GEOMETRY CONDITIONING IN NEURAL OPERATORS THROUGH REPRESENTATION ALIGNMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural operators provide efficient surrogates for physical simulation by learning mappings from problem specifications to solution fields. However, effectively incorporating geometric information into these models often relies on specialized architectural mechanisms, tightly coupling geometry handling with the underlying operator design. In this work, we explore a different perspective: geometry can guide operator learning independently of the operator architecture. We introduce GEometry-aware Alignment of Representations (GEAR), a lightweight regularization framework that treats geometry as a relational training prior. Rather than matching representations to predefined geometric embeddings, GEAR aligns their local relations with those defined by geometry, enabling lightweight geometric guidance directly during training. To balance geometric guidance with the physical prediction objective, GEAR suppresses alignment updates that conflict with the physical prediction objective, allowing geometric structure to guide learning without over-constraining task-specific features. This decouples geometry-aware learning from architectural design and enables the same training strategy to be applied across neural operator backbones. Experiments across multiple operator architectures and simulation datasets involving varying geometries demonstrate consistent improvements in predictive performance.

1 INTRODUCTION

Physical simulation typically relies on numerical solvers to repeatedly solve governing partial differential equations (PDEs), which can be computationally expensive for large-scale or many-query settings (Umetani & Bickel, 2018). Neural simulators provide an efficient alternative by learning direct mappings from problem specifications to physical solution fields (Li et al., 2021; Wang et al., 2023; Wu et al., 2026). Among them, neural operators learn mappings between function spaces, enabling a single model to approximate families of PDE solutions across varying geometries, boundary conditions, and physical parameters. Their efficient inference has made them attractive for simulation-intensive scientific and engineering applications (Altair Engineering Inc., 2026; Ansys Inc., 2026).

However, generalization across complex and irregular geometries remains challenging for neural operators. Real-world physical domains often deviate from the regular grids on which many operator architectures are naturally defined. Recent work has addressed this issue through geometry-aware coordinate mappings (Li et al., 2021; 2025), operators on unstructured meshes (Li et al., 2023d; Bryutkin et al., 2024), and latent geometry conditioning mechanisms (Wu et al., 2024; Li et al., 2023d). More recent approaches further inject geometric context into intermediate representations through cross-attention (Adams et al., 2025; Zhang et al., 2026) or repeated geometry updates across network layers Xia & Aviles-Rivero (2026); Zeng et al. (2026). Despite their differences, these approaches generally rely on explicit geometry-conditioning mechanisms to shape intermediate representations, coupling the use of geometric priors with dedicated architectural designs.

Our approach. In this work, we investigate geometry handling from the perspective of representation learning, rather than architectural conditioning. Recent advances in representation alignment,

Anonymous code is available at
`https://anonymous.4open.science/r/artifact-7f3c2a9e`.

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.

such as REPA, show that intermediate representations can be effectively shaped by informative external representations without modifying the underlying model architecture (Yu et al., 2025; Wang et al., 2025; Singh et al., 2026). Our key premise is that geometric information need not be explicitly injected throughout a neural operator to influence its internal representations. Instead, geometry can serve as a training-time prior that guides how physical representations are organized. To this end, we introduce GEometry-aware Alignment of Representations (GEAR), a lightweight regularization framework that aligns intermediate features with local geometric relations while leaving the underlying operator architecture unchanged. Unlike conventional representation alignment with a predefined target embedding, geometry specifies relational structure among spatial points rather than a unique feature representation. GEAR therefore aligns intermediate features with these geometric relations while suppressing alignment updates that conflict with the prediction objective.

Recent findings on representation alignment suggest that auxiliary alignment can be particularly beneficial early in training, but may become less useful or even restrictive as task-specific representations develop (Wang et al., 2025). Motivated by this observation, we do not assume that geometric alignment should persist throughout optimization. Instead, we treat geometry as a transient training prior and assess whether its updates remain beneficial to the prediction objective across batches, progressively reducing the alignment when this utility declines. This allows geometric guidance to shape representation learning when useful, while leaving later optimization increasingly driven by the physical prediction task. Based on our analysis, we evaluate our scheme across multiple neural operator architectures and geometry-dependent physical simulation tasks. Across these settings, GEAR consistently provides competitive or improved predictive performance while preserving the original inference architecture. Further analyses examine the effect of geometric alignment during training, supporting the use of geometry as a transient representation-learning prior. We highlight the main contributions of this paper below:

- We introduce a simple training-time representation alignment framework for geometry-aware neural operators that aligns intermediate neural operator representations without requiring geometry-specific architectural design.
- We formulate geometry as a relational prior rather than a predefined target embedding, enabling architecture-independent alignment without relying on separately pretrained geometry representations.
- We further regulate geometric guidance and demonstrate improvements across multiple neural operator backbones and simulation datasets.

2 Related Work

2.1 Neural simulators

Neural operators. A wide range of neural architectures have been developed for data-driven physical simulation (Raissi et al., 2020; Pfaff et al., 2021; Wu et al., 2026). Among them, neural operators have emerged as a prominent paradigm that formulates physical simulation as learning mappings between function spaces, enabling direct approximation of solution operators for partial differential equations (PDEs) (Kovachki et al., 2023). Representative approaches include the Fourier Neural Operator (FNO) (Li et al., 2021) and its variants (Li et al., 2023c; Wu et al., 2023), which parameterize integral operators through efficient transformations in the Fourier domain. To extend neural operators beyond regular grids, subsequent works have introduced graph-based formulations and other geometry-aware architectures for irregular domains and unstructured meshes (Li et al., 2023d; Sarkar & Chakraborty, 2025; Pfaff et al., 2021). By representing spatial interactions through flexible neighborhoods or mesh connectivity, these methods broaden neural operator learning to more general physical discretizations.

Attention-based physical simulation. Beyond spectral and graph-based formulations, recent studies (Zeng et al., 2026; Luo et al., 2025; Nabian et al., 2026) have explored attention mechanisms (Vaswani et al., 2017) for neural simulation, leveraging their flexibility in modeling long-range interactions among spatially distributed physical states (Kovachki et al., 2023). However, directly applying standard self-attention to dense physical fields incurs quadratic computational and memory costs with respect to the number of spatial elements. To improve scalability, Galerkin Transformer (Cao, 2021) reformulates attention through a Galerkin-type projection. FactFormer (Li et al.,

2023b) exploits structural properties of physical domains to further reduce computational overhead. GNOT (Hao et al., 2023) introduces a Transformer-based neural operator with heterogeneous attention to handle irregular meshes and multiple physical inputs. Transolver (Wu et al., 2024) introduces latent physical states to mediate interactions among mesh elements, achieving competitive accuracy and scalability across complex simulation and engineering design tasks. Recent studies further improve the efficiency of attention-based neural operators through linearized or low-rank formulations, such as LinearNO (Hu et al., 2026) and low-rank spatial attention (Yang et al., 2026), which reduce the cost of global interaction modeling while preserving long-range dependencies.

2.2 Geometry Integration in Neural Operators

Beyond advances in operator architectures, increasing attention has been devoted to explicitly incorporating geometric information into neural operators (Li et al., 2023c;d; Cheng et al., 2024). Early approaches primarily accommodate geometric variability through domain transformations or geometry-aware mappings, while more recent studies explore using geometry as persistent contextual information throughout deep operator networks (Adams et al., 2025; Zhang et al., 2026). GeoTransolver (Adams et al., 2025) constructs multi-scale geometric context, which is repeatedly queried by latent physical states through cross-attention across network layers. Similarly, Zhang et al. (Zhang et al., 2026) pretrain latent geometry representations from geometry-only data and integrate them into downstream neural operators through cross-attention, allowing physical prediction to directly query pretrained geometric features. Beyond using fixed geometric conditioning, CoEvol-NO (Zeng et al., 2026) jointly evolves coordinate representations and latent physical states across layers, allowing geometric context to be progressively adapted along with the physical representation. Finally, Xia et al. (Xia & Aviles-Rivero, 2026) identify geometry forgetting in deep neural operators and show that reinjecting geometric information can mitigate this effect, although the optimal injection strategy varies across architectures and network depths.

Comparison with existing works. Existing methods typically integrate geometry through architectural conditioning or repeated reinjection. In contrast, GEAR uses geometry as a training-time prior to shape intermediate representations through relational alignment.

3 Methodology

This section introduces GEAR and describes how geometric structure is incorporated into neural operator training through representation alignment. We first present the overall formulation, then detail the geometry-aware alignment strategy and the adaptive mechanism for controlling geometric guidance during optimization.

3.1 Overview

We consider neural operators that predict physical solution fields over geometrically varying domains. Given spatial coordinates $\mathbf{X}$ and physical conditions $\mathbf{c}$, the operator produces the corresponding solution field $\hat{\mathbf{y}} = F_\theta(\mathbf{X}, \mathbf{c})$. The model is trained to match the reference field $\mathbf{y}$ through the task objective

$$\mathcal{L}_{\text{task}} = \mathbb{E}_{(\mathbf{X},\mathbf{c},\mathbf{y})\sim\mathcal{D}} \left[\ell(F_\theta(\mathbf{X}, \mathbf{c}), \mathbf{y})\right], \tag{1}$$

where ℓ denotes the prediction loss for the physical field. During this process, the operator forms intermediate point-wise representations $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_N]^\top \in \mathbb{R}^{N\times d}$, which encode evolving physical states used for prediction. Rather than modifying operators to repeatedly inject geometric information, we use the geometric structure associated with $\mathbf{X}$ to regularize these representations during training.

Unlike representation alignment with a predefined target embedding, geometry does not uniquely specify what each $\mathbf{h}_i$ should encode. Instead, it provides relational structure among spatial points, indicating which points should be represented more similarly than others. We therefore formulate geometric guidance as a relational alignment problem: geometry defines a target relation over a local candidate set, and the hidden representations are encouraged to preserve the same structure.

