Probabilistic Directed Distance Fields for Ray-Based Shape Representations

Tristan Aumentado-Armstrong, Stavros Tsogkas, Sven Dickinson, Allan Jepson

***Abstract*—In modern computer vision, the optimal representation of 3D shape remains task-dependent. One fundamental operation applied to such representations is differentiable rendering, which enables learning-based inverse graphics approaches. Standard explicit representations are often easily rendered, but can suffer from limited geometric fidelity, among other issues. On the other hand, implicit representations generally preserve greater fidelity, but suffer from difficulties with rendering, limiting scalability. In this work, we devise *Directed Distance Fields* (DDFs), which map a ray or oriented point (position and direction) to surface visibility and depth. This enables efficient differentiable rendering, obtaining depth with a single forward pass per pixel, as well as higher-order geometry with only additional backward passes. Using probabilistic DDFs (PDDFs), we can model the inherent discontinuities in the underlying field. We then apply DDFs to single-shape fitting, generative modelling, and 3D reconstruction, showcasing strong performance with simple architectural components via the versatility of our representation. Finally, since the dimensionality of DDFs permits view-dependent geometric artifacts, we conduct a theoretical investigation of the constraints necessary for view consistency. We find a small set of field properties that are sufficient to guarantee a DDF is consistent, *without knowing* which shape the field is expressing.**

***Index Terms*—3D shape representations, differentiable rendering, implicit shape fields, multiview consistency.**

I. Introduction

THE representation of 3D geometry remains an open problem, with significant implications across computer vision, graphics, and artificial intelligence. Generally, the appropriate representation is application-dependent, determined by the ease of relevant operations, such as deformation, composition, learnability, and rendering. The latter operation is particularly important: since the earliest days of computer vision (e.g., [1], [2]), extraction of 3D structure by matching 2D observations has been an essential task. In the modern context, the analysis-by-synthesis (AbS) approach to representation learning [3], which treats vision as "inverse graphics", relies on rendering to form the bridge between 3D and 2D. For example, at the single-scene scale, differentiable volume rendering in neural radiance fields (NeRFs) [4] enables AbS extraction of a 3D representation that is highly effective for novel view synthesis (NVS). Similarly, implicit geometric fields, such as occupancy fields [5] and signed distance fields (SDFs) [6], have been used in conjunction with differentiable rendering as well (e.g., [7]–[11]). Finally, more abstract representations, learned through generative models, have also provided impressive results, connecting 2D and 3D via rendering [12]–[17].

All authors were with the Samsung AI Centre, Toronto, at the time of this work. Tristan Aumentado-Armstrong, Sven Dickinson, and Allan Jepson are also with the Department of Computer Science, University of Toronto.

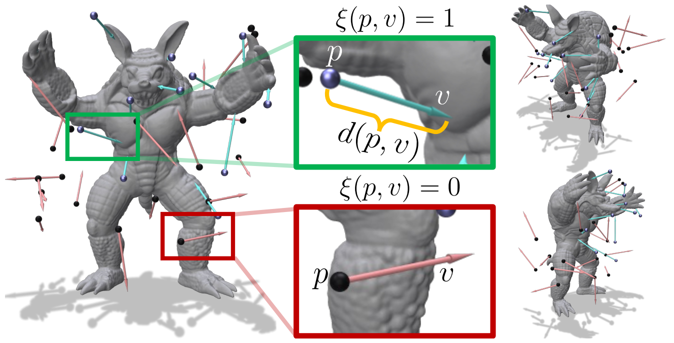

Figure 1: Basic depiction of a Directed Distance Field (DDF). Oriented points (or rays) are shown as a position p and direction v. Each ray is assigned a *visibility* value: $\xi(p,v)=1$ means it hits the shape, while $\xi(p,v)=0$ means it missed. For rays with $\xi(p,v)=1$, the *distance field*, $d(p,v)$, returns the distance between p and that intersection point (green box).

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 (e.g., [18], [19]), often leading to topological and textural fidelity constraints, Meshes can also struggle with the dependence of rendering efficiency on shape complexity, and the ad hoc "softening" procedures need to enable differentiability (e.g., [20], [21]). 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 [4], [7], [22], 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 general, rendering relies on "directed geometric queries" (DGQs), which ask whether and where a surface lies on a given ray; our strategy is to make such queries the basic operation of the representation.

More specifically, in this work, we explore *directed distance fields* (DDFs), a neural field representation of shape that uses only a single forward pass per DGQ. The definition is simple (see Fig. 1): for a given shape, we learn a 5D field that maps any position and orientation (i.e., oriented point or ray) to *visibility* (i.e., whether the shape exists from that position along that direction) and *distance* (i.e., how far the shape is along that ray, if it is visible). This enables efficient differentiable geometric rendering, compared to other common implicit

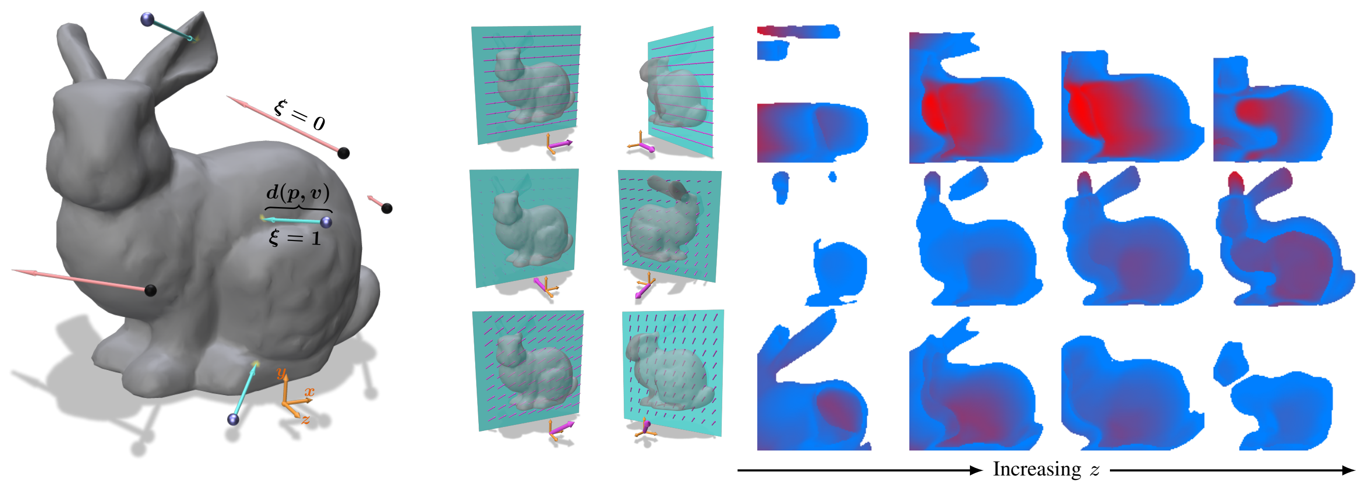

Figure 2: Two-dimensional spatial slices of a directed distance field (DDF). *Left*: depiction of visible oriented points (blue points, turquoise directions) that intersect the shape and those that miss the shape (black points, red directions) with $\xi = 0$. *Middle*: per row, illustrations of one slice 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.

approaches, but can still capture detailed geometry, including higher-order differential quantities and internal structure.

Fig. 2 illustrates how DDFs can be viewed as implicitly storing all possible depth images of a given shape (i.e., from all possible cameras), reminiscent of a light field, but with geometric distance instead of radiance. Such a field is inherently discontinuous (see Fig. 3), presenting issues for differentiable neural networks, but is advantageous in rendering, since a depth image can be computed with a single forward pass per pixel. More significantly, however, the increased dimensionality permits *view inconsistencies* (meaning the DDF can represent an aberrant 5D entity, where *geometry is view-dependent*, rather than a consistent 3D shape). We mitigate these in our applications by regularizing properties of the field, including analogues of the Eikonal constraint for classical distance fields, though these alone cannot *guarantee* view consistency (VC) in a mathematical sense.

Yet, knowing the theoretical conditions (i.e., field properties) necessary to avoid such inconsistencies can be practically valuable (e.g., for formally defining geometric consistency, or for regularizer design). Hence, we examine sufficient conditions to ensure VC, by introducing a notion of "inducement", whereby a shape can be used to construct a perfect DDF. Then, starting from an *un*constrained field (not necessarily representing a shape), we devise local conditions (independent of a fixed pre-defined shape) that ensure the existence of a shape that induces the field. Specifically, the DDF subfields (i.e., visibility ξ and depth d) each have three corresponding constraints, which independently ensure each field is "internally" consistent, as well as two *compatibility* conditions that mutually constrain ξ and d. To our knowledge, this analysis is the first theoretical investigation of when the higher-dimensional, ray-based DDFs formally represent 3D shapes.

Thus, our contributions can be summarized as follows:

1) We define the DDF, a 5D mapping from any position and viewpoint to depth and visibility (§III).
2) By construction, our representation allows differentiable rendering via a single forward pass per pixel (§III-C), without restrictions on the shape (topology, watertightness) or field queries (internal structure).
3) We construct probabilistic DDFs (PDDFs), which differentiably model discontinuities (§III-D).
4) We examine several geometric properties of DDFs (§III-A–III-C) and use them in our method (§IV).
5) We apply DDFs to fitting shapes (§V-A), single-image reconstruction (§V-B), and generative modelling (§V-C).
6) We show that DDFs can be used for path tracing, using internal structure modelling capabilities (§VI).
7) We present a theoretical analysis of the conditions under which DDFs can guarantee multiview consistency (i.e., act as an shape representation) in §VII. These results hold for any ray-based geometry field, including concurrent work independent from DDFs (see also §II).

We remark that this work is an extension of a previous conference paper [23]. As noted just above in (6), the largest novel contribution of this work is a theoretical analysis of the conditions required for view consistency (§VII). Since the DDF has a direction-dependence (i.e., is 5D), it can produce fields that do not properly correspond with a 3D shape, due to inconsistencies (i.e., geometry existing from one viewpoint but not others). Hence, our interest is in what constraints on the field can be imposed, to ensure consistency and thus a representation of shape. In addition, we demonstrate rendering efficiency in §V-A4, and provide a new interpretation of recursive DDF calls as inter-reflections between surfaces, and apply this to path tracing (§VI).

II. Related Work

Our work falls under distance field representations of shape, which have a long history in computer vision [24], recently culminating in signed/unsigned distance fields (S/UDFs) [6], [25], [26] and related methods [27], [28]. Compared to explicit ones, implicit shapes can capture arbitrary topologies with high fidelity [5], [6], [29]. Several works examine differentiable rendering of implicit fields [7]–[9], [22], [29]–[31] (or combine it with neural volume rendering [32]–[35]), generally based on approaches that require many field queries per pixel, such as sphere tracing. 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 §III-A and §V-A3).

Separately, neural radiance fields (NeRFs) [4] provide a powerful 3D representation of both geometry and texture, with increasingly impressive results in NVS (e.g., [36]). However, the standard differentiable volume rendering used by NeRFs is computationally expensive, requiring many forward passes per pixel, though recent work has improved on this (e.g., [37]–[43]). Furthermore, the distributed nature of the density makes extracting explicit geometric details (including higher-order surface information, such as normals and curvatures) more difficult (e.g., [33], [34]). In contrast, though focused purely on geometry, DDFs are rendered with a single forward pass and enable non-local computation of higher-order surface properties. Further, by noting that light paths between surfaces correspond to recursive DDF calls, we show that DDFs are amenable to path-tracing (see §VI), whereas NeRFs require substantial modification (e.g., [44]).

More similar to DDFs are Light Field Networks (LFNs) [45], which render 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, LFNs cannot render from viewpoints between occluded objects. The neural 4D light field (NeuLF) [46] focuses on NVS and leverages per-ray depth estimates, but is also only 4D, like LFNs. More recently, Attal et al. [47] improve on efficient NVS via a ray-space embedding within the light field. In this work, we instead focus on (i) the representation of geometry and (ii) theoretical analysis of the conditions under which higher-dimensional shape fields can ensure multiview consistency.

Several ray-based shape representations have appeared concurrently or subsequently to our prior work [23]. The Signed Directional Distance Field (SDDF) [48] also maps position and direction to depth, but introduces a fundamental difference in structure modelling, due to its signed nature. 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, including a lack of internal structure, in a manner similar to LFNs. Houchens et al. [49] showcase the utility of DDFs (called neural omnidirectional distance fields) in inter-converting between a variety of shape representations. The Primary Ray-based Implicit Function (PRIF) [50] representation also suggested the use of DDFs, using a novel ray parameterization, on several tasks; however, it did not investigate the theoretical requirements of exact representation, as we consider in this work, nor certain other issues (e.g., discontinuity modelling, relations to light transport).

Several other new representations and models are closely related to DDFs. Neural vector fields (NVFs) [51] compute the displacement from any given query to the *closest* surface point. Like UDFs, this representation is closely connected to DDFs, in that it corresponds to fixing a special direction field, which we call the Minimal Direction Field (MDF), $v^* : \mathbb{R}^3 \to \mathbb{S}^2$, which points to the closest surface point from p. The NVF can then be computed via $\text{NVF}(p) = d(p, v^*(p))v^*(p)$, for a given DDF d. Note also that $\text{UDF}(p) = d(p, v^*(p))$; see §V-A3 for details. In terms of other tasks, Directed Ray Distance Functions [52] model surfaces via per-ray depth functions and their zeroes, to obtain scene representations from posed RGBD images. RayMVSNet [53] learns to generate per-ray depths, but in a multiview stereo setting, based on learned features from images. In a robotics context, Zhang et al. [54] apply DDFs to modelling hand-object interaction. Finally, the FIRe [55] model combines an SDF with a DDF, in order to obtain efficiency while maintaining multiview consistency.

In contrast to these works, the primary novel contribution of this paper is the comprehensive characterization of theoretical requirements for view consistency of DDFs (§VII). Since the aforementioned works largely describe the same field mathematically, our results are equally applicable to them as well, and may suggest regularizations and architectural designs with better theoretical properties.

