CEIG - Spanish Computer Graphics Conference (2026)
E. Garces and I. Muñoz-Pandiella (Editors)

Predicting Deformation Fields on 3D Stroke Clouds

S. Rasoulzadeh[1], R. Suliman[1], A. Rasoulzadeh[2], I. Kovacic[1], M. Wimmer[1]

[1] TU Wien, Center for Geometry and Computational Design, Austria
[2] METROM GmbH, Germany

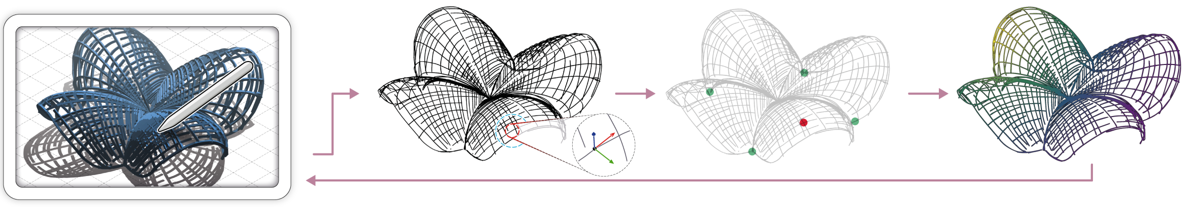

Figure 1: *Given an unoriented set of 3D strokes depicting a user-intended surface (a), our model encodes the stroke cloud topology together with per-point orthonormal frame components and user-specified boundary-condition annotations (load and support points) (b, c), and predicts the induced deformation field directly onto the strokes, providing immediate feedback highlighting structurally weak regions (d).*

Abstract

Deformation analysis is typically performed only after geometric modeling in design workflows, depriving designers of this physical insight during the early sketching phase, when the design has the highest optimization potential. To bridge this gap, we propose a physics-informed neural model that takes as input a 3D sketch stroke cloud and user-specified boundary conditions, and directly predicts the induced 3D deformation field on the stroke cloud without requiring reconstruction-and-simulation. To represent unstructured 3D sketches and user-annotated boundary conditions in a form suitable for learning, we use fVDB to encode stroke clouds into sparse voxel grids and leverage its differentiable splatting and sampling operators for bidirectional point–voxel mappings. Trained on our synthetic dataset of architectural thin-shell sketch–deformation pairs, our dual-head model predicts per-point displacement magnitudes and unit-length vectors, and is optimized using physics-informed loss terms inspired by stretching and bending as the two modes of deformation.

CCS Concepts

• ***Computing methodologies*** → ***Computer graphics; Machine learning; Modeling and simulation;***

1. Introduction

Beyond traditional 2D sketching, the growing availability of 3D sketching interfaces, such as 3D pen-on-tablet devices and VR/AR interfaces, is prompting designers to explore these mediums across architectural and industrial design workflows. In such workflows, deformation analysis of design objects is an important step, yet it remains inaccessible during the early sketching phase, limiting designers' ability to reason about a shape's physical behavior and structural soundness. As a result, designers are forced to follow a cumbersome loop of sketching, modeling, running physical simulations, and revising the sketch based on the feedback. However, this loop counteracts the premise that design workflows should be as short as possible and should not disrupt seamless creative flow.

Inspired by this metaphor, we present a physics-informed neural network that enables deformation-aware 3D sketching by providing structural feedback directly on 3D stroke clouds. Key to our method is: (i) a synthetic data generation pipeline producing 40K pairs of architectural thin-shell sketches and their corresponding deformation fields, (ii) encoding of unstructured 3D stroke-cloud topologies and per-point features into structured sparse fVDB voxel grids suitable for processing within neural networks, and (iii) devising physics-informed loss functions derived from thin-shell deformation theory, where stretching and bending energies, as the two modes of deformation, are discretized over local stroke neighborhoods to regularize the predicted deformation field. Our source code is available at: https://gitlab.cg.tuwien.ac.at/srasoulzadeh/strokes2deform.git.

