3D VR Sketch Guided 3D Shape Prototyping and Exploration

Ling Luo[1,2] Pinaki Nath Chowdhury[1,2] Tao Xiang[1,2] Yi-Zhe Song[1,2]
Yulia Gryaditskaya[1,3]

[1]SketchX, CVSSP, University of Surrey, United Kingdom
[2]iFlyTek-Surrey Joint Research Center on Artificial Intelligence
[3]CCM, Surrey Institute for People-Centered AI and CVSSP, University of Surrey, United Kingdom

Abstract

3D shape modeling is labor-intensive, time-consuming, and requires years of expertise. To facilitate 3D shape modeling, we propose a 3D shape generation network that takes a 3D VR sketch as a condition. We assume that sketches are created by novices without art training and aim to reconstruct geometrically realistic 3D shapes of a given category. To handle potential sketch ambiguity, our method creates multiple 3D shapes that align with the original sketch's structure. We carefully design our method, training the model step-by-step and leveraging multi-modal 3D shape representation to support training with limited training data. To guarantee the realism of generated 3D shapes we leverage the normalizing flow that models the distribution of the latent space of 3D shapes. To encourage the fidelity of the generated 3D shapes to an input sketch, we propose a dedicated loss that we deploy at different stages of the training process. The code is available at https://github.com/Rowl1ng/3Dsketch2shape.

1. Introduction

The demand for convenient tools for 3D content creation constantly grows as the creation of virtual worlds becomes an integral part of various fields such as architecture and cinematography. Recently, several works have demonstrated how text and image priors can be used to create 3D shapes [43, 28, 27, 16, 26]. However, it is universally accepted that text is much less expressive or precise than a 2D freehand sketch in conveying spatial or geometric information [53, 12, 44]. Therefore, many works focus on sketch-based modeling from 2D sketches [38, 5, 4, 6, 31, 25, 13, 46, 58, 57, 55, 17, 21] as a convenient tool for creating virtual 3D content. Yet, 2D sketches are ambiguous, and depicting a complex 3D shape in 2D requires substantial sketching expertise. As Virtual Reality (VR) headsets and associated technologies progress [20, 19, 35], more and more works consider 3D VR sketch as an input modality in the context of 3D modeling [52, 54, 42] and retrieval [23, 24, 22, 51, 33, 34, 32]. Firstly, 3D VR sketches are drawn directly in 3D and therefore provide a more immersive and intuitive design experience. Secondly, 3D VR sketches offer a natural way to convey volumetric shapes and spatial relationships. Moreover, the use of 3D VR sketches aligns with advancements in virtual reality technology, making the process of sketching and designing more future-proof and adaptable to emerging technologies.

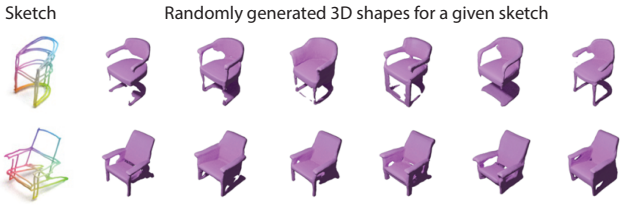

Figure 1. Given a VR (Virtual Reality) sketch input, we generate 3D shape samples that satisfy three requirements: (1 - *fidelity*) reconstructed shapes follow the overall structure of a quick VR sketch; (2 - *diversity*) reconstructed shapes contain some diversity in shape details: such as a hollow or solid backrest, and (3 - *realism*) reconstructions favor geometrically realistic 3D shapes of a given category.

Existing works on 3D shape modeling assume carefully created inputs and focus primarily on the interface and logistics of the sketching process. In this paper, we introduce a novel method for 3D shape modeling that utilizes 3D VR sketching. Our method does not require professional sketch training or detailed sketches and is trained and tested on a dataset of sketches created by participants without art experience. This approach ensures that our model can accommodate a diverse range of users and handle sketches that might be less polished or precise, making it more accessible and practical for real-world applications. Considering the sparsity of VR sketches, a single 3D shape model may

not match the user's intention. Therefore, we advocate for generating multiple shape variations that closely resemble the input sketch, as demonstrated in Fig. 1. The user can then either directly pick one of the models, or refine the design given the visualized 3D shapes, or multiple shapes can be used in some physical simulation process to select the optimal shape within the constraints of the VR sketch.

Working with freehand VR sketches presents several challenges due to the lack of datasets. We are aware of only one fine-grained dataset of VR sketches by Luo et al. [34], which we use in this work. The challenge of working with this data comes from its limited size and the misalignment between sketches and 3D shapes. Luo et al. [34] let participants sketch in an area different from the one where the reference 3D shape is displayed. This allows to model the scenario of sketching from memory or imagination, however, results in a lack of alignment between 3D shapes and sketches, as shown in Fig. 2. Considering the misalignment of sketches and shapes in the dataset, and the ambiguity of the VR sketches, we aim to generate shapes with good fidelity to an input sketch, rather than the reference shape.

We represent our sketches as point clouds, and regress Signed Distance Fields (SDFs) values [39] representing 3D shapes. Despite the seemingly simple nature of the problem, we found that training an auto-encoder in an end-to-end manner results in poor performance due to a dataset's limited size and sketch-shape misalignments as discussed above. We, therefore, start by training an SDF auto-decoder, similar to the one proposed by Park et al. [39]. We then propose several losses that allow us to efficiently train our sketch encoder. In particular, we design a sketch fidelity loss, exploiting the fact that sketch strokes represent 3D shape surface points. Leveraging the properties of SDF, this implies that the regressed SDF values in the points sampled from sketch strokes should be close to zero. To be able to sample multiple 3D shapes for a given input sketch, we adopt a conditional normalizing flow (CNF) model [14], trained to model the distribution of the latent space of 3D shapes. During the training of CNF, we again leverage the introduced sketch fidelity loss, improving the fidelity of reconstruction to the input sketch.

In summary, our contributions are the following:

- We, for the first time, study the problem of conditional 3D shape generation from rapid and sparse 3D VR sketches and carefully design our method to tackle the problem of (1) limited data, (2) misalignments between sketches and 3D shapes and (3) abstract nature of freehand sketches.

- Taking into consideration the ambiguity of VR sketch interpretation, we design our method so that diverse 3D shapes can be generated that follow the structure of a given 3D VR sketch.

