

Strictly Conservative Neural Distance Fields

I. Ludwig^{ID} M. Campen^{ID}

Paderborn University, Germany

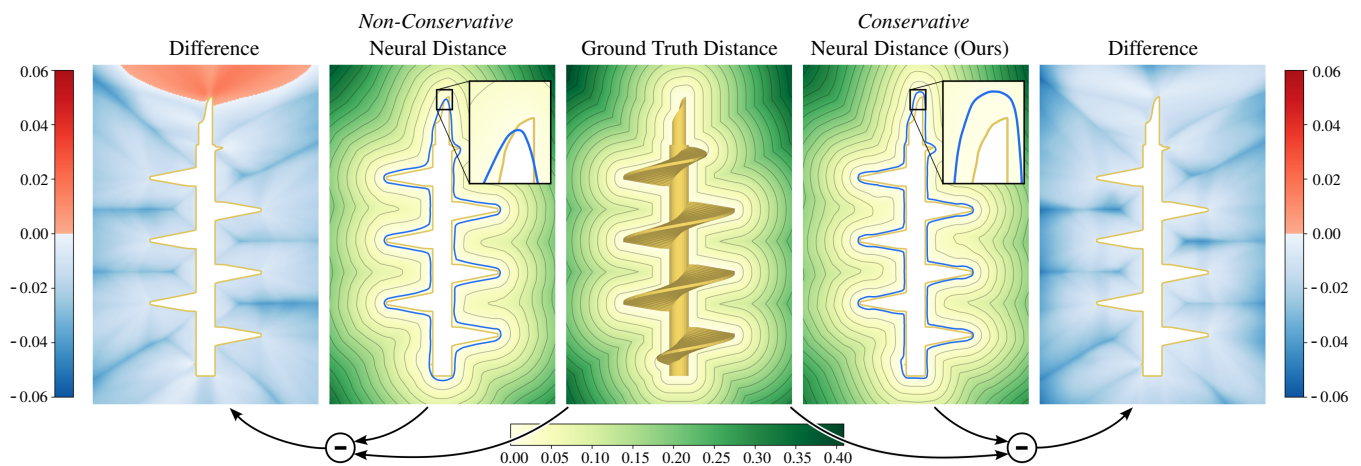


Figure 1: Our method allows converting 3D shapes into Neural Distance Fields that, despite being approximate, are guaranteed to be conservative: The true distance to the shape is never overestimated anywhere (no red region), thus can reliably be used for, e.g., collision avoidance, and the zero-level set is guaranteed to enclose the shape (see zoom-in), thus can reliably serve as, e.g., a bounding volume.

Abstract

We propose a first method to generate neural unsigned or signed distance fields (SDFs) that are guaranteed to be conservative with respect to a given 3D shape. This means the true distance is never overestimated and the zero-level set is a bounding volume for the shape. The method makes use of neural network architectures that ensure Lipschitz continuity by design in combination with a novel tailored training data selection scheme and constrained training strategy. We demonstrate that this yields both theoretical and empirical benefits over previous approaches to conservativeness (for non-distance neural implicits), allowing for tighter approximation and additionally providing the valuable distance information.

Keywords: neural implicit, neural shape representation, conservative shape approximation, bounding volume

CCS Concepts

• Computing methodologies → Shape analysis; Mesh models; Neural networks;

1. Introduction

Since their introduction a few years ago, neural implicits – i.e. a shape representation defining surfaces as level sets of neural network based functions – have rapidly gained interest. A particularly prominent variant are neural (unsigned or signed) distance fields (*neural DF/SDF*), which, in addition to describing a surface, also provide an (approximate) distance to this surface for any point in space. This distance information is valuable for a variety of applications [MSLJ23].

Given a shape (in some other representation), a neural network can be trained to (approximately) represent this shape – its surface and its distance field. We describe the first training method that allows ensuring that the resulting neural distance field is *conservative*. This means the true distance to the given shape is nowhere overestimated (Figure 1). This property is of relevance for diverse scenarios including collision avoidance and maintenance of safety margins, and generally for ray marching-style techniques, ensuring that no part of a shape is missed.

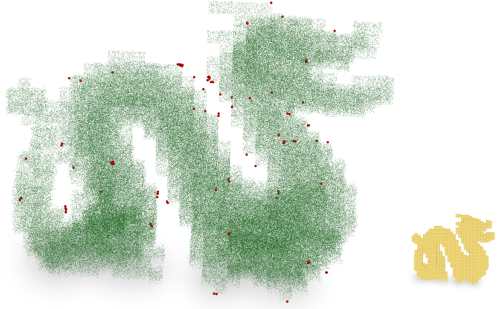


Figure 2: A neural implicit f trained to enclose a given shape S (voxel-based, yellow) using the method of [LFYR24]. Shown are 1 million random test points inside S , colored green (red) if they are inside (outside) the zero-level set of f . The 45 red points (effectively false negatives) demonstrate the non-conservativeness of f .

Conservativeness has been addressed before for (non-distance) neural implicits. In this case a conservative neural implicit defines a surface that *contains* or *encloses* the shape it was generated for. This relationship plays an important role for use cases such as bounding volumes, deformation cages, nested constructions, occlusion proxies and others. Liu et al. [LFYR24] incentivize conservativeness of neural implicits for such scenarios using an asymmetric loss. However, guarantees are not provided and conservativeness is not always achieved perfectly, as demonstrated in Figure 2. Ludwig and Campen [LC24] make use of range analysis techniques to provide these guarantees.

Such a range analysis system, based on some form of interval arithmetic, does not readily extend to distance fields. Furthermore, already for the case of non-distance neural implicits, the error margin of range analysis increases with every additional hidden layer of a neural network, fundamentally limiting the achievable approximation quality, as the increase in expressive power of deeper networks is quickly outweighed by the increased error margins, leading to unnecessarily non-tight results for shallow and for deep neural network configurations alike, see Figure 3.

We instead propose to build guarantees on top of Lipschitz continuity properties of neural architectures. This not only facilitates the generalization from implicits to distance fields, but interestingly

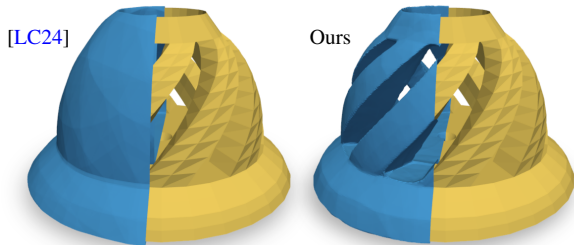


Figure 3: Conservative neural implicit surfaces (blue) for an input shape (yellow). The tightest possible result (over all neural network depths) of a former method based on range analysis is shown. Our method (right) facilitates visibly tighter approximation.



Figure 4: A neural DF f trained for a given shape S (yellow) using the method of [CB24]. Left: true positives (green) and false negatives (red) among 5M random test points; 796 points misclassified as outside. Right: points outside S where the distance is overestimated by f (red, 17% of 5M points). Notice how these volumetric overestimation regions originate from regions of false negatives.

provides benefits regarding the achievable approximation quality – even for the case of non-distance implicits, where only the surface is of interest. In this sense our approach comes with a dual benefit.

Such Lipschitz properties have recently been exploited for the generation of neural SDFs [CB24]. While this naturally ensures (in that work as a by-product) a certain form of conservativeness (of distances with respect to their own zero-level set [MSLJ23] – one could say *self-conservativeness*), it does not yield a guarantee *with respect to the given shape* the network is trained for, neither in terms of containment nor in terms of distance non-overestimation, see Figure 4 and Figure 1 left.

1.1. Conservativeness

Let us define formally what notion of conservativeness of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ we are concerned with, relative to a given shape $S \subset \mathbb{R}^n$, formally a compact point set. We are mainly interested in $\mathbb{R}^3$ herein, but most concepts apply to $\mathbb{R}^n$ in general. $d(x, S)$ denotes the Euclidean distance between point x and point set S .

Conservative Implicit: f is a conservative implicit representation of shape S iff $x \in S \Rightarrow f(x) \leq 0$. Note that this implies $S \subseteq \{x \mid f(x) \leq 0\}$, i.e. the zero-level set of f encloses S .

Conservative DF: f is a conservative distance field (DF) for shape S iff it nowhere overestimates the true distance, $f(x) \leq d(x, S)$. Note that a conservative DF is also a conservative implicit but not vice versa in general.

Conservative SDF: f is a conservative signed distance field (SDF) for shape S iff its magnitude nowhere overestimates the true boundary distance, $|f(x)| \leq d(x, \partial S)$, and its sign nowhere contradicts the true shape, $x \in S \Rightarrow f(x) \leq 0$, $x \notin S \Rightarrow f(x) \geq 0$.

While a shape has a well-defined unique true distance field, $f(x) = d(x, S)$, and signed distance field, $f(x) = \pm d(x, \partial S)$, (neural) approximations thereof are often also referred to as (signed) distance fields for simplicity, a convention we adopt herein. The approximation quality, i.e. the tightness within the conservative regime, can be assessed using these measures:

Volume Error: The relative volume excess of a conservative implicit, called VOL^+ , with 0 indicating perfect reproduction, is computed (analogously to the *overestimation rate* OER [LC24]) as

$$\text{VOL}^+(f, S) = \frac{\text{vol}(\{x \mid f(x) \leq 0\} \setminus S)}{\text{vol}(S)}.$$

Distance Error: The quality of the approximation in terms of distance can be quantified as (sampled) mean (*absolute or relative*) *distance error*, computed over a point sample $\{p_i\}$ as

$$\begin{aligned} \text{DE}_{\text{abs}}(f, S) &= \frac{1}{n} \sum_{i=1}^n |f(p_i) - d(p_i, S)|, \\ \text{DE}_{\text{rel}}(f, S) &= \frac{1}{n} \sum_{i=1}^n \frac{|f(p_i) - d(p_i, S)|}{|d(p_i, S)|}. \end{aligned}$$

1.2. Approach Overview

The problem we aim to address can be stated as follows: Given a shape S , generate a neural implicit or DF or SDF that is provably conservative, yet of high accuracy, i.e. of low volume error VOL^+ and/or distance error DE_{abs} or DE_{rel} .

A key challenge lies in conservativeness being a property characterized by *infinitely* many constraints: a condition needs to be met at each point x within S or $\mathbb{R}^n$. We overcome this by constructively establishing an auxiliary property at a *finite* set of points. By a careful choice of these points in combination with a suitable choice of neural architecture that offers Lipschitz bounds, this extends to the conservativeness property necessarily being satisfied at all points.

Our approach has these main ingredients and steps:

- Construct a *ball cover* for a given shape S ; this provides the required finite point set C .
- Represent f using a *Lipschitz neural network*, an architecture with inherent Lipschitz constant.
- *Initialize* f such that the auxiliary property is satisfied at C by construction.
- Train f for accuracy, i.e. towards reproducing the surface or distance function of S using a tailored *loss*.
- Constrain and control this training process to ensure that conservativeness is *maintained*.

2. Related Work

Neural Implicit Implicit shape representations, indicating point membership based on sign or value, have a long history in computer graphics. In their simplest form, they define a binary occupancy field, amounting to a classification problem. Proposed use cases include 3D reconstruction [MON*19], shape interpolation [CZ19; LWJ*22], shape inference [MPJ*19], and compression [DNJ20]; an overview in the neural context is given by Xie et al. [XTS*22].

Neural SDFs, first introduced by Park et al. [PFS*19], treat the problem as supervised regression, predicting not only membership but also the distance from the surface. Use cases include shape approximation [CP*20] and general geometry processing tasks such as deformation, filtering, and others [YBHK21; MSLJ23]. Many approaches use a classical MLP architecture and focus on improving the loss, e.g. using eikonal regularization [GYH*20] or a total variation loss [CD23]; however, no kinds of guarantees can be given, since the networks are only numerically *optimized* towards the true SDF in a restricted function space.

