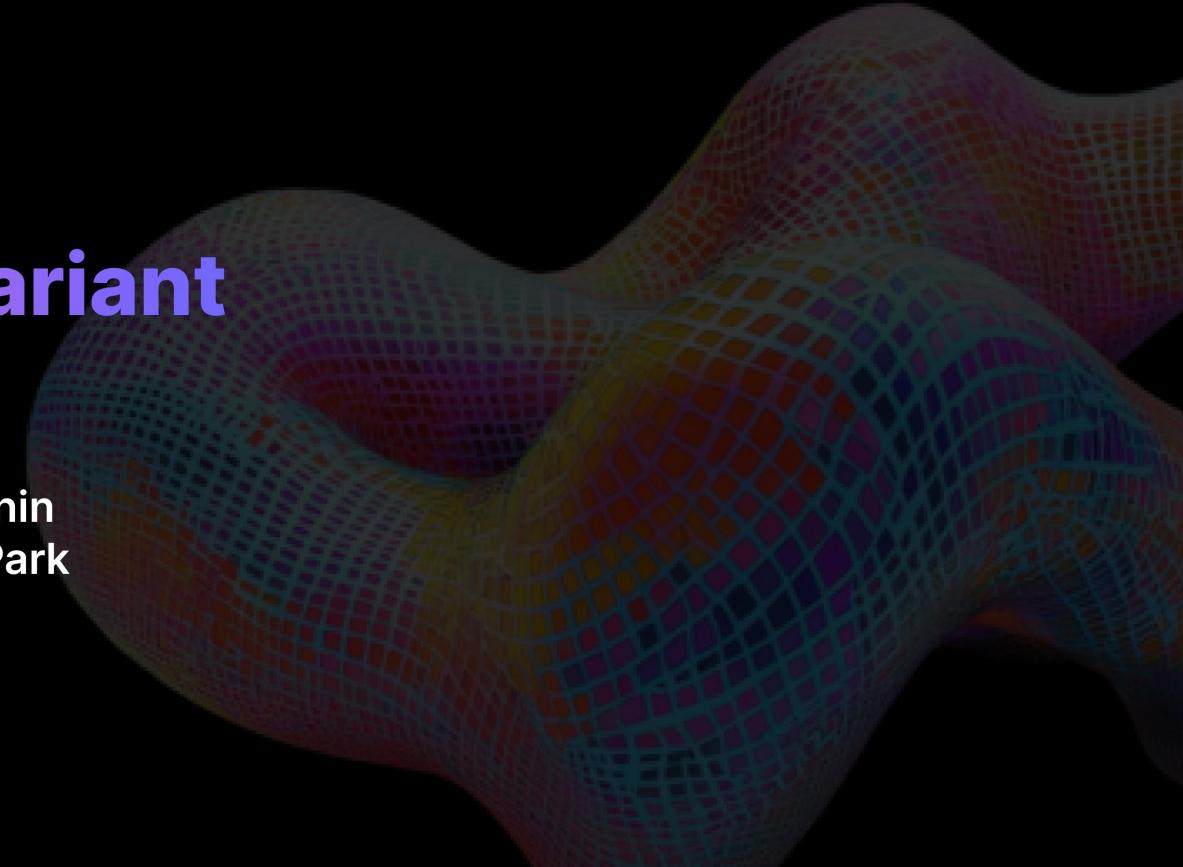


**ICML 2025**

# **Thickness-aware E(3)-Equivariant 3D Mesh Neural Network**

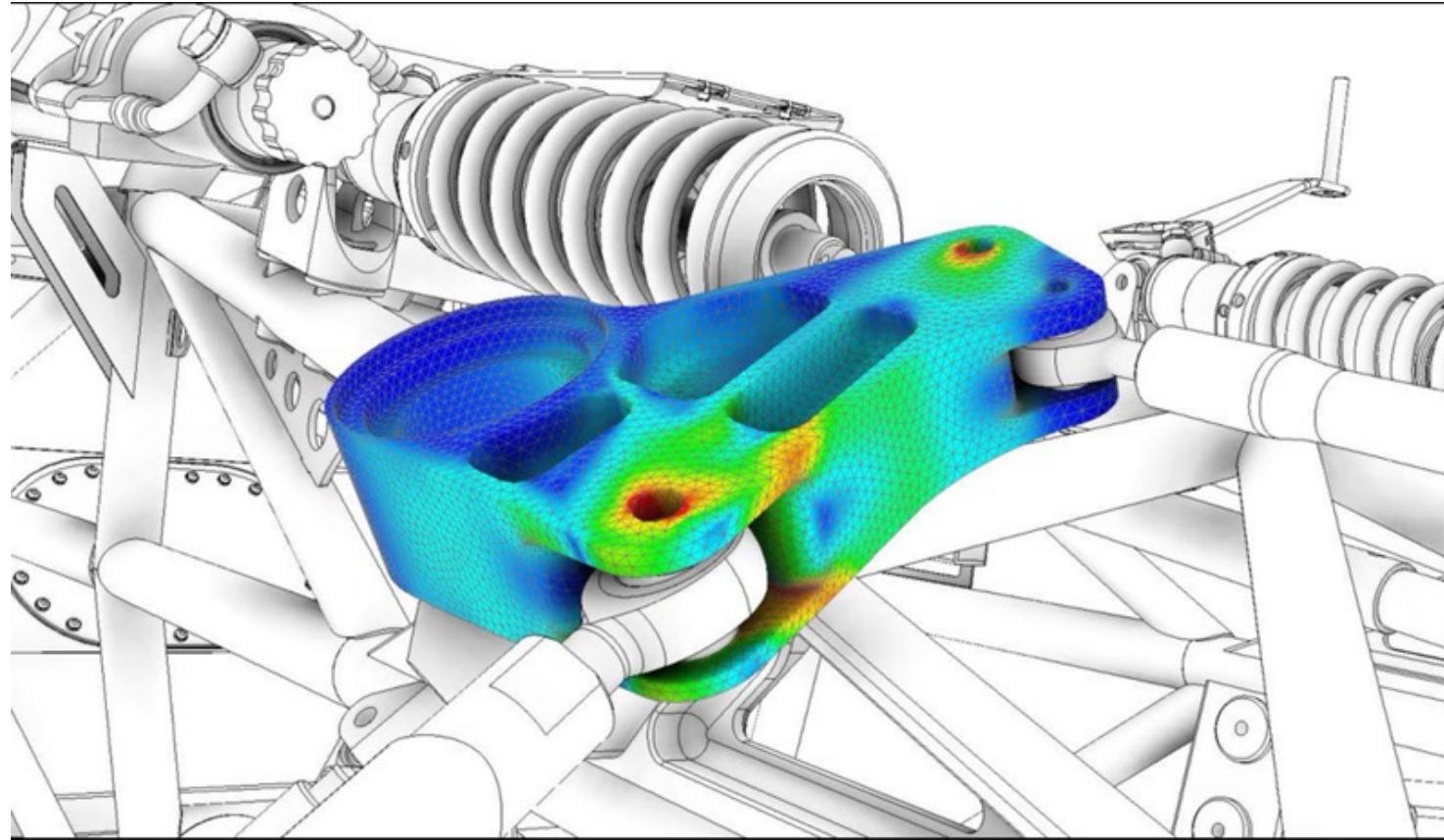
Sungwon Kim, Namkyeong Lee, Yunyoung Doh, Seungmin Shin  
Guimok Cho, Seung-Won Jeon, Sangkook Kim, Chanyoung Park

Presenter: Sungwon Kim



# WHY 3D SIMULATION MATTERS

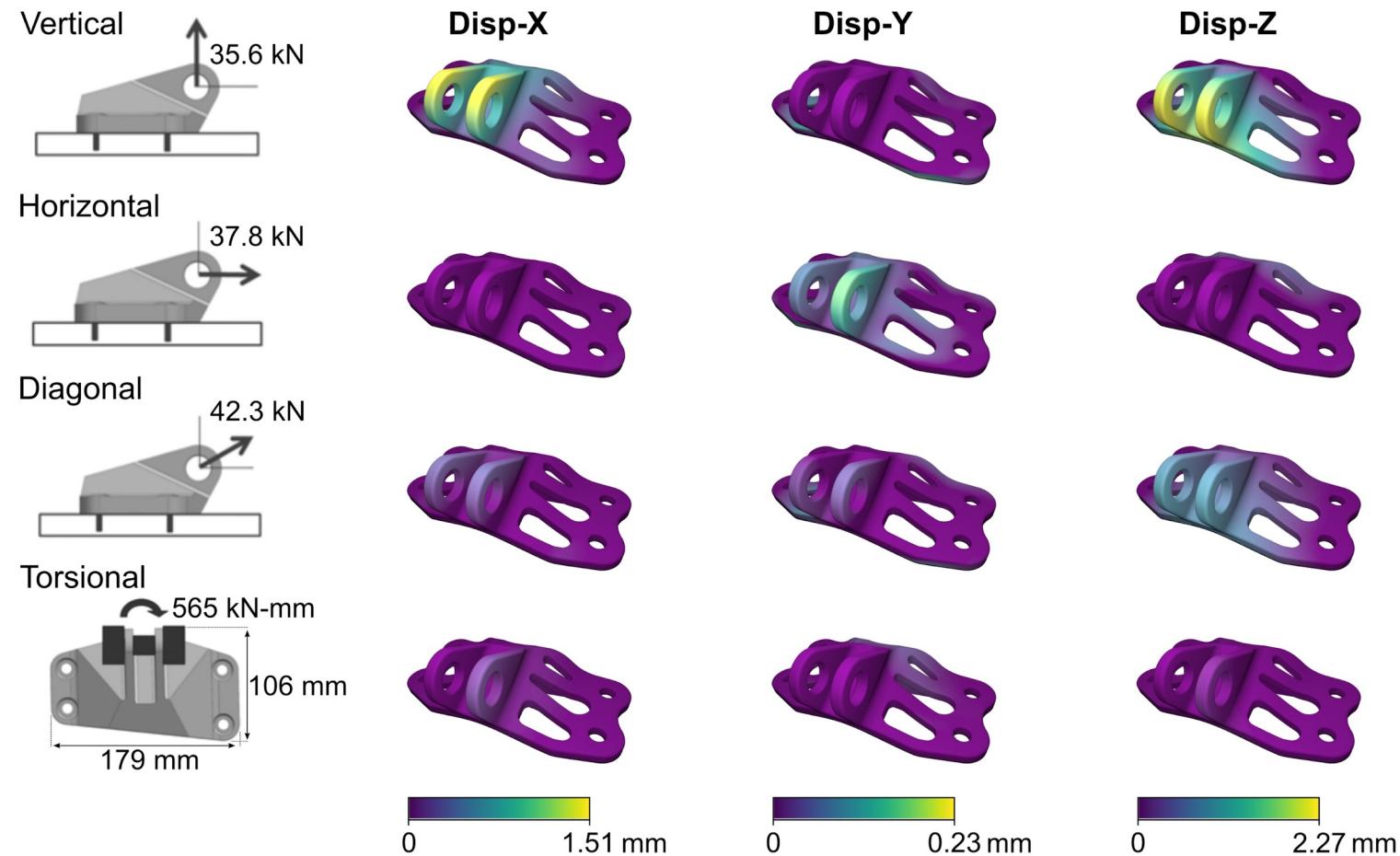
- The role of static analysis in manufacturing



**Analysis & Simulation:** Identify **potential product risks** under real-world conditions **before moving into production.**

