Directed Ray Distance Functions for 3D Scene Reconstruction

Nilesh Kulkarni[0000−0002−5114−2995], Justin Johnson[0000−0002−1251−088X], and David F. Fouhey[0000−0001−5028−5161]

University of Michigan, Ann Arbor, MI 48105
{nileshk, justincj, fouhey}@umich.edu

Abstract. We present an approach for full 3D scene reconstruction from a single unseen image. We trained on dataset of realistic non-watertight scans of scenes. Our approach uses a predicted distance function, since these have shown promise in handling complex topologies and large spaces. We identify and analyze two key challenges for predicting such image conditioned distance functions that have prevented their success on real 3D scene data. First, we show that predicting a conventional scene distance from an image requires reasoning over a large receptive field. Second, we analytically show that the optimal output of the network trained to predict these distance functions does not obey all the distance function properties. We propose an alternate distance function, the *Directed Ray Distance Function* (DRDF), that tackles both challenges. We show that a deep network trained to predict DRDFs outperforms all other methods quantitatively and qualitatively on 3D reconstruction from single image on Matterport3D, 3DFront, and ScanNet. [1]

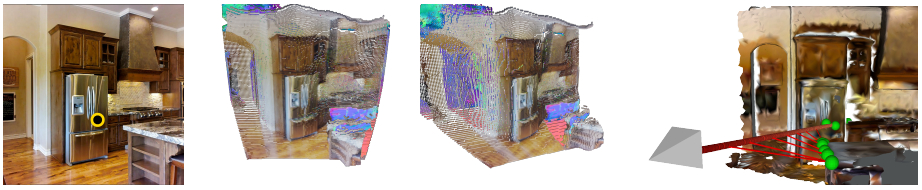
Keywords: Single Image 3D, Distance Functions

1 Introduction

Consider the image in Figure 1. What happens if you look behind the kitchen counter? To a layman, this single image shows a rich 3D world in which the floor continues behind the counter, and there are cabinets below the kitchen top. Our work aims to learn a mapping from a single image to the complete 3D, including visible *and* occluded surfaces. We learn such mapping from real, unstructured scans like Matterport3D [5] or ScanNet [11]. Unstructured scans are currently one of the richest sources of real-world 3D ground truth, and as more sensors like LIDAR scanners become ubiquitous, their importance will only grow.

Learning from these real-world scans poses significant challenges to the existing methods in computer vision. Voxel-based methods [18,10] scale poorly with size due to their memory requirements, and mesh-based ones [61] struggle with varying topology. Implicit functions [37,47] show promise for overcoming these size and topology challenges, but mostly focus on watertight data [37,8,42,47,52]

[1] Project Page: https://nileshkulkarni.github.io/scene_drdf



(a) Image (b) 3D outputs rendered from our model (c) Ray through the scene

Fig. 1: Given a single input image (a) our model generates its full 3D shown in (b) as two rendered novel views of our method's 3D output revealing the predicted occluded cabinet and floor. Visible surfaces are colored with image pixels; occluded ones show surface normals (pink: upwards; lavender: towards camera). In (c) we show a third person view of the scene with a red-ray from camera. The ray projects at the yellow-dot in the image (a). Nearest points to the ray shown as green spheres.

with a well-defined inside and outside regions for objects. This watertightness property enables *signed distance functions* (SDF) or occupancy functions, but limits them to data like ShapeNet [6], humans [47], or memorizing single watertight scenes [52]. Real 3D scans like Matterport3D [5] are off-limits for these methods. Exceptions include [9], which fits a single model with *unsigned distance function* (UDF) to a scene, and SAL [1,2] which learns SDFs on objects with well-defined insides and outsides that have scan holes. We believe that the lack of success in predicting implicit functions conditioned on previously unseen single image on datasets like Matterport3D [5] stems from two key challenges.

First, conventional distance functions depend on the distance to the nearest point in the full 3D scene. We show that this requires complex reasoning across an image. To see this, consider Fig. 1. The yellow point in (a) is the projection of the red ray in (c). We show the nearest point in the scene to each point on the ray in green. Near the camera, these are all over the kitchen counter to the right. Closer to the refrigerator, they finally are on the refrigerator. This illustrates that the projection of the nearest points to a point is often far from the projection of that point. Models estimating scene distances must integrate information across vast receptive fields to find the nearest points, which makes learning hard. We examine this in more detail in §4.1.

We propose to overcome these issues with a new distance-like function named the *Directed Ray Distance Function* (DRDF). Unlike the *Unsigned Distance Function* (defined by the nearest points in the scene), the DRDF is defined by points along the ray through a pixel; these project to the same pixel, facilitating learning. Unlike standard distance functions, DRDF's expected value under uncertainty behaves like a true distance function close to the surface. We learn to predict the DRDF with a PixelNerf [65]-style architecture and compare it with other distance functions. We also compare it to other conventional methods such as Layered Depth Images (LDI)[51]. Our experiments (§5) on Matterport3D [5], 3DFront [17], and ScanNet [11] show that the DRDF is substantially better at 3D scene recovery (visible and occluded) across all (three) metrics.

2 Related Work

Our approach aims to infer the full 3D structure of a scene from a single image using implicit functions, which relates with many tasks in 3D computer vision.

Scenes from a Single Image. Reconstructing the 3D scene from image cues is a long-term goal of computer vision. Most early work in 3D learning focuses on 2.5D properties [4] that are visible in the image, like qualitative geometry [24,15], depth [49] and normals [16]. Our work instead aims to infer the full 3D of the scene, including invisible parts. Most work on invisible surfaces focuses on single objects with voxels [18,10,21], point-clouds [35,14], CAD models [27] and meshes [19,20]. These approaches are often trained with synthetic data, e.g., ShapeNet [6] or images that have been aligned with synthetic ground-truth 3D [54]. Existing scene-level work, e.g., [58,34,33,41] trains on synthetic datasets with pre-segmented, watertight objects like SunCG [53]. Our work instead can be learned on real 3D like Matterport3D [5]. In summary, our work aims to understand the interplay between 3D, uncertainty, and learning [30,44,3] that has largely been explored in the depth-map space.