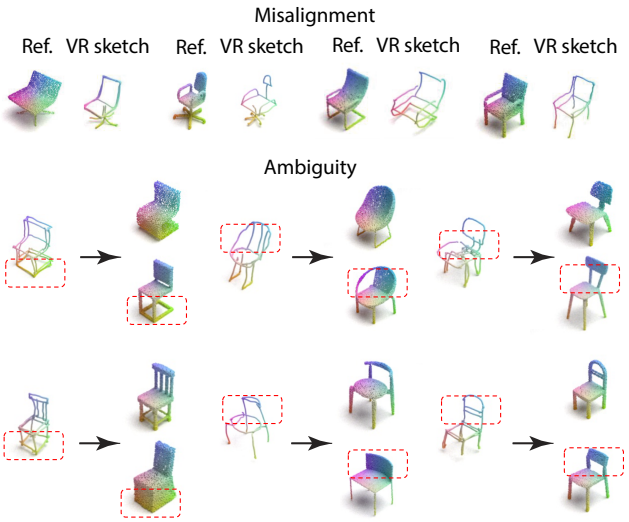

Figure 2. Example of misalignment and ambiguity of 3D sketch. Misalignment: the collected sketches and reference shapes have deviations in terms of the position and proportion of their parts. Ambiguity: due to the sparsity and abstract nature of sketches, strokes can be interpreted differently. For example, the strokes of a cube can represent either slender bars or a closed solid shape.

2. Related Work

2.1. Shape reconstruction and retrieval from 3D sketches

Many earlier works targeted category-level retrieval [23, 24, 22, 51, 33], but the field advancement stagnated due to the lack of fine-grained datasets. Recently several works addressed the problem of fine-grained retrieval from VR sketches [34, 32], and introduced the first dataset which we use in this work. Luo et al. [32] proposed a structure-aware retrieval that allows increasing relevance of the top retrieval results. However, retrieval methods are always limited to the existing shapes. Therefore, we explore conditional sketch generation, aiming to achieve good fidelity to the input, combined with generation results diversity.

Recently, Yu et al. [52] considered the problem of VR sketch surfacing. Their work is optimization-based and assumes professional and detailed sketch input. Similarly to their work, we aim to achieve good fidelity of the reconstruction to an input sketch but take as input sparse and abstract sketches. Moreover, our approach is learning-based, which means we require a class shape prior, but we can handle abstract sketches, and inference is instant.

2.2. 3D shapes generation

In addition to maximizing the fidelity to the input sketch, we aim to generate a set of shapes that are distinctive from each other. This diversity allows users to efficiently explore the design space of shapes resembling their initial sketch.

2.2.1 Generative models

Various 3D shape generative models have been proposed in the literature based on Generative Adversarial Networks (GANs) [48, 1, 9, 49, 56], Variational Auto-Decoders (VADs)[39, 11], autoregressive models [**?**, 47, 36, 50], normalizing flow models [43] and more recently diffusion models [21, 10, 37, 59, 40]. We chose to use normalizing flows trained in the latent space of our auto-encoder due to the simplicity of this model and the fact that it can be easily combined with any pretrained auto-encoder. Recently, several concurrent works proposed the use of diffusion models in the latent space [10, 37, 40]. Our network can easily be adapted to the usage of diffusion models instead of normalizing flows. The contribution of our work lies in the definition of the problem and the overall network architecture, as well as the introduction of appropriate loss functions and the training setup, allowing to generate multiple samples fitting the input 3D VR sketch, given limited training data.

2.2.2 Conditional shape generation

Similarly, diverse conditional generative models were considered that takes as input a sketch [4, 6, 31, 25, 13, 46, 58, 57, 55, 17, 21, 11], an image [18, 10], an incomplete scan in a form of a point cloud [3, 49, 2, 59, 50], a coarse voxel shape [8], or a textual description [10, 43, 16, 29, 40].

Sketch-/image-based reconstruction methods typically focus on the generation of only one output result for each input, while we aim at the generation of multiple 3D shapes. Meanwhile, in the point cloud completion task, it is typically to infer the missing parts from the observed parts and generate multiple possible completion results. Their task, however, differs from our goal as we do not want the network to synthesize non-existent parts, but only to create various shapes that match the sparse freehand sketch taking into account how humans might abstract 3D shapes. This is also the reason why the autoregressive approaches, such as [45, 36, 50], are not suitable for our problem.

Text-guided 3D shape generation shares similar ambiguity properties as VR sketch-guided, i.e., diverse results may match the same input text. CLIP-Forge [43] employs pretrained visual-textual embedding model CLIP to bridge text and 3D domains, and uses conditional normalizing flow to model the conditional distribution of latent shape representation given text or image embeddings. Zhengzhe et al. [29] introduce shape IMLE (Implicit Maximum Likelihood Estimation) to boost results diversity while utilizing a cyclic loss to encourage consistency. In our work, we condition on a VR sketch rather than text and aim to obtain diverse 3D shapes that follow the input sketch structure.

3. Method

We present a conditional generation method that generates *geometrically realistic* shapes of a specific category conditioned on abstract, sparse, and inaccurate freehand VR sketches. Our goal is to enforce the generation to stay close to the input sketch (*sketch fidelity*) while providing sufficient *diversity* of 3D reconstructions.

The architecture of our method is shown in Fig. 3. The method consists of two stages, where the first stage (Fig. 3 (a)) enables deterministic reconstruction for an input sketch, and the second stage allows for multiple sample generation (Fig. 3 (a)). We next describe the details of each stage.

3.1. Shape decoder

We represent 3D shapes using truncated signed distance functions (SDFs), as one of the most common 3D shape representations. This representation is limited to watertight meshes, but without loss of generality, here we assume that our meshes are watertight.

An SDF is a continuous function of the form:

$$\mathrm{SDF}(x) = s \ : \ x \in \mathbb{R}^3, \ s \in \mathbb{R}, \tag{1}$$

where x is a 3D point coordinates and s is the signed distance to the closest shape surface (a negative/positive sign indicates that the point is inside/outside the surface). The underlying surface is implicitly represented by the iso-surface of $\mathrm{SDF}(\cdot) = 0$, and can be reconstructed using marching cubes [30].

Our goal is to reconstruct a 3D shape from a given VR sketch, however, we found that classical auto-encoder training frameworks on our problem perform poorly when trained in an end-to-end manner. This is caused by (1) a limited training set size, and (2) the fact that the sketches are not perfectly aligned with 3D shapes. Therefore, we first train a 3D shape auto-decoder, following Park et al. [39].

Shape auto-decoder The auto-decoder is trained by minimizing an L_1 loss between the ground truth and predicted truncated signed distance values. The decoder

$$\mathrm{D}_\theta([\boldsymbol{d}^g, p_i]) = \tilde{s}_i, \ p_i \in \mathbb{R}^3, \tilde{s}_i \in \mathbb{R} \tag{2}$$

takes as input the 3D shape latent code $\boldsymbol{d}^g$ and the 3D point coordinates p_i; $[\cdot,\cdot]$ represents a concatenation operation. The decoder predicts the per point signed distance value $\tilde{s}_i$. Once the decoder is trained, we freeze its parameters θ.

