2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Congress Center Hamburg
Sept 28 - Oct 2, 2015. Hamburg, Germany

3D Selective Search for Obtaining Object Candidates

Asako Kanezaki[1] and Tatsuya Harada[1]

Abstract— **We propose a new method for obtaining object candidates in 3D space. Our method requires no learning, has no limitation of object properties such as compactness or symmetry, and therefore produces object candidates using a completely general approach. This method is a simple combination of Selective Search, which is a non-learning-based objectness detector working in 2D images, and a supervoxel segmentation method, which works with 3D point clouds. We made a small but non-trivial modification to supervoxel segmentation; it brings better "seeding" for supervoxels, which produces more proper object candidates as a result. Our experiments using a couple of publicly available RGB-D datasets demonstrated that our method outperformed state-of-the-art methods of generating object proposals in 2D images.**

I. INTRODUCTION

Our daily environment is filled with various objects: a broad array of objects from small items such as pencils and cups to large furniture such as beds and shelves. Non-rigid objects such as clothes and towels are abundant in indoor scenes. It is crucially important for intelligent autonomous systems to recognize such "general" objects of great diversity in the real environment. In particular for autonomous mobile robots, real-time vision systems to detect objects are necessary to avoid collision with obstacles, and for finding informative objects to take care of.

Image recognition technology has advanced dramatically because of large-scale web images. In the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014, the top five classification errors of 1,000 object categories achieved less than 7% [1]. Krizhevsky et al. [2] sparked a revolution of deep learning at ILSVRC 2011. Their deep convolutional neural network (DCNN) responses from the middle layers are now used widely as discriminative image descriptors. R-CNN [3] is a state-of-the-art object detector using the DCNN features described above. It is noteworthy that object detection (or localization) is more important than object classification for embodied systems such as autonomous mobile robots. Conventional object detection methods such as Deformable Part Models [4] used HOG descriptors [5] and sliding windows, which is an exhaustive search in multiple image scales. In the sliding window approach, approximately 500,000 windows per image are examined, which makes it inapplicable to real-time systems with costly image features such as DCNN features because it takes 4 ms using GPU to extract DCNN features from one window. (Therefore, 2,000 s would be needed for 500,000 windows.)

To ensure the efficiency of object detection, many studies have specifically examined the selection of regions for object candidates. Selective Search [6] is the de facto standard method of this topic, which is actually used in R-CNN [3] described above. This method first extracts superpixels from an image; then it combines neighboring superpixels in order of decreasing similarity of the two superpixels. This process is repeated until it obtains the whole image as a largest segment. Finally, it outputs the bounding boxes of all the segments obtained in the hierarchical grouping process. Unlike other state-of-the-art methods on object region proposal [7], [8], Selective Search is a completely general approach that involves no learning or assumption of objects (e.g., compactness or symmetry). The window number can be reduced to approximately 1,000 for an image using Selective Search. Even fewer windows are preferable for real-time systems such as autonomous mobile robots. Therefore, we consider the accuracy of windows judged in an early stage (which is for example top 100 windows) as important.

As described in this paper, we propose 3D Selective Search, which uses 3D structures of point clouds captured by consumer RGB-D cameras such as Kinect sensors. Strictly speaking, we substitute supervoxels estimated in 3D space [9] for superpixels in 2D image [10] used in the original Selective Search. Using supervoxels presents two merits. First, supervoxel segmentation is reportedly superior to 2D image based superpixel segmentation such as SLIC [11] both with boundary recall and with undersegmentation error. A better segmentation method can produce better object candidates. Second, supervoxels are accompanied with a better adjacency matrix because the adjacency relation between two supervoxels is examined in 3D space instead of the 2D image plane, which can prevent foreground objects from being combined with background objects. We achieved improvement in prediction of the object candidates by adding modifications to the initialization process of the original supervoxel segmentation proposed by Papon et al. [9]. The related details are presented in Section III-B.

Main contributions of this work are the following.

