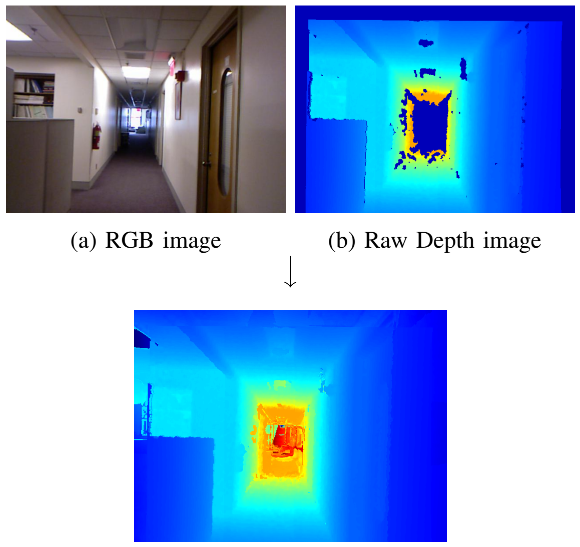
# Plane-Based Depth Image Completion

Nishchal Bhandari[1], Julian Straub[2], and John W. Fisher III[3]

***Abstract*— The Kinect has encountered widespread use in the areas of 3D mapping and computer vision thanks to its low cost, portability, and depth sensing capability. Unfortunately, its depth sensor has a maximum range of eight meters, and past four meters it tends to be very noisy. This work presents an approach to augment the Kinect with the ability to estimate the data that is missing from its sensors, providing a more complete picture of what is present in a given scene. In contrast to much of what has been done, our work takes an approach more suited to the three-dimensional structure of the world rather than relying on image processing techniques. We develop a model that utilizes image position, existing depth, color, and surface normal measurements and derive an algorithm that can estimate missing data in a depth image using only sensor data as input. Our results show that the generated estimates have a root mean square error of about 25 centimeters.**


(a) RGB image (b) Raw Depth image

(c) Complete depth image

Fig. 1: Due to limitations of RGB-D sensors, there is missing depth information (denoted in the darkest shade of blue in the raw depth image) for remote parts of the scene. We propose a Bayesian nonparametric approach to filling in the missing data (c).

## I. INTRODUCTION

The Microsoft Kinect, Asus Xtion PRO, and other RGB-D (red, green, blue, depth) sensors have encountered widespread use in the areas of 3D mapping and computer vision thanks to their low cost, portability, and depth sensing capabilities. While these sensors were originally designed to be used for motion sensing, gesture detection, and body recognition, researchers and engineers have taken advantage of them for use in collision avoidance, mapping and localization, and 3D model generation. Unfortunately, since these sensors were designed for very specific applications, their depth sensors tend to be quite limited in range and noisy in certain situations. As a result, they perform poorly in larger, open areas such as hallways where a large portion of the scene is too far away (see Fig. 1). This is not a problem if the sensor is being used for close-range applications, such as object tracking, but it can be harmful when it is used to make larger-scale inferences about the surrounding area. Our goal is to augment the Kinect sensor by using the available depth and RGB information to estimate the missing depth information, providing a more complete picture of what is present in a given scene.

A key assumption we make is that the environment can be accurately represented as a set of planes. This assumption is usually reasonable in man-made environments, where the structure of the world tends to actually be composed mostly of flat surfaces. For the purpose of estimating depth, we argue that this assumption is reasonable even in the presence of curved surfaces. These can be approximated by multiple planes. Higher order approximations would result in a more accurate model, but may not be worth the trade-offs in complexity and processing time for real world applications. Planes segmentations have the benefit that they are possible to obtain in real time [4] and can therefore be used in common applications such as mobile robotics.

Our contributions include using this assumption to formulate a probabilistic Bayesian model that also takes into consideration color information regarding the scene, and using that model to perform inference on missing depth information. We utilize small-variance asymptotics in our inference approach to arrive at a simple DP-means-like algorithm to estimate missing depth information in a scene. We demonstrate the advantages and disadvantages of our approach using quantitative analysis against ground truth data in a variety of scenes.

## II. RELATED WORK

Completion and inference of depth information utilizing RGB images has been experimented with in a variety of ways. Using stereo images, depth can be inferred in a fairly

[1] Nishchal Bhandari is an undergraduate with the Department of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology `nishchal@mit.edu`

[2]Julian Straub is a graduate student within the Computer Science and AI Laboratory at the Massachusetts Institute of Technology `jstraub@csail.mit.edu`

[2] John W. Fisher III is a Senior Research Scientist within the Computer Science and AI Laboratory at the Massachusetts Institute of Technology, where he runs the Sensing, Learning, and Inference Group `fisher@csail.mit.edu`

straightforward manner by looking at the displacement of corresponding points between the two images. Saxena et al. [10] inferred a depth image from a single monocular image using a discriminatively-trained Markov Random Field that takes into consideration both global and local image features such as texture variations, texture gradients, and texture energy. Their method worked well in finding relative depth differences between objects in a scene, but struggled at finding accurate absolute depth values. Eigen et al. [3] also take on the problem of predicting depth from a single image using global and local properties but instead utilize a Multi-Scale Deep Network that consists of coarse and fine-scale networks, taking advantage of large, raw datasets as training data.

