NFL: Normal Field Learning for 6-DoF Grasping of Transparent Objects

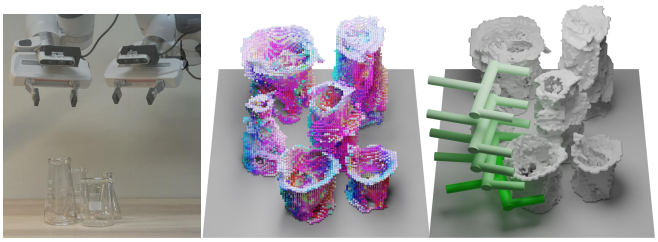
Junho Lee[1], Sang Min Kim[1], Yonghyeon Lee[2], Young Min Kim[1*]

***Abstract*—We present Normal Field Learning (NFL), a robust yet practical solution to perceive 3D layouts of transparent objects and grasp them quickly. Conventional input modalities for vision-based grasping do not provide sufficient information for transparent objects. However, with the recent advance on datasets and algorithms for transparent objects, we can at least obtain noisy estimates of normals and masks for various real-world conditions. Instead of directly using the RGB images, we propose to use the estimates to train a neural volume, which serves as an intermediate representation ignorant of challenging appearance variations. We formulate the training objective to account for inherent uncertainty in individual estimation, and together with the volumetric aggregation, we can reliably extract useful geometric information for grasping. Our neural volume deploys a voxel-grid based representation, motivated by acceleration techniques of neural radiance fields. However, we directly store the normal and density values in the grid cells instead of latent features. Our modification allows direct access to the geometric values without additional inference or volume rendering, further enhancing the efficiency. Our results show over 85% success rates in grasping in cluttered scenes with only 40 seconds of training time.**

***Index Terms*—Deep Learning for Visual Perception, Deep Learning in Grasping and Manipulation, Grasping**

I. Introduction

TRANSPARENT objects exhibit unreliable measurements, making it difficult for robots to grasp reliably. The images of transparent objects often contain minimal visual cues, and depth cameras also miss the transparent surfaces. Recent approaches build a designated module for transparent objects using different modalities, such as thermal cameras or polarized cameras [1], [2], [3], [4], but such approaches require additional hardware. Meanwhile, for images including transparent objects, some data-driven approaches have shown promising results in 2D vision tasks (e.g., segmentation) [5], [6], [7], [8]. On the other hand, existing data-driven methods for 3D recognition for scenes including transparent objects [9] have shown less-than-desirable performance. To this end, we propose a framework for learning 3D volume given multiple RGB images including transparent objects, by exploiting the 2D recognition results of their mid-level representations (e.g., segmentation masks).



(a) RGB capture (b) Normal field (c) Grasp algorithm

Fig. 1. Overview of NFL method. Our method collects RGB images with a robot arm (a), then represents the scene as a grid-based normal field (b). We search for viable grasps via the reconstructed geometry obtained from the normal field (c).

Recently, neural field representations have experienced tremendous success in various vision tasks [10], [11], [12]. While Neural Radiance Fields (NeRF) [13] primarily synthesizes impressive novel view images, it also captures 3D geometric information. Specifically, the neural network simultaneously outputs the RGB color value and the volume density σ for a 3D location after trained only with multiple 2D images. Adopting the NeRF framework as their foundation, recent works such as Dex-NeRF [14] and Evo-NeRF [15] have performed robotic grasping, where the parameter σ serves as an indicator to determine the presence of objects.

However, we observe that raw images of transparent objects cannot directly train an accurate NeRF volume, as the volume density σ indicates opacity rather than existence. That means, according to the definition, a perfectly transparent object should have zero σ values on the surface, leading to the failure of the existing NeRF-based grasping methods. Instead, we propose Normal Field Learning (NFL), where we train a neural volume from pixel-wise surface normal estimate instead of RGB images. Specifically, we focus on the normal field defined on objects' surfaces; to identify object pixels, segmentation masks are employed. In this case, the volume density σ directly relates to the objects' existence.

Our framework compensates for inevitable errors in surface normals and segmentation masks, estimated from pre-trained networks. Through the process of aggregating multi-view esti-

Manuscript received: July, 11, 2023; Revised October, 8, 2023; Accepted November, 3, 2023. This paper was recommended for publication by Editor Markus Vincze upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2023-00208197) and Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) [NO.2021-0-01343, Artificial Intelligence Graduate School Program (Seoul National University)]. Yonghyeon Lee was the beneficiary of an individual grant from CAINS supported by a KIAS Individual Grant (AP092701) via the Center for AI and Natural Sciences at Korea Institute for Advanced Study.

[1]Junho Lee, Sang Min Kim, and Young Min Kim are with School of Engineering, Department of Electrical and Computer engineering, Seoul National University, South Korea. [2]Yonghyeon Lee is with Center for Artificial Intelligence and Natural Sciences, Korea Institute for Advanced Study, South Korea. *Young Min Kim is the corresponding author of this work. youngmin.kim@snu.ac.kr

Digital Object Identifier (DOI): see top of this page.

mations, we can mitigate spurious noises and train the coherent volume similar to the conventional NeRF [13]. Additionally, to devise a more robust algorithm, we take into account the estimation uncertainty in pixel-wise normals and masks and train the normal field in a way that it trusts more confident estimations. Specifically, the normal uncertainty is measured as the degree of disagreement among the network outputs when subjected to specific input image transformations, such as a color-jittering transformation. We use von Mises-Fisher distribution [16], similar to Gaussian distribution on S^2, to represent this stochasticity of the normal estimation. The mask uncertainty is naturally captured in the probability vector output of the segmentation network, modeled as a Bernoulli distribution.

While the vanilla NeRF is notorious for slow training and rendering, the feature-grid representation (e.g., DVGO [17]) is one of the recent advances in acceleration. NFL adopts the DVGO approach and further boosts its training speed by eliminating an intermediate module (i.e., an MLP module that converts feature vectors to color and density values) that we found is not necessary for our setup. As a result, we can train a normal field as in Figure 1 (b) in 40 seconds using 30 images of transparent objects. Our feature-grid normal field representation can be directly used to quickly find a collision-free robot grasping trajectory as in Figure 1 (c). This contrasts previous NeRF-based grasping works [14], [15] that render multiple-depth images with volumetric rendering to infer grasping points.

