UniCAD: A Prototype-Enhanced Unified Framework for CAD Construction Sequence Generation

Meng Yuan
Jilin University, China
mengyuan23@mails.jlu.edu.cn

Dawei Lin
Jilin University, China
lindw23@mails.jlu.edu.cn

Tieru Wu*
Jilin University, China
wutr@jlu.edu.cn

Rui Ma*
Jilin University, China
Engineering Research Center of Knowledge-Driven Human-Machine Intelligence, MOE, China
ruim@jlu.edu.cn

Abstract

Rapid generation of CAD models is essential for industrial product design. However, existing approaches mainly focus on generating CAD construction sequences from a single input modality, such as text or image, which often contains ambiguous or limited information. To address this, we propose UniCAD, a prototype-enhanced unified CAD generation framework that supports free-form user inputs, including text descriptions and/or images, allowing users to express versatile design intentions. Specifically, we construct two sets of prototypes from the textual and visual modalities using online update and cumulative averaging strategies. With these prototypes, we not only enhance unimodal input prompts by incorporating information from the complementary modality, but also facilitate consistent representation learning across modalities in the latent space. Additionally, a progressive learning strategy, which first incorporates the full CAD construction sequences and then gradually masks out the CAD tokens during the training, is proposed to enable the generation of CAD construction sequences based on prompts of a single modality or their combinations. Furthermore, we construct a new dataset comprising image-text-CAD triplets to support multimodal learning. Extensive experiments demonstrate the effectiveness of UniCAD in CAD sequence generation and completion tasks.

***Keywords:** Computer-aided design, CAD generative modeling, CAD construction sequence, multimodal learning*

*Corresponding authors.

1. Introduction

In industrial product design, designers express their understanding, imagination, and invention of a product in 3D forms [36]. From gaskets to automobiles and aerospace components, designers need to rapidly create preliminary models, which are refined and revised iteratively until detailed Computer-Aided Design (CAD) models are constructed for product manufacturing [25].

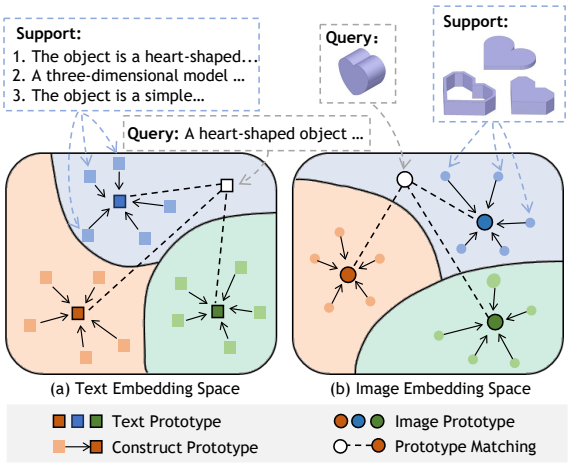

Figure 1. An illustrative example for textual and visual prototypes. These prototype representations serve as central reference points for the support samples, capturing the key features of each class calculate the cosine distance with all prototypes, where the smallest distance is its corresponding class. With the prototypes, a single modality can compensate for the missing or suboptimal information from the other modality, enhancing the quality of the CAD sequence generation.

Recent advances in deep learning have driven automation efforts in the CAD domain. The CAD construction sequence, a data representation method consisting of a sequence of commands such as lines, arcs, and circles [36],

has attracted significant attention due to its flexibility and reusability, which help reduce designers' workload in the time-consuming design process and enable seamless integration with rapidly advancing deep learning techniques. As an example, DeepCAD [36] learns a transformer-based autoencoder to encode the CAD construction sequence into a latent code and trains a latent-GAN for the generation of CAD construction sequences. A series of works [16, 42] further introduce additional prompts as conditions to generate CAD construction sequences. Generally, contrastive learning-based (CL) [1, 22, 26] and transformer-based [5, 16] methods have become the mainstream methods. The former methods aim to establish a joint embedding space to align representations across different modalities, while the latter dynamically adjust the importance of each modality through attention mechanisms, thereby facilitating effective cross-modal interaction. For example, CAD-Translator [22] introduces a cascaded contrastive learning strategy to align text descriptions with CAD construction sequences in a shared semantic space. Another potential solution is to leverage an image [5] or a point cloud [26] to guide the generation of CAD construction sequences.

Despite recent advances, these methods are mainly designed for single-modal inputs. However, in practical design scenarios, a single textual description may refer to multiple objects, as illustrated in Fig.2 (b). Therefore, users often resort to multiple modalities, such as text and images, to generate CAD construction sequences, which motivates us to develop a unified generation framework. Developing such a framework requires consideration of the following inherent properties: (1) *Diversity of representations.* The same object can be expressed in multiple ways, rather than being limited to a single prompt, as shown in Fig.2 (a). However, CL-based methods [1, 22, 26] rely on well-structured and descriptive textual input, making them less robust when handling diverse and varied textual inputs. (2) *Modal complementarity.* The text and image modalities present information in distinctly different formats. Specifically, the former provides abstract semantic information, whereas the latter conveys concrete geometric structures and spatial relationships. By integrating multimodal information, a more comprehensive understanding of design intent can be achieved, enhancing the quality of CAD model generation. However, an important challenge remains: how to effectively complete missing information with only a single modality as input?

To address these issues, we propose UniCAD, a novel prototype-enhanced unified framework for CAD construction sequence generation. Specifically, we introduce a progressive training strategy that learns the mapping from a cross-modal joint embedding space to the structured CAD operation space during model training. During inference, it directly transforms raw single-modal or multimodal inputs

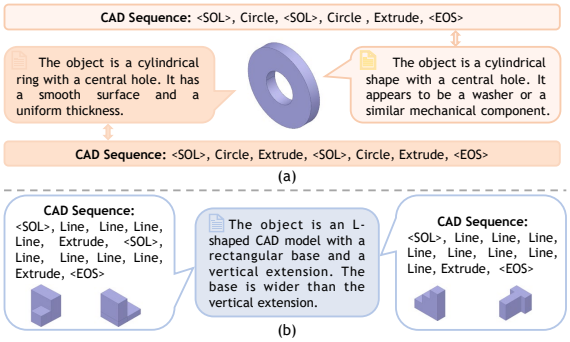

Figure 2. CAD models and their text descriptions are not in a one-to-one correspondence. The same CAD model may have multiple text descriptions (top). The same text description may correspond to different CAD models (bottom).

into CAD construction sequences. To enhance the model's generation capability, we construct two sets of class-specific prototypes from textual and visual modalities (as shown in Fig.1) using online updates and cumulative averaging. These prototypes enhance the framework's effectiveness in three aspects: (1) The prototype encodes comprehensive class-level information. By computing the cosine similarity between the input prompt and the learned prototypes, the model enables flexible and robust cross-modal alignment. (2) Given a unimodal input prompt, the prototype from the complementary modality can be leveraged to supply additional information, thereby enriching the multimodal feature representations and improving the generation of CAD construction sequences. (3) During generalization to new domains, the prototypes are dynamically updated to align with the target data distribution, thereby enhancing the model's generation capability in unseen domains.

To support the training of UniCAD, we construct a multimodal dataset comprising 100 categories of common CAD models. Specifically, it contains CAD construction sequences, text descriptions, and single-view images, with a total of 49,586 triples. In summary, our main contributions can be summarized as follows:

- We propose UniCAD, a unified framework capable of generating CAD construction sequences from flexible inputs, including images, text descriptions, or their combination, making it well-suited for deployment in real-world scenarios.
- Our multimodal prototype is capable of compensating for the missing modality information in unimodal inputs. Combined with a progressive training strategy, the model effectively learns the mapping from enriched multimodal features to CAD sequences, enabling robust sequence generation during inference.
- We introduce a new dataset comprising high-quality image-text-CAD triplets to promote advances in CAD

construction sequence generation from multimodal data and will make it publicly available in the future.

- Experiments demonstrate the effectiveness of our UniCAD framework on CAD construction sequence generation and completion tasks.

2. Related Work

Data representation for CAD In the field of 3D generative modeling, a variety of representation formats have been extensively studied, including point clouds [31, 40], 3D meshes [27, 41, 45], voxel grids [19, 29], and signed distance functions (SDFs) [6]. Unlike these representations, CAD models offer parameterized, editable, and manufacturable designs, making them essential in industrial 3D modeling and digital product development. Currently, CAD models are primarily represented in three forms: Constructive Solid Geometry (CSG), Boundary Representation (B-rep), and CAD construction sequences.