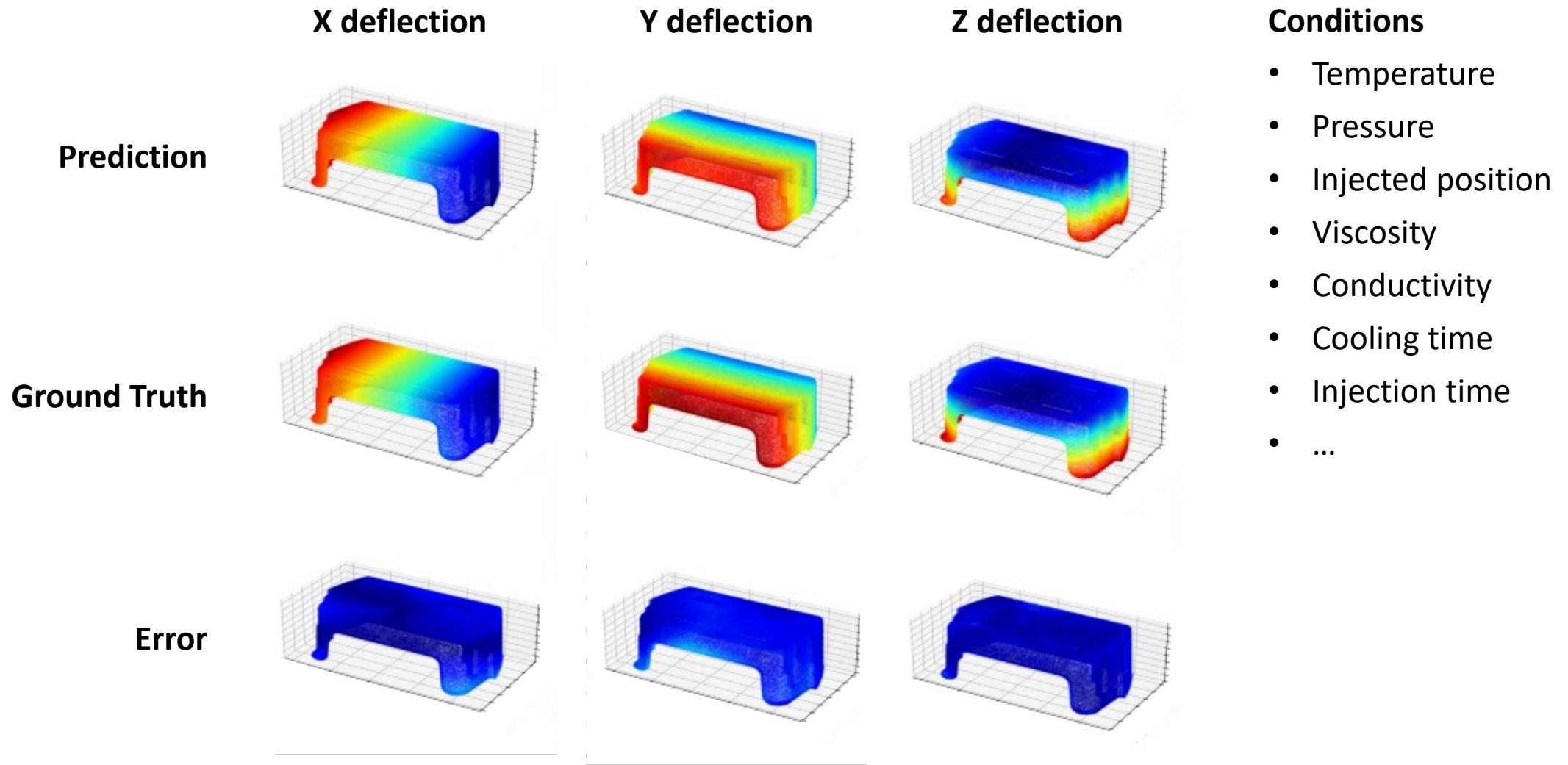
# WHY 3D SIMULATION MATTERS

- The role of static analysis in manufacturing



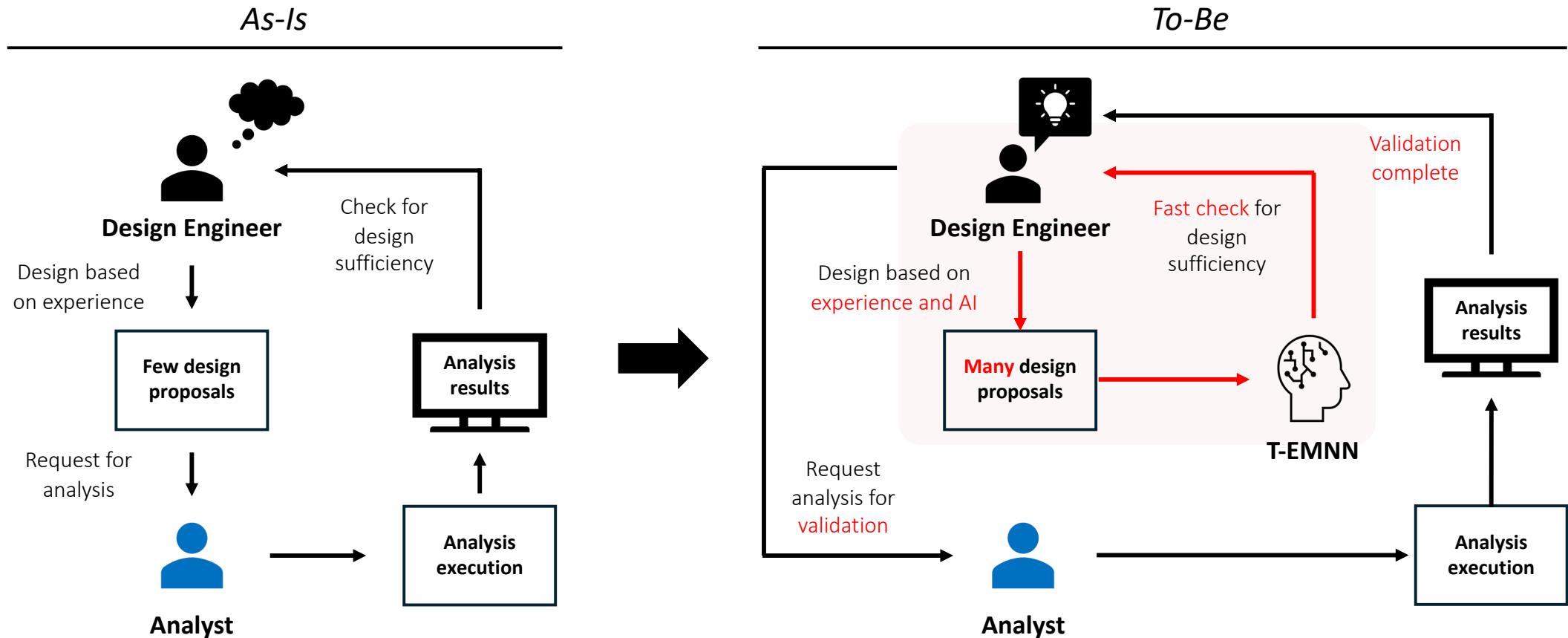
# WHY 3D SIMULATION MATTERS

## ► The role of static analysis in manufacturing



# WHY 3D SIMULATION MATTERS

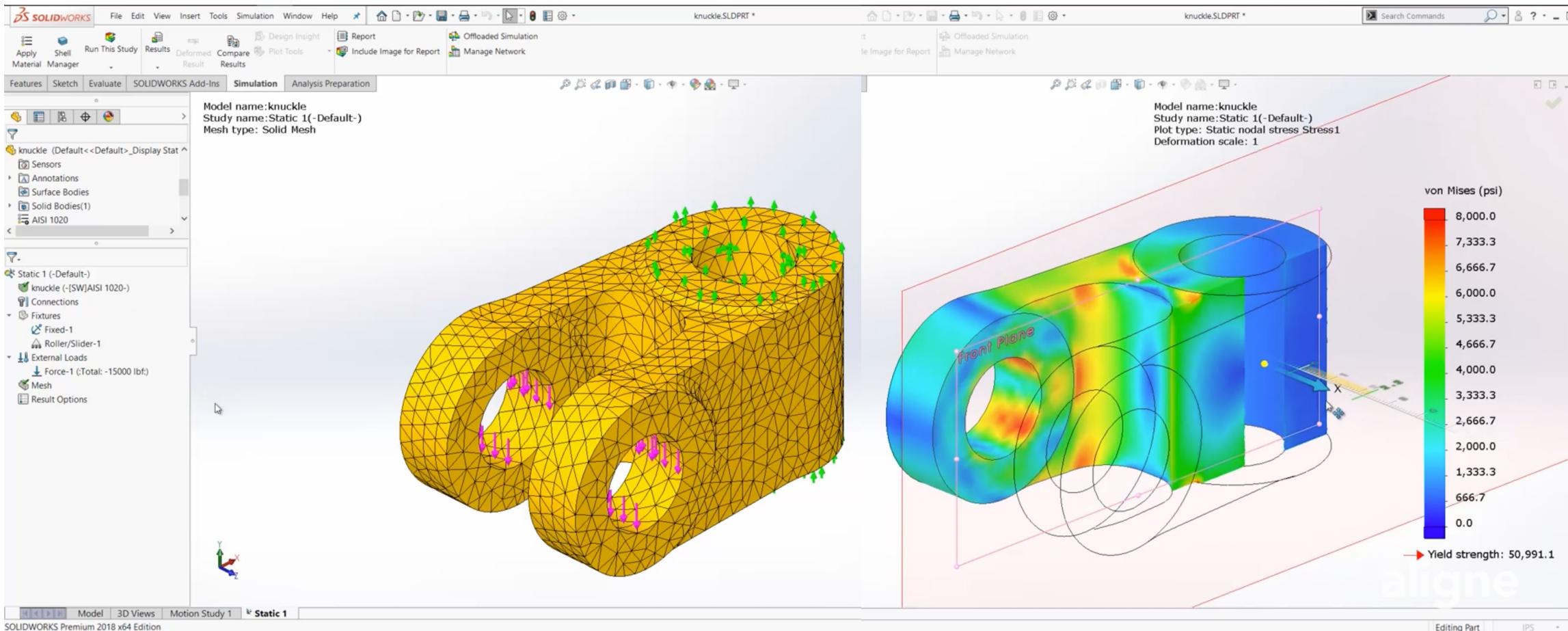
- The role of static analysis in manufacturing



3D simulation is the key to building **better products, faster and cheaper**.

# CHALLENGE: CONVENTIONAL SOLVERS

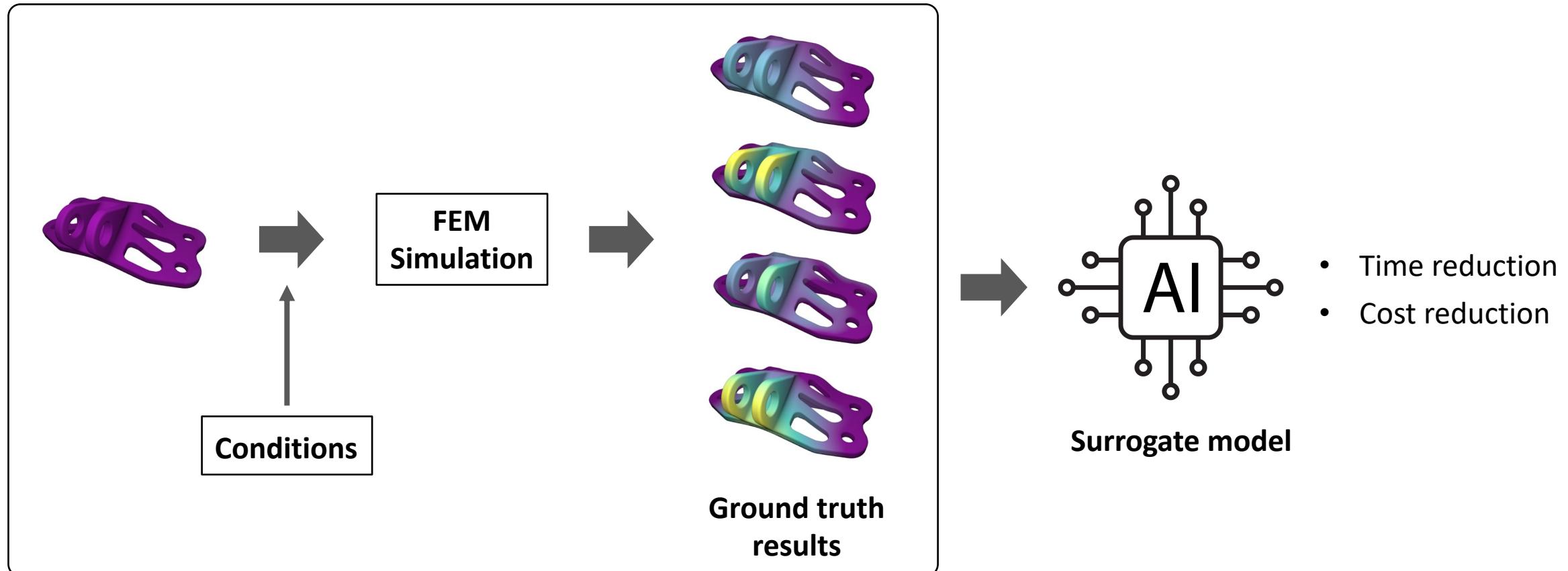
- Finite Element Method (FEM): High Computational Cost



Performing a single FEM analysis involves **high computational costs and extended runtimes**.

# CHALLENGE: CONVENTIONAL SOLVERS

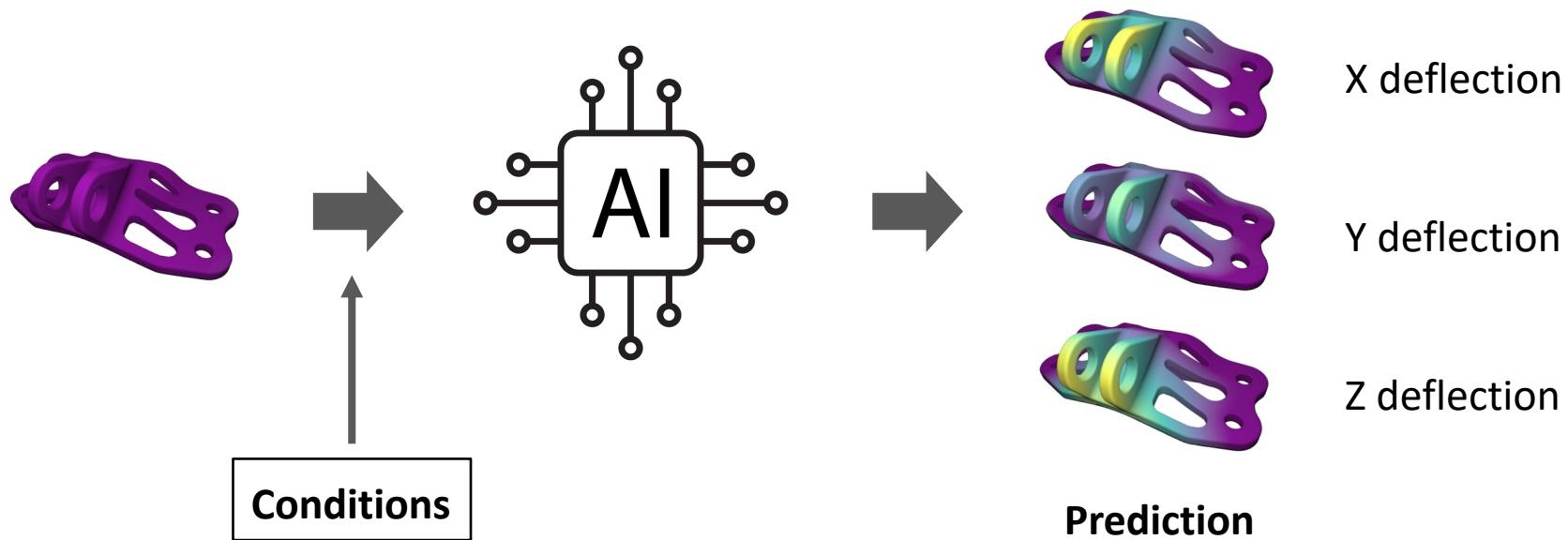
- Finite Element Method (FEM): High Computational Cost



Learning-based AI models can serve as powerful surrogates, significantly reducing the high computational cost of FEM simulations.

