Unsupervised Feature Learning for 3D Scene Labeling

Kevin Lai, Liefeng Bo, and Dieter Fox

***Abstract*— This paper presents an approach for labeling objects in 3D scenes. We introduce HMP3D, a hierarchical sparse coding technique for learning features from 3D point cloud data. HMP3D classifiers are trained using a synthetic dataset of virtual scenes generated using CAD models from an online database. Our scene labeling system combines features learned from raw RGB-D images and 3D point clouds directly, without any hand-designed features, to assign an object label to every 3D point in the scene. Experiments on the RGB-D Scenes Dataset v.2 demonstrate that the proposed approach can be used to label indoor scenes containing both small tabletop objects and large furniture pieces.**

I. Introduction

As robots move from the factory floor into our homes and offices, they will need increasingly sophisticated tools for analyzing and understanding the environment around them. Semantic scene understanding is one important capability that robots must master if they are to execute commands like "fetch me the can of soda from the coffee table". In such applications, the robot must not only be able to recognize soda cans and coffee tables, but also to accurately infer their positions and spatial extent in 3D space. RGB-D cameras like the Microsoft Kinect have enabled rapid progress in robot perception. In this work, we study scene labeling of RGB-D videos.

There is a large body of literature on 3D scene labeling in indoor environments ([1], [2], [3], [4], [5]). Like approaches designed for outdoor environments ([6], [7]), many of these operate on a 3D point cloud, often obtained using a laser rangefinder. 3D point clouds contain very important shape and spatial information, which allows for models that take advantage of context and spatial relationships between objects [8], [9]. However, using an RGB-D camera it is possible to also collect data from a completely different modality. Not only can we now get the color of each 3D point, but we also have access to a set of RGB and depth image pairs. Existing works have shown that RGB-D images can be used to recognize objects to a high degree of accuracy [10], [11], [12], [13].

Another recent trend that has shown promising progress is unsupervised learning of feature representations directly from sensor data. Sparse coding and convolutional neural networks have shown promising results on image classification [14], [15], object detection [16], and scene understanding [17]. Unlike hand-designed features which are implicitly limited by the design choices of their creators, these algorithms are fully data-driven and can learn sophisticated representations that capture statistical patterns and relationships in the data.

In previous work [18], we presented a technique for 3D scene labeling of tabletop objects using RGB-D videos as input data. Sliding window object detectors were used to assign a class probability distribution to every pixel in every RGB-D video frame. A probabilistic Markov Random Field (MRF) over a voxel representation of the 3D scene was used to combine evidence from the whole video sequence. We demonstrated that it achieves excellent results on the RGB-D Scenes Dataset, which consists of 8 scenes with tabletop objects in kitchen and office environments.

This work builds on our previous work by modifying the MRF framework to integrate classifier responses from both the constituent RGB-D video frames and also responses from a classifier using features extracted from the 3D scene point cloud itself. We introduce HMP3D, a novel hierarchical sparse coding technique for learning features over 3D point cloud data. HMP3D adapts the hierarchical matching pursuit (HMP) algorithm [13], a state-of-the-art sparse coding technique for image data, for use in learning features for 3D point clouds. We also use HMP to learn sparse coding features for RGB-D image detectors [16]. The resulting system learns feature representations on both RGB-D images and 3D point clouds, and does not use any hand-designed features. HMP3D is trained using data generated from synthetic 3D scenes created using CAD models from an online database. We show excellent results on both the original RGB-D Scenes Dataset and the new RGB-D Scenes Dataset v.2, a new dataset of 14 indoor scenes containing tabletop objects and large furniture pieces.

II. 3D Scene Labeling from RGB-D Videos

Given a 3D point cloud of a scene, the scene labeling task is to assign an object class label to every point. We tackle the problem of labeling point clouds created by aligning a contiguous sequence of images from RGB-D videos, i.e. the output of 3D reconstruction techniques like Kinect Fusion [19] and RGB-D Mapping [20], [21]. Fig. 1 presents an overview of the proposed system. The system learns and extracts features on both the constituent RGB-D video frames, and on a voxel representation of the scene. The classifier responses from these two feature representations are combined using an MRF to produce a labeling of

This work was funded in part by the Intel Science and Technology Center for Pervasive Computing (ISTC-PC), ONR MURI grant N00014-07-1-0749, and the NSF under contract IIS-0812671.

Kevin Lai and Dieter Fox are with the Department of Computer Science & Engineering, University of Washington, Seattle, WA 98195, USA. {kevinlai,fox}@cs.washington.edu

Liefeng Bo is with Amazon Inc. The work was completed while he was affiliated with the Intel Science and Technology Center on Pervasive Computing, Seattle, WA 98195, USA. liefeng.bo@gmail.com

