From Transolver to Transolver++

Enabling PDE Solving on Million-Scale Geometries

Haixu Wu

School of Software, Tsinghua University

June 28, 2025

Tsinghua University

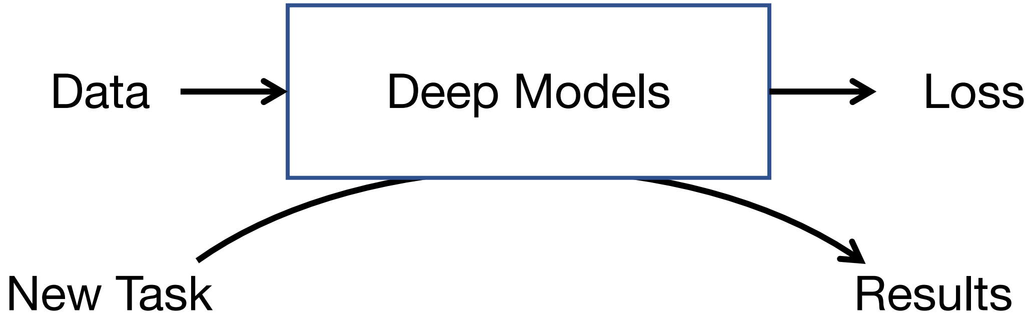
Solving PDEs

Classic Numerical Methods

- Recalculation for every new sample
- Each round will take hours or even days for a precise simulation

Huge computation costs

Neural PDE Solver

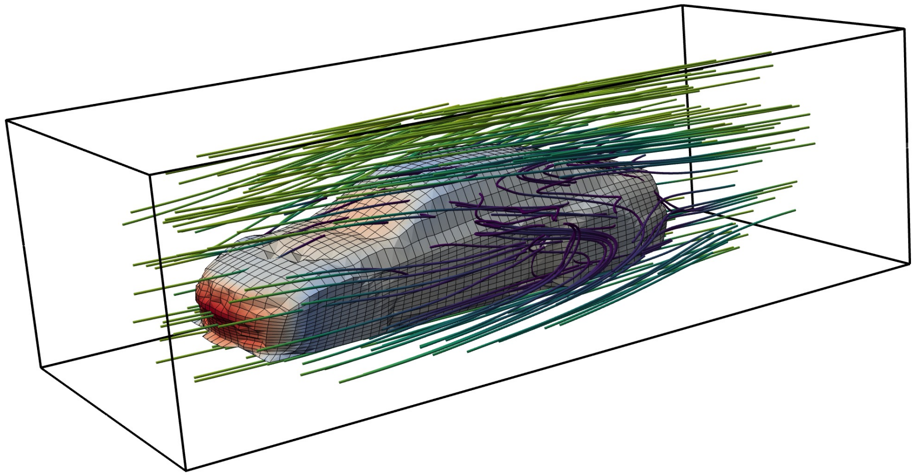


- Training once, inference a lot
- Each inference needs several seconds

An efficient surrogate tool

(In expectation)

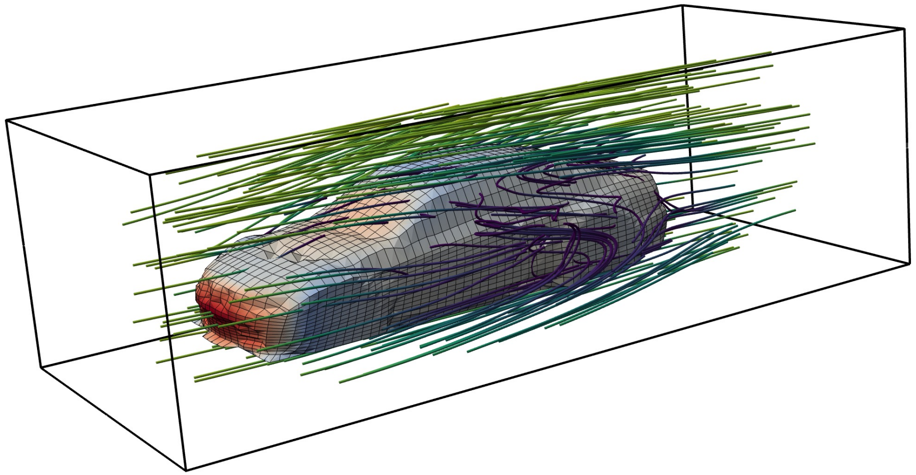
Challenges in Practical Industrial Design



Task: Estimate the drag coefficient of a given shape:

Surrounding Wind & Surface Pressure

Challenges in Practical Industrial Design



Task: Estimate the drag coefficient of a given shape:

Surrounding Wind & Surface Pressure

1. Large-scale meshes → **Huge computation cost**
2. Complex and unstructured geometrics → **Complex geometric learning**
3. Multiphysics interaction → **Intricate physical correlations**

ICML | 2024

The Forty-first International Conference on Machine Learning

Transolver: A Fast Transformer Solver for PDEs on General Geometries

Haixu Wu [1] Huakun Luo [1] Haowen Wang [1] Jianmin Wang [1] Mingsheng Long [1]

Haixu Wu

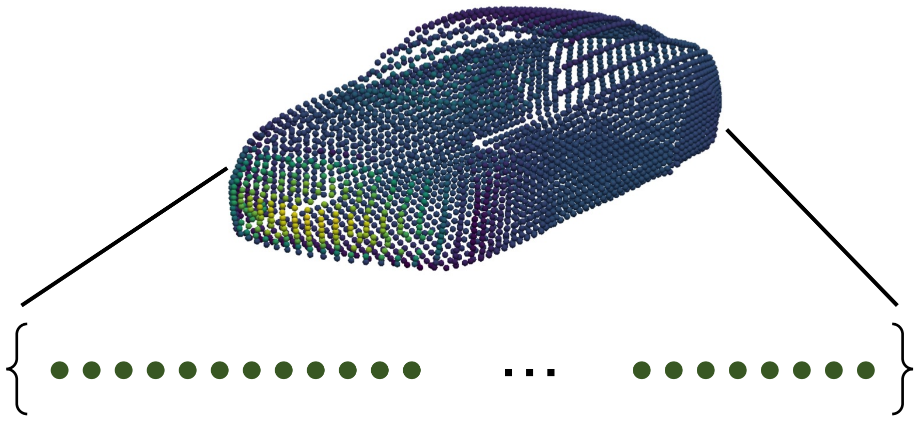
Huakun Luo

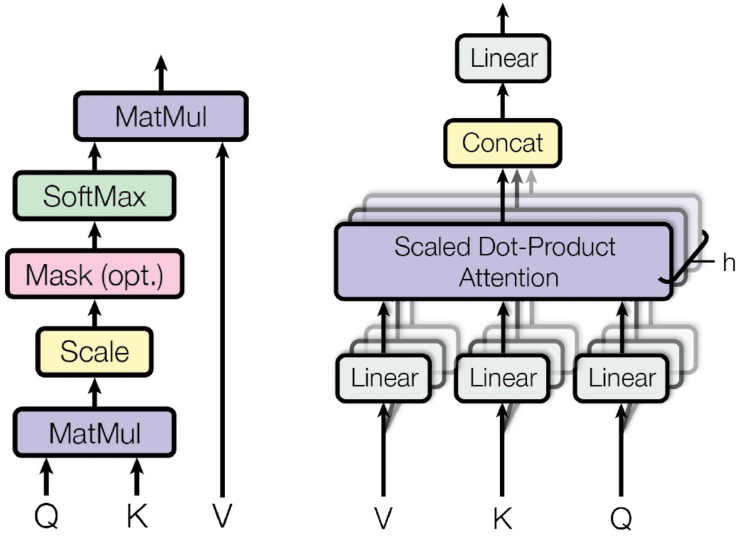
Haowen Wang

Jianmin Wang

Mingsheng Long

Transformer-based PDE Solvers



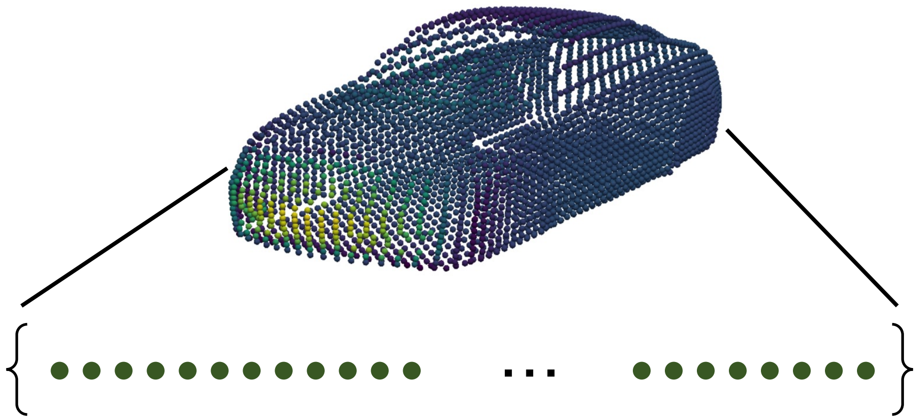


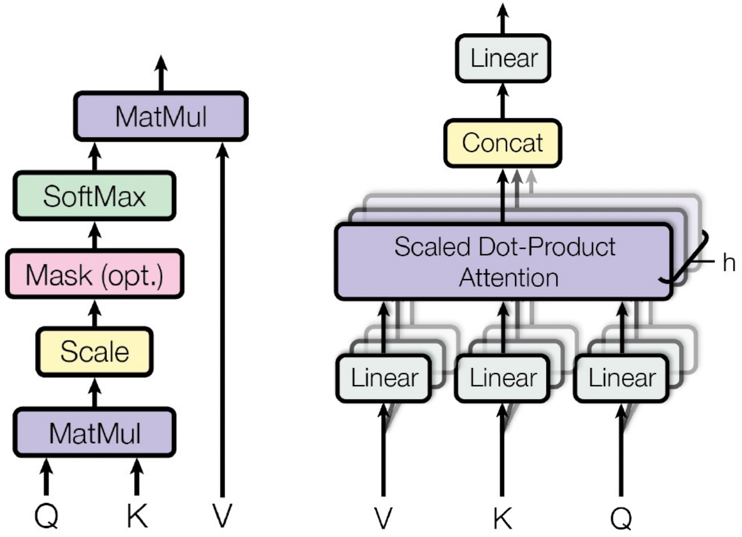
(1) Geometries as point sequences (2) Attention as Monte Carlo Integral

OFormer, Galerkin Transformer, GNOT, etc

1. **Quadratic complexity**
2. **Hard to capture physical correlations among massive points**

Transformer-based PDE Solvers





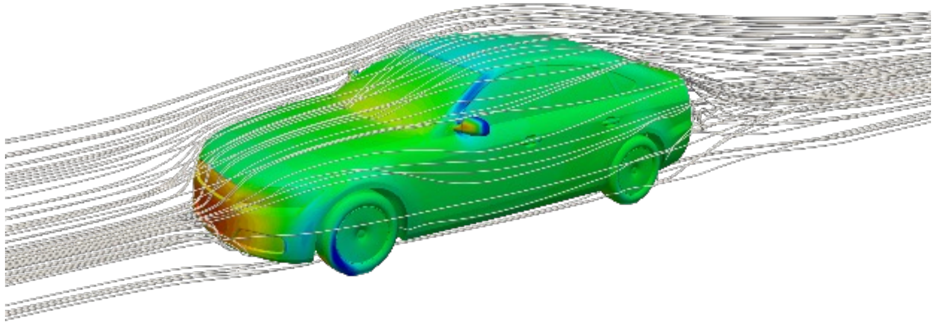
(1) Geometries as point sequences (2) Attention as Monte Carlo Integral

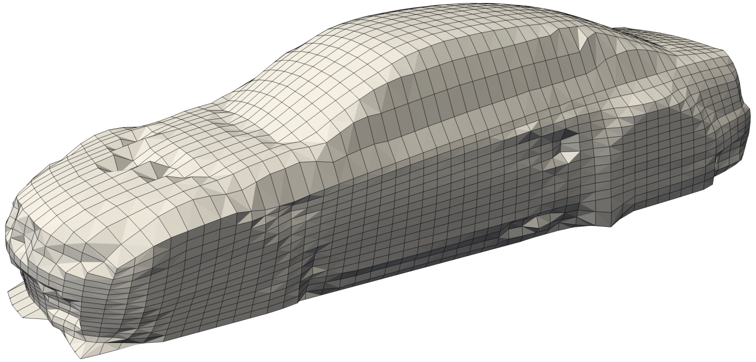
OFormer, Galerkin Transformer, GNOT, etc

How to efficiently capture physical correlations underlying discretized meshes

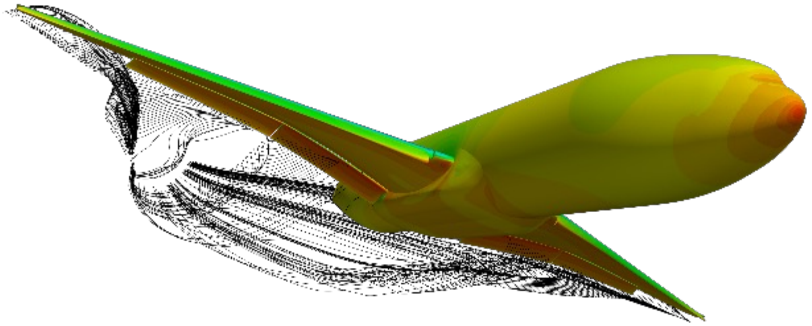
is the key to “transform” Transformers into practical PDE solvers

