BAYESIAN METHODS FOR MULTI-OBJECT RECONSTRUCTION WITH HINGE POINT REPRESENTATIONS

by

Herbert Wright

A Senior Honors Thesis Submitted to the Faculty of
The University of Utah
In Partial Fulfillment of the Requirements for the

Honors Degree in Bachelor of Science

In

Computer Science

Approved:

[redacted]

Tucker Hermans
Thesis Faculty Supervisor

[redacted]

Mary Hall
Chair, Kahlert School of Computing

[redacted]

Tom Henderson
Honors Faculty Advisor

Monisha Pasupathi, PhD
Dean, Honors College

December 2024

ABSTRACT

Creating 3D representations of multi-object scenes is crucial for many robotic manipulation tasks. These representations must be inferred from noisy partial-view observations. This thesis focuses on the problem of building a 3D representation for multi-object tabletop scenes from a single RGBD image. A common approach is to use deep learning for this problem, however these approaches are not robust enough nor contain prinicipled uncertainty about object geometry. This thesis examines two Bayesian approaches. Both approaches rely on a hinge point representation. Each approach solves for a posterior distribution over object shapes. This allows both methods to capture principled uncertainty. Experimentally, each method is shown to be accurate, robust, and capture uncertainty. Experiments are performed qualitatively in the real world as well as quantitatively on procedurally generated scenes. A deep learning approach is used as a baseline for experiments on both methods.

TABLE OF CONTENTS

This thesis synthesizes work from the following papers which I am the first author on:

- “V-PRISM: Probabilistic Mapping of Unknown Tabletop Scenes” By **Herbert Wright**, Weiming Zhi, Matthew Johnson-Roberson, Tucker Hermans. Published in 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).
- “Robust Bayesian Scene Reconstruction by Leveraging Retrieval-Augmented Priors” By **Herbert Wright**, Weiming Zhi, Matthew Johnson-Roberson, Tucker Hermans. Submitted to for RA-L.

1 INTRODUCTION

Intelligent robots have the potential to do a lot of good for the world. Consider the task of caretaking: robots could significantly reduce the burden of caretakers *if* they had the capability to perform tasks such as cooking, cleaning, dressing, and other necessary chores. Such activities would require robots to *manipulate* their surrounding by interacting with different objects. Interactions like these require a detailed 3D understanding of the robot’s environment. For example, it would be hard for a robot to pick up most objects without having an explicit or implicit understanding of the object’s shape. While there has been some work on robotic systems that implicitly have 3D understanding, many robotic algorithms require explicitly representing the 3D geometry of the scene [1], [2].

In less-structured environments, the robot doesn’t necessarily know what objects it is looking at, so it must *infer* the shape of objects in a scene from partial-view observations. This thesis focuses on the problem of constructing a continuous 3D representation of a multi-object scene from a single RGBD image. This is a non-trivial problem. These observations can be noisy and contain significant *occlusion*. fig. 1 shows an example of such a scene. One common approach to the 3D reconstruction problem is to train a *neural network* to perform *shape completion*. Unfortunately, this approach can struggle for a couple of reasons. (1) Deep learning approaches can be brittle on real world scenes; (2) Deep learning approaches generally do not provide *principled uncertainty*.

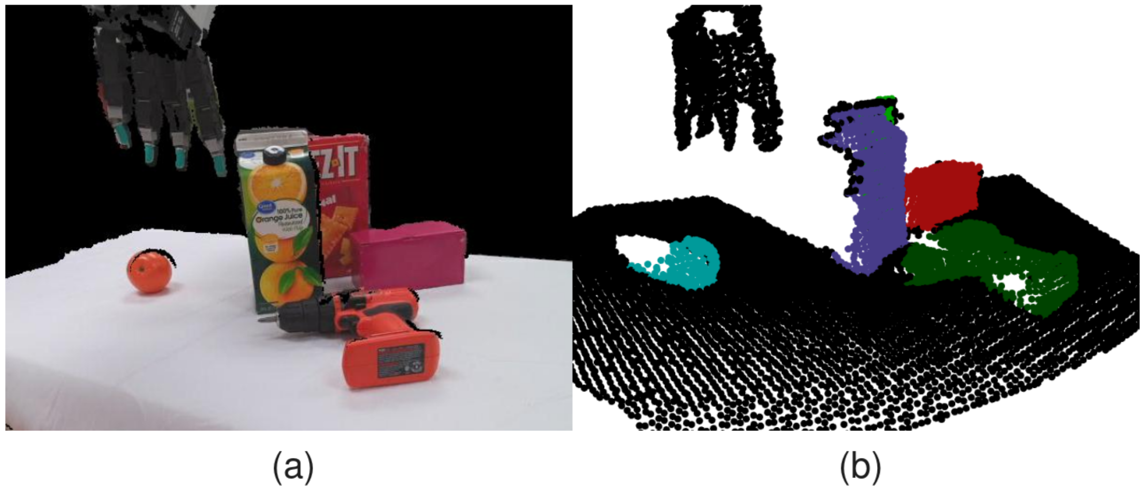

Figure 1: (a) An example RGB image of a tabletop scene. (b) A segmented point cloud of the same scene.

Uncertainty and uncertainty-awareness are important for the safe operation of robots. Uncertainty about object shape, specifically, can be utilized in downstream tasks such as grasping [3], [4]. Many deep learning approaches lack the ability to reason about this uncertainty. Neural networks have been known to predict incorrect labels with high confidence [5], [6]. This thesis focuses on two method that use a *Bayesian* approach to the 3D reconstruction problem.

This thesis covers two methods for building 3D representations for tabletop scenes. Both methods are able to capture principled uncertainty and are robust to many of the pitfalls of deep learning methods. Both methods also share another commonality: the use of a hinge point representation borrowed from the Hilbert maps [7] literature.

This thesis begins by giving an overview of some related works (sec. 2), then explains some preliminary mathematical concepts (sec. 3). Then, in sec. 4, the V-PRISM method is explained. Following this, the BRRP method is explained in sec. 5. Finally, sec. 6 contains a brief conclusion.

2 RELATED WORKS

3D Representations. There are many different ways of representing 3D geometry of a scene. In the mapping literature, techniques such as truncated signed distance functions [8] build voxelized representations of an environment. Hilbert maps [7], on the other hand, are a continuous occupancy map that takes the form of a linear function over some hinge point feature space. Hilbert map representations have also been extended to Bayesian Hilbert maps of various forms [11]. Neural implicit functions have also been used to represent continuous 3D geometry [14]. Other representations are built using differentiable rendering and combining multiple views. Neural radiance fields [15] are an example of this, in which a neural network maps 3D position to density and color. 3D Gaussian splatting [16] does a similar thing but with a set of Gaussians instead of a neural network. Foundation models have also been applied to this task. For example, [17] was applied to robotics in [18]. Other representation primitives have been studied, including super quadrics [19].

3D Reconstruction with Deep Learning. Many methods have been proposed as ways to leverage deep learning to reconstruct scenes or objects. While some methods aim to predict object shape from RGB data only [23], we instead focus on using depth measurements during reconstruction. DeepSDF [12] is a method to reconstruct an object by running inference-time optimization to recover a latent code for a neural implicit function. In the context of robotics, [24] extends DeepSDF to have uncertainty-awareness. Other work, such as occupancy networks [13] or PointSDF [25] try to directly predict such a latent code without inference-time optimization. Deep learning has also been leveraged to learn kernels, which are used to construct a continuous signed distance function [27]. Language is also used during reconstruction in [28] and [29]. Another work uses a voxel-to-voxel variational autoencoder conditioned on bounding boxes [30]. While these works typically focus on single object scenes, other work focuses on reconstruction scenes with multiple occluding objects. For example, [31] learns a reconstruction from a voxel representation with different channels to account for occlusion. Another method uses silhouettes to refine the initial reconstructions [32]. In practice, these deep learning approaches can struggle to reconstruct noisy scenes with multiple, highly occluded, unknown objects on real world depth cameras. Inaccurate segmentation can also be a problem for many of these methods as well.

3D Reconstruction without Deep Learning. There are also many approaches to perform probabilistic 3D reconstruction without deep learning. Some of such methods for reconstruction use informative prior information by assuming fixed classes of objects, such as 3DP3 [33]. Other methods use an uninformative prior, such as Gaussian process implicit surfaces (GPIS) [34]. There is an extension of GPIS to a slightly more informative prior [35]. The only priors that can be enforced are specifically spherical, ellipsoidal, cylidrical, or planar priors. Both the methods explored in this thesis reconstruct the scene without deep learning. V-PRISM (sec. 4) doesn't enforce an informative prior, but BRRP (sec. 5) uses pre-existing mesh datasets to create a prior for reconstruction.

Using Reconstructions in Manipulation. 3D reconstruction methods have seen extensive use in manipulation. In [25], PointSDF provides collision constraints during grasping. PointSDF is also utilized in [36], where tactile sensors are used along with the reconstruction during grasping. A learning-based voxel representation is used for grasping in [37]. Neural shape completion is also used during the anthropomorphic grasping pipeline proposed in [38]. GPIS is also a common representation for manipulation applications. Some recent work has utilized the uncertainty from GPIS representations during grasp selection [4]. The methods explored in this thesis provide principled uncertainty measurements that can potentially be similarly utilized in downstream manipulation tasks.

3 BACKGROUND

This section covers Hilbert maps and Bayesian Hilbert maps in sec. 3.1. Then, in sec. 3.2, Stein variational gradient descent is covered.

