Semantic Parsing for Priming Object Detection in Indoors RGB-D Scenes*

César Cadena †and Jana Košecká ‡

Abstract

The semantic mapping of the environment requires simultaneous segmentation and categorization of the acquired stream of sensory information. The existing methods typically consider the semantic mapping as the final goal and differ in the number and types of considered semantic categories. We envision semantic understanding of the environment as an on-going process and seek representations which can be refined and adapted depending on the task and robot's interaction with the environment. In this work we propose a novel and efficient method for semantic parsing, which can be adopted to the task at hand and enables localization of objects of interest in indoors environments. For basic mobility tasks we demonstrate how to obtain initial semantic segmentation of the scene into *ground, structure, furniture* and *props* categories which constitute the first level of hierarchy. Then, we propose a simple and efficient method for predicting locations of objects that based on their size afford a manipulation task. In our experiments we use the publicly available NYU V2 dataset [35] and obtain better or comparable results than the state of the art at the fraction of computational cost. We show the generalization of our approach on two more publicly available datasets.

1 Introduction

In recent years numerous advances have been made in semantic mapping of environments. It has been recognized that the capability of associating semantic concepts with geometric entities in robot's surroundings can enhance

*This research has been funded by the US Army Research Office Grant W911NF-1110476 and the Australian Research Council grant DP130104413.

†César Cadena is with the Department of Computer Science, at the University of Adelaide, Adelaide, SA 5005, Australia. cesar.cadena@adelaide.edu.au.

‡Jana Košecká is with the Computer Science Department, Volgenau School of Engineering at George Mason University, Fairfax, VA 20030, US. kosecka@gmu.edu.

robot's autonomy and robustness, facilitate more complex tasks and enable better human robot interaction. These observations led to a large number of approaches, which proposed varying numbers and types of semantic concepts and means of associating them with different parts of the environments. Examples of these included names of rooms/locations and various object and non-object categories. With the exception of few approaches, semantic parsing at the basic level was formulated as a classification problem, where simple mapping between the sensory data and semantic concepts has been considered. In the proposed work we argue that semantic concepts are naturally arranged in a hierarchy and hierarchical categorization is often more suitable, both in terms of efficiency and adaptability for specific tasks. While in the context of lifelong learning it is desirable to be able to recognize large number of object categories, it is not necessary to try to recognize them at all times. For example for basic mobility tasks it is sufficient to be able to understand the free space, walls and obstacles, but it is not necessary to recognize variety of objects in indoors environments. On the other hand in the context of object search, more detailed categorization and localization is needed, if one were to facilitate 'pick up and place' tasks. In this work we demonstrate how to obtain initial semantic segmentation of the scene into *ground, structure, furniture* and *props* categories, which are commonly encountered in indoors environments. These categories along with the geometric features will constitute the first level of hierarchy. This initial semantic segmentation and confidences about presence of objects, will be followed by computation of additional features and generation of hypotheses for objects that based on their size afford a manipulation task.

Proposed Approach We formulate the semantic labeling problem in the CRF framework, where the dependencies between random variables are represented by a graph, induced by both image superpixels and 3D scene structure. The distinguishing features of our approach are: a) the use of a tree graph structure in the CRF setting which effectively approximates the dependencies and enables exact and efficient inference amenable for real-time implementation; b) the use of simple and efficient appearance and geometric cues, providing evidence about depth discontinuities. We carry out the semantic parsing experiments on NYU V2 dataset [35] [1], which contains 464 diverse indoor scenes and 1449 annotated frames, achieving superior

[1] Authors of NYU V2 dataset maintain an excellent website with full description of the data and labeling provided, at: `http://cs.nyu.edu/~silberman/datasets/nyu_depth_v2.html`

or comparable performance at the fraction of computational cost. We also show qualitative and quantitative results on the B3DO and UWobj datasets [19, 24]. c) We show how our system can be integrated with object detectors, in two ways: by pruning the set of candidate locations for less expensive feature computation, and by using the semantic classes' probabilities outcome as a part of the feature vector.

In the next section, we provide an overview of the related work. In Section 3 we describe the details of our approach. Section 5 describes the experiments on NYU V2 dataset and compares our approach with the state of the art methods. Finally, in Section 6 we present discussions and conclusions of the presented work and discuss possible future directions.

2 Related Work

Previous approaches for semantic segmentation varied in the number and type of semantic concepts, means of associating them with different parts of the environments and computational models considered. Examples of most commonly encountered semantic categories included names of rooms or locations and various object and non-object categories. In the following section we mostly focus on the review of more recent methods which exploit both 3D geometric and appearance features computed from RGB-D sensors (e.g. Kinect) or obtained by 3D reconstruction.

In indoors environments several methods have been developed exploiting the RGB-D data. In [22] authors highlighted the need for efficiency of the final inference and used up to 17 object classes. They were able to exploit stronger appearance and contextual cues due to the scale and different nature of the environment. Recently several researchers carried out more comprehensive experiments on larger NYU RGB-D dataset introduced in [34]. Authors in [32] focus on local patch based kernel features and achieved very good average performance while considering 13 structural and furniture classes and grouping all the smaller objects in 'other' category. The proposed features are computed over high quality superpixels obtained with computationally expensive boundary detector [27]. In addition to inference of semantic labels in the work of [35] the authors simultaneously considered the problem of inference of the support relations between different semantic categories. The approach relied on elaborate pre-processing stage involving hierarchical segmentation stage, reasoning about occlusion boundaries and piece-wise planar segmentation. All these stages required a solution to a separate inference problem, using additional features and stage specific energy functions. In the final inference problem the feature vectors computed over superpixels were over 1000 dimensions. A similar strategy is followed by [16] where the authors used the available depth information to improve the initial segmentation, followed by classification of obtained segments. In order to associate disconnected segments belonging to the same object category, they also propose a long-range amodal completion to improve the segmentation consistency. The above mentioned approaches relied on improvements of bottom up segmentation using the depth data as additional cue, following by classification of obtained regions. In the work of [10] authors bypass the complex feature computation and segmentation stage and and use convolutional networks for semantic segmentation. The final hypotheses are then evaluated on over-segmentation for the indoors scenes labeling task.

In contrast to our work the methods reviewed above consider the semantic segmentation as a final goal and are computationally expensive. The proposed approach infers the coarsest level of the semantic hierarchy which can be further refined based on the task.

Another class of approaches for semantic scene understanding focus on the problem of object detection, with or without the bottom up segmentation stage. The framework based on sliding window object detectors has been explored in [25, 36] using RGB-D sensors. In their work the objects are viewed in a table top setting at moderate scale. The authors formulated the object detection problem as an inference on a voxel grid, reconstructed from multiple frames of RGB-D data. The final inference is carried in MRF framework, where the data term accumulates evidence from the sliding window based detectors trained on different views of the objects. A variant of the HOG descriptor [24] was used for capturing the appearance and shape information of each view of an object and trained using SVMs. The larger extent of the objects in the dataset [24] and sufficient number of training examples made the use of HOG based detector feasible. Another related work on unsupervised object discovery [8] has shown promising results for closer range and small amount of clutter. The work of [28] also considers the problem of detection of generic simple objects in an unsupervised setting, relying on the computation of the boundary using both RGB and depth data, followed by a selection of salient points and boundary completion. This methods is very effective for closer range table top settings, where both depth discontinuities and support surfaces can be well estimated and the process of detection of image contours is more reliable. Their methods relies on a high quality contour detector [27], which is quite expensive to compute. The generic object hypotheses proposed by this detector were then fed to a specific contour based classifier using so called torque descriptor [38]. The approaches, which focus on object detection and categorization are suitable for the close range operation where the place that contain the objects has been already determined and the robot is only deciding which object pick up.

Several approaches have been proposed for “objecteness” detection using image-only information [2, 5, 6, 12]. These approaches again are focused on work well on object-centered images and, with the exception of [6], typically take in the order of seconds or even minutes to process every frame. In Section 5.2.2 we include comparisons with the recent proposed generic object detector [6], which has shown better performance than similar image-only systems.

