OPERATOR LEARNING FOR PDES ON UNSTRUCTURED MESHES WITH VARYING GEOMETRIES VIA SECOND-GENERATION WAVELETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning solution operators for partial differential equations on unstructured meshes with varying geometry remains a fundamental challenge for neural operators. Existing approaches either transform scattered data onto a regular grid as a preprocessing step, discarding what the grid cannot represent, or encode data on the mesh graph, where cost and receptive field are tied to a constructed neighborhood. We propose LiGNO, a lifting-based geometric neural operator built on a second-generation wavelet in which we treat the point cloud itself as the finest level of a multiresolution hierarchy. A geometric lifting step decomposes nodal features into a coarse field on a fixed lattice and a nodal residual carrying the fine-scale detail. Together, they reconstruct the nodal data exactly, at any lattice resolution, so no information is discarded before the operator acts. A multi-level parameterized second-generation wavelet transform then recursively decomposes coarse and directional detail coefficients, enabling the operator to model physical interactions at multiple scales. We evaluate LiGNO against state-of-the-art methods on Darcy flow, the Poisson equation, and aerodynamic flows on DrivAer and airfoil geometries, with deformed domains and varying numbers of nodal points.

1 INTRODUCTION

Surrogate modeling of computationally expensive simulations is central to many scientific and engineering applications, including design optimization (Queipo et al., 2005), uncertainty quantification (Zhu & Zabaras, 2018), inverse problems, and real-time prediction. In this study, we particularly focus on surrogate modeling of parametric partial differential equations (PDEs) defined on complex domains over a scattered point set. Many physical phenomenon are governed by parametric PDEs, where accessing effect of various material properties, forcing terms, boundary conditions, or geometry require repeated numerical experiments. Neural operators offer promising alternative by learning mapping between function spaces (Kovachki et al., 2023; Lu et al., 2021), bypassing cost of repeated experiments.

Remarkable developments such as spectral-based (Li et al., 2021; Cao et al., 2024), U-shaped (Rahman et al., 2023), and convolutional (Raonic et al., 2023) neural operators have shown promising results for directly mapping functions from data. Despite their effectiveness, these operators are naturally formulated on regular cartesian grid. However, in practical applications, solution fields are frequently available on irregular discretizations, including unstructured meshes, scattered point sets, and geometry-dependent meshes. This motivates the development of neural operators for unstructured data with varying geometry. To address this, most existing approaches adapt one of two routes. The first transform scattered data onto a regular grid before the neural operator acts on, for example, Geo-FNO (Li et al., 2023b) learns a deformation from the physical domain to a latent regular grid followed by the FNO. This approach is also explored on the DeepONet framework (Yin et al., 2024; Xiao et al., 2024). The deformation presumes that one smooth map onto a fixed reference domain exists, which restricts the geometries and topologies it covers. On the other hand, GINO (Li et al., 2023c) combines graph-based operators for encoding and decoding irregular geometries with FNO layers on a latent regular grid. The effectiveness of these approaches depends on the chosen geometry transformation, interpolation, or encoding mechanism and may face information loss that the grid cannot represent before the operator acts. Alternatively, other approaches act directly on the point

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.

set, exploiting geometric learning techniques to process unstructured data. Graph-based operators (Li et al., 2020) use neighborhood-based message passing or kernel integration to represent nonlocal operators on irregular discretizations. More recent approaches use attention-based architectures, including OFormer (Li et al., 2023a), GNOT (Hao et al., 2023), and Transolver (Wu et al., 2024), to model interactions directly between irregularly sampled points. Further, PCNO (Zeng et al., 2025) formulates neural operators directly on point clouds and explicitly targets variable geometries and discretizations. PCNO incorporates geometric information through a graph-based differential operator and requires mesh connectivity. While these approaches avoid explicit mapping to a common grid, their computational structure is tied to graph construction, neighborhood interactions, or attention mechanisms.

In this study, we develop a neural operator built on the lifting scheme (Sweldens, 1996; 1998) constructs second-generation wavelets directly in the spatial domain, without the Fourier transform, enabling multiresolution analysis to irregular sampling. We treat scattered data as a lifting decomposition onto a coarser regular grid, with the corresponding residual representing fine-scale details. Learnable lifting has recently demonstrated accurate operator learning on regular grids (Pandey et al., 2026). We extend this framework to a broader class of point-cloud datasets with varying geometries by introducing LiGNO, a lifting-based geometric neural operator built on this observation. A geometric lifting step splats nodal features onto a fixed lattice by B-spline quasi-interpolation, extends them harmonically into unoccupied cells, and retains the nodal residual as a detail coefficient, so that the two components reconstruct the nodal data exactly. A multilevel learnable predict–update levels then decompose the lattice field, and the operator acts on the coarse and directional detail coefficients at every scale before synthesis back onto the nodes. The model requires nodal coordinates and values, making it invariant to node ordering, needs no mesh connectivity, and computation scales linearly with nodes. We evaluate LiGNO over diverse benchmarks on an unstructured mesh with varying geometries, including Darcy flow and the Poisson equation, airfoil flow, and flow on the 3D DrivAer dataset. We compare proposed model against geometry-aware neural operators based on latent-grid representations, point-cloud operators, and attention-based architectures. Our results demonstrate the ability of LiGNO to learn solution operators across varying nodes and geometries while preserving fine-scale solution structure. Our key contributions are summarized as follows:

- We introduce a geometric lifting transform that treats an unstructured point set as the finest level of a learnable second-generation wavelet hierarchy, while retaining an exactly reconstructible nodal representation.
- We develop LiGNO, a multiresolution neural operator that performs learnable lifting and operator evolution on coarse and directional detail coefficients while supporting varying irregular geometries.
- LiGNO operates directly on pointwise coordinates and features without requiring explicitly constructed mesh connectivity, providing a connectivity-free framework for operator learning on irregular geometries.

2 LIFTING-BASED GEOMETRIC NEURAL OPERATOR

We propose LiGNO, a lifting-based geometric neural operator for learning the solution operator of parametric PDEs discretized on unstructured meshes or point clouds with varying geometries and nodal counts. The method consists of two stages of similar algebraic operations built upon the construction of the second-generation wavelets. A geometric lifting stage that maps scattered nodal data to a pair of coarse field on a structured grid and a corresponding nodal detail residual, which retains information specific to the original unstructured data. Followed by a learnable multi-level lifting stage that models lifting transform via parametrized predict-update operators on the coarse field.

