Object Modeling and Recognition from Sparse, Noisy Data via Voxel Depth Carving

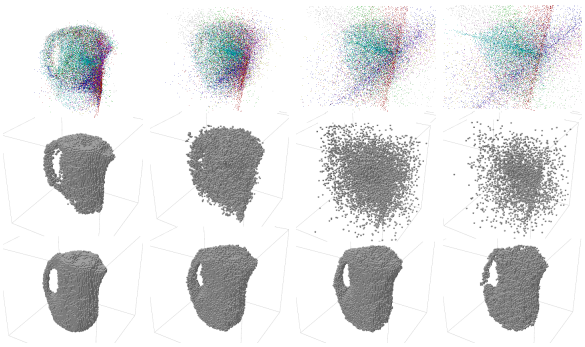
Matthew Klingensmith, Martin Herrmann, and Siddhartha S. Srinivasa

The Robotics Institute, Carnegie Mellon University

Abstract. In this work, we make the case for using volumetric information for shape reconstruction and recognition from noisy depth images for robotic manipulation. We provide an efficient algorithm, Voxel Depth Carving (a variant of Occupancy Grid Mapping) which accomplishes this goal. Real-world experiments with lasers, RGB-D cameras, and simulated sensors in both 2D and 3D verify the effectiveness of our algorithm in comparison to traditional point-cloud based methods.

1 Introduction

3D sensors are cheaper and more readily available than ever before. Commercial depth sensors, such as Microsoft's *Kinect* or the Asus *Xtion Pro* provide inexpensive, low-latency, colored depth data. However, this comes at the expense of significant noise and missing data [22]. The rise of cheap 3D sensing presents the challenge of using noisy, incomplete depth data for robotic manipulation. Here, we focus on two key perception challenges: *shape reconstruction* and *object recognition*, which are necessary for robotic manipulation of everyday objects.



(a) 0cm (b) 5cm (c) 10cm (d) 15cm

Fig. 1: The effect of noisy data: the top row shows the raw point clouds. The middle row shows the result of the reconstruction using hit data only. The bottom row shows the result of our method. We vary σ_z, the standard deviation of noise on the depth sampled from $\mathcal{N}(0, \sigma_z)$ from 0cm to 15cm.

Many previous works on object recognition [14,17,25] and reconstruction [12, 19,21,26] use a point cloud, a set of 3D points which encode data from the sensor.

Unfortunately, point clouds throw away important information encoded in depth images: locality of adjacent depth pixels, the implicit *ray* from the focal point of the depth camera to each point in the cloud, and, importantly, the implicit *volumetric* information implied by the ray passing through empty space from the camera to the scene. Consequently, corruption from noise and missing data can severely distort a point cloud, reducing its usefulness.

In contrast, researchers in the mobile robot navigation community have long used occupancy grids [5] and ray clouds rather than point clouds to overcome the problem of noisy, sparse data. Occupancy grids provide a natural way to reason about the uncertainty of noisy sensors by incorporating ray noise models into the occupancy update probabilities.

However, directly applying occupancy grid mapping to the domain of recognizing and reconstructing common objects presents many challenges. Traditional occupancy grid mapping techniques [5] assume laser-like or sonar-like sensors carving out grid cells in spaces much larger than the robot, wheras we are concerned with very dense depth scans covering a much smaller space at higher resolution. In our domain, ray rasterization is slowed by the extreme number of rays, and results in artifacts around the fine details of objects. To solve this problem, we iterate over a fixed set of voxels, rather than over rays. We use projection and interpolation rather than rasterization to carve out space much more quickly and conservatively. We call this technique *Voxel Depth Carving*. Recognizing objects in a noisy occupancy grid also presents difficulties. We are unable to use typical surface descriptors or point descriptors used in 2D images or point clouds, and instead we must use volumetric descriptors to recognize objects. In this work, we show that simple affine-invariant geometric moments can sufficiently recognize objects from a database of hundreds of candidates when only noisy volumetric information is given. Further, in our domain, the majority of the space around objects may be dominated by occlusion, and remains "unknown." To deal with unknown space from occlusions, we construct a Markov Random Field with strong structural priors to make assumptions about the space behind objects. Doing so gives us much better object recognition performance when the number of views of the object is small.

By applying these techniques from occupancy grid mapping to object recognition and reconstruction, our work aims to exploit and recover the implicit volumetric information encoded in noisy, incomplete depth images; while at the same time being faster than typical occupancy grid mapping techniques in our domain. As a result, we are able to reconstruct and recognize objects even in extremely noisy conditions – where the corresponding point cloud is so distorted as to be unrecognizable (Fig.1).

2 Related Works

Despite extensive research, object recognition and reconstruction remains challenging in both the 2D [24] and 3D [11] cases. We are interested in a more specific subset of the problem where multiple, registered viewpoints are considered.

Shape Reconstruction When only 2D data is given, silhouette information may be used in the form of a *visual hull* [18]. However, generating silhouettes is not always feasible. Algorithms that incorporate color information from Lambertian scenes by evaluating *photo consistency* [16] produce high quality, high resolution geometry reconstruction, but fail for lower resolutions.

Another body of work concerns constructing geometry from high resolution laser scans [13, 21, 26]. Such scans typically contain very little noise, and are extremely dense. Most algorithms use only hit information and discard passthrough

information. In the absence of noise, these approaches are well suited, but with real-world data from commercial sensors, their prerequisite of having dense, watertight point clouds falls short. Our approach begins by assuming large chunks of missing data, occlusion, and noise are all present.

