Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images

Saurabh Gupta, Pablo Arbeláez, and Jitendra Malik
University of California, Berkeley - Berkeley, CA 94720
{sgupta, arbelaez, malik}@eecs.berkeley.edu

Abstract

We address the problems of contour detection, bottom-up grouping and semantic segmentation using 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 [27]. We propose algorithms for object boundary detection and hierarchical segmentation that generalize the gPb–ucm approach of [2] 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 then turn to the problem of semantic segmentation and propose a simple approach that classifies superpixels into the 40 dominant object categories in NYUD2. We use both generic and class-specific features to encode the appearance and geometry of objects. We also show how our approach can be used for scene classification, and how this contextual information in turn improves object recognition. In all of these tasks, we report significant improvements over the state-of-the-art.

1. Introduction

In this paper, we study the problem of image understanding in indoor scenes from color and depth data. The output of our approach is shown in Figure 1: given a single RGB-D image, our system produces contour detection and bottom-up segmentation, grouping by amodal completion, and semantic labeling of objects and scene surfaces.

The problem of three dimensional scene understanding from monocular images has received considerable attention in recent years [11, 8, 10, 20, 19, 25], and different aspects have been explored. Some works have addressed the task of inferring coarse 3D layout of outdoor scenes, exploiting appearance and geometric information [11, 25]. Recently, the focus has shifted towards the more difficult case of cluttered indoor scenes [10, 20, 19, 9]. In this context, the notion of affordance and the functionality of objects for human use acquires importance. Thus, [10] recovers walkable surfaces by reasoning on the location and shape of furniture. [20, 19] reason about the 3D geometry of the room and objects, while [9] focuses on interpreting the scene in a human-centric perspective.

With the availability of consumer RGB-D sensors (like the Microsoft Kinect), a new area of research has opened up in computer vision around using 3D information for tasks which have traditionally been very hard. A good example is its first application for real-time human pose recognition [26], in use in the Microsoft XBOX. Subsequently, there have also been numerous papers in both robotics and vision communities looking at object and instance recognition using RGB-D data [28, 17] and dense reconstruction of indoor scenes and objects from multiple scenes. Using range data for recognition has a long history, some examples being spin images [13] and 3D shape contexts [7]. The focus of these works has mostly been object recognition and not scene understanding.

More recent research which directly relates to our work is Silberman et al. [27]. They also look at the task of bottom-up RGB-D segmentation and semantic scene labeling. They modify the algorithm of [12] to use depth for bottom-up segmentation and then look at the task of seman-
tic segmentation using context features derived from inferring support relationships in the scene. Ren et al. [23] use features based on kernel descriptors on superpixels and its ancestors from a region hierarchy followed by a Markov Random Field (MRF) based context modeling. Koppula et al. [15] 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 3D point cloud is over-segmented and used as underlying structure for an MRF model.

Our work differs from the references above in both our approach to segmentation and to recognition. We visit the segmentation problem afresh from ground-up and develop a gPb [2] like machinery which combines depth information naturally, giving us significantly better bottom-up segmentation when compared to earlier works. We also look at the interesting problem of amodal completion [14] and obtain long range grouping giving us much better bottom-up region proposals. Finally, we are 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 [6]. We observe that approaches like [4, 1], which focus on classifying bottom-up regions candidates using strong features on the region have obtained significantly better results than MRF-based methods [16]. We build on this motivation and propose new features to represent bottom-up region proposals (which in our case are non-overlapping superpixels and their amodal completion), and use randomized decision tree forest [3, 5] and SVM classifiers.

In addition to semantic segmentation, we look at the problem of RGB-D scene classification and show that knowledge of the scene category helps improve accuracy of semantic segmentation.

This paper is organized as follows. In Sect. 2 we propose an algorithm for estimating the gravity direction from a depth map. In Sect. 3 we describe our algorithm and results for perceptual organization. In Sect. 4 we describe our semantic segmentation approach, and then apply our machinery to scene classification in Sect. 5.

2. 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. [9] 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 \( g_{i-1} \), make hard-assignments of local surface normals to aligned set \( N_\parallel \) and orthogonal set \( N_\perp \), (based on a threshold \( d \) on the angle made by the local surface normal with \( g_{i-1} \)). Stack the vectors in \( N_\parallel \) to form a matrix \( N_{\parallel} \), and similarly in \( N_\perp \) to form \( N_{\perp} \).

   \[
   \begin{align*}
   N_\parallel &= \{ n : \theta(n, g_{i-1}) < d \text{ or } \theta(n, g_{i-1}) > 180^\circ - d \} \\
   N_\perp &= \{ n : 90^\circ - d < \theta(n, g_{i-1}) < 90^\circ + d \}
   \end{align*}
   \]

   where, \( \theta(a, b) = \text{Angle between } a \text{ and } b \).

