Explorable Mesh Deformation Subspaces from Unstructured 3D Generative Models

Arman Maesumi
arman_maesumi@brown.edu
Brown University
Providence, USA

Paul Guerrero
Adobe Research
London, UK

Vladimir G. Kim
Adobe Research
Seattle, USA

Matthew Fisher
Adobe Research
San Francisco, USA

Siddhartha Chaudhuri
Adobe Research
New York, USA

Noam Aigerman
University of Montreal
Adobe Research
Montreal, Canada

Daniel Ritchie
Brown University
Providence, USA

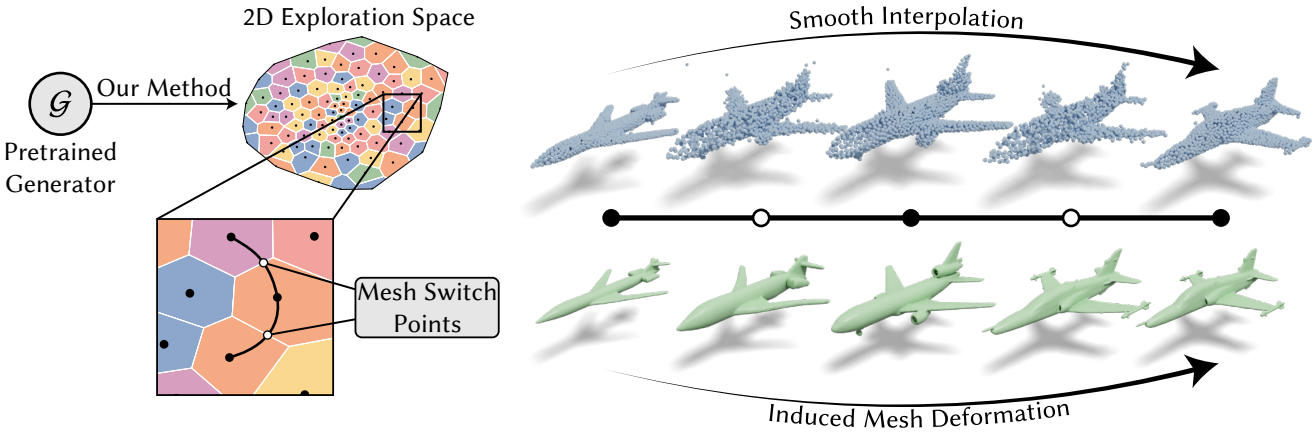


Figure 1: We present a method to explore variations among a given set of input shapes (denoted by black Voronoi centers on the left) using a two-dimensional exploration space. This exploration space smoothly and naturally interpolates between the input shapes by constructing a mapping to a sub-space of a pre-trained generator's latent space that optimizes the smoothness of interpolations along any trajectory. Additionally, we transfer the variation over these interpolation trajectories onto the original high-quality meshes, avoiding loss of detail from the unstructured generator output.

ABSTRACT

Exploring variations of 3D shapes is a time-consuming process in traditional 3D modeling tools. Deep generative models of 3D shapes often feature continuous latent spaces that can, in principle, be used to explore potential variations starting from a set of input shapes; in practice, doing so can be problematic—latent spaces are high dimensional and hard to visualize, contain shapes that are not relevant to the input shapes, and linear paths through them often lead to sub-optimal shape transitions. Furthermore, one would ideally be able to explore variations in the original high-quality meshes used to train the generative model, not its lower-quality output geometry. In this paper, we present a method to explore variations among a given set of *landmark* shapes by constructing a mapping from an easily-navigable 2D exploration space to a subspace of a pre-trained generative model. We first describe how to find a mapping that spans the set of input landmark shapes and exhibits smooth variations between them. We then show how to turn the variations in this subspace into deformation fields, to transfer those variations to high-quality meshes for the landmark shapes. Our results show that our method can produce visually-pleasing and easily-navigable 2D exploration spaces for several different shape categories, especially as compared to prior work on learning deformation spaces for 3D shapes.
https://github.com/ArmanMaesumi/generative-mesh-subspaces

CCS CONCEPTS

• **Computing methodologies** → **Shape analysis**.

KEYWORDS

shape deformation, generative model, 3D shape generation

1 INTRODUCTION

When designing a 3D shape, artists, designers, or engineers may want to explore variations of initial designs, to find shapes with better aesthetics or functional properties. In traditional 3D modelling tools, this exploration is time-consuming, as each variation needs to created manually. Data-driven generative models have introduced revolutionary new capabilities to the practice of 3D shape design and exploration. This trend began over a decade ago with methods for generating 3D shapes by recombining parts from existing shapes [Chaudhuri et al. 2013, 2011; Kalogerakis et al. 2012]. With the emergence of deep learning, the movement has shifted to deep generative models [Achlioptas et al. 2017; Chen and Zhang 2019a; Li et al. 2021; Wu et al. 2016; Yang et al. 2019; Zheng et al. 2022]. These latter methods are particularly transformative: in addition to synthesizing new shapes, these models often feature *latent spaces* that allow navigation through a continuous manifold of shapes.

In principle, these latent spaces are useful to explore variations of initial designs. However, latent spaces have some limitations when used for this purpose: First, latent spaces contain the full distribution of shapes that the generative model was trained on, but we want to focus on variations among a small set of initial shapes of interest. Second, while individual points in the latent space typically correspond to plausible shapes, following a linear path between two such points typically does not produce the most natural transition between the shapes they represent—often, extraneous shape variations and/or noise occur along the way. Third, these latent spaces are typically high dimensional (e.g. $\mathbb{R}^{128}$ or higher), making them hard to visualize and interact with. Such spaces may be acceptable for goal-directed interpolation between two shapes, but they remain difficult to use for open-ended, interactive exploration. Finally, the best state-of-the-art generative models output point clouds or implicit fields [Li et al. 2021; Zheng et al. 2022] to handle the challenge of modeling shape distributions with varying topologies, yet many downstream graphics applications demand mesh representations. While point clouds and implicit fields can be meshed, the resulting meshes are not as high-quality as those designed by expert artists.

In this paper, we propose a new method for exploring continuous variations among a set of given *landmark* meshes, along with discrete transitions between them. Rather than trying to explore the full latent space of a pre-trained generative model, or finding an embedding that preserves latent space distances, we extract a smaller shape space that smoothly spans the landmark. This reduced space is used as a prior for downstream tasks (i.e. shape deformation).

Our method consists of two main components. First, we develop a technique to construct a mapping from an easily-navigable *exploration space* to a subspace of a pre-trained shape generative model. Given a set of landmark shapes (which may be from the generative model's training set, or another set of shapes of the same category), we embed these shapes in two dimensions to facilitate interaction. We then learn a map from this 2D exploration space to the generative model's latent space, such that (a) the embedded points map to latent space points that produce their corresponding landmark shapes when put through the pre-trained generator, and importantly (b) navigating between landmark points produces smooth shape variations, rather than preserving latent space distances.

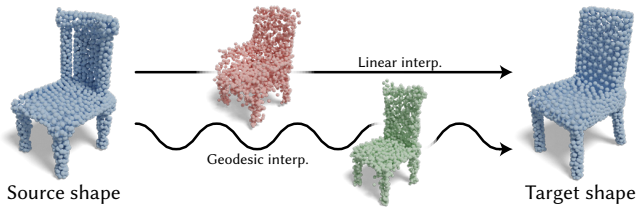


Figure 2: Linearly interpolating between shapes in latent space may produce poor intermediate samples. Here we see the difference between linear and non-linear (geodesic) interpolation at $t = 0.5$ in SP-GAN's latent space. The linear interpolation is noisy and is beginning to grow armrests, despite the source and target shapes lacking such features.