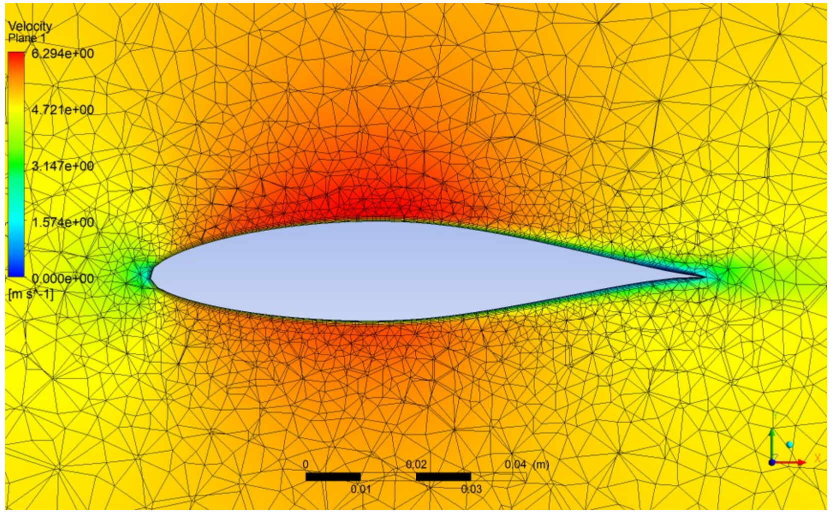
Solving PDEs: Discretization





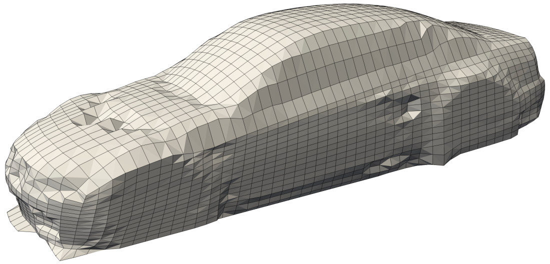
Car





Airplane

A Foundational Idea of Transolver



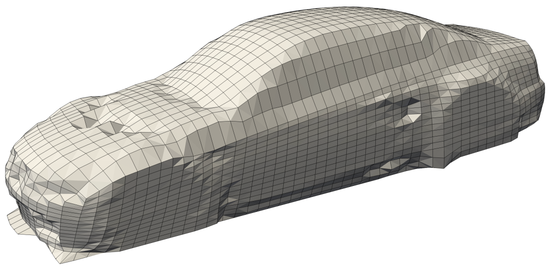
Discretized Domain

Previous Work

Being “trapped” to superficial and unwieldy meshes

Difficulties in Complexity, Geometry, Physics

A Foundational Idea of Transolver

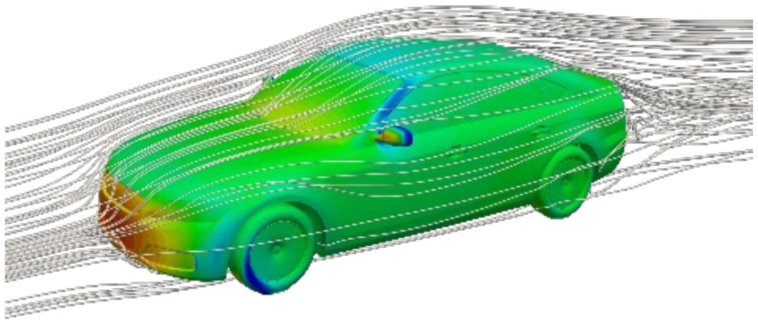


Discretized Domain

Previous Work

Being “trapped” to superficial and unwieldy meshes

Difficulties in Complexity, Geometry, Physics



Physics Domain

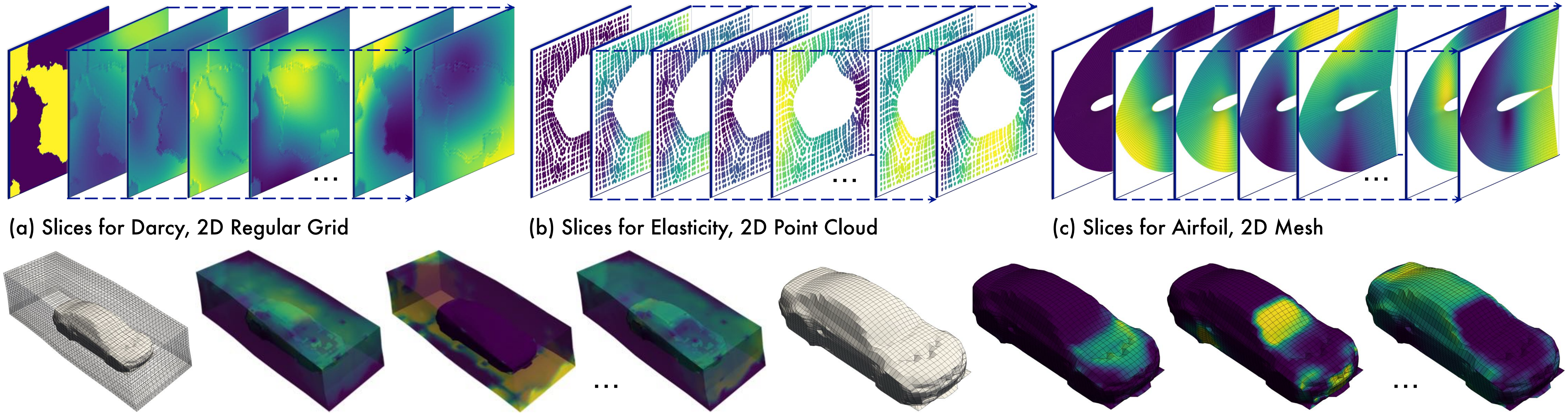
Transolver

Learning **intrinsic physical states** under

complex and large-scale geometrics

Better Complexity, Geometry, Physics Modeling

Learning Physical States

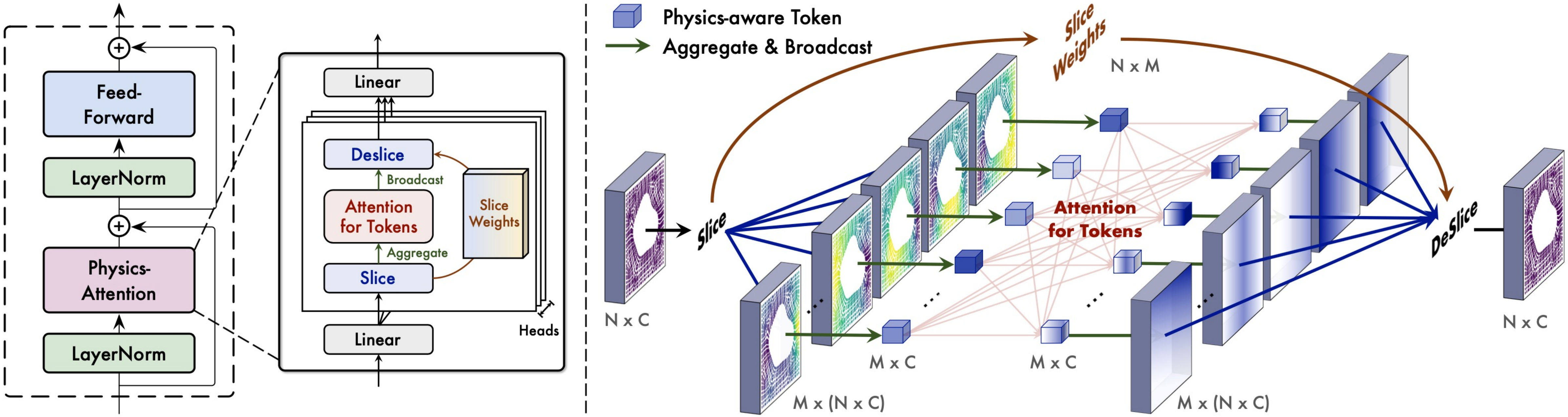


(a) Slices for Darcy, 2D Regular Grid (b) Slices for Elasticity, 2D Point Cloud (c) Slices for Airfoil, 2D Mesh

(d) Slices for Shape-Net Car Surrounding Velocity, 3D Volumes (e) Slices for Shape-Net Car Surface Pressure, 3D Mesh

Mesh points under **similar physical states** will be ascribed to the same **slice** and then encoded into a physics-aware token.

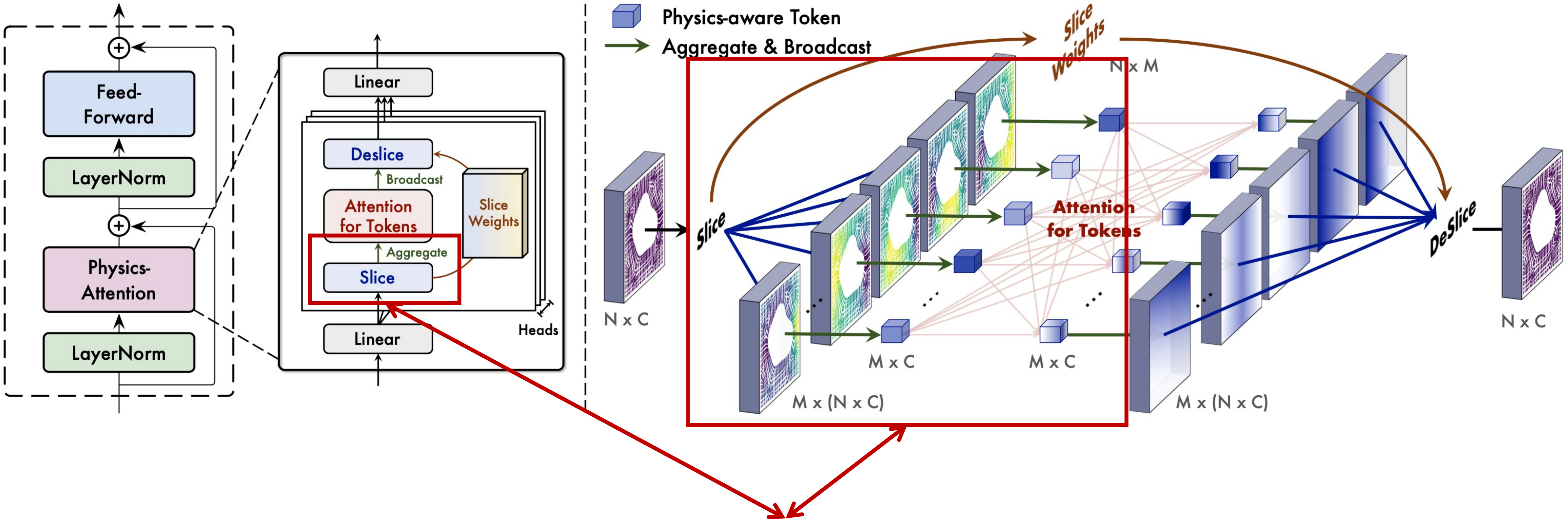
Overview of Transolver



Transolver applies attention to learned physical states **(Physics-Attention)**

① Mesh → physics ② Attention (Integral) ③ Physics → Mesh

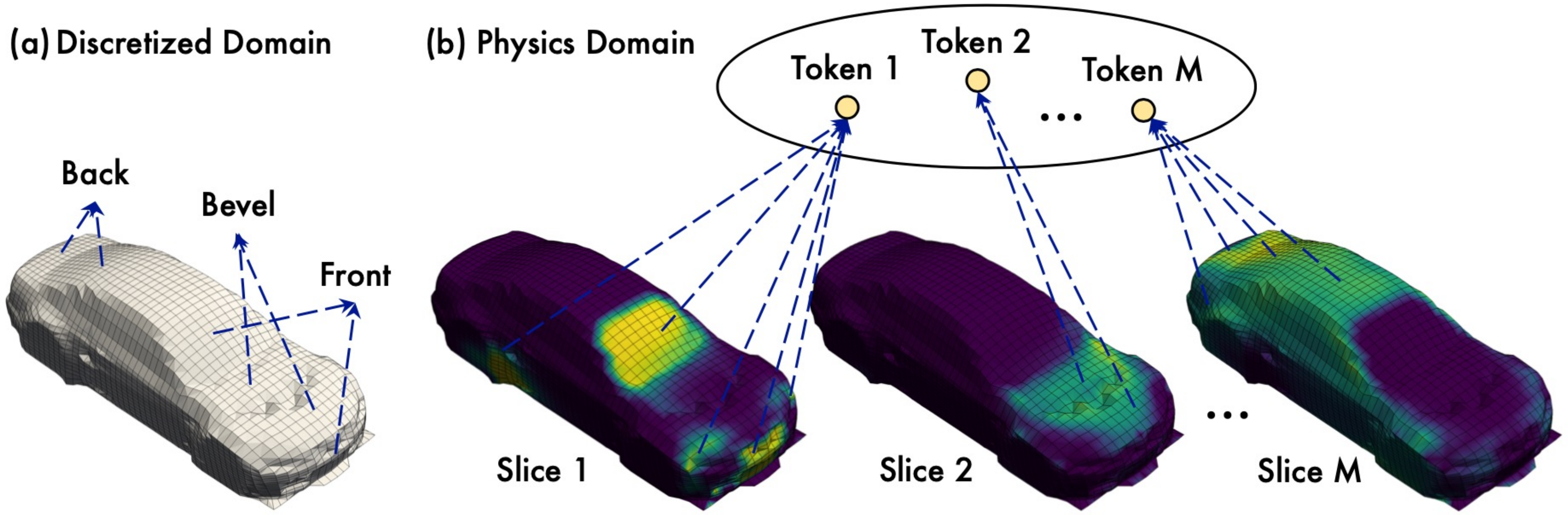
Overview of Transolver



① Mesh → physics

To obtain physics-aware tokens

Mesh → physics



1. Assign each point to slices with weights learned from features

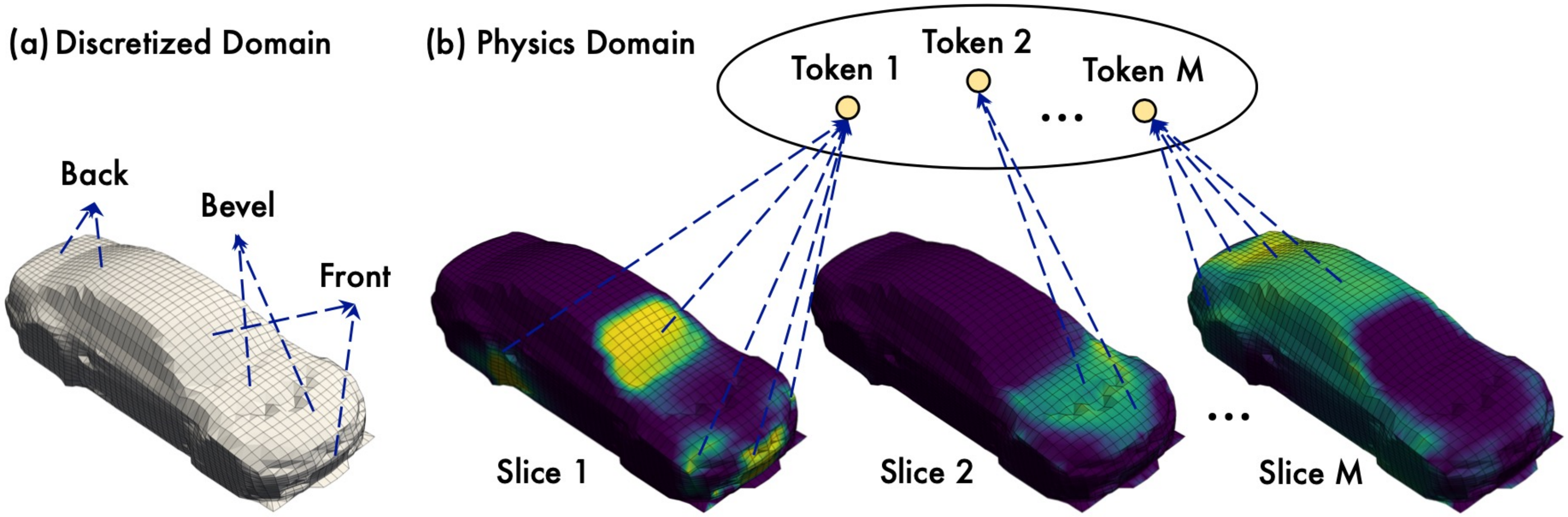
$$\{\mathbf{w}_i\}_{i=1}^{N} = \left\{ \underline{\mathrm{Softmax}} \left(\mathrm{Project}\left(\mathbf{x}_i\right) \right) \right\}_{i=1}^{N}$$

$$\mathbf{s}_j = \{\mathbf{w}_{i,j}\mathbf{x}_i\}_{i=1}^{N},$$

***N* Points to *M* Slices**

Softmax for low-entropy slices

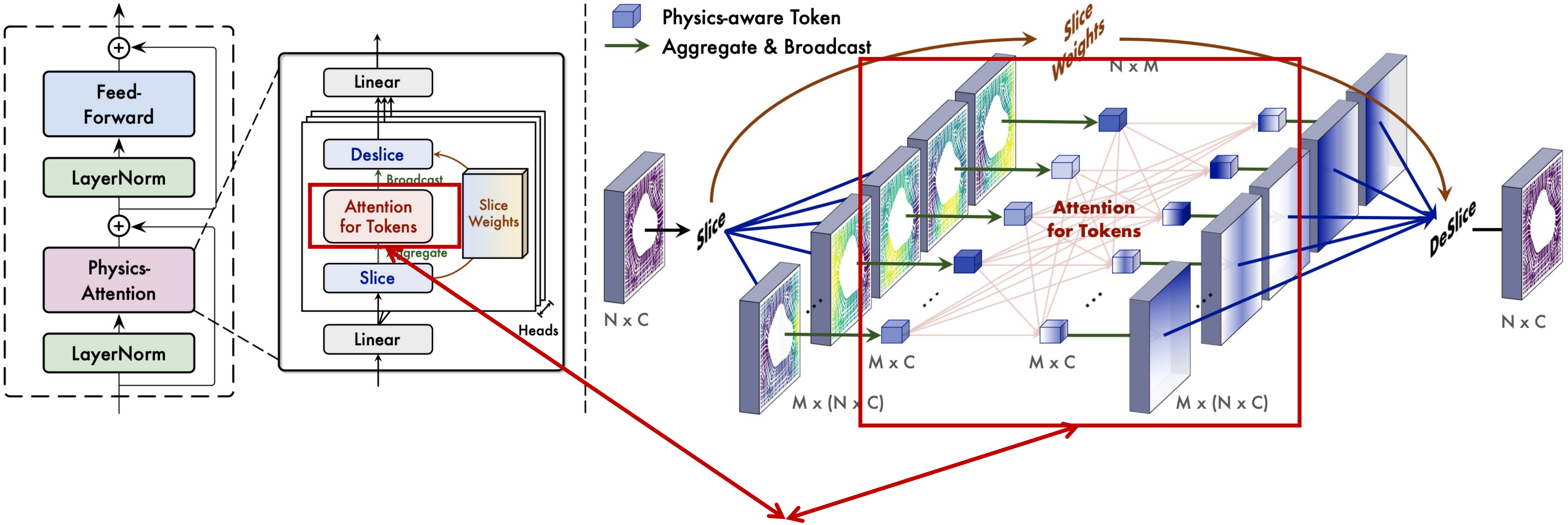
Mesh → physics



1. Assign each point to slices 2. Aggregate slices for physics-aware tokens

$$\mathbf{z}_j = \frac{\sum_{i=1}^{N} \mathbf{s}_{j,i}}{\sum_{i=1}^{N} \mathbf{w}_{i,j}} = \frac{\sum_{i=1}^{N} \mathbf{w}_{i,j}\mathbf{x}_i}{\sum_{i=1}^{N} \mathbf{w}_{i,j}}$$