# TASK

- Node-level deflection (3d) prediction



# 3D REPRESENTATION

## ► 3D Mesh: Vertices, Edges, and Faces

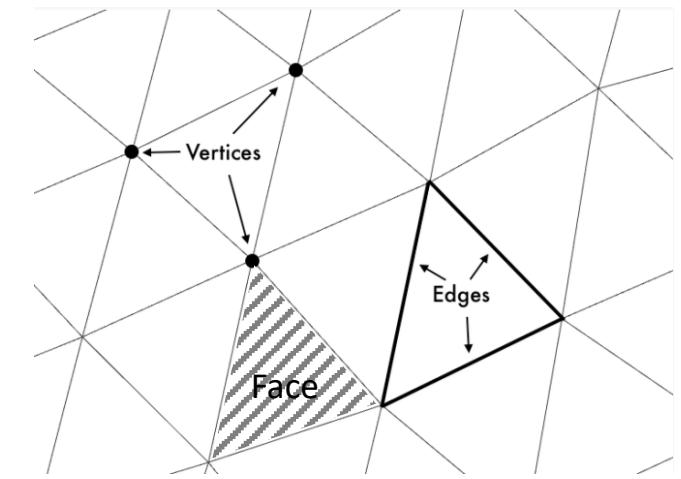
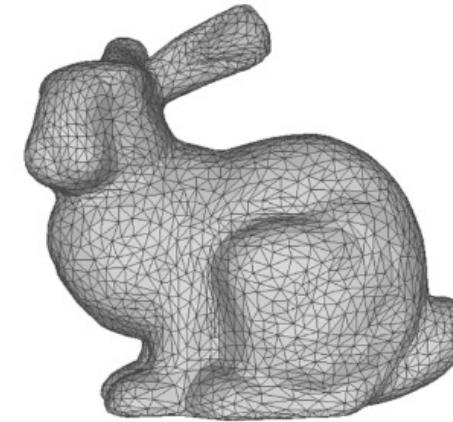
Point cloud



Voxel



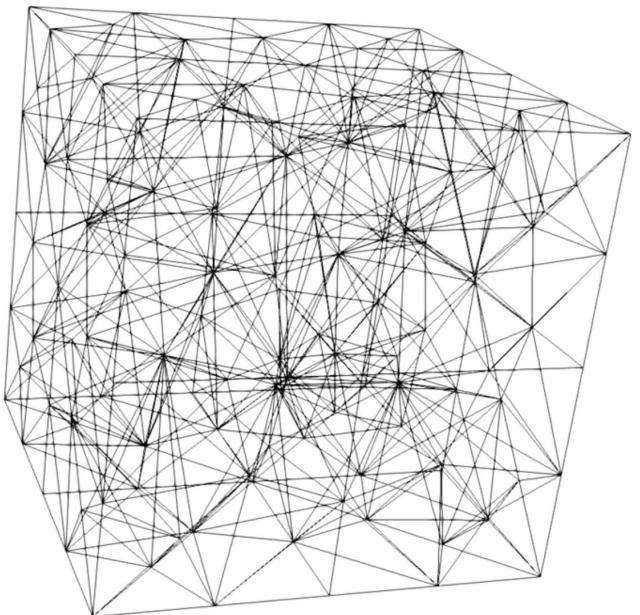
Polygon mesh



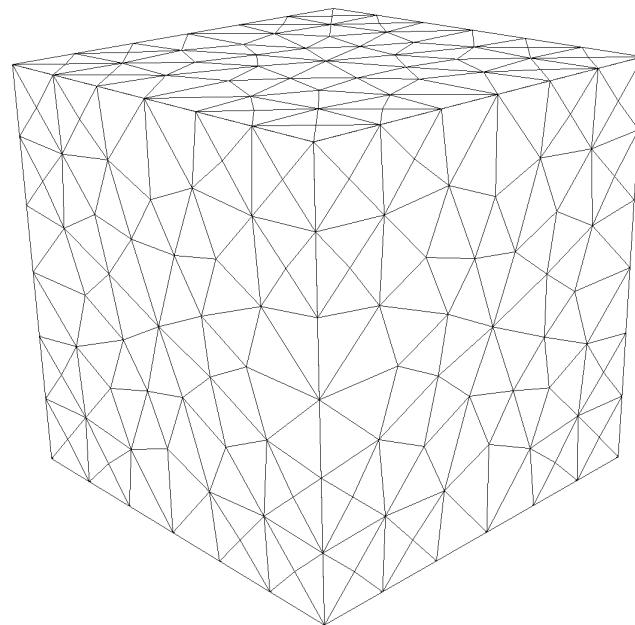
3D Mesh is one the best way to **describe surfaces and volumes of object!**

# 3D REPRESENTATION

- 3D Mesh: Volume mesh vs. Surface mesh



**Volume mesh**



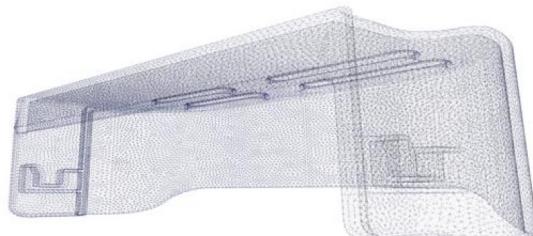
**Surface mesh**

# 3D REPRESENTATION

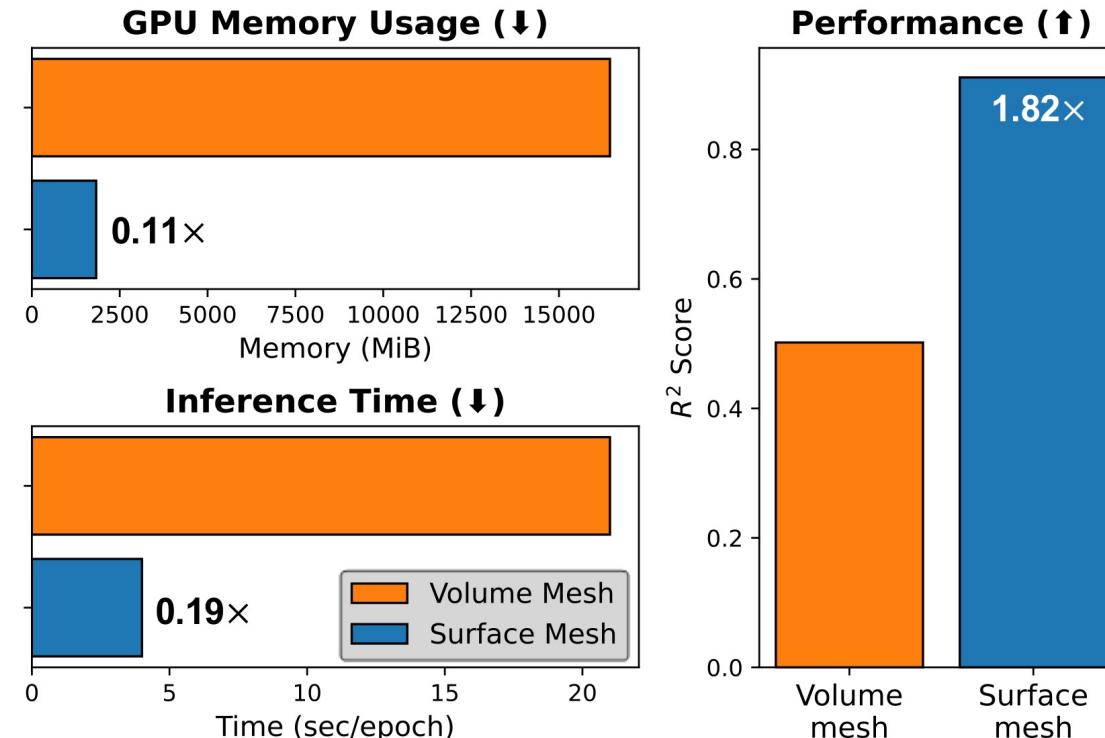
## ► 3D Mesh: Volume mesh vs. Surface mesh



Volume mesh



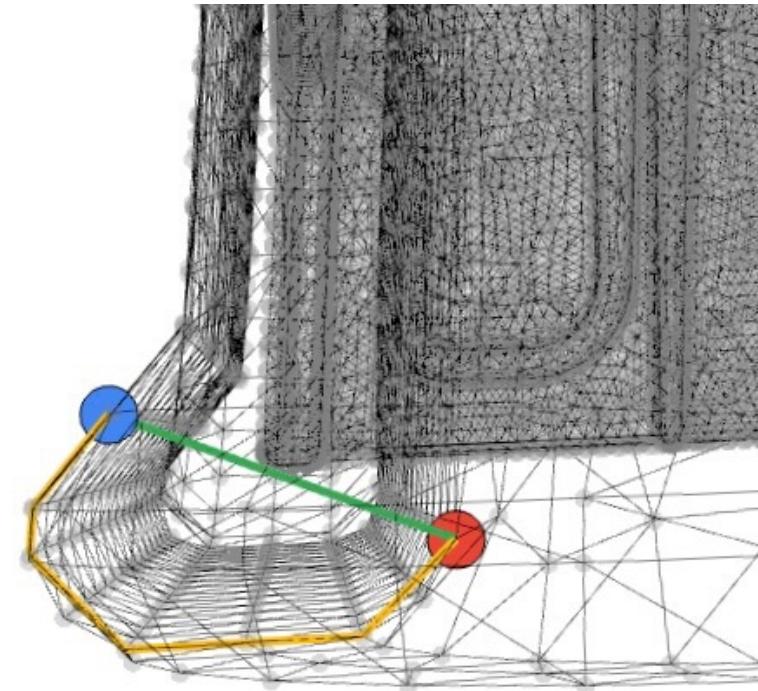
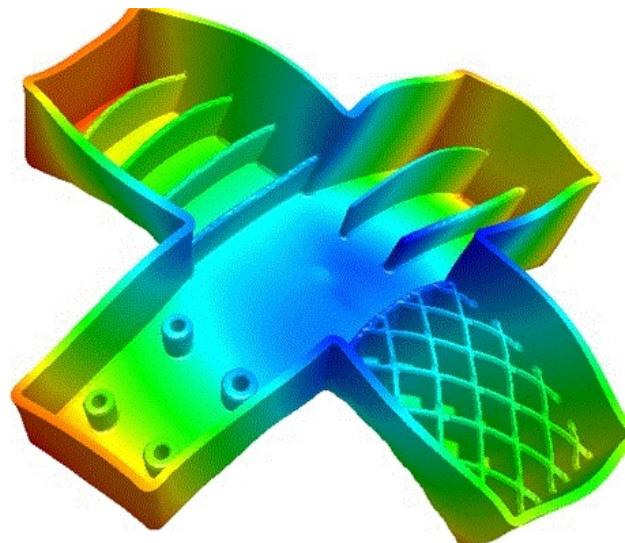
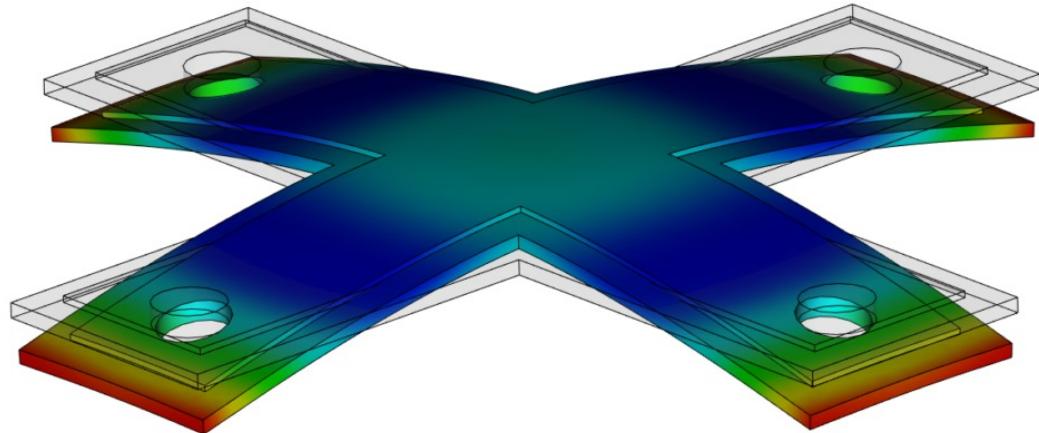
Surface mesh



Surface meshes are **significantly faster, more cost-effective**, and better suited for modeling the **geometric properties** of a shape.

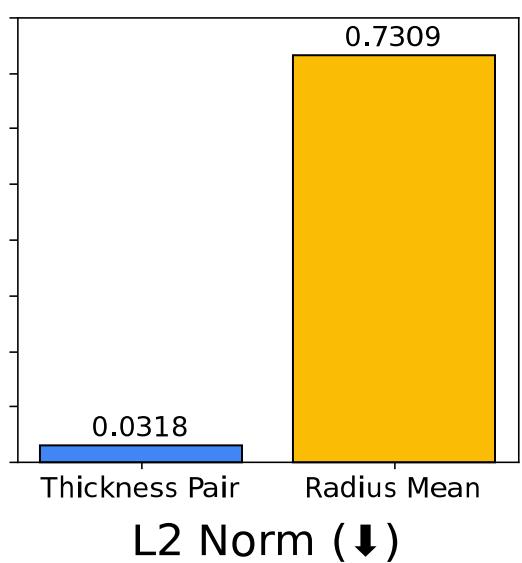
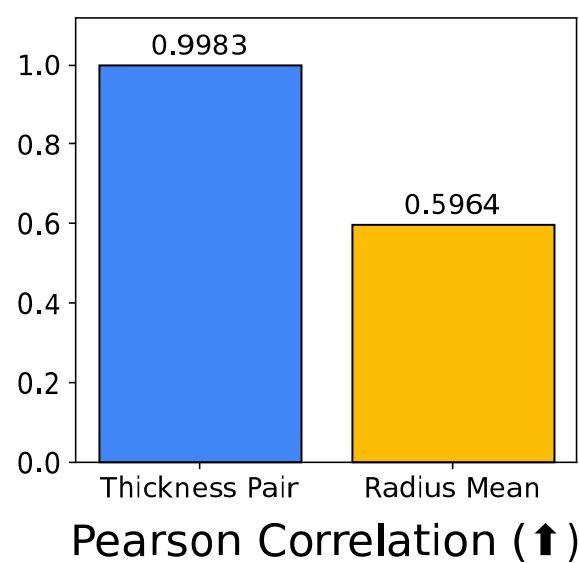
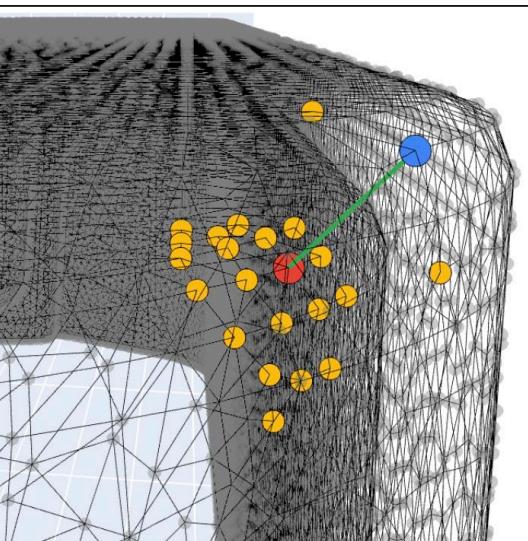
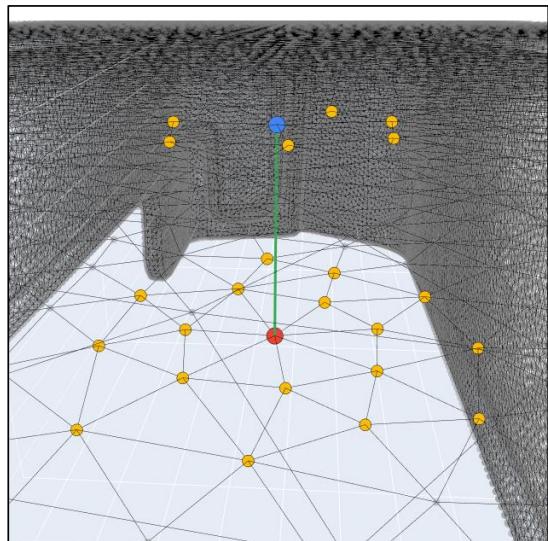
# PROBLEM WITH SURFACE MESH

