# Geometry in Style: 3D Stylization via Surface Normal Deformation

Nam Anh Dinh
University of Chicago

Itai Lang
University of Chicago

Hyunwoo Kim
University of Chicago

Oded Stein
University of Southern California

Rana Hanocka
University of Chicago

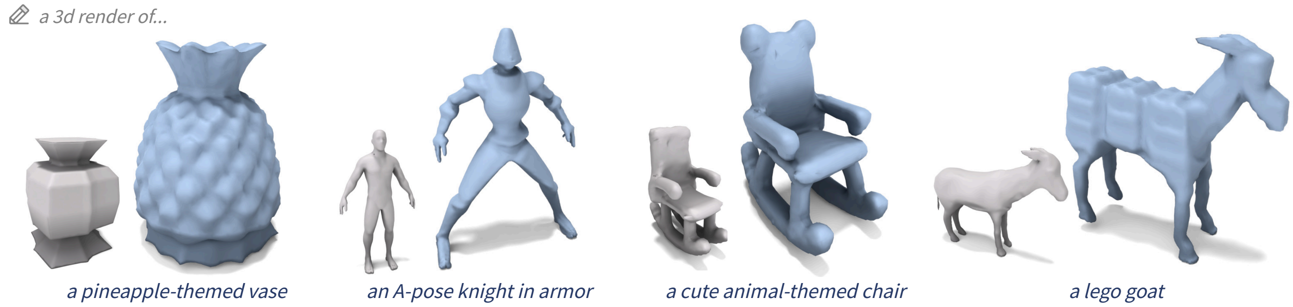



Figure 1. Our method deforms a source shape (gray) into a text-specified *semantic style* (blue). While the deformations are expressive, they *preserve the identity* of the original shape.

## Abstract

*We present Geometry in Style, a new method for identity-preserving mesh stylization. Existing techniques either adhere to the original shape through overly restrictive deformations such as bump maps or significantly modify the input shape using expressive deformations that may introduce artifacts or alter the identity of the source shape. In contrast, we represent a deformation of a triangle mesh as a target normal vector for each vertex neighborhood. The deformations we recover from target normals are expressive enough to enable detailed stylizations yet restrictive enough to preserve the shape's identity. We achieve such deformations using our novel differentiable As-Rigid-As-Possible (dARAP) layer, a neural-network-ready adaptation of the classical ARAP algorithm which we use to solve for per-vertex rotations and deformed vertices. As a differentiable layer, dARAP is paired with a visual loss from a text-to-image model to drive deformations toward style prompts, altogether giving us Geometry in Style. Our project page is at* `https://threedle.github.io/geometry-in-style`.

## 1. Introduction

Semantically deforming triangle meshes is a basic task in 3D surface modeling. A common paradigm for shape creation is to take a base 3D object and deform it to sculpt a desired shape. For example, a human artist creates an intricate surface by starting with a simple generic version of an object (from a shape library, or quickly sketched), and successively deforms parts of the object like a sculptor modeling clay.

Recently, machine learning pipelines have adopted deformations as a strategy for neural shape manipulation [2, 38, 77]. However, existing learning-based methods do not always fully mimic the classical approach: they either make small, surface-level changes for simple stylization [30, 60], or they generate large deformations that destroy the identity of the base object [26, 41]. Our method, *Geometry in Style*, uses text prompts to generate large deformations that create unique shapes while preserving the *identity* of the base shape (see Fig. 1) – much like a human sculptor would.

We posit that existing learning methods' failure to generate the kinds of large, identity-preserving deformations that a human artist would is due to not employing the right representation for the space of deformations. Existing methods either (a) adhere to the original shape through overly restrictive deformations, such as bump maps [30, 60]; or they (b) significantly modify the input shape using expressive deformations based on gradient fields that may introduce artifacts or destroy the identity of the original shape [26, 41].

In this work, a deformation of a triangle mesh is repre-

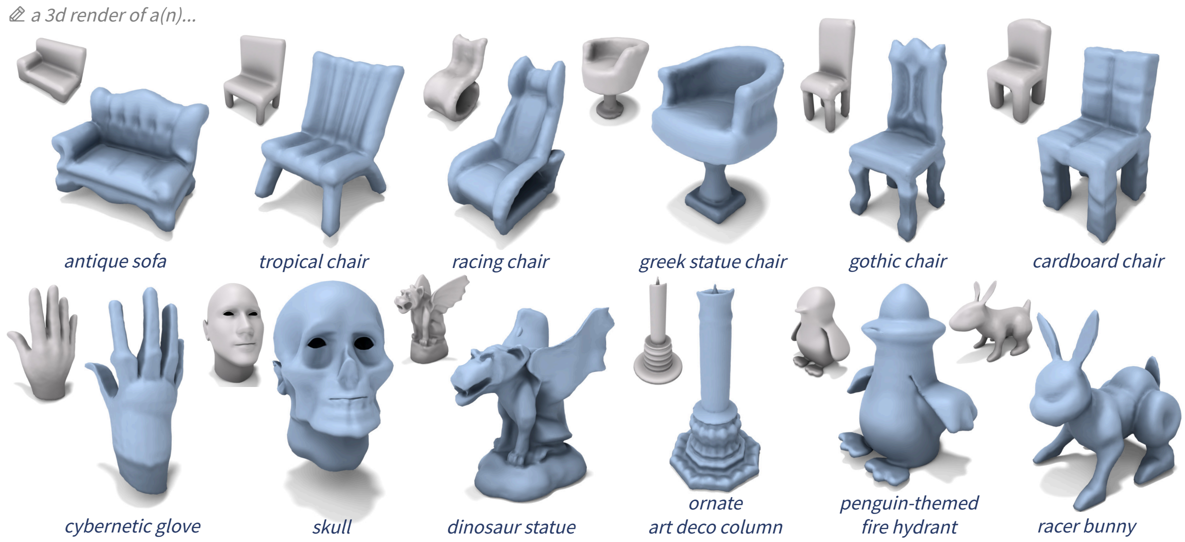



Figure 2. **Style diversity.** Our method is capable of deforming various input meshes towards a variety of text-specified styles. The style can be manifested as fine geometric details, like in the *ornate art deco column*, or as low-frequency deformations, such as the joints of the *cybernetic glove*. Our method retains the structural features of the input shape, such as a flat arm on the *antique sofa*. Moreover, the resultant stylizations are in accordance with prompt semantics and part-aware semantics: the folds in the tropical chair are on the seat and backrest as opposed to the legs, the head of the penguin becomes like the top of a fire hydrant, and the racer bunny's thigh turns into the shape of a wheel.

sented by target normals for vertex neighborhoods. We recover a deformation from this representation using our differentiable As-Rigid-As-Possible method (dARAP), whose formulation optimizing for local rigidity yields detailed and salient deformations that are nonetheless restrictive enough to preserve the identity of the base shape. dARAP locally rotates each vertex neighborhood individually to fit its normal to a desired target normal, and follows this local rotation with a global step that finds a global deformation that best fits all individual rotated neighborhoods. Crucially, dARAP is differentiable, and can be used as a layer in a neural network. This is achieved by replacing classical ARAP's iteration of local and global steps until convergence (impractical to backpropagate through) with dARAP's easily differentiable use of a single local and global step. Where classical ARAP needs many iterations to converge in deformation tasks with fixed target vertex positions, dARAP's use with target normals inside the *Geometry in Style* method (where dARAP is run once per iteration of a larger gradient descent optimization problem) achieves high-quality deformations from only a single iteration.

