GACO-CAD: Geometry-Augmented and Conciseness-Optimized CAD Model Generation from Single Image

Yinghui Wang
East China Normal University
Shanghai, China
51265902004@stu.ecnu.edu.cn

Xinyu Zhang
East China Normal University
Shanghai, China
xyzhang@sei.ecnu.edu.cn

Peng Du
State Key Laboratory of CAD & CG, Zhejiang University
Hangzhou, Zhejiang, China
dp@zju.edu.cn

Abstract

Generating editable, parametric CAD models from a single image holds great potential to lower the barriers of industrial concept design. However, current multi-modal large language models (MLLMs) still struggle with accurately inferring 3D geometry from 2D images due to limited spatial reasoning capabilities. We address this limitation by introducing GACO-CAD, a novel two-stage post-training framework. It is designed to achieve a joint objective: simultaneously improving the geometric accuracy of the generated CAD models and encouraging the use of more concise modeling procedures. First, during supervised fine-tuning, we leverage depth and surface normal maps as dense geometric priors, combining them with the RGB image to form a multi-channel input. In the context of single-view reconstruction, these priors provide complementary spatial cues that help the MLLM more reliably recover 3D geometry from 2D observations. Second, during reinforcement learning, we introduce a group length reward that, while preserving high geometric fidelity, promotes the generation of more compact and less redundant parametric modeling sequences. A simple dynamic weighting strategy is adopted to stabilize training. Experiments on the DeepCAD and Fusion360 datasets show that GACO-CAD achieves state-of-the-art performance under the same MLLM backbone, consistently outperforming existing methods in terms of code validity, geometric accuracy, and modeling conciseness.

***Keywords:** CAD Generation, MLLM, Spatial Understanding, Reinforcement Learning*

1. Introduction

Computer-Aided Design (CAD) provides powerful support for industrial part design through a formalized language of modeling operations. In modern smart manufacturing, nearly all products are digitally defined and iterated as CAD models before entering mass production. However, mastering this operational language entails a steep learning curve: engineers typically require years of training to fluently manipulate parameters, constraints, and feature trees to translate abstract geometric concepts into manufacturable and maintainable digital prototypes. This difficulty poses a critical bottleneck within the established ”design–verify–manufacture” cycle.

Consequently, the ability to rapidly generate accurate and editable CAD models—much like text or image generation—has emerged as a key challenge for both industry and academic research. Traditional 3D generation methods [28, 36, 15, 3, 22] mostly produce non-procedural geometric representations, lacking parametric modeling histories and thus unsuitable for downstream industrial editing. A number of recent studies[37, 42, 32, 12, 24, 13, 10, 20] adopted the Large Language Models (LLMs) framework to treat CAD generation as a code-generation problem. Despite this progress, the generated outputs currently lack the necessary geometric precision and code correctness required for real-world industrial applications.

Single-view reconstruction is highly valuable during prototyping, as it allows users to rapidly generate a 3D model from a single-perspective sketch for immediate feasibility validation; hence, this paper focuses on single-view reconstruction. The core difficulty stems from the need to infer accurate 3D spatial relationships using just a single 2D image. Existing Multi-modal Large Language Models (MLLMs) are predominantly trained on tasks like image-

Figure 1: Demonstration of various CAD models generated by GACO-CAD.

text matching, visual question answering and document understanding, which do not explicitly encode the 3D geometry implied by the images. Consequently, these models exhibit limited spatial reasoning abilities for single-view reconstruction. Although some studies [34, 6, 4] explored spatial reasoning in general vision-language contexts, such efforts remain limited in CAD model generation.

Meanwhile, reinforcement learning (RL) as a key post-training technique for large language models (LLMs) [41, 43, 26, 23] recently shows strong potential in CAD generation [12] and effectively improves output quality. However, existing approaches focus exclusively on geometric accuracy and code validity while neglecting modeling conciseness—a critical factor in industrial practice, where redundant operations reduce code readability and increase editing costs.

To address the two key challenges of limited geometric understanding and redundant generation sequences, we propose GACO-CAD, a two-stage post-training framework for single-view image-to-CAD generation. In the supervised fine-tuning(SFT) stage, we introduce RGB images, depth maps, and surface normal maps as multi-channel inputs, leveraging dense geometric priors to enhance the MLLM's 3D spatial reasoning. In the RL stage, we propose a group length reward mechanism that jointly optimizes geometric accuracy and sequence conciseness, stabilized by a simple dynamic weighting strategy. Experimental results show that GACO-CAD outperforms existing methods on both DeepCAD and Fusion360 datasets, achieving state-of-the-art performance across code validity, geometric accuracy, and modeling conciseness. Some successful CAD models generated by GACO-CAD are shown in Figure 1. The key contributions of our study are summarized as follows:

- We analyze the significance of dense geometric priors in enhancing space understanding for MLLM-based single-view CAD generation and introduce these priors during MLLM training and inferring. By integrating these priors during both MLLM training and inference, we effectively improve the geometric accuracy of the generated CAD models.

- We design a group length reward mechanism that explicitly represents generation conciseness as an optimization objective, expanding a new evaluation dimension for generating CAD models.

- Our approach outperforms previous MLLM-based methods on both DeepCAD and Fusion360 datasets under the same MLLM backbone, substantially improving the performance of single-view CAD generation task.

2. Related Work

CAD Model Generation. Editable CAD models are typically represented as parametric operation sequences or code with modeling APIs[1]. Early works[35, 39, 7] employed Transformer architectures[29] to autoregressively generate modeling command sequences, laying the foundation for procedural CAD generation. Recent studies further introduced hierarchical structures [38] or diffusion-based mechanisms[19, 5, 17] to generate CAD sequences conditioned on text or images. However, these approaches often rely on non-intuitive, low-level command representations, limiting code readability and editability.

