

CADKnitter: Compositional CAD Generation from Text and Geometry Guidance

Tri Le¹, Khang Nguyen², Baoru Huang³, Tung D. Ta⁴, and Anh Nguyen³

¹FPT Software AI Center, Vietnam

²Department of Robotics, Mohamed bin Zayed University of Artificial Intelligence (MBZUAI), UAE

³Department of Computer Science, University of Liverpool, UK ⁴Department of Creative Informatics, The University of Tokyo, Japan

<https://cadknitter.github.io/>

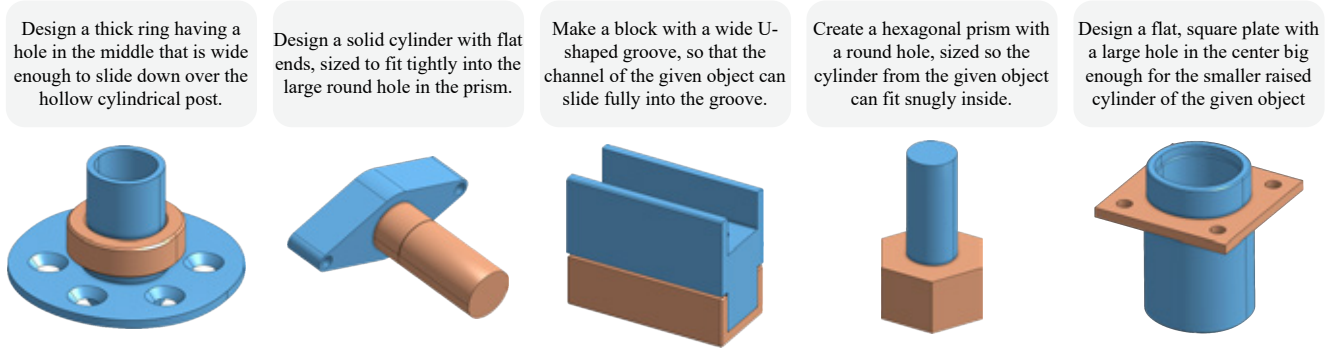


Figure 1. Compositional CAD Generation. The CADKnitter model takes in a **text prompt** and an **existing CAD model** to generate a **complementary CAD model** that geometrically fits with the input CAD and semantically aligns with the design prompt.

Abstract

Crafting computer-aided design (CAD) models has long been a painstaking and time-intensive task, demanding both precision and expertise from designers. With the emergence of 3D generation, this task has undergone a transformative impact, shifting not only from visual fidelity to functional utility but also enabling editable CAD designs. Prior works have achieved early success in single-part CAD generation, which is not well-suited for real-world applications, as multiple parts need to be assembled under semantic and geometric constraints. In this paper, we propose CAD-Knitter, a compositional CAD generation framework with a geometry-guided diffusion sampling strategy. CADKnitter is able to generate a complementary CAD part that follows both the geometric constraints of the given CAD model and the semantic constraints of the desired design text prompt. We also curate a dataset, so-called KniCAD, containing over 310,000 samples of CAD models, along with textual prompts and assembly metadata that provide semantic and geometric constraints. Intensive experiments demonstrate that our proposed method outperforms other state-of-the-art baselines by a clear margin.

1. Introduction

3D content generation is shifting beyond visual fidelity to focus on the functional utility of generated objects. Many studies target the physical plausibility of generated shapes [4, 25, 52], while others explore compositional generation [3, 62] or synthesize complementary parts that geometrically align with existing ones [33, 53]. As the field shifts toward functionality, *computer-aided design (CAD)* generation is gaining attention in research and industry [23, 26] because CAD models are precise and editable parametric representations, making them directly suitable for real-world design and manufacturing tasks across product design, mechanical engineering [34], and robotics [36, 53]. Practically, CAD demands not only the creation of individual parts but also meticulous precision in how distinct parts interact, align, and assemble [17, 56]. To support this need, we enhance the utility of CAD generation by making it aware of existing CAD models while considering both semantic and geometric constraints. Semantic constraints refer to the functional and contextual relationships between parts, ensuring that the components align with the user-intended design (e.g., “generating a bolt to fasten a provided nut”). Meanwhile, geometric compliance defines the spatial and structural relationships that govern how the generated CAD parts fit and align with the existing ones.

Recent efforts have shown promising results in unconditional CAD generation [21, 61], text-conditioned generation [18, 51], image-based reconstruction [26, 58], and point cloud-based rendering [29]. Despite these advancements, existing generation methods primarily focus on *single CAD generation* and overlook the complex inter-dependencies between multiple parts. In practice, CAD not only requires a single model; it typically involves an assembly of multiple parts subjected to strict geometric and semantic constraints [17, 49, 56]. Moreover, the current state of CAD generation is inapplicable to real assemblies due to its object-centric nature, lack of assembly awareness, and geometry-guided mechanisms for enforcing how such components should be connected and function together.

In this paper, we propose *CADKnitter*, a method for enhancing the CAD design process. Different from traditional text-to-CAD approaches, our method takes an input CAD model and a text prompt to generate a new CAD model that aligns with the given inputs. The generated CAD model should be consistent with the input text prompt while maintaining geometric compatibility with the existing CAD model, as shown in Fig. 1. Our *CADKnitter* introduces a geometry-guided conditional diffusion model that explicitly enforces assembly compatibility during generation by using geometric cues from optimizing contact faces between the generated and conditional parts. Building upon prior text-to-CAD approaches that focus solely on single-object fidelity, *CADKnitter* explicitly models inter-part relationships, enabling it to synthesize components that fit and function together within real-world assemblies. We further construct *KnitCAD*, a large-scale dataset comprising text-CAD pairs with detailed assembly metadata that enables scalable learning under semantic-geometric constraints. The intensive experiments show that our method outperforms recent baselines. In summary, our contributions are threefold:

- We introduce *KnitCAD*, a large-scale dataset for the compositional CAD generation of over 310,000 text-CAD pairs with detailed assembly metadata and automatically annotated contact faces.
- We propose a geometry-guided conditional diffusion model that enforces geometric compatibility, while preserving the synthesis of CAD components that semantically assemble with the given parts.
- We empirically demonstrate that our model significantly improves assembly accuracy and semantic fidelity compared to state-of-the-art baselines.

2. Related Work

CAD Generation. Many methods have been proposed for CAD generation, such as text to CAD [18, 51, 58], image to CAD [2, 26, 65], or point cloud CAD rendering [10, 29]. Among these works, several methods represent a CAD model as a sequence of sketch and extru-

sion operations, generating it in an auto-regressive fashion [18, 51, 57, 60, 69]. Other work [35, 45, 66] focuses on generating Constructive Solid Geometry, which represents 3D shapes through hierarchies of Boolean operations and basic primitive shapes. While these representations are flexible for generative models, their capability to present complex CAD models is limited [16, 24, 67]. Instead, the predominant format for CAD models is Boundary-representation (B-rep) [1, 21, 48, 54]. B-rep presents a CAD model by combining the continuous geometry and discrete topology of primitives, which is challenging for current generative models to learn for both data types. To synthesize B-rep models directly, several works [16, 22, 61] propose cascaded generative models, while others [11, 21, 30] explore unified representations that encode both types of data. In this work, we use B-rep representation due to its ability to model real-world CAD models.

Compositional Shape Generation. Recent research has investigated the synthesis of object parts through different representation inputs, such as images and point clouds [3, 27, 28, 62, 64]. However, these typically focus on the semantic relationship and overlook the fine-grained geometric compatibility between distinct parts. More recent efforts explore the paradigm of generating complementary components that geometrically fit with existing objects. For example, PhysPart [33] utilizes cascaded generative models to synthesize physically consistent mesh parts, enabling articulated interactions. Fit2Form [12] addresses assembly feasibility and contact modeling, which relies solely on geometric conditioning and cannot capture semantic intent or design context. Similar to our goal, MatchMaker [53] is a multi-stage framework that produces CAD parts satisfying assembly constraints from given parts. Herein, our work addresses compositional CAD generation, guided by both semantic descriptions from input text prompts and geometric compatibility constraints from provided CAD models that adhere to real-world functional design.

Geometry-Guided Sampling. Several guided sampling strategies [4, 7, 37, 38, 52, 59, 63, 68] have been proposed to improve the generation of diffusion models. PhysDiff [68] leverages a physics-based motion projection module to adjust the intermediate steps of the reverse process. DiffuseBot [52] leverages gradients from a differentiable simulation to improve the physical utility for task-specific robot generation. PhyScene [63] integrates physical and interactivity guidance to generate physically interactable 3D scenes. Drawing insights from these works, we employ geometry-guided sampling to enforce generation according to the geometric information of the existing CAD input.

3. The *KnitCAD* Dataset

Our proposed *KnitCAD* dataset builds upon two recent datasets: (i) Fusion 360 Gallery Assembly – Joint Data

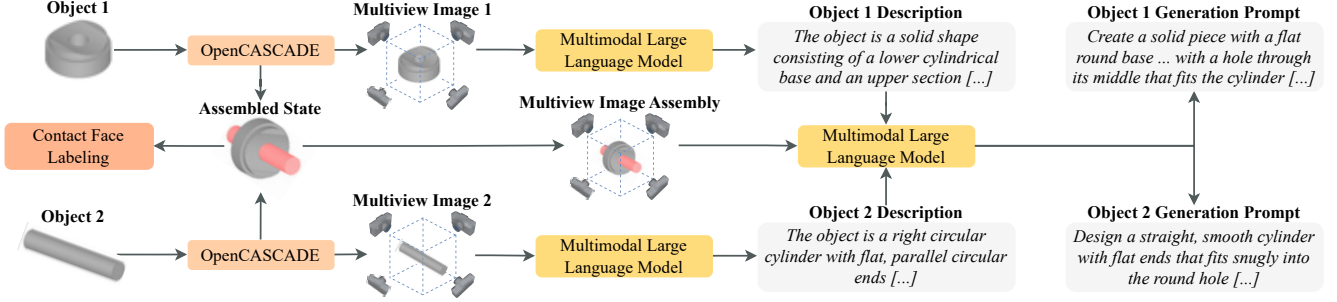


Figure 2. Construction of *KnitCAD*. The dataset is curated from prior CAD datasets. For each pair of B-rep models, OpenCASCADE [40] is used to render multiview images and label contact faces in their fully assembled states. Multimodal LLM generates textual descriptions of shapes and prompts for generation from rendered images.

(Fusion 360 Joint) [56] contains 19,156 joint pairs from 23,029 B-rep models, and (ii) Automate [17] includes 541,635 mate pairs from 376,362 B-rep models. These datasets provide detailed joint-axis annotations that fully describe the connection between two CAD parts, making them directly relevant to our compositional CAD generation task under geometric constraints and text descriptions.

Specifically, we extend the original datasets [17, 56] by generating text descriptions for each pair using state-of-the-art multi-modal large language models (MLLM) and by labeling the contact faces of every B-rep model with OpenCASCADE [40]. Here, a contact face (*i.e.*, a bounded surface in B-rep representation) is defined as a face that is within a small distance tolerance of a face on the complementary CAD model. These contact-face labels provide important geometric information, indicating exactly how the parts interact in the assembled state. The comparisons among the attributes of open-sourced CAD datasets and our curated *KnitCAD* dataset are illustrated in Table 1.

