Self-Supervised Dual Contouring

Supplementary Material

1. 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 2. Then, we provide additional implementation details pertaining to our three experimental settings in Section 3. We elaborate on various evaluation metrics used in the main paper in Section 4. Finally, we provide additional quantitative evaluation and qualitative results in Section 5. Our entire code will be released upon publication.

2. Ablation Studies

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

2.1. 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 [9] 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 [11, 13, 14]. To that end, we consider quadratic approximation of point-point distance proposed in [11] as our baseline. In particular, [11] 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 [9]. Since we explicitly use the mesh, we consider this to be supervised baseline. This supervised training objective is given as follows:

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

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

Our quantitative results are summarized in Table 1. 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 [18, 19] 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.

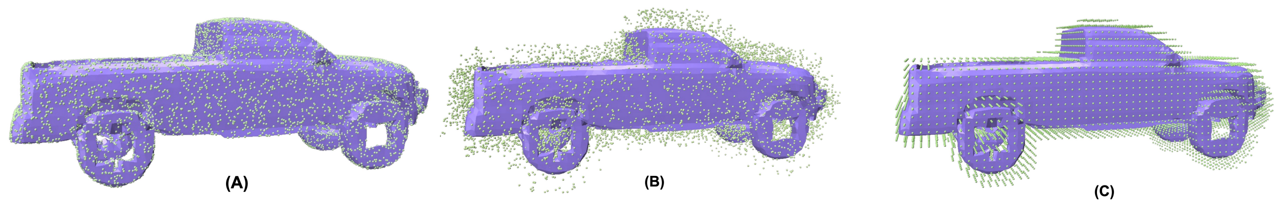

Figure 1. Depicting point samples (in green) at which our signed distance based regularization is applied. The implicit function is meshed using SDC and rendered in purple. (A) denotes surface sampling, (B) Denotes irregular near-surface sampling and (C) is regular near-surface sampling.

Sampling Type	CD ↓ (x10ˆ3)	NC ↑ (%)	IoU ↑ (%)	LSD-A ↓ (x10ˆ3)	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 2. Comparing quantitative reconstruction results and timing between different sampling strategies for applying our mesh-based regularization.

2.2. 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 1. 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 2. 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 [4] is required to determine the SDF values. All experiments were performed on Ampere-A100 GPUs for fairness in comparison.

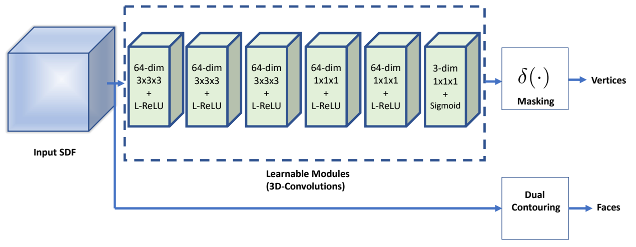

Figure 2. Detailed depiction of SDC framework. Starting from input SDF defined on a regular grid, we predict the vertices of the mesh while constructing the faces following the Dual Contouring [7] technique.

3. 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.

3.1. 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:

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

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

At the start boundary:

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

At the end boundary:

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

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 [8] 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 2.

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 3. 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 [3] emulates standard Dual Contouring using a local augmentation strategy for individual grid-cells, mitigating noisy triangulation from DC (refer to Figure 5 in [3]). 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 [3] 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 [3] in Table 3. 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 training 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.

3.2. Regularizing DIN

As mentioned in the main paper, we used Curriculum DeepSDF [12] 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 1. 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.

3.3. Mesh from Image(s)

We jointly train a ResNet-18 [6] and DeepSDF [12] to learn an implicit surface for each image from the training set. We train over ShapeNet [1] dataset and use the rendering provided by [20] 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 [12]. We train for a total of 2,000 epoch per-category using ADAM [8] 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 3.

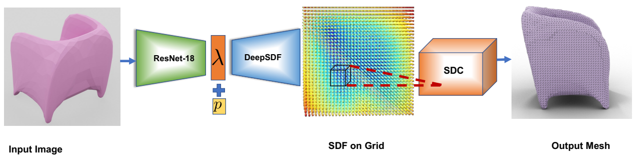

Figure 3. Network architecture for joint training for the task of Single View Reconstruction. Starting from an image, we construct an implicit representation via a latent code and then reconstruct the mesh using SDC.

4. 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 [1] 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 [1] 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.

