IJCV manuscript No.
(will be inserted by the editor)

Indoor Scene Understanding with RGB-D Images

Bottom-up Segmentation, Object Detection and Semantic Segmentation

Saurabh Gupta · Pablo Arbeláez · Ross Girshick · Jitendra Malik

Received: date / Accepted: date

Abstract In this paper, we address the problems of contour detection, bottom-up grouping, object detection and semantic segmentation on RGB-D data. We focus on the challenging setting of cluttered indoor scenes, and evaluate our approach on the recently introduced NYU-Depth V2 (NYUD2) dataset [45].

We propose algorithms for object boundary detection and hierarchical segmentation that generalize the $gPb - ucm$ approach of [3] by making effective use of depth information. We show that our system can label each contour with its type (depth, normal or albedo). We also propose a generic method for long-range amodal completion of surfaces and show its effectiveness in grouping.

We train RGB-D object detectors by analyzing and computing Histogram of Oriented Gradients (HOG) on the depth image and using them with deformable part models (DPM) [16]. We observe that this simple strategy for training object detectors significantly outperforms more complicated models in the literature.

We then turn to the problem of semantic segmentation for which we propose an approach that classifies superpixels into the dominant object categories in the NYUD2 dataset. We design generic and class-specific features to encode the appearance and geometry of objects. We also show that additional features computed from RGB-D object detectors and scene classifiers further improves semantic segmentation accuracy. In all of these tasks, we report significant improvements over the state-of-the-art.

Keywords RGB-D contour detection · RGB-D image segmentation · RGB-D object detection · RGB-D semantic segmentation · RGB-D scene classification

This work was sponsored by ONR SMARTS MURI N00014-09-1-1051, ONR MURI N00014-10-10933, and a Berkeley Fellowship.

S. Gupta[1], P. Arbeláez[1,2], R. Girshick[1], J. Malik[1]
[1] University of California at Berkeley, Berkeley, CA
[2] Universidad de los Andes, Colombia
E-mail: {sgupta, arbelaez, rbg, malik}@eecs.berkeley.edu

1 Introduction

The problem of scene and image understanding from monocular images has been studied very well in recent years [18,21,22,34,35,43]. Some works have addressed the task of inferring coarse 3D layout of outdoor scenes, exploiting appearance and geometric information [22,43]. Recently, the focus has shifted towards the more difficult case of cluttered indoor scenes [19,21,34,35]. In this context, the notion of affordance and the functionality of objects for human use acquires importance. Thus, [21] recovers walk-able surfaces by reasoning on the location and shape of furniture, [34, 35] reason about the 3D geometry of the room and objects, while [19] focuses on interpreting the scene in a human-centric perspective.

Another major line of work has been object detection. Most notable among them is the work in the sliding window paradigm one of the first example being Viola and Jones [47], which considered the task of face detection, Dalal and Triggs [11], which proposed and benchmarked various feature choices for use with sliding window detectors, and more recent works [5,16] which extends the sliding window approaches to reason about parts and their relative arrangements. Notably, Felzenszwalb *et al.*'s deformable part models (DPM) [16], is the widely accepted state-of-the-art method for object detection.

(a) Color (b) Normal (c) Contours (d) Amodal Comp (e) Contour Type (f) Obj Det (g) Semantic Segm

Fig. 1: **Output of our system**: We take in as input a single color and depth image ((a) and (b)) and produce as output a bottom-up segmentation (c), long range completions (d), contour classification (e) (into depth discontinuities (red), concave normal discontinuities (green) and convex normal discontinuities (blue)), object detections (f), and a semantic segmentation (g).

With the recent introduction of a commodity depth sensor (like the Microsoft Kinect), a new area of research has opened up in computer vision of looking at tasks which have traditionally been very hard. For example, recent works have considered 3D reconstruction tasks such as real-time scene reconstruction [24], and recovering high fidelity albedo, shape and illumination [4].

There has also been a lot of work on semantic understanding of images given RGB-D input from a depth sensor. A particularly striking first work among this is that of real-time human pose estimation from single RGB-D images [44], in which they demonstrate that with the availability of RGB-D input they can solve the hard problem of human joint localization well enough to be used in a practical application. Subsequently, there have been numerous papers in both robotics and vision communities looking at various image and scene understanding problems namely, bottom-up segmentation [12,38,45], semantic segmentation [7,29,39,45], and object detection [25,28,32,46,48].

In this paper we tackle all these 3 tasks - bottom-up segmentation, object detection and semantic segmentation for indoor RGB-D images. The output of our approach is shown in Figure 1: given a single RGB-D image ((a) and (b)), our system produces contour detection, bottom-up segmentation ((c)), contour classification ((d)), grouping by amodal completion ((e)), object detection ((f)) and semantic labeling of objects and scene surfaces ((g)).

This is an extended version of the work that appeared in [20]. It differs from [20], in that we also investigate the problem of RGB-D detection, and show that incorporating additional features from object detector activations further improves the semantic segmentation accuracy.

This paper is organized as follows: we review related work in Section 2. We describe our algorithm and results for perceptual re-organization (bottom-up segmentation and amodal completion) in Section 3. We then describe how we train RGB-D object detectors and compare them with existing methods in the literature in Section 4. We then describe our system for semantic segmentation in Section 5. Finally, we use the output from our object detectors and scene classifiers for the task of semantic segmentation, and show how this additional knowledge can help us improve the performance of our semantic segmentation system in Section 6.

2 Related Work

2.1 Bottom-Up and Semantic Segmentation

One of the first attempts at bottom-up and semantic segmentation is that of Silberman *et al.* [45], in which they consider the task of bottom-up RGB-D segmentation and semantic scene labeling, by modifying the algorithm of [23] to use depth for bottom-up segmentation and then using context features derived from inferring support relationships in the scene for performing semantic segmentation. Ren *et al.*'s work [39] uses features based on kernel descriptors on superpixels and their ancestors from a region hierarchy, followed by a Markov Random Field (MRF) context model. Koppula *et al.* [29] also study the problem of indoor scene parsing with RGB-D data in the context of mobile robotics, where multiple views of the scene are acquired with a Kinect sensor and subsequently merged into a full 3D reconstruction. The full 3D point cloud is oversegmented and used as underlying structure for an MRF model. A rich set of features is defined, describing local appearance, shape and geometry, and contextual relationships among object classes. A max-margin formulation is proposed to learn the model parameters and inference is performed via LP relaxation.

Our work differs from the references above in both our approach to segmentation and to recognition. We visit the segmentation problem afresh by extending the gPb-ucm [3] machinery to leverage depth information, giving us significantly better bottom-up segmentation

when compared to earlier works. We also consider the interesting problem of amodal completion [27] and obtain long range groups, which gives us better bottom-up region proposals for scene surfaces which are often interrupted by objects in front of them. Finally, we are also able to label each edge as being a depth edge, a normal edge, or neither.

Our approach for recognition builds on insights from the performance of different methods on the PASCAL VOC segmentation challenge [14]. We observe that approaches like [2,8,9], which focus on classifying bottom-up region candidates using strong features on the region have obtained significantly better results than MRF-based methods [30]. Based on this motivation, we propose new features to represent bottom-up region proposals (which in our case are non-overlapping superpixels and their amodal completion), and use additive kernel SVM classifiers.

2.2 Object Detection

For object detection, from a robotics perspective, Lai *et al.* collect a dataset of day-to-day objects, and propose novel kernel descriptor features to recognize these objects [31,32]. We study the same problem, but consider it in uncontrolled and cluttered real world scenes, and develop techniques which can generalize across instances of the same category. Moreover, we are more interested in the problem of detecting large furniture like items. Johnson *et al.*, Rusu *et al.* and Frome *et al.* look at computing features for describing points in point cloud data [17,26,40], but in this work we want to design features for complete objects. [25] also consider the task of object detection in RGB-D settings, and propose modifications to the approach of [16], and re-scoring and pruning detections to improve detection accuracy. In more recent work [28], propose modifications to DPMs to reason in 3D and take into account bottom-up grouping cues, and show improvements over the approach of [25]. [46] also look at the task of object detection and work in the same framework, but do not reason about perspective in their calculations for depth image gradients. [48] also look at the same task but compute features on the surface normal images. Our work is more similar to that of [46,48], but we differ in the features that we use, and observe that even a simple model with the right features can outperform more complicated approaches.