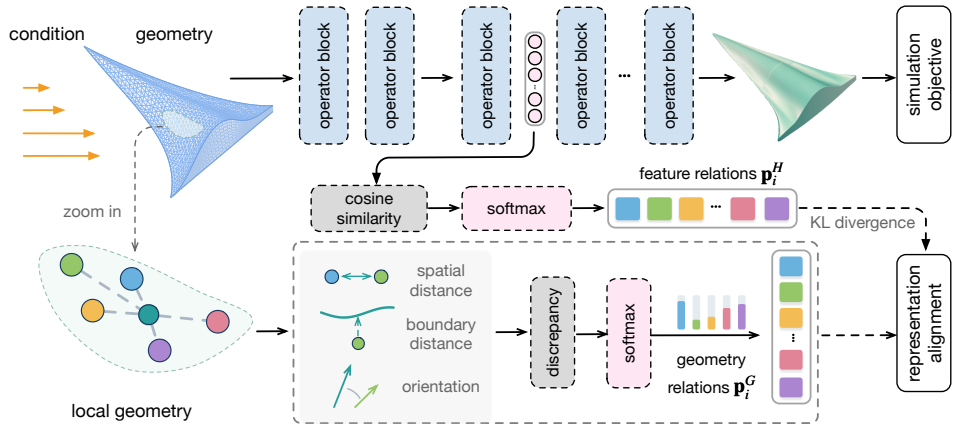


Figure 1: Overview of the geometry-aware alignment in GEAR. Given geometry and physical conditions, the neural operator predicts the simulation field through its original backbone. During training, local geometric relations are constructed from spatial and structural cues, while intermediate features induce corresponding representation relations. The two distributions are aligned by KL divergence, allowing geometric structure to guide representation learning.

3.2 GEOMETRY-AWARE ALIGNMENT

To construct geometric supervision, we first define a local candidate set $\mathcal{C}_i$ for each point i using multi-scale voxel grids Adams et al. (2025) and characterize its relation to nearby candidates using local geometric cues. Beyond Euclidean proximity, we incorporate structural information commonly used in geometry-aware operator modeling, such as distance-to-boundary and orientation-related features (Li et al., 2023d; Kim et al., 2026). Let D_{ij}^G denote the resulting geometric discrepancy between points i and j. Rather than treating these discrepancies as independent pairwise constraints, we normalize them over candidate set to describe relative geometric organization around each point:

$$\mathbf{p}_i^G = \mathrm{softmax}_{j\in\mathcal{C}_i}\left(-D_{ij}^G\right). \tag{2}$$

This distribution serves as the geometric alignment target, assigning larger probability to candidates that are more closely related to point i.

The intermediate representation induces a corresponding relation in feature space. Since geometry does not prescribe a unique feature vector for each point, directly matching geometric descriptors would impose an unnecessarily specific target on the hidden representation. Instead, we ask whether the learned features preserve the same relative organization: candidates with stronger geometric relations should also exhibit greater similarity in the hidden space. We therefore define

$$\mathbf{p}_i^H = \mathrm{softmax}_{j\in\mathcal{C}_i}\left(\frac{\mathrm{sim}(\mathbf{h}_i,\mathbf{h}_j)}{\tau}\right), \tag{3}$$

where $\mathrm{sim}(\cdot,\cdot)$ denotes cosine similarity and τ controls the concentration of the distribution. GEAR then aligns the geometry-induced and representation-induced relations through

$$\mathcal{L}_{\mathrm{align}} = \mathbb{E}_{(\mathbf{X},\mathbf{c})\sim\mathcal{D},\, i\sim\mathcal{V}}\left[D_{\mathrm{KL}}\left(\mathbf{p}_i^G \,\|\, \mathbf{p}_i^H\right)\right], \tag{4}$$

where $\mathcal{V}$ denotes the set of points with valid candidate relations. In this way, geometric supervision constrains the local organization of the representation rather than its exact feature values, leaving the hidden space free to retain information required for physical prediction.

Let $\mathbf{t}_i = \nabla_{\mathbf{h}_i}\mathcal{L}_{\mathrm{task}}$ and $\mathbf{a}_i = \nabla_{\mathbf{h}_i}\mathcal{L}_{\mathrm{align}}$ denote the task and alignment gradients, respectively. Their cosine similarity measures whether geometric alignment agrees with the prediction objective at point i. When $\cos(\mathbf{t}_i,\mathbf{a}_i) < 0$, we remove the component of the alignment gradient (Yu et al., 2020) that opposes the task direction:

$$\tilde{\mathbf{a}}_i = \mathbf{a}_i - \|\mathbf{a}_i\|_2 \cos(\mathbf{t}_i,\mathbf{a}_i)\hat{\mathbf{t}}_i, \tag{5}$$

where $\hat{\mathbf{t}}_i = \mathbf{t}_i/\|\mathbf{t}_i\|_2$. When the two gradients are compatible, $\mathbf{a}_i$ is retained unchanged. This operation removes only the conflicting component while preserving the remaining geometric guidance.

Then, the geometric alignment can be incorporated as a regularizer on top of the original prediction objective. The overall training objective is

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \lambda \mathcal{L}_{\text{align}}, \tag{6}$$

where λ controls the strength of geometric supervision. Unlike standard joint optimization, however, GEAR does not directly propagate the raw gradient of $\mathcal{L}_{\text{align}}$ through the intermediate representation. Instead, it uses the filtered gradient $\tilde{\mathbf{a}}_i$, so that geometric alignment contributes only through directions compatible with physical prediction. The resulting representation-level update can be written as

$$\mathbf{g}_i = \mathbf{t}_i + \lambda \tilde{\mathbf{a}}_i. \tag{7}$$

This retains the simplicity of an auxiliary alignment objective while preventing geometric supervision from overriding task-specific representation learning.

Together, these steps make geometric alignment selective rather than prescriptive. The relational objective encourages the hidden space to respect local geometric structure, while gradient filtering prevents this supervision from overriding the prediction objective.

3.3 Adaptive Geometric Guidance

The alignment above determines how geometry influences the representation, but still assumes that its contribution remains equally useful throughout training. Recent studies suggest that auxiliary representation alignment can become less beneficial as task-specific representations gradually emerge (Wang et al., 2025). Motivated by this observation, we treat geometric alignment as a transient source of supervision and adapt its strength according to its utility to the prediction objective.

Since local conflicts have already been resolved in the representation space, we further examine whether the resulting geometric update remains useful beyond the batch from which it is obtained. Specifically, we compute the filtered geometric update from one training batch B_a and compare it with the task gradient evaluated on another batch B_b. Let $\mathbf{d}_a = \mathbf{J}_a^\top \tilde{\mathbf{a}}_a$ denote the resulting parameter-space geometric update, where $\tilde{\mathbf{a}}_a$ is the filtered alignment gradient and $\mathbf{J}_a$ maps the intermediate representation to model parameters. Similarly, $\mathbf{g}_b = \nabla_\theta \mathcal{L}_{\text{task}}(B_b)$ denotes the parameter-space gradient of the physical prediction objective on B_b. We define the cross-batch utility as

$$u_e = \cos(\mathbf{d}_a, \mathbf{g}_b). \tag{8}$$

A positive u_e indicates that the geometric update derived from B_a is also directionally consistent with task optimization on B_b, whereas a negative value suggests potential interference. In this way, the utility reflects whether geometric guidance remains beneficial beyond its immediate training batch rather than merely fitting the local alignment objective.

Because estimates from individual batch pairs can be noisy, we maintain an exponentially smoothed utility $\bar{u}_e = \beta \bar{u}_{e-1} + (1-\beta) u_e$. This signal controls an internal guidance state $z_e \in [0, 1]$,

$$z_{e+1} = \text{clip}_{[0,1]} \left(z_e + \eta_s \tanh\left(\frac{\bar{u}_e}{T_u} \right) \right), \tag{9}$$

where η_s controls the adaptation rate and T_u determines the sensitivity to utility changes. Positive utility supports retaining geometric supervision, while sustained weak or negative utility drives the guidance state downward. The actual alignment strength s_e is constrained to be non-increasing, preventing geometric supervision from being reintroduced once it has been released. We use s_e to modulate the alignment weight defined in Section 3.2, allowing GEAR to progressively shift from geometry-guided representation learning toward optimization dominated by the original prediction objective.

4 EXPERIMENTS

In this section, we introduce the simulation datasets and neural operator baselines, followed by quantitative and qualitative evaluations. Additional experimental details are provided in the Appendix A.

