TopoEdit: Fast Post-Optimization Editing of Topology Optimized Structures

Hongrui Chen, Josephine V. Carstensen, Faez Ahmed

Massachusetts Institute of Technology, Cambridge, MA, 02139,

Abstract

Despite topology optimization producing high-performance structures, late-stage localized revisions remain brittle: direct density-space edits (e.g., warping pixels, inserting holes, swapping infill) can sever load paths and sharply degrade compliance, while re-running optimization is slow and may drift toward a qualitatively different design. We present TopoEdit, a fast post-optimization editor that demonstrates how structured latent embeddings from a pre-trained topology foundation model (OAT) can be repurposed as an interface for physics-aware engineering edits. Given an optimized topology, TopoEdit encodes it into OAT's spatial latent, applies partial noising to preserve instance identity while increasing editability, and injects user intent through an edit-then-denoise diffusion pipeline. We instantiate three edit operators: drag-based topology warping with boundary-condition–consistent conditioning updates, shell–infill lattice replacement using a lattice-anchored reference latent with updated volume-fraction conditioning, and late-stage no-design region enforcement via masked latent overwrite followed by diffusion-based recovery. A consistency-preserving guided DDIM procedure localizes changes while allowing global structural adaptation; multiple candidates can be sampled and selected using a compliance-aware criterion, with optional short SIMP refinement for warps. Across diverse case studies and large edit sweeps, TopoEdit produces intention-aligned modifications that better preserve mechanical performance and avoid catastrophic failure modes compared to direct density-space edits, while generating edited candidates in sub-second diffusion time per sample.

Keywords:
Topology Optimization, Machine Learning

1. Introduction

Topology optimization has become a powerful approach in engineering design, allowing the development of lightweight, high-performance structures through the optimal distribution of material within a specified design domain. However, the complex shape of the solutions generated by topology optimization often poses significant challenges for practical design workflows, where engineers must interpret, trust, and revise the optimized result to satisfy downstream considerations and design intent. In these settings, even small local changes, such as adjusting a joint location, replacing an interior region with an infill, or enforcing a last-minute no-design region, typically require re-running optimization, which is time-consuming and can drift toward a qualitatively different design. Localized editing offers a promising alternative: rather than restarting the full solver or regenerating a new sample, a designer should be able to "drag" structural features toward a target configuration while still leveraging the model's learned physical priors to maintain mechanical performance. Despite this potential, there is a lack of methods for topology editing that support fast, localized, and physics-aware modifications of existing topology-optimized designs.

Current approaches that target user control in topology optimization can be broadly categorized into those that rely on restrictive problem reformulations, post-processing techniques, or human-in-the-loop optimization schemes. Methods that modify the optimization formulation, such as adding constraints or tuning feature-size/length-scale controls, can incorporate certain preferences, but they still require iterative solves and careful parameter choices, limiting responsiveness in topology editing settings [1]. Similarly, post-processing techniques can modify the geometry after optimization, but they are typically not context-aware and do not explicitly account for structural performance under the applied loads and boundary conditions during the edit [2]. Interactive topology optimization frameworks that incorporate user intent can fill a practical gap, yet they primarily steer ongoing optimization via region-of-interest feature-size changes or pattern guidance, rather than enabling targeted, direct edits to an already-optimized design [3, 4, 5]. Data-driven generative models accelerate topology generation and can generalize across diverse conditions, but existing pipelines are typically designed to generate a solution from a specification rather than to edit a specific solution instance while preserving its identity [6, 7]. In this work, we build upon these advances and describe a method that takes into account post-optimization editability and fast, localized control, while preserving physics-aware structural behavior.

A key to achieving our goal lies in the ability to perform localized edits without discarding the physical information encoded in the original optimized topology. Directly editing the density field in image space, such as warping pixels or insert-

ing holes or infill, is ineffective. These operations can easily disrupt load paths or disconnect critical boundary-condition attachments, leading to significant compliance degradation (for example, direct warp in Figure 2 (b)). Moreover, treating editing as iterative optimization is often slow, as it requires the user to intervene during the optimization and then still perform many small update steps to achieve the intended semantic change [4]. Additionally, enforcing practical edit constraints, such as preserving similarity, respecting boundary conditions near the edit region, and maintaining volume-fraction consistency, these constraints and objectives further complicate successful editing. Topology optimization produces high-performing density fields, but downstream engineering requirements may not be fully captured in the original optimization formulation. Topology warp addresses geometric alignment, clearance changes, or weld angle constraint by allowing structural members to be rearranged while retaining the reference design. Lattice replacement addresses a different class of downstream goals, where an engineer wants to replace a thick solid interior with a patterned infill for design exploration or manufacturability-oriented redesign. No-design region enforcement addresses late-stage passive regions, where material must be removed from a specified area. Together, we bring the existing latent diffusion, partial noising, reference guidance, best-of-N selection, to post-optimization structural edits where the three operators are intended to demonstrate how different engineering revisions can be expressed as latent edits and then denoised under the original or updated problem specification.
Our main contributions are:

- TopoEdit: We show that the structured latent embeddings of a pre-trained topology foundation model (OAT latent diffusion) can be repurposed for engineering post-optimization edits, enabling fast, instance-preserving modifications via partial noising and an edit-then-denoise pipeline.

- Unified latent edit operators: We introduce three practical edit operators—topology warping, lattice infill replacement, and no-design region enforcement, implemented as latent interventions with edit-specific conditioning updates.

- Physics-aware guidance and refinement: We propose a consistency-preserving guided DDIM denoising procedure (with localized reference guidance for warps), plus compliance-aware best-of-N candidate selection, and an optional short SIMP post-processing step to refine edited results.

- Empirical validation of edit quality and efficiency: Through case studies and quantitative evaluation, we show TopoEdit achieves localized, intention-aligned edits while better preserving mechanical performance than direct density-space edits, with sub-second diffusion sampling in the reported setup.

2. Previous Work

Our review focuses on data-driven topology optimization, editing in topology optimization, and diffusion models.

Data-driven topology optimization. Topology optimization distributes material within a design domain to satisfy performance objectives under constraints such as volume fraction. Early works include homogenization-based formulations [8], shape/topology perspectives beyond homogenization [9], Solid Isotropic Materials with Penalization (SIMP) interpolation [10], and level-set approaches [11, 12]. These methods remain widely used and supported by compact educational implementations [13]; however, their slow and iterative nature motivates faster surrogates for high-throughput exploration and interaction.

Data-driven methods skip optimization by learning a direct mapping from problem setup to a near-optimal topology. Machine learning methods include conditional GANs and transfer learning [14, 15], supervised iteration-free predictors [16, 17], hybrids that couple learned components with optimization structure [18], and architectures designed to improve generalization and extend beyond 2D settings [19, 20, 21, 22]. More recent work also explores implicit representations to reduce fixed-resolution dependence [23]. While these approaches substantially accelerate generation, most are not designed for instance-preserving edits. Editing one optimized design rather than generating a new acceptable design can be difficult with aformentioned data-driven method.

Generative modeling for topology progressed from GANs conditioned on physical fields [7] to resolution-free implicit generators [6], and most recently to diffusion-based models [24, 25, 26] and finally foundation modeling (OAT) that targets shape- and resolution-free structural topology distributions [27]. OAT is trained on 2.2 million optimized structures. These structures consist of 2 million unique boundary conditions, varying numbers of loads, and volume fractions. As a latent diffusion model, OAT significantly reduces the mean compliance by up to 90% compared to the best prior models and achieves sub-second inference time. Additionally, OAT demonstrates excellent generalizability, performing well on out-of-distribution datasets like heatsink optimization. To edit topology-optimized results, the edit moves the result outside the original distribution of the trained manifold. This motivates us to leverage the model's generalizability to generate edited topologies. The versatile latent representation of OAT aligns well with recent advancements in latent-based image editing using latent diffusion models.

Editing in topology optimization. Raw topology optimization results can meet structural requirements, but can be difficult to turn into a manufacturable design. Thus, a line of work focuses on making topology optimization editable by exposing handles for designer intent and localized control. One class of methods introduces explicit feature-size, thickness, and porosity controls to shape member widths and manufacturable detail [3, 4, 28, 29, 30]. Another class enables appearance-guided

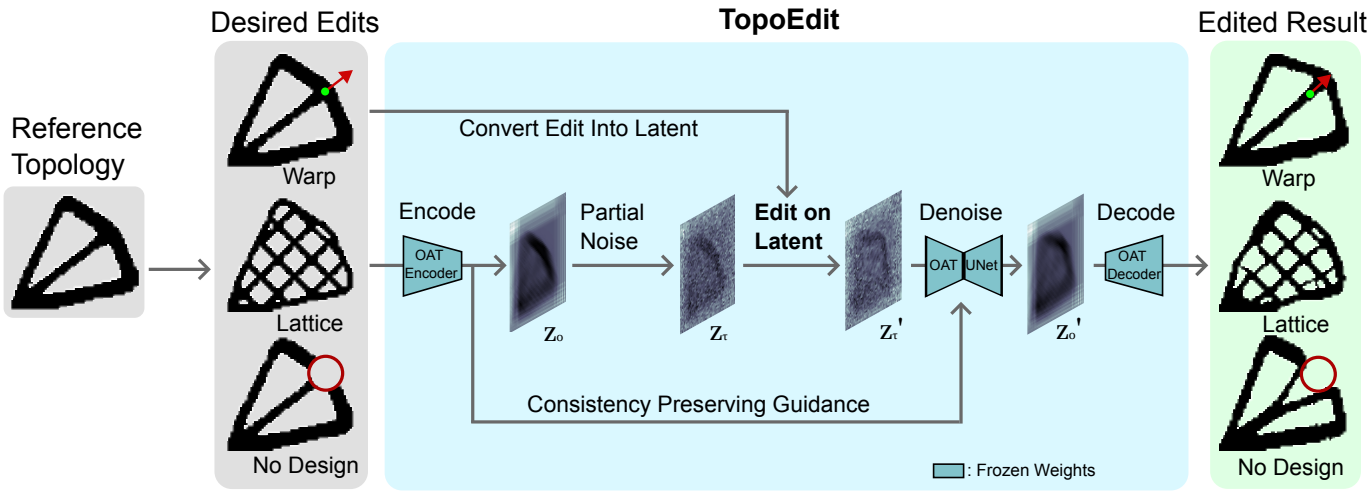


Figure 1: TopoEdit uses a pretrained OAT with frozen weights. Given a reference topology and a user-specified desired edit (topology warp, lattice infill replacement, or no-design region enforcement), TopoEdit encodes the input into OAT's structured latent, applies partial noising, and converts the edit into a corresponding latent-space intervention to produce an edited initialization. Starting from the edited latent, we run diffusion denoising with consistency-preserving guidance to recover an edited clean latent that preserves the instance identity away from the edit while allowing localized structural adaptation. Finally, the edited latent is decoded to yield the edited topology.

design, where users steer infill motifs or geometric textures through drawn patterns, pattern libraries, or appearance constraints, extending from 2D layouts to patterned shells and multi-pattern trade-offs [5, 31, 32, 33, 34, 35]. In parallel, interactive frameworks incorporate subjective preferences and sketches as inputs to guide exploration and promote diverse candidate structures [36, 37, 38, 39, 40]. Beyond topology optimization specific formulations, patch-based correspondence and synthesis in graphics provides a conceptual toolkit for local replacement and pattern transfer [41, 42, 43, 44], and related ideas have been adapted to inject geometric priors into topology optimization via neural style transfer [45].

Despite this progress, many topology optimization editing mechanisms remain constrained by how they incorporate intent. Methods that encode edits as additional constraints or controls are typically physics-aware, but they still require reformulating and re-solving a global, iterative optimization, which is slow and can drift away from the original topology. Human-in-the-loop approaches similarly improve controllability, but they primarily steer an ongoing solve and do not directly support post-hoc edits of an already-optimized design without continued iterations. At the other extreme, post-processing geometry operations such as warps, hole insertion, or infill replacement can be fast, but they are not inherently aware of the boundary conditions; even localized changes may sever load paths or detach boundary-condition attachments, leading to large compliance degradation. Together, these limitations motivate an edit mechanism that is localized while remaining better aligned with the structural semantics of optimized designs.

