COMPOSABLE SPATIAL SEQUENCE LEARNING FOR STEADY FLOW PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

There is current interest in neural surrogates for steady flow-field solutions to partial differential equations (PDEs). These surrogates are typically trained as global mappings from geometry and operating conditions to complete solution fields, but extrapolation to domains with substantially greater spatial extent remains challenging. This study investigates whether reformulating the learning problem as repeated local transitions can improve domain-length extrapolation, without increasing the number of model parameters or changing the model architecture. Chip-to-chip (C2C) learning represents a steady velocity-magnitude field as an ordered sequence of narrow slices taken perpendicular to the streamwise direction, termed chips, and learns the geometry-conditioned transitions from upstream chip states to downstream chip states. Repeated composition of such transitions allows the domain length to change through rollout depth rather than model architecture. The proposed C2C-MLP is compared with five non-C2C baselines: a Multilayer Perceptron (MLP), a Physics-Informed Neural Network (PINN), a Graph Neural Operator (GNO), a General Neural Operator Transformer (GNOT), and Transolver, on domains of one, two, and three times the training length and on four out-of-distribution (OOD) geometries. All baseline models are provided with the same initial velocity-magnitude profile and geometric information available to C2C. Across five independent training runs, the C2C-MLP achieves the lowest mean absolute error (MAE) across all three domain lengths; on the $3\times$ domain, its MAE was $0.0679\ \mathrm{m/s}$, a 39.5% reduction relative to the most accurate baseline. OOD performance is geometry-dependent: C2C-MLP achieved the lowest MAEs on the Ahmed body and airfoil cases, while its MAE remained within 5.4% of the best baseline on the periodic-hill and backward-facing step cases. These four geometries are widely used CFD benchmark configurations representing bluff-body, aerodynamic, and separated-flow problems. Overall, the results of this study highlight spatial composition as a promising strategy for surrogate models that must operate on domains extending beyond those represented during training, without retraining or changing the model architecture.

1 INTRODUCTION

Computational fluid dynamics (CFD) provides detailed flow-field solutions by numerically solving governing partial differential equations (PDEs) on discretised domains Ferziger et al. (2020). Machine-learning surrogates can reduce the cost of repeated flow evaluations by learning mappings from geometry and flow conditions to solution fields Guo et al. (2016); Bhatnagar et al. (2019); Thuerey et al. (2020), supporting applications such as optimisation, uncertainty analysis, and digital twins Vinuesa & Brunton (2022). A persistent challenge is extrapolation when the spatial extent or geometry of a test domain extends beyond the range represented by the training cases. Among different forms of domain-scale variation, length generalisation is practically important in applications such as vascular-flow modelling, pipeline analysis, and cooling-channel design, where geometrically similar flow passages may differ considerably in their streamwise extent. A surrogate that remains applicable as domain length changes can therefore be reused without retraining or architectural modification.

Neural flow surrogates employ several spatial representations. Pointwise networks predict local quantities from coordinates and geometric descriptors (Leer & Kempf, 2021), while physics-

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.

informed methods introduce governing-equation constraints (Raissi et al., 2019; Karniadakis et al., 2021; Jin et al., 2021). Graph-based models and neural operators learn interactions across spatial locations and field-level mappings (Pfaff et al., 2021; Li et al., 2020b;c; Kovachki et al., 2023). Geometry-conditioned operators such as Geo-FNO and GINO extend operator learning to varying domains (Li et al., 2023a;b), while GNOT and Transolver use attention-based mechanisms to model geometric and latent spatial interactions (Hao et al., 2023; Wu et al., 2024).

Spatial partitioning provides another way to organise large spatial problems. Vision Transformer (ViT) represents fixed image patches as tokens for global self-attention (Dosovitskiy et al., 2021), while neural domain-decomposition methods such as DeepDDM, cPINNs, XPINNs, and SNAP-DDM divide PDE domains into coupled local subproblems (Li et al., 2020a; Jagtap et al., 2020; Jagtap & Karniadakis, 2020; Mao et al., 2024). These approaches use spatial partitioning either to construct spatial representations or to couple local solutions within a prescribed global problem.

However, flexible spatial representation or domain decomposition does not by itself address extrapolation to substantially greater domain extent. Conventional surrogates still typically learn pointwise or field-level mappings tied to the spatial scales represented during training. This motivates a different factorisation of the prediction problem: representing a solution as a composition of reusable local mappings, so that increasing domain extent changes the number of compositions rather than the structure of the learned mapping. The central hypothesis of this study is that such spatial compositionality encourages the model to learn reusable local relationships, thereby improving domain-length extrapolation.

C2C implements this idea by representing the flow domain as an ordered sequence of streamwise chips, and learning the transition from an upstream velocity state and local geometry to a downstream state (Figure 1). Transitions can be composed along the flow direction, allowing flow prediction of domains of greater spatial extent than those learned, without the need for retraining or modifying the architecture of the model.