Implicit Functions for 3D Reconstruction. We approach the problem with learning implicit functions [37,42,8,43], which have shown promise in addressing scale and varying topology. These implicit functions have also been used in novel view synthesis [38,36,66,65], collision prediction [45], which differs from our work in goals. In reconstruction, implicit functions have shown impressive results on two styles of task: fitting to a single model to a fixed 3D scene (e.g., SIREN [52,9]) and predicting new single objects (e.g., PIFu [47,64]). Our work falls in the latter category as it predicts new scenes. While implicit functions have shown results on humans [47,48] and ShapeNet objects [64], most work relies on watertight meshes. Our non-watertight setting is more challenging. Few solutions have been proposed: assuming the SDF's existence and supervising it indirectly (SAL: [1,2],), voxelization of surfaces([69]), or predicting an unsigned distance function (UDF) [9] – we stress that [9] does not predict from RGB images. Our work can be trained with non-watertight 3D meshes and outperforms these approaches.

Recovering Occluded Surfaces. Our system produces the full 3D of a scene, including occluded parts, from a single image. This topic has been of interest to the community beyond previously mentioned volumetric 3D work (e.g., [18,10]). Early work often used vanishing-point-aligned box [23,12] trained on annotated data. While our approach predicts floors, this is learned, not baked in, unlike modern inheritors that have explicit object and layout components [57,28] or the ability to query for two layers [26,28]. An alternate approach is layered depth images (LDI) [51,13] or multi-plane depthmaps. LDIs can be learned without supervision [59], but when trained directly, they fare worse than our method.

3 Learning Pixel Aligned Distance Functions

We aim to reconstruct the full 3D of an unseen scene, including occluded parts, from a single RGB image while training on real scene captures [5]. Towards this

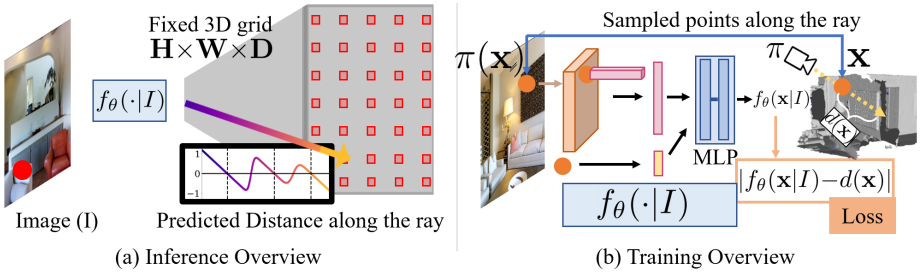


Fig. 2: Approach Overview. (a) At inference our model, $f_\theta(\cdot|I)$, conditioned on an input image (I) predicts a pixel conditioned distance for each point in a 3D grid. This frustum volume is then converted to surface locations using a decoding strategy. (b) At training time, our model takes an image and set of 3D points. It is supervised with the ground truth distance function for scene at these points. More details in §3.

goal we train an image-conditioned neural network to predict distance functions for 3D points in the camera frustum. Our training set consists of images and corresponding 3D meshes for supervision. We supervise our network with any ground-truth distance function *e.g.*, the *Unsigned Distance Function* (UDF).

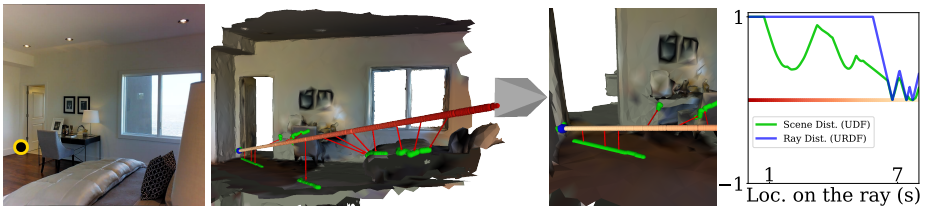
At test time, we consider only a single input image and a fixed set of 3D points in the camera view frustum. Our network predicts the distance function for each point using pixel aligned image features. The inference produces a grid of distances instead of a surface; we extract a surface with a *decoding strategy* (e.g., a thresholding strategy that defines values close to zero as surface locations).

Our setup is generic and can be paired with any distance function and a decoding strategy. We will discuss particular distance functions and decoding strategies later while discussing experiments. Experimentally, we will show that commonly used distance functions [9,1] do not work well when they are predicted in pixel conditioned way from a single image when trained on raw 3D data.

Inference. Given a input image like in Fig. 2 (left), we evaluate our model $f_\theta(\mathbf{x}; I)$ on pre-defined grid of points, $H \times W \times D$, in the 3D camera frustum to predict the distance function. It is then *decoded* to recover surface locations.

Training. At train time we are given n samples $\{(\mathbf{x}_i, I_i, d(\mathbf{x}_i)\}_{i=1}^n$ representing the 3D points ($\mathbf{x}_i$), input image (I_i) and the ground truth distance, $d(\mathbf{x}_i)$, computed using the 3D mesh. We find parameters θ that minimize the empirical risk $\frac{1}{n}\sum_{i=1}^n \mathcal{L}(f_\theta(\mathbf{x}_i, I_i), d(\mathbf{x}_i))$ with a loss function $\mathcal{L}$ *e.g.* the L1-Loss.

Model Architecture. We use a PixelNerf [65]-like architecture containing an encoder and multi layer perceptron (MLP). The encoder maps the image I to a feature map $\mathbf{B}$. Given a point $\mathbf{x}$ and the camera (π) we compute its projection on the image $\pi(\mathbf{x})$. We extract a feature at $\pi(\mathbf{x})$ from $\mathbf{B}$ with bilinear interpolation; the MLP uses the extracted image feature and a positional encoding [38] of $\mathbf{x}$ to make a final prediction $f_\theta(\mathbf{x}; I)$. Our code is hosted at `https://github.com/nileshkulkarni/scene_drdf` for reproducibility. Other details appear in the supplement.



(a) Image with ray center (b) Third person 3D views with the red ray and nearest points (c) Plot of Dist. Func.

Fig. 3: Scene vs ray distances . (a) The red ray intersects the scene at the black and yellow point in the image. Scene vs Ray distances along the points on red-shaded ray through the camera. (b) Two different 3D views showing intersections between the ray and the scene (which define the ray distance) in blue and the nearest points in the scene to the ray in green. These nearest scene points define scene distance. A network predicting scene distance must look all over the image (e.g., looking at the bed and chair to determine it for the ray). (c) Ground truth scene *vs.* ray distance functions for points on the ray. There are occluded intersections not visible in the image.