In contrast to the methods used in 3D object recognition and reconstruction, works in the robotic navigation and mapping community have long made extensive use of volumetric information in the form of Occupancy Grid Maps [5] for 2D navigation, and in the case of 3D navigation, as Octree mapping [13] and related algorithms. Unlike point cloud based methods, which only consider the endpoints of rays, these methods integrate passthrough information to construct a probabilistic representation of the space around the robot. Our work can be seen as an extension of Occupancy Grid Mapping for object modeling which uses a voxel-centric approach rather than a ray-centric approach to more efficiently compute the occupancy probability of the space around the object.

This paper builds on one of our earlier unpublished reports in which we first proposed the method of Voxel Depth Carving [9]. Since then, another work by Guggeri1 *et. al.* [6] independently introduced the same method to the computer graphics community, where it is known by the same name. They [6] define the concept of the *depth hull*, a complement to the visual hull, which plays a key role in our work. While their method is very similar to ours, they have not verified the method using real-world sensor data, as we have – and only consider the problem of high-resolution surface reconstruction, whereas we are also interested in object matching and probabilistic object models.

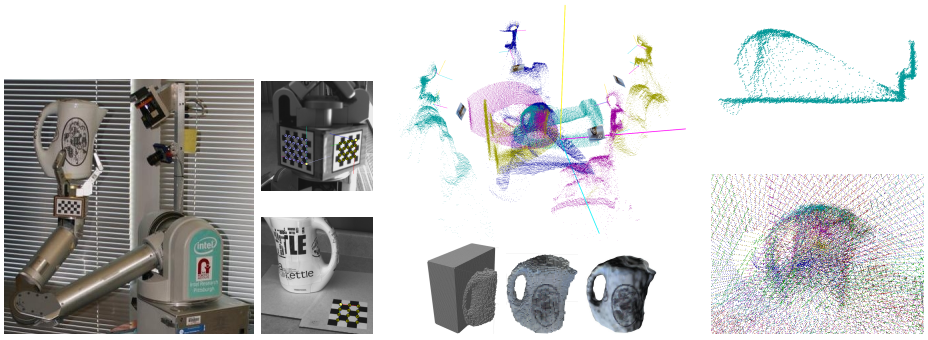
Object Recognition Template matching approaches such as LINEMOD [10] have been extended to noisy 3D data successfully, but such techniques cannot efficiently select models from thousands of candidates in a database, which is our goal. Hsiao *et. al.* [14] explore the problem of recognizing houshold objects for grasping from extremely noisy and incomplete depth data by considering alignment to a number of 3D objects from a very small database – but their approach (which relies only on simple ICP [1]) only scales to tens of objects. The Point Cloud Library [25] includes several methods of 3D object detection and recognition based on point features, but (as the name of the library suggests) all such methods rely purely on point cloud hits, and do not consider volumetric data.

Our work complements other object recognition techniques by using volumetric descriptors (such as moments) to match against a database of thousands of objects. We build on the work of Goldfeder [8] *et. al*, who recognize everyday objects by the Zernike [4] moments of their point clouds; we simply compute the moments of their depth hulls instead.

3 Technical Approach

Problem Assume that the robot takes N scans of a scene. We have $H_1, \ldots, H_N$ globally registered rigid poses of the robot's sensor. For each scan, assume we have M rays emanating from the sensor $R_k = \{r_1, \ldots, r_M\}$; where a ray $r_i = \{o_i, p_i\}$ has an origin o_i, and an endpoint "hit" p_i. We may further assume that for each R_k, all $o_i \in R_k$ are the same (that is, all the rays pass through a focal point). This is often the case for RGB-D sensors and laser scanners. We will use o_k to mean the focal point of scan R_k.

In the absence of noise, each $p_i \in R_k$ is a point on the surface of an object. However, we will assume that every ray in the scan has a length which is corrupted by noise, *i.e*



(a) (b) (c)

Fig. 2: Kettle reconstruction: (a) Experimental setup. (b) (Top) Multiple registered views, (Bottom) Voxel carving and mesh generation. (c) (Top) Grazing incidence from a single view, (Bottom) Aggregated views.

$$\tilde{d}_i = \|p_i - o_i\| = d_i + n_i$$

where d_i is the true length of the ray, and a random variable n_i denotes noise, drawn from a probability distribution $f_{n_i}(x)$. In our experiments, we use a simple depth-dependant Gaussian model of the noise *i.e.*: $f_{n_i}(x, d_i) = \mathcal{N}\big(d_i, \sigma_z(d_i)\big)$, where σ_z is a funciton which varies with depth.

The goal is to find a function $S(x) : \mathbf{R}^3 \to \{-1, 1\}$, the *shape function*, which is -1 whenever the space is *free* (*i. e* it does not contain an object), and 1 whenever the space is *filled*.

Specifcially, we may consider the probabilistic shape function $P_s(x|R_1, \ldots R_N)$ We will assume that a discrete representation (in the form of voxels) of the probabilistic shape function is sufficient. We construct a voxel grid representation of the space $V \in \mathbf{R}^{X \times Y \times Z}$, where X, Y, and Z are the number of cells along each axis of the workspace. And, $V[x] = P_s(x|R_1 \ldots R_N)$. We can similarly write the probability that a cell is free, $\bar{P}_s(x|\ldots)$. The joint distribution over all the cells, $P_s(x_1, \ldots, x_{XYZ}|R_1 \ldots R_N)$ is labeled $P_s(V|R_1, \ldots, R_N)$ for convenience.

