NeuVAS: Neural Implicit Surfaces for Variational Shape Modeling

PENGFEI WANG, Shandong University, China
QIUJIE DONG, Shandong University, China, The University of Hong Kong, China, and TransGP, China
FANGTIAN LIANG, Shandong University, China
HAO PAN, Tsinghua University, China
LEI YANG, University of Hong Kong, China
CONGYI ZHANG, University of British Columbia, Canada
GUYING LIN, University of Hong Kong, China
CAIMING ZHANG, Shandong University, China
YUANFENG ZHOU, Shandong University, China
CHANGHE TU, Shandong University, China
SHIQING XIN, Shandong University, China
ALLA SHEFFER, University of British Columbia, Canada
XIN LI, Texas A&M University, United States of America
WENPING WANG*, Texas A&M University, United States of America

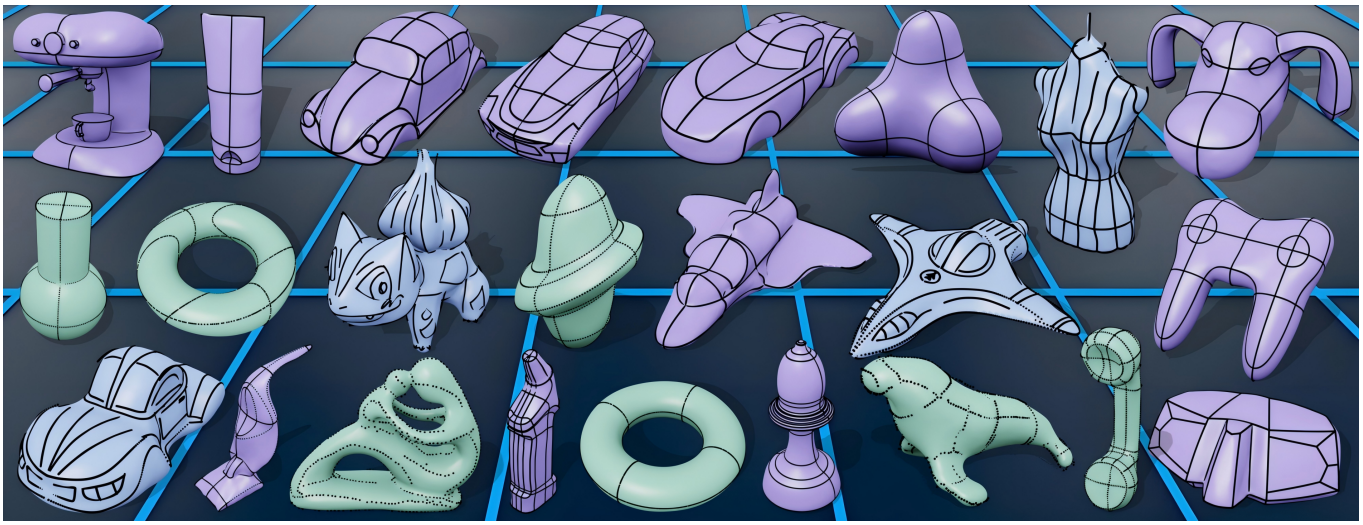


Fig. 1. A gallery of surfaces constructed using our method NeuVAS, with three kinds of sparse shape control input: connected curve networks (purple), unstructured curve sketches (blue), and sparse point clouds (green). Connected curve networks are a set of curves connected to form loops to define surface patches. In contrast, unstructured curve sketches are a collection of disconnected curve segments defining the outline of a shape to be designed. Both curve networks and curve sketches can be discretized into point clouds, which serve as the input for NeuVAS.

Neural implicit shape representation has drawn significant attention in recent years due to its smoothness, differentiability, and topological flexibility. However, directly modeling the shape of a neural implicit surface, especially as the zero-level set of a neural signed distance function (SDF), with sparse geometric control is still a challenging task. Sparse input shape control typically includes 3D curve networks or, more generally, 3D curve sketches, which are unstructured and cannot be connected to form a curve network,

*Corresponding author: Wenping Wang.

Authors' Contact Information: Pengfei Wang, Shandong University, Jinan, China, 8144756@qq.com; Qiujie Dong, Shandong University, Qingdao, China and The University of Hong Kong, Hong Kong, China and TransGP, Hong Kong, China, qiujie.jay.dong@gmail.com; Fangtian Liang, Shandong University, Jinan, China, lft00@foxmail.com; Hao Pan, Tsinghua University, Beijing, China, haopan@tsinghua.edu.cn; Lei Yang, University of Hong Kong, Hong Kong, China, yanglei.dalian@gmail.com; Congyi Zhang, University of British Columbia, Vancouver, Canada, zhcy@outlook.com; Guying Lin, University of Hong Kong, Hong Kong, China, guyinglin2000@gmail.com; Caiming Zhang, Shandong University, Jinan, China, czhang@sdu.edu.cn; Yuanfeng Zhou, Shandong University, Jinan, China, yfzhou@sdu.edu.cn; Changhe Tu, Shandong University, Qingdao, China, chtu@sdu.edu.cn; Shiqing Xin, Shandong University, Qingdao, China, xinshiqing@sdu.edu.cn; Alla Sheffer, University of British Columbia, Vancouver, Canada, sheffa@cs.ubc.ca; Xin Li, Texas A&M University, Texas, United States of America, xinli@tamu.edu; Wenping Wang, Texas A&M University, Texas, United States of America, wenping@tamu.edu.

ACM 1557-7368/2025/12-ART
https://doi.org/10.1145/3763331

and therefore more difficult to deal with. While 3D curve networks or curve sketches provide intuitive shape control, their sparsity and varied topology pose challenges in generating high-quality surfaces to meet such curve constraints. In this paper, we propose NeuVAS, a variational approach to shape modeling using neural implicit surfaces constrained under sparse input shape control, including unstructured 3D curve sketches as well as connected 3D curve networks. Specifically, we introduce a smoothness term based on a functional of surface curvatures to minimize shape variation of the zero-level set surface of a neural SDF. We also develop a new technique to faithfully model G^0 sharp feature curves as specified in the input curve sketches. Comprehensive comparisons with the state-of-the-art methods demonstrate the significant advantages of our method.

CCS Concepts: • **Computing methodologies** → *Parametric curve and surface models.*

ACM Reference Format:
Pengfei Wang, Qiujie Dong, Fangtian Liang, Hao Pan, Lei Yang, Congyi Zhang, Guying Lin, Caiming Zhang, Yuanfeng Zhou, Changhe Tu, Shiqing Xin, Alla Sheffer, Xin Li, and Wenping Wang. 2025. NeuVAS: Neural Implicit Surfaces for Variational Shape Modeling. *ACM Trans. Graph.* 44, 6 (December 2025), 14 pages. https://doi.org/10.1145/3763331

1 Introduction

Surfacing a collection of 3D curves, also known as lofting or skinning, is a fundamental and challenging task in geometric modeling. This task involves three primary challenges: 1) finding a surface that accurately approximates the given curves; 2) controlling the shape away from the input curves to produce natural, aesthetic shape interpolation; and 3) constructing piecewise smooth surfaces to faithfully model G^0 sharp feature curves. In fact, most curve-based inputs are now manually drawn using AR/VR devices [Yu et al. 2022], and therefore often unconnected and inaccurate curve collections. This lack of structure makes the surface modeling even more difficult and renders some existing algorithms (e.g. [Pan et al. 2015]) ineffective because they rely on structured curve networks as input.

Existing methods for surfacing 3D curve collections can be roughly divided into two categories: mesh-based methods [Pan et al. 2015] and implicit surface methods [Huang et al. 2019]. The mesh-based methods can faithfully preserve sharp curve features by decomposing curve collections into independent loops to define individual patches. However, they require structured curve networks as inputs and cannot process unstructured curve sketches, thus limiting the scope of their application. Moreover, the performance of these methods is limited by the quality and resolution of mesh surfaces. Implicit surface methods, on the other hand, are more suitable for constructing surfaces from curve sketches due to their flexible topology and the access to accurate curvature computation. However, these methods face challenges in precise shape control, especially when constructing shapes with sharp features.

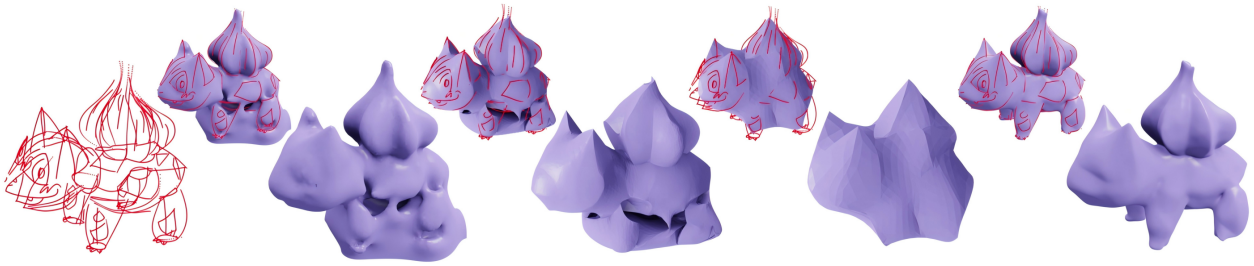
There is also a hybrid approach that combines both explicit and implicit surface representation [Yu et al. 2022]. The method of [Yu et al. 2022] needs a mesh surface as input, which is presumably provided by another surfacing method from input curves. Then it segments the input mesh into patches and fits these patches with implicit polynomial surfaces. Moreover, it needs a very good initial surface to start with to be successful. As acknowledged in [Yu et al. 2022], since the existing methods sometimes cannot produce such a good initial surface, one has to manually design an initial surface, making the method inefficient and unreliable. See Fig. 2, and more details of comparison in the Experiment Section.

We propose a variational approach based on neural implicit surfaces for modeling high-quality surfaces from sparse shape control input, which including disconnected curve sketches, as well as connected curve networks and sparse points, therefore extending beyond only taking 3D connected curve networks as input. Here the neural implicit surface is modeled by the zero-level set of a neural Signed Distance Function (SDF) encoded by a Multi-Layer Perceptron (MLP). To regularize this surface, we use the Dirichlet condition [Lipman 2021] and the Eikonal condition [Gropp et al. 2020] to ensure interpolation of the curves. To regularize the surface shape in regions away from the input curves, we adopt a smoothness energy term based on the thin-plate energy [Welch and Witkin 1994] (Figure 3(a)).