With the rise of LLMs and MLLMs, CAD generation is increasingly framed as a structured code generation task. Recent work [21, 13] directly leverages the reasoning capabilities of general-purpose large models via agent-based systems that decompose complex design requirements. However, in the absence of task-specific training, these systems yield low pass-at-one accuracy and demand elaborate prompt crafting plus multi-turn dialogues, inflating inference costs. Concurrent efforts focus on dedicated training: OpenECAD[42] firstly explored image-to-CAD

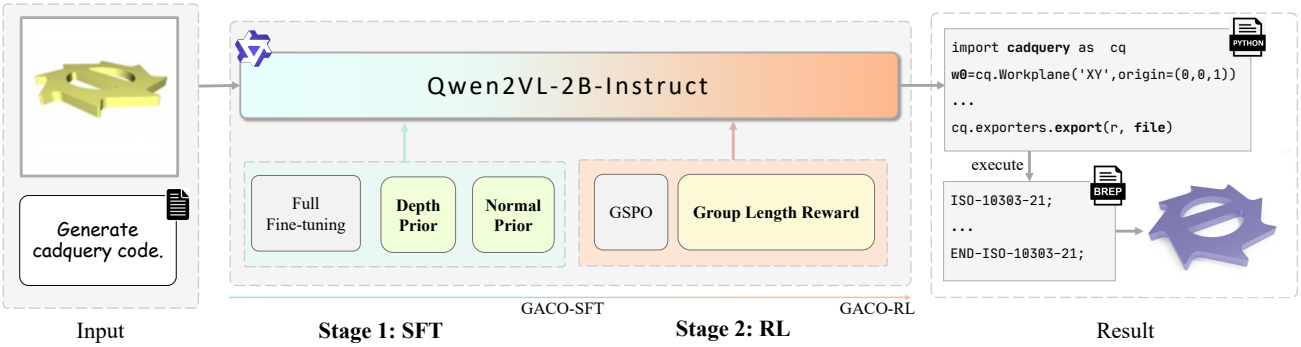

Figure 2: **Main Pipeline of Two-stage Training.** Starting from pretrained weights, we obtain the final model through two training stages: SFT and RL. In the SFT stage, we perform full fine-tuning with depth and normal priors; in the RL stage, we optimize with the GSPO algorithm incorporating a length-reward rule. After training, the model takes a single-view image and simple text as input and generates target Python code, whose execution yields the final B-rep file.

generation on multimodal large models, though it was limited by early model capabilities and context length. Subsequent efforts[37, 24, 11] extended input modalities and refined architectures, but still struggle to meet industrial demands for precision and robustness.

Spatial Understanding in MLLMs. Current MLLMs[30, 44, 2] excel on standard image-text understanding benchmarks, but exhibit clear limitations in spatial perception. They struggle to accurately infer 3D positions, distances, or geometric structures from a single 2D image. This stems primarily from their pre-training objectives, which emphasize 2D image-text alignment rather than geometrically precise spatial understanding.

To bridge this gap, some studies[34, 6, 16, 4] attempt to inject depth estimation or spatial relation data during pre-training or fine-tuning to endow MLLMs with better 3D reasoning capabilities. In the CAD domain, CAD-GPT[32] introduced an innovative spatial localization mechanism using dedicated tokens to enhance contextual awareness, but this approach relies on known modeling histories and is unsuitable for open-ended generation scenarios. Recent advances in dense geometric estimation [8, 9, 40, 31] made it possible to reliably estimate high-fidelity depth and surface normal maps from a single RGB image. Building on this foundation, we incorporate depth and normal maps as geometric priors into the MLLM input to enhance the model's shape perception and generalization.

Reinforcement Learning for LLM Post-Training. Reinforcement learning plays a pivotal role in post-training large models, widely applied to human preference alignment, tool usage and code generation. Early approaches like RLHF-PPO[25] require maintaining multiple networks and are complex to implement. DPO[23] reformulates preference data into a cross-entropy loss, simplifying training but lacking support for online exploration. GRPO[26], DAPO[41] and CPPO[14] improve efficiency through group sampling and self-normalized advantage estimation for on-policy learning. The recent GSPO[43] further introduces sequence-level importance sampling and clipping, achieving enhanced training stability without sacrificing simplicity.

Despite these advances in general domains, RL applications in CAD generation remain in the early stage of exploration. Cadrille[12] was the first to apply RL training to MLLM-based CAD generation, using rewards based on code validity and IoU to improve output quality. We argue that RL's potential in this domain is far from fully exploited. Building upon these foundations, we are the first to explicitly optimize modeling conciseness in the RL stage. We design a holistic reward function that enforces geometric correctness and simultaneously encourages the generation of more concise and efficient modeling programs.

3. Preliminary

3.1. LLM Supervised Fine-Tuning

Supervised Fine-Tuning (SFT) adapts a pretrained model to a target output distribution using labeled data, thereby improving downstream accuracy and consistency. It maximizes the likelihood of the target sequence given the context. Let x denote the multimodal context and $y = (y_1, \ldots, y_T)$ the desired cadquery code [1]. The negative log-likelihood loss is

$$\mathcal{L}_{\text{SFT}}(\theta) = -\mathbb{E}_{(x,y)\sim\mathcal{D}} \sum_{t=1}^{T} \log \pi_\theta(y_t \mid y_{<t}, x), \quad (1)$$

where $\mathcal{D}$ is the supervised dataset, θ the model parameters, and π_θ the token-level probability. In this paper, x contains a single RGB image, depth and normal maps, plus brief text. More details are given in § 4.1.2.