3 Perceptual Organization

One of our main goals is to perform perceptual organization on RGB-D images. We would like an algorithm that detects contours and produces a hierarchy of bottom-up segmentations from which we can extract superpixels at any granularity. We would also like a generic machinery that can be trained to detect object boundaries, but that can also be used to detect different types of geometric contours by leveraging the depth information. In order to design such a depth-aware perceptual organization system, we build on the architecture of the $gPb - ucm$ algorithm [3], which is a widely used software for monocular image segmentation.

3.1 Geometric Contour Cues

In addition to color data, we have, at each image pixel, an estimation of its $3D$ location in the scene from which we can infer its surface normal orientation. We use this local geometric information to compute three oriented contour signals at each pixel in the image: a depth gradient DG which identifies the presence of a discontinuity in depth, a convex normal gradient NG_+ which captures if the surface bends-out at a given point in a given direction, and a concave normal gradient NG_-, capturing if the surface bends-in.

Generalizing the color and texture gradients of gPb to RGB-D images in not a trivial task because of the characteristics of the data, particularly: (1) a nonlinear noise model of the form $|\delta Z| \propto Z^2|\delta d|$, where δZ is the error in depth observation, Z is the actual depth, δd is the error in disparity observation (due to the triangulation-based nature of the Kinect), causing non-stochastic and systematic quantization of the depth, (2) lack of temporal synchronization between color and depth channels, resulting in misalignment in the dataset being used, (3) missing depth observations. We address these issues by carefully designing geometric contour cues that have a clear physical interpretation, using multiple sizes for the window of analysis, not interpolating for missing depth information, estimating normals by least square fits to disparity instead of points in the point cloud, and independently smoothing the orientation channels with Savitsky-Golay [42] parabolic fitting.

In order to estimate the local geometric contour cues, we consider a disk centered at each image location. We split the disk into two halves at a pre-defined orientation and compare the information in the two disk-halves, as suggested originally in [37] for contour detection in monocular images. In the experiments, we consider 4 different disk radii varying from 5 to 20 pixels and 8 orientations. We compute the 3 local geomet-

ric gradients DG, NG_+ and NG_- by examining the point cloud in the 2 oriented half-disks. We first represent the distribution of points on each half-disk with a planar model. Then, for DG we calculate the distance between the two planes at the disk center and for NG_+ and NG_- we calculate the angle between the normals of the planes.

3.2 Contour Detection and Segmentation

We formulate contour detection as a binary pixel classification problem where the goal is to separate contour from non-contour pixels, an approach commonly adopted in the literature [3,23,37]. We learn classifiers for each orientation channel independently and combine their final outputs, rather than training one single classifier for all contours.

Contour Locations We first consider the average of all local contour cues in each orientation and form a combined gradient by taking the maximum response across orientations. We then compute the watershed transform of the combined gradient and declare all pixels on the watershed lines as possible contour locations. Since the combined gradient is constructed with contours from all the cues, the watershed oversegmentation guarantees full recall for the contour locations. We then separate all the boundary location candidates by orientation.

Labels We transfer the labels from ground-truth manual annotations to the candidate locations for each orientation channel independently. We first identify the ground-truth contours in a given orientation, and then declare as positives the candidate contour pixels in the same orientation within a distance tolerance. The remaining boundary location candidates in the same orientation are declared negatives.

Features For each orientation, we consider as features our geometric cues DG, NG_+ and NG_- at 4 scales, and the monocular cues from gPb: BG, CG and TG at their 3 default scales. We also consider three additional cues: the depth of the pixel, a spectral gradient [3] obtained by globalizing the combined local gradient via spectral graph partitioning, and the length of the oriented contour.

Oriented Contour Detectors We use as classifiers support vector machines (SVMs) with additive kernels [36], which allow learning nonlinear decision boundaries with an efficiency close to linear SVMs, and use their probabilistic output as the strength of our oriented contour detectors.

Hierarchical Segmentation Finally, we use the generic machinery of [3] to construct a hierarchy of segmentations, by merging regions of the initial oversegmentation based on the average strength of our oriented contour detectors.

3.3 Amodal Completion

The hierarchical segmentation obtained thus far only groups regions which are continuous in 2D image space. However, surfaces which are continuous in $3D$ space can be fragmented into smaller pieces because of occlusion. Common examples are floors, table tops and counter tops, which often get fragmented into small superpixels because of objects resting on them.

In monocular images, the only low-level signal that can be used to do this long-range grouping is color and texture continuity which is often unreliable in the presence of spatially varying illumination. However, in our case with access to $3D$ data, we can use the more robust and invariant geometrical continuity to do long-range grouping. We operationalize this idea as follows:

1. Estimate low dimensional parametric geometric models for individual superpixels obtained from the hierarchical segmentation.
2. Greedily merge superpixels into bigger more complete regions based on the agreement among the parametric geometric fits, and re-estimate the geometric model.

In the context of indoor scenes we use planes as our low dimensional geometric primitive. As a measure of the agreement we use the (1) orientation (angle between normals to planar approximation to the 2 superpixels) and (2) residual error (symmetrized average distance between points on one superpixel from the plane defined by the other superpixel); and use a linear function of these 2 features to determine which superpixels to merge.

As an output of this greedy merging, we get a set of non-overlapping regions which consists of both long and short range completions of the base superpixels.

3.4 Results

We train and test our oriented contour detectors using the instance level boundary annotations of the NYUD2 as the ground-truth labels. We follow the standard train-test splits of NYUD2 dataset with 795 training images and 654 testing images (these splits make sure that images from the same scene are either entirely in the test set or entirely in the train set).

We evaluate performance using the standard benchmarks of the Berkeley Segmentation Dataset [3]: precision and recall on boundaries and Ground Truth Covering of regions. We consider two natural baselines for bottom-up segmentation: the algorithm $gPb-ucm$, which does not have access to depth information, and the approach of [45], made available by the authors (labeled NYUD2 baseline), which produces a small set (5) of nested segmentations using color and depth.

Figure 2 and Table 1 [1] present the results. Our depth-aware segmentation system produces contours of far higher accuracy than $gPb-ucm$, improving the Average Precision (AP) from 0.55 to 0.70 and the maximal F-measure (ODS in Table 1 - left) from 0.62 to 0.69. In terms of region quality, the improvement is also significant, increasing the best ground truth covering of a single level in the hierarchy (ODS in Table 1 - right) from 0.55 to 0.62, and the quality of the best segments across the hierarchy from 0.69 to 0.75. Thus, on average, for each ground truth object mask in the image, there is one region in the hierarchy that overlaps 75% with it. The comparison against the NYUD2 baseline, which has access to depth information, is also largely favorable for our approach. In all the benchmarks, the performance of the NYUD2 baseline lies between $gPb-ucm$ and our algorithm.

In [45], only the coarsest level of the NYUD2 baseline is used as spatial support to instantiate a probabilistic model for semantic segmentation. However, a drawback of choosing one single level of superpixels in later applications is that it inevitably leads to over- or under-segmentation. Table 2 compares in detail this design choice against our amodal completion approach. A first observation is that our base superpixels are finer than the NYUD2 ones: we obtain a larger number and our ground truth covering is lower (from 0.61 to 0.58), indicating higher over-segmentation in our superpixels. The boundary benchmark confirms this observation, as our F-measure is slightly lower, but with higher Recall and lower Precision.

The last row of Table 2 provides empirical support for our amodal completion strategy: by augmenting our fine superpixels with a small set of amodally completed regions (6 on average), we preserve the boundary Recall of the underlying over-segmentation while improving the quality of the regions significantly, increasing the bestC score from 0.58 to 0.63. The significance of this jump can be judged by comparison with the ODS score of the full hierarchy (Table 1 - right), which is 0.62: no single level in the full hierarchy would produce better regions than our amodally completed superpixels.

Our use of our depth-aware contour cues DG, NG_+, and NG_-, is further justified because it allows us to also infer the type for each boundary, whether it is an depth edge, concave edge, convex edge or an albedo edge. We simply average the strengths across the different scales for each of these channels, and threshold them appropriately to obtain labels for each contour. We show some qualitative examples of the output we get in the last column of Figure 3 (5th column).

[1] ODS refers to optimal dataset scale, OIS refers to optimal image scale, bestC is the average overlap of the best segment in the segmentation hierarchy to each ground truth region. We refer the reader to [3] for more details about these metrics.