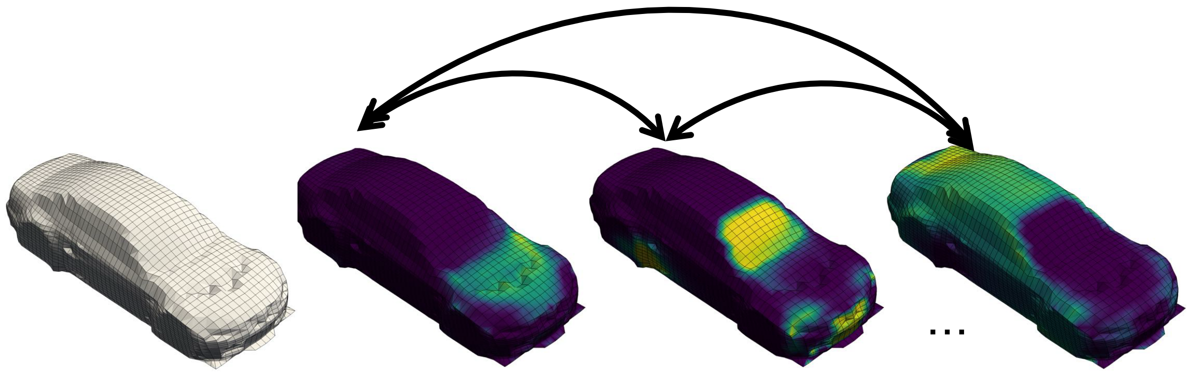
Overview of Transolver



② Attention among physics tokens

Approximate Integral to solve PDEs

Attention among physics tokens

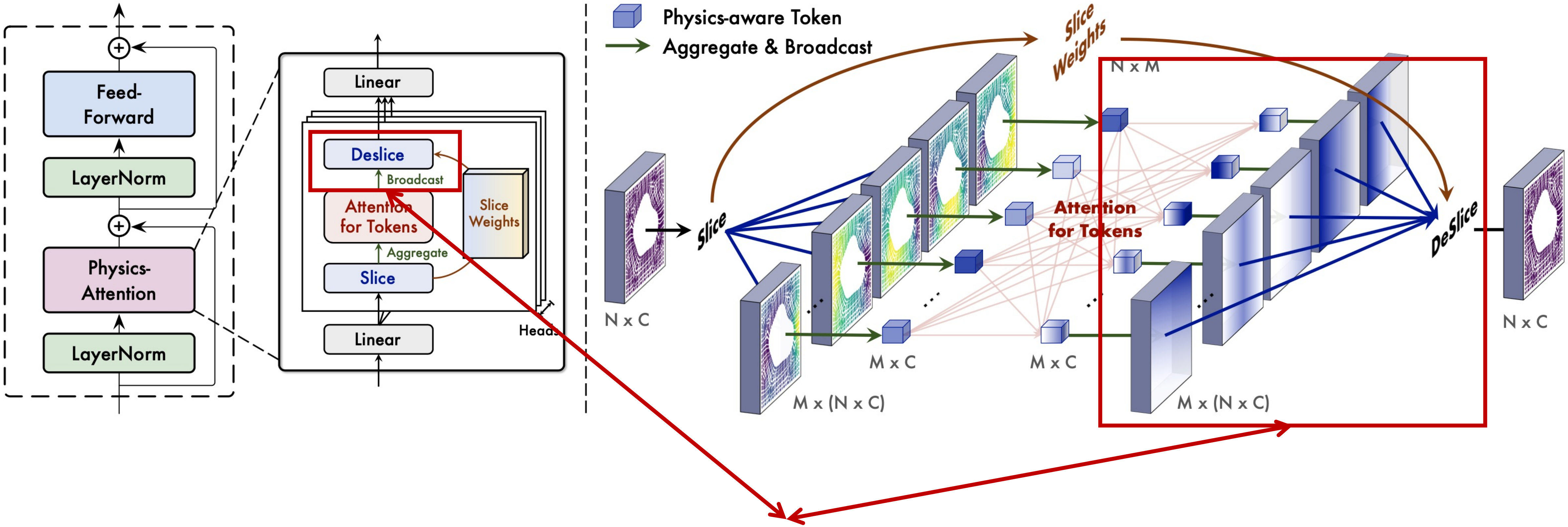


$$\mathbf{q}, \mathbf{k}, \mathbf{v} = \mathrm{Linear}(\underline{\mathbf{z}}), \quad \mathbf{z}' = \mathrm{Softmax}\left(\frac{\mathbf{q}\mathbf{k}^{\mathsf{T}}}{\sqrt{C}}\right)\mathbf{v}$$

Canonical attention among physics tokens

1. Complexity: $\mathcal{O}(N^2C) \rightarrow \mathcal{O}(M^2C)$
2. Capture interactions among physics states
3. Theorem: Attention as learnable integral operator

Overview of Transolver



③ Physics → Mesh

Project physics information back to mesh

$$\mathbf{x}'_i = \sum_{j=1}^{M} \mathbf{w}_{i,j}\mathbf{z}'_j$$

Theoretical Understanding of Transolver

1. Corollary of *Attention is a learnable integral*

Since attention mechanism is applied to tokens encoded from slices, **the step 2 (attention part of Transolver) is a learnable integral for the <u>physics domain</u>**

Is Physics-Attention still an <u>input domain</u> integral?

$$\mathcal{G}(\boldsymbol{u})(\mathbf{g}^*) = \int_{\Omega} \kappa(\mathbf{g}^*, \boldsymbol{\xi})\boldsymbol{u}(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}$$

Theoretical Understanding of Transolver

$$\mathcal{G}(\boldsymbol{u})(\mathbf{g}) = \int_{\Omega} \kappa(\mathbf{g},\boldsymbol{\xi})\boldsymbol{u}(\boldsymbol{\xi})\mathrm{d}\boldsymbol{\xi}$$

$$= \int_{\Omega_{\mathrm{s}}} \kappa_{\mathrm{ms}}\left(\mathbf{g},\boldsymbol{\xi}_{\mathrm{s}}\right)\boldsymbol{u}_{\mathrm{s}}\left(\boldsymbol{\xi}_{\mathrm{s}}\right)\mathrm{d}\boldsymbol{g}^{-1}(\boldsymbol{\xi}_{\mathrm{s}}) \qquad (\kappa_{\mathrm{ms}}(\cdot,\cdot):\Omega\times\Omega_{\mathrm{s}}\to\mathbb{R}^{C\times C} \text{ is a kernel function})$$

$$= \int_{\Omega_{\mathrm{s}}} \kappa_{\mathrm{ms}}\left(\mathbf{g},\boldsymbol{\xi}_{\mathrm{s}}\right)\boldsymbol{u}_{\mathrm{s}}\left(\boldsymbol{\xi}_{\mathrm{s}}\right)|\det(\nabla_{\boldsymbol{\xi}_{\mathrm{s}}}\boldsymbol{g}^{-1}(\boldsymbol{\xi}_{\mathrm{s}}))|\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}}$$

$$= \int_{\Omega_{\mathrm{s}}} \left(\frac{\int_{\Omega_{\mathrm{s}}} w_{\mathbf{g},\boldsymbol{\xi}_{\mathrm{s}}'}\kappa_{\mathrm{ss}}\left(\boldsymbol{\xi}_{\mathrm{s}}',\boldsymbol{\xi}_{\mathrm{s}}\right)\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}}'}{\int_{\Omega_{\mathrm{s}}} w_{\mathbf{g},\boldsymbol{\xi}_{\mathrm{s}}'}\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}}'}\right)\boldsymbol{u}_{\mathrm{s}}\left(\boldsymbol{\xi}_{\mathrm{s}}\right)|\det(\nabla_{\boldsymbol{\xi}_{\mathrm{s}}}\boldsymbol{g}^{-1}(\boldsymbol{\xi}_{\mathrm{s}}))|\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}} \qquad (\kappa_{\mathrm{ms}} \text{ is a linear combination of } \kappa_{\mathrm{ss}} \text{ with weights } w_{*,*})$$

$$= \int_{\Omega_{\mathrm{s}}} \underbrace{w_{\mathbf{g},\boldsymbol{\xi}_{\mathrm{s}}'}}_{\text{DeSlice}} \int_{\Omega_{\mathrm{s}}} \underbrace{\kappa_{\mathrm{ss}}\left(\boldsymbol{\xi}_{\mathrm{s}}',\boldsymbol{\xi}_{\mathrm{s}}\right)}_{\text{Attention among slice tokens}} \underbrace{\boldsymbol{u}_{\mathrm{s}}\left(\boldsymbol{\xi}_{\mathrm{s}}\right)}_{\text{Slice token}} |\det(\nabla_{\boldsymbol{\xi}_{\mathrm{s}}}\boldsymbol{g}^{-1}(\boldsymbol{\xi}_{\mathrm{s}}))|\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}}\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}}' \qquad \left(\text{Suppose that } \int_{\Omega_{\mathrm{s}}} w_{\mathbf{g},\boldsymbol{\xi}_{\mathrm{s}}'}\mathrm{d}\boldsymbol{\xi}_{\mathrm{s}}' = 1\right)$$

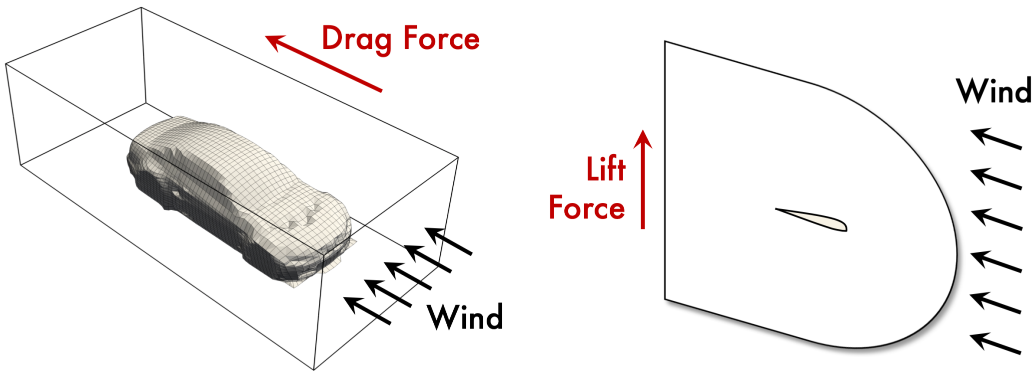
$$\approx \underbrace{\sum_{j=1}^{M}\mathbf{w}_{i,j}}_{\text{Eq. (4)}} \underbrace{\sum_{t=1}^{M}\frac{\exp\left(\left(\mathbf{W_q}\boldsymbol{u}_{\mathrm{s}}(\boldsymbol{\xi}_{\mathrm{s},j})\right)\left(\mathbf{W_k}\boldsymbol{u}_{\mathrm{s}}(\boldsymbol{\xi}_{\mathrm{s},t})\right)^{\mathsf{T}}/\tau\right)}{\sum_{p=1}^{M}\exp\left(\left(\mathbf{W_q}\boldsymbol{u}_{\mathrm{s}}(\boldsymbol{\xi}_{\mathrm{s},j})\right)\left(\mathbf{W_k}\boldsymbol{u}_{\mathrm{s}}(\boldsymbol{\xi}_{\mathrm{s},p})\right)^{\mathsf{T}}/\tau\right)}\mathbf{W_v}}_{\text{Eq. (3)}} \underbrace{\left(\frac{\sum_{p=1}^{N}\mathbf{w}_{p,t}\boldsymbol{u}(\mathbf{g}_p)}{\sum_{p=1}^{N}\mathbf{w}_{p,t}}\right)}_{\text{Eq. (2)}} \qquad \text{(Lemma A.1)}$$

$$= \sum_{j=1}^{M}\mathbf{w}_{i,j}\sum_{t=1}^{M}\frac{\exp(\mathbf{q}_j\mathbf{k}_t^{\mathsf{T}}/\tau)}{\sum_{p=1}^{M}\exp(\mathbf{q}_j\mathbf{k}_p^{\mathsf{T}}/\tau)}\mathbf{v}_t,$$

All the designs in Transolver can be directly derived.

