TextDeformer: Geometry Manipulation using Text Guidance

William Gao
University of Chicago
USA
wmg@uchicago.edu

Noam Aigerman
Adobe Research
USA
aigerman@adobe.com

Thibault Groueix
Adobe Research
USA
groueix@adobe.com

Vladimir G. Kim
Adobe Research
USA
vokim@adobe.com

Rana Hanocka
University of Chicago
USA
ranahanocka@uchicago.edu

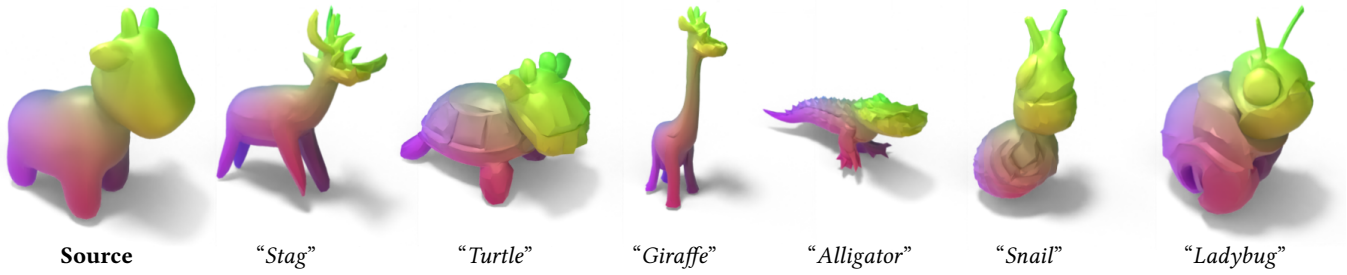

Figure 1: TextDeformer deforms a source shape into various text-specified targets. The mesh colors visualize the smoothness of the mappings.

ABSTRACT

We present a technique for automatically producing a deformation of an input triangle mesh, guided solely by a text prompt. Our framework is capable of deformations that produce both large, low-frequency shape changes, and small high-frequency details. Our framework relies on differentiable rendering to connect geometry to powerful pre-trained image encoders, such as CLIP and DINO. Notably, updating mesh geometry by taking gradient steps through differentiable rendering is notoriously challenging, commonly resulting in deformed meshes with significant artifacts. These difficulties are amplified by noisy and inconsistent gradients from CLIP. To overcome this limitation, we opt to represent our mesh deformation through Jacobians, which updates deformations in a global, smooth manner (rather than locally-sub-optimal steps). Our key observation is that Jacobians are a representation that favors smoother, large deformations, leading to a global relation between vertices and pixels, and avoiding localized noisy gradients. Additionally, to ensure the resulting shape is coherent from all 3D viewpoints, we encourage the deep features computed on the 2D encoding of the rendering to be consistent for a given vertex from all viewpoints. We demonstrate that our method is capable of smoothly-deforming a wide variety of source mesh and target text prompts, achieving both large modifications to, e.g., body proportions of animals, as well as adding fine semantic details, such as shoe laces on an army boot and fine details of a face.

SIGGRAPH '23 Conference Proceedings, August 06–10, 2023, Los Angeles, CA, USA

ACM ISBN 979-8-4007-0159-7/23/08…$15.00
https://doi.org/10.1145/3588432.3591552

CCS CONCEPTS

• **Computing methodologies** → **Mesh geometry models**; **Neural networks**;

KEYWORDS

CLIP, text guidance, mesh, deformation

ACM Reference Format:
William Gao, Noam Aigerman, Thibault Groueix, Vladimir G. Kim, and Rana Hanocka. 2023. TextDeformer: Geometry Manipulation using Text Guidance. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 06–10, 2023, Los Angeles, CA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3588432.3591552

1 INTRODUCTION

This paper proposes a method to deform 3D meshes into other shapes through text guidance. Deforming meshes is a highly researched problem in computer graphics and geometry processing, with applications in content creation [Gal et al. 2009], character posing [Jacobson 2013], and morphing [Kraevoy and Sheffer 2004]. Most existing techniques provide a user with the ability to control a deformation through control handles [Jakab et al. 2020; Shechter et al. 2022] which expose a space of coarse, low-frequency deformations. Such deformations are often referred to as detail-preserving.

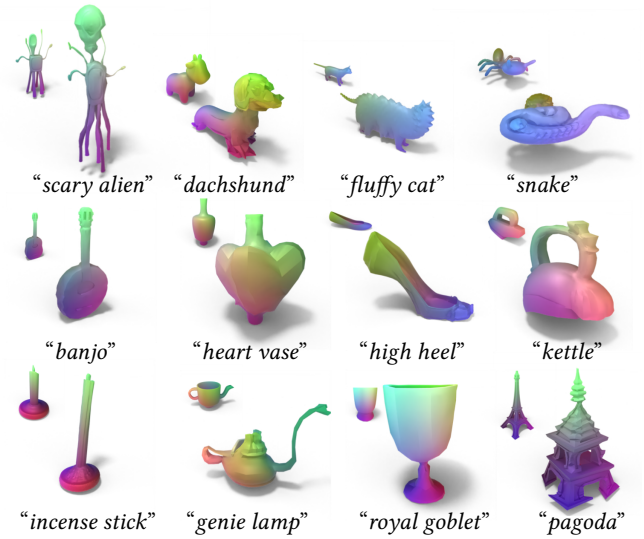

Figure 2: Gallery of results. Source meshes shown in inset and target text shown below.

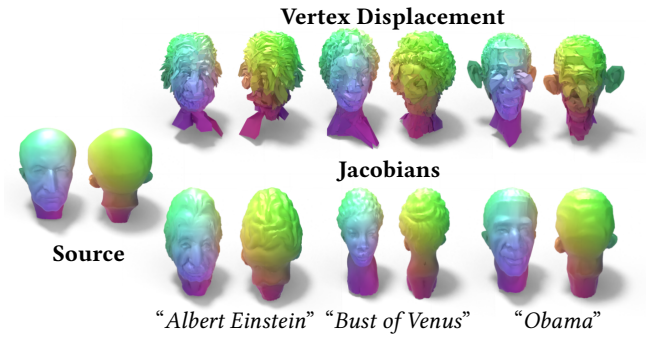

Figure 3: Globally-Coherent Deformations. Max Planck deforms into different targets using our method (bottom row), where the front/back view is shown in pairs of left/right. Removing Jacobians and predicting displacements (top row) takes locally sub-optimal steps, which results in distorted shapes with significant artifacts. Jacobians produce global deformations, resulting in cleaner geometries and even prevents the spurious face mirroring (for example, in "*Obama*").

