CADMATE: Generating CAD Assembly Plan with Geometric Chain-of-Thought and Spatial Physical Rewards

Jiali Chen[1,2,3,4], Dingba Fu[1,2,3], Xusen Hei[1,2,3], Yuhang Liu[1,2,3], Yiyang Chen[1], Jiayuan Xie[4], Wenqi Fan[4], Yi Cai[2,1,3]*,

[1]School of Software Engineering, South China University of Technology,
[2]Key Laboratory of Big Data and Intelligent Robot, Ministry of Education,
[3]Joint Guangdong-Hong Kong-Macao Research Laboratory of Big Data and Robotic Intelligence, Ministry of Education,
[4]The Hong Kong Polytechnic University

segarychen@mail.scut.edu.cn gaalik.chen@connect.polyu.hk

Abstract

Computer-aided design (CAD) is crucial in prototyping complex 3D objects through precise geometric modeling. In practical design workflows, designers manually define assembly sequences for individual CAD parts, a process that is both time-consuming and expertise-intensive. To address this challenge, we formulate CAD assembly as a parametric action prediction task: given a reference design image and disassembled parts, the model predicts 6-DoF transformations (*i.e.*, actions) to progressively assemble each part. This paradigm enables multimodal large language models (MLLMs) to solve the task through autoregressive action generation. While recent MLLMs demonstrate promising spatial reasoning, they struggle with fine-grained geometric structure understanding and physical collision avoidance during assembly. In this paper, we propose CADMATE, an MLLM-based framework for sequential CAD assembly action generation. Our training strategy comprises three stages: (i) CAD domain adaptation for spatial geometry and position understanding, (ii) supervised fine-tuning with geometric chain-of-thought (CoT) reasoning for action generation, and (iii) reinforcement learning with spatial-physical rewards jointly optimize spatial accuracy and collision avoidance. Additionally, we also construct *CADBuilder* dataset, comprising over 45*K* CAD assemblies with annotated action sequences. Our experiments demonstrate that CADMATE significantly outperforms existing prominent MLLMs (*e.g.*, GPT-5), showing great potential in design applications[1].

1 Introduction

Computer-aided design (CAD) plays a crucial role in industrial design and manufacturing, serving as the fundamental tool for precise 3D product prototyping (Wu et al., 2021; Chen et al., 2025a; Doris et al., 2025; Mallis et al., 2024). In practical CAD workflows, the creation of complex products necessitates not only the meticulous individual component design but also the strategic planning of their assembly—a process that remains labor-intensive and expertise-dependent. Despite the proliferation of advanced CAD software (*e.g.*, AutoCAD, SolidWorks and Blender), designers must manually determine the optimal assembly sequence and position each part consistent with the reference design with multiple attempts, as shown in Fig. 1 (a).

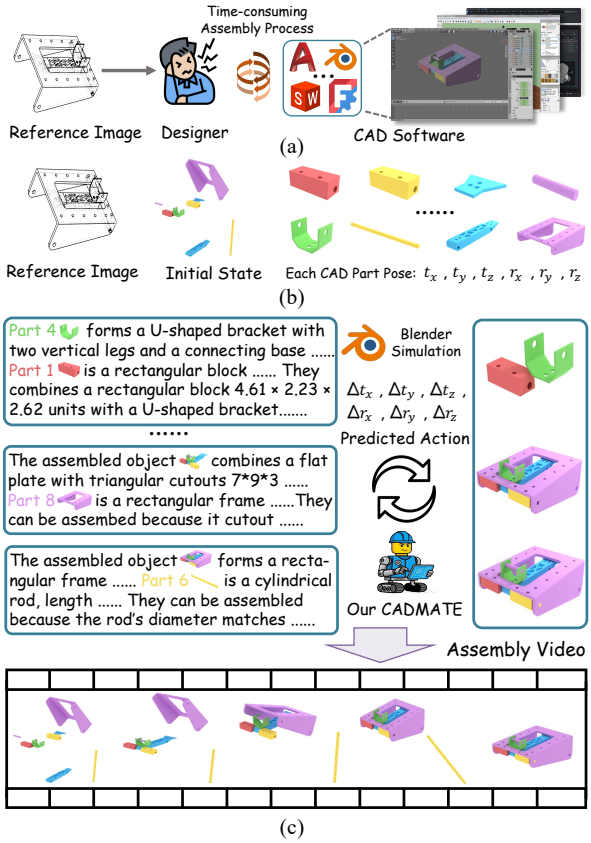

Figure 1: Overview of different ways (*i.e.*, designer and our CADMATE) to CAD assembly. (b) Input for CADMATE. (c) multi-turn interactive assembly process with geometric chain-of-thought.

Towards this end, several automatic CAD assem-

*Corresponding author: ycai@scut.edu.cn
[1]Project page: https://cgl-pro.github.io/cadmate

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 18324–18348
July 2-7, 2026

bly approaches have been proposed to streamline the process. Most approaches focus on reversing disassembly simulations (Tian et al., 2022, 2024) or predicting only final CAD part configurations (Li et al., 2024b; Zhang et al., 2025a), but they limit by reliance on complete objects or cannot be physically realized due to neglect of the assembly process. To overcome these limitations, we formulate CAD assembly as a sequential action prediction task, where the multimodal large language model (MLLMs) generates step-by-step 6-DoF transformations to progressively assemble each part. It naturally aligns with the autoregressive paradigm of MLLMs, enabling end-to-end learning of assembly actions from reference design.

However, generating assembly actions for individual CAD parts with MLLMs remains nontrivial challenges. **First**, CAD parts exhibit complex geometric attributes (*e.g.*, shapes and dimensions) and intricate spatial constraints governing how they fit together. Existing MLLMs lack fine-grained perception of such CAD-specific geometric features. **Second**, while MLLMs have explored spatial reasoning in natural scenes (Yang et al., 2025; Deng et al., 2025), CAD assembly requires multi-turn interactive spatial reasoning to plan sequential transformations, where the next action depends on the current assembly state. As shown in Fig. 1 (c), the MLLM observes images with the updated state after each step to plan subsequent actions. **Third**, beyond geometric accuracy, the predicted assembly actions should be physically feasible, *i.e.*, each assembly step requires planning a collision-free trajectory for a specific CAD part, to meet the realistic industrial requirement (Boothroyd, 1987; Grieves and Vickers, 2017; Pun et al., 2025). Existing MLLMs lack explicit mechanisms to ensure such physical constraints.

To address these challenges, we develop CADMATE, an MLLM-based framework that interprets CAD assembly as sequential action generation. CADMATE progressively builds spatial perception, geometric reasoning, and physical awareness through a three-stage training strategy. We begin by performing CAD domain adaptation to refine the MLLM's spatial perception with CAD captioning and pose prediction tasks. We then conduct supervised fine-tuning (SFT) with geometric chain-of-thought (CoT), enabling explicit reasoning over part attributes and assembly constraints in a multi-turn interactive manner, *i.e.*, each turn generates actions for one selected part. To ensure assembled CAD objects are both geometrically accurate and physically feasible, we further apply reinforcement learning with spatial-physical rewards to jointly optimize precision and collision avoidance. To support CADMATE, we construct *CADBuilder*, a large-scale dataset comprising over 45K CAD assemblies with annotated assembly action sequences. *CADBuilder* includes CAD objects with 2$\sim$15 parts across diverse structural complexities, serving as a comprehensive benchmark for the CAD assembly task with MLLMs. We summarize our key contributions as follows: **(i)** We formulate CAD assembly as a parametric action generation task that naturally aligns with the autoregressive generation paradigm of MLLMs, enabling end-to-end learning of 6-DoF pose transformation for each CAD part. **(ii)** We propose CADMATE with a three-stage training strategy that combines CAD domain adaptation, supervised fine-tuning, and reinforcement learning. It progressively builds spatial perception, multi-turn interactive geometric chain-of-thought reasoning, and spatial-physical awareness. **(iii)** We construct *CADBuilder*, a dataset consisting of 45K CAD assemblies with annotated action sequences. Extensive experiments show that CADMATE consistently outperforms strong MLLMs across assembly complexities.

2 Related Work

CAD Assembly. Previous CAD assembly studies primarily utilize heuristic rules and physics-based simulation to derive assembly sequences. Specifically, JoinABLe (Willis et al., 2022) utilizes a graph representation to predict joint connections between CAD part pairs, which fails to apply in the realistic multi-part assembly scenarios. Tian et al. (2022, 2024) infers assembly sequences by reversing simulated disassembly motions with heuristic rules. Recent works (Zhang et al., 2025a; Tie et al., 2025) leverage 3D point clouds of CAD parts to learn detailed geometric features for assembly.

MLLMs in Spatial Reasoning. Multimodal large language models (MLLMs) have advanced through evolving reasoning paradigms (Bai et al., 2025; Wu et al., 2026). Several benchmarks (Yang et al., 2025; Li et al., 2025; Cai et al., 2025) are proposed to evaluate their spatial reasoning ability. Beyond evaluation, other efforts focus on advancing through perception enhancement approaches (Wu et al., 2025a; Daxberger et al., 2025; Huang et al., 2025). However, they are limited to natural

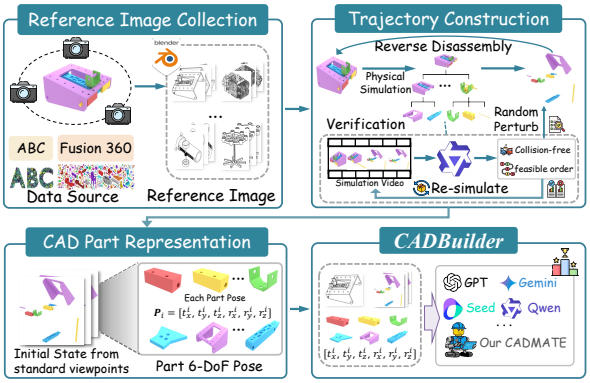

Figure 2: Construction workflow of our *CADBuilder*, including reference image collection, assembly trajectory construction and CAD parts representation.

scenes, whereas CAD assembly requires geometric understanding and precise transformations.

3 *CADBuilder* Dataset

We introduce *CADBuilder*, a parametric CAD assembly dataset that contains over 45*K* unique CAD objects from Fusion 360 Gallery (Willis et al., 2021) and ABC (Koch et al., 2019) datasets. It contains a wide range of assembly complexities, with the number of CAD parts per object varying from 2 to 15. Each sample in *CADBuilder* consists of a reference design image (*i.e.*, target assembly) and disassembled CAD parts with corresponding initial 6 degrees of freedom (6-DoF) poses. Our task requires multimodal large language models (MLLMs) to predict geometric transformations (*i.e.*, actions) for each distinct part, reconstructing the complete CAD object that matches the reference. In the following, we describe construction process, as shown in Fig. 2. More details are provided in Appendix A.

3.1 CAD Parts Representation

We formulate the CAD assembly task as predicting parametric actions of disassembled parts under a unified coordinate system. Specifically, a CAD object $\mathcal{O} = \{p_1, \ldots, p_N\}$ contains N parts, each is represented by 6-DoF pose $\mathbf{p}_i = [t_x^i, t_y^i, t_z^i, r_x^i, r_y^i, r_z^i]$. The translation components (t_x^i, t_y^i, t_z^i) define the part's position, while the rotation components (r_x^i, r_y^i, r_z^i) specify its orientation in Euler angles. During the assembly process, part p_1 serves as the fixed base. For each remaining part p_i $(i \geq 2)$, we define an assembly action as a relative transformation $\Delta\mathbf{p}_i = [\Delta t_x^i, \Delta t_y^i, \Delta t_z^i, \Delta r_x^i, \Delta r_y^i, \Delta r_z^i]$ that progressively move the specific part from the initial disassembled pose to the final assembled position. This parametric formulation provides a compact spatial representation of individual CAD parts, enabling MLLMs to directly generate these actions through autoregression.

