Faster 3D Object Detection in RGB-D Image Using 3D Selective Search and Object Pruning

Jiang Liu[1], Hongliang Chen[2] Jianxun Li[1]
1. Department of Automation, Shanghai Jiao Tong University, Shanghai, 200240
E-mail: jiangliux@sjtu.edu.cn, lijx@sjtu.edu.cn
2. Institute of Electro-optical Equipment, AVIC, Luoyang, 471009
E-mail: chenhongliangly@163.com

Abstract: 3D object detection in RGB-D image has received considerable attention recently. But potential searching space in testing image is large, and extracting handcraft feature for every candidate bounding box is computational expensive. In this paper we introduce 3D Selective Search(SS) to generate high quality cuboids that most likely to contain object. Besides, Object Pruning is proposed to speed up the testing process by cutting hypothesis cuboids. The result evaluated on SUN RGB-D dataset demonstrates that our method is able to speed up testing process more than 50% without performance loss, compared with exhaustive searching in 3D space.

Key Words: 3D Object Detection, RGB-D Image, 3D Selective Search, Object Pruning, COG

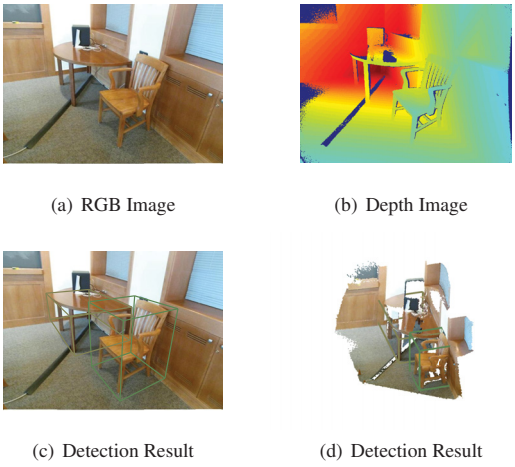
1 Introduction

3D object detection has received considerable attention recently for the wide range of applications in autonomous driving and personal robotics[1]. Traditional detection method only localize objects in 2D bounding boxes, missing the information of 3D location, orientation and 3D extent. In contrast, 3D object detection provide accurate 3D information to understand real world, thus plays an important role in morden computer vision.

It is difficult to solve this task in RGB image due to its natural ambiguity of missing depth. The most common approach is to discretize the viewing sphere into bins and train a 2D detector for each viewpoint [2]. However, only weak 3D information can be obtained in these methods. Besides, object-centerd methods establishes spatial connections betwee views by mapping them directly to the surface of 3D model. Though these types of models seems attractive for the continuous viewpoint represantations, their detection performance has typically been inferior to 2D deformable models. More recently, [3] extends 2D deformable part-based models(DPMs)[4] to 3D space by means of a deformable 3D cuboid.

With the the availability of inexpensive RGB-D sensors, such as Microsoft Kinect, Apple PrimeSense, Intel RealSense, and Google Project Tango, larger and more ambitious RGB-D datasets are created which enabled major breakthroughs for highlevel scene understanding[5, 6]. SUN RGB-D dataset[7], which contains 10,335 RGB-D images with dense annotations in 3D, has become a defacto standard for scene understanding.

This work was jointly supported by National Natural Science Foundation (61673265); 973 project(6133190302); Aeronautical Science Foundation of China(20140157001); 2015 Industry-university-research cooperation project of AVIC



(a) RGB Image (b) Depth Image

(c) Detection Result (d) Detection Result

Figure 1: A illustrative example define the task of 3D object detection. Top row are the RGB image and depth image of same scene from SUN RGB-D dataset. Given them as input, we need to find out the cuboid containing the objects of interst. 1(c) is the detection result in projected RGB image while 1(d) display them in 3D.