DOI: 10.2312/ceig.20261012

2. State of the Art

Closest in motivation to our work is Sketch2Stress [YXLF23], which introduces a data-driven generative model that enables users to perform stress prediction over 2D sketched objects by transforming the problem into an image-to-image translation task. Related works targeting 3D sketches often rely on an intermediate surface reconstruction step followed by Finite Element (FE) analysis for physical simulation. These approaches are prone to latency and reconstruction artifacts or errors that propagate into the simulation, thereby causing inaccuracies in the output results [RSK*23]. Another related work addresses curve simplification w.r.t to structural stability and material cost [NPTB22], however, it assumes regular networks with *a priori* known connectivities and does not extend to unstructured sketches. Our work is also inspired by the growing body of Physics-Informed Neural Networks (PINNs) that integrate prior physical knowledge into training in order to achieve better physical consistency, higher accuracy, increased data efficiency, improved generalization, and greater interpretability [WY25].

3. Method

We address deformation analysis for unstructured 3D sketches of architectural thin-shell structures under user-specified boundary conditions (load and support points) at arbitrary locations. As 3D sketches lack sizable ground-truth datasets with corresponding surfaces or simulations, we collect a set of unique 3D sketches and devise a synthetic data generation pipeline coupling 3D sketch stroke clouds with induced deformation fields (see Section 3.1). With the resulting dataset, we devise a physics-informed neural network trained end-to-end on 3D stroke clouds and capable of predicting deformation fields directly on stroke polyline vertices without requiring reconstruction-and-simulation (see Section 3.2).

Notation. Throughout the remainder of this manuscript, we denote a stroke cloud as a set of stroke polylines $\mathcal{S} = \{\mathcal{S}_i\}_{i=1}^{M}$, where each stroke polyline $\mathcal{S}_i = \{p_{i,j}\}_{j=1}^{N_i} \subset \mathbb{R}^3$ is an ordered sequence of vertices in 3D, and M and N_i denote the total number of strokes and the number of points in the i-th stroke, respectively. The union of all polyline points forms the stroke point cloud $\mathcal{P} = \bigcup_{i=1}^{M} \mathcal{S}_i = \{p_{i,j}\}_{j=1}^{N_i} \subset \mathbb{R}^3$.

3.1. Synthetic Data Generation

We curate 3D sketches of architectural thin-shell structures and devise a three-stage synthetic data generation pipeline yielding 40K sketch-deformation pairs. In the first stage, we apply Poisson reconstruction to stroke polyline vertices as-is, manually refine the mesh into the user-intended surface model, and remesh it into a uniform triangulation preserving salient features for deformation analysis. In the second stage, each surface model is treated as an FE shell element of concrete with fixed thickness under linear-elastic small-deformation assumptions, and is simulated under a preset number of boundary-condition configurations. Each configuration applies a 10 kN load along the inward surface normal at one randomly selected load vertex, and selects support points by fitting a Gaussian Mixture Model (GMM) with 3–8 components to the sketch point cloud and sampling one vertex per cluster. Each support point is assigned six Degrees of Freedom (6 DoF), with translations fixed and rotations free. Finally, in the third stage, we align each stroke cloud with its surface model, and construct per-vertex input feature vectors comprising boundary conditions and local orthonormal frames as geometry-aware features and map resultant deformation fields from surface mesh to stroke cloud vertices as output vectors:

- **Input Boundary Conditions** $\mathbf{x}_p^{\text{BC}} \in \{0,1\}^2$: a one-hot binary vector, computed by transferring load/support labels from mesh vertices to stroke points via 1-Nearest-Neighbor matching.
- **Input Orthonormal Frame** $\mathbf{x}_p^{\text{Basis}} = (\mathbf{t}_p, \mathbf{n}_p, \mathbf{b}_p) \in \mathbb{R}^9$: a local frame at p with unit tangent $\mathbf{t}_p$ computed from consecutive stroke vertices, surface normal $\mathbf{n}_p$ obtained via PCA-based normal estimation on the stroke point cloud, and binormal $\mathbf{b}_p = \mathbf{t}_p \times \mathbf{n}_p$.
- **Output Deformation Field** $\mathbf{d}_p \in \mathbb{R}^3$: the ground-truth deformation vector at p, computed by Inverse-Distance Weighting (IDW) interpolation over the deformation vectors of its $k = 4$ nearest surface mesh vertices.