Diffusion models and edits. Diffusion models learn data distributions by inverting a noising process [46, 47, 48], with practical accelerations and control via implicit sampling and guidance [49, 50] and strong empirical image-generation performance [51]. Latent diffusion improves efficiency by performing diffusion in a compressed latent space [52], with few-step sampling further enabled by distillation and consistency formulations [53, 54, 55]. Building on this, diffusion-based editing has been demonstrated for semantic/text-guided manipulation [56, 57, 58, 59, 60, 61] and for interactive drag/point edits that propagate sparse constraints through learned features [62, 63, 64, 65, 66, 67, 68]. A consistent theme is that editing in a learned latent/feature space can preserve instance identity while enabling coherent local geometric change, often more robustly than direct pixel-space warping.

Motivated by latent editing in image-focused diffusion models, we investigate whether analogous principles can enable instance-preserving edits of topology-optimized designs. OAT is particularly suitable as a foundation because it models a broad distribution of optimized topologies through a latent diffusion pipeline [27], allowing edits to be expressed compactly in latent space while remaining compatible with diverse shapes and resolutions.

3. Proposed Method

Our method performs topology edits by manipulating the latent vectors of a diffusion model. We first encode the optimized density field into the latent space used by OAT (Section 3.1), then apply partial noising so the latent remains close to the original design manifold while introducing controllable stochasticity for editability. Given this partially noised state, we inject the user's edit in an edit-then-denoise pipeline (Section 3.2). We then run guided denoising with a target latent that encodes the intended post-edit structure, using this reference to enforce consistency with the desired outcome while leveraging the UNet's locality bias to keep edits predominantly local (Section 3.3),

and finally decode the resulting latents back to density fields, with an optional short SIMP refinement step for the topology warp (Section 3.4).

The common idea behind the three operators is to first express the engineering edit in the model domain, where it can be specified as a geometric transform, a modified reference topology, or a prescribed mask. We then map this model-domain edit to OAT's structured latent by applying the corresponding spatial transform to the latent, encoding the edited reference topology, or overwriting the affected latent region. When the edit changes the problem specification, such as the support/force coordinates or the volume fraction, we update the conditioning so that denoising is performed under a consistent specification. Thus, a new edit operator can be formulated by identifying which model-domain quantity is changed, how that change should be represented in the latent space, and which conditioning variables need to be updated during denoising.

3.1. Diffusion Model

We build our topology editing pipeline on top of the pretrained OAT latent diffusion model [27]. OAT is particularly well-suited for topology editing because it compresses a topology of arbitrary aspect ratio and resolution into a fixed, structured spatial latent and models the space of feasible topologies as a conditional probability distribution, enabling fast sampling of multiple post-edit candidates by noising and denoising in latent space. Here, we provide a brief overview of the architecture of OAT.

Neural-field autoencoder. Let $T \in [0,1]^{H\times W}$ denote an optimized density field on a rectangular design domain. OAT first maps T into a fixed-resolution latent z using a neural-field autoencoder (which we refer to as the NFAE):

$$z = E(T), \qquad \phi = D(z), \qquad \tilde{T} = R(\phi, c, s). \tag{1}$$

Here, $E(\cdot)$ is an encoder, $D(\cdot)$ is a decoder producing a feature tensor ϕ, and $R(\cdot)$ is a neural renderer that reconstructs the topology at arbitrary resolution using (i) spatial coordinates c (pixel/element centers) and (ii) cell/pixel sizes s. In OAT, variable-resolution/shape inputs are padded and resized to a fixed resolution for encoding, while the renderer enables decoding back to the native (or any desired) resolution. The autoencoder is trained with an ℓ_1 reconstruction objective,

$$\mathcal{L}_{\text{AE}} = \|\tilde{T} - T_{\text{GT}}\|_1, \tag{2}$$

where T_{GT} denotes the ground-truth topology and $\tilde{T}$ is its reconstruction.

Problem representation for conditional generation. Each topology optimization instance is defined by the domain shape, loads, fixtures, and a target volume fraction. We denote the raw problem specification as

$$\hat{P} = \Big(S_{\text{boundary}}, S_{\text{force}}, \text{VF}, n, a\Big), \tag{3}$$

where S_{boundary} is a point set describing boundary-condition locations and their directional constraints, S_{force} is a point set describing load locations and their 2D force vectors, $\text{VF} \in (0,1)$ is the target volume fraction, n encodes cell/pixel size (resolution), and $a \in \mathbb{R}^2$ encodes the domain aspect ratio. Following OAT, the order-invariant point sets $(S_{\text{boundary}}, S_{\text{force}})$ are embedded with Boundary Point Order-invariant MLP (BPOM)-style encoders, and the remaining scalars are embedded with MLPs, producing a fixed-size conditioning vector

$$\begin{aligned} P = \text{concat}\Big(&\text{BPOM}_b(S_{\text{boundary}}), \text{BPOM}_f(S_{\text{force}}), \\ &\text{MLP}_{\text{vf}}(\text{VF}), \text{MLP}_{\text{cell}}(s), \text{MLP}_{\text{ratio}}(a)\Big). \end{aligned} \tag{4}$$

In our later editing operators, we will occasionally need to reconstitute or update parts of $\hat{P}$ (VF and the point coordinates in $S_{\text{boundary}}/S_{\text{force}}$) so that conditioning remains consistent with the edited design-domain geometry.

Latent Diffusion. Let $z_0 = E(T)$ denote the clean latent encoding of a topology T. A forward diffusion process corrupts z_0 by gradually adding Gaussian noise:

$$z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \tag{5}$$

where $\bar{\alpha}_t$ is a pre-defined noise schedule decreasing with timestep $t \in \{0, \dots, T\}$. OAT trains a conditional UNet to predict the velocity parameterization

$$v = \sqrt{\bar{\alpha}_t}\epsilon - \sqrt{1-\bar{\alpha}_t} z_0, \tag{6}$$

given the noisy latent z_t, timestep t, and the problem specification $\hat{P}$. The diffusion training objective is

$$\mathcal{L}_{\text{LDM}} = \mathbb{E}_{z_0,t,\epsilon}\Big[\Big\|v - v_\theta(z_t, t \mid \hat{P})\Big\|_2^2\Big], \tag{7}$$

where $v_\theta(\cdot)$ is the conditional UNet predictor.

Partial noising and denoising for editable sampling. A core operation in our editor is partial noising: instead of initializing sampling from pure noise ($t = t_{total}$), we start from an intermediate noise level $t = \tau$ with $\tau \ll t_{total}$. Given an input topology T we compute its latent $z_0 = E(T)$ and sample a partially noised latent z_τ via Eq. (5). We then apply an edit operator to z_τ (or to a clean reference latent that is subsequently noised) to obtain $z'_\tau$and run a deterministic DDIM-style reverse process from τ down to 0 to obtain one or more edited latents z'_0, which are decoded back to density fields by the NFAE renderer.

Partial noising preserves the instance identity encoded in z_0 (global layout, major load-path "semantics," and coarse structural features) because z_τ remains correlated with z_0 when $\bar{\alpha}_\tau$ is not too small. At the same time, injecting moderate noise introduces controllable stochastic degrees of freedom that make the representation more pliable to localized edits and allow sampling a spectrum of mechanically acceptable post-edit outcomes by varying ϵ. In contrast, starting from full noise would largely discard the original instance information, turning editing into full regeneration rather than an instance-preserving modification.

3.2. Latent Edits

We define three edit operators that act on the latent representation of an existing topology instance. Let $z_0 = E(T)$ be the clean latent code of the input density field T, and let z_τ denote its partially noised state obtained by applying the forward process in Eq. (5) at an intermediate noise level τ. Each edit produces an edited latent initialization z'_τ and, when needed, an updated problem specification $\hat{P}'$ whose boundary-condition geometry is consistent with the edit. Guided denoising then maps $z'_\tau \mapsto z'_0$, and we decode z'_0 with the NFAE to obtain the edited topology.

3.2.1. Topology Warp

Gaussian coordinate displacement. A drag edit is specified by a handle location $\mathbf{h} \in [0,1]^2$, a displacement $\mathbf{\Delta} \in \mathbb{R}^2$, and an influence radius $\sigma > 0$. We define a smooth displacement field

$$\mathbf{u}(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{h}\|_2^2}{2\sigma^2}\right)\mathbf{\Delta}, \qquad \mathbf{x} \in [0,1]^2. \tag{8}$$

Using destination-to-source (inverse-map) sampling, the induced warp operator $\mathcal{W}$ acting on an image-like field f is

$$[\mathcal{W}(f)](\mathbf{x}_{\text{dst}}) = f(\mathbf{x}_{\text{dst}} - \mathbf{u}(\mathbf{x}_{\text{dst}})), \tag{9}$$

implemented with continuous interpolation.

Latent warp and BC-consistent conditioning update. We apply the same geometric warp to the partially noised latent:

$$z'_\tau = \mathcal{W}(z_\tau). \tag{10}$$

To keep conditioning consistent with the deformed geometry, we warp the point-set components of $\hat{P}$ (Eq. (4)), i.e., the support and force locations. Given a source point $\mathbf{p}_{\text{src}}$ (from S_{boundary} or S_{force}), the corresponding destination point $\mathbf{p}_{\text{dst}}$ satisfies the inverse-map relation

$$\mathbf{p}_{\text{src}} = \mathbf{p}_{\text{dst}} - \mathbf{u}(\mathbf{p}_{\text{dst}}). \tag{11}$$

We solve Eq. (11) via relaxed fixed-point iteration

$$\mathbf{p}^{(k+1)} = (1-\rho)\mathbf{p}^{(k)} + \rho\left(\mathbf{p}_{\text{src}} + \mathbf{u}(\mathbf{p}^{(k)})\right), \qquad \mathbf{p}^{(0)} = \mathbf{p}_{\text{src}}, \tag{12}$$

with relaxation $\rho \in (0,1]$. Applying this to all points yields a warped specification $\hat{P}_{\text{warp}}$. We use $\hat{P}' = \hat{P}_{\text{warp}}$ during denoising.

3.2.2. Lattice Infill Replacement

Lattice replacement differs from warping in that we first construct an edited reference topology in image space, then use its latent as the anchor for guided denoising.

Shell-infill decomposition and lattice reference. We define a replaceable infill mask $M_{\text{infill}} \in [0,1]^{H\times W}$ by identifying pixels that lie sufficiently deep inside solid regions. Let $d(p)$ be a thickness proxy evaluated for pixels p inside the solid, and let t_{shell} be a user-specified shell thickness. We set

$$M_{\text{infill}}(p) = [d(p) > t_{\text{shell}}]. \tag{13}$$

Given a parametric lattice pattern $L \in \{0,1\}^{H\times W}$ such as a grid or cross-shaped lattice controlled by pitch and member thickness. We form a lattice-replaced reference

$$T_{\text{lat}} = T \odot (1 - M_{\text{infill}}) + L \odot M_{\text{infill}}. \tag{14}$$

Because T_{lat} changes the material volume, we recompute the induced volume fraction

$$\text{VF}_{\text{lat}} = \frac{1}{HW}\sum_p T_{\text{lat}}(p), \tag{15}$$

and update the volume-fraction entry in $\hat{P}$ accordingly, yielding $\hat{P}_{\text{lat}}$.

Latent initialization. We encode the edited reference and use it to define the reference latent

$$z_{\text{ref}} = E(T_{\text{lat}}). \tag{16}$$

We then obtain z_τ by noising z_{ref} using Eq. (5) at $t = \tau$, and set $z'_\tau = z_\tau$ with no additional latent-space operator is applied. During denoising we use $\hat{P}' = \hat{P}_{\text{lat}}$.

3.2.3. No-design Region Enforcement

We enforce a circular no-design region by overwriting the latent inside a prescribed disk, while allowing the diffusion model to restore the mechanical performance through diffusion.