We use dARAP together with a visual loss from a text-to-image model (T2I) which drives our deformation to arrive at *Geometry in Style*. Our visual loss leverages a cascaded T2I model to achieve high-fidelity deformations [21], allowing the use of a user-specified text prompt to deform any base shape into a stylized object without any dedicated 3D supervision data. As such, our method allows the application of a wide variety of styles, indicated intuitively by text prompts, to a wide variety of shapes (Figs. 1 and 2). Our stylization can manifest as different types of geometric manipulation, such as local surface texture, as well as global low-frequency deformations. Additionally, we show that our method offers control over the deformation result, where the user can easily change the strength of the stylization effect even after optimization. We contrast our method with recent deformation techniques and find that it can better achieve the target style with a lower surface distortion.

In this work, we present:

- *dARAP*, a differentiable neural network layer that deforms a triangle mesh to specified target normals; and
- *Geometry in Style*, an identity-preserving shape deformation method from user text input using target normals as a representation of the space of deformations;

We achieve high-quality stylization of shapes through deformation that is faithful to the identity of the input shape with a simple and easy-to-implement framework.

## 2. Related work

### 2.1. Classical Deformation Representations

Our deformation technique builds on the extensive literature on variational surface deformation and mapping methods [8]. Methods in this family commonly compute a local differential deformation quantity, such as a per-element rotation, normal, jacobian matrix, or differential coordinates, subject to modeling constraints such as handles or cages. The vertex positions are then recovered using a global linear

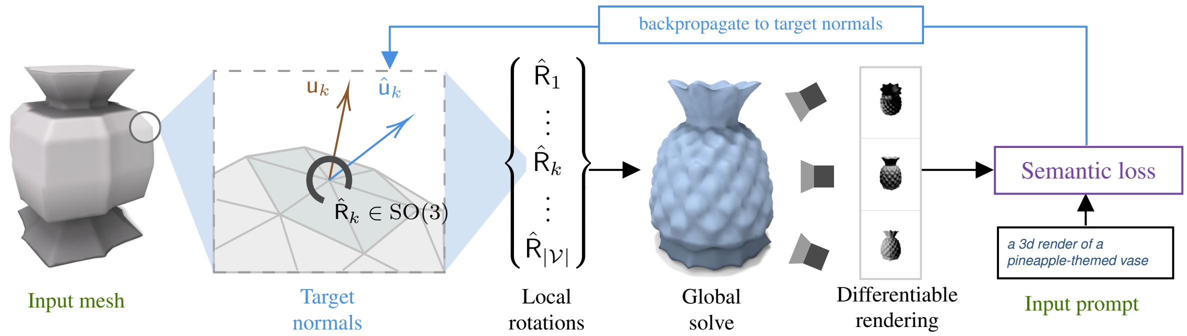



Figure 3. Overview of our stylization pipeline. Geometry in Style optimizes vertex normals to deform the mesh surface, subject to a stylization text prompt. Using the normals undergoing optimization as a target for our differentiable As-Rigid-As-Possible method (dARAP), the dARAP local step computes a rotation matrix per vertex; we then obtain the deformed surface via our dARAP global solve. Then, we utilize a differentiable renderer and a diffusion model-based semantic loss to guide the normals being optimized towards a deformation matching the desired style prompt.

solve, usually derived from the least-squares minimization of energy that encodes the desired local properties.

Important classical deformation approaches include Laplacian surface editing [84], gradient field mesh editing [96], or skinning-based approaches [24, 37]. A seminal work in this field is "as-rigid-as-possible" (ARAP) shape modeling which regularizes the local deformations of a surface to be rigid [83]. This approach promotes local rigidity with smooth and detail-preserving solutions, with various downstream applications including editing, parameterization, registration, shape optimization, and simulation [9, 12, 36, 52, 54, 107]. We adopt ARAP as the basis for our geometric stylization pipeline's deformation method. This is similar to recent neural methods that have also taken advantage of ARAP in other shape representations [6, 10, 35], especially the work of Yan et al. [92] who incorporate a differentiable ARAP loop into an image-to-3D face reconstruction pipeline. Yan et al. [92]'s use of ARAP serves to smoothen an assembly of 3D patches and requires multiple ARAP steps; on the other hand, our dARAP method is meant for optimizations of a deformation quantity (in our case, per-vertex normals) to achieve a desired deformation in a *single* local step-global step pair.

Often related to such differential deformation methods, the manipulation of surface normals is a cornerstone useful for a variety of applications: shape abstraction [3], texture mapping [33, 87, 101], mesh parameterization [102], generative shape refinement [45], and more. Operating on surface normals has also been particularly core to cubic stylization [25, 50, 102] as well as geometric filters [46, 49, 70, 100, 105]. Some approaches use an ARAP-like optimization to achieve desired target normals (similar to our goal) for manufacturing [29, 85].

**Normal-Driven Shape stylization.** Liu and Jacobson [51] propose a normal-based stylization approach by shape analogies. Given a source object and a sphere-based normal template, modeled as a normal-to-normal function $S^2 \to S^2$, the source shape's normals are locally rotated to match target normals dictated by the template; the deformation is obtained via ARAP solve using these local rotations. Similar to this work, we also use target normal vectors as the driving tool for our deformation. However, we use a text prompt to describe the desired style rather than a geometric exemplar, enabling semantic styles (*e.g.* "antique") that are not easily represented by a spherical normal template. Not being tied to a normal template, our deformations are part-aware, *i.e.* different parts with the same source normal do not have to receive the same target normal, and can be stylized differently as can be seen in Figs. 1 and 2).

## 2.2. Neural Shape Manipulation

Following the success of generative methods that optimize 2D representations via text-to-image guidance from diffusion models [31, 40, 74, 80–82] or CLIP-based scores [72, 73], there has been a large body of work using score distillation-based approaches [69, 90] to achieve 3D generation using 2D diffusion priors. These methods use a variety of shape representations, mostly implicits: Signed Distance Fields (SDF) [19, 64, 66], other implicit neural fields [4, 11, 16, 27, 59, 79] or variations on Neural Radiance Fields (NeRF) and Gaussian splatting [5, 14, 17, 18, 44, 47, 48, 53, 55, 56, 62, 71, 73, 76, 78, 89, 91, 93, 99, 103, 104, 106]. Some methods leverage the rapidly growing size of 3D datasets to train text-to-3D models that directly generate 3D representations [39, 63, 98].

More relevant to our present work, recent methods have used the strong representation power of neural networks to drive not just generation but also the editing and manipulation of shapes [2, 30, 60]. Hertz et al. [30]'s network predicts local vertex displacements to match local geometric characteristics of an exemplar shape. The follow-up Text2Mesh [60] replaced reference shapes by style text prompts. Similarly, our shape editing is flexibly guided by text, but rather than *raw vertex displacements*, we find rotations of the surface normal and recover an identity-preserving deformation via the dARAP solver.

Other neural deformation and manipulation methods based include data-driven cage deformations [94], geometric fields for skinning [22], vector displacement maps [58], or deformation fields using neural representations [23, 28, 57, 88]. There is also a large family of 3D editing methods built on implicit shape representations such as NeRF, Gaussian splatting, occupancy fields, and signed distance fields [7, 13, 15, 43, 61, 65, 67, 75, 97].

**Neural Jacobian Fields.** Neural Jacobian Fields (NJF) [2] pioneered in connecting classical differential deformation methods such as Yu et al. [96] to neural, differentiable pipelines with geometric or semantic losses. This approach has powered compelling results for applications including UV mapping, re-posing, handle-based deformation [95], and unsupervised shape morphing and correspondence [86].