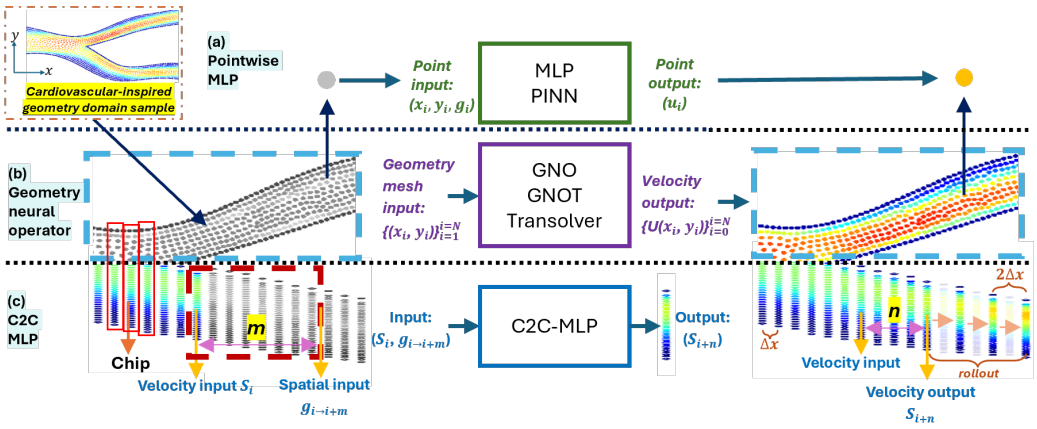

Figure 1: Surrogate formulations compared in this work. (a) MLP and PINN use pointwise geometric inputs. (b) GNO, GNOT, and Transolver process field-level spatial information. (c) C2C-MLP predicts a downstream velocity-magnitude state from an upstream chip state and local geometry, then reuses the same transition during spatial rollout.

The contributions of this study are threefold. First, steady flow prediction is reformulated as a repeated geometry-conditioned local-mapping learning problem, decoupling the learned mapping from overall domain length and enabling longer-domain prediction without retraining or architectural modification. Second, under matching initial-state conditioning, the proposed local-transition formulation yields consistently more accurate domain-length extrapolation than pointwise, graph-based, and attention-based global surrogates, despite using a simple MLP transition model. Third, CFD-forced prediction, in which the ground-truth upstream state from CFD is supplied at each step, is compared with free rollout, in which the model's own predicted upstream state is recursively reused for subsequent predictions. Together with a target-offset sensitivity analysis, this examines how the position of the target chip within the local geometric context affects prediction error. These analyses provide insight into the main sources of error in long-range C2C prediction.

2 PROBLEM FORMULATION AND C2C LEARNING

2.1 FLOW-FIELD AND CHIP REPRESENTATION

This study considers steady, two-dimensional through-flow in a fluid domain Ω with specified geometry, boundary conditions, and physical parameters. The prediction target $U(x, y)$ is the velocity magnitude, with reference values obtained from CFD simulations.

The domain is divided along the streamwise direction into L ordered slices called 'chips', $C_1, \ldots, C_L$, each with thickness Δx. CFD velocity magnitudes are interpolated onto c sampling points along the y-direction within each chip (Figure 1). The velocity state $\mathbf{s}_i$ and local geometry input $\mathbf{g}_i$ are

$$\mathbf{s}_i = \begin{bmatrix} U_{i,1} \\ U_{i,2} \\ \vdots \\ U_{i,c} \end{bmatrix}, \qquad \mathbf{g}_i = \begin{bmatrix} g_{i,1} & g_{i+1,1} & \cdots & g_{i+m,1} \\ g_{i,2} & g_{i+1,2} & \cdots & g_{i+m,2} \\ \vdots & \vdots & \ddots & \vdots \\ g_{i,c} & g_{i+1,c} & \cdots & g_{i+m,c} \end{bmatrix}, \tag{1}$$

where $U_{i,p}$ is the velocity magnitude at sampling point p in chip C_i, and $g_{j,p} = (x_{j,p}, y_{j,p})$ denotes the coordinates of the p-th point in chip j, with $x_{j,p}$ and $y_{j,p}$ representing the streamwise and transverse coordinates. The geometry matrix contains c rows (one per sampling point) and $m+1$ columns (one per chip from C_i to C_{i+m}), and contains geometric information only, with no downstream CFD velocity values.

2.2 CONDITIONAL TRANSITION LEARNING AND SPATIAL ROLLOUT

C2C factorises full-domain prediction into repeated predictions of local transitions. During supervised training,

$$\hat{\mathbf{s}}_{i+n} = f_\theta\big(\mathbf{s}_i^{\mathrm{CFD}}, \mathbf{g}_i\big), \tag{2}$$

where f_θ is a single MLP transition model that is repeatedly queried and trained for all chip positions and training geometries. The C2C configuration is summarised in Table 1.

Table 1: C2C configuration. The main model uses $n = 5$; prediction-offset sensitivity is evaluated separately.