Sheng and Cheung [12] attempt to solve the same problem as this work is: completing a depth image given an RGB image and a noisy, incomplete depth image such as from a Kinect sensor. They use an approach that splits an image into depth layers that are obtained using color-depth correlation, background modeling, and spatial smoothness and then removes noise and interpolates using a bilateral filter. Because their approach relies on splitting the image into a discrete number of foreground and background layers, it is not well suited to situations where objects span a larger, continuous range of depth values. Such examples of scenes where this situation could arise are the walls or floors of a long hallway or a large room. Lu et al. [6] tackle the problem by splitting it into two parts: dealing with the noise and holes in the depth map separately. They utilize edge orientation along with directional filters to improve on other filtering-based approaches to the problem.

In contrast to much of what has been done, this work takes an approach more suited to the three-dimensional structure of the world. Rather than relying on image processing techniques, we propose an approach that utilizes surface normal measurements derived from depth information along with reasonable assumptions about the approximate structure of the world to arrive at a method that can better handle the general cases of depth image completion.

## III. MODEL

In this work, the world is modeled using a Multivariate Dirichlet Process Gaussian von Mises-Fisher Mixture Model (DP-GvMF-MM). Data points that are generated by this model are represented as a vector containing position in image space $(u, v)$, depth $d$, color $(L, a, b)$ expressed in the L*a*b* colorspace (where $L$ captures the lightness of a color, and $a$ and $b$ capture hue), and surface normal $(n_x, n_y, n_z)$. We express the parts of this vector that are in Euclidean space as $e$ and the parts that lie in spherical space as $n$:

$$x = \begin{pmatrix} e \\ n \end{pmatrix} \tag{1}$$

$$e = \begin{pmatrix} u & v & d & L & a & b \end{pmatrix}^T \tag{2}$$

$$n = \begin{pmatrix} n_x & n_y & n_z \end{pmatrix}^T \tag{3}$$

Spatial coordinates in the image space $(u, v)$ along with depth $d$ are used rather than world coordinates $(x, y, z)$ in relation to the camera frame or some global coordinate system. While world coordinates could theoretically be used by utilizing existing depth measurements, the inference process would be made more difficult, as the $x$ and $y$ of each data point would be dependant on the depth measurement for that value.

Rather than working directly with the RGB color values the Kinect sensor provides, we choose to map the measured color into the L*a*b* colorspace (also known as the CIELAB colorspace). L*a*b* has been successfully used by other works [1] [2] for the purposes of clustering and segmentation, which makes it a better choice than RGB for the problem at hand. Euclidean distances between colors in this space roughly correspond to differences in color perceived by humans [16], which will prove useful during inference steps. We initially chose not to use the lightness channel of L*a*b* to avoid the effects of shadows and highlights making a single surface look like multiple ones, but in practice observed that including the lightness channel provided more accurate depth estimates. Typically, in indoor scenes, the various surfaces present tend to vary in lightness as well as in hue, which the $L*$ channel helps capture.

From a generative standpoint, mixture weights $\pi_\infty$ are first sampled according to a Dirichlet Process [15] (DP) with hyperparamter $\alpha$:

$$\pi_\infty \sim \mathcal{DP}(\alpha) \tag{4}$$

The Dirichlet Process is used to allow for a possible countably infinite number of mixtures [9]. An additional benefit to using DP is the presence of many existing inference algorithms, including ones that can be used on spherical data [14] like the surface normals $n$ in this model.

Mean $\mu_k$ and covariance $\Sigma_k$ for the portion of each mixture component that exists in Euclidean space are sampled from a Normal Inverse Wishart (NIW) distribution parameterized by $\theta$:

$$\mu_k, \Sigma_k \sim \mathcal{NIW}(\theta) \tag{5}$$

The NIW is used because it is a conjugate prior to the multivariate Gaussian [7]. This makes inference easier because it guarantees that the posterior distribution will be in closed form.

Similarly, mean $m_k$ and concentration parameter $\tau$ for the portion of each mixture component that exists in spherical space are sampled from a prior distribution $\psi$.

Labels $z_i$ are distributed according to a categorical distribution:

$$z_i \sim \text{Categorical}(\pi) \tag{6}$$

Each of the $N$ data points $x_i$ in the image is then sampled from the mixture component it is assigned to via the label $z_i$. The Euclidean part of $x_i$ is distributed normally according to the mean and covariance of the component that label $z_i$ corresponds to:

$$e_i \sim \mathcal{N}(\mu_{z_i}, \Sigma_{z_i}) \tag{7}$$

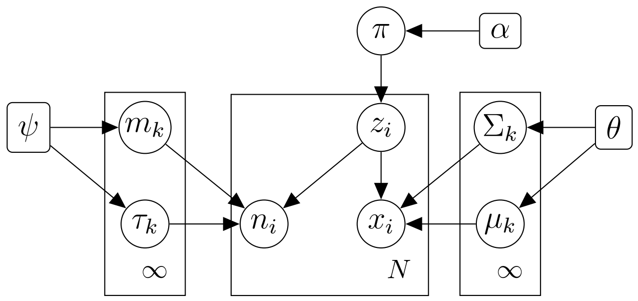


Fig. 2: The graphical model of the proposed Dirichlet process multivariate Gaussian von Mises-Fisher mixture model