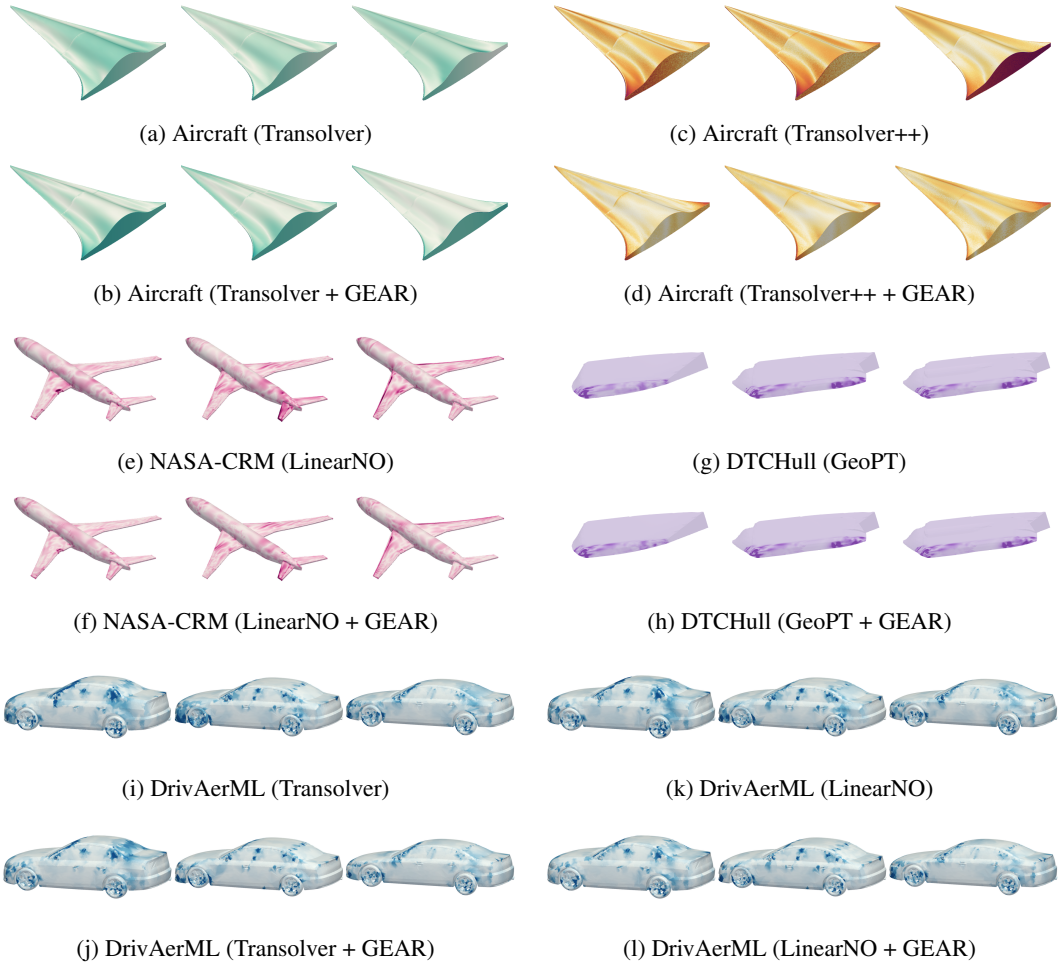
Simulation datasets. We evaluate our method on five 3D simulation datasets covering aerodynamic, hydrodynamic, and structural impact problems. DrivAerML (Ashton et al., 2024) contains high-fidelity automotive CFD simulations over varying vehicle geometries, while NASA-CRM (Bekemeyer et al., 2025) and AirCraft (Luo et al., 2025) cover aircraft aerodynamics under variations in

Table 1: Comparison of relative L_2 errors across five engineering datasets. Lower values are better. Best results are shown in **bold**, and second-best results are underlined.

method	AirCraft	NASA–CRM	Car–Crash	DTCHull		DrivAerML	
				volume	surface	volume	surface
Neural Operators							
F-FNO	0.6058	0.9416	0.4762	0.6335	0.8988	0.6757	0.7136
Geo-FNO	0.6044	0.6037	0.3280	0.1911	0.3605	0.3952	0.4600
LSM	0.6055	0.4693	0.3799	0.2032	0.3762	0.4549	0.4976
GINO	0.1886	0.3580	0.3337	0.2173	0.4014	0.5306	0.5787
LaMO	0.3147	0.0961	0.2087	0.1749	0.3094	0.2328	0.2869
EqGINO	0.1772	0.3524	0.2999	0.2428	0.4226	0.5266	0.5798
Transformer-Based Operators							
Galerkin	0.5945	0.7634	0.5561	0.6429	0.9276	0.6094	0.6345
GNOT	0.2998	0.0922	0.1867	0.1680	0.2929	0.3622	0.4054
LNO	0.5945	0.9961	0.5260	0.6429	0.9276	0.7652	0.7644
ONO	0.6937	0.9972	0.7103	0.6450	0.9275	0.7664	0.7634
GeoTransolver	0.0568	0.0993	0.2196	0.2141	0.3400	0.4330	0.4981
CoEvol-NO	0.1625	0.5394	0.2035	0.1338	0.2539	0.2387	0.2916
Operators with GEAR							
OFormer	0.5867	0.7088	0.3860	0.4786	0.6934	0.5282	0.5644
+GEAR	0.2408	0.5259	0.3760	0.4276	0.6892	0.5263	0.5586
GeoPT	0.1097	0.0867	0.1776	0.1037	0.2279	0.1732	0.2119
+GEAR	0.0465	**0.0807**	0.1756	**0.1028**	**0.2205**	**0.1713**	**0.1926**
Transolver++	0.1020	0.0961	0.1872	0.1361	0.2598	0.2782	0.3256
+GEAR	0.0567	0.0890	0.1843	0.1240	0.2604	0.2618	0.3112
Transolver	0.1432	0.1015	0.1886	0.1251	0.2540	0.2665	0.3202
+GEAR	0.0955	0.0898	**0.1715**	0.1206	0.2418	0.2262	0.2706
LinearNO	0.0698	0.0974	0.1896	0.1204	0.2488	0.2306	0.2811
+GEAR	**0.0454**	0.0864	0.1828	0.1156	0.2467	0.2027	0.2586

geometry and operating conditions. These aerodynamic datasets involve predicting physical quantities such as surface pressure and surrounding flow fields. DTCHull (Bagazinski & Ahmed, 2023) considers ship hydrodynamics over parametrically generated hull geometries and varying yaw angles, with RANS simulations performed using OpenFOAM (Jasak, 2009) to obtain time-averaged surface pressure and flow velocity. Car-Crash (Wu et al., 2026) models structural impact under different collision angles using OpenRadioss (Altair Engineering, 2026), with the maximum Von Mises stress during deformation used as the prediction target. Across these datasets, approximately 100 cases are used for training, with an additional 20–50 cases reserved for evaluation.

Baselines. We compare GEAR with neural operators spanning spectral, latent-space, attention-based, and recent geometry-aware designs. The spectral and latent-space baselines include F-FNO (Tran et al., 2023), Geo-FNO (Li et al., 2023c), LSM (Wu et al., 2023), GINO (Li et al., 2023d), and LaMO (Tiwari et al., 2025), covering factorized Fourier operators, geometry-aware coordinate transformation, latent spectral modeling, graph-based processing of irregular domains, and state-space operator learning. We further include Galerkin Transformer (Cao, 2021), GNOT (Hao et al., 2023), LNO (Wang & Wang, 2024), ONO (Xiao et al., 2024), and CoEvol-NO (Zeng et al., 2026) to cover attention-based and recent neural operator architectures. To examine whether our alignment strategy can be applied across different operator designs, we additionally integrate GEAR with OFormer (Li et al., 2023a), GeoPT (Wu et al., 2026), Transolver++ (Luo et al., 2025), Transolver (Wu et al., 2024), and LinearNO (Hu et al., 2026), and compare each variant with its corresponding original backbone under the same training setting.



(a) Aircraft (Transolver) (c) Aircraft (Transolver++)

(b) Aircraft (Transolver + GEAR) (d) Aircraft (Transolver++ + GEAR)

(e) NASA-CRM (LinearNO) (g) DTCHull (GeoPT)

(f) NASA-CRM (LinearNO + GEAR) (h) DTCHull (GeoPT + GEAR)

(i) DrivAerML (Transolver) (k) DrivAerML (LinearNO)

(j) DrivAerML (Transolver + GEAR) (l) DrivAerML (LinearNO + GEAR)

Figure 2: Qualitative comparison of prediction errors across different datasets and neural operator backbones. For each backbone, the baseline model is shown above its GEAR-enhanced counterpart. The three columns correspond to matched test cases in the same left-to-right order; each baseline–GEAR pair uses identical samples, geometries, conditions, and viewpoints.

4.1 Main Results

Quantitative comparison. Table 1 reports the relative L_2 errors across different simulation datasets and neural operator backbones. We apply GEAR to several typical and recent operators that expose point-wise intermediate representations, covering different architectural designs while satisfying the basic requirement of our alignment framework. Across these backbones, GEAR improves the corresponding model in nearly all paired evaluations. For example, the error of Transolver on AirCraft decreases from 0.1432 to 0.0955, while LinearNO improves from 0.0974 to 0.0864 on NASA-CRM; similar gains are observed for both volume and surface prediction on DrivAerML. These results suggest that the proposed alignment is not tied to a particular operator design and can provide complementary geometric supervision across different point-wise neural operators. Additional quantitative results are provided in Appendix D.2.

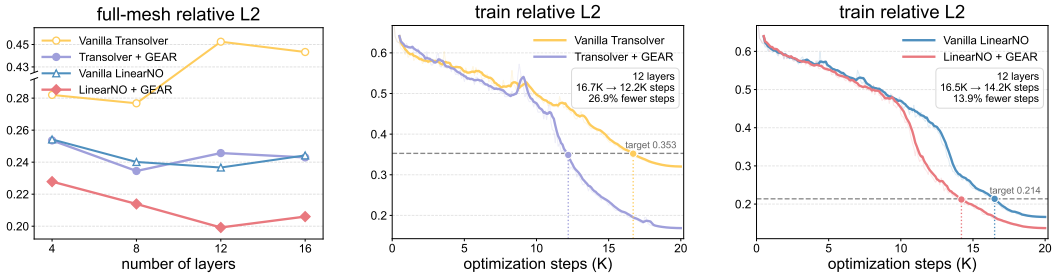
Qualitative comparison. Figure 2 compares prediction errors before and after applying GEAR on several test cases. Across the examples, the enhanced models generally exhibit weaker localized error regions than their original backbones, while the overall spatial distribution of the predictions remains similar. The reduction is noticeable in regions where the baseline produces relatively concentrated errors, suggesting that geometric alignment helps the learned representation better resolve locally difficult structures. These observations are consistent with the quantitative improvements in

Table 1 and provide further evidence that the proposed training-time guidance can improve prediction quality without altering the underlying operator.

4.2 Scaling and Convergence Analysis

Depth scaling. We examine whether the effectiveness of GEAR depends on model depth. Across all tested configurations, geometric alignment consistently improves the corresponding backbone despite the non-monotonic effect of increasing depth itself (Fig. 3a). This indicates that the gain is not specific to a particular depth, but remains complementary across different model capacities.