2.1 PROBLEM FORMULATION

Let $\Omega \subset \mathbb{R}^d$ be a bounded domain and let $\mathfrak{D}$ denote a family of bounded Lipschitz domains $D \subseteq \Omega$. For each $D \in \mathfrak{D}$, let $\mathcal{A}(D) = \mathcal{A}\big(D; \mathbb{R}^{d_a}\big)$ and $\mathcal{U}(D) = \mathcal{U}\big(D; \mathbb{R}^{d_u}\big)$ be separable Banach spaces of

input and output fields. We consider the geometry-parameterised solution operator

$$\mathcal{G}^\dagger : \bigcup_{D\in\mathfrak{D}} \{D\} \times \mathcal{A}(D) \longrightarrow \bigcup_{D\in\mathfrak{D}} \mathcal{U}(D), \qquad u = \mathcal{G}^\dagger(D, a), \quad u \in \mathcal{U}(D).$$

Notice that here the domain is an argument of the operator, as its shape changes with the sample. The input field is a collection of scattered point clouds, and a sample j consists of

$$\mathcal{X}_j = \{x_1^j, \ldots, x_{N_j}^j\} \subset \overline{D}_j, \qquad a_j : \mathcal{X}_j \to \mathbb{R}^{d_a}, \qquad u_j : \mathcal{X}_j \to \mathbb{R}^{d_u},$$

where both the domain D_j and the node count N_j vary across samples. LiGNO requires no mesh element connectivity, it operates from nodal coordinates and nodal values alone. Given M samples, $\{(\mathcal{X}_j, a_j, u_j)\}_{j=1}^M$ optimizer seek parameters θ that minimizes the mean per-sample relative ℓ_2-error

$$\mathcal{J}(\theta) = \frac{1}{M}\sum_{j=1}^{M} \frac{\|\mathcal{G}_\theta(\mathcal{X}_j, a_j) - u_j\|_{\ell_2(\mathcal{X}_j)}}{\|u_j\|_{\ell_2(\mathcal{X}_j)}}. \tag{1}$$

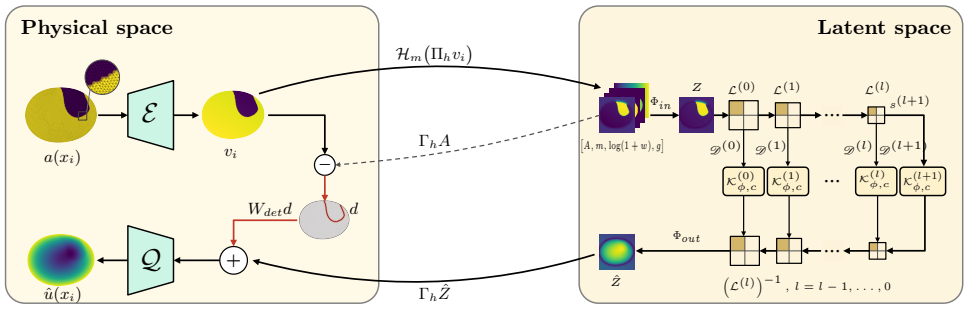


Figure 1: **LiGNO architecture.** Mesh features $v_i = \mathcal{E}([\xi_i,\ a(x_i)])$ are projected onto a latent grid by $\mathcal{H} \circ \Pi_h$, combined with geometric channels, and passed through a multilevel learned lifting scheme. Γ_h maps the output back to the mesh. The sub-grid detail d reaches the output through W_{det}, and $\mathcal{Q}$ decodes $\hat{u}(x_i)$.

2.2 Architecture Overview

The LiGNO architecture can be summarized algebraically as

$$\mathcal{G}_\theta = \mathcal{Q} \circ \Big[\big(\Gamma_h \circ \Phi_{\text{out}} \circ \mathcal{L}^{-1} \circ \mathcal{K}_\Phi \circ \mathcal{L} \circ \Phi_{\text{in}}\big) \oplus W_{\text{det}}\Big] \circ \mathcal{T}_{\text{geo}} \circ \mathcal{E}, \tag{2}$$

where we represent $\mathcal{E}$ for pointwise encoder, $\mathcal{T}_{\text{geo}}$ for the geometric lifting transform, Φ_{in} and Φ_{out} for channel-mixing projections, $\mathcal{L}$ for the L-level lifting transform, $\mathcal{K}_\Phi$ for the operator evolution, Γ_h for the lattice-to-node sampling operator, W_{det} for the learned map acting on the nodal detail branch and $\mathcal{Q}$ for the pointwise decoder.

The geometry-aware transform $\mathcal{T}_{\text{geo}}$ produces two representations, a coarse field on the latent lattice and a nodal detail, respectively. Coarse field is processed through the multiresolution lifting operator, while the nodal detail is propagated through the learned map W_{det} and recombined with the lattice pathway at the nodes. Here, $\oplus$ denotes this node-wise fusion of the two branches. Figure 1 presents a schematic diagram of the LiGNO model. The individual components of the architecture are described in the following subsections.

2.2.1 Pointwise Encoding.

Coordinates are normalized by the affine map $\xi_i \in [0,1]^{r_d}$ using componentwise division by extreme coordinate values. A shared pointwise encoder lifts the normalized position and input field to a latent space as

$$v_i = \mathcal{E}([\xi_i,\ a(x_i)]) \in \mathbb{R}^C, \qquad i = 1, \ldots, N, \tag{3}$$

with $\mathcal{E}$ an affine map applied identically at every node. Because $\mathcal{E}$ acts nodewise, equation 3 is equivariant under permutation of $\mathcal{X}$ and its parameter count is independent of N.

2.2.2 GEOMETRIC LIFTING AT LEVEL ZERO.

This section constructs $\mathcal{T}_{\text{geo}}$, the transform that takes the nodal features v to a coarse lattice field together with a nodal residual. It is the component that makes the framework geometry-aware, and it is built so that the pair it produces determines v exactly.

B-spline splatting and its adjoint. Let $\Lambda = \{c_k\}_{k=1}^{|\Lambda|}$ be a uniform lattice on $[0,1]^{r_d}$ with spacing h, and let β_k denote the tensor-product linear B-spline centred at c_k, so that

$$\beta_k(\xi) \;=\; \prod_{r=1}^{r_d} \max\Big(0,\; 1 - \tfrac{|\xi^{(r)} - c_k^{(r)}|}{h}\Big), \qquad \sum_{k=1}^{|\Lambda|} \beta_k(\xi) = 1 \quad \forall\, \xi \in [0,1]^{r_d}. \tag{4}$$

The second identity in equation 4 is the partition-of-unity property of the first-order B-spline family. We define the splat and gather operators

$$(\mathsf{S}_h v)_k \;=\; \sum_{i=1}^{N} \beta_k(\xi_i)\, v_i \;\in\; \mathbb{R}^C, \qquad (\Gamma_h z)_i \;=\; \sum_{k=1}^{|\Lambda|} \beta_k(\xi_i)\, z_k \;\in\; \mathbb{R}^C, \tag{5}$$

together with the accumulated weight $w_k = \sum_{i=1}^{N} \beta_k(\xi_i)$, which measures the local sampling density of $\mathcal{X}$ near c_k. The two operators are not independent, they are adjoint to each other.

Proposition 1 (Adjointness)**.** *With respect to the Euclidean inner products on $\mathbb{R}^{N\times C}$ and $\mathbb{R}^{|\Lambda|\times C}$, the splat and gather operators of equation 5 satisfy $\mathsf{S}_h = \Gamma_h^{\top}$.*

Proof. For $v \in \mathbb{R}^{N\times C}$ and $z \in \mathbb{R}^{|\Lambda|\times C}$, $\langle \mathsf{S}_h v, z\rangle = \sum_k \sum_i \beta_k(\xi_i)\, \langle v_i, z_k\rangle = \sum_i \langle v_i, \sum_k \beta_k(\xi_i) z_k\rangle = \langle v, \Gamma_h z\rangle$, the interchange being finite. Hence S_h is the adjoint of Γ_h. □

Proposition 1 states that analysis and synthesis at level zero are performed by a single pair of mutually adjoint operators built from one basis, rather than by two independently designed mechanisms. Normalizing the splat by the accumulated weight gives

$$\Pi_h v \;=\; D_w^{-1}\, \mathsf{S}_h v, \qquad D_w = \operatorname{diag}\big(w_k\big). \tag{6}$$

Equation 6 assigns to each lattice node the β-weighted mean of the nodal features in its support, and is therefore well-defined, irrespective of how many nodes fall within that support.

