Single Multi-feature detector for Amodal 3D Object Detection in RGB-D Images

Qianhui Luo, Huifang Ma, Yue Wang, Li Tang, Rong Xiong
Institute of Cyber-Systems and Control, Zhejiang University
{qianhuiluo, hfma, rxiong}@zju.edu.cn, wangyue@iipc.zju.edu.cn, litang.cv@gmail.com

Abstract

This paper aims at fast and high-accuracy amodal 3D object detections in RGB-D images, which requires a compact 3D bounding box around the whole object even under partial observations. To avoid the time-consuming proposals preextraction, we propose a single end-to-end framework based on the deep neural networks which hierarchically incorporates appearance and geometric features from 2.5D representation to 3D objects. The depth information has helped on reducing the output space of 3D bounding boxes into a manageable set of 3D anchor boxes with different sizes on multiple feature layers. At prediction time, in a convolutional fashion, the network predicts scores for categories and adjustments for locations, sizes and orientations of each 3D anchor box, which has considered multi-scale 2D features. Experiments on the challenging SUN RGB-D datasets show that our algorithm outperforms the state-of-the-art by 10.2 in mAP and is $88 \times$ faster than the Deep Sliding Shape. In addition, experiments suggest our algorithm even with a smaller input image size performs comparably but is $454 \times$ faster than the state-of-art on NYUV2 datasets.

1. Introduction

Object detection has currently achieved high accuracy in real-time due to extensive use of large-scale datasets and convolution neural networks (ConvNets). The task is to produce the category and location with a bounding box for objects in 2D image plane, which is very useful for scene understanding related applications. But in real-time perceptive system such as SLAM, robotic navigation, obstacle avoidance, and augmentation reality, 2D bounding box is not sufficient because of a lack of 3D information, like pose, size as well as geometric location in real world. In this paper, we revisit the problem of amodal 3D object detection for indoor scenarios, which aims to predict a 3D bounding box surrounding the whole object even if it is only partially observed.

Due to the heavy occlusion in indoor scenes and prediction for one more dimension, 3D object detection itself is a challenging task with great difficulty. However, the launch of RGB-D sensors (e.g. Kinect and Xtion) and emergence of large-scale 3D dataset [21] bring an opportunity for it. The mainstream approaches fall into two categories: Some works [24, 14, 6, 3] encode depth as extra channels to color information in order to make best use of successful 2D ConvNets and design methods to convert 2D proposals to counterpart 3D boxes, this kind of works is called as 2.5D approaches. The others, named 3D approaches, to explore better 3D representations, convert depth image into point cloud firstly, then design manually [22, 19] or learn geometric features for detection proposals in 3D space using 3D ConvNets [23]. So far, both 2.5D and 3D methods lead the 3D detection performance on the SUN RGB-D datasets and NYUV2 datasets.

Nevertheless, three main issues still exist in the current literature: (1) Runtime for operations in 3D is long. For both tasks of search for object candidates in 3D space and computation of features for 3D data, adding one more spatial dimension consumes a significant amount of computation time. (2) Features of incomplete sparse 3D data are hardly salient. When projecting pixels of depth image back to 3D space, the generated 3D point cloud are usually noisy and sparse due to the low resolution of RGB-D sensors as well as reflections and occlusions. The precision of extracted geometric features for incomplete 3D shape are sometimes insufficient. (3) 2.5D approaches are all based on a proposal-driven mechanism, which is more complicated and time-consuming. Although these approaches utilize dense and contiguity characteristics in 2D images, obtaining 2D bounding box proposals spends more time and it is more complicated to train a model containing more than one stage. Therefore, our intuition is to design a single approach to obtain higher accuracy along with faster runtime.

