

3DMorph: Single-Image-Guided Local 3D Shape Editing and Morphing

Tobias Preintner^{1,2}, Yunfei Deng², Phillip Müller², Sebastian Illing²,
Adrian König², Thomas Bäck¹, Elena Raponi¹, Niki van Stein¹

¹Institute of Advanced Computer Science, Leiden University, Leiden, The Netherlands

²BMW Group, Munich, Germany

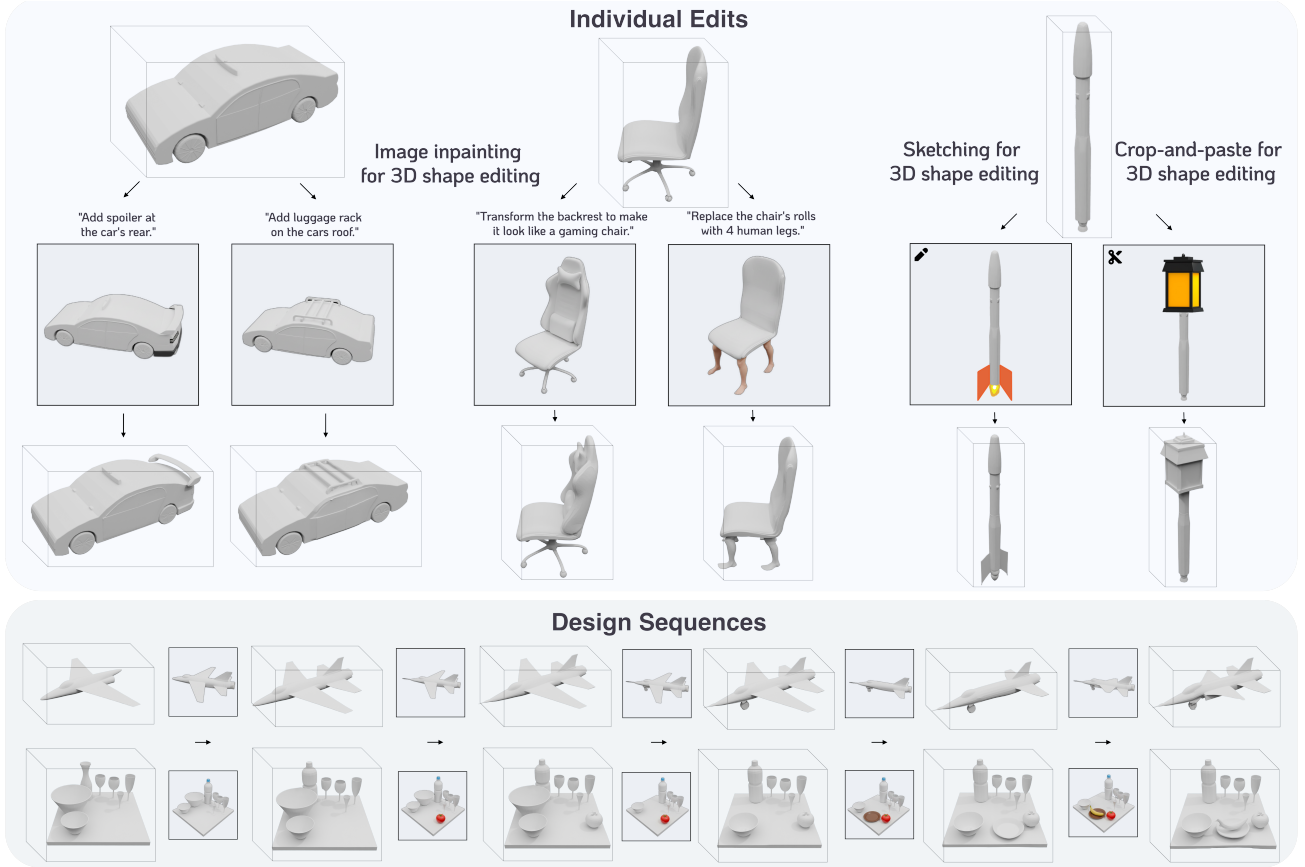


Fig. 1. High-quality 3D editing results from our method. Using rendered and edited images as input (e.g., via inpainting), 3DMorph lifts 2D modifications into 3D while preserving unedited regions. It can be used for both individual edits and multi-step editing sequences.

Abstract—Despite recent progress in 3D generation, intuitive editing of existing shapes remains limited. Unlike images, which benefit from well-established inpainting tools, general 3D objects such as meshes still lack simple and effective methods for local shape editing. Existing approaches are often global, domain-specific, require complex user interaction, or focus on appearance (color and texture) rather than geometry. We introduce 3DMorph, a training-free framework for single-image-guided local 3D shape editing and morphing. Given an edited image showing a desired shape modification, our method automatically localizes the relevant 3D region and transfers 2D modifications to 3D while preserving unmodified areas. 3DMorph also enables intermediate shape generation between the original and edited objects, facilitating design exploration. To benchmark editing quality, we introduce Delta3D, an image-guided local 3D editing benchmark with paired ground-truth edits. Experimental results

show that 3DMorph translates intuitive 2D edits into 3D, outperforming state-of-the-art generative and editing methods.

Index Terms—Shape Editing, Local 3D Editing, Image-Guided 3D Editing, 3D Morphing

I. INTRODUCTION

Meshes are the backbone of industrial and design workflows due to their precision and compatibility with established computer-aided design (CAD) and computer-aided engineering (CAE) toolchains. In practice, meshes used in engineering workflows often originate from CAD models, but even editing only local parts can require re-running the entire mesh-generation pipeline, including modifying the CAD model, re-meshing, and performing manual cleanup.