In the NJF approach, given a local transformation matrix $\mathsf{M}_k$ per face $k$, the least-squares best fit deformed vertices $\Phi^*$ to these differential transforms can be found by solving a Poisson equation with the cotangent Laplacian:

$$\Phi^* = \operatorname*{argmin}_{\Phi} \sum_{k \in \text{all faces}} a_k \|\Phi \nabla_k^\top - \mathsf{M}_k\|_2^2 = L^{-1} \mathcal{A} \nabla^\top \mathsf{M} \quad (1)$$

where $\mathsf{M}$ is all $\mathsf{M}_k$ stacked, $a_k$ is the area of face $k$, $\nabla$ is the gradient operator, and $\mathcal{A}$ is the face mass matrix. This solve is differentiable with respect to $\mathsf{M}$. We note that the global step to optimize ARAP energy [83] is also a Poisson equation with the cotangent Laplacian (Eq. (6)). As such, in dARAP, we can use a similar differentiable solver while taking advantage of the regularization inherent in ARAP.

Neural methods that use NJF's differential deformation for text-based deformation include TextDeformer [26] and MeshUp [41] which optimize jacobians to deform a source shape into a different semantic target *e.g.* turning a dog into a frog via a text prompt. Their deformations are not sufficiently restricted by construction and require an extra L2 loss between identity and the estimated jacobians to prevent losing the shape identity altogether. In contrast, our deformation framework is more contained by construction (Sec. 3.1), preserving the source details and updating the geometry to the desired style (see Fig. 9 and Sec. 4.2).

# 3. Method

Our method takes as input a source triangle mesh $\mathcal{M} = (\mathcal{V}, \mathcal{F})$ and a text prompt $\mathbf{x}$. Our goal is to obtain a deformed mesh $\mathcal{M}^* = (\mathcal{V}^*, \mathcal{F})$ that semantically matches the style indicated by $\mathbf{x}$.

We find this deformation by optimizing per-vertex unit normals of $\mathcal{M}$. For a mesh with $|\mathcal{V}|$ vertices, gradient descent directly optimizes a $|\mathcal{V}| \times 3$ array of real numbers, *i.e.*, a 3-element vector per vertex. These per-vertex vectors $\hat{\mathcal{U}}$ are treated as *target normals* used to solve for per-vertex rotations $\hat{\mathcal{R}}$, and then, the deformed positions $\hat{\mathcal{V}}$. Specifically, from a current estimate $\hat{\mathcal{U}} = \{\hat{\mathsf{u}}_k \mid k \in \{1 \dots |\mathcal{V}|\}\}$ of target normals, we first perform a *local step* that obtains per-vertex rotation matrices $\hat{\mathcal{R}}_k = \{\hat{\mathsf{R}}_k \in \mathrm{SO}(3) \mid k \in \{1 \dots |\mathcal{V}|\}\}$ from the normals $\hat{\mathcal{U}}$ (Sec. 3.1), followed by a *global step* that obtains the deformed vertex locations $\hat{\mathcal{V}}$ from the per-vertex rotations $\hat{\mathcal{R}}$ (Sec. 3.2).

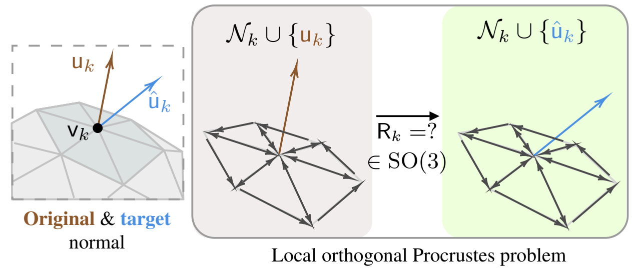



Figure 4. **Local Orthogonal Procrustes**. The single-iteration local step of our dARAP energy solves for the best fit rotation given the original and and target normal.

We refer to this pair of local step and global step as *dARAP*. dARAP is closely inspired by the multiple alternating local-global iterations of classical ARAP optimizations (which are repeated many times until convergence) [83], but here condensed into *a single local step and single global step* as a differentiable module, usable in a neural optimization or deep learning pipeline. While for classical deformation applications, ARAP is run until convergence to achieve satisfactory results, in the context of our pipeline (Fig. 3), dARAP running only one iteration is sufficient and offers the benefits of efficient differentiability.

## 3.1. Local Rotations from Normals

For a vertex $k$ with edge neighborhood $\mathcal{N}_k$, with current estimated target vector $\hat{\mathsf{u}}_k$ (normalized to unit length) and original normal vector $\mathsf{u}_k$ (the area-weighted unit normal of vertex $k$ of the undeformed mesh), we compute a best fit rotation that transforms the bundle of vectors $\mathcal{N}_k \cup \{\mathsf{u}_k\}$ to the bundle $\mathcal{N}_k \cup \{\hat{\mathsf{u}}_k\}$ (see Fig. 4). The best fit rotation $\hat{\mathsf{R}}_k$ minimizes the ARAP energy assuming fixed vertices $\hat{\mathsf{v}}_k$, *i.e.*

$$\begin{aligned} \hat{\mathsf{R}}_k = \operatorname*{argmin}_{\mathsf{R}_k} & \sum_{(i,j) \in \mathcal{N}_k} w_{ij} \|\mathsf{R}_k \mathsf{e}_{ij} - \mathsf{e}_{ij}\|_2^2 \\ & + \lambda a_k \|\mathsf{R}_k \mathsf{u}_k - \hat{\mathsf{u}}_k\| \end{aligned} \quad (2)$$

where $a_k$ is the Voronoi mass of vertex $k$; $\lambda$ is a hyperparameter that scales the strength of the rotation matching the source to the target normal; $\mathsf{e}_{ij} \in \mathcal{N}_k$ are all the edge vectors in the neighborhood of vertex $k$, and $w_{ij}$ are the cotangent weights [68] of these edges. Like Liu and Jacobson [51], we choose the spokes-and-rims neighbor-

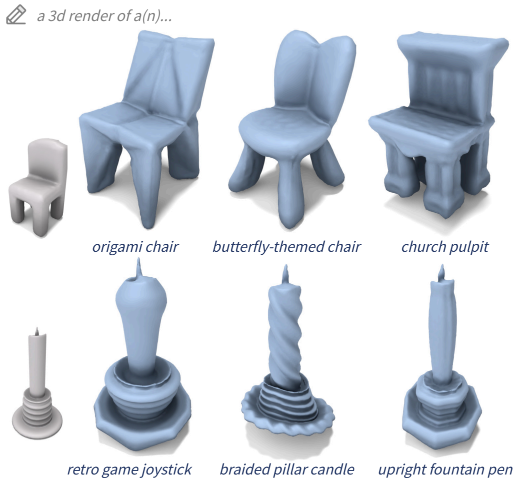



Figure 5. Our method is capable of deforming the same mesh towards different text-specified styles.

hood, consisting of halfedges in the vertex 1-ring, their twin halfedges, and halfedges opposite the vertex [12].