Béthune et al. [BNC*23] learn SDFs in a semi-supervised fashion using Lipschitz neural networks, training the SDF solely based on point membership labels by leveraging the Lipschitz property. Coiffier and Béthune [CB24] further explore the use of such neural networks for SDFs, exploiting their close relation to the eikonal equation. Their inherent non-overestimation of the distance relative to their own zero-level set is valuable for applications such as sphere tracing (of that zero-level set), e.g., for rendering purposes.

Lipschitz Neural Networks Lipschitz neural networks are architectures whose function space inherently is a subset of L -Lipschitz functions [SMG*21], see Definition 3.1. Initial work on such networks focused on their ability to reduce overfitting [YM17] and their robustness against adversarial attacks [ACB17]. Various approaches to enforcing the Lipschitz property have been proposed, differing in expressiveness and computational cost, including controlling the singular values [YM17] or ∞ -normalization of weight matrices [LWJ*22], regularizing weight matrices to be orthogonal [CBG*17; TK21], and using sorting as a nonlinearity [ALG19]. More recent constructions reduce computational cost further, such as an almost-orthogonal layer [PL22] with singular value updates directly during training, and the Semi-definite Programming Lipschitz Layer (SLL) [AHD*23].

In theory, Lipschitz networks can be as expressive as unconstrained MLPs for classification [BBS*22]. In practice, the hyperparameters of the loss are crucial for controlling the accuracy-robustness trade-off, partly necessitating the introduction of new hyperparameters such as a temperature for the binary cross-entropy loss [BBS*22; SMG*21; BNC*23; CB24].

Conservative Approximation Conservative approximation of geometry has been considered for representations other than neural implicit, for instance conservative voxelizations [ZCEP07] and conservative mesh generation [vdHS15; SVJ15; PK08]. For neural implicit, Liu et al. [LFYR24] proposed a training procedure that incentivizes conservativeness to yield bounding volumes for collision queries. Ludwig et al. [LC24] described an affine arithmetic-based process that converts 3D shapes into neural implicit form, providing formal guarantees of conservative approximation for neural implicit – with certain limitations, as analyzed in Section 5.

Marschner et al. [MSLJ23] discuss a notion of self-conservativeness on an abstract level; Coiffier and Béthune [CB24] constructively yield such self-conservativeness. An example of techniques assuming distance conservativeness is sphere carving [STL*25]. To the best of our knowledge no prior method provides guarantees of neural distance field conservativeness with respect to a given shape.

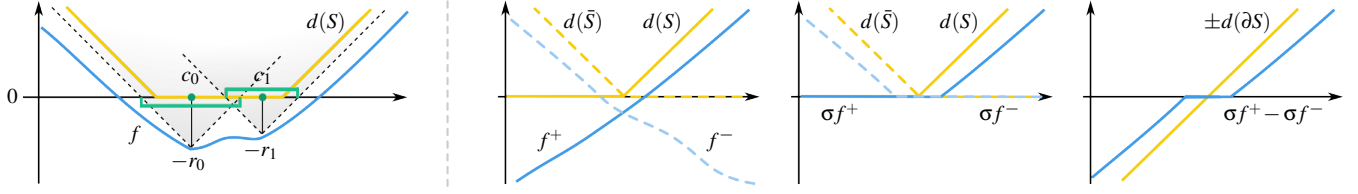


Figure 5: 1D illustration of Theorems 3.1 and 3.2. Left: True distance function (yellow) of a 1D shape (yellow interval) and a ball cover thereof with two balls (green intervals); the 1-Lipschitz function f (blue), if it passes sufficiently low below the ball centers, cannot enter the grey region that contains the true distance function. Right: True distance functions of a 1D shape and its complement (yellow, solid and dashed) with conservative DFs f^+ and f^- (blue, solid and dashed); clamping and combining them yields a conservative SDF.

3. Conservative Distances via Ball Covers

Definition 3.1 (*K*-Lipschitz). A function $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is *K*-Lipschitz w.r.t. the Euclidean norm if $|f(x_1) - f(x_2)| \leq K\|x_1 - x_2\| \forall x_1, x_2$. Where f is differentiable, this is locally equivalent to $\|\nabla f\| \leq K$.

Let $B_i = (c_i, r_i)$ define a ball with radius r_i centered at point c_i . The set of points contained in a ball B_i is referred to as $[B_i] = \{x \mid \|x - c_i\| \leq r_i\}$. For brevity, we will refer to both B_i and $[B_i]$ as ball.

Definition 3.2 (Ball Cover). A set of balls $\mathcal{B} = \{B_i \mid 0 \leq i < n\}$ is a ball cover of a point set S iff $S \subseteq \bigcup_{0 \leq i < n} [B_i]$.

In other words, iff any point of S is contained in at least one ball, the set of balls is a ball cover of S . Figure 6 illustrates this in 2D.

3.1. Conservative Distance Functions

Making use of the above definitions, our method builds on the following central theorems (illustrated in Figure 5):

Theorem 3.1 (Distance Underestimation). If $\mathcal{B} = \{B_i = (c_i, r_i)\}$ is a ball cover of S and $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is a 1-Lipschitz function that satisfies $f(c_i) \leq -r_i$ for all i , then f is a conservative DF, i.e. f globally underestimates the distance to S :

$$f(x) \leq d(x, S) \quad \forall x \in \mathbb{R}^n.$$

Proof. Let $x \in \mathbb{R}^n$ be arbitrary and let $s^* \in S$ be a closest point to x in S , so $\|x - s^*\| = d(x, S)$. Since $\mathcal{B}$ covers S , there exists a ball $B_j = (c_j, r_j) \in \mathcal{B}$ with $s^* \in [B_j]$, i.e. $\|s^* - c_j\| \leq r_j$.

By the triangle inequality, the distance from x to the ball center c_j is bounded by the two legs via s^* :

$$\|x - c_j\| \leq \|x - s^*\| + \|s^* - c_j\| \leq d(x, S) + r_j.$$

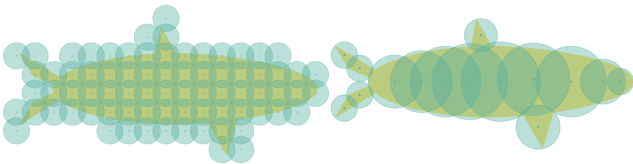


Figure 6: Illustration of 2D ball covers (green semi-transparent) of a shape (yellow). The left cover uses uniformly sized regularly arranged balls while the right cover uses adaptively sized strategically arranged balls, yielding a tighter cover with fewer balls.

Substituting this and $f(c_j) \leq -r_j$ (as assumed) into the inequality $f(x) \leq f(c_j) + \|x - c_j\|$ implied by 1-Lipschitzness of f yields:

$$f(x) \leq \underbrace{f(c_j)}_{\leq -r_j} + \underbrace{\|x - c_j\|}_{\leq d(x, S) + r_j} \leq -r_j + d(x, S) + r_j = d(x, S).$$

□

Corollary 3.1 (Sublevel Set Containment). If f is a conservative DF of S , then S is entirely contained in the zero-sublevel set $L_f^- = \{x \mid f(x) \leq 0\}$ of f .

Proof. As for any $x \in S$ we have $d(x, S) = 0$ by definition and $f(x) \leq d(x, S)$ by assumption, we immediately yield $f(x) \leq 0$. □

3.2. Conservative Signed Distance Functions

In case one is interested in representing not the distance to a shape S but the *signed* distance to a shape boundary ∂S , positive outside, negative inside, the following theorem is valuable. It essentially builds an SDF out of two DFs, f^+ and f^- (see Figure 5 right). $\bar{S}$ denotes the complement of S within some finite definition volume D , i.e. $\bar{S} = D \setminus S$, and $\sigma(x) = \max(0, x)$ is the ReLU function:

Theorem 3.2 (Signed Distance Underestimation). If f^+ satisfies the conditions of Theorem 3.1 for S , and f^- satisfies it for $\bar{S}$, then $f = \sigma(f^+) - \sigma(f^-)$ is a conservative SDF of ∂S within D , i.e.

$$\begin{aligned} |f(x)| &\leq d(x, \partial S) & \forall x \in D \\ f(x) &\geq 0 & \forall x \in \bar{S} \\ f(x) &\leq 0 & \forall x \in S \end{aligned}$$

Proof. By Theorem 3.1 $f^+(x) \leq 0 \forall x \in S$ and $f^-(x) \leq 0 \forall x \in \bar{S}$. With $\sigma(x) \geq 0$ this yields the latter two inequalities. Furthermore, $d(x, X) = d(x, \partial X) \forall x \notin X$. Hence we have

$$f(x) = \begin{cases} |f(x)| = \sigma(f^+(x)) \leq \sigma(d(x, S)) = d(x, \partial S) & \forall x \in \bar{S} \\ -|f(x)| = -\sigma(f^-(x)) \geq -\sigma(d(x, \bar{S})) = -d(x, \partial S) & \forall x \in S \end{cases}$$

□

Corollary 3.2 (Level Set Containment). If f is a conservative SDF of S , then ∂S is entirely contained in the zero-level set $L_f = \{x \mid f(x) = 0\}$ of f .

Note that here L_f is not generally a surface (unless it equals ∂S , i.e. when f^+ and f^- have zero VOL⁺) but rather a volumetric shell, or band, around ∂S . This is crucial to even enable conservativeness.

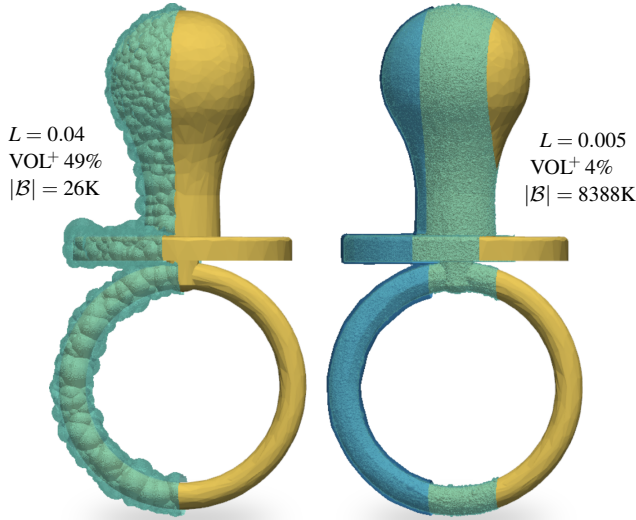


Figure 7: Ball covers (green semi-transparent, clipped view) of a shape (yellow) at two resolutions. Notice how tightness (measured by volume excess VOL^+) improves with increasing resolution (controlled by L , relative to the shape's bounding box extent). On the right, additionally a conservative neural implicit surface (blue semi-transparent) trained based on this ball cover is shown.

3.3. Ball Cover Construction

The above theorems hinge on ball covers of a given shape (and its complement). Note that these theorems formulate sufficient conditions for conservativeness, not necessary conditions. Their tightness is determined directly by the tightness of the ball cover $\mathcal{B}$ employed, which can be quantified via the relative volume excess

$$VOL^+(\mathcal{B}, S) = \frac{\text{vol}(\cup[B_i] \setminus S)}{\text{vol}(S)},$$

with 0 indicating perfect tightness. In particular, note that the ball cover's VOL^+ forms a lower bound for the VOL^+ (and thus a limit for the accuracy) of the zero-sublevel set L_f^- , i.e. of the shape implicitly represented by f . Figures 6 and 7 show ball covers of varying tightness for given shapes.

For practical purposes, a ball cover should therefore be tight, enabling high approximation accuracy within the conservative regime – but at the same time the number of balls should be low, as this number directly impacts the run time of algorithms making use of the above theorems. Hence, there is an inherent trade-off to be made. Exposing a single parameter that controls this trade-off is therefore desirable, allowing the user to balance approximation quality against computational cost as needed. As Theorem 3.1 does not put any additional requirements on the ball cover, a variety of approaches to their algorithmic construction are imaginable.

3.3.1. Delaunay Ball Cover

We propose a ball cover construction based on Delaunay techniques. It offers direct control over the tightness vs simplicity trade-off, includes adaptive sizing relative to local geometric complexity, and allows building on a well-proven theoretically solid foundation.

