OAT3D: A GENERATIVE MODEL FOR 3D SHAPE- AND RESOLUTION-FREE STRUCTURAL TOPOLOGY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Topology optimization determines how material should be distributed within a design domain to optimize structural performance under prescribed loads, supports, and constraints. The problems are highly nonconvex, and conventional solvers require repeated finite element analyses, making high-resolution 3D topology optimization computationally expensive. Data-driven approaches face an additional bottleneck: generating training datasets requires solving large numbers of these costly problems, so existing work in data-driven topology optimization primarily focuses on 2D problems. Large-scale level 3D datasets remain scarce, and existing models remain limited in problem scale and boundary-condition diversity. Here, we introduce Optimize Any Topology-3D (OAT3D), a generative model for high-resolution 3D structural topology optimization. OAT3D introduces a novel sparse problem encoding to represent load and fixity conditions and combines a sparse volumetric variational autoencoder with a conditional diffusion transformer. We construct a dataset of 501,865 optimized topologies, generated using 51,800 GPU-hours and encompassing structures with up to one million elements and substantially more complex loading and support configurations than any existing structural topology optimization dataset. To our knowledge, this is the largest dataset for 3D topology optimization structures. Trained on this dataset, OAT3D generates structures at 512^3 resolution and achieves a 45-fold speedup over conventional topology optimization. These contributions expand the scale and diversity of data-driven topology optimization for three-dimensional structural design.

1 INTRODUCTION

Topology optimization is a physics-based approach to engineering design that determines how material should be distributed within a prescribed domain to optimize for structural performance ob-

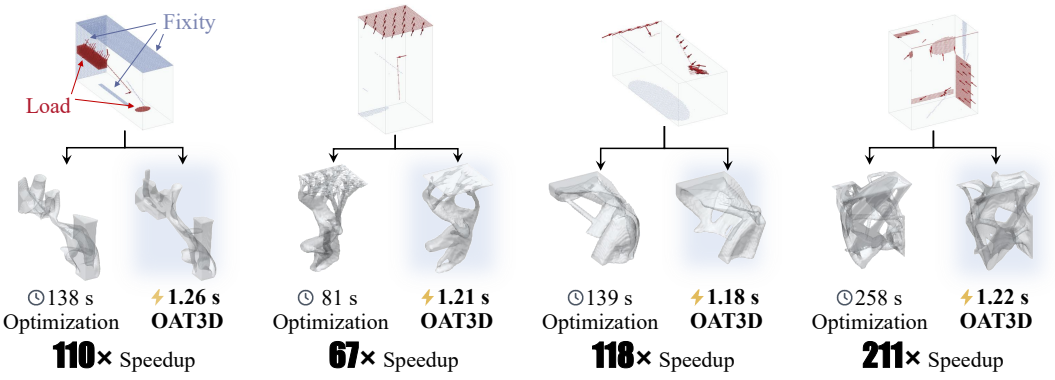


Figure 1: Topology optimization finds optimal structures for given loads and fixities, but requires multiple finite element analyses. OAT3D, trained on **501,865 optimized structures**, generates comparable designs without iterative optimization and finite element analysis: $\sim$**45**$\times$ **faster** on average.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

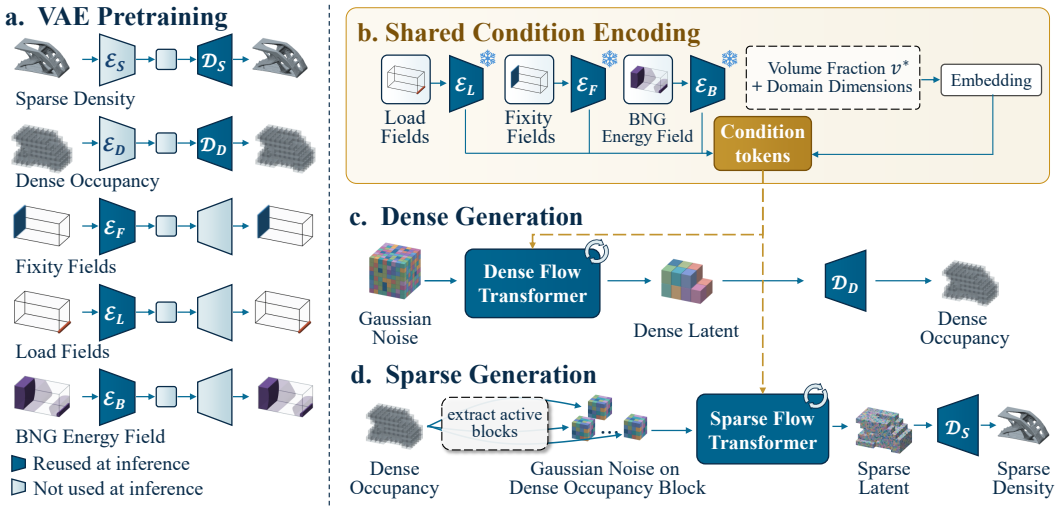


Figure 2: **OAT3D framework.** (a) Variational Autoencoder (VAE)s are pretrained to encode the load and fixity proximity fields and the BNG fields and to represent topology at two resolutions. (b) Frozen condition encoders combine the spatial field latents with embeddings of the native domain dimensions and volume fraction to form a shared conditioning sequence. (c) Conditioned dense flow matching transforms Gaussian noise into a dense latent, which is decoded into dense occupancy. (d) The active dense occupancy cells define the sparse latent generation site, on which conditioned sparse flow matching generates topology latents that are decoded into the final density field.

Table 1: Comparison of 3D topology optimization datasets. Elements are counted in the finite element mesh. Aspect ratios are counted as unique combinations of domain dimensions.

Dataset	Number of Topologies (↑)	Minimum Elements (↑)	Maximum Elements (↑)	Total Elements (↑)	Aspect Ratios (↑)
Keshavarzzadeh et al. (2021)	2.0K	0.4K	3.2K	3.6M	1
Behzadi & Ilieş (2021)	2.2K	8.0K	128K	75M	5
SELTO (Dittmer et al., 2022)	9.8K	6.1K	32K	315M	2
Herron et al. (2024)	66K	**155K**	155K	10.2B	1
NITO-3D (Nobari et al., 2024)	122K	32K	48K	4.3B	7
OAT3D dataset (ours)	**502K**	64K	**1.05M**	**256B**	**1.5K**

jectives, with applications in aerospace engineering (Zhu et al., 2016), civil engineering (Baandrup et al., 2020), biomedical engineering (Sutradhar et al., 2010), and beyond. Rather than only adjusting the dimensions or boundaries of a predefined layout, topology optimization allows the arrangement and connectivity of material to change, so that structural members and internal voids emerge through optimization (Bendsøe & Kikuchi, 1988; Bendsøe et al., 2004). A common formulation minimizes compliance, seeking a stiff material layout under prescribed loads and fixities for a given material-volume budget. Density-based methods enable gradient-based optimization (Sigmund, 2001; Bendsøe et al., 2004). The design is obtained through an iterative coupling between optimization and physical simulation, which can become computationally demanding as the resolution increases, especially in the three-dimensional setting. Furthermore, full optimization must be rerun for every new loading or fixity configuration.

To reduce the cost of solving many distinct design problems, pretrained models learn to predict material layouts from problem conditions rather than running the full iterative topology optimization procedure (Yu et al., 2019; Nie et al., 2021; Giannone et al., 2023; Lutheran et al., 2025). However, these predictors are largely tied to a fixed discretization. Most 3D models are trained on one or a few grid sizes and aspect ratios (Table 1), so a new domain shape or resolution requires retraining a new model with a different size input. This motivates a large-scale generative model that is shape- and resolution-free: a single model, trained on a broad collection of optimized structures, that designs across varied domains, resolutions, and boundary conditions. Since the model must produce topologies that satisfy mechanical constraints, a generative model for optimal structural design requires coupling geometric representation and mechanical conditions. Such a model requires vast training

examples that capture material distribution variability across problem specifications, but obtaining a diverse collection of data requires repeatedly solving large linear systems, which limits availability of training data.

We introduce Optimize Any Topology-3D (OAT3D), a generative model that combines spatial mechanical conditioning with two-stage latent flow generation (Figure 2). OAT3D generates designs for any rectangular domain and resolution that fits within the grid, which is what we mean by shape- and resolution-free. The first stage generates dense occupancy, while the second generates continuous material density within the occupied regions. To support learning across varied problems, we construct a collection of 501,865 optimized structures[1] on rectangular domains with randomized loads, fixities, dimensions, and prescribed volume fraction targets. This is the largest, most diverse collection of three-dimensional topology optimized structures. We encode loads and supports as proximity fields, and a Balanced Navier-Green (BNG) field supplements these encodings with an auxiliary description of load-support interaction. The trained model can operate at 512^3 resolution with a 45 $\times$ speedup compared to direct topology optimization (Figure 1). We study the linear elastic minimum-compliance problem and evaluate compliance, material usage, and failure rates for designs from single-generation, best-of-N generation, and warm-started optimization. Lastly, we show that the pretrained model can be reused without retraining for downstream design tasks such as joint-count control and lattice replacement.

Our contributions are as follows:

- **Largest dataset of topology optimized 3D structures.** Our dataset contains 501,865 voxelized structures optimized for minimum compliance (maximum stiffness) with randomized mechanical (load and fixity) conditions, prescribed volume fractions from 0.05 to 0.30, and native meshes containing up to approximately one million elements.
- **Shape- and resolution-free generative model for 3D topology optimization.** We develop a conditional generative framework at 512^3 resolution with direct density field tailored to structural design problems. As far as we are aware, this is the first work in shape- and resolution-free generative model of three-dimensional structural topology optimization benefiting from the enormous dataset and recent advances in three-dimensional shape generation models.
- **Guided generation with spatial encoding of mechanical information.** Combined load, support, and BNG field embeddings supply both generation stages with spatial and mechanical information as guidance.

2 Related work

3D Generative Models. Recent 3D generation models advance in both geometric representations and generative capabilities. DreamFusion distills pretrained image diffusion priors into 3D shapes through differentiable rendering (Poole et al., 2023). 3DShape2VecSet develops latent-set representations of neural fields (Zhang et al., 2023). CLAY and TripoSG develop 3D generation using latent diffusion and rectified flow models (Zhang et al., 2024; Li et al., 2025). CAD-oriented shape generation methods such as DeepCAD and BrepGen generate construction sequences or boundary representations that support editability of engineering components (Wu et al., 2021b; Xu et al., 2024), but do not consider the design's physical performance.

More recently, sparse structured representations have enabled detailed 3D generation without modeling every spatial location at the same granularity. TRELLIS first generates a sparse spatial structure and then local latent features at active locations, which can be decoded into multiple 3D representations (Xiang et al., 2025b). Direct3D-S2 improves sparse volumetric generation through spatial sparse attention, while TRELLIS.2 introduces compact structured latents for both geometry and appearance (Wu et al., 2025; Xiang et al., 2025a). This coarse-to-fine and spatially selective modeling is a good fit for low-volume-fraction topology optimization, where material occupies only part of

[1] We generated approximately 1.2 million topology-optimization solutions before filtering, retaining 501,865 topologies. Successful dataset generation is estimated to have required approximately 51,800 L40S-equivalent GPU-hours, extrapolated from 25,996 logged cases with a mean generation time of 155.43 seconds. This estimate excludes failed attempts, startup overhead, and subsequent filtering.

the design domain and fine-scale modeling can be concentrated within occupied regions. However, the objectives of these visual asset generators focus on geometric fidelity and appearance rather than structural performance under prescribed loads, supports, and material budgets. Although latent diffusion has been explored for 3D structural component generation (Herron et al., 2024), high-resolution 3D topology generation conditioned on diverse mechanical boundary conditions remains significantly underexplored relative to visual asset generation.

Topology Optimization: Dataset and Deep Learning Approaches. Learning the relationship between mechanical constraints and optimal structure has been explored through deep learning and generative methods (Yu et al., 2019; Nie et al., 2021; Mazé & Ahmed, 2023), but their generalization to broader domain and resolution sizes are typically limited due to, in part, inherent machine learning architectural limitations, but, majorly, the size, diversity, and availability of topology optimized structure datasets. Table 1 outlines a survey of the scale and diversity of three-dimensional topology optimized structures in existing literature.