The normal of each data point $x_i$ is sampled from a von Mises-Fisher (vMF) distribution with mean and concentration also determined by label $z_i$:

$$n_i \sim \text{vMF}(m_{z_i}, \tau_{z_i}) \tag{8}$$

Given this model, we can write the joint distribution over unknown parameters as:

$$\begin{aligned} &p(\pi, z_1^N, x_1^N, \Sigma_1^\infty, \mu_1^\infty, m_1^\infty, \tau_1^\infty; \alpha, \theta, \psi) \\ &\quad = p(\pi; \alpha) \prod_{k=1}^{\infty} p(\Sigma_k; \theta) p(\mu_k; \theta) p(m_k; \psi) p(\tau_k; \psi) \\ &\quad \prod_{i=1}^{N} p(z_i \mid \pi; \alpha) p(x_i \mid \Sigma_{z_i}, \mu_{z_i}, m_{z_i}, \tau_{z_i}, z_i; \alpha, \theta, \psi) \end{aligned} \tag{9}$$

## IV. INFERENCE

In the context of the given model, our problem is that for some number of $x_i$, we only have their location in image space $(u, v)$ and their color $(L, a, b)$. In order to generate the missing parts of each $x_i$, the other parameters in the model that determine $x_i$ must first be inferred using the input data: all values of $x_i$ that are known completely.

### *A. Labeling Existing Data*

Because part of the data exists in Euclidean space, and part exists in spherical space, we designed a hard clustering algorithm that mixes two existing ones: DP-means [5] and DP-vMF-means [13]. Both of these algorithms make the small variance assumption in order to simplify the inference procedure. We choose to take advantage of these hard clustering algorithms for their simplicity and scalability. Our hope is that in the future, this method could be used to perform depth completion in real-time.

These hard clustering algorithms are derived by starting at some algorithm that estimates the desired parameters without any assumptions, such as the EM algorithm or Gibbs sampling, and then analysing the asymptotic behavior when a certain parameter approaches an extreme value. In order to perform small variance analysis on our model, we will assume that

$$\Sigma_k = I\sigma^2 \forall k \tag{10}$$

$$\tau_k = \frac{\beta}{\sigma^2} \forall k \tag{11}$$

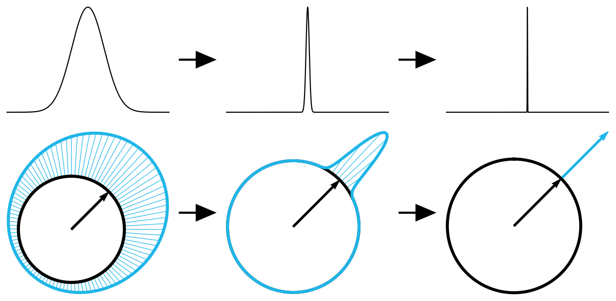


Fig. 3: As $\sigma$ is decreased the distributions become more concentrated about their respective mode. The top row shows how the variance of a Guassian decreases with $\sigma$. The bottom row shows how the concentration of a vMF increases as $\sigma$ decreases.

where $I$ is the identity matrix of appropriate dimension, and $\beta$ is a known parameter we set. We make this assumption so that as $\sigma$ goes to 0, the variances $\Sigma_k$ of the Gaussians become small, and the concentration parameters $\tau_k$ of the vMFs goes to $\infty$, resulting in the peaky behavior shown in Fig. 3. We rely on the distributions being concentrated around their respective modes for the following analysis.

If we integrate out mixing weights $\pi$, the result is the Chinese Restaurant Process [13]:

$$\begin{aligned} &p(z_i = k \mid z_{-i}, \mu_1^\infty, e_1^N, m_1^\infty, n_1^N; \sigma, \rho^2, \tau_0, m_0) \\ &\quad \propto \begin{cases} |\mathcal{I}_k| p(x_i \mid \mu_k, m_k; \sigma, \rho^2, \mu_0, \tau_0, m_0) \\ \qquad \alpha p(x_i; \sigma, \rho^2, \tau_0, m_0) \end{cases} \end{aligned} \tag{12}$$

where $\mathcal{I}_k$ is the set of $i$ such that label $z_i = k$ and

$$\begin{aligned} &p(x_i \mid \mu_k, m_k; \sigma, \rho^2, \tau_0, m_0) \\ &\quad = p(e_i \mid \mu_k; \sigma, \rho^2, \tau_0, m_0) p(n_i \mid m_k; \sigma, \rho^2, \mu_0, \tau_0, m_0) \\ &\quad = \mathcal{N}(e_i; \mu_k, I\sigma^2) \text{vMF}(n_i; m_k, \frac{\beta}{\sigma^2}) \end{aligned} \tag{13}$$

and

$$\begin{aligned} &\alpha p(x_i; \sigma, \rho^2, \tau_0, m_0) \\ &\quad = \alpha p(e_i; \sigma, \rho^2, \tau_0, m_0) p(n_i; \sigma, \rho^2, \tau_0, m_0) \\ &\quad = \alpha \mathcal{N}(e_i; I(\sigma^2 + \rho^2)) \frac{Z(\frac{\beta}{\sigma^2}) Z(\tau_0)}{Z(\|\frac{\beta}{\sigma^2} + \tau_0 m_0\|_2)} \end{aligned} \tag{14}$$

