

# Text2VDM: Text to Vector Displacement Maps for Expressive and Interactive 3D Sculpting

## Supplementary Material

In this supplementary material, we provide additional details and results that are not included in the main paper due to the space limit.

### A. Implementation details

**Shape Control.** Our framework provides a simple and intuitive interactive interface (Figure S1) to assist users in drawing user-specified VDMs to initialize the base mesh.

**Brush Optimization.** In each iteration, We randomly sample camera poses in multi-view and render  $N$  view normal images at a resolution of  $512 \times 512$  pixels. We sample the elevation angle as  $\phi_{\text{elev}} \sim \mathcal{U}(0, \frac{\pi}{3})$ , and the azimuth angle as  $\phi_{\text{azim}} \sim \mathcal{U}(0, 2\pi)$ . We set  $N = 4$  in our experiment. We then feed them into the Stable Diffusion 2.1 to calculate the score distillation sampling (SDS) loss. The generation process runs on a single NVIDIA RTX 4090 GPU with 10,000 iterations per brush, which takes around 40 minutes.

**VDM Baking.** For the VDM baking, we write the coordinate values of each vertex of the final mesh into the corresponding pixels, resulting in a three-channel image with a resolution of  $512 \times 512$  in EXR file format. This EXR image serves as the final VDM output of the generated brush.

#### A.1. Interface Design for Shape Control

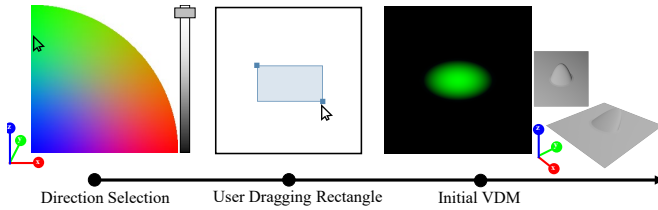


Figure S1. Interface design.

#### A.2. Details of Weighted Blending

To obtain semantically focused text embeddings for a more precise target distribution to mitigate the semantic coupling issue in SDS, we propose enhancing the semantics of part-related words by applying weighted blending to the tokens in the prompt. Specifically, we assign each word in the prompt a weight  $s$  and compute the weighted embedding  $e_w$  for each word by blending the original text embedding  $e$  with the empty text embedding  $e_\phi$  as follows:  $e_w = e_\phi + s \cdot (e - e_\phi)$ . By concatenating the weighted embeddings of each word in sequence, we obtain the final semantically focused text embedding. For instance, consider

the prompt: “A horn++ of a deer”, where ++ indicates that the word “horn” is weighted with a weight of  $1.1^2$ . In the example prompt shown in the main paper, words with yellow underlines represent those weighted with a weight value of 1.21 during the computation of text embeddings. Notably, the weights used for text embedding computation are separate from the CFG guidance scale applied during the computation of the SDS loss. Our experiment uses a CFG guidance scale of 100. Additionally, our method not only effectively alleviates semantic coupling during geometric structures generation but also enhances specific words in the prompt to produce surface details that better align with the text. For example, Figure S2 demonstrates how enhancing the word “cannonball” results in a better results.

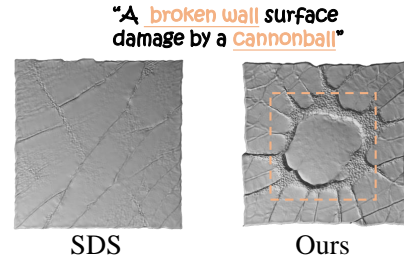


Figure S2. Effect of Semantic Enhancement SDS on surface details generation.

#### A.3. Region Control

In our framework, we provide a region mask to restrict mesh deformation to the user-defined region during optimization. By adjusting the activation ratio of the region mask, the final brush effect can effectively match the user’s guidance. When generating surface detail brushes, we activated the region mask during the first half of the total iterations as a warm-up stage, with the specific effects shown in Figure S3. For geometric structures generation, we activated the region mask throughout the entire optimization process to maintain zero values in unused VDM areas.

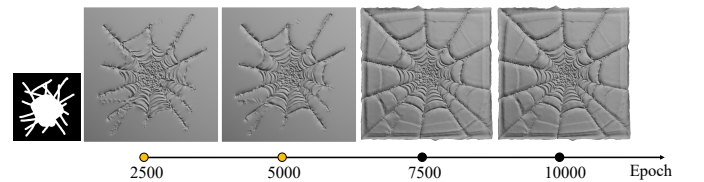


Figure S3. Region control.