- We developed a method to produce general object candidates using 3D structures, which is the new combination of Selective Search [6] and supervoxel segmentation [9].
- We proposed a better "seeding" method for supervoxels, which improves the accuracy of object candidates.
- We demonstrated that our method outperformed 2D-image-based state-of-the-art methods of generating object candidates for the top 70 or 100 windows per image, as evaluated using publicly available RGB-D datasets.

[1]Grad. School of Information Science and Technology, The University of Tokyo, Japan `mi.t.u-tokyo.ac.jp`

II. Related Work

Several works have used 3D information to generate object candidates. Holz et al. [12] detected planes from point clouds captured by RGB-D cameras to cluster each object's region. This method is suitable for clustering planar objects such as a box or a book, as well as extracting table-top objects placed distant from each other. However, it cannot detect objects of more complicated shapes or objects touching other objects. Karpathy et al. [13] defined "objectness" measures in 3D data by compactness, symmetry, smoothness, local convexity, and recurrence. Although it can detect rather common objects such as cups, it is unable to detect large furniture or asymmetric objects. Aldoma et al. [14] defined more general "3D objectness" following a related work on 2D objectness [15]. Here, 3D objectness is measured by the amount of edge density, free space, and smooth superpixels for each bounding box defined in a 3D space. This approach is actually a rather time-consuming method similar to sliding windows in 2D object detection because bounding boxes should be densely sampled in a 3D space.

Among the state-of-the-art methods for generating object regions for more efficient 2D object detection than sliding windows are several learning-based approaches. Zhang et al. [7] extracted simple gradient features from each window and calculated the score of objectness via a cascaded ranking SVM (separately for each window's scale/aspect-ratio). BING [8] takes a similar approach, but it resizes all the windows of any scale and aspect ratio into a window of 8×8 pixels. Then it learns a linear classifier to compute the score of objectness against the 64-dimensional binarized normed gradients features. Although such learning-based approaches achieve extremely high performance, it remains questionable whether a truly general objectness can be learned getting rid of dataset bias. In fact, we confirmed in our experiment that the BING model learned with PASCAL VOC2007 dataset (which includes animals in an open field) is unsuitable for another dataset of indoor scenes (see Section V-C). Unlike those works, Selective Search [6] requires no learning process. Therefore, it can produce completely general object candidates with no limitations.

Object detection in 2D images has advanced remarkably. It achieves further progress with the practical use of 3D information. Gupta et al. [16] developed a system of object detection using RGB-D images, which reported relatively 56% higher performance than R-CNN [3], the state-of-the-art method for object detection in 2D images. Here, they used the combination of RGB-D contour detection and Multiscale Combinatorial Grouping [17] as an object region proposal module. We take a similar idea that is aimed at improving a 2D-image-based method via additional depth information. However, we explicitly use 3D information for supervoxel segmentation, whereas [16] used 2.5-dimensional information for object contour detection. Furthermore, our supervoxel segmentation is more efficient than their RGB-D contour detection, which requires normal gradient computation on two scales.

III. Supervoxel segmentation

Our proposed method for object regions executes supervoxel segmentation as a first step. This method works with color 3D point clouds captured by RGB-D cameras such as Kinect sensors. We added a small but non-trivial modification to this method, which brings better "seeding" of supervoxels by consideration of color information. We first describe the original supervoxel segmentation method [9] in Section III-A. Then we describe our seeding method in Section III-B.

A. *supervoxels for point clouds*

Supervoxel segmentation [9] first converts the input color 3D point cloud to equally partitioned (e.g. $10\text{mm} \times 10\text{mm} \times 10\text{mm}$ for each) voxel data. For the initialization of supervoxels, it "seeds" the centroids of voxel clusters by further partitioning the voxel data. The seed resolution R_{seed} should be significantly higher than the voxel resolution. Higher seed resolution produces fewer supervoxels. It then iteratively performs local clustering of voxels iteratively until it assigns all the voxels to supervoxels. The labels of voxels flow according to the adjacency graph, which stores 26 neighboring voxels of each voxel. This process ensures that the supervoxel labels do not straddle object boundaries that are disconnected in 3D space.