Assume S is a solid defined by a piecewise linear boundary ∂S , in the form of a triangle mesh or more generally a piecewise linear complex. We build a constrained Delaunay tetrahedralization [SG05] of S , and refine it using constrained Delaunay refinement [Si08] so as to meet a desired element sizing along ∂S , controlled by edge length limit L . The circumballs of the tetrahedra (spanned by a tetrahedron's four vertices) form a ball cover of S : each point in S is contained in a tetrahedron, which is contained in its circumball. We can, however, do better: The circumball is not generally the smallest ball containing the tetrahedron (whenever the circumcenter lies outside of the tetrahedron), hence instead we compute the *smallest enclosing ball* – which is the circumball of either the tetrahedron or its face of maximum circumradius or its edge of maximum length.

Figure 7 illustrates the resulting ball cover, depending on the sizing parameter L , which effectively controls the resolution – and thereby the tightness – of the ball cover. The choice of L is discussed in Section 5.1.

Note that we enforce the sizing limit L only on the boundary ∂S , leading to a *graded* mesh with typically much larger tetrahedra in the interior, conveniently implying large balls covering large parts of the volume. Note that, despite their large size, due to the empty circumsphere property of Delaunay meshes [Ede01] these interior balls cannot protrude out of S arbitrarily far. With decreasing L and thus increasing vertex density on the boundary, they are necessarily increasingly confined to the interior.

Alternatives. Potential alternative ball cover construction approaches, not explored in the present work, include:

- One circumball per voxel of a conservative voxelization (uniform or adaptive octree). Voxel intersections can be determined (conservatively) for all kinds of representations, such as triangle meshes [ZCEP07], parametric [PM09], or implicit [SK01].
- One (appropriately enlarged) ball per medial ball of a discrete medial axis transform with bounded approximation error [AK00; DLX*22].
- One ball per leaf of a bounding sphere tree [BO04; ACB17].

3.4. Neural Architecture

In order to learn a function f that is inherently 1-Lipschitz, thus does not require any additional technical measures to meet the conditions of Theorem 3.1, we make use of a fully-connected neural network composed of Semi-definite Programming Lipschitz Layers (SLL) [AHD*23] – essentially an affine layer with ReLU with a form of built-in normalization to keep the Lipschitz constant ≤ 1 . We opt for this choice because it was recently shown to perform well in the context of distance functions [CB24]. The network ends with a final (normalized) affine layer to yield a single output scalar.

4. Algorithms

Making use of the ball cover construction (Section 3.3) and the Lipschitz neural architecture (Section 3.4) and employing Theorems 3.1 and 3.2 we now suggest algorithms for generating, analyzing, and optimizing neural DFs and SDFs (or general neural

implicits) in the conservative regime. Let us remark that these algorithms are generic in the sense that they do not rely on this specific ball cover construction or this specific architecture. They only assume that the learnable function f is inherently 1-Lipschitz.

4.1. Verifying Conservativeness

Unsigned case Given a ball cover $\mathcal{B}$ of a shape S and a 1-Lipschitz neural function f we can easily perform the following test: For each ball $B_i \in \mathcal{B}$ feed its center c_i through the neural network f and check whether the output does not exceed the ball's negative radius $-r_i$. If this is the case for all balls, the neural function f is verified to be a conservative DF.

Signed case Given a ball cover $\mathcal{B}^+$ of a shape S , a ball cover $\mathcal{B}^-$ of its complement $\bar{S}$ and a 1-Lipschitz neural function f of the form $f = \sigma(f^+) - \sigma(f^-)$ we can easily perform the following test: For each ball $B_i^+ \in \mathcal{B}^+$ feed its center c_i^+ through the neural network f^+ and check whether the output does not exceed the ball's negative radius $-r_i^+$. Then, for each ball $B_i^- \in \mathcal{B}^-$ feed its center c_i^- through the neural network f^- and check whether the output does not exceed the ball's negative radius $-r_i^-$. If this is the case for all balls, the neural function f is verified to be a conservative SDF.

4.2. Establishing Conservativeness

Given a 1-Lipschitz neural function f that is not yet a provably conservative DF (via Section 4.1), we wish to modify it into a conservative state. This can be achieved by subtracting a positive constant δ of sufficient magnitude. Similar to the isosurface inflation of [LC24] this can easily be achieved by solely adjusting the bias weight b of the final network layer ($b \leftarrow b - \delta$), leaving all other network weights untouched. Importantly, this does not affect the Lipschitz constant of f . While this approach does not yield a minimal modification into a conservative state, it is unconditionally reliable and provides a feasible starting point for subsequent tightening (see Section 4.3).

A suitable constant can be determined by a single forward pass of the ball cover through f . Concretely: $\delta = \max_i \{\sigma(f(c_i) + r_i)\}$. Then $f^\delta = f - \delta$ is a conservative DF; it satisfies the condition of Theorem 3.1, because for all i :

$$f^\delta(c_i) = f(c_i) - \max_i \{\sigma(f(c_i) + r_i)\} \leq f(c_i) - (f(c_i) + r_i) = -r_i.$$

In practice we increase δ by a small constant (10^{-6}) to guard against floating-point inaccuracies.

In the case of an SDF, both f^+ and f^- can individually be modified in this manner, yielding a provably conservative SDF.

4.3. Optimizing Tightness

The simple procedure of Section 4.2 guarantees a conservative outcome, but it is arbitrarily non-tight, VOL^+ and DE_{abs} are arbitrarily large. We now describe how to improve this through optimization, while maintaining the conservativeness guarantee (Section 4.3.2). We initially consider the case of a DF (for the outside of S); for an SDF the process, as before, can be applied to two complementary DFs (Section 4.3.3).

As training data we use a set of points, consisting of two subsets: set P_{in} of points inside S and set P_{out} of points outside of S (in $\bar{S} = D \setminus S$). As P_{in} we can conveniently reuse the set $\{c_i\}$ of ball centers of the ball cover $\mathcal{B}$; note that these, when constructed as in Section 3.3.1 always lie inside S because a tetrahedron contains its smallest enclosing ball's center. For $P_{\text{out}} = \{p_j\}$ we uniformly sample n_{out} points from $\bar{S}$. This binary labeling of data points is sufficient, ground truth distance labels are not required by the employed loss.

4.3.1. Loss

To fit our setting, we modify the hinge-Kantorovich-Rubinstein (hKR) loss [SMG*21], used for training Lipschitz distance fields [CB24], combining a hinge term $\mathcal{L}_{\text{hinge}}$ with a KR term $\mathcal{L}_{\text{KR}}$:

$$\mathcal{L}_{\text{hKR}} = \mathcal{L}_{\text{hinge}} + \gamma \mathcal{L}_{\text{KR}}, \quad (1)$$

where $\gamma \geq 0$ regulates the relative influence of the two terms.

$\mathcal{L}_{\text{KR}}$ maximizes the magnitude of the network output by minimizing it where it should be negative (inside S) and maximizing it where it should be positive (outside of S):

$$\mathcal{L}_{\text{KR}}(\theta) = \frac{1}{\sum_i \text{vol}(B_i)} \sum_{i=1}^{n_{\text{in}}} \text{vol}(B_i) f(c_i) - \frac{1}{n_{\text{out}}} \sum_{j=1}^{n_{\text{out}}} f(p_j). \quad (2)$$

It is due to the Eikonal property of distance fields ($\|\nabla f\| = 1$) that minimization of this loss in a setting with Lipschitz constant 1 leads to approximation of the SDF, without the need for regression based on ground truth distance values. In the left (inside) term we use relative ball volume weights (with $\text{vol}(B_i) = \frac{4}{3}\pi r_i^3$) to account for varying ball sizes: a region covered by a large ball contributes only one sample, just as a region covered by a tiny ball does; volume weighting ensures each center represents the region it covers proportionally. The right (outside) term uses points that are drawn uniformly from D , so a plain mean provides an unbiased weighting.

$\mathcal{L}_{\text{hinge}}$ aims to tighten specifically the zero-level set towards ∂S , by focusing on points with values near zero, within a narrow margin, and penalizing improper signs:

$$\mathcal{L}_{\text{hinge}}(\theta) = \frac{\lambda}{\sum_i \text{vol}(B_i)} \sum_{i=1}^{n_{\text{in}}} \text{vol}(B_i) \max(0, m^{\text{in}} + r_i + f(c_i)) + \frac{1}{n_{\text{out}}} \sum_{j=1}^{n_{\text{out}}} \max(0, m^{\text{out}} - f(p_j)). \quad (3)$$

Note that in contrast to former uses of a hinge loss, here we introduce the offset r_i to account for the fact that the point c_i actually represents a (supposedly interior) ball of radius r_i . With this, $\mathcal{L}_{\text{hinge}}$ reaches its theoretical minimum when all outside points attain a sufficiently positive value (above m^{out}) and the condition of Theorem 3.1 is well satisfied, i.e. $f(c_i) \leq -(r_i + m^{\text{in}})$. Here $m^{\text{in}}, m^{\text{out}} > 0$ are margin hyperparameters. Their choice and that of the loss balancing parameter γ are discussed in Appendix A.

The weight $\lambda \geq 0$ controls the relative weight of inside versus outside and can be used to (more or less strongly) encourage conservativeness during training [LFYR24]; we exploit this below.

4.3.2. Maintaining Conservativeness

To maintain the initially established conservativeness throughout the process of optimizing f , i.e. incrementally updating the neural network's learnable parameters based on the loss, we control these updates: Per epoch, parameter updates are performed only tentatively. At the end of an epoch, if conservativeness can still be verified (Section 4.1) the updates are accepted. Otherwise, they are discarded. In case of acceptance the above weight λ in the hinge loss is decreased, in case of discardance increased (default factors stated in Section 5). In this manner, λ self-regulates with the current training situation. Intuitively, it determines how “repellent” the interior region is for the zero-level set, relative to the exterior. Hence, in case of discardance the next attempt will be more restrained; in case of acceptance, the next epoch can make a larger step.

Note that this strategy, inspired by a former non-distance conservativeness preservation approach [LC24], has conceptual relations to interior-point methods with adaptive barrier penalty [BV04], or to line search step size adaptation techniques. It could also be applied on the batch level rather than the epoch level. Empirically, however, the latter proved to perform better. Also, instead of discarding an update entirely, a truncated version thereof could be accepted, in the style of a back-tracking line search. This proved to be difficult to tune and to coordinate with updates of λ towards consistently favorable behavior, though.

4.3.3. Signed Case

If an SDF rather than a DF is sought, f is combined from two conservative DFs, $f = \sigma(f^+) - \sigma(f^-)$, as described in Theorem 3.2. These can be initialized and optimized, as above, separately and then combined. Alternatively, they can be integrated into one neural network: The final (normalized) affine layer (Section 3.4) is changed from a $\mathbb{R}^k \rightarrow \mathbb{R}$ to a $\mathbb{R}^k \rightarrow \mathbb{R}^2$ configuration, and the two outputs treated as f^+ and f^- during initialization and optimization. At inference time, a ReLU and difference are appended.

5. Experiments

We analyze our method's behavior in terms of choices of ball cover (Section 5.1) and network configuration (Section 5.2), and provide comparisons to the two most closely related state-of-the-art methods, one for conservative neural implicits (Section 5.3), i.e. the task of learning zero-level sets without regard for distance values, and one for neural distance fields (Section 5.4). The employed implementation is available at github.com/IngmarLudwig/ConservativeNeuralDistanceFields.

To construct the Delaunay ball covers (Section 3.3), we opt for TetGen [Si15] (with moderate element quality settings: min. dihedral angle 10° , max. radius-edge ratio 1.4) to determine the underlying tetrahedralizations of the input shapes, enabling its boundary refinement option with an edge length upper limit of L .

As training data $P^{out} = \{p_j\}$ for the complementary space, we sample $n_{out} = 32M$ points uniformly from $\bar{S}$ (or S , respectively), to be used for all epochs. As definition volume D we use an axis-aligned bounding box of S , expanded by an offset of 0.1.