4 Behavior of Pixel Conditioned Distance Functions

Recent works have demonstrated overfitting of neural networks to single scenes [9,52,1,2] but none attempt to predict *scene-level 3D* conditioned on an image. We believe this problem has not been tackled due to two challenges. First, predicting a standard scene distance from a single image requires reasoning about large portions of the image. As we will show in §4.1, this happens because predicting scene distance for a point $\mathbf{x}$ requires finding the nearest point to $\mathbf{x}$ in the scene. This nearest point often projects to a part of the image that is far from $\mathbf{x}$'s projection in the image. Secondly, we will show in §4.3 that the uncertainty present in predicting pixel conditioned distance function incentivizes networks to produce outputs that lack basic distance function properties. These distorted distance functions do not properly decode into surfaces. To overcome the above challenges, we introduce a new distance function, the *Directed Ray Distance Function (DRDF)*. We will show analytically that DRDF retains distance-function like properties near the 3D surface under uncertainty. Yes

All distance functions are denoted with $d(\mathbf{x})$ where $\mathbf{x}$ is the query point in 3D space. We use M to denote the mesh of the 3D scene and $\overrightarrow{\mathbf{r}}$ to denote the ray originating from the camera passing through $\mathbf{x}$.

4.1 Scene *vs.* Ray Distances

A standard scene distance for a point $\mathbf{x}$ in a 3D scene M is the minimum distance from $\mathbf{x}$ to the points in M. If there are no modifications, this distance is called the *Unsigned Distance Function* (UDF) and can be operationalized by finding the nearest point $\mathbf{x}'$ in M to $\mathbf{x}$ and returning $||\mathbf{x} - \mathbf{x}'||$. We now define a *ray distance* for a point $\mathbf{x}$ as the minimum distance of $\mathbf{x}$ to any of the intersections between $\overrightarrow{\mathbf{r}}$ and M, which is operationalized similarly. The main distinction

between scene *vs.* ray distances boils down to which points define the distance. When calculating scene distances, all points in M are candidates for the nearest point. When calculating ray distances, only the intersections of $\overrightarrow{\mathbf{r}}$ and M are candidates for the nearest point. These intersections are a much smaller set.

We will now illustrate the above observation qualitatively with Fig. 3. We show in Fig 3(a) the projection of $\overrightarrow{\mathbf{r}}$ (and all points on it) onto the image as the yellow-center. We show in (b) a third person view of the scene with $\overrightarrow{\mathbf{r}}$ as the red-shaded-ray. We show the intersection point of $\overrightarrow{\mathbf{r}}$ with the scene M as blue points. For each point on the red ray, we show the nearest point on the mesh in green with an arrow going to that green point. The scene distance for points on the ray is defined by these nearest points in green. These nearest points are distributed all over M including the floor, bed, and chair, *etc.*. A pixel conditioned neural network predicting scene distances needs to integrate information of all the green regions to estimate scene distance for points projecting to the yellow ray projection. To show that this is not an isolated case, we quantify the typical projection of nearest points for scene distance to get an estimate of the minimum receptive field need to predict a distance using a neural network. We measure the distance between projections of the nearest points from the ray center. The average maximum distance to the ray center is $0.375\times$*image width* – averaged over 50K rays on Matterport3D [5]. Thus, a neural network predicting the scene distance needs to look at least this far to predict it.

This problem of integrating evidence over large regions vanishes for a ray distance function. By definition, the only points involved in defining a ray distance function for a point $\mathbf{x}$ lie on the ray $\overrightarrow{\mathbf{r}}$ since they are at the intersection of the mesh and the ray; these points project to the same location as $\mathbf{x}$. This simplifies a network's job considerably. We define the *Unsigned Ray Distance Function* (URDF) as the Euclidean distance to the nearest of these ray intersections.

We finally plot the UDF (scene) and URDF (ray) for the points along the red $\overrightarrow{\mathbf{r}}$, both truncated at 1m, in Fig 3(c). The UDF is fairly complex because different parts of the scene are nearest to the points along the ray at different distances. In contrast, the URDF is piecewise linear due to the few points defining it. We hypothesize this simplified form of a ray distance aids learning. More details in the supp.

4.2 Ray Distance Functions

It is convenient when dealing with a ray $\overrightarrow{\mathbf{r}}$ to parameterize each point $\mathbf{x}$ on the ray by a scalar multiplier z such that $\mathbf{x} = z\overrightarrow{\mathbf{r}}$. Now the distance functions are purely defined via the scalar multiplier along the ray. Suppose we define the set of scalars along the ray $\overrightarrow{\mathbf{r}}$ that correspond to intersections as $D_{\overrightarrow{\mathbf{r}}} = \{s_i\}_0^k$ (i.e., each point $s\overrightarrow{\mathbf{r}}$ for $s \in D_{\overrightarrow{\mathbf{r}}}$ is an intersection location). We can then define a variety of ray distances using these intersections. For instance, given any point along the ray, $z\overrightarrow{\mathbf{r}}$, we can define $d_{\text{UR}}(z) = \min_{s\in D_{\overrightarrow{\mathbf{r}}}} \|s - z\|$ as the minimum distance to the intersections. As described earlier, we call this *Unsigned Ray Distance Function* (URDF) – R here indicates it is a ray distance function. For watertight meshes, one can have a predicate inside($\mathbf{x}$) that is 1 when $\mathbf{x}$ is *inside* an object

and -1 otherwise. We can then define the *Signed Ray Distance Function* (SRDF) as $d_{\text{SR}}(z) = -\text{inside}(z\overrightarrow{\mathbf{r}})d_{\text{UR}}(z)$. Signed functions are standard in the literature but since our setting is non-watertight, the SRDF is impossible. Now we show how we can modify the SRDF for non-watertight settings.

Directed Ray Distance Function. We introduce a new ray based distance function called the *Directed Ray Distance Function* (DRDF). This can be seen as a modification to both URDF and SRDF; We define $d_{\text{DRDF}}(z) = \text{direction}(z)d_{\text{UR}}(z)$ where our predicate *direction*(z) is $\text{sgn}(s - z)$ where s is the nearest intersection to z. In practice DRDF is positive before the nearest intersection and negative after the nearest intersection. We call it *Directed* because the sign depends on the positioning along the ray. Unlike SRDF, there is no notion of inside, so the DRDF can be used with unstructured scans. Near an intersection, DRDF behaves like SRDF and crosses zero. DRDF has a sharp discontinuity midway between two intersections due to a sign change. We will analyze the importance of adding directional behavior to DRDF in the subsequent sections. Fig. 4 shows the difference between DRDF *vs.* URDF for multiple intersections on a ray.