and $Z$ is the normalizing function for vMFs. In the limit as $\sigma \to 0$, the analysis performed in [13] lets us simplify the part of the above equation that comes from surface normals $n_i$. If we let $\alpha = e^{\lambda_1 \frac{\beta}{\sigma^2} - \frac{\lambda_2}{2\sigma^2}}$, where $\lambda_1$ captures the vMF contribution to $\alpha$ and $\lambda_2$ captures the Gaussian contribution, then

$$\begin{aligned} &\lim_{\sigma \to 0} \alpha \frac{Z(\frac{\beta}{\sigma^2}) Z(\tau_0)}{Z(\|\frac{\beta}{\sigma^2} + \tau_0 m_0\|_2)} \\ &\quad = \lim_{\sigma \to 0} Z(\tfrac{\beta}{\sigma^2}) \exp(\tfrac{\beta}{\sigma^2}(\lambda_1 + 1) - \tfrac{\lambda_2}{2\sigma^2}) \end{aligned} \tag{15}$$

Therefore, as $\sigma \to 0$, the labeling step becomes

$$\lim_{\sigma\to 0} p(z_i = k \mid z_{-i}, \mu_1^\infty, e_1^N, m_1^\infty, n_1^N; \sigma, \rho^2, \tau_0, m_0)$$
$$= \lim_{\sigma\to 0} \begin{cases} p_{\text{ex}} & k \le K \\ p_{\text{new}} & k = K+1 \end{cases} \tag{16}$$

where

$p_{\text{ex}} =$
$$\frac{|\mathcal{I}_k| \exp(\frac{\beta}{\sigma^2} m_k^T n_i - \frac{\|e_i-\mu_k\|^2}{2\sigma^2} - \frac{\beta}{\sigma^2}(\lambda_1+1) + \frac{\lambda_2}{2\sigma^2} + \frac{\|e_i-\mu_0\|^2}{2(\sigma^2+\rho)})}{\sum_{j=1}^K |\mathcal{I}_j| \exp(\frac{\beta}{\sigma^2} m_j^T n_i - \frac{\|e_i-\mu_j\|^2}{2\sigma^2} - \frac{\beta}{\sigma^2}(\lambda_1+1) + \frac{\lambda_2}{2\sigma^2} + \frac{\|e_i-\mu_0\|^2}{2(\sigma^2+\rho)})+1} \tag{17}$$

and

$p_{\text{new}} =$
$$\frac{1}{\sum_{j=1}^K |\mathcal{I}_j| \exp(\frac{\beta}{\sigma^2} m_j^T n_i - \frac{\|e_i-\mu_j\|^2}{2\sigma^2} - \frac{\beta}{2\sigma^2}(\lambda_2+1) + \frac{\lambda_2}{2\sigma^2} + \frac{\|e_i-\mu_0\|^2}{2(\sigma^2+\rho)})+1} \tag{18}$$

We have used that the normalizers for the Gaussians and vMFs cancelled out. After factoring out $\sigma^{-2}$ from each term in the numerator of the exponent, we notice that the final term in the exponent goes to zero as $\sigma$ does:

$$\lim_{\sigma\to 0} \frac{\|e_i - \mu_0\|^2}{2(1+\frac{\rho}{\sigma^2})} = 0 \tag{19}$$

Hence the following assignment rule is equivalent to sampling from $p(z_i = k \mid z_{-i}, \mu_1^\infty, e_1^N, m_1^\infty, n_1^N; \sigma, \rho^2, \tau_0, m_0)$:

$$z_i = \operatorname*{arg\,max}_{k\in\{1,\dots,K+1\}} \begin{cases} \beta m_k^T n_i - \frac{\|e_i-\mu_k\|^2}{2} & k \le K \\ \beta(\lambda_1+1) - \lambda_2 & k = K+1 \end{cases} \tag{20}$$

Compared to the label assignment rule for DP-means:

$$z_i = \operatorname*{arg\,min}_{k\in\{1,\dots,K+1\}} \begin{cases} \|x_i - \mu_k\|^2 & k \le K \\ \lambda_2 & k = K+1 \end{cases} \tag{21}$$

and for DP-vmF-means:

$$z_i = \operatorname*{arg\,max}_{k\in\{1,\dots,K+1\}} \begin{cases} x_i^T m_k & k \le K \\ \lambda_1 + 1 & k = K+1 \end{cases} \tag{22}$$

our algorithm's assignment rule takes into account the distance metrics for both spherical and Euclidean space and attempts a joint optimization over both.

If a new cluster is created (the second case in Eq. (20)), the number of clusters $K$ is updated:

$$K \leftarrow K + 1 \tag{23}$$

and new means are initialized using the value of the data point being labeled:

$$\mu_K = e_i \tag{24}$$

$$m_K = n_i \tag{25}$$

Unlike our labels, we have separate means for $e_i$ and $n_i$. Hence, we can update them separately using the updates derived in [5] and [13]:

$$\mu_k = \frac{1}{|\mathcal{I}_k|} \sum_{i\in\mathcal{I}_k} e_i \tag{26}$$

$$m_k = \frac{\sum_{i\in\mathcal{I}_k} n_i}{\|\sum_{i\in\mathcal{I}_k} n_i\|_2} \tag{27}$$

## *B. Assigning Labels to Partial Data Points*

Now that labels can be assigned to existing data points, we must decide how to assign labels to incomplete data points. Because we are trying to maximize the probability that an incomplete data point $x_i$ has label $z_i = k$, we can use the same rule that assigned existing labels to complete data points, but slightly modified to account for the information we do not have:

$$z_i = \operatorname*{arg\,min}_{k\in\{1,\dots,K\}} ||e_i^{\setminus d} - \mu_k^{\setminus d}|| \tag{28}$$

where $e_i^{\setminus d}$ and $\mu_k^{\setminus d}$ denote the respective vector without the depth component.