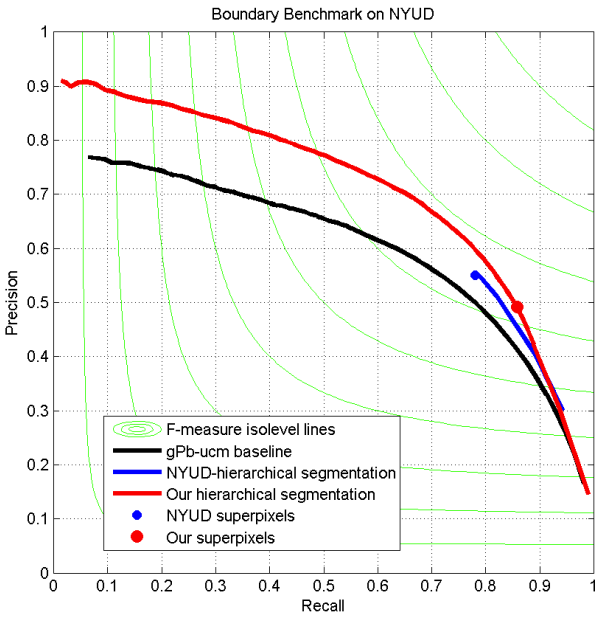


Fig. 2: **Boundary Benchmark on NYUD2**: Our approach (red) significantly outperforms baselines [3](black) and [45](blue).

	Boundary Benchmark			**Region Benchmark**		
	ODS	OIS	AP	ODS	OIS	bestC
gPb-ucm	0.62	0.65	0.55	0.55	0.60	0.69
NYUD2 hierarchy	0.65	0.65	–	0.61	0.61	0.63
Our hierarchy	0.69	0.71	0.70	0.62	0.67	0.75

Table 1: Segmentation benchmarks for hierarchical segmentation on NYUD2. See Footnote 1 for explanation of ODS, OIS, bestC.

	Rec	Prec	F-meas	bestC	Total
NYUD2 superpixels	0.78	0.55	0.65	0.61	87
Our superpixels	0.86	0.51	0.64	0.58	144
Our amodal completion	0.86	0.51	0.64	0.63	150

Table 2: Segmentation benchmarks for superpixels on NYUD2. See Footnote 1 for explanation of bestC.

4 RGB-D Detector

Given access to point cloud data, it is natural to think of a 3D model which scans a 3D volume in space and reasons about parts and deformations in 3D space. While it is appealing to have such a model, we argue that this choice between a 3D scanning volume detector and a 2D scanning window detector only changes the way computation is organized, and that the same 3D reasoning can be done in windows extracted from the 2D image. For example, this reasoning can be in the form of better 3D aware features that can be computed from the points in the support of the 2D sliding window. Not only does this approach deal with the issue of computational complexity, but also readily allows us to extend existing methods in computer vision to RGB-D data.

Hence, we generalize the Deformable Parts Model detector from Felzenszwalb *et al.* [16] to RGB-D images by computing additional features channels on the depth image. We adopt the paradigm of having a multi-scale scanning window detector, computing features from organized spatial cells in the detector support, and learning a model which has deformable parts.

4.1 Features

Note that our sliding window detector searches over scale, so when we are thinking of the features we can assume that the window of analysis has been normalized for scale variations. In addition to the HOG features to capture appearance we use the following features to encode the shape information from the depth image.

4.1.1 Histogram of Depth Gradients

In past work which studied the task of adapting 2D object detectors to RGB-D data [25,46], a popular choice is to simply extend the histogram of oriented gradients (HOG) used on color images to depth images. One would think that this primarily captures depth discontinuities and object boundaries. However as we show in Appendix B, the histogram of depth gradients actually captures the orientation of the surface and not just the depth discontinuities. Very briefly, the *gradient orientation* at a point is along the direction in which the surface is receding away from the viewer (the *tilt*), and the *gradient magnitude* captures the rate at which the surface is receding away (or the *slant* of the surface). Note that when the surface is more or less parallel to the viewing plane, then the estimate for the gradient orientation is inaccurate, and thus the contribution of such points should be down-weighed, and this is precisely what happens when we accumulate the gradient magnitude over different orientations.

The final step in HOG computation involves contrast normalization. We stick with this step, as it makes the feature vector around depth discontinuities (where the surface recedes very sharply) in the same range as the the feature vector around non-depth discontinuity areas.

With this contrast normalization step, it turns out that the histogram of depth gradients is very similar to the histogram of disparity gradients (the gradient orientation is exactly the same, the gradient magnitude are somewhat different, but this difference essentially goes away due to contrast normalization, the complete justification of this is given in Appendix B). In all our experiments we use *HHG*, histogram of oriented horizontal disparity gradients, as this has better error properties than histogram of depth gradients (since a stereo sensor actually measures disparity and not depth).

4.1.2 Histogram of Height

As we show in Appendix A, we can estimate the direction for gravity and estimate the absolute height above the ground plane for each point. We use this estimate of height, to compute a histogram capturing the distribution of heights of the points in each cell. We use the L2 normalized square root of the counts in each bin as features for each cell. We call this feature *HH*.

4.2 Results

In this section, we validate empirically our design choices and compare our results to related work We report experiments on NYUD2 and B3DO.

Performance on NYUD2: The NYUD2 dataset was originally proposed to study bottom-up segmentation, semantic segmentation and support surface inference [45]. However, since it provides dense pixel labels for each object instance, we can easily derive bounding box annotations (by putting a tight bounding box around each instance) and study the task of object detection.

Since, we are interested in investigating the task of detecting furniture like objects in indoor scenes, we select the following five most common (by number of pixels) furniture categories in the dataset - bed, chair, sofa, counter, and table (we exclude cabinets because they are more a part of the scene rather than being a furniture item). For the sake of comparison to past and future work we also include all categories studied by [48], and all categories that are part of the RMRC challenge [1].

We follow the same standard train and test sets (of 795 and 654 images respectively as explained in Sect 3). We found that training with multiple components did not improve performance given the small amount of data.

We follow the standard PASCAL [15] metric of average precision (AP) for measuring detection performance. We report the performance that we obtain in Table 3.

We compare against the state of the art appearance only method [16] and other approaches which make use of depth information [48]. We also compare against the output of our semantic segmentation system as proposed in [20]. We compute bounding box predictions for a class c from the semantic segmentation output by putting a tight bounding box around each connected component of pixels belonging to class c, and assigning each such box a score based on the confidence score for class c of pixels within the box (note that the semantic segmentation output does not have instance information and the tightest bounding box around a connected component often includes multiple instances). We observe that we are able to consistently outperform the baselines. We provide some qualitative visualizations for our bed, chair, sofa, table and counter detections in Figure 3 (6th column).

Performance on B3DO: The B3DO dataset considers the task of detecting mostly small 'prop-like' objects which includes bottle, bowls, cups, keyboards, monitors, computer mouse, phones, pillows and a larger furniture object, chair, and provides 2D bounding box annotations for objects of these categories. For this dataset, we only use the HHG and HOG features and do not use the HH features since the gravity estimate fails because there are a lot of images where the camera is not roughly horizontal (like when over looking the top of a table).

We follow the standard evaluation protocol of training on the 6 train sets and testing on the 6 corresponding validation sets, and reporting the average AP obtained for each category. We report the performance in Table 4. We compare against the approach of [28], who also studied the same task of object detection in RGB-D images.

Although, we designed our model and features with large furniture like objects in mind, we see that our approach works reasonably well, on this task and we get competitive performance even on small 'prop-like' objects. We consistently outperform past approaches which have studied this task in the past.

	DPM [16]	segToDet (CVPR13[20])	Ye *et al.* [48]	Our
bed	27.6	52.1	37.5	56.0
chair	7.8	6.4	15.1	23.5
sofa	9.4	17.5	15.5	34.2
counter	7.3	32.7	16.4	24.0
lamp	22.2	1.4	23.4	26.7
pillow	4.3	3.3	16.9	20.7
sink	5.9	14.0	23.0	22.8
garbage-bin	6.6	-	16.4	26.7
table	5.5	9.3	-	17.2
bathtub	0.9	28.4	-	19.3
television	5.8	3.1	-	19.5
bookshelf	9.0	6.7	-	17.5
toilet	34.4	13.3	-	45.1
box	0.1	0.7	-	0.6
desk	0.7	0.8	-	6.2
door	2.5	5.0	-	9.5
dresser	1.4	13.3	-	16.4
monitor	10.0	-	-	34.9
night-stand	9.2	-	-	32.6
mean over 19 categories	9.0	-	-	23.9
mean over common categories	12.1	18.2	21.1	29.7

Table 3: **Performance on NYUD2 [45]**: We use the standard PASCAL [15] metric of average precision (AP) for measuring detection performance. We compare against an appearance only baseline [16], putting bounding boxes around semantic segmentation produced by our approach in [20], and the publicly reported performance numbers of [48]. Note that the semantic segmentation output from [20] does not give instance labels.