III. Directed Distance Fields

Let $S \subset \mathcal{B}$ be a 3D shape, where $\mathcal{B} \subset \mathbb{R}^3$ is a bounding volume acting as the field domain. Consider a position $p \in \mathcal{B}$ and view direction $v \in \mathbb{S}^2$. We define S to be *visible* from (p, v) if the line $\ell_{p,v}(t) = p + tv$ intersects S for some $t \geq 0$.
• **Visibility.** We write the binary visibility field (VF) for S as $\xi(p, v) = \mathbb{1}[S \text{ is visible from } (p, v)]$. For convenience, we refer to an oriented point (p, v) as visible if $\xi(p, v) = 1$.
• **Depth.** We then define a distance or depth field as a non-negative scalar field $d : \mathcal{B} \times \mathbb{S}^2 \to \mathbb{R}_+$, which maps from any visible oriented point 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) (i.e., $\xi(p, v) = 1$).
• **DDF.** A *Directed Distance Field* (DDF) is simply a tuple of fields (ξ, d). See Figs. 1 and 2 for illustrations.

A. Geometric Properties

DDFs satisfy several useful properties (see §A for proofs).

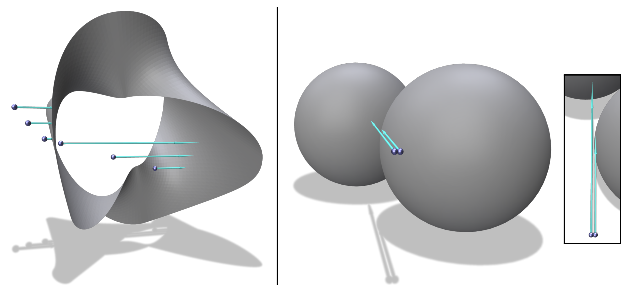
Figure 3: Inherent DDF discontinuities. Left: *surface discontinuities*, where p passes through S. Right: *occlusion discontinuities*, where v or p moves over an occlusion boundary.

• **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 $\nabla_p d(p, v) \in \mathbb{R}^{1\times 3}$. This also implies $||\nabla_p d(p, v)||_2 \geq 1$. The equivalent property for ξ asks that locally moving along the viewing line cannot change visibility[1]: $\nabla_p \xi(p, v) v = 0$.

• **Property II: Surface Normals.** The derivatives of implicit fields are closely related to the normals $n \in \mathbb{S}^2$ of S; e.g., $\nabla_q \text{SDF}(q)^T = n(q) \ \forall \ q \in S$. 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 $q(p, v) \in S$ and $n(p, v) \not\perp v$ (i.e., the change in d moving *off* the surface is undefined). From any (p, v) that "looks at" $q \in S$, the normals at q can be computed via $n(p, v) = \varsigma \nabla_p d(p, v)^T / ||\nabla_p d(p, v)||_2$, where we choose $\varsigma \in \{-1, 1\}$ such that $n^T v < 0$ (so that n always points back to the query)[2].

• **Property III: Gradient Consistency.** Intuitively, given a visible (p, v), infinitesimally perturbing the *viewpoint* v by δ_v should be similar to pushing the *position* p along δ_v. In fact, the following differential constraint holds (see Supp. §A-C): $\nabla_v d(p, v) = d(p, v) \nabla_p d(p, v) \mathcal{P}_v$, where $\mathcal{P}_v = I - vv^T$ is an orthogonal projection, since v cannot be perturbed along itself.

• **Property IV: Deriving Unsigned Distance Fields.** An unsigned distance field (UDF) can be extracted from a DDF via the following optimization: $\text{UDF}(p) = \min_{v \in \mathbb{S}^2} d(p, v)$, constrained such that $\xi(p, v) = 1$, allowing them to be procured if needed (see §V-A3). UDFs remove the discontinuities from DDFs (see §III-D and Fig. 3), but cannot be rendered as easily nor 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 Supp. §A-D). This allows computing surface properties, such as curvatures, from any visible oriented position, simply by querying the network; see Fig. 8 for an example.

• **Neural Geometry Rendering.** Differentiable generation of geometry, such as depth and normals (e.g., [56]–[59]), can sometimes be written as parallelized DDFs. In such cases, regardless of architecture, the properties of DDFs discussed in this paper still hold (see also Supp. §A-E).

[1]Except when moving *through* certain surfaces (see §VII).

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

B. The View Consistency Inequality

Ideally, DDFs should maintain *view consistency* (VC). One form of VC can be expressed by a simple inequality, which demands that an *opaque* position viewed from one direction (e.g., the point q_1, from v_1, in Fig. 4) must be opaque from all directions. That is, when a point q_1 is opaque from one direction, but is also on another directed ray from, say, $\tau_2 = (p_2, v_2)$, then its depth (from τ_2) is lower-bounded by the distance to that known surface position.

More specifically, consider two oriented points $\tau_1 = (p_1, v_1)$ and $\tau_2 = (p_2, v_2)$. Assume that τ_1 is visible, so $q_1 = p_1 + d(\tau_1) v_1 \in S$, and $\exists \ t > 0$ such that $\ell_{\tau_2}(t) = p_2 + t v_2 = q_1$. I.e., the lines of sight from τ_1 and τ_2 intersect at the surface point q_1. Then, (i) $\xi(\tau_2) = 1$, and (ii) $d(\tau_2) \leq ||p_2 - q_1||_2 = t$. These can be seen by the definition of ξ and d. For (i), since there exists a surface (at q_1) along ℓ_{τ_2}, the oriented point (τ_2) must be visible. For (ii), $d(\tau_2)$ can be no further than t, since DDFs return the minimum distance to a point on S along the line $\ell_{\tau_2}(t)$, and hence its output can be no greater than the assumed distance t. In §VII, we examine the question of view consistency in greater detail.

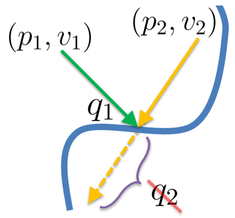

Figure 4: The view consistency inequality, $d(p_2, v_2) \leq ||p_2 - q_1||_2$.

C. Rendering

A primary application of DDFs is rapid differentiable rendering. In contrast to some mesh rasterizers (e.g., [21]), there is no dependence on the complexity of the underlying shape, after training. Unlike most implicit shape fields [4], [7], [22], DDFs only require a single forward pass per pixel.

DDF rendering is simply ray-casting. Given a camera Π with position $p_0 \in \mathcal{B}$, for a pixel with 3D position ρ, we cast a ray $r(t) = p_0 + t v_\rho$, where $v_\rho = (\rho - p_0)/||\rho - p_0||_2$, into the scene via a single query $d(p_0, v_\rho)$, which provides the depth pixel value. For $p \notin \mathcal{B}$, we first compute the intersection $p_r \in \partial\mathcal{B}$ 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 (ξ, d) from arbitrary oriented points, including those unseen during fitting.

D. Discontinuity Handling: Probabilistic DDFs

DDFs are inherently discontinuous functions of p and v. Whenever (i) p passes through the surface S or (ii) p or v moves across an occlusion boundary, a discontinuity in $d(p, v)$ occurs (see Fig. 3). We therefore modify the DDF, to allow a C^1 network to represent the discontinuous field. In particular, we alter d to output probability distributions over depths, rather than a single value. Let $\mathbb{P}_\ell$ be the set of probability distributions with support on some ray $\ell_{p,v}(t) = p + tv$, $t \geq 0$.

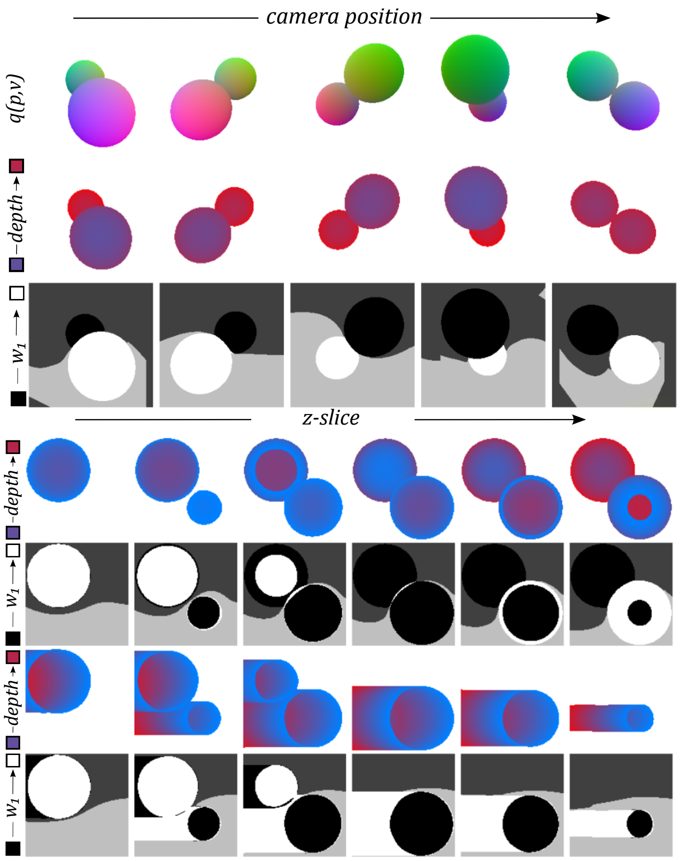

Figure 5: Discontinuous depth with PDDFs, via the weight field (WF). Here, $K = 2$, so $w_1 = 1 - w_2$ (see §III-D). The *upper inset* (rows 1–3) shows WF transitions in renders. In the third row, white vs black mark high vs low w_1, and thus which surface (d_1 vs d_2) is active, for high ξ. Light and dark grey demarcate the non-visible (low ξ) counterparts of white and black. The change in dominant weight (w_1 vs w_2) at occlusion edges permits discontinuities. The *lower inset* (rows 4–7) shows WF transitions using slices in z. Rows four and six depict distances, with fixed v ($(0,0,-1)$ and $(1,0,0)$, respectively) and varying p across the image. Rows five and seven show WF values, as in row three. Notice the WF switching upon p transitioning through a surface.

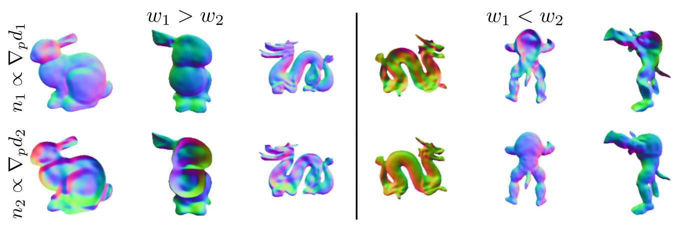

Figure 6: PDDF renders of n_1 and n_2. Though not explicitly enforced, a "see-through effect" occurs when the lower-weight field models the surface *behind* the currently visible one (i.e., the PDDF components model separate surfaces).

Then $d : \mathcal{B} \times \mathbb{S}^2 \to \mathbb{P}_\ell$ is a *probabilistic DDF* (PDDF). The visibility field, $\xi(p, v)$, is unchanged in the PDDF.

For simplicity, herein we restrict $\mathbb{P}_\ell$ to be the set of mixtures of Dirac delta functions with K components. Thus, the network output is $P_{p,v}(d) = \sum_i w_i \delta(d - d_i)$ 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. 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. 5 showcases example transitions, with respect to occlusion (a) and surface collision (b) discontinuities; Fig. 6 visualizes the normals field for each depth component. Notationally, a PDDF is simply a DDF: $d(p, v) := d_{i^*}(p, v)$.

IV. Learning DDFs

1) Mesh Data Extraction: Given a mesh, we can obtain visibility ξ and depth d by ray-casting from any (p, v). In total, we consider six types of data samples (see Fig. 7): *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.

2) 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* trains the highest probability depth component: $\mathcal{L}_d = \xi|\widehat{d}_{i^*} - d|^2$.The *visibility objective*, $L_\xi = \text{BCE}(\xi, \widehat{\xi})$, is the binary cross entropy between the visibility prediction and the ground truth. A first-order *normals loss*, $\mathcal{L}_n = -\xi|n^T \widehat{n}_{i^*}(p, v)|$, 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

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

applied on the visibility and each delta component of d, analogous to prior SDF work (e.g., [60]–[64]).

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. 3 and 5), namely surface (S) and tangent (T) data. The complete PDDF shape-fitting loss is then

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

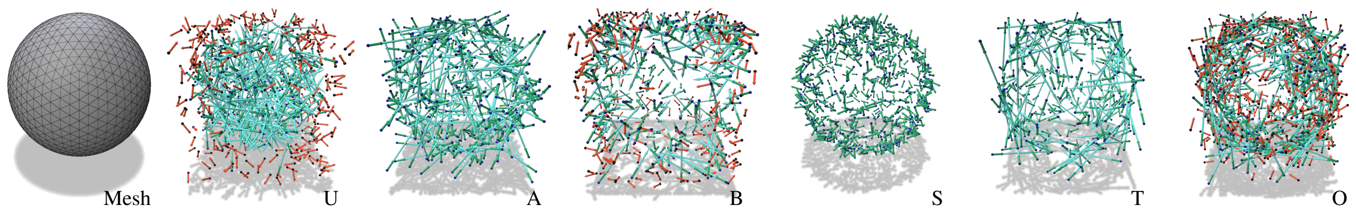

Figure 7: Illustration of mesh-derived data types. Left to right: input sphere mesh, U, A, B, S, T, and O data. Visible points depict p in blue, v in green, and a line from p to q in turquoise; non-visible points depict p in black and v in red.