## *C. Estimating Depth*

With the labels assigned to the incomplete data points, we now must estimate the depth values of these points. For convenience, we describe the covariance of each cluster $k$ as being partitioned as

$$\Sigma_k = \begin{bmatrix} \Sigma_{d,d} & \Sigma_{k,k} \\ \Sigma_{k,k}^T & \Sigma_{k,k}^{\setminus d} \end{bmatrix} \tag{29}$$

Given a label $z_i$, a data point $e_i$ is dependent only on a single mean and covariance, so

$$p(d_i \mid z_i, \mu_{z_i}, \Sigma_{z_i}, e_i^{\setminus d}) = \mathcal{N}(d_i; \bar{\mu}, \bar{\Sigma}) \tag{30}$$

where, as the result of conditioning on a multivariate Gaussian,

$$\bar{\mu} = d_{z_i} + \Sigma_{z_i,z_i} \left(\Sigma_{z_i,z_i}^{\setminus d}\right)^{-1} (e_i^{\setminus d} - \mu_{z_i}^{\setminus d}) \tag{31}$$

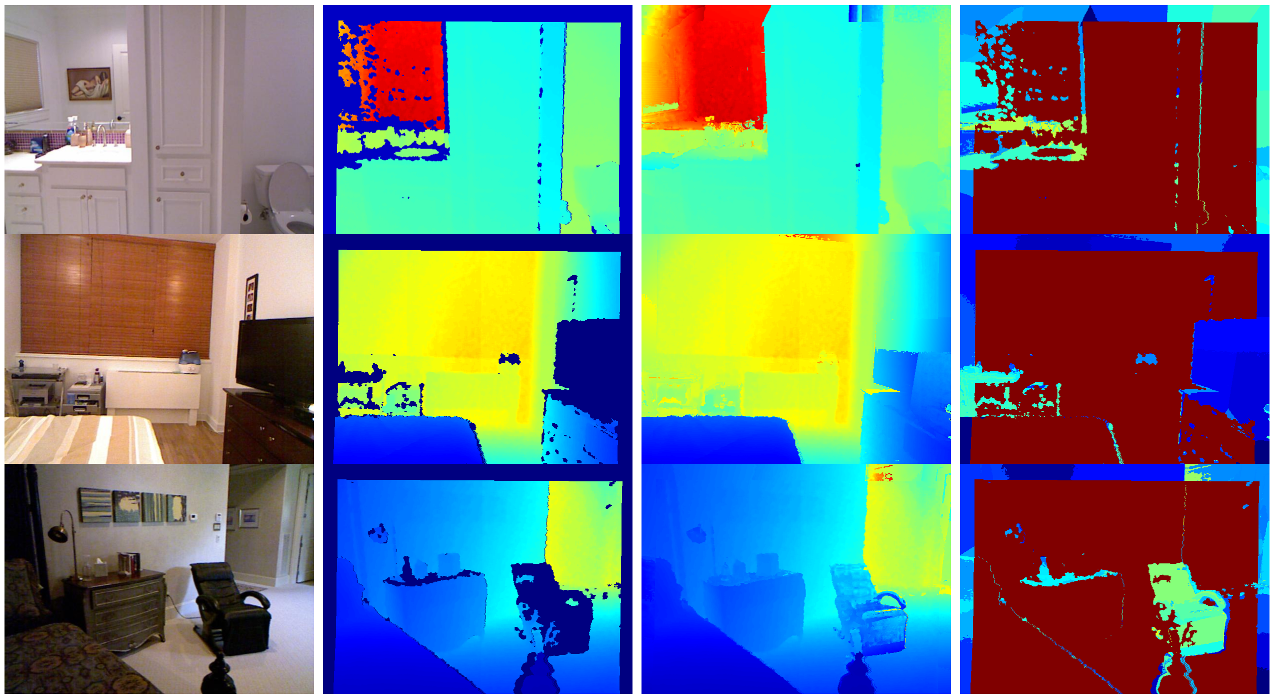
and

$$\bar{\Sigma} = \Sigma_{d,d} - \Sigma_{z_i,z_i} (\Sigma_{z_i,z_i}^{\setminus d})^{-1} \Sigma_{z_i,z_i}^T \tag{32}$$

The maximum likelihood estimate of a Gaussian is the mean, so in order to maximize the likelihood of our depth estimates, we assign

$$d_i = \bar{\mu} \quad \text{for} \quad i \in \mathcal{M} \tag{33}$$

where $\mathcal{M}$ denotes the set of indices from 1 to $N$ where $e_i$ is missing a value for $d_i$. Algorithm 1 shows what the final procedure looks like for depth completion under the inference method we describe.



(a) RGB Images of scenes (b) Raw depth images (c) Estimated depth image (d) Log-likelihood of estimates

Fig. 5: Columns (a) and (b) show the input color and depth images to our algorithm. These images are typical of what a RGB-D sensor measures. Column (c) shows the output of the proposed depth estimation method. Column (d) is an image representing the log-likelihood of the estimates, where colors that are more red are more likely, and the darkest shade of red represents information that already existed.