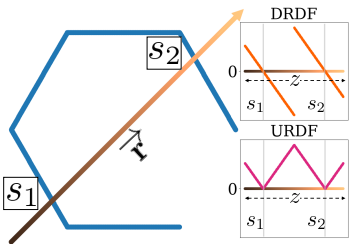


Fig. 4: DRDF *vs.* URDF in case of two intersections along the ray. Unlike URDF, DRDF is positive and negative

4.3 Modeling Uncertainty in Ray Distance Functions

When we predict distances in a single RGB image, the distance to an object in the scene is intrinsically uncertain. We may have a sense of the general layout of the scene and a rough distance, but the precise location of each object to the millimeter is not known. We investigate the consequences of this uncertainty for neural networks that predict distance functions conditioned on single view images. We analyze a simplified setup that lets us derive their optimal behavior.

In particular, if the network minimizes the MSE (mean-squared-error), its optimal behavior is to produce the expected value. In many cases, the expected value is precisely what is desired like in object detection [56,68] or in ARIMA models [40], weather prediction [63] but in others it leads to poor outcomes. For instance, in colorization [67,25], where one is uncertain of the precise hue, the expected value averages the options, leading to brown results; similar effects happen in rotation [7,39,32] and 3D estimation [62,31,29].

We now gain insights into the optimal output by analyzing the expected distance functions under uncertainty about the location of a surface. For simplicity, we derive results along a ray, although the supplement shows similar results hold true for scene distances. Since there is uncertainty about the surface location, the surface location is no longer a fixed scalar s but instead a random variable

S. The distance function now depends on the value s that the random variable S takes on. We denote the ray distance at z if the intersection is at s as $d(z;s)$.

The network's output at a location z is optimal in this setting if it equals the expected distance under S or $\mathrm{E}_S[d(z;s)] = \int_{\mathbb{R}} d(z;s)p(s)ds$, where $p(s)$ is the density of S. Thus, by analyzing $\mathrm{E}_S[d(z;s)]$ we can understand the optimal behavior. We note that this expectation is also optimal for other losses: under many conditions (see supp) $\mathrm{E}_S[d(z;s)]$ also is optimal for the L1 loss, and if $d(z;s)$ is $\{0,1\}$ such as in an occupancy function, then $\mathrm{E}_S[d(z;s)]$ is optimal for a cross-entropy loss. For ease of derivation, we derive results for when S is Gaussian distributed with its mean μ at the true intersection, standard deviation σ and CDF $\Phi(s)$. Since distance functions also depend on the next intersection, we assume it is at $S+n$ for some constant $n \in R^+$.

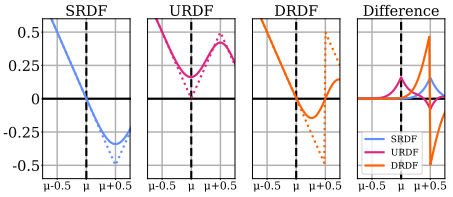


Fig. 5: True *vs.* Expected distance functions under uncertainty. Suppose the surface's location is normally distributed with mean μ at its true location and σ=0.2, and the next surface 1 is unit away. We plot the expected (solid) and true (dashed) distance functions for the SRDF, URDF, and DRDF and their difference (expected - true). The SRDF and DRDF closely match the true distance near the surface; the URDF does not.

We summarize salient results here, and a detailed analysis appears in the supplement. Figure 5 shows $\mathrm{E}_S[d(z;s)]$ for three ray distance functions (for $n = 1, \sigma = 0.2$). No expected distance function perfectly matches its true function, but each varies in where the distortion occurs. At the intersection, the expected SRDF and DRDF closely match the true function while the expected URDF is grossly distorted. Full derivations appear in the supplemental. The expected URDF has a minimum value of $\approx\sigma\sqrt{2/\pi}$ rather than 0. Similarly, its previously sharp derivative is now $\approx 2\Phi(z) - 1$, which is close to ± 1 *only* when z is far from the intersection. In contrast, the expected DRDF's distortion occurs at $\mu + \frac{n}{2}$, and its derivative $(np(z - \frac{n}{2}) - 1)$ is close to -1, except when z is close to $\mu + \frac{n}{2}$.

These distortions in expected distance function disrupt the decoding of distance functions to surfaces. For instance, a true URDF can turned into to a surface by thresholding, but the expected URDF has an uncertainty-dependent minimum value $\approx\sigma\sqrt{2/\pi}$, not 0. Since a nearby intersection often has less uncertainty than a far intersection, a threshold that works for near intersections may miss far intersections. Conversely, a threshold for far intersections may dilate nearby intersections. One may try alternate schemes, e.g., using the zero-crossing of the derivative. However, the expected URDF's shape is blunted; our empirical results suggest that finding its zero-crossing is ineffective in practice.

DRDF is more stable under uncertainty and requires just finding a zero-crossing. The zero-crossing at the intersection is preserved except when σ is

large (e.g., $\sigma=\frac{n}{3}$) in such cases other distance functions also break down. This is because the distortion for DRDF occur halfway to the other intersection. The only nuance is to filter out second zero-crossing after the intersection based on the crossing direction. Further analysis appears in the supplemental.

5 Experiments

We evaluate DRDF on real images of scenes and compare it to alternate choices of distance functions as well as conventional approaches such as Layered Depth Images[51]. We extensively optimize decoding schemes for our baseline methods. Their detailed description appear in the supplemental.

Our experiments evaluate each method's ability to predict the visible and occluded parts of the scene using standard metrics and a new metric that evaluates along rays.

Metrics. We use three metrics. A single metric cannot properly quantify reconstruction performance as each metric captures a different aspect of the task [55]. The first is scene Chamfer errors. The others are accuracy/completeness [50] and their harmonic mean, F1-score [55], for scenes and rays (on occluded points).

Chamfer L1. We compute Symmetric Chamfer L1 error for each scene with 30K points sampled from the ground truth and the prediction. We plot the fraction of scenes with Symmetric Chamfer L1 errors that are less than t for $t \in [0, 1]$m. It is more informative than just the mean across the dataset and compares performance over multiple thresholds.

Scene (Acc/Cmp/F1). Like [50,55], we report accuracy/Acc (% of predicted points within t of the ground-truth), completeness/Cmp (% of ground-truth points within t of the prediction), and their harmonic mean, F1-score. This gives a overall summary of scene-level reconstruction.

Rays (Acc/Cmp/F1), Occluded Points. We also evaluate reconstruction performance along each ray independently, measuring Acc/Cmp/F1 on each ray and reporting the mean. The paper shows results for occluded points, defined as all surfaces past the first intersection; the supplement contains full results. Evaluating each ray independently is a more stringent test for occluded surfaces than a scene metric: with scene-level evaluation on a image, a prediction can miss a surface (e.g., the 2nd intersection) on every other pixel. These missing predictions will be covered for by hidden surfaces on adjacent rays. Ray-based evaluation, however, requires each pixel to have all surfaces present to receive full credit.

