# MUFI: HIDDEN-STATE TRANSFER FOR MULTI-FIDELITY MESH PDE SURROGATES VIA COARSE PROPAGATION AND FINE DECODING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

High-resolution mesh PDE surrogates face a mismatch between geometric resolution and the resolution required for physical propagation. Fine meshes are often necessary to represent boundaries and localized structures, while global communication over all fine nodes can introduce substantial computational cost. This raises a key question in multi-fidelity surrogate modeling: what information should be transferred across fidelity levels to reconstruct fine-resolution fields? We introduce MUFI, a coarse-propagation/fine-decoding framework that studies hidden-state transfer across mesh resolutions. MUFI runs an existing mesh surrogate backbone on a coarsened mesh, transfers its intermediate hidden representations through a cached geometry-aware sparse operator, and reconstructs the solution on the original mesh using a FiLM-conditioned neural field decoder. Unlike coarse-field transfer approaches that only propagate predicted outputs, MUFI preserves pre-readout backbone states generated during physical propagation. A matched coarse-field transfer control indicates that hidden-state transfer provides additional reconstruction benefits beyond coarse propagation and decoder capacity alone. Across CFD and CSD benchmarks from the PLAID datasets, multiple mesh surrogate backbones, and a varying-geometry surface benchmark, MUFI improves the accuracy-cost trade-off and provides controllable operating points through fidelity selection. These results indicate that the choice of transferred representation is an important design factor in multi-fidelity mesh surrogate learning.

## 1 INTRODUCTION

High-fidelity computational mechanics, including computational fluid dynamics (CFD) and computational solid dynamics (CSD), relies on fine unstructured meshes to represent complex geometries and localized phenomena such as shocks, wakes, and stress concentrations. Neural PDE surrogates reduce solver cost by directly predicting solution fields on these meshes, but their computational cost often grows with mesh resolution because global communication is performed over increasingly large discretizations. Consequently, the resolution required to represent geometry is not necessarily identical to the resolution required for physical information propagation.

For message-passing mesh models (Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021; Brandstetter et al., 2023; Gilmer et al., 2017; Zhou et al., 2021; Zhao et al., 2024), propagation cost scales with the number of mesh edges and the number of communication steps required to capture distant dependencies (Alon & Yahav, 2021; Li et al., 2020; Khemani et al., 2024; Sun et al., 2024). Transformer-style and token-based PDE solvers improve global interaction, but still incur costs from point embeddings, token construction, and projections on large meshes (Wu et al., 2024; Luo et al., 2025).

Classical solvers address this mismatch with multiple discretization levels, such as multigrid, adaptive refinement, and mesh-transfer operators, and learned surrogates have adopted multiscale graphs, latent representations, and neural-field decoders for the same purpose (Section 2). However, an important question remains open for learned mesh surrogates: *What information should be transferred across fidelity levels?*

A straightforward approach is to transfer the coarse prediction field and learn a fine-scale super-resolution model. However, the final coarse prediction is produced after the backbone has compressed



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.

its internal representation into the output space. Intermediate hidden states generated during propagation may contain richer spatial context that is discarded by the final readout layer.

We introduce MUFI to study this representation transfer problem. MUFI treats an existing mesh surrogate backbone as a coarse physical propagator. Instead of operating the backbone on the original fine mesh, MUFI performs propagation on a coarsened mesh, transfers intermediate hidden states to the original geometry, and reconstructs the fine field using a neural field decoder conditioned through feature-wise linear modulation (Perez et al., 2017). This separates two roles that are commonly coupled in high-resolution surrogates: global physical communication and fine-scale geometric reconstruction.

The central design choice of MUFI is therefore not only reducing the propagation resolution, but selecting an informative interface between coarse and fine fidelities. We evaluate this choice through matched controls comparing hidden-state transfer against coarse-field transfer under the same propagation mesh, decoder architecture, and training protocol.

Our contributions are summarized as follows:

- We introduce MUFI, a coarse-propagation/fine-decoding framework that transfers intermediate hidden representations across mesh resolutions while preserving the internal propagation mechanism of existing mesh surrogate backbones. It combines a cached geometry-aware alignment operator with a FiLM-conditioned neural field decoder.
- We provide controlled evidence that hidden-state transfer improves fine-field reconstruction compared with matched coarse-field transfer, isolating the contribution of the transferred representation from decoder capacity and coarse propagation.
- We characterize the accuracy-cost behavior of multi-fidelity mesh surrogates through backbone comparisons, fidelity sweeps, localized physics diagnostics, multi-seed checks, cost accounting, and varying-geometry 3D experiments. These reveal both the operating points where MUFI is faster and more accurate and the regime where localized stresses require finer propagation.

## 2 RELATED WORK

**Mesh-based neural surrogates:** Beyond single-level message passing, multiscale and hierarchical graph networks add coarser graphs or hierarchical long-range edges inside the processor (Fortunato et al., 2022; Cao et al., 2023; Ripken et al., 2023), and Transformer-style solvers aggregate nodes into learned physical tokens (Wu et al., 2024; Luo et al., 2025). In both cases, the backbone still embeds and processes every fine node. MUFI is complementary: it moves an unchanged backbone, including a multiscale one, to a remeshed coarse domain, and MUFI-MGN-Tree improves over MGN-Tree on all four 2D benchmarks (Section 4.3).

**Latent encodings and neural fields:** Neural operators (Li et al., 2021; Lu et al., 2021; Li et al., 2022) and geometry-aware encode-process-decode models such as GINO (Li et al., 2023) and UPT (Alkin et al., 2024) map irregular inputs to latent grids or supernodes and query outputs at arbitrary points. Neural field decoders reconstruct PDE solutions from compact latent codes (Serrano et al., 2023; 2024; Catalani et al., 2025; Guo et al., 2025; Jo et al., 2025; Hagnberger et al., 2024). These methods define new architectures and typically condition a coordinate network on global or token-level latents, whereas MUFI keeps an existing mesh backbone and conditions its decoder locally on aligned node states.

**Coarse-to-fine:** RELift combines coarse-grid evolution with a learned lifting operator for time-dependent problems (Bassi et al., 2026), and solver-in-the-loop training learns correctors through a differentiable iterative solver (Um et al., 2020). Both refine coarse solution fields. MUFI targets steady problems without a numerical solver at training or test time and transfers pre-readout states rather than predicted fields; Section 4.2 compares it with a matched coarse-field super-resolution baseline.

# 3 METHOD

## 3.1 PROBLEM SETUP

Each sample is defined on a fine mesh graph $G_f = (V_f, E_f)$ with node coordinates $X_f = \{x_i^f\}_{i \in V_f}$, node-wise input features $F_f = \{f_i^f\}_{i \in V_f}$, and target fields $Y_f = \{y_i^f\}_{i \in V_f}$, where $y_i^f \in \mathbb{R}^{d_y}$. The features contain the inputs available in each dataset, such as node types, signed distance values, and sample-level scalar parameters of the simulation. An offline remeshing step with MMG (Dobrzynski, 2012) through the Muscat library (Bordeu et al., 2024) constructs a coarse mesh $G_c = (V_c, E_c)$ with coordinates $X_c$ and features $F_c$, and Muscat's interpolation-based field transfer maps $Y_f$ to a coarse target $Y_c$ used only for auxiliary supervision. Each dataset has a hierarchy of coarse levels (L1–L4, Table 8); the selected level is the fidelity knob of MUFI. Coarsening, target transfer, and the alignment operator below are computed once per geometry and cached, and all errors are evaluated on the original fine mesh.

## 3.2 HIDDEN-STATE COARSE-TO-FINE INTERFACE

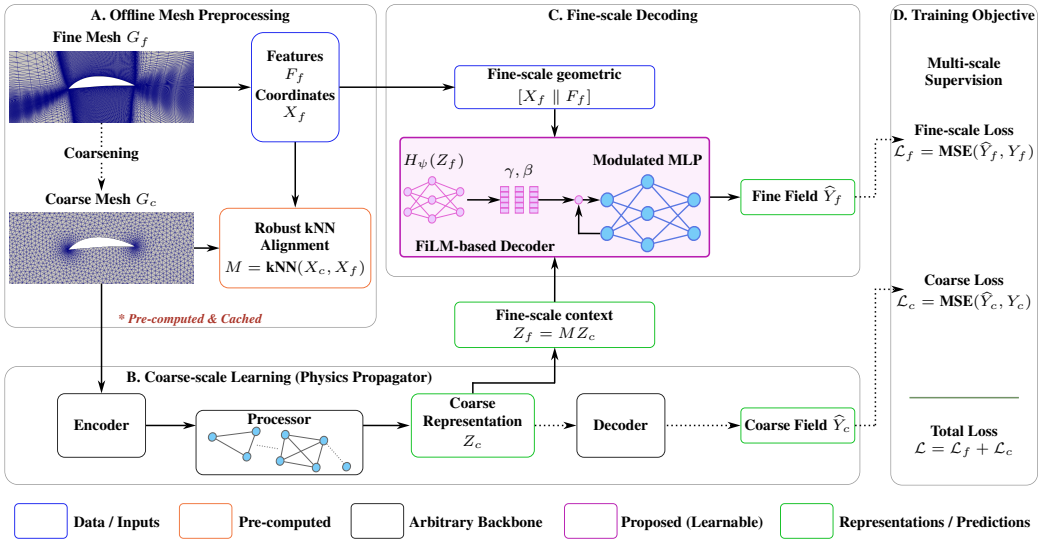



Figure 1: Overview of MUFI. (A) Offline preprocessing builds a coarse mesh and a sparse coarse-to-fine alignment matrix $M$. (B) An unchanged backbone propagates on the coarse mesh and exposes pre-readout states $Z_c$; its readout provides the coarse loss. (C) The aligned states $Z_f = MZ_c$ condition a FiLM-modulated coordinate decoder on the original mesh. (D) Training combines the fine and coarse losses. A GNN backbone is depicted, but any backbone exposing node-wise states can be used.

MUFI formulates multi-fidelity mesh surrogate learning as a representation transfer problem and treats an existing mesh surrogate as a coarse propagator. Let $B_\theta$ denote the backbone representation network and $R_\theta$ its original prediction head:

$$Z_c = B_\theta(G_c, X_c, F_c), \qquad \widehat{Y}_c = R_\theta(Z_c), \qquad Z_c \in \mathbb{R}^{|V_c| \times d_z} \tag{1}$$

where $Z_c$ contains node-wise intermediate hidden representations before the final readout. The backbone may communicate through local message passing, multiscale edges, error-guided rewiring, or attention over learned physical tokens; MUFI does not modify this mechanism and only requires a spatially indexed state before the readout (per-node states in this work; per-cell states could be mapped through a location adapter).

The key design choice of MUFI is to transfer $Z_c$ rather than only the coarse prediction $\widehat{Y}_c$. The hidden representation may retain intermediate information generated during propagation, while the coarse prediction is already compressed into the output space. The coarse representation is transferred to the

original mesh through a sparse alignment operator $M \in \mathbb{R}^{|V_f| \times |V_c|}$, and a fine decoder $P_\phi$ predicts the reported output:

$$Z_f = MZ_c, \qquad \widehat{Y}_f = P_\phi(X_f, F_f, Z_f) \tag{2}$$

The coarse readout $\widehat{Y}_c$ is used only for supervision; the reported prediction is always decoded on the original mesh (Figure 1).

**Cost:** This decomposition executes backbone communication on the coarse mesh and pointwise reconstruction on the fine mesh. For message-passing backbones, edge computations drop from $\mathcal{O}(|E_f|)$ to $\mathcal{O}(|E_c|)$ per step. For token-based solvers such as Transolver, which already avoid dense $\mathcal{O}(|V_f|^2)$ attention, MUFI reduces the number of points involved in embeddings, token construction, and projections. The decoder adds a linear $\mathcal{O}(|V_f|)$ pointwise cost and the sparse product $MZ_c$ costs $\mathcal{O}(K|V_f|d_z)$, so the gains are largest when $|V_c| \ll |V_f|$.

### 3.3 GEOMETRY-AWARE LATENT ALIGNMENT

After the coarse backbone computes node-wise representation states on the coarsened mesh, MUFI must transfer these states back to the original fine mesh. This step is independent of the chosen backbone: regardless of the specific architecture (e.g., GNNs, Transformers, or Neural Operators), the fine decoder only requires a coarse representation context at each fine node. A naive Euclidean $k$NN transfer is efficient but can be unstable near boundaries, holes, and thin structures, where the nearest coarse nodes may lie across a wall, be nearly collinear, or fail to enclose the query; since the decoder is conditioned on the aligned state at every fine node, such errors are amplified (Figure 4). For each fine node, MUFI therefore selects a stencil $\mathcal{N}_K(i)$ of $K = 3$ coarse nodes among its nearest candidates, penalizing boundary-interior mixing, near-duplicate points, low triangle area, and queries lying outside the triangle. The weights remain inverse-distance weights (Shepard, 1968):

$$z_i^f = \sum_{j \in \mathcal{N}_K(i)} w_{ij} z_j^c, \qquad w_{ij} = \frac{(\|x_i^f - x_j^c\|_2 + \epsilon)^{-1}}{\sum_{k \in \mathcal{N}_K(i)} (\|x_i^f - x_k^c\|_2 + \epsilon)^{-1}} \tag{3}$$

Stacking the rows gives $M$, whose rows are nonnegative and sum to one. $M$ depends only on the fine/coarse mesh pair, adds no learnable parameters, and is cached before training. On 3D surface meshes, the stencil is instead ranked by distance, normal consistency, and the offset along the surface normal, and the weights are modulated by normal similarity. Scoring rules, hyperparameters, and stencil-quality diagnostics are given in Appendix A.

### 3.4 FILM-CONDITIONED FINE DECODER

A decoder that concatenates $z_i^f$ with the coordinate $x_i^f$ forces a single coordinate MLP to represent both spatial basis functions and sample-dependent physical variation. MUFI instead uses feature-wise linear modulation: a hypernetwork $H_\psi$ maps the aligned state to layer-wise modulation parameters $\{\gamma_i^{(l)}, \beta_i^{(l)}\}_{l=1}^L = H_\psi(z_i^f)$, which modulate a coordinate network with input $r_i^{(0)} = [x_i^f \| f_i^f]$:

$$r_i^{(l+1)} = \sigma\big((1 + \gamma_i^{(l)}) \odot \mathrm{LN}(W_l r_i^{(l)} + b_l) + \beta_i^{(l)}\big), \qquad \hat{y}_i^f = W_{\text{out}}\, r_i^{(L)} + b_{\text{out}} \tag{4}$$

where $W_l$ and $b_l$ denote the learnable weight matrix and bias vector of the $l$-th layer in the coordinate decoder, LN is layer normalization (Ba et al., 2016), $\sigma$ is GELU, and the $(1 + \gamma)$ form initializes the modulation close to the identity. $H_\psi$ is applied pointwise, so the same weights serve meshes with any number of nodes. Decoder hyperparameters are listed in Table 10. In this design, the aligned coarse state modulates every hidden layer of the coordinate decoder rather than entering only as an input feature.

### 3.5 TRAINING OBJECTIVE

MUFI jointly optimizes fine-field reconstruction and coarse representation supervision:

$$\mathcal{L} = \underbrace{\mathrm{MSE}(\widehat{Y}_f, Y_f)}_{\mathcal{L}_f} + \underbrace{\mathrm{MSE}(\widehat{Y}_c, Y_c)}_{\mathcal{L}_c} \tag{5}$$

The fine loss evaluates the final prediction on the original mesh. The auxiliary coarse loss is not required by the interface, but it keeps the transferred state predictive and prevents the decoder from compensating for an uninformative coarse state (Section 4.7). Backbone-specific auxiliary terms, such as the error head of MiSe-GNN (Nguyen et al., 2026), are kept inside $\mathcal{L}_c$. This decoupling ensures a consistent MuFi interface across different backbones.

# 4 Experiments

## 4.1 Setup

**Benchmarks:** We use five datasets from PLAID (Casenave et al., 2026), a public collection of physics simulations on unstructured meshes for machine learning: two 2D CFD datasets, `2D_Profile` (airfoil flow, 37k nodes, 5 fields) and `VKI-LS59` (turbine blade cascade, 36k nodes, 2 fields); two 2D CSD datasets, `Tensile2D` (9.4k nodes, 5 fields) and `2D_MultiScHypEl` (hyperelastic domain with holes, 5.7k nodes, 7 fields); and `Rotor37`, 3D compressor-blade surface meshes whose geometry varies across samples (30k nodes, 3 fields). Statistics, splits, and mesh hierarchies are given in Appendix B.