3.1. Generating Prompts for CAD Generation

For each pair of B-rep models, we synthesize two generation prompts that describe the target CAD model and how they are semantically complementary to each other, as shown in Fig. 2. We first render multi-view images of each part and their assembled configuration using OpenCASCADE [40]. These rendered views are then fed into an MLLM to produce textual descriptions for each object; meanwhile, another MLLM subsequently fuses the individual object descriptions to form paired assembly prompts that capture the semantic intent. Note that we employ GPT-4.1 [41] as the main MLLM for all text generation.

For the Fusion 360 Joint dataset, we select one representative joint per model pair, while for Automate, duplicate mate definitions between identical model pairs are removed to ensure uniqueness. In total, we curate 156,654 unique assembled pairs derived from 172,265 distinct B-rep models. Therefore, our dataset contains 313,308 samples, each of which includes a target CAD model, a condition CAD

Dataset	Attribute	No. Samples	Representation	Object Pair Label	Contact Face Label	Text
ABC [19]		1M	B-rep	✗	✗	✗
DeepCAD [57]		178K	B-rep	✗	✗	✗
Fusion 360 Joint [56]		19K	B-rep	✓	✓	✗
Automate [17]		542K	B-rep	✓	✗	✗
2BY2 [43]		517	Mesh	✓	✗	✗
ATA [49]		8800	Mesh	✓	✗	✗
Factory [36]		60	B-rep	✓	✗	✗
Text2CAD [18]		170K	B-rep	✗	✗	✓
CADfusion [51]		20K	B-rep	✗	✗	✓
Omni-CAD [58]		453K	B-rep	✗	✗	✓
<i>KnitCAD</i> (Ours)		157K	B-rep	✓	✓	✓

Table 1. Dataset Attributes. The attributes of the existing and our *KnitCAD* dataset, in terms of number of samples, type of representation, object pairs, contact face data, and text descriptions.

model, and a textual prompt for learning CAD generation.

3.2. Contact Face Conditions and Labeling

To enable compositional CAD generation, we use OpenCASCADE [40] to label the contact faces between two CAD models. Two faces are defined to be in contact if they have common curvatures that overlap or are within a small tolerance of each other. We uniformly sample a discretized face $s = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{N_s}\}$ of N_s points from a parametric surface, where each point $\mathbf{p}_i \in \mathbb{R}^3$, with $N_s = 32 \times 32$ [15]. The conditions for overlapping or being within a small tolerance between a point $\mathbf{p} \in s$ and a face s' are mathematically described by the following properties:

- (i) **Point-to-Point Proximity:** $\|\mathbf{p} - \mathbf{q}\|_2 \leq \delta$, where $\mathbf{q} \in s'$ is defined as the closest point to $\mathbf{p}$ and δ represents the tolerance threshold,
- (ii) **Inverted Normals:** $\mathbf{n}(\mathbf{p}) \cdot \mathbf{n}(\text{Pr}_{s'}(\mathbf{p})) < 0$, where $\text{Pr}_{s'}(\mathbf{p}) \in \mathbb{R}^3$ denotes the projection of $\mathbf{p}$ onto the parametric surface of s' and $\mathbf{n}(\cdot) \in \mathbb{R}^3$ represents the face normal vector at the specified point.

Therefore, two faces s_1 and s_2 are said to be in contact if a subset of points $\mathbf{p}_{i,1} \in s_1$ satisfies both conditions with respect to the face s_2 , or vice versa. These serve as explicit conditions for Compositional CAD generation in our dataset.

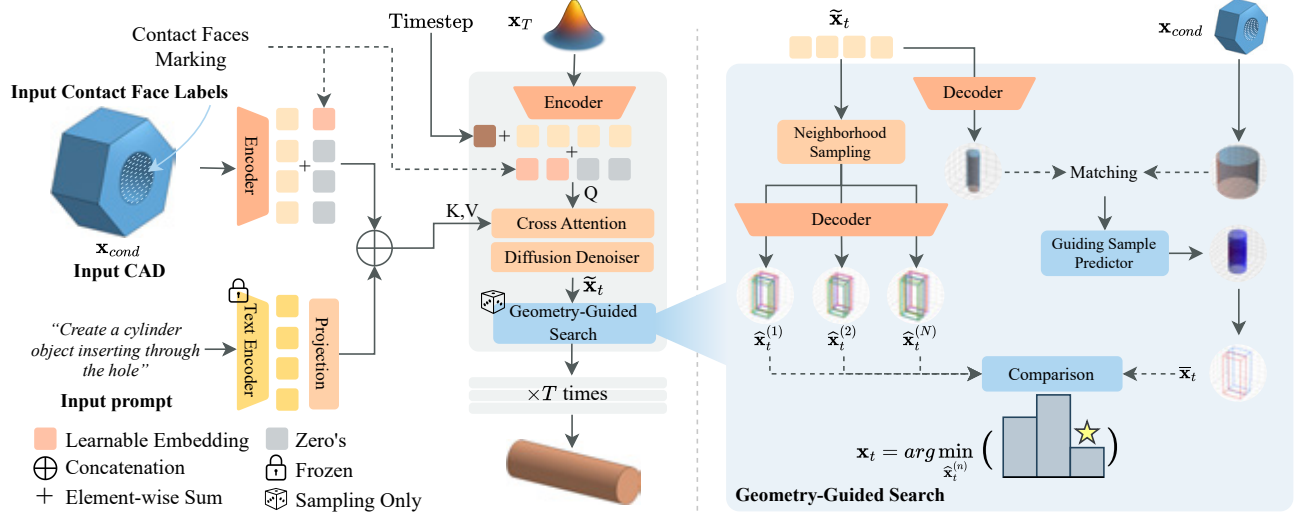


Figure 3. Overview of CADKnitter. Given a conditional CAD model with desired contact faces and a text prompt, our method first encodes them using corresponding encoders. These embeddings thus condition a **diffusion-based CAD generator**. At specific reverse steps, **Geometry-Guided Search** refines intermediate samples by sampling a neighborhood of candidate samples, decoding face geometry, and selecting samples mostly aligned with approximate geometric cues from **Guiding-Sample Predictor**.

4. Compositional CAD Generation

4.1. Problem Formulation

A B-rep model consists of geometric elements (faces, edges, and vertices) with pairwise topological relationships (face-edge, edge-vertex adjacency matrices) [61]. In our work, we write a B-rep model as a set of face entities $\mathbf{x} = \{x^{(i)}\}_{i=1}^N$, where each element $x^{(i)} \in \mathbb{R}^{D_x}$ represents a B-rep face geometric encoding. It is noted that D_x can differ from one another depending on whether it is a bounding box [61], UV-sampled geometric points [15], or a latent representation [11, 21, 30]. The edges, vertices, and the topological information are represented based on x_i as they can be implicitly unified as once in x_i [21, 30] or explicitly provided by other sets and adjacency matrices [22, 61]. For brevity, we omit the explicit formulation.

Given a text prompt $\mathcal{T}$, a B-rep model $\mathbf{x}_{\text{cond}}$, and a set of indices $\mathcal{I} \subseteq \{1, 2, \dots, N\}$ for labeling the desired contact faces, compositional CAD generation aims to learn a model f_θ parameterized by θ that is able to generate the complementary CAD $\mathbf{x}'$ satisfying both semantic and geometric constraints learned from a dataset $\mathcal{D}$:

$$\mathbf{x}' \leftarrow f_\theta(\mathcal{T}, \mathbf{x}_{\text{cond}}, \mathcal{I} \mid \mathcal{D}). \quad (1)$$

In Eq. 1, the input set of indices is used to identify the subset of B-rep entities, representing the desired contact faces in $\mathbf{x}_{\text{cond}}$ based on $\mathcal{I}$. We divide the problem into two parts: designing a conditional diffusion-based CAD generation model to learn the distribution of semantic data (Sec. 4.2) and devising a geometry-guided sampling mechanism to enforce generation that follows geometric constraints (Sec. 4.3).

4.2. Diffusion-based CAD Generation

We adopt a denoising diffusion probabilistic model (DDPM) [13, 47] to generate complementary B-rep models $\mathbf{x}'$, conditioned on a text prompt $\mathcal{T}$ and an existing B-rep model $\mathbf{x}_{\text{cond}}$, where the data distribution is learned by reversing a forward Gaussian noise process over T timesteps. The forward process is $q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t} \mathbf{x}_0, (1 - \alpha_t) \mathbf{I})$, where $\alpha_t = \prod_{s=1}^t \alpha_s$ is the cumulative product of the noise schedule. A noise predictor ϵ_θ is trained to predict the added noise at step t , using both text and geometry-based conditions in our dataset $\mathcal{D}$ with the loss $L_{\text{MSE}} = \mathbb{E}_{t, \mathbf{x}_0, \epsilon} [\|\epsilon - \epsilon_\theta(\mathbf{x}_t, t, \mathcal{T}, \mathbf{x}_{\text{cond}})\|_2^2]$.

In Fig. 3, for semantic conditioning, we encode the text prompt using a pre-trained language model followed by a linear projection. For geometric conditioning, each element in the conditional B-rep is embedded using a dedicated encoder. To emphasize face-level assembly cues, we inject learnable embeddings at elements corresponding to annotated contact faces in the conditional input. Formally, we denote the text embeddings as $\mathbf{t} \in \mathbb{R}^{N_{\text{text}} \times D}$ and the conditional B-rep embeddings as $\mathbf{e}_c \in \mathbb{R}^{N_c \times D}$, where D is the hidden dimension. The two are concatenated into a single conditioning sequence $\mathbf{c} = [\mathbf{t}; \mathbf{e}_c]$ of total length $N_{\text{text}} + N_c$. For the generated B-rep, we predefine a set of M elements associated with contact faces and similarly enhance their embeddings with shared, learnable tokens.

The condition sequence $\mathbf{c}$ is fused with the noisy latent representation $\mathbf{x}_t$ via a cross-attention mechanism prior to denoising. During inference, at each timestep t , a diffusion denoiser predicts $\tilde{\mathbf{x}}_t$ through a reverse process $p_\theta(\tilde{\mathbf{x}}_t \mid \mathbf{x}_{t+1}, \mathbf{c})$ which leverages the trained noise predictor ϵ_θ [13, 46]. To further enforce geometric compatibility, we

introduce a geometry-guided sampling strategy that adjusts intermediate predictions $\tilde{\mathbf{x}}_t$ to conform better to contact face (geometric) constraints in the following section (Sec. 4.3).

4.3. Geometry-Guided Sampling

Compositional CAD generation requires precise contact between specified faces of the conditional and generated CAD models; however, diffusion models lack mechanisms to enforce such geometric constraints due to their stochastic nature and denoising approximation errors. Inspired by prior work on guided diffusion [14, 59, 68], at specific sampling steps, we use a predicted guiding signal to search for direction toward geometric-compatibility outcomes and update the intermediate steps accordingly. Our key idea is to approximate geometric cues by optimizing the geometry of generated faces with relaxed face-level constraints. The overall guided sampling procedure is outlined in Alg. 1.