Specifically, CSG represents complex CAD models through Boolean operations (e.g., union, difference) applied to simple geometric primitives such as cubes and cylinders [14, 44]. B-rep describes CAD models using interconnected networks of faces, edges, and vertices, thereby capturing both geometric and topological relationships [8, 10, 13, 39, 46]. In contrast, CAD construction sequences can be parsed and executed by industry-standard CAD geometric kernels, enabling the direct generation of editable and parameterized CAD files that seamlessly integrate into real-world industrial design workflows. Its flexibility and reusability alleviate the workload for designers, thereby attracting substantial attention from the CAD community [15, 16, 21, 25, 34, 36, 37, 43]. These approaches can be broadly categorized into two groups: traditional methods and LLMs-based methods.

Traditional method for CAD generation The CAD community initially focused on unconditional modeling, employing transformer and Mamba-based architectures for CAD generation and reconstruction [21, 36, 38]. Although these methods have demonstrated promising results, they are unable to perform conditional CAD generation, which limits their applicability in practical design tasks. Recently, several approaches have introduced additional modalities—such as point clouds [9, 15, 25, 32], images [4, 11, 43], and text descriptions [2, 16, 22, 33]—as conditional signals for CAD sequence generation. Among these, contrastive learning-based, transformer-based, and diffusion-based models have emerged as dominant paradigms. However, most existing methods are designed for specific input modalities or tasks, which constrains their generalization capabilities across diverse representation formats and design scenarios. In this work, we propose UniCAD, a unified framework capable of generating or completing CAD construction sequences from diverse input modalities, including images, text, or their combinations.

LLMs for CAD generation LLMs have achieved remarkable success across various domains. Recently, the CAD community [33, 34, 47] has made initial attempts to leverage SOTA LLMs for the creation of CAD models. Specifically, the innovation of these works primarily lies in the development of domain-specific CAD representations that are compatible with LLMs, as well as in the design of effective fine-tuning strategies tailored to these representations. For instance, CADFusion [33] utilizes LLaMA3-8b as the backbone and both the sequential signal and visual signal for text-to-CAD tasks. CAD-GPT [34] fine-tunes the pre-trained LLaVA model [23] to enable multimodal CAD generation. Note that these methods introduce substantial computational overhead, typically requiring high-end hardware configurations such as GPU clusters with 16×H800 [37] or 4×A800 [34] GPUs. Although some works have explored challenges similar to ours, their reliance on specific data representations and high computational requirements consequently limits their practicality and scalability.

In contrast, we propose a lightweight framework based on structural innovations and efficient training strategies, offering novel and practical solutions to the aforementioned challenges. Our approach offers two key advantages: (1) UniCAD adopts a general command-and-parameter CAD construction sequence used by many CAD generators (e.g., GenCAD [1], Text2CAD [16], and CAD Translator [22]), and thus avoids model-specific serialized representations (e.g., LLM-oriented prompt formats). (2) UniCAD achieves comparable performance to LLMs, while significantly reducing both model parameters and inference time, making it more suitable for deployment in resource-constrained environments.

3. Method

As shown in Fig.3 (A), UniCAD consists of a multimodal encoder, a fusion layer, and a CAD construction sequence decoder (Section 3.1). We propose a progressive training strategy (Section 3.2), which initially incorporates the full CAD construction sequence and gradually masks out CAD tokens during training. This strategy enables the generation of CAD construction sequences from either single-modality prompts or their combinations. Additionally, two sets of class-level prototypes are constructed based on textual and visual modalities, which are presented in Section 3.3. When only a single modality is provided, the prototype from the other modality can be leveraged to supplement the input with complementary information (Fig.3 (B)). Furthermore, we present the domain adaptation process for textual prototypes on cross-domain datasets in Section 3.4. Finally, Section 3.5 introduces the loss function used for model training.

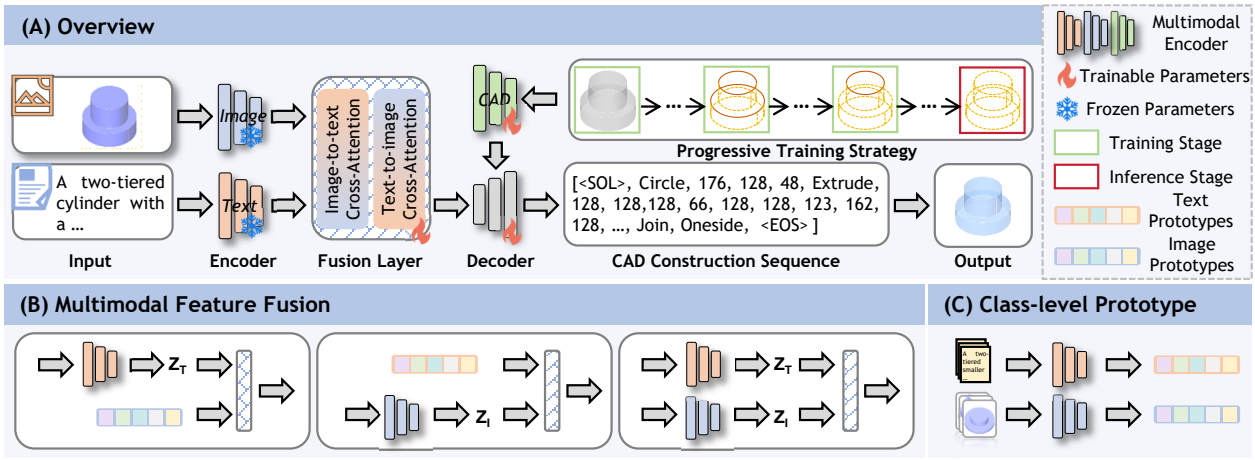

Figure 3. Overview of UniCAD. **Training:** images and texts are processed by Swin-S [24] and DistilBERT [30] to extract corresponding features, which are used to generate corresponding category predictions by utilizing multimodal prototypes. The fusion layer fuses cross-modal features by using strategy (B). CAD tokens are gradually replaced with masked tokens, according to a progressive training strategy (top row). Finally, multimodal features (image, text, CAD) are fed into the decoder to generate the CAD construction sequence. **Inference:** we use the cross-modality data (text, image, or both) and all masked CAD tokens to generate the CAD construction sequence.

3.1. Network Architecture

CAD construction sequence encoder For a CAD construction sequence S, which consist of commands S_c and parameters S_p, we follow the method in [36] to formulate it to an embedding $e_s \in \mathbb{R}^{N_s \times d_E}$:

$$e_s\left(i\right) = e_i^{(S_c)} + e_i^{(S_p)} + e_i^{pos}, \tag{1}$$

where $e_i^{(S_c)} = W_{S_C}\delta_i^c$, $e_i^{(S_p)} = W_{S_p}^2 f(W_{S_p}^1 \delta_i^p)$, $e_i^{pos} = W_{pos}\delta_i$. $e_i^{(S_c)}$ encodes the command type S_c, $e_i^{(S_p)}$ represents the embedding of the command parameters S_p, and e_i^{pos} serves as a positional encoding to capture the index of the i-th command within the full CAD construction sequence. The learnable parameters include: $W_{S_C} \in \mathbb{R}^{d_E \times 6}$, $W_{S_p}^1 \in \mathbb{R}^{d_E \times 257}$, $W_{S_p}^2 \in \mathbb{R}^{d_E \times 16 d_E}$, and $W_{pos} \in \mathbb{R}^{N_s \times d_E}$. The one-hot vectors $\delta_i^c \in \mathbb{R}^6$, $\delta_i^p \in \mathbb{R}^{257 \times 16}$, and $\delta_i \in \mathbb{R}^{N_s}$ correspond to the command type, parameter encoding, and positional index, respectively. Specifically, $\delta_i^c \in \mathbb{R}^6$ indicates one of the six command types. Each of the 16 command parameters is quantized into an 8-bit integer, and each bit is encoded as a one-hot vector of dimension $2^8 + 1 = 257$. These are then stacked into a matrix $\delta_i^p \in \mathbb{R}^{257 \times 16}$. The positional vector $\delta_i \in \mathbb{R}^{N_s}$ has a 1 at position i and 0 elsewhere. The function $f(\cdot)$ flattens its input matrix into a vector. Here, we set the embedding dimension d_E to 256.