It is important to exploit the depth information to advance 3D object detection. Sliding Shapes[8] was proposed to slide a 3D detection window in 3D space, detect objects by matching to CAD models in sliding locations. While CAD models can potentially provide abundant information, the number of models for all categories is limited, and thus this method focused only on a small number of categories. Ren[9] proposed the Clouds of Oriented Gradients(COG) feature that links the 2D appearance and 3D pose of object categories. COG accurately describes the 3D appearance

978-1-5386-1243-9/18/$31.00

of objects with complex 3D geometry. But its testing process is time-consuming by evaluating every combination of voxel size, 3D location and orientation.
We improve 3D object deteion algorithm based on [9]. Our contributions are mainly two folds:

1. First, instead of scanning every possible location and orientation in 3D space, we use 3D Selective Search(SS) to generate candidate cuboids that are most possible to contain object.

2. Second, Object Pruning is uesed to reduce hypothesis cuboids in testing stage, thus furthur speed up testing process.

Follow the introduction, related work about object detection in RGB-D image is described in Section 2 and our method is presented in Section 3. The result is described in Section 4 and then summarized in Section 5.

2 Related Works

In this section, related works about 3D object detection in RGB-D images are metioned. We introduce the works by two key factor of object detection: feature representation of objects, and searching method in testing image.

2.1 3D Feature

The histogram of oriented gradient (HOG) descriptor [10] is widely used in object detection. Basiclly, it counts occurrences of gradient orientation in localized portions of an image, and its performance is good enough for 2D detection. However, since gradient orientations are determined by 3D object orientation and perspective projection, HOG descriptors that are naively extracted in 2D image coordinates is not suitable for 3D object description.
To handle this issue, many work has been done. some used extra CAD model to describe the edge from various viewpoints[11].Other assumed that object in sepecific views are near-planar[3]. previous extensions of the HOG descriptor to 3D space require full mesh model[12]. Rencently the cloud of oriented gradient(COG) feature[9] is proposed to accurately model the 3D apperence of objects. Like HOG , the calculation of COG can be divided into 3 steps:

1. Gradient computation. Gradient of the image is computed by applying filters [-1,0,1], [-1,0,1]$^{\mathrm{T}}$ to RGB channels. Gradient in position (x, y) is obtained by setting maximum responses across color channels to (dx, dy), with corresponding magnitude $\sqrt{(dx^2 + dy^2)}$.

2. 3D orientation bin construction. Cuboid contains the object is devided it into $6 \times 6 \times 6$ voxels, and 9 orientation bins are constructed for each voxel to model the distribution of local 3D gradient orientation. Perspective projection is used to find corresponding 2D bin boundaries. For each point lies in a voxel, its projected gradient magnitude is sumed into corresponding 2D orientation bin, which is back-projected into 3D bin.

3. Normalization. Bilinearly interpolation is performed between neighboring bins. Histogram $\phi^a_{il}$for voxel l in cuboid i is normalized by setting $\phi^a_{il} \leftarrow \phi^a_{il}/\sqrt{\|\phi^a_{il}\|^2 + \epsilon}$ for a small $\epsilon > 0$, so the dimension of COG feature is a fixed size of $6^3 \times 9 = 1944$.

Besides the apperance features like COG, other 3D features aim to model the object geometry like point cloud density and 3D normal orientation. For voxel l in cuboid i contains N_{il} points, the point cloud density feature equals $\phi^b_{il} = N_{il}/A_{il}$, where A_{il} is the are of silhoutte of the voxel. This kind of normalization makes the object in the scene robust to depth variation and partial occlusion. 3D normal orientation feature ϕ^c_{il} is obtained by building a 25-bin histogram of normal orientation within voxel l. The normnal orientation of each 3D point is calculated via a plane fit to its some nearest neighbors. These geometry features combined with appearence features are able to most accurately model the object in 3D.

2.2 3D Search

Search algorithm define the hypothesis space for potention object location in testing image. Sliding window method is widely used in 2D detection algorithm[4, 10] and the core idea is intuitive. An exhaustive search with predefined step is performed where all the location within the image is scanned to not miss any potential object location. Scale is considered by examined different size of image patch at that location. However, searching all the possible location is computationally expensive. Selective search is proposed for fast computation with high recall[13]. First, initial regions is obtained by performing fast segmentation[14]. Later, the most similar neighbouring region pair is merged into a new region and add to the set in every iteration. Finally, object location boxes are extracted from all regions in the set.
The current search methods in 3D naively generlized from 2D. Sliding shape[8] train an Exemplar-SVM on a CG model rendered at a specific 3D location relative to the virtual camera, then perform exhaust search only at the nearby location of the CG model in order to improve the speed. This 3D local search approach results a relative low recall rate. In [9], 3D space is discritized into grids. Then at each possible location, certain size of bounding boxes which based on the empirical statistics of trainning bounding boxes, along with 16 candidate orientation are examined. This strategy brings a high recall rate but is rather slow by extracing features for every candidate cuboid in testing image.