- Ignoring Critical Thickness Information



# PROBLEM WITH SURFACE MESH

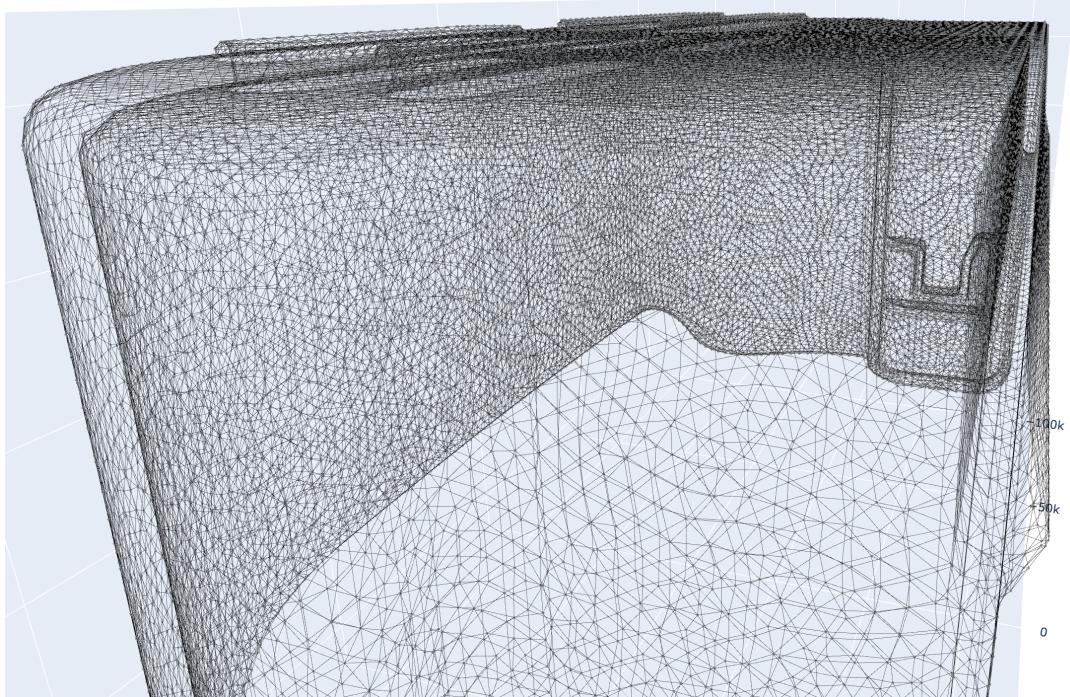
- Ignoring Critical Thickness Information



Surface meshes face **challenges in modeling the interactions between thickness pair** due to the lack of connections between opposing surfaces within the mesh

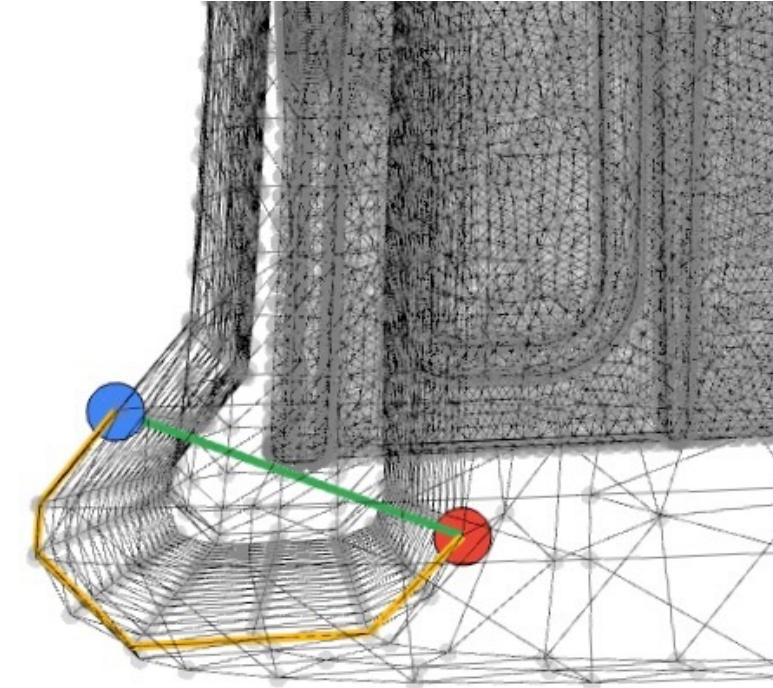
# OUR MOTIVATION

- ▶ Bringing the gap between surface mesh efficiency and physical accuracy



Surface mesh

+

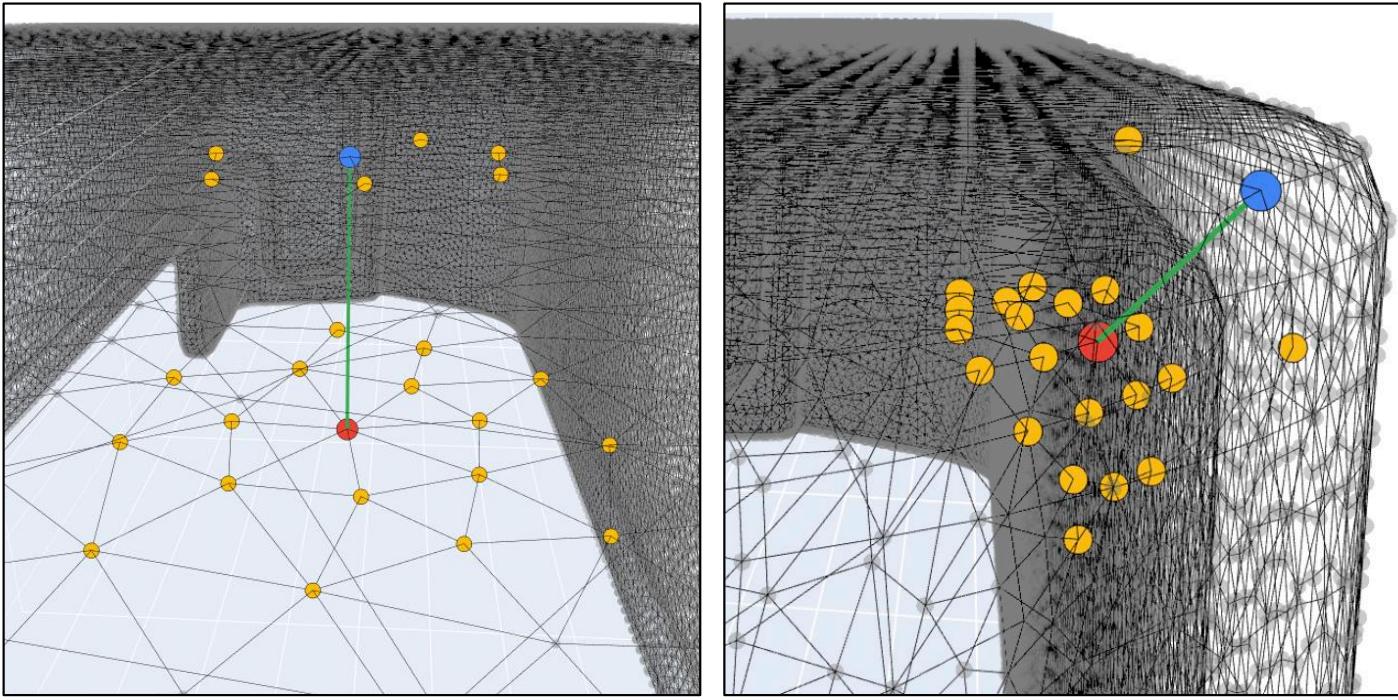


Thickness interaction

We achieve accurate and efficient 3D analysis by modeling  
**thickness interactions directly on the surface mesh.**

# PROPOSED METHOD

## ► Core Idea 1: Thickness-Awareness



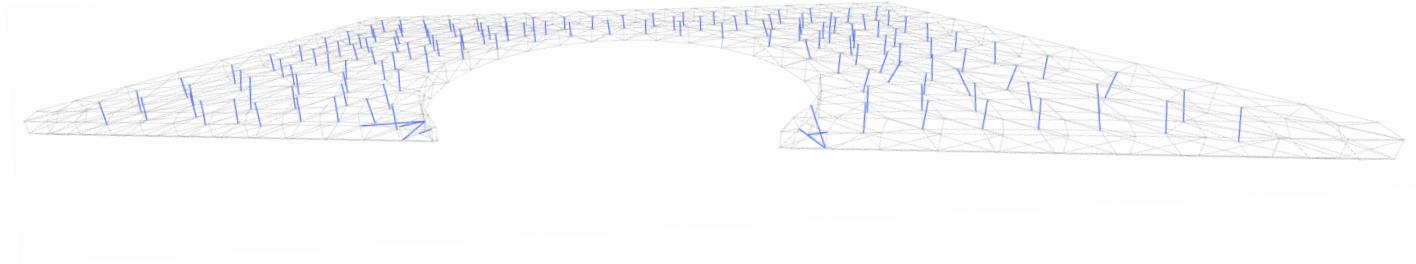
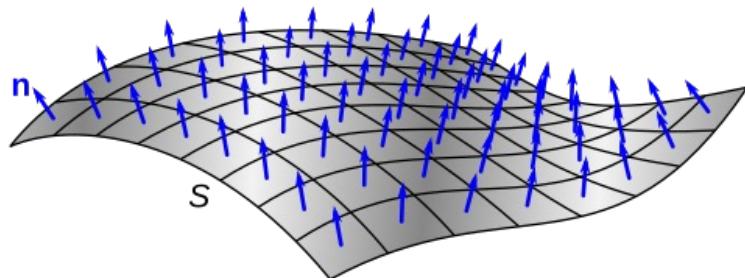
: Thickness pair nodes

: Thickness edges

We introduce a '**thickness edge**' to connect a node with its corresponding 'thickness paired node' on the opposing surface.

# PROPOSED METHOD

## ► Core Idea 1: Thickness-Awareness

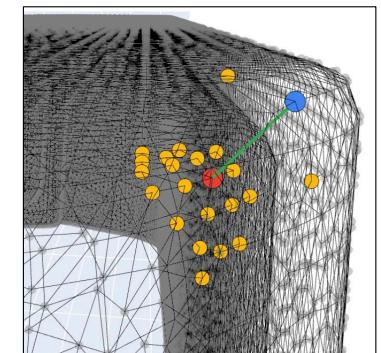


- Definition of a '**thickness paired node**'

$$\mathcal{T}(v_i) = \arg \min_{v_j \in V, v_j \neq v_i} \|\mathbf{x}_j - (\mathbf{x}_i - d \cdot \mathbf{n}_i^{\text{node}})\|,$$

$\mathcal{T}(v_i) \in V$ , where  $\mathcal{T}(v_i) \neq v_i$

$\mathbf{x}$  : coordinate,  $\mathbf{n}^{\text{node}}$  : normal vector

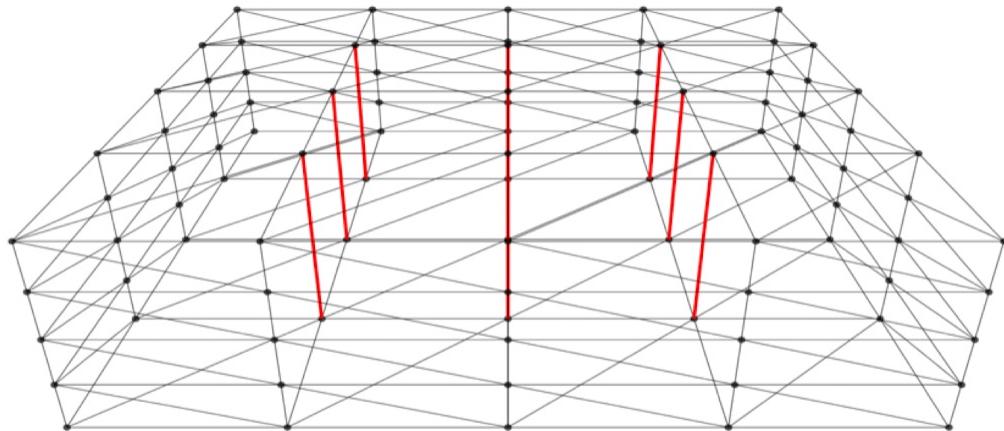


- Definition of a '**thickness**'

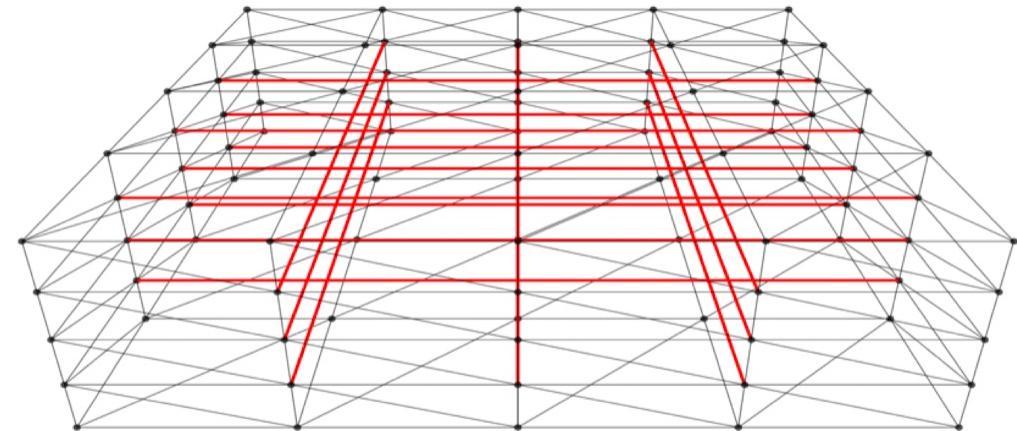
$$t(v_i) = \|\mathbf{x}_i - \mathbf{x}_{\mathcal{T}(v_i)}\|$$