In this paper, we propose a single multi-feature detector which takes a pair of RGB-D images as input and detects 3D objects. Our detector is an end-to-end framework based on 2D ConvNets without 2D bounding box proposals. As illustrated in Figure 1, the single multi-feature detector consists of two parts: a features extraction network and a hier-
architectural fusion network. In the features extraction network, two separate subnetworks are used to learn appearance features in the RGB image and geometric features in the depth image. The multi-feature fusion network concatenates the feature maps from multiple inputs. Fusing RGB and depth information in the early stage of a deep neural network may ignore the surrounding context while fusion in the late stage may lose some details, so we propose to hierarchically fuse RGB and depth information in multiple stages. To avoid searching a huge number of candidates in the full 3D space, every location of a feature map for predicting detections is attached with a set of 3D anchor boxes with different sizes. And in addition to supplement geometric information, depth image has helped to determine the locations of 3D anchor boxes. Additionally, to address 2D objects of various size, our detector incorporates multiple feature maps with different resolutions to regress locations, sizes, orientations and predict categories for these 3D anchor boxes in a convolutional way. We evaluate our approach for the amodal 3D object detection in challenging SUN RGB-D and NYUV2 datasets. Experiments on the SUN RGB-D datasets show that our algorithm outperforms the state-of-the-art by 10.2 in mAP and is 88× faster than the Deep Sliding Shape [23]. In addition, experiments suggest our algorithm even with a smaller input image size performs comparably but is 454× faster than the state-of-art on NYUV2 dataset. Our contributions can be summarized as follows:

1. We are the first to design a single end-to-end framework based on 2D ConvNets to predict locations, sizes, orientations and categories for objects in 3D space. The proposed method takes only a pair of RGB-D images as input and does not need to extract 2D bounding box proposals.

2. We propose a hierarchical fusion structure to concatenate the features in RGB and depth images.

3. We do not make Manhattan world assumption and 3D CAD models for orientations estimations.

2. Related works

Object detection has progressed rapidly since RCNN[5] used ConvNets to predict a bounding box around the visible part of an object in image plane. The following Faster-RCNN[4], Faster-RCNN[18], YOLO[16, 17], SSD[13], Mask-RCNN[8], Focal Loss[12] have gradually improved accuracy and speed, but their results are all in 2D. Earlier work[24, 25] have focus on detecting objects in 3D from RGB-D images. [24] generates multiple hypothesical object foreground masks for each detected 2D bounding box and then generates corresponding point cloud as well as its handcrafted features for each mask. The 3D DPM algorithm generalized to RGB-D images is used to determine the 3D location for the foreground object. [25] first generates candidate cuboids through the CPMC algorithm and utilizes features extracted from RGB-D images as well as contextual relations to assign semantic labels for these cuboids. Then the success of ConvNets used in 2D detection and arrival of 3D sensors drive a new era in 3D detection. We briefly review existing work on 3D object detection in RGB-D images, multiple feature maps for 3D detection and multi-feature fusion.

3D object detection in RGB-D images. Sliding shapes[22] firstly converts depth image into point cloud and then slides 3D detection windows in 3D space. Handcrafted features for each 3D window are later fed into exemplar SVMs for a object class. An exemplar SVM is trained by rendering a CAD model into synthetic depth images. To slide a window in 3D space and repeated 3D feature computation along with the use of many exemplar SVMs cause this method very slow. Like [22], [14] also adopts a sliding-window fashion in the 3D point cloud where raw features for each 3D box are computed by applying two separate RCNNs to 2D projection of the 3D box in RGB and depth images. A deep Boltzmann Machine trained with CAD
models is acted on raw features to exploit cross-modality features from RGB-D images. At last, exemplar SVMs are also used to determine detections. In the same year, different from previous sliding-window methods, [6] employs the R-CNN to estimate a coarse pose for each instance segmentation in the 2D image by encoding a depth image as the input 3-channels image(local surface normal vector $N_x, N_y$ and $N_z$ w.r.t gravity estimation) and then fits a CAD modal to the point cloud inside the segmentation by ICP alignment. To explore better 3D representation, inspired by the Faster-RCNN, [23] divides a 3D scene recovered from RGB-D images into 3D voxel grids and designs a 3D ConvNet(named 3D RPN) to extract 3D region proposals. Next, for each 3D proposal, another separate 3D ConvNet(named 3D ORN) is used to learn object category and for 3D box regression. The involved 3D ConvNets result a significant improvement in performance and efficiency than [22], [19] proposes a cloud of oriented gradient descriptor for a sliding 3D box and improves [22] dramatically in performance. To reduce the search space in 3D, [10] makes use of the Faster-RCNN to detect objects in the RGB image and then takes out point cloud inside each 2D bounding box to regress locations and sizes in 3D by using a multilayer perceptron. [22], [23], [19] and [10] all make Manhattan world assumption and align sliding boxes in 3D with the estimated room. [3] chooses time-consuming external multiscale combinatorial grouping(MCG)[1] in RGB-D images[7] to obtain 2D bounding box proposals. Based on the Faster-RCNN, for each 2D proposal, features from RGB and depth images are integrated for classification and 3D box regression. In this paper, we do not restructure point cloud from depth images because it is often noisy and operations in 3D are time-consuming. Our method is a single end-to-end framework which operates in 2D images and does not require 2D bounding box proposals.

