sung, Purvi Goel, Joseph G Lambourne, Tolga Birdal, and Leonidas J Guibas. 2022. Point2cyl: Reverse engineering 3d objects from point clouds to extrusion cylinders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11850–11860.

Ruiyu Wang, Yu Yuan, Shizhao Sun, and Jiang Bian. 2025a. Text-to-cad generation through infusing visual feedback in large language models. *arXiv preprint arXiv:2501.19054*.

Siyu Wang, Cailian Chen, Xinyi Le, Qimin Xu, Lei Xu, Yanzhou Zhang, and Jie Yang. 2025b. Cad-gpt: Synthesising cad construction sequence with spatial reasoning-enhanced multimodal llms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 7880–7888.

Haoyang Xie and Feng Ju. 2025. Text-to-cadquery: A new paradigm for cad generation with scalable large model capabilities. *arXiv preprint arXiv:2505.06507*.

Jingwei Xu, Chenyu Wang, Zibo Zhao, Wen Liu, Yi Ma, and Shenghua Gao. 2024. Cad-mllm: Unifying multimodality-conditioned cad generation with mllm. *arXiv preprint arXiv:2411.04954*.

Yang You, Mikaela Angelina Uy, Jiaqi Han, Rahul Thomas, Haotong Zhang, Yi Du, Hansheng Chen, Francis Engelmann, Suya You, and Leonidas Guibas. 2024. Img2cad: Reverse engineering 3d cad models from images through vlm-assisted conditional factorization. *arXiv preprint arXiv:2408.01437*.

Yu Yuan, Shizhao Sun, Qi Liu, and Jiang Bian. 2025. Cad-editor: A locate-then-infill framework with automated training data synthesis for text-based cad editing. *arXiv preprint arXiv:2502.03997*.

Zhanwei Zhang, Kaiyuan Liu, Junjie Liu, Wenxiao Wang, Binbin Lin, Liang Xie, Chen Shen, and Deng Cai. 2025. Geocad: Local geometry-controllable cad generation. *ArXiv*, abs/2506.10337.

Zhanwei Zhang, Shizhao Sun, Wenxiao Wang, Deng Cai, and Jiang Bian. 2024. Flexcad: Unified and versatile controllable cad generation with fine-tuned large language models. *arXiv preprint arXiv:2411.05823*.