Moreover, well-parameterized CAD models are often unavailable: scanned or publicly shared assets typically provide only a static mesh (vertices and faces) without any editable parametric structure. This limitation has become more salient with recent advances in 3D generative models [21, 40, 34], which can rapidly produce high-quality meshes suitable as starting points for downstream design. However, directly editing meshes remains challenging. Unlike images, with mature inpainting and editing tools, there is a lack of intuitive methods for local mesh modification.

We introduce 3DMorph, a training-free framework for local 3D shape editing that requires only a single rendered and edited view to drive the modification. Using standard image-editing operations (e.g., inpainting, sketching, or crop-and-paste) to communicate desired local edits, 3DMorph transfers the specified changes back to the mesh while preserving unedited regions.

Beyond the target edit, 3DMorph can generate intermediate meshes along a continuous morph trajectory. This supports design-space exploration, which is common in engineering and design, where multiple alternatives must be compared (e.g., aerodynamic evaluation of spoiler variants). The candidates differ only within the locally edited region, enabling controlled comparison, optimization, and visualization.

Unlike artistic 3D generation, where color and texture dominate, engineering workflows focus on geometry. We therefore introduce Delta3D, a single-image-guided 3D editing benchmark tailored to geometry-centric evaluation. Each sample includes a source mesh, an edited image that defines the local change, and a 3D ground-truth mesh containing the edit.

Our main contributions are:

- A training-free method for intuitive local 3D editing of arbitrary shapes from a single edited view.
- A benchmark dataset with ground-truth local modifications for quantitative 3D editing evaluation.
- A bounding-box estimation module that automatically localizes the 3D edit region from a single edited image.
- A local morphing mechanism that generates intermediate shapes for design exploration and visualization.

Together, these components form a modular framework for single-image-guided 3D editing and morphing. Our code and dataset is available at <https://github.com/toprei/3dmorph>.

II. RELATED WORK

3D Shape Generation. State-of-the-art 3D generation methods produce high-quality geometric assets by adapting diffusion-based approaches [23, 43, 34] and rectified flow transformers [21, 40] to the 3D domain. Although these methods generate high-quality shapes, their applicability in engineering and design is limited, as even minor local edits require a full regeneration, often causing unwanted changes.

3D Shape Editing. To avoid full regeneration and increase editing control, a growing line of work has focused on shape editing. Early approaches learn 3D-language grounding for text-guided editing from object pairs or sequences accompa-

nied by textual difference descriptions [15, 1, 31], but remain sensitive to subtle grammatical and phrasing variations [26].

Other text-guided methods leverage text-image losses to edit 3D objects [30, 12, 7, 22, 6]. Recently, training-free text-guided methods have emerged that leverage multimodal language models [38, 42, 5, 45]. However, textual spatial descriptions are often ambiguous and verbose, hindering precise 3D editing in engineering and design.

To mitigate this issue, sketch-based approaches [3, 2, 18] capture spatial details but demand significant drawing proficiency and spatial reasoning skills. A more intuitive editing approach mixes parts of existing shapes [14, 16], but is limited by available source objects.

Compared to text, images represent intended 3D modifications more precisely and compactly. Some methods [32, 19] use edited images to guide 3D edits, but require additional complex user inputs such as segmentation maps or 3D masks. Only a few methods support single-image 3D editing. [11] formulates shape editing as a conditional reconstruction problem but is limited to edits within the trained canonical view. CMD [20] enables 3D editing and progressive generation.

Beyond these limitations, most existing methods focus on visual appearance (color and texture) rather than geometric quality, largely due to the lack of ground-truth 3D edits [20]. Since color and texture can mask geometric artifacts, geometric precision is often under-emphasized in these works.

To address this, we propose a geometry-focused, training-free, single-image-guided 3D editing method and a corresponding benchmark, demonstrating both the benchmark’s utility and the effectiveness of our approach.

Shape Morphing. Shape morphing smoothly transforms one object into another by generating a sequence of intermediate shapes. Accordingly, shape editing can be regarded as its single-step variant.

Early geometric methods model deformation using continuum mechanics [36], geometric flows [4], splines in shell space [13], or Hamiltonian dynamics [9]. More recent work, such as 4Dform [29], learns continuous velocity fields in Euclidean space, enabling morphing directly on point clouds. In contrast, most recent approaches leverage latent spaces: TextDeformer [12] enables text-driven mesh edits with a 3D diffusion prior, NeuroMorph [10] applies graph convolutions to interpolate local features, and SRIF [33] synthesizes intermediate point clouds and images.

Existing methods perform global morphing that alters the entire mesh. In contrast, our approach deforms only the modified region while preserving the rest, supporting sampling of intermediate variations for efficient comparison, optimization, and design-space exploration.

III. METHODOLOGY

We propose 3DMorph, a training-free method for image-guided local 3D shape editing and morphing. From a single edited image, it transfers local 2D edits to the 3D domain while preserving unaltered regions. Moreover, it supports

