S2MDF: A Plug-And-Play Layer for Intersection-Free Multi-Object Signed Distance Fields

Deniz Sayin Mercadier* **Federico Stella*** **Aurel Bizeau** **Nicolas Talabot** **Pascal Fua**
CVLab, Ecole Polytechnique Fédérale de Lausanne (EPFL)
Lausanne, Switzerland
{firstname.lastname}@epfl.ch

Abstract

Compositional implicit surface representations model scenes as collections of objects, each encoded by a Signed Distance Field (SDF). A fundamental limitation of this approach is that multiple SDFs can produce geometries that interpenetrate, violating physical plausibility. Existing mitigation strategies rely on soft penalty terms that reduce but do not eliminate intersections, and require careful loss weighting. To truly prevent interpenetration, we propose a hard constraint on vector-valued SDFs and introduce S2MDF, a lightweight plug-and-play module that enforces the constraint on any object-compositional SDF representation without architectural modifications. It introduces negligible computational overhead and is compatible with linearly-interpolated standard meshing algorithms such as Marching Cubes. It can be applied during training or as a post-processing step. Experiments on multiple state-of-the-art compositional methods show that S2MDF reduces intersections to numerical precision while preserving reconstruction quality, outperforming existing mitigation strategies.

1 Introduction

Over the past few years, implicit neural representations (INRs) have reshaped the landscape of 3D vision and graphics. By encoding geometry or appearance as continuous functions parameterized by neural networks, methods such as DeepSDF [41], 3DShape2VecSet [72], NeRF [36], NeuS [60] and their many extensions have demonstrated remarkable performance in tasks such as novel view synthesis [53], high-fidelity surface reconstruction [18], shape generation [28], manipulation [22] and optimization [23]. Unlike older discrete representations such as meshes and voxels, INRs provide resolution-independent modeling, differentiability, and a natural framework for learning complex geometric priors. This paradigm shift has fostered significant progress in fields as diverse as robotics, medical imaging, and digital content creation, where accurate and flexible 3D representations are paramount.

As these methods mature, a growing body of work has focused on extending INRs beyond single-object descriptions towards more realistic multi-object representations of scenes [63, 54, 64] and multi-part models of composite objects [14, 43, 52]. These are particularly valuable for downstream tasks such as manipulation, simulation, and design, where reasoning about individual entities or parts is key. For example, when designing the 3D shape of a car to improve its aerodynamics, one has to consider its body, its wheels, and all the accessories that are usually attached to it. Similarly, when modeling a heart from medical imagery, one should consider its four chambers and the blood vessels attached to them. Numerous object-compositional implicit representations [76, 74, 32, 64, 52, 29] have been introduced to this end. A common approach is to rely on a set of Signed Distance Fields (SDF) [60], one per object [63, 64, 52], and to combine them appropriately.

*Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

While effective and popular, this approach faces a fundamental challenge: Even though physical objects cannot share the same volume in space, multiple SDFs can easily produce configurations that violate this. A number of strategies have been proposed to mitigate this problem. They include minimizing loss terms that penalize overlap [52], visibility-based constraints to handle occlusions [64], and regularization techniques inspired by physical reasoning [29]. While these approaches reduce the frequency and severity of intersections, they do not guarantee their complete removal. Furthermore, because the intersection loss terms are usually minimized as part of a weighted sum of several other loss terms, careful tuning of their respective weights is often required.

In this work, we propose an approach that strictly prevents intersections without needing a weighting scheme. To this end, we define a hard-constraint on vector-valued SDFs that produces an intersection-free geometry. We will refer to such a constrained field as a *Multi-object Distance Field (MDF)*. Given this formalism, we implement *S2MDF*, a simple yet effective module that enforces the constraint and can be seamlessly integrated into existing pipelines with a negligible computational overhead. It can be applied either during training or as a post-processing step to enforce the intersection-free constraint on virtually any object-compositional SDF-based representation, without requiring any architectural change. Furthermore, it remains compatible with standard linearly-interpolated meshing techniques such as Marching Cubes [34] and Marching Tetrahedra [5].

To summarize our contributions:

- We formalize the problem of generating intersection-free multi-object implicit surface representations by constraining vector-valued SDFs to encode intersection-free geometry as a Multi-object Distance Field (MDF);
- We propose S2MDF, a fast, plug-and-play module that projects object-compositional SDF-based representations into an MDF, ensuring intersection-free geometry with negligible computational overhead.

We validate our approach on several state-of-the-art SDF-based object-compositional methods, demonstrating that S2MDF, paired with existing linearly-interpolated meshing techniques, reduces intersections to numerical precision while preserving reconstruction quality. Through both qualitative and quantitative evaluations, we show that our method outperforms existing mitigation strategies and provides a robust, generic solution to a longstanding limitation of compositional neural surface representations.

2 Related work

We first put Implicit Neural Representations in the more general context of representing the 3D shape of single objects and then briefly review existing approaches to jointly model several objects.

Representing the Shape of 3D Objects. Many different representations have been proposed over the years. Voxel-based methods represent the space as a grid [66, 62, 12, 13], but suffer from high memory requirements and limited resolution. Point-cloud-based methods represent surfaces as sets of points [17, 68, 1, 42, 70], which improves the memory footprint but ignores connectivity, as they do not explicitly model the surface as a continuous entity. Mesh-based methods represent surfaces as collections of vertices, edges and faces [19, 59, 25, 40], which allows for efficient rendering and manipulation, but generally require a fixed topology that is difficult to modify. Implicit Neural Representations represent a powerful alternative, encoding geometry as continuous functions parameterized by neural networks [41, 35, 9]. Among these, Signed Distance Fields (SDFs) and occupancy fields are particularly popular for representing watertight surfaces, which can then be extracted from the field using standard meshing techniques [34, 15, 30, 24]. When the surface is not watertight, Unsigned Distance Fields (UDFs) can be used instead [11, 33, 31], with specially designed meshing techniques [21, 73, 49]. Differentiability between the explicit surface and the field can be achieved thanks to the implicit function theorem, whose practical viability has been shown for both SDFs [45, 20] and UDFs [21], allowing shape optimization tasks such as drag reduction. Recent works extend these approaches using new latent spaces [51, 37, 71, 72] and more complex network architectures [75, 6], allowing for highly detailed generative modeling and shape manipulation.

Handling Multiple Objects Jointly. Object-aware implicit representations have been proposed in the context of NeRFs [39, 54], often involving a separate semantic branch to predict object labels [58, 16], and in some cases two separate radiance fields, one for the scene and one for the objects [67]. However these approaches do not directly model geometry. ObjectSDF [63] and

ObjectSDF++ [64] are among the first works to propose object-compositional implicit surface representations, where multiple SDFs are used to represent different objects. ObjectSDF [63] defines an approach to directly render the semantic field, coupling semantics and geometry, while ObjectSDF++ [64] supervises the geometry with semantic information and adds a loss term to penalize object collisions. However, neither approach prevents interpenetrations between objects. Part-based generative models [65, 14, 43] represent individual parts of objects but typically output independent, potentially non-physical, SDFs. The more recent PartSDF [52] proposes a single decoder to produce all fields jointly, with a loss term to penalize intersections, but again without guarantees. Other works in the medical and robotics domains, where physical plausibility is key, also use special losses to reduce penetration between parts [26, 74, 29], while sometimes enforcing contact [29], but they either do not guarantee the absence of intersections or they are not based on INRs. Instead, we propose a simple module that can be applied to any object-compositional SDF-based representation, either during training or as a post-processing step, to prevent interpenetration while preserving the quality of the reconstructions.

3 Method

Signed Distance Fields (SDFs) naturally represent single, watertight, objects by encoding the positive or negative distance to the closest surface for each point in space, the typical convention being that negative values are inside the object and positive ones outside. For a watertight object O and its boundary ∂O, this can be written as

$$SDF_O : \mathbb{R}^3 \to \mathbb{R} \ , \quad SDF_O(\mathbf{p}) = \begin{cases} -\min_{\mathbf{q}\in\partial O} \|\mathbf{p}-\mathbf{q}\|, & \text{if } \mathbf{p} \in O \\ \min_{\mathbf{q}\in\partial O} \|\mathbf{p}-\mathbf{q}\|, & \text{if } \mathbf{p} \notin O \end{cases} \tag{1}$$

In an object-compositional setting, each one of K objects is typically assigned its own SDF_k for $1 \leq k \leq K$ and we can define the composite $SDF = [SDF_1, \ldots, SDF_K] : \mathbb{R}^3 \to \mathbb{R}^K$. The problem with this is that each SDF is independent from the others: Nothing stops two of the SDF_k from returning negative values at the same spatial location, which would mean that this location is inside two different objects, a configuration that should not be allowed. This is what we want to prevent by appropriately constraining SDF. We first formalize this constraint and then discuss various approaches to enforcing it.

3.1 Multi-Object Signed Distance Fields

Let $\min^{(n)}(\cdot)$ be the operator that returns the n-th smallest element of its input vector. Given a spatial location $\mathbf{p}$, let $\mathbf{u} = SDF(\mathbf{p})$ be its vector-valued composite SDF. To ensure that $\mathbf{p}$ is not inside more than one object it should satisfy

$$\min^{(2)}(\mathbf{u}) \geq 0 \ . \tag{2}$$

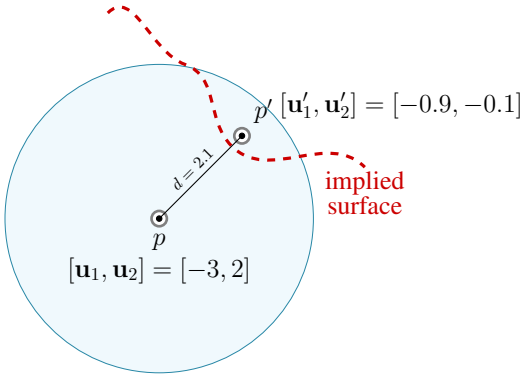
This constraint ensures that the second smallest signed distance value at any point is non-negative, which means that at most one of the SDF_k can be negative at any location. If we could truly enforce it everywhere, this would be enough. But, in practice, we have to sample the space more or less densely and we can only enforce the constraint at those sample points. Furthermore, the deep networks used to implement the SDF_k functions do not guarantee that they satisfy the eikonal property $\|\nabla SDF_k\| = 1$. Taken together, these two limitations mean that SDF approximations can satisfy the constraint of Eq. (2) on sample points and yet produce interpenetrations, as shown in Fig. 1a. To remedy this, we propose to use an equivalent constraint based on the following theorem

Theorem 1. *A vector-valued $SDF = [SDF_1, \ldots, SDF_K] : \mathbb{R}^3 \to \mathbb{R}^K$ whose components are exact SDFs as in Eq.* (1) *satisfies constraint 2 at every point $\mathbf{p} \in \mathbb{R}^3$, with $\mathbf{u} = SDF(\mathbf{p})$, if and only if it satisfies the following constraint at every point $\mathbf{p} \in \mathbb{R}^3$:*

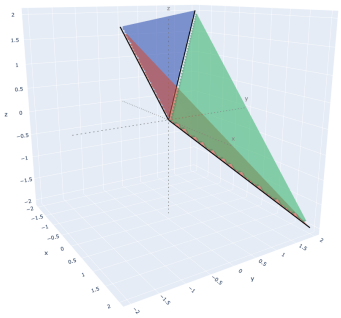
$$\min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}) \geq 0 \ . \tag{3}$$

Proof: Appendix A.2.1, using Lipschitz continuity of exact SDFs.

This constraint is visualized in Fig. 1b. Even though Eq. (2) and Eq. (3) are equivalent for exact SDFs sampled at every point in space, the second constraint is *stronger* for neural, imperfect SDFs, sampled on a discrete set of points, which is a typical case in practice. In such case, Eq. (3) remains sufficient for Eq. (2), but it is no longer necessary. In fact, the following holds:



(a) Given a point $\mathbf{p}$ inside the circle and its SDF $\mathbf{u}$ satisfying $\min^{(2)}(\mathbf{u}) \geq 0$, there is an implication of a nearby point $\mathbf{p}'$ with SDF $\mathbf{u}'$ having both entries negative, thus $\min^{(2)}(\mathbf{u}') \not\geq 0$. This value of $\mathbf{u}$ would not be allowed if Eq. (2) was satisfied everywhere.



(b) Illustration of the feasible space for the MDF constraint in Eq. (3) for $K = 3$ objects: an intersection of $C(K, 2)$ hyperplanes in $\mathbb{R}^K$ delimits the feasible region on the right. Each plane corresponds to the constraint imposed by the sum of one pair of SDFs.

Figure 1: Illustrations of properties related to the constraints in Eq. (2) and Eq. (3).

