Self-Supervised Dual Contouring

Ramana Sundararaman, Roman Klokov, Maks Ovsjanikov

Self-Supervised Dual Contouring

Ramana Sundararaman
LIX, Ecole Polytechnique
sundararman@lix.polytechnique.fr

Roman Klokov
LIX, Ecole Polytechnique
klokov@lix.polytechnique.fr

Maks Ovsjanikov
LIX, Ecole Polytechnique
maks@lix.polytechnique.fr

Abstract

Learning-based isosurface extraction methods have recently emerged as a robust and efficient alternative to axiomatic techniques. However, the vast majority of such approaches rely on supervised training with axiomatically computed ground truths, thus potentially inheriting biases and data artefacts of the corresponding axiomatic methods. Steering away from such dependencies, we propose a self-supervised training scheme to the Neural Dual Contouring meshing framework, resulting in our method: Self-Supervised Dual Contouring (SDC). Instead of optimizing predicted mesh vertices with supervised training, we use two novel self-supervised loss functions that encourage the consistency between distances to the generated mesh up to the first order. Meshes reconstructed by SDC surpass existing data-driven methods in capturing intricate details while being more robust to possible irregularities in the input. Furthermore, we use the same self-supervised training objective linking inferred mesh and input SDF, to regularize the training process of Deep Implicit Networks (DINs). We demonstrate that the resulting DINs produce higher-quality implicit functions, ultimately leading to more accurate and detail-preserving surfaces compared to prior baselines for different input modalities. Finally, we demonstrate that our self-supervised losses improve meshing performance in the single-view reconstruction task by enabling joint training of predicted SDF and resulting output mesh. We open-source our code at https://github.com/Sentient07/SDC.

1. Introduction

Surface mesh extraction from implicit functions, often referred to as isosurfacing [30, 38, 46, 64], is a fundamental problem in computer graphics and geometry processing as the quality of reconstructed mesh impacts algorithms used in numerous downstream tasks [50]. Prominent primal axiomatic methods such as Marching Cubes [38, 46], are computationally efficient but fail to reconstruct sharp features. Dual methods [22, 30, 35, 64] achieve sharper reconstructions using Quadratic Error Function (QEF) but their performance heavily relies on the data fidelity.

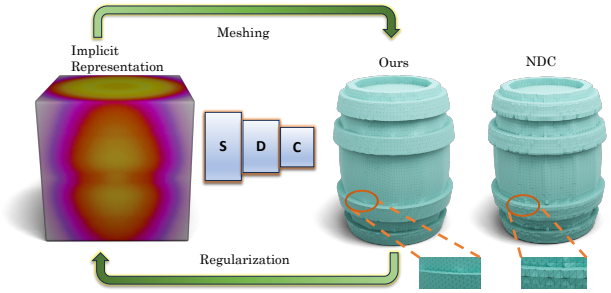


Figure 1. We propose SDC, a data-driven self-supervised isosurfacing method that reconstructs better feature-preserving meshes from learnt implicit functions compared to the baseline NDC [13]. Our framework is extendable as a regularizer for training Deep Implicit Networks (DINs) for improved reconstruction.

Following the ubiquitous success of neural methods in 2D and 3D applications, recent learning-based meshing approaches [11, 13] propose to predict mesh vertices and connectivity from various inputs in a feed-forward manner. When trained on extensive, augmented mesh collections, these methods demonstrate enhanced generalization and yield higher-quality reconstructions. However, these approaches rely either directly on the ground truth meshes [65] or on some intermediate pre-computed proxy meshes, like complex tessellation templates [11] and outputs of axiomatic approaches [13].

Neural Dual Contouring (NDC) [13], the most relevant method to our work, is a promising recent approach that enhances standard Dual Contouring (DC) [30, 64] with a data-driven pipeline. It aims to replicate DC-produced meshes, resulting in a framework capable of generating high-quality,

feature-preserving meshes. However, since vertex estimation of DC follows QEF minimization, it requires precise SDF values at grid points and normals at edge intersections for training. Due to QEF minimization being ill-defined in particular scenarios [30], and susceptible to irregularities in the input SDF, NDC may exhibit a bias towards poor triangulation quality for challenging inputs and intricate details. Furthermore, its reliance on DC meshes for training restricts its use in an end-to-end scenario where both the implicit function and the explicit mesh can be optimized jointly.

To overcome these limitations, we propose Self-Supervised Dual Contouring (SDC), a meshing framework that extends Neural Dual Contouring with two novel geometrically motivated self-supervised losses to learn the mesh vertex positions from SDF inputs while relying on sign changes to extract connectivity. The first self-supervised objective is a distance-based loss which minimizes the difference between the input implicit function values at points close to the surface and distances from the same points to the reconstructed mesh. This ensures that the predicted reconstructed mesh agrees with the input SDF. Secondly, we align the normals corresponding to reconstructed faces to those *estimated* from the input grid of SDFs. Since our training objective relies *only* on the input SDF and is agnostic to any explicit representation of the shape, e.g., based either on surface samples [26, 65] or on mesh vertices [11, 13], we refer to our approach as *Self-Supervised*. Together these two terms produce vertices that best explain the input grid of signed distance function values up to the first order like dual methods, while avoiding shortcomings of the quadratic error function (QEF) minimization [13, 30]. In particular, unlike DC, our method produces vertices that lie within each voxel-cell, and optimization of our self-supervised objectives leads to overall better results for input data with imperfections. As a result, we demonstrate better performance than supervised baselines in obtaining sharper meshes with negligible self-intersections and better generalization to unseen input data.

In addition, leveraging the connection between reconstructed meshes and input SDF grids, we propose a mesh-based *regularization* for training Deep Implicit Networks (DINs) - MLPs which represent zero-level sets of a shape. In a standard setup, training of DINs is completely decoupled from meshing, which is only performed a posteriori. Thus, existing DINs might produce implicit functions that do not correspond to a distance function arising from an extracted mesh. This motivates us to use SDC as a regularization to train DINs. In particular, we enforce the distance function produced by DINs during training to remain as close as possible to the distance function arising from the mesh produced by SDC. This forms the *converse* of our self-supervised meshing losses, but now geared to produce more coherent *implicit functions*. We demonstrate that this regularization can be directly adopted into existing training paradigms for Deep Implicit Networks and it that leads to more plausible reconstructions.

Finally, since SDC training does not require ground truth meshes, it can be seamlessly integrated within end-to-end training paradigms. To highlight this, we consider the task of mesh reconstruction from images where we first construct an implicit representation given an image using an existing approach [56, 62] and then reconstruct a mesh using SDC. The resulting model is trained end-to-end by *jointly* minimizing the standard surface reconstruction loss, our self-supervised loss and the proposed regularization for Deep Implicit Networks. The resulting approach demonstrates compelling surface reconstruction efficacy across multiple object categories from the ShapeNet dataset [9]. In summary, our contributions are as follows:

- We extend Neural Dual Contouring with two novel self-supervised loss functions ensuring consistency between input SDF and distance to output meshes, which is geometrically better motivated than the standard QEF mininmization.
- We introduce a novel regularizer for training Deep Implicit Networks (DINs), by penalizing SDFs that do not correspond to underlying meshes.
- We show that SDC generalizes better than supervised meshing methods, and further demonstrate its utility in feature-preserving mesh reconstruction from images.

2. Related work

Surface reconstruction and meshing tasks have been extensively studied within computer graphics and vision. Below, we review the methods that are most closely related to ours and refer interested readers to several surveys [5, 33, 68] for a more in-depth discussion.

Axiomatic contouring techniques. The process of iso-surface extraction from volumetric data is referred to as contouring. The most widely-used pipeline for surface reconstruction consists of two major steps. First, volumetric data is computed using a signed distance function [14, 29, 32, 52]. In the second step, a mesh is extracted from this representation, using Marching Cubes [46], Dual Contouring [30] or related approaches [17, 53, 74]. This pipeline can produce very high quality meshes, but is sensitive to the quality of input data and is not differentiable and thus does not easily fit within modern learning-based pipelines [39]. Another line of approaches is based on Delaunay triangulations, [7, 8, 37], as well as closely-related constructions [2, 3, 6, 19] which often come with strong theoretical guarantees and are typically geared towards preserving some input point set [1, 6]. Such methods, however, provide little control over the output triangulation. Moreover, classical approaches are typically not differentiable and thus cannot be easily integrated into learning pipelines.

Data-driven mesh reconstruction. To address differentiability issues of classical mesh reconstruction approaches, a number of data-driven techniques have recently been proposed. This includes differentiable variants of contouring methods, such as the Marching Cubes [11, 39], dual Marching Cubes [67], Marching Tetrahedra [66], and Dual Contouring [13]. Our approach can be categorized into this family of differentiable contouring techniques. In addition, surface reconstruction has also been tackled using template-based techniques which fit a template mesh to the input [31, 40, 43], or deform an initial mesh while potentially updating its connectivity [20, 55, 75]. Furthermore, other local approaches propose to fit parameterized surface patches to points [24, 76], or to decompose space into convex sets [12, 16, 47]. Finally, several learning-based methods mesh an input point set [15, 45, 61, 65]. These approaches strongly rely on the given input point cloud and can thus be sensitive to artifacts and noise.

Neural fields. There has been a recent surge in methods for learning implicit functions, referred to as neural fields [78]. This includes pioneering methods for predicting grid occupancy and signed distance functions [10, 49, 56] as well as their many follow-up works [21, 63, 69, 72]. Often, training Deep Implicit Networks introduces inductive bias [60] thereby producing inexplicable surface behaviors. Multiple regularization techniques [4, 23, 41, 42, 44, 60, 69, 72] have been proposed to address that issue. They largely focus on either learning high-frequency signals [41, 72] or designing loss functions based to guide the gradient level-set [23, 42, 60]. While the latter category of methods has convergence guarantees [42], the former helps in producing detail preserving reconstruction. Our work, on the other hand, provides a *novel* regularization by utilizing a meshing framework to ensure that the predicted signed distance field arises from a mesh. Similar to recent works [25, 48, 57, 62, 66, 67, 80] which optimize the mesh by controlling the underlying implicit function, our approach also enables back-propagation from the mesh to the input SDF. However, our approach is non-iterative and produces a mesh in one feed-forward pass.

3. Background

We first begin by providing the notations used across the paper in Section 3.1 and review the Dual Contouring and Neural Dual Contouring algorithms in Section 3.2.