Experiments

Geometry	Benchmarks	#Dim	#Mesh
Point Cloud	Elasticity	2D	972
Structured Mesh	Plasticity Airfoil Pipe	2D+Time 2D 2D	3,131 11,271 16,641
Regular Grid	Navier–Stokes Darcy	2D+Time 2D	4,096 7,225
Unstructured Mesh	Shape-Net Car AirfRANS	3D 2D	32,186 32,000



(a) Shape-Net Car

(b) AirfRANS

Six standard benchmarks, two practical design tasks

More than 20 baselines

Standard PDE-Solving Benchmarks

Model	Point Cloud	Structured Mesh			Regular Grid	
	Elasticity	Plasticity	Airfoil	Pipe	Navier–Stokes	Darcy
FNO (Li et al., 2021)	/	/	/	/	0.1556	0.0108
WMT (Gupta et al., 2021)	0.0359	0.0076	0.0075	0.0077	0.1541	0.0082
U-FNO (Wen et al., 2022)	0.0239	0.0039	0.0269	0.0056	0.2231	0.0183
geo-FNO (Li et al., 2022)	0.0229	0.0074	0.0138	0.0067	0.1556	0.0108
U-NO (Rahman et al., 2023)	0.0258	0.0034	0.0078	0.0100	0.1713	0.0113
F-FNO (Tran et al., 2023)	0.0263	0.0047	0.0078	0.0070	0.2322	0.0077
LSM (Wu et al., 2023)	0.0218	0.0025	<u>0.0059</u>	0.0050	0.1535	<u>0.0065</u>
Galerkin (Cao, 2021)	0.0240	0.0120	0.0118	0.0098	0.1401	0.0084
HT-Net (Liu et al., 2022)	/	0.0333	0.0065	0.0059	0.1847	0.0079
OFormer (Li et al., 2023c)	0.0183	<u>0.0017</u>	0.0183	0.0168	0.1705	0.0124
GNOT (Hao et al., 2023)	<u>0.0086</u>	0.0336	0.0076	<u>0.0047</u>	0.1380	0.0105
FactFormer (Li et al., 2023d)	/	0.0312	0.0071	0.0060	0.1214	0.0109
ONO (Xiao et al., 2024)	0.0118	0.0048	0.0061	0.0052	<u>0.1195</u>	0.0076
Transolver (Ours)	**0.0064**	**0.0012**	**0.0053**	**0.0033**	**0.0900**	**0.0057**
Relative Promotion	25.6%	29.4%	10.2%	29.7%	24.7%	12.3%

Transolver achieves 22% error reduction over the second-best model

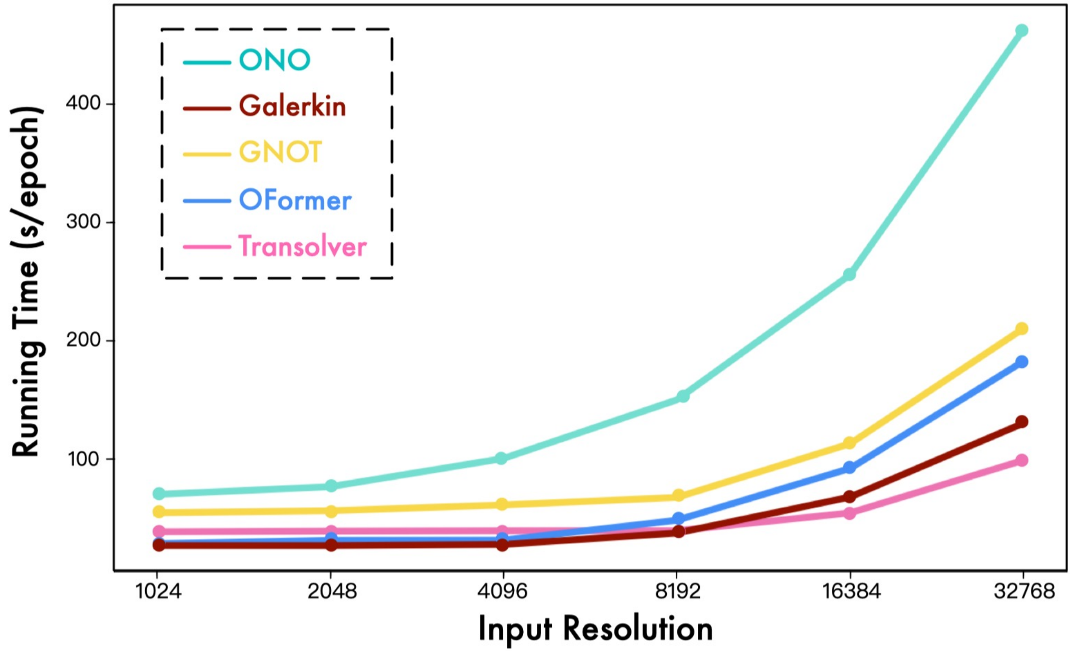
Practical Design Tasks

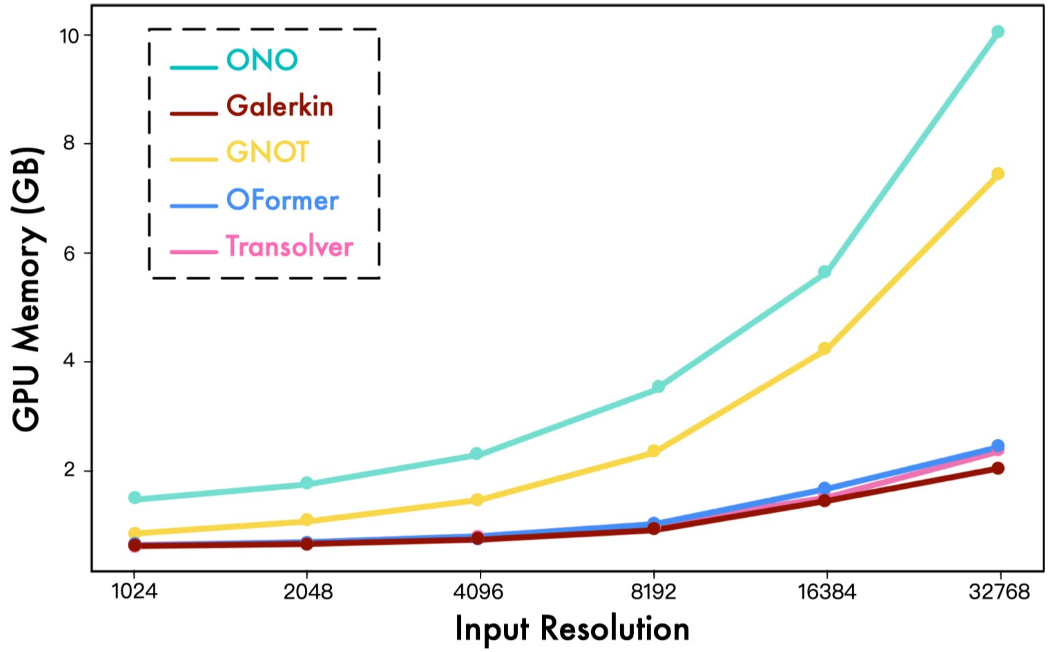
Model*	Shape-Net Car				AirfRANS			
	Volume ↓	Surf ↓	C_D ↓	ρ_D ↑	Volume ↓	Surf ↓	C_L ↓	ρ_L ↑
Simple MLP	0.0512	0.1304	0.0307	0.9496	0.0081	0.0200	0.2108	0.9932
GraphSAGE (Hamilton et al., 2017)	0.0461	0.1050	0.0270	0.9695	0.0087	0.0184	0.1476	0.9964
PointNet (Qi et al., 2017)	0.0494	0.1104	0.0298	0.9583	0.0253	0.0996	0.1973	0.9919
Graph U-Net (Gao & Ji, 2019)	0.0471	0.1102	0.0226	0.9725	0.0076	0.0144	0.1677	0.9949
MeshGraphNet (Pfaff et al., 2021)	0.0354	0.0781	0.0168	0.9840	0.0214	0.0387	0.2252	0.9945
GNO (Li et al., 2020a)	0.0383	0.0815	0.0172	0.9834	0.0269	0.0405	0.2016	0.9938
Galerkin (Cao, 2021)	0.0339	0.0878	0.0179	0.9764	0.0074	0.0159	0.2336	0.9951
geo-FNO (Li et al., 2022)	0.1670	0.2378	0.0664	0.8280	0.0361	0.0301	0.6161	0.9257
GNOT (Hao et al., 2023)	0.0329	0.0798	0.0178	0.9833	0.0049	0.0152	0.1992	0.9942
GINO (Li et al., 2023a)	0.0386	0.0810	0.0184	0.9826	0.0297	0.0482	0.1821	0.9958
3D-GeoCA (Deng et al., 2024)	0.0319	0.0779	0.0159	0.9842	/	/	/	/
Transolver (Ours)	**0.0207**	**0.0745**	**0.0103**	**0.9935**	**0.0037**	**0.0142**	**0.1030**	**0.9978**

Design-oriented metrics: Drag/lift coefficients and their Spearman's correlation

Transolver performs best in both physics and design-oriented metrics

Efficiency

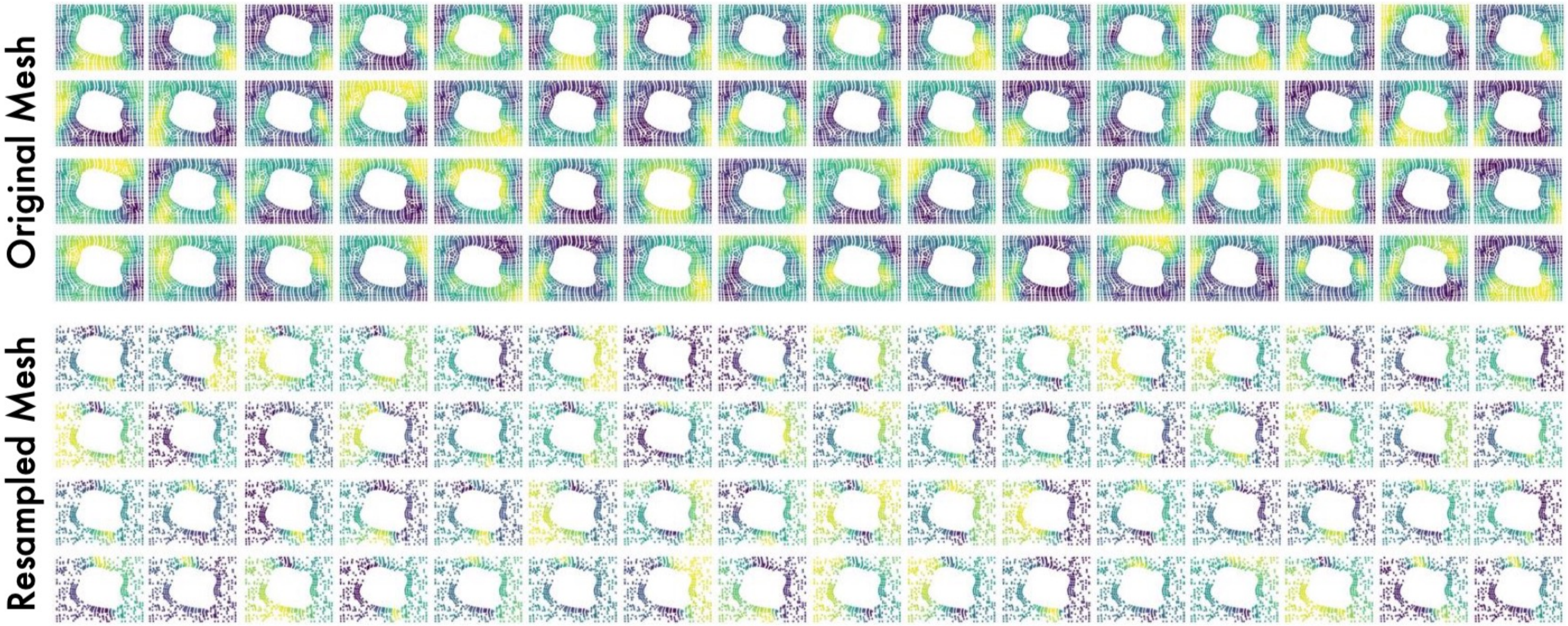




Favorable efficiency and performance balance

Transolver is faster than linear Transformers in large-scale meshes.

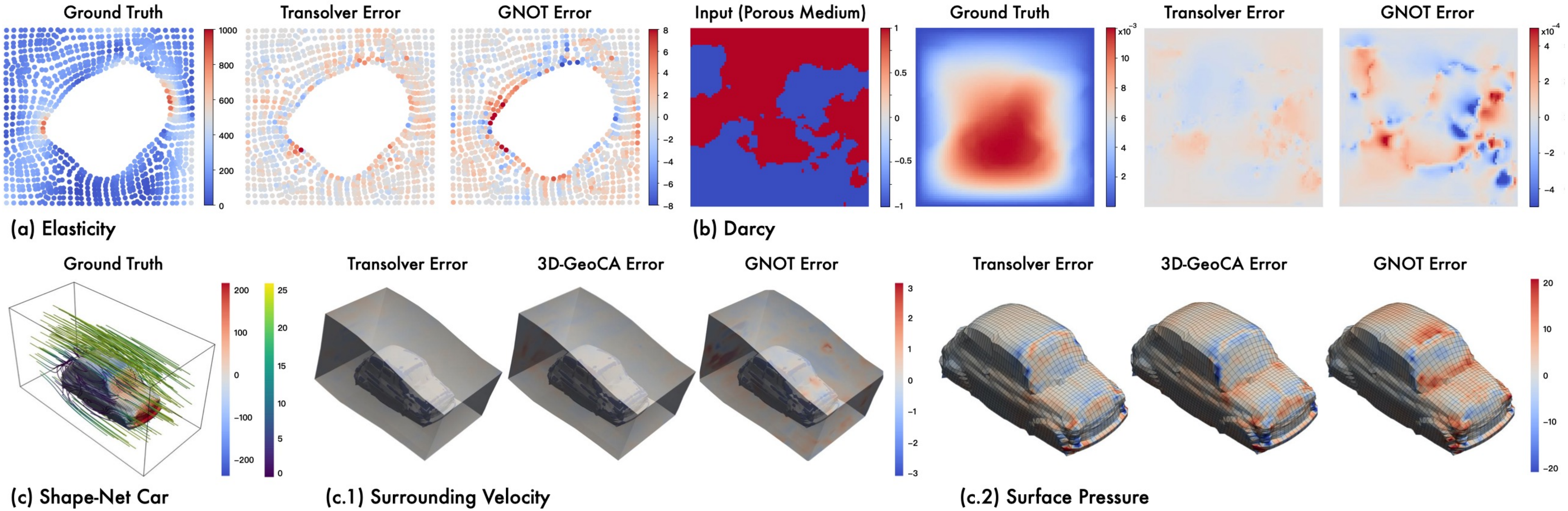
Physics-Attention Visualization



Slice visualization on Elasticity

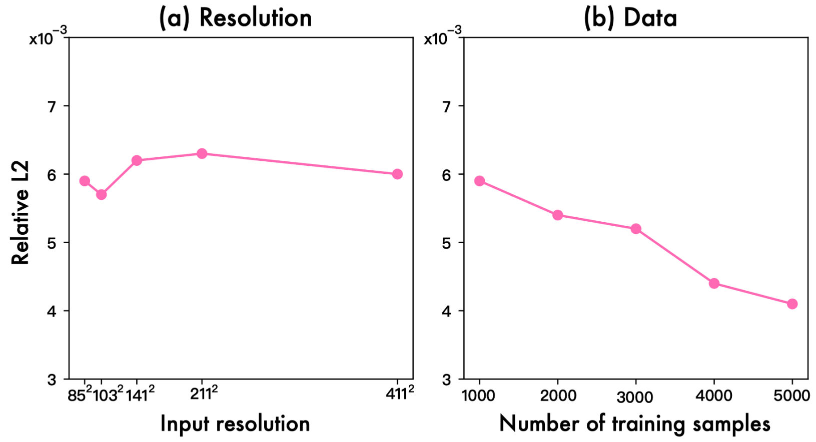
Transolver is mesh-free, precisely captures states **even on broken meshes**

