Representing 3D Shapes with Probabilistic Directed Distance Fields

Tristan Aumentado-Armstrong[1,2,3] Stavros Tsogkas[1] Sven Dickinson[1,2,3] Allan Jepson[1,2]
[1]Samsung AI Centre Toronto [2]University of Toronto [3]Vector Institute for AI
tristan.a@partner.samsung.com, {stavros.t,s.dickinson,allan.jepson}@samsung.com

Abstract

Differentiable rendering is an essential operation in modern vision, allowing inverse graphics approaches to 3D understanding to be utilized in modern machine learning frameworks. Explicit shape representations (voxels, point clouds, or meshes), while relatively easily rendered, often suffer from limited geometric fidelity or topological constraints. On the other hand, implicit representations (occupancy, distance, or radiance fields) preserve greater fidelity, but suffer from complex or inefficient rendering processes, limiting scalability. In this work, we endeavour to address both shortcomings with a novel shape representation that allows fast differentiable rendering within an implicit architecture. Building on implicit distance representations, we define Directed Distance Fields (DDFs)*, which map an oriented point (position and direction) to surface visibility and depth. Such a field can render a depth map with a single forward pass per pixel, enable differential surface geometry extraction (e.g., surface normals and curvatures) via network derivatives, be easily composed, and permit extraction of classical unsigned distance fields. Using probabilistic DDFs (PDDFs), we show how to model inherent discontinuities in the underlying field. Finally, we apply our method to fitting single shapes, unpaired 3D-aware generative image modelling, and single-image 3D reconstruction tasks, showcasing strong performance with simple architectural components via the versatility of our representation.*

1. Introduction

Three-dimensional shapes are represented in a variety of ways in modern computer vision and machine learning systems, with differing utilities depending on the task to which they are applied. Recent advances in representation learning, however, capitalize on the inherent 3D structure of the world, and its link to generating the 2D images seen by our eyes and algorithms, via *differentiable rendering* procedures compatible with neural network architectures. This enables an analysis-by-synthesis paradigm [90] that treats vision as "inverse graphics" [35, 63], wherein the model attempts to infer the 3D factors (e.g., shape, pose, texture, lighting) that gave rise to its 2D perceptions.

This can permit learning more powerful representations with weaker supervision. Neural radiance fields (NeRFs) [47], for instance, can be used for 3D inference [89] and 3D-aware generative image modelling [4, 51, 67], trained entirely on 2D data. Similarly, implicit geometric fields, such as occupancy fields [46] and signed distance fields (SDFs) [55], have recently been used in conjunction with differentiable renderering as well [25, 41, 52]. Other works have learned textured mesh inference and/or generation via rendering-based approaches (e.g., [1, 13, 19, 57, 76]).

Nevertheless, it is still not always clear which representation is best for a given task. Voxels and point clouds tend to have reduced geometric fidelity, while meshes suffer from the difficulties inherent in discrete structure generation, often leading to topological and textural fidelity constraints, or dependence of rendering efficiency on shape complexity [39]. While implicit shapes can have superior fidelity, they struggle with complex or inefficient rendering procedures, requiring multiple network forward passes and/or complex calculations per pixel [41, 47, 72], and may be difficult to use for certain tasks (e.g., deformation, segmentation, or correspondence). Thus, a natural question is how to design a method capable of fast differentiable rendering, yet still retaining high-fidelity geometric information that is useful for a variety of downstream applications.

In this work, we explore *directed distance fields* (DDFs), a representation that (i) captures the detailed geometry of a scene or object, including higher-order differential quantities and internal geometry, (ii) can be differentiably rendered efficiently, compared to common implicit shape or radiance-based approaches, (iii) is trainable with (pointwise) depth data, (iv) can be easily composed, and (v) allows extraction of classical unsigned distance fields. The definition is simple: for a given shape, we learn a field that maps any position and orientation to *visibility* (i.e., whether the surface exists from that position along that direction) and *distance* (i.e., how far the surface is along that ray, if it is visible). Fig. 1 illustrates how DDFs can be viewed as implicitly storing all possible depth images of a given

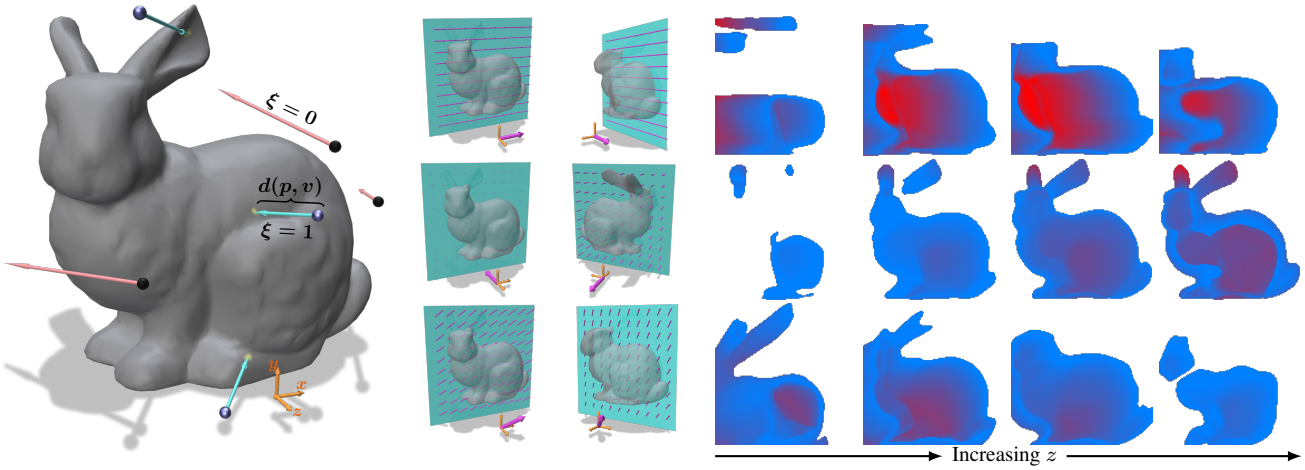


Figure 1. Illustrative example of a directed distance field, fit to the Stanford bunny. *Left*: depiction of visible oriented points (blue points, turquoise directions) that intersect the shape and those that miss the shape (black points, red directions) with $\xi = 0$. *Middle*: per row, illustrations of one slice plane (from two different views) and the fixed v vector per slice plane (pink arrows), corresponding to the insets on the right (i.e., v is the same across all p for each row). *Right*: resulting depth field evaluated across positions p at fixed orientations v (rows: top, middle, and bottom show different v values, parallel to $(1, 0, 0)$, $(0, 0, -1)$, and $(1, 1, 1)$, respectively; columns: different slices in 3D with each having fixed z, effectively sliding the turquoise plane from the middle inset in z). Each pixel value is coloured with the distance value $d(p, v)$ obtained for that position p and direction v (red to blue meaning further to closer). Non-visible oriented points ($\xi = 0$) are shown as white. Notice the depth changes at intersections between the slice plane and shape (i.e., when p moves through S).

shape (i.e., from all possible cameras), reminiscent of a light field, but with geometric distance instead of radiance (see Fig. 1). Such a field is inherently discontinuous (see Fig. 2), presenting issues for differentiable neural networks, but has a powerful advantage in rendering, since a depth image can be computed with a single forward pass per pixel. Its high input dimension (5D) incurs greater difficulty in learning, but the additional information increases its versatility (e.g., higher-order local geometry, internal structure); furthermore, several geometric properties define constraints on the field and its derivatives, reducing the effective degrees of freedom. We summarize our contributions as follows:

1. We define directed distance fields (DDFs), a 5D mapping from any position and viewpoint to depth and visibility (§3), and a probabilistic variant (PDDFs) that can model surface and occlusion discontinuities (§3.3).