Recent efforts have focused on making these predictors less dependent on fixed discretizations and narrowly prescribed problem settings. NITO-3D (Nobari et al., 2024) combines a conditional neural implicit density field with an order-invariant point-set representation of loads and supports, enabling prediction at queried coordinates without requiring simulation-derived input fields, and Nobari et al. (2026) took it further with conditional latent diffusion and a neural-field autoencoder trained on completely randomized two-dimensional problems. Its combination of vast training data and conditional generation provides a basis for learning across diverse structural problems. Building on these advances, we aim to develop a unified model for high-resolution three-dimensional minimum-compliance structures across diverse loading and support configurations.

3 METHODOLOGY

3.1 THREE-DIMENSIONAL TOPOLOGY OPTIMIZATION DATASET GENERATION

We generate three-dimensional topology optimized structures by sampling rectangular domains, loads, supports, and prescribed volume fractions. Each domain contains between 195k and 3.1M degrees of freedom. We minimize compliance (maximize stiffness) under linear elastic physics and a volume constraint using Solid Isotropic Material with Penalization (SIMP) method (Bendsøe, 1989; Bendsøe & Sigmund, 1999). Optimization is performed with Projected Gradient Descent (PGD) (Nobari & Ahmed, 2026), and finite element analysis is done with GPU-acceleration (Träff et al., 2023; Padhi et al., 2023). A detailed description of the minimum compliance problem formulation and dataset sampling is provided in Appendix A and Appendix B, respectively.

3.2 TOPOLOGY AND BOUNDARY CONDITION REPRESENTATION

We use a two-stage (dense/sparse) volumetric representation based on TRELLIS and Direct3D-S2 (Xiang et al., 2025b; Wu et al., 2025). Our generation resolution is limited to 512^3, while Direct3D-S2 enables up to 1024^3 resolution.

Sparse density VAE The sparse stage VAE ($\mathcal{E}_S$) encodes the volume from 512^3 resolution with 8^3 blocks down to sparse $(512/8)^3 = 64^3$ with 16 channel latents. Here, sparse means that only 8^3 blocks with any solid voxels will be encoded. The encoded solid voxels creates the dense occupancy that will further be encoded by the dense VAE.

Dense occupancy VAE The dense VAE ($\mathcal{E}_D$) only encodes dense occupancy for the 64^3 latents provided by the sparse VAE. The occupancy of 64^3 will be encoded into 16^3 with 8 channel latents by the dense VAE. This means that the input to the dense VAE is a *single* channel of 64^3 occupancy instead of 16 from the latent of sparse VAE. The two VAEs can be trained independently: the dense VAE only encodes occupancy, which can be computed directly from the density field without requiring the sparse VAE during training.

Previous work that uses this sparse-dense representation method typically aims to generate surface meshes with no internal structures. In our application, topology optimized structures have complex internals, so we specifically choose to train the sparse VAE to directly represent the continuous densities. Additionally, direct density representation is more efficient in this scenario because it

requires fewer occupancy blocks than a signed distance field (SDF), which is the typical method of mesh representation. In fact, we show empirically in Table 2 that using direct density results in over 33.7% reduction across the mean, median, and 90th percentile quantity in active occupancy blocks.

Boundary condition VAE. OAT3D topology generation with diffusion transformer (DiT) requires the information of the boundary condition. Unfortunately, the boundary condition of topology optimization problems are sparsely populated and spread across multiple channels $\{x, y, z\} \times \{\text{fixity}, +\text{ load direction}, -\text{ load direction}\}$. If we were to directly rasterize them into voxels, directly training a VAE solely based on voxel occupancy to capture boundary condition information would be difficult due to the sparsity. Instead, we represent it as a proximity field (outlined in Appendix C.1). Sparse VAEs $(\mathcal{E}_L, \mathcal{E}_F, \mathcal{E}_B)$ encode these fields into latents that conditions the topology optimization generation. Across 2,000 test cases, proximity encoding increases the mean values within retained 8^3 blocks from 0.0680 to 0.1561 for fixity and from 0.00773 to 0.07761 for loads, providing denser, stronger reconstruction targets to counteract the background void dominance that can favor near-zero VAE reconstructions, while amplitude normalization alone leaves the spatial sparsity unchanged. We further introduce BNG conditioning to approximate the interaction of loads and supports across the domain (with computation detail in Appendix C.1). While existing literature uses strain-energy field through a finite element analysis computation (Nie et al., 2021; Kuszczak et al., 2026), our proposed BNG field approximates this without any finite element analysis.

The boundary conditions are then combined with fixed-length embeddings of the domain dimensions and volume fraction. The complete sequence of embedding conditions the transformers through cross-attention. Additionally, the volume fraction and problem dimension embeddings also condition adaptive layer normalization (Peebles & Xie, 2023). Appendix C.1 gives the boundary condition conditioning field definitions and encoding procedure. We evaluate reconstruction of the conditioning fields to assess how well their spatial information is preserved in latent space. The mean intersection-over-union (IoU) measures are 0.9926 for fixities, 0.9764 for loads, and 0.9459 for BNG. Appendix D gives the evaluation thresholds and detailed results.

3.3 Conditional Generation and Inference

OAT3D generates structures in two stages (Figure 2). Specifically, the dense DiT starts with a Gaussian noise of $16^3 \times 8$ and uses flow matching (Lipman et al., 2023) to denoise it into the dense occupancy latent. It then passes into the dense VAE decoder ($\mathcal{D}_D$) to obtain the dense occupancy of 64^3 (after thresholding the output with 0.3). The dense occupancy resembles a subsampled topology but still lacks the fine detail of the final topology. Next, for each of the occupied dense blocks, we assign a 16 channel Gaussian noise making it sparse $64^3 \times 16$. This 16 channel latent encodes the fine grained detail of the topology. The sparse DiT uses flow matching to denoise the sparse latent at the same sparse $64^3 \times 16$ resolution. Finally, the sparse density decoder ($\mathcal{D}_S$) generates the topology within 512^3 domain.

During both phases of generation, we supply conditioning tokens assembled from the loads, fixities and BNG latents, along with domain dimensions and volume fraction, with cross attention. We train with conditioning dropout enabling classifier-free guidance (CFG) (Ho & Salimans, 2022). Appendix C.2 gives the training objective, inference settings, and post-processing procedure.

4 Experiments

4.1 Model training and ablation

We evaluate reconstruction, BNG conditioning, sampling, and structural performance on 2,000 held out OAT3D problems in the test set. We use 10 flow matching steps with dense CFG 5 and sparse CFG 2.5 with ablation studies in Appendix E. Appendix F gives the experimental setup and model configurations. Appendix D gives the conditioning VAE reconstruction metrics.

Effect of density representation. OAT3D uses density instead of SDF for the sparse VAE representation. The motivation comes from the sparsely populated topology optimized structure. We analyze the number of active blocks between encoding the density field direct versus first setting a threshold of 0.3 then compute a narrow band SDF similar to Direct3D-S2.

Table 2: Active dense occupancy blocks of density vs SDF for test set.

Representation	Mean	Median	90th percentile
Density	**516.55**	**449.5**	**960**
SDF	806.37	734.5	1,449

Table 3: Topology VAE reconstruction on 2,000 problems in the test set. Direct3D-S2 models are from their release.

Representation/ model	Mean IoU ↑	Mean precision ↑	Mean recall ↑
Direct3D-S2, dense	0.9989	0.9993	0.9995
OAT3D, dense	**0.9999**	**0.9999**	**1.0000**
Direct3D-S2, sparse SDF	0.9531	**0.9882**	0.9634
OAT3D, sparse density	**0.9762**	0.9866	**0.9892**

The result in Table 2 shows the comparison for the number of active voxels in the dense occupancy. On average direct density representation uses 35.9% fewer active voxels compared to the SDF representation while retaining the continuous density values without thresholding. We then retrain the dense and sparse VAE based on the density field with results summarized in Table 3. We observe on dense occupancy, OAT3D increases mean IoU from 0.9989 to 0.9999, corresponding to a 90.9% reduction in residual IoU error. For the sparse representation, OAT3D improves mean IoU from 0.9531 to 0.9762, reducing residual IoU error by 49.3%.

Effect of BNG conditioning. We formulated a BNG conditioning field that requires no finite element analysis, in place of a strain energy conditioning field that requires an expensive finite element solve. We examine the effect of BNG by comparing models trained with and without BNG conditioning at matched training epochs. Adding BNG increases mean IoU from 0.5958 to 0.6178 for dense occupancy and from 0.7133 to 0.7183 for sparse density generation (Table 4). These results show that the BNG representation of load and support interactions improves generation accuracy, motivating its use in subsequent evaluations.

Effect on speedup compared to topology optimization. The core of this work is to increase the speed of topology optimization with a structural generative model. We compare the computational cost of OAT3D generation with the recorded topology optimization time for dataset generation. Table 5 shows that generation takes 1.234 seconds per candidate on an H200 GPU. The dataset was generated on L40S GPUs (86.45 s per topology on average); for a same-hardware comparison, we re-ran topology optimization on 100 problems on an H200, averaging 56.63 s. This corresponds to an approximately **45.9**× amortized speedup.

Table 4: BNG conditioning ablation.

Stage	BNG	IoU ↑
Dense	×	0.5958
Dense	✓	**0.6178**
Sparse	×	0.7133
Sparse	✓	**0.7183**

Table 5: Computational cost of OAT3D and dataset.

Measurement	Hardware	Cases	Mean	Median
BNG (ms)	L40S	2,000	36.54	31.94
Load voxel filling (ms)	CPU	2,000	19.61	4.514
Per candidate (s)	H200	307	**1.234**	1.226
Dataset (s)	L40S	501,865	86.45	57.94
Test generation (s)	H200	100	56.63	41.12

4.2 Topology Performance Evaluation Against Other Baseline Models

Metrics. Beyond geometric reconstruction measures, we measure the performance of the generated design: compliance, material usage, and failure for (1) **raw predictions**, (2) **selected candidates**, and (3) **post-optimized designs** under the same problem specifications. These comparisons separate the contributions of generation, candidate selection, and post-optimization.

During evaluation, the input density field is processed as one optimization iteration with the objective evaluated at the end. We do this to ensure consistency between the dataset generation, and evaluation of performance and constraint violation.

For test point i, the signed compliance error (CE) relative to the reference topology is given by: $\mathrm{CE}_i = \left(C_i - C_i^{\mathrm{ref}}\right)/C_i^{\mathrm{ref}}$. The reference is an optimized design without a guarantee of global optimality, so negative errors remain valid. For element volumes v_e, the realized volume fraction and signed relative volume fraction error (VFE) are: $\hat{v}_i = (\sum_e v_e \rho_{i,e})/\sum_e v_e$ and $\mathrm{VFE}_i = (\hat{v}_i - v_i^\star)/v_i^\star$. where $v_i^\star$ is the prescribed volume fraction. Positive VFE denotes excess

material. Tables report the mean signed VFE since under-filling naturally leads to higher compliance due to less material presence.

We report mean and median compliance error, mean signed relative volume error, and the fraction of designs that fail at least one numerical, structural, or volume check. A failed numerical solve is possible because a model can output a nonphysical structure that does not satisfy static equilibrium, which is equivalent to an equality constraint violation in the optimization problem. Structural failure denotes a numerically usable evaluation with $C_i > 2C_i^{\text{ref}}$. We classify a structure as a volume failure when its volume fraction exceeds the prescribed value by more than 0.10 (i.e., $\hat{v}_i - v_i^\star > 0.10$). This volume criterion is one-sided, so under-filling is permitted. Failed generations are excluded from compliance and volume error summaries.

Baselines. The evaluation set comprises 2,000 problems from the OAT3D test set, spanning varying grid dimensions with both point and distributed loading. Each problem specifies constrained displacement components, nodal force vectors, and a prescribed volume fraction, along with a ground-truth voxelized topology and its corresponding compliance measure as reference targets.