In summary, our contributions are as follows:

- We propose to use estimated surface normals and masks, rather than raw RGB images, to achieve more accurate geometric reconstructions for transparent objects;
- We formulate a probabilistic framework robust to prediction errors, by taking into account the estimation uncertainty of surface normals and masks;
- Our method is fast and can be directly used for grasping without the need for rendering depth images, leveraging the feature-grid representation of the volume.

Our experiments display the performance and practicality of NFL in terms of reconstruction quality, speed, and the grasping success rate in real-world scenarios under significant domain discrepancies. We additionally evaluate the functionality of our algorithm on a photorealistic scene created using Blender Cycles [18].

II. Related work

Transparent objects. Perceiving transparent objects usually requires a dedicated framework. While depth sensors are common input modality for robotic grasping, the measurement rays are not reflected on the surface of transparent objects and thus fail to provide reliable measurements [19]. Different input modalities, such as as polarization cameras or thermal infrared cameras [20], [21], are known to stably perceive objects despite transparency. However, we target relying only on conventional RGB cameras instead of special hardware.

The visual appearances on RGB images are significantly different for transparent objects, and the performance of ordinary vision modules, such as depth estimation, segmentation, and object detection, often deteriorates when observing them. Several works propose synthetic datasets with transparent objects [19], [22] which allow training networks that perform various tasks. However, it is questionable whether the synthetic data extensively capture possible real-world variations. As an alternative to synthetic datasets, large-scale real-world datasets emerge, which contain high-quality annotation of surface normals, depth, and segmentation masks [23], [24], [25]. The diversity of the dataset leads to stable performance in novel real-world scenarios without further adaptations. MonoGraspNet [26] utilizes such real-world datasets to perform grasping of transparent objects, but rely on a single-view estimation without volumetric aggregation.

Vanilla and grid-based NeRF. Neural radiance fields (NeRF) [13] represent the scene with a neural network that maps a position and viewing direction into color and volume density. The field can generate a color image from an arbitrary viewpoint via volume rendering. However, the training and inference of the neural volume are notorious for slow speed, as volume rendering requires integrating multiple inferred sample values per pixel ray. Among many follow-up works to overcome limitations of the vanilla NeRF, several methods expedite the rendering speed with explicit grid representations [17], [27], [28]. Our practical pipeline adapts the grid-based acceleration of DVGO [17] into normal field learning, and stores density and normal values in the grid cells. We obtain necessary geometric representation of real scenes within 40 seconds while achieving remarkable grasping performance.

NeRF for grasping transparent objects. In addition to novel view synthesis, NeRF can extract coherent volumetric representation only from image inputs. Several works proposed deploying NeRF to acquire geometric information for robotic tasks, including grasping transparent objects. Dex-NeRF [14] is the first attempt to grasp transparent objects using NeRF. Their grasping algorithm uses depth image, rendered from the density estimates of NeRF. However, the density values of σ are inevitably erroneous for transparent objects because σ represents opacity, which is the opposite of transparency, in RGB images. Furthermore, Dex-NeRF is based on the original NeRF implementation, which suffers from slow training speed. Evo-NeRF [15] accelerates training by building upon Instant NGP [28]. GraspNeRF [22] employs few-shot NeRF [29], [30] and predicts the signed distance fields in 90 miliseconds from only six input images. Both works are aware of the inherent noise in density values obtained from RGB images and train modules robust to errors: Evo-NeRF directly trains a grasping module on rendered depth of transparent objects and GraspNeRF concurrently predicts grasping points from multi-view feature aggregation and the recovered geometry. Both Evo-NeRF and GraspNeRF use a synthetic dataset rendered with Blender [18] to obtain training data, which may be susceptible to domain discrepancy when applied in real environments.

III. Method

In this section, we present a probabilistic framework that learns the 3D geometric field of a scene that contains multiple transparent objects, from which we can assess reliable grasp

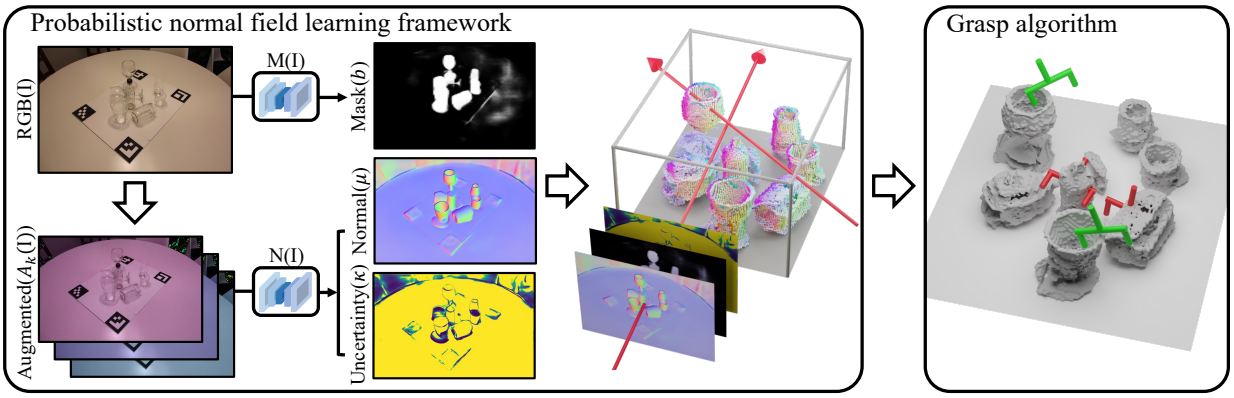


Fig. 2. The structure of NFL model. The inputs for probabilistic normal field learning are the pixel-wise estimation of surface normal modeled as von Mises-Fisher distribution and estimated object mask modeled as a Bernoulli distribution. The output is a 3D normal field where each point is mapped to a normal vector n and density σ. From the normal field, we sample reliable grasps, among which we select one that can induce trajectory without collision.

poses. We assume that the only available observations are multiple RGB images taken from different angles with known camera poses. We do not utilize any depth image as input to our algorithm. The surfaces of objects are assumed to be smooth almost everywhere so that the surface normals are well-defined for most parts of the objects.