# PROPOSED METHOD

## ► Core Idea 1: Thickness-Awareness



(a) Thickness



(b) Width

**Opposing nodes do not always define true 'thickness'.**

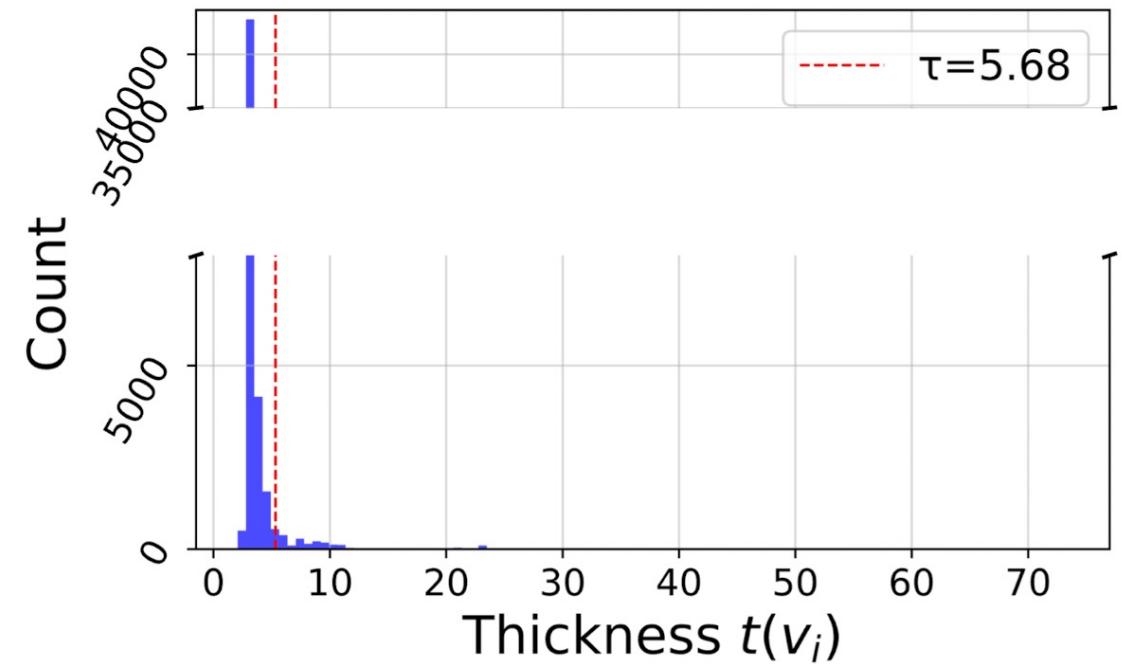
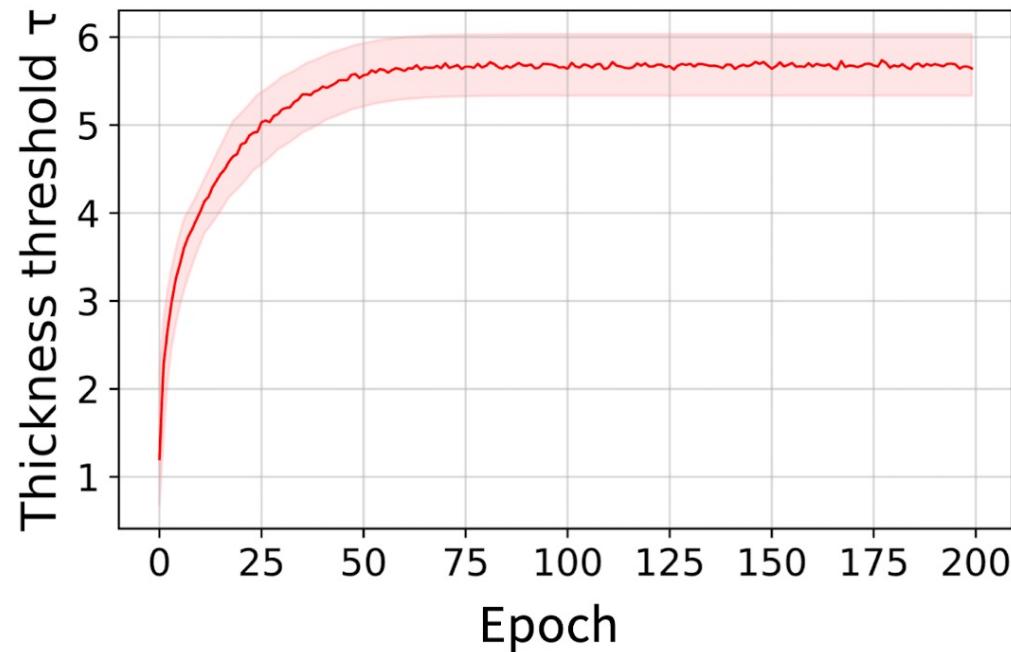
They can also represent 'width', which has a weaker dynamic relationship.

# PROPOSED METHOD

- Core Idea 1: Thickness-Awareness

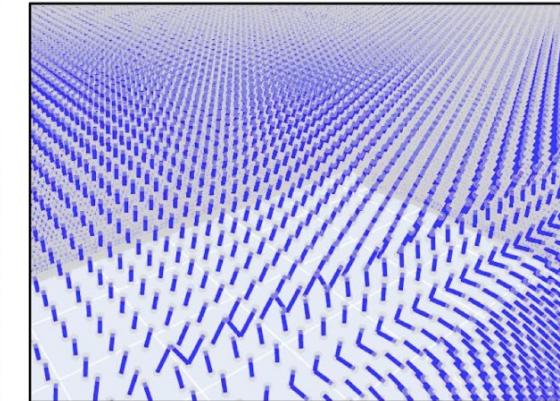
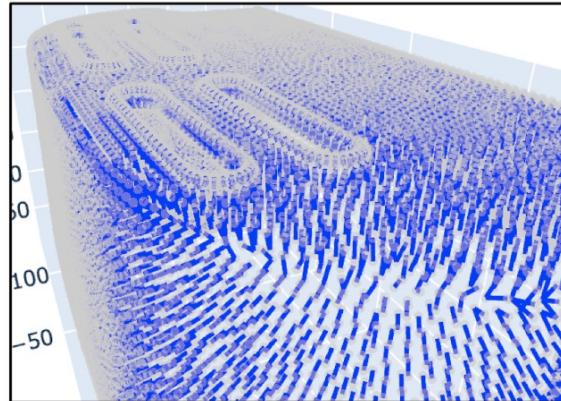
$$I_i = \frac{1}{1 + e^{\alpha(t(v_i) - \tau)}}$$

$\tau$  : learnable threshold for the thickness edge

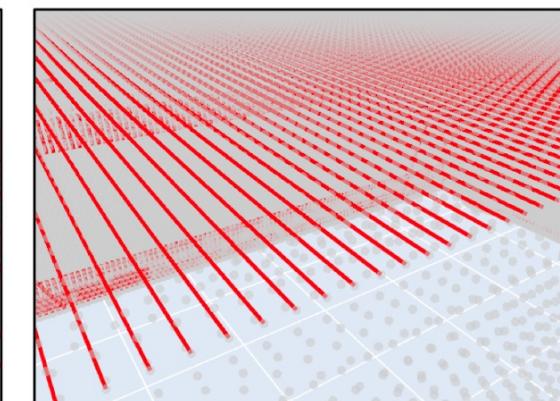
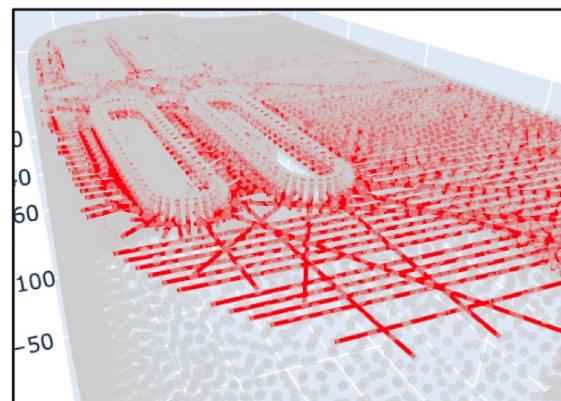