To tackle the challenge of preserving sharp feature curves, we assign a location-dependent weight to the smooth energy term. More specifically, the weight at a surface location depends on its distance to sharp feature curves - the weight is larger when its location is away from the sharp feature curves, and the weight vanishes to zero when its location approaches the sharp feature curve. This strategy facilitates the modeling of shape feature curves because it separates the influences of the smoothness terms from the adjacent surfaces sharing a sharp feature curve, as shown in Figure 3(b). A gallery of surfaces constructed using NeuVAS from various sparse inputs is shown in Figure 1. Figure 1 includes structured curve networks (purple) as well as more challenging inputs: unstructured and imperfect sketches (blue), and sparse points sampled along curves (green). The latter two are commonly recognized as lower-quality input. The high-quality curves are typically used in CAD modeling, and the lower-quality inputs are more common in sketch-based modeling, especially in immersive VR/AR environments. A key contribution of our method is to handle not only clean structured curves but also disconnected and inaccurate curve collections. To further validate the robustness of our method, we add Gaussian noises to the input curves and show the results in Figure 13.

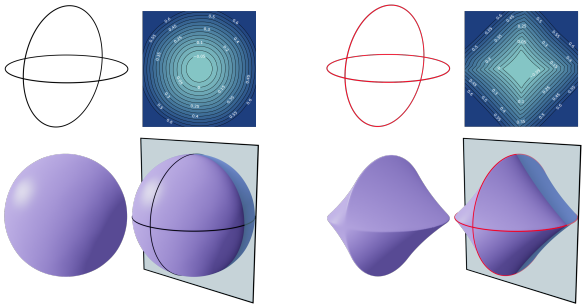
We note that the problem of variational shape modeling addressed in the present paper is fundamentally different from the conventional surface fitting task that seeks to over-fit a surface to a dense set of data points, possibly associated with surface normals. In the surface fitting task, the primary focus is the accurate approximation to the data points, therefore on-surface constraints and the Eikonal constraint for SDF regularization are sufficient. In contrast, for variational shape modeling, the input is a sparse set of 3D curves, which are insufficient to completely determine a definite surface shape. Therefore, in addition to ensuring interpolation of the curves, it is essential to include a surface energy term to regularize the surface shape in the region between the curves to produce an overall natural and aesthetic surface shape. In the Experiment Section 4.2, we will show that simply applying a surface fitting method to sparse input curves will lead to unsatisfactory results (Figure 6).

Our main contributions are as follows:



(a) Input (b) VIPSS [Huang et al. 2019] (c) Yu (+VIPSS) [Yu et al. 2022] (d) Yu (+Sphere) [Yu et al. 2022] (f) NeuVAS (Ours)

Fig. 2. Comparisons with baseline methods capable of surfacing unstructured curve sketches as input. More comparison results are shown later in Section 4.2.



(a) Smooth (b) Piecewise Smooth

Fig. 3. Shape control with sharp feature curves. (a) *The input curves contain no sharp features (in black)*: Using a smooth energy term, we produce a smooth transition between adjacent surface patches. (b) *The input curves are designated to be sharp features (in red)* : By reducing the influence of the smooth energy constraint near the input curves, sharp features are achieved.

- We present a new method, NeuVAS, for high-quality variational surface modeling based on a neural implicit surface representation. Our method can take general sparse shape control as input, including disconnected curve sketches, as well as connected curve networks and sparse points.
- We devise an effective strategy to preserve sharp feature curves on the output surface. This is achieved by properly reducing the influence of the surface energy constraints near designated sharp feature curves.
- We conduct extensive experiments to validate our method and present comprehensive comparisons to demonstrate its advantages over existing methods.

2 Related Work

Surfacing curve networks. The problem of surfacing curve networks emerged with the advent of practical user interfaces, algorithms, and devices designed to create these networks. Early 2D interfaces deduced the depth of curves from their intersections with other curves [Schmidt et al. 2009; Xu et al. 2014] or with sketching planes [Bae et al. 2008], effectively producing well-connected curve networks by design. As a result, numerous surfacing algorithms heavily rely on the connectivity of the curve network to identify closed loops that delimit surface patches [Abbasinejad et al. 2012; Orbay and Kara 2012; Sadri and Singh 2014; Zhuang et al. 2013]. Each such patch can then be surfaced by propagating geometric information from the boundary curves [Pan et al. 2015; Yu et al. 2021; Zhuang et al. 2013]. Curve networks connectivity information provides strong geometric hints, as surface normals can be estimated at each intersection, as demonstrated by [Pan et al. 2015] to detect sharp features and determine which curves are flow lines. However, the reliance on accurate curve network topology prevents these methods from working on raw, unstructured stroke clouds typically produced via freehand AR/VR sketching.

Recent studies have shown that precise sketching in VR is more challenging than in 2D due to the lack of a supporting surface for the hand and the need for finer motor control to position curves in 3D [Arora et al. 2017; Barrera Machuca et al. 2019]. Automatic snapping can help correct inaccuracies [Machuca et al. 2018; Yu et al. 2021], but such assistance can be detrimental to user creativity. For instance, users of the recent CASSIE system [Yu et al. 2021] criticized the system for forcing them to think of their designs in terms of curve networks. Processing 3D sketches to form well-connected curve networks is both challenging and often undesirable. First, many curves in 3D sketches are not intended to connect to others, and forcing such connections can alter the original design intent. Secondly, 3D sketches often exhibit oversketching and imprecision, making the formation of clean curve networks from 3D curves a challenging problem, similar to 2D vectorization.

We propose an approach that is oblivious to curve network connectivity, achieving high robustness to inaccuracies and to detailed curves that lie on the envisioned surface but are not connected to other curves. Our work also differs from interactive systems designed to model 3D surfaces by drawing in 2D [Dvorožňák et al. 2018; Li et al. 2017; Nealen et al. 2007]. Users of these systems draw in dedicated interfaces and provide annotations of surface discontinuities. In contrast, our method takes point clouds as input and employs a smooth distance function to accurately recover sharp features.

Surfacing curve sketches. Curve sketches are typically manually drawn using AR/VR devices [Rosales et al. 2019]. However, there are currently few methods that specifically target curve sketches.

In the computer vision community, a few methods have been proposed to construct curve sketches by matching edges in multi-view stereo algorithms [Fabbri and Kimia 2010]. [Usumezbas et al. 2017] proposed a surfacing algorithm dedicated to such unstructured 3D curve sketches, where candidate surface patches are lofted between pairs of curves. However, this method heavily relies on the availability of multiple photographs of the shape to select valid candidate patches based on occlusion reasoning. [Batuhan Arisoy et al. 2012] surface sparse and imprecise curve sketches by smoothly deforming an initial low-fidelity surface of correct topology, using a discrete guidance vector field that points towards the closest curve point. This approach produces globally smooth surfaces and requires user intervention to specify curves that should be inserted into the mesh as sharp-edge polylines.

Recently, [Yu et al. 2022] introduced a multi-model fitting approach that generates piecewise-smooth surfaces. This method segments the surface into multiple patches, representing each patch as the zero-level set of an implicit polynomial. However, their approach has two notable shortcomings. Firstly, it heavily relies on the accuracy of the initial surface; when the initial surface deviates significantly from the real geometry, the method fails to restore the correct shape. Secondly, the representation ability of implicit polynomials of degree up to 4 is limited, leading to the inability to recover small geometric details. In contrast, our method does not require any other initialization methods; we simply take point clouds as input and leverage a neural network with strong representation abilities to learn the signed distance function (SDF).

Surfacing point clouds. The curve networks and curve sketches we consider can be easily converted to point clouds by sampling points along each curve. This approach enables us to leverage the wealth of methods developed for surfacing such unstructured 3D data [Berger et al. 2017]. However, most existing methods are designed to process dense point clouds acquired using 3D scanning technology [Hoppe et al. 1992; Kazhdan et al. 2006], and thus struggle when applied to 3D sketches, which provide a very sparse, non-uniform sampling of the envisioned surface. While some methods are robust to missing data, they typically only address relatively small holes compared to the overall surface scale [Hornung and Kobbelt 2006] or guide surface completion by fitting geometric primitives on dense parts of the point cloud [Schnabel et al. 2009; Tagliasacchi et al. 2009].

The thin-plate RBF [**?**] and VIPSS [Huang et al. 2019] algorithm achieves impressive resilience to sparsity and non-uniformity of point clouds, and has even been demonstrated on unstructured point clouds. However, this robustness is achieved through a global smoothness energy that may overlook sharp surface features. Moreover, to avoid degenerate solutions such as a constant zero interpolant, thin-plate RBF requires additional spatial constraints with prescribed signed values. Additionally, both thin-plate RBF and VIPSS have a computational complexity of $O(n^3)$, which limits their practical application to point clouds of up to around 5k points. [Xu et al. 2023] proposed point cloud orientation techniques and applied them to construct unstructured 3D data. They compute the normal vectors based on global consistency and then use Poisson reconstruction [Kazhdan and Hoppe 2013] to generate the mesh surface. However, their approach is ill-suited for reconstructing curve networks. On one hand, [Xu et al. 2023] relies on the global consistency of normal vectors to generate smooth surfaces, which makes it incapable of handling curve networks with sharp features. On the other hand, even with accurately specified normal vectors, [Xu et al. 2023] struggles to generate a reasonable shape due to significant gaps in regions far from the curve network. Furthermore, [Xu et al. 2023] exhibits high sensitivity to both the number of points and the shape of the curve network. It is limited to point cloud scales below 10K, rendering it inadequate for representing complex geometric shapes.

Our approach, based on neural signed distance functions (SDF) [Jones et al. 2006], is self-supervised, and its loss includes a thin-plate energy term to constrain the empty space between the curve network while remaining unconstrained near feature curves. We optimize the patches on both sides of the feature curve independently, naturally achieving piecewise smoothness.