In NFL, the primary step is to learn normal and density fields simultaneously, where the normal field $n : \mathbb{R}^3 \to S^2$ maps a 3D point to a unit vector and the density field $\sigma : \mathbb{R}^3 \to \mathbb{R}$ maps a 3D point to a non-negative scalar. Specifically, for any point $x \in \mathbb{R}^3$ on the surface of an object, $n(x)$ is defined to be the surface normal. For any point $x \in \mathbb{R}^3$ not on the surface of an object, $n(x)$ is undefined and we allow it to take any arbitrary value. This arbitrary assignment will not be problematic since our grasp pose generation only uses normal vectors on object surfaces. The density $\sigma(x)$ of a point can serve as an indicator for surface points with valid normal values. Non-zero density values indicate surface points, and zero otherwise.

In Section III-A, we propose a probabilistic method that fits the normal and density fields $n(x), \sigma(x)$ from the RGB image set. Section III-B describes the grasp pose generation and motion planning algorithms based on the estimated normal and density fields.

A. Probabilistic Normal Field Learning Framework

Our normal field adopts the standard volume rendering technique along the camera rays [31], [13]. However, we propose to learn $n(x)$ and $\sigma(x)$ with 2D mid-level representations, namely normal maps and segmentation masks estimated with pre-trained networks, instead of directly using the RGB input images. Figure 2 (*Left*) illustrates an overview of our normal field learning framework. In the following sections, we propose stochastic representations of the estimated mid-level representations, and a maximum likelihood training of the normal field, where we take into account estimation uncertainties of both the normal maps and segmentation masks.

1) Stochastic Normal and Mask from RGB Images: We find the stochastic representation of the estimated normal maps with test-time augmentation [32]. We denote a pre-trained normal estimator by $N : \mathrm{I} \mapsto N(\mathrm{I})$ where I is an input RGB image and $N(\mathrm{I})$ is an estimated normal map. The normal vector at the (i, j)-th pixel is denoted by $N_{ij}(\mathrm{I}) \in S^2$. We consider a class of operators that transform input data $\mathcal{A} : \mathrm{I} \mapsto \mathcal{A}(\mathrm{I})$ that should not alter the outputs if N is a robust estimator. That is, $N(\mathrm{I}) = N(\mathcal{A}(\mathrm{I}))$. For instance, if we emulate subtle changes in lighting with a color-jittering transformation, the estimated shape should remain constant. However, pre-trained models often fail to remain invariant under those transformations; we use the extent of deviations as a measure of estimation uncertainty.

Let $\mathcal{A}_k$ be a normal-preserving transformation operator for $k = 1, \ldots, m$ (including the identity map), as discussed earlier, and consider m normal estimates of an image I, $\{N(\mathcal{A}_k(\mathrm{I}))\}_{k=1}^m$. For each (i, j)-th pixel, there are m estimated normal vectors $\{N_{ij}(A_k(\mathrm{I})) \in S^2\}_{k=1}^m$. By using these estimates, we fit a continuous probability density function for each pixel of the normal map.

We use the von Mises-Fisher distribution [16] as a density model for $N \in S^2$:

$$f(N; \mu, \kappa) := \frac{\kappa}{2\pi(e^{\kappa} - e^{-\kappa})} \exp(\kappa \mu^T N), \tag{1}$$

where f is a probability density function, $\mu \in S^2$ is the mean direction parameter, and $\kappa \in \mathbb{R}$ is the concentration parameter. The greater the value of κ, the higher the concentration of f around μ, and the lower the uncertainty of N. For each (i, j)-th pixel in RGB image I, excluding the background regions, the Maximum Likelihood Estimates (MLEs) of the parameters can be computed as follows. The MLE of the mean parameter is simply given as $\mu_{ij}(\mathrm{I}) = \bar{N}/\|\bar{N}\|$ where $\bar{N}$ is the arithmetic mean $\bar{N} := \frac{1}{m} \sum_{k=1}^m N_{ij}(\mathcal{A}_k(\mathrm{I}))$. On the other hand, the MLE of the concentration parameter has no closed-form expression, yet instead, a simple approximation to $\kappa_{ij}(\mathrm{I})$ is available [33]:

$$\kappa_{ij}(\mathrm{I}) = \frac{\|\bar{N}\|(3 - \|\bar{N}\|^2)}{1 - \|\bar{N}\|^2}. \tag{2}$$

In addition, we find the stochastic representation of the estimated segmentation masks using a pretrained mask estimator $M : \mathrm{I} \mapsto M(\mathrm{I})$. Let the estimated segmentation

mask at the (i,j)-th pixel be denoted by $M_{ij}(\mathrm{I}) \in [0,1]$. We then interpret each pixel of the segmentation masks as the Bernoulli distribution with a parameter $M_{ij}(\mathrm{I})$, denoted by $\mathrm{B}(1, M_{ij}(\mathrm{I}))$, since our segmentation network is trained with the cross entropy loss.

2) Maximum likelihood normal field learning: Given the stochastic representations of the normal maps and segmentation masks, we can formulate the normal field learning as a variant of maximum likelihood training with the differentiable volume rendering [31]. Let $r_{ij}(\cdot;\mathrm{I})$ be a ray emitted from the camera that passes through (i,j)-th pixel of an image I. We accumulate $n(x)$ and $\sigma(x)$ along a ray $r_{ij}(t;\mathrm{I})$ with near and far bounds t_n and t_f, and define a projected normal map as

$$N_{ij}^{\text{proj}}(\mathrm{I}) := \text{Normalize}\Big(\int_{t_n}^{t_f} T(t)\sigma(r_{ij}(t;\mathrm{I}))n(r_{ij}(t;\mathrm{I}))dt \Big), \tag{3}$$

where Normalize$(\cdot)$ maps a vector to a unit vector and $T(t) = \exp(-\int_{t_n}^{t} \sigma(r_{ij}(s;\mathrm{I}))ds)$ is the accumulated transmittance along the ray. The projected normal map has a dependency to $n(x)$ and $\sigma(x)$, so it may be better to write as $N_{ij}^{\text{proj}}(\mathrm{I}; n, \sigma)$, but we omit n and σ for notation convenience.