Datasets. We see three key properties for datasets: the images should be real to avoid networks using rendering artifacts; the mesh should be a real capture since imitating capture holes is a research problem; and there should be lots of occluded regions. Our main dataset is Matterport3D [5], which satisfies all properties.

We also evaluate on 3DFront [17] and ScanNet [11]. While 3DFront has no capture holes, cutting it with a view frustum creates holes. ScanNet [11] is a popular in 3D reconstruction, but has far less occluded geometry compared to the other datasets. A full description of the datasets appears in the supplement.

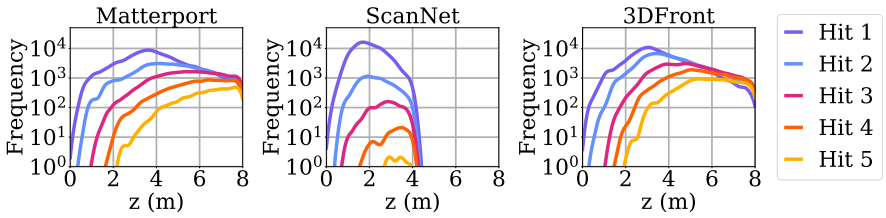


Fig. 6: Ray hit count distribution. We compare the distribution over surface hit (intersection) locations for first 4 hits over 1M rays. ScanNet has $\leq 1\%$ rays as compared to Matterport and 3DFront which have $\geq 25\%$ rays with more than 2 hits

Matterport3D [5]. We use the *raw images* captured with the Matterport camera. We split the 90 scenes into train/val/test (60/15/15) and remove images that are too close to the mesh ($\geq$60% of image within 1m) or are $>20°$ away from level. We then sample 13K/1K/1K images for train/val/test set.
3DFront [17]. This is a synthetic dataset of houses created by artists with a hole-free 3D geometry. We collect 4K scenes from 3DFront [17] after removing scenes with missing annotations. We select 20 camera poses and filter for bad camera poses similar to Matterport3D [5]. Our train set has 3K scenes with approximately 47K images. Val/Test sets have 500 scenes with 1K images each.
ScanNet [11]. We use splits from [11] (1045/156/312 train/val/test scenes) and randomly select 5 images per scene for train, and 10 images per scene for val/test. We then sample to a set of 33K/1K/1K images per train/val/test.
Dataset Scene Statistics. To give a sense of scene statistics, we plot the frequency of the locations of the first 5 ray hits (intersections) for each dataset (computed on 1M rays each) in Fig. 6. We show 99% of ScanNet rays have 1 or 2 hits, while $\geq$24% of Matterport3D [5] and 3DFront [17] rays have more than 2 hits.

5.1 Baselines

We compare against baselines to test our contributions. For fair comparison, all approaches use the same ResNet-34 [22] backbone and the same MLP. We extract features from multiple layers via bilinear interpolation [65]. Thus, different distance functions are trained identically by replacing the target distance. Each method's description consists of two parts: a prediction space parameterization and a decoding strategy to convert the inferred distances to surfaces.
Picking decoding strategies. Most baselines predict a distance function rather than a set of intersections and need a decoding strategy to convert distances to a set of surface locations. Some baselines have trivial strategies (e.g., direct prediction or zero-crossings); others are more sensitive and have parameters.

We tried multiple strategies for each baseline based on past work and theoretical analysis of their behavior. We report the best one by Scene F1 on Matterport3D [5]. When there are parameters, we tune them to ensure similar completeness to our method. Accuracy and completeness have a trade-off; fixing one

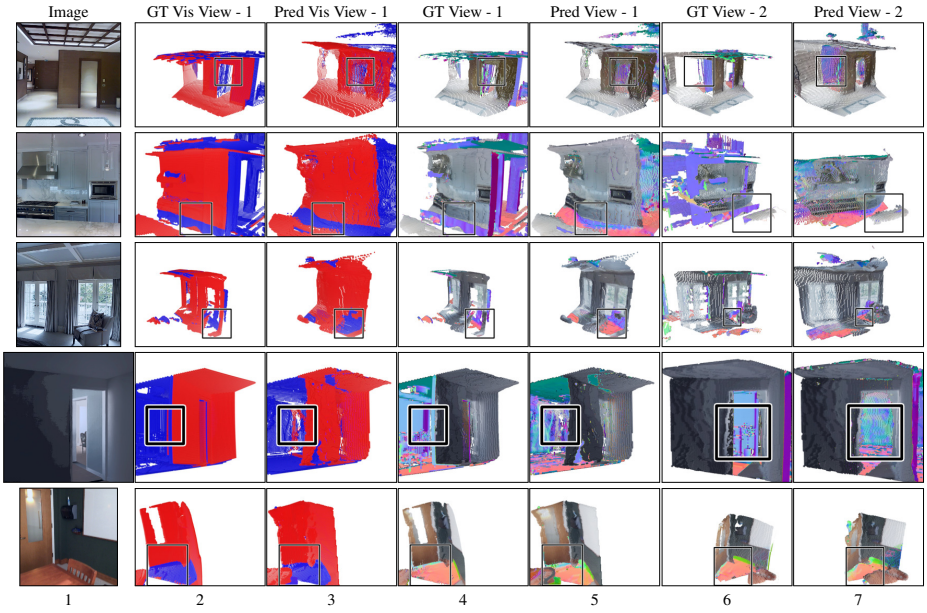


Fig. 7: Outputs from DRDF and ground-truth from new viewpoints. Columns 2,3 show visible points in red and occluded points in blue. Other columns, show the visible regions with the image and occluded regions with computed surface normals (, scheme from camera inside a cube). DRDF recovers occluded regions, such as a room behind the door (row 1 & 4), a floor behind the kitchen counter (row 2), and a wall and floor behind the chair/couch (row 3 & 5). Rows 1-3: Matterport3D; 4: 3DFront; 5: ScanNet.

ensures that methods are compared at similar operating points, making F1 score meaningful.

Layered Depth Images (LDI). To test the value of framing the problem as implicit function prediction, we train a method to predict a k-channel depthmap where the i^{th} output predicts the i^{th} intersection along the pixels. We use a L1 loss per pixel and intersection. We set $k = 4$, the same number of intersections the proposed approach uses. *Decoding.* LDI directly predicts surface locations.

Layered Depth Images with Confidence (LDI + C). We augment the the LDI baseline with k additional channels that represent the likelihood the i^{th} intersection exists. These added channels are trained with binary cross-entropy. *Decoding.* For each pixel we accept layers with predicted probability ≥ 0.5.