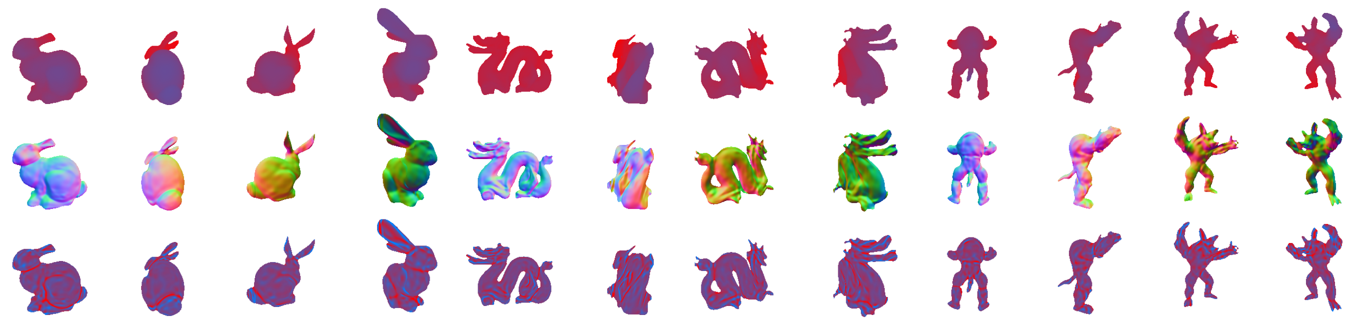

Figure 8: Renders of DDF fits to shapes. Rows: depth, normals, and mean curvature. Each quantity is directly computed from the learned field, using network derivatives at the query oriented point (p, v) per pixel.

V. Applications

A. Single Field Fitting

Fig. 8 shows single-object fits, via PDDF renderings with a *single network evaluation per pixel*. Normals and curvatures are obtained using only one or more backward passes for the same oriented point used in the single forward pass. We implement the PDDFs with SIREN [65], as it allows for higher-order derivatives and has previously proven effective (e.g., [66], [67]). (See Supp. §B for details.)

1) Data-type Ablation: We perform a small-scale ablation experiment to discern the importance of each data sample type. In particular, we consider six scenarios on a single shape (the Stanford Bunny), in each of which we train with 100K samples of each type *except one*, which is removed. We consider one other scenario that has the same number of total points, but *no* single type is ablated (i.e., 83,333 samples per type). We then measure the depth and visibility prediction error on 25K held-out samples of each data type, including the ablated one.

Results are shown in Table I. In most cases, the worst or second-worst errors on a given data type are incurred by models without access to that type. One anomaly is ablating S-type data, which damages performance across all sample types. This may be due to difficulties in knowing when to transition between weight field components. Another outlier is $\mathcal{L}_\xi$-A, where ablating T-type data is the most damaging (removing A-type is third); we suspect this is because T-type samples are effectively the hardest subset of A-type data, and hence have an outsized impact upon removal. Further, A-type data has more overlap with others (e.g., U and B).

Finally, we remark that our data sampling strategies provide an inductive bias. For instance, B-type data will be most important when rendering far from the shape, while more A-type samples upweight visibile vs. non-visible parts of the scene. Overall, while this is a small-scale (single-shape) analysis, it does suggest that each data type has information that the others cannot fully replace, especially U, O, S, and T.

2) Internal Structure and Composition: 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 works [45], [48], which do 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 [21], [68]. Formally, given a set of N DDFs $\zeta = \{T^{(i)}, \xi^{(i)}, d^{(i)}, \mathcal{B}^{(i)}\}_{i=1}^N$, 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

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

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

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

	Minimum Distance Error (L_1; ×10) ↓						Visibility Error (BCE) ↓					
	$\mathcal{L}_{d,1}$-U	$\mathcal{L}_{d,1}$-A	$\mathcal{L}_{d,1}$-B	$\mathcal{L}_{d,1}$-O	$\mathcal{L}_{d,1}$-T	$\mathcal{L}_{d,1}$-S	$\mathcal{L}_{\xi}$-U	$\mathcal{L}_{\xi}$-A	$\mathcal{L}_{\xi}$-B	$\mathcal{L}_{\xi}$-O	$\mathcal{L}_{\xi}$-T	$\mathcal{L}_{\xi}$-S
U	**0.58**	0.79	0.18	0.49	0.75	0.47	**2.11**	0.03	0.07	0.56	0.14	0.05
A	0.48	0.82	0.20	0.49	0.81	0.54	0.20	0.05	0.07	0.49	0.18	0.07
B	0.37	0.67	0.20	0.46	0.73	0.58	0.20	0.03	0.10	0.50	0.15	0.06
O	0.39	0.67	0.18	**0.56**	0.70	0.59	0.28	0.01	**0.11**	**1.71**	0.03	0.03
T	0.39	0.70	0.19	0.45	0.84	0.65	0.19	**0.09**	0.06	0.32	**0.48**	0.07
S	0.55	**0.95**	**0.23**	0.51	**0.89**	**1.43**	0.14	0.06	0.06	0.42	0.17	**0.48**
–	0.45	0.75	0.19	0.50	0.77	0.67	0.23	0.04	0.08	0.56	0.15	0.07

Table I: Data type ablation results (see §V-A1). Rows: type of data (i.e., (p, v) sample type) ablated. Columns: errors on held-out data (left: min. distance loss, but computed with L_1 instead of L_2; right: binary cross-entropy-based visibility loss). Each column computes the error on a different sample type (e.g., $\mathcal{L}_{d,1}$-A is the minimum distance error on A-type data), via 25K held-out samples per type. Per loss type, the **red** numbers are the *worst* (highest) error cases; the **pink** numbers are the *second-worst* error cases. Each scenario uses 100K samples of each data type, except for the ablated one; the "–" case uses 83,333 samples of all six data types (to control for the total dataset size). Usually, performance on a data type is worst or second-worst when that type is ablated, suggesting the utility of different samples, particularly for hard types (e.g., S and T).

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. 9, 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. For comparison, to show the improved scaling of composing DDFs, we also attempted to fit the same scene using the single-object fitting procedure above. For fairness, we doubled the number of data samples extracted, as well as the size of each hidden layer. In comparison to the top inset of Fig. 9, this naive approach struggles to capture some high frequency details, though we suspect this could be mitigated to some extent by better sampling procedures.

3) 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 $\text{UDF}(p) = d(p, v^*(p))$. This *Minimal Direction Field* (MDF), v^*, points to the closest point on S.[3] We obtain it by optimizing an objective that prefers high visibility and low depth for a given $v^*(p)$. Unlike directly fitting a UDF, this requires handling local minima for v^* and non-visible (low ξ) directions. (See Fig. 10 for visualizations and Supp. §C for details.) When the normals exist, $v^*(p) = -n(p, v^*(p))$, in the notation of Property II. Recent work has highlighted the utility of UDFs over SDFs [25], [26], [69]; in the case of DDFs, extracting a UDF or v^* may provide useful auxiliary information for some tasks.

4) Efficiency: To illustrate the gain in rendering efficiency, we compare to an existing differentiable sphere tracer, DIST [7], on DeepSDF [6] models. Both representations rendered a 1024^2 depth image. The implicit field architectures are roughly comparable: DeepSDF used a ReLU-based MLP with eight hidden layers, while our PDDF used a SIREN MLP [65] (i.e., sine nonlinearity) with seven hidden layers, both with hidden dimensionalities all equal to 512. Per pixel, DIST must sphere trace along each camera ray, while the PDDF need only call the network once. We find that DIST takes 5.3 seconds per render, while the DDF takes <0.01 seconds, a roughly $\sim 100\times$ speedup. While this simplified setting does not completely characterize the tradeoff, it does suggest that DDFs obtain

[3]Discontinuities in v^* occur at surfaces as before, but also on the medial surface of S in $\mathcal{B}$. At such positions, there are multiple valid values of v^*.

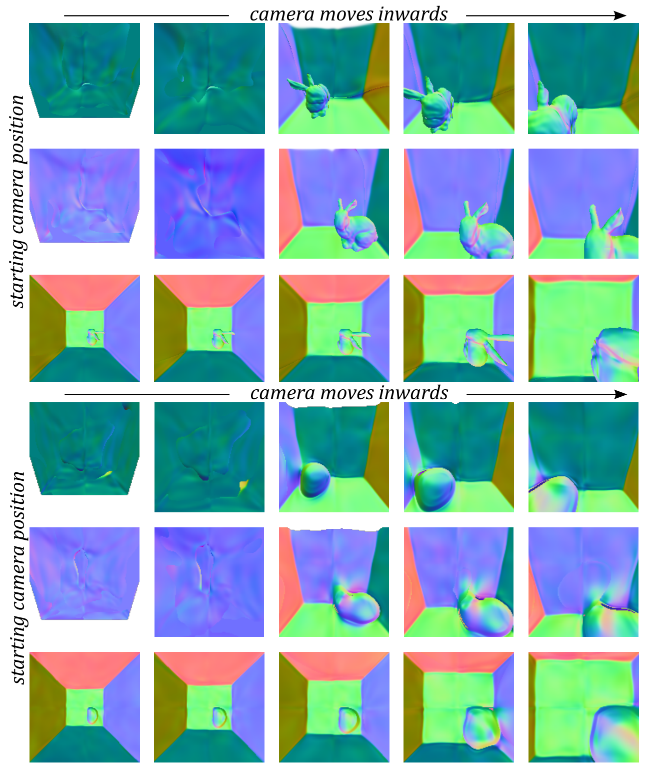

Figure 9: Example of internal structure rendering and composition. Colours correspond to DDF surface normals (as in Fig. 8), from the DDF. The top insets compose two smaller DDFs, while the bottom set uses a single larger monolithic one.

an advantage in efficiency. Further, the runtime of sphere tracers depends on both the shape and the camera: for instance, simply moving the camera closer to the shape, which reduces the opportunities for certain runtime optimizations, increases rendering time by $\sim 1.8\times$. In contrast, DDFs are unaffected by either element. See Supp. §B-C for details.

B. Single-Image 3D Reconstruction (SI3DR)

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

	DDF					PC-SIREN				P2M [70], [71]	3DR [72]
	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		
$D_C \downarrow$	0.300	0.346	0.629	0.730	0.787	0.339	0.364	0.814	0.842	0.452	1.057
$F_\tau \uparrow$	68.95	61.65	53.25	57.98	51.36	69.44	64.76	56.74	52.65	64.45	39.83
$F_{2\tau} \uparrow$	83.15	78.65	70.41	72.83	68.41	81.94	79.91	70.47	68.49	78.65	57.76

Table II: Single-image 3D reconstruction results. Columns: L/S refer to sampling 5000/2466 points (2466 being used in P2M), $\Pi_g/\widehat{\Pi}$ means using the true versus predicted camera, and $\widehat{\Pi}_\nabla$ denotes camera position correction using gradient descent. Metrics: D_C is the Chamfer distance ($\times 1000$) and F_τ is the F-score ($\times 100$) at threshold $\tau = 10^{-4}$. PC-SIREN is our matched-architecture baseline; Pixel2Mesh (P2M) and 3D-R2N2 (3DR) are baselines using different shape modalities (numbers from [71]). Note: scenarios using Π_g (i.e., evaluating in canonical object coordinates) are not directly comparable to P2M or 3DR, but serve to isolate pose vs. shape error. 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.

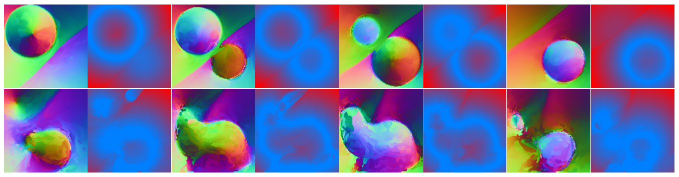
Figure 10: The MDF, v^*, and its UDF (i.e., $d(p, v^*(p))$), in odd and even columns respectively, for two scenes (rows). For v^*, colours are 3D components; for the UDF, blue to red means near to far distances. Each image is a spatial slice in z, where adjacent MDF-UDF pairs have the same z. Notice the colour change in v^* as the slice moves through the shape, due to the closest surface switching from the front to the back of the shape (and thus flipping v^*). Some difficulties are also visible, (e.g., near surface intersections, where v^* is ill-defined).

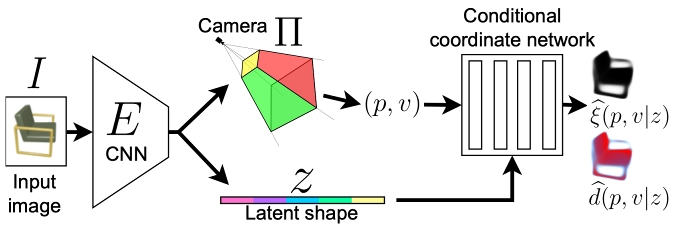

Figure 11: CPDDF-based SI3DR. A camera and latent shape are predicted from an image. The latent vector conditions a CPDDF, which can render geometry from any viewpoint.

encoder $E(I) = (\widehat{z}_s, \widehat{\Pi})$ (see Fig. 11). For decoding, we use a *conditional* PDDF (CPDDF), which computes depth $\widehat{d}(p, v|z_s)$ and visibility $\widehat{\xi}(p, v|z_s)$. 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 full objective is $\mathcal{L}_{\text{SI3DR}} = \gamma_{R,S}\mathfrak{L}_S + \gamma_{R,\Pi}\mathcal{L}_\Pi + \gamma_{R,M}\mathcal{L}_M$. We implement E as two ResNet-18 networks [73], while the CPDDF is a modulated SIREN [74]. For evaluation, camera extrinsics are either the predicted $\widehat{\Pi}$ or ground-truth Π_g (to separate shape and camera errors). See Supp. §D for details.

1) Explicit Sampling: SI3DR evaluation commonly uses point clouds (PCs). Thus, we present a simple PC extraction method, though it cannot guarantee uniform sampling over the shape S. Recall that $q(p, v) = p + d(p, v)v \in S$, if $\xi(p, v) = 1$.