**Multiple feature maps for 3D detection.** SSD[13] and FPN[11] use multi-scale feature maps for 2D object detection. While they achieve inspiring performance, none of existing work opts multi-scale features for 3D object detection. In this work, we design a 3D object detection framework by adopting multi-scale feature maps.

**Multi-feature fusion.** Most existing work combine features from RGB and depth images in the late stage of a deep neural network for 3D object detection. [13] uses a deep Boltzmann Machine to fuse features outputted from two independent RCNNs. [23], [7], [3] concatenate the features from RGB-D images before predictions. Our network differs from previous methods because we hierarchically fuse features both in earlier and middle stages.

### 3. Single Multi-feature detector

Our single multi-feature detector takes a pair of RGB-D images as input and hierarchically fuses features extracted from RGB and depth images. Without 2D bounding box proposals, we use depth image to help on generating a set of 3D anchor boxes with different sizes in every location of a feature map. Multi-scale feature maps are used for predicting object category and regressing 3D bounding box offsets relative to anchor boxes.

#### 3.1. Model

**Multi-scale feature maps for 3D detection.** Utilize only 2D features in images but be able to detect objects in 3D world, one problem needs to be considered: objects in 2D images vary a lot in size. When 2D bounding box proposals are not provided, a small(3×3) convolutional filter(ConvFilter) needs to be applied in every location of a feature map and then output detection predictions. When this small ConvFilter acts on the shallow feature maps, its receptive field is limited and only captures local features for big objects. However, deep feature maps possessed with high-level abstract information may miss details for small objects. To address this issue, we use multiple feature maps of different-scales for 3D object detection. As is shown in Figure 1, we use 6 feature maps to produce detections, followed by a 3D non-maximum suppression to get the final detected objects.

**3D anchor boxes.** We tag 13 3D anchor boxes with different sizes in every location of a feature map, the ConvFilter applied in each location produces adjustments for locations, sizes and orientations relative to these 13 anchor boxes, as well as the scores over each category for each of those boxes. Our algorithm combines 6 feature maps with different resolutions for predictions by assigning shallow feature maps to aim at small objects in images in contrast to deep feature maps for big objects. The visible parts in images of a big 3D object may be very small due to a long distance to sensors, for example, the pixel areas of a bed and a chair can be similar in picture while their 3D physical sizes are very different. Thus shallow features maps are used to predict detections for not only big but also small objects in 3D space and so do deep feature maps. Hence, we fix the scales of these 13 3D anchor boxes on these 6 feature maps although the sizes of these 13 3D anchor boxes are different. For an illustration of 3D anchor boxes, please refer to Figure 2. For each 3D anchor box at a given location, the small ConvFilter computes its scores for $c$ classes and 7 offsets relative to its location, size and orientation. Thus, ConvFilter at each location outputs $13 \times (c + 7)$ values.

**Hierarchical fusion.** We adopt SSD[13] as our base network because it considers multi-scale issue and has achieved high-accuracy in real-time 2D object detection. As shown in Figure 1, the input RGB and depth images go through two atrous VGG-16 networks[20] subsampled by SSD to compute appearance and geometric features. To combine features from different inputs, prior work usually...
use early fusion[2] or late fusion[23][3]. Early fusion combines multiple images in the input stage. And late fusion use independent subnetworks to learning respective features for multi-inputs and combine outputs from subnetworks before prediction. To integrate details captured in early stage with context information modeled in deeper layers, we propose to adopt a hierarchical fusion structure, which incorporates features from different inputs both in the early and middle stage. We combine the two layers with identical receptive field from different input data, followed by two $1 \times 1$ convolution layers to shuffle and choose the concatenated features. As shown in Figure[1] in the first time, we concatenate Conv4-3-rbg and Conv4-3-d. Receptive fields of these two layers are identical, which means we just need to use $1 \times 1$ convolution layers to fuse appearance and geometric features from the same region in the input images. The second time, we concatenate Conv7-rbg and Conv7-d followed by two $1 \times 1$ convolution layers to fuse different features again. The fused feature maps in two stages with different scales are a part of the final multiple feature maps used to predict detections.