# PROPOSED METHOD

- Core Idea 1: Thickness-Awareness



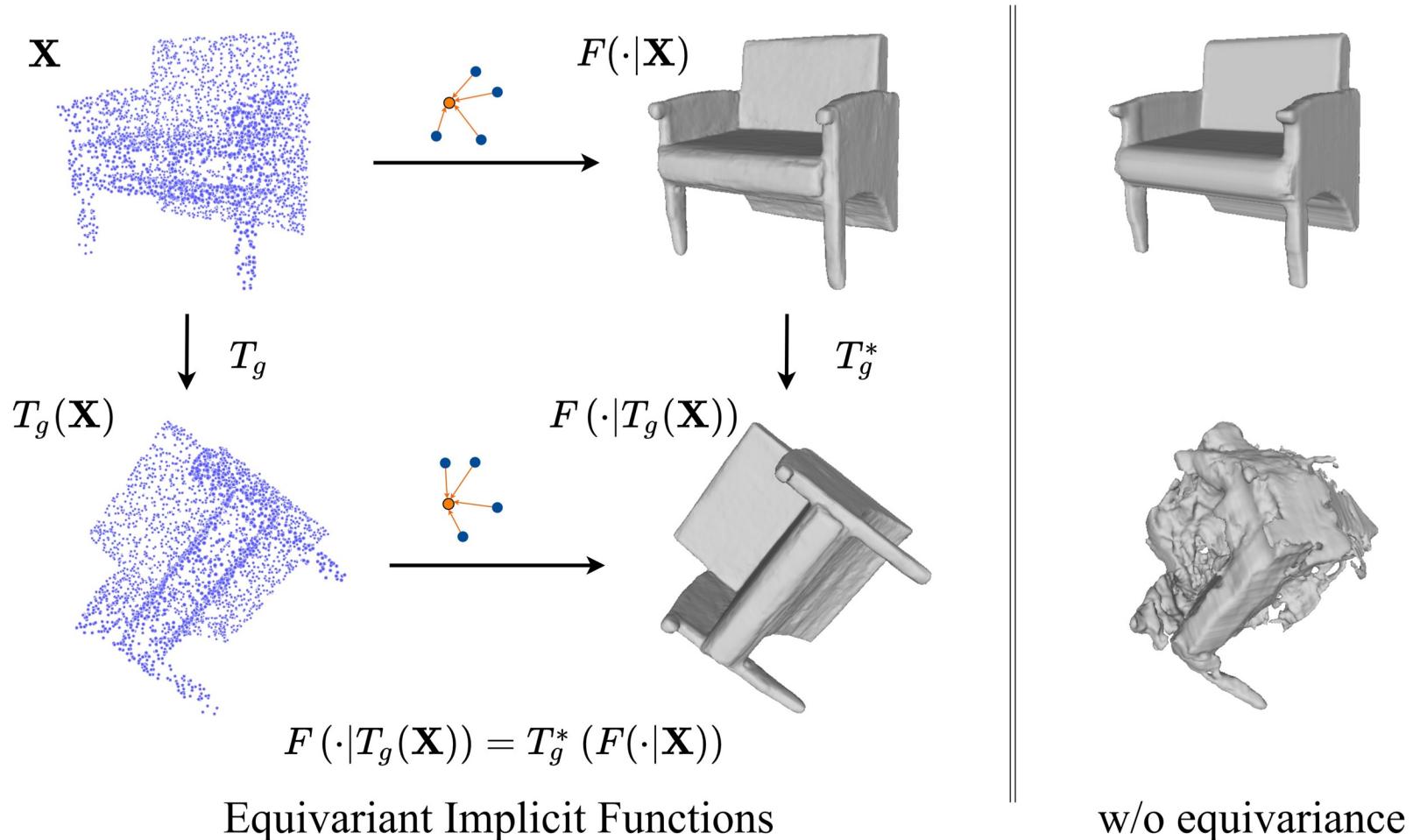
(a) Below the threshold  $\tau$



(b) Above the threshold  $\tau$

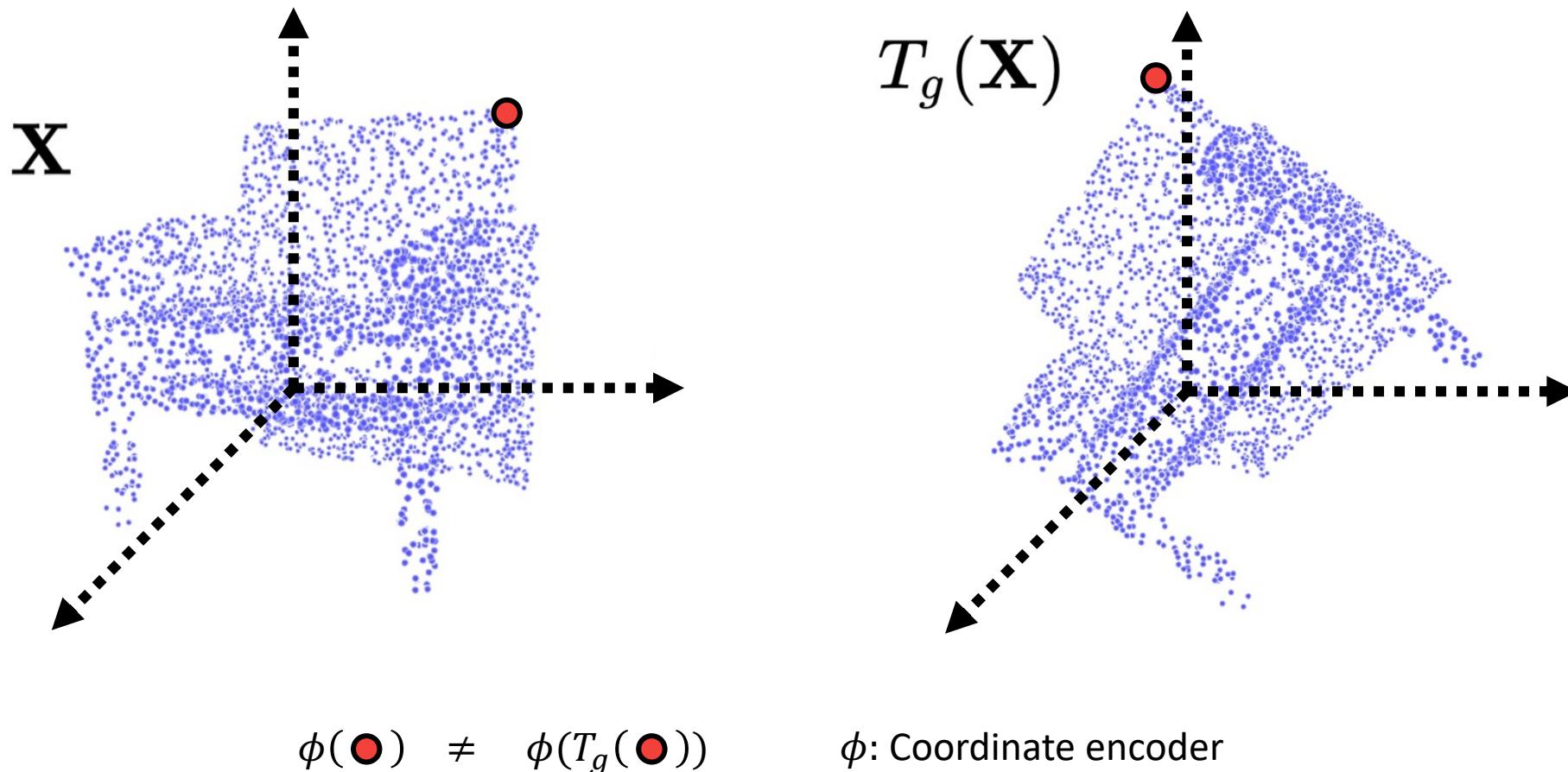
# PROPOSED METHOD

## ► Core Idea 2: E(3)-Equivariance



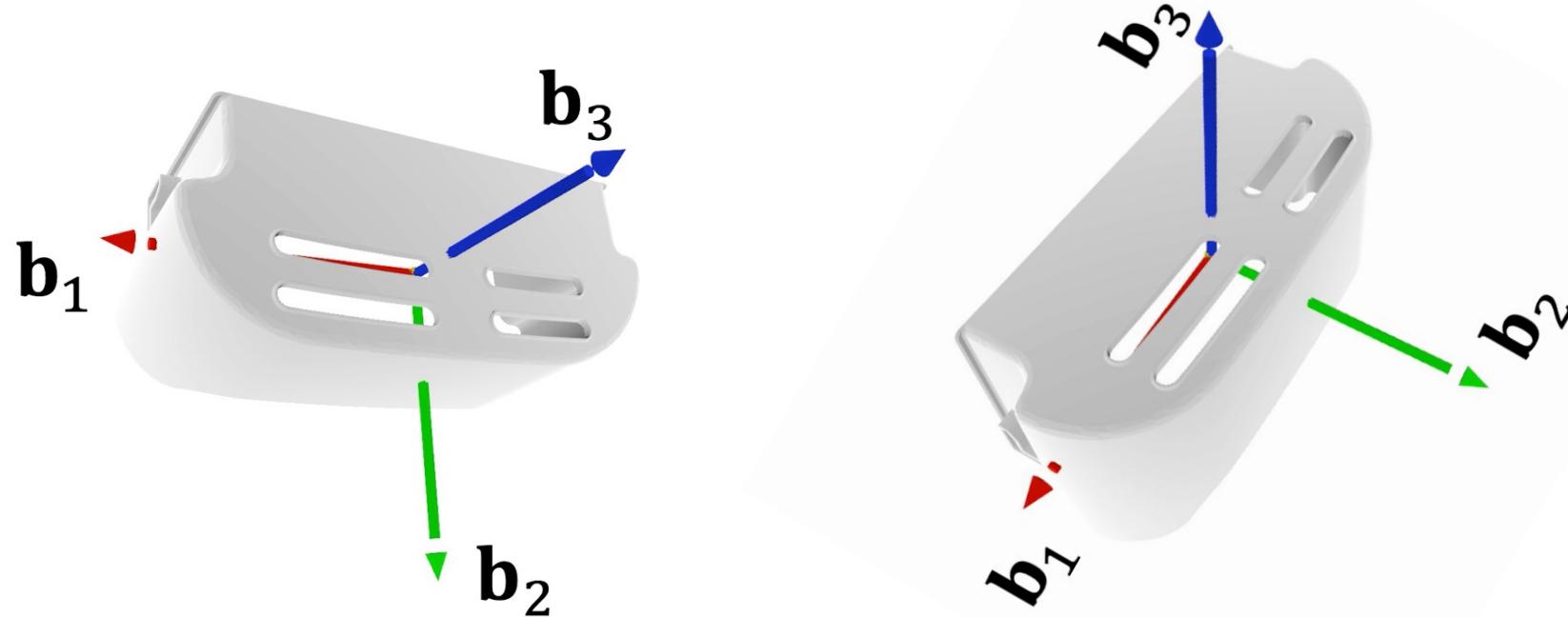
# PROPOSED METHOD

- Core Idea 2: E(3)-Equivariance



# PROPOSED METHOD

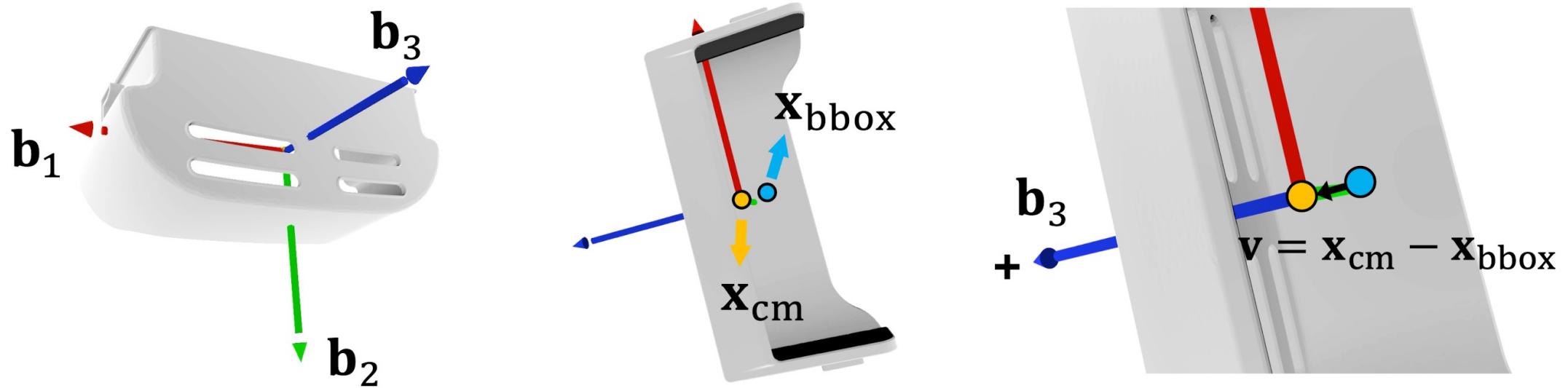
- Core Idea 2: E(3)-Equivariance



Our proposed **data-driven coordinate system is invariant to E(3) transformations**, which include rotations, translations, and reflections.

# PROPOSED METHOD

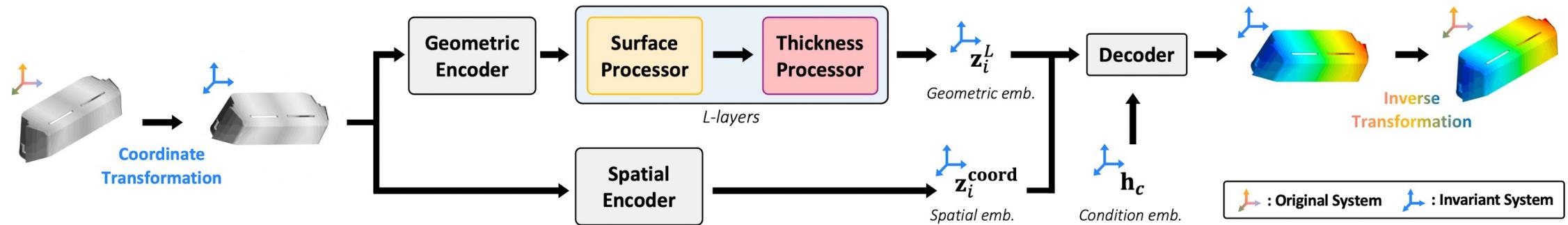
- Core Idea 2: E(3)-Equivariance



Our proposed **data-driven coordinate system is invariant to E(3) transformations**, which include rotations, translations, and reflections.

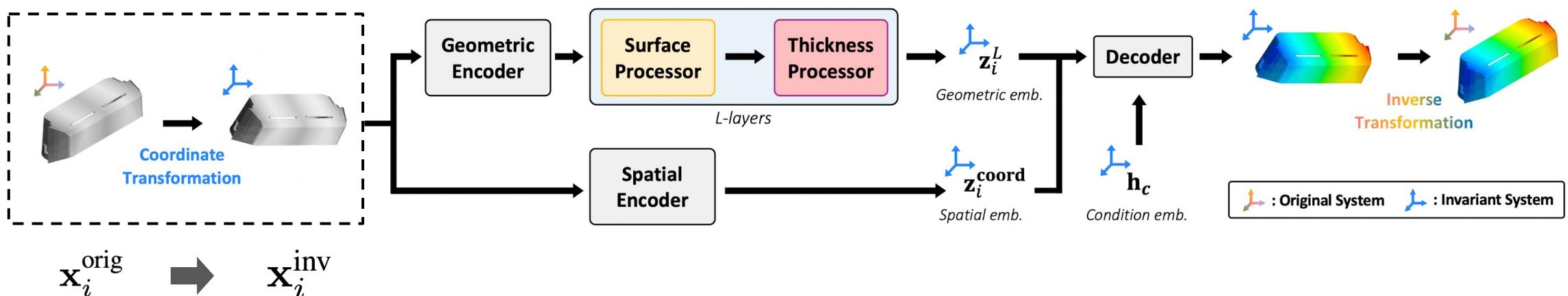
# PROPOSED METHOD

## ► Overall Architecture



# PROPOSED METHOD

## ► Overall Architecture



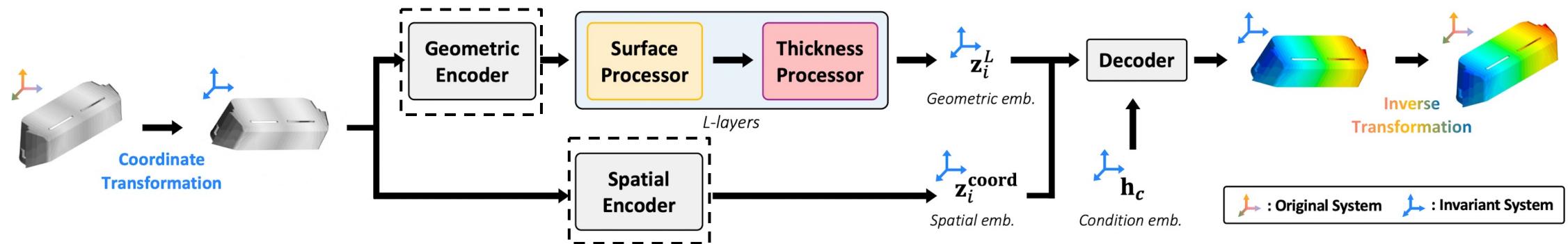
**Step 1. Translate** the original coordinates relative to the center of mass  $\mathbf{x}_{\text{cm}}$        $\tilde{\mathbf{x}}_i = \mathbf{x}_i^{\text{orig}} - \mathbf{x}_{\text{cm}}$

**Step 2.** The adjusted coordinates are **rotated** to align with the data-driven principal axes.       $\mathbf{x}_i^{\text{inv}} = \mathbf{R}^\top \tilde{\mathbf{x}}_i$

\* For each shape, the center of mass ( $\mathbf{x}_{\text{cm}}$ ) and the rotation matrix ( $\mathbf{R}$ ) are stored for the inverse transformation.

# PROPOSED METHOD

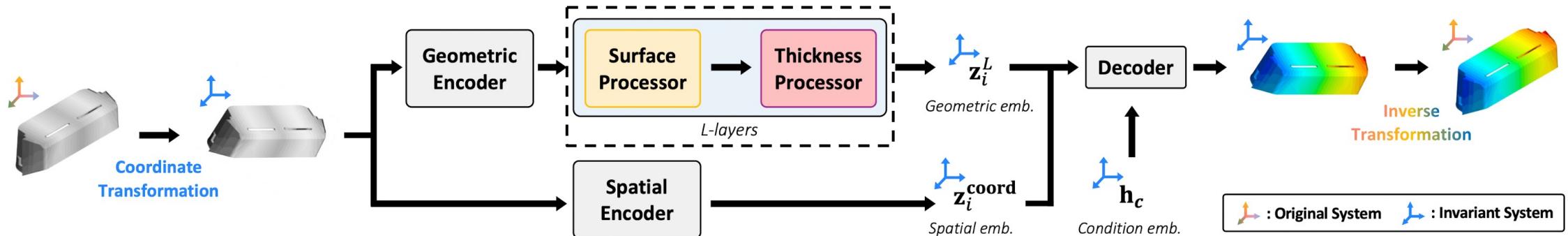
## ► Overall Architecture



- Geometric Encoder:  $\mathbf{z}_i^{(0)} = \phi_{\text{node}}(\mathbf{f}_i), \quad \mathbf{e}_{ij}^{(0)} = \phi_{\text{edge}}(\mathbf{f}_{ij})$
- Spatial Encoder:  $\mathbf{z}_i^{\text{coord}} = \phi_{\text{coord}}(\mathbf{x}_i^{\text{inv}})$

# PROPOSED METHOD

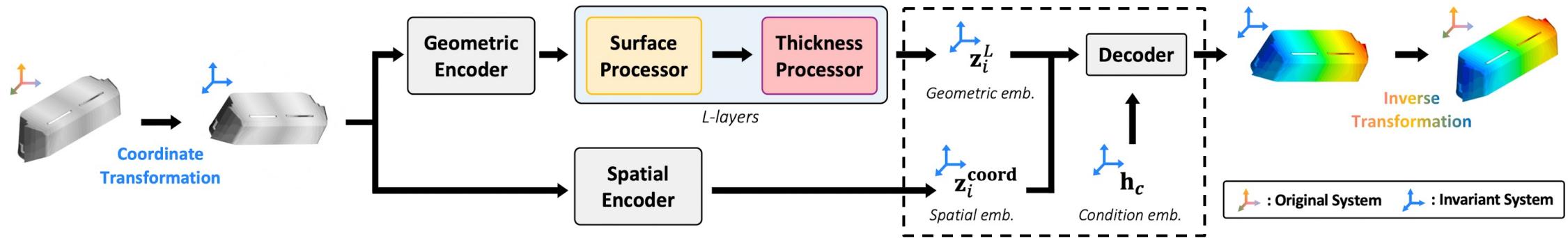
## ► Overall Architecture



- Surface Encoder:  $\mathbf{e}_{ij}^{(l+1)} \leftarrow f_{\text{surf}}^M(\mathbf{e}_{ij}^{(l)}, \mathbf{z}_i^{(l)}, \mathbf{z}_j^{(l)}),$   
 $\mathbf{z}_i^{\text{surf},(l)} \leftarrow f_{\text{surf}}^V(\mathbf{z}_i^{(l)}, \sum_{j \in \mathcal{N}(i)} \mathbf{e}_{ij}^{(l+1)})$
- Thickness Processor:  $I_i = \frac{1}{1 + e^{\alpha(t(v_i) - \tau)}}$        $\mathbf{e}_{i,\text{thick}}^{(0)} \leftarrow \phi_{\text{thick}}(\mathbf{f}_{i,\text{thick}})$        $\mathbf{f}_{i,\text{thick}} = [t(v_i), \mathbf{n}_i \cdot \mathbf{n}_{i\tau}]$   
 $\mathbf{e}_{i,\text{thick}}^{(l+1)} \leftarrow I_i \cdot f_{\text{thick}}^M(\mathbf{e}_{i,\text{thick}}^{(l)}, \mathbf{z}_i^{\text{surf},(l)}, \mathbf{z}_{\mathcal{T}(v_i)}^{\text{surf},(l)})$   
 $\mathbf{z}_i^{(l+1)} \leftarrow f_{\text{thick}}^V(\mathbf{z}_i^{\text{surf},(l)}, \mathbf{e}_{i,\text{thick}}^{(l+1)})$