The above cited works differ significantly in the nature of the datasets used to evaluate approaches to semantic segmentation and object detection. The most important distinguishing characteristics is the scale at which objects appear in images. For the datasets, where the objects appear in a table top setting, the challenges come from proper feature selection and efficient categorization in the presence of clutter. The datasets which contain more open scenes focus mostly on improvements in classification of non-object categories [10, 35] and large appearing objects in the dataset [16].

In our work we attempt to bridge methods which rely on semantic segmentation of open scenes with larger depth variation and table top object detection. Instead of striving to achieve high accuracy of object detection for objects appearing at smaller scales in open scenes, we propose a representation and efficient framework which can be used for priming general object search and semantic understanding of non-object components of the environment.

Another line of works attempts to bring in the semantic information to the existing systems and approaches for simultaneous localization and mapping. Authors in Häne et al. [17] have proposed a batch algorithm for jointly segmenting the scene and optimize the 3D reconstruction. Their system is a very good option for post-processing but hardly applicable in online robotic operation. An online SLAM system recognizing objects is proposed by [7], where in parallel to a monocular EKF SLAM thread there is a thread comparing visual features with the visual features of objects in the database. The recognition thread is always comparing against all the specific instances and as such is not scalable in the number of different objects and instances. The approach of [33] solve the SLAM problem aided by the a priori known of some objects. In real-time this SLAM system is able to recognize these objects and optimize their pose and the full map given the object models in the database. Although impressive in the results this approach does not scale well with the number of different objects in the database and it is not suitable neither for exploration of new environments nor discovery new object instances.

In our approach we also strive to develop framework which is efficient and amenable for real-time implementation. While efficiency has been considered extensively in the context of vision-based SLAM methods, the current techniques for semantic segmentation are computationally expensive and work predominantly in a off-line setting.
The complex issues of representation of semantic knowledge along with the spatial knowledge and the role of ontologies has been pointed out in the work of [29]. The authors focused more on exploiting the interplay between room locations and object identities, without considering more detailed semantic segmentation.

3 Semantic Segmentation

We formulate the semantic parsing in the framework of Conditional Random Fields (CRFs) with a tree graph structure encoding the pairwise relationships. We assume that an image and a 3D point cloud of the scene are available. Our approach starts by over-segmenting the image using the efficient *simple linear iterative clustering* (SLIC) algorithm [1]. Every superpixel in the image is interpreted as a cluster in the 3D point cloud for further computations. The 3D centroid of each cluster is used to compute a minimum spanning tree over Euclidean distances, defining the edges for the graphical model. The data and pairwise terms are determined using simple yet discriminative appearance and geometric cues for the classes that we are interested in. The learning and the final inference process is carried out over the graph in the CRFs framework. In the remainder of this section we detail the components of our approach and explain the intuition behind them.

3.1 Framework: Conditional Random Fields

Conditional random fields are probabilistic undirected graphical models first developed by [23] for labelling sequence data. CRFs are a case of Markov Random Fields, and thus satisfy the Markov properties. Instead of relying on Bayes' rule to estimate the distribution over hidden states $\mathbf{x}$ from observations $\mathbf{z}$, CRFs directly model $p(\mathbf{x}|\mathbf{z})$, the *conditional* distribution over the hidden variables given observations. Due to this structure, CRFs can handle arbitrary dependencies between the observations. This makes them substantially more flexible when using attributes that are too complex to model their probability distribution and the assumption of independence, as in a naive Bayes setting, is too strong [21].

The nodes in a CRF are denoted $\mathbf{x} = \langle \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n \rangle$, and the observations are denoted $\mathbf{z}$. In our framework the hidden states correspond to the m possible classes: $\mathbf{x}_i = \{ground, structure, furniture, props\}$.

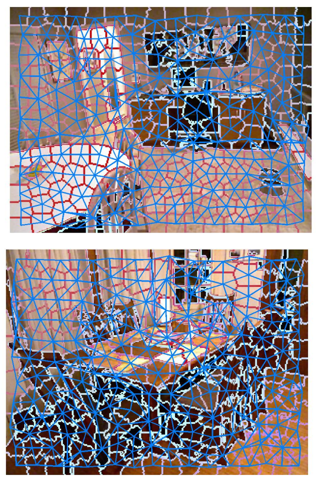
A CRF factorizes the conditional distribution into a product of *potentials*. We consider only the potentials for nodes $\phi(\mathbf{x}_i, \mathbf{z})$ (data-term) and edges $\psi(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z})$ (pairwise-term). This choice is commonly referred as pairwise CRFs. The potentials are functions that map variable configurations to non-negative numbers capturing the agreement among the involved variables: the larger a potential value, the more likely the configuration. Using the data and pairwise potentials, the conditional distribution over hidden states is written as:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \prod_{i \in \mathcal{N}} \phi(\mathbf{x}_i, \mathbf{z},) \prod_{i,j \in \mathcal{E}} \psi(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}) \tag{1}$$

where $Z(\mathbf{z})$ is the normalizing partition function, and $\langle \mathcal{N}, \mathcal{E} \rangle$ are the set of nodes and edges on the graph. The computation of this function can be exponential in the size of $\mathbf{x}$. Hence, exact inference is possible for a limited class of CRF models only, e.g. in tree-structured graphs. Potentials are described by log-linear combinations of *feature functions*, $\mathbf{f}$ and $\mathbf{g}$, i.e., the conditional distribution in Eq. 1 can be rewritten as:

$$p(\mathbf{x}|\mathbf{z}) = \frac{1}{Z(\mathbf{z})} \exp \left(\mathbf{w}_1 \sum_{i \in \mathcal{N}} \mathbf{f}(\mathbf{x}_i, \mathbf{z}) + \mathbf{w}_2 \sum_{i,j \in \mathcal{E}} \mathbf{g}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}) \right) \tag{2}$$

where $[\mathbf{w}_1, \mathbf{w}_2]$ are weights representing the importance of each term. CRFs learn these weights discriminatively by maximizing the conditional likelihood of labeled training data. We will describe every term of Eq. 2 in detail in 3.3. With this formulation we can obtain either the marginal distribution over the class of each variable $\mathbf{x}_i$ by solving Eq. 2, or the most likely classification of all the hidden variables $\mathbf{x}$. The latter can be formulated as the *maximum a posteriori* (MAP) problem, seeking the assignment of $\mathbf{x}$ for which $p(\mathbf{x}|\mathbf{z})$ is maximal.


(a) Dense graph on Image.

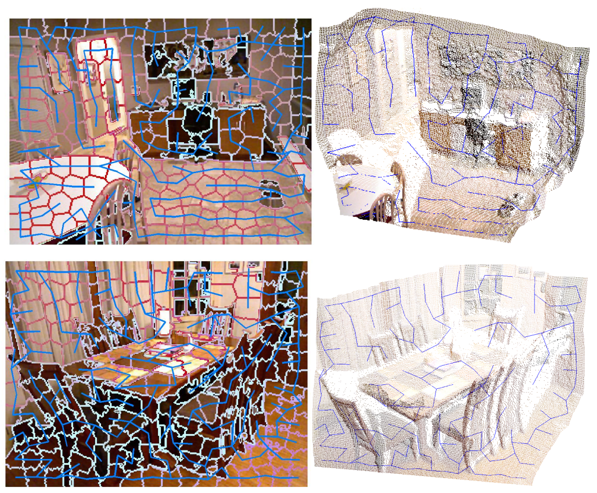

(b) MST over 3D.

Figure 1: Graph Structures (blue lines). On the left the most common graph structure used in the computer vision community. On the right the graph structure selected by us, a minimum spanning tree over 3D.

3.2 Minimum Spanning Tree over 3D distances

Instead of computing the graph at the pixel level, we over segment the image into superpixels. The CRF graph structure is typically determined by the neighbourhood relations between superpixels. This often yields dense graph structure, see Fig. 1(a), which is prone to over connect unrelated classes. We define the graph structure for the CRF as a minimum spanning tree over the Euclidean distances between 3D superpixel's centroids in a scene. By definition, the minimum spanning tree connects points that are close in the measurement space, highlighting intrinsic locality in the scene, see Fig. 1(b). This graph structure was already used in the context of object recognition [30] and place recognition [4]. In [4] was used for graph matching for place recognition, where the nodes correspond to a sparse set of detected keypoints and the idea was keep the structure of the graph between the matched nodes in a pair of scenes, while in the current work we handle a denser and structured 3D point cloud and we exploit that physical entities are continues by nature.