Second, we show how to use this exploration space to explore variations within and between the high-quality original *meshes* for the landmark shapes. Specifically, we develop a technique for computing continuous deformation fields from small steps within the exploration space, allowing its rich semantic variations to be transferred to meshes at real-time interactive rates. We also consider how and when to switch between deformed landmark meshes as the user navigates the 2D space to minimize jarring visual discontinuities.

We evaluate our method by producing easily-navigable and visually-pleasing exploration spaces for several different shape categories. We compare the paths through latent space that our method produces to those produced by alternative approaches, including a method for directly learning a deformation space from a large collection of shapes [Jiang et al. 2020]. We also present a set of ablations and analyses of our method's components.

In summary, our contributions are:

- A technique for building a 2D exploration space that naturally interpolates between a set of landmark shapes while staying on the shape manifold induced by a pre-trained generative model.
- A method for transferring latent space variations into continuous deformations of high-quality landmark meshes and discrete transitions between them in real-time (accompanied by a graphical user interface).

2 RELATED WORK

Deep generative models of 3D shapes There has been an explosion of work in recent years on applying deep generative models to the problem of synthesizing 3D shapes. In this paper, we are concerned with *latent variable deep generative models*, i.e. generative models that map points in a latent space $\mathcal{Z}$ to the output shape domain via some map f. A variety of such models exist, including variational autoencoders (VAEs) [Jones et al. 2020; Li et al. 2017], generative adversarial networks (GANs) [Achlioptas et al. 2017; Chen and Zhang 2019b; Wu et al. 2016; Zheng et al. 2022], normalizing flows [Yang et al. 2019], and denoising diffusion probabilistic models (DDPMs) [Hui et al. 2022b; Zeng et al. 2022]. Our method is designed to work well with such generative models in which f is a single forward pass through a neural network (e.g. VAEs, GANs). We discuss challenges with, and potential ideas for, applying it to multi-step f models (e.g. flows, DDPMs) in Section 8. Our method is

not designed to work with autoregressive models, which generate outputs through a sequence of sampling steps [Mittal et al. 2022].

There have also been attempts at learning generative models that output meshes directly. Thus far, these have suffered from various limitations: restriction to genus zero topology [Wang et al. 2018], requiring part-segmented data [Gao et al. 2019; Yang et al. 2020], or unpredictable output quality [Nash et al. 2020]. Rather than *generate* meshes directly, our approach allows exploration of variations between existing high-quality meshes by borrowing the shape variations exhibited by a pre-trained non-mesh generator.

Exploring collections of 3D shapes Several prior techniques have been proposed for exploring collections of 3D shapes by deforming a template [Ovsjanikov et al. 2011], highlighting regions on some reference shapes [Kim et al. 2012], or varying manually annotated language-based attributes [Chaudhuri et al. 2013]. These interfaces typically focus on modifying a single reference shape, and thus are more suitable for local search, exploring immediate neighbors of the reference model. A global analysis has been proposed to identify main modes of variations of shapes in a collection [Kim et al. 2013], and 2D layouts are commonly used to jointly explore shape collections or generative shape spaces [Averkiou et al. 2014; Rowaida 2021], as well as other visual domains in general [Kleiman et al. 2015; Marks et al. 1997; Talton et al. 2009]. These type of 2D layouts give a holistic view of the entire space, while still providing intuitive interfaces in 2D. Various techniques have been developed in mapping high-dimensional similarity data to 2D (e.g., PCA, MDS, [Fried et al. 2015], [Dym et al. 2017]) for exploration and visualization. In this work we propose a novel technique for constructing 2D exploration space from arbitrary high-dimensional latent spaces learned with some shape generation techniques.

Learning to deform 3D shapes Classical methods typically define shape deformation in terms of optimizing a physical or quasi-physical energy, e.g. see the survey by Botsch and Sorkine [2008]. These methods, while mathematically well-founded and tied to actual physical behavior, can be difficult to control, do not capture non-physical effects (e.g. interpolating in a heterogeneous shape space as studied in this paper) and frequently involve numerically challenging optimization (sometimes bypassable with "forward" models like skinning, e.g. [Jacobson et al. 2011]). To address these issues, a number of papers have tried to apply neural networks to learn deformation models in a data-driven way. Here, we discuss a few representative ones. To address the challenge of smoothly deforming complex geometry, several papers use machine learning to infer classical low-dimensional controls. Yifan et al. [2020] train a network to fit and deform low-dimensional cages for 3D shapes. Liu et al. [2021] take a different approach, learning combinations of shape control points that factorize the deformation space into intuitive "meta-handles". Xu et al. [2022] learn to infer animation rigs for meshes from motion-captured point cloud sequences. In contrast, Aigerman et al. [2022] learn an implicit Jacobian field that deforms meshes to targets *without* an intermediate classical proxy. While all of these methods provide interesting ways to deform or interpolate between meshes, they are not well suited for shape space exploration, which is the problem we study in this paper.

A work more directly related to ours is the ShapeFlow system of Jiang et al. [2020]. This system learns, in parallel, neural flow fields that deform one shape to another, as well as an embedding of all training shapes to a latent space. In principle, ShapeFlow can be applied to our problem, and we show in the evaluation section that our method produces qualitatively better deformations, see Figure 8. However, beyond that, ShapeFlow does not optimize the latent space for interactive two-dimensional exploration – thus, much of the machinery we develop for extracting a human-friendly 2D re-embedding of the original latent space would be necessary in any case. Secondly, our method can leverage any arbitrary pre-trained generative 3D model as a backbone, including ones trained on much larger 3D (or 2D) datasets than the collection being explored.

A method that does use latent-space generative models for shape space exploration is the GLASS system of Muralikrishnan et al. [2022]. However, the problem addressed by this paper is very different from ours – they focus on discovering new deformations of a single template mesh by alternately training a generative model, and incrementally exploring its latent space guided by a pre-defined physical energy. This method does not apply to heterogeneous shape collections, nor does it address interactive exploration.

3 OVERVIEW

Given a pre-trained, unstructured shape generative model $\mathcal{G}$ (e.g. a point cloud generator) and a set of *landmark meshes* $\mathcal{M} = \{M_1, ..., M_N\}$ from the same shape category of which $\mathcal{G}$ was trained, our goal is to create a two-dimensional *exploration space*, $\mathcal{E}$, that can be used to navigate a deformation subspace over the landmark meshes induced by $\mathcal{G}$. Walking through exploration space should produce smoothly varying and sensible deformations of the meshes, and the space's layout should be such that similar meshes in $\mathcal{M}$ are embedded closer together, while dissimilar meshes are farther apart.

We define the exploration space via a function Φ that maps from $\mathcal{E}$ to the generator's latent space $\mathcal{Z}$. The mapping is designed to give the exploration space several desirable properties that the generator lacks; for instance, interpolating between shapes in $\mathcal{E}$ produces intermediate outputs that vary smoothly, whereas naive linear interpolation through most generative models may produce samples that contain excessive variation, as illustrated in Figure 2. At a high level, Φ reparametrizes $\mathcal{Z}$ in a way that minimizes energy in the *primal space* of the generator (its geometric output space), thus avoiding such issues. Additionally, Φ allows the region of $\mathcal{Z}$ spanned by our given meshes to be visualized in two dimensions, making the high-dimensional latent space easily navigable.

Armed with Φ, there is one undesirable property remaining: $\mathcal{E}$ only permits exploration over the unstructured outputs of $\mathcal{G}$ (e.g. point clouds), rather than variations of the high-quality input meshes. To solve this problem, we introduce a deformation module that interprets an interpolation through the latent space of $\mathcal{G}$ as a flow on a mesh's vertices, producing detail-preserving deformations of a wide variety of shapes. We show how to advect the input meshes through this flow, as well as how to switch between which mesh is being advected based on a partitioning of the 2D space.

