Detecting and segmenting objects for mobile manipulation

Andreas Holzbach

2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops

Detecting and Segmenting Objects for Mobile Manipulation

Radu Bogdan Rusu, Andreas Holzbach, Michael Beetz
Intelligent Autonomous Systems, Technische Universität München
Boltzmannstr. 3, Garching bei München, 85748, Germany
{rusu, holzbach, beetz}@cs.tum.edu

Gary Bradski
Willow Garage, Inc.
68 Willow Road, Menlo Park, CA 94025, USA
{bradski}@willowgarage.com

Abstract

This paper proposes a novel 3D scene interpretation approach for robots in mobile manipulation scenarios using a set of 3D point features (Fast Point Feature Histograms) and probabilistic graphical methods (Conditional Random Fields). Our system uses real time stereo with textured light to obtain dense depth maps in the robot's manipulators working space. For the purposes of manipulation, we want to interpret the planar supporting surfaces of the scene, recognize and segment the object classes into their primitive parts in 6 degrees of freedom (6DOF) so that the robot knows what it is attempting to use and where it may be handled. The scene interpretation algorithm uses a two-layer classification scheme: i) we estimate Fast Point Feature Histograms (FPFH) as local 3D point features to segment the objects of interest into geometric primitives; and ii) we learn and categorize object classes using a novel Global Fast Point Feature Histogram (GFPFH) scheme which uses the previously estimated primitives at each point. To show the validity of our approach, we analyze the proposed system for the problem of recognizing the object class of 20 objects in 500 table settings scenarios. Our algorithm identifies the planar surfaces, decomposes the scene and objects into geometric primitives with 98.27% accuracy and uses the geometric primitives to identify the object's class with an accuracy of 96.69%.

1. Introduction

In this paper we develop some key visual capabilities to enhance a mobile robot's ability to manipulate objects in the world. We concentrate on the manipulation needs within the immediate area where the robot's arms can reach which we call the robot's "workspace". For us, the fact that our robot is mobile simply means that we can't depend on instrumenting the external world with active vision systems or special lighting, but we can put such devices on the robot. In our case, we use projected texture to enhance stereo depth perception. We focus here on 3D depth features, adding 2D imagery will be studied in future work.

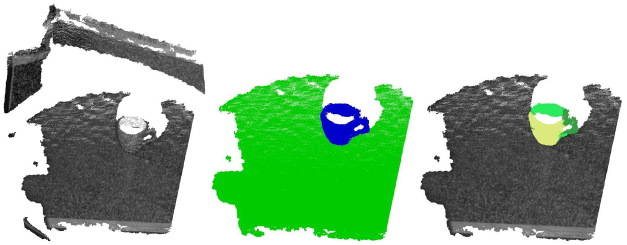

Figure 1. From left to right: raw point cloud dataset, planar and cluster segmentation, and the classification of the cluster using point based FPFH features.

The 3D visual capabilities that we desire for mobile manipulation are: the robot needs to be able to interpret surfaces, for example to know which surfaces support, contain or partially occlude objects; the robot needs to segment the objects from their support to know what it is that can be handled; the vision system needs to find the object's pose in 6 degrees of freedom (6DOF) so that the robot knows where the object is; the robot should be able to decompose the object into geometric primitives and recognize handles or more generally object affordances for grasping; and finally, the robot should be able to recognize the object's class in order to index its knowledge of such object sets.

To achieve the above aims, we use a two step process (depicted in Figure 2) in which we first classify points with respect to local surface features in order to determining stable grasp regions for manipulation applications. This is different from approaches where only distinctive points are detected for the purpose of object recognition. In the second step we classify objects into object categories based on relations between these different surface components that have been detected before.

In order to accurately and robustly classify points with respect to their underlying surface, we adopt as our starting point the Fast Point Feature Histogram (FPFH) [13] as described in Section 5.1 which has shown good results in interpreting the surfaces of scenes in terms of geometric primitives (here: planar, concave cylinder, convex cylinder and a class for handles and stems - see Figure 2).

Previously, FPFH has only been used with dense 3D point clouds resulting from panning laser scanners. Here, we move to dense stereo point clouds using projected light textures. The advantage of using stereo over panning laser scans is that it is much faster (30Hz-60Hz vs. 5 seconds to pan), but at the cost of more noise. The advantage of textured rather than structured light is that 2 robots can look at the same scene without interference. Using the resulting stereo depth map, we compute normal vectors and then segment the points into planar surfaces and points supported by planes. The new FPFH features are estimated and then fed into a Conditional Random Field (CRF) which labels local surface patches with our geometric primitives. At this point surface types are labeled at each place in the depth map giving the robot a basic interpretation of the scene. Next, we propose a novel feature, the Global Fast Point Feature Histograms (GFPFH) consisting of a histogram of geometric primitives at each patch and feed the GFPFH features into a Support Vector Machine model in order to label the whole object as an object classes. We then have what we sought: the robot has object locations (from the depth map), the parts of the object in terms of geometric primitives, and the earlier planar surface segmentations.

We tested our system on twenty cups and bowls from IKEA, see Figure 9. Figure 1 shows an example segmentation of a cup on a table. A database of images, point clouds and 3D models of these IKEA objects will be release publically for comparison testing. We chose IKEA objects because they are sold worldwide and anyone may then replicate this database.