We then define a per-pixel loss function for an (i,j)-pixel of an input image I as the negative log-likelihood that measures how unlikely the projected normal map $N_{ij}^{\text{proj}}(\mathrm{I})$ is, given the probability density function of the estimated normal map $f(N; \mu_{ij}(\mathrm{I}), \kappa_{ij}(\mathrm{I}))$, as follows:

$$l_{ij}(\mathrm{I}) := -\log f(N_{ij}^{\text{proj}}(\mathrm{I}); \mu_{ij}(\mathrm{I}), \kappa_{ij}(\mathrm{I})). \tag{4}$$

Ignoring the normalization constant that does not depend on both n and σ, the per-pixel loss further simplifies to

$$l_{ij}(\mathrm{I}) = -\kappa_{ij}(\mathrm{I})\mu_{ij}(\mathrm{I})^T N_{ij}^{\text{proj}}(\mathrm{I}). \tag{5}$$

By minimizing the loss, the projected normal $N_{ij}^{\text{proj}}(\mathrm{I})$ is fitted to $\mu_{ij}(\mathrm{I})$ – since the inner product of two unit vectors is maximal when they are equal – with the weight of $\kappa_{ij}(\mathrm{I})$. Higher weights are assigned to pixels with more certain normal estimations, i.e., those with higher values of $\kappa_{ij}(\mathrm{I})$.

Although it is tempting to sum $l_{ij}(\mathrm{I})$ over all the indices i,j to define the final loss function, it is unnecessary to take into account l_{ij} for the background pixels where no object exists. We use the stochastic representation of the segmentation mask to minimize l_{ij} only when (i,j) pixel is an object pixel. Specifically, we sample $b_{ij}(\mathrm{I})$ from the per-pixel Bernoulli distribution $\mathrm{B}(1, M_{ij}(\mathrm{I}))$, and consider the product $b_{ij}(\mathrm{I})l_{ij}(\mathrm{I})$ as a new loss term. Therefore, when $b_{ij}(\mathrm{I}) = 0$ (i.e., (i,j) belongs to background pixels), the loss will be ignored.

Additionally, it is important to learn accurate σ since we use density values in practice to distinguish between object and non-object regions. Up to this point, our attention has been on the normal field component for object pixels, i.e., when $b_{ij}(\mathrm{I}) = 1$, and the loss is not sufficient to learn the correct σ. We therefore introduce a density penalization term $(1 - 2b_{ij}) \int_{t_n}^{t_f} \sigma(r_{ij}(t;\mathrm{I}))dt$ into the loss function. For an object pixel $b_{i,j}(\mathrm{I}) = 1$, minimizing the loss encourages the accumulated density σ to maintain a positive value. When $b_{i,j}(\mathrm{I}) = 0$, or it is a background pixel, the loss effectively suppresses the density along the ray $r_{ij}(t;\mathrm{I}), t \in [t_n, t_f]$ to be zero.

In summary, the loss function for an image I is:

$$\mathcal{L}(\mathrm{I}; n, \sigma) := \sum_{i,j} b_{ij}(\mathrm{I})l_{ij}(\mathrm{I}) + (1 - 2b_{ij}(\mathrm{I})) \int_{t_n}^{t_f} \sigma(r_{ij}(t;\mathrm{I}))dt, \tag{6}$$

where $b_{ij}(\mathrm{I}) \sim \mathrm{B}(1, M_{ij}(\mathrm{I}))$. And, given a set of images $\{\mathrm{I}^{(l)}\}_{l=1}^{L}$, the final loss function for the normal and density field is the empirical mean of losses for input images:

$$\mathcal{L}(n, \sigma) := \frac{1}{L} \sum_{l=1}^{L} \mathcal{L}(\mathrm{I}^{(l)}; n, \sigma). \tag{7}$$

B. Grasping Algorithm Based on Normal and Density Grids

After we obtain the 3D geometric layout parameterized by normal and density functions, we can regress for 6 DoF grasp positions and collision-free trajectories as shown in Figure 2 (***Right***). Since the process to train normal and density values is similar to conventional NeRF formulation [13], we can accelerate the training by employing discrete voxel grid representations as suggested by recent works [17]. Fast speed is particularly useful where robot concurrently observes the scene and grasps an object. Furthermore, our formulation can avoid volume rendering to find the surface points and their normals, and estimate them directly from individual grid points. Note that the original voxel-grid implementation trained with color images stores feature vectors on the grid and uses an additional shallow MLP to regress for the color values. However, our approach heavily utilizes the density and normal grid, where the grid points contain the raw density and normal values. The direct access of grid representation enables us to quickly find feasible grasping points, and generate collision-free paths.

1) 6-DoF Grasp Candidate Generation: The density and normal grids provide surface point positions and their surface normal vectors, respectively, which can be directly used to find feasible grasping points for a two-finger gripper [34], [35]. The grasp candidate generation algorithm is as follows: From a set of 3D points on the density grid, we extract a subset of points $\{x_1, \ldots, x_N\}$ that have density values higher than a threshold value δ_{density}, which represent points on the surface. Denote the corresponding normal vectors for those surface points by $\{n_1, \ldots, n_N\}$, obtained from the normal grid. Then, we first find a set of index pairs (i,j) that satisfies two conditions: (i) $|x_i - x_j| < \delta_{\text{dist}}$ with some distance threshold δ_{dist} defined considering the gripper width and (ii) $n_i \cdot (x_i - x_j) \geq 0.99$ to find antipodal points. We denote the set of these index pairs S, which serves as the candidate grasp points.