Occupancy and harmonic extension. There may be a finite set of lattice points without data, for two distinct reasons, a lattice node may lie outside D, or it may lie inside D in a region where no node of $\mathcal{X}$ falls within its support, which is more probable once $|\Lambda|$ is comparable to or larger than N. Both cases give $w_k \approx 0$, and the normalized value $(\Pi_h v)_k$ then carries no information about the field. To handle this, we define the occupancy mask and the occupied index set

$$m_k \;=\; \mathbf{1}[\, w_k > \tau\,], \qquad \Lambda_{\text{occ}} \;=\; \{k : m_k = 1\}, \tag{7}$$

for a fixed threshold $\tau > 0$, chosen to exclude lattice nodes whose support receives only negligible weight. The mask is a discrete, resolution-dependent representation of the sampled geometry, it indicates the part of Ω that the sample actually resolves, and it varies from sample to sample as D_j and $\mathcal{X}_j$ do. It is supplied to the network as an input channel, so that the operator can distinguish coefficients carried by data from those produced by the extension.

Values on $\Lambda \setminus \Lambda_{\text{occ}}$ are assigned by the extension that is smoothest in the discrete Dirichlet sense. Let E be the edge set of the 4-neighbourhood graph on Λ and consider

$$\mathcal{H}_m(A_0) \;=\; \arg\min_{z} \sum_{(k,k')\in E} \big\| z_k - z_{k'} \big\|_2^2 \quad \text{subject to} \quad z_k = (A_0)_k \;\; \forall\, k \in \Lambda_{\text{occ}}. \tag{8}$$

This is the discrete Dirichlet problem with data on Λ_{occ}, whose solution is discrete-harmonic on the complement and introduces no oscillation there, which is the relevant property, since any extension that did oscillate would inject spurious high-frequency content into the transform that follows. We approximate equation 8 using T Jacobi iterations. For each iteration $t = 0, \ldots, T-1$, the approximation is updated as

$$z_k^{(t+1)} = \begin{cases} (A_0)_k, & k \in \Lambda_{\text{occ}}, \\ \frac{1}{4} \sum_{k' \sim k} z_{k'}^{(t)}, & k \notin \Lambda_{\text{occ}}, \end{cases} \qquad z^{(0)} = A_0. \tag{9}$$

Therefore, for a two-dimensional domain, level-zero coarse field is: $A = \mathcal{H}_m\big(\Pi_h v\big) \;\in\; \mathbb{R}^{C\times s_1 \times s_2}$.

Detail coefficients and exact reconstruction. The coarse field A does not reproduce the nodal features as several nodes may share the support of one lattice basis function, and the extension equation 8 is independent of the data it is asked to smooth into. The discrepancy is retained explicitly as a nodal detail coefficient,

$$d_i \;=\; v_i - \big(\Gamma_h A\big)_i \;\in\; \mathbb{R}^C, \qquad i = 1, \dots, N, \tag{10}$$

which defines the geometric lifting transform

$$\mathcal{T}_{\mathrm{geo}} :\; v \;\longmapsto\; (A,\, d), \qquad \mathcal{T}_{\mathrm{geo}} = \big(\mathcal{H}_m \Pi_h,\;\; I - \Gamma_h \mathcal{H}_m \Pi_h\big). \tag{11}$$

Proposition 2 (Exact reconstruction at level zero)**.** *For every point set $\mathcal{X}$, every nodal feature array v, and every choice of lattice resolution, threshold τ, and sweep count T, the pair $(A, d) = \mathcal{T}_{\mathrm{geo}}(v)$ satisfies*

$$\Gamma_h A + d \;=\; v. \tag{12}$$

The transform $\mathcal{T}_{\mathrm{geo}}$ is therefore injective and admits the explicit left inverse $(A, d) \mapsto \Gamma_h A + d$.

Proof. Immediate from equation 10: $\Gamma_h A + d = \Gamma_h A + v - \Gamma_h A = v$. No property of Π_h, $\mathcal{H}_m$ or the lattice is used, because d is defined as the residual against the same synthesis operator Γ_h that the inverse applies. □

2.2.3 Geometry channels and the latent field

The occupancy mask, m, and sampling density, w, provide complementary geometric information that is not captured by the coarse field alone. The m localizes the boundary of D, and w distinguishes refined from coarse regions of the point set. Both are supplied to the network alongside a normalized coordinate grid $g \in \mathbb{R}^{2\times s_1 \times s_2}$,

$$Z \;=\; \Phi_{\mathrm{in}}\Big(\big[\, A,\; m,\; \log(1+w),\; g \,\big]\Big) \;\in\; \mathbb{R}^{C \times s_1 \times s_2}. \tag{13}$$

The logarithmic compression of w keeps the density channel on a scale comparable to the remaining features across the wide range of local node counts encountered in graded meshes.

2.2.4 Multiresolution lifting on the latent lattice

We adopt the learnable lifting transform of LiNO (Pandey et al., 2026) to obtain a multiresolution representation of the latent field Z. A fixed low-pass filter is first applied for anti-aliasing, followed by a dyadic split into four interleaved sub-lattices, producing a coarse component X_{00} and three directional components X_{01}, X_{10}, X_{11}. The directional components are predicted from the coarse component by learnable orientation-specific predictors, with the prediction residuals forming the detail coefficients:

$$d_{pq} = X_{pq} - \big(\mathcal{P}_\omega([X_{00}, g])\big)_{pq}, \qquad (p,q) \neq (0,0). \tag{14}$$

The coarse component is subsequently updated using the detail coefficients,

$$s = X_{00} + \mathcal{U}_\phi([d_{01}, d_{10}, d_{11}, g]), \tag{15}$$

where $\mathcal{P}_\omega$ and $\mathcal{U}_\phi$ are local residual convolutional networks. This predict–update operation is recursively applied to the coarse component for L levels,

$$\big(s^{(l+1)}, \mathscr{D}^{(l)}\big) = \mathcal{L}^{(l)}(s^{(l)}), \qquad s^{(0)} = Z, \qquad l = 0, \dots, L-1, \tag{16}$$

where $\mathscr{D}^{(l)}$ contains the three directional detail coefficients at level l. Thus, the transform represents Z using a hierarchy of coarse and directional components at progressively lower spatial resolutions.

The operator evolution is performed directly on the multiresolution coefficients rather than on the latent field. The last level coarse representation and the directional details at each level are processed by separate local residual convolutional operators,

$$s^{(L)} \leftarrow \mathcal{K}_{\Phi,\mathrm{c}}(s^{(L)}), \qquad d^{(l)}_{pq} \leftarrow \mathcal{K}^{(l,pq)}_{\Phi}(d^{(l)}_{pq}), \quad (p,q) \neq (0,0), \tag{17}$$

with independent parameters across levels and orientations. This provides scale- and direction-specific operator evolution while preserving the multiresolution structure of the representation. The evolved coefficients are then mapped back to the latent space through the inverse lifting transform and projected to the output field,

$$\widehat{Z} = \sigma\Big(\Phi_{\mathrm{out}}\Big(\mathcal{L}^{-1}\big(s^{(L)}, \mathscr{D}^{(0)}, \dots, \mathscr{D}^{(L-1)}\big)\Big)\Big), \tag{18}$$

where σ is a pointwise activation function.

