GLaDiT: Graph-Local Latent Diffusion Transformer for Flow Generation

Anonymous authors
Paper under double-blind review

Abstract

Predicting unsteady fluid flow on irregular meshes requires resolving coupled velocity and pressure dynamics over extended horizons. While graph-based simulators accommodate unstructured geometries natively, autoregressive rollouts suffer from compounding error accumulation. Conversely, trajectory diffusion models bypass recursive drift but rely on regular Euclidean grids, requiring lossy interpolations that distort mesh connectivity and inflate memory overhead. We introduce GLaDiT (Graph-Local Latent Diffusion Transformer), a graph-native Diffusion Transformer for flow generation. Operating directly on coarse mesh representations via a pretrained graph autoencoder, GLaDiT concurrently refines an entire multi-frame trajectory block in parallel, eliminating intermediate grid projections and recursive error feedback. Within the generative backbone, graph-local spatiotemporal attention pairs 1-hop spatial graph communication with global temporal attention, maintaining sparse per-block computation while structurally permitting domain-wide dependency paths across network depth. Evaluations demonstrate that GLaDiT achieves leading accuracy on most evaluation metrics across multiple datasets, while offering an adjustable compute-accuracy budget and reducing peak inference memory by $18.5\times$ compared to Cartesian grid trajectory diffusion.

1 Introduction

Simulating unsteady fluid dynamics across complex geometries is foundational to engineering and scientific computing. Traditional numerical simulations resolve the detailed spatiotemporal evolution of coupled velocity and pressure fields (Stam, 1999), but their substantial computational cost motivates learned neural surrogates (Tompson et al., 2017; Bar-Sinai et al., 2019; Kochkov et al., 2021).

Existing neural flow models fall broadly into graph-based simulators and generative PDE models. Graph-based simulators naturally accommodate unstructured meshes by representing irregular geometries directly, but typically advance states autoregressively, recursively feeding predictions back into the model and accumulating compounding errors (Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021; Janny et al., 2023; Brandstetter et al., 2022). In contrast, generative PDE models avoid recursive drift by refining multi-frame trajectory segments jointly (Rozet & Louppe, 2023; Zhou et al., 2025; Hu et al., 2025), but operate primarily in regular latent domains. Handling irregular geometries requires mapping unstructured mesh fields onto intermediate Cartesian grids (e.g., via GINO (Li et al., 2023)), which incurs heavy memory overhead and departs from the native mesh topology. These limitations motivate unifying non-autoregressive trajectory generation with an intrinsic graph representation.

We introduce GLaDiT (Graph-Local Latent Diffusion Transformer), a graph-native Diffusion Transformer framework for flow generation. Conditioned on initial states and static boundary conditions, GLaDiT synthesizes the entire future trajectory concurrently as a spatiotemporal block indexed over coarse mesh nodes. A pretrained graph autoencoder (Lino et al., 2025) maps physical fields to compact graph latents, while a DiT denoiser (Peebles & Xie, 2023) jointly refines these trajectory states over the coarse mesh topology. Maintaining latent variables directly on coarse mesh nodes eliminates the intermediate mesh-to-grid projections of regular-grid models (Zhou et al., 2025), while joint spatiotemporal denoising optimizes all future frames in parallel, circumventing autoregressive

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.

error buildup. To balance sparse computation with domain-wide communication, we design a graph-local spatiotemporal attention mechanism where tokens communicate strictly within one-hop graph neighborhoods while attending freely across all time steps. Across transformer layers and iterative denoising steps, this local communication expands across the coarse mesh, structurally permitting domain-wide dependency paths at a fraction of the per-update training cost of dense spatial attention.

Experiments across challenging flow regimes demonstrate that GLaDiT achieves leading accuracy on most evaluation metrics across multiple datasets. These results show that our approach couples pointwise field fidelity with accurate derived flow statistics. We further characterize GLaDiT's computational profile, systematically examining its memory footprint and sampling budget trade-offs.

Our contributions are threefold:

- We introduce GLaDiT, a graph-native Diffusion Transformer for flow generation, unifying native mesh representations with non-autoregressive temporal modeling and eliminating intermediate grid projections.
- We design a one-hop spatiotemporal attention mechanism that couples local physical connectivity with global temporal context, characterizing its spatial receptive-field expansion and training efficiency.
- We demonstrate leading accuracy on most evaluation metrics across multiple datasets, while reducing peak inference memory by $18.5\times$ compared to Cartesian grid trajectory diffusion.

2 Related Work

Learning-based fluid dynamics modeling primarily spans three paradigms: static flow generation (Lienen et al., 2024; Lino et al., 2025; Song & Yu, 2026), autoregressive rollout (Pfaff et al., 2021; Janny et al., 2023; Lippe et al., 2023; Serrano et al., 2024), and joint trajectory generation (Zhou et al., 2025; Hu et al., 2025). Unsteady flow simulation over complex geometries requires simultaneously respecting native mesh connectivity and capturing dynamic trajectories conditioned on initial states.

Static flow generation. Static generative models synthesize isolated physical snapshots without temporal evolution. For example, TurbDiff samples turbulent flow fields on regular Cartesian grids (Lienen et al., 2024), while LDGN combines a graph autoencoder with latent diffusion (Lino et al., 2025) and PIFM uses flow matching (Song & Yu, 2026) to generate equilibrium states on unstructured meshes. While capturing stationary distributions, these methods lack temporal mechanisms to forecast unsteady flows from initial states. We build upon the graph latent space of LDGN (Lino et al., 2025) to enable joint, non-autoregressive trajectory generation.

Autoregressive flow prediction. Graph neural networks natively operate on irregular geometries by modeling physical dynamics over mesh edges and particles (Battaglia et al., 2016; Sanchez-Gonzalez et al., 2020; Pfaff et al., 2021). In mesh domains, simulators such as MGN (Pfaff et al., 2021) and EAGLE (Janny et al., 2023) advance states through stepwise updates, but recursive feedback accumulates prediction errors over extended rollouts (Brandstetter et al., 2022). Alternatively, dynamics can be propagated in reduced latent spaces (Morton et al., 2018; Kim et al., 2019; Serrano et al., 2024); for instance, AROMA (Serrano et al., 2024) integrates neural fields with latent diffusion transitions. However, these formulations remain fundamentally autoregressive, carrying forward rollout drift over time.

Joint trajectory generation. To avoid recursive error accumulation, non-autoregressive models synthesize multi-frame horizons concurrently as a unified spatiotemporal block (Rozet & Louppe, 2023; Zhou et al., 2025; Hu et al., 2025). Text2PDE (Zhou et al., 2025) enables conditioned trajectory prediction, and WDNO (Hu et al., 2025) operates in wavelet representations. However, these models are designed for regular Euclidean grids; handling unstructured geometries requires mapping irregular mesh data onto intermediate grids (e.g., via GINO (Li et al., 2023)) and projecting back, incurring high memory overhead. In contrast, GLaDiT performs trajectory diffusion directly on coarse mesh nodes, natively retaining mesh topology without auxiliary grid projections. Within this

backbone, we incorporate a one-hop spatiotemporal attention mechanism pairing full temporal receptive fields with strictly local spatial communication, achieving high accuracy and computational efficiency.

3 BACKGROUND

3.1 PROBLEM FORMULATION

Let $\mathcal{G}_{\mathrm{p}} = (V_{\mathrm{p}}, E_{\mathrm{p}})$ be an unstructured triangular mesh with N_{p} vertices, connectivity, and node types, whose spatial structure remains static across physical time frames $t_0, \ldots, t_T$ within each simulation trajectory. Its state $X_r \in \mathbb{R}^{N_{\mathrm{p}} \times 3}$ contains velocity (u, v) and pressure p (UVP) at physical time t_r. Given the initial condition X_0, the mesh $\mathcal{G}_{\mathrm{p}}$, and static boundary information b, our goal is to generate the future trajectory $X_{1:T}$. A pretrained graph autoencoder maps each frame to a normalized posterior mean on the coarse graph $\mathcal{G}_\ell = (V_\ell, E_\ell)$:

$$\zeta_r = \mu_\phi(\mathcal{S}_x(X_r); \mathcal{G}_{\mathrm{p}}, b) \in \mathbb{R}^{N_\ell \times C_z}, \qquad z_r = (\zeta_r - \mu_z) \oslash \sigma_z. \tag{1}$$

Here $N_\ell = |V_\ell|$ and $C_z = 4$ count coarse nodes and latent channels. $\mathcal{S}_x$ normalizes physical channels, $\oslash$ denotes channelwise division, and all statistics are fitted on training data. Each coarse node has position ξ_i and static condition c_i. The denoiser models $p_\theta(z_{1:T} \mid z_0, c, \xi, \mathcal{G}_\ell)$ over the entire future block. We distinguish physical frame $r \in \{0, \ldots, T\}$ for flow dynamics from diffusion level $s \in \{0, \ldots, N_{\mathrm{diff}} - 1\}$ for noise corruption.

3.2 GRAPH LATENT REPRESENTATION

We adopt the multiscale graph autoencoder of Lino et al. (2025) to construct our compact latent representation. Its condition encoder maps static node and edge descriptors to multiscale features shared by both the field encoder and decoder. The field encoder processes normalized UVP fields through the graph hierarchy, yielding a posterior mean and variance at each coarse node. We retain the posterior mean as in Equation (1), and the decoder then reconstructs physical-node fields from these latent states and condition features.

Following the two-stage latent diffusion paradigm (Rombach et al., 2022), we train the autoencoder for reconstruction and freeze its parameters and normalization statistics during trajectory learning. The denoiser receives coarse features c and positions ξ, and the decoder additionally receives finer-scale condition and edge features, denoted by $\mathcal{C}$. This static context is shared across all frames throughout generation.

3.3 LATENT DIFFUSION

Following Ho et al. (2020), we use N_{diff} diffusion levels, draw s uniformly from $\{0, \ldots, N_{\mathrm{diff}} - 1\}$, and corrupt a clean latent variable Z as

$$Y^{(s)} = \sqrt{\bar{\alpha}_s}\, Z + \sqrt{1 - \bar{\alpha}_s}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I), \qquad \bar{\alpha}_s = \prod_{j=0}^{s} (1 - \beta_j). \tag{2}$$

The Gaussian noise elements are independent across coordinates. A denoiser is trained to predict the added noise. Following the Diffusion Transformer (DiT) architecture (Peebles & Xie, 2023), we parameterize the denoiser using transformer blocks conditioned on the diffusion step s. During inference, clean latents are iteratively recovered from pure noise using DDIM sampling (Song et al., 2021).

4 GRAPH-LOCAL JOINT TRAJECTORY DIFFUSION

GLaDiT jointly denoises an entire future trajectory over a coarse mesh graph (Figure 1). A pretrained graph autoencoder provides the compact latent representation and physical field decoder, while our trajectory denoiser coordinates temporal evolution via graph-local spatiotemporal attention.

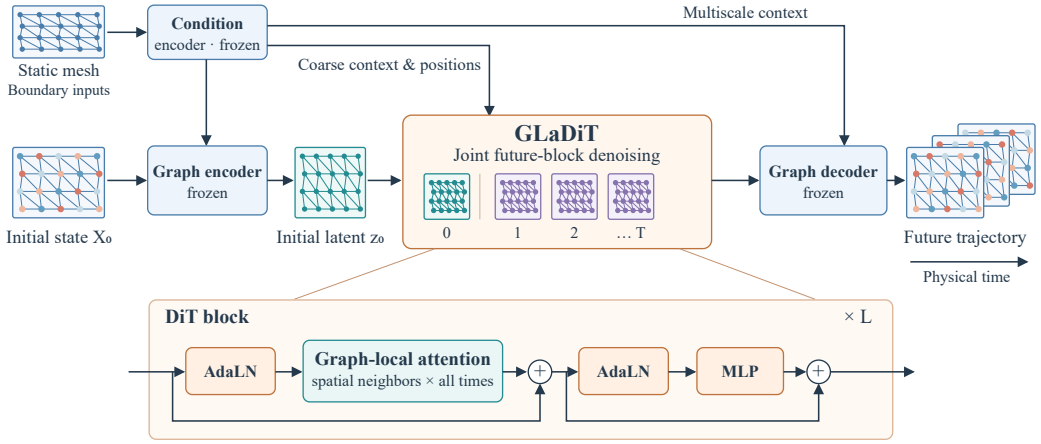


Figure 1: **GLaDiT overview.** Frozen graph encoders provide the initial latent and multiscale mesh context. GLaDiT jointly denoises all future slots with these conditions and subsequently converts them back to physical fields via the frozen decoder. The inset illustrates a DiT block , where graph-local spatiotemporal attention is employed.

Motivation and design. Synthesizing fluid trajectories requires synchronizing temporal evolution while respecting the non-Euclidean geometry of unstructured meshes. Because the underlying graph hierarchy and node indexing remain fixed across time, each coarse node maintains a consistent spatial identity. Fluid flow inherently couples local advective-diffusive transport with non-local elliptic pressure constraints. To reflect physical differential stencils while avoiding quadratic spatial attention computation, we introduce graph-indexed spatiotemporal tokens and a graph-local attention mechanism. Each token accesses the full temporal window through its local spatial neighborhood, while stacked transformer layers progressively expand spatial communication across the mesh.

4.1 Joint trajectory diffusion

Similar to multi-frame video diffusion (Ho et al., 2022), we form the entire future block $Z = [z_1, \ldots, z_T] \in \mathbb{R}^{T \times N_\ell \times C_z}$ as one diffusion variable. During training, we corrupt this block using Equation (2), where all future frames, coarse nodes, and channels share a common diffusion level s. Conditioned on the clean initial latent and static graph information, the denoiser predicts noise by minimizing

$$\mathcal{L}_\epsilon = \mathbb{E}_{Z,s,\epsilon}\left[\frac{\|f_\theta(z_0, Y^{(s)}, s, c, \xi, \mathcal{G}_\ell) - \epsilon\|_F^2}{TN_\ell C_z}\right]. \tag{3}$$

This training objective supervises all future temporal slots simultaneously in a single forward pass, eliminating the compounding error accumulation inherent to autoregressive rollouts. The variance schedule and training settings are detailed in Appendix C.2.

4.2 Graph-local spatiotemporal attention

Each node attends across the full temporal window within its spatial neighborhood. The clean initial latent and noisy future block give $T+1$ slots, with $q_{0,i} = z_{0,i}$ and $q_{r,i} = Y^{(s)}_{r,i}$ for $r > 0$. Their input tokens are

$$h^{(0)}_{r,i} = W_z q_{r,i} + W_c c_i + \psi(\widetilde{\xi}_i, r/T), \tag{4}$$

where W_z, W_c are learned linear projections and ψ embeds normalized node coordinates $\widetilde{\xi}_i$ and normalized time r/T. AdaLN-Zero layers (Peebles & Xie, 2023) modulate hidden tokens using diffusion noise level s (with $s = 0$ for initial tokens $r = 0$).

Let $d_{\mathcal{G}_\ell}(i,j)$ be graph distance and $\mathcal{N}_h(i) = \{j : d_{\mathcal{G}_\ell}(i,j) \le h\}$ the allowed neighborhood. We apply a graph-neighborhood mask (Velickovic et al., 2018) to scaled dot-product attention (Vaswani

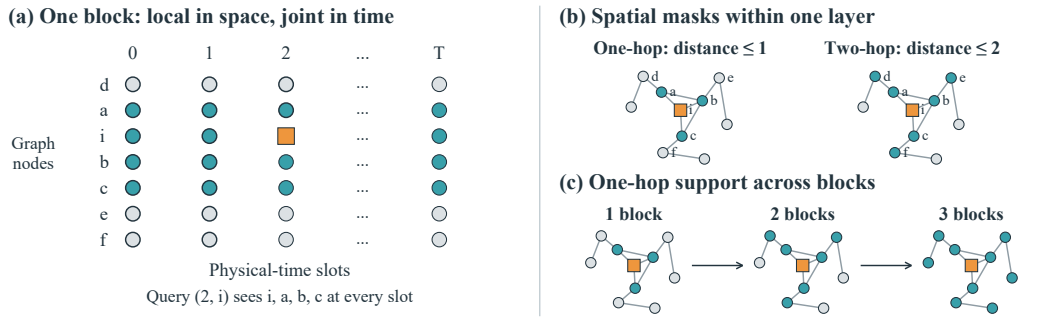


Figure 2: **Graph-local communication within and across blocks.** (a) A query at node i, slot 2 (orange), accesses all temporal slots at i and its neighbors a, b, c (teal). (b) Within one layer, one-hop and two-hop masks admit nodes at graph distance at most one and two. (c) Stacked one-hop blocks extend spatial dependencies. Gray nodes lie outside the indicated support.

et al., 2017). For head width d_a, the mask and weights are

$$M_{ij}^{(h)} = \begin{cases} 0, & j \in \mathcal{N}_h(i), \\ -\infty, & j \notin \mathcal{N}_h(i), \end{cases} \tag{5}$$

$$A_{ri,qj} = \frac{\exp\left(Q_{ri}^{\mathsf{T}} K_{qj}/\sqrt{d_a} + M_{ij}^{(h)}\right)}{\sum_{v=0}^{T} \sum_{m \in V_\ell} \exp\left(Q_{ri}^{\mathsf{T}} K_{vm}/\sqrt{d_a} + M_{im}^{(h)}\right)}. \tag{6}$$

The default setting $h = 1$ permits each node to access itself and its immediate neighbors at every temporal slot, including the initial condition (Figure 2(a)).