Comparative Baseline	CPDDF Setting	CPDDF D_C	Baseline D_C	Diff. $\uparrow$
PC-SIREN Id. Ar. Baseline	$-\nabla_\Pi$ / $\widehat{\Pi}$	0.787	0.842	+0.055
	$+\nabla_\Pi$ / $\widehat{\Pi}$	0.629	0.842	+0.213
3DR [72] Voxel Baseline	$+\nabla_\Pi$ / $\widehat{\Pi}$	0.629	1.057	+0.428
P2M [70] Mesh Baseline	$+\nabla_\Pi$ / $\widehat{\Pi}$	0.629	0.451	-0.178
	$+\nabla_\Pi$ / Π_g	0.346	0.451	+0.105

Table III: We compare CPDDFs to a field with identical architecture (Id. Ar.), which outputs PCs, and two explicit-shape-based baselines, with modality-specialized architectures (vs. our use of generic networks). The Chamfer distance D_C compares to the GT shape. "CPDDF Setting" records: (i) silhouette-based optimization (i.e., "$-\nabla_\Pi$" vs. "$+\nabla_\Pi$") and (ii) evaluation with the predicted vs. the GT camera (i.e., "$\widehat{\Pi}$" vs. "Π_g"). The Π_g scenario is not fair, (as it uses the GT Π), but shows that much error is in pose, not shape, prediction.

Thus, we sample $p \sim \mathcal{U}[\mathcal{B}]$, and wish to project them onto S. Then, v should be chosen to avoid $\xi(p, v) = 0$. Hence, for each p, we sample $V(p) = \{v_i(p) \sim \mathcal{U}[\mathbb{S}^2]\}_{i=1}^{n_v}$ and "compose" them to estimate $\widehat{v}^*(p)$ (as in §V-A2 and §V-A3, but without optimization), giving $q(p, \widehat{v}^*(p))$ as a point on S. Repeating this N_H times (starting from $p \leftarrow q$) also helps, if depths are less accurate far from S. We set $n_v = 128$ and $N_H = 3$ (see Supp. §D-C for ablation with $N_H = 1$).

2) Baselines: Our primary baseline alters the *shape representation*, while keeping the architecture and training setup as similar as possible. Specifically, it uses the same encoders as the DDF and an almost identical network for the decoder (changing only the input and output dimensionalities), but altered to output PCs directly (denoted PC-SIREN). In particular, the decoder is 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 $q = f_b(p) \in S$. Training uses the Chamfer distance D_C and $\mathcal{L}_\Pi$. We also compare to the mesh-based Pixel2Mesh (P2M) [70], [71] and voxel-based 3D-R2N2 (3DR) [72].

3) Results: We consider cars, planes, and chairs from ShapeNet [77], using the data from [72] (as in [70]). In Tables II and III, we show DDFs perform comparably to the architecture-matched PC-SIREN baseline. See Fig. 12 and Supp. Fig. 18 for visualizations. Generally, the inferred DDF shapes correctly reconstruct most inputs, including thin structures like chair legs, regardless of topology. The most

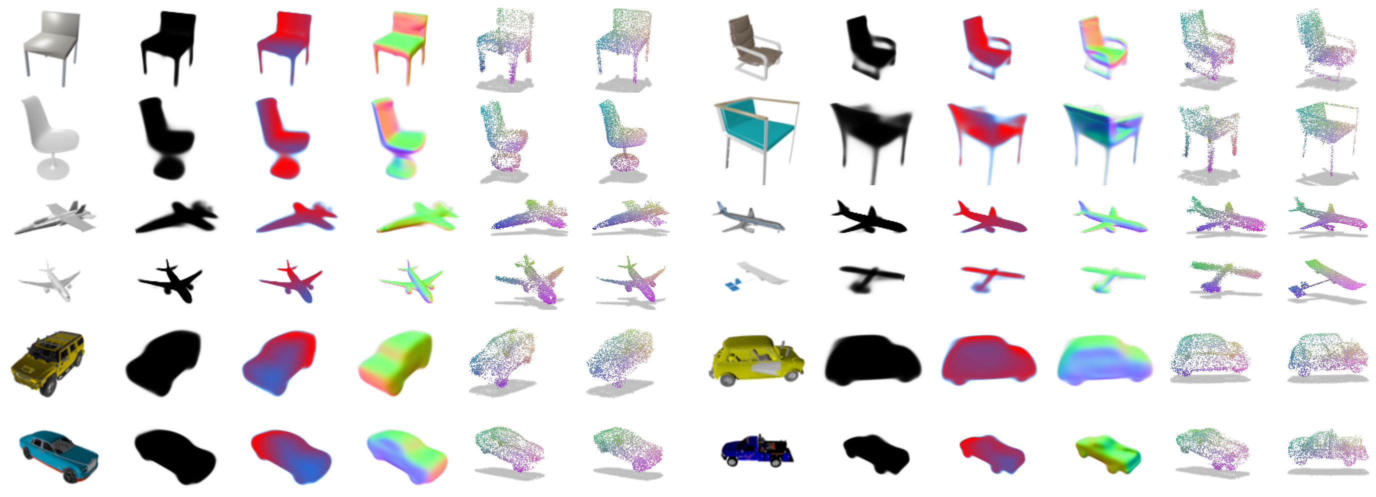

Figure 12: Single-image 3D reconstruction visualizations on held-out test data. Columns represent (i) the input RGB image, (ii) the visibility $\widehat{\xi}$, (iii) the depth $\widehat{d}$, (iv) the normals $\widehat{n}$, (v) the sampled point cloud (PC) from the DDF, and (vi) a sample from the ground-truth PC. Quantities (ii-v) are all differentiably computed from the CPDDF and $\widehat{\Pi}$, per point or pixel, without post-processing. PC colours denote 3D coordinates. A high-error example is in the lower-right of each category.

	Image GAN	CPDDF-based GAN	CGR [75]	Image VAE
FID [76] ↓	15	27	120	194
3D?	No	Yes	Yes	No

Table IV: Evaluation of our CPDDF-based 3D-aware generative model. We compared to a 3D-*un*aware image GAN (with the same critic), the Cyclic Generative Renderer (CGR) [75], which samples textured meshes and uses a differentiable rasterizer, and a 3D-*un*aware VAE. (See §V-C).

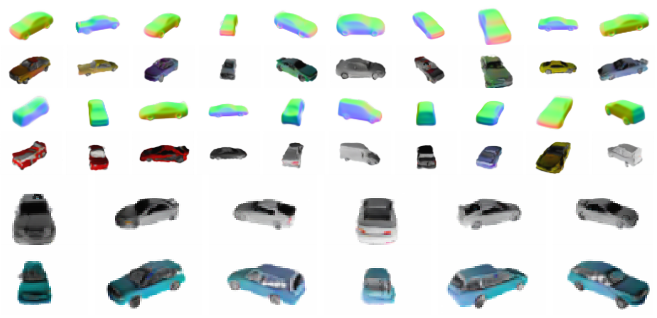

Figure 13: Above: CPDDF-based VAE and image GAN samples. Below: views with fixed texture and shape.

obvious errors are in pose estimation, but the DDF can also sometimes output "blurry" shape parts when it is uncertain (e.g., for relatively rare shapes). 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 input image alpha channel. While explicit modalities can be differentiably rendered (e.g., [20], [58], [78]), DDFs can do so by construction, without additional heuristics or learning. Further, (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 more versatile than PCs, enabling higher-order geometry extraction, built-in rendering, and PC extraction. Thus, 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 much of this error is due to camera prediction, though this is not directly comparable to P2M or 3DR. With Π_g, we are predicting in the object-centric, rather than camera-centric, frame. While each frame has benefits and downsides [79], [80], 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.

C. Generative Modelling with Unpaired Data

We also apply CPDDFs to 3D-aware generative modelling, using 2D-3D unpaired data (see, e.g., [75], [81]–[83]). 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) [84] on 3D shapes, then (ii) a generative adversarial network (GAN) [85], which convolutionally translates CPDDF-derived surface normal renders into colour images. Briefly, the VAE trains a PointNet encoder [86] and CPDDF decoder, while the GAN performs cycle-consistent image-to-image translation [87] (from normals to RGB). Fig. 13 displays some example samples, while Table IV shows quantitative results. While this underperforms a 2D image GAN with the same critic, it still outperforms samples from image VAEs or GAN-based textured low-poly 3D mesh renders [75] in image quality. See Supp. §E for details.

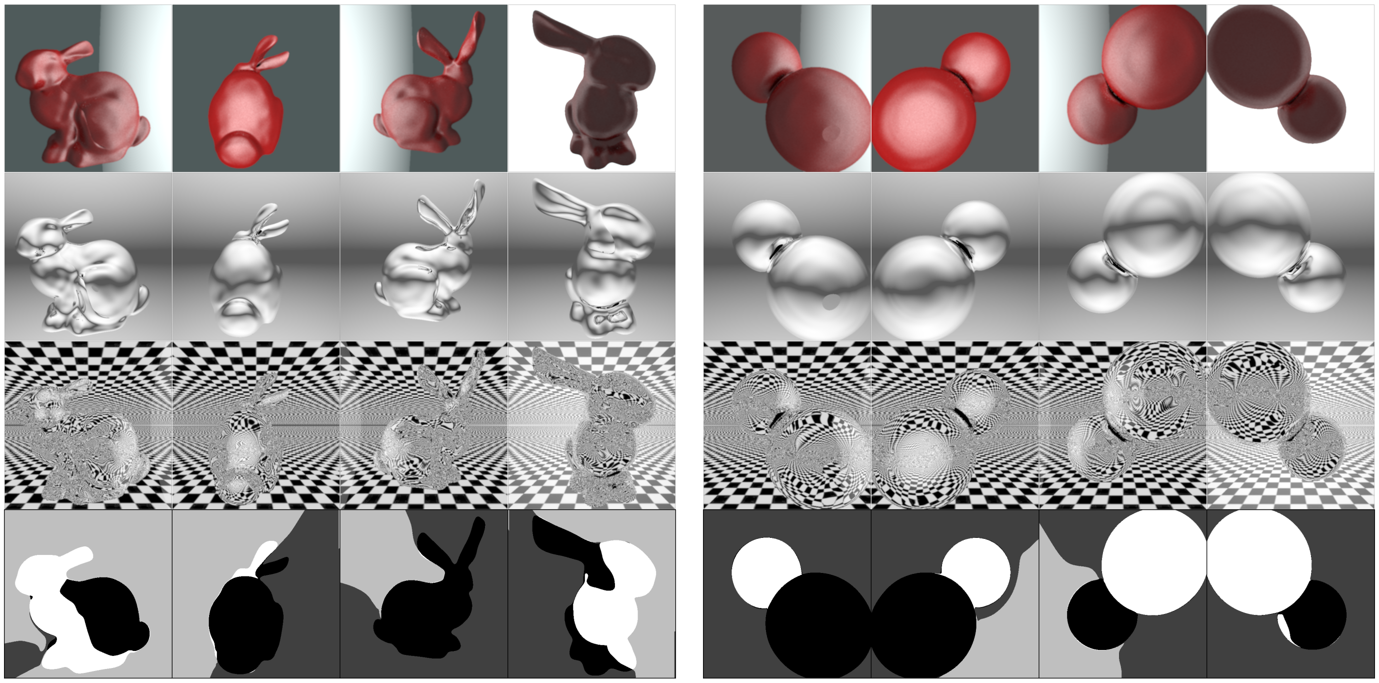

Figure 14: Examples of path-traced PDDFs under various lighting and material conditions. To better illustrate the lighting conditions, for rays that do not intersect the shape (i.e., $\xi = 0$), we return the environmental lighting associated to that direction. Rows 1 and 2-3 show glossy and mirror materials, respectively, while row 4 shows the weight field, as in Fig. 5. The contribution of multibounce lighting can be seen, for instance, in the bunny's ear reflected off its back (row 2, column 3). See also Supp. Fig. 25 for "glowing" 3D shapes (i.e., with non-zero M_L), where the geometry itself can be a light source.

VI. Path Tracing with Intrinsic Appearance DDFs

A natural interpretation of a DDF is as the field of all possible depth images for a given scene. However, this extends to *any* ray (or camera) start point. In particular, this includes *inter-surface* rays: a DDF provides the "next" surface that the ray would intersect. Importantly, this corresponds to a fundamental operation when modelling the physics of *light*: surface inter-reflections. In other words, DDFs can also be viewed as modelling the set of all light paths through a scene, where each path segment corresponds to a DDF call. Thus, any light path is equivalent to recursively applying a DDF. Given this connection to light transport, we therefore explore how to integrate DDFs into a path tracing framework.

Such an approach confers several benefits. First, a major focus in computer vision has been the disentanglement of images into constituent components (e.g., [88], [89]): geometry (shading), intrinsic appearance (reflectance), and illumination (lighting). A DDF, combined with a reflectance and lighting model, naturally fits within this decomposition. Second, many shape representations are not as amenable to light transport. For instance, the standard NeRF [4] acts as a radiance-emitting "cloud of particles". As such, appearance remains entangled. In contrast, the interpretation of recursive DDF calls as inter-surface light transport is naturally applicable to path tracing.

As a proof of concept, we consider a scenario with simplified materials and lighting, to demonstrate path tracing of a DDF. (see Supp. §F for details). We augment a PDDF (ξ, d) with two additional models. The first is the *material appearance*, given by a Bidirectional Reflectance Distribution Function (BRDF) $f_B : \mathbb{R}^3 \times \mathbb{S}^2 \times \mathbb{S}^2 \to \mathbb{R}_+^{|\mathcal{C}|}$, where $\mathcal{C}$ is the colour channel set, and an importance sampler $\Psi : \mathbb{R}^3 \times \mathbb{S}^2 \to \mathbb{P}[\mathbb{S}^2]$, where $\mathbb{P}[\mathbb{S}^2]$ is the set of distributions on $\mathbb{S}^2$, which stochastically decides how light bounces from a surface. The second is the *lighting model*, which includes an emission field (i.e., glow), $M_L : \mathbb{R}^3 \times \mathbb{S}^2 \to \mathcal{C}$, and an environment light, $E_L : \mathbb{S}^2 \to \mathcal{C}$, which represents incoming radiance from a faraway source. The combined model, which we call the intrinsic appearance DDF (IADDF) is $(\xi, d, f_B, \Psi, M_L, E_L)$.