Given the above attributes, we associate each stroke vertex with an 11-dimensional input feature vector $\mathbf{x}_p = (\mathbf{x}_p^{\text{BC}}, \mathbf{x}_p^{\text{Basis}})$, formed by concatenating its boundary-condition features and flattened orthonormal-frame components. We also associate each vertex with a target 3D deformation vector $\mathbf{d}_p$ as the output vectors. Overall, this yields pairs $(\mathcal{X}, \mathcal{Y})$, where $\mathcal{X} = (\mathcal{S}, \{\mathbf{x}_p\}_{p\in\mathcal{P}})$ containing stroke cloud topology with its input per-vertex features, and the output $\mathcal{Y} = \{\mathbf{d}_p\}_{p\in\mathcal{P}}$ containing per-vertex deformation vectors.

3.2. Physics-Informed Neural Network

To process unstructured 3D sketches in an end-to-end learning pipeline, we use fVDB [WHS*24] to encode 3D stroke clouds (with their associated per-vertex input/output vectors) into a sparse voxel grid as a structured representation suitable for learning. The key advantage of fVDB in our setting is that it provides bidirectional point-voxel information flow via differentiable splatting and sampling operations. On the one hand, splatting per-vertex input feature vectors from stroke points onto voxels enables the network to operate on the sparse voxel grid while effectively using these features within its layers during forward propagation. On the other hand, sampling projects voxel-wise network predictions back onto the original stroke points, allowing the training loss to be computed between the output predictions and ground-truth deformation vectors originally defined on stroke polyline vertices.

3.2.1. Network Architecture

Our network is a 3D Sparse Res-UNet with four downsampling and four upsampling stages, each composed of stacked residual blocks connected via skip connections, where dense 3D convolutions are replaced with sparse counterparts to operate efficiently on sparse voxel grids (see Figure 2). Given the decomposition of the ground-truth deformation vectors $\mathbf{d}_p$ into unit-length displacement vectors $\mathbf{e}_p = \mathbf{d}_p / \|\mathbf{d}_p\|$ and scalar displacement magnitudes $\|\mathbf{d}_p\|$, we use a dual-head output layer in which one head predicts $\mathbf{e}_p$ and the other predicts $\log_2(\|\mathbf{d}_p\| + \epsilon)$. During inference, the predicted unit-length displacement vectors are combined with the inverse-transformed displacement magnitudes, obtained using base-2 exponentiation, to construct the full deformation field.