Hole mask and reference latent.. The no-design region is specified by a center $\mathbf{h} \in [0,1]^2$ and radius $\sigma > 0$. Let $\mathbf{x}(p) \in [0,1]^2$ denote the normalized coordinate of pixel p. We define the binary hole mask

$$M_{\text{hole}}(p) = [\|\mathbf{x}(p) - \mathbf{h}\|_2 \le \sigma], \tag{17}$$

and the directly holed reference topology

$$T_{\text{hole}} = T \odot (1 - M_{\text{hole}}). \tag{18}$$

Encoding T_{hole} gives the reference latent

$$z_{\text{ref}} = E(T_{\text{hole}}). \tag{19}$$

Latent overwrite inside the no design region.. We start from z_τ obtained by noising the original instance latent z_0 (Eq. (5) at $t = \tau$). Let M^z_{hole} denote the hole mask evaluated on the latent grid (using the same normalized $(\mathbf{h}, \sigma)$). We enforce the void by overwriting the latent within the hole:

$$z'_\tau = z_\tau \odot (1 - M^z_{\text{hole}}) + z_{\text{void}} M^z_{\text{hole}}, \tag{20}$$

where z_{void} is a calibrated constant representing a "void" latent value in the autoencoder space. We keep the boundary-condition geometry unchanged and set $\hat{P}' = \hat{P}$.

3.3. Guidance and Denoising

All edits share a common guided sampling procedure in the latent space. We first describe the reference guidance that biases the reverse trajectory toward a target latent, and then describe the denoising update (DDIM-style reverse step) driven by OAT's UNet. The three edit types differ only in the choice of conditioning $\hat{P}'$ and reference latent z_{ref}; among them, only topology warp additionally uses a non-constant spatial weight map to localize the effect.

Reference guidance.. At a reverse-time step t (starting from $t = \tau$ down to 0), the current latent state is z_t. We define an ℓ_2 reference loss on the estimated clean latent $z'_0(z_t, t)$:

$$\mathcal{L}_{\text{ref}}(z_t) = \left\| z'_0(z_t, t) - z_{\text{ref}} \right\|_2^2, \tag{21}$$

and apply a gradient step

$$z_t \leftarrow z_t - s\nabla_{z_t}\mathcal{L}_{\text{ref}}(z_t), \tag{22}$$

where s is a guidance step size.

Edit-specific choices of z_{ref}. The reference latent for guidance helps maintain consistency between the edited and the original topology. Therefore, we select a reference latent that is close to the reference topology while still allowing the network to incorporate the edit within the region of interest.

- Topology warp: we set $z_{\text{ref}} = z_0$ to preserve the original instance identity away from the edit.
- Lattice infill replacement: we set $z_{\text{ref}} = E(T_{\text{lat}})$, anchoring sampling to the lattice-replaced reference while allowing BC-aware structural adaptations.
- No-design region: we define $z_{\text{ref}} = E(T_{\text{hole}})$, so the reference latent explicitly encodes the desired void geometry.

For topology warp, the global loss in Eq. (21) relies on the latent z_0 from the reference topology. Using it directly unmasked would penalize deviation from z_0 at every latent location, pulling it towards the unedited reference. We therefore apply guidance only outside a neighborhood of the dragged handle via a spatial weight map on the latent grid. Let $\mathbf{h}_\star$ be the achieved handle location obtained by warping $\mathbf{h}$ with the point-warp solver in Eq. (12). For latent-grid coordinate $\mathbf{x}$, define

$$m(\mathbf{x}) = \text{sigmoid}\left(\frac{\|\mathbf{x} - \mathbf{h}_\star\|_2 - r_{\text{out}}}{k}\right), \tag{23}$$

so that $m \approx 0$ near $\mathbf{h}_\star$ (no guidance) and $m \approx 1$ far away (guidance on). The guided loss for topology warp becomes

$$\mathcal{L}_{\text{warp}}(z_t) = \frac{\left\| (z'_0(z_t, t) - z_{\text{ref}}) \odot m \right\|_2^2}{\langle m \rangle + \varepsilon}, \tag{24}$$

and Eq. (22) is applied with $\mathcal{L}_{\text{ref}}$ replaced by $\mathcal{L}_{\text{warp}}$.

DDIM-style denoising. Given the guidance-adjusted latent z'_τ, we apply a conditional DDIM reverse step driven by OAT under the conditioning of $\hat{P}'$. This step acts as a learned physics-aware prior that projects the edited initialization back toward the manifold of feasible topologies consistent with the updated problem specification. We estimate the clean latent by

$$z'_0(z_t, t) = \sqrt{\bar{\alpha}_t} z_t - \sqrt{1 - \bar{\alpha}_t} v_\theta(z_t, t \mid \hat{P}'), \tag{25}$$

and form the corresponding noise estimate

$$\epsilon(z_t, t) = \sqrt{1 - \bar{\alpha}_t} z_t + \sqrt{\bar{\alpha}_t} v_\theta(z_t, t \mid \hat{P}'). \tag{26}$$

Parameter	Warp	Lattice	No Design
DDIM Total Steps (t_{total})	100	200	100
DDIM Partial Steps (τ)	20	20	25
Guidance Step Size (s)	1000	1000	1000

Table 1: Hyperparameter configuration for the three topology editing tasks. The ratio between partial τ and total steps t_{total} determines the noise level added to the latent vector. For lattice infill, we use a smaller partial-noise ratio τ/t_{total} to keep denoising in a refinement regime that preserves the reference lattice's periodic member layout while permitting small boundary condition-aware corrections.

We then advance one reverse step along the deterministic DDIM trajectory:

$$z_{t-1} = \sqrt{\bar{\alpha}_{t-1}} z'_0(z_t, t) + \sqrt{1 - \bar{\alpha}_{t-1}} \epsilon(z_t, t). \tag{27}$$

Repeating the sequence from $t = \tau$ down to 0 yields the final edited latent, which is decoded by the NFAE renderer to produce the edited topology.

The hyperparameter for the three types of edit is summarized in Table 1.

3.4. SIMP Post-processing

We optionally refine edited designs with a short run of density-based topology optimization using the classical SIMP formulation. Here the design variable is the density field $T \in [0, 1]^{H \times W}$, where $T(p)$ denotes the material density at pixel/element p (1 = solid, 0 = void). Structural stiffness is interpolated from T using SIMP penalization:

$$\mathbf{K}(T) = \sum_p \Big(E_{\min} + T(p)^p (E_0 - E_{\min})\Big)\mathbf{K}_p, \tag{28}$$

where $\mathbf{K}_p$ is the element stiffness matrix for unit modulus, E_0 is the solid Young's modulus, $E_{\min} \ll E_0$ is a small void modulus to avoid singular stiffness, and $p = 3$ is the penalization exponent that discourages intermediate densities.

In our setting, SIMP is used purely as a lightweight post-processing stage rather than a full optimization from scratch. Instead of initializing from a homogeneous field, we warm start from the diffusion-edited topology: we set the initial density to the decoded result $\hat{T}$ (obtained by decoding the final edited latent) and apply only a few iterations of SIMP updates using an optimality-criteria scheme under the same boundary conditions and a prescribed volume fraction VF (as specified in $\hat{P}'$). This short refinement sharpens boundaries, repairs minor local inconsistencies introduced by sampling, and improves compliance while preserving the intended edit. We apply this post-processing only for the topology warp setting, as direct lattice replacement and no design region benchmark could lead to exploding compliance due to missing boundary conditions and severing load members.

4. Results

In this section, we set up experiments to demonstrate the aforementioned topology warp, lattice infill replacement, and late-stage no-design region enforcement with latent-based

Algorithm 1 Latent-based topology editing

Encode latent: $z_0 \leftarrow E(T)$
Add partial noise: sample $\epsilon \sim \mathcal{N}(0, I)$ and set $z_\tau \leftarrow q_\tau(z_0, \epsilon)$
Initialize $z'_\tau \leftarrow z_\tau,\ \hat{P}' \leftarrow \hat{P},\ z_{\text{ref}} \leftarrow z_0$
Latent edit:
if warp then
    Define a displacement field $\Delta(\mathbf{x})$ from user handle/drag parameters
    Warp the noised latent: $z'_\tau \leftarrow \mathcal{W}(z_\tau; \Delta)$
    Update conditioning to match the warped geometry: $\hat{P}' \leftarrow \text{WarpBC}(\hat{P}; \Delta)$
else if lattice infill then
    Form a lattice-replaced reference topology T_{lat} from $(T,\ M_{\text{infill}},\ L)$
    Update conditioning (e.g., volume fraction) to obtain $\hat{P}_{\text{lat}}$ and set $\hat{P}' \leftarrow \hat{P}_{\text{lat}}$
    Convert the replaced reference to a latent target: $z_{\text{ref}} \leftarrow E(T_{\text{lat}})$
    Re-initialize by partial noising: $z'_\tau \leftarrow q_\tau(z_{\text{ref}}, \epsilon)$
else if no-design region then
    Construct a hole mask M_{hole} from center $\mathbf{h}$ and radius σ
    Define a holed reference topology T_{hole} and set $z_{\text{ref}} \leftarrow E(T_{\text{hole}})$
    Map the hole mask to the latent grid (M^z_{hole}) and overwrite inside:
        $z'_\tau \leftarrow \text{OverwriteVoid}(z'_\tau, M^z_{\text{hole}}, z_{\text{void}})$
end if
Denoising: for $t = \tau, \ldots, 1$ do
    Optionally apply reference guidance toward z_{ref} (masked/weighted for locality)
    Take one DDIM reverse step using the UNet: $z_{t-1} \leftarrow \text{DDIMStep}(z_t, t, \hat{P}')$
Obtain edited latent $z'_0 \leftarrow z_0$ (final denoised state)
Decode structure: $\phi' \leftarrow D(z'_0); \quad T' \leftarrow R(\phi', c, s)$
Optional SIMP post-processing: run K refinement steps on T' to obtain $T^\star$

method. For the topologies, we select from the training and testing datasets of OAT. We perform the experiment using a PC with RTX Pro 6000 graphics card with 96G of VRAM. Since the diffusion model is stochastic, we generated 64 latent edit results for each of the subsequent edit operations for the best-of-N analysis.

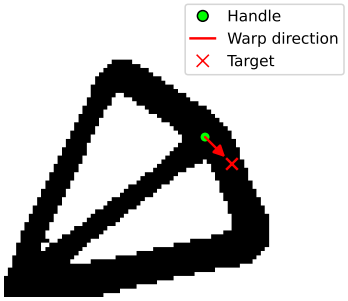
4.1. Topology Warp

Topology warp is motivated by post-optimization situations in which an engineer needs to relocate or realign structural members after an optimized design has already been generated. For example, a joint may need to be moved away from an interfering component, aligned with an assembly interface, or adjusted so that connected members meet at a desired welding angle. In these cases, a useful edit should not only deform the density field locally, but should also allow neighboring structural members to adapt while retaining the overall reference design.

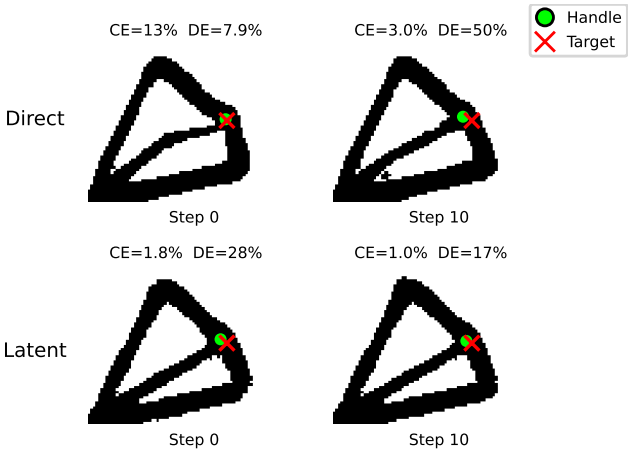
The OAT dataset contains entirely random boundary conditions and volume fractions. We filter the dataset based on member thickness and remove examples whose estimated thickness exceeds 10 in pixel or element units. We revisit the thicker geometries in the next subsection on lattice replacement, where replacing an interior region remains meaningful even when members are thick. After filtering, we select edit points from joints detected on the medial-axis skeletonization. This is a morphological method of finding the skeleton of an image, which is the collection of all points equidistant from the boundary. For each selected joint, we apply eight warp directions with equal angular spacing. We select a total of 122 topologies and 2208 warp operations across the filtered training set and 16 topologies and 123 warp operations across the filtered testing set.