3 Method

Our NeuVAS method takes sparse shape control as input, including disconnected curve sketches, connected curve networks and sparse points, produces a neural implicit surface that is a natural, visually aesthetic surface, while satisfying the input curve constraints. The design of NeuVAS has three key aspects. (1) It uses the Dirichlet condition [Lipman 2021] and the Eikonal condition [Gropp et al. 2020] to ensure the input curves are accurately interpolated; (2) NeuVAS uses a surface smoothness term based on the thin-plate energy to produce smooth and natural transitions between individual curve; and (3) NeuVAS relaxes the smooth constraint near those curves that are designated as sharp feature curves to yield a final surface that faithfully preserves the sharp features. These three aspects will be elaborated in the following subsections.

3.1 Interpolating Input Curves

The input to our method is assumed to be curve sketches, which are typically a set of sparse, unstructured 3D curve segments. Given such curve sketches as input, our first goal is to compute a neural implicit surface that interpolates the individual input curves. This surface, denoted by S, is represented as the zero-level set surface of a neural Signed Distance Function (SDF), denoted as $f(\boldsymbol{x};\Theta) : \mathbb{R}^3 \rightarrow \mathbb{R}$, which is encoded as an MLP, where Θ are the network parameters (i.e., weights). That is, $S = \{\boldsymbol{x} \in \mathbb{R}^3 | f(\boldsymbol{x};\Theta) = 0\}$.

Fitting a neural implicit surface S to data points is a well-studied topic [Atzmon and Lipman 2020; Ben-Shabat et al. 2022; Gropp et al. 2020; Ma et al. 2021; Sitzmann et al. 2020; **?**]. In the following, we provide a brief account of the loss terms used in our implementation, and refer to the details in the existing works, e.g. [Sitzmann et al. 2020]. For interpolating the curve sketches, our loss terms are based on the Eikonal condition, the Dirichlet condition, and a regularization term to avoid extraneous zero-level set points.

Eikonal Condition. We use the Eikonal loss to promote an approximate SDF field. First, it helps regularize the implicit field, leading to cleaner zero-level sets, as discussed in [Gropp et al. 2020]. Second, our point sampling strategy relies on having an approximate SDF to ensure distances and gradients are reliable near the surface. Let $\mathcal{P}$ denote the set of sample points of all the input curve sketches, where

the SDF is supposed to be zero. Let Q be a bounding region of the input data, which is assumed to be normalized to the range $[-0.5, 0.5]^3$ by default. By the Eikonal condition, an SDF $f(\boldsymbol{x};\Theta)$ possesses a unit gradient, i.e. $\|\nabla_{\boldsymbol{q}} f\| = 1$, at any point $\boldsymbol{q} \in Q$. To constrain our neural function $f(\boldsymbol{x};\Theta)$ to represent an SDF, we use the Eikonal condition to define the loss term:

$$\mathcal{L}_E = \frac{1}{|Q|} \sum_{\boldsymbol{q}\in Q} |1 - \|\nabla f(\boldsymbol{q};\Theta)\|| \,. \tag{1}$$

Dirichlet Condition. To ensure that the neural implicit surface S passes through the input curve sketches, we require that for any point $\boldsymbol{p} \in \mathcal{P}$, we have $f(\boldsymbol{p};\Theta) = 0$ as much as possible. This leads to the loss term:

$$\mathcal{L}_{DM} = \frac{1}{|\mathcal{P}|} \sum_{\boldsymbol{p}\in\mathcal{P}} |f(\boldsymbol{p};\Theta)|. \tag{2}$$

Meanwhile, we keep the function f from being zero at points away from the curve sketches, using the constraint term [Sitzmann et al. 2020]:

$$\mathcal{L}_{\text{DNM}} = \frac{1}{|Q|} \sum_{\boldsymbol{q}\in Q} \exp(-\alpha|f(\boldsymbol{q};\Theta)|). \tag{3}$$

Put together, the total loss for interpolating the curve sketches is:

$$\mathcal{L}_{\text{interp}} = \lambda_{\text{E}}\mathcal{L}_{\text{E}} + \lambda_{\text{DM}}\mathcal{L}_{\text{DM}} + \lambda_{\text{DNM}}\mathcal{L}_{\text{DNM}}. \tag{4}$$

This ensures that the neural function f properly represents an SDF that accurately passes through the input curves.

3.2 Surface Smoothness Energy

To encourage the final surface S to take a natural smooth shape in regions between the sparse input curves, we adopt the following smoothness energy term, based on the thin-plate energy, as an additional loss term:

$$\mathcal{L}_{\text{Smooth}} = \frac{1}{|S|} \sum_{s\in S} \left(\kappa_1^2(s) + \kappa_2^2(s)\right), \tag{5}$$

where κ_1 and κ_2 are the principal curvatures of the underlying surface S. Let $H(\boldsymbol{s})$ and $K(\boldsymbol{s})$ denote the mean curvature and the Gaussian curvature at the surface point $\boldsymbol{s}$, respectively. Since

$$H(\boldsymbol{s}) = [\kappa_1(\boldsymbol{s}) + \kappa_2(\boldsymbol{s})]/2, \quad K(\boldsymbol{s}) = \kappa_1(\boldsymbol{s})\kappa_2(\boldsymbol{s}),$$

the term $\mathcal{L}_{\text{Smooth}}$ can be rewritten as:

$$\mathcal{L}_{\text{Smooth}} = \frac{1}{|S|} \sum_{s\in S} \left(4H^2(\boldsymbol{s}) - 2K(\boldsymbol{s})\right), \tag{6}$$

where $H(\boldsymbol{s})$ and $K(\boldsymbol{s})$ can be computed from the Hessian matrix $\mathbb{H}_f(\boldsymbol{s})$ of f as follows [Goldman 2005; Knoblauch 1913; Spivak 1975]:

$$H(\boldsymbol{s}) = \frac{\nabla f(\boldsymbol{s};\Theta)\mathbb{H}_f(\boldsymbol{s})\nabla f^T(\boldsymbol{s};\Theta) - \|\nabla f(\boldsymbol{s};\Theta)\|^2 Trace(\mathbb{H}_f(\boldsymbol{s}))}{2\|\nabla f(\boldsymbol{s};\Theta)\|^3}. \tag{7}$$

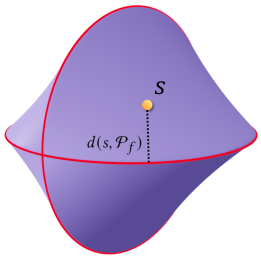
$$K(\boldsymbol{s}) = -\frac{\begin{vmatrix} \mathbb{H}_f(\boldsymbol{s}) & \nabla f^T(\boldsymbol{s};\Theta) \\ \nabla f(\boldsymbol{s};\Theta) & 0 \end{vmatrix}}{\|\nabla f(\boldsymbol{s};\Theta)\|^4}. \tag{8}$$

Remark. Note that we use general curvature formulas instead of simpler formulas specific to SDFs. The reason is that, although our network is trained to approximate an SDF via the Eikonal loss, the resulting field is only an approximate SDF and may have considerable deviations from a true distance field, especially during training. As our focus is on accurately estimating curvature near the zero-level set, we chose to use the general formulas for mean and Gaussian curvature that apply to implicit fields, rather than relying on properties specific to true SDFs.

3.3 Creating Sharp Feature Curves

In order to create a surface with sharp feature curves, i.e., curves along which the surface has G^0 continuity but non-continuous normal directions, the user may designate some of the input curves as feature curves. Let $\mathcal{P}$ denote the set of points on *all* the input curves (Fig 4 both red and black curve), and $\mathcal{P}_f$ denote the set of points on the feature curves (Fig 4 red curve), thus $\mathcal{P}_f \subset \mathcal{P}$. To faithfully create the sharp features as specified, we need to reduce the influence of the smoothness term near and on the specified input feature curves; otherwise, enforcing the smoothness energy everywhere on the surface S would result in an overall smooth surface without any sharp features, as shown in Fig. 3(a).

To make the smoothness energy constraints decay around the feature curves, we introduce a weight function $d^2(s,\mathcal{P}_f)$, where $d(s,\mathcal{P}_f)$ is the Euclidean distance from a surface point $s \in S$ to $\mathcal{P}_f$, the set of points on the input feature curves. See the inset figure. Then, incorporate this weight function to devise the following modified smoothness energy term:

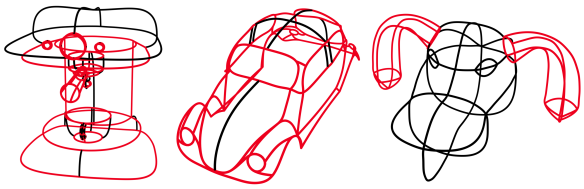


$$\mathcal{L}_{\text{Smooth}} = \frac{1}{|S|} \sum_{s\in S} \left(4H^2(\boldsymbol{s}) - 2K(\boldsymbol{s})\right) \cdot d^2(\boldsymbol{s},\mathcal{P}_f). \tag{9}$$

The geometric intuition of this design is clear. The $\mathcal{L}_{\text{Smooth}}$ has small effect around the feature curves P_f because the weight $d^2(\boldsymbol{x},\mathcal{P}_f)$ is small there, and $d^2(\boldsymbol{x},\mathcal{P}_f) = 0$ for $\boldsymbol{x}$ on the feature curves. As a consequence, there is no smoothness constraint across the feature curves, leading to two adjacent surface patches sharing a feature curve to join with G^0 continuity. Furthermore, the absence of the smoothness energy would facilitate the accurate surface interpolation on the feature curve, where the loss term $\mathcal{L}_{\text{interp}}$ in Eq 4 is dominant. In our implementation, due to the normalization, the distance weight can be naturally bounded.

Designating Feature Curves. For the curve networks, we extract feature curves by measuring the variation in surface normals at their two endpoints, as described in [Pan et al. 2015]. These feature curves are detected automatically. For further details, please refer to [Pan et al. 2015]. Figures 4 and 7 illustrate examples of detected feature curves and the resulting surfaces. In the case of an curve sketches, where feature curves cannot be automatically extracted, we treat all curves as feature curves. For sparse point clouds, all

curves are considered smooth curves by default. Additionally, we support user-specified sharp feature curves. In the following, feature curves are shown in red, and smooth curves in black.