Algorithm 1: Geometry-Guided Sampling

Require: Sample $\mathbf{x}_{t+1}$ at time $t + 1$, condition $\mathbf{c}$, guiding-sample predictor $\mathcal{G}_\pi$.

- 1: Sample denoised output $\tilde{\mathbf{x}}_t \sim p_\theta(\tilde{\mathbf{x}}_t \mid \mathbf{x}_{t+1}, \mathbf{c})$.
- 2: **if** guidance is performed at time t **then**
- 3: *# Predict geometric guiding sample*
- 4: $\tilde{\mathbf{x}}_t \leftarrow \mathcal{G}_\pi(\tilde{\mathbf{x}}_t, t, \mathbf{c})$
- 5: *# Neighborhood sampling*
- 6: $\{\hat{\mathbf{x}}_t^{(n)}\}_{n=1}^{N_p} \sim p_\theta(\hat{\mathbf{x}}_t \mid \tilde{\mathbf{x}}_t, \mathbf{c})$
- 7: *# Compare the neighbors with the guiding sample*
- 8: $n^* \leftarrow \arg \min_{1 \leq n \leq N_p} [D_{\text{geo}}(\hat{\mathbf{x}}_t^{(n)}, \tilde{\mathbf{x}}_t) + \omega_u D_{\text{reg}}(\hat{\mathbf{x}}_t^{(n)}, \tilde{\mathbf{x}}_t)]$
- 9: $\mathbf{x}_t \leftarrow \hat{\mathbf{x}}_t^{(n^*)}$
- 10: **else**
- 11: $\mathbf{x}_t \leftarrow \tilde{\mathbf{x}}_t$
- 12: **end if**

4.3.1. Geometry-Guided Search

Leveraging the blend of diffusion and energy-based models [7], we incorporate geometry-guided search at specific timesteps of the reverse diffusion process during inference. Specifically, for an intermediate sample $\tilde{\mathbf{x}}_t$ produced by the denoiser, we generate a set of neighborhood candidates $\{\hat{\mathbf{x}}_t^{(n)}\}_{n=1}^{N_p}$ using Unadjusted Langevin Dynamics (ULA) [6, 39], a form of MCMC sampling [7, 52], (*i.e.*, get N_p samples from $p_\theta(\hat{\mathbf{x}}_t \mid \tilde{\mathbf{x}}_t, \mathbf{c})$). Each candidate is then evaluated using a composite score that comprises a geometry fitness term to encourage precise contact alignment with the conditional B-rep model, and a regularization term that preserves semantic consistency with the text prompt. The candidate with the lowest score is chosen to update the intermediate sample, guiding the diffusion trajectory

toward geometry-aware and semantically coherent generations, similar to Zero-Order Optimization [9], which approximates gradients toward a desired space by using neighborhood samples.

For the geometry fitness term, we measure the mean minimum discrepancy between the neighborhood candidate $\hat{\mathbf{x}}_t^{(n)}$ and the guiding sample $\tilde{\mathbf{x}}_t$, which is obtained from $\mathcal{G}_\pi$ (described next). Let $c_i, c'_j \in \mathbb{R}^3$ denote the center coordinates, and $d_i, d'_j \in \mathbb{R}^3$ denote the aspect dimensions of the bounding boxes associated with $x^{(i)} \in \hat{\mathbf{x}}_t^{(n)}$ and $x^{(j)} \in \tilde{\mathbf{x}}_t$, respectively. The geometry fitness term D_{geo} is mathematically defined as:

$$D_{\text{geo}}(\hat{\mathbf{x}}_t^{(n)}, \tilde{\mathbf{x}}_t) = \mathbb{E}_{x^{(j)} \in \tilde{\mathbf{x}}_t} \left[\min_{x^{(i)} \in \hat{\mathbf{x}}_t^{(n)}} \phi(x^{(i)}, x^{(j)}) \right], \quad (2)$$

where $\phi(x^{(i)}, x^{(j)}) = \|c_i - c'_j\|_2 + \|d_i - d'_j\|_2$.

To preserve the semantic alignment of the generated samples, we regularize them through the Fused Gromov–Wasserstein (FGW) distance [50] to quantify scale-invariant structural similarity between the sampling candidate $\hat{\mathbf{x}}_t^{(n)}$ and the intermediate sample $\tilde{\mathbf{x}}_t$. As shown in Fig. 3, the bounding box aspect ratio $r_i \in \mathbb{R}^3$ is a feature of element $x^{(i)} \in \hat{\mathbf{x}}_t^{(n)}, \tilde{\mathbf{x}}_t$, while the scale-invariant distances are measured between bounding box centers. Let $\mathbf{C}, \mathbf{C}'$ and r, r' denote the normalized centers and aspect ratios of bounding boxes in two sets, respectively. The FGW distance between $\hat{\mathbf{x}}_t^{(n)}$ and $\tilde{\mathbf{x}}_t$ is:

$$D_{\text{reg}}(\hat{\mathbf{x}}_t^{(n)}, \tilde{\mathbf{x}}_t) = (1 - \lambda) \sum_{i, i', j, j'} W_{i, i', j, j'}^2 T_{ij} T_{i'j'} + \lambda \sum_{i, j} \|r_i - r'_j\|_2^2 T_{ij} \quad (3)$$

$$W_{i, i', j, j'} = \|\mathbf{C}_i - \mathbf{C}_{i'}\|_2 - \|\mathbf{C}'_j - \mathbf{C}'_{j'}\|_2,$$

where $\lambda \in [0, 1]$ is predefined and T represents the transport plan, which is optimized using the algorithm in [42].

4.3.2. Guiding-Sample Predictor

The guiding-sample predictor $\mathcal{G}_\pi$ returns a set of optimized contact faces of the generated CAD model $\tilde{\mathbf{x}}_t$, providing approximate geometric cues for the search stage. We denote $\mathbf{S}_t = \{s_t^i\}_{i=1}^M$ and $\mathbf{S}_c = \{s_c^j\}_{j=1}^{|Z|}$ as the sets of contact faces on the generated and condition CAD models, respectively, where each s_t^i in either face is defined similarly to Sec. 3.2 and $\mathbf{S}_t$ is decoded from intermediate samples $\tilde{\mathbf{x}}_t$. While defining exact contact face constraints between two CAD models is nontrivial [17], we relax this problem as one-to-one face optimization with position and shape objectives.

We establish one-to-one correspondences between faces in the two sets by matching them to their closest faces. Namely, we use the Hungarian matching algorithm [20],

where the cost matrix is defined by the point-to-mesh distance [8, 44] of every face pair. For each matched pair (s_t^i, s_c^i) , we optimize the translation and edges of s_t^i to align with s_c^i by minimizing the following cost function:

$$\mathcal{C} = \lambda_{\text{pos}}(t) \mathcal{C}_{\text{pos}} + \lambda_{\text{shape}}(t) \mathcal{C}_{\text{shape}}, \quad (4)$$

where $\mathcal{C}_{\text{pos}}$ is the positional cost function, $\mathcal{C}_{\text{shape}}$ denotes the shape cost function, and $\lambda_{\text{pos}}(t)$ and $\lambda_{\text{shape}}(t)$ are time-dependent weight functions, as we observe that the positional structures of B-rep elements tend to form during earlier generation steps, while fine-grained shapes emerge in later steps.

As the first term of Eq. 4, the positional cost encourages each generated face s_t^i to minimize its distance to the corresponding conditional face s_c^i . Formally:

$$\mathcal{C}_{\text{pos}}(s_t^i, s_c^i) = \min_{p \in s_t^i} \min_{\tau \in T(s_c^i)} d(p, \tau), \quad (5)$$

where $d(p, \tau)$ denotes the point-to-mesh distance function from a point p on s_t^i to the triangle τ tessellated from the triangle sets $T(\cdot)$ of s_c^i . Gaining insights from [56], which notes the importance of edges in predicting the joint axis between two CAD models, we define the shape cost function, the second term of Eq. 4, as a combination of the weighted costs of edge lengths and edge angles:

$$\mathcal{C}_{\text{shape}} = \lambda_{\text{len}} \mathcal{C}_{\text{len}} + \lambda_{\text{angle}} \mathcal{C}_{\text{angle}}. \quad (6)$$

In Eq. 6, the edge length term is to measure the cost between the edge lengths of s_t^i and s_c^i with $N_e = \sqrt{N_s} = 32$:

$$\mathcal{C}_{\text{len}}(e, e') = \frac{1}{N_e - 1} \sum_{i=1}^{N_e-1} (\|e_{i+1} - e_i\|_2 - \|e'_{i+1} - e'_i\|_2)^2,$$

where e and e' denote the sampled boundary-edge points from s_t^i and s_c^i , respectively. While the cost function for edge angles is computed as:

$$\mathcal{C}_{\text{angle}}(e, e') = \frac{1}{N_e - 1} \sum_{i=1}^{N_e-1} (1 - \mathbf{u}_i \cdot \mathbf{u}'_i),$$

where $\mathbf{u}_i = (e_{i+1} - e_i) / \|e_{i+1} - e_i\|_2$ and $\mathbf{u}'_i = (e'_{i+1} - e'_i) / \|e'_{i+1} - e'_i\|_2$ denote unit direction vectors of the i -th edge segments on s_t^i and s_c^i , respectively. More in the Supplemental Material.

5. Experiments

5.1. Baselines & Experimental Setups

Baselines. We evaluate our proposed method with the similar work, MatchMaker [53]. As MatchMaker [53] lacks public codebases, we implement and extend it with text conditioning. Besides, we compare our method with

3D shape generation methods. As demonstrated in prior works [2, 26], methods trained with implicit representations of 3D shapes often produce outputs that are unsuitable for reconstructing CAD models. We compare our method with a recent direct mesh generation method, namely PivotMesh [55]. For PivotMesh, we adapt it to incorporate additional conditions. Further, we also compare our method with and without geometry-guided sampling. We detail all the baselines in our Supplementary Material.

Evaluation Protocol & Metrics. We use BrepGen [61] as the diffusion denoiser, representing each face by its bounding box (*i.e.*, the first stage) and decoding the face by using the face generator (*i.e.*, the second stage in the BrepGen pipeline). Our method and all baselines are trained and evaluated on our proposed *KnitCAD* dataset. We then evaluate our method on 200 random test samples, generating 16 B-rep models per sample and reporting averaged metrics over the successfully built generated CAD models. Following prior works [16, 18, 33, 61], we report the following evaluation metrics: (1) Chamfer Distance (CD) is used to measure the geometric alignment with the ground truth; (2) Intersection Volume Percentage (IV) quantifies interpenetration under the condition of CAD models; (3) Proximity (PR) computes the average distance between corresponding contact faces (in $\times 10$ mm); and (4) Valid Ratio (VR) calculates the percentage of watertight B-rep generation. For PivotMesh, we only report CD, as it directly generates the mesh. More in the Supplemental Material.