Showcases

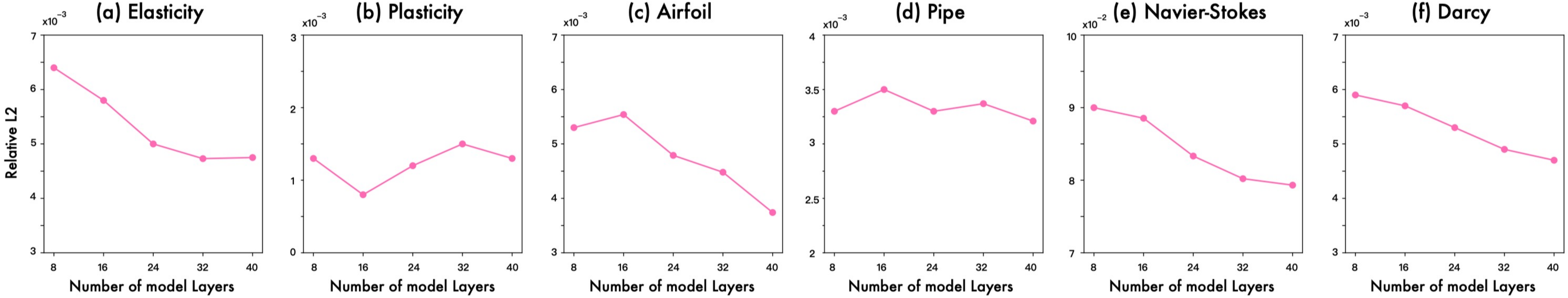


Transolver excels in solving **multiphysics PDEs on hybrid geometrics**

Pursuing PDE Foundation Models: **Scalability**

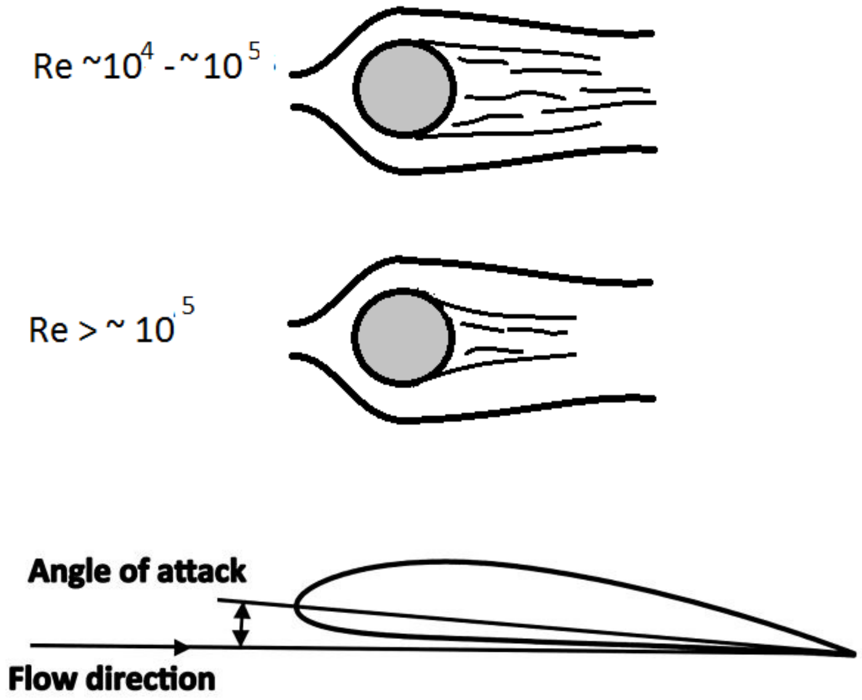


1. **Resolution:** Consistent performance at varied scales
2. **Data:** Benefiting from larger training data
3. **Parameter:** Benefiting from more parameters



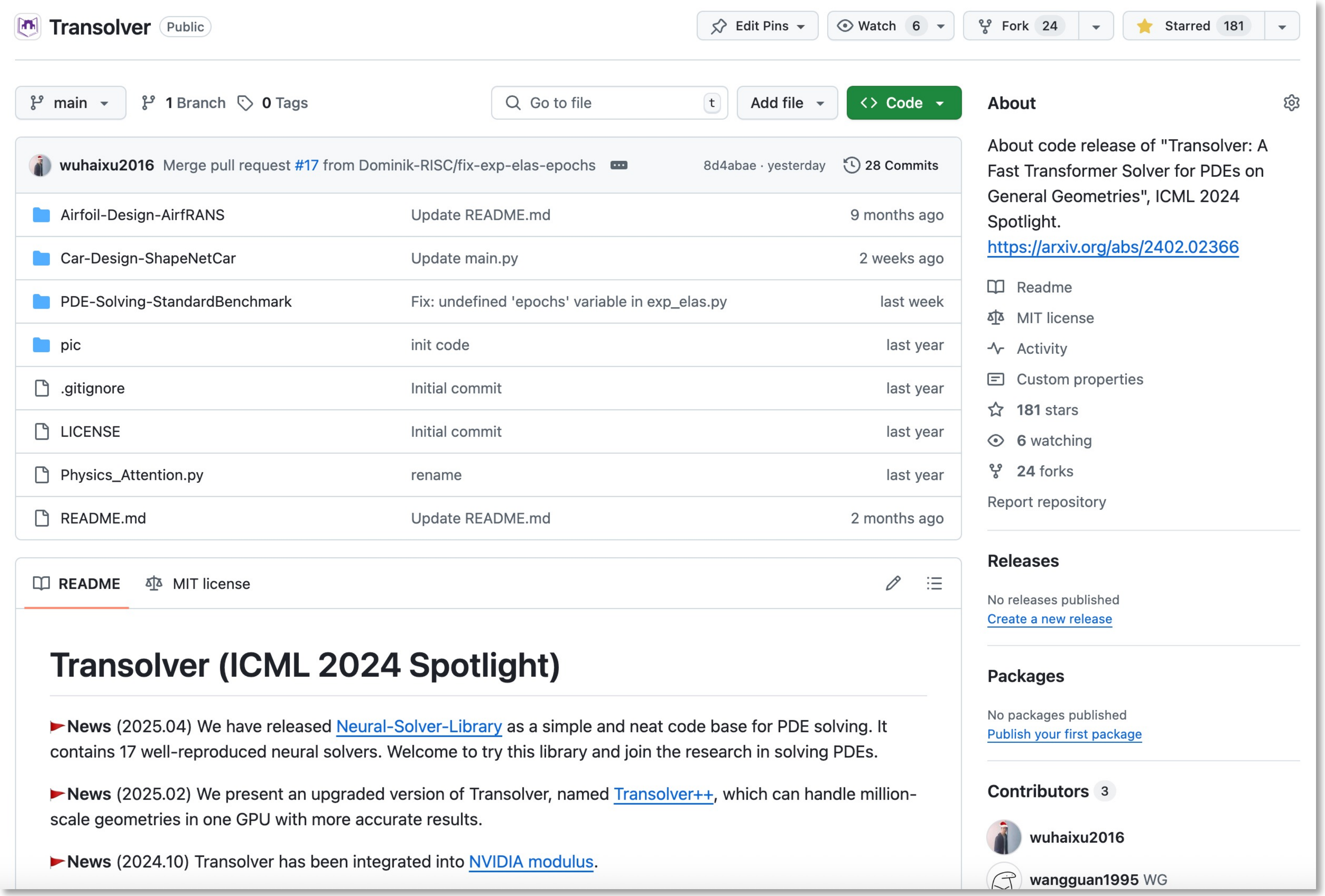
Pursuing PDE Foundation Models: **Generalization**

Models	OOD Reynolds		OOD Angles	
	$C_L \downarrow$	$\rho_L \uparrow$	$C_L \downarrow$	$\rho_L \uparrow$
Simple MLP	0.6205	0.9578	0.4128	0.9572
GraphSAGE (2017)	0.4333	0.9707	0.2538	0.9894
PointNet (2017)	0.3836	0.9806	0.4425	0.9784
Graph U-Net (2019)	0.4664	0.9645	0.3756	0.9816
MeshGraphNet (2021)	1.7718	0.7631	0.6525	0.8927
GNO (2020a)	0.4408	0.9878	0.3038	0.9884
Galerkin (2021)	0.4615	0.9826	0.3814	0.9821
GNOT (2023)	0.3268	0.9865	0.3497	0.9868
GINO (2023a)	0.4180	0.9645	0.2583	0.9923
Transolver (Ours)	**0.2996**	**0.9896**	**0.1500**	**0.9950**



Transolver still performs best (**Spearman's correlation ~ 99%**) in OOD settings

Open-Source Code



NVIDIA

physicsnemo

Code for Transolver in Modulus

Code for Transolver

Code Link: *https://github.com/thuml/Transolver*

ICML | 2025

The Forty-second International Conference on Machine Learning

Transolver++: An Accurate Neural Solver for PDEs on Million-Scale Geometries

Huakun Luo [*,1] **Haixu Wu** [*,1] **Hang Zhou** [1] **Lanxiang Xing** [1] **Yichen Di** [1] **Jianmin Wang** [1] **Mingsheng Long** [1]


