PhysGen: Physically Grounded 3D Shape Generation for Industrial Design

Supplementary Material

This supplemental material includes the following sections:

(A) Implementation details.
(B) Additional experiments.
(C) Network architectures.
(D) Dataset details.
(E) Evaluation metrics.
(F) CFD simulation setup in OpenFOAM [5].
(G) Generalization to structural optimization.

A. Implementation Details

A.1. SP-VAE

For each 3D mesh used for training, we extract $32,768$ uniform surface points $\mathbf{P}_u$ and $32,768$ salient edge points $\mathbf{P}_s$ using the Sharp Edge Sampling (SES) strategy [2]. As shown in Fig. A, queries $\mathbf{Q}$ for cross-attention are constructed by applying Farthest Point Sampling (FPS) [9] to $\mathbf{P}_u$ and $\mathbf{P}_s$, each downsampled to 1024 points and concatenated into a 2048-point set. We supervise the shape encoder-decoder using both uniformly sampled coarse points in the bounding box and sharp points perturbed around the ground-truth surface, with loss weights $\lambda_{\text{SDF}} = 1$ and $\lambda_{\text{KL}} = 0.001$. The shape encoder-decoder is trained for 1000 epochs on $4\times$H100 GPUs (about 2 days). The pressure decoder is trained using pressure values sampled near the surface for 1500 epochs on $4\times$H100 GPUs (about 21 hours). The drag decoder, which predicts a single global drag coefficient, is trained for 1500 epochs and completes within one day on a single H100 GPU. After individual training, we perform joint fine-tuning for 500 epochs on $4\times$H100 GPUs (about 15 hours) using a combined loss with $\lambda_{\text{shape}} = 10$, $\lambda_{\text{physics}} = 0.1$, and $\lambda_{\text{drag}} = 10$. All experiments use 5819 training samples and 1147 test samples from DrivAerNet++ [3].

A.2. Physics-Guided Shape Generation

For flow-based generation, we adopt the rectified flow formulation [7], using 100 sampling steps at inference. In physics-guided shape generation, we incorporate a physics-based regularization term with weight $\lambda_d = 0.03$ for drag guidance during velocity-based updates, while directional weights $\lambda_x = 0.2$, $\lambda_y = 0.1$, $\lambda_z = 0.1$ are applied during the physical refinement phase. For alternating generation, we perform $K = 20$ alternating iterations. In each iteration k, we first apply 20 steps of physical refinement to obtain the refined latent $\hat{\mathbf{z}}_1^k$, then re-noise it back to timestep $t_{n_s} = 0.75$ to produce $\mathbf{z}_{t_{n_s}}^{k+1}$, which initializes the next velocity-based update phase. The full set of iterations takes roughly 210 seconds. Overall, the procedure iteratively alternates between velocity-based updates and physics-based refinement, with each stage performing only a small number of steps (25 steps for velocity updates and 20 steps for physical refinement), gradually converging toward shapes that satisfy both geometric plausibility and physical efficiency.

Table A. Shape generation toward minimizing the drag coefficient. Image-unconditional generation (Unc.) minimizes drag without image, while conditional generation (Cond.) minimizes drag with image conditioning. Average drag coefficients simulated by OpenFOAM indicate aerodynamic performance. (SN: ShapeNet, DAN+: DrivAerNet++.)

Shape	**Average Drag Coefficient**		
	SN (Unc.)	**DAN+** (Unc.)	**DAN+** (Cond.)
w/o minimizing	0.393	0.324	0.334
w/ minimizing	0.304	0.274	0.312
Improvement	**22.70%** ↑	**15.47%** ↑	**6.53%** ↑

B. Additional Experiments

Shape Generation toward Minimizing Drag. Beyond leveraging a known target drag coefficient to improve reconstruction accuracy, our framework can also enhance aerodynamic performance by minimizing the drag coefficient. Table A reports average results over 20 samples per dataset, each simulated using OpenFOAM [5] (see Sec. F for simulation details), covering both the in-distribution DrivAerNet++ [3] dataset and the out-of-distribution ShapeNet [1] car set. Despite never observing ShapeNet geometries during training, our method achieves a substantial 22.7% drag reduction, demonstrating the generalization and the ability to maintain shape plausibility while improving aerodynamic performance. On DrivAerNet++, unconditional physics-guided generation (minimizing drag without image conditioning) reduces drag by 15.47%, whereas the conditional setting (minimizing drag with image conditioning) achieves a 6.53% reduction, as it balances aesthetic alignment with physical efficiency. These findings confirm that alternating prior- and physics-guided generation generalizes robustly to unseen geometries and improves aerodynamic performance while maintaining visual alignment when a conditional image is provided.