3.1 Hilbert Maps

Hilbert Maps: Introduced in [7], Hilbert maps are a method for building an occupancy map of an environment given depth observations. Hilbert maps represent the environment with a continuous function, defined by a linear function of a fixed feature transform. Typically, this feature transform is induced by a set of *hinge points*, $\{\mathbf{h}_1, ..., \mathbf{h}_H\} \subset \mathbb{R}^3$ and a translation-invariant kernel $k(d)$. The transform is then defined as:

$$\phi(\mathbf{x}) = [1, k(\mathbf{x} - \mathbf{h}_1)..., k(\mathbf{x} - \mathbf{h}_H)]^\top. \tag{1}$$

Usually, a Gaussian kernel is used and hinge points are placed in an evenly-spaced grid. An occupancy map can then be defined by a single weight vector, $\mathbf{w} \in \mathbb{R}^{H+1}$, as such:

$$m(\mathbf{x}) = \sigma(\mathbf{w}^\top \phi(\mathbf{x})).$$

To recover the weight vector corresponding to a given depth observation, *negative sampling* is performed along the unoccupied portions of the depth rays. These negative samples are assigned a label of unoccupied and the points at the end of the ray are labeled as occupied. Then, stochastic gradient descent (SGD) is performed on the binary cross entropy (BCE) of the negative samples and terminal points of the ray. The binary cross entropy measures the *likelihood* of the samples, and is defined as:

$$\text{BCE}(y, \mathbf{w}^\top \phi(\mathbf{x})) = \begin{cases} -\ln\left[\sigma\left(\mathbf{w}^\top \phi(\mathbf{x})\right)\right], & y = 1 \\ -\ln\left[1 - \sigma\left(\mathbf{w}^\top \phi(\mathbf{x})\right)\right], & y = 0 \end{cases} \tag{2}$$

fig. 2 shows an illustration of both the hinge point feature transform and the negative samples used during Hilbert map construction.

Bayesian Hilbert Maps: Hilbert Maps were extended to the Bayesian setting in [9]. In-

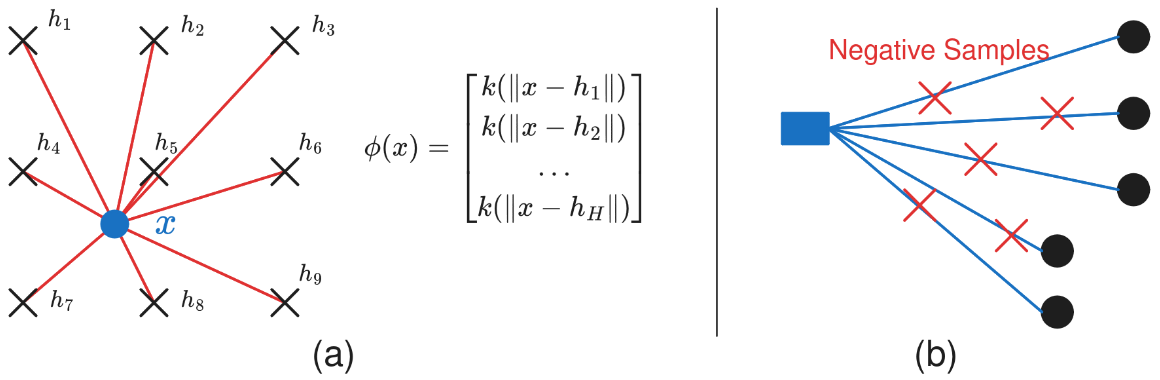
h_1
h_2
h_3
h_4
h_5
h_6
h_7
h_8
h_9
x
$\phi(x) = \begin{bmatrix} k(\|x-h_1\|) \\ k(\|x-h_2\|) \\ \cdots \\ k(\|x-h_H\|) \end{bmatrix}$
Negative Samples
(a)
(b)

Figure 2: (a) A *hinge point* feature transform induced by a set of hinge points is used by Hilbert maps [7]; (b) these maps are built by first sampling *negative samples* along the unoccupied portions of the camera ray.

stead of an individual weight vector, the weight is treated as a normally distributed random variable, $\mathbf{w} \sim P(\mathbf{w})$. Variational Bayesian logistic regression as described in [39] is then performed over data $D = \{(\phi(\mathbf{x}_i), y_i)\}_{i\in[n]}$ in order to obtain the approximate posterior distribution:

$$\hat{P}(\mathbf{w}|D) \propto Q(D|\mathbf{w};\xi)P(\mathbf{w}) \approx P(D|\mathbf{w})P(\mathbf{w}),$$

where the variational parameter ξ is introduced. The method relies on an EM algorithm that alternates between calculating the posterior $\hat{P}(\mathbf{w}|D) = \mathcal{N}(\hat{\mu}, \hat{\Sigma})$ from an approximate likelihood function and obtaining a better likelihood approximation. The specific approximation used for the likelihood takes the form of a normal distribution, and ensures that the approximated likelihood is conjugate to a normal prior $P(\mathbf{w}) = \mathcal{N}(\bar{\mu}, \bar{\Sigma})$.

Once the posterior weight distribution is obtained, the map m is defined by the expectation:

$$m(\mathbf{x}) = \mathbb{E}_{\mathbf{w}}[\sigma(\mathbf{w}^\top \phi(\mathbf{x}))].$$

Because there is not an analytic solution for this expectation, approximations are used. The most common approximation is

$$\mathbb{E}_{\mathbf{w}}[\sigma(\mathbf{w}^\top \phi(\mathbf{x}))] \approx \sigma\left(\frac{\mathbb{E}_{\mathbf{w}}[\mathbf{w}^\top \phi(\mathbf{x})]}{\sqrt{1+\frac{\pi}{8}\mathrm{Var}(\mathbf{w}^\top \phi(\mathbf{x}))}}\right), \tag{3}$$

which is easily obtained for any $\mathbf{w}$ following a normal distribution.

Extensions of BHMs include Bayesian treatment of kernel parameters and hinge point placement [10], fusing two BHMs [11], and mapping environments with moving actors [40].

3.2 Stein Variational Gradient Descent

Stein Variational Gradient Descent (SVGD) [41] is an algorithm for variational inference that closely resembles gradient descent. The general problem of variational inference is to find a distribution $q^* \in \mathcal{Q}$ that is close to some target distribution p. Usually, this takes the form of an optimization problem over the Kullback-Leibler (KL) divergence:

$$q^* = \arg\min_{q\in\mathcal{Q}} \mathbb{KL}(q\|p).$$

SVGD aims to iteratively transform q in descent directions of the KL divergence in a d-dimensional reproducing kernel Hilbert space (RKHS), $\mathcal{H}^d$. Because this Hilbert space is a space of functions, a descent direction requires deriving the *functional gradient* of our KL divergence objective.

Theorem 1: From [41]. Let $T(\mathbf{x}) = \mathbf{x} + f(\mathbf{x})$, where $f \in \mathcal{H}^d$ and $q_{[T]}$ is the density of random variable $\mathbf{z} = T(\mathbf{x})$ when $\mathbf{x} \sim q$. Then

$$\nabla_f \mathbb{KL}(q_{[T]}\|p)|_{f=0} = -g^*_{q,p},$$

where $g^*_{q,p} = \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}[k(\mathbf{x},\cdot)\nabla_{\mathbf{x}} \ln p(\mathbf{x}) + \nabla_{\mathbf{x}} k(\mathbf{x},\cdot)]$.

In SVGD, q is approximated by a set of particles $\mathbf{x}_1^{(0)}, ..., \mathbf{x}_P^{(0)} \sim q(\mathbf{x})$. This can be used to approximate the gradient in Theorem 1 with $\hat{g}^*$:

$$\hat{g}^*(\mathbf{x}) = \frac{1}{P}\sum_{i=1}^{P} k(\mathbf{x}_i, \mathbf{x})\nabla_{\mathbf{x_i}} \ln p(\mathbf{x}_i) + \nabla_{\mathbf{x_i}} k(\mathbf{x}, \cdot). \tag{4}$$

The particles can then be iteratively updated according to $\hat{g}^*$ in eq. 4 with:

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \epsilon \hat{g}^*(\mathbf{x}_i^{(t)})$$

The result of these iterations is that the set of particles converges to an approximation of the target distribution p. Importantly, eq. 4 only relies on the gradient of the log of p, which means we can perform variational inference to an unnormalized distribution. Such unnormalized distributions are commonplace in many Bayesian inference problems, including ours.

4 VOLUMETRIC, PROBABILISTIC, AND ROBUST SEGMENTATION MAPS

This section introduces the V-PRISM method, which stands for **V**olumetric, **P**robabilistic, and **R**obust **I**nstance **S**egmentation **M**aps. In sec. 4.1, I provide a formulation for the multi-object reconstruction problem and give a brief overview of the method. The method is further explained in sec. 4.2 and sec. 4.3. Then, sec. 4.4 details experiments on the V-PRISM method.

4.1 Overview of Method

Problem Formulation. Instead of predicting an occupancy map for each object, we phrase our problem as a multi-class mapping problem. This ensures that each point in space can only be occupied by a single object. Without this constraint, reconstructions of objects can intersect each other as shown in fig. 3. Formally, we receive observations $\{(\mathbf{x}_i, y_i)\}_{i\in[n]}$ where $\mathbf{x}_i \in \mathbb{R}^d$ corresponds to an observed point with segmented class $y_i \in [c]$. We assume $y_i = 1$ denotes $\mathbf{x}_i$ being segmented to no specific object and is part of the background or

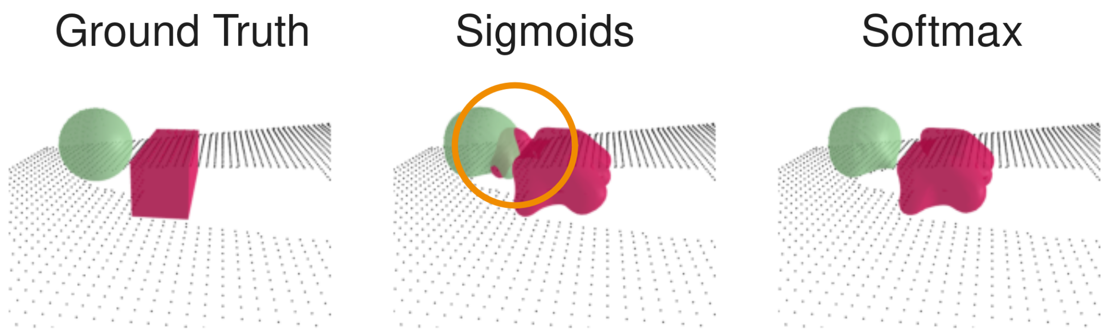

Figure 3: Running a separate sigmoid model per object can cause unwanted intersections between the reconstructions (circled). Our multi-class formulation uses a softmax model that avoids this problem

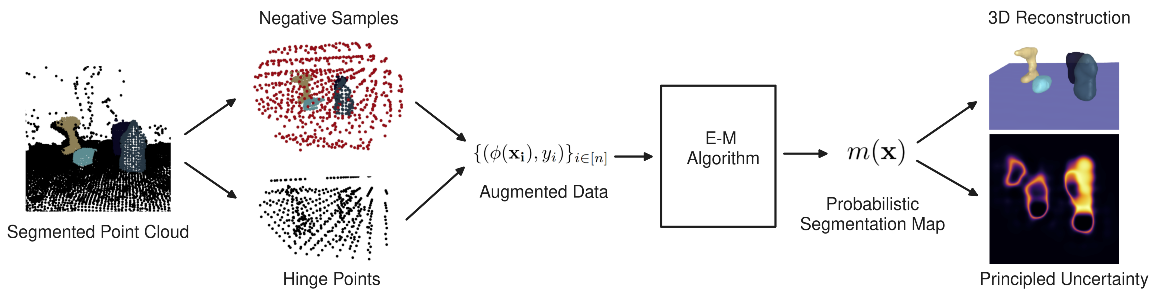

Figure 4: Overview of V-PRISM method

table. We also assume that these observations came from a camera with a known location $\mathbf{o} \in \mathbb{R}^d$. The goal is to build a map function $m : \mathbb{R}^d \to [0, 1]^c$ such that $m(\mathbf{x})$ corresponds to the probability distribution over classes that the point $\mathbf{x}$ could belong to.