## B. Details of Experiment Setting

### B.1. Details of Re-framed Paint-it

Paint-it originally uses SDS to optimize a UNet for generating PBR textures. We reframed it to suit our VDM brush generation task. Specifically, in the original Paint-it architecture, a UNet is used to predict a nine-channel output from a fixed 512×512 Gaussian noise image, where every three channels correspond to diffuse, specular, and normal maps, respectively. We modified the UNet to output only three channels, representing a VDM. Each pixel’s three values in the VDM are directly applied to the corresponding mesh vertices for three-axis displacement, resulting in the deformed mesh. We then render the normal map from the deformed mesh to compute the SDS loss. To ensure the quality of the mesh, we introduced regularization terms to facilitate mesh smoothness, such as Laplacian, edge, and normal consistency regularization.

### B.2. Base Mesh Initialization

In our experiments for generating geometric structures, to ensure a fair comparison with Text2Mesh and TextDeformer, we used the same initial base mesh and masks as shown in Figure S4. The specific correspondence between prompts and base meshes is detailed in Table 1. Since Paint-it directly generates VDMs, we only used a planar base mesh and do not use masks.

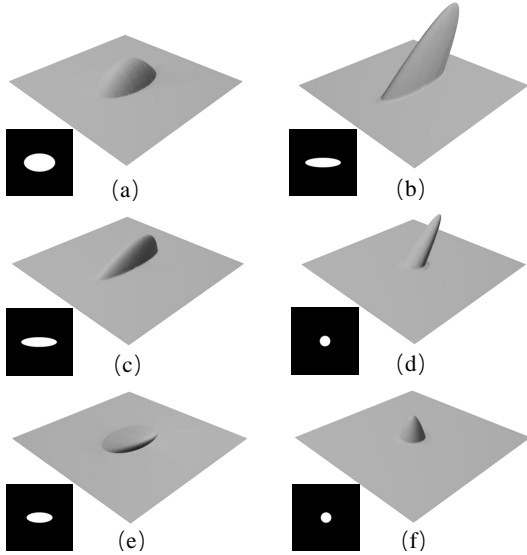


Figure S4. Base mesh initialization.

### B.3. Text Prompt of Quantitative Comparison

As described in Section 4.2 of the main paper, we used 40 text prompts for VDM generation and quantitative evaluation, with 20 focusing on surface details and the remaining

20 on geometric structures. The specific text prompts are detailed in Table 1.

Table 1. Text Prompts for VDM Generation.

VDM Type	Text Prompt
Surface Details	A blanket surface with many flowers
	A brick wall surface with neat brick
	A broken brick wall surface with many cracks
	A broken glass surface like spider web
	A broken wall surface damaged by a cannonball
	A broken wall surface with many cracks
	A cloth surface with a rose pattern
	A cloth surface with a sunflower pattern
	A torn cloth surface
	An arid land surface
	A stone surface with many cracks
	A broken stone surface with many cracks
	A dragon skin surface with many scales
	A lion fur surface
	A new wooden floor
	An aged wooden surface
	A rusty metal surface with many cracks
	A surface of rusty metal
	A skin surface with a scar mended by needle and thread
	A skin surface with terrible wounds
Geometric Structures	A lip of human (a)
	A human spine (a)
	A mouth of a monster (a)
	A skeleton hand (a)
	A tortoiseshell (a)
	A round snail shell (a)
	An eye of a monster (a)
	A dragon wing (b)
	An angel wing (b)
	An ear of the devil (b)
	A dorsal fin of a fish (c)
	A horn of a dragon (d)
	A goat horn (d)
	A horn of the devil (d)
	A horn of a deer (d)
	An octopus tentacle (d)
	An ox horn (d)
	A royal pauldron (e)
	A beard of a man (e)
	A human ear (f)

## C. Additional Results

### C.1. More Examples of Semantic Coupling

In this subsection, we present more examples of semantic coupling. As shown in Figure S6, directly using SDS for generating a deer horn leads to the generation of the entire deer head, or generating a beard also results in the generation of the nose. Our method significantly alleviates this semantic coupling issue, enabling the generation of high-quality sub-object structures.