(a) Espresso (b) Loftbug (c) Doghead

Fig. 4. Results for detecting curve features in curve networks. The detected feature curves are shown in red.

3.4 Implementation Details

Combining Eq. 4 and Eq. 9, our total loss is given by:

$$\mathcal{L} = \lambda_{\mathrm{E}}\mathcal{L}_{\mathrm{E}} + \lambda_{\mathrm{DM}}\mathcal{L}_{\mathrm{DM}} + \lambda_{\mathrm{DNM}}\mathcal{L}_{\mathrm{DNM}} + \tau\lambda_{\mathrm{Smooth}}\mathcal{L}_{\mathrm{Smooth}}, \quad (10)$$

where τ is the cosine factor [Loshchilov and Hutter 2022] used to modulate the influence of the thin-plate energy loss term during training. Empirically, the cosine factor τ is initialized to 1, with 1K iterations constituting one cycle. Indeed, due to the conflict between the thin-plate energy term $\mathcal{L}_{\mathrm{Smooth}}$ and the Dirichlet term $\mathcal{L}_{\mathrm{DM}}$, the zero-level set of SDF f struggles to interpolate the input curve network. We use the cosine factor to gradually reduce the thin-plate term $\mathcal{L}_{\mathrm{Smooth}}$, encouraging the zero level set of SDF to interpolate the curve networks, and then increase the smoothing term $\mathcal{L}_{\mathrm{Smooth}}$ to evolve towards a reasonable shape. Note that once the surface aligns with the curve networks, the zero level set of SDF f near the curve networks will not be constrained by the thin-plate term $\mathcal{L}_{\mathrm{Smooth}}$ due to its distance based weighting, resulting in the surfacing sticking to the curve network. In practice, a complex model may contain features at significantly different scales; by leveraging the cosine factor with periodic annealing and restarts, our algorithm is able to recover these feature curves effectively.

Sampling on SDF Zero-Level Set. To impose constraints on a surface region between curve, we need to sample at the zero-level set to measure the loss. However, accurate sampling at the zero-level set in each iteration would incur significant computational overhead, whether using Marching Cubes [Lorensen and Cline 1998] to extract zero-level set surfaces explicitly or iteratively projecting spatial samples away from the zero-level set onto it implicitly [Wang et al. 2020]. To mitigate this issue, we note that the change of the learned zero-level set between two consecutive iterations is small. Hence, for two iterations, we combine the strategy of zero-level set extraction and sample points projection. Namely, we reuse the point samples on the previous zero-level set and project them onto the current one. Meanwhile, to ensure sampling uniformity and avoid cumulative errors, we schedule a step of Marching Cubes to regenerate sample points for every 100 iterations (including the initial iteration).

In particular, for a given neural SDF f, we extract the zero-level set mesh using Marching Cubes at resolution 128^3. Then, we use Poisson-disk sampling on the zero-level set mesh to obtain uniform points. Through experiments, we find that setting the number of sampling points to 10K is reasonable and sufficient for almost all models, as shown in Figure 1. In each iteration, we employ a projection strategy [Wang et al. 2020] to project the point set onto the zero-level. For a point $\boldsymbol{x}$, the projection $\boldsymbol{x}'$ is calculated as follows:

$$\boldsymbol{x}' = \boldsymbol{x} - \frac{\nabla f(\boldsymbol{x};\Theta)}{\|\nabla f(\boldsymbol{x};\Theta)\|} \cdot f(\boldsymbol{x};\Theta). \quad (11)$$

4 Experiments

4.1 Experiment Setup

Architecture. Similar to various implicit surface reconstruction methods [Ben-Shabat et al. 2022; Lipman 2021; Wang et al. 2022], our NeuVAS employs the IGR [Gropp et al. 2020] network architecture, featuring 8 hidden layers, each with 256 units, resulting in a model of 1.86M parameters. The activation function used is softplus. Inputs are normalized to the range $[-0.5, 0.5]^3$ before being fed into the network.

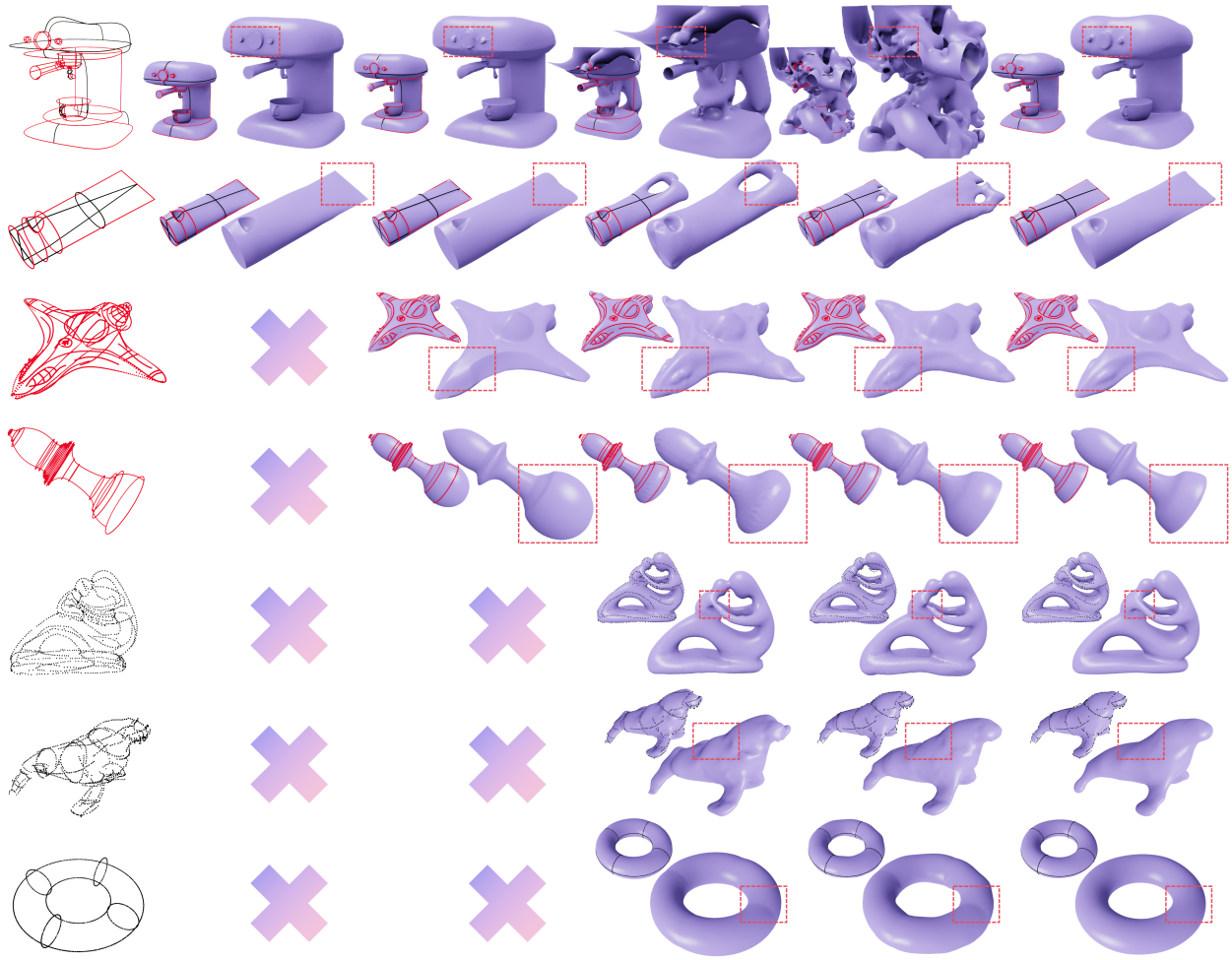
Parameters. In our experiments, we utilized the thin-plate energy and determined the weights $\lambda_{\mathrm{E}} = 0.1$, $\lambda_{\mathrm{DM}} = 100$, and $\lambda_{\mathrm{DNM}} = 10$. Empirically, we recommend setting $\lambda_{\mathrm{Smooth}}$ to 5×10^{-4}. In each iteration, we sample Q of 10K points uniformly inside the bounding box (normalized to $[-0.5, 0.5]^3$); additionally, a Q_{zero} of 10K points are uniformly sampled on the zero-level set. Throughout the training phase, we applied the Adam optimizer [Kingma and Ba 2014] with a default learning rate of 5×10^{-5} and completed training in 10,000 iterations. For visualization, meshes are extracted from the zero-level set using the Marching Cubes algorithm at a consistent resolution of 512^3 for all methods under comparison. The experiments were executed on an NVIDIA GeForce RTX 4090 graphics card equipped with 24GB of video memory along with an Intel(R) Core i9-13900k processor.

Datasets. The input data we use include curve networks [Pan et al. 2015], curve sketches [Yu et al. 2022], and sparse point clouds [Huang et al. 2019].

Curve networks are well-connected curves that can be decomposed into separate curve loops (e.g., Espresso and Toothpaste, the first and second rows in Figure 5). However, in practice, the 3D sketch inputs typically do not have such clean structures. *Curve sketches* contain curves that cannot be clearly organized into loops and represent more frequent user creations through AR/VR devices (e.g., Spaceship and Bishop, the third and fourth rows in Figure 5). *Sparse point clouds* contain scattered points that are hard to organize or connect into curves, making them unsuitable for methods like [Pan et al. 2015], which requires closed loops, or [Yu et al. 2022], which depends on explicit curves (e.g., Fertility, Walrus, and Torus, the fifth to seventh rows in Figure 5).