Unsigned Distance Function (UDF) [9]. Chibane *et al.* [9] fit UDF to a single 3D scene. We predict it from images. *Decoding.* We use `scipy.argrelextrema` [60] to find local extrema. We find local minima of the distance function within a 1m window along the ray. We found this works better than absolute thresholding (by 14.7 on F1). Sphere tracing and gradient-based optimization proposed by [9] performs substantially worse (25.7 on F1), likely since it assumes the predicted UDF behaves similar to a GT UDF.

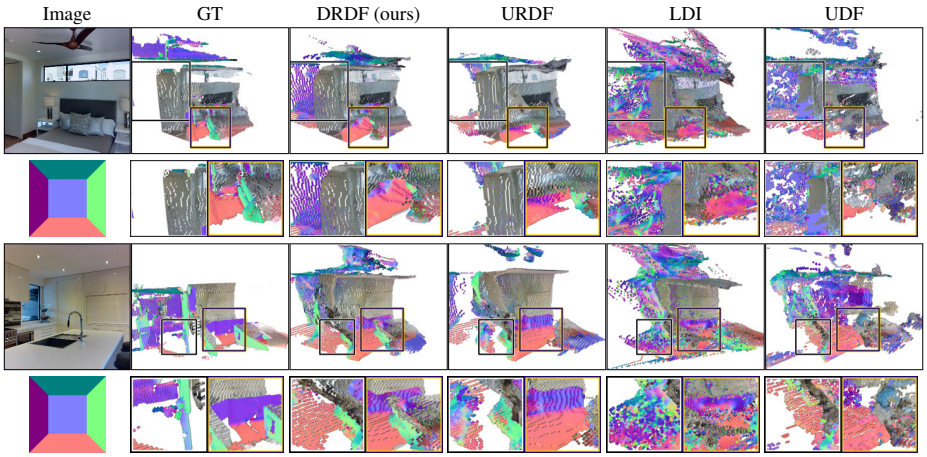


Fig. 8: We render the generated 3D outputs in a new view (rotated ←) with 2 crops for better visual comparison. Visible regions show the image; occluded regions show surface normals (legend shows a camera in a cube). DRDF produces higher quality results compared to LDI and UDF (row 1, 2, more consistency, smoother surface, no blobs). URDF misses parts of the floor (row 1/crop 2) and the green colored side of the kitchen counter (row2/crop 2). See supp. for more results.

Unsigned Ray Distance Function (URDF). Inspired from Chibane *et al.* [9] we compare UDF against its ray based version(URDF). Now, for direct comparison between ray distance functions we compare URDF against DRDF.

Decoding. We do NMS on thresholded data with connected components on the ray with predicted distance below a tuned constant τ, and keep the first prediction. This outperforms: thresholding (by 5.3 on F1); finding 0-crossings of the numerical gradient (by 11 on F1); and sphere tracing and optimization [9] (by 6.6 on F1).

Ray Sign-Agnostic Learning Loss (SAL) [1]. Traditional SDF learning is impossible due to the non-watertightness of the data and so we use the sign agnostic approach proposed by [1]. We initialize our architecture with the SAL initialization and train with the SAL loss. The SAL approach assumes that while the data may not be watertight due to noisy capture, the underlying model is watertight. In this case, rays start and end *outside* objects (and thus the number of hits along each ray is even). This is not necessarily the case on Matterport3D [5] and 3DFront [17].

Decoding. Following [1], we find surfaces as zero-crossings of the predicted distance function along the ray.

Ray Occupancy (ORF). Traditional interior/exterior occupancy is not feasible on non-watertight data, but one can predict whether a point is within r of a surface as a classification problem. This baseline tests the value of predicting ray distances, and not just occupancy. We tried several values of r ([0.1, 0.25, 0.5, 1]m) and report the best-performing version.

Table 1: Acc/Comp/F1Score. Thresholds: 0.5m (MP3D [5], 3DFront [17]), 0.2m (ScanNet [11]). **Bold is best**, underline is 2nd best per column. DRDF is best in F1 and accuracy in case of both metrics. For scene based it is comparable to the best in completeness and for ray based is occasionally 2nd best on Cmp. Gains on F1 score for occluded points are even larger than the full scene.

	Scene Based									**Ray Based (Occluded)**								
	MP3D [5]			3DFront [17]			ScanNet [11]			MP3D [5]			3DFront [17]			ScanNet [11]		
Method	Acc	Cmp	F1	Acc	Cmp	F1	Acc	Cmp	F1	Acc	Cmp	F1	Acc	Cmp	F1	Acc	Cmp	F1
LDI [51]	66.2	72.4	67.4	68.6	46.5	52.7	19.3	28.6	21.5	13.9	**42.8**	19.3	17.8	35.8	22.2	0.5	9.0	2.4
LDI +C	64.8	55.1	57.7	70.8	45.1	52.4	19.9	32.0	23.3	18.7	21.7	19.3	17.7	22.6	19.9	1.1	2.4	3.5
SAL [1]	66.1	25.5	34.3	80.7	28.5	39.5	51.2	**70.0**	57.7	5.5	0.5	3.5	24.1	4.3	11.4	2.4	**38.7**	5.6
UDF [9]	58.7	**76.0**	64.7	70.1	51.9	57.4	44.4	62.6	50.8	15.5	23.0	16.6	29.3	21.3	23.4	1.8	7.8	5.5
ORF	73.4	69.4	69.6	86.4	48.1	59.6	51.5	58.5	53.7	26.2	20.5	21.6	53.2	22.0	31.0	6.6	12.3	11.4
URDF	74.5	67.1	68.7	85.0	47.7	58.7	61.0	57.8	58.2	24.9	20.6	20.7	47.7	23.3	30.2	8.4	11.6	13.8
DRDF	**75.4**	72.0	**71.9**	**87.3**	**52.6**	**63.4**	**62.0**	62.7	**60.9**	**28.4**	30.0	**27.3**	**54.6**	**56.0**	**52.6**	**9.0**	20.4	**16.0**

Decoding. Each surface, in theory, produces two locations with probability 0.5: an onset and offset crossing. Finding all 0.5-crossings leads to doubled predictions. Instead, we consider all adjacent and nearby pairs of offsets and onsets, and average them; unpaired crossings are kept. This outperforms keeping just a single 0.5-crossing (by 4.7 on F1).