Passthrough Information To find the shape function, it is necessary not only to consider the end-points of the rays in each scan (called the *point cloud*), but also the presence of rays passing through space between the scan origin and the end-point. To see why this is important, consider the physical process behind a laser scan. Rays of light (which have non-zero thickness)) emenate from a central point through the scene. Some of the rays strike objects in the scene directly (we will call these Type I rays), others will not hit any objects (Type II rays), and still others will *graze* objects (Type III rays). Grazing ray hits [23] are the most interesting of these, as in practice, the sensor will randomly return depths intermediate between the surface of the object and the background (Fig.3).

If we only consider the endpoints of rays, Type II and III rays immediately become useless, since for Type II rays, there is no endpoint, and for Type III rays, the endpoint is wildly corrupted by noise. In contrast, by using the entirety of the ray, we are able to use *all three* kinds of rays to determine which parts of the space are *free*, even if we can't say anything about which parts of the space are *occupied*.

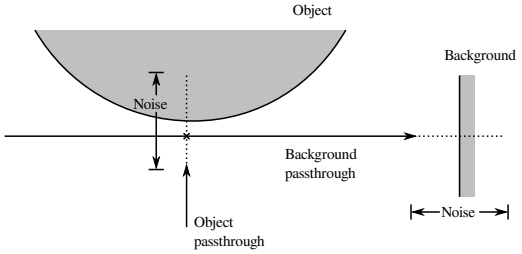
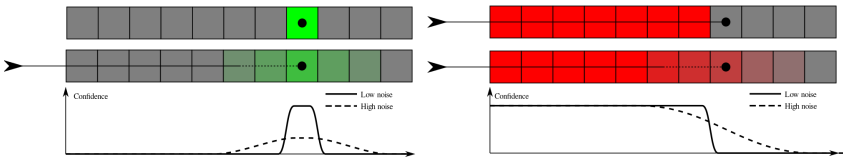


Fig. 3: With noise, the hit information becomes meaningless for evaluating the occupancy of the marked point. The passthrough information from the ray passing by the object remains useful.



(a) (b)

Fig. 4: The relative confidence of hit and passthrough information under low and high noise. While the usefulness of hit information declines rapidly with noise (a), most of the passthrough information (b) remains intact.

Consider the simple case where $f_{n_i}(x)$ is Gaussian, centered at the true depth with some standard deviation σ_n (Fig.4). As we increase σ_n, the occupancy probability of space becomes more and more blurred around the hit, while the probability of space around the sensor being free remains high. This fact remains true even for non-Gaussian noise models – notably, it is true for Type III (grazing) rays.

Depth Images In addition to storing the rays from each scan, we can also construct the *depth image* of the scan. We define a depth image of sensor j as a function $D_j[u, v]$, which takes in a 2d point and returns the length of a stored ray at that location. If no ray is stored there, the value is interpolated linearly from nearest neighbors. For RGB-D images, the depth image is merely a 2-dimensional grid of depth values, where each grid cell is the length of the ray passing through that grid cell on the image plane of the camera to the scene. For values of u and v which do not fall in the center of a grid cell, we use bilinear interpolation to determine the depth value. On the other hand, for laser scanners, the depth image must be synthesized.

We will also assume there exists a *projection mapping* of the depth image $\text{proj}_j(x)$, which, given a point x in the scene, projects that point back onto the depth image. In the case of RGB-D sensors, the projection mapping is simply a perspective projection of the scene onto the image plane. For laser scanners, we can use the pinhole camera model:

$$\text{proj}_j(x) = \left[\frac{(x - o_j) \cdot e_2}{(x - o_j) \cdot e_1}, \frac{(x - o_j) \cdot e_3}{(x - o_j) \cdot e_1}\right]^T$$

where (e_1, e_2, e_3) form an arbitrary orthonormal basis, with e_1 being the viewing direction. This model is only suitable for scans with a field of view significantly less than 180°.

Occupancy Grid Mapping Consider the typical occupancy grid mapping problem. If we first assume that each of the voxels is conditionally independent of one another given measurements, then what we wish to find is the probability of a cell being occupied given all of the measurements:

$$P_s(V|R_1, \ldots, R_N) = \prod_{c_i \in V} P_s(c_i|R_1, \ldots, R_N)$$

which, with the further assumption that each sensor measurement is independent becomes

$$\prod_i \prod_j \prod_{r_k \in R_j} P_s(c_i|r_k)$$

in other words, the probability of any instatiation of the voxel grid's MRF becomes a matter of simply computing the likelihood of each voxel being occupied given each ray in every scan independently. To imrove numerical stability, we can use the log-odds of the occupancy probability rather than the probability distribution itself:

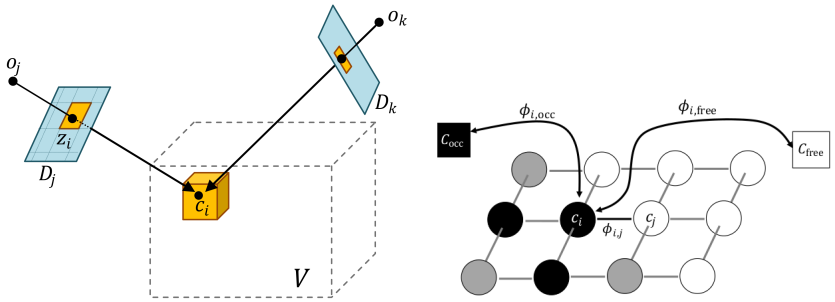
$$l_i = \sum_j \sum_{r_k \in R_j} \log \frac{P_s(c_i|r_k)}{1 - P_s(c_i|r_k)}$$

Traditional occupancy grid map approaches [5] solve the problem of finding the probabilistic shape function by *rasterizing* each ray using Brensham's algorithm [3], and lowering the occupancy probability for each cell that the ray passes through, while raising the occupancy probability for the cell that the ray ends in. Unfortunately, this approach has the disadvantage that as rays diverge from the sensor, they cover less space, and thus the reconstruction degrades as distance from the sensor increases. This problem is especially visible when the resolution of the voxel grid is high. Rasterizing each ray also takes a considerable amount of time when the resolution of the grid is high.