Multimodal encoder We employ a dual-branch feature extraction architecture to process text or image inputs. For the textual modality, we utilize DistilBERT [30], a compact model derived from the original BERT architecture [7] via knowledge distillation techniques. Here, we denote the text encoder as Enc_T. Given an input textual description $T = (t_1, t_2, \ldots, t_{N_T})$, where t_i represents the i-th token in the sequence, we first tokenize T into a sequence of token IDs, which is then fed into Enc_T to generate contextual embeddings:

$$Z_T = \mathrm{Enc}_T(T) \in \mathbb{R}^{N_T \times d_T}, \tag{2}$$

where N_T denotes the number of tokens, and d_T is the embedding dimension. For visual inputs, we utilize the Swin Transformer-Small (Swin-S) [24] as the image encoder, denoted as Enc_I, to extract visual features from the input image I, resulting in a visual embedding matrix:

$$Z_I = \mathrm{Enc}_I(I) \in \mathbb{R}^{N_I \times d_I} \tag{3}$$

In our implementation, we set $N_T = N_I = 64$ and $d_T = d_I = 768$, ensuring consistent sequence lengths and embedding dimensions across both modalities.

Fusion layer After extracting the text features Z_T and image features Z_I, we feed them into a fusion layer to perform cross-modal feature interaction. This process enables the learned language features to become vision-aware and the visual features to become language-aware. Inspired by GLIP [20], we employ image-to-text and text-to-image cross-attention modules to facilitate feature fusion, aligning representations across modalities. As illustrated in Fig.3 (B), each attention head computes context vectors for one modality by incorporating information from the other. The fusion outputs A_{I2T} and A_{T2I}, which are concatenated and linearly transformed, enabling bidirectional contextual integration between modalities.

CAD construction sequence decoder Drawing inspiration from recent advances in Natural Language Processing (NLP) [7, 18], we employ a natural language generation

model as the decoder. The decoder architecture is based on BART [18], which takes as input both the sequence embeddings and the cross-modally fused features from the fusion layer to generate CAD construction sequences. The output features of the decoder are passed through a linear projection layer to predict the CAD construction sequence $S^* = [S_1^*, \cdots, S_{N_c}^*]$, where each token includes a command S_c^* and its corresponding parameters S_p^*.

3.2. Progressive Training Strategy

Generating CAD construction sequences from fully masked CAD tokens using multimodal features is a highly challenging task. This difficulty primarily arises from the absence of visible CAD tokens during the inference phase, which may hinder the model's ability to effectively retain and apply the modeling patterns learned during previous training stages, degrading overall generation performance.

To this end, we design a progressive training strategy inspired by curriculum learning [17, 35]. The strategy begins with training the model using complete CAD construction sequences, and gradually introduces masked CAD tokens as training progresses. Specifically, we introduce a mask token (i.e., [MASK]) into the CAD token sequence with probability γ, where the masking rate γ_t at training timestep t is defined as follows:

$$\gamma_t = \begin{cases} 0 & \text{if } 1 \le t \le t_w \\ \frac{t - t_w}{T} & \text{if } t_w < t \le T \end{cases} \tag{4}$$

where γ_t is set to 0 during the warm-up phase t_w, indicating that all CAD construction tokens are preserved. This initial stage enables the model to learn fundamental associations and feature representations across modalities. As training progresses, a portion of the CAD construction tokens is gradually masked, where the masked tokens are randomly sampled from the entire dataset. The masking rate γ_t increases progressively from 0 to 0.9 throughout the training process. The overall objective is to generate the corresponding CAD construction sequence by relying solely on multimodal prompts as input.

3.3. Multimodal Prototype Construction

Image prototypes construction To construct the image prototype $P_I(k)$ for a class k, we first utilize the image encoder Enc_I to obtain the corresponding visual representations $f^I \in \mathbb{R}^{N \times N_I \times d_I}$, where N is the number of samples belonging to category k in the training set. Then $P_I(k)$ is constructed by averaging these representations:

$$P_I(k) = \frac{1}{N} \sum_{i=1}^{N} f_i^I \tag{5}$$

Textual prototypes construction. Different from the image modality, text descriptions of the same object typically exhibit a long-tail distribution. Specifically, some common expressions are frequently used and account for the majority of the description samples, while a large number of other expressions are rarely adopted. This distribution may bias the prototype toward more frequent descriptions, while ignoring those less frequently used but equally important expressions, thereby failing to fully capture the object's comprehensive features. To address this, UniCAD employs a memory bank (MB) for each category to store evolving text prototypes over time. Specifically, the MB for each class k is initialized as $M_k = \{P_T(k) : \varnothing, N_k : 0\}$. Since the prototype $P_T(k)$ is non-parametric, we consider the following two scenarios to refine it iteratively: (1) If the text T is a new sample that describes the category k, we store it in a current list $\mathcal{B}_k$. (2) If T has already been used to calculate the prototype or already exists in the current list $\mathcal{B}_k$, we discard it and leave the current list unchanged. At the end of each training step, we update MB using the following equation:

$$P_T(k) \leftarrow \frac{N_k \cdot P_T(k) + \sum_{i \in \mathcal{B}_k} f_i^T}{N_k + |\mathcal{B}_k|}, \tag{6}$$

where $f^T \in \mathbb{R}^{N_T \times d_T}$ is the text feature of text T encoded by text encoder Enc_T. UniCAD maximizes the utilization of all descriptions to construct textual prototypes. This process ensures that each prototype continuously evolves in a balanced manner, mitigating the impact of the long-tail distribution.

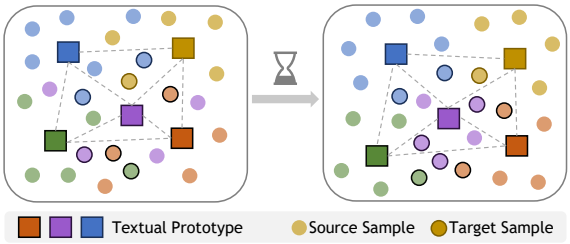

Figure 4. Textual Prototype Evolution. The prototypes of the source domain are adjusted and updated based on the features extracted from the target domain samples.

3.4. Cross-Domain Generalization

As previously discussed, a single CAD model may correspond to multiple textual descriptions. Notably, when different individuals describe the same object, they often exhibit significant variations in linguistic patterns, stylistic choices, and emphasis on specific attributes. When the styles of the source and target domains differ substantially, directly applying a model trained on the source domain to extract textual features in the target domain becomes highly challenging. The main reason is that high-dimensional textual features in the target domain are not adequately covered during training.

A straightforward approach is to utilize the object's im-

age prototype as input to substitute for unseen textual descriptions, as these representations have been effectively learned during model training. The acquisition of image prototypes depends on accurate class information, which is derived by computing the cosine similarity between the input text and the text prototypes. However, as the current text prototypes are learned from the source domain, directly applying them to the target domain results in unreliable class predictions. To address this issue, we propose to dynamically update the text prototypes to align with the data distribution of the target domain, as illustrated in Fig.4.

Algorithm 1 Inference in the Target Domain.

Input: textual prototype P_T, image prototype P_I, target data x_i with N_T, the source-trained text encoder Enc_T, the source-trained decoder Dec.
Output: CAD construction sequences S^*.

Cosine similarity is denoted as Cos.
for $i = 0$ **to** $N_T - 1$ **do**
  $f_i^T = Enc_T(x_i)$
  Category $c = argmax(Cos(\{f_i^T, P_T\}))$
  Obtain the $P_I(c)$ corresponding to c
  $S^* = Dec(P_I(c))$
end for
Return: S^*

Specifically, we propose a progressive adaptation strategy (PAS) for textual prototype updating in cross-domain generalization. As illustrated in Algorithm 1, different from the inference stage on the source domain, we obtain the sample class information by calculating the cosine similarity between the text and the class prototype. The corresponding image prototype is then searched as input to the decoder to obtain the CAD sequence. The prototype P_T for class k is given by:

$$P_T(k) \leftarrow \alpha P_T(k) + (1-\alpha)\frac{\sum_{i \in \mathcal{B}_k} f_i^T}{|\mathcal{B}_k|}, \tag{7}$$

where f_i^T represents the feature representation of the target domain. Let α denote the momentum coefficient, which is set to 0.9 in this work. The results of the quantitative experiments are presented in Section 4.7.