2.2.5 RECONSTRUCTION ON THE POINT SET

The evolved lattice representation is mapped back to the point set using the same gather operator used in geometric lifting. The lattice and nodal detail pathways are combined as

$$\widehat{v}_i = \big(\Gamma_h \widehat{Z}\big)_i + W_{\mathrm{det}} d_i, \qquad \widehat{u}(x_i) = \mathcal{Q}\big(\widehat{v}_i\big), \tag{19}$$

where $\mathcal{Q}$ is a two-layer pointwise network that maps the latent representation to the output dimension d_u. This construction follows the reconstruction equation 12, with the mapping on the detail branch replaced by the learnable transformation W_{det}. This modification is necessary because the network maps input coefficients to output coefficients rather than reconstructing the input field itself.

Computational Complexity

Proposition 3 (Complexity)**.** *Consider one sample with N nodes, a lattice of $|\Lambda|$ sites, latent width C, L levels and T Jacobi sweeps, with $T \le C$ and $2^{r_d} \le C$. The forward pass costs*

$$\mathcal{O}\big(\underbrace{2^{r_d} NC + NC^2}_{\textit{nodal}} + \underbrace{TC|\Lambda| + C^2|\Lambda|}_{\textit{lattice}}\big) \;=\; \mathcal{O}\big(C^2\,(N + |\Lambda|)\big), \tag{20}$$

and the representation carried between analysis and synthesis, namely the coefficient hierarchy together with the nodal detail, occupies $\mathcal{O}\big(C\,(N + |\Lambda|)\big)$ values.

Proof. Each node contributes to 2^{r_d} lattice sites in both the splat and the gather, over C channels, giving $\mathcal{O}(2^{r_d} NC)$. The encoder and decoder act nodewise at fixed widths, contributing $\mathcal{O}(NC)$, while the detail map $W_{\mathrm{det}} : \mathbb{R}^C \to \mathbb{R}^C$ contributes $\mathcal{O}(NC^2)$. Each Jacobi sweep is a fixed-stencil convolution over the lattice, giving $\mathcal{O}(TC|\Lambda|)$. At level l the coefficient tensors hold $|\Lambda|4^{-l}$ entries and the local blocks cost $\mathcal{O}(C^2)$ per entry, so analysis and synthesis together cost $\sum_{l=0}^{L-1} \mathcal{O}(C^2|\Lambda|4^{-l})$, and since $\sum_{l\ge 0} 4^{-l} = \frac{4}{3}$ this is $\mathcal{O}(C^2|\Lambda|)$. Under $T \le C$ and $2^{r_d} \le C$ the four terms are bounded by $C^2(N + |\Lambda|)$ up to a constant. For the memory statement, the details at level l occupy $3C|\Lambda|4^{-(l+1)}$ entries and the terminal coarse field $C|\Lambda|4^{-L}$, summing to at most $C|\Lambda|$, while the nodal detail array holds NC values. □

3 NUMERICAL EXPERIMENTS

All experiments are conducted on an NVIDIA RTX PRO 4500 Blackwell 32GB GPU using CUDA 13.2, with fixed random seeds across all runs to ensure reproducibility. The experiments span a diverse range of challenging problems, including variable-domain elliptic problems, multi-body aerodynamic geometries, industry-scale three-dimensional surfaces, and time-dependent compressible flows. The hyperparameters for each benchmark are given in Appendix A. The source code and data used in this study are publicly available through an anonymous repository at https://anonymous.4open.science/r/LiGNO-91A9/README.md.

Darcy flow on variable domains. To assess performance on varying geometries, we consider the steady-state Darcy flow (Zeng et al., 2025), posed on a sample-dependent domain D,

$$\begin{cases} -\nabla \cdot \big(a(x,y)\,\nabla u(x,y)\big) = 1, & (x,y) \in D, \\ u(x,y) = 0, & (x,y) \in \partial D, \end{cases}$$

where a is the positive permeability field and u is the pressure and ∂D denotes the boundary of D. Each boundary is written in polar form, $(x,y) = \big(r(\theta)\cos(2\pi\theta),\, r(\theta)\sin(2\pi\theta)\big)$ for $\theta \in [0,1]$. The boundary radius $r(\theta)$ is generated via Gaussian regression with RBF kernel over six periodic control points $(j/5, r_j)$, with $r_j \sim \mathcal{U}(0.5, 1.5)$. Consequently, the generated boundaries can be non-smooth and exhibit a kink at $\theta = 0$. The permeability field is generated via Gaussian regression over $a(ih, jh) \sim \mathcal{U}(0.5, 1.5)$ for $h = 0.75$ and $i, j \in \{-2, \dots, 2\}$. Reference solutions are computed via the finite element method on unstructured Gmsh meshes (Geuzaine & Remacle, 2009), resulting in variable node counts across samples. We use 800 samples for training and 200 for testing. The operator $\mathcal{G}_\theta : (a, D) \mapsto u$ maps the permeability, together with the node coordinates that define the domain, to the pressure at every mesh node.

Flow over an airfoil with flap. To test geometry-only prediction on multi-body domains, we consider the transonic airfoil benchmark of Zeng et al. (2025), governed by the steady two-dimensional Euler equations

$$\begin{cases} \nabla \cdot (\rho \boldsymbol{v}) = 0, \\ \nabla \cdot (\rho \boldsymbol{v} \otimes \boldsymbol{v} + p\mathbf{I}) = 0, & \text{in } D, \\ \nabla \cdot \big((E+p)\boldsymbol{v}\big) = 0, \end{cases}$$

where ρ is the density, $\boldsymbol{v}$ the velocity, p the pressure and E the total energy. The far field is prescribed by $\rho_\infty = 1$, $p_\infty = 1$ and $M_\infty = 0.8$, where M_∞ denotes Mach number, so shocks form on the airfoil surface. A no-penetration condition $\boldsymbol{v} \cdot \boldsymbol{n} = 0$ is imposed on the wall. The domain is a circle of radius 50 with the airfoil geometry removed. Each configuration pairs a NACA four-digit main airfoil with a chord length of 1 and a flap of chord 0.2, offset by $(-0.015, 0.05)$. Solutions are computed with AERO-Suite (Wang et al., 2011; Farhat et al., 1998) on adaptive Gmsh meshes (Geuzaine & Remacle, 2009), refined to roughly 5×10^{-3} at the main airfoil and 2.5×10^{-3} at the flap, and coarsened to 5 in the far field. We use 1500 samples for training and 400 for testing. The operator $\mathcal{G}_\theta : D \mapsto p$ maps the node coordinates that define the domain directly to the pressure at every mesh node.

Poisson equation on variable domains. To isolate the effect of geometry, we consider the unstructured-mesh Poisson benchmark of Liu et al. (2026),

$$\begin{cases} -\nabla^2 u(x,y) = 1, \quad (x,y) \in D, \\ u(x,y) = 0, \quad (x,y) \in \partial D. \end{cases}$$

The boundary conditions are the same for every sample, so the solution depends solely on the domain. Each boundary ∂D passes through 144 points in polar form, $(x_i, y_i) = (r_i \cos\theta_i,\ r_i \sin\theta_i)$, with $\theta_i = 2\pi i/144$, $i = 1, \ldots, 144$, and radii $r_i \in [0.2, 0.8]$ that vary across samples. Reference solutions are computed with the finite element method in FEniCSx on unstructured triangular meshes. We use 1500 samples for training and 500 for testing. The operator $\mathcal{G}_\theta : D \mapsto u$ maps the node coordinates alone to the solution at every mesh node.