In the following two sections of the paper, we first outline the construction of the energy-minimizing map Φ into the generator's latent space. Then, we introduce our routine that transforms the unstructured shape manifold into a mesh deformation space.

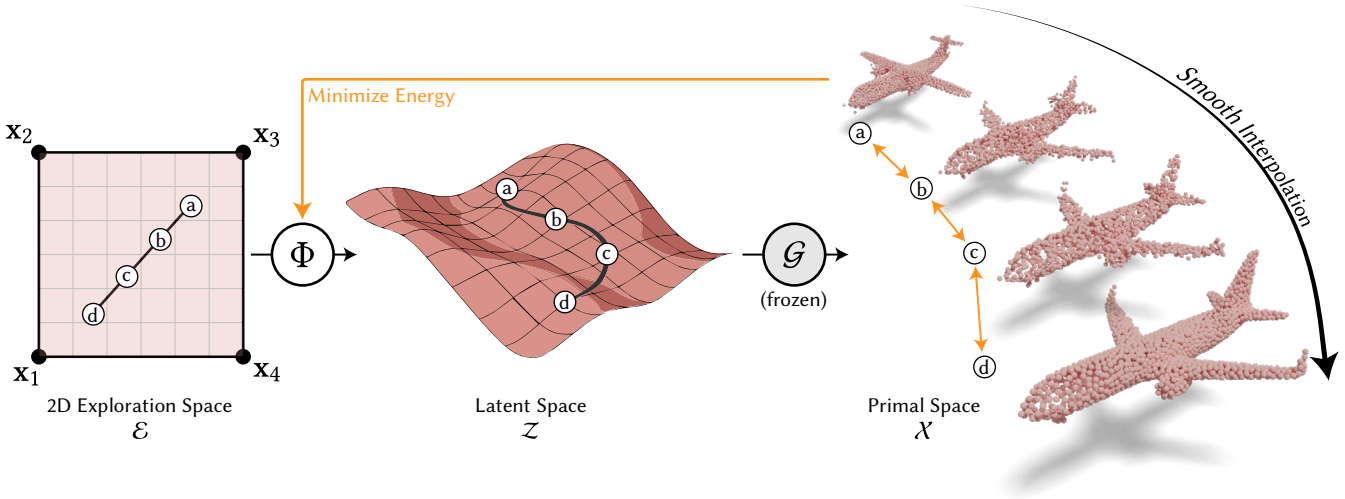


Figure 3: We illustrate the effect of our energy-minimizing submanifold in a simple case with four landmarks in exploration space. The region inscribed by the landmarks is mapped to $\mathcal{G}$'s latent space via Φ, as shown by the four interior points a, b, c, d. These points are decoded by the generator, producing point clouds of a particular shape category. The energy over these point clouds is said to be minimized by Φ through the optimization outlined in Section 4.2.

4 CONSTRUCTING THE 2D EXPLORATION SPACE $\mathcal{E}$

We now detail the inner workings of our exploration space. We assume to be given a *shape manifold*, i.e., a continuous space in which each point represents a shape — this space can be thought of as a submanifold of primal space (the space of all possible shapes). We consider a shape manifold defined by a pretrained generative neural network, $\mathcal{G}$. Concretely, such generators indeed define a k-manifold, if one considers it as a function taking us from latent (Euclidean) space, $\mathcal{Z} \in \mathbb{R}^k$, onto points in primal space $\mathcal{G}(\mathbf{z}) \subset \mathcal{X}$.

Our goal is to extract a submanifold from the shape manifold that 1) includes all of the given landmark shapes; 2) naturally interpolates between the landmarks along the shape manifold 3) is a 2-manifold, i.e., is parameterized via a map Φ, which maps the two-dimensional space into the shape manifold, $\Phi : \mathcal{E} \subset \mathbb{R}^2 \mapsto \mathcal{Z}$.

Say we have a two-dimensional embedding of our landmark meshes (detailed in Section 4.4) that will act as our parametrization of the aforementioned map, i.e. for each mesh we associate with it a point in exploration space (2D) and its corresponding projection into latent space: $\mathbf{L} = \{(\mathbf{x}_1, \mathbf{z}_1), ..., (\mathbf{x}_N, \mathbf{z}_N)\}$. As illustrated in Figure 3 we seek a map that "naturally" lifts this exploration space into the shape manifold, i.e., in a manner that does not exhibit excessive variation or rapid changes in the features of the shapes.

We can formalize this requirement: the shape manifold may be defined in the Euclidean space $\mathcal{Z}$, however the shapes themselves are generated by a non-linear mapping (the generator), thus small, linear steps in $\mathcal{Z}$ may result in arbitrarily large changes in the resulting shape. In other words, we need to account for the *metric* of the shape manifold and the curvature it induces. We will require that the map Φ is smooth w.r.t the metric of the shape manifold.

By casting this as an embedding problem, we formulate a constrained optimization objective that ensures the landmark shapes are always mapped correctly and the embedding minimizes an energy accounting for the metric of the shape manifold – the Dirichlet energy [Karcher 1977]. We first discuss this energy in the context of one-dimensional interpolation problems. Then, the two-dimensional problem is laid out for a simple case where just three landmarks are specified in Section 4.2. Finally, we generalize our method for collections of scattered landmarks in Section 4.3.

4.1 Geodesic paths between landmarks

Linear interpolation in high-dimensional latent spaces is known to produce samples that exhibit undesirable properties [Karras et al. 2018; Laine 2018]. Intermediate samples may contain features that do not exist at either the source or target; similarly, $\mathcal{G}$ may contain regions with excessive variation, resulting in a "jittery" output.

In order to avoid such interpolants, we optimize for paths in latent space that are *geodesics* with respect to primal space, $\mathcal{X}$. This is done by finding paths, $p(t), t \in [0, 1]$, where $p(0) = \mathbf{z}_0, p(1) = \mathbf{z}_1$, that minimize the *Dirichlet energy* in primal space

$$\mathcal{J}_{\text{1-D}}(p) = \tfrac{1}{2} \int_0^1 \|\nabla \mathcal{G}(p(t))\|^2 dt. \tag{1}$$

As geodesics are (locally) shortest paths, the shapes along them vary as gradually as possible, and thus do not contain superfluous movement in the generated samples. Figure 2 illustrates the difference between linear and geodesic interpolation through $\mathcal{Z}$.

These geodesics are useful for smoothly interpolating between samples along a one dimensional path, but they provide no information about smoothly interpolating through a solid region in latent space. In the simplest case, imagine three shapes connected by geodesic paths; we have a well-defined way to walk directly from shape to shape, but this formulation does not tell us how to walk to an arbitrary point in the *interior* of the region.