Our current robot [1] uses stereo cameras as its primary vision system. In order to increase the density of the depth map in the robot's workspace, we employ alternating flashes of textured light. When the texture is on, dense stereo correspondence can be found, when the texture is off we get 2D imagery (which is not used in this paper). The stereo algorithm was developed in [10] and uses the implementation in the OpenCV library [2] as described in detail in [4]. In Figure 3, the left column shows the untextured left stereo image at top and the corresponding disparity map at bottom; in the right column we see the image at top when the texture is on and the corresponding disparity map at bottom right.

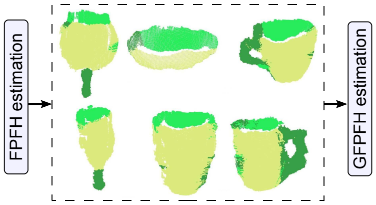

Figure 2. Examples of layer-1 point annotations for individual objects in the database using our modified FPFH variant. The color legend is: dark green for handle/stem, lighter green for convex cylinders, and greenish-yellow for concave cylinders.

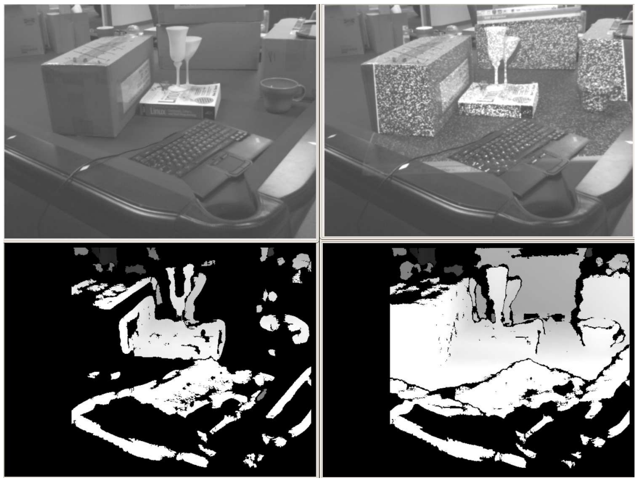

Figure 3. Textured light in the right column yields a denser disparity map.

The key contributions of the research reported in this paper include the following: i) a modified weighting scheme for the FPFH; ii) development of GFPFH for sets of points; iii) using FPFH (layer 1) + CRF for point classification; and finally iv) using GFPFH (layer 2) + SVM for object classification.

The structure of this paper is as follows. Related work is described in Section 2. Next, we give a brief description of our system architecture in Section 3. We present the segmentation algorithm in Section 4 followed by a discussion of the features used in Section 5. Our two layer classification formulation is described in Section 6 followed by results in Section 7. Conclusions are described in Section 8.

2. Related Work

The problem that we are trying to solve tackles both local (3D point level) and global (3D object level) classification based on estimated features. This has been under investigation for a long time in various research fields, such as computer graphics, robotics, and pattern matching, see [3, 8, 16] for comprehensive reviews. We address the most relevant work below.

Some of the widely used 3D point feature extraction approaches include: spherical harmonic invariants [5], spin images [9], curvature maps [6], or more recently, Point Feature Histograms (PFH) [13], and conformal factors [12]. Spherical harmonic invariants and spin images have been successfully used for the problem of object recognition for densely sampled datasets, though their performance seem to degrade for noisier and sparser datasets [3]. Our stereo data is noisier and sparser than typical line scan data which motivated the use of our new features. Conformal factors are based on conformal geometry, which is invariant to isometric transformations, and thus obtains good results on databases of watertight models. Its main drawback is that it can only be applied to manifold meshes which can be problematic in stereo. Curvature maps and PFH descriptors have been studied in the context of local shape comparisons for data registration. A side study [14] applied the PFH descriptors to the problem of surface classification into 3D geometric primitives, although only for data acquired using precise laser sensors. A different point fingerprint representation using the projections of geodesic circles onto the tangent plane at a point $\boldsymbol{p}_i$ was proposed in [15] for the problem of surface registration. As the authors note, geodesic distances are more sensitive to surface sampling noise, and thus are unsuitable for real sensed data without a priori smoothing and reconstruction. A decomposition of objects into parts learned using spin images is presented in [7] for the problem of vehicle identification.

Our local point classification is based on an extension of the recently proposed FPFH descriptors [13], coupled with Conditional Random Field models that classify noisy data acquired using our stereo setup into basic primitive partial shapes. To obtain a global object classification from the annotated point labels, we developed a novel Global FPFH descriptor which inherits some of the theoretical aspects of the local FPFH.

3. System Architecture

The architecture of our system together with the geometric and learning processing pipeline is presented in Figure 4. For every point cloud $\mathcal{P}$ acquired from the stereo system, our pipeline first estimates the underlying surface normal estimates $\boldsymbol{n}_i$ at every point $\boldsymbol{p}_i \in \mathcal{P}$, by approximating them with the normals of least-squares planes fit to local k-nearest neighbors patches centered around each $\boldsymbol{p}_i$. By transforming the input data into the robot coordinate system ($\mathcal{Z}$ pointing upwards), and using the previously estimated surface normals we devise a parallelized segmentation scheme for all horizontal planar surfaces $\mathtt{t}_i \in \mathcal{T}$ sampled in $\mathcal{P}$ using robust MSAC (M-Estimator Sample Consensus) estimators [17]. For segmentation we search nearby horizontal planes and extract all Euclidean point clusters supported by the plane (i.e., sitting on it). Since these clusters are independent with respect to each other, we start estimating local FPFH 3D features at every point in a cluster, for each cluster in parallel. Using a previously learned model (a CRF that assigns geometric primitive labels), the system outputs local point classes as the first annotation layer.