Symbol	Meaning	Setting
c	Speed samples per chip	66
n	Prediction offset	5 (main); 1, 3, 5, 7, 9
m	Geometry downstream offset	10 chips
$m+1$	Chips in geometry context	11
Δx	Streamwise chip thickness	0.001 m
L_{train}	Chips per training geometry	120

The main prediction offset is $n = 5$, placing target chip C_{i+5} at the centre of the 11-chip geometry window. Thus, the prediction is conditioned on the surrounding geometry while using only the upstream velocity state as flow information. Training input-target pairs are extracted by sliding the geometry window along each training case. Each pair defines an independent chip-level transition $(\mathbf{s}_i^{\mathrm{CFD}}, \mathbf{g}_i) \rightarrow \mathbf{s}_{i+n}^{\mathrm{CFD}}$, and the C2C model is trained using mini-batches of such transition samples. For a transition from C_i to C_{i+n}, the loss is calculated as follows:

$$\mathcal{L}_i = \frac{1}{c} \sum_{p=1}^{c} \left(\hat{U}_{i+n,p} - U_{i+n,p}^{\mathrm{CFD}} \right)^2, \tag{3}$$

where p indexes the c sampling points within the target chip. The model parameters are updated by minimising the mean transition loss within each mini-batch. Mini-batch construction and other training settings, including the optimiser, learning rate, batch size, and number of training epochs, are provided in Supplementary Material S1.

C2C is a state-conditioned continuation model and therefore requires an upstream state from which rollout begins in the prediction phase. The initial profile $\mathbf{s}_0^{\mathrm{CFD}}$ is the prescribed inlet velocity bound-

ary condition at C_0, rather than a downstream state extracted from the computed CFD solution. Because the main offset is $n = 5$, five interleaved rollout chains are required. The velocity-magnitude variation across $C_0, \ldots, C_4$ is below 2%, so the same profile from C_0 is used to initialise all five chains. More generally, the offset sensitivity study uses n chains, all initialised from the same single profile:

$$\hat{\mathbf{s}}_k = \mathbf{s}_0^{\mathrm{CFD}}, \qquad k = 0, \ldots, n-1. \tag{4}$$

Downstream predictions are then generated recursively,

$$\hat{\mathbf{s}}_{i+n} = f_\theta(\hat{\mathbf{s}}_i, \mathbf{g}_i)\,. \tag{5}$$

In the main configuration, each predicted state becomes the upstream input for a target five chips downstream; for a general offset n, it is reused for a target n chips downstream. After initialisation, no additional CFD velocity states are supplied during free rollout. Increasing domain length therefore changes rollout depth rather than f_θ, model size, or input dimensionality. For a domain containing $C_0, \ldots, C_{L-1}$, the unpadded geometry window is complete for $i = 0, \ldots, L-m-1$. For the main setting $m = 10$ and $n = 5$, predictions cover $C_5, \ldots, C_{L-6}$; the five initialisation chips and five unpredicted outlet chips are excluded from the main accuracy evaluation.

2.3 CFD-Reference State Evaluation for Transition Analysis

To separate the local transition error from the error introduced by recursive state reuse, free rollout is compared with CFD-forced evaluation. CFD-forced evaluation supplies the reference upstream state at every transition:

$$\hat{\mathbf{s}}_{i+n}^{\text{CFD-F}} = f_\theta\big(\mathbf{s}_i^{\mathrm{CFD}}, \mathbf{g}_i\big)\,. \tag{6}$$

The same trained model, geometry inputs, and target chips are used in both protocols. CFD-forced evaluation therefore measures the transition accuracy under the reference-state distribution seen during supervised training, whereas free rollout evaluates the same transition under a recursively generated state distribution. Their difference reflects error propagation and accumulation during recursive prediction.

3 Experimental Setup

3.1 Dataset for Training and Testing

The dataset contains 2100 steady-state two-dimensional CFD cases generated from cardiovascular-inspired wall-bounded through-flow geometries, with an example shown in Figure 1 (orange box). Cases share the same prescribed flow conditions, and CFD-predicted velocity magnitudes provide the reference values for training and evaluation.

The length tests use domains of one, two, and three times the training length for all models. An additional $5\times$ extended domain is used only for the C2C model to compare CFD-forced and free rollout, and to evaluate the role of the prediction offset by shifting the target-chip position within the local geometry context. Geometry transfer is evaluated on four OOD geometry families: backward-facing steps (Armaly et al., 1983), periodic hills (Xiao et al., 2020), two-dimensional Ahmed-body-inspired profiles (Ahmed et al., 1984), and airfoils (Bonnet et al., 2022). These references provide background for the selected geometry families. The test geometries and CFD reference solutions used here are generated for the present study, with their construction and flow conditions detailed in Supplementary Material S2. For each domain-length setting and each OOD geometry family, evaluation is performed on ten distinct test geometries.

3.2 Baseline non-C2C Models and Training

Six models are compared: MLP, PINN, GNO, GNOT, Transolver, and C2C-MLP. MLP predicts local quantities from coordinates and geometric descriptors (Leer & Kempf, 2021). PINN uses the same pointwise inputs and adds a mass-conservation regularisation term (Raissi et al., 2019; Jin et al., 2021). GNO models field interactions through graph-based message passing (Li et al., 2020b;c); GNOT uses heterogeneous attention with geometric gating (Hao et al., 2023), while Transolver applies Physics-Attention over learned physical-state slices (Wu et al., 2024).