We compare OAT3D with three variants of NITO-3D (Nobari et al., 2024): the original model trained on its released dataset, a model retrained on OAT3D dataset, and a parameter-matched model trained on OAT3D dataset. Appendix G documents the NITO-3D training procedure.

Candidate selection and post-optimization (PO). While NITO-3D produces a single deterministic prediction per problem, OAT3D supports inference-time scaling through repeated generative sampling (Nobari et al., 2026). We evaluate best-of-N sampling for $N \in \{1, 2, 4, 8, 16, 32\}$ by selecting the raw prediction with the lowest compliance. The selected candidate is used to warm-start topology optimization. We do up to 15 Optimality Criteria (OC) updates (Bendsøe, 1995; Sigmund, 2001; Andreassen et al., 2011).

4.3 Evaluation Results

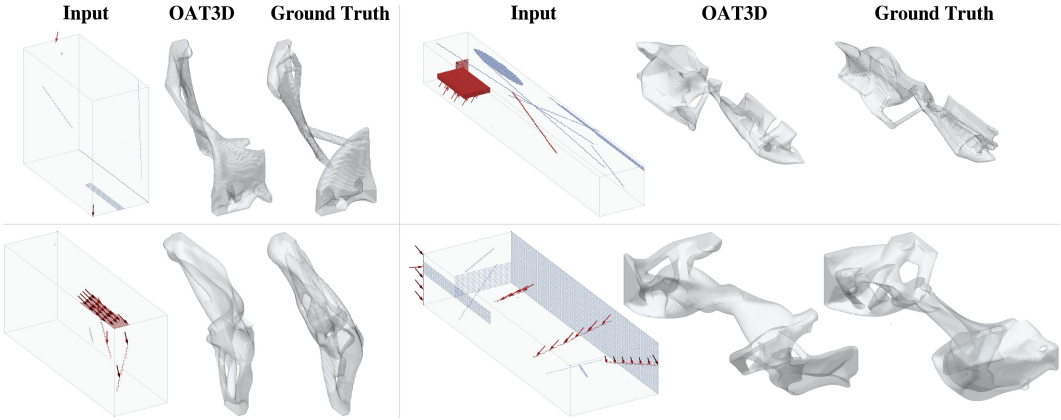


Figure 3: Visualization of (1) The boundary condition input to OAT3D and topology optimization, (2) OAT3D generated topology, (3) Ground truth topology generated from direct optimization.

Single Generation. We select topologies from the test set and visualize them along with the ground truth topologies in Figure 3, 5, and 6. OAT3D has an overall failure rate of 42.40%, compared with 87.00 and 99.50% in the NITO-3D baselines (Table 6a). In terms of mechanical performance, it also performs better than all other baselines. Among eligible designs, OAT3D attains lower mean and median CE than the three baselines, at 31.97% and 23.88%, respectively. Retraining NITO-3D on OAT3D improves its performance, but increasing its width yields only a small marginal improvement.

The original NITO-3D dataset contains one point load per problem and prescribed volume fractions in the range $[0.15, 0.75]$, whereas OAT3D uses a range of $[0.05, 0.30]$. We additionally report a comparison in Appendix G on the 71 evaluation problems from the OAT3D dataset with exactly one loaded node and a prescribed volume fraction in the shared range. In that setting, OAT3D

still significantly outperforms NITO-3D on compliance error and failure measurements across all baselines.

Strikingly, a lower volume fraction range increases the difficulty of the prediction task, demonstrated in a volume-stratification study in Appendix I. The failure rate within the volume fraction range of $[0.05, 0.10]$ is triple that of $[0.20, 0.30]$.

Best-of-N. Table 6b demonstrates that repeated sampling without retraining improves performance. Increasing the candidate budget from $N = 2$ to $N = 32$ reduces overall failure from 34.80% to 19.85%, while mean and median CE decrease from 30.87% and 22.80% to 17.92% and 11.97%, respectively. Selection is also accompanied by a shift in mean signed VFE from -2.18% to 2.17%, so the compliance gains should be interpreted alongside the change in material usage. Figure 7 in Appendix H illustrates the variation in generated structures for a fixed problem, providing qualitative context for the benefit of candidate selection.

Table 6: Performance Evaluation Results: (a) Single-generation, (b) Best-of-N selection from OAT3D, (c) Post-optimized from best-of-2 selection.

Method	N	PO Steps	CE mean (%) ↓	CE median (%) ↓	Mean VFE (%)	Failure (%) ↓
(a) Raw single-generation						
NITO (NITO-trained)	-	-	43.49	41.29	27.42	99.50
NITO (OAT-trained, 23M)	-	-	42.59	40.99	3.75	88.65
NITO (OAT-trained, 811M)	-	-	44.58	41.28	**0.88**	87.00
OAT3D (ours)	-	-	**31.97**	**23.88**	−4.37	**42.40**
(b) Best-of-N Selection from OAT3D						
OAT3D (ours)	2	-	30.87	22.80	−2.18	34.80
OAT3D (ours)	4	-	26.25	18.27	−0.79	29.70
OAT3D (ours)	8	-	22.74	15.99	0.06	24.80
OAT3D (ours)	16	-	20.23	14.30	1.06	22.15
OAT3D (ours)	32	-	**17.92**	**11.97**	2.17	**19.85**
(c) Post-optimized from best-of-2 selection						
OAT3D (ours)	2	5	17.01	12.30	0.00	18.35
OAT3D (ours)	2	10	2.71	1.42	0.00	6.45
OAT3D (ours)	2	15	**1.15**	**0.50**	0.00	**3.35**

Best-of-N + Post Optimization. Post-optimization substantially improves the selected designs (Table 6c). Starting from the best-of-2 prediction, 15 OC updates reduce overall failure from 34.80% to 3.35%, with mean and median CE decreasing from 30.87% and 22.80% to 1.15% and 0.50%, respectively. Much of this improvement is achieved within 10 updates, which yield 6.45% failure and mean CE of 2.71%.

4.4 Application of OAT3D to broader areas of structural optimization

Structural design often requires modifying an optimized structure to address requirements beyond minimum compliance. Because OAT3D generates by integrating a learned flow, sampling can be steered with external guidance, so the same pretrained generator can be reused for these tasks without retraining or fine-tuning. We demonstrate two engineering applications: joint-count control to adjust structural complexity and lattice replacement based on a selected cell type and member thickness. These applications are conventionally complex direct optimization problems, and their demonstration highlights the value of a large generative model for structural optimization.

Structural characteristics control. In density-based topology optimization, structural complexity is typically controlled only implicitly, through the filter radius or length-scale constraints (Lazarov et al., 2016), which alter feature sizes but cannot target a specific characteristic such as the number of joints. Explicit control is difficult because topological quantities such as the number of holes or tunnels are discrete invariants of the design, so existing methods rely on specialized representations, heuristic hole filling, or integer programming (Zhang et al., 2017; Liang et al., 2022; He et al., 2023; Zuo et al., 2023). Joint-count control lets a designer request fewer or more branch junctions in the structural skeleton. We adapt TopoCtrl (Chen et al., 2026b) to OAT3D by training separate joint-count regression model on the dense and sparse latents. Gradients of the prediction error guide generation toward the requested count, while a reference-latent penalty encourages similarity to the

original structure. Figure 4 shows selected examples with fewer or more joints. Appendix J.1 gives the implementation and more results is in Figure 9.

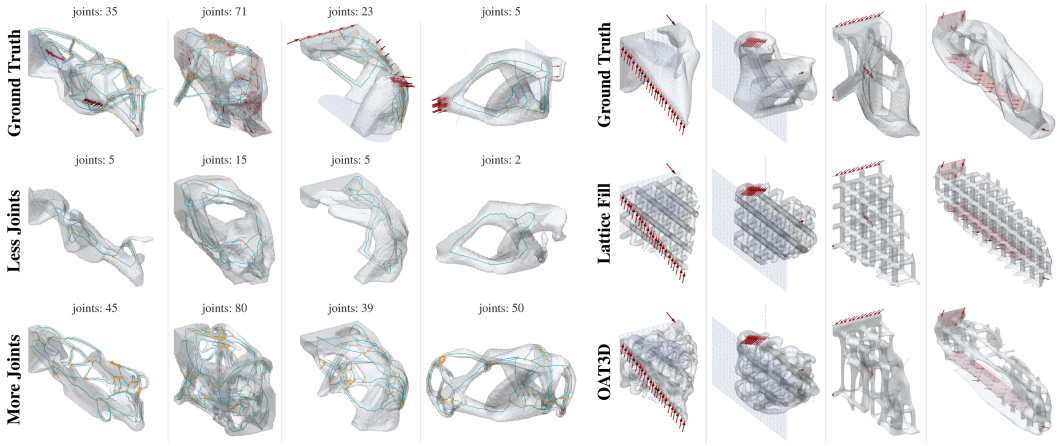


Figure 4: Joint-count control (left) and lattice replacement (right) using OAT3D. Each column shows the reference structure and generated alternatives with fewer and more joints. Orange markers indicate the skeleton junctions used to calculate joint count. Rows show the original structures, direct lattice replacements, and generated results. The first two columns use body-centered cubic lattices; the remaining two use simple cubic lattices.

Lattice replacement. Lattice replacement allows a designer to prescribe cell type and member thickness, but directly substituting a lattice can interrupt the structure because the lattice replacement may miss the load boundary condition completely, resulting in significant increase in compliance. Prior approaches therefore map optimized densities to strut sizes (Alzahrani et al., 2015), optimize porous infill directly at high resolution (Wu et al., 2017), or use homogenization followed by dehomogenization (Groen & Sigmund, 2018; Wu et al., 2021a). Maintaining connectivity between adjacent cells and with the surrounding mechanical inputs remains a central challenge. Following TopoEdit (Chen et al., 2026a), we encode the lattice replacement, partially noise its latents, and denoise them under the original loads and supports with guidance toward the edited structure. Figure 4 compares the original structures, direct replacements, and OAT3D outputs. Appendix J.2 gives the implementation with more results in Figure 8.

5 Discussion and Future Work

We presented OAT3D, a conditional generative model for three-dimensional minimum-compliance topology optimization. OAT3D combines a dataset of 501,865 optimized structures, a two-stage density representation, and learned boundary-condition encodings augmented by BNG conditioning. We also demonstrate how generation can be guided toward a selected structural characteristic, providing a way to control design features that are difficult to impose directly in conventional density-based optimization.

The present work does have limitations: the dataset does not include prescribed solid or void regions that must remain unchanged during optimization (passive regions). We study linear elastic structures with a single objective. Multi-physics and multi-objective optimization are outside the present scope. Fixity regions act along the global coordinate axes and do not account for rigid kinematic coupling constraints.

Future work will extend the framework to additional design constraints, including requirements for casting, welded assemblies, and sheet-metal fabrication. Developing representations that facilitate conversion of generated density fields into editable CAD models is another direction toward practical structural design. The pretrained model could be extended into other topology optimization problem formulations and physics such as the finite-strain formulation.

AI USE STATEMENT

We used generative AI tools during the initial exploration of related work and to suggest an initial organization for the plots. AI tools assisted with writing code for training the machine learning models and with deploying training and evaluation code on a compute cluster. We also used them to proofread grammar, suggest table layouts, and generate preliminary drafts of some sections. AI was used to polish and refine writing of this manuscript. The authors take responsibility for the final manuscript, code, and reported results.

ETHICS STATEMENT

This work involves no human subjects and no personal data. All topology optimization data are generated using computer simulation. The generated topology both from the dataset and the OAT3D model cannot replace real world structural engineering. Even with compliance analysis, the generated design cannot be guaranteed to be manufactured and manufactured design may not satisfy certification without more rigorous validation.