3.2. Generation for 3D anchor boxes

In addition to providing the geometric information, the depth image is also used to generate 3D anchor boxes. Inspired by [3] that 3D proposals are initialized from 2D segment proposals, we propose to generate 3D anchor boxes from 2D blocks corresponding to every location on multiple feature maps, as shown in Figure[2].

**2D cell.** For a feature map of size $m \times n$ with $p$ channels, at each of the $m \times n$ locations, we define a square region on the initial input image as its 2D cell by dividing the input image into $m \times n$ grids. And the grid in $i^{th}$ row and $j^{th}$ column is the 2D cell related to the location in $i^{th}$ row and $j^{th}$ column of the feature map.

**2D block.** At each of the $m \times n$ locations of the feature map, its 2D block is composed of its 2D cell along with the 8 neighboring cells, namely a region of $3 \times 3$ cells. The 2D block is designed for covering most of an object on the image and is smaller than actual receptive fields of the layer.

**Determining centers for oriented 3D anchor boxes.** We represent a 3D bounding box by its center $(x_0, y_0, z_0)$ and size $(w, l, h)$ in the world coordinate system, along with its orientation angle $\theta$, defined as its longer edge in $xy$-plane rotated around the positive direction of $z$ axis. At each location in a feature map, we set 13 kinds of sizes for anchor boxes based on statistics of object sizes. The planes of each anchor box out of 13 are separately parallel to three axises of the world system, that is, its orientation angle $\theta$ is set to $0^\circ$. The centers of these 13 anchor boxes are the same and are inferred by point cloud projected from the corresponding 2D block region. Since the depth images are usually noisy, we choose the median depth value in the 2D block $z_{med}$ as $z_0$, and map the center $(c_x, c_y)$ of the 2D block to $(x_0, y_0)$ in the 3D space using camera intrinsic and extrinsic parameters:

$$
\begin{pmatrix}
    x_0 \\
    y_0 \\
    z_0 
\end{pmatrix}
= R_{tilt} \cdot \begin{pmatrix}
    z_{med} \cdot (c_x - o_x)/f_x \\
    z_{med} \cdot (c_y - o_y)/f_y \\
    z_{med}
\end{pmatrix}
$$

where $(o_x, o_y)$ is the principal point, $(f_x, f_y)$ is the focal length of camera, $R_{tilt}$ is the transform matrix between camera and world system.

3.3. Matching strategy

At training time, we need to determine positive examples and train the network accordingly. Although every detection is related to a fixed anchor box, we do not directly match 3D ground truth boxes to these anchor boxes because depth images are usually noisy and $z_{med}$ extracted as the center for a anchor box is possibly inaccurate, hence we resort to 2D ground truth box and 2D default boxes defined in SSD(shown in Figure[3] to solve this problem. SSD associates a set of default boxes( rectangles with different aspect ratios ) with each location in a feature map and regresses the bounding box offset values relative to a default box for 2D object detection. Unlike SSD, in our work, default boxes are used to determine which locations in feature maps are expected to output positive examples and have nothing with 3D anchor boxes generation and regression. No matter how many default boxes related to a location in a feature map are
3.4. Training objective

For each anchor box, we will predict the offsets relative to its centers, sizes and orientations along with the scores for the presence of each object category. Our training objective is similar with that of SSD but is generalized to 3D object detection. In the matching stage, we have found \( N \) positive anchor boxes, let \( x_{ij} \in \{0,1\} \) indicate whether the \( i^{th} \) 3D anchor box is matched with the \( j^{th} \) 3D ground truth of object class \( c \). Our objective loss function is a sum of the 3D bounding box regression loss and the classification loss, denoted as \( L_{reg} \) and \( L_{cls} \) respectively:

\[
L(x,c) = \frac{1}{N} (L_{cls}(x,p) + L_{reg}(x,l,g))
\]

The loss is set to 0 when \( N \) equal to 0. \( L_{reg} \) is a Smooth L1 loss\(^4\) between the predicted 3D box \( l_i \) and the 3D ground truth box \( g \). We regress to offsets for the set \( M = \{x_0, y_0, z_0, w, h, l, \theta\} \) which consists the center \((x_0, y_0, z_0)\) of the 3D anchor box, its width \(w\), length \(l\) and height \(h\) and its angle \(\theta\).