This minimization is the *orthogonal Procrustes problem* for each neighborhood, and can be solved [51] by finding

$$\mathsf{X}_k = \begin{bmatrix} \mathsf{E}_k & \mathsf{u}_k \end{bmatrix} \begin{bmatrix} \mathsf{W}_k & \\ & \lambda a_k \end{bmatrix} \begin{bmatrix} \mathsf{E}_k^\top \\ \hat{\mathsf{u}}_k^\top \end{bmatrix} \tag{3}$$

where $\mathsf{E}_k$ is a $3 \times |\mathcal{N}_k|$ matrix whose columns are the undeformed edge vectors in $\mathcal{N}_k$, and $\mathsf{W}_k$ is a $|\mathcal{N}_k| \times |\mathcal{N}_k|$ diagonal matrix with the cotangent weights of the $\mathcal{N}_k$ edges as the entries. (The $|\mathcal{N}_k|$ dimensions in these matrices can be zero-padded to $\max_{k \in \{1 \ldots |\mathcal{V}|\}} |\mathcal{N}_k|$ for batched solutions.) Taking the SVD of $\mathsf{X}_k$, we can find $\hat{\mathsf{R}}_k$ (up to multiplying the last column of $\mathbf{U}_k$ by $-1$ to ensure $\det(\mathsf{R}_k) > 0$) as

$$\begin{aligned} \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^\top &= \mathsf{X}_k \\ \hat{\mathsf{R}}_k &= \mathbf{V}_k \mathbf{U}_k^\top \end{aligned} \tag{4}$$

Note that in classical ARAP iterative optimization, the term in (2) is normally $\sum_{\mathsf{e}_{ij} \in \mathcal{N}_k} \left( w_{ij} \| \mathsf{R}_k \mathsf{e}_{ij} - \mathsf{e}'_{ij} \|_2^2 \right)$ where $\mathsf{e}'_{ij}$ is the vector of the most recent deformed edge $(i, j)$ in $\mathcal{N}_k$ output by the previous ARAP optimization iteration. Since dARAP is meant as a differentiable module in a larger optimization process or learning pipeline, we condense the typically many local-global alternating steps of classical ARAP to just *one local step and one global step*, hence the identification of $\mathsf{e}'_{ij}$ with $\mathsf{e}_{ij}$. Coupled with setting $\lambda$ to an appropriately large value, which scales the strength of the rotation towards the requested normal $\hat{\mathsf{u}}_k$, our *single* local step is still able to achieve the required expressiveness and strength to make detailed deformations, yet regularized by the Procrustes solve to retain shape identity *without* requiring an extra identity regularization loss as in Kim et al. [41] and Gao et al. [26].

Note also that in NJF-based methods such as [26, 41], the local step would be the identity function; a jacobian matrix per face is assumed given or predicted from upstream components. In our case, a matrix (a rotation) is not given, but computed from the *target normal vector* for each element.

### 3.2. Global Solve from Local Rotations

Having obtained a rotation per neighborhood with our local step, we minimize the energy fixing the rotation matrices and solving for deformed vertex locations, i.e., finding the deformed vertices $\hat{\mathcal{V}}$ such that

$$\hat{\mathcal{V}} = \underset{\tilde{\mathcal{V}}}{\operatorname{argmin}} \sum_{k \in \{1 \ldots |\mathcal{V}|\}} \sum_{(i,j) \in \mathcal{N}_k} w_{ij} \| \mathsf{R}_k \mathsf{e}_{ij} - \tilde{\mathsf{e}}_{ij} \|_2^2 \tag{5}$$

where $\mathsf{e}_{ij} = (\mathsf{v}_j - \mathsf{v}_i)$, $\tilde{\mathsf{e}}_{ij} = (\tilde{\mathsf{v}}_j - \tilde{\mathsf{v}}_i)$, and $w_{ij}$ is the cotangent weight of edge $(i, j)$. This is a linear least squares optimization for $\hat{\mathcal{V}}$. As such, for the spokes-and-rims neighborhood, taking the derivative with respect to $\hat{\mathcal{V}}$ and setting it to zero yields a linear equation in $\hat{\mathcal{V}}$

$$L\hat{\mathcal{V}} = \begin{bmatrix} \mathrm{rhs}(1)^\top \\ \vdots \\ \mathrm{rhs}(|\mathcal{V}|)^\top \end{bmatrix} \tag{6}$$

$$\mathrm{rhs}(k) = \sum_{(k,m,n) \in \mathcal{N}_k^F} \frac{\mathsf{R}_k + \mathsf{R}_m + \mathsf{R}_n}{3} \left( \frac{w_{km}}{2} \mathsf{e}_{km} + \frac{w_{kn}}{2} \mathsf{e}_{kn} \right) \tag{7}$$

where $\mathcal{N}_k^F$ are the faces adjacent to vertex $k$, each one having vertices $(k, m, n)$ (*i.e.* even permutation such that $k$ is in front); $w_{km}, w_{kn}$ are the (undirected) cotangent weights of edges $(k, m), (k, n)$; and $L$ is the cotangent Laplacian.

Equation (6) is a Poisson equation. As such, we can use the same solving and pre-factorization techniques as in NJF [2] to make our global solve step differentiable and efficient.

### 3.3. Optimization Using a Semantic Visual Loss

**Differentiable renderer and semantic loss.** Our optimization is guided by a powerful pretrained text-to-image (T2I) diffusion model. We render the deformed mesh from multiple views using a differentiable rasterizer [42], then feed the rendered views into a semantic visual loss, in our case Cascaded Score Distillation (CSD) [21] using stages 1 and 2 of the image diffusion model DeepFloyd IF [1].

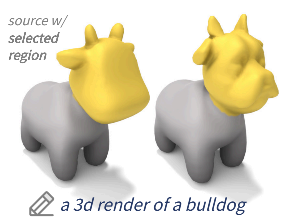
## 4. Experiments

**Optimization details.** Our initial guess for the target normals $\hat{\mathcal{U}}$ is $\mathcal{U}$, the original area-weighted vertex normals of the undeformed mesh $\mathcal{M}$. We set the local step hyperparameter $\lambda = 8$ (Sec. 3.1) for all of our optimizations. We run our optimization for 2500 epochs at a constant learning rate of 0.002 using the Adam optimizer, each epoch being a batch of 8 renders fed to CSD loss. A full optimization run takes about 2 hours 15 minutes using a single A40 GPU. We remesh our source meshes for better behavior with the cotangent Laplacian (see Sec. 4.3.) Further details on view sampling settings, CSD configuration, and source mesh preprocessing can be found in the supplementary material.

### 4.1. Properties of Geometry in Style

**Generality and expressivity.** Our method is highly versatile, and is able to deform meshes from varied domains towards a wide range of styles (Figs. 1 and 2). Our method handles organic and articulated surfaces, such as animals and the human body, as well as man-made objects with sharp features and complex topology such as chairs. The target style is specified by an open-vocabulary text prompt and can thus be described flexibly and intuitively.

The stylization manifests in a part-aware manner, conforming to the shape's geometry. For the *pineapple-themed* vase in Fig. 1, our method adds a pineapple-like *geometric texture* to the vase's body, while the vase's head is deformed with a different ripple pattern to resemble a pineapple head. For a human body, our method creates *geometric details* to reflect a knight's armor in the appropriate locations, such as shoulder pads, a large chest plate, a crease across the waistline, and a hat on the head. In Fig. 2, the *antique*, *gothic*, and *cardboard* chairs' styles are reflected by both local geometric details and the silhouette of the deformed mesh.




We can further take advantage of part awareness to stylize *only* select regions. In the inset, we localize the deformation to the head by setting the rotation of vertices outside the region to the identity matrix every iteration. The deformation is contained within the local region, yet detailed and appropriate for the rest of the body. We observe no boundary artifacts, showing dARAP's beneficial regularizing effects.