# PROPOSED METHOD

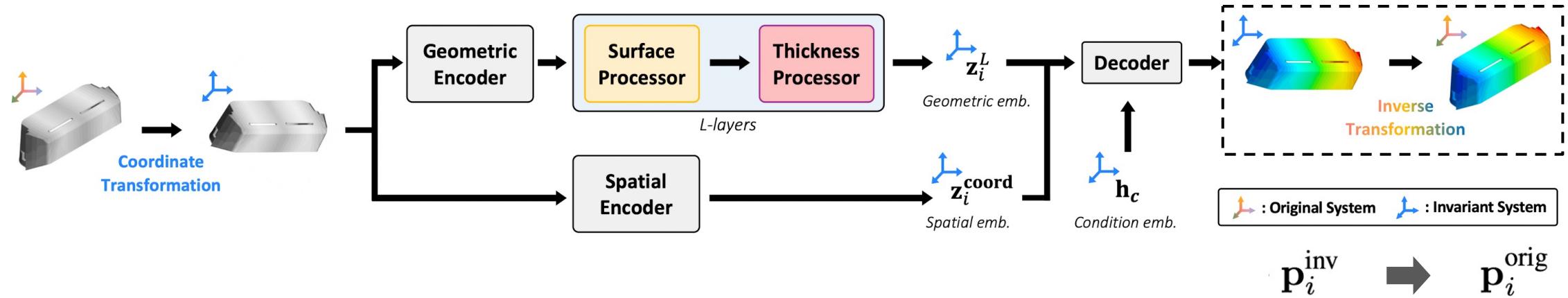
## ► Overall Architecture



- Decoder:  $\mathbf{z}_i^{\text{final}} = \phi_{\text{combine}}([\mathbf{z}_i, \mathbf{z}_i^{\text{coord}}])$   
 $\mathbf{p}_i^{\text{inv}} = \phi_{\text{decode}}([\mathbf{z}_i^{\text{final}}, \mathbf{h}_c])$

# PROPOSED METHOD

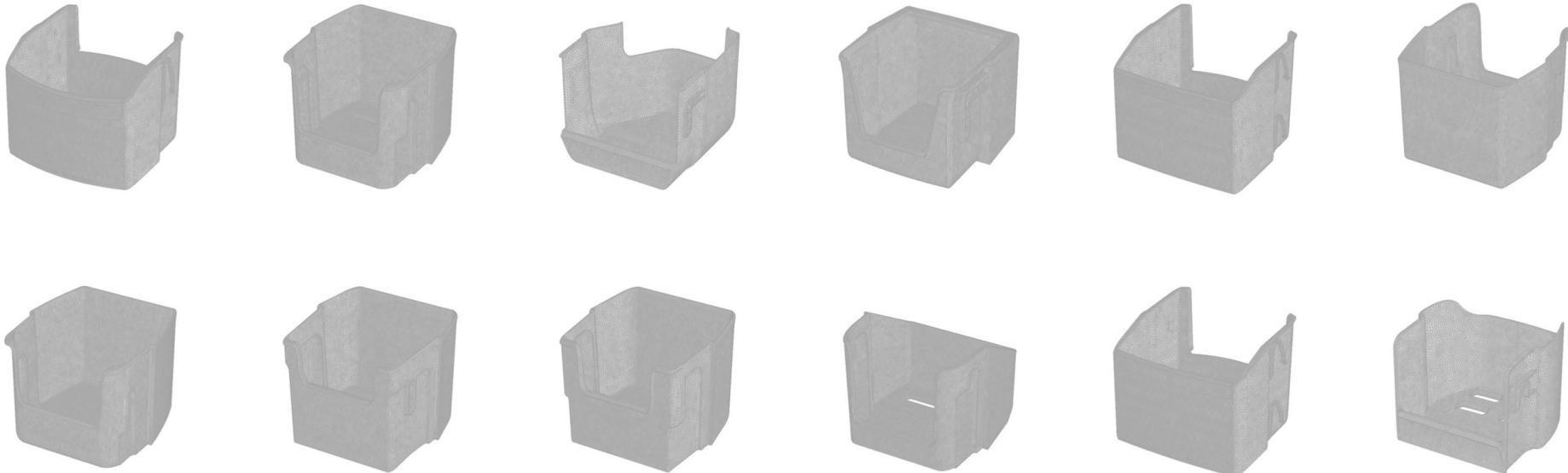
## ► Overall Architecture



- Inverse Transform:  $p_i^{\text{orig}} = \mathbf{R} \cdot p_i^{\text{inv}} + \mathbf{x}_{\text{cm}}$

# EXPERIMENTAL SETUP

- ▶ Dataset: Real-world injection molding dataset (basket)



- 28 unique geometries, each with 18 experimental conditions
  - **Test set**
    - Unseen geometries + Seen 18 experimental conditions
    - Seen geometries + Unseen experimental conditions
  - **Out-of-distribution test set:** randomly rotated geometries
- Target: Node-level deflection (3-dimensions prediction for each node)

# EXPERIMENTS

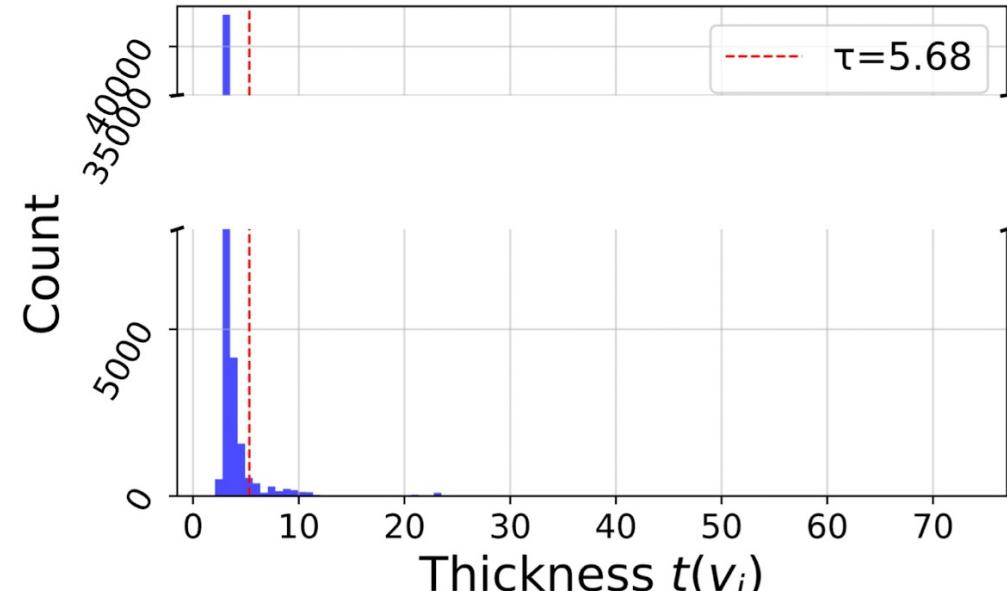
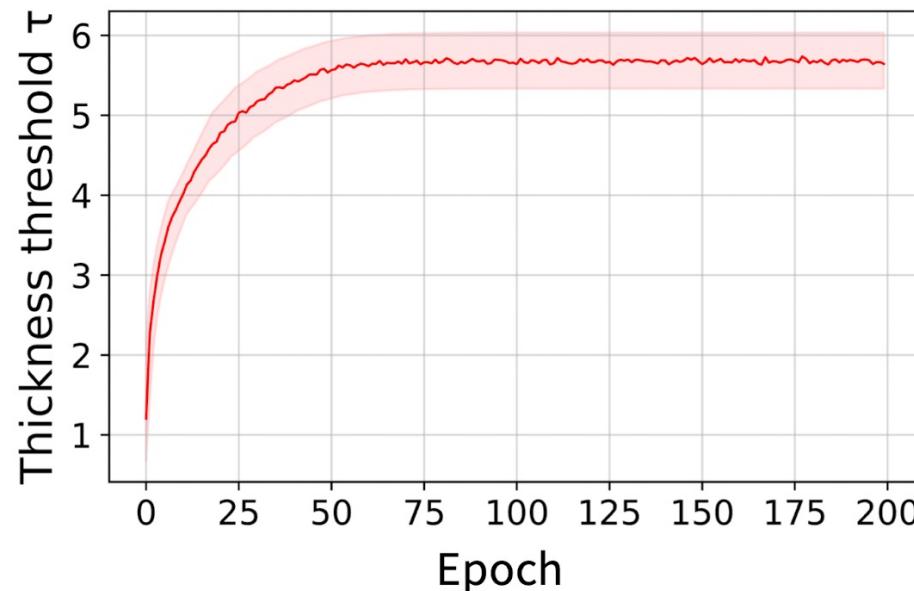
## ► Result 1: Quantitative Analysis

Model	Equivariance	Spatial information	Thickness edges	Input of $\phi_{\text{coord}}$	Edge Feature $f_{ij}$	Node Feature $f_i$	In Distribution (Original)			Out of Distribution (Rotated)		
							RMSE ( $\downarrow$ )	MAE ( $\downarrow$ )	$R^2$ ( $\uparrow$ )	RMSE ( $\downarrow$ )	MAE ( $\downarrow$ )	$R^2$ ( $\uparrow$ )
(a) MLP	✗	✓	✗	-	-	$x_{ij}^{\text{orig}}$	0.2818 (0.0061)	0.1164 (0.0035)	0.8984 (0.0029)	0.4789 (0.0181)	0.1939 (0.0070)	0.7393 (0.0248)
(b) MLP	✓	✓	✗	-	-	$x_{ij}^{\text{inv}}$	0.2546 (0.0015)	0.1043 (0.0008)	0.9154 (0.0016)	0.2545 (0.0015)	0.1071 (0.0007)	0.9385 (0.0009)
(c) MGN	✗	✗	✗	-	$x_{ij}, \ x_{ij}\ $	$n_i, g_i, r_i$	1.2608 (0.0107)	0.5607 (0.0041)	0.0782 (0.0315)	1.3188 (0.0164)	0.6199 (0.0064)	-0.0903 (0.0315)
(d) MGN	✗	✓	✗	$x_{ij}^{\text{orig}}$	$x_{ij}, \ x_{ij}\ $	$n_i, g_i, r_i$	0.2854 (0.0046)	0.1176 (0.0017)	0.8724 (0.0037)	0.4514 (0.0190)	0.1938 (0.0067)	0.7917 (0.0180)
(e) MGN	✓	✓	✗	$x_{ij}^{\text{inv}}$	$x_{ij}, \ x_{ij}\ $	$n_i, g_i, r_i$	0.2241 (0.0042)	0.0938 (0.0029)	0.9113 (0.0099)	0.2241 (0.0042)	0.0965 (0.0024)	0.9446 (0.0033)
(f) EGNN	✓	✗	✗	-	$\ x_{ij}\ $	$g_i, r_i$	153.051 (4.2992)	54.363 (2.1000)	-14341.0 (1214.1)	196.343 (1.6422)	89.049 (1.2804)	-32260.9 (1039.3)
(g) EGNN	✗	✓	✗	$x_{ij}^{\text{orig}}$	$\ x_{ij}\ $	$g_i, r_i$	0.2944 (0.0045)	0.1220 (0.0021)	0.8680 (0.0056)	0.4576 (0.0184)	0.1958 (0.0064)	0.8074 (0.0206)
(h) EGNN	✓	✓	✗	$x_{ij}^{\text{inv}}$	$\ x_{ij}\ $	$g_i, r_i$	0.2270 (0.0019)	0.0963 (0.0008)	0.9129 (0.0026)	0.2271 (0.0019)	0.0987 (0.0009)	0.9443 (0.0012)
(i) EMNN	✓	✗	✗	-	$\ x_{ij}\ $	$g_i, r_i$	166.077 (1.5226)	58.467 (2.0000)	-16034.0 (975.8)	201.450 (1.7433)	92.237 (1.3366)	-34302.7 (644.62)
(j) EMNN	✗	✓	✗	$x_{ij}^{\text{orig}}$	$\ x_{ij}\ $	$g_i, r_i$	0.3056 (0.0246)	0.1284 (0.0131)	0.8626 (0.0052)	0.4668 (0.0180)	0.2024 (0.0092)	0.7972 (0.0097)
(k) EMNN	✓	✓	✗	$x_{ij}^{\text{inv}}$	$\ x_{ij}\ $	$g_i, r_i$	0.2210 (0.0057)	0.0937 (0.0034)	0.9149 (0.0034)	0.2210 (0.0057)	0.0963 (0.0052)	0.9473 (0.0012)
(l) T-EMNN	✓	✓	✓	$x_{ij}^{\text{inv}}$	$\ x_{ij}\ $	$g_i, r_i$	<b>0.2132</b> (0.0046)	<b>0.0892</b> (0.0025)	<b>0.9228</b> (0.0063)	<b>0.2131</b> (0.0046)	<b>0.0918</b> (0.0023)	<b>0.9513</b> (0.0031)

T-EMNN outperformed the other baselines thanks to two key components: **our thickness address** and **our data-driven coordinate system**, which conferred equivariance on both our method and the baselines in OOD settings.