4.2 Energy-minimizing submanifolds of $\mathcal{G}$

We now formulate the constrained optimization problem in a case where three landmarks $\mathbf{z}_1$, $\mathbf{z}_2$, and $\mathbf{z}_3$ are specified. Then, our solution is generalized in Section 4.3. Since geodesic paths between any

pair of landmarks map to straight lines in exploration space $\mathcal{E}$, the three landmarks and the geodesic paths connecting them, $p_{12}(t)$, $p_{23}(t)$, $p_{31}(t)$ form the vertices $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ and edges $\vec{\mathbf{x}}_{12}, \vec{\mathbf{x}}_{23}, \vec{\mathbf{x}}_{31}$ of a triangle in $\mathcal{E}$. We then define the solution surface $\mathcal{S}(x) = \mathcal{G}(\Phi(\mathbf{x}))$ with $\mathbf{x} \in \mathcal{E}$ inside the triangle as the minimizer of the Dirichlet energy w.r.t to the Jacobian $\mathbf{J}$ of $\mathcal{G} \circ \Phi$ over exploration space:

Submanifold objective

$$\arg\min_{\Phi} \quad \frac{1}{2} \int_{\mathcal{E}} \|\mathbf{J}(\mathbf{x})\|_F^2 \, dA,$$

subject to a Dirichlet boundary condition on $\partial\mathcal{E}$ given by $\vec{\mathbf{x}}_{ij}$

$$\Phi(\mathbf{x}) = b(\mathbf{x}) \text{ if } \mathbf{x} \in \vec{\mathbf{x}}_{ij}$$

$$\text{with } b(\mathbf{x}) = p_{ij}\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|}{\|\mathbf{x}_j - \mathbf{x}_i\|}\right)$$

In order to optimize for Φ, we start by discretizing the function $\mathcal{G}(\Phi(\mathbf{x}))$ on a dense finite-element triangulation $(\mathbf{V}, \mathbf{F})$ over the exploration space, with vertices $\mathbf{V}$ and faces $\mathbf{F}$. Details of this discretization can be found in Section 6. Assuming the discretized function is piecewise linear on each triangle, the Jacobian $\mathbf{J}$ is piecewise constant, and we denote the Jacobian for each triangle $f_i \in \mathbf{F}$ as $\mathbf{J}^i$. The discretized objective function for our optimization is then

$$\mathcal{J}_{\text{2-D}} = \sum_{f_i \in \mathbf{F}} \|\mathbf{J}^i\|_F^2 \cdot \text{Area}(f_i). \tag{2}$$

In order to enforce our boundary condition, we could add a point-matching regularizer in similar spirit to the thin plate spline energy; thus making the energy

$$\mathcal{J}_{\text{reg}} = \sum_{f_i \in \mathbf{F}} \|\mathbf{J}^i\|_F^2 \cdot \text{Area}(f_i) + \underbrace{\lambda \int_{\partial\mathcal{E}} \|b(\mathbf{x}) - \Phi(\mathbf{x})\|^2 dA}_{\text{unstable soft constraint}}.$$

This energy, however, is unstable in practice due to the regularizer acting as a *soft* constraint. The choice of λ may catastrophically cause the solution surface to collapse to a point, or cause the surface to have regions with large gradients. Instead, we treat $\Phi(\mathbf{x})$ as a semi-parametric function, where its functional value is replaced with the corresponding boundary value $b(\mathbf{x})$ if $\mathbf{x} \in \partial\mathcal{E}$, thereby "pinning" the boundary points in place, turning them into a *hard* constraint. The remaining regions of the surface are represented by a multi-layer perceptron parametrized by θ:

$$\Phi(\mathbf{x}) = \begin{cases} b(\mathbf{x}) & \text{if } \mathbf{x} \in \partial\mathcal{E} \\ \text{MLP}_\theta(\mathbf{x}) & \text{otherwise.} \end{cases} \tag{3}$$

In summary, we resolve the optimal mapping by minimizing the energy given by Equation 2 w.r.t. the semi-parametric MLP above.

4.3 Generalizing to scattered landmarks in $\mathcal{E}$

Rather than operating on just three landmarks, our method facilitates exploration across an entire collection of shapes by defining Φ as a map from a convex region in $\mathcal{E}$ populated by a set of scattered landmarks $\mathbf{L}$. However, Laplace's equation (which is implicitly being solved by our energy) and similar PDEs are ill-defined with scattered boundary conditions[1].

[1]See anonymous StackExchange answer [2014] for further details.

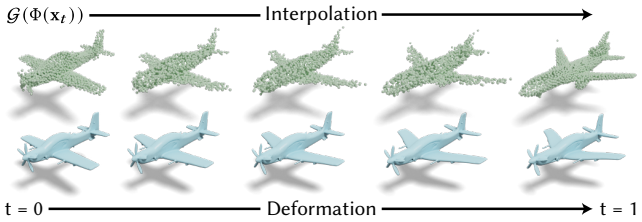
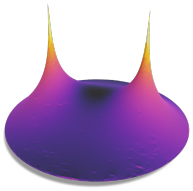


Figure 4: We exploit $\mathcal{G}$'s unstructured shape manifold to produce plausible deformations of meshes. Our deformation module is driven by interpolations through our map $\mathcal{G}(\Phi(\mathbf{x}))$, which offers smoother variation through exploration space.



For instance, if we take our boundary condition to be, say, a set of values along the convex hull of $\mathcal{E}$ along with the points in $\mathbf{L}$, then the solution surface $\mathcal{S}$ will exhibit a "tent-pole" effect—areas with large gradients will protrude from the surface, like poles poking through a circus tent (see inset figure).

Instead, we partition $\mathcal{E}$ into triangular facets via a Delaunay triangulation of the landmarks. This creates disjoint regions, which can be thought of as separate interpolation problems; that is, we can imagine $\mathcal{S}$ as being defined by a collection of neighboring surface patches that are "stiched" together. So long as each patch shares identical boundary conditions with its neighbors, then $\mathcal{S}$ will be continuous. We map each edge $(\mathbf{x}_i, \mathbf{x}_j)$ in the triangulation to a corresponding geodesic connecting $\mathbf{z}_i$ to $\mathbf{z}_j$. Hence our full boundary condition becomes $\text{lerp}(\mathbf{x}_i, \mathbf{x}_j) \hookrightarrow \text{geodesic}(\mathbf{z}_i, \mathbf{z}_j)$ for all edges in the triangulation, where lerp is linear interpolation.

4.4 Embedding meshes into $\mathcal{E}$

To initially embed our meshes into $\mathcal{E}$, we create a k-nearest neighbor graph of the meshes based on a suitable distance metric (specified in Section 6). This graph is embedded into $\mathcal{E}$ using a triplet margin loss [Schroff et al. 2015], where triplets of meshes (a tuple comprised of an anchor, positive neighboring mesh, and negative non-neighbor) are used to minimize distances between the anchors and their neighbors, while maximizing their distance to the remaining shapes. The triplet loss over all such tuples is given as

$$\mathcal{L}_{\text{triplet}}(a, p, n) = \sum_{i}^{k \cdot N} \max(\|a_i - p_i\|_2^2 - \|a_i - n_i\|_2^2 + \alpha, 0)$$

where α is the margin parameter that controls the separation between positive and negative pairs. We note that the choice of triplet loss is arbitrary (methods such as t-SNE or UMAP may be used as well [McInnes et al. 2020; van der Maaten and Hinton 2008]); however, in our experiments we found the embeddings produced by the triplet loss contained sufficient local similarity between landmarks.

We include additional regularizer terms that maintain uniformity in spacing between the embedded points. In particular, we penalize areas and interior angles of triangles induced by a Delaunay triangulation of the embedded points via ℓ^2 norms of the areas and minimum interior angles of each triangle. Avoiding a poor triangulation of these points is pertinent to the boundary conditions established in Section 4.2.

5 EXPLORABLE MESH DEFORMATION SUBSPACES

We now have an explorable subspace—parametrized by our two-dimensional energy-minimizing map Φ—in $\mathcal{Z}$ that spans a set of landmark shapes. Instead of simply synthesizing unstructured shapes from $\mathcal{G}$ (i.e. point clouds), we would like to produce high quality *meshes* that vary smoothly as we walk through the subspace. In the following sections, we introduce a novel method for transforming $\mathcal{E}$ into a mesh deformation subspace. Then, we generalize this method to accommodate meshes with varying topology.

5.1 Transforming $\mathcal{E}$ into a mesh deformation subspace

Given a landmark point in exploration space, $\mathbf{x}_{\text{src}} \in \mathcal{E}$, and its corresponding mesh $\mathbf{M}(\mathbf{V}, \mathbf{F}) \in \mathcal{M}$, how can we deform the mesh to match an arbitrary point $\mathbf{x}_{\text{tar}}$ whose structure is given by $\mathcal{S}(\mathbf{x}_{\text{tar}})$? The key to our deformation module is to interpret an interpolation through exploration space as a flow on the mesh's vertices, $\mathbf{V}$.