	DPM [16]	Janoch [25]-Prn	Janoch [25]-Rscr	[28]	Our
bottle	10.1	10.4	10.1	10.1	21.9
bowl	37.8	38.8	38.0	45.4	47.8
chair	16.8	21.8	23.0	17.1	39.9
cup	30.9	33.6	35.6	38.3	47.0
keyboard	22.3	24.2	25.0	25.6	25.7
monitor	66.8	64.8	66.7	68.2	64.9
mouse	22.8	25.2	27.6	25.4	48.8
phone	18.0	19.2	19.7	19.8	19.4
mean	28.2	29.7	30.7	31.2	39.4

Table 4: **Performance on B3DO:** Comparison with [25], [28] on B3DO dataset.

Ablation Study: Here we study the impact of each of our features towards the performance of our proposed detector. We do an ablation study by removing each component of our detector. We do this analysis on the train set of the NYUD2 dataset. We split the train set into 2 halves and train on one and report performance on the other. We report the ablation study in Table 5.

We see that all features contribute to the performance. The most important features are HOG on the appearance image and Histogram of Disparity Gradient features.

	full	no hhg	no hh	no hog	# Train	# Val
chair	22.5	19.6	22.6	20.4	547	611
pillow	21.1	8.15	18.9	21.4	266	276
table	14.2	9.06	11.2	11.7	186	211
box	0.498	0.466	0.217	0.225	163	210
sofa	28.4	16.1	19.8	25.6	117	129
door	4.68	6.33	2.72	1.86	136	129
lamp	25.9	16.7	26	25	123	143
counter	14.9	7.28	11.7	14.3	130	104
desk	2.34	1.64	1.78	2.09	59	122
bed	56.6	45.3	51.5	57	94	96
bookshelf	6.3	4.1	5.02	2.69	46	87
sink	36.1	12.1	28.7	30.7	63	47
monitor	27.6	21.8	29.4	8.62	40	37
night-stand	16.5	16.6	18.3	15.6	45	51
garbage-bin	26.6	13	24.7	15.6	51	55
dresser	23.2	3.03	13.1	9.51	40	25
television	23.5	19.2	24.8	9.41	47	33
toilet	48.3	50.3	50.8	48.3	20	17
bathtub	12.2	11.6	7.73	11.8	15	19
mean	21.7	14.9	19.4	17.5	115	126

Table 5: **Ablation Study:** See Section 4.2. We remove the different features from the full detector system and study how the performance degrades.

To gain further understanding of what the detector is learning, we provide visualizations of the model and its various parts in Appendix C.

5 Semantic Segmentation

We now turn to the problem of semantic segmentation on NYUD2. The task proposed in [45] consists of labeling image pixels into just 4 super-ordinate classes - ground, structure, furniture and props. We study a more fine-grained 40 class discrimination task, using the most common classes of NYUD2. These include scene structure categories like walls, floors, ceiling, windows, doors; furniture items like beds, chairs, tables, sofa; and objects like lamps, bags, towels, boxes. The complete list is given in Table 6.

We leverage the reorganization machinery developed in Section 3 and approach the semantic segmentation task by predicting labels for each superpixel. We define features based on the geocentric pose, shape, size and appearance of the superpixel *and* its amodal completion. We then train classifiers using these features to obtain a probability of belonging to each class for each superpixel. We experiment with random decision tree forests [6,10] (RF), and additive kernel [36] support vector machines (SVM).

5.1 Features

As noted above, we define features for each superpixel based on the properties of both the superpixel and its amodal completion. As we describe below, our features capture affordances via absolute sizes and heights which are more meaningful when calculated for the amodal completion rather than just over the superpixel. Note that we describe the features below in context of superpixels but we actually calculate them for both the superpixel and its amodal completion.

5.1.1 Generic Features

Geocentric Pose: These features capture the pose - orientation and height, of the superpixel relative to the gravity direction. These features include (1) *orientation features*: we leverage our estimate of the gravity direction from Section A, and use as features, the angle with respect to gravity, absolute orientation in space, fraction of superpixel that is vertical, fraction of superpixel that is horizontal, and (2) *height above the ground*: we use height above the lowest point in the image as a surrogate for the height from the supporting ground plane and use as features the minimum and maximum height above ground, mean and median height of the horizontal part of the superpixel.

Size Features: These features capture the spatial extent of the superpixel. This includes the size of the 3D bounding rectangle, the surface area - total area, vertical area, horizontal area facing up, horizontal area facing down, if the superpixel is clipped by the image and what fraction of the convex hull is occluded.

Shape Features: These include - planarity of the superpixel (estimated by the error in the plane fitting), average strength of local geometric gradients inside the region, on the boundary of the region and outside the region, average orientation of patches in the regions around the superpixel. These features are relatively crude and can be replaced by richer features such as spin images [26] or $3D$ shape contexts [17].

In total, these add up to 101 features each for the superpixel and its amodal completion.

5.1.2 Category Specific Features

In addition to features above, we train one-versus-rest SVM classifiers based on appearance and shape of the superpixel, and use the SVM scores for each category as features along with the other features mentioned above. To train these SVMs, we use (1) histograms of vector quantized color SIFT [41] as the appearance features, and (2) histograms of *geocentric textons* (vector quantized words in the joint 2-dimensional space of height from the ground and local angle with the gravity direction) as shape features. This makes up for 40 features each for the superpixel and its amodal completion.

5.2 Results

With the features as described above we experiment with 2 different types of classifiers - (1) random forest classifiers with 40 trees with randomization happening both across features and training points for each tree (we use TreeBagger function in MATLAB), (2) SVM classifiers with additive kernels. At test time, for both these methods, we get a posterior probability for each superpixel of belonging to each of the 40 classes and assign the most probable class to each superpixel.

We use the standard split of NYUD2 with 795 training set images and 654 test set images for evaluation. To prevent over-fitting because of retraining on the same set, we train our category specific SVMs only on half of the train set.

Performance on the 40 category task We measure the performance of our algorithm using the Jaccard index (true predictions divided by union of predictions and true labels - same as the metric used for evaluation in the PASCAL VOC segmentation task) between the predicted pixels and ground truth pixels for each category. As an aggregate measure, we look at the frequency weighted average of the class-wise Jaccard index (*fwavacc*), but for completeness also report the average of the Jaccard index (*avacc*), and the pixel-level classification accuracy (*pixacc*). To understand the quality of the classifiers for each individual category independent of calibration, we also compute *maxIU*, the maximum intersection over union for all thresholds of the classifier score for each category individually, and report their average, and denote this with *mean(maxIU)*.

We report the performance in Table 6 (first 4 rows in the two tables). As baselines, we use [45]-Structure Classifier, where we retrain their structure classifiers for the 40 class task, and [39], where we again retrained their model for this task on this dataset using code available on their website[2]. We observe that we are able to do well on scene surfaces (walls, floors, ceilings, cabinets, counters), and most furniture items (bed, chairs, sofa). We do poorly on small objects, due to limited training data and weak shape features (our features are designed to describe big scene level surfaces and objects). We also consistently outperform the baselines. Fig. 3 presents some qualitative examples, more are provided in the supplemental material.

Ablation Studies In order to gain insights into how much each type of feature contributes towards the semantic segmentation task, we conduct an ablation study by removing parts from the final system. We report our observations in Table 7. Randomized decision forests (RF) work slightly better than SVMs when using only generic or category specific features, but SVMs are able to more effectively combine information when using both these sets of features. Using features from amodal completion also provides some improvement. [45]-SP: we also retrain our system on the superpixels from [45] and obtain better performance than [45] (36.51) indicating that the gain in performance comes in from better features and not just from better bottom-up segmentation. [39] features: we also tried the RGB-D kernel descriptor features from [39] on our superpixels, and observe that they do slightly worse than our category specific features. We also analyse the importance of our RGB-D bottom-up segmentation, and report performance of our system when used with RGB based superpixels from Arbelaez *et al*. [3] (SVM color sp). We note that an improved bottom-up segmentation boosts performance of the semantic segmentation task.