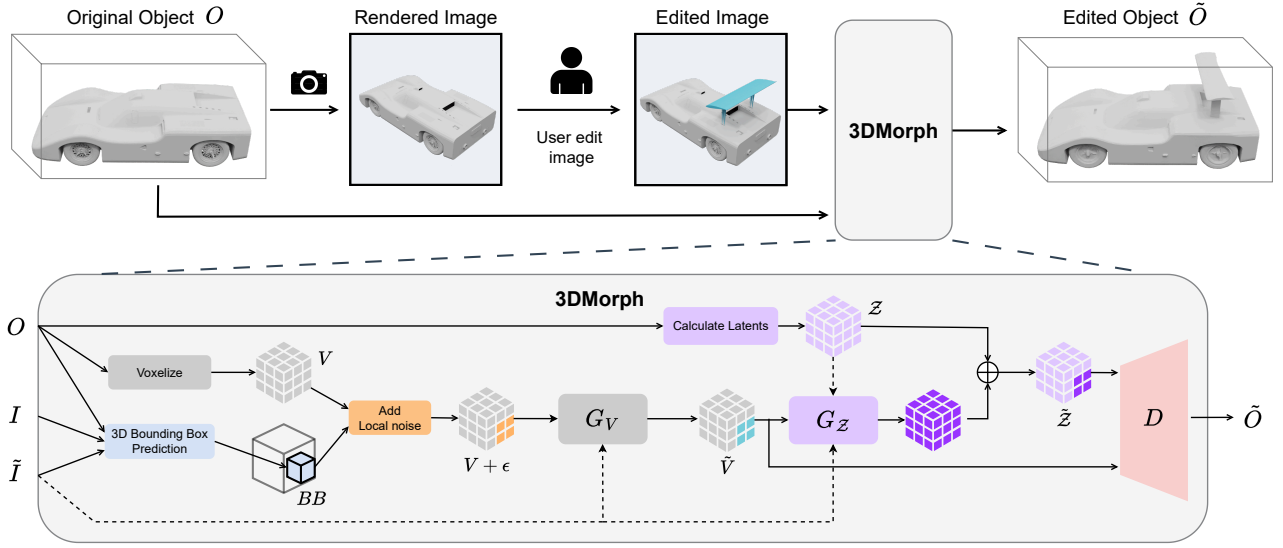


Fig. 2. Overview of 3DMorph. Our method enables intuitive, training-free 3D shape editing guided by 2D image modifications. After a user edits a chosen rendered view, the 2D changes are transferred back to 3D while retaining unedited regions, enabling simple local shape editing and high-quality mesh outputs.

smooth morphing from the original to the edited object through intermediate shape generation.

A. Problem Definition

We formally define a generative model $f(O, \tilde{I})$, that takes as input a 3D object O and an edited image $\tilde{I}$. The image $\tilde{I}$ is obtained by applying local edits to an image I , where I is a rendered view of O . A local edit is defined as a modification confined to a specific region in both 2D and 3D, typically involving the addition or adaptation of specific parts. The goal is to find a model f that transfers the edits from $\tilde{I}$ to a modified 3D object $\tilde{O}$, while preserving the geometry of O in the unaltered regions. $\tilde{I}$ serves solely as a medium to specify shape edits, decoupling the formulation from low-level visual attributes such as color. In this way, intuitive 2D editing techniques, such as inpainting, hand-sketching, or crop-and-paste manipulations, can be leveraged to modify 3D objects. Moreover, the approach should support local 3D morphing from O to $\tilde{O}$ via intermediate 3D objects.

B. Preliminary

Image editing. Image editing tools have evolved for decades, with traditional software manipulating pixels directly or applying user-controlled transformations. However, these tools demand considerable user expertise.

Deep learning-based methods enable local image modifications while preserving surrounding regions. Techniques, such as RePaint [24], Palette [28], Paint-by-Example [41], Stable Diffusion Inpainting [27], ControlNet [44], GeoDrag [25], FLUX.1-Fill [17] and Gemini Flash [8] make local image editing fast, intuitive, and controllable.

Structured latents. 3D objects can generally be decomposed into geometry and semantics. This decomposition can be represented by a coarse voxel structure V and per-voxel

latent features Z , which encode both texture and geometric surface details. Together, they form the structure latent (SLAT) representation, employed by recent generative models [40, 37, 39] to synthesize new 3D assets from images or text.

Trellis [40] utilizes two specialized rectified flow transformers to generate SLAT: one, G_V , for the sparse voxel structure and another, G_Z , for the latent features. The resulting SLAT is then decoded by a decoder D into various 3D formats, including meshes. Although Trellis is primarily designed for generation, we incorporate its generative transformer and decoder architecture into our framework and leverage its SLAT representation for editing. Notably, SLAT can be derived not only through G_V and G_Z , but also deterministically from existing 3D objects by voxelizing them and computing per-voxel latents from projections of multi-view renderings.

C. 3D Shape Editing

Our method enables intuitive, training-free local 3D editing from a single edited rendering. We first detect the modified local region between the original and edited images and project it into 3D to obtain an axis-aligned bounding box. This step is non-trivial and critical in our pipeline. Instead of requiring the user to manually estimate a 3D box, we predict it automatically (see Sec. III-D). Within the bounding box, a coarse voxel grid and latent features are generated by G_V and G_Z , respectively, using the edited view. Outside the bounding box, we retain the original voxel grid and latents, which are deterministically computed from the original object.

Reusing the original voxel structure preserves global geometry, while incorporating the high-quality original latents recovers fine details, including those not visible in the edited view. The resulting hybrid structure, combining original and modified features, is decoded into a high-quality 3D mesh that seamlessly integrates the edit while preserving unmodified re-

gions, subject to the limits of the current SLAT resolution and decoder capacity. This supports long edit sequences without degrading quality in unaffected areas.

A key innovation of this approach is the use of a hard constraint on the voxel structure, in contrast to commonly used flow-consistent conditioning. We describe this design choice in detail in the following subsection.

Conditioning Schedule for Local Inpainting. The objective of local 3D editing is to modify geometry only within the predicted edit region while preserving the original structure and details elsewhere. During reverse sampling, that is, while synthesizing the edited region, this requires enforcing consistency on the unmodified SLAT to prevent drift and ensure fidelity to the original object.

Classical RePaint conditioning re-injects the known region at every step using its flow-consistent value $\mathcal{F}_t(x_0, z)$, preventing covariate shift and matching the model’s training dynamics. However, in our 3D setting this can introduce stochastic variation in the preserved geometry early in the trajectory, which is undesirable when high-fidelity context preservation is required.