Aerodynamics of parametric car designs. To assess scalability to industry-scale 3D geometries, we consider the DrivAerNet dataset (Elrefaie et al., 2024), in the preprocessed form released with the 3D engineering benchmark of Zhong et al. (2025). The flow around each car is governed by the steady, incompressible Reynolds-averaged Navier–Stokes equations

$$\begin{cases} \nabla \cdot \bar{\boldsymbol{u}} = 0, \\ (\bar{\boldsymbol{u}} \cdot \nabla)\bar{\boldsymbol{u}} = -\dfrac{1}{\rho}\nabla \bar{p} + \nabla \cdot \Big[(\nu + \nu_t)\big(\nabla \bar{\boldsymbol{u}} + \nabla \bar{\boldsymbol{u}}^\top\big)\Big], & \text{in } D, \end{cases}$$

closed with the k–ω SST model for the eddy viscosity ν_t. The geometries are morphs of the DrivAer fastback reference model, which includes a detailed underbody, wheels and mirrors. They are varied through 50 geometric parameters using optimal Latin hypercube sampling. Each design is simulated in OpenFOAM with a uniform inflow of $u_\infty = 30$ m/s ($Re \approx 9.39 \times 10^6$ based on car length), no-slip conditions on the car and ground, and rotating wheels, on meshes of 8–16 million cells. We use the 2500 designs in this release, 2000 for training and 500 for testing. Each sample is a surface point cloud of about 5.2×10^5 points with no element connectivity. Denoting the car surface by Γ, the operator $\mathcal{G}_\theta : \Gamma \mapsto \bar{p}|_\Gamma$ maps the surface point coordinates alone to the surface pressure; the design-parameter vector is not supplied.

Compressible flow around an airfoil. To test long-horizon autoregressive prediction on a strongly graded mesh, we consider the AIRFOIL dataset of Pfaff et al. (2021), which simulates unsteady compressible flow around the cross-section of a wing,

$$\begin{cases} \partial_t \rho + \nabla \cdot (\rho \boldsymbol{v}) = 0, \\ \partial_t (\rho \boldsymbol{v}) + \nabla \cdot (\rho \boldsymbol{v} \otimes \boldsymbol{v} + p\mathbf{I}) = 0, & \text{in } D \times (0, T], \\ \partial_t E + \nabla \cdot \big((E+p)\boldsymbol{v}\big) = 0, \end{cases}$$

where ρ is the density, $\boldsymbol{v}$ the velocity, p the pressure and E the total energy. A slip condition $\boldsymbol{v} \cdot \boldsymbol{n} = 0$ is imposed on the wing. All trajectories are computed with SU2 on a single fixed triangular mesh

Table 1: Relative ℓ_2-error (mean $\pm$ standard deviation) in % computed over the test set for each neural operator across all steady state problems

	GINO	Transolver	PCNO	**LiGNO (proposed)**
Darcy flow	0.97 ± 0.71	2.15 ± 1.36	1.14 ± 0.85	$\mathbf{0.63 \pm 0.08}$
Poisson equation	1.53 ± 0.72	0.61 ± 1.17	$\mathbf{0.33 \pm 0.24}$	0.48 ± 0.18
Air foil flap	4.37 ± 1.96	1.95 ± 0.73	2.09 ± 0.75	$\mathbf{1.74 \pm 0.22}$
DrivAer	20.83 ± 3.84	19.57 ± 4.12	25.32 ± 2.83	$\mathbf{17.85 \pm 1.92}$

Table 2: Relative ℓ_2-error (mean $\pm$ standard deviation) in % computed over the test set for Airfoil flow upto 400 rollouts

Field	Model	1	10	20	50	100	200	400
v_x	GINO	3.31 ± 1.28	5.08 ± 2.03	5.72 ± 2.37	7.46 ± 3.77	12.10 ± 7.56	22.63 ± 14.70	49.82 ± 35.00
	Transolver	5.53 ± 1.63	11.17 ± 4.19	11.93 ± 4.91	12.82 ± 5.72	16.33 ± 9.45	21.74 ± 14.30	25.28 ± 16.40
	PCNO	3.62 ± 0.99	7.20 ± 2.28	10.63 ± 3.30	20.58 ± 7.11	37.41 ± 14.40	74.54 ± 31.10	175.50 ± 89.50
	LiGNO	**1.76 ± 0.78**	**1.65 ± 0.63**	**1.73 ± 0.60**	**2.02 ± 0.75**	**2.87 ± 1.59**	**5.34 ± 4.29**	**10.33 ± 9.29**
v_y	GINO	14.52 ± 3.46	18.61 ± 2.97	19.37 ± 3.01	21.64 ± 3.79	27.85 ± 7.68	45.11 ± 17.30	95.81 ± 45.70
	Transolver	25.40 ± 4.10	43.52 ± 0.96	48.39 ± 1.69	54.27 ± 4.22	57.12 ± 5.32	58.97 ± 5.93	60.26 ± 6.15
	PCNO	15.88 ± 3.39	34.89 ± 6.70	47.98 ± 12.90	77.12 ± 33.50	122.33 ± 66.10	202.23 ± 123.00	352.80 ± 245.00
	LiGNO	**6.96 ± 1.96**	**5.90 ± 2.28**	**6.11 ± 2.70**	**5.98 ± 2.44**	**6.39 ± 2.59**	**9.15 ± 4.32**	**15.48 ± 9.46**
ρ	GINO	1.38 ± 0.69	1.96 ± 1.08	2.32 ± 1.35	3.23 ± 2.00	4.48 ± 2.75	7.22 ± 4.14	13.42 ± 7.66
	Transolver	2.41 ± 1.09	4.38 ± 2.16	5.27 ± 2.70	6.82 ± 3.95	7.63 ± 4.59	8.26 ± 4.79	8.67 ± 4.79
	PCNO	1.43 ± 0.70	3.78 ± 1.50	5.46 ± 2.03	8.03 ± 2.91	10.40 ± 3.65	14.02 ± 4.44	22.18 ± 6.85
	LiGNO	**0.54 ± 0.30**	**0.48 ± 0.29**	**0.54 ± 0.31**	**0.67 ± 0.41**	**0.84 ± 0.70**	**1.49 ± 1.49**	**2.82 ± 2.80**

of 5233 nodes. Its edge lengths range from 2×10^{-4} to 3.5 m, so it covers a far field much larger than the wing while resolving sub-millimetre detail at the trailing edge. Trajectories differ in angle of attack (-25° to 25°) and Mach number (0.2 to 0.7); both enter only through the flow state, and the flow is sampled at $\Delta t = 0.008$ s. Inflow nodes are prescribed, while wing and interior nodes are predicted. We keep the first 400 transitions of each trajectory and use 1000 trajectories for training and 200 for testing. The state at every node is $\boldsymbol{u}^t = (\boldsymbol{v}^t, \rho^t)$. The operator $\mathcal{G}_\theta : \boldsymbol{u}^t \mapsto \boldsymbol{u}^{t+1}$ is trained on single steps and rolled out autoregressively from $\boldsymbol{u}^0$.