Optimization convergence. We further ask whether the benefit appears only at the final checkpoint or also during optimization. Under the matched-loss criterion, GEAR requires 26.9% fewer steps for Transolver and 13.9% fewer steps for LinearNO (Figs. 3b–3c). The separation emerging later in training suggests that the benefit accumulates through representation learning, with its magnitude varying across backbones. Since GEAR introduces additional per-step computation, reduced optimization steps do not necessarily translate into lower wall-clock cost; we provide a detailed timing analysis in Appendix D.1.



(a) Depth scaling. (b) Transolver convergence. (c) LinearNO convergence.

Figure 3: Depth scaling and optimization convergence on DrivAerML. (a) Full-mesh relative L_2 error across different network depths. (b–c) Training convergence of 12-layer Transolver and LinearNO under a matched-loss criterion. GEAR reaches the corresponding target errors with 26.9% and 13.9% fewer optimization steps, respectively.

4.3 Ablation Study

In this subsection, we further analyze how geometric information is incorporated into GEAR through a series of ablation studies. In particular, we examine where geometric alignment is applied within the operator and how long geometric guidance should remain active during training.

Guidance schedule. Persistent geometric alignment is not consistently beneficial, degrading DrivAerML while offering little gain on Car–Crash. Restricting guidance to early training avoids this issue, while adaptive withdrawal performs best on both tasks, as shown in Table 2. This supports treating geometry as a transient signal whose useful duration varies across settings.

Alignment position. As shown in Figure 4, alignment is most effective at an intermediate representation: h_0 provides no gain, h_2 reduces the relative L_2 error from 0.1432 to 0.0955 on AirCraft, while deeper h_4 alignment is less effective. This suggests that geometric guidance is most useful after initial physical features emerge but before deeper transformations, consistent with prior findings on intermediate representation guidance and geometry forgetting (Wang et al., 2025; Xia & Aviles-Rivero, 2026).

Geometry-relation alignment. We track how learned representation relations evolve relative to the geometric target during training in Figure 5. Without explicit alignment, the KL divergence increases on NASA–CRM, suggesting that task optimization alone does not preserve geometric relations. In contrast, GEAR progressively reduces the discrepancy by several orders of magnitude, indicating that geometric structure is increasingly reflected in the learned representation.

Geometry interaction strategies. Table 3 compares different ways of interacting geometry with latent physical representations. Transolver updates the representation layer by layer without an explicit

Table 2: Ablation of geometry-guidance schedules, comparing persistent, fixed-duration, and adaptive guidance. Full-mesh relative L_2 is reported; lower is better.

method	schedule	DrivAerML	Car–Crash
Vanilla	None	0.2768	0.1887
GEAR-Persistent	All epochs	0.4033	0.1886
GEAR-Fixed	First 40%	0.2720	0.1849
GEAR-Adaptive	Learned	**0.2718**	**0.1715**

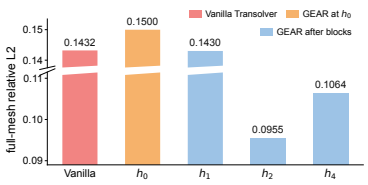


Figure 4: Effect of alignment position on AirCraft using the 8-layer Transolver backbone.

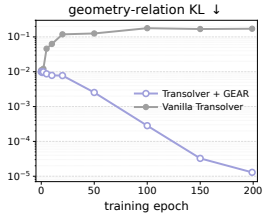


Figure 5: Geometry relation alignment.

Table 3: Comparison of geometry conditioning and alignment strategies on DrivAerML.

Method	Geometry source	Utilization	Rel. $L_2 \downarrow$
Vanilla Transolver	—	—	0.2768
GeoTransolver	Geometry encoder	Cross-attention	0.4447
CoEvol-NO	Native geometry branch	Native coupling	0.2490
Geometry encoder‡			
Transolver + Add	GeoTransolver encoder	Addition	0.2964
Transolver + Concat	GeoTransolver encoder	Concatenation	0.3390
Transolver + FiLM	GeoTransolver encoder	FiLM	0.3264
GEAR, encoder target	GeoTransolver encoder	Relational alignment	0.2399
Explicit geometric relations			
Transolver + Add†	Geometric relation features	Addition	0.3186
Transolver + Concat†	Geometric relation features	Concatenation	0.4352
Transolver + FiLM†	Geometric relation features	FiLM	0.3128
GEAR	Explicit geometry relations	Relational alignment	**0.2345**

‡Uses a frozen geometry encoder from a separately pretrained GeoTransolver (Adams et al., 2025). †Injection explicit geometry relations (Xia & Aviles-Rivero, 2026).

geometry branch; GeoTransolver introduces a geometry encoder and repeatedly queries encoded geometry through cross-attention, while CoEvol-NO (Zeng et al., 2026) jointly optimizes geometric and task representations. Building on these designs, we further control the geometry source and compare direct feature injection with relational alignment. Across both encoder-based geometry and explicit geometric relations, direct injection is less reliable, whereas relational alignment yields more consistent improvements. This suggests that the benefit of geometric priors depends not only on what information is provided, but also on how it is used to shape the learned representation.

4.4 Discussion

Geometric priors can guide intermediate representations without being tied to a specific geometry encoding or injection mechanism, allowing alignment target to adapt across tasks; more informative geometric representations can also provide stronger supervision and potentially yield larger gains through alignment. Together with gradual withdrawal, this supports viewing geometry as a flexible and transient training scaffold rather than a permanent architectural component. A current limitation is additional training-time computation and hyperparameters introduced by relation construction and adaptive guidance, while choice of geometric cues may still require adaptation.

5 Conclusion

We presented GEAR, a lightweight framework that uses geometric relations as a training-time prior to align intermediate neural operator representations without modifying the underlying architecture. By filtering conflicting geometric guidance and adaptively reducing its influence during training, GEAR improves prediction across different backbones while preserving their original designs.

AI USE STATEMENT

Large language models (LLMs) were employed solely for auxiliary purposes, including language polishing, assistance with code implementation and debugging, and support for experiment evaluation. They were not used to determine the research questions, methodological design, or scientific conclusions of this work. All AI-assisted outputs were reviewed and verified by the authors, who take full responsibility for the final content.

REFERENCES

Corey Adams, Rishikesh Ranade, Ram Cherukuri, and Sanjay Choudhry. GeoTransolver: Learning physics on irregular domains using multi-scale geometry-aware physics attention transformer. *arXiv preprint arXiv:2512.20399*, 2025. doi: 10.48550/arXiv.2512.20399. Preprint.

Altair Engineering. OpenRadioss. Open-source finite element solver, 2026. Accessed 2026.

Altair Engineering Inc. Altair PhysicsAI, 2026. URL `https://www.altair.com/physicsai`. Accessed: 2026-01-06.

Ansys Inc. Ansys SimAI, 2026. URL `https://www.ansys.com/products/simai`. Accessed: 2026-01-06.

Neil Ashton, Charles Mockett, Marian Fuchs, Louis Fliessbach, Hendrik Hetmann, Thilo Knacke, Norbert Schonwald, Vangelis Skaperdas, Grigoris Fotiadis, Astrid Walle, Burkhard Hupertz, and Danielle Maddix. DrivAerML: High-fidelity computational fluid dynamics dataset for road-car external aerodynamics. *arXiv preprint arXiv:2408.11969*, 2024. doi: 10.48550/arXiv.2408.11969. Preprint.

Noah J. Bagazinski and Faez Ahmed. ShipGen: A diffusion model for parametric ship hull generation with multiple objectives and constraints. *Journal of Marine Science and Engineering*, 11 (12):2215, 2023. doi: 10.3390/jmse11122215.

Philipp Bekemeyer, Nathan Hariharan, Andrew M. Wissink, and Jason Cornelius. Introduction of applied aerodynamics surrogate modeling benchmark cases. In *AIAA SCITECH 2025 Forum*, 2025. doi: 10.2514/6.2025-0036.

Andrey Bryutkin, Jiahao Huang, Zhongying Deng, Guang Yang, Carola-Bibiane Schönlieb, and Angelica Aviles-Rivero. HAMLET: Graph transformer neural operator for partial differential equations. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 4624–4641. PMLR, 2024.

Shuhao Cao. Choose a transformer: Fourier or galerkin. In *Advances in Neural Information Processing Systems*, volume 34, pp. 24924–24940, 2021.

Ze Cheng, Zhongkai Hao, Xiaoqiang Wang, Jianing Huang, Youjia Wu, Xudan Liu, Yiru Zhao, Songming Liu, and Hang Su. Reference neural operators: Learning the smooth dependence of solutions of PDEs on geometric deformations. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 8060–8076. PMLR, 2024.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569. PMLR, 2023.

Wenjie Hu, Sidun Liu, Peng Qiao, Zhenglun Sun, and Yong Dou. Transolver is a linear transformer: Revisiting physics-attention through the lens of linear attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 408–416, 2026. doi: 10.1609/aaai.v40i1.37003.

Hrvoje Jasak. OpenFOAM: Open source CFD in research and industry. *International Journal of Naval Architecture and Ocean Engineering*, 1(2):89–94, 2009. doi: 10.3744/JNAOE.2009.1.2.089.

Sungwon Kim, Juho Song, Seungmin Shin, Guimok Cho, Sangkook Kim, and Chanyoung Park. EqGINO: Equivariant geometry-informed fourier neural operators for 3D PDEs. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Xinyi Li, Zongyi Li, Nikola Kovachki, and Anima Anandkumar. Geometric operator learning with optimal transport. *Journal of Machine Learning Research*, 26(282):1–38, 2025.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning. *Transactions on Machine Learning Research*, 2023a.

Zijie Li, Dule Shu, and Amir Barati Farimani. Scalable transformer for PDE surrogate modeling. In *Advances in Neural Information Processing Systems*, volume 36, pp. 28010–28039, 2023b. doi: 10.52202/075280-1216.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *Proceedings of the International Conference on Learning Representations*, 2021.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for PDEs on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023c.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-informed neural operator for large-scale 3D PDEs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 35836–35854, 2023d. doi: 10.52202/075280-1556.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for PDEs on million-scale geometries. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 41432–41449. PMLR, 2025.