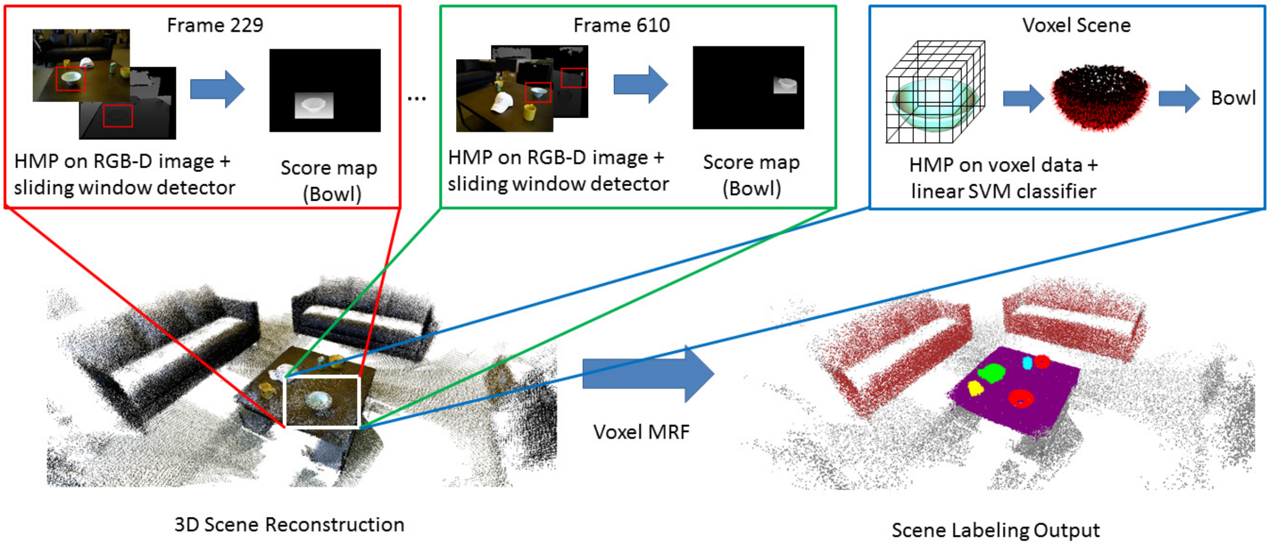
Frame 229
HMP on RGB-D image + sliding window detector
Score map (Bowl)
...
Frame 610
HMP on RGB-D image + sliding window detector
Score map (Bowl)
Voxel Scene
HMP on voxel data + linear SVM classifier
Bowl
Voxel MRF
3D Scene Reconstruction
Scene Labeling Output

Fig. 1. System Overview: Given a scene (left), the system extracts learned features from its constituent RGB-D video frames and the voxel representation obtained from the 3D scene point cloud. Classifiers are evaluated separately on these sensor modalities, and their responses are combined using a Markov Random Field to produce the scene labeling (right).

the scene. Images, being two-dimensional, capture objects situated in front of a background, which often provides useful context. On the other hand, 3D point clouds provide spatial information that is possible, but difficult, to estimate from images [22]. As we will demonstrate, it can be beneficial to learn and extract features from both. Another advantage of our system is that it does not require point/pixel-level labeling of scenes as training data, which is expensive to obtain. Instead, the proposed approach only requires images containing views of each objects in isolation and synthetic 3D models downloaded from an online database on the Web.

We use the Patch Volumes 3D reconstruction algorithm of Henry et al. [21] to estimate the six-dimensional camera poses of each RGB-D video frame and create the scene reconstruction. The system creates a voxel representation of the resulting 3D point cloud by discretizing it into a regular grid structure. The individual cells in this grid structure are called a voxel. Let each voxel v be associated with a label $y_v \in \{1, ..., C, c_B\}$, where $1, ..., C$ are object classes and c_B is the background class. As was done in [18], we model the joint distribution of voxel labels using a Markov Random Field with pairwise interactions. The optimal labeling of the scene minimizes the following energy:

$$E(y_1, \cdots, y_{|\mathcal{V}|}) = \sum_{v \in \mathcal{V}} \psi_v(y_v) + \sum_{\{i,j\} \in \mathcal{N}} \phi_{i,j}(y_i, y_j) \quad (1)$$

where $\mathcal{N}$ is the set of all pairs of neighboring voxels. We connect each voxel to its 26 adjacent voxels (9 below, 8 at the same height, 9 above). The data term (*first sum*) measures how well the assigned label fits the observed data and the pairwise term (*second sum*) models interactions between adjacent voxels.

In [18], the data term $\psi_v(y_v)$ in eq. 1 was computed using responses from sliding window detectors on RGB-D video frames. In this work, we also integrate responses from a classifier over the 3D voxel data into this term.

Voxel Classification: There is a voxel classifier for each object class we wish to label. Each voxel classifier assigns a likelihood of each voxel belonging to its class by extracting features from a rectangular prism shaped region around that voxel based on the expected true size of the object, which is estimated from training data. For example, in our experiments a coffee mug voxel classifier extracts features from a 16.2cm×16.8cm×14.4cm window. We describe the feature extraction process in Section III and the training process in Section IV. The voxel classifiers yield $p_{vox}(y_v|x)$ for each 3D point x falling inside the voxel v.

RGB-D Image Classification: We learn a single-layer dictionary by sampling 5×5 patches from grayscale and depth images and extract sparse coding features [23]. These features are then used to train object detectors [16]. For our experiments we only use a root template so we do not need to use HOG features to first learn a deformable parts structure. We train sliding window detectors for each object class. Each pixel in an RGB-D frame has a corresponding 3D point x, and as was done in [18], the detectors are used to compute a class distribution $p_{im}(y_v|x)$ for every pixel in every frame, where a higher probability means it is more likely that an object of that class is present at that pixel location.

The data term is computed as

$$\psi_v(y_v) = -\ln p(y_v|\Omega_v) = \\ -\frac{1}{|\Omega_v|} \sum_{x \in \Omega_v} \alpha \ln p_{vox}(y_v|x) + (1-\alpha) \ln p_{im}(y_v|x) \quad (2)$$

In other words, the log likelihood of a voxel v is the arithmetic mean of the log likelihoods of its constituent points Ω_v. The probabilities from voxel classifiers and image classifiers are multiplied (summed in log-space), with $0 \leq \alpha \leq 1$ being the tradeoff parameter for balancing their respective influence.

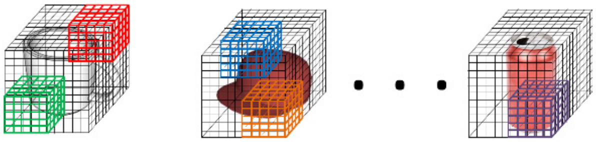

Fig. 2. HMP over voxel data: To learn the first-layer dictionary, $5 \times 5 \times 5$ voxel patches are randomly sampled from the entire dataset of objects.