5.2. Quantitative Results

Method	CD ↓	PR ↓	IV ↓	VR ↑
PivotMesh [55]	137.70	-	-	-
MatchMaker [53]	102.15	0.48	18.95	0.29
Ours (without guidance)	88.69	0.24	9.42	0.45
Ours (with guidance)	86.03	0.23	6.90	0.44

Table 2. Quantitative Results. Our method’s performance compared to other baselines [53, 55] in terms of defined metrics.

Table 7 compares MatchMaker [53], PivotMesh [55], and our method with and without guidance. Our approach achieves better overall performance on defined metrics with lower CD, lower PR, lower IV, and higher VR, particularly when guided sampling is applied. The guidance introduces an empirical trade-off in which VR is approximately 2% lower, while IV improves by approximately 27%, indicating a substantially better satisfaction of geometric constraints with a modest reduction in plausibility.

5.3. Human Evaluations & Qualitative Results

We perform user preference studies with 15 users and 750 comparisons to PivotMesh and MatchMaker. Participants are tasked with annotating which of two generated CAD models (or tie) is (1) more semantically aligned with the in-

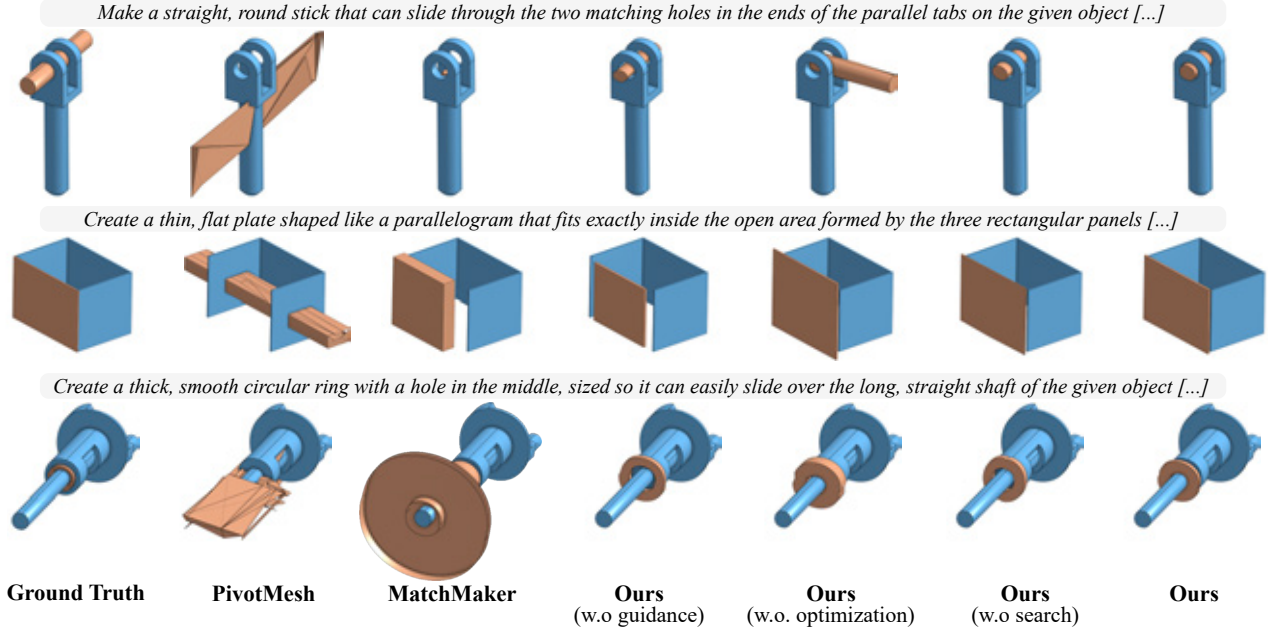


Figure 4. Qualitative Results. We demonstrate the qualitative results of our method and the other two baselines. The condition CAD models are in **blue** and the generated CAD models are in **orange**.

put text prompt and (2) more geometrically fit with the conditional CAD model. Table 3 shows the rate of outputs from our method compared to baselines for questions (1) and (2) under “Semantics” and “Geo. Compatibility”, respectively. The result indicates that MatchMaker gains a competitive semantics alignment with the highest tie rate. The remaining results show that users favor outputs from our method over those from the baselines, as evidenced by a win-rate that is 3.5-9.5 times higher than the tie rate.

Furthermore, Fig. 4 presents qualitative comparisons across the evaluated methods. While PivotMesh and MatchMaker can produce accurate output for conditional CAD models with few faces, they struggle with those having either more faces or complex contact face constraints. Without guidance, our method tends to produce CAD models that are undersized relative to the ground truth. In contrast, applying our guidance strategy yields models that align more accurately with the conditional CAD geometry, demonstrating improved fit and geometric fidelity.

	Semantics	Geo. Compatibility
Win vs. PivotMesh	0.84	0.86
Tie vs. PivotMesh	0.11	0.09
Win vs. MatchMaker	0.49	0.67
Tie vs. MatchMaker	0.37	0.19

Table 3. Human evaluations. The user preferences on the outputs from our method and baselines over two axes.

5.4. Ablation Studies & Analysis

How effectively does the geometry-guided mechanism support the generation of a complementary part? Table 4 demonstrates the result for our guidance strategy

and three ablations. We first experimented with a simpler heuristic that selects candidates based on the minimum Chamfer Distance to the conditional contact faces, as reported in the “**✗** Optimization” row. This criterion provides an insufficient geometric guiding signal, resulting in uniformly worse metrics and underscoring the need for richer cues. In the second variant, we replace intermediate samples directly with predicted guiding samples. As shown in “**✗** Search” row, skipping the search step imposes strong guidance that attains the lowest PR and a competitive IV, but this excessive guidance significantly degrades the generation quality, yielding the worst CD and VR. To assess the effect of softer guidance, we compute the candidate scores by using only the geometry fitness term. The result is reported in “**✗** Regularization” row. Without regularization, search-based candidate selection still preserves plausibility, as evidenced by the competitive VR. However, solely relying on geometry-optimized contact faces as guidance cues disrupts the alignment of other faces and increases the CD. Overall, the results demonstrate that our comprehensive method achieves a more balanced approach between plausibility and geometric constraint satisfaction. The analysis in Fig. 5 further highlights that our guidance strategy generates CAD models that align more accurately.

Ablation	CD ↓	PR ↓	IV ↓	VR ↑
Ours	86.03	0.23	6.90	0.44
✗ Optimization	88.95	0.25	7.37	0.43
✗ Search	90.30	0.21	7.34	0.25
✗ Regularization	87.50	0.26	8.52	0.43

Table 4. Ablations on Guidance Methods. Our method’s performance compared to its variants.

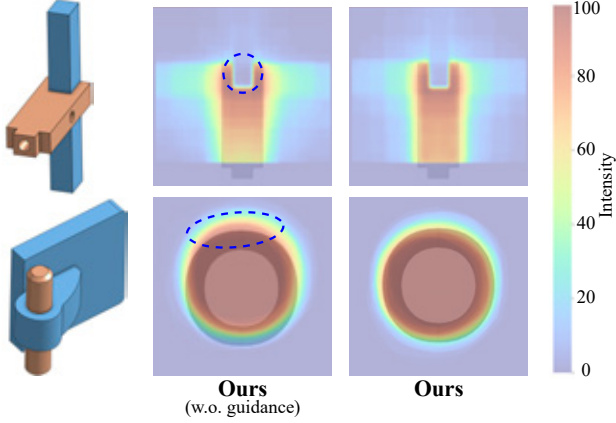


Figure 5. Heatmap of Generated Shapes. Visualization of the top-down view of 100 generated CAD models using our method, with the blue dashed ellipses highlighting the intersection areas with higher intensity.

How does the number of sampled neighbors affect the effectiveness of guidance? We investigate the impact of the number of neighbor candidates, N_p , used during guidance on the quality of the generated complementary CAD models. As illustrated in Table 5, increasing N_p strengthens the guidance but over-constrains the sampling generation, leading to less plausible and shape-degraded outputs indicated by lower VR, higher CD, and IV. In contrast, imposing a small N_p leads to insufficient sampling constraints, resulting in poor generation. The lowest IV is likely due to the larger gaps between contact faces, as evidenced by the highest PR. We find that $N_p = 6$ consistently offers a balance between constraint satisfaction and generation quality.

No. Neighbors	CD ↓	PR ↓	IV ↓	VR ↑
Ours ($N_p = 4$)	90.37	0.26	6.75	0.44
Ours ($N_p = 6$)	86.03	0.23	6.90	0.44
Ours ($N_p = 8$)	89.58	0.25	7.22	0.43

Table 5. Performance with Different Numbers of Neighbors. The influence of setting number of neighbors as 4, 6, and 8.

How does our method perform with various types of conditional CAD models? Table 6 shows how the type of conditional CAD model affects the performance of our method. When only one contact face is provided, the optimization becomes easier, leading to the lowest PR and IV. However, the weak constraint allows excessive geometric freedom, often resulting in overly extended shapes and a higher CD (as shown in the first sample in Fig. 6). With two to three contact faces, the conditional CAD imposes stronger constraints, guiding both the placement and extent of the generated part and yielding lower CD. In contrast, four or more contact faces introduce excessive complexity, increasing interpenetration IV and degrading CD performance. The VR in the “Other” column shows that our method produces valid outputs at a similar rate, even when the one-to-one contact-face assumption is violated.

	One-to-One				Other
	$ \mathcal{I} = 1$	$ \mathcal{I} = 2$	$ \mathcal{I} = 3$	$ \mathcal{I} \geq 4$	
CD ↓	97.86	58.64	52.88	105.82	64.24
PR ↓	0.18	0.36	0.30	0.42	0.30
IV ↓	4.91	6.67	12.62	19.17	8.25
VR ↑	0.50	0.30	0.28	0.20	0.56

Table 6. Performance under different condition types. We group conditions by ground-truth contact-face labels. The first four columns report cases with one-to-one contact faces, divided by the number of condition contact faces. “Other” shows cases without one-to-one contact faces.

6. Discussions & Conclusions

In this work, we address the task of generating compositional CAD. In particular, we present *KnitCAD*, a new dataset with metadata and a textual prompt to facilitate research in this direction. We proposed *CADKnitter*, a conditional diffusion model with a geometry-guided sampling strategy to generate CAD models that align with the textual prompt and can be assembled with existing CAD models. Our proposed guidance strategy utilizes an optimization-based method to predict the desired geometry and searches in the sampling steps for a direction that leads to the assembly space. Empirically, the proposed method outperforms other baselines.

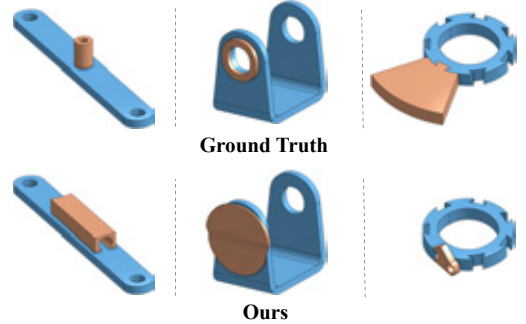


Figure 6. Failure Cases. Our method fails when the contact face constraints are either ambiguous or complex.