Theorem 2. *Given two points $\mathbf{p_1}, \mathbf{p_2} \in \mathbb{R}^3$ and an arbitrary vector-valued function $f : \mathbb{R}^3 \to \mathbb{R}^K$ evaluated at those points, $\mathbf{u}_1 = f(\mathbf{p_1})$ and $\mathbf{u}_2 = f(\mathbf{p_2})$ that satisfy constraint 3, then the constraint is also satisfied for every point along the line segment connecting $\mathbf{p_1}$ and $\mathbf{p_2}$, where intermediate vectors are obtained via linear interpolation as $\mathbf{u}_\alpha = \alpha\mathbf{u}_1 + (1 - \alpha)\mathbf{u}_2$ for $\alpha \in [0, 1]$. Proof: Appendix A.2.3.*

In other words, a linear interpolation of an imperfect SDF that satisfies constraint 3 on a finite set of points, also satisfies the constraint everywhere. The same is not true for constraint 2, as shown in Appendix A.2.4 and hinted in Fig. 1a. This is important because, as mentioned above, SDF parametrizations are not perfect, and are often evaluated at a finite number of points and interpolated in between, for example by iso-surface extraction algorithms. Thus the formulation of Theorem 1 is the one we use in practice and we define our Multi-object Distance Fields as follows:

Definition 1. A Multi-object Distance Field (MDF) is a vector-valued SDF $\mathbb{R}^3 \to \mathbb{R}^K$ that maps each point in space to a vector of K signed distance values, one for each of K objects, and satisfies the constraint of Eq. (3).

3.2 Enforcing the MDF Constraint

Given a vector-valued SDF, independently from its parametrization, we want to ensure that the K-dimensional vector $\mathbf{u} = SDF(\mathbf{p})$ meets the constraint of Eq. (3) for all $\mathbf{p}$ it is evaluated at. The natural way to do this is to perform a projection, that is, replace $\mathbf{u}$ by the vector $\mathbf{d}$ that is closest to $\mathbf{u}$ and meets the constraints. We write this as

$$\text{minimize} \|\mathbf{d} - \mathbf{u}\|^2 \quad \text{subject to} \quad \min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0 \; . \tag{4}$$

Theorem 2 ensures that the linear interpolation of those samples satisfies the MDF constraint. Thus Eq. (4) gives us a separate problem to solve for each sample point. As we normally perform computations on batches of points, the solution must be regular enough to be easy to parallelize for a fast computation. When $K = 2$, Eq. (4) has a simple analytical solution and this is no issue. However, for $K > 2$, the non-linearity of $\min^{(1)}$ and $\min^{(2)}$ becomes a problem and there are no standard solvers that can be used for our purposes. We implemented two distinct solutions to this problem.

3.2.1 Quadratic Programming

The minimization of Eq. (4) is subject to a non-linear constraint but we can rewrite it as

$$\text{minimize} \|\mathbf{d} - \mathbf{u}\|^2 \quad \text{subject to} \quad d_i + d_j \geq 0, \quad 1 \leq i < j \leq K \; . \tag{5}$$

We prove in Appendix A.2.2 that the constraints of Eqs. (4) and (5) are *strictly* the same, meaning the problem and by extension its solutions are the same. While the formulation of Eq. (5) may seem more complex, it has the advantage to be a standard Quadratic Program (QP) for which there are mature solvers [2, 3] whose solution can be readily differentiated. Thus, they can be directly integrated inside neural networks for an acceptably small runtime cost as long as K is not too large. In our implementation, we embed one in the final layer of the neural network parameterizing all the signed-distance values, effectively projecting SDF values onto the manifold of intersection-free MDFs.

3.2.2 Analytical Solution

The drawback of our QP formulation is that the number of constraints, and by extension the runtime cost of solving the problems, scales quadratically with the number of objects K, which can become prohibitive, especially during training when they are invoked at every iteration. This can be addressed using the following heuristic. When $K = 2$, the analytical solution to Eq. (4) is to add $-(\min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}))/2$ to each $\mathbf{u}$, see Appendix A.2.6 for the proof. When $K > 2$, we can still replace each $\mathbf{u}$ by

$$\text{Shift-All}(\mathbf{u}) = \begin{cases} \mathbf{u} & \text{if } \min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}) \geq 0 \ , \\ \mathbf{u} - \dfrac{\min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u})}{2} & \text{otherwise.} \end{cases} \tag{6}$$

This is no longer the optimal solution to the problem of Eq. (4), but it is a feasible one, and we show in Appendix A.2.7 that it is the solution to the problem of Eq. (4) under the additional constraint that the pairwise differences between the different SDF values be preserved, a reasonable constraint to avoid disrupting the ordering of the original fields during the projection. It has the added benefit to be much simpler and faster to compute than solving the QP problem.

This mapping is continuous, differentiable almost everywhere, and the non-differentiable points are typically not on the object surfaces. In other words, it does not create issues either for meshing or training, unlike the non-differentiability of Unsigned Distance Fields on the surface that is well known to create problems [21, 73, 49]. Moreover, being an optimal solution to problem 4 does not necessarily mean that the QP solution is better in practice, as optimality is defined in terms of the L_2 distance to the original vector $\mathbf{u}$, which is an arbitrary metric. In practice, we show in the results section that this heuristic performs equally well as the QP solution, and even better in some cases, at a much lower computational cost.

3.3 Meshing

If explicit representations are required by downstream applications, our MDF needs to be turned into multi-object 3D meshes that preserve their non-intersecting nature. Doing this for standard SDFs usually relies on isosurface extraction algorithms such as Marching Cubes [34, 30], Marching Tetrahedra [5, 55], or Dual Contouring [24]. When dealing with multiple SDFs, these algorithms are often applied independently to each SDF, and the resulting meshes are interpreted as lying in the same physical space, with potential intersections. A partial exception to this is Dual Contouring, for which a multi-object-aware version exists [44], but we do not know of any readily accessible implementation of this algorithm.

Fortunately, this is not an issue thanks to Theorem 2 that states that the linear interpolation of an MDF is still an MDF. This guarantees that, when the MDF constraint is satisfied at the grid points, it is possible to extract intersection-free meshes using linearly interpolated meshing algorithms, including their adaptive variations [46], on each SDF independently, as we prove in Appendix A.2.5. In other words, we can use some of the existing meshing methods as they are, on each MDF component independently.

4 Experiments

Our approach is designed to be generic and usable in conjunction with any signed distance function. To demonstrate this, we use it to impose non-intersection constraints on three modern multi-object modeling techniques [64, 7, 52], some of which come with their own approaches to addressing the

Data	Model	MCD ($\times 10^{-4}$) ↓	MNC ↑	F1 (0.01) ↑	IoU ↑	IV ↓
TS-Lung	Vanilla MedTet	5.462 ± 0.118	96.729 ± 0.059	99.601 ± 0.008	93.737 ± 0.038	$(2.755 \pm 0.767) \times 10^{-4}$
	S2MDF (QP) - PP	5.800 ± 0.314	96.766 ± 0.004	99.575 ± 0.035	94.011 ± 0.850	$(1.190 \pm 0.407) \times 10^{-6}$
	S2MDF (QP) - Train	5.357 ± 0.309	96.791 ± 0.019	99.594 ± 0.007	94.255 ± 0.033	$(1.907 \pm 0.444) \times 10^{-7}$
	S2MDF (Shift-All) - PP	5.600 ± 0.289	96.708 ± 0.023	99.579 ± 0.036	94.000 ± 0.207	$(1.132 \pm 0.245) \times 10^{-6}$
	S2MDF (Shift-All) - Train	5.394 ± 0.056	96.788 ± 0.055	99.620 ± 0.010	93.863 ± 0.397	$(1.391 \pm 0.934) \times 10^{-7}$
MM-WHS (Heart)	Vanilla MedTet	23.863 ± 1.330	94.333 ± 0.157	97.470 ± 0.142	83.580 ± 0.473	$(3.445 \pm 1.390) \times 10^{-4}$
	S2MDF (QP) - PP	29.420 ± 4.430	94.293 ± 0.062	97.367 ± 0.073	83.591 ± 0.081	$(8.446 \pm 3.149) \times 10^{-11}$
	S2MDF (QP) - Train	24.796 ± 0.670	94.386 ± 0.106	97.465 ± 0.308	83.544 ± 0.481	$(2.674 \pm 3.486) \times 10^{-9}$
	S2MDF (Shift-All) - PP	31.099 ± 7.072	94.316 ± 0.151	97.297 ± 0.229	83.694 ± 0.375	$(9.888 \pm 1.793) \times 10^{-12}$
	S2MDF (Shift-All) - Train	23.094 ± 0.331	94.407 ± 0.089	97.556 ± 0.173	83.901 ± 0.039	$(4.635 \pm 5.606) \times 10^{-11}$

Table 1: Experimental results on MedTet. Means are across all organ components and samples, averaged over runs with 3 different random seeds. The IV of our variants is limited by the 16-bit storage of the predicted SDFs (Sec. 4.4).

	MCD ($\times 10^{-2}$) ↓	MNC ↑	F1 (0.01) ↑	IV ↓
Vanilla ObjectSDF++	10.677 ± 5.817	77.953 ± 0.331	66.946 ± 0.097	$(1.391 \pm 0.124) \times 10^{-3}$
S2MDF (QP) - PP	4.145 ± 1.234	78.023 ± 0.221	68.275 ± 0.153	$(1.834 \pm 0.584) \times 10^{-7}$
S2MDF (QP) - Train	-	-	-	-
S2MDF (Shift-All) - PP	3.345 ± 0.315	78.058 ± 0.207	68.325 ± 0.131	$(1.264 \pm 0.237) \times 10^{-7}$
S2MDF (Shift-All) - Train	6.869 ± 0.633	76.411 ± 0.223	66.646 ± 0.666	$(2.278 \pm 0.199) \times 10^{-7}$
S2MDF (Shift-All Skip-Conn.) - Train	3.942 ± 1.000	78.430 ± 0.528	68.623 ± 0.585	$(1.492 \pm 1.925) \times 10^{-6}$

Table 2: ObjectSDF++ [64] trained for object-compositional 3D scene reconstruction on 8 Replica [50] scenes. Mean and standard deviation across 3 runs. QP could not be run for the training setting due to the time complexity of the optimization for a large number of objects.

problem. This allows for comparisons between the different variants of our intersection prevention method and theirs, and shows the effectiveness of our method across different architectures, tasks and domains. We present the methods below.

4.1 Target Methods

MedTet [7] is a medical imaging model designed to perform 3D organ reconstruction from CT/MRI scans using the Deep Marching Tetrahedra [47] algorithm. The model learns to deform and predict SDF values per-vertex on a tetrahedral grid, and then extracts surfaces using Marching Tetrahedra. Losses can be applied both on SDF predictions and the extracted mesh thanks to the surface extraction being differentiable. It does not include any intersection-mitigating mechanisms. We evaluate MedTet in Tab. 1 and Fig. 4 on two datasets:

MM-WHS [77]: a dataset for whole heart segmentation, containing 20 annotated CT samples with 7 heart components – left ventricle, right ventricle, left atrium, right atrium blood cavities, left ventricle myocardium, aorta and pulmonary artery.

TS-Lung [61]: the lung component of the TotalSegmentator dataset, containing 318 lung CT scans with 5 parts: 2 lobes of the left lung and 3 lobes of the right lung. While lung segmentation is considered a simple task with many approaches being able to achieve very high Dice scores [56], we find this dataset useful due to the large number of samples present and the possibility of multi-way intersections.

ObjectSDF++ [64] is a neural implicit framework designed for high-fidelity, object-compositional 3D scene reconstruction. It utilises an occlusion-aware opacity rendering to reconstruct a scene as a collection of distinct, individual objects alongside a separate background. It includes an intersection-mitigating loss which substantially degrades the method if not used [64], so we do not ablate it. We evaluate the model for scene reconstruction using the Replica [50] dataset, adapted from MonoSDF [69]. Our evaluation covers 8 distinct scenes and is carried out as in the original paper, with each scene comprising dozens of individual objects, including the background. The ground-truth meshes used to compute our evaluation metrics are extracted with the vMAP framework [27]. We extract the meshes with Marching Cubes at resolution 256. The results are presented in Tab. 2 and Fig. 2, excluding the IoU metric which cannot be computed in this setting due to non-watertight ground-truth meshes.

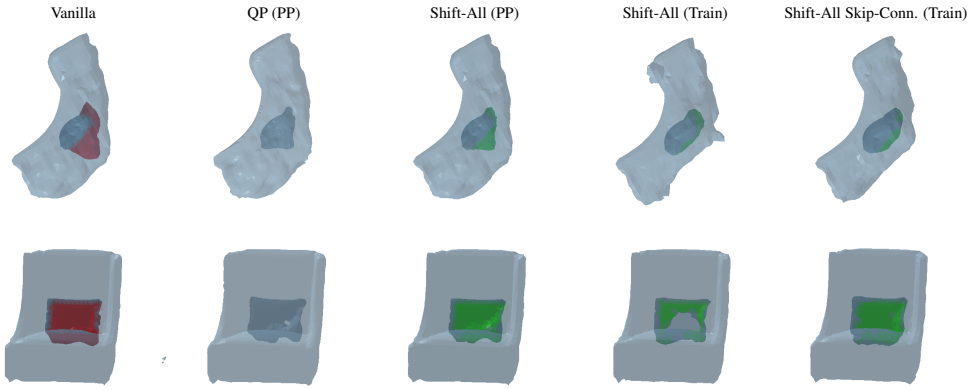


Figure 2: Transversal (top) and frontal (bottom) views of a chair (light grey) and a pillow (dark grey) from Room 0 of the Replica dataset, reconstructed under five scenarios for ObjectSDF++ [64], illustrating inter-object intersections in red and contacts in green. The vanilla version has significant intersections, while the constrained versions do not.

