Detection-based Object Labeling in 3D Scenes

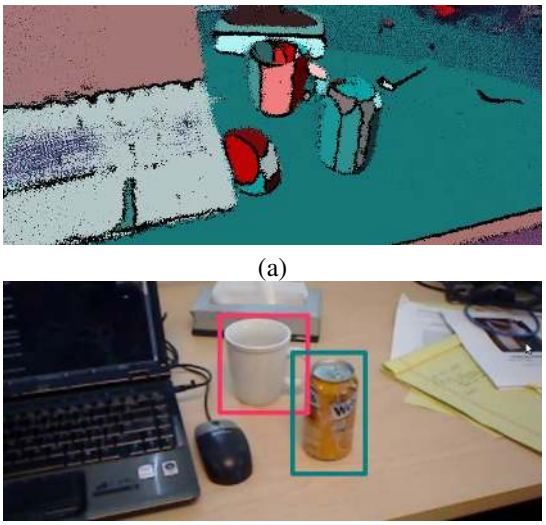
Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox

***Abstract*— We propose a view-based approach for labeling objects in 3D scenes reconstructed from RGB-D (color+depth) videos. We utilize sliding window detectors trained from object views to assign class probabilities to pixels in every RGB-D frame. These probabilities are projected into the reconstructed 3D scene and integrated using a voxel representation. We perform efficient inference on a Markov Random Field over the voxels, combining cues from view-based detection and 3D shape, to label the scene. Our detection-based approach produces accurate scene labeling on the RGB-D Scenes Dataset and improves the robustness of object detection.**

I. Introduction

3D scene understanding is a fundamental problem in robotics and perception, as knowledge of the environment is a prerequisite for complex robotic tasks and interactions. Thus far, most robotic scene understanding work has focused on outdoor scenes captured with laser scanners, where technologies have matured and enabled high-impact applications such as mapping and autonomous driving [2], [25], [21], [7], [30]. Indoor scenes, in comparison, prove more challenging and cover a wider range of objects, scene layouts, and scales. A variety of approaches have been proposed for labeling 3D indoor scenes and detecting objects in them [29], [14], [12], [4], [19], [26], typically on a single camera view or laser scan. Purely shape-based labeling is often very successful on large architectural elements (e.g. wall, chair) but less so on small objects (e.g. coffee mug, bowl). Robust labeling of objects requires the use of rich visual information in addition to shape. While image-only object matching can work very well for textured objects [4], recent works on RGB-D perception combine both shape and color and show promising results for generic object detection [26], [19], [20].

Advances in RGB-D matching and scene reconstruction make it feasible to go beyond single-view labeling and allow the continuous capture of a scene, where RGB-D *videos* can be robustly and efficiently merged into consistent and detailed 3D point clouds [18]. One way to label a multi-frame scene would be to directly work with the merged point cloud. Point cloud labeling has worked very well for outdoor scenes [21], [7], [30], and the recent work of [1] illustrates how labeling can be done on indoor 3D scenes, where local features are computed from segments of point clouds and integrated in a Markov Random Field (MRF) model. Such approaches focus on 3D point clouds and do not fully utilize the RGB-D frames, nor do they make use of detection techniques developed for images [11].



(a)

(b)

Fig. 1. An illustration of our detection-based approach versus a 3D segmentation approach. (a) A typical approach to 3D labeling segments the scene into components, inevitably causing some objects to be oversegmented. (b) We train object detectors and use them to localize entire objects in individual frames, taking advantage of rich visual information.

In this work, we emphasize an object-centric view of 3D labeling, combining techniques in object detection, scene labeling, and 3D reconstruction. Instead of learning from entire scenes, we train object detectors on view-based data, i.e. views of objects possibly isolated and/or from a different domain. We run object detectors on the RGB-D frames to score individual pixels and then project these into the reconstructed 3D scene. A voxel representation is used in an MRF model to integrate these projections to produce accurate labeling and segmentation of detected objects.

The resulting approach works on non-textured objects and does not require complete 3D object models or finding supporting surfaces in the scene. Combining views from multiple frames, we greatly improve the accuracy of object detection on the RGB-D Scene Dataset [19], where our detection-based labeling achieves almost 90% F-score. We also experimentally demonstrate another advantage of our approach: the ability to train object detectors using large repositories on the Web (e.g. [6], [13]).

II. Related Work

Scene labeling has been extensively studied under many different settings. Techniques have been developed for label-

This work was funded in part by an Intel grant, by ONR MURI grant N00014-07-1-0749 and by the NSF under contract IIS-0812671.

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

Xiaofeng Ren is with Intel Science and Technology Center on Pervasive Computing, Seattle, WA 98195, USA. xiaofeng.ren@intel.com

ing surfaces such as grass, walls, or pavement, and small sets of object types such as foliage, people, and cars in 3D *outdoor scenes*. Most of these approaches label individual 3D laser points using features describing local shape and appearance in combination with spatial and temporal smoothing via graphical model inference [2], [25], [21], [7], [30]. Features are either extracted from a fixed neighborhood around 3D points [2], [30] or from small patches generated via an over-segmentation of the scene [25], [21]. Instead of labeling individual points, Douillard and colleagues [8] first generate a voxel representation and then classify voxels based on cumulative shape features within. Unlike these approaches that only analyze the reconstructed scene, we take advantage of rich visual information by first extracting features from and detecting objects in the constituent RGB-D images.