**Identity preservation.** Our method applies the prescribed style to the input mesh expressively while preserving important characteristics. In Fig. 7, each input animal has a unique pose, *e.g.* the folded front leg of the horse. The deformation recognizably keeps the pose while stylizing the body towards the *skeletal* style. Similarly, the stylization of the person shape from Fig. 1 maintains body proportions. Additionally, our method preserves other distinct shape properties, *e.g.*, the animal's facial features (Fig. 7) and the chairs' parts (Fig. 2), with semantic correspondences maintained (Fig. 10).

We attribute our pronounced yet identity-preserving stylizations to our normal-based deformation representation. A pretrained vision foundation model provides strong guidance toward the style prompt but can also easily impart de-

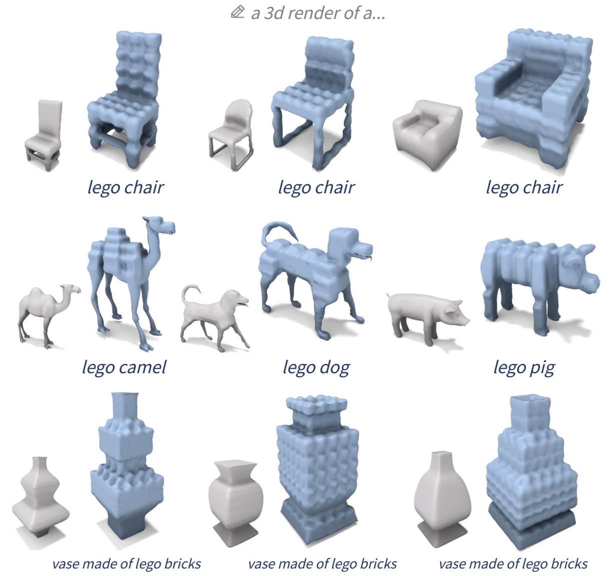



Figure 6. Our method is capable of deforming *different* input meshes towards the same text-specified style. Even with the same prompt in a shape class (*a 3d render of a lego chair*, *a 3d render of a vase made of lego bricks*), different starting shapes (different chairs, different vases) result in stylizations that closely align to the identity of the original shape, while still strong enough to induce more blocky components and *lego*-like surface textures.

formations that significantly alter the identity of the shape, as witnessed in previous work that used jacobian-based deformations [26, 41], see Figs. 9 and 11. By contrast, our deformation is driven by *rotations* of normals, further regularized as a best-fit rotation over the spokes-and-rims neighborhood (Sec. 3.1.) This formulation selects for local rigid changes that are discouraged from scaling or shearing, thus preventing excessive structural changes while requiring no extra identity regularization loss on the local transforms.

**Specificity.** Our method performs diverse shape stylizations that adhere to the target style prompt with high detail. In Fig. 5, we deform the source shape into different styles: the *origami* chair's backrest is thin and has creases as with paper folds; the *church pulpit* style is thicker with overhangs appropriate of church furniture. As further seen by the detailed prompts and styles in Figs. 1 and 2, our method produces distinct styles and shows granular effectiveness.

**Robustness.** Our method exhibits robustness across shape categories and instances within the category. In Fig. 6, the same *lego* style is applied to chairs, animals, and vases. Each domain has unique geometry: the chairs have varying parts (*e.g.*, the types of legs and backrests), the animals have smooth geometry, and the vases have sharp edges and rotational symmetry. Still, our method consistently conveys the style on the source shape with a *lego* brick-like surface pattern and by cubifying the geometry.

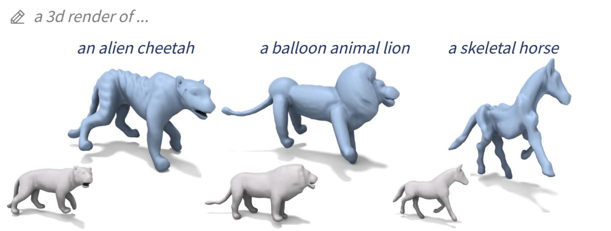



Figure 7. **Pose preservation.** Geometry in Style stylize the shape according to the text prompt, while decently keeping characteristics of the source shape like the relative positioning of the limbs and the angle of the head.

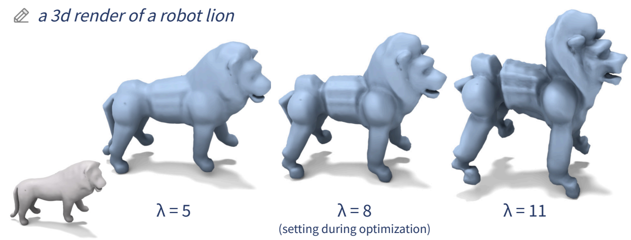



Figure 8. **Changing stylization strength after optimization.** Normals found by optimization using $\lambda = 8$ can be conveniently re-applied after optimization using a different $\lambda$ to tune the stylization strength on demand. Both larger and smaller $\lambda$ result in salient and clean stylizations at the required strength.

**Tunable stylization strength.** A specific value of $\lambda$ is used by deformations during the optimization process. However, this $\lambda$ can be changed when re-applying the saved optimized normals to the source mesh, allowing *user control* separate from the optimization pipeline itself, as shown in Fig. 8. Recall from Eq. (2) that the hyperparameter $\lambda$ influences the match from the original vertex normal to the target normal. As seen in Fig. 8, as the value of $\lambda$ increases, the geometry of the deformed shape is sharpened to strengthen to "robot" style effect, while a lower $\lambda$ results in deformed normals being closer to the original ones, though the desired style is still visible. Notably, we observe that using an inference $\lambda$ value larger than that used during optimization results in a more geometrically salient yet still sensible stylization, further demonstrating the robustness of our method.

### 4.2. Evaluation

We contrast our method to two recent text-guided mesh deformation methods, TextDeformer [26] and MeshUp [41] using public code released by the authors.

**Qualitative comparison.** In Fig. 9, we show deformation results for TextDeformer [26], MeshUp [41], and our method. For comparison, all three methods use the same text prompt, source mesh, and view sampling settings. TextDeformer distorts the surface, changes the pose of the source shape, and does not achieve the target style. MeshUp does stylize following the text prompt, but in some cases, its surface texture may be weaker than ours, as seen in the *lego* goat example.

In other cases, MeshUp's stylization is strong but produces notable distortion in arms and body proportions: in the examples *knight in armor* and *Chinese terracotta warrior*, the body gets a broader, stouter stature and does not preserve the identity and body proportions of the source shape as well as our method does. Moreover, our method does not encounter certain artifacts that MeshUp introduces, like the crushed head of the *knight in armor* example, or the Janus effect with a duplicate face and chestplate on the back for *Chinese terracotta warrior.* We achieve a prominent desired style with fewer artifacts and better preservation of source shape features. More qualitative examples are given in the supplementary material.

**Quantitative results.** As a proxy for evaluating identity preservation, we measure the mean and standard deviation of the ratio (deformed triangle area / original triangle area) summarized over all triangles across the chosen meshes. We use 20 mesh-prompt pairs chosen from results seen throughout the paper; per-mesh triangle counts range from 10 thousand to 20 thousand. Source meshes are normalized to fit a side-length-2 cube centered at the origin; deformed meshes are normalized to have the same bounding box diagonal length as the source, as is done after the Poisson solve of MeshUp and our method. This quantity measures distortion and estimates how well the bounding box is respected: deformations that shrink the mesh lead to a larger rescale factor during the bounding-box-restoring normalization, thus inflating the face area ratio, and vice versa. Ideal identity-preserving values are ratio 1 with 0 standard deviation.