**Backbones and protocol:** We wrap MeshGraphNets (MGN, message passing) (Pfaff et al., 2021), MGN-Tree (multiscale) (Ripken et al., 2023), MiSe-GNN (error-driven graph augmentation) (Nguyen et al., 2026), and Transolver (attention over learned physical tokens) (Wu et al., 2024). Each MuFi-$B$ keeps the architecture and hyperparameters of $B$ (Table 9) and adds only the fine decoder. One model is trained per dataset and shared across all sample geometries; each sample has its own cached coarse mesh and alignment matrix. Unless stated otherwise, all MuFi variants for a dataset use one common fidelity level, so comparisons isolate the wrapper rather than per-backbone level tuning. All models use the same split, Adam with learning rate $10^{-3}$, batch size 1, and 1000 epochs. Epoch time is measured on one NVIDIA A100 GPU (A30 for `Tensile2D`) and excludes cached preprocessing, which is accounted for separately in Section 4.6. Main results are single runs; Section 4.5 reports three-seed checks.

**Metrics:** Following PLAID, we report for each output field the relative root mean square error (RRMSE), i.e., the root of the node- and sample-averaged squared error, normalized in each sample by the squared maximum magnitude of the reference field (Eq. 20), and the macro RRMSE, i.e., the unweighted mean over output fields; lower is better. Section 4.4 complements these global metrics with localized physics diagnostics.

## 4.2 Hidden-state transfer attribution

We first isolate the contribution of the transferred quantity. A coarse-to-fine surrogate can transfer either the coarse prediction field or the intermediate hidden representation produced before the final readout layer. We therefore construct a matched coarse-field transfer baseline, Field-SR-FiLM: it predicts the coarse field $\widehat{Y}_c$ and feeds the aligned prediction $M\widehat{Y}_c$ to the same FiLM decoder, whereas MuFi feeds $MZ_c$. Both use the same coarse propagation mesh, alignment operator, decoder width, training protocol, and nearly the same runtime, and their parameter counts differ by 1.7%, so only the transferred quantity changes. We also add the same decoder at full resolution (MiSe-GNN + FiLM) to separate decoder capacity from coarse propagation.

Table 1: Hidden-state transfer attribution on `2D_Profile` Temperature at Level 3 (MiSe-GNN backbone). All coarse models use the same propagation level, alignment, and decoder width.

| Model | Propagation Mesh | Fine conditioning | Parameters | RRMSE | Epoch Time (s) |
|---|---|---|---|---|---|
| MiSe-GNN | Fine | Native Head | 929,282 | 0.02041 | 196.70 |
| MiSe-GNN + FiLM | Fine | Fine Hidden State | 1,480,707 | 0.01797 | 213.45 |
| Field-SR-FiLM | Coarse | Aligned Coarse Prediction $M\widehat{Y}_c$ | 1,459,971 | 0.01919 | 32.16 |
| **MuFi-MiSe-GNN** | Coarse | Aligned Hidden State $MZ_c$ | 1,484,291 | **0.01561** | 33.81 |

Table 1 separates the three factors. Decoder capacity alone does not explain the gain: MiSe-GNN + FiLM improves over the native head, but it is less accurate than MuFi and $6.3\times$ slower per epoch at

a nearly identical parameter count. Coarse propagation alone does not either: with the same coarse pipeline, transferring hidden states instead of the coarse prediction reduces error by $18.7\%$. At Level 2, deterministically projecting the coarse prediction to the fine mesh gives $0.0294$, and anchoring the hidden-state FiLM decoder on this projection (residual prediction) does not help either ($0.0180$ versus $0.0157$ for direct decoding; Appendix F). A plausible explanation is that the coarse prediction is a low-dimensional post-readout bottleneck that carries the coarse prediction error, whereas $Z_c$ retains richer pre-readout context for fine reconstruction.

### 4.3 BACKBONE-AGNOSTIC ACCURACY-COST IMPROVEMENT

We next evaluate whether MUFI can be applied across different mesh surrogate architectures. The backbone's internal propagation mechanism is unchanged, MUFI only modifies the fidelity level where propagation occurs and provides the coarse-to-fine reconstruction interface.

Table 2: Backbone-agnostic effect of MUFI. Each entry reports the macro-RRMSE change and epoch time speedup of MUFI-$B$ relative to the corresponding full-resolution backbone $B$. Negative error change is better. All MUFI variants use one common fidelity level per dataset; the first row gives the mean node counts of the fine and coarse meshes.

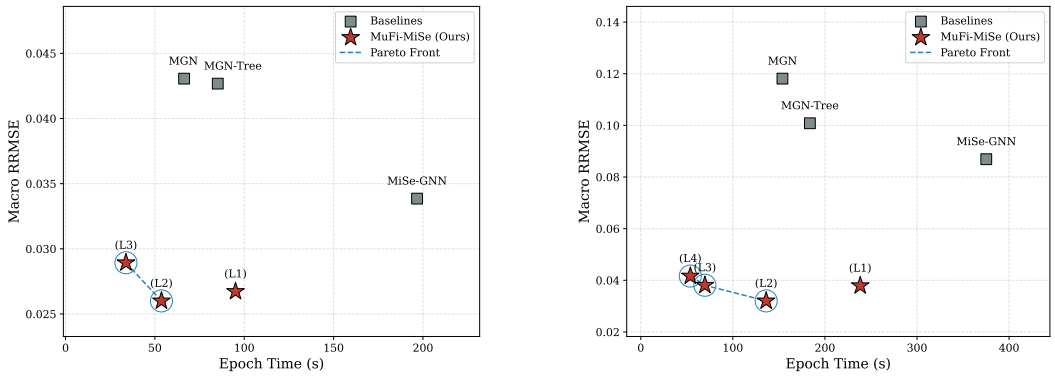
| **Backbone** | **`2D_Profile` L2** | | **`VKI-LS59` L3** | | **`Tensile2D` L1** | | **`2D_MultiScHypEl` L1** | |
|---|---|---|---|---|---|---|---|---|
| | $\Delta$ RRMSE | Speedup | $\Delta$ RRMSE | Speedup | $\Delta$ RRMSE | Speedup | $\Delta$ RRMSE | Speedup |
| *Nodes $\lvert V_f\rvert \to \lvert V_c\rvert$* | *37.0k → 8.8k* | | *36.4k → 4.3k* | | *9.4k → 3.7k* | | *5.7k → 2.9k* | |
| MGN | $-35.3\%$ | $2.90\times$ | $-28.5\%$ | $4.29\times$ | $-96.8\%$ | $1.85\times$ | $-22.2\%$ | $1.05\times$ |
| MGN-Tree | $-35.5\%$ | $3.10\times$ | $-62.1\%$ | $4.75\times$ | $-40.5\%$ | $2.00\times$ | $-0.7\%$ | $1.35\times$ |
| MiSe-GNN | $-23.2\%$ | $3.66\times$ | $-56.2\%$ | $5.39\times$ | $-27.4\%$ | $2.13\times$ | $+4.4\%$ | $1.90\times$ |
| Transolver | $-11.7\%$ | $1.84\times$ | $-28.4\%$ | $2.44\times$ | $-14.6\%$ | $1.14\times$ | $-40.6\%$ | $1.01\times$ |

Across 16 controlled backbone-dataset comparisons, MUFI improves macro RRMSE in 15 cases and reduces epoch time in every comparison (Table 2). The consistent trend across graph-based, multiscale, error-aware, and Transformer-style backbones suggests that MUFI can be applied across diverse mesh surrogate architectures when intermediate spatial representations are available. Speedups follow the node reduction: they are largest on the CFD meshes (up to $5.39\times$ with $8.5\times$ fewer nodes on `VKI-LS59`) and smallest on `2D_MultiScHypEl`, where Level 1 only halves the node count. The only degradation (MiSe-GNN on `2D_MultiScHypEl`, $+4.4\%$) comes from stress components (Section 4.4); per-field results are in Appendix C.

**Why can coarsening improve accuracy:** For a $K$-step message-passing GNN, information travels at most $K$ graph hops; on a coarser graph, the same budget of 15 steps spans a larger physical region. On the `VKI-LS59` Mach field, the full-resolution GNNs differ substantially (MGN $0.0322$, MGN-Tree $0.0217$, MiSe-GNN $0.0190$), whereas their Level 3 MUFI variants converge to similar errors ($0.0141$, $0.0135$, $0.0139$): the spread shrinks from $0.0132$ to $0.0005$, consistent with reduced under-reaching and lower sensitivity to the backbone's fine-mesh communication mechanism. The largest change in Table 2 ($-96.8\%$) mainly reflects a weak baseline: full-resolution MGN is an outlier on `Tensile2D` (macro RRMSE $0.0945$ versus $0.0040$-$0.0049$ for the other backbones), whereas MUFI-MGN ($0.0030$) is on par with the other wrapped backbones.

### 4.4 FIDELITY SELECTION AND LOCALIZED PHYSICS

The coarse mesh fidelity provides an explicit operating point between retained spatial information and computational cost (Figure 2, Table 3). Moderate coarsening can improve accuracy by reducing propagation difficulty while preserving relevant physical structures, and deeper levels trade accuracy for speed: the fastest levels that still improve macro RRMSE reach $5.82\times$ and $6.97\times$ epoch-time speedups on `2D_Profile` and `VKI-LS59`. On `2D_MultiScHypEl`, no level dominates the full-resolution backbone: displacements improve ($u_1$: $0.0146 \to 0.0092$), but stress components degrade ($P_{11}$: $0.0217 \to 0.0316$) because narrow stress bands near holes and thin ligaments are close to the original mesh resolution (Figure 10). Qualitative Mach predictions across levels (Figure 12, Appendix H) show that MUFI preserves the dominant shock structure, with remaining errors concentrated near high-gradient regions.




(a) `2D_Profile` (b) `VKI-LS59`

Figure 2: Accuracy-cost trade-off across fidelity levels with the MiSe-GNN backbone: macro RRMSE on the original fine mesh versus epoch time (lower left is better). Squares are full-resolution baselines, stars are MuFi-MiSe at the coarse levels in parentheses, and the dashed line is the Pareto front. On both datasets, every full-resolution baseline is dominated by at least one MuFi level. `Tensile2D` and `2D_MultiScHypEl` are shown in Figure 11 (Appendix D).

Table 3: Operating points from the MuFi-MiSe fidelity sweep, relative to full-resolution MiSe-GNN. Best is the level with the lowest macro RRMSE; fastest improving is the fastest level that still improves macro RRMSE.

| Dataset | Full macro | Best MuFi | | | | Fastest improving | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Level | RRMSE | Δ | Speedup | Level | RRMSE | Δ | Speedup |
| `2D_Profile` | 0.03385 | L2 | 0.02600 | −23.2% | 3.66× | L3 | 0.02893 | −14.5% | 5.82× |
| `VKI-LS59` | 0.08690 | L2 | 0.03202 | −63.2% | 2.76× | L4 | 0.04165 | −52.1% | 6.97× |
| `Tensile2D` | 0.00402 | L1 | 0.00292 | −27.4% | 2.13× | L3 | 0.00331 | −17.8% | 3.47× |
| `2D_MultiScHypEl` | 0.02894 | L1 | 0.03022 | +4.4% | 1.90× | | | – | |

Table 4: Localized and integrated physics diagnostics: relative change (%) of MuFi-$B$ with respect to the full-resolution backbone $B$ at the levels of Table 2. Negative is better; degradations in bold. †Mean over three seeds. Absolute values are in Appendix E.

| Backbone | `2D_Profile` | | | `VKI-LS59` | `Tensile2D` | `2D_MultiScHypEl` |
|---|---|---|---|---|---|---|
| | $E^{\text{Mach}}_{\text{HG},5\%}$ | $E^{p}_{\Gamma}$ | $E_{F_p}$ | $E^{\text{Mach}}_{\text{HG},5\%}$ | $E^{\sigma_{\text{vm}}}_{\text{top},5\%}$ | $E^{sF}_{\text{near-hole}}$ |
| MGN | −10.4 | −65.6 | −82.4 | −37.8 | −95.0 | −1.1 |
| MGN-Tree | −12.3 | −32.0 | −49.0 | −12.7 | **+18.0** | **+36.2**† |
| MiSe-GNN | −19.6 | −20.3 | −27.5 | −12.1 | −25.9 | **+68.0** |
| Transolver | **+306.4** | −1.9 | **+4.9** | −4.7 | −23.6 | −45.3 |

**Localized physics diagnostics:** Macro RRMSE measures global regression quality but can hide localized errors. On the original mesh, we therefore evaluate the relative $L_2$ error on the 5% of nodes with the highest reference gradient ($E_{\text{HG},5\%}$), the airfoil surface-pressure error ($E^{p}_{\Gamma}$), the discrepancy of the integrated pressure force ($E_{F_p}$), the error on the 5% highest von Mises stresses (`Tensile2D`), and the first-Piola stress-intensity error near holes (`2D_MultiScHypEl`); definitions and absolute values are given in Appendix E. Table 4 reveals three regimes. (i) When the coarse level still resolves the relevant structures, local and integrated quantities improve together with the macro error: all four backbones reduce the high-gradient Mach error on `VKI-LS59`, and the three GNN wrappers improve every `2D_Profile` diagnostic, including a 27-82% lower force discrepancy. (ii) For the global-attention Transolver on `2D_Profile`, macro RRMSE improves while the shock error rises from 0.23 to 0.92: a likely explanation is that coarsening removes resolution without the communication benefit that local message passing gains, so a Mach-critical application should select a finer level or another backbone. (iii) Local stress maxima are the most fidelity-sensitive quantity: near-hole stress

errors on `2D_MultiScHypEl` increase for MGN-Tree and MiSe-GNN, and the top-5% von Mises error on `Tensile2D` increases for MGN-Tree, even when macro RRMSE improves. As with mesh density in classical numerics, the fidelity level should therefore be chosen for the target quantity.

### 4.5 Varying-geometry 3D surfaces and seed variability

`Rotor37` tests whether the interface carries over to 3D surface meshes whose geometry varies across samples; the normal-aware alignment is built once per geometry (Appendix A). At Level 1, both backbones improve accuracy while training faster (Table 5): MGN from $0.00778$ to $0.00579$ with $1.68\times$ faster epochs and $1.86\times$ faster inference, and Transolver from $0.00683$ to $0.00575$ with $1.45\times$ faster epochs. Over three seeds, these gains are large relative to the seed variability, and deeper levels provide lower-cost operating points that remain at least as accurate as the baseline (Transolver L3: $0.00671 \pm 0.00001$ versus $0.00677 \pm 0.00008$ with $2.2\times$ faster epochs). In 2D, the smallest improvement in Table 2 (MGN-Tree on `2D_MultiScHypEl`, $-0.7\%$) is confirmed over three seeds with a lower standard deviation ($0.0324 \pm 0.0019 \rightarrow 0.0303 \pm 0.0006$, Appendix D).

Table 5: `Rotor37` varying-geometry 3D surface results. Propagation nodes are the mean number of nodes seen by the backbone. Three-seed values are mean $\pm$ standard deviation over seeds 42, 43, and 44; inference uses cached preprocessing on one A100 GPU. "–" marks settings that were not run.

| Model | Level | Propagation Nodes | Macro RRMSE | Macro RRMSE (3 seeds) | Epoch Time (s) | Inference (ms) |
|---|---|---|---|---|---|---|
| MGN | Full | 29,773 | 0.00778 | $0.00779 \pm 0.00004$ | 174.27 | 80.06 |
| MuFi-MGN | L1 | 14,941 | **0.00579** | $\mathbf{0.00597 \pm 0.00028}$ | 103.76 | 43.01 |
| MuFi-MGN | L2 | 10,491 | 0.00644 | $0.00659 \pm 0.00019$ | 78.70 | 32.28 |
| MuFi-MGN | L3 | 6,039 | 0.00738 | – | **59.72** | **24.25** |
| Transolver | Full | 29,773 | 0.00683 | $0.00677 \pm 0.00008$ | 74.56 | – |
| MuFi-Transolver | L1 | 14,941 | **0.00575** | $\mathbf{0.00583 \pm 0.00017}$ | 51.28 | – |
| MuFi-Transolver | L2 | 10,491 | 0.00681 | – | 48.64 | – |
| MuFi-Transolver | L3 | 6,039 | 0.00670 | $0.00671 \pm 0.00001$ | **33.36** | – |

### 4.6 Training and inference efficiency

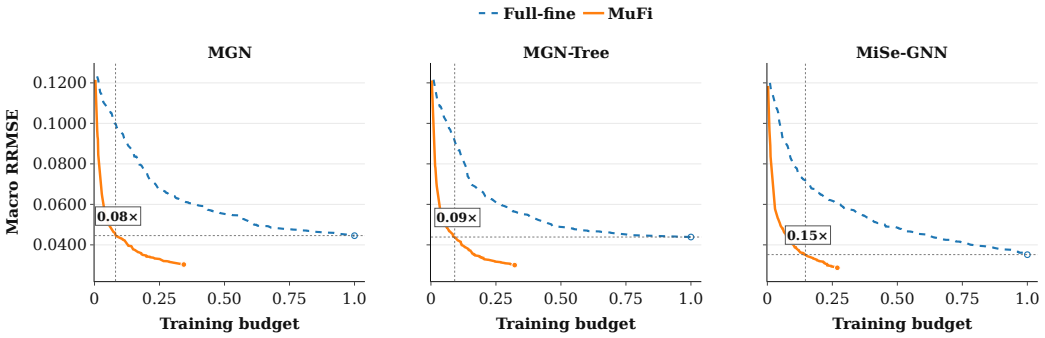



Figure 3: Budget-limited training on `2D_Profile` at coarse Level 2. For each backbone, the $x$-axis is normalized by the total training budget of the corresponding full-resolution model, and the $y$-axis reports absolute macro RRMSE. The horizontal dotted line marks the final full-resolution error, while the vertical dotted line marks the first budget at which MuFi matches that error.