Object detection in 3D *indoor scenes* is challenging due to variation in object appearance and size. Triebel *et al.* [29] built on associative Markov networks introduced in [2] to detect large objects such as chairs, tables, and fans in 3D laser scans. They label individual points based on local shape features extracted from one point cloud. Lai *et al.* [19] detect objects with sliding window object detectors using shape and color features extracted from a single RGB-D frame collected by a Kinect style camera. Unlike this previous work, the proposed approach here analyzes an entire video sequence covering objects from multiple viewpoints and performs spatial reasoning in 3D. Anand *et al.* [1] segment a point cloud generated from an aligned set of RGB-D frames and performs classification based on local features accumulated over segments. An alternative approach is to match existing 3D object models into a scene [9], [12], [8], [14]. However, these techniques require either annotated 3D scenes for training, or the availability of 3D object models. Many also focus on shape information, thereby ignoring valuable appearance information provided by RGB cameras. In contrast, Collet *et al.* [4] used only visual features to build sparse 3D models of textured objects and efficiently match them. Our technique combines both appearance and shape information without requiring annotated scenes or complete 3D object models for training.

In computer vision, both scene labeling and object detection have been extensively studied. [15] is an example of scene labeling in images using a multi-scale CRF model to integrate local and contextual cues. Image labeling has focused mostly on large scene elements (such as sky, ground, building) and less on small everyday objects. For object detection, HOG (histogram of oriented gradients) templates based sliding window classifiers have been shown to be robust and is widely used [5], [11]. However, most existing object detection systems only output bounding box detections, as in Helmer *et al.* [17] who used sliding window classifiers to detect objects from several viewpoints. We instead analyze RGB-D videos and provide dense labeling of every point in the 3D scene.

Algorithm 1: Detection-based Scene Labeling

1. **Input:** Aligned RGB-D frames $f_1, ..., f_K$, trained object detectors $1, ..., C$.
2. **For each** k **in** $\{1, ..., K\}$
3. Project all N pixels in f_k into 3D points $x_1, ..., x_N$ using the camera pose estimate of f_k.
4. **For each** c **in** $\{1, ..., C\}$
5. Run sliding window detector c on frame f_k to obtain $p(c|x), \forall x \in \{x_1, .., x_N\}$ // Eq.3.
6. **End**
7. Compute background probabilities $p(c_B|x)$ // Eq.5
8. **End**
9. Create voxel map $\mathcal{V}$, each voxel v containing 3D points Ω_v.
10. **For each** voxel v **in** $\mathcal{V}$
11. Compute $\psi_v(y_v)$ using $p(y_v|x), \forall x \in \Omega_v$. // Eq.7
12. **End**
13. **For each** pair of neighboring voxels $\{i, j\}$
14. Compute $\phi_{i,j}(y_i, y_j)$ using surface normals. // Eq.10
15. **End**
16. Minimize $E(y_1, ..., y_{|\mathcal{V}|})$ using multi-class graph cuts [3] and return voxel labels $y_1, ..., y_{|\mathcal{V}|}$. // Eq.1

III. 3D Object Labeling

We consider the problem of object labeling in RGB-D scenes, i.e. 3D scenes that are captured from a set of RGB-D video frames. We utilize RGB-D mapping [18] to globally aligned and merge each frame with the scene under a rigid transform. The goal is to label objects of interest in such scenes, where these objects may comprise only a small part of the 3D point cloud.

We represent a 3D scene as a set of voxels $\mathcal{V}$. Each voxel v is associated with a label $y_v \in \{1, ..., C, c_B\}$, where $1, ..., C$ are object classes and c_B is the background class. We model the joint distribution of voxel labels using an MRF 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. 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.

MRF-based techniques have been used for many labeling tasks, providing a unified framework for combining local evidence, such as appearance and shape, with dependencies across regions like label smoothness. An approach that has shown some promise is to define the data term using local features on over-segmentations (e.g. [21], [30], [1]). These segmentation-based approaches deal with imperfect segmentation by generating a "soup of segments" [23] by running multiple segmentation algorithms on the same scene and aggregating classification results, hoping that at least some of segments will be distinctive enough to be recognized.

We instead train view-based object detectors that run on individual RGB-D frames and incorporate the responses

into the data term. Sliding window based approaches learn templates from object views that have been annotated with bounding boxes encompassing the entire object. An object is localized in an image by scanning over multiple positions and scales.

We illustrate the differences pictorially in Fig. 1. A typical 3D scene labeling approach first aggregates the data into the reconstructed scene and then extracts features and performs classification on the scene. Our view-based approach instead evaluates classifiers on the individual frames before merging the evidence into the 3D scene using an MRF. This allows us to extract rich visual and depth features and leverage state-of-the-art detection techniques, including max-margin detector training, searching over image scale pyramids, and mining hard negative examples. We now describe our approach, summarized in Algorithm 1, in detail.

A. Detection-based Probability Maps

We compute the data term $\psi_v(y_v)$ in Eq.1 using responses from object detectors that have been independently trained using a database of objects and background images. The detectors are run on individual RGB-D frames and their responses are used to estimate an object class probability for every pixel in every frame.