To address this, we adopt a two-stage schedule:

- **Stage-1 (voxel prediction / sparse structure):** enforce a hard constraint on known voxels to stabilize global shape.
- **Stage-2 (latent refinement / per-voxel detail):** use flow-consistent conditioning to smoothly merge the new geometry and details with the preserved shape.

$$\Pi_{\text{hard}}(\hat{x}_{t-\Delta}) = (1 - m) \odot x_0 + m \odot \hat{x}_{t-\Delta} \quad (1)$$

$$\Pi_{\text{flow}}(\hat{x}_{t-\Delta}) = (1 - m) \odot \underbrace{\mathcal{F}_{t-\Delta}(x_0, z)}_{\substack{\text{path-consistent} \\ \text{known region}}} + m \odot \hat{x}_{t-\Delta} \quad (2)$$

Here, m denotes the edit mask (in our case, the bounding-box BB), x_0 is the unedited context (SLAT of O), and $z \sim \pi$ is noise sampled from the prior. $\mathcal{F}_t(x_0, z)$ denotes the training-time rectified-flow path that transports the clean context x_0 toward a noise sample z , evaluated at time t , with $F_0(x_0, z) = x_0$ and $F_1(x_0, z) = z$.

In Stage-1, the context voxels are kept fixed, ensuring that preserved regions remain unchanged while the model infers a coarse modified voxel structure. In Stage-2 the focus shifts to making the edited region match the original shape and blend in smoothly. We set the known region to its state along the flow-matching path at time t . This matches the training setup, yields seamless transitions between edited and preserved regions.

Algorithm. Figure 2 illustrates our approach, which is detailed in Algorithm 1. Given the original object O , its rendered image I , and an edited version $\tilde{I}$, our method first applies the 3D bounding box prediction module described in Sec. III-D (line 1). After voxelizing O (line 2), we add noise within the bounding box region to enable new voxel generation during denoising (line 3). This is possible as the stage-1 latent space is also structured, which allows us to inject noise in a

way that aligns with the original voxel space. In combination with $\tilde{I}$, the partially noised voxel grid $\tilde{V}$ is input to G_V to generate the edited voxel grid $\tilde{V}$ (line 4). Following the approach of [40], the object latents $\mathcal{Z}$ are deterministically computed from multiple rendered views for all L voxels, where L denotes the number of surface voxels in the grid (line 5). Together with $\tilde{V}$ and $\tilde{I}$, these provide context to $G_{\mathcal{Z}}$, which generates new latents $\tilde{\mathcal{Z}}$ for the edited voxel grid $\tilde{V}$, including the modified region (line 6). To preserve quality in the unmodified regions, the new latents are applied only within the edit region defined by BB , while the original latents are retained outside BB (line 7). Finally, $\tilde{V}$ and $\tilde{\mathcal{Z}}$ are decoded to produce the edited object $\tilde{O}$ (line 8).

Algorithm 1: Local 3D Editing with 3DMorph

Input : Original object O , rendered image I ,
modified image $\tilde{I}$
Output: Edited object $\tilde{O}$

```

// Generate Local Voxels
1  $BB \leftarrow \text{getBB}(O, I, \tilde{I}) ;$  //  $BB \in \{0, 1\}^{64^3}$ 
2  $V \leftarrow \text{voxelize}(O) ;$  //  $V \in \{0, 1\}^{64^3}$ 
3  $\dot{V} \leftarrow V + \epsilon ;$  //  $\epsilon \sim \mathcal{N}(0, I) \odot BB$ 
4  $\tilde{V} \leftarrow G_V(\dot{V}, \tilde{I}) ;$ 

// Generate Local Latents
5  $\mathcal{Z} \leftarrow \text{getLatents}(O, V) ;$  //  $\mathcal{Z} \in \mathbb{R}^{L \times d}$ 
6  $\tilde{\mathcal{Z}} \leftarrow G_{\mathcal{Z}}(\tilde{V}, \tilde{I}, \mathcal{Z}) ;$ 
7  $\tilde{\mathcal{Z}} \leftarrow BB \cdot \tilde{\mathcal{Z}} + (1 - BB) \cdot \mathcal{Z} ;$ 
8  $\tilde{O} \leftarrow D(\tilde{V}, \tilde{\mathcal{Z}}) ;$  // decode

```

D. Bounding Box Prediction

Our method detects structural changes by first identifying 2D differences between image pairs and lifting them into 3D. SSIM-based differencing yields 2D change boxes, which we anchor in 3D by projecting their centers into the voxel grid. This procedure assumes accurate camera poses and sufficient visibility of the edited region.

Box extents are estimated by projecting the 2D box into 3D using the camera parameters, the 2D box width and height, and a pixel-to-metric scale factor derived from the top view, which maps pixel distances to 3D object-space units. Instead of merging all boxes, we compute a tight axis-aligned bounding box that encloses them, yielding a single 3D box in voxel space that localizes the edit region in voxel space.

E. Local 3D Morphing

Our morphing method leverages the SLAT difference between the two objects to generate intermediate shapes between the original object O and the edited object $\tilde{O}$. Voxels inside the modification bounding box are sequentially activated in batches, ordered by their Euclidean distance to O , though arbitrary distance functions and activation policies can also be applied. After each batch activation, the corresponding SLAT is decoded into a mesh. This procedure can be interpreted

as a form of geometric and latent-space morphing. Repeating this process yields a series of intermediate objects between O and $\tilde{O}$, enabling visualization, design-space exploration, and creative discovery.