**Algorithm 1** Complete Algorithm for Depth Completion

```
1: Initialize $K = 0, \mu = \emptyset, m = \emptyset, z = \emptyset$
2: while Not converged do
3:     for every complete data point $x_i$ do
4:         Assign label $z_i$ according to Eq. (20)
5:         if $z_i = K + 1$ then
6:             $\mu_{z_i} = e_i, m_{z_i} = n_i$
7:             $K = K + 1$
8:         end if
9:     end for
10:    for $k \in \{1, ..., K\}$ do
11:        $\mu_k = \frac{1}{|\mathcal{I}_k|} \sum_{i \in \mathcal{I}_k} e_i$
12:        $m_k = \frac{\sum_{i \in \mathcal{I}_k} n_i}{\|\sum_{i \in \mathcal{I}_k} n_i\|_2}$
13:    end for
14: end while
15: for every incomplete data point $x_i$ do
16:     Assign label $z_i$ according to Eq. (28)
17:     $d_i = \bar{\mu}$, where $\bar{\mu}$ is calculated according to Eq. (31)
18: end for
```

## V. Results

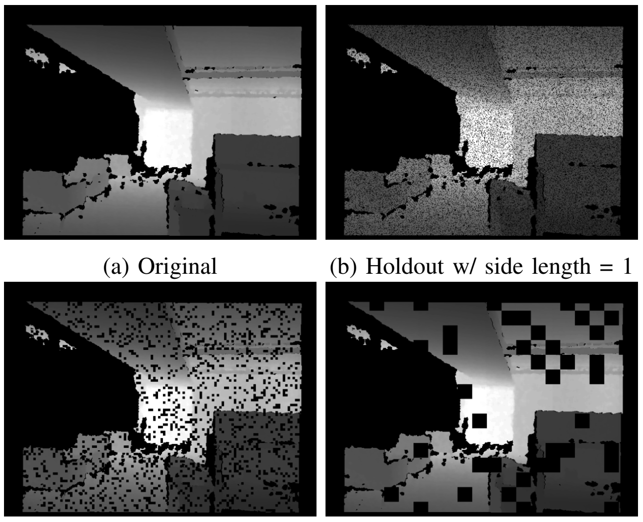
We give quantitative measurements of our method on several images from a large dataset to show how it performs in general, as well as qualitative analysis on certain images to highlight particular pros and cons. Fig. 5 depicts qualitative results of how our method performs on actual data from a Kinect without any added noise. We evaluate the root mean squared error (RMSE) of our estimates against ground truth data where random chunks of the depth image have been removed to simulate missing data as depicted in Fig. 4. We vary the size of the missing chunks to see how our method performs against different severities of data loss. Even when the size of the chunks is increased, however, the net amount of removed data remains at about 20% to isolate the effects of the size of the chunks from the net amount of data missing. In order to control how heavily we weigh surface normal values when performing the clustering step, we use a parameter $\beta$, which is the same as the one described in the Inference section. The parameter appears in the label update step:

$$z_i = \underset{k \in \{1,\ldots,K+1\}}{\arg\max} \begin{cases} \beta m_k^T n_i - \frac{\|e_i - \mu_k\|^2}{2} & k \leq K \\ \beta(\lambda_1 + 1) - \lambda_2 & k = K + 1 \end{cases} \tag{34}$$

In the first case, it controls how heavily to weigh the surface normal contribution for this maximization. In the second case, it controls the threshold for creating new clusters. In our tests we varied the value of $\beta$ to see it effects on the error of our estimate.

### A. *Overall Accuracy and Effect of Varying Parameters*

We tested our method on a subset of the NYU Depth Dataset V2 [8] dataset. The size of the subset we tested on



(a) Original (b) Holdout w/ side length = 1

(c) Holdout w/ side length = 5 (d) Holdout w/ side length = 30

Fig. 4: Depth images with missing data simulated

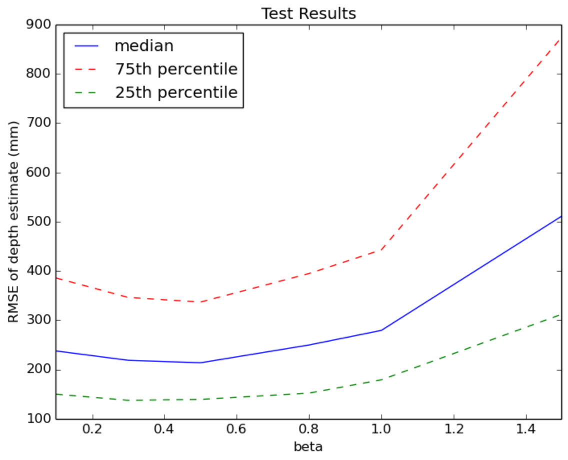


Fig. 6: RMSE in the depth estimate as parameter $\beta$ is varied, and chunks of size 3-by-3 are removed from the depth image. Notice that a minimum exists at roughly $\beta = 0.5$

was 778 images. Figures Fig. 6 and Fig. 7 depict the 25th, 50th, and 75th percentiles of the RMSE of the tests. The median RMSE on these images with a hole size of 3 and a value of $\beta$ equal to 0.5 was about 230 millimeters.