3 Faster 3D Detector

Instead of exhuastively searching testing image as in [9], we speed up testing step dramaticlly using 3D Selective Search. With object pruning, we further reduce the testing time by saving time for feature computing, and reduce false positive without much recall decrease.
The complete pipeline is shown in Fig 2. During trainning(Sec. 3.1), structred SVM is learned for each class with COG combined with geometry features described in

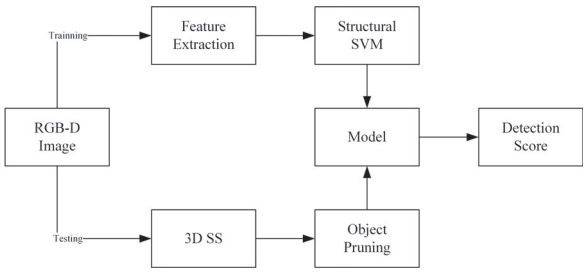


Figure 2: Pipeline of proposed method

Sec. 2.1, without the extra information of CG model. During testing(Sec. 3.2), we use learned structured SVMs to classify candidate cuboids , and output 3D bounding boxes with detection scores.

3.1 Trainning

Our goal is, given certain categroy, to learn a prediction function which maps an RGB-D image I to a 3D cuboid $B = (L, \theta, S)$. L refers to the cuboid center, θ is the cuboid orientation, and S is the physical size(length, width, height) of the cuboid. For each voxel l in cuboid B_i annotated in trainning image I_i, we have the COG feature ϕ^a_{il}, point cloud density feature ϕ^b_{il}, and normal histogram feature ϕ^c_{il}. Thus the feature of cuboid i is $\phi(I_i, B_i) = \{\phi^a_{il}, \phi^b_{il}, \phi^c_{il}\}^{216}_{l=1}$.
Given n trainning examples of category c, we train a structural SVM[18] as in [9] with margin rescaling constraints:

$$\begin{aligned} \min_{\omega_c, \xi_i \geq 0} \quad & \frac{1}{2}\omega_c^{\mathrm{T}}\omega_c + \frac{C}{n}\sum_{i=1}^{n}\xi_i \quad \text{subject to} \\ \omega_c^{\mathrm{T}}[\phi(I_i, B_i) - \phi(I_i, \bar{B}_i)] &\geq \Delta(B_i, \bar{B}_i) - \xi_i, \\ & \text{for all } \bar{B}_i \in \beta_i, i = 1, ..., n. \end{aligned} \tag{1}$$

$\phi(I_i, B_i)$ are the features for 3D bounding box B_i in the given RGBD image I_i. B_i is the ground truth annotated bounding box, and β_i is the set of candidate bounding boxes. If multiple instances are in the same image, we add images to the trainning set multutiple times, each time removing the 3D points contained in other bounding boxes. Also, we use the following loss function to describe the fitness of groundtruth cuboid B and estimated cuboid $\bar{B}$:

$$\Delta(B, \bar{B}) = 1 - \text{IOU}\left(B, \bar{B}\right) \cdot \left(\frac{1 + \cos(\theta - \bar{\theta})}{2}\right) \tag{2}$$

$\text{IOU}(B, \bar{B})$ is the volume of intersection of the cuboids, divided by the volume of their 3D union. $\theta - \bar{\theta}$ is the diffrence between cuboid orientation. After trainning, we obtain the model to dscriminate wether a cuboid belongs to this categroy.

3.2 Testing

Searching candidate cuboid exhaustively in 3D space is time consuming and results many false positive. In order to fast compute cuboid hypotheses in testing image, first 3D SS is applied to obtain about 2000 candidate 3D bounding boxes, then Object Pruning is used to further select these candidates to save time for feature computation and testing.