We integrate the IADDF into a simple path tracing algorithm. Briefly, for each pixel, we cast a ray into the scene, finding the scene point q and normal n via the DDF. Based on Ψ, we sample new outgoing directions from q, and use the DDF to find the next surface in that direction. These "bounces" are the segments of a light path, being built backwards from the eye to the light source. This continues until the ray misses any geometry (i.e., $\xi = 0$), returning E_L in that direction. The final pixel irradiance corresponds to this value (attenuated by the material f_B through the bounces), plus any emissions (from M_L) encountered along the way. All surface intersections and normals are computed from the DDF.

We display example renders in Fig. 14. As this is only a proof-of-concept of the forward rendering process, we implement (f_B, Ψ, M_L, E_L) as analytic functions, while (ξ, d) are fit to single shapes (see §V-A). We utilize simple glossy and mirror materials to showcase the accuracy of the DDF geometry (as accurate depths, visibilities, and normals are all necessary for visual plausibility). We see that not only are the "first-order" values accurate (i.e., those generated

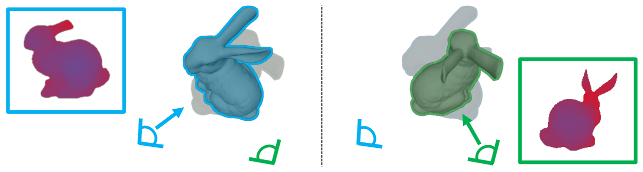

Figure 15: An illustration of a view-*in*consistent DDF. Due to its 5D nature, depth-rendering the same field from different viewpoints can produce incompatible geometries.

by a standard initial ray-casting through the image pixels), but the values computed at the surface (for a bounce) are sufficiently accurate for the tracing process to work. However, certain pathologies in the underlying PDDF geometry are apparent: the weight field seam that manifests as a "cut" (e.g., column one), difficulty with occlusion (e.g., "black spots" where the two spheres meet), and inaccuracies in the normals (e.g., waviness in the reflected "horizon" on the second-row spheres). Nevertheless, this appears to be a promising approach to combining disentangled appearance modelling with differentiable path tracing. We also emphasize that *internal structure modelling* (see also §V-A2), which is ignored by similar contemporary methods (e.g., [45], [48]), is essential – without it, inter-surface bounces cannot be computed.

VII. Theory: The Geometry of View Consistency

The defining characteristic of a DDF is its five-dimensional nature. While one naturally associates 3D shapes with 3D constructs, the computational advantages of DDFs stem fundamentally from the increase in dimensionality. However, the tradeoff in this case is the lack of theoretical guarantees of *view consistency* (VC). Specifically, unlike representations that are inherently 3D, DDFs require regularization to ensure predicted surface points do not vary with view direction. Thus, for a DDF implemented with a neural network, the question of how to enforce VC is of great practical importance. In particular, can we guarantee such an implementation is actually VC?

More precisely, consider an unconstrained differentiable 5D field. Assuming noisy initialization, before fitting, such a field will be akin to a cloud of noise, without sensible structure across viewpoints. Fig. 15 also provides an example of inconsistent geometry. Clearly, such entities do *not* represent any reasonable notion of a 3D shape. Under what conditions, then, can we say such a field *does* represent a 3D shape?

In this section, we answer these questions through a comprehensive theoretical investigation of (i) when a DDF is VC and (ii) show that this is conceptually equivalent to the DDF being an accurate representation of some 3D shape. We show that a straightforward set of constraints on the field, which can be checked locally (i.e., properties based on pointwise evaluations in 5D), are sufficient to guarantee consistency. Due to space constraints, we present a more complete story, with additional details and full proofs, in Supp. §G and §H.

A. Preliminaries and Notation

We begin by defining the basic setting, including the domain of our fields, the definition of a shape, and the oriented points that characterize our ray-based geometric representation.

Definition VII.1: Domain

Let $\mathcal{B} \subset \mathbb{R}^3$ be a convex and compact domain, where $\partial\mathcal{B}$ is smooth with outward-facing surface normals. Further, we define the 5D product space $\Gamma = \mathcal{B} \times \mathbb{S}^2$.

We primarily operate in the "interior" of $\mathcal{B}$, written $\mathcal{B}_\varepsilon$:

Definition VII.2: ε-Domain

For $\varepsilon > 0$, let $\mathcal{B}_\varepsilon$ be the ε-domain of $\mathcal{B}$, defined via

$$\mathcal{B}_\varepsilon = \left\{ p \in \mathcal{B} \;\middle|\; \min_{b\in\partial\mathcal{B}} ||p-b|| \geq \varepsilon \right\}. \tag{6}$$

As for $\mathcal{B}$, we define $\Gamma_\varepsilon = \mathcal{B}_\varepsilon \times \mathbb{S}^2$.

The role of $\mathcal{B}_\varepsilon$ is to enable us to define a notion of "away from the boundary" (i.e., $\mathcal{B}\setminus\mathcal{B}_\varepsilon$ is a thin ε-width shell around the edges of the domain). Our shapes will thus reside in $\mathcal{B}_\varepsilon \subset \mathbb{R}^3$, but our field will operate in 5D ray-space (i.e., Γ).

Definition VII.3: Oriented Points and Rays

We denote an *oriented point* via $\tau = (p, v) \in \Gamma$. Any oriented point induces a 3D *ray*: $r_\tau(t) = p + tv$, $t \geq 0$. In a slight abuse of notation, we may refer to the τ and r_τ forms of such 5D elements interchangeably.

Fundamentally, our interest is in representing 3D shapes, which are defined simply as follows.

Definition VII.4: Shapes

We define a *shape* to be a compact set $S \subset \mathcal{B}$.

Usually, we will be interested in shapes $S \subset \mathcal{B}_\varepsilon$, for $\varepsilon > 0$. Note that using $\mathcal{B}_\varepsilon$ does not strongly constrain the shape: for example, one can simply use the bounding sphere of the given point set, and then inflate it enough to satisfy the ε condition.

Next, for notational simplicity, we consider restrictions of functions to a fixed ray.

Definition VII.5: Along-Ray Functions

We define the "along-ray" form of a function $g : \Gamma \to \mathcal{X}$, which maps into a set $\mathcal{X}$, to be:

$$f_g(s \mid \tau) := g(p + sv, v) = g(r_\tau(s), v), \tag{7}$$

where $s \geq 0$ and $\tau = (p, v) \in \Gamma$.

Finally, we denote intersections of rays and point sets via:

Definition VII.6: Intersecting Rays and Point Sets

Consider $S \subseteq \mathcal{B}$ and $\tau \in \Gamma$.

- *Intersected Points.* Let $S_\tau \subseteq S$ denote the points in S intersected by r_τ (i.e., $q \in S_\tau$ iff $\exists\, t \geq 0$ such that $r_\tau(t) = q$). For clarity, we may write $[S]_\tau$ as well.
- *Intersecting Rays.* Let $\mathcal{I}_S \subseteq \Gamma$ be the set of rays that intersect S (i.e., $\tau \in \mathcal{I}_S$ if $\exists\, q \in S$ such that $r_\tau(s) = q$ for some $s \geq 0$). For $q \in \mathcal{B}$, we denote $\mathcal{I}_q := \mathcal{I}_{\{q\}}$.

B. View Consistency for Visibility Fields

First, we consider when a visibility field (VF) alone will be VC. We do so by defining an *un*constrained field, and then devising constraints that can ensure VC.

Definition VII.7: BOZ Field

Let $\xi : \Gamma \to \{0,1\}$. We restrict ξ to have an open zero set; i.e., $\forall\, \tau \in \Gamma$, if $\xi(\tau) = 0$, then $\exists\, \varepsilon > 0$ such that $\xi(\widetilde{\tau}) = 0 \;\forall\, \widetilde{\tau} \in B_\varepsilon(\tau) \cap \Gamma$, where $B_\varepsilon(\tau)$ is the open ball centered at τ of radius ε. We call any such binary field, with an open zero set, a BOZ field.

Without further conditions, a BOZ field does not have an obvious connection to 3D shapes or their multiview silhouettes (i.e., most BOZ fields do not represent a consistent 3D point set from all viewpoints). Hence, our goal is to understand *when* (or under what conditions) a BOZ field acts as a continuous representation of the silhouettes of some coherent shape (i.e., when it assigns every ray a binary indication as to whether a shape point exists along that ray or not). Thus, we next define three conditions that we will show impose consistency on ξ. We will find a close analogy between these constraints and the ones needed for the depth field within the DDF.

Define $\mathcal{O}[V,U] := \{\tau = (p,v) \in \Gamma \,|\, p \in V, r_\tau(s) \notin U \,\forall\, s \geq 0\}$ as the rays *starting* in V that *miss* U. Hence, $\mathcal{O}[\mathcal{B} \setminus \mathcal{B}_\varepsilon, \mathcal{B}_\varepsilon]$ are the "outward rays" from $\mathcal{B} \setminus \mathcal{B}_\varepsilon$.

Constraint Definition VII.1: BC$_\xi$

A BOZ field ξ satisfies the *Non-Visible Boundary Condition*, denoted BC$_\xi$, if $\xi(\tau) = 0 \;\forall\, \tau \in \mathcal{O}[\mathcal{B} \setminus \mathcal{B}_\varepsilon, \mathcal{B}_\varepsilon]$.

In words, BC$_\xi$ demands that any rays that (i) start close to the boundary and (ii) do not intersect the inner domain $\mathcal{B}_\varepsilon$ must be non-visible. Intuitively, if ξ were to represent a shape $S \subset \mathcal{B}_\varepsilon$, then BC$_\xi$ prevents S from being visible on rays that start outside of $\mathcal{B}_\varepsilon$ and "look away" from it.

Constraint Definition VII.2: DE$_\xi$

A BOZ field ξ satisfies the *Directed Eikonal Constraint* DE$_\xi$ if ξ is always non-increasing along a ray (i.e., $f_\xi(s_1|\tau) \leq f_\xi(s_2|\tau) \;\forall\, \tau \in \Gamma, s_1 > s_2$).

Intuitively, DE$_\xi$ demands that visibility can only ever go from "seeing" to "not seeing" along a ray (i.e., one should not suddenly see a new point in the shape become visible).

The last constraint requires characterizing field behaviour at specific special points. In particular, we are interested in $q \in \mathcal{B}$ where ξ "flips" along some direction $v \in \mathbb{S}^2$. So, let us first consider the field value as one approaches q along v:

$$\mathcal{F}_{\xi,v}[\mathrm{Op},\varsigma](q) = \lim_{\epsilon \downarrow 0} \operatorname*{Op}_{s \in (0,\epsilon)} \xi(q + \varsigma s v, v), \tag{8}$$

where $\mathrm{Op} \in \{\inf, \sup\}$ and $\varsigma \in \{-1,+1\}$. Let $\mathcal{F}_{\to,\xi,v}(q) := \mathcal{F}_{\xi,v}[\sup,-1](q)$ and $\mathcal{F}_{\leftarrow,\xi,v}(q) := \mathcal{F}_{\xi,v}[\inf,1](q)$. Intuitively, $\mathcal{F}_{\to,\xi,v}$ is the maximum value as one moves forward along a ray, while $\mathcal{F}_{\leftarrow,\xi,v}$ is the minimum as one moves backward. A "flip" therefore occurs when these two do not match at a point. In particular, a "one-to-zero flip" along v occurs when

$$\mathcal{C}_{\mathrm{Flip}}(q|\xi,v) := (\mathcal{F}_{\to,\xi,v}(q) = 1) \wedge (\mathcal{F}_{\leftarrow,\xi,v}(q) = 0)\,. \tag{9}$$

Our field constraint uses this concept as follows:

Constraint Definition VII.3: IO$_\xi$

A BOZ field ξ satisfies the *Isotropic Opaqueness Constraint* (IOC) at $q \in \mathcal{B}$ iff the following holds:

$$\big(\exists\, v \text{ s.t. } \mathcal{C}_{\mathrm{Flip}}(q|\xi,v)\big) \implies \big(\xi(\tau) = 1 \;\forall\, \tau \in \mathcal{I}_q\big),$$

where $\mathcal{I}_q$ is the set of rays in Γ that intersect q. An *IO field* (IO$_\xi$) satisfies the IOC everywhere.

In words, if there is *any* direction along which the field flips from one to zero, then the point at which that occurs must be "isotropically opaque" (i.e., from any direction, any ray that hits that point must produce a visibility value of one). Intuitively, IO$_\xi$ says that such "one-to-zero“ discontinuities in ξ are special. As we shall see, they act similar to surface points. In particular, for any point $q \in \mathcal{B}$, if even a single direction v exists such that a one-to-zero discontinuity occurs along v at q, then *any* ray that goes through q must also be visible.

With these three constraints (see Fig. 16 for an illustration), we can now define a subset of BOZ fields, called *visibility fields* (VFs), which are intimately tied to shapes.

Definition VII.8: Visibility Field (VF)

We define a Visibility Field to be a BOZ field such that IO$_\xi$, BC$_\xi$, and DE$_\xi$ are satisfied.

Conceptually, VFs (i) bound visible geometry within a finite domain, (ii) prevent new geometry from "appearing out-of-nowhere" (when it should have been visible earlier), and (iii) enforce that "flip positions" (Eq. 9) are visible from all viewpoints. Such discontinuities are effectively *surface points*, where visibility "flips" because one has moved *through* the surface. These special positions link the 5D field to a 3D shape.

Definition VII.9: Positional Discontinuities of VFs

The positional discontinuities of a VF ξ are defined by

$$Q_\xi := \mathscr{C}\left(\{q \in \mathcal{B} \mid \exists\, v \text{ s.t. } \mathcal{C}_{\mathrm{Flip}}(q|\xi,v)\}\right), \tag{10}$$

where $\mathscr{C}$ is the closure operator on sets.