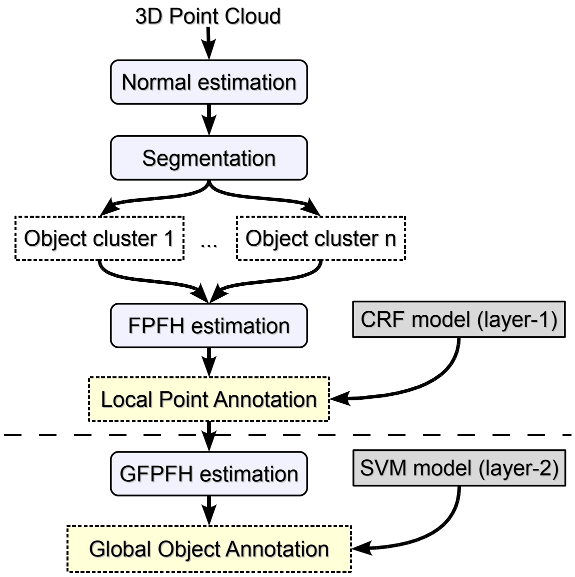

Figure 4. The architecture of our mapping system, with the major processing steps. The two classification layers produce local point annotations (layer-1), and global object annotations (layer-2) respectively.

The local point annotations are then used by the GFPFH scheme to compute a global representation for each such cluster. The second annotation layer into object classes is then obtained by using an SVM to classify each cluster with an object class lable.

In the following sections we will discuss each of the above mentioned steps separately and give insight on their implementation details.

4. Segmentation

The reason behind using a segmentation scheme a priori to the actual feature estimation is that in robotic manipulation scenarios we are only interested in certain precise parts of the environment, and thus computational resources can be saved by tackling only those parts. Here, we are looking to manipulate reachable objects objects that lie on horizontal surfaces. Therefore, our segmentation scheme proceeds at extracting these horizontal surfaces first.

To speed up the planar segmentation we proceed as follows. For a given point cloud dataset $\mathcal{P}$, we construct a downsampled version of it, $\mathcal{P}_d$, where $\boldsymbol{p}_j \in \mathcal{P}_d$ represents the centroid of a set of points $\mathcal{P}_j = \{\boldsymbol{p}_i \in \mathcal{P}\}$ obtained using spatial decomposition techniques (e.g. octree). We then search for planar models using MSAC [17] in $\mathcal{P}_d$ but constrain the sample selection step such that for every pair of points $\boldsymbol{p}_i, \boldsymbol{p}_j$ (with their estimated surface normals

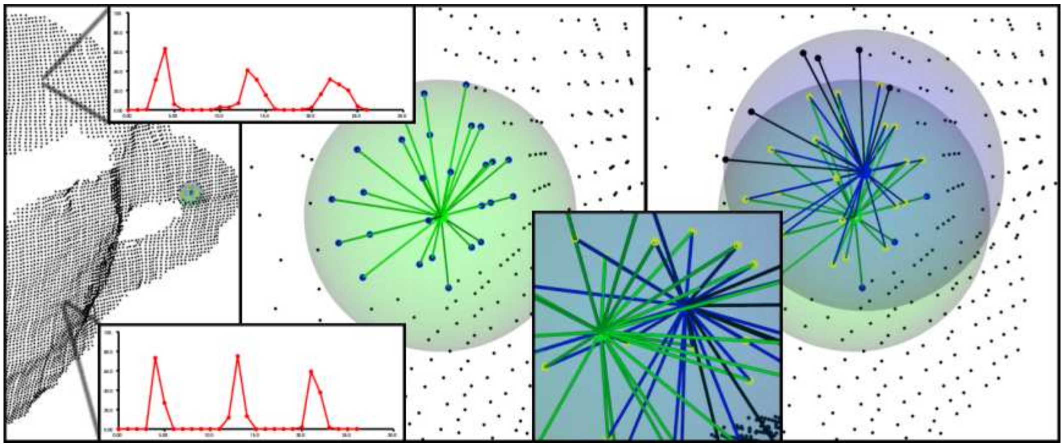

Figure 5. The estimation of a FPFH feature for a 3D point. From left to right: a point $\boldsymbol{p}$ (green) is selected from a dataset, and a set of $\mathcal{P}_k$ neighbors (black) enclosed in a sphere (green) with radius r are selected. For each pair of points $\langle \boldsymbol{p}, \boldsymbol{p}^k \rangle$, $\boldsymbol{p}^k \in \mathcal{P}_k$ (connections showed with green lines), three angular features are computed as presented in Figure 6. These steps are repeated for each neighbor as shown in the right part of the figure, and the FPFH of $\boldsymbol{p}$ will be weighted with the estimated features of all its $\boldsymbol{p}^k$ neighbors, as presented in Equation 2.

$\boldsymbol{n}_i, \boldsymbol{n}_j$) of each set of three points that could define a plane: $\boldsymbol{n}_i \cdot \boldsymbol{p}_j \approx 0$. Furthermore, we only look at points which have their estimated surface normals $\boldsymbol{n}_i$ approximatively parallel with the world $\mathcal{Z}$-axis, i.e. $\boldsymbol{n}_i \times \mathcal{Z} \approx 0$. Once a set of models has been obtained, we perform a fast Euclidean clustering of the selected points and construct a set $\mathcal{T} = \{\mathtt{t}_1 \cdots \mathtt{t}_n\}$ of *table* candidate clusters. The heuristic for selecting likely $\mathtt{t}_i$ candidates includes a weighted proportion of the number of inliers which support $\mathtt{t}_i$ as well as their proximity to the camera viewpoint. This approach emphasizes search in that part of the space where the robot manipulators can reach and grasp the objects.