IV. EXPERIMENTS

In this section, we quantitatively and qualitatively evaluate our single-image-guided 3D editing framework for local shape edits using our Delta3D benchmark. We further analyze the impact of the bounding-box prediction module through an ablation study and present results on inpainted image edits as 3D editing guidance, as well as on local shape morphing.

In summary, 3DMorph consistently outperforms alternative methods both quantitatively and qualitatively, leverages object context more effectively than the baseline editing approach, remains robust with both predicted and ground-truth bounding boxes, and can handle individual edits and multi-step editing sequences.

A. Delta3D benchmark

No existing benchmark provides paired 3D ground truth for precise local edits; we therefore introduce Delta3D to fill this gap. Our Delta3D benchmark is built on the Fusion 360 Gallery Assembly dataset [35], a diverse collection of user-submitted CAD models composed of multi-part 3D objects. Object pairs $(O, \tilde{O})$ are generated by removing a single part from an assembly. Pairs are chosen such that both the original object O and the ground-truth edit $\tilde{O}$ are meaningful objects on their own, resulting in 74 carefully chosen object pairs.

For each object pair, we rotate 360° around the ground-truth object in 30° increments, simulating 12 edited images $\tilde{I}$. To ensure that the local modification is visible, we compare each image pair $(I, \tilde{I})$ using the SSIM to identify 2D difference regions, from which bounding boxes are computed as described in Section III-D. Views in which the modified part is fully occluded are excluded, resulting in a final set of 864 evaluation samples. This removes unsolvable cases while retaining challenging conditions such as oblique angles, heavy occlusions, and complex depth discontinuities.

B. Experimental Setup

We evaluate 3DMorph on our Delta3D benchmark dataset and compare it against state-of-the-art image-to-3D generation methods, including Trellis [40], Hunyuan3D [34], and TripoSG [21]. In addition to generative baselines, we include a 3D editing method built on Trellis and RePaint [24], hereafter referred to as Trellis-RePaint, which uses purely flow-consistent conditioning (see Sec. III-C). This approach was briefly mentioned in [40] but not evaluated in detail. Since it requires an edit region as input, we pair it with our bounding box prediction module to enable a fair comparison with 3DMorph.

We also conduct an ablation in which both 3DMorph and Trellis-RePaint are given a perfect bounding box that exactly encloses the region of difference between O and $\tilde{O}$. This setup (i) enables a direct comparison under identical idealized

conditions and (ii) quantifies the effect of bounding box prediction on performance. All methods are evaluated on the full benchmark dataset.

C. Metrics

We evaluate the geometric quality of the edited 3D object by comparing it with the ground truth. Chamfer Distance (CD) and Hausdorff distance (HD) quantify differences between the generated object $\tilde{O}$ and ground truth $\tilde{O}$. Here, $d(\bar{o}, \tilde{o})$, denotes the Euclidean distance between surface points $\bar{o} \in \tilde{O}$ and $\tilde{o} \in \tilde{O}$. For each object, we sample 20,000 points.

$$CD(\tilde{O}, \tilde{O}) = \frac{1}{2|\tilde{O}|} \sum_{\bar{o} \in \tilde{O}} \min_{\tilde{o} \in \tilde{O}} d(\bar{o}, \tilde{o}) + \frac{1}{2|\tilde{O}|} \sum_{\tilde{o} \in \tilde{O}} \min_{\bar{o} \in \tilde{O}} d(\tilde{o}, \bar{o}) \quad (3)$$

$$HD(\tilde{O}, \tilde{O}) = \max\{\sup_{\bar{o} \in \tilde{O}} \inf_{\tilde{o} \in \tilde{O}} d(\bar{o}, \tilde{o}), \sup_{\tilde{o} \in \tilde{O}} \inf_{\bar{o} \in \tilde{O}} d(\tilde{o}, \bar{o})\} \quad (4)$$

While CD considers only surfaces, intersection-based metrics capture volume overlap. Thus, we also report Intersection over Union (IoU) and Dice coefficient (DSC).

$$IoU(\tilde{O}, \tilde{O}) = \frac{|\tilde{O} \cap \tilde{O}|}{|\tilde{O} \cup \tilde{O}|} \quad DSC(\tilde{O}, \tilde{O}) = \frac{2|\tilde{O} \cap \tilde{O}|}{|\tilde{O}| + |\tilde{O}|} \quad (5) \quad (6)$$

Unless stated otherwise, metrics are computed on the entire object, covering both preserved and modified regions.

We complement the 3D evaluation with image similarity metrics suited for spatial analysis, including Multi-Scale SSIM (MS-SSIM), Feature Similarity Index (FSIM), and Gradient Magnitude Similarity Deviation (GMSD). For all image similarity evaluations, both the generated and ground-truth objects are rendered from the same viewing angle as $\tilde{I}$ for comparison. These traditional metrics emphasize structural properties, such as edges and contours, rather than appearance features like brightness or texture. Deep feature-based metrics such as FID or LPIPS primarily capture appearance differences, which are not the focus of our evaluation.

D. Results

Quantitative evaluation. Tab. I shows that our method consistently and statistically significantly outperforms prior methods across all metrics. The closest competitor is Trellis-RePaint, which, despite using the same predicted bounding boxes, performs notably worse than 3DMorph. Compared to Trellis-RePaint, 3DMorph reduces CD by 19% and increases IoU by 3 points. Interestingly, Trellis-MV, which has access to twelve hypothetically available multi-view images of the modified object, still fails to match 3DMorph’s performance.

Bounding box prediction. We evaluate bounding box quality using Intersection over Ground Truth (IoGT) between predicted and ground-truth boxes. IoGT prioritizes target coverage over penalizing excess area, reflecting the goal of avoiding missing edited regions. Ground-truth boxes occupy only 4.1% of the voxel volume, while the predicted boxes remain compact at 8.2% and achieve an average IoGT of 58%. The larger predicted boxes stem from assumptions about unknown depth, with modest overestimation helping to mitigate this ambiguity.