Given that our graph structure is a tree we can use the *belief propagation* (sum-product) algorithm to exactly infer the marginal distribution of each node, and the max-product algorithm to find the MAP assignment [21].

Default	Observation	Dim.	Comments
Image Features			
	LAB color	6	Mean and std
	y_l	1	Vertical pixel location
	H_s	1	Entropy of $hg_s(y)$
3D Features			
	(h_s, d_s)	2	Height and Depth
0	$(\mu_{\Delta d_s}, \sigma_{\Delta d_s})$	2	if $d_s < \frac{1}{\|N\|} \sum_{j \in N}(d_j)$ $\Delta d_s = \|d_s - d_{j \in N}\|$
0	$\frac{3\sigma_3}{\sigma_1+\sigma_2+\sigma_3}$	1	Superpixel Planarity
0	$1 - mean(\|\vec{n}_s \vec{n}_N\|)$	1	Neighbouring Planarity
0	$\|\vec{n}_s \vec{j}\|$	1	Superpixel Orientation

Table 1: Local observations

3.3 Feature description

With the graph structure defined for our CRF model, we have to define feature functions $\mathbf{f}(\mathbf{x}, \mathbf{z})$ and $\mathbf{g}(\mathbf{x}, \mathbf{z})$ in Eq. 2. The features for the data-term are computed as:

$$\mathbf{f}(\mathbf{x}_s, \mathbf{z}) = -\log P_s(\mathbf{x}_s|\mathbf{z}) \tag{3}$$

where the local prior $P_s(\mathbf{x}_s|\mathbf{z})$ is the output of a k-nearest neighbours (k-NN) classifier from a set of observations $\mathbf{z}$. We compute $P_s(\mathbf{x}_s|\mathbf{z})$ as proposed by [39] in Eq. 4, k is fixed to 10.

$$P_s(\mathbf{x}_s = l_j|\mathbf{z}) = \frac{1}{\sum_{j=1}^{m} \left(\frac{f(l_j)}{\overline{f}(l_j)} \frac{\overline{F}(l_j)}{F(l_j)} \right)} \frac{f(l_j)}{\overline{f}(l_j)} \frac{\overline{F}(l_j)}{F(l_j)} \tag{4}$$

where $f(l_j)$ (resp. $\overline{f}(l_j)$) is the number of neighbours to s with label l_j (resp. not l_j) in the kd-tree. And $F(l_j)$ (resp. $\overline{F}(l_j)$) is the counting of all the observations in the training data with label l_j (resp. not l_j). The observations $\mathbf{z}$ computed for every superpixel s capturing the appearance cues obtained from RGB image (*Image Features*) and the depth cues (*3D Features*) are summarized in Table 1 and described next.

3.3.1 Image Features

- The mean and standard deviation of each channel in the LAB-color space for the superpixel.

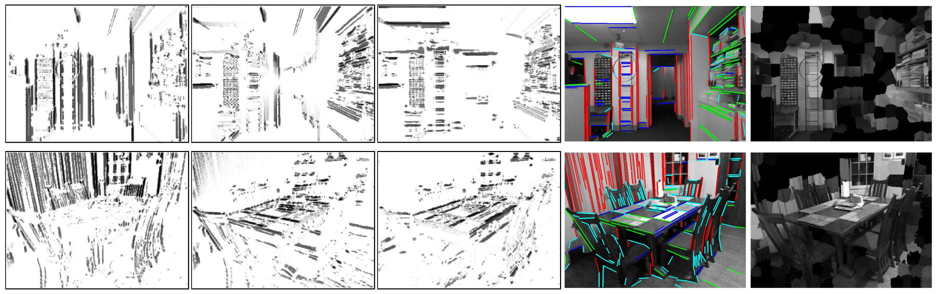


Figure 2: Gradient mixture model [9]. From left: three gradient probability images of being aligned to each of three vanishing points, a color-coded membership image with lines to one of three vanishing points (red, green, blue), not to be consistent with any (cyan). Rightmost, the image with bright proportional to the entropy of each superpixel.

- The vertical pixel coordinate for the superpixel's centroid.
- The entropy of the probability distribution for the superpixel boundaries belonging to the dominant vanishing points in the scene, see Fig. 2 rightmost.

The entropy feature expresses geometric consistency of a superpixel boundary with a particular vanishing direction. This feature is motivated by the observation that in indoors environments superpixel boundaries are often aligned with the vanishing directions. This observation has been also widely utilized in single-view reconstruction techniques, e.g. [18]. We employ 5-component gradient mixture model for the Manhattan world described in [9]. For each image pixel, the model provides the probability of the pixel lying on an edge, probability to pointing to each of the three vanishing points, see Fig. 2, and the probability of being noise. We take into account only those pixels having the probability of being on an edge higher than being noise. For each of those points a maximum over last 4 probabilities is chosen as a membership of the point to either being consistent with one of the 3 vanishing points or not to be consistent with any, see fourth column in Fig. 2. For a particular superpixel s, we compute a normalized histogram $hg_s(y)$ with four bins $y = \{1, 2, 3, 4\}$ from memberships of all pixels lying along the superpixel boundary. In order to differentiate between clutter or small objects and structural classes we use the entropy of this normalized histogram, Eq. 5.

$$H_s = -\sum_{j=1}^{4} hg_s(y=j)\log\left(hg_s(y=j)\right) \tag{5}$$

3.3.2 3D Features

For the 3D point cloud computed with the depth information we use cues from the 3D position and planarity, for the superpixel itself and for the superpixel with respect to its neighbourhood. The cues are:

- The depth (d_s) and height (h_s) for the superpixel's centroid.
- The mean and standard deviation of the absolute difference between the depth d_s and the neighbourhood's depths: $\|d_s - d_{j\in N}\|$. These are only computed if $d_s < \frac{1}{\|N\|}\sum_{j\in N}(d_j)$, with this condition we encode the *in-front-of* property.
- The superpixel planarity encoded by the curvature of a superpixels' point cloud [20] using the SVD and sort the singular values such that $\sigma_1 > \sigma_2 > \sigma_3$.
- The neighbourhood planarity computed as one minus the mean of the dot product between the normal to the plane against to the neighbourhood normals [43].
- The superpixel orientation, taken as the projection of the superpixel's normal on the horizontal plane [42].

The superpixel neighbourhood N refers to all the superpixels in contact with superpixel s in the image. In Table 1 we also show the default values and the dimensionality of these observations. As a result we compute for each superpixel 15 dimensional feature vector with 8 elements from *Image features* and 7 from *3D features.*

Pairwise term

The pairwise feature $\mathbf{g}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z})$ is computed for every edge in the graph as:

$$\mathbf{g}(\mathbf{x}_i, \mathbf{x}_j, \mathbf{z}) = \begin{cases} 1 - \exp\left(-\|c_i - c_j\|_2\right) & \rightarrow \quad l_i = l_j \\ \exp\left(-\|c_i - c_j\|_2\right) & \rightarrow \quad l_i \neq l_j \end{cases} \tag{6}$$

where $\|c_i - c_j\|_2$ is the L2-Norm of the difference between the mean colors of two superpixels in the LAB-color space and l is the class label.

Given the graph structure and the features outlined above, we proceed with the description of the learning and inference stage and the performance evaluation of the final semantic segmentation approach in Section 5. An example of the probability distribution outcome from our approach is shown in Fig. 3.

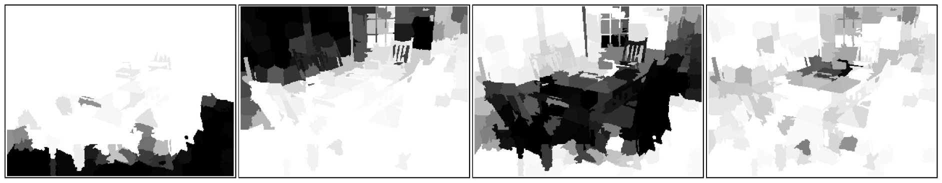


Figure 3: Exact marginals obtained with the sum-product algorithm. From left: probability to be *ground*, *structure*, *furniture* and *props*. Darker means higher probability.