The neural network f is initialized randomly and then made

conservative as described in Section 4.2, such that the condition $f(c_i) \leq -r_i$ is satisfied for all balls in $\mathcal{B}$ before optimization starts. To account for variance due to random initialization, in the following all quantitative plots and tables show averages over two training runs, unless otherwise noted.

Optimization. In our experiments, we use relatively generous training settings, as we want to gauge what can be achieved in terms of approximation quality with the various configurations and alternatives, rather than focusing on tuning for highest run time efficiency at this point. Network f is trained with the Adam optimizer for 7 stages of 300 epochs each. The model with lowest VOL^+ among all epochs' results is finally returned. Each epoch iterates over mini-batches of 2^{14} ball centers drawn without replacement; P^{out} is partitioned into the same number of batches so that every ball and every point is seen exactly once per epoch. The loss is $\mathcal{L}_{hKR}$ (1), with parameters as stated in Appendix A; only for experiments where we are not interested in distances but only the surface, γ is set to 0 so as to focus on pure zero-level set optimization.

All default hyperparameter settings, used unless stated otherwise for individual experiments, are listed in Appendix B.

Input Shapes. As example shapes for demonstration we make use of the 8 shapes also used in [LC24] (in particular for comparisons against that method) and, to increase diversity, 15 additional shapes from the Thingi10K and SHREC'16 dataset. Table 1 lists all of them. They are all scaled into $[-1, 1]^3$.

Metrics We evaluate volume excess VOL^+ and distance error DE_{abs} , DE_{rel} , defined in Section 1.1. In the computation of DE_{rel} , points extremely close to the surface (within 10^{-5}) are excluded to yield meaningful results, avoiding division by zero and numerical blow-up. Analogous to VOL^+ we also define the volume deficiency

$$VOL^-(f, S) = \frac{\text{vol}(S \setminus \{x \mid f(x) \leq 0\})}{\text{vol}(S)}.$$

While this is always 0 for our method, it is valuable for comparison to other methods. For the assessment of (non)conservativeness of

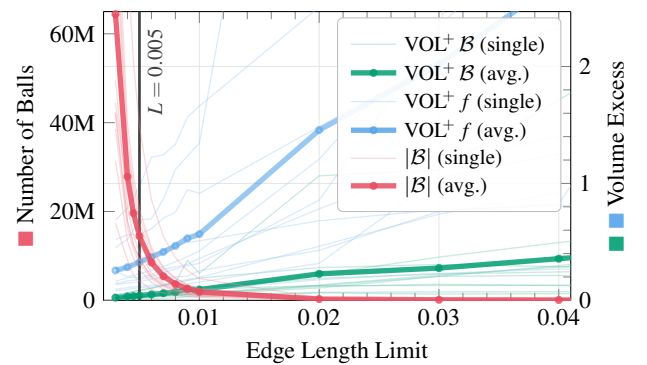


Figure 8: Average (over 11 shapes) volume excess (VOL^+) of the ball cover (green), VOL^+ of neural implicits trained with ball covers constructed at the respective L (blue), and number of balls in the ball cover (red) as a function of the maximum edge length L . The chosen default value $L = 0.005$ is marked with a line.

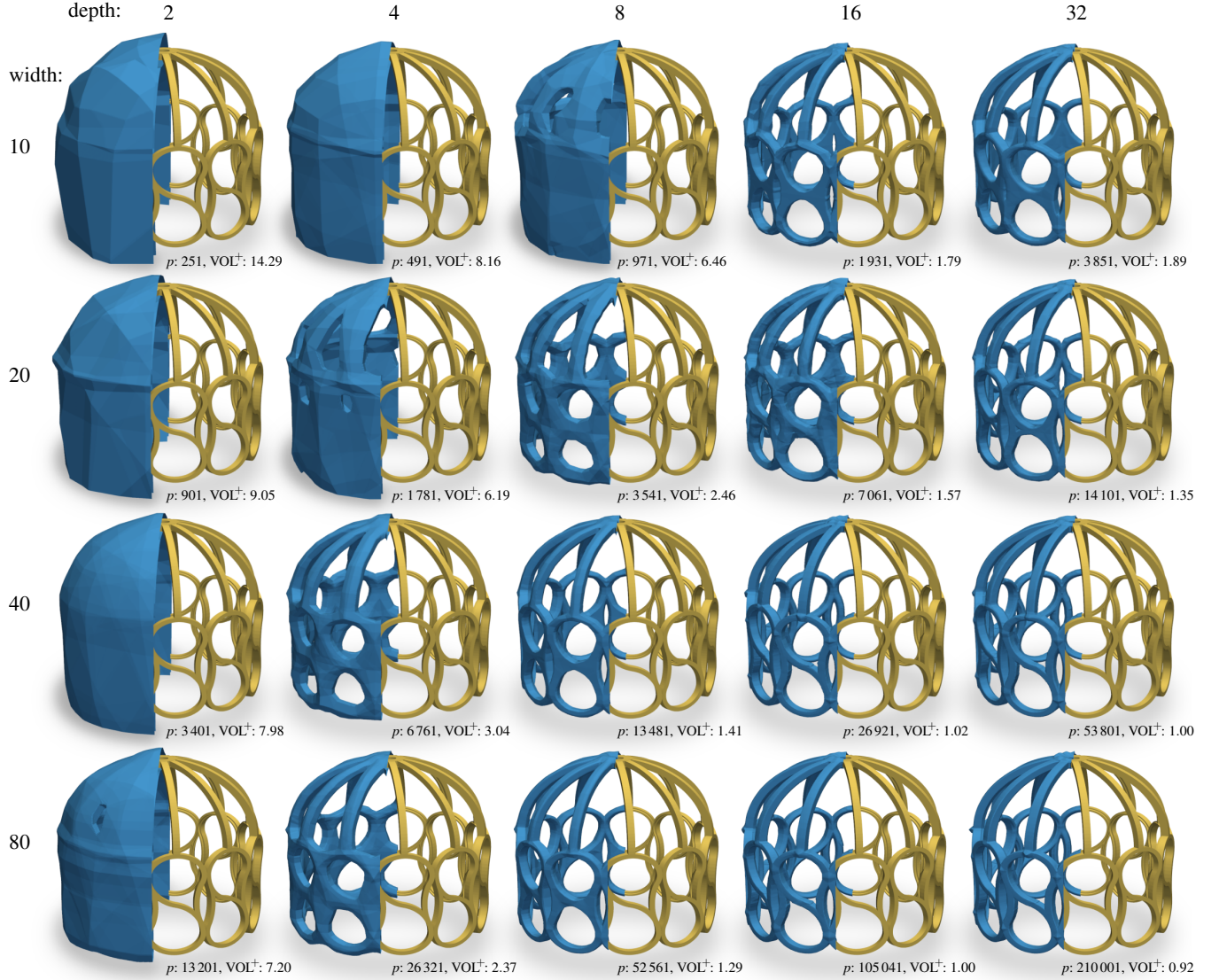


Figure 9: Zero level sets of conservative neural implicits for the CAGE shape across different network configurations in terms of layer width w and number of hidden layers d . The number of neural network parameters (p) and volume excess (VOL^+) are shown for each.

distances, we furthermore define the distance overestimation rate, estimated via a point sample $\{p_i\}$ of size n :

$$DOR(f, S) = \frac{1}{n} |\{p_i \mid f(p_i) > d(p_i, S)\}|.$$

5.1. Ball Cover Resolution

The resolution and tightness of the ball cover can be adjusted by parameter L . The larger this value, the smaller the number of balls (i.e. the faster the training) but the less tight the shape and distance approximation by f . Based on experimental results (configuration $[w: 50, d: 5]$, Section 5.2), reported in Figure 8, we choose $L = 0.005$ (assuming the input is scaled to a unit bounding box $[-1, 1]^3$) as a default for all subsequent experiments. A further decrease of L typically shows diminishing returns in terms of tightness of f .

The difference between the VOL^+ of $\mathcal{B}$ and that of f in Figure 8 can be explained as follows. For large L , the ball cover with large balls forms deep valleys between adjacent balls that protrude out of S , and the neural implicit lacks the representational power to reach into every such valley, causing excess volume on top of that already present in $\mathcal{B}$. As L decreases, these valleys become shallower, enabling a tighter fit against $\mathcal{B}$ (Figure 7 right). Eventually, the geometric complexity of S itself becomes the dominant factor.

5.2. Network Architecture: Depth and Width

We explore how the neural network architecture configuration, its depth d (number of hidden layers) and its width w (neurons per hidden layer) affect both the volume error (VOL^+) and the distance error (DE_{abs} and DE_{rel}). Figure 9 shows visual results on an exam-

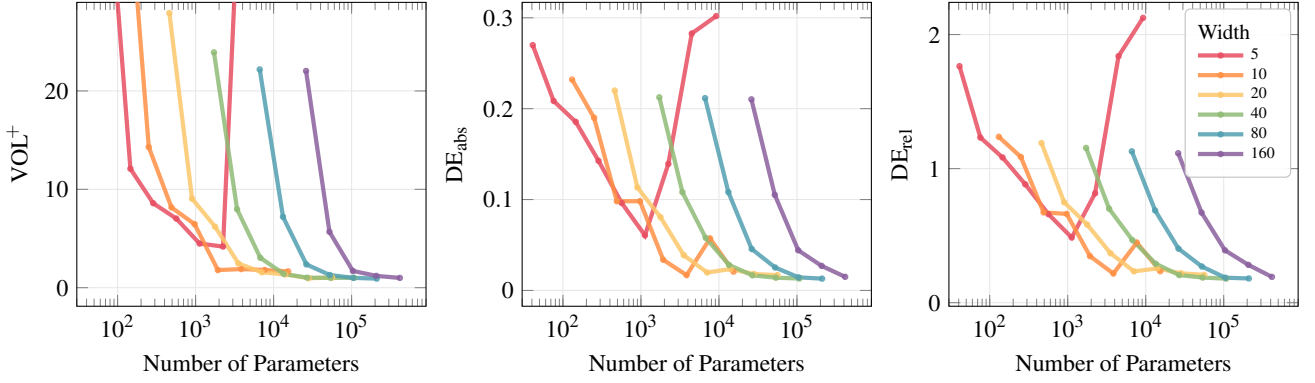


Figure 10: Various tightness measures (VOL^+ , DE_{abs} , DE_{rel}) versus number of parameters for different network widths and depths on the CAGE shape from Figure 9. Data points of the same width are connected for clarity.

ple shape for 5×4 different configurations. These results are also presented quantitatively in Figure 11, across an even wider range of configurations. Note how, per width, an increasing depth typically monotonically improves the VOL^+ – with the exception of very narrow very deep configurations, a combination with known performance limitations [LSK18]. The VOL^- , as expected, is always 0, i.e. the resulting zero-level sets are conservative.

When plotting against network parameter count rather than depth (Figure 10), it can be observed that, for any given parameter budget, it is typically a rather low-width-high-depth configuration that fares best. The same pattern holds for distance errors DE_{abs} and DE_{rel} . Again, however, very low widths show limited performance at high depths. Wider networks ($w \geq 40$) do not show such instability within our test range.

It is worth noting that parameter count does not directly translate to training time or inference time requirements. Wider layers parallelize better on modern GPUs, so the optimal architecture configuration choice depends on whether one aims to minimize neural model size or wall-clock time. Widths around 40 showed the best quality over training time ratio.

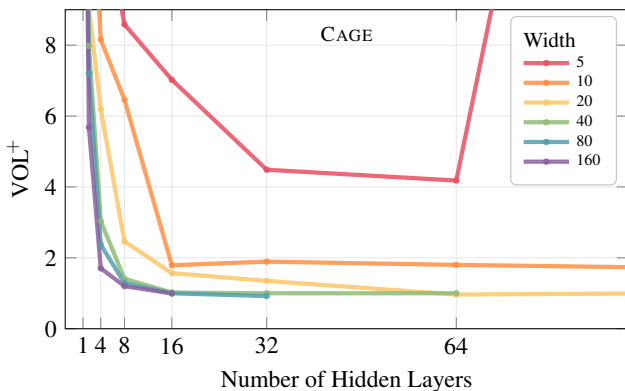


Figure 11: Volume excess (VOL^+) versus number of hidden layers for different network widths on the CAGE shape from Figure 9.