5. Additional Results

Dataset	Method	CD (↓)	NC (↑)	SI (↓)	ECD (↓)	LSD-P (↓)
ABC	MC [10]	3.62	93.69	**0**	2.72	9.60
	NMC [2]	3.53	96.18	12.00	2.16	9.02
	NDC [3]	3.39	95.70	14.61	2.22	9.26
	Ours	**3.15**	**96.34**	6.84	**2.04**	**8.60**
Thingi10K	MC [10]	3.89	64.10	**0**	11.70	10.98
	NMC [2]	3.58	68.34	27.40	10.92	10.56
	NDC [3]	3.52	67.20	40.82	11.08	10.60
	Ours	**3.41**	**69.8**	14.80	**10.21**	**9.96**

Table 4. 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 4 and provide qualitative illustration in Figure 4.

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

In Figure 5 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 [15], Fourier Feature Network [17] and NGLOD as our neural field to produce implicit functions. Finally, in Figure 6 we show the additional qualitative comparison of our proposed regularization to the relevant baselines.

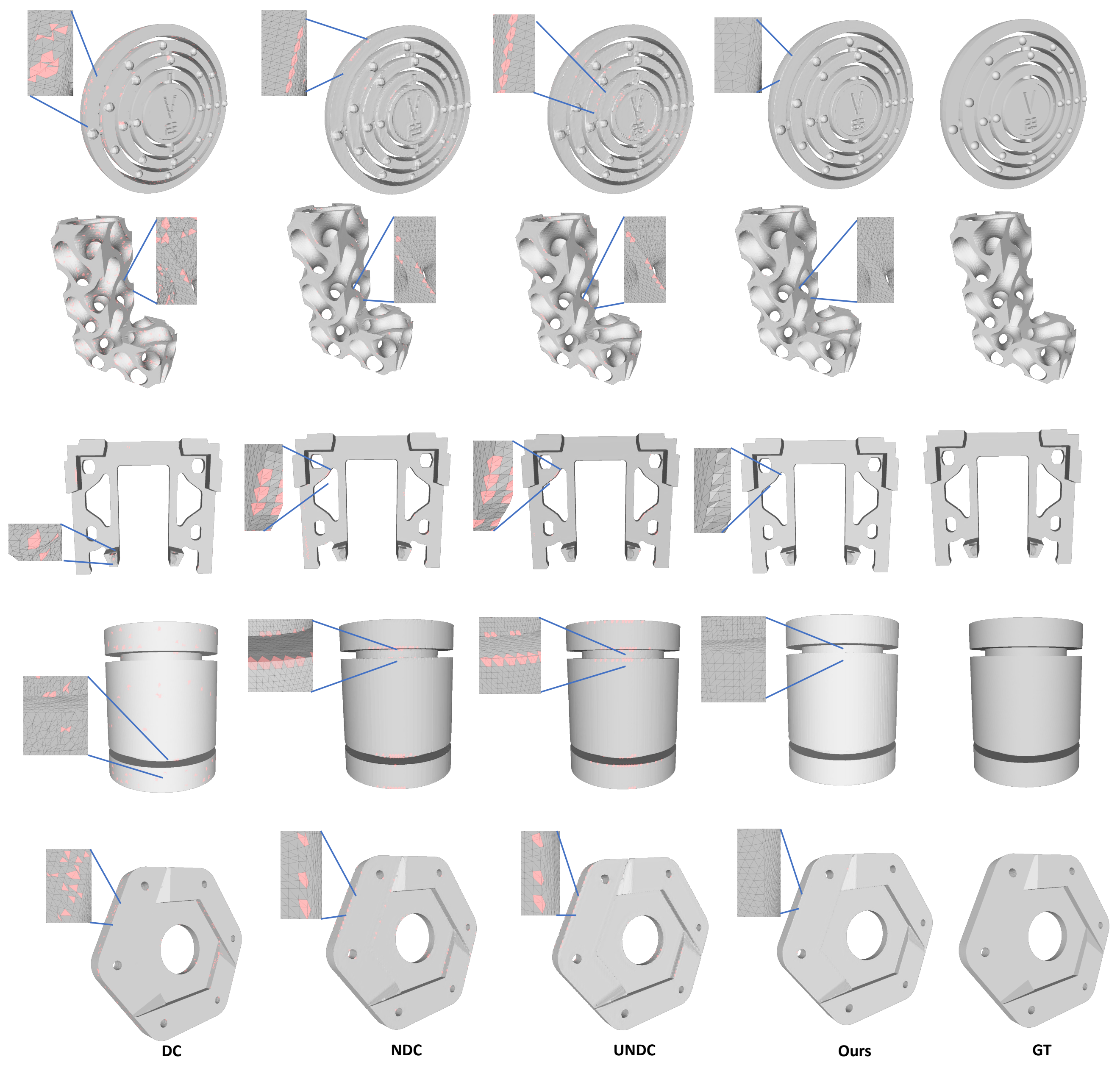

Figure 4. Qualitative comparison between meshes reconstructed from an input SDF of 128^3 resolution between DC [7], NDC and UNDC [3]. Self-intersecting faces are highlighted in red. Please zoom-in to see the detailed tessellation. First three rows are from Thingi10K dataset and last two rows are from the ABC dataset.