Supervoxel clustering is performed in the feature space defined as follows.

$$\mathbf{F} = [x, y, z, L, a, b, \text{FPFH}_{1\ldots33}]. \tag{1}$$

Here, x, y, z denote spatial coordinates, L, a, b denote values in CIELab color space, and $\text{FPFH}_{1\ldots33}$ denotes the 33 elements of Fast Point Feature Histograms (FPFH) [18]. Normalizing each component, the distance in this feature space is measured as

$$D = \sqrt{\alpha D_c^2 + \frac{\beta D_s^2}{R_{\text{seed}}^2} + \gamma D_{HiK}^2}, \tag{2}$$

where D_c denotes the Euclidean distance in CIELab color space, D_s is the spatial distance, and D_{HiK} denotes the Histogram Intersection Kernel of FPFH. α, β, and γ are the respective weights of corresponding components. We set $\alpha = 0.2$, $\beta = 0.4$, and $\gamma = 1.0$ in our experiments.

B. *spatial and color seeding method*

The original supervoxel segmentation method (described in the previous section) used "spatial seeding," which places the initial centroids of supervoxels at regular intervals in 3D space. Here, the seed resolution R_{seed} becomes the minimum unit of the final supervoxels, which means that objects smaller than R_{seed} are unable to be segmented. To detect a small object as an independent segment, R_{seed} should be sufficiently small, whereas overly small R_{seed} yields inefficient (or meaningless) supervoxels because supervoxels should have a certain size to be characteristic when used in the post processing of hierarchical grouping. We propose a new seeding method for the consideration of color information as well as spatial distribution. Intuitively, we seed the initial

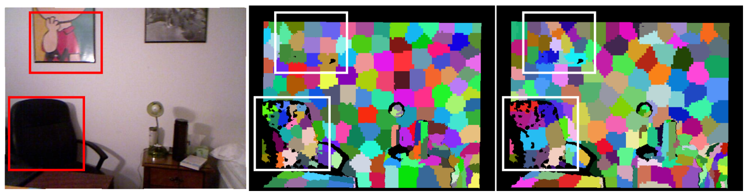
Fig. 1. Comparison of original spatial seeding (middle) and proposed seeding (right). The total number of supervoxels is 184 in both cases.

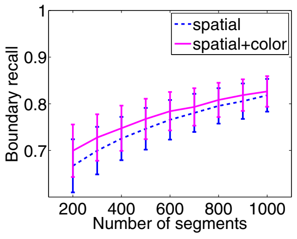

Fig. 2. Boundary recall.

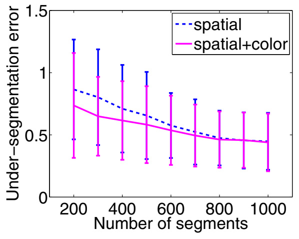

Fig. 3. Undersegmentation error.

centroids of supervoxels more densely in colorful regions than in unicolor regions.

Our seeding method is explained below. First, we perform k-means clustering of the input 3D point cloud in RGB color space. Let K denote the number of clusters. Then we perform spatial seeding at a certain seed resolution R_{seed} in the point cloud belonging to the k-th cluster and obtain a subset of seed voxels S_k. Finally, we concatenate all subsets S_k to obtain a set of seed voxels $S = S_1 \cup \cdots \cup S_K$. The comparison of original spatial seeding and the proposed seeding is presented in Fig. 1. In the case of our seeding, supervoxels are sparser on the black chair and denser on the colorful picture on the wall. The total number of supervoxels is the same (184) in both cases.