### C.2. More Comparison Results

We provide more qualitative results for surface details and geometric structures. Our method can generate higher-quality and more vivid results, as shown in Figure S9 and Figure S10.



Figure S5. More application examples of interactive modeling.

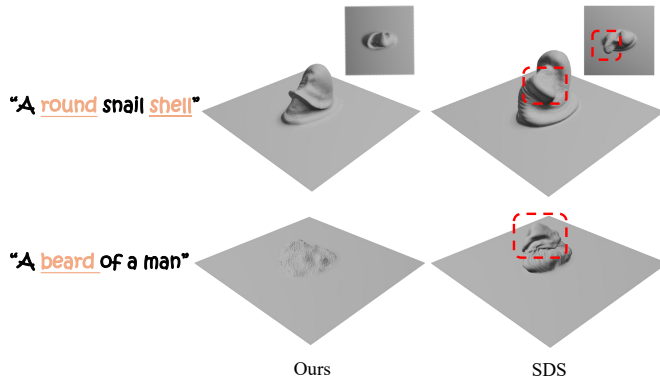


Figure S6. More examples of semantic coupling.

### C.3. More Application Examples

We also provide more examples of sculpted models using the generated VDM brushes. We enable users to interactively use a variety of generated brushes to sculpt diverse and expressive models from a plain shape.

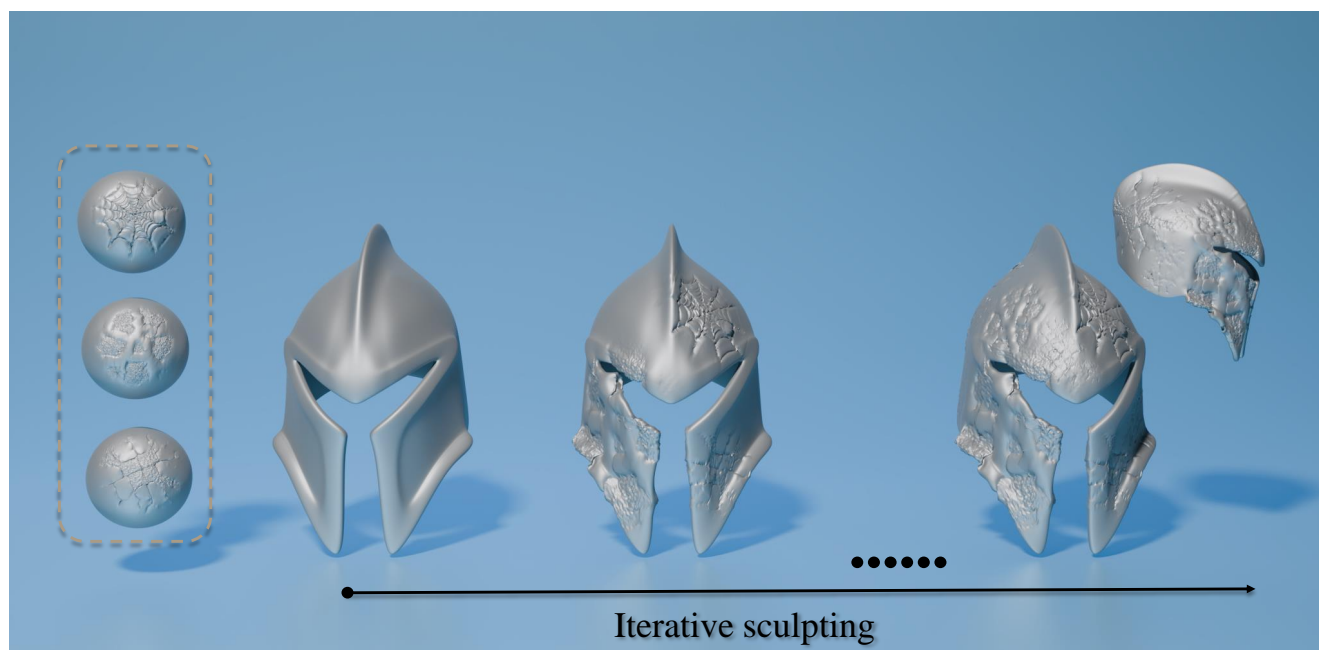


Figure S7. More application examples of mesh stylization on a helmet.