Huakun Luo


Haixu Wu


Hang Zhou

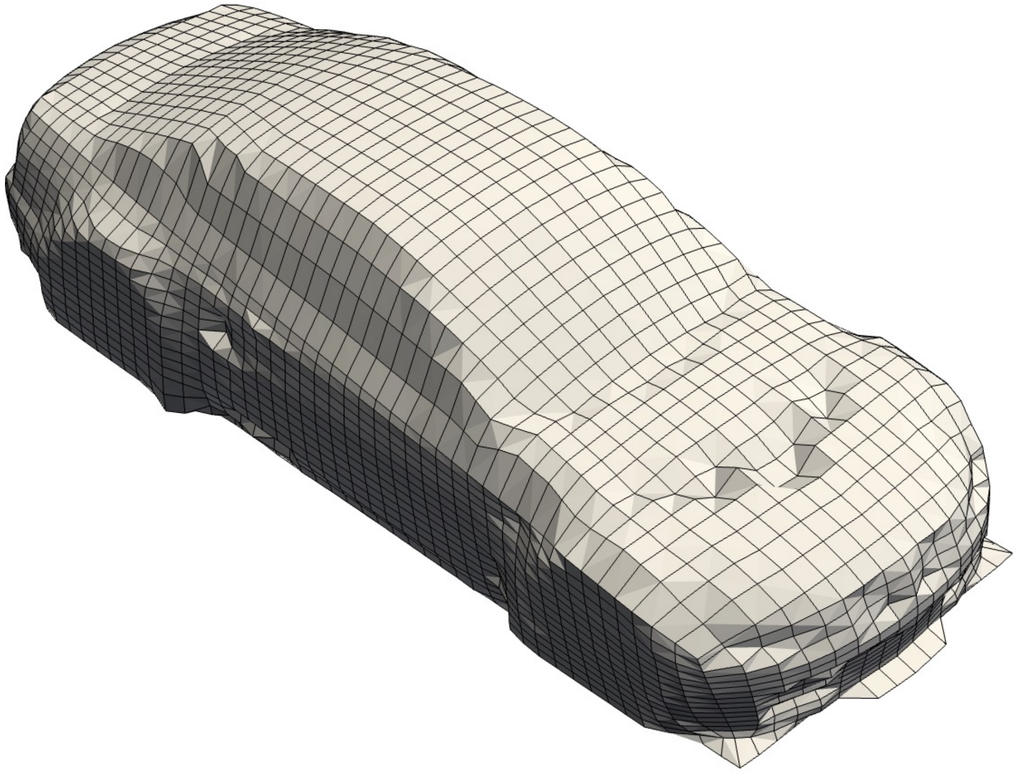

Lanxiang Xing


Yichen Di


Jianmin Wang


Mingsheng Long

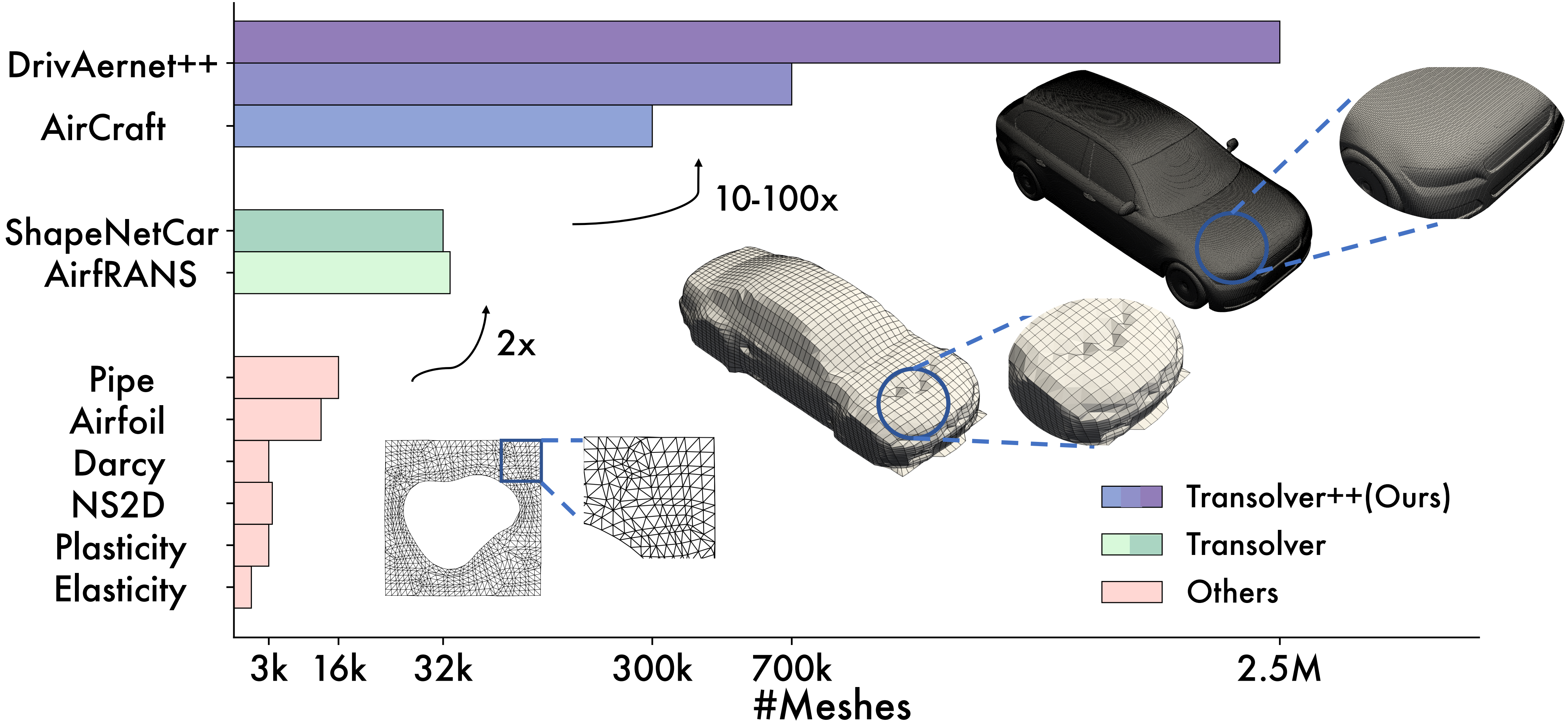
Difficulties on Applicability



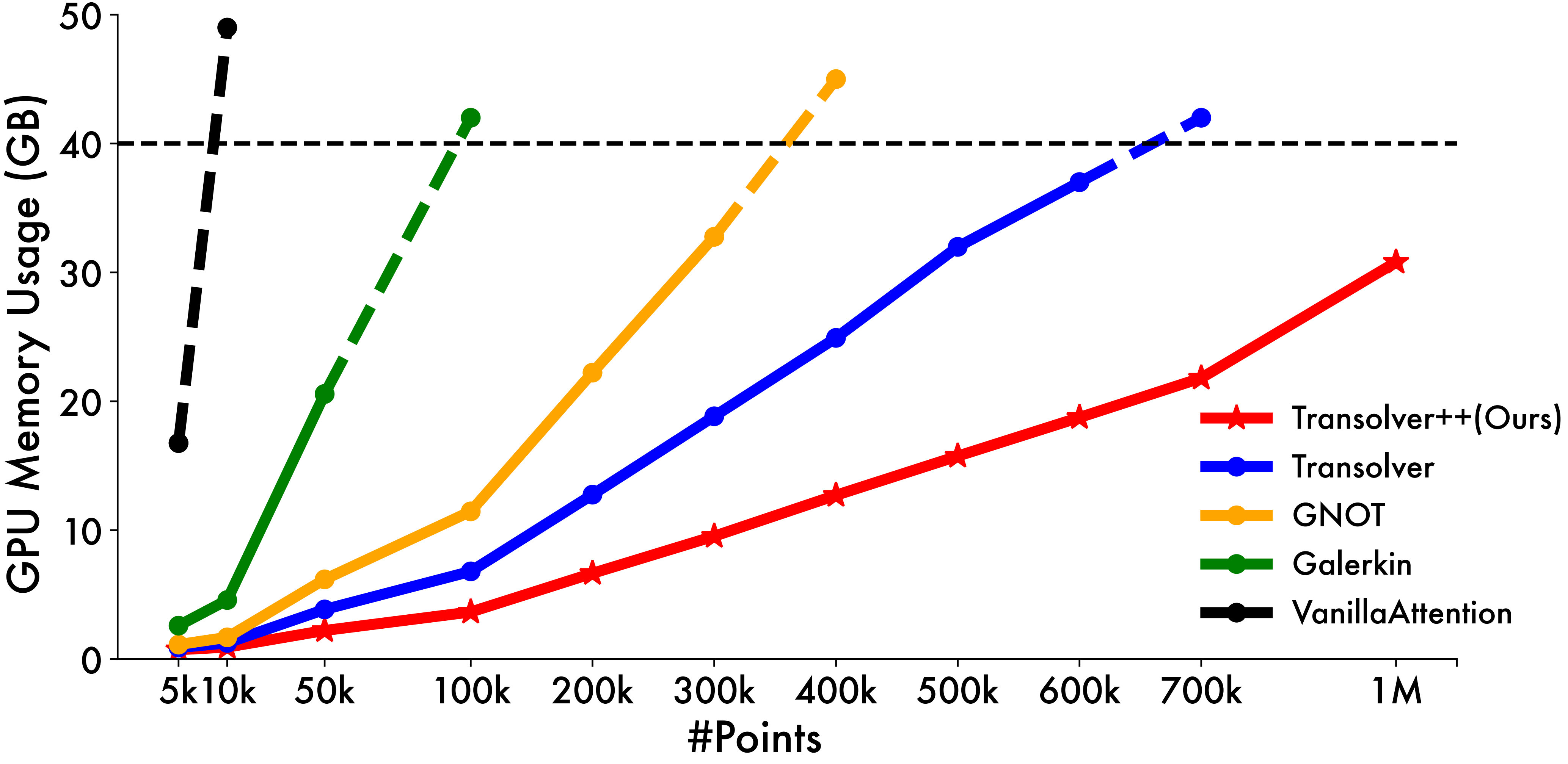
32k Mesh Points

2.5M Mesh Points

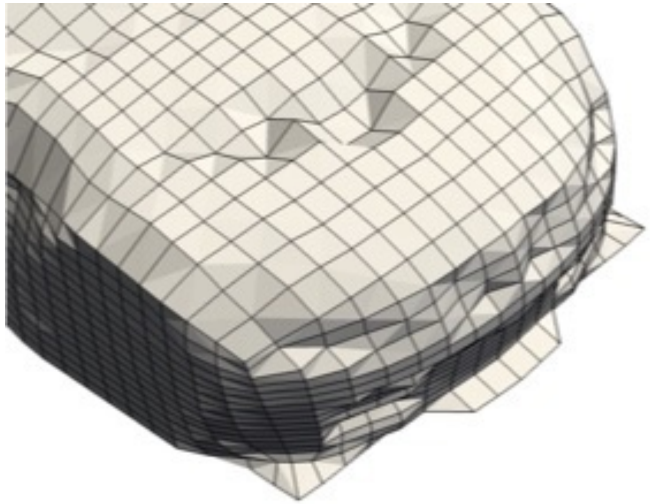
Difficulties on Applicability

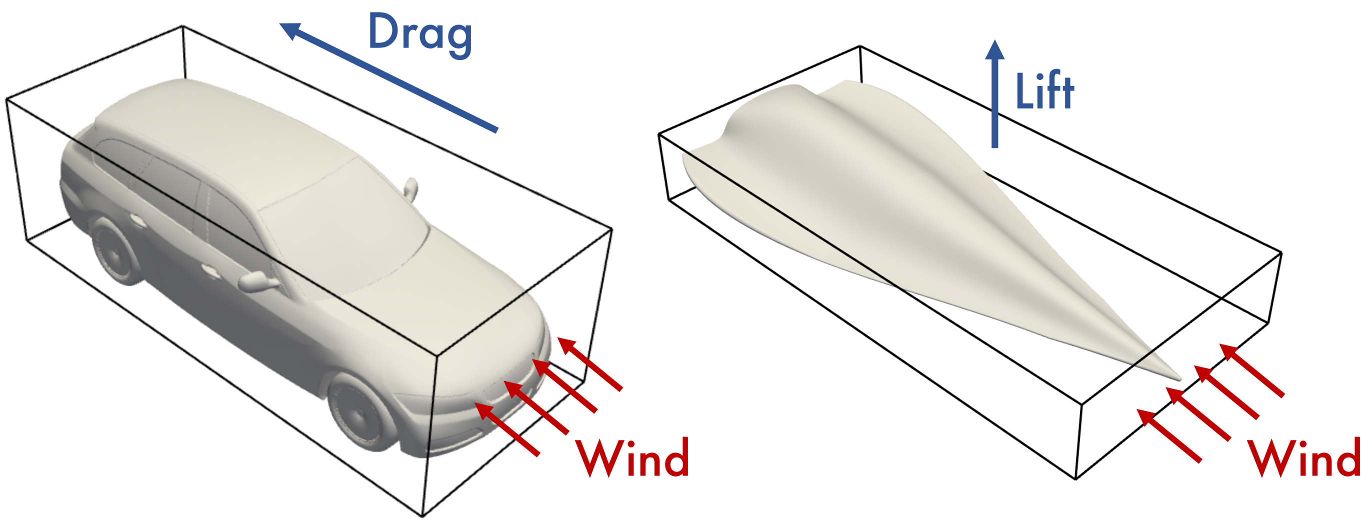


Difficulties on Applicability



Difficulties on Applicability





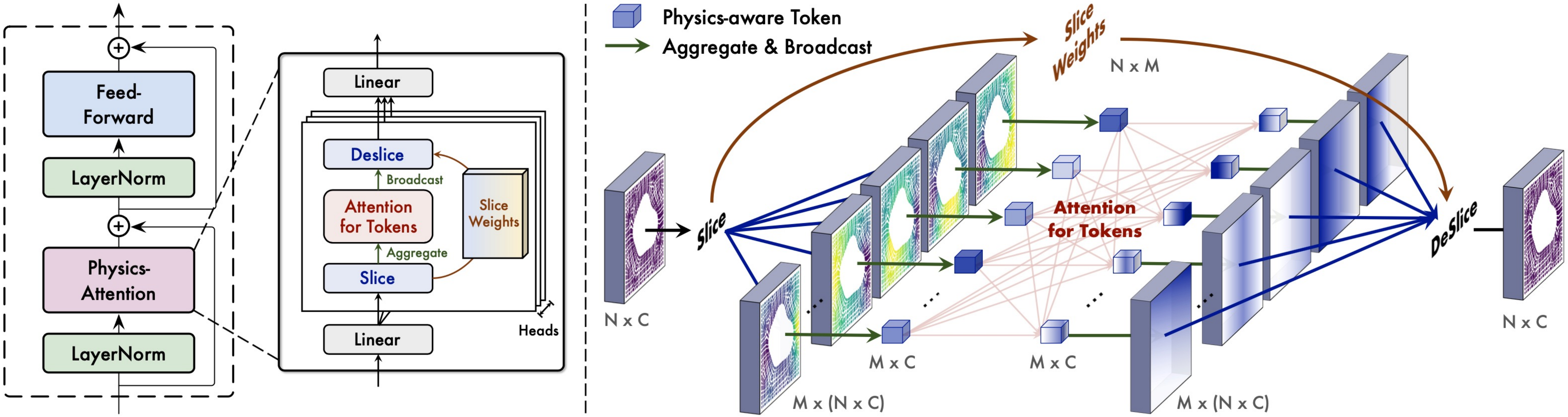
Large Geometrics In real-world applications

1. More complex geometrics with plenty of details

2. Deep models are expected to be Scalable

3. Models are expected to be more accurate

Revisiting Transolver

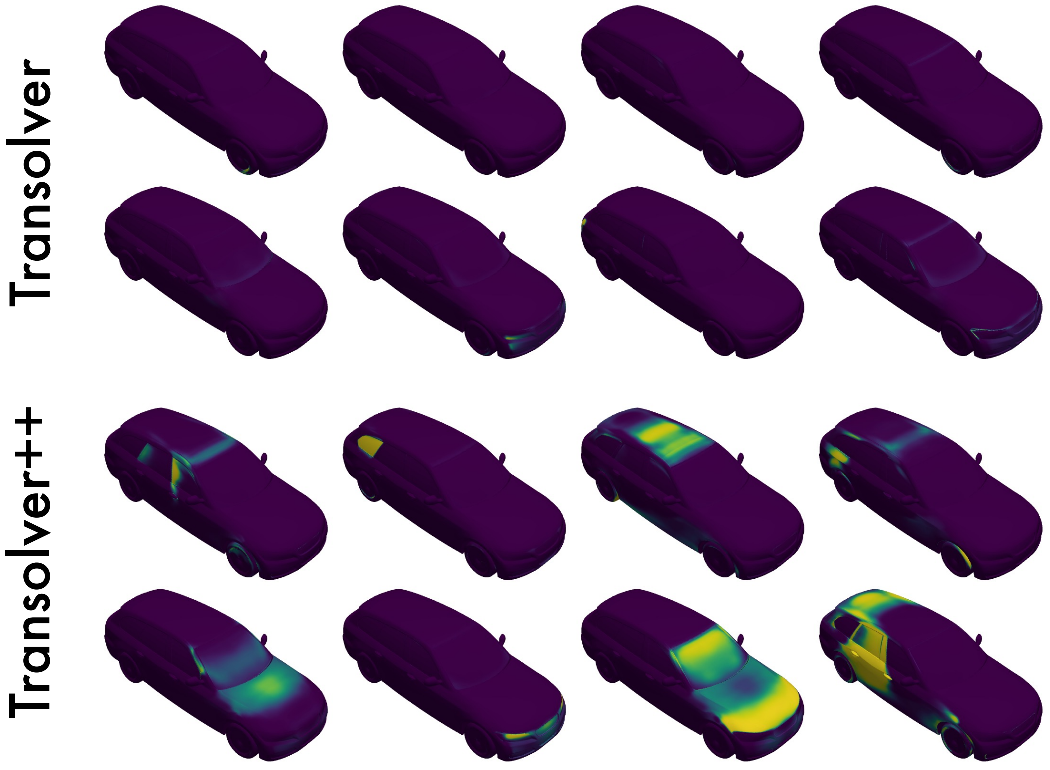


Transolver applies attention to learned physical states

① Mesh → physics ② Physics-Attention ③ Physics → Mesh

Challenges within Transolver

1. Homogeneous physical states



(b) Slice Weights Visualization

2. Efficiency Bottleneck

$$\textbf{Slice weights: } \mathbf{w} = \mathrm{Softmax}\left(\mathrm{Linear}(\mathbf{x})/\tau_0\right)$$

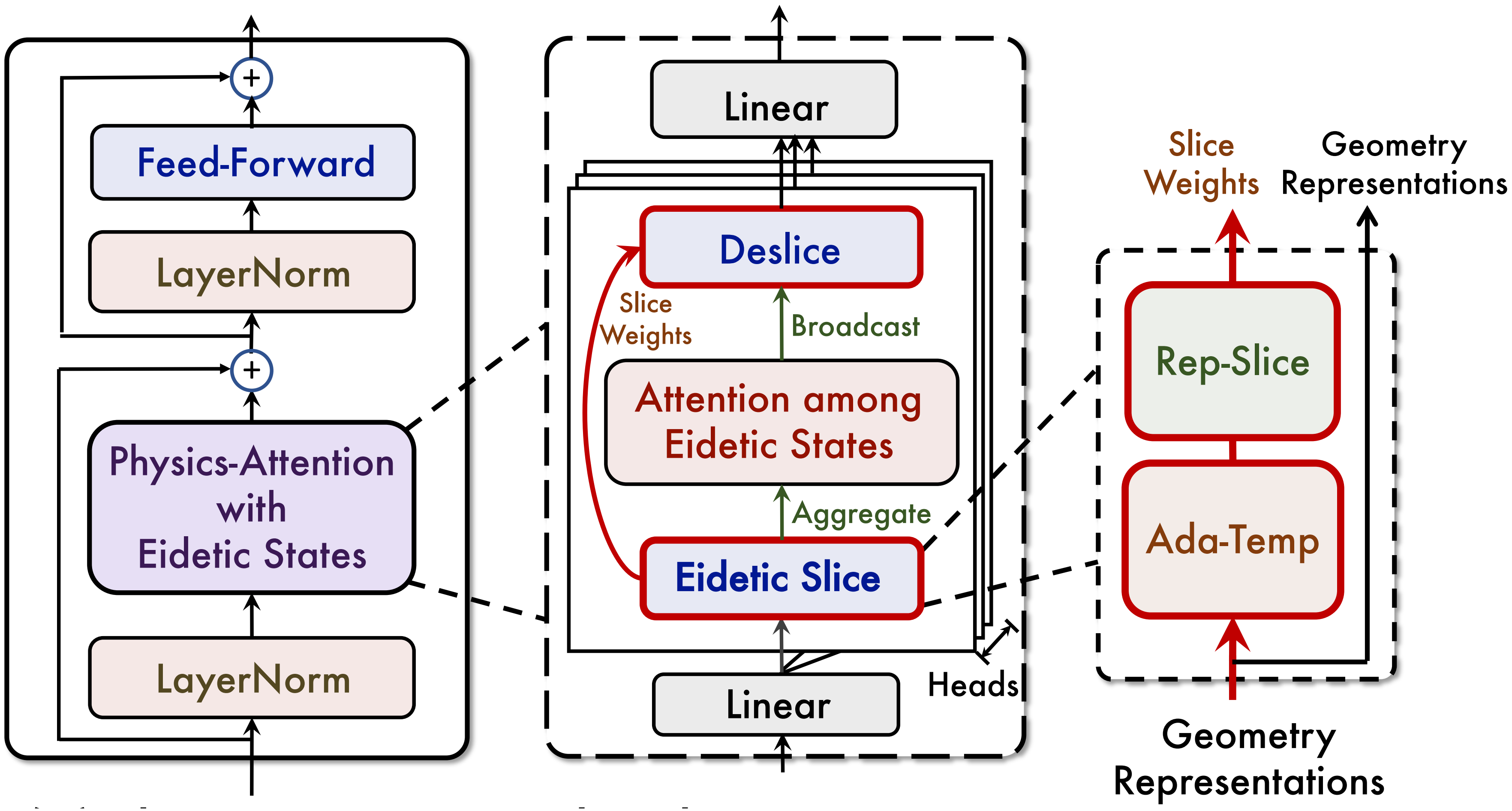
$$\textbf{Physical states: } \{\mathbf{s}_j\}_{j=1}^{M} = \left\{ \frac{\sum_{i=1}^{N} \mathbf{w}_{ij}\mathbf{x}_i}{\sum_{i=1}^{N} \mathbf{w}_{ij}} \right\}_{j=1}^{M}$$

- Even a single intermediate representation of one million mesh points will consume 2GB GPU memory

- Previous upper bound of geometry scale is 600k on one single GPU supported by Transolver