\[
L_{reg}(x,l,g) = \sum_{i \in Pos} \sum_{m \in M} x_{ij}^k \text{smooth}_{L1}(l_{ij}^m - \hat{g}_{ij}^m)
\]

\[
\hat{g}_{ij}^x = (g_{ij}^x - d_{ij}^x)/d_{ij}^w
\]

\[
\hat{g}_{ij}^y = (g_{ij}^y - d_{ij}^y)/d_{ij}^h
\]

\[
\hat{g}_{ij}^z = (g_{ij}^z - d_{ij}^z)/d_{ij}^l
\]

\[
\hat{g}_{ij}^\theta = \ln(g_{ij}^\theta/d_{ij}^\theta)
\]

\[
L_{cls} \text{ is the classification loss over predicted probabilities for multiple object categories}(p) \text{ and negative background is labeled as 0.}
\]

\[
L_{cls}(x,p) = - \sum_{i \in Pos} x_{ij}^k \ln(p_{ij}^c) - \sum_{i \in Neg} \ln(p_{ij}^0)
\]

4. Experiments

We evaluate our method on challenging SUN RGB-D dataset\(^21\). The dataset consists 5285 images for training and 5050 images for testing. And we also conduct experiments on NYUV2 dataset\(^15\) whose annotations are improved by \(^3\) for evaluation. NYUV2 dataset provides 785 images for training and 654 images for testing. Our network is implemented in Caffe\(^9\) on a Nvidia Titan X GPU. During testing, our model takes 0.22s per RGB-D image pair, which is the fastest in 3D object detection to our knowledge.

Evaluation Metric. We evaluate our 3D detection using 3D volume Intersection over Union(IoU) metric defined in \(^22\). A detection is considered as a true positive if its IoU with the ground truth is larger than 0.25. Similar to \(^23\) and \(^3\), we train our model for 19 object classes detection, and calculate mean Average Precision for performance comparison.

Comparison to the state-of-arts. Table\(^1\) and Table\(^2\) show the quantitative comparison with three state-of-the-arts for amodal 3D object detection in SUN RGB-D datasets. Our methods outperforms DSS\(^23\) by a large margin of 10.2 in mAP and is 88x faster. When compared with COG\(^19\)
and 2DD\cite{10}, we choose the results of the same 10 object classes as they reported and recompute mAP for them. The accuracy mAP is 3 higher than \cite{19} and 5.6 higher than \cite{10}. Moreover, we have significantly improved the speed and get 18 \times faster than the current level\cite{10}. Different from DSS, COG and 2DD that all align detected object with the estimated room, our method predicts orientations without Manhattan world assumption and so performs better when postures of objects vary greatly in indoor scene.

DSS, 2DD and COG extract features for point cloud that is restructured from depth images, the point cloud is usually sparse and noisy, especially for small objects(e.g. tv, monitor, lamp, garbage bin, pillow) or part-visible objects(e.g. dresser). In contrast, our approach explores dense and contiguity characteristics in the initial RGB and depth images, which avoids the impact of the sparsity of 3D data and is more robust to the aforementioned categories.

2DD uses single RGB images to detect objects in 2D image planes and then regresses 3D location and sizes for each detected 2D bounding box with the 3D information from the depth image, it ignored the complementary between texture and geometric features.

Moreover, we test our algorithm on the NYUV2 dataset to compare with another state-of-the-art method of Deng\cite{3}. The size of all input images is 300 \times 300 in our network, whereas Deng uses the original(730 \times 530) images. Results in Table\cite{3} show our approach achieves comparative performance to them with a much faster speed even with a smaller input image size.

We also supply some examples of our qualitative results in SUN RGB-D datasets and NYUV2 datasets, shown in Figure\cite{4} and Figure\cite{5}.

Next, several control experiments are conducted to understand contributions of each component in our model.

Multi-feature is better. To study the importance of the conjunction for features from different input images, we perform experiments with a different combination of the RGB image, the depth image and the HHA image\cite{7} (encodes depth image as Horizontal disparity, Height above ground and angle of local surface normal within estimated gravity direction). As is shown in the Table\cite{4}, if using only one image data as input, the RGB image, the depth image and the HHA image perform comparably. Combining the RGB and depth(or HHA) images can improve over individual images. This illustrates that depth information is better to detect objects with less clear texture(e.g. bathtub) and features learned from RGB and depth images are complementary.