Scoring 3D points using object detection. View-based object detection has been extensively studied [5], [10], [16]. Sliding window detectors assign scores to a grid of locations in the image on which the detector window has been centered, often across multiple image scales. The standard object detection framework uses this score map pyramid to select a subset of high scoring detector bounding boxes to label as object candidates.

For scene labeling, we instead want to assign a label to every 3D point in the scene. For scenes reconstructed from Kinect videos, each point is a pixel in some RGB-D frame and the system can remember this one-to-one mapping. Instead of using the score map pyramid to select bounding box candidates, we use it to compute a score map over all pixels (and hence 3D points).

Throughout this paper we use x to denote a 3D point. When computing features for x, the system looks up the corresponding pixel and computes them using the source RGB-D frame. Let $f_c^h(x)$ be the feature vector of x for object detector c at scale h. We use features extracted from both the RGB and depth images (see Section III-C). An object can appear in different sizes depending on its distance from the camera, so it is necessary to run sliding window detectors on multiple scaled versions of the image. While RGB-D data makes it plausible to select the "proper" scale based on the physical sizes of objects, we choose not to enforce any hard scale constraint and instead use scale as an input feature to the detector, as in [19]. Hence, the maximum detector response across all scales is the best estimate of the true object scale at a particular image position and we obtain the score map of a linear SVM object detector c:

$$s_c(x) = \max_h \{w_c^\top f_c^h(x) + b_c\} \tag{2}$$

where w_c and b_c is the weight vector and bias of the object detector c, learned from training data. We train one detector for each object class, so $1 \leq c \leq C$. We convert these linear score maps into probability maps that define the probability of point x belonging to class c, using Platt scaling [24] (Algorithm 1, line 5):

$$p(c|x) = \frac{1}{1 + \exp\{us_c(x) + v\}} \tag{3}$$

where u and v are the parameters of the sigmoid and can be found by minimizing negative log likelihood of the training or validation set. This is not a probability over all classes, but instead a probability obtained from the binary classifier between class c and background [27]. The advantage of training C binary classifiers instead of one multi-class classifier is that we can train each detector with a different template window size and aspect ratio, using a different training dataset.

Background probability. A point belongs to the background class if it does not lie on any of the objects. Detectors provide evidence for the background class when they do not fire at a location. The object detectors are trained and evaluated independently, each seeing other objects as negative examples. To obtain a background probability, we observe that when there are C foreground classes, the following holds for the probability of point x being background:

$$p(c_B|x) \leq 1 - p(c|x),\ \forall c \in \{1, ..., C\} \tag{4}$$

Hence, we can use the smallest probability of the negative classes as the upper bound for that of the background class. In practice, we use the discounted value

$$p(c_B|x) = \alpha \min_{1 \leq c \leq C} \{1 - p(c|x)\} \tag{5}$$

where $0 < \alpha \leq 1$ controls the looseness of the upper bound (Algorithm 1, line 7).

Integrating evidence from multiple frames. The RGB-D mapping algorithm [18] performs frame alignment and estimates the camera pose of each frame in the global 3D scene. We voxelize the resulting point cloud so that each voxel v contains a set of 3D points Ω_v.

We generate the likelihood of voxel v by multiplying the likelihoods of its constituent points Ω_v and normalizing the resulting likelihood using the geometric mean

$$p(y_v|\Omega_v) = \left\{ \prod_{x \in \Omega_v} p(y_v|x) \right\}^{\frac{1}{|\Omega_v|}} \tag{6}$$

where $y_v \in \{1, ..., C, c_B\}$ and $p(y_v|x)$ is looked-up from the corresponding probability map. Here we cannot simply take the product because the class probabilities of each 3D point are not independent; without discounting, the resulting probabilities will be overconfident. Hence, we discount the probabilities by normalizing them by the number of points in the voxel [28]. As is standard in multi-class graph cuts, the data term in the MRF is the negative log likelihood:

$$\psi_v(y_v) = -\ln p(y_v|\Omega_v) = -\frac{1}{|\Omega_v|} \sum_{x \in \Omega_v} \ln p(y_v|x) \tag{7}$$

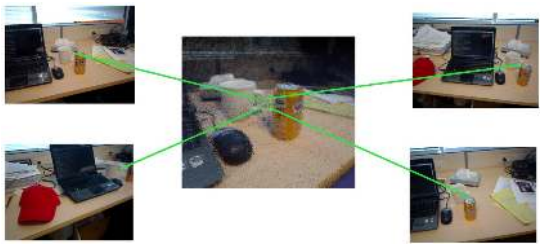


Fig. 2. Each voxel in the scene (center) contains 3D points projected from multiple RGB-D frames. The points and their projections are known, so we can compute average likelihoods for each voxel based on the likelihoods of constituent points. Combining detections from multiple frames greatly improves the robustness of object detection.

This corresponds to setting the log likelihood of a voxel v to be the arithmetic mean of the log likelihoods of its constituent points Ω_v (Algorithm 1, line 11).

B. Label Consistency and Geometric Information

The pairwise term $\phi_{i,j}(y_i, y_j)$ in Eq.1 encodes interactions between nearby voxels. The simplest pairwise interaction is the Potts model [3]

$$\lambda \cdot \mathbf{1}_{y_i \neq y_j} \tag{8}$$

where $\mathbf{1}_{y_i \neq y_j}$ evaluates to 1 when $y_i \neq y_j$ and 0 otherwise. This adds a constant penalty λ when adjacent elements do not share the same label. This is based on the assumption that the world is "smooth" in the sense that nearby elements tend to share the same label. However, in reality abrupt label changes do occur at object boundaries.