To segment a set of object candidates on the chosen $\mathtt{t}_i$ table surface, we first create a bounding 2D polygon for it. Then, we look at the points whose projection on the $\mathtt{t}_i$ model falls inside the polygon. The result of these processing steps is a set of Euclidean point clusters $\mathcal{C}^i = \{\mathtt{c}_1^i \cdots \mathtt{c}_n^i\}$. An example can be seen in Figure 1.

To resolve further ambiguities with respect to the chosen candidate clusters, such as objects stacked on other planar objects (such as books), we repeat the previously mentioned step by treating each additional horizontal planar structure on top of the $\mathtt{t}_i$ table as a table itself and repeating the segmentation step (see results in Figure 13).

5. Feature Estimation

Our feature estimation scheme follows a twofold approach. First we estimate local point features using a modified version of the FPFH descriptors [13] to learn classes of geometric primitives using a CRF model. Then, we take the estimated point labels and combine them together in a Global FPFH scheme, to learn the object classes.

5.1. Local Fast Point Feature Histograms (FPFH)

The FPFH formulation as described in [13] uses pairs of points $\boldsymbol{p}_i, \boldsymbol{p}_j$, where $\boldsymbol{p}_j$ is said to be in the neighborhood of $\boldsymbol{p}_i$, and their estimated surface normals $\boldsymbol{n}_i, \boldsymbol{n}_j$ to create a multi-dimensional feature histogram space that represents the underlying surface geometry. The computational steps of a FPFH for a given point $\boldsymbol{p}$ includes the estimation of a set of angular features between the point's normal $\boldsymbol{n}$ and the normals of all the other points situated in the neighborhood $\mathcal{P}_k$ of $\boldsymbol{p}$. To estimate these features, a local uvn coordinate system is defined at $\boldsymbol{p}$. Figure 6 presents a diagram which describes the aforementioned coordinate system and the three chosen angular features.

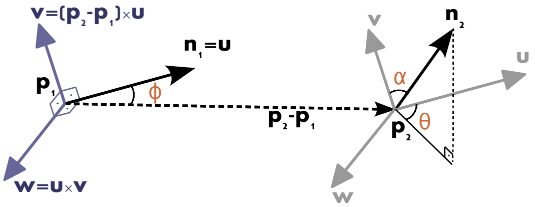

Figure 6. A graphical formulation of the three estimated FPFH angular features for a pair of points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ with their associated normals $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$.

The FPFH formulation for two points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ and their normals $\boldsymbol{n}_1$ and $\boldsymbol{n}_2$ is given as follows:

$$\begin{aligned} \alpha &= v \cdot \boldsymbol{n}_2 \\ \phi &= (u \cdot (\boldsymbol{p}_2 - \boldsymbol{p}_1))/\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|_2 \\ \theta &= \arctan(w \cdot \boldsymbol{n}_2, u \cdot \boldsymbol{n}_2) \end{aligned} \qquad (1)$$

where $\|\boldsymbol{p}_2 - \boldsymbol{p}_1\|_2$ represents the Euclidean distance between the two points.

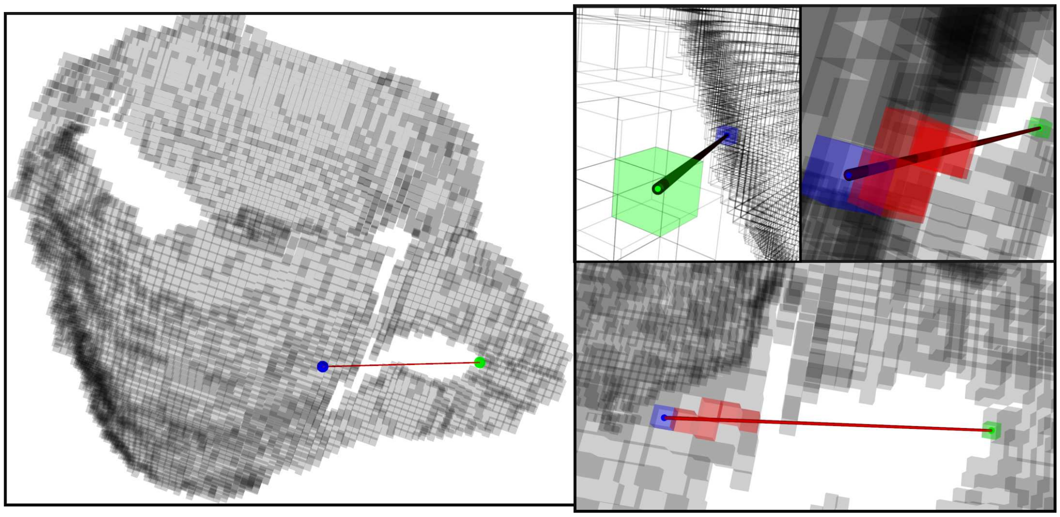

Figure 7. The estimation of a GFPFH for a 3D point cluster. After a voxelized representation is created (left), for every two pairs of leaves, a ray is casted from the start leaf (green) to the goal one (blue). All intersections with other leaves and free space are recorded and ported into a leaf class pair histogram (see Figure 8).