However, 3D modeling often also requires incorporating geometric details, where an artist needs to meticulously add details in a laborious process. In this work, we aim to automate the entire deformation process, in order to automatically deform the mesh from its initial shape into the desired target shape, while preserving semantic correspondence between the source and the final shape. To achieve this goal, we follow the recent success of text-guided generative methods for images [Ramesh et al. 2022; Patashnik et al. 2021], meshes [Michel et al. 2021; Khalid et al. 2022], and NeRFs [Poole et al. 2022], by leveraging language as an intuitive tool for deforming shapes. Similarly to these previous works, our formulation does not rely on 3D training data, but instead leverages differentiable rendering to connect powerful pre-trained image encoders (such as CLIP [Radford et al. 2021]) to provide a signal for modifying the geometry. After the deformation process, the resultant geometry respects the structure and characteristics of the source mesh, while visually adhering to the text specifications.

In contrast to previous text-guided works which aim to either *hallucinate* geometry from scratch [Jain et al. 2021; Poole et al. 2022; Wang et al. 2022] or preserve the geometry of an input mesh [Michel et al. 2021] while adding detail, we instead focus on the *shape deformation task*.

Our framework manipulates an existing input shape, to enable producing high-quality geometry from the source mesh. Moreover,as can be seen in Fig. 1, our framework is capable of producing both low-frequency shape changes and high-frequency details (e.g. the cow's neck is elongated when deforming to a giraffe) and incorporate details (scales are added when deforming to an alligator). The resulting correspondences from source shape to target are continuous and semantically meaningful ("leg deforms to leg"), which we visualize by coloring the source mesh (e.g. in Fig. 1 and throughout). This property is especially critical for shape-morphing applications.

Thus, our framework is must satisfy several properties: 1) produce high-quality surface geometry, with minimal self-intersections and noisy normals; 2) produce plausible results which match the text description; 3) adhere to the input geometry (e.g., deform the source's head into the target's head and not into body). This leads to several challenges which we solve through our technical contributions.

First, straightforward optimization of mesh vertices through differentiable rendering, as previous text-to-3D methods displayed, often converges to undesirable local minima, and gradient steps often turn parts of the mesh inside-out, introducing significant artifacts. The crux of the difficulty lies in that the back-propagated gradient from CLIP is noisy, with many undesirable local minima and arbitrary directions. Thus, instead of displacing vertices, we take inspiration from Neural Jacobian Fields [Aigerman et al. 2022] to devise a more robust representation of deformations. We optimize matrices representing the deformation's gradients, i.e., the Jacobians of each of the triangles, and compute the deformed vertex positions from them, by solving Poisson's equation. Our key observation is that representing the deformation through Jacobians in this way leads to a representation that favors smoother, large deformations, and leads to a global relation between vertices and pixels, thus avoiding localized noisy gradients. More precisely: i) smooth Jacobians represent low-frequency, large-scale deformations, and ii) Poisson's equation leads to each Jacobian affecting all vertices, and in turn all pixels of the rendering. Thus, when CLIP's gradients back-propagate to the Jacobians, each pixel's gradient has a global effect, leading to a more regularized solution, see Fig. 3.

Second, we observe that CLIP per-pixel embedded features are unfortunately view-dependent and the same 3D coordinate may be assigned significantly different features in different 2D viewpoints. This in turn implies a given vertex on the mesh may receive conflicting gradients from different viewpoints (conceptually, when deforming a cow into an elephant, one viewpoint may try to tug on a vertex to grow a trunk out of it, while the other will try to use it as a tip of a tusk). This leads to incoherent results, and, in some cases, lack of global consistency (e.g., adding multiple trunks when deforming a cow into an elephant). To counter that, we devise a novel

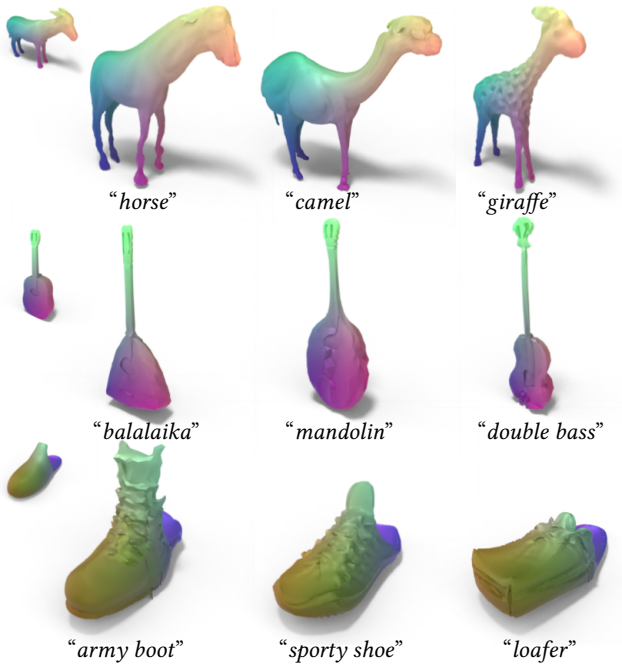

Figure 4: Text-driven deformation. Our method can deform the same source into different text-specified geometries, achieving local geometric details (shoe laces and giraffe) and low-frequency shape modifying deformations (body of guitar, shape of animal).

loss, which encourages vertices to achieve similar CLIP features from different viewpoints, thereby leading to global coherency in the deformations.

Third, to ensure the deformed shape still lies in semantic correspondence to the input shape, we add a identity-preserving term, which ensures that the deformation optimization step does not stray too far from the initial input mesh, thereby preventing the optimization from ignoring the input geometry.

Through experiments, we show we can apply text-driven deformations to a large class of source shapes and desired targets (from organic to man-made shapes). TextDeformer produces plausible shapes, beyond the capabilities of previous text-driven mesh generation techniques, while additionally providing abilities achieved solely through a deformation framework, such as significant resemblance to the input shape, and providing meaningful correspondences between the source and deformed shapes.

2 RELATED WORK

There has been a large variety of works in the space of text-guided content synthesis driven by CLIP [Radford et al. 2021], a foundational model which learns a joint embedding space for text and images. Many generative models such as StyleCLIP [Patashnik et al. 2021], GLIDE [Nichol et al. 2021], and DALLE-2 [Ramesh et al. 2022] leverage text during training by computing distance between text and images in the embedding space of CLIP. Additionally, there has been work on using CLIP guidance in fine-tuning the latest state-of-the-art diffusion models to achieve even higher quality results [Kim et al. 2022].

2.1 Text-Guided 3D Synthesis.

In comparison to text guided image generation, text-to-3D is relatively undeveloped. While there are some works that propose training joint embeddings of text descriptions and 3D objects [Chen et al. 2018], these works are lacking in scale as the largest captioned 3D dataset (the recent ObjaVerse dataset with 800k assets [Deitke et al. 2022]) is several orders of magnitude smaller than the LAION-5B dataset [Schuhmann et al. 2022]. Nevertheless, there have been large strides in text-to-3D leveraging large pre-trained 2D models such as CLIP.