# EXPERIMENTS

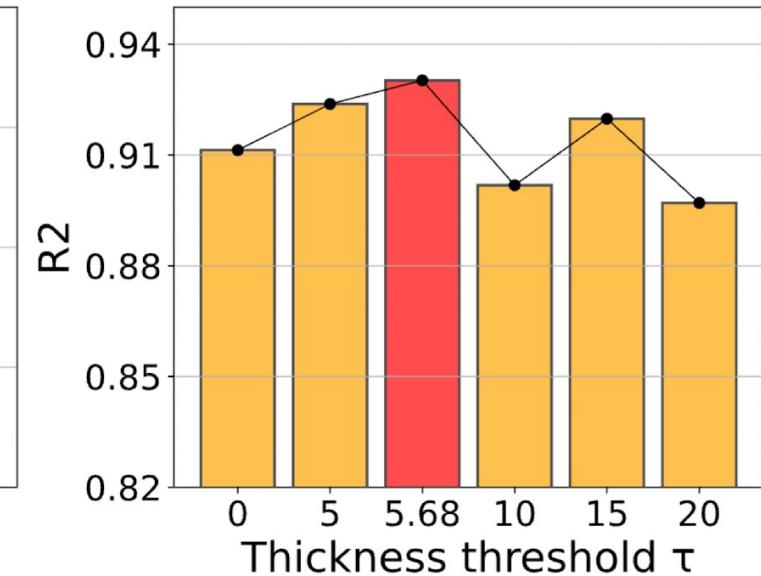
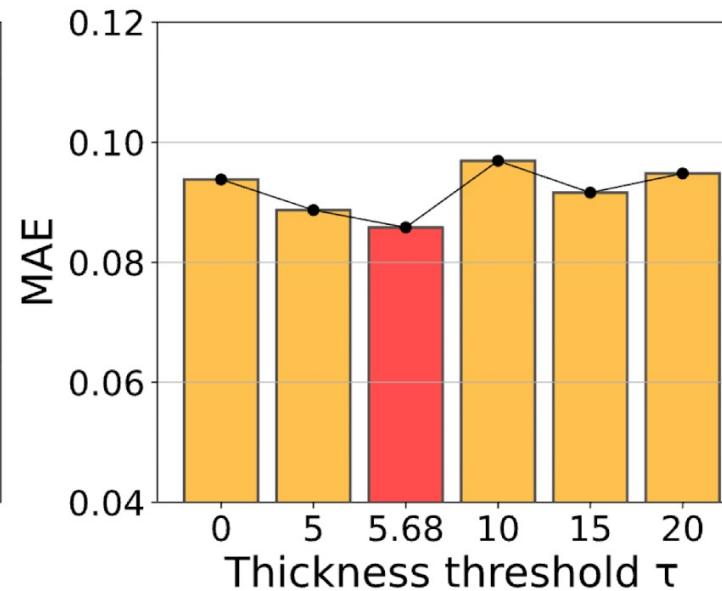
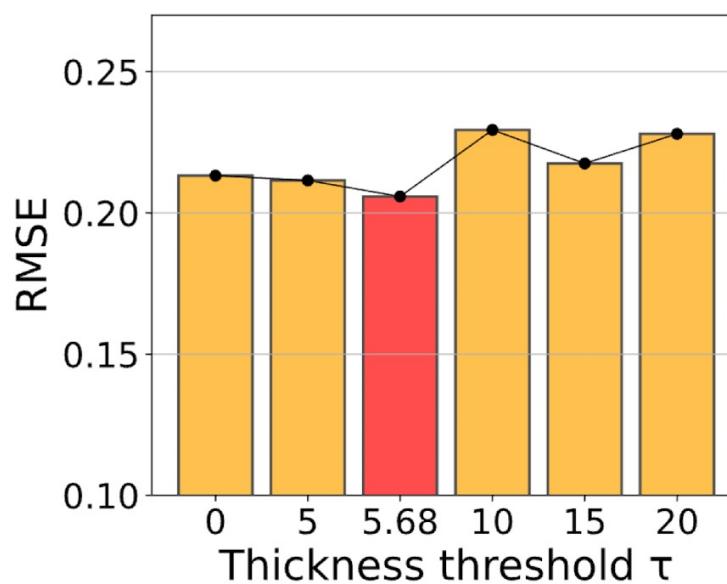
## ► Result 2: Thickness Framework Validation



The learnable thickness threshold **converges to a specific value**, which enables interaction between highly correlated nodes on opposing surfaces.

# EXPERIMENTS

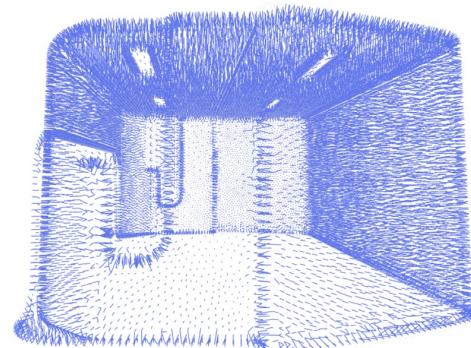
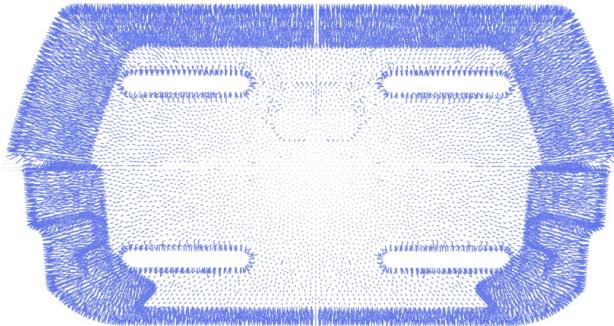
## ► Result 2: Thickness Framework Validation



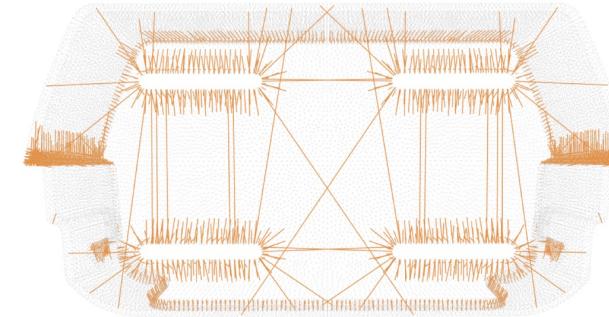
The learned thickness threshold (i.e., 5.68) outperforms the use of manually set, fixed threshold values (e.g., 0, 5, 10, 15, 20).

# EXPERIMENTS

## ► Result 2: Thickness Framework Validation



Below the threshold  $\tau$

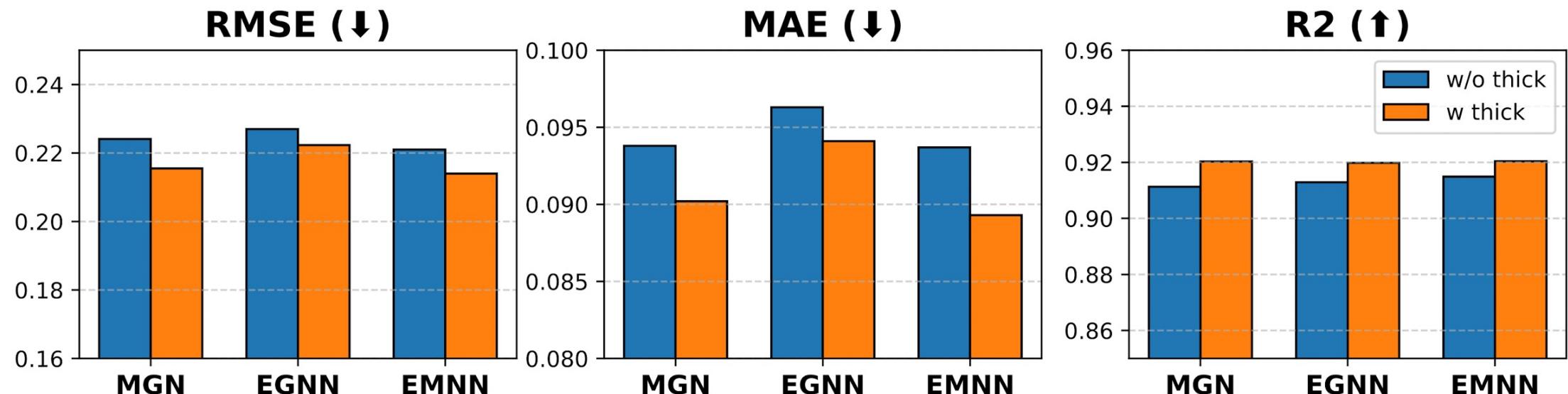


Above the threshold  $\tau$

The learned thickness threshold (i.e., 5.68) effectively **filters out noisy thickness edges** that can be seen as width rather than thickness.

# EXPERIMENTS

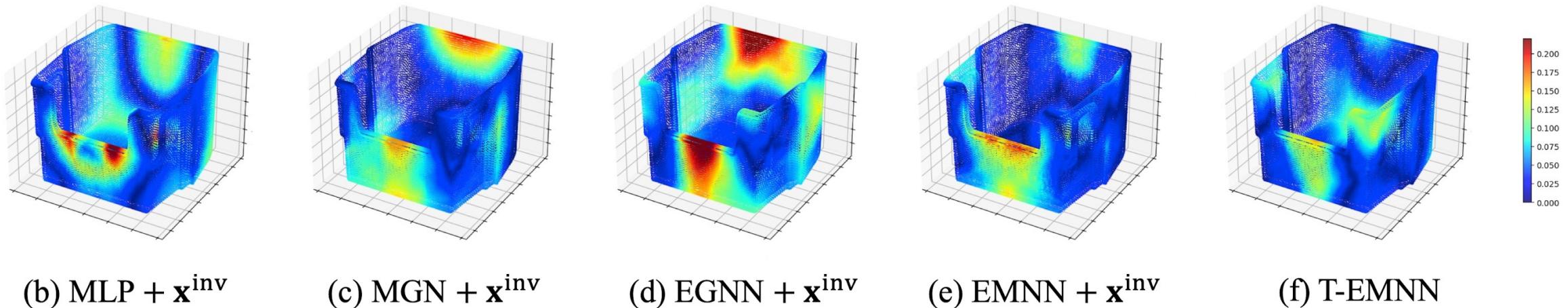
## ► Result 2: Thickness Framework Validation



Applying the filtered thickness edges to the baselines improves their performance.

# EXPERIMENTS

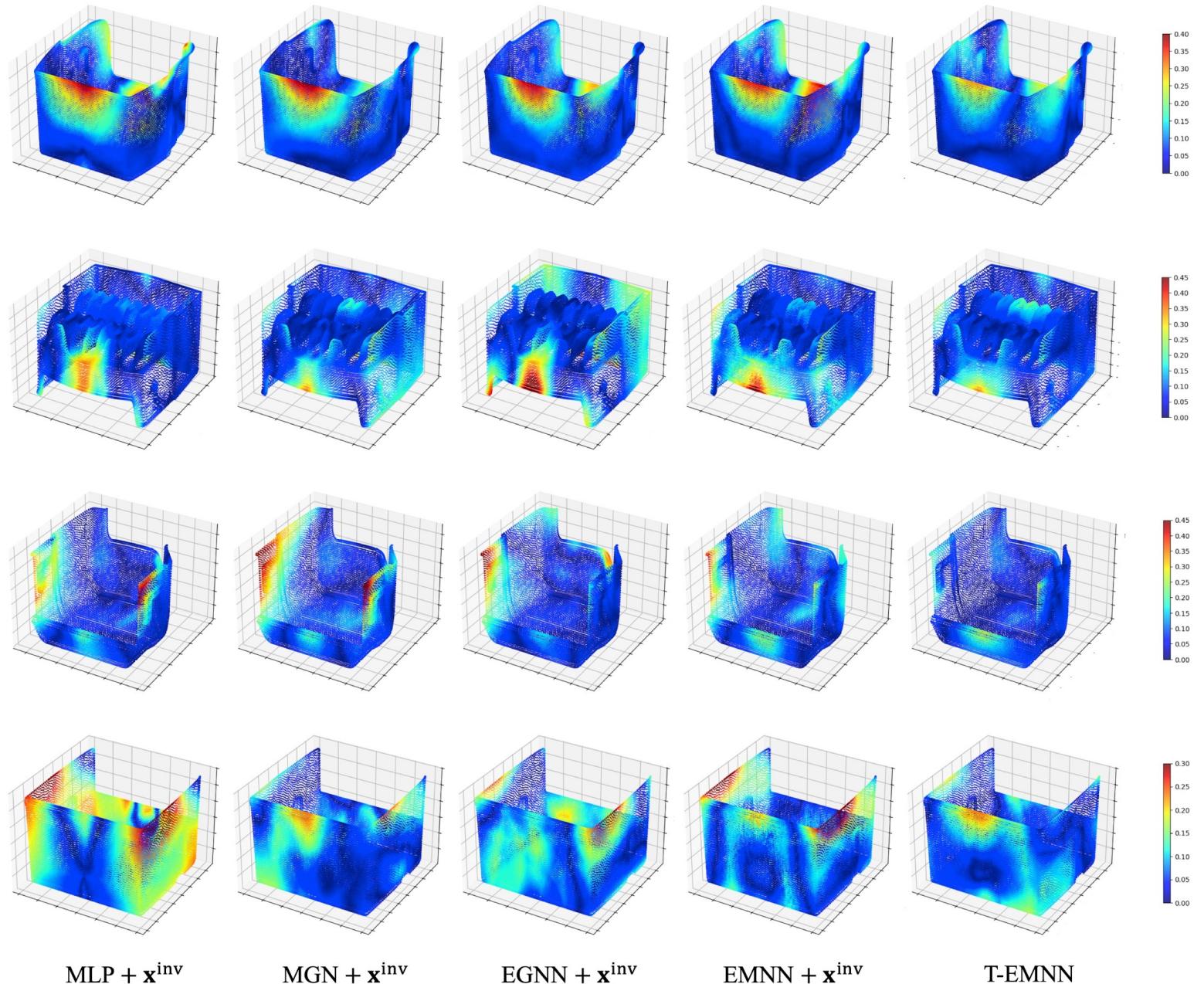
## ► Result 3: Qualitative Analysis



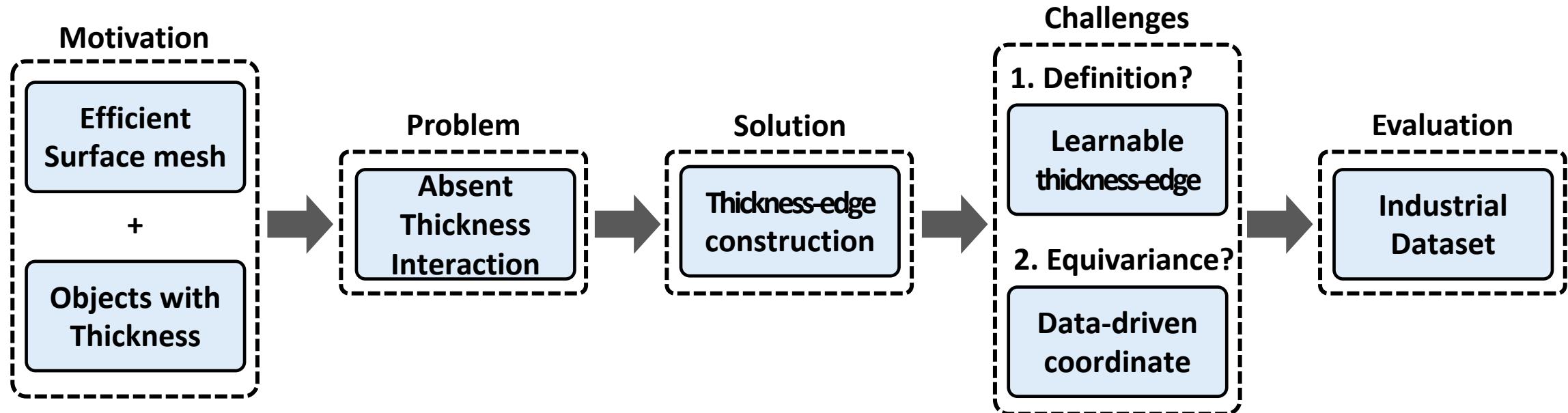
Using thickness edges **reduces the overall error** by allowing messages to pass more effectively between opposing surfaces.

# EXPERIMENTS

## ► Result 3: Qualitative Analysis



# CONCLUSIONS



Paper

Thickness-aware  $E(3)$ -Equivariant 3D Mesh Neural Network

Paper: <https://arxiv.org/abs/2505.21572>

Email: [swkim@kaist.ac.kr](mailto:swkim@kaist.ac.kr)