Data	Model	MCD ($\times 10^{-4}$) ↓	MNC ↑	F1 (0.01) ↑	IoU ↑	IV ↓
Chairs	Vanilla PartSDF	4.331 ± 0.690	95.119 ± 0.015	96.731 ± 0.044	85.630 ± 0.101	$(1.876 \pm 0.062) \times 10^{-3}$
	S2MDF (QP) - PP	2.126 ± 0.165	94.773 ± 0.014	97.902 ± 0.042	89.151 ± 0.107	$(6.157 \pm 1.394) \times 10^{-9}$
	S2MDF (QP) - Train	2.189 ± 0.364	94.885 ± 0.078	98.047 ± 0.031	89.289 ± 0.126	$(7.000 \pm 0.816) \times 10^{-9}$
	S2MDF (Shift-All) - PP	1.973 ± 0.166	94.824 ± 0.008	98.005 ± 0.022	89.173 ± 0.104	$(5.194 \pm 1.040) \times 10^{-9}$
	S2MDF (Shift-All) - Train	2.120 ± 0.117	94.823 ± 0.032	98.039 ± 0.017	89.297 ± 0.262	$(5.665 \pm 2.370) \times 10^{-9}$
Mixers	Vanilla PartSDF	1.386 ± 0.090	88.013 ± 0.074	98.441 ± 0.107	76.667 ± 0.168	$(4.317 \pm 0.221) \times 10^{-5}$
	S2MDF (QP) - PP	1.362 ± 0.102	88.000 ± 0.135	98.441 ± 0.107	75.769 ± 0.892	$(1.018 \pm 0.349) \times 10^{-7}$
	S2MDF (QP) - Train	1.247 ± 0.032	88.147 ± 0.090	98.595 ± 0.026	76.648 ± 0.170	$(1.239 \pm 0.831) \times 10^{-8}$
	S2MDF (Shift-All) - PP	1.362 ± 0.102	88.000 ± 0.135	98.441 ± 0.107	75.774 ± 0.887	$(1.020 \pm 0.349) \times 10^{-7}$
	S2MDF (Shift-All) - Train	1.234 ± 0.137	88.175 ± 0.045	98.593 ± 0.153	76.595 ± 0.201	$(1.680 \pm 0.851) \times 10^{-8}$

Table 3: Shape reconstruction metrics for PartSDF on 100 chairs and mixers from the test set. Mean and standard deviation across 3 runs.

PartSDF [52] is an SDF-based method for part-aware shape representation, with applications in shape reconstruction, generation and optimization. It represents shapes as a collection of parts, each represented by a separate SDF, and includes a loss term to mitigate intersections between parts akin to [64, 29]. We follow the experimental setup of the original paper for shape reconstruction and train one model on 100 chairs from PartNet [38], with 8 parts each, and a separate one on 100 mixers from [57], with 4 parts each. We then evaluate the models by freezing the weights and optimizing the latent codes on 100 chairs and 100 mixers from the test set, and we extract them with Marching Cubes at resolution 128. Notice that, while the mixers are made of watertight parts, the chairs are not, which serves as an interesting test for the robustness of our approaches to a scenario in which supervision is missing specifically at the intersection between parts. We report the results in Tab. 3 and Fig. 3, with additional results in the Appendix.

4.2 Experimental Protocol

In all three cases discussed above, we compare the original version of the method against four variants of our S2MDF layer: S2MDF (QP) - PP applies the QP-based layer only at meshing time, effectively as a post-processing step for the SDF values; S2MDF (QP) - Train appends the QP-based layer to the end of the network, affecting both training and inference; S2MDF (Shift-All) - PP and S2MDF (Shift-All) - Train apply the Shift-All approach in the same way, respectively as a post-processing step and during training.

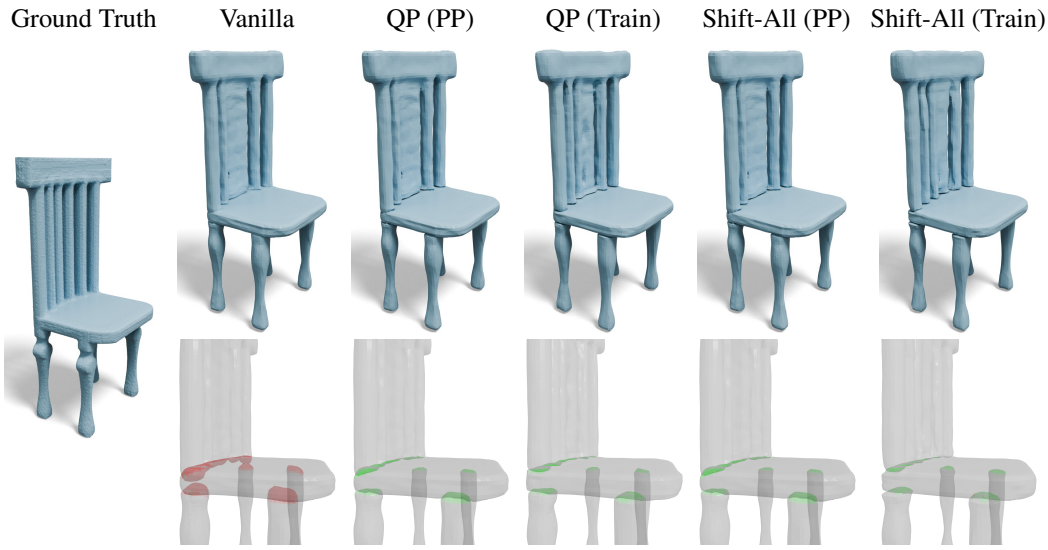


Figure 3: PartSDF [52] on PartNet [38] chairs: the top row shows ray-traced renderings, the bottom row visualizes intersections. Intersecting surfaces are in red, while touching surfaces are in green.

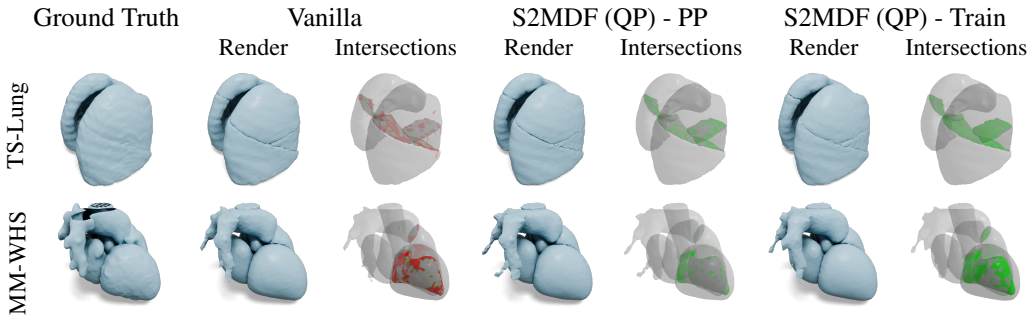


Figure 4: **Qualitative Comparisons.** We compare the outputs of the vanilla approach to our QP variants on MedTet. The vanilla variants have significant intersections (red), while the constrained ones have only contact regions (green).

4.3 Evaluation Metrics

To evaluate the results, we rely on metrics commonly used in 3D deep learning, along with one that computes intersection volume between objects and is key for assessing the validity of our method. When possible, we base them on point cloud to facet matches rather than point cloud to point cloud for increased accuracy, and thus we add a **Mesh** prefix to the metric.

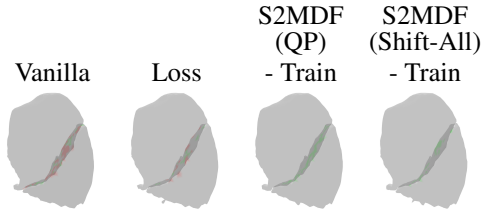


Figure 5: Qualitative visualization on left-lung reconstructions. The loss term reduces the intersections compared to vanilla, but is far from completely satisfying non-intersection constraints.

Model	MCD ($\times 10^{-4}$) ↓	IV ↓
Vanilla	6.270 ± 0.201	$(2.282 \pm 1.060) \times 10^{-5}$
Loss	6.469 ± 0.372	$(2.069 \pm 0.772) \times 10^{-5}$
S2MDF (QP) - Train	6.096 ± 0.343	$(1.352 \pm 0.544) \times 10^{-9}$
S2MDF (Shift-All) - Train	6.469 ± 0.479	$(7.412 \pm 10.603) \times 10^{-12}$

Table 4: Metric comparison between the application of a soft intersection loss and our S2MDF layer during training for MedTet on the left lung. The soft loss only marginally decreases the intersection volume, while our approach reduces it by several orders of magnitude.

Mesh Chamfer Distance. When measuring Chamfer Distance between two meshes M_1 and M_2, points are uniformly sampled on the surface of the meshes: $P_1 \sim \mathcal{U}(M_1)$ and $P_2 \sim \mathcal{U}(M_2)$. However, this formulation is biased for a finite number of points (it is an overestimate), and inaccurate compared to finding the distance from a point to a mesh directly [49]. We therefore define Mesh Chamfer Distance using sampled point to mesh distances rather than point to point, as

$$d_{MCD}(M_1, M_2) = \frac{1}{|P_1|} \sum_{x \in P_1} \min_{y_m \in M_2} \|x - y_m\|_2^2 + \frac{1}{|P_2|} \sum_{y \in P_2} \min_{x_m \in M_1} \|x_m - y\|_2^2 \tag{7}$$

Note that x_m and y_m can be anywhere on the mesh: a vertex, edge, or on the surface of any of its constituent triangles. This distance can be calculated efficiently.

Mesh Normal Consistency. Measures agreement between the normals of two point clouds with cosine similarity, using nearest neighbor matches as with the Chamfer Distance. As with the Mesh Chamfer Distance, we find the nearest points directly on the target mesh and use interpolated vertex normals rather than per-triangle flat normals.

F1 score (τ). As done in ObjectSDF++ [64], measures the ratio of points from M_1 at distance less than τ from M_2. Precision is from predicted mesh to ground truth, recall is vice-versa, and F1 score is the harmonic mean of the two, being bidirectional. We once again measure distances with point-to-mesh approaches.

Mesh IoU. Measures the overall alignment between two meshes, as their volumetric intersection over union: $\frac{\text{vol}(M_1 \cap M_2)}{\text{vol}(M_1 \cup M_2)}$. We measure this approximately using a discrete voxel grid on the union bounding box of the two meshes.

Intersection Volume. The volume of the intersection between two meshes, M_1 and M_2. This is calculated between predicted meshes for different parts of the same object or scene. It is always 0 in the ground truth.

4.4 Comparative Results

We execute experiments on single NVIDIA A100 or V100 GPUs, ensuring consistency when measuring times. We refer to the original papers for more details on the setup and the compute requirements. We report the results of our experiments in Tabs. 1 to 3 and Figs. 2 to 4 for MedTet, ObjectSDF++ and PartSDF respectively. We run each experiment 3 times, to compute mean and standard deviations. Across all results, we observe that both QP and Shift-All reduce intersections by several orders of magnitude with respect to the original methods, including PartSDF and ObjectSDF++ which already include a loss term to mitigate them. The reduction in intersections is qualitatively visible in Figs. 2 to 4, where the red color indicates intersecting surfaces and the green color indicates touching surfaces. In terms of reconstruction quality, we observe that the metrics are quite noisy, but overall remain close to the original methods in mean and variance, with some exceptions. In particular, we notice that our S2MDF methods tend to perform slightly better when used during training than only as a post-processing step, as the network can adapt to the constraint during training. We also notice that the Shift-All approach performs as well as the QP-based one during post-processing and slightly better during training, while being much faster. For few objects, like in the case of PartSDF mixers which only have 4 parts, QP does not add significant overhead when used as post-processing, however it increases the training time from 3h30m to 5h40m when used during training. For more complex objects, like the PartSDF chairs with 8 parts, the overhead of the QP-based approach is much more significant, increasing the training time from 4 to 9 hours for 2000 epochs, and the inference time from 20 to 60 seconds per object. On ObjectSDF++, where there are several dozens of objects per scene, the overhead of the QP-based approach makes it infeasible to use it during training. On the other hand, the Shift-All approach did not add any measurable overhead during training or post-processing in any of our tests, across all methods. As such, we recommend the use of S2MDF with the Shift-All approach.