CLIP-Forge [Sanghi et al. 2021] overcomes the lack of pre-trained counterpart to CLIP for 3D by using renderings of training shapes to bridge the gap between text and 3D data. They first train a voxel encoder and an implicit decoder on available 3D datasets using CLIP image embeddings, then swap image embeddings for text embeddings at inference time. However, their method is still limited by the availability of 3D datasets used to train their autoencoder. Point-E [Nichol et al. 2022] proposes to generate an image from text using a 2D diffusion model, then trains a point cloud diffusion model conditioned on images on a private dataset of millions of 3D shapes. While not on par with the state-of-the-art in terms of shape quality, their approach can generate 3D shapes significantly faster.

Several approaches tackle zero-shot geometry synthesis, bypassing the need of a 3D dataset. Dreamfields [Jain et al. 2021] leverage volume rendering with a Neural Radiance Field (NeRF) [Mildenhall et al. 2020] to directly optimize views of a 3D shape against a desired text prompt in CLIP's embedding space. Recently, leveraging 2D diffusion models [Rombach et al. 2022; Saharia et al. 2022], DreamFusion [Poole et al. 2022] and [Wang et al. 2022] propose to distill such models as a differentiable image-based loss. Extracting and editing an explicit mesh from these works is not straightforward, since NeRFs represent shapes through network weights.

Other works have used surface-based differentiable rendering in order to pass views of explicit 3D objects to CLIP. Using this method, Text2Mesh [Michel et al. 2021] employs a network to predict colors and deformations along the normals of a template mesh. Their objective is to *stylize* the template mesh while *preserving* the initial content. In contrast, CLIP-Mesh [Khalid et al. 2022] proposes deforming the vertices of a sphere in accordance to an input text prompt to synthesize a completely new geometry. Magic3D [Lin et al. 2022] combines both representations by first optimizing a radiance field using score-distillation similar to DreamFusion, then extracts an explicit mesh from the radiance field and optimizing its vertices via differentiable surface rendering and score distillation. The code is not public at the time of submission. Our work also leverages differentiable rendering and CLIP, but focuses on the problem of *deforming* explicit geometry rather than generating it from scratch.

2.2 Neural Shape Deformation.

Shape deformation has been traditionally approached by providing a user with handles that control a deformation space, either through an energy-minimizing formulation [Sorkine et al. 2004; Sorkine and

Alexa 2007] or through skinning the mesh with weights that interpolate coordinates with respect to the handles [Jacobson et al. 2014; Fulton et al. 2019]. Skinning methods have been used extensively in learning context [Xu et al. 2019, 2020; Holden et al. 2015; Li et al. 2021], point handles [Liu et al. 2021], and cages [Yifan et al. 2020], while variational formulations are used as regularizers, e.g., ARAP [Sun et al. 2021] or the Laplacian [Kanazawa et al. 2018]. Both of these approaches span only a subset of possible deformations and prevent fine-grained control over details.

Other methods focus on learning tasks over one template mesh [Tan et al. 2018; Gao et al. 2018], and assign per-vertex coordinates [Shen et al. 2021] or offsets from a simpler (e.g., linear) model [Bailey et al. 2018, 2020; Romero et al. 2021; Zheng et al. 2021; Yin et al. 2021].

While some recent works propose data-driven approaches to predict realistic deformations [Aigerman et al. 2022; Jakab et al. 2020; Yifan et al. 2020; Hanocka et al. 2018; Yumer et al. 2015], the semantic capabilities of all these works are limited once again by the lack of 3D datasets pairing shapes and captions. Our work is similar in spirit to these methods, but instead of requiring explicit supervision, we leverage differentiable rendering and powerful visual models such as CLIP to drive deformations of a template shape.

3 METHOD

Fig. 5 shows an overview of our method. Given an input shape, TextDeformer enables manipulating the geometry guided by a user-specified text description.

We represent the geometry of the input shape using a mesh $\mathcal{M}$ defined by a set of vertices $\mathcal{V} \in \mathbb{R}^{n\times 3}$ and faces $\mathcal{F}$. We optimize a displacement map $\Phi : \mathbb{R}^3 \to \mathbb{R}^3$ over the vertices through differentiable rendering.

3.1 Deformations through Jacobians.

A naive optimization of Φ would simply entail directly displacing each vertex $\mathcal{V}$, which may overly distort the original shape, especially when the target text describes a highly-detailed texture. This is due to this representation exposing high-frequency, oscillatory modes of deformation. Thus, in this work we opt for a different parameterization of deformations. Inspired by Neural Jacobian Fields [Aigerman et al. 2022], we parameterize the shape using a set of per-triangle Jacobians which define a deformation. Specifically, we represent per-triangle jacobians by matrices $J_i \in \mathbb{R}^{3\times 3}$ for every face $f_i \in \mathcal{F}$. Following [Aigerman et al. 2022], we solve a Poisson problem to compute a deformation map Φ^* as the mapping with Jacobian matrices for each face that are closest to $\{J_i\}$ in the least-squares sense, that is:

$$\Phi^* = \min_{\Phi} \sum_{f_i \in \mathcal{F}} |f_i| \|\nabla_i(\Phi) - J_i\|_2^2 \tag{1}$$

where $\nabla_i(\Phi)$ denotes the Jacobian of Φ at triangle f_i and $|f_i|$ is the area of that triangle. Hence, we may optimize the deformation mapping Φ indirectly by optimizing the matrices $\{J_i\}$ which define Φ^*. These Jacobians are initialized to identity matrices, thereby initializing Φ^* as the identity mapping. Please refer to [Aigerman et al. 2022] for the full technical details.

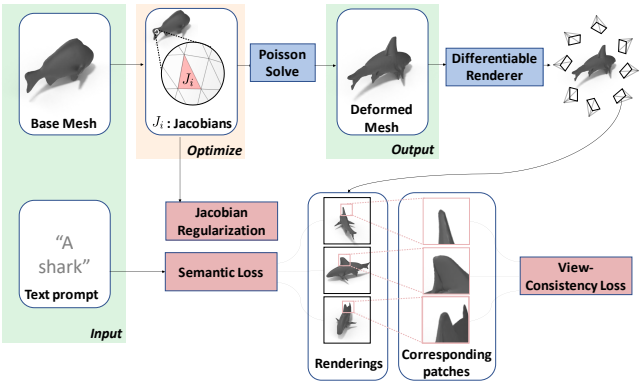

Figure 5: Overview. TextDeformer deforms a base mesh by optimizing per-triangle Jacobians using natural language as a guide. We optimize the deformation using three losses: a CLIP-based semantic loss drives the deformation toward the text prompt, a view-consistency loss matches multiple views of the same surface patch to ensure a coherent deformation, and our regularization on the Jacobians controls the fidelity to the base mesh.