The pairwise term between voxels in eq. 1 is defined to be

$$\phi_{i,j}(y_i, y_j) = \lambda \cdot \frac{\mathbf{1}_{y_i \neq y_j}}{d(n_i, n_j)} (\mathbf{I}(n_i, n_j) + \epsilon) \tag{3}$$

where λ and ϵ are balancing parameters and $\mathbf{1}_{y_i \neq y_j}$ evaluates to 1 when $y_i \neq y_j$ and 0 otherwise. This term, first proposed in [18], captures two aspects of 3D shape information that improve segmentation of objects. First, $d(n_i, n_j)$ measures the l_2 distance between surface normals n_i and n_j of voxels i and j. This captures the intuition that objects tend to contain smooth surfaces, so adjacent voxels with smooth surfaces should tend to share the same label. Second, $\mathbf{I}(n_i, n_j)$ is an indicator variable expressing whether i and j are part of a convex surface or a concave one. This captures the intuition that objects are often convex in shape, while transitions between objects and the surface on which it sits tends to be concave. ϵ determines the ratio between the penalty of assigning adjacent voxels different labels when the surface is convex or concave. We compute the term $\mathbf{I}(n_i, n_j) = [(n_i - n_j) \cdot (i - j) > 0]$ as proposed in [24].

The energy (eq. 1) of the multiclass MRF is efficiently minimized using graph cuts [25], which completes the process in less than 500 ms for a 5m×5m×5m scene.

III. Feature Learning for 3D Point Cloud Data

The problem of learning feature representations has been studied extensively in the computer vision and machine learning communities ([26], [27], [23], [15]). In particular, Hierarchical Matching Pursuit (HMP) is an unsupervised learning algorithm for learning multiple levels of feature representation in a bottom-up fashion that has been applied to RGB-D images and achieves state-of-the-art performance on object recognition [13] and attribute learning [28].

In this section we propose HMP3D, a method for learning hierarchical feature representations over 3D point clouds that uses the hierarchical matching pursuit algorithm. Given a 3D point cloud of an object, we first discretize the point cloud to create a voxel representation. Each voxel is described by a set of attributes computed from the 3D points falling inside the voxel, such as RGB value and surface normal vector. In the context of scene labeling, we wish to learn a feature representation over a dataset of objects stored in this voxel representation such that the features can be used for effective object recognition.

A. Dictionary Learning

The goal of dictionary learning is to compute a set of vectors (codes) such that the data can be represented using the linear combination of a sparse set of codes. Bo et al. [23] presented a method for learning features over RGB-D pixel data using the K-SVD algorithm [29]. For HMP3D, we apply the same technique on voxel data. For example, consider the case where each 3D voxel stores a surface normal vector computed from its local neighborhood. HMP3D learns a dictionary over this data by sampling $5 \times 5 \times 5$ voxels (i.e. $5 \times 5 \times 5 \times 3 = 375$-dimensional patches) to create a set of code words that can accurately reconstruct local geometry 5 voxels across each dimension (see Fig. 2). HMP3D applies the same technique to learn features over other voxel attributes, such as color and binary occupancy.

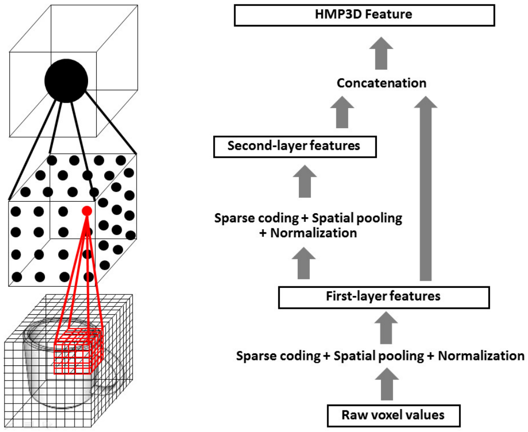
HMP3D Feature
Concatenation
Second-layer features
Sparse coding + Spatial pooling + Normalization
First-layer features
Sparse coding + Spatial pooling + Normalization
Raw voxel values

Fig. 3. HMP3D: Hierarchical matching pursuit for 3D voxel data. In the first layer, sparse codes are learned over $5 \times 5 \times 5$ patches of voxel attributes. These sparse codes are pooled into features representing $20 \times 20 \times 20$ patches using spatial pyramid max pooling. A second layer of sparse codes are learned for encoding these features using a dictionary from sampled first-layer features. The HMP3D feature representing the whole object is a concatenation of the spatial pyramid pooled and contrast normalized sparse codes from both the first and second layers.

B. HMP3D: Hierarchical Matching Pursuit for 3D voxel data

HMP3D uses the hierarchical matching pursuit algorithm [23] to build a two-layer feature hierarchy by learning dictionaries and applying the orthogonal matching pursuit encoder recursively (Fig. 3).