At inference time, we estimate the 3D shape latent code via Maximum-a-Posterior estimation as follows:

$$\hat{\boldsymbol{d}^g} = \arg\min_{\boldsymbol{d}^g} \sum_{(p_i, s_i) \in G} \mathcal{L}\left(D_\theta\left(\boldsymbol{d}^g, p_i\right), s_i\right) + \frac{1}{\sigma^2} \|\boldsymbol{d}^g\|_2^2 \tag{3}$$

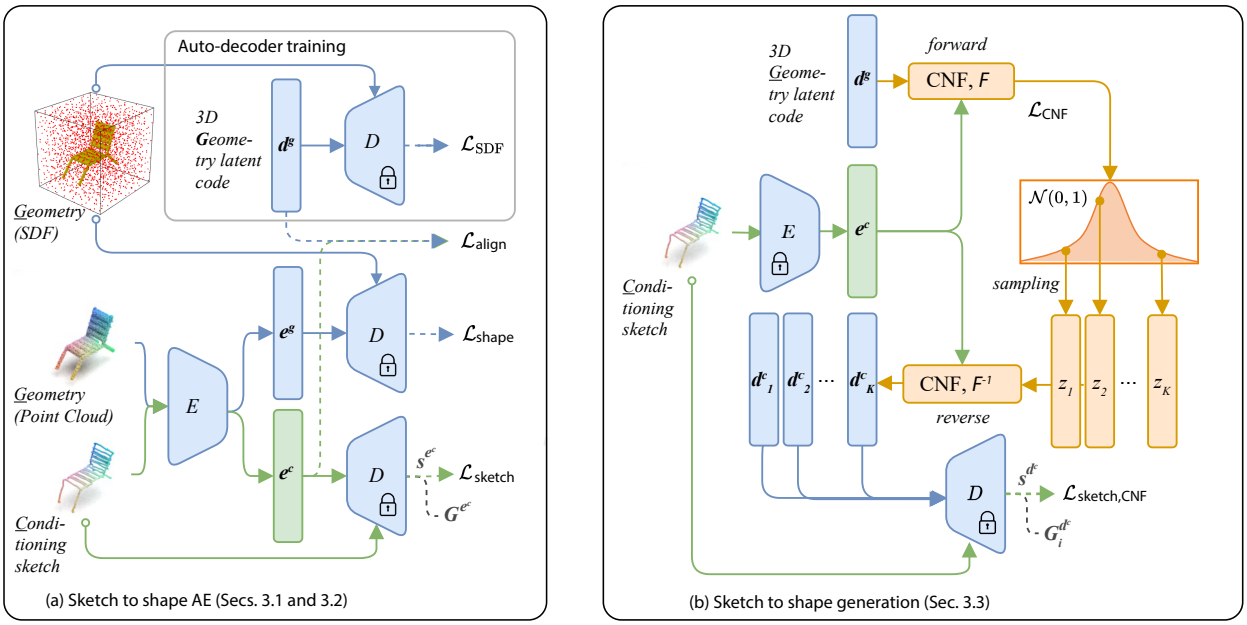

Figure 3. Our method consists of 2 stages: (a) the first stage allows to obtain deterministic 3D shape reconstructions from input sketches, as described in Secs. 3.1 and 3.2, while (b) the second stage enables conditional 3D shape sample generation, as described in Sec. 3.3. The auto-encoder (AE) is trained in three steps: first auto-decoder is trained, then the shape encoder is trained, and finally, the encoder is fine-tuned to jointly encode sketches and shapes.

where the latent vector $\boldsymbol{d}^g$ is initialized randomly, and the sum is taken over points of the 3D shape geometry G. We treat the estimated latent vector $\hat{\boldsymbol{d}}^g$ as a ground truth shape embedding and denote it as $\boldsymbol{d}^g$ for simplicity.

3.2. Encoding VR sketches.

We then train a sketch encoder that maps sketches to the latent space of our 3D shape decoder.

Due to the sparsity of sketch inputs, we represent them as point clouds. We observe that if we only use sketches for training, we obtain very poor generalization to the test set. Therefore, we exploit a joint encoder for sketches and 3D shapes, using a PointNet++ [41] encoder. The encoder $\mathrm{E}_\phi(\cdot) : \mathbb{R}^{N_s \times 3} \rightarrow \mathbb{R}^{256}$ embeds randomly sampled points from input shape surface $G \in \mathbb{R}^{N_s \times 3}$ or sketch strokes $C \in \mathbb{R}^{N_s \times 3}$ to a feature vector $\boldsymbol{e}^g \in \mathbb{R}^{256}$ and $\boldsymbol{e}^c \in \mathbb{R}^{256}$, respectively.

The encoder parameters $\{\phi\}$ are optimized by several losses at training time:

$$\mathcal{L} = \mathcal{L}_{\text{shape}} + \mathcal{L}_{\text{sketch}} + \mathcal{L}_{\text{align}}, \tag{4}$$

where $\mathcal{L}_{\text{shape}}$ ensures that we can accurately regress 3D shapes SDF from a 3D shape point cloud representation, $\mathcal{L}_{\text{sketch}}$ ensures that the reconstructed 3D shape is close to the input sketch, and $\mathcal{L}_{\text{align}}$ establishes a connection between sparse VR sketches and 3D shapes.

3.2.1 3D shape auto-encoder

First loss, $\mathcal{L}_{\text{shape}}$, operates only on 3D shape inputs, and ensures that the full network $\mathrm{D}_\theta([\mathrm{E}_\phi(\cdot), p_i])$ functions as a 3D shape autoencoder. First, we minimize the sum of L_1 losses between the truncated predicted and ground truth SDF values of sampled points $p_i \in \mathbb{R}^3$:

$$\mathcal{L}_{\text{SDF}}(\phi) = \frac{1}{N_s} \sum_{p_i} |\mathrm{tr}(\mathrm{D}_\theta([\boldsymbol{e}^g, p_i])) - \mathrm{tr}(s(p_i))| \tag{5}$$

where the decoder parameters θ are from our auto-decoder and are frozen when training the sketch/shape encoder; $s(\cdot)$ denotes ground-truth 3D shape SDF values, and $\mathrm{tr}(\cdot) \triangleq \min(\delta, \max(-\delta, \cdot))$.