A set of $\langle \alpha, \phi, \theta \rangle$ triplets are computed betweeen a central point $\boldsymbol{p}$ and its neighbors. These triplets are binned into a multi-value histogram called SPFH (Simplified Point Feature Histogram) using an n^3 binning scheme described in [14], where n is the number of subdivision intervals in the feature value range. To capture surface structure across the neighborhood of $\boldsymbol{p}$, the procedure is repeated for each of the point's neighbors $\boldsymbol{p}^k$ (see the right side of Figure 5). The neighboring values are used to weight the final histogram (FPFH) of the point as follows:

$$FPFH(\boldsymbol{p}) = SPFH(\boldsymbol{p}) + \frac{1}{k}\sum_{j=1}^{k}\frac{1}{\omega_j^k}\cdot SPFH(\boldsymbol{p}_j^k) \quad (2)$$

The FPFH weight ω_j^k is given as the distance between the query point $\boldsymbol{p}$ and the $\boldsymbol{p}_j^k$ neighbor in [13]. In our experiments we noticed that this weight is penalizing the neighboring points on the surface too much. Therefore we adopted a different weight ω_j^k as:

$$\omega_j^k = \sqrt{\exp \|\boldsymbol{p} - \boldsymbol{p}_j^k\|} \quad (3)$$

which leads to a larger influence of the neighboring points $\boldsymbol{p}^k$ features in the resultant FPFH for the query point $\boldsymbol{p}$. Using the new weighting scheme improved the overall classification results (see Table 1).

5.2. Global Fast Point Feature Histograms (GFPFH)

As shown, the Fast Point Feature Histograms are local descriptors that can be used to classify various types of surface geometries around a chosen point $\boldsymbol{p}$. In the following, we generalize the FPFH idea to create a feature that captures the relationship of local geometric parts in whole objects. We call this new feature the Global Fast Point Feature Histogram (GFPFH).

The GFPFH computational steps are presented in Algorithm 1. The method assumes that for a set of input given points $\boldsymbol{p}_i \in \mathcal{P}$, a geometric primitive type class label $\boldsymbol{c}_i \in \mathcal{C}$ has already been assigned through an a priori point based classification step using FPFH descriptors. The algorithm begins by building an octree representation that spatially decomposes $\mathcal{P}$ into a set of *leaves* $\mathcal{L}$. Each leaf $\boldsymbol{l}_j \in \mathcal{L}$ contains a set of points $\mathcal{P}_j^l$ inside it, and due to the octree representation, each leaf is encapsulated into a parent leaf. This allows us to quickly create a multi-lod (multiple levels of detail) representation for the resultant GFPFH scheme.

A set of class probabilities $\mathsf{P}_{\boldsymbol{l}_i}(\boldsymbol{c}_j)$ are created for each leaf, where $\mathsf{P}_{\boldsymbol{l}_i}$ represents the probability of leaf $\boldsymbol{l}_i$ being of class $\boldsymbol{c}_j$, and is estimated as:

$$\mathsf{P}_{\boldsymbol{l}_i}(\boldsymbol{c}_j) = \frac{\mathsf{n}_j}{\mathsf{n}}\cdot 100, \quad (4)$$

where n represents the total number of points in $\mathcal{P}_i^l$, and n_j represents the number of points having been estimated as class label j using the FPFH CRF model.

Then, for each pair of two leaves $\langle \boldsymbol{l}_i, \boldsymbol{l}_j \rangle$, a line segment r_{ij} is created, and a set of leaves $\mathcal{L}_{ij}$ intersecting with r_{ij} is obtained. Each leaf in $\mathcal{L}_{ij}$ is checked whether it's occupied or not (i.e., it contains points in it), and a histogram $\mathcal{H}$ is created for the pair, where:

$$\mathcal{H} = \mathsf{h}_{ij} = \begin{cases} 0, & \boldsymbol{l}_{ij} \text{ unoccupied} \\ \mathsf{P}_{\boldsymbol{l}_{ij}}, & \boldsymbol{l}_{ij} \text{ occupied} \end{cases} \quad (5)$$

This results in $\left((\mathsf{n}_l+1)^2-(\mathsf{n}_l+1)\right)/2 = (\mathsf{n}_l+1)\cdot\mathsf{n}_l/2$ histograms $\mathcal{H}$ of variable length, one for each pair of leaves including the unoccupied leaf, where n_l is the number of occupied leaves in $\mathcal{L}$. An example of such histogram is shown in Figure 8. For simplicity, we take the most dominant point class label as the leaf representative in the figure. The starting leaf $\boldsymbol{l}_i$ is represented with **S**, and its class is *7*, and the goal leaf $\boldsymbol{l}_j$ (represented with **G**) has class *6* as the most dominant one. Along the ray r_{ij}, three occupied leaves are intersected, with classes *7*, *6*, and *6*, and then three unoccupied ones.