3.1. Notations

We let $f : \mathbb{R}^3 \rightarrow \mathbb{R}$ be an implicit function that determines the signed distance of a query point $x \in \mathbb{R}^3$ from an underlying surface χ. We let f_θ denote the implicit function produced by a neural network parameterized by θ. We use $\mathcal{G}$ to denote an $l \times m \times n$ regular grid in the Euclidean space, $g \in \mathcal{G}$ to be a node in the grid and $e_{ij} \in \mathcal{G}$ to be an edge between adjacent nodes g_i, g_j respectively. We denote $\mathcal{S}_\mathcal{G}$ as the discretization of f on $\mathcal{G}$ *i.e.* SDF evaluated at every grid node. Finally, we let λ denote the latent vector associated with a shape computed using an encoder.

3.2. Dual contouring

The Dual Contouring (DC) [30] framework is designed to produce quadrilateral meshes from input signed distance functions discretized on a grid $\mathcal{S}_\mathcal{G}$. It can be decomposed into two main steps: (1) construction of mesh faces $\mathcal{F}$; (2) computation of mesh vertices $\mathcal{V}$. To construct mesh faces, DC considers adjacent nodes $g_i, g_j \in \mathcal{G}$ such that their signed distance are of opposite signs, $f(g_i) \neq f(g_j)$. Then, for those pairs, a unique quadrilateral face q_k is constructed such that q_k crosses the edge e_{ij}. Vertices of this quadrilateral q_k reside in adjacent grid cells. The faces of the resulting mesh are obtained as a union of all quadrilateral faces $\mathcal{F} = \bigcup_{k \in I} q_k$, where I is a set of indices of grid edges connecting grid vertices of different signs. Final mesh vertex positions within each cell are obtained by minimizing the Quadratic Error Function (QEF). More precisely, given p_e to be the intersection position between edge e_{ij} and the face q_{ij} (along e_{ij}) and n_e to be the gradients of the SDF at positions e_{ij} (or surface normals), vertices are determined for each grid cell c by solving:

$$v_c = \arg\min_x \sum_{e \in I_c} [n_e * (x - p_e)], \tag{1}$$

where I_c is a set of indices of intersected grid edges corresponding to cell c. Recently introduced Neural Dual Contouring [13] attempts to emulate Dual Contouring via a neural network in order to reduce the inference time while achieving optimal reconstruction.

4. Method

We describe our Self-supervised Dual Contouring (SDC) loss functions in Section 4.1, discuss how to use them for regularization while training deep implicit networks in Section 4.2, and apply them for mesh reconstruction from input images in Section 4.3.

4.1. Self-supervised dual contouring (SDC)

We introduce a Self-supervised training scheme for Dual Contouring, and refer to a model trained with it as to SDC. It excludes the dependence on vertex positions obtained via QEF minimization. SDC produces quad meshes from the signed distance function discretized on a grid $\mathcal{S}_\mathcal{G}$ similarly to any Dual Contouring method. We triangulate quadrilaterals by joining the top-left and bottom-right vertex as a convention. $\mathcal{S}_\mathcal{G}$ can either be measured [27] or predicted using a DIN [56] f_θ. SDC consists of two main components constructing faces and vertices. To construct faces,

we follow the Dual Contouring [30] algorithm to determine the connectivity based on signed distance value at grid nodes. A quadrilateral face is constructed in the cells incident on every edge $e_{ij} \in \mathcal{G}$ connecting grid nodes of different signs $\text{sign}(f(g_i)) \neq \text{sign}(f(g_j))$. For vertex prediction, we use a 6-layered 3D Convolutional Neural Network $h_\phi : \mathcal{S}_\mathcal{G} \rightarrow \mathcal{P} \in \mathbb{R}^{(l-1)\times(m-1)\times(n-1)\times 3}$, which takes the grid of signed distance values as input and predicts a single point per each grid cell as the output. Then, we apply a masking to select grid cells, that bear node SDF values of opposite signs. Please refer to Suppl. for more details.

SDC is trained without explicit fitting to the ground truth meshes using two loss functions. Firstly, to ensure formal correspondence of produced surfaces to input SDF grids, we propose a distance-based loss $\mathcal{L}_D$. This objective minimizes the differences between 1) the absolute distance values at the nodes of the discretization grid $f(g_i) \in \mathcal{S}_\mathcal{G}$ which are provided as inputs and 2) the distance $\mathbf{d}^p(g_i, \mathcal{M})$ *measured* from the same point g_i to the *predicted* mesh $\mathcal{M}$. For a smooth or gently undulating surface, this distance provides a reasonable approximation of the underlying geometry [51, 58]. However, in the presence of sharp features, the local geometry changes direction rapidly. Close query points residing on different sides of a sharp feature have similar distances to the mesh but may be projected to very different locations on the actual surface. To address this, we also introduce a normal consistency loss, which aims to align the normals estimated at the nodes of the discretization grid $f(g_i) \in \mathcal{S}_\mathcal{G}$ and normals corresponding to the generated mesh face.

We refer to the losses proposed above as *self-supervised* since the training signal for our pipeline comes from the input alone. We do not use any explicit surface discretization for training and our losses are solely based on the input implicit function values and the generated mesh. In contrast, we refer to methods that predict vertices to fit a specific mesh structure as supervised meshing methods [11, 13]. We provide visual explanations of our loss functions in Figure 2. In summary, our combined self-supervised objective for the vertex prediction network is given by:

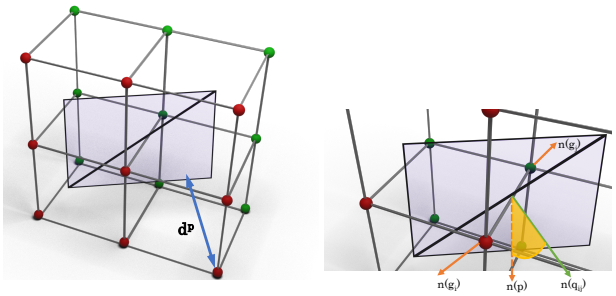
$$\mathcal{L}_{\text{Mesh}} = \mathcal{L}_D + \alpha_1 \mathcal{L}_N, \tag{2}$$

where α_1 is a hyperparameter to weigh $\mathcal{L}_N$.

Distance loss. The objective of our distance loss is to minimize the discrepancy between surfaces produced by SDC and zero-level sets defined by the input SDFs. In order to achieve this, given grid nodes $g_i \in \mathcal{G}$, we measure the distance between the grid nodes and the mesh produced by SDC and penalize its deviation from the input absolute distance values $f(g_i)$. In summary, our distance loss is defined as:

$$\mathcal{L}_D = \sum_{g_i \in \mathcal{G}} ||\text{abs}(f(g_i)) - \mathbf{d}^p(g_i, \mathcal{M})||_2^2, \tag{3}$$

where $\mathbf{d}^p$ is the point-to-face distance, computed between g_i and its projection on $\mathcal{M}$. For computational efficiency, we compute $\mathcal{L}_D$ only at grid nodes that are close to the surface. Please refer to Suppl for details.



(a) (b)

Figure 2. Illustration of our self-supervised losses. (a) Measurements of $\mathbf{d}^p$ between grid nodes and the closest point on a mesh face. For visualization purposes, we only show this estimation at one node. (b) Normal consistency loss is measured as the discrepancy between interpolated normals (orange) and the face normal (green). Red nodes denote negative SDF grid and green denotes positive. The constructed quad is shown in transparent purple.

Normal consistency loss. We introduce a novel normal consistency loss, $\mathcal{L}_N$ which aims to align the level-sets up to the *first order*. In particular, given the grid of scalar SDFs, we compute normals as gradients $n(g_i) := f'(g_i)/||f'(g_i)||_2$ in grid nodes g_i using Five-point stencil, a commonly used finite difference approach for solving partial differential equations on regular grids [70]. More details on gradient estimation is provided in the Suppl. Given an edge $e_{ij} \in \mathcal{G}$ connecting nodes g_i, g_j of the opposite signs, we first estimate a point $p \in e_{ij}$ along the edge where $f(p) = 0$. The point p can be represented in terms of g_i and g_j as $p = (1-t)g_i + tg_j$. where t is a parameter that determines the position of p along the edge e_{ij}. Solving for $p = 0$, the parameter t can be expressed in closed form as:

$$t = \frac{f(g_i)}{f(g_i) - f(g_j)}. \tag{4}$$

Similarly, we can estimate the normal at the point p by linear interpolation. Using the previously computed parameter t, the normal at p can be represented as:

$$n(p) = \text{sign}(f(g_i))(1-t)n(g_i) + \text{sign}(f(g_j))tn(g_j). \tag{5}$$

This expression combines the known gradients at the neighboring nodes, weighted by the parameter t, to estimate the gradient at a point p. Since the inward normal is oriented

opposite to the outward normal, we multiply by the sign for consistency. As our mesh is constructed following Dual Contouring [30], we denote the quad-face constructed dual to the edge e_{ij} as q_{ij} and its normal as $n(q_{ij})$. With the aim of aligning these two normals, our normal consistency loss can be written as:

$$\mathcal{L}_N = 1 - \frac{n(q_{ij}) \cdot n(p)}{\|n(q_{ij})\| \|n(p)\|}. \quad (6)$$

Training augmentation. At training time, we add synthetic noise which is zero-mean Gaussian noise with a standard deviation equivalent to a third grid edge length e_{ij} to the input SDF. Since SDC constructs mesh faces based on Dual Contouring, a noise that induces a sign change to the initial SDF will also lead to a topological change. More precisely, augmentation will lead to inclusion of vertices (and faces) in cells whose nodes differ in sign. Similarly, faces are removed dual to cells whose signs agree after augmentation. More training details are provided in the Suppl.

4.2. Deep implicit network regularizer

Deep Implicit Networks (DINs), commonly represented by Multi-Layer Perceptrons (MLP), are trained to acquire an implicit representation of a shape and are primarily utilized for various inference-based surface reconstruction tasks. In a traditional setup, DINs are trained by direct supervision of the SDF values at query points sampled close to the surface. Such a training scheme, as also observed by previous works [23, 42, 60], is known to produce inexplicable behavior, such as ambiguous level sets. To address this, we propose a novel mesh-based training regularization for DINs to ensure that the Distance Field (DF) produced by DIN agrees well with the resulting extracted mesh. Our regularization minimizes the discrepancy between the predicted DF (absolute value of SDF) and the distance computed from the mesh corresponding to the SDF extracted using SDC.