We would like our map to satisfy that $m(\mathbf{x}_i) \approx \mathbf{e}_{y_i}$ for all i, where $\mathbf{e}_{y_i}$ is the one hot encoding of y_i. We can infer that for any $\mathbf{x}_i$, because the camera ray started at $\mathbf{o}$ and terminated at $\mathbf{x}_i$, all points in between are unoccupied. We would like our map to reflect this realization. This forms the basis for the negative sampling performed in [9]. We will also assume that objects in the scene are resting on or above a planar surface. While this typically means a table, our method is agnostic to the type of surface.

Method Description. Our method builds a map $m(\mathbf{x})$ from segmented camera depth observations of a multi-object scene through two main steps. A high level overview is displayed in fig. 4. First, negative sampling is performed as described in sec. 4.3, where additional points are added to the observed ones in order to form a new labelled point cloud. During this step, the RANSAC [42] algorithm is run in order to recover the surface plane the objects are resting on. The points are also subsampled in order to increase efficiency. We then generate a set of hinge points that are used to construct a feature transform according to Equation (1). This transform, along with our sampled points, creates a set of augmented data.

Once we have our transformed data, we perform Bayesian multi-class regression over the data with an expectation maximization (EM) algorithm. The specific technique makes use of mathematical ideas from [43]. The full EM algorithm and model are explored in sec. 4.2. Efficiently evaluating $m(\mathbf{x})$ for query $\mathbf{x}$ values is also covered in sec. 4.2, where we make use of an approximation proposed in [44]. The segmentation map produced maps each point in 3D space to a distribution over c classes, where one class denotes not belonging to an object and the other $c-1$ classes denote the segmented objects observed.

Once we have our map, we can use it to evaluate how likely different points are to be in occupied by different objects. This is useful in many motion planning algorithms in order to minimize unwanted collisions. We can also reconstruct the meshes of each object by running the marching cubes algorithm [45]. These meshes can be used to create a signed distance function, simulate physics, or to visualize the scene. Our map also encodes principled uncertainty about the geometry of the scene which can be used for active inference.

4.2 Softmax EM Algorithm

Training. To create a Bayesian multi-class map, we consider using a weight matrix $\mathbf{W} \in \mathbb{R}^{c \times m}$ where each row is normally distributed, giving the following likelihood function:

$$P(y = k|\mathbf{W}, \mathbf{x}) = \text{softmax}(\mathbf{W}\phi(\mathbf{x}))_k,$$

where the softmax function is defined as

$$\text{softmax}(\mathbf{W}\phi(\mathbf{x}))_k = \frac{\exp(\mathbf{W}\phi(\mathbf{x})_k)}{\sum_{i=1}^{c}\exp(\mathbf{W}\phi(\mathbf{x})_i)}.$$

Because a conjugate prior for the softmax likelihood doesn't exist, we must use variational inference to find a posterior Gaussian distribution. In our case, we will maximize a lower bound on the likelihood. A useful inequality for this is given in [43], and is stated in the following theorem:

Theorem 2: From [43]. Let $\mathbf{z} \in \mathbb{R}^c$, $\alpha \in \mathbb{R}$, and $\xi \in \mathbb{R}^c_+$. Then the following inequality holds:

$$\ln\sum_{k=0}^{c}\exp(\mathbf{z}_k) \leq \alpha + \sum_{k=0}^{c}\frac{\mathbf{z}_k - \alpha - \xi_k}{2}$$
$$+\lambda(\xi_k)((\mathbf{z}_k - \alpha)^2 - \xi_k^2) + \ln(1+\exp(\xi_k)),$$

where $\lambda(\xi_k) = ((1+\exp(-\xi_k))^{-1} - (1/2))/2\xi_k$.

Applying Theorem 2 to $\mathbf{z} = \mathbf{W}\phi(\mathbf{x})$, we can bound the likelihood by introducing the two variational parameters α and ξ with the inequality,

$$\ln P(y = k|\mathbf{W},\mathbf{x}) \geq \ln Q(y = k|\mathbf{W},\mathbf{x};\alpha,\xi).$$

We can maximize this lower bound and use it as an approximation to the true likelihood by solving the following:

$$\arg\max_{\alpha,\xi}\mathbb{E}_{\mathbf{W}}\left[\ln Q(y = y_i|\mathbf{x}_i,\mathbf{W};\alpha,\xi)\right].$$

This can be analytically solved for $\mathbf{W}_k \sim \mathcal{N}(\mu_k,\Sigma_k)$, yielding the following optimal values

Algorithm 1 V-PRISM

```
Input:
Observed, segmented points $o = \{(\mathbf{x}_i, y_i)\}_{i\in[n']}$
Prior means $\{\bar{\mu}_k\}_{k\in[c]}$ and covariances $\{\bar{\Sigma}_k\}_{k\in[c]}$
 1: $\mathcal{D} \leftarrow$ NegativeSample$(o)$
 2: $\phi \leftarrow$ HingePointTransform$(o)$
 3: $\xi_{i,k} \leftarrow 1$ for $i \in [m], k \in [c]$
 4: $\alpha_i \leftarrow 0$ for $i \in [m]$
 5: for $p$ iterations do
 6:     $\hat{\Sigma}^{-1} \leftarrow \bar{\Sigma}^{-1} + 2\sum_i |\lambda(\xi_i)|\phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^\top$
 7:     $\hat{\mu}_k \leftarrow \hat{\Sigma}\left(\bar{\Sigma}^{-1}\bar{\mu} + \sum_i(y_i - \frac{1}{2} + 2\alpha_i\lambda(\xi_{i,k}))\phi(\mathbf{x}_i)\right)$
 8:     $\alpha_i \leftarrow$ UpdateAlpha$(\xi_i, \mathbf{x}_i, \hat{\mu}, \hat{\Sigma})$
 9:     $\xi_{i,k} \leftarrow$ UpdateXi$(\alpha_i, \mathbf{x}_i, \hat{\mu}, \hat{\Sigma})$
10: end for
11: return $\hat{\mu}, \hat{\Sigma}$
```

Figure 5: Psuedo-code for the V-PRISM algorithm

found in [43]:

$$\alpha_i = \frac{\frac{1}{2}(\frac{c}{2} - 1) + \sum_{k=1}^{c} \lambda(\xi_k)\mu_k^\top \phi(\mathbf{x}_i)}{\sum_{k=1}^{c} \lambda(\xi_k)}, \tag{5}$$

$$\xi_{i,k}^2 = \phi(\mathbf{x}_i)^\top \Sigma_k \phi(\mathbf{x}_i) + (\mu_k^\top \phi(\mathbf{x}_i))^2 + \alpha_i^2 - 2\alpha_i \mu_k^\top \phi(\mathbf{x}_i). \tag{6}$$

Due to the inequality used, $P(y = k|\mathbf{W}, \mathbf{x}; \alpha, \xi)$ is normally distributed for any α, ξ and will be conjugate to our prior weight distribution. Thus, we have a closed-form for the approximate posterior distribution, $P(\mathbf{W}|y = k, \mathbf{x}) = \mathcal{N}(\hat{\mu}, \hat{\Sigma})$. The update equations mirror those found in [43] and are as follows:

$$\hat{\Sigma}_k^{-1} = \bar{\Sigma}^{-1} + 2\sum_{i=1}^{n} \lambda(\xi_{i,k})\phi(\mathbf{x}_i)\phi(\mathbf{x}_i)^\top \tag{7}$$

$$\hat{\mu}_k = \hat{\Sigma}_k \left[\bar{\Sigma}_k^{-1}\bar{\mu}_k + \sum_{i=1}^{n}\left(y_{i,k} - \frac{1}{2} + 2\alpha_i\lambda(\xi_{i,k})\right)\phi(\mathbf{x}_i)\right]. \tag{8}$$

We can use eq. 5, eq. 6, eq. 7, and eq. 8 to create an EM algorithm to iterate between calculating our posterior distribution and optimizing our variational parameters, shown in fig. 5. The size of $\hat{\Sigma}_k$ scale quadratically with the feature dimension.

Inference. In order to make predictions about new points we need to evaluate the following expectation:

$$\hat{P}(y = k|\mathbf{x}) = \mathbb{E}_{\mathbf{W}}\left[\text{softmax}(\mathbf{W}\phi(\mathbf{x}))\right]_k . \tag{9}$$

There is not a closed form solution to this expectation, so we must approximate it. While we could use sampling to estimate the expectation, we instead use a more computationally efficient approximation.

As described in [44], we can write the softmax in terms of the sum of sigmoidal terms with the following equality:

$$\text{softmax}(\mathbf{a})_k = \frac{1}{2 - c + \sum_{i \neq k} \sigma(\mathbf{a}_k - \mathbf{a}_i)^{-1}},$$

where c is the number of classes. This is then used as motivation for the approximating the expectation with

$$\mathbb{E}_{\mathbf{W}}\left[\text{softmax}(\mathbf{W}\phi(\mathbf{x}))\right]_k \approx \frac{1}{2 - c + \sum_{i \neq k} \mathbb{E}[\sigma(\tilde{\mathbf{z}}_i)]^{-1}},$$

with $\tilde{\mathbf{z}}_i = [\mathbf{W}\phi(\mathbf{x})]_k - [\mathbf{W}\phi(\mathbf{x})]_i$. When combined with the sigmoidal approximation in Equation (3), this becomes an easily computable approximation to Equation (9).