In image labeling, researchers have used the contrast-dependent smoothness prior [22],

$$\lambda \cdot \mathbf{1}_{y_i \neq y_j} \exp(-\theta \|x_i - x_j\|^2). \tag{9}$$

Here x_i and x_j are the intensities/colors of pixels i and j, so label changes between dissimilar pixels are penalized less, with θ controlling the sensitivity. This captures the intuition that label changes tend to occur at sharp edges, where adjacent pixels have very different intensities/colors. However, intensity/color changes in an image do not necessarily correspond to object boundaries, since objects can often have internal edges (e.g. logos on a soda can). Furthermore, the boundary between two similarly colored objects may not have strong image gradients.

For labeling RGB-D scenes, 3D shape information can be used as a more reliable cue for object boundaries. We incorporate this information into our model by defining

$$\phi_{i,j}(y_i, y_j) = \lambda \cdot \frac{\mathbf{1}_{y_i \neq y_j}}{d(n_i, n_j)} \left(\mathbf{I}(n_i, n_j) + \epsilon\right) \tag{10}$$

where λ and ϵ are balancing parameters (Algorithm 1, line 14).

As in the Potts model [3], the pairwise term is non-zero only when $y_i \neq y_j$. $d(n_i, n_j)$ measures the difference between surface normals n_i and n_j of, respectively, voxels i and j; if $d(\cdot)$ is small, the normals are similar, i and j are on a smooth surface and the cost is high to assign them different labels. We use the L_2-distance plus a small constant as our distance metric between surface normals. $\mathbf{I}(n_i, n_j)$ is an indicator variable expressing whether i and j are part of a convex surface (such as the top of a cereal box) or a concave one (where a cereal box touches its supporting surface). Since objects tend to have convex shapes and concave surface transitions are more likely to occur at object boundaries, the cost is lower to cut the scene at a concave shape than a convex one, with the parameter ϵ controlling the trade-off.

This pairwise term captures the intuition that object boundaries tend to have large changes in surface orientation, and that objects tend to be supported by flat surfaces, leading to concave surface transitions. As was done in [1], we use

$$\mathbf{I}(n_i, n_j) = [(n_i - n_j) \cdot (i - j) > 0] \tag{11}$$

to indicate whether the surface transition between voxels i and j is convex. To compute this we need to ensure that all surface normals point outward and not into the object. This can be done because the camera pose of the video frame from which each point originates is known, and the surface normal should form a sharp ($> 90°$) angle with the camera view vector.

The data term $\psi_v(v)$ and the pairwise term $\phi_{i,j}(y_i, y_j)$ together define a multi-class pairwise MRF, whose energy is quickly minimized using graph cuts [3] (Algorithm 1, line 16). There are three free parameters in our model (α,λ,ϵ) and they are easy to set by hand. We leave learning the parameters to future work.

C. RGB-D Object Detection

The HOG based sliding window classifier is among the most successful object detection techniques [5], [10]. We use its extension to RGB-D data, first proposed in [19], as the object detector for generating our probability maps. We now describe this particular detector in detail. Note that the proposed MRF-based scene labeling approach only uses the probability maps. If desired, this detector can be substituted with, for example, the deformable parts-based model of Felzenszwalb *et al.* [11], without changes to the rest of the framework.

Following Lai *et al.* [19], we extract HOG features over both the RGB and depth image to capture appearance and shape information of each view of an object. We use a variant of the feature that has been shown to slightly outperform the original HOG descriptors [10]. The gradient orientations in each cell (8×8 pixel grid) are encoded using two different quantization levels into 18 ($0° - 360°$) and 9 orientation bins ($0° - 180°$). The resulting $4 \times (18 + 9) = 108$-dimensional feature vector is analytically projected into a 31-dimensional feature vector. The first 27 dimensions correspond to different orientation channels (18 contrast sensitive and 9 contrast insensitive). The last 4 dimensions capture the overall gradient energy in four blocks of 2×2 cells. As in [19], we use a recursive median filter to fill in missing values in the depth image and then compute HOG features in the same way. This feature is sensitive to the large depth discontinuities that occur at object boundaries, enabling it to capture object silhouette information. Finally, we also

compute a normalized depth histogram. We normalize the depth image using the size of the bounding box and then compute histograms over an 8×8 grid. We used a histogram of 20 bins with each bin having a range of 0.30m. This is based on the observation that an object that is d times as far away will appear $1/d$ times as large in the image. Hence, this feature is sensitive to whether the detector is centered on a set of points that are representative of the true size of the object.

The performance of classifiers heavily depends on its abilities to exploit negative data. In a typical image or depth map captured by Kinect, there are many (10^5) potential negative examples, which makes it impractical to use all negative examples. We use a bootstrapping procedure to mine the hard negative examples. The initial negative examples consists of cropped rectangles from background videos and objects from other categories. After training a classifier, the resulting classifier is used to search background images and select the false positives with the highest scores (hard negatives). These hard negatives are added as negative examples and the classifier is retrained. This procedure is repeated 10 times to obtain the final classifier.