Voxel Depth Carving To speed up occupancy grid mapping in our domain, we make an approximation which uses projection instead of rasterization to carve away voxels which are likely to be free. Instead of iterating over rays and rasterizing them, we instead iterate over voxel cells, and determine whether a voxel cell should be marked as *free* from the collected sensor measurements. This method makes more sense in our domain, because typically the number of rays (from a collection of registered depth camera views) will be much higher than the number of voxels (which need only capture the shape of a small object). However, we can only approximate the true depth in areas that no rays actually pass through by interpolating between depth values of nearby rays. The approximation will be worse where there are rapid changes in depth smaller than the resolution of the sensor.

For each voxel cell in the scene, we project its center c_i onto the depth images of each sensor (Fig.5(a)). We then compare the linearly-interpolated value from the depth image $z_{i,j}$,with the Euclidean distance from the voxel cell to the sensor $(e_{i,j})$, which will tell us whether or not the cell should be free.

$$z_{i,j} = D_j\big[\text{proj}_j(c_i)\big], \quad e_{i,j} = \|c_i - o_j\|$$



(a) (b)

Fig. 5: (a) Two scans, R_i and R_k are shown, with one voxel c_i. Projections onto the depth images D_j and D_k, with the minimal projection shown as z_i. (b) A diagram of the Markov Random Field in 2D. $\phi_{i,j}$ is the edge with weight $\phi(c_i, c_j)$. The two special nodes, c_{free} and c_{occ} are shown as boxes. In 3D, cells are 6-connected. The darkness of a cell indicates its label.

From here, we can compute the probability that a cell is free given a particular ray. When the cell is sufficiently close to the sensor, we have:

$$\bar{P}_s(c_i|r_j) = P(z_{i,j} > e_{i,j}) = \int_{e_{i,j}}^{z_{\text{MAX}}} P(z_{i,j} = z)dz$$

but as the distance to the cell greatly exceeds $z_{i,j}$, the probability a cell is free becomes unknown. For an ideal sensor (*i.e* no noise), the probability distribution is a step function. The more noise is added to the sensor, the less steep the step function becomes.

Because of this, $P(z_{i,j} > e_{i,j})$ is very large near sensors, and very small near the vicinity of hits. If all we are interested in is whether or not a cell is free (and make no claims about the occupancy of voxels otherwise), we can make use of this fact by only considering the *minimum* distance to any sensor to decide whether a cell is free. That is,

$$z_i = \min_j z_{i,j}, e_i = \min_j e_{i,j}$$

we can then simply threshold the differences in the depth of the cell and measurements so as to only consider a cell free when it is sufficiently far away from a hit. This approach is a very conservative approximation of occupancy grid mapping: hit data can not be used at all, and only cells which are very likely to be free will be updated in each step. However, under conditions of very high noise, this conservative approach prevents us from carving important features of the object away.

$$\text{label}(c_i) = \begin{cases} -1 \text{ (free)} & \text{if } e_i < t_i \\ 0 \text{ (unknown)} & \text{else} \end{cases} \quad (1)$$

with some threshold t_i. All points which are closer to the origin than the threshold are classified as "free". We choose the threshold by subtracting a *margin* Δd from the measured depth, $t_i = z_i - \Delta d$. We generally want to use a low value for Δd because otherwise, cavities will look flatter than they actually are. Cavities with a depth less than Δd cannot be reconstructed at all.Using the noise model

of the sensor, we can determine a suitable value for the margin by considering the probability that there are points which are misclassified as "free", that is:

$$\exists p \in [o_i, p_i] : (\|p - o_i\| > d_i) \text{ and } (\|p - o_i\| < t_i) \iff d_i < z_i - \Delta d$$

The probability for a misclassification is given by

$$\begin{aligned} P_{\text{mis}} &= P(d_i < z_i - \Delta d) \\ &= 1 - \int_{-\infty}^{\Delta d} f_n(x)dx \\ &= 1 - F_n(\Delta d) \end{aligned}$$

where F_n is the probability distribution of the noise. Given a maximum acceptable misclassification probability $P_{\text{mis,max}}$, we can determine the smallest value of Δd that results in $P_{\text{mis}} \leq P_{\text{mis, max}}$.

In particular, for Guassian noise with mean μ and standard deviation σ, we have an optimal Δd of

$$\Delta d = \Phi^{-1}(1 - P_{\text{mis, max}})\sigma + \mu$$

by iterating over each voxel, we are able to "carve" large volumes of space which are likely to be free, leaving only the occupied space of the object as the number of views of the scene increase (Fig.2(b)). Note that since we take the center of the voxel only, as cell resolution decreases, the accuracy of our method degrades. This effect can be mitigated by additionally projecting the vertices of the voxel to the depth image and taking the minimum depth over the convex hull of the projected vertices. We are left with an algorithm which has performance characteristics linear in the number of voxels (rather than the number of rays, as in the occupancy grid mapping case). With several dense RGB-D scans, this is a significant performance improvement; at the cost of throwing away data near ray hits.

Unknown Space and Shape Priors Until now, we have only been concerned with determining whether or not a cell is "free", and make no claims about whether a cell is "occupied", instead opting to call all cells which are not free "unknown."