Comparison against loss-based solutions To compare our method against loss-based intersection mitigations we run an experiment on TS-Lung for MedTet using the left lung only, where we apply an additional loss term to penalize intersections that does not exist in the base model: $\mathcal{L}_{inter} = \frac{1}{K-1} \sum_{k \neq min} \max(0, -(SDF_{min} + SDF_k))$. This equation penalizes overlapping geometry by enforcing that if a point is deeply inside one object ($SDF_{min} < 0$), it must be sufficiently outside

all other objects ($SDF_k > 0$) so that their sum remains non-negative. This loss is the same as the one applied in ObjectSDF++ and similar to the one in PartSDF and [29]. We show that the improvement offered by this loss in terms of intersection volume is marginal in Fig. 5 and Tab. 4, whereas our approach reduces it by several orders of magnitude. Our approach remains compatible with non-intersection losses, as they can help regularizing the underlying field, but it still vastly improves the results. We refer to Appendix A.1.2.3 for an ablation on the non-intersection loss in PartSDF, showing similar results.

Bringing intersections to zero The intersection volumes that remain after applying S2MDF come from numerical precision. Due to our use of mixed-precision training, the predicted SDFs are stored in 16-bit floating point, which limits how small the IV in Tab. 1 can get. On TS-Lung, storing them in 32-bit instead brings the IV down to around 10^{-22} with Shift-All and 10^{-12} with QP, where the latter is limited by the maximum number of solver iterations we allow. To show that our approach is capable of fully eliminating intersections, we perform a small experiment comparing $\epsilon = 0$ and $\epsilon = 10^{-4}$ for the QP-based solution where we set $d_i + d_j \geq \epsilon$. We note that the first one evaluates to around 1.2×10^{-8} in terms of intersection volume, while the second one evaluates to exactly 0, showing that setting a small threshold effectively eliminates intersections, which can otherwise appear due to numerical errors.

Eikonal loss Regularization losses such as the eikonal loss are commonly used in SDF-based methods to encourage smoothness and regularity of the learned fields. We perform an ablation study on ObjectSDF++ to evaluate the effect of applying the Shift-All approach during training with and without a skip connection that allows the eikonal loss to be computed on the original network outputs. We observe that our layer performs overall similarly to the original method in both cases, but produces a noticeably lower MCD when the eikonal loss is applied to the original network output. We conjecture that regularization losses are better suited for the original field, as they help the network learn a naturally better field before applying the constraint layer. Thus, we suggest to apply regularization losses to the original field.

5 Limitations

Our approach guarantees an intersection-free signed-distance field. However, the extracted meshes can only be intersection-free if they are extracted by linear interpolation of the field. This holds for the commonly used Marching Tetrahedra and Marching Cubes. For methods where vertex placement is learned rather than linear, such as FlexiCubes [48] (Appendix A.3.1), our guarantee does not hold. Moreover, when S2MDF is only applied as a post-processing step, overlapping surfaces are both moved to where their fields cross, which may not be how the real geometry behaves. When it is applied during training instead, the network can adapt to the constraint. We analyze these limitations in more detail in Appendix A.3, together with some edge cases of Marching Cubes.

6 Conclusion

In this work, we have introduced a constrained SDF formulation to guarantee intersection-free multi-object implicit surface representations, and we have proposed S2MDF, a fast, plug-and-play module that can be applied to any object-compositional SDF-based representation to enforce the constraint, whose output remains compatible with linearly-interpolated meshing techniques. We have shown that our method eliminates intersections in theory and reduces them to numerical error in practice, without degrading the accuracy of the reconstructions and sometimes even improving it, which existing loss-based solutions cannot do. While we can address the numerical issue by adding a small margin to our constraint, an even more principled solution would be to design meshing algorithms that take into account multiple objects and can handle them correctly, which is a topic for future work.

Acknowledgments and Disclosure of Funding

This work was funded in part by the Swiss National Science Foundation.

References

[1] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning Representations and Generative Models for 3D Point Clouds. In *International Conference on Machine Learning*, pages 40–49, 2018.

[2] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International conference on machine learning*, pages 136–145. PMLR, 2017.

[3] Antoine Bambade, Fabian Schramm, Adrien Taylor, and Justin Carpentier. Leveraging augmented-lagrangian techniques for differentiating over infeasible quadratic programs in machine learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[5] S.L. Chan and E.O. Purisima. A new tetrahedral tesselation scheme for isosurface generation. *Computers & Graphics*, 22(1):83–90, 1998.

[6] Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. In *Conference on Computer Vision and Pattern Recognition*, 2025.

[7] Y. Chen, J. Yang, D. S. Mercadier, H. Le, and P. Fua. Medtet: An Online Motion Model for 4D Heart Reconstruction. In *arXiv Preprint*, 2024.

[8] Z. Chen, A. Tagliasacchi, T. Funkhouser, and H. Zhang. Neural Dual Contouring. In *arXiv Preprint*, 2022.

[9] Z. Chen and H. Zhang. Learning Implicit Fields for Generative Shape Modeling. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[10] Z. Chen and H. Zhang. Neural Marching Cubes. In *ACM Transactions on Graphics (Special Issue of SIGGRAPH Asia)*, 2021.

[11] J. Chibane, A. Mir, and G. Pons-Moll. Neural Unsigned Distance Fields for Implicit Function Learning. In *Advances in Neural Information Processing Systems*, 2020.

[12] C. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3D-R2n2: A Unified Approach for Single and Multi-View 3D Object Reconstruction. In *European Conference on Computer Vision*, pages 628–644, 2016.

[13] A. Dai, C. Qi, and M. Nießner. Shape Completion Using 3D-Encoder-Predictor CNNs and Shape Synthesis. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[14] B. Deng, S. Kulal, Z. Deng, C. Deng, Y. Tian, and J. Wu. Unsupervised Learning of Shape Programs with Repeatable Implicit Parts. In *Advances in Neural Information Processing Systems*, 2022.

[15] A. Doi and A. Koide. An Efficient Method of Triangulating Equivalued Surfaces by Using Tetrahedral Cells. *Transactopms on Information and Systems*, 74(1):214–224, 1991.

[16] C. Dumery, A. Fan, R. Li, N. Talabot, and P. Fua. Enforcing View-Consistency in Class-Agnostic 3D Segmentation Fields. In *Conference on Computer Vision and Pattern Recognition*, June 2025.

[17] H. Fan, H. Su, and L. Guibas. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[18] Wenhang Ge, Tao Hu, Haoyu Zhao, Shu Liu, and Ying-Cong Chen. Ref-neus: Ambiguity-reduced neural implicit surface learning for multi-view reconstruction with reflection. *arXiv Preprint*, 2023.

[19] T. Groueix, M. Fisher, V. Kim, B. Russell, and M. Aubry. Atlasnet: A Papier-Mâché Approach to Learning 3D Surface Generation. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[20] B. Guillard, E. Remelli, A. Lukoianov, S. Richter, T. Bagautdinov, P. Baque, and P. Fua. Deepmesh: Differentiable Iso-Surface Extraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(11):7072–7087, 2024.

[21] B. Guillard, F. Stella, and P. Fua. Meshudf: Fast and Differentiable Meshing of Unsigned Distance Field Networks. In *European Conference on Computer Vision*, pages 576–592, 2022.

[22] Z. Hao, H. Averbuch-Elor, N. Snavely, and S. Belongie. Dualsdf: Semantic Shape Manipulation Using a Two-Level Representation. In *Conference on Computer Vision and Pattern Recognition*, pages 7631–7641, 2020.

[23] Y. Jiang, D. Ji, Z. Han, and M. Zwicker. Sdfdiff: Differentiable Rendering of Signed Distance Fields for 3D Shape Optimization. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[24] T. Ju, F. Losasso, S. Schaefer, and J. Warren. Dual Contouring of Hermite Data. In *ACM SIGGRAPH*, 2002.

[25] A. Kanazawa, S. Tulsiani, A. Efros, and J. Malik. Learning Category-Specific Mesh Reconstruction from Image Collections. In *Conference on Computer Vision and Pattern Recognition*, 2018.

[26] K. Karunratanakul, J. Yang, Y. Zhang, M. Black, K. Muandet, and S. Tang. Grasping Field: Learning Implicit Representations for Human Grasps. In *International Conference on 3D Vision*, 2020.

[27] Xin Kong, Shikun Liu, Marwan Taher, and Andrew J Davison. vmap: Vectorised object mapping for neural field slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 952–961, 2023.

[28] Zeqiang Lai, Yunfei Zhao, Haolin Liu, Zibo Zhao, Qingxiang Lin, Huiwen Shi, Xianghui Yang, Mingxin Yang, Shuhui Yang, Yifei Feng, et al. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details. *arXiv Preprint*, 2025.

[29] H. Le, J. Xu, N. Talabot, J. Yang, and P. Fua. Pairwise-Constrained Implicit Functions for 3D Human Heart Modeling. In *Conference on Medical Image Computing and Computer Assisted Intervention*, 2025.

[30] T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares. Efficient Implementation of Marching Cubes' Cases with Topological Guarantees. In *Journal of Graphics Tools*, 2003.

[31] Y.-T. Liu, L. Wang, J. Yang, W. Chen, X. Meng, B. Yang, and L. Gao. NeUDF: Leaning Neural Unsigned Distance Fields with Volume Rendering. In *Conference on Computer Vision and Pattern Recognition*, 2023.

[32] Yuchun Liu, Benjamin Planche, Meng Zheng, Zhongpai Gao, Pierre Sibut-Bourde, Fan Yang, Terrence Chen, and Ziyan Wu. Implicit modeling of non-rigid objects with cross-category signals, 2023.

[33] X. Long, C. Lin, L. Liu, Y. Liu, P. Wang, C. Theobalt, T. Komura, and W. Wang. Neuraludf: Learning Unsigned Distance Fields for Multi-View Reconstruction of Surfaces with Arbitrary Topologies. In *Conference on Computer Vision and Pattern Recognition*, 2022.

[34] W.E. Lorensen and H.E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *ACM SIGGRAPH*, pages 163–169, 1987.

[35] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy Networks: Learning 3D Reconstruction in Function Space. In *Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019.

[36] Ben Mildenhall, S. P. P., M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*, 2020.

[37] P. Mittal, Y.-C. Cheng, M. Singh, and S. Tulsiani. AutoSDF: Shape Priors for 3D Completion, Reconstruction and Generation. In *Conference on Computer Vision and Pattern Recognition*, 2022.

[38] K. Mo, S. Zhu A. X., Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su. PartNet: A Large-Scale Benchmark for Fine-Grained and Hierarchical Part-Level 3D Object Understanding. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[39] M. Niemeyer and A. Geiger. GIRAFFE: Representing Scenes As Compositional Generative Neural Feature Fields. In *Conference on Computer Vision and Pattern Recognition*, 2021.

[40] J. Pan and K. Jia. Deep Mesh Reconstruction from Single RGB Images via Topology Modification Networks. In *International Conference on Computer Vision*, 2019.

[41] J. J. Park, P. Florence, J. Straub, R. A. Newcombe, and S. Lovegrove. Deepsdf: Learning Continuous Signed Distance Functions for Shape Representation. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[42] S. Peng, C. Jiang, Y. Liao, M. Niemeyer, M. Pollefeys, and A. Geiger. Shape as Points: A Differentiable Poisson Solver. In *Advances in Neural Information Processing Systems*, 2021.

[43] D. Petrov, M. Gadelha, R. Měch, and E. Kalogerakis. ANISE: Assembly-Based Neural Implicit Surface Reconstruction. In *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[44] Tanweer Rashid, Sharmin Sultana, Mallar Chakravarty, and Michel Albert Audette. Atlas-based shared-boundary deformable multi-surface models through multi-material and two-manifold dual contouring. *Information*, 14(4), 2023.

[45] E. Remelli, A. Lukoianov, S. Richter, B. Guillard, T. Bagautdinov, P. Baque, and P. Fua. Meshsdf: Differentiable Iso-Surface Extraction. In *Advances in Neural Information Processing Systems*, 2020.

[46] D. Ren, H. Shi, J. Zheng, and J. Cai. *Mcgrids: Monte Carlo-Driven Adaptive Grids For iso-Surface Extraction*, page 127144. Springer Nature Switzerland, September 2024.

[47] T. Shen, J. Gao, K. Yin, M.-Y. Liu, and S. Fidler. Deep Marching Tetrahedra: A Hybrid Representation for High-Resolution 3D Shape Synthesis. In *Advances in Neural Information Processing Systems*, 2021.

[48] T. Shen, J. Munkberg, J. Hasselgren, K. Yin, Z. Wang, W. Chen, Z. Gojcic, S. Fidler, N. Sharp, and J. Gao. Flexible Isosurface Extraction for Gradient-Based Mesh Optimization. *ACM Trans. Graph.*, 42(4), July 2023.