When using RGB-D object detectors, we found it helpful to perform a depth-based refinement to improve the probability map defined in Eq.10. In Fig. 3, we show the probability map of a cereal box sliding window detector. In this example the detector correctly finds the location of the cereal box; however, strong signals are only localized around the center of the object. The probabilities quickly decrease near the edges of the cereal box. This is not surprising since bounding boxes centered near the edge contain a large amount of background and look very different from the cereal box itself. To better capture the spatial extents of detected objects, we refine the probability map using high scoring bounding boxes found by our object detectors. For each bounding box above a detector score threshold of -0.6, we set

$$p(c|x) = p(c|x_0) \exp(-\gamma \|x - x_0\|^2), x \in \mathcal{B} \qquad (12)$$

where $\mathcal{B}$ is the set of 3D points in a bounding box, x_0 is the center of the corresponding bounding box, and the parameter γ controls how quickly the probability decreases with increasing depth difference. This expansion is analogous to bilateral filtering using depth. From the example in Fig. 3, we can see that many more pixels inside the cereal box now have a strong signal. We note that this procedure can falsely introduce some strong signals on non-object pixels (the table in this case) into the refined probability map. While a more expensive approach using more cues may yield better refinement, we found that our simple and quick approach is sufficient, as the false signals can be cleaned up by the MRF.

IV. Experiments

We evaluate the proposed detection-based 3D scene labeling on the RGB-D Object Dataset [19], which contains everyday objects captured individually on a turntable, as well as kitchen and office scenes containing multiple objects captured with a freely moving Kinect style camera (RGB-D Scenes Dataset). We train our object detectors using individually captured objects and evaluate scene labeling on point clouds reconstructed from the RGB-D scene videos. We show how to use 3D scene labeling results to improve the performance of bounding box based object detection. Finally, we show examples of 3D labeling using only web image data for training.

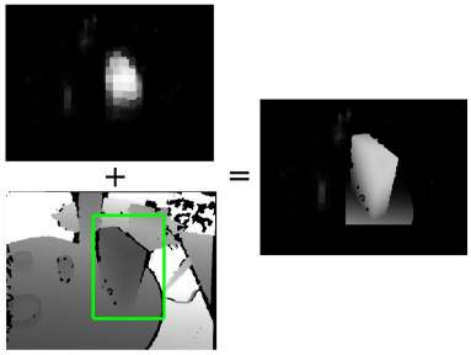


Fig. 3. An example of using RGB-D object detectors to obtain a probability map defined on all pixels. We first run the cereal box detector to obtain a class probability at each pixel (top left). We then use high scoring bounding box candidates (bottom left) to perform depth-based refinement and obtain the final probability map that better fits the shape of the object (right).

A. Experimental Setup

The RGB-D Object Dataset [19] contains 250,000 segmented RGB-D images of 300 objects in 51 categories. It also includes the RGB-D Scenes Dataset, which consists of eight video sequences of office and kitchen environments. The videos are recorded with a Kinect-style sensor from Primesense and each contain from 1700 to 3000 RGB-D frames. We evaluate our system's ability to detect and label objects in the five categories bowl, cap, cereal box, coffee mug, and soda can, and distinguish them from everything else, which we label as background. Each scene used in our evaluation contains things ranging from walls, doors, and tables, to a variety of objects placed on multiple supporting surfaces. Considering only the five categories listed above, each scene contains between 2 to 11 objects in up to all five categories. There are a total of 19 unique objects appearing in the eight scenes.

We train a linear SVM sliding window detector for each of the five object categories using views of objects from the RGB-D Object Dataset. Each view is an RGB-D image of the object as it is rotated on a turntable. This provides dense coverage of the object as it is rotated about its vertical axis. The data contains views from three different camera angles at 30, 45, and 60 degrees with the horizon. Each detector is trained using all but one of the object instances in the dataset, so 5 of the 19 objects that appear in the evaluation were never seen during training. The aspect ratio of the detector's bounding box is set to be the tightest box that can be resized to contain all training images. In total, the positive training set of each category detector consists of around 500 RGB-D images. We use the 11 background videos in the RGB-D

Technique	Micro F-score	Macro F-score
Random	16.7	7.4
AllBG	94.9	15.8
DetOnly	72.5	71.3
PottsMRF	87.7	87.4
ColMRF	88.6	88.5
Det3DMRF	89.9	89.8

Fig. 4. Micro and Macro F-scores for two naïve algorithms (random and labeling everything as background) and the proposed detection-based 3D scene labeling approach and its variants.

Scenes Dataset to mine for hard negative examples.

We use a voxel representation with 1cm×1cm×1cm voxels, where the neighborhood of each voxel is the 26 directly adjacent voxels. We experimented with larger neighborhoods but found no appreciable difference in performance. The surface normal of each voxel is computed using voxels within a $\sqrt{27}$cm radius. Energy minimization via multi-class graph cuts [3] yields the optimal labels for all voxels, which is used to label the constituent points to obtain the final scene labeling. In our experiments we use the parameter settings $\alpha = 0.1$, $\lambda = 100$, $\epsilon = 1/49$, which we found to work well for all video sequences.