4 Priming Object Detection

In this section we show how the marginals from the semantic segmentation approach can be used in the task of localizing objects in the scene. We propose a simple detector for generic objects and show how to combine it with the semantic segmentation results to improve the detection rate and computational cost. Instead of training an object detector separately for each object category we consider single category of generic objects whose extent can be well approximated by a bounding box. Each bounding box is then characterized by a boundary feature characterizing the orientation of the boundary close to the bounding box border. The feature is motivated by previous works on saliency and detection of generic objects [2] and works for objects of relatively simple shape, where majority of the true object boundary is close to the bounding box boundary. In the next section we describe in more detail the boundary feature, the generic object detector and the method for proposing and scoring of candidate bounding boxes to generate object hypotheses.

4.1 Features and Classifier

We compute the gradient orientations in four blocks depicted in Fig. 4 right, obtained by shrinking and enlarging the bounding box by 5 pixels each, to

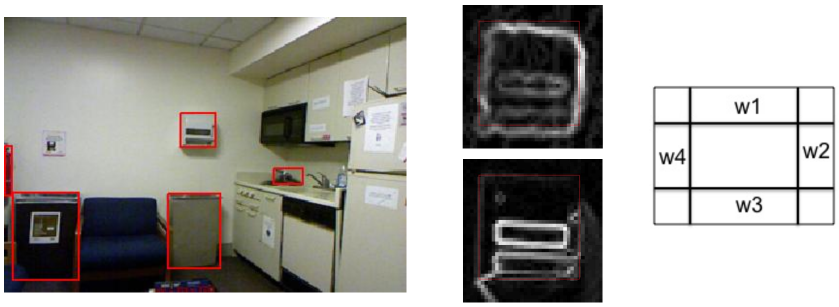


Figure 4: Examples of objects and their bounding boxes and close-up of the orientation energy for both intensity and depth channel. Center: orientation energy for paper towel dispenser, where the depth gradients and image gradients complement each other well. Right: window blocks in the vicinity of the boundary where the histograms are computed.

capture the variation around the actual boundary. In each block we quantize the orientations from ($0°$- $360°$) into 9 orientation bins. This is done both for intensity and depth channel yielding a $2 \times 4 \times 9 = 72$-dimensional feature.

At this point we can introduce the first modification to include the semantic segmentation information into the feature vector. We compute the probability distribution of the bounding box over the four classes (*ground, structure, furniture, props*), by averaging and re-normalizing the pixel's probabilities inside the window. Now, the feature vector is augmented with the four values resulting in a 76-dimensional feature vector.

We train an SVM classifier with histogram intersection kernel [26] over the mentioned feature vector. The positive examples come from the ground truth bounding boxes and the negative ones are picked from randomly the generated windows using the proposal of [40]. In the window generation method of [40] the image is first over segmented in superpixels, which are then merged using the criteria of similarity over the color space and superpixels' sizes building a fine to coarse hierarchical over-segmentation. The bounding boxes of each superpixel at different levels in the hierarchy are possible candidates.

The ground truth is constrained to those objects with sides smaller that $50cm$ and greater that $5cm$. We obtain the metric dimensions of the bounding boxes directly from the depth map. The negative examples are constrained by the same rule.

4.2 Object Search Strategy

At the test time, instead of generating candidate windows with different predefined aspect ratios and scales, we use again the strategy of [40] with the same constraints on the size than before.

Now, we introduce the second modification which is sampling the candidate windows with the probability of the bounding box belonging to *furniture* or *props*. The idea behind this sampling strategy is that if our semantic segmentation is confident that one candidate box covers the wall or the floor we don't need to evaluate the classifier on it or compute the features for that window. On the other hand, it is highly probable the objects we are looking for lie over *furniture* pieces or belong to the *props* semantic class.

5 Experiments

In our experiments we use the NYU V2 RGB-D dataset [35], which contains 1449 labeled frames. The labeling spans over 894 different classes produced using Amazon Mechanical Turk. The authors of the dataset also provide a train and test splits and a mapping from the 894 categories to 4 classes: *ground*, *structure*, *furniture* and *props*, as was used in [35]. We take 795 training frames for building the kd-tree and for CRF parameter learning, and the remaining 654 frames for testing and quantitative comparison against state of the art methods in RGB-D semantic segmentation.[2]

We obtain superpixel segmentation using SLIC implementation from the VLFeat library of [41], followed by the computation of the features described in Table 1. With the computed features in the training set we build a kd-tree using the implementation of [3] with the default parameters. Then for every superpixel in the training set we obtain the k-NN classification for the training data using Eq. 4 with the $k = 10$ nearest neighbours.

The minimum weight spanning tree (MST) is computed from 3D centroids of all the superpixels. Now, using the MST graph, the output of the k-NN classifier in Eq. 3 and the pairwise potentials, Eq. 6, we learn the parameters in the CRF setting. For the learning, inference and decoding with CRFs we use the Matlab code for undirected graphical models (UGM).[3]

[2]The input images are cropped, the external blank boundary is removed, and then re-scaled to the half yielding images of 313x234 in resolution.

[3]Code made available by Mark Schmidt at `http://www.di.ens.fr/~mschmidt/Software/UGM.html`

Recall	ground	furniture	props	structure
ground	**88.5**	8.8	1.3	1.4
furniture	6.6	**63.5**	12.5	17.4
props	8.1	**32.9**	31.8	27.2
structure	1.0	14.0	6.3	**78.7**

Table 2: Confusion matrix for the pixel-wise accuracy in %.

	ground	furniture	props	structure	Average	Global
Ours	88.5	63.5	31.8	78.7	**65.6**	**67.3**
only Im Feat.	68.5	46.1	30.6	75.3	55.1	57.9
only 3D Feat.	88.9	65.4	18.2	80.3	63.2	66.4
with D_s & no H_s	**90.3**	64.3	25.3	78.0	64.5	66.6
with D_s	89.4	61.9	30.6	78.3	65.1	66.5
Ours & N-Im graph	88.9	64.9	22.4	81.4	64.4	**67.3**
Ours & Delaunay graph	89.1	63.6	19.6	81.4	63.5	66.5
k-NN, Eq. 3	88.5	50.2	**43.7**	72.7	63.8	62.6
Silberman *et al.* [35]	68	**70**	42	59	59.6	58.6
Couprie *et al.* [10]	87.3	45.3	35.5	**86.1**	63.5	64.5

Table 3: Pixel-wise percentage recall accuracy.

5.1 Results on Semantic Segmentation

At testing time, to obtain the most likely label assignment for the superpixels we solve the MAP problem over the CRFs. This problem does not require any threshold selection and all the parameters are computed/learned from the data. The inference results give us the labeling assignments over superpixels, we transfer those to every pixel in the superpixel to compute the pixel-wise accuracy of semantic labeling. In Fig. 5 we show several examples of the output of our approach.

Table 2 shows the confusion matrix normalized by rows (recall on the diagonal). The *props* class is the most frequently confused. Props class is defined in [35] as “small objects that can be easily carried”. We can observe in the results, Fig. 5, several planar objects (posters, carpets, placemats) are labeled as *structure* or *furniture*. Another source of confusion is due to the quality of the ground truth, where we found several ambiguities in the *structure* and *furniture* categories, where kitchen tables, stoves, dishwashers and cabinets were labeled as the two interchangeably.