Despite promising results, our method still has limitations, as shown in Fig. 6. On one hand, the one-to-one contact face optimization might provide unstable directions during the sampling process. On the other hand, contact faces might not offer sufficient geometric constraints for the conditional CAD models. For instance, the inner diameter of the generated ring in the second sample in Fig. 6 must be matched with the diameter of the conditioning hole, which is not considered to be part of the contact faces. Furthermore, conditional CAD models with complex geometries of contact faces can reduce the performance of our method. These highlight the challenges of compositional CAD generation, indicating its need for further investigation. We anticipate that our research will stimulate further exploration in generating CAD models for assembly, advancing towards practical applications.

CADKnitter: Compositional CAD Generation from Text and Geometry Guidance

Supplementary Material

This Supplementary Material provides extra material for the paper “CADKnitter: Compositional CAD Generation from Text and Geometry Guidance”. The material is organized as follows:

- Section A provides details of the automatic dataset annotation pipeline and statistics.
- Section B provides the detailed implementation of our method.
- Section C provides the detailed experiment setups, including the implementation of evaluation metrics and baselines.
- Section D presents more results, including additional guidance analysis and more qualitative results.

A. Additional Details on KnitCAD Dataset

A.1. Annotation Pipeline

Contact Face Labeling. For each pair of B-rep models, we first assemble the two parts using the joint metadata from Fusion 360 Joint [56] and Automate [17]. We set the distance threshold to $\delta = 0.1$, so a point is considered close to a face if it lies within 0.1 mm of that face. Fig. 7 shows an example of the contact faces and the points from the two objects that satisfy the contact conditions defined in Sec. 3.2 in the main paper.

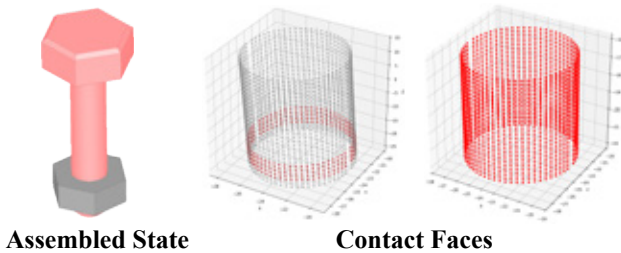


Figure 7. Contact faces. Left: two CAD models in their assembled state. Right: the contact faces of the pink and gray parts, along with points from both parts that satisfy the contact conditions defined in the paper.

Shape description and textual prompt generation. We resize all multi-view images to a resolution of 512×512 before providing them, along with the prompts, to the MLLM. We show the prompts that we use with the MLLM to describe the shape of each CAD model and generate the compositional text prompts for each pair of CAD models in Fig. 9 and Fig. 10.

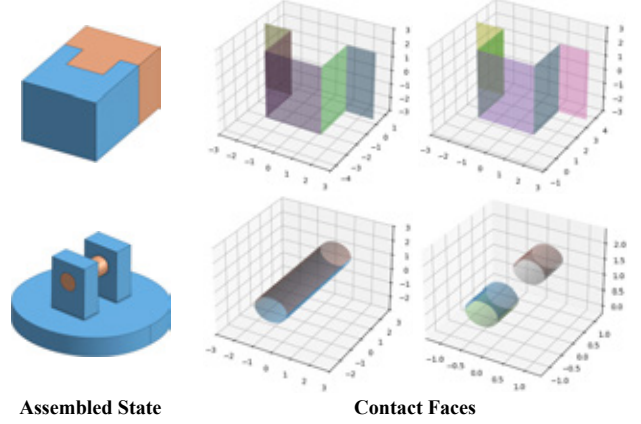


Figure 8. Contact face conditions. We categorize the condition CAD models by the contact faces between the condition CAD models and target CAD models. Top: each contact face on the condition model matches exactly one contact face on the target model (one-to-one cases). Bottom: one target contact face matches multiple condition contact faces (other cases).

A.2. Dataset Statistics

Condition Types. We group the contact-face conditions into two types. The first type contains pairs where each contact face on the condition CAD model matches exactly one contact face on the target CAD model (one-to-one). The second type contains all other cases, where one contact face can match multiple faces, or multiple contact faces can match a single face. Fig. 8 shows examples of both types. The distribution of these two types is shown in Fig. 11a. For the one-to-one cases, we also report the distribution of the number of contact faces.

Prompt Length Distribution. We show the distribution of text prompt lengths in Fig. 11b.

A.3. Dataset Details

Train-test Split. For Fusion 360 Joint, we use the official train-test split provided with the dataset. For Automate, we split the data into train, validation, and test sets using a 60/20/20 ratio.

Data Representation. For each pair of B-rep models, we first transform both models into their assembled state. We then sample UV faces and edges on each B-rep model in this assembled configuration. Each data sample in KnitCAD consists of a pair of B-rep models that are already assembled and share the same global coordinate frame. Our method and all baselines are trained to generate B-rep models in this shared global coordinate frame.

You are given four images in a grid of 2x2 of a single object. Your task is to write a description for the object from these images.

Follow these rules strictly:

- Goal: Produce a precise, compact description of geometry and topology (not color, brand, text, background, lighting, or materials).
- Views: Use all provided views; reconcile conflicts. Do not describe the object from a single specific reference view (top, left, right, bottom).

Output: A detailed description of the object’s geometry and topology. Do not mention features that are not visible in all four images.

Examples:

- “The object is a rectangular cuboid with six faces. Two opposite faces are open rectangular frames, each with rounded-rectangle cutouts along all four edges. The remaining four faces are solid panels. Each solid panel has a row of evenly spaced, elongated rounded-rectangle holes parallel to the long axis of the cuboid.”
- “The object is a thin, square plate with rounded edges and corners. One face of the plate is solid and featureless. The opposite face features a recessed square grid, forming a pattern of evenly spaced square openings bounded by a raised border along all four edges. The thickness of the object remains consistent throughout, and the grid pattern occupies the entire face except for the surrounding raised border.”

Figure 9. Prompt used to synthesize shape description from multi-view images.

The image shows two CAD models that fit together. The engineer is only given the condition model (not the image or descriptions). Your task is to act as a layperson and write a short prompt (1–2 sentences) for the engineer to design the target model so that it can connect with the condition model.

The prompt should clearly describe the shape and structure of the target object, and specify which part of it will attach or fit into the condition object. Do not mention colors, functions, or purposes of the object. Use simple, everyday words and avoid vague or technical terms.

Example prompts:

- “Make a piece shaped like a flat circle with a hole in the middle, so it can slide snugly onto the tall round stick of the given object and rest flush against the wide round part at the base.”
- “Generate a long, straight stick with six flat sides that smoothly bends into a nearly full open hook at one end, so that the curved hook can wrap snugly around the round cylinder of the given object and the straight part sits flat against its side.”

Use the following descriptions to identify the objects in the image and generate the prompt:

Condition Object Description: {condition object description}

Target Object Description: {target object description}

Prompt for the engineer:

Figure 10. Prompt used to synthesize instructions for generating complementary CAD parts.

B. CADKnitter Implementation Details

B.1. Diffusion-based CAD generation

Implementation Details. In our work, the set of face entities is denoted by $\mathbf{x} = \{x^{(i)}\}_{i=1}^N$, where each $x^{(i)} \in \mathbb{R}^6$ is an axis-aligned bounding box. Each bounding box $x^{(i)}$ encloses one face and is represented as

$$x^{(i)} = [x_{\min}, y_{\min}, z_{\min}, x_{\max}, y_{\max}, z_{\max}],$$

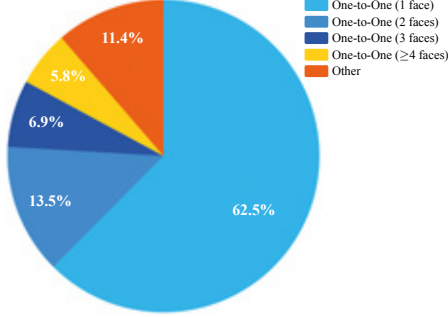
which stores the coordinates of the bottom-left and top-right corners. We encode each bounding box $x^{(i)}$ using an MLP that maps the 6D coordinates to a latent feature with hidden dimension D . The diffusion timestep t is encoded and added element-wise to the noisy latent feature. The noise

predictor ϵ_θ is implemented as a Transformer-based module. After denoising, another MLP maps the latent features back to bounding box coordinates.

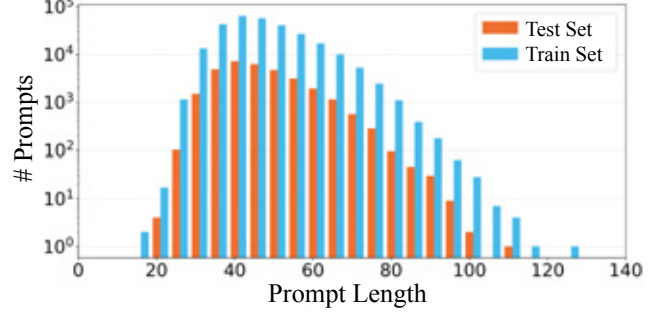
For the text input, we extract embeddings using a text encoder, specifically BERT [5]. We then apply a linear projection to map the text embeddings into the same latent space as the B-rep face bounding box embeddings. All latent features of both face bounding boxes and texts use the same hidden dimension $D = 768$.

To decode the UV face geometry, we leverage the second stage in BrepGen. Given the bounding boxes $\mathbf{x}$, we use the pretrained face decoder to reconstruct the UV-sampled face points

$$s = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{N_s}\} \in \mathbb{R}^3,$$



(a) Distribution of condition types.



(b) Distribution of text prompt lengths.

Figure 11. Dataset statistics. (a) Distribution of contact-face condition types. (b) Distribution of text prompt lengths.

where $N_s = 32 \times 32$ is the number of sampled points on the face.

Training. Following [16, 61], we split closed faces (e.g., cylindrical faces) along their seams (e.g., a closed cylinder is split into two four-sided half-cylinders). To keep training within GPU memory limits, we filter out B-rep models that have more than 70 faces, more than 10 contact faces, more than 40 edges per face, or that are composed of multiple solid bodies. After this filtering, we obtain 95,004 samples to train the face bounding-box denoiser. Before training, we normalize B-rep models in the shared global coordinate frame. For each pair of B-rep models, we compute a translation and a scaling factor so that the condition B-rep model is centered at the origin and lies within the range $[-3, 3]$ along each axis. We then apply this same translation and scaling to both the condition and the target B-rep models. For training, we set the number of contact faces for generated CAD models to $M = 10$. We train the denoiser on a single NVIDIA A100 80GB GPU and use half-precision to reduce memory usage and speed up training. We use AdamW [31] with a learning rate of 5×10^{-4} , a batch size of 256, and train the diffusion model for 5,000 epochs.

B.2. Geometry-Guided Search

Following [52, 68], we only apply guidance at a few late steps of the reverse diffusion process. We set the weight for regularization term in computing candidate score to $\omega_u = 1$. We evaluate different numbers of guidance steps, applied from $t = 110$ down to $t = 50$. As shown in Table 7, using 4 guidance steps gives the best overall performance.