Algorithm 1 GFPFH computational steps

```
$\mathcal{L} = \{\boldsymbol{l}_1 \cdots \boldsymbol{l}_m\}$   // set of octree leaves encapsulating $\mathcal{P}$
$\mathcal{C} = \{\boldsymbol{c}_1 \cdots \boldsymbol{c}_o\}$   // set of point classes (generated from FPFH)
for all $\boldsymbol{l}_i \in \mathcal{L}$
    $\mathsf{P}_{\boldsymbol{l}_i}(\boldsymbol{c}_j)$   // compute a list of probabilities $\mathsf{P}_{\boldsymbol{l}_i}$ of $\boldsymbol{l}_i$ being of class $\boldsymbol{c}_j$
for all $\boldsymbol{l}_i \in \mathcal{L}$
    // histogram holding the class probabilities of each leaf pair started from $\boldsymbol{l}_i$
    $\mathcal{H} = \{\mathsf{h}_k | \mathsf{h}_k = 0\}$
    for all $\boldsymbol{l}_j \in \mathcal{L}$
        $\mathsf{r}_{ij} \leftarrow (\boldsymbol{l}_i, \boldsymbol{l}_j)$   // create a line segment $\mathsf{r}_{ij}$ between $\boldsymbol{l}_i$ and $\boldsymbol{l}_j$
        $\mathcal{L}_{ij} = \{\boldsymbol{l}_{ij} | \boldsymbol{l}_{ij} = \mathcal{L} \cap \mathsf{r}_{ij}\}$ // get intersected leaves $\mathcal{L}_{ij}$ along the ray
        if $\boldsymbol{l}_{ij}$ unoccupied   // is $\boldsymbol{l}_{ij}$ a free, unoccupied leaf in $\mathcal{L}$
            $\mathsf{h}_{ij} = 0$
        else
            $\mathsf{h}_{ij} = \mathsf{P}_{\boldsymbol{l}_{ij}}$
    // count the class changes for neighboring leaves in $\mathcal{H}$ and store in $\mathcal{H}_f$
    $\mathcal{H}_f \leftarrow \mathcal{H}$
```

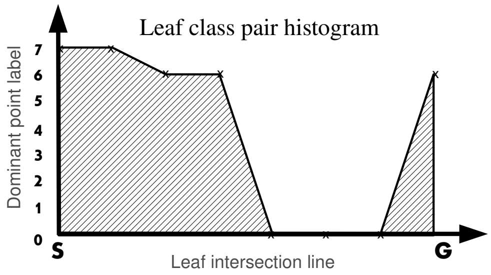

Figure 8. An example of a resultant leaf class pair histogram, with the start leaf having the dominant class *7*, and the goal one *6*. Along the casted ray, three other occupied leaves are intersected, with classes *7*, *6*, and *6*, as well as three free leaves (represented with class *0* here).

Then, for each $\mathcal{H}_{ij}$ histogram, we estimate a fixed length histogram representation $\mathcal{H}_{f_{ij}}$. $\mathcal{H}_f$ consists of each possible combination of class transitions, including the empty leaves, where the tuples $\langle \text{class}_a, \text{class}_b \rangle$ and $\langle \text{class}_b, \text{class}_a \rangle$ are equivalent. This results in a fixed-length histogram of size $\left((\mathsf{n}_c+1)^2+(\mathsf{n}_c+1)\right)/2 = (\mathsf{n}_c+2)\cdot(\mathsf{n}_c+1)/2$ bins. The bins are filled in by counting which transitions are encountered along the line segment r_{ij}.

The final GFPFH representation is obtained by computing a distance metric from each histogram in $\mathcal{H}_{f_{ij}}$ to the mean histogram of the set, and binning the values in a final histogram $\mathcal{G}$. We made experiments with several metrics including the Histogram Intersection Kernel proposed by [13], but the best overall results were obtained using the Kullback-Leibler divergence:

$$d_k = \sum_{i=1}^{\frac{(\mathsf{n}_c+2)\cdot(\mathsf{n}_c+1)}{2}} (p_i^f - \mu_i) \cdot \ln \frac{p_i^f}{\mu_i} \tag{6}$$

where p_i^f and μ_i represent the $\mathcal{H}_f$ histogram value at bin i and the mean histogram of the entire set of $\mathcal{H}_f$ histograms at bin i respectively. Figure 7 presents some of the general theoretical aspects of the GFPFH estimation as presented in Algorithm 1.

6. Model Learning

Generative graphical models represent a joint probability distribution $p(x, y)$, where x symbolizes the observations and y the classification label. Naive Bayes and Hidden Markov Model are two examples of those models, but due to the inference problem this approach is not well applicable for fast labeling of multiple point histograms like in our case. Discriminative models, such as Conditional Random Fields, are a solution to avoid this disadvantage of generative models. They represent a conditional probability distribution $p(y|x)$. Its advantage is that there is no need to model the observations and we avoid making potentially erroneous independence assumptions among these features [11].

A CRF model can be formulated as:

$$\begin{aligned} p(y|x) &= \frac{1}{Z(x)} \prod_{c \in C} \psi_c(x_c, y_c) \\ &= \frac{1}{Z(x)} \prod_{(i,j) \in C} \psi_{ij}(y_i, y_j, x_i, x_j) \prod_{i=1}^{N} \psi_i(y_i, x_i) \end{aligned} \tag{7}$$

where $Z(x) = \sum_{y'} \prod_{c \in C} \psi_c(x_c, y')$. The factors ψ_c are potential functions of the random variables v_C within a clique $c \in C$.

The node and edge potentials can be written as

$$\psi_i(y_i, x_i) = \exp\{\sum_L (\lambda_i^L x_i) y_i^L\} \tag{8}$$

$$\text{and} \quad \psi_{ij}(y_i, y_j, x_i, x_j) = \exp\{\sum_L (\lambda_{ij} x_i x_j) y_i^L y_j^L\} \tag{9}$$

Learning in a Conditional Random Field is performed by estimating the node weights $\lambda_i = \{\lambda_i^1, \ldots, \lambda_i^L\}$ and the edge weights $\lambda_{ij} = \{\lambda_{ij}^1, \ldots, \lambda_{ij}^L\}$. The estimation is done by maximizing the log-likelihood of $p(y|x)$.