We assume that we are endowed with a learnable DIN $f_\theta(\lambda, p) : p \in \mathbb{R}^3 \rightarrow s_p \in \mathbb{R}$, represented as an MLP, whose weights θ encode the implicit surface of a shape conditioned by a latent vector λ, p is a point in space such that $f_\theta(\lambda, p) = s_p$ is its signed distance from the zero-level set. Then, for a given $\mathcal{S}_G$ - a discretization of f_θ on a regular grid and $\mathcal{M}^*(\mathcal{S}_\mathcal{G})$ - a mesh that is produced following any iso-surfacing algorithm, it is not necessarily true that $\text{abs}(f_\theta(p, \lambda)) = \mathbf{d}^p(p, \mathcal{M}^*(\mathcal{S}_\mathcal{G}))$, where $\mathbf{d}^p$ is the point-to-face distance. While this disparity is viewed as an inherent drawback of primal iso-surfacing methods [54], for dual methods, this disparity can, in principle, be minimized due to their ability to preserve sharp details in the reconstructed meshes. This motivates us to use our SDC for iso-surface extraction and regularize the training of DINs by minimizing the aforementioned disparity. Assuming we have access to a dataset consisting of N shapes and a pre-trained SDC with frozen weights, our regularization objective can be summarized as follows:

$$\mathcal{L}_{SDR} = \sum_{j=1}^{N} \sum_{g_i \in \mathcal{G}} || \text{abs}(f_\theta(\lambda_j, g_i) - \mathbf{d}^p(g_i, \mathcal{M}) ||_2^2. \quad (7)$$

This regularization penalizes the discrepancy between the DF at grid nodes predicted by the DIN $f_\theta(g_i)$ and the distance $\mathbf{d}^p$ computed between the grid nodes and the mesh $\mathcal{M} = \text{SDC}(\mathcal{S}_\mathcal{G})$. To obtain $\mathcal{M}$, we first evaluate f_θ over a discrete grid $\mathcal{G}$ and obtain $\mathcal{M} = \text{SDC}(\mathcal{S}_\mathcal{G})$. Note that the above regularization is similar to the signed distance loss defined in Eq. (3), but with an important difference. Here we do not use ground-truth SDF values but instead, regularize the *predicted* signed distance. This regularization is applied alongside the main objective function to reconstruct the implicit surface commonly used to train DINs [18, 56] as follows:

$$\mathcal{L}_{\text{SDF}} = \sum_{j=1}^{N} \sum_{i}^{K} || \left(f_\theta\left(\lambda_j, x_i\right) - s_i\right| |_2^2 + \frac{1}{\sigma^2} \left\|\lambda_j\right\|_2^2, \quad (8)$$

where K is the number of points with annotated SDF values per shape j, s_i is the ground-truth SDF value, and σ is a parameter used to promote compactness in latent space [56].

Combining the standard training loss for DINs [56] alongside our regularization weighted with a scalar α_2, our learning objective is formulated as follows:

$$\mathcal{L}_{\text{train}} = \mathcal{L}_{\text{SDF}} + \alpha_2 \mathcal{L}_{SDR}. \quad (9)$$

At the inference time, we reconstruct surfaces from point clouds by first recovering the optimal latent vectors (See Eq. (4) in [18]). Then, we predict a grid of SDF values $\mathcal{S}_\mathcal{G}$ and extract the iso-surface using our SDC.

4.3. Joint learning of implicit surface and meshing

Differently from existing data-driven meshing methods, SDC relies neither on explicit mesh data [61, 65] nor on axiomatically produced proxy meshes [11, 13]. This means that SDC can be used to optimize both SDF and reconstructed meshes jointly in an end-to-end manner. To demonstrate this we consider the task of predicting surface meshes from images. We first model the SDF representation of a shape given an image and then predict the mesh using SDC. In terms of the architectures, we follow MeshSDF [62]: we produce latent vectors λ from input images with a ResNet-18 [28] encoder and learn an implicit shape representation of shapes using DeepSDF [56] conditioned on the aforementioned latent vectors. Instead of performing test-time optimization like MeshSDF, we predict SDF values on a regular grid $f_\theta(\mathcal{G})$ and use SDC to reconstruct a mesh $\mathcal{M} = \text{SDC}(\mathcal{S}_\mathcal{G})$ by a simple forward pass of our SDC network. Initially, we train the networks (1) DeepSDF (with

an image encoder) and (2) SDC independently, then jointly fine-tune them for 200 epochs. This fine-tuning uses an image as input and the mesh $\mathcal{M}$ from SDC as output, while minimizing the combined objective terms for surface reconstruction and meshing as follows:

$$\mathcal{L}_{\text{SVR}} = \mathcal{L}_{\text{SDF}} + \alpha_3 \mathcal{L}_{SDR} + \alpha_4 \mathcal{L}_{\text{D}}, \tag{10}$$

where $\mathcal{L}_{\text{SDF}}$ and $\mathcal{L}_{\text{SDR}}$ (defined in Eq. (8) and Eq. (7) respectively) are used to reconstruct and regularize the implicit representation, while $\mathcal{L}_D$, defined in Eq. (3), is used to train SDC. Since the evaluated SDF might be noisy during training, we observed that using $\mathcal{L}_N$ results in instability.

5. Experiments, results and discussion

We show the effectiveness of SDC across three tasks, namely, implicit function meshing, DIN training regularization and joint SDF and mesh prediction from images.

Evaluation Metrics. We use Chamfer Distance (CD), Normal Consistency (NC), Self-Intersection (SI), Edge Chamfer Distance (ECD), 3D-IoU, Precise Level-Set Discrepancy (LSD-P), and Approximate Level-Set Discrepancy (LSD-A) as metrics to compare all approaches. For measuring Level-Set discrepancy, we use two terms, Precise (LSD-P) and Approximate (LSD-A). The former is used in case where shapes are water-tight and analytical SDF can be measured. In such cases, we measure the discrepancy between the ground truth analytical SDF and the distance function measured from the generated mesh. LSD-A is used for non-watertight shapes and is measured by sampling points on the ground truth mesh and measuring its distance from the generated mesh's faces. Self-intersection is reported as the total number of intersecting triangles per mesh averaged across the test set. For the single-view reconstruction task, we also measure the Structural Similarity Index (SSIM) between the rendering of the reconstructed mesh and the ground truth mesh. We measure 3D-IoU for experiments where the generated mesh could topologically differ from the ground truth. CD, ECD, LSD are scaled by 10^3 while NC, SSIM 3D IoU are in %. We provide more details on all the metrics in the Suppl.

5.1. Meshing analytical implicit functions

We first consider the surface mesh reconstruction task given the *ground truth* SDFs of shapes discretized on a regular grid. For this task, we train SDC on 3000 shapes from the 1^{st} split of the ABC dataset [36] for 100 epochs. We scale each shape to fit a unit sphere and estimate the SDF on a regular grid of dimension 64^3 using SDFGen library [1]. We evaluate all methods on 300 shapes from the test set of the ABC dataset. Additionally, we evaluate on 300 shapes from the Thingi10K [81] dataset to demonstrate generalization to unseen data domains. We compare SDC against four baselines including two axiomatic baselines, namely the improved Marching Cubes (MC) [38] and the standard Dual Contouring (DC) [30] where positions of each vertex are estimated by solving the Quadratic Error Function (QEF), and three of the recent Neural Meshing techniques, Neural Marching Cubes (NMC) [11], Neural Dual Contouring [13] and FlexiCubes [67] respectively. For fairness, we compare to variants of comparable network capacity and for FlexiCubes, we compare without test-time optimization.

We summarize our quantitative results in Table 1. Our self-supervised method SDC shows better performance and generalization to unseen data domains compared to the data-driven baselines across all metrics. While Marching

[1] https://github.com/christopherbatty/SDFGen

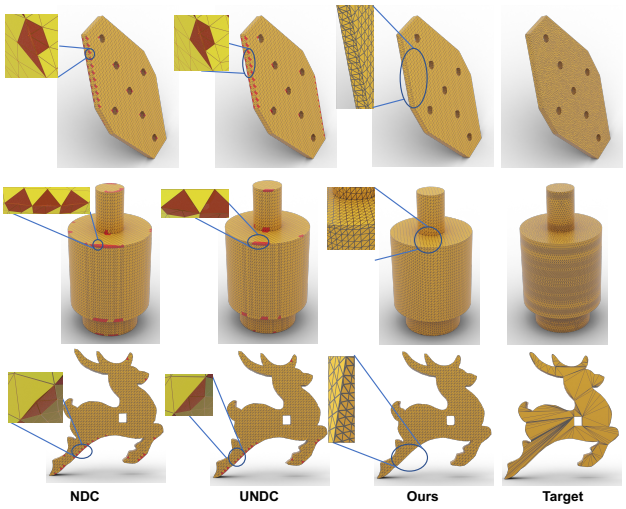


Figure 3. We show three qualitative mesh reconstruction examples from the ABC dataset (Row 1,2) and the Thingi10k dataset (Row 3). Self-intersecting faces are highlighted in red. QEF-based learning methods NDC and UNDC show significantly higher self-intersecting faces along sharp edges while ours does not.

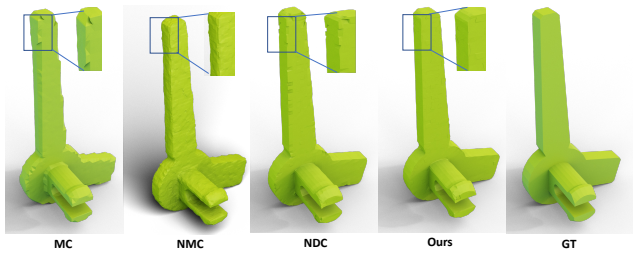


Figure 4. Qualitative examples of meshing SDF produced by NGLOD [71]. MC, NMC, and NDC produce noisy meshes while SDC produces a smoother yet feature-preserving reconstruction.

Dataset	Input SDF	Method	CD (↓)	NC (↑)	#SI (↓)	ECD (↓)	LSD-P (↓)
ABC	Analytical	MC [38]	4.7	92.1	**0**	4.8	16.1
		NMC [11]	3.7	*94.7*	34.5	3.6	12.1
		DC [30]	*3.5*	93.5	96.3	*3.4*	*11.6*
		NDC [13]	3.6	94.3	43.1	3.5	11.9
		FlexiCubes [67]	4.5	92.4	0	4.8	15.8
		Ours	**3.3**	**94.9**	*9.7*	**3.2**	**11.3**
Thingi10K	Analytical	MC [38]	5.1	61.4	**0**	13.3	15.3
		NMC [11]	4.1	*66.0*	32.4	11.9	12.2
		DC [30]	*4.0*	63.8	105.6	11.5	*11.8*
		NDC [13]	*4.0*	64.9	70.4	*11.3*	12.0
		FlexiCubes [67]	5.2	55.3	0	13.6	15.4
		Ours	**3.6**	**68.0**	*15.7*	**10.9**	**11.6**
Thingi10K	Predicted [71]	MC [38]	5.8	56.7	**0**	14.2	16.5
		NMC [11]	*4.7*	*63.2*	30.1	*12.6*	*12.9*
		DC [30]	6.2	54.0	230.4	14.8	15.0
		NDC [13]	*4.7*	62.8	42.6	12.8	13.1
		Ours	**4.2**	**65.7**	*15.9*	**12.2**	**12.4**

Table 1. Quantitative mesh reconstruction results on the ABC and the Thingi10k datasets. The best scores are highlighted in bold and the second-best scores are in italics. The first two row-blocks correspond to input SDF which was computed analytically while in the last row-block, we mesh the SDF predicted by NGLOD [71]

.