2. By construction, our representation allows differentiable rendering via a single forward pass per pixel (§3.2), without restrictions on the shape (topology, water-tightness) or field queries (internal structure).

3. We prove several geometric properties of DDFs (§3.1), and use them in our method.

4. We apply DDFs to fitting shapes (§4.1), single-image reconstruction (§4.3), and generative modelling (§4.4).

2. Related Work

Implicit Shape Representations Our work falls under distance field representations of shape, which have a long history in computer vision [64], recently culminating in signed/unsigned distance fields (S/UDFs) [6, 55, 80] and related methods [3, 34, 79]. Compared to explicit ones, implicit shapes can capture arbitrary topologies with high fidelity [40, 46, 55]. Several works examine differentiable rendering of implicit fields [25, 40, 41, 52, 72, 74, 91] (or combine it with neural volume rendering [29, 54, 83, 87]). In contrast, by conditioning on both viewpoint and position, DDFs can flexibly render depth, with a single field query per pixel. Further, a UDF can actually be extracted from a DDF (see §3.1 and §4.2).

The closest current model to ours is the Signed Directional Distance Field (SDDF), independently and concurrently developed by Zobeidi et al. [95], which also maps position and direction to depth. However, the lack of a sign in DDFs introduces a fundamental difference in structure modelling: starting from a point p, consider a ray that intersects with a wall; evaluating a DDF at a point after the intersection provides the distance to the next object, while the SDDF continues to measure the signed distance to the wall. This reduces complexity and dimensionality, but may limit representational utility for some tasks and/or shapes.

Neural Radiance Fields NeRFs [47] are powerful 3D representations, capable of novel view synthesis for reconstruction [89] and image generation [4] with very high fi-

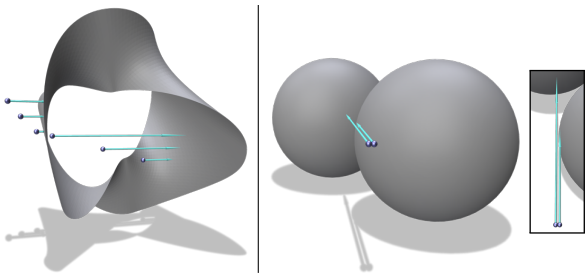


Figure 2. Inherent discontinuities with DDFs. Left: *surface discontinuities*, where p passes through S. Right: *occlusion discontinuities*, where v or p is moved over an occluding boundary edge.

delity. However, the standard differentiable volume rendering formulation of NeRFs is computationally expensive, requiring many forward passes per pixel, though recent work has improved on this (e.g., [2,11,18,29,38,60,61,88]). Furthermore, the distributed nature of the density makes extracting explicit geometric details (including higher-order surface information) more difficult (e.g., [54, 87]).

Most similar to DDFs are Light Field Networks (LFNs) [71], which enable rendering with a single forward pass per pixel, and permit sparse depth map extraction (assuming a Lambertian scene). Unlike LFNs, DDFs model geometry rather than radiance as the primary quantity, computing depth with a single forward pass, and surface normals with a single backward pass, while LFNs predict RGB and sparse depth from such a forward-backward operation. Finally, the 4D parameterization of LFNs does not permit rendering from viewpoints between occluded objects.

3. Directed Distance Fields

Definition Let $S \subset \mathcal{B}$ be a 3D shape, where $\mathcal{B} \subset \mathbb{R}^3$ is a bounding volume that will act as the domain of the field. Consider a position $p \in \mathcal{B}$ and view direction $v \in \mathbb{S}^2$. We define S to be *visible* from an oriented point (p, v) if the line $\ell_{p,v}(t) = p + tv$ intersects S for some $t \geq 0$. We write the binary visibility field for S as $\xi(p, v) = \mathbb{1}[(p, v) \text{ is visible}]$. For convenience, we refer to an oriented point (p, v) as visible if $\xi(p, v) = 1$.

We then define a *directed distance field* (DDF) as a non-negative scalar field $d : \mathcal{B} \times \mathbb{S}^2 \to \mathbb{R}_+$, which maps from any visible position and orientation in space to the minimum distance from p to S along v (i.e., the first intersection of $\ell_{p,v}(t)$ with S). In other words, $q(p, v) = d(p, v)v + p$ is a map to the shape, and thus satisfies $q(p, v) \in S$ for visible (p, v) (meaning $\xi(p, v) = 1$). See Fig. 1 for an illustration.

3.1. Geometric Properties

DDFs satisfy several useful geometric properties. We provide proofs in Appendix A.

Property I: Directed Eikonal Equation. Similar to SDFs, which satisfy the eikonal equation $||\nabla_p \text{SDF}(p)||_2 = 1$, a DDF enforces a directed version of this property. In particular, for any visible (p, v), we have $\nabla_p d(p, v)v = -1$, with $\nabla_p d(p, v) \in \mathbb{R}^{1\times 3}$. Note this implies $||\nabla_p d(p, v)||_2 \geq 1$ as well. There is also a directed eikonal property for the visibility field, as locally moving along the viewing line cannot change visibility: $\nabla_p \xi(p, v)v = 0$.

Property II: Surface Normals. The derivatives of implicit fields are closely related to the surface normals $n \in \mathbb{S}^2$ of S; e.g., $\nabla_q \text{SDF}(q)^T = n(q)$ for any $q \in S$. For DDFs, a similar relation holds (*without* requiring $p \in S$): $\nabla_p d(p, v) = -n(p, v)^T/(n(p, v)^T v)$, for any visible (p, v) such that $n(p, v) := n(q(p, v))$ are the normals at $q(p, v) = d(p, v)v+p \in S$ and $n(p, v) \not\perp v$ (i.e., the change in d moving *off* the surface is undefined). This allows recovering the surface normals of any point $q \in S$, simply by querying any (p, v) on the line that "looks at" q, and computing $n(p, v) = \varsigma \nabla_p d(p, v)^T/||\nabla_p d(p, v)||_2$, where we choose $\varsigma \in \{-1, 1\}$ such that $n^T v < 0$ (so that n always points back to the query oriented point)[1]. In this sense, $n(p, v)$ is the visible surface normal on S, as seen from (p, v).

Property III: Gradient Consistency. Consider a visible (p, v). Notice that changing the viewpoint by some infinitesimal δ_v would seem to have a similar effect as pushing the position p in the direction δ_v. In fact, it can be shown that $\nabla_v d(p, v)\delta_v = d(p, v)\nabla_p d(p, v)\delta_v$, where $\delta_v = \omega \times v$ for any $\omega \in \mathbb{R}^3$. This relates the directional derivatives of d, along a rotational perturbation δ_v, with respect to both viewpoint and position (see also Appendix A.3 for alternative expressions). As with Property I, any d must satisfy gradient consistency to be a true DDF.

Property IV: Deriving Unsigned Distance Fields. We remark that an unsigned distance field (UDF) can be extracted from a DDF via the following optimization problem: $\text{UDF}(p) = \min_{v\in\mathbb{S}^2} d(p, v)$, constrained such that $\xi(p, v) = 1$, allowing them to be procured if needed (see §4.2). UDFs remove the discontinuities from DDFs (see §3.3 and Fig. 2), but are not rendered as easily nor can they be queried for distances in arbitrary directions.