1) First Layer: The first layer generates features for $5 \times 5 \times 5$ voxel patches. Orthogonal matching pursuit is used to compute M-dimensional sparse codes representing each patch. Spatial pyramid max pooling is then applied to these sparse codes to generate patch level features. The voxel representation of the object is partitioned into several levels of spatial cells. The component-wise maximum over all sparse codes in a cell is taken to generate the sparse code of that cell. For example, consider a 3 level spatial pyramid pooling where the first layer divides the voxel grid into $3 \times 3 \times 3$ cells, the second layer into $2 \times 2 \times 2$ cells, and the final layer into 1 cell (see Fig. 4). The resulting feature vector describing the whole object would be $(27 + 8 + 1)M$-dimensional, where M is the size of the dictionary. Spatial pyramid pooling adds multiple levels of invariance to local deformation. The spatial nature of the pooling process increases the discriminative power of the

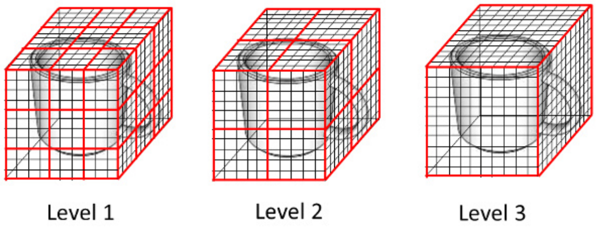

Fig. 4. 3D spatial pyramid max pooling divides voxels into several levels of cells. Sparse codes within each cell are combined by taking component-wise maximum.

resulting feature. Finally, feature vectors are normalized by their L_2 norm. This makes the features robust to changes in magnitude, which can happen, for example, due to changes in illumination in the case of grayscale intensity and RGB values.

2) Second Layer: The second layer of HMP3D computes a higher-level representation over larger regions of the object. The dictionary for the second layer is learned by sampling patches over the first-layer features. For example, when sampling $4 \times 4 \times 4$ first-layer patches, each of which is an M-dimensional sparse code vector, the second layer dictionary would be learned over regions spanning $20 \times 20 \times 20$ voxels and the sampled patches would be $4 \times 4 \times 4 \times M$-dimensional. Similar to the first layer, spatial pyramid pooling and contrast normalization is used to generate a fixed-length feature vector describing the entire object.

The final feature representation of the object is the concatenation of the spatial pyramid pooled features from the first and second layers. Note that Hierarchical Matching Pursuit is an unsupervised learning algorithm. These feature vectors are used in conjunction with class labels to train linear Support Vector Machine (SVM) classifiers.

IV. Synthetic Data Generation

Given a large set of real scenes with ground truth labels, HMP3D features and classifiers can be learned by sampling from the dataset. However, ground truth real data is very expensive and time consuming to obtain, especially for scene labeling where every 3D point must be annotated with its label. In this work, we explore the use of synthetic data to learn HMP3D features and train voxel classifiers that we then apply to real data. A key difficulty of using synthetic data is ensuring that classifiers learned using it remains effective when applied to real sensor data. Inspired by the success of virtual training data synthesis for human pose estimation from depth images [30], we generate a large synthetic training dataset using 3D models downloaded from Trimble 3D Warehouse [31], a large Internet repository of CAD models created by hobbyists and professionals.

To closely model real-world environments as seen from videos recorded using an RGB-D camera, synthetic object models are placed in automatically generated virtual scenes where they all lie on the same plane and may be very close to and/or occlude each other. Training data is generated by sampling rectangular prism shaped regions centered on objects in these virtual scenes. For each sampled object, raycasting is used to generate a 3D point cloud of the object aggregated from a randomly selected number of viewpoints between three to eight. Gaussian noise is added to simulate sensor noise in real data. While the sensor noise of RGB-D cameras like the Kinect increases with distance from the sensor and is thus non-Gaussian, our approximation worked well in practice because each object is imaged multiple times from a variety of distances in the videos we use for evaluation. The sampled object is scaled uniformly at random to be between 0.85 to 1 unit in length along its longest dimension (for a small amount of scale variance) and rotated by a random amount. A voxel representation for the object is obtained such that a unit length contains a constant number of voxels (80 in our experiments), yielding a representation that captures the same level of detail regardless of true object size. In our experiments, we compute as voxel attributes grayscale intensity, RGB value, binary occupancy, and surface normal vector. Objects are sampled this way until the desired training set size is reached.

Separate datasets are generated using the above procedure for each object class because the number of voxels along each dimension is dependent on the true size of the object. Positive examples are generated by sampling objects of the target class from the virtual scenes. Negative examples are generated by sampling objects from other classes, as well as regions that contain objects from the target class but where the object is positioned sufficiently off-center. As our experiments will demonstrate, the resulting training data sufficiently captures variance that may be encountered in real-world data: intra-class appearance and shape variation is captured by differences in the object models downloaded from Trimble 3D Warehouse, while scale, orientation, clutter, and occlusion variation is captured via the virtual environment generation and object sampling procedure.

V. Experiments

In this section, we evaluate the proposed system on a 3D scene labeling task and the HMP3D feature representation for 3D object recognition.

A. 3D Scene Labeling

To investigate the proposed scene labeling approach, we evaluate it on the task of detecting small tabletop objects and large furniture in typical home and office environments. We collected a new dataset containing 14 such scenes: the RGB-D Scenes Dataset v.2, which is publicly available at http://www.cs.washington.edu/rgbd-dataset. The original RGB-D Scenes Dataset [10] contains 8 scenes recorded in office and kitchen environments. Each scene contains between three to twelve objects placed on a flat surface, including bowls, caps, cereal boxes, coffee mugs, and soda cans. The new RGB-D Scenes Dataset v.2 augments these eight existing scenes with 14 new scenes recorded from a lounge, a coffee room, a meeting area, and an office desk. Each scene may contain furniture including chairs, coffee tables, sofas, and tables. Three to five objects