3D selective search is first proposed in [15]. But it finally outputs candidate bounding boxes in 2D image. [16] rectified this method and output 3D bounding boxes. First, RANSAC is used to fit the pane of 3D point cloud to obtain an initial segmentation. For each plane that its projection covers more than 10% of the total image area, RGB-D UCM segmentation from [17](with threshold 0.2) is used for further splitting. Then starting with this oversegmentation, different segmentation regions are hierarchically grouped, with the following similarity measures:

- $s_{color}(r_i, r_j)$ is the measurment of color similarity between region r_i and r_j using RGB color histogram intersection.
- $s_{\#pixels}(r_i, r_j) = 1 - \frac{\#pixels(r_i) + \#pixels(r_j)}{\#pixels(im)}$, where $\#pixels(\cdot)$ is number of pixels in this region.
- $s_{volume}(r_i, r_j) = 1 - \frac{volume(r_i) + volume(r_j)}{volume(room)}$, where $volume(\cdot)$ is the volume of 3D bounding boxes of the points in this region.
- $s_{fill}(r_i, r_j) = 1 - \frac{volume(r_i) + volume(r_j)}{volume(r_i \cup r_j)}$ describe how well region r_i and r_j fit into each other.

Algorithm 1 3D Selective Search

Input: RGB-D image
Output: Set of 3D object location hypotheses B
obtain initial region $R = \{r_1, ..., r_n\}$ use plane fitting
Initialize similarity set $S = \emptyset$
for all Neighbouring region pair (r_i, r_j) **do**
  Calaulate similarity $s(r_i, r_j)$
  $S = S \bigcup s(r_i, r_j)$
end for
while $S \neq \emptyset$ **do**
  Get highest similarity $s(r_i, r_j) = \max(S)$
  Merge corresponding regions $r_t = r_i \bigcup r_j$
  Remove similarities regarding $r_i : S = S \setminus s(r_i, r_*)$
  Remove similarities regarding $r_j : S = S \setminus s(r_j, r_*)$
  Calculate similarity set S_t between r_t and its neighbours
  $S = S \bigcup S_t$
  $R = R \bigcup R_t$
end while
Extract object location boxes B from all regions in R

Algorithm 2 Object Pruning

Input: Candidate boxes B obtained from Algorithm 1.
Output: Pruned boxes B for category c
Compute distribution of box size in each direction, aspect ratio of each pair of box edge, and height above ground from trainning samples of category c
for all candidate cuboid $B_i \in B$ **do**
  Calaulate the same feature of B_i
  if B_i falls outside the distribution **then**
    $B = B \setminus B_i$
  end if
end for

Table 1: Average Precision for five object categories

method	bed	table	sofa	chair	toilet
Sliding-Shape	42.95	19.66	20.60	28.21	60.89
Geom	7.14	14.76	26.02	23.79	1.16
Geom+COG	51.98	27.94	41.36	45.04	42.97
Ours	52.49	28.97	42.78	44.04	42.53

The weighted sum of these four terms form the final output of similarity measuremenmt. Since both 3D and color cues are combined, this very strong baseline achieves an average recall 74.2. However, for convinience, orientations of all the cuboid are the same with room orientation, which reduce the accuracy of detection.

When using cuboids to represent objects, the cuboid sizes provide underlying information about the object categories. That is the main motivation to use object pruning. First, we precompute the distribution of phisical size along each direction, aspect ratio of each pair of cuboid edge, and the centroid above ground for category c. Then, for each candidate cuboid, we compare these numbers with the corresponding distribution. If these values falls outside 1st to 99th percentile of the distribution, which indicate the cuboid has abnomal size, we abandon this cuboid for further testing of this catergory.

After object pruning, we computer the feature for prunned candidate cuboids as in trainning step. All these cuboids are evaluated use structral SVM trainned for this categroy.

As in [16], 3D Non-maximum suppression(NMS) is applied to remove redundant bounding boxes. First, all the detected cuboids of same category are sorted based on the detection score. Then in each iteration, always select the first cuboid which has greatest detection score, and other cuboids with IOU greater than 0.35 are abandoned. At the end of iteration, we manually put the first cuboid to the last. The whole process stoped when no more cuboids are discarded.