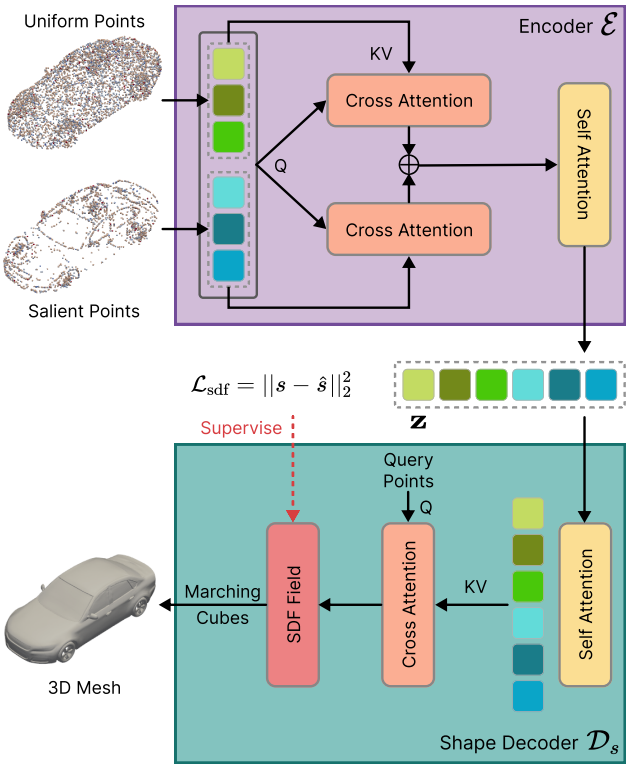

Figure A. Overview of the SP-VAE shape encoder-decoder. The encoder fuses uniform and salient surface points via bidirectional cross-attention and self-attention to produce a latent code. The decoder predicts an SDF field from query points using cross-attention and reconstructs the mesh via marching cubes.

C. Network Architectures

C.1. Shape Encoder and Decoder

We build our shape encoder-decoder architecture upon Dora [2], while extending it to support SDF prediction [6], enabling finer geometric reconstruction than the original occupancy-based representation. As illustrated in Fig. A, the mesh is first extracted into two complementary point sets: (1) uniformly sampled surface points $\mathbf{P}_u$, which capture global coverage of the geometry, and (2) salient edge points $\mathbf{P}_s$, which preserve high-curvature and structurally important regions. These two sets provide separate geometric cues to the encoder. The encoder fuses them via bidirectional cross-attention, letting salient regions inform uniform samples and vice versa. The fused features then pass through self-attention layers to produce the latent code $\mathbf{z}$, capturing both coarse structure and fine details. On the decoding side, we deviate from Dora's occupancy-based formulation and instead predict an SDF field to better preserve high-frequency geometry. The latent code $\mathbf{z}$ is first enriched through several self-attention layers, and a set of 3D query points $\mathbf{x} \in \mathbb{R}^3$ is fed into a linear projection to form the attention queries. Through cross-attention between $\mathbf{x}$ and the latent features, the decoder estimates the corresponding signed distance value $s = \mathcal{D}_s(\mathbf{x}, \mathbf{z})$, effectively conditioning the local geometry on the global shape embedding. The predicted SDF field is then supervised with ground-truth distances sampled around the mesh, and the final mesh is extracted using the marching cubes algorithm [8], yielding a high-quality reconstruction faithful to both global shape and local geometric details.

C.2. Diffusion Transformer Network

We employ a Diffusion Transformer (DiT) [11] within our flow-matching framework to parameterize the velocity field that transports noisy latent codes toward clean representations. As shown in Fig. B, optional conditioning is introduced at the beginning of the network, where $\mathbf{c} = \begin{cases} \mathbf{I}, & \text{if conditional on image,} \\ \varnothing, & \text{if unconditional.} \end{cases}$ For image-conditioned generation, the input image is encoded by DINOv2 [10], and the resulting feature tokens are embedded into a separate conditioning sequence that is injected into every DiT block via cross-attention. The noised latent $\mathbf{z}_{t_n}$ is mapped through a linear projection, while the timestep t_n is encoded using a sinusoidal timestep embedder followed by an MLP. These two embeddings are concatenated to form the token sequence. Each DiT block adopts a pre-norm structure with residual connections and consists of