In first row of Table 3 we show the pixel-wise recall accuracy along with the average and global accuracy for our approach: CRF-MST and k-NN+SLIC with Image and 3D features. To study the importance of image

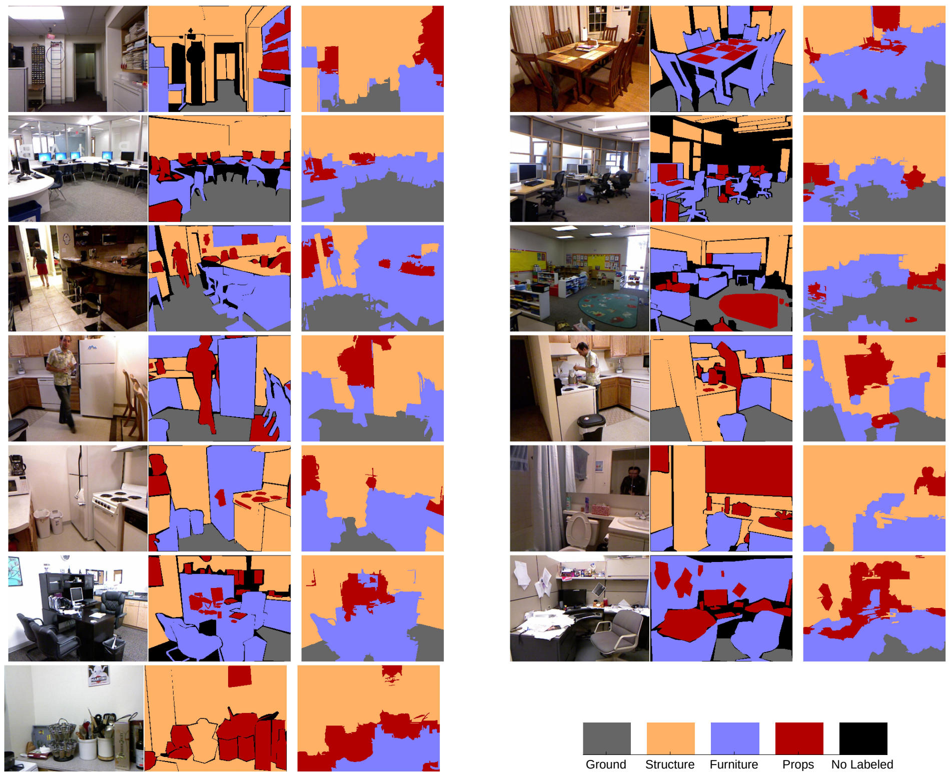


Figure 5: Original images, ground truth labeling and MAP result from our approach in the NYU dataset.

features vs 3D features, we remove one set at a time keeping the rest of the system intact. The rows *Image Features* and *3D Features* show the corresponding performance when using only one type of features. Using only 3D features we can see better results for three out of four classes at the cost of very poor accuracy in *props* and smaller average and global accuracy. Our full system, with both sets of features obtains the best trade off between for all performance measures. In Table 3 we also show the result by only using the data-term, k-NN row, using the image and 3D information. It is clear that the MST and the CRF framework improve the general performance.

Manhattan world assumption: We implicitly use the Manhattan World assumption when using our entropy feature H_s, Eq. 5. This feature helps the system to identify clutter or small objects that are not aligned with the dominant directions in a scene. H_s is an image-only feature, but is also possible compute dominant plane normals from depth data aligned with the Manhattan World [35]. Following [37] we find the dominant orthogonal directions of the world frame then we normalize the absolute values of the superpixel's normals projected on this world frame to compute their entropy, Eq. 7, similarly as in the case of H_s.

$$D_s = -\sum_{j=1}^{3} m_{sj} \log(m_{sj}), \quad \text{with} \quad m_{sj} = \frac{|\vec{n}_{sj}|}{\sum_{k=1}^{3} |\vec{n}_{s,k}|} \tag{7}$$

We have evaluate our system removing H_s from our features and adding D_s instead, and also using both, i.e. our features and D_s, (with D_s & no H_s) and (with D_s) in Table 3, respectively. In the first case the accuracy for *ground* is improved but for *props* is hurt. The second case, the performance for *props* improves again but in general is not as good as our initial proposal.

Graph structure: In order to compare the advantage of the MST as graph structure we have also included the results of our system with other two graphs, first the classical choice of connecting each superpixel with its neighbours in the image (N-Im), see Fig. 1(a) and the second, by connecting the nodes through the Delaunay triangulation over their 3D positions. Given that both graphs contain loops we have used *loopy belief propagation* (LBP) for inference. In Table 3 we can see that their average and global accuracies are still competitive but at expense of poor accuracy in *props*. They also incur computationally most expensive inference and no warranty of convergence. For N-Im graph LBP took, with a maximum of 100 iterations, in average 0.96s per frame and did not converge on 19 out of 654 frames. If we reduce the maximum iterations to 20 we reduce the cost to 0.83s per frame but in 455 out of 654 frames the inference does not converge. For Delaunay graph LBP took, with a maximum of 100 iterations, in average 2.8s per frame and did not converge on 15 out of 654 frames. Our MST graph structure allows exact inference using belief propagation in only 42ms in Matlab.

Comparisons with the state of the art: We also compare against the full method proposed by [10]. That method uses the four channels (RGB and depth) in a multi-scale convolutional network to learn the features, and a 2-layer multi-perceptron as classifier, followed by the aggregation of the final assignments over superpixels computed by [15]. We can observe that our approach is competitive or better for all the classes, with better average and global accuracy. Their system takes 2 days to train but is very efficient in the

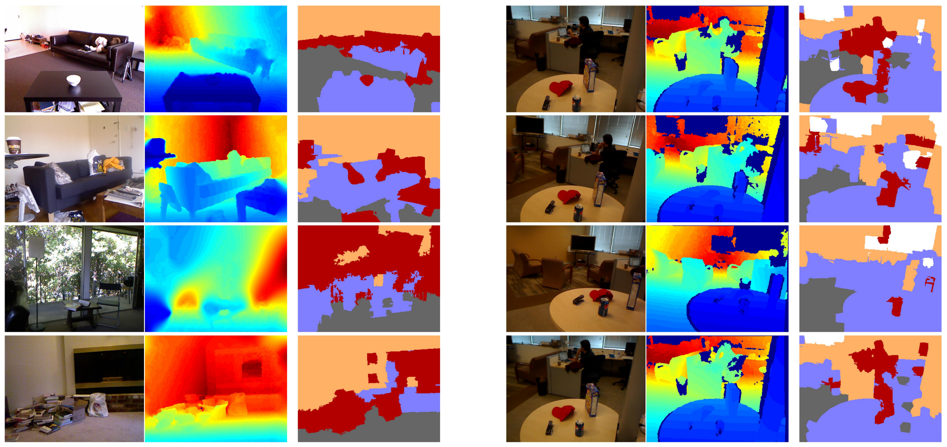


Figure 6: Original images, depth and MAP result from our approach in B3DO [19] (left) and UWobj [24] datasets (right).

testing stage, spending 0.7 seconds per frame to perform the segmentation using a parallel implementation for the convolution stage [14].

In comparison to the original work of authors who released this dataset [35], as shown in Table 3, their system still obtains the best accuracy in the *furniture* and *props* classes. They use a feature vector of 1128 elements to compute the data term with a logistic regression classifier. The cues come from color (36), shape (1086) and scene (6) information (supplementary material in [35]). Our representation and features are notably simpler enabling more efficient feature computation and inference while still achieving comparable performance.

5.1.1 Independent Datasets

We evaluate our semantic segmentation on two independent datasets; Berkeley 3-D Object Dataset, B3DO [19], and the RGB-D Object Dataset from University of Washington, UWobj [24]. B3DO with 849 frames has more variability in the indoors scenes and spans from offices to house rooms. The smoothed depth information is obtained by another in-painting technique, different from that used by NYU V2. UWobj is a dataset focuses on close range object instance detection and object categorization and contains mostly table top scenes from the lab with large extent objects. This dataset is released without in-painting in the depth maps. From both datasets we evaluate our semantic segmentation approach on the scenes that correspond

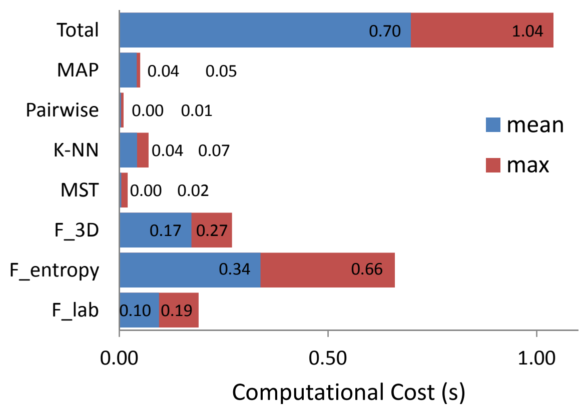


Figure 7: Detailed mean and maximum computational timing of 654 NYU V2 frames in testing excluding SLIC over-segmentation cost.