Table 1 summarizes the quantitative comparison with the baseline methods [26, 41]. These deformations use jacobians, which are more prone to changing triangle scale and compromising the integrity of the mesh. In contrast, we represent a target deformation by surface normals, coupled with our dARAP layer that regularizes the resulting deformation and better preserves the original triangle area, improving faithfulness to the input shape. Indeed, as Tab. 1 shows, our triangle area ratio has an average closer to 1 with a lower standard deviation than MeshUp and TextDeformer. We also include a quantitative evaluation of CLIP similarity to the prompt for these three methods on the same shapes (see the supplementary material); our method achieves better CLIP similarity to the prompt.

### 4.3. Limitations

Our method uses the cotangent Laplacian, which works well only on manifold meshes and is sensitive to triangle aspect ratios. This is also true for other methods that use

| Method | Ratio mean | Ratio std. dev. |
|---|---|---|
| TextDeformer [26] | 0.827 | 0.360 |
| MeshUp [41] | 1.288 | 0.363 |
| Geometry in Style (**ours**) | **1.080** | **0.233** |

Table 1. **Triangle area preservation.** As a surrogate for measuring identity preservation, we compute the mean and standard deviation of the triangle area ratio between the deformed and source shapes. Our method preserves the triangle area better than the other methods, with an **average ratio closer to 1** and a lower standard deviation.

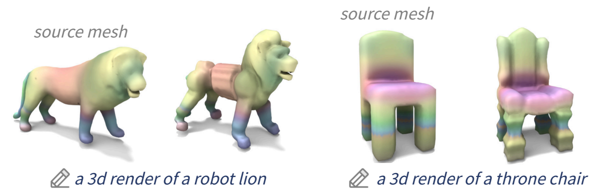



Figure 10. **Correspondence.** The meshes deformed with our method preserve semantic correspondence to the source mesh. Deformed vertices have the same color as the corresponding source vertices, colored by the source shape's wave kernel signature.

the cotangent Laplacian, such as TextDeformer [26] and MeshUp [41]. To mitigate this, we remesh input meshes with isotropic explicit remeshing [32], optionally after manifold preprocessing [34].

Another limitation is the possibility of self-intersection in deformed meshes (Fig. 11). The rods of the source lamp are rotated towards the center and intersect each other. This can be somewhat mitigated by adjusting the deformation strength parameter $\lambda$ after optimization, as discussed in Fig. 8, a strategy not straightforwardly available to MeshUp.

## 5. Conclusion

In this work, we presented Geometry in Style, a technique for deforming meshes to achieve a text-specified style. A key claim of the work is that prescribing a deformation via surface normals allows the recovery of deformed vertices that adhere well to the input geometry while still being expressive. We demonstrate high-quality, detailed geometric stylizations that respect the input shape's identity.

As part of Geometry in Style, we introduce dARAP, a differentiable neural network layer that deforms a surface to achieve target normals. Our layer is simple yet effective: while traditional applications require iterating ARAP to convergence for a desirable solution, we find that within our neural network pipeline, good results can be achieved with *only a single step*. We speculate that by iteratively updating target normals through gradient descent, we can avoid (in a dARAP forward pass) the standard practice of needing to repeatedly iterate between local and global ARAP steps. Moreover, dARAP is general and may be used for other geometry tasks where ARAP is useful, such as parameterization, re-posing, collisions, editing, and more.

In the future, we are interested in leveraging unsupervised segmentation strategies [20, 21] to perform localized geometric stylization. In addition, while our method is topology-preserving, follow-up work could explore edits and deformations that add explicit parts or change topology.

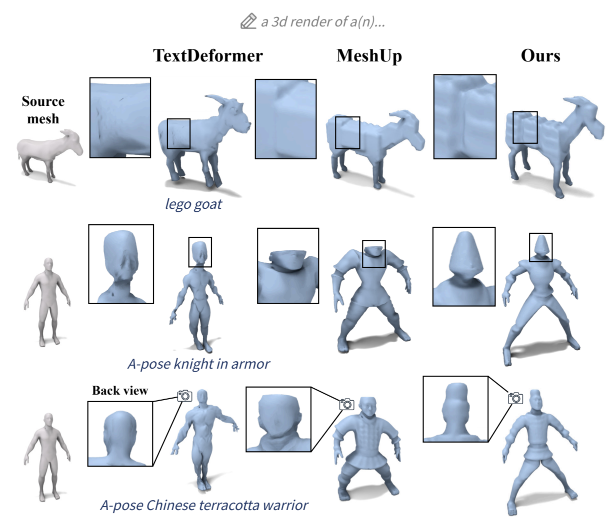



Figure 9. **Comparison with baselines.** We compare our method with the alternative deformation techniques TextDeformer [26] and MeshUp [41]. While the baseline methods have a weaker stylization effect, change the poses, or create geometric artifacts on some examples, Geometry in Style cleanly achieves both the desired style and remains faithful to the shape of the source meshes.

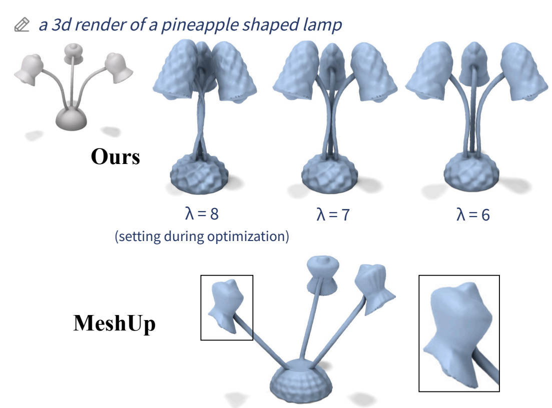



Figure 11. **Limitations.** Our method may produce self-intersections. Decreasing the parameter $\lambda$ *after* the optimization process can alleviate the self-intersection with only a mild reduction in the stylistic surface details. MeshUp warps the 3-lamp structure and the individual lamps and exhibits less geometric pineapple texture.

## 6. Acknowledgments

We would like to thank Justin Solomon, Ana Dodik, Haochen Wang, Richard Liu, and members of the 3DL lab for their helpful comments, suggestions, and insightful discussions. We are grateful for the AI cluster resources, services, and staff expertise at the University of Chicago. This work was supported by BSF grant 2022363, NSF grants 2304481, 2241303, 2402894, and 2335493, and gifts from Snap Research, Adobe Research, and Google Research.