The proposed seeding method is extremely simple, but it yields a good result. We evaluated its effects with NYU depth dataset v2 [19]. This dataset includes 1,449 RGB-D images that are manually annotated with dense object labels. We show the boundary recall of supervoxels in Fig. 2 and undersegmentation error in Fig. 3. Boundary recall measures what fraction of the ground truth edges fall within at least two pixels of a supervoxel boundary, whereas undersegmentation error measures the amount of leakage across object boundaries. Letting $g_1, \ldots, g_M$ be the regions of ground truth segmentation, $s_1, \ldots, s_K$ be the regions of supervoxels, and $\mathscr{M}_i$ be the set of supervoxels necessary to cover a ground truth label g_i, undersegmentation error is defined as follows.

$$E = \frac{1}{\sum_{i=1}^{M} |g_i|} \sum_{i=1}^{M} \left(\sum_{s_j \in \mathscr{M}_i} |s_j| \right) - 1. \tag{3}$$

The lines labeled "spatial" represent the results with the original spatial seeding whereas the lines labeled "spatial+color" represent the results with the proposed seeding. We varied $R_{\text{seed}} = 0.1, 0.11, \ldots, 0.3$ for the original spatial seeding and added the choices $R_{\text{seed}} = 0.32, 0.34, \ldots, 0.5$ for the proposed method, and calculated the average scores of all the trials. These figures show that the proposed method brings better results when the number of segments (that is, supervoxels) is small. It is noteworthy that a larger number of supervoxels does not always reflect higher accuracy of predicting object candidates. Actually, the proposed seeding method yields better results with fewer supervoxels than the original seeding method. This fact is verified in the experiment described in Section V-B.

IV. Obtaining Object Candidates

We propose "3D Selective Search", which outputs object region candidates using 3D point clouds reconstructed from RGB-D images. We describe the original 2D image based Selective Search [6] in Section IV-A and the proposed 3D Selective Search in Section IV-B.

A. Selective Search

Selective Search [6] first applies superpixel segmentation [10], and then performs bottom-up hierarchical grouping of regions by combining neighboring ones iteratively, starting with superpixels. Hierarchical grouping is conducted as follows. First, a set of regions $R = \{r_1, \ldots, r_n\}$ is obtained, each region of which corresponds to a superpixel. Let S denote a set of similarity values $s(r_i, r_j)$ for all neighboring region pairs (r_i, r_j). Then, a region pair (r_i, r_j) that has the largest similarity value is selected; its union is added to the region set R $(R = R \cup r_t)$. From the similarity values set S, the elements related to r_i or r_j are subtracted ($S = S \setminus \{s(r_i, r_*), s(r_*, r_j)\}$), and a set of similarities between r_t and its neighbors S_t is added $(S = S \cup S_t)$. These processes are repeated until S becomes empty.

The similarity between a region pair $s(r_i, r_j)$ is defined as a simple summation of $s_{color}(r_i, r_j)$, $s_{texture}(r_i, r_j)$, $s_{size}(r_i, r_j)$, and $s_{fill}(r_i, r_j)$[1], as described below.

$s_{color}(r_i, r_j)$: For a region r_i, color histograms of 25 bins in three color channels are concatenated into a 75-dimensional descriptor vector $C_i = \{c_i^1, \ldots, c_i^{75}\}$. This vector is normalized by the number of pixels in r_i. Then, $s_{color}(r_i, r_j)$ is calculated using the Histogram Intersection described as follows.

$$s_{color}(r_i, r_j) = \sum_{k=1}^{75} min(c_i^k, c_j^k). \tag{4}$$

$s_{texture}(r_i, r_j)$: For a region r_i, SIFT-like gradient derivation of eight orientations in three color channels are divided into 10 bin histograms, which are concatenated into a 240-dimensional descriptor vector $T_i = \{t_i^1, \ldots, t_i^{240}\}$. This vector is normalized by its L_1 norm. Then, $s_{texture}(r_i, r_j)$ is calculated using the Histogram Intersection described as follows.

$$s_{texture}(r_i, r_j) = \sum_{k=1}^{240} min(t_i^k, t_j^k). \tag{5}$$

[1]It is an option whether any of them is omitted.

$s_{size}(r_i,r_j)$: This measure is defined as the fraction of the ratio of the union of r_i and r_j to the whole image, calculated as follows.

$$s_{size}(r_i,r_j) = 1 - \frac{size(r_i) + size(r_j)}{size(im)}. \tag{6}$$

Therein, $size(r_i)$ denotes the number of pixels in r_i; $size(im)$ denotes the number of pixels in the whole image. This measure prefers smaller regions to be merged earlier.