All models use the same CFD database, data partitions, and test cases, while retaining their model-specific architectures and training objectives. The global baselines are trained for direct point-wise or field-level prediction, whereas C2C is trained on local transitions using reference upstream states $\mathbf{s}_i^{\mathrm{CFD}}$. All models use supervised prediction losses, with PINN additionally including mass-conservation regularisation. Each model is trained independently five times with different random seeds and fixed model-specific hyperparameters, then evaluated on geometries of different shape and length without retraining. Detailed architectures, batch definitions, conditioning implementations, optimisation settings, and the PINN loss are provided in Supplementary Material S1.

Additionally, the C2C offset sensitivity comparison uses free rollout. CFD-forced evaluation is used only as a transition diagnostic and in the prediction-offset study, not as a primary comparison with the global baselines.

3.3 Evaluation Metrics and Computational Cost

Prediction accuracy is measured geometry-wise using the MAE of the velocity magnitude, while chip error plots use NMAE:

$$\mathrm{MAE} = \frac{1}{N}\sum_{p=1}^{N}\left|\hat{U}_p - U_p^{\mathrm{CFD}}\right|, \qquad \mathrm{NMAE}\ (\%) = 100\frac{\mathrm{MAE}}{U_{\max}^{\mathrm{CFD}} - U_{\min}^{\mathrm{CFD}}}. \tag{7}$$

Here, N contains all evaluated points for geometry-level MAE and only the corresponding chip points for chip-wise MAE. MAE is reported in $\mathrm{m/s}$. The same geometry-level CFD reference range is used for every chip and model.

Computational cost is measured as the training time per epoch and the prediction time per case in the same hardware and software environment. Prediction time is measured on the $3\times$ domain and includes all model evaluations required to generate the field, including recursive C2C state updates, while excluding loading, I/O, plotting, and metric calculation. Additional details on the computation of the computational overheads are given in Supplementary Material S3.

4 Results and Discussion

This section compares C2C with the baseline models on domain-length generalisation, prediction-offset sensitivity, OOD geometries, and computational cost, and analyses the effect of recursive state reuse on long-range and OOD geometry prediction.

4.1 Domain-Length Generalisation

Figure 2(a) compares CFD and predicted velocity-magnitude fields on training-like geometry domains of one, two, and three times the training length. MLP and PINN capture the main flow pattern but show local fluctuations; GNO reproduces several upstream features but develops larger downstream discrepancies; GNOT and Transolver perform best among the baselines on the $1\times$ domain, but their predictions deteriorate substantially as the domain length increases, leading to the highest baseline errors on the $2\times$ and $3\times$ domains. C2C-MLP achieves the lowest error at the training-domain length and continues to outperform all baselines as the domain is extended.

The chip-wise NMAE curves in Figure 2(b) show lower C2C error over much of the evaluated region, although local peaks remain. Unlike C2C, the global baselines do not recursively reuse upstream predictions, so their spatial error variation should not be interpreted as autoregressive accumulation.

Table 2 reports C2C-MLP mean MAEs of 0.0490, 0.0599, and $0.0679\ \mathrm{m/s}$ on the $1\times$, $2\times$, and $3\times$ domains, respectively, the lowest among the evaluated models at each length. On the $3\times$ domain, C2C-MLP reduces mean MAE by 39.5% relative to MLP, the strongest baseline, and by 53.9% relative to GNO, the strongest field-level baseline in this test.

Because all baselines receive the same single initial velocity profile available to C2C, the length-extrapolation advantage cannot be explained solely by unequal initial-state information. The result is consistent with the proposed factorisation: C2C reuses the same local transition as domain length