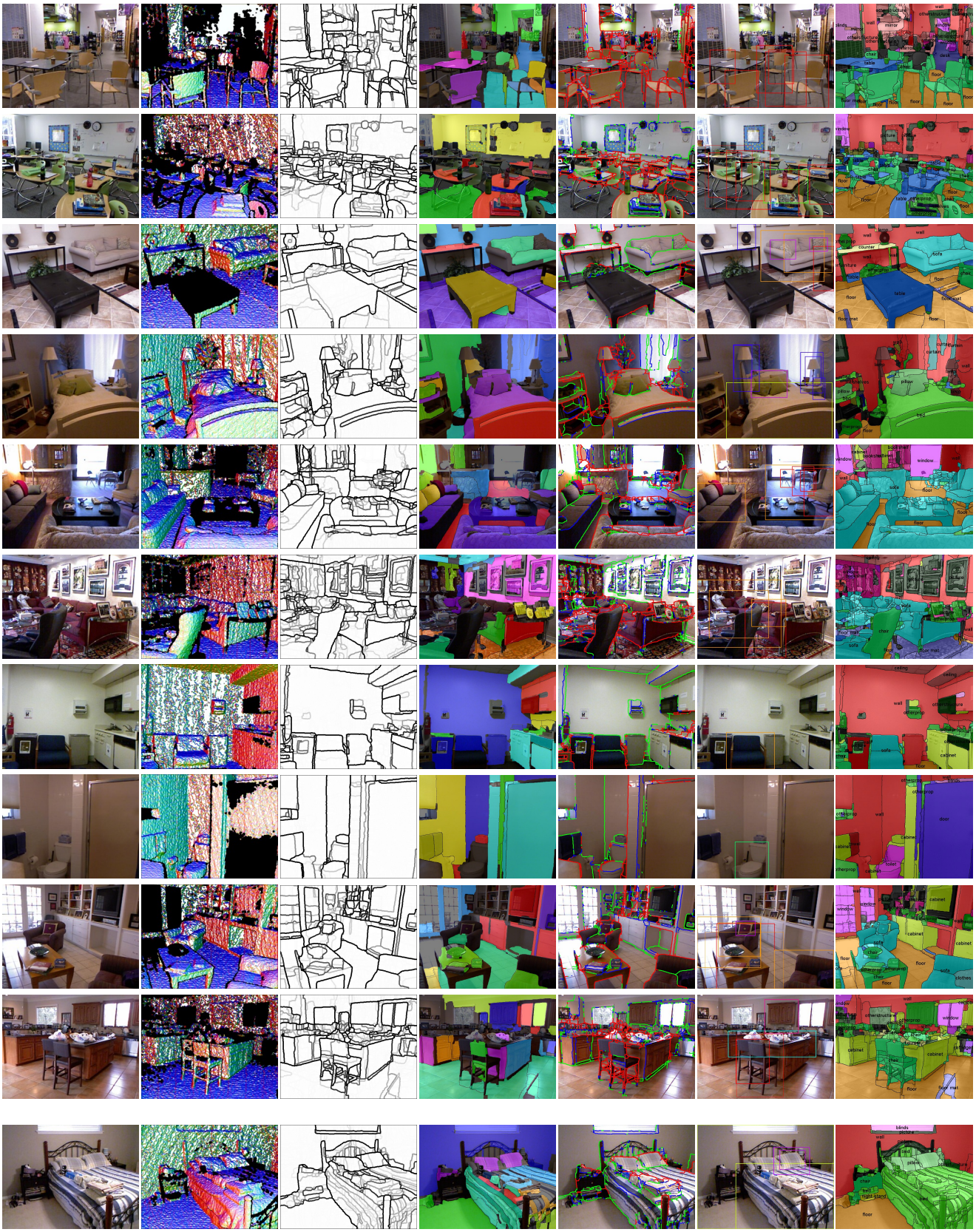
Performance on NYUD2 4 category task We compare our performance with existing results on the super-ordinate category task as defined in [45] in Table 8. To generate predictions for the super-ordinate categories, we simply retrain our classifiers to predict the 4 super-ordinate category labels. As before we report the pixel wise Jaccard index for the different super-categories. Note that this metric is independent of the segmentation used for recognition, and measures the end-to-end performance of the system unlike the metric originally used by [45] (which measures performance in terms of accuracy in predictions on superpixels which vary from segmentation to segmentation). As before, we report *fwavacc*, *avacc*, *pixacc* and *mean(maxIU)* aggregate metrics. As baselines, we compare against [45, 39][3].

6 Detectors and Scene Context for Semantic Segmentation

The features that we proposed in Section 5 try to classify each superpixel independently and do not reason about full object information. To address this limitation, we propose augmenting the features for a superpixel with additional features computed from activations of object detectors (which have access to whole object information), and scene classifiers (which have access to the whole image). The features from object detector activations provide the missing top-down information and scene classifier outputs provide object scene

[2] We run their code on NYUD2 with our bottom-up segmentation hierarchy using the same classifier hyperparameters as specified in their code.

[3] We thank the authors of [45] for providing us with their precomputed results. For [39], as before we retrained their algorithm for the 4 class task.



(a) Color (b) Normal (c) Contours (d) Amodal Comp (e) Contour Type (f) Obj Det (g) Semantic Segm

Fig. 3: **Output of our system**: We take in as input a single color and depth image ((a) and (b)) and produce as output bottom up segmentation (c), long range completion (d), contour classification (e) (into depth discontinuities (red), concave normal discontinuities (green) and convex normal discontinuities (blue)), object detection (f), and semantic segmentation (g).

	wall	floor	cabinet	bed	chair	sofa	table	door	window	book shelf	picture	counter	blinds	desk	shelves
[45]-SC	60.7	77.8	33.0	40.3	32.4	25.3	21.0	5.9	29.7	**22.7**	35.7	33.1	40.6	4.7	3.3
[39]	60.0	74.4	37.1	42.3	32.5	28.2	16.6	12.9	27.7	17.3	32.4	38.6	26.5	**10.1**	**6.1**
our	66.7	80.0	44.0	53.6	35.4	36.3	26.4	11.9	32.4	19.7	39.3	44.2	41.4	3.7	1.4
our + det(RGB)	67.2	81.0	44.0	54.4	36.5	37.8	23.8	**13.7**	**35.9**	18.1	39.8	46.3	42.7	7.4	2.0
our + det	67.4	80.5	41.4	56.4	40.4	44.8	**30.0**	12.1	34.1	20.5	38.7	50.7	**44.7**	10.1	1.6
our + scene	67.6	81.2	44.8	57.0	36.7	40.8	28.0	13.0	33.6	19.5	**41.2**	**52.0**	44.4	7.1	4.5
our + det + scene	**67.9**	**81.5**	**45.0**	**60.1**	**41.3**	**47.6**	29.5	12.9	34.8	18.1	40.7	51.7	41.2	6.7	5.2

	curtain	dresser	pillow	mirror	floor mat	clothes	ceiling	books	fridge	tele vision	paper	towel	shower curtain	box	white board
[45]-SC	27.4	13.3	18.9	4.4	7.1	6.5	**73.2**	5.5	1.4	5.7	12.7	0.1	3.6	0.1	0.0
[39]	27.6	7.0	19.7	17.9	20.1	9.5	53.9	**14.8**	1.9	**18.6**	11.7	12.6	5.4	**3.3**	0.2
our	26.3	19.1	29.5	17.3	25.8	**10.1**	62.3	5.3	12.3	5.2	11.4	20.2	5.5	2.1	11.1
our + det(RGB)	**28.9**	22.6	29.0	13.2	28.7	8.5	61.2	1.9	10.5	8.5	13.8	18.5	4.3	2.1	11.8
our + det	26.3	21.6	31.3	14.6	28.2	8.0	61.8	5.8	14.5	14.4	14.1	19.8	6.0	1.1	**12.9**
our + scene	28.6	24.3	30.3	**23.1**	26.8	7.4	61.1	5.5	**16.2**	4.8	**15.1**	**25.9**	**9.7**	2.1	11.6
our + det + Scene	26.9	**25.0**	**32.8**	21.2	**30.7**	7.7	61.2	7.5	11.8	15.8	14.7	20.0	4.2	1.1	10.9

	person	night stand	toilet	sink	lamp	bathtub	bag	other str	other furntr	other prop	fwavacc	avacc	mean (maxIU)	pixacc	avacc*
[45]-SC	6.6	6.3	26.7	25.1	15.9	0.0	0.0	6.4	3.8	22.4	38.2	19.0	-	54.6	17.5
[39]	**13.6**	9.2	35.2	28.9	14.2	7.8	**1.2**	5.7	5.5	9.7	37.6	20.5	21.4	49.3	20.2
our	0.0	17.3	45.3	31.6	19.7	30.3	0.0	4.2	1.8	22.8	43.4	24.3	27.9	57.9	25.4
our + det(RGB)	4.4	10.2	**56.4**	33.1	19.6	12.3	0.2	1.8	1.0	**23.1**	43.9	24.4	28.3	57.0	25.4
our + det	1.5	15.7	52.5	**47.9**	**31.2**	29.4	0.2	6.4	2.1	21.5	44.9	26.5	29.2	58.9	**30.0**
our + scene	5.0	**21.5**	46.5	35.7	16.3	**31.1**	0.0	**7.9**	**5.7**	22.7	45.2	26.4	29.1	**59.1**	27.5
our + det + scene	1.4	17.9	48.1	45.1	31.1	19.1	0.0	7.6	3.8	22.6	**45.9**	**26.8**	**30.7**	58.3	29.6