3.5. Loss Function

In this paper, we employ the Cross-Entropy loss L_{CE} to train the model. The loss is defined between the ground-truth CAD construction sequence $S = (S_c, S_p)$ and the predicted sequence $S^* = (S_c^*, S_p^*)$ as:

$$L_{CE} = \alpha \sum_{i=1}^{N_{s_c}} l\left(S_c,\ S_c^*\right) + \beta \sum_{i=1}^{N_{s_c}} \sum_{j=1}^{N_{s_p}} l\left(S_p,\ S_p^*\right) \tag{8}$$

where $l(\cdot,\cdot)$ denotes the cross-entropy function, N_{s_c} and N_{s_p} are the number of command and parameter tokens, respectively, and α and β are weighting coefficients used to balance the two terms ($\alpha = 1$, $\beta = 2$). In cases where certain commands are empty or specific parameters are unused in the ground-truth sequence, the corresponding terms in the summation are omitted from the loss computation.

4. Experiments

4.1. Dataset

The multimodal dataset used in this paper consists of three components: CAD construction sequences, text descriptions, and single-view images.

4.1.1 Data Collection

CAD construction sequences We utilize the large-scale DeepCAD dataset, which contains vectorized representations of CAD construction sequences. Specifically, the original files are parsed to extract key structural information, which is then converted into vectorized representations.

Single-view images To generate single-view images corresponding to the DeepCAD dataset, we utilize PythonOCC (the Python implementation of OpenCASCADE technology) running on an NVIDIA RTX 4090 GPU. The construction sequences are visualized to enable the capture of images with a clean white background.

Text description To generate high-quality textual descriptions, we employ the Vision-Language Model (VLM) Qwen-VL-Max to produce text descriptions from rendered images of CAD models. To further evaluate the generalization of our method in the text-to-CAD task, we leverage the other VLM, GLM-4v-Flash, to generate alternative descriptions for 1,000 CAD objects. These descriptions exhibit distinct linguistic styles compared to those produced by Qwen-VL-Max, simulating diverse real-world input variations. We conduct manual filtering to ensure the accuracy and consistency of the generated descriptions.

4.1.2 Dataset Process

The JSON files released by DeepCAD contain 242,456 CAD objects, while the vectorized files include 178,238 objects. From the original files, we generate 198,719 images, with 43,737 failures in image retrieval. Among these, 166,610 CAD objects overlap with the vectorized files, indicating that 11,628 objects from the vectorized files failed to produce corresponding images. Additionally, 32,109 images derived from the original files are not utilized for vectorization in DeepCAD. Upon inspecting these 32,109 images, we find the standard CAD model. Specifically, these defects include: (a) the presence of two or more parts in a

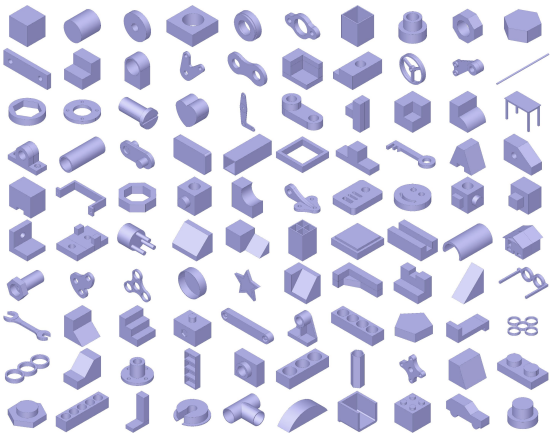

Figure 5. Representative samples from our dataset.

single object; and (b) irregularities or missing depth information. To ensure data quality and consistency, we adopt a manual approach for cleaning the dataset.

The cleaning criteria applied to the dataset are as follows: (a) CAD models consisting of two or more independent parts; (b) models with irregular geometry or lacking depth information; (c) text descriptions significantly differ from the rendered image; (d) images exhibiting excessive visual repetition. After applying these criteria, we obtain a final dataset of 126,992 text-image-CAD construction sequence triples, collected over four days. The representation format of the CAD construction sequence comprises a diverse set of commands and a large number of associated parameters. Over an additional four days, we select 49,586 representative samples across 100 categories, ensuring class diversity and sample quality. The representative samples are shown Fig.5.

4.2. Implementation Details

Baseline To validate the effectiveness of our approach, we evaluate its performance from two perspectives: text-to-CAD generation and image-to-CAD generation. We compare our method with several representative CAD generation techniques, including four traditional approaches and two LLM-based methods. Specifically, DeepCAD is a pioneering framework for generating CAD construction sequences and has been widely adopted in this field. Since it is originally an unconditional generation method, we build upon the baseline implementation proposed in Text2CAD [16] to extend it into a conditional generation framework that supports image or text inputs as conditions. In addition, we include two recent text-to-CAD models (CAD Translator [22] and Text2CAD [16]) and one image-to-CAD model (GenCAD [1]) in our evaluation. Since CAD Translator is not open-source, we carefully reimplement it based on the method described in [22]. For Text2CAD and GenCAD, we re-trained both models on the same training set as ours and evaluated them under the identical protocol to ensure a fair comparison. These methods, like ours, do not rely on large language models (LLMs), enabling a fair comparison in terms of model computational efficiency and generation quality. Given that existing LLM-based methods are closed-source in both data and code implementation, we are unable to provide direct qualitative comparisons. Therefore, we introduce two representative alternative models: Qwen 2.5-7B [28] and LLaMA3.1-8B, which are currently among the most advanced open-source large language models. To ensure a fair comparison, they adopt the same data representation and fine-tuning strategy. It is worth noting that, due to limited computational resources, we report results only for the text-to-CAD generation task.

Finally, we introduce a unimodal model based on UniCAD, named SinCAD. We replace the original multimodal input with a unimodal input, do not adopt the prototype-based strategy, and keep all other settings unchanged. This allows us to further investigate the effectiveness of the prototype learning strategy.

Details for UniCAD The experiments are conducted on an NVIDIA 4090 GPU, with a batch size of 256 and a total of 200 training epochs. The Adam optimizer is used, with a learning rate of 0.001. The dataset is split into training, validation, and test sets in an 8:1:1 ratio.

Details for LLM-based method Due to the complexity of the current data representation, it does not conform to the prompt paradigm commonly used for fine-tuning LLMs. To address this, we further convert the current data representation into a textual sequence format. Specifically, we first map numerical commands to semantically meaningful terms according to the following rule: {0: Line, 1: Arc, 2: Circle, 3: EOS, 4: SOL, 5: Extrude}. For the parameter part, we retain only the non-zero elements and compress the two-dimensional matrix into a one-dimensional vector. Based on this, we construct prompt data in the form of instruction-response pairs, which is suitable for fine-tuning LLMs. These models are configured with a maximum sequence length of 384. For efficient fine-tuning, we employ low-rank adaptation (LoRA) [12], with hyperparameters set to $r = 8$, $\alpha = 32$, and $dropout = 0.1$. During training, the batch size and gradient accumulation steps are both set to 4, and the learning rate is set to 1×10^{-4}. The model is trained for a total of 3 epochs, with checkpoints saved every 100 training steps. Consistent with our main experiments, all models are trained on the same GPU.

Metrics For quantitative evaluations, we adopt two types of metrics: one based on generated CAD construction sequences and the other based on point-cloud data transformed from CAD construction sequences. Command Accuracy (Acc_c) and Parameter Accuracy (Acc_p) are used to measure the accuracy of predicted CAD construction sequence S^* =(S_c^*, S_p^*), respectively. We also report the

Table 1. Comparison with conditioned CAD model generation methods. Acc_c and Acc_p are both multiplied by 100%. JSD, MMD, and COV are multiplied by 10^2. IR is an Invalid Ratio multiplied by 100%. SinCAD is trained by single-modal data (i.e., text or image).