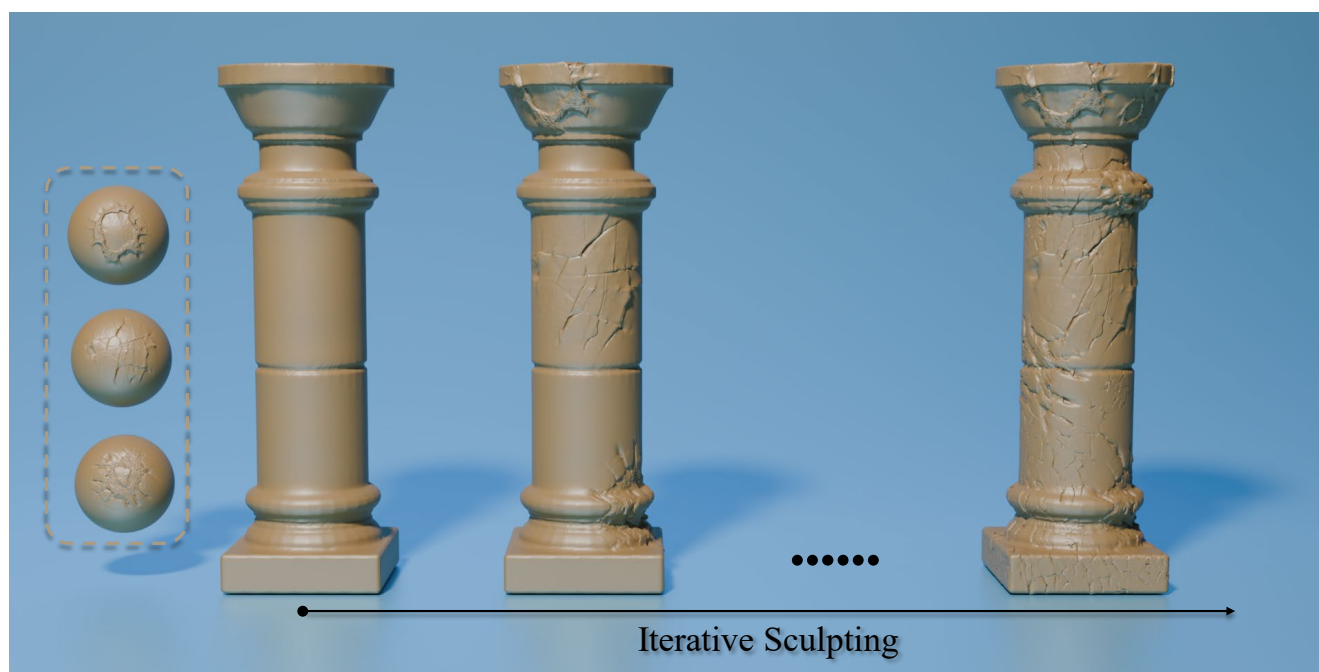


Figure S8. More application examples of mesh stylization on a pillar.



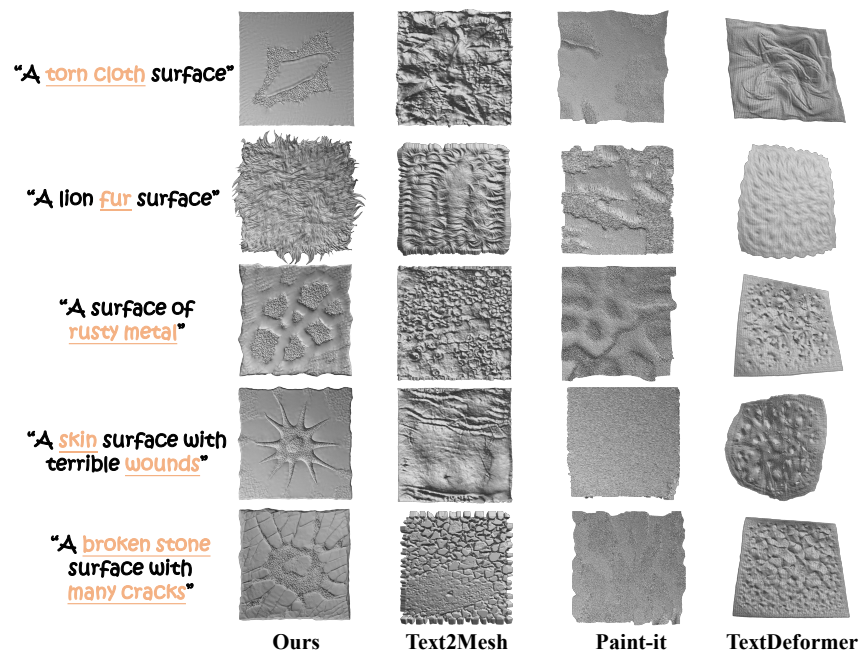


Figure S9. More comparison results of brushes for surface details.