B. 3D Scene Labeling Results

The primary task of the proposed approach is to perform 3D scene labeling, which is to assign a label to every point in a 3D scene. We evaluate the proposed approach described in Section III (*Det3DMRF*), as well as variants involving (a) only the data term in our model (*DetOnly*); (b) incorporating the standard Potts smoothness term, Eq.8 (*PottsMRF*); (c) incorporating the contrast-dependent smoothness term, Eq.9, over voxels that take on the mean color of its constituent points (*ColMRF*).

Fig. 4 compares the micro and macro F-scores of variants of the proposed approach, as well as the naïve approaches of randomly labeling each point (*Random*) and labeling everything as background (*AllBG*). The Micro F-score is computed from the overall precision and recall of the scene labeling, while Macro F-score is the average of the F-scores of the five categories, each computed separately. Random is obviously terrible regardless of the performance metric used. Since most points are background, AllBG actually performs well in terms of micro F-score. However, it fails to detect any of the objects and hence performs poorly when each category is given equal weight in the macro F-score.

Fig. 5 shows the per-category and overall precisions and recalls of variants of the proposed approach. The overall precision/recall is a macro-average across categories; a micro-average would not be informative with the majority of points being background. The results show that PottsMRF significantly improves precision while also yielding modest gains in recall. Det3DMRF boosts precision and recall further; table points that are mislabeled as objects because they often lie inside high scoring detector bounding boxes can be removed because there is often a sharp concave normal transition between the table and the object. ColMRF can also robustly segment objects if the color is significantly different, but overall does not perform as well as Det3DMRF on the RGB-D Scenes Dataset. Although the gains from Det3DMRF seem modest when looking at the numbers, qualitatively segmentation is improved substantially as points on the table are now almost always excluded from the objects (see Fig. 6).

In Fig. 7 we show three complex scenes that were labeled by our 3D scene labeling technique (Det3DMRF). The top row shows the reconstructed 3D scene and the bottom row shows results obtained by Det3DMRF. Objects are colored by their category label, where bowl is red, cap is green, cereal box is blue, coffee mug is yellow, and soda can is cyan.

Running time. The RGB-D Mapping algorithm [18] used for scene reconstruction runs in real-time for our Kinect videos, which were collected at 15-20 Hz. We evaluate object detectors on every 10th frame, or every 0.5 seconds. Our current single-threaded MATLAB implementation is not yet real-time, requiring 4 seconds to process each frame. Voxelization and graph cut inference take negligible time. The overwhelming majority of computation is spent on feature extraction and sliding window classification, each taking around 1.8 seconds. Given that both of these procedures are highly parallelizable, we believe that a more optimized, multi-threaded implementation will run in real-time. When we run detection on every 80th frame, i.e. in real-time for our current implementation, the system only suffers slight performance degradation: precision= 83.6%, recall=79.5%, micro f-score=81.5%, macro f-score=81.0%. Due to the speed at which the camera is moved, there can be large viewpoint changes between 80 frames in the RGB-D Scenes Dataset. Slowing down camera movement would narrow the performance gap.

C. Refining Image-based Object Detection

After labeling a 3D scene, it is possible to use it to validate bounding box proposals from the constituent video frames. We do this by running the object detectors with a low threshold and pruning out bounding box candidates whose labels do not agree with the majority label of points in a central region of the bounding box. Fig. 8 shows precision-recall curves obtained from both the individual frame-by-frame object detections (red) and detections validated by our 3D scene labeling (blue). Each point along the curve is generated by ranking detections from all five category detectors together and thresholding on the detection score. It is clear that 3D scene labeling can significantly reduce false positives by aggregating evidence across the entire video sequence. While the precision of the frame-by-frame detection approach rapidly decreases beyond 60% recall for all eight scenes, using 3D scene labeling it is possible to obtain 80% recall and 80% precision in a majority of them.

D. 3D Scene Labeling using ImageNet

Up until now we have focused on evaluating object detectors trained with images and depth maps captured with an RGB-D camera. Though we can collect such training

Technique	Precision/Recall						
	Bowl	Cap	Cereal Box	Coffee Mug	Soda Can	Background	Overall
DetOnly	46.9/90.7	54.1/90.5	76.1/90.7	42.7/74.1	51.6/87.4	98.8/93.9	61.7/87.9
PottsMRF	84.4/90.7	74.8/91.8	88.8/94.1	87.2/73.4	87.6/81.9	99.0/98.3	86.9/88.4
ColMRF	93.7/86.9	81.3/92.2	91.2/89.0	88.3/73.6	83.5/86.5	98.7/98.8	89.4/87.8
Det3DMRF	91.5/85.1	90.5/91.4	93.6/94.9	90.0/75.1	81.5/87.4	99.0/99.1	91.0/88.8

Fig. 5. Per-category and overall (macro-averaged across categories) precisions and recalls for the proposed detection-based 3D scene labeling approach and its variants. Our approach works very well for all object categories in the RGB-D Scene Dataset.