More concretely, say we have a continuous path on our submanifold, $\mathrm{S} = (\mathcal{S}(\mathbf{x}_0), ..., \mathcal{S}(\mathbf{x}_t) \mid \forall t \in [0, 1])$ where $\mathbf{x}_0 = \mathbf{x}_{\text{src}}$, $\mathbf{x}_1 = \mathbf{x}_{\text{tar}}$. At time t, an instantaneous discrete flow field is induced by

$$f_t : \mathrm{S}_t \rightarrow \mathrm{S}_{t+\epsilon} - \mathrm{S}_t,$$

assuming we have a dense correspondence between the samples on $\mathcal{S}$. This formulation is similar to prior flow-based deformation techniques [Xu et al. 2022]; however, due to our choice of $\mathcal{G}$, we do not need to explicitly compute point correspondences (see Section 6 for details). We deform our mesh by integrating the flow vectors on each mesh vertex. Since the flow field f is spatially discrete, it must be interpolated to be defined at all vertex locations $\mathbf{v} \in \mathbf{V}$. One such method for doing this is by means of a radial basis function (RBF) interpolator [Anjyo et al. 2014]. This method is particularly well-suited for our task because RBF interpolation is grid-free (f does not lie on a regular grid), and it exhibits smooth and stable interpolation for large numbers of points. Flowing each mesh vertex, $\mathbf{v}^i$, can now be done by integrating the flow field through time

$$\mathbf{v}^i_{t+\epsilon} = \mathbf{v}^i_t + \mathrm{RBF}_{f_t}(\mathbf{v}^i_t) \tag{4}$$

We employ a smoothing RBF interpolator that takes as input a parameter λ, which controls how well the interpolant fits the displacement vectors f_t. An example of our mesh deformations is shown in Figure 4, where we see that interpolations over our energy-minimizing submanifold provide smooth flows on the mesh vertices.

5.2 Topology-aware deformation subspaces

Ultimately, we define a smooth space of mesh deformations over a collection of meshes $\mathcal{M}$, akin to the latent space of a generative model. This goal comes with an added challenge: the meshes in $\mathcal{M}$ have varying topology, necessitating a need to discretely switch between topologies while walking through our space. At a high level, we establish a Voronoi partitioning of $\mathcal{E}$, such that the mesh topology at any point $\mathbf{x} \in \mathcal{E}$ is decided by the nearest landmark's corresponding mesh. In the one dimensional case (i.e. with two landmarks and a line connecting them), this is equivalent to switching mesh topologies at the midpoint $t = 0.5$.

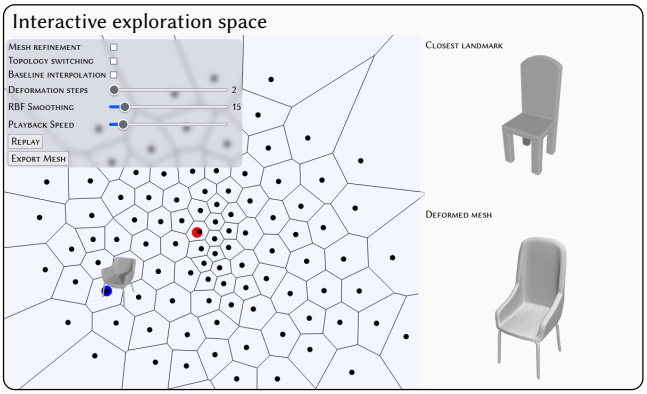


Figure 5: Our interface for real-time shape exploration and deformation, which facilitates interactive exploration of continuous deformations of the landmark meshes. Source and target points are marked by blue and red.

It is apparent, both qualitatively and quantitatively, that naively switching topologies at $t = 0.5$ is suboptimal. We instead compute the optimal switch point, t^*, as the time at which the switch incurs minimal change in the deforming mesh. In order to preserve the Voronoi partioning as-is, we transform the boundary conditions on each Delaunay edge to remap t^* to $t = 0.5$. Hence, in the final exploration space, the optimal switch points are demarcated simply along Voronoi cells. Quantitative comparisons can be found in Section 7.

6 IMPLEMENTATION

All experiments were conducted on a single NVIDIA RTX 3090. We use PyTorch and Adam for all optimization routines [Kingma and Ba 2014; Paszke et al. 2019]. Details regarding the computation of boundary conditions, discretization of exploration space, training for Φ, and embedding of landmark meshes can be found in the supplemental material. We additionally implement a graphical user interface that allows one to interact with our explorable deformation space in real time, as seen in Figure 5 and included videos.

Shape generator. For our generator $\mathcal{G}$, we employ SP-GAN [Li et al. 2021], a state-of-the-art point cloud GAN. SP-GAN is particularly well-suited for our goals because it is capable of representing detailed shapes while implicitly providing a dense correspondence between them, which we utilize in our shape deformation module. SP-GAN takes as input a *latent matrix* which contains a 128-d Gaussian i.i.d vector for each of the 2048 points in the resulting point cloud, thus the generative mapping is $\mathcal{G} : \mathcal{Z} \in \mathbb{R}^{2048\times128} \mapsto \mathcal{X} \in \mathbb{R}^{2048\times3}$.

7 RESULTS AND EVALUATION

We test our method using SP-GAN trained on ShapeNet as our shape generator $\mathcal{G}$ [Chang et al. 2015]. In particular, we construct three exploration spaces containing chairs, tables, and airplanes—these spaces contain 100, 50, and 25 shapes respectively and are denoted as *chairs-100*, *tables-50*, *airplanes-25*. We demonstrate the flexibility of our exploration space by showing applications such

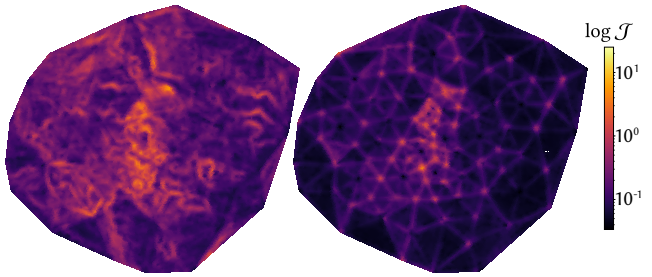


Figure 6: Visualization of the log-energy, as per Equation 2. Naively lifting exploration spaces into the primal domain (e.g. by barycentrically interpolating landmark latents) results in noisy interpolants. Our method incurs far less energy overall, which primarily occurs on the facet boundaries.

as shape-to-shape deformation (i.e. matching the structure of a target shape with the geometry of a source shape), and free-form exploration of the exploration space (i.e. interpolating in $\mathcal{E}$ freely without a particular target shape). Our results are compared with ShapeFlow quantitatively and qualitatively; and we similarly use the ShapeNet Manifold dataset [Huang et al. 2018; Jiang et al. 2020].

Shape-to-shape deformation. Our results on shape-to-shape deformation are presented in Figure 8, whereby source and target landmarks are connected via linear paths in exploration space. We additionally apply our deformation method to an out-of-distribution shape, as seen in Figure 7. We compare our results qualitatively and quantitatively with ShapeFlow. The landmark pairs were chosen randomly. We only show comparisons for shapes that are in ShapeFlow's training data (out-of-distribution shapes are not selected). We specifically compare against ShapeFlow's chair model, as other models were unavailable.