3.2 Language Guidance.

Our objective is to use text to guide the deformation of the source shape. We leverage the pre-trained vision-language CLIP [Radford et al. 2021], which provides a shared embedding space between images and text. In order to connect geometry to images, we pass our shape through a differentiable renderer $\mathcal{R}$ [Laine et al. 2020]. Hence, we may differentiably embed the renders of the deformed shape

$$e_{\mathcal{M}} = \texttt{CLIP}(\Phi^*(\mathcal{M})) \in \mathbb{R}^{512}$$

We abuse notation and the rendering function $\mathcal{R}$ is understood to be implicit when passing shapes to CLIP. The desired deformation, described with a natural language prompt $\mathcal{P}$, is also embedded $e_{\mathcal{P}} = \texttt{CLIP}(\mathcal{P}) \in \mathbb{R}^{512}$. Then, we may optimize Φ^* such that $e_{\mathcal{M}}$ and $e_{\mathcal{P}}$ agree, by maximizing the cosine similarity between the embeddings:

$$\mathcal{L}_{\mathcal{P}}(\Phi^*, \mathcal{M}, \mathcal{P}) = \text{sim}\left(e_{\mathcal{M}}, e_{\mathcal{P}}\right) \tag{2}$$

where $\text{sim}(\cdot,\cdot)$ stands for cosine similarity. As in StyleCLIP [Patashnik et al. 2021], we find that incorporating relative directions in CLIP's embedding space can give stronger signals when the optimization landscape between Φ^* and $\mathcal{P}$ is unclear. Given a base caption $\mathcal{P}_0$ that describes $\mathcal{M}$, we compute the direction between the target prompt and the base prompt: $\Delta\text{CLIP}(\mathcal{P}, \mathcal{P}_0) = \texttt{CLIP}(\mathcal{P}) - \texttt{CLIP}(\mathcal{P}_0)$. We compute the direction of the deformations and aim to optimize:

$$\mathcal{L}_{\Delta\mathcal{P}}(\Phi^*, \mathcal{P}, \mathcal{P}_0) = \text{sim}\left(\Delta\text{CLIP}(\mathcal{P}, \mathcal{P}_0), \Delta\texttt{CLIP}(\Phi^*(\mathcal{M}), \mathcal{M})\right). \tag{3}$$

3.3 Jacobian Regularization.

To prevent the deformation from straying too far from the input undeformed geometry, we introduce another regularization term on the predicted Jacobians, which penalizes the difference between

the Jacobians $\{J_i\}$ and the identity, i.e., no deformation:

$$\mathcal{L}_I(t_j) = \alpha \sum_{i=1}^{|\mathcal{F}|} \|J_i - I\|_2 \tag{4}$$

where α is a hyper-parameter which may be tuned to control the strength of the deformations defined by $\{J_i\}$.

3.4 View Consistency.

A common problem when performing multi-view optimization through CLIP is the lack of view-consistency, *i.e.*, a particular view may pull the shape toward a specific deformation while another view pulls toward a different deformation. Averaging gradients on the Jacobians for each view does not necessarily lead to a coherent 3D deformation which may manifest in various artifacts, including muddled details and incorrect geometry.

We introduce another regularization term to tackle this problem by utilizing the patch-level deep features of CLIP's vision transformer (ViT). In ViTs, the image is split into non-overlapping patches $P_0, P_1, \ldots P_n$, which are then projected into a higher-dimensional space and passed through transformer encoder blocks $\mathcal{T}_1, \mathcal{T}_2, \ldots \mathcal{T}_n$. For each vertex $v \in \mathcal{V}$, and each render $r \in \mathcal{R}(\mathcal{M})$, if v is visible, we compute the pixel $p(v, r)$ in r that contains v. Then, by associating $p(v, r)$ with the nearest corresponding patch center $P(v, r)$, we extract a deep feature vector corresponding to v and r and encourage vertices to have similar deep features across renders from different viewpoints:

$$\mathcal{L}_{\text{VC}}(v) = \sum_{i=1}^{|\mathcal{R}(\mathcal{M})|} \sum_{\substack{j=1 \\ j \neq i}}^{|\mathcal{R}(\mathcal{M})|} \text{sim}\left(\mathcal{T}_k(P(v, r_i)), \mathcal{T}_k(P(v, r_j))\right) \tag{5}$$

for some chosen layer $\mathcal{T}_k$. In practice, we choose to use the *token* output of the final transformer block. Then, we simply penalize this loss over all vertices $v \in \mathcal{M}$:

$$\mathcal{L}_{\text{VC}}(\mathcal{M}) = \beta \sum_{v \in \mathcal{V}} \mathcal{L}_{\text{VC}}(v) \tag{6}$$

where β is another tunable hyper-parameter. To compute $P(v, r)$, we follow [Amir et al. 2022] by modifying CLIP's ViT to use a smaller stride in its initial convolution, obtaining *overlapping* patches. Then, by interpolating the positional encoding, we achieve finer-resolution deep features.

4 EXPERIMENTS

We run TextDeformer on a variety of text prompt, source mesh pairs. Each pair takes approximately 1.5 hours for 5000 iterations.

4.1 Generality of TextDeformer

We show that TextDeformer can handle a wide variety of source and target prompts. We represent the source mesh as a small inset next to the deformed shapes unless otherwise specified. Fig. 2 shows one such collection of source and targets. We see that TextDeformer is capable of producing high-quality results for different types of source and target pairs.

Adjective Targets. We see that TextDeformer is capable of deforming source meshes in accordance to a target *adjective*. In Fig. 2, we see that TextDeformer deforms a cat to be "*fluffy*," a vase to be "*heart-shaped*," an alien to be "*scary*," and a shoe to be "*high-heeled*."

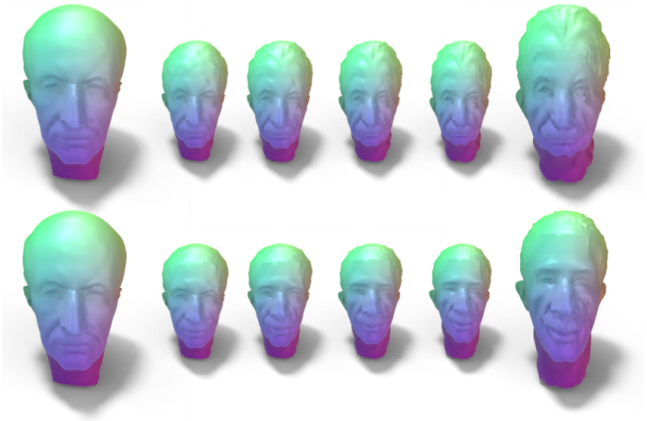

Figure 6: Visualizing iterations of the optimization as the source mesh is deformed to the target ("*Einstein*", "*Obama*"). Due to our formulation and energy, each facial feature of Max Planck is deformed into the corresponding facial feature of the target (nose to nose, eyes to eyes etc.).

For this style of target text, our method produces deformation maps that preserve the overall structure of the source shape.

Related Targets. TextDeformer is also capable of deforming source meshes into *related* shapes, which are not exactly descriptors of the original mesh, but also not too semantically different. For example, in Fig. 2, we see that TextDeformer is able to deform an acoustic guitar into a "*banjo*," a candle into an "*incense stick*," the Eiffel tower into a "*pagoda*," a vase into a "*royal goblet*," and so on.