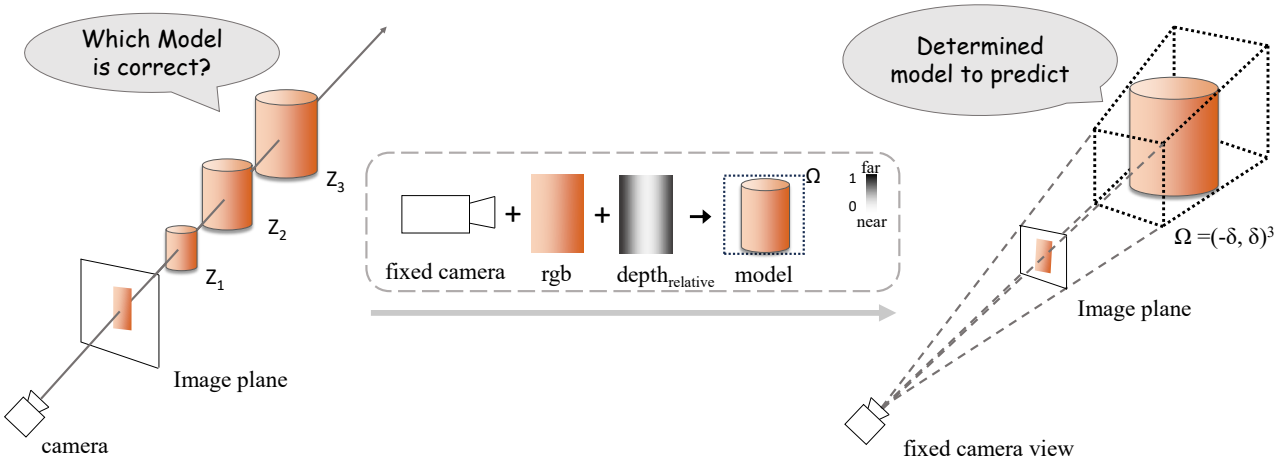

Figure 3: **The Role of Deep Prior in Monocular 3D Reconstruction.** When the camera parameters and the limited output model bounding box are defined, the introduction of the relative depth map theoretically provides sufficient conditions for reconstructing the 3D model of the visible part in a single view.

3.2. LLM Reinforcement Learning

After SFT, Reinforcement Learning refines the policy without additional labels by optimizing a quality-oriented reward. We adopt Group Sequence Policy Optimization (GSPO)[43], which performs group-wise updates using sequence-level rewards. The GSPO objective $\mathcal{J}_{\text{GSPO}}(\theta)$ is

$$\mathbb{E}\Big[\frac{1}{G}\sum_{i=1}^{G}\min\big(s_i(\theta)\hat{A}_i,\ \text{clip}(s_i(\theta),1-\varepsilon,1+\varepsilon)\hat{A}_i\big)\Big], \quad (2)$$

where the expectation is taken over $x \sim \mathcal{D}$ and G candidates $y_i \sim \pi_{\theta_{\text{old}}}$, $s_i(\theta)$ is the importance weight, and ε the clipping radius. The advantage $\hat{A}_i$ is computed via group normalization

$$\hat{A}i = \frac{r(x,y_i) - \text{mean}\big(r(x,y_i)_{i=1}^{G}\big)}{\text{std}\big(r(x,y_i)_{i=1}^{G}\big)}, \quad (3)$$

with $r(\cdot)$ a rule-based reward that evaluates the quality of the generated cadquery code.We will provide further details on the reward rules in § 4.2.

4. Method

As illustrated in Figure 2, our training proceeds in two tightly coupled stages. First, SFT injects geometric priors of depth and surface normals into a pretrained MLLM, yielding an initial policy (GACO-SFT) that already produces executable CAD code from a single image and brief prompt. Second, we refine the policy via a novel group-length reward that balances geometric accuracy with code conciseness. This yields the final model, GACO-RL, which generates compact yet accurate Python scripts; executing them directly returns the target B-rep.

4.1. Enhancing Spatial Understanding with Geometric Priors

Here, we propose a geometric prior-based approach to enhance the spatial understanding of the MLLM for single-view CAD model generation. We first analyze the non-uniqueness of 3D reconstruction from a single 2D image in § 4.1.1, emphasizing the importance of depth priors. Then, in § 4.1.2, we detail how geometric priors are integrated into the SFT stage.

4.1.1 Depth Prior in Single-View 3D Reconstruction

Reconstructing accurate 3D structures from a single RGB image is an ill-posed problem due to its geometric non-uniqueness. For a pixel coordinate $\mathbf{x} = (u, v)$, its corresponding 3D point $\mathbf{X}_w$ satisfies:

$$\mathbf{X}_w = \mathbf{R}^{-1}\left(Z\mathbf{K}^{-1}\tilde{\mathbf{x}} - \mathbf{t}\right), \quad Z > 0, \quad (4)$$

where $\tilde{\mathbf{x}} = (u, v, 1)^T$ is the homogeneous coordinate, $\mathbf{K}$ is the camera intrinsic matrix, and $(\mathbf{R}, \mathbf{t})$ denotes the extrinsic parameters. Since both depth Z and camera parameters are unknown, there exist infinitely many possible 3D points corresponding to a single pixel, making single-view 3D reconstruction inherently ambiguous.

In practice, training data for image-to-CAD generation is often rendered using fixed camera intrinsics and extrinsics, and the ground-truth 3D models are constrained within a bounding box $\Omega = [-\delta, \delta]^3$, as illustrated in Figure 3. Under this condition, the depth variable Z is implicitly restricted to a known interval. If the relative depth ordering of surface points is provided, the 3D coordinates of visible regions in the single view can be uniquely determined. The

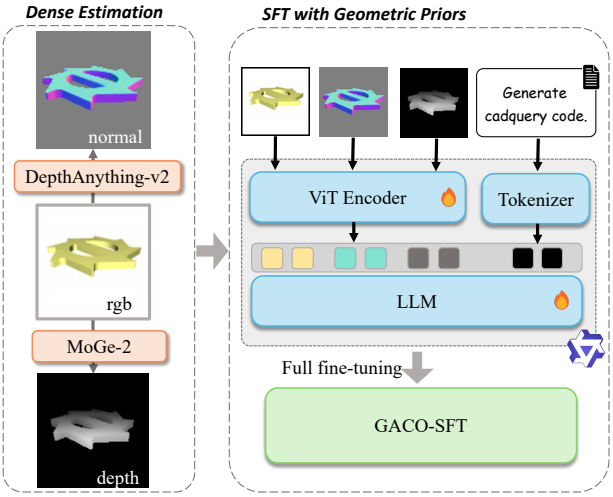

Figure 4: **SFT Pipeline.** We use Moge-2 and DepthAnything v2 to estimate normal map and depth map. All three images are processed by the shared ViT encoder, concatenated with the text, and fed into the LLM and get the first-stage model: GACO-SFT.