4 Results and Discussion

We evaluate LiGNO on predictive accuracy across varying geometries, stability under long-horizon autoregressive prediction, and computational cost. LiGNO is compared against geometry-aware neural operators based on latent-grid representations, GINO (Li et al., 2023c), point-cloud operators, PCNO (Zeng et al., 2025), and attention-based architecture, transolver (Wu et al., 2024). Table 1 compares the relative ℓ_2-error (in percentage) along with the standard deviation over the test sets for steady state benchmarks. Across these benchmarks, LiGNO consistently achieves the best or is highly competitive with other methods. We also evaluate long-horizon autoregressive prediction on

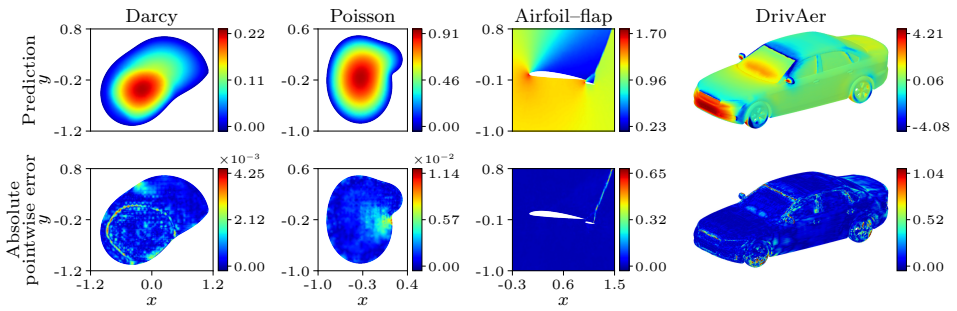


Figure 2: Visual representation of LiGNO's prediction on steady state benchmarks. The top row shows LiGNO's predictions, and the bottom row shows the corresponding absolute pointwise error.

the airfoil flow dataset. Table 2 reports the relative ℓ_2-error for the velocity components and density at different rollout horizons. LiGNO consistently outperforms other models across all reported rollout horizons and for all predicted quantities, with the advantage becoming more pronounced for longer rollouts. Further, visual representation of LiGNO's predictions is presented in Figure 2 and extended qualitative performance is given in Appendix C in Figures [3–7]. It is evident that LiGNO accurately reproduces the solution field across different geometries, while the corresponding error fields remain low throughout the domain. Moreover, we include computational efficiency in Appendix B, where Table 4, Table 5, and Table 6 report training time in hours, peak GPU usage in GB, and total trainable parameters in millions, respectively.

5 Conclusion and Limitations

We presented LiGNO, a lifting-based geometric neural operator that couples irregular point cloud data with multiresolution operator learning via an exactly reconstructible geometric lifting representation. Across diverse geometries and PDEs, LiGNO achieves the best or highly competitive performance and demonstrates stability under long-horizon autoregressive prediction. Despite these results, LiGNO currently relies on dyadic multiresolution decomposition and a fixed cartesian latent lattice, whose memory cost can become significant for higher-dimensional problems. The geometric lifting also depends on harmonic extension, which may become challenging for highly irregular or sparsely sampled geometries. In future, we aim to develop second-generation multiresolution operators directly on irregular point clouds, thereby eliminating the intermediate cartesian lattice. This requires developing lifting schemes tailored to irregular geometries that can provide localized multiscale representations while preserving stable reconstruction and computational efficiency.

AI use statement

We used generative AI tools solely for grammar correction and language polishing of the manuscript.

Ethics statement

This work does not involve human subjects, sensitive personal data, or other ethical concerns beyond those addressed in the paper. We have followed appropriate research and data-use practices and take responsibility for the ethical conduct of the work.

Reproducibility statement

All code and datasets have been made publicly available through an anonymous repository or provided as part of the supplementary material.

REFERENCES

Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace neural operator for solving differential equations. *Nature Machine Intelligence*, 6(6):631–640, 2024. doi: 10.1038/s42256-024-00844-4.

Mohamed Elrefaie, Angela Dai, and Faez Ahmed. Drivaernet: A parametric car dataset for data-driven aerodynamic design and graph-based drag prediction. *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Volume 3A: 50th Design Automation Conference (DAC):V03AT03A019, 08 2024. doi: 10.1115/DETC2024-143593.

C. Farhat, M. Lesoinne, and P. Le Tallec. Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: Momentum and energy conservation, optimal discretization and application to aeroelasticity. *Computer Methods in Applied Mechanics and Engineering*, 157(1):95–114, 1998. ISSN 0045-7825. doi: https://doi.org/10.1016/S0045-7825(97)00216-8.

Christophe Geuzaine and Jean-François Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering*, 79(11):1309–1331, 2009.

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, 23–29 Jul 2023.

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. URL http://jmlr.org/papers/v24/21-1524.html.

Zijie Li, Kazem Meidani, and Amir Barati Farimani. Transformer for partial differential equations' operator learning, 2023a. URL https://arxiv.org/abs/2205.13671.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Andrew Stuart, Kaushik Bhattacharya, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6755–6766. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4b21cf96d4cf612f239a6c322b10c8fe-Paper.pdf.

Zongyi Li, Nikola Borislavov Kovachki, Kamyar Azizzadenesheli, Burigede liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=c8P9NQVtmnO.

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, 2023b. URL http://jmlr.org/papers/v24/23-0064.html.

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. Curran Associates, Inc., 2023c. doi: 10.52202/075280-1556. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/70518ea42831f02afc3a2828993935ad-Paper-Conference.pdf.

Qibang Liu, Weiheng Zhong, Hadi Meidani, Diab Abueidda, Seid Koric, and Philippe Geubelle. Geometry-informed neural operator transformer for partial differential equations on arbitrary geometries. *Computer Methods in Applied Mechanics and Engineering*, 451:118668, 2026. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2025.118668.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021. doi: 10.1038/s42256-021-00302-5.

Himanshu Pandey, Subham Patel, and Ratikanta Behera. LiNO: Lifting based multiresolution neural operator, 2026. URL https://arxiv.org/abs/2607.02715.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks, 2021. URL https://arxiv.org/abs/2010.03409.

Nestor V. Queipo, Raphael T. Haftka, Wei Shyy, Tushar Goel, Rajkumar Vaidyanathan, and P. Kevin Tucker. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41(1):1–28, 2005. ISSN 0376-0421. doi: https://doi.org/10.1016/j.paerosci.2005.02.001.

Md Ashiqur Rahman, Zachary E Ross, and Kamyar Azizzadenesheli. U-NO: U-shaped neural operators. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=j3oQF9coJd.

Bogdan Raonic, Roberto Molinaro, Tim De Ryck, Tobias Rohner, Francesca Bartolucci, Rima Alaifari, Siddhartha Mishra, and Emmanuel de Bézenac. Convolutional neural operators for robust and accurate learning of PDEs. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 77187–77200. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/f3c1951b34f7f55ffaecada7fde6bd5a-Paper-Conference.pdf.

Wim Sweldens. The lifting scheme: A custom-design construction of biorthogonal wavelets. *Applied and Computational Harmonic Analysis*, 3(2):186–200, 1996. ISSN 1063-5203. doi: https://doi.org/10.1006/acha.1996.0015.

