Cross-Modal 3D Shape Generation and Manipulation

Zezhou Cheng[1⋆], Menglei Chai[2], Jian Ren[2], Hsin-Ying Lee[2], Kyle Olszewski[2], Zeng Huang[2], Subhransu Maji[1], and Sergey Tulyakov[2]

[1] University of Massachusetts, Amherst
[2] Snap Inc.

Abstract. Creating and editing the shape and color of 3D objects require tremendous human effort and expertise. Compared to direct manipulation in 3D interfaces, 2D interactions such as sketches and scribbles are usually much more natural and intuitive for the users. In this paper, we propose a generic multi-modal generative model that couples the 2D modalities and implicit 3D representations through shared latent spaces. With the proposed model, versatile 3D generation and manipulation are enabled by simply propagating the editing from a specific 2D controlling modality through the latent spaces. For example, editing the 3D shape by drawing a sketch, re-colorizing the 3D surface via painting color scribbles on the 2D rendering, or generating 3D shapes of a certain category given one or a few reference images. Unlike prior works, our model does not require re-training or fine-tuning per editing task and is also conceptually simple, easy to implement, robust to input domain shifts, and flexible to diverse reconstruction on partial 2D inputs. We evaluate our framework on two representative 2D modalities of grayscale line sketches and rendered color images, and demonstrate that our method enables various shape manipulation and generation tasks with these 2D modalities.

1 Introduction

With the growth in 3D acquisition and visualization technology, there is an increasing need of tools for 3D content creation and editing tasks such as deforming the shape of an object, changing the color of a part, or inserting or removing a component. The graphics and vision community has proposed a number of tools for these tasks [46,13,2,41]. Yet, manipulating 3D still requires tremendous human labor and expertise, prohibiting wide-scale adoption by non-professionals. Compared to the traditional 3D user interfaces, 2D interactions on view-dependent image planes can be a more intuitive way to edit the shape. This has motivated the community to leverage advances in shape representations using deep networks [40,37,9,50] for 3D shape manipulation with 2D controls, such as mesh reconstruction from sketches [20] and color editing with scribbles [36].

⋆ This work was mainly done while the first author was an intern at Snap Inc. Code and data are available at https://people.cs.umass.edu/~zezhoucheng/edit3d/

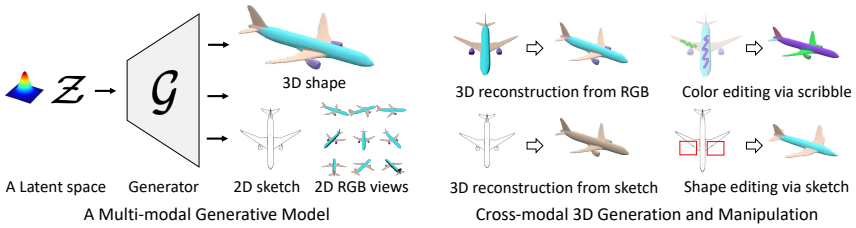

Fig. 1. We propose a multi-modal generative model that bridges multiple 2D (*e.g.*, sketch, color views) and 3D modalities via shared latent spaces (*left*). Versatile 3D shape generation and manipulation tasks can be tackled via simple latent optimization method (*right*).

However, most prior works on 2D-to-3D shape manipulation are tailored to a particular editing task and interaction format, which makes generalization to new editing tasks or controls challenging, or even infeasible. This is important because there is often no single interaction that fits every use case – the preferred 2D user control depends on the editing goals, scenarios, devices, or targeted users.

Motivated by this, we propose a 2D-to-3D framework that not only works on a single control modality but also enjoys the flexibility of handling various types of 2D interactions without the need for changing the architecture or even re-training (Fig. 1 left). Our framework bridges various 2D interaction modalities and the target 3D shape through a uniform editing propagation mechanism. The key is to construct a shared latent representation across generative models of each of the 2D and 3D modalities. The shared latent representation enforces that an arbitrary latent code corresponds to a 3D model that is consistent with every modality, in terms of both shape and color. With our model, any editing can be achieved by an objective that aims to match the corresponding editing modality and backpropagating the error to estimate the latent code. Moreover, different editing operations and modalities can be combined and interleaved leading to a versatile tool for editing the shape (Fig. 1 right). The approach can be extended to a new user control by simply adding a generator for the corresponding modality in the framework.

We evaluate our framework on two representative 2D modalities, *i.e.*, grayscale line sketches, and rendered color images. We provide extensive quantitative and qualitative results in shape and color editing with sketches and scribbles, as well as single-view, few-shot, or even partial-view cross-modal shape generation. The proposed method is conceptually simple, easy to implement, robust to input domain shifts, and generalizable to new modalities with no special requirement on the network architecture.

2 Related Work

Multi-Modal Generative Models. There has been much work on learning a joint distribution of multiple modalities $p(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_n)$ where each modality $\boldsymbol{x}_i$ rep-

Table 1. Comparisons to cross-modal 3D editing and generation works.

Methods	*Manipulation*		*Generation*		
	Shape	Color	Single view	Partial view	Few shot
Sketch2Mesh [20]	✓	✗	✓	✗	✗
DualSDF [22]	✓	✗	✗	✗	✗
EditNeRF [36]	✓	✓	✗	✗	✗
Ours	✓	✓	✓	✓	✓

resents one representation (*e.g.*, images, text) of underlying signals. Multi-modal VAEs [52,56,57,49,29] learn a joint distribution $p_\theta(\boldsymbol{x}_0, \ldots, \boldsymbol{x}_n \mid \boldsymbol{z})$ conditioned on common latent variables $\boldsymbol{z} \in \mathcal{Z}$. Without the assumption of paired multi-modal data, multi-modal GANs [33,10,17] learn the joint distribution by sharing a latent space and model parameters across modalities. These multi-modal generative models have enabled versatile applications such as cross-modal image translation [10,33] and domain adaptation [33]. Similar to these works, we build a multi-modal generative model that bridges multiple modalities via a shared latent space. However, we generate and edit 3D shapes with sparse 2D inputs (*e.g.*, scribbles, sketches) and build a 2D-3D generative model based on variational auto-decoders (VADs) [60,22]. Prior work [60] has shown that VADs excel at generative modeling from incomplete data. In this work, we demonstrate that the multi-modal VADs (MM-VADs) are ideally suited for the task of 3D generation and manipulation from sparse 2D inputs (*e.g.*, color scribble or partial inputs).

Shape and Appearance Reconstruction. Extensive works have explored the problem of 3D reconstruction from different modalities, such as RGB images [27,11], videos [59], sketches [26,20,64,63], or even text [8]. This problem has also been explored under diverse representations [11,15,35,54,14,40,9,37,50,58] and different levels of supervision [11,15,27,16,59]. Despite the diverse settings of this problem, the encoder-decoder network, which maps the source modalities to 3D shape directly in a feed-forward manner, remains the most popular 3D reconstruction model [11,54,27,40]. However, such feed-forward networks are not robust to input domain shift (*e.g.*, incomplete data). In this work, we demonstrate that the proposed MM-VADs perform more robustly and could provide multiple 3D reconstructions that fit the given input (*e.g.*, partial 2D views).

Shape and Appearance Manipulation. Numerous interactive tools have been developed for image editing [31,62,44,32,18,30] and 3D shape manipulations [46,13,2,41]. More recently, generative modeling of natural images [17,51] has became a "Swiss knife" for image editing problems [65,48,47,19,1,4,5,39,45]. Similar to these works, we build a multi-modal generative model that is able to tackle versatile 3D shape generation and editing tasks with 2D inputs. Novel interactive tools have also been proposed recently to edit implicit 3D representations [40,38]. For example, DualSDF [22] edits the SDFs [40] via shape primitives (*e.g.*, spheres). Sketch2Mesh [20] reconstructs shapes from sketch with an encoder-decoder network and refines 3D shapes via differentiable rendering.

EditNeRF[36] edits the radiance field [38] by fine-tuning the network weights based on user's scribbles.

Tab. 5 summarizes the commons and differences between our work and recent efforts [22,20,36] on 3D manipulation and generation. Similar to Sketch2Mesh [20], we edit and reconstruct 3D shape from 2D sketch. However, we tackle this problem via a novel multi-modal *generative* model that performs more robust to input domain shift (*e.g.*, partial input, sparse color scribble). Furthermore, the shape and color edits can be combined and interleaved with our model; Like EditNeRF, we edit the appearance of 3D shapes via 2D color scribbles. However, we conduct the 3D editing via a simple latent optimization, instead of finetuning the network weights per edit; Akin to DualSDF [22], we build a generative model for 3D manipulation, yet we generate and edit shapes from 2D modalities which is more intuitive to edit the shape than using 3D primitives. Moreover, our generative model can be adapted to generate 3D shapes of a certain category (*e.g.*, armchairs) given a few 2D examples, namely, *few-shot cross-modal shape generation*.