Additionally, we ensure that the 3D shape embedding $\boldsymbol{e}^g = \mathrm{E}_\phi(f)$ maps directly to a latent representation of a 3D shape $\boldsymbol{d}^g$, computed with Eq. (3):

$$\mathcal{L}_{g\text{-}L_1}(\phi) = |\boldsymbol{e}^g - \boldsymbol{d}^g|. \tag{6}$$

We also found that adding contrastive latent term loss increases performance. Let's assume that our mini-batch consists of N_g shapes. First, we obtain latent representations $\{\boldsymbol{d}^g_1, \ldots, \boldsymbol{d}^g_{N_g}\}$ of 3D shapes, using Eq. (3). Then, we formulate our contrastive loss term as follows:

$$\mathcal{L}_{\text{g-NCE}}(\phi) = -\sum_{i=1}^{N_g} \left[\log \frac{\exp\left(-|\boldsymbol{e}^g_i - \boldsymbol{d}^g_i|\right)}{\sum_{j=1}^{N_g} \exp\left(-|\boldsymbol{e}^g_i - \boldsymbol{d}^g_j|\right)} \right]. \tag{7}$$

Therefore, the shape loss $\mathcal{L}_{\text{shape}}$ is the sum of the three losses defined above:

$$\mathcal{L}_{\text{shape}} = \mathcal{L}_{\text{SDF}}(\phi) + \mathcal{L}_{g\text{-}L_1}(\phi) + \mathcal{L}_{g\text{-NCE}}(\phi). \quad (8)$$

3.2.2 Sketch loss

Given the misalignment between sketches and reference 3D shapes in the dataset [33], as we show in Fig. 2, we aim for the reconstruction result to stay close to the sketch input. To achieve this goal, we design a sketch loss $\mathcal{L}_{\text{sketch}}$.

Since a sketch is a sparse representation of a 3D shape, the intended 3D shape surface should lie in the vicinity of sketch stroke points. Therefore, the reconstructed SDF values at those points should be close to zero. Formally, we define this loss as follows:

$$\mathcal{L}_{\text{sketch}}(\phi) = \frac{1}{N_s} \sum_{i=1}^{N_s} |D(\boldsymbol{e}^c, p_i^c)|, \quad (9)$$

where $p_i^c \in C$ is the i-th sample points from the conditioning sketch, and N_s is the number of sampled points in a sketch.

3.2.3 Sketch-shape latent space alignment

The considered so far losses do not explicitly ensure that there is a meaningful mapping between sketches and 3D shapes' latent representations. Therefore, we design additional losses encouraging an alignment in the feature space.

First, we introduce a contrastive loss, similar to the one in Eq. (10), leveraging that sketches in our dataset contain reference shapes. It takes the following form:

$$\mathcal{L}_{c\text{-NCE}}(\phi) = -\sum_{i=1}^{N_c} \left[\log \frac{\exp\left(-\left|\boldsymbol{e}_i^c - \boldsymbol{d}_i^g\right|\right)}{\sum_{j=1}^{N_g} \exp\left(-\left|\boldsymbol{e}_i^c - \boldsymbol{d}_j^g\right|\right)} \right], \quad (10)$$

where N_c is the number of sketches and N_g is the number of shapes in the mini-batch. This loss pulls the encodings of a sketch and a reference shape closer than the encodings of a sketch and non-matching 3D shapes.

Additionally, we minimize the L_1 distance between the sketch C embedding $\boldsymbol{e}^c = E_\phi(C)$ to the *ground-truth* shape latent code $\boldsymbol{d}^g$:

$$\mathcal{L}_{c\text{-}L_1}(\phi) = |\boldsymbol{e}^c - \boldsymbol{d}^g|. \quad (11)$$

Finally, the alignment loss is the sum of the two losses:

$$\mathcal{L}_{\text{align}} = \mathcal{L}_{c\text{-}L_1} + \mathcal{L}_{c\text{-NCE}} \quad (12)$$

3.3. Conditional shape generation

As shown in Fig. 2, a sparse sketch can represent multiple 3D shapes, generally following the sparse sketch strokes. Therefore, we would like to be able to generate multiple 3D shapes given a sketch. We achieve this by training a conditional normalizing flow (CNF) model in the latent space.

Specifically, we model the conditional distribution of shape embeddings using a RealNVP network [15] with five layers as in [43]. It transforms the probability distribution of shape feature embedding $p_{\boldsymbol{d}}(\boldsymbol{d}^g)$ to a unit Gaussian distribution $p_z(\boldsymbol{z})$. We obtain the sketch embedding vector $\boldsymbol{e}^c$ as described in the previous section, which serves as a condition for our normalizing flow model. Please note that the sketch encoder parameters ϕ are frozen at this stage. The sketch condition $\boldsymbol{e}^c$ is concatenated with the matching 3D shape feature vector $\boldsymbol{d}^g$ at each scale and translation coupling layers, following RealNVP [15]:

$$z^{1:d} = d^{g\,1:d} \quad \text{and} \quad (13)$$

$$z^{d+1:D} = d^{g\,d+1:D} \odot \exp\left(s\left(\left[\boldsymbol{e}^c; d^{g\,1:d}\right]\right)\right) + t\left(\left[\boldsymbol{e}^c; d^{g\,1:d}\right]\right) \quad (14)$$

where $s(\cdot)$ and $t(\cdot)$ are the scale and translation functions, parameterized by a neural network, as described in [15].

The idea of the normalizing flow model is to approximate a complicated probability distribution with a simple distribution through a sequence of invertible nonlinear transforms. We train the flow model by maximizing the log-likelihood $\log\left(p_{\boldsymbol{d}}\left(\boldsymbol{d}\right)\right)$:

$$\begin{aligned} \mathcal{L}_{\text{CNF}} &= -\log\left(p_{\boldsymbol{d}}\left(\boldsymbol{d}\right)\right) \\ &= -(\log\left(p_z\left(z\right)\right) + \log\left(\left|\det\left(\frac{\partial F\left(\boldsymbol{d}\right)}{\partial z^T}\right)\right|\right)), \end{aligned} \quad (15)$$

where $F(\cdot)$ is the normalizing flow model, and $\partial F\left(\boldsymbol{d}\right)/\partial z^T$ is the Jacobian of F at $\boldsymbol{d}$.

Sketch fidelity. To ensure the fidelity of the generated 3D shapes to an input sketch, we additionally train the flow model with a loss similar to an $\mathcal{L}_{sketch}$ loss (Eq. (9)).

First, the flow module F is updated with the gradients from $\mathcal{L}_{\text{CNF}}$. Then, for each sketch condition $\boldsymbol{e}^c$, we randomly sample K different noise vectors $\{z_k\}$ from the unit Gaussian distribution $z_k \in \mathcal{N}(0, 1)$, as shown in Fig. 3 (b). These noise vectors are mapped back to the shape embedding space through the reverse path of the flow model. During training, the obtained shape embeddings $\{\boldsymbol{d}_k^c\}$ are fed to the implicit decoder $D_\theta(\cdot)$ together with sketch stroke points p_i^c. Formally, this loss takes the following form:

$$\mathcal{L}_{\text{sketch,CNF}} = \frac{\lambda}{N_s K} \sum_{i=1}^{N_s} \sum_{k=1}^{K} |D\left(\boldsymbol{d}_k^c, p_i^c\right)|, \quad (16)$$

where N_s is a number of sketch stroke points, as before, and λ is a hyper-parameter set to 100 to increase the relative importance of this loss. In each mini-batch, we first propagate gradients from $\mathcal{L}_{\text{CNF}}$, and then from $\mathcal{L}_{\text{sketch,CNF}}$.