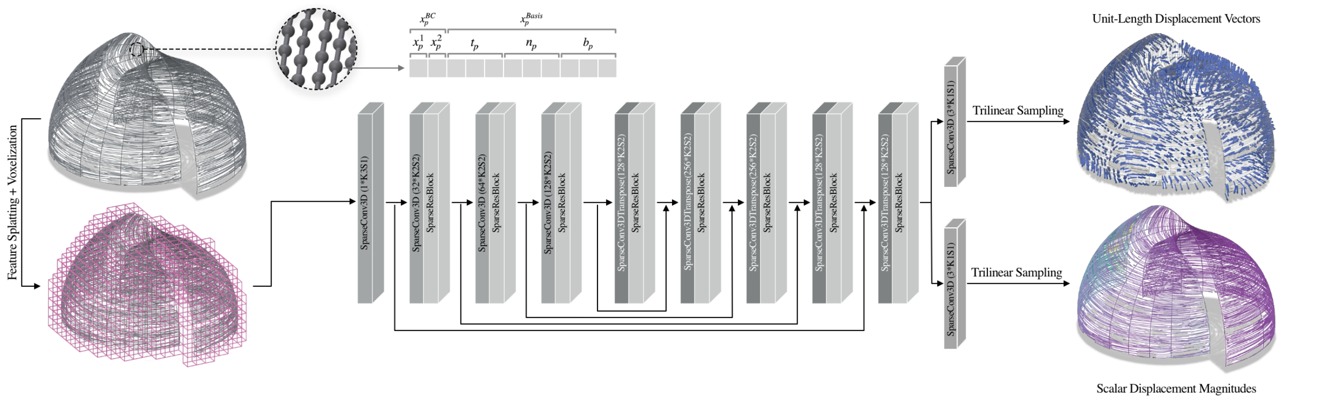

Figure 2: ***Network architecture overview***. *Input sketch stroke clouds with their 11-dimensional per-point features are encoded on a sparse voxel grid and processed by a four-level encoder–decoder with residual blocks and skip connections. All operations are performed using sparse 3D convolutions. A shared backbone branches into two output heads that separately predict unit-length displacement vectors and scalar displacement magnitudes, which are recombined during inference to recover the full deformation field on the input sketch.*

3.2.2. Losses

For training, we define an angular loss, decomposed into tangential and normal components, and a magnitude loss for the two heads predicting the unit-length displacement vectors and the scalar displacement magnitudes, respectively. We further exploit the physical decomposition of the displacement into its underlying two modes of deformation, namely in-plane (stretching) and out-of-plane (bending), components under the small-deformation assumption, and introduce additional physics-guided regularization terms.

Angular Loss. In a shell element, stretching arises from in-plane node translations, while bending arises from out-of-plane translations and in-plane rotations. Inspired by this, in order to measure the angular difference between the predicted and ground-truth displacement vectors, we leverage the local frame components (t_p, n_p, b_p) and devise an angular loss consisting of a sign-invariant tangential alignment loss $\mathcal{L}_{\text{ang},t}$ and a sign-aware normal alignment loss $\mathcal{L}_{\text{ang},n}$ corresponding to the two modes of deformation:

$$\mathcal{L}_{\text{ang},t} = \sum_{p \in \mathcal{P}} m_p \left(1 - |\hat{\mathbf{e}}_p^{||} \cdot \mathbf{e}_p^{||}|\right), \tag{1}$$

$$\mathcal{L}_{\text{ang},n} = \sum_{p \in \mathcal{P}} m_p \, |\hat{\mathbf{e}}_p \cdot \mathbf{n}_p - \mathbf{e}_p \cdot \mathbf{n}_p|, \tag{2}$$

where $\hat{\mathbf{e}}_p^{||} = (\hat{\mathbf{e}}_p - (\hat{\mathbf{e}}_p \cdot \mathbf{n}_p)\, \mathbf{n}_p) / \, \|\hat{\mathbf{e}}_p - (\hat{\mathbf{e}}_p \cdot \mathbf{n}_p)\, \mathbf{n}_p\|$ denotes the normalized projection of the predicted displacement vector onto the local tangent plane spanned by $(\mathbf{t}_p, \mathbf{b}_p)$, with $\mathbf{e}_p^{||}$ defined analogously. Moreover, m_p masks vertices below the 10% displacement-magnitude quantile, suppressing points with negligible deformation where the field is ill-defined.

Magnitude Loss. We supervise the magnitude head with Mean Squared Error (MSE) loss between predicted and ground-truth magnitudes at each stroke vertex:

$$\mathcal{L}_{\text{mag}} = \frac{1}{|\mathcal{P}|} \sum_{p \in \mathcal{P}} \left(\log_2\left(\|\hat{\mathbf{d}}_p\| + \epsilon\right) - \log_2\left(\|\mathbf{d}_p\| + \epsilon\right)\right)^2. \tag{3}$$