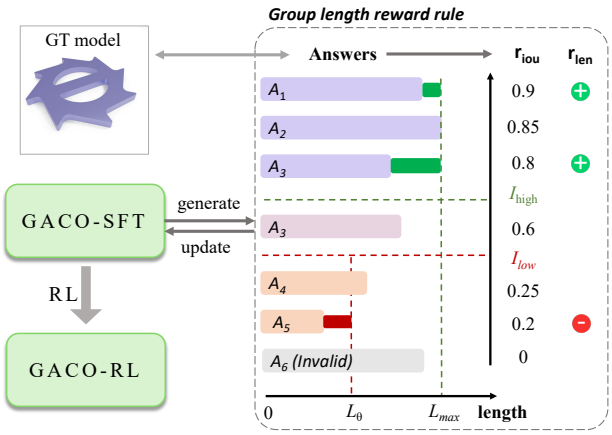

Figure 5: **Reinforcement Learning with Group Length Reward Rule.** We conducted RL training in GACO-SFT. In the generated responses, those with high IoU accuracy would be rewarded based on the group relative token length, while responses with low IoU and below the length threshold would be penalized. Finally, we obtained the two-stage model: GACO-RL.

depth map $Z_{\text{rel}}(u, v)$ offers this relative ordering, serving as a key geometric constraint.

It is crucial to highlight that MLLMs do not engage in explicit back-projection. Instead, their utilization of depth information is grounded not in geometric reasoning but in the alignment of RGB-depth statistical co-occurrences throughout the comprehensive pre-training and fine-tuning phases. Despite this, depth information remains vital in conveying geometric relationships. By transforming "geometric correctness" into an easily learnable mapping characterized by "high feature co-occurrence probability," the complexity of reconstructing the 3D structure from solely RGB views is diminished. Guided by this understanding, we incorporate depth priors as additional input during training to augment the model's spatial comprehension and enhance the accuracy of single-view CAD generation.

4.1.2 Supervised Fine-Tuning with Geometric Priors

During SFT, in addition to the RGB image I^{rgb} and depth map I^{dep}, we introduce surface normals I^{nor} as an extra geometric modality. While depth provides relative ordering among pixels, normals offer texture-independent surface orientation cues, helping the model focus on shape boundaries and curvature changes without being distracted by RGB texture ambiguities.

All three visual modalities $(I^{\text{rgb}}, I^{\text{dep}}, I^{\text{nor}})$ are processed by the shared pretrained vision encoder $E_v(\cdot)$ to extract patch-level features: $H^{rgb} = E_v(I^{rgb}), H^{\text{dep}} = E_v(I^{\text{dep}}), H^{\text{nor}} = E_v(I^{\text{nor}})$. These features are then projected into the LLM embedding space via the shared projection layer $P(\cdot)$: $Z^{rgb} = P(H^{rgb}), Z^{\text{dep}} = P(H^{\text{dep}}), Z^{\text{nor}} = P(H^{\text{nor}})$. The final input sequence is a concatenation of visual tokens and text prompt embeddings: $x = [Z^{rgb}, Z^{\text{dep}}, Z^{\text{nor}}, T]$, where T denotes the brief text prompt embedding, and $[\cdot]$ represents concatenation along the token dimension.

The training pipeline is illustrated in Figure 4. Surface normals are estimated using MoGe-2 [31], and depth maps are generated using DepthAnything v2 [40]. Ablation studies on different processing and fusion strategies for geometric priors are further presented in § 5.3.

4.2. Accuracy and Simplicity Balanced Reinforcement Learning

While supervised fine-tuning aligns the model well with task objectives, it struggles to optimize for conciseness. Redundant operations reduce code interpretability and increase editing costs, while long sequences incur higher computational and memory overhead. Without expert-annotated concise data, optimizing for simplicity is challenging under pure supervision.

Inspired by GSPO [43], we propose a group relative length reward mechanism that balances geometric accuracy and code simplicity via reinforcement learning.

4.2.1 Group Length Reward Function

We define a rule-based reward function $r(x, y)$ composed of three terms: code validity, geometric accuracy, and se-

quence length:

$$r(x,y) = \lambda_{\text{len}} r_{\text{len}}(x,y) + \lambda_{\text{iou}} r_{\text{iou}}(x,y) + \lambda_{\text{val}} r_{\text{val}}(x,y), \quad (5)$$

where $\lambda_{\text{len}}, \lambda_{\text{iou}}, \lambda_{\text{val}}$ are corresponding weights.

Code validity reward r_{val} checks whether the generated code is syntactically correct and executable. Geometric accuracy reward r_{iou} measures the IoU between the generated and ground-truth CAD models. Length reward r_{len} encourages conciseness among high-accuracy candidates.

As shown in Figure 5, given input x, the old policy generates G candidate sequences $\{y_i\}_{i=1}^{G}$ with token lengths $\{L_i\}_{i=1}^{G}$ and IoU scores $\{I_i\}_{i=1}^{G}$. Among high-accuracy candidates ($I_i \geq I_{\text{high}}$), we define a relative length ratio:

$$\alpha_i = \frac{L_i}{\max_j L_j}, \quad \alpha_i \in (0,1]. \quad (6)$$

The length reward is then:

$$r_{\text{len}}(x,y_i) = 0.5 + 0.5(1-\alpha_i)\cdot I_i, \quad \text{if } I_i \geq I_{\text{high}}. \quad (7)$$

Among accurate candidates, shorter sequences receive higher rewards.

For low-accuracy candidates ($I_i \leq I_{\text{low}}$), if the length is below a threshold L_θ, a penalty is applied:

$$r_{\text{len}}(x,y_i) = 0.5 - 0.5 \cdot \left(1 - \frac{L_i}{L_\theta}\right), \quad \text{if } L_i < L_\theta, \quad (8)$$

otherwise, a neutral reward $r_{\text{len}} = 0.5$ is given. No length guidance is applied for medium-accuracy candidates ($I_{\text{low}} < I_i < I_{\text{high}}$), ensuring the model prioritizes geometric correctness.