Table 6: **Performance on the 40 class task**: We report the pixel-wise Jaccard index for each of the 40 categories. We compare against 2 baselines (row 1 and row 2). *our* is the version of our system as introduced in Section 5 with additive kernel SVM as classifier. *our+det(RGB)* is the version of our system which uses features from RGB object detector activations as described in Section 6.1, *our+det* is the version of our system which uses features from RGB-D object detector activations as described in Section 6.1, *our+scene* is the version of our system which uses 'scene-context' features as described in Section 6.2, and finally *out+scene+det* is the system with both 'scene context' features and RGB-D object detector activation based feature. Categories for which we added detectors are shaded in gray (avacc* is the average for categories for which we added detectors).

	full	only generic	only category	only geom	only app	no amodal	[45]-SP	[39] features
SVM	42.06	35.51	38.69	37.55	31.8	41.17	41.19	36.68
RF	39.4	36.09	39.14	35.08	30.62	39.07	39.92	-
SVM (color sp)	38.45	32.09	35.68	34.92	28.81	37.93	-	-

Table 7: **Ablation study on half of the train set**: All components of our semantic segmentation system contribute to the performance. See text for details.

context information (of the form that night stands occur more frequently in bedrooms than in living rooms). In this section, we describe how we compute these features and show experimental results which illustrate that adding these features helps improve performance for the semantic segmentation task. Figure 4 shows examples of the error modes that get fixed on using these additional features.

6.1 Detector Activations Features

We compute the output of the RGB-D detector that we trained in Section 4, and do the standard DPM non-max suppression. Then, for each class we pick a threshold for the score of the detector such that the detector obtains a precision of $p(= 0.50)$ on the validation set. We then prune away all detections which have a score smaller than this threshold. We use the remaining detections to compute features for each superpixel. We can see this pruning as introducing a non-linearity on the detection scores, allowing the classifier to use information from the good detections more effectively and not getting influenced by the bad detections which are not as informative.

For each superpixel, for each category for which we have a detector, we compute all detections whose

	floor	structure	furniture	prop	fwavacc	avacc	mean (maxIU)	pixacc
[45]-SC	79.5	66.2	51.9	27.1	56.3	56.2	-	71.9
[45]-LP	65.5	65.9	49.9	24	53.4	51.3	-	70
[39]	75	69	54	35	59	58	-	73
our	80.8	72.6	63.1	37.5	64.5	63.5	64.1	77.8
our + det(RGB)	81.0	72.5	62.9	37.5	64.4	63.5	63.9	77.9
our + det	80.9	73.6	**64.1**	38.0	65.3	64.1	64.6	**78.4**
our + scene	81.1	72.9	62.9	36.8	64.4	63.4	64.1	78.0
our + det + scene	**81.1**	**74.0**	64.0	**38.5**	**65.5**	**64.4**	**64.7**	78.1

Table 8: **Performance on the 4 class task**: Comparison with [45,39] on the 4 super-ordinate categories task. We report the pixel-wise Jaccard index for 4 categories, and 3 aggregate metrics: *avg* - average Jaccard index, *fwavacc* - pixel-frequency weighted average Jaccard index, *mean(maxIU)* - average of maxIU for each category, and *pixacc* - pixel accuracy. [45]-SC is the output of the [45]'s structure classifier, [45]-LP is the output obtained after solving the linear program for support inference in [45][3]. See caption from Table 6 for description of last four lines in the Table.

bounding box overlaps with the bounding box of the superpixel. Among these detections, we pick the detection with maximum overlap, and then compute the following features between the superpixel and the picked detection: score of the detection selected, overlap between the detector and superpixel bounding boxes, mean and median depth in the detector box and the superpixel.

With these additional features, we train the same superpixel classifiers that we trained in Section 5. We report the performance we get in Tables 6 and 8. *our + det(RGB)* corresponds to when we use RGB DPMs to compute these features and *our + det* corresponds to when we use our proposed RGB-D DPM detectors. We observe very little improvement when using RGB DPMs but a large improvement when using RGB-D DPMs, for which we see improvement in performance across all aggregate metrics and for most of the categories for which we added object detectors (these categories are shaded in gray).

Fig. 4: **Examples illustrating where object detectors and scene classification help**: Semantic segmentation output improves as we add features from object detector activations and scene classifiers (going from left image to right image).

6.2 Scene Classifier Features

We use the scene label annotations provided in the NYUD2 dataset (we only consider the most common 9 scene categories and map the remaining into a class 'other'), to train a scene classifier. To train these scene classifiers, we use features computed by average pooling the prediction for each of the 40 classes in a $1, 2\times2, 4\times4$ spatial pyramid [33], and training an additive kernel SVM. We find that these features perform comparable to the other baseline features that we tried Appendix D.

We then use these scene classifiers to compute additional features for superpixels in the image and train the same superpixel classifiers that we trained in Section 5. We report the performance we get in Tables 6 and 8 (our + scene). We observe that there is a consistent improvement which is comparable to the improvement that we get when using detector activation features. As a final experiment, we use both scene classifier features and object detector activation feature and see a further improvement in performance.

7 Conclusion

We have developed a set of algorithmic tools for perceptual organization and recognition in indoor scenes from RGB-D data. Our system produces contour detection, hierarchical segmentation, grouping by amodal completion, object detection and semantic labeling of objects and scene surfaces. We report significant improvements over the state-of-the-art in all of these tasks.

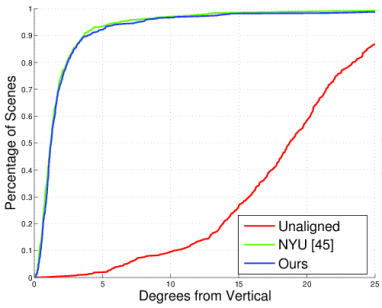


Fig. 5: Cumulative distribution of angle of the floor with the estimated gravity direction.

A Extracting a Geocentric Coordinate Frame

We note that the direction of gravity imposes a lot of structure on how the real world looks (the floor and other supporting surfaces are always horizontal, the walls are always vertical). Hence, to leverage this structure, we develop a simple algorithm to determine the direction of gravity.

Note that this differs from the Manhattan World assumption made by, *e.g.* [19] in the past. The assumption that there are 3 principal mutually orthogonal directions is not universally valid. On the other hand the role of the gravity vector in architectural design is equally important for a hut in Zimbabwe or an apartment in Manhattan.

Since we have depth data available, we propose a simple yet robust algorithm to estimate the direction of gravity. Intuitively, the algorithm tries to find the direction which is the most aligned to or most orthogonal to locally estimated surface normal directions at as many points as possible. The algorithm starts with an estimate of the gravity vector and iteratively refines the estimate via the following 2 steps.