5.2 Results

Qualitative Results. Qualitative results of our method appear throughout the paper (by itself in Fig. 7 and compared to baselines in Fig. 8). Our approach is is often able to generate parts of the occluded scene, such as a room behind a door, cabinets and floor behind kitchen counters, and occluded regions behind furniture. Sometimes the method completes holes in the ground-truth that are due to scanning error. On the other hand we see our method sometimes fails to characterize the missing parts as detailed occluded 3D *e.g.* plants. Compared to baselines, our approach does qualitatively better. LDI and UDF often have floating blobs or extruded regions, due to either predicting too many layers (LDI) or having a distance function that is challenging to predict (UDF). URDF produces qualitatively better results, but often misses points in occluded regions.
Quantitative Results. These results are borne out in the quantitative results. Figure 9 shows the Chamfer plot and Table. 1 reports the scene distance and occluded surfaces metrics along rays. DRDF consistently does at least as well, or substantially better than the baselines on Chamfer. In general, DRDF does substantially better than all baselines. In a few situations, a baseline beats DRDF in completeness at the cost of substantially worse accuracy. However, a single baseline is not competitive with DRDF *across* datasets: SAL works well on ScanNet [11] and LDI works well on Matterport3D [5].

LDI performs worse than DRDF because it cannot vary its number of intersections; simply adding a second stack of outputs (LDI + C.) is insufficient. This is because DRDF can learn *where* things tend to be, while LDI-based methods have to learn the order in which things occur (e.g., is the floor 2nd or the 3rd intersection at a pixel?). SAL performs competitively on ScanNet, likely because

of the relatively limited variability in numbers of intersections per ray; when tested on Matterport3D and 3DFront, its performance drops substantially.

We compare against a Monocular Depth Estimation (MDE) baseline with a pre-trained MiDaS [46] model. It has been trained on more datasets and has an optimal scale and translation fit per-image (which our models do not get). As it predicts one intersection its F1 is lower, 57.2 *vs.* 71.9 for DRDF on Matterport3D [5]. Nonetheless, we see advances in MDE complementary to advances in DRDF.

The most straightforward way to learn on non-watertight data is to predict unsigned scene distances [9] which has been shown to work with memorizing 3D scenes. However, predicting it from a single image is a different problem entirely, and scene distances require integration of information over large areas. This leads to poor performance. Predicting distances on rays alleviates this challenge, but recovering intersections remains hard even with multiple decoding strategies. Thus, DRDF outperforms URDF. ORF similarly requires decoding strategies and is sensitive to training parameters. In contrast, by accounting for the uncertainty in surface location, DRDF requires a simple decoding strategy and outperforms other methods.

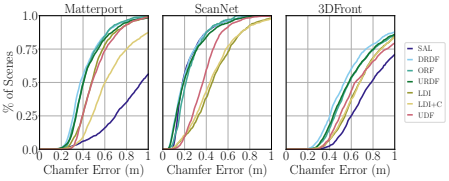


Fig. 9: Chamfer L1: % of scenes on Y-axis as a function of symmetric Chamfer L1 error $\leq t$ on X-axis. DRDF is better on Matterport and 3DFront, and comparable to the best other method on ScanNet.

Conclusions. This paper introduced a new distance function, DRDF, for 3D reconstruction from an unseen image. We use real 3D, non-watertight data at training. We showed that DRDF does not suffer from pitfalls of other distance functions and outperforms other conventional methods. DRDF achieves substantially better qualitative results and has a simple *decoding strategy* to recover intersections – thanks to its stable behavior near intersections. DRDF's progress in learning 3D from real data is extendable to learning from multi-view data. Our approach, however, has societal limitations as our data does not reflect most peoples' reality: Matterport3D for instance, has many lavish houses and this may widen the technological gap. However, we are optimistic that our system will enable learning from scans collected by ordinary people rather than experts.

Acknowledgements. We would like the thank Alexandar Raistrick and Chris Rockwell for their help with the 3DFront dataset. We like to thank Shubham Tulsiani, Ekdeep Singh Lubana, Richard Higgins, Sarah Jabour, Shengyi Qian, Linyi Jin, Karan Desai, Mohammed El Banani, Chris Rockwell, Alexandar Raistrick, Dandan Shan, Andrew Owens for comments on the draft versions of this paper. NK was supported by TRI. Toyota Research Institute ("TRI") provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity

References

1. Atzmon, M., Lipman, Y.: Sal: Sign agnostic learning of shapes from raw data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2565–2574 (2020) 2, 3, 4, 5, 12, 13
2. Atzmon, M., Lipman, Y.: Sal++: Sign agnostic learning with derivatives. arXiv preprint arXiv:2006.05400 (2020) 2, 3, 5
3. Bae, G., Budvytis, I., Cipolla, R.: Estimating and exploiting the aleatoric uncertainty in surface normal estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13137–13146 (2021) 3
4. Barrow, H., Tenenbaum, J., Hanson, A., Riseman, E.: Recovering intrinsic scene characteristics. Comput. Vis. Syst **2**(3-26), 2 (1978) 3
5. Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., Song, S., Zeng, A., Zhang, Y.: Matterport3d: Learning from rgb-d data in indoor environments. arXiv preprint arXiv:1709.06158 (2017) 1, 2, 3, 6, 9, 10, 12, 13, 14
6. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 2, 3
7. Chen, K., Snavely, N., Makadia, A.: Wide-baseline relative camera pose estimation with directional learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 3258–3268 (June 2021) 7
8. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019) 1, 3
9. Chibane, J., Mir, A., Pons-Moll, G.: Neural unsigned distance fields for implicit function learning. In: Advances in Neural Information Processing Systems (NeurIPS) (December 2020) 2, 3, 4, 5, 11, 12, 13, 14
10. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: European conference on computer vision. pp. 628–644. Springer (2016) 1, 3
11. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5828–5839 (2017) 1, 2, 9, 10, 13
12. Del Pero, L., Bowdish, J., Kermgard, B., Hartley, E., Barnard, K.: Understanding bayesian rooms using composite 3d object models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2013) 3
13. Dhamo, H., Navab, N., Tombari, F.: Object-driven multi-layer scene decomposition from a single image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5369–5378 (2019) 3
14. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017) 3
15. Fidler, S., Dickinson, S., Urtasun, R.: 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In: Advances in neural information processing systems. pp. 611–619 (2012) 3
16. Fouhey, D.F., Gupta, A., Hebert, M.: Data-driven 3d primitives for single image understanding. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3392–3399 (2013) 3