As the size of the chunks of missing data was increased, the RMSE of the estimate also increased. This result isn't surprising, as each missing data point has less neighbors with information around it to help it get labeled appropriately. What was surprising, however, was the RMSE only increased by 65 millimeters when the size of the missing chunks was increased from 3 to 40. This suggests that although the quality of the estimate decreases when tasked with completing larger chunks of missing data, the deterioration in quality is not too large.

As $\beta$ was varied from 0.1 to 1.5, the RMSE typically decreased at first, then increased. The values of $\beta$ that

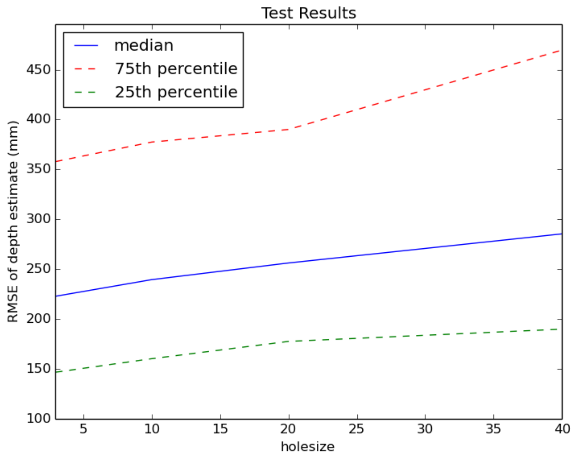


Fig. 7: Mean error in depth estimate as size of holdout data is varied, and $\beta$ is held to a constant value of 1.0. The RMSE tends to increase as larger chunks of depth data are removed from the image, suggesting that the task of estimating depth grows more difficult as the size of the missing chunks increases, even though the total amount of missing data remains the same.The fact that the variance (indicated by the spread between percentiles) increases as well supports this as well.

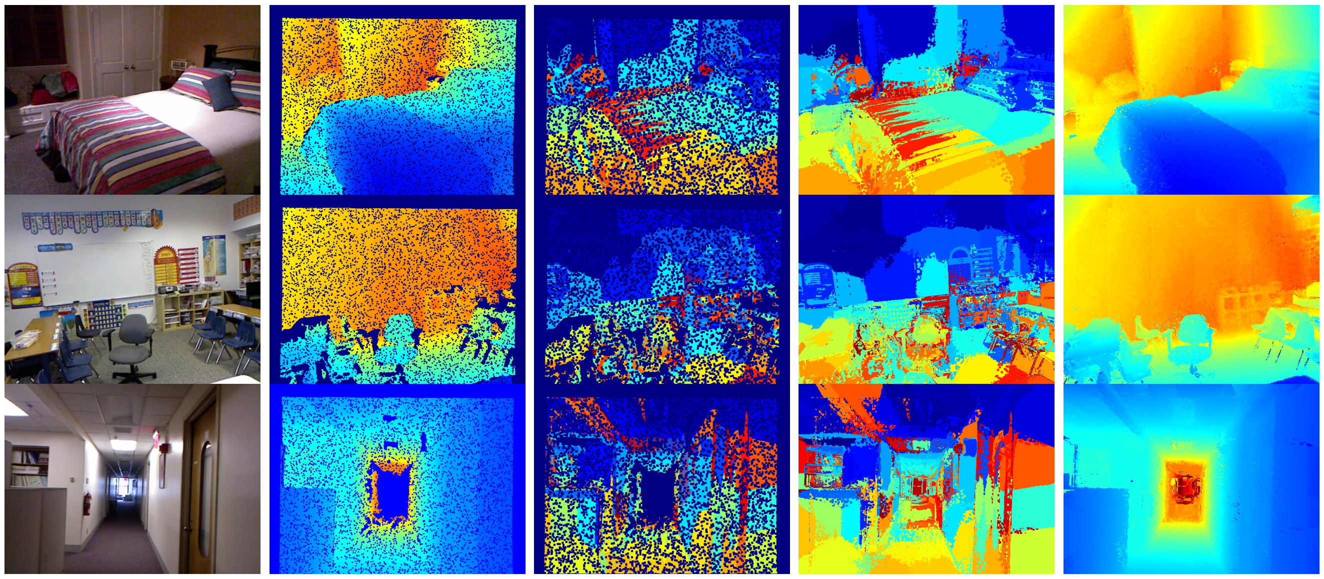
tended to have the smallest RMSE were between 0.4 and 0.6. When $\beta = 0.5$, Eq. (20) gives equal weight to the parts of the data points that exist in Euclidean and Spherical space. The results from this experiment suggest that the proposed method performs best when the joint optimization gives equal weight to the Gaussian components and the vMF components.

## B. Analysis of Specific Cases

We examine specific images at different points during the inference pipeline to better understand both the advantages of our method as well as the particular areas in scenes where errors in the depth estimate are more likely. For all of the specific cases examined, the parameter $\beta$ was set to 0.5 (the optimal value determined from the prior testing), and random 3-by-3 chunks of data were removed from the depth image.

In the first row of images in Fig. 8, a simple case is shown where only very small chunks of depth data were missing initially. In the assignment step, an interesting labeling appears near the foot of the bed corresponding to a striped design in the RGB image. This is a result of including color in our data vector. While one might think that having separate labels for points that lie on the same surface could be harmful, the final depth estimate for this image shows that this is not the case.