Model	Venue	Acc_c ↑	Acc_p ↑	IR ↓	COV ↑	MMD ↓	JSD ↓	Params*(M) ↓	Params+(M) ↓	Inference(s) ↓
Text Input										
DeepCAD	ICCV'21	84.84	52.25	21.35	4.60	9.99	52.98	43.37	1.30	0.05
CAD Translator	MM'24	73.37	52.83	78.35	62.13	8.26	28.33	65.43	23.36	0.13
Text2CAD	NeurIPS'24	-	-	50.53	-	-	-	48.26	6.19	0.15
LLaMA3.1-8B	arXiv'24	88.73	64.04	1.79	11.60	4.66	12.75	8051.23	20.97	3.06
Qwen2.5-7B	arXiv'25	88.85	63.64	3.70	13.07	4.62	12.01	7635.80	20.19	4.42
SinCAD	-	96.76	61.69	7.96	26.30	3.69	26.30	46.28	4.21	0.06
UniCAD	-	98.64	64.03	3.53	30.70	2.69	16.59	46.28	4.21	0.09
Image Input										
DeepCAD	ICCV'21	90.18	57.01	28.41	6.70	7.76	42.97	52.24	2.23	0.06
GenCAD	TMLR'25	90.19	55.17	11.68	33.60	4.17	9.94	34.92	23.14	1.02
SinCAD	-	98.47	70.05	5.24	46.99	2.07	8.97	52.55	2.54	0.08
UniCAD	-	98.70	70.08	4.77	45.60	2.31	8.95	52.55	2.54	0.11
Multimodal Input										
UniCAD	-	98.71	70.18	4.94	48.80	2.30	8.98	95.15	5.07	0.08

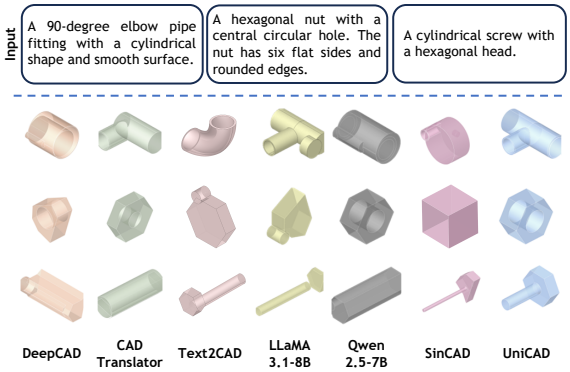

Figure 6. Qualitative comparison for image input.

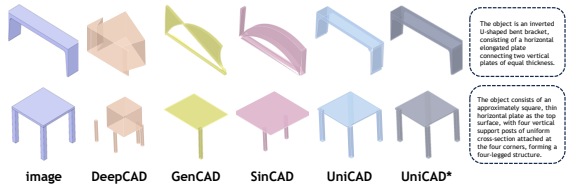

Figure 7. Qualitative comparison for image input. $*$ represents the result of multimodal input (i.e., image and text)

Invalid Ratio (IR), defined as the percentage of generated CAD models that fail to be converted into valid point clouds. Using the reconstructed point clouds, we further evaluate shape quality by sampling 2,000 points from each generated and ground-truth shape and comparing the resulting point sets [36, 38]. Specifically, Coverage (COV) measures the percentage of ground-truth shapes that can be closely approximated by at least one shape from the generated set. The Jensen–Shannon Divergence (JSD) quantifies the distributional difference between the generated and ground-truth point clouds. Finally, to assess the geometric alignment between a real and predicted shape, we compute the Minimum Matching Distance (MMD).

4.3. CAD Construction Sequence Generation

Quantitative comparison As shown in Table 1, UniCAD achieves the best performance across multiple metrics in the text-to-CAD task. Specifically, UniCAD achieves a 36.92% improvement in JSD and a 55.65% improvement in IR compared to SinCAD, demonstrating that the text modality can compensate for the missing structural information by leveraging input from the image prototype. In terms of model efficiency, UniCAD and SinCAD have the same number of parameters, as the incorporation of prototypes does not introduce any additional learnable parameters. However, UniCAD incurs a slight increase in inference time due to the need to compute the distance to class prototypes during inference for category prediction and corresponding text prototype retrieval. Since Text2CAD employs a different data representation method than our approach, we only evaluate its effectiveness. Notably, nearly 80% of the outputs generated by CAD Translator and 50% of those generated by Text2CAD cannot be converted into valid CAD models, whereas our method achieves an IR of only 3.53%. Although CAD Translator achieves better performance in reconstruction quality (e.g., COV), this metric is computed only over a subset of valid CAD models, leading to potentially overestimated results. The primary reason is that CAD Translator excels at learning the one-to-one mapping relationship in the latent space, while the class-level prototypes in UniCAD contain comprehensive information about the corresponding class. Moreover, UniCAD achieves comparable Acc_p scores to LLM-based methods, while offering significantly lower inference latency and fewer model parameters. This enables UniCAD to achieve a more favorable trade-off between generation quality and computational efficiency, making it better suited for deployment in resource-limited settings.

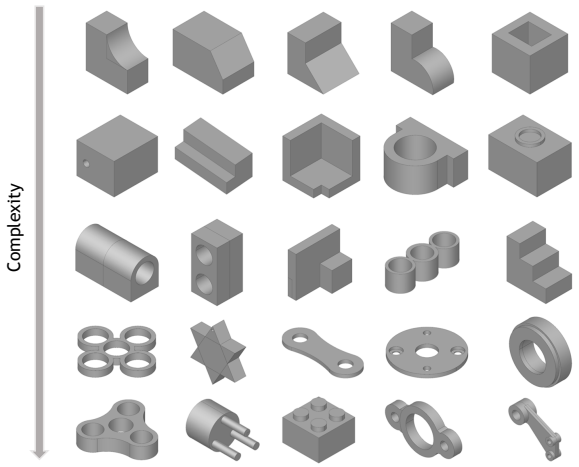

Figure 8. More generated CAD models from our method.

For the image-to-CAD task, UniCAD drops by 8.97% in IR compared to SinCAD, indicating that the missing semantic information in the image modality is effectively supplemented by the text prototype. Compared with GenCAD, our approach achieves superior performance across multiple metrics. Moreover, our method requires significantly less inference time, achieving a 92.16% reduction in latency. The primary reason lies in the diffusion-based architecture of GenCAD, which requires a sequential denoising process from Gaussian noise across a predefined number of time steps to reconstruct CAD objects. In contrast, our decoder employs a feed-forward strategy [3], which generates the complete CAD operation sequence in a single forward pass, thereby requiring significantly less inference latency.

In addition, image-based prompts can provide more detailed geometric and structural information of objects compared to text-based prompts, thereby facilitating the generation of more accurate CAD construction sequences. Specifically, image-guided UniCAD achieves a 9.45% improvement in Acc_p over text-guided UniCAD. Furthermore, image-guided and multimodal-guided UniCAD exhibit nearly identical accuracy and invalid generation rates, indicating that the semantic information provided by text prototypes is both accurate and well-preserved.

Qualitative comparison As shown in Fig.6 and 7, we can draw the following two interesting conclusions. Specifically, CAD models generated based on text descriptions can usually only present the basic shape and structure of an object, lacking details such as length, angle, and position. However, compared with the other three methods, the shape generated by UniCAD is closer to ground truth (GT), indicating that the image prototype with structured knowledge has been successfully integrated with the text features. Meanwhile, several key features of the 3D shapes generated by DeepCAD exhibit notable deviations from the GT, indicating a significant gap between multimodal inputs and the

Table 2. Results of text-guided CAD completion task.

Model	Acc_c (↑)	Acc_p (↑)	IR (↓)	COV (↑)	MMD (↓)	JSD (↓)
UniCAD						
20%CAD	99.06	70.06	3.39	63.80	1.63	7.27
40%CAD	99.59	77.26	2.92	69.70	1.33	5.79
60%CAD	99.79	82.88	3.13	74.29	1.15	5.69
80%CAD	99.84	85.09	2.66	73.90	1.13	5.47
CAD Translator						
20%CAD	78.28	55.34	69.48	45.60	1.93	6.37
40%CAD	84.26	56.51	55.74	50.99	1.65	6.02
60%CAD	89.19	57.32	41.20	52.30	1.63	6.04
80%CAD	88.80	57.20	42.36	42.93	2.18	5.72

Table 3. Results of image-guided CAD completion task.

Model	Acc_c (↑)	Acc_p (↑)	IR (↓)	COV (↑)	MMD (↓)	JSD (↓)
UniCAD						
20%CAD	99.48	73.48	2.40	60.50	1.53	6.98
40%CAD	99.70	78.93	2.28	68.90	1.29	5.71
60%CAD	99.79	82.62	2.38	74.40	1.10	5.37
80%CAD	99.84	85.10	2.36	74.69	1.10	5.38
SinCAD						
20%CAD	98.30	72.12	2.94	31.00	2.65	11.23
40%CAD	99.48	78.72	3.55	38.70	1.37	8.09
60%CAD	99.74	81.03	3.63	40.40	2.41	7.19
80%CAD	99.91	84.69	3.79	39.67	2.18	6.31

corresponding CAD construction sequences. Furthermore, we provide additional visualization results demonstrating the generation of CAD construction sequences from multimodal prompts. As shown in Fig.8, UniCAD achieves accurate and robust generation not only for basic and simple models but also for complex CAD structures.