4.3 Negative Sampling

Similar to many mapping methods, V-PRISM requires sampling negative unoccupied points along depth camera rays. The traditional negative sampling used, mentioned in sec. 3.1, is meant for mapping environments where the robot is in an enclosed space and each camera ray is detecting a wall or sufficiently large object. This sampling performs poorly when the goal is to map a relatively small object resting on a tabletop or other surface. To fully utilize

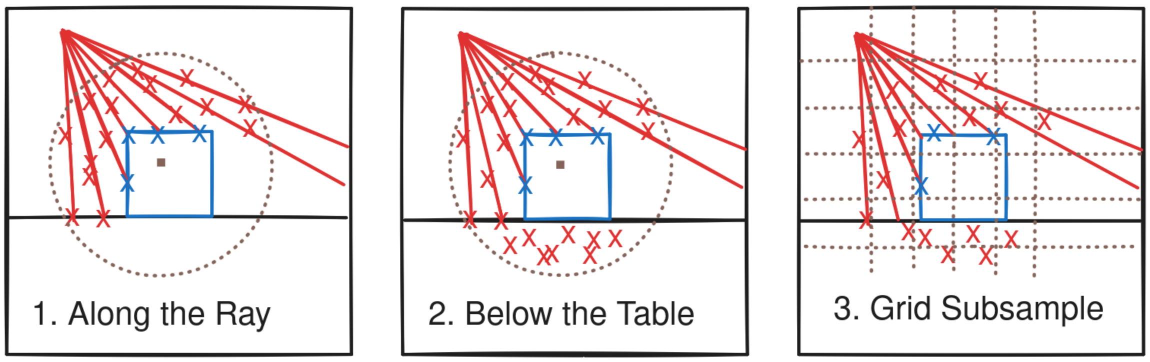

Figure 6: Overview of our sampling method. 1. We perform stratified sampling along camera rays within r_{obj} of the object. 2. Points are sampled below the table within r_{obj} of the object. 3. Grid subsampling is performed.

the tabletop structure within the environment, we propose a new negative sampling method designed for object-centric mapping. Our sampling method rests on two main realizations:

1. Along the ray, negative samples are most useful when near known objects.
2. Points below a surface plane cannot be occupied by objects resting entirely on or above that surface.

We assume we have a segmented point cloud of the scene $\{(\mathbf{x}_i, y_i)\}_{i\in[n']}$ where each y_i corresponds to the segmentation label of the respective $\mathbf{x}_i$. We also assume a known position of the camera $\mathbf{o}$. Our sampling method begins by finding the center of the smallest axis-aligned bounding box that contains all of the segmented points for each individual object in the scene. We denote these centers with $\mathbf{o}_k$. We then perform stratified uniform sampling along each ray, only keeping points that are within r_{obj} distance from at least one $\mathbf{o}_k$. Sampled points within the desired radius of a center are labeled as unoccupied and added to the collection of points for the algorithm.

Next, we run RANSAC [42] on the observed point cloud to recover the table plane. Once we have the plane, we uniformly randomly sample points within r_{obj} from each object center and keep any such points that fall below the plane. These points are labelled as unoccupied and added to our collection.

Finally, we perform grid subsampling as described in [46] with each label in parallel in order to reduce the number of points our algorithm is fed. In practice, we choose different resolutions to subsample empty points and points on object surfaces. This can dramatically increase the efficiency of our method by removing redundant points. The entire negative sampling process is shown in fig. 6.

The resulting points are then transformed to construct our set of augmented data. The transform used is induced by a set of hinge points according to Equation (1). In practice, we choose a set of hinge points consisting of a fixed grid around the scene as well as a fixed number of random points sampled from the surface points of each object.

4.4 Experiments

We perform experiments aimed to answer the following questions: (1) Does our method result in accurate reconstructions? (2) Does our sampling method improve map quality for object-centric mapping? (3) Is our method robust to unknown, noisy scenes? 4. Does our map accurately capture uncertainty about the scene geometry? We implement V-PRISM in PyTorch and run our algorithm on an NVIDIA GeForce RTX 2070 GPU.

Baselines: We compare our method to two different baselines. The first is a voxel-based heuristic that labels observed unoccupied voxels as unoccupied, observed occupied voxels as their corresponding segmentation label, and unobserved voxels with the same label as the nearest observed voxel. To prevent incorrect predictions below the table plane, we also run RANSAC during our baseline and label all voxels under the plane as unoccupied. We refer to this approach as the **Voxel** baseline. The second baseline is a learning-based approach using a state of the art neural network architecture for continuous object reconstructions in robotics. We take the PointSDF architecture from [25] and replace the final activation with a sigmoid function to predict occupancy probabilities. We train this model on a dataset of scenes generated in simulation. The scenes are composed of a subset of the ShapeNet [47] dataset. Training it on these scenes instead of the original dataset PointSDF was trained on

Method	ShapeNet Scenes IoU ↑	ShapeNet Scenes Chamfer ↓	YCB Scenes IoU ↑	YCB Scenes Chamfer ↓	Objaverse Scenes IoU ↑	Objaverse Scenes Chamfer ↓
Voxel	0.198	0.014	0.324	0.018	0.336	0.024
PointSDF	**0.360**	**0.010**	0.460	0.015	0.347	0.025
V-PRISM	0.309	0.011	**0.500**	**0.012**	**0.464**	**0.018**

Table 1: Quantitative experiments comparing our method to two baseline methods on procedurally generated scenes from benchmark mesh datasets.

allows it to better function under occlusion and different scales. We refer to this baseline as **PointSDF**.

Metrics: We use two main metrics for comparison: **intersection over union (IoU)** and **Chamfer distance**. IoU is calculated by evaluating points in a fixed grid around each object. Chamfer distance is calculated by first reconstructing the predicted mesh by running the marching cubes algorithm [45] on a level set of $\hat{P}(y = 1|x) = \tau$ for a chosen τ of the prediction function. Then, points are sampled from both the predicted mesh and ground truth mesh and the Chamfer distance is calculated between these two point clouds.

Procedurally Generated Scenes: we evaluate our method against the two baseline methods on procedurally generated scenes, from large object datasets. We generate a scene by randomly picking a mesh and placing it at a random pose within predefined bounds with a random scale. We draw meshes from the ShapeNet [47], YCB [48], and Objaverse [49] datasets. We generate 100 scenes for each dataset with up to 10 objects in each scene. Objects are placed relatively close together in order to ensure significant occlusion in the scenes. Once the poses have been selected, we simulate physics for a fixed period of time to ensure objects can come to rest.

Our first experiment on simulated scenes compares our method with the two baselines. Similar to [13], we use a level set other than $\tau = 0.5$ for constructing the mesh with the neural network. We found $\tau = 0.3$ to provide the best reconstructions for our version of PointSDF. For other methods, we use $\tau = 0.5$. We report the IoU and Chamfer distance in Table 1. PointSDF outperforms other methods on the ShapeNet scenes, where the meshes

Method	ShapeNet Scenes IoU ↑	ShapeNet Scenes Chamfer ↓	YCB Scenes IoU ↑	YCB Scenes Chamfer ↓	Objaverse Scenes IoU ↑	Objaverse Scenes Chamfer ↓
w/ BHM Sampling	0.156	0.031	0.313	0.030	0.326	0.035
V-PRISM (ours)	**0.309**	**0.011**	**0.500**	**0.012**	**0.464**	**0.018**
w/o Under the Table	0.291	0.019	**0.500**	0.014	0.439	0.024
w/o Stratified Sampling	0.145	0.024	0.294	0.023	0.291	0.029

Table 2: Ablation experiments on our negative sampling method.

are drawn from the same mesh dataset that it was trained on. On other datasets, our method outperforms PointSDF. This aligns with other work demonstrating that neural networks perform worse the further from the training distribution you get. Because our method has no reliance on a training distribution, it shows consistency across all datasets. Both our method and PointSDF consistently outperform the voxel baseline on most datasets and metrics. The only exception is Chamfer distance on Objaverse scenes, where the voxel baseline outperforms PointSDF. The performance of our method relative to our baselines indicate that our method results in accurate reconstructions

Our second experiment on simulated scenes ablates our negative sampling method. We observe the effect of removing sampling under the table plane and removing the stratified sampling along the ray. In order to remove the stratified sampling, we replace it with taking discrete, fixed steps along each ray instead. We also compare against the original BHM sampling method explained in [9], where there negative samples are drawn randomly along the whole ray instead of near objects. This is labeled as **BHM Sampling**. The IoU and Chamfer distance are reported in Table 2. Our negative sampling method outperforms the others on each dataset and metric. This implies that our proposed sampling method does improve reconstruction quality when compared to alternatives.

The hyperparameters used for the simulated experiments are shown in Table 3. These were kept constant across all procedurally generated datasets and corresponding experiments.

Real World Scenes: We evaluate our method by qualitatively comparing reconstructions on real world scenes. We use a Intel RealSense D415C camera to obtain point clouds of

Hyperparameters (Learning)	Value	Hyperparameters (Sampling)	Value (cm)
kernel type	Gaussian	grid length	5.0
kernel γ	1000	sampling r_{obj}	25.0
surface hinge pts.	32	subsample res. (objects)	1.0
iterations	3	subsample res. (empty)	1.5

Table 3: Hyperparameters for experiments on procedurally generated scenes.

tabletop scenes. In order to get accurate segmentations of the scene, we use the Segment Anything Model (SAM) [50]. We compute reconstructions on five scenes consisting of multiple objects. Each map of these scenes took between 2 and 5 seconds to compute. We compare our method to PointSDF. The qualitative comparison can be seen in fig. 7. Because these scenes are significantly more noisy than simulated scenes, PointSDF struggles to coherently reconstruct the scene. In contrast, our method is capable of producing quality reconstructions even with very noisy input point clouds. This suggests that our method is capable of bridging the sim to real gap and is robust to unknown, noisy scenes.

Uncertainty: To show how our model captures uncertainty about the scene, we need a way to quantify uncertainty. We use the entropy of our map at each point in space as a measurement of uncertainty:

$$H_m(\mathbf{x}) = -\sum_{k=1}^{c} \hat{P}(y = c|\mathbf{x}) \ln \hat{P}(y = c|\mathbf{x}).$$

This is maximized when the model predicts a uniform distribution over classes and minimized when the model predicts a single class with a probability of 1.