**Time to accuracy:** Epoch time alone does not show when a model becomes useful. Normalizing training time by the budget of the corresponding full-resolution backbone on identical hardware, MuFi reaches the final full-resolution error on `2D_Profile` after $0.08\times$, $0.09\times$, and $0.15\times$ of the MGN, MGN-Tree, and MiSe-GNN budgets (Figure 3).

**Inference and preprocessing:** With cached preprocessing, MuFi-MiSe-GNN predicts a `2D_Profile` sample in 0.375 s versus 0.834 s for full-resolution MiSe-GNN ($2.23\times$ at Level

3) with 41% lower peak GPU memory, while the elsA solver takes 216.9 s on 512 CPU cores. Coarsening, target transfer, and alignment are computed once per geometry: coarsening takes 0.02–0.84 s per sample and 2D alignment 0.89–7.21 s with our reference Python implementation (the vectorized 3D surface implementation takes 0.19 s), and the cached data occupy 0.2-2.4 MB per sample. On `2D_Profile`, this one-time cost amortizes after about 13 queries of the same geometry at inference and after about 11 of the 1000 training epochs (Appendix G).

### 4.7 Ablations

Table 6 isolates the components specific to MuFi. FiLM conditioning is the key decoder choice: it lowers the error by 44% relative to the best alternative (geo-attention) at half its epoch time, whereas concatenation-based neural fields, cross-attention, and SIREN (Sitzmann et al., 2020) perform on par with or worse than the non-learned projection. Predicting a residual over the projected coarse field helps weaker decoders but not FiLM, which reconstructs the full field directly. Robust alignment reduces the error by 9% and the fraction of extrapolative stencils from 48% to 15%, and $K = 3$ outperforms $K = 5$, indicating that a minimal non-degenerate stencil is preferable to a larger but smoother neighborhood. Removing the coarse loss increases the error by 33%.

Table 6: Ablation summary with the MiSe-GNN backbone at Level 2. Values are `2D_Profile` Temperature RRMSE unless noted; lower is better. Full results are in Appendix F.

| **Study** | **Comparison** | **Result** |
|---|---|---|
| Decoder | FiLM vs. geo-attention / concatenation / SIREN | **0.0157** vs. 0.0282 / 0.0296 / 0.0840 |
| Decoder target | direct vs. residual over the projected coarse field | **0.0157** vs. 0.0180 |
| Alignment | robust vs. Euclidean $k$NN | **0.0157** vs. 0.0173 |
| Stencil quality | robust vs. Euclidean $k$NN, extrapolation rate | **0.147** vs. 0.480 |
| Coarse loss | $\mathcal{L}_f + \mathcal{L}_c$ vs. $\mathcal{L}_f$ only | **0.0157** vs. 0.0210 |
| Stencil size | $K = 3$ vs. $K = 5$, `2D_MultiScHypEl` $u_1$ / $u_2$ | **0.0106 / 0.0112** vs. 0.0113 / 0.0131 |

## 5 Conclusion and Limitations

We studied what information to transfer across fidelity levels in mesh PDE surrogates. MuFi runs an unchanged backbone on a coarsened mesh, transfers its pre-readout hidden states through a cached geometry-aware operator, and decodes the field on the original mesh with a FiLM-conditioned neural field. In a matched control, transferring hidden states instead of coarse predictions lowers the error by 18.7% at nearly identical parameter count and cost, and across four backbones on four 2D benchmarks and two backbones on a 3D surface benchmark, the fidelity level exposes explicit accuracy-cost operating points.

**Limitations:** MuFi is a data-driven regression wrapper: neither the alignment nor the decoder enforces conservation laws or stress equilibrium, and our diagnostics cover the quantities available in the benchmarks rather than full PDE residuals. Sharply localized quantities degrade when the coarse spacing does not resolve the feature scale, as for shocks with the global-attention backbone and for stress concentrations near holes, so the level must be chosen for the target quantity. Performance depends on the mesh hierarchy and alignment quality; our 3D evaluation covers surface meshes, and volumetric meshes remain to be tested. Each new geometry requires one-time preprocessing, and the interface assumes a spatially indexed backbone state. Predictions should be validated before use in safety-critical engineering workflows.

**Future work:** Task-aware or spatially adaptive fidelity selection, multi-level state fusion, conservation-preserving transfer operators and objectives, and adapters that connect grid-based PDE foundation models (Herde et al., 2024; McCabe et al., 2024) to irregular meshes through the same state interface are natural extensions.

## AI USE STATEMENT

LLMs were used to polish the writing of this paper and to help find related work. The authors thoroughly evaluated all suggestions against the study's methodology and findings, assuming full accountability for the final text. Crucially, generative AI was not utilized to produce, analyze, or synthesize any results presented herein.

## ETHICS STATEMENT

All experiments in this study rely exclusively on publicly available simulation data, involving neither human participants nor personally identifiable information. Furthermore, any surrogate model predictions applied in engineering design must be validated against full numerical solvers prior to making safety-critical decisions.

## REPRODUCIBILITY STATEMENT

Appendix A specifies the alignment algorithm and its hyperparameters; Appendix B describes the datasets, splits, mesh hierarchies, backbone and decoder hyperparameters, training protocol, and metric; Appendices C-F report per-field results, fidelity sweeps, physics diagnostics, and ablations; and Appendix G reports preprocessing, storage, and inference measurements. All datasets are publicly available through PLAID (Casenave et al., 2026). Anonymized code for preprocessing, training, and evaluation is provided in the supplementary material.

## REFERENCES


Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal physics transformers: A framework for efficiently scaling neural operators. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications, 2021. URL https://arxiv.org/abs/2006.05205.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

Hardeep Bassi, Yuanran Zhu, Erika Ye, Pu Ren, Alec Dektor, Michael W. Mahoney, Harish S. Bhat, and Chao Yang. RELift: Learned coarse-to-fine propagators for time-dependent PDEs with applications to electron dynamics, 2026. URL https://arxiv.org/abs/2509.12220.

Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

Felipe Bordeu, Maya de Buhan, Julien Cortial, Fabien Casenave, and Ramzi Messahel. Muscat : Mesh manipulation and finite element engine for engineering and science. In *16ème Colloque National en Calcul de Structures (CSMA 2024)*, Hyères, France, May 2024. CNRS and CSMA and ENS Paris-Saclay and CentraleSupélec. URL https://hal.science/hal-04823002.

Johannes Brandstetter, Daniel Worrall, and Max Welling. Message passing neural PDE solvers, 2023. URL https://arxiv.org/abs/2202.03376.

Yadi Cao, Menglei Chai, Minchen Li, and Chenfanfu Jiang. Efficient learning of mesh-based physical simulation with bi-stride multi-scale graph neural network. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 3541–3558, 2023.

Fabien Casenave, Brian Staber, and Xavier Roynard. MMGP: a Mesh Morphing Gaussian Process-based machine learning method for regression of physical problems under nonparametrized geometrical variability. *Advances in Neural Information Processing Systems*, 36, 2024.

Fabien Casenave, Xavier Roynard, Brian Staber, Alexandre Devaux-Rivière, William Piat, Michele Alessandro Bucci, Nissrine Akkari, Abbas Kabalan, Xuan Minh Vuong Nguyen, Luca Saverio, Raphaël Carpintero Perez, Anthony Kalaydjian, Samy Fouché, Thierry Gonon, Ghassan Najjar, Thomas Daniel, Emmanuel Menier, Matthieu Nastorg, Giovanni Catalani, and Christian Rey. Plaid: A unified data model for machine learning on heterogeneous physics simulations, 2026. URL https://arxiv.org/abs/2505.02974.

Giovanni Catalani, Jean Fesquet, Xavier Bertrand, Frédéric Tost, Michael Bauerheim, and Joseph Morlier. Towards scalable surrogate models based on neural fields for large scale aerodynamic simulations, 2025. URL https://arxiv.org/abs/2505.14704.

Cecile Dobrzynski. *MMG3D: User guide*. PhD thesis, Inria, 2012.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL https://arxiv.org/abs/2010.11929.

Meire Fortunato, Tobias Pfaff, Peter Wirnsberger, Alexander Pritzel, and Peter Battaglia. Multiscale MeshGraphNets, 2022. URL https://arxiv.org/abs/2210.00612.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry, 2017. URL https://arxiv.org/abs/1704.01212.

Junyi Guo, Pan Du, Xiantao Fan, Yahui Li, and Jian-Xun Wang. Conditional neural field for spatial dimension reduction of turbulence data: a comparison study, 2025. URL https://arxiv.org/abs/2510.25135.

Jan Hagnberger, Marimuthu Kalimuthu, Daniel Musekamp, and Mathias Niepert. Vectorized conditional neural fields: A framework for solving time-dependent parametric partial differential equations, 2024. URL https://arxiv.org/abs/2406.03919.

Maximilian Herde, Bogdan Raonić, Tobias Rohner, Roger Käppeli, Roberto Molinaro, Emmanuel de Bézenac, and Siddhartha Mishra. Poseidon: Efficient foundation models for PDEs. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Minju Jo, Woojin Cho, Uvini Balasuriya Mudiyanselage, Seungjun Lee, Noseong Park, and Kookjin Lee. PDEfuncta: Spectrally-aware neural representation for PDE solution modeling, 2025. URL https://arxiv.org/abs/2506.12790.

Bharti Khemani, Shruti Patil, Ketan Kotecha, and Sudeep Tanwar. A review of graph neural networks: concepts, architectures, techniques, challenges, datasets, applications, and future directions. *Journal of Big Data*, 11(1):18, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Zong-Yi Li, D. Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for PDEs on general geometries. *J. Mach. Learn. Res.*, 24:388:1–388:26, 2022.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations, 2020. URL https://arxiv.org/abs/2006.09535.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2021. URL https://arxiv.org/abs/2010.08895.

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, 2023.

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.

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, 2025. URL https://arxiv.org/abs/2502.02414.

Michael McCabe, Bruno Régaldo-Saint Blancard, Liam Parker, Ruben Ohana, Miles Cranmer, Alberto Bietti, Michael Eickenberg, Siavash Golkar, Geraud Krawezik, Francois Lanusse, Mariel Pettee, Tiberiu Tesileanu, Kyunghyun Cho, and Shirley Ho. Multiple physics pretraining for spatiotemporal surrogate models. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Xuan Minh Vuong Nguyen, Nissrine Akkari, Fabien Casenave, Jonathan Viquerat, and Elie Hachem. Error-driven graph augmentation for mesh-based PDE surrogates. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer, 2017. URL https://arxiv.org/abs/1709.07871.

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.

Winfried Ripken, Lisa Coiffard, Felix Pieper, and Sebastian Dziadzio. Multiscale neural operators for solving time-independent PDEs, 2023. URL https://arxiv.org/abs/2311.05964.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter W. Battaglia. Learning to simulate complex physics with graph networks, 2020. URL https://arxiv.org/abs/2002.09405.

Louis Serrano, Lise Le Boudec, Armand Kassaï Koupaï, Thomas X. Wang, Yuan Yin, Jean-Noël Vittaut, and Patrick Gallinari. Operator learning with neural fields: Tackling PDEs on general geometries. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Louis Serrano, Thomas X Wang, Etienne Le Naour, Jean-Noël Vittaut, and Patrick Gallinari. AROMA: Preserving spatial structure for latent PDE modeling with local neural fields, 2024. URL https://arxiv.org/abs/2406.02176.

Donald Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 23rd ACM national conference*, pp. 517–524, 1968.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7462–7473. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf.

Junshu Sun, Chenxue Yang, Xiangyang Ji, Qingming Huang, and Shuhui Wang. Towards dynamic message passing on graphs, 2024. URL https://arxiv.org/abs/2410.23686.

Kiwon Um, Robert Brand, Yun Fei, Philipp Holl, and Nils Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE-solvers. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

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.

Yingxue Zhao, Haoran Li, Haosu Zhou, Hamid Reza Attar, Tobias Pfaff, and Nan Li. A review of graph neural network applications in mechanics-related domains, 2024. URL https://arxiv.org/abs/2407.11060.

Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications, 2021. URL https://arxiv.org/abs/1812.08434.

## A ROBUST ALIGNMENT ALGORITHMS

MUFI transfers node-wise coarse states to the original fine mesh through a precomputed sparse alignment matrix. Let $Z_c \in \mathbb{R}^{|V_c| \times d_z}$ be the coarse states produced by the wrapped backbone. The aligned fine states are:

$$Z_f = M Z_c, \qquad Z_f \in \mathbb{R}^{|V_f| \times d_z} \tag{6}$$

where each row of $M$ contains only $K$ nonzero alignment weights. The matrix $M$ depends only on the fine/coarse mesh pair and is computed once offline before training.

A naive Euclidean kNN alignment chooses the $K$ nearest coarse nodes to each fine query and applies inverse-distance weights. This rule is efficient, but it uses distance as the only criterion. Near boundaries and curved geometric features, the nearest nodes can be a poor alignment support: an interior fine node may receive coarse boundary states, the selected nodes may be nearly collinear, or the fine query may lie outside the triangle formed by the selected coarse nodes. These cases produce boundary leakage or extrapolative aligned states. Since the fine decoder is conditioned on the aligned state at every fine node, local alignment errors can be amplified in the reconstructed field.

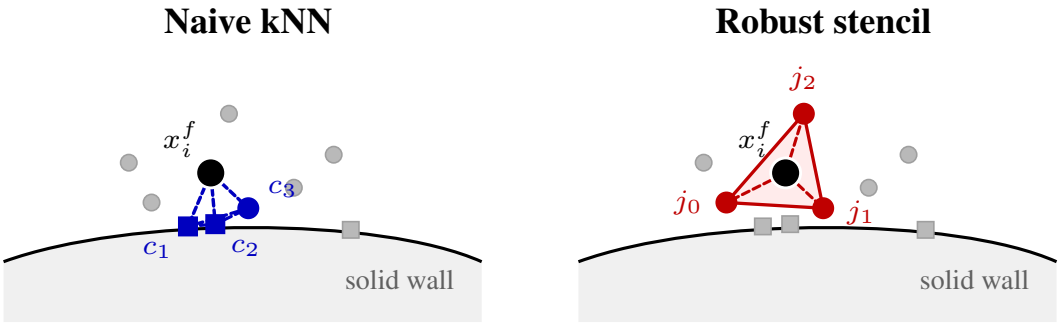



Figure 4: Schematic of coarse-to-fine support selection near a curved boundary. Both panels show the same fine query $x_i^f$ and the same coarse candidate pool, squares denote coarse boundary candidates and gray circles denote interior coarse candidates. The naive kNN approach uses only Euclidean distance and selects the three nearest candidates $(c_1, c_2, c_3)$, which can form a low-area, nearly collinear, and extrapolative support for an interior fine query. In contrast, the robust rule selects a better conditioned stencil $(j_0, j_1, j_2)$ by balancing proximity, boundary consistency, separation, and non-degenerate area, before applying fixed inverse-distance weights on the selected support.

For each fine query node $x_i^f$, we first retrieve a candidate set $\mathcal{C}_i$ of $K_{\text{cand}}$ nearest coarse nodes using a Euclidean KD-tree (Bentley, 1975). In all reported 2D experiments we use $K_{\text{cand}} = 30$. Let $h$ be the median nearest-neighbor spacing on the coarse mesh:

$$h = \text{median}_{j \in V_c} \min_{q \in V_c \setminus \{j\}} \|x_j^c - x_q^c\|_2 \tag{7}$$

We also compute binary boundary masks $b_i^f$ and $b_j^c$ from mesh connectivity. If boundary information is unavailable, all nodes are treated as interior and the boundary penalty is disabled.

For a boundary fine query, the implementation first attempts to select the nearest $K$ coarse boundary candidates. This preserves boundary-to-boundary alignment when enough boundary candidates exist. If fewer than $K$ coarse boundary candidates are found in $\mathcal{C}_i$, the algorithm falls back to the generic robust rule described below.

For an interior fine query, coarse boundary candidates are discouraged through

$$\rho_b(i, j) = \mathbf{1}\{b_i^f = 0 \text{ and } b_j^c = 1\} \tag{8}$$

The scores below are the $h$-scaled selection costs used to build the saved alignment matrices. The coefficients are fixed hyperparameters: $\lambda_b = 2$, $\lambda_{\text{sep}} = 1$, $\lambda_{\text{area}} = 1$, and $\lambda_{\text{ext}} = 4$.