3.2 Reference Image Collection

To ensure consistent size scales across objects, we first place each collected CAD object to the $(0, 0, 0)$ coordinate and uniformly scale it where its bounding box fits within a $10 \times 10 \times 10$ cube, with the maximum dimension reaching 10 units. To obtain guidance for the assembly task, we utilize Blender to render reference images of the CAD object. Specifically, candidate rendering viewpoints are sampled from both the upper and lower hemispheres around the object, with azimuth angles in $[0, 2\pi)$ at an interval of $\pi/10$. An automatic filtering is then applied to remove suboptimal views, followed by manually selecting a representative view that best depicts the CAD object. In particular, the perspective view rendering is utilized to preserve internal components. More details can be found in the Appendix A.1.

3.3 Assembly Trajectory Construction

Following previous research (Tian et al., 2022, 2024), we obtain plausible assembly trajectories by reversing the disassembly simulation. Specifically, we perform the simulation (Tian et al., 2022) to generate a disassembly trajectory for each part under gravity and contact constraints. All trajectories are rendered into videos to visualize the process using Blender[2]. To ensure the physical and geometric validity, we employ Qwen3-VL-235B-A22B-Instruct (Bai et al., 2025) to verify the video for physical plausibility, focusing on inter-part collisions and the feasibility of assembly order. We re-simulate the trajectory when it fails verification, and samples that fail to meet the verification criteria after five re-simulation attempts are discarded. The detailed prompt for Qwen3-VL is provided in Fig. 7 of Appendix A.3. To meet the realistic industrial requirements, we also perturb the positions of disassembled parts to obtain their corresponding initial 6-DoF poses $\{\mathbf{p}_i\}_{i=1}^N$, where N is the number of CAD parts. The assembly trajectories are obtained by reversing the simulated disassembly ones. The action $\Delta\mathbf{p}_i$ is identified from key pose variations

[2]https://www.blender.org

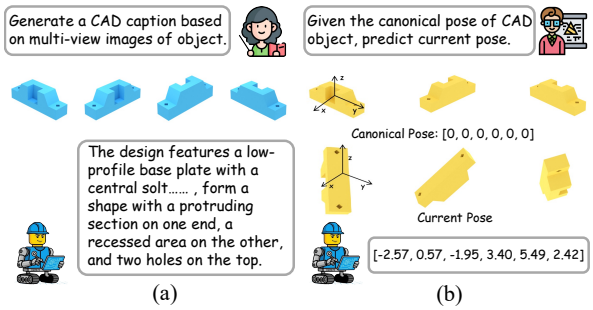

Figure 3: Overview of spatial perception adaptation. (a) CAD Caption. (b) 6-DoF Pose Prediction.

along the trajectory of part p_i. Thus, the trajectory of part p_i can be represented by continuous M actions $\Delta\mathbf{P}_i = [\Delta\mathbf{p}_i^1, \dots \Delta\mathbf{p}_i^M]$.

4 CADMATE Model

Given a reference design image and the initial disassembled state, our goal is to generate assembly actions for each movable CAD part. To this end, we build CADMATE by fine-tuning a pre-trained multimodal large language model (MLLM), with a three-stage training strategy: spatial perception adaptation to capture spatial features in the CAD domain, supervised fine-tuning (SFT) to perform interactive multi-turn action generation, followed by reinforcement learning (RL) improve further geometric precision and avoid collisions. To standardize the visual perspectives during training, we use images from 3 fixed standard camera viewpoints[3] (*i.e.*, **standard viewpoints**) to represent the current assembled object state. It ensures the model learns spatial information under uniform conditions. We further explore different model scales, *i.e.*, CADMATE-4B/8B based on the Qwen3-VL-Instruct. The overview of CADMATE is shown in Fig. 4.

4.1 Spatial Perception Adaptation

We first enhance CADMATE to capture geometric structures and spatial relationships within the CAD domain. Specifically, we collect 320*K* CAD caption and 6-DoF poses data for this adaptation stage, as shown in Fig. 3 More details about the data processing are in Appendix B.

CAD Caption. In contrast to traditional image caption, CAD captioning focuses on textually describing the individual components that constitute a 3D object rather than the overall object itself. In this task, we utilize images and beginner-level textual CAD modeling instructions as image-text pairs from the Text2CAD (Khan et al., 2024) dataset. The instruction primarily includes shape properties (Fig. 3 (a)), *e.g.*, a rectangular CAD object with a central hole, guiding the MLLM to visually recognize its geometric features.

6-DoF Pose Prediction. To facilitate accurate assembly, we design a 6-DoF pose prediction task, which involves determining both the position and orientation of the CAD model within a unified coordinate system, as illustrated in Fig. 3 (b). It aims to predict the 6-DoF pose of the entire CAD object. To build the training data, we collect over 160*K* 3D objects from the DeepCAD dataset (Wu et al., 2021). Next, we apply random pose perturbations to these objects and obtain corresponding 6-DoF poses. We train the MLLMs to predict the pose based on rendered images of canonical[4] and perturbed poses at standard viewpoints.

4.2 SFT with Geometric Chain-of-Thought

Building upon the spatial perception from the adaptation stage, we conduct supervised fine-tuning (SFT) to equip the MLLM with the capability to generate assembly actions. It requires the MLLM to understand the geometric attributes (*e.g.*, size, shape) and account for assembly constraints that define how parts fit together. Therefore, we train on *CADBuilder* with additional geometric chain-of-thought CoT for the assembly part. Moreover, we adopt a multi-turn interactive training scheme, enabling the model to understand the progressive assembly state. Specifically, CADMATE generates actions (*i.e.*, trajectory) for a single CAD part at each turn. The updated assembly state is rendered as multi-view images for the next turn generation.

Geometric CoT Annotation. We augment each sample from *CADBuilder* with geometric CoT, including (i) shape descriptions of each part and the current assembled object; (ii) constraints that define how they can be assembled. For shape descriptions, we first use Blender to render each part and the current assembled object and extract their geometric attributes (*e.g.*, size, number of holes) into JSON format (Khan et al., 2024). Inspired by (Khan et al., 2024; Chen et al., 2025b; Wang et al., 2024; Luo et al., 2023), we feed these images and properties into GPT-5 (OpenAI, 2025) to generate their shape descriptions, such as "a hollow cylindrical tube with an outer diameter of 1.46 units …".

[3] At 45°, 135°, 315° with a radius of 32 and height of 15.

[4] All six degrees of freedom set to zero (Ma et al., 2024).

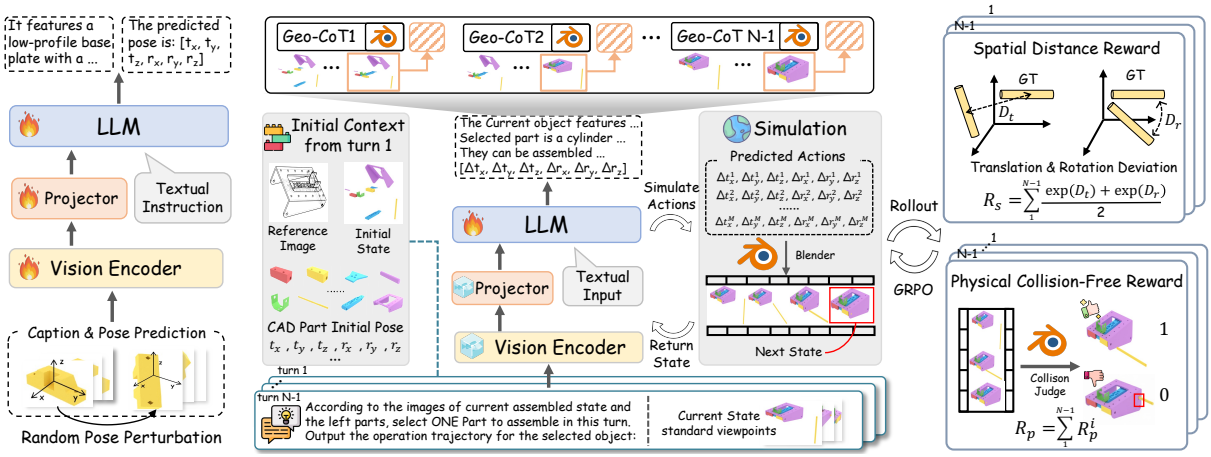

Figure 4: Overview of CADMATE. **Left**: spatial perception adaptation. **Middle**: multi-turn interactive supervised fine-tuning with geometric chain-of-thought. **Right**: reinforcement learning with spatial-physical rewards.

Next, we use the ground truth assembly video of the current turn as a reference. The video, along with shape descriptions and their corresponding images, prompts GPT-5 to annotate the compatibility constraints of the current assembly step, *e.g.*, "the hollow cylindrical shape aligns precisely with the open top of the cylindrical tube ...". To ensure the reliability of our geometric CoT annotations, we further verify them in Appendix A.4.1.

Training. Each training sample is structured as a multi-turn sequential assembly. At the first turn, the input includes: a reference image of the target assembled design, multi-view images of the initial disassembled state from standard viewpoints, and individual states (*i.e.*, image and pose) of each part. At subsequent turns, the model receives updated multi-view images about the current partially assembled state. For each turn, the model generates geometric chain-of-thought reasoning (*i.e.*, shape descriptions and assembly constraints) within `<description>...</description>` and `<constraint>...</constraint>` tags, and corresponding assembly actions $\Delta\mathbf{P}_i$ of selected part p_i (Fig. 4). We minimize the standard autoregressive negative log-likelihood:

$$\mathcal{L}_{\text{SFT}} = -\sum_{i=2}^{N} \log p_\theta(\mathbf{y}_i \mid \mathbf{x}_i), \tag{1}$$

where N denotes the total number of parts in the CAD object, $\mathbf{x}_i$ and $\mathbf{y}_i$ denote the input and output for assembling part p_i, respectively. Since part p_1 serves as the fixed base, the process involves $N-1$ sequential turns to assemble the remaining parts.

4.3 RL via Spatial-Physical Rewards

After the SFT stage, our CADMATE shows promising capabilities in generating assembly actions with geometric reasoning. However, the predicted trajectories (*i.e.*, action sequences) lack geometric precision compared to the target object, and the model struggles with collision-free assembly planning in complex multi-part scenarios. To address these limitations, we propose spatial-physical rewards for the reinforcement learning stage. Specifically, drawing inspiration from Group Relative Policy Optimization (GRPO), we extend beyond traditional textual assessment to evaluate the 3D assembly process, which optimizes spatial alignment accuracy and physical collision avoidance.

Spatial Distance Reward $\mathcal{R}_s$. To efficiently evaluate the assembly accuracy from generated actions, we compute the 6-DoF pose deviation between each predicted part and its ground-truth configuration. Specifically, after executing all predicted actions, we obtain the final pose for part p_i. We then compute its' translation and rotation deviations (*i.e.*, D_t^i and D_r^i):

$$\begin{aligned} D_t^i &= \sqrt{(\delta t_x^i)^2 + (\delta t_y^i)^2 + (\delta t_z^i)^2}, \\ D_r^i &= \sqrt{(\delta r_x^i)^2 + (\delta r_y^i)^2 + (\delta r_z^i)^2}, \end{aligned} \tag{2}$$

where $(\delta t_x^i, \delta t_y^i, \delta t_z^i)$ is the axial translation deviation from ground-truth and $(\delta r_x^i, \delta r_y^i, \delta r_z^i)$ denotes the axial rotation deviation in radians. Since translation and rotation have different units, we normalize them separately into reward signals ($\mathcal{R}_t^i$ and $\mathcal{R}_r^i$)

using exponential decay:

$$\mathcal{R}_t^i = \exp\left(-\frac{D_t^i}{\tau_t}\right), \quad \mathcal{R}_r^i = \exp\left(-\frac{D_r^i}{\tau_r}\right), \tag{3}$$

where τ_t and τ_r are scale parameters controlling the decay rate of rewards. We set $\tau_t = 0.5$ and $\tau_r = \pi/18$, which provide appropriate reward gradients for assembly tolerances. The final spatial reward averages the rewards across all movable parts:

$$\mathcal{R}_s = \frac{1}{N-1}\sum_{i=2}^{N}\frac{\mathcal{R}_t^i + \mathcal{R}_r^i}{2}. \tag{4}$$