We train two different classifier models, the first one deals with annotating the surface into geometric classes (layer-1) and uses a CRF, the second one uses a SVM to label the objects (layer-2). For the first model, we estimate point features using our modified FPFH variant with 27-bins. This constitutes the input to the layer-1 CRF model, which annotates the points with geometric surface type labels. For the second model, we estimate the Global Fast

Point Feature Histogram (GFPFH) of the local geometric surface types and use a previously learned SVM model (see Section 7) to produce object class labels. We chose these two different classification techniques because the CRF takes advantage of local surface correlations, whereas at the top layer we just need to map separate features to object classes.

7. Experimental Results

To validate our proposed framework, we have performed several experiments of point and object annotation using the two-layered CRF/SVM classification approach. In particular, we have gathered over 500 datasets of table setting scenes containing objects such as the ones presented in Figure 9. Each dataset was segmented using the method described in Section 4. The resulting segmented point clusters averaged 600 points with an average radius of 3 cm. The radius used for the FPFH was 0.5 cm. The GFPFH used all the points in each segmented cluster.

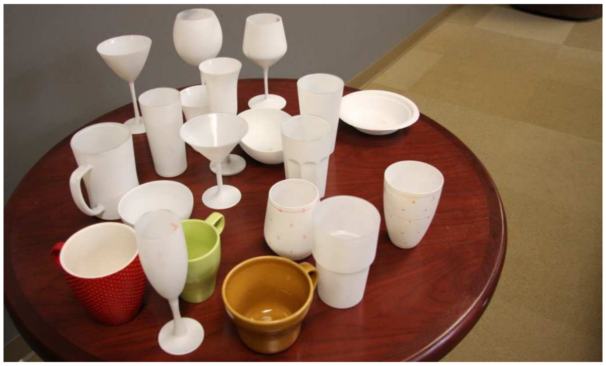

Figure 9. The complete collection of IKEA objects used in our experiments.

Due to the nature of the objects selected for these experiments, we defined four major geometric surface classes: planar, cylindrical concave, cylindrical convex, and stem or handle (represented by thin and slightly planar patches). Though a few glasses in the dataset include conical parts, we went with a computationally more efficient small radius for the FPFH features that was too small to reliably distinguish between conical from cylindrical surfaces. We found such distinction unnecessary for the goal of recognition of functional categories of tableware here.

We ran the local feature estimation three times for each point, computing the PFH and FPFH descriptors as described in [13], and our modified FPFH variant (see Section 5.1). Figure 2 presents 6 classified objects from the training dataset used to learn the layer-1 CRF model. Figure 10 and 13 present results obtained on various cluttered scenes. The overall classification results are presented in Table 1, together with the total computation time required for feature estimation, model learning and testing, on a standard Intel Centrino 1.2Ghz notebook, and the training error curves are given in Figure 12.

Table 1. Feature estimation, model learning, and overall testing results for the CRF layer-1 model. FPFH^m represents our modified FPFH variant.

Method	Feature Estimation (pts/s)	Model Training (s)	Model Testing (pts/s)	Accuracy
PFH	13.33	10.43	3936	57.51 %
FPFH	1128.22	4.84	1699	90.49 %
FPFH^m	1203.39	0.65	10087	98.27 %

Once all points were annotated using one of the three defined classes, our system estimates the Global FPFH histograms. The chosen FPFH radius of 0.5cm led to an average of ≈ 300 leaves and a mean computation time of ≈ 2 s per cluster on the 1.2Ghz notebook above. To train the model we selected a subset of 247 labeled objects, and devised four functional categories of objects: i) glasses (no handles or stems); ii) glasses with stems (e.g. wine glass); iii) mugs (with handles) and finally iv) bowls. We chose these classes because objects with similar geometry are grasped in a similar manner by our robot and thus they form functionally similar classes.

The object views were split into two equal parts, one for training the model and one for testing. The overall classification accuracy of the resultant SVM layer-2 model was 96.69%. Figure 11 presents the classification of the 20 types of objects tested into the 4 devised categories.

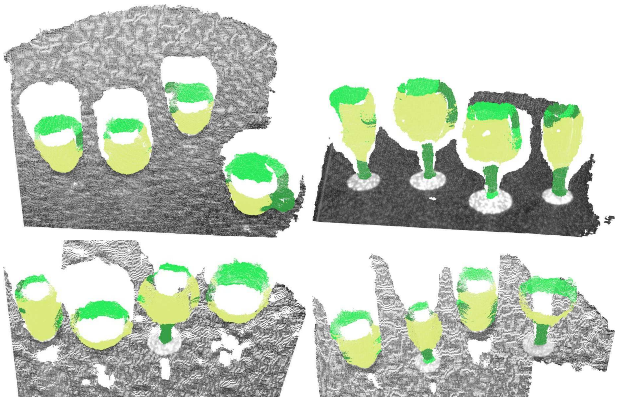

Figure 10. Example of scene segmentation and classification for datasets with multiple objects.