4.4. CAD Construction Sequence Completion

Following [22], we input incomplete CAD construction sequences (i.e., partially masked CAD tokens) along with multimodal prompts (image, text, or both) to reconstruct the complete sequence. For example, "20% CAD" indicates that 20% of the CAD tokens are retained while 80% are masked. We define more challenging completion levels (20% to 80% token retention) to simulate greater sequence incompleteness, compared to the range used in CAD Translator (60% to 80%). We evaluate the performance of CAD completion under various modality-conditioned settings and report the results in Tables 2 to 4, respectively.

As shown in Table 2, even with only 20% of the CAD tokens available, our model achieves over 70% in Accp when reconstructing CAD sequences, representing a substantial improvement over CAD-Translator. This demonstrates that UniCAD can effectively extract key features from mul-

Table 4. Results of multimodal-guided CAD completion task.

Model	Acc_c (↑)	Acc_p (↑)	IR (↓)	COV (↑)	MMD (↓)	JSD (↓)
20%CAD	99.48	76.62	3.39	66.00	1.50	6.11
40%CAD	99.70	80.38	2.98	73.20	1.19	5.63
60%CAD	99.76	83.06	2.90	73.80	1.16	5.64
80%CAD	99.83	84.82	2.58	72.80	1.06	5.77

Table 5. Quantitative comparison for the ablation study.

Model	Acc_c (↑)	Acc_p (↑)	IR (↓)	JSD (↓)	MMD (↓)	COV (↑)
$w/o\ Pro$ (T)	96.32	61.07	9.62	21.74	3.79	24.10
$w/o\ AD$ (T)	97.57	62.38	6.27	18.68	3.19	26.80
$w/o\ Mask$ (T)	30.64	29.67	99.92	88.68	34.85	33.33
Ours (T)	98.64	64.03	3.53	16.59	2.69	30.70
$w/o\ Pro$ (I)	98.32	68.42	5.85	10.17	2.33	41.89
$w/o\ AD$ (I)	96.91	67.26	9.27	10.73	2.18	45.80
$w/o\ Mask$ (I)	30.89	32.52	99.88	88.47	36.97	50.00
Ours (I)	98.70	70.08	4.77	9.57	2.27	47.99

timodal inputs and utilize them to reconstruct complete CAD construction sequences. Moreover, as the number of masked tokens decreases, UniCAD achieves higher reconstruction accuracy. Specifically, with 80% of tokens visible, the model improves Accp by 21.45% and IR by 21.53% compared to the 20% visibility setting. These results validate that our progressive training strategy effectively mitigates catastrophic forgetting of previously knowledge.

As shown in Table 3, SinCAD and UniCAD achieve comparable generation results, indicating that prototype-based features do not provide significant advantages when explicit structural information is available. In contrast, during the generation phase, where access to valid CAD sequences is limited or unavailable, prototype features become more valuable due to the lack of structural guidance.

Under all level settings, the reconstruction accuracy of utilizing text prompts and image prompts is almost identical, which differs from the results of the generation task. A key reason is that CAD tokens can provide fine-grained structural information, which is already included in the image content. Consequently, the performance gain from CAD tokens is more limited for the image-based approach compared to the text-based method.

4.5. Ablation Study

As shown in Table 5, the performance of UniCAD employing the prototype strategy has improved, regardless of whether the input is text or image. Notably, the IR has increased by 63.31% and 18.46% for text and image inputs, respectively. This indicates that the multimodal prototype strategy not only complements missing information from another modality but also promotes consistent representations in the latent space. In addition, UniCAD (T) introduces the adaptive layer to fuse the cross-modality feature, resulting in a significant performance improvement of 2.64 % in Acc_p and 43.70% in IR. UniCAD ($w/oMask$) is almost unable to generate some valid CAD models, indicating that the model overfits the mapping relationship between multimodal data and complete CAD tokens, rendering it incapable of generating valid results at the inference stage.

4.6. Evaluation of the Progressive Training Strategy

During training, the mask rate γ_t gradually increases from 0 to 0.9. To further investigate the effectiveness of γ_t, we incrementally increase γ_t from 0.9 to 1.0, referred to as Criteria†, as presented in Eq.9. In addition, the model is further trained for 20 epochs using Criteria‡, which comprises multimodal features and all masked CAD tokens. The implementation details of Criteria‡ are shown in Eq.10.

$$\gamma_t^\dagger = \begin{cases} 0 & \text{if } 1 \le t \le t_w \\ \frac{t-t_w}{t_\delta} & \text{if } t_w < t \le t_\delta \\ 0.9 + \frac{t-t_\delta}{T-t_\delta} \times 0.1 & \text{if } t_\delta < t \le T_1, \end{cases} \tag{9}$$

$$\gamma_t^\ddagger = \begin{cases} 0 & \text{if } 1 \le t \le t_w \\ \frac{t-t_w}{t_\delta} & \text{if } t_w < t \le t_\delta \\ 0.9 + \frac{t-t_\delta}{t_\eta-t_\delta} \times 0.1 & \text{if } t_\delta < t < t_\eta \\ 1 & \text{if } t_\eta \le t \le T_2, \end{cases} \tag{10}$$

where t_δ, T_1, t_η, and T_2 are set to 200, 240, 240, and 260 respectively. The quantitative comparison of different masking rate strategies is shown in Table 6 and the qualitative comparison is shown in Fig.9. Criteria‡ outperforms both Criteria† and our strategy in terms of the accuracy of Command and Parameter. However, it fails to produce any meaningful result with an invalid ratio of 100. UniCAD consistently achieves the best results among all completion tasks. A key reason is that Criteria† and Criteria‡ overfit the masked CAD tokens and catastrophically forget the feature mappings between different modalities.

4.7. Cross-Domain Generalization

As previously described in Section 3.4, we construct a cross-domain experiment to simulate personalized language expressions in real-world scenarios. To this end, we adopt the ChatGLM-V4 model to generate textual descriptions, which differs from the source domain. Specifically, we build a small-scale dataset containing 30 CAD objects and a total of 1,000 textual descriptions. Among them, 500 samples from the target domain are used to incrementally update the text prototypes, while the remaining samples are held out as an independent test set to evaluate the model's final performance on the target domain.

We use the un-updated prototype as the baseline (i.e., $N_T = 0$), which directly takes the target data as input to

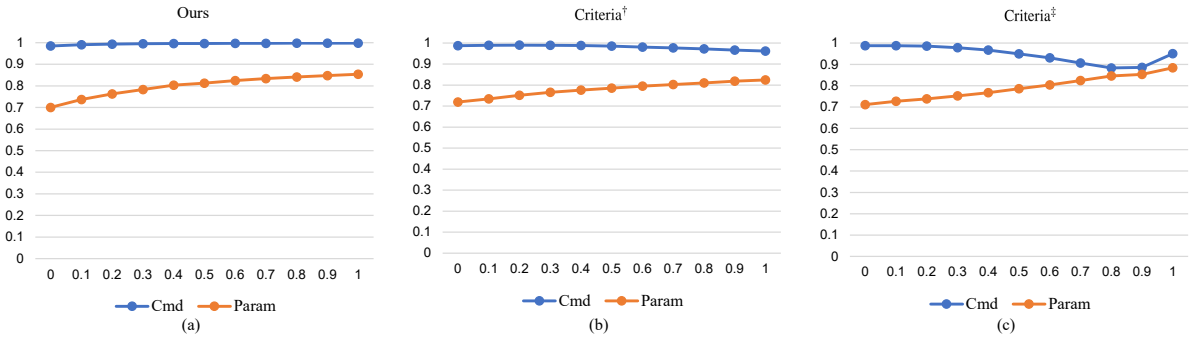

Figure 9. Quantitative experiment between our strategy and other criteria for the image to CAD sequence generation and completion. The vertical axis represents the accuracy rates predicted by command and parameter, while the horizontal axis represents the value of γ_t. Here, $\gamma_t = 0$ corresponds to a generation task, and values of γ_t ranging from 0.1 to 0.9 represent completion tasks.

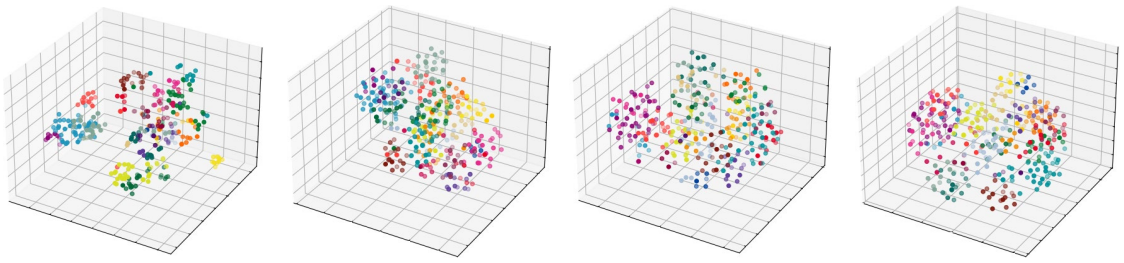

Figure 10. T-SNE visualizations of text features in the target domain. N_T represents the number of target domain samples used for updating the prototypes. As the samples in the target domain continue to increase, the text prototype will be dynamically adjusted to gradually adapt to the features of the target domain, thereby enhancing the model's performance in new fields.