Within each DiT block, unconstrained temporal attention enables tokens across all frames to interact globally, while the graph-local mask confines spatial communication to one-hop stencils. Stacking $L = 24$ transformer blocks expands the spatial dependency radius to $Lh = 24$ hops per denoiser call (Proposition 1, Appendix A). Compounded across $S = 6$ iterative DDIM steps, the theoretical spatial dependency bound reaches $SLh = 144$ graph hops (Corollary 1), substantially exceeding the coarse mesh diameter ($\mathrm{diam}(\mathcal{G}_\ell) \ll 144$) and structurally permitting domain-wide dependency paths across the generative trajectory. In practice, our implementation evaluates key-value pairs directly over allowed neighborhood graphs, achieving sparse $O(|E_\ell|T^2)$ computation per block and bypassing dense attention matrices (Appendix C.2).

4.3 Sampling and Physical Output

During inference, we initialize the future trajectory block with Gaussian noise $Y^{[0]} \sim \mathcal{N}(0, I)$ and apply S deterministic DDIM updates (Song et al., 2021) to jointly refine all T frames over the forecast window. The final clean latent trajectory $Z^\star = \widehat{Z}^{[S-1]}$ is inverse-normalized and decoded framewise with the shared multiscale mesh context:

$$\widetilde{X}_r = \mathcal{S}_x^{-1}(\mathcal{D}_\phi(\sigma_z \odot z_r^\star + \mu_z; \mathcal{G}_{\mathrm{p}}, \mathcal{C})). \tag{7}$$

Under a squared L_2 error objective, the Bayes-optimal point forecast is given by the conditional posterior expectation $\mathbb{E}[X_{1:T} \mid X_0]$. To estimate this expectation under a nonlinear graph decoder $\mathcal{D}_\phi$ (where $\mathbb{E}[\mathcal{D}_\phi(z)] \neq \mathcal{D}_\phi(\mathbb{E}[z])$), we evaluate the sample mean directly in physical space across K independent generative realizations: $\widehat{X}_r = \frac{1}{K}\sum_{b=1}^{K} \widehat{X}_r^{(b)}$. This physical-field averaging aligns the Monte Carlo ensemble with the conditional mean in physical space. Appendix C.3 gives the generation algorithm and dataset-specific boundary assignments.

5 Experimental Setup

Data and prediction horizon. We evaluate models on two benchmarks with distinct flow characteristics: (i) *Airfoil* (NACA0012) contains transonic compressible flow with shock-boundary layer

interactions on an irregular mesh ($N_{\mathrm{p}} = 5{,}233$, $N_{\ell} = 407$), predicting all UVP fields over 64 frames ($\Delta t = 0.0016\,\mathrm{s}$, horizon $0.1024\,\mathrm{s}$). (ii) *CylinderFlow* models 2D incompressible flow with periodic vortex shedding ($N_{\mathrm{p}} = 1{,}885$, $N_{\ell} = 156$) over 64 frames ($\Delta t = 0.08$, horizon 5.12), with prescribed inlet/wall velocities and predicted pressure/outlet values. Each dataset contains 1,000 training and 100 validation trajectories (Pfaff et al., 2021), and all reported comparisons evaluate the full 100 validation trajectories across all future frames.

Comparison methods and evaluation protocol. We compare with autoregressive mesh simulators (**MGN** (Pfaff et al., 2021), **EAGLE** (Janny et al., 2023)), a latent neural-field model (**AROMA** (Serrano et al., 2024)), and a Cartesian grid joint diffusion model (**Text2PDE** (Zhou et al., 2025)). MGN and EAGLE provide deterministic rollouts, while AROMA and Text2PDE average metric scores over three sampling realizations. GLaDiT evaluates the physical-field mean of eight realizations ($K = 8$) with $S = 6$ DDIM steps. Appendix D details baseline architectures, training settings, and matched-sample comparisons at identical K budgets.

Metrics and computational efficiency. We evaluate relative velocity error e_{UV}, gauge-adjusted pressure RMSE (Pa), vorticity and divergence RMSE (s^{-1}), and relative kinetic-energy and enstrophy errors (e_E, e_Ω). For compressible Airfoil flow, divergence error measures agreement with the reference field (Appendix B). We measure end-to-end inference latency on a single H20 GPU, including all generated samples and CPU–GPU data transfers (Appendix F.1).

6 Results

6.1 Prediction accuracy and physical statistics

Table 1: Airfoil prediction quality on 100 validation trajectories across 64 future frames. Pressure RMSE is gauge-adjusted in Pa, vorticity and divergence RMSE are in s^{-1}, while velocity and enstrophy errors are relative. Parameter counts include both representation and prediction networks. Training and inference memory are reported as peak allocated GPU memory per device (Appendix F.1). Bold and underline mark best and second-best values.

Method	$e_{\mathrm{UV}} \downarrow$	Pressure $\downarrow$	Vorticity $\downarrow$	Divergence $\downarrow$	$e_\Omega \downarrow$	Params (M) $\downarrow$	Train GiB $\downarrow$	Infer MiB $\downarrow$
MGN	0.0470	568.7	76.7	71.2	128.4722	<u>2.33</u>	<u>1.67</u>	<u>158</u>
EAGLE	0.0201	540.8	22.6	11.1	2.8801	10.32	9.10	545
AROMA	<u>0.0163</u>	<u>354.5</u>	<u>9.9</u>	<u>5.6</u>	<u>0.1946</u>	**2.00**	**0.15**	**88**
Text2PDE	0.0310	492.0	27.8	16.8	3.2650	198.20	74.22	27,381
GLaDiT	**0.0137**	**189.0**	**8.14**	**3.28**	**0.1414**	177.37	42.57	1,483

To evaluate full-horizon trajectory prediction and derived flow statistics, we summarize 64-frame forecast quality on Airfoil in Table 1. GLaDiT achieves leading velocity and pressure accuracy that extends to velocity gradients and enstrophy, outperforming AROMA by 15.6% in relative velocity error and reducing pressure RMSE from 354.5 Pa to 189.0 Pa. The gains hold for both primary fields and derivative metrics, indicating that joint spatiotemporal denoising tracks large-scale transport as well as local velocity gradients. This predictive accuracy is underpinned by the high representation fidelity of the frozen graph autoencoder, which retains velocity, pressure, and spatial derivatives prior to trajectory learning (Table 3), while independent training seeds confirm stable optimization and reproducible performance (Table 18). Comparisons of localized flow features show that AROMA retains lower kinetic-energy and wing-surface velocity errors (Appendix F.1), reflecting complementary trade-offs between domain-wide trajectory refinement and localized surface constraints. GLaDiT's kinetic-energy error is governed by the frozen autoencoder's reconstruction floor (autoencoder-only $e_E = 0.0159$; Appendix F.1).

Repeating the comparison on CylinderFlow shows leading enstrophy accuracy (0.0321) and a pressure RMSE within 0.0002 of the best baseline (0.0366 vs. 0.0364 for AROMA), reported in Appendix F.2.

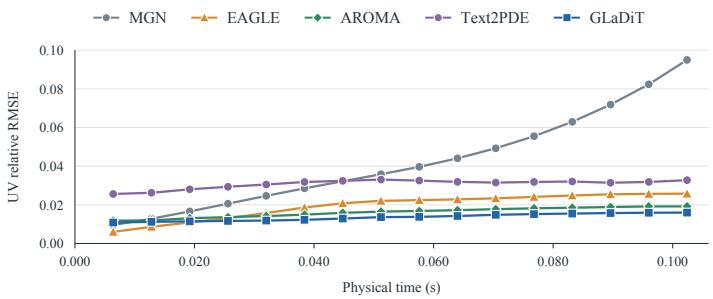


Figure 3: **Airfoil velocity error over physical time.** Markers show trajectory-mean errors on 100 validation trajectories at every fourth stored frame (frames 4–64).

6.2 Temporal evolution and generated fields

In Figure 3, we compare how single-frame velocity error evolves over time. GLaDiT maintains the lowest velocity error from intermediate forecast times through the final frame: EAGLE is more accurate during the first 13 frames ($t \leq 0.0208$ s), and GLaDiT has the lowest error from frame 14 onward. Joint denoising gives every predicted state access to temporal context throughout this window, allowing the model to refine the trajectory as a coupled sequence, while autoregressive baselines exhibit compounding error growth.

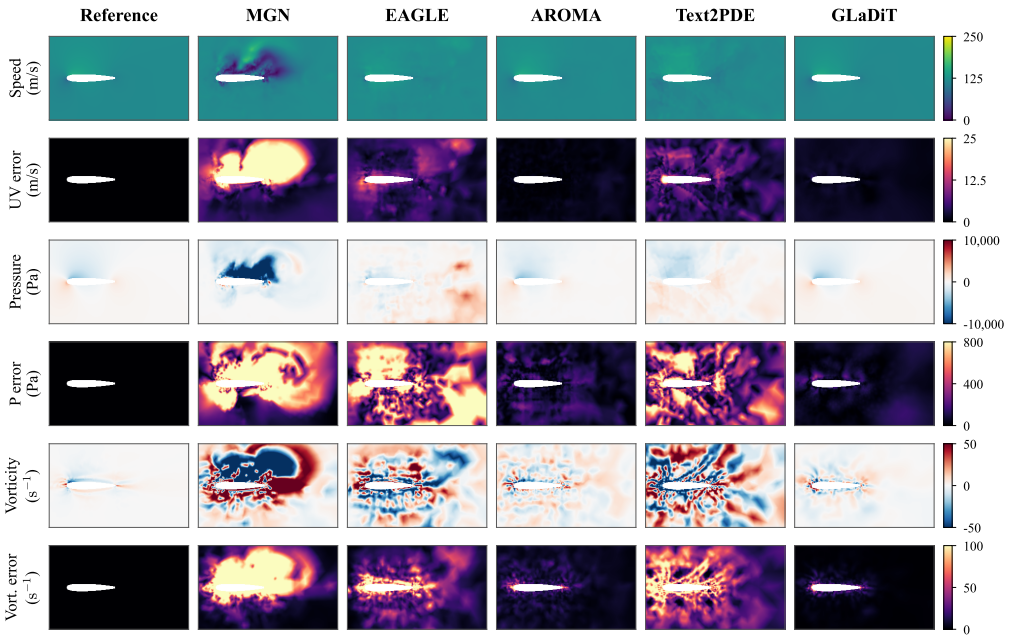


Figure 4: **Airfoil fields at the final forecast frame.** Reference and available method predictions on trajectory 1095 at future frame 64 ($t = 0.1024$ s). Rows pair velocity magnitude, gauge-adjusted pressure, and vorticity with velocity-vector error magnitude or absolute scalar error. Each row shares a linear color scale across methods. Stochastic methods show single realizations; the GLaDiT quantitative results score an eight-realization physical-field mean.

We compare final-frame predictions with reference fields to locate spatial errors as shown in Figure 4. The error maps reveal discrepancies around the airfoil and its nearby wake; GLaDiT maintains lower vector errors across the boundary layer and downstream wake. Additional cases and forecast times in Appendix F.3 extend this comparison across trajectories, with shared scales for the available methods.

Computational efficiency and memory footprint. Comparing peak GPU memory on Airfoil, we find that GLaDiT has a substantially smaller peak inference footprint than Text2PDE (1,483 MiB vs. 27,381 MiB, an $18.5\times$ reduction, Table 1). Its native graph representation concentrates latent states on coarse mesh nodes and removes the auxiliary grid and mesh-to-grid mappings used by regular-grid trajectory models. Appendix F.1 accounts for representation and dynamics training costs separately, while Table 22 extends the memory comparison to CylinderFlow.

6.3 Spatial attention ablation

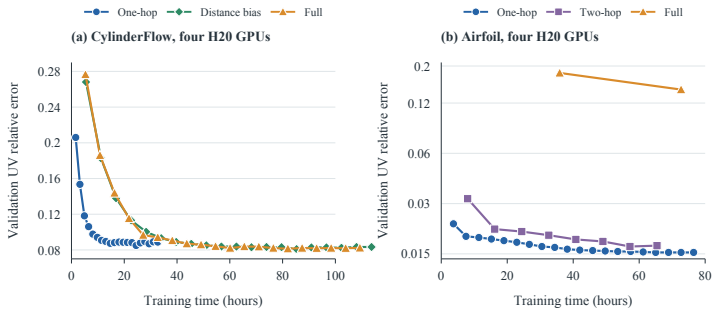


Figure 5: **Attention accuracy versus training time.** Validation relative errors versus training time on four H20 GPUs (autoencoder excluded). Curves include all recorded checkpoints.

We compare one-hop, two-hop, and full attention in Figure 5: one-hop reaches low error earlier on both datasets, reducing spatial complexity per block to $O(T^2|E_\ell|)$. At a matched budget of 62.5k updates, one-hop attention matches two-hop attention on Airfoil and comes within about 2% of full attention on CylinderFlow; with longer training, denser variants reach lower error on CylinderFlow (Appendix E.1). The compact neighborhood improves training efficiency while preserving competitive accuracy, with depth ($SLh \gg \mathrm{diam}(\mathcal{G}_\ell)$) structurally permitting domain-wide dependency paths (Appendix A). The benefit is training throughput rather than memory: one-hop attention runs $9.7\times$ faster per update than full attention on Airfoil, yet at these coarse graph sizes its Gather-SDPA kernel peaks above full attention (42.6 vs. 27.6 GiB), an implementation overhead that FlexAttention reduces (Appendix E.4).

6.4 Inference ablations

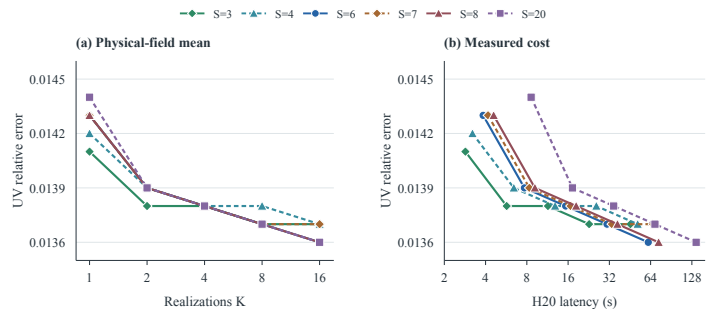


Figure 6: **GLaDiT sampling accuracy and inference cost on Airfoil.** (a) Velocity error of the physical-field mean versus K across six schedules ($S = 3, 4, 6, 7, 8, 20$; $S = 2$ appears in Appendix E.2). (b) Validation100 error versus single-H20 latency (Appendix E.2).

We vary DDIM steps S and averaged realizations K to compare accuracy and latency (Figure 6). Physical-field averaging improves velocity accuracy with diminishing returns beyond $K = 8$. Intermediate denoising schedules retain strong accuracy at lower latency than the main six-step schedule: with $S = 3, K = 1$ (2.85 s on an H20 GPU), GLaDiT achieves $e_{\mathrm{UV}} = 0.0141$, outperforming all evaluated baselines on Airfoil. Step count and averaging budget provide complementary controls over prediction quality and cost (Appendix E.3).

7 DISCUSSION

Physical inductive bias and spatial attention. Fluid dynamics inherently couples local advective-diffusive transport with non-local elliptic pressure constraints. A one-hop spatial graph mask encodes this differential stencil, accelerating training convergence while eliminating redundant dense attention computation. Compositional depth expands the spatial reach: stacking $L = 24$ transformer layers and $S = 6$ denoising updates yields a spatial dependency bound of $SLh = 144$ graph hops, well exceeding the coarse graph diameter ($\mathrm{diam}(\mathcal{G}_\ell) \ll 144$). This compositional connectivity provides the training efficiency of sparse differential stencils while structurally permitting domain-wide dependency paths.

Monte Carlo physical averaging. Under squared L_2 error objectives, the theoretical Bayes-optimal point forecast is the posterior conditional mean $\mathbb{E}[X_{1:T} \mid X_0]$. Because the graph decoder $\mathcal{D}_\phi$ is nonlinear ($\mathbb{E}[\mathcal{D}_\phi(z)] \neq \mathcal{D}_\phi(\mathbb{E}[z])$), averaging in latent space fails to yield the true physical conditional expectation. Evaluating the ensemble mean across K decoded physical realizations provides a principled Monte Carlo estimator of the model's conditional mean in physical space. Tuning the ensemble budget K allows selecting between single-shot inference speed ($K = 1$) and high-precision trajectory forecasts ($K \geq 8$).

Long-horizon rollouts and limitations. Unlike autoregressive rollouts that accumulate integration error, GLaDiT refines trajectories as a joint block, eliminating compounding rollout errors across the forecast window. For forecasting beyond fixed windows, GLaDiT can serve as a block-autoregressive generator, predicting successive chunks conditioned on the previous block's terminal state. Our current framework assumes a fixed mesh topology, fitting Eulerian CFD. Additionally, the frozen autoencoder bounds the achievable kinetic-energy accuracy on Airfoil; constraining the far-field speed in the representation objective is a direction for future work. Future directions also include dynamic mesh adaptation, investigating block-autoregressive stability over extended horizons, and scaling graph latent diffusion to 3D aerodynamic configurations.

8 CONCLUSION

We presented GLaDiT, a graph-based latent Diffusion Transformer for flow generation on unstructured meshes. Operating directly on coarse mesh hierarchies, GLaDiT models irregular geometries natively without intermediate Cartesian grid projections or lossy coordinate interpolations. Our one-hop spatiotemporal attention combines sparse local communication with global temporal context, structurally permitting domain-wide dependency paths through network depth while reducing computational overhead. GLaDiT achieves leading velocity, pressure, and enstrophy accuracy on Airfoil alongside leading enstrophy accuracy on CylinderFlow, while reducing peak inference memory by $18.5\times$ compared to Cartesian grid trajectory diffusion.

AI USE STATEMENT

Generative AI tools were used to assist with preliminary literature search and survey, experimental code development, statistical analysis of experimental results, and manuscript text polishing. Descriptions of related work and baseline methods were verified against primary sources, and experimental procedures, diagnostic results, and mathematical definitions were checked against the codebase. The authors reviewed and revised all resulting material and take full responsibility for the paper's claims, analyses, and research artifacts.