Conditional shape generation. During inference, given an input sketch, represented as a set of points, we first obtain its embedding $\boldsymbol{e}^c$ using the encoder $E_\phi(\cdot)$. We then condition the normalizing flow network with $\boldsymbol{e}^c$ and a random noise vector sampled from the unit Gaussian distribution to obtain a shape embedding $\boldsymbol{d}^c$. We obtain the mean embedding by using the mean of the normal distribution. Finally, this shape embedding is fed into implicit decoder $D_\theta(\cdot)$ to obtain a new set of SDF values $\{s^{d^c} | s^{d^c} = D(\boldsymbol{d}^c)\}$. A 3D geometry is then reconstructed by applying the marching cubes algorithm [30].

4. Experiments

4.1. Implementation Details

Auto-decoder We train a decoder, similar to [39], to regress the continuous SDF value for a given 3D space point and latent space feature vector. Our decoder $\mathrm{D}_\theta : \mathbb{R}^{(256+3)} \rightarrow \mathbb{R}$ consists of 5 feed-forward layers, each with dropouts. All internal layers are 512-dimensional and have ReLU non-linearities. The output layer uses tanh non-linearity to directly regress the continuous SDF scalar values. Similar to [39], we found training with batch normalization to be unstable and applied the weight-normalization technique.

During training, for each shape, we sample locations of the 3D points at which we calculate SDF values. We sample two sets of points: close to the shape surface and uniformly sampled in the unit box. Then, the loss is evaluated on the random subsets of those pre-computed points. During inference, the 3D points are sampled on a regular $(256 \times 256 \times 256)$ grid.

Encoder and Normalizing flow We train with an Adam optimizer, where for the encoder training the learning rate is set to $1e-3$, and for the normalizing flow model, it is set to $1e-5$. Training is done on 2 Nvidia A100 GPUs.

When training a sketch encoder jointly on sketches and shapes each mini-batch consists of 12 sketch-shape pairs and additional 24 shapes that do not have a paired sketch. When training CNF model, each mini-batch consists of 12 sketch-shape pairs.

We train the encoder and the conditional normalizing flow for 300 epochs each. The encoder is however trained in two steps. First, it is pre-trained using 3D shapes only, using $\mathcal{L}_{\text{shape}}$ loss, defined in Eq. (8). The performance of the shape reconstruction from this step is provided for reference in the 1st line in Tab. 1. The encoder is then fine-tuned using sketches and shapes with the full loss given by Eq. (4).

To train the sketch/shape encoder we sample $N_s = 4096$ points from sketch strokes and shape surface, respectively. Please refer to the supplemental for additional details.

4.2. Datasets

For training and testing, we use the only available fine-grained dataset of freehand VR sketches by Luo et al. [34][1]. The dataset consists of 1,005 sketch shape pairs for the chair category of ShapeNet [7]. We follow their split to training and test sets, containing 803 and 202 shape-sketch pairs, respectively. The 6,576 shapes from the ShapeNetCore-v2, non-overlapping with the 202 shapes in the test set, are used for training the auto-decoder and sketch/shape encoder.

Alignment of multiple data types: The sketches in the used dataset have a consistent orientation with reference 3D shapes, but might be not well aligned horizontally and vertically to the references, and can have a different scale. We sample shape point clouds and compute SDF values for the normalized 3D shapes as provided in ShapeNetCore-v2, which ensures consistency between the two 3D shape representations. We then normalize the sketches to fit a unit bounding box, following the normalization in the ShapeNetCore-v2. To further improve alignment between sketches and 3D shapes, we translate sketches, so that their centroids match the centroids of reference shapes.

4.3. Evaluation Metrics

Following prior work, we choose a bidirectional Chamfer distance (CD) as the similarity metric between two 3D shapes. CD measures the average shortest distance from one set of points to another. To compute CD, we randomly sample 4,096 points from 3D meshes.

Shape fidelity, $\mathcal{F}_{\text{shape}}(\cdot)$ First, we evaluate the ability of our auto-encoder to faithfully regress 3D shape SDF values given a 3D shape point cloud. We evaluate the fidelity of the regressed 3D shape, G^{e^g}, to the ground-truth 3D shape, G, as follows: $\mathcal{F}_{\text{shape}}(G^{e^g}) = CD(G, G^{e^g})$.

Then, while the sketches in the used dataset do not align perfectly with reference 3D shapes and contain ambiguity, it is meaningful to expect that the reconstructed 3D shape still should be close to the reference 3D shape. Therefore, we evaluate how close the reconstructed 3D shapes are to the ground-truth when (1) the shape is reconstructed from the sketch embedding $\boldsymbol{e}^c$, denoted as G^{e^c}; (2) the shape is reconstructed from the predicted conditional mean $\bar{z}^c$ of the CNF model, denoted as $G^{\bar{z}^c} = D(F^{-1}(\bar{z}^c))$; and (3) the shape is reconstructed from a random sample

[1] https://cvssp.org/data/VRChairSketch/

Method	Loss	$\mathcal{F}_{shape}(G^{e^c}) \downarrow$	$\mathcal{F}_{shape}(G^{e^g}) \downarrow$
Shape AE	$\mathcal{L}_{\text{shape}}$	0.834	**0.110**
Sketch AE	$\mathcal{L}_{\text{SDF}}$	0.437	0.581
	$\mathcal{L}_{g\text{-}L_1} + \mathcal{L}_{\text{sketch}}$	0.504	0.321
Joint AE	$\mathcal{L}$	**0.357**	0.126

Table 1. Evaluation of auto-encoder training strategies with respect to the fidelity ($\mathcal{F}_{shape}(\cdot)$) of the reconstructed 3D shapes to the reference/ground-truth 3D shapes, depending on the used data. Here, G^{e^c} and G^{e^g} are reconstructions from an input sketch and 3D shape, respectively. Shape AE, Sketch AE, and Joint AE stand for training encoder only with shape inputs, sketch inputs, or both, respectively.

Method	$\mathcal{F}_{shape}(G^{e^c}) \downarrow$	$\mathcal{F}_{shape}(G^{e^g}) \downarrow$
$\mathcal{L}_{\text{L1}}$	0.418	0.199
$\mathcal{L}_{\text{L1}} + \mathcal{L}_{\text{SDF}}$	0.374	0.140
$\mathcal{L}_{\text{L1}} + \mathcal{L}_{\text{SDF}} + \mathcal{L}_{\text{NCE}}$	0.373	**0.126**
$\mathcal{L}_{\text{L1}} + \mathcal{L}_{\text{SDF}} + \mathcal{L}_{\text{NCE}} + \mathcal{L}_{\text{sketch}}$	**0.357**	**0.126**

Table 2. Evaluation of auto-encoder training strategies with respect to the fidelity ($\mathcal{F}_{shape}(\cdot)$) of the reconstructed 3D shapes to the reference/ground-truth 3D shapes, depending on the used loss function. Here, we group together sketch and shape L_1 and NCE losses.

from the latent space of the CNF model, denoted as G^{d^c}. The respective losses are: $\mathcal{F}_{\text{shape}}(G^{e^c})$, $\mathcal{F}_{\text{shape}}(G^{z^c})$ and $\mathcal{F}^{\text{avg}}_{\text{shape}}(G^{d^c}) = \sum_{i=1}^{5} \mathcal{F}_{\text{shape}}(G^{d^c}_i)$, where in the latter case we generate 5 samples and report an average loss value.