3 Method

We describe the Variational Auto-Decoders (VADs) [60] in § 3.1, introduce the proposed VAD-based multi-modal generative model (dubbed MM-VADs) in § 3.2, and illustrate the application of MM-VADs in cross-modal 3D shape generation and manipulation tasks in § 3.3.

3.1 Background: Variational Auto-Decoder

Given observation variables $\boldsymbol{x} \sim p(\boldsymbol{x})$ and latent variables $\boldsymbol{z} \sim p(\boldsymbol{z})$, a variational auto-decoder (VAD) approximates the data distribution $p(\boldsymbol{x})$ via a parametric family of distributions $p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$ with parameters θ. Similar to variational auto-encoders (VAEs) [29], VADs are trained by maximizing the marginal distribution $p(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z}$. In practice this integral is expensive or intractable, so the model parameters θ are learned instead by maximizing the Evidence Lower Bound (ELBO):

$$\mathcal{V}(\phi, \theta \mid \boldsymbol{x}) = -\operatorname{KL}\big(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z})\big) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\big[\log p_\theta(\boldsymbol{x} \mid \boldsymbol{z})\big], \tag{1}$$

where $\operatorname{KL}(\cdot \parallel \cdot)$ is the Kullback-Leibler divergence that encourages the posterior distribution to follow the latent prior $p(\boldsymbol{z})$, and $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ is an approximation of the posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$. In VAEs, $q_\phi(\boldsymbol{z} \mid \boldsymbol{x})$ is parametrized by a neural network and ϕ are the parameters of the encoder. In VADs, ϕ are instead learnable similar to the parameters θ in the decoder $p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$. For example, the multivariate Gaussian approximate posterior for a data instance $\boldsymbol{x}_i$ is defined as:

$$q_\phi(\boldsymbol{z} \mid \boldsymbol{x}_i) \coloneqq \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2}$$

where $\phi = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}$. The reparametrization trick is applied in order to back-propagate the gradients to the mean $\boldsymbol{\mu}_i$ and variance $\boldsymbol{\Sigma}_i$ in VADs. In comparison,

VAEs back-propagate the gradients through the mean $\boldsymbol{\mu}_i$ and variance $\boldsymbol{\Sigma}_i$ to learn the parameters of the encoder. At inference time, the parameters ϕ of the approximate posterior distribution can be estimated by maximizing the ELBO in Eqn. 1 while the parameters θ of the decoder are frozen:

$$\phi^* = \arg\max_{\phi} \ \mathcal{V}(\phi \mid \theta, \boldsymbol{x}_i). \tag{3}$$

Despite the similarity between VAEs and VADs, prior works [60] demonstrate that VADs perform approximate posterior inference more robustly on *incomplete data* and *input domain shifts* than VAEs.

3.2 Multi-Modal Variational Auto-Decoder

We consider two modalities $\boldsymbol{x}, \boldsymbol{w}$ and an *i.i.d.* dataset with paired instances $(\boldsymbol{X}, \boldsymbol{W}) = \{(\boldsymbol{x}_0, \boldsymbol{w}_0), \ldots, (\boldsymbol{x}_N, \boldsymbol{w}_N)\}$. We target at learning a joint distribution of both modalities $p(\boldsymbol{x}, \boldsymbol{w})$. Like VADs [60], the multi-modal VADs (MM-VADs) are trained by maximizing the ELBO:

$$\mathcal{V}(\phi, \theta \mid \boldsymbol{x}, \boldsymbol{w}) = -\operatorname{KL}\big(q_\phi(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{w}) \parallel p(\boldsymbol{z})\big) + \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{w})}\big[\log p_\theta(\boldsymbol{x}, \boldsymbol{w} \mid \boldsymbol{z})\big], \tag{4}$$

where $\boldsymbol{z}$ is the latent variable shared by the two modalities $\boldsymbol{x}$ and $\boldsymbol{w}$, $p_\theta(\boldsymbol{x}, \boldsymbol{w} \mid \boldsymbol{z}) = p_{\theta_x}(\boldsymbol{x} \mid \boldsymbol{z}) p_{\theta_w}(\boldsymbol{w} \mid \boldsymbol{z})$ under the assumption that the two modalities $\boldsymbol{x}$ and $\boldsymbol{w}$ are independent conditioned on the latent variable $\boldsymbol{z}$ (*i.e.*, $\boldsymbol{x} \perp\!\!\!\perp \boldsymbol{w} \mid \boldsymbol{z}$). In practice, $p_{\theta_x}(\boldsymbol{x} \mid \boldsymbol{z})$ or $p_{\theta_w}(\boldsymbol{w} \mid \boldsymbol{z})$ can be parameterized by different networks for the two modalities $\boldsymbol{x}$ and $\boldsymbol{w}$ respectively. The parameters ϕ of the approximate posterior distribution $q_\phi(\boldsymbol{z} \mid \boldsymbol{x}, \boldsymbol{w})$ are learnable parameters where $\phi = \{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ under the assumption of multivariate Gaussian posterior distribution. At inference time, the parameters ϕ are estimated via maximizing the ELBO with frozen decoder parameters θ:

$$\phi^* = \arg\max_{\phi} \ \mathcal{V}(\phi \mid \theta, \boldsymbol{x}_i, \boldsymbol{w}_i). \tag{5}$$

When one of the modalities is missing during inference, the inputs of the missing modalities are simply set to zero. This is the case when we want to infer one modality from the other (*e.g.*, 3D reconstruction from 2D sketch). This framework can be trivially extended to learn a joint distribution of more than two modalities.

3.3 Learning a Joint 2D-3D Prior with MM-VADs

Here we introduce the application of MM-VADs in cross-modal 3D shape generation and manipulation. Specifically, we learn a joint distribution of 2D and 3D modalities with MM-VADs. Once trained, MM-VADs can be applied to versatile shape generation and editing tasks via a simple posterior inference (or latent optimization). We explore three representative modalities, including 3D shape with colorful surface, 2D sketch in grayscale, and 2D rendered image in

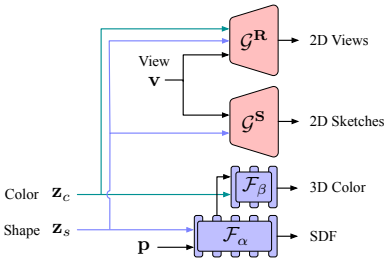

Fig. 2. Network architecture. We propose a multi-modal variational auto-decoder consisting of compact shape and color latent spaces shared across multiple 2D (*e.g.*, sketch, RGB views) or 3D modalities (*e.g.*, signed distance function and 3D surface color).

RGB color, donated as $\boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R}$ respectively. Given a dataset $\{(\boldsymbol{C}_i, \boldsymbol{S}_i, \boldsymbol{R}_i)\}$, we target at learning a joint distribution of the three modalities $p(\boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R})$. Fig. 2 presents the overview of the MM-VADs framework. We provide more details in the following sections.

Joint Latent Space. The MM-VADs share a common latent space $\mathcal{Z}$ across different modalities (Eqn. 4). Targeting at editing 3D shape and surface color independently, we further disentangle the shared latent space into the shape and color subspaces, denoted as $\mathcal{Z}_s$ and $\mathcal{Z}_c$ respectively. Therefore, each latent code $\boldsymbol{z} = \boldsymbol{z}_s \oplus \boldsymbol{z}_c$, where $\boldsymbol{z}_s \in \mathcal{Z}_s$, $\boldsymbol{z}_c \in \mathcal{Z}_c$, and $\oplus$ denotes the concatenation operator.