REPRODUCIBILITY STATEMENT

Section 3 defines the input information and graph representation. Section 4 gives the joint objective, attention rule, and generation procedure. Section 5 specifies the prediction and evaluation task. Appendix A supplies dependency bounds and their assumptions; Appendix B gives metric formulas, physical units, and aggregation. Appendix C records architecture, normalization, optimization, complete sampling equations, and the generation algorithm. Baseline training and prediction conventions appear in Appendix D, with ablation studies in Appendix E, and supplementary benchmark diagnostics, training curves, and flow field visualizations in Appendix F. The main Airfoil comparison uses the evaluated update-200k snapshot; its sampling and checkpoint identities are recorded alongside the supplementary results.

REFERENCES

Yohai Bar-Sinai, Stephan Hoyer, Jason Hickey, and Michael P. Brenner. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019. doi: 10.1073/pnas.1814058116. URL https://doi.org/10.1073/pnas.1814058116.

Peter W. Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/3147da8ab4a0437c15ef51a5cc7f2dc4-Abstract.html.

Johannes Brandstetter, Daniel Worrall, and Max Welling. Message Passing Neural PDE Solvers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vSix3HPYKSU.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1263–1272, 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David Fleet. Video Diffusion Models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 8633–8646, 2022. doi: 10.52202/068431-0628. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/39235c56aef13fb05a6adc95eb9d8d66-Paper-Conference.pdf.

Peiyan Hu, Rui Wang, Xiang Zheng, Tao Zhang, Haodong Feng, Ruiqi Feng, Long Wei, Yue Wang, Zhi-Ming Ma, and Tailin Wu. Wavelet diffusion neural operator. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/20eadb6338a53114a6371f102c5ded95-Paper-Conference.pdf.

Steeven Janny, Aurélien Béneteau, Madiha Nadri, Julie Digne, Nicolas Thome, and Christian Wolf. EAGLE: Large-scale learning of turbulent fluid dynamics with mesh transformers. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=mfIX4QpsARJ.

Byungsoo Kim, Vinicius C. Azevedo, Nils Thuerey, Theodore Kim, Markus Gross, and Barbara Solenthaler. Deep fluids: A generative network for parameterized fluid simulations. *Computer Graphics Forum*, 38(2):59–70, 2019. doi: 10.1111/cgf.13619. URL https://doi.org/10.1111/cgf.13619.

Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021. doi: 10.1073/pnas.2101784118. URL https://doi.org/10.1073/pnas.2101784118.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-Informed Neural Operator for Large-Scale 3D PDEs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 35836–35854, 2023. doi: 10.52202/075280-1556. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/70518ea42831f02afc3a2828993935ad-Paper-Conference.pdf.

Marten Lienen, David Lüdke, Jan Hansen-Palmus, and Stephan Günnemann. From zero to turbulence: Generative modeling for 3D flow simulation. In *International Conference on Learning Representations*, 2024. URL

https://proceedings.iclr.cc/paper_files/paper/2024/file/15972a9575e0f03bf82f00aebeb40774-Paper-Conference.pdf.

Mario Lino, Tobias Pfaff, and Nils Thuerey. Learning distributions of complex fluid simulations with diffusion graph networks. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/bfc02f1f412e8a6e710b1e00ea9ca670-Paper-Conference.pdf.

Phillip Lippe, Bastiaan S. Veeling, Paris Perdikaris, Richard E. Turner, and Johannes Brandstetter. PDE-Refiner: Achieving accurate long rollouts with neural PDE solvers. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://doi.org/10.52202/075280-2946.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Jeremy Morton, Antony Jameson, Mykel J. Kochenderfer, and Freddie Witherden. Deep dynamical modeling and control of unsteady fluid flows. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/hash/2b0aa0d9e30ea3a55fc271ced8364536-Abstract.html.

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

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter W. Battaglia. Learning mesh-based simulation with graph networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=roNqYL0_XP.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, 2022.

François Rozet and Gilles Louppe. Score-based Data Assimilation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 40521–40541, 2023. doi: 10.52202/075280-1763. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7f7fa581cc8a1970a4332920cdf87395-Paper-Conference.pdf.

Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8459–8468, 2020. URL https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html.

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. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://doi.org/10.52202/079017-0431.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Xiaozhuang Song and Tianshu Yu. Physically-Informed Flow Matching with Graph Neural Networks for Complex Fluid Dynamics. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(2):1024–1032, 2026. doi: 10.1609/aaai.v40i2.37072. URL http://dx.doi.org/10.1609/aaai.v40i2.37072.

Jos Stam. Stable fluids. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*, pp. 121–128, 1999. doi: 10.1145/311535.311548. URL http://dx.doi.org/10.1145/311535.311548.

Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating Eulerian Fluid Simulation With Convolutional Networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 3424–3433, 2017. URL https://proceedings.mlr.press/v70/tompson17a.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Anthony Zhou, Zijie Li, Michael Schneier, John R. Buchanan, Jr., and Amir Barati Farimani. Text2PDE: Latent diffusion models for accessible physics simulation. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/7274ed909a312d4d869cc328ad1c5f04-Paper-Conference.pdf.

A Theoretical Analysis and Spatial Dependency Bounds

In this appendix, we analyze the spatial dependency propagation, receptive-field expansion, and estimation risk of graph-local attention under transformer depth and iterative diffusion sampling. Layerwise neighborhood aggregation extends spatial dependencies in message-passing networks (Gilmer et al., 2017). We analyze this composition for graph-local attention layers and denoising calls. The statements concern encoded inputs on a fixed graph and describe allowed dependency paths. They provide an interpretation of the architecture, and the accuracy of trained models is evaluated through matched experiments.

A.1 Single-Step Spatial Dependency

Proposition 1 (Single-call dependency)**.** *Fix the graph, coordinates, model weights, noise level, and normalization statistics. Suppose input tokens are constructed nodewise, each of the L blocks mixes nodes only through h-hop attention, and all remaining operations act tokenwise. Then the noise prediction at node i depends on input latents and encoded node conditions only within Lh graph hops of i, across any temporal slot.*

In particular, at differentiable inputs,

$$d_{\mathcal{G}_\ell}(i,j) > Lh \quad \Longrightarrow \quad \frac{\partial \widehat{\epsilon}_{r,i}}{\partial Y^{(s)}_{q,j}} = 0. \tag{8}$$

Here, Lh bounds the allowed spatial support, within which all temporal slots are accessible. The learned weights and gates determine which allowed paths contribute. Geometry is held fixed because coordinate normalization uses the graph's bounding box. Figure 2(c) illustrates the support after one and two blocks of one-hop attention on the same graph.

Proof of Proposition 1. Let $D_\ell(i)$ be the set of original input-node indices on which a hidden token at node i can depend after ℓ blocks, taking the union over time slots. Nodewise input construction gives $D_0(i) \subseteq \{i\}$. At block $\ell+1$, the queries at i and the keys and values at $j \in \mathcal{N}_h(i)$ determine the attention output. Its softmax denominator uses this same neighborhood because all other logits are masked. Thus

$$D_{\ell+1}(i) \subseteq \bigcup_{j \in \mathcal{N}_h(i)} D_\ell(j). \tag{9}$$

Tokenwise normalization, modulation, MLPs, and output projections preserve the same spatial bound. Residual connections preserve it because $i \in \mathcal{N}_h(i)$. If every node in $D_\ell(j)$ lies within ℓh hops of j, the triangle inequality places every node on the right-hand side within $(\ell+1)h$ hops of i. Induction gives the Lh bound. Functional independence outside that set gives Equation (8) where derivatives exist. □

Here, the encoded conditions are input arrays to the denoiser. A change to a physical boundary descriptor before condition encoding can change several arrays at once, and moving a mesh vertex can change graph construction or coordinate scaling. Both involve operations outside the fixed-geometry denoiser covered by the statement.

A.2 Multi-Step Receptive Field Expansion

Corollary 1 (Repeated denoising)**.** *Under the same assumptions, S denoiser calls interleaved with pointwise DDIM updates give a final latent at node i whose dependencies on the initial Gaussian noise, clean initial latents, and encoded conditions lie within at most SLh graph hops of i.*

Proof of Corollary 1. Each nonterminal DDIM update can be written

$$Y^{[k+1]} = a_k Y^{[k]} + b_k f_\theta(z_0, Y^{[k]}, s_k, c, \xi, \mathcal{G}_\ell), \tag{10}$$

where, writing $\alpha = \bar{\alpha}_{s_k}$ and $\alpha' = \bar{\alpha}_{s_{k+1}}$, $a_k = \sqrt{\alpha'/\alpha}$ and $b_k = \sqrt{1-\alpha'} - a_k\sqrt{1-\alpha}$. The terminal clean estimate has the same pointwise form with $a_k = 1/\sqrt{\alpha}$ and $b_k = -\sqrt{(1-\alpha)/\alpha}$. All coefficients are fixed scalars for the selected schedule.

Initially, $Y^{[0]}$ depends only on the Gaussian noise at the same node. Proposition 1 shows that a call expands dependence on the current sample by at most Lh hops. The scalar combination introduces no additional spatial mixing, so after S calls the noise dependence is at most SLh hops. The first call introduces z_0 and c within Lh hops. Each later call both propagates this dependence by at most Lh and reintroduces the same conditions locally. Their union after S calls also lies within SLh hops. □

This argument counts denoiser evaluations, including the terminal evaluation. These iterations operate on a fixed physical-time window.

For our primary configuration ($L = 24, S = 6, h = 1$), the maximum spatial support bound is $SLh = 144$ hops. On both Airfoil ($\mathrm{diam}(\mathcal{G}_\ell) \approx 28$) and CylinderFlow ($\mathrm{diam}(\mathcal{G}_\ell) \approx 22$), this bound substantially exceeds the coarse mesh diameter ($\mathrm{diam}(\mathcal{G}_\ell) \ll 144$). Consequently, the composition of sparse, one-hop attention blocks across network depth and iterative denoising structurally permits domain-wide dependency paths across the full trajectory without dense $O(N_\ell^2)$ attention computation. The graph constraint specifies the paths through which the denoiser combines spatial information. The encoder and decoder introduce their own graph dependencies. These bounds concern latent communication measured in graph hops. Physical propagation speed concerns the evolution of the flow in physical space and time.

A.3 Conditional Risk Decomposition

To describe the effect of restricting information on ideal noise prediction, let Y denote a target noise element, X the full denoiser input, and U its Lh-local subinformation, including the noise level. For square-integrable Y, the corresponding squared risks satisfy the conditional-expectation identity

$$\begin{aligned}\mathbb{E}[(Y - \mathbb{E}[Y \mid \mathsf{U}])^2] = \mathbb{E}[(Y - \mathbb{E}[Y \mid \mathsf{X}])^2] \\ + \mathbb{E}[(\mathbb{E}[Y \mid \mathsf{X}] - \mathbb{E}[Y \mid \mathsf{U}])^2].\end{aligned} \tag{11}$$

The final term is the predictive information lost under the restriction: it is zero exactly when the full conditional mean is determined by the local information. This identity concerns unrestricted square-integrable predictors with different information sets. In finite, trained GLaDiT and Full networks, the errors also depend on estimation, approximation, and optimization. Their net effect on generated physical trajectories can be assessed by the matched comparison.

For a fixed graph, let $\sigma(\mathsf{U}) \subseteq \sigma(\mathsf{X})$ and $Y \in L^2$. Set $m = \mathbb{E}[Y \mid \mathsf{X}]$ and $m_U = \mathbb{E}[Y \mid \mathsf{U}]$. Expanding $Y - m_U = (Y - m) + (m - m_U)$ gives two squared terms and a cross term. The latter has zero expectation because

$$\mathbb{E}[(Y - m)(m - m_U)] = \mathbb{E}[(m - m_U)\mathbb{E}[Y - m \mid \mathsf{X}]] = 0. \tag{12}$$

This proves Equation (11). Its excess-risk term is nonnegative and is zero precisely when $m = m_U$ almost surely. This is the standard orthogonal-projection property of conditional expectation. It requires neither a conditional-independence assumption for the full target distribution nor that a finite Transformer attain either conditional mean.

A.4 Noise-to-Latent Error Relation and Physical Decoding

For a forward-corrupted sample in Equation (2), the clean estimate in Equation (24) satisfies

$$\widehat{Z} - Z = -\sqrt{\frac{1 - \bar{\alpha}_s}{\bar{\alpha}_s}}(\widehat{\epsilon} - \epsilon). \tag{13}$$

Substituting the forward process into the clean estimate and canceling the clean term gives this relation. The coefficient depends on the noise level, so the same noise-prediction error yields different clean latent errors at different levels. Accordingly, averaging the unweighted noise objective over levels differs from averaging clean latent error. Furthermore, because the graph decoder $\mathcal{D}_\phi$ is nonlinear, latent-space errors and expectations do not map linearly into physical space ($\mathbb{E}[\mathcal{D}_\phi(z)] \neq \mathcal{D}_\phi(\mathbb{E}[z])$). This mathematical distinction reinforces direct evaluation on complete decoded physical trajectories, and justifies evaluating the Monte Carlo physical-field mean rather than the latent mean.

B Physical Metrics and Evaluation Setup

This appendix specifies the mathematical definitions, nodal quadrature weights, and aggregation procedures for all field, spatial derivative, and integral physical metrics evaluated across benchmarks. The metrics operate on velocity and pressure in the dataset's physical coordinates after inverse normalization. All temporal sums below cover the 64 future frames. Let $W = \sum_i w_i = \sum_\tau A_\tau$ denote the area of the discretized fluid domain.

Airfoil predicts stored frames 1–64 at $\Delta t = 0.0016$ s, through $t = 0.1024$ s, and CylinderFlow uses $\Delta t = 0.08$, through $t = 5.12$ in dataset time units. Both datasets use stride-eight temporal subsampling. Airfoil evaluates predicted UVP at every physical node, including the airfoil surface and inflow. Density is excluded. CylinderFlow assigns the observed initial inlet and wall velocities before evaluation, while pressure and outlet values remain predicted.

B.1 Field Errors and Area Quadrature

The nodal quadrature weights are $w_i = \sum_{\tau \ni i} A_\tau/3$, where A_τ is the area of an incident mesh triangle. The full-horizon velocity relative error is

$$e_{\mathrm{UV}} = \left(\frac{\sum_{r=1}^{T} \sum_i w_i \|\widehat{\boldsymbol{u}}_{r,i} - \boldsymbol{u}_{r,i}\|_2^2}{\sum_{r=1}^{T} \sum_i w_i \|\boldsymbol{u}_{r,i}\|_2^2} \right)^{1/2}, \qquad \boldsymbol{u} = (u, v). \tag{14}$$

The absolute, component-averaged velocity RMSE and framewise relative error are

$$\mathrm{RMSE}_{\mathrm{UV}} = \sqrt{\frac{\sum_{r,i} w_i \|\widehat{\boldsymbol{u}}_{r,i} - \boldsymbol{u}_{r,i}\|_2^2}{2TW}}, \qquad e_{\mathrm{UV},r} = \sqrt{\frac{\sum_i w_i \|\widehat{\boldsymbol{u}}_{r,i} - \boldsymbol{u}_{r,i}\|_2^2}{\sum_i w_i \|\boldsymbol{u}_{r,i}\|_2^2}}. \tag{15}$$

The full-horizon relative error aggregates the squared quantities before taking the square root, and the temporal plot reports $e_{\mathrm{UV},r}$ at each frame.

For pressure error $e^p_{r,i} = \widehat{p}_{r,i} - p_{r,i}$, define $\overline{e}^p_r = W^{-1} \sum_i w_i e^p_{r,i}$. The raw and gauge-adjusted pressure scores are

$$\mathrm{RMSE}_p = \sqrt{\frac{\sum_{r,i} w_i (e^p_{r,i})^2}{TW}}, \qquad \mathrm{RMSE}_{p,\mathrm{gauge}} = \sqrt{\frac{\sum_{r,i} w_i (e^p_{r,i} - \overline{e}^p_r)^2}{TW}}. \tag{16}$$

The gauge adjustment removes one additive offset independently at each frame. For visual comparisons, reference and predicted pressure are each centered by their own area-weighted spatial means, which makes the displayed pressure difference equal to the gauge-adjusted error.

B.2 Spatial Derivatives and Physical Constraints

Spatial derivatives use the linear interpolant on each original mesh triangle. For triangle vertices (x_j, y_j), $j = 0, 1, 2$, define $\Delta_\tau = (x_1 - x_0)(y_2 - y_0) - (x_2 - x_0)(y_1 - y_0)$. For any scalar nodal field q, its constant triangle derivatives are

$$(\partial_x q)_\tau = \frac{q_0(y_1 - y_2) + q_1(y_2 - y_0) + q_2(y_0 - y_1)}{\Delta_\tau}, \tag{17}$$

$$(\partial_y q)_\tau = \frac{q_0(x_2 - x_1) + q_1(x_0 - x_2) + q_2(x_1 - x_0)}{\Delta_\tau}. \tag{18}$$

Vorticity and divergence follow as $\omega_\tau = (\partial_x v)_\tau - (\partial_y u)_\tau$ and $d_\tau = (\partial_x u)_\tau + (\partial_y v)_\tau$. For $q \in \{\omega, d\}$, the reported error is

$$\mathrm{RMSE}_q = \sqrt{\frac{\sum_{r,\tau} A_\tau (\widehat{q}_{r,\tau} - q_{r,\tau})^2}{TW}}. \tag{19}$$

This compares predicted and reference derivatives under the same mesh discretization. The divergence RMS of each field is also reported separately in the supplementary table. A reference flow

represented by piecewise-linear nodal fields can have nonzero discrete divergence, and the reference-based error complements the predicted RMS. For incompressible CylinderFlow, divergence RMSE evaluates fidelity to the reference velocity gradients under the solenoidal constraint $\nabla \cdot \boldsymbol{u} = 0$, while the direct divergence RMS quantifies intrinsic discrete constraint satisfaction. For compressible Airfoil flow where physical expansion and shock compression occur, divergence RMSE evaluates pointwise agreement with the reference field's true velocity gradients.