17. Fu, H., Cai, B., Gao, L., Zhang, L., Li, C., Zeng, Q., Sun, C., Fei, Y., Zheng, Y., Li, Y., Liu, Y., Liu, P., Ma, L., Weng, L., Hu, X., Ma, X., Qian, Q., Jia, R., Zhao, B., Zhang, H.: 3d-front: 3d furnished rooms with layouts and semantics. arXiv preprint arXiv:2011.09127 (2020) 2, 9, 10, 12, 13
18. Girdhar, R., Fouhey, D.F., Rodriguez, M., Gupta, A.: Learning a predictable and generative vector representation for objects. In: European Conference on Computer Vision. pp. 484–499. Springer (2016) 1, 3
19. Gkioxari, G., Malik, J., Johnson, J.: Mesh r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9785–9795 (2019) 3
20. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: A papier-mâché approach to learning 3d surface generation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 216–224 (2018) 3
21. Häne, C., Tulsiani, S., Malik, J.: Hierarchical surface prediction for 3d object reconstruction. In: 2017 International Conference on 3D Vision (3DV). pp. 412–420. IEEE (2017) 3
22. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 10
23. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the spatial layout of cluttered rooms. In: 2009 IEEE 12th international conference on computer vision. pp. 1849–1856. IEEE (2009) 3
24. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 1, pp. 654–661. IEEE (2005) 3
25. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. CVPR (2017) 7
26. Issaranon, T., Zou, C., Forsyth, D.: Counterfactual depth from a single rgb image. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 0–0 (2019) 3
27. Izadinia, H., Shan, Q., Seitz, S.M.: Im2cad. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5134–5143 (2017) 3
28. Jiang, Z., Liu, B., Schulter, S., Wang, Z., Chandraker, M.: Peek-a-boo: Occlusion reasoning in indoor scenes with plane representations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 113–121 (2020) 3
29. Jin, L., Qian, S., Owens, A., Fouhey, D.F.: Planar surface reconstruction from sparse views. International Conference on Computer Vision (ICCV) (2021) 7
30. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 5580–5590 (2017) 3
31. Ku, J., Pon, A.D., Waslander, S.L.: Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11867–11876 (2019) 7
32. Kulkarni, N., Gupta, A., Tulsiani, S.: Canonical surface mapping via geometric cycle consistency. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2202–2211 (2019) 7
33. Kulkarni, N., Misra, I., Tulsiani, S., Gupta, A.: 3d-relnet: Joint object and relational network for 3d prediction. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2212–2221 (2019) 3
34. Li, L., Khan, S., Barnes, N.: Silhouette-assisted 3d object instance reconstruction from a cluttered scene. In: ICCV Workshops (2019) 3

35. Lin, C.H., Kong, C., Lucey, S.: Learning efficient point cloud generation for dense 3d object reconstruction. arXiv preprint arXiv:1706.07036 (2017) 3
36. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. arXiv preprint arXiv:2008.02268 (2020) 3
37. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019) 1, 3
38. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. arXiv preprint arXiv:2003.08934 (2020) 3, 4
39. Mousavian, A., Anguelov, D., Flynn, J., Kosecka, J.: 3d bounding box estimation using deep learning and geometry. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 7074–7082 (2017) 7
40. Newbold, P.: Arima model building and the time series analysis approach to forecasting. Journal of forecasting **2**(1), 23–35 (1983) 7
41. Nie, Y., Han, X., Guo, S., Zheng, Y., Chang, J., Zhang, J.J.: Total3dunderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 55–64 (2020) 3
42. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019) 1, 3
43. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: European Conference on Computer Vision. pp. 523–540. Springer (2020) 3
44. Poggi, M., Aleotti, F., Tosi, F., Mattoccia, S.: On the uncertainty of self-supervised monocular depth estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3227–3237 (2020) 3
45. Raistrick, A., Kulkarni, N., Fouhey, D.F.: Collision replay: What does bumping into things tell you about scene geometry? arXiv preprint arXiv:2105.01061 (2021) 3
46. Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., Koltun, V.: Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2020) 14
47. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2304–2314 (2019) 1, 2, 3
48. Saito, S., Simon, T., Saragih, J., Joo, H.: Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 84–93 (2020) 3
49. Saxena, A., Sun, M., Ng, A.Y.: Make3d: Learning 3d scene structure from a single still image. IEEE transactions on pattern analysis and machine intelligence **31**(5), 824–840 (2008) 3

50. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In: 2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06). vol. 1, pp. 519–528. IEEE (2006) 9
51. Shade, J., Gortler, S., He, L.w., Szeliski, R.: Layered depth images. In: Proceedings of the 25th annual conference on Computer graphics and interactive techniques. pp. 231–242 (1998) 2, 3, 9, 13
52. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. Advances in Neural Information Processing Systems **33** (2020) 1, 2, 3, 5
53. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1746–1754 (2017) 3
54. Sun, X., Wu, J., Zhang, X., Zhang, Z., Zhang, C., Xue, T., Tenenbaum, J.B., Freeman, W.T.: Pix3d: Dataset and methods for single-image 3d shape modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2974–2983 (2018) 3
55. Tatarchenko, M., Richter, S.R., Ranftl, R., Li, Z., Koltun, V., Brox, T.: What do single-view 3d reconstruction networks learn? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3405–3414 (2019) 9
56. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: A Simple and Strong Anchor-Free Object Detector. TPAMI (2020) 7
57. Tulsiani, S., Gupta, S., Fouhey, D.F., Efros, A.A., Malik, J.: Factoring shape, pose, and layout from the 2d image of a 3d scene. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 302–310 (2018) 3
58. Tulsiani, S., Su, H., Guibas, L.J., Efros, A.A., Malik, J.: Learning shape abstractions by assembling volumetric primitives. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2635–2643 (2017) 3
59. Tulsiani, S., Tucker, R., Snavely, N.: Layer-structured 3d scene inference via view synthesis. In: ECCV (2018) 3
60. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods **17**, 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2 11
61. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: ECCV (2018) 1
62. Wang, X., Fouhey, D.F., Gupta, A.: Designing deep networks for surface normal estimation. In: CVPR (2015) 7
63. Weyn, J.A., Durran, D.R., Caruana, R.: Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere. Journal of Advances in Modeling Earth Systems **12**(9), e2020MS002109 (2020) 7
64. Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U.: Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In: Advances in Neural Information Processing Systems. pp. 492–502 (2019) 3
65. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: CVPR (2021) 2, 3, 4, 10

66. Zhang, K., Riegler, G., Snavely, N., Koltun, V.: Nerf++: Analyzing and improving neural radiance fields. arXiv preprint arXiv:2010.07492 (2020) 3
67. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV (2016) 7
68. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv preprint arXiv:1904.07850 (2019) 7
69. Zhu, S., Ebrahimi, S., Kanazawa, A., Darrell, T.: Differentiable gradient sampling for learning implicit 3d scene reconstructions from a single image. In: International Conference on Learning Representations (2021) 3