3D Colorful Shape. Targeting at generating and editing 3D shapes and their appearance, we use the 3D colorful shape as one of our modalities. Among various representations of 3D shapes (*e.g.*, voxel, mesh, point clouds), the implicit representations [40,37,50] model 3D shapes as isosurfaces of functions and are capable of capturing high-level details. We adopt the DeepSDF [40] to regress the signed distance functions (SDFs) from point samples directly using a MLP-based *3D shape network* $\mathcal{F}_\alpha(\boldsymbol{z}_s \oplus \boldsymbol{p})$, whose input is a shape latent code $\boldsymbol{z}_s \in \mathcal{Z}_s$ and 3D coordinates $\boldsymbol{p} \in \mathbb{R}^3$. We predict the surface color with another feed-forward *3D color network* $\mathcal{F}_\beta(\boldsymbol{z}_c \oplus \boldsymbol{z}_s^k)$, whose input is a color latent code $\boldsymbol{z}_c \in \mathcal{Z}_c$ and the intermediate features from the k-th layer of 3D shape network $\mathcal{F}_\alpha$. The generator of the 3D modality $\mathcal{G}^{\boldsymbol{C}}$ is the combination of the 3D shape and color network:

$$\mathcal{G}^{\boldsymbol{C}}(\boldsymbol{z}_s \oplus \boldsymbol{z}_c \oplus \boldsymbol{p}) = \big\{\mathcal{F}_\alpha(\boldsymbol{z}_s \oplus \boldsymbol{p}), \mathcal{F}_\beta(\boldsymbol{z}_c \oplus \boldsymbol{z}_s^k)\big\}. \tag{6}$$

Both networks are trained using the same set of spatial points. The objective function $\mathcal{L}^{\boldsymbol{C}}$ for $\mathcal{G}^{\boldsymbol{C}}$ is the $\mathcal{L}_1$ loss defined between the prediction and the ground-truth SDF values and surface colors on the sampled points.

2D Sketch. The 2D sketch depicts the 3D structures and provides a natural way for the user to manipulate the 3D shapes. For the purpose of generalization, we adopt a simple and standard fully convolutional network [42] as our sketch generator $\mathcal{G}^{\mathrm{S}}(\boldsymbol{z}_s \oplus \boldsymbol{v})$ with the shape code $\boldsymbol{z}_s \in \mathcal{Z}_s$ and the viewpoint $\boldsymbol{v}$ as input. The objective function $\mathcal{L}^{\boldsymbol{S}}$ is defined as a cross-entropy loss between the reconstructed and ground-truth sketches.

2D Rendering. The 2D color rendering reflects a view-dependent appearance of the 3D surface. Drawing 2D scribbles on the renderings provides an efficient and straightforward interactive tool for the user to edit the 3D surface color. Similar to the 2D sketch modality, we use the standard fully convolutional architecture [42]

as our 2D rendering generator $\mathcal{G}^{\boldsymbol{R}}(\boldsymbol{z}_s \oplus \boldsymbol{z}_c \oplus \boldsymbol{v})$, which takes the concatenation of the shape code $\boldsymbol{z}_s \in \mathcal{Z}_s$, the color code $\boldsymbol{z}_c \in \mathcal{Z}_c$ and the viewpoint $\boldsymbol{v}$. We adopt Laplacian-$\mathcal{L}_1$ loss [3] to train $\mathcal{G}^{\boldsymbol{R}}$:

$$\mathcal{L}^{\boldsymbol{R}}(\boldsymbol{z}_i \oplus \boldsymbol{v}, \boldsymbol{R}_i) = \frac{1}{N} \sum_{j}^{J} 4^{-j} \left\| \mathrm{L}^j(\mathcal{G}^{\boldsymbol{R}}(\boldsymbol{z}_i \oplus \boldsymbol{v})) - \mathrm{L}^j(\boldsymbol{R}_i) \right\|_1, \tag{7}$$

where $\boldsymbol{z}_i$ is the concatenation of the shape and color codes for the target image $\boldsymbol{R}_i$, N is the total number of pixels in the image $\boldsymbol{R}_i$, J is the total number of levels of the Laplacian pyramid (*e.g.*, 3 by default), and $\mathrm{L}^j(x)$ is the j-th level in the pyramid of image x [6]. This loss encourages sharper output [3] compared to the standard $\mathcal{L}_1$ or MSE loss.

Summary. The proposed MM-VAD framework for learning the joint distribution of the three modalities can be learned with the following objective:

$$\begin{aligned} \mathcal{V}(\phi, \theta \mid \boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R}) = &-\mathrm{KL}\big(q_\phi(\boldsymbol{z} \mid \boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R}) \parallel p(\boldsymbol{z})\big) \\ &+ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{C},\boldsymbol{S},\boldsymbol{R})}\big[\log p_\theta(\boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R} \mid \boldsymbol{z})\big], \end{aligned} \tag{8}$$

where the first term regularizes the posterior distribution to a latent prior (*e.g.*, $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$), and the second term can be factorized into three components under the assumption that modalities are independent conditioned on the shared latent variable $\boldsymbol{z}$:

$$\begin{aligned} \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{C},\boldsymbol{S},\boldsymbol{R})}\big[\log p_\theta(\boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R} \mid \boldsymbol{z})\big] &= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{C})}\big[\log p_\theta(\boldsymbol{C} \mid \boldsymbol{z})\big] \\ &+ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{C})}\big[\log p_\theta(\boldsymbol{S} \mid \boldsymbol{z})\big] \\ &+ \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{C})}\big[\log p_\theta(\boldsymbol{R} \mid \boldsymbol{z})\big] \\ &= \mathcal{L}^{\boldsymbol{C}} + \mathcal{L}^{\boldsymbol{S}} + \mathcal{L}^{\boldsymbol{R}}, \end{aligned} \tag{9}$$

where each term corresponds to the reconstruction loss per modality as described above. Notice that the 3D shape modality $\boldsymbol{C}$ contains all the information in the latent variable $\boldsymbol{z}$, therefore $q_\phi(\boldsymbol{z} \mid \boldsymbol{C}, \boldsymbol{S}, \boldsymbol{R}) = q_\phi(\boldsymbol{z} \mid \boldsymbol{C})$.

3.4 Cross-Modal Shape Manipulation with MM-VADs

Given an initial latent code $\boldsymbol{z}_0$ that corresponds to the initial 3D shape $\mathcal{G}^{\boldsymbol{C}}(\boldsymbol{z}_0)$ and any 2D control $\mathcal{G}^{\boldsymbol{M}}(\boldsymbol{z}_0)$ of the 2D modality $\boldsymbol{M} \in \{\boldsymbol{S}, \boldsymbol{R}\}$, the shape manipulation is conducted by optimizing within the latent space to get the updated code $\hat{\boldsymbol{z}}$ such that $\mathcal{G}(\hat{\boldsymbol{z}})^{\boldsymbol{M}}$ matches the 2D edits $\boldsymbol{e}^{\boldsymbol{M}}$:

$$\hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{z}} \ \mathcal{L}_{\text{edit}}(\mathcal{G}^{\boldsymbol{M}}(\boldsymbol{z}), \boldsymbol{e}^{\boldsymbol{M}}) + \mathcal{L}_{\text{reg}}(\boldsymbol{z}), \tag{10}$$

where $\mathcal{L}_{\text{edit}}$ could be any loss (*e.g.*, $\mathcal{L}_1$ loss) that encourages the 2D modalities $\mathcal{G}(\hat{\boldsymbol{z}})^{\boldsymbol{M}}$ to match the 2D edits $\boldsymbol{e}^{\boldsymbol{M}}$, and $\mathcal{L}_{\text{reg}}(\boldsymbol{z})$ encourages the latent code to stay in the latent prior of MM-VADs. We apply the regularization loss proposed in DualSDF [22]:

$$\mathcal{L}_{\text{reg}} = \gamma \max(\|\boldsymbol{z}\|_2^2, \beta), \tag{11}$$

where γ and β controls the strength of the regularization loss. The latent optimization is closely related to the posterior inference (Eqn. 5) of MM-VADs.

MM-VADs allows free-form edits $e^{\boldsymbol{M}}$. For example, the edits $e^{\boldsymbol{M}}$ could be local modifications on the sketch or sparse color scribbles on 2D renderings. This makes the MM-VADs ideally suited for the interactive 3D manipulation tasks. In comparison, the encoder-decoder networks [20] are not robust to the input domain shift (*e.g.*, incomplete data [60]) and require re-training per type of user interactions (*e.g.*, sketch, color scribble).

3.5 Cross-Modal Shape Generation with MM-VADs

Single-View Reconstruction. Given a single input $\boldsymbol{x}^{\boldsymbol{M}}$ of the 2D modality $\boldsymbol{M} \in \{\boldsymbol{C}, \boldsymbol{R}\}$, the task of single-view cross-modal shape generation is to reconstruct the corresponding 3D shape satisfying the 2D constraint. Without the need of training one model per pair of 2D and 3D modalities [20,53] or designing differentiable renderers [34] for each 2D modalities [20], like shape manipulation (§3.4), this task can be tackled via the latent optimization:

$$\hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{z}} \ \mathcal{L}_{\text{recon}}(\mathcal{G}^{\boldsymbol{M}}(\boldsymbol{z}), \boldsymbol{x}^{\boldsymbol{M}}) + \mathcal{L}_{\text{reg}}(\boldsymbol{z}), \tag{12}$$

Partial-View Reconstruction. The MM-VADs are flexible to reconstruct 3D shapes from partially visible inputs. More interestingly, when the input is ambiguous, it provides diverse 3D reconstructions by performing the latent optimization with different initialization of the latent code $\boldsymbol{z}$. This property has practical applications. For example, the MM-VAD could provide multiple 3D shape suggestions interactively while the user is drawing sketches.