However, it is possible to determine which cells are likely to be occupied based on the free and unknown cells using a prior on the structure of objects. We can capture this structure using a Markov Random Field that has pairwise energies between cells (in addition to energies associated with sensor data). In this sense, Voxel Depth Carving becomes a method of updating the MRF given sensor data, with strong structural priors determining which cells are actually labeled as "occupied."

Markov Random Field Using a Markov Random Field (MRF), we can encode prior assumptions about the structure of objects into the pairwise energies between adjacent cells. That is:

$$P_s(V) = \frac{1}{Z} \prod_{c_i, c_j \in V} \phi(c_i, c_j)$$

where ϕ is the joint probability of two adjacent cells being occupied, and Z is a normalizing constant. One choice of ϕ which results in local smoothing and preserves boundaries is an Ising-like local model

$$\phi(c_i, c_j) = \exp\big(-\gamma L_i L_j\big)$$

where γ is a smoothness parameter, and L_i, L_j are the labels of voxel i and j (for the MRF we are required to label voxels containing points from the point cloud as "occupied" ($L_i = 1$)). This model penalizes labelings which are not locally contiguous. If we use the Ising model to propagate occupancy probability from ray hits to the rest of the voxel grid, what we will be left with is a distribution which is smooth in "unknown" areas, but maintains the hard edges obtained from Voxel Carving. Additionally, we can apply a prior to the structure of the Markov Random Field by adding additional vertices representing the labels "free" and "occupied", called c_{free} and c_{occ} (Fig.5(b)). Every voxel in the grid is then connected to these vertices with an energy computed from a prior and evidence from voxel carving. That is:

$$\phi(c_i, c_{\text{free}}) = \alpha I(L_i = -1) + \pi_{\text{free}}(c_i)$$
$$\phi(c_i, c_{\text{occ}}) = \alpha I(L_i = 1) + \pi_{\text{occ}}(c_i)$$

where π_{free}, π_{occ} are priors which weight "unknown" cells based on assumptions about the shape of the object.

Minimum Graph Cut We are left with a binary labeling problem on a Markov Random Field. The goal is to segment the field into two components: "occupied" and "free", in a way that minimizes the energy of the field. With a field of this type, it is true from the MaxFlow Mincut theorem [28] that the minimum-cut of the graph produces the optimal labeling. That is, we want to find a set of edges, $C = \{\phi_1, \ldots, \phi_M\}$, where ϕ_i is an edge between a cell that is "occupied" and "free"; and we want this set of edges to have minimal energy. The edges must divide the graph into two connected components, one containing c_{free}, the other containing c_{occ} The minimum graph cut can be found efficiently using Ford-Fulkerson algorithm [7]. Our problem is thus reduced to the binary image segmentation problem, which is efficiently solved by finding the minimum graph cut [2].

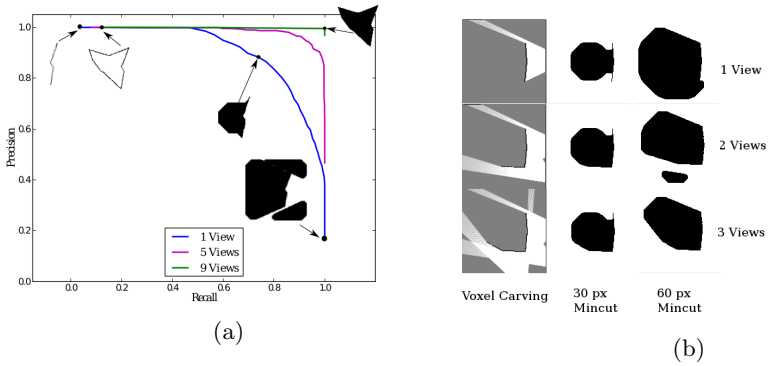
Shape Priors One prior we consider is to assume that the object is essentially a sphere of radius r_s and center c_r, called the "sphere prior" (Fig.6(b)). In this case,

$$\pi_{\text{free}}(c_i) = \frac{\|c_i - c_r\|}{r_s}$$

that is, a cell is less likely to be free the closer it is to the center of the sphere. Notice however that if α is very high, the minimum cut of the MRF will only use the prior in areas that are "unknown", using evidence from the sensor in other areas. We can also use a prior for the field which considers *each ray* independently using the following formula (which we call the "thin object prior"):

$$\pi_{\text{free}}(c_i) = \exp(-\beta h_x)$$

where $h_x = \min_j(\|x - o_j\| - z_j)$. This prior causes us to be less confident about voxels being occupied the further they are away from observed sensor points (Fig.8(b)). The parameter β controls just how thin we believe objects to be.



(a) (b)

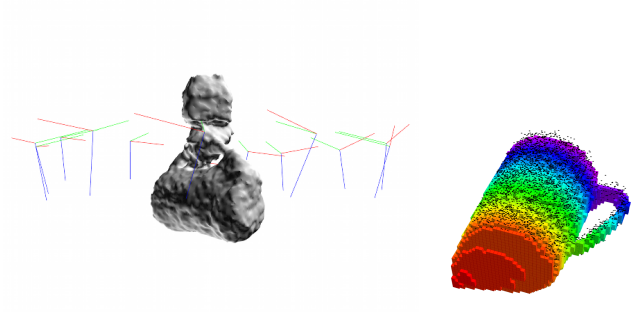
Fig. 6: (a)Precision vs. recall graph from varying r_s from 0 to 100 for several views of a random polygon. The graph cut is shown for the bet and worst values of r_s for 1 and 9 views.(b)The minimum cut of an 8-connected 2D voxel grid obtained by voxel-carving a random polygon. Three views are shown, with two prior radii shown for the sphere prior.