TABLE I
QUANTITATIVE COMPARISON. **BOLD** MARKS THE BEST RESULTS, UNDERLINED DENOTES A STATISTICALLY SIGNIFICANT GAP (PAIRED T-TEST, 5% SIGNIFICANCE LEVEL). † INDICATES HYPOTHETICAL AVAILABILITY OF MULTI-VIEW IMAGES OF THE MODIFIED OBJECT.

Method	CD↓	HD↓	IoU↑(%)	DSC↑(%)	MS-SSIM↑	FSIM↑	GMSD↓
TripoSG [21]	0.0454	0.126	66.2	77.5	0.721	0.675	0.218
Trellis [40]	0.0458	0.123	74.0	84.1	0.771	0.690	0.206
Trellis-MV† [40]	0.0342	0.096	77.9	86.9	0.781	0.696	0.202
Hunyuan3D [34]	0.0263	0.074	82.2	89.6	0.843	0.752	0.174
Trellis-RePaint [40, 24]	0.0129	0.036	85.2	91.0	0.861	0.795	0.162
3DMorph	0.0105	0.029	88.2	93.3	0.877	0.806	0.156

TABLE II
BOUNDING-BOX ABLATION USING GROUND-TRUTH BOUNDING BOXES.

Method	GT-BB	CD↓	HD↓	IoU↑(%)	DSC↑(%)	MS-SSIM↑	FSIM↑	GMSD↓
Trellis-RePaint	—	0.0129	0.036	85.2	91.0	0.861	0.795	0.162
	✓	0.0102	0.028	88.6	93.4	0.899	0.827	0.148
3DMorph	—	0.0105	0.029	88.2	93.3	0.877	0.806	0.156
	✓	0.0089	0.024	91.0	95.0	0.909	0.836	0.144

TABLE III
EDIT-REGION EVALUATION WITHIN THE TARGET REGION.

Method	GT-BB	CD↓	HD↓	IoU↑(%)	DSC↑(%)
Trellis-RePaint	—	0.0333	0.107	56.7	70.5
3DMorph	—	0.0305	0.097	59.6	72.9
Trellis-RePaint	✓	0.0268	0.073	61.7	75.0
3DMorph	✓	0.0230	0.063	65.0	77.8

Bounding box ablation. We assess the impact of the bounding box prediction module through an ablation study on 3DMorph and Trellis-RePaint, replacing predicted boxes with ground-truth ones obtained by comparing O and $\hat{O}$.

Table II shows that this substitution improves performance for both methods, but 3DMorph remains superior to Trellis-RePaint. Notably, 3DMorph without access to ground-truth boxes nearly matches the performance of Trellis-RePaint with ground-truth box access. This demonstrates the robustness of 3DMorph when using predicted boxes and its competitiveness even against methods with perfect bounding box information.

Edit-region evaluation. Our Delta3D benchmark enables quantitative 3D evaluation of edits, filling a gap in the assessment of 3D editing methods noted in prior work [20]. Tab. III compares the editing quality of 3DMorph with Trellis-RePaint, demonstrating consistent superiority of our method when using either predicted or ground-truth bounding boxes.

Image edits for 3D shape editing. Fig. 1 demonstrates our method in its intended application scenario: lifting single-image edits into 3D. Image edits can be flexibly created by the user, including inpainting, hand-sketching, or crop-and-paste operations using parts from other images. Inpainting is performed using FLUX.1 Fill [17] and Gemini 2.5 Flash Image [8]. Our method supports diverse 3D modifications, including adding, removing, or altering parts, as well as composing multiple edits sequentially.

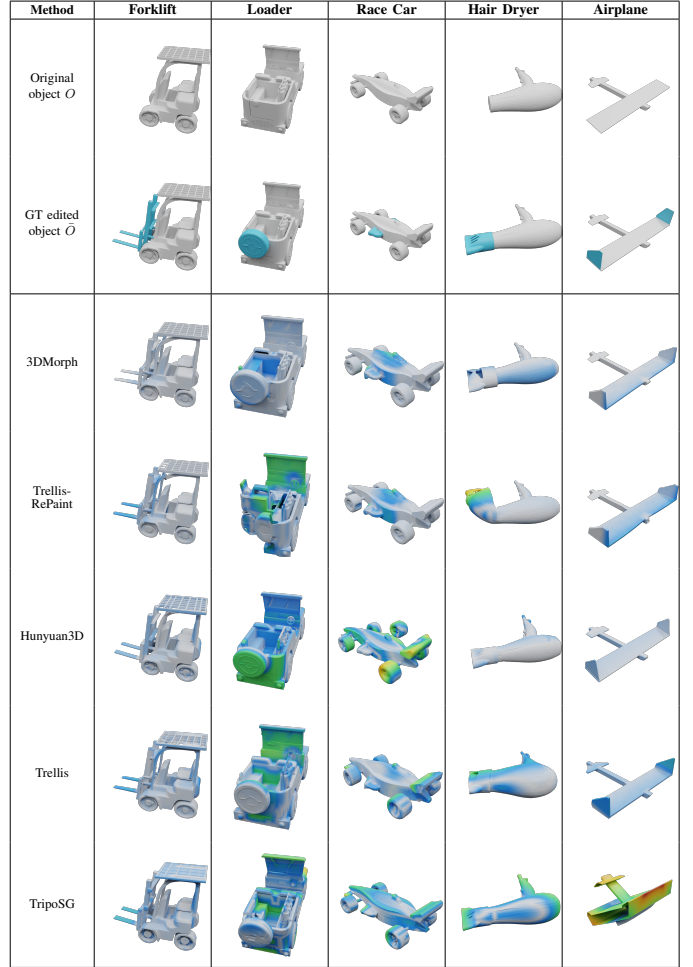


Fig. 3. Qualitative 3D editing results. The first two rows show the original object and the ground-truth edit, where blue denotes the desired local edit. Differences between the edited object and $\hat{O}$ are visualized, with colors ranging from blue (small deviation) to red (large deviation).