Sketch fidelity, $\mathcal{F}_{\text{sketch}}(\cdot)$ Since the used sketches are ambiguous and are not perfectly aligned to a reference, we evaluate the fidelity of the reconstructions to the sketch input, using the loss similar to Eq. (9): $\mathcal{F}_{\text{sketch}}(G^c) = \frac{1}{N_s} \sum_{i=1}^{N_s} s^c(p^c_i)$, where p^c_i is the i-th sample point from the input sketch and s^c denotes the predicted SDF. With that, we define $\mathcal{F}^{avg}_{\text{sketch}}(s^{d^c}) = \frac{1}{5} \sum_{i=1}^{5} \mathcal{F}_{\text{sketch}}(s^{d^c}_i)$, as the average fidelity of multiple samples from the CNF model space to an input sketch.

Diversity, $\mathcal{D}_{\text{gnrtns}}$ To measure the diversity of the generated shapes, we formulate the pair-wise similarity of generated shapes using CD. Specifically, for any two generated shapes conditioned on the same sketch, we compute their CD, and finally report the mean of all pairs, which we refer to as $\mathcal{D}_{\text{gnrtns}}$.

4.4. Results

We first evaluate the reconstruction performance of our AE and then evaluate multiple shape generation, conditioned on the input sketch. Fig. 4 shows qualitative results for both stages.

4.4.1 Deterministic sketch to shape generation

Our first goal is to learn to map sketches to 3D shapes in a deterministic fashion. One of the challenges in our work comes from the limited dataset size, which is a common factor that should be taken into consideration when working with freehand sketches. Therefore, we proposed training the sketch-to-shape auto-encoder in multiple steps, and in addition, we propose to use a joint auto-encoder, and we use 3D shapes without paired sketches in the NCE loss to improve robustness. Tab. 1 shows that our strategy indeed outperforms alternative strategies. It allows reconstructing 3D shapes similar to reference 3D shapes, as shown by $\mathcal{F}_{shape}(G^{e^c})$. The fact that $\mathcal{F}_{shape}(G^{e^g})$ stays low in our proposed design implies that if the sketch is very detailed and accurate, we will obtain careful 3D shape reconstructions.

Tab. 2 demonstrates the importance of individual loss terms. It shows that the shape reconstruction loss $\mathcal{L}_{\text{SDF}}$ ensures that we can reconstruct shapes well when the input is dense (the case for the shape point cloud or very detailed sketches). The sketch fidelity loss $\mathcal{L}_{\text{sketch}}$ ensures that the reconstructed shape is following the structure of an input sketch. Finally, NCE losses improve both the sketch and shape fidelity criteria of the reconstructed results.

4.4.2 Conditional sketch to shape generation

Next, we conduct a number of experiments to assess the proposed conditional generation framework.

Shape encoding choices Note that when training CNF model, we use the shape latent code, $\boldsymbol{d}^g$, obtained via an inversion process with Eq. (3). Tab. 3, lines 2 and 3, shows that this allows to greatly increase the diversity of the generated results compared to using latent shape codes, $\boldsymbol{e}^g$, obtained from the encoder. This comes with a small decrease in fidelity to the reference shape, while the fidelity to the sketch increases a little bit. This result reinforces our design choice of training the auto-decoder first, providing a richer latent space.

Sketch fidelity loss in CNF model Tab. 3, lines 3 and 4, show that sketch consistency loss $\mathcal{L}_{\text{sketch, CNF}}$ results in much better sketch fidelity while maintaining comparable diversity. Varying the number of samples K from the CNF latent space, we can further adjust the balance between sketch fidelity and diversity (Tab. 3, lines 4-6). We use the model with $K = 8$ samples for the visual results in all our figures. The advantage of this loss is demonstrated visually in Fig. 5. It can be observed that the proposed loss encourages

Method	K	$\mathcal{F}_{shape}(G^{e^c}) \downarrow$	$\mathcal{F}^{avg}_{\text{sketch}}(G^{d^c}) \downarrow$	$\mathcal{F}^{\text{avg}}_{\text{shape}}(G^{d^c}) \downarrow$	$\mathcal{D}_{\text{gnrtns}} \uparrow$
Joint AE	-	**0.357**	-	-	-
CNF(e^g)	-	0.373	0.026±0.036	**0.380**	0.043
CNF(d^g)	-	0.385	0.030±0.039	0.420	**0.165**
CNF(d^g) + $\mathcal{L}_{\text{sketch,CNF}}$	1	0.366	0.019±0.034	0.422	0.158
	4	0.397	0.018±0.034	0.448	**0.165**
	8	0.368	**0.017±0.034**	0.431	0.161

Table 3. Ablation of design choices for the CNF model. 'Joint AE' stands for the result of our autoencoder model, and provides the estimated fidelity of the deterministic reconstruction to a sketch reference 3D shape. 'CNF' stands for a conditional normalizing flow. K refers to the number of samples used to compute $\mathcal{L}_{\text{sketch,CNF}}$ during training. $\mathcal{F}^{avg}_{\text{sketch}}$ measures the average fidelity of the reconstructed 3D shape samples to an input sketch. $\mathcal{F}^{\text{avg}}_{\text{shape}}$ measures the average fidelity of the reconstructed 3D shape samples to a reference shape. $\mathcal{D}_{\text{gnrtns}}$ measures the diversity of the the reconstructed 3D shape samples. All fidelity measures are multiplied by $1e2$.

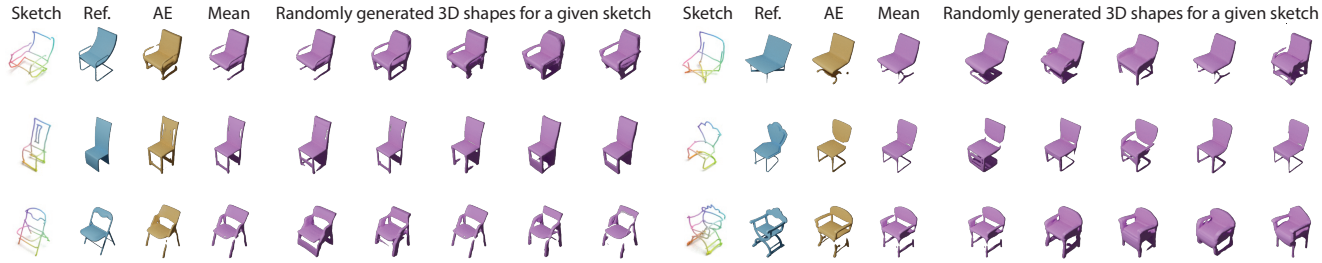

Figure 4. Generation results. 'Ref.' shows the reference 3D shape. 'AE' shows the deterministic prediction by our AE from the first stage of our method. 'Mean' denotes the shape reconstructed from the sample corresponding to the mean of the conditional distribution. And finally, we show 5 randomly generated shapes conditioned on the input sketch, sorted in the order of fidelity to a reference shape.

the network to always reconstruct some shape structure near the sketch strokes.