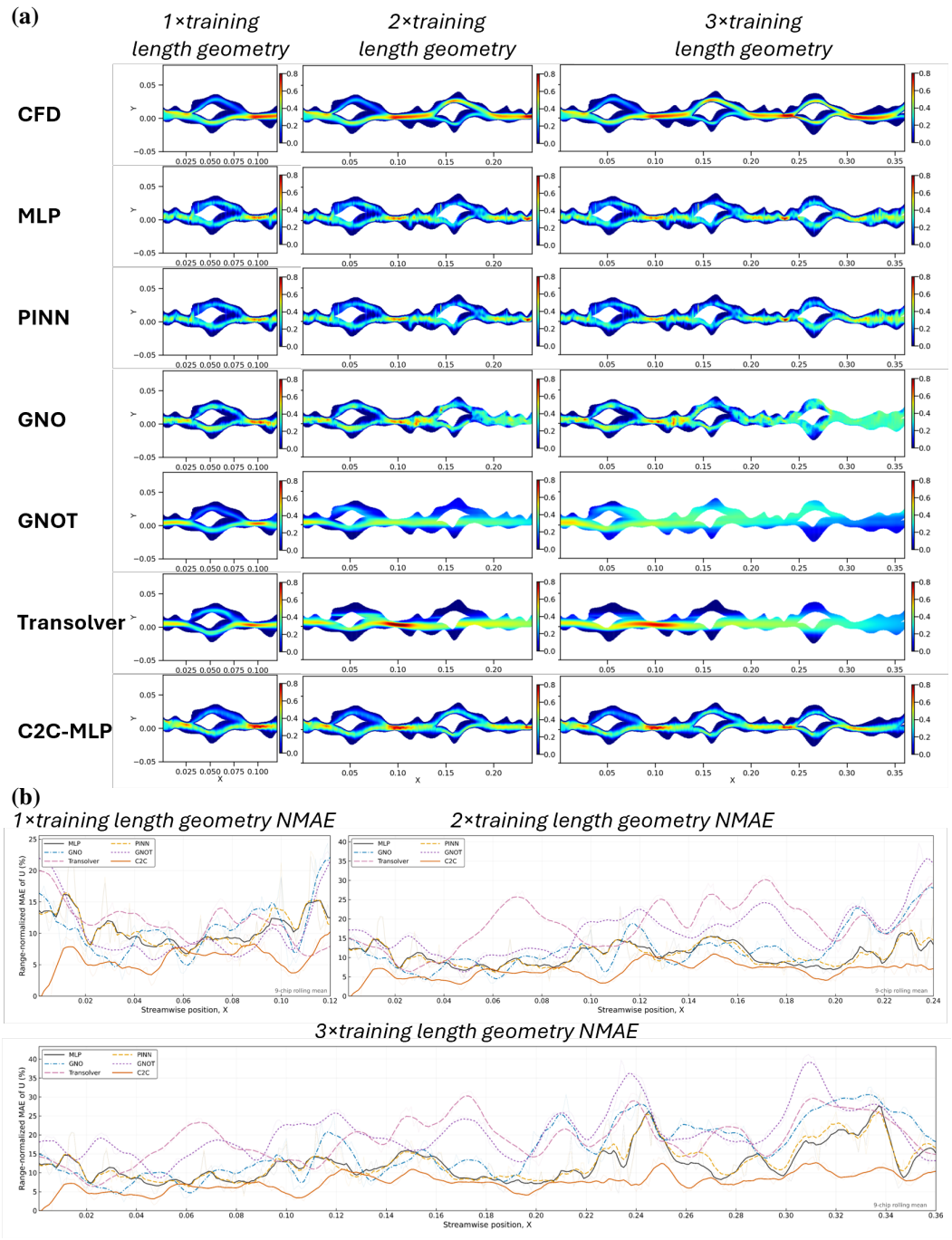

Figure 2: Prediction across domain lengths. (a) CFD reference and velocity-magnitude predictions on $1\times$, $2\times$, and $3\times$ training-length domains. Rows show CFD, MLP, PINN, GNO, GNOT, Transolver, and C2C-MLP. (b) Corresponding chip-wise range-normalised MAE of U. Faint curves show the raw chip-wise NMAE values, while prominent curves show a centred nine-chip rolling mean, obtained by averaging each chip with the four neighbouring chips on either side.

increases. Its error nevertheless rises with domain length, showing that predictions while improved, do not eliminate degradation under deeper rollout.

4.2 PREDICTION-OFFSET SENSITIVITY

Figure 3 compares CFD-forced evaluation and free rollout on the $5\times$ extended domain for $n \in \{1, 3, 5, 7, 9\}$. Across all offsets, CFD-forced evaluation gives lower error, indicating that predicted-state reuse is a major source of long-range error.

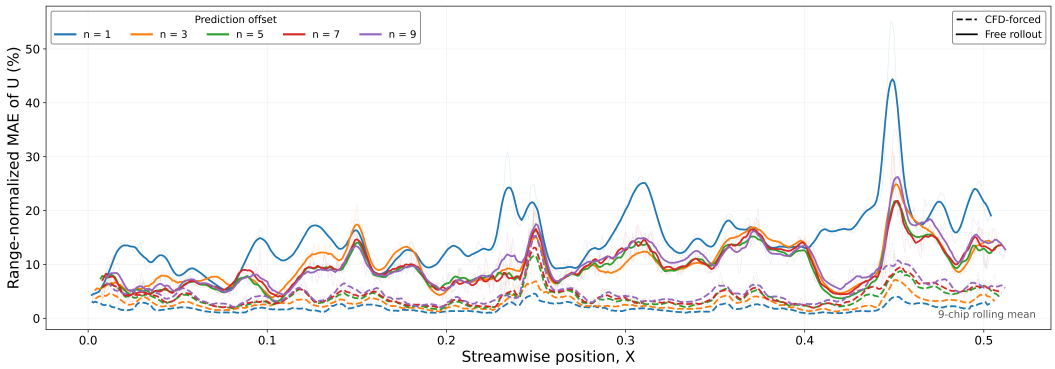

Figure 3: Prediction-offset sensitivity on the $5\times$ extended domain for $n \in \{1, 3, 5, 7, 9\}$. Colours denote prediction offset; dashed and solid curves denote CFD-forced evaluation and free rollout, respectively. Faint curves show the raw chip-wise NMAE values, while prominent curves show a centred nine-chip rolling mean, obtained by averaging each chip with the four neighbouring chips on either side.

The advantage of C2C on longer domains does not imply that an MLP is more expressive than graph- or attention-based models. Instead, it reflects a better match between the compositional prediction strategy and the extrapolation task. C2C learns a fixed-dimensional local transition from an upstream state and local geometry to a downstream state. As the domain becomes longer, the transition structure remains unchanged; only the number of transition applications increases. By contrast, the global baselines face a different extrapolation problem. Although GNO, GNOT, and Transolver can represent flexible spatial domains, their field-level mappings must generalise to domain extents and spatial interactions not seen during training. C2C instead applies the same local geometry context repeatedly along the domain. This local factorisation is consistent with its stronger length extrapolation.