We focus on joint points because they provide a measurable number to evaluate edit performance. For each edited topology, we recompute the medial axis, extract skeleton junctions, and measure the distance from the intended target location to the closest detected joint. This makes it possible to quantify whether the edit moved a joint toward the desired location. For both direct warp and latent warp, we monitor compliance and the closest-joint distance across post-processing steps. We treat step 0 as the immediate output of the edit and track the metrics through the subsequent SIMP post-processing iterations. The warp operation itself, both when applied directly to the topology and when applied as a latent warp, is too fast to be reliably registered. For the latent-based edit, a single diffusion sample takes around 0.58 seconds on the GPU. We then apply 10 steps of SIMP post-processing on the CPU, which takes 1.13 seconds for both the direct and latent cases.

Figure 2 shows a single warp operation and compares direct and latent edits across post-processing steps. For the direct warp, applying the deformation in density space causes a strong local distortion while keeping the joint displacement aligned with the intended direction, so the distance error can be small. At the same time, this local distortion bends straight members and increases compliance error. As SIMP post-processing pro-



(a) Single warp operation



(b) Comparing direct and latent warp. Step 0 is no post-processing, and step 10 is after 10 steps of post-processing.

Figure 2: Illustration of a warp edit. A green handle point is selected on the topology, and a red arrow represents a warp vector. The vector's magnitude determines the target's location (red cross), indicating the handle point's movement. The percentage difference in distance between the green point and the target location is calcualted Distance Error (DE). Additionally, the Compliance Error (CE) can be computed. (b) Direct warp: When the warp is performed directly on the topology, the edit causes significant local deformation, while the distance error remains relatively small. However, the compliance error increases as straight members become distorted. SIMP post-processing partially reverts some of the local changes, which reduces the distance error. Latent warp: In this case, the edit is reflected on the global topology, where post-processing has not made substantial changes to the topology. As a result, the topology remains stable, and the distance error is smaller.

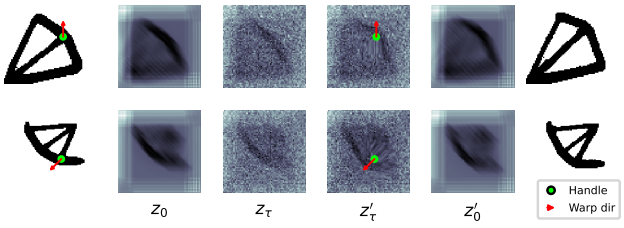


Figure 3: TopoEdit workflow for warping the topology. The topology is first converted into latent z_0, then partial noise is added z_τ, then the warp operation is mapped onto the latent space z'_τ, then the warped latent is denoised to obtain the latent z'_0. Decoding z'_0 gives the edited topology.

(a) No post processing (Step 0)

Best of	CE%	DE%	VFE%	Failure Rate%
2	28.25	49.09	1.16	39.06
4	26.65	40.23	1.27	33.59
8	27.23	36.42	1.25	25.78
16	26.82	31.00	1.32	18.75
32	24.77	26.25	1.25	17.19
64	24.34	25.86	1.25	10.94
Direct	28.61	21.95	5.93	8.59

(b) 10 steps of post processing (Step 10)

Best of	CE%	DE%	VFE%	Failure Rate%
2	5.21	42.19	0.00	2.34
4	4.89	31.97	0.00	0.00
8	4.86	24.46	0.00	0.00
16	4.76	19.49	0.00	0.00
32	4.27	15.82	0.00	0.00
64	4.15	12.27	0.00	0.00
Direct	1.76	50.05	0.00	0.00

Table 2: The best of N performance for the testing set. We see the same trend with the seaborn contour plot. The failure rate is evaluated as an edited result with more than 100% compliance with the original structure.

ceeds, the optimizer tends to repair the distorted region, which reduces compliance error while also changing the achieved joint location. For the latent warp, the edit is reflected globally, and the post-processing steps introduce fewer visible changes, so the topology remains stable across steps and the achieved joint location remains close to the target.

The mapped warp edit from the original topology onto the latent is visualized in Figure 3. To highlight how this behavior appears across different geometries, Figure 4 visualizes selected edits after 10 steps of SIMP post-processing. In the direct warp results, post-processing often leads to larger deviations from the reference topology. In cases where the topology changes substantially during refinement, the closest joint to the target can end up on a different edge, which reflects that the optimizer altered connectivity in response to the abrupt local deformation. The latent warp results in better preservation of the overall appearance of the structure, and the achieved joint tends to remain closer to the target with smaller compliance error.

Figure 5 contrasts determinism and stochasticity. Direct warp on the topology is deterministic and produces a single result for a fixed edit specification, while latent warp is stochastic and the same latent warp can yield different outcomes. This stochasticity can introduce alternative local adaptations beyond a pure warp, including cases where a warped joint is interpreted as a split and members branch to form different connections. These variations give an engineer additional options to choose a desirable edit outcome, and the cost of sampling multiple candidates remains small compared to re-running full optimization from scratch.

We select a representative latent result by choosing the best-of-N candidate that minimizes a sum of the compliance error and distance error. Figure 6 summarizes the distribution of direct and latent results across all warp operations using seaborn contours. At step 0, without any post-processing, the latent warp achieves a smaller compliance error while exhibiting a slightly larger distance error. As post-processing proceeds, the

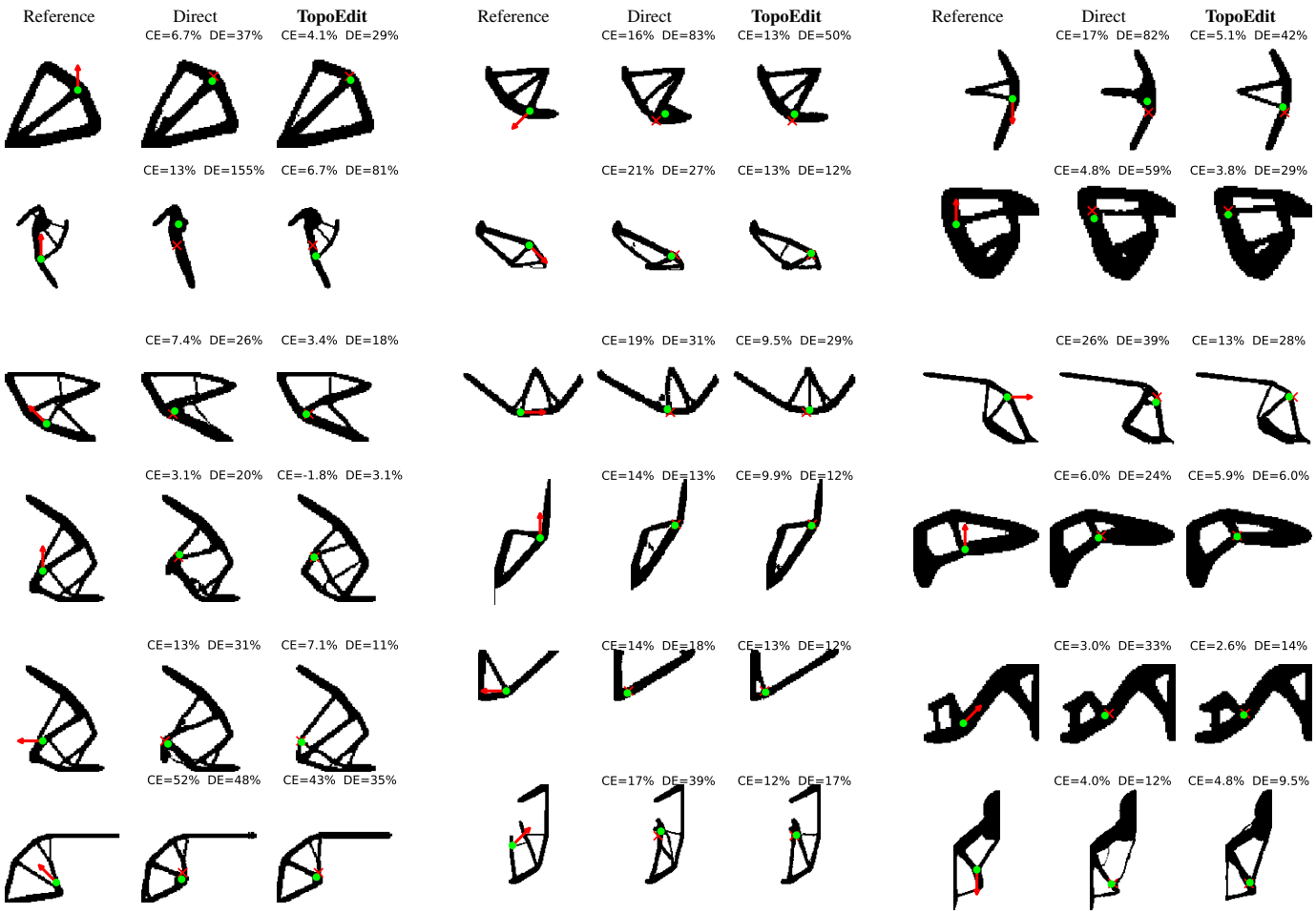
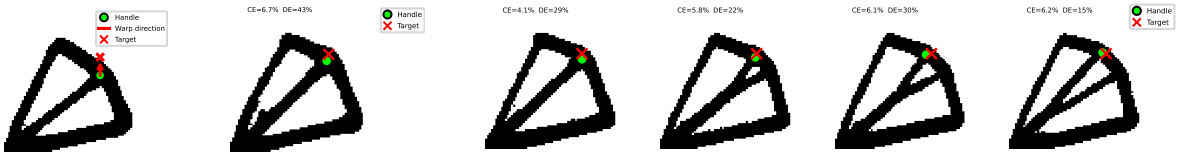
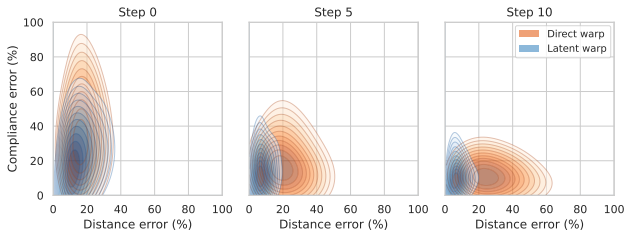


Figure 4: We select examples from the warp experiments to visualize the direct and latent-based edit. All edit results shown are after 10 steps of SIMP post-processing. The last three example is from the testing set. Left column: the reference topology. The warp edit is placed on a joint and shown as a red arrow with the end located at the point where the warp is applied and the arrow head pointing towards the target edit direction. Middle column: applying the warp edit directly to the topology, after SIMP post-processing, a larger deviation from the reference topology occurs. We show the closest joint point to the target edit location as a green dot and the edit target location as a red cross. In cases with large changes in topology after post-processing, the closest joint location is located on a different edge completely. The abrupt changes caused by the direct warp cause the post-processing optimizer to alter the topology and connectivity. Right column: with latent-based warp edits, the overall appearance of the structure is better preserved, resulting in smaller error compliance and a closer distance to the target location.

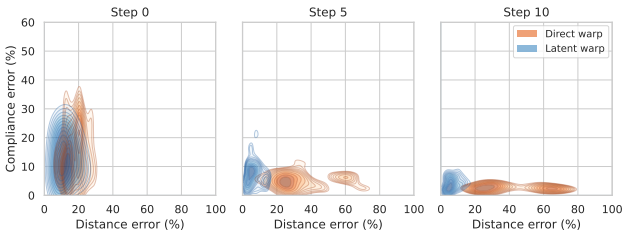


(a) Topology with edit (b) Direct warp (c) Latent warp

Figure 5: Applying the edit (a), direct warp (b) is deterministic and produces a single result. Latent warp (c) based on the diffusion model is stochastic; the same edit on the latent can produce different results. The number of results generated can be selected as one of the configuration parameters. The stochastic nature of the diffusion model allows the user to explore a distribution of possible outcomes.