Cubes [38] and FlexiCubes [67] do not produce any self-intersecting faces, the vertex placement is restricted for these methods, leading to inferior results with respect to other reconstruction metrics. In the qualitative examples visualized in Figure 3, we highlight an inherent issue with NDC and UNDC - they produce a considerable number of self-intersections along sharp edges. We reason this to be an undesirable trait that is inherited from the standard Dual Contouring as previously elaborated. SDC, on the other hand, bears a minimal number of self-intersections, as our training objective for mesh prediction is geometrically well-motivated. Additionally, we report quantitative and qualitative results for 128^3 grid resolution in the Suppl.

5.2. Meshing predicted implicit functions

Differently from the previous section, where the SDF is computed analytically, here, we use an open-source implementation (with suggested hyperparameters) of a recent deep implicit network NGLOD [71] to predict SDF values on a regular grid. We evaluate the meshing efficacy of SDC over a set of 300 shapes from the Thingi10k [81] dataset. To avoid possible biases, we chose these 300 shapes to be different from the ones used in the previous section. Our quantitative results are summarized in Table 1. Consistent with our observation from the previous section, SDC outperforms all baselines. We provide a qualitative example of meshing a noisy SDF in Figure 4. To generate the noisy SDF, we prematurely terminate NGLOD [71] fitting to the shape and evaluate different Neural Meshing methods. While all baselines produce a noisy reconstruction, SDC recovers a smoother surface while preserving sharp edges, thanks to our regularization and self-supervised losses. This suggests that our self-supervised training objectives and data augmentation are capable of generalizing to noisy imperfect predicted data *without explicit training on this data*.

5.3. Regularizing implicit surface learning

As illustrated in the previous section, SDC successfully produces sharp meshes from both analytical and parameterized implicit functions. We now employ SDC for regularizing DIN training, focusing on the task of surface reconstruction using the ShapeNet dataset [9]. In particular, we evaluate over 4 categories of objects, namely, cars, planes, tables, and cabinets with 1000 shapes per category as our training set, and set aside a separate set of 200 unseen shapes for evaluation. We compare our regularization against three possible baselines. Firstly, we train Curriculum DeepSDF (CSDF) [18] over each category separately. Secondly, we re-train the same network by enforcing the Eikonal constraint, *i.e*. enforcing a unit norm for SDF's gradient in a setup similar to Implicit Geometric Regularization [23]. Thirdly, we use the supervised Dual Contouring baseline NDC [13] as the mesh-based regularizer. More specifically, we replace SDC with NDC in computing the regularization introduced in Eq. (7). For each shape, we sample 400,000 points in the shape volume, aggressively near the surface following [56] to supervise the SDF prediction. For fairness, we use the same sampled points for all methods we compare.

We summarize our quantitative results in Table 2. We observe that regularizing the network using SDC produces a noticeable improvement in reconstruction across all metrics in comparison to baselines. Moreover, we observe a poorer reconstruction when using NDC [13] for regularization. We argue that this is due to their inability to handle noisy predicted SDF values as inputs as this approach learns to emulate solution to QEF (c.f. Eq. (1)), which is ill-defined for imperfect implicit functions. Our self-supervised loss functions do not rely on the ill-defined vertices produced by QEF minimization and align produced meshes to the input SDF grids, resulting in more plausible surface predictions. We believe this difference to be the key reason behind SDC's efficacy as a regularizer for training DINs. We also show three qualitative results in Figure 5. In some examples, we also observe topological differences although our regularization does not explicitly penalize it. Since the loss function which we minimize is highly non-convex, we believe that our regularization has aided in discovering more "plausible" latent space which could lead to better quality of implicit surfaces. We report additional qualitative and quantitative results in Suppl demonstrating the regularizer's generalization capabilities with other DINs.

5.4. Single view reconstruction

In this section, we consider the task of reconstructing surface meshes from images. We use 4 categories from the ShapeNet dataset [9], namely planes, chairs, rifles, and tables. We train and evaluate our approach and baseline (using the official codebase) on the same training and evalua-

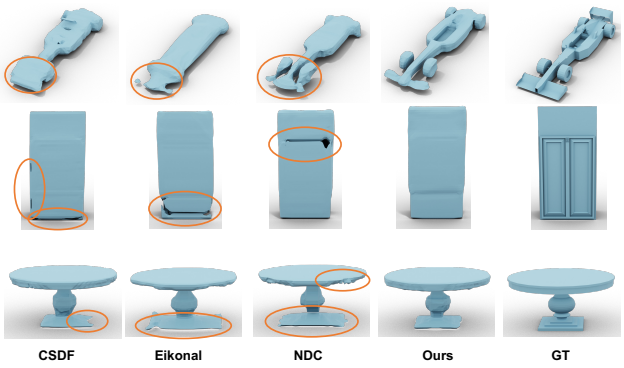


Figure 5. Qualitative examples comparing surface reconstruction from the ShapeNet [9] dataset. The columns compare different methods, whose training objectives differ. Our mesh-based regularization produces reasonable results compared to baselines.

Method	CD (↓)	NC (↑)	3D IoU (↑)	LSD-A (↓)
CSDF [18]	3.0	78.1	83.9	3.1
+ Eikonal [23]	3.2	78.0	83.0	3.2
+ NDC [13]	*2.9*	*78.4*	*84.8*	*2.9*
+ SDC	**2.6**	**78.9**	**86.0**	**2.7**

Table 2. Comparison of surface reconstruction accuracy across object categories from the ShapeNet dataset [9]. Methods are trained with different regularization (see text) while evaluated alike.

tion split for a fair comparison. We compare against three baselines, namely, MeshSDF [62], DISN [79], and a variant of our approach referred to as WoBW. More specifically, WoBW meshes the implicit function produced by pretrained SV-DIN using SDC, without the joint fine-tuning. For MeshSDF [62] and DISN [79] we use the official codebase for re-training and evaluation.

Quantitative results averaged across 4 object categories are summarized in Table 3. Notably, our SDC outperforms MeshSDF [62] without additional test-time optimization fitting a mesh to the rendering. In addition, our end-to-end model consistently outperforms the WoBW baseline, implying the efficacy of our mesh-based regularization and end-to-end training. We also show three qualitative examples in Figure 6. Our SDC demonstrates an improved ability to reconstruct sharp features in the predicted meshes, resulting in surfaces more faithful to the ground truth. In the third row, we highlight an example showing significant improvements in the quality of predicted geometry. It shows that our method is capable of better thin surface reconstructions for the same inputs.

6. Conclusion and future work

We introduced SDC, a Self-Supervised training approach for Neural Dual Contouring. Differently from previous

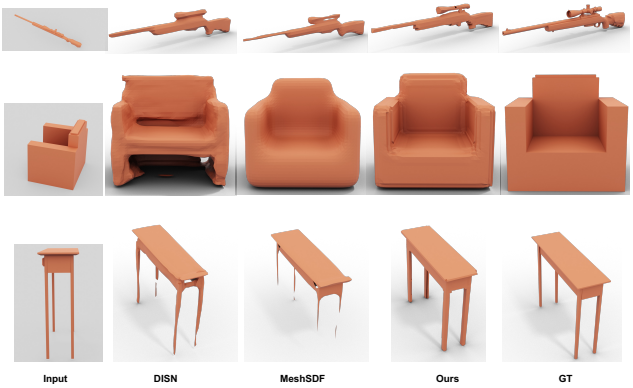


Figure 6. Qualitative comparison of Single View Reconstruction (SVR) accuracy among three objects from ShapeNet [9] dataset. Input denotes the input view and GT denotes ground-truth mesh. SDC produces more sharper and plausible surfaces in comparison to baselines.

Method	CD (↓)	NC (↑)	SSIM (↑)	LSD-A (↓)
DISN [79]	13.0	70.2	82.0	10.5
MeshSDF [62]	11.3	71.4	84.1	9.1
WoBW	*9.9*	*72.0*	*86.4*	*8.1*
Ours	**9.1**	**72.8**	**88.5**	**7.7**

Table 3. Quantitative results for single-view reconstruction across different object categories from ShapeNet [9] dataset.

work, we do not fit the generated mesh to an axiomatically produced proxy mesh but instead use geometrically motivated self-supervised losses which only depends on the input SDF. SDC shows consistent improvements over baselines when applied to end-to-end surface reconstruction and meshing tasks. In addition, our work forges a link between signed distance fields and meshes and shows how the former can be regularized by guiding the network to produce distance fields better corresponding to resulting meshes. Our meshing and regularization are applied on a regular grid, which limits the resolution of a reconstructed mesh. It would be interesting to explore adaptive grids within the learning framework. Finally, another related and interesting scope for future work is the exploration of strict manifoldness conditions [64] and theoretical guarantees of self-intersection free surfaces.

Acknowledgements This work was granted access to the HPC resources of IDRIS under the allocation 2023-AD011013104R2 made by GENCI. Parts of this work were supported by the ERC Starting Grant 758800 (EXPROTEA), ERC Consolidator Grant 101087347 (VEGA), ANR AI Chair AIGRETTE, and gifts from Ansys and Adobe Research.

Self-Supervised Dual Contouring

Supplementary Material

7. Introduction

This document serves as the supplemental material to our main work "Self Supervised Dual Contouring". We perform ablation studies on potential alternative reconstruction losses in Section 8. Then, we provide additional implementation details pertaining to our three experimental settings in Section 9. We elaborate on various evaluation metrics used in the main paper in Section 10. Finally, we provide additional quantitative evaluation and qualitative results in Section 11. Our entire code will be released upon publication.

8. Ablation Studies

We perform Ablation studies over our self-supervised loss function and sampling strategy for applying our proposed mesh-based regularization.