1. Using the current estimate of the gravity direction $\boldsymbol{g}_{i-1}$, make hard-assignments of local surface normals to aligned set $\mathcal{N}_{\parallel}$ and orthogonal set $\mathcal{N}_{\perp}$, (based on a threshold d on the angle made by the local surface normal with $\boldsymbol{g}_{i-1}$). Stack the vectors in $\mathcal{N}_{\parallel}$ to form a matrix $N_{\parallel}$, and similarly in $\mathcal{N}_{\perp}$ to form $N_{\perp}$.
$$\mathcal{N}_{\parallel} = \{\boldsymbol{n} : \theta(\boldsymbol{n}, \boldsymbol{g}_{i-1}) < d \text{ or } \theta(\boldsymbol{n}, \boldsymbol{g}_{i-1}) > 180^\circ - d\}$$
$$\mathcal{N}_{\perp} = \{\boldsymbol{n} : 90^\circ - d < \theta(\boldsymbol{n}, \boldsymbol{g}_{i-1}) < 90^\circ + d\}$$
where, $\theta(\boldsymbol{a}, \boldsymbol{b})$ = Angle between $\boldsymbol{a}$ and $\boldsymbol{b}$.
Typically, $\mathcal{N}_{\parallel}$ would contain normals from points on the floor and table-tops and $\mathcal{N}_{\perp}$ would contain normals from points on the walls.
2. Solve for a new estimate of the gravity vector $\boldsymbol{g}_i$ which is as aligned to normals in the aligned set and as orthogonal to the normals in the orthogonal set as possible. This corresponds to solving the following optimization problem, which simplifies into finding the eigen-vector with the smallest eigen value of the 3×3 matrix, $N_{\perp} N_{\perp}^t - N_{\parallel} N_{\parallel}^t$.
$$\min_{\boldsymbol{g}:\|\boldsymbol{g}\|_2=1} \sum_{\boldsymbol{n} \in \mathcal{N}_{\perp}} \cos^2(\theta(\boldsymbol{n}, \boldsymbol{g})) + \sum_{\boldsymbol{n} \in \mathcal{N}_{\parallel}} \sin^2(\theta(\boldsymbol{n}, \boldsymbol{g}))$$

Our initial estimate for the gravity vector $\boldsymbol{g}_0$ is the Y-axis, and we run 5 iterations with $d = 45^\circ$ followed by 5 iterations with $d = 15^\circ$.

To benchmark the accuracy of our gravity direction, we use the metric of [45]. We rotate the point cloud to align the Y-axis with the estimated gravity direction and look at the angle the floor makes with the Y-axis. We show the cumulative distribution of the angle of the floor with the Y-axis in figure 5. Note that our gravity estimate is within 5° of the actual direction for 90% of the images, and works as well as the method of [45], while being significantly simpler.

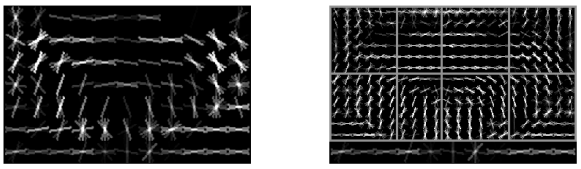


Fig. 7: Root and part filters for the bed. We can see that the model captures the shape for a bed. Horizontal lines correspond to horizontal surfaces and the vertical lines correspond to vertical surface. We can see that the model learnt a box which we are looking at towards one of its corners.

B Histogram of Depth Gradients

Suppose we are looking at a plane $N_X X + N_Y Y + N_Z Z + d = 0$ in space. A point (X, Y, Z) in the world gets imaged at the point $\left(x = \frac{fX}{Z}, y = \frac{fY}{Z}\right)$, where f is the focal length of the camera. Using this in the first equation, we get the relation, $N_X \frac{Zx}{f} + N_Y \frac{Zy}{f} + N_Z Z + d = 0$, which simplifies to give $Z = \frac{-fd}{fN_Z + N_Y y + N_X x}$. Differentiating this with respect to the image gradient gives us,

$$\frac{\partial Z}{\partial x} = \frac{N_X Z^2}{df} \tag{1}$$

$$\frac{\partial Z}{\partial y} = \frac{N_Y Z^2}{df} \tag{2}$$

Using this with the relation that relates disparity δ with depth value Z, $\delta = \frac{bf}{Z}$, where b is the baseline for the Kinect, gives us the derivatives for the disparity δ to be

$$\frac{\partial \delta}{\partial x} = \frac{-bN_X}{d} \tag{3}$$

$$\frac{\partial \delta}{\partial y} = \frac{-bN_Y}{d} \tag{4}$$

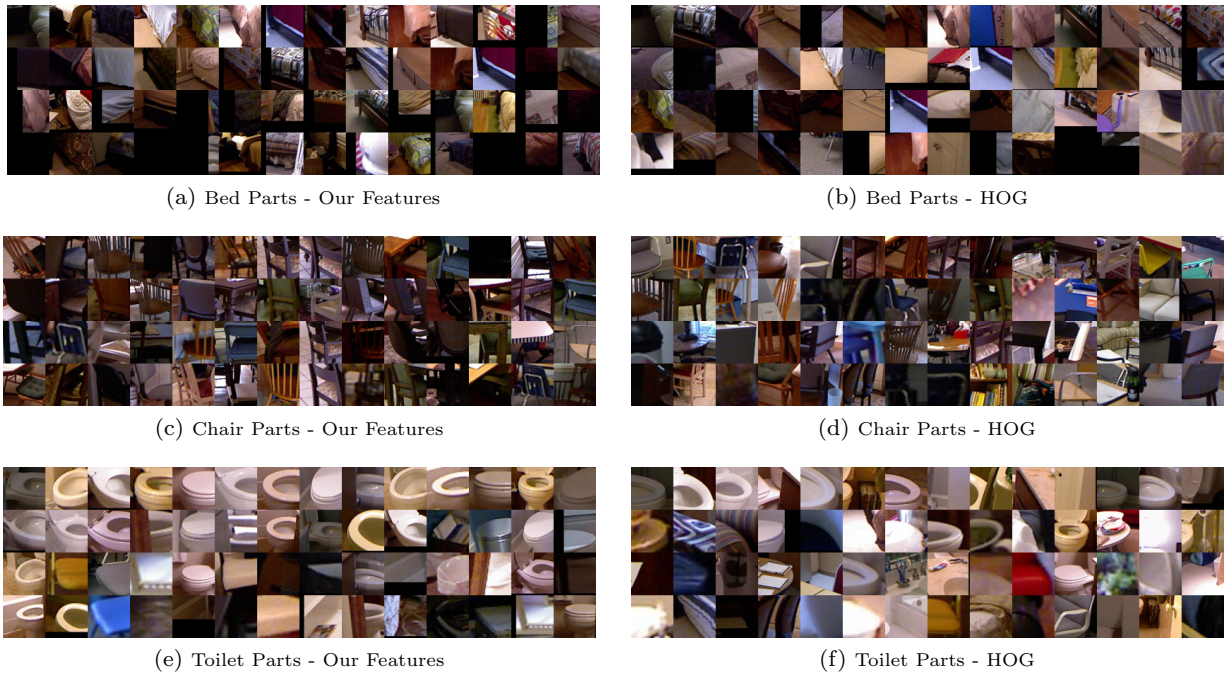
Thus, the *gradient orientation* for both the disparity and the depth image comes out to be $tan^{-1}(\frac{N_Y}{N_X})$ (although the contrast is swapped). The *gradient magnitude* for the depth image is $\frac{Z^2\sqrt{1-N_Z^2}}{df} = \frac{Z^2 \sin(\theta)}{df}$, and for the disparity image is $\frac{b\sqrt{1-N_Z^2}}{d} = \frac{b \sin(\theta)}{d}$, where θ is the angle that the normal at this point makes with the image plane.

Note that, the *gradient magnitude* for the disparity and the depth image differ in that the depth gradient has a factor of Z^2, which makes points further away have a larger gradient magnitude. This agrees well with the noise model for the Kinect (quantization of the disparity value, which leads to an error in Z which value proportional to Z^2). In this sense, the disparity gradient is much better behaved than the gradient of the depth image.

Note that, the subsequent contrast normalization step in standard HOG computation, essentially gets rid of this difference between these 2 quantities (assuming that the close by cells have more or less comparable Z values).

C Visualization for RGB-D Detector Parts

One interesting thing to visualize is what the DPM is learning with these features. The question that we want to ask here is



(a) Bed Parts - Our Features (b) Bed Parts - HOG

(c) Chair Parts - Our Features (d) Chair Parts - HOG

(e) Toilet Parts - Our Features (f) Toilet Parts - HOG

Fig. 6: Visualization for the DPM parts for bed, chairs and toilets

that whether the parts that we get semantically meaningful? The hope is that with access to depth data, the parts that get discovered should be more meaningful than ones you get with purely appearance data.

In Figure 6, we visualize the various DPM parts for the bed, chair and toilet detector. We run the detector on a set of images that the detector did not see at train time, pick the top few detections based on the detector score. We then crop out the part of the image that a particular part of the DPM got placed at, and visualize these image patches for the different DPM parts.

We observe that the parts are tight semantically - that is, a particular part likes semantically similar regions of the object class. For comparison, we also provide visualizations for the parts that get learnt for an appearance only DPM. As expected, the parts from our DPM are semantically tighter than the parts from an appearance only DPM. Recently, in the context of intensity images there has been a lot of work in trying to get mid-level parts in an unsupervised manner from weak annotations like that of bounding boxes in intensity images [13], and in a supervised manner from strong annotations like that of keypoint annotation [5]. These visualizations suggest that it may be possible to get very reasonable mid-level parts from weak annotations in RGB-D images, which can be used to train appearance only part detectors.