REPRODUCIBILITY STATEMENT

We will release the OAT3D dataset of 501,865 topology-optimized structures and the trained model checkpoint upon paper acceptance. Each example contains the problem specification (domain size, load and fixities, volume fraction), the optimized density field. Every problem is generated with a deterministic 16 bit hexadecimal seed. Together with the sampling procedure in Appendix B, the seed is fully deterministic. We will also release the model evaluation code with all result generated with fixed sampling seeds. The NITO-3D baselines are retrained from the authors' public code and dataset, and Appendix G documents the changes for the 23M and 811M variants and compares our reproduction with the published results. Appendix J gives the regressors, guidance weights, and seeds for the application experiments.

REFERENCES

Mahmoud Alzahrani, Seung-Kyum Choi, and David W Rosen. Design of truss-like cellular structures using relative density mapping method. *Materials & Design*, 85:349–360, 2015.

Erik Andreassen, Anders Clausen, Mattias Schevenels, Boyan S Lazarov, and Ole Sigmund. Efficient topology optimization in matlab using 88 lines of code. *Structural and Multidisciplinary Optimization*, 43(1):1–16, 2011.

Mads Baandrup, Ole Sigmund, Henrik Polk, and Niels Aage. Closing the gap towards super-long suspension bridges using computational morphogenesis. *Nature communications*, 11(1):2735, 2020.

Mohammad Mahdi Behzadi and Horea T Ilieş. Real-time topology optimization in 3d via deep transfer learning. *Computer-Aided Design*, 135:103014, 2021.

Martin P Bendsøe. Optimal shape design as a material distribution problem. *Structural optimization*, 1(4):193–202, 1989.

Martin P Bendsøe. *Optimization of structural topology, shape, and material*, volume 414. Springer, 1995.

Martin P Bendsøe and Ole Sigmund. Material interpolation schemes in topology optimization. *Archive of applied mechanics*, 69(9):635–654, 1999.

Martin P Bendsøe, Ole Sigmund, et al. *Topology optimization: theory, methods, and applications*, volume 2. Springer Berlin, 2004.

Martin Philip Bendsøe and Noboru Kikuchi. Generating optimal topologies in structural design using a homogenization method. *Computer methods in applied mechanics and engineering*, 71 (2):197–224, 1988.

Hongrui Chen, Josephine V. Carstensen, and Faez Ahmed. TopoEdit: Fast post-optimization editing of topology optimized structures. *arXiv preprint arXiv:2602.22430*, 2026a. doi: 10.48550/arXiv.2602.22430. URL https://arxiv.org/abs/2602.22430.

Hongrui Chen, Dat Quoc Ha, Josephine V. Carstensen, and Faez Ahmed. TopoCtrl: Post-optimization topology editing toward target structural characteristics. *arXiv preprint arXiv:2603.26926*, 2026b. doi: 10.48550/arXiv.2603.26926. URL https://arxiv.org/abs/2603.26926.

S Dittmer, D Erzmann, H Harms, and P Maass. Selto: Sample-efficient learned topology optimization. preprint. *arXiv*, 2209, 2022.

Giorgio Giannone, Akash Srivastava, Ole Winther, and Faez Ahmed. Aligning optimization trajectories with diffusion models for constrained design generation. *Advances in neural information processing systems*, 36:51830–51861, 2023.

Jeroen P Groen and Ole Sigmund. Homogenization-based topology optimization for high-resolution manufacturable microstructures. *International Journal for Numerical Methods in Engineering*, 113(8):1148–1163, 2018.

Yunzhen He, Zi-Long Zhao, Xiaoshan Lin, and Yi Min Xie. A hole-filling based approach to controlling structural complexity in topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 416:116391, 2023.

Ethan Herron, Jaydeep Rade, Anushrut Jignasu, Baskar Ganapathysubramanian, Aditya Balu, Soumik Sarkar, and Adarsh Krishnamurthy. Latent diffusion models for structural component design. *Computer-Aided Design*, 171:103707, 2024.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. doi: 10.48550/arXiv.2207.12598. URL https://arxiv.org/abs/2207.12598.

Vahid Keshavarzzadeh, Mitra Alirezaei, Tolga Tasdizen, and Robert M Kirby. Image-based multiresolution topology optimization using deep disjunctive normal shape model. *Computer-Aided Design*, 130:102947, 2021.

Igor Kuszczak, Gaweł Kuś, Federico Bosi, and Miguel A Bessa. Meta-neural topology optimization: knowledge infusion with meta-learning. *Engineering Applications of Artificial Intelligence*, 182: 115763, 2026.

Boyan S Lazarov, Fengwen Wang, and Ole Sigmund. Length scale and manufacturability in density-based topology optimization. *Archive of Applied Mechanics*, 86(1):189–218, 2016.

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, and Yan-Pei Cao. TripoSG: High-fidelity 3D shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608*, 2025. URL https://arxiv.org/abs/2502.06608.

Yuan Liang, XinYu Yan, and GengDong Cheng. Explicit control of 2d and 3d structural complexity by discrete variable topology optimization method. *Computer Methods in Applied Mechanics and Engineering*, 389:114302, 2022.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Aaron Lutheran, Srijan Das, and Alireza Tabarraei. Latent space diffusion for topology optimization. *arXiv preprint arXiv:2508.05624*, 2025.

François Mazé and Faez Ahmed. Diffusion models beat gans on topology optimization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 9108–9116, 2023.

Zhenguo Nie, Tong Lin, Haoliang Jiang, and Levent Burak Kara. Topologygan: Topology optimization using generative adversarial networks based on physical fields over the initial domain. *Journal of Mechanical Design*, 143(3):031715, 2021.

Amin Heyrani Nobari and Faez Ahmed. Pgd-to: a scalable alternative to mma using projected gradient descent for multi-constraint topology optimization. *Structural and Multidisciplinary Optimization*, 69(9):209, 2026.

Amin Heyrani Nobari, Lyle Regenwetter, and Faez Ahmed. Towards domain-adaptive, resolution-free 3d topology optimization with neural implicit fields. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 88360, pp. V03AT03A012. American Society of Mechanical Engineers, 2024.

Amin Heyrani Nobari, Lyle Regenwetter, Cyril Picard, Ligong Han, and Faez Ahmed. Optimize any topology: a foundation model for shape-and resolution-free structural topology optimization. *Advances in Neural Information Processing Systems*, 38:153374–153419, 2026.

Arya Prakash Padhi, Souvik Chakraborty, Anupam Chakrabarti, and Rajib Chowdhury. Efficient hybrid topology optimization using gpu and homogenization-based multigrid approach. *Engineering with Computers*, 39(5):3593–3615, 2023.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Peebles_Scalable_Diffusion_Models_with_Transformers_ICCV_2023_paper.html.

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D diffusion. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.14988.

Ole Sigmund. A 99 line topology optimization code written in matlab. *Structural and multidisciplinary optimization*, 21(2):120–127, 2001.

Alok Sutradhar, Glaucio H Paulino, Michael J Miller, and Tam H Nguyen. Topological optimization for designing patient-specific large craniofacial segmental bone replacements. *Proceedings of the National Academy of Sciences*, 107(30):13222–13227, 2010.

Erik A Träff, Anton Rydahl, Sven Karlsson, Ole Sigmund, and Niels Aage. Simple and efficient gpu accelerated topology optimisation: Codes and applications. *Computer Methods in Applied Mechanics and Engineering*, 410:116043, 2023.

Jun Wu, Niels Aage, Rüdiger Westermann, and Ole Sigmund. Infill optimization for additive manufacturing—approaching bone-like porous structures. *IEEE transactions on visualization and computer graphics*, 24(2):1127–1140, 2017.

Jun Wu, Ole Sigmund, and Jeroen P Groen. Topology optimization of multi-scale structures: a review. *Structural and Multidisciplinary Optimization*, 63(3):1455–1480, 2021a.

Rundi Wu, Chang Xiao, and Changxi Zheng. DeepCAD: A deep generative network for computer-aided design models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021b. URL https://openaccess.thecvf.com/content/ICCV2021/html/Wu_DeepCAD_A_Deep_Generative_Network_for_Computer-Aided_Design_Models_ICCV_2021_paper.html.

Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Yikang Yang, Yajie Bao, Jiachen Qian, Siyu Zhu, Xun Cao, Philip Torr, and Yao Yao. Direct3D-S2: Gigascale 3D generation made easy with spatial sparse attention. *arXiv preprint arXiv:2505.17412*, 2025. doi: 10.48550/arXiv.2505.17412. URL https://arxiv.org/abs/2505.17412.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, and Jiaolong Yang. Native and compact structured latents for 3D generation. *arXiv preprint arXiv:2512.14692*, 2025a. URL https://arxiv.org/abs/2512.14692.

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3D latents for scalable and versatile 3D generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21469–21480, 2025b. URL https://openaccess.thecvf.com/content/CVPR2025/papers/Xiang_Structured_3D_Latents_for_Scalable_and_Versatile_3D_Generation_CVPR_2025_paper.pdf.

Xiang Xu, Joseph G. Lambourne, Pradeep Kumar Jayaraman, Zhengqing Wang, Karl D. D. Willis, and Yasutaka Furukawa. BrepGen: A b-rep generative diffusion model with structured latent geometry. *ACM Transactions on Graphics*, 2024. URL https://arxiv.org/abs/2401.15563.

Yonggyun Yu, Taeil Hur, Jaeho Jung, and In Gwun Jang. Deep learning for determining a near-optimal topological design without any iteration: Y. yu et al. *Structural and Multidisciplinary Optimization*, 59(3):787–799, 2019.

Biao Zhang, Jiapeng Tang, Matthias Nießner, and Peter Wonka. 3DShape2VecSet: A 3D shape representation for neural fields and generative diffusion models. *ACM Transactions on Graphics*, 2023. URL https://arxiv.org/abs/2301.11445.

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. CLAY: A controllable large-scale generative model for creating high-quality 3D assets. *ACM Transactions on Graphics*, 2024. URL https://arxiv.org/abs/2406.13897.

Weisheng Zhang, Ying Liu, Peng Wei, Yichao Zhu, and Xu Guo. Explicit control of structural complexity in topology optimization. *Computer Methods in Applied Mechanics and Engineering*, 324:149–169, 2017.

Ji-Hong Zhu, Wei-Hong Zhang, and Liang Xia. Topology optimization in aircraft and aerospace structures design. *Archives of computational methods in engineering*, 23(4):595–622, 2016.

Tongxing Zuo, Haitao Han, and Zhenyu Liu. Explicit tunnels and cavities control using simp and mma in structural topology optimization. *Computer-Aided Design*, 158:103482, 2023.

A Topology Optimization and Compliance Minimization Problem Formulation for Dataset Generation

We use a density-based formulation with Solid Isotropic Material with Penalization (SIMP). The design variables $\boldsymbol{\phi} \in [0,1]^{n_d}$ determine material allocation. They are filtered with a spatial density filter, $\mathbf{H}(\boldsymbol{\phi})$, and the resulting densities are assigned to each finite element. We consider compliance minimization subject to mechanical equilibrium and a material-volume constraint:

$$\begin{aligned} \min_{\boldsymbol{\phi}\in[0,1]^{n_d}} \quad & C(\boldsymbol{\phi}) = \mathbf{f}^{\mathsf{T}}\mathbf{u} \\ \text{subject to} \quad & \mathbf{K}(\mathbf{H}(\boldsymbol{\phi}))\,\mathbf{u} = \mathbf{f} \\ & V(\boldsymbol{\phi}) \le v^{\star} \end{aligned} \tag{1}$$

where

$$V(\boldsymbol{\phi}) = \frac{\sum_{e=1}^{n_d} v_e \phi_e}{\sum_{e=1}^{n_d} v_e} \tag{2}$$

Here, $\mathbf{u}$ is the displacement vector for all degrees of freedom in the static system, $\mathbf{f}$ is the assembled force vector, v_e is the volume of element e, and $v^{\star}$ is the prescribed volume fraction. The equilibrium system incorporates zero Dirichlet conditions on the constrained displacement degrees of freedom.