Technique	Precision / Recall										
	Bowl	Cap	Cereal Box	Coffee Mug	Soda Can	Coffee Table	Chair	Sofa	Table	Background	Overall
Det3DMRF	65.4 / 84.7	62.9 / 98.7	64.2 / 96.3	48.1 / 84.7	61.1 / 95.1	12.7 / 15.8	18.4 / 20.8	25.1 / 30.9	14.6 / 22.1	43.9 / 99.5	41.6 / 64.9
HMP2D	74.4 / 85.0	74.9 / 98.6	79.9 / 98.6	64.4 / 87.8	78.2 / 98.1	11.9 / 17.9	17.7 / 17.2	29.3 / 39.8	16.4 / 23.3	42.9 / 99.6	49.0 / 66.6
HMP3D	**100.0 / 96.2**	**91.3 / 98.2**	85.1 / 100.0	**90.0 / 93.9**	100.0 / 81.9	96.1 / 100.0	57.6 / 100.0	82.7 / 100.0	**95.3 / 98.7**	99.9 / 80.0	89.8 / 94.9
HMP2D+3D	97.0 / 89.1	82.7 / 99.0	**96.2 / 99.3**	81.0 / 92.6	**97.7 / 98.0**	**98.7 / 98.0**	**89.7 / 94.5**	**92.5 / 92.0**	97.6 / 96.0	**95.8 / 95.0**	**92.8 / 95.3**

TABLE I

PER-CATEGORY AND OVERALL (MACRO-AVERAGED ACROSS CATEGORIES) PRECISION AND RECALL ON THE RGB-D SCENES DATASET V.2.

from the same 5 categories as the original dataset are placed on each coffee table or table.

1) Experiment Setup: To train voxel classifiers, we downloaded models of objects from all nine classes from the Trimble 3D Warehouse. In addition, we also downloaded models of other objects commonly found in office and home environments for use as negative data. We then use the procedure for generating synthetic training data described in Section IV to generate approximately $100,000$ training examples. In this procedure, each virtual scene contains only either tabletop objects or furniture. The real-world dimensions of the voxel representations of each object class were computed from training data. Images from the RGB-D Object Dataset [10] are used to train RGB-D image sliding window detectors for the tabletop objects.

Often a robot can accurately estimate the location of the ground plane and objects are often placed on top of furniture. Taking advantage of such prior knowledge about the environment often improves both labeling accuracy and computational efficiency. In our experiments, we assume that furniture is always placed on the ground and tabletop objects are always placed on tables and the ground plane is known, which is true for RGB-D Scenes Dataset v.2. The ground plane is removed and a region growing technique is used to segment out isolated blobs of voxels. The system then applies furniture classifiers on each blob and classifier responses are applied to every voxel in the blob. When the system detects a coffee table or a table, it uses the same region growing algorithm to segment isolated blobs on top of the table and evaluates the tabletop object voxel classifiers on them.

We compare the performance of the detection-based scene labeling technique of [18] (Det3DMRF), using HMP over RGB-D images with the scene labeling MRF (HMP2D), using HMP3D with the scene labeling MRF (HMP3D), and the combination of both as proposed in Section II (HMP2D+3D). HMP2D is very similar to Det3DMRF, except that Histogram of Oriented Gradients (HOG) features have been replaced by sparse coding features learned from RGB and depth image data, an idea first proposed in [16].

For HMP2D, we use a one-layer architecture with a dictionary of size $M = 100$ for both grayscale and depth attributes and is learned from 5×5 image patches. For HMP3D, we use $M = 150$ for the first layer dictionary and $M = 1000$ for the second layer. The first-layer patch size is $5 \times 5 \times 5$ and for second-layer it is $4 \times 4 \times 4$ first-layer patches. The three-level spatial pyramid pooling in Fig. 4 is used. SVM training was performed using LIBLINEAR [32] and the outputs are transformed by a sigmoid to yield probabilities for the scene labeling MRF. The MRF parameters were $\alpha = 1/6$, $\epsilon = 1/4$, and $\lambda = 50$.

2) Results: Table I shows the precision and recall of scene labeling for each object category as well as the overall performance, with each category given equal weight. The precision and recall is computed on a per-point basis. For example the recall of bowl is the proportion of 3D points actually belonging to the bowl class successfully labeled by the system as bowl, while its precision is the proportion of 3D points labeled as bowl that actually belong in the bowl class.

We found that RGB-D image detectors did not perform well on furniture. While there are many images of such objects on the Internet, RGB-D data from many different viewpoints is scarce. Also, furniture is categorized by function rather than form, so there can be a lot of intraclass variation in shape and appearance. Finally, furniture pieces are large objects which are often only partially visible due to occlusion or from being cut off by the camera view cone. This makes detecting them using RGB-D image detectors difficult. On the other hand, HMP3D can handle objects of any size since it operates on the 3D scene reconstruction built from aggregating many video frames. Det3DMRF and HMP2D both achieve similar accuracy. As was the case in [16], using sparse coding features learned from image data is usually slightly better than using HOG features. The new RGB-D Scenes Dataset v.2 is more challenging than the original RGB-D Scenes Dataset for RGB-D image detectors because each video is only one rotation around the table, while in the RGB-D Scenes Dataset objects are often seen from similar viewpoints multiple times. The shorter videos make it harder to prune out false positives, as objects are not detected repeatedly as often. HMP3D performs well on most objects, but sometimes confuses bowls as being coffee mugs. Often HMP2D and HMP3D do not make mistakes on the same objects, and hence the combination of the two yields the best scene labeling accuracy.