Property V: Local Differential Geometry. For any visible (p, v), the geometry of a 2D manifold S near $q(p, v)$ is completely characterized by $d(p, v)$ and its derivatives. In particular, we can estimate the first and second fundamental forms using the gradient and Hessian of $d(p, v)$ (see Appendix A.4). This allows computing surface properties, such as curvatures, from any visible oriented position, simply by querying the network; see Fig. 5 for an example.

Neural Geometry Rendering. Many methods utilize differentiable rendering of geometric quantities, such as depth and surface normals (e.g., [50, 77, 84, 85]). Often, such methods can be written as parallelized DDFs (see Appendix A.6). Thus, the properties above hold, regardless of

[1] This defines the normal via v, even for non-orientable surfaces.

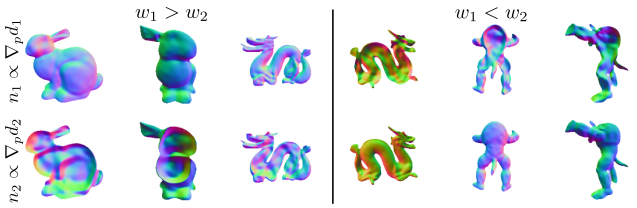


Figure 3. PDDF renders of n_1 and n_2. Though not explicitly enforced, a "see-through effect" occurs when the lower-weight field models the surface behind the currently visible one.

architecture; we believe this can improve such frameworks.

3.2. Rendering

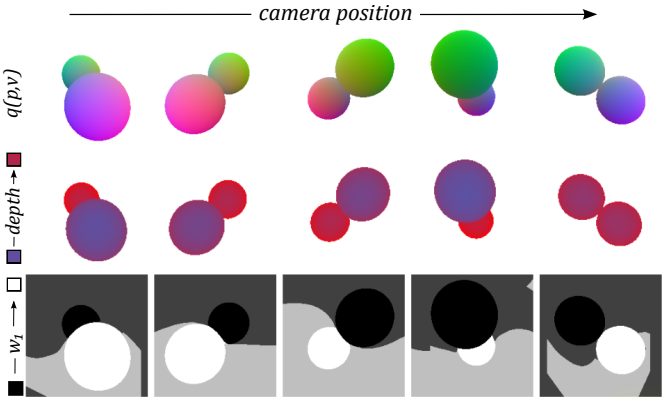
A primary application of DDFs is rapid differentiable rendering. In contrast to some differentiable mesh renderers (e.g., [39]), there is no dependence on the complexity of the underlying shape, after training. Unlike classical NeRFs [47] or other standard implicit shape fields [41, 72], DDFs only require a single forward pass per pixel.

The process itself is a straightforward ray casting procedure. Given a camera with position $p_0 \in \mathcal{B}$, for a pixel with 3D position ρ, we effectively cast a ray $r(t) = p_0 + tv_\rho$ with $v_\rho = (\rho - p_0)/||\rho - p_0||_2$ into the scene with a single query $d(p_0, v_\rho)$, which provides the depth image pixel value. Note that ρ, and thus $d(p_0, v_\rho)$, depend on the camera parameters. Finally, consider $p \notin \mathcal{B}$. In this case, we first compute the intersection point $p_r \in \partial\mathcal{B}$ between the ray r and the boundary $\partial\mathcal{B}$. We then use $d(p_r, v) + ||p - p_r||_2$ as the output depth (or set $\xi(p_r, v) = 0$ if no intersection exists). This allows querying the network from arbitrary positions and directions, including those unseen in training.

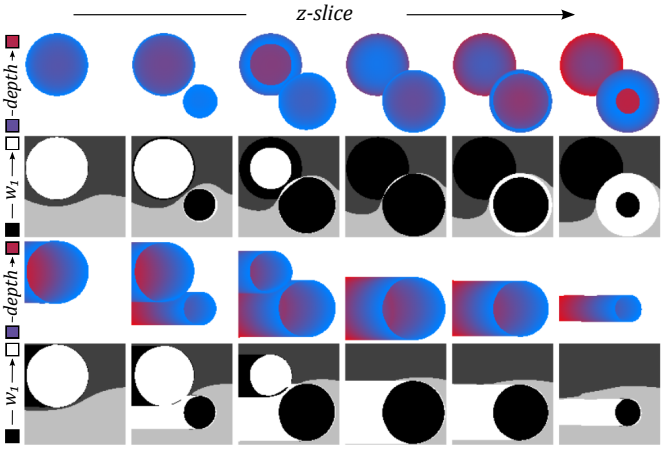
3.3. Discontinuity Handling: Probabilistic DDFs

DDFs are inherently discontinuous functions of p and v. As shown in Fig. 2, whenever (i) p passes through the surface S or (ii) p or v is moved across an occlusion boundary, a discontinuity in $d(p, v)$ will occur. We therefore modify the DDF formulation, to allow a C^1 network to represent the discontinuous field. In particular, we alter d to output probability distributions over rays, rather than a single value. Let $\mathcal{P}_\ell$ be the set of probability distributions with support on some ray $\ell_{p,v}(t) = p + tv$, $t \geq 0$. Then $d : \mathcal{B} \times \mathbb{S}^2 \to \mathcal{P}_\ell$ is a *probabilistic DDF* (PDDF). The visibility field, $\xi(p, v)$, is unchanged in the PDDF.

For simplicity, herein we restrict $\mathcal{P}_\ell$ to be the set of mixtures of Dirac delta functions with K components. Thus, the network output is a density field $P_{p,v}(d) = \sum_i w_i \delta(d - d_i)$ over depths, where w_i's are the mixture weights, with $\sum_i w_i = 1$, and d_i's are the delta locations. Our output depth is then d_{i^*}, where $i^* = \text{argmax}_i\, w_i$; i.e., the highest weight delta function marks the final output location.



(a) Weight field transitions in DDF renders. In row three, white vs black mark high vs low w_1 values, and thus which surface (d_1 vs d_2) is active, when ξ is high. Light and dark grey demarcate the non-visible (low ξ) counterparts of white and black. The change in dominant weight (w_1 vs $1 - w_1$) at occlusion edges permits discontinuities.



(b) Weight field transitions using 3D slices in z. Rows 1 and 3 depict (discontinuous) distance values, with fixed v ($(0, 0, -1)$ and $(1, 0, 0)$, respectively) and varying p across image pixels. Rows 2 and 4 show weight values for w_1 and ξ, as in (a) above. Notice the field switching upon p transitioning through a surface discontinuity.

Figure 4. Illustration of probabilistic DDFs for discontinuous depth modelling, on a simple two-sphere scene with renders (a) and spatial slices (b). Here, $K = 2$, so $w_1 = 1 - w_2$ (see §3.3).

As w_i changes continuously, w_{i^*} will switch from one d_i to another d_j, which may be arbitrarily far apart, resulting in a discontinuous jump. Thus, by having the weight field $w(p, v)$ smoothly *transition* from one index i^* to another, at the site of a surface or occlusion discontinuity, we can obtain a discontinuity in d as desired. In this work, we use $K = 2$, to represent discontinuities without sacrificing efficiency. Fig. 4 showcases example transitions, with respect to (a) occlusion discontinuities and (b) surface collision; Fig. 3 visualizes the difference in the normals fields. Notationally, we may treat a PDDF as a DDF, by setting $d(p, v) := d_{i^*}(p, v)$.