B.3 Integral Invariants and Temporal Correlations

The domain-mean velocity-based kinetic energy and enstrophy are

$$E_r = \frac{\sum_i w_i \|\boldsymbol{u}_{r,i}\|_2^2}{2W}, \qquad \Omega_r = \frac{\sum_\tau A_\tau \omega_{r,\tau}^2}{2W}. \tag{20}$$

Multiplying either sequence by the fixed domain area W gives its spatial integral and leaves the relative error and temporal correlation unchanged for that trajectory. For either scalar sequence q_r, we report

$$e_q = \sqrt{\frac{\sum_r (\widehat{q}_r - q_r)^2}{\sum_r q_r^2}}, \qquad \rho_q = \frac{\sum_r (\widehat{q}_r - \overline{\widehat{q}})(q_r - \overline{q})}{\sqrt{\sum_r (\widehat{q}_r - \overline{\widehat{q}})^2 \sum_r (q_r - \overline{q})^2}}. \tag{21}$$

The correlation centers each sequence in time and measures alignment of its variation, and the relative error also retains magnitude and offset differences.

B.4 Temporal Velocity Spectra

We center each nodal velocity component over the future window, $v'_{r,i,a} = u_{r,i,a} - T^{-1} \sum_{q=1}^{T} u_{q,i,a}$. Using the unnormalized discrete Fourier transform $\mathcal{F}_T$ along physical time, the area-averaged power is

$$P_m = \frac{\sum_i w_i \sum_{a=1}^{2} |(\mathcal{F}_T v'_{i,a})_m|^2}{W}, \qquad m = 1, \ldots, T/2. \tag{22}$$

The retained bins run from the first nonzero frequency through the Nyquist frequency, using their Fourier power directly at the scale defined above. The frequency of bin m is $m/(T\Delta t)$. CylinderFlow spectra use $T = 64$ and $\Delta t = 0.08$, with resolution 0.1953125 inverse-time units and Nyquist frequency 6.25. For Airfoil, the same window length at $\Delta t = 0.0016$ s gives resolution 9.765625 Hz and Nyquist frequency 312.5 Hz. These spectra describe temporal velocity variation at fixed mesh nodes, and reference and generated curves use identical transform and quadrature conventions.

B.5 Surface and Localized Flow Metrics

Velocity-field correlation is the quadrature-weighted Pearson correlation between predicted and reference velocity components over all future frames and nodes. Airfoil wing-surface error is component-averaged velocity RMSE over the nodes with the airfoil label, using equal node weights and all 64 future frames. It retains velocity units. This surface metric complements the area-weighted domain error.

B.6 Aggregation and Reporting Standards

Each scalar score is first computed over one trajectory's physical mesh and future window. The main tables then take the arithmetic mean of the 100 trajectory scores, giving each flow case equal weight. All physical diagnostics, predictive accuracy tables, and same-K sampling comparisons evaluate the full 100 validation trajectories. For GLaDiT, the field supplied to the evaluator is the physical average in Equation (27). AROMA and Text2PDE results average three realization-level scores within each trajectory, while MGN and EAGLE provide one deterministic forecast per trajectory. These conventions retain each method's reported prediction procedure.

Airfoil velocity RMSE uses $\mathrm{m\,s^{-1}}$ and pressure RMSE uses Pa, with coordinates in meters and time in seconds. CylinderFlow retains its supplied velocity, pressure, length, and time units. The energy statistic above uses squared velocity per unit mass, and Airfoil density is excluded from both the prediction and this diagnostic. Vorticity and divergence have inverse-time units, kinetic

energy has squared-velocity units, and enstrophy has inverse-time-squared units. Relative errors and correlations are dimensionless. Relative metrics with a zero reference denominator equal zero when the error is also zero and $+\infty$ otherwise, and a correlation requires nonconstant reference and predicted sequences.

C Architecture and Implementation Details

This appendix details the two-stage model architecture, neural network parameterizations, training configurations, and joint generation procedures introduced in Section 4.

C.1 Multiscale Graph Autoencoder

The graph autoencoder adapted from Lino et al. (2025) compresses unstructured physical mesh observations into compact coarse graph latents across three hierarchical mesh levels with four latent channels ($C_z = 4$). The node- and edge-update functions are parameterized by multi-layer perceptrons of hidden width 512 with two affine layers, combining relational messages via sum aggregation. The field encoder uses message-passing depths $(4, 4, 2)$, the decoder uses depths $(2, 4, 4)$, and the condition encoder uses $(1, 1, 1)$. Following the two-stage latent diffusion paradigm (Rombach et al., 2022), the trained graph autoencoder is frozen during trajectory denoiser training, with posterior means retained for generative modeling. In evaluation mode, dropout (0.1) is inactive, batch normalization statistics in the condition network remain fixed, and all encoding and decoding operations run in full FP32 precision.

The condition encoder embeds flow parameters and geometric boundaries into multiscale static descriptors. Nodal inputs include normalized inlet velocity and one-hot node-type indicators (CylinderFlow distinguishes interior/outlet, inlet, and wall nodes; Airfoil distinguishes fluid interior, airfoil surface, and inflow boundary nodes). Edge inputs contain normalized directed spatial displacement vectors divided by 0.30. The mesh hierarchy comprises five coarsening levels with scale parameters $(0.15, 0.30, 0.60, 1.20, 2.40)$; the autoencoder operates on the first three levels, with the third level defining the coarse graph $\mathcal{G}_\ell = (V_\ell, E_\ell)$. All temporal frames share this static topology and vertex ordering. The decoder receives multiscale geometric condition features $\mathcal{C}$, while the trajectory denoiser receives 512-dimensional condition features c_i and 2D spatial coordinates ξ_i at each coarse node.

Physical flow states undergo channelwise standardization $\mathcal{S}_x(X) = (X - \mu_x) \oslash \sigma_x$. Latent standardization statistics (μ_z, σ_z) are computed offline from the FP32 posterior means of all frames across the training set using FP64 accumulators (with variance floored at 10^{-12}). All normalization statistics remain fixed during trajectory learning and evaluation.

C.2 Spatiotemporal DiT Backbone

The trajectory denoiser consists of $L = 24$ transformer blocks of width 512, each with 8 attention heads (dimension $d_a = 64$) and a feedforward expansion factor of 4. Input tokens are constructed via Equation (4), where W_z, W_c are learned affine maps and ψ is a two-layer MLP embedding normalized coordinates $\widetilde{\xi}_i \in [-1, 1]^2$ and physical time r/T. Diffusion noise level s is embedded via sinusoidal features and an MLP, providing scale, shift, and gate modulation parameters in each block via AdaLN-Zero (Peebles & Xie, 2023). All modulation parameters and final output projections are initialized to zero.

Spatial attention is masked using the coarse mesh adjacency matrix (including self-connections), restricting per-block communication to $h = 1$ graph hops while attending freely across all $T + 1$ temporal slots (Figure 2). In our main implementation, key-value pairs are gathered directly from allowed topological neighborhoods across all temporal slots, and nodes with identical neighborhood degrees are grouped for batched scaled dot-product attention (SDPA). This sparse gathering evaluates non-masked token pairs in $O(|E_\ell|T^2)$ time and memory per block, bypassing dense attention matrices.

The denoiser is trained using the future-block objective in Equation (3) with $N_{\text{diff}} = 1000$ diffusion levels and linear variance schedule $\beta_j \in [10^{-4}, 2 \times 10^{-2}]$. Each training sample comprises an initial clean latent z_0 and 64 target future frames. We optimize the denoiser with AdamW (Loshchilov & Hutter, 2019) (peak learning rate 10^{-4}, linear warm-up followed by cosine decay to 10^{-7}, weight decay 10^{-6}, gradient clipping norm 1.0). Table 2 summarizes the shared architecture and dataset-specific optimization settings.

Table 2: Architecture specifications and training hyperparameters. The graph autoencoder is frozen during trajectory denoiser optimization. EMA decay is specified per update unit.

Component	Airfoil	CylinderFlow
Graph representation	Three levels; width 512; four latent channels ($C_z = 4$)	
Encoder / decoder	Message-passing depths $(4, 4, 2)$ / $(2, 4, 4)$	
Coarse condition	512 channels per coarse node	
Representation optimizer	Adam; learning rate 10^{-4}; global batch 16	
Representation objective	Normalized UVP reconstruction; KL weight 10^{-6}	
Representation schedule	Training-loss plateau reduction; gradient norm limit 1	
Evaluated representation	Epoch 2090	Epoch 1180
Denoiser backbone	24 blocks; width 512; eight heads; MLP ratio 4	
Denoiser parameters	115,013,124	
Temporal slots	One clean initial state ($r = 0$); 64 future states ($r = 1, \ldots, 64$)	
Diffusion	1,000 levels; linear variance schedule; noise prediction ϵ-target	
Denoiser optimizer	AdamW; peak 10^{-4}; weight decay 10^{-6}	
Hardware	Four NVIDIA H20 GPUs	One NVIDIA RTX 5090 GPU
Per-device / global batch	1 / 4	1 / 1
Gradient accumulation	1	1
Budget: updates / windows	250k / 1M	1M / 1M
Warm-up: updates / windows	1,000 / 4,000	4,000 / 4,000
Evaluated denoiser	200k updates; seed 0	550k updates; seed 0
EMA / decay unit	0.999 / window	0.9999 / update
Arithmetic	FP32; AMP and TF32 disabled	

Representation fidelity. Table 3 reports the reconstruction quality of the frozen graph autoencoder on the 100 validation trajectories across all 64 future frames. Each physical field is encoded to posterior mean latents and decoded back to physical nodes, confirming that the compact latent space retains fine-scale velocity, pressure, and derivative features prior to trajectory learning.

Table 3: Frozen graph autoencoder reconstruction quality on Validation100. Velocity error is relative. Airfoil pressure is in Pa; derivative errors are in s^{-1}. All pressure errors are gauge-adjusted.

Metric	Airfoil	CylinderFlow
Velocity relative error (e_{UV})	0.0129	0.0266
Pressure RMSE	171.2	0.0089
Vorticity RMSE	5.30	1.160
Divergence RMSE	3.37	0.539

C.3 Joint Sampling and Boundary Conditions

Inference is executed in normalized latent space, as detailed in Algorithm 1. Given initial physical state X_0 and static mesh features, the encoder computes clean latent z_0 and condition tensors $(c, \xi, \mathcal{G}_\ell, \mathcal{C})$. The future trajectory block is initialized with Gaussian noise $Y^{[0]} \sim \mathcal{N}(0, I)$ and refined over S deterministic DDIM updates (Song et al., 2021):

$$\widehat{\epsilon}^{[k]} = f_\theta(z_0, Y^{[k]}, s_k, c, \xi, \mathcal{G}_\ell), \tag{23}$$

$$\widehat{Z}^{[k]} = \frac{Y^{[k]} - \sqrt{1 - \bar{\alpha}_{s_k}}\,\widehat{\epsilon}^{[k]}}{\sqrt{\bar{\alpha}_{s_k}}}, \tag{24}$$

$$Y^{[k+1]} = \sqrt{\bar{\alpha}_{s_{k+1}}}\,\widehat{Z}^{[k]} + \sqrt{1 - \bar{\alpha}_{s_{k+1}}}\,\widehat{\epsilon}^{[k]}. \tag{25}$$

The final clean estimate $Z^\star = \widehat{Z}^{[S-1]}$ is inverse-standardized and decoded framewise:

$$\widetilde{X}_r = \mathcal{S}_x^{-1}(\mathcal{D}_\phi(\sigma_z \odot z_r^\star + \mu_z; \mathcal{G}_{\mathrm{p}}, \mathcal{C})), \qquad r = 1, \ldots, T. \tag{26}$$

For Airfoil, all physical nodes are predicted concurrently. For CylinderFlow, prescribed inlet and wall velocities from X_0 are assigned to the decoded output, while pressure and outlet velocities retain model predictions.

Algorithm 1 Joint Non-Autoregressive Trajectory Generation

Require: Initial field X_0, mesh $\mathcal{G}_{\mathrm{p}}$, and boundary inputs b; frozen autoencoder $(\mu_\phi, \mathcal{D}_\phi)$, denoiser f_θ, normalization $(\mathcal{S}_x, \mu_z, \sigma_z)$, and step budget S.

Encode initial field: $\zeta_0 = \mu_\phi(\mathcal{S}_x(X_0); \mathcal{G}_{\mathrm{p}}, b)$ and standardize $z_0 = (\zeta_0 - \mu_z) \oslash \sigma_z$.
Extract static condition tensors $(c, \xi, \mathcal{G}_\ell, \mathcal{C})$.
Initialize future trajectory latent block with Gaussian noise: $Y^{[0]} \sim \mathcal{N}(0, I) \in \mathbb{R}^{T \times N_\ell \times C_z}$.
Compute DDIM noise schedule $(s_0, \ldots, s_{S-1})$.
for $k = 0, \ldots, S-1$ **do**
    Predict noise block: $\widehat{\epsilon}^{[k]} \leftarrow f_\theta(z_0, Y^{[k]}, s_k, c, \xi, \mathcal{G}_\ell)$.
    Estimate clean latent block $\widehat{Z}^{[k]}$ via Equation (24).
    if $k < S-1$ **then**
        Compute next sample $Y^{[k+1]}$ via Equation (25).
    end if
end for
Final clean latent trajectory: $Z^\star \leftarrow \widehat{Z}^{[S-1]}$.
for $r = 1, \ldots, T$ **do**
    Decode physical field $\widetilde{X}_r = \mathcal{S}_x^{-1}(\mathcal{D}_\phi(\sigma_z \odot z_r^\star + \mu_z; \mathcal{G}_{\mathrm{p}}, \mathcal{C}))$.
    if CylinderFlow **then**
        Assign inlet and wall velocities from X_0.
    end if
end for
return Full forecast trajectory $[X_0, \widehat{X}_1, \ldots, \widehat{X}_T]$.

Monte Carlo physical-field averaging. For independent Gaussian draws $b \in \{1, \ldots, K\}$, let $\widehat{X}_r^{(b)}$ denote the decoded physical field. The ensemble mean prediction is evaluated as:

$$\widehat{X}_r = \frac{1}{K} \sum_{b=1}^{K} \widehat{X}_r^{(b)}. \tag{27}$$

Under squared L_2 error objectives, the Bayes-optimal point forecast is the posterior expectation $\mathbb{E}[X_{1:T} \mid X_0]$. For our generative model, this sample mean provides an unbiased Monte Carlo estimator of the conditional expectation under the induced physical distribution. Moreover, because the physical decoder $\mathcal{D}_\phi$ is nonlinear ($\mathbb{E}[\mathcal{D}_\phi(z)] \neq \mathcal{D}_\phi(\mathbb{E}[z])$), evaluating the mean directly in physical space estimates the conditional mean in physical space, whereas averaging in latent space does not correspond to a valid physical expectation and distorts localized gradients. All evaluation metrics are computed on $\widehat{X}_{1:T}$.

D BASELINE MODELS AND COMPARISON SETUP

This appendix provides architecture specifications, temporal supervision units, optimization schedules, and evaluation setups for all comparison models on Airfoil and CylinderFlow. Every method predicts 64 future frames conditioned on the initial observation X_0 and static mesh boundaries, with physical time interval $\Delta t = 0.0016$ s for Airfoil and $\Delta t = 0.08$ for CylinderFlow.

Table 4: Comparison of model representations, temporal supervision, and prediction procedures. All models forecast $T = 64$ future frames from initial state X_0 and static mesh boundaries.

Method	State Representation	Temporal Supervision Unit	Future Trajectory Prediction
MGN	Physical-node velocity, mesh edges, and node types	Adjacent velocity-frame pairs from training trajectories	64 recurrent velocity updates, each with a pressure output
EAGLE	Physical-node (u, v, p, q) with clustered graph features	Six-frame window; one sampled start per trajectory per epoch	64 recurrent physical-state updates
AROMA	Initial UVP encoded into neural-field latent tokens	Adjacent latent-frame pairs from training trajectories	64 stochastic latent transitions and field decoding
Text2PDE	Initial UVP condition; joint spatiotemporal latent grid	Joint encoding of frames 0–64	Non-autoregressive joint generation; 20 DDIM steps
GLaDiT	Initial graph latent, static conditions, and future graph latents	Full trajectory block (frames 0–64)	Non-autoregressive joint generation; 6 DDIM steps ($S = 6$)

D.1 BASELINE ARCHITECTURES AND TRAINING UNITS

MeshGraphNets (MGN). MGN (Pfaff et al., 2021) advances physical-node velocity components using directed mesh-edge features and node types, predicting pressure alongside velocity updates. Training optimizes single-step transitions across adjacent frames, with Gaussian velocity noise ($\sigma = 0.02$) added to interior nodes for rollout regularization. At inference, predictions are recursively fed back for 64 autoregressive steps.

EAGLE. EAGLE (Janny et al., 2023) operates on (u, v, p, q) states ($q = \|\boldsymbol{u}\|^2/2$) using a GraphViT backbone combining multi-scale mesh clusters (capacity 10 nodes per cluster, hidden width 512). The model is supervised over 6-frame unrolled windows and evaluated via recursive autoregressive rollout for 64 steps.

AROMA. AROMA (Serrano et al., 2024) encodes physical fields into 32 neural-field latent tokens (8 channels, hidden width 128, 3 self-attention layers). The dynamics transformer (width 128, depth 4, 4 heads) models state transitions using a 4-level diffusion process. Inference applies 64 successive stochastic latent transitions followed by implicit field decoding.