Few-Shot Generation. Given a few 2D images spanning a subspace in the 3D distribution that represents a certain semantic attribute (*e.g.*, armchairs, red chairs), the task of few-shot shape generation is to learn a 3D shape generative model that conceptually aligns with the provided 2D images. Given our pre-trained MM-VAD, we tackle this task by steering the latent space with adversarial loss, borrowing the idea from MineGAN [55]. Specifically, we learn a mapping function $h_\omega(\boldsymbol{z})$ that maps the prior distribution of the latent space $\boldsymbol{z} \sim \hat{p}(\boldsymbol{z})$ (*i.e.*, $\mathcal{N}(\mathbf{0}, \mathbf{I})$) to a new distribution such that samples from the 2D generators $\mathcal{G}^{\boldsymbol{M}}(h_\omega(\boldsymbol{z}))$ aligns the target data distribution $\boldsymbol{x} \sim \hat{p}(\boldsymbol{x})$ depicted by the provided 2D images. We apply the WGAN-GP loss [21] with frozen generators to learn the mapping function $h_\omega(\boldsymbol{z})$:

$$\min_{\omega} \max_{\mathcal{D}} \ \mathbb{E}_{\boldsymbol{x} \sim \hat{p}(\boldsymbol{x})}\big[\mathcal{D}(\boldsymbol{x})\big] - \ \mathbb{E}_{\boldsymbol{z} \sim p(\boldsymbol{z})}\big[\mathcal{D}(\mathcal{G}^{\boldsymbol{M}}(h_\omega(\boldsymbol{z})))\big], \tag{13}$$

where both the mapping function h_ω and the discriminator $\mathcal{D}$ are trained from scratch.

4 Experiments

This section provides qualitative and quantitative results of the proposed MM-VADs in versatile tasks of 3D shape manipulation (§ 4.1) and generation (§ 4.2).

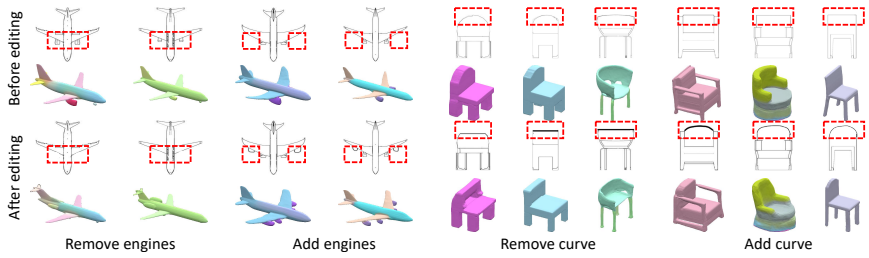

Fig. 3. Editing shape via sketch. The proposed method enables fine-grained editing of shape geometry, *e.g.*, removing the engine of an airplane or reshaping the back of a chair. Interestingly, new engines often appears at the tail of airplane after removing the engines on the wing. This is because airplanes without any engines rarely exist in the domain of our generative model. The edited local regions are highlighted in red bounding boxes.

Dataset. We conduct evaluations and comparisons mainly on 3D ShapeNet dataset [7]. For 3D shapes, We follow DeepSDF [40] to sample 3D points and their signed distances to the object surface. The points that are far from the surface (*i.e.*, with the absolute distance higher than a threshold) are assigned with a pre-defined background color (*e.g.*, white) while points surrounding the surface are assigned with the color of the nearest surface point. For 2D sketches, we use suggestive contours [12] to generate the synthetic sketches. For 2D renderings, we randomize the surface color of 3D shapes per semantic part. We use ShapeNet chairs and airplanes with the same training and test splits as DeepSDF [40].
Implementation Details. We use an 8-layer MLP as the 3D shape network which outputs SDF and a 3-layer MLP as the 3D color network which predicts RGB. We concatenate the features from the 6-th hidden layer of the 3D shape network with the color code as the input to the 3D color network. We train our MM-VADs using Adam [28]. We present more implementation details in the Appendix.
Baselines. We use the following state-of-the-arts as our baselines:

- **Encoder-Decoder Networks** [20]. This model is trained per task of 3D generation from 2D modalities (sketches or RGB images). We do not use the differentiable rendering proposed in [20] which requires auxiliary information (*e.g.*, segmentation mask, depth) and is applicable to MM-VADs.
- **EditNeRF** [36]. This model edits 3D neural radiance field (including shape and color) by updating the neural network weights based on the user's scribbles. We make comparisons with the pre-trained EditNeRF models.

4.1 Cross-modal Shape Manipulation

Sketch-Based Shape Manipulation. The proposed MM-VADs allow users to edit the fine geometric structures via 2D sketches, as described in § 3.4. We provide

Table 2. Editing shape via sketch. We report the Chamfer distance (CD) between the manually edited shapes and our editing results (*lower* is *better*).

	Airplane		Chair	
	− engine	+ engine	− curve	+ curve
Initial shape	0.096	**0.123**	0.066	**0.085**
Edited shape	**0.059**	0.134	**0.054**	0.124

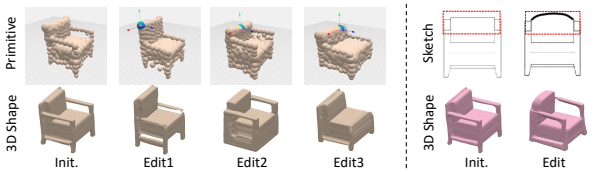

Fig. 4. Comparison with DualSDF. **Left:** DualSDF [22] edits 3D shapes via 3D primitives. Editing different primitives on the same part may lead to dramatically different editing results (2nd - 4th columns). **Right:** our sketch-based interactions is more intuitive for the user.

users with an interactive interface where users can edit the initial sketch by adding or removing a certain part or even deforming a contour line. Fig. 3 presents some qualitative results of sketch-based shape manipulation. Interestingly, we find that our manipulation is semantics-aware. For example, removing the airplane engines on the wings will automatically add new engines to the tail. Such shape priors are absent in non-generative models (*e.g.*, EditNeRF [36]).

It is challenging to quantitatively evaluate the sketch-based shape editing due to the lack of ground-truth paired 3D shapes before and after editing. For this reason, prior works [20] report the quantitative results of 3D reconstruction from sketches as a proxy. We follow prior works and report the same quantitative evaluations in Sec. 4.2. Furthermore, we manually edit the 3D shapes presented in Fig. 3 such that their sketches align with the human edits. Tab. 2 reports the Chamfer distance (CD) between the manually edited shapes and our editing results. We see that CD improves when removing a part, but adding parts unfortunately increases the CD as it induces more changes to the overall shape. This is often desirable, but the CD metric does not reflect that.

Fig. 4 provides a comparison with DualSDF [22]. A fair comparison is not possible, as DualSDF edits shapes via 3D primitives instead of 2D views. We find that DualSDF requires users to select *right* primitives to achieve certain edits (*e.g.*, adding a curve to the chair back). In comparison, our sketch-based shape editing is more intuitive.

Scribble-Based Color Manipulation. MM-VADs allow users to edit the appearance of 3D shapes via color scribbles. Fig. 5 shows that MM-VADs propagate the sparse color scribbles into desired regions (*e.g.*, from the left wing of the airplanes to the right, from the left leg of chairs to the right). We provide more color editing results with diverse color scribbles in the appendix. As a quantitative

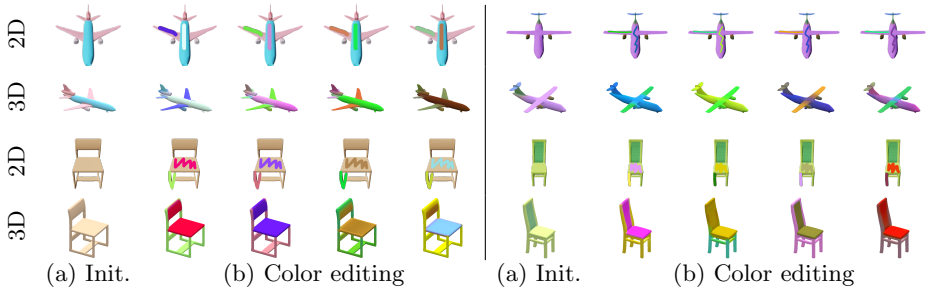

Fig. 5. Editing shape via color scribble. (a) presents the initial 2D and 3D view of the object. **(b)** shows the 2D color scribbles and 3D color editing results.