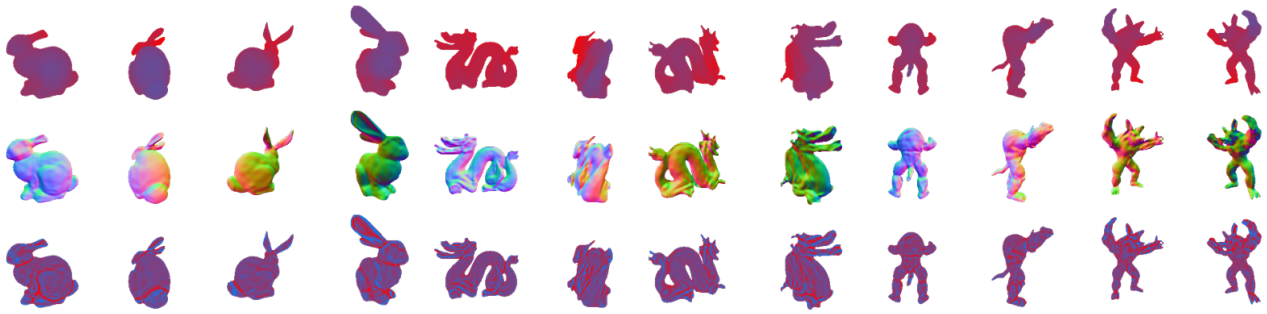


Figure 5. Renders of DDF fits to shapes. Rows: depth, normals, and mean curvature. Columns: different camera positions per object. Each quantity is directly computed from the learned field, using network derivatives at the query oriented point (p, v) per pixel.

3.4. Learning DDFs

Mesh Data Extraction Given a mesh specifying S, we can obtain visibility ξ and depth d by ray-casting from any (p, v). In total, we consider six types of data samples (visualized in Appendix B): *uniform* (U) random (p, v); *at-surface* (A), where $\xi(p, v) = 1$; *bounding* (B), where $p \in \partial\mathcal{B}$ and v points to the interior of $\mathcal{B}$; *surface* (S), where $p \in S$ and $v \sim \mathcal{U}[\mathbb{S}^2]$; *tangent* (T), where v is in the tangent space of $q(p, v) \in S$; and *offset* (O), which offsets p from T-samples along $n(p, v)$ by a small value. See Appendix C.2 for a data sample type ablation study.

Loss Functions Our optimization objectives are defined per oriented point (p, v). We denote ξ, n, and d as the ground truth visibility, surface normal, and depth values, and let $\widehat{\xi}$, $\widehat{d}_i$, and w_i denote the network predictions. Recall $i^* = \arg\max_j w_j$ is the maximum likelihood PDDF index.

The *minimum distance loss* ensures that the correct depth is output for the highest probability component: $\mathcal{L}_d = \xi|\widehat{d}_{i^*} - d|^2$.The *visibility objective*, $L_\xi = \mathrm{BCE}(\xi, \widehat{\xi})$, is the binary cross entropy between the visibility prediction and the ground truth. A first-order *normals loss* (as in [12]), $\mathcal{L}_n = -\xi|n^T\widehat{n}_{i^*}(p, v)|$, uses Property II to match surface normals to the underlying shape, via $\nabla_p\widehat{d}_{i^*}$. A *Directed Eikonal regularization*, based on Property I, is given by

$$\mathcal{L}_{\mathrm{DE}} = \gamma_{\mathrm{E},d}\sum_i \xi\left[\nabla_p\widehat{d}_i v + 1\right]^2 + \gamma_{\mathrm{E},\xi}[\nabla_p\widehat{\xi} v]^2, \quad (1)$$

applied on the visibility and each delta component of d, analogous to prior SDF work (e.g., [15, 37, 86]).

Finally, we utilize two weight field regularizations, which encourage (1) low entropy PDDF outputs (to prevent i^* from switching unnecessarily), and (2) the maximum likelihood delta component to *transition* (i.e., change i^*) when a discontinuity is required: $\mathcal{L}_W = \gamma_V\mathcal{L}_V + \gamma_T\mathcal{L}_T$. The first is a *weight variance loss*: $\mathcal{L}_V = \prod_i w_i$. The second is a *weight transition loss*: $\mathcal{L}_T = \max(0, \varepsilon_T - |\nabla_p w_1 n|)^2$, where ε_T is a hyper-parameter controlling the desired transition speed. Since $K = 2$, using w_1 alone is sufficient to enforce changes along the normal. Note that $\mathcal{L}_T$ is *only* applied to oriented points that we wish to undergo a transition (i.e., where a discontinuity is desired, as illustrated in Fig. 2 and 4), namely surface (S) and tangent (T) data. The complete PDDF shape-fitting loss is then

$$\mathcal{L}_S = \gamma_d\mathcal{L}_d + \gamma_\xi\mathcal{L}_\xi + \gamma_n\mathcal{L}_n + \mathcal{L}_{\mathrm{DE}} + \mathcal{L}_W. \quad (2)$$

Other regularizations could be applied (e.g., gradient and view consistency; see Property III and Appendix A.5), but for simplicity we leave them to future work.

4. Empirical Results

4.1. Single Field Fitting

We use the SIREN neural architecture [70] for all field parameterizations, as it allows for higher order derivative calculations and has shown powerful representational capabilities (e.g., [4, 26]). We use $K = 2$, an axis-aligned bounding box for $\mathcal{B}$, Adam [30] for optimization, and PyTorch [56] for all implementations. (See Appendix C for details.) In Fig. 5, we show results for fitting single objects, via PDDF renderings with a single network evaluation per pixel. Surface normals and curvatures are obtained using only additional backward passes for the same oriented points used in the single forward pass. Note that our simple architecture does not guarantee view consistency (see Appendix A.5) by construction.

We discuss two additional modelling capabilities of DDFs: (i) internal structure representation and (ii) compositionality. The first refers to the ability of our model to handle multi-layer surfaces: we are able to place a camera inside a scene, within or between multiple surfaces, along a given direction. This places our representation in contrast with recent work [71, 95], which does not model internal structure. The second lies in the ease with which we can combine multiple DDFs, which is useful for manipulation without retraining and scaling to more complex scenes. Our approach is inspired by prior work on soft rendering [10, 39]. Formally, given a set of N DDFs

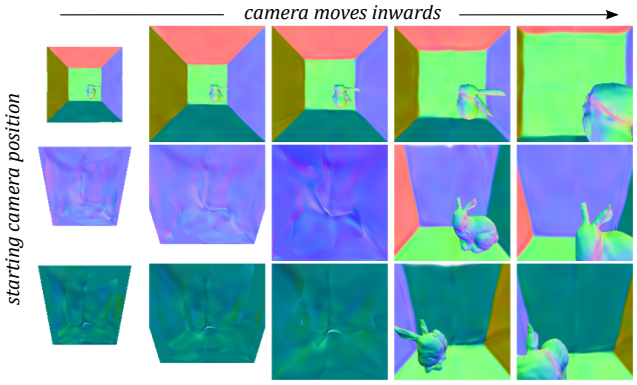


Figure 6. Example of internal structure rendering and compositional scene construction. Colours correspond to surface normals (as in Fig. 5), estimated via the DDF (with Property II).