2) Collision-Free Path Planning: Given the set S of candidate grasp points, we find a robot configuration and path that grasps the object while avoiding collisions with the surrounding environments. The collision against 3D scene layout is approximated by comparing against the set of surface points $\{x_1, \ldots, x_N\}$ which are already extracted. Ideally, looking at all of the candidate grasps in S would lead to better performance. However, in order to expedite the process of selecting grasps, we sort the index pair set S with the density

score $\sigma(x_i) + \sigma(x_j)$ for $(i, j) \in S$, and start from the one with the highest density score. We examine the top 100 pairs from grasp candidates in practice. For each candidate grasp pair, we test 8 pitch angles for a gripper and search for the configuration that does not collide with any of surface points. Then we find the joint trajectory of a robot that arrives at the target pose without collision. We use PyBullet planning library [36] for the collision detection and path planning.

IV. Experiments

In this section, we compare our NFL-based 3D reconstruction method and 6-DoF grasping algorithm with existing RGB image-based 3D reconstruction methods and depth rendering-based grasping methods. In Section IV-A we compare geometry reconstruction results, and in Section IV-B we compare the grasping perfomance in the real world.

Baselines. We select baselines for comparison that satisfy two conditions. First, baselines should take as input multiview RGB images along with their camera parameters. Second, baselines should be trained solely by real-world data, since our main target is grasping in complex configurations in the real world. The compared baselines are: NeRF [13], DVGO [17], Dex-NeRF [14], Dex-DVGO and GraspNeRF [22]. [1]. All of them train the color and density fields directly from RGB images. NeRF uses neural networks to represent the fields, whereas DVGO employs voxel grid representations followed by a shallow MLP. Dex-NeRF applies a threshold on the density values to better capture transparent objects; we implement the same technique for DVGO and denote it by Dex-DVGO. For DVGO and Dex-DVGO, we retain the MLP in the original implementation of DVGO, since they could not converge without retraining for transparent objects.

Implementation Details. We estimate surface normals and masks given RGB images by neural networks trained with the large-scale real-world dataset [24]. For surface normal estimation, we finetune the work by Bae et al. [32] for 20000 steps. To predict segmentation masks, we train a CNN based model [37] for 20 epochs. For test-time augmentations, we employ color jittering transformations (hue transformations) provided by the Torchvision library [38]. We use one original and nine augmented images to fit von Mises-Fisher distribution on pixel-wise normals.

We train NFL on an RTX 3090 for 5000 steps with a grid resolution of 150^3 for our real scene. The bounding box dimensions are 50cm $\times$ 60cm $\times$ 40cm and it is positioned to enclose the robot's workspace. It takes around 40 seconds to train a normal field.

A. 3D Scene Reconstruction: Synthetic and Real

First, we provide a quantitative evaluation of geometry reconstruction using synthetic scenes with transparent objects. We create a photo-realistic rendering of a scene with 3 objects of glass textures on top of a wooden table using Blender Cycles [18]. We train all models until convergence given 100 input images. We compare the accuracy of rendered depth images on three metrics from ClearGrasp [19]. Specifically, we render test-view depth images from viewpoints equally spaced on a cylinder bounding the objects. Accuracy is defined as the ratio of object pixels where the error is within a threshold. Compared to RMSE, this metric is agnostic of scene scale. The threshold is selected as 5%, 10%, and 25% of the groundtruth depth, as in [19].

[1]ClearGrasp [19] is not included since ClearGrasp uses only a single image. Although Evo-NeRF [15] satisfies both conditions, we are unable to find the custom grasping dataset that is necessary to implement the algorithm.

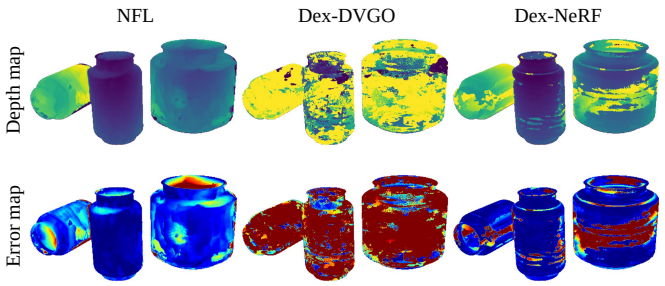


Fig. 3. Qualitative results on synthetic data. Top row shows rendered depth for object pixels. Bottom row depicts error maps with respect to groundtruth depth (red: high error, blue: low error). Our model captures more accurate depth of all objects.

TABLE I
Depth reconstruction results on Blender dataset. Bold represents best results.

Config	Grid-based (DVGO)			Non grid-based (NeRF)	
Method	**NFL**	Dex-DVGO	DVGO	Dex-NeRF	NeRF
$\delta_{0.05}$	**85.35%**	20.48%	19.13%	74.96%	27.17%
$\delta_{0.10}$	**92.64%**	29.25%	38.02%	81.39%	56.25%
$\delta_{0.25}$	**97.49%**	46.50%	82.47%	95.15%	93.81%
Time (min.)	**2**	15	15	720	720

Table I contains the quantitative results on the depth accuracy. Our model, NFL, marks the best accuracy in all of the depth accuracy metrics while taking less time than others. Dex-NeRF and NeRF are the vanilla representation that utilize a single MLP to represent the entire scene, and take about 12 hours. The grid-based acceleration shortens the training time of Dex-DVGO and DVGO into 15 minutes. NFL further accelerates the time into 120 seconds by removing MLPs, which are required to synthesize novel view images for Dex-DVGO and DVGO. We also observed that depth rendering technique of Dex-NeRF improves the accuracy of geometry compared to NeRF.

The qualitative results are presented in Figure 3. The top row shows the depth map for the object pixels, whereas the bottom row shows the error map relative to the groundtruth depth. For the error map, red pixels indicate higher error while blue pixels indicate more accurate depth measurements. Our method reconstructs more accurate depth for most of the object pixels while Dex-DVGO fails to capture geometry. Dex-NeRF performs reasonably except the middle part of the large bowl, where the object appears more transparent and lacks visual evidence in RGB images.