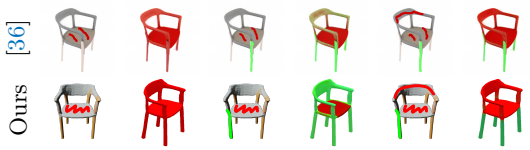

Fig. 6. Comparison with EditNeRF. Our model (**bottom**) achieves comparable editing performance with EditNeRF [36] (**top**). We provide three color edits on 2D views (**odd columns**), each followed by the 3D editing result (**even columns**).

evaluation ,we select 10 shapes per category (including airplanes and chairs) and edit the surface color to make it visually similar to reference shapes with same geometry yet different surface color. The editing quality is measured by the similarity between the renderings of the edited 3D shapes and the reference shapes. Tab. 3 reports the PSNR and LPIPS [61] metrics of the evaluation. The surface color of 3D shapes is much closer to the reference after editing, compared to the initial shapes, suggesting the effectiveness of our MM-VAD model in editing color via scribbles.

A similar task has recently been explored in EditNeRF [36]. However, an apple-to-apple comparison with EditNeRF is not possible due to the intrinsically different 3D representations (NeRF [38] vs SDFs [40]). Moreover, the proposed MM-VADs are generative models while EditNeRF is non-generative; The MM-VADs bridge 2D and 3D via shared latent spaces while EditNeRF relies on differentiable rendering. We present more detailed comparisons in the appendix. We provide qualitative comparisons with EditNeRF on chairs with similar structures using their pre-trained models. Fig. 6 shows that the color editing from MM-VADs is on par with EditNeRF. The MM-VADs achieve the editing via simple latent optimization (Eqn. 12), while EditNeRF requires updating the network weights per instance and fails to generate meaningful color editing results via optimizing the color code alone. Furthermore, MM-VADs take 0.06 seconds per edit and 6.78 seconds to render our 3D shapes into 256×256 RGB images, while EditNeRF takes over a minute per edit including rendering.

Table 3. Quantitative results of editing 3D via 2D scribbles. We edit the surface color of 3D shape based on reference shapes, and report the similarity between the editing results and the target (bottom row). As a reference, we also report the metrics before editing (top row).

Methods	Airplane		Chair	
	PSNR ↑	LPIPS ↓	PSNR ↑	LPIPS ↓
Initial	19.84	0.23	16.20	0.33
Edited	**26.41**	**0.13**	**22.08**	**0.20**

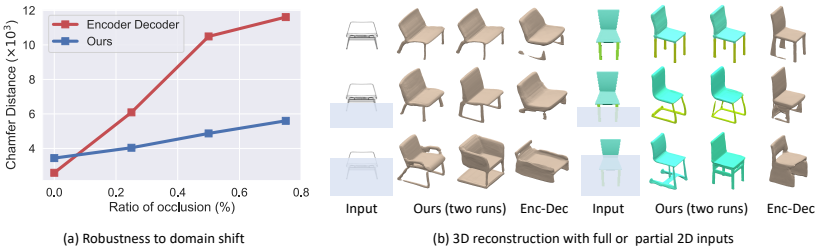

Fig. 7. (a) Robustness to domain shift. We report the Chamfer distance (*lower* is *better*) between 3D reconstructions and the groundtruth under different ratios of image occlusion. **(b) 3D reconstruction with full or partial 2D inputs.** When the full views are available, our model produces consistent 3D reconstruction in different trials. When only partial views are given, our model produces multiple different 3D reconstructions. In comparison, the encoder-decoder networks [20] trained on full-view sketches are not robust to the domain shift induced by occlusion and unable to provide multiple 3D shapes given partial views. Notice that the predictions of surface color is not available in the encoder-decoder networks from the prior work [20].

4.2 Cross-Modal Shape Generation

Single-View and Partial-View Shape Reconstruction. Fig. 7 compares the performance of our model and the encoder-decoder networks [20] under different occlusion ratios in the lower part of the objects in 2D views. The proposed model only has a slight performance drop as the occluded parts increase (Fig. 7a), mainly because of the ambiguity of 3D reconstruction given partial views. In fact, our reconstructions results fit the partial views quite well. Even though our model performs slightly worse than the encoder-decoder networks on full-view inputs, the proposed model is more robust to the input domain shift. This is because compared to task-specific training, our model achieves a better trade-off between reconstruction accuracy and domain generalization. More interestingly, our model can achieve diverse and reasonable 3D reconstruction by sampling different initialization for latent optimization (Fig. 7b).

Few-Shot Shape Generation. The proposed method is able to adapt the pre-trained multi-modal generative model with as few as 10 training samples of a specific 2D modality. Fig. 8 presents some of the few-shot cross-modal shape

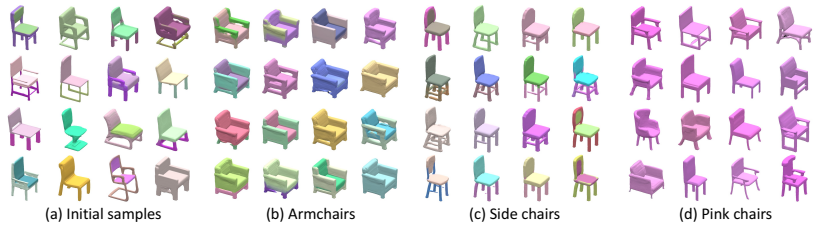

Fig. 8. Few-shot cross-modal shape generation. (a) presents random 3D samples from our model before the adaptation. Given a few 2D exemplars of a certain category (*e.g.*, armchair), our model can be adapted to generate corresponding 3D shapes (b-d).

Table 4. Quantitative results of few-shot cross-modal shape generation. We report Frechet Inception Distance (FID) (*lower* is *better*) and classification error (Cls. Err) (*lower* is *better*). We effectively adapt the pretrained multi-modal VAD model using a few 2D images to a desired 3D shape generator. As a reference, we report the metrics before the few-shot adaptation (top row).

Stage	Metrics	Arm	Side	Red	Avg.
Init.	FID ↓	138.1	95.2	93.7	109.0
	Cls.Err. ↓	0.79	0.64	0.82	0.75
Adapt.	FID ↓	**130.4**	**92.4**	**93.0**	**105.3**
	Cls.Err. ↓	**0.01**	**0.10**	**0.00**	**0.04**

generation results. To quantitatively evaluate the few-shot shape generation performance, we render the 3D shapes into 2D RGB images and report the Frechet Inception Distance (FID) scores [24] between the rendered images and the ground-truth samples. Since the FID score is not sensitive to the semantic difference between two image sets, we also report the classification error on the random samples from the model before and after the adaptation. Specifically, we train a binary image classifier to identify the target image categories (*e.g.*, armchairs vs. other chairs), and we run the trained classifier on the 2D renderings of the 3D samples before and after the adaptation. As presented in Tab. 4, our pre-trained generative model can be effectively adapted to a certain shape subspace given as few as 10 2D examples. This capability allows us to agilely adapt our generative model to a subspace defined by a few unlabelled samples, so that users can easily narrow down the target shape during the manipulation by providing a few samples of a common attribute, such as a specific category, style, or color. We are unaware of any prior works that can tackle this task in the literature. The 2D examples used to adapt the pre-trained generative model are provided in our appendix.

Shape and Color Transfer. Transferring shape and color across different 3D instances can be achieved by simply swapping the latent codes. Fig. 9 shows that the shape and color are well disentangled in the proposed generative model. The

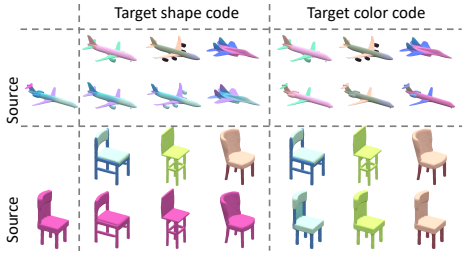

Fig. 9. Shape and color transfer. The reference 3D shapes (top row) provide the shape codes or color codes for each source instances (first column).

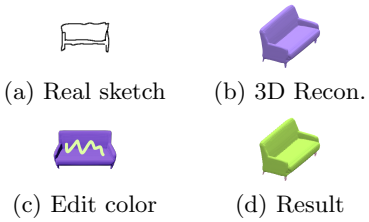

Fig. 10. Our model enables consecutive 3D reconstruction and manipulation given a hand-drawn sketch.

transfer results also are semantically meaningful, *i.e.*, the color is only transferred across the same semantic parts (*e.g.*, seats for the chair, wings for the airplane) even though the geometry of the source and target instances are quite different.