We compare our method with an alternate non-Bayesian version of our method, where we train a single weight vector with stochastic gradient descent (SGD) instead of the EM algorithm, to minimize the negative log-likelihood of our augmented data.

To visualize this uncertainty, we calculate this uncertainty over a 2D slice from each of our 5 real world scenes. The heat maps for each slice can be seen in fig. 8. Qualitatively, we

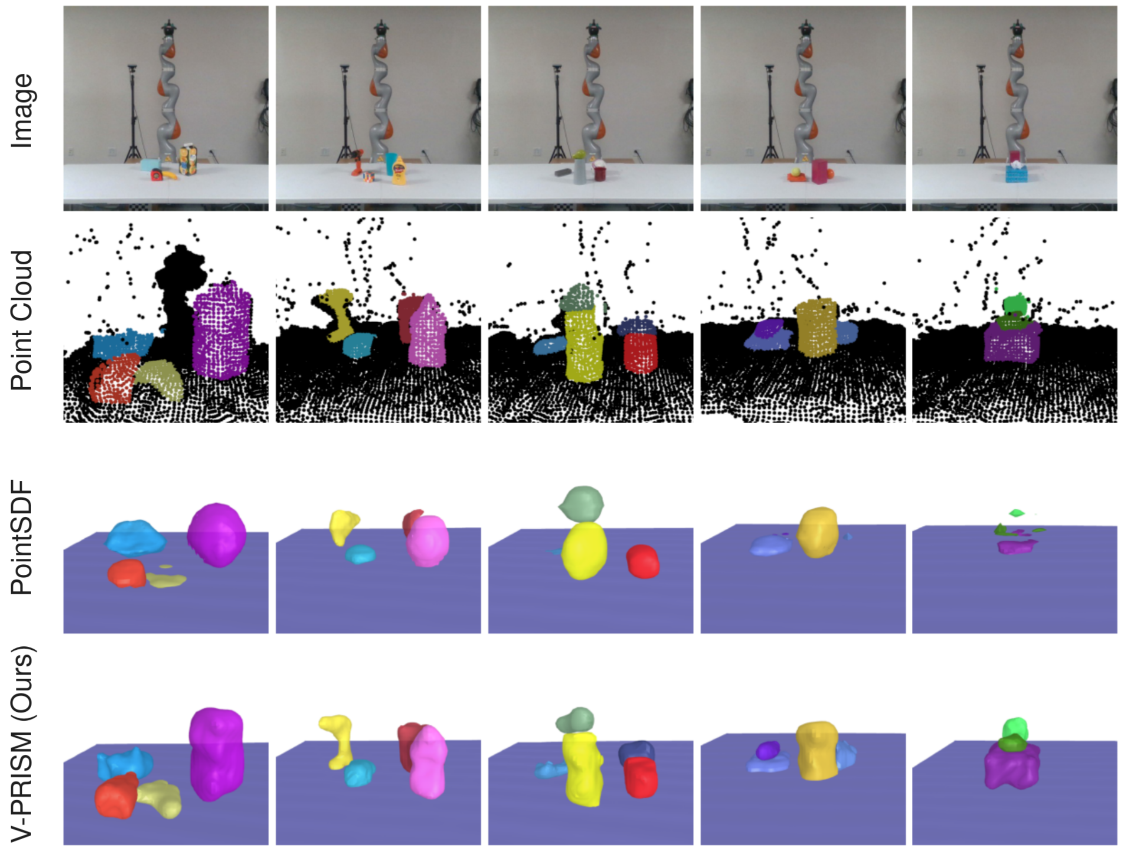

Figure 7: Qualitative comparisons with PointSDF reconstructions. **First row:** RGB images. **Second row:** the segmented point cloud used as input. **Third row:** PointSDF reconstructions. **Last row:** V-PRISM's (our method) reconstructions. V-PRISM results in quality reconstructions on noisy scenes.

can see that our method obtains high uncertainty values in occluded sections of the scene. This contrasts to the non-probabilistic model that does not accurately capture uncertainty about occluded regions. The heat maps showing occlusion-aware uncertainty suggest our model captures principled and accurate uncertainty measures.

5 BAYESIAN RECONSTRUCTION WITH RETRIEVAL-AUGMENTED PRIORS

5.1 Retrieval-Augmented Priors

Retrieval-augmented generation [51] was originally introduced in the context of improving language generation. The work has served as inspiration for an approach to affordance-

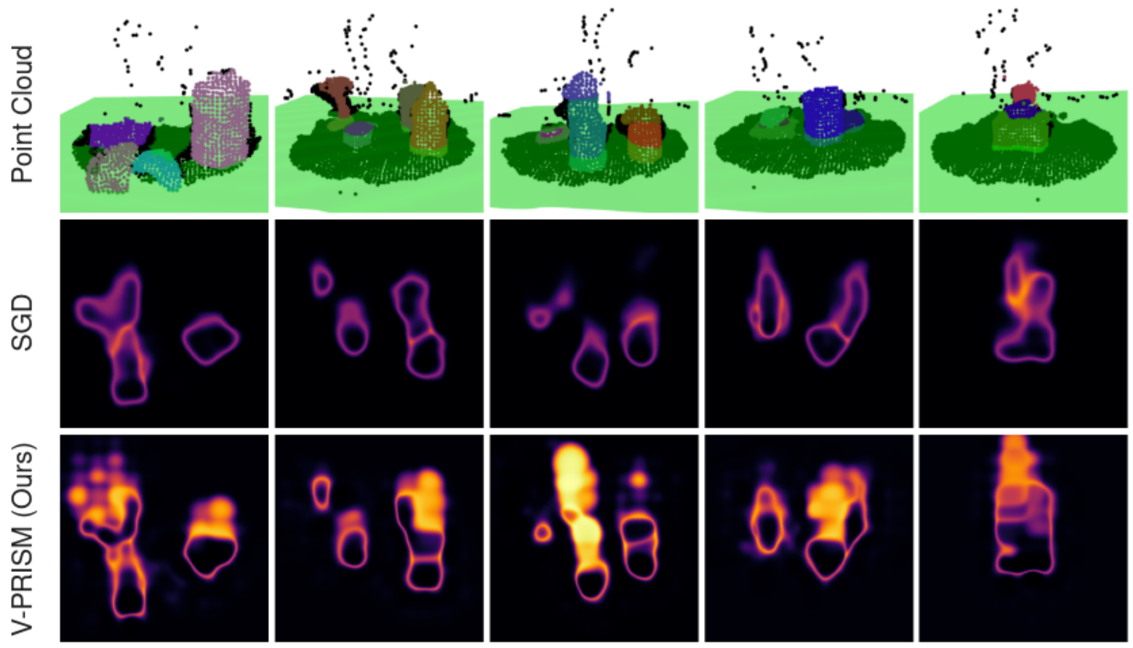

Figure 8: Qualitative comparison of uncertainty. **Top row:** the observed point cloud with a green plane corresponding to the 2D slice where the heat maps were calculated. We compare a non-probabilistic variant of V-PRISM trained with gradient descent (**middle row**) and our method (**bottom row**). In the heat maps, the bottom is closer to the camera and the top is farther from the camera. Lighter areas correspond to more uncertainty. Our method predicts high uncertainty in occluded areas of the scene.

prediction in [52]. In our case, we draw inspiration from retrieval-augmented generation, but we use the retrieved results to improve efficiency in certain *explicit* formulations for prior density functions during Bayesian inference.

To motivate retrieval-augmented priors, consider the problem of Bayesian inference with a mixture model acting as the prior distribution. Given some data, we would like to infer a posterior distribution over hypotheses. If we have a mixture model as a prior distribution, then:

$$P(H|D) \propto P(D|H) \sum_{c=1}^{C} P(H|c). \tag{10}$$

If our prior distribution has a lot of components, it may be inefficient to fully evaluate. This could be a serious problem for algorithms like SVGD, which requires iteratively computing the gradient of both the likelihood and prior. Inspired by [51], the insight behind retrieval-

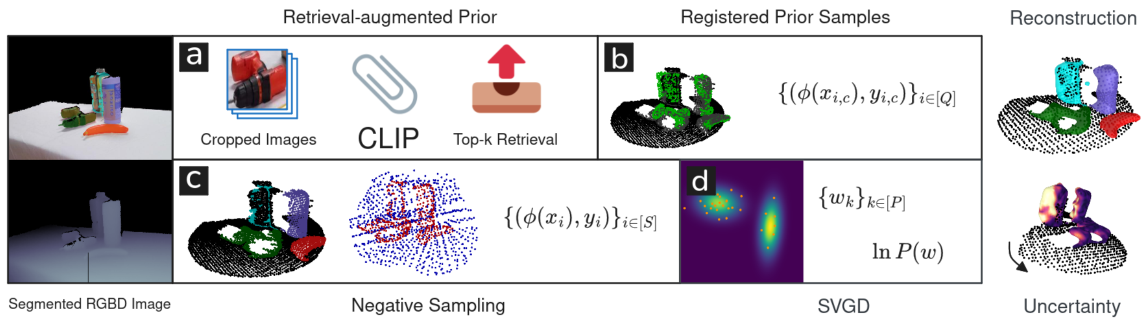

Figure 9: Overview of BRRP method. We begin with a segmented RGBD image and (a) feed cropped images of each segment into CLIP to get object probabilities. Then, we retrieve and (b) register the top-k objects in the prior. This gives us a set of registered prior samples. We also (c) compute negative samples based on the observed segmented point cloud. Finally, (d) we run SVGD optimization to recover a posterior distribution over Hilbert map weights. We can use this distribution to both reconstruct the scene as well as measure uncertainty.

augmented priors is to determine which subset of the prior distribution components to retrieve and use given some detection result R. Conditioning on this detection result, we have a new posterior distribution, $P(H|D, R)$. Making an independence assumption,

$$P(H|D,R) \propto P(D|H) \cdot \mathbb{E}_{c\sim P(c|R)}[P(H|c)].$$

Comparing to eq. 10, the expectation now replaces the true prior. Then, we can use a top-k approximation for the expectation:

$$P(H|D,R) \propto P(D|H) \sum_{c\in\text{topk}} P(H|c)P(c|R) \tag{11}$$

This means that we only need to evaluate a subset of the prior distribution components.