Text2PDE. Text2PDE (Zhou et al., 2025) maps irregular mesh fields onto a regular Cartesian latent grid (16^3 resolution, 16 channels) via a GINO-based variational autoencoder (Li et al., 2023). A 3D Diffusion Transformer (24 blocks, width 512, 16 heads) refines the joint spatiotemporal latent block conditioned on initial fields, using 20 DDIM steps for trajectory generation.

D.2 OPTIMIZATION SCHEDULES AND CHECKPOINT EVALUATION

Table 5 summarizes the training hyperparameters across all comparison models. All models are trained on the official 1,000 training trajectories. MGN, EAGLE, and the AROMA dynamics model train on the first 65 frames of every trajectory (the observed state and the 64-frame forecast window), the span that Text2PDE and GLaDiT generate jointly; the AROMA field autoencoder is shared with its full-length run. The evaluated checkpoint of each model is taken once its training loss has converged.

Table 5: Baseline training configurations shared across Airfoil and CylinderFlow.

Method	Stage	Learning rate	Global batch	Optimization Schedule
MGN	Predictor	10^{-4}	4	25 epochs
EAGLE	Predictor	10^{-4}	4	1,000 epochs
AROMA	Autoencoder	10^{-3}	64	10,000 epochs
	Dynamics	10^{-3}	128	5,000 epochs
Text2PDE	Autoencoder	10^{-5}	4	250,000 updates
	Diffusion	5×10^{-5}	4	250,000 updates

D.3 Matched-Budget Evaluation Setup

MGN and EAGLE provide deterministic trajectory rollouts. AROMA and Text2PDE average metric scores over three independent sampling realizations. GLaDiT evaluates the physical-field mean across eight independent realizations ($K = 8$) with $S = 6$ DDIM steps (Equation (27)). To ensure rigorous evaluation and isolate generative fidelity from ensemble post-processing, Appendix E.3 provides matched comparisons across equal sample budgets $K \in \{1, 2, 4, 8, 16\}$. All metrics are computed on the full 100 validation trajectories in physical coordinates after dataset-specific boundary assignment.

E Ablation Studies

This appendix presents ablations on the core architectural components and inference mechanisms of GLaDiT, including: (i) spatial attention connectivity stencils, (ii) temporal denoising step schedules, (iii) Monte Carlo physical-field averaging budgets, and (iv) sparse attention kernel implementations. Each ablation is evaluated from first principles with quantitative accuracy, computational throughput, and peak memory analysis across both benchmarks.

E.1 Spatial Attention Stencils

Motivation. Fluid dynamics inherently couples local transport (hyperbolic advection and parabolic diffusion) with domain-wide elliptic constraints (pressure Poisson equation). To reflect physical differential stencils, GLaDiT restricts spatial communication to local one-hop graph neighborhoods ($h = 1$), avoiding dense $O(N_\ell^2)$ spatial attention. We test the hypothesis that local spatial connectivity provides higher training throughput while network depth ($L = 24$) and iterative diffusion updates ($S = 6$) structurally permit domain-wide dependency paths ($SLh = 144 \gg \mathrm{diam}(\mathcal{G}_\ell)$).

Experimental setup. We compare four spatial communication schemes under identical DiT backbones: (1) **One-hop mask** ($h = 1$): spatial attention is confined to adjacent vertices in $\mathcal{G}_\ell$; (2) **Two-hop mask** ($h = 2$): spatial attention extends to graph distance $d_{\mathcal{G}_\ell} \leq 2$; (3) **Graph distance bias**: dense attention modulated by learned shortest-path distance biases; (4) **Full attention**: unconstrained dense spatial attention across all coarse nodes. Evaluations are conducted on both CylinderFlow ($N_\ell = 156$) and Airfoil ($N_\ell = 407$) on four NVIDIA H20 GPUs.

Table 6: Computational throughput, training speed, and peak GPU memory across spatial attention mechanisms on Airfoil and CylinderFlow (4×H20 GPUs, batch size 4). Time is seconds per optimizer update; throughput is updates per second; peak memory is maximum allocated GPU memory per device.

Benchmark	Attention Stencil	s / upd ↓	upd / s ↑	Peak GiB ↓
Airfoil ($N_\ell = 407$)	One-hop ($h = 1$, sparse)	**1.06**	**0.94**	42.6
	Two-hop ($h = 2$, sparse)	2.29	0.44	70.8
	Two-hop ($h = 2$, dense)	11.13	0.09	92.6
	Full attention	10.33	0.10	**27.6**
CylinderFlow ($N_\ell = 156$)	One-hop ($h = 1$, sparse)	**0.45**	**2.22**	17.5
	Distance bias	1.59	0.63	21.9
	Full attention	1.53	0.65	**12.4**

Table 7: Validation predictive accuracy (e_{UV}) across spatial attention mechanisms at matched training update budgets (Validation100, $S = 6, K = 8$). CylinderFlow entries average training seeds 0 and 1. Full attention on Airfoil has not reached 62.5k updates.

Benchmark	Attention Stencil	Updates	Validation100 e_{UV} ↓
Airfoil	One-hop ($h = 1$, seed 0)	62.5k	**0.0152**
	One-hop ($h = 1$, seed 1)	62.5k	**0.0149**
	Two-hop ($h = 2$, seed 0)	62.5k	<u>0.0153</u>
CylinderFlow	One-hop ($h = 1$)	62.5k	0.0998
	Graph distance bias	62.5k	<u>0.0991</u>
	Full dense attention	62.5k	**0.0977**

Empirical results and physical analysis. Table 6 and Table 7 demonstrate that: (1) **Throughput**: On Airfoil ($N_\ell = 407$), sparse one-hop attention executes in 1.06 s per update (0.94 updates/s), delivering a 9.7× speedup over full dense attention (10.33 s/update) and a 2.2× speedup over two-hop attention. On CylinderFlow, one-hop attention achieves a 3.3× speedup (7,645 updates/hour vs. 2,286 for full attention, Figure 7). (2) **Predictive fidelity**: At matched update budgets, one-hop attention is on par with two-hop attention on Airfoil (0.0152 vs. 0.0153, seed 0) and slightly

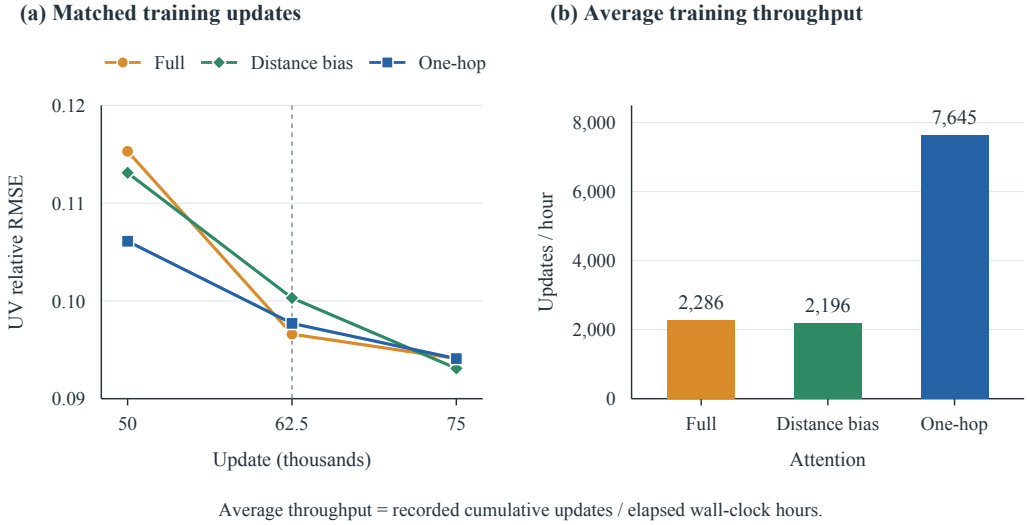


Figure 7: **Training efficiency and convergence across spatial attention mechanisms.** (a) Validation error at matched optimization checkpoints on CylinderFlow. (b) Training throughput (updates/hour) on four H20 GPUs.

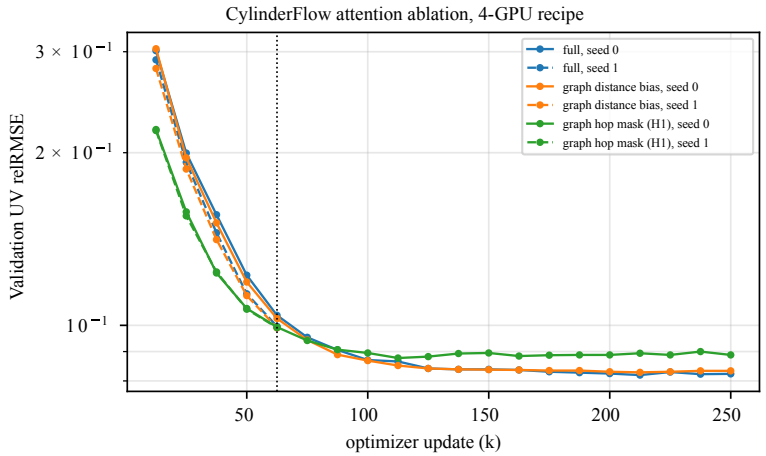


Figure 8: **Validation convergence trajectories for attention variants.** Curves track validation relative velocity error over training updates across two independent seeds.

behind full attention on CylinderFlow (0.0998 vs. 0.0977, mean of two seeds), while full attention on Airfoil was not trained to the matched budget. With longer training, denser stencils reach lower error on CylinderFlow (Figure 8). One-hop attention thus trades a small accuracy gap at long training for substantially higher training throughput, with depth ($SLh = 144 \gg \mathrm{diam}(\mathcal{G}_\ell)$) structurally permitting domain-wide dependency paths (Appendix A).

E.2 Temporal Denoising Schedules

Motivation. Standard autoregressive simulators advance physical flow states sequentially frame-by-frame, whereas GLaDiT refines the entire $T = 64$ trajectory block concurrently via iterative DDIM updates. We analyze the Pareto frontier between sampling step budget $S \in \{2, 3, 4, 5, 6, 7, 8, 20\}$ and physical trajectory accuracy.

Table 8: Airfoil Validation100 velocity relative error ($e_{\mathrm{UV}} \downarrow$) versus single-H20 latency (seconds↓) across diffusion schedules S and ensemble sizes K (update-200k checkpoint).

K	$S=2$	$S=3$	$S=4$	$S=6$	$S=20$
1	0.0248 / 2.52	**0.0141** / 2.85	0.0142 / 3.21	0.0143 / 3.84	0.0144 / 8.60
2	0.0227 / 5.06	**0.0138** / 5.70	0.0139 / 6.43	0.0139 / 7.68	0.0139 / 17.21
4	0.0217 / 10.12	**0.0138** / 11.40	**0.0138** / 12.86	**0.0138** / 15.36	**0.0138** / 34.42
8	0.0211 / 20.25	**0.0137** / 22.79	0.0138 / 25.72	**0.0137** / 30.77	**0.0137** / 68.83
16	0.0209 / 40.55	0.0137 / 45.57	0.0137 / 51.47	**0.0136** / 61.51	**0.0136** / 137.56
$S=5$	$K=1$	$K=2$	$K=4$	$K=8$	$K=16$
Error / s	0.0142 / 3.51	0.0139 / 7.02	0.0138 / 14.05	0.0137 / 28.12	0.0137 / 56.25

Table 9: CylinderFlow Validation100 velocity relative error ($e_{\mathrm{UV}} \downarrow$) versus single-H20 latency (seconds↓) across diffusion schedules S and ensemble sizes K (update-550k checkpoint).

K	$S=2$	$S=3$	$S=6$	$S=20$
1	0.1506 / 1.10	**0.0829** / 1.23	0.0830 / 1.62	0.0840 / 3.45
2	0.1452 / 2.19	0.0790 / 2.45	**0.0775** / 3.25	0.0784 / 6.87
4	0.1408 / 4.38	0.0773 / 4.90	**0.0749** / 6.50	0.0752 / 13.74
8	0.1389 / 8.75	0.0760 / 9.79	**0.0725** / 13.02	**0.0725** / 27.60
16	0.1383 / 17.50	0.0758 / 19.58	0.0721 / 26.04	**0.0720** / 54.93
Intermediate S	$S=4, K=1$	$S=4, K=8$	$S=5, K=1$	$S=5, K=8$
Error / s	0.0824 / 1.35	0.0731 / 10.84	0.0826 / 1.49	0.0726 / 11.89

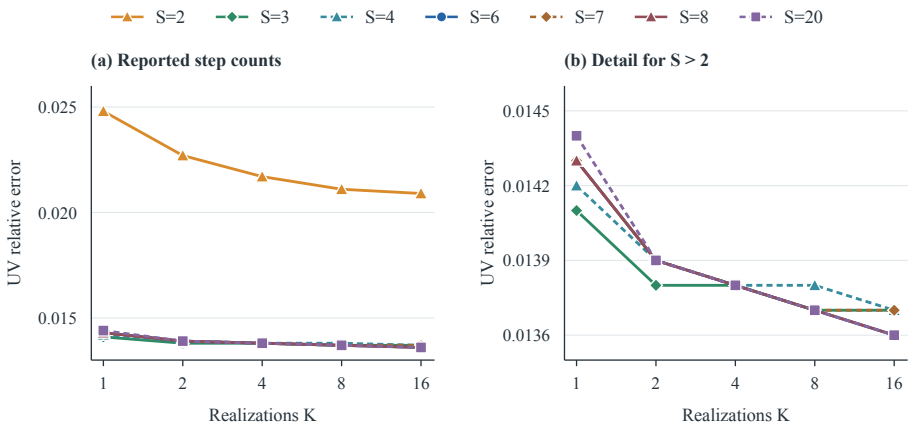


Figure 9: **Airfoil velocity error across step counts S and ensemble sizes K.** Left: full spectrum $S \in [2, 20]$. Right: expanded comparison for $S \geq 3$.

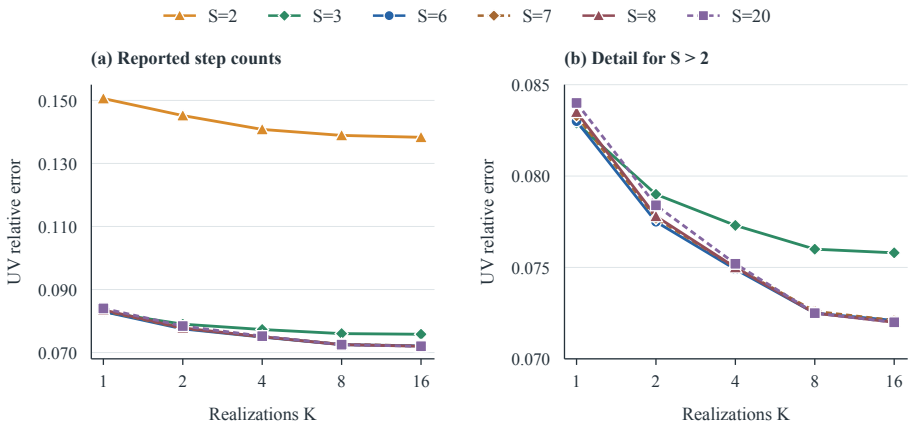


Figure 10: **CylinderFlow velocity error across step counts S and ensemble sizes K.** Left: full spectrum $S \in [2, 20]$. Right: expanded comparison for $S \geq 3$.

Empirical results and physical analysis. As shown in Table 8, Table 9, Figure 9, and Figure 10: (1) **Fast generation**: With only $S = 3$ DDIM steps (2.85 s on a single H20 GPU), GLaDiT achieves $e_{\mathrm{UV}} = 0.0141$ on Airfoil, outperforming all baseline models (AROMA 0.0163, Text2PDE 0.0310). (2) **Saturation point**: Increasing sampling steps from $S = 6$ to $S = 20$ yields minimal additional accuracy gain (0.0137 vs 0.0137) while increasing inference latency by $2.24\times$. Thus, $S = 6$ provides an effective operating point on the accuracy–latency Pareto frontier.

E.3 Monte Carlo Physical Averaging

Motivation. Under a squared L_2 loss, the Bayes-optimal point forecast is the conditional expectation $\mathbb{E}[X_{1:T} \mid X_0]$. Because the physical decoder $\mathcal{D}_\phi$ is nonlinear ($\mathbb{E}[\mathcal{D}_\phi(z)] \neq \mathcal{D}_\phi(\mathbb{E}[z])$), averaging in latent space fails to yield the conditional mean in physical space. We evaluate the impact of Monte Carlo sample size $K \in \{1, 2, 4, 8, 16\}$ when computing the physical-field expectation.

Table 10: Airfoil Validation100 velocity relative error (e_{UV}) across matched sample budgets K (update-200k checkpoint). Bold and underline mark best and second-best values per row.

K	GLaDiT ($S = 6$)	AROMA	Text2PDE ($S = 20$)
1	**0.0143**	<u>0.0163</u>	0.0314
2	**0.0139**	<u>0.0161</u>	0.0306
4	**0.0138**	<u>0.0160</u>	0.0302
8	**0.0137**	<u>0.0160</u>	0.0300
16	**0.0136**	<u>0.0160</u>	0.0300

Table 11: CylinderFlow Validation100 velocity relative error (e_{UV} ↓) / single-H20 latency (seconds↓) across matched sample budgets K.