$\zeta = \{T^{(i)}, \xi^{(i)}, d^{(i)}, \mathcal{B}^{(i)}\}_{i=1}^{N}$, where $T^{(i)}$ is a transform on oriented points converting world to object coordinates for the ith DDF (e.g., scale, rotation, and translation), we can aggregate the visibility and depth fields into a single combined DDF. For visibility of the combination of objects, we ask that *at least* one surface is visible, implemented as:

$$\xi_\zeta(p, v) = 1 - \prod_k (1 - \xi^{(k)}(T^{(k)}(p, v))). \tag{3}$$

For depth, we want the closest visible surface to be the final output. One way to perform this is via a linear combination

$$d_\zeta(p, v) = \sum_i a_\zeta^{(k)}(p, v)\, d^{(k)}(T^{(k)}(p, v)), \tag{4}$$

where $a_\zeta^{(k)}$ are computed via visibility and distance:

$$a_\zeta(p, v) = \text{Softmax}\left(\left\{\frac{\eta_T^{-1}\xi^{(k)}(T^{(k)}(p, v))}{\varepsilon_s + d^{(k)}(T^{(k)}(p, v))}\right\}_k\right), \tag{5}$$

with temperature η_T and maximum inverse depth scale ε_s as hyper-parameters. This upweights contributions when distance is small, but visibility is high. We exhibit these capabilities in Fig. 6, which consists of two independently trained DDFs (one fit to five planes, forming a simple room, and the other to the bunny mesh), where we simulate a camera starting outside the scene and entering the room.

4.2. UDF Extraction

As noted in Property IV, one can extract a UDF from a DDF. In particular, we optimize a field $v^* : \mathcal{B} \to \mathbb{S}^2$, such that $\text{UDF}(p) = d(p, v^*(p))$. We solve this by gradient descent on a loss that maximizes visibility while minimizing depth for a given $v^*(p)$. Compared to directly fitting a UDF, this requires handling local minima for v^* and non-visible (low ξ) directions. (See Fig. 7 for visualizations and Appendix D for optimization details.)

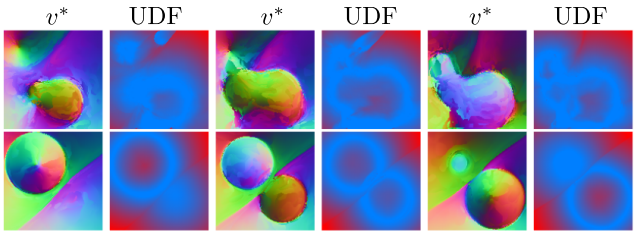


Figure 7. v^* fields (colours are 3D components, pointing to the closest visible surface) and respective UDFs (blue to red means near to far distances). Each image is a slice in z, with adjacent pairs having the same z. The differing colours in v^* for the multi-sphere in column 3 are due to the slice breaching the front versus the back of the two spheres.

The vector field v^* points in the direction of the closest point on S. Notice that discontinuities in v^* occur at surfaces as before, but also on the medial surface of S in $\mathcal{B}$.[2] When the surface normals exist, v^* is closely related to them: $v^*(p) = -n(p, v^*(p))$, in the notation of Property II. Recent work has highlighted the utility of UDFs over SDFs [6, 80]; in the case of DDFs, extracting a UDF or v^* may provide useful auxiliary information for some tasks.

4.3. Single-Image 3D Reconstruction

We next utilize DDFs for single-image 3D reconstruction. Given a colour image I, we predict the underlying latent shape z_s and camera Π that gave rise to the image, via an encoder $E(I) = (\widehat{z}_s, \widehat{\Pi})$. For decoding, we use a *conditional* PDDF (CPDDF), which computes depth $\widehat{d}(p, v|z_s)$ and visibility $\widehat{\xi}(p, v|z_s)$. For evaluation, we use the extrinsics of a camera: either the predicted $\widehat{\Pi}$ or ground-truth Π_g (to separate shape and camera errors).

We use three loss terms: (a) shape DDF fitting in canonical pose $\mathfrak{L}_S$ (eq. 2), (b) camera prediction $\mathcal{L}_\Pi = ||\Pi_g - \widehat{\Pi}||_2^2$, and (c) mask matching $\mathcal{L}_M = \text{BCE}(I_\alpha, \mathcal{R}_\xi(z_s|\widehat{\Pi}))$, where I_α is the input alpha channel and $\mathcal{R}_\xi$ renders the DDF visibility. The total objective, $\mathcal{L}_{\text{SI3DR}} = \gamma_{R,S}\mathfrak{L}_S + \gamma_{R,\Pi}\mathcal{L}_\Pi + \gamma_{R,M}\mathcal{L}_M$, is optimized by AdamW [42]. We implement E as two ResNet-18 networks [17], while the CPDDF is a modulated SIREN [44]. See Appendix E for details.

Explicit Sampling. Evaluating 3D reconstruction often involves metrics based on point clouds (PCs). We present a simple approach to PC extraction from DDFs, though it cannot guarantee uniform sampling over the shape. Analogous to prior work [6], we recall that $q(p, v) = p + d(p, v)v \in S$, if $\xi(p, v) = 1$. Thus, we sample $p \sim \mathcal{U}[\mathcal{B}]$, and wish to compute their projections q onto the shape. However, we cannot choose an arbitrary v, as many will not be visible. Instead, for each p, we uniformly sample directions $V(p) = \{v_i(p) \sim \mathcal{U}[\mathbb{S}^2]\}_{i=1}^{n_v}$ and utilize our composition

[2] At such positions, there are multiple valid values of v^*.

		DDF					PC-SIREN				P2M	3DR
		Π_g-L	Π_g-S	$\widehat{\Pi}_\nabla$-S	$\widehat{\Pi}$-L	$\widehat{\Pi}$-S	Π_g-L	Π_g-S	$\widehat{\Pi}$-L	$\widehat{\Pi}$-S		
Chairs	$D_C \downarrow$	0.459	0.512	0.823	0.855	0.919	0.431	0.465	0.876	0.915	0.610	1.432
	$F_\tau \uparrow$	55.47	48.40	42.28	47.51	41.08	62.25	56.36	50.56	45.57	54.38	40.22
	$F_{2\tau} \uparrow$	72.82	67.75	60.39	63.81	58.98	77.38	74.56	65.43	62.76	70.42	55.20
Planes	$D_C \downarrow$	0.210	0.239	0.673	0.793	0.836	0.215	0.227	0.829	0.844	0.477	0.895
	$F_\tau \uparrow$	80.46	76.62	63.32	63.75	60.54	81.49	80.11	63.64	62.34	71.12	41.46
	$F_{2\tau} \uparrow$	90.05	88.55	76.16	74.96	73.47	89.71	89.18	74.76	74.18	81.38	63.23
Cars	$D_C \downarrow$	0.231	0.288	0.390	0.541	0.606	0.371	0.400	0.737	0.768	0.268	0.845
	$F_\tau \uparrow$	70.91	59.93	54.16	62.69	52.47	64.57	57.82	56.01	50.04	67.86	37.80
	$F_{2\tau} \uparrow$	86.57	79.66	74.68	79.71	72.78	78.72	76.00	71.22	68.52	84.15	54.84

Table 1. Single-image 3D reconstruction results. Rows: ShapeNet categories and performance metrics. Columns: L/S refer to sampling 5000/2466 points for evaluation (2466 being the output size of P2M), Π_g/$\widehat{\Pi}$ denote using the true versus predicted camera for evaluation (the former case removing camera prediction error), and $\widehat{\Pi}_\nabla$ test-time camera correction from the predicted position using gradient descent. Metrics: D_C is the Chamfer distance ($\times 1000$), F_τ is the F-score ($\times 100$) at threshold $\tau = 10^{-4}$. PC-SIREN is our matched-architecture baseline; Pixel2Mesh (P2M) [81, 82] and 3D-R2N2 (3DR) [7] are baselines using different shape modalities (numbers from [82]). Note that scenarios using Π_g (effectively evaluating shapes in canonical object coordinates) are not directly comparable to P2M or 3DR. Overall, DDF-derived PCs (1) perform similarly to directly learning to output a PC and (2) underperform P2M overall, but outperform it in terms of shape quality when camera prediction error is excluded.

technique to estimate $\widehat{v}^*(p)$ by weighted average over $V(p)$ (as in §4.2, but without optimization), giving $q(p, \widehat{v}^*(p))$ as a point on the shape. Repeating this process N_H times (starting from $p \leftarrow q$) can also help, if depths are less accurate far from the shape. We set $n_v = 128$ and $N_H = 3$ (see Appendix E.1 for ablation with $N_H = 1$).