The prediction-offset study further shows that recursive state reuse is a major source of long-range error. CFD-forced errors remain lower than free-rollout errors for all tested offsets, confirming the additional error introduced by recursive rollout. At $n = 1$, CFD-forced evaluation gives the lowest error, indicating that the shortest transition is locally the easiest to predict. However, the same setting gives the largest free-rollout error. This behaviour may reflect three related effects: adjacent chip states differ slightly, so the one-step mapping can approach a near-identity transformation; the target is located close to the upstream edge of the geometry window, producing a downstream-biased rather than balanced geometric context; and $n = 1$ requires the greatest number of recursive transition applications over a fixed physical length. Small local errors can therefore be repeatedly reused and accumulated during rollout, so strong one-step accuracy does not necessarily translate into strong long-range prediction. The main setting $n = 5$ provides a more balanced configuration, placing the target at the centre of the 11-chip geometry context while reducing rollout depth relative to $n = 1$.

4.3 OOD GENERALISATION

Figure 4 compares predictions on four OOD geometries, while Table 2 reports the corresponding geometry-level MAEs. C2C-MLP achieves the lowest mean MAEs on the Ahmed body and airfoil cases, at 0.1210 and 0.0940 $\mathrm{m/s}$, corresponding to reductions of 4.7% and 14.5% relative to Transolver, the most accurate baseline for these benchmark cases. For the periodic-hill, C2C-MLP predicts 0.0550 $\mathrm{m/s}$, compared with 0.0533 $\mathrm{m/s}$ when using GNO. On the backward-facing step, it obtains 0.0766 $\mathrm{m/s}$, compared with GNO's 0.0727 $\mathrm{m/s}$. The C2C errors are therefore 3.2% and 5.4% higher than the respective best results.

These results show that C2C remains competitive across all four OOD geometries. It achieves the lowest mean errors on the Ahmed body and airfoil cases, while on the backward-facing step and periodic-hill cases its errors remain close to the best baseline results. The results indicate that the learned local geometry–flow transitions retain useful transferability across unseen geometries, although the largest advantage of C2C is observed in domain-length extrapolation. The conclusions concern velocity magnitude only and do not establish accuracy for flow direction or full vector-field physical consistency.

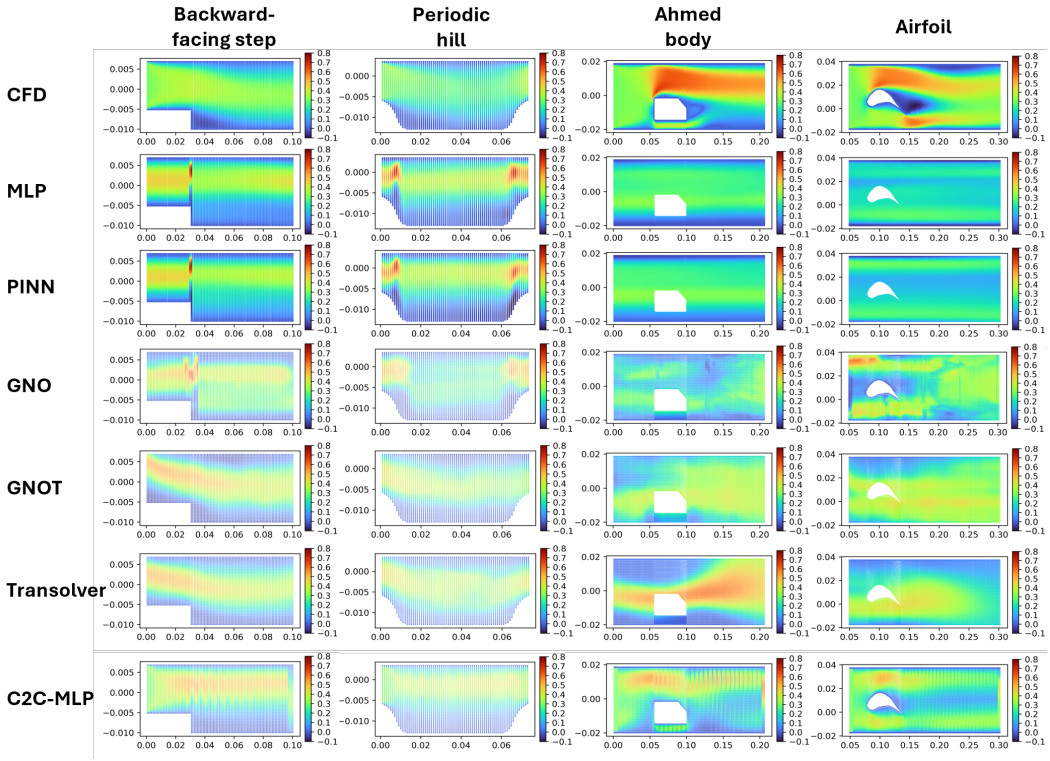

Figure 4: Velocity-magnitude predictions on OOD geometries. Columns show the backward-facing step, periodic hill, Ahmed body, and airfoil. Rows show CFD, MLP, PINN, GNO, GNOT, Transolver, and C2C-MLP.