References

[1] 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. 4, 5, 8

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

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

[4] Yueqi Duan, Haidong Zhu, He Wang, Li Yi, Ram Nevatia, and Leonidas J. Guibas. Curriculum deepsdf. *CoRR*, abs/2003.08593, 2020. 2, 8

[5] 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. 8

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun.

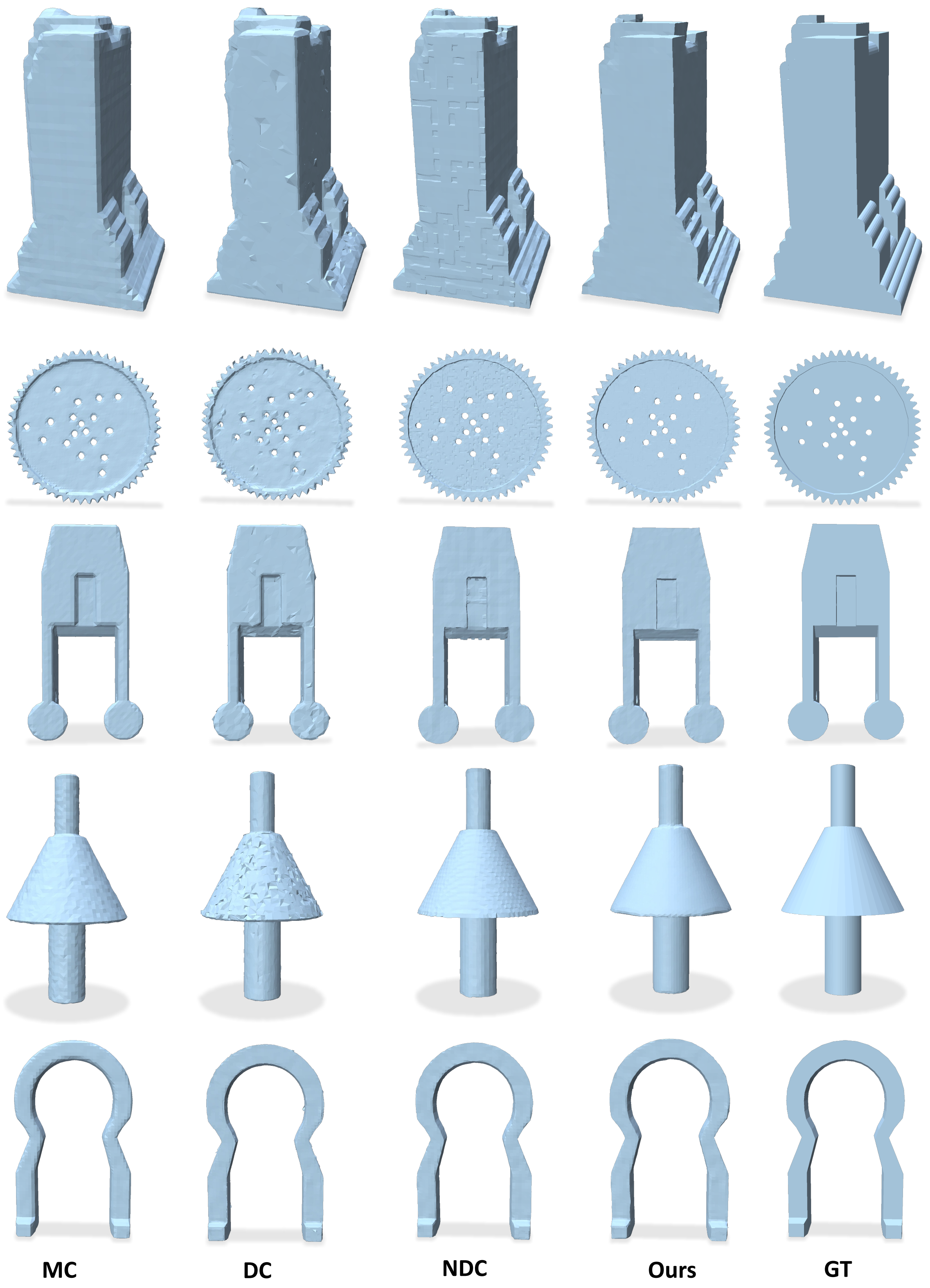

Figure 5. Qualitative comparison between reconstructed surfaces from the Thingi10k [21] dataset with input from various learnt methods. First two rows are from SIREN [15], second two are from FIN [17] and last row is from NGLOD [16]. Our approach produces sharper reconstruction than baselines.