4.2 Results and Comparisons

In this section, we compare our method with state-of-the-art variational surface modeling and neural implicit reconstruction methods. We then show extensive experiments evaluating runtime performance, the impact of different curve types, accuracy against ground-truth shapes, visualizations of energy distribution, and a discussion on how inputs affect the resulting shapes. We include more extensive



(a) Input (b) [Pan et al. 2015] (c) [Yu et al. 2022] (d) [Huang et al. 2019] (e) [Xu et al. 2023] (f) Ours

Fig. 5. The comparisons on the dataset include three different input types: curve networks [Pan et al. 2015], curve sketches [Yu et al. 2022], and sparse point clouds [Huang et al. 2019]. Note that, among the models produced by [Yu et al. 2022], two models Espresso and Toothpaste rely on their manually constructed initial proxy meshes.

experiments in the appendix and refer readers to our *supplementary video* for a better visualization of our results using a turntable view.

Comparing to variational surface modeling methods. We compare our approach with 4 representative methods that are most relevant to variational surface modeling, including [Pan et al. 2015], [Yu et al. 2022], VIPSS [Huang et al. 2019], and [Xu et al. 2023].

Among these methods, [Pan et al. 2015] is a mesh-based method that requires curve networks as input. With the high-quality curve network inputs, it can generate ideal shapes and recover sharp features in Espresso and Toothpaste (the first and the second row in Fig.5). However, due to the limitation of their curve loop-based algorithm, [Pan et al. 2015] cannot handle either curved sketches or sparse point clouds.

[Yu et al. 2022] uses a proxy mesh as input for initialization and then segments the mesh into different patches, hence it requires the proxy mesh to have the same topology as the expected result. This requirement of the method for a very close proxy model makes it difficult to use. Specifically, such a proxy mesh needs to be generated using another method such as [Huang et al. 2019] and sometimes the mesh thus generated does not work well; when this happens, the proxy mesh has to be constructed manually. Therefore, for the models produced by [Yu et al. 2022] that are included for comparison in Figure 5, we generate their initial proxy mesh automatically using the output of [Huang et al. 2019] as suggested by [Yu et al. 2022], and switch to manual construction whenever this automatic initialization fails. The models that need to rely on manual initialization are Espresso and Toothpaste in Figure 5. Note that [Yu et al.

2022] loses features at the bottom of Bishop and fails to capture fine geometric details at the front of Ship.

[Xu et al. 2023] first computes the normal vector of the point cloud, and then uses Poisson surface reconstruction [Kazhdan et al. 2006] to recover the shape. Similarly, [Huang et al. 2019] models an implicit function that trades off between smoothness and fitting to sparse points. Note that both [Xu et al. 2023] and [Huang et al. 2019] take only points as input and rely on the global consistency of normal vectors to generate smooth surfaces. As a result, they fail to achieve the correct topology when the input point clouds are too sparse (e.g., Espresso and Toothpaste), and they also cannot handle sharp features. Moreover, in the comparison, they introduce undesirable surface artifacts in Fertility and Walrus, such as unwanted bumps. In contrast, our method produces results that are more consistent with the ideal interpolated surfaces. Besides, they are constrained by their algorithmic complexity ($\Omega(n^3)$) and limited to handling no more than 10K points. However, a curve network with fewer than 10K points may not be sufficient to accurately represent complex models, e.g., the Espresso model shown in Figure 5 has 28K points. To ensure proper execution, in this comparison, when the number of points exceeds 10K, we uniformly downsample them to 10K for both [Xu et al. 2023] and [Huang et al. 2019].

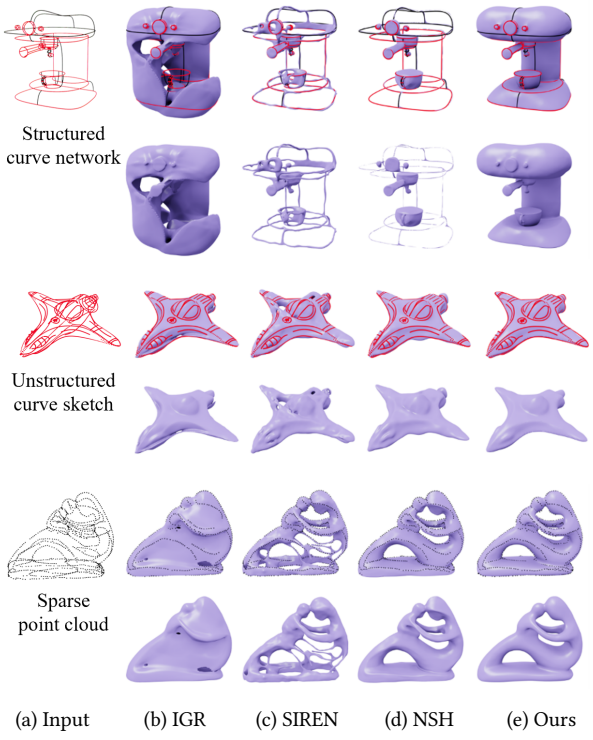


Fig. 6. Comparison with surface reconstruction methods, including IGR [Gropp et al. 2020], SIREN [Sitzmann et al. 2020] and NSH [Wang et al. 2023]. These methods cannot yield visually pleasing shapes, due to the lack of surface modeling objectives away from the input points.

Table 1. We report the time cost per iteration of our method across different zero-level set sample sizes, measured in milliseconds (ms). The mesh is first extracted and sampled, followed by the computation of the thin-plate energy for the samples using the neural network.

Sample Size (Q_{zero})	10K	20K	50K	80K
time [ms]	127.53	132.39	259.33	387.13

Table 2. The run-time performance comparison, corresponding to the curve networks list in Figure 5 (top three rows), presents timing statistics in seconds (s). The time complexity of the method in [Huang et al. 2019] is $O(n^3)$, more than 10K points will time out, necessitating downsampling of the curve networks to 6K (Espresso), 4K (Toothpaste) and 4K (Roadster) vertices. Similarly, the algorithm in [Xu et al. 2023] requires downsampling to 10K (Espresso), 5K (Toothpaste) and 5K (Roadster).

Method	Espresso	Toothpaste	Roadster
[Pan et al. 2015]	24.12	20.73	22.35
[Yu et al. 2022]	527.84	152.83	51.37
[Huang et al. 2019]	4274.32	1321.65	1274.53
[Xu et al. 2023]	1457.32	850.71	328.47
[Gropp et al. 2020]	433.45	446.33	442.12
[Sitzmann et al. 2020]	113.94	124.75	119.41
[Wang et al. 2023]	411.22	408.31	414.97
Ours	1343.71	1344.92	1336.86

Comparing to neural implicit reconstruction methods. Due to the similarity in neural implicit representation, we have compared our method with the recent representative surface reconstruction works, such as IGR [Gropp et al. 2020], SIREN [Sitzmann et al. 2020], and NeuralSingularHessian [Wang et al. 2023]. While these methods handle diverse input types, they cannot yield visually pleasing shapes, due to the lack of surface modeling objectives away from the input points. As shown in Figure 6, our method produces much more plausible shapes than these methods. All methods introduce incorrect topology or undesirable surface artifacts on our test shapes. In contrast, our method reliably constructs the plausible geometry with fine details, correct topology, and sharp features.

Runtime performance. The computational cost primarily arises from evaluating the loss terms (Eq. 10) on three distinct point sets: $\mathcal{P}, \mathcal{Q}, \mathcal{Q}_{zero}$. Among these, the thin-plate energy estimation at $\mathcal{Q}_{zero}$ dominates the time complexity. We fix the size of $\mathcal{P}, \mathcal{Q}$ to 10K, and reports the running time per iteration under different sample sizes for $\mathcal{Q}_{zero}$, as detailed in Table 1. Additionally, we apply MC to extract zero iso-surfaces once for every 100 iterations, so MC cost is amortized across the iterations. The results indicate that our method is still efficient as the number of sample points increases. Thus in all experiments, we set the size of $\mathcal{P}, \mathcal{Q}, \mathcal{Q}_{zero}$ to 10K and the MC resolution to 128^3, resulting in a time cost of 127.53 ms per iteration. The total number of training iterations is set to 10K, which is a conservative estimate; in most cases, convergence is achieved within 5K iterations.

The processing times for the curve networks in Figure 5 ((top three rows) are summarized in Table 2. The curve networks—Espresso,

Toothpaste and Roadster—comprise 25K, 28K and 21K vertices, respectively. The method proposed by [Pan et al. 2015] can be decomposed into several stages: detecting closed loops[Zhuang et al. 2013], constructing the initial surface [Andrews et al. 2011; Zou et al. 2013], and mesh optimization. Due to the decomposition of the curve network into multiple individual surface patches, this method lends itself to high parallelization. Although it offers the fastest running time, it is restricted to curve networks, significantly limiting its applicability. The time complexity of [Huang et al. 2019] is $O(n^3)$, necessitating downsampling of the curve networks to 6K (Espresso), 4K (Toothpaste) and 4K (Roadster) vertices. Similarly, the algorithm in [Xu et al. 2023] is highly sensitive to both the number of points and the shape of the curve network, requiring downsampling to 10K (Espresso), 5K (Toothpaste) and 5K (Roadster). It suggests that both [Huang et al. 2019] and [Xu et al. 2023] lack the capability to handle complex models effectively. The approach in [Yu et al. 2022] includes three stages: initial surface construction, iterative segmentation, and final mesh optimization. To mitigate the initialization effects, [Yu et al. 2022] requires multiple runs, selecting the solution with the lowest energy.

In contrast, [Gropp et al. 2020], [Sitzmann et al. 2020], [Wang et al. 2023], and our method leverage stochastic gradient descent (SGD) on input points, impose no such constraints on the number of input points. Although our method is slower than the other neural implicit reconstruction methods, they fail to generate plausible results for these inputs (Fig 6).

Effects of different curve types. To justify our design choice of using two different types of curves, we show the effects of using the feature curve in Fig. 7. Since this input is a curve network, we automatically detected feature curves by measuring the variation in surface normals at their two endpoints [Pan et al. 2015]. The shapes are plausible when the feature curves are properly marked, and thus, the smoothness energy term does not dominate the shape control around the features. In comparison, when all curves are treated as smooth ones, it leads to an over-smoothed and bulged result (see the first row in Fig. 7).

Evaluation against ground truth. We evaluate our method quantitatively by comparing our results to ground truth surfaces whose curve networks are available. In particular, we use the surface and curvature network data from FlowRep [Gori et al. 2017], which generates descriptive curve networks from 3D shapes. Figure 8 visualizes the results of this experiment, where the color map represents the Hausdorff distance between our result and the ground truth shape. The color map indicates that our results are very close to the ground truth.