(a) Training set



(b) Test set

Figure 6: We compute the distribution of the direct and latent warp (best of 64) across all warp operations for both the training and the testing set. For this plot, we apply a failure cutoff and drop cases where the compliance exceeds 1000% of the original topology, since these cases typically correspond to broken load transfer and dominate aggregate statistics. The distribution is visualized using seaborn contours. Initially, without any post-processing (step 0), the latent warp achieved a smaller compliance error but with a slightly larger error in distance. As more steps of post-processing are applied, we observe that the center of the latent warp remains consistent, indicating that the latent warp successfully performed a more global edit. Post-processing primarily reduced the spread of the warp. In contrast, the direct warp shifted towards the right, suggesting that the local edit was reversed.

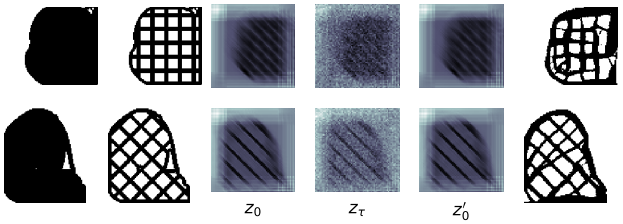


Figure 7: TopoEdit workflow for lattice replacement. We take an existing topology and then apply lattice infill into it. The topology is first converted into latent z_0, then partial noise is added z_τ, then the noised latent is denoised to obtain the latent z_0'. Decoding z_0' gives the topology with lattice infill.

center of the latent distribution remains consistent, which indicates that the latent warp performed a more global edit and refinement mainly reduces the spread. In contrast, the direct warp distribution shifts toward larger distance error with post-processing, suggesting that the local edit is being reversed as the optimizer repairs the deformed region. Table 2 summarizes the best of N performance for the latent versus direct edit for the test set. We observe a similar trend with the contour plots with improving performance as more latent example sampled.

4.2. Lattice Infill

Infill/lattice control is an important design objective because it provides a route to improve manufacturability and failure modes beyond stiffness, such as buckling, enabling porous shell–infill structures that can reduce support requirements and residual-stress issues in additive manufacturing. Prior work on interactive infill topology optimization addresses this need by pausing an optimization run, prompting the user to draw an infill pattern and region of interest, and resuming optimization under an appearance constraint [5]. In contrast, we treat lattice insertion as a post-optimization edit. We automatically identify replaceable interior regions using a shell–infill decomposition and employ latent diffusion denoising to adapt the inserted lattice to boundary conditions and load paths. This approach avoids catastrophic compliance failures that can occur when directly swapping infill (Figure 8 (b) cases with extremely large complaince).

In contrast to the topology warp example, we restrict the lattice experiment to reference designs that contain sufficiently thick interior regions. We compute the medial axis of each binarized reference topology and use the associated local thickness estimate to filter examples whose interior thickness exceeds a threshold of 20 (in pixel/element units). We then run lattice replacement on 150 such reference designs for the training set and 50 for the testing set.

We evaluate mechanical performance using compliance computed under the original loads and boundary conditions, and we report it for both the direct lattice replacements and the latent reconstructions. To assess whether the latent edit preserves the intended lattice identity, we compute the Intersection over Union (IoU) between the latent result and its corresponding direct lattice replacement. We also track the volume fraction of the edited topology relative to the target design to ensure that compliance changes are not explained by increasing material usage.

We select two examples to show the partial noise and denoise with lattice replaced topology in Figure 7. While the structure still retains similar visuals to the original lattice, the structure is rearranged. Figure 8 illustrates a recurring failure mode in direct lattice swapping. When the interior region is replaced with a lattice, the resulting structure may fail to adequately cover boundary-condition points. This can lead to ill-conditioned load transfer and potentially cause compliance values to explode. In contrast, the latent-based pipeline tends to preserve the overall lattice appearance while allowing adjustments that improve mechanical performance. As shown in the examples, the diffusion model nudges lattice members to better cover boundary-condition points and to better align with dominant load paths, producing lower compliance while maintaining an acceptable IoU relative to the direct lattice-replaced reference.

Table 3 summarizes the best of N performance on the test set for the lattice replacement experiment for the two lattice types. Across the grid- and cross-shaped lattice families, the latent lattice replacement achieves lower compliance than the corresponding direct lattice replacement. In addition, we do not observe cases where the volume fraction deviates from the target topology by more than 0.05, suggesting that the compli-

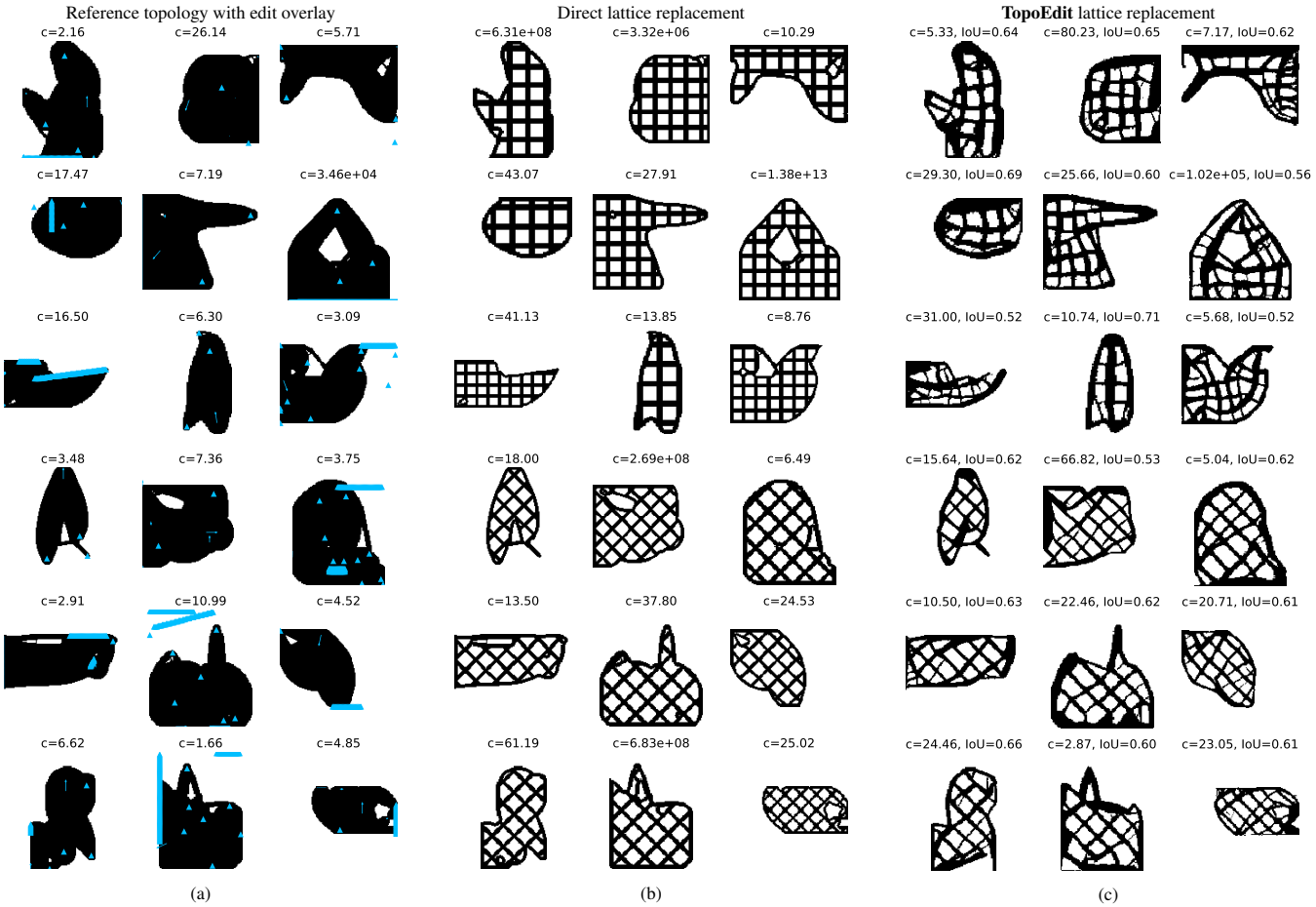


Figure 8: We visualize the result of direct and latent-based lattice replacement. The third and last row is from the testing set. (a) The original reference topology with boundary condition overlaid in blue and the compliance value. (b) The first three rows are using a grid-shaped lattice, and the last three rows are a cross-shaped lattice. When directly replacing the internal structure with lattices, the potential of missing a boundary condition is high, resulting in exploding compliance values. (c) With latent-based edits, the overall appearance of the lattice structure is preserved, while OAT managed to nudge the lattice structure to cover boundary condition points and align with the load paths. Resulting in lower compliance while still maintaining an acceptable IoU score to the original latent-replaced structures.

(a) Grid lattice replacement

Best of	VFE%	IoU	Beat Rate%	Failure Rate%
2	2.79	0.57	66.00	38.00
4	2.58	0.56	84.00	30.00
8	2.63	0.55	92.00	22.00
16	2.85	0.55	98.00	12.00
32	2.91	0.55	98.00	8.00
64	2.47	0.54	100.00	8.00
Direct	N/A	1.00	N/A	52.00

(b) Cross lattice replacement

Best of	VFE%	IoU	Beat Rate%	Failure Rate%
2	2.10	0.58	52.00	46.00
4	1.98	0.58	66.00	38.00
8	1.80	0.58	76.00	32.00
16	1.82	0.59	90.00	22.00
32	1.70	0.59	96.00	16.00
64	2.06	0.59	96.00	12.00
Direct	N/A	1.00	N/A	48.00

Table 3: Best of N result for the lattice replacement with the testing set. The beat rate is defined as the lattice having smaller compliance than the direct replacement. Due to the dramatic change of volume fraction and chances of missing boundary conditions, we define the failure rate here as the edited topology having 1000 times the compliance as the original structure.

ance gains do not come from increasing material usage. These results demonstrated the benefit of the latent edit in the lattice setting, where the diffusiom model provides a mechanism for producing lattice infills that remain recognizable while being better aligned with boundary-condition attachment and global load transfer.

We emphasize that the lattice results reported here are raw decoded outputs from the latent editing pipeline, rather than finalized manufacturable CAD geometries. We intentionally evaluate these raw outputs to provide a direct assessment of the model behavior, since additional cleanup steps such as island removal, smoothing, minimum member-size filtering, or CAD reconstruction would introduce extra assumptions beyond the latent edit itself. Therefore, the lattice replacement operator should be interpreted as a post-optimization design exploration tool for generating mechanically connected shell/infill candidates, rather than as a complete manufacturing pipeline. Fabrication-oriented use would require additional post-processing or manufacturing-aware constraints to ensure clean connectivity, minimum feature size, and process-compatible geometry.

4.3. No-design Region

No-design regions are often specified before topology optimization, since the optimization process can only account for constraints that are known at solve time. When a no-design region is introduced late in the workflow, updating an already-optimized design typically requires re-running optimization, which is time-consuming and can drift toward a different design. We therefore study late-stage no-design-region enforcement as a post-optimization edit and use it as a case study for applying a topology foundation model outside its original training distribution.

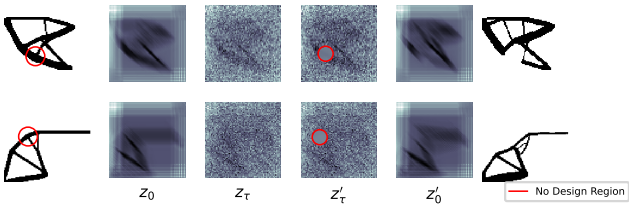


Figure 9: TopoEdit workflow with no design region enforcement. The topology is first converted into latent z_0, then partial noise is added z_τ, then the no design region is mapped as a mask onto the latent space z'_τ, then the masked latent is denoised to obtain the latent z'_0. Decoding z'_0 gives the edited topology.