8.1. Reconstruction losses

In this section, we compare alternative loss functions for meshing a given implicit function. To recall, we refer to our loss as Self-Supervised as our learning framework does not rely on a reference object and our two loss functions are computed purely in terms of input grid of SDF. Herewith, we compare loss functions which uses explicit supervision w.r.t. a reference mesh to our self-supervised loss function. *I.e.*, instead of aligning the generated surface to best fit the input SDF, we compare our SDC with loss functions that try best aligns the generated surface to some discrete set of surface samples. These discrete surface samples are points sampled on the mesh from ABC [36] dataset, which we used for training SDC as stated in our main paper. We remove our SDC losses and replace them with different loss functions as elaborated below:

1. **Random Sampling**: We randomly sample points on generated surface and ground truth surface and minimize the Chamfer's Distance between them.
2. **Area Sampling**: We use an area based sampling where we sample the generated surface proportional to the area of triangles. Then, Chamfer's Distance is minimized between the aforementioned samples and ground-truth surface samples.
3. **Vert CD**: We apply Chamfer Distance, between generated vertex (produced by SDC) and mean coordinate of point samples within the same voxel as the generated vertex. This loss function is local with locality defined by voxel cell.
4. **Second-Order CD**: Minimizing the distance between two surfaces has been well-studied in the context of shape registration with theoretical guarantees [51, 58, 59]. To that end, we consider quadratic approximation of point-point distance proposed in [51] as our baseline. In particular, [51] uses local curvature information of the surface to incorporate second order information into the function which measures the distance between query point and reference surface. In our case, we consider the query point to be the mesh vertex predicted by our network h_ϕ and the reference surface to be the ground truth mesh from our training dataset, ABC [36]. Since we explicitly use the mesh, we consider this to be supervised baseline. This supervised training objective is given as follows:

$$\mathcal{L} = \hat{\delta}_1 \left(\vec{e}_1 \cdot (\mathbf{x} - \mathbf{y})\right)^2 + \hat{\delta}_2 \left(\vec{e}_2 \cdot (\mathbf{x} - \mathbf{y})\right)^2 + \left(\vec{n} \cdot (\mathbf{x} - \mathbf{y})\right)^2,$$

   where $\mathbf{x}$ denotes the query point, $\mathbf{y}$ denotes the closest point on the surface where the surface normal is given by $\vec{n}$ and the direction of principal curvature is given by $\vec{e_1}, \vec{e_2}$. Finally, δ_1, δ_2 denote the magnitude of principal curvatures at $\mathbf{y}$.
5. **W/o NC**: We do not use the normal consistency loss and only use $\mathcal{L}_D$ defined in Eqn.3 of the main paper.
6. **Ours**: Denotes the loss function which we report in the paper.

Type	Method	CD (↓)	NC (↑)	SI (↓)	ECD (↓)	LSD-P (↓)
Explicit Supervision	Random Sampling	3.90	90.20	77.60	3.85	13.1
	Area Sampling	3.72	91.60	50.44	3.70	12.6
	Vert CD	3.52	91.70	72.70	3.54	11.9
	2^{nd} Order CD	3.47	91.00	14.55	3.49	11.8
Implicit Supervision	W/o NC	3.40	93.80	11.64	3.40	**11.3**
	Ours	**3.30**	**94.90**	**9.67**	**3.20**	**11.3**

Table 4. Ablation study on different unsupervised losses for the task of meshing an implicit function.

Our quantitative results are summarized in Table 4. We observe a noticeable improvement in performance when using our self-supervised loss function compared to supervision with surface sampling. As discussed in related works [73, 77] sampling surfaces with discrete points could lead to attraction of points to a single source (or sink) at regions of uneven sampling density. This could potentially explain the higher self-intersection. Also, more importantly, Self-Supervised loss (Ours) shows a significant improvement over supervised baseline (Second-Order). The reported experiments were performed on the test-set of the ABC dataset defined in the main paper.

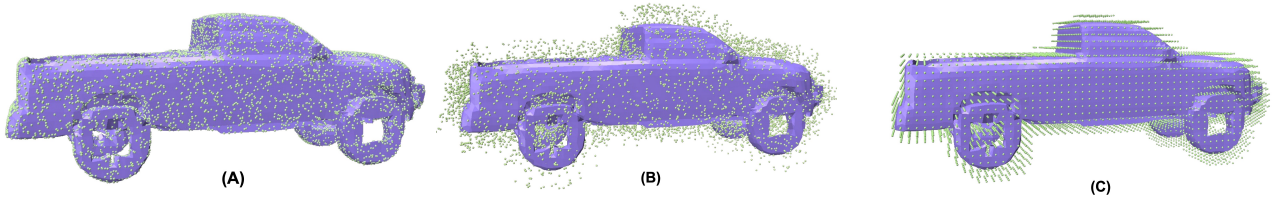


Figure 7. Depicting point samples (in green) at which our signed distance based regularization is applied. The implicit function is meshed using SDC and rendered in purple. (A) denotes surface sampling, (B) Denotes irregular near-surface sampling and (C) is regular near-surface sampling.

Sampling Type	CD ↓ (x10^3)	NC ↑ (%)	IoU ↑ (%)	LSD-A ↓ (x10^3)	Time ↓ (m-sec)
Surface	3.6	74.2	84.9	2.8	2.3
Volume	2.8	77.5	86.9	2.8	2.3
Reg Grid	**2.6**	**79.0**	**87.3**	**2.7**	**2.1**

Table 5. Comparing quantitative reconstruction results and timing between different sampling strategies for applying our mesh-based regularization.

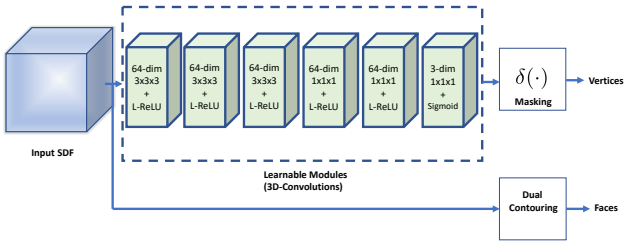


Figure 8. Detailed depiction of SDC framework. Starting from input SDF defined on a regular grid, we predict the vertices of the mesh while constructing the faces following the Dual Contouring [30] technique.

8.2. Sampling for regularization

In this section, we ablate various sampling strategies which could be used to establish the points for which $\mathcal{L}_{\text{SDR}}$ (c.f Eqn.7, main paper) can be computed. In particular, we compare between three types of sampling points in space for which SDF prediction is regularized w.r.t. the mesh produced by SDC. Firstly, we compare between points that are defined on a regular grid, close to the surface of the mesh. Secondly, we consider points sampled uniformly on the surface of the mesh produced by SDC. Thirdly, we add small random displacement along the normal vector such that the points on the mesh are close to but not necessarily on the surface of the mesh. The three sampling strategies mentioned above are visualized in Figure 7. We sample 20,000 points for each strategy. We compare the reconstruction accuracy of the implicit surface while using the proposed regularization along with different sampling strategies. The quantitative results are summarized in Table 5. We observe that a regular sampling along the grid shows overall better performance compared to random sampling. We also report the average time for performing a single forward pass using the aforementioned regularization. We observe that using a regular sampling strategy is less time-consuming in comparison to other sampling strategies. This is because, we can *re-use* the SDF values computed at those grid points as SDC requires SDF values at grid points to reconstruct a mesh. This is unlike the latter two cases where another forward pass through CSDF [18] is required to determine the SDF values. All experiments were performed on Ampere-A100 GPUs for fairness in comparison.

9. Additional implementation details

We first provide additional details on estimating normals and then provide implementational details for the three experiments performed in our main paper.

9.1. Meshing implicit surfaces

Normal Estimation. We estimate the normals from grid of SDF using Finite-Difference gradient estimation. Given a 3D scalar field (SDF in this case) $f(x, y, z)$, where x, y, z are grid coordinates, we can compute the gradient at each grid point using finite differences.

For an interior grid point, we use the 5-point stencil:

$$\frac{\partial f}{\partial x} = \frac{-f(x+2h,y,z) + 8f(x+h,y,z)}{12h} - \frac{8f(x-h,y,z) - f(x-2h,y,z)}{12h},$$
$$\frac{\partial f}{\partial y} = \frac{-f(x,y+2h,z) + 8f(x,y+h,z)}{12h} - \frac{8f(x,y-h,z) - f(x,y-2h,z)}{12h},$$
$$\frac{\partial f}{\partial z} = \frac{-f(x,y,z+2h) + 8f(x,y,z+h)}{12h} - \frac{8f(x,y,z-h) - f(x,y,z-2h)}{12h}.$$

For boundary grid points, using a 2-point stencil:

At the start boundary:

$$\frac{\partial f}{\partial x} = \frac{f(x+h,y,z) - f(x,y,z)}{h},$$
$$\frac{\partial f}{\partial y} = \frac{f(x,y+h,z) - f(x,y,z)}{h},$$
$$\frac{\partial f}{\partial z} = \frac{f(x,y,z+h) - f(x,y,z)}{h}.$$

At the end boundary:

$$\frac{\partial f}{\partial x} = \frac{f(x,y,z) - f(x-h,y,z)}{h},$$
$$\frac{\partial f}{\partial y} = \frac{f(x,y,z) - f(x,y-h,z)}{h},$$
$$\frac{\partial f}{\partial z} = \frac{f(x,y,z) - f(x,y,z-h)}{h}.$$

The resulting gradient vector at any grid point is then given by:

$$\mathbf{df} = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}\right).$$

To get the normal vector, normalize the gradient:

$$\mathbf{n} = \frac{\mathbf{df}}{||\mathbf{df}||}.$$

Training Details. We train our SDC with ADAM optimizer [34] for 200 epochs with a learning rate of 0.0001. The final layer of SDC is a scaled-sigmoid, with scaling factor of $\sigma = 10$ to facilitate learning. We use sigmoid activation so that the predicted vertex does not leave the cell. For all experiments in our main paper, we train on 3,000 shapes from the ABC dataset. All shapes are scaled to fit a unit-sphere. We pre-compute SDFs of all clean shapes at 64^3 resolution and apply augmentation on-the fly. We used $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 0.01$ in all our experiments. Detailed architecture depicting SDC is illustrated in Figure 8.

Method	Pre-Process	Training	Inference
NDC	9.0	**0.07**	**0.025**
UNDC	9.0	0.14	0.05
SDC	**2.7e-3**	0.10	**0.025**

Table 6. We compare timing between our meshing method (SDC) and two supervised Dual Contouring based baselines, NDC and UNDC. Reported timings are in seconds per-shape for 64^3 grid.

Why avoid supervision? The supervised data-driven approach NDC [13] emulates standard Dual Contouring using a local augmentation strategy for individual grid-cells, mitigating noisy triangulation from DC (refer to Figure 5 in [13]). Though this augmentation effectively enhances triangle quality, it remains an ad-hoc and heuristic solution. Conversely, SDC determines vertex positioning through a loss function mirroring the motivation of standard Dual Contouring, without relying on ad-hocs. Instead of addressing vertex positioning as a potentially ill-posed Linear Least Squares problem, we employ iterative optimization, a fitting choice for training neural networks. The axiomatic method's supervision is limiting since training is feasible only with pristine input data. SDC neither relies on clean input data nor a heuristic augmentation to circumvent poor triangulation.