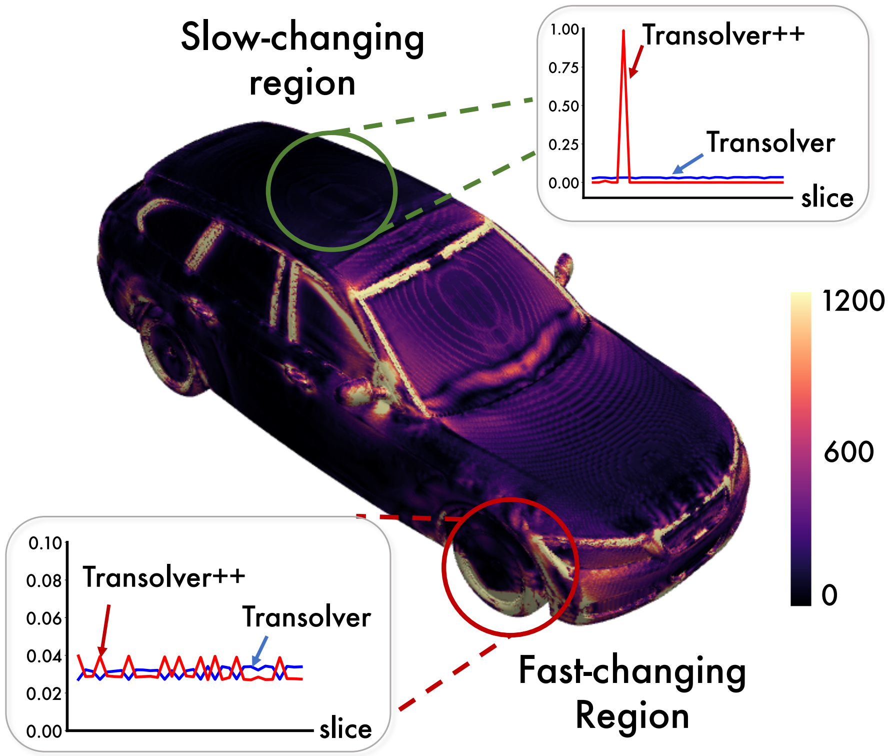
Solutions to challenges – Physics-Attention with Eidetic States

Architectural Design



Solutions to challenges – Physics-Attention with Eidetic States

Local Adaptive Mechanism



$$\textbf{Ada-Temp: } \tau = \{\tau_i\}_{i=1}^{N} = \{\tau_0 + \mathrm{Linear}(\mathbf{x}_i)\}_{i=1}^{N},$$

- Utilize the local properties of each mesh point
- Learns the uncertainty of each points
- Adaptively change the temperature of each point

Slice reparameterization

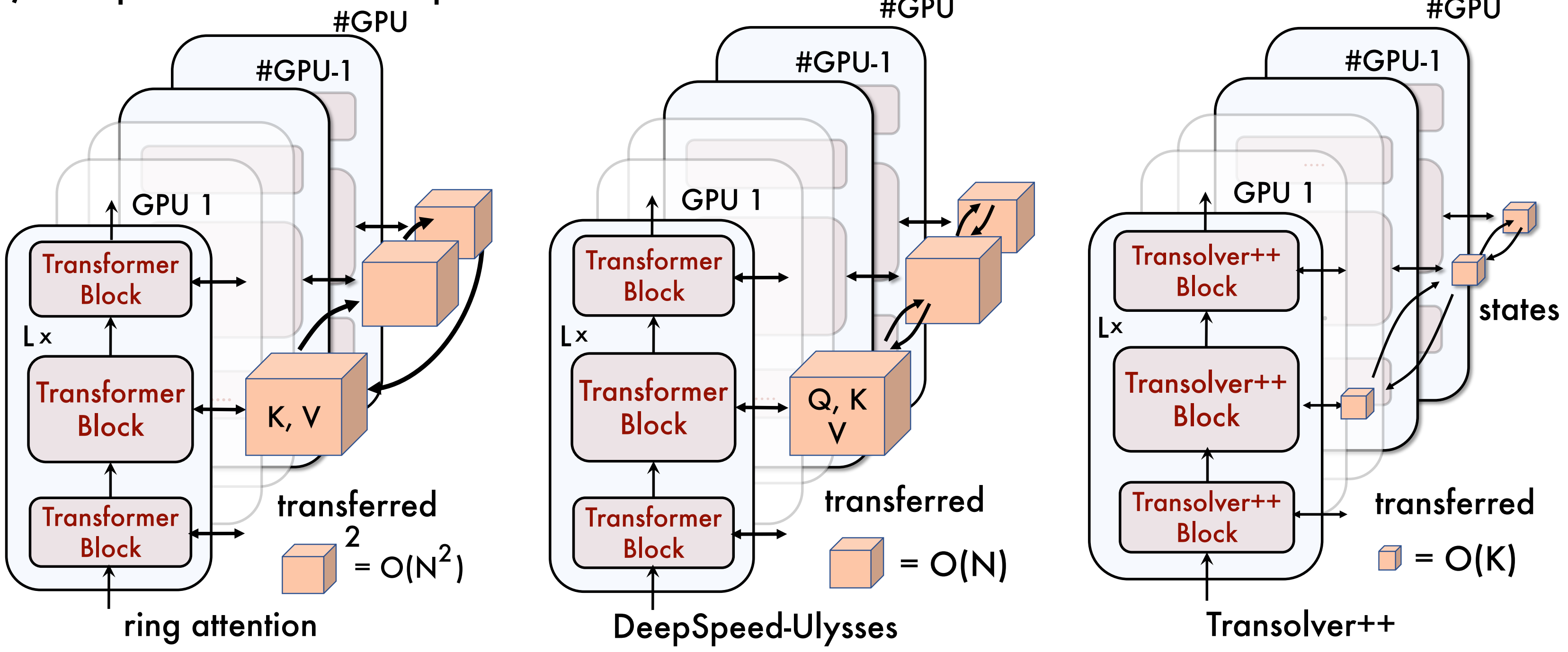
$$\textbf{Rep-Slice}(\mathbf{x}, \tau) = \mathrm{Softmax}\left(\frac{\mathrm{Linear}(\mathbf{x}) - \log(-\log \epsilon)}{\tau}\right), \tag{4}$$

Solutions to challenges – Parallel Transolver++

Parallel Formulation

$$\mathbf{s}_j = \frac{\sum_{i=1}^{N_1} \mathbf{w}_{ij}^{(1)} \mathbf{x}_i^{(1)} \oplus \cdots \oplus \sum_{i=1}^{N_{\#\text{gpu}}} \mathbf{w}_{ij}^{(\#\text{gpu})} \mathbf{x}_i^{(\#\text{gpu})}}{\sum_{i=1}^{N_1} \mathbf{w}_{ij}^{(1)} \oplus \cdots \oplus \sum_{i=1}^{N_{\#\text{gpu}}} \mathbf{w}_{ij}^{(\#\text{gpu})}}$$

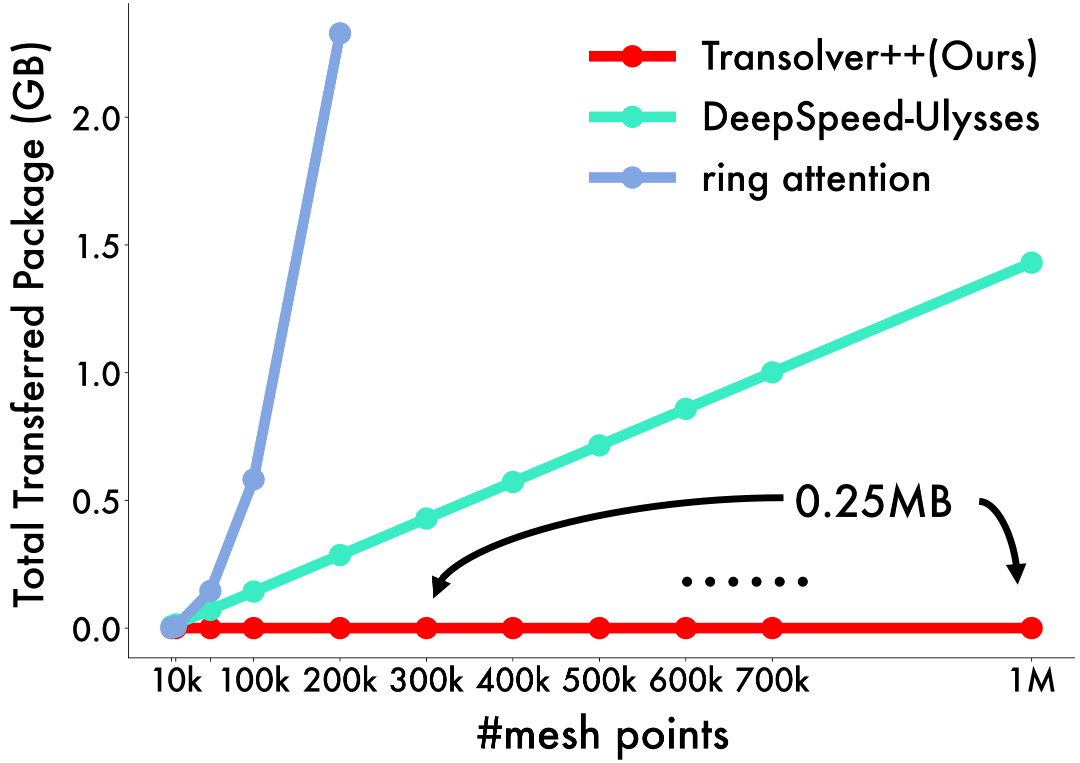
(a) Comparison with other parallel methods



Solutions to challenges – Parallel Transolver++

Overhead Analysis

(b) Scalability of Transferred Package

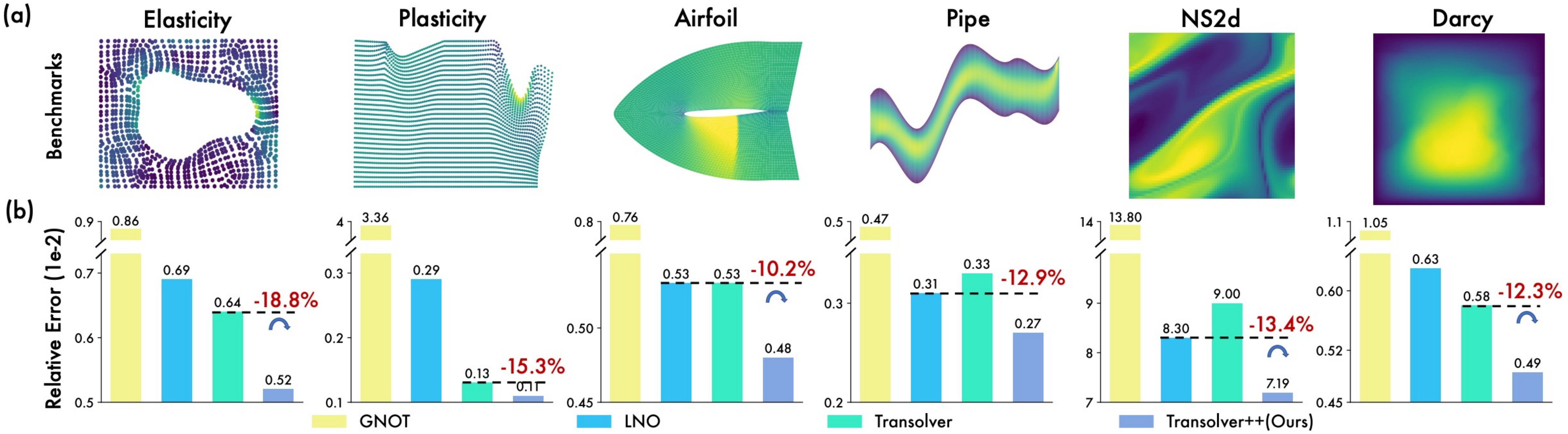


Further SpeedUp

Algorithm 1 Parallel Physics-Attention with Eidetic States

Input: Input features $\mathbf{x}^{(k)} \in \mathbb{R}^{N_k \times C}$ on the k-th GPU.
Output: Updated output features $\mathbf{x}'^{(k)} \in \mathbb{R}^{N_k \times C}$.
// drop f to save 50% memory.
Compute $\cancel{\mathbf{f}^{(k)}}, \mathbf{x}^{(k)} \leftarrow \text{Project}(\mathbf{x}^{(k)})$
Compute $\tau^{(k)} \leftarrow \tau_0 + \text{Ada-Temp}(\mathbf{x}^{(k)})$
Compute weights $\mathbf{w}^{(k)} \leftarrow \text{Rep-Slice}(\mathbf{x}^{(k)}, \tau^{(k)})$
Compute weights norm $\mathbf{w}_{\text{norm}}^{(k)} \leftarrow \sum_{i=1}^{N_k} \mathbf{w}_i^{(k)}$
Reduce slice norm $\mathbf{w}_{\text{norm}} \leftarrow \text{AllReduce}(\mathbf{w}_{\text{norm}}^{(k)})$ $\mathcal{O}(M)$
Compute eidetic states $\mathbf{s}^{(k)} \leftarrow \frac{\mathbf{w}^{(k)\top}\mathbf{x}^{(k)}\cancel{\mathbf{f}^{(k)}}}{\mathbf{w}_{\text{norm}}}$
Reduce eidetic states $\mathbf{s} \leftarrow \text{AllReduce}(\mathbf{s}^{(k)})$ $\mathcal{O}(MC)$
Update eidetic states $\mathbf{s}' \leftarrow \text{Attention}(\mathbf{s})$
Deslice back to $\mathbf{x}'^{(k)} \leftarrow \text{Deslice}(\mathbf{s}', \mathbf{w}^{(k)})$
Return $\mathbf{x}'^{(k)}$

Standard PDE-Solving Benchmarks



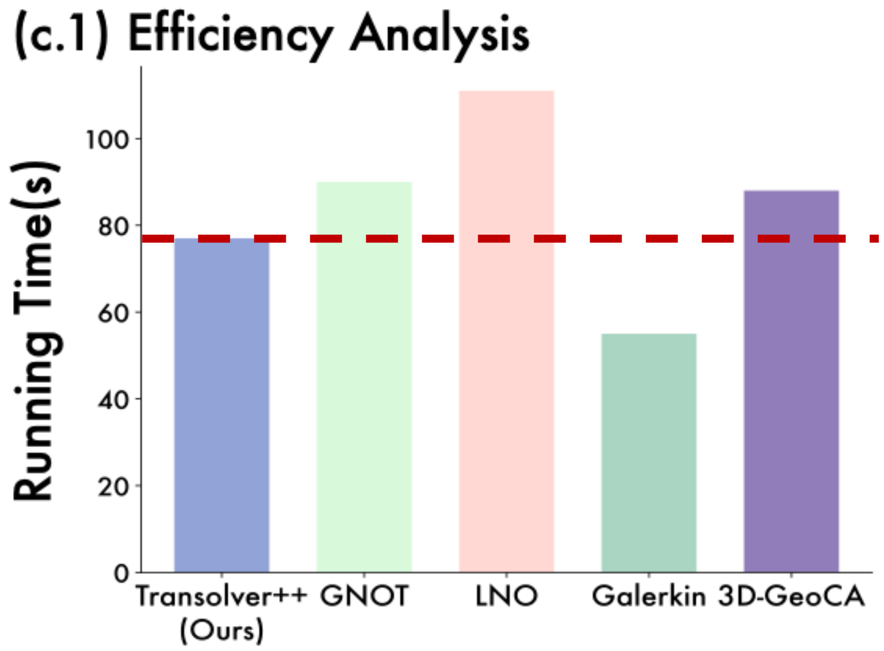
Transolver++ achieves averaged 13% error reduction than previous methods.