The default stencil size is $K = 3$. The first anchor balances distance and boundary consistency:

$$j_0 = \arg\min_{j \in \mathcal{C}_i} \|x_i^f - x_j^c\|_2 + \lambda_b h \, \rho_b(i, j) \tag{9}$$

The second anchor is selected to remain close to the query while avoiding a nearly duplicated support point:

$$j_1 = \arg\min_{j \in \mathcal{C}_i \setminus \{j_0\}} \|x_i^f - x_j^c\|_2 + \lambda_{\text{sep}} \frac{h}{\|x_j^c - x_{j_0}^c\|_2 + \epsilon} + \lambda_b h\, \rho_b(i,j) \tag{10}$$

The third anchor is chosen to avoid low-area and extrapolative triangles. Let

$$A_2(j_0, j_1, j) = \left|\det\left[x_j^c - x_{j_0}^c \quad x_j^c - x_{j_1}^c\right]\right| \tag{11}$$

denote twice the area of the triangle formed by the three coarse candidates. Let $\Delta(j_0, j_1, j)$ be this triangle. The third anchor is

$$\begin{aligned} j_2 = \arg\min_{j \in \mathcal{C}_i \setminus \{j_0, j_1\}} & \|x_i^f - x_j^c\|_2 + \lambda_{\text{area}} \frac{h^2}{A_2(j_0, j_1, j) + \epsilon} \\ & + \lambda_{\text{ext}} h\, \mathbf{1}\{x_i^f \notin \Delta(j_0, j_1, j)\} + \lambda_b h\, \rho_b(i,j). \end{aligned} \tag{12}$$

The main model uses $K = 3$. In the $K = 5$ ablation, we do not construct a new five-point geometric simplex. Instead, the implementation keeps the robust triad $\{j_0, j_1, j_2\}$ as the geometric core and adds the nearest unused candidates from the original candidate pool:

$$j_m = \arg\min_{j \in \mathcal{C}_i \setminus \{j_0, \ldots, j_{m-1}\}} \|x_i^f - x_j^c\|_2, \qquad m = 3, 4 \tag{13}$$

Thus the $K = 5$ experiment should be interpreted as testing whether two additional inverse-distance support points improve the already selected robust triad. It is not a higher order alignment rule and does not change the core robust stencil principle.

After the support set

$$\mathcal{N}_K(i) = \{j_0, \ldots, j_{K-1}\} \tag{14}$$

has been selected, the reported experiments use fixed inverse-distance weights:

$$w_{ij} = \frac{\left(\|x_i^f - x_j^c\|_2 + \epsilon\right)^{-1}}{\sum_{k \in \mathcal{N}_K(i)} \left(\|x_i^f - x_k^c\|_2 + \epsilon\right)^{-1}}, \qquad j \in \mathcal{N}_K(i) \tag{15}$$

The aligned state at the fine node is

$$z_i^f = \sum_{j \in \mathcal{N}_K(i)} w_{ij} z_j^c \tag{16}$$

Equivalently, the sparse alignment matrix is

$$M_{ij} = \begin{cases} w_{ij}, & j \in \mathcal{N}_K(i) \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

Each row of $M$ is nonnegative and sums to one. Since the matrix is fixed by the mesh hierarchy, it adds no learnable parameters and is reused across all epochs.

The robust rule changes only the support selection step. The final alignment weights remain simple inverse-distance weights. This isolates the benefit of geometry-aware support selection: compared with naive kNN, the selected stencil is less likely to mix boundary and interior states, less likely to be nearly collinear, and less likely to place the fine query outside the selected coarse support. These properties are important for MUFI because the fine decoder receives the aligned coarse state as conditioning information at every fine node.

The 2D benchmarks use the triangle-based robust rule above, whereas `Rotor37` uses the normal-aware surface alignment described below. Extension to volumetric 3D meshes remains future work.

$$\mathcal{N}_K(i) = \arg\min_{j \in \mathcal{C}_i} \left[ \frac{\|x_i^f - x_j^c\|_2}{h} + \lambda_n (1 - n_i^f \cdot n_j^c) + \lambda_o \frac{|(x_j^c - x_i^f) \cdot n_i^f|}{h} \right] \tag{18}$$

$$w_{ij} = \frac{d_{ij}^{-p} \left[ \mathrm{clip}\left( \frac{1+n_i^f \cdot n_j^c}{2}, \epsilon, 1 \right) \right]^\gamma}{\sum_{l \in \mathcal{N}_K(i)} d_{il}^{-p} \left[ \mathrm{clip}\left( \frac{1+n_i^f \cdot n_l^c}{2}, \epsilon, 1 \right) \right]^\gamma}, \quad d_{ij} = \frac{\|x_i^f - x_j^c\|_2}{h} \tag{19}$$

For each fine node, we retrieve a local candidate set $\mathcal{C}_i$ of $K_{\mathrm{cand}} = 64$ nearest coarse nodes using a KD-tree, rank the candidates using a geometry-aware score combining spatial distance, normal consistency, and normal-direction offset, and construct normalized inverse-distance weights modulated by normal similarity. For `Rotor37`, we retain $K = 3$ neighbors and set $\lambda_n = 5.0$, $\lambda_o = 0.25$, $p = 1.0$, and $\gamma = 3.0$. Vertex normals are computed from surface mesh geometry via area weighted face normals, sign aligned with reference fine mesh orientations, and the alignment operator is computed once per sample geometry and cached.

# B Datasets, mesh hierarchies, and training protocol

## B.1 Datasets and splits

We evaluate MuFi on five PLAID benchmarks (see Table 7 for detailed statistics and splits) covering CFD and CSD: `2D_Profile`, `VKI-LS59`, `Tensile2D`, `2D_MultiScHypEl`, and `Rotor37`. The datasets differ in mesh size, output dimensionality, and field regularity, allowing us to test both smooth flow quantities and stress-dominated mechanics fields. For each dataset, we construct a hierarchy of coarsened meshes and run MuFi at selected fidelity levels.

Table 7: Dataset statistics and train/test splits. Node counts are averaged over the dataset; * denotes fixed connectivity with varying node coordinates. The `VKI-LS59` quadrilateral mesh is converted to a triangular mesh.

| Dataset | Mesh (mean nodes) | Inputs | Outputs | Splits (train/test) |
|---|---|---|---|---|
| `2D_MultiScHypEl` | tri (5,692) | mesh, 3 scalars | 7 fields | 764 / 376 |
| `2D_Profile` | tri (37,042) | mesh | 5 fields | 300 / 100 |
| `VKI-LS59` | tri (36,421*) | mesh, 2 scalars | 2 fields | 671 / 168 |
| `Tensile2D` | tri (9,428) | mesh, 6 scalars | 5 fields | 500 / 200 |
| `Rotor37` | tri (29,773*) | mesh, 2 scalars | 3 fields | 1,000/200 |

Table 8: Mesh hierarchy used in the experiments. Entries report the mean number of nodes at the original resolution and after remeshing. "–" indicates that the corresponding fidelity level is not used.

| Mesh | `2D_Profile` | `VKI-LS59` | `Tensile2D` | `2D_MultiScHypEl` | `Rotor37` |
|---|---|---|---|---|---|
| Original | tri (37,042) | tri (36,421) | tri (9,428) | tri (5,692) | tri (29,773) |
| Level 1 (L1) | tri (17,557) | tri (20,759) | tri (3,650) | tri (2,931) | tri (14,941) |
| Level 2 (L2) | tri (8,811) | tri (10,786) | tri (1,044) | tri (1,297) | tri (10,491) |
| Level 3 (L3) | tri (4,489) | tri (4,265) | tri (516) | — | tri (6,039) |
| Level 4 (L4) | — | tri (1,060) | — | — | — |

Table 8 summarizes the mesh hierarchy used by MuFi. The coarse levels determine where the backbone performs propagation, but all reported errors are evaluated on the original fine mesh.

## B.2 Coarsening levels

For each dataset, we construct a hierarchy of coarse meshes using MMG (Dobrzynski, 2012) through the Muscat (Bordeu et al., 2024) interface. The hierarchy is generated offline and reused for all

backbones. Each fidelity level defines a coarse graph $G_c$ on which the backbone performs physical propagation, together with a transferred coarse target field $Y_c$ used for auxiliary supervision. The alignment matrix $M$ from the coarse mesh to the original fine mesh is also precomputed at this stage.

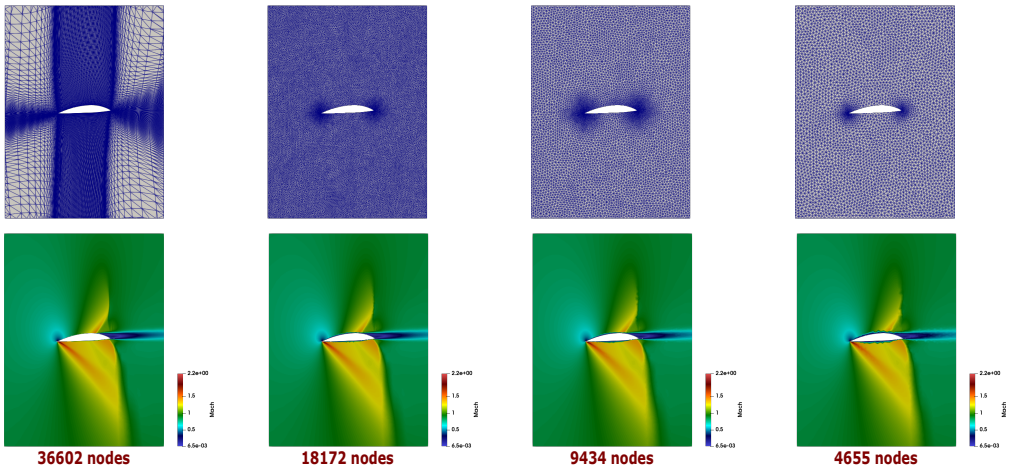



Figure 5: `2D_Profile` mesh hierarchy. Top row: representative meshes from the original resolution to progressively coarser levels. Bottom row: the corresponding transferred field used for coarse-level supervision. Node counts are shown below each panel.

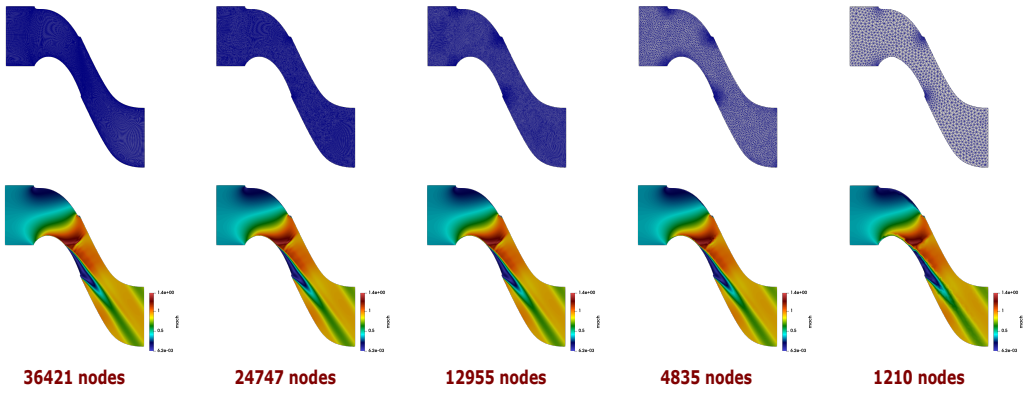



Figure 6: `VKI-LS59` mesh hierarchy. Representative meshes and transferred fields across the original and coarse fidelity levels. This dataset supports the deepest hierarchy used in our experiments, up to Level 4.

The selected levels trade off geometric resolution and propagation cost. Mild coarsening preserves more local detail, while deeper levels reduce the number of nodes and edges seen by the backbone. We omit fidelity levels that produce unstable mesh quality or do not provide a meaningful computational reduction for a given dataset. Representative mesh hierarchies and transferred fields are shown in Figures 5, 6, 7, 8, 9.

### B.3 BACKBONES

We evaluate MUFI as a wrapper around four mesh surrogate backbones: MeshGraphNets (MGN), MGN-Tree, MiSe-GNN, and Transolver. For each backbone $B$, the corresponding MUFI-$B$ variant keeps the backbone's internal propagation mechanism unchanged, but applies it on the selected coarse mesh $G_c$ rather than on the original fine mesh. The only interface requirement is that the backbone exposes node-wise coarse representation states $Z_c$ before the final prediction layer. These states are passed to the shared MUFI alignment and FiLM decoder.

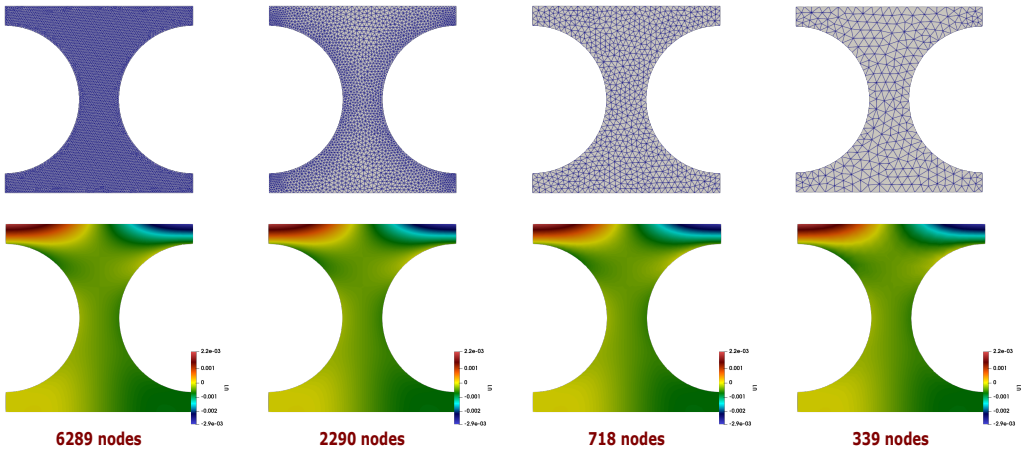



Figure 7: `Tensile2D` mesh hierarchy. Representative meshes and transferred mechanics field across fidelity levels. Coarsening reduces the propagation graph substantially while preserving the global specimen geometry.

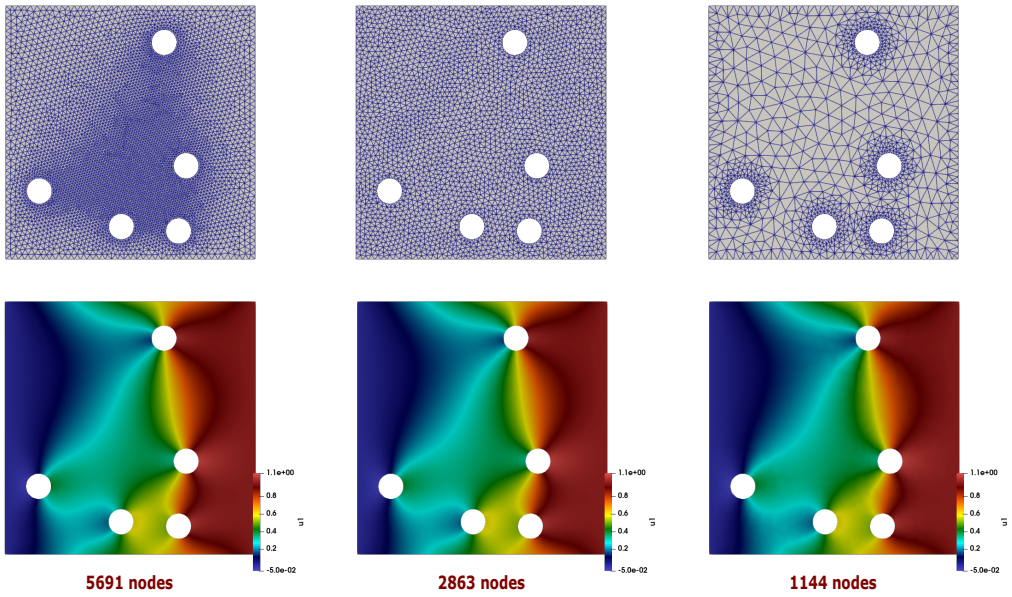



Figure 8: `2D_MultiScHypEl` mesh hierarchy. Representative meshes and transferred hyperelastic field across the fidelity levels used in our experiments. The holes and thin regions make this dataset useful for testing the robustness of coarse-to-fine alignment.