4.2.2 Dynamic Weight Scheduling

The importance of each reward component varies throughout the training process. In the early stages, the model focuses on ensuring code validity and geometric accuracy, whereas later stages emphasize conciseness. To achieve a smooth transition between these objectives, we adopt a linear weight scheduling strategy:

$$\lambda_k(t) = \lambda_k^{\text{start}} + \frac{t}{T}\left(\lambda_k^{\text{end}} - \lambda_k^{\text{start}}\right), \quad (9)$$

where t denotes the current training step, T is the total number of steps, and $\lambda_k^{\text{start}}, \lambda_k^{\text{end}}$ are initial and final weights for each component.

Specifically, we set $\lambda_{\text{len}}^{\text{start}} < \lambda_{\text{len}}^{\text{end}}$, $\lambda_{\text{iou}}^{\text{start}} > \lambda_{\text{iou}}^{\text{end}}$ and $\lambda_{\text{val}}^{\text{start}} > \lambda_{\text{val}}^{\text{end}}$ ensuring that early training focuses on correctness, while later stages gradually emphasize conciseness. This dynamic scheduling encourages the model to progressively balance geometric accuracy and brevity, resulting in compact yet precise CAD modeling code.

5. Experiments

To validate the effectiveness of GACO-CAD, we conducted a two-stage post-training process as previously described and compared our method with existing works on the single-view CAD model generation task. In the ablation study, we further investigated the impact of two types of geometric priors and the length reward mechanism on model performance. Additionally, we analyzed the network architecture used for processing geometric prior images to demonstrate the effectiveness and design rationality of the simple integration strategy adopted in GACO-CAD.

5.1. Experimental Setups

Datasets. During the SFT stage, we used approximately one million samples from the Recode dataset[24] as training data. All CAD models in the Recode dataset were constructed via procedurally generated CAD code within a bounded domain $\Omega = [-100, 100]^3$. Although the model distribution differs from real industrial parts, its highly randomized spatial structures provide strong training signals for MLLMs to understand spatial mappings.

In the RL stage, we utilized approximately 50,000 real industrial CAD models from DeepCAD[22] and 3,000 from Fusion360[33] to guide the model toward generating CAD structures that better align with real-world engineering requirements. The test data includes 8,046 samples from DeepCAD and 1,725 from Fusion360 for a comprehensive evaluation of model performance.

Details. For a fair comparison with prior work[12], we adopted the same pretrained model, Qwen2VL-2B-Instruct[30], as the base architecture and performed two-stage post-training. The SFT stage was conducted on a single H800 GPU using the AdamW optimizer[18] with a learning rate of 2×10^{-4}, weight decay of 0.01, a total batch size of 32 and 4 epochs. The RL stage was carried out on four H800 GPUs with VeRL[27] framework, with a learning rate of 1×10^{-6}, a mini-batch size of 128, policy update every 4 iterations and 2 epochs in total.

In RL training, we set the IoU upper threshold $I_{\text{high}} = 0.8$ and lower threshold $I_{\text{low}} = 0.4$. The length threshold L_{theta} was set to 110, determined by the statistical mean of generated code lengths in the RL training dataset. Reward weights were dynamically adjusted during training: the length reward weight increased from $\lambda_{\text{len}}^{\text{start}} = 0$ to $\lambda_{\text{len}}^{\text{end}} = 0.4$; the IoU reward weight decayed from $\lambda_{\text{iou}}^{\text{start}} = 0.8$ to $\lambda_{\text{iou}}^{\text{end}} = 0.5$; and the validity reward weight decayed from $\lambda_{\text{val}}^{\text{start}} = 0.2$ to $\lambda_{\text{val}}^{\text{end}} = 0.1$.

All input images were uniformly resized to 134×134. RGB images were rendered following the same pipeline as in cadrille[12] that rendering the normalized model within the bounding box $[-1, 1]^3$ from a fixed camera viewpoint.

Evaluation Metrics. We adopt the evaluation metrics from prior works[24, 12] to assess the quality of generated

Table 1: **Quantitative evaluation of single view CAD generation On DeepCAD dataset and Fusion360 Dataset.** The best results are **bold**. CD uses the median val multiplied by 10^3; ATL is the average token length produced by the Qwen2VL-2B-Instruct tokenizer. ↓: lower is better; ↑: higher is better.

Method	IR (%)↓	IoU (%)↑	CD↓	ATL↓
DeepCAD dataset				
CAD-GPT			9.77	
CAD-MLLM			3.77	
Img2CAD			1.61	
CADCrafter			0.82	
cadrille-SFT(80k)	1.75	73.55	0.37	99.8
Ours-SFT(80k)	1.57	75.44	0.33	98.8
cadrille-SFT(1M)	1.24	84.54	0.22	97.6
Ours-SFT(1M)	1.12	85.67	0.19	99.2
cadrille-RL	0.71	85.99	0.19	97.7
Ours-RL	**0.67**	**86.49**	**0.18**	**93.1**
Fusion360 dataset				
cadrille-SFT(80k)	3.45	64.58	0.54	121.8
Ours-SFT(80k)	3.30	65.87	0.51	122.6
cadrille-SFT(1M)	1.57	77.15	0.22	126.2
Ours-SFT(1M)	1.39	78.60	0.20	125.5
cadrille-RL	1.20	78.72	0.20	125.9
Ours-RL	**1.16**	**79.47**	**0.19**	**118.4**

CAD code from three perspectives: Invalid Rate (IR), Intersection over Union (IoU) and Chamfer Distance (CD). IR reflects the proportion of generated code that fails to execute successfully. IoU and CD measure the geometric accuracy of the generated models. When computing CD, all models are normalized to the $[-0.5, 0.5]^3$ space, and results are reported as the median CD multiplied by 10^3.

Additionally, to evaluate the conciseness of the generated code, we introduced the Average Token Length (ATL) metric, representing the token length of the generated code text. This metric is computed using the tokenizer of Qwen2VL-2B-Instruct and is only compared among baseline methods using the same cadquery code.

5.2. Performance

Quantitative Results. Table 1 presents the quantitative comparison between GACO-CAD and existing methods on the single-view CAD generation task. We report the performance of three model variants: Ours-SFT(80k), a small-scale SFT model trained on 80,000 Recode samples to assess the role of geometric priors under limited data; Ours-SFT(1M), the SFT model trained on the full 1,000,000 data; and Ours-RL, the final version further fine-tuned with RL. As the primary baseline, we trained three corresponding versions of the cadrille method on the same data scales for fair comparison.