Why less self-intersection? As mentioned above, NDC [13] is trained to emulate standard Dual Contouring vertices. Since Dual Contouring computes the intersection of the surface inside each voxel cube using a linear interpolation scheme, this implies, DC assumes the surface to be a smooth function that can be approximated by a linear function. However, at sharp edges and corners, the surface is not smooth, and this linear interpolation produces self-intersecting faces. On the other hand, the zero-level set of any function is free of self-intersection. Therefore, since SDC produces vertices to minimize discrepancy between SDFs, it is geometrically better motivated and as a result produces fewer self-intersections.

Timings. We report pre-processing, training and inference timing of our SDC and compare against supervised baselines NDC and UNDC [13] in Table 6. SDC has a negligible pre-processing time as it only involves scaling of the mesh. On the other-hand, NDC and UNDC computes dual-contouring ground-truth vertices which involves solving a linear system of equation for every occupied voxel and hence their pre-processing time is costlier. However, since our approach relies on point-to-face distance estimation (c.f. Eqn.7, main paper), our training is slightly costlier than both NDC and UNDC. On the other-hand, UNDC requires twice the training effort as it separately learns connectivity information. In practise, we require roughly 10hrs for train-

ing SDC. Similarly, for inference, UNDC requires twice the amount of time per-shape while SDC and NDC have similar inference time. All experiments were performed on Ampere-A100 GPU on a grid of resolution 64^3, averaged across our training set.

9.2. Regularizing DIN

As mentioned in the main paper, we used Curriculum DeepSDF [56] as our choice of DIN and followed the same hyper-parameters and training strategy advocated by the respective author. Since the curriculum learning strategy increases the level-of-detail of the reconstructed SDF gradually, we apply our regularization for the last 200 epochs. We use the weighting factor of our regularization term $\epsilon = 10$. We used points sampled on a regular grid that are closer to the surface as illustrated in Figure 7. To construct a mesh by SDC, we used a regular 64^3 grid. We trained all baselines, including ours for a total of 2000 epochs.

9.3. Mesh from Image(s)

We jointly train a ResNet-18 [28] and DeepSDF [56] to learn an implicit surface for each image from the training set. We train over ShapeNet [9] dataset and use the rendering provided by [79] as the input to ResNet-18 encoder. Rendering of each shape is encoded into a latent vector using ResNet-18, which is then concatenated along with a query point for which SDF is learnt by DeepSDF. To obtain query points, we sample 400,000 points close to the surface similar to [56]. We train for a total of 2,000 epoch per-category using ADAM [34] optimizer. For the baselines, use same hyper-parameters provided by the author for a fair comparison. We visualize the network used for end-to-end training in Figure 9.

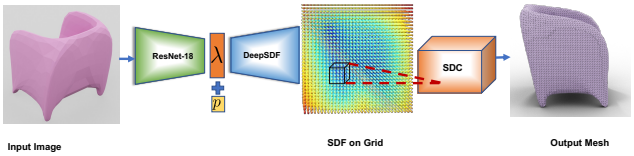


Figure 9. Network architecture for joint training for the task of Single View Reconstruction. Starting from an image, we construct an implicit representation via a latent code and then reconstruct the mesh using SDC.

10. Evaluation Metrics

We provide more details on various evaluation metrics used throughout the paper.:

- Chamfer Distance (CD) calculates a symmetric distance between two sets of points. Given χ_1 and χ_2 to be two point clouds,

$$CD(\chi_1, \chi_2) = \frac{1}{|\chi_1|} \sum_{x \in \chi_1} \min_{y \in \chi_2} ||x - y||_2 + \frac{1}{|\chi_2|} \sum_{y \in \chi_2} \min_{x \in \chi_1} ||y - x||_2.$$

- The Normal Consistency (NC) is often used as a metric for 3D surface reconstruction tasks to measure how well the estimated surface is consistent with the underlying geometry of the object:

$$\text{NC} = \frac{1}{N} \sum_{i=1}^{N} (1 - \cos(N_i, N_i^*)) \times 100\%,$$

  N_i and N_i^* are the estimated and ground truth surface normals, respectively, for the i^{th} point on the surface, and $\cos(N_i, N_i^*)$ represents the cosine similarity between the two vectors. The normalization term $\frac{1}{N}$ ensures that the metric is independent of the number of points on the surface.
- Edge Chamfer distance (ECD) is similar to the regular Chamfer Distance, but it is calculated for two sets of points, sampled on the edges of considered meshes. This metric better gauges how well the edges are reconstructed, or, another way of measuring sharpness.
- 3D Intersection-over-Union (IoU) is a metric used to compare pairs of 3D shapes, represented as 3D voxel grids G^1, G^2. It considers a ratio of the number of occupied voxels in the intersection of two occupancy grids to the number of occupied voxels in the union of occupancy grids:

$$IoU(G^1, G^2) = 100 * \frac{\sum_{ijk} G^1_{ijk} \wedge G^2_{ijk}}{\sum_{ijk} G^1_{ijk} \vee G^2_{ijk}},$$

  where, i, j, k are indices of voxel-cell along x,y,z dimensions.
- Precise Level Set Discrepancy (LSD-P). We propose this metric to gauge the discrepancy in the signed distance values between a reconstructed mesh and the ground truth mesh at fixed points in space. The fixed points are sampled on a regular grid $\mathcal{G}$ that are close to the surface of the ground truth mesh. This metric is defined as follows:

$$\text{LSD-P} = \sum_{g \in \bar{\mathcal{G}}} ||\text{abs}(\mathbf{d}^p(\mathcal{M}^*, g)) - \text{abs}(\mathbf{d}^p(\mathcal{M}, g))||_2,$$

  where $\bar{\mathcal{G}}$ denotes the grid points sampled close to the ground truth surface $\mathcal{M}^*$. Following the similar definition in our main paper, $\mathbf{d}^p$ denotes the distance between g and its closest point on a given surface. $\mathcal{M}$ refers to the reconstructed surface.

- Approximate Level Set Discrepancy (LSD-A). We introduce this metric to gauge the discrepancy in zero-level set between generated mesh and the ground truth mesh in circumstances where ground truth mesh might not be water-tight. This metric was used in our main paper for experiments pertaining to ShapeNet [9] dataset since it contains meshes that are non-watertight and estimating analytic SDF is ill-defined. We first sample points on the ground truth mesh and then measure the point-to-face distance to the generated mesh's closest face. It is defined as follows:

$$\text{LSD-A} = \frac{1}{N} \sum_{q \in \mathcal{M}^*} d^p(q, \mathcal{M}).$$

- SSIM (Structural Similarity Index) is a quality metric that measures the similarity between two images. The metric measures three components of image similarity: luminance, contrast, and structure. The structural information of two images include features such as edges, contrast, and texture.
- Self-Intersection. We use VCGlib [2] to determine face self-intersections. Initially, the algorithm employs spatial indexing using a grid to efficiently organize the mesh faces. This structure allows for identification of potentially intersecting faces by comparing their bounding boxes, thereby significantly reducing the number of detailed intersection tests required. Once potential intersecting pairs are identified, the algorithm determines actual geometric intersections. It first assesses the number of shared vertices between each pair of faces. If no vertices are shared, a direct triangle-to-triangle intersection test is performed. For pairs with a single shared vertex, the algorithm evaluates by creating segments from the non-shared vertices of each face, offset towards the shared vertex, and then examines these segments for intersections with the opposite triangle. For pairs of triangles that share an edge, intersection test is performed by checking the position of the third vertex.

11. Additional Results

Dataset	Method	CD (↓)	NC (↑)	SI (↓)	ECD (↓)	LSD-P (↓)
ABC	MC [38]	3.62	93.69	**0**	2.72	9.60
	NMC [11]	3.53	96.18	12.00	2.16	9.02
	NDC [13]	3.39	95.70	14.61	2.22	9.26
	Ours	**3.15**	**96.34**	6.84	**2.04**	**8.60**
Thingi10K	MC [38]	3.89	64.10	**0**	11.70	10.98
	NMC [11]	3.58	68.34	27.40	10.92	10.56
	NDC [13]	3.52	67.20	40.82	11.08	10.60
	Ours	**3.41**	**69.8**	14.80	**10.21**	**9.96**

Table 7. Quantitative mesh reconstruction results on the ABC and the Thingi10k dataset evaluated on a 128^3 SDF grid.

In this section, we provide additional qualitative and quantiative results. Throughout our main paper, we considered regular grids of size 64^3 for all our experiments. Now, we show that our method can be scaled to grid resolution of 128^3 without any additional training. SDC still produces minimal self-intersection and superior reconstruction in comparison to supervised baselines. We summarize our quantiative results in Table 7 and provide qualitative illustration in Figure 10.

In Figure 11 we show additional qualitative examples of meshes predicted from the learnt SDF grids of size 128^3. To show generalization of SDC, we use three Neural Fields, namely SIREN [69], Fourier Feature Network [72] and NGLOD as our neural field to produce implicit functions. Finally, in Figure 12 we show the additional qualitative comparison of our proposed regularization to the relevant baselines.