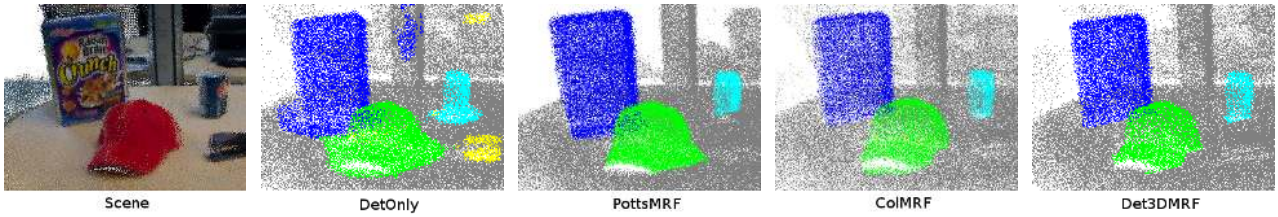


Fig. 6. Close-up of a scene containing a cap (green), a cereal box (blue), and a soda can (cyan). From left to right, the 3D scene; Detection-only leaves patches of false positives; Potts MRF removes isolated patches but cannot cleanly segment objects from the table; Color MRF includes part of the table with the cap because it is similar in color due to shadows; the proposed detection-based scene labeling obtains clean segmentations. Best viewed in color.

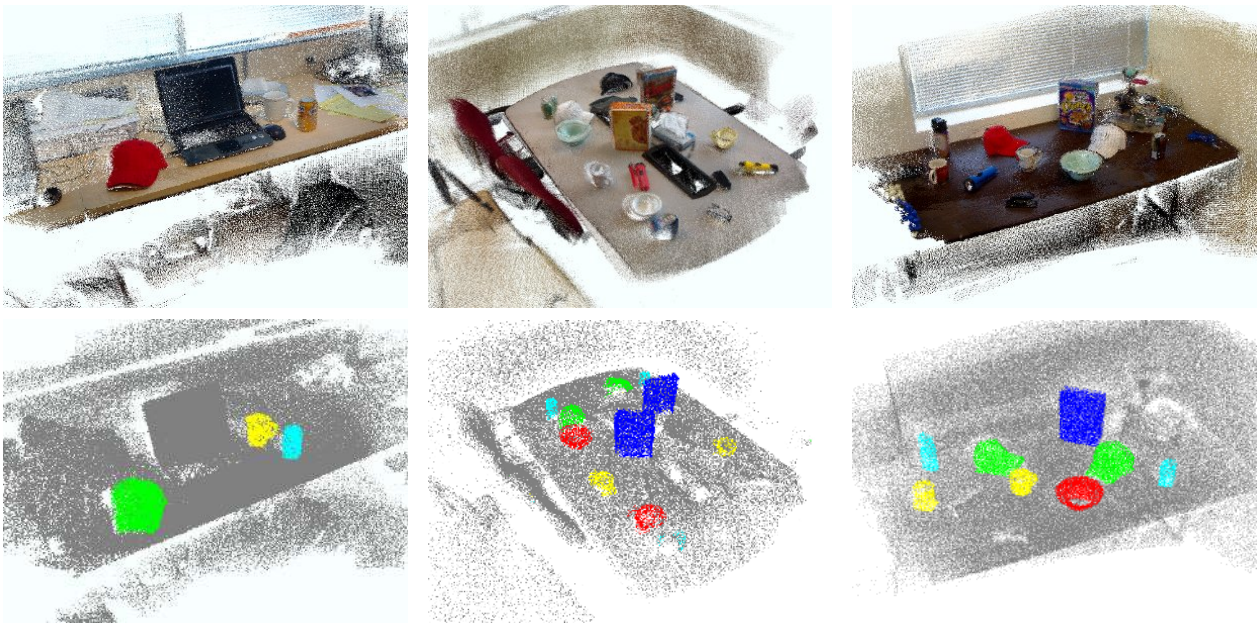


Fig. 7. 3D scene labeling results for three complex scenes in the RGB-D Scenes Dataset. 3D reconstruction (top), our detection-based scene labeling (bottom). Objects colored by their category: bowl is red, cap is green, cereal box is blue, coffee mug is yellow, and soda can is cyan. Best viewed in color.

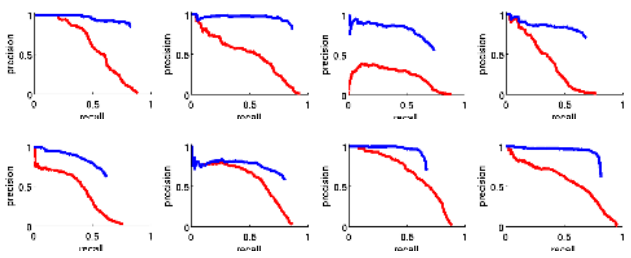


Fig. 8. Precision-recall curves comparing the performance of labeling images with bounding boxes of detected objects. Each plot shows results on one of the eight video sequences in the RGB-D Scenes Dataset, aggregated over all five category detectors. Frame-by-frame object detection is drawn in red, while 3D scene labeling (our approach) is drawn in blue.

data using the approach proposed in [19], it is still time-consuming to get enough data to train good object detectors. Given that large labeled image collections are available online now [6], in this experiment we try to train object detectors using an existing image dataset (here we use ImageNet) and use the resulting detector responses in our system to perform scene labeling. This can be done in exactly the same framework, except that now the probability map is obtained from detectors that use only HOG features extracted from the RGB image. Since only the detectors need to be changed, we can still use the depth map to refine the probability map and perform 3D scene reconstruction.