Fig. 5 shows scene labeling results from six of the fourteen scenes in RGB-D Scenes Dataset v.2. The scene labeling in Fig. 1 is also an actual result produced by the system. The figures show that the proposed approach is able to correctly identify most tabletop and furniture objects and

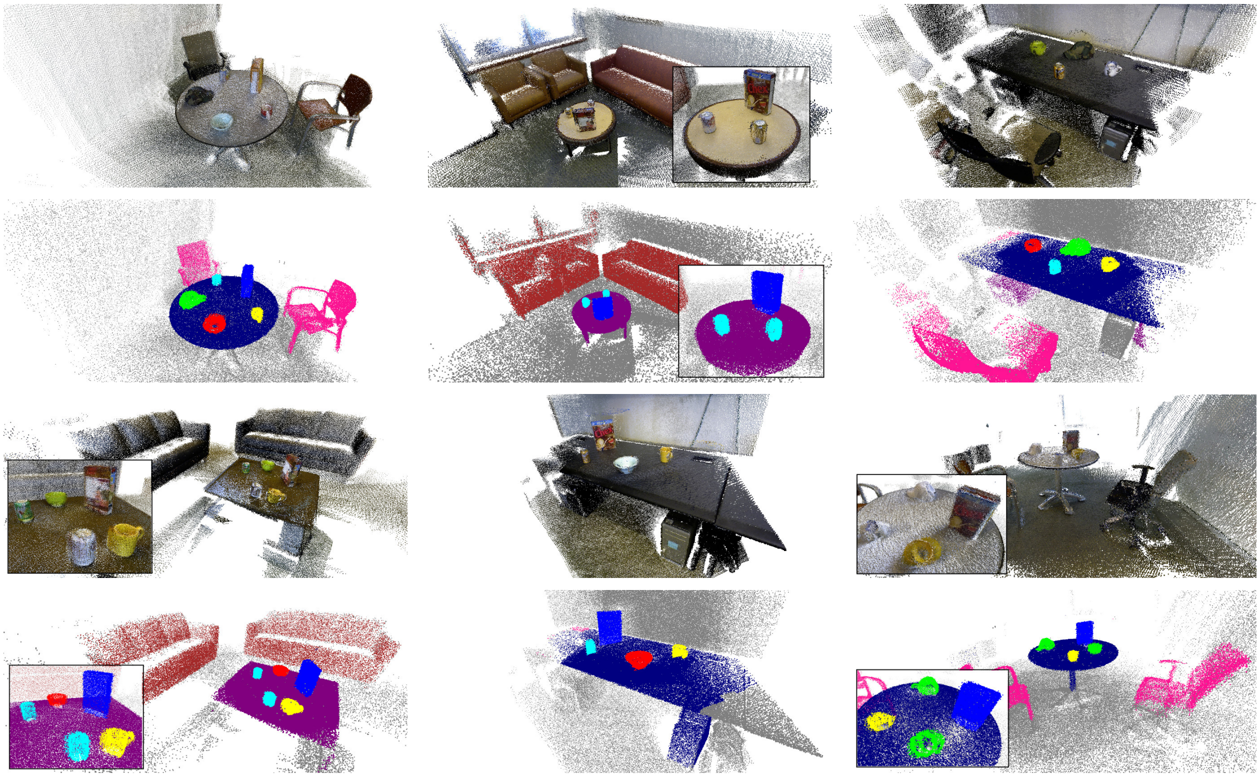

Fig. 5. 3D scene labeling results for six of the fourteen scenes in RGB-D Scenes Dataset v.2. 3D reconstruction (top) and our results using a combination of HMP on RGB-D images and HMP over voxel data (bottom). Objects are colored by their category: bowl (red), cap (green), cereal box (blue), coffee mug (yellow), soda can (cyan), chair (pink), coffee table (purple), sofa (brown), table (navy blue). Best viewed in color.

object boundaries are very clean. Notice that the system is able to correctly distinguish between coffee tables and tables, which are two object categories with similar shape and appearance, but different height. Since our approach estimates the true size of objects from training data, the voxel classifiers use features computed from different sized regions for each class. The region used for the coffee table classifier is shorter, and thus only the legs of a table would be visible. On the other hand, a coffee table would only occupy the lower portion of the region used by a table classifier. Hence, the features after 3D spatial pooling look very different for coffee table and table. This ability is unique to HMP over voxel data and cannot be achieved by previous feature learning methods applied to RGB-D images, which perform spatial pooling over the 2D grid structure of images as opposed to pooling in 3D space.

There were a few failure cases on this dataset. For example, the system confused two computer workstations placed side by side underneath an office desk as being a coffee table (top right of Fig. 5). A more comprehensive training dataset may have prevented this. When the region-growing segmentation algorithm fails, the system will also fail to segment out the exact extents of large furniture objects, such as the sofa in the coffee room (top middle of Fig. 5). Finally, when there are errors in the 3D scene reconstruction, the system may also fail to correctly classify an object (e.g. bowl in the bottom right of Fig. 5).

We also evaluated HMP2D on the original RGB-D Scenes Dataset containing 8 video sequences in which tabletop objects from the same five categories are annotated. We found that the system performs similarly to the detection-based object labeling approach of [18]. The overall precision and recall of HMP2D was 89.8% and 89.1% respectively, which is within 2% of the results reported in [18] (91.0% precision, 88.8% recall). It appears that in the case of the RGB-D Scenes Dataset, HOG features already work very well and there is no improvement from using HMP2D features instead.

3) Running Time: The running time of the proposed approach is dominated by the time it takes to run sliding window object detectors on the video frames. Our current single-threaded MATLAB implementation requires four seconds per image to extract features and run sliding window object detectors. In future work, we plan to run sliding window detectors in parallel and also take advantage of GPU computation, which on a multi-core machine should bring the running time down to one second. Since the scene does not change significantly in one second, running sliding window detectors at this rate should yield sufficient accuracy for real-time scene labeling. Voxel classification is run once per video sequence, and the speed of HMP3D feature extraction depends on the voxel resolution. In our experiments we use a voxel resolution of around 80 voxels along the longest dimension and HMP3D feature extraction took around ten seconds per object on a single-threaded MATLAB implementation. We extract HMP3D features from all objects in the scene in parallel. Once HMP3D features have been extracted, voxel classification took less than one second for the entire scene. Finally, MRF inference takes less than 500 ms per scene for the environments in the two RGB-D Scenes Datasets.