Unrelated Targets. Finally, we observe that TextDeformer is capable of deforming source meshes into completely *unrelated* shapes, which are far from the source mesh. This capability is illustrated in Fig. 2 where an ant is deformed into a "*snake*," and a clothing iron is deformed into a "*kettle*," as well as in Fig. 1 where a cow is deformed into various unrelated animals such as a "*turtle*".

4.2 Expressiveness of TextDeformer

Frequency. We study the expressiveness of our method by experimenting with different text prompts for the same source mesh. We observe that TextDeformer is powerful enough to express both high-frequency texture details as well as low-frequency deformations to the shape structure. In Fig. 4, we see in the top row that TextDeformer can produce the requisite low-frequency deformations to change the donkey into various other animal shapes, but also add high-frequency deformations to emulate "*giraffe*" spots. Similarly in middle row, TextDeformer produces different low-frequency maps to change the shape of the guitar body while preserving the neck. Alternately in the last row, TextDeformer produces high-frequency deformations to emulate fine details such as the laces of an "*army boot*" or the creases in a "*sporty shoe*." Similarly, we observe TextDeformer producing high-frequency deformations in Fig. 1, particular in the "*alligator*" example.

Dense Matching. In Fig. 6, we demonstrate the ability of TextDeformer to preserve highly detailed semantics throughout the optimization process. Not does TextDeformer correctly deform features

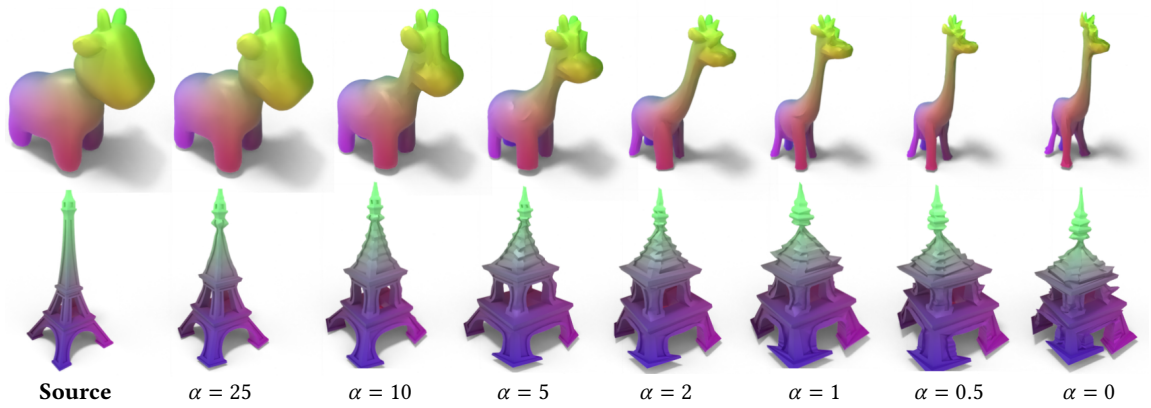

Figure 7: Identity preservation. The source shape (left) is deformed into a "*giraffe*" (top), "*pagoda*" (bottom) using decreasing amount of weight on the proposed Jacobian regularization. Higher weight encourages preservation of the identity of the source. Note that results with zero Jacobian regularization (right column) may contain artifacts.

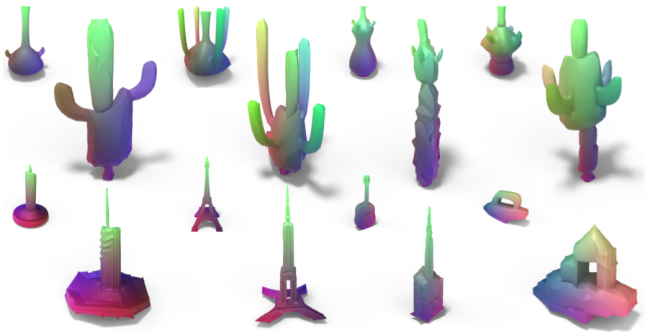

Figure 8: Deformations of different source vases into the same target text, "*a cactus*" and different source meshes into the same target text, "*a skyscraper*". Our method meaningfully respects the input geometry while conforming to the desired target text.

of Max Planck to the corresponding features of the target ("*Einstein*," "*Obama*"), it is consistent at every step. We observe at different intermediate faces that the corresponding features are always mapped correctly.

4.3 Identity Preservation

We demonstrate our method's ability to preserve the source shape in two ways.

Impact of the Input Geometry. In Fig. 8, we use TextDeformer to deform different vases with various appendages into "*a cactus*." Note that the resulting deformation map grows cactus branches depending on where appendages are on the source mesh. Hence, TextDeformer is able to produce a deformation map that conforms to the target text, "*cactus*," in an adaptive manner, borrowing coarse structures from the input geometry. We also observe a similar effect when deforming semantically diverse meshes into the target text, "*skyscraper*". TextDeformer preserves different aspects of each mesh, such as the base of the candle and the body of the guitar. It also produces interesting variation e.g. deforming the flatter iron into a dome structure instead of a thinner needle.

Jacobian Regularization. Recall that we also define a Jacobian regularization term $\mathcal{L}_I$ in (2) which is scaled by a hyper-parameter α. In Fig. 7, we show that adjusting α controls how far the deformation map Φ^* is allowed to deviate from the source mesh. With $\alpha = 25$, we see that the cow and the Eiffel tower do not change meaningfully in accordance to their respective text prompts ("*giraffe*" and "*pagoda*"), while setting $\alpha = 0$ may result in some artifacts in the deformed shape. We observe that setting α to intermediate values offers the best results.

4.4 Viewpoint Consistency

We experiment with the qualitative effect of the viewpoint consistency loss $\mathcal{L}_{\text{VC}}$ by using TextDeformer with and without this loss term and noting the differences in the deformed meshes. We observe in Figure 9 that the deformations produced without $\mathcal{L}_{\text{VC}}$, although smooth due to our choice in representation, often contain unrealistic geometric features. Such abnormalities can be caused by outliers in the sampled camera views during optimization. This problem is especially apparent in the "*gaming chair*," in which the backrest is crooked. This inaccuracy may appear correct in some perspectives, but is overall an undesirable feature. Another example is the tip of the "*skyscraper*," which is pulled to one side of the building during optimization. Finally, we observe that the back of the "*fluffy poodle*," and the wings of the "*bat*" are incorrectly curved inwards when they are optimized without $\mathcal{L}_{\text{VC}}$. We provide a quantitative evaluation of these observations in Sec. 4.7.

4.5 Effect of Jacobians

We also experiment with replacing Jacobians with vertex displacements in our pipeline, but keeping the additional losses we introduce, $\mathcal{L}_{\Delta\mathcal{P}}$ and $\mathcal{L}_{\text{VC}}$.