Mohammad Amin Nabian, Sudeep Chavare, Deepak Akhare, Rishikesh Ranade, Ram Cherukuri, and Srinivas Tadepalli. Automotive crash dynamics modeling accelerated with machine learning. In *WCX SAE World Congress Experience*, pp. 1–19. SAE International, 2026. doi: 10.4271/2026-01-0568.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W Battaglia. Learning mesh-based simulation with graph networks. In *Proceedings of the International Conference on Learning Representations*, 2021.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020. doi: 10.1126/science.aaw4741.

Subhankar Sarkar and Souvik Chakraborty. Spatio-spectral graph neural operator for solving computational mechanics problems on irregular domain and unstructured grid. *Computer Methods in Applied Mechanics and Engineering*, 435:117659, 2025. doi: 10.1016/j.cma.2024.117659.

Jaskirat Singh, Xingjian Leng, Zongze Wu, Liang Zheng, Richard Zhang, Eli Shechtman, and Saining Xie. What matters for representation alignment: Global information or spatial structure? In *Proceedings of the International Conference on Learning Representations*, 2026.

Karn Tiwari, Niladri Dutta, N M Anoop Krishnan, and Prathosh Ap. Latent mamba operator for partial differential equations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 59639–59667. PMLR, 2025.

Alasdair Tran, Alexander Mathews, Lexing Xie, and Cheng Soon Ong. Factorized fourier neural operators. In *Proceedings of the International Conference on Learning Representations*, 2023.

Nobuyuki Umetani and Bernd Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics*, 37(4):1–10, 2018. doi: 10.1145/3197517.3201325.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017.

Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. Scientific discovery in the age of artificial intelligence. *Nature*, 620(7972):47–60, 2023.

Tian Wang and Chuang Wang. Latent neural operator for solving forward and inverse PDE problems. In *Advances in Neural Information Processing Systems*, volume 37, pp. 33085–33107, 2024. doi: 10.52202/079017-1042.

Ziqiao Wang, Wangbo Zhao, Yuhao Zhou, Zekai Li, Zhiyuan Liang, Mingjia Shi, Xuanlei Zhao, Pengfei Zhou, Kaipeng Zhang, Zhangyang Wang, Kai Wang, and Yang You. REPA works until it doesn't: Early-stopped, holistic alignment supercharges diffusion training. In *Advances in Neural Information Processing Systems*, volume 38, pp. 136854–136887, 2025.

Haixu Wu, Tengge Hu, Huakun Luo, Jianmin Wang, and Mingsheng Long. Solving high-dimensional PDEs with latent spectral models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 37417–37438. PMLR, 2023.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705. PMLR, 2024.

Haixu Wu, Minghao Guo, Zongyi Li, Zhiyang Dou, Mingsheng Long, Kaiming He, and Wojciech Matusik. GeoPT: Scaling physics simulation via lifted geometric pre-training. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Yanming Xia and Angelica I Aviles-Rivero. Do neural operators forget geometry? the forgetting hypothesis in deep operator learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Zipeng Xiao, Zhongkai Hao, Bokai Lin, Zhijie Deng, and Hang Su. Improved operator learning by orthogonal attention. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54288–54299. PMLR, 2024.

Zherui Yang, Haiyang Xin, Tao Du, and Ligang Liu. Simple yet effective: Low-rank spatial attention for neural operators. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026.

Sihyun Yu, Sangkyung Kwak, Huiwon Jang, Jongheon Jeong, Jonathan Huang, Jinwoo Shin, and Saining Xie. Representation alignment for generation: Training diffusion transformers is easier than you think. In *Proceedings of the International Conference on Learning Representations*, 2025.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 5824–5836, 2020.

Jianqiao Zeng, Ruocheng Wang, Yanzhi Liu, Hao Xiong, and Junchi Yan. CoEvol-NO: State and coordinate co-evolution with an error-driven predictor-corrector paradigm for neural operator transformer. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Zhizhou Zhang, Youjia Wu, Kaixuan Zhang, and Yanjia Wang. From cheap geometry to expensive physics: A physics-agnostic pretraining framework for neural operators. In *Proceedings of the International Conference on Learning Representations*, 2026.

CONTENTS OF APPENDIX

A Implementation Details **14**

A.1 Simulation Datasets 14

A.2 Baselines 14

A.3 Qualitative visualization settings 15

B Additional Analysis **16**

B.1 Preliminary: Surrogate Objective 16

B.2 Relational Alignment 16

B.3 Properties of Gradient Filtering 17

B.4 Adaptive Geometric Guidance 17

C Analysis of Learned Representations **18**

D Additional results **19**

D.1 Training Efficiency and Cost 19

D.2 Additional Quantitative Analysis 20

D.3 Flow-field Visualization 21

A Implementation Details

All experiments are implemented in PyTorch and conducted on NVIDIA A100 GPUs. Unless otherwise specified, models are trained for 200 epochs with a batch size of 1 using AdamW. The initial learning rate is set to 10^{-3} and scheduled using OneCycleLR with a warm-up ratio of 0.3. We use a weight decay of 10^{-5}. For Transolver-based models, we use eight layers, a hidden dimension of 256, eight attention heads, 32 physical-state slices, an MLP expansion ratio of 2, GELU activations, and zero dropout. For GEAR, local geometric relations are constructed using voxel resolutions of $\{4, 8\}$. We perform validation after each epoch and evaluate the final checkpoint on the full-resolution test meshes.

A.1 Simulation Datasets

We evaluate GEAR on five 3D simulation datasets covering aerodynamic, hydrodynamic, and structural impact problems. Table 4 summarizes the dataset configurations used in our experiments.

Table 4: Summary of the simulation datasets used in our experiments. #Mesh denotes the number of points per case in the stored point-cloud data, and #Sampled denotes the number used during training. Varied factors describe the geometry and operating conditions across cases. Numbers in parentheses indicate the output channel counts. AoA denotes angle of attack.

Type	Dataset	#Mesh	#Sampled	Varied factors	#Train	#Test	Output physics (channels)
Aerodynamics	DrivAerML	70k	70k	Geometry	100	20	Mean pressure and velocity (4)
	NASA–CRM	$\sim$ 454k	50k	Geometry, Mach number, AoA	105	44	Pressure coefficient (1)
	AirCraft	$\sim$ 332k	30k	Geometry, Mach number, AoA, sideslip	100	50	Six aerodynamic components (6)
Hydrodynamics	DTCHull	229–247k	80k	Geometry, yaw angle	100	20	Mean pressure and velocity (4)
Structural impact	Car–Crash	$\sim$ 1.10M	100k	Impact angle	100	30	Maximum von Mises stress (1)

DrivAerML. DrivAerML (Ashton et al., 2024) contains high-fidelity automotive CFD simulations over varying vehicle geometries. We use a processed subset of 120 cases, each containing 60,000 volume points and 10,000 surface points. The four output channels are mean pressure and the three mean velocity components. Operating conditions are fixed across cases.

NASA–CRM. NASA–CRM (Bekemeyer et al., 2025) covers aircraft aerodynamics with variations in geometry, Mach number, and angle of attack. The condition vector contains Mach number and angle of attack, and the prediction target is the point-wise pressure coefficient.

AirCraft. AirCraft (Luo et al., 2025) contains aircraft aerodynamic simulations under varying geometries and operating conditions. The condition vector consists of Mach number, angle of attack, and sideslip angle. Each point has six aerodynamic output components.

DTCHull. DTCHull (Bagazinski & Ahmed, 2023) considers ship hydrodynamics over parametrically generated hull geometries and varying yaw angles. RANS simulations performed using OpenFOAM (Jasak, 2009) provide time-averaged pressure and three velocity components. The yaw angle serves as the operating condition.

Car–Crash. Car–Crash (Wu et al., 2026) contains structural impact simulations generated using OpenRadioss (Altair Engineering, 2026). The impact angle serves as the condition, and the scalar prediction target is the maximum von Mises stress attained at each point during deformation.

A.2 Baselines

We compare GEAR with neural operators covering spectral, latent-space, attention-based, and geometry-aware architectures. The comparison methods are summarized below.

- **F-FNO** (Tran et al., 2023) factorizes multidimensional Fourier operations into axis-wise spectral transformations, reducing the cost of spectral operator learning.
- **Geo-FNO** (Li et al., 2023c) learns a coordinate transformation that maps irregular geometries into a computational domain suitable for Fourier-based operator learning.

- **LSM** (Wu et al., 2023) performs spectral modeling in learned latent spaces, combining multiscale representations with latent spectral transformations.
- **GINO** (Li et al., 2023d) uses graph neural operators to transfer information between irregular input geometries and a regular latent grid, where a Fourier neural operator models global interactions.
- **LaMO** (Tiwari et al., 2025) aggregates point-wise features into latent representations and models their interactions using state-space modules. We use its shared irregular-mesh variant.
- **Galerkin Transformer** (Cao, 2021) approximates operator interactions through Galerkin-type attention without softmax normalization. We adapt its task-specific decoder to a point-wise prediction head.
- **GNOT** (Hao et al., 2023) combines heterogeneous attention with coordinate-conditioned mixture-of-experts modules. Our implementation uses a single point-feature input branch for the simulation datasets.
- **LNO** (Wang & Wang, 2024) maps spatial inputs into a fixed set of latent modes, performs operator learning in the latent space, and reconstructs predictions at the query coordinates.
- **ONO** (Xiao et al., 2024) learns an orthogonal basis for integral operator approximation, coupling basis construction with the evolution of hidden field representations.
- **CoEvol-NO** (Zeng et al., 2026) jointly evolves mesh-point features and persistent latent states through predictor–corrector updates, enabling bidirectional information exchange.

F-FNO, Geo-FNO, and LSM are evaluated using adaptations to three-dimensional point-cloud inputs. Their nonuniform input projections are normalized relative to the training point count when evaluating meshes of different sizes. Input and output interfaces are adjusted to the features and target channels of each dataset.