Table 2: **Ablative evaluation of geometric prior combinations On DeepCAD dataset.** $\text{Ours}_{\text{wo-dn}}$: RGB only; $\text{Ours}_{\text{wo-n}}$: RGB + depth; Ours: RGB + depth + normal.

Method	IR (%)↓	IoU (%)↑	CD↓
$\text{Ours}_{\text{wo-dn}}$	1.75	73.55	0.37
$\text{Ours}_{\text{wo-n}}$	1.61	74.37	0.35
Ours	1.57	75.44	0.33

Results show that our method outperforms the previous best approach in terms of IR, IoU, and CD under both 80,000 and 1,000,000 data scales, validating the effectiveness of introducing geometric priors for MLLM in understanding single-view geometry. Notably, the performance gain from geometric priors is more significant under small-scale training (80,000). We conjecture that large-scale SFT allows the model to implicitly learn accurate depth and spatial relationships from RGB views, partially reducing the marginal benefit of geometric priors. This outcome is consistent with our mechanism analysis.

Furthermore, the RL-trained model not only improves geometric accuracy and code validity but also demonstrates advantages in ATL. On the DeepCAD and Fusion360 test sets, the token lengths of generated code are reduced by approximately 6.1% and 5.7%, respectively. Despite cadquery's inherent conciseness, the proposed length reward mechanism successfully guides the policy network to generate more compact modeling code. Overall, GACO-CAD achieves state-of-the-art performance on the single-view CAD generation task across all metrics.

Qualitative results. Qualitative results are shown in Figure 6. We compare the models produced by the previous best baseline and our method on the single-view CAD generation task. Owing to the injected geometric priors, our approach yields CAD models with more accurate geometry and finer, more plausible details. Although we used 134×134 ultra-low resolution images, after introducing geometric priors, MLLM can more easily distinguish key geometric information such as holes in the model.

5.3. Ablation Study

Types of Geometric Priors. To analyze the contribution of the two types of geometric priors (depth and normal maps), we trained three model variants on 80,000 Recode data and conducted ablation experiments on the DeepCAD test set. As shown in Table 2, the model using only RGB input ($\text{Ours}_{\text{wo-dn}}$) performs the worst; introducing depth priors ($\text{Ours}_{\text{wo-n}}$) brings some improvement; and the full model using both depth and normal priors performs the best, val-

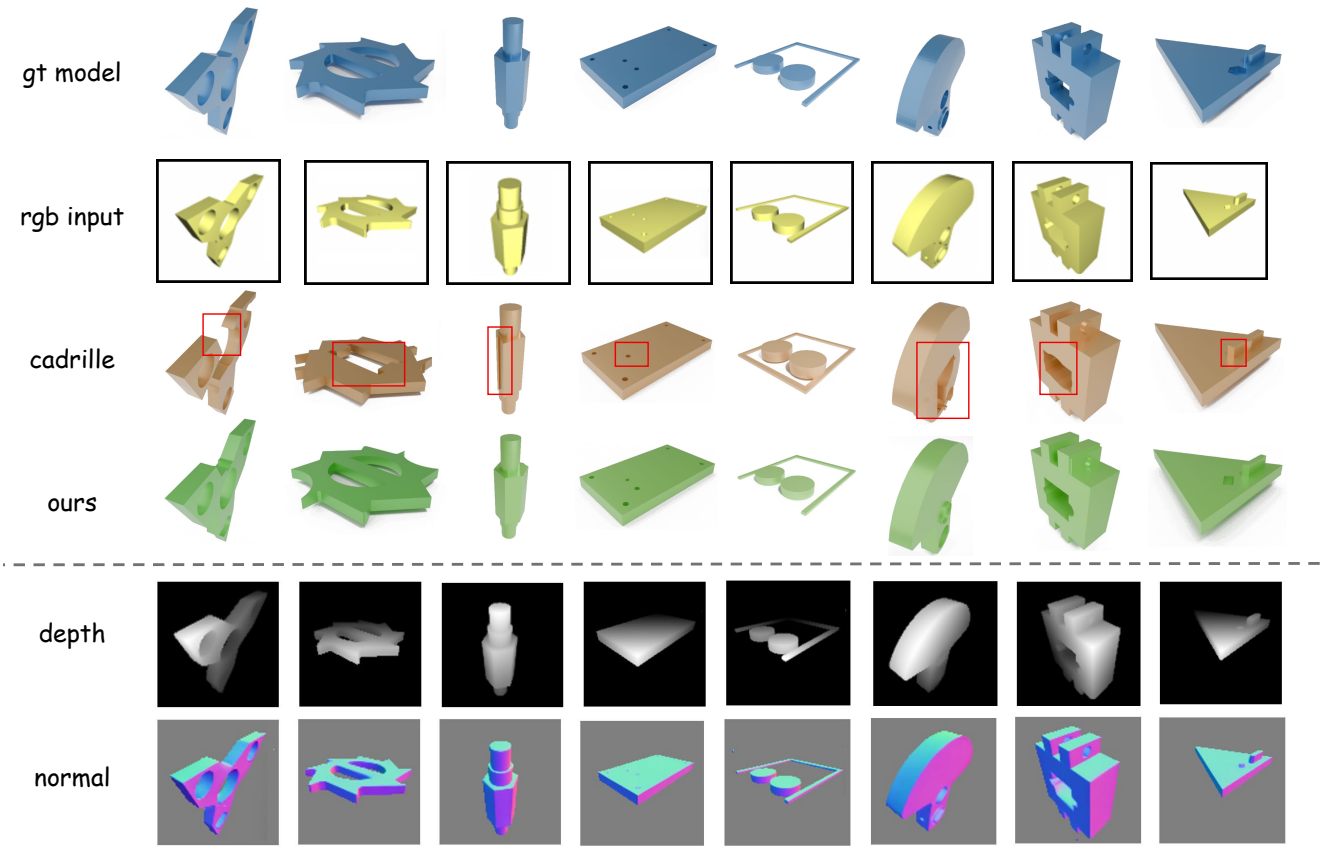

Figure 6: **Qualitative comparison of the generated CAD models**. Depth and normal inputs are only used in our method. The resolution of all input images is 134×134. Our method excels in terms of geometric details.