When $\beta = \infty$, for instance, we are left with a reconstruction of the surface of the object (which is less noisy than the point cloud); and when $\beta = 0$, we simply have the depth hull of the object. Varying β produces shapes which interpolate between the depth hull and the object surface.

Surface Reconstruction. Based on the carved voxel shape, the surface of the object can be reconstructed for visualization or analysis. Many algorithms may be used for this task, including the simplest, Marching Cubes [20]. Additionally, color can be applied to the reconstructed surface by the projection of each vertex onto the color image of the sensor. (Fig.2(b)) Admittedly, the surfaces we have constructed are quite ugly in comparison to state-of-the-art techniques from computer graphics (as in [6]) , but we are concerned with creating a probabilistically accurate representation of the occupied portions of the space for use in grasping and recognition – and not with constructing an aesthetically pleasing surface mesh of the object.

Object Recognition Once the shape of the object has been reconstructed using voxel depth carving, we may want to match the object to one of several in a database, or else classify the object. Recognizing objects from their volumetric representations is a very well-studied topic, known variously as Shape Retrieval or Shape Classification. The Princeton Shape Benchmark [27] lists several volumetric descriptors, including Shape Histograms, the Spherical Harmonic Descriptor, and others. The volumetric descriptor we chose for our experiments is the 3D Zernike [4] descriptor (as in [8]). The Zernike descriptor is constructed by projecting the object's voxelization onto a set of N basis polynomials defined on the unit sphere. It is affine invariant, compact, and descriptive.

Matching is performed by comparing the Zernike descriptor of the carved object with pre-computed Zernike descriptors in an object database, and taking the nearest neighbor in the Euclidean sense.



(a) (b)

Fig. 7: (a), Thirteen registered views of a kettle held in the robot's hand are used to carve a voxel grid, which is then meshed using Marching Cubes. The robot's hand is reconstructed as well. (b) A coffee mug scanned by a simulated Kinect at a depth of 3 meters in the intermediate stages of voxel carving. Voxels (colorful) are constructed from multiple noisy depth scans (black points).

4 Experiments

Voxel Carving Experiments We validated our method in two real-world setups: the first one is an object standing on a table, and the second one is an object in the robot's hand. When recording ray clouds, we have determined the pose of the laser scanner. Pose registration is beyond the scope of this work. For our purposes, we found that determining the pose of checkerboards on the wrist of the robot or on the table, respectively, with a camera mounted on the robot, is sufficient. The camera was also used to colorize the resulting model, as seen in Fig.2(b).

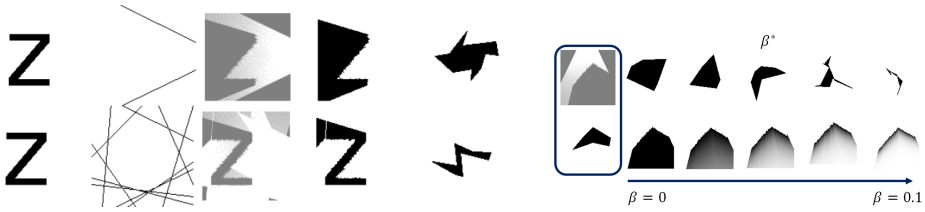
Object on the Table For the object on the table, we recorded seven ray clouds at a low resolution, taking about 1.5s per cloud. Each of the clouds contains about 64,000 points. After filtering out points lying outside the region of interest, about 10,000 points wer cloud remain. The point clouds after filtering are shown in Fig.2(c).

From each of the point clouds, we synthesize a depth image. After carving the voxel grid, we use the Marching Cubes algorithm and Laplacian smoothing to create a mesh from the voxel reconstruction. Finally, we use the camera images to colorize the mesh by re-projecting vertices onto RGB cameras co-registered with the laser scanner. More sophisticated methods for creating a mesh out of a voxel grid are available, but are beyond the scope of this work.

Creation of the depth images from the laser scan takes less than half a second. Reconstruction of the voxel grid takes about 14.7 seconds for a grid of 456×10^3 voxels. The time for generating and colorizing the mesh is less than one second. All of these results were obtained on an Intel Core 2 duo processor.

Object in the Hand We had the HERB hold a tea-kettle in its hand. HERB then rotated its wrist to take 13 scans of the object Fig.2(a). These scans were co-registered using a fiducial on HERB's wrist, and were then carved using the depth carving method. For this experiment, we did not colorize the mesh (Fig.7(a)).

Robustness to Noise For evaluating the robustness of our method to noise, we



(a) Effect of increasing views (b) Single view thin-ness prior

Fig. 8: 2D experiments. (a) Matching the letter Z to 10k random 2D polygons. The number of views increases from 1 to 4, and the match becomes more semantically similar. The leftmost image is the input, followed by the camera FOVs, the carved voxels, the occupancy probability, segmentation, and finally the top match. (b) As β increases from 0 to 0.1, we increase the weight of the object surface. At β^*, the match to the nearest polygon in the database is semantically closest. The leftmost column represents the occupancy probability, and the rightmost is the top match.