We additionally integrate GEAR with the following five backbones to examine its applicability across operator designs:

- **OFormer** (Li et al., 2023a) uses an attention-based encoder to represent input functions and a cross-attention decoder to evaluate outputs at query locations. We use its Galerkin attention variant and query predictions at the input mesh points.
- **GeoPT** (Wu et al., 2026) uses geometry-informed pretraining to initialize neural operators for downstream physical prediction. We apply GEAR during downstream training.
- **Transolver++** (Luo et al., 2025) extends Physics-Attention with an eidetic mechanism designed to preserve detailed information during point-to-slice aggregation. We adapt its input interface to the shared point-wise features and operating conditions.
- **Transolver** (Wu et al., 2024) groups mesh points into learned physical slices, applies attention among slice representations, and projects the updated information back to mesh points.
- **LinearNO** (Hu et al., 2026) uses separately normalized query and key projections to aggregate and redistribute information through latent states, without additional attention among the latent states.

Each backbone augmented with GEAR is compared with its original counterpart under the same training setting. The alignment objective is applied to intermediate representations during training, leaving the backbone's inference architecture unchanged.

A.3 Qualitative visualization settings

Main text Figure 2 compares point-wise surface prediction errors between each baseline and its GEAR-enhanced counterpart. The configurations used for the three columns in each comparison are detailed here. AirCraft and NASA-CRM visualize the absolute pressure-coefficient error $|C_p^{\mathrm{pred}} - C_p^{\mathrm{GT}}|$, whereas DTCHull and DrivAerML visualize the absolute pressure error $|p^{\mathrm{pred}} - p^{\mathrm{GT}}|$. For every baseline–GEAR pair in Fig. 2, the corresponding columns use the same

samples, geometries, viewpoints, and ordering. On AirCraft, the three columns share geometry 149 and differ only in flow conditions, with $(M, \alpha, \beta) = (7.0, 0^\circ, 0^\circ)$, $(7.0, 0^\circ, 2^\circ)$, and $(7.0, 7^\circ, 0^\circ)$ from left to right. For NASA-CRM, the three columns correspond to samples 14, 30, and 43 with $(M, \alpha) = (0.696875, -1.820988^\circ)$, $(0.676562, 1.882716^\circ)$, and $(0.689063, 4.104938^\circ)$, respectively. DTCHull uses cases 119, 126, and 104 with recorded condition angles of -6.596176°, 1.408130°, and 1.651012°. DrivAerML uses runs 125, 124, and 120, corresponding to three different vehicle geometries. These settings ensure that the paired visualizations in Fig. 2 compare the baseline and GEAR under identical test configurations.

B Additional Analysis

This section provides supplementary derivations for the surrogate implementation, relational alignment, gradient filtering, and adaptive geometric guidance of GEAR. We follow the notation introduced in the main text.

B.1 Preliminary: Surrogate Objective

A surrogate objective provides a convenient way to propagate a prescribed gradient through an intermediate representation. For $\mathbf{H} = [\mathbf{h}_1, \dots, \mathbf{h}_N]^\top$ and desired auxiliary gradients $\mathbf{v}_i$, define

$$\mathcal{L}_{\text{sur}} = \sum_i \mathbf{h}_i^\top \, \text{sg}(\mathbf{v}_i), \tag{10}$$

where $\text{sg}(\cdot)$ denotes stop-gradient. During backpropagation, $\mathbf{v}_i$ is treated as constant, giving

$$\nabla_{\mathbf{h}_i} \mathcal{L}_{\text{sur}} = \mathbf{v}_i, \qquad \nabla_\theta \mathcal{L}_{\text{sur}} = \sum_i \left(\frac{\partial \mathbf{h}_i}{\partial \theta} \right)^\top \mathbf{v}_i. \tag{11}$$

This construction also applies when $\mathbf{v}_i$ is computed from the current representations or their gradients, since stop-gradient prevents differentiation through its construction.

In GEAR, we set $\mathbf{v}_i = \lambda \tilde{\mathbf{a}}_i$, where $\tilde{\mathbf{a}}_i$ is the filtered alignment gradient. Backpropagating the task loss together with the surrogate yields

$$\nabla_{\mathbf{h}_i} \left(\mathcal{L}_{\text{task}} + \mathcal{L}_{\text{sur}} \right) = \mathbf{t}_i + \lambda \tilde{\mathbf{a}}_i = \mathbf{g}_i. \tag{12}$$

The surrogate therefore implements the representation-level gradient rule in the main text. Its parameter gradient on batch B_a defines the geometric direction $\mathbf{d}_a = \nabla_\theta \mathcal{L}_{\text{sur}}(B_a)$ used to evaluate cross-batch utility.

B.2 Relational Alignment

For a fixed training batch, let $q_{ij} = \text{sim}(\mathbf{h}_i, \mathbf{h}_j)/\tau$. The empirical alignment objective is

$$\mathcal{L}_{\text{align}} = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{C}_i} p_{ij}^G \log \frac{p_{ij}^G}{p_{ij}^H}, \qquad p_{ij}^H = \frac{\exp(q_{ij})}{\sum_{k \in \mathcal{C}_i} \exp(q_{ik})}. \tag{13}$$

Since the geometric target is independent of the hidden representations, using $\partial \log p_{ik}^H / \partial q_{ij} = \mathbb{I}[k = j] - p_{ij}^H$ gives

$$\frac{\partial \mathcal{L}_{\text{align}}}{\partial q_{ij}} = -\frac{1}{|\mathcal{V}|} \sum_{k \in \mathcal{C}_i} p_{ik}^G \left(\mathbb{I}[k = j] - p_{ij}^H \right) = \frac{1}{|\mathcal{V}|} \left(p_{ij}^H - p_{ij}^G \right). \tag{14}$$

Thus, alignment increases relation logits that are underestimated relative to the geometric target and decreases those that are overestimated.

For nonzero features, cosine similarity also satisfies

$$\text{sim}(r_i \mathbf{Q} \mathbf{h}_i, r_j \mathbf{Q} \mathbf{h}_j) = \text{sim}(\mathbf{h}_i, \mathbf{h}_j), \tag{15}$$

where $\mathbf{Q}$ is orthogonal and $r_i, r_j > 0$. These transformations preserve the induced relation distributions and the alignment loss. Geometric supervision therefore constrains local relations while leaving freedom in the feature coordinate system and feature norms.

B.3 Properties of Gradient Filtering

The point-wise filtering rule can be formulated as

$$\tilde{\mathbf{a}}_i = \underset{\mathbf{v}\in\mathbb{R}^d}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{v}-\mathbf{a}_i\|_2^2 \quad \text{subject to} \quad \langle \mathbf{t}_i, \mathbf{v}\rangle \geq 0. \tag{16}$$

It selects the closest alignment direction compatible with the task gradient.

For $\mathbf{t}_i \neq \mathbf{0}$, the Lagrangian is

$$\mathcal{J}(\mathbf{v},\mu) = \frac{1}{2}\|\mathbf{v}-\mathbf{a}_i\|_2^2 - \mu\langle \mathbf{t}_i, \mathbf{v}\rangle, \qquad \mu \geq 0. \tag{17}$$

Stationarity gives $\mathbf{v} = \mathbf{a}_i + \mu\mathbf{t}_i$. Together with feasibility and complementary slackness, this yields

$$\mu^\star = \frac{\max(-\langle \mathbf{t}_i, \mathbf{a}_i\rangle, 0)}{\|\mathbf{t}_i\|_2^2}, \tag{18}$$

and hence

$$\tilde{\mathbf{a}}_i = \mathbf{a}_i - \frac{\min(\langle \mathbf{t}_i, \mathbf{a}_i\rangle, 0)}{\|\mathbf{t}_i\|_2^2}\mathbf{t}_i. \tag{19}$$

When $\mathbf{t}_i = \mathbf{0}$, the constraint is automatically satisfied and we retain $\tilde{\mathbf{a}}_i = \mathbf{a}_i$.

Two properties follow:

$$\langle \mathbf{t}_i, \tilde{\mathbf{a}}_i\rangle = \max(\langle \mathbf{t}_i, \mathbf{a}_i\rangle, 0) \geq 0, \tag{20}$$

$$\|\tilde{\mathbf{a}}_i\|_2^2 = \|\mathbf{a}_i\|_2^2 - \frac{\min(\langle \mathbf{t}_i, \mathbf{a}_i\rangle, 0)^2}{\|\mathbf{t}_i\|_2^2} \leq \|\mathbf{a}_i\|_2^2, \tag{21}$$

where the subtracted term is defined as zero when $\mathbf{t}_i = \mathbf{0}$. Thus, filtering removes the conflicting component without increasing the alignment-gradient norm. These properties hold at the representation where filtering is applied; cross-batch utility evaluates the resulting direction after it is mapped into parameter space.

Let $\mathbf{T}$ and $\tilde{\mathbf{A}}$ stack the task and filtered alignment gradients over points. The effective weight λ is calibrated using a relative budget $\rho \geq 0$:

$$\lambda = \begin{cases} \rho s_e \dfrac{\|\mathbf{T}\|_F}{\|\tilde{\mathbf{A}}\|_F}, & \|\tilde{\mathbf{A}}\|_F > 0, \\ 0, & \|\tilde{\mathbf{A}}\|_F = 0. \end{cases} \tag{22}$$

Consequently,

$$\|\lambda\tilde{\mathbf{A}}\|_F \leq \rho s_e \|\mathbf{T}\|_F, \tag{23}$$

with equality when $\tilde{\mathbf{A}} \neq \mathbf{0}$ under the analytical normalization. In implementation, numerical denominator floors and an upper cap on λ can only reduce this budget. The calibration expresses alignment strength relative to the current task gradient.