B. 3D Object Recognition on the Princeton Shape Benchmark

We also evaluated the HMP3D feature representation independent of the overall scene labeling approach. One large and commonly used database for 3D object recognition is the Princeton Shape Benchmark (PSB) [33]. The PSB contains $1,814$ 3D polygonal models of objects collected from the World Wide Web. The models are split into training and testing databases of 907 models each. We evaluate the usefulness of HMP3D in distinguishing between the 92 base categories in the PSB testing database. Following the procedure of [34], we learn HMP3D features using all 907 models in the training database, and then use the learned dictionaries to compute features for the 907 models in the testing database. We perform leave-one-out linear SVM classification and report overall accuracy averaged over 907 runs, each run using a different object model as test data and the other 906 as training data.

We first convert each 3D polygonal model into a 3D point cloud by randomly sampling points from each polygonal surface with constant density. The resulting point cloud is rescaled and voxelized so that the longest edge has unit length and a resolution of 80 voxels. A two-layer feature hierarchy is learned over surface normals as the voxel attribute. The first-layer dictionary is learned over $5 \times 5 \times 5$ voxel patches and contains 144 codewords. The second-layer dictionary is learned over $4 \times 4 \times 4$ pooled first-layer patches and contains 2000 codewords. Three-level spatial pyramid max pooling is used with 1, 8, and 27 cells in each level respectively.

We compare the performance of HMP3D with the spherical harmonic descriptor (SHD) [35] and Harmonic Shape Histograms (HSH) [34]. Table II shows the classification accuracy on the Princeton Shape Benchmark testing database of the various approaches. Each feature is paired with either a nearest neighbor (NN) classifier or a support vector machine (SVM) classifier. The results for SHD and HSH were reported in [34]. The results show that HMP3D is very powerful feature for 3D data, outperforming the alternative 3D shape descriptors when paired with a linear SVM classifier. Since HMP3D is a very high dimensional feature ($403,776$ dimensions in this experiment), it is not surprising that it perform less well when paired with a nearest neighbor classifier. Nevertheless, HMP3D still outperforms SHD, HSH and their combination when these features are also paired with a nearest neighbor classifier. Only the combination of both HSH and SHD features with SVM classification outperforms using HMP3D alone.

Feature	Accuracy (NN)	Accuracy (SVM)
SHD	55.6	55.4
HSH	59.8	68.4
HSH+SHD	63.5	72.9
HMP	65.4	71.0

TABLE II
COMPARISON OF HMP3D WITH ALTERNATIVE SHAPE DESCRIPTORS ON BASE CATEGORY CLASSIFICATION ON THE PRINCETON SHAPE BENCHMARK TESTING DATABASE. EACH FEATURE IS PAIRED WITH EITHER A NEAREST NEIGHBOR (NN) CLASSIFIER OR A SUPPORT VECTOR MACHINE (SVM) CLASSIFIER.

Feature	Category		
	RGB only	Depth only	RGB-D
HMP2D [13]	82.4 ± 3.1	81.2 ± 2.3	87.5 ± 2.9
HMP3D	77.8 ± 2.5	76.5 ± 2.2	82.1 ± 3.5

TABLE III
ACCURACY OF HMP OVER RGB-D IMAGES (HMP2D) AND HMP OVER 3D VOXELS (HMP3D). IN BOTH CASES, FEATURES ARE LEARNED OVER GRAYSCALE INTENSITIES AND RGB VALUES FOR RGB-ONLY, DEPTH VALUES AND SURFACE NORMAL VECTORS FOR DEPTH-ONLY, AND ALL FOUR ATTRIBUTES FOR RGB-D.

C. RGB-D Object Recognition

HMP3D is designed for 3D point clouds of objects generated from multiple views, while many RGB-D object recognition datasets are captured using a Kinect from a single view[10], [36]. In such data, the point cloud may not capture the entire object because it is constructed from only one image. HMP3D is intended for learning feature representations on point clouds created from RGB-D videos, where there is coverage of objects from multiple viewpoints so that the object model is relatively complete. Nevertheless, we wanted to evaluate HMP3D on an existing, extensive, object recognition dataset to see how HMP3D will perform single-view RGB-D images, even though that is not its intended use case. For this, we conducted a category recognition experiment on the RGB-D Object Dataset [10]. We use the evaluation set and the same ten train/test splits from [10].

We compared HMP over voxel data (HMP3D) and HMP over RGB-D images (HMP2D). Both approaches use the same pixel/voxel attributes: grayscale and RGB value from RGB images, and depth and surface normals from depth images. Two-layer dictionaries were learned on each attribute independently. HMP2D used a patch size of 5×5 pixels and HMP3D used a patch size of $5 \times 5 \times 5$ voxels. Both features used a first-layer dictionary size of 150, second-layer dictionary size of 1000, and three-level spatial pyramid pooling. The difference is that HMP2D sparse codes are pooled over cells on an image plane, while HMP3D pools sparse codes over cells in 3D space.

The results in Table III show that while HMP2D and HMP3D both achieve good performance on the RGB-D Object Dataset, HMP2D is still better. This is due to several factors. First, the RGB-D Object Dataset contains many dark, shiny, and thin objects (e.g. stapler, battery, cell phone) with many missing depth values. While HMP2D can still extract features over RGB images in such cases, HMP3D must operate on a point cloud that does not fully capture the object. Even when using RGB and grayscale attributes, they are only available to HMP3D for pixels that have valid depth values. Secondly, the depth image contains large depth discontinuities between the object the background, and HMP2D is able to capture the object's silhouette, which is a very informative cue. On the other hand, HMP3D works only on the points on the object itself, so it does not capture information about the shape of the object against the background. HMP3D has performed well on this dataset considering that it was designed for 3D point clouds created from RGB-D videos rather than single RGB-D images where missing depth values is a big problem.