## References


[1] Stability AI. Deepfloydif, 2023. 5

[2] Noam Aigerman, Kunal Gupta, Vladimir G. Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. Neural jacobian fields: learning intrinsic mappings of arbitrary meshes. *ACM Trans. Graph.*, 41(4), 2022. 1, 3, 4, 5

[3] Marc Alexa. Polycover: Shape approximating with discrete surface orientation. *IEEE Computer Graphics and Applications*, 41(3):85–95, 2021. 3

[4] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data, 2020. 3

[5] Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B. Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling, 2024. 3

[6] Daniele Baieri, Filippo Maggioli, Zorah Lähner, Simone Melzi, and Emanuele Rodolà. Implicit-arap: Efficient handle-guided deformation of high-resolution meshes and neural fields via local patch meshing. *arXiv preprint arXiv:2405.12895*, 2024. 3

[7] Amir Barda, Vladimir G. Kim, Noam Aigerman, Amit H. Bermano, and Thibault Groueix. Magicclay: Sculpting meshes with generative neural fields. *SIGGRAPH Asia (Conference track)*, 2024. 4

[8] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, 2008. 2

[9] Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. Shape-up: Shaping discrete geometry with projections. In *Computer Graphics Forum*, pages 1657–1667. Wiley Online Library, 2012. 3

[10] Aljaz Bozic, Pablo Palafox, Michael Zollhöfer, Angela Dai, Justus Thies, and Matthias Nießner. Neural non-rigid tracking. *Advances in Neural Information Processing Systems*, 33:18727–18737, 2020. 3

[11] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction, 2020. 3

[12] Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. A simple geometric model for elastic deformations. *ACM Trans. Graph.*, 29(4), 2010. 3, 5

[13] Hansheng Chen, Ruoxi Shi, Yulin Liu, Bokui Shen, Jiayuan Gu, Gordon Wetzstein, Hao Su, and Leonidas Guibas. Generic 3d diffusion adapter using controlled multi-view editing, 2024. 4

[14] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation, 2023. 3

[15] Yun-Chun Chen, Selena Ling, Zhiqin Chen, Vladimir G. Kim, Matheus Gadelha, and Alec Jacobson. Text-guided controllable mesh refinement for interactive 3d modeling. *SIGGRAPH Asia (Conference track)*, 2024. 4

[16] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling, 2019. 3

[17] Zilong Chen, Feng Wang, and Huaping Liu. Text-to-3d using gaussian splatting, 2023. 3

[18] Zilong Chen, Feng Wang, Yikai Wang, and Huaping Liu. Text-to-3d using gaussian splatting, 2024. 3

[19] Julian Chibane, Aymen Mir, and Gerard Pons-Moll. Neural unsigned distance fields for implicit function learning, 2020. 3

[20] Dale Decatur, Itai Lang, and Rana Hanocka. 3d highlighter: Localizing regions on 3d shapes via text descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20930–20939, 2023. 8

[21] Dale Decatur, Itai Lang, Kfir Aberman, and Rana Hanocka. 3d paintbrush: Local stylization of 3d shapes with cascaded score distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4473–4483, 2024. 2, 5, 8

[22] Ana Dodik, Vincent Sitzmann, Justin Solomon, and Oded Stein. Robust biharmonic skinning using geometric fields. *arXiv preprint arXiv:2406.00238*, 2024. 4

[23] Marvin Eisenberger, David Novotny, Gael Kerchenbaum, Patrick Labatut, Natalia Neverova, Daniel Cremers, and Andrea Vedaldi. Neuromorph: Unsupervised shape interpolation and correspondence in one go, 2021. 4

[24] Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. Latent-space dynamics for reduced deformable simulation. *Computer Graphics Forum*, 2019. 3

[25] Marco Fumero, Michael Möller, and Emanuele Rodolà. Nonlinear spectral geometry processing via the tv transform. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020. 3

[26] William Gao, Noam Aigerman, Thibault Groueix, Vova Kim, and Rana Hanocka. Textdeformer: Geometry manipulation using text guidance. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. 1, 4, 5, 6, 7, 8

[27] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes, 2020. 3

[28] Rana Hanocka, Noa Fish, Zhenhua Wang, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. Alignet: Partial-shape agnostic alignment via unsupervised learning. *ACM Trans. Graph.*, 2018. 4

[29] Philipp Herholz, Wojciech Matusik, and Marc Alexa. Approximating free-form geometry with height fields for manufacturing. *Computer Graphics Forum*, 34(2):239–251, 2015. 3

[30] Amir Hertz, Rana Hanocka, Raja Giryes, and Daniel Cohen-Or. Deep geometric texture synthesis. *ACM Trans. Graph.*, 39(4), 2020. 1, 3

[31] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020. 3

[32] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Mesh optimization. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, page 19–26, New York, NY, USA, 1993. Association for Computing Machinery. 8

[33] Jin Huang, Tengfei Jiang, Zeyun Shi, Yiying Tong, Hujun Bao, and Mathieu Desbrun. l1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics (TOG)*, 33(3):1–11, 2014. 3

[34] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018. 8

[35] Qixing Huang, Xiangru Huang, Bo Sun, Zaiwei Zhang, Junfeng Jiang, and Chandrajit Bajaj. Arapreg: An as-rigid-as possible regularization loss for learning deformable shape generators. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5815–5825, 2021. 3

[36] Takeo Igarashi, Tomer Moscovich, and John F Hughes. As-rigid-as-possible shape manipulation. *ACM transactions on Graphics (TOG)*, 24(3):1134–1141, 2005. 3

[37] Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*, 2014. 3

[38] Tomas Jakab, Richard Tucker, Ameesh Makadia, Jiajun Wu, Noah Snavely, and Angjoo Kanazawa. Keypointdeformer: Unsupervised 3d keypoint discovery for shape control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 1

[39] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions, 2023. 3

[40] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models, 2022. 3

[41] Hyunwoo Kim, Itai Lang, Noam Aigerman, Thibault Groueix, Vladimir G Kim, and Rana Hanocka. Meshup: Multi-target mesh deformation via blended score distillation. *arXiv preprint arXiv:2408.14899*, 2024. 1, 4, 5, 6, 7, 8

[42] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 5

[43] Verica Lazova, Vladimir Guzov, Kyle Olszewski, Sergey Tulyakov, and Gerard Pons-Moll. Control-nerf: Editable feature volumes for scene rendering and manipulation, 2022. 4

[44] Weiyu Li, Rui Chen, Xuelin Chen, and Ping Tan. Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d, 2023. 3

[45] Weiyu Li, Jiarui Liu, Rui Chen, Yixun Liang, Xuelin Chen, Ping Tan, and Xiaoxiao Long. Craftsman: High-fidelity mesh generation with 3d native generation and interactive geometry refiner, 2024. 3

[46] Zhiqi Li, Yingkui Zhang, Yidan Feng, Xingyu Xie, Qiong Wang, Mingqiang Wei, and Pheng-Ann Heng. Normalf-net: Normal filtering neural network for feature-preserving mesh denoising. *Computer-Aided Design*, 127:102861, 2020. 3

[47] Yixun Liang, Xin Yang, Jiantao Lin, Haodong Li, Xiaogang Xu, and Yingcong Chen. Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching, 2023. 3

[48] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation, 2023. 3

[49] Bin Liu, Junjie Cao, Weiming Wang, Ning Ma, Bo Li, Ligang Liu, and Xiuping Liu. Propagated mesh normal filtering. *Computers & Graphics*, 74:119–125, 2018. 3

[50] Hsueh-Ti Derek Liu and Alec Jacobson. Cubic stylization. *arXiv preprint arXiv:1910.02926*, 2019. 3

[51] Hsueh-Ti Derek Liu and Alec Jacobson. Normal-driven spherical shape analogies. In *Computer Graphics Forum*, pages 45–55. Wiley Online Library, 2021. 3, 4, 5

[52] Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. A local/global approach to mesh parameterization. In *Proceedings of the Symposium on Geometry Processing*, page 1495–1504, Goslar, DEU, 2008. Eurographics Association. 3

[53] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object, 2023. 3

[54] Tiantian Liu, Adam W Bargteil, James F O'Brien, and Ladislav Kavan. Fast simulation of mass-spring systems. *ACM Transactions on Graphics (TOG)*, 32(6):1–7, 2013. 3