Physical Collision-Free Reward $\mathcal{R}_p$. Although $\mathcal{R}_s$ ensures the part-level spatial accuracy, practical assembly requires CAD parts do not collide with each other during the assembly process. Therefore, we verify the collision feasibility based on the predicted assembly simulation. Specifically, for each assembly turn $i \in \{2, \ldots, N\}$, we execute the predicted action sequence $\Delta\mathbf{P}_i$ and perform mesh-based intersection tests between the moving part p_i and other CAD components. A turn is considered collision-free if no geometric intersection occurs throughout the trajectory. We define the collision-free reward $\mathcal{R}_p^i$ for each turn and the final physical collision-free reward $\mathcal{R}_p$ is also computed as the average across all assembly turns:

$$\begin{aligned}\mathcal{R}_p^i &= \begin{cases}1, & \text{if no collision detected}\\ 0, & \text{otherwise}\end{cases},\\ \mathcal{R}_p &= \frac{1}{N-1}\sum_{i=2}^{N}\mathcal{R}_p^i.\end{aligned} \tag{5}$$

Training with GRPO. We adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to train CADMATE with the spatial-physical rewards. Before RL training, for each training sample, we first use CADMATE after SFT to generate $K = 8$ complete assembly candidates through rollout sampling. We then evaluate each candidate by its spatial deviations (D_i^t and D_i^r, Eq. 2) from ground truth. A candidate is considered positive if both $D_i^t \le 0.5$ and $D_i^r \le \pi/18$ hold for all parts p_i, representing acceptable assembly tolerances; otherwise, it is labeled negative. Inpired by previous research (Wu et al., 2025b; An et al., 2025), we retain samples achieving pass ratio (*i.e.*, pass@k) criterion with $k \in \{1, 2, \ldots, 7\}$ to provide appropriate challenging data for RL training. Subsequently, we adopt GRPO to refine CADMATE with spatial-physical rewards. For data preparation, we use the SFT checkpoint to generate 8 complete assembly candidates for each training sample via rollout. Each candidate is then evaluated by computing its spatial deviations (D_i^t and D_i^r from Eq. 2) against ground truth. A candidate is labeled positive if all parts satisfy both $D_i^t \le 0.5$ and $D_i^r \le \pi/18$; otherwise negative. Following (An et al., 2025), we retain samples achieving pass@k with $k \in \{1, \ldots, 7\}$ to ensure appropriate training difficulty. During RL training, we also utilize the format reward $\mathcal{R}_f$ to force the model to generate required tag format of <description> and <constraint>. Finally, we compute the overall reward $\mathcal{R}$ as a weighted combination of $\mathcal{R}_s$, $\mathcal{R}_p$ and $\mathcal{R}_f$:

$$\mathcal{R} = \lambda_s\mathcal{R}_s + \lambda_c\mathcal{R}_p + \lambda_f\mathcal{R}_f. \tag{6}$$

$\mathcal{R}$ is used to compute the advantage $A_j = \frac{(\mathcal{R}_j - \mu)}{\sigma}$ for the j-th rollout, where μ and σ are the mean and standard deviation over all K rollouts. The model is optimized via the GRPO objective:

$$\begin{aligned}\mathcal{L}_{\text{RL}} = \mathbb{E}\Bigg[\frac{1}{G}\sum_{j=1}^{G}\Big(\min(r_j(\theta)A_j, \text{clip}(r_j(\theta),\\ 1-\varepsilon, 1+\varepsilon)A_j\Big) - \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})\Bigg],\end{aligned} \tag{7}$$

where G is the group size, and $r_j(\theta) = \frac{\pi_\theta(o_j|x)}{\pi_{\theta_{\text{old}}}(o_j|x)}$ is the probability ratio between the current policy π_θ and the sampling policy $\pi_{\theta_{\text{old}}}$. ϵ denotes the clipping range used to constrain policy updates. To prevent excessive deviation from the initial capabilities, a KL divergence penalty is applied relative to the frozen SFT reference policy π_{ref}. This mechanism allows CADMATE to iteratively refine its assembly planning toward spatially precise and collision-free solutions.

5 Experiments

5.1 Experiment Setting

Implementation Details. All experiments are conducted with 16 NVIDIA H100-80GB GPUs. We build CADMATE by fine-tuning Qwen3-VL-Instruct (Bai et al., 2025) with two different model scales: 4B and 8B parameters, referred to as CADMATE-4B and CADMATE-8B, respectively. For the spatial perception adaptation (SPA) stage, we perform full fine-tuning on both the vision encoder and LLM for 3 epochs with a global batch size of 64

Model	2 ∼ 5 parts				5 ∼ 10 parts				10 ∼ 15 parts			
	MMD↓	CD↓	PA↑	S_a↑	MMD↓	CD↓	PA↑	S_a↑	MMD↓	CD↓	PA↑	S_a↑
Few-shot												
GLM-4.5V	6.44	23.95	50.62	2.85	12.11	35.52	25.38	2.21	34.86	52.50	21.44	1.76
Qwen3-VL-Plus	6.11	21.53	52.85	2.98	11.12	30.36	28.47	2.35	38.47	47.45	22.84	1.89
Seed-1.6-Vision	6.73	24.97	51.55	2.76	12.89	36.23	25.25	2.18	32.74	41.22	22.25	1.72
Gemini-2.5-Flash	6.76	22.21	56.46	3.12	11.01	33.23	29.34	2.48	27.55	36.21	21.32	1.95
Gemini-2.5-Pro	6.31	21.45	57.96	3.25	11.13	30.47	30.42	2.56	31.00	39.15	24.12	2.08
GPT-5-mini	6.70	23.92	51.52	3.08	11.52	32.39	26.71	2.42	26.87	36.95	20.65	1.92
GPT-5	5.97	23.27	55.18	3.18	11.95	33.59	27.62	2.51	28.06	38.53	20.71	2.01
Fine-tuning												
Qwen3-VL-8B	4.14	1.81	77.12	4.85	6.52	5.74	51.03	4.32	7.57	7.49	45.58	3.78
InternVL3.5-8B	4.28	2.03	74.56	4.68	6.89	6.21	47.89	4.15	8.12	8.35	42.34	3.56
GLM-4.1V-9B	4.45	2.24	72.31	4.52	7.15	6.58	49.76	3.98	8.56	8.92	45.12	3.42
Manul-PA†	1.97	2.47	85.09	-	5.45	5.10	56.81	-	**3.31**	4.10	49.52	-
CADMATE-4B w/o SPA	2.08	2.15	84.56	4.85	4.65	5.38	58.72	4.52	5.92	4.45	46.58	3.68
CADMATE-4B w/o CoT	3.78	3.04	78.85	4.72	5.85	5.62	53.42	4.18	6.95	6.58	47.25	3.62
CADMATE-4B w/o $\mathcal{R}_s$	2.52	2.65	82.18	4.82	4.95	5.25	56.85	4.58	5.68	4.65	48.95	3.78
CADMATE-4B w/o $\mathcal{R}_p$	2.25	2.35	84.25	4.52	4.42	4.88	60.75	4.28	5.15	3.85	51.35	3.45
CADMATE-4B	1.70	1.77	87.42	5.00	3.77	4.49	66.58	4.96	4.26	2.77	54.82	4.12
CADMATE-8B w/o SPA	1.85	1.98	87.42	4.98	3.92	4.78	68.25	4.65	5.25	3.65	56.45	3.92
CADMATE-8B w/o CoT	3.28	2.08	82.65	4.85	5.02	5.18	62.35	4.38	6.18	5.92	55.28	3.88
CADMATE-8B w/o $\mathcal{R}_s$	1.71	2.36	88.17	4.92	3.18	4.30	73.77	4.88	3.96	2.15	64.76	4.52
CADMATE-8B w/o $\mathcal{R}_p$	1.88	2.18	88.72	4.68	3.52	4.58	72.45	4.48	4.32	2.52	62.58	3.82
CADMATE-8B	**1.48**	**1.62**	**90.35**	**5.15**	**2.82**	**3.85**	**76.92**	**5.08**	3.45	**1.78**	**68.56**	**4.75**

Table 1: Evaluation results of different models on *CADBuilder*. † denotes model without considering initial part poses (Appendix D.3). The best results are shown in **bold** and the second best results are underlined.

and a learning rate of 2e-4, using the AdamW optimizer (Loshchilov et al., 2017). For the SFT stage, we freeze the vision encoder and fully fine-tune the LLM for 4 epochs. The learning rate is set to 1e-5, with a cosine decay schedule with a 3% warmup ratio. The maximum sequence length is 16,196 tokens. We utilize 28K samples for SFT cold-start to initialize the policy, preserving the remaining 15K to facilitate effective exploration for the subsequent RL stage. For the RL stage, the model is initialized with SFT weights and trained for 1 epoch with a fixed learning rate of 5e-6. During GRPO training, we obtain 10K training samples through pass@k criteria. Each input generates $G = 8$ candidate completions with with sampling temperature of 0.95 and top-p value 0.9. The KL divergence coefficient is set to $\beta = 0.02$. The hyperparameters of the overall reward in Eq. 6 are 0.5, 0.4, 0.1. All geometric computations, including action execution, image rendering, and collision detection are performed with Blender.

Evaluation Metrics. Following previous studies on 3D reconstruction (Wu et al., 2021; Zhang et al., 2025a), we adopt Minimum Matching Distance (MMD), Mean Chamfer Distance (CD) and Part Accuracy (PA) to evaluate fidelity between 3D point clouds of the assembled and ground truth CAD object. Beyond static 3D shape accuracy, we further assess assembly quality through video-based evaluation in three aspects: motion collision, assembly gaps, and structural coherence. Each aspect is scored by Gemini-2.5-Flash on a scale of 0 to 2, where higher scores indicate better quality. The overall assembly score S_a is computed as the sum of these three scores. The calculation process of these metrics is in Appendix D.1.

Baselines. We evaluate CADMATE against three kinds of baselines: (i) 7 large-scale leading MLLMs, including GPT-5/5-mini (OpenAI, 2025), Gemini-2.5-Pro/Flash (Comanici et al., 2025), Doubao-Seed-1.6-vision (Guo et al., 2025) and Qwen3-VL-Plus (Bai et al., 2025) and GLM-4.5V (Hong et al., 2025); (ii) full fine-tuning 3 small open-source reasoning MLLMs, *i.e.*, GLM-4.1V-Thinking (Hong et al., 2025), InternVL3.5-8B (Wang et al., 2025) and Qwen3-VL-8B-Thinking (Bai et al., 2025); (iii) training Manul-PA (Zhang et al., 2025a), 3D point cloud based model, on our *CADBuilder* dataset. Details are in Appendix D.3.

5.2 Performance Comparison

Table 1 reports results on automatic metrics. We have the following findings: **(i) Proprietary MLLMs exhibit limited spatial reasoning for CAD assembly.** Despite their remarkable reasoning ability, their performance on CAD assembly task remains notably limited under the few-shot setting. On the 2∼5 parts subset, GPT-5 and Gemini-2.5-Pro achieve CD of 23.27 and 21.45 with PA below 58%, whereas CADMATE-8B attains substantially better 1.62 CD and 90.35% PA. This gap indicates that these leading MLLMs struggle to understand the CAD geometric structures and 6-DoF spatial transformations. **(ii) Fine-tuning alone is insufficient to transfer of general reasoning to geometric reasoning abilities.** Fine-tuning on the *CADBuilder* dataset leads to notable performance gains for open-source models. Nevertheless, on the most challenging 10∼15 parts subset, fine-tuned Qwen3-VL-8B lags behind CADMATE-4B across all metrics (*e.g.*, CD: 7.49 vs. 2.77, PA: 45.58% vs. 54.82%). This suggests that standard fine-tuning on reasoning MLLMs cannot effectively transfer their abilities to this geometric reasoning scenarios. Notably, Manual-PA still underperforms CADMATE, even it simplifies the task by directly predicting final poses without considering actions and initial poses. **(iii) CADMATE exhibits robust geometric reasoning across varying assembly complexities.** As assembly complexity increases, CADMATE maintains more stable performance compared to baselines. From the 2∼5 to 10∼15 parts subsets, Qwen3-VL-8B's PA drops by 40.90% (from 77.12% to 45.58%), whereas CADMATE-8B shows a smaller 24.11% decline (from 90.35% to 68.56%). On the challenging 10∼15 parts subset, CADMATE-8B still achieves 68.56% PA and 1.78 CD, substantially outperforming all baselines. It benefits from our three-stage training strategy that progressively builds spatial perception, geometric reasoning, and physical awareness. **(iv) Physical feasibility remains a major challenge for existing models.** Both proprietary and fine-tuned open-source models exhibit limited performance on the S_a metric. While our spatial-physical rewards alleviate this issue, CADMATE-8B still achieves unsatisfactory 4.75/6.0 on the 10∼15 parts subset.