Table 9: Backbone and training configurations used in the controlled comparisons. MuFi variants use the same backbone configuration as their full-resolution counterparts.

| | Architecture Hyperparameters | | | | | | Training Hyperparameters | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Backbone** | **Hidden Dim** | **Depth** | **Activation** | **MLP** | **Aggregation** | **Attention Params** | **Optimizer** | **LR** | **Batch** | **Epochs** |
| MGN | 80 | 15 steps | Leaky ReLU | 2 layers | Sum | - | Adam | $10^{-3}$ | 1 | 1000 |
| MGN-Tree | 80 | 15 steps | Leaky ReLU | 2 layers | Sum | - | Adam | $10^{-3}$ | 1 | 1000 |
| MiSe-GNN | 80 | 15 steps | Leaky ReLU | 2 layers | Sum | - | Adam | $10^{-3}$ | 1 | 1000 |
| Transolver | 128 | 4 blocks | GELU | ratio 4 | - | 4 heads, 16 slices | Adam | $10^{-3}$ | 1 | 1000 |

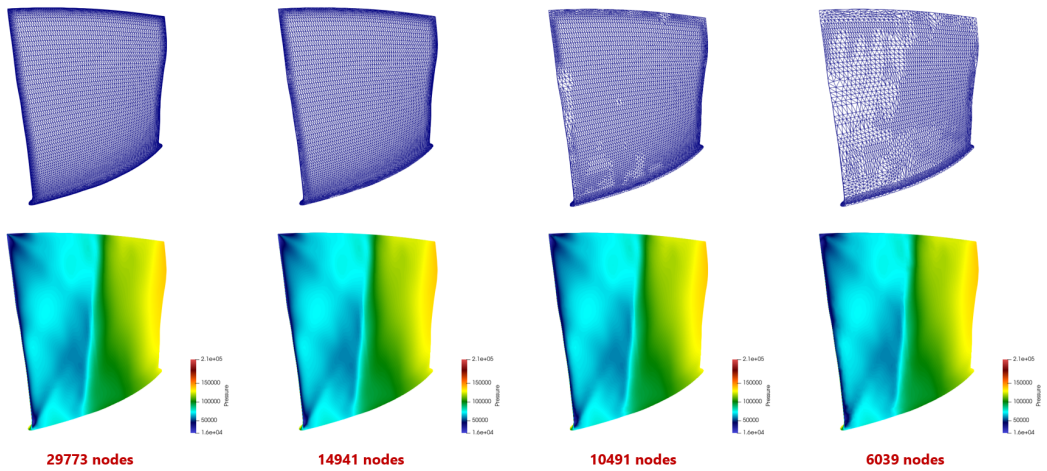



Figure 9: `Rotor37` mesh hierarchy. Representative meshes and transferred Pressure field across fidelity levels.

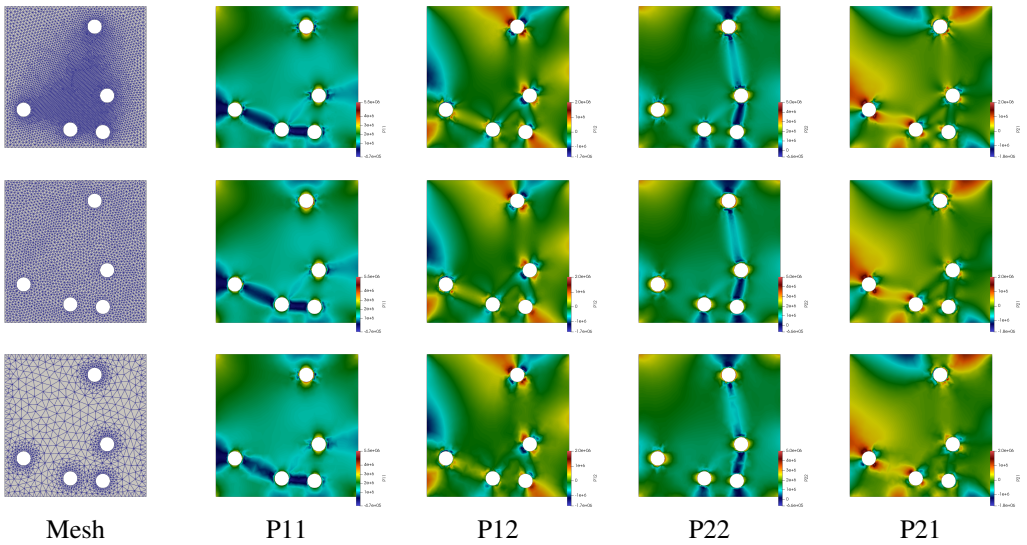



Figure 10: Stress-component visualization for `2D_MultiScHypEl` across the mesh hierarchy. Rows show the original mesh, Level 1, and Level 2 coarsened meshes; the first column shows the corresponding mesh and the remaining columns show reference stress components $P_{11}$, $P_{12}$, $P_{22}$, and $P_{21}$ transferred to each mesh level for visualization. These stress components contain narrow high-gradient bands and localized concentrations around holes and thin ligaments. Such structures are represented by relatively few elements even on the original mesh and are progressively smoothed or aliased after coarsening. This is consistent with the field-wise results in Table 18: MuFi improves the smoother displacement fields $u_1, u_2$, but stress components are more sensitive to loss of fine-scale resolution.

### B.4 Training protocol

All models are trained with the same train/test split for each dataset. Unless otherwise stated, we use the Adam optimizer (Kingma & Ba, 2017) with learning rate `1e-3`, batch size `1`, and train for `1000` epochs. Epoch time is measured on a single `NVIDIA A100 GPU` for all datasets, except for the `Tensile2D` dataset, which is trained on a single `NVIDIA A30 GPU`. This timing excludes offline preprocessing, as coarsening, field transfer, and alignment construction are performed once and cached prior to training.

Table 10: FiLM-conditioned fine decoder hyperparameters.

| Hyperparameter | Value |
|---|---|
| Hidden dimension | 256 |
| Number of modulated layers $L$ | 3 |
| Hypernetwork hidden dimension | 256 |
| Activation | GELU |
| Normalization | LayerNorm |

For each controlled comparison, MUFI-$B$ and the corresponding full-resolution backbone $B$ use the same backbone hidden dimension, processor depth, optimizer, and training schedule. The additional MUFI parameters are restricted to the FiLM-conditioned fine decoder.

### B.5 METRIC

In alignment with the evaluation protocol established by the PLAID benchmark Casenave et al. (2026), we assess the accuracy of our mesh field predictions using the Relative Root Mean Square Error (RRMSE). This metric ensures fair comparisons across datasets characterized by varying scales and node counts by normalizing the error against both the number of mesh nodes and the maximum amplitude of the ground truth. Specifically, the field RRMSE is computed as follows:

$$\mathrm{RRMSE}(\mathbf{f}_{\mathrm{ref}}, \mathbf{f}_{\mathrm{pred}}) = \left( \frac{1}{n_\star} \sum_{i=1}^{n_\star} \frac{\|\mathbf{f}_{\mathrm{ref}}^i - \mathbf{f}_{\mathrm{pred}}^i\|_2^2}{N^i \|\mathbf{f}_{\mathrm{ref}}^i\|_\infty^2} \right)^{1/2} \tag{20}$$

where $n_\star$ represents the total number of test samples, $N^i$ denotes the node count for the $i$-th sample, and $\|\mathbf{f}_{\mathrm{ref}}^i\|_\infty$ is the peak absolute value of the reference field $\mathbf{f}_{\mathrm{ref}}^i$.

For multi output datasets, we report both field-wise RRMSE and macro RRMSE. The macro score is the unweighted average over output fields:

$$\mathrm{MacroRRMSE} = \frac{1}{d_y} \sum_{m=1}^{d_y} \mathrm{RRMSE}(\mathbf{f}_{\mathrm{ref}}^{(m)}, \mathbf{f}_{\mathrm{pred}}^{(m)}) \tag{21}$$

This metric gives equal weight to each physical quantity, independent of its numerical scale.

## C FULL CONTROLLED RESULTS

Tables 11, 12, 13, 14 report the full per-field cross-backbone results corresponding to the summary in Table 2. Each MUFI-$B$ row uses the same fidelity level as in the main comparison for that dataset. The goal of these controlled experiments is to isolate the effect of the MUFI wrapper relative to the same underlying backbone, rather than to tune a separate fidelity level for each model.

Table 12 underlies the communication-radius analysis of Section 4.3: on the `VKI-LS59` Mach field, the max-min spread across the three GNN backbones decreases from $0.01323$ at full resolution to $0.00052$ at Level 3. This convergence is consistent with reduced under-reaching; it does not imply that the transferred latent states are numerically identical, which would require a separate representation-similarity analysis.

## D FIDELITY-LEVEL SWEEPS

We use MiSe-GNN (Nguyen et al., 2026), an error-aware mesh GNN, as a representative strong coarse backbone for the fidelity sweep. The level acts as a discrete operating knob: milder coarsening preserves more local information, while deeper coarsening reduces propagation cost but may remove information needed by high-gradient or stress-dominated fields. Figure 11 shows this accuracy-cost

Table 11: Full cross-backbone results on `2D_Profile` at Level 2. Each MuFi row wraps the corresponding full-resolution backbone using the same MuFi alignment and FiLM decoder. RRMSE is reported per field; macro RRMSE is the unweighted field average. Lower RRMSE and lower epoch time are better.

| Model | Mach | Pressure | Velocity-x | Velocity-y | Temperature | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|---|---|---|
| MiSe-GNN | 0.04120 | 0.02242 | 0.04914 | 0.03609 | 0.02041 | 0.03385 | 196.70 |
| MuFi-MiSe-GNN | 0.02790 | 0.02035 | 0.03282 | 0.03323 | 0.01571 | 0.02600 | 53.67 |
| MGN | 0.05082 | 0.04430 | 0.05497 | 0.04208 | 0.02312 | 0.04306 | 66.36 |
| MuFi-MGN | 0.02854 | 0.01809 | 0.03606 | 0.03967 | 0.01689 | 0.02785 | 22.90 |
| MGN-Tree | 0.05719 | 0.02975 | 0.06400 | 0.03958 | 0.02286 | 0.04268 | 85.18 |
| MuFi-MGN-Tree | 0.03026 | 0.01866 | 0.03659 | 0.03462 | 0.01752 | 0.02753 | 27.50 |
| Transolver | 0.02137 | 0.01365 | 0.02699 | 0.04640 | 0.01816 | 0.02531 | 28.48 |
| MuFi-Transolver | 0.02913 | 0.01569 | 0.03049 | 0.02368 | 0.01275 | 0.02235 | 15.45 |

Table 12: Full cross-backbone results on `VKI-LS59` at Level 3. Each MuFi row wraps the corresponding full-resolution backbone using the same MuFi alignment and FiLM decoder. RRMSE is reported per field; macro RRMSE is the unweighted field average. Lower RRMSE and lower epoch time are better.

| Model | nut | mach | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|
| MiSe-GNN | 0.15478 | 0.01901 | 0.08690 | 375.24 |
| MuFi-MiSe-GNN | 0.06225 | 0.01392 | 0.03808 | 69.63 |
| MGN | 0.20396 | 0.03224 | 0.11810 | 153.86 |
| MuFi-MGN | 0.15488 | 0.01405 | 0.08447 | 35.90 |
| MGN-Tree | 0.17992 | 0.02171 | 0.10082 | 183.67 |
| MuFi-MGN-Tree | 0.06299 | 0.01353 | 0.03826 | 38.64 |
| Transolver | 0.10911 | 0.02054 | 0.06483 | 64.23 |
| MuFi-Transolver | 0.07992 | 0.01298 | 0.04645 | 26.34 |

Table 13: Full cross-backbone results on `Tensile2D` at Level 1. Each MuFi row wraps the corresponding full-resolution backbone using the same MuFi alignment and FiLM decoder. RRMSE is reported per field; macro RRMSE is the unweighted field average. Lower RRMSE and lower epoch time are better.

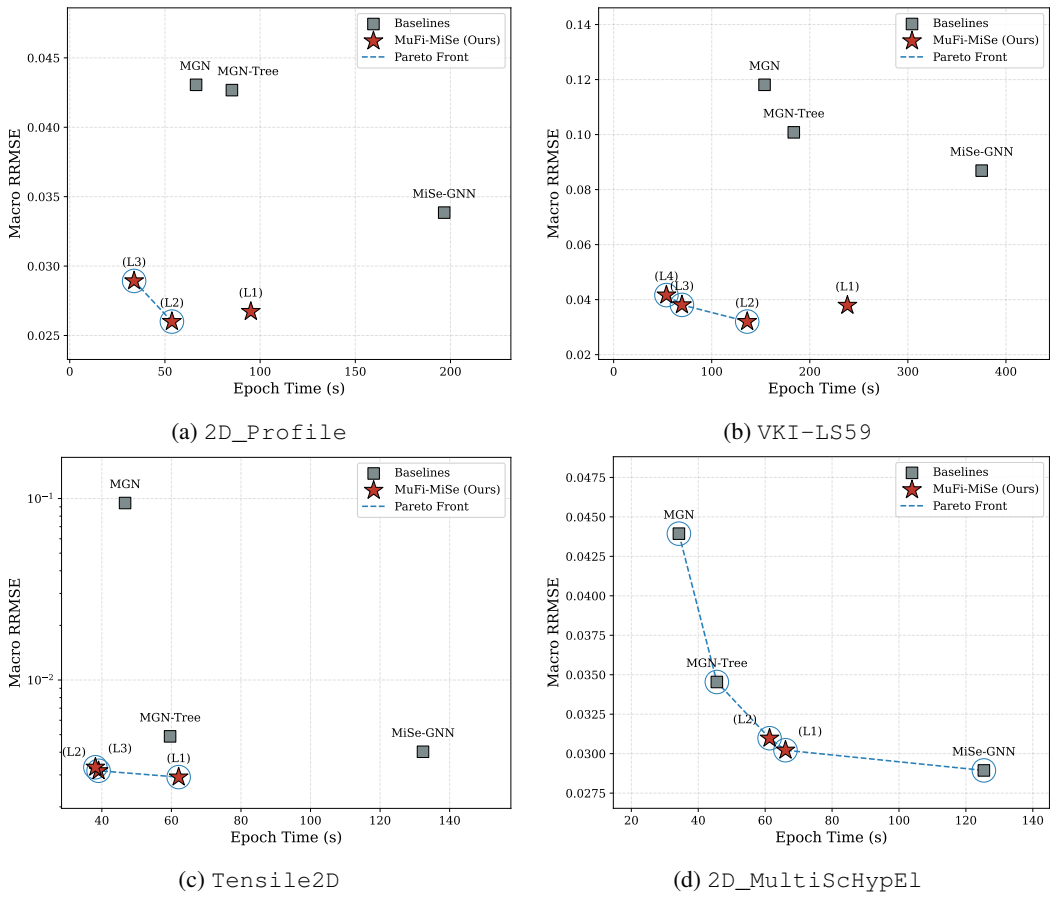
| Model | U1 | U2 | sig11 | sig22 | sig12 | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|---|---|---|
| MiSe-GNN | 0.00402 | 0.00609 | 0.00597 | 0.00178 | 0.00226 | 0.00402 | 132.34 |
| MuFi-MiSe-GNN | 0.00239 | 0.00376 | 0.00439 | 0.00185 | 0.00220 | 0.00292 | 62.09 |
| MGN | 0.08729 | 0.18587 | 0.18274 | 0.01236 | 0.00413 | 0.09448 | 46.68 |
| MuFi-MGN | 0.00216 | 0.00426 | 0.00464 | 0.00187 | 0.00206 | 0.00300 | 25.22 |
| MGN-Tree | 0.00655 | 0.00742 | 0.00615 | 0.00205 | 0.00229 | 0.00489 | 59.62 |
| MuFi-MGN-Tree | 0.00231 | 0.00363 | 0.00425 | 0.00200 | 0.00238 | 0.00291 | 29.88 |
| Transolver | 0.00204 | 0.00716 | 0.00606 | 0.00212 | 0.00279 | 0.00403 | 15.68 |
| MuFi-Transolver | 0.00278 | 0.00576 | 0.00458 | 0.00179 | 0.00229 | 0.00344 | 13.77 |

trade-off across all four datasets. In all cases, predictions are decoded and evaluated on the original fine mesh.

Tables 15, 16, 17, 18 report the per-field fidelity sweeps for MuFi-MiSe. These experiments vary only the coarse fidelity level while keeping the coarse MiSe-GNN backbone, alignment rule, decoder, training schedule, and data split fixed. The sweep shows how the selected level controls the accuracy-

Table 14: Full cross-backbone results on `2D_MultiScHypEl` at Level 1. Each MuFi row wraps the corresponding full-resolution backbone using the same MuFi alignment and FiLM decoder. RRMSE is reported per field; macro RRMSE is the unweighted field average. Lower RRMSE and lower epoch time are better.

| Model | u1 | u2 | P11 | P12 | P22 | P21 | psi | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|---|---|---|---|---|
| MiSe-GNN | 0.01464 | 0.01503 | 0.02174 | 0.04812 | 0.02181 | 0.04725 | 0.03397 | 0.02894 | 125.37 |
| MuFi-MiSe-GNN | 0.00922 | 0.01079 | 0.03155 | 0.04749 | 0.03151 | 0.04887 | 0.03209 | 0.03022 | 66.08 |
| MGN | 0.03797 | 0.03725 | 0.03391 | 0.05971 | 0.03397 | 0.06099 | 0.04376 | 0.04394 | 34.22 |
| MuFi-MGN | 0.01308 | 0.01428 | 0.03238 | 0.06027 | 0.03347 | 0.05323 | 0.03249 | 0.03417 | 32.51 |
| MGN-Tree | 0.01779 | 0.01966 | 0.02698 | 0.05658 | 0.02752 | 0.05661 | 0.03664 | 0.03454 | 45.56 |
| MuFi-MGN-Tree | 0.01288 | 0.01210 | 0.03268 | 0.06132 | 0.03320 | 0.05437 | 0.03354 | 0.03430 | 33.84 |
| Transolver | 0.01115 | 0.01083 | 0.03902 | 0.18239 | 0.08477 | 0.09043 | 0.05151 | 0.06716 | 16.50 |
| MuFi-Transolver | 0.01067 | 0.01098 | 0.03623 | 0.06491 | 0.03676 | 0.07119 | 0.04858 | 0.03990 | 16.30 |




(a) `2D_Profile` (b) `VKI-LS59`

(c) `Tensile2D` (d) `2D_MultiScHypEl`

Figure 11: Accuracy-cost trade-off across fidelity levels. Each point is a MuFi-MiSe model using MiSe-GNN as the coarse backbone at a different fidelity level. MuFi exposes multiple operating points: accuracy oriented levels can improve macro RRMSE, while deeper levels provide larger epoch time speedups. The `2D_MultiScHypEl` case shows a limitation of aggressive compression for stress-dominated fields.

cost trade-off: shallow levels preserve more local detail, while deeper levels reduce epoch time more aggressively.