Hierarchical fusion is better. To verify the improvement on accuracy from the hierarchical fusion structure, we conduct another comparison experiment with different feature fusion strategy: Fusion in only one stage. In the features extraction network, we concatenate Conv4-3-rgb and Conv4-3-d, followed by two 1 \times 1 convolutional layers. The fused

| Table 1. Evaluation for 19-class 3D object detection on SUN RGB-D test set. |
| Methods | | | | | | | | | | | mAP | Runtime |
|---------|---|---|---|---|---|---|---|---|---|---|---|---|
| DSS\cite{23} | 44.2 | 78.8 | 11.9 | 1.5 | 61.2 | 4.1 | 20.4 | 0.2 | 15.4 | 13.3 | 32.3 | 53.5 | 50.3 | 0.5 | 78.9 |
| Ours | 57.1 | 76.2 | 29.4 | 9.2 | 56.8 | 12.3 | 21.9 | 1.7 | 32.5 | 38 | 23.4 | 51.8 | 26.6 | 52.9 | 54.8 | 40.6 | 20.9 | 85.8 | 37.1 | 0.22s |

| Table 2. Evaluation for 10-class 3D object detection on SUN RGB-D test set. |
| Methods | | | | | | | | | | | mAP | Runtime |
|---------|---|---|---|---|---|---|---|---|---|---|---|---|
| COG\cite{19} | 58.26 | 63.67 | 31.8 | 62.17 | 45.19 | 15.47 | 27.36 | 51.02 | 51.29 | 70.07 | 47.63 | 10-30min |
| 2DD\cite{10} | 43.45 | 64.48 | 31.40 | 48.27 | 27.93 | 25.92 | 41.92 | 50.39 | 37.02 | 80.4 | 45.1 | 4.15s |
| Ours | 57.1 | 76.2 | 29.4 | 56.8 | 21.9 | 32.5 | 51.8 | 54.8 | 40.6 | 85.8 | 50.7 | 0.22s |

| Table 3. Evaluation for 19-class 3D object detection on NYUV2 RGB-D test set. |
| Methods | | | | | | | | | | | mAP | Runtime |
|---------|---|---|---|---|---|---|---|---|---|---|---|---|
| DSS\cite{3} | 62.3 | 81.2 | 23.9 | 3.8 | 58.2 | 24.5 | 36.1 | 0.0 | 31.6 | 27.2 | 28.7 | 2.0 | 54.5 | 38.5 | 40.5 | 55.2 | 43.7 | 1.0 | 76.3 | 36.3 | 19.55s |
| Deng\cite{10} | 36.1 | 84.5 | 40.6 | 4.9 | 46.4 | 44.8 | 33.1 | 10.2 | 44.9 | 33.3 | 29.4 | 3.6 | 60.6 | 46.3 | 58.3 | 61.8 | 43.2 | 16.3 | 79.7 | 40.9 | 100s |
| Ours | 48.9 | 84 | 26.1 | 2.2 | 50.7 | 44.4 | 32.8 | 9.2 | 29.1 | 30.8 | 32.2 | 11.2 | 64.1 | 40.2 | 64.1 | 57.8 | 39 | 9.1 | 79.4 | 39.7 | 0.22s |