5.3 Ablation Study

Table 1 also presents ablation results. We observe: (i) Removing spatial perception adaptation (SPA), CADMATE-8B increases CD from 1.78 to 3.65 in the 10∼15 parts subset, confirming its role in learning fine-grained part-level spatial awareness in CAD domain. (ii) CoT causes the substantial improvement on the 8B variant (CD: from 5.92 to 1.78), indicating that geometric reasoning is essential for capturing assembly constraints. (iii) For our spatial-physical rewards, $\mathcal{R}_s$ is important for spatial accuracy (*e.g.*, CD), while $\mathcal{R}_p$ primarily affects S_a, validating that they target intended objective.

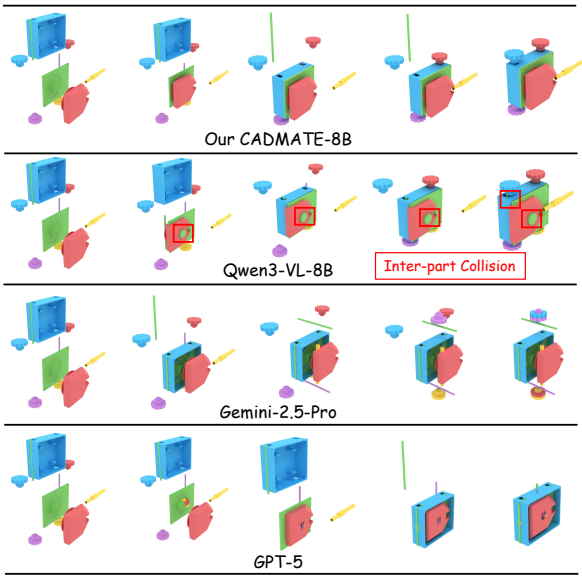

Figure 5: Key assembly steps from different methods.

5.4 Qualitative Analysis

Fig. 5 showcases assembly sequences from various models. CADMATE generates physically plausible trajectories with precise 6-DoF alignment and logical ordering. In contrast, although fine-tuned Qwen3-VL-8B reconstructs the general shape, it suffers from inter-part collisions (red boxes). Moreover, leading proprietary MLLMs like Gemini-2.5-Pro and GPT-5 fail fundamentally in spatial reasoning, resulting in disconnected and misplaced components. More cases with generated geometric CoT are provided in Appendix E.

6 Conclusion

In this paper, we formulate CAD assembly as a sequential action prediction task, where the goal is to generate 6-DoF transformations to progressively assemble individual parts based on reference design images. To address this task, we construct *CADBuilder* dataset and propose CADMATE model with a three-stage training strategy that progressively builds spatial perception, geometric reason-

ing via chain-of-thought, and physical awareness through spatial-physical rewards. Extensive experiments demonstrate that CADMATE outperforms existing MLLMs, providing a strong potential for the automated CAD assembly planning in industrial design applications.

Limitations

In this paper, we focus on image-guided CAD assembly, where the reference design is provided with an image. In other scenarios, designers may specify assembly requirements through textual descriptions. We believe that our method could be extended to these contexts with sufficient text-guided assembly data. Additionally, due to the limitations about availability of sufficient complex CAD assemblies, our current *CADBuilder* dataset contains objects with 2$\sim$15 disassembled parts. In the future, we plan to expand data with greater structural complexity, which offers challenges for advancing spatial understanding and physical feasibility.

Acknowledgments

This research is supported by the Guangdong Provincial Natural Science Foundation for Outstanding Youth Team Project (2024B1515040010), the Guangdong Provincial Fund for Basic and Applied Basic Research—Regional Joint Fund Project (Key Project) (2023B1515120078), National Natural Science Foundation of China (62476097), the Science and Technology Planning Project of Guangdong Province (2025B0101120003), the Fundamental Research Funds for the Central Universities, South China University of Technology (x2rjD2250190).

References

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and 1 others. 2025. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. *URL https://hkunlp. github. io/blog/2025/Polaris*.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-vl technical report.

Geoffrey Boothroyd. 1987. Design for assembly—the key to design for manufacture. *The International Journal of Advanced Manufacturing Technology*, 2(3):3–11.

Zhongang Cai, Yubo Wang, Qingping Sun, Ruisi Wang, Chenyang Gu, Wanqi Yin, Zhiqian Lin, Zhitao Yang, Chen Wei, Oscar Qian, and 1 others. 2025. Holistic evaluation of multimodal llms on spatial intelligence. *arXiv preprint arXiv:2508.13142*.

Jiali Chen, Xusen Hei, Hongfei Liu, Yuancheng Wei, Zikun Deng, Jiayuan Xie, Yi Cai, and Qing Li. 2025a. Cadreview: Automatically reviewing CAD programs with error detection and correction. In *Proc. of ACL*, pages 9909–9927. Association for Computational Linguistics.

Yanjun Chen, Yirong Sun, Xinghao Chen, Jian Wang, Xiaoyu Shen, Wenjie Li, and Wei Zhang. 2025b. Integrating chain-of-thought for multimodal alignment: A study on 3d vision-language learning. *CoRR*, abs/2503.06232.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Erik Daxberger, Nina Wenzel, David Griffiths, Haiming Gang, Justin Lazarow, Gefen Kohavi, Kai Kang, Marcin Eichner, Yinfei Yang, Afshin Dehghan, and 1 others. 2025. Mm-spatial: Exploring 3d spatial understanding in multimodal llms. In *Proc. of ICCV*, pages 7395–7408.

Jiajun Deng, Tianyu He, Li Jiang, Tianyu Wang, Feras Dayoub, and Ian Reid. 2025. 3d-llava: Towards generalist 3d lmms with omni superpoint transformer. In *Proc. of CVPR*, pages 3772–3782.

Anna C. Doris, Md Ferdous Alam, Amin Heyrani Nobari, and Faez Ahmed. 2025. Cad-coder: An open-source vision-language model for computer-aided design code generation. *Proc. of NeurIPS*.

Michael Grieves and John Vickers. 2017. *Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems*.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, Jingji Chen, Jingjia Huang, Kang Lei, Liping Yuan, Lishu Luo, Pengfei Liu, Qinghao Ye, Rui Qian, Shen Yan, and 81 others. 2025. Seed1.5-vl technical report. *CoRR*, abs/2505.07062.

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, and 68 others. 2025. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Ting Huang, Zeyu Zhang, and Hao Tang. 2025. 3d-r1: Enhancing reasoning in 3d vlms for unified scene understanding. *arXiv preprint arXiv:2507.23478*.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, and 7 others. 2024. Mixtral of experts. *CoRR*, abs/2401.04088.

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.

Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. 2019. Abc: A big cad model dataset for geometric deep learning. In *Proc. of CVPR*, pages 9601–9611.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Feng Li, Renrui Zhang, Hao Zhang, Yuanhan Zhang, Bo Li, Wei Li, Zejun Ma, and Chunyuan Li. 2024a. Llava-next-interleave: Tackling multi-image, video, and 3d in large multimodal models. *CoRR*, abs/2407.07895.

Yichen Li, Kaichun Mo, Yueqi Duan, He Wang, Jiequan Zhang, Lin Shao, Wojciech Matusik, and Leonidas J. Guibas. 2024b. Category-level multi-part multi-joint 3d shape assembly. In *Proc. of CVPR*, pages 3281–3291. IEEE.

Yun Li, Yiming Zhang, Tao Lin, XiangRui Liu, Wenxiao Cai, Zheng Liu, and Bo Zhao. 2025. Sti-bench: Are mllms ready for precise spatial-temporal world understanding? *Proc. of ICCV*.

Ilya Loshchilov, Frank Hutter, and 1 others. 2017. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 5(5):5.

Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. 2023. Scalable 3d captioning with pretrained models. In *Proc. of NeurIPS*.

Wufei Ma, Guofeng Zhang, Qihao Liu, Guanning Zeng, Adam Kortylewski, Yaoyao Liu, and Alan L. Yuille. 2024. Imagenet3d: Towards general-purpose object-level 3d understanding. In *Proc. of NeurIPS*.

Dimitrios Mallis, Ahmet Serdar Karadeniz, Sebastian Cavada, Danila Rukhovich, Niki Maria Foteinopoulou, Kseniya Cherenkova, Anis Kacem, and Djamila Aouada. 2024. Cad-assistant: Tool-augmented vllms as generic CAD task solvers? *Proc. of ICCV*.

OpenAI. 2025. gpt-5 system card.

Ava Pun, Kangle Deng, Ruixuan Liu, Deva Ramanan, Changliu Liu, and Jun-Yan Zhu. 2025. Generating physically stable and buildable brick structures from text. In *Proc. of ICCV*, pages 14798–14809.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Proc. of NeurIPS*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Yunsheng Tian, Karl D. D. Willis, Bassel Al Omari, Jieliang Luo, Pingchuan Ma, Yichen Li, Farhad Javid, Edward Gu, Joshua Jacob, Shinjiro Sueda, Hui Li, Sachin Chitta, and Wojciech Matusik. 2024. ASAP: automated sequence planning for complex robotic assembly with physical feasibility. In *Proc. of ICRA*, pages 4380–4386. IEEE.

Yunsheng Tian, Jie Xu, Yichen Li, Jieliang Luo, Shinjiro Sueda, Hui Li, Karl DD Willis, and Wojciech Matusik. 2022. Assemble them all: Physics-based planning for generalizable assembly by disassembly. *ACM Transactions on Graphics (TOG)*, 41(6):1–11.

Chenrui Tie, Shengxiang Sun, Jinxuan Zhu, Yiwei Liu, Jingxiang Guo, Yue Hu, Haonan Chen, Junting Chen, Ruihai Wu, and Lin Shao. 2025. Manual2skill: Learning to read manuals and acquire robotic skills for furniture assembly using vision-language models. *CoRR*, abs/2502.10090.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, and 1 others. 2025. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Zhengyi Wang, Jonathan Lorraine, Yikai Wang, Hang Su, Jun Zhu, Sanja Fidler, and Xiaohui Zeng. 2024. Llama-mesh: Unifying 3d mesh generation with language models. *CoRR*, abs/2411.09595.

Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh Tran, Joseph G Lambourne, Armando Solar-Lezama, and 1 others. 2022. Joinable: Learning bottom-up assembly of parametric cad joints. In *Proc. of the CVPR*, pages 15849–15860.

Karl DD Willis, Yewen Pu, Jieliang Luo, Hang Chu, Tao Du, Joseph G Lambourne, Armando Solar-Lezama, and Wojciech Matusik. 2021. Fusion 360 gallery: A dataset and environment for programmatic cad construction from human design sequences. *ACM Transactions on Graphics (TOG)*, 40(4):1–24.

Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. 2025a. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. *Proc. of NeurIPS*.

Jinyang Wu, Chonghua Liao, Mingkuan Feng, Shuai Zhang, Zhengqi Wen, Haoran Luo, Ling Yang, Huazhe Xu, and Jianhua Tao. 2025b. Templaterl: Structured template-guided reinforcement learning for llm reasoning. *arXiv preprint arXiv:2505.15692*.

Jinyang Wu, Guocheng Zhai, Ruihan Jin, Jiahao Yuan, Yuhao Shen, Shuai Zhang, Zhengqi Wen, and Jianhua Tao. 2026. Atlas: Orchestrating heterogeneous models and tools for multi-domain complex reasoning. *arXiv preprint arXiv:2601.03872*.

Rundi Wu, Chang Xiao, and Changxi Zheng. 2021. Deepcad: A deep generative network for computer-aided design models. In *Proc. of CVPR*, pages 6772–6782.

Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. 2025. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proc. of CVPR*, pages 10632–10643.

Jiahao Zhang, Anoop Cherian, Cristian Rodriguez, Weijian Deng, and Stephen Gould. 2025a. Manual-pa: Learning 3d part assembly from instruction diagrams. In *Proc. of ICCV*, pages 6304–6314.

Xueyan Zhang, Jinman Zhao, Zhifei Yang, Yibo Zhong, Shuhao Guan, Linbo Cao, and Yining Wang. 2025b. UORA: uniform orthogonal reinitialization adaptation in parameter efficient fine-tuning of large models. In *Proc. of ACL*, pages 11709–11728. Association for Computational Linguistics.

Jinman Zhao, Xueyan Zhang, Jiaru Li, Jingcheng Niu, Yulan Hu, Erxue Min, and Gerald Penn. 2025. Tiny budgets, big gains: Parameter placement strategy in parameter super-efficient fine-tuning. In *Proc. of EMNLP*, pages 6315–6333. Association for Computational Linguistics.

A *CADBuilder* Dataset Construction

In this section, we provide more details about the construction process of our *CADBuilder* dataset. Fig. 6 shows the demonstration of diverse 3D objects in our dataset.

A.1 Reference Image Collection

We collect reference views for each multi-part assembly using an automated rendering pipeline in Blender. All parts are first normalized to a canonical scale, and centered at the origin to ensure consistent camera placement.Predefined viewpoints are ranked based on part visibility and projected area. For each candidate view, we define

$$\text{Score} = 0.5 \times S_{\text{vis}} + 0.5 \times S_{\text{area}} \tag{8}$$

The visibility score S_{vis} is computed via ray-casting on part samples and defined as a weighted sum of minimum and average ratios:

$$S_{\text{vis}} = 0.7 \times \min_i r_i + 0.3 \times \frac{1}{M} \sum_{i=1}^{M} r_i \tag{9}$$

where r_i denotes the visible-point ratio of part i. The projected area score S_{area} measures the 2D coverage of the assembly in camera space and is defined as

$$S_{\text{area}} = A_{\text{bbox}} \sqrt{d_{\text{avg}}} \tag{10}$$

where the bounding box is computed from all parts as $A_{\text{bbox}} = (x_{\max} - x_{\min})(y_{\max} - y_{\min})$ with $x, y \in [0, 1]$ and d_{avg} denotes the average camera-to-part distance. Views with higher combined scores are rendered with emphasized visible and hidden edges to highlight structural details. Multiple reference views are generated for each assembly and manually verified to ensure they sufficiently capture the essential geometric and structural characteristics of the object.

A.2 6-DoF Pose Initialization

We initialize the 6-DoF pose of each part using a collision-aware placement strategy. To avoid inter-part penetration, we perform an overlap check based on a projection-distance criterion between the current part and the already placed parts. Specifically, for two parts A and B, let $\mathbf{c}_A$ and $\mathbf{c}_B$ denote their geometric centers, and define the direction vector $\mathbf{d} = \frac{\mathbf{c}_A - \mathbf{c}_B}{\|\mathbf{c}_A - \mathbf{c}_B\|}$. We project all vertices of each part onto $\mathbf{d}$ and compute their extents along this direction. Let $L_A(\mathbf{d})$ and $L_B(\mathbf{d})$ denote the projected lengths of parts A and B, respectively. The two parts are considered non-overlapping if $\|\mathbf{c}_A - \mathbf{c}_B\| > \alpha\,(L_A(\mathbf{d}) + L_B(\mathbf{d}))$,where $\alpha \in [0.5, 0.6]$ is a relaxation factor to improve robustness. If overlap is detected, the pose is iteratively updated along the predefined disassembly direction $\mathbf{v}$ as $\mathbf{t}_{k+1} = \mathbf{t}_k + \lambda \mathbf{v}$, until the non-overlapping condition or a maximum iteration limit is reached, where $\mathbf{t}_k$ denotes the translation at iteration k and λ is a fixed step size. Following the initial disassembly order, pose initialization is performed sequentially. At order t, all parts initialized in earlier steps are merged into an aggregated part $\mathcal{G}_t = \bigcup_{i<t} \mathcal{P}_i$, and the overlap constraint is evaluated between the current part $\mathcal{P}_t$ and $\mathcal{G}_t$.This procedure yields a collision-free 6-DoF pose initialization that strictly adhere to the assembly logic.

A.3 Disassembly Simulation

We obtain plausible assembly trajectories by reversing physics-based disassembly simulations (Tian et al., 2022, 2024) and render them into videos using Blender. Although the simulation framework provides basic collision-free guarantees, we further verify each trajectory via Qwen3-VL-235B-A22B-Instruct (Bai et al., 2025) to ensure inter-part collision avoidance, assembly order feasibility, and physical plausibility (Fig. 7).

A.4 Training Data

A.4.1 Geometric CoT Annotation

To construct geometric chain-of-thought annotations for CAD objects, we design three complementary prompts covering shape descriptions of the selected part and the current assembled object (*i.e.*, sub-assembly), and their assembly compatibility constraints to construct our geometric chain-of-thought. Specifically, the selected part captioning prompt in Fig. 8 focuses on capturing the shape, assembly features, and functional role of individual parts. The current assembled object captioning prompt in Fig. 9 describes partially assembled structures by summarizing their combined appearance, engaged assembly features, and remaining exposed interfaces. Based on these captions, the assembly compatibility constraints reasoning prompt in Fig. 10 guides the model to explain why and how the candidate part can be assembled with the current assembled object, emphasizing geometric compatibility and structural alignment. Consequently, they enable geometric reasoning at each assembly

Figure 6: Demonstration of various 3D objects from our *CADBuilder* dataset.

step. To ensure the reliability of our geometric CoT annotations, we verify annotation quality through manual inspection of 100 random samples and automated verification using Gemini-2.5-Flash (Comanici et al., 2025). Over 95% of annotations pass automated verification, with the remaining samples manually reviewed and corrected.

A.4.2 Training Data Format

We adopt a supervised fine-tuning (SFT) formulation to train the model for step-by-step CAD assembly planning. Each training sample is organized as a multi-turn conversation, summarized in Fig.11. Specifically, each conversation begins with a system message that defines the assembly task, constraints, and the required output format. The user then provides the reference image of the complete assembly, multi-view images of the initial disassembled state, a fixed base object, and the initial 6-DoF poses of all remaining parts. At each turn, the assistant selects exactly one object to assemble and outputs three components: a geometric description of the selected part and the current sub-assembly, an assembly compatibility constraints explaining how these parts fit together, and a precise action trajectory represented as a sequence of relative 6-DoF transformations. After each step, the user supplies updated images of the partially assembled state, and the process continues until all parts are assembled. This SFT format explicitly couples geometric reasoning (CoT) with executable assembly actions, enabling the model to learn structured assembly planning. To ensure numerical stability and prevent overfitting to spurious precision, we round all 6-DoF transformation values to two decimal places, which aligns with practical CAD design tolerances while maintaining sufficient geometric accuracy. We additionally provide a subset of the SFT training data in the supplementary material.

A.5 Dataset Statistics

We construct *CADBuilder*, a parametric CAD assembly dataset that contains over 45K unique CAD objects across a wide range of assembly complexities in terms of part counts. Specifically, CAD objects with 2$\sim$5 individual parts include 35,483 samples, those with 6$\sim$10 parts contain 7,302 samples, and more complex objects with 11$\sim$15 parts comprise 2,276 samples. It provides enough multi-part data to learn complex structural and geometric relations. This distribution ensures coverage of diverse assembly difficulty levels, enabling models to learn assembly reasoning from simple to highly complex structural configurations.

B Additional Training Data for SPA Stage

We utilize Text2CAD (Khan et al., 2024) and DeepCAD (Wu et al., 2021) datasets as the data sources for the Spatial Perception Adaptation (SPA) stage.

Verification Prompt for Assembly Trajectory Validation

You are an expert in CAD assembly and mechanical engineering. Your task is to verify the physical plausibility of assembly trajectories by analyzing video demonstrations of the assembly process. Given the assembly video showing the sequential assembly of CAD parts, evaluate the following criteria:

1. Inter-Part Collision Detection:

- Are there any collisions or geometric interferences between parts during movement?
- Do moving parts penetrate through already-assembled components?

2. Assembly Order Feasibility:

- Is the assembly sequence logically sound and executable?
- Can each part be physically accessed and positioned without obstruction by other parts?

3. Physical Plausibility:

- Do the trajectories respect gravity and contact constraints?
- Are all movements physically realizable in real-world assembly?

Required Output Format (JSON):

```
{
    "collision_free": true/false,
    "feasible_order": true/false,
    "physically_plausible": true/false,
    "overall_valid": true/false,
}
```

Note: A trajectory is considered valid **only if** all three criteria are satisfied, i.e., `overall_valid = true`.

Figure 7: Prompt template for verification of assembly trajectory videos. The model evaluates three key aspects—collision-free movement, assembly order feasibility, and physical plausibility—to ensure the quality and executability of trajectories in the *CADBuilder* dataset.

Text2CAD provides image-text pairs with beginner-level CAD modeling instructions, which we use for the CAD caption task. DeepCAD contains parametric 3D objects, which we use for the 6-DoF pose prediction task. To prevent overlap with our *CADBuilder* test set, we filter out potential duplicates from the SPA training data. Specifically, we compute the chamfer distance (CD) between each SPA sample and all test samples in *CADBuilder*. Samples with CD < 0.03 are removed from the SPA training set, ensuring no data leakage during evaluation. After filtering, we retain $320K$ samples for SPA training.

B.1 Text2CAD

The Text2CAD dataset (Khan et al., 2024) is constructed based on DeepCAD (Wu et al., 2021), generating textual modeling instructions through large language and vision-language models. The construction pipeline first employs LLaVA-NeXT (Li et al., 2024a) to generate shape descriptions for CAD models and their intermediate components. These descriptions, along with design specifications from DeepCAD, are then fed into Mixtral-50B (Jiang et al., 2024) to produce multi-level CAD modeling instructions. We select beginner-level instructions that focus on how 3D objects are composed of basic geometric primitives. To increase data diversity, each CAD model is rendered from four viewpoints, yielding over 600K image-text pairs in total. During training, we prompt the MLLM to describe the compositional structure of CAD objects based on these image-text pairs. After filtering samples overlapping with our *CADBuilder* test set, we retain $160K$ samples for the CAD caption task in the SPA stage.

B.2 DeepCAD

The DeepCAD dataset (Wu et al., 2021) contains 178,238 CAD designs from Onshape's repository[5], each represented as sketch and extrusion command sequences. The MLLM is trained to predict their perturbed 6-DoF poses from standard viewpoints.

C More Analysis

C.1 Human Evaluation

Considering that automatic metrics may not fully capture the quality of assembly sequences from a design perspective, we conduct human evaluation to assess whether the generated assembly actions

[5]http://http://onshape.com

Prompt for Selected CAD Part Description

Prompt:
You are a CAD part description assistant. Your task is to describe the candidate part in a way that highlights its overall shape, key assembly features, and functional role.
Requirement for input:

- The ID of the corresponding part
- Multi-view rendered images of the candidate part (<image> tags)
- Basic geometric information such as length, width, and height (LWH)

Requirement for output:

- The selected part:
  - The approximate overall shape of the part (*e.g.*rod, plate, block, funnel, bracket)
  - Key assembly-related features (e.g., holes, slots, grooves, protrusions)
  - Functional role in an assembly (e.g., used for connection, used as a container, can be inserted into another part)

Rules:

- Do NOT use words such as "image", "CAD model", or any color or material descriptors, and avoid uncertain terms like "may", "likely", or "possibly".
- The description must always start with: `Part <|Part_x|>`, where x is the part ID.
- Focus on shape, structure, holes or perforations, topology, and proportions inferred from the images and JSON data.
- The output must be plain natural language text only.