Table 15: Fidelity-level sweep on `2D_Profile`. MuFi-MiSe is evaluated at increasingly coarse fidelity levels and compared with full-resolution baselines. Level 2 gives the best macro RRMSE among MuFi-MiSe variants, while Level 3 gives the fastest speed oriented point, achieving a $5.82\times$ epoch time speedup over full-resolution MiSe-GNN while retaining lower macro RRMSE.

| Model | Mach | Pressure | Velocity-x | Velocity-y | Temperature | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|---|---|---|
| MGN | 0.05082 | 0.04430 | 0.05497 | 0.04208 | 0.02312 | 0.04306 | 66.36 |
| MGN-Tree | 0.05719 | 0.02975 | 0.06400 | 0.03958 | 0.02286 | 0.04268 | 85.18 |
| MiSe-GNN | 0.04120 | 0.02242 | 0.04914 | 0.03609 | 0.02041 | 0.03385 | 196.70 |
| MuFi-MiSe (Level 1) | 0.02988 | **0.01799** | 0.03690 | **0.03154** | 0.01732 | 0.02673 | 95.12 |
| MuFi-MiSe (Level 2) | **0.02790** | 0.02035 | **0.03282** | 0.03323 | 0.01571 | **0.02600** | 53.67 |
| MuFi-MiSe (Level 3) | 0.03179 | 0.01909 | 0.03999 | 0.03817 | **0.01561** | 0.02893 | **33.81** |

Table 16: Fidelity-level sweep on `VKI-LS59`. Level 2 gives the best macro RRMSE among MuFi-MiSe variants, while Level 4 gives the fastest speed oriented operating point. Relative to full-resolution MiSe-GNN, Level 4 reduces epoch time from $375.24$s to $53.81$s ($6.97\times$) while still reducing macro RRMSE from $0.08690$ to $0.04165$.

| Model | nut | mach | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|
| MGN | 0.20396 | 0.03224 | 0.11810 | 153.86 |
| MGN-Tree | 0.17992 | 0.02171 | 0.10082 | 183.67 |
| MiSe-GNN | 0.15478 | 0.01901 | 0.08690 | 375.24 |
| MuFi-MiSe (Level 1) | 0.06250 | **0.01332** | 0.03791 | 238.34 |
| MuFi-MiSe (Level 2) | **0.05034** | 0.01370 | **0.03202** | 136.17 |
| MuFi-MiSe (Level 3) | 0.06225 | 0.01392 | 0.03808 | 69.63 |
| MuFi-MiSe (Level 4) | 0.06901 | 0.01429 | 0.04165 | **53.81** |

Table 17: Fidelity-level sweep on `Tensile2D`. The sweep highlights the trade-off between faster coarse propagation and field-wise mechanics accuracy.

| Model | U1 | U2 | sig11 | sig22 | sig12 | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|---|---|---|
| MGN | 0.08729 | 0.18587 | 0.18274 | 0.01236 | 0.00413 | 0.09448 | 46.68 |
| MGN-Tree | 0.00655 | 0.00742 | 0.00615 | 0.00205 | 0.00229 | 0.00489 | 59.62 |
| MiSe-GNN | 0.00402 | 0.00609 | 0.00597 | **0.00178** | 0.00226 | 0.00402 | 132.34 |
| MuFi-MiSe (Level 1) | 0.00239 | **0.00376** | **0.00439** | 0.00185 | **0.00220** | **0.00292** | 62.09 |
| MuFi-MiSe (Level 2) | 0.00240 | 0.00423 | 0.00464 | 0.00203 | 0.00251 | 0.00316 | 39.06 |
| MuFi-MiSe (Level 3) | **0.00235** | 0.00416 | 0.00515 | 0.00211 | 0.00276 | 0.00331 | **38.14** |

Table 18: Fidelity-level sweep on `2D_MultiScHypEl`. The displacement fields $u_1, u_2$ benefit from the coarse-to-fine interface, while the stress components $P_{ij}$ are more sensitive to coarsening because they contain localized high-gradient structures near holes and thin ligaments; see Figure 10.

| Model | u1 | u2 | P11 | P12 | P22 | P21 | psi | Macro RRMSE | Epoch time (s) |
|---|---|---|---|---|---|---|---|---|---|
| MGN | 0.03797 | 0.03725 | 0.03391 | 0.05971 | 0.03397 | 0.06099 | 0.04376 | 0.04394 | 34.22 |
| MGN-Tree | 0.01779 | 0.01966 | 0.02698 | 0.05658 | 0.02752 | 0.05661 | 0.03664 | 0.03454 | 45.56 |
| MiSe-GNN | 0.01464 | 0.01503 | **0.02174** | 0.04812 | **0.02181** | **0.04725** | 0.03397 | **0.02894** | 125.37 |
| MuFi-MiSe (Level 1) | **0.00922** | **0.01079** | 0.03155 | **0.04749** | 0.03151 | 0.04887 | 0.03209 | 0.03022 | 66.08 |
| MuFi-MiSe (Level 2) | 0.01057 | 0.01118 | 0.03083 | 0.05091 | 0.03131 | 0.05024 | **0.03180** | 0.03098 | 61.32 |

We repeated the smallest 2D improvement of Table 2 and representative 3D operating points with three seeds (42, 43, 44; Table 19). The `Rotor37` L1/L2 mean gains are large relative to the seed

Table 19: Three seed robustness check on the smallest reported 2D improvement and representative `Rotor37` operating points. Values are mean ± standard deviation over seeds 42, 43, and 44.

| Case | Setting | Full-B | MuFi |
|---|---|---|---|
| `2D_MultiScHypEl` | MGN-Tree L1 | $0.03241 \pm 0.00188$ | $0.03032 \pm 0.00056$ |
| `3D Rotor37` | MGN L1 | $0.00779 \pm 0.00004$ | $0.00597 \pm 0.00028$ |
| `3D Rotor37` | MGN L2 | $0.00779 \pm 0.00004$ | $0.00659 \pm 0.00019$ |
| `3D Rotor37` | Transolver L1 | $0.00677 \pm 0.00008$ | $0.00583 \pm 0.00017$ |
| `3D Rotor37` | Transolver L3 | $0.00677 \pm 0.00008$ | $0.00671 \pm 0.00001$ |

variability. For MGN-Tree on `2D_MultiScHypEl`, MuFi improves the mean macro RRMSE and reduces its standard deviation. Transolver L3 provides comparable accuracy at a lower-cost operating point, with the accuracy difference within baseline variability.

## E Physics diagnostics

Macro RRMSE measures global regression quality but may not fully characterize localized physical structures. We therefore report complementary diagnostics, all evaluated on the original fine mesh and using the same test split:

- $E_{\mathrm{HG},5\%}$: relative $L_2$ error restricted to the top 5% of nodes ranked by the reference mesh edge gradient.
- $d_{\mathrm{hot},1\%}$: symmetric Chamfer distance between reference and predicted top-1% hotspot sets, normalized by the domain diagonal.
- $E^p_\Gamma$: relative $L_2$ pressure error restricted to the body boundary $\Gamma$.
- $E_{F_p}$: normalized discrepancy of the pressure only integrated surface force vector.
- for `Tensile2D`, the 2D von Mises field $\sigma_{\mathrm{vm}}$ and benchmark provided maximum/top-boundary scalar targets.
- for `2D_MultiScHypEl`, the first-Piola stress intensity $s_P = \|P\|_F$, evaluated near holes, in the top 5%, at the peak, and by hotspot location.

These diagnostics use the physical quantities available in the benchmark outputs. The force diagnostic is pressure only because viscous wall shear is unavailable, and the field set does not support a complete PDE-residual/conservation audit, accordingly, we use these as targeted physical diagnostics alongside RRMSE.

These diagnostics sharpen the operating-point interpretation. All models are trained with the global MSE loss of Eq. 5; the diagnostics are used for evaluation only. On `2D_Profile` at the fixed Level 2 of Table 2, the three GNN wrappers improve all three localized and integrated diagnostics (Table 20).

The Transolver row provides a useful model-selection signal: its macro improvement is driven mainly by Velocity-y and Temperature, while Mach and Pressure RRMSE are higher in the original field-wise table. The high-gradient Mach and pressure-force diagnostics make this field-specific trade-off explicit, so an application prioritizing Mach gradients would select a different backbone or fidelity operating point rather than relying on macro RRMSE alone.

On `VKI-LS59` at Level 3, the Mach high-gradient error improves for all four backbones: MGN $0.13720 \to 0.08535$, MGN-Tree $0.09847 \to 0.08597$, MiSe-GNN $0.09355 \to 0.08224$, and Transolver $0.07433 \to 0.07082$. On `Tensile2D` at Level 1, MuFi-MGN strongly improves all five diagnostics. MuFi-MGN-Tree improves macro RRMSE and the top-boundary displacement/stress targets, while the local von Mises metrics favor a different fidelity/objective; reporting both makes the operating-point choice transparent.

`2D_MultiScHypEl` is the most fidelity-sensitive case. MuFi-MiSe-GNN improves the displacement fields ($u_1$: 0.01464 → 0.00922; $u_2$: 0.01503 → 0.01079), while macro RRMSE changes from 0.02894 to 0.03022 because localized stress components benefit from a finer operating point. The

Table 20: Physics diagnostics on `2D_Profile` at Level 2. The table reports global macro RRMSE together with high-gradient Mach error, boundary pressure error, and integrated pressure force discrepancy.

| Model | Macro RRMSE | $E^{\mathrm{Mach}}_{\mathrm{HG},5\%}$ | $E^{p}_{\Gamma}$ | $E_{F_p}$ |
|---|---|---|---|---|
| MGN | 0.04306 | 0.396441 | 0.121549 | 0.044244 |
| MuFi-MGN | 0.02785 | 0.355383 | 0.041771 | 0.007805 |
| MGN-Tree | 0.04268 | 0.411612 | 0.066206 | 0.015060 |
| MuFi-MGN-Tree | 0.02753 | 0.360881 | 0.044991 | 0.007688 |
| MiSe-GNN | 0.03385 | 0.375225 | 0.052928 | 0.010805 |
| MuFi-MiSe-GNN | 0.02600 | 0.301596 | 0.042194 | 0.007837 |
| Transolver | 0.02531 | 0.227485 | 0.040873 | 0.006860 |
| MuFi-Transolver | 0.02235 | 0.924386 | 0.040114 | 0.007197 |

Table 21: Global and high-gradient Mach errors on `VKI-LS59` at Level 3.

| Model | RRMSE | $E^{\mathrm{Mach}}_{\mathrm{HG},5\%}$ |
|---|---|---|
| MGN | 0.03224 | 0.137203 |
| MuFi-MGN | 0.01405 | 0.085354 |
| MGN-Tree | 0.02171 | 0.098465 |
| MuFi-MGN-Tree | 0.01353 | 0.085969 |
| MiSe-GNN | 0.01901 | 0.093551 |
| MuFi-MiSe-GNN | 0.01392 | 0.082244 |
| Transolver | 0.02054 | 0.074327 |
| MuFi-Transolver | 0.01298 | 0.070815 |

Table 22: Physics diagnostics on `Tensile2D` at Level 1, including von Mises and top-boundary displacement and stress errors.

| Model | Macro RRMSE | $E_{\max \sigma_{\mathrm{vm}}}$ | $E^{\sigma_{\mathrm{vm}}}_{\mathrm{top},5\%}$ | $d^{\sigma_{\mathrm{vm}}}_{\mathrm{hot},1\%}$ | $E_{\max U_2,\mathrm{top}}$ | $E_{\max \sigma_{22},\mathrm{top}}$ |
|---|---|---|---|---|---|---|
| MGN | 0.09448 | 0.012702 | 0.145224 | 0.015019 | 0.560894 | 0.013068 |
| MuFi-MGN | 0.00300 | 0.005036 | 0.007269 | 0.000124 | 0.009489 | 0.003004 |
| MGN-Tree | 0.00489 | 0.003951 | 0.006087 | 0.000088 | 0.019519 | 0.003424 |
| MuFi-MGN-Tree | 0.00291 | 0.005422 | 0.007181 | 0.000124 | 0.006625 | 0.003339 |
| MiSe-GNN | 0.00402 | 0.006994 | 0.007555 | 0.000108 | 0.016212 | 0.002855 |
| MuFi-MiSe-GNN | 0.00292 | 0.004770 | 0.005601 | 0.000092 | 0.007671 | 0.002959 |
| Transolver | 0.00403 | 0.007261 | 0.007892 | 0.000115 | 0.010450 | 0.002445 |
| MuFi-Transolver | 0.00344 | 0.006235 | 0.006032 | 0.000096 | 0.014306 | 0.001958 |

three-seed MGN-Tree Level 1 results in Table 23 show the same global/local distinction even when macro RRMSE improves, so it is reproducible rather than seed noise. MGN and Transolver behave differently: MuFi-Transolver improves all four stress diagnostics by 27–45%, and MuFi-MGN slightly improves the near-hole and top-5% errors but worsens the peak and hotspot location.

In computational physics, mesh density is chosen to control numerical discretization error; in MuFi, it can be chosen jointly for backbone-plus-decoder regression accuracy and the task metric (for example, top-5% stress error). Spatially adaptive MMG remeshing, multi-level state fusion, and task-aware training losses or fidelity selection are direct extensions. Overall, MuFi improves the global accuracy-cost trade-off on the tested tasks and exposes multiple operating points, while task-relevant local accuracy is strongest when the coarse spacing resolves the target feature scale.

Table 23: Physics diagnostics on `2D_MultiScHypEl`. We report macro RRMSE and errors of the first-Piola stress intensity $s_P = \|P\|_F$ near holes, over the top $5\%$ highest-stress nodes, at the peak, and in hotspot location. Lower values are better.

| Model | Macro RRMSE | $E^{s_P}_{\text{near-hole}}$ | $E^{s_P}_{\text{top,5\%}}$ | $E^{s_P}_{\text{peak}}$ | $d^{s_P}_{\text{hot,1\%}}$ |
|---|---|---|---|---|---|
| MGN | 0.04394 | 0.119794 | 0.123102 | 0.114452 | 0.011369 |
| MuFi-MGN | 0.03417 | 0.118442 | 0.120501 | 0.143358 | 0.012921 |
| MGN-Tree | $0.03241 \pm 0.00188$ | $0.07733 \pm 0.00072$ | $0.07851 \pm 0.00107$ | $0.07356 \pm 0.00279$ | $0.00685 \pm 0.00020$ |
| MuFi-MGN-Tree | $0.03032 \pm 0.00056$ | $0.10530 \pm 0.00316$ | $0.11862 \pm 0.00337$ | $0.07726 \pm 0.00420$ | $0.01072 \pm 0.00042$ |
| MiSe-GNN | 0.02894 | 0.060463 | 0.059332 | 0.057138 | 0.004554 |
| MuFi-MiSe-GNN | 0.03022 | 0.101558 | 0.111426 | 0.100377 | 0.012134 |
| Transolver | 0.06716 | 0.201162 | 0.159376 | 0.194391 | 0.017406 |
| MuFi-Transolver | 0.03990 | 0.110116 | 0.113503 | 0.106483 | 0.012650 |

# F Ablations

We now isolate the contribution of the decoder design and of the coarse-to-fine alignment module. Unless otherwise stated, all ablations keep the coarse MiSe-GNN backbone, training schedule, and data split fixed, and modify only the component under study. For decoder, alignment, and loss ablations, we report the RRMSE on the Temperature field of `2D_Profile` at Level 2; for neighborhood quality analysis, we additionally report results on `2D_MultiScHypEl` to verify that the observations are not dataset-specific.