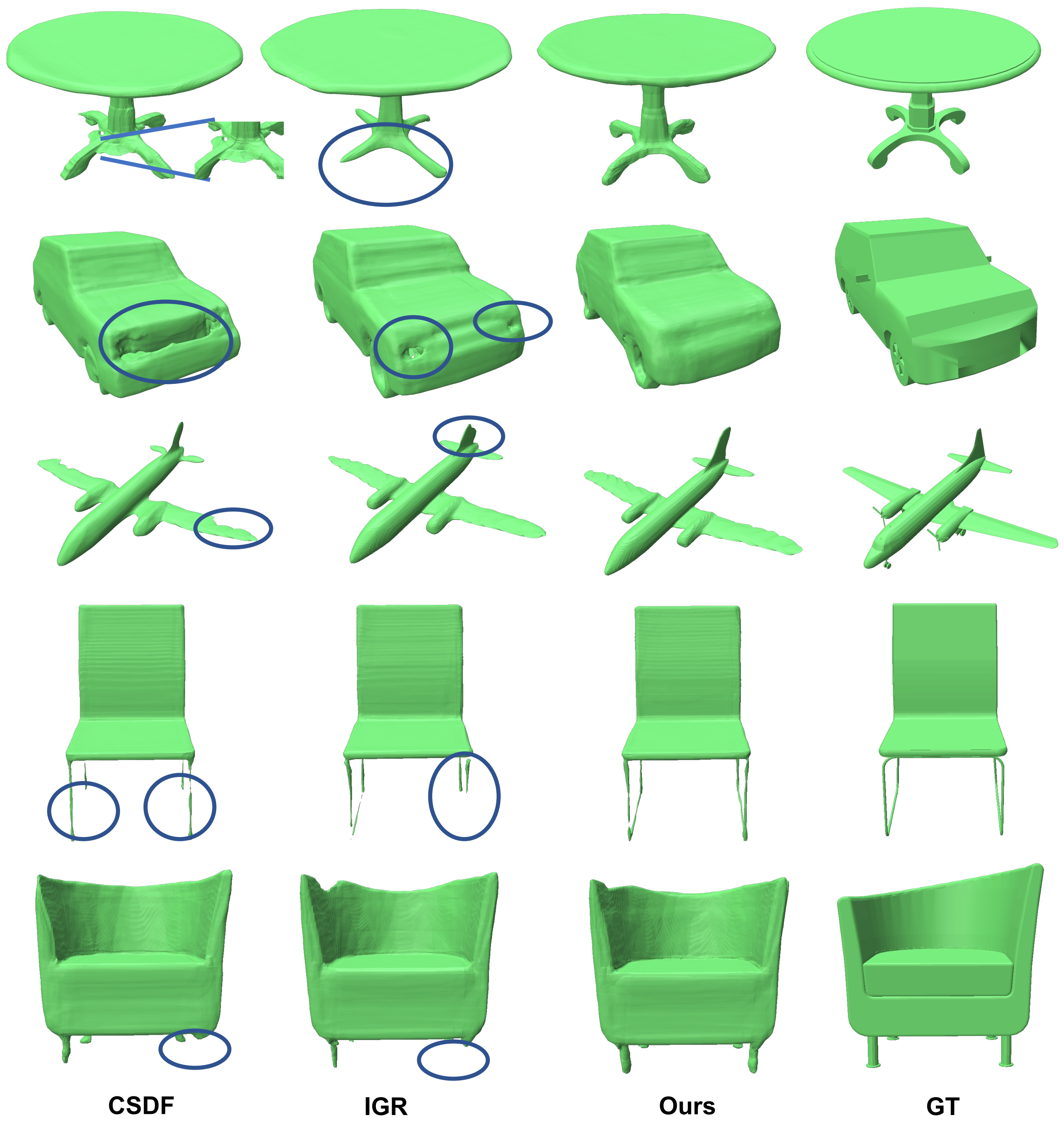

Figure 6. Qualitative comparison between reconstructed surfaces from the ShapeNet [1] dataset trained using different implicit surface reconstruction methods. Our approach captures finer details better than the baselines CSDF [4] and IGR [5], as indicated by blue circles.

Deep residual learning for image recognition, 2015. 4

[7] 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. 2, 6

[8] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 3, 4

[9] 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. 1

[10] 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. 5

[11] 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. 1

[12] 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. 4

[13] 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. 1

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

[15] 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. 5, 7

[16] 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. 7

[17] 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. 5, 7

[18] 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. 1

[19] 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. 1

[20] 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. 4

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