Physics-Guided Regularization. A stroke cloud can be viewed as a family of curves lying on an ambient user-intended parametric surface $\sigma(u,v)$, whose deformation $\mathbf{d}(u,v) := \sigma'(u,v) - \sigma(u,v)$ can be described by the following *thin-shell energy* consisting of stretching and bending terms:

$$E(\mathbf{d}) = \iint_{\Omega} k_s \left(\|\mathbf{d}_u\|^2 + \|\mathbf{d}_v\|^2\right) + k_b \left(\|\mathbf{d}_{uu}\|^2 + 2\,\|\mathbf{d}_{uv}\|^2 + \|\mathbf{d}_{vv}\|^2\right) du\, dv, \tag{4}$$

where k_s and k_b are the corresponding stiffness coefficients. Since the stroke cloud is a set of vertices along curves, we interpret v as the along-curve parameter and u as the parameter indexing the M curves in the family, corresponding to the vertex index j and stroke index i, respectively. The discrete counterpart of Equation (4) reads as follows (see Supplementary Material for more details):

$$E(\mathbf{d}) = \underbrace{k_s \sum_{i=1}^{M} \sum_{j=1}^{N_i - 1} \left\|\mathbf{d}_{i,j+1} - \mathbf{d}_{i,j}\right\|^2}_{\mathcal{L}_s} + \underbrace{k_b \sum_{i=1}^{M} \sum_{j=2}^{N_i - 1} \left\|\mathbf{d}_{i,j+1} - 2\mathbf{d}_{i,j} + \mathbf{d}_{i,j-1}\right\|^2}_{\mathcal{L}_b}, \tag{5}$$

where $\mathbf{d}_{i,j}$ is the displacement at the j-th vertex of the i-th stroke and N_i is its corresponding vertex count. We use this formulation as our regularization term: $\mathcal{L}_s$ penalizes displacement differences between adjacent stroke vertices, corresponding to in-plane stretching, while $\mathcal{L}_b$ penalizes non-smooth displacement variation along each stroke, corresponding to out-of-plane bending.

Therefore, in the light of Equations (1), (2), (3), and (5), we propose the following objective function:

$$\mathcal{L} = \lambda_{\text{ang}}(\mathcal{L}_{\text{ang},n} + \mathcal{L}_{\text{ang},t}) + \lambda_{\text{mag}}\mathcal{L}_{\text{mag}} + \lambda_s \mathcal{L}_s + \lambda_b \mathcal{L}_b, \tag{6}$$

where we set $\lambda_{\text{ang}} = 100$, $\lambda_{\text{mag}} = 10$, $\lambda_s = 10$, and $\lambda_b = 10^{-5}$ following our ablations (see Section 4).

Proceedings published by Eurographics - The European Association for Computer Graphics.

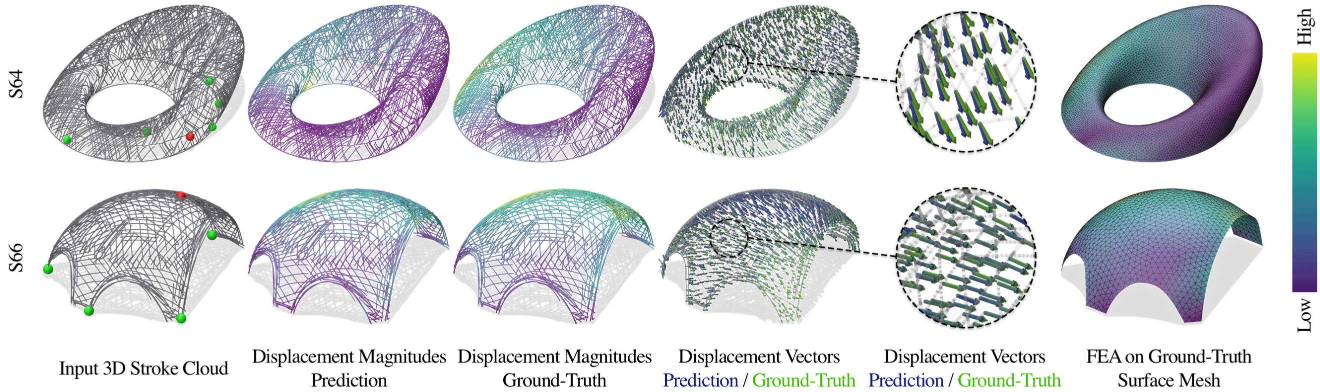

Figure 3: *Qualitative results showcasing predicted and ground-truth displacement magnitudes and vectors, alongside reference FE simulations on the underlying surface mesh. Close-ups highlight the alignment of predicted (blue) and ground-truth (green) displacement vectors.*