We evaluate our deformations quantitatively by measuring their distributional similarity to the ShapeNet dataset under the Frechet Distance (FD) of a pre-trained PointNet++ classifier's feature space [Heusel et al. 2017; Qi et al. 2017]. Table 1 presents our FD scores alongside ShapeFlow and SP-GAN's meshed outputs (using SAP [Peng et al. 2021]). Meshing the point clouds directly leads to geometry that lacks detail and contains artifacts (i.e. holes). We also measure similarity of deformed shapes to their targets under earth mover's distance (EMD). Our deformations are more natural: they preserve the original structure and match the target shapes more closely without significant distortion. ShapeFlow fits the target slightly better under EMD, but this is due to overfitting to a Chamfer distance loss, leading to unnatural surface bowing.

Free-form shape deformation. We demonstrate the advantage of our two-dimensional exploration space by showing free-form shape deformations, whereby a trajectory starting from a landmark shape can be specified without a particular landmark target. Visualizations of these trajectories can be found in Figure 10.

Local smoothness of Φ. We measure the effect of our mapping on local 1-d trajectories through the exploration space. Specifically, we sample random paths through $\mathcal{E}$ with lengths proportional to Delaunay facets bounding box dimensions. We compare the energy of our paths to linear paths and optimized geodesics in Table 2. Additionally, the Dirichlet energy of our mapping over the entire exploration space is visualized in Figure 6 alongside a baseline.

Table 1: FD and EMD scores of our chair deformations compared with Shapeflow and SP-GAN's meshed outputs. For FD we sample at times $t = 0.5, 1.0$. We measure EMD between the terminal shape and the target.

Method	FD $\{t = 0.5, 1.0\} \downarrow$	EMD $\{t = 1.0\} \downarrow$
Ours	**31.6**	0.082
ShapeFlow	78.4	**0.069**
SP-GAN (meshed)	79.2	-

Table 2: Energy of paths from our *chairs-100* exploration space, versus linear interpolations in $\mathcal{Z}$, and optimized geodesics. Our mean energy is smaller than the optimized geodesics, suggesting that the local support provided by our 2-d method gives smoother results than 1-d interpolation in general.

Metric	Ours	$\mathcal{Z}$-linear	$\mathcal{Z}$-opt
Mean Energy $\downarrow$	**1.071**	1.267	1.144
Max Energy $\downarrow$	3.741	2.796	**2.415**

Table 3: The amount of change incurred by the meshes in *tables-50* exploration space at random switch points is computed. We compare using remapping of optimal switch points, t^*, versus no remapping.

Metric	Remapping t^*	w/o Remapping
Mean CD $\downarrow$	**262**	285
Max CD $\downarrow$	493	**488**

Boundary remapping. We measure the effect of our boundary remapping (Section 5.2). The average Chamfer distance of shapes immediately before and after switching topology are in Table 3. We see that remapping boundary conditions moderately reduces the geometric distance between shapes at the switch points on average.

8 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a new technique for extracting an easily-navigable space for exploring deformations within and between high-quality meshes using a pre-trained 3D shape generative model. We described an algorithm for computing a smooth 2D parametrization of the generative model's latent space that interpolates a set of given landmark shapes, and we showed how to turn paths along the manifold induced by our parametrization into continuous deformation fields for transferring shape variations from the latent space to high-quality landmark meshes. We demonstrated our technique by constructing exploration spaces for several shape categories and showed that it produces better interpolations between shapes than an alternative approach, including a recently proposed method for directly learning a deformation space from a large shape collection.

Limitations. Our method is not without limitations. For instance, switching between meshes can result in visual discontinuities, especially if the landmarks are tightly packed. One way to alleviate this issue would be to produce more gradual topology changes between the two meshes, rather than a single discrete switch: for example, by gradually replacing parts of the source mesh [Hui et al. 2022a;

Figure 7: An out-of-distribution airplane[2] (left), unseen by the pre-trained generator, and a resulting mesh from our deformation pipeline (right).

Jain et al. 2012]. Additionally, like other flow-based deformation methods, our method may produce warped geometry, particularly when 'rigid' parts are deformed non-rigidly (e.g. column 3 in Figure 8). Imposing regularization energies (e.g. an ARAP energy [Huang et al. 2021; Sorkine and Alexa 2007]) on the induced flow field is a potential avenue for improvement. Finally, while our method supports real-time interaction, the training remains time consuming. Amortizing the training of Φ by *learning to predict* an exploration space given a set of landmarks is of much interest. Such a system could facilitate powerful forms of interaction: e.g., 'coarse-to-fine' exploration, in which a user starts with an exploration space over a large set of shapes, then iteratively drills down into spaces of smaller subsets of shapes. Currently our method cannot easily accommodate this interaction paradigm.

Future work. One immediate direction for future work is to apply our method to other latent variable generative models. The challenge here is in producing deformation fields when there does not exist a dense correspondence in $\mathcal{G}$'s output space (as in SP-GAN). Also, given their recent popularity, it would be interesting to apply our method to diffusion models. The challenge here is the expensive computation of Jacobians that are necessary for training the map Φ. One possibility is to train a surrogate forward-pass model that approximates multiple diffusion steps [Xiao et al. 2022].

It would also be interesting to apply the subcomponents of our method to different applications or domains. For instance, our procedure for building $\mathcal{E}$ is not specific to 3D shape latent spaces; one could imagine using it to construct easily-navigable exploration spaces for other visual data domains, such as image collections.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 2040433.

REFERENCES

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas J. Guibas. 2017. Learning Representations and Generative Models for 3D Point Clouds. In *International Conference on Machine Learning*.

Noam Aigerman, Kunal Gupta, Vladimir Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. 2022. Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 41, 4 (2022).

[2]Airplane model from CGTrader user Boba3D.

Ken Anjyo, J. P. Lewis, and Frédéric Pighin. 2014. Scattered Data Interpolation for Computer Graphics. In *ACM SIGGRAPH 2014 Courses* (Vancouver, Canada) *(SIGGRAPH '14)*. Association for Computing Machinery, New York, NY, USA, Article 27, 69 pages. https://doi.org/10.1145/2614028.2615425

Melinos Averkiou, Vladimir G. Kim, Youyi Zheng, and Niloy J. Mitra. 2014. ShapeSynth: Parameterizing Model Collections for Coupled Shape Exploration and Synthesis. *Computer Graphics Forum (Proc. of Eurographics)* 33, 2 (2014).

Mario Botsch and Olga Sorkine. 2008. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics* 14, 1 (2008), 213–230.

Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. ShapeNet: An Information-Rich 3D Model Repository. *CoRR* abs/1512.03012 (2015). arXiv:1512.03012 http://arxiv.org/abs/1512.03012

Siddhartha Chaudhuri, Evangelos Kalogerakis, Stephen Giguere, and Thomas Funkhouser. 2013. AttribIt: Content Creation with Semantic Attributes. 193–202.

Siddhartha Chaudhuri, Evangelos Kalogerakis, Leonidas Guibas, and Vladlen Koltun. 2011. Probabilistic Reasoning for Assembly-Based 3D Modeling. *ACM Transactions on Graphics* 30, 4 (2011), 35.

Zhiqin Chen and Hao Zhang. 2019a. Learning Implicit Fields for Generative Shape Modeling. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

Zhiqin Chen and Hao Zhang. 2019b. Learning Implicit Fields for Generative Shape Modeling. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

Nadav Dym, Haggai Maron, and Yaron Lipman. 2017. DS++: A Flexible, Scalable and Provably Tight Relaxation for Matching Problems. In *ACM SIGGRAPH Asia*.

Ohad Fried, Stephen DiVerdi, Maciej Halber, Elena Sizikova, and Adam Finkelstein. 2015. IsoMatch: Creating Informative Grid Layouts. *Computer Graphics Forum (Proc. Eurographics)* 34, 2 (May 2015).

Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao (Richard) Zhang. 2019. SDM-NET: Deep Generative Network for Structured Deformable Mesh.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Nash Equilibrium. *CoRR* abs/1706.08500 (2017). arXiv:1706.08500 http://arxiv.org/abs/1706.08500

Jingwei Huang, Hao Su, and Leonidas Guibas. 2018. Robust Watertight Manifold Surface Generation Method for ShapeNet Models. *arXiv preprint arXiv:1802.01698* (2018).

Qixing Huang, Xiangru Huang, Bo Sun, Zaiwei Zhang, Junfeng Jiang, and Chandrajit Bajaj. 2021. ARAPReg: An As-Rigid-As Possible Regularization Loss for Learning Deformable Shape Generators. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 5795–5805. https://doi.org/10.1109/ICCV48922.2021.00576

K. Hui, R. Li, J. Hu, and C. Fu. 2022a. Neural Template: Topology-aware Reconstruction and Disentangled Generation of 3D Meshes. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society. https://doi.org/10.1109/CVPR52688.2022.01802

Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. 2022b. Neural Wavelet-Domain Diffusion for 3D Shape Generation. In *SIGGRAPH Asia 2022 Conference Papers* (Daegu, Republic of Korea) *(SA '22)*. Association for Computing Machinery, New York, NY, USA, Article 24, 9 pages. https://doi.org/10.1145/3550469.3555394

Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. 2011. Bounded Biharmonic Weights for Real-Time Deformation. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 30, 4 (2011), 78:1–78:8.

Arjun Jain, Thorsten Thormählen, Tobias Ritschel, and Hans-Peter Seidel. 2012. Exploring Shape Variations by 3D-Model Decomposition and Part-based Recombination. *Comp. Graph. Forum (Proc. Eurographics 2012)* 31, 2 (2012).

Chiyu Jiang, Jingwei Huang, Andrea Tagliasacchi, and Leonidas Guibas. 2020. ShapeFlow: Learnable Deformations Among 3D Shapes. In *Advances in Neural Information Processing Systems*.

R. Kenny Jones, Theresa Barton, Xianghao Xu, Kai Wang, Ellen Jiang, Paul Guerrero, Niloy J. Mitra, and Daniel Ritchie. 2020. ShapeAssembly: Learning to Generate Programs for 3D Shape Structure Synthesis. *ACM Transactions on Graphics* 39, 6 (2020).

Evangelos Kalogerakis, Siddhartha Chaudhuri, Daphne Koller, and Vladlen Koltun. 2012. A Probabilistic Model of Component-Based Shape Synthesis. *ACM Transactions on Graphics* 31, 4 (2012).

H. Karcher. 1977. Riemannian center of mass and mollifier smoothing. *Communications on Pure and Applied Mathematics* 30, 5 (1977), 509–541. https://doi.org/10.1002/cpa.3160300502 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160300502

Tero Karras, Samuli Laine, and Timo Aila. 2018. A Style-Based Generator Architecture for Generative Adversarial Networks. *CoRR* abs/1812.04948 (2018). arXiv:1812.04948 http://arxiv.org/abs/1812.04948

Vladimir G. Kim, Wilmot Li, Niloy J. Mitra, Siddhartha Chaudhuri, Stephen DiVerdi, and Thomas Funkhouser. 2013. Learning Part-based Templates from Large Collections of 3D Shapes. *Transactions on Graphics (Proc. of SIGGRAPH)* 32, 4 (2013).

Vladimir G. Kim, Wilmot Li, Niloy J. Mitra, Stephen DiVerdi, and Thomas Funkhouser. 2012. Exploring Collections of 3D Models using Fuzzy Correspondences. *Transactions on Graphics (Proc. of SIGGRAPH)* 31, 4 (2012).

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

Yanir Kleiman, Joel Lanir, Dov Danon, Yasmin Felberbaum, and Daniel Cohen-Or. 2015. DynamicMaps: Similarity-Based Browsing through a Massive Set of Images. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, 995–1004.

Samuli Laine. 2018. Feature-Based Metrics for Exploring the Latent Space of Generative Models. https://openreview.net/forum?id=BJslDBkwG

Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. 2017. GRASS: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics* 36, 4 (2017), 52.

Ruihui Li, Xianzhi Li, Ke-Hei Hui, and Chi-Wing Fu. 2021. SP-GAN:Sphere-Guided 3D Shape Generation and Manipulation. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 40, 4 (2021).

Minghua Liu, Minhyuk Sung, Radomir Mech, and Hao Su. 2021. DeepMetaHandles: Learning Deformation Meta-Handles of 3D Meshes with Biharmonic Coordinates. *arXiv preprint arXiv:2102.09105* (2021).

Joe Marks, Brad Andalman, Paul A Beardsley, William Freeman, Sarah Gibson, Jessica Hodgins, Thomas Kang, Brian Mirtich, Hanspeter Pfister, Wheeler Ruml, et al. 1997. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 389–400.

Leland McInnes, John Healy, and James Melville. 2020. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv:1802.03426 [stat.ML]

Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. 2022. AutoSDF: Shape Priors for 3D Completion, Reconstruction and Generation. In *CVPR*.

Sanjeev Muralikrishnan, Siddhartha Chaudhuri, Noam Aigerman, Vladimir Kim, Matthew Fisher, and Niloy Mitra. 2022. GLASS: Geometric Latent Augmentation for Shape Spaces. In *CVPR*.

Charlie Nash, Yaroslav Ganin, SM Ali Eslami, and Peter Battaglia. 2020. Polygen: An autoregressive generative model of 3d meshes. In *International Conference on Machine Learning*. PMLR, 7220–7229.

Maks Ovsjanikov, Wilmot Li, Leonidas Guibas, and Niloy Mitra. 2011. Exploration of Continuous Variability in Collections of 3D Shapes. *ACM Transactions on Graphics (Proc. SIGGRAPH)* (2011).

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *CoRR* abs/1912.01703 (2019). arXiv:1912.01703 http://arxiv.org/abs/1912.01703

Songyou Peng, Chiyu "Max" Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. 2021. Shape As Points: A Differentiable Poisson Solver. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv preprint arXiv:1706.02413* (2017).

Raghad Rowaida. 2021. *Exploration of Latent Spaces of 3D Shapes via Isomap and Inverse Mapping*. Master's thesis. Carleton University.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. *CoRR* abs/1503.03832 (2015). arXiv:1503.03832 http://arxiv.org/abs/1503.03832

Olga Sorkine and Marc Alexa. 2007. As-Rigid-As-Possible Surface Modeling. *Symposium on Geometry Processing*, 109–116. https://doi.org/10.1145/1281991.1282006

Jerry O. Talton, Daniel Gibson, Lingfeng Yang, Pat Hanrahan, and Vladlen Koltun. 2009. Exploratory Modeling with Collaborative Design Spaces. 28, 5 (2009).

user147263. 2014. You cant solve Laplaces equation with boundary conditions on isolated points. But why? (2014). https://math.stackexchange.com/q/1031377

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, 86 (2008), 2579–2605. http://jmlr.org/papers/v9/vandermaaten08a.html

Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. 2018. Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images. In *ECCV*.

Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum. 2016. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. In *Advances In Neural Information Processing Systems*. 82–90.

Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. 2022. Tackling the Generative Learning Trilemma with Denoising Diffusion GANs. In *International Conference on Learning Representations (ICLR)*.

Zhan Xu, Yang Zhou, Li Yi, and Evangelos Kalogerakis. 2022. Morig: Motion-Aware Rigging of Character Meshes from Point Clouds. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* 41, 6 (2022).

Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge Belongie, and Bharath Hariharan. 2019. PointFlow: 3D Point Cloud Generation with Continuous Normalizing Flows. *arXiv* (2019).