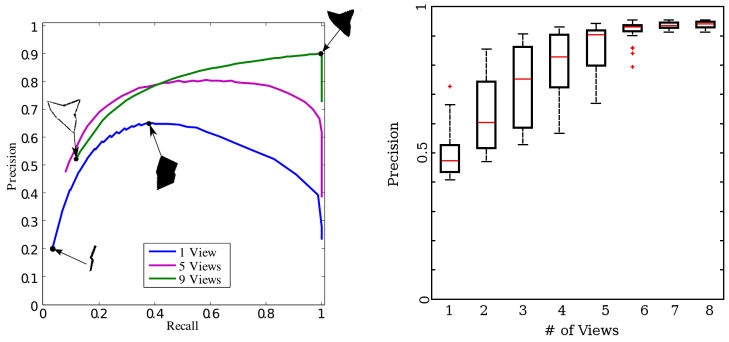
used the acquired data and added zero-mean Gaussian noise to the depth. Note that neither of these properties are required for our method – the probability distribution can be arbitrary and have non-zero mean. We chose a maximum acceptable misclassification probability of $P_{\text{mis, max}} = 0.2$, resulting in a margin of $\Delta d \approx 1\sigma$. The results of the reconstruction can be seen in Fig.1.

Fig.1(a) shows the original point cloud with no additional noise. Both the conventional method, using hit information, and our method, using depth images, approximate the shape well, but with the traditional method, part of the surface to the lower left is missing due to the low density of points. In Fig.1(b), noise with a standard deviation of $\sigma = 5$cm was added. The conventional method results in a very jagged surface. With even more noise (Fig.1(c), and Fig.1(d)), the object can barely be recognized in the point cloud anymore. The conventional method does not produce any useful results, while our method still performs well. Note that in Fig.1(d), the standard deviation of the noise is more than *half* the size of the object.

Simulation Environment To explore the effect of noise, multiple views, and object priors, we created both a 2D and 3D simulation environment. In the 2D environment, laser scans are simulated by casting 2D rays from the focal point of a camera in an arc. The rays strike a binary image representing an object. Noise is then added to the length of the rays. In our 3D simulation environment (Fig.7(b)), we simulate a *Kinect* with realistic resolution, noise parameters [22], and depth discretization using ray casting against 3D meshes.

Classification Accuracy with Shape Priors To explore the behavior of the object "thin-ness" prior, we varied the falloff parameter β over a number of views of the object. Voxels were said to be "occupied" if their occupancy probability was greater than 0.49, and were said to be "free" otherwise. After carving, the voxels were compared to the true object in simulation to see how accurately the space would be classified into "free" and "occupied" cells. We calculated Precision and Recall for each shape estimation where shape reconstruction is treated as a classification problem for each voxel independently.

In the 2D case, we took between 1 and 9 scans of a randomly generated polygon, and varied β for the thin object prior between 0 and 0.5; in another experiment we vary r_s in the sphere prior from 0 to 100. An example result is shown in Fig.9(a). In the 3D case, we took between 1 and 9 simulated scans



(a) 2D precision vs. Recall (b) 3D precision distribution

Fig. 9: Precision vs. Recall curves varying the thin-object prior's falloff parameter β. Each curve represents varying β for the same collection of scans. The most and least accurate reconstructions are shown for 1 and 9 views.

of objects from the Willow Garage Household Objects Database, and varied β between 0 and 40. Since the resolution of our 3D voxel grid was smaller than in the 2D case, Recall was very high for all scans, so instead we plot a distribution of precision vs. β (Fig.9(b)).

Moment Matching Experiments. We additionally explored the possibility of using volumetric shape descriptors to match objects with a database of known object models in our simulation environment.

In the 2D case, we generated 10,000 random polygons and rasterized them to 256×256 images. We then computed their Hu [15] moments, and stored them in a database. Novel test objects (Latin letters) were scanned in the simulation, and the Hu moments of their probabilistic shape functions were computed. The test objects were then matched to their nearest counterparts in the database by comparing their Hu moments in the Euclidean sense.

In the 3D case, we voxelized 256 objects in the Willow Garage Household database, and matched their 20-dimensional Zernike moments against scans of novel test objects. We compared the matching performance (with realistic sensor noise) while using only the hits to using Voxel Depth Carving (Fig.10).

5 Results and Main Experimental Insights

Voxel Carving Experiments. Figure 2 shows an intermediate stage during carving and the completely carved and colorized voxel grid. Figure 2(b) shows the colorized and smoothed trimesh generated from the voxel grid. Fig.1 shows the effect of Gaussian noise on voxel carving vs. reconstruction from the point cloud alone.

Even when the noise of the sensor approaches the scale of the object, our method is able to extract its shape. Our analysis indicates that our robustness to noise is greatly enhanced by integrating *near misses* (Fig.2(c)) of the object, which discriminate the object from the background even when the noise is exceedingly high. In contrast, the pointcloud rapidly decays with noise, losing its descriptive power.

(a) Matching a novel object

(b) Matching a known object

Fig. 10: The top 4 matches according to the L2 norm of Zernike Moments to a voxel-carved pitcher (far left) from 3 simulated views from a Microsoft *Kinect* sensor. The top row shows matches using hits only. The bottom row shows matches using passthrough information

Classification Accuracy and Shape Priors Even when only one view of the object is given, using a simple prior on object thin-ness greatly affects occupancy classification accuracy (Fig.9(a)). The prior becomes less important as the number of views increases (Fig.9(b)). This is because regardless of the prior, as more views are taken the set of un-carved voxels approaches the true shape of the object. When $\beta = 0$, the estimated object shape is simply a representation fo the object's surface which is much less noisy than the input point cloud. A similar effect happens when we vary r_s for the sphere prior. When $r_s = 0$, we again get a representation of the object's surface, and as r_s increases toward infinity, the occupied space approaches the depth hull.