Ablation on Input Modality. We verify that normals and masks are more effective to reconstruct the geometry of trans-

TABLE II
DEPTH RECONSTRUCTION ACCURACY DEPENDING ON INPUT MODALITY. ON BOTH GRID-BASED AND NON-GRID-BASED METHODS

Config	Grid-based (DVGO)				Non grid-based (NeRF)			
Modality	Normal	Mask	Normal + Mask	RGB	Normal	Mask	Normal + Mask	RGB
$\delta_{0.05}$	11.48%	90.57%	**96.85%**	19.13%	83.01%	89.94%	**93.40%**	27.17%
$\delta_{0.1}$	72.30%	94.18%	**97.52%**	38.02%	92.26%	94.33%	**96.97%**	56.25%
$\delta_{0.25}$	**99.10%**	97.78%	98.17%	82.47%	99.66%	98.72%	**99.93%**	93.81%

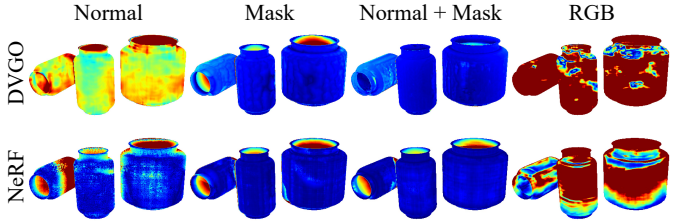


Fig. 4. Error maps of depth obtained from different input modalities (red: high error, blue: low error). For both grid-based (DVGO) and non-grid-based (NeRF) methods, RGB input cannot accurately reconstruct depth for transparent objects. Using both normal and mask leads to the best results. Grid-based method (DVGO) also struggles to capture geometry when using only normals for an input.

parent objects compared to directly using RGB images. Similar to the previous experiment, we test on a scene with 3 objects with 100 images as input and compare the reconstruction results in Table II. Given the same set of images and the camera parameters, the reconstruction is significantly more effective with surface normals and masks than with RGB images. Although all combinations show better reconstruction performance compared to RGB input, using normals and masks together demonstrates the best performance in both grid based (DVGO) and non-grid based (NeRF) approaches. The error maps in Figure 4 show similar results. The normal maps without masks are not sufficient to reconstruct accurate geometry. Masks alone cannot accurately capture the concave parts of the bottle.

TABLE III
EFFECTS OF MASK SAMPLING AND STOCHASTIC NORMALS

Uncertainty	None	Mask Sampling	Stochastic Normal Mask Sampling
$\delta_{0.05}$	84.71%	85.25%	**85.35%**
$\delta_{0.10}$	**92.85%**	91.27%	92.64%
$\delta_{0.25}$	97.24%	96.25%	**97.49%**

Effect of Considering Input Uncertainty. While normals and masks are useful in training the field to obtain geometry of transparent objects, the estimations can be erroneous. NFL employs probabilistic formulation as described in Section III-A1 to consider the uncertainty in the estimated inputs. Stochastic normal incorporates the distribution of normal estimation from test-time augmentation, and it is ablated by considering all rays equally in Eq. (6). Mask sampling can be ablated by using binary values for $b_{ij}(\mathrm{I})$ after thresholding. Table III shows that using both stochastic normal and mask sampling records the best results.

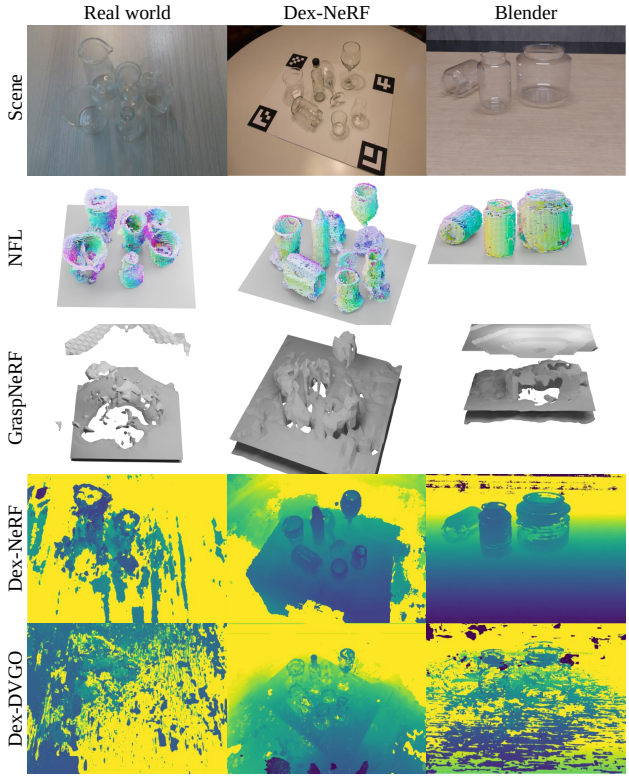


Fig. 5. Robustness across various scenes. We visualize the geometric representations that different methods use for grasping (normal field for ours, depth image for baselines). Our method stably creates normal fields for real world, Dex-NeRF, and blender scenes. GraspNeRF finds the ground, but occasionally fails to reconstruct the geometry. Both Dex-NeRF and Dex-DVGO fail to demonstrate stable performance in all scenes.

Robustness Across Scenes. Our normal field aggregates normal estimates from pre-trained networks and shows robust performance across challenging appearance variations. Figure 5 compares the reconstructed geometry of our model and two baselines in different representations. While NFL builds the 3D normal field, baselines use depth maps rendered from the learned neural volume to obtain grasp points. NFL successfully builds normal fields for all cases using the same setup despite the variation in objects, lighting, camera parameters and more. The input from our real world scene (left) is especially challenging, containing fewer visual cues (edges of transparent objects) compared to other datasets. In contrast, Dex-NeRF and Dex-DVGO are sensitive to the appearance or lighting of the transparent objects and could not render an accurate depth map from our real-world images. The dataset of Dex-NeRF exhibits considerable visual evidence, such as stark edges of objects, although the objects are transparent.