Jie Yang, Kaichun Mo, Yu-Kun Lai, Leonidas J Guibas, and Lin Gao. 2020. DSM-Net: Disentangled Structured Mesh Net for Controllable Generation of Fine Geometry. *arXiv preprint arXiv:2008.05440* (2020).

Wang Yifan, Noam Aigerman, Vladimir G. Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. 2020. Neural Cages for Detail-Preserving 3D Deformations. In *CVPR*.

Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. 2022. LION: Latent Point Diffusion Models for 3D Shape Generation. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Xin-Yang Zheng, Yang Liu, Peng-Shuai Wang, and Xin Tong. 2022. SDF-StyleGAN: Implicit SDF-Based StyleGAN for 3D Shape Generation. In *Comput. Graph. Forum (SGP)*.

Figure 8: We demonstrate our shape-to-shape deformation results against ShapeFlow, whereby we take random source and target meshes, and deform between them continuously. We visualize the final deformed shapes (at $t = 1$) as well as the source and target meshes. We can see that our method exhibits deformations that better preserve the fine details of the original shapes, while matching the structure of the target shapes more closely compared to ShapeFlow.

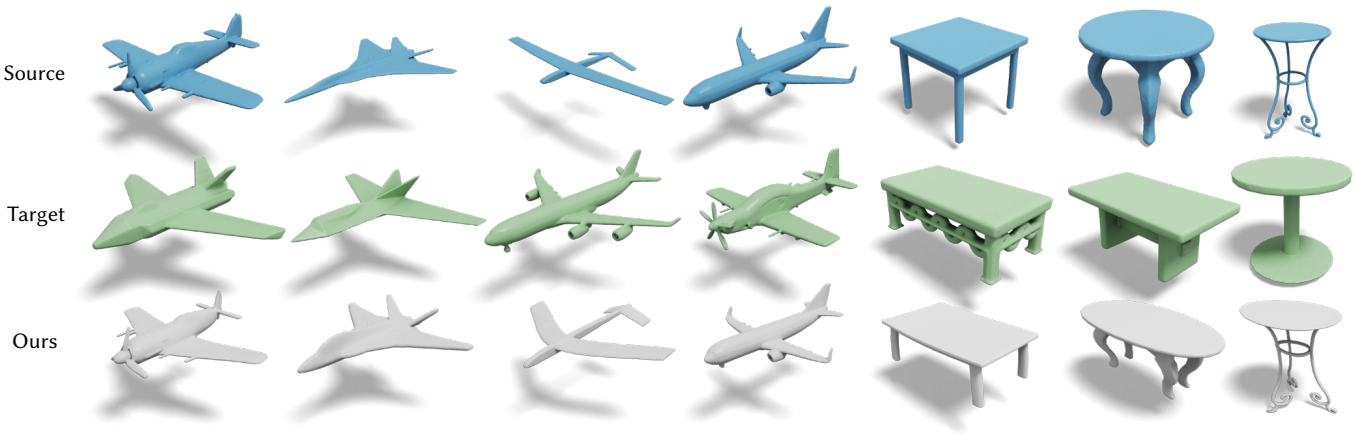


Figure 9: Additional shape-to-shape deformation results from our *airplanes-25* and *tables-50* exploration spaces. Our method is able to capture complex deformations; for instance, in column four we see the airliner's wings bending forward to match the straight wing's of the propeller plane. Additionally, in the last column we see that our method is able to locally deform the table top while maintaining the geometry elsewhere.

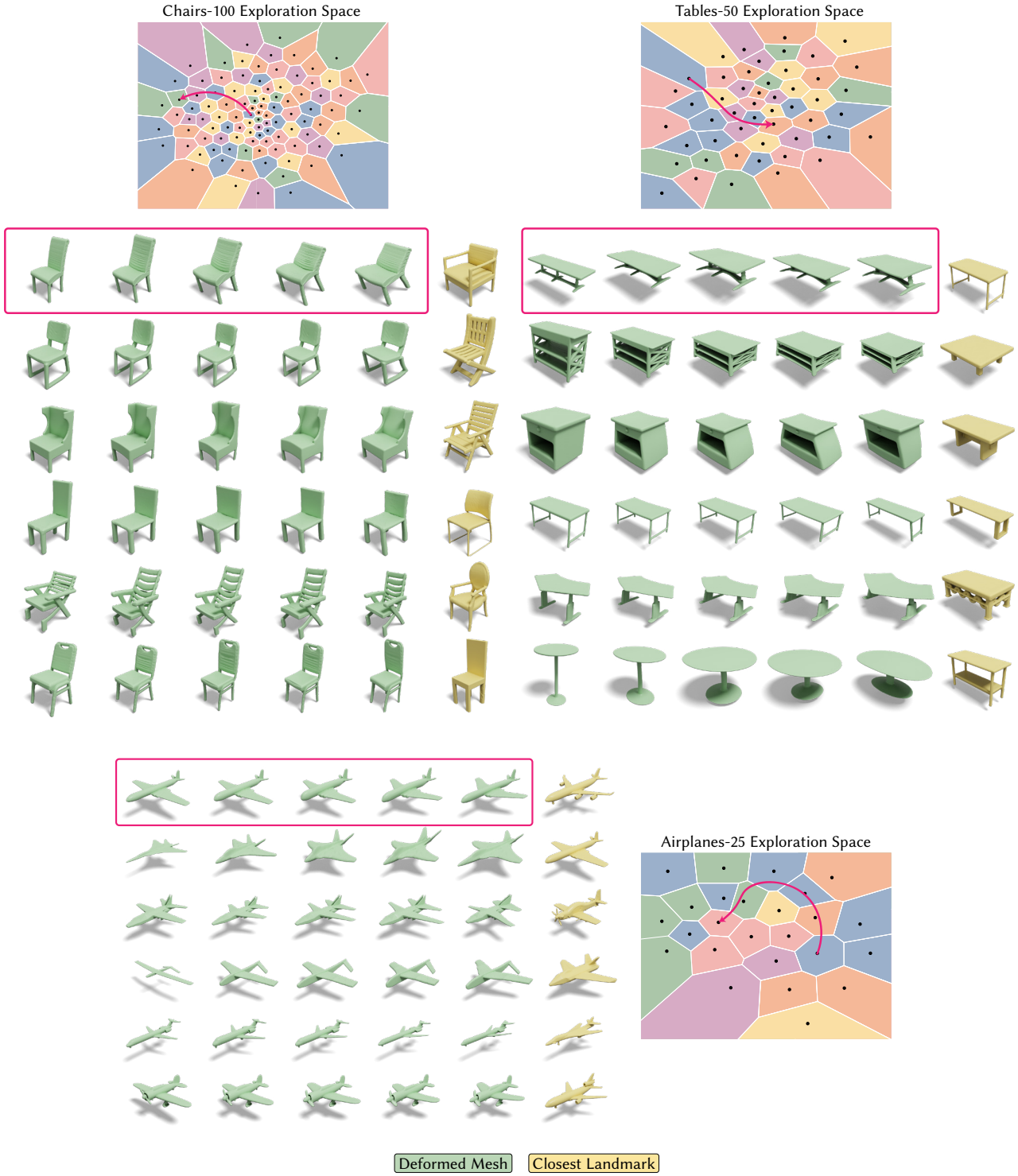


Figure 10: Our method also facilitates free-form exploration of the deformation spaces. Here we show example deformations (in green) in our *chairs-100*, *tables-50*, and *airplanes-25* exploration spaces respectively, where times $t = 0, 0.25, 0.5, 0.75, 1$ are shown, $t = 0$ corresponds to the source mesh. The closest landmark shapes (in yellow) located at the end of the deformation trajectories are shown, along with an example trajectories for the first rows of each category.