B.4 Adaptive Geometric Guidance

The cross-batch utility controls the internal guidance state through

$$\bar{u}_e = \beta\bar{u}_{e-1} + (1-\beta)u_e, \tag{24}$$

$$z_{e+1} = \operatorname{clip}_{[0,1]}\left(z_e + \eta_s \tanh\left(\frac{\bar{u}_e}{T_u}\right)\right). \tag{25}$$

Starting from $z_0 = s_0 = 1$, the effective guidance strength is

$$s_{e+1} = \min(s_e, z_{e+1}). \tag{26}$$

Equivalently, $s_e = \min_{0\leq k\leq e} z_k$. Since $z_e \in [0,1]$, it follows that

$$0 \leq s_{e+1} \leq s_e \leq 1. \tag{27}$$

Negative smoothed utility decreases the internal state, while zero utility leaves it unchanged. Positive utility may increase z_e, but cannot restore previously released guidance because of Equation (26). Once $s_e = 0$, geometric guidance remains inactive.

Together with Equation (23), this progressively restricts the relative contribution of geometric alignment as guidance is released.

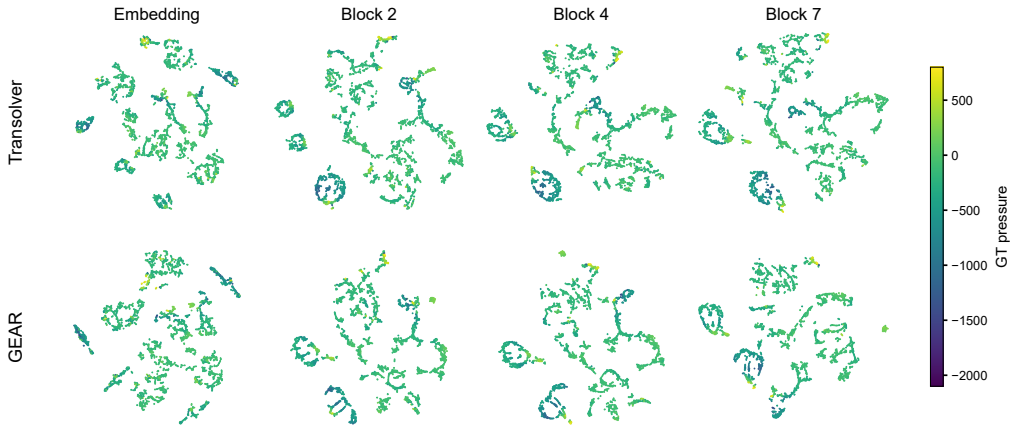


Figure 6: Pressure-colored t-SNE visualizations of frozen representations on DrivAerML. Rows show Transolver and GEAR, while columns correspond to the embedding and blocks 2, 4, and 7. Colors denote ground-truth pressure; each projection is fitted independently.

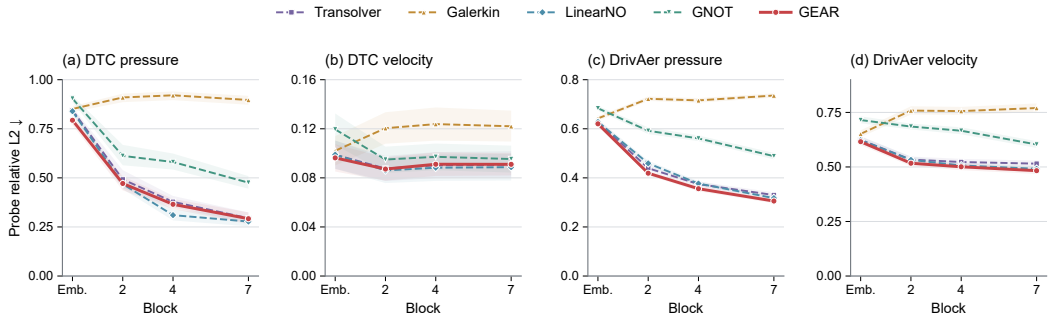


Figure 7: Frozen-feature linear probing across network depth on DTCHull and DrivAerML. Lower relative ℓ_2 indicates better linear decodability of the physical quantities; shaded regions denote 95% bootstrap confidence intervals.

C Analysis of Learned Representations

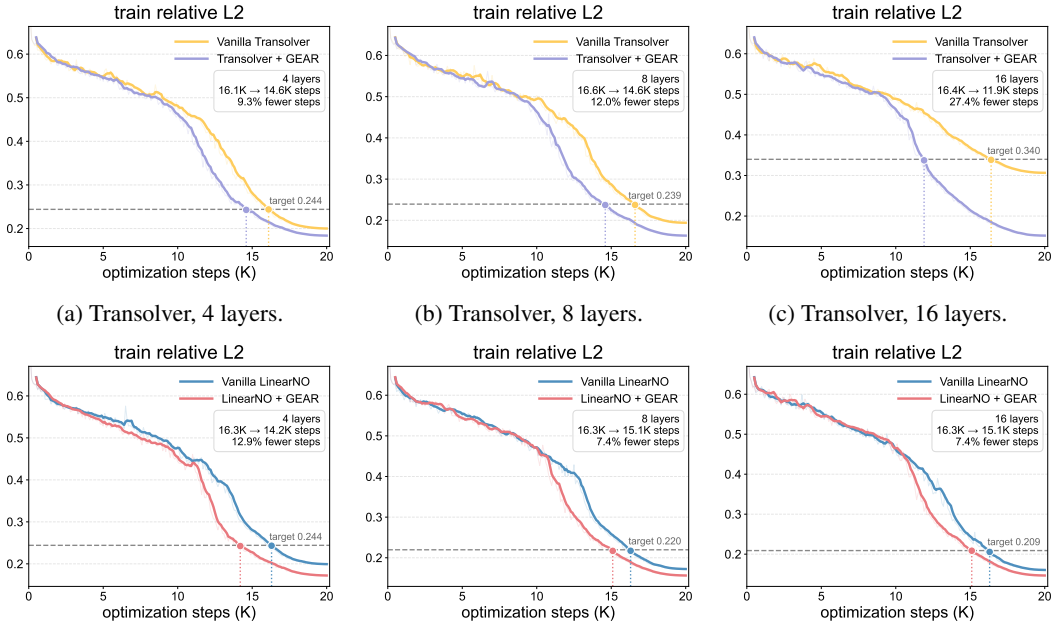
We further analyze how geometric alignment affects the internal representations learned by the operator. Figure 6 visualizes frozen representations from different depths using the same DrivAerML geometry. Since each t-SNE projection is fitted independently, we focus on the local organization with respect to pressure rather than absolute cluster positions. Compared with vanilla Transolver, GEAR exhibits clearer pressure-consistent organization across deeper representations, suggesting that geometric alignment influences how physically relevant variations are structured in the hidden space.

To quantify this behavior beyond visualization, we fit identical linear probes to frozen representations from several neural operators (Fig. 7). The probing error generally decreases with depth for GEAR, and its deeper representations remain among the most linearly decodable across the evaluated pressure and velocity targets. The advantage is not uniform at every layer or task, but the overall trend suggests that geometric alignment helps retain task-relevant physical information in a form that is easier to recover with a simple readout. Together, these analyses indicate that GEAR changes not only the final prediction accuracy but also the organization of intermediate physical representations.

D ADDITIONAL RESULTS

D.1 TRAINING EFFICIENCY AND COST

We further examine whether the reduction in optimization steps translates into lower training cost. Additional convergence curves are reported across different backbone depths, while normalized time-to-target is estimated for configurations with available complete-run timing records.



(a) Transolver, 4 layers. (b) Transolver, 8 layers. (c) Transolver, 16 layers.

(d) LinearNO, 4 layers. (e) LinearNO, 8 layers. (f) LinearNO, 16 layers.

Figure 8: Additional convergence curves on DrivAerML across different network depths. Transolver (top) and LinearNO (bottom) are evaluated at 4, 8, and 16 layers; the 12-layer results are reported in the main text. Solid curves show five-epoch moving averages of training relative L_2, horizontal dashed lines denote matched-loss targets, and vertical dotted lines mark the corresponding steps to target.

Across all tested depths, GEAR reaches the matched training-loss target in fewer optimization steps. The reduction ranges from 7.4% to 27.4%, with the magnitude depending on both backbone and depth. This indicates a consistent improvement in optimization-step efficiency, although it does not by itself imply lower wall-clock cost.

Table 5: Optimization steps and estimated normalized time-to-target on DrivAerML. Each backbone–depth pair uses the same matched training-loss target, and time-to-target is normalized to the corresponding vanilla model ($1.00\times$); lower is better.

Backbone	Layers	Target τ	Steps to target ↓		Step reduction	Est. time-to-target ↓
			Vanilla	+GEAR		+GEAR/ Vanilla
Transolver	4	0.2440	16.1K	14.6K	9.3%	$1.28\times$
Transolver	12	0.3526	16.7K	12.2K	26.9%	$0.99\times$
Transolver	16	0.3398	16.4K	11.9K	27.4%	$0.98\times$
LinearNO	4	0.2441	16.3K	14.2K	12.9%	$1.33\times$
LinearNO	12	0.2138	16.5K	14.2K	13.9%	$1.29\times$
LinearNO	16	0.2090	16.3K	15.1K	7.4%	$1.38\times$

The reduced number of optimization steps only partially offsets the additional per-step computation introduced by GEAR. The 12- and 16-layer Transolver configurations are close to time parity,

whereas the shallower Transolver and LinearNO settings remain above $1.00\times$. We therefore interpret the results as improved optimization-step efficiency rather than a uniform wall-clock speedup.

Table 6: Representative improvements on additional backbones. Results are overall relative L_2 errors on complete test meshes. Galerkin uses a replacement baseline trained under the same settings;

Dataset	Backbone	Baseline	+GEAR	Reduction
NASA–CRM	LaMO	0.0961	**0.0863**	10.19%
	CoEvol-NO	0.5394	**0.0883**	83.64%
Car–Crash	CoEvol-NO	0.2035	**0.2000**	1.68%
	GNOT	0.1867	**0.1838**	1.54%
DTCHull	LaMO	0.1986	**0.1441**	27.44%
	Galerkin	0.6463	**0.5152**	20.29%
DrivAerML	LaMO	0.2435	**0.2237**	8.15%

Table 7: Channel-wise relative L_2 errors on DTCHull. Errors are computed separately for each output channel on complete test meshes and averaged over the 20 test cases. Galerkin uses the same replacement baseline as Table 6.