[55] Artem Lukoianov, Haitz Sáez de Ocáriz Borde, Kristjan Greenewald, Vitor Campagnolo Guizilini, Timur Bagautdinov, Vincent Sitzmann, and Justin Solomon. Score distillation via reparametrized ddim, 2024. 3

[56] Baorui Ma, Haoge Deng, Junsheng Zhou, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Geodream: Disentangling 2d and geometric priors for high-fidelity and consistent 3d generation, 2023. 3

[57] Arman Maesumi, Paul Guerrero, Noam Aigerman, Vladimir G. Kim, Matthew Fisher, Siddhartha Chaudhuri, and Daniel Ritchie. Explorable mesh deformation subspaces from unstructured 3d generative models. *SIGGRAPH Asia (Conference track)*, 2023. 4

[58] Hengyu Meng, Duotun Wang, Zhijing Shao, Ligang Liu, and Zeyu Wang. Text2vdm: Text to vector displacement maps for expressive and interactive 3d sculpting, 2025. 4

[59] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space, 2019. 3

[60] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13492–13502, 2022. 1, 3

[61] Aryan Mikaeili, Or Perel, Mehdi Safaee, Daniel Cohen-Or, and Ali Mahdavi-Amiri. Sked: Sketch-guided text-based 3d editing, 2023. 4

[62] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020. 3

[63] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts, 2022. 3

[64] Helen Oleynikova, Alexander Millane, Zachary Taylor, Enric Galceran, Juan I. Nieto, and Roland Y. Siegwart. Signed distance fields: A natural representation for both mapping and planning. In *RSS Workshop: Geometry and Beyond - Representations, Physics, and Scene Understanding for Robotics*, 2016. 3

[65] JangHo Park, Gihyun Kwon, and Jong Chul Ye. ED-neRF: Efficient text-guided editing of 3d scene with latent space neRF. In *The Twelfth International Conference on Learning Representations*, 2024. 4

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

[67] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5865–5874, 2021. 4

[68] Ulrich Pinkall and Konrad Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics*, 2(1):15–36, 1993. 4

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

[70] Fabian Prada and Misha Kazhdan. Unconditionally stable shock filters for image and geometry processing. In *Computer Graphics Forum*, pages 201–210. Wiley Online Library, 2015. 3

[71] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, and Bernard Ghanem. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors, 2023. 3

[72] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 3

[73] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. 3

[74] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 3

[75] Benet Oriol Sabat, Alessandro Achille, Matthew Trager, and Stefano Soatto. Nerf-insert: 3d local editing with multimodal control signals, 2024. 4

[76] Junyoung Seo, Wooseok Jang, Min-Seop Kwak, Jaehoon Ko, Hyeonsu Kim, Junho Kim, Jin-Hwa Kim, Jiyoung Lee, and Seungryong Kim. Let 2d diffusion model know 3d-consistency for robust text-to-3d generation, 2023. 3

[77] Meitar Shechter, Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. NeuralMLS: Geometry-Aware Control Point Deformation. In *Eurographics 2022 - Short Papers*. The Eurographics Association, 2022. 1

[78] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation, 2024. 3

[79] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions, 2020. 3

[80] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models, 2022. 3

[81] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models, 2021.

[82] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021. 3

[83] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, page 109–116, Goslar, DEU, 2007. Eurographics Association. 3, 4

[84] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, page 175–184, New York, NY, USA, 2004. Association for Computing Machinery. 3

[85] Oded Stein, Alec Jacobson, and Eitan Grinspun. Interactive design of castable shapes using two-piece rigid molds. *Computers & Graphics*, 80:51–62, 2019. 3

[86] Ramana Sundararaman, Nicolas Donati, Simone Melzi, Etienne Corman, and Maks Ovsjanikov. Deformation recovery: Localized learning for detail-preserving deformations. *arXiv preprint arXiv:2410.08225*, 2024. 4

[87] Marco Tarini, Kai Hormann, Paolo Cignoni, and Claudio Montani. Polycube-maps. *ACM transactions on graphics (TOG)*, 23(3):853–860, 2004. 3

[88] Mikaela Angelina Uy, Vladimir G. Kim, Minhyuk Sung, Noam Aigerman, Siddhartha Chaudhuri, and Leonidas Guibas. Joint learning of 3d shape retrieval and deformation. *CVPR*, 2021. 4

[89] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields, 2022. 3

[90] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A. Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation, 2022. 3

[91] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *arXiv preprint arXiv:2305.16213*, 2023. 3

[92] Peizhi Yan, James Gregson, Qiang Tang, Rabab Ward, Zhan Xu, and Shan Du. Neo-3df: Novel editing-oriented 3d face creation and reconstruction. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 486–502, 2022. 3

[93] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models, 2024. 3

[94] Wang Yifan, Noam Aigerman, Vladimir G Kim, Siddhartha Chaudhuri, and Olga Sorkine-Hornung. Neural cages for detail-preserving 3d deformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 75–83, 2020. 4

[95] Seungwoo Yoo, Kunho Kim, Vladimir G Kim, and Minhyuk Sung. As-plausible-as-possible: Plausibility-aware mesh deformation using 2d diffusion priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4315–4324, 2024. 4

[96] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with poisson-based gradient field manipulation. In *ACM SIGGRAPH 2004 Papers*, page 644–651, New York, NY, USA, 2004. Association for Computing Machinery. 3, 4

[97] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: Geometry editing of neural radiance fields, 2022. 4

[98] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models, 2023. 3

[99] Jingbo Zhang, Xiaoyu Li, Ziyu Wan, Can Wang, and Jing Liao. Text2nerf: Text-driven 3d scene generation with neural radiance fields, 2023. 3

[100] Wangyu Zhang, Bailin Deng, Juyong Zhang, Sofien Bouaziz, and Ligang Liu. Guided mesh normal filtering. In *Computer Graphics Forum*, pages 23–34. Wiley Online Library, 2015. 3

[101] Hui Zhao, Na Lei, Xuan Li, Peng Zeng, Ke Xu, and Xianfeng Gu. Robust edge-preserving surface mesh polycube deformation. *Computational Visual Media*, 4:33–42, 2018. 3

[102] Hui Zhao, Kehua Su, Chenchen Li, Boyu Zhang, Lei Yang, Na Lei, Xiaoling Wang, Steven J Gortler, and Xianfeng Gu. Mesh parametrization driven by unit normal flow. In *Computer Graphics Forum*, pages 34–49. Wiley Online Library, 2020. 3

[103] Minda Zhao, Chaoyi Zhao, Xinyue Liang, Lincheng Li, Zeng Zhao, Zhipeng Hu, Changjie Fan, and Xin Yu. Efficientdreamer: High-fidelity and robust 3d creation via orthogonal-view diffusion prior, 2023. 3

[104] Yuyang Zhao, Zhiwen Yan, Enze Xie, Lanqing Hong, Zhenguo Li, and Gim Hee Lee. Animate124: Animating one image to 4d dynamic scene, 2024. 3

[105] Youyi Zheng, Hongbo Fu, Oscar Kin-Chung Au, and Chiew-Lan Tai. Bilateral normal filtering for mesh denoising. *IEEE transactions on visualization and computer graphics*, 17(10):1521–1530, 2010. 3

[106] Junzhe Zhu, Peiye Zhuang, and Sanmi Koyejo. Hifa: High-fidelity text-to-3d generation with advanced diffusion guidance, 2024. 3

[107] Michael Zollhöfer, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (ToG)*, 33 (4):1–12, 2014. 3