	SLIC	F-lab	F-entr.	F-3D	MST	k-NN	Pairw.	MAP	Total
Matlab/mex	611	95	338	173	4	42	4	42	1309
C++/GPU	10.58	3.50	28.57	19.62	0.34	77.97	0.39	0.99	141.0

Table 4: Mean computational timing for the semantic segmentation system in milliseconds in the NYU-V2 test set.

to open space, instead to table top scenes; as our semantic segmentation focuses on semantic classes relevant during navigation or exploration rather than for manipulation. We consider an open space any scene where the median of the depth map is greater than 2.5 meters leaving us with 182 frames from B3DO and 45 frames from UWobj. Some result are shown in Fig. 6, we can see a similar performance that for the NYU V2 dataset, even with the differences in depth pre-processing, type of scenes and poses of the sensor, which show that our proposal is general and powerful tool for semantic segmentation.

5.1.2 Timing

The experiments were carried out with a research implementation of our approach in Matlab. The computational cost, for NYU V2 in testing, is detailed in Fig. 7, excluding the superpixel over-segmentation. The system runs in average at 1 fps in a single-thread of a 3.4 GHz IntelCore i7-2600 CPU M350 and 7.8 GB of RAM. Including the SLIC over-segmentation is still able to run at 0.5 fps. For the whole system, the mean and the maximum computational times are 0.7s and 1.0s, respectively. The average cost to obtain the SLIC superpixels is 611ms, although a C++ implementation would take half of that time as reported by [1] In the feature computation the main bottleneck is the computation of the gradient mixture model [9] for vanishing directions. The training stage takes, less than 2 seconds to build the kd-tree, and 180 seconds for the CRF parameter learning. In total including the feature computation, the computational cost for all the training data is less than 20 minutes.

An initial implementation of semantic segmentation approach in C++ and using the GPU implementation of SLIC (gSLIC) of Ren and Reid [31] is closer to a real time operation working around 5 fps. Table 4 shows the mean timing for both versions, Matlab/mex and C++/GPU. A more efficient system is possible by implementing other stages, e.g. kNN on the GPU.

5.2 Object Localization

We train the object detector as described in Section 4 on the training set of the NYU V2 dataset and evaluate it on the test set from the same dataset and, on B3DO and UWobj datasets. Fig. 8 shows some example results from testing in all the datasets.

We use the intersection over union measure greater than 0.5 to accept a detection [13]. We also use the concept of ground truth bounding boxes to ignore (BB_{GT-ig}) as proposed in [11]. Any proposal matched to a BB_{GT-ig} does not count as good detection but neither as false positive. This solves issues for example when we detect individual instances of objects and the BB_{GT-ig} is covering group of objects (book vs books).

5.2.1 Performance Evaluation

In Fig. 9 we show the performance of our approach for priming object detection in terms of F1-score vs detector SVM-score, the SVM score is calibrated to span from 0 to 1. In order to show the contribution of using the semantic segmentation information we present three versions for our system, as described in Section 4.1 and 4.2.

1. Ours [I,D], No sampling: This is the basic system, descriptor with intensity and depth channels without using the results from the semantic segmentation in any stage.

2. Ours [I,D,P], No sampling: Here we augment the descriptor with the probabilities from the semantic segmentation.

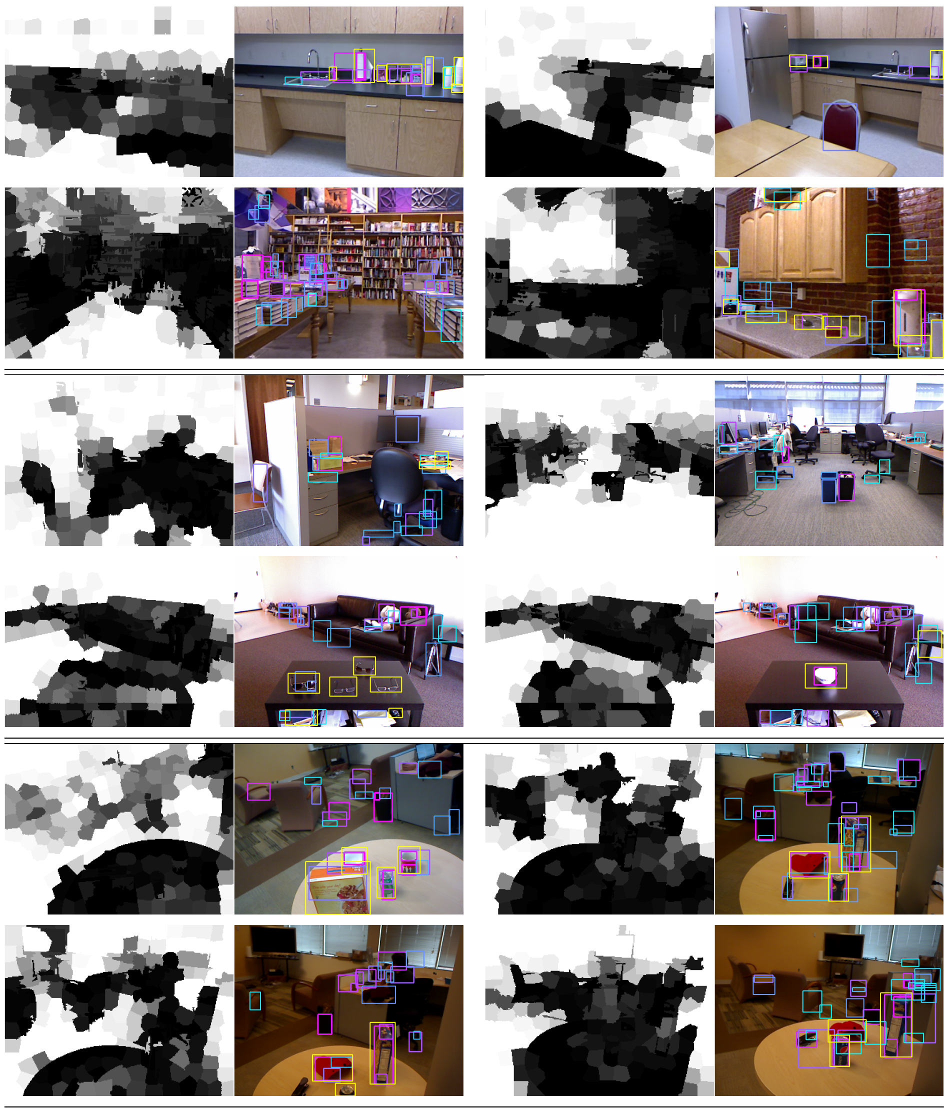
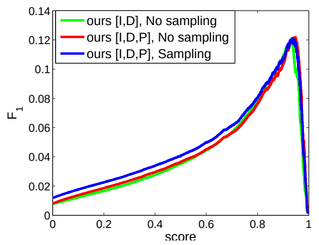
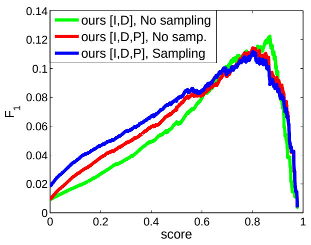


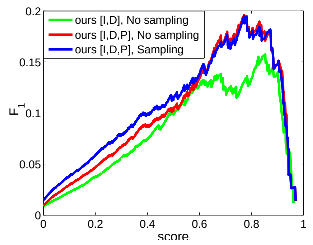
Figure 8: Priming object detection. The probability map, $p(furniture) + p(props)$, for the sampling of candidates is on the left of every pair, darker means higher probability. On the right the image with ground truth bounding boxes (yellow) and detections. The color of the detection windows goes from cyan to pink as the SVM score increases. We show four examples for every dataset, NYU V2 (top), B3DO (middle), and UWobj (bottom).



(a) NYU-v2



(b) B3DO



(c) UWobj

Figure 9: Comparison of F_1-score vs detector SVM-score when using the feature vector composed only by the intensity and depth information (ours [I,D]), when augmented by the semantic segmentation probability (ours [I,D,P]) and when pruning the candidate windows with the sampling proposed.

3. Ours [I,D,P], Sampling: Here we sample the bounding boxes proposals by $p(furniture) + p(props)$ and augment the descriptor with the probabilities from the semantic segmentation.