Input:
Part ID: <|Part_x|>
Part images: <image><image><image>
Dimensions: length=l, width=w, height=h

Figure 8: Prompt for the selected part description annotation.

align with human designers' expectations and practical assembly workflows. Evaluators were compensated at a rate of $0.50 per sample, equivalent to approximately $20 per hour. We randomly select 100 samples from the CADBuilder test set where CADMATE-8B and the strongest baseline (*i.e.*, Qwen3-VL-8B-Thinking) both achieve CD < 5.0 and PA > 50%. The selected samples span different complexity levels and are consistent across all evaluated methods. We invite six graduate students majoring in industrial design with CAD design experience to evaluate the generated assemblies based on three criteria: **Assembly Rationality (Rat)** measures whether the predicted assembly sequence and 6-DoF transformations follow logical and practical assembly workflows that a human designer would adopt. **Spatial Accuracy (Spa)** assesses whether the predicted part poses are spatially accurate with minimal positioning errors. **Physical Feasibility (Phy)** evaluates whether the assembly process avoids collisions, penetrations, and physically implausible movements. Each criterion is scored on a scale from 0 to 2, with higher scores indicating greater alignment with human design practices. Table 2 presents the human evaluation results. The standard deviations confirm the reliability of our assessment. We observe that: **(i)** Fine-tuned models demonstrate reasonable spatial accuracy, suggesting they have learned to predict plausible part positions. **(ii)** Our CADMATE-8B outperforms the baseline in assembly rationality and physical feasibility, demonstrating that our geometric chain-of-thought reasoning and spatial-physical rewards enable more human-like and physically valid assembly generation.

Model	Rat	Spa	Phy
Qwen3-VL-8B	1.24/0.35	1.31/0.28	0.96/0.31
CADMATE-4B	1.42/0.32	1.38/0.26	1.18/0.29
CADMATE-8B	**1.58**/0.28	**1.47**/0.24	**1.32**/0.27

Table 2: Human evaluation results. Each value is presented as μ/σ, where μ is the mean score and σ is the standard deviation. **Bold**: best performance.

C.2 Impact of RL Training Data Scale

To investigate how the scale and source of RL training data affect model performance, we conduct experiments with different data compositions. We apply the pass@k criterion ($1 \leq k \leq 7$) to obtain trajectories from two sources: $\mathcal{D}_1$ denotes the 28K SFT training set, yielding approximately 4K RL samples after filtering; $\mathcal{D}_2$ denotes an additional 15K CAD assemblies, contributing around

Prompt for Current Assembled Object Description

Prompt:
You are a CAD assembled object description assistant. Your task is to describe partially assembled CAD structures in a way that highlights the combined appearance, the assembly features already engaged, and the remaining exposed interfaces for further assembly.

Requirement for input:

- The IDs of the current assembled object
- Multi-view rendered images of the current assembled object state (`<image>` tags)
- Basic geometric information such as length, width, and height (LWH)

Requirement for output:

- A description should cover three aspects:
  - Overall appearance of the combined structure (e.g., "a flat plate with a cylinder attached on top")
  - Completed assembly features already engaged (e.g., "the cylinder is inserted into the central hole of the plate")
  - Exposed interfaces available for further assembly (e.g., "the cylinder's upper end remains exposed, and the plate has unused side holes")

Rules:

- Do NOT use words such as "image", "CAD model", or any color or material descriptors, and avoid uncertain terms like "may", "likely", "probably", or "possibly".
- The description must always start with: `The assembled part (<|Part_x|> <|Part_y|> ...)` using the full list of part IDs.
- Focus on shape, structure, holes or perforations, topology, and proportions inferred from the images and JSON data.
- The output must be plain natural language text only.

Input:
IDs of the current assembled object: `The assembled part (<|Part_x|> <|Part_y|> ...)`
Images of the current assembled object: `<image><image><image><image>`
Dimensions: length=l, width=w, height=h

Figure 9: Prompt for the current assembled object description annotation.

6K RL samples. We train CADMATE-4B and CADMATE-8B using different combinations of these data sources and report results on the 10∼15 parts subset in Table 3. **(i) RL training on SFT data alone yields marginal improvements.** When training with only $\mathcal{D}_1$, CADMATE-4B shows modest gains (CD: 3.05 vs. 3.42 for SFT-only baseline), suggesting that in-domain RL refinement provides limited benefit when the data distribution is already well covered during SFT. **(ii) Scaling RL data with additional samples substantially enhances performance.** Incorporating $\mathcal{D}_2$ leads to more pronounced improvements, with CADMATE-4B achieving CD of 2.92 using $\mathcal{D}_2$ alone, and the combined $\mathcal{D}_1 + \mathcal{D}_2$ further reduces CD to 2.77, approaching our full model's performance. This demonstrates that diversifying RL training data is crucial for learning robust spatial-physical reasoning. **(iii) Larger models benefit more from scaled RL training data.** The performance gain from data scaling is more pronounced for CADMATE-8B. While the 4B model improves CD by 0.28 (from 3.05 to 2.77) when scaling from $\mathcal{D}_1$ to $\mathcal{D}_1 + \mathcal{D}_2$, the 8B model achieves a larger gain of 0.45 (from 2.23 to 1.78). This suggests that SFT primarily provides syntax grounding and basic assembly understanding for smaller models, while larger models already encode richer geometric and spatial priors that RL can more effectively amplify when exposed to diverse training signals.

Model	$\mathcal{D}_1$	$\mathcal{D}_2$	MMD↓	CD↓	PA↑	S_a↑
CADMATE-*4B*						
SFT (no RL)	-	-	4.85	3.42	49.25	3.58
	✓	-	4.52	3.05	51.68	3.85
	-	✓	4.38	2.92	53.12	3.98
	✓	✓	4.26	2.77	54.82	4.12
CADMATE-*8B*						
SFT (no RL)	-	-	4.12	2.68	58.42	4.05
	✓	-	3.86	2.23	62.35	4.32
	-	✓	3.72	2.05	64.85	4.48
	✓	✓	**3.45**	**1.78**	**68.56**	**4.75**

Table 3: Impact of RL training data scale and composition on the 10∼15 parts subset. ✓ indicates the data source is used. Best results are shown in **bold**.

C.3 Impact of Hyperparameters on Rewards

As described in Eq. 6, our overall reward is a weighted combination of spatial reward $\mathcal{R}_s$, physical reward $\mathcal{R}_p$, and format reward $\mathcal{R}_f$, with hyper-

Prompt for CAD Assembly Compatibility Constraints

Prompt:
You are a CAD assembly compatibility constraints reasoning assistant. Your task is to explain why and how the candidate part can be assembled with the current assembled object, based on geometric and structural compatibility.
Requirement for input:

- The ID of the corresponding part
- Multi-view rendered images of the candidate part
- The IDs of the current assembled components are already combined
- Multi-view rendered images of the current assembled object
- descriptions of both the candidate part and the current assembled object from the previous step

Requirement for output:

- A short reasoning (2–4 sentences) explaining why the part can be assembled with the current assembled object
- The reasoning focuses on geometric fit, matching features (e.g., hole–peg, slot–tab, flat–flat contact), and functional alignment such as support, fixation, or insertion
- Both the part and the sub-assembly must be explicitly mentioned

Rules:

- Do NOT use words such as "image", "CAD model", or any color or material descriptors, and avoid uncertain expressions like "may", "likely", or "possibly"
- The reasoning must always start with:
  `Part <|Part_x|> can be assembled with Sub-assembly <|Part_a, Part_b|>`
  where x is the part ID and $(a, b, \ldots)$ are the sub-assembly part IDs
- Focus on geometric compatibility such as holes, protrusions, alignment surfaces, and structural matching
- The output must be plain natural language text only

Input (User Prompt):
Part ID: `<|Part_x|>`
Part images: `<image><image><image>`
Part description (from previous step): `<description_single>`
IDs of the current assembled object: `<|Part_a|> <|Part_b|> ...`
Images of the current assembled object: `<image><image><image>`
Description of the current assembled object (from previous step): `<daption_assembly>`

Figure 10: Prompt for assembly compatibility constraints between a candidate part and current assembled object.

parameters λ_s, λ_p, and λ_f respectively. We investigate how different weight configurations affect the performance of CADMATE-8B on the 10∼15 parts subset, as shown in Table 4. We observe that: **(i)** The balanced configuration ($\lambda_s = 0.5$, $\lambda_p = 0.4$, $\lambda_f = 0.1$) achieves the best overall performance. Over-emphasizing spatial accuracy ($\lambda_s = 0.7$) improves CD to 1.65 but degrades S_a to 4.52, indicating increased collisions. Conversely, prioritizing physical feasibility ($\lambda_p = 0.6$) improves S_a to 4.88 but worsens CD to 2.12, demonstrating the trade-off between spatial precision and collision avoidance. **(ii)** While removing format reward ($\lambda_f = 0$) causes minor degradation, the model still maintains reasonable performance, suggesting that $\mathcal{R}_s$ and $\mathcal{R}_p$ are the primary drivers of performance, with $\mathcal{R}_f$ serving as an auxiliary constraint.

λ_s	λ_p	λ_f	**MMD↓**	**CD↓**	**PA↑**	**S_a↑**
0.7	0.2	0.1	3.28	1.65	70.42	4.52
0.3	0.6	0.1	3.82	2.12	65.38	4.88
0.45	0.45	0.1	3.52	1.85	67.85	4.72
0.55	0.45	0.0	3.68	1.92	66.28	4.58
0.5	0.4	0.1	**3.45**	**1.78**	**68.56**	**4.75**

Table 4: Impact of reward hyperparameters on CADMATE-8B on the 10∼15 parts subset. Best results are shown in **bold**.

C.4 Comparison between GRPO and DPO

To further validate the effectiveness of our reward-based RL approach, we compare Group Relative Policy Optimization (GRPO) with Direct Preference Optimization (DPO) (Rafailov et al., 2023), a widely-used preference-based RL method. For DPO training, we construct preference pairs by sampling multiple trajectories for each assembly and ranking them based on our spatial-physical reward scores. We report results for CADMATE-8B in Table 5. We observe that: **(i)** DPO underperforms the SFT baseline across all complexity levels, with CD increasing from 4.78 to 5.23 on the 5∼10 parts subset and from 3.65 to 4.15 on the 10∼15 parts subset. This suggests that preference-

SFT Training Data Format

Conversation Format (JSON-style)
System Message:
Defines the CAD assembly task, including assembly constraints, coordinate conventions, and output format requirements for single-object trajectory generation.
User Message (Turn 1):

- Reference image of the complete assembly
- Multi-view images of the initial disassembled state
- Fixed base object with its 6-DoF pose
- List of disassembled objects with object IDs (<|Part_ID|>) and initial 6-DoF poses

Assistant Message (Turn t):

- <description>: Geometric chain-of-thought describing the selected part and the current sub-assembly
- <constraint>: Reasoning on assembly compatibility and geometric constraints
- Action sequence for assembling exactly one object, formatted as:

Select <|Part_X|>,
<|begin_of_action|>[Δx,Δy,Δz,Δrx,Δry,Δrz], ... <|end_of_action|>

User Message (Turn $t+1$):

- Updated multi-view images of the partially assembled state
- Instruction to select the next object and generate its assembly trajectory

Termination:
The conversation ends when all disassembled objects have been sequentially assembled.

Figure 11: SFT training data format with geometric chain-of-thought for CAD assembly.

based training struggles to optimize tasks requiring precise spatial reasoning. **(ii)** In contrast, GRPO achieves substantial improvements over SFT. It validates that directly optimizing spatial-physical rewards is more effective than learning from preference comparisons for CAD assembly tasks.

D More Experiment Details

D.1 Evaluation Metrics

To comprehensively evaluate the quality of generated CAD objects, we adopt four complementary evaluation metrics that assess geometric fidelity, part-level accuracy, structural correctness, and assembly-process consistency. Specifically, we report Minimum Matching Distance (MMD), Chamfer Distance (CD), Part Accuracy (PA), and Model-based Assembly Scoring (S_a). Together, these metrics provide a multi-level evaluation from individual part geometry to overall assembly validity and dataset-level distribution alignment.