[49] F. Stella, N. Talabot, H. Le, and P. Fua. High Resolution UDF Meshing via Iterative Networks. In *Advances in Neural Information Processing Systems*, 2025.

[50] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.

[51] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler. Neural Geometric Level of Detail: Real-Time Rendering with Implicit 3D Shapes. In *Conference on Computer Vision and Pattern Recognition*, 2021.

[52] Nicolas Talabot, Olivier Clerc, Arda Cinar Demirtas, Alexis Goujon, Hieu Le, Doruk Oner, and Pascal Fua. PartSDF: Part-Based Implicit Neural Representation for Composite 3D Shape Parametrization and Optimization. *Transactions on Machine Learning Research*, October 2025.

[53] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, J. Kerr, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. McAllister, and A. Kanazawa. Nerfstudio: A Modular Framework for Neural Radiance Field Development. In *ACM SIGGRAPH*, 2023.

[54] K. Tertikas, D. Paschalidou, B. Pan J. J.and Park, M. A. Uy, I. Emiris, Y. Avrithis L., and Guibas. Generating Part-Aware Editable 3D Shapes Without 3D Supervision. In *Conference on Computer Vision and Pattern Recognition*, 2023.

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

[56] Fuat Turk and Mahmut Kılıçaslan. Lung image segmentation with improved u-net, v-net and seg-net techniques. *PeerJ Computer Science*, 11:e2700, 2025.

[57] S. Vasu, N. Talabot, A. Lukoianov, P. Baque, J. Donier, and P. Fua. Hybridsdf: Combining Free Form Shapes and Geometric Primitives for Effective Shape Manipulation. In *International Conference on 3D Vision*, 2022.

[58] B. Wang, C. Lu, and Y. Bo. DM-NeRF: 3D Scene Geometry Decomposition and Manipulation from 2D Images. *International Conference on Learning Representations*, 2023.

[59] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y. Jiang. Pixel2mesh: Generating 3D Mesh Models from Single RGB Images. In *European Conference on Computer Vision*, 2018.

[60] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang. Neus: Learning Neural Implicit Surfaces by Volume Rendering for Multi-View Reconstruction. In *Advances in Neural Information Processing Systems*, 2021.

[61] Jakob Wasserthal, Hanns-Christian Breit, Manfred T Meyer, Maurice Pradella, Daniel Hinck, Alexander W Sauter, Tobias Heye, Daniel T Boll, Joshy Cyriac, Shan Yang, et al. Totalsegmentator: Robust Segmentation of 104 Anatomic Structures in CT Images. *Radiology: Artificial Intelligence*, 2023.

[62] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. In *Advances in Neural Information Processing Systems*, pages 82–90, 2016.

[63] Qianyi Wu, Xian Liu, Yuedong Chen, Kejie Li, Chuanxia Zheng, Jianfei Cai, and Jianmin Zheng. Object-compositional neural implicit surfaces. In *European Conference on Computer Vision*, 2022.

[64] Qianyi Wu, Kaisiyuan Wang, Kejie Li, Jianmin Zheng, and Jianfei Cai. Objectsdf++: Improved object-compositional neural implicit surfaces. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21764–21774, 2023.

[65] R. Wu, Y. Zhuang, K. Xu, H. Zhang, and B. Chen. PQ-NET: A Generative Part Seq2seq Network for 3D Shapes. In *Conference on Computer Vision and Pattern Recognition*, 2020.

[66] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D Shapenets: A Deep Representation for Volumetric Shapes. In *Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

[67] C. Xie, K. Park, R. Martin-Brualla, and M. Brown. FiG-NeRF: Figure-Ground Neural Radiance Fields for 3D Object Category Modelling, 2021.

[68] Y. Yang, C. Feng, Y. Shen, and D. Tian. Foldingnet: Point Cloud Auto-Encoder via Deep Grid Deformation. In *Conference on Computer Vision and Pattern Recognition*, June 2018.

[69] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[70] X. Zeng, A. Vahdat, F. Williams, Z. Gojcic, O. Litany, S. Fidler, and K. Kreis. LION: Latent Point Diffusion Models for 3D Shape Generation. In *Advances in Neural Information Processing Systems*, 2022.

[71] B. Zhang, M. Nießner, and P. Wonka. 3DILG: Irregular Latent Grids for 3D Generative Modeling. In *Advances in Neural Information Processing Systems*, 2022.

[72] B. Zhang, J. Tang, M. Nießner, and P. Wonka. 3DShape2VecSet: A 3D Shape Representation for Neural Fields and Generative Diffusion Models. In *ACM SIGGRAPH*, 2023.

[73] C. Zhang, G. Lin, L. Yang, X. Li, T. Komura, S. Schaefer, J. Keyser, and W. Wang. Surface Extraction from Neural Unsigned Distance Fields. In *International Conference on Computer Vision*, pages 0000–0000, October 2023.

[74] Congyi Zhang, Mohamed Elgharib, Gereon Fox, Min Gu, Christian Theobalt, and Wenping Wang. An implicit parametric morphable dental model. *ACM Trans. Graph.*, 41(6), nov 2022.

[75] Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics*, 2024.

[76] S. Zhi, T. Laidlow, S. Leutenegger, and A. Davison. In-Place Scene Labelling and Understanding with Implicit Scene Representation. In *International Conference on Computer Vision*, 2021.

[77] X. Zhuang. Multivariate Mixture Model for Myocardial Segmentation Combining Multi-Source Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):2933–2946, 2018.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We back the claims with theorems and experimental results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss them in Sec. 5, with a more detailed analysis in Appendix A.3.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We provided proofs and statements to the best of our knowledge.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We describe in detail the experimental settings and the characteristics of our approach.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Code will be released upon publication, however the approach consists of a simple mapping that can be implemented in a few lines of code, and the data and methods used are all publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Experiments are fully described and follow the descriptions of the original papers of the methods we use.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report standard deviation across multiple runs when possible, otherwise we report the standard deviation across the test set.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We disclose GPU type, number and execution time.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: Our method is a fairly theoretical method with applications in surface reconstruction domains, but we do not see direct significant societal impacts as of now.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification:

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

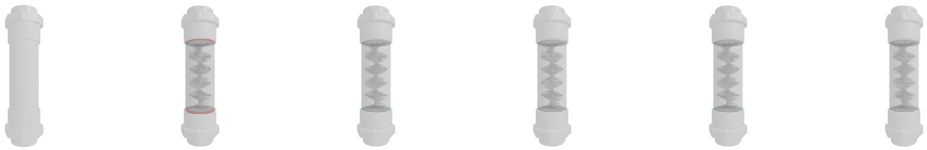
Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification:

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.



Ground Truth | Vanilla | S2MDF (QP) - PP | S2MDF (QP) - Train | S2MDF (Shift-All) - PP | S2MDF (Shift-All) - Train

Figure A.1: PartSDF [52]: visualization of intersections. Intersecting surfaces are in red, while touching surfaces are in green.

A.1 Additional Results

We show here additional results and figures that were not included in the main text. In Fig. A.1, we show visualizations of the intersections for the PartSDF [52] experiment on the Mixers category. Similar conclusions as the main text apply.

A.1.1 An alternative solution

The solutions we presented are not the only possible ways to enforce the MDF constraint. A simple but more heuristic approach consists in always taking the minimum SDF value across all objects whenever the MDF constraint is violated. This can be expressed as a projection operator $\text{Min}(\mathbf{u})$, which takes as input the vector of SDF values $\mathbf{u} \in \mathbb{R}^K$ for K objects at a given point and outputs a projected vector that satisfies the MDF constraint. The projection operator can be defined as follows:

$$\text{Min}(\mathbf{u}) = \begin{cases} \mathbf{u} & \text{if } \min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}) \geq 0 \ , \\ \min^{(1)}(\mathbf{u}) \cdot \mathbf{1}_{\text{argmin}^{(1)}(\mathbf{u})} & \text{otherwise.} \end{cases} \tag{8}$$

Where $\mathbf{1}_{\text{argmin}^{(1)}(\mathbf{u})} \in \mathbb{R}^K$ is a vector with 1 at the index of the minimum entry of $\mathbf{u}$ and -1 elsewhere. This makes the $Min(\mathbf{u})$ vector negative only at the index of the minimum entry of $\mathbf{u}$, and positive (with an equal but opposite value) at all other entries, thus satisfying the MDF constraint.

This approach works well in practice, but it does not supersede the QP and Shift-All approaches we presented in the main text, which are more principled and provide better results. We include a comparison of this approach with the other methods in Tabs. A.1 and A.2.

A.1.2 Additional results on PartSDF

A.1.2.1 Inference-time optimization with S2MDF

Contrary to the other methods we evaluate, PartSDF [52] is not limited to applying our S2MDF module either during training or as a post-processing step for the field at meshing time. Instead, it can also be applied during the inference-time optimization process that PartSDF performs to refine the geometry of each part, freezing the weights and optimizing the latent code. In Tabs. A.1 and A.2, we present quantitative results for the same PartSDF [52] experiment as in Tab. 3, additionally comparing the application of S2MDF during the inference-time optimization ("Optim"). We observe that, while not disrupting the optimization process, applying S2MDF during the optimization does not provide advantages over applying it during training or as a simple post-processing step, and it is slightly outperformed by both in most occasions. This is reasonable, as the network weights have been trained without such layer, and the resulting latent space may not be optimal to allow for a good optimization with the S2MDF module.

A.1.2.2 Artifacts introduced by S2MDF

In a few situations, we observe that the application of S2MDF can introduce artifacts in the resulting meshes. This is quantitatively hinted at by the slightly worse MNC scores in the PartSDF chairs experiments (Tabs. 3 and A.1). We noticed that, when S2MDF is applied, some of the junctions

	MCD ($\times 10^{-4}$) ↓	MNC ↑	F1 (0.001) ↑	IoU ↑	IV ↓
Vanilla PartSDF	4.331 ± 0.690	95.119 ± 0.015	96.731 ± 0.044	85.630 ± 0.101	$(1.876 \pm 0.062) \times 10^{-3}$
S2MDF (Min) - PP	2.017 ± 0.165	94.728 ± 0.011	97.933 ± 0.025	89.172 ± 0.106	$(1.500 \pm 0.032) \times 10^{-7}$
S2MDF (Shift-All) - PP	1.973 ± 0.166	94.824 ± 0.008	98.005 ± 0.022	89.173 ± 0.104	$(5.194 \pm 1.040) \times 10^{-9}$
S2MDF (QP) - PP	2.126 ± 0.165	94.773 ± 0.014	97.902 ± 0.042	89.151 ± 0.107	$(6.157 \pm 1.394) \times 10^{-9}$
S2MDF (Min) - Optim	2.200 ± 0.160	94.687 ± 0.016	97.846 ± 0.024	89.013 ± 0.103	$(1.652 \pm 0.064) \times 10^{-7}$
S2MDF (Shift-All) - Optim	2.616 ± 0.576	94.879 ± 0.039	98.116 ± 0.021	89.664 ± 0.048	$(3.194 \pm 0.566) \times 10^{-9}$
S2MDF (QP) - Optim	2.195 ± 0.428	94.842 ± 0.035	98.076 ± 0.011	89.618 ± 0.086	$(8.740 \pm 7.756) \times 10^{-9}$
S2MDF (Min) - Train	2.154 ± 0.137	94.723 ± 0.009	97.896 ± 0.035	89.063 ± 0.155	$(1.467 \pm 0.093) \times 10^{-7}$
S2MDF (Shift-All) - Train	2.120 ± 0.117	94.823 ± 0.032	98.039 ± 0.017	89.297 ± 0.262	$(5.665 \pm 2.370) \times 10^{-9}$
S2MDF (QP) - Train	2.189 ± 0.364	94.885 ± 0.078	98.047 ± 0.031	89.289 ± 0.126	$(7.000 \pm 0.816) \times 10^{-9}$

Table A.1: Shape reconstruction metrics for PartSDF on 100 chairs from the test set. Mean and standard deviation across 3 runs.