The F_1-score curve shows how we improve the baseline system, first by augmenting the descriptor with the probabilities from the semantic segmentation and then by sampling using the $p(furniture) + p(props)$. When setting a robotic system for real operation, we may determine a threshold for the SVM-score, Fig. 9 depicts that after choose a threshold on one of the datasets the rank of the three versions is maintained to new environments. The major benefit in the sampling is reflected in the computational cost given the number of discarded candidates before the feature computation. For example in Fig. 10 left, without sampling, the cost for feature computation and SVM evaluation is 760ms and 60ms, respectively. Instead, with the sampling, Fig. 10 right, the cost is 440ms and 20ms. In Fig. 11 we show the frequency of speed-up for all the datasets. In the worst case there is almost no gain in efficiency in cluttered scenes, and in other scenes (e.g. corridors) we can get speed-ups of up to 4x.

The low F_1-scores for the object detector are due to the high proportion of false alarms. We can see in Fig. 8 that we obtain several false alarms, but a lot of them are because the ground truth bounding boxes are incomplete and many objects which we successfully detect are not labelled at all. This problem is easy to find in the research community as the labeling is generating only for the specific objects of interest in each paper. However, in real life applications a robot can find a possible unlimited number of different

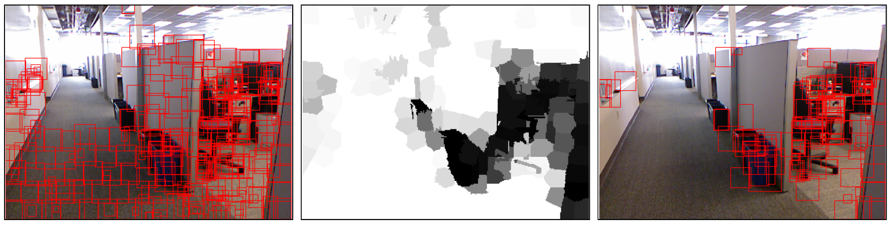


Figure 10: On the left all the candidates, on the right the candidates after pruning by the probability of the window to be furniture or props shown in the middle (the darker the higher the probability).

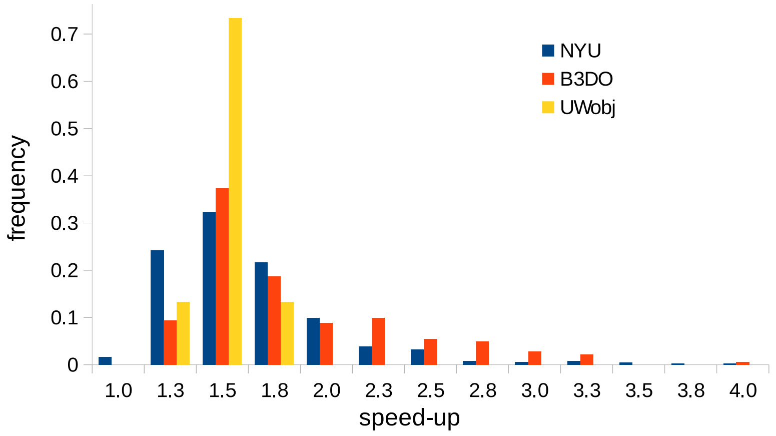


Figure 11: Frequency of speed-up for all datasets when using sampling driven by $p(props) + p(furniture)$.

objects and instances.

5.2.2 Comparisons

As comparison with the state of the art object detectors we include the results from the very recent BInarized Normed Gradients (BING) objectness estimation [6]. The source code is provided by the authors with a training on PASCAL-VOC 2007 dataset. In Figs. 12, 13 and 14, we depict the performance of BING detector and of our approach in terms of detection rate (DR) vs false positives per image (fppi) and precision - recall (PR) curves. The DR-fppi and PR curves show that our object proposal system is better than the only-image state of the art. In the UWobj dataset BING is better than ours in one portion of the curve, if more than 50 fppi are allowed in a

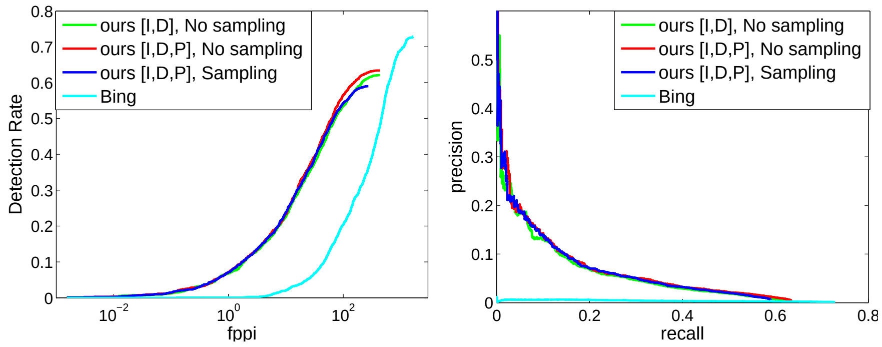


Figure 12: NYU-V2 dataset. Left: Detection rate vs false positive per image. Right: Precision vs Recall.

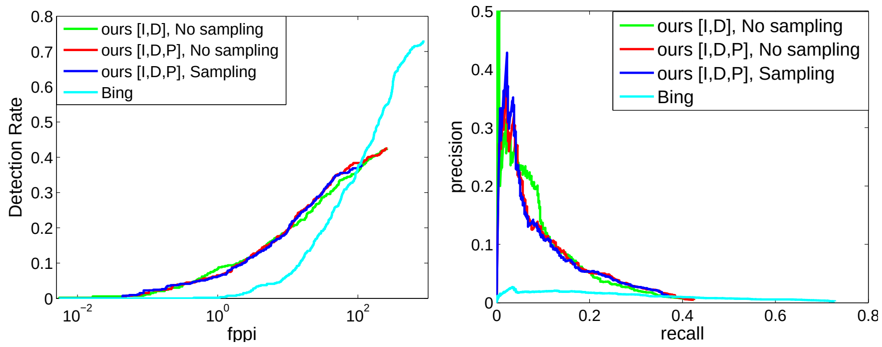


Figure 13: B3DO dataset. Left: Detection rate vs false positive per image. Right: Precision vs Recall.

real system, and the reason for this is that this dataset is the most structured one among the three datasets, with objects in table top setting and in the center of the image.

6 Discussion

We have shown a basic implementation with real time capabilities that effectively uses appearance and 3D cues to generate evidence about the structure of the scene, while achieving better average and global accuracy of semantic labeling compared to the state of the art. Note that the accuracy reported by [10] was computed after training the system for 849 different classes and then, the results were clustered into the four categories, indicating that their

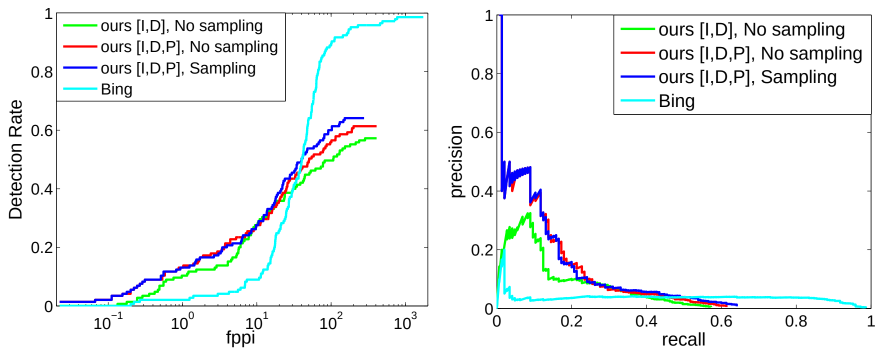


Figure 14: UWobj dataset. Left: Detection rate vs false positive per image. Right: Precision vs Recall.

approach and features are less robust to the high intra class variability in the four class problem.

We have shown that our graph structure induced by the MST over 3D does not sacrifice the labeling accuracy, and keeps the intra-class components coherently connected. Furthermore, by this selection we gain an exact and efficient inference. The computational complexity for the inference is $\mathcal{O}(nm^2)$, where n is the number of nodes in the graph, and m the number of classes. In that sense our approach is suitable for segmentation problems with small number of classes. While the computational cost could violate a real-time constraint for problems with large number of classes, we believe that a reliable semantic segmentation system should follow a coarse to fine strategy, where the labels for specific objects should be sought if necessary, and not classify large number of different objects classes at once.