		Topology Split			Boundary Condition Split		
Variant	Weights $(\lambda_{\text{ang}}, \lambda_{\text{mag}}, \lambda_s, \lambda_b)$	MAE	EMD	COS	MAE	EMD	COS
w/o physics	$(100, 10, 0, 0)$	0.0402	0.0381	0.1076	0.0258	0.0227	0.0766
w/ stretching only	$(100, 10, 10, 0)$	0.0381	**0.0358**	0.1158	0.0241	0.0209	0.0634
w/ bending only	$(100, 10, 0, 10^{-5})$	0.0392	0.0366	0.1097	**0.0229**	**0.0196**	0.0672
w/ stretching and bending	$(100, 10, 10, 10^{-5})$	**0.0377**	0.0361	**0.0987**	0.0272	0.0242	**0.0580**

Table 1: *Quantitative results across four loss variants on the test set of each split, analyzing the effect of physics-informed regularization terms. Mean Absolute Error (MAE) and Earth Mover's Distance (EMD) measure the average error and distributional discrepancy of predicted displacement magnitudes, while Cosine distance (COS) measures the angular deviation in displacement vectors. Across all evaluations, the reference FEA results are computed on the ground-truth surface models and mapped to the corresponding stroke cloud vertices.*

4. Evaluation

We train the network using an 8:1:1 train/validation/test split under two strategies—topology-based and boundary-condition-based—to assess generalization to out-of-distribution sketch topologies and load/support placements (see Supplementary Material for implementation details). We present qualitative results on unseen 3D sketches of architectural thin-shell structures (see Figure 3) and report quantitative comparisons across four loss variants to analyze the individual and combined effects of the physics-informed regularization terms, namely stretching and bending (see Table 1).

5. Conclusion and Future Work

We presented a physics-informed neural network for predicting 3D deformation fields directly on 3D sketch stroke clouds without requiring intermediate surface reconstruction or explicit FE simulation. Our method accepts inputs stemming from a variety of sources, such as 3D sketching interfaces, and traditional modeling software (such as Blender or Rhino) or various curve tracing algorithms. On both unseen sketch topologies and unseen boundary configurations, our evaluations show close agreement with reference FEA results. For future work, we envision extending the synthetic data pipeline to a wider range of materials and multi-material structures, predicting stress fields, and using the inferred fields to guide topology optimization of 3D sketched structures.

Acknowledgments

This research was funded in whole or in part by the Austrian Science Fund (FWF) [10.55776/F77] through the SFB "Advanced Computational Design" (subproject 2). The third author, A. Rasoulzadeh (METROM GmbH), was financed by the European Union, Sächsische Aufbaubank – Förderbank (application number 100750317).

References

[NPTB22] Neveu W., Puhachov I., Thomaszewski B., Bessmeltsev M.: Stability-aware simplification of curve networks. In *ACM SIGGRAPH 2022 Conference Proceedings* (2022), pp. 1–9. 2

[RSK*23] Rasoulzadeh S., Senk V., Königsberger M., Reisinger J., Kovacic I., Füssl J., Wimmer M.: A novel integrative design framework combining 4d sketching, geometry reconstruction, micromechanics material modelling, and structural analysis. *Advanced Engineering Informatics 57* (2023), 102074. 2

[WHS*24] Williams F., Huang J., Swartz J., Klar G., Thakkar V., Cong M., Ren X., Li R., Fuji-Tsang C., Fidler S., et al.: fvdb: A deep-learning framework for sparse, large scale, and high performance spatial intelligence. *ACM Transactions on Graphics (TOG) 43*, 4 (2024), 1–15. 2

[WY25] Wang R., Yu R.: Physics-guided deep learning for dynamical systems: A survey. *ACM Computing Surveys 58*, 5 (2025), 1–31. 2

[YXLF23] Yu D., Xiao C., Lau M., Fu H.: Sketch2stress: sketching with structural stress awareness. *IEEE Transactions on Visualization and Computer Graphics 30*, 10 (2023), 6851–6865. 2