Wim Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM Journal on Mathematical Analysis*, 29(2):511–546, 1998. doi: https://doi.org/10.1137/S0036141095289051.

K. Wang, A. Rallu, J.-F. Gerbeau, and C. Farhat. Algorithms for interface treatment and load computation in embedded boundary methods for fluid and fluid–structure interaction problems. *International Journal for Numerical Methods in Fluids*, 67(9):1175–1206, 2011. doi: https://doi.org/10.1002/fld.2556. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.2556.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries, 2024. URL https://arxiv.org/abs/2402.02366.

Shanshan Xiao, Pengzhan Jin, and Yifa Tang. Learning solution operators of pdes defined on varying domains via mionet, 2024. URL https://arxiv.org/abs/2402.15097.

Minglang Yin, Nicolas Charon, Ryan Brody, Lu Lu, Natalia Trayanova, and Mauro Maggioni. Dimon: Learning solution operators of partial differential equations on a diffeomorphic family of domains, 2024. URL https://arxiv.org/abs/2402.07250.

Chenyu Zeng, Yanshu Zhang, Jiayi Zhou, Yuhan Wang, Zilin Wang, Yuhao Liu, Lei Wu, and Daniel Zhengyu Huang. Point cloud neural operator for parametric pdes on complex and variable geometries. *Computer Methods in Applied Mechanics and Engineering*, 443:118022, 2025. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2025.118022.

Weiheng Zhong, Qibang Liu, Diab Abueidda, Seid Koric, and Hadi Meidani. A comprehensive comparison of neural operators for 3d industry-scale engineering designs, 2025. URL https://arxiv.org/abs/2510.05995.

Yinhao Zhu and Nicholas Zabaras. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. *Journal of Computational Physics*, 366:415–447, 2018. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.04.018.

A HYPERPARAMETERS

Table 3: Hyperparameters used for LiGNO across all benchmark problems.

Hyperparameter	Darcy flow	Poisson equation	Airfoil flap	DrivAer	Airfoil flow
Data					
Input fields	`Permeability field`	`Node coordinates`	`Node coordinates`	`Surface coordinates`	`Node state (vx, vy, ρ)`
Target fields	`Pressure field`	`Potential field`	`Pressure field`	`Surface pressure`	`State increment`
Node count	3465 - 12509	1635 - 2675	32991 - 35986	438173 - 556577	5233
Latent grid	80×80	80×80	240×240	$64 \times 64 \times 64$	128×128
n_{train}	800	1500	1500	2000	1000
n_{test}	200	500	400	500	200
Training					
Epochs	500	500	500	500	500
Batch size	32	32	24	12	16
Initial lr η	1e-3	1e-3	1e-3	1e-3	1e-3
Minimum lr	1e-5	1e-5	1e-5	1e-5	1e-5
Optimizer	Adam	Adam	Adam	Adam	Adam
Weight decay	1e-4	1e-4	1e-4	1e-4	1e-4
Scheduler	Cosine Annealing	Cosine Annealing	Cosine Annealing	Cosine Annealing	Cosine Annealing
Architecture					
Width C	64	64	64	64	128
Levels L	4	4	3	4	4
LocalMLP hidden	32	32	32	32	32
LocalMLP conv layers	3+1×1	3+1×1	3+1×1	2+1×1	3+1×1
$\mathcal{K}_\Phi$ Operator	3 conv	3 conv	3 conv	3 conv	3 conv

B COMPUTATIONAL EFFICIENCY

Table 4: Wall-clock time (in hr) elapsed during training of each neural operator across all benchmark problems

Model	GINO	Transolver	PCNO	LiGNO
Darcy flow	7.89	9.13	3.98	0.55
Poisson equation	14.70	12.20	1.49	0.86
Air foil flap	13.01	19.12	15.62	4.62
DrivAer	21.62	23.40	25.40	31.74
Air foil flow	4.29	7.62	5.54	11.63

Table 5: Maximum GPU memory usage (in GB) recorded during training of each neural operator across all benchmark problems

Model	GINO	Transolver	PCNO	LiGNO
Darcy flow	12.70	1.87	26.21	4.62
Poisson equation	15.03	0.25	5.14	3.99
Air foil flap	7.71	4.40	28.52	25.60
DrivAer	3.80	5.37	18.75	25.13
Air foil flow	7.35	4.40	7.05	6.04

Table 6: Total trainable parameter count (in millions) of each model across all benchmark problems

Model	GINO	Transolver	PCNO	LiGNO
Darcy flow	1.29 M	0.97 M	71.58 M	2.40 M
Poisson equation	5.09 M	0.99 M	71.58 M	2.40 M
Air foil flap	3.72 M	0.97 M	71.58 M	1.84 M
DrivAer	13.67 M	14.42 M	441.65 M	14.21 M
Air foil flow	7.75 M	7.61 M	71.58 M	7.72 M

C QUALITATIVE PERFORMANCE

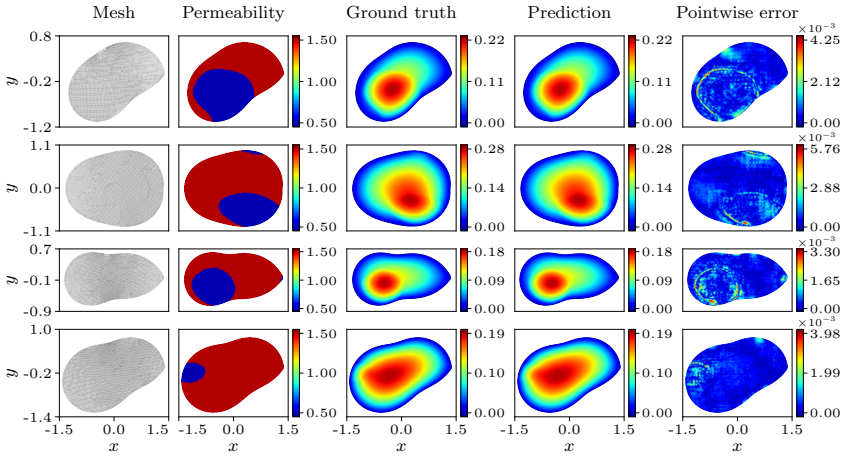


Figure 3: LiGNO prediction on Darcy flow with varying domains. Columns: mesh connectivity, permeability, ground truth, prediction, and pointwise absolute error, respectively.

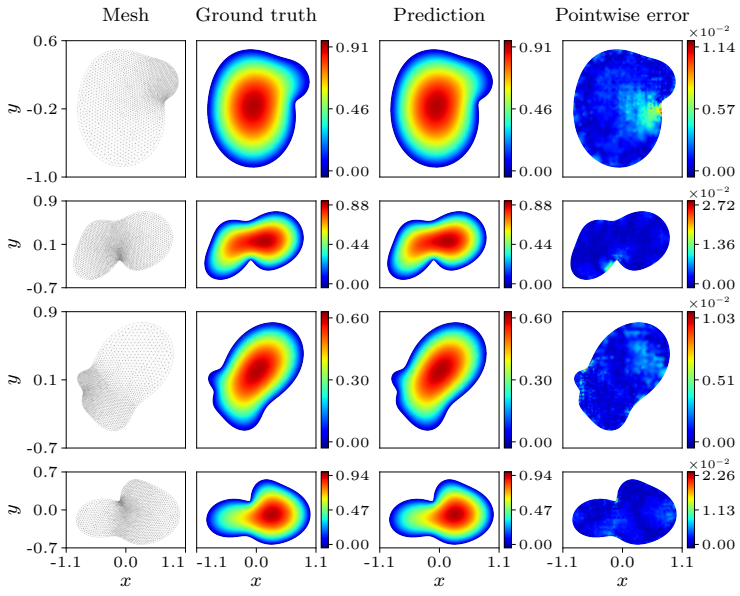


Figure 4: LiGNO prediction on Poisson equation with varying domains. Columns: mesh-connectivity, ground truth, prediction, and pointwise absolute error, respectively.