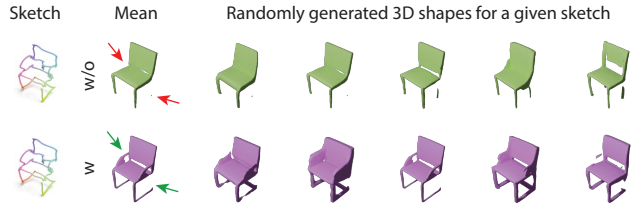

Figure 5. Comparison of the generated samples conditioned on the input sketch when $\mathcal{L}_{\text{sketch, CNF}}$ is used (purple) or not (green). This example shows that the sketch fidelity loss indeed results in better fidelity to a sketch input: all generated shapes when the loss is used contain handles and better respect the shape of chair legs/support. 'Mean' denotes the shape reconstructed from the sample corresponding to the mean of the conditional distribution.

4.5. Comparison to retrieval

Fig. 6 shows comparison to the retrieval results by the state-of-the-art method [32] that is designed to retrieve structurally-similar shapes. It can be observed that generation can be more robust to shapes that are not common shapes in a 3D shape gallery. However, the reconstruction quality of our method is limited, and some shapes still do not look like real-world shapes, missing details.

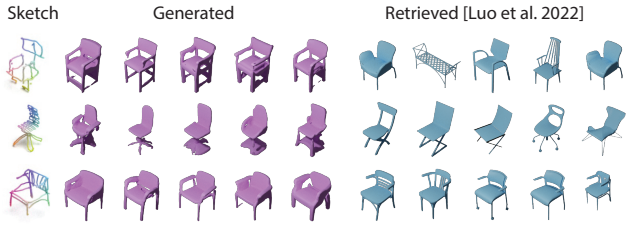

Figure 6. Comparison to the retrieval results by Luo et al. [32].

5. Conclusion and Discussion

We present the first method for multiple 3D shape generation conditioned on sparse and abstract sketches. We achieve good fidelity to the input sketch combined with the shape diversity of the generated results. In our work, we show how to efficiently overcome the limitation of small datasets. Our experiments are currently limited to a single category, but none of the components of our method explicitly exploits any priors about this category. In the future, we would like to extend this work by (1) further improving the input sketch fidelity, potentially taking perceptual multi-view losses into account during training, and (2) considering alternative shape representation for our auto-decoder.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICML*, 2018. 3

[2] Himanshu Arora, Saurabh Mishra, Shichong Peng, Ke Li, and Ali Mahdavi-Amiri. Multimodal shape completion via imle. *arXiv:2106.16237*, 2021. 3

[3] Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction from point clouds. In *Comput. Graph. Forum*, 2017. 3

[4] Sukanya Bhattacharjee and Parag Chaudhuri. A survey on sketch based content creation: from the desktop to virtual and augmented reality. In *Comput. Graph. Forum*, 2020. 1, 3

[5] Alexandra Bonnici, Alican Akman, Gabriel Calleja, Kenneth P Camilleri, Patrick Fehling, Alfredo Ferreira, Florian Hermuth, Johann Habakuk Israel, Tom Landwehr, Juncheng Liu, et al. Sketch-based interaction and modeling: Where do we stand? *AI EDAM*, 2019. 1

[6] Jorge D Camba, Pedro Company, and Ferran Naya. Sketch-based modeling in mechanical engineering design: Current status and opportunities. *Computer-Aided Design*, 2022. 1, 3

[7] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv:1512.03012*, 2015. 6

[8] Zhiqin Chen, Vladimir G Kim, Matthew Fisher, Noam Aigerman, Hao Zhang, and Siddhartha Chaudhuri. Decor-gan: 3D shape detailization by conditional refinement. In *CVPR*, 2021. 3

[9] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019. 3

[10] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander Schwing, and Liangyan Gui. Sdfusion: Multimodal 3D shape completion, reconstruction, and generation. *arXiv:2212.04493*, 2022. 3

[11] Zezhou Cheng, Menglei Chai, Jian Ren, Hsin-Ying Lee, Kyle Olszewski, Zeng Huang, Subhransu Maji, and Sergey Tulyakov. Cross-modal 3D shape generation and manipulation. In *ECCV*, 2022. 3

[12] Pinaki Nath Chowdhury, Aneeshan Sain, Ayan Kumar Bhunia, Tao Xiang, Yulia Gryaditskaya, and Yi-Zhe Song. Fs-coco: towards understanding of freehand sketches of common objects in context. In *ECCV*, 2022. 1

[13] Johanna Delanoy, Mathieu Aubry, Phillip Isola, Alexei A Efros, and Adrien Bousseau. 3D sketching using multi-view deep volumetric prediction. *ACM on Computer Graphics and Interactive Techniques*, 2017. 1, 3

[14] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv:1410.8516*, 2014. 2

[15] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv:1605.08803*, 2016. 5

[16] Rao Fu, Xiao Zhan, Yiwen Chen, Daniel Ritchie, and Srinath Sridhar. Shapecrafter: A recursive text-conditioned 3D shape generation model. *arXiv:2207.09446*, 2022. 1, 3

[17] Benoit Guillard, Edoardo Remelli, Pierre Yvernay, and Pascal Fua. Sketch2mesh: Reconstructing and editing 3D shapes from sketches. In *ICCV*, 2021. 1, 3

[18] Xian-Feng Han, Hamid Laga, and Mohammed Bennamoun. Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era. *IEEE TPAMI*, 2019. 3

[19] Zhiming Hu. Gaze analysis and prediction in virtual reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, 2020. 1

[20] Jason Jerald. *The VR book: Human-centered design for virtual reality*. 2015. 1

[21] Di Kong, Qiang Wang, and Yonggang Qi. A diffusion-refinement model for sketch-to-point modeling. In *ACCV*, 2022. 1, 3

[22] Bo Li, Yijuan Lu, Fuqing Duan, Shuilong Dong, Yachun Fan, Lu Qian, Hamid Laga, Haisheng Li, Yuxiang Li, Peng Liu, et al. SHREC'16 Track: 3D Sketch-Based 3D Shape Retrieval. In *Proceedings of the Eurographics 2016 Workshop on 3D Object Retrieval*, 2016. 1, 2