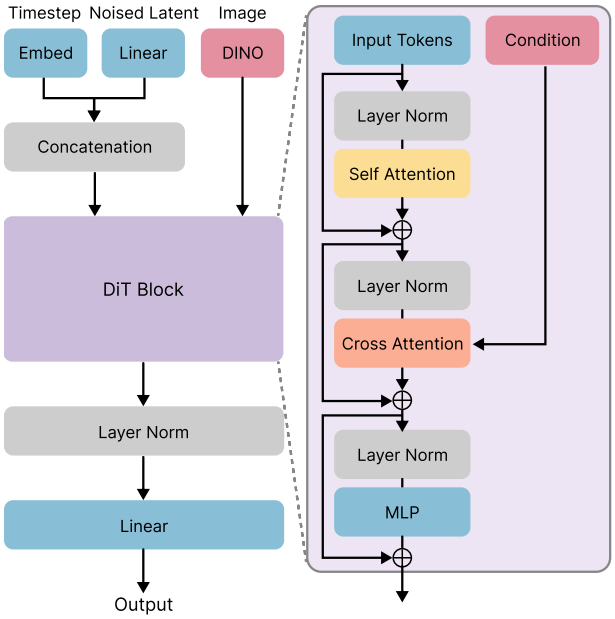

Figure B. Diffusion Transformer (DiT) architecture. Noised latent and timestep embeddings form the input token sequence, while optional DINO-based conditioning is injected via cross-attention in each block. Each DiT block applies self-attention, cross-attention, and an MLP to produce the final velocity prediction.

self-attention over latent tokens, cross-attention with the conditioning tokens $\mathbf{c}$, and a feed-forward network. After all blocks, the final tokens are normalized and projected to produce the velocity field $\hat{\mathbf{u}}(\mathbf{z}_{t_n}, t_n, \mathbf{c})$ required by the flow-matching solver.

D. Dataset details

DrivAerNet++ [3] is a large-scale aerodynamic design dataset comprising $8,000$ high-quality vehicle geometries, each accompanied by high-fidelity CFD simulations, including aerodynamic quantities such as drag coefficients, and both surface pressure and volumetric flow fields. It spans a wide range of automotive body styles, including fastback, notchback, and estateback, and features variations in underbody structure and wheel configurations. Our SP-VAE and flow-based generator are trained on this dataset.

ShapeNet [1] car split is used to evaluate the generalization ability of our method, as it contains vehicle geometries that are not present in the training set. All shapes are uniformly rescaled to match the scale of DrivAerNet++. Although these meshes are physically imperfect yet geometrically reasonable, we use them as initial shapes for our *Physics-Guided Shape Generation* pipeline. By optimizing from these initializations, our method refines the designs into physically efficient and aesthetically pleasing 3D shapes.

E. Evaluation Metrics

For shape generation quality, we evaluate geometric fidelity using F-score, Chamfer Distance, Accuracy, and IoU.

E.1. Shape Generation

F-score. F-score ($\tau = 0.01$) measures consistency between predicted mesh vertices M and ground-truth vertices G, with threshold $\tau = 0.01$:

$$\text{F-score}(\tau) = 2 \cdot \frac{\text{Precision}(\tau) \cdot \text{Recall}(\tau)}{\text{Precision}(\tau) + \text{Recall}(\tau)}, \tag{A}$$

where

$$\begin{aligned} \text{Precision}(\tau) &= \frac{|\{\, m \in M \mid d(m, G) < \tau \,\}|}{|P|}, \\ \text{Recall}(\tau) &= \frac{|\{\, g \in G \mid d(g, M) < \tau \,\}|}{|G|}. \end{aligned} \tag{B}$$

Here, $d(m, G) = \min_{g \in G} \|m - g\|_2$ denotes the nearest-neighbor distance from point m to the ground-truth surface G. Thus, $d(m, G) < \tau$ indicates that the point m lies within a tolerance τ of the target surface.