$s_{fill}(r_i,r_j)$: It measures how well regions r_i and r_j fit each other. Here, the degree to which areas in the bounding box of the union of r_i and r_j are filled is evaluated according to the following equation.

$$s_{fill}(r_i,r_j) = 1 - \frac{size(BB_{ij}) - size(r_i) - size(r_j)}{size(im)}. \tag{7}$$

In that equation, $size(BB_{ij})$ is the number of pixels in the bounding box of the union of r_i and r_j.

B. 3D Selective Search

Our method is the simple extension of Selective Search to 3D data. The modifications are the following.

- We use supervoxels obtained using the method described in Section III-B instead of superpixel segmentation proposed by Felzenszwalb and Huttenlocher [10].
- We check the adjacency of regions not in 2D images but in 3D space.
- We updated the definition of the similarity value $s(r_i,r_j)$.

Our definition of $s(r_i,r_j)$ is described below. First, $s_{color}(r_i,r_j)$ and $s_{texture}(r_i,r_j)$ are calculated in the same manner as the original one (see Section IV-A). We skip $s_{texture}(r_i,r_j)$ for computational efficiency and use $s_{color}(r_i,r_j)$ only as appearance similarity. For the scale similarity, we replace $s_{size}(r_i,r_j)$ by $s_{volume}(r_i,r_j)$, which uses the volume of a 3D bounding box instead of the number of pixels. Letting $volume(r_i)$ be the volume of the 3D bounding box of a point cloud belonging to region r_i and letting $volume(im)$ be the volume of the 3D bounding box of a point cloud belonging to the whole image, $s_{volume}(r_i,r_j)$ is defined as follows.

$$s_{volume}(r_i,r_j) = 1 - \frac{volume(r_i \cup r_j)}{volume(im)}. \tag{8}$$

$volume(r_i \cup r_j)$ is propagated efficiently by storing the minimum and maximum values of x,y,z coordinates of a point cloud belonging to r_i. It is noteworthy that $size(r_i)$ defined on 2D images differs from an object's real size, so it becomes larger when it is closer to a camera. Consequently, a bias exists by which the segments on distant objects tend to be merged earlier than those on nearby objects. Unlike in the above case, $volume(r_i)$ reflects the real size of objects. Therefore, no such bias exists. Finally, we discard $s_{fill}(r_i,r_j)$, which was used originally to ensure the compactness of regions. Our method requires no $s_{fill}(r_i,r_j)$ because it applies spatial seeding for supervoxel estimation, which tends to produce primarily compact regions. In summary, we define $s(r_i,r_j)$ as

$$s(r_i,r_j) = s_{color}(r_i,r_j) + s_{volume}(r_i,r_j). \tag{9}$$

We perform hierarchical grouping of regions by the similarity values defined above. Then we output all regions in inverse order of their birth. Here, we eliminate overlapping regions similarly as non-maximum suppression: we eliminate a sub-region of a region r_i if the number of the pixels in the sub-region is more than 50% of the number of pixels in r_i.

V. Experiments

A. datasets

We evaluated our method using the Berkeley 3-D Object (B3DO) dataset [20] and the NYU depth dataset v2 [19]. The B3DO dataset consists of RGB-D images associated with 2D bounding boxes of objects in 83 categories such as "chair," "keyboard," and "towel." There are 306 RGB-D images with 1,958 objects' bounding boxes in a training set and 237 RGB-D images with 1,484 objects' bounding boxes in a testing set. The NYU depth dataset v2 consists of 1,449 RGB-D images that are manually annotated with dense object labels. There are 894 labels in all. For our experiment, we used 29,095 objects' bounding boxes in 890 categories, excluding "ceiling," "floor," "unknown," and "wall."