VI. Discussion

This paper presented an approach for labeling 3D scenes reconstructed from RGB-D videos. We introduced HMP3D, a hierarchical sparse coding technique for learning features from 3D point cloud data. HMP3D classifiers are trained using a synthetic dataset of virtual scenes generated using CAD models from an online database. Our scene labeling system combines features learned from RGB-D images and 3D point clouds to assign a object class label to every 3D point in the scene. Experiments on the RGB-D Scenes Dataset v.2 demonstrate that the proposed approach can be used to label indoor scenes containing both small table-top objects and large furniture pieces. We also showed that HMP3D achieves excellent object retrieval performance on the Princeton Shape Benchmark. Finally, HMP3D also achieves good category recognition results on the RGB-D Object Dataset even though it is designed for 3D point clouds from RGB-D videos rather than single-view RGB-D images.

References

[1] R. Triebel, R. Schmidt, O. Martinez Mozos, and W. Burgard, "Instance-based amn classification for improved object recognition in 2d and 3d laser range data," in *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2007.

[2] M. Quigley, S. Batra, S. Gould, E. Klingbeil, Q. Le, A. Wellman, and A. Ng, "High-accuracy 3d sensing for mobile manipulation: Improving object detection and door opening," in *Proc. of ICRA*, 2009.

[3] G. Hager and B. Wegbreit, "Scene parsing using a prior world model," *IJRR*, 2011, to appear.

[4] J. Glover, G. Bradski, and R. Rusu, "Monte Carlo pose estimation with quaternion kernels and the bingham distribution," in *Proc. of RSS*, 2011.

[5] A. Collet, M. Martinez, and S. Srinivasa, "Object recognition and full pose registration from a single image for robotic manipulation," *IJRR*, vol. 30, no. 10, 2011.

[6] K. Lai and D. Fox, "Object Recognition in 3D Point Clouds Using Web Data and Domain Adaptation," *The International Journal of Robotics Research*, 2010.

[7] H. Hu, D. Munoz, J. A. D. Bagnell, and M. Hebert, "Efficient 3-d scene analysis from streaming data," in *ICRA*, May 2013.

[8] X. Xiong, D. Munoz, J. Bagnell, and M. Hebert, "3-D scene analysis via sequenced predictions over points and regions," in *Proc. of ICRA*, 2011.

[9] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena, "Contextually guided semantic labeling and search for three-dimensional point clouds," *IJRR*, vol. 32, no. 1, pp. 19–34, 2013.

[10] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *ICRA*, 2011.

[11] ——, "Sparse distance learning for object recognition combining rgb and depth information," in *ICRA*, 2011.

[12] M. Blum, J. T. Springenberg, J. Wulfing, and M. Riedmiller, "A learned feature descriptor for object recognition in rgb-d data," in *ICRA*, 2012, pp. 1298–1303.

[13] L. Bo, X. Ren, and D. Fox, "Unsupervised Feature Learning for RGB-D Based Object Recognition," in *ISER*, 2012.

[14] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1106–1114.

[15] L. Bo, X. Ren, and D. Fox, "Multipath sparse coding using hierarchical matching pursuit," in *CVPR*, June 2013.

[16] X. Ren and D. Ramanan, "Histograms of sparse codes for object detection," in *CVPR*, June 2013.

[17] R. Socher, C. C. Lin, A. Ng, and C. Manning, "Parsing natural scenes and natural language with recursive neural networks," in *ICML*, 2011, pp. 129–136.

[18] K. Lai, L. Bo, X. Ren, and D. Fox, "Detection-based object labeling in 3d scenes," in *ICRA*, 2012.

[19] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, 2011, pp. 127–136.

[20] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D Mapping: Using depth cameras for dense 3D modeling of indoor environments," in *the 12th International Symposium on Experimental Robotics (ISER)*, December 2010.

[21] P. Henry, D. Fox, A. Bhowmik, and R. Mongia, "Patch Volumes: Segmentation-based Consistent Mapping with RGB-D Cameras," in *International Conference on 3D Vision (3DV)*, 2013.

[22] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE PAMI*, vol. 31, no. 5, pp. 824–840, 2009.

[23] L. Bo, X. Ren, and D. Fox, "Hierarchical Matching Pursuit for Image Classification: Architecture and Fast Algorithms," in *NIPS*, 2011.

[24] A. Anand, H. Koppula, T. Joachims, and A. Saxena, "Semantic labeling of 3D point clouds for indoor scenes," in *NIPS*, 2011.

[25] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE PAMI*, vol. 23, no. 11, pp. 1222–1239, 2001.

[26] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear Spatial Pyramid Matching using Sparse Coding for Image Classification," in *CVPR*, 2009.

[27] H. Lee, R. Grosse, R. Ranganath, and A. Ng, "Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations," in *ICML*, 2009.

[28] Y. Sun, L. Bo, and D. Fox, "Attribute Based Object Identification," in *ICRA*, 2013.

[29] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[30] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[31] "Trimble 3d warehouse," http://sketchup.google.com/3dwarehouse/.

[32] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin, "Liblinear: A library for large linear classification," *Journal of Machine Learning Research (JMLR)*, vol. 9, pp. 1871–1874, 2008. [Online]. Available: http://www.jmlr.org/papers/volume9/fan08a/fan08a.pdf

[33] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser, "The princeton shape benchmark," in *Shape Modeling International*, 2004.

[34] J. Fehr and H. Burkhardt, "Harmonic shape histograms for 3d shape classification and retrieval." in *IAPR Conference on Machine Vision Applications*, 2007, pp. 384–387.

[35] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3d shape descriptors," in *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*. Eurographics Association, 2003, pp. 156–164.

[36] B. Browatzki, J. Fischer, B. Graf, H. Bulthoff, and C. Wallraven, "Going into depth: Evaluating 2d and 3d cues for object classification on a new, large-scale object dataset," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1189–1195.