Table 3: **Ablative evaluation of length reward rule On DeepCAD dataset.** $\text{Ours}_{\text{wo-len}}$: RL without length reward.

Method	IR (%)↓	IoU (%)↑	CD↓	ATL↓
$\text{Ours}_{\text{wo-len}}$	0.66	86.50	0.18	97.5
Ours	0.67	86.49	0.18	93.1

idating the complementarity of multi-modal geometric information.

Length Reward Rule. Table 1 shows that models trained with the length reward rule in RL generate more concise modeling code than those trained with SFT alone, while the cadrille baseline without this mechanism shows no improvement in conciseness. To exclude the influence of different RL algorithms, we trained a control model under the same GSPO framework without length rewards, setting reward weights as λ_{iou}=0.8 and λ_{val}=0.2. As shown in Table 3, this control model shows no improvement in ATL, while other metrics remain comparable to the main model. This indicates that our proposed length reward rule effectively improves the conciseness of generated modeling code with only a marginal impact on geometric accuracy and code validity.

Table 4: **Ablative evaluation of Networks for geometric prior images On DeepCAD dataset.** $\text{Ours}_{\text{3-vits}}$: extra ViT encoders; $\text{Ours}_{\text{3-Projs}}$: extra projection layers; $\text{Ours}_{\text{crossAttn}}$: cross-attention fusion; Ours: shared encoder + concat.

Method	IR (%)↓	IoU (%)↑	CD↓
$\text{Ours}_{\text{3-vits}}$	1.82	72.97	0.43
$\text{Ours}_{\text{3-Projs}}$	1.86	73.51	0.43
$\text{Ours}_{\text{crossAttn}}$	1.93	72.56	0.45
Ours	1.57	75.44	0.33

Processing and Fusion networks for Geometric Priors. Regarding the processing and fusion of geometric prior images, we experimented with three alternative network architectures under the same settings. In $\text{Ours}_{\text{3-vits}}$, depth and normal maps are processed by two additional ViT encoders initialized from the original visual encoder. In $\text{Ours}_{\text{3-Projs}}$,

independent projection modules are introduced for depth and normal features. In $\text{Ours}_{\text{crossAttn}}$, geometric features are fused into RGB features via cross-attention before being fed into the LLM.

However, as shown in Table 4, all three structures result in performance degradation. We argue that the visual encoder of Qwen2VL has been sufficiently pretrained on large-scale image data and already generalizes well to various input types, including geometric images. Introducing insufficiently trained additional modules instead disrupts the original representation capacity of the model. Therefore, we adopt the simplest and most effective strategy—feeding all images (RGB, depth, normal) directly into the same visual encoder and projection layer, and concatenating them before inputting into the LLM.

6. Limitations and Future Work

Although our method demonstrates strong performance in terms of both accuracy and simplicity, several limitations remain. The introduction of geometric priors incurs a small amount of overhead, manifested in three aspects: (1) dense estimation of geometric priors requires additional computation time; (2) a modest increase in the number of tokens during large-model training and inference (25 tokens per image in our implementation); and (3) a corresponding minor increase in GPU memory consumption due to the additional tokens. In practical applications, this overhead is negligible.

In our approach, fixed-style synthetic rendered images are used as training inputs. This design choice may lead to a degree of overfitting to specific input characteristics, and achieving robust reconstruction performance on other image types may require integration with image style transfer techniques. Furthermore, both training and inference rely on normal and depth estimates produced by pretrained models. These estimates are inherently imperfect and contain noise; replacing them with more powerful dense geometric estimation models is a promising direction for improving reconstruction accuracy.

Single-view reconstruction inevitably suffers from missing geometric information in unobserved regions (e.g., the back side of objects). The proposed method is currently unable to fully and reliably recover these regions. Future work may explore incorporating additional constraints—such as axial symmetry and rotational symmetry—into the reward functions of reinforcement learning frameworks to mitigate reconstruction errors caused by such occlusions.

Besides, the current pipeline still falls short of perfectly accurate reconstruction; adopting newer MLLM network architectures, larger MLLMs, or higher-resolution inputs offers a straightforward route to further boost accuracy.

Finally, while the incorporation of geometric priors has been shown to substantially improve the accuracy of single-view reconstruction, an important avenue for future research is extending this idea to multi-view reconstruction tasks, where geometric priors may likewise be leveraged during training to further enhance reconstruction fidelity.

7. Conclusion

We present a systematic post-training study for MLLM in the single-view CAD generation task. Motivated by a theoretical analysis of how geometric priors improve 3D recovery from a single image, we are the first to inject both depth and normal cues into an MLLM-based CAD generation pipeline. Furthermore, we introduce a group-wise length reward during reinforcement learning, which explicitly encourages compact modeling programs and offers a fresh perspective on code quality evaluation. Experiments on multiple benchmarks establish new state-of-the-art results.

8. Acknowledgement

This work was funded by "Pioneer" and "Leading Goose" R & D Program of Zhejiang Province (No. 2025C01086)

References

[1] AU, J. Wright, et al. Cadquery/cadquery: Cadquery 2.4.0, Jan. 2024. 2, 3

[2] S. Bai et al. Qwen2.5-vl technical report, 2025. 3

[3] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan. Learning gradient fields for shape generation, 2020. 1

[4] B. Chen, Z. Xu, S. Kirmani, B. Ichter, D. Driess, P. Florence, D. Sadigh, L. Guibas, and F. Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities, 2024. 2, 3

[5] C. Chen, J. Wei, T. Chen, C. Zhang, X. Yang, S. Zhang, B. Yang, C.-S. Foo, G. Lin, Q. Huang, and F. Liu. Cadcrafter: Generating computer-aided design models from unconstrained images, 2025. 2

[6] A.-C. Cheng, H. Yin, Y. Fu, Q. Guo, R. Yang, J. Kautz, X. Wang, and S. Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *NeurIPS*, 2024. 2, 3