Density filtering and material interpolation. The relative densities are computed using a normalized cone filter:

$$\begin{aligned} \rho_e = [\mathbf{H}(\boldsymbol{\phi})]_e &= \frac{\sum_{j=1}^{n_d} w_{e,j}\phi_j}{\sum_{j=1}^{n_d} w_{e,j}} \\ w_{e,j} &= \max\left(0, 1 - \frac{\|\mathbf{q}_e - \mathbf{q}_j\|_2}{r_{\min}}\right) \end{aligned} \tag{3}$$

Here, $\mathbf{q}_e$ denotes the center of element e, measured in element-width units. We set $r_{\min} = \sqrt{3}$.

We assume isotropic linear elasticity with Young's modulus $E_0 = 1$ and Poisson's ratio $\nu = 1/3$. The filtered densities determine the element stiffness matrices through

$$\mathbf{K}_e(\rho_e) = \max(\rho_e^p, \epsilon)\,\mathbf{K}_e^{\text{solid}}, \qquad p = 3, \qquad \epsilon = 10^{-9} \tag{4}$$

where $\mathbf{K}_e^{\text{solid}}$ is the stiffness matrix of the solid element. The stiffness floor ϵ supplies a small residual stiffness to prevent the stiffness matrix from becoming singular. We keep the penalization exponent fixed throughout optimization and apply SIMP directly to the filtered densities.

Optimization. We initialize the design uniformly at $\phi_e^{(0)} = v^{\star}$ and optimize it using pyFANTOM's GPU implementation of projected gradient descent, with adaptive step sizes and conjugate search directions. Each update is projected onto the box and volume constraints. Mechanical equilibrium is solved using multigrid-preconditioned conjugate gradients, with a target relative residual of 10^{-5} and at most 400 solver iterations per solve. The residual is recomputed as

$$r_{\text{rel}} = \frac{\|\mathbf{f} - \mathbf{K}\mathbf{u}\|_2}{\|\mathbf{f}\|_2} \tag{5}$$

Optimization terminates after at most 500 updates, or when both the following criteria are satisfied:

$$\begin{aligned} \frac{|C^{(t)} - C^{(t-1)}|}{|C^{(t)}|} &< 10^{-5} \\ \frac{\|\boldsymbol{\phi}^{(t)} - \boldsymbol{\phi}^{(t-1)}\|_2}{\|\boldsymbol{\phi}^{(t)}\|_2 + 10^{-12}} &< 10^{-2} \end{aligned} \tag{6}$$

B Dataset Sampling

We generate three-dimensional topology optimization problems by randomly sampling the domain dimensions, prescribed volume fraction, loads, and displacement constraints. Each instance has a hexadecimal seed that reproduces its sampled problem configuration. Below, $\mathcal{U}[a, b]$ denotes a continuous uniform distribution, $\mathcal{U}\{a, \dots, b\}$ denotes a uniform distribution over the integers from a to b, inclusive, and $\mathrm{Bernoulli}(p)$ returns one with probability p.

B.1 Domain Definition and Discretization

Each design domain is a cuboid discretized into uniform eight-node hexahedral elements. We determine its dimensions as follows:

1. **Target element count.** We sample an element budget
$$B \sim \mathcal{U}\{2^{16}, \dots, 2^{20}\}. \tag{7}$$
2. **Axis counts.** With a minimum of $m = 32$ elements along each axis, we sample two preliminary dimensions and calculate the third from the remaining budget:
$$\begin{aligned} \widetilde{n}_x &\sim \mathcal{U}\{m, \dots, \lfloor B/m^2 \rfloor\}, \\ \widetilde{n}_y \mid \widetilde{n}_x &\sim \mathcal{U}\{m, \dots, \lfloor B/(m\widetilde{n}_x) \rfloor\}, \\ \widetilde{n}_z &= \left\lfloor \frac{B}{\widetilde{n}_x \widetilde{n}_y} \right\rfloor. \end{aligned} \tag{8}$$
We round each count upward to a multiple of eight, $n_i = 8\lceil \widetilde{n}_i/8 \rceil$, for $i \in \{x, y, z\}$. If $n_x n_y n_z \notin [2^{16}, 2^{20}]$, the configuration proposal is restarted. Rounding can cause the final element count to differ from B.
3. **Physical dimensions.** We normalize the longest dimension to one.
$$s = \max(n_x, n_y, n_z) \tag{9}$$
$$(L_x, L_y, L_z) = \frac{(n_x, n_y, n_z)}{s}. \tag{10}$$
Each element is a cube of edge length $1/s$. Geometry is sampled in integer node coordinates $\mathbf{z} = (z_x, z_y, z_z)$, where $0 \le z_i \le n_i$. Physical coordinates can be retrieved with $\mathbf{z}/s$.
4. **Material and volume fraction.** We use one isotropic material with solid Young's modulus $E_0 = 1$ and Poisson's ratio $\nu = 1/3$. The prescribed volume fraction is sampled as
$$v^\star \sim \mathcal{U}[0.05, 0.30]. \tag{11}$$

B.2 Load and Constraint Specification

Loads and displacement constraints are assigned to regions of mesh nodes. A region can consist of a point, line, surface patch, or volume, and may contain many nodes. We first sample the numbers of regions, then their locations and geometries, and finally the force vectors or constrained displacement components.

1. **Numbers of regions.** The numbers of load regions and fixity regions are
$$N_L \sim \mathcal{U}\{1, \dots, 10\}, \qquad N_S \sim \mathcal{U}\{3, \dots, 10\}. \tag{12}$$
These count regions, rather than individual loaded nodes or constrained degrees of freedom.
2. **Interior or exterior placement.** An interior region contains only nodes with $1 \le z_i \le n_i - 1$ in every coordinate direction. An exterior region lies on the domain boundary, where at least one coordinate equals zero or the corresponding axis count. Each load region is assigned to the interior with probability 0.25. Each fixity region with is assigned to the interior with probability 0.10. Otherwise, it is assigned to the exterior.
3. **Interior geometry.** Conditional on interior placement, the following four types are equally likely:

- **Point:** Sample a node uniformly from the interior, with independent coordinates $z_i \sim \mathcal{U}\{1, \ldots, n_i - 1\}$.
- **Line:** Sample two distinct interior nodes and discretize the segment between them.
- **Surface:** Choose a rectangle or ellipse with equal probability. Sample its center coordinates from $\mathcal{U}[1, n_i - 1]$ and its normal by normalizing a vector with independent components from $\mathcal{U}[-1, 1]$. Construct two perpendicular unit vectors in the patch plane and independently sample the full lengths along them from $\mathcal{U}[2, \min(n_x, n_y, n_z)/2]$. Sample points across the patch and round them to grid nodes and remove duplicates.
- **Volume:** Choose an axis-aligned cuboid or ellipsoid with equal probability. Independently sample its center coordinates from $\mathcal{U}[2, n_i - 2]$ and the distance from the center to the edge of the shape along each axis from $\mathcal{U}[2, 0.3n_i]$, then enumerate the grid nodes belonging to the discretized region.

For interior surfaces and volumes, duplicate nodes are removed and only strictly interior nodes are retained.

4. **Exterior geometry.** Conditional on exterior placement, points, lines, and surfaces are equally likely. Faces are selected uniformly from the six domain faces, without weighting by area.
   - **Point:** With probability 0.25, select one of the eight corners uniformly. Otherwise, select a face uniformly and then a node uniformly on that face, including its edges and corners.
   - **Line:** With probability 0.25, select one of the twelve full domain edges uniformly and include all its nodes. Otherwise, sample two distinct nodes uniformly on a selected face and discretize their segment using the same procedure as for interior lines.
   - **Surface:** With probability 0.25, include an entire selected face. Otherwise, choose a rectangular or elliptical patch with equal probability. Along each in-plane axis i of a selected face, sample the center from $\mathcal{U}[1, n_i - 1]$ and the half-extent from $\mathcal{U}[2, n_i/2]$. Construct the patch on the face's integer grid and trim it to the face boundaries.
5. **Load directions and magnitudes.** For each load region r, we sample a vector $\mathbf{g}_r$ whose three components are independently drawn from $\mathcal{U}[-0.5, 0.5]$. We normalize this vector to obtain the unit load direction:
$$\mathbf{d}_r = \frac{\mathbf{g}_r}{\|\mathbf{g}_r\|_2}. \tag{13}$$
We then independently sample a magnitude $a_r \sim \mathcal{U}[0, 1]$ for each region and divide it by the largest sampled magnitude in the configuration:
$$\alpha_r = \frac{a_r}{\max(a_1, \ldots, a_{N_L})}. \tag{14}$$
Let $m_r = |\mathcal{R}_r|$ denote the number of nodes in load region $\mathcal{R}_r$. Each node in this region receives the force vector $\alpha_r \mathbf{d}_r / m_r$. Thus, before removing constrained components, each region has resultant magnitude α_r, independent of the node count in a region.
6. **Constrained displacement components.** Each fixity region receives a mask $\mathbf{b}_r = (b_{r,x}, b_{r,y}, b_{r,z})$, with independent entries
$$b_{r,i} \sim \text{Bernoulli}(0.5). \tag{15}$$
The mask is shared by all nodes in the region. An entry of one imposes a zero Dirichlet condition on that displacement component; an entry of zero leaves it unconstrained by this region. A fixity node need not have all three degrees of freedom fixed. Overlapping fixity regions impose the union of their constraints.

Force components on constrained degrees of freedom are set to zero, and load rows that become entirely zero are removed. At least one nonzero load row must remain. Forces are not renormalized after this step.

C Topology Representation and Conditional Generation

C.1 Problem Representation

Boundary-condition proximity fields. We encode boundary conditions (BCs) using three fields for constrained displacement components and six nonnegative fields for signed force directions. For a source at position $\mathbf{x}_i$, the proximity kernel is $k_R(r) = \max(1 - r/R, 0)$, where $R = 5$ native-grid spacings. The fixity proximity field for axis a and load field for signed direction d are

$$b_a(\mathbf{x}) = \max_{i \in \mathcal{B}_a} k_R(\|\mathbf{x} - \mathbf{x}_i\|_2), \qquad \ell_d(\mathbf{x}) = \max_{i \in \mathcal{F}_d} k_R(\|\mathbf{x} - \mathbf{x}_i\|_2)^{p(m_i)}. \tag{16}$$

Here, $\mathcal{B}_a$ contains nodes constrained in displacement component a, and $\mathcal{F}_d$ contains force components with signed direction d. Forces are normalized by the largest absolute component in each problem. The normalized component magnitude m_i is clipped to $[10^{-6}, 1]$ and sets the exponent $p(m) = 3 - 2.7(m - 10^{-6})/(1 - 10^{-6})$. Larger force values give a slower spatial decay within the same radius. The field value at a source remains one, and an empty source set gives a zero field.

A shared fixity proximity field encoder and a shared load proximity field encoder process active directions separately. We assemble their latents by spatial coordinate and direction to retain the information of each constrained or loaded component. The two boundary condition encoders operate at 513^3 resolution as the boundary conditions for topology optimization problems are defined at the nodes while the densities are represented at the voxels. The resulting sparse latent resolution is instead 65^3 with 16 channel latent. The benefit of using sparse VAE to encode the boundary condition is that the number of conditioning tokens required to encode varies with the complexity of the boundary conditions and on average load and fixity requires 483.29 tokens to encode.

Balanced Navier-Green conditioning. BNG conditioning field serves as the surrogate of strain energy field requiring expensive finite element analysis. A discrete Navier-Green operator, constructed from the zero-mean inverse Neumann Laplacian, maps the balanced load field to an auxiliary displacement field $\widetilde{\mathbf{u}}$. Because the prescribed loads alone are not in equilibrium, we first add the minimum-norm support reactions satisfying the six force and moment balance equations. We calculate the strain-energy-density surrogate

$$W = \mu\, \boldsymbol{\varepsilon}(\widetilde{\mathbf{u}}) : \boldsymbol{\varepsilon}(\widetilde{\mathbf{u}}) + 0.5\lambda \big[\operatorname{tr} \boldsymbol{\varepsilon}(\widetilde{\mathbf{u}})\big]^2, \tag{17}$$

where λ and μ are the Lamé parameters and $\boldsymbol{\varepsilon}$ is the symmetric discrete displacement gradient. A clipped power transform of the median-normalized energy gives the scalar BNG field. The construction uses the loads and supports without finite element analysis of a candidate structure.