8. Conclusions and Future Work

In this paper we presented a novel scene interpretation approach for mobile manipulation scenarios. Our system uses a two layer classification scheme. A Conditional Random Field uses the previously proposed FPFH features [13] with a modified weighting scheme to annotate the individual points with geometric surface primitives. These local annotations were then represented in a novel global descrip-

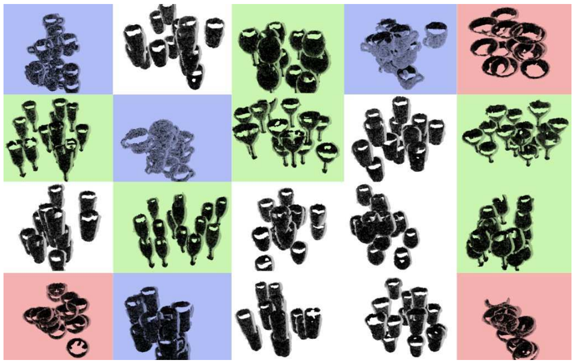

Figure 11. The classification results for the objects in our dataset using the 4 devised GFPFH classes: i) simple glasses (white); ii) glasses with stems (green); iii) mugs with handles (blue); and iv) large bowls (red).

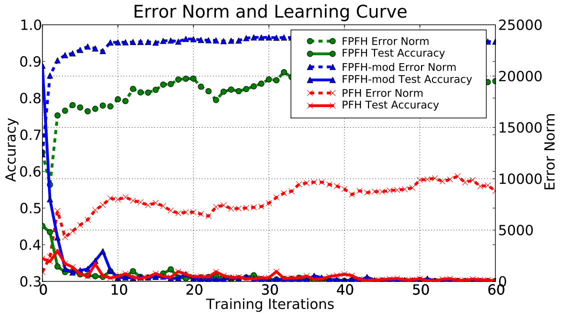

Figure 12. Classification accuracy and training error curves for the 3 different feature estimation methods: PFH, FPFH, and our modified FPFH.

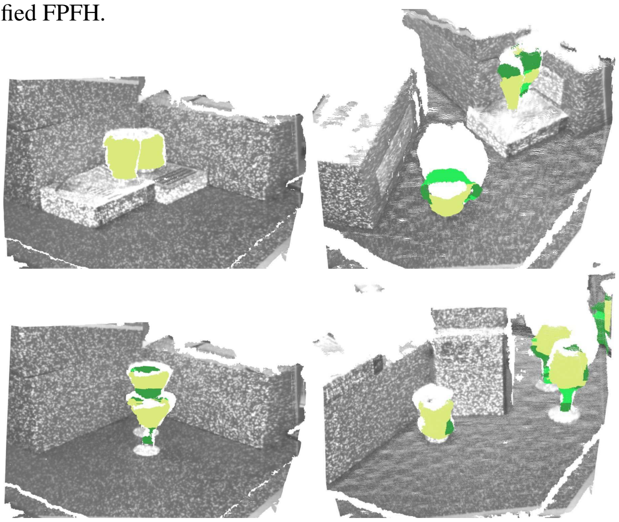

Figure 13. Example of more complex scene segmentation and classification for datasets with multiple cluttered objects.

tor GFPFH described in Section 5.2, which borrows from the techniques of the local FPFH descriptor. An SVM then uses the GFPFH descriptors to label the object classes. We validated our framework on a dataset of IKEA objects acquired using real time stereo with textured light, and obtained 98.27% accuracy on the local point interpretation using geometric primitives, and 96.69% accuracy for the identification of object classes. Currently this representation is being used for grasping work on our robot and we are adding 2D and 3D boundary features to further improve classification results.

References

[1] Our robot description, suppressed for anon. In *suppressed*.

[2] OpenCV, Open source Computer Vision library. In *http://opencv.willowgarage.com/wiki/*, 2009.

[3] A. D. Bimbo and P. Pala. Content-based retrieval of 3D models. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):20–43, 2006.

[4] G. Bradski and A. Kaehler. Learning OpenCV: Computer Vision with the OpenCV Library. In *O'Reilly Media, Inc.*, pages 415–453, 2008.

[5] G. Burel and H. Hénocq. Three-dimensional invariants and their application to object recognition. *Signal Process.*, 45(1):1–22, 1995.

[6] T. Gatzke, C. Grimm, M. Garland, and S. Zelinka. Curvature Maps for Local Shape Comparison. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005 (SMI' 05)*, pages 246–255, 2005.

[7] D. Huber, A. Kapuria, R. R. Donamukkala, and M. Hebert. Parts-based 3D object classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 04)*, June 2004.

[8] A. K. Jain and C. Dorai. 3D object recognition: Representation and matching. *Statistics and Computing*, 10(2):167–182, 2000.

[9] A. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3D scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, May 1999.

[10] K. Konolige. Small vision systems: hardware and implementation. In *In Eighth International Symposium on Robotics Research*, pages 111–116, 1997.

[11] J. Lafferty, A. Mccallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, 2001.

[12] B.-C. M. and G. C. Characterizing shape using conformal factors. In *Eurographics Workshop on 3D Object Retrieval*, 2008.

[13] R. B. Rusu, N. Blodow, and M. Beetz. Fast Point Feature Histograms (FPFH) for 3D Registration. In *In Proceedings of the International Conference on Robotics and Automation (ICRA)*, 2009.

[14] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz. Learning Informative Point Classes for the Acquisition of Object Model Maps. In *In Proceedings of the*

10th International Conference on Control, Automation, Robotics and Vision (ICARCV), 2008.

[15] Y. Sun and M. A. Abidi. Surface matching by 3D point's fingerprint. In *Proc. IEEE Int'l Conf. on Computer Vision*, volume II, pages 263–269, 2001.

[16] J. W. Tangelder and R. C. Veltkamp. A Survey of Content Based 3D Shape Retrieval Methods. In *SMI '04: Proceedings of the Shape Modeling International*, pages 145–156, 2004.

[17] P. Torr and A. Zisserman. MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78:138–156, 2000.