Intuitively, given a VF ξ, the set Q_ξ simply contains all of the one-to-zero discontinuities in ξ. More specifically, if there exists some direction v such that the field ξ flips from one to zero (i.e., from "seeing something" to "seeing nothing") at $q \in \mathcal{B}$, then $q \in Q_\xi$. In Supp. Lemma G.2, we show that $Q_\xi \subseteq \mathcal{B}_\varepsilon$ is a shape. Hence, the discontinuities Q_ξ are a good candidate for a potential shape that ξ is implicitly representing. First, we need to define what it means for a shape S to *induce* a ξ-field (i.e., for ξ to "exactly" represent S).[4]

[4]See Supp. Def. G.17 for an explanation of notation at discontinuities.

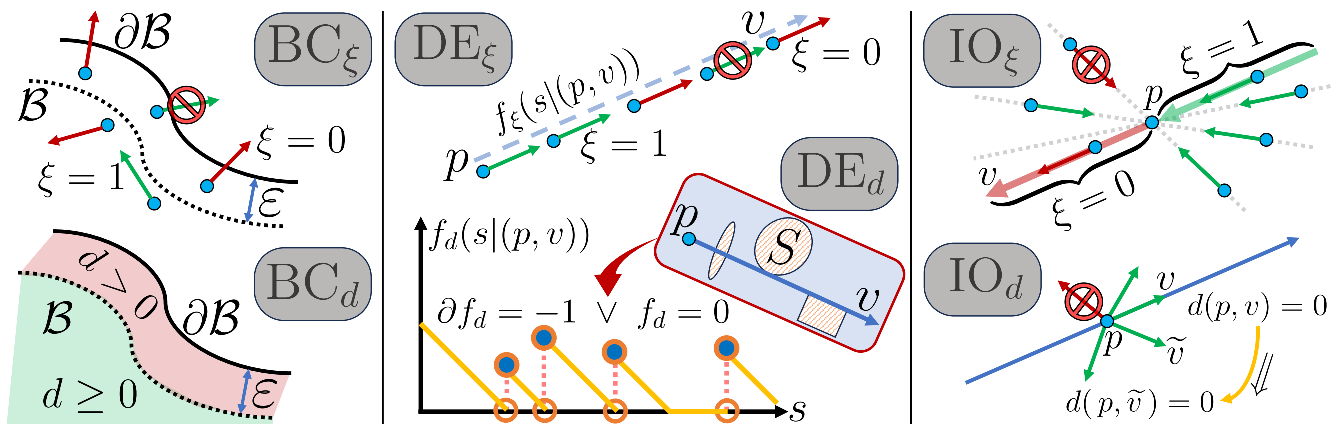

Figure 16: The fundamental constraints that link visibility and depth fields to being shape representations. The boundary conditions (BC) demand that d does not predict geometry too close to $\partial\mathcal{B}$ and that $\xi = 0$ whenever it looks "outward". For the directed Eikonal (DE) constraints, we ask that f_d decreases at unit rate and f_ξ never increases. Finally, isotropic opaqueness (IO) asks that zeroes in d are zeroes from all directions, while one-to-zero flips in ξ must be visible from all directions.

Definition VII.10: Shape-Induced Binary Fields

A BOZ field ξ is *induced* by a shape $S \subseteq \mathcal{B}_\varepsilon$ iff

$$\xi(\tau) = 1 \iff \tau \in \mathcal{I}_S. \tag{11}$$

where $\mathcal{I}_S \subseteq \Gamma$ is the set of rays that intersect S. Further, given a shape S, its induced field ξ is unique.

Notice that the induction relation is an "iff", meaning $\xi(\tau) = 0$ implies that *no* $q \in S$ can be present along the ray r_τ. In addition, importantly, while the S-induced field ξ is unique, the inducing shape for a given ξ is *not* necessarily unique. For instance, imagine a sphere with another shape (say, another sphere) inside it – such a shape will induce the same VF as using the outermost sphere alone for the inducement. Given this definition, our first result in this section shows that any shape-induced field satisfies the constraints of a VF.

Theorem VII.1: Induced Binary Fields Are VFs

Let $S \subseteq \mathcal{B}_\varepsilon$ be a shape, with an induced field ξ. Then ξ is a Vf (i.e., IO_ξ, BC_ξ, and DE_ξ hold). Also, the positional discontinuities of ξ satisfy $Q_\xi \subseteq S$.

So far, we have shown that any shape can be used to generate or induce a VF (in the sense of Def. VII.8). This defines a notion of "shape representation" for such fields.

Definition VII.11: Shape Indicators as VC VFs

A VF ξ is *View Consistent* (VC) iff there exists a shape S such that S induces ξ. We can therefore call any such ξ a *shape indicator* for the point set S.

Note we call it a shape indicator instead of a representation because we cannot necessarily reconstruct S from ξ (i.e., it would be an incomplete representation). Instead, we call it an indicator, because it "indicates" whether or not a shape exists along a given ray (i.e., acts as a per-ray indicator function). We next show that the local constraints of a visibility field are sufficient for it to be a representation of some shape.

Theorem VII.2: Every VF is a Shape Indicator

Let ξ be a visibility field. Then there exists a shape S such that ξ is induced by S. Further, Q_ξ induces ξ.

So far, we have shown a close duality between visibility fields (VFs) and shape indicators (i.e., between binary fields with specific local constraints on the field, namely IO_ξ, DE_ξ, and BC_ξ, and binary fields constructed from a given point set). However, it is clear that many inducing point sets can induce the same field. We can therefore ask for a "minimal example" over an equivalence class of inducing point sets (i.e., given many shapes S that all induce the same ξ, which is the "simplest"?), which we answer in the following corollary.

Corollary VII.1: Minimal Characterization of VFs

Let $\mathfrak{S}[\xi]$ be the equivalence class of ξ-inducing shapes (i.e., $\mathfrak{S}[\xi] = \{S \mid S \subseteq \mathcal{B} \text{ induces } \xi\}$). Then, $Q_\xi \in \mathfrak{S}[\xi]$ is the *smallest* closed point set among all such inducers.

Hence, Q_ξ is special among shapes that induce ξ, in that it forms a subset of any other shape that also induces ξ. As an aside, Supp. §G-C provides another interpretation. For any ξ-inducing S, the subset of S that is "directly visible" from $\partial\mathcal{B}$ is equivalent to Q_ξ. In other words, to get the minimal inducing points from S, we need not construct ξ; instead, we can "look inwards" from $\partial\mathcal{B}$ and find the "observable" points.

In summary, (i) "inducement" defines how a shape S gives rise to a VF ξ, which acts as its indicator along rays; (ii) the constraints that define a VF ξ (i.e., IO_ξ, DE_ξ, BC_ξ) are sufficient to guarantee that *some* point set must exist that induces ξ (i.e., every VF is the indicator of some shape); and (iii) the positional discontinuities Q_ξ are the *minimal* closed subset of S that induces a given ξ. Note, in Supp. §G-C1, we survey work in shape-from-silhouette (e.g., [90]–[92]), visual hulls (e.g., [93]), and space carving [94].

Nevertheless, questions still arise regarding VFs and their representational capacity. One aspect of shapes "missed" by VFs is *internal structure*: shape points that are completely surrounded by other shape points cannot be preserved by the field. In other words, given a shape S with internal structure, any S-induced field will not be able to recover such structure (hence the existence of $\mathfrak{S}[\xi]$). One can see this by noting that Q_ξ will not include such internal structure. These points, however, fully encode the *field* (as they induce it). Combining a VF with a distance field, as we will do next, enables a complete shape representation, including internal structure.

C. View Consistency for Directed Distance Fields

We now move on to the theory of DDFs, where we will combine a visibility field (VF) with a depth field. Analogous to the previous section, we define view consistency (VC) through a notion of "shape representation"; i.e., whether or not some shape (point set) exists that induces the DDF. In this case, the VF will be used to control where (in Γ) the distance field needs to be constrained.[5] Specifically, starting from a VF ξ (with its associated constraints), on visible rays ($\xi = 1$), we derive constraints on a depth field d that are analogous to those of the VF. With two additional constraints that ensure the compatibility between ξ and d, we then show the resulting DDF (ξ, d) is a VC shape representation.

Analogous to the BOZ field (Def. VII.7), we start with the following definition of a general non-negative field, which is generally not sufficiently constrained to be represent a shape.

Definition VII.12: NNBC Field

An *NNBC field* on Γ is a non-negative, bounded scalar field that is piece-wise continuously differentiable along rays, written $d : \Gamma \to \mathbb{R}_{\geq 0}$.

In general, NNBC fields need not have an obvious connection to any shapes. We treat such fields as *putative* representations for shapes – the goal is to understand the conditions under which such fields are "equivalent to" some shape. Such constrained fields should generate view-consistent depths.

Constraint Definition VII.4: BC$_d$

An NNBC field d satisfies the *Positive Boundary Condition* BC$_d$ iff $\inf_{v\in\mathbb{S}^2} d(p,v) > 0 \;\forall\, p \in \mathcal{B} \setminus \mathcal{B}_\varepsilon$.

Recall that $\mathcal{B} \setminus \mathcal{B}_\varepsilon$ is the "outer shell" of $\mathcal{B}$. In other words, BC$_d$ demands that d cannot have any zeroes close to $\partial\mathcal{B}$.

Next, we denote the infimum approached by d along v via

$$\mathcal{A}_d(q,v) = \lim_{\epsilon\downarrow 0} \inf_{s\in(0,\epsilon)} d(q - sv, v). \tag{12}$$

Constraint Definition VII.5: DE$_d$

An NNBC field d satisfies the *Directed Eikonal Constraint* DE$_d$ if $\partial_s f_d(s|\tau) = -1$, except at along-ray zeroes (q,v), such that $\mathcal{A}_d(q,v) = 0$.

The second constraint is on the derivative of d: it says that, along any ray, d must decrease linearly, at unit rate, unless $d = 0$ (a value at which it may potentially stay). Intuitively, d must act like a distance function, along any ray.

Next, we denote the inf approached over *all* v's via

$$\mathcal{A}_d(q) = \lim_{\epsilon\downarrow 0} \inf_{\substack{s\in(0,\epsilon)\\ v\in\mathbb{S}^2}} d(q - sv, v). \tag{13}$$

Constraint Definition VII.6: IO$_d$

An NNBC field d satisfies the *Isotropic Opaqueness Constraint* at a point $q \in \mathcal{B}$ if

$$(\mathcal{A}_d(q) = 0) \implies \left(\mathcal{A}_d(q,v) = 0 \;\forall\, v \in \mathbb{S}^2\right). \tag{14}$$

We say d is isotropically opaque, denoted IO$_d$, if it satisfies the Isotropic Opaqueness Constraint $\forall\, q \in \mathcal{B}$.

Notice that IO$_d$ is *stronger* than (i.e., it implies) the following constraint: for any $(q,v) \in \Gamma$,

$$(\exists\, v \mid \mathcal{A}_d(q,v) = 0) \implies (\mathcal{A}_d(q,v) = 0 \;\forall\, v). \tag{15}$$

Namely, if d approaches zero at some position q from *any* direction v, then it must approach zero at q from *all* directions.

Together, these field constraints enable us to define a *simple DDF*, which we show in Supp. G-B satisfies certain theoretical consistency guarantees *without* the need for a VF.

Definition VII.13: Simple DDF

We define a simple DDF to be an NNBC field such that IO$_d$, BC$_d$, and DE$_d$ are satisfied.

Notice the close analogy between the constraints for simple DDFs and VFs. Fig. 16 illustrates all six fundamental constraints. While we do not dwell on simple DDFs alone, they enable us to define the following special set of points.

Definition VII.14: Positional Zeroes of Simple DDFs

We denote the *positional zeroes* of a Simple DDF as

$$Q_d = \{p \in \mathcal{B} \mid \mathcal{A}_d(q) = 0\}. \tag{16}$$

For simple DDFs alone, Q_d defines the shape represented by the field in a formal sense (see Supp. Thm. G.2). However, the positional zeroes will be fundamental to full DDFs as well.

Notice that the simple DDF constraints do not involve a VF; hence, we next begin to link them. This actually relaxes some requirements on d, as only *visible* rays are constrained.

Definition VII.15: Visible and Non-Visible Ray Sets

Let ξ be a visibility field. We define the sets of visible and non-visible rays, respectively, as

$$\mathcal{R}_V[\xi] = \{\tau \in \Gamma \mid \xi(\tau) = 1\} \tag{17}$$
$$\mathcal{R}_N[\xi] = \{\tau \in \Gamma \mid \xi(\tau) = 0\}. \tag{18}$$

We next limit the simple DDF constraints to $\mathcal{R}_V[\xi]$. Define $\mathcal{A}_{d,\xi}(q)$ as the *visible* infimum of d at q:

$$\mathcal{A}_{d,\xi}(q) = \lim_{\epsilon\downarrow 0} \inf_{\substack{s\in(0,\epsilon)\\ v\in\mathbb{S}^2 \,|\, \xi(q,v)=1}} d(q - sv, v).$$

[5]Note: in Supp. §G-B, we also discuss a way to define distance fields without a VF, which can be constrained similarly.

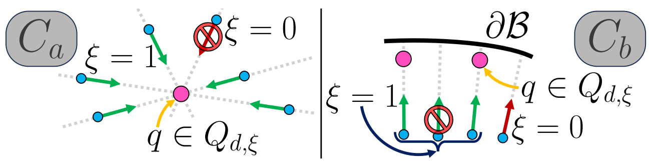

Figure 17: Illustration of inter-field compatibility (see Con. Def. VII.7), which ensures rays are visible iff they intersect the LVDZs (Def. VII.17). See Fig. 16 for intra-field constraints.