	MCD ($\times 10^{-4}$) ↓	MNC ↑	F1 (0.001) ↑	IoU ↑	IV ↓
Vanilla PartSDF	1.386 ± 0.090	88.013 ± 0.074	98.441 ± 0.107	76.667 ± 0.168	$(4.317 \pm 0.221) \times 10^{-5}$
S2MDF (Min) - PP	1.358 ± 0.105	88.113 ± 0.059	98.441 ± 0.107	76.752 ± 0.145	$(1.251 \pm 0.170) \times 10^{-7}$
S2MDF (Shift-All) - PP	1.362 ± 0.102	88.000 ± 0.135	98.441 ± 0.107	75.774 ± 0.887	$(1.020 \pm 0.349) \times 10^{-7}$
S2MDF (QP) - PP	1.362 ± 0.102	88.000 ± 0.135	98.441 ± 0.107	75.769 ± 0.892	$(1.018 \pm 0.349) \times 10^{-7}$
S2MDF (Min) - Optim	1.358 ± 0.107	88.108 ± 0.060	98.440 ± 0.116	76.747 ± 0.143	$(9.715 \pm 2.166) \times 10^{-8}$
S2MDF (Shift-All) - Optim	1.395 ± 0.127	88.066 ± 0.069	98.407 ± 0.146	76.524 ± 0.159	$(1.694 \pm 1.634) \times 10^{-9}$
S2MDF (QP) - Optim	1.397 ± 0.122	88.055 ± 0.077	98.391 ± 0.130	76.517 ± 0.163	$(1.525 \pm 1.822) \times 10^{-9}$
S2MDF (Min) - Train	1.243 ± 0.191	88.190 ± 0.193	98.535 ± 0.203	76.749 ± 0.357	$(1.200 \pm 0.604) \times 10^{-8}$
S2MDF (Shift-All) - Train	1.234 ± 0.137	88.175 ± 0.045	98.593 ± 0.153	76.595 ± 0.201	$(1.680 \pm 0.851) \times 10^{-8}$
S2MDF (QP) - Train	1.247 ± 0.032	88.147 ± 0.090	98.595 ± 0.026	76.648 ± 0.170	$(1.239 \pm 0.831) \times 10^{-8}$

Table A.2: Shape reconstruction metrics for PartSDF on 100 mixers from the test set. Mean and standard deviation across 3 runs.

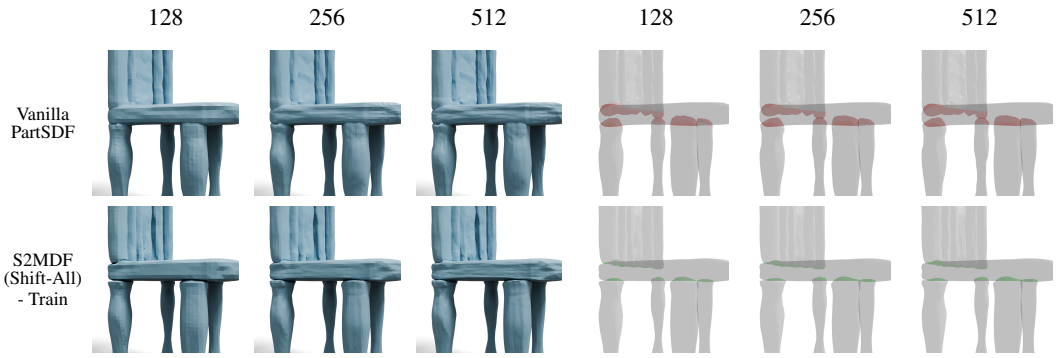
between parts are not as straight as in the vanilla PartSDF meshes. This effect is visible in the junctions between the chair legs and the seat in Fig. A.2, where we show the meshes of a chair at increasing meshing resolutions. The issue appears because the GT parts of the chairs are not watertight: they are missing the junction surfaces, which means that PartSDF lacks supervision in the most delicate regions. This is not an issue in vanilla PartSDF because the legs become elongated, heavily intersecting with the seat (Fig. A.2b), but it can create small artifacts when S2MDF is applied. We did not observe this issue in the Mixers dataset, as well as in the other experiments in the main paper, where the GT parts are watertight and the junctions are well defined.

A.1.2.3 Ablating the non-intersection loss

	MCD ($\times 10^{-4}$) ↓	MNC ↑	F1 (0.001) ↑	IoU ↑	IV ↓
Vanilla PartSDF	4.331 ± 0.690	95.119 ± 0.015	96.731 ± 0.044	85.630 ± 0.101	$(1.876 \pm 0.062) \times 10^{-3}$
Vanilla PartSDF w/o int. loss	40.891 ± 28.217	94.473 ± 0.158	92.493 ± 0.928	77.759 ± 0.898	$(8.746 \pm 3.431) \times 10^{-3}$
S2MDF (Shift-All) - PP	1.973 ± 0.166	94.824 ± 0.008	98.005 ± 0.022	89.173 ± 0.104	$(5.194 \pm 1.040) \times 10^{-9}$
S2MDF (Shift-All) - PP w/o int. loss	3.735 ± 0.459	94.235 ± 0.069	96.579 ± 0.187	86.752 ± 0.151	$(2.868 \pm 0.407) \times 10^{-8}$
S2MDF (Shift-All) - Optim	2.616 ± 0.576	94.879 ± 0.039	98.116 ± 0.021	89.664 ± 0.048	$(3.194 \pm 0.566) \times 10^{-9}$
S2MDF (Shift-All) - Optim w/o int. loss	3.683 ± 2.270	94.813 ± 0.028	98.023 ± 0.084	89.381 ± 0.156	$(3.304 \pm 0.208) \times 10^{-9}$
S2MDF (Shift-All) - Train	2.120 ± 0.117	94.823 ± 0.032	98.039 ± 0.017	89.297 ± 0.262	$(5.665 \pm 2.370) \times 10^{-9}$
S2MDF (Shift-All) - Train w/o int. loss	13.040 ± 1.498	94.737 ± 0.100	97.939 ± 0.046	88.957 ± 0.156	$(8.697 \pm 0.612) \times 10^{-9}$

Table A.3: Non-intersection loss ablation: shape reconstruction metrics for PartSDF on 100 chairs from the test set. Mean and standard deviation across 3 runs.

PartSDF includes a non-intersection loss term in its training objective, which is designed to penalize intersections between parts. In Tab. A.3, we present quantitative results for the same PartSDF [52] experiment as in Tab. 3, on PartNet [38] chairs, additionally comparing the application of S2MDF with and without the non-intersection loss term. We observe that, while the non-intersection loss term does improve the results of PartSDF, it is not sufficient to prevent intersections between parts, and applying S2MDF provides a significant improvement over both cases. We also observe that, due to the non-watertight GT parts of the chairs dataset, the MCD scores become significantly worse when the non-intersection loss is removed. Our method is able to remove intersections nonetheless, but it can benefit from the non-intersection loss to improve the overall quality of the reconstruction.



(a) Renderings (b) Intersections

Figure A.2: PartSDF on PartNet chairs at increasing meshing resolutions. Intersecting surfaces are shown in red, while touching surfaces are shown in green.

A.2 Proofs

A.2.1 Theorem 1 (Constraint Equivalence for SDFs)

A vector-valued $SDF = [SDF_1, \ldots, SDF_K] : \mathbb{R}^3 \to \mathbb{R}^K$ whose components are exact SDFs as in Eq. (1) satisfies constraint $\min^{(2)}(\mathbf{u}) \geq 0$ at every point $\mathbf{p} \in \mathbb{R}^3$, with $\mathbf{u} = SDF(\mathbf{p})$, if and only if it satisfies $\min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}) \geq 0$ at every point $\mathbf{p} \in \mathbb{R}^3$.

Proof. Through the second constraint and the definition of $\min^{(n)}$, we have both $\min^{(2)}(\mathbf{u}) \geq -\min^{(1)}(\mathbf{u})$ and $\min^{(2)}(\mathbf{u}) \geq \min^{(1)}(\mathbf{u})$ and, therefore, $\min^{(2)}(\mathbf{u}) \geq 0$ at every point $\mathbf{p} \in \mathbb{R}^3$.

For the inverse implication, we consider an arbitrary point $\mathbf{p} \in \mathbb{R}^3$. If $\mathbf{p}$ lies outside all objects, the second condition is trivially satisfied as all SDFs are positive. If it is inside an object, we can use the Lipschitz continuity of SDFs to show that the second constraint must also be satisfied.

For this, we first define some notation for point $\mathbf{p}$ with SDF vector $\mathbf{u}$. Let:

- a and b be the indices of the objects with the smallest and second smallest SDF, respectively.
- $\mathbf{y}_a$ and $\mathbf{y}_b$ be points on the surfaces of these objects that are closest to $\mathbf{p}$.
- $s_a : \mathbb{R}^3 \to \mathbb{R}$ and $s_b : \mathbb{R}^3 \to \mathbb{R}$ be the signed distance functions of these objects.

Based on the definition of SDFs of Equation (1) and using the first constraint, we can write

$$\begin{aligned} \min^{(1)}(\mathbf{u}) &= s_a(\mathbf{p}) = -\|\mathbf{p} - \mathbf{y}_a\| \text{ and} \\ \min^{(2)}(\mathbf{u}) &= s_b(\mathbf{p}) = \|\mathbf{p} - \mathbf{y}_b\| \ . \end{aligned}$$

Using that SDFs are 1-Lipschitz continuous, we have

$$\begin{aligned} & s_a(\mathbf{y}_b) - s_a(\mathbf{p}) \leq |s_a(\mathbf{y}_b) - s_a(\mathbf{p})| \leq \|\mathbf{y}_b - \mathbf{p}\| \\ \Leftrightarrow \ & s_a(\mathbf{y}_b) \leq s_a(\mathbf{p}) + \|\mathbf{y}_b - \mathbf{p}\| \\ & \qquad = s_a(\mathbf{p}) + s_b(\mathbf{p}) \\ & \qquad = \min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}) \end{aligned}$$

Suppose now that the second constraint is not satisfied at $\mathbf{p}$. Then

$$s_a(\mathbf{y}_b) \leq \min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}) < 0 \ ,$$

which means that $\mathbf{y}_b$, a point on the surface of object b, is strictly inside a. Thus, in its vicinity, there must be a point that is both within object a and b, which is impossible as it violates the first constraint. Therefore, there cannot be any point $\mathbf{p}$ such that the second constraint is not satisfied. □

A.2.2 Sum of two mins is equivalent to pairwise sums

Consider an arbitrary vector $\mathbf{d} \in \mathbb{R}^K$ and its elements $d_i,\ 1 \leq i \leq K$. The inequality $\min^{(2)}(\mathbf{d}) + \min^{(1)}(\mathbf{d}) \geq 0$ is satisfied if and only if $d_i + d_j \geq 0,\ 1 \leq i < j \leq K$.

Proof. Because the $\min^{(2)}$, $\min^{(1)}$ and the set of all pairwise sums are invariant under permutation of elements, we can assume without loss of generality that the elements of $\mathbf{d}$ are sorted in ascending order $\mathbf{d} = (d_1, d_2, \ldots, d_K)$, $d_1 \leq d_2 \leq \cdots \leq d_K$.

Forward Proof. We are given that $\min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0$, i.e. $d_1 + d_2 \geq 0$.

- Given $i \geq 1$, it follows that $d_i \geq d_1$.
- As $j > i \geq 1$, $j \geq 2$. It follows that $d_j \geq d_2$.

We can combine the two inequalities to get $d_i + d_j \geq d_1 + d_2$. This completes the forward proof.

Backward Proof. We are given that $d_i + d_j \geq 0,\ 1 \leq i < j \leq K$. We can set $i = 1$ and $j = 2$, meaning $d_1 + d_2 \geq 0$. Because $\min^{(1)}(\mathbf{d}) = d_1$ and $\min^{(2)}(\mathbf{d}) = d_2$, it follows that $\min^{(2)}(\mathbf{d}) + \min^{(1)}(\mathbf{d}) \geq 0$. This completes the backward proof. □

A.2.3 Theorem 2: Linear Interpolation under MDF constraint

Given two points $\mathbf{p_1}, \mathbf{p_2} \in \mathbb{R}^3$ and an arbitrary vector-valued function $f : \mathbb{R}^3 \to \mathbb{R}^K$ evaluated at those points, $\mathbf{d}_1 = f(\mathbf{p_1})$ and $\mathbf{d}_2 = f(\mathbf{p_2})$ that satisfy constraint 3, i.e. $\min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0$, then the constraint is also satisfied for every point along the line segment connecting $\mathbf{p_1}$ and $\mathbf{p_2}$, where intermediate vectors are obtained via linear interpolation as $\mathbf{d}_\alpha = \alpha\mathbf{d}_1 + (1-\alpha)\mathbf{d}_2$ for $\alpha \in [0, 1]$.