Baselines. Our primary baseline is designed to alter the *shape representation*, while keeping the remaining architecture and training setup as similar as possible. We do this by using the same encoders as the DDF and an almost identical network for the decoder (changing only the input and output layer dimensionalities), but altered to output PCs directly (denoted PC-SIREN). In particular, we treat the decoder as an implicit shape mapping $f_b : \mathbb{R}^3 \to \mathbb{R}^3$, which takes $p \sim \mathcal{U}[-1, 1]^3$ as input and directly returns $q = f_b(p) \in S$ as output. Training uses the Chamfer distance D_C to the ground truth in object coordinates and $\mathcal{L}_\Pi$. We also consider two other baselines: the mesh-based Pixel2Mesh (P2M) [81, 82] and voxel-based 3D-R2N2 (3DR) [7].

Results. We consider cars, planes, and chairs from ShapeNet [5], using the data from [7] (as in [81]). In Table 1, we show DDFs perform comparably to the architecture-matched PC-SIREN baseline. See Fig. 8 for visualizations, as well as Appendix Fig. 12. Generally, the inferred DDF shapes correctly reconstruct most inputs, including thin structures like chair legs, and regardless of topology. The most obvious errors are in pose estimation, but the DDF can also sometimes output "blurry" shape parts when it is uncertain (e.g., for shapes far from the majority of training examples). However, results can be improved by correcting $\widehat{\Pi}$ to $\widehat{\Pi}_\nabla = \text{argmin}_\Pi \mathcal{L}_M$, via gradient descent (starting from $\widehat{\Pi}$) on the test image alpha channel. While explicit modalities, like PCs, can be differentiably rendered (e.g., [23, 27, 77]), DDFs can do so by construction, without additional heuristics or learning. Further, note that (i) the DDF sampling procedure is not learned, (ii) our model is not trained with D_C (on which it is evaluated), and (iii) DDFs are a richer representation than PCs, capable of representing higher-order geometry, built-in rendering, and even PC extraction. Thus, for reconstruction, changing from PCs to DDFs can enrich the representation without quality loss.

Compared to the other baselines, DDFs with predicted $\widehat{\Pi}$ underperform P2M, but outperform 3DR. Results with the ground-truth Π_g indicate that much of this error is due to (imperfect) camera prediction, though this case is not directly comparable to P2M or 3DR. With Π_g, the task becomes prediction in object – rather than camera – coordinates. While each frame has benefits and downsides [69, 75], in our case it is useful to separate shape vs. camera error. Our scores with Π_g suggest DDFs can infer shape at similar quality levels to existing work, despite the naive architecture and sampling strategy. We remark that we do not expect DDFs to directly compare to highly tuned, specialized models at the state-of-the-art. Instead, we show that DDFs can achieve good performance (especially for shape alone), even using simple off-the-shelf components (ResNets and SIREN MLPs), without losing versatility.

4.4. Generative Modelling with Unpaired Data

Finally, we apply CPDDFs to 3D-aware generative modelling, using 2D-3D unpaired data (see, e.g., [1, 28, 48, 94]). This takes advantage of 3D model data, yet avoids requiring paired data. We utilize a two-stage approach: (i) a CPDDF-based variational autoencoder (VAE) [21, 31, 62] on 3D

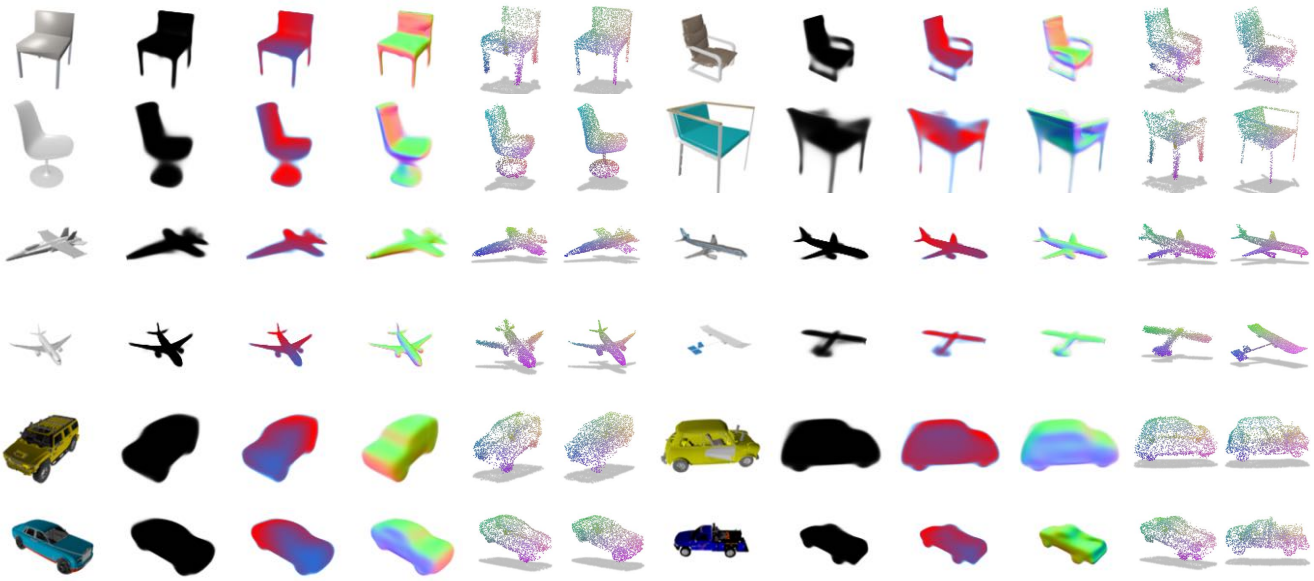


Figure 8. Single-image 3D reconstruction visualizations on held-out test data. Per inset, columns represent (i) the input RGB image, (ii) the visibility $\widehat{\xi}$, (iii) the depth $\widehat{d}$, (iv) the normals $\widehat{n}$, (v) the sampled point cloud (PC) from the DDF, and (vi) a sample from the ground-truth PC. Quantities (ii-v) are all differentiably computed directly from the CPDDF and $\widehat{\Pi}$, per point or pixel (i.e., no post-processing needed). PC colours denote 3D coordinates. A high-error example is in the lower-right of each category. See Appendix E for more examples.