5.3. Comparison with CNI [LC24]

We compare our method with Strictly Conservative Neural Implicits (CNI) [LC24] regarding zero-level set tightness. For both methods their respective default training setup is used. Both methods are evaluated at widths $w \in \{25, 50, 75, 100\}$ and depths $d \in \{1, \dots, 6\}$. We show individual results for three example shapes in Figure 12, and joint results averaged over all the mesh shapes also used in the CNI paper in Figure 13.

The fundamental technical difference between the two approaches lies in how they enforce conservativeness: CNI relies on range analysis using affine arithmetic, whose error margin inherently grows with each additional (non-affine) layer, whereas our Lipschitz and ball cover based approach is, in theory, independent of network depth. As a result, CNI's VOL^+ typically improves only up to $d = 2$ or 3 and starts degrading from there, regardless of width (Figure 13), whereas our method shows further monotonic improvement far beyond that (Figures 11 and 13).

In absolute terms, at $d = 1$ (yielding very coarse approximations only) CNI can achieve better results than our method (at least for high widths); in such extremely shallow networks range analysis is still tight. But for $d > 2$ our method overtakes and consistently yields better results, across widths. At $d = 6$ our method approximately halves the VOL^+ that can be achieved by CNI at its best depth (Figure 13 left, Appendix C).

These differences in characteristics furthermore are interesting in terms of parameter efficiency: CNI is effectively limited to low depths d , therefore requires high widths w – and the number of network parameters grows linearly with d but quadratically with w . Our method is more flexible in this regard. As can be observed in Figure 12, using our method with lower width but higher depth than CNI, often equal quality can be achieved with much fewer parameters, or better quality at similar parameter count. The effective depth limitation of CNI becomes especially apparent on shapes with complex topology: Figure 3 visually demonstrates this for an example shape – using the configuration $[w: 100, d: 4]$, the depth that for CNI leads to its best possible result. CNI produces a visibly loose enclosing surface, unable to fit the intricate geometry and topology. Our method delivers a substantially tighter zero-level set.

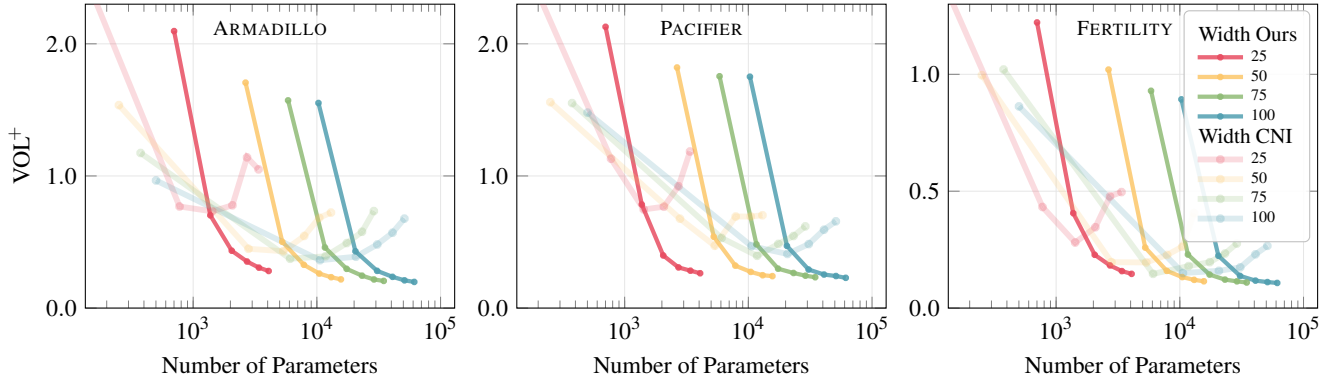


Figure 12: VOL^+ versus number of parameters for our method and CNI [LC24] on three example shapes. Data points for the same network width are connected for clarity. Network depths (1, ..., 6). Each point is the average of 10 runs. Note that the two methods use different types of layers; this is the reason for equal network depths corresponding to different numbers of parameters.

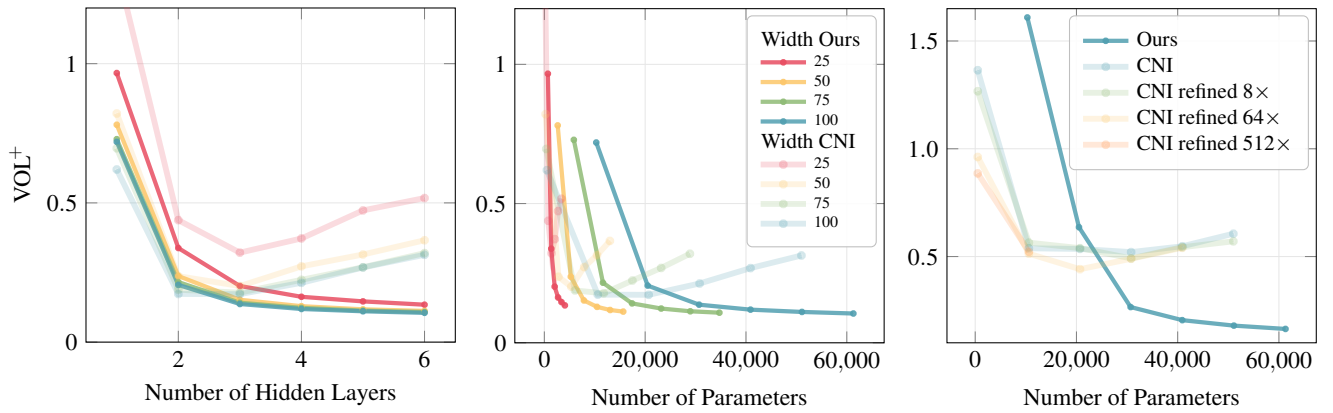


Figure 13: Comparison of our method and CNI [LC24]. Left and center: VOL^+ versus numbers of hidden layers and parameters, averaged over the 8 mesh shapes used in the CNI paper. Data points for the same network width are connected for clarity. Right: VOL^+ versus number of hidden layers (exemplarily for $w = 100$ on the shape from Figure 3), with 1, 2, or 3 additional levels of range refinement in CNI [LC24].

To investigate to what extent CNI’s depth limitation can perhaps be mitigated, we tested a range subdivision procedure. Each volumetric input range is split along each axis, i.e. into 8 smaller sub-ranges, before feeding it into the system. Conceptually, this should allow for tighter error bounds in the range analysis system. Figure 13 right shows the results for the same shape from Figure 3. It can be seen that this refinement improves VOL^+ to some extent, but it does not resolve the fundamental degradation at greater depths and even with multiple levels of refinement (splitting into 64 or 512 subranges) does not yield a quality close to our method’s. Also, this refinement drastically increases training time per epoch, by factors of $8\times$, $64\times$, or $512\times$ – in the latter case exceeding a set time limit of 1 week for $d > 2$ (hence the only partial plot line).

5.4. Comparison with 1-Lip [CB24]

We compare our method with 1-Lipschitz Neural Distance Fields (1-Lip) [CB24] regarding zero-level set tightness (VOL^+) as well as distance accuracy (DE_{abs} , DE_{rel}) on all shapes of our test dataset. For fairness, we use that method’s default configuration [w : 128,

d : 20] for both methods, and the respective default training setup. We first focus on one-sided DFs on the outside of shapes S . Quantitative results are reported in Table 1, cross-section visualizations in Figure 14, and plots of distance error in Figure 15. Let us point out that the main purpose of these comparisons is to demonstrate fundamental differences and to gauge trends, rather than to derive quantitative benefit statements – the latter is hard to do fairly in an abstract setting, as the concrete numbers depend on various hyper-parameter settings, configuration choices, and training setups.

The central property distinguishing our method is the guaranteed conservativeness, not only of the zero-level set, but of the distance field. Hence, as expected, across all shapes our method’s results have a VOL^- of 0 and a DOR of exactly 0%, i.e. the field’s distance did not exceed the true distance at any test point. By contrast, 1-Lip overestimates distances on most shapes: on average at 3.25% of the test points the field’s distance exceeds the ground truth, in the worst case 18.30% (Table 1). This is visible as well in the distance error visualizations in Figure 14: the red regions indicate regions where the neural DF overshoots the true distance. The distance er-

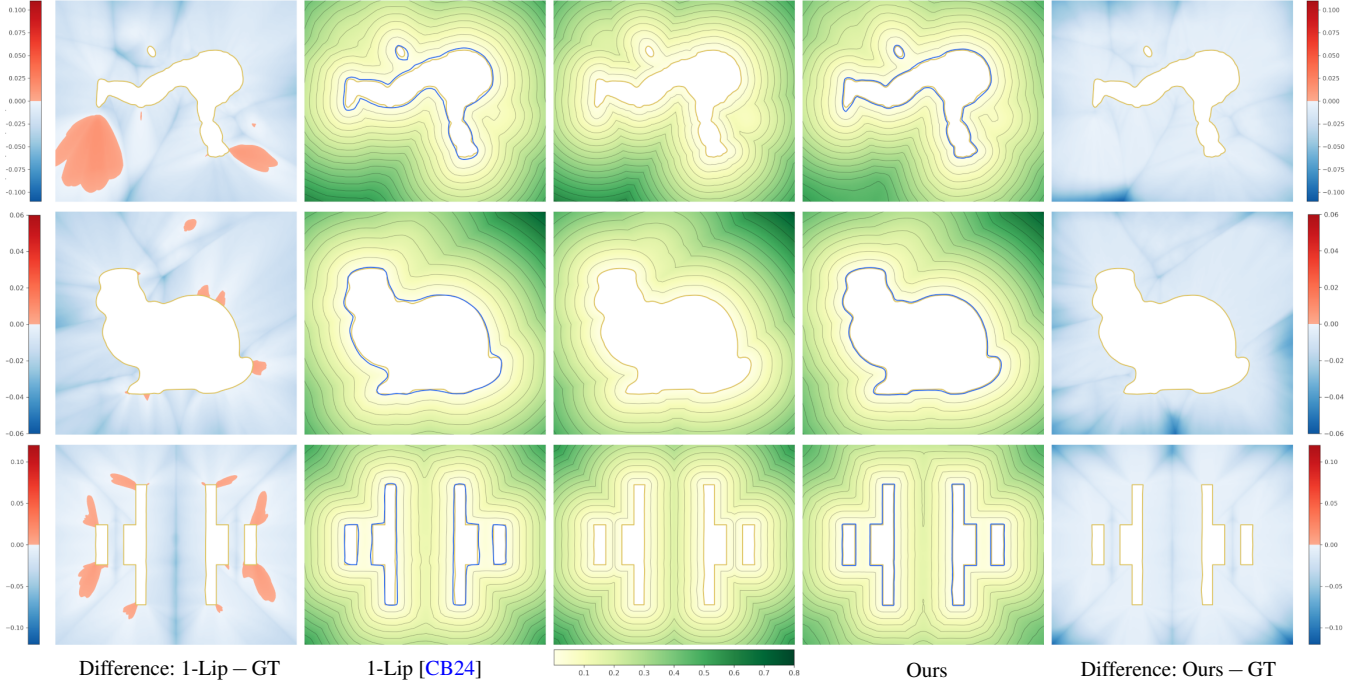


Figure 14: Cross-sections through DFs of 1-Lip [CB24] and our method, both at $[w: 128, d: 20]$. Ground truth (GT) shown in the center. Our fields are rescaled to the coordinate system of [CB24] for comparability. Shapes: ARMADILLO, BUNNY, SOCKET.