Best of	VFE%	Violation%	Beat Rate%	Failure Rate%
2	-1.07	4.90	81.25	12.50
4	-1.33	4.93	87.50	6.25
8	-0.47	5.08	93.75	6.25
16	-0.67	4.69	100.00	0.00
32	-0.63	4.60	100.00	0.00
64	0.06	5.37	100.00	0.00
Direct	N/A	0.00	N/A	18.75

Table 4: Best of N result for the no design region with the testing set. We define the beat rate as latent based no design region enforcement having smaller compliance than directly applying the no design region onto the topology. We define the failure rate as the edited topology having 1000 times the compliance as the original structure.

We use the same filtered subset as in the warp experiment, where the reference topologies have clearly defined joints and relatively uniform member thickness. For each reference topology, we define a circular no-design region by placing a hole at the location of a joint. We exclude candidate joints that coincide with boundary-condition points, since removing material at supports can cause the compliance to spike, and leaves little room for a feasible design to recover. In total, we evaluate 126 reference topologies for the training set and 15 topologies for the testing set. Figure 9 illustrates the edit operation on the noised latent z'_τ where we map the no design region on the topology as a mask onto the noised latent.

We compare a direct baseline that applies the no-design region in density space against a latent edit that enforces the same void constraint during diffusion-based reconstruction. For evaluation, we compute compliance under the original loads and boundary conditions for both approaches. We also compute a violation ratio for the latent result, defined as the fraction of the no-design-region area that is occupied by material in the reconstructed topology, so that lower values indicate better adherence to the prescribed region.

Figure 10 illustrates a recurring failure mode in the direct baseline. When the hole masks out an important joint and its nearby members, key load paths may be severed, which can cause the compliance to become extremely large. In contrast, the latent-based edit tends to preserve the overall structural appearance while adjusting members to route around the forbidden region, producing feasible alternatives that respect the hole while maintaining connectivity.

Across all no-edit design placements, the latent-based edit achieves lower compliance than directly applying the no-design region (Table 4). Among these results, the average violation

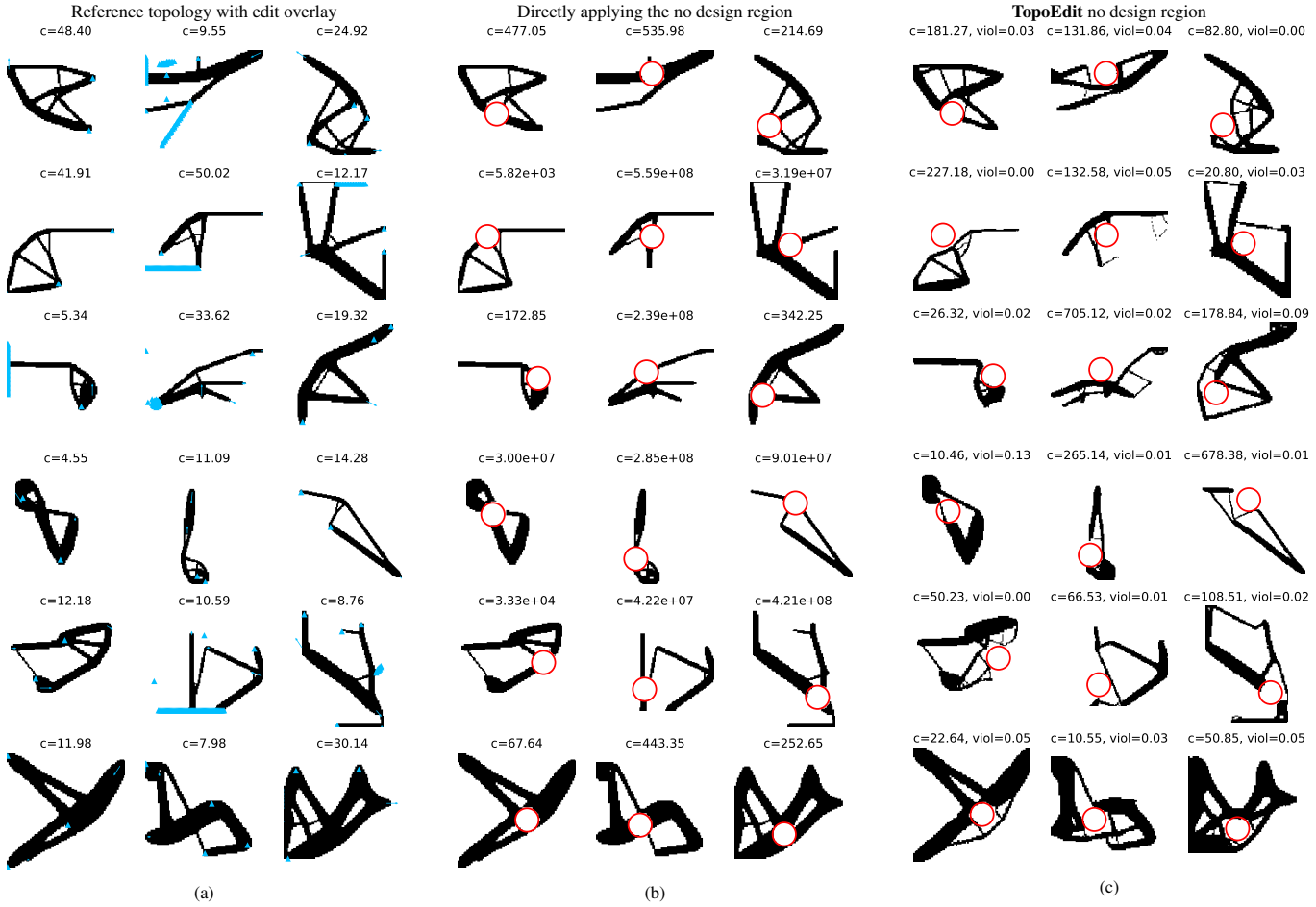


Figure 10: We visualize the result of directly applying the circular no design region onto the topology versus applying it in the latent vector. The last row is selected from the testing set. (a) The original reference topology with boundary condition overlaid in blue and the compliance value. (b) With the no design region directly applied to joint points, the hole masks out important load paths on the topology, causing the compliance to dramatically increase. (c) In contrast, latent-based editing managed to route around the holes and significantly reduce the compliance compared to directly applying the no design region in (b).

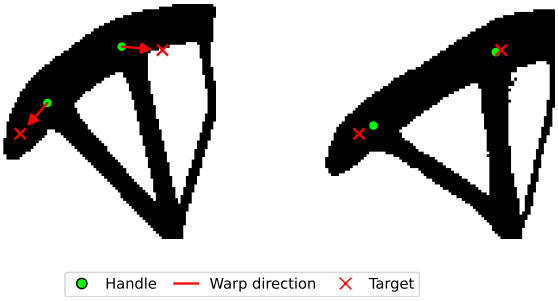


Figure 11: Multiple warp operations can occur simultaneously. We apply two warp operations on the topology from the testing set (left), with the latent-based warp result shown on the right after 10 steps of post-processing. The compliance error is 26%, and the distance error is 27%. The higher compliance error could be due to the two edit operations causing greater changes for the topology and moving it away from the compliance optima.

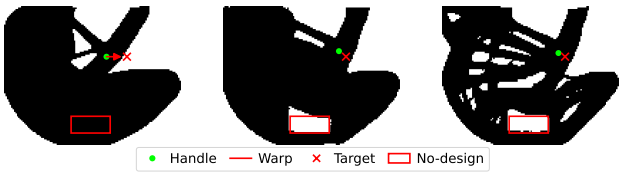


Figure 12: Applying all three types of edit on a single example (left). We apply the topology warp and no design region first (middle), then apply the grid-shaped lattice infill replacement (right). The compliance of the original structure is 3.9, the middle structure is 4.12 and the lattice replaced structure is 5.74. The distance error is 43% and 36%.

ratio is small, indicating that the generated topologies generally avoid encroaching into the no-design region while still finding alternative load paths around the hole.

4.4. Multiple Edits

In addition to the single edit demonstrated and benchmarked in the previous sections, TopoEdit supports simultaneous editing. We first showcase an example in Figure 11 with two handle points and two different warp directions. The two handle points can be accommodated by passing in two points instead of one for the handle point $\mathbf{h}$ and the warp direction Δ. With two edits, we observe a slight increase in compliance error because the two edits cause greater changes to the topology away from the compliance optima. However, the distance error remains consistent with the topology warp effect of a single handle point.

We also set up an example shown in Figure 12 to demonstrate all three types of edits: topology warp, lattice replacement, and no design region combined. Due to the slightly different noise schedule of the lattice replacement compared to topology warp and no design region enforcement, we perform the warp and no design edit first. Then, we take the edited structure and apply lattice replacement. Throughout the edits, the compliance remains stable without significant increase, while the topology warp effect remains persistent. We also observe no significant violation in the no design region.

Partial steps	CE%	DE%	Failure Rate%
15	24.62	34.73	40.48
20	24.93	30.41	34.52
25	25.88	36.20	25.00
30	25.20	36.90	25.00
35	25.71	34.61	23.81

Table 5: Effect of DDIM partial steps τ on topology warp. Increasing τ changes the amount of latent noise before denoising. We select 20 partial steps of noise as the default warp setting because it gives a lower distance error among the tested values while avoiding the high failure rate observed at smaller τ.

Partial steps	Mean IoU	Beat Rate%	Failure Rate%
10	0.6617	48.00	47.00
15	0.6072	57.00	40.00
20	0.5636	61.00	35.00
25	0.5458	60.00	32.00

Table 6: Effect of DDIM partial steps τ on grid lattice replacement. Larger τ reduces failure rate but also reduces IoU with the lattice-replaced reference. We select 20 partial steps as the default lattice setting because it reduces failure rate relative to smaller τ while retaining the reference lattice layout.

4.5. Ablation Studies

The post-optimization edit with latent diffusion model for the three types edit involves the selection of the number of partial noise steps. Furthermore, for the topology warp, varying the influence radius σ and warp magnitude Δ also affects the warp results. In the previous results for topology warp, we only select points that are identified as joints by medial axis skeletonization, however, boundary condition points can also be warped, where a direct comparison with SIMP based topology optimization with the warped boundary condition can be established. Despite in most cases the boundary condition points does not overlap with the joint points determined by medial axis, meaning no distance error can be computed, nevertheless, this comparison provides insight on the structural performance of latent based warp. For the no design region, a SIMP based baseline run with the no design region directly enforced can also establish the structural peformanece on the latent based no design region enforcement. We summarize the ablation study results in this subsection.

4.5.1. Partial steps

The partial step parameter τ controls the amount of Gaussian noise injected into the latent before denoising. Small τ keeps the noised latent close to the encoded reference topology, which helps retain the intended edit information but limits the ability of the diffusion model to repair disrupted load paths. Large τ gives the diffusion model more freedom to recover mechanically optimal structures, but excessive noising can move the sample toward regeneration and reduce retention of the intended edit. We therefore treat τ as an operator-level sampling hyperparameter rather than as an optimization variable. The ablation in Tables 5, 6, and 7 is used to verify the sensitivity of each edit operator to τ and to select a fixed default value that keeps the edit in the intended post-optimization editing regime. For the partial steps ablation study, we run best of 16 to reduce the total runtime.

Partial steps	Violation%	Beat Rate%	Failure Rate%
10	1.51	77.08	54.17
15	1.45	77.08	56.25
20	1.38	77.08	54.17
25	1.72	81.25	47.92
30	2.12	77.08	50.00

Table 7: Effect of DDIM partial steps τ on no-design region enforcement. Larger τ gives the diffusion model more freedom to restore load paths after the latent overwrite, but excessive τ may reduce retention of the prescribed no-design region. We select $\tau = 25$ as the default no-design setting based on the tested range.

σ	Mean CE%	Mean DE%
0.050	19.24	47.80
0.075	22.23	45.38
0.100	24.93	30.41
0.150	32.39	28.44

Table 8: Effect of Gaussian influence radius σ on topology warp. A smaller influence radius better preserves compliance but produces larger distance error, while a larger influence radius improves the achieved displacement at the cost of higher compliance error.