We encode BNG over the native domain with a sparse VAE, including values below the reconstruction threshold. Each conditioning token packs up to 4^3 latent sites with 12 channels per site. Table 7 gives the token counts. Adding BNG increases sequence length by a mean of 9.81%, computed from the relative increase for each problem.

Table 7: Conditioning-token counts on 2,000 problems. L_0 contains the BC tokens and four scalar tokens; L_1 also contains packed BNG tokens.

Quantity	Mean	Median	90th percentile
BC and scalar sequence, L_0	483.29	405	913
Added BNG tokens	37.22	32	64
Complete sequence, L_1	520.52	442	958.3
Added length relative to L_0 (%)	9.81	7.60	17.95

Conditioning interface. We combine the BC and BNG tokens with four global tokens for the native domain dimensions and volume fraction. The transformers use cross-attention over the complete sequence, and the global tokens also condition adaptive layer normalization (Peebles & Xie, 2023).

C.2 Conditional Generation and Inference

We train the dense latent generator for dense occupancy and the sparse latent generator for density within active blocks. Sparse training uses occupancy derived from reference densities. At inference, we retain dense cells whose decoded sigmoid probability exceeds 0.3.

Both transformers are trained by flow matching (Lipman et al., 2023). For normalized target latent z_{data}, Gaussian noise ϵ, and noise level σ, the noisy latent and training objective are

$$\begin{aligned} z_\sigma &= (1-\sigma) z_{\mathrm{data}} + \sigma\epsilon, \\ \mathcal{L}_{\mathrm{flow}} &= \mathbb{E}\Big[\| u_\theta(z_\sigma, \sigma, c) - (\epsilon - z_{\mathrm{data}}) \|_2^2 / D_z \Big], \end{aligned} \tag{18}$$

where c is the encoded problem condition and D_z is the number of latent scalar entries in that problem.

Condition dropout enables classifier-free guidance (Ho & Salimans, 2022). BNG is also dropped whenever fixity-constraint or load information is dropped. The guided velocity is

$$u_{\mathrm{CFG}} = u_\theta(z_\sigma, \sigma, c_\varnothing) + w\big[u_\theta(z_\sigma, \sigma, c) - u_\theta(z_\sigma, \sigma, c_\varnothing)\big], \tag{19}$$

where $c_\varnothing$ is the dropped-condition input. Euler integration runs from noise to data with decreasing σ.

Load voxel filling. After sparse decoding, we apply load voxel filling to the generated density field. For each original load region, a density-based score selects an individual load feature (for example the points, lines, surface patches and volumes), and the corresponding voxels are set to solid.

D RECONSTRUCTION AND CONDITIONING

Table 8 reports reconstruction of the condition fields. Fixity and load proximity field encodings reach mean IoUs of 0.9926 and 0.9764 at a threshold of 0.9. BNG reaches 0.9459 at a threshold of 0.1, with native-domain mean absolute error 0.00550 and root mean squared error 0.00946.

Table 8: Conditioning-field reconstruction on 2,000 problems. BC metrics average active scalar directions within each problem, then average problems. BNG is evaluated over its native domain.

Field	Threshold	Mean IoU	Mean precision	Mean recall
Fixity proximity	≥ 0.9	0.9926	0.9947	0.9976
Load proximity	≥ 0.9	0.9764	0.9864	0.9894
BNG	≥ 0.1	0.9459	0.9658	0.9791

E Sampling Steps and Classifier-Free Guidance

We vary Euler steps over $\{10, 20, 30, 50\}$ and CFG weights over $\{0.1, 1, 2.5, 5, 10, 20\}$. Table 9 shows dense occupancy IoU. At 10 steps, increasing CFG from 5 to 10 changes mean IoU from 0.6178 to 0.6185. Table 10 reports sparse material IoU and binary volume-fraction MAE. At 10 steps, sparse CFG 1 gives slightly higher IoU than CFG 2.5 (0.7194 versus 0.7183), while CFG 2.5 gives lower volume-fraction MAE (0.8414 versus 0.8494 percentage points). At sparse CFG 2.5, all larger step counts give lower mean IoU than 10 steps.

Table 9: Dense occupancy IoU with BNG across Euler steps and CFG weights.

Steps / CFG	0.1	1	2.5	5	10	20
10	0.1667	0.5815	0.6121	0.6178	0.6185	0.6089
20	0.1630	0.5754	0.6088	0.6155	0.6156	0.6082
30	0.1632	0.5736	0.6075	0.6149	0.6146	0.6070
50	0.1673	0.5717	0.6064	0.6143	0.6141	0.6071

Table 10: Sparse generation with BNG across Euler steps and CFG weights. Binary volume-fraction MAE is in percentage points relative to reference volume fraction (VF).

Steps	Metric / CFG	0.1	1	2.5	5	10	20
10	IoU ↑	0.6086	0.7194	0.7183	0.6987	0.6601	0.5372
10	VF MAE ↓	1.9581	0.8494	0.8414	1.0193	1.4274	2.2281
20	IoU ↑	0.5958	0.7086	0.7110	0.6946	0.6658	0.6114
20	VF MAE ↓	2.0368	0.8334	0.8507	1.0170	1.4097	2.2281
30	IoU ↑	0.5913	0.7045	0.7081	0.6925	0.6650	0.6224
30	VF MAE ↓	2.0686	0.8351	0.8557	1.0188	1.4051	2.2368
50	IoU ↑	0.5907	0.7024	0.7059	0.6908	0.6641	0.6253
50	VF MAE ↓	2.0788	0.8378	0.8616	1.0263	1.4243	2.2765

F EXPERIMENTAL SETUP

Data and models. Reconstruction and generation comparisons use the first 2,000 reserved problems. We reconstruct with VAE posterior modes and generate with exponential-moving-average (EMA) transformer weights. Dense models are trained for 31 epochs and sparse models for 40 epochs. Unless a parameter is varied, sampling uses 10 Euler steps, scheduler shift 6, dense CFG 5, and sparse CFG 2.5. Generation comparisons use one draw per problem with a fixed seed. Sparse generation comparisons use reference-derived dense occupancy to evaluate.

Reconstruction and generation metrics. We report means of IoU, precision, and recall, giving each problem equal weight. Dense occupancy is selected at sigmoid probability > 0.3 and material occupancy at native density > 0.3. Fixity and load fields use a localization threshold of ≥ 0.9, and BNG uses ≥ 0.1. The sparse SDF comparison uses its signed zero level. Binary volume-fraction error compares predicted and reference volume fraction:

$$\mathrm{MAE}_{v,\mathrm{bin}} = \frac{100}{N} \sum_{i=1}^{N} \left| \frac{N_i^{\mathrm{pred}} - N_i^{\mathrm{ref}}}{n_{x,i} n_{y,i} n_{z,i}} \right|, \tag{20}$$

where N_i^{pred} and N_i^{ref} count occupied elements on the native mesh. This error is in percentage points.

G NITO-3D BASELINE TRAINING

We train three NITO-3D baselines following the original training setup of Nobari et al. (2024), except where noted below.

NITO-3D trained on its original dataset. Since the original work does not supply model checkpoints and training splits, we retrain NITO-3D using the authors' released dataset and code, and split the data using a random seed. Table 11 compares our reproduction with the published results.

Exclusively for Table 11, we evaluate the performance of the NITO-3D model using NITO-3D's evaluation protocol, which is not identical to the one presented in the main body of this paper. Predictions are binarized at $\rho > 0.5$ and evaluated before and after five and ten direct-optimization steps. Problems with $|\mathrm{CE}| \geq 1{,}000\%$ are excluded from statistical summaries.

Table 11: NITO-3D reproduction using the authors' public code and an independent test split. Published values are from Table 2 of Nobari et al. (2024). All errors are percentages: CE is signed and VFE is unsigned.

Stage	Source	CE mean (%) ↓	CE median (%) ↓	|VFE| mean (%) ↓	|VFE| median (%) ↓
Raw	Published	5.95	0.32	4.38	2.40
Raw	Retrained	6.552	0.414	4.393	2.755
5 steps	Published	0.52	0.11	0.96	0.81
5 steps	Retrained	0.267	0.111	0.939	0.795
10 steps	Published	0.31	0.077	0.67	0.54
10 steps	Retrained	0.150	0.071	0.608	0.532

NITO-3D trained on OAT3D dataset. We train the original architecture, with approximately 23M parameters, on the OAT3D training set with a held out validation set. We select the checkpoint with the lowest validation loss, at epoch 9.

Parameter-matched NITO-3D trained on OAT3D dataset. We widen the original architecture without changing its depth, yielding approximately 811M parameters. This model uses the same data and training settings as the 23M baseline, with additional learning-rate warmup and gradient clipping. We select the checkpoint with the lowest validation loss, at epoch 6.

Evaluation on OAT3D dataset. Because the original NITO-3D work uses a narrower boundary condition sampling distribution, we additionally evaluate these NITO models on the 71-problem OAT3D subset with exactly one loaded node and a prescribed volume fraction $\in [0.15, 0.30]$ using this paper's evaluation protocol.

Table 12: Single-generation performance without candidate selection or refinement on the 71-problem subset with exactly one loaded node and prescribed volume fraction in the nominal shared range of $[0.15, 0.30]$. Evaluation process and metric aggregation follow Section 4.2.

Method	CE mean (%) ↓	CE median (%) ↓	Mean VFE (%)	Failure (%) ↓
Shared loading and volume conditions: 71 problems				
NITO (NITO-trained)	25.64	24.26	11.85	94.37
NITO (OAT-trained, 23M)	26.93	18.19	7.11	39.44
NITO (OAT-trained, 811M)	25.82	19.48	**-0.37**	39.44
OAT3D	**8.88**	**3.17**	-4.41	**5.63**

Findings. Our reproduction achieves performance comparable to the published results on the original dataset (Table 11). On the full OAT3D evaluation set, retraining reduces overall failure from 99.50% to 88.65%, but widening the OAT3D-trained model from approximately 23M to 811M parameters yields only a small marginal reduction to 87.00%.

The shared-condition comparison in Table 12 reinforces this pattern. On these 71 problems, retraining with the OAT3D dataset reduces overall failure on the OAT3D dataset from 94.37% to 39.44%, but widening the model leaves this rate unchanged. OAT3D achieves an overall failure rate of 5.63%, with mean and median CE of 8.88% and 3.17%, respectively. Within the overlapping load-count and volume-fraction conditions, OAT still retains a performance advantage.

Together, these results suggest that width scaling alone, despite a more than 35-fold increase in parameters, does not close the performance gap.