We retrieve the coffee mug category from ImageNet and

obtain 2200 images containing coffee mugs. We generate around 200 positive examples by cropping the coffee mug windows from images where ground truth bounding boxes were provided and resizing them to a 104×96 window. The negative examples are exactly the same background dataset used to train object detectors in our previous experiments. We train the coffee mug detector using linear SVMs with the same hard negative mining procedure. In our evaluation, all coffee mugs in the eight scenes from the RGB-D Scenes Dataset were correctly detected by this detector. We show two example scenes in Fig. 9.

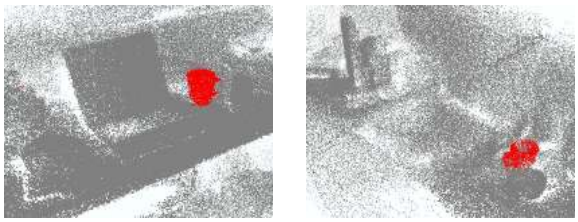


Fig. 9. Labeling coffee mugs in 3D scenes using a sliding window detector trained with ImageNet data using only visual HOG features.

V. Discussion

RGB-D mapping techniques have made it feasible to capture and reconstruct 3D indoor scenes from RGB-D videos. In this work we pursue an object-centric view of scene labeling, utilizing view-based object detectors to find objects in frames, project detections into 3D and integrate them in a voxel-based MRF. We show that our detection-based approach, reasoning about whole objects on the source RGB-D frames (instead of 3D point clouds), leads to very accurate labelings of objects and can be used to makes object detection in still images much more robust.

Labeling objects in indoor scenes is very challenging, partly because there exists a huge range of objects, scene types, and scales. One needs to utilize the full resolution of sensor data and detailed object knowledge. We see a promising direction forward by focusing on objects and their learning. There exists a large amount of object data, such as in ImageNet [6] and Google 3D Warehouse [13]. We have shown an example where we reliably label objects in 3D scenes using only ImageNet for training. There are many such opportunities where object knowledge may be acquired and applied to scene understanding.

References

[1] A. Anand, H.S. Koppula, T. Joachims, and A. Saxena. Semantic labeling of 3D point clouds for indoor scenes. In *Advances in Neural Information Processing Systems (NIPS)*, 2011.

[2] D. Anguelov, B. Taskar, V. Chatalbashev, D. Koller, D Gupta, G. Heitz, and A. Ng. Discriminative learning of Markov random fields for segmentation of 3D scan data. In *Proc. of CVPR*, 2005.

[3] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE PAMI*, 23(11):1222–1239, 2001.

[4] A. Collet, M. Martinez, and S. Srinivasa. Object recognition and full pose registration from a single image for robotic manipulation. *IJRR*, 30(10), 2011.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of CVPR*, 2005.

[6] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proc. of CVPR*, 2009.

[7] B. Douillard, D. Fox, F. Ramos, and H. Durrant-Whyte. Classification and semantic mapping of urban environments. In *IJRR*, volume 30, 2011.

[8] B. Douillard, J. Underwood, V. Vlaskine, A. Quadros, and S. Singh. A pipeline for the segmentation and classification of 3D point clouds. In *Proc. of the International Symposium on Experimental Robotics (ISER)*, 2010.

[9] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3D object recognition. In *Proc. of CVPR*, 2010.

[10] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proc. of CVPR*, 2008.

[11] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:1627–1645, 2010.

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

[13] Google. 3d warehouse. http://sketchup.google.com/3dwarehouse/, 2008.

[14] G.D. Hager and B. Wegbreit. Scene parsing using a prior world model. *IJRR*, 2011. To appear.

[15] X. He, R.S. Zemel, and M.A. Carreira-Perpinán. Multiscale conditional random fields for image labeling. In *Proc. of CVPR*, volume 2, pages 695–702, 2004.

[16] Scott Helmer and David G. Lowe. Using stereo for object recognition. In *Proc. of ICRA*, pages 3121–3127, 2010.

[17] Scott Helmer, David Meger, Marius Muja, James J. Little, and David G. Lowe. Multiple viewpoint recognition and localization. In *ACCV (1)*, pages 464–477, 2010.

[18] 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.

[19] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *Proc. of ICRA*, 2011.

[20] K. Lai, L. Bo, X. Ren, and D. Fox. A scalable tree-based approach for joint object and pose recognition. In *Twenty-Fifth Conference on Artificial Intelligence (AAAI)*, August 2011.

[21] K. Lai and D. Fox. Object recognition in 3D point clouds using web data and domain adaptation. *IJRR*, 29(8), 2010.

[22] B. Liu, S. Gould, and D. Koller. Single image depth estimation from predicted semantic labels. In *Proc. of CVPR*, 2010.

[23] T. Malisiewicz and A. Efros. Improving spatial support for objects via multiple segmentations. In *Proc. of the British Machine Vision Conference*, 2007.

[24] J. Platt. Probabilistic outputs for support vectormachines and comparisons to regularized likelihood methods. In *Advances in large margin classifiers*. Cambridge: MIT Press, 1999.

[25] I. Posner, M. Cummins, and P. Newman. A generative framework for fast urban labeling using spatial and temporal context. *Autonomous Robots*, 26(2-3), 2009.

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

[27] R. Salakhutdinov, A. Torralba, and J. Tenenbaum. Learning to share visual appearance for multiclass object detection. In *Proc. of CVPR*, 2011.

[28] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, September 2005. ISBN 0-262-20162-3.

[29] 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.

[30] 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.