5.2 Bayesian Scene Reconstruction

Our method takes a single RGBD image and produces reconstructions for each object in the scene. We treat the problem as a Bayesian inference problem over an observation described by negative samples. We incorporate prior information on the shape of the object by leveraging retrieval-augmented priors introduced in sec. 5.1. We use CLIP [53] to determine which objects to retrieve and define our object-specific priors by a registered set of pre-computed samples from the stored mesh. We then use SVGD to optimize for a set of samples over map weights. We can generate predicted reconstructions by taking the expected occupancy over our weights for a given location. fig. 9 shows a visual overview of our method.

Negative Samples as Reconstruction Priors: We want to leverage existing mesh assets as our priors during Bayesian reconstruction. We define our prior as a mixture over different objects, $c_1, ..., c_C$. Because there is not a direct way to convert a mesh into a Hilbert map, we instead *sample* points $\tilde{\mathbf{x}}_{c,1}, ..., \tilde{\mathbf{x}}_{c,Q} \in \mathbb{R}^3$ around each object c's mesh. We refer to these samples as the *prior samples*. We give them labels $\tilde{y}_{c,1}, ..., \tilde{y}_{c,1} \in \{-1, 1\}$ determined by whether they are outside or inside the mesh. Then we simply define our prior using this data combined with a Gaussian prior over weight norm:

$$\begin{aligned} P(\mathbf{w}|c) :=& P(\{\tilde{y}_{c,i}\}|\{\tilde{\mathbf{x}}_{c,i}\}, \mathbf{w})P(\mathbf{w}) && (12) \\ &\propto \exp(\lambda\|w\|^2)\prod_{i=1}^{Q}\exp\left(\text{BCE}(\tilde{y}_{c,i}, \mathbf{w}^\top\phi(\tilde{\mathbf{x}}_{c,i})\right), && (13) \end{aligned}$$

where BCE is the same as in eq. 2.

In order to enforce pose-invariance, we first register a small stored point cloud of the object to the observed points and then transform the prior samples to this reference frame. In practice we use RANSAC [42] and the FPFH features from [54] to perform registration. In order to also have scale invariance, we do a linear scan over 10 different scales and select the the scale that resulted in the most inlier pairs from the registration.

Retrieval-Augmented Priors for Hilbert Maps: Because it would be inefficient to register all meshes that are part of the prior mixture model, we propose using the retrieval-augmented prior approach introduced in sec. 5.1. In order to determine which objects to use, we need to compute $P(c|R)$ from Equation (11). In our case, we use CLIP [53] as a zero-shot classifier for our different objects. For each object in our prior, we store a small textual description of the object. These descriptions are then used as classes for CLIP to classify each segmented object. In order to make sure CLIP knows which object we are targeting, we crop the RGB image to fit the predicted segmentation of each object and feed the cropped images as input into CLIP.

Once we have the probability of each object, we retrieve and register the stored point clouds of the top-k objects. After registration, we retrieve the prior samples corresponding to these objects to define our prior according to Equation (12).

Negative Sampling: We adopt the negative sampling method introduced in sec. 4.3. The negative sampling method makes the assumption that all objects are lying on or above a planar surface. We begin by labeling the points segmented to each object as occupied for that object. Next, we perform stratified sampling along each camera ray near each object to recover a set of negatively sampled points, labeled as unoccupied. Then, we use RANSAC over points not segmented to any object to recover the flat surface all objects are resting on. This plane is used to randomly sample points in a sphere underneath each object that are near the object. We also label these points as occupied. Finally, we use grid subsampling from [46] to reduce the number of points and increase uniformity of sampled points. We refer to these points and labels as *observed samples* and denote them as $\{\mathbf{x}_i\}_{i\in[S]}, \{y_i\}_{i\in[S]}$. The entire negative sampling process can be easily parallelized for efficient computation.

SVGD Reconstruction: Once we have retrieved our prior samples and computed our observed samples, we can perform optimization-based reconstruction with SVGD. Given both sets of samples and our prior definition from Equation (12), we have the following posterior distribution:

$$P(\{y_{c,i}\}|\{\mathbf{x}_{c,i}\}, \mathbf{w})P(\mathbf{w}) \sum_{c\in\text{topk}} P(c|R)P(\{\tilde{y}_{c,i}\}|\{\tilde{\mathbf{x}}_{c,i}\}, \mathbf{w}),$$

taking the log and applying Equation (13) gives us the following objective:

$$\begin{aligned}
&\sum_{i=1}^{S} \text{BCE}(y_i, \mathbf{w}^\top \phi(\mathbf{x}_i)) \\
&+ \sum_{c \in \text{topk}} P(c|R) \ln \left[\sum_{i=1}^{Q} \exp\left(\text{BCE}(\tilde{y}_{c,i}, \mathbf{w}^\top \phi(\tilde{\mathbf{x}}_{c,i})) \right) \right] \\
&+ \lambda \|\mathbf{w}\|^2 + \text{const.}
\end{aligned}$$

In practice, we introduce multipliers to each term as hyperparameters during optimization, drop the constant, and use means instead of sums, creating the following objective:

$$\frac{\lambda_3}{S} \sum_{i=1}^{S} \text{BCE}(y_i, \mathbf{w}^\top \phi(\mathbf{x}_i)) \tag{14}$$

$$+ \frac{\lambda_2}{K} \sum_{c \in \text{topk}} P(c|R) \ln \left[\frac{1}{Q} \sum_{i=1}^{Q} \exp\left(\text{BCE}(\tilde{y}_{c,i}, \mathbf{w}^\top \phi(\tilde{\mathbf{x}}_{c,i})) \right) \right] \tag{15}$$

$$+ \lambda_3 \|\mathbf{w}\|^2, \tag{16}$$

where K is the number of objects retrieved for the prior. This objective is used as the log of the target distribution, $\ln P(\mathbf{w})$, in Equation (4), where we also adopt the original median kernel suggested in [41]. We also opt to use SVGD in a stochastic manner, where both the observed samples and query samples are mini-batched.

From a non-probabilistic standpoint, one can interpret Equation (14) as the likelihood of the observed data, Equation (15) as the object shape prior, and Equation (16) as a regularization term.

5.3 Experiments

In this section, we aim to experimentally validate the following claims: (1) BRRP is more *robust* than deep learning methods; (2) BRRP is more *accurate* than methods that use uninformative priors; (3) BRRP can capture principled uncertainty. We begin by providing

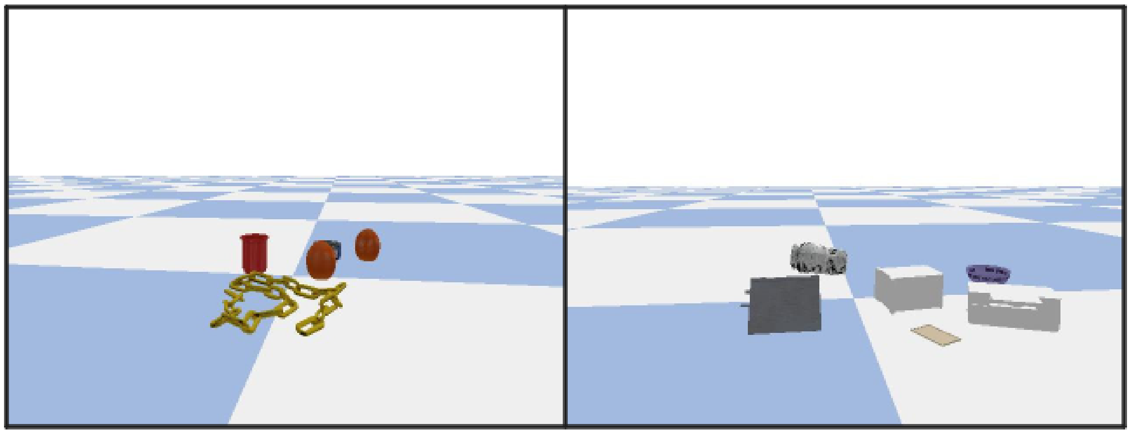

Figure 10: Sample images of procedurally generated scenes used to evaluate BRRP. **Left:** a YCB scene. **Right:** a ShapeNet scene.

details on the experiments such as the baselines and metrics, then we introduce results and analyses.

BRRP Implementation: We use a set of 50 objects from the YCB dataset [48] to act as the prior for our experiments with BRRP. We implement the method in PyTorch and run the method on an NVIDIA RTX GeForce 2070 GPU.

Baselines: We compare our work against two main baselines, V-PRISM from sec. 4.1 and a version of PointSDF [25] that predicts occupancy and is trained on ShapeNet [47] scenes. V-PRISM is a probabilistic mapping method that uses an uninformative prior. This means that it is robust to novel objects, but doesn't accurately reconstruct object backsides. We refer to this baseline as **V-PRISM**. In contrast, PointSDF is a learning-based method. This means it can leverage prior information from mesh datasets to accurately reconstruct the backside, but can suffer in performance under significant distributional shift. We refer to this baseline as **PointSDF**. When computing reconstruction meshes with PointSDF, a level set of $\tau = 0.3$ is used.

Procedurally Generated Scenes: We use the generated scenes from sec. 4.4 to evaluate our method. These scenes are constructed with objects from ShapeNet [47], YCB [48], and Objaverse [49] datasets. There are 100 multi-object scenes for each mesh dataset. Each

Method	ShapeNet Scenes	YCB Scenes	Objaverse Scenes
V-PRISM	0.3092	0.5003	0.4640
PointSDF	**0.3600**	0.4601	0.3471
BRRP (ours)	0.3124	**0.5277**	**0.4809**

Table 4: Intersection over union (IoU) on procedurally generated scenes from three different mesh datasets. BRRP uses a YCB prior and PointSDF is trained on ShapeNet scenes.

scene contains up to 10 objects. Some meshes in the Objaverse and ShapeNet scenes did not have correctly rendered textures and were instead rendered as plain white objects. fig. 10 contains two example images of these procedurally generated scenes. We also conduct an experiment on robustness where we perturb instance segmentation of the ShapeNet scenes by 2 pixels and evaluate reconstructions.

We evaluate performance on the procedurally generated scenes with two metrics: intersection over union (IoU) and chamfer distance. See sec. 4.4 for further explanation of these metrics.

Real World Scenes: In order to showcase robustness to real-world noise, we evaluate on real world scenes collected with a Kinect depth camera. In order to obtain instance segmentations, we use Grounded SAM [55] along with some depth filters. We evaluate on these real world scenes qualitatively with images of scene reconstruction and visualizing surface uncertainty.