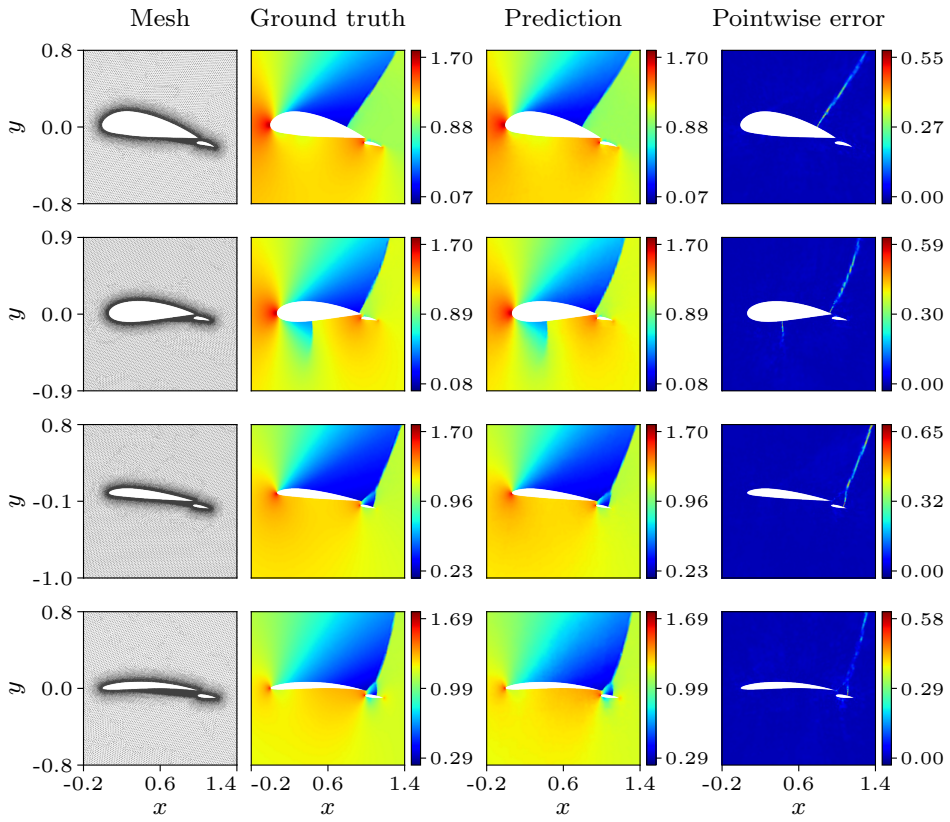


Figure 5: LiGNO prediction on flow across Airfoil flap with varying domains. Columns: mesh-connectivity, ground truth, prediction, and pointwise absolute error, respectively.

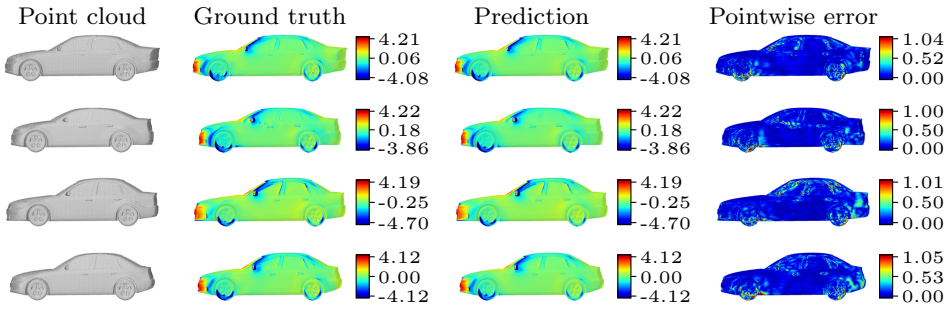


Figure 6: LiGNO prediction of surface pressure across different 3D car geometries. Columns: surface point cloud, ground truth, prediction, and pointwise absolute error, respectively.

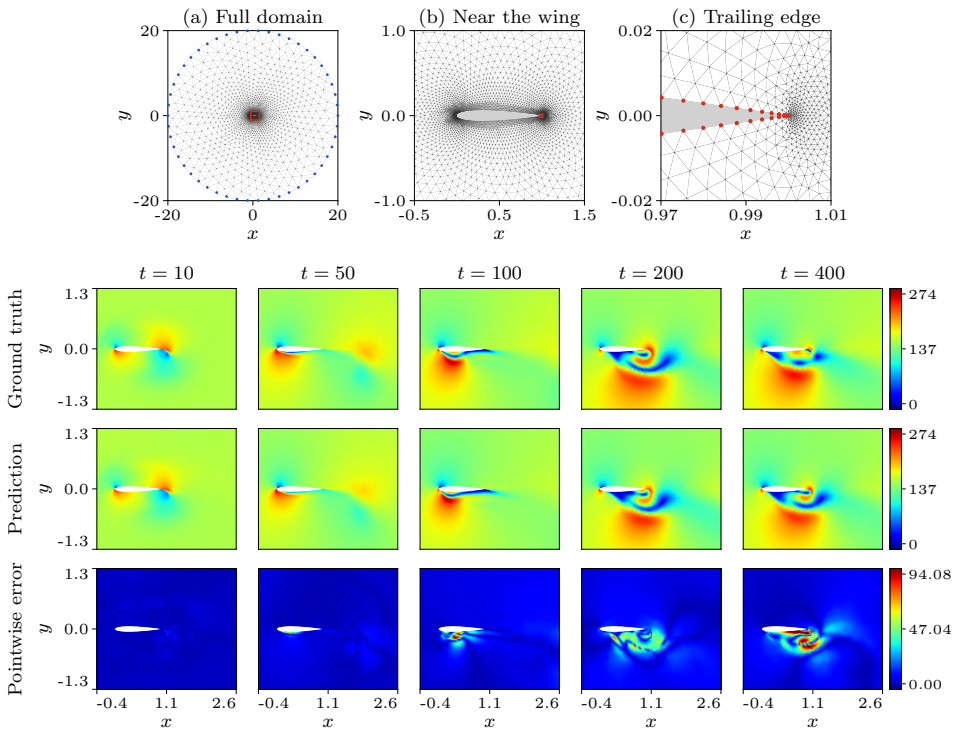


Figure 7: Rollout of LiGNO on one AIRFOIL test trajectory, started from $\boldsymbol{u}^0$. The top panel shows the computational mesh. Rows in the bottom panel correspond to ground-truth velocity magnitude $|\boldsymbol{v}|$, prediction, and pointwise absolute error, while columns correspond to rollout steps.