TABLE IV
REAL WORLD GRASP SUCCESS RATES FOR SEVERAL CONFIGURATIONS: SINGLE SMALL, SINGLE BIG, AND CLUTTER.

Model	**NFL**	GraspNeRF [22]	Dex-NeRF[14]	Dex-DVGO	NeRF [13]	DVGO [17]
Single Small	**71.43**%	14.28%	28.57%	0%	0%	14.28%
Single Big	**57.14**%	14.28%	0%	0%	0%	0%
Clutter	**85.71**%	28.57%	28.57%	0%	14.28%	0%
Time	40 sec	**90ms**	12 hr	15 min	12 hr	15 min

GraspNeRF utilizes only 6 images and has a fast prediction time, but fails to accurately capture the geometry of the objects. In order to successfully run GraspNeRF, we need to find the six viewpoints that are similar to the original implementation. In addition, we re-scale the scene to fit in the 30cm cube originally used in GraspNeRF. After these measures, GraspNeRF captures the ground stably. However, GraspNeRF's reconstruction performance fluctuates depending on the scene. Dex-NeRF and Dex-DVGO directly use color images and only obtain the volume density σ for such opaque appearances out of transparent objects.

B. Real Robot 6-DoF Grasping

We use a real-world robot to capture input images to acquire geometric layout and perform grasping tasks. We attach a Realsense d435i camera at the end effector of a Panda Franka Emika robot using a 3D printed mount, and use only RGB images for input. We calculate the camera poses using the end effector location and the relative transformation between the mounted camera and the end effector. We utilize 30 images for our model and baselines other than GraspNeRF. Since GraspNeRF is a pretrained network as a whole, adjusting the number of utilized images is not straight-forward. Thus for GraspNeRF, we match the 6 viewpoints utilized in the original GraspNeRF paper. Our image capturing system takes up to 1 second to move and capture to each viewpoint. To assist placing the objects in the same configuration for different methods, we built a GUI that overlays object positions from the previous observations. For grasping baselines other than GraspNeRF, we render a depth image from the view looking straight down at the objects as in Dex-NeRF. Then we calculate the best top-down grasp points using the model from Dex-Net [39], which is pretrained for two-fingered grippers. We move the gripper 20 cm above the grasping point then lower it to grasp. For GraspNeRF, we utilize the pretrained model which predicts the neural field and grasping end effector pose. We follow the Gaussian smoothing process, then select the grasp with the highest quality value. Each trial is classified as a success if the robot successfully picks up an object and places it into a bin.

Table IV contains the grasp success rates after seven grasps for three different scene configurations: Single Small, Single Big, and Cluttered. The Big and Small are assessed based on the relative size compared to the gripper width, which reflects the grasping difficulty. For Cluttered scenes, we put six objects within a 30 cm $\times$ 30 cm square region. While our model shows good performance on all three configurations, baselines struggle to effectively grasp objects, even with more training time. Specifically, NFL excels in the cluttered scene thanks to a rich set of candidate grasps obtained from accurate holistic reconstruction. All of the baselines struggle on the Single Big scene. In Single Big configuration, the thickness for a candidate grasp is comparable to the width of an open gripper, and therefore we need to find the precise grasp location from accurate geometry. In contrast NFL succeeds in grasping over 50% of single big scenes, indicating the superior reconstruction accuracy of NFL. GraspNeRF and Dex-NeRF show the second best performance. Especially, GraspNeRF succeeds for at least one experiment for all configurations, with the least inference time. Dex-DVGO, NeRF, and DVGO fail in our grasping experiments. GraspNeRF marks the least time to build a neural field. Different to the Blender dataset experiment of Table I, our method takes 40 seconds to train. This is because we use less images as input (100 vs 30), which allows shorter training time.

V. CONCLUSION AND LIMITATIONS

We proposed NFL, a robust and practical solution to perform grasping transparent objects. NFL models predicted surface normals and masks as a probablistic distribution and learns a normal field of a real scene in 40 seconds. The normal field includes accurate holistic 3D geometry from which we can quickly infer grasp positions. Experiments on various datasets show the robustness of NFL to many real-world scenes and superior grasping performance. We also conduct ablation studies to support the choice of using surface normals and segmentation masks rather than RGB to form neural fields for transparent objects.

Although NFL exhibits stable performance for a variety of scenes, it is still far from perfect. First, expediting the algorithm will definitely improve the practicality of grasping. Even building on the grid-based DVGO algorithm, NFL is still slower than GraspNeRF. With faster speed, one can deploy the algorithm to quickly refresh the geometry in sequential grasping. Additionally, NFL finds the 3D geometry of the scene from input images surrounding the bounded volume of known workspace. Although the required setting is not difficult for conventional manipulation setting, it may hinder generalizing to all images in the wild. For example, we could not evaluate NFL on the HAMMER dataset [40] which has images captured from only one side of the objects.

ACKNOWLEDGMENT

We would like to thank Cheolhui Min for helping with operating the Panda Franka.

REFERENCES

[1] D. Miyazaki, M. Saito, Y. Sato, and K. Ikeuchi, "Determining surface orientations of transparent objects based on polarization degrees in visible and infrared wavelengths," *JOSA A*, vol. 19, no. 4, pp. 687–694, 2002.

[2] D. Huo, J. Wang, Y. Qian, and Y.-H. Yang, "Glass segmentation with rgb-thermal image pairs," *IEEE Transactions on Image Processing*, vol. 32, pp. 1911–1926, 2023.

[3] A. Kalra, V. Taamazyan, S. K. Rao, K. Venkataraman, R. Raskar, and A. Kadambi, "Deep polarization cues for transparent object segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8602–8611.

[4] H. Mei, B. Dong, W. Dong, J. Yang, S.-H. Baek, F. Heide, P. Peers, X. Wei, and X. Yang, "Glass segmentation using intensity and spectral polarization cues," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 622–12 631.