B. validation of our supervoxel seeding method

We compared our method of spatial seeding with color clustering (described in Section III-B) with the original spatial seeding of supervoxels using the test set of B3DO dataset. Detection rate vs. the number of windows is shown in Fig. 4. The lines labeled "spatial+color" represent the results with the proposed seeding. The lines labeled "spatial" represent the results with the original spatial seeding. In the case with the proposed seeding, R_{seed} does not change the performance dramatically. Although the performance of the original seeding increases when R_{seed} decreases, the proposed seeding is still better. It is noteworthy that a larger number of supervoxels does not always improve the results. The average number of supervoxels generated for an image was 269 when we use the original spatial seeding at $R_{\text{seed}} = 0.1$, although it was 150 in the case with the proposed seeding at $R_{\text{seed}} = 0.3$. Therefore, our method is more memory-efficient because it requires fewer supervoxels.

C. comparison to state-of-the-art methods

We compared our 3D Selective Search with state-of-the-art methods of generating object proposals in 2D images: Selective Search [6] and BING [8]. We used publicly available source codes of Selective Search in `http://koen.me/research/selectivesearch/` and BING in `https://github.com/bittnt/Objectness`.

The detection rate vs. number of windows evaluated with the testing set of B3DO dataset is shown in Fig. 5. "BING (VOC2007)" denotes the BING model learned with PASCAL VOC2007 dataset, and "BING (B3DO)" denotes the BING model learned with the training set of B3DO dataset. We

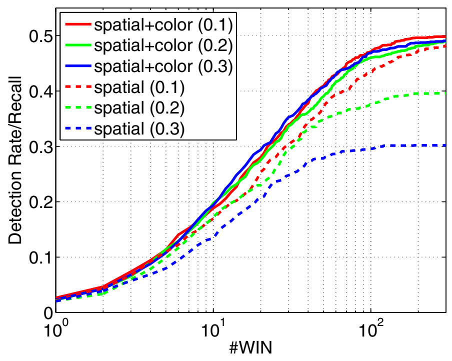

Fig. 4. Detection rate vs. number of windows. Numbers next to the methods' names are R_{seed} (m).

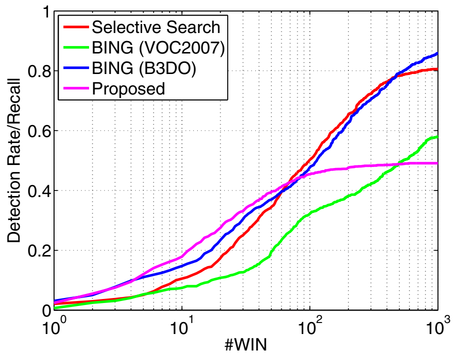

Fig. 5. Detection rate vs. number of windows. (B3DO dataset)

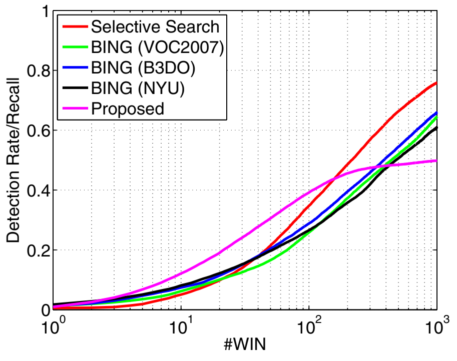

Fig. 6. Detection rate vs. number of windows. (NYU depth dataset v2)

set the size of initial segmentation for Selective Search to 100, which was slightly better than the default number of 200. We set the seed resolution R_{seed} for supervoxels to 0.15. BING (B3DO) is actually considerably superior to BING (VOC2007). This is true probably because of the dataset bias that B3DO dataset mostly emphasizes objects in indoor scenes, whereas the VOC2007 dataset includes outdoor animals such as "horse" and "sheep." Therefore, the objectness is probably difficult to be generalized. Learning-based methods tend to be affected by the dataset characteristics. The proposed method performed best when the number of windows was less than 70, but it was outperformed by other methods when it was greater than that. We consider the reason to be the following. The proposed method fails when depth data for an object are missing (see Fig. 7), which is actually not rarely the case. Because we included such objects with missing depth data as ground truth, 2D image-based methods are beneficial when the number of windows is sufficiently large. We believe that this problem will be resolved if the quality of depth sensors is improved.