Proof. For notational convenience, let $S_2(\mathbf{d}) = \min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d})$, allowing us to state the constraint as $S_2(\mathbf{d}) \geq 0$. Example 3.6, p. 80 in [4] proves that the function $L_r(\mathbf{d})$, representing the sum of the r largest components of a vector, is convex. The sum of the r smallest components can be expressed as $S_r(\mathbf{d}) = -L_r(-\mathbf{d})$. Since L_r is convex, $L_r(-\mathbf{d})$ is convex, and its negation $-L_r(-\mathbf{d})$ is concave. Because S_2 is a special case of S_r, it is also concave. Using the definition of concavity for $\alpha \in [0, 1]$:

$$S_2(\mathbf{d}_\alpha) = S_2(\alpha\mathbf{d}_1 + (1-\alpha)\mathbf{d}_2) \geq \alpha S_2(\mathbf{d}_1) + (1-\alpha)S_2(\mathbf{d}_2) \tag{9}$$

Since $S_2(\mathbf{d}_1) \geq 0$ and $S_2(\mathbf{d}_2) \geq 0$, and both α and $(1-\alpha)$ are non-negative, it follows that $\alpha S_2(\mathbf{d}_1) + (1-\alpha)S_2(\mathbf{d}_2) \geq 0$. Thus, $S_2(\mathbf{d}_\alpha) \geq 0$, guaranteeing that $\mathbf{d}_\alpha$ always satisfies the MDF constraint Eq. (3). □

A.2.4 Constraint 2 does not allow linear MDF interpolation

Given two points $\mathbf{p_1}, \mathbf{p_2} \in \mathbb{R}^3$ and an arbitrary vector-valued function $f : \mathbb{R}^3 \to \mathbb{R}^K$ evaluated at those points, $\mathbf{d}_1 = f(\mathbf{p_1})$ and $\mathbf{d}_2 = f(\mathbf{p_2})$ that satisfy constraint 2, i.e. $\min^{(2)}(\mathbf{d}) \geq 0$, then the constraint is not guaranteed to be satisfied for every point along the line segment connecting $\mathbf{p_1}$ and $\mathbf{p_2}$, where intermediate vectors are obtained via linear interpolation as $\mathbf{d}_\alpha = \alpha\mathbf{d}_1 + (1-\alpha)\mathbf{d}_2$ for $\alpha \in [0, 1]$.

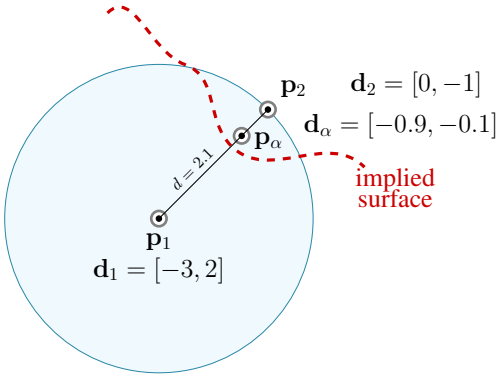


Figure A.3: When dealing with general functions (e.g. imperfect SDFs), the constraint 2 does not guarantee that a linear interpolation of two points $\mathbf{p}_1$ and $\mathbf{p}_2$ that satisfy the constraint will also satisfy the constraint. In fact, there exists a point $\mathbf{p}_\alpha$ on the line segment between $\mathbf{p}_1$ and $\mathbf{p}_2$ that violates the constraint.

Proof. We prove it by counterexample. Consider the two points $\mathbf{p}_1$ and $\mathbf{p}_2$ in Fig. A.3 with $\mathbf{d}_1 = [-3, 2]$ and $\mathbf{d}_2 = [0, -1]$. Both of these vectors satisfy the constraint $\min^{(2)}(\mathbf{d}) \geq 0$. However, the linearly interpolated point $\mathbf{p}_\alpha$ between them has $\mathbf{d}_\alpha = [-0.9, -0.1]$, which violates the constraint $\min^{(2)}(\mathbf{d}_\alpha) \geq 0$. This counterexample demonstrates that the constraint 2 does not guarantee that a linear interpolation of two points satisfying the constraint will also satisfy the constraint when dealing with general functions (e.g. imperfect SDFs). □

A.2.5 Line Crossing

Given two points $\mathbf{p_1}, \mathbf{p_2} \in \mathbb{R}^3$ and an arbitrary vector-valued function $f : \mathbb{R}^3 \to \mathbb{R}^K$ evaluated at those points, $\mathbf{d}_1 = f(\mathbf{p_1})$ and $\mathbf{d}_2 = f(\mathbf{p_2})$, if they both satisfy the MDF constraint (Eq. (3)), i.e. $\min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0$, and an edge between them is linearly interpolated by a meshing algorithm for surface reconstruction, we claim that there are at most two object surfaces between these points and that these surfaces do not cross.

Proof. We already know from Theorem 1 that $\min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0$ implies $\min^{(2)}(\mathbf{d}) \geq 0$, which implies a vector $\mathbf{d}$ satisfying this constraint can have at most one negative value.

Let us assume for now that neither of the two vectors contain zero values. Let us perform a case analysis on the possibilities of the signs of the values of the two vectors:

- Neither $\mathbf{d}_1$ nor $\mathbf{d}_2$ contain negative values. In this case, there are no object surfaces between them.
- One of them has a negative value. In this case, there is only one object surface, and we are outside every other object.
- Both of them have a negative value, and these negative values are in the same index i. In this case, both points are inside the same object and there will be no surface between them.
- Both of them have a negative value, and these are in different indices, i and j. Then, there will be two zero-crossings and two object surfaces.

The first three cases have no risk of having two surfaces intersect, so our interest is in the final case. Let $\mathbf{d}_1 = \mathbf{p}$ and $\mathbf{d}_2 = \mathbf{q}$ for notational convenience, and i, j be the minimum indices in $\mathbf{p}$ and $\mathbf{q}$, respectively. Then, $p_i < 0$ and $q_j < 0$. We know from Appendix A.2.2 that satisfaction of the MDF

constraint implies all pairwise sums in a vector are positive. Thus, we have that:

$$p_i < 0,\ p_i + p_j \geq 0 \implies p_j > 0 \tag{10}$$

$$q_j < 0,\ q_i + q_j \geq 0 \implies q_i > 0 \tag{11}$$

Surfaces along the line segment will appear at zero-crossings. If we parameterize points along this segment with $\alpha \in [0, 1]$, assuming $\alpha = 0$ at the point where $\mathbf{p}$ is evaluated, and $\alpha = 1$ for $\mathbf{q}$, we can find the zero-crossing for both object i and j:

$$\alpha_i = \frac{-p_i}{q_i - p_i} \tag{12}$$

$$\alpha_j = \frac{-p_j}{q_j - p_j} \tag{13}$$

We will now derive the relationship between α_i and α_j based on Eq. (10) and Eq. (11). Given $p_j > 0$, we divide $p_i + p_j \geq 0$ to obtain$\frac{p_i}{p_j} + 1 \geq 0$. Rearranging this, and with the knowledge that $p_i < 0$, we obtain the bounds:

$$1 \geq -\frac{p_i}{p_j} > 0 \tag{14}$$

With a similar approach, dividing $q_i + q_j \geq 0$ by $q_i > 0$, we can also obtain:

$$1 \geq -\frac{q_j}{q_i} > 0 \tag{15}$$

Multiplying both inequalities:

$$1 \geq \frac{p_i q_j}{p_j q_i} > 0 \tag{16}$$

Since $p_j > 0$ and $q_i > 0$, $p_j q_i > 0$ and we can multiply both sides with it, dropping the lower bound:

$$p_j q_i \geq p_i q_j \tag{17}$$

Now, subtract $p_i p_j$ from both sides and factor:

$$p_j(q_i - p_i) \geq p_i(q_j - p_j) \tag{18}$$

We know $p_i < 0$ and $q_i > 0$, so $q_i - p_i > 0$, we divide both sides:

$$p_j \geq (q_j - p_j)\frac{p_i}{q_i - p_i} \tag{19}$$

Similarly, $p_j > 0$ and $q_j < 0$, so $q_j - p_j < 0$, divide both sides and flip the inequality:

$$\frac{p_j}{q_j - p_j} \leq \frac{p_i}{q_i - p_i} \tag{20}$$

Multiply both sides by -1:

$$-\frac{p_j}{q_j - p_j} \geq -\frac{p_i}{q_i - p_i} \tag{21}$$

And we finally obtain α_i and α_j:

$$\alpha_j \geq \alpha_i \implies \alpha_i \leq \alpha_j \tag{22}$$

This proves that the surface of the object j will always be closer to the point where $\mathbf{q}$ is evaluated than object i. The surfaces can at most touch each other, or will be disjoint. This completes our proof in the case where neither of the vectors contain zero values. If they do contain zero values, the following cases can happen:

- One of the vectors $\mathbf{d}_1, \mathbf{d}_2$ has one or more zero values, let us assume $\mathbf{d}_1$, and the other vector does not. A surface point will be placed exactly at the point where $\mathbf{d}_1$ is evaluated. Due to Eq. (3), the zero value in $\mathbf{d}_1$ implies the absence of negative values in $\mathbf{d}_1$, which in turn implies that its point is on the surface of one or more objects but outside every other object.
- Both vectors have one or more zero values in the same indices. This implies that one or more surfaces lie exactly on the line segment, which does not cause a crossing.

- Both vectors have one or more zero values, and these zero values are potentially in different indices. This implies the presence of multiple surface points at the points where the vectors are evaluated, however neither of the vectors can have negative values, due to Eq. (3), which implies that both points are on the surface of one or more objects but outside every other object.

□

A.2.6 QP analytical solution for the two-object case

Given the optimization problem:

$$\text{minimize} \|\mathbf{d} - \mathbf{u}\|^2 \quad \text{subject to} \quad \min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0 \; . \tag{23}$$

in two dimensions, i.e. $\mathbf{d} \in \mathbb{R}^2$, its optimal solution is equivalent to Shift-All, i.e. not changing $\mathbf{u}$ in the satisfied case, and shifting both variables by $-(\min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}))/2$ in the unsatisfied case.

Proof. Let us state the common way to formulate an optimization problem with inequality constraints:

$$\text{minimize } f(\mathbf{d}) \quad \text{subject to} \quad g(\mathbf{d}) \leq 0 \; . \tag{24}$$

Let $\mathbf{u} = (u_1, u_2)$ and $\mathbf{d} = (d_1, d_2)$. As there are only two elements, $\min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) = d_1 + d_2 \geq 0 \implies -d_1 - d_2 \leq 0$.

Then, for our case, $f(\mathbf{d}) = \|\mathbf{d} - \mathbf{u}\|^2 = (d_1 - u_1)^2 + (d_2 - u_2)^2$ and $g(\mathbf{d}) = -d_1 - d_2$. Let's define the Lagrangian function:

$$\mathcal{L}(\mathbf{d}, \lambda) = f(\mathbf{d}) + \lambda g(\mathbf{d}) = (d_1 - u_1)^2 + (d_2 - u_2)^2 - \lambda(d_1 + d_2) \tag{25}$$

We now write down the KKT conditions:

1. **Stationarity:** $\nabla f(\mathbf{d}^*) + \lambda \nabla g(\mathbf{d}^*) = 0$.
2. **Primal feasibility:** $g(\mathbf{d}^*) \leq 0$.
3. **Dual feasibility:** $\lambda \geq 0$.
4. **Complementary slackness:** $\lambda g(\mathbf{d}^*) = 0$.

Before continuing our solution, we derive the gradients:

$$\nabla f(\mathbf{d}) = 2 \begin{bmatrix} d_1 - u_1 \\ d_2 - u_2 \end{bmatrix}, \quad \nabla g(\mathbf{d}) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \tag{26}$$

Inactive case. Assume $g(\mathbf{d}^*) < 0$, meaning the constraint is already satisfied. Then, by stationarity, $\nabla f(\mathbf{d}^*) = 0$:

$$\nabla f(\mathbf{d}^*) = 2 \begin{bmatrix} d_1^* - u_1 \\ d_2^* - u_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \implies \begin{matrix} d_1^* = u_1 \\ d_2^* = u_2 \end{matrix}, \mathbf{d}^* = \mathbf{u} \tag{27}$$

By complementary slackness, $\lambda = 0$. Let us check the feasibility:

- **Primal:** $g(\mathbf{d}^*) = g(\mathbf{u}) = -u_1 - u_2 < 0 \implies u_1 + u_2 > 0$.
- **Dual:** $\lambda = 0 \geq 0$, satisfied.

Feasibilities are satisfied and the inactive solution is valid when $u_1 + u_2 > 0$. The solution intuitively makes sense, as our projection does not need to modify the initial value $\mathbf{u}$ in case the constraint is already satisfied.

Active case. Assume $g(\mathbf{d}^*) = 0$, meaning the solution is at the constraint boundary. By complementary slackness:

$$\lambda g(\mathbf{d}^*) = -\lambda(d_1^* + d_2^*) = 0 \implies d_2^* = -d_1^* \tag{28}$$

By stationarity, $\nabla f(\mathbf{d}^*) + \lambda \nabla g(\mathbf{d}^*) = 0$, and substituting the above $d_2^* = -d_1^*$:

$$2\begin{bmatrix} d_1^* - u_1 \\ -d_1^* - u_2 \end{bmatrix} - \lambda \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \implies \begin{matrix} d_1^* - u_1 = \lambda/2 \\ -d_1^* - u_2 = \lambda/2 \end{matrix} \tag{29}$$

Summing the two equations, we get $\lambda = -(u_1 + u_2)$. We can plug this result back into the equations to derive $\mathbf{d}^*$:

$$d_1^* = u_1 + \frac{\lambda}{2} = u_1 - \frac{u_1 + u_2}{2} \tag{30}$$

$$d_2^* = u_2 + \frac{\lambda}{2} = u_2 - \frac{u_1 + u_2}{2} \tag{31}$$

Let's consider the feasibilities:

- **Primal:** $g(\mathbf{d}^*) = -d_1^* - d_2^* = -u_1 + \frac{u_1+u_2}{2} - u_2 + \frac{u_1+u_2}{2} = 0 \leq 0$, satisfied.
- **Dual:** $\lambda = -u_1 - u_2 \geq 0 \implies u_1 + u_2 \leq 0$.