Moment Matching Experiments. Figure 8(a) shows the effect of multiple views on voxel carving and the resulting top match from Hu moments in a database of 10k random polygons and the simple prior that voxels which are not carved are occupied. With 1 view, the match is not very semantically similar to the letter Z. With 4 views, the match is closer.When using the naive prior that un-carved voxels are occupied, our method fails to reconstruct the shape from a single view, instead leaving a long "shadow" behind the object. This result highlights the need for strong priors on the shape of objects in the scene given depth measurements. Figure 8(b) shows how the thin object prior can be used to find a good match via moments, even when there is only a single view of the object. As the weight on surface voxels increases, the occupancy probability tends to favor the thin shell of the object over the interior.

Figure 10 shows the top four matches from the Willow Garage Household Objects database to a novel test object (a pitcher, not already in the database), and a known object according to the L2 norm of their 20-dimensional Zernike moments. The top row shows matches using the hit information only, the bottom row shows matches using passthrough information. In the experiment with the known test object (Fig.10(b)) viewed under favorable conditions, the true object is the second match in the database using our method, whereas it is fourth when using only the hits. In the experiment with the novel object (Fig.10(a)), the handle's interior was not fully visible, and so during voxel carving it was left occupied – while the hit data leaves the hole unoccupied.

This result shows again the need for strong priors on occupancy when using voxel carving. Nevertheless, the first two object matches (both water filters) are more semantically similar to the test object than the first two matches using only the hits. We are left to wonder whether Zernike descriptors are strong enough descriptors to capture the kind of information we desire about manipulable ob-

jects. Indeed, the fact that Zernike descriptors are affine-invariant might actually be an undesirable property when considering household objects – whose uses are strongly correlated with scale.

6 Conclusion

We have shown that Occpuancy Grid Mapping techniques taken from robotic navigation is useful in a robotic manipulation setting. When only noisy, partial views of the object of interest are given, a volumetric approach to object modelling and recognition can be more useful than a surface-centric or point-based approach. Our experiments show that voxel carving leads to much more robust shape reconstruction under highly noisy, low-resolution conditions than methods which use point clouds alone. Using techniques from graph-based image segmentation, we introduced an efficient way to reconstruct 3D shapes with the minimum graph cut of a Markov Random Field. By estimating the occluded volumes of sensed objects, we are able to utilize implicit information from depth sensors which would otherwise be thrown away in surface or point-based models.

References

1. P. Besl. A Method for Registration of 3-D Shapes. In *IEEE Trans. Pattern Anal. Mach. Intell.*, 1992.
2. Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient n-d image segmentation. *Int. J. Comput. Vision*, 70(2):109–131, November 2006.
3. J.E. Bresenham. Algorithm for computer control of a digital plotter. *IBM-S.J*, 4(1):25–30, 1965.
4. N. Canterakis. 3D Zernike moments and Zernike affine invariants for 3D image analysis and recognition. *In 11th ICSA*, 1999.
5. A Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer (Long. Beach. Calif).*, 1989.
6. Renato Pajarola Fabio Guggeri1, Riccardo Scateni1. Shape Reconstruction from Raw Point Clouds using Depth Carving. *Eurographics*, 2012.
7. L. R. Ford and D. R. Fulkerson. A simple algorithm for finding maximal network flows and an application to the hitchcock problem. *Canadian Journal of Math.*, 1957.
8. C. Goldfeder. Data-driven grasping. *Autonomous Robots*, 31(1):1–20, April 2011.
9. M. Herrmann. Exploiting Passthrough Information for Multi-view Object Reconstruction with Sparse and Noisy Laser Data. 2010.
10. Stefan Hinterstoisser, Vincent Lepetit, and Slobodan Ilic. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. *ACCV*, 7724:548–562, 2013.
11. D. Hoiem and S. Savarese. Representations and Techniques for 3D Object Recognition and Scene Interpretation. *Synthesis Lect. on AI and Machine Learning*, 5(5):1–169, August 2011.
12. A. Hornung and L. Kobbelt. Robust Reconstruction of Watertight 3D Models from Non-uniformly Sampled Point Clouds Without Normal Information. *SGP*, 2006.
13. A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: an efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, February 2013.
14. K. Hsiao, M. Ciocarlie, and P. Brook. Bayesian grasp planning. *ICRA 2011*, 2011.
15. M.K. Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, pages 66–70, 1962.
16. K.N. Kutulakos and S.M. Seitz. A theory of shape by space carving. *ICCV*, 1, 1999.
17. K. Lai and D. Fox. Object Recognition in 3D Point Clouds Using Web Data and Domain Adaptation. *IJRR*, 29(8):1019–1037, May 2010.

18. A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2), 1994.
19. Y. Li, X. Wu, Y. Chrysathou, and A. Sharf. GlobFit: consistently fitting primitives by discovering global relations. *ACM Transactions on*, 2011.
20. W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph*, 21(4):163–169, 1987.
21. Z.C. Marton, R.B Rusu, and M. Beetz. On fast surface reconstruction methods for large and noisy point clouds. *2009 ICRA*, pages 3218–3223, May 2009.
22. C.V. Nguyen, S. Izadi, and D. Lovell. Modeling Kinect Sensor Noise for Improved 3D Reconstruction and Tracking. *3D Imaging, Modeling*, pages 524–530, October 2012.
23. M.J Olsen. Avoiding Indicents with Incidence. *LIDAR Magazine*, 2.2, 2012.
24. D.K. Prasad. Survey of the problem of object detection in real images. *IJIP*, (6):441–466, 2012.
25. R.B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). *2011 ICRA*, pages 1–4, May 2011.
26. J.R. Shewchuk and J.F.O Brien. Spectral Surface Reconstruction from Noisy Point Clouds. *SGP*, 14, 2004.
27. P. Shilane and P. Min. The princeton shape benchmark. *Shape Modeling*, 08540, 2004.
28. A. Staples-moore. Network flows and the max-flow min-cut theorem.