Surface Quality. We observe that our use of Jacobians plays a key in obtaining high-quality surface geometries. In Fig. 10, we show that replacing Jacobians with vertex displacements in TextDeformer

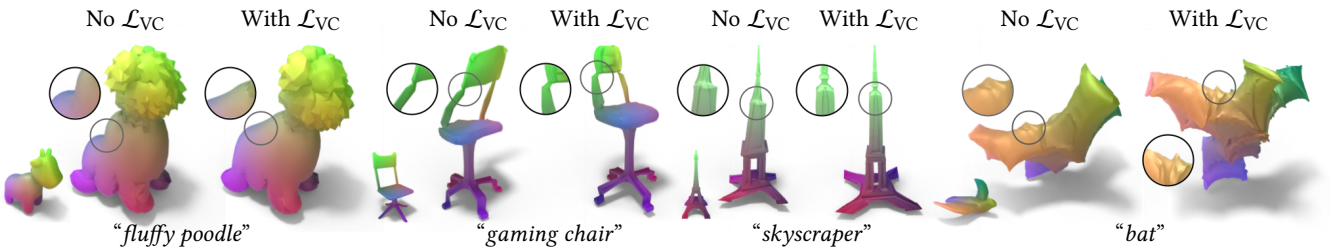

Figure 9: Viewpoint Consistency Ablation. Ablation results of removing $\mathcal{L}_{VC}$ for four different source-text pairs. In each example we see instances of incorrect geometry in the shapes deformed without $\mathcal{L}_{VC}$.

drastically decreases the resulting surface quality for three different examples. In the "*diamond-shaped vase*," vertex displacements cause the shape to collapse inwards. In the other two examples, vertex displacements create details which do not respect the initial shape template and contain many self-intersections. The bottom row of Fig. 12 highlights this deterioration of surface quality for another example. We observe that vertex displacements cause a large number of self-intersections, even in areas where the resulting surface appears to require a relatively simple deformation, such as the lens of the "*goggles*."

Globally-Coherent Deformations. Recall from (1) that we solve a global Poisson system to compute the deformation map from the Jacobians. Hence, unlike the gradients of vertices which only affect one point, gradients propagated through Jacobians influence a large surface mesh area, leading to a *globally-coherent* deformation. In Fig. 3, in which a bust of Max Planck is deformed to various other faces, we observe that vertex displacements take local, sub-optimal steps, not only contributing to poor surface quality, but also, in the example of "*Obama*," causing spurious face mirroring on the back of the head. Since there are few camera views in which the front and back of the head are visible, the optimization process naturally leads to this result. However, we observe that Jacobians do not experience this artifact.

4.6 Qualitative Comparisons

Beyond ablations, we also qualitatively evaluate our method against two existing text-based methods for 3D synthesis and editing, CLIP-Mesh and Text2Mesh. In Fig. 11, we show results for 5 different target text prompts: "*an alligator*", "*a camel*", "*a chinese lantern*", "*a comfortable chair*", and "*an octupus*". TextDeformer outperforms the baselines in multiple aspects. First, our method produces more semantically accurate deformations: a more elongated "*alligator*", a larger "*comfortable chair*", and a rounder "*chinese lantern*". In the examples of "*a camel*" and "*an octopus*" where CLIP-Mesh produces semantically correct results, we observe that it produces poor quality surfaces with irregular triangulation and self-intersections, whereas TextDeformer produces smoother, more realistic geometry.

4.7 Quantitative Evaluation

We evaluate TextDeformer quantitatively in two ways, comparing to CLIP-Mesh [Khalid et al. 2022], Text2Mesh [Michel et al. 2021], as well as our method without $\mathcal{L}_{VC}$ and our method but using vertex displacements.

Table 1: Quantitative evaluation. We use our text prompts and deformed meshes in a retrieval task to compute R-Precision. We observe that regularizing for viewpoint consistency improves TextDeformer R-Precision. TextDeformer and CLIP-Mesh achieve quantitatively comparable R-Precision, TextDeformer (Ours) produces higher-quality geometry both qualitatively (see Fig. 12) and quantitatively (significantly fewer self-intersections). All methods significantly outperform Text2Mesh in R-Precision.

	CLIP R-Precision (L/14) ↑	Intersections ↓
Ours	55.2%	3.2%
Ours-noVP	51.5%	3.3%
Ours-Verts	55.4%	67.7%
CLIP-Mesh	57.4%	62.8%
Text2Mesh	12.7%	17.3%

Retrieval Precision. First, following CLIP-Mesh [Khalid et al. 2022], we compute the R-Precision of CLIP-L/14 on a retrieval task. Specifically, we use a set of 111 text prompt, source mesh pairs to generate a set of deformed meshes. Then, we retrieve deformed shapes for each text prompt through CLIP (L/14) cosine similarity. The first column of Tab. 1 shows the results of this experiment. We observe that the viewpoint consistency loss $\mathcal{L}_{VC}$ increases the deformation quality of the model, affirming our observations in Sec. 4.4. We also observe that the pipelines using vertex displacements achieve the highest R-Precision scores. This is to be expected as vertex displacements give a high amount of freedom when deforming the shape towards the text prompt, without respect to the initial template geometry. Finally, all methods outperform Text2Mesh, which is more suited for stylization and texturization problems rather than our semantic editing problem.

Geometric Quality (Self-Intersections). We also measure the ratio between the number of self-intersections in the deformed mesh and the number of faces. This validates the qualitative evaluation of in Fig. 10, Fig. 11, and Fig. 12 that the vertex displacement pipelines disregard the triangulation of the shape in order to optimize CLIP cosine-similarity.

4.8 Qualitative Comparison with Stable Dreamfusion

In Figure 13, we compare qualitatively against Stable Dreamfusion. We use the third-party open source implementation of the

method [1] [Tang 2022]. We show the geometry extracted from the neural radiance field with the representative neural image as an inset. We observe that the surfaces of geometries from Dreamfusion lack smoothness, which we naturally obtain through our Jacobian representation. Second, we observe that DreamFusion often suffers from the Janus effect, even wiht the use of prompt augmentation on the renderings, whereas our View-Consistency loss helps produce more coherent geometry.

5 DISCUSSION AND FUTURE WORK

In this paper, we propose TextDeformer, a zero-shot text-driven mesh deformation technique which does not need to be trained on *any* 3D dataset or 3D annotations. Instead, it is guided by pre-trained vision-language models trained on billions of visual and language concepts.

Our work aims to produce high-quality geometry outputs by predicting low-frequency shape changes and high-frequency details through source shape *deformations*. We opt to use per-face *Jacobians* as a means for predicting smooth mesh deformations, which enables retaining interesting characteristics of the source shape. This leads to high-quality mesh outputs with useful geometry and mitigates local artifacts commonly caused by vertex displacements.

We presented a view consistency loss, which avoids over-fitting geometry to specific salient views, and ensures that the same region is roughly interpreted the same from all viewpoints. Our novel loss significantly reduces the number of visual artifacts. We also propose an identity regularization term, which can be controlled by the user to control the magnitude of the deformation. We demonstrate that our method enables retaining interesting global characteristics of the source shape, while still matching it to a highly dissimilar term, providing the user with a controllable and expressive system.