We also visualize what the *HHG* features learn. In Figure7, we see that the model as expected picks on the shape cues. There is a flat horizontal surface along the sides and on the middle portion which corresponds to the floor and the top of the bed and there are vertical surfaces going from the horizontal floor to the top of the bed.

D Scene Classification

We address the task of indoor scene classification based on the idea that a scene can be recognized by identifying the objects in it. Thus, we use our predicted semantic segmentation maps as features for this task. We use the spatial pyramid (SPM) formulation of [33], but instead of using histograms of vector quantized SIFT descriptors as features, we use the average presence of each semantic class (as predicted by our algorithm) in each pyramid cell as our feature.

To evaluate our performance, we use the scene labels provided by the NYUD2. The dataset has 27 scene categories but only a few are well represented. Hence, we reduce the 27 categories into 10 categories (9 most common categories and the rest). As before, we train on the 795 images from the train set and test on the remaining 654 images. We report the diagonal of the confusion matrix for each of the scene class and use the mean of the diagonal, and overall accuracy as aggregate measures of performance.

We use a 1, 2×2, 4×4 spatial pyramid and use a SVM with an additive kernel as our classifier [36]. We use the 40 category output of our algorithm. We compare against an appearance-only baseline based on SPM on vector quantized color SIFT descriptors [41], a geometry-only baseline based on SPM on *geocentric textons* (introduced in Section 5.1.2), and a third baseline which uses both SIFT and Geocentric Textons in the SPM.

We report the performance we achieve in Table 9.

Acknowledgements We are thankful to Jon Barron, Bharath Hariharan, and Pulkit Agrawal for the useful discussions.

	SPM on SIFT	SPM on G. Textons	SPM on SIFT + G.Textons	SPM on Our Output
bedroom	78.0	70.7	80.6	77.5
kitchen	67.9	58.5	73.6	76.4
living room	39.3	32.7	40.2	41.1
bathroom	44.8	56.9	65.5	74.1
dining room	41.8	23.6	45.5	30.9
office	34.2	15.8	34.2	5.3
home office	0.0	8.3	16.7	4.2
classroom	60.9	43.5	60.9	52.2
bookstore	0.0	18.2	0.0	72.7
others	22.0	9.8	31.7	19.5
Mean Diagonal	38.9	33.8	44.9	45.4
Accuracy	53.2	46.2	58.4	55.7

Table 9: **Performance on the scene classification task**: We report the diagonal entry of the confusion matrix for each category; and the mean diagonal of the confusion matrix and the overall accuracy as aggregate metrics. 'G. Textons' refer to Geocentric Textons introduced in Section 5.1.2.

References

1. Reconstruction meets recognition challenge, iccv 2013. http://ttic.uchicago.edu/~rurtasun/rmrc/index.php (2013)
2. Arbelaez, P., Hariharan, B., Gu, C., Gupta, S., Bourdev, L., Malik, J.: Semantic segmentation using regions and parts. CVPR (2012)
3. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. TPAMI (2011)
4. Barron, J.T., Malik, J.: Intrinsic scene properties from a single RGB-D image. In: CVPR (2013)
5. Bourdev, L., Maji, S., Brox, T., Malik, J.: Detecting people using mutually consistent poselet activations. In: ECCV (2010)
6. Breiman, L.: Random forests. Machine Learning (2001)
7. Carreira, J., Caseiro, R., Batista, J., Sminchisescu, C.: Semantic segmentation with second-order pooling. In: ECCV (2012)
8. Carreira, J., Caseiro, R., Batista, J., Sminchisescu, C.: Semantic segmentation with second-order pooling. In: ECCV. Springer Berlin Heidelberg (2012)
9. Carreira, J., Li, F., Sminchisescu, C.: Object Recognition by Sequential Figure-Ground Ranking. IJCV (2012)
10. Criminisi, A., Shotton, J., Konukoglu, E.: Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. Found. and Trends in Comp. Graphics and Vision (2012)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR (2005)
12. Dollár, P., Zitnick, C.L.: Structured forests for fast edge detection. ICCV (2013)
13. Endres, I., Shih, K.J., Jiaa, J., Hoiem, D.: Learning collections of part models for object recognition. In: CVPR (2013)
14. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
15. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes (VOC) Challenge. IJCV (2010)
16. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. TPAMI (2010)
17. Frome, A., Huber, D., Kolluri, R., Bülow, T., Malik, J.: Recognizing objects in range data using regional point descriptors. ECCV (2004)
18. Gupta, A., Efros, A., Hebert, M.: Blocks world revisited: Image understanding using qualitative geometry and mechanics. ECCV (2010)
19. Gupta, A., Satkin, S., Efros, A., Hebert, M.: From 3D scene geometry to human workspace. CVPR (2011)
20. Gupta, S., Arbelaez, P., Malik, J.: Perceptual organization and recognition of indoor scenes from RGB-D images. CVPR (2013)
21. Hedau, V., Hoiem, D., Forsyth, D.: Recovering free space of indoor scenes from a single image. CVPR (2012)
22. Hoiem, D., Efros, A., Hebert, M.: Recovering surface layout from an image. IJCV (2007)
23. Hoiem, D., Efros, A., Hebert, M.: Recovering occlusion boundaries from an image. IJCV (2011)
24. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A.: KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. UIST (2011)
25. Janoch, A., Karayev, S., Jia, Y., Barron, J.T., Fritz, M., Saenko, K., Darrell, T.: A category-level 3D object dataset: Putting the kinect to work. In: Consumer Depth Cameras for Computer Vision, pp. 141–165. Springer (2013)
26. Johnson, A., Hebert, M.: Using spin images for efficient object recognition in cluttered 3D scenes. TPAMI (1999)
27. Kanizsa, G.: Organization in Vision: Essays on Gestalt Perception. Praeger Publishers (1979)
28. soo Kim, B., Xu, S., Savarese, S.: Accurate localization of 3D objects from RGB-D data using segmentation hypotheses. In: CVPR (2013)
29. Koppula, H., Anand, A., Joachims, T., Saxena, A.: Semantic labeling of 3d point clouds for indoor scenes. NIPS (2011)
30. Ladicky, L., Russell, C., Kohli, P., Torr, P.H.S.: Graph cut based inference with co-occurrence statistics. ECCV (2010)
31. Lai, K., Bo, L., Ren, X., Fox, D.: A large-scale hierarchical multi-view RGB-D object dataset. ICRA (2011)
32. Lai, K., Bo, L., Ren, X., Fox, D.: Rgb-d object recognition: Features, algorithms, and a large scale benchmark. In: A. Fossati, J. Gall, H. Grabner, X. Ren, K. Konolige (eds.) Consumer Depth Cameras for Computer Vision: Research Topics and Applications, pp. 167–192. Springer (2013)
33. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. CVPR (2006)
34. Lee, D., Gupta, A., Hebert, M., Kanade, T.: Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. NIPS (2010)
35. Lee, D., Hebert, M., Kanade, T.: Geometric reasoning for single image structure recovery. CVPR (2009)
36. Maji, S., Berg, A.C., Malik, J.: Efficient classification for additive kernel svms. TPAMI (2013)
37. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color and texture cues. TPAMI (2004)

38. Ren, X., Bo, L.: Discriminatively trained sparse code gradients for contour detection. NIPS (2012)
39. Ren, X., Bo, L., Fox, D.: RGB-(D) scene labeling: Features and algorithms. CVPR (2012)
40. Rusu, R.B., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: ICRA (2009)
41. van de Sande, K.E.A., Gevers, T., Snoek, C.G.M.: Evaluating color descriptors for object and scene recognition. TPAMI (2010)
42. Savitsky, A., Golay, M.: Smoothing and differentiation of data by simplified least squares procedures. Analytical Chemistry (1964)
43. Saxena, A., Chung, S., Ng, A.: 3-d depth reconstruction from a single still image. IJCV (2008)
44. Shotton, J., Fitzgibbon, A.W., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. CVPR (2011)
45. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from RGBD images. ECCV (2012)
46. Tang, S., Wang, X., Lv, X., Han, T.X., Keller, J., He, Z., Skubic, M., Lao, S.: Histogram of oriented normal vectors for object recognition with a depth sensor. In: ACCV (2012)
47. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR (2001)
48. Ye, E.S.: Object detection in RGB-D indoor scenes. Master's thesis, EECS Department, University of California, Berkeley (2013)