Definition VII.16: ξ-Coherent Simple DDFs

Given a VF ξ, an NNBC field d is a ξ-coherent simple DDF iff it satisfies

(IO$_{d,\xi}$) *Isotropic opaqueness on visible rays.*

$$\forall\,(q,v) \in \mathcal{R}_V[\xi] : \mathcal{A}_{d,\xi}(q) = 0 \implies \mathcal{A}_d(q,v) = 0.$$

(BC$_{d,\xi}$) *Positive boundary condition on visible rays.*

$$\big((p,v) \in \mathcal{R}_V[\xi] \,\wedge\, p \in \mathcal{B} \setminus \mathcal{B}_\varepsilon\big) \implies d(p,v) > 0.$$

(DE$_{d,\xi}$) *Directed Eikonal constraint on visible rays.*

$$(\tau \in \mathcal{R}_V[\xi] \,\wedge\, d(\tau) > 0) \implies \partial_s f_d(s \mid \tau)|_{s=0} = -1.$$

In words, a ξ-coherent depth field d satisfies the IO, BC, and DE constraints whenever $\xi = 1$ (i.e., it is a simple DDF on $\mathcal{R}_V[\xi]$). Outside of the visible rays $\mathcal{R}_V[\xi]$, d is essentially unconstrained. These modified constraints allow us to define the following modification of the positional zeroes.

Definition VII.17: Locally Visible Depth Zeroes

Let ξ be a visibility field and d be ξ-coherent. Then the *locally visible depth zeroes* (LVDZs) are given by

$$Q_{d,\xi} = \mathscr{C}\left(\{\widetilde{q} \in Q_d \mid \exists\, v \text{ s.t. } \xi(\widetilde{q}, v) = 1\}\right), \tag{19}$$

where $\mathscr{C}$ is the closure operator.

The LVDZs thus include any depth zero (i.e., $q \in Q_d$) that is "locally visible" along some direction.

We now have three "fundamental point sets" associated to a DDF, illustrated in Figs. 18 and 19. The LVDZs ($Q_{d,\xi}$) are analogous to the zeroes Q_d for simple DDFs and the discontinuities Q_ξ for VFs, defining *which shape* is represented.

First, note that ξ-coherence merely enforces d to be "internally" consistent (i.e., a simple DDF along visible rays). Hence, it is *not* sufficient to ensure (ξ, d) actually represents a shape. Two additional constraints are needed, which ensure ξ and d are compatible (see Fig. 17). Both rely on the LVDZs.

Constraint Definition VII.7: Compatibility

A visibility field ξ and simple DDF d are *compatible* iff (a) LVDZs are isotropically opaque with respect to ξ:

$$q \in Q_{d,\xi} \implies \tau \in \mathcal{R}_V[\xi]\ \forall\, \tau \in \mathcal{I}_q. \tag{20}$$

(b) Every visible ray must hit an LVDZ:

$$\tau \in \mathcal{R}_V[\xi] \implies [\mathcal{B}]_\tau \cap Q_{d,\xi} \neq \varnothing. \tag{21}$$

This is finally enough to define a full DDF, which we will show is a VC shape representation.

Definition VII.18: Directed Distance Field (DDF)

A BOZ field ξ and an NNBC field d form a *Directed Distance Field* $F = (\xi, d)$ iff:

1) ξ is a visibility field.
2) d is a ξ-coherent Simple DDF.
3) ξ and d are compatible.

The definition of a DDF can be intuited as combining a VF with a simple DDF that is constrained only along visible rays, along with two additional restrictions that force the two to be consistent. The latter constraints hinge on the "locally visible" zeroes $q \in Q_{d,\xi}$ (i.e., at least one ray exists, along which q is visible at an infinitesimal distance). Such zeroes are (i) always visible and (ii) every visible ray must intersect at least one.

Following the previous section on VFs, we next define an approach to "field induction": given a shape S, how can we construct a DDF, consisting of a "field pair" (ξ, d), that appropriately represents S? We then show that inducing a field in this manner, versus having a DDF with constraints on local field behaviour (i.e., Def. VII.18) are equivalent.

Definition VII.19: Induced Field Pair

Consider a shape $S \subseteq \mathcal{B}_\varepsilon$. By Def. VII.10, S induces a unique visibility field, ξ. We also define an induced distance field d to be any NNBC field that satisfies

$$d(\tau) = \min_{q \in [S]_\tau} ||q - p||\ \forall\, \tau \in \mathcal{R}_V[\xi]. \tag{22}$$

Then, any such pair (ξ, d) is an *S-induced field pair*.

This builds on the notion of shape-induced VFs (Def. VII.10). Recall, the induced visibility $\xi(\tau)$ is one iff τ intersects S. Hence, for a visible ray $\tau \in \mathcal{R}_V[\xi]$, d is constrained to always predict the distance to the closest $q \in S_\tau$. Thus, Eq. 22 simply follows the definition of a ray-based depth field from §III. Again, similar to Theorem VII.1, when a shape S induces a field pair F, we find that F follows the field requirements of DDFs (Def. VII.18).

Theorem VII.3: Induced Field Pairs are DDFs

Let S be a shape and $F = (\xi, d)$ be an S-induced field pair. Then F is a DDF that satisfies $Q_{d,\xi} = S$.

Thus, any field generated from a shape is a DDF. We next want to show the converse: any DDF is a VC representation of some shape. Intuitively, this would imply our field constraints (which do not refer to a predefined shape) are equivalent to deriving the field from a shape. First, we require a notion of equivalence between fields that ignores non-visible rays.

Definition VII.20: DDF Equivalence

Consider two DDFs, $F_1 = (\xi_1, d_1)$ and $F_2 = (\xi_2, d_2)$. Then F_1 and F_2 are equivalent iff (i) $\xi_1(\tau) = \xi_2(\tau)\ \forall\, \tau \in \Gamma$ and (ii) $d_1(\tau) = d_2(\tau)\ \forall\, \tau \in \mathcal{R}_V[\xi_1]$.

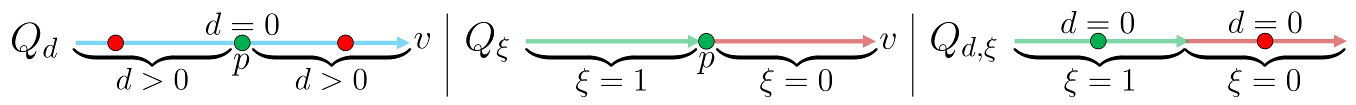

Figure 18: Illustration of the fundamental point sets of a DDF. The positional depth zeroes, Q_d (Def. VII.14), occur when $d(\tau) = 0$ for some $\tau = (p, v)$. The positional visibility discontinuities, Q_ξ (Def. VII.9), demarcate where the field flips from one ("seeing something") to zero ("seeing nothing"). The locally visible zeroes, $Q_{d,\xi}$ (Def. VII.17), mark positions that (i) are zeroes of d (i.e., in Q_d) and (ii) are "visible" along some direction, at an infinitesimal distance. See also Fig. 19.

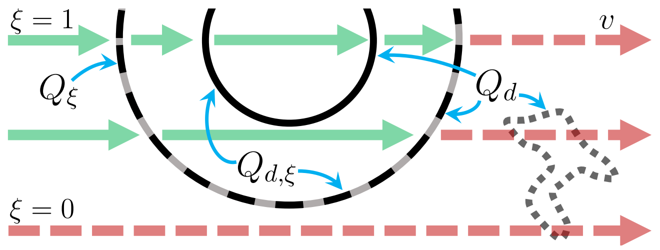

Figure 19: Example of the fundamental DDF point sets. The VF is shown with green/solid ($\xi = 1$) and red/dashed ($\xi = 0$) arrows. Additional *non*-visible points are on the right (grey dots). All such points are part of Q_d; however, Q_ξ includes only the outer circle, where ξ flips. In contrast, the LVDZs, $Q_{d,\xi}$, encompass *both* the inner and outer circle. For DDFs, $Q_\xi \subseteq Q_{d,\xi} \subseteq Q_d$ (see Supp. Lemma G.3).

This enables defining when a DDF is a shape representation.

Definition VII.21: DDFs as Shape Representations

A DDF F is view consistent (VC) iff it is equivalent to a DDF that has been induced by a shape S. I.e., F is VC iff $\exists$ S s.t. F is an S-induced field pair. In such a case, we say that F *is a shape representation for* S.

Finally, analogously to Theorem VII.2, we can assert that the field constraints of Def. VII.18 are sufficient to guarantee that a DDF represents some shape.

Theorem VII.4: Every DDF is View Consistent

Let $F = (\xi, d)$ be a DDF (Def. VII.18). Then F must be VC (i.e., $\exists$ a shape S such that F is an S-induced field pair). Further, $Q_{d,\xi}$ is a shape that induces F.

This theorem finally links *full* DDFs and view consistency. A DDF can therefore represent a shape in two equivalent ways: (i) beginning with a point set and inducing a field pair from it, or (ii) starting from a visibility field and a distance field, and enforcing per-field requirements (BC, IO, DE), as well as compatibility. We reiterate that Supp. §G has a more complete exploration of these results, with proofs in Supp. §H.

The nature of the field constraints suggests an important use case for DDFs in practice. Since field properties can be checked in a point-wise manner (e.g., DE$_d$ asks for along-ray derivatives to be -1), they are amenable to use as differentiable regularization losses. This is similar to the Eikonal loss used in SDFs (e.g., [60]–[64]), for example; indeed, resolving these constraints for DDFs is reminiscent of solving a 5D partial differential equation, where BC and IO act like boundary conditions, while DE needs to be solved across space. Advances in optimization of implicit fields with differential constraints should make implementation of DDFs more viable. In general, when designing a neural architecture for a DDF, one can also check which theoretical properties are guaranteed by the design and which may be violated. Hence, we expect that our theoretical results will be useful in the construction and analysis of any 5D geometry field, where view consistency needs to be enforced. In this work, we trained directly with 3D data, mitigating consistency issues; however, deriving fields from less constrained data, such as images, will require addressing such problems.

VIII. Discussion

We have devised *directed distance fields* (DDFs), a novel shape representation that maps oriented points to depth and visibility values, and a probabilistic extension for handling discontinuities. In contrast to NeRFs or U/SDFs, depth requires a single per-pixel forward pass, while normals use one further backward pass. Efficient differentiable normals rendering is thus much easier for DDFs than for for voxels, NeRFs, or occupancy fields. Unlike meshes, DDFs are topologically unconstrained, and rendering is independent of shape complexity.

We investigated a number of properties and use-cases for DDFs, including 3D data extraction, fitting shapes, UDF extraction, composition, generative modelling, and single-image 3D reconstruction. While DDFs form a continuous field of depth images, recursive calls can be interpreted as tracing inter-surface light paths. Finally, we examined the theory of *view consistency* (VC) for DDFs, as their 5D nature permits view-dependent geometric inconsistencies. We showed that a small set of field constraints guarantee a given DDF is VC, and hence a proper representation for some 3D shape.

For future work, using DDFs for inverse graphics (e.g., for multiview stereo or novel view synthesis) seems promising, particularly via constraints based on our VC results, as well as handling volumetricity (not just surfaces), light transport, and translucency. Finally, we hope to apply DDFs to other areas, such as virtual reality and visuotactile perception, where collision modelling and geometric rendering are important.

Acknowledgments

We are grateful for support from NSERC (CGSD3-534955-2019) and Samsung Research.

References

[1] L. G. Roberts, "Machine perception of three-dimensional solids," Ph.D. dissertation, Massachusetts Institute of Technology, 1963. 1