K	GLaDiT ($S = 6$)	GLaDiT ($S = 20$)	AROMA	Text2PDE ($S = 20$)
1	<u>0.0809</u> / 1.62	0.0820 / 3.45	**0.0608 / 0.50**	0.0817 / <u>0.52</u>
2	<u>0.0750</u> / 3.25	0.0753 / 6.87	**0.0588 / 1.00**	0.0776 / <u>1.04</u>
4	0.0716 / 6.50	<u>0.0714</u> / 13.74	**0.0592 / 2.02**	0.0726 / <u>2.07</u>
8	0.0695 / 13.02	<u>0.0690</u> / 27.60	**0.0585 / 4.01**	0.0705 / <u>4.15</u>
16	0.0688 / 26.04	<u>0.0686</u> / 54.93	**0.0577 / 8.05**	0.0700 / <u>8.30</u>

Empirical results and physical analysis. (1) **Single-realization performance**: In Table 10, a single GLaDiT sample ($K = 1$, $e_{\mathrm{UV}} = 0.0143$) outperforms the 16-sample ensemble means of both AROMA (0.0160) and Text2PDE (0.0300), demonstrating that the predictive gain is primarily

Table 12: Airfoil physical diagnostics for single realizations ($K = 1$) versus physical-field averages ($K = 8$) across diffusion schedules on Validation100 (update-200k checkpoint). Pressure RMSE (p_{RMSE}) is in Pa; vorticity and divergence RMSE ($\omega_{\mathrm{RMSE}}, d_{\mathrm{RMSE}}$) are in s^{-1}; energy and enstrophy errors (e_E, e_Ω) are relative.

Configuration	K	$e_{\mathrm{UV}} \downarrow$	$p_{\mathrm{RMSE}} \downarrow$	$\omega_{\mathrm{RMSE}} \downarrow$	$d_{\mathrm{RMSE}} \downarrow$	$e_E \downarrow$	$e_\Omega \downarrow$
GLaDiT, $S = 3$	1	0.0141	198.7	9.28	3.59	0.0159	0.1262
	8	0.0137	193.4	8.37	3.33	0.0159	0.1396
GLaDiT, $S = 6$	1	0.0143	199.7	9.71	3.59	0.0159	0.1189
	8	0.0137	189.0	8.14	3.28	0.0159	0.1414
GLaDiT, $S = 20$	1	0.0144	201.6	9.90	3.61	0.0159	0.1199
	8	0.0137	188.7	8.12	3.26	0.0159	0.1460
AROMA	1	0.0163	354.8	9.91	5.64	0.0038	0.1925
	8	0.0160	349.9	9.13	5.40	0.0037	0.2101
Text2PDE, $S = 20$	1	0.0314	491.2	27.8	16.8	0.0300	3.2636
	8	0.0300	478.0	27.7	16.6	0.0283	3.2631

Table 13: Airfoil sampling variance over $R = 5$ independent repetitions of the complete $K = 8$ prediction procedure on Validation100 (mean $\pm$ sample standard deviation).

Metric	GLaDiT ($S = 6$)	AROMA ($K = 8$)	Text2PDE ($S = 20$)
Velocity relative error (e_{UV})	**0.01371** $\pm$ 0.00003	0.01600 $\pm$ 0.00003	0.03018 $\pm$ 0.00012
Pressure RMSE (Pa)	**189.0** $\pm$ 0.71	350.0 $\pm$ 0.54	477.8 $\pm$ 0.27
Vorticity RMSE (s^{-1})	**8.13** $\pm$ 0.065	9.11 $\pm$ 0.089	27.70 $\pm$ 0.007
Divergence RMSE (s^{-1})	**3.28** $\pm$ 0.004	5.40 $\pm$ 0.007	16.61 $\pm$ 0.011
Energy relative error (e_E)	0.0159 $\pm$ 0.00000	**0.0036** $\pm$ 0.00001	0.0286 $\pm$ 0.00029
Enstrophy relative error (e_Ω)	**0.1426** $\pm$ 0.00119	0.2109 $\pm$ 0.00060	3.2649 $\pm$ 0.00475

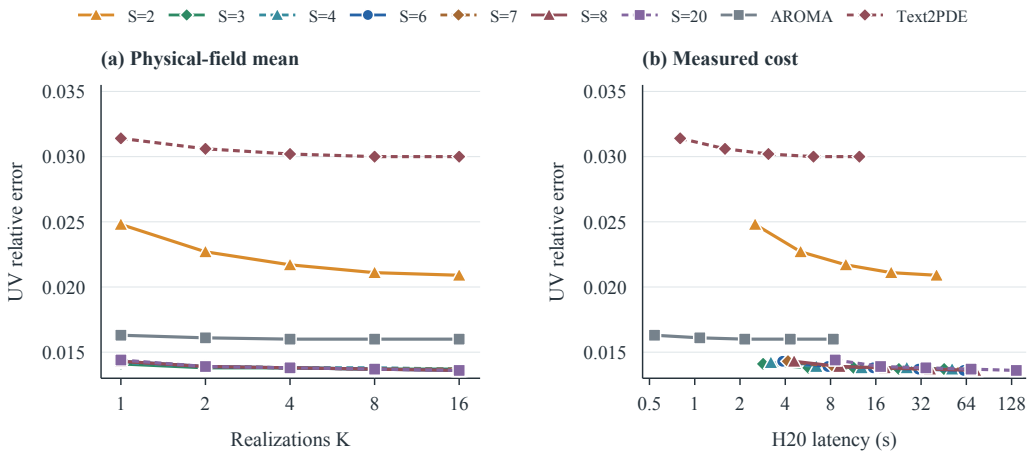


Figure 11: **Airfoil predictive accuracy and latency scaling versus ensemble budget K.** Left: relative velocity error across sample budgets. Right: accuracy–latency Pareto frontier.

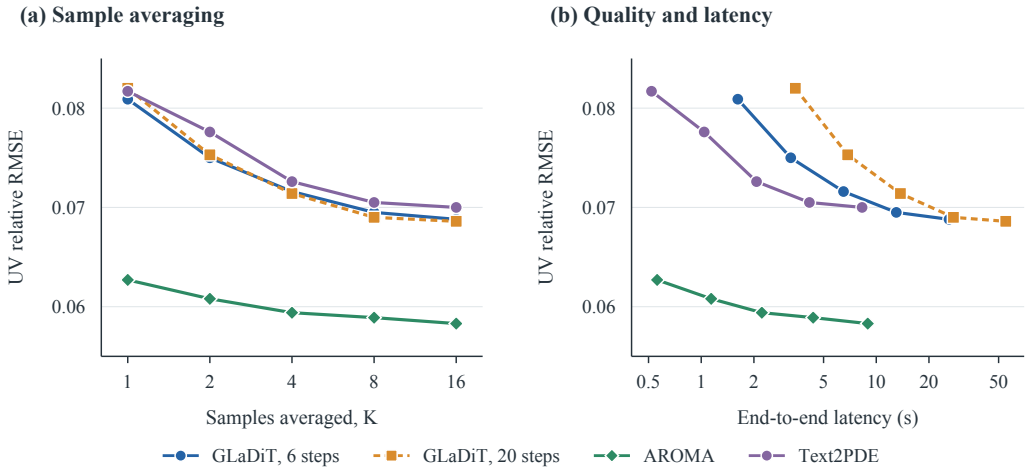


Figure 12: **CylinderFlow predictive accuracy and latency scaling versus ensemble budget K.** Left: relative velocity error across sample budgets. Right: accuracy–latency Pareto frontier.

driven by the graph-native joint spatiotemporal diffusion architecture rather than ensemble averaging. (2) **Consistent accuracy gains**: Physical-space averaging improves velocity, pressure, vorticity, and divergence across all schedules (Table 12), with small sample standard deviation across independent sample pools ($< 3 \times 10^{-5}$, Table 13). The kinetic-energy error stays at $e_E = 0.0159$ for every S and K, reflecting the deterministic autoencoder reconstruction offset rather than sampling behavior.

E.4 Sparse Attention Kernel Implementations

Motivation. To assess practical deployment efficiency, we compare our custom graph-neighborhood Gather-SDPA kernel against PyTorch's block-sparse FlexAttention kernel on one NVIDIA RTX 5090 GPU (batch size 1, $N_\ell = 156$, $T = 64$).

Table 14: Benchmark of sparse attention kernel implementations on an RTX 5090 GPU.

Kernel Implementation	Time per update (s) ↓	Peak Training Memory (GiB) ↓
Neighborhood Gather-SDPA	**0.2202**	17.14
Block-Sparse FlexAttention	0.2411	**12.47**

Empirical analysis. As reported in Table 14, neighborhood Gather-SDPA achieves faster wall-clock execution (0.220 s vs 0.241 s), while FlexAttention reduces peak activation memory by 27.2% (12.47 GiB vs 17.14 GiB), offering complementary options for speed-optimized and memory-constrained training regimes.

F BENCHMARK EVALUATIONS AND FLOW VISUALIZATIONS

This appendix presents detailed numerical diagnostics, temporal evolution analyses, multi-seed training stability results, and spatiotemporal flow field visualizations on the transonic Airfoil and periodic CylinderFlow benchmarks.

F.1 AIRFOIL BENCHMARK DETAILED DIAGNOSTICS

Framewise accuracy over forecast horizon. Table 15 provides the temporal progression of relative velocity errors (e_{UV}) evaluated at future frames 24, 48, and 64 ($t = 0.0384\,\mathrm{s}, 0.0768\,\mathrm{s}, 0.1024\,\mathrm{s}$) on Validation100. While autoregressive baselines (MGN and EAGLE) suffer from compounding rollout error as forecast time increases, GLaDiT maintains low and stable prediction error throughout the entire physical horizon.

Table 15: Airfoil framewise relative velocity error ($e_{\mathrm{UV}} \downarrow$) on Validation100 across physical forecast times.

Method	Frame 24 ($t = 0.0384\,\mathrm{s}$)	Frame 48 ($t = 0.0768\,\mathrm{s}$)	Frame 64 ($t = 0.1024\,\mathrm{s}$)
MGN	0.0285	0.0555	0.0950
EAGLE	0.0186	0.0241	0.0258
AROMA	<u>0.0149</u>	<u>0.0182</u>	<u>0.0192</u>
Text2PDE	0.0318	0.0318	0.0327
GLaDiT	**0.0122**	**0.0152**	**0.0160**

Physical field and surface diagnostics. Table 16 reports velocity correlation, wing-surface velocity RMSE, vorticity RMSE, divergence RMSE, and kinetic energy error across all comparison models. GLaDiT achieves the highest velocity field correlation (0.9998) and lowest derivative errors (vorticity $8.14\,\mathrm{s}^{-1}$, divergence $3.28\,\mathrm{s}^{-1}$). AROMA achieves the lowest wing-surface error ($8.58\,\mathrm{m/s}$) and kinetic energy error (0.0038), closely followed by GLaDiT on surface velocity ($8.835\,\mathrm{m/s}$). For GLaDiT, the kinetic-energy error ($e_E = 0.0159$) is dominated by the frozen graph autoencoder: encoding and decoding the reference fields directly yields $e_E = 0.0159$ due to a near-uniform freestream speed deficit (mean speed error $-1.45\,\mathrm{m/s}$ over the domain, whose far field covers $\approx 75\%$ of the area and dominates the energy integral). Because every sample passes through the same frozen decoder, this energy offset is deterministic and invariant to sampling steps S, sample budget K, and training seeds, while GLaDiT continues to capture the temporal evolution of kinetic energy more faithfully than AROMA (temporal correlation 0.41 vs. 0.16).

Table 16: Airfoil physical diagnostics on Validation100. ρ_{UV} denotes velocity correlation (dimensionless); e_{surf} is wing-surface velocity RMSE ($\mathrm{m\,s}^{-1}$); $\omega_{\mathrm{RMSE}}, d_{\mathrm{RMSE}}$ are vorticity and divergence RMSEs (s^{-1}); e_E is relative kinetic-energy error.

Method	$\rho_{\mathrm{UV}} \uparrow$	$e_{\mathrm{surf}} \downarrow$	$\omega_{\mathrm{RMSE}} \downarrow$	$d_{\mathrm{RMSE}} \downarrow$	$e_E \downarrow$
MGN	0.9971	86.051	76.7	71.2	0.0255
EAGLE	0.9992	40.525	22.6	11.1	<u>0.0052</u>
AROMA	<u>0.9995</u>	**8.582**	<u>9.9</u>	<u>5.6</u>	**0.0038**
Text2PDE	0.9990	71.616	27.8	16.8	0.0297
GLaDiT	**0.9998**	<u>8.835</u>	**8.14**	**3.28**	0.0159

Training stability across seeds and compute costs. Table 17 evaluates GLaDiT across two independent training seeds (seed 0 with EMA 0.999 and seed 1 with EMA 0.9999) at the 200k update checkpoint. Both seeds demonstrate tight reproducibility across velocity, pressure, and derivative metrics. Table 19 provides the detailed breakdown of representation and denoiser training times alongside single-H20 inference latencies.

F.2 CYLINDERFLOW BENCHMARK DETAILED DIAGNOSTICS

Prediction accuracy and derived statistics. Table 20 summarizes full-horizon predictive accuracy on 100 validation trajectories for CylinderFlow. GLaDiT achieves the lowest enstrophy relative

Table 17: Airfoil GLaDiT physical diagnostics across independent training seeds at 200k updates (Validation100, $S = 6, K = 8$). Pressure RMSE (p_{RMSE}) in Pa; derivative RMSEs in s^{-1}; surface UV in $\mathrm{m\,s}^{-1}$.

Metric	Seed 0 (EMA 0.999)	Seed 1 (EMA 0.9999)
Relative velocity error ($e_{\mathrm{UV}} \downarrow$)	**0.01371**	**0.01373**
Gauge-adjusted pressure RMSE ($p_{\mathrm{RMSE}} \downarrow$)	189.0	189.5
Vorticity RMSE ($\omega_{\mathrm{RMSE}} \downarrow$)	8.14	8.13
Divergence RMSE ($d_{\mathrm{RMSE}} \downarrow$)	3.28	3.27
Velocity correlation ($\rho_{\mathrm{UV}} \uparrow$)	0.9998	0.9998
Wing-surface UV RMSE ($e_{\mathrm{surf}} \downarrow$)	8.835	8.747
Kinetic energy error ($e_E \downarrow$)	0.0159	0.0159
Enstrophy error ($e_\Omega \downarrow$)	0.1414	0.1438

Table 18: Validation100 relative velocity errors ($e_{\mathrm{UV}} \downarrow$) across independent training seeds.

Benchmark	Method	Seeds	Relative Velocity Errors (e_{UV})
Airfoil	MGN	2	0.0470, 0.0564
	EAGLE	1	0.0201
	AROMA	1	<u>0.0163</u>
	Text2PDE	1	0.0310
	GLaDiT ($S = 6, K = 8$)	2	**0.01371**, 0.01373
CylinderFlow	MGN	2	**0.0612**, 0.0545
	EAGLE	1	0.0914
	AROMA	1	<u>0.0634</u>
	Text2PDE	2	0.0828, 0.0820
	GLaDiT ($S = 6, K = 8$)	3	0.0843, 0.0695, 0.0811 (mean: 0.0783)

Table 19: Airfoil training compute costs and single-H20 inference latency. Parentheses indicate total GPU-hours across devices. † denotes projected 250k budget.

Method	To Checkpoint (h)	Full Budget (h)	Latency (s) ↓
MGN (4×L20)	0.2 (0.8 GPU-h)	5.7 (22.8 GPU-h)	**0.660**
EAGLE (4×L20)	10.3 (41.2 GPU-h)	10.3 (41.2 GPU-h)	**0.555**
AROMA (4×H20 AE, 4×L20)	12.4 + 32.2 (178 GPU-h)	12.4 + 32.2 (178 GPU-h)	**0.540**
Text2PDE (4×H20)	39.0 + 35.3 (297 GPU-h)	78.0 + 35.3 (453 GPU-h)	0.779
GLaDiT ($S = 6, K = 1$)	36.2 + 61.2 (390 GPU-h)	36.2 + 76.4† (451 GPU-h)	3.84
GLaDiT ($S = 6, K = 8$)	36.2 + 61.2 (390 GPU-h)	36.2 + 76.4† (451 GPU-h)	30.77

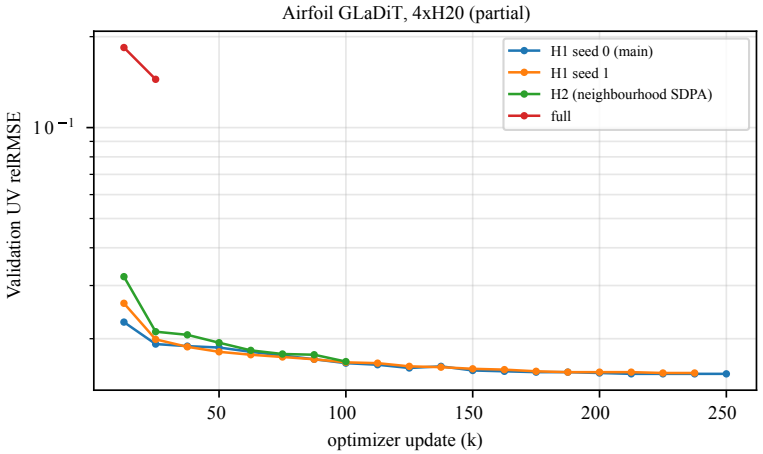


Figure 13: **Airfoil training convergence curves.** Comparison across training seeds and spatial attention configurations on four H20 GPUs.

error (0.0321), a pressure RMSE within 0.0002 of AROMA (0.0366 vs. 0.0364), and the lowest kinetic-energy error among joint generation models. Table 21 further reports framewise relative velocity error across forecast horizons on Validation100.

Table 20: Prediction quality on 100 CylinderFlow validation trajectories across 64 future frames. Pressure RMSE (p_{RMSE}) is gauge-adjusted in dataset units; other metrics are relative errors.

Method	$e_{\mathrm{UV}} \downarrow$	$p_{\mathrm{RMSE}} \downarrow$	$e_E \downarrow$	$e_\Omega \downarrow$
MGN	**0.0612**	0.0373	0.0441	0.0438
EAGLE	0.0914	0.0570	0.0372	0.0555
AROMA	<u>0.0634</u>	**0.0364**	**0.0170**	<u>0.0370</u>
Text2PDE	0.0828	0.0538	0.0299	0.0571
GLaDiT ($S = 6, K = 8$)	0.0725	<u>0.0366</u>	<u>0.0210</u>	**0.0321**

Table 21: CylinderFlow framewise relative velocity error ($e_{\mathrm{UV}} \downarrow$) on Validation100.