Results: In Table 4, we display the IoU results from procedurally generated scenes. The chamfer distances for the procedurally generated scenes is shown in fig. 11. The qualitative reconstructions on real world scenes can be seen in fig. 12.

Insight 1: *BRRP is more accurate than a method with an uninformative prior.*

As showcased in Table 4, BRRP outperforms V-PRISM on each set of procedurally generated scenes. It has the highest IoU improvement from V-PRISM on the YCB scenes. A

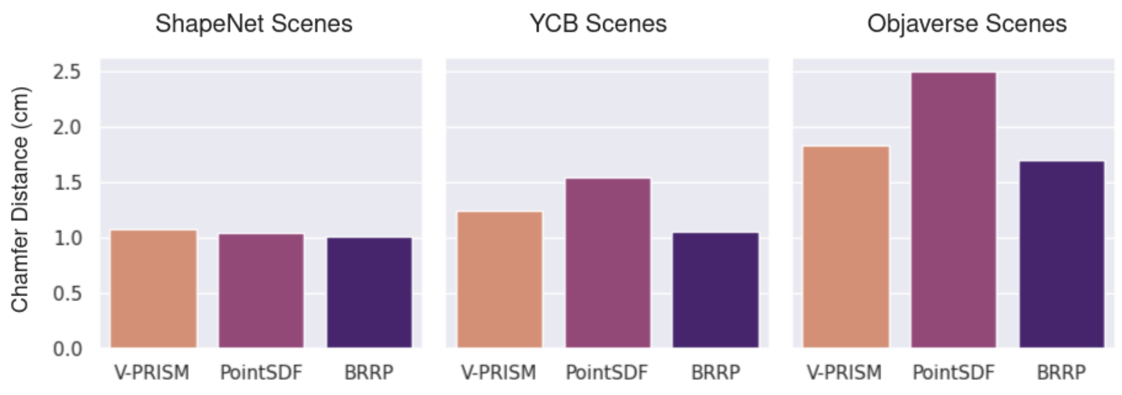

Figure 11: Chamfer distances (lower is better) for various methods across the procedurally generated scenes. Values are reported in centimeters. BRRP has the lowest chamfer distance on each dataset.

similar pattern can be seen in the chamfer distances in fig. 11, where BRRP consistently outperforms V-PRISM, with the biggest gap of 0.19 (cm) occuring on the YCB scenes. This makes sense because BRRP uses a subset of the YCB objects as its prior.

We can see this improvement qualitatively in fig. 12. While BRRP and V-PRISM are comparable for most objects, there exist certain objects that V-PRISM predicts to occupy a large portion of space that the object *doesn't* occupy. BRRP is able to more accurately reconstruct these objects. The clearest example of this is the dark green object in the right-most scene in fig. 12.

Both of these quantitative and qualitative results suggest BRRP is generally more accurate than V-PRISM. It is the most accurate when evaluated on objects in its prior distribution.

Insight 2: *BRRP is more robust than a deep learning method.*

While PointSDF outperformed BRRP on the ShapeNet scenes on IoU (Table 4), BRRP had a lower Chamfer distance (Figure 11). BRRP also performed better on mesh datasets that PointSDF was not trained on. In Table 4, we can see that on Objaverse scenes, where the objects were novel to both methods, BRRP performed better than PointSDF. When measuring chamfer distance, BRRP outperformed PointSDF on all datasets as shown in

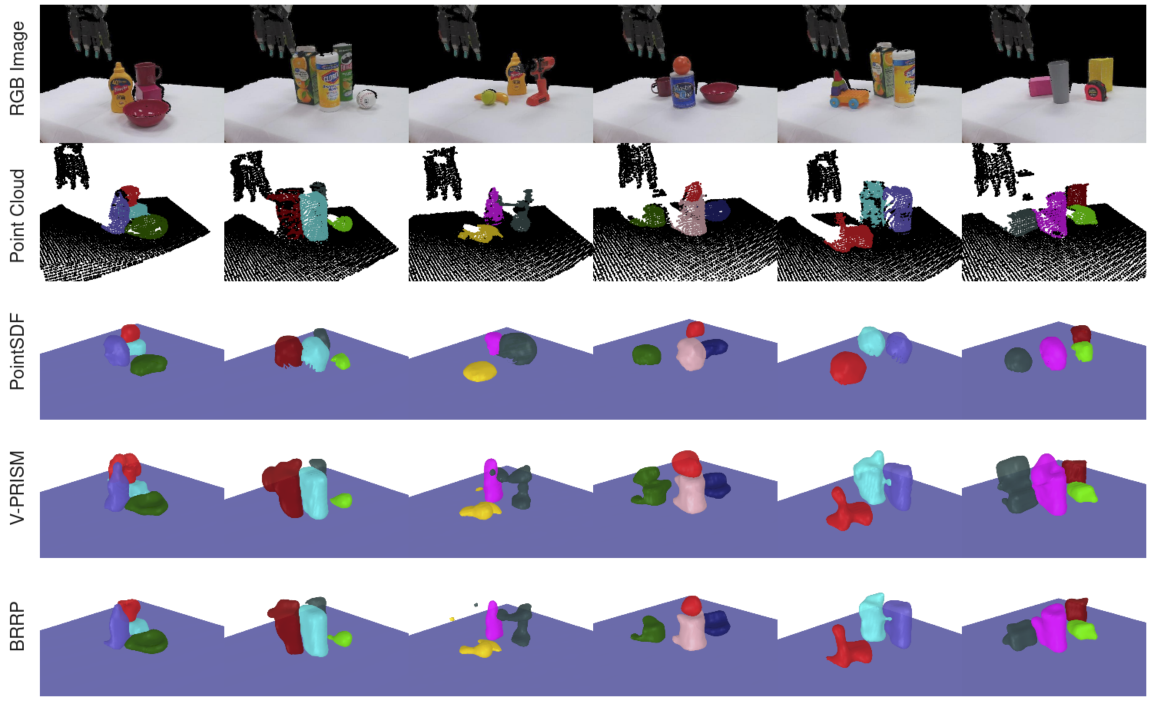

Figure 12: Qualitative comparison of BRRP and our baselines. PointSDF tends to predict a spherical shape for many non-spherical objects. V-PRISM can sometimes predict occupancy in portions of the scene that are not occupied. Our method is more robust and can more accurately reconstruct the scenes.

Figure 11. These results suggests BRRP is more robust to different object distributions than PointSDF.

Next, we evaluate robustness to slightly incorrect instance segmentations. We take the procedurally generated ShapeNet scenes and shift the segmentation over by 2 pixels. In Figure 13, we compare the IoU of BRRP and PointSDF on the scenes with and without the shift. Our method performs better on the shifted scenes compared to PointSDF.

On the real world scenes in Figure 12, BRRP is qualitatively more robust than PointSDF. PointSDF struggles with the noise associated with real world scenes as well as the novel objects. It tends to predict a spherical object on many non-spherical objects in the real world scenes. Our method on the other hand, is better able to reconstruct these scenes, including the objects that are out-of-distribution for its prior.

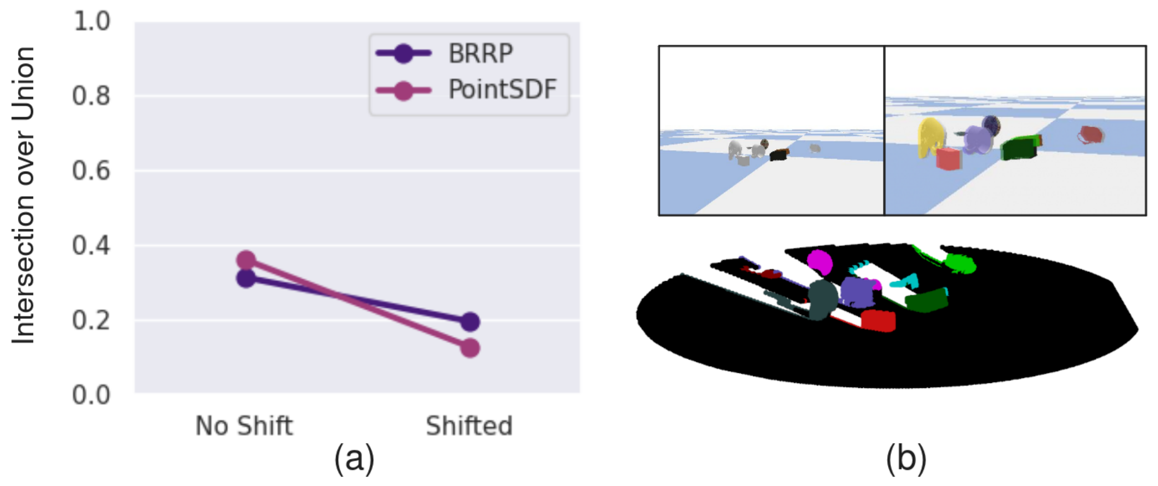

Figure 13: (a) IoU of BRRP and PointSDF on ShapeNet scenes with and without shifted segmentations. Our method is more robust to segmentation shifts. (b) An example of a scene and the corresponding point cloud with shifted segmentation.

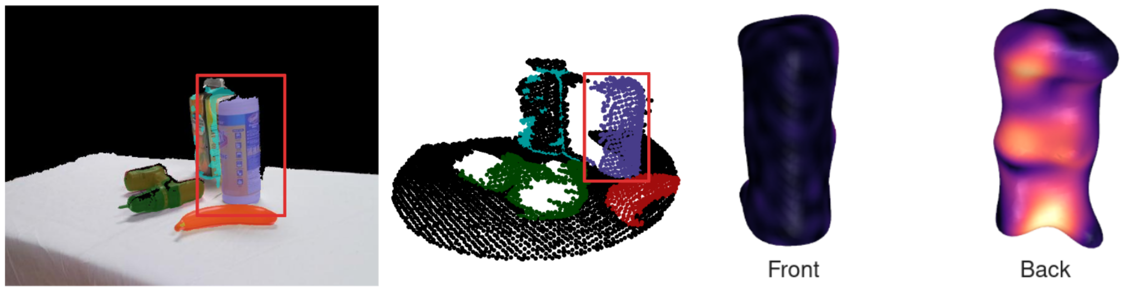

Figure 14: Visualization of a cylindrical container surface uncertainty from BRRP. Lighter areas correspond to higher uncertainty about the shape. Uncertainty is high on the occluded side of the container.