B.3. Guiding-Sample Predictor

The guiding-sample predictor $\mathcal{G}_\pi$ returns a set of optimized contact faces for the generated CAD model $\bar{\mathbf{x}}_t$. These optimized faces provide approximate geometric cues for the search stage. Defining exact contact-face constraints between two CAD models is nontrivial [17]. To make the problem tractable, we relax it to a one-to-one face optimization problem. Under this setting, the guiding-sample pre-

No. Guidance Steps	CD ↓	PR ↓	IV ↓	VR ↑
2	88.72	0.26	6.53	0.44
4	86.03	0.23	6.90	0.44
8	86.58	0.24	7.43	0.43

Table 7. Effect of the number of guidance steps. We compare different numbers of geometry-guidance steps during the reverse diffusion process.

dictor has two stages: (1) find matching face pairs and edge pairs between the generated and condition contact faces; (2) optimize the generated faces using the condition faces as references. We treat contact-face constraints that are not one-to-one as ambiguous constraints, as illustrated by the second example in Fig. 8.

Face and Edge Matching. We first establish one-to-one correspondences between the two sets of contact faces using Hungarian matching [20]. The cost matrix is defined by the point-to-mesh distance for every pair of faces. For point-to-mesh distance, we use the implementation from PyTorch3D [44]. For each matched face pair, we match the boundary edges of the condition contact face to the boundary edges of the generated contact face, again using Hungarian matching [20]. In this step, the cost matrix is defined by the Chamfer Distance [44] between every pair of boundary edges. We extract boundary edges by removing edges that belong to more than one face. An illustration of the face and edge matching is shown in Fig. 13.

Optimization. In our implementation, we use the same time-dependent weight function for both the positional and shape cost terms:

$$\lambda_{\text{pos}}(t) = \lambda_{\text{shape}}(t) = \begin{cases} 1, & t > 0.7, \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

Following [16, 61], after splitting closed faces, the faces are still in contact with each other, but their rotations can be misaligned. Because of this, we do not use the edge-angle

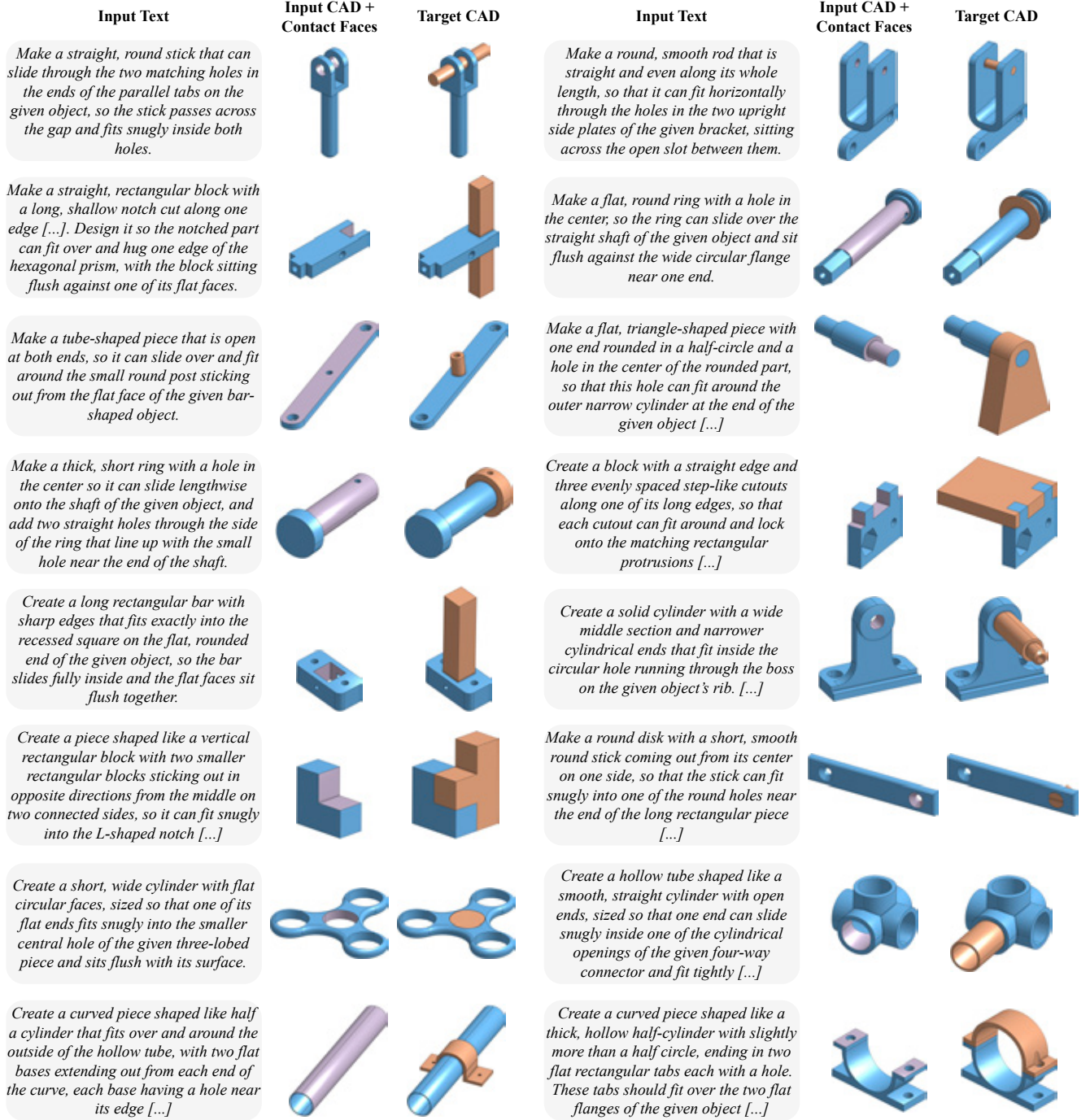


Figure 12. Examples from KnitCAD dataset. We demonstrate examples from KnitCAD. The condition CAD models are in **blue**, the target CAD models are in **orange**, and the desired contact faces are in **purple**.

cost term in our dataset. Concretely, we set $\lambda_{\text{len}} = 1.0$ and $\lambda_{\text{angle}} = 0.0$. Fig. 12 shows examples from our KnitCAD dataset. Fig. 14 shows an example of the optimization process using our objectives.

C. Details of Experimental Setups

C.1. Evaluation Metrics

In this section, we provide more details about each evaluation metric used in the paper.

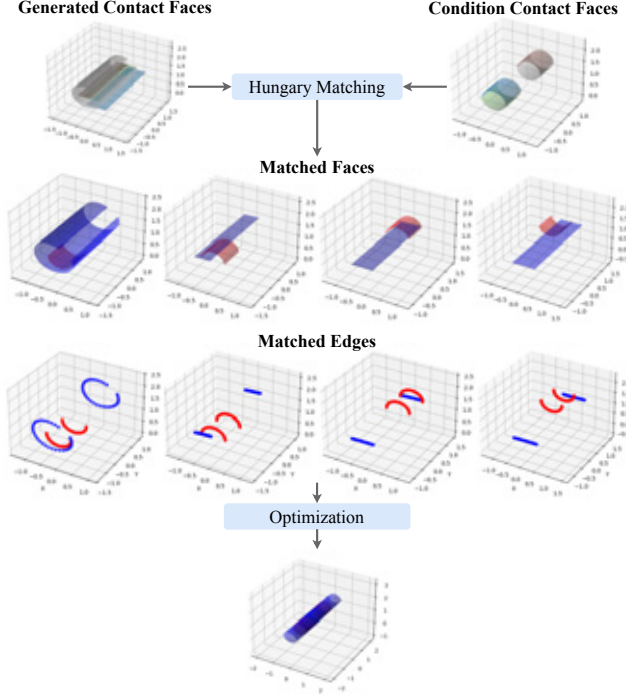


Figure 13. Matching example. Illustration of one-to-one face and edge matching for optimization. From the decoded generated contact faces, we use Hungarian matching to find their corresponding condition contact faces. The boundary edges of the condition contact faces are then matched to boundary edges of the generated contact faces. The red and blue point sets represent boundary edges from the condition and generated contact faces, respectively.

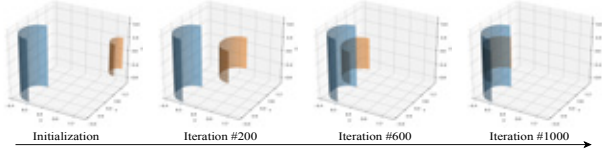


Figure 14. Optimization. Our optimization objectives deform the yellow face so that it fits the blue face.

- **Chamfer Distance (CD).** Following [61], we sample 2,000 points from the surfaces of both the generated and ground-truth CAD. The Chamfer Distance is computed as the average minimum distance between the two sets.
- **Intersection Volume Percentage (IV).** We compute the volume of the intersection solid between the generated object and the condition object. This metric is only defined for watertight objects.
- **Proximity (PR).** We first identify the faces that are in contact with the desired contact faces of the condition CAD model by applying the contact face conditions defined in the main paper. We then compute PR as the minimum distance between these contact faces on the generated and condition CAD models.

- **Valid Ratio (VR).** We use OpenCASCADE [40] as the CAD kernel to construct the B-rep models. A CAD model is counted as valid if OpenCASCADE can successfully build it. VR is the fraction of valid models among all generated samples.

C.2. Implementation Details of Baselines

In this section, we describe the implementation details of the baselines and our method variants.

Text-conditioned version of MatchMaker [53]. MatchMaker is a three-stage framework: (1) contact surface extraction, (2) shape completion, and (3) clearance specification. In our problem, we already provide the contact surface labels in the dataset, and we do not use the clearance specification stage. Therefore, we only use the second stage, which performs shape completion using CAD autocompletion [61]. MatchMaker uses BrepGen [61] as the backbone and adapts RePaint [32] to generate complementary geometry for the extracted contact surface. In this way, the geometric constraints are directly enforced during sampling. Similar to our method, we train the first diffusion model of BrepGen with text conditioning. The text condition is injected into the noisy latent features in the same way as in our model. We train the text-conditioned MatchMaker for 5,000 epochs, with the same training setup as our method. During inference, we apply RePaint [32] at all timesteps $t > 100$. We do not apply RePaint at smaller timesteps, since we observe that using RePaint at very small t leads to less plausible generations and a lower VR compared to the results reported in the main paper.

Text- and Mesh-conditioned version of PivotMesh [55]. PivotMesh is an auto-regressive model that first generates a coarse mesh representation (pivot vertices), and then refines it to a full mesh in a coarse-to-fine manner. As in our method, we first normalize the condition and target meshes using the same translation and scaling, so that the condition mesh is zero-centered and lies inside a canonical cube $[-1, 1]^3$. PivotMesh discretizes mesh vertices using 7-bit quantization. To support this, we further scale all meshes into a unit cube using a global scale factor computed from the maximum and minimum coordinates over the training set. In practice, we use the 90th percentile of the maximum and minimum coordinates to ignore very large outlier meshes and reduce vertex collapse for nearby vertices. For conditioning, we introduce text tokens and condition-mesh tokens into the generation process via extra cross-attention layers in the Transformer blocks of PivotMesh. We use the pretrained auto-encoder from the original PivotMesh model, and train the autoregressive Transformer from scratch for 165,000 steps with a batch size of 6 on 4 NVIDIA A100 80GB GPUs. We use gradient accumulation with 2 steps, set the learning rate to 1×10^{-4} , and use 2,000 warm-up steps.