Table 1: Comparison of DFs computed using 1-Lip [CB24] and our method at $[w: 128, d: 20]$. VOL^+ , VOL^- , and DOR are specified in % here for compactness. Distance related values are computed based on a point sample of 5M points over each shape’s axis-aligned bounding box enlarged by 25% in each direction. Red color indicates conservativeness violation of the zero-level set (VOL^-) and of distances (DOR).

Metric	Arches	Armadil.	Bigfoot	Bimba	Bone	Bones	Holder	Bunny	Cage	Cylinder	Drill	Duck	Fertility	Kitten	Knot	Octocat	Octopus	Tentacle	Pacifier	Weight	Pipes	Socket	Hand	Avg.
DE_{abs} 1-Lip	0.015	0.015	0.015	0.010	0.008	0.017	0.011	0.013	0.018	0.012	0.017	0.008	0.013	0.010	0.025	0.018	0.017	0.017	0.013	0.016	0.015	0.010	0.015	0.014
DE_{abs} Ours	0.018	0.019	0.019	0.012	0.012	0.020	0.025	0.015	0.013	0.016	0.032	0.008	0.011	0.014	0.015	0.032	0.027	0.022	0.027	0.011	0.011	0.013	0.020	0.018
DE_{rel} 1-Lip	0.189	0.110	0.103	0.079	0.068	0.114	0.168	0.089	0.179	0.166	0.139	0.065	0.114	0.074	0.219	0.116	0.112	0.115	0.098	0.153	0.190	0.109	0.099	0.125
DE_{rel} Ours	0.146	0.095	0.093	0.076	0.067	0.105	0.166	0.086	0.099	0.143	0.172	0.065	0.075	0.077	0.132	0.162	0.133	0.117	0.113	0.099	0.112	0.111	0.088	0.110
VOL^+ 1-Lip	52.559	27.118	20.262	8.575	15.124	39.611	77.112	7.762	223.363	17.808	71.865	3.908	20.554	6.953	92.986	38.849	43.489	27.054	32.525	26.423	13.968	8.113	29.520	39.370
VOL^+ Ours	27.298	16.556	12.931	7.465	10.568	26.769	38.639	6.773	88.977	11.957	54.155	4.289	11.678	6.305	48.418	40.234	33.617	18.486	20.194	15.858	8.863	7.903	16.806	23.250
VOL^- 1-Lip	0.009	0.187	0.163	0.222	0	0.002	0.002	0.255	0	0.207	0.021	0.144	0.008	0.180	0.009	0.210	0.115	0.030	0	0.138	0.136	0.384	0.006	0.106
VOL^- Ours	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DOR 1-Lip	0.035	18.332	8.065	11.009	0	0.032	0.078	5.545	0	0.771	7.129	6.529	0.104	2.061	0.009	2.849	0.349	0.117	0	0.411	0.217	10.939	0.163	3.250
DOR Ours	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

ror plots in Figure 15 confirm the same pattern: many test points appear above the zero-error bar for 1-Lip, none for our method.

This guarantee comes at a cost in absolute accuracy. The average DE_{abs} of 1-Lip is 0.014 versus 0.018 for our method. This is not surprising; conceptually, higher approximation quality should always be achievable in the non-conservative regime. Our method, however, achieves a lower relative DE_{abs} (0.110 vs. 0.125) and a lower VOL^+ (23.25% vs. 39.37%), indicating tighter approximation near the surface (where accuracy often matters most). This is visible in the cross-sections in Figures 1 and 14: our DFs often fol-

low the surface S somewhat more closely, while 1-Lip tends to have lower error farther away from the surface.

Signed Distances Fields Examples of two-sided, i.e. signed DFs, are shown in Figure 16. 1-Lip shows a non-symmetric behavior: as discussed above, distance overestimation typically affects around 3% of the outside volume; for the inside, it is commonly much higher, as is evident in the examples, where the distance magnitude is overestimated in major shares of the inside volume (92.6% and 81.6%). Note that wherever the zero-level set of 1-Lip does not per-

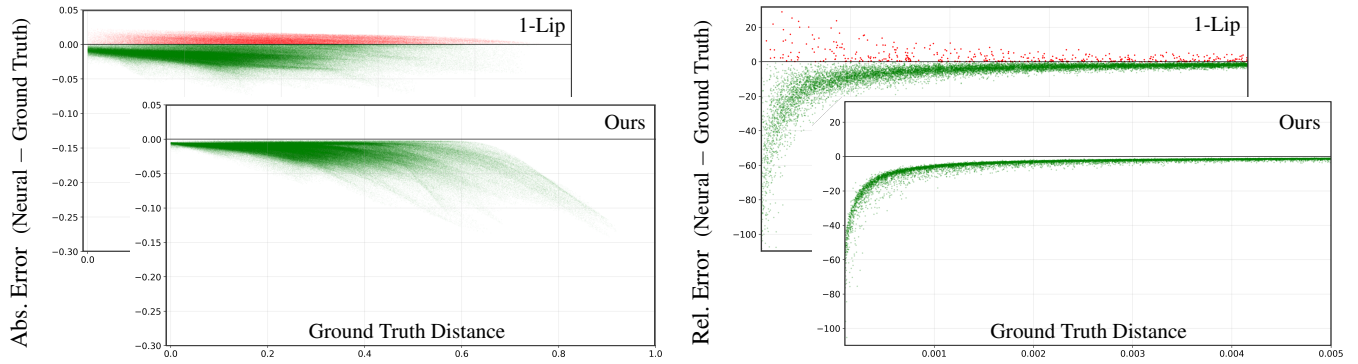


Figure 15: Absolute (left) and relative (right) distance differences between neural fields and the ground truth distance fields, for 5 000 000 random test points on the ARMADILLO shape. Top: 1-Lip [CB24]. Bottom: Ours. Both models use $[w: 128, d: 20]$. The right plots clip at 0.05 to focus on the near-surface region. Red indicates distance overestimation, i.e. non-conservativeness.

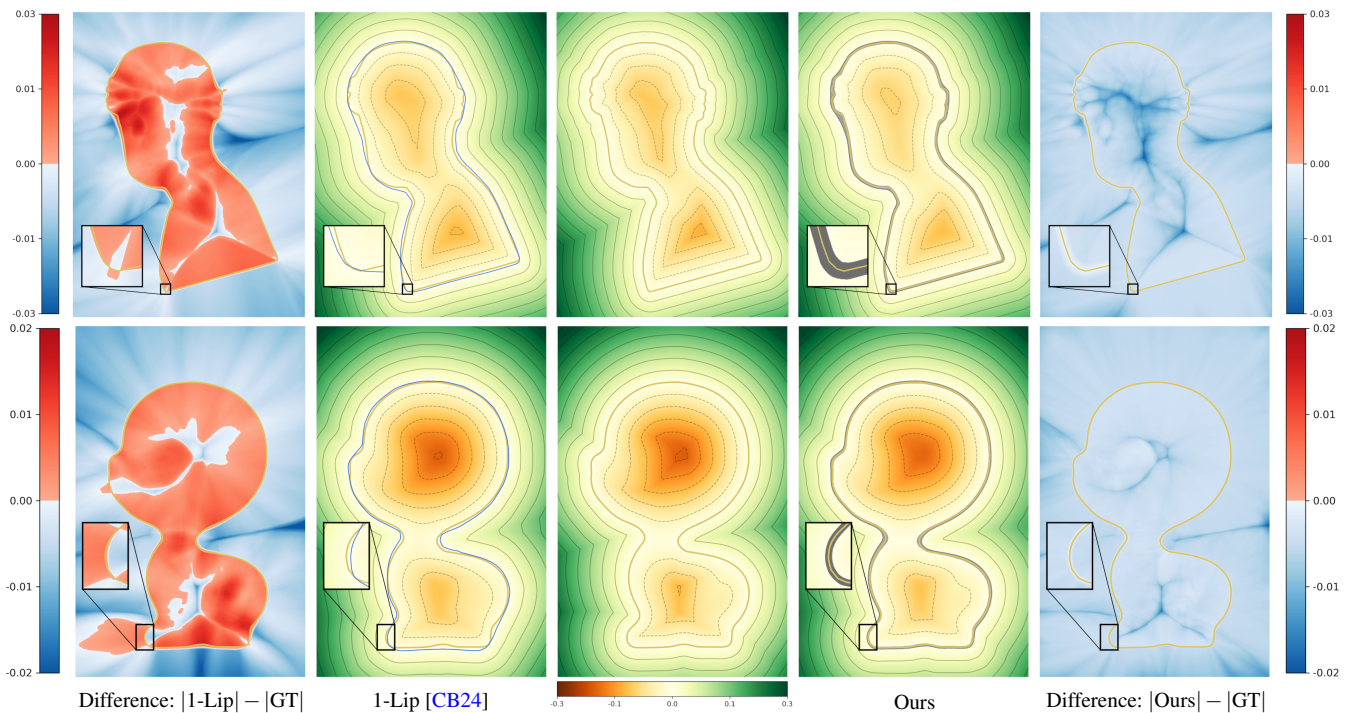


Figure 16: Cross-sections through SDFs of 1-Lip [CB24] and our method, both at $[w: 128, d: 20]$. Ground truth (GT) shown in the center. Our fields are rescaled to the coordinate system of [CB24] for comparability. In the zoom insets, notice how the zero-level set (gray) in our case is a thin band (containing the input surface) rather than a surface (meandering around the input surface). Shapes: BIMBA, KITTEN.

fectly match the surface ∂S , either an adjacent inside or outside region inevitably suffers from distance overestimation (see zoom insets). Our dual-DF based SDF construction (based on Theorem 3.2) circumvents this by effectively wrapping ∂S in a zero-level band.

5.5. Further Analysis

In Appendix D we analyze an alternative loss (binary cross-entropy), perform an ablation of the volume weighting in the losses, and discuss run time performance.

6. Conclusion

We presented a first method for generating neural signed distance fields that are guaranteed to be conservative: the predicted distance never overestimates the true distance anywhere, and the zero level set provably contains the original shape. The method combines 1-Lipschitz neural network architectures with ball-cover-based underestimation tests and employs a controlled training strategy that maintains conservativeness, ensuring it holds throughout training.

Compared to a previous approach for conservative neural im-

plicitis based on range analysis [LC24], our Lipschitz-based formulation circumvents their fundamental depth limitation: deeper networks enable monotonically tighter approximations, without the increasing error margins inherent to interval arithmetic. Experiments confirm that deeper, narrower architectures often outperform shallower, wider ones at comparable parameter budgets, demonstrating the practical value of this property.

In comparison with 1-Lipschitz Neural Distance Fields [CB24], which exhibit self-conservativeness with respect to their own zero-level set, our method achieves guaranteed conservativity with respect to any given shape. This naturally comes at a cost, lower approximation tightness and higher training time.

Limitations and Future Work

The ball cover construction via Delaunay enclosing balls introduces an initial excess that bounds the achievable tightness; investigating alternative ball cover construction strategies that produce tighter covers could reduce this. The use of other Lipschitz architectures is another promising direction.

Regarding run time, our current implementation makes use of uniformly sampled outside points for training; replacing this with an adaptive, surface-aware sampling strategy [CB24; DNJ20] could improve convergence speed (and also tightness). Coarse-to-fine training schemes, e.g. exploiting hierarchical ball covers built using methods for bounding sphere hierarchy construction [BO04], can be another path towards increased efficiency.

As an inevitable consequence of conservativeness without exactness, in the two-sided case the zero-level set is a band, i.e. a plateau in the field. While this is not an issue for some downstream methods, including sphere tracing for rendering or Marching Cubes for iso-surfacing (used for the visualizations herein) which effectively consider the band's outer or inner surface, it may constitute a problem for others, for instance due to the vanishing gradient, though we are unaware of concrete examples.

Acknowledgements

The authors thank Guillaume Coiffier and Louis Béthune for making their code [CB24] available, directly benefiting this project. Open Access funding enabled and organized by Projekt DEAL.