Table 6. Quantitative comparison on the image to CAD sequence generation and completion. * indicates the generation task, while the others represent completion tasks.

Input	Strategy	$Acc_c(\uparrow)$	$Acc_p(\uparrow)$	IR ($\downarrow$)
0% CAD*	Criteria†	98.74	70.94	4.07
	Criteria‡	98.76	71.15	100
	Ours	98.47	70.06	5.24
20% CAD	Criteria†	98.98	75.14	4.42
	Criteria‡	98.54	73.83	6.97
	Ours	99.33	76.34	3.58
40% CAD	Criteria†	98.82	77.62	100
	Criteria‡	96.67	76.75	15.99
	Ours	99.64	80.04	3.04

obtain the corresponding CAD sequence. In contrast, other approaches (e.g., $N_T = 100$) employ PAS to update the text prototypes. In Fig.10, we utilize t-SNE to visualize the continuous evolution of text prototypes within the target domain and highlight the stored features from 25 random classes using different colors. These visualizations illustrate that the text prototypes are constantly adjusted based on the features of the target domain samples, gradually adapting to the characteristics of the target domain.

Table 7. Quantitative comparison results. N_T equals 0 indicates that the source domain prototype has not been updated.

N_T	Acc_c ($\uparrow$)	Acc_p ($\uparrow$)	COV ($\uparrow$)	MMD ($\downarrow$)	JSD ($\downarrow$)
0	97.95	58.96	41.37	1.28	4.05
100	98.25	60.66	69.99	1.15	4.68
300	98.25	62.52	69.99	0.88	4.83
500	98.31	63.90	89.99	0.87	4.39

As shown in Table 7, all PAS-based methods outperform baseline (i.e., $N_T = 0$). The main reason is that there is a significant domain gap between the source domain and the target domain. By contrast, other PAS-based methods can effectively reduce the impact of the domain gap. The model ($N_T = 500$) achieves the best results, proving the effectiveness of PAS and Algorithm 1. Figure 11 shows that descriptions of the same object in the source domain and the target domain exhibit completely different styles and characteristics. UniCAD consistently generates accurate CAD models in most cases, which is mainly attributed to the fact that our prototype can dynamically update its class-level center to adapt to the data distribution of the target domain.

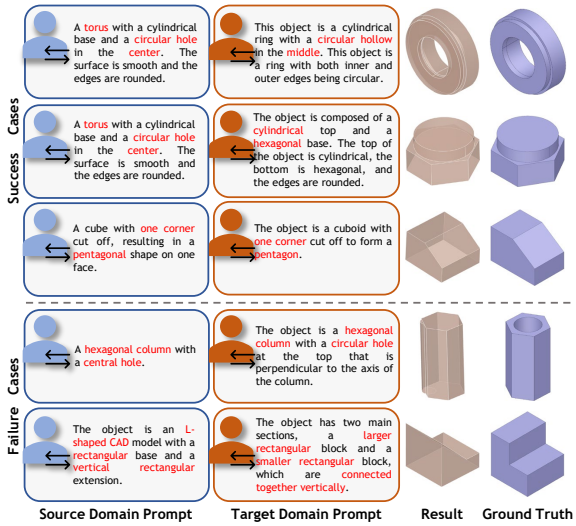

Figure 11. Qualitative results of UniCAD on cross-domain text-to-CAD construction sequence generation.

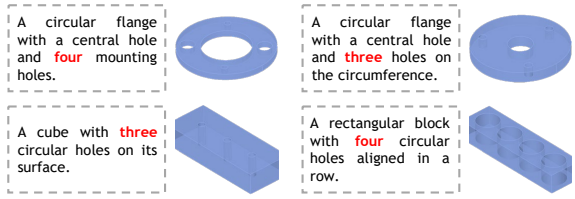

Figure 12. The potential of the model in controllable CAD generation.

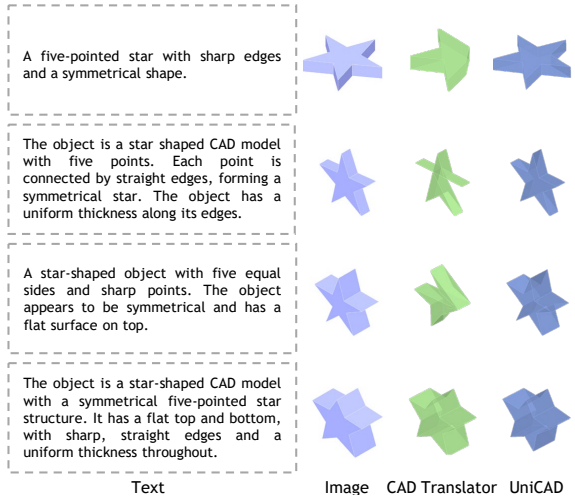

Figure 13. The potential of the model in handling diverse data.

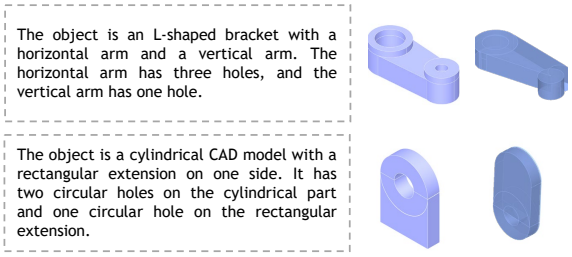

Figure 14. Some failure cases for our UniCAD. The complex objects often consist of multiple primitives and require longer sequences that exceed setting in this paper.

4.8. Applications of UniCAD

In this section, we discuss the practicality of UniCAD from two perspectives. (1) *Controllable CAD generation.* When we input the keywords 'three' and 'four' in the text, the model will obtain the corresponding results, as illustrated in Fig.12. This indicates that UniCAD not only supports multimodal input but also effectively interprets textual content, enabling more natural and flexible human-computer interaction. (2) *The diversity of text descriptions.* When users describe the same object, there may be considerable differences in their language expressions. As shown in Fig.13, UniCAD accurately generates the same object under multiple text descriptions, demonstrating that the model can effectively learn the essential features and key information of the object through our prototype strategy.

4.9. Limitations

Despite the promising results obtained by our UniCAD, there are still some failure cases, as shown in Fig.14. Several factors contribute to these failures. First, although we set the length of CAD construction sequences to 64, complex objects often consist of multiple primitives and require longer sequences that exceed this limit. Meanwhile, text descriptions cannot accurately reflect details such as lengths and angles specified in CAD construction sequences, leading to discrepancies between the CAD models generated by UniCAD (text only) and the labels. Furthermore, the DeepCAD dataset, which forms the basis of our training data, is unbalanced, resulting in sparse samples for certain categories and potentially causing generation failures.

5. Conclusion

In this paper, we propose UniCAD, a novel prototype-enhanced unified framework for generating CAD construction sequences from images or text descriptions. UniCAD significantly improves design efficiency and lowers entry barriers for CAD modeling. Specifically, our multimodal prototypes can compensate for the missing information in

unimodal inputs, thereby improving the quality of CAD model generation. Furthermore, the textual prototypes can dynamically evolve to support the generation of CAD objects from diverse text descriptions. Extensive experiments demonstrate the superiority of UniCAD and validate the effectiveness of each proposed strategy. In the future, it is also worthwhile to develop a more complex CAD dataset, which contains detailed multimodal annotations, to enable finer-grained control over CAD model generation.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (No. 62572212), Science and Technology Development Plan of Jilin Province (No. 20260203049SF) and the Fundamental Research Funds for the Central Universities.