Minimum Matching Distance (MMD) measures the discrepancy between the distribution of generated assemblies and that of the ground-truth assemblies. Given the ground-truth set $\mathcal{G}$ and the generated set $\mathcal{S}$, MMD is defined as:

$$\mathrm{MMD} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \min_{s \in \mathcal{S}} \mathrm{CD}(g, s), \tag{11}$$

where $\mathrm{CD}(\cdot, \cdot)$ denotes the Chamfer Distance between two point clouds. For each ground-truth assembly, we compute its Chamfer Distance to all generated samples and select the minimum value.

Chamfer Distance (CD) evaluates the geometric similarity between a generated assembly and its corresponding ground-truth assembly. Given a generated point cloud (X) and the ground-truth point cloud (Y), the Chamfer Distance is defined as:

$$\begin{aligned} \mathrm{CD}(X, Y) = &\frac{1}{|X|} \sum_{x \in X} \min_{y \in Y} \|x - y\|_2^2 \\ &+ \frac{1}{|Y|} \sum_{y \in Y} \min_{x \in X} \|y - x\|_2^2, \end{aligned} \tag{12}$$

For each matched assembly pair, CD is computed on normalized point clouds. We report both the mean and the median CD over all test samples. Lower CD values indicate higher geometric accuracy of the generated assemblies.

Part Accuracy (PA) measures part-level geometric correctness within an assembly. We define a part as accurate if its Chamfer Distance (CD) to the ground truth is within a predefined threshold τ ($\mathrm{CD} \leq \tau$) set as 0.01. For an assembly with N parts, PA is defined as:

$$\mathrm{PA}_{\text{assembly}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\mathrm{CD}_i \leq \tau)\,. \tag{13}$$

Method	2 ∼ 5 parts				5 ∼ 10 parts				10 ∼ 15 parts			
	MMD↓	CD↓	PA↑	S_a↑	MMD↓	CD↓	PA↑	S_a↑	MMD↓	CD↓	PA↑	S_a↑
SFT-only	1.85	1.98	87.42	4.98	3.92	4.78	68.25	4.65	5.25	3.65	56.45	3.92
DPO	2.12	2.28	85.15	4.72	4.38	5.23	64.82	4.35	5.78	4.15	52.68	3.65
GRPO	**1.48**	**1.62**	**90.35**	**5.15**	**2.82**	**3.85**	**76.92**	**5.08**	**3.45**	**1.78**	**68.56**	**4.75**

Table 5: Comparison between GRPO and DPO on CADMATE-8B. Best results are shown in **bold**.

Model-based Assembly Scoring (S_a) is a model-based scoring metric that evaluates the quality of a generated CAD assembly using rendered videos and images with Gemini-2.5-Flash. Specifically, S_a consists of three independent scores, each assigned by a multimodal vision-language model: (i) **Part Movement (PM)** ($0 \sim 2$): evaluates whether part motions during assembly contain penetrations or physically implausible collisions. (ii) **Assembly Gap (AG)** ($0 \sim 2$): assesses whether visible gaps exist between parts in the final assembled state. (iii) **Structure Similarity (SS)** ($0 \sim 2$): measures the structural consistency between the final assembled result and the ground-truth CAD object. The complete evaluation prompt is shown in Fig. 12. The score S_a is computed as:

$$S_a = S_{\text{PM}} + S_{\text{AG}} + S_{\text{SS}}, \quad S_a \in [0, 6]. \tag{14}$$

We validate this metric on 100 randomly sampled assemblies, achieving a Pearson correlation of 0.89 with human ratings, confirming its reliability.

D.2 RL Settings

We utilize the Hugging Face Transformers library, GRPO implementation from TRL (von Werra et al., 2020) and DeepSpeed ZeRO-2 for distributed training. For efficient model inference, we employed vLLM framework (Kwon et al., 2023) to rollout action sequences. For RL training data preparation, we apply the pass@k criterion (with $1 \leq k \leq 7$) with Eq. 2 on filter both the 28K SFT training set and an additional 15K samples. Finally, we obtain over 10K data samples for RL training.

D.3 Baselines Implementation

We evaluate CADMATE against three kinds of baselines: leading large-scale MLLMs (*i.e.*, over 100B parameters), fine-tuning small reasoning-oriented MLLMs, and a 3D point cloud based CAD assembly model. Notably, in our preliminary experiments, we utilize parameter-efficient strategies (Hu et al., 2021; Zhao et al., 2025; Zhang et al., 2025b) and find that the models fail to learn correct coordinate outputs. Therefore, we utilize the full fine-tuning for our CADMATE Table 6 provides details of these baseline models. We elaborate on their implementation as follows.

Leading Large-scale MLLMs. We evaluate 7 leading large-scale MLLMs using the same prompting protocol (*i.e.*, system message and few-shot examples), as shown in Fig. 13. Each model is provided with a reference image of the target CAD object, multi-view images of individual parts with object IDs, and the initial 6-DoF poses, with one part fixed as the base. The models are instructed to generate assembly trajectory, each consisting of a minimal sequence of relative 6-DoF actions, following a strict output format.

Fine-tuning Small MLLMs. We fully fine-tune three reasoning-oriented MLLMs: Qwen3-VL-8B, GLM-4.1V-Thinking-9B, and InternVL-3.5-8B. These models are trained on the same data as CADMATE's SFT stage, but with chain-of-thought reasoning removed to predict action sequences directly. We train for 3 epochs with a learning rate of 1e-5, and report results from the best-performing checkpoint in Table 1.

3D Point Cloud based Model. We implement Manual-PA (Zhang et al., 2025a), a point cloud-based assembly method that directly predicts final 6-DoF poses for each part without modeling initial states or action sequences. Manual-PA operates in a simplified setting by directly predicting final poses rather than generating assembly actions. We train it on *CADBuilder* following the original protocol. The video-based S_a evaluation is not applicable as it does not generate assembly processes.

E More Cases

Fig. 14, 15 and 16 present representative cases of assembly processes generated by our CADMATE-8B. In contrast, Fig. 17 illustrates a failure case.

Model-based Assembly Scoring (S_a) Evaluation Protocol

Task Description:
Evaluate CAD assembly quality by analyzing assembly videos and comparing final results with ground-truth structures using a multimodal vision-language model.

Evaluation Dimensions (Each scored 0–2, integer only):

1. Part Movement (PM) — Video-based collision detection
- **Description:** Observe part movements throughout assembly to detect penetrations or collisions
- **0:** Clear penetration or collision between parts during movement
- **1:** Minor contact or near-collision, but not severe
- **2:** Smooth movement with no penetration or collision at any stage

2. Assembly Gap (AG) — Video-based gap assessment
- **Description:** Inspect fit between parts after assembly completion for visible gaps
- **0:** Obvious gaps between assembled parts, indicating poor fit
- **1:** Minor gaps exist, but overall assembly is mostly tight
- **2:** No gaps visible; parts fit precisely with perfect alignment

3. Structure Similarity (SS) — Image-based structural comparison
- **Description:** Compare final assembly frame with ground-truth CAD structure images
- **0:** Final structure differs significantly from ground truth with major discrepancies
- **1:** Somewhat similar to ground truth, but with noticeable differences
- **2:** Very close to ground truth with minimal or no differences

Output Format:
- `<think>`Detailed scoring rationale for each dimension `</think>`
- `<answer>`Three comma-separated scores (e.g., `2,1,2`)`</answer>`
- **Final Score:** $S_a = S_{\text{PM}} + S_{\text{AG}} + S_{\text{SS}} \in [0, 6]$

Figure 12: Model-based assembly scoring prompt. The framework assesses assembly quality through three complementary dimensions: part movement collision detection, assembly gap measurement, and structural similarity to ground truth.

MLLMs	LLM (Size)	Vision Encoder	Link
	Open-Source Models		
Qwen3-VL-8B	Qwen3 (8B)	SigLip-400M	Qwen/Qwen3-VL-8B-Thinking
GLM-4.1V-9B	GLM-4-9B-0414 (9B)	Aimv2-Huge	zai-org/GLM-4.1V-9B-Thinking
InternVL3.5-8B	Qwen3-8B (8B)	InternViT-300M	OpenGVLab/InternVL3_5-8B
GLM-4.5V	GLM-4.5-Air (106BA12B)	Aimv2-Huge@224px	zai-org/GLM-4.5V
Manual-PA	N / A	DINOv2 & PointNet	DavidZhang73/Manual-PA
	Proprietary Models		
Qwen3-VL-Plus	Qwen3-Plus	N / A	Qwen3-VL-Plus
Seed-1.6-Vision	N / A	N / A	Seed-1.6-Vision
Gemini-2.5-Flash	N / A	N / A	Gemini-2.5-Flash
Gemini-2.5-Pro	N / A	N / A	Gemini-2.5-Pro
GPT-5-mini	N / A	N / A	GPT-5-mini
GPT-5	N / A	N / A	GPT-5

Table 6: The details of the baseline models in our comparison.

Few-shot Prompt for Large-scale MLLM Baselines

System: You are a CAD assembly assistant. For the given parts, generate operation instructions to move and assemble ALL parts. Each part must be represented by exactly ONE trajectory, and each trajectory can contain multiple actions (separated by commas). Use as few actions as possible for each trajectory, and ensure that during movement, collisions or mesh penetrations are avoided as much as possible.

Inputs:

1. Complete assembly design (Reference Image)
2. Multi-view images of each part with its object ID (<|Part_ID|>)
3. 6-DoF pose of each part ([x, y, z, rx, ry, rz])
4. One part is fixed as the base; all others are assembled onto it.

Task: Output exactly one trajectory per part to assemble. Each output must include the <|Part_ID|> of the operated part and its action sequence.

CRITICAL - OUTPUT REQUIREMENTS:

- ONLY output trajectory blocks in the exact format shown in examples
- DO NOT include ANY reasoning, analysis, explanatory text, or thinking process
- DO NOT add comments, descriptions, or additional information
- Start your response directly with <trajectory> tags
- Follow the exact output format from the examples below
- The number of <trajectory> blocks MUST be exactly equal to the number of moveable parts

Motion Planning Guidelines:

- Move ALL parts except the fixed part; each part must have only one trajectory
- Each trajectory can contain multiple actions (separated by commas), Use the fewest possible action steps for each trajectory
- Plan trajectories that avoid collision with existing parts (both fixed and already assembled parts)
- Each action vector contains six floating-point values: [Δx, Δy, Δz, Δrx, Δry, Δrz]
- Δx, Δy, Δz: relative translation toward final assembled pose (in units)
- Δrx, Δry, Δrz: relative rotation parameters (in radians)

Few-shot Examples: INPUT:

Reference Image: <image>
Images of initial disassembled state: <image><image><image>
Object initial 6-DoF pose:
Fixed object: <|Part_1|> [-0.01, -2.35, -0.24, 0.00, 0.00, 0.00] <image>
Moveable object:
<|Part_0|> [2.62, 0.05, 9.43, 0.00, 0.00, 0.00] <image>
<|Part_2|> [-1.23, -10.33, 0.17, 0.00, 0.00, 0.00] <image>
<|Part_3|> [-2.79, 5.99, -0.01, 0.00, 0.00, 0.00] <image>

OUTPUT:

<trajectory>Select <|Part_3|> to assemble, <|begin_of_action|>[0.7, -1.33, -0.02, 0.00, 0.00, 0.00], [0.00, -5.05, 0.00, 0.00, 0.00, 0.00]<|end_of_action|></trajectory>
<trajectory>Select <|Part_2|> to assemble, <|begin_of_action|>[1.25, 4.84, -0.16, 0.00, 0.00, 0.00], [0.00, 6.06, 0.00, 0.00, 0.00, 0.00]<|end_of_action|></trajectory>
<trajectory>Select <|Part_0|> to assemble, <|begin_of_action|>[-0.51, -0.46, -1.88, 0.00, 0.00, 0.00], [0.00, 0.00, -7.57, 0.00, 0.00, 0.00]<|end_of_action|></trajectory>
Please refer to the EXAMPLE above. For the following INPUT, output ONLY the trajectories for the given input, following the exact format and not providing any explanation or extra text.

Output Format (STRICT):

- One <trajectory>...</trajectory> block per part to assemble
- Inside each trajectory: 'Select <|Part_ID|> to assemble, <|begin_of_action|>...<|end_of_action|>'
- Multiple trajectories separated by \n
- NO additional text outside this format

Figure 13: The few-shot prompt for large-scale MLLM baselines to perform the CAD assembly task.