## F.1 Comparison of fine-scale decoders

Table 24 compares six decoder families. Among these, the Muscat Projector ($P_{\text{Muscat}}$) serves as a purely deterministic, non-learnable baseline that directly interpolates the coarse field prediction to the fine mesh. The FiLM-based decoder achieves the lowest error ($0.0157140$), outperforming this deterministic projector, as well as the Neural Field decoder, Cross Attention decoder, and Geo-Attention decoder by roughly $44\%$ to $47\%$, and outperforming SIREN (Sitzmann et al., 2020) by $81.3\%$. In terms of efficiency, FiLM is not the absolute fastest, but its epoch time ($53.67$ s) remains moderate: it is substantially cheaper than Geo-Attention ($106.31$ s) and Cross Attention ($71.42$ s), while delivering much lower error than the lightest baselines. This comparison supports the main architectural choice of MuFi: conditioning the coordinate decoder through feature-wise modulation is more effective than naive coordinate-latent concatenation and more accurate than heavier attention-based fusion.

Table 24: Fine-scale decoder ablation on the `2D_Profile` Temperature field at Level 2. All decoders use the same aligned coarse representation and training protocol. Lower RRMSE and lower epoch time are better.

| **Model** | **RRMSE** | **Epoch time (s)** | **GPU** |
|---|---|---|---|
| Muscat Projector | 0.0294213 | **52.52** | 1 A100 |
| Neural Field Decoder | 0.0296497 | 53.83 | 1 A100 |
| Cross Attention Decoder | 0.0293720 | 71.42 | 1 A100 |
| SIREN Decoder | 0.0839516 | 63.29 | 1 A100 |
| Geo-Attention Decoder | 0.0282387 | 106.31 | 1 A100 |
| **FiLM-based Decoder** | **0.0157140** | 53.67 | 1 A100 |

## F.2 Residual correction over a deterministic field projection

This ablation is distinct from the main MuFi pipeline. In the main model, the fine decoder predicts the full field directly from the aligned coarse states $Z_f = MZ_c$. In the residual variant, we additionally project the coarse field prediction $\widehat{Y}_c$ to the fine mesh with a deterministic Muscat field-transfer operator and ask the decoder to predict only the remaining correction. Thus the deterministic projection is a baseline field approximation, not a replacement for the robust alignment of coarse

states. Let

$$\widetilde{Y}_f = P_{\text{Muscat}}(\widehat{Y}_c) \tag{22}$$

denote the Muscat projection of the predicted coarse field onto the fine mesh. The learnable decoder then predicts a residual:

$$\Delta\widehat{Y}_f = g_\phi(Z_f, X_f) \tag{23}$$

The final fine prediction is:

$$\widehat{Y}_f = \widetilde{Y}_f + \Delta\widehat{Y}_f \tag{24}$$

This residual design serves as an ablation of the decoder target, testing whether the learned decoder should reconstruct the full fine field or merely correct a deterministic coarse field projection, and is not the default MuFi architecture. The formulation is conceptually appealing because the deterministic projector already provides a smooth low-frequency approximation, allowing the learnable decoder to bypass a full reconstruction from scratch and focus purely on the unresolved local mismatch left by the baseline.

Table 25: Direct fine field prediction versus residual correction on the `2D_Profile` Temperature field. The residual variant first projects the coarse prediction to the fine mesh and then predicts the remaining correction. Lower RRMSE is better.

| Model | Direct Prediction | Residual Correction |
|---|---|---|
| Neural Field Decoder | 0.0296497 | 0.0277244 |
| Cross Attention Decoder | 0.0293720 | 0.0273424 |
| SIREN Decoder | 0.0839516 | 0.0519829 |
| Geo-Attention Decoder | 0.0282387 | 0.0280663 |
| **FiLM-based Decoder** | **0.0157140** | **0.0179803** |

Table 25 shows that this residual formulation benefits the simpler decoder families. Neural Field and Cross Attention both improve over direct prediction, and SIREN improves substantially, indicating that a fixed coarse-to-fine baseline reduces the burden of recovering the dominant low-frequency structure. In contrast, the FiLM-based decoder degrades slightly under residualization. This suggests that once the decoder is already strongly conditioned by the interpolated coarse state at every layer, forcing it to predict only a residual becomes unnecessary and can slightly restrict its flexibility. We therefore keep direct fine-field prediction in the final MuFi model and use residual correction only as an ablation.

### F.3 Absolute vs. relative coordinate parameterization

Within the residual decoder setting above, we also test whether it is preferable to decode from absolute coordinates or from coordinates expressed relative to the local coarse stencil. Specifically, we define a weighted local coarse centroid:

$$\bar{x}_i^c = \sum_{j \in N_R(i)} w_{ij}\, x_j^c, \qquad \Delta x_i = x_i^f - \bar{x}_i^c, \tag{25}$$

and feed $\Delta x_i$ to the decoder instead of the absolute coordinate $x_i^f$.

Table 26: Effect of absolute versus relative coordinate parameterization in the residual decoder on the `2D_Profile` Temperature field. Lower RRMSE is better.

| Model | Absolute Coord | Relative Coord |
|---|---|---|
| Neural Field Decoder | 0.0277244 | 0.0240143 |
| **FiLM-based Decoder** | **0.0179803** | **0.0181715** |

Table 26 shows that this change benefits the plain Neural Field decoder, reducing the error from $0.0277244$ to $0.0240143$ ($13.4\%$ relative improvement). For the FiLM decoder, however, the effect

is negligible and slightly negative ($0.0179803$ vs. $0.0181715$). The interpretation is that relative coordinates are useful when the decoder must learn local structure directly from raw coordinates, since they remove the burden of modeling global translation. Once the coarse context already modulates the decoder through FiLM, this additional recentering becomes largely redundant. Relative coordinates are therefore a helpful inductive bias for simpler coordinate decoders, but not a necessary ingredient of the final MuFi design.

### F.4 Simple vs. robust kNN neighborhoods

We then compare simple Euclidean $k$NN with the geometry-aware robust $k$NN introduced in Section A. Table 27 shows that robust $k$NN reduces the Level 2 temperature error on `2D_Profile` from $0.0173029$ to $0.0157140$, corresponding to a relative improvement of about $9.2\%$.

Table 27: Fine-field reconstruction error for simple and geometry-aware robust $k$NN alignment on the `2D_Profile` Temperature field at Level 2. Lower RRMSE is better.

| kNN method | Temperature (Level 2) |
|---|---|
| Simple kNN | 0.0173029 |
| **Robust kNN** | **0.0157140** |

Table 28: Geometric quality of the selected coarse-to-fine alignment stencil on `2D_Profile`. We report the 5th percentile of normalized doubled-triangle area $A^2/h^2$ and the extrapolation rate. Higher area and lower extrapolation rate indicate better conditioned stencils.

| kNN method | tri area2 p05 over h2 | extrapolation rate |
|---|---|---|
| Simple kNN | 0.000006 | 0.479619 |
| **Robust kNN** | **0.002792** | **0.146784** |

Table 29: Geometric quality of the selected coarse-to-fine alignment stencil on `2D_MultiScHypEl`. We report the 5th percentile of normalized doubled-triangle area $A^2/h^2$ and the extrapolation rate. Higher area and lower extrapolation rate indicate better-conditioned stencils.

| kNN method | tri area2 p05 over h2 | extrapolation rate |
|---|---|---|
| Simple kNN | 0.060036 | 0.334815 |
| **Robust kNN** | **4.954448** | **0.016861** |

To understand why this happens, Tables 28 and 29 report two geometric diagnostics of the alignment stencil: the 5th percentile of the normalized doubled-triangle area $A_2/h^2$ (higher is better) and the extrapolation rate, defined as the fraction of fine nodes that lie outside the triangle formed by their first three selected coarse neighbors (lower is better). On `2D_Profile`, robust $k$NN increases $\mathrm{p05}(A_2/h^2)$ from $6\times10^{-6}$ to $2.792\times10^{-3}$ and reduces the extrapolation rate from $0.4796$ to $0.1468$. On `2D_MultiScHypEl`, the same trend is even stronger: $\mathrm{p05}(A_2/h^2)$ increases from $0.0600$ to $4.9544$, while the extrapolation rate drops from $0.3348$ to $0.0169$. These results indicate that decoder performance depends not only on latent quality but also on stencil geometry. Avoiding nearly collinear, boundary-dominated, and extrapolative neighborhoods leads to more reliable fine-scale decoding.

### F.5 Effect of neighborhood size K

Finally, we study the neighborhood size $K$ used in the robust alignment. In 2D, $K = 3$ is the minimal geometrically complete choice, since three coarse points are sufficient to define a local triangle around the query node. Table 30 confirms that increasing the neighborhood from $K = 3$ to $K = 5$ does not improve performance. On `2D_MultiScHypEl`, $K = 3$ yields lower errors on both $u_1$ ($0.0105730$ vs. $0.0112568$) and $u_2$ ($0.0111825$ vs. $0.0130936$), with essentially identical epoch time ($61.32$ s vs.

62.99 s). Additional neighbors therefore contribute more smoothing than useful geometric coverage. We adopt $K = 3$ as the default throughout the paper.

Table 30: Effect of the alignment neighborhood size $K$ on `2D_MultiScHypEl`. The robust alignment uses $K = 3$ as the default; $K = 5$ adds two nearest unused candidates to the selected geometric triad. Lower RRMSE and lower epoch time are better.

| kNN | u1 | u2 | Epoch time (s) | GPU |
|---|---|---|---|---|
| **K=3** | **0.0105730** | **0.0111825** | **61.32** | 1 A100 |
| K=5 | 0.0112568 | 0.0130936 | 62.99 | 1 A100 |

Taken together, the ablations show that the best configuration is not the one with the heaviest decoder or the largest neighborhood, but the one with the cleanest factorization: a strong coarse solver, a geometry-aware local stencil, and a FiLM-conditioned decoder. Residualization and coordinate recentering can help simpler decoders, but they are not the source of the best final performance.

### F.6 With and without coarse loss

The coarse loss $\mathcal{L}_c$ is not required by the computational interface, but it regularizes the representation passed to the fine decoder and prevents the decoder from compensating for an uninformative coarse state. In a fixed controlled loss ablation, adding $\mathcal{L}_c$ reduced FiLM Temperature RRMSE by 25.1% and cross-attention RRMSE by 20.2%.

Table 31: Ablation of the auxiliary coarse loss $\mathcal{L}_c$ on the `2D_Profile` Temperature field. All models are evaluated with fine-field loss $\mathcal{L}_f$ alone or with the combined objective $\mathcal{L}_f + \mathcal{L}_c$. Lower RRMSE is better.

| Loss | Muscat Projector | Neural Field Decoder | Cross Attention Decoder | FiLM Decoder |
|---|---|---|---|---|
| $\mathcal{L}_f$ | 0.0303974 | 0.0302571 | 0.0367875 | 0.0209735 |
| $\mathcal{L}_f + \mathcal{L}_c$ | **0.0294213** | **0.0296497** | **0.0293720** | **0.0157140** |

## G Preprocessing, storage, and inference accounting

For each distinct fine/coarse mesh pair, coarsening, target transfer, and the sparse alignment matrix are constructed once and cached. Training and cached inference use only the sparse product $MZ_c$, whose cost is already included in the reported epoch/inference times. A new geometry requires new preprocessing, while the learned backbone and decoder weights remain shared.

The measured 2D alignment times (0.89-7.21 s) come from the reference Python implementation, which loops over fine-node queries. We use these measured values in every cold-start and break-even calculation. A vectorized 3D surface implementation constructs alignment in 0.190–0.196 s/sample, indicating substantial optimization headroom for the 2D code; Table 32 reports the costs of the current implementation.

On `2D_Profile`/L3, one-time inference preprocessing is 5.54 s, cached MuFi inference is 0.375 s/sample, and full MiSe-GNN inference is 0.834 s/sample. The cold-start overhead amortizes after approximately 13 repeated queries on the same geometry; thereafter cached inference is 2.23× faster. During training, preprocessing all 300 meshes (including target transfer) takes about 28 min and is recovered after approximately 11 of the 1000 training epochs through the 162.9 s/epoch saving.

We report numerical solver context only for `2D_Profile`, because this is the only dataset for which we have both solver wall time and hardware metadata. The recorded values are 216.9 s/sample for elsA solver on 512 CPU cores, 0.834 s/sample for full MiSe-GNN on one A100, and 0.375 s/sample for cached MuFi on one A100. Solver times for the other datasets are unavailable.

The `Rotor37` surface-mesh experiment provides an additional 3D test: MGN goes from $0.00778/174.27s$ to $0.00579/103.76s$ at L1 and $0.00738/59.72s$ at L3, Transolver goes from $0.00683/74.56s$ to $0.00575/51.28s$ at L1 and $0.00670/33.36s$ at L3.

Table 32: Preprocessing, alignment, and storage costs across datasets and fidelity levels. Times are seconds per sample, preprocessing and alignment are performed once and cached. Classic and robust alignment are compared by construction time and stencil quality.

| | | | Timing Breakdown (s/sample) | | | Alignment Time $t_{\text{align}}$ (s) | | tri_area2_p05_over_h2 | | Storage |
|---|---|---|---|---|---|---|---|---|---|---|
| **Dataset** | **Level** | **Nodes ($\lvert V_f\rvert \to \lvert V_c\rvert$)** | $t_{\text{coarse}}$ | $t_{\text{target}}$ **(Train-only)** | $t_{\text{total}} = t_{\text{coarse}} + t_{\text{align}}$ **(Inference)** | **Classic** | **Robust (Ours)** | **Classic** | **Robust (Ours)** | **(MB)** |
| **2D_Profile** | Level 1 | $37,042 \to 17,557$ | 0.8443 | 0.0844 | 6.1666 | 0.4909 | 5.3223 | 0.001006 | 0.012146 | 1.42 |
| | Level 2 | $37,042 \to 8,811$ | 0.4637 | 0.0639 | 5.6868 | 0.4150 | 5.2231 | 0.000006 | 0.002792 | 1.35 |
| | Level 3 | $37,042 \to 4,489$ | 0.3411 | 0.0515 | 5.5398 | 0.3862 | 5.1987 | 0.002560 | 0.031307 | 1.31 |
| **VKI-LS59** | Level 1 | $36,421 \to 20,759$ | 0.5321 | 0.0845 | 7.5094 | 0.4097 | 6.9773 | 0.113639 | 2.230189 | 1.41 |
| | Level 2 | $36,421 \to 10,786$ | 0.3239 | 0.0644 | 7.2876 | 0.3249 | 6.9637 | 0.095424 | 1.074336 | 1.34 |
| | Level 3 | $36,421 \to 4,265$ | 0.2324 | 0.0504 | 7.4472 | 0.2738 | 7.2148 | 0.083199 | 0.867100 | 1.29 |
| | Level 4 | $36,421 \to 1,060$ | 0.1784 | 0.0435 | 6.9843 | 0.2424 | 6.8059 | 0.097614 | 1.309297 | 1.27 |
| **Tensile2D** | Level 1 | $9,428 \to 3,650$ | 0.0425 | 0.0143 | 1.3307 | 0.0739 | 1.2882 | 0.545679 | 4.102975 | 0.25 |
| | Level 2 | $9,428 \to 1,044$ | 0.0261 | 0.0111 | 1.2716 | 0.0616 | 1.2455 | 0.727949 | 4.308281 | 0.23 |
| | Level 3 | $9,428 \to 516$ | 0.0202 | 0.0104 | 0.9092 | 0.0587 | 0.8890 | 0.495067 | 3.423569 | 0.21 |
| **2D_MultiScHypEl** | Level 1 | $5,692 \to 2,931$ | 0.0417 | 0.0127 | 1.0637 | 0.0712 | 1.0220 | 0.240129 | 5.725106 | 0.21 |
| | Level 2 | $5,692 \to 1,297$ | 0.0222 | 0.0086 | 1.0257 | 0.0576 | 1.0035 | 0.060036 | 4.954448 | 0.19 |
| **3D Rotor37** | Level 1 | $29,773 \to 14,941$ | 0.1048 | 0.1279 | 0.3009 | - | 0.1961 | - | - | 2.38 |
| | Level 2 | $29,773 \to 10,491$ | 0.1183 | 0.1025 | 0.3079 | - | 0.1896 | - | - | 1.52 |
| | Level 3 | $29,773 \to 6,039$ | 0.1325 | 0.0894 | 0.3273 | - | 0.1948 | - | - | 1.02 |

Table 33: End-to-end computational accounting on `2D_Profile`. We compare the reference elsA solver, full-resolution MiSe-GNN, and MuFi-MiSe-GNN (Level 3) in terms of resource usage, preprocessing, inference time, speedup, and Temperature RRMSE. Deep learning inference is measured on one NVIDIA A100 GPU, while the elsA result uses 512 CPU cores.