The detection rate vs. number of windows evaluated using the NYU depth dataset v2 is shown in Fig. 6. The figure shows a similar tendency of the results with B3DO dataset shown in Fig. 5. "BING (NYU)" denotes the BING model learned with all images in the NYU depth dataset v2. Surprisingly, BING (NYU) was not better than BING (B3DO). This is probably attributable to the diversity of objects of 890 categories in the NYU depth dataset v2, which made it difficult to learn a single model of objectness. In contrast, Selective Search and our 3D Selective Search performed better than the others. The proposed method was best when the number of windows was less than 110.

Figure 8 presents an illustration of true positive windows in top 30 windows for each test image of NYU depth dataset v2. We consider a window to be true positive if the overlap region of the window and the bounding box of a ground truth is greater than 50% of their union. Results show that the windows generated using our method fit ground truth objects well. In addition, our method can detect small objects, which are difficult to detect using 2D image based methods.

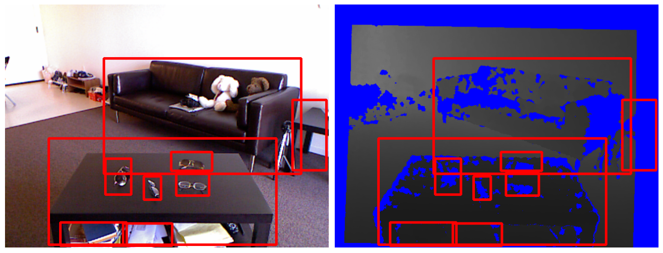

Fig. 7. Example of missing depth data. Blue pixels on the depth image (right) have NaN values. See glasses on a table and the rightmost object.

VI. Conclusion

As described in this paper, we proposed a novel method for obtaining object candidates in 3D space. The proposed method is a simple extension of Selective Search to 3D data which substitutes supervoxels estimated in 3D space for superpixels in a 2D image. For supervoxel segmentation, we improved the seeding method by considering color information, which brought better object candidates in a memory-efficient way. We compared our method to several 2D image based methods of generating object proposals. The proposed method was the best with the top 70 windows per image for B3DO dataset and with top 100 windows per image for NYU depth dataset v2. Our future work shall consider not only color/size similarity but also similarity of the 3D structure (e.g. curvature or the direction of normals) of two regions in the hierarchical region grouping process.

References

[1] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, pp. 1–42, 2014.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2014.

[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32, no. 9, 2010.