Ours (w.o. search). This variant removes the search stage. In particular, we directly update the intermediate samples with the predicted guiding samples:

$$\mathbf{x}_t \leftarrow \mathcal{G}_\pi(\tilde{\mathbf{x}}_t, t, \mathbf{c}).$$

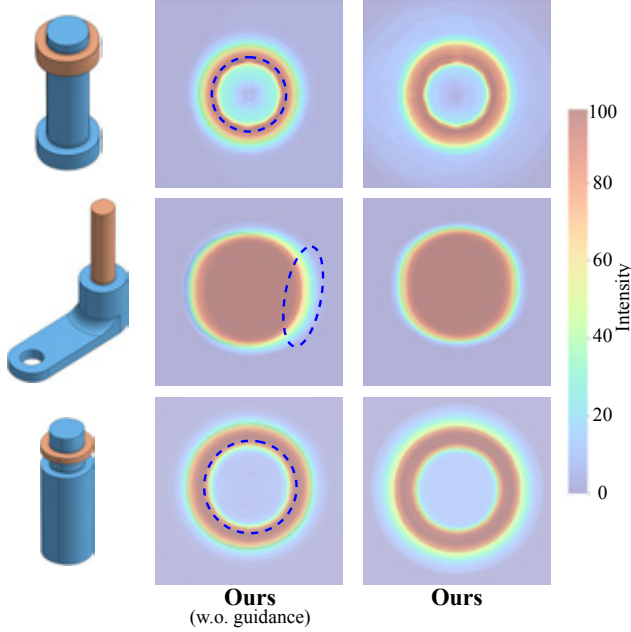


Figure 15. Heatmap of Generated Shapes. Visualization of the top-down view of 100 generated CAD models using our method, with the blue dashed ellipses highlighting the intersection areas with higher intensity.

Ours (w.o. optimization). In this variant, we replace our proposed composite score with a simpler heuristic. The score of a candidate is defined as

$$D_{\text{close}}(\mathbf{S}_t, \mathbf{S}_c) = \mathbb{E}_{s_c^j \in \mathbf{S}_c} \left[\min_{s_t^i \in \mathbf{S}_t} d_{\text{CD}}(s_t^i, s_c^j) \right], \quad (8)$$

where $\mathbf{S}_t = \{s_t^i\}_{i=1}^M$ and $\mathbf{S}_c = \{s_c^j\}_{j=1}^{|Z|}$ are the sets of contact faces on the generated and condition CAD models, respectively. Each s^i is a sampled discretized face with N_s points. The set $\mathbf{S}_t$ is decoded from the intermediate samples $\tilde{\mathbf{x}}_t$, and d_{CD} is the Chamfer Distance between two point sets. We select the candidate with the lowest score to update the intermediate samples.

Ours (w.o. regularization). For this ablation, we set $\omega_u = 0$. All other implementation details are the same as in the full method.

D. Additional Analysis and Qualitative Results

We provide more analysis of the guidance in Fig. 15. In the first and third examples, the rings generated by our method with guidance have a more accurate inner diameter and

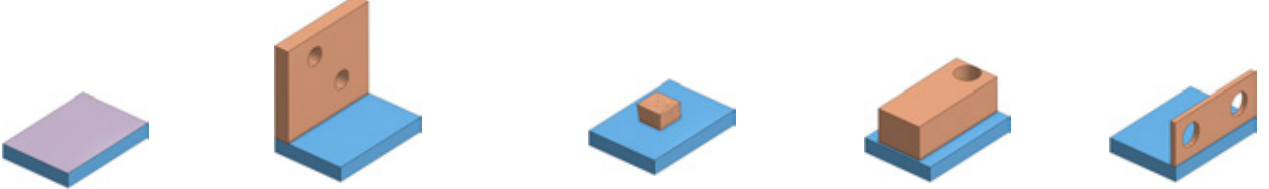
a larger overall size compared to the generations without guidance. It is likely due to the regularization term encourages the model to preserve the global shape ratio, while the geometry fitness term pushes the inner diameter to be large enough to fit the condition contact surfaces.

We also show additional qualitative results from our method and the baselines in Fig. 16. Since MatchMaker relies on CAD autocompletion, it often generates objects that are larger than desired. In contrast, our method applies guidance at selected steps, which helps the generated objects remain semantically aligned with the text while also fitting the condition objects geometrically.

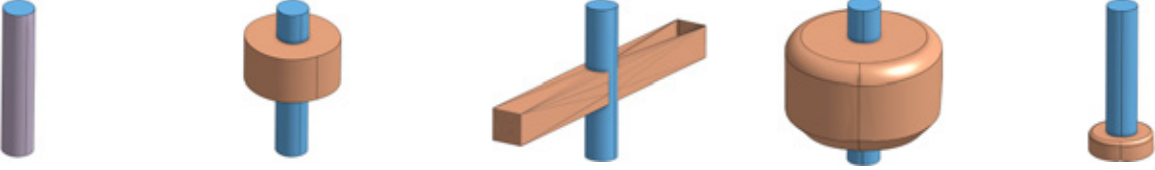
Make a piece with a long round stick that has a six-sided short block connected to one end, so that the round stick can fit closely through the hole in the center of the ring-shaped disk and the flat face of the six-sided block can press up against one side of the disk.



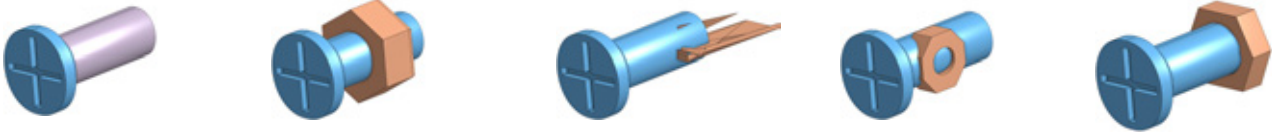
Create a flat rectangular block with two round holes going all the way through one of the larger flat faces, so this face can line up and attach to the flat, featureless face of the given solid block.



Make a thick ring-shaped piece with a smooth round hole through the middle, so that this hole can slide over and fit snugly around the given smooth round cylinder.



Make a solid piece shaped like a hexagon when viewed from the ends, with flat parallel ends and six flat sides, and add a smooth, round hole straight through the center so it can slide onto the round shaft of the given object.



Make a block that is shaped like a rectangle with straight sides, and add three holes going all the way through one of its big flat faces: put a large hole in the middle and two smaller holes [...] so the big hole can fit the straight round stick of the given object.



Make a solid, smooth round rod that is longer than it is wide, so that one end of the rod can fit snugly through the large circular hole near the rounded end of the given plate.



**Input CAD +
Contact Faces**

Ground Truth

PivotMesh

MatchMaker

Ours

Figure 16. Qualitative Results. We demonstrate the qualitative results of our method and the other two baselines. The condition CAD models are in **blue**, the generated CAD models are in **orange**, and the desired contact face are in **purple**.

References

- [1] Silvia Ansaldi, Leila De Floriani, and Bianca Falcidieno. Geometric modeling of solid objects by using a face adjacency graph representation. *SIGGRAPH*, 1985. 2
- [2] Cheng Chen, Jiacheng Wei, Tianrun Chen, Chi Zhang, Xiaofeng Yang, Shangzhan Zhang, Bingchen Yang, Chuan-Sheng Foo, Guosheng Lin, Qixing Huang, et al. Cad-crafter: Generating computer-aided design models from unconstrained images. In *CVPR*, 2025. 2, 6

- [3] Minghao Chen, Roman Shapovalov, Iro Laina, Tom Monnier, Jianyuan Wang, David Novotny, and Andrea Vedaldi. Partgen: Part-level 3d generation and reconstruction with multi-view diffusion models. In *CVPR*, 2025. 1, 2
- [4] Yunuo Chen, Tianyi Xie, Zeshun Zong, Xuan Li, Feng Gao, Yin Yang, Ying Nian Wu, and Chenfanfu Jiang. Atlas3d: Physically constrained self-supporting text-to-3d for simulation and fabrication. *NeurIPS*, 2024. 1, 2
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 10
- [6] Yilun Du and Igor Mordatch. Implicit generation and modeling with energy based models. *NeurIPS*, 2019. 5
- [7] Yilun Du, Conor Durkan, Robin Strudel, Joshua B Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Sussman Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc. In *ICML*, 2023. 2, 5
- [8] Christer Ericson. *Real-time collision detection*. Crc Press, 2004. 6
- [9] Abraham D Flaxman, Adam Tauman Kalai, and H Brendan McMahan. Online convex optimization in the bandit setting: gradient descent without a gradient. *arXiv*, 2004. 5
- [10] Haoxiang Guo, Shilin Liu, Hao Pan, Yang Liu, Xin Tong, and Baining Guo. Complexgen: Cad reconstruction by b-rep chain complex generation. *TOG*, 2022. 2
- [11] Hao Guo, Xiaoshui Huang, Yunpeng Bai, Hongping Gan, Yilei Shi, et al. Brepdiff: Lightweight generation of complex b-rep with 3d gat diffusion. In *CVPR*, 2025. 2, 4
- [12] Huy Ha, Shubham Agrawal, and Shuran Song. Fit2form: 3d generative model for robot gripper form design. In *CoRL*, 2021. 2
- [13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 4
- [14] Yufei Huang, Yunshu Liu, Lirong Wu, Haitao Lin, Cheng Tan, Odin Zhang, Zhangyang Gao, Siyuan Li, Zicheng Liu, Yunfan Liu, et al. Eva: Geometric inverse design for fast protein motif-scaffolding with coupled flow. In *ICLR*, 2025. 5
- [15] Pradeep Kumar Jayaraman, Aditya Sanghi, Joseph G Lambourne, Karl DD Willis, Thomas Davies, Hooman Shayani, and Nigel Morris. Uv-net: Learning from boundary representations. In *CVPR*, 2021. 3, 4
- [16] Pradeep Kumar Jayaraman, Joseph G Lambourne, Nishkrit Desai, Karl DD Willis, Aditya Sanghi, and Nigel JW Morris. Solidgen: An autoregressive model for direct b-rep synthesis. *TMLR*, 2022. 2, 6, 11
- [17] Benjamin Jones, Dalton Hildreth, Duowen Chen, Ilya Baran, Vladimir G Kim, and Adriana Schulz. Automate: A dataset and learning approach for automatic mating of cad assemblies. *TOG*, 2021. 1, 2, 3, 5, 9, 11
- [18] Mohammad Sadil Khan, Sankalp Sinha, Talha Uddin, Didier Stricker, Sk Aziz Ali, and Muhammad Zeshan Afzal. Text2cad: Generating sequential cad designs from beginner-to-expert level text prompts. *NeurIPS*, 2024. 2, 3, 6
- [19] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. in 2019 IEEE. In *CVPR*, 2018. 3
- [20] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 1955. 5, 11
- [21] Mingi Lee, Dongsu Zhang, Clément Jambon, and Young Min Kim. Brepdiff: Single-stage b-rep diffusion model. In *SIGGRAPH Conference*, 2025. 2, 4
- [22] Jing Li, Yihang Fu, and Falai Chen. Dtgbbrep: A novel b-rep generative model through decoupling topology and geometry. In *CVPR*, 2025. 2, 4
- [23] Jiahao Li, Weijian Ma, Xueyang Li, Yunzhong Lou, Guichun Zhou, and Xiangdong Zhou. Cad-llama: leveraging large language models for computer-aided design parametric 3d model generation. In *CVPR*, 2025. 1
- [24] Pu Li, Jianwei Guo, Huibin Li, Bedrich Benes, and Dong-Ming Yan. Sfmcad: Unsupervised cad reconstruction by learning sketch-based feature modeling operations. In *CVPR*, 2024. 2
- [25] Ruining Li, Chuanxia Zheng, Christian Rupprecht, and Andrea Vedaldi. Dso: Aligning 3d generators with simulation feedback for physical soundness. *arXiv*, 2025. 1
- [26] Yuan Li, Cheng Lin, Yuan Liu, Xiaoxiao Long, Chenxu Zhang, Ningna Wang, Xin Li, Wenping Wang, and Xiaohu Guo. Caddreamer: Cad object generation from single-view images. In *CVPR*, 2025. 1, 2, 6
- [27] Jiayi Liu, Hou In Ivan Tam, Ali Mahdavi-Amiri, and Manolis Savva. Cage: Controllable articulation generation. In *CVPR*, 2024. 2
- [28] Jiayi Liu, Denys Iliash, Angel X Chang, Manolis Savva, and Ali Mahdavi-Amiri. Singapo: Single image controlled generation of articulated parts in objects. *ICLR*, 2025. 2
- [29] Yujia Liu, Anton Obukhov, Jan Dirk Wegner, and Konrad Schindler. Point2cad: Reverse engineering cad models from 3d point clouds. In *CVPR*, 2024. 2
- [30] Yilin Liu, Duoteng Xu, Xingyao Yu, Xiang Xu, Daniel Cohen-Or, Hao Zhang, and Hui Huang. Hola: B-rep generation using a holistic latent representation. *TOG*, 2025. 2, 4
- [31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv*, 2017. 11
- [32] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022. 13
- [33] Rundong Luo, Haoran Geng, Congyue Deng, Puhao Li, Zan Wang, Baoxiong Jia, Leonidas Guibas, and Siyuan Huang. Physpart: Physically plausible part completion for inter-actable objects. In *ICRA*, 2025. 1, 2, 6
- [34] Liane Makatura, Michael Foshey, Bohan Wang, Felix HahnLein, Pingchuan Ma, Bolei Deng, Megan Tjandra-suwita, Andrew Spielberg, Crystal Elaine Owens, Peter Yichen Chen, et al. How can large language models help humans in design and manufacturing? *arXiv*, 2023. 1
- [35] Maximilian Mews, Ansar Aynettinov, Vivian Schiller, Peter Eisert, and Alan Akbik. Don't mesh with me: Generating constructive solid geometry instead of meshes by fine-tuning a code-generation llm. *CVPR*, 2025. 2