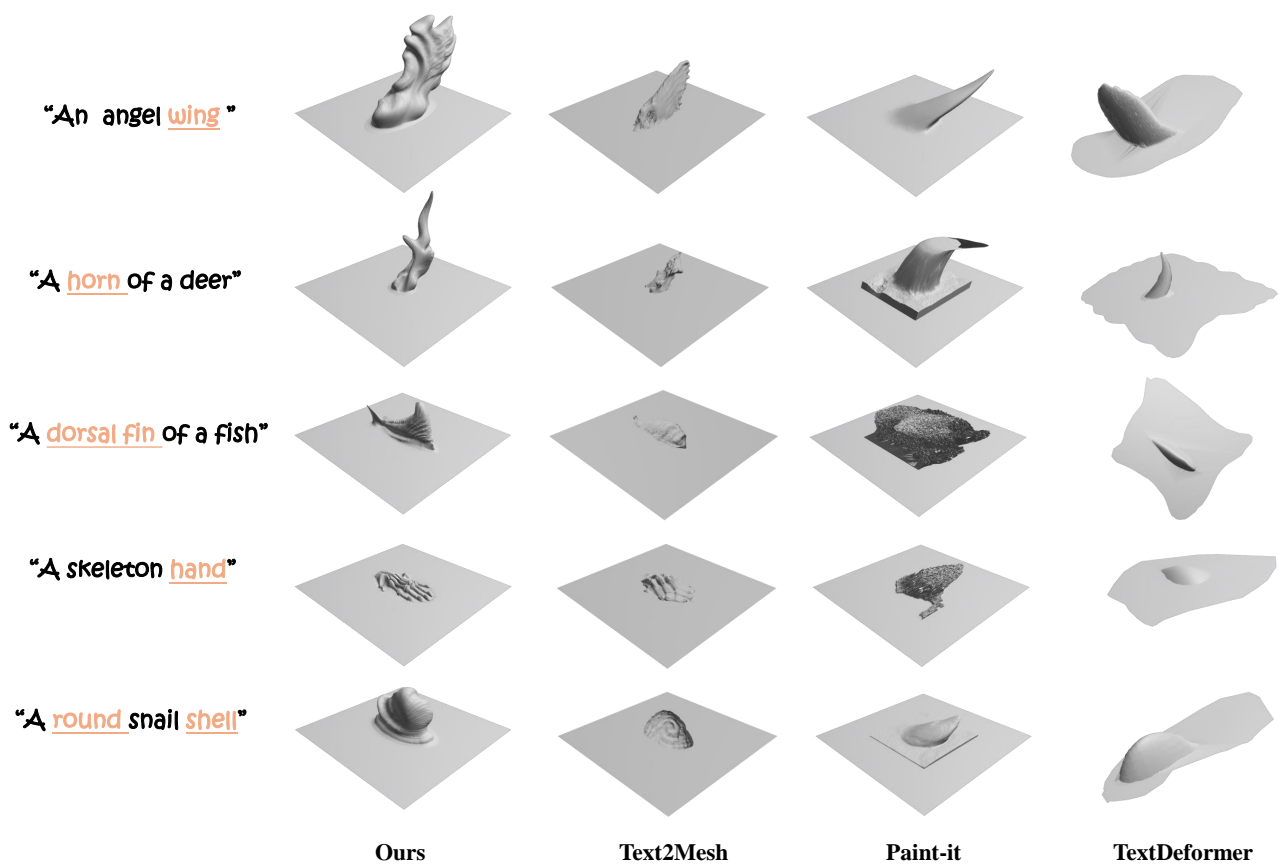


Figure S10. More comparison results of brushes for geometric structures.