4.3 Case Study on Real Images

The workflow of 3D designers usually starts by drawing a 2D sketch to portray the coarse 3D geometry and then colorizes the sketch to depict the 3D appearance. These 2D arts are used as a reference to build 3D objects. Undoubtedly this procedure requires extensive human efforts and expertise. Such tasks can be automated with our MM-VADs. As shown in Fig. 10, we first reconstruct the 3D shape from a hand-drawn sketch. We then assign a surface color by randomly sampling a color code from the latent space of the MM-VADs, which can be easily edited by drawing color scribbles on the surface. Our model does not require any re-training on each of these steps and provides a tool to conduct shape generation and color editing consecutively. Such a task is infeasible with the existing works that train an encoder-decoder network to predict 3D shape from sketch [20].

5 Discussion

We propose a multi-modal generative model which bridges multiple 2D and 3D modalities through a shared latent space. One limitation of the proposed method is that we are only able to provide editing results in the prior distribution of our generative model (see appendix for more details). Despite this limitation, our model has enabled versatile cross-modal 3D generation and manipulation tasks without the need of re-training per task and demonstrates strong robustness to input domain shift.

Acknowledgements. Subhransu Maji acknowledges support from NSF grants #1749833 and #1908669. Our experiments were partially performed on the University of Massachusetts GPU cluster funded by the Mass. Technology Collaborative.

Appendix

A Implementation Details

The implementation details of 3D shape and color networks are included in the main text. Here we provide additional implementation details.

- **Joint latent space.** The shape and color latent codes are both of dimension 128 throughout our experiments. We observe that lower-dimensional latent codes (*e.g.*, 32) lead to worse shape reconstruction.
- **2D sketch and renderings.** The image resolution of all 2D modalities is set to 128×128. We use the generator architecture from DCGAN [42] for all 2D modalities.
- **Few-shot shape generation.** We use the discriminator from DCGAN [42]. The mapping function $h_w(\boldsymbol{z})$ in the MineGAN framework [55] is a two-layer MLP with batch normalization [25] and a ReLU activation function.
- **Latent optimization.** In the task of shape and appearance manipulation, we conduct the latent optimization for 5 steps starting from a known initial latent code that corresponds to the initial 2D and 3D instances. The hyperparameter γ and β in Eqn.11 is 0.02 and 0.5 respectively by default. For the single-view shape generation tasks, we run multiple trials of latent optimization from different randomly sampled latent codes. The optimized code with minimal reconstruction loss is used as the final result. We observe that such multi-trial optimization significantly stabilizes the performance of 3D reconstruction (see Sec. B for more details).

B Ablation study

The latent optimization is crucial to the performance of our shape reconstruction and manipulation tasks. In this section, we provide ablation studies on the regularization loss (Eqn. 10 in the main text) and the multi-trial latent optimization method (as described in Sec. A).

Regularization Loss. We apply the same regularization loss as DualSDF [22], *i.e.*, $\mathcal{L}_{\text{reg}} = \gamma \max(\|\boldsymbol{z}\|_2^2, \beta)$, where two hyperparameters γ and β control the strength of the regularization. The regularization term $L_{\text{REG}}(\boldsymbol{z})$ effectively constrains the optimization of the latent code $\boldsymbol{z}$ in the prior distribution of the pretrained MM-VADs. Without such regularization, we find that the single-view 3D reconstruction fails in most cases. Fig. 11 provides one example.

Multi-trial latent optimization for 3D reconstruction. Similar to other generative models (*e.g.*, GANs), the latent optimization with the proposed MM-VADs is a highly non-convex problem and prone to local minimal. To relieve this issue, we conduct the latent optimization for multiple rounds with different initial latent codes. We use the latent codes with minimal reconstruction loss in

(a) Input (b) with $\mathcal{L}_{\text{REG}}$ (e) w/o $\mathcal{L}_{\text{REG}}$

Fig. 11. The effect of $\mathcal{L}_{\text{REG}}$. Without the regularization term, our model fails to reconstruct 3D shapes from a sketch image.

multiple trials as the final results of the latent optimization. We find this simple strategy significantly stabilizes our model in the 3D reconstruction task. For example, the mean Chamfer distance decreases from 5.50 to 1.73 in the task of 3D reconstruction from single-view sketch on ShapeNet airplanes and from 9.10 to 4.70 on ShapeNet chairs. In 3D shape manipulation, the latent optimization starts from a known latent code corresponding to the target shape to be edited, and we only run the latent optimization once.

C Baselines

Here we present more details about the baselines used in our experiments.

- **Encoder-Decoder Networks** [20,43]. This model is originally designed for predicting 3D shapes from sketches, followed by a shape refinement step based on differentiable rendering. We re-purpose this model to reconstruct 3D shapes from RGB images by simply modifying the input channels in the first convolutional layer. We use the official implementations with default hyperparameter settings [3].
- **EditNeRF** [36] edits a conditional radiance field representation of 3D scenes with sparse scribbles as input. The shape and color of 3D objects are edited by updating the neural network weights. We make qualitative comparisons with the EditNeRF using their pre-trained models[4]. Our model shares many similarities with EditNeRF (*e.g.*, network architecture, scribble-based interaction). However, the proposed model is significantly different from EditNeRF in terms of shape representation (SDFs [40] vs NeRF [38]), shape manipulation method (latent optimization vs network fine-tuning), and the way to bridge the 3D and 2D modalities (shared latent spaces vs differentiable rendering). Tab. 5 provides detailed comparisons between EditNeRF and our model.

[3] https://github.com/cvlab-epfl/MeshSDF
[4] https://github.com/stevliu/editnerf

Table 5. Comparisons with EditNeRF [36]. † The shape reconstruction and manipulation can be combined and interleaved with the proposed model. This enables us to edit novel instances (Fig. 10 in the main manuscript provides an example). ‡ The time cost of rendering a 256×256 image is included in the editing time

	EditNeRF [36]	Ours
Latent codes	Separate shape and color codes	
Network	A common network shared by all training instances	
Task	Shape/color manipulation with sparse scribbles	
Instance-specific sub-networks	✓	✗
Generative model	✗	✓
3D recon. from sketch or RGB	✗	✓
Editing novel instances †	✗	✓
Shape representation	NeRF [38]	SDFs [40]
Bridge of 2D/3D modalities	Differentiable rendering	Shared latent spaces
Editing method	Update network weights	Latent optimization
Estimated editing time ‡	60s	7s

D More Experimental details

D.1 Training and Testing Dataset

We train the proposed multi-modal variational auto-decoders (MM-VADs) on the ShapeNet dataset [7]. The training and testing split is the same as DeepSDF [40] and DualSDF [22]. We use the same pre-trained MM-VADs throughout our experiments. For airplanes, there are 1780 shapes for training and 456 shapes for testing. For chairs, there are 3281 training shapes and 833 testing instances. For 3D shape manipulation, we present the results on known shapes (*i.e.*, shapes from training data), similar to EditNeRF [36] and DualSDF [22].

D.2 3D reconstruction from Sketch or RGB modalities.

Table 6 presents quantitative evaluations of the 3D reconstruction from sketch and RGB inputs under different occlusion ratios, corresponding to the curves in Fig. 7 in the main manuscript. We report results on both vertically and horizontally occluded inputs. Since 3D shapes and their 2D views are generally symmetric horizontally, the proposed model has almost no performance drop when masking out the right-half regions of the inputs. In comparison, the encoder-decoder networks [20] that is trained on full-view inputs suffers from the input domain shift induced by the occlusion.