H FURTHER VISUALIZATION OF RESULTS

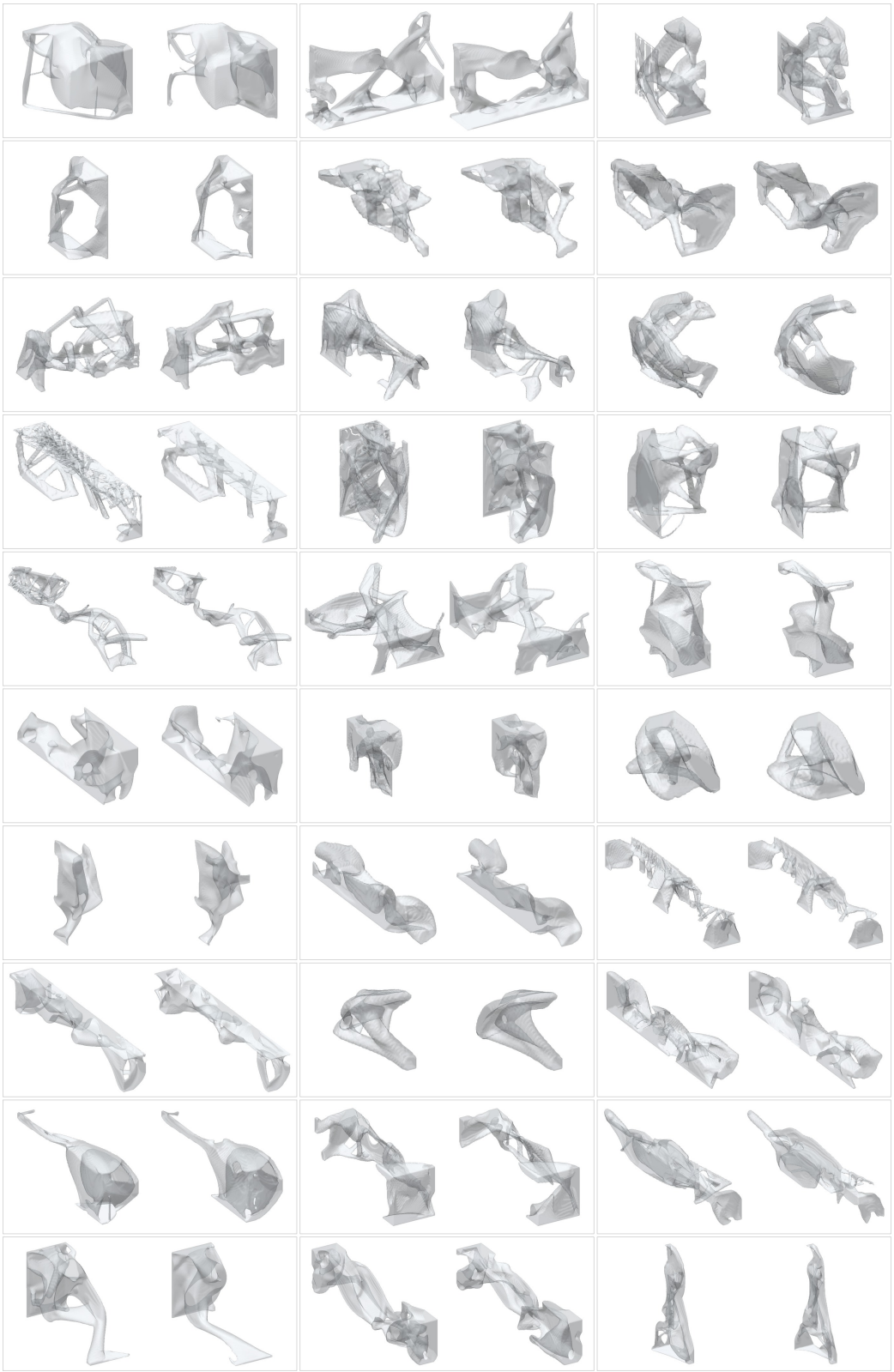


Figure 5: Selection of optimized topologies from the topology optimization (left) and OAT3D (right)



Figure 6: (continued) Selection of optimized topologies from the topology optimization (left) and OAT3D (right)

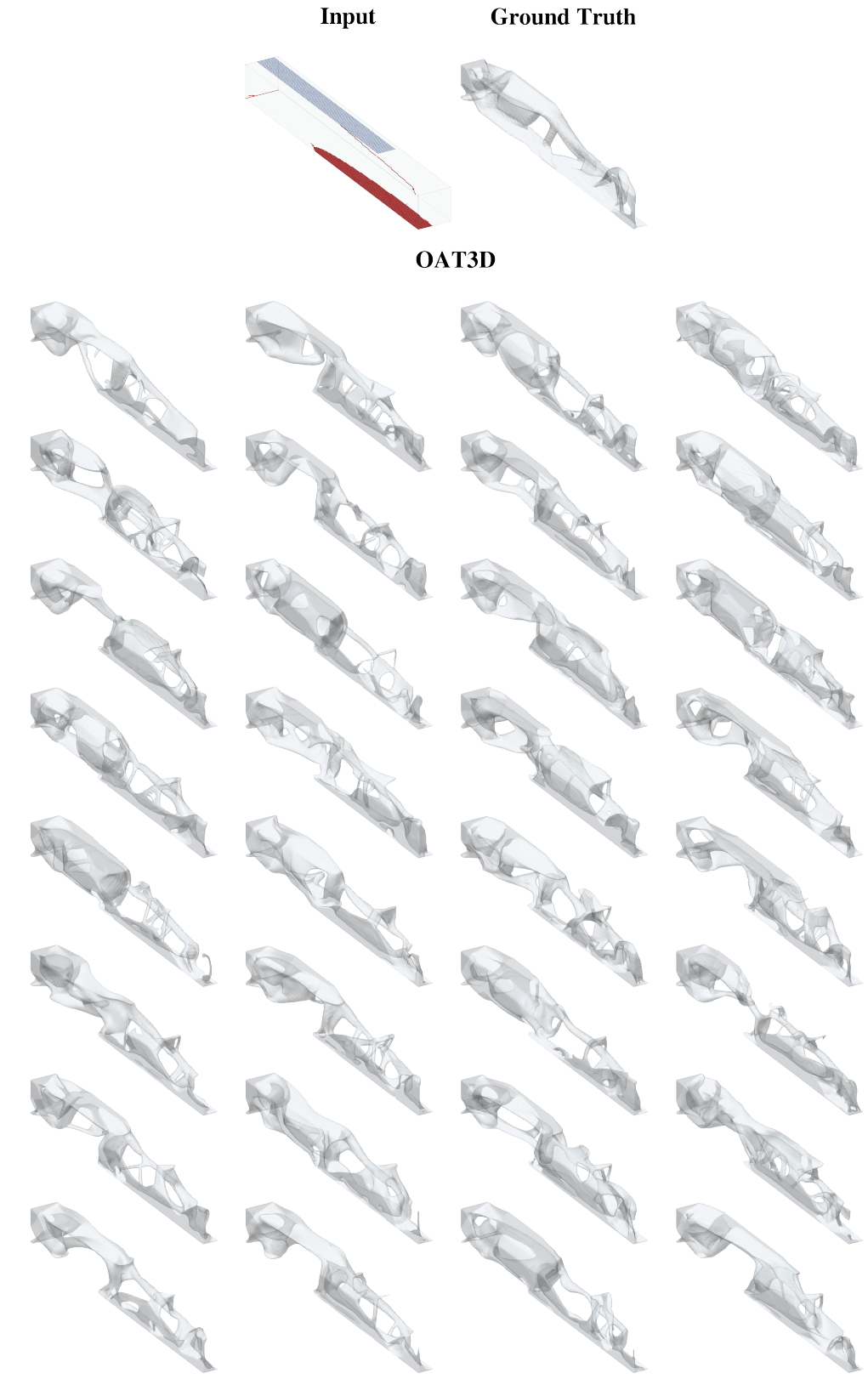


Figure 7: Visualizing all 32 candidates generated by OAT3D given a selected problem. The input boundary condition and the ground truth topology are shown at the top.

I VOLUME FRACTION STRATIFICATION STUDY

2D Topology Optimization. We examine how topology-prediction performance varies with the prescribed volume fraction. All results use single generation per problem, without candidate selection or refinement. We first evaluate the 2D OAT model of Nobari et al. (2026) on 7,500 problems balanced across five volume-fraction bins (Table 13).

3D Topology Optimization. We repeat this analysis on the 2,000-problem OAT3D evaluation set, stratifying it into three volume-fraction bins and comparing OAT3D with the three NITO-3D baselines (Table 14).

Findings. Lower prescribed volume fractions are associated with higher failure rates in both studies. From the highest to the lowest volume-fraction bin, OAT's overall failure rate increases from 32.40% to 74.00%, while OAT3D's increases from 24.80% to 72.12%. Their median CE among passing predictions also increases, from 2.98% to 10.06% and from 19.87% to 35.33%, respectively. OAT3D achieves the lowest failure rate in every 3D bin, but low-volume prediction remains challenging: both OAT3D-trained NITO-3D baselines exceed 96% failure in the lowest bin.

Table 13: Single-generation 2D OAT performance on 7,500 distinct problems, with 1,500 per prescribed-volume-fraction bin. The set combines official-test problems with additional problems generated using the official sampler; the second column gives the number of newly generated problems. Overall failure follows Section 4.2: numerical failure, structural failure ($C > 2C^{\mathrm{ref}}$), or volume excess ($\widehat{v} - v^\star > 0.10$). Overlapping failures are counted once and failure rates use all problems in each bin. CE statistics exclude failed predictions, and mean VFE is signed.

Prescribed VF	Newly generated	CE mean (%) ↓	CE median (%) ↓	Mean VFE (%)	Overall failure (%) ↓
$[0.20, 0.30)$	1,390	16.53	10.06	-0.20	74.00
$[0.30, 0.40)$	510	13.01	6.86	-0.14	53.07
$[0.40, 0.50)$	318	9.55	4.67	-0.20	46.73
$[0.50, 0.60)$	201	7.11	3.79	-0.19	40.20
$[0.60, 0.70]$	81	7.14	2.98	-0.06	32.40

Table 14: Single-generation performance on the 2,000-problem OAT3D evaluation set, stratified by prescribed volume fraction. All models are evaluated on the same problems within each bin. The NITO-3D baselines are trained on either the original NITO-3D dataset or OAT3D, with the latter evaluated at 23M and 811M parameters. Failure criteria and metric aggregation follow Table 13; failure rates use all problems in each bin. Dashes indicate that no predictions pass the failure criteria, leaving CE undefined.

Method	CE mean (%) ↓	CE median (%) ↓	Mean VFE (%)	Overall failure (%) ↓
(a) Prescribed VF $[0.05, 0.10)$*: 312 problems*				
NITO (NITO-trained)	-	-	-	100.00
NITO (OAT-trained, 23M)	41.84	29.67	5.20	97.44
NITO (OAT-trained, 811M)	56.64	65.13	−4.51	96.47
OAT3D	38.88	35.33	−4.29	72.12
(b) Prescribed VF $[0.10, 0.20)$*: 793 problems*				
NITO (NITO-trained)	56.23	56.79	39.22	99.37
NITO (OAT-trained, 23M)	44.92	40.04	5.00	93.82
NITO (OAT-trained, 811M)	45.62	33.42	3.92	91.55
OAT3D	37.60	31.76	−4.71	50.57
(c) Prescribed VF $[0.20, 0.30]$*: 895 problems*				
NITO (NITO-trained)	30.75	14.57	15.62	99.44
NITO (OAT-trained, 23M)	41.96	41.15	3.33	81.01
NITO (OAT-trained, 811M)	43.48	42.71	0.08	79.66
OAT3D	27.80	19.87	−4.19	24.80

J Implementation of OAT3D Applications

J.1 Joint-count control

Joint definition. We use branch junctions in the structural skeleton as a measure of structural complexity. Reference densities are thresholded at 0.3 and skeletonized with the C++ VoxelCore implementation. Candidate joints are curve-skeleton vertices of degree at least three; medial-sheet edges are excluded. Nearby candidates are merged using the interior-distance radius r_i at each candidate. For candidates i and j, the graph distance must satisfy

$$d_G(i,j) \le \min(8,\, 2\min(r_i, r_j))\,, \tag{21}$$

with distances measured in native voxel units. Every pair in a merged group must satisfy this bound. A connecting path must have radius at least $0.5\min(r_i, r_j)$, sampled at intervals no larger than one voxel, and cannot pass through an unrelated candidate. Each merged group is counted as one joint. Generated structures are thresholded at 0.3 and restricted to their largest six-connected solid component before skeletonization and counting.

Joint-count regressors. Following TopoCtrl (Chen et al., 2026b), we train separate 3D CNNs to predict joint count from the clean dense and sparse latents. The dense input has 8 channels on a 16^3 grid. Sparse 16-channel features are scattered onto a 64^3 grid with an additional active-cell mask. Each network uses channel widths 16, 32, 64, and 128, three residual downsampling blocks with GroupNorm and SiLU, and global mean and max pooling followed by a scalar MLP. We predict the standardized log count

$$q(J) = \frac{\log(1+J) - \mu_J}{s_J}, \qquad \mu_J = 2.511887, \quad s_J = 1.775882, \tag{22}$$

where μ_J and s_J are computed from the training labels. Training minimizes squared prediction error on 496,858 labeled training structures, using one of eight reflected views per structure per epoch. We use AdamW with learning rate 3×10^{-4}, weight decay 10^{-4}, batch size 256, a 200-step warmup, and gradient-norm clipping at 1.0. We use checkpoints saved after 30 and 40 epochs for the dense and sparse regressors, respectively.