Chamfer Distance (CD). CD measures the geometric discrepancy between the predicted point set M and the ground-truth point set G. We use the bidirectional form, defined as:

$$\begin{aligned} \text{CD}(M, G) = &\frac{1}{|M|} \sum_{m \in M} \min_{g \in G} \|m - g\|_2^2 \\ &+ \frac{1}{|G|} \sum_{g \in G} \min_{m \in M} \|g - m\|_2^2. \end{aligned} \tag{C}$$

Accuracy (Coarse, Sharp, Overall). Classification accuracy evaluates how well the predicted SDF-based inside/outside labels match the ground truth at sampled query points. For a point $\mathbf{x}$, the predicted label is $y = \mathbf{1}[\, s(\mathbf{x}) \leq 0\,]$, where $s(\mathbf{x})$ is the predicted SDF. The ground-truth label is $\hat{y} = \mathbf{1}[\, \hat{s}(\mathbf{x}) \leq 0\,]$, where $\hat{s}(\mathbf{x})$ is the ground truth SDF. For each sampling split $k \in \{\text{coarse}, \text{sharp}, \text{overall}\}$:

$$\text{Acc}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \mathbf{1}[y_i = \hat{y}_i]. \tag{D}$$

where coarse points are uniformly sampled within the bounding box, sharp points are generated by perturbing points around the ground-truth surface, and overall points are the union of the two.

Intersection over Union (IoU). Using the same binary inside–outside labels, IoU quantifies how well the predicted inside region overlaps with the ground-truth inside region. It is computed as the ratio between the number of points correctly classified as inside (intersection) and the number of points labeled as inside by either the prediction or the ground truth (union). A higher IoU indicates closer agreement between the predicted and true shape boundaries. For a given split $k \in \{\text{coarse}, \text{sharp}, \text{overall}\}$, IoU is computed as:

$$\text{IoU}_k = \frac{\sum_{i=1}^{N_k} \hat{y}_i \, y_i}{\sum_{i=1}^{N_k} \mathbf{1}[\hat{y}_i + y_i > 0] + \varepsilon}. \tag{E}$$

E.2. Physical Estimation

For the estimation task, we adopt standard regression metrics including Mean Squared Error (MSE), Mean Absolute Error (MAE), Maximum Absolute Error (Max AE), Relative L_2 Error (Rel L2), and Relative L_1 Error (Rel L1). For each sampled point i, let p_i denote the predicted pressure and $\hat{p}_i$ the ground-truth pressure, with N being the total number of evaluated points.

Mean Squared Error (MSE). MSE measures the average squared deviation between predicted and true pressures, emphasizing larger errors more strongly:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (p_i - \hat{p}_i)^2. \tag{F}$$

Mean Absolute Error (MAE). MAE computes the average absolute difference between predicted and ground-truth

pressures, providing a more outlier-robust accuracy measure:

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|p_i - \hat{p}_i|. \tag{G}$$

Maximum Absolute Error (Max AE). Max AE Quantifies the worst-case prediction error by identifying the largest absolute pressure deviation across all points:

$$\text{Max AE} = \max_{1\le i\le N}|p_i - \hat{p}_i|. \tag{H}$$

Relative $\mathbf{L_2}$ Error (Rel L2). Rel L2 evaluates the global Euclidean discrepancy normalized by the magnitude of the ground-truth pressure field:

$$\text{Rel L2} = \frac{\|p - \hat{p}\|_2}{\|\hat{p}\|_2}, \tag{I}$$

where $\|\cdot\|_2$ denotes the ℓ_2 norm over all points.

Relative $\mathbf{L_1}$ Error (Rel L1). Rel L1 measures the normalized sum of absolute pressure errors relative to the total absolute ground-truth pressure:

$$\text{Rel L1} = \frac{\|p - \hat{p}\|_1}{\|\hat{p}\|_1}, \tag{J}$$

where $\|\cdot\|_1$ is the ℓ_1 norm, equal to the sum of absolute values over all points.

F. CFD Simulation Setup in OpenFOAM

To evaluate the aerodynamic performance of our generated vehicle geometries, as shown in Table A, we perform high-fidelity CFD simulations using OpenFOAM [5] to compute the drag coefficient, surface pressure distributions, and airflow velocity fields. A uniform inlet freestream velocity of 30 m/s, aligned with the vehicle's longitudinal axis and directed toward the frontal surface, is prescribed to represent standard automotive aerodynamic operating conditions. We employ the steady-state simpleFoam [5] solver together with the k-ω SST turbulence model. Each simulation is run on 32 CPU cores for about 8 hours and proceeds through 2500 iterations to ensure convergence. The final aerodynamic drag coefficient is obtained by averaging the flow fields across the last 500 iterations.