[5] G. Chen, K. Han, and K.-Y. K. Wong, "Tom-net: Learning transparent object matting from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9233–9241.

[6] H. Mei, X. Yang, Y. Wang, Y. Liu, S. He, Q. Zhang, X. Wei, and R. W. Lau, "Don't hit me! glass detection in real-world scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3687–3696.

[7] E. Xie, W. Wang, W. Wang, M. Ding, C. Shen, and P. Luo, "Segmenting transparent objects in the wild," in *European Conference on Computer Vision*. Springer, 2020, pp. 696–711.

[8] J. Lin, Z. He, and R. W. Lau, "Rich context aggregation with reflection prior for glass surface detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 415–13 424.

[9] K. Chen, S. James, C. Sui, Y.-H. Liu, P. Abbeel, and Q. Dou, "Stereopose: Category-level 6d transparent object pose estimation from stereo images via back-view nocs," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 2855–2861.

[10] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 786–12 796.

[11] B. Poole, A. Jain, J. T. Barron, and B. Mildenhall, "Dreamfusion: Text-to-3d using 2d diffusion," *arXiv preprint arXiv:2209.14988*, 2022.

[12] K. Zhang, F. Luan, Z. Li, and N. Snavely, "Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5565–5574.

[13] 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," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[14] J. Ichnowski*, Y. Avigal*, J. Kerr, and K. Goldberg, "Dex-NeRF: Using a neural radiance field to grasp transparent objects," in *Conference on Robot Learning*, 2021.

[15] J. Kerr, L. Fu, H. Huang, Y. Avigal, M. Tancik, J. Ichnowski, A. Kanazawa, and K. Goldberg, "Evo-nerf: Evolving nerf for sequential robot grasping of transparent objects," in *6th Annual Conference on Robot Learning*, 2022.

[16] N. I. Fisher, T. Lewis, and B. J. Embleton, *Statistical analysis of spherical data*. Cambridge university press, 1993.

[17] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5459–5469.

[18] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

[19] S. Sajjan, M. Moore, M. Pan, G. Nagaraja, J. Lee, A. Zeng, and S. Song, "Clear grasp: 3d shape estimation of transparent objects for manipulation," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 3634–3642.

[20] D. Gao, Y. Li, P. Ruhkamp, I. Skobleva, M. Wysocki, H. Jung, P. Wang, A. Guridi, and B. Busam, "Polarimetric pose prediction," in *European Conference on Computer Vision*. Springer, 2022, pp. 735–752.

[21] J. Kim, M.-H. Jeon, S. Jung, W. Yang, M. Jung, J. Skin, and A. Kim, "Transpose: Large-scale multispectral dataset for transparent object," *arXiv preprint arXiv:2307.05016*, 2023.

[22] Q. Dai, Y. Zhu, Y. Geng, C. Ruan, J. Zhang, and H. Wang, "Graspnerf: Multiview-based 6-dof grasp detection for transparent and specular objects using generalizable nerf," in *IEEE International Conference on Robotics and Automation*, 2023.

[23] H. Fang, H.-S. Fang, S. Xu, and C. Lu, "Transcg: A large-scale real-world dataset for transparent object depth completion and a grasping baseline," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7383–7390, 2022.

[24] X. Chen, H. Zhang, Z. Yu, A. Opipari, and O. Chadwicke Jenkins, "Clearpose: Large-scale transparent object dataset and benchmark," in *European Conference on Computer Vision*. Springer, 2022, pp. 381–396.

[25] E. Xie, W. Wang, W. Wang, P. Sun, H. Xu, D. Liang, and P. Luo, "Segmenting transparent object in the wild with transformer," *arXiv preprint arXiv:2101.08461*, 2021.

[26] G. Zhai, D. Huang, S.-C. Wu, H. Jung, Y. Di, F. Manhardt, F. Tombari, N. Navab, and B. Busam, "Monograspnet: 6-dof grasping with a single rgb image," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 1708–1714.

[27] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "Tensorf: Tensorial radiance fields," in *European Conference on Computer Vision*, 2022.

[28] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022.

[29] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, "Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 124–14 133.

[30] Y. Liu, S. Peng, L. Liu, Q. Wang, P. Wang, C. Theobalt, X. Zhou, and W. Wang, "Neural rays for occlusion-aware image-based rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7824–7833.

[31] J. Kajiya and B. Von Herzen, "Ray tracing volume densities," *ACM SIGGRAPH Computer Graphics*, vol. 18, pp. 165–174, 1984.

[32] G. Bae, I. Budvytis, and R. Cipolla, "Estimating and exploiting the aleatoric uncertainty in surface normal estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 137–13 146.

[33] A. Banerjee, I. S. Dhillon, J. Ghosh, S. Sra, and G. Ridgeway, "Clustering on the unit hypersphere using von mises-fisher distributions." *Journal of Machine Learning Research*, vol. 6, no. 9, 2005.

[34] J. Mahler, S. Patil, B. Kehoe, J. Van Den Berg, M. Ciocarlie, P. Abbeel, and K. Goldberg, "Gp-gpis-opt: Grasp planning with shape uncertainty using gaussian process implicit surfaces and sequential convex programming," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 4919–4926.

[35] H. Huang, D. Wang, X. Zhu, R. Walters, and R. Platt, "Edge grasp network: A graph-based se (3)-invariant approach to grasp detection," in *IEEE International Conference on Robotics and Automation*, 2023, pp. 3882–3888.

[36] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," 2016.

[37] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[38] T. maintainers and contributors, "Torchvision: Pytorch's computer vision library," https://github.com/pytorch/vision, 2016.

[39] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics," *arXiv preprint arXiv:1703.09312*, 2017.

[40] H. Jung, P. Ruhkamp, G. Zhai, N. Brasch, Y. Li, Y. Verdie, J. Song, Y. Zhou, A. Armagan, S. Ilic *et al.*, "On the importance of accurate geometry data for dense 3d vision tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 780–791.