4 Experiments

We evaluate our model on SUN RGBD dataset[7], and compare the result with [8] and [9]. Due to the different implementation, there is slightly performance difference compared with original paper in [9]. We learn and evaluate appreance model of 5 object categories as in [8]. Object cuboid are generated and evaluated as described in the previous section. Our computer is equipped with i5 and 8G RAM.

we follow the evaluation metric in [8] with assumption that all predictions and ground truth boxes are aligned in the gravity direction. Intersection-over-union with ground-truth cuboid annotations are evaluated, and candidate boxes with score above 0.25 are consider to be correct detected. These boxes will be used to calculate the average recall for candidate cuboids generation and average precision for detection.

4.1 Object Detection Evaluation

Table 1 shows the result of 3D object Detectin. Compared with using geometry features alone, adding COG feature can greatly improve the detection performance. This is mainly due to the noisy point clouds obtained from depth image are imperfect.

Ground Truth Detection Result

Figure 3: Detections with confidence scores larger than the same threshold.

Table 2: Average Testing Time per Image

method	proposal number	time(min)
Exhaust search as in [9]	$\geq 10^8$	≥ 60
3D SS	2000	30
3D SS + Object Pruning	1200	19

With 3D SS and Object Pruning, there is nearly no performance loss. The main reasons are as follows: first, quality of object hypotheses generated by 3D Selective Search is high, and reasonablely consistent over different object categories; second, the cuboid sizes provide underlying information about the object categories. So the possibility of pruned boxes containing object is rather low.

4.2 Computation Speed Evaluation

The final result in Table 2 is optimized by parallel computing. It demonstrates that 3D SS greatly reduce the candidate bounding boxes to evaluate. And object pruning can further reduce about 800 proposals per image.

In object detection, the most time spends on feature extracting. It takes about 2 to 3 second per bounding box to extract feature.Thus less bounding boxes to evaluate would result faster evaluation process.

5 Comculusion

We imrpoved the method in [9] for amodal 3D object detection in RGB-D images. Using 3D Selective Search and Object Prunning, our testing time reduced more than 50%. We trained accurate 3D detectors for five object categories using COG and geometry features. Without external CAD models, our detectors can be generalized to more categories.

REFERENCES

[1] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond pascal: A benchmark for 3d object detection in the wild," in *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pp. 75–82, IEEE, 2014.

[2] C. Gu and X. Ren, "Discriminative mixture-of-templates for viewpoint classification," *Computer Vision–ECCV 2010*, pp. 408–421, 2010.

[3] S. Fidler, S. Dickinson, and R. Urtasun, "3d object detection and viewpoint estimation with a deformable 3d cuboid model," in *Advances in neural information processing systems*, pp. 611–619, 2012.

[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[5] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," *Computer Vision–ECCV 2012*, pp. 746–760, 2012.

[6] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, "A category-level 3d object dataset: Putting the kinect to work," in *Consumer Depth Cameras for Computer Vision*, pp. 141–165, Springer, 2013.

[7] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 567–576, 2015.

[8] S. Song and J. Xiao, "Sliding shapes for 3d object detection in depth images," in *European conference on computer vision*, pp. 634–651, Springer, 2014.

[9] Z. Ren and E. B. Sudderth, "Three-dimensional object detection and layout prediction using clouds of oriented gradients," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1525–1533, 2016.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1, pp. 886–893, IEEE, 2005.

[11] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic, "Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3762–3769, 2014.

[12] N. Buch, J. Orwell, and S. A. Velastin, "3d extended histogram of oriented gradients (3dhog) for classification of road users in urban scenes," 2009.

[13] 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.

[14] 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.

[15] A. Kanezaki and T. Harada, "3d selective search for obtaining object candidates," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 82–87, IEEE, 2015.

[16] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 808–816, 2016.

[17] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *European Conference on Computer Vision*, pp. 345–360, Springer, 2014.

[18] T. Joachims, T. Finley, and C.-N. J. Yu, "Cutting-plane training of structural svms," *Machine Learning*, vol. 77, no. 1, pp. 27–59, 2009.