Feasibilities are satisfied and the active solution is valid when $u_1 + u_2 \leq 0$, meaning both values need to be shifted by $-\dfrac{u_1 + u_2}{2}$ when the constraint is not satisfied on $\mathbf{u}$.

Because our objective function $f(\mathbf{d}) = \|\mathbf{d} - \mathbf{u}\|^2$ is strictly convex and the inequality constraint $g(\mathbf{d}) = -d_1 - d_2 \leq 0$ is affine, the optimization problem is strictly convex. Under these conditions, the KKT conditions are both necessary and sufficient for the $\mathbf{d}^*$ we derived to be the **global optimum** [4, Section 5.5.3, p. 244].

This analytical solution leads to the exact same formulations as our Shift-All. □

A.2.7 Solution to the modified QP problem

Let $\mathbf{u} \in \mathbb{R}^K$ be the K-dimensional SDF vector at a point in space, the following optimization problem:

$$\begin{aligned} \underset{\mathbf{d} \in \mathbb{R}^K}{\text{minimize}} \quad & \|\mathbf{d} - \mathbf{u}\|^2 && \text{(32a)} \\ \text{subject to} \quad & \min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d}) \geq 0, , && \text{(32b)} \\ & \mathbf{d}_i - \mathbf{d}_j = \mathbf{u}_i - \mathbf{u}_j, \forall i, j \in [K], i < j && \text{(32c)} \end{aligned}$$

has Shift-All as its optimal solution, i.e. not changing $\mathbf{u}$ in the satisfied case, and shifting all variables by $-(\min^{(1)}(\mathbf{u}) + \min^{(2)}(\mathbf{u}))/2$ in the unsatisfied case.

Proof. The unconstrained minimizer of the objective function is $\mathbf{d} = \mathbf{u}$. If $\mathbf{u}$ satisfies the MDF constraint, then $\mathbf{d} = \mathbf{u}$ is also the optimal solution to the constrained problem.

If $\mathbf{u}$ does not satisfy the MDF constraint, we notice that the second constraint forces solutions to be of the form $\mathbf{d} = \mathbf{u} + c\mathbf{1}$, where $\mathbf{1}$ is the vector of all ones. For simplicity, we define $\mathbf{d}_m = \min^{(1)}(\mathbf{d})$ and $\mathbf{d}_n = \min^{(2)}(\mathbf{d})$. Then, we have that:

$$\mathbf{d}_m = \mathbf{u}_m + c \tag{33}$$

$$\mathbf{d}_n = \mathbf{u}_n + c \tag{34}$$

From the first constraint of the optimization problem, it follows that:

$$\mathbf{d}_m + \mathbf{d}_n = \mathbf{u}_m + \mathbf{u}_n + 2c \geq 0 \implies c \geq -\frac{\mathbf{u}_m + \mathbf{u}_n}{2} \tag{35}$$

The objective function is convex so, if its global minimum is not feasible, the optimal solution $\mathbf{d}^*$ will be at the boundary of the feasible region. In fact, if it wasn't at the boundary, then it would be in the interior of the feasible region. In such case, $\nabla\|\mathbf{d}-\mathbf{u}\|^2 = 2(\mathbf{d}-\mathbf{u}) = 0$ at $\mathbf{d}^*$, otherwise it would not be optimal, but this implies that $\mathbf{d}^* = \mathbf{u}$, which is not feasible. Therefore, $\mathbf{d}^*$ must be at the boundary, meaning $c = -\frac{\mathbf{u}_m+\mathbf{u}_n}{2}$.

In short, if $\mathbf{u}$ satisfies the MDF constraint, then $\mathbf{d}^* = \mathbf{u}$ is the optimal solution. If $\mathbf{u}$ does not satisfy the MDF constraint, then $\mathbf{d}^* = \mathbf{u} - \frac{\min^{(1)}(\mathbf{u})+\min^{(2)}(\mathbf{u})}{2}\mathbf{1}$ is the optimal solution. This is exactly the Shift-All solution as defined in the main text.

□

A.2.8 Proof for trilinearity

Claim: Let $\mathbf{d}_{i,j,k} \in \mathbb{R}^K$, $i,j,k \in \{0,1\}$, be the vector-valued SDF evaluations at the vertices of a voxel cell, each satisfying the MDF constraint $\min^{(1)}(\mathbf{d}_{i,j,k}) + \min^{(2)}(\mathbf{d}_{i,j,k}) \geq 0$. Then the trilinearly interpolated field satisfies the constraint at every point of the cell.

Proof: Let $S_2(\mathbf{d}) = \min^{(1)}(\mathbf{d}) + \min^{(2)}(\mathbf{d})$. We are given that $S_2(\mathbf{d}_{i,j,k}) \geq 0$ for all $i,j,k \in \{0,1\}$. As established in Theorem 2 (Appendix A.2.3), the linear interpolation $\mathbf{d}_\alpha = \alpha\mathbf{d}_a + (1-\alpha)\mathbf{d}_b$ between any two valid vectors guarantees $S_2(\mathbf{d}_\alpha) \geq 0$ for $\alpha \in [0,1]$. Trilinear interpolation for a point at local coordinates $(x,y,z) \in [0,1]^3$ decomposes into three sequential linear interpolations. By applying Theorem 2 on S_2 at each step, the constraint is preserved:

Step 1: x-axis. Interpolating the 8 vertices along the x-axis yields four vector-valued functions along the four edges in the x direction:

$$\mathbf{d}_{0,0}(x) = (1-x)\mathbf{d}_{0,0,0} + x\mathbf{d}_{1,0,0}$$
$$\mathbf{d}_{1,0}(x) = (1-x)\mathbf{d}_{0,1,0} + x\mathbf{d}_{1,1,0}$$
$$\mathbf{d}_{0,1}(x) = (1-x)\mathbf{d}_{0,0,1} + x\mathbf{d}_{1,0,1}$$
$$\mathbf{d}_{1,1}(x) = (1-x)\mathbf{d}_{0,1,1} + x\mathbf{d}_{1,1,1}$$

Since the endpoints satisfy the constraint, Theorem 2 dictates that $S_2(\mathbf{d}_{j,k}(x)) \geq 0$ for $j,k \in \{0,1\}$.

Step 2: y-axis. Interpolating these 4 along the y-axis yields two vector-valued functions on the $z=0$ and $z=1$ faces:

$$\mathbf{d}_0(x,y) = (1-y)\mathbf{d}_{0,0}(x) + y\mathbf{d}_{1,0}(x)$$
$$\mathbf{d}_1(x,y) = (1-y)\mathbf{d}_{0,1}(x) + y\mathbf{d}_{1,1}(x)$$

Applying Theorem 2 again, $S_2(\mathbf{d}_0(x,y)) \geq 0$ and $S_2(\mathbf{d}_1(x,y)) \geq 0$.

Step 3: z-axis. Interpolating the 2 along the z-axis gives the composite field inside the full cube, defined at (x,y,z):

$$\mathbf{d}(x,y,z) = (1-z)\mathbf{d}_0(x,y) + z\mathbf{d}_1(x,y)$$

By a final application of Theorem 2, $S_2(\mathbf{d}(x,y,z)) \geq 0$, which proves that trilinearly interpolated values also satisfy the constraint.

A.3 Additional Analyses

A.3.1 Compatibility with learning-based isosurface extraction methods

The deciding property for the use of our method is how the extractor produces geometry between the discrete samples: Our guarantee applies to any algorithm that places vertices by linear interpolation of the sampled values, since Theorem 2 extends constraint (3) exactly along linearly interpolated segments.

This includes learning-based pipelines. DMTet [47] learns both the grid deformation and the per-vertex SDF values, but extracts surfaces with Marching Tetrahedra via linear interpolation, so our guarantee applies in full. Our MedTet experiments (Tab. 1) run on exactly this setting. The same holds for deformable and hierarchical grids in general, provided extraction remains linearly interpolated.

Methods that instead learn the geometry placement itself fall outside our guarantee: Neural Marching Cubes [10] and Neural Dual Contouring [8] predict vertex positions with a network, and FlexiCubes [48] repositions crossings with learned weights, so the extracted triangles are no longer tied to the linearly interpolated field. Even there, our constraint would guarantee an intersection-free interpolated field if used, but the transfer of this property to the extracted mesh would be lost.

Finally, linear interpolation is a choice we made to be compatible with existing algorithms that mesh each SDF channel independently. An alternative direction would be to mesh all channels jointly, as in multi-object Dual Contouring [44]. This would complement our approach without substituting for it. Such a mesh extraction algorithm would enforce non-intersection, but only in the extracted mesh. In contrast, our constraint yields an intersection-free field, which is what enables its use during training in SDF-based rendering and optimization where no mesh is involved, and with any linearly interpolated extractor.

A.3.2 Marching Tetrahedra and Marching Cubes

For **Marching Tetrahedra**, the field inside each tetrahedron is linearly interpolated from the values at its four vertices. As in Appendix A.2.8, this interpolation decomposes into three sequential linear interpolations, adding one vertex at a time, so that by Theorem 2 the interpolated field satisfies the constraint everywhere inside the tetrahedron. The extracted triangles always lie directly on the interpolated surfaces, and the absence of intersections is guaranteed.

For **Marching Cubes**, the interpolated field inside each cell is trilinear and satisfies the constraint everywhere, as proven in Appendix A.2.8. The vertices lie exactly on the interpolated surface, but the full triangles do not always. This introduces two considerations, both of which are properties of Marching Cubes itself and not related to our method:

- There are ambiguous configurations in Marching Cubes. When these are picked inconsistently between objects in a cell, as can happen in the original approach [34], unwarranted intersections can appear even in valid fields. However, newer implementations, such as [30], use a calculation called "Asymptotic Decider" to choose a particular configuration in ambiguous cases. This yields consistent reconstructions without intersections in all practical cases.
- Intersections are still possible on very low-resolution grids, even on fields that are guaranteed not to intersect, due to the placement of triangles. This can happen when one object is locally concave within a single cell, and its triangles "bow" outward towards the other object. However, in practice, this requires significant bending inside a single cell and does not happen when the grid resolution is large enough.

In short, if theoretical guarantees are needed, we recommend using Marching Tetrahedra. Nevertheless, for all practical purposes, our method is also compatible with Marching Cubes as shown in our experiments.

A.3.3 Surface analysis for Shift-All

We perform an additional experiment on our trained heart model (MM-WHS) to analyze the effect of Shift-All on the surfaces, and report the results in Tab. A.4. Shift-All does not thicken the shapes and removes their overlaps, while the resulting displacement and thinning are small, local to the contact regions, and do not reduce accuracy near contacts.

At every grid point where two shapes overlap (the two smallest signed distances sum to a negative value), Shift-All adds the same positive constant to all channels so that the sum becomes zero; points not in violation are left untouched.

Thickening. Shift-All only adds non-negative constants to the signed distances, so every shape can only shrink, and previously separated shapes cannot be pushed into new overlaps (row 1).

Thinning. Shapes lose volume only where they were overlapping: summed over shapes, the removed volume matches the overlap that was present, so each shape roughly contributes half the reduction (rows 2–3).

Property	Metric	Value
Thickening	Min. SDF change over grid	0.000
Thinning	IV	$2.0 \times 10^{-4} \to \sim 0$
	Removed volume $\div$ initial IV	≈ 1.0
Displacement	Unmoved surface	89%
	Surface moved $< 10^{-2}$	$>99.9\%$
	Max. move	1.9×10^{-2}
Contact	Chamfer dist. in contact regions	$2.5 \times 10^{-3} \to 2.5 \times 10^{-3}$
	Surface inside another part	$6.40\% \to 0.00\%$
	Surface within 10^{-2} of another part	$30.1\% \to 30.2\%$
	Surface within 10^{-3} of another part	$6.6\% \to 9.1\%$
	Surface within 10^{-4} of another part	$0.6\% \to 3.2\%$

Table A.4: Shift-All as post-processing on our trained heart model (MM-WHS). Arrows indicate values before $\to$ after Shift-All. Distances are in the $[-1, 1]$ object scale, so the largest move is about 1% of it. Distances to other parts are unsigned, so penetrating surface is already counted in a band when its depth is below the threshold.

Displacement. Points not in violation are untouched, so most of the surface does not move. The part that moves lies inside the overlap region and moves by at most 1.9×10^{-2} (rows 4–6).

Contact. Shift-All moves both overlapping surfaces to where their fields cross, so that the shapes are brought into contact and not pushed apart. Accuracy in contact regions, measured against the ground truth that has no overlaps, is unchanged (row 7).

This experiment uses a vanilla model with Shift-All as post-processing; for models trained with Shift-All, the pre-shift field lacks supervision and is not meaningful for this comparison.