In the future, we would like to explore the possibility of learning the space of prompt-driven deformations instead of just optimizing them for a single mesh instance. This strategy would be fast, since Jacobians can be predicted in a single feed-forward pass, instead of through optimization. In addition, training over a collection of shapes may induce a neural-regularization which may improve the results further. We would like to also connect our method with a retrieval module to provide a more comprehensive artist-driven creation tool that enables users to explore the results arising from different combinations of sources and prompts.

REFERENCES

Noam Aigerman, Kunal Gupta, Vladimir G. Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. 2022. Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes. *ACM Trans. Graph.* 41, 4, Article 109 (jul 2022), 17 pages. https://doi.org/10.1145/3528223.3530141

Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. 2022. Deep ViT Features as Dense Visual Descriptors. *ECCVW What is Motion For?* (2022).

Stephen W Bailey, Dalton Omens, Paul Dilorenzo, and James F O'Brien. 2020. Fast and deep facial deformations. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 94–1.

Stephen W. Bailey, Dave Otte, Paul Dilorenzo, and James F. O'Brien. 2018. Fast and Deep Deformation Approximations. *ACM Transactions on Graphics* 37, 4 (Aug. 2018), 119:1–12. https://doi.org/10.1145/3197517.3201300 Presented at SIGGRAPH 2018, Los Angeles.

Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. 2018. Text2Shape: Generating Shapes from Natural Language by Learning Joint Embeddings. *arXiv preprint arXiv:1803.08495* (2018).

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. 2022. Objaverse: A Universe of Annotated 3D Objects. *arXiv preprint arXiv:2212.08051* (2022).

Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. 2019. Latent-space Dynamics for Reduced Deformable Simulation. *Computer Graphics Forum* (2019).

Ran Gal, Olga Sorkine, Niloy J. Mitra, and Daniel Cohen-Or. 2009. iWIRES: An Analyze-and-Edit Approach to Shape Manipulation. *ACM Transactions on Graphics (Siggraph)* 28, 3 (2009), #33, 1–10.

Lin Gao, Jie Yang, Yi-Ling Qiao, Yu-Kun Lai, Paul L Rosin, Weiwei Xu, and Shihong Xia. 2018. Automatic Unpaired Shape Deformation Transfer. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH Asia 2018)* 37, 6 (2018), To appear.

Rana Hanocka, Noa Fish, Zhenhua Wang, Raja Giryes, Shachar Fleishman, and Daniel Cohen-Or. 2018. ALIGNet: partial-shape agnostic alignment via unsupervised learning. *ACM Transactions on Graphics (TOG)* 38, 1 (2018), 1.

Daniel Holden, Jun Saito, and Taku Komura. 2015. Learning an Inverse Rig Mapping for Character Animation. In *Proceedings of the 14th ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '15)*. Association for Computing Machinery, New York, NY, USA, 165–173. https://doi.org/10.1145/2786784.2786788

Alec Jacobson. 2013. Algorithms and Interfaces for Real-Time Deformation of 2D and 3D Shapes. In *PhD Thesis, ETH Zurich*.

Alec Jacobson, Zhigang Deng, Ladislav Kavan, and JP Lewis. 2014. Skinning: Real-time Shape Deformation. In *ACM SIGGRAPH 2014 Courses*.

Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. 2021. Zero-Shot Text-Guided Object Generation with Dream Fields. *arXiv* (December 2021).

Tomas Jakab, Richard Tucker, Ameesh Makadia, Jiajun Wu, Noah Snavely, and Angjoo Kanazawa. 2020. KeypointDeformer: Unsupervised 3D Keypoint Discovery for Shape Control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. 2018. Learning Category-Specific Mesh Reconstruction from Image Collections. In *ECCV*.

Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. 2022. CLIP-Mesh: Generating textured meshes from text using pretrained image-text models. *SIGGRAPH Asia 2022 Conference Papers* (December 2022).

Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. 2022. DiffusionCLIP: Text-Guided Diffusion Models for Robust Image Manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2426–2435.

Vladislav Kraevoy and Alla Sheffer. 2004. Cross-Parameterization and Compatible Remeshing of 3D Models. In *ACM Transactions on Graphics (Proc. SIGGRAPH)*.

Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. 2020. Modular Primitives for High-Performance Differentiable Rendering. *ACM Transactions on Graphics* 39, 6 (2020).

Peizhuo Li, Kfir Aberman, Rana Hanocka, Libin Liu, Olga Sorkine-Hornung, and Baoquan Chen. 2021. Learning Skeletal Articulations with Neural Blend Shapes. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 1.

Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. 2022. Magic3D: High-Resolution Text-to-3D Content Creation. *arXiv preprint arXiv:2211.10440* (2022).

Minghua Liu, Minhyuk Sung, Radomir Mech, and Hao Su. 2021. DeepMetaHandles: Learning Deformation Meta-Handles of 3D Meshes with Biharmonic Coordinates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12–21.

Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. 2021. Text2Mesh: Text-Driven Neural Stylization for Meshes. *arXiv preprint arXiv:2112.03221* (2021).

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. 2021. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. *CoRR* abs/2112.10741 (2021). arXiv:2112.10741 https://arxiv.org/abs/2112.10741

Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. 2022. Point-E: A System for Generating 3D Point Clouds from Complex Prompts. *arXiv preprint arXiv:2212.08751* (2022).

Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. 2021. StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2085–2094.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. 2022. DreamFusion: Text-to-3D using 2D Diffusion. *arXiv* (2022).

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 8748–8763.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. *ArXiv* abs/2204.06125 (2022).

[1] https://github.com/ashawkey/stable-dreamfusion

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10684–10695.

Cristian Romero, Dan Casas, Jesus Perez, and Miguel A. Otaduy. 2021. Learning Contact Corrections for Handle-Based Subspace Dynamics. *ACM Trans. on Graphics (Proc. of ACM SIGGRAPH)* 40, 4 (2021). http://gmrv.es/Publications/2021/RCPO21

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. 2022. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. *arXiv preprint arXiv:2205.11487* (2022).

Aditya Sanghi, Hang Chu, Joseph G Lambourne, Ye Wang, Chin-Yi Cheng, and Marco Fumero. 2021. Clip-forge: Towards zero-shot text-to-shape generation. *arXiv preprint arXiv:2110.02624* (2021).

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. 2022. Laion-5b: An open large-scale dataset for training next generation image-text models. *NeurIPS* (2022).

Meitar Shechter, Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. 2022. NeuralMLS: Geometry-Aware Control Point Deformation. (2022).

Siyuan Shen, Yin Yang, Tianjia Shao, He Wang, Chenfanfu Jiang, Lei Lan, and Kun Zhou. 2021. High-order differentiable autoencoder for nonlinear model reduction. *ACM Transactions on Graphics*.

Olga Sorkine and Marc Alexa. 2007. As-Rigid-As-Possible Surface Modeling. In *Proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*. 109–116.

Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and Hans-Peter Seidel. 2004. Laplacian Surface Editing. In *Proceedings of the EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*. ACM Press, 179–188.

Bo Sun, Xiangru Huang, Qixing Huang, Zaiwei Zhang, Junfeng Jiang, and Chandrajit Bajaj. 2021. ARAPReg: An As-Rigid-As Possible Regularization Loss for Learning Deformable Shape Generators. In *ICCV*.

Qingyang Tan, Lin Gao, Yu-Kun Lai, and Shihong Xia. 2018. Variational Autoencoders for Deforming 3D Mesh Models. In *CVPR*.

Jiaxiang Tang. 2022. Stable-dreamfusion: Text-to-3D with Stable-diffusion. https://github.com/ashawkey/stable-dreamfusion.

Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A. Yeh, and Greg Shakhnarovich. 2022. Score Jacobian Chaining: Lifting Pretrained 2D Diffusion Models for 3D Generation. *arXiv preprint arXiv:2212.00774* (2022).

Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. 2020. RigNet: Neural Rigging for Articulated Characters. *ACM Trans. on Graphics* 39 (2020).

Zhan Xu, Yang Zhou, Evangelos Kalogerakis, and Karan Singh. 2019. Predicting Animation Skeletons for 3D Articulated Models via Volumetric Nets. In *2019 International Conference on 3D Vision (3DV)*.

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

Kangxue Yin, Jun Gao, Maria Shugrina, Sameh Khamis, and Sanja Fidler. 2021. 3DStyleNet: Creating 3D Shapes with Geometric and Texture Style Variations. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Mehmet Ersin Yumer, Siddhartha Chaudhuri, Jessica K Hodgins, and Levent Burak Kara. 2015. Semantic shape editing using deformation handles. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–12.

Mianlun Zheng, Yi Zhou, Duygu Ceylan, and Jernej Barbic. 2021. A Deep Emulator for Secondary Motion of 3D Characters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5932–5940.

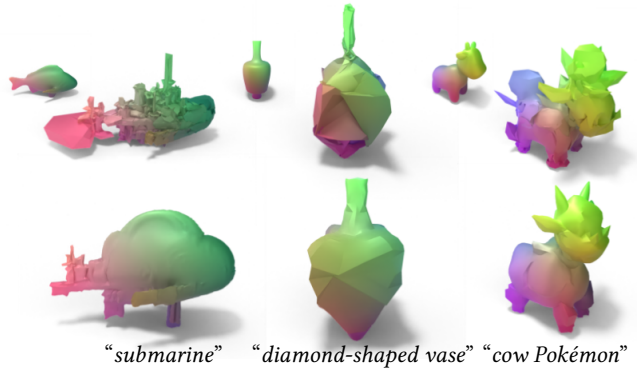

Figure 10: Deformation Ablation. Top row: TextDeformer with vertex displacements. Bottom row: TextDeformer with Jacobians. Jacobians are crucial to preserving the structure of the input geometry and maintaining high-quality surfaces.

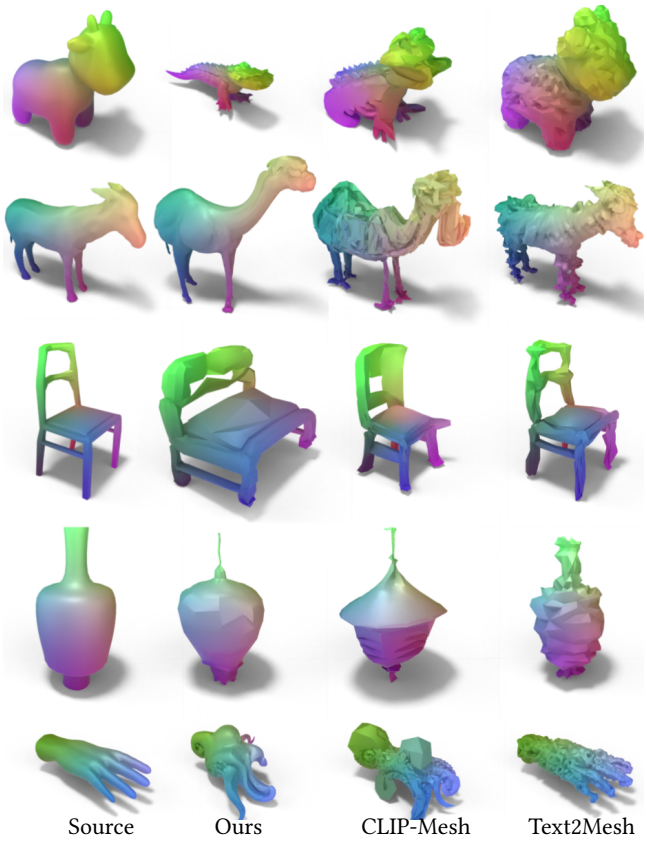

Figure 11: Qualitative Comparison From top to bottom, the target text prompts are "*an alligator*", "*a camel*", "*a comfortable chair*", "*a chinese lantern*", "*an octupus*". Compared to CLIP-Mesh, our method produces more semantically correct and higher quality surfaces. Text2Mesh fails to produce semantically meaningful deformations in all examples.

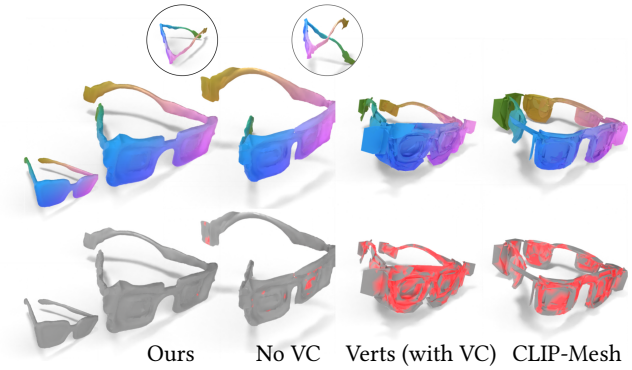

Figure 12: Self-intersections. Comparison results for the shown source and target text "goggles". Self-intersections are highlighted in red (bottom row). Removing view-consistency (VC) losses causes distortion on the temple arms. Removing Jacobians and optimizing vertices introduces further surface distortion and self-intersections which may impede utility. When applying CLIP-Mesh to this template, we observe the "janus" effect e.g. unrealistic repeated geometry on each side.

Figure 13: Comparison to Stable Dreamfusion We compare TextDeformer against the open-source implementation of Dreamfusion [Tang 2022] based on Stable Diffusion [Rombach et al. 2022]. For Stable Dreamfusion, we show a representative neural rendering as well as the extracted geometry. For TextDeformer, we show the initial mesh. First, notice that the surface of Dreamfusion meshes has heavy artifacts compared to the smoothness of our deformations obtained through Jacobians. Second, we notice that Dreamfusion suffers frequently from the Janus problem (see the high heels for instance) which we help alleviate with our View-Consistency loss.