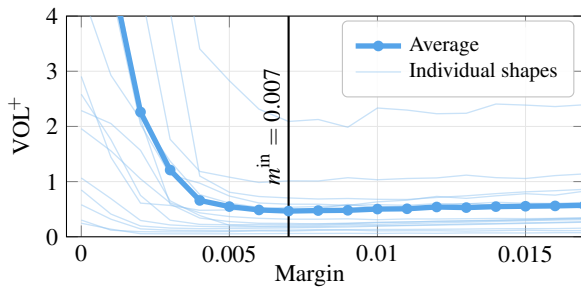


Figure 17: Average volume excess (VOL^+) of neural implicit trained with different inside margin m^{in} values, evaluated on 1M test points. The selected value $m^{in} = 0.007$ is marked with a line.

Appendix A: Loss Parameters

Margins

Smaller margins enable more accurate fits of the zero-level set in principle, but at some point – essentially when the margins become significantly smaller than the average spacing of training data points – results deteriorate because too few data points even have an impact in $\mathcal{L}_{hinge}$. This was discussed before [CB24] and is observed in our setting as well, see Figure 17. We selected the default value $m^{in} = 0.007$ empirically by training on a set of shapes across a range of margin values and choosing the value that minimized the average volume excess. Note that this is directly connected to the choice of L , which dictates the ball size, thus ball center spacing, near the boundary; hence for choices of L deviating from our default (Section 5.1), m^{in} should ideally be adjusted as well.

Due to our non-symmetric setting, with a focus on conservativeness, we set the outside margin m^{out} differently, in a shape adaptive manner. Concretely, the volume of $\tilde{S}$ is computed and divided by the number of outside points n_{out} , yielding the average volume per point; m^{out} is then set to the radius of a sphere of that volume. This makes m^{out} approximately proportional to the average spacing between neighboring outside points, adapting automatically to the sampling density.

Balance

We selected a default value for γ , balancing the two loss terms, empirically by training on a set of shapes across a range of values and evaluating approximation quality by means of DE_{rel} . We prefer this relative error notion as the selection criterion because it weighs errors proportionally to the scale of the true distance. Results are shown in Figure 18; based on this we selected $\gamma = 0.006$. Applications that require accurate distance estimates far from the surface, however, may be better served by low DE_{abs} values. The right plot in the figure indicates that an increase of γ (to around 0.02) yields small additional benefits in that scenario.

Appendix B: Hyperparameter Settings

All default hyperparameter values are listed in Table 2, with the justification source stated for each.

Table 2: Default hyperparameter values. $\gamma = 0.006$ if distances matter, $\gamma = 0$ if only the surface matters.

Parameter	Sym.	Value	Justification
Max edge length	L	0.005	VOL^+ sweep, Section 5.1
$\mathcal{L}_{KR}$ weight	γ	0.006 / 0	DE_{rel} sweep, Appendix A
Inside margin	m^{in}	0.007	Sweep, Appendix A
Outside margin	m^{out}	per-shape	Formula, Appendix A
BBox offset	—	0.1	Focus on near-surface region
Outside/inside pts	n	32M	Empirical, reduce for mem. constr.
Epochs / stage	—	300	Empirical, reduce for time constr.
Batch size inside	—	2^{14}	Empirical
Learning rate: init.	—	0.00125	Empirical
reduction	—	0.7	Empirical
stages	—	7	Empirical
λ adjust factors	—	0.96 / 1.3	Empirical

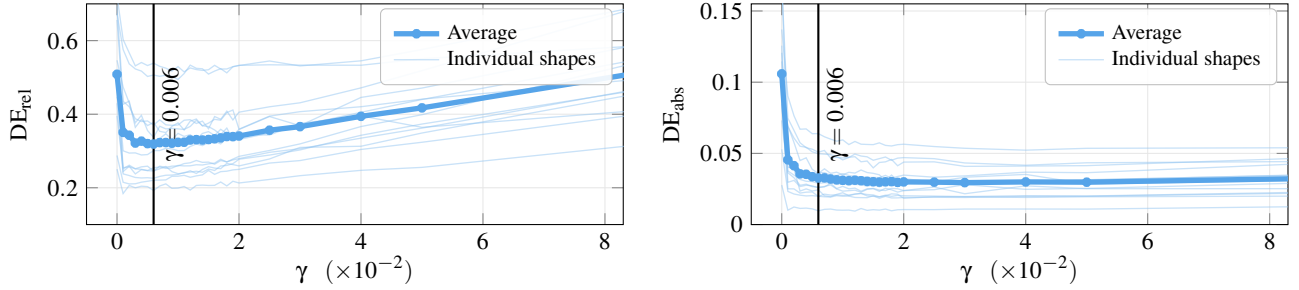


Figure 18: Relative distance error (DE_{rel}) and absolute distance error (DE_{abs}) of DFs trained with different γ values, evaluated on 1M test points per shape. Based on this, $\gamma = 0.006$ is selected as default.

Appendix C: Comparison with CNI

Figure 19 gives a visual impression of the tightness differences observed quantitatively in Figure 12.

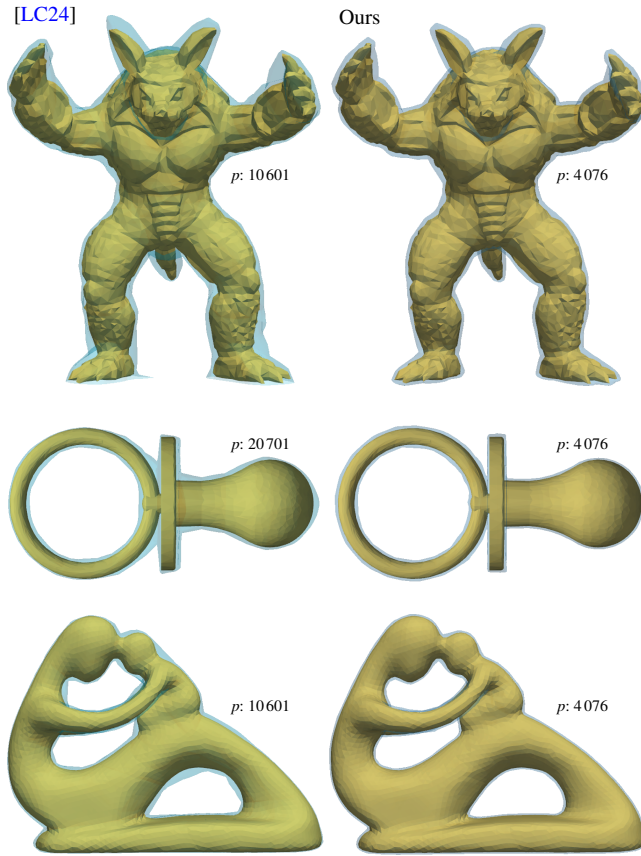


Figure 19: Semi-transparent neural implicits (blue) around given shapes (yellow) from Figure 12. Left: tightest outcomes of CNI [LC24] across all width-depth combinations from $\{25, 50, 75, 100\} \times \{1, 2, 3, 4, 5, 6\}$. Right: our results for configuration $[w: 25, d: 6]$, tighter despite being simpler. Numbers of parameters are reported next to each result.

Appendix D: Further Experiments

We analyze an alternative loss (binary cross-entropy, demonstrating moderate benefits for the non-distance scenario), perform an ablation of the volume weighting in the losses (demonstrating its benefit), and discuss run time performance.

Loss Comparison: hKR vs BCE

Variants of the binary cross-entropy (BCE) loss have been employed for training neural implicits, e.g. in [LFYR24] and [LC24]. Based on the BCE loss for Lipschitz neural networks introduced in [BBS*22], we therefore evaluate whether substituting the BCE loss for our default hKR loss (1) can improve zero-level set tightness – in scenarios where distances are of no interest. We adapt the BCE loss to our setting as follows, using volume weighting and ball-radius offsets as in $\mathcal{L}_{hinge}$ (3):

$$\mathcal{L}_{BCE}(\theta) = -\frac{\lambda}{V(\mathcal{B}_S)} \sum_{i=1}^{n_{in}} V(B_i) \log s(-T(f(c_i) + r_i)) - \frac{1}{n_{out}} \sum_{j=1}^{n_{out}} \log s(T(f(p_j))), \quad (4)$$

where s is the sigmoid function and $T > 0$ is a temperature hyperparameter that controls the sharpness of the sigmoid transition, playing a role analogous to the margin m in the hinge loss [BBS*22]. We select a default value for T empirically in the same manner as the inside margin m^{in} in (3): by training on a set of shapes across a range of temperature values and choosing the one that minimizes the average VOL^+ , yielding $T = 550$.

Table 3 compares $\mathcal{L}_{BCE}$ against $\mathcal{L}_{hKR}$ (with $\gamma = 0$, our non-distance default) for several example shapes using network configuration $[w: 50, d: 5]$. Often, $\mathcal{L}_{BCE}$ yields moderately improved tightness of the zero-level set (at the expense of lower distance accuracy), with an VOL^+ lower by roughly 8% on average. In use cases that seek the lowest possible VOL^+ and do not care about distances at all, it can therefore be considered a valuable alternative.

Volume Normalization

As described in Section 4.3.1, the inside terms in $\mathcal{L}_{hinge}$ and $\mathcal{L}_{KR}$ employ ball volume weighting. The purpose of this relative volume weighting is to correct for the potentially non-uniform spatial

Table 3: Alternative loss comparison for non-distance case: VOL^+ of $\mathcal{L}_{hKR}$ ($\gamma = 0$, $m^{in} = 0.007$) and $\mathcal{L}_{BCE}$ ($T = 550$) for config. $[w: 50, d: 5]$.

	ARCHES	ARMADI.	BONE	BONES	CAGE	DRILL	FERTILITY	KITTEN	KNOT	OCTOPUS	PACIFIER	WEIGHT	Average
$VOL^+ \mathcal{L}_{hKR}$	0.46	0.16	0.10	0.32	2.09	0.70	0.12	0.06	1.01	0.59	0.22	0.20	0.50
$VOL^+ \mathcal{L}_{BCE}$	0.52	0.16	0.09	0.29	1.40	0.65	0.12	0.05	0.84	0.50	0.21	0.20	0.42
Difference	+14.85%	+2.05%	-12.25%	-8.02%	-32.89%	-8.10%	-2.19%	-11.57%	-16.99%	-15.07%	-3.61%	-0.10%	-7.82%

Table 4: Comparing VOL^+ , DE_{abs} , DE_{rel} with and without volume weighting in the loss, for configuration $[w: 50, d: 5]$. Notice that in the cases where “without” fares better, the difference is typically very small, whereas “with” often shows larger benefits.

	ARCHES	ARMADI.	BONE	BONES	CAGE	DRILL	FERTILITY	KITTEN	KNOT	OCTOPUS	PACIFIER	WEIGHT	Average
VOL^+ (w/)	0.495	0.220	0.112	0.371	2.022	0.770	0.141	0.073	1.039	0.644	0.240	0.222	0.529
VOL^+ (w/o)	0.711	0.217	0.117	0.389	7.754	1.419	0.135	0.072	1.249	0.900	0.300	0.228	1.124
DE_{abs} (w/)	0.037	0.044	0.017	0.036	0.034	0.071	0.029	0.030	0.047	0.052	0.047	0.034	0.040
DE_{abs} (w/o)	0.050	0.042	0.022	0.038	0.061	0.096	0.027	0.029	0.049	0.062	0.050	0.028	0.046
DE_{rel} (w/)	0.275	0.190	0.092	0.174	0.235	0.341	0.155	0.139	0.299	0.240	0.186	0.197	0.210
DE_{rel} (w/o)	0.347	0.183	0.111	0.182	0.378	0.469	0.146	0.136	0.317	0.290	0.213	0.184	0.246

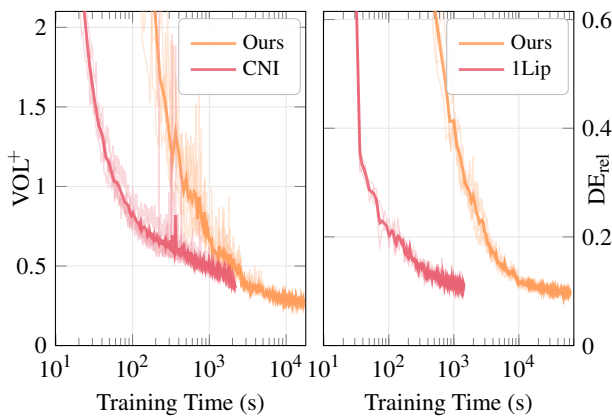
density of ball centers in the ball cover. Without this weighting, regions with dense tessellation in the Delaunay mesh (Section 3.3.1), typically near thin features or high curvature, would dominate the loss. This correction is not crucial but it typically improves result tightness as shown in Table 4.