D.3 Few-shot 3D Generation

Fig. 12 presents the 2D examples used in our few-shot shape generation experiments. For each category (*e.g.*, armchair), we randomly sample 10 images from our training data. We then adapt a pre-trained MM-VAD using these 2D

Table 6. Quantitative results of single-view reconstruction. We report the average Chamfer Distance (30,000 points) multiplied by 10^3 between the reconstructed 3D shapes and the groundtruth (*lower* is *better*). The performance of the proposed model is slightly worse than the encoder-decoder networks [20] trained on the full-view inputs. However, MM-VADs perform more robustly to the input domain shift (*e.g.*, only partial view of input is available). The first column presents the occlusion rate in the input, where "Full" means no occlusion in the input, "1/2-horizontal" the left half of the input is visible, and "3/4-vertical" the top 3/4 region of the object is available. Superscripts in the last row denote the performance drop under the input domain shift (*lower* is *better*). This table corresponds to Fig. 7 in the main text

View	Model	Airplane		Chair		Avg.
		Sketch	RGB	Sketch	RGB	
Full	Enc-Dec	**1.45**	**1.21**	**4.24**	**3.45**	**2.59**
	Ours	1.73	1.40	5.96	4.70	3.44
1/2-horizontal	Enc-Dec	3.30	6.18	16.34	7.61	$8.36^{+5.77}$
	Ours	**1.79**	**1.38**	**6.07**	**5.00**	$\mathbf{3.56}^{+0.12}$
3/4-vertical	Enc-Dec	2.33	1.94	13.10	6.99	$6.09^{+3.50}$
	Ours	**2.07**	**1.55**	**6.91**	**5.64**	$\mathbf{4.04}^{+0.60}$
1/2-vertical	Enc-Dec	3.97	3.56	24.31	10.13	$10.49^{+7.90}$
	Ours	**2.39**	**1.89**	**8.01**	**7.06**	$\mathbf{4.87}^{+1.43}$
1/4-vertical	Enc-Dec	4.28	4.77	27.64	9.77	$11.61^{+9.02}$
	Ours	**3.32**	**2.63**	**8.27**	**8.19**	$\mathbf{5.60}^{+2.16}$

examples based on the MineGAN framework [55]. We further collect 200 images per category from our training data for training binary classifiers and calculating FID scores. The classifiers are fine-tuned from a ResNet18 [23] pre-trained on ImageNet.

E Limitations

3D reconstruction from 2D modalities. The proposed model fails to reconstruct *fine structures* of 3D shapes from sketches or RGB views, for example, the holes on the back of chairs (Fig. 13a, b, g, h), fine textures on the seat of chairs (Fig. 13e), or the wheelbase of desk chairs (Fig. 13c, f). The capability of modeling fine structures is mainly determined by the 3D shape representation (*i.e.*, SDFs [40]), training samples of SDFs, and the capacity of the proposed generative model. This issue can be potentially relieved by sampling more 3D training points surrounding the surface or increasing the capacity of the proposed model (*e.g.*, enlarging the dimension of the latent space, increasing the depth of 3D shape networks)

3D manipulation with 2D color scribble. Similar to GAN-based image manipulation models [65,19,5,39], we are only able to provide editing results within the prior distribution of a pre-trained MM-VAD. For example, in the task of

Fig. 12. 2D examples for few-shot shape generation. Each row presents the 10 2D examples used to adapt a pre-trained MM-VADs to generate armchairs, side chairs, and pink chairs respectively.

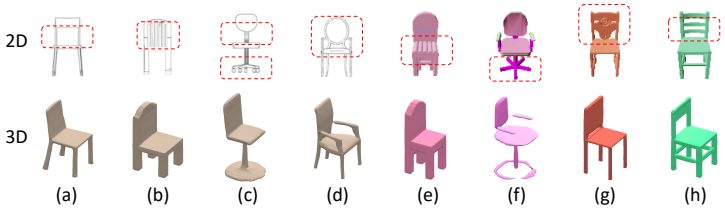

Fig. 13. Limitations of 3D reconstruction from 2D modalities. The proposed model fails to generate *fine structures* of 3D shapes from sketches (a-d) or RGB renderings (e-h). The red bounding boxes highlight the object parts where our model fails to reconstruct the 3D structures.

editing shape with color scribbles, if there are multiple scribbles of different colors on the same part of a shape (*e.g.*, the seat of a chair), our model either edits the shape based on one of the scribbles or generates a surface color that is completely different from all scribbles, as shown in Fig. 14. We notice that the editing results of EditNeRF [36] are similar to ours based on their released demo[5]. Our model may produce unexpected color editing results, for example, the edited 3D surface color may not match the 2D color scribbles provided by the user (Fig. 14d), probably due to bad initialization of the latent code or suboptimal hyperparameter settings. The multi-trial latent optimization described in Sec. B may relieve this issue.

3D manipulation via 2D sketch. In this task, the major issue is that editing one part of a shape usually leads to changes in other parts. For example, removing the engines on the wing of airplanes results in new engines on the tail in many cases, as shown in Fig. 3 in the main text. Fig. 15 in this section provides more examples. This is mainly because editing shapes via latent optimization can only produce new shapes in the prior distribution of the generative model. It is potentially useful to add more constraints upon the latent optimization, *e.g.*,

[5] https://github.com/stevliu/editnerf

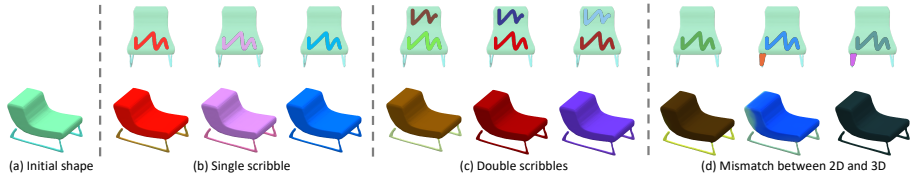

Fig. 14. Limitations of editing shape via color scribble. We are only able to provide color editing results in the prior distribution of the generative model. For example, if there are two scribbles of different color on the same part of a chair, our model either propagates one of the scribbles (*e.g.*, first two columns in (c)) or generates a surface color that are different from both scribbles (*e.g.*, last column in (c)). As a reference, we provide the editing results with single scribble in (b). Our model also produces 3D color editing results that do not match with the 2D input scribbles, as shown in (d).

enforcing the output of the 2D sketch generator to be as similar as possible to the original sketch. However, our preliminary experiments show that the latent optimization with such constraint typically under-fits the edited parts of the sketch and fails to achieve desired edits in 3D shape. In addition, the proposed model fails to add more complicated structures into the shape, for example, adding holes onto the back of chairs (Fig. 15c). We will investigate these issues further in our future work.

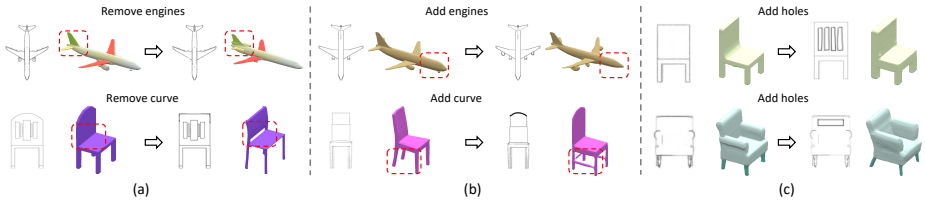

Fig. 15. Limitations of editing shape via sketch. **(a-b)** Editing one part via sketch leads to changes in other parts which are not edited in the sketch. The parts where our model fails to maintain are annotated in red bounding boxes. **(c)** The proposed method fails to add fine structures onto the shape via sketch (*e.g.*, adding holds onto the back of chairs).

Few-shot shape generation. We are unable to adapt a pre-trained MM-VAD to generate shapes of *fine-grained categories* (*e.g.*, single-engine airplanes) using a few 2D RGB images. We also fail to adapt a pre-trained MM-VAD using a few *2D sketches*. We hypothesize that this is because the discriminator is trained from scratch and unable to learn discriminative representations among fine-grained categories or sparse inputs (*e.g.*, sketches) with limited 2D examples. These issues

may be relieved by initializing the discriminator with a pre-trained classifier. We leave this in our future work.

F Diverse color scribbles.

Fig. 16 shows more 3D color editing results with diverse color scribbles. Our method is robust to color scribbles of different shapes/amounts/positions.

Fig. 16. Diverse color scribbles. The first column presents the initial 2D and 3D modalities. The following columns present the color editing results with diverse scribbles.