Method	Frame 24 ($t = 1.92$)	Frame 48 ($t = 3.84$)	Frame 64 ($t = 5.12$)
MGN	**0.0477**	**0.0734**	0.0907
EAGLE	0.0773	0.1096	0.1189
AROMA	<u>0.0512</u>	<u>0.0743</u>	<u>0.0841</u>
Text2PDE	0.0804	0.0893	0.1036
GLaDiT	0.0737	0.0800	**0.0815**

Table 22: CylinderFlow model parameter count and peak allocated inference GPU memory on one H20 GPU.

Method	Parameters (M)	Peak Inference Memory (MiB) $\downarrow$
MGN	<u>2.33</u>	<u>86</u>
EAGLE	10.32	254
AROMA	**2.00**	**59**
Text2PDE	198.20	10,607
GLaDiT	177.37	990

Table 23: CylinderFlow baseline training cost breakdown. Parentheses indicate total GPU-hours.

Method	To evaluated checkpoint	Full budget
MGN (4×H20)	4.90 h (19.6 GPU-h)	6.44 h (25.8 GPU-h)
EAGLE (4×L20)	5.8 h (23.2 GPU-h)	7.3 h (29.2 GPU-h)
AROMA (4×H20 AE, 4×L20)	14.9 + 27.5 h (170 GPU-h)	14.9 + 32.4 h (189 GPU-h)
Text2PDE (4×H20)	4.5 + 21.9 h (106 GPU-h)	9.1 + 21.9 h (124 GPU-h)
GLaDiT (1×RTX 5090)	23.45 + 110.6 h (134 GPU-h)	23.45 + 110.6 h (134 GPU-h)

Table 24: CylinderFlow GLaDiT validation velocity error across training seeds on the 4×H20 recipe (batch 4, 250k updates).

Training recipe	Seed	Update	Weights	Validation100 $e_{\mathrm{UV}} \downarrow$
4×H20, batch 4	0	187.5k	Raw	0.0843
	1	162.5k	EMA 0.999	0.0695
	2	187.5k	Raw	0.0811
Mean ± sample standard deviation				0.0783 ± 0.0078

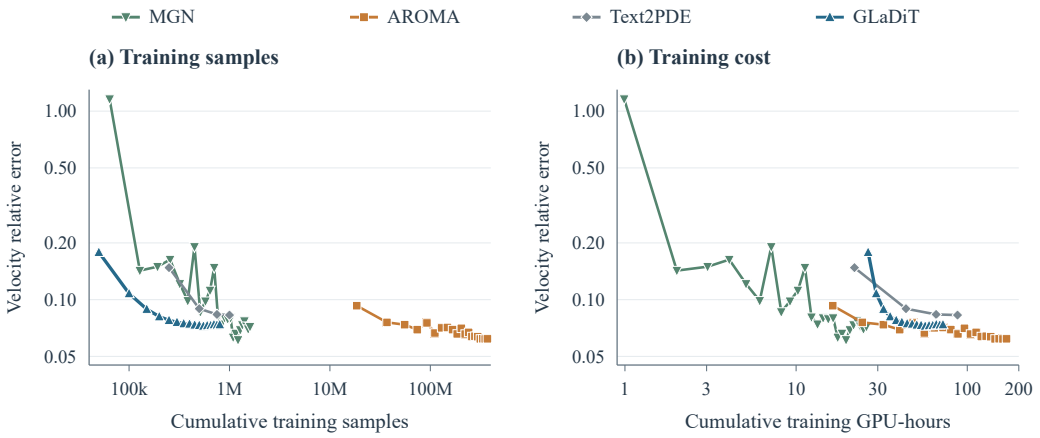


Figure 14: **Training progress and computational cost on CylinderFlow.** (a) Validation velocity error versus cumulative training samples. (b) Validation error versus total GPU-hours. Baseline histories come from the full-length (75-frame) training runs, which evaluated every checkpoint; the 65-frame retrains reported in the tables recorded only their final checkpoints.

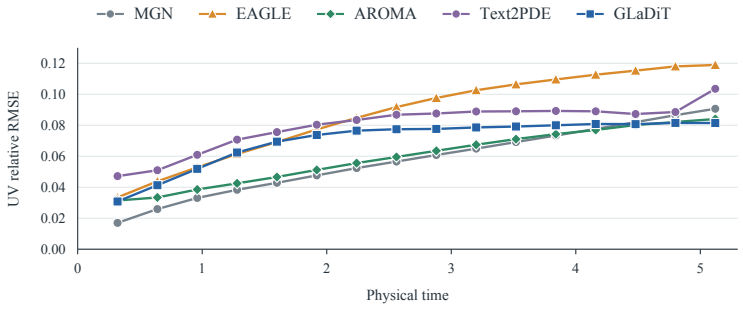


Figure 15: **CylinderFlow velocity error over physical time.** Markers indicate trajectory-mean errors on Validation100 at every fourth stored frame (frames 4–64).

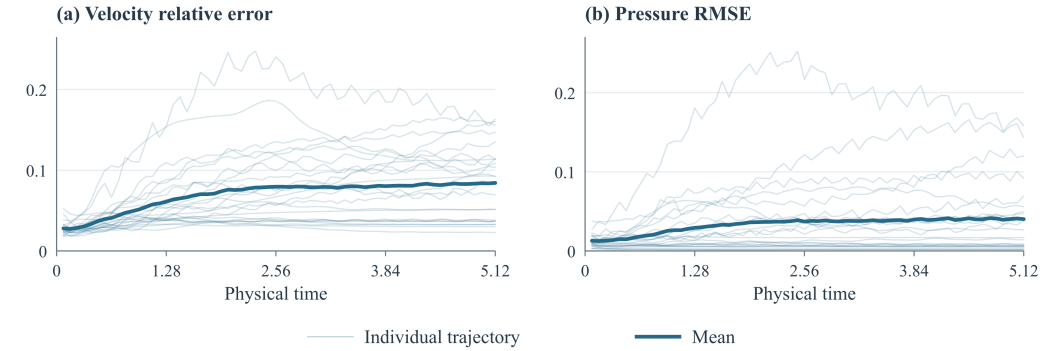


Figure 16: **CylinderFlow per-trajectory temporal error evolution.** Thin lines show individual validation trajectories; the solid curve indicates the mean.

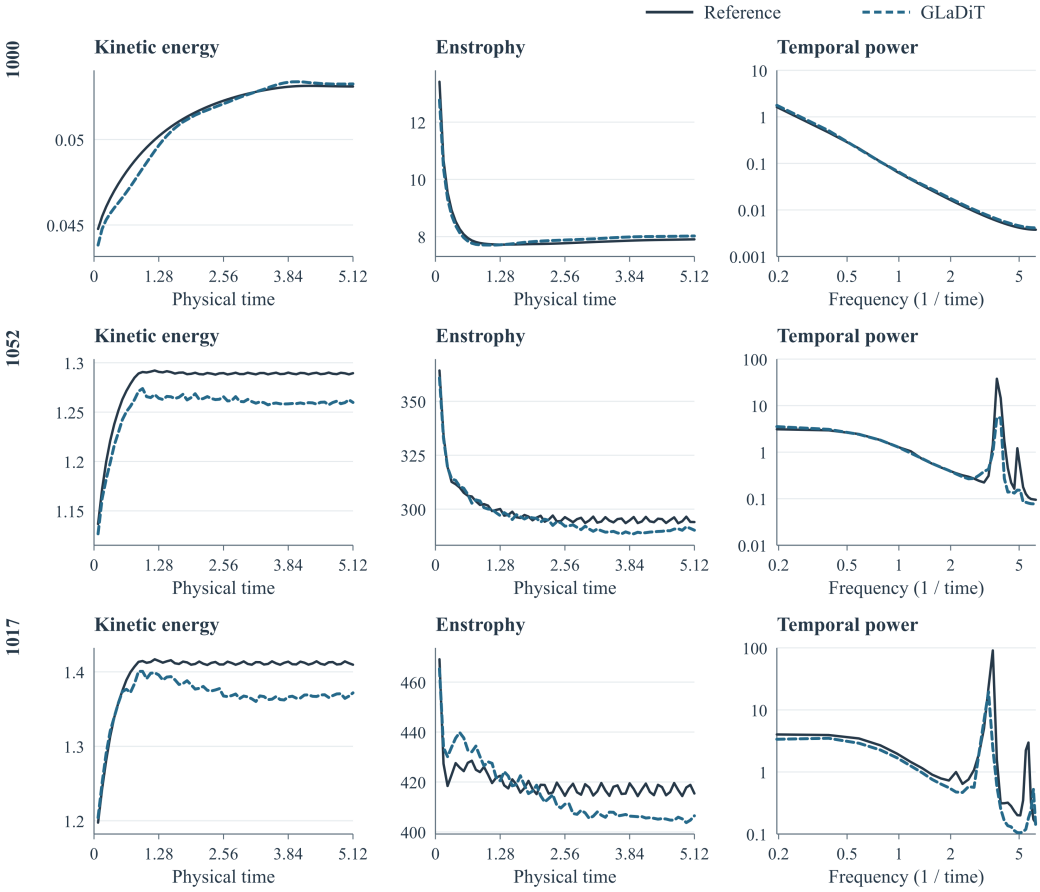


Figure 17: **CylinderFlow physical invariants and temporal velocity power spectra.** Reference (solid) and predicted (dashed) kinetic energy, enstrophy, and Fourier power spectra across three sample validation trajectories (1000, 1052, 1017).

F.3 SPATIOTEMPORAL FLOW VISUALIZATIONS

Figures 18–20 visualize reference and predicted physical fields alongside absolute error maps on the Airfoil benchmark at physical times $t = 0.0256\,\mathrm{s}$ (frame 16) and $t = 0.1024\,\mathrm{s}$ (frame 64). Figures 21–24 provide qualitative flow field comparisons for CylinderFlow vortex shedding across early, intermediate, and late simulation horizons.

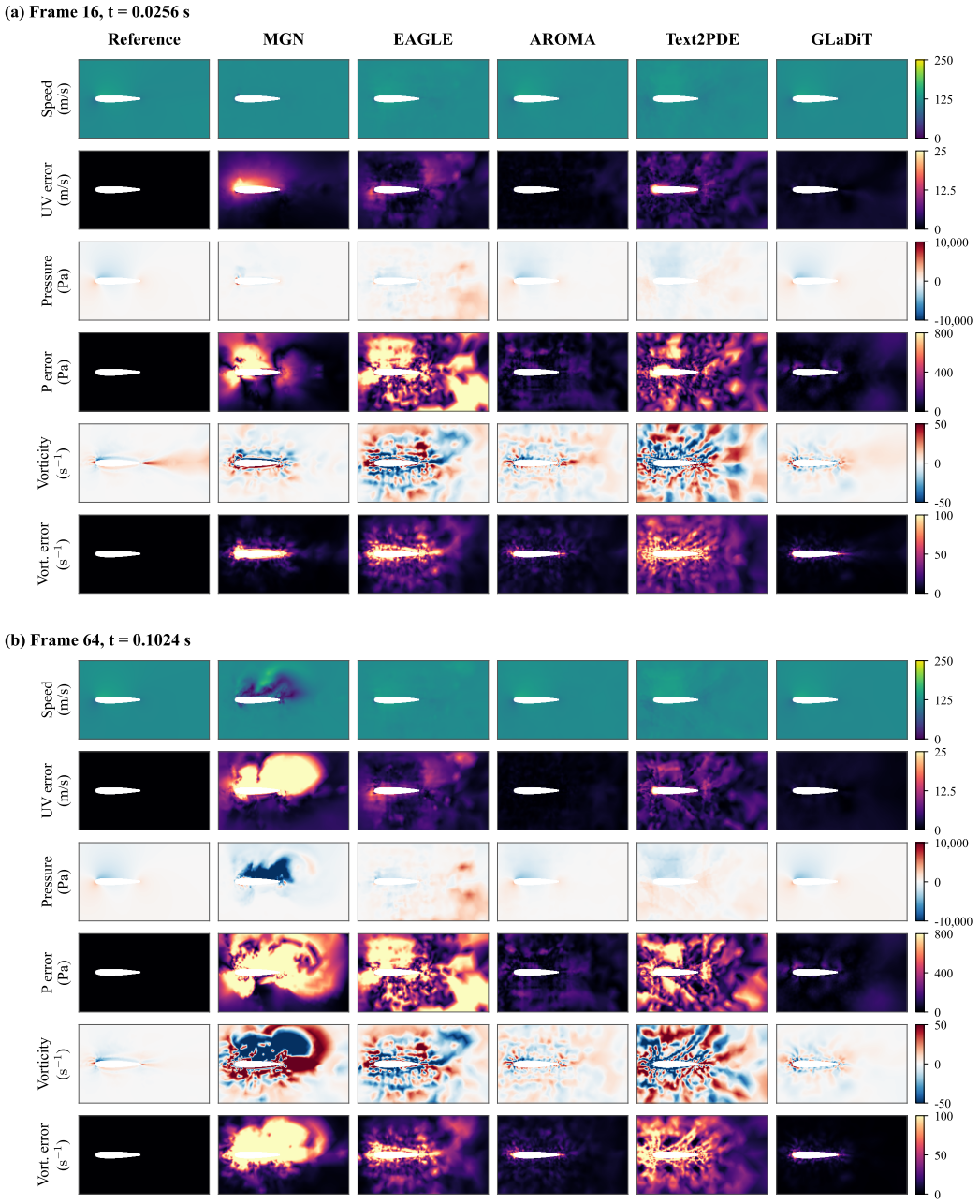


Figure 18: **Airfoil flow fields and error maps on Trajectory 1095 ($t = 0.0256\,\mathrm{s}$ and $t = 0.1024\,\mathrm{s}$).** Rows display velocity magnitude, gauge-adjusted pressure, and vorticity alongside their absolute error fields.

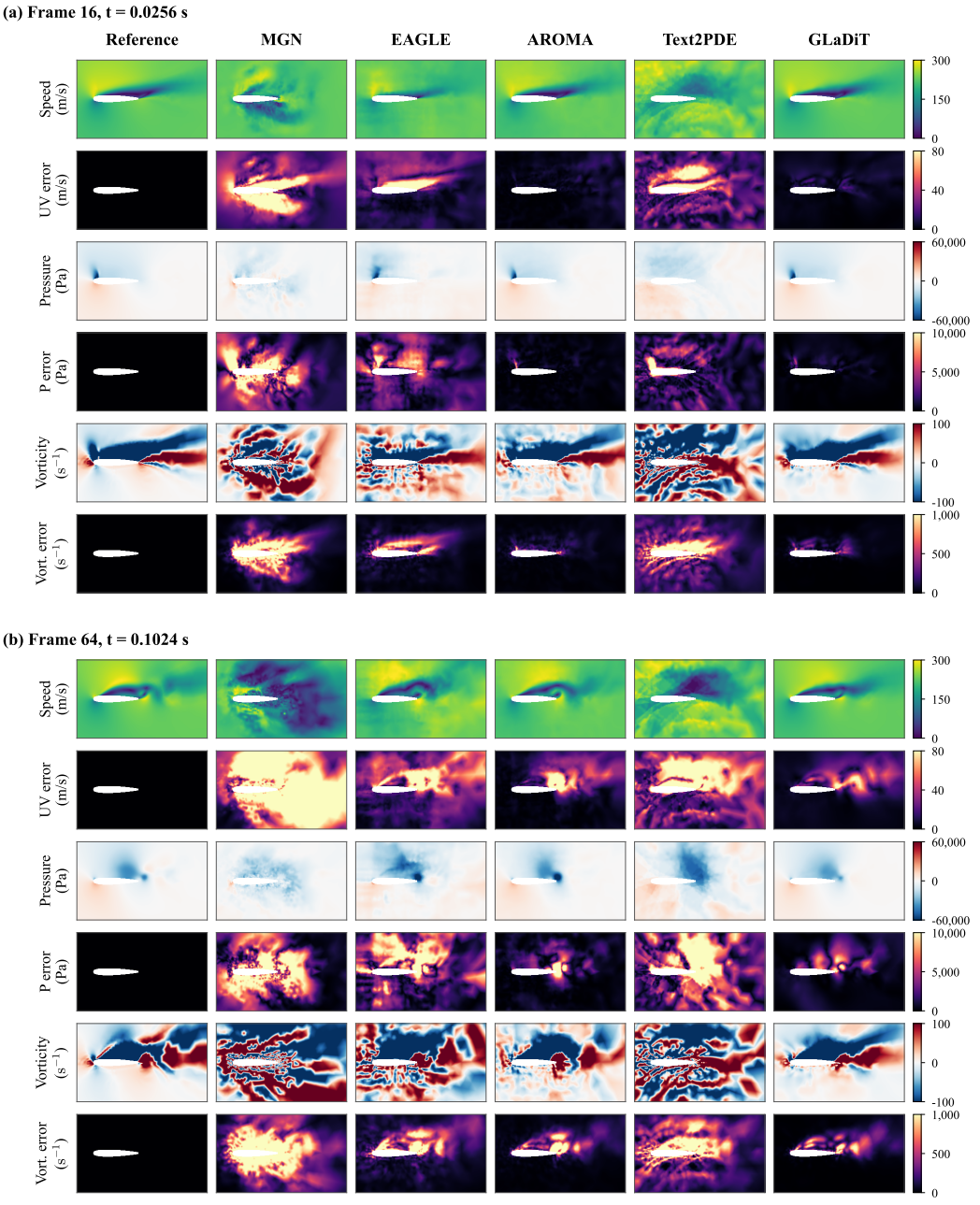


Figure 19: **Airfoil flow fields and error maps on Trajectory 1013** ($t = 0.0256$ **s and** $t = 0.1024$ **s).**