References

[1] M. F. Alam and F. Ahmed. Gencad: Image-conditioned computer-aided design generation with transformer-based contrastive representation and diffusion priors. *arXiv preprint arXiv:2409.16294*, 2024. 2, 3, 7

[2] A. Badagabettu, S. S. Yarlagadda, and A. B. Farimani. Query2cad: Generating cad models using natural language queries. *arXiv preprint arXiv:2406.00144*, 2024. 3

[3] A. Carlier, M. Danelljan, A. Alahi, and R. Timofte. Deepsvg: a hierarchical generative network for vector graphics animation. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020. 9

[4] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11073–11082, June 2025. 3

[5] T. Chen, C. Yu, Y. Hu, J. Li, T. Xu, R. Cao, L. Zhu, Y. Zang, Y. Zhang, Z. Li, and L. Sun. Img2cad: Conditioned 3d cad model generation from single image with structured visual geometry. *arXiv preprint arXiv:2410.03417*, 2024. 2

[6] G. Chou, Y. Bahat, and F. Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. In *Proceedings of the IEEE International Conference on Computer Vision*, 2023. 3

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019. 4

[8] E. Dupont, K. Cherenkova, A. Kacem, S. A. Ali, I. Arzhannikov, G. Gusev, and D. Aouada. Cadops-net: Jointly learning cad operation types and steps from boundary-representations. In *Proceedings of the International Conference on 3D Vision*, pages 114–123, 2022. 3

[9] E. Dupont, K. Cherenkova, D. Mallis, G. Gusev, A. Kacem, and D. Aouada. Transcad: A hierarchical transformer for cad sequence inference from point clouds. In *European Conference on Computer Vision*, pages 19–36, 2024. 3

[10] H. Guo, S. Liu, H. Pan, Y. Liu, X. Tong, and B. Guo. Complexgen: Cad reconstruction by b-rep chain complex generation. *ACM Transactions on Graphics*, 41(4), 2022. 3

[11] E. Hong, M. H. Nguyen, M. A. Uy, and M. Sung. Mv2cyl: Reconstructing 3d extrusion cylinders from multi-view images. *arXiv preprint arXiv:2406.10853*, 2024. 3

[12] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al. Lora: Low-rank adaptation of large language models. *International Conference on Learning Representations*, 1(2):3, 2022. 7

[13] P. K. Jayaraman, A. Sanghi, J. G. Lambourne, K. D. Willis, T. Davies, H. Shayani, and N. Morris. Uv-net: Learning from boundary representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11698–11707, 2021. 3

[14] K. Kania, M. Zieba, and T. Kajdanowicz. Ucsg-net-unsupervised discovering of constructive solid geometry tree. In *Advances in Neural Information Processing Systems*, pages 8776–8786, 2020. 3

[15] M. S. Khan, E. Dupont, S. A. Ali, K. Cherenkova, A. Kacem, and D. Aouada. Cad-signet: Cad language inference from point clouds using layer-wise sketch instance guided attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4713–4722, 2024. 3

[16] M. S. Khan, S. Sinha, S. T. Uddin, D. Stricker, S. A. Ali, and M. Z. Afzal. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. In *Advances in Neural Information Processing Systems*, 2024. 2, 3, 7

[17] Y. Kong, L. Liu, J. Wang, and D. Tao. Adaptive curriculum learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5047–5056, 2021. 5

[18] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020. 4, 5

[19] J. Li, K. Xu, S. Chaudhuri, E. Yumer, H. Zhang, and L. Guibas. Grass: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics*, 36(4):1–14, 2017. 3

[20] L. H. Li, P. Zhang, H. Zhang, J. Yang, C. Li, Y. Zhong, L. Wang, L. Yuan, L. Zhang, J.-N. Hwang, K.-W. Chang, and J. Gao. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 4

[21] X. Li, Y. Lou, Y. Song, and X. Zhou. Mamba-cad: State space model for 3d computer-aided design generative modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. 3

[22] X. Li, Y. Song, Y. Lou, and X. Zhou. Cad translator: An effective drive for text to 3d parametric computer-aided design generative modeling. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 8461–8470, 2024. 2, 3, 7, 9

[23] H. Liu, C. Li, Y. Li, and Y. J. Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 26286–26296, 2024. 3

[24] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9992–10002, 2021. 4

[25] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 27144–27153, 2024. 1, 3

[26] W. Ma, M. Xu, X. Li, and X. Zhou. Multicad: Contrastive representation learning for multi-modal 3d computer-aided design models. In *Proceedings of the ACM International Conference on Information and Knowledge Management*, pages 1766–1776, 2023. 2

[27] C. Nash, Y. Ganin, S. M. A. Eslami, and P. W. Battaglia. Polygen: an autoregressive generative model of 3d meshes. In *Proceedings of the 37th International Conference on Machine Learning*, 2020. 3

[28] Qwen, :, A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, H. Lin, J. Yang, J. Tu, J. Zhang, J. Yang, J. Yang, J. Zhou, J. Lin, K. Dang, K. Lu, K. Bao, K. Yang, L. Yu, M. Li, M. Xue, P. Zhang, Q. Zhu, R. Men, R. Lin, T. Li, T. Tang, T. Xia, X. Ren, X. Ren, Y. Fan, Y. Su, Y. Zhang, Y. Wan, Y. Liu, Z. Cui, Z. Zhang, and Z. Qiu. Qwen2.5 technical report, 2025. 7

[29] X. Ren, J. Huang, X. Zeng, K. Museth, S. Fidler, and F. Williams. Xcube: Large-scale 3d generative modeling using sparse voxel hierarchies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4209–4219, 2024. 3

[30] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2020. 4

[31] Y. Sun, Y. Wang, Z. Liu, J. E. Siegel, and S. E. Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 61–70, 2018. 3

[32] M. A. Uy, Y.-Y. Chang, M. Sung, P. Goel, J. Lambourne, T. Birdal, and L. Guibas. Point2cyl: Reverse engineering 3d objects from point clouds to extrusion cylinders. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11840–11850, 2022. 3

[33] R. Wang, Y. Yuan, S. Sun, and J. Bian. Text-to-cad generation through infusing visual feedback in large language models, 2025. 3

[34] 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, 2024. 3

[35] X. Wang, Y. Chen, and W. Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, 2022. 5

[36] R. Wu, C. Xiao, and C. Zheng. Deepcad: A deep generative network for computer-aided design models. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 1, 2, 3, 4, 8

[37] J. Xu, C. Wang, Z. Zhao, W. Liu, Y. Ma, and S. Gao. Cad-mllm: Unifying multimodality-conditioned cad generation with mllm, 2025. 3

[38] X. Xu, P. K. Jayaraman, J. G. Lambourne, K. D. Willis, and Y. Furukawa. Hierarchical neural coding for controllable cad model generation. In *Proceedings of the 40th International Conference on Machine Learning*, 2023. 3, 8

[39] X. Xu, J. Lambourne, P. Jayaraman, Z. Wang, K. Willis, and Y. Furukawa. Brepgen: A b-rep generative diffusion model with structured latent geometry. *ACM Transactions on Graphics*, 43(4), 2024. 3

[40] G. Yang, X. Huang, Z. Hao, M.-Y. Liu, S. Belongie, and B. Hariharan. Pointflow: A point cloud generation network with sequential convolutions and dynamic pooling. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019. 3

[41] X. Yang, F. Yu, Y. Zhou, P. Zhou, G. Zhao, and Z. Guan. A method of surface mesh generation for industrial cad models by constructing conforming discrete representation. *Computer-Aided Design*, 188:103914, 2025. 3

[42] M. Yavartanoo, S. Hong, R. Neshatavar, and K. M. Lee. Text2cad: Text to 3d cad generation via technical drawings. *arXiv preprint arXiv:2411.06206*, 2024. 2

[43] Y. You, M. A. Uy, J. Han, R. Thomas, H. Zhang, S. You, and L. Guibas. Img2cad: Reverse engineering 3d cad models from images through vlm-assisted conditional factorization. *arXiv preprint arXiv:2408.01437*, 2024. 3

[44] F. Yu, Z. Chen, M. Li, A. Sanghi, H. Shayani, A. Mahdavi-Amiri, and H. Zhang. Capri-net: Learning compact cad shapes with adaptive primitive assembly. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11758–11768, 2022. 3

[45] K. Yu, B. Wang, X. Chen, Y. He, and J. Chen. Minimal surface-guided higher-order mesh generation for cad models. *Computer-Aided Design*, 178:103810, 2025. 3

[46] C. Zhang, A. Polette, R. Pinquié, G. Carasi, H. De Charnace, and J.-P. Pernot. ecad-net: Editable parametric cad models reconstruction from dumb b-rep models using deep neural networks. *Computer-Aided Design*, 178:103806, 2025. 3

[47] Z. Zhang, S. Sun, W. Wang, D. Cai, and J. Bian. Flexcad: Unified and versatile controllable cad generation with fine-tuned large language models. In *Proceedings of the International Conference on Learning Representations*, 2025. 3