[7] Y. Ganin, S. Bartunov, Y. Li, E. Keller, and S. Saliceti. Computer-aided design as language, 2021. 2

[8] J. He, H. Li, W. Yin, Y. Liang, L. Li, K. Zhou, H. Liu, B. Liu, and Y.-C. Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. *arXiv preprint arXiv:2409.18124*, 2024. 3

[9] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024. 3

[10] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4), July 2023. 1

[11] M. S. Khan, S. Sinha, T. U. Sheikh, D. Stricker, S. A. Ali, and M. Z. Afzal. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 7552–7579. Curran Associates, Inc., 2024. 3

[12] M. Kolodiazhnyi, D. Tarasov, D. Zhemchuzhnikov, A. Nikulin, I. Zisman, A. Vorontsova, A. Konushin, V. Kurenkov, and D. Rukhovich. cadrille: Multi-modal cad reconstruction with online reinforcement learning. *arXiv preprint arXiv:2505.22914*, 2025. 1, 2, 3, 6

[13] X. Li, J. Li, Y. Song, Y. Lou, and X. Zhou. Seek-cad: A self-refined generative modeling for 3d parametric cad using local inference via deepseek, 2025. 1, 2

[14] Z. Lin, M. Lin, Y. Xie, and R. Ji. Cppo: Accelerating the training of group relative policy optimization-based reasoning models, 2025. 3

[15] L. Ling, C.-H. Lin, T.-Y. Lin, Y. Ding, Y. Zeng, Y. Sheng, Y. Ge, M.-Y. Liu, A. Bera, and Z. Li. Scenethesis: A language and vision agentic framework for 3d scene generation, 2025. 1

[16] Y. Liu, M. Ma, X. Yu, P. Ding, H. Zhao, M. Sun, S. Huang, and D. Wang. Ssr: Enhancing depth perception in vision-language models via rationale-guided spatial reasoning. *CoRR*, abs/2505.12448, 2025. 3

[17] Y. Liu, D. Xu, X. Yu, X. Xu, D. Cohen-Or, H. Zhang, and H. Huang. Hola: B-rep generation using a holistic latent representation. *ACM Trans. Graph.*, 44(4), July 2025. 2

[18] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019. 6

[19] W. Ma, S. Chen, Y. Lou, X. Li, and X. Zhou. Draw step by step: Reconstructing cad construction sequences from point clouds via multimodal diffusion. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 27144–27153, 2024. 2

[20] 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, 2020. 1

[21] F. Ocker, S. Menzel, A. Sadik, and T. Rios. From idea to cad: A language model-driven multi-agent system for collaborative design, 2025. 2

[22] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation, 2019. 1, 6

[23] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, and C. Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. 2, 3

[24] D. Rukhovich, E. Dupont, D. Mallis, K. Cherenkova, A. Kacem, and D. Aouada. Cad-recode: Reverse engineering cad code from point clouds. *arXiv preprint arXiv:2412.14042*, 2024. 1, 3, 6

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. 3

[26] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. K. Li, Y. Wu, and D. Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. 2, 3

[27] G. Sheng, C. Zhang, Z. Ye, X. Wu, W. Zhang, R. Zhang, Y. Peng, H. Lin, and C. Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv: 2409.19256*, 2024. 6

[28] J. Sun, B. Zhang, R. Shao, L. Wang, W. Liu, Z. Xie, and Y. Liu. Dreamcraft3d: Hierarchical 3d generation with bootstrapped diffusion prior, 2023. 1

[29] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023. 2

[30] P. Wang et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution, 2024. 3, 6

[31] R. Wang, S. Xu, Y. Dong, Y. Deng, J. Xiang, Z. Lv, G. Sun, X. Tong, and J. Yang. Moge-2: Accurate monocular geometry with metric scale and sharp details, 2025. 3, 5

[32] S. Wang, C. Chen, X. Le, Q. Xu, L. Xu, Y. Zhang, and J. Yang. Cad-gpt: Synthesising cad construction sequence with spatial reasoning-enhanced multimodal llms. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(8):7880–7888, Apr. 2025. 1, 3

[33] K. D. D. Willis, Y. Pu, J. Luo, H. Chu, T. Du, J. G. Lambourne, A. Solar-Lezama, and W. Matusik. Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences, 2021. 6

[34] D. Wu, F. Liu, Y.-H. Hung, and Y. Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. *arXiv preprint arXiv:2505.23747*, 2025. 2, 3

[35] 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 (ICCV)*, pages 6772–6782, October 2021. 2

[36] J. Xiang, Z. Lv, S. Xu, Y. Deng, R. Wang, B. Zhang, D. Chen, X. Tong, and J. Yang. Structured 3d latents for scalable and versatile 3d generation, 2025. 1

[37] J. Xu, C. Wang, Z. Zhao, W. Liu, Y. Ma, and S. Gao. Cad-mllm: Unifying multimodality-conditioned cad generation with mllm, 2024. 1, 3

[38] X. Xu, P. K. Jayaraman, J. G. Lambourne, K. D. Willis, and Y. Furukawa. Hierarchical neural coding for controllable cad model generation. *arXiv:2307.00149*, 2023. 2

[39] X. Xu, K. D. D. Willis, J. G. Lambourne, C.-Y. Cheng, P. K. Jayaraman, and Y. Furukawa. Skexgen: Autoregressive generation of cad construction sequences with disentangled codebooks, 2022. 2

[40] L. Yang, B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao. Depth anything v2. *arXiv:2406.09414*, 2024. 3, 5

[41] Q. Yu et al. Dapo: An open-source llm reinforcement learning system at scale, 2025. 2, 3

[42] Z. Yuan, J. Shi, and Y. Huang. Openecad: An efficient visual language model for editable 3d-cad design. *Comput. Graph.*, 124(C), Nov. 2024. 1, 2

[43] C. Zheng, S. Liu, M. Li, X.-H. Chen, B. Yu, C. Gao, K. Dang, Y. Liu, R. Men, A. Yang, J. Zhou, and J. Lin. Group sequence policy optimization, 2025. 2, 3, 4, 5

[44] J. Zhu et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models, 2025. 3