Training Time

To exemplarily give an idea of the training time behavior of our method in comparison to other methods, we consider training for the ARMADILLO shape from Figure 4. Figure 20 left shows the progression of the VOL^+ over time during training in comparison to CNI [LC24]; for CNI its optimal configuration $[w: 100, d: 2]$ is used, for our method the configuration $[w: 50, d: 4]$ because it has a near equal parameter count. It can be seen that early in training CNI improves quicker, but towards convergence our method essentially catches up and overtakes. Figure 20 right shows the progression of the DE_{rel} over time during training in comparison to 1-Lip [CB24];

for both the default configuration $[w: 128, d: 20]$ of 1-Lip is used. It can be observed that it takes about an order of magnitude longer for our method to reach or beat the accuracy of 1-Lip. In part this is due to the generous training setup mentioned in Section 5, in part it is the cost of guaranteed conservativeness, requiring additional computation and limiting the space of solutions. As the maintenance of conservativeness (causing backtracking and retrial costs) is responsible for only around 15% of the overall training time in our experiments, there likely is significant potential for training time reduction through tuning of the various training parameters.

As a first test in this direction, we repeated training (on the shapes from Tables 3 and 4) with the number of points in P_{out} reduced from 32M to 1M. After readjusting hyperparameters (batch size inside $\times 4$, initial learning rate $+40\%$) this turns out to yield a training time reduction by factors of 3–5 (the number of P_{in} points becomes dominant) while increasing VOL^+ just mildly, by around 3% on average. Non-uniform sampling strategies likely have the potential to provide further improvements in this direction.

**Figure 20:** Progression over training time of VOL^+ in comparison to CNI (left) and of DE_{rel} to 1-Lip (right). Thin graphs represent 5 individual training runs, bold graphs their average.

References

- [ACB17] ARJOVSKY, MARTIN, CHINTALA, SOUMITH, and BOTTOU, LÉON. “Wasserstein generative adversarial networks”. *International conference on machine learning*. PMLR. 2017, 214–223 3, 5.
- [AHD*23] ARAUJO, ALEXANDRE, HAVENS, AARON J, DELATTRE, BLAISE, et al. “A Unified Algebraic Perspective on Lipschitz Neural Networks”. *The Eleventh International Conference on Learning Representations*. 2023 3, 5.
- [AK00] AMENTA, NINA and KOLLURI, RAVI KRISHNA. “Accurate and efficient unions of balls”. *Proc. 16th annual symposium on Computational geometry*. 2000, 119–128 5.
- [ALG19] ANIL, CEM, LUCAS, JAMES, and GROSSE, ROGER. “Sorting out Lipschitz function approximation”. *International conference on machine learning*. PMLR. 2019, 291–301 3.
- [BBS*22] BÉTHUNE, LOUIS, BOISSIN, THIBAUT, SERRURIER, MATHIEU, et al. “Pay attention to your loss: understanding misconceptions about Lipschitz neural networks”. *Advances in Neural Information Processing Systems* 35 (2022), 20077–20091 3, 14.

- [BNC*23] BÉTHUNE, LOUIS, NOVELLO, PAUL, COIFFIER, GUILLAUME, et al. "Robust One-Class Classification with Signed Distance Function using 1-Lipschitz Neural Networks". *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, 2245–2271 3.
- [BO04] BRADSHAW, GARETH and O'SULLIVAN, CAROL. "Adaptive medial-axis approximation for sphere-tree construction". *ACM Transactions on Graphics* 23.1 (2004), 1–26 5, 13.
- [BV04] BOYD, STEPHEN and VANDENBERGHE, LIEVEN. *Convex optimization*. Cambridge university press, 2004 7.
- [CB24] COIFFIER, GUILLAUME and BÉTHUNE, LOUIS. "1-Lipschitz Neural Distance Fields". *Computer Graphics Forum*. Vol. 43. 5. 2024, e15128 2, 3, 5, 6, 10–13, 15.
- [CBC*17] CISSE, MOUSTAPHA, BOJANOWSKI, PIOTR, GRAVE, EDOUARD, et al. "Parseval Networks: Improving Robustness to Adversarial Examples". *Proceedings of the 34th International Conference on Machine Learning*. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, 854–863 3.
- [CD23] CLÉMOT, MATTÉO and DIGNE, JULIE. "Neural skeleton: Implicit neural representation away from the surface". *Computers & Graphics* 114 (2023), 368–378 3.
- [CP*20] CHIBANE, JULIAN, PONS-MOLL, GERARD, et al. "Neural unsigned distance fields for implicit function learning". *Advances in Neural Information Processing Systems* 33 (2020), 21638–21652 3.
- [CZ19] CHEN, ZHIQIN and ZHANG, HAO. "Learning implicit fields for generative shape modeling". *Proc. IEEE/CVF conference on CVPR*. 2019, 5939–5948 3.
- [DLX*22] DOU, ZHIYANG, LIN, CHENG, XU, RUI, et al. "Coverage Axis: Inner Point Selection for 3D Shape Skeletonization". *Computer Graphics Forum*. Vol. 41. 2. 2022, 419–432 5.
- [DNJ20] DAVIES, THOMAS, NOWROUZEZAHRAI, DEREK, and JACOBSON, ALEC. "On the effectiveness of weight-encoded neural implicit 3d shapes". *arXiv:2009.09808* (2020) 3, 13.
- [Ede01] EDELSBRUNNER, HERBERT. *Geometry and Topology for Mesh Generation*. Cambridge University Press, 2001 5.
- [GYH*20] GROPP, AMOS, YARIV, LIOR, HAIM, NIV, et al. "Implicit Geometric Regularization for Learning Shapes". *Proceedings of the 37th International Conference on Machine Learning*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, 3789–3799 3.
- [LC24] LUDWIG, INGMAR and CAMPEN, MARCEL. "Strictly Conservative Neural Implicits". *Computer Graphics Forum*. Vol. 43. 7. 2024, e15241 2, 3, 6, 7, 9, 10, 13–15.
- [LFYR24] LIU, STEPHANIE WENXIN, FISCHER, MICHAEL, YOO, PAUL D, and RITSCHER, TOBIAS. "Neural bounding". *ACM SIGGRAPH 2024 Conference Papers*. 2024, 1–10 2, 3, 6, 14.
- [LSK18] LU, LU, SU, YANHUI, and KARNIAKAKIS, GEORGE EM. "Collapse of Deep and Narrow Neural Nets". *arXiv:1808.04947* (2018) 9.
- [LWJ*22] LIU, HSUEH-TI DEREK, WILLIAMS, FRANCIS, JACOBSON, ALEC, et al. "Learning smooth neural functions via Lipschitz regularization". *ACM SIGGRAPH 2022 conference proceedings*. 2022, 1–13 3.
- [MON*19] MESCHER, LARS, OECHSLE, MICHAEL, NIEMEYER, MICHAEL, et al. "Occupancy networks: Learning 3d reconstruction in function space". *Proc. CVPR*. 2019, 4460–4470 3.
- [MPJ*19] MICHALKIEWICZ, MATEUSZ, PONTES, JHONY K, JACK, DOMINIC, et al. "Deep level sets: Implicit surface representations for 3d shape inference". *arXiv:1901.06802* (2019) 3.
- [MSLJ23] MARSCHNER, ZOË, SELLÁN, SILVIA, LIU, HSUEH-TI DEREK, and JACOBSON, ALEC. "Constructive solid geometry on neural signed distance fields". *SIGGRAPH Asia 2023 conference papers*. 2023, 1–12 1–3.
- [PFS*19] PARK, JEONG JOON, FLORENCE, PETER, STRAUB, JULIAN, et al. "DeepSDF: Learning continuous signed distance functions for shape representation". *Proc. of the IEEE/CVF conference on CVPR*. 2019, 165–174 3.
- [PK08] PAVIĆ, DARKO and KOBELT, LEIF. "High-resolution volumetric computation of offset surfaces with feature preservation". *Computer Graphics Forum*. Vol. 27. 2. 2008, 165–174 3.
- [PL22] PRACH, BERND and LAMPERT, CHRISTOPH H. "Almost-orthogonal layers for efficient general-purpose Lipschitz networks". *European Conference on Computer Vision*. Springer. 2022, 350–365 3.
- [PM09] PATRIKALAKIS, NICHOLAS M. and MAEKAWA, TAKASHI. *Shape Interrogation for Computer Aided Design and Manufacturing*. Springer Publishing Company, 2009 5.
- [SG05] SI, HANG and GÄRTNER, KLAUS. "Meshing Piecewise Linear Complexes by Constrained Delaunay Tetrahedralizations". *Proc. 14th International Meshing Roundtable*. Springer, 2005, 147–163 5.
- [Si08] SI, HANG. "Adaptive tetrahedral mesh generation by constrained Delaunay refinement". *International Journal for Numerical Methods in Engineering* 75.7 (2008), 856–880 5.
- [Si15] SI, HANG. "TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator". *ACM Trans. Math. Softw.* 41.2 (2015), 11:1–11:36 7.
- [SK01] STOLTE, NILO and KAUFMAN, ARIE. "Novel techniques for robust voxelization and visualization of implicit surfaces". *Graphical Models* 63.6 (2001), 387–412 5.
- [SMG*21] SERRURIER, MATHIEU, MAMALET, FRANCK, GONZÁLEZ-SANZ, ALBERTO, et al. "Achieving robustness in classification using optimal transport with hinge regularization". *Proc. IEEE/CVF Conference on CVPR*. 2021, 505–514 3, 6.
- [STL*25] SCHOTT, HUGO, THONAT, THEO, LAMBERT, THIBAUD, et al. "Sphere carving: Bounding volumes for signed distance fields". *ACM Transactions on Graphics* 44.4 (2025), 1–13 3.
- [SVJ15] SACHT, LEONARDO, VOUGA, ETIENNE, and JACOBSON, ALEC. "Nested cages". *ACM Transactions on Graphics* 34.6 (2015), 1–14 3.
- [TK21] TROCKMAN, ASHER and KOLTER, J ZICO. "Orthogonalizing convolutional layers with the Cayley transform". *International Conference on Learning Representations*. 2021 3.
- [vDHS15] VON DZIEGIELEWSKI, ANDREAS, HEMMER, MICHAEL, and SCHÖMER, ELMAR. "High Precision Conservative Surface Mesh Generation for Swept Volumes". *IEEE Transactions on Automation Science and Engineering* 12.1 (2015), 183–191 3.
- [XTS*22] XIE, YIHENG, TAKIKAWA, TOWAKI, SAITO, SHUNSUKE, et al. "Neural fields in visual computing and beyond". *Computer graphics forum*. Vol. 41. 2. 2022, 641–676 3.
- [YBHK21] YANG, GUANDAO, BELONGIE, SERGE, HARIHARAN, BHARATH, and KOLTUN, VLADLEN. "Geometry processing with neural fields". *Advances in Neural Information Processing Systems* 34 (2021), 22483–22497 3.
- [YM17] YOSHIDA, YUICHI and MIYATO, TAKERU. "Spectral norm regularization for improving the generalizability of deep learning". *arXiv:1705.10941* (2017) 3.
- [ZCEP07] ZHANG, LONG, CHEN, WEI, EBERT, DAVID S., and PENG, QUNSHENG. "Conservative voxelization". *The Visual Computer* 23.9 (2007), 783–792 3, 5.