Result energy distribution. Figure 9 illustrates the spatial distribution of thin-plate energy across the surface. We colorize the thin-plate energy values with blue and red representing the minimum and maximum values, respectively. The color from blue to red corresponds to increasing thin-plate energy values. Notably, despite implementing a distance attenuation strategy, the thin-plate energy remains uniformly distributed in empty regions.

Topology transition. We conducted a test on our algorithm to see how much input is needed to produce plausible results. Using a torus model, we progressively removed curves until only a single circle remained, then incrementally reintroduced the curves to observe the resulting variations. Results are shown in Figure 10. Given a single circle as input, only a sphere is generated, which is a reasonable result. As more guiding curves are added along the three different paths, we recover the torus shape at different steps, showing the impact of different curves and their combinations.

5 Limitation

Watertight manifold. NeuVAS is based on the assumption of a watertight manifold, which limits our ability to handle models with non-manifold geometry. When dealing with open surface shapes, our method uses a watertight manifold as a prior to generate a closed surface, as shown in Figure 11.

Sharpness. Due to the inherent limitations of the SDF representation, our method, despite achieving a high degree of sharpness, is unable to attain perfectly sharp features. This behavior is illustrated in Figure 20.

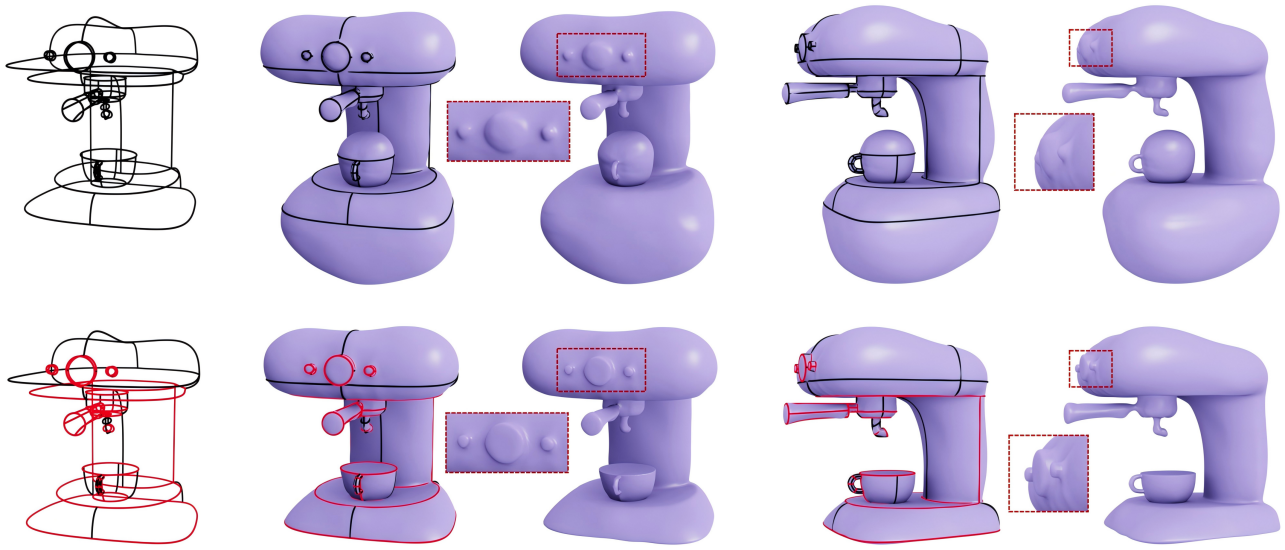
Euclidean distance weight. In extreme cases, the shape may contain two very close, nearly parallel surfaces. These surfaces are disconnected from each other, one is defined by a smooth curve, while the other is defined by a sharp curve. Under such conditions, interactions between the two surfaces can occur, leading to slight irregularities or a reduction in smoothness.

Imperfect spherical or cylindrical shapes. In theory, the TPS energy converges to a state that tends towards local flatness, which excludes spherical or cylindrical shapes theoretically. However, as shown in Figure 12 (a), in the presence of a smooth curve constraint, the TPS energy can still stabilize to a spherical shape whose residuals of two principal curvatures are evenly distributed (i.e. $k1 = k2$). On the other hand, the TPS energy does not stabilize on a cylinder, as shown in Figure 12 (b), making it slightly concave. To this end, the curvature variation energy [Joshi and Séquin 2007; Nealen et al. 2007] permit to converge to cyclide shapes including spherical and cylindrical surfaces. However, being a third-order functional, it suffers from inherent instability and incurs substantial computational overhead. We leave it as future work to address these challenges in our framework.

Noise. Our method demonstrates notable noise resistance, yet exhibits limitations when processing significantly noisy inputs, as illustrated in Figure 13. In our experiments, Gaussian noise was introduced to the curve to evaluate robustness. The algorithm maintains its ability to generate high-quality surfaces at noise levels up to 0.05. However, beyond a threshold of 0.1 noise, the method fails to produce satisfactory results.

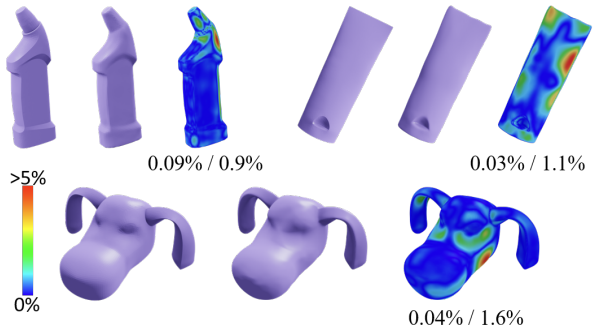
6 Conclusion

We introduce a variational surface modeling framework via representing signed distance function (SDF) encoded shapes by neural networks. Our framework incorporates the Dirichlet and Eikonal conditions to ensure boundary fidelity, while introducing a surface smoothness loss to regulate the shape in empty space between curves. This smoothness energy achieves piecewise smoothness,



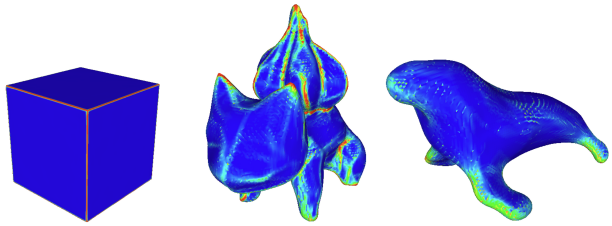
(a) Input (b) Front view with curves overlaid (c) Front view (d) Side view with curves overlaid (e) Side view

Fig. 7. The effects with or without marking the feature curves. Feature curves are highlighted in red. When the feature curves are well marked and represented in our NeuVAS, we generate both the sharp features and the smooth curves in the constructed model, resulting in a more reasonable shape, compared to setting all curves as smooth curves.



(a) GT (b) Ours (c) Error

Fig. 8. Comparing our results to ground truth, the color map shows the Hausdorff Distance from our result to Ground truth. The red region implies a deviation from the ground truth. The deviation is normalized by the bounding-box diagonal of the curve sketch. We provide the median and maximum deviation for each curve sketch.



(a) Cube (b) Bulbasaur (c) Walrus

Fig. 9. Analysis of thin-plate energy distribution on results. Red color means a larger energy value. Although we use the distance attenuation strategy, the thin-plate energy in the empty region is evenly distributed.

leaving regions near curves constrained by boundary conditions to preserve sharp features. Thanks to the neural implicit formulation, our method accepts point clouds as input, and flexibly changes surface topology to find plausible shapes with enhanced robustness. Through comprehensive comparisons with existing state-of-the-art methods, we demonstrate the significant advantages of our approach in constructing complex models.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported by the National Key Research and Development Program of china (2024YFB3309500), the National Natural Science Foundation of China (62402295, 62172257, U23A20312, 62272277), the Natural Science Foundation of Shandong Province (ZR2024QF087, ZR2024ZD12), the Innovation and Technology Commission of the HKSAR Government under the InnoHK initiative and the Hong Kong RGC (Ref. T45-205/21-N), the Joint Fund of the National Natural Science Foundation of China (U22A2033, U24A20219), the National Key R&D Program of China (2022YFB3303200), and the Natural Sciences and Engineering Research Council of Canada (RGPIN-2024-03981).

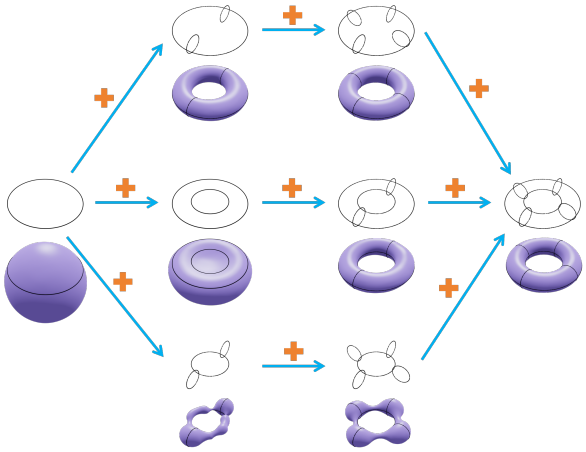
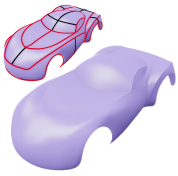


Fig. 10. Topology transition by incrementally adding curves to the input.



(a) [Pan et al. 2015]

(b) Ours

Fig. 11. Limitation of handling open surfaces.