An interesting discussion is related to the choice of basic representation and the features. Authors in [35] use over 1000 dimensional feature vectors, concatenating many engineered features, most of them developed previously for diverse but related tasks. On the other hand, authors in [14] propose to avoid the feature engineering and learn the features from the data using convolutional networks. They are still using a 768 dimensional feature vector for RGB channels, and the feature vectors have over 1000 dimensions when including the depth channel [10]. We can see that in both approaches, feature engineering vs feature learning, high dimensional feature space is constructed in which the different classes are more easily separable. In this work we propose and demonstrate, that a fewer (12 dimensional) but meaningful features are sufficient to obtain better semantic segmentation in RGB-D scenes. In addition to the choice of features there are at least two more reasons for this improvement. First, selecting a well-performing local classifier to handle high intra class variability: k-NN in our case vs logistic regression [35], vs 2-layer multi-perceptron [10]. Second, defining a more natural connections between neighbours given the type of data: minimum weighted spanning tree over 3D distances in our system, vs connecting with all the neighbours in the image [35], vs no connections at all between superpixels [10]. Authors [14] shows a similar comparison.

We have also shown how to integrate the results of semantic segmentation with object detection, proposing a simple detector of generic objects who's shape can be well approximated by a bounding box. The performance of the object detection in terms of precision recall is not on par with other object detection results where large objects and more discriminative features are considered [32]. This is partly due to the lack of correctly labeled objects in the dataset (many small objects are part of the background class) and simplistic boundary feature descriptor. More sophisticated descriptors (e.g. [38]) and more accurate labeling can significantly improve the proposed approach.

We plan to explore strategies for obtaining the observations in a multi-scale way to improve the performance, this is inspired on the boosting achieved by [14] when compare the multi-scale vs one-scale convolutional networks.

We expect a C++/GPU implementation for our object detection to work at more tha 10fps, a similar gain as for our semantic segmentation. Note that BING detector [6] works with image information only, and it is still possible to get a better performance by adapting it to take into account the depth channel, and maybe the semantic class probabilities, although a reduction in its efficiency is expected.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11): 2274 –2282, nov. 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.120.

[2] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 73 –80, june 2010. doi: 10.1109/CVPR.2010.5540226.

[3] A. M. Buchanan and A. W. Fitzgibbon. Interactive feature tracking

using K-D trees and dynamic programming. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 626–633, 2006.

[4] C. Cadena, D. Gálvez-López, J.D. Tardós, and J. Neira. Robust place recognition with stereo sequences. *IEEE Transaction on Robotics*, 28 (4):871 –885, 2012. ISSN 1552-3098. doi: DOI:10.1109/TRO.2012.2189497.

[5] J. Carreira and C. Sminchisescu. Constrained parametric min-cuts for automatic object segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3241–3248. IEEE, 2010.

[6] M.M. Cheng, Z. Zhang, W.Y. Lin, and P.H.S. Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *IEEE CVPR*, 2014.

[7] J. Civera, D. Galvez-Lopez, L. Riazuelo, J.D. Tardos, and J. M M Montiel. Towards semantic SLAM using a monocular camera. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1277–1284, 2011. doi: 10.1109/IROS.2011.6094648.

[8] A. Collet, S.S. Srinivasa, and M. Hebert. Structure discovery in multimodal data: a region-based approach. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2011.

[9] J.M. Coughlan and A. Yuille. Manhattan world: Orientation and outlier detection by bayesian inference. *Neural Computation*, 15(5):1063–1088, 2003.

[10] C. Couprie, C. Farabet, L. Najman, and Y. LeCun. Indoor semantic segmentation using depth information. *CoRR*, abs/1301.3572, 2013.

[11] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *PAMI*, 34, 2012.

[12] I. Endres and D. Hoiem. Category independent object proposals. In *Computer Vision–ECCV 2010*, pages 575–588. Springer, 2010.

[13] M. Everingham, L. Van Gool, C.KI Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[14] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PP(99):1–1, 2013. ISSN 0162-8828. doi: 10.1109/TPAMI.2012.231.

[15] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59:167–181, 2004. ISSN 0920-5691. doi: 10.1023/B:VISI.0000022288.19776.77.

[16] S. Gupta, P. Arbeláez, and J. Malik. Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, 2013.

[17] C. Häne, C. Zach, A. Cohen, R. Angst, and M. Pollefeys. Joint 3d scene reconstruction and class segmentation. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, 2013.

[18] D. Hoiem, A. Efros, and M. Hebert. Recovering surface layout from an image. *International Journal of Computer Vision*, 75:151–172, 2007. ISSN 0920-5691. doi: 10.1007/s11263-006-0031-y.

[19] A. Janoch, S. Karayev, Y. Jia, J.T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *ICCV Workshop on Consumer Depth Cameras for Computer Vision*, 2011.

[20] K. Klasing. *Aspects of 3D Perception, Abstraction, and Interpretation in Autonomous Mobile Robotics*. PhD thesis, Technical Univeristy of Munich, Munich, Germany, April 2010.

[21] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[22] H.S. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *In 25th annual conference on neural information processing systems*, 2011.

[23] J. Lafferty, A. McCallum, and F. Pereira. Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[24] K. Lai, B. Liefeng, R. Xiaofeng, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824, 2011. doi: 10.1109/ICRA.2011.5980382.

[25] K. Lai, L. Bo, X. Ren, and D. Fox. Detection-based object labeling in 3d scenes. In *IEEE International Conference on on Robotics and Automation*, 2012.

[26] S. Maji, A.C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, 2008. doi: 10.1109/CVPR.2008.4587630.

[27] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color and texture cue. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):530–549, 2004.

[28] A.K. Mishra, A. Srivastav, and Y. Aloimonos. Segmenting simple objects using rgb-d. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2012.

[29] A. Pronobis, K. Sjöö, A. Aydemir, A.N. Bishop, and P. Jensfelt. Representing spatial knowledge in mobile cognitive systems. In *11th International Conference on Intelligent Autonomous Systems (IAS-11)*, 2010.

[30] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell. Hidden conditional random fields. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(10):1848–1852, Oct. 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1124.

[31] C.Y. Ren and I. Reid. gSLIC: a real-time implementation of SLIC superpixel segmentation. Technical report, University of Oxford, Department of Engineering, 2011.

[32] X. Ren, L. Bo, and D. Fox. Rgb-(d) scene labeling: Features and algorithms. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, 2012.

[33] R. Salas-Moreno, R. Newcombe, H. Strasdat, P. Kelly, and A. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Ob-

jects. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, 2013.

[34] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *Proceedings of the International Conference on Computer Vision - Workshop on 3D Representation and Recognition*, 2011.

[35] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*, 2012.

[36] Y. Sun, L. Bo, and D. Fox. Attribute based object identification. In *IEEE International Conference on on Robotics and Automation*, 2013.

[37] C. Taylor and A. Cowley. Parsing indoor scenes using rgb-d imagery. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[38] C.L. Teo, A. Myers, C. Fermller, and Y. Aloimonos. Embedding high-level information into low level vision: Efficient object search in clutter. In *IEEE International Conference on on Robotics and Automation*, 2013.

[39] J. Tighe and S. Lazebnik. Superparsing: Scalable nonparametric image parsing with superpixels. In *Computer Vision - ECCV 2010*. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15554-3. doi: 10.1007/978-3-642-15555-0_26.

[40] K.E.A. Van de Sande, J.R.R. Uijlings, T. Gevers, and A.W.M. Smeulders. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1879–1886, 2011. doi: 10.1109/ICCV.2011.6126456.

[41] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. `http://www.vlfeat.org/`, 2008.

[42] J. Xiao and L. Quan. Multiple view semantic segmentation for street view images. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 686 –693, oct. 2009. doi: 10.1109/ICCV.2009.5459249.

[43] C. Zhang, L. Wang, and R. Yang. Semantic segmentation of urban scenes using dense depth maps. In *Computer Vision - ECCV 2010*. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15560-4. doi: 10.1007/978-3-642-15561-1_51.