   Typically, \( N_\parallel \) would contain normals from points on the floor and table-tops and \( N_\perp \) would contain normals from points on the walls.

2. Solve for a new estimate of the gravity vector \( g_i \), 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 \( 3 \times 3 \) matrix, \( N_\perp N_\parallel^T - N_\parallel N_\perp^T \).

   \[
   \min_{g:||g||_2=1} \sum_{n \in N_\perp} \cos^2(\theta(n, g)) + \sum_{n \in N_\parallel} \sin^2(\theta(n, g))
   \]

   Our initial estimate for the gravity vector \( g_0 \) 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 [27]. 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 2.
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 [2], 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 and of 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 $\delta Z$ is the error in depth observation, $Z$ is the actual depth, $\delta d$ 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 [24] 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 [22] 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 geometric 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 [22, 12, 2]. 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 over-segmentation guarantees full recall for the contour locations. We then separate all the 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 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 [2] 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 [21], 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 [2] to construct a hierarchy of segmentations, by merging regions of the initial over-segmentation 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 train on the 795 images of the train set and evaluate our performance on the 654 images of the test set. In the dominant orientations (i.e. horizontal and vertical), we obtain in the order of $10^8$ data points both for training and testing, and an order of magnitude less candidates for the other orientations.

We evaluate performance using the standard benchmarks of the Berkeley Segmentation Dataset [2]: Precision-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 [27], made available by the authors (noted NYUD2 baseline), which produces a small set (5) of nested segmentations using color and depth.

We perform 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 0.56 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.

Figure 3 and Table 1 present the results. Our depth-aware segmentation system produces contours of far higher quality 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 [27], 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 0.56), 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 0.56 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
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 2, 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.

**Shape Features:** These include - planarity of the superpixel, this makes up 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.

**Size Features:** These 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.

**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 2, 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, this makes up 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.

**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 2, 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.

**Shape Features:** These include - planarity of the superpixel, this makes up 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.

**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.

**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 2, 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.
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.

We report the performance in Table 3 (first 4 rows in the two tables). As baselines, we use [27]-Structure Classifier, where we retrain their structure classifiers for the 40 class task, and [23], where we again retrained their model for this task on this dataset using code available on their website. 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. 4 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 4. 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. [27]-SP: we also retrain our system on the superpixels from [27] and obtain better performance than [27] (36.51) indicating that the gain in performance comes in from better features and not just from better bottom-up segmentation. [23] features: we also tried the RGB-D kernel descriptor features from [23] on our superpixels, and observe that they do slightly worse than our category specific features.

**Performance on NYUD2 4 category task** We compare our performance with existing results on the super-ordinate category task as defined in [27] in Table 5. 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 [27] (which measures performance in terms of accuracy in predictions on superpixels which vary from segmentation to segmentation). We use the pixel-frequency weighted average of the Jaccard index as the aggregate metric (fwavacc) and pixel accuracy. [27]-SC is the output of the [27]'s structure classifier, [27]-LP is the output obtained after solving the linear program for support inference in [27].

**5. 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 [18], 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.

---

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

3We thank the authors of [27] for providing us with their precomputed results. For [23], as before we retrained their algorithm for the 4 class task.
Table 3: 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(RF), Our(SVM) are versions of our system using randomized decision forests, and additive kernel support vector machines respectively (Section 4.2). Our(RF+Scene) and Our(SVM+Scene) are versions of our system which use the inferred scene category as additional context features (Section 5).

Figure 4: Output of our system: We take in as input a single color and depth image and produce as output bottom up segmentation, long range completion, semantic segmentation and contour classification (into depth discontinuities (red), concave normal discontinuities (green) and convex normal discontinuities (blue)). Best viewed in color.

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 [21]. 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 [29], a geometry-only baseline based on SPM on geocentric textons (introduced in Sect. 4.1.2), and a third baseline which uses both SIFT and Geocentric Textons in the SPM.

We report the performance we achieve in Table 7. We observe that we are able to correctly classify 58% of the scenes, outperforming all the baselines.
We then do an additional experiment of using the predicted scene information as additional features (‘scene context’) for the superpixels and obtain a performance of 45.3% (up from 44%) for the 40 category task, while giving about the same performance for the 4 class task. Performance on the individual categories is given in Tables 3, 5, 6.

6. 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 and hierarchical segmentation, grouping by amodal completion, and semantic labeling of objects and scene surfaces. We report significant improvements over the state-of-the-art in all of these tasks.

Acknowledgements: This work was sponsored by ONR SMARTS MURI N00014-09-1-1051, ONR MURI N00014-10-10933, and a Berkeley Fellowship. We are also thankful to Jon Barron, Bharath Hariharan, and Pulkit Agrawal for the useful discussions.

References