Table 2: Case MAE of velocity magnitude across domain lengths and OOD geometry families. For each category and each training run, Case MAE is the median geometry-level MAE across ten test geometries. Values are mean±standard deviation over five independent training runs. C2C-MLP uses free rollout. Best results are shown in bold and second-best results are underlined.

Model	**Case MAE** (m/s)						
	1×	**2×**	**3×**	**Backward-facing step**	**Periodic hill**	**Ahmed body**	**Airfoil**
MLP	0.0890±0.0034	0.0964±0.0062	0.1123±0.0079	0.0809±0.0041	0.0763±0.0048	0.1630±0.0096	0.1650±0.0103
PINN	0.0891±0.0032	0.0963±0.0053	0.1153±0.0065	0.0732±0.0029	0.0762±0.0040	0.1660±0.0093	0.1680±0.0096
GNO	0.0884±0.0049	0.1107±0.0167	0.1473±0.0184	**0.0727±0.0043**	**0.0533±0.0039**	0.1940±0.0076	0.1360±0.0088
GNOT	0.0645±0.0036	0.1527±0.0101	0.1784±0.0128	0.0796±0.0052	0.0617±0.0073	0.1650±0.0263	0.1340±0.0210
Transolver	0.0575±0.0021	0.1684±0.0114	0.1830±0.0136	0.0876±0.0037	0.0618±0.0024	0.1270±0.0059	0.1100±0.0049
C2C-MLP	**0.0490±0.0023**	**0.0599±0.0022**	**0.0679±0.0024**	0.0766±0.0035	0.0550±0.0023	**0.1210±0.0052**	**0.0940±0.0047**

4.4 Computational Cost

Table 3 shows that C2C-MLP trains faster per epoch than MLP and PINN but slower than GNOT and Transolver. Its inference time of 2.575 s is substantially higher than the 0.013–0.071 s range of

the global neural baselines because downstream states are evaluated sequentially. Under the same hardware environment, the automated CFD workflow, including mesh generation and steady-state simulation, requires approximately 10–13 s per case. C2C-MLP therefore remains approximately 3.9–5.0× faster than CFD while providing the strongest length-extrapolation accuracy among the evaluated neural surrogates.

Table 3: Computational cost. Training time is per epoch; prediction time is per case on the 3× test domain.

Model	**Training time (s/epoch)**	**Prediction time (s/case)**
MLP	158 ± 1.6	0.058 ± 0.001
PINN	171 ± 3.3	0.071 ± 0.002
GNO	14.5 ± 2.0	0.030 ± 0.001
GNOT	4.3 ± 0.6	0.013 ± 0.001
Transolver	4.6 ± 0.8	0.015 ± 0.002
C2C-MLP	18.2 ± 1.0	2.575 ± 0.004

4.5 Scope and Limitations

The current C2C formulation is designed for flows with a dominant streamwise direction, so that the domain can be represented as an ordered sequence of spatial chips. This assumption is well suited to the through-flow configurations considered here, but it limits direct application to geometries with strongly multidirectional transport, multiple competing flow paths, or spatial structures that cannot be naturally ordered along a single propagation direction. Extending C2C to such cases will require a more general spatial decomposition or multidirectional transition formulation. A second limitation is that the present study is restricted to two-dimensional geometries. The chip representation and transition model are therefore defined on one-dimensional cross-sections arranged along the streamwise direction. Extension to three-dimensional flows would require replacing these cross-sections with surface- or volume-based local states while preserving the compositional transition structure.

Regarding evaluation scope, the present experiments use a fixed prescribed inlet condition to isolate domain-length and geometry effects; because the upstream flow state is an explicit model input, the C2C formulation itself does not require a fixed inlet velocity, although systematic evaluation across varying inlet velocities and profiles is outside the scope of this study. The present evaluation therefore focuses on domain-length extrapolation and geometry transfer under fixed operating conditions.

5 Conclusion

This study investigated spatial compositionality for domain-length extrapolation and transfer to unseen flow geometries. C2C learns a shared local transition between an upstream flow state, local geometry, and a downstream state, and reuses this transition to extend the solution without retraining or architectural modification. C2C-MLP achieved the lowest mean MAE on the 1×, 2×, and 3× domains, including a 39.5% reduction relative to the strongest baseline on the 3× case. OOD performance was geometry-dependent, with the best results on the Ahmed body and airfoil cases and competitive performance on the periodic-hill and backward-facing step cases. The CFD-forced evaluation and the prediction-offset study show that recursive state reuse is a major source of long-range error. Although $n = 1$ gives the lowest CFD-forced error, it produces the largest free-rollout error under the deepest recursive rollout, showing that accurate local transitions do not necessarily lead to accurate long-range prediction. These results support repeated local transition learning as the main source of C2C's length-extrapolation advantage. The current formulation is limited to two-dimensional flows with a dominant streamwise direction. Future work will extend C2C to multidirectional and three-dimensional flows and improve robustness under recursive rollout.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript organisation, language refinement, and LaTeX formatting. The authors remain fully responsible for the technical content, numerical results, citations, and final submitted text.

REPRODUCIBILITY STATEMENT

The main text defines the prediction target, chip representation, conditional transition, matched initial-state conditioning, rollout procedure, evaluation region, metrics, and timing scope. Supplementary Materials S1–S3 document data generation and preprocessing, model and training settings, metric implementation, and computational-cost measurements.