G. Generalization to Structural Optimization

We further evaluate our physics-guided shape generation framework on *structural optimization* by following the setting of PhysiOpt [12]. Given external loads $\mathbf{f}$ and user-specified boundary conditions, we map the latent code $\mathbf{z}$ to an SDF representation, convert it into density-weighted finite elements, and then solve the linear static equilibrium equation:

$$\mathbf{K}(\mathbf{z})\,\mathbf{u} = \mathbf{f}, \tag{K}$$

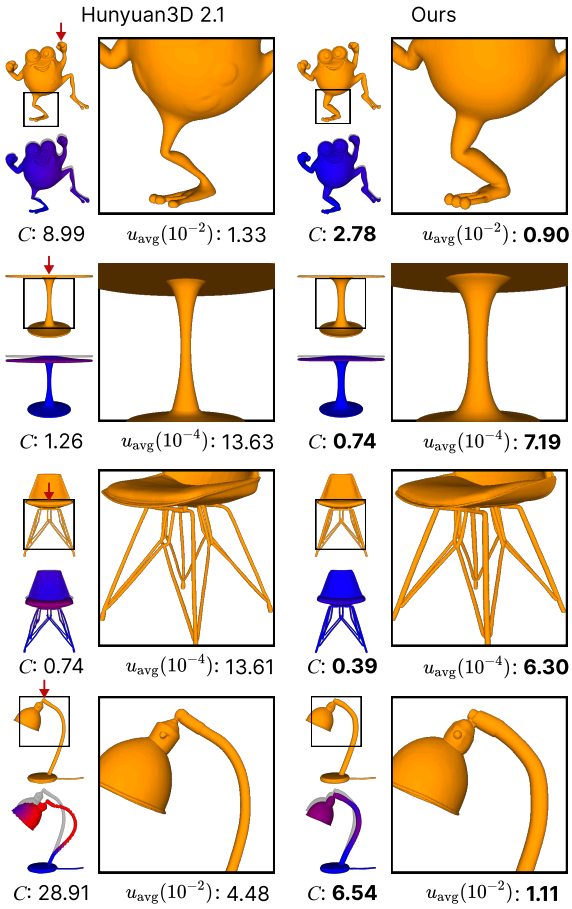

Figure C. Additional results on structural optimization.

where $\mathbf{K}$ is the stiffness matrix and $\mathbf{u}$ denotes the displacement field. As in PhysiOpt, the optimization objective is to reduce the compliance:

$$C = \mathbf{f}^\top \mathbf{u}, \tag{L}$$

which measures the structural deformation, or equivalently the strain energy, under the applied load. Lower compliance indicates a stiffer structure. We also report the average displacement:

$$u_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{u}_i\|_2, \tag{M}$$

where $\mathbf{u}_i$ is the displacement of the i-th FEM node. This formulation follows PhysiOpt's differentiable FEM pipeline, which optimizes shapes directly in latent space under prescribed loads and boundary conditions.

In this task, both PhysiOpt and our method are optimized for 180 steps. Specifically, we perform one velocity-based update after every 5 steps of physical refinement, so that shape plausibility and structural performance are improved jointly throughout the optimization. Fig. C presents additional structural optimization results of our method, where

Hunyuan3D 2.1 [4] generates initial 3D shapes from images without physical awareness. It can be seen that our method improves physical performance while preserving shape quality, yielding structurally stronger and visually appealing shapes.

References

[1] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 3

[2] Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. In *Conference on Computer Vision and Pattern Recognition*, 2025. 1, 2

[3] M. Elrefaie, F. Morar, A. Dai, and F. Ahmed. DrivAerNet++: A Large-Scale Multimodal Car Dataset with Computational Fluid Dynamics Simulations and Deep Learning Benchmarks. In *Advances in Neural Information Processing Systems*, 2024. 1, 3

[4] Team Hunyuan3D, Shuhui Yang, Mingxin Yang, Yifei Feng, Xin Huang, Sheng Zhang, Zebin He, Di Luo, Haolin Liu, Yunfei Zhao, et al. Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material. *arXiv preprint arXiv:2506.15442*, 2025. 5

[5] Hrvoje Jasak, Aleksandar Jemcov, and Zeljko Tukovic. OpenFOAM: A C++ Library for Complex Physics Simulations. In *International workshop on coupled methods in numerical dynamics*, 2007. 1, 4

[6] Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 2

[7] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023. 1

[8] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH Computer Graphics*, 21(4):163–169, 1987. 2

[9] Carsten Moenning and Neil A Dodgson. Fast marching farthest point sampling. Technical report, University of Cambridge, Computer Laboratory, 2003. 1

[10] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 2

[11] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *International Conference on Computer Vision*, 2023. 2

[12] Xiao Zhan, Clément Jambon, Evan Thompson, Kenney Ng, and Mina Konaković Luković. Physiopt: Physics-driven shape optimization for 3d generative models. In *SIGGRAPH Asia*, 2025. 4