[2] https://github.com/cnr-isti-vclab/vcglib/

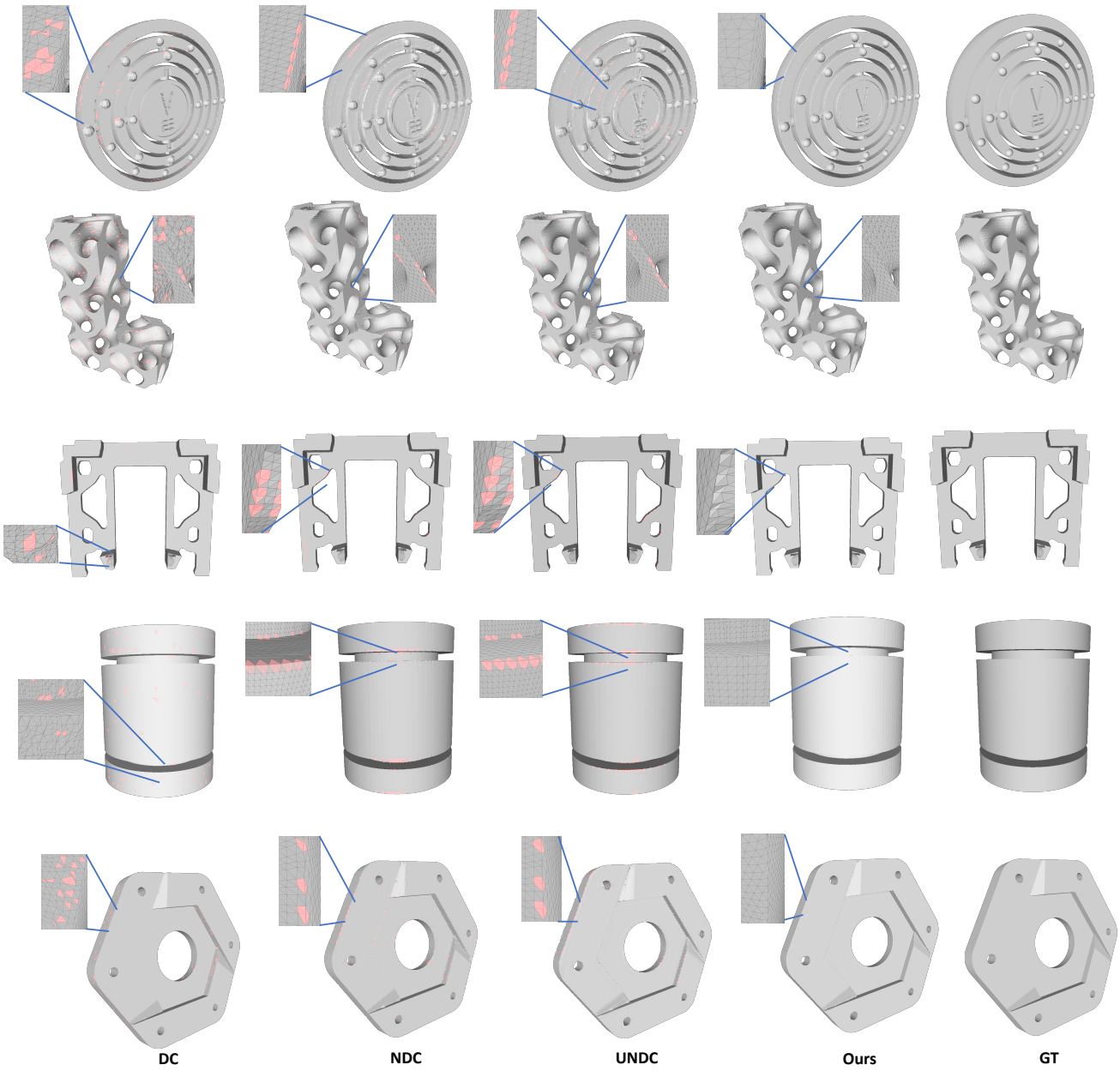


Figure 10. Qualitative comparison between meshes reconstructed from an input SDF of 128^3 resolution between DC [30], NDC and UNDC [13]. Self-intersecting faces are highlighted in red. Please zoom-in to see the detailed tessellation. First three rows are from Thingi10K dataset and last two rows are from the ABC dataset.

References

[1] Nina Amenta and Marshall Bern. Surface reconstruction by voronoi filtering. *Discrete & Computational Geometry*, 22 (4):481–504, 1999. 2

[2] Nina Amenta, Marshall Bern, and Manolis Kamvysselis. A new voronoi-based surface reconstruction algorithm. *Proc. SIGGRAPH*, pages 415–421, 1998. 2

[3] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry*, 19(2-3):127–153, 2001. 2

[4] Yizhak Ben-Shabat, Chamin Hewa Koneputugodage, and Stephen Gould. Digs: Divergence guided shape implicit neural representation for unoriented point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19323–19332, 2022. 3

[5] Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction from point clouds. *Computer Graphics Forum*, 36(1):

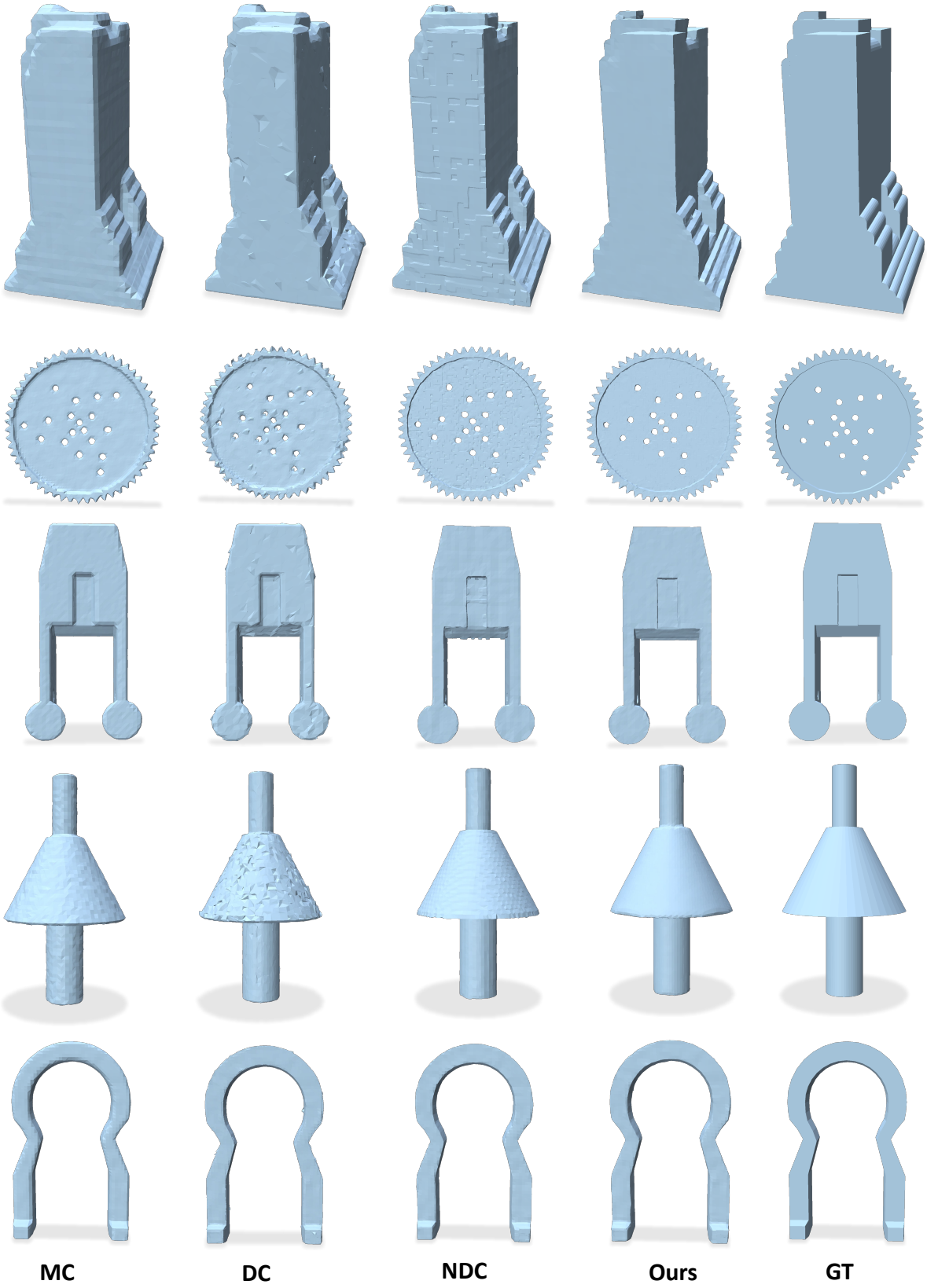


Figure 11. Qualitative comparison between reconstructed surfaces from the Thingi10k [81] dataset with input from various learnt methods. First two rows are from SIREN [69], second two are from FIN [72] and last row is from NGLOD [71]. Our approach produces sharper reconstruction than baselines.

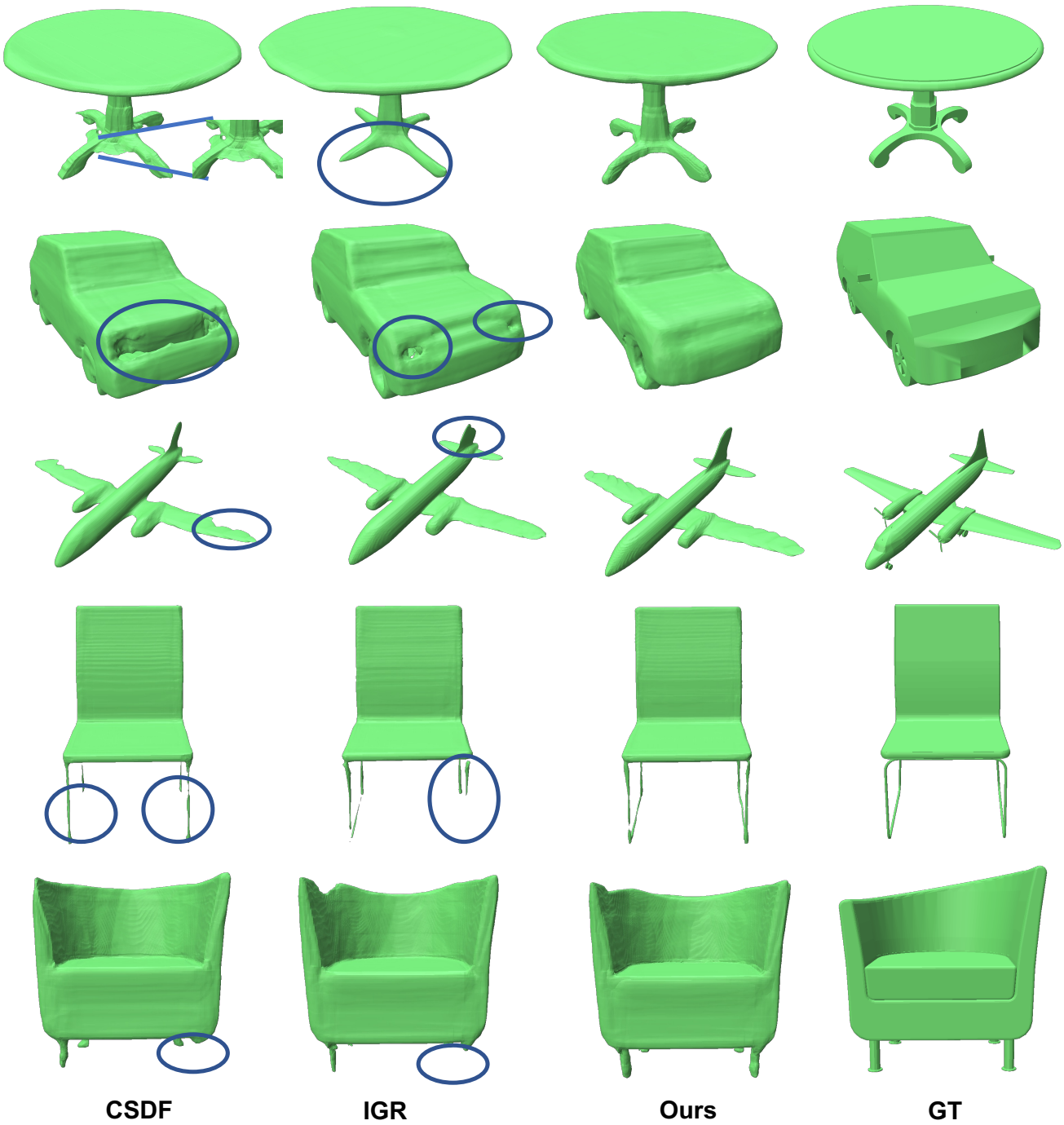


Figure 12. Qualitative comparison between reconstructed surfaces from the ShapeNet [9] dataset trained using different implicit surface reconstruction methods. Our approach captures finer details better than the baselines CSDF [18] and IGR [23], as indicated by blue circles.