REFERENCES

S. R. Ahmed, G. Ramm, and G. Faltin. Some salient features of the time-averaged ground vehicle wake. SAE Technical Paper 840300, SAE International, 1984.

B. F. Armaly, F. Durst, J. C. F. Pereira, and B. Schönung. Experimental and theoretical investigation of backward-facing step flow. *Journal of Fluid Mechanics*, 127:473–496, 1983. doi: 10.1017/S0022112083002839.

Saakaar Bhatnagar, Yaser Afshar, Shaowu Pan, Karthik Duraisamy, and Shailendra Kaushik. Prediction of aerodynamic flow fields using convolutional neural networks. *Computational Mechanics*, 64(2):525–545, 2019. doi: 10.1007/s00466-019-01740-0.

Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. AirfRANS: High fidelity computational fluid dynamics dataset for approximating Reynolds-Averaged Navier–Stokes solutions. In *Advances in Neural Information Processing Systems*, volume 35, pp. 23463–23478. Curran Associates, Inc., 2022. doi: 10.52202/068431-1705. URL https://papers.nips.cc/paper_files/paper/2022/hash/94ab7b23a345f93333eac8748a66c763-Abstract-Datasets_and_Benchmarks.html.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Joel H. Ferziger, Milovan Perić, and Robert L. Street. *Computational Methods for Fluid Dynamics*. Springer International Publishing, Cham, 4 edition, 2020. ISBN 978-3-319-99691-2. doi: 10.1007/978-3-319-99693-6.

Xiaoxiao Guo, Wei Li, and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 481–490. ACM, 2016. doi: 10.1145/2939672.2939738.

Zhongkai Hao, Zhengyi Wang, Hang Su, Chengyang Ying, Yinpeng Dong, Songming Liu, Ze Cheng, Jian Song, and Jun Zhu. GNOT: A general neural operator transformer for operator learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 12556–12569. PMLR, 2023. URL https://proceedings.mlr.press/v202/hao23c.html.

Ameya D. Jagtap and George Em Karniadakis. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, 2020. doi: 10.4208/cicp.OA-2020-0164.

Ameya D. Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse

problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020. doi: 10.1016/j.cma.2020.113028.

Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, 426:109951, 2021. doi: 10.1016/j.jcp.2020.109951.

George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021. doi: 10.1038/s42254-021-00314-5.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023. URL https://jmlr.org/papers/v24/21-1524.html.

Michael Leer and Andreas Kempf. Fast flow field estimation for various applications with a universally applicable machine learning concept. *Flow, Turbulence and Combustion*, 107(1):175–200, 2021. doi: 10.1007/s10494-020-00234-x.

Wuyang Li, Xueshuang Xiang, and Yingxiang Xu. Deep domain decomposition method: Elliptic problems. In *Proceedings of The First Mathematical and Scientific Machine Learning Conference*, volume 107 of *Proceedings of Machine Learning Research*, pp. 269–286. PMLR, 2020a. URL https://proceedings.mlr.press/v107/li20a.html.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Graph kernel network for partial differential equations. *arXiv preprint arXiv:2003.03485*, 2020b. URL https://arxiv.org/abs/2003.03485.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Andrew Stuart, Kaushik Bhattacharya, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. In *Advances in Neural Information Processing Systems*, volume 33, pp. 6755–6766. Curran Associates, Inc., 2020c. URL https://papers.nips.cc/paper/2020/hash/4b21cf96d4cf612f239a6c322b10c8fe-Abstract.html.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for PDEs on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023a. URL https://jmlr.org/papers/v24/23-0064.html.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-informed neural operator for large-scale 3D PDEs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 35836–35854. Curran Associates, Inc., 2023b. doi: 10.52202/075280-1556. URL https://papers.neurips.cc/paper_files/paper/2023/hash/70518ea42831f02afc3a2828993935ad-Abstract-Conference.html.

Chenkai Mao, Robert Lupoiu, Tianxiang Dai, Mingkun Chen, and Jonathan Fan. Towards general neural surrogate solvers with specialized neural accelerators. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 34693–34711. PMLR, 2024. URL https://proceedings.mlr.press/v235/mao24b.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.

M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045.

Nils Thuerey, Konstantin Weißenow, Lukas Prantl, and Xiangyu Hu. Deep learning methods for Reynolds-Averaged Navier–Stokes simulations of airfoil flows. *AIAA Journal*, 58(1):25–36, 2020. doi: 10.2514/1.J058291.

Ricardo Vinuesa and Steven L. Brunton. Enhancing computational fluid dynamics with machine learning. *Nature Computational Science*, 2(6):358–366, 2022. doi: 10.1038/s43588-022-00264-7.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for PDEs on general geometries. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 53681–53705. PMLR, 2024. URL https://proceedings.mlr.press/v235/wu24r.html.

Heng Xiao, Jin-Long Wu, Sylvain Laizet, and Lian Duan. Flows over periodic hills of parameterized geometries: A dataset for data-driven turbulence modeling from direct simulations. *Computers & Fluids*, 200:104431, 2020. doi: 10.1016/j.compfluid.2020.104431.