Qualitative results. Fig. 3 qualitatively demonstrates the superiority of our method in editing object shapes using only a single image as guidance. We additionally visualize differences between generated edits and the ground truth $\hat{O}$. This color-coded visualization provides only a rough indication, as it assumes geometric correspondence via surface-distance alignment, which may break down for severe deformations where unrelated surfaces lie close in space.

Single-image generative approaches (last three rows) often

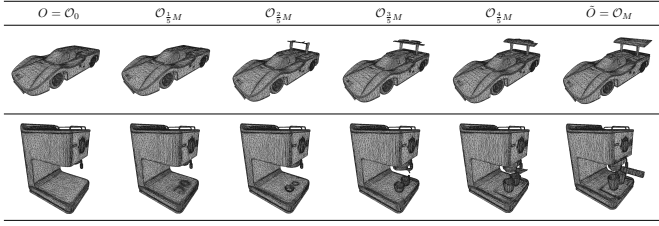


Fig. 4. Morphing results. The left and right columns show the original object O and the edited object $\hat{O}$, respectively. The four middle columns visualize shapes at 20%, 40%, 60%, and 80% along a series of M morphing steps.

introduce unintended edits, such as closing the forklift roof or airplane rear wing, adding or removing components (e.g., a gas bottle in the loader), misinterpreting geometry (e.g., treating the race-car side wing as a fifth wheel), or failing to preserve the global shape (e.g., deforming the hair dryer).

Despite using the same bounding box estimate, Trellis-RePaint either fails entirely (e.g., loader, hair dryer) or struggles near bounding box boundaries, introducing unintended changes such as deforming the race-car front wheel or the airplane nose. In contrast, our method preserves global structure, avoids unwanted changes, and produces high-quality 3D edits.

Morphing results. Fig. 4 qualitatively illustrates local morphing results consisting of a series of M objects $\{O_i\}_{i=0}^M$. We show four intermediate shapes, along with the original object O and the edited object $\hat{O}$ as the first and last, respectively. For space efficiency, intermediate shapes are displayed at evenly spaced interpolation steps. Wireframe visualizations highlight our ability to generate high-quality intermediate 3D objects with changes confined to a single local region

V. CONCLUSION

In this paper, we present 3DMorph, a method for single-image-guided local 3D shape editing and morphing, operating within a confined region, preserving unaltered areas with high quality, and requiring only a single edited image to specify the desired change. Moreover, we introduce Delta3D, the first geometric benchmark for this task with 3D editing ground truth, enabling geometric evaluation of the entire edited object. Our quantitative and qualitative results demonstrate that our method outperforms state-of-the-art generative and editing approaches on local 3D editing tasks.

A current limitation of this work lies in the bounding box prediction module, which can fail when an unfavorable viewing angle is chosen. However, in practical applications, users typically perform edits from an appropriate viewing angle to better express their intended modifications. Moreover, the current level of fine-grained geometric detail is primarily determined by the resolution and capacity of the underlying SLAT decoder. This is a natural property of generative pipelines and is expected to improve as higher-resolution decoders become available, as demonstrated by recent work [39].

In future work, we plan to expand Delta3D and further improve our 2D-to-3D bounding box prediction module, a crucial component of single-image editing.

REFERENCES

- [1] Panos Achlioptas et al. “ShapeTalk: A language dataset and framework for 3d shape edits and deformations”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 12685–12694.
- [2] Hmrishav Bandyopadhyay et al. “Doodle your 3d: From abstract freehand sketches to precise 3d shapes”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 9795–9805.
- [3] Alexandre Binniger et al. “SENS: Part-Aware Sketch-based Implicit Neural Shape Modeling”. In: *Computer Graphics Forum*. Vol. 43. 2. Wiley Online Library. 2024, e15015.
- [4] Christopher Brandt, Christoph von Tycowicz, and Klaus Hildebrandt. “Geometric flows of curves in shape space for processing motion of deformable objects”. In: *Computer Graphics Forum*. Vol. 35. 2. Wiley Online Library. 2016, pp. 295–305.
- [5] Weiwei Cai et al. “Native 3D Editing with Full Attention”. In: *arXiv preprint arXiv:2511.17501* (2025).
- [6] Minghao Chen et al. “Partgen: Part-level 3d generation and reconstruction with multi-view diffusion models”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 5881–5892.
- [7] Minghao Chen et al. “Shap-editor: Instruction-guided latent 3d editing in seconds”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 26456–26466.
- [8] Gheorghe Comanici et al. “Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities”. In: *arXiv preprint arXiv:2507.06261* (2025).
- [9] Marvin Eisenberger and Daniel Cremers. “Hamiltonian dynamics for real-world shape interpolation”. In: *European conference on computer vision*. Springer. 2020, pp. 179–196.
- [10] Marvin Eisenberger et al. “Neuromorph: Unsupervised shape interpolation and correspondence in one go”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 7473–7483.
- [11] Will Gao et al. *3D Mesh Editing using Masked LRMs*. 2024. arXiv: 2412.08641 [cs.CV]. URL: <https://arxiv.org/abs/2412.08641>.
- [12] William Gao et al. “Textdeformer: Geometry manipulation using text guidance”. In: *ACM SIGGRAPH 2023 conference proceedings*. 2023, pp. 1–11.
- [13] Behrend Heeren et al. “Splines in the space of shells”. In: *Computer Graphics Forum*. Vol. 35. 5. Wiley Online Library. 2016, pp. 111–120.
- [14] Amir Hertz et al. “Spaghetti: Editing implicit shapes through part aware generation”. In: *ACM Transactions on Graphics (TOG)* 41.4 (2022), pp. 1–20.
- [15] Ian Huang et al. “LADIS: Language disentanglement for 3D shape editing”. In: *arXiv preprint arXiv:2212.05011* (2022).