Guided generation. For each stage, let z_{ref} be the normalized latent of the original structure and f the trained regressor. At noise level σ, the estimated clean latent and the two guidance losses are

$$\widehat{z}_0 = z_\sigma - \sigma v_{\mathrm{CFG}}(z_\sigma, \sigma), \tag{23}$$

$$\mathcal{L}_J = \big(f(\widehat{z}_0) - q(J^\star)\big)^2, \tag{24}$$

$$\mathcal{L}_{\mathrm{ref}} = \mathrm{MSE}(\widehat{z}_0, z_{\mathrm{ref}}). \tag{25}$$

Before each Euler step, we apply

$$z'_\sigma = z_\sigma - \nabla_{z_\sigma}(100\mathcal{L}_J + 100\mathcal{L}_{\mathrm{ref}})\,. \tag{26}$$

Gradients pass through the frozen generator and regressor. We recompute the velocity at z'_σ before taking the Euler step. Reference MSE averages over all dense latent features; in the sparse stage, it uses only coordinates shared with the reference and is zero if none match. Sparse guidance updates feature values while keeping the active dense occupancy blocks fixed.

Both stages start from full Gaussian noise and retain the original problem conditioning. We use 20 Euler steps with scheduler shift 6, dense CFG 5, and sparse CFG 2.5, without clipping or normalizing the guidance gradients. For each problem, we request $J^\star \in \{5, 10, 20, 50, 100\}$ and draw 32 candidates per target, using the same candidate seeds across targets. The joint-control examples are selected from the first 100 eligible held-out problems.

J.2 LATTICE REPLACEMENT

Edited structure. We adapt the lattice replacement procedure of TopoEdit (Chen et al., 2026a) to OAT3D. Starting from the ground truth density field thresholded at 0.3, we compute the convex hull of the occupied voxel centers and intersect it with a binary body-centered cubic or simple cubic lattice. The cell pitch is 16 native voxels and the member diameter is 4. Body-centered cubic members connect each cell center to its corners; simple cubic members follow the cell edges. Cell corners are aligned to multiples of 16 in native coordinates, and occupancy is evaluated at voxel centers. This construction fills the convex hull without retaining an outer shell. We encode the edited density field with the dense and sparse VAEs. Loads, supports, domain dimensions, and BNG conditioning remain unchanged; the volume-fraction input becomes the mean density of the lattice filled structure.

Guided editing. Let z_{edit} denote a normalized latent of the edited structure. We initialize

$$z_{\sigma_s} = (1 - \sigma_s) z_{\mathrm{edit}} + \sigma_s \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \quad \sigma_s \approx 0.427314, \tag{27}$$

and take the final three Euler updates of a 20-step schedule with shift 6 in both stages. Dense and sparse CFG weights are 5 and 2.5. At each update, we compute the clean estimate $\widehat{z}_0$ as above and apply

$$z'_\sigma = z_\sigma - 100 \nabla_{z_\sigma} \mathrm{MSE}(\widehat{z}_0, z_{\mathrm{edit}}). \tag{28}$$

The gradient passes through the generator's clean-latent output. We recompute the velocity after this correction and then take the Euler step. Generator weights remain fixed, and guidance gradients are neither clipped nor normalized.

Decoded dense occupancy above 0.3, cropped to the original domain, determines the sparse active blocks. Edited sparse features are aligned to these blocks by coordinate, with zero reference values at coordinates absent from the edited target. Sparse guidance uses only shared coordinates, and the active dense blocks remain fixed during sparse generation.

We use 16 candidate seeds per lattice type and problem. The examples are drawn from the first 500 eligible held-out problems after the first 2,000 entries, with the same domain-dimension criterion as joint control. Both application figures present visually selected examples. Finite element evaluation was not performed for these application experiments.

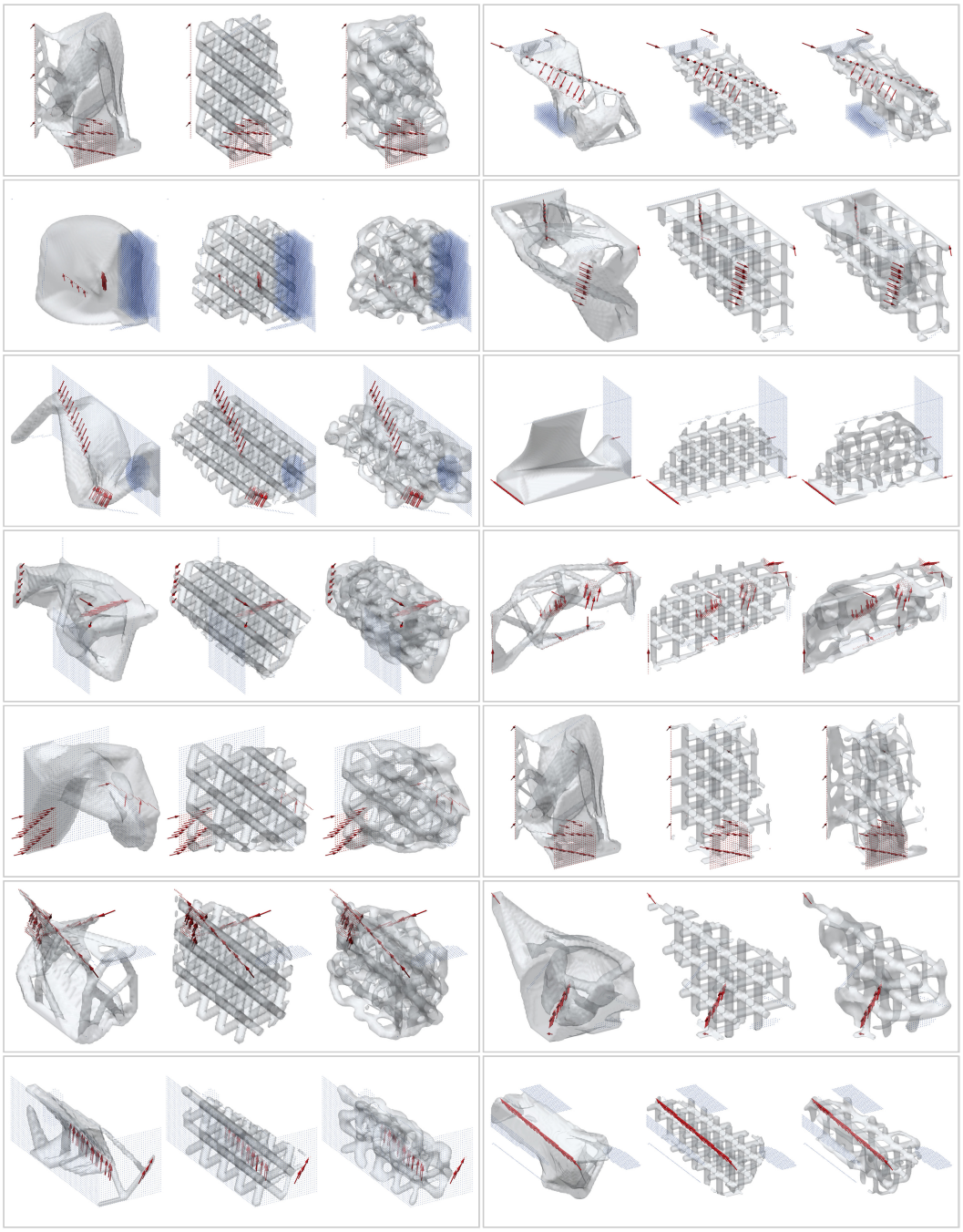


Figure 8: OAT3D on lattice redesign. Given an existing topology (left), we can use vanilla lattice fill (middle) to replace the structure in the convex hull. However, vanilla replacement can miss boundary conditions. OAT3D (right) can further improve the replaced structure through partial noising to recover the load coverage.

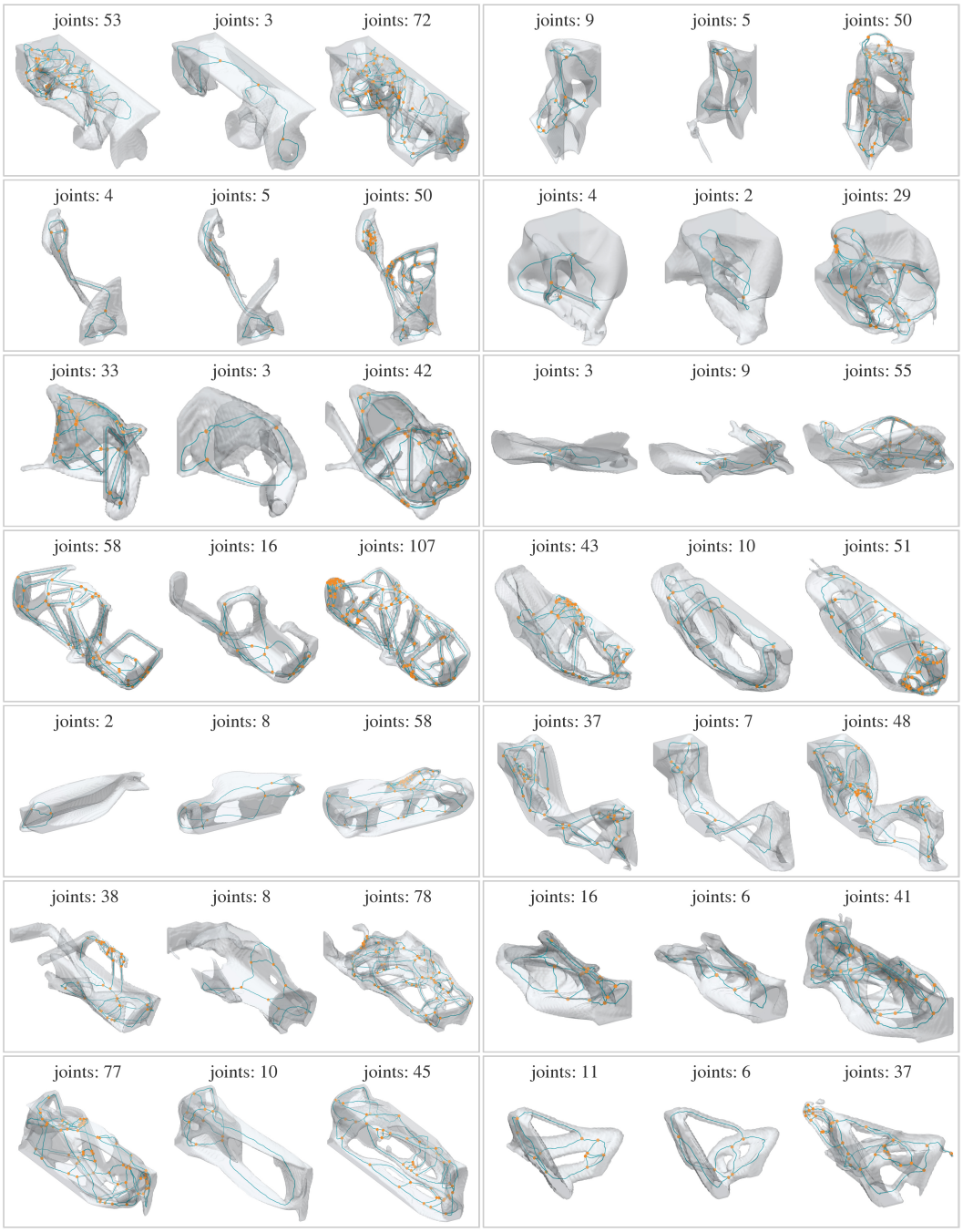


Figure 9: Joint number control with OAT3D. Joints are highlighted in orange while the skeleton is in blue. We tally up the joint number and label it on top of every topology. Left is the original topology, middle is OAT3D controlling the joint number to a lower range, while the right is OAT3D increasing the joint number.