Figure 14: A case of generated actions with geometric CoT and rendered assembly process for our CADMATE-8B.

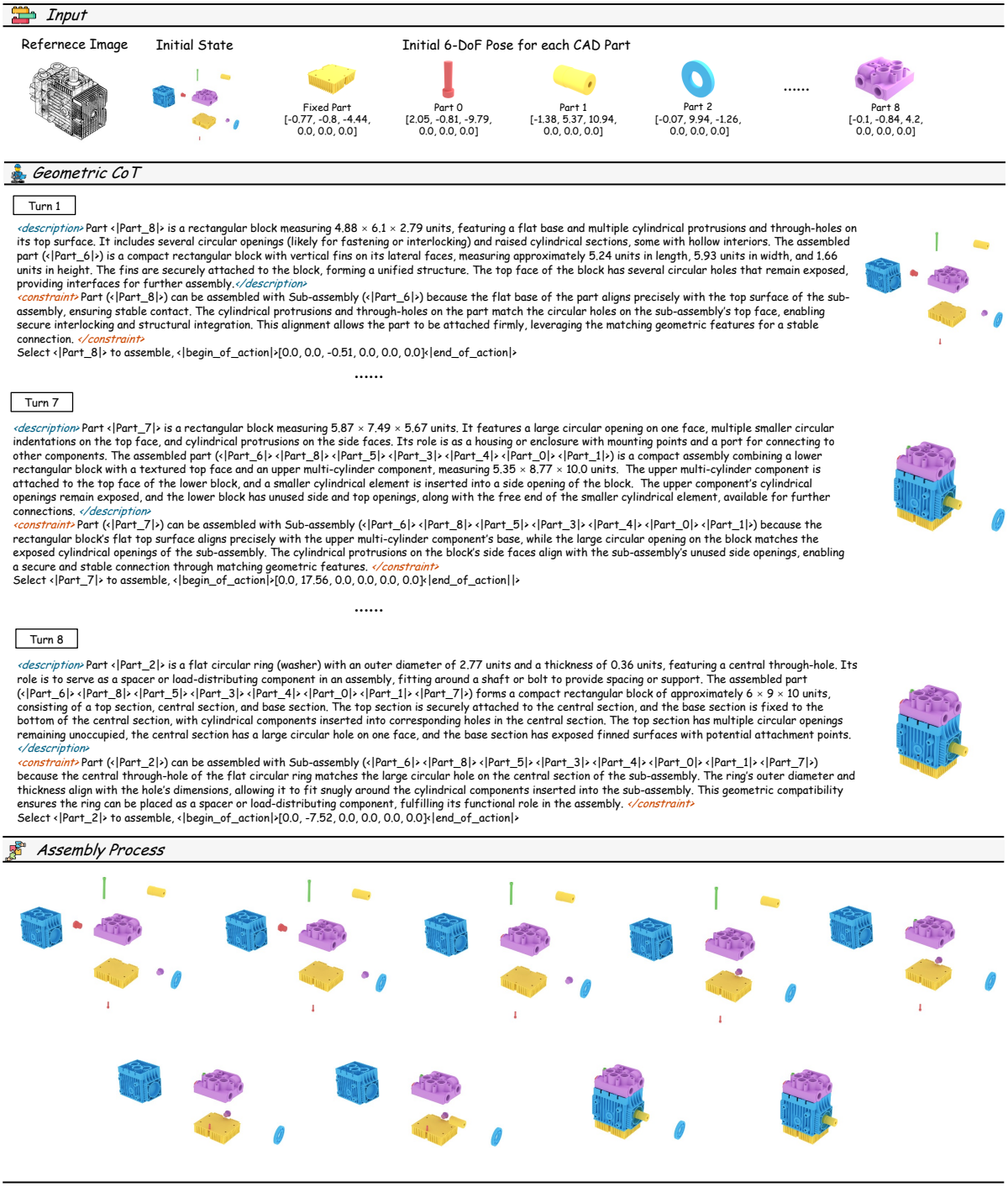

Figure 15: A case of generated actions with geometric CoT and rendered assembly process for our CADMATE-8B.

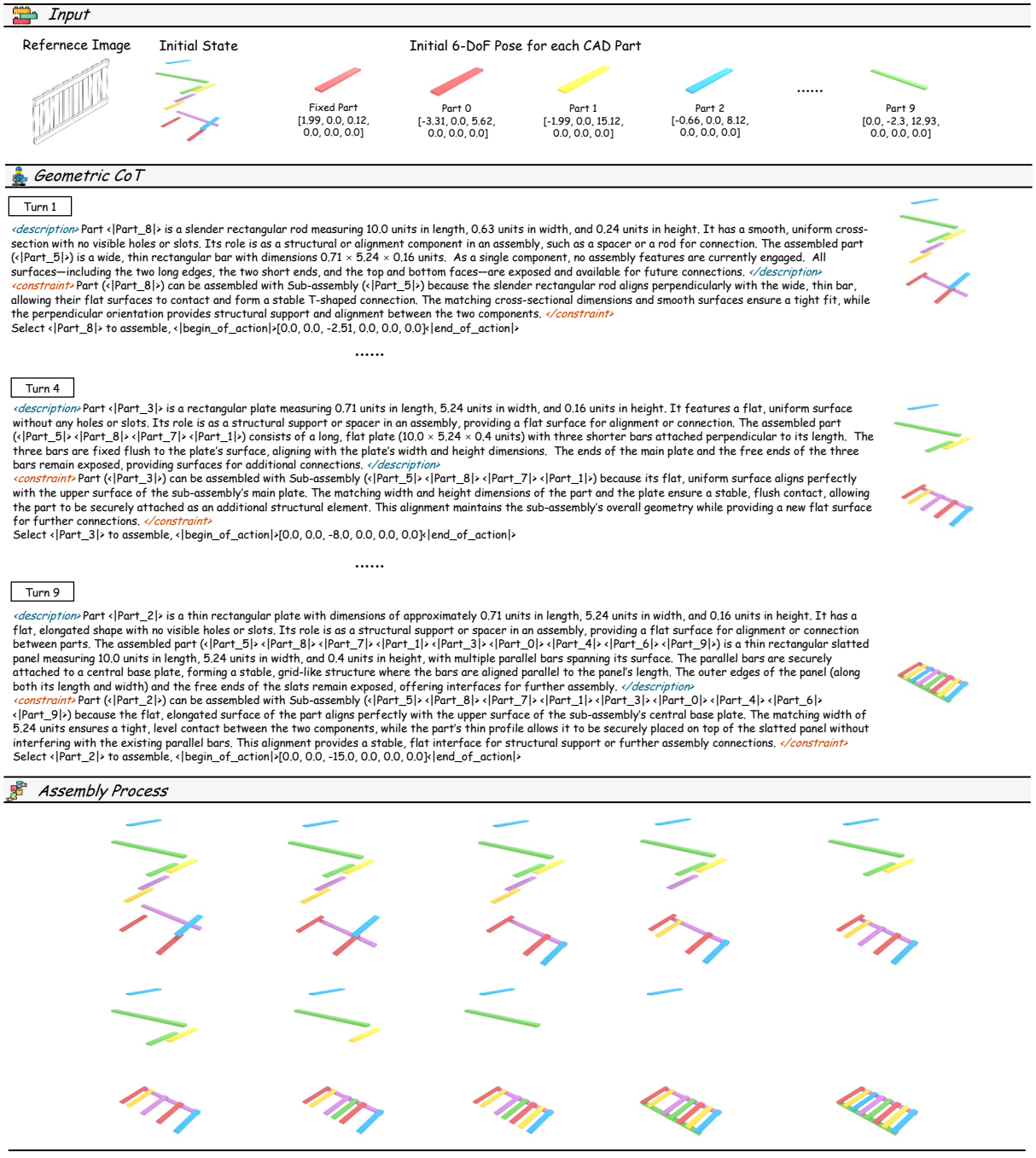

Figure 16: A case of generated actions with geometric CoT and rendered assembly process for our CADMATE-8B.

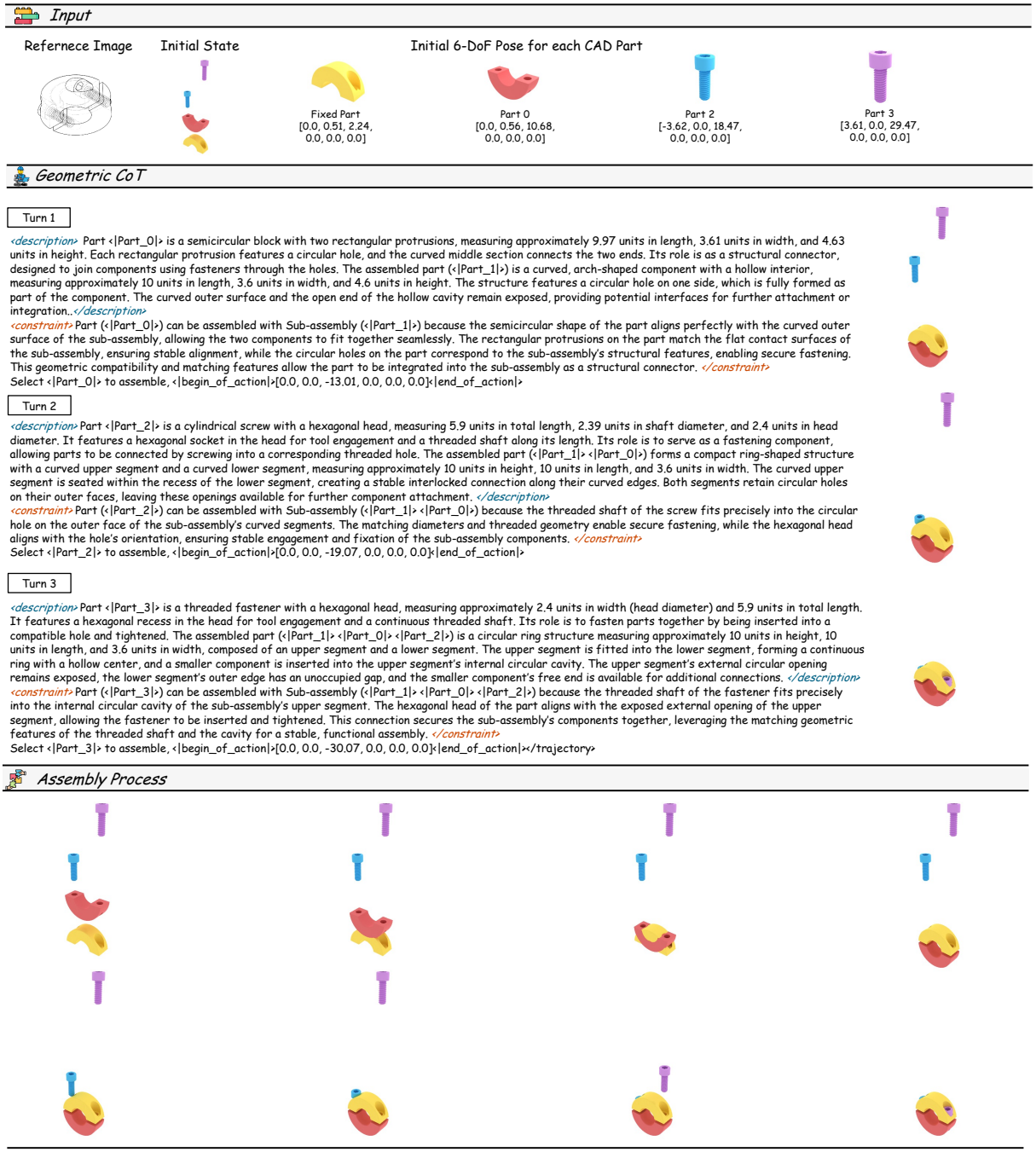

Figure 17: A representative failure case from CADMATE-8B. While the model correctly infers the assembly sequence, it produces physically implausible motions with inter-part penetrations during intermediate steps. It illustrates that physical collision avoidance remains a significant challenge for current MLLMs.