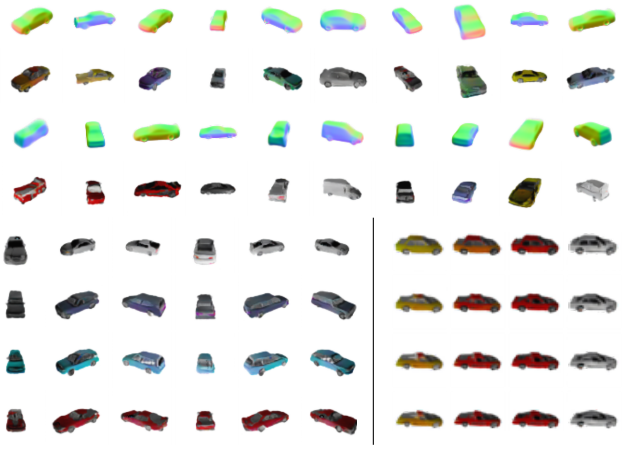


Figure 9. Upper inset: random ShapeVAE and image GAN samples. Left inset: views with fixed texture and shape. Right inset: latent interpolations in shape (vertical) and texture (horizontal).

shapes, then (ii) a generative adversarial network (GAN) [14], which convolutionally translates CPDDF-derived surface normal renders into colour images. Briefly, the VAE trains a PointNet encoder [58] and CPDDF decoder, while the GAN performs cycle-consistent image-to-image translation [92] (from normals to RGB). Fig. 9 displays results on ShapeNet cars [5, 7], including disentanglement of shape, viewpoint, and appearance (see also, e.g., [19, 28, 67]). While this underperforms a 2D image GAN (15 versus 27 FID [20,53]), it still outperforms samples from image VAEs or GAN-based textured low-poly 3D mesh renders (>100 FID [1]) in image quality. See Appendix F for details.

5. Discussion

We have devised *directed distance fields* (DDFs), a novel shape representation, which maps oriented points to depth and visibility values. We have examined several useful theoretical properties (including a probabilistic extension for handling discontinuities), illustrated the fitting process for single objects (as well as composition and UDF extraction) and applied it to single-image reconstruction and generative modelling. DDFs are easily differentiably rendered to a surface normal image, which is non-trivial for voxels, NeRFs, or occupancy fields. Unlike meshes, DDFs are topologically unconstrained, and rendering is independent of shape complexity. In contrast to NeRFs or SDFs, we require just a single forward pass per pixel for depth (plus a single backward pass to obtain normals). One limitation is greater difficulty fitting complex scenes, partly due to the higher input dimensionality; architectural improvements (e.g., multiscale representations [43, 49, 66]) could mitigate this. Our simplistic approaches to geometric property enforcement and UDF/PC extraction can also be improved, which we leave for future work, along with extending the representation to allow translucency, and using a better model of materials, appearance, and lighting. We also hope to apply DDFs to other tasks, such as haptics and navigation.

References

[1] Tristan Aumentado-Armstrong, Alex Levinshtein, Stavros Tsogkas, Konstantinos G Derpanis, and Allan D Jepson. Cycle-consistent generative rendering for 2D-3D modality translation. In *2020 International Conference on 3D Vision (3DV)*, pages 230–240. IEEE, 2020. 1, 7, 8, 22

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. *arXiv preprint arXiv:2103.13415*, 2021. 3

[3] Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. Learning gradient fields for shape generation. In *European Conference on Computer Vision*, pages 364–381. Springer, 2020. 2

[4] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5799–5809, 2021. 1, 2, 5, 21

[5] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015. `https://shapenet.org/`. Terms of use: `https://shapenet.org/terms`. 7, 8, 19, 20, 21

[6] Julian Chibane, Aymen Mir, and Gerard Pons-Moll. Neural unsigned distance fields for implicit function learning. *arXiv preprint arXiv:2010.13938*, 2020. 2, 6

[7] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 7, 8, 19, 22

[8] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 16

[9] Aman Dalmia. dalmia/siren. *`https://doi.org/10.5281/zenodo.3902941`*, June 2020. Zenodo: 10.5281/zenodo.3902941. `https://github.com/dalmia/siren` (MIT License). 16

[10] Jun Gao, Wenzheng Chen, Tommy Xiang, Clement Fuji Tsang, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Learning deformable tetrahedral meshes for 3D reconstruction. *arXiv preprint arXiv:2011.01437*, 2020. 5

[11] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. FastNeRF: High-fidelity neural rendering at 200fps. *arXiv preprint arXiv:2103.10380*, 2021. 3

[12] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9785–9795, 2019. 5

[13] Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoint without keypoints. In *European Conference on Computer Vision*, pages 88–104. Springer, 2020. 1

[14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 8, 21, 22

[15] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099*, 2020. 5

[16] Thibault Groueix. Pytorch chamfer distance. *Thibault-GROUEIX/ChamferDistancePytorch*, Nov. 2021. `https://github.com/ThibaultGROUEIX/ChamferDistancePytorch` (MIT License). 20

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6, 21, 22

[18] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. *arXiv preprint arXiv:2103.14645*, 2021. 3

[19] Paul Henderson, Vagia Tsiminaki, and Christoph H Lampert. Leveraging 2D data to learn textured 3D mesh generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7498–7507, 2020. 1, 8

[20] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. 8

[21] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. Beta-VAE: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations (ICLR)*, 2017. 7, 20

[22] Yerlan Idelbayev. Proper ResNet implementation for CIFAR10/CIFAR100 in PyTorch. `https://github.com/akamaster/pytorch_resnet_cifar10`. Accessed: Nov 2021. (BSD-2-clause License). 21, 22

[23] Eldar Insafutdinov and Alexey Dosovitskiy. Unsupervised learning of shape and pose with differentiable point clouds. *Advances in Neural Information Processing Systems*, pages 2807–2817, 2018. 7

[24] Krishna Murthy Jatavallabhula, Edward Smith, Jean-Francois Lafleche, Clement Fuji Tsang, Artem Rozantsev, Wenzheng Chen, Tommy Xiang, Rev Lebaredian, and Sanja Fidler. Kaolin: A PyTorch library for accelerating 3D deep learning research. *arXiv:1911.05063*, 2019. `https://github.com/NVIDIAGameWorks/kaolin` (Apache License). 19

[25] Yue Jiang, Dantong Ji, Zhizhong Han, and Matthias Zwicker. Sdfdiff: Differentiable rendering of signed distance fields for 3D shape optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1251–1261, 2020. 1, 2

[26] Kyungmin Jo, Gyumin Shim, Sanghun Jung, Soyoung Yang, and Jaegul Choo. Cg-nerf: Conditional generative neural radiance fields. *arXiv preprint arXiv:2112.03517*, 2021. 5

[27] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3D mesh renderer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3907–3916, 2018. 7

[28] Berk Kaya and Radu Timofte. Self-supervised 2d image to 3D shape translation with disentangled representations. In *2020 International Conference on 3D Vision (3DV)*, pages 1039–1048. IEEE, 2020. 7, 8, 22

[29] Petr Kellnhofer, Lars Jebe, Andrew Jones, Ryan Spicer, Kari Pulli, and Gordon Wetzstein. Neural lumigraph rendering. In *CVPR*, 2021. 2, 3

[30] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[31] Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013. 7, 20

[32] Erwin Kreyszig. *Differential Geometry*. University of Toronto Press, 1959. Mathematical Expositions, No. 11. Dover edition. 15

[33] Venkat Krishnamurthy and Marc Levoy. Fitting smooth surfaces to dense polygon meshes. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 313–324, 1996. 16

[34] Nilesh Kulkarni, Justin Johnson, and David F Fouhey. What's behind the couch? directed ray distance functions (drdf) for 3d scene reconstruction. *arXiv e-prints*, pages arXiv–2112, 2021. 2

[35] Tejas D Kulkarni, Will Whitney, Pushmeet Kohli, and Joshua B Tenenbaum. Deep convolutional inverse graphics network. *arXiv preprint arXiv:1503.03167*, 2015. 1