| **Method** | **Output Nodes** | **Propagation nodes** | **Param Count** | **Peak VRAM (MB)** | **Preprocessing Time (s)** | **Inference Time (ms/sample)** | **Speedup vs elsA** | **Speedup vs Full-B** | **RRMSE** |
|---|---|---|---|---|---|---|---|---|---|
| elsA (Solver)* | Fine | Fine | N/A | N/A | N/A | 216,900 (216.9s) | 1.00× | – | Ground Truth |
| MiSe-GNN (Full-B) | Fine | Fine | 929,282 | 739.32 | 0 s | 834.05 | 260× | 1.00× | 0.02041 |
| **MuFi-MiSe-GNN (Ours)** | Fine | Coarse | 1,484,291 | **436.52** | $T_{prc}$ (Cached) | **374.70** | **579**× | **2.23**× | **0.01561** |

* ***Hardware Setup:*** *elsA simulation ran on 4 HPC CPU nodes (512 AMD EPYC 9534 cores total, 32 MPI × 16 OpenMP threads). Deep learning models were evaluated on a single NVIDIA A100 GPU.*

Table 34: Computational accounting for full-resolution and MuFi models on the 3D `Rotor37` surface benchmark. Inference is measured on one NVIDIA A100 GPU, preprocessing is performed once and cached.

| **Method** | **Output Nodes** | **Propagation nodes** | **Param Count** | **Peak VRAM (MB)** | **Preprocessing Time (s)** | **Inference Time (ms/sample)** | **Speedup vs Full-B** | **Macro RRMSE** |
|---|---|---|---|---|---|---|---|---|
| MGN (Full-B) | Fine | Fine | 916,563 | 424.52 | 0 s | 80.06 | 1.00× | 0.0077819 |
| **MuFi-MGN (Level 1)** | Fine | Coarse (L1) | 1,798,658 | **342.15** | $T_{pre}$ (Cached) | **43.01** | **1.86**× | **0.0057948** |
| **MuFi-MGN (Level 2)** | Fine | Coarse (L2) | 1,798,658 | **339.25** | $T_{pre}$ (Cached) | **32.28** | **2.48**× | **0.0064366** |
| **MuFi-MGN (Level 3)** | Fine | Coarse (L3) | 1,798,658 | **338.20** | $T_{pre}$ (Cached) | **24.25** | **3.30**× | **0.0073765** |

* ***Hardware Setup:*** *All evaluations were performed on a single NVIDIA A100 GPU on the 3D Rotor37 benchmark.*

## H Qualitative results

Figures 12, 13, 14, 15, 16, 17 provide additional qualitative comparisons. Each figure follows the same layout: the top row shows predicted fields on the original fine mesh, and the bottom row shows relative error maps for the corresponding predictions. The rightmost column shows the reference field and the mesh. These visualizations complement the RRMSE tables by indicating where the remaining errors occur spatially, especially near shocks, wakes, holes, and stress-concentration regions.

## I PLAID leaderboard comparisons

We include PLAID leaderboard comparisons as external context. These results are not the primary controlled comparison in the paper, because public leaderboard entries may differ in feature availability, training resources, and implementation details. Our main claims are based on the same backbone controlled comparisons in Section 4 and Appendix C. Nevertheless, the leaderboard tables show where MuFi-MiSe stands relative to representative published PLAID submissions. The MuFi-MiSe entries take, for each output field, the best fidelity level of the sweep in Appendix D, so they combine several single-level models. With a single model per dataset (the level with the best macro RRMSE), the total errors are 0.0286 on `2D_Profile` (L2), 0.03202 on `VKI-LS59` (L2),

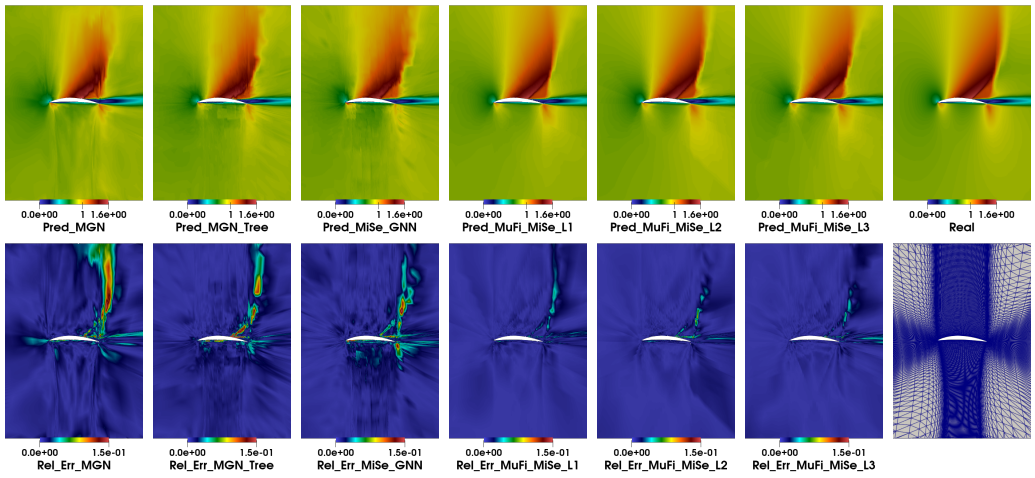



Figure 12: Qualitative `2D_Profile` example across fidelity levels. Top row: predicted Mach field on the original fine mesh. Bottom row: relative error maps for the corresponding predictions; the rightmost bottom panel shows the mesh used for reference visualization. From left to right, columns correspond to: MGN, MGN-Tree, MiSe-GNN, MuFi-MiSe (L1, L2, L3), and the reference field. MuFi preserves the dominant shock structure while reducing localized errors near the airfoil.

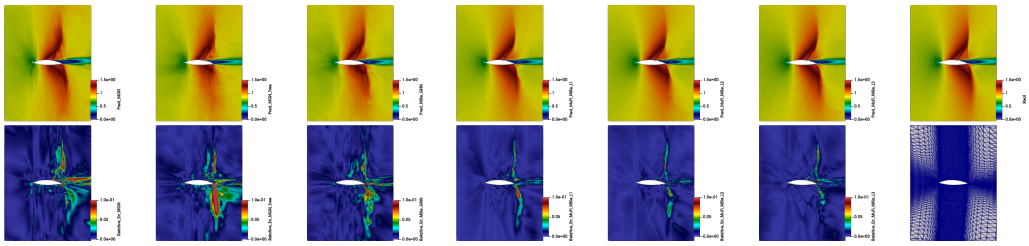


Figure 13: Additional `2D_Profile` qualitative example. The error maps show that MuFi preserves the global flow structure while concentrating remaining errors near localized high-gradient regions.

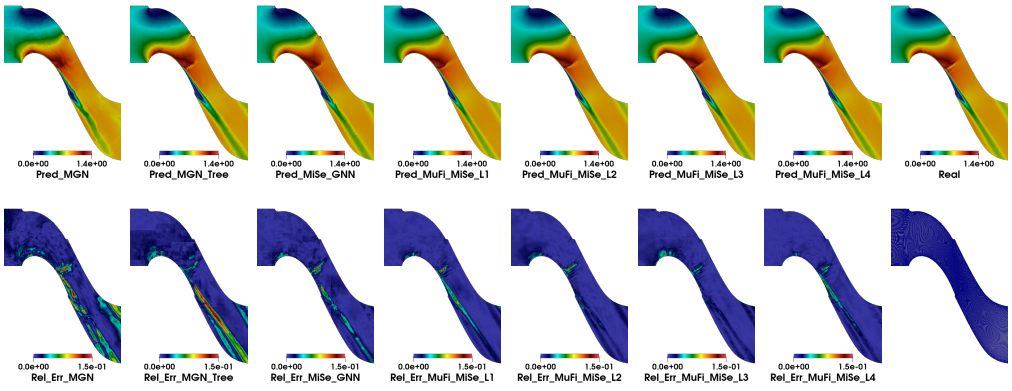



Figure 14: Qualitative `VKI-LS59` example across fidelity levels. Top row: predicted Mach field on the original fine mesh. Bottom row: relative error maps; the rightmost bottom panel shows the reference mesh. From left to right, columns correspond to: MGN, MGN-Tree, MiSe-GNN, MuFi-MiSe (L1, L2, L3, L4), and the reference solution. Deeper coarsening mainly increases localized errors in high-gradient regions, while preserving the global field structure.

0.00292 on `Tensile2D` (L1), and 0.03022 on `2D_MultiScHypEl` (L1); this changes the rank only on `2D_Profile`, from third to fourth. The public leaderboard entries encompass a wide variety

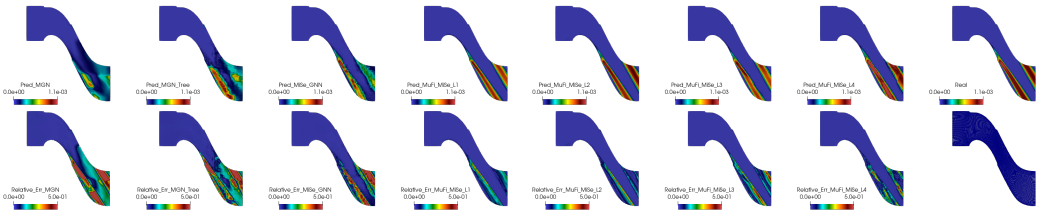


Figure 15: `VKI-LS59` qualitative example. MuFi-MiSe preserves the dominant field structure across fidelity levels, while deeper levels mainly increase localized relative errors in high-gradient regions.

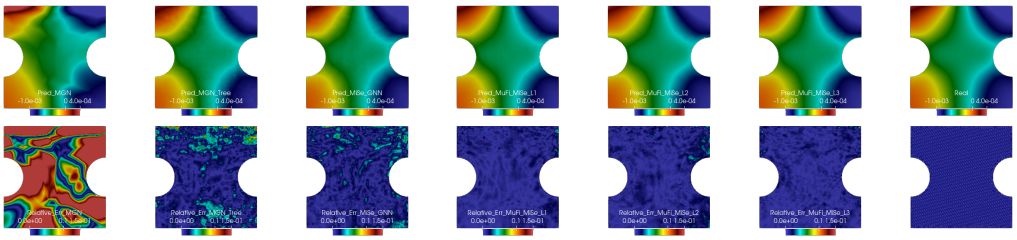


Figure 16: `Tensile2D` qualitative example. Predictions and relative errors are shown on the original fine mesh. MuFi-MiSe captures the global mechanics field while reducing the cost of backbone propagation.

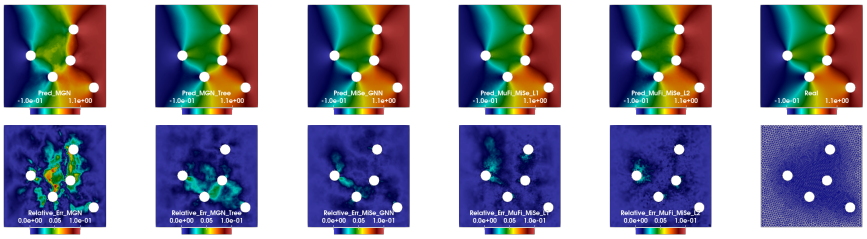


Figure 17: `2D_MultiScHypEl` qualitative example. The visualization highlights the difficulty of stress-dominated fields with holes and localized concentrations. MuFi improves displacement-like structures but can retain localized stress errors under aggressive coarsening.

of state-of-the-art neural PDE solvers, ranging from foundational neural operators like FNO (Li et al., 2021) to recent advanced graph and transformer-based architectures including PXTransolver[1], MARIO (Catalani et al., 2025), Vi-Transformer (Dosovitskiy et al., 2021), MMGP (Casenave et al., 2024), and Augur[2].

[1]Commercial solution by PhysicsX.
[2]Commercial solution by Augur.

Table 35: `2D_Profile` PLAID benchmarks. Here (w FE) and (wo FE) denote whether feature engineering methods are used.

| Rank | Id | Total error | Mach | Pressure | Velocity-x | Velocity-y |
|---|---|---|---|---|---|---|
| 1 | PXTransolver | **0.0137** | **0.0145** | **0.0100** | **0.0169** | **0.0133** |
| 2 | MARIO (11/25) | 0.0213 | 0.0230 | 0.0146 | 0.0264 | 0.0210 |
| 3 | MuFi-MiSe (wo FE) (Ours) | 0.0276 | 0.0279 | 0.0180 | 0.0328 | 0.0315 |
| 4 | MiSe-GNN (w FE) | 0.0285 | 0.0309 | 0.0212 | 0.0358 | 0.0260 |
| 5 | MARIO (10/25) | 0.0307 | 0.0337 | 0.0156 | 0.0379 | 0.0355 |
| 6 | Vi-Transformer | 0.0309 | 0.0360 | 0.0167 | 0.0403 | 0.0307 |
| 7 | MMGP | 0.0365 | 0.0439 | 0.0208 | 0.0471 | 0.0342 |
| 8 | Augur | 0.0425 | 0.0469 | 0.0248 | 0.0538 | 0.0445 |
| 9 | FNO | 0.0972 | 0.0988 | 0.0785 | 0.1148 | 0.0967 |

Table 36: `VKI-LS59` PLAID benchmarks (nut and mach fields only). Here (w FE) and (wo FE) denote whether feature engineering methods are used.

| Rank | Id | Total error | nut | mach |
|---|---|---|---|---|
| 1 | PXTransolver | **0.01445** | **0.0205** | **0.0084** |
| 2 | MARIO | 0.01855 | 0.0259 | 0.0112 |
| 3 | MuFi-MiSe (wo FE) (Ours) | 0.03180 | 0.0503 | 0.0133 |
| 4 | Augur | 0.03225 | 0.0424 | 0.0221 |
| 5 | Vi-Transformer | 0.03650 | 0.0498 | 0.0232 |
| 6 | MiSe-GNN (w FE) | 0.04480 | 0.0740 | 0.0156 |
| 7 | FNO | 0.05130 | 0.0846 | 0.0180 |
| 8 | MMGP+ | 0.05655 | 0.0822 | 0.0309 |

Table 37: `Tensile2D` PLAID benchmarks (U1, U2, sig11, sig22 and sig12 fields only). Here (w FE) and (wo FE) denote whether feature engineering methods are used.

| Rank | Id | Total error | U1 | U2 | sig11 | sig22 | sig12 |
|---|---|---|---|---|---|---|---|
| 1 | PXTransolver | **0.00068** | **0.0003** | **0.0003** | **0.0013** | **0.0006** | **0.0009** |
| 2 | MMGP | 0.00178 | 0.0015 | 0.0009 | 0.0031 | 0.0013 | 0.0021 |
| 3 | MARIO | 0.00266 | 0.0023 | 0.0030 | 0.0040 | 0.0017 | 0.0023 |
| 4 | MiSe-GNN (w FE) | 0.00288 | 0.0029 | 0.0040 | 0.0047 | 0.0013 | 0.0016 |
| 5 | MuFi-MiSe (wo FE) (Ours) | 0.00291 | 0.00235 | 0.00376 | 0.00439 | 0.00185 | 0.00220 |
| 6 | Augur | 0.00502 | 0.0037 | 0.0048 | 0.0081 | 0.0035 | 0.0050 |
| 7 | MMVT | 0.00542 | 0.0043 | 0.0051 | 0.0089 | 0.0037 | 0.0051 |
| 8 | Vi-Transformer | 0.01218 | 0.0086 | 0.0091 | 0.0184 | 0.0102 | 0.0146 |
| 9 | FNO | 0.01452 | 0.0174 | 0.0110 | 0.0250 | 0.0057 | 0.0135 |

Table 38: `2D_MultiScHypEl` PLAID benchmarks. Here (w FE) and (wo FE) denote whether feature engineering methods are used.

| Rank | Id | Total error | u1 | u2 | P11 | P12 | P22 | P21 | psi |
|---|---|---|---|---|---|---|---|---|---|
| 1 | PXTransolver | **0.01163** | **0.0026** | **0.0029** | **0.0099** | **0.0148** | **0.0101** | **0.0147** | **0.0264** |
| 2 | MiSe-GNN (w FE) | 0.02233 | 0.0120 | 0.0140 | 0.0173 | 0.0312 | 0.0176 | 0.0314 | 0.0328 |
| 3 | Augur | 0.02261 | 0.0109 | 0.0114 | 0.0208 | 0.0336 | 0.0212 | 0.0330 | 0.0274 |
| 4 | MuFi-MiSe (wo FE) (Ours) | 0.03004 | 0.0092 | 0.0108 | 0.0308 | 0.0475 | 0.0313 | 0.0489 | 0.0318 |
| 5 | FNO | 0.03280 | 0.0115 | 0.0117 | 0.0353 | 0.0513 | 0.0359 | 0.0510 | 0.0329 |
| 6 | Vi-Transformer | 0.03556 | 0.0173 | 0.0172 | 0.0337 | 0.0581 | 0.0343 | 0.0571 | 0.0312 |
| 7 | Unet | 0.03740 | 0.0291 | 0.0283 | 0.0349 | 0.0498 | 0.0347 | 0.0500 | 0.0350 |
| 8 | MARIO | 0.06234 | 0.0336 | 0.0377 | 0.0536 | 0.1067 | 0.0539 | 0.1053 | 0.0456 |