- [16] Juil Koo et al. “Salad: Part-level latent diffusion for 3d shape generation and manipulation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 14441–14451.
- [17] Black Forest Labs. *FLUX*. <https://github.com/black-forest-labs/flux>. 2024.
- [18] Haoxuan Li et al. “MeshPad: Interactive Sketch-Conditioned Artist-Designed Mesh Generation and Editing”. In: *arXiv preprint arXiv:2503.01425* (2025).
- [19] Lin Li et al. “VoxHammer: Training-Free Precise and Coherent 3D Editing in Native 3D Space”. In: *arXiv preprint arXiv:2508.19247* (2025).
- [20] Peng Li et al. “CMD: Controllable Multiview Diffusion for 3D Editing and Progressive Generation”. In: *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 2025, pp. 1–10.
- [21] Yangguang Li et al. “Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models”. In: *arXiv preprint arXiv:2502.06608* (2025).
- [22] Yuhan Li et al. “Focaldreamer: Text-driven 3d editing via focal-fusion assembly”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 38. 4. 2024, pp. 3279–3287.
- [23] Xiaoxiao Long et al. “Wonder3D: Single Image to 3D using Cross-Domain Diffusion”. In: *arXiv preprint arXiv:2310.15008* (2023).
- [24] Andreas Lugmayr et al. “Repaint: Inpainting using denoising diffusion probabilistic models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 11461–11471.
- [25] Phillip Mueller et al. “GeoDiffusion: A Training-Free Framework for Accurate 3D Geometric Conditioning in Image Generation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2025, pp. 6374–6384.
- [26] Tobias Preintner et al. “Why Are You Wrong? Counterfactual Explanations for Language Grounding with 3D Objects”. In: *2025 International Joint Conference on Neural Networks (IJCNN)*. 2025. DOI: 10.1109/IJCNN64981.2025.11227256.
- [27] Robin Rombach et al. “High-Resolution Image Synthesis With Latent Diffusion Models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 10684–10695.
- [28] Chitwan Saharia et al. “Palette: Image-to-image diffusion models”. In: *ACM SIGGRAPH 2022 conference proceedings*. 2022, pp. 1–10.
- [29] Lu Sang et al. “4Deform: Neural Surface Deformation for Robust Shape Interpolation”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 6542–6551.
- [30] Etai Sella et al. “Vox-e: Text-guided voxel editing of 3d objects”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 430–440.
- [31] Habib Slim and Mohamed Elhoseiny. “Shapewalk: Compositional shape editing through language-guided chains”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 22574–22583.
- [32] Hao Su et al. “3DFaceSculptor: A Common Framework for Image-Guided 3D Face Deformation”. In: *IEEE Transactions on Visualization & Computer Graphics* 01 (Aug. 5555), pp. 1–18. ISSN: 1941-0506. DOI: 10.1109/TVCG.2025.3596482. URL: <https://doi.ieeecomputersociety.org/10.1109/TVCG.2025.3596482>.
- [33] Mingze Sun et al. “Srf: Semantic shape registration empowered by diffusion-based image morphing and flow estimation”. In: *SIGGRAPH Asia 2024 Conference Papers*. 2024, pp. 1–11.
- [34] Tencent Hunyuan3D Team. *Hunyuan3D 2.1: From Images to High-Fidelity 3D Assets with Production-Ready PBR Material*. 2025. arXiv: 2506.15442 [cs.CV].
- [35] Karl DD Willis et al. “Joinable: Learning bottom-up assembly of parametric cad joints”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 15849–15860.
- [36] Benedikt Wirth et al. “A continuum mechanical approach to geodesics in shape space”. In: *International Journal of Computer Vision* 93.3 (2011), pp. 293–318.
- [37] Tianhao Wu et al. “Amodal3r: Amodal 3d reconstruction from occluded 2d images”. In: *arXiv preprint arXiv:2503.13439* (2025).
- [38] Ruihao Xia, Yang Tang, and Pan Zhou. “Towards Scalable and Consistent 3D Editing”. In: *arXiv preprint arXiv:2510.02994* (2025).
- [39] Jianfeng Xiang et al. “Native and Compact Structured Latents for 3D Generation”. In: *arXiv preprint arXiv:2512.14692* (2025).
- [40] Jianfeng Xiang et al. “Structured 3d latents for scalable and versatile 3d generation”. In: *Proceedings of the Computer Vision and Pattern Recognition Conference*. 2025, pp. 21469–21480.
- [41] Binxin Yang et al. “Paint by example: Exemplar-based image editing with diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 18381–18391.
- [42] Junliang Ye et al. “NANO3D: A Training-Free Approach for Efficient 3D Editing Without Masks”. In: *arXiv preprint arXiv:2510.15019* (2025).
- [43] Taoran Yi et al. “GaussianDreamer: Fast Generation from Text to 3D Gaussians by Bridging 2D and 3D Diffusion Models”. In: *CVPR*. 2024.
- [44] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. “Adding conditional control to text-to-image diffusion models”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2023, pp. 3836–3847.
- [45] Zhenglin Zhou et al. “AnchorFlow: Training-Free 3D Editing via Latent Anchor-Aligned Flows”. In: *arXiv preprint arXiv:2511.22357* (2025).