These results suggest BRRP is more robust than PointSDF. Even though by one metric PointSDF outperforms BRRP on ShapeNet scenes, when the scenes are perturbed BRRP performs better.

Insight 3: *BRRP can capture principled uncertainty about object shape.*

Figure 14 shows a qualitative example of uncertainty from BRRP. We measure the uncertainty by taking the variance of logits over weight particles. Our method predicts the highest uncertainty in areas of the surface that are occluded. This suggests that we can utilize surface uncertainty from BRRP in a similar way to how GPIS surface uncertainty is utilized in many grasping applications.

6 CONCLUSION

Creating a 3D representation of a multi-object tabletop scene is crucial for many manipulation tasks. In this thesis, I have explored two *Bayesian* methods for 3D scene reconstruction: V-PRISM and BRRP. Both methods used a hinge point representation. V-PRISM used an *uninformative* prior, whereas BRRP used an *informative* prior during reconstruction. Qualitative real world experiments as well as quantitative experiments on procedurally generated scenes were performed on each method. Each method was shown to be robust and accurate as well as capture principled uncertainty.

REFERENCES

[1] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "CHOMP: Gradient optimization techniques for efficient motion planning," in *2009 IEEE international conference on robotics and automation*, 2009, pp. 489–494.

[2] S. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Research Report 9811*, 1998.

[3] S. Chen, J. Bohg, and C. K. Liu, "SpringGrasp: An optimization pipeline for robust and compliant dexterous pre-grasp synthesis," *arXiv preprint arXiv:2404.13532*, 2024.

[4] C. de Farias, B. Tamadazte, M. Adjigble, R. Stolkin, and N. Marturi, "Task-informed grasping of partially observed objects," *IEEE Robotics and Automation Letters*, 2024.

[5] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *3rd international conference on learning representations, ICLR 2015, san diego, CA, USA, may 7-9, 2015, conference track proceedings*, 2015.

[6] A. Nguyen, J. Yosinski, and J. Clune, “Deep neural networks are easily fooled: High confidence predictions for unrecognizable images,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 427–436.

[7] F. Ramos and L. Ott, “Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent,” *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1717–1730, 2016.

[8] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, “Signed distance fields: A natural representation for both mapping and planning,” in *RSS 2016 workshop: Geometry and beyond-representations, physics, and scene understanding for robotics*, 2016.

[9] R. Senanayake and F. Ramos, “Bayesian hilbert maps for dynamic continuous occupancy mapping,” in *Conference on robot learning*, 2017, pp. 458–471.

[10] R. Senanayake, A. Tompkins, and F. Ramos, “Automorphing kernels for nonstationarity in mapping unstructured environments.” in *CoRL*, 2018, pp. 443–455.

[11] W. Zhi, L. Ott, R. Senanayake, and F. Ramos, “Continuous occupancy map fusion with fast bayesian hilbert maps,” in *2019 international conference on robotics and automation (ICRA)*, 2019, pp. 4111–4117.

[12] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “Deepsdf: Learning continuous signed distance functions for shape representation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.

[13] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy networks: Learning 3d reconstruction in function space,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4460–4470.

[14] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, “Implicit neural representations with periodic activation functions,” *Advances in neural information processing systems*, vol. 33, pp. 7462–7473, 2020.

[15] 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.

[16] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3D gaussian splatting for real-time radiance field rendering,” *ACM Transactions on Graphics*, vol. 42, no. 4, 2023.

[17] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, “DUSt3R: Geometric 3D vision made easy,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2024.

[18] W. Zhi, H. Tang, T. Zhang, and M. Johnson-Roberson, “Simultaneous geometry and pose estimation of held objects via 3D foundation models,” *IEEE Robotics and Automation Letters*, 2024.

[19] W. Liu, Y. Wu, S. Ruan, and G. S. Chirikjian, “Robust and accurate superquadric recovery: A probabilistic approach,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 2676–2685.

[20] J. Xu, W. Cheng, Y. Gao, X. Wang, S. Gao, and Y. Shan, “Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models,” *arXiv preprint arXiv:2404.07191*, 2024.

[21] H. Jun and A. Nichol, “Shap-e: Generating conditional 3d implicit functions,” *arXiv preprint arXiv:2305.02463*, 2023.

[22] M. Liu, C. Xu, H. Jin, L. Chen, M. Varma T, Z. Xu, and H. Su, “One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[23] F. Engelmann, K. Rematas, B. Leibe, and V. Ferrari, “From points to multi-object 3D reconstruction,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4588–4597.

[24] Z. Liao, J. Yang, J. Qian, A. P. Schoellig, and S. L. Waslander, “Uncertainty-aware 3D object-level mapping with deep shape priors,” in *2024 IEEE international conference on robotics and automation (ICRA)*, 2024, pp. 4082–4089.

[25] M. Van der Merwe, Q. Lu, B. Sundaralingam, M. Matak, and T. Hermans, “Learning continuous 3d reconstructions for geometrically aware grasping,” in *2020 IEEE international conference on robotics and automation (ICRA)*, 2020, pp. 11516–11522.

[26] F. Williams, Z. Gojcic, S. Khamis, D. Zorin, J. Bruna, S. Fidler, and O. Litany, “Neural fields as learnable kernels for 3d reconstruction,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 18500–18510.

[27] J. Huang, Z. Gojcic, M. Atzmon, O. Litany, S. Fidler, and F. Williams, “Neural kernel surface reconstruction,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 4369–4379.

[28] Y. Kasten, O. Rahamim, and G. Chechik, “Point cloud completion with pretrained text-to-image diffusion models,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[29] Y.-C. Cheng, H.-Y. Lee, S. Tulyakov, A. G. Schwing, and L.-Y. Gui, “Sdfusion: Multimodal 3d shape completion, reconstruction, and generation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 4456–4465.

[30] B. Saund and D. Berenson, “Diverse plausible shape completions from ambiguous depth images,” in *Conference on robot learning*, 2021, pp. 1802–1813.

[31] W. Agnew, C. Xie, A. Walsman, O. Murad, Y. Wang, P. Domingos, and S. Srinivasa, "Amodal 3d reconstruction for robotic manipulation via stability and connectivity," in *Conference on robot learning*, 2021, pp. 1498–1508.

[32] L. Li, S. Khan, and N. Barnes, "Silhouette-assisted 3d object instance reconstruction from a cluttered scene," in *Proceedings of the IEEE/CVF international conference on computer vision workshops*, 2019, pp. 0–0.

[33] N. Gothoskar, M. Cusumano-Towner, B. Zinberg, M. Ghavamizadeh, F. Pollok, A. Garrett, J. Tenenbaum, D. Gutfreund, and V. Mansinghka, "3DP3: 3D scene perception via probabilistic programming," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9600–9612, 2021.

[34] S. Dragiev, M. Toussaint, and M. Gienger, "Gaussian process implicit surfaces for shape estimation and grasping," in *2011 IEEE international conference on robotics and automation*, 2011, pp. 2845–2850.

[35] W. Martens, Y. Poffet, P. R. Soria, R. Fitch, and S. Sukkarieh, "Geometric priors for gaussian process implicit surfaces," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 373–380, 2016.

[36] M. Matak and T. Hermans, "Planning visual-tactile precision grasps via complementary use of vision and touch," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 768–775, 2022.

[37] J. Lundell, F. Verdoja, and V. Kyrki, "Robust grasp planning over uncertain shape completions," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2019, pp. 1526–1532.

[38] D. Hidalgo-Carvajal, H. Chen, G. C. Bettelani, J. Jung, M. Zavaglia, L. Busse, A. Naceri, S. Leutenegger, and S. Haddadin, "Anthropomorphic grasping with neural object shape completion," *IEEE Robotics and Automation Letters*, 2023.

[39] T. S. Jaakkola and M. I. Jordan, “A variational approach to bayesian logistic regression models and their extensions,” in *Sixth international workshop on artificial intelligence and statistics*, 1997, pp. 283–294.

[40] R. Senanayake and F. Ramos, “Building continuous occupancy maps with moving robots,” in *Proceedings of the AAAI conference on artificial intelligence*, 2018, vol. 32.

[41] Q. Liu and D. Wang, “Stein variational gradient descent: A general purpose bayesian inference algorithm,” *Advances in neural information processing systems*, vol. 29, 2016.

[42] M. A. Fischler and R. C. Bolles, “Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[43] G. Bouchard, “Efficient bounds for the softmax function and applications to approximate inference in hybrid models,” in *NIPS 2007 workshop for approximate bayesian inference in continuous/hybrid systems*, 2007, vol. 6.

[44] J. Daunizeau, “Semi-analytical approximations to statistical moments of sigmoid and softmax mappings of normal variables,” *arXiv preprint arXiv:1703.00091*, 2017.

[45] W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3D surface construction algorithm,” in *Proceedings of the 14th annual conference on computer graphics and interactive techniques*, 1987, pp. 163–169.

[46] H. Thomas, “Learning new representations for 3D point cloud semantic segmentation,” PhD thesis, Université Paris sciences et lettres, 2019.

[47] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, “ShapeNet: An Information-Rich 3D Model Repository,” Stanford University — Princeton University — Toyota Technological Institute at Chicago, arXiv:1512.03012 [cs.GR], 2015.

[48] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The ycb object and model set: Towards common benchmarks for manipulation research," in *2015 international conference on advanced robotics (ICAR)*, 2015, pp. 510–517.

[49] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi, "Objaverse: A universe of annotated 3d objects," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 13142–13153.

[50] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," *arXiv:2304.02643*, 2023.

[51] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, and others, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[52] Y. Kuang, J. Ye, H. Geng, J. Mao, C. Deng, L. Guibas, H. Wang, and Y. Wang, "Ram: Retrieval-based affordance transfer for generalizable zero-shot robotic manipulation," *arXiv preprint arXiv:2407.04689*, 2024.

[53] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, and others, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, 2021, pp. 8748–8763.

[54] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *2009 IEEE international conference on robotics and automation*, 2009, pp. 3212–3217.

[55] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, and others, “Grounded sam: Assembling open-world models for diverse visual tasks,” *arXiv preprint arXiv:2401.14159*, 2024.

Name of Candidate: Herbert Wright

Date of Submission: December 6, 2024