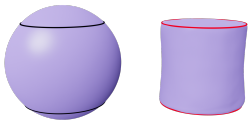
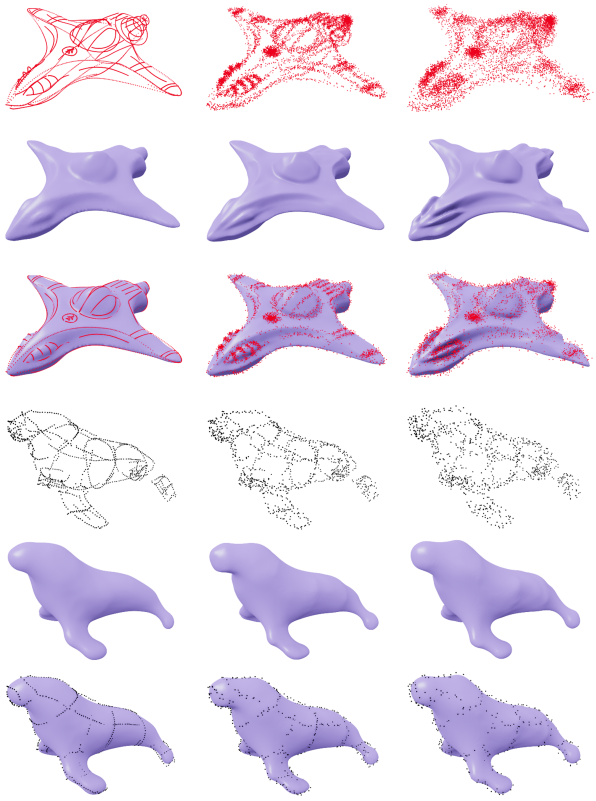


Fig. 12. Imperfect spherical and cylindrical surfaces.



(a) No Noise (b) 0.05 Noise (c) 0.1 Noise

Fig. 13. Introducing Gaussian noise to the input curves. NeuVAS successfully generates quality surfacing results at noise levels up to 0.05. However, our method may not handle data with a noise level exceeding 0.1.

References

Fatemeh Abbasinejad, Pushkar Joshi, and Nina Amenta. 2012. Surface patches from unorganized space curves. In *Proceedings of the twenty-eighth annual symposium on Computational geometry*. 417–418.

James Andrews, Pushkar Joshi, and Nathan Carr. 2011. A linear variational system for modelling from curves. In *Computer Graphics Forum*, Vol. 30. Wiley Online Library, 1850–1861.

Rahul Arora, Rubaiat Habib Kazi, Fraser Anderson, Tovi Grossman, Karan Singh, and George W Fitzmaurice. 2017. Experimental Evaluation of Sketching on Surfaces in VR.. In *CHI*, Vol. 17. 5643–5654.

M. Atzmon and Y. Lipman. 2020. SAL: Sign Agnostic Learning of Shapes From Raw Data. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, Los Alamitos, CA, USA, 2562–2571.

Seok-Hyung Bae, Ravin Balakrishnan, and Karan Singh. 2008. ILoveSketch: as-natural-as-possible sketching system for creating 3d curve models. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*. 151–160.

Mayra Donaji Barrera Machuca, Wolfgang Stuerzlinger, and Paul Asente. 2019. The effect of spatial ability on immersive 3d drawing. In *Proceedings of the 2019 Conference on Creativity and Cognition*. 173–186.

Erhan Batuhan Arisoy, Gunay Orbay, and Levent Burak Kara. 2012. Free form surface skinning of 3d curve clouds for conceptual shape design. *Journal of computing and information science in engineering* 12, 3 (2012), 031005.

Yizhak Ben-Shabat, Chamin Hewa Koneputugodage, and Stephen Gould. 2022. DiGS: Divergence guided shape implicit neural representation for unoriented point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. 2017. A survey of surface reconstruction from point clouds. In *Computer graphics forum*, Vol. 36. Wiley Online Library, 301–329.

Marek Dvorožňák, Saman Sepehri Nejad, Ondřej Jamriška, Alec Jacobson, Ladislav Kavan, and Daniel Sỳkora. 2018. Seamless reconstruction of part-based high-relief models from hand-drawn images. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*. 1–9.

Ricardo Fabbri and Benjamin Kimia. 2010. 3D curve sketch: Flexible curve-based stereo reconstruction and calibration. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 1538–1545.

Ron Goldman. 2005. Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design* 22, 7 (2005), 632–658.

Giorgio Gori, Alla Sheffer, Nicholas Vining, Enrique Rosales, Nathan Carr, and Tao Ju. 2017. FlowRep: Descriptive Curve Networks for Free-Form Design Shapes. *ACM Transaction on Graphics* 36, 4 (2017). https://doi.org/10.1145/3072959.3073639

Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. 2020. Implicit geometric regularization for learning shapes. In *Proceedings of the 37th International Conference on Machine Learning*. 3789–3799.

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. 1992. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques*. 71–78.

Alexander Hornung and Leif Kobbelt. 2006. Robust reconstruction of watertight 3 d models from non-uniformly sampled point clouds without normal information. In *Symposium on geometry processing*, Vol. 41. 50.

Zhiyang Huang, Nathan Carr, and Tao Ju. 2019. Variational implicit point set surfaces. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–13.

Zhiyang Huang, Ming Zou, Nathan Carr, and Tao Ju. 2017. Topology-controlled reconstruction of multi-labelled domains from cross-sections. *ACM Trans. Graph.* 36, 4, Article 76 (July 2017), 12 pages.

Mark W Jones, J Andreas Baerentzen, and Milos Sramek. 2006. 3D distance fields: A survey of techniques and applications. *IEEE Transactions on visualization and Computer Graphics* 12, 4 (2006), 581–599.

Pushkar Joshi and Carlo Séquin. 2007. Energy minimizers for curvature-based surface functionals. *Computer-Aided Design and Applications* 4, 5 (2007), 607–617.

Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. 2006. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, Vol. 7.
Michael Kazhdan and Hugues Hoppe. 2013. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)* 32, 3 (2013), 1–13.
Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization.
Johannes Knoblauch. 1913. *Grundlagen der differentialgeometrie*. BG Teubner. 89–94 pages.
Changjian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. 2017. Bendsketch: Modeling freeform surfaces through 2d sketching. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–14.
Yaron Lipman. 2021. Phase Transitions, Distance Functions, and Implicit Neural Representations. In *Proceedings of the International Conference on Machine Learning (ICML)*.
William E Lorensen and Harvey E Cline. 1998. Marching cubes: A high resolution 3D surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*. 347–353.
Ilya Loshchilov and Frank Hutter. 2022. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*.
Baorui Ma, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. 2021. Neural-Pull: Learning Signed Distance Functions from Point Clouds by Learning to Pull Space onto Surfaces. In *Proceedings of the International Conference on Machine Learning (ICML)*.
Mayra D Barrera Machuca, Paul Asente, Wolfgang Stuerzlinger, Jingwan Lu, and Byungmoon Kim. 2018. Multiplanes: Assisted freehand vr sketching. In *Proceedings of the 2018 ACM Symposium on Spatial User Interaction*. 36–47.
Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2007. Fibermesh: designing freeform surfaces with 3d curves. In *ACM SIGGRAPH 2007 papers*. 41–es.
GüNay Orbay and Levent Burak Kara. 2012. Sketch-based surface design using malleable curve networks. *Computers & Graphics* 36, 8 (2012), 916–929.
Hao Pan, Yang Liu, Alla Sheffer, Nicholas Vining, Chang-Jian Li, and Wenping Wang. 2015. Flow aligned surfacing of curve networks. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–10.
Enrique Rosales, Jafet Rodriguez, and Alla Sheffer. 2019. SurfaceBrush: from virtual reality drawings to manifold surfaces. *arXiv preprint arXiv:1904.12297* (2019).
Bardia Sadri and Karan Singh. 2014. Flow-complex-based shape reconstruction from 3d curves. *ACM Transactions on Graphics (TOG)* 33, 2 (2014), 1–15.
Ryan Schmidt, Azam Khan, Karan Singh, and Gord Kurtenbach. 2009. Analytic drawing of 3D scaffolds. In *ACM SIGGRAPH Asia 2009 papers*. 1–10.
Ruwen Schnabel, Patrick Degener, and Reinhard Klein. 2009. Completion and reconstruction with primitive shapes. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 503–512.
Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. 2020. Implicit Neural Representations with Periodic Activation Functions. In *Advances in Neural Information Processing Systems (NeurIPS)*.
Michael David Spivak. 1975. A comprehensive introduction to differential geometry, Vol. 3. 204.
Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. 2009. Curve skeleton extraction from incomplete point cloud. In *ACM SIGGRAPH 2009 papers*. 1–9.
Anil Usumezbas, Ricardo Fabbri, and Benjamin B Kimia. 2017. The surfacing of multiview 3d drawings via lofting and occlusion reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2980–2989.
Yifan Wang, Lukas Rahmann, and Olga Sorkine-Hornung. 2022. Geometry-consistent neural shape representation with implicit displacement fields. In *The Tenth International Conference on Learning Representations*. OpenReview.
Yifan Wang, Shihao Wu, A. Cengiz Öztireli, and Olga Sorkine-Hornung. 2020. IsoPoints: Optimizing Neural Implicit Surfaces with Hybrid Representations. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), 374–383.
Zixiong Wang, Yunxiao Zhang, Rui Xu, Fan Zhang, Peng-Shuai Wang, Shuangmin Chen, Shiqing Xin, Wenping Wang, and Changhe Tu. 2023. Neural-Singular-Hessian: Implicit Neural Representation of Unoriented Point Clouds by Enforcing Singular Hessian. *ACM Trans. Graph.* 42, 6 (dec 2023).
William Welch and Andrew Witkin. 1994. Free-form shape design using triangulated surfaces. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. 247–256.
Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. 2014. True2Form: 3D curve networks from 2D sketches via selective regularization. *ACM Transactions on Graphics* 33, 4 (2014).
Rui Xu, Zhiyang Dou, Ningna Wang, Shiqing Xin, Shuangmin Chen, Mingyan Jiang, Xiaohu Guo, Wenping Wang, and Changhe Tu. 2023. Globally consistent normal orientation for point clouds by regularizing the winding-number field. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–15.
Emilie Yu, Rahul Arora, J Andreas Baerentzen, Karan Singh, and Adrien Bousseau. 2022. Piecewise-smooth surface fitting onto unstructured 3D sketches. *ACM Transactions on Graphics (TOG)* 41, 4 (2022), 1–16.
Emilie Yu, Rahul Arora, Tibor Stanko, J Andreas Bærentzen, Karan Singh, and Adrien Bousseau. 2021. Cassie: Curve and surface sketching in immersive environments. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.
Yixin Zhuang, Ming Zou, Nathan Carr, and Tao Ju. 2013. A general and efficient method for finding cycles in 3D curve networks. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–10.
Ming Zou, Michelle Holloway, Nathan Carr, and Tao Ju. 2015. Topology-constrained surface reconstruction from cross-sections. *ACM Trans. Graph.* 34, 4, Article 128 (July 2015), 10 pages.
Ming Zou, Tao Ju, and Nathan Carr. 2013. An algorithm for triangulating multiple 3D polygons. In *Computer graphics forum*, Vol. 32. Wiley Online Library, 157–166.