[36] Kwot Sin Lee and Christopher Town. Mimicry: Towards the reproducibility of GAN research. *CVPR Workshop on AI for Content Creation*, 2020. `https://github.com/kwotsin/mimicry` (MIT License). 22

[37] Chen-Hsuan Lin, Chaoyang Wang, and Simon Lucey. SDF-SRN: Learning signed distance 3D object reconstruction from static images. *arXiv preprint arXiv:2010.10505*, 2020. 5

[38] David B Lindell, Julien NP Martel, and Gordon Wetzstein. Autoint: Automatic integration for fast neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14556–14565, 2021. 3

[39] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7708–7717, 2019. 1, 4, 5

[40] Shichen Liu, Shunsuke Saito, Weikai Chen, and Hao Li. Learning to infer implicit surfaces without 3D supervision. *arXiv preprint arXiv:1911.00767*, 2019. 2

[41] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020. 1, 2, 4

[42] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 6, 18

[43] Julien NP Martel, David B Lindell, Connor Z Lin, Eric R Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: Adaptive coordinate networks for neural scene representation. *arXiv preprint arXiv:2105.02788*, 2021. 8

[44] Ishit Mehta, Michaël Gharbi, Connelly Barnes, Eli Shechtman, Ravi Ramamoorthi, and Manmohan Chandraker. Modulated periodic activations for generalizable local functional representations. *arXiv preprint arXiv:2104.03960*, 2021. 6, 18, 20, 21

[45] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018. 21, 22

[46] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 1, 2

[47] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 1, 2, 4

[48] Yutaro Miyauchi, Yusuke Sugano, and Yasuyuki Matsushita. Shape-conditioned image generation by learning latent appearance representation from unpaired data. In *Asian Conference on Computer Vision*, pages 438–453. Springer, 2018. 7, 22

[49] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022. 8

[50] Thu Nguyen-Phuoc, Chuan Li, Stephen Balaban, and Yong-Liang Yang. RenderNet: a deep convolutional network for differentiable rendering from 3D shapes. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7902–7912, 2018. 3, 15

[51] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 1, 21

[52] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3504–3515, 2020. 1, 2

[53] Anton Obukhov, Maximilian Seitzer, Po-Wei Wu, Semen Zhydenko, Jonathan Kyl, and Elvis Yu-Jing Lin. High-fidelity performance metrics for generative models in PyTorch, 2020. Version: 0.3.0, DOI: 10.5281/zenodo.4957738 (Apache License). 8, 22

[54] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. *arXiv preprint arXiv:2104.10078*, 2021. 2, 3

[55] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 1, 2

[56] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alch'e Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[57] Dario Pavllo, Graham Spinks, Thomas Hofmann, Marie-Francine Moens, and Aurelien Lucchi. Convolutional generation of textured 3D meshes. *arXiv preprint arXiv:2006.07660*, 2020. 1

[58] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 8, 20, 21

[59] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 22

[60] Daniel Rebain, Wei Jiang, Soroosh Yazdani, Ke Li, Kwang Moo Yi, and Andrea Tagliasacchi. DeRF: Decomposed radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14153–14161, 2021. 3

[61] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. KiloNeRF: Speeding up neural radiance fields with thousands of tiny mlps. *arXiv preprint arXiv:2103.13744*, 2021. 3

[62] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014. 7, 20

[63] Lukasz Romaszko, Christopher KI Williams, Pol Moreno, and Pushmeet Kohli. Vision-as-inverse-graphics: Obtaining a rich 3D explanation of a scene from a single image. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 851–859, 2017. 1

[64] Azriel Rosenfeld and John L Pfaltz. Distance functions on digital pictures. *Pattern recognition*, 1(1):33–61, 1968. 2

[65] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. *arXiv preprint arXiv:1705.09367*, 2017. 21, 22

[66] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G Baraniuk, and Ashok Veeraraghavan. Miner: Multiscale implicit neural representations. *arXiv preprint arXiv:2202.03532*, 2022. 8

[67] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3D-aware image synthesis. *arXiv preprint arXiv:2007.02442*, 2020. 1, 8

[68] Nicholas Sharp et al. Polyscope, 2019. `www.polyscope.run` v1.2.0. (MIT License). 16

[69] Daeyun Shin, Charless C Fowlkes, and Derek Hoiem. Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3061–3069, 2018. 7

[70] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020. 5

[71] Vincent Sitzmann, Semon Rezchikov, William T Freeman, Joshua B Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *arXiv preprint arXiv:2106.02634*, 2021. 3, 5

[72] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*, 2019. 1, 2, 4

[73] Stanford Computer Graphics Laboratory. The Stanford 3D scanning repository. `http://graphics.stanford.edu/data/3Dscanrep/`. Accessed: 09/08/21. 16

[74] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3D shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021. 2

[75] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3D reconstruction networks learn? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3405–3414, 2019. 7

[76] Shubham Tulsiani, Nilesh Kulkarni, and Abhinav Gupta. Implicit mesh reconstruction from unannotated image collections. *arXiv preprint arXiv:2007.08504*, 2020. 1

[77] Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017. 3, 7, 15

[78] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 311–318, 1994. 16

[79] Rahul Venkatesh, Tejan Karmali, Sarthak Sharma, Aurobrata Ghosh, R Venkatesh Babu, László A Jeni, and Maneesh Singh. Deep implicit surface point prediction networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12653–12662, 2021. 2

[80] Rahul Venkatesh, Sarthak Sharma, Aurobrata Ghosh, Laszlo Jeni, and Maneesh Singh. Dude: Deep unsigned distance embeddings for hi-fidelity representation of complex 3D surfaces. *arXiv preprint arXiv:2011.02570*, 2020. 2, 6

[81] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3D mesh models from single RGB images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 7, 19

[82] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Hang Yu, Wei Liu, Xiangyang Xue, and Yu-Gang Jiang. Pixel2mesh: 3D mesh model generation via image guided deformation. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 7, 19

[83] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 2

[84] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3D shape reconstruction via 2.5D sketches. *Advances in Neural Information Processing Systems*, 30:540–550, 2017. 3, 15

[85] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3D object reconstruction without 3D supervision. *Advances in Neural Information Processing Systems*, 29:1696–1704, 2016. 3, 15

[86] Mingyue Yang, Yuxin Wen, Weikai Chen, Yongwei Chen, and Kui Jia. Deep optimized priors for 3D shape modeling and reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3269–3278, 2021. 5

[87] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *arXiv preprint arXiv:2106.12052*, 2021. 2, 3

[88] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. *arXiv preprint arXiv:2103.14024*, 2021. 3

[89] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. PixelNeRF: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 1, 2

[90] Alan Yuille and Daniel Kersten. Vision as Bayesian inference: analysis by synthesis? *Trends in cognitive sciences*, 10(7):301–308, 2006. 1

[91] Jingyang Zhang, Yao Yao, and Long Quan. Learning signed distance field for multi-view surface reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6525–6534, 2021. 2

[92] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017. 8, 21, 22

[93] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Cyclegan and pix2pix in PyTorch. *junyanz/pytorch-CycleGAN-and-pix2pix*, Nov. 2021. `https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix` (BSD License). 22

[94] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3D representations. *Advances in Neural Information Processing Systems*, 31:118–129, 2018. 7

[95] Ehsan Zobeidi and Nikolay Atanasov. A deep signed directional distance function for object shape representation. *arXiv preprint arXiv:2107.11024*, 2021. 2, 5