In the second row of Fig. 8, a case is presented where medium sized chunks of missing data are present in the raw depth image. Notice in the raw depth image that the armrests of the rolling chair are missing completely, as well as the legs on some of the chairs in the right hand side of the image. In this case, some image filtering based approaches



(a) RGB Images of scenes (b) Raw depth images (c) Labelling Step (d) Assignment step (e) Estimated depth image

Fig. 8: Columns (a) and (b) show the input color and depth images to our algorithm. 3-by-3 chunks of pixels are removed from the depth images in column (b) so that we can use the raw depth image as ground truth to evaluate against. Column (c) shows the output of the labelling step, where every color represents a different label. Column (d) shows how the pixels with missing depth data (the darkest blue pixels in column (b)) are assigned to labels. Column (d) shows the final output, a completed depth image.

might have mistaken these missing areas as a part of the wall or floor, since the areas around are primarily a part of the wall or floor. In the estimated depth image that our method produced, such a mistake is not made. The armrests and the chair legs are given reasonable estimates. However, in the top right corner of this image, a poster on the wall is incorrectly labeled and given a very obviously incorrect depth estimate. While the method presented does occasionally make poor estimates, this can be prevented. In the images presented as well as in the tests ran, the algorithm was allowed to assign depth values even when it was uncertain of them for testing purposes. For practical applications, the algorithm could easily be modified to only supply an estimate for a point when Eq. (30) is above a certain threshold.

In the final row of Fig. 8, a case is presented where a very large chunk of missing data is present in the raw depth image. In this case, the large chunk can be attributed specifically to the range limitations of the depth sensor. In this case, an image filtering approach definitely would not suffice. Notice that in the estimated depth image, our method actually extrapolates that depth values even larger than those in the original depth image are required (the red and dark red represent these values). For robots using depth sensors to navigate buildings such as hospitals or retirement homes, scenes like this hallway are fairly common. Having even a rough estimate at such a long range, such as the one generated, could be used to improve localization capabilities.

## VI. CONCLUSIONS

We have designed a model and derived an inference procedure that allows us to estimate missing depth information given a color image and partial depth image. Our method achieves mean errors of about 250 millimeters in a large variety of test scenes, and produces a reasonable complete depth image. Our method relies only on a few parameters being set ahead of time and standard output data from an RGB-D sensor, meaning it has the potential to run in real-time without extra input.

Our method takes a drastically different approach than existing solutions and has the potential to be more robust to the different types of depth data loss that occur when using a RGB-D sensor, especially cases where large chunks of data are lost due to range limitations.

Future work will aim to experiment with small changes in the model and inference procedure to try and reduce error rates in the produced depth estimate, as well as well as optimizing the algorithm to run in real-time to test how the completed depth image affects common applications that rely on it. There are many small modifications that could be tested, and have the potential to cumulatively improve the estimated depth image by a large amount. For example, we currently use the standard distance metric for Euclidean space for color, but another metric that could be used is the CIEDE2000 Color Difference Formula [11], which supposedly better accounts for perceptual color differences. The current method also currently weighs all Euclidean data equally, but possible improvements could arise by granting different weights to image location, color, and depth values.

## REFERENCES

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Susstrunk. Slic superpixels compared to

state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012.

[2] Seema Bansal and Deepak Aggarwal. Color image segmentation using cielab color space using ant colony optimization. *International Journal of Computer Applications*, 29(9):28–34, 2011.

[3] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems*, pages 2366–2374, 2014.

[4] Dirk Holz, Stefan Holzer, Radu Bogdan Rusu, and Sven Behnke. Real-time plane segmentation using rgb-d cameras. In *RoboCup 2011: Robot Soccer World Cup XV*, pages 306–317. Springer, 2011.

[5] Brian Kulis and Michael I Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. *ICML*, 2011.

[6] Anh Vu Le, Seung-Won Jung, and Chee Sun Won. Directional joint bilateral filter for depth images. *Sensors*, 14(7):11362–11378, 2014.

[7] Kevin P Murphy. Conjugate bayesian analysis of the gaussian distribution. *def*, $1(2\sigma 2)$:16, 2007.

[8] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[9] Radford M Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

[10] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. Learning depth from single monocular images. In *Advances in Neural Information Processing Systems*, pages 1161–1168, 2005.

[11] Gaurav Sharma, Wencheng Wu, and Edul N Dalal. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30, 2005.

[12] Ju Shen and Sen-Ching S Cheung. Layer depth denoising and completion for structured-light rgb-d cameras. In *Computer Vision and Pattern Recognition*, pages 1187–1194, 2013.

[13] Julian Straub, Trevor Campbell, Jonathan P. How, and John W. Fisher III. Small-variance nonparametric clustering on the hypersphere. In *CVPR*, 2015.

[14] Julian Straub, Jason Chang, Oren Freifeld, and John W Fisher III. A dirichlet process mixture model for spherical data. In *AISTATS*, 2015.

[15] Yee Whye Teh. Dirichlet process. In *Encyclopedia of Machine Learning*, pages 280–287. Springer, 2011.

[16] Xuemei Zhang, D Amnon Silverstein, Joyce E Farrell, and Brian A Wandell. Color image quality metric s-cielab and its application on halftone texture visibility. In *Compcon*, pages 44–48. IEEE, 1997.