301–329, 2017. 2

[6] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999. 2

[7] Jean-Daniel Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics (TOG)*, 3(4):266–286, 1984. 2

[8] Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graphical Models*, 67

(5):405–451, 2005. 2

[9] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2, 7, 1, 5, 6, 9

[10] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 3

[11] Zhiqin Chen and Hao Zhang. Neural marching cubes. *ACM Transactions on Graphics (TOG)*, 40(6):1–15, 2021. 1, 2, 3, 4, 5, 6, 7

[12] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. Bsp-net: Generating compact meshes via binary space partitioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 45–54, 2020. 3

[13] Zhiqin Chen, Andrea Tagliasacchi, Thomas Funkhouser, and Hao Zhang. Neural dual contouring. *arXiv preprint arXiv:2202.01999*, 2022. 1, 2, 3, 4, 5, 6, 7

[14] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312, 1996. 2

[15] Angela Dai and Matthias Nießner. Scan2mesh: From unstructured range scans to 3d meshes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5574–5583, 2019. 3

[16] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. Cvxnet: Learnable convex decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 31–44, 2020. 3

[17] Akio Doi and Akio Koide. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE TRANSACTIONS on Information and Systems*, 74(1):214–224, 1991. 2

[18] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J. Guibas. Curriculum deepsdf. *CoRR*, abs/2003.08593, 2020. 5, 7, 1, 3, 9

[19] Herbert Edelsbrunner and Ernst P Mücke. Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)*, 13(1):43–72, 1994. 2

[20] Jun Gao, Wenzheng Chen, Tommy Xiang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3d reconstruction. *Advances In Neural Information Processing Systems*, 33:9936–9947, 2020. 3

[21] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. *arXiv preprint arXiv:1904.06447*, 2019. 3

[22] S.F.F. Gibson. Using distance maps for accurate surface representation in sampled volumes. In *IEEE Symposium on Volume Visualization (Cat. No.989EX300)*, pages 23–30, 1998. 1

[23] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proceedings of Machine Learning and Systems 2020*, pages 3569–3579. 2020. 3, 5, 7, 1, 9

[24] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 3

[25] Benoit Guillard, Federico Stella, and Pascal Fua. Meshudf: Fast and differentiable meshing of unsigned distance field networks. In *European Conference on Computer Vision*, 2022. 3

[26] Rana Hanocka, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Point2mesh: A self-prior for deformable meshes. *ACM Trans. Graph.*, 39(4), 2020. 2

[27] John C. Hart. Sphere tracing: a geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10):527–545, 1996. 3

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 5

[29] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. *ACM SIGGRAPH Computer Graphics*, 26 (2), 1992. 2

[30] Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 339–346, 2002. 1, 2, 3, 4, 5, 6, 7

[31] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018. 3

[32] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, 2006. 2

[33] Alireza Khatamian and Hamid R Arabnia. Survey on 3d surface reconstruction. *Journal of Information Processing Systems*, 12(3), 2016. 2

[34] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4, 5

[35] Leif P. Kobbelt, Mario Botsch, Ulrich Schwanecke, and Hans-Peter Seidel. Feature sensitive surface extraction from volume data. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001. 1

[36] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6, 2

[37] Ravikrishna Kolluri, Jonathan Richard Shewchuk, and James F O'Brien. Spectral surface reconstruction from noisy point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 11–21. ACM, 2004. 2

[38] Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. Efficient implementation of marching

cubes' cases with topological guarantees. *Journal of Graphics Tools*, 8(2):1–15, 2003. 1, 6, 7

[39] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018. 2, 3

[40] Chen-Hsuan Lin, Oliver Wang, Bryan C Russell, Eli Shechtman, Vladimir G Kim, Matthew Fisher, and Simon Lucey. Photometric mesh optimization for video-aligned 3d object reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 969–978, 2019. 3

[41] David B. Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. *arXiv preprint arXiv:0000.00000*, 2021. 3

[42] Yaron Lipman. Phase transitions, distance functions, and implicit neural representations. In *International Conference on Machine Learning*, 2021. 3, 5

[43] Or Litany, Alex Bronstein, Michael Bronstein, and Ameesh Makadia. Deformable shape completion with graph convolutional autoencoders. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1886–1895, 2018. 3

[44] Hsueh-Ti Derek Liu, Francis Williams, Alec Jacobson, Sanja Fidler, and Or Litany. Learning smooth neural functions via lipschitz regularization, 2022. 3

[45] Minghua Liu, Xiaoshuai Zhang, and Hao Su. Meshing point clouds with predicted intrinsic-extrinsic ratio guidance. In *European Conference on Computer Vision*, pages 68–84. Springer, 2020. 3

[46] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 1, 2

[47] N. Maruani, R. Klokov, M. Ovsjanikov, P. Alliez, and M. Desbrun. Voromesh: Learning watertight surface meshes with voronoi diagrams. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 3

[48] Ishit Mehta, Manmohan Chandraker, and Ravi Ramamoorthi. A level set theory for neural implicit evolution under explicit flows. *arXiv preprint arXiv:2204.07159*, 2022. 3

[49] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 3

[50] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*, pages 35–57. Springer, 2003. 1

[51] Niloy J. Mitra, Natasha Gelfand, Helmut Pottmann, and Leonidas Guibas. Registration of point cloud data from a geometric optimization perspective. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, page 22–31, New York, NY, USA, 2004. Association for Computing Machinery. 4, 2

[52] Patrick Mullen, Fernando De Goes, Mathieu Desbrun, David Cohen-Steiner, and Pierre Alliez. Signing the unsigned: Robust surface reconstruction from raw pointsets. *Computer Graphics Forum*, 29(5):1733–1741, 2010. 2

[53] Timothy S Newman and Hong Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879, 2006. 2

[54] G.M. Nielson. Dual marching cubes. In *IEEE Visualization 2004*, pages 489–496, 2004. 5

[55] Junyi Pan, Xiaoguang Han, Weikai Chen, Jiapeng Tang, and Kui Jia. Deep mesh reconstruction from single rgb images via topology modification networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9964–9973, 2019. 3

[56] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2, 3, 5, 7

[57] Songyou Peng, Chiyu Jiang, Yiyi Liao, Michael Niemeyer, Marc Pollefeys, and Andreas Geiger. Shape as points: A differentiable poisson solver. *Advances in Neural Information Processing Systems*, 34:13032–13044, 2021. 3

[58] Helmut Pottmann and Michael Hofer. Geometry of the squared distance function to curves and surfaces. In *Mathematics and Visualization*, pages 221–242. Springer Berlin Heidelberg, 2003. 4, 2

[59] Helmut Pottmann, Stefan Leopoldseder, and Michael Hofer. Registration without ICP. *Computer Vision and Image Understanding*, 95(1):54–71, 2004. 2

[60] Albert Pumarola, Artsiom Sanakoyeu, Lior Yariv, Ali Thabet, and Yaron Lipman. Visco grids: Surface reconstruction with viscosity and coarea grids. In *Advances in Neural Information Processing Systems*, 2022. 3, 5

[61] Marie-Julie Rakotosaona, Paul Guerrero, Noam Aigerman, Niloy J Mitra, and Maks Ovsjanikov. Learning delaunay surface elements for mesh reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22–31, 2021. 3, 5

[62] Edoardo Remelli, Artem Lukoianov, Stephan Richter, Benoît Guillard, Timur Bagautdinov, Pierre Baque, and Pascal Fua. Meshsdf: Differentiable iso-surface extraction. *Advances in Neural Information Processing Systems*, 33:22468–22478, 2020. 2, 3, 5, 1

[63] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2304–2314, 2019. 3

[64] Scott Schaefer, Tao Ju, and Joe Warren. Manifold dual contouring. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):610–619, 2007. 1

[65] Nicholas Sharp and Maks Ovsjanikov. Pointtrinet: Learned triangulation of 3d point sets. In *European Conference on Computer Vision*, pages 762–778. Springer, 2020. 1, 2, 3, 5

[66] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems*, 34:6087–6101, 2021. 3

[67] Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. Flexible isosurface extraction for gradient-based mesh optimization. *ACM Trans. Graph.*, 42(4), 2023. 3, 6, 7

[68] Jonathan Shewchuk, Tamal K Dey, and Siu-Wing Cheng. *Delaunay mesh generation*. Chapman and Hall/CRC, 2016. 2

[69] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 3, 6, 8

[70] G D Smith. *Numerical solution of partial differential equations*. Clarendon Press, Oxford, England, 3 edition, 1985. 4

[71] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3D shapes. 2021. 6, 7, 8

[72] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 3, 6, 8

[73] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019. 2

[74] Graham M Treece, Richard W Prager, and Andrew H Gee. Regularised marching tetrahedra: improved iso-surface extraction. *Computers & Graphics*, 23(4):583–598, 1999. 2

[75] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 3

[76] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019. 3

[77] T. Wu, L. Pan, J. Zhang, T. Wang, Z. Liu, and D. Lin. Balanced chamfer distance as a comprehensive metric for point cloud completion. In *Advances in Neural Information Processing Systems*, 2021. 2

[78] Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41(2):641–676, 2022. 3

[79] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems 32*, pages 492–502. Curran Associates, Inc., 2019. 1, 5

[80] Guandao Yang, Serge Belongie, Bharath Hariharan, and Vladlen Koltun. Geometry processing with neural fields. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 3

[81] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10, 000 3d-printing models. *CoRR*, abs/1605.04797, 2016. 6, 7, 8