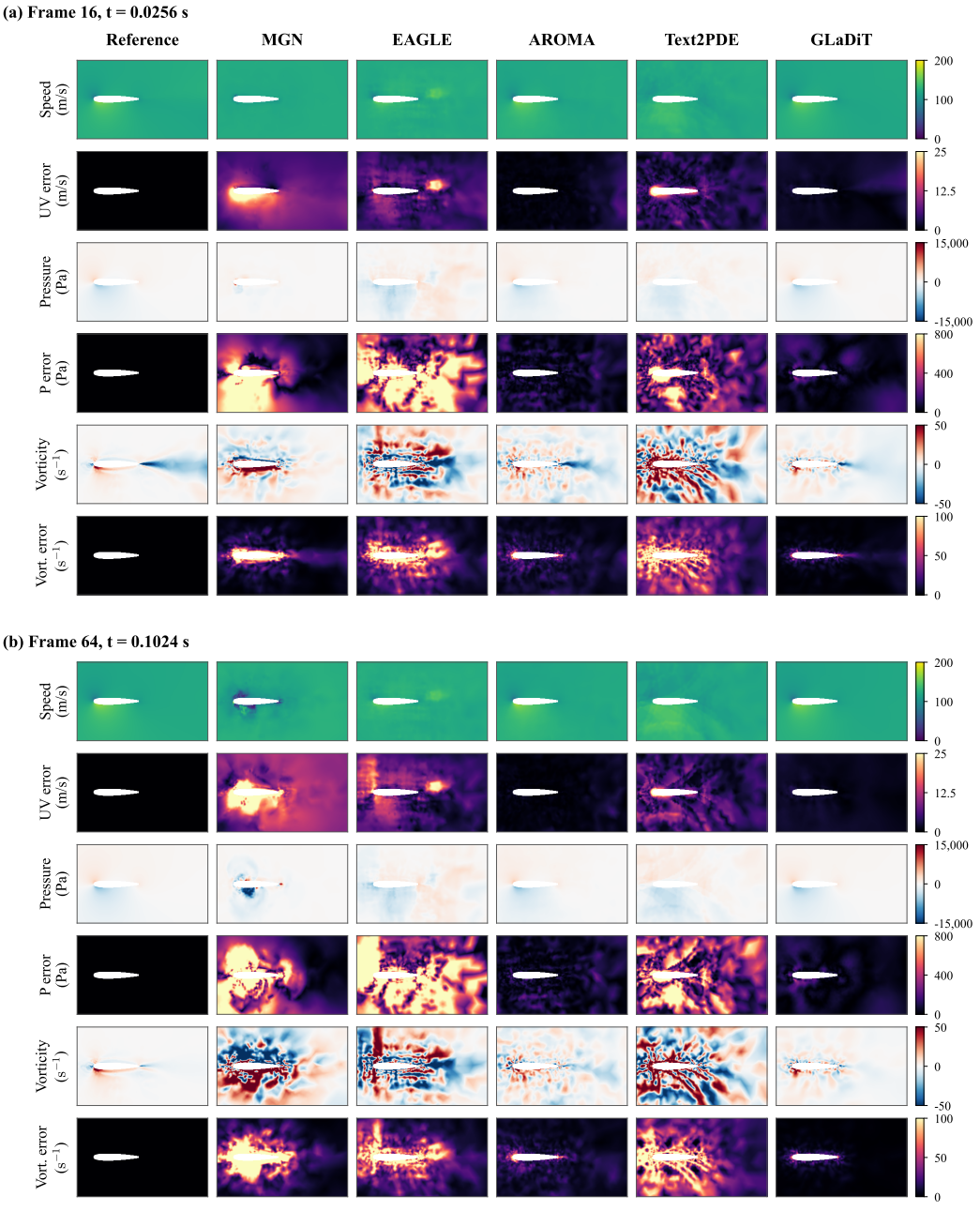


Figure 20: **Airfoil flow fields and error maps on Trajectory 1086 ($t = 0.0256$ s and $t = 0.1024$ s).**

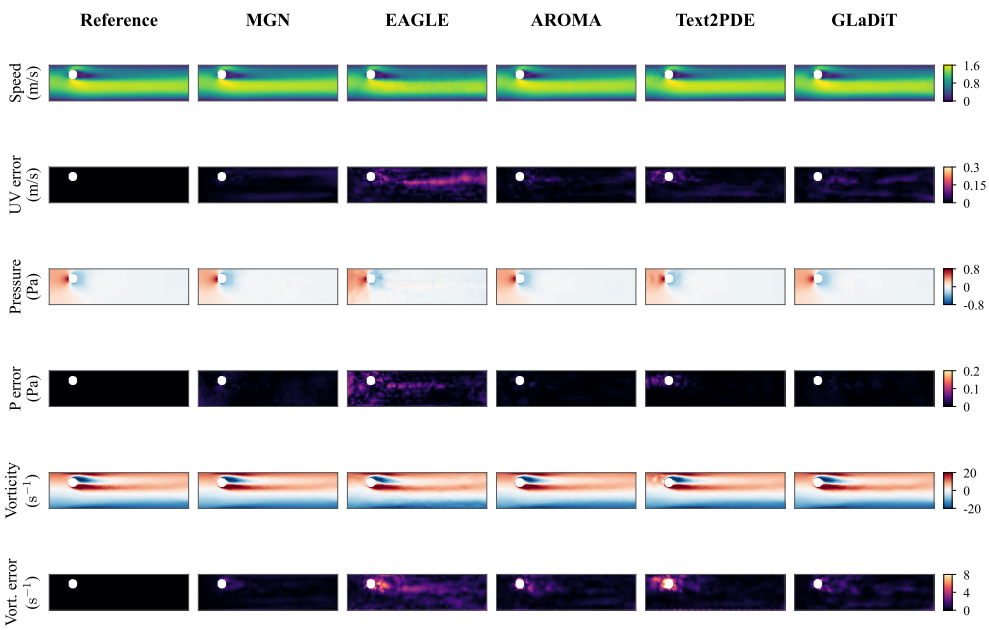


Figure 21: **CylinderFlow vortex shedding flow field comparisons across models.**

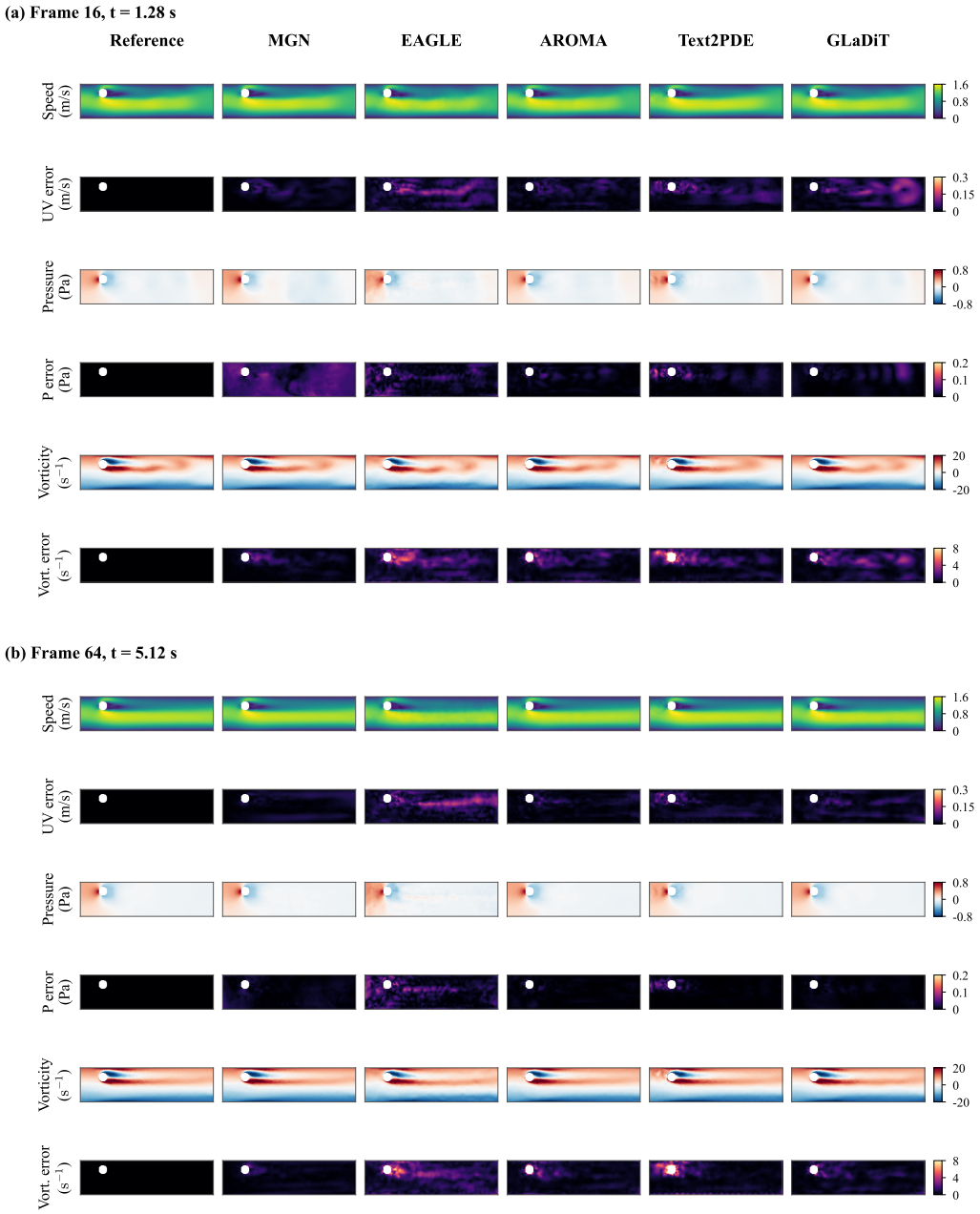


Figure 22: **CylinderFlow flow fields and error maps on Trajectory 1095.**

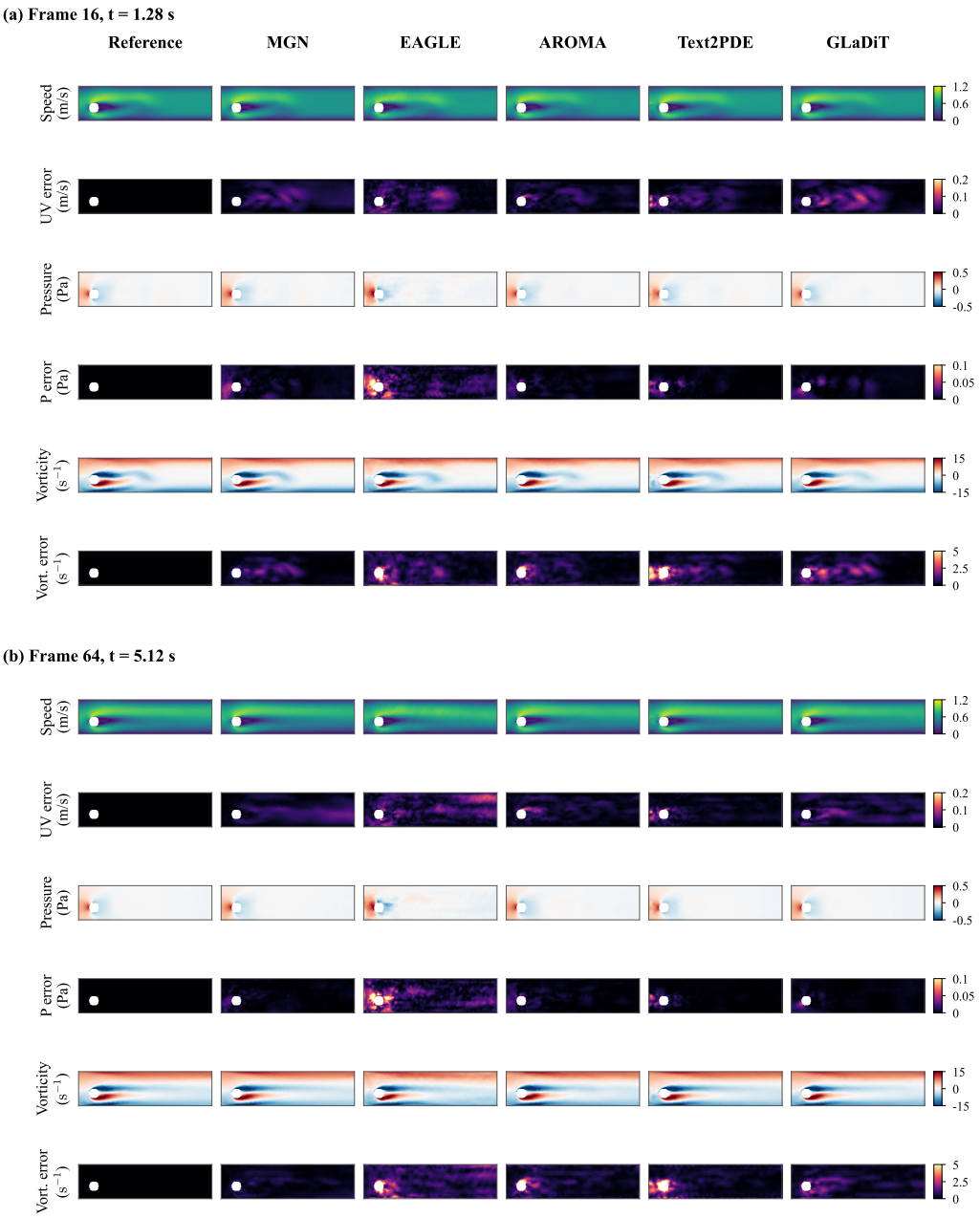


Figure 23: **CylinderFlow flow fields and error maps on Trajectory 1013.**

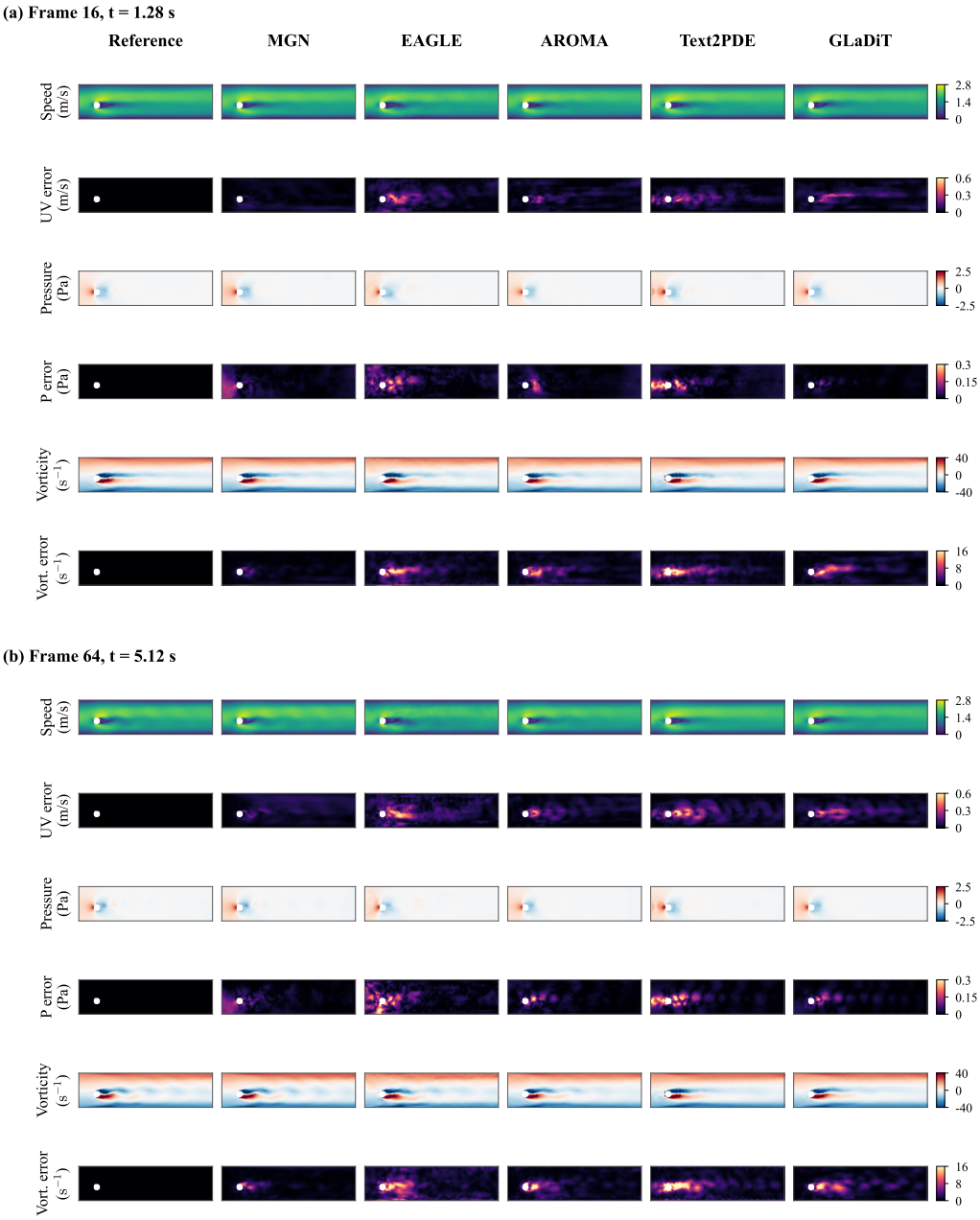


Figure 24: **CylinderFlow flow fields and error maps on Trajectory 1086.**