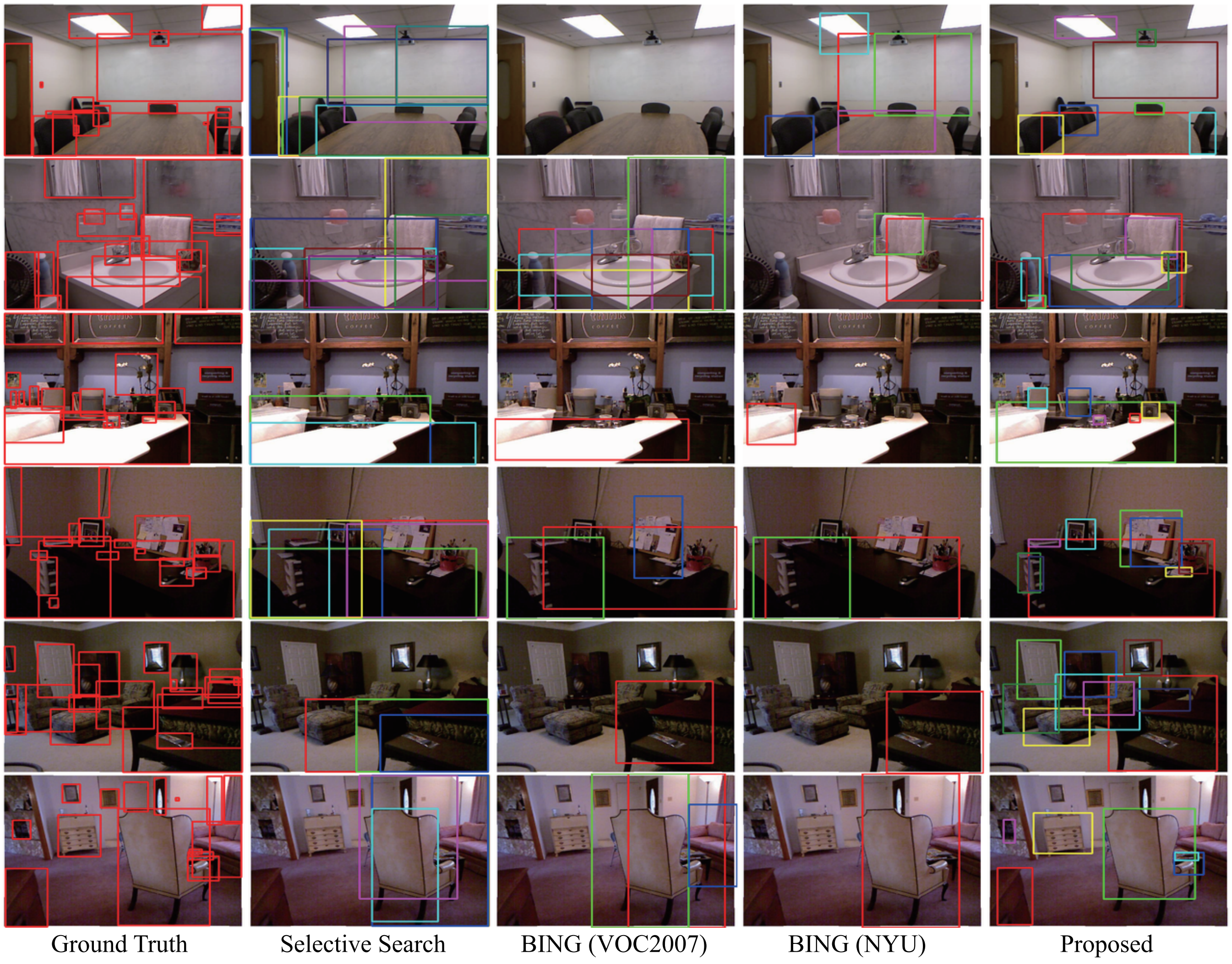

Fig. 8. True positive windows in top 30 windows for each test image of NYU depth dataset v2.

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2005.

[6] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[7] Z. Zhang, J. Warrell, and P. H. Torr, "Proposal generation for object detection using cascaded ranking SVMs," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2011.

[8] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "BING: Binarized normed gradients for objectness estimation at 300fps," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2014.

[9] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter, "Voxel cloud connectivity segmentation-supervoxels for point clouds," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2013.

[10] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[11] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, no. 11, pp. 2274–2282, 2012.

[12] D. Holz, S. Holzer, R. B. Rusu, and S. Behnke, "Real-time plane segmentation using RGB-D cameras," in *RoboCup 2011: Robot Soccer World Cup XV*, 2012.

[13] A. Karpathy, S. Miller, and L. Fei-Fei, "Object discovery in 3D scenes via shape analysis," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[14] A. Aldoma, M. Vincze, and F. Tombari, "Localizing and segmenting objects with 3D objectness," in *Proc. the 18th Computer Vision Winter Workshop (CVWW 2013)*, 2013.

[15] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, no. 11, pp. 2189–2202, 2012.

[16] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. European Conference on Computer Vision (ECCV)*, 2014.

[17] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proc. IEEE Computer Vision and Pattern Recognition (CVPR)*, 2014.

[18] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

[19] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. European Conference on Computer Vision (ECCV)*, 2012.

[20] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. F. K. Saenko, and T. Darrell, "A category-level 3-D object dataset: Putting the kinect to work," in *Proc. IEEE ICCV Workshop on Consumer Depth Cameras in Computer Vision*, 2011.