- [36] Yashraj Narang, Kier Storey, Ireteyio Akinola, Miles Macklin, Philipp Reist, Lukasz Wawrzyniak, Yunrong Guo, Adam Moravanszky, Gavriel State, Michelle Lu, et al. Factory: Fast contact for robotic assembly. *arXiv*, 2022. 1, 3
- [37] Quang Nguyen, Nhat Le, Baoru Huang, Minh Nhat Vu, Chengcheng Tang, Van Nguyen, Ngan Le, Thieu Vo, and Anh Nguyen. Egomusic-driven human dance motion estimation with skeleton mamba. In *ICCV*, 2025. 2
- [38] Toan Nguyen, Minh Nhat Vu, Baoru Huang, An Vuong, Quan Vuong, Ngan Le, Thieu Vo, and Anh Nguyen. Language-driven 6-dof grasp detection using negative prompt guidance. In *ECCV*, 2024. 2
- [39] Erik Nijkamp, Mitch Hill, Tian Han, Song-Chun Zhu, and Ying Nian Wu. On the anatomy of mcmc-based maximum likelihood learning of energy-based models. In *AAAI*, 2020. 5
- [40] OCCT3D. Open CASCADE Technology. Software. Accessed: April 30th 2025. 3, 13
- [41] OpenAI. Introducing GPT-4.1 in the API. Software. Accessed: May 12th 2025. 3
- [42] Gabriel Peyré, Marco Cuturi, and Justin Solomon. Gromov-wasserstein averaging of kernel and distance matrices. In *ICML*, 2016. 5
- [43] Yu Qi, Yuanchen Ju, Tianming Wei, Chi Chu, Lawson LS Wong, and Huazhe Xu. Two by two: Learning multi-task pairwise objects assembly for generalizable robot manipulation. In *CVPR*, 2025. 3
- [44] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv*, 2020. 6, 11
- [45] Daxuan Ren, Jianmin Zheng, Jianfei Cai, Jiatong Li, Haiyong Jiang, Zhongang Cai, Junzhe Zhang, Liang Pan, Mingyuan Zhang, Haiyu Zhao, et al. Csg-stump: A learning friendly csg-like representation for interpretable shape parsing. In *ICCV*, 2021. 2
- [46] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. 4
- [47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021. 4
- [48] Ian Stroud and Hildegard Nagy. *Solid modelling and CAD systems: how to survive a CAD system*. 2011. 2
- [49] Yunsheng Tian, Jie Xu, Yichen Li, Jieliang Luo, Shinjiro Sueda, Hui Li, Karl DD Willis, and Wojciech Matusik. Assemble them all: Physics-based planning for generalizable assembly by disassembly. *TOG*, 2022. 2, 3
- [50] Titouan Vayer, Laetitia Chapel, Rémi Flamary, Romain Tavenard, and Nicolas Courty. Optimal transport for structured data with application on graphs. *ICML*, 2019. 5
- [51] Ruiyu Wang, Yu Yuan, Shizhao Sun, and Jiang Bian. Text-to-cad generation through infusing visual feedback in large language models. *ICML*, 2025. 2, 3
- [52] Tsun-Hsuan Johnson Wang, Juntian Zheng, Pingchuan Ma, Yilun Du, Byungchul Kim, Andrew Spielberg, Josh Tenenbaum, Chuang Gan, and Daniela Rus. Diffusebot: Breeding soft robots with physics-augmented generative diffusion models. *NeurIPS*, 2023. 1, 2, 5, 11
- [53] Yian Wang, Bingjie Tang, Chuang Gan, Dieter Fox, Kaichun Mo, Yashraj Narang, and Ireteyio Akinola. Matchmaker: Automated asset generation for robotic assembly. *ICRA*, 2025. 1, 2, 6, 13
- [54] Kevin J Weiler. *Topological structures for geometric modeling (Boundary representation, manifold, radial edge structure)*. 1986. 2
- [55] Haohan Weng, Yikai Wang, Tong Zhang, CL Chen, and Jun Zhu. Pivotmesh: Generic 3d mesh generation via pivot vertices guidance. *NeurIPS*, 2024. 6, 13
- [56] Karl DD Willis, Pradeep Kumar Jayaraman, Hang Chu, Yunsheng Tian, Yifei Li, Daniele Grandi, Aditya Sanghi, Linh-Tam Tran, J Lambourne, Armando Solar-Lezama, et al. Joinable: Learning bottom-up assembly of parametric cad joints. 2022 iee. In *CVPR*, 2021. 1, 2, 3, 6, 9
- [57] Rundi Wu, Chang Xiao, and Changxi Zheng. Deepcad: A deep generative network for computer-aided design models. In *ICCV*, 2021. 2, 3
- [58] Jingwei Xu, Chenyu Wang, Zibo Zhao, Wen Liu, Yi Ma, and Shenghua Gao. Cad-lm: Unifying multimodality-conditioned cad generation with mllm. *arXiv*, 2024. 2, 3
- [59] Sirui Xu, Zhengyuan Li, Yu-Xiong Wang, and Liang-Yan Gui. Interdiff: Generating 3d human-object interactions with physics-informed diffusion. In *ICCV*, 2023. 2, 5
- [60] Xiang Xu, Karl DD Willis, Joseph G Lambourne, Chin-Yi Cheng, Pradeep Kumar Jayaraman, and Yasutaka Furukawa. Skexgen: Autoregressive generation of cad construction sequences with disentangled codebooks. *ICML*, 2022. 2
- [61] Xiang Xu, Joseph Lambourne, Pradeep Jayaraman, Zhengqing Wang, Karl Willis, and Yasutaka Furukawa. Brepgen: A b-rep generative diffusion model with structured latent geometry. *TOG*, 2024. 2, 4, 6, 11, 13
- [62] Han Yan, Mingrui Zhang, Yang Li, Chao Ma, and Pan Ji. Phycage: Physically plausible compositional 3d asset generation from a single image. *arXiv*, 2024. 1, 2
- [63] Yandan Yang, Baoxiong Jia, Peiyuan Zhi, and Siyuan Huang. Physcene: Physically interactable 3d scene synthesis for embodied ai. In *CVPR*, 2024. 2
- [64] Yunhan Yang, Yufan Zhou, Yuan-Chen Guo, Zi-Xin Zou, Yukun Huang, Ying-Tian Liu, Hao Xu, Ding Liang, Yan-Pei Cao, and Xihui Liu. Omnipart: Part-aware 3d generation with semantic decoupling and structural cohesion. *SIG-GRAPH Asia*, 2025. 2
- [65] Yang You, Mikaela Angelina Uy, Jiaqi Han, Rahul Thomas, Haotong Zhang, Yi Du, Hansheng Chen, Francis Engelmann, Suya You, and Leonidas Guibas. Img2cad: Reverse engineering 3d cad models from images through vlm-assisted conditional factorization. *arXiv*, 2024. 2
- [66] Fenggen Yu, Zhiqin Chen, Manyi Li, Aditya Sanghi, Hooman Shayani, Ali Mahdavi-Amiri, and Hao Zhang. Capri-net: Learning compact cad shapes with adaptive primitive assembly. In *CVPR*, 2022. 2
- [67] Fenggen Yu, Qimin Chen, Maham Tanveer, Ali Mahdavi Amiri, and Hao Zhang. D²csg: Unsupervised learning of compact csg trees with dual complements and dropouts. *NeurIPS*, 2023. 2

- [68] Ye Yuan, Jiaming Song, Umar Iqbal, Arash Vahdat, and Jan Kautz. Physdiff: Physics-guided human motion diffusion model. In *ICCV*, 2023. [2](#), [5](#), [11](#)
- [69] Zhanwei Zhang, Shizhao Sun, Wenxiao Wang, Deng Cai, and Jiang Bian. Flexcad: Unified and versatile controllable cad generation with fine-tuned large language models. *ICLR*, 2024. [2](#)