Δ	CE%	DE%
0.150	21.14	36.95
0.200	24.93	30.41
0.250	35.77	31.81

Table 9: Effect of displacement magnitude Δ on topology warp. Increasing the requested displacement improves distance error up to Δ=0.20, but the largest tested displacement increases compliance error without further improving the achieved edit, indicating the onset of the unreliable warp regime.

Best of	Step 0 CE%	Step 0 Fail%	Step 10 CE%	Step 10 Fail%
2	11.849	44.643	-0.248	1.339
4	12.489	36.607	-2.027	0.446
8	12.279	26.786	-2.236	0.000
16	10.596	18.750	-2.453	0.000
32	10.059	16.071	-2.771	0.000
64	8.514	11.607	-2.915	0.000
Direct	12.721	2.679	-2.323	0.000

Table 10: Boundary condition warp compared with a 200-step SIMP reference. Compliance error is computed relative to the SIMP solution under the warped boundary-condition specification. Step 0 reports the raw edited result, and Step 10 reports the result after 10 SIMP post-processing steps.

For each edit operator, we select a single default τ from the tested range and use it for all reported examples of that operator. The selected value is chosen from a stable region of the ablation where failure rate is reduced without a large loss of the measured edit-retention metric: distance error for topology warp, IoU for lattice replacement, and no-design violation for no-design enforcement. This selection is intended to define a practical default sampling configuration, not to solve a weighted multi-objective optimization problem for each instance.

4.5.2. Topology warp related hyperparameters and benchmarking

We first vary the Gaussian influence radius σ while keeping the remaining warp settings fixed (Table 8). Increasing σ reduces the distance error from 47.80% to 28.44%, indicating that a broader neighborhood allows the dragged joint to move more coherently with the surrounding structure. However, the mean compliance error simultaneously increases from 19.24% to 32.39%. This behavior is expected: a small influence radius confines the deformation to a narrow region and therefore perturbs fewer load paths, but the edit is also more likely to be partially suppressed during denoising and post-processing. A larger influence radius better propagates the intended geometric displacement, but it also changes a larger portion of the structure and can move the topology farther from the original compliance optimum.

We next vary the displacement magnitude Δ (Table 9). Increasing Δ from 0.15 to 0.20 reduces the distance error from 36.95% to 30.41%, but further increasing it to 0.25 does not improve the achieved distance error and instead increases the mean compliance error from 24.93% to 35.77%. This indicates that the useful edit range has been exceeded: the requested displacement becomes large enough to significantly distort the local load-transfer geometry, while the diffusion and post-processing stages can no longer convert the stronger drag into a proportionally better achieved joint movement.

This trend is also consistent with the geometry of the Gaussian inverse-map warp. For a displacement field $u(x)$ computed in Equation 8, the local invertibility of the inverse-map sampling becomes increasingly ill-conditioned as $\frac{|\Delta|}{\sigma}$ grows. In the continuous idealized case, the Jacobian determinant of the inverse mapping can approach zero when the displacement magnitude becomes large relative to the influence radius, corresponding to local compression or folding of the coordinate map. In practice, TopoEdit applies the warp in latent space and then denoises the result, so this ratio should not be interpreted as a strict failure threshold. However, it explains why the largest tested displacement produces a sharp compliance increase without a corresponding improvement in distance error.

These results show that topology warp has a finite operating range. Within the tested range, σ=0.10 and Δ=0.20 provide a practical default setting: smaller edits can be partially pulled back toward the reference topology, while stronger edits begin to mechanically deteriorate. We therefore do not claim that TopoEdit can recover arbitrary geometric drags. Instead, the method is most suitable for moderate post-optimization revisions where the requested displacement is large enough to be reflected in the topology but not so large that the induced coordinate warp strongly compresses or folds the local geometry.

For very small or localized deformations, direct density-space warping followed by short SIMP post-processing may be sufficient, since the edit does not strongly disturb the structure members. The benefit of TopoEdit is more apparent when the edit requires structural adaptation beyond a purely local geometric displacement. Conversely, when the requested displacement becomes too large, even the latent edit begins to lose reliability, as shown by the increase in compliance error at Δ=0.25.

For generic joint dragging, there is no direct equivalent standard topology optimization problem unless an additional geometric constraint is introduced to force a particular internal joint to move. To provide a conventional optimization reference in

Best of	CE%
2	537.40
4	487.12
8	159.10
16	139.24
32	122.24
64	114.34
Direct	1670.25

Table 11: Best of N result for no-design region compared with applying the no design region on a SIMP rerun. We filter out the cases with more than 1000× the original structure compliance consistent with the previous table.

a setting where such a comparison is well-defined, we evaluate a consistent warp experiment in which the support and force locations are warped and compare the edited results against a 200-step SIMP solution under the warped boundary-condition specification (Table 10).

At step 0, the raw latent samples still show nontrivial failure rates, indicating that diffusion output alone is not always mechanically stabilized relative to the SIMP reference. However, best-of-N sampling reduces both compliance error and failure rate. After 10 steps of SIMP post-processing, the latent results reach the same compliance range as the 200-step SIMP reference, with failure rates below 1.34% for best-of-2 and 0% for best-of-8 and above. The direct warp baseline also reaches this range after post-processing. Therefore, the BC-consistent experiment should be interpreted as evidence that TopoEdit provides a stable warm start for edited boundary-condition problems, rather than as evidence that it outperforms full topology optimization.

The slightly negative compliance errors after 10 post-processing steps should not be interpreted as surpassing a mathematical optimum. The reference is a finite 200-step SIMP run, and the edited structures receive additional warm-started post-processing. These values indicate that the edited designs are in the same compliance range as the conventional SIMP reference, not that TopoEdit provides a lower bound on compliance.

4.5.3. No-design region benchmarking

Unlike topology warp, no-design region enforcement has a natural conventional reference: the no-design mask can be imposed directly during a new SIMP optimization. We therefore compare TopoEdit against a 200-step SIMP rerun with the no-design region enforced. As expected, the full SIMP rerun gives better compliance, since it solves the constrained optimization problem from scratch. The purpose of this comparison is not to show that latent editing replaces re-optimization, but to quantify the compliance penalty of using TopoEdit as a fast post-optimization edit. Compared with directly removing material inside the no-design region, the latent-based edit reduces compliance error, especially as more candidates are sampled. This suggests that TopoEdit can provide a rapid preview of how the structure may adapt to a newly imposed no-design region, while full SIMP rerunning remains the appropriate choice when final compliance optimality is required.

Table 11 shows the no-design region benchmark against the SIMP rerun reference. Since the SIMP rerun explicitly solves the constrained optimization problem with the no-design region imposed, it achieves lower compliance than the post-optimization edits. The direct no-design edit performs poorly because removing material can sever load paths without allowing structural recovery. The latent-based edit reduces this compliance penalty, with best-of-64 decreasing the mean CE from 1670.25% for the direct edit to 114.34%. Although this remains above the SIMP rerun reference, it demonstrates that TopoEdit can provide a fast post-optimization estimate of the structural adaptation induced by a late no-design constraint. Therefore, we view this operator as a rapid editing and screening tool rather than a substitute for full re-optimization when final compliance is the primary objective.

5. Conclusion and Future Work

In this work, we introduced TopoEdit, a post-optimization topology editor built on a latent diffusion foundation model. By encoding an optimized topology into a structured spatial latent, applying partial noising to retain instance identity while improving editability, and performing an edit-then-denoise procedure with consistency-preserving guidance, TopoEdit enables fast, localized, and physics-aware modifications of existing designs. Across diverse case studies, latent-space editing consistently produced intention-aligned changes while better preserving mechanical performance than direct density-space edits baselines.

While TopoEdit already demonstrates strong performance across three editing tasks, several directions can further improve both edit fidelity and robustness. First, because the editor inherits its priors from the underlying topology foundation model, advances in latent diffusion for data-driven topology optimization, with higher reconstruction accuracy, stronger conditioning generalization, and improved sampling success rates, should directly translate into improved edit performance and fewer failure cases. Second, our current editor uses a fixed-resolution latent representation; increasing latent spatial resolution and/or using richer multi-channel latents may enable finer, more localized edits.

Beyond improving the latent representation, expanding the edit operator set is another potential direction. A natural extension of the drag-based warp is explicit edge-thickening and thinning, where the user can locally increase or decrease member width while preserving connectivity and global volume constraints. More broadly, more manufacturing-aware edits can be beneficial to engineers. The lattice replacement study already suggests a surrogate route to reduce support requirements by introducing porous shell–infill structures, but an explicit treatment of overhang angle control implemented as either a latent-space constraint, a guidance term during denoising, or a learned conditioning signal would provide a more direct handle on printability and process-driven geometric feasibility.

The current validation is limited to compliance-minimizing structural topology optimization, and extensions to other physics should be treated carefully. Although partial noising and denoising allow a locally specified edit to induce changes

in surrounding structural members, strongly non-local or nonlinear physics, such as acoustics, photonics, or resonance-dominated design, would require a foundation model trained on the corresponding optimized solutions and separate validation before the same latent editing strategy can be considered reliable.

Finally, TopoEdit can be extended to three-dimensional domains. Once a 3D latent diffusion model for topology optimization is available, the same pipeline can be generalized to volumetric warps, 3D lattice insertion, and 3D no-design enforcement, enabling post-optimization edits for practical 3D structural components while maintaining physics-aware behavior under complex boundary conditions.

References

[1] A. Chandrasekhar, K. Suresh, Length scale control in topology optimization using fourier enhanced neural networks, CoRR abs/2109.01861 (2021). arXiv:2109.01861.

[2] K. Hu, S. Jin, C. C. Wang, Support slimming for single material based additive manufacturing, Computer-Aided Design 65 (2015) 1–10. doi:https://doi.org/10.1016/j.cad.2015.03.001.

[3] D. Q. Ha, J. V. Carstensen, Human-informed topology optimization: interactive application of feature size controls, Structural and Multidisciplinary Optimization 66 (2023) 59. doi:10.1007/s00158-023-03512-0.

[4] G. Schiffer, D. Q. Ha, J. V. Carstensen, Hitop 2.0: combining topology optimisation with multiple feature size controls and human preferences, Virtual and Physical Prototyping 18 (1) (2023). doi:10.1080/17452759.2023.2268603.

[5] G. Schiffer, M.-P. Schmidt, C. B. W. Pedersen, J. V. Carstensen, Interactive infill topology optimisation guided by user drawn patterns, Virtual and Physical Prototyping 19 (1) (2024) e2361864. doi:10.1080/17452759.2024.2361864.

[6] A. Heyrani Nobari, G. Giannone, L. Regenwetter, F. Ahmed, NITO: Neural implicit fields for resolution-free topology optimization, arXiv preprint arXiv:2402.05073 (2024). arXiv:2402.05073.

[7] Z. Nie, T. Lin, H. Jiang, L. B. Kara, TopologyGAN: Topology optimization using generative adversarial networks based on physical fields over the initial domain, Journal of Mechanical Design 143 (3) (2021) 031715. doi:10.1115/1.4049533.

[8] M. P. Bendsøe, N. Kikuchi, Generating optimal topologies in structural design using a homogenization method, Computer Methods in Applied Mechanics and Engineering 71 (2) (1988) 197–224. doi:10.1016/0045-7825(88)90086-2.

[9] G. I. N. Rozvany, M. Zhou, T. Birker, Generalized shape optimization without homogenization, Structural Optimization 4 (1992) 250–252. doi:10.1007/BF01742754.

[10] M. P. Bendsøe, O. Sigmund, Material interpolation schemes in topology optimization, Archive of Applied Mechanics 69 (1999) 635–654. doi:10.1007/s004190050248.

[11] J. A. Sethian, A. Wiegmann, Structural boundary design via level set and immersed interface methods, Journal of Computational Physics 163 (2) (2000) 489–528. doi:10.1006/jcph.2000.6581.