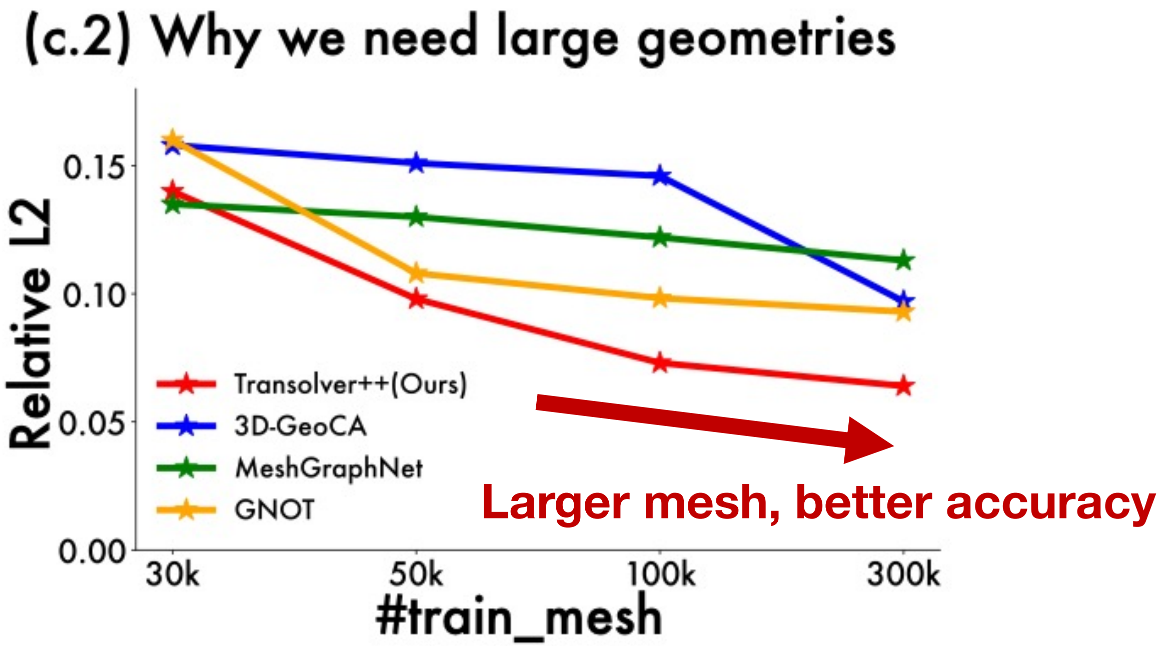
Industrial Applications

Model	DrivAerNet++ Full		DrivAerNet++ Surf			AirCraft		
	Volume ↓	Surf ↓	C_D ↓	R_L^2 ↑	Surf ↓	C_L ↓	R_L^2 ↑	Surf ↓
GraphSAGE (2017)	0.328	0.284	0.282	0.859	0.294	0.040	0.988	0.109
PointNet (2017)	0.285	0.478	0.301	0.831	0.237	0.095	0.982	0.169
Graph U-Net* (2019)	0.241	0.260	0.272	0.876	0.193	0.063	0.953	0.161
MeshGraphNet* (2021)	0.529	0.422	0.260	0.870	0.209	0.038	0.993	0.113
GNO* (2020a)	0.510	0.664	0.252	0.882	0.196	0.031	0.991	0.129
Galerkin* (2021)	0.234	0.274	0.267	0.792	0.235	0.069	0.879	0.118
Geo-FNO* (2022)	0.718	0.892	0.288	0.831	0.291	0.243	0.903	0.395
GINO (2023a)	0.586	0.638	0.323	0.725	0.220	0.047	0.983	0.133
GNOT* (2023)	0.174	0.171	0.158	0.901	0.167	0.033	0.991	0.093
LNO* (2024)	0.180	0.203	0.208	0.855	0.195	0.091	0.992	0.137
3D-GeoCA* (2024)	0.389	0.224	0.205	0.883	0.175	<u>0.022</u>	0.993	0.097
Transolver* (2024)	<u>0.173</u>	<u>0.167</u>	<u>0.061</u>	<u>0.931</u>	<u>0.145</u>	0.037	<u>0.994</u>	<u>0.092</u>
Transolver++ (Ours)	**0.154**	**0.146**	**0.036**	**0.997**	**0.110**	**0.014**	**0.999**	**0.064**
Relative Promotion	11.0%	12.6%	41.0%	-	24.1%	36.3%	-	30.4%

Transolver++ achieves over 20% error reduction.

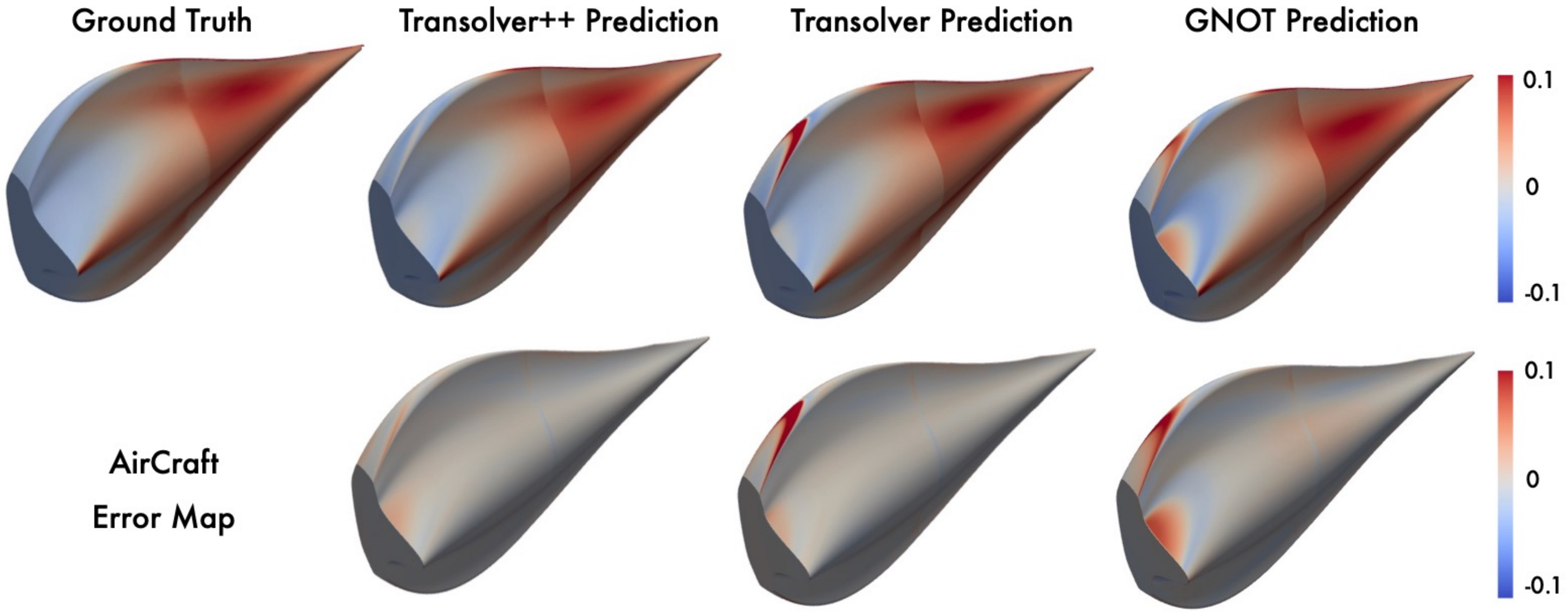
Efficiency and Scalability



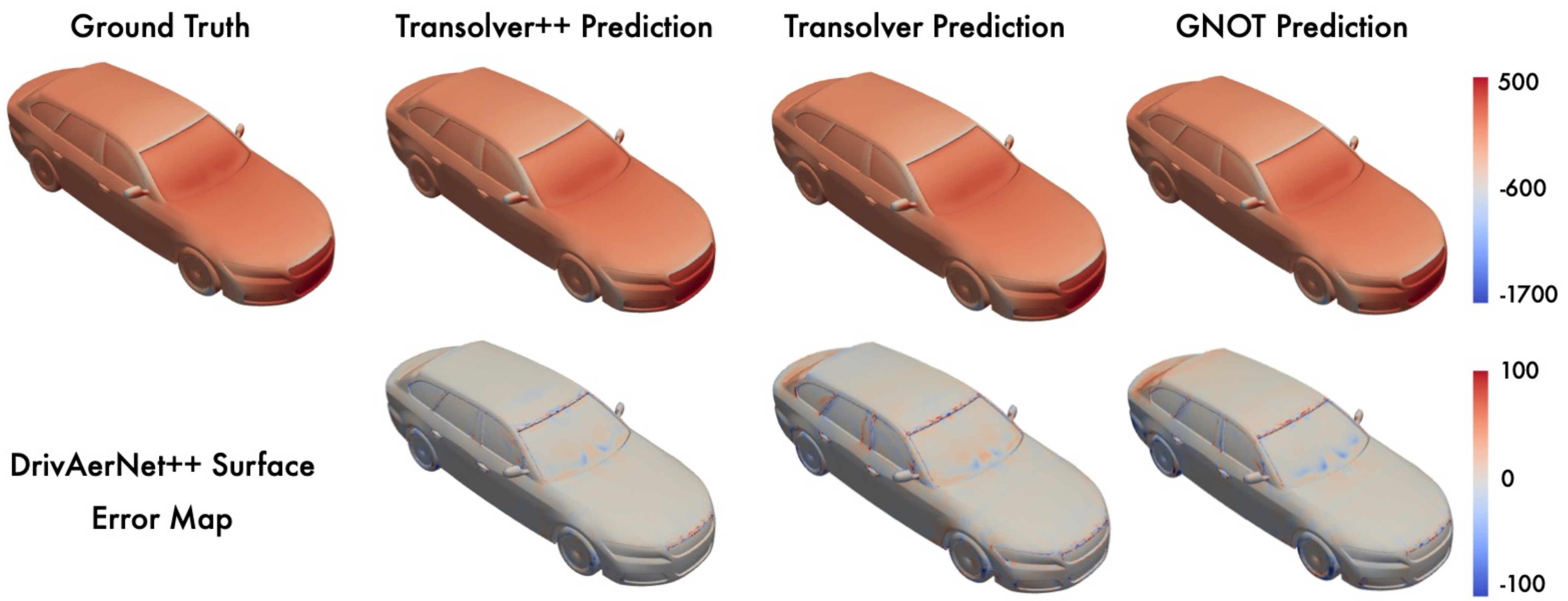


Transolver++ strikes a favorable balance between performance and efficiency.

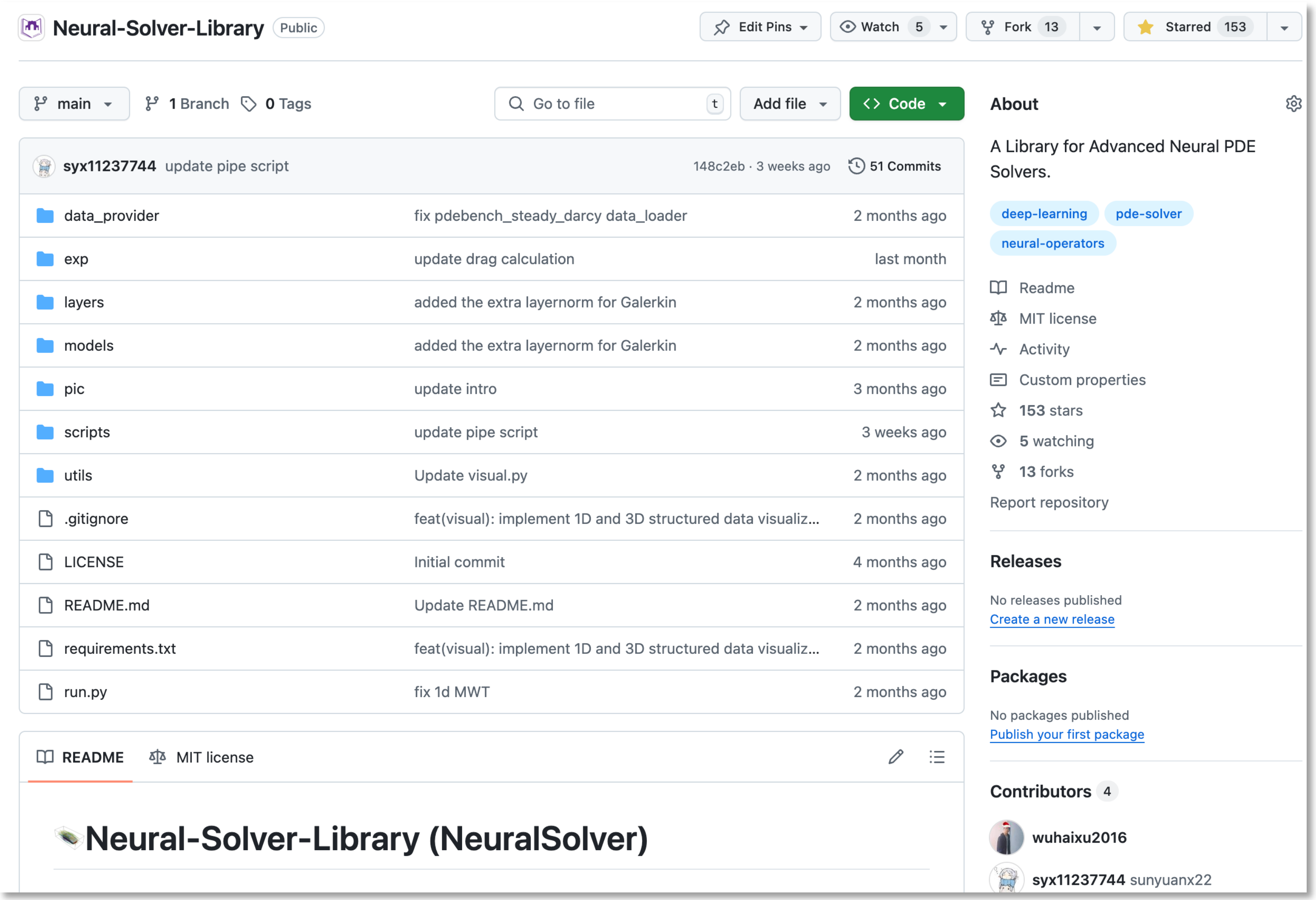
Showcases


Ground Truth
Transolver++ Prediction
Transolver Prediction
GNOT Prediction
0.1
0
-0.1
AirCraft
Error Map
0.1
0
-0.1

Showcases

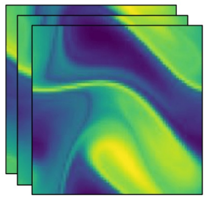

Ground Truth
Transolver++ Prediction
Transolver Prediction
GNOT Prediction
500
-600
-1700
DrivAerNet++ Surface
Error Map
100
0
-100

Neural-Solver-Library

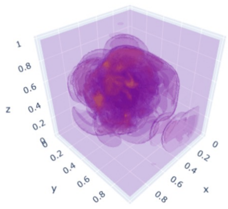


Code Link: *https://github.com/thuml/Neural-Solver-Library*

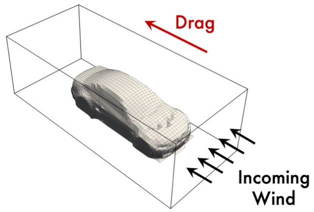
- ✓ 17 different PDE solvers
- ✓ 6 standard benchmarks, PDEBench and design tasks



Task 1: Standard



Task 2: PDEBench



Task 3: ShapeNet Car

Welcome to join us and add a new feature to this Library!

Code for NeuralSolver

Acknowledgement

Jianmin Wang

Mingsheng Long

NVIDIA

physicsnemo

SIEMENS

长按关注，获取最新资讯

Hang Zhou

Yuezhou Ma

Huakun Luo

Yuanxu Sun

Huikun Weng

Haowen Wang