[2] B. G. Baumgart, "Geometric modeling for computer vision." Ph.D. dissertation, Stanford University, 1974. 1
[3] A. Yuille and D. Kersten, "Vision as Bayesian inference: Analysis by synthesis?" *Trends in Cognitive Sciences*, 2006. 1
[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020. 1, 3, 4, 10
[5] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3D reconstruction in function space," in *CVPR*, 2019. 1, 3
[6] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "DeepSDF: Learning continuous signed distance functions for shape representation," in *CVPR*, 2019. 1, 3, 7
[7] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and Z. Cui, "DIST: Rendering deep implicit signed distance function with differentiable sphere tracing," in *CVPR*, 2020. 1, 3, 4, 7
[8] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision," in *CVPR*, 2020. 1, 3
[9] Y. Jiang, D. Ji, Z. Han, and M. Zwicker, "SDFDiff: Differentiable rendering of signed distance fields for 3D shape optimization," in *CVPR*, 2020. 1, 3
[10] D. Vicini, S. Speierer, and W. Jakob, "Differentiable signed distance function rendering," *ACM TOG*, 2022. 1
[11] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi, T. Simon, C. Theobalt, M. Nießner, J. T. Barron, G. Wetzstein, M. Zollhöfer, and V. Golyanik, "Advances in neural rendering," *Computer Graphics Forum*, 2022. 1
[12] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, "Zero-1-to-3: Zero-shot one image to 3D object," *ICCV*, 2023. 1
[13] A. Bhattad, D. McKee, D. Hoiem, and D. Forsyth, "StyleGAN knows normal, depth, albedo, and more," *NeurIPS*, 2024. 1
[14] A. Tewari, T. Yin, G. Cazenavette, S. Rezchikov, J. Tenenbaum, F. Durand, B. Freeman, and V. Sitzmann, "Diffusion with forward models: Solving stochastic inverse problems without direct supervision," *NeurIPS*, 2024. 1
[15] N. Müller, Y. Siddiqui, L. Porzi, S. R. Bulo, P. Kontschieder, and M. Nießner, "DiffRF: Rendering-guided 3D radiance field diffusion," in *CVPR*, 2023. 1
[16] H. Wang, X. Du, J. Li, R. A. Yeh, and G. Shakhnarovich, "Score Jacobian chaining: Lifting pretrained 2D diffusion models for 3D generation," in *CVPR*, 2023. 1
[17] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "DreamFusion: Text-to-3D using 2D diffusion," *arXiv:2209.14988*, 2022. 1
[18] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia, "Deep mesh reconstruction from single RGB images via topology modification networks," in *ICCV*, 2019. 1
[19] J. Tang, X. Han, J. Pan, K. Jia, and X. Tong, "A skeleton-bridged deep learning approach for generating meshes of complex topologies from single RGB images," in *CVPR*, 2019. 1
[20] H. Kato, Y. Ushiku, and T. Harada, "Neural 3D mesh renderer," in *CVPR*, 2018. 1, 9
[21] S. Liu, T. Li, W. Chen, and H. Li, "Soft rasterizer: A differentiable renderer for image-based 3D reasoning," in *ICCV*, 2019. 1, 4, 6
[22] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3D-structure-aware neural scene representations," in *NeurIPS*, 2019. 1, 3, 4
[23] T. Aumentado-Armstrong, S. Tsogkas, S. Dickinson, and A. D. Jepson, "Representing 3D shapes with probabilistic directed distance fields," in *CVPR*, 2022. 2, 3
[24] A. Rosenfeld and J. L. Pfaltz, "Distance functions on digital pictures," *Pattern recognition*, vol. 1, no. 1, pp. 33–61, 1968. 3
[25] J. Chibane, A. Mir, and G. Pons-Moll, "Neural unsigned distance fields for implicit function learning," *NeurIPS*, 2020. 3, 7
[26] R. Venkatesh, S. Sharma, A. Ghosh, L. Jeni, and M. Singh, "DUDE: Deep unsigned distance embeddings for hi-fidelity representation of complex 3D surfaces," *arXiv:2011.02570*, 2020. 3, 7
[27] R. Cai, G. Yang, H. Averbuch-Elor, Z. Hao, S. Belongie, N. Snavely, and B. Hariharan, "Learning gradient fields for shape generation," in *ECCV*, 2020. 3
[28] R. Venkatesh, T. Karmali, S. Sharma, A. Ghosh, R. V. Babu, L. A. Jeni, and M. Singh, "Deep implicit surface point prediction networks," in *ICCV*, 2021. 3
[29] S. Liu, S. Saito, W. Chen, and H. Li, "Learning to infer implicit surfaces without 3D supervision," in *NeurIPS*, 2019. 3
[30] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler, "Neural geometric level of detail: Real-time rendering with implicit 3D shapes," in *CVPR*, 2021. 3
[31] J. Zhang, Y. Yao, and L. Quan, "Learning signed distance field for multi-view surface reconstruction," in *ICCV*, 2021. 3
[32] P. Kellnhofer, L. Jebe, A. Jones, R. Spicer, K. Pulli, and G. Wetzstein, "Neural lumigraph rendering," in *CVPR*, 2021. 3
[33] M. Oechsle, S. Peng, and A. Geiger, "UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction," *ICCV*, 2021. 3
[34] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, "Volume rendering of neural implicit surfaces," *arXiv:2106.12052*, 2021. 3
[35] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *NeurIPS*, 2021. 3
[36] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Zip-NeRF: Anti-aliased grid-based neural radiance fields," *arXiv:2304.06706*, 2023. 3
[37] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, "Fast-NeRF: High-fidelity neural rendering at 200fps," *arXiv:2103.10380*, 2021. 3
[38] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," *ICCV*, 2021. 3
[39] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "KiloNeRF: Speeding up neural radiance fields with thousands of tiny MLPs," *ICCV*, 2021. 3
[40] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," *ICCV*, 2021. 3
[41] T. Aumentado-Armstrong, A. Mirzaei, M. A. Brubaker, J. Kelly, A. Levinshtein, K. G. Derpanis, and I. Gilitschenski, "Reconstructive latent-space neural radiance fields for efficient 3D scene representations," *arXiv:2310.17880*, 2023. 3
[42] Z. Wan, C. Richardt, A. Božič, C. Li, V. Rengarajan, S. Nam, X. Xiang, T. Li, B. Zhu, R. Ranjan *et al.*, "Learning neural duplex radiance fields for real-time view synthesis," in *CVPR*, 2023. 3
[43] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D gaussian splatting for real-time radiance field rendering," *ACM TOG*, 2023. 3
[44] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron, "NeRFactor: Neural factorization of shape and reflectance under an unknown illumination," *ACM TOG*, 2021. 3
[45] V. Sitzmann, S. Rezchikov, B. Freeman, J. Tenenbaum, and F. Durand, "Light field networks: Neural scene representations with single-evaluation rendering," *NeurIPS*, 2021. 3, 6, 11
[46] Z. Li, L. Song, C. Liu, J. Yuan, and Y. Xu, "NeuLF: Efficient novel view synthesis with neural 4D light field," *arXiv:2105.07112*, 2021. 3
[47] B. Attal, J.-B. Huang, M. Zollhöfer, J. Kopf, and C. Kim, "Learning neural light fields with ray-space embedding," in *CVPR*, 2022. 3
[48] E. Zobeidi and N. Atanasov, "A deep signed directional distance function for object shape representation," *arXiv:2107.11024*, 2021. 3, 6, 11
[49] T. Houchens, C.-Y. Lu, S. Duggal, R. Fu, and S. Sridhar, "NeuralODF: Learning omnidirectional distance fields for 3D shape representation," *arXiv:2206.05837*, 2022. 3
[50] B. Y. Feng, Y. Zhang, D. Tang, R. Du, and A. Varshney, "PRIF: Primary ray-based implicit function," in *ECCV*, 2022. 3
[51] X. Yang, G. Lin, Z. Chen, and L. Zhou, "Neural vector fields: Implicit representation by explicit learning," in *CVPR*, 2023. 3
[52] N. Kulkarni, J. Johnson, and D. F. Fouhey, "Directed ray distance functions for 3D scene reconstruction," in *ECCV*, 2022. 3
[53] J. Xi, Y. Shi, Y. Wang, Y. Guo, and K. Xu, "RayMVSNet: Learning ray-based 1D implicit fields for accurate multi-view stereo," *CVPR*, 2022. 3
[54] C. Zhang, Y. Di, R. Zhang, G. Zhai, F. Manhardt, F. Tombari, and X. Ji, "DDF-HO: Hand-held object reconstruction via conditional directed distance field," *NeurIPS*, 2024. 3
[55] T. Yenamandra, A. Tewari, N. Yang, F. Bernard, C. Theobalt, and D. Cremers, "FIRe: Fast inverse rendering using directional and signed distance functions," in *WACV*, 2024. 3
[56] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee, "Perspective transformer nets: Learning single-view 3D object reconstruction without 3D supervision," *NeurIPS*, 2016. 4
[57] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum, "MarrNet: 3D shape reconstruction via 2.5D sketches," *NeurIPS*, 2017. 4
[58] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik, "Multi-view supervision for single-view reconstruction via differentiable ray consistency," in *CVPR*, 2017. 4, 9

[59] T. H. Nguyen-Phuoc, C. Li, S. Balaban, and Y. Yang, "RenderNet: A deep convolutional network for differentiable rendering from 3D shapes," in *NeurIPS*, 2018. 4
[60] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," *ICML*, 2020. 5, 16
[61] C.-H. Lin, C. Wang, and S. Lucey, "SDF-SRN: Learning signed distance 3D object reconstruction from static images," *NeurIPS*, 2020. 5, 16
[62] M. Yang, Y. Wen, W. Chen, Y. Chen, and K. Jia, "Deep optimized priors for 3D shape modeling and reconstruction," in *CVPR*, 2021. 5, 16
[63] S. P. Bangaru, M. Gharbi, F. Luan, T.-M. Li, K. Sunkavalli, M. Hasan, S. Bi, Z. Xu, G. Bernstein, and F. Durand, "Differentiable rendering of neural sdfs through reparameterization," in *SIGGRAPH Asia Conference Papers*, 2022. 5, 16
[64] Z. Yu, S. Peng, M. Niemeyer, T. Sattler, and A. Geiger, "MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction," *NeurIPS*, 2022. 5, 16
[65] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *NeurIPS*, 2020. 6, 7
[66] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, "pi-GAN: Periodic implicit generative adversarial networks for 3D-aware image synthesis," in *CVPR*, 2021. 6
[67] K. Jo, G. Shim, S. Jung, S. Yang, and J. Choo, "CG-NeRF: Conditional generative neural radiance fields," *arXiv:2112.03517*, 2021. 6
[68] J. Gao, W. Chen, T. Xiang, A. Jacobson, M. McGuire, and S. Fidler, "Learning deformable tetrahedral meshes for 3D reconstruction," *NeurIPS*, 2020. 6
[69] Y.-T. Liu, L. Wang, J. Yang, W. Chen, X. Meng, B. Yang, and L. Gao, "NeUDF: Leaning neural unsigned distance fields with volume rendering," in *CVPR*, 2023. 7
[70] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2Mesh: Generating 3D mesh models from single RGB images," *ECCV*, 2018. 8
[71] N. Wang, Y. Zhang, Z. Li, Y. Fu, H. Yu, W. Liu, X. Xue, and Y.-G. Jiang, "Pixel2Mesh: 3D mesh model generation via image guided deformation," *IEEE PAMI*, 2020. 8
[72] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3D-R2N2: A unified approach for single and multi-view 3D object reconstruction," in *ECCV*, 2016. 8
[73] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 8
[74] I. Mehta, M. Gharbi, C. Barnes, E. Shechtman, R. Ramamoorthi, and M. Chandraker, "Modulated periodic activations for generalizable local functional representations," in *ICCV*, 2021. 8
[75] T. Aumentado-Armstrong, A. Levinshtein, S. Tsogkas, K. G. Derpanis, and A. D. Jepson, "Cycle-consistent generative rendering for 2D-3D modality translation," *3DV*, 2020. 9
[76] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," *NeurIPS*, 2017. 9
[77] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," *arXiv:1512.03012*, 2015, https://shapenet.org/. 8
[78] E. Insafutdinov and A. Dosovitskiy, "Unsupervised learning of shape and pose with differentiable point clouds," *NeurIPS*, 2018. 9
[79] D. Shin, C. C. Fowlkes, and D. Hoiem, "Pixels, voxels, and views: A study of shape representations for single view 3D object shape prediction," in *CVPR*, 2018. 9
[80] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox, "What do single-view 3D reconstruction networks learn?" *CVPR*, 2019. 9
[81] B. Kaya and R. Timofte, "Self-supervised 2D image to 3D shape translation with disentangled representations," in *3DV*, 2020. 9
[82] Y. Miyauchi, Y. Sugano, and Y. Matsushita, "Shape-conditioned image generation by learning latent appearance representation from unpaired data," in *ACCV*, 2018. 9
[83] J.-Y. Zhu, Z. Zhang, C. Zhang, J. Wu, A. Torralba, J. Tenenbaum, and B. Freeman, "Visual object networks: Image generation with disentangled 3D representations," in *NeurIPS*, 2018. 9
[84] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *ICLR*, 2014. 9
[85] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NeurIPS*, 2014. 9
[86] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," *CVPR*, 2017. 9
[87] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *ICCV*, 2017. 9
[88] H. Barrow, J. Tenenbaum, A. Hanson, and E. Riseman, "Recovering intrinsic scene characteristics," *Computer Vision Systems*, 1978. 10
[89] E. H. Adelson and A. P. Pentland, "The perception of shading and reflectance," *Perception as Bayesian inference*, 1996. 10
[90] A. Laurentini, "How many 2D silhouettes does it take to reconstruct a 3D object?" *Computer Vision and Image Understanding*, 1997. 13
[91] W. Richards, J. J. Koenderink, and D. D. Hoffman, "Inferring three-dimensional shapes from two-dimensional silhouettes," *Journal of the Optical Society of America A*, 1987. 13
[92] K.-M. Cheung, S. Baker, and T. Kanade, "Shape-from-silhouette across time part I: Theory and algorithms," *IJCV*, 2005. 13
[93] A. Laurentini, "The visual hull concept for silhouette-based image understanding," *IEEE PAMI*, 1994. 13
[94] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *IJCV*, 2000. 13

BIOGRAPHY SECTION

Tristan Aumentado-Armstrong received a BSc from McGill University in 2016 and an MSc from the University of Toronto (UofT) in 2018. He is currently a PhD student in AI at UofT and works at the Samsung Artificial Intelligence Center in Toronto on problems in computer vision. His research interests include 3D shape, disentangled representation learning, and generative modelling.

Stavros Tsogkas is a Research Scientist at the Samsung AI Center in Toronto. He is a member of the Image Quality Enhancement team, working on problems related to super-resolution, denoising and obstruction removal. His research interests also include 2D/3D shape understanding, object recognition, and segmentation. Stavros received his degree in Electrical and Computer Engineering from the National Technical University of Athens, Greece, and his PhD in Applied Math and Computer Science from Université Paris-Saclay, in France.

Sven Dickinson received the B.A.Sc. degree in Systems Design Engineering from the University of Waterloo, in 1983, and the M.S. and Ph.D. degrees in Computer Science from the University of Maryland, in 1988 and 1991, respectively. He is Professor and past Chair of the Department of Computer Science at the University of Toronto, and is also Vice President and Head of the new Samsung Toronto AI Research Center, which opened in May, 2018. Prior to that, he was a faculty member at Rutgers University where he held a joint appointment between the Department of Computer Science and the Rutgers Center for Cognitive Science (RuCCS). His research interests revolve around the problem of shape perception in computer vision and, more recently, human vision. He has received the National Science Foundation CAREER award, the Government of Ontario Premiere's Research Excellence Award (PREA), and the Lifetime Research Achievement Award from the Canadian Image Processing and Pattern Recognition Society (CIPPRS). He was the Editor-in-Chief of the IEEE Transactions on Pattern Analysis and Machine Intelligence, from 2017-2021, currently serves on seven editorial boards, and is co-editor of the Morgan & Claypool Synthesis Lectures on Computer Vision. He is a Fellow of the Institute of Electrical and Electronics Engineers (IEEE), a Fellow of the International Association for Pattern Recognition (IAPR), and an IEEE Golden Core Member.

Allan Jepson received his B.Sc. in 1976 from the University of British Columbia, his Ph.D. in Applied Mathematics in 1980 from Caltech, and then moved to a postdoctoral position in the Mathematics Department at Stanford University. In 1982 he joined the faculty at the Department of Computer Science at the University of Toronto, becoming a full professor in 1991. Dr. Jepson was an Associate of the Canadian Institute of Advanced Research (CIFAR) for 1986 to 1989, for 2004 to 2009, and was a Scholar at CIFAR for 1989 to 1995. He was Chief Scientist at Samsung's AI Center in Toronto from Sept. 2018 to Dec. 2022.