[12] G. Allaire, F. Jouve, A.-M. Toader, Structural optimization using sensitivity analysis and a level-set method, Journal of Computational Physics 194 (1) (2004) 363–393. doi:10.1016/j.jcp.2003.09.032.

[13] O. Sigmund, A 99 line topology optimization code written in Matlab, Structural and Multidisciplinary Optimization 21 (2) (2001) 120–127. doi:10.1007/s001580050176.

[14] M. M. Behzadi, H. T. Ilieş, GANTL: Toward practical and real-time topology optimization with conditional generative adversarial networks and transfer learning, Journal of Mechanical Design 144 (2) (2022) 021711. doi:10.1115/1.4052757.

[15] C. Sharpe, C. C. Seepersad, Topology design with conditional generative adversarial networks, in: Proceedings of the ASME 2019 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE), 2019, p. V02AT03A062. doi:10.1115/DETC2019-97833.

[16] B. Li, C. Huang, X. Li, S. Zheng, J. Hong, Non-iterative structural topology optimization using deep learning, Computer-Aided Design 115 (2019) 172–180. doi:10.1016/j.cad.2019.05.038.

[17] Y. Yu, T. Hur, J. Jung, I. G. Jang, Deep learning for determining a near-optimal topological design without any iteration, Structural and Multidisciplinary Optimization 59 (2019) 787–799. doi:10.1007/s00158-018-2101-5.

[18] Z. Zhang, Y. Li, W. Zhou, X. Chen, W. Yao, Y. Zhao, TONR: An exploration for a novel way combining neural network with topology optimization, Computer Methods in Applied Mechanics and Engineering 386 (2021) 114083. doi:10.1016/j.cma.2021.114083.

[19] D. Wang, C. Xiang, Y. Pan, A. Chen, X. Zhou, Y. Zhang, A deep convolutional neural network for topology optimization with perceptible generalization ability, Engineering Optimization 54 (6) (2022) 973–988. doi:10.1080/0305215X.2021.1902998.

[20] S. Rawat, M.-H. H. Shen, A novel topology optimization approach using conditional deep learning, arXiv preprint arXiv:1901.04859 (2019). arXiv:1901.04859.

[21] S. Zheng, Z. He, H. Liu, Generating three-dimensional structural topologies via a U-Net convolutional neural network, Thin-Walled Structures 159 (2021) 107263. doi:10.1016/j.tws.2020.107263.

[22] V. Keshavarzzadeh, M. Alirezaei, T. Tasdizen, R. M. Kirby, Image-based multiresolution topology optimization using deep disjunctive normal shape model, Computer-Aided Design 130 (2021) 102947. doi:10.1016/j.cad.2020.102947.

[23] J. Hu, Y. He, B. Xu, S. Wang, N. Lei, Z. Luo, IF-TONIR: Iteration-free topology optimization based on implicit neural representations, Computer-Aided Design 167 (2024) 103639. doi:10.1016/j.cad.2023.103639.

[24] F. Mazé, F. Ahmed, Diffusion models beat GANs on topology optimization, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 37, 2023, pp. 9108–9116. doi:10.1609/aaai.v37i8.26093.

[25] G. Giannone, F. Ahmed, Diffusing the optimal topology: A generative optimization approach, arXiv preprint arXiv:2303.09760 (2023). arXiv:2303.09760.

[26] G. Giannone, A. Srivastava, O. Winther, F. Ahmed, Aligning optimization trajectories with diffusion models for constrained design generation, in: Advances in Neural Information Processing Systems, Vol. 36, 2023, pp. 51830–51861.

[27] A. Heyrani Nobari, L. Regenwetter, C. Picard, L. Han, F. Ahmed, Optimize any topology: A foundation model for shape- and resolution-free structural topology optimization, arXiv preprint arXiv:2510.23667 (2025). arXiv:2510.23667.

[28] G. Allaire, F. Jouve, G. Michailidis, Thickness control in structural optimization via a level set method, Structural and Multidisciplinary Optimization 53 (6) (2016) 1349–1382. doi:10.1007/s00158-016-1453-y.

[29] X. Guo, W. Zhang, W. Zhong, Explicit feature control in structural topology optimization via level set method, Computer Methods in Applied Mechanics and Engineering 272 (2014) 354–378. doi:10.1016/j.cma.2014.01.010.

[30] M.-P. Schmidt, C. B. W. Pedersen, C. Gout, On structural topology optimization using graded porosity control, Structural and Multidisciplinary Optimization 60 (2019) 1437–1453. doi:10.1007/s00158-019-02275-x.

[31] T. Navez, M.-P. Schmidt, O. Sigmund, C. B. W. Pedersen, Topology optimization guided by a geometrical pattern library, Structural and Multidisciplinary Optimization 65 (2022) 108. doi:10.1007/s00158-022-03197-x.

[32] L. Zhu, T. Zuo, C. Wang, Q. Wang, Z. Yu, Z. Liu, Hole appearance constraint method in 2d structural topology optimization, Mathematics 12 (17) (2024) 2645. doi:10.3390/math12172645.

[33] X. Meng, L.-Y. Zhang, Z.-L. Zhao, Y. M. Xie, A direct approach to achieving efficient free-form shells with embedded geometrical patterns, Thin-Walled Structures 185 (2023) 110559. doi:10.1016/j.tws.2023.110559.

[34] Y. Li, Z. Zhang, J. Luo, W. Peng, W. Zhou, W. Yao, Concurrent topology optimization of shells with pattern-guided infills for intuitive design and additive manufacturing, Computer Methods in Applied Mechanics and Engineering 418 (2024) 116485. doi:10.1016/j.cma.2023.116485.

[35] Z. Zhang, L. Jiang, T. Yarlagadda, Y. Zheng, A. Usmani, A novel multi-pattern control for topology optimization to balance form and performance needs, Engineering Structures 303 (2024) 117581. doi:10.1016/j.engstruct.2024.117581.

[36] Z. Li, T.-U. Lee, Y. M. Xie, Interactive structural topology optimization with subjective scoring and drawing systems, Computer-Aided Design 160 (2023) 103532. doi:10.1016/j.cad.2023.103532.

[37] Z. Li, T.-U. Lee, Y. M. Xie, Interactive 3d structural design in virtual reality using preference-based topology optimization, Computer-Aided Design 180 (2025) 103826. doi:10.1016/j.cad.2024.103826.

[38] S. Zhu, J. Hu, J. Qi, L. Wang, J. Guo, J. Ma, G. Zhu, Sketch-guided topology optimization with enhanced diversity for innovative structural design, Applied Sciences 15 (5) (2025) 2753. doi:10.3390/app15052753.

[39] W. Zhang, Y. Wang, S.-K. Youn, X. Guo, Machine learning powered sketch aided design via topology optimization, Computer Methods in Applied Mechanics and Engineering 419 (2024) 116651. doi:10.1016/j.cma.2023.116651.

[40] C. T. Mueller, J. Ochsendorf, Combining structural performance and designer preferences in evolutionary design space exploration, Automation in Construction 52 (2015) 70–82. doi:10.1016/j.autcon.2015.02.011.

[41] C. Barnes, E. Shechtman, A. Finkelstein, D. B. Goldman, Patchmatch: A randomized correspondence algorithm for structural image editing, ACM Transactions on Graphics 28 (3) (2009) 24. doi:10.1145/1531326.1531330.

[42] C. Barnes, D. B. Goldman, E. Shechtman, A. Finkelstein, The patchmatch randomized matching algorithm for image manipulation, Communications of the ACM 54 (11) (2011) 103–110. doi:10.1145/2018396.2018421.

[43] C. Barnes, F.-L. Zhang, A survey of the state-of-the-art in patch-based synthesis, Computational Visual Media 3 (2017) 3–20. doi:10.1007/s41095-016-0064-2.

[44] H. A. Khojasteh, A survey on patch-based synthesis: Gpu implementation and optimization (2020). arXiv:2005.06278.

[45] P. S. Vulimiri, H. Deng, F. Dugast, X. Zhang, A. C. To, Integrating geometric data into topology optimization via neural style transfer, Materials 14 (16) (2021) 4551. doi:10.3390/ma14164551.

[46] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, S. Ganguli, Deep unsupervised learning using nonequilibrium thermodynamics, in: Proceedings of the 32nd International Conference on Machine Learning, Vol. 37 of Proceedings of Machine Learning Research, 2015, pp. 2256–2265.

[47] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, in: Advances in Neural Information Processing Systems, 2020. arXiv:2006.11239.

[48] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, B. Poole, Score-based generative modeling through stochastic differential equations, in: International Conference on Learning Representations, 2021. arXiv:2011.13456.

[49] J. Song, C. Meng, S. Ermon, Denoising diffusion implicit models, in: International Conference on Learning Representations, 2021. arXiv:2010.02502.

[50] J. Ho, T. Salimans, Classifier-free diffusion guidance (2022). arXiv:2207.12598.

[51] P. Dhariwal, A. Nichol, Diffusion models beat GANs on image synthesis, in: Advances in Neural Information Processing Systems, 2021. arXiv:2105.05233.

[52] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, B. Ommer, High-resolution image synthesis with latent diffusion models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. arXiv:2112.10752.

[53] T. Salimans, J. Ho, Progressive distillation for fast sampling of diffusion models, in: International Conference on Learning Representations, 2022. arXiv:2202.00512.

[54] Y. Song, P. Dhariwal, M. Chen, I. Sutskever, Consistency models, in: Proceedings of the 40th International Conference on Machine Learning, 2023. arXiv:2303.01469.

[55] S. Luo, Y. Tan, L. Huang, J. Li, H. Zhao, Latent consistency models: Synthesizing high-resolution images with few-step inference (2023). arXiv:2310.04378.

[56] G. Kim, T. Kwon, J. C. Ye, Diffusionclip: Text-guided diffusion models for robust image manipulation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022. arXiv:2110.02711.

[57] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, D. Cohen-Or, Prompt-to-prompt image editing with cross attention control (2022). arXiv:2208.01626.

[58] T. Brooks, A. Holynski, A. A. Efros, Instructpix2pix: Learning to follow image editing instructions, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023. arXiv:2211.09800.

[59] B. Kawar, S. Zada, O. Lang, O. Tov, H. Chang, T. Dekel, I. Mosseri, M. Irani, Imagic: Text-based real image editing with diffusion models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023. arXiv:2210.09276.

[60] R. Mokady, A. Hertz, K. Aberman, Y. Pritch, D. Cohen-Or, Null-text inversion for editing real images using guided diffusion models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023. arXiv:2211.09794.

[61] M. Cao, X. Wang, Z. Qi, Y. Shan, X. Qie, Y. Zheng, Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2023. arXiv:2304.08465.

[62] X. Pan, A. Tewari, T. Leimkühler, L. Liu, A. Meka, C. Theobalt, Drag your GAN: Interactive point-based manipulation on the generative image manifold (2023). arXiv:2305.10973.

[63] Y. Shi, C. Xue, J. H. Liew, J. Pan, H. Yan, W. Zhang, V. Y. F. Tan, S. Bai, Dragdiffusion: Harnessing diffusion models for interactive point-based image editing, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024. arXiv:2306.14435.

[64] C. Mou, X. Wang, J. Song, Y. Shan, J. Zhang, Dragondiffusion: Enabling drag-style manipulation on diffusion models, in: International Conference on Learning Representations, 2024. arXiv:2307.02421.

[65] C. Mou, X. Wang, J. Song, Y. Shan, J. Zhang, Diffeditor: Boosting accuracy and flexibility on diffusion-based image editing, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024. arXiv:2402.02583.

[66] P. Ling, L. Chen, P. Zhang, H. Chen, Y. Jin, J. Zheng, Freedrag: Feature dragging for reliable point-based image editing (2023). arXiv:2307.04684.

[67] X. Zhao, J. Guan, C. Fan, D. Xu, Y. Lin, H. Pan, P. Feng, Fastdrag: Manipulate anything in one step, in: Advances in Neural Information Processing Systems, 2024. arXiv:2405.15769.

[68] H. Liu, C. Xu, Y. Yang, L. Zeng, S. He, Drag your noise: Interactive point-based editing via diffusion semantic propagation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024. arXiv:2404.01050.