| Table 4. An ablation study of different features: Performance are evaluated on SUN RGB-D test set. |
| Data | | | | | | | | | | | mAP | Runtime |
|-------|---|---|---|---|---|---|---|---|---|---|---|---|
| RGB | 42.9 | 68.2 | 22.5 | 10.4 | 48.6 | 10.6 | 0.4 | 10.6 | 26 | 33.8 | 17.1 | 13.5 | 42 | 16 | 45.3 | 46.2 | 34.6 | 14.8 | 80.1 | 31 | 0.21s |
| HHA | 57.1 | 74.5 | 18.9 | 3.7 | 55.3 | 6.6 | 17.3 | 0.1 | 26.7 | 33.1 | 17.4 | 13.8 | 48.2 | 22.7 | 48.1 | 52.2 | 37.9 | 15.2 | 78.9 | 33 | 0.21s |
| Depth | 53.7 | 74.4 | 19.1 | 9.7 | 53.8 | 4.5 | 19.2 | 1.3 | 22.8 | 27.6 | 23 | 8.4 | 47.5 | 22.5 | 40.2 | 51.1 | 38.9 | 7.6 | 79 | 31.8 | 0.21s |
| RGB+HHA | 59 | 78 | 26.1 | 7 | 57.6 | 11.7 | 22.4 | 0.3 | 32.8 | 40.4 | 18.3 | 12.9 | 51.2 | 26.8 | 51.1 | 54.8 | 42.1 | 19.9 | 84.6 | 36.7 | 0.22s |
| RGB+Depth | 57.1 | 76.2 | 29.4 | 9.2 | 56.8 | 12.3 | 21.9 | 1.7 | 32.5 | 38 | 23.4 | 12.9 | 51.8 | 26.6 | 52.9 | 54.8 | 40.6 | 20.9 | 85.8 | 37.1 | 0.22s |
feature map goes through the rest 5 convolutional layers of the subsampled VGG-16 network and the last 8 convolutional layers of our original network. As shown in Table 5, fusion in one stage performs worse than two stages by 9.5 in mAP.

| Hierarchically fused layers from: | mAP |
|----------------------------------|-----|
| Conv4-3s Conv7s                  |     |
| ✓                                | 27.6|
| ✓ ✓                              | 37.1|

Table 5. Effects of hierarchicall fusion.

5. Conclusion

We present a single end-to-end hierarchical fusion detector for amodal 3D object detection in RGB-D images. Our system use multi-scale feature maps to output detections. Experiments shows our method significantly outperforms the state-of-the-arts which use 3D representation or 2.5D representation, illustrating that using a single network can obtain higher accuracy and efficiency for 3D detection.

References

[1] P. Arbelaez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In CVPR. 2014.
[2] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In ECCV. 2016.
[3] Z. Deng and L. J. Latecki. Amodal detection of 3d objects: Inferring 3d bounding boxed from 2d ones in rgb-depth images. In CVPR. 2017.
[4] R. Girshick. Fast r-cnn. In ICCV. 2015.
[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and segmentation. In CVPR. 2014.
[6] S. Gupta, P. Arbelaez, R. Girshick, and J. Malik. Aligning 3d models to rgb-d images of cluttered scenes. In CVPR. 2015.
[7] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In ECCV. 2014.
[8] K. He, G. Gkioxari, P. Dollr, and R. Girshick. Mask r-cnn. In ICCV. 2017.
[9] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolut-sional architecture for fast feature embedding. In ACM. 2014.
[10] J. Lahoud and B. Ghanem. 2d-driven 3d object detection in rgb-d images. In ICCV. 2017.
[11] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In CVPR. 2017.
[12] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollr. Focal loss for dense object detection. In ICCV. 2017.
[13] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed. Indoor segmentation and support inference from rgbd images. In ECCV. 2016.
[14] W. Liu, R. Ji, and S. Li. Towards 3d object detection with bimodal deep boltzmann machines over rgbd imagery. In CVPR. 2015.
[15] S. Nathan, H. Derek, K. Pushmeet, and R. Fergus. Indoor segmentation and support inference from rgbd images. In ECCV. 2012.
[16] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In CVPR. 2016.
[17] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In CVPR. 2017.
[18] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In NIPS. 2015.
[19] Z. Ren and E. B. Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. In CVPR. 2016.
[20] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.
[21] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In ECCV. 2015.
[22] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In ECCV. 2014.
[23] S. Song and J. Xiao. Deep sliding shapes for amodal 3d object detection in rgb images. In CVPR. 2016.
[24] B. soo Kim, S. Xu, and S. Savarese. Accurate localization of 3d object from rgb-d data using segmentation hypothesis. In ICCV. 2013.
[25] B. soo Kim, S. Xu, and S. Savarese. Holistic scene understanding for 3d object detection with rgbd cameras. In ICCV. 2013.
Figure 4. Examples of detected objects in SUN RGB-D datasets. We show detections with scores higher than a threshold (0.55). The 1st line and 2nd line show the input RGB and depth images. Detected boxes are shown in the 3rd line. Ground truth boxes are shown in the 4th line.

Figure 5. Examples of detected objects in NYUV2 datasets. We show detections with scores higher than a threshold (0.55). The 1st line and 2nd line show the input RGB and depth images. Detected boxes are shown in the 3rd line. Ground truth boxes are shown in the 4th line.