[23] Bo Li, Yijuan Lu, Azeem Ghumman, Bradley Strylowski, Mario Gutierrez, Safiyah Sadiq, Scott Forster, Natacha Feola, and Travis Bugerin. 3D sketch-based 3D model retrieval. In *ACM ICMR*, 2015. 1, 2

[24] Bo Li, Yijuan Lu, Azeem Ghumman, Bradley Strylowski, Mario Gutierrez, Safiyah Sadiq, Scott Forster, Natacha Feola, and Travis Bugerin. KinectSBR: A kinect-assisted 3D sketch-based 3D model retrieval system. In *ACM ICMR*, 2015. 1, 2

[25] Changjian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. Robust flow-guided neural prediction for sketch-based freeform surface modeling. *ACM TOG*, 2018. 1, 3

[26] Gang Li, Heliang Zheng, Chaoyue Wang, Chang Li, Changwen Zheng, and Dacheng Tao. 3ddesigner: Towards photorealistic 3D object generation and editing with text-guided diffusion models. *arXiv:2211.14108*, 2022. 1

[27] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3D content creation. *arXiv:2211.10440*, 2022. 1

[28] Zhengzhe Liu, Peng Dai, Ruihui Li, Xiaojuan Qi, and Chi-Wing Fu. ISS: Image as stepping stone for text-guided 3D shape generation. *arXiv:2209.04145*, 2022. 1

[29] Zhengzhe Liu, Yi Wang, Xiaojuan Qi, and Chi-Wing Fu. Towards implicit text-guided 3D shape generation. In *CVPR*, 2022. 3

[30] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM siggraph computer graphics*, 1987. 3, 6

[31] Zhaoliang Lun, Matheus Gadelha, Evangelos Kalogerakis, Subhransu Maji, and Rui Wang. 3D shape reconstruction from sketches via multi-view convolutional networks. In *3DV*, 2017. 1, 3

[32] Ling Luo, Yulia Gryaditskaya, Tao Xiang, and Yi-Zhe Song. Structure-aware 3D VR sketch to 3D shape retrieval. In *3DV*, 2022. 1, 2, 8

[33] Ling Luo, Yulia Gryaditskaya, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. Towards 3D VR-sketch to 3D shape retrieval. In *3DV*, 2020. 1, 2, 5

[34] Ling Luo, Yulia Gryaditskaya, Yongxin Yang, Tao Xiang, and Yi-Zhe Song. Fine-grained vr sketching: Dataset and insights. In *3DV*, 2021. 1, 2, 6

[35] Daniel Martin, Ana Serrano, Alexander W Bergman, Gordon Wetzstein, and Belen Masia. Scangan360: A generative model of realistic scanpaths for 360 images. *IEEE TVCG*, 2022. 1

[36] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3D completion, reconstruction and generation. In *CVPR*, 2022. 3

[37] Gimin Nam, Mariem Khlifi, Andrew Rodriguez, Alberto Tono, Linqi Zhou, and Paul Guerrero. 3D-ldm: Neural implicit 3d shape generation with latent diffusion models. *arXiv:2212.00842*, 2022. 3

[38] Luke Olsen, Faramarz F Samavati, Mario Costa Sousa, and Joaquim A Jorge. Sketch-based modeling: A survey. *Computers & Graphics*, 2009. 1

[39] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019. 2, 3, 6

[40] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3D using 2d diffusion. *arXiv:2209.14988*, 2022. 3

[41] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *NeurIPS*, 2017. 4

[42] Enrique Rosales, Jafet Rodriguez, and ALLA SHEFFER. Surfacebrush: from virtual reality drawings to manifold surfaces. *ACM TOG*, 2019. 1

[43] Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, Marco Fumero, and Kamal Rahimi Malekshan. Clip-forge: Towards zero-shot text-to-shape generation. In *CVPR*, 2022. 1, 3, 5

[44] Patsorn Sangkloy, Wittawat Jitkrittum, Diyi Yang, and James Hays. A sketch is worth a thousand words: Image retrieval with text and sketch. In *ECCV*, 2022. 1

[45] Yongbin Sun, Yue Wang, Ziwei Liu, Joshua Siegel, and Sanjay Sarma. Pointgrow: Autoregressively learned point cloud generation with self-attention. In *WACV*, 2020. 3

[46] Jiayun Wang, Jierui Lin, Qian Yu, Runtao Liu, Yubei Chen, and Stella X Yu. 3d shape reconstruction from free-hand sketches. In *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VIII*. Springer, 2023. 1, 3

[47] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. In *3DV*, 2021. 3

[48] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. *NeurIPS*, 2016. 3

[49] Rundi Wu, Xuelin Chen, Yixin Zhuang, and Baoquan Chen. Multimodal shape completion via conditional generative adversarial networks. In *ECCV*, 2020. 3

[50] Xingguang Yan, Liqiang Lin, Niloy J Mitra, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Shapeformer: Transformer-based shape completion via sparse representation. In *CVPR*, 2022. 3

[51] Yuxiang Ye, Bo Li, and Yijuan Lu. 3D sketch-based 3D model retrieval with convolutional neural network. In *ICPR*, 2016. 1, 2

[52] Emilie Yu, Rahul Arora, J Andreas Baerentzen, Karan Singh, and Adrien Bousseau. Piecewise-smooth surface fitting onto unstructured 3D sketches. *ACM TOG*, 2022. 1, 2

[53] Qian Yu, Feng Liu, Yi-Zhe Song, Tao Xiang, Timothy M Hospedales, and Chen-Change Loy. Sketch me that shoe. In *CVPR*, 2016. 1

[54] Xue Yu, Stephen DiVerdi, Akshay Sharma, and Yotam Gingold. Scaffoldsketch: Accurate industrial design drawing in vr. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, 2021. 1

[55] Song-Hai Zhang, Yuan-Chen Guo, and Qing-Wen Gu. Sketch2model: View-aware 3D modeling from single free-hand sketches. In *CVPR*, 2021. 1, 3

[56] X Zheng, Yang Liu, P Wang, and Xin Tong. SDF-StyleGAN: Implicit sdf-based stylegan for 3D shape generation. In *Comput. Graph. Forum*, 2022. 3

[57] Yue Zhong, Yulia Gryaditskaya, Honggang Zhang, and Yi-Zhe Song. Deep sketch-based modeling: Tips and tricks. In *3DV*, 2020. 1, 3

[58] Yue Zhong, Yonggang Qi, Yulia Gryaditskaya, Honggang Zhang, and Yi-Zhe Song. Towards practical sketch-based 3D shape generation: The role of professional sketches. *IEEE TCSVT*, 2020. 1, 3

[59] Linqi Zhou, Yilun Du, and Jiajun Wu. 3D shape generation and completion through point-voxel diffusion. In *ICCV*, 2021. 3