References

1. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4432–4441 (2019) 3
2. An, X., Tong, X., Denning, J.D., Pellacini, F.: Appwarp: Retargeting measured materials by appearance-space warping. In: Proceedings of the 2011 SIGGRAPH Asia Conference. pp. 1–10 (2011) 1, 3
3. Athar, S., Burnaev, E., Lempitsky, V.: Latent convolutional models. In: ICLR (2018) 7
4. Bau, D., Liu, S., Wang, T., Zhu, J.Y., Torralba, A.: Rewriting a deep generative model. In: eccv. pp. 351–369. Springer (2020) 3
5. Bau, D., Strobelt, H., Peebles, W., Zhou, B., Zhu, J.Y., Torralba, A., et al.: Semantic photo manipulation with a generative image prior. arXiv preprint arXiv:2005.07727 (2020) 3, 18
6. Burt, P.J., Adelson, E.H.: The laplacian pyramid as a compact image code. In: Readings in computer vision, pp. 671–679. Elsevier (1987) 7
7. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 9, 17
8. Chen, K., Choy, C.B., Savva, M., Chang, A.X., Funkhouser, T., Savarese, S.: Text2shape: Generating shapes from natural language by learning joint embeddings. In: ACCV. pp. 100–116. Springer (2018) 3
9. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019) 1, 3

10. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 8789–8797 (2018) 3
11. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: European conference on computer vision. pp. 628–644. Springer (2016) 3
12. DeCarlo, D., Finkelstein, A., Rusinkiewicz, S., Santella, A.: Suggestive contours for conveying shape. In: ACM SIGGRAPH 2003 Papers, pp. 848–855 (2003) 9
13. Delanoy, J., Aubry, M., Isola, P., Efros, A.A., Bousseau, A.: 3d sketching using multi-view deep volumetric prediction. Proceedings of the ACM on Computer Graphics and Interactive Techniques **1**(1), 1–22 (2018) 1, 3
14. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017) 3
15. Gkioxari, G., Malik, J., Johnson, J.: Mesh r-cnn. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9785–9795 (2019) 3
16. Goel, S., Kanazawa, A., Malik, J.: Shape and viewpoint without keypoints. In: European Conference on Computer Vision. pp. 88–104. Springer (2020) 3
17. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014) 3
18. Grady, L.: Random walks for image segmentation. IEEE transactions on pattern analysis and machine intelligence **28**(11), 1768–1783 (2006) 3
19. Gu, J., Shen, Y., Zhou, B.: Image processing using multi-code gan prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3012–3021 (2020) 3, 18
20. Guillard, B., Remelli, E., Yvernay, P., Fua, P.: Sketch2mesh: Reconstructing and editing 3d shapes from sketches. In: ICCV (2021) 1, 3, 4, 8, 9, 10, 12, 14, 16, 17, 18
21. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.: Improved training of wasserstein gans. In: NeurIPS (2017) 8
22. Hao, Z., Averbuch-Elor, H., Snavely, N., Belongie, S.: Dualsdf: Semantic shape manipulation using a two-level representation. In: CVPR. pp. 7631–7641 (2020) 3, 4, 7, 10, 15, 17
23. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 18
24. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017) 13
25. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. PMLR (2015) 15
26. Jin, A., Fu, Q., Deng, Z.: Contour-based 3d modeling through joint embedding of shapes and contours. In: Symposium on Interactive 3D Graphics and Games. pp. 1–10 (2020) 3
27. Kanazawa, A., Tulsiani, S., Efros, A.A., Malik, J.: Learning category-specific mesh reconstruction from image collections. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 371–386 (2018) 3
28. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 9

29. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013) 3, 4
30. Lempitsky, V., Kohli, P., Rother, C., Sharp, T.: Image segmentation with a bounding box prior. In: 2009 IEEE 12th international conference on computer vision. pp. 277–284. IEEE (2009) 3
31. Levin, A., Lischinski, D., Weiss, Y.: Colorization using optimization. In: ACM SIGGRAPH 2004 Papers, pp. 689–694 (2004) 3
32. Li, Y., Sun, J., Tang, C.K., Shum, H.Y.: Lazy snapping. ACM Transactions on Graphics (ToG) **23**(3), 303–308 (2004) 3
33. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. Advances in neural information processing systems **29**, 469–477 (2016) 3
34. Liu, S., Zhang, Y., Peng, S., Shi, B., Pollefeys, M., Cui, Z.: Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In: CVPR. pp. 2019–2028 (2020) 8
35. Liu, S., Li, T., Chen, W., Li, H.: Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7708–7717 (2019) 3
36. Liu, S., Zhang, X., Zhang, Z., Zhang, R., Zhu, J.Y., Russell, B.: Editing conditional radiance fields. In: ICCV (2021) 1, 3, 4, 9, 10, 11, 16, 17, 19
37. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019) 1, 3, 6
38. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision. pp. 405–421. Springer (2020) 3, 4, 11, 16, 17
39. Pan, X., Zhan, X., Dai, B., Lin, D., Loy, C.C., Luo, P.: Exploiting deep generative prior for versatile image restoration and manipulation. In: European Conference on Computer Vision. pp. 262–277. Springer (2020) 3, 18
40. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: CVPR. pp. 165–174 (2019) 1, 3, 6, 9, 11, 16, 17, 18
41. Pellacini, F., Battaglia, F., Morley, R.K., Finkelstein, A.: Lighting with paint. ACM Transactions on Graphics (TOG) **26**(2), 9–es (2007) 1, 3
42. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015) 6, 15
43. Remelli, E., Lukoianov, A., Richter, S.R., Guillard, B., Bagautdinov, T., Baque, P., Fua, P.: Meshsdf: Differentiable iso-surface extraction. arXiv preprint arXiv:2006.03997 (2020) 16
44. Rother, C., Kolmogorov, V., Blake, A.: " grabcut" interactive foreground extraction using iterated graph cuts. ACM transactions on graphics (TOG) **23**(3), 309–314 (2004) 3
45. Saharia, C., Chan, W., Chang, H., Lee, C.A., Ho, J., Salimans, T., Fleet, D.J., Norouzi, M.: Palette: Image-to-image diffusion models. arXiv preprint arXiv:2111.05826 (2021) 3
46. Schmidt, T.W., Pellacini, F., Nowrouzezahrai, D., Jarosz, W., Dachsbacher, C.: State of the art in artistic editing of appearance, lighting and material. In: Computer Graphics Forum. vol. 35, pp. 216–233. Wiley Online Library (2016) 1, 3

47. Shen, Y., Gu, J., Tang, X., Zhou, B.: Interpreting the latent space of gans for semantic face editing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9243–9252 (2020) 3
48. Shen, Y., Yang, C., Tang, X., Zhou, B.: Interfacegan: Interpreting the disentangled face representation learned by gans. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020) 3
49. Shi, Y., Siddharth, N., Paige, B., Torr, P.H.: Variational mixture-of-experts autoencoders for multi-modal deep generative models. arXiv preprint arXiv:1911.03393 (2019) 3
50. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. arXiv preprint arXiv:1906.01618 (2019) 1, 3, 6
51. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. In: ICML. pp. 2256–2265. PMLR (2015) 3
52. Suzuki, M., Nakayama, K., Matsuo, Y.: Joint multimodal learning with deep generative models. arXiv preprint arXiv:1611.01891 (2016) 3
53. Tatarchenko, M., Richter, S.R., Ranftl, R., Li, Z., Koltun, V., Brox, T.: What do single-view 3d reconstruction networks learn? In: CVPR. pp. 3405–3414 (2019) 8
54. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: ECCV. pp. 52–67 (2018) 3
55. Wang, Y., Gonzalez-Garcia, A., Berga, D., Herranz, L., Khan, F.S., Weijer, J.v.d.: Minegan: effective knowledge transfer from gans to target domains with few images. In: CVPR. pp. 9332–9341 (2020) 8, 15, 18
56. Wu, M., Goodman, N.: Multimodal generative models for scalable weakly-supervised learning. Advances in Neural Information Processing Systems **31** (2018) 3
57. Wu, M., Goodman, N.: Multimodal generative models for compositional representation learning. arXiv preprint arXiv:1912.05075 (2019) 3
58. Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U.: Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. arXiv preprint arXiv:1905.10711 (2019) 3
59. Yang, G., Sun, D., Jampani, V., Vlasic, D., Cole, F., Chang, H., Ramanan, D., Freeman, W.T., Liu, C.: Lasr: Learning articulated shape reconstruction from a monocular video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15980–15989 (2021) 3
60. Zadeh, A., Lim, Y.C., Liang, P.P., Morency, L.P.: Variational auto-decoder: A method for neural generative modeling from incomplete data. arXiv preprint arXiv:1903.00840 (2019) 3, 4, 5, 8
61. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) 11
62. Zhang, R., Zhu, J.Y., Isola, P., Geng, X., Lin, A.S., Yu, T., Efros, A.A.: Real-time user-guided image colorization with learned deep priors. ACM Transactions on Graphics (TOG) **9**(4) (2017) 3
63. Zhang, S.H., Guo, Y.C., Gu, Q.W.: Sketch2model: View-aware 3d modeling from single free-hand sketches. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6012–6021 (2021) 3
64. Zhong, Y., Gryaditskaya, Y., Zhang, H., Song, Y.Z.: Deep sketch-based modeling: Tips and tricks. In: 2020 International Conference on 3D Vision (3DV). pp. 543–552. IEEE (2020) 3
65. Zhu, J., Shen, Y., Zhao, D., Zhou, B.: In-domain gan inversion for real image editing. In: ECCV. pp. 592–608. Springer (2020) 3, 18