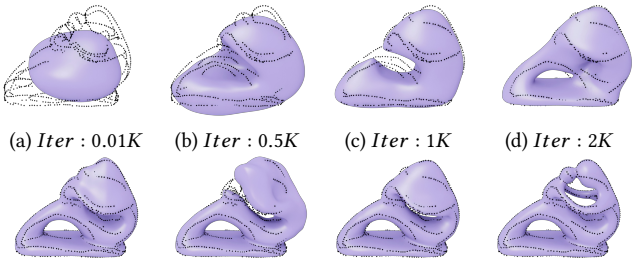
A Additional Experiments

We conducted extensive experiments to evaluate our method and analyze our results in the following aspects: loss functions, curve types, surface sampling numbers, convergence, initialization, result energy distribution, result topology change, and stress tests.

A.1 Topological Change

(a) 5×10^{-7} (b) 5×10^{-6} (c) $5 \times 10^{-5\sim-4}$ (d) 5×10^{-3} (e) 5×10^{-2}

Fig. 14. The effect of thin-plate energy by varying its weight

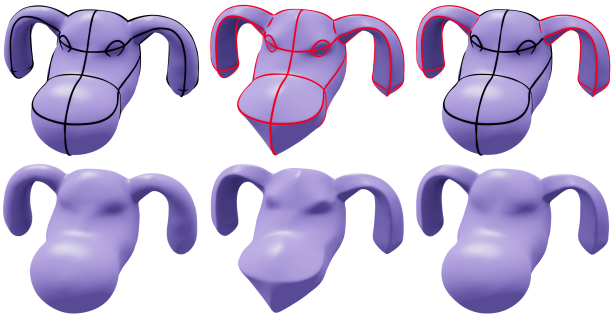


(a) $Iter$: $0.01K$ (b) $Iter$: $0.5K$ (c) $Iter$: $1K$ (d) $Iter$: $2K$
(e) $Iter$: $3K$ (f) $Iter$: $4K$ (g) $Iter$: $5K$ (h) $Iter$: $6K$

Fig. 15. Change of genus over iterations. During the evolution of shape, the surface remains smooth.

We demonstrate the change of topology under different weights of the thin-plate energy. As show in Figure 14, when the weight is between 5×10^{-5} and 5×10^{-4}, we achieve correct topology and obtain an ideal surface. We also demonstrate the change of genus with iteration in Figure 15. As the iteration progresses, the genus continues to increase, and by the time the iteration reaches 6K, it has already converged. Note that during the evolution of shape, the surface remains smooth throughout.

From these results, we can see that, while our method allows flexible change of topology, it is however, difficult to directly control the topology, for example by prescribing genus. To this end, constraints in the line of [Huang et al. 2017; Zou et al. 2015] may potentially allow for more direct control.



(a) Smoothness (b) Feature (c) Mixed

Fig. 16. Experiment on different curve types. (a) All curves are treated as smoothness; (b) all curves are treated as features; (c) specifying only the ear as a feature and incorporating it into our loss term.

A.2 Different Curve Types

We show more results on how different types of curves influence the output. By specifying only the ear as a feature and incorporating it into our loss term, we generate a more reasonable shape that better represents the real geometry (Figure 16(c)). It shows that our feature preservation strategy gives attention to both smoothness and features.

A.3 Effect of Q_{zero} Size.

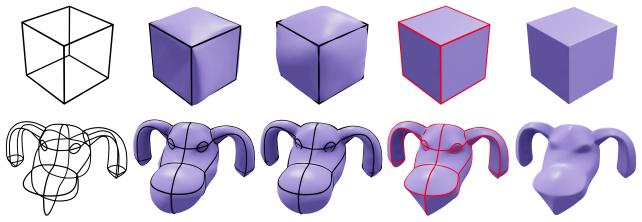
(a) $N : 0.2K$ (b) $N : 1K$ (c) $N : 2K$ (d) $N : 5K$ (e) $N : 10K$

Fig. 17. Ablation study on sampling at the zero-level of the SDF, where N is the number of sampling points. When the number of sampling points exceeds 5K, the improvements become marginal.

We analyze the impact of the number of Q_{zero} sampling points on the construction quality. By adjusting the number of Q_{zero} points from 0.2K to 10K, we observe that increasing the number of sampling points generally leads to more accurate surface construction, as shown in Figure 17, where N denotes the number of sampling points. However, beyond a certain threshold (e.g., 5K), the improvements become marginal, and the computational cost increases significantly. Therefore, a conservative choice of 10K points is adopted for all experiments, as it provides a good trade-off between accuracy and efficiency.

A.4 Effects of Loss Function Terms.

Compared to the original version of IGR [Gropp et al. 2020], we introduce the thin-plate energy loss term to constrain the empty space between the curve networks. As shown in Figure 18, S represents the thin-plate energy loss term, and F denotes the feature curve constraint. When the loss function only incorporates the Dirichlet and Eikonal conditions, it fails to produce a smooth and regular

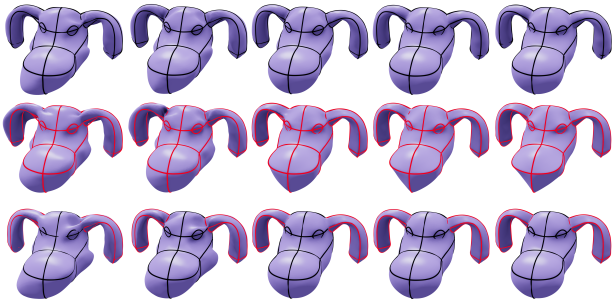


(a) Input (b) $S(\times)\ F(\times)$ (c) $S(\checkmark)\ F(\times)$ (d) $S(\checkmark)\ F(\checkmark)$ (e) $S(\checkmark)\ F(\checkmark)$

Fig. 18. Ablation study on loss functions. S represents the thin-plate energy term, and F represents the feature curve (red) constraint. (b) The loss function includes only the Dirichlet and Eikonal conditions; (c) incorporating the thin-plate energy constraint; (d) a piecewise smooth surface is achieved when all curves are treated as feature curves (red).

shape in the empty space between the curve networks (see Figure 18 (b)). By introducing the thin-plate energy constraint, the generated shape becomes more reasonable (see Figure 18 (c)). Furthermore, we obtain a piecewise smooth surface when all curves are treated as feature curves (red), as shown in Figure 18 (d).

A.5 Shape Evolution over Training Iterations



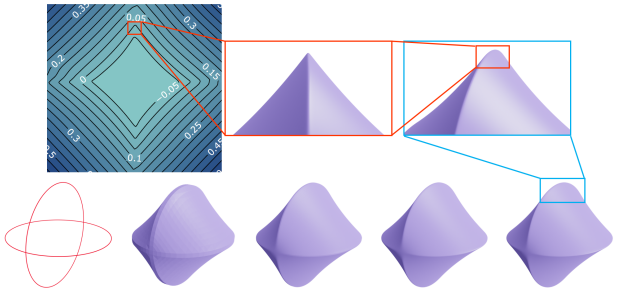
(a) $Iter : 1K$ (b) $Iter : 3K$ (c) $Iter : 5K$ (d) $Iter : 7K$ (e) $Iter : 9K$

Fig. 19. Shape evolution over training iterations. $Iter$ represents the number of iterations. The curve types are set to smooth (top), feature (middle), and mixed (bottom), where only the ear is designated as a feature. The results show that the shape stabilizes around 5K iterations.

We conducted a series of experiments to assess the convergence speed of NeuVAS, as depicted in Figure 19, where $Iter$ denotes the number of iterations. Three different configurations of curve networks were evaluated: all curves treated as smooth, all curves as feature curves, and a mixed configuration, where only a specific part is designated as a feature. The results show that the shape converges after approximately 5K iterations. To ensure reliable convergence, we conservatively set the number of iterations to 10K for all subsequent experiments.

A.6 Sharpness

We employ Marching Cubes at multiple resolutions to visualize the sharpness of the constructed surfaces, as shown in Figure 20. The resolution of the Marching Cubes is progressively increased to



(a) Input (b) $Res : 128^3$ (c) $Res : 256^3$ (d) $Res : 512^3$ (e) $Res : 1024^3$

Fig. 20. Marching Cubes at multiple resolutions are used to demonstrate the sharpness.

clearly illustrate the improvement in sharp feature, accompanied by 2D SDF slices for further clarity. At a resolution of 1024^3, the constructed surface closely approximates the true SDF. Although perfect sharpness is not achieved, the results exhibit a high degree of geometric fidelity and are visually near-sharp.

A.7 Additional datasets

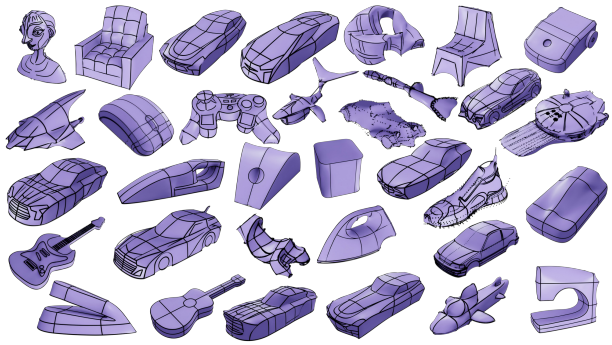


Fig. 21. A gallery of our results on [Yu et al. 2022] dataset.

We present results on the [Yu et al. 2022] dataset in Figure 21 to further demonstrate the generality and robustness of our approach. Since our method has difficulty handling open surfaces, models containing such geometries were excluded from the evaluation.