	LaMO			Galerkin		
Quantity	Baseline	+GEAR	Reduction	Baseline	+GEAR	Reduction
Pressure p	0.1985	**0.1441**	27.44%	0.6466	**0.5154**	20.29%
Velocity u_x	0.1927	**0.1294**	32.86%	0.2076	**0.1860**	10.40%
Velocity u_y	1.0036	**0.9574**	4.60%	0.9997	**0.9539**	4.58%
Velocity u_z	0.9654	**0.8936**	7.44%	0.9835	**0.8975**	8.75%

Table 8: Channel-wise relative L_2 errors of LaMO with and without GEAR on DrivAerML. Errors are computed separately for each output channel on complete test meshes and averaged over the 20 test cases.

Quantity	Baseline	+GEAR	Reduction
Pressure p	0.2414	**0.2216**	8.19%
Velocity u_x	0.4245	**0.4057**	4.42%
Velocity u_y	0.8389	**0.7349**	12.40%
Velocity u_z	0.7732	**0.7510**	2.87%

D.2 Additional Quantitative Analysis

We provide additional quantitative results to examine GEAR beyond the backbone–dataset combinations reported in the main paper. The analysis covers additional neural operator backbones, individual prediction channels, and spatial regions on DTCHull and DrivAerML.

Additional backbones. Table 6 shows that the improvement from GEAR extends to several additional operator architectures. The gains vary substantially across backbone–dataset pairs, ranging from modest improvements on Car–Crash to larger reductions on NASA–CRM and DTCHull. For example, LaMO improves by 10.19% on NASA–CRM and 27.44% on DTCHull, while Galerkin improves by 20.29% on DTCHull. The variation in magnitude suggests that geometric alignment interacts differently with the representations learned by different operators, but the direction of improvement remains consistent across the reported settings.

Channel-wise behavior. Tables 7 and 8 further decompose the overall error into individual physical quantities. On DTCHull, both LaMO and Galerkin improve across all four output channels, with the largest reductions appearing in pressure and u_x, while the gains on u_y and u_z are more moderate. A similar pattern of non-uniform improvement appears on DrivAerML, where LaMO benefits across all channels but with different magnitudes. These results indicate that the effect of geometric alignment is not restricted to a single predicted variable, although some physical quantities benefit more strongly than others.

Table 9: Regional improvements for selected backbone–dataset combinations. Relative L_2 errors are computed jointly over the output channels within each region and averaged over test cases. Galerkin uses the same replacement baseline as Tables 6 and 7.

Dataset	Backbone	Region	Baseline	+GEAR	Reduction
DTCHull	LaMO	Volume	0.1749	**0.1192**	31.89%
		Surface	0.3094	**0.2476**	19.98%
	Galerkin	Volume	0.5924	**0.4773**	19.42%
		Surface	0.9078	**0.6972**	23.20%
DrivAerML	LaMO	Volume	0.2328	**0.2119**	9.00%
		Surface	0.2869	**0.2712**	5.46%

Regional behavior. Table 9 separates the prediction errors over volume and surface regions. Improvements are observed in both regions for DTCHull and DrivAerML, but their relative magnitude depends on the backbone and dataset. For instance, LaMO gains more strongly in the DTCHull volume, whereas Galerkin shows a slightly larger reduction on the surface. This suggests that the benefit of GEAR is not confined to a particular spatial region, but depends on how the underlying operator represents geometry and physical fields.

D.3 Flow-field Visualization

We provide additional flow-field visualizations on DTCHull and DrivAerML to complement the point-wise error maps in the main paper. Figures 9 and 10 jointly visualize the predicted physical fields, surrounding flow structures, and spatial error distributions for several representative test geometries and operator backbones. This provides a more direct view of how the learned representations affect the reconstructed flow beyond the aggregate relative L_2 metric.

For DTCHull, the three examples cover different hull geometries and yaw conditions. The predicted pressure fields and surrounding streamlines from GEAR remain close to the reference solutions, while the corresponding error maps show reduced discrepancies around the hull and downstream wake compared with the selected baselines. The improvements are also visible in the underside pressure distributions, where large localized errors produced by some backbones are noticeably reduced. These examples suggest that relational geometric guidance helps preserve geometry-dependent flow organization rather than only improving a single scalar metric.

The DrivAerML examples show a similar pattern across different vehicle geometries. Most models recover the dominant external-flow structure, while the main differences appear in localized wake regions and surface-pressure errors. GEAR generally reduces these concentrated discrepancies while maintaining the large-scale flow pattern, although the relative improvement varies across backbone and case. This is consistent with the quantitative results, where the benefit of geometric alignment is systematic but not uniform across operator architectures.

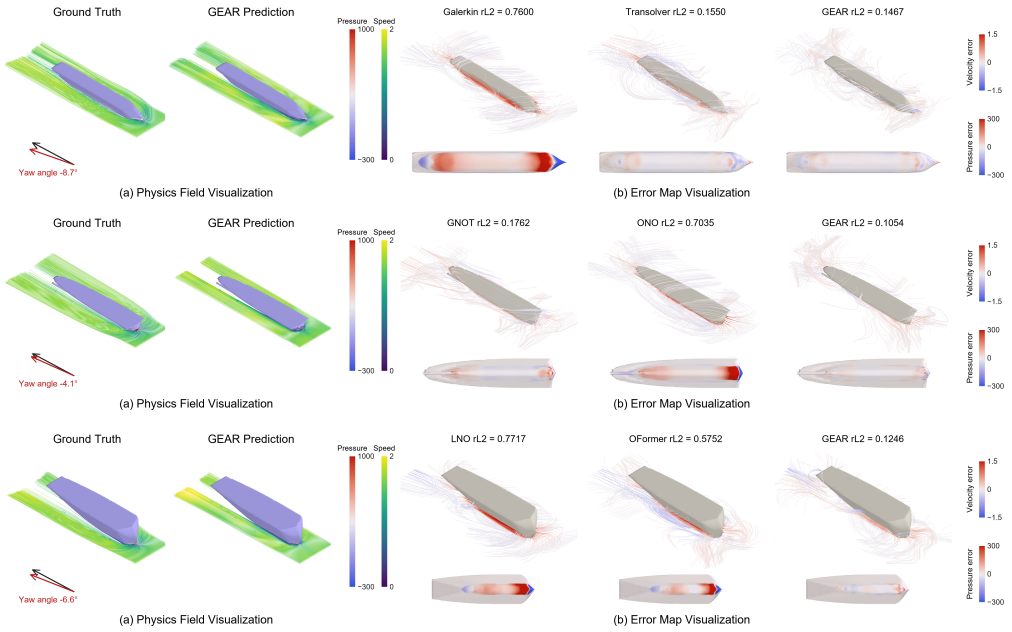


Figure 9: Flow-field prediction showcase on three DTCHull test geometries using different neural operator backbones and GEAR.

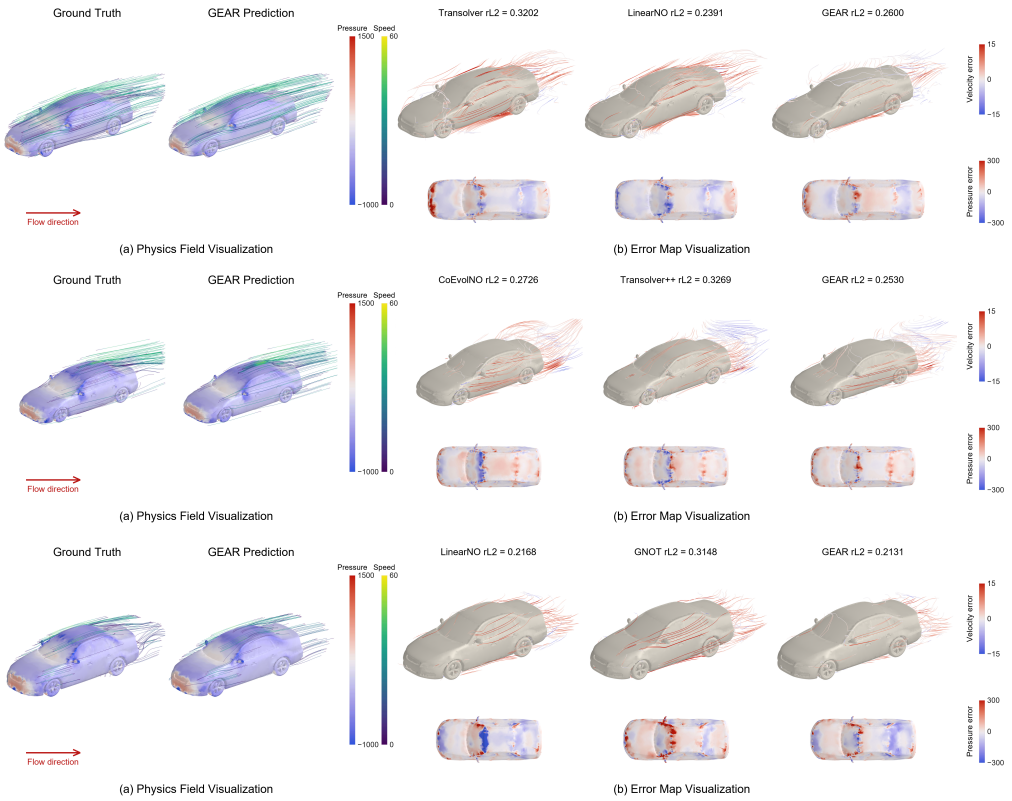


Figure 10: Flow-field prediction showcase on three DrivAerML test geometries using different neural operator backbones and GEAR.