Combining Semantic and Geometric Features for Object Class Segmentation of Indoor Scenes

Farzad Husain, Hannes Schulz, Babette Dellen, Carme Torras, and Sven Behnke

***Abstract*—Scene understanding is a necessary prerequisite for robots acting autonomously in complex environments. Low-cost RGB-D cameras such as Microsoft Kinect enabled new methods for analyzing indoor scenes and are now ubiquitously used in indoor robotics. We investigate strategies for efficient pixelwise object class labeling of indoor scenes that combine both pretrained semantic features transferred from a large color image dataset and geometric features, computed relative to the room structures, including a novel distance-from-wall feature, which encodes the proximity of scene points to a detected major wall of the room. We evaluate our approach on the popular NYU v2 dataset. Several deep learning models are tested, which are designed to exploit different characteristics of the data. This includes feature learning with two different pooling sizes. Our results indicate that combining semantic and geometric features yields significantly improved results for the task of object class segmentation.**

***Index Terms*—Semantic scene understanding, categorization, segmentation.**

I. Introduction

UNDERSTANDING complex scenes has gained much in importance as the applications of service robots for homes and offices is increasing. Dense structural description of the indoor scenes is vital for performing accurate analyses. To serve this purpose, the usage of RGB-D cameras is becoming ubiquitous, as they provide color images and dense depth maps of the scene. Tasks such as “picking up objects” and “planning manipulation actions” [1]–[3] are simplified once the precise location of objects in the scene is identified. One way to facilitate object localization is to perform pixelwise semantic labeling of the scene [4]. This involves identification and labeling of different object classes based on the semantics of the scene. Recently, Convolutional Neural Networks (CNNs) have shown impressive results for semantic labeling [5], [6]. The architecture of the CNN together with the used input features are important factors determining the quality of learned scene semantics. This work addresses these aspects with proposed novelties to improve the accuracy in semantic labeling.

The accuracy of semantic labeling depends on the consistency of the model description of the scenes under naturally occurring variations such as camera pose, lighting conditions, and position of the objects. Object surfaces such as “floors” and “walls” are easy to identify and segment, as they usually follow a similar pattern. Movable objects such as “small structures”, “furniture” and “wall hangings” are harder to identify. This is also evident by comparing the individual class labeling accuracies of different approaches (see Table V). In order to mitigate this, [7] proposed a depth normalization scheme where the furthest point is assigned a relative depth of one. Using the depth normalization scheme improved the segmentation results, which motivated us to explore further the explicit modeling of the indoor environments.

For indoor environments, the height of an object above the ground plane gives a strong indication for the corresponding object class. For example, a “tv” is usually placed at a higher position from the ground plane than a “bed” or a “sofa”. [8] showed that this feature improves the results for object class segmentation problem. We use the HHA representation of [9], which encodes the depth data into three channels, height above the ground, horizontal disparity, and angle of the normals with the inferred gravity direction. This encoding has been demonstrated to give good results for object detection and labeling tasks.

As the indoor scenes are always captured in a confined environment, i.e., in the presence of surrounding walls, we propose to exploit this structural information. Based on a bounding hull heuristic developed by [7], we construct a novel feature that we name *distance-from-wall,* which indicates the proximity of scene points to some major detected room wall. This feature is defined as the minimum point-to-plane distance between a scene point and the planes detected at the outermost region of a scene, saturated beyond a certain distance threshold. Figure 1 shows an example of the computed distances from two walls for a sample taken from the NYU v2 dataset [10]. It can be observed that—compared to the original depth image in Fig. 1(b)—the objects closer to the wall become more distinguishable after thresholding in Fig. 1(d). Wall distances greater than a fixed threshold are clipped, since we can assume that their precise wall distance is not informative. The proposed distance-from-wall feature facilitates the detection of objects such as “windows”, “wall hangings” and “tv” which are usually found in close proximity to the walls. As a result, we observed an improvement in the overall object class segmentation accuracies.

One issue arising when training CNNs on RGB-D inputs is the limited size of the available RGB-D data sets. It has been

Manuscript received August 31, 2015; accepted January 28, 2016. Date of publication February 29, 2016; date of current version April 11, 2016. This paper was recommended for publication by Associate Editor H. Kjellstrom and Editor J. Kosecka upon evaluation of the reviewers’ comments. This work was supported in part by the CSIC project MANIPlus (201350E102) and in part by the project RobInstruct (TIN2014-58178-R).

F. Husain and C. Torras are with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona 08028, Spain (e-mail: shusain@iri.upc.edu; torras@iri.upc.edu).

H. Schulz and S. Behnke are with the Institute of Computer Science VI, University of Bonn, Bonn 53113, Germany (e-mail: schulz@ais.uni-bonn.de; behnke@ais.uni-bonn.de).

B. Dellen is with the RheinAhrCampus der Hochschule Koblenz, Remagen 53424, Germany (e-mail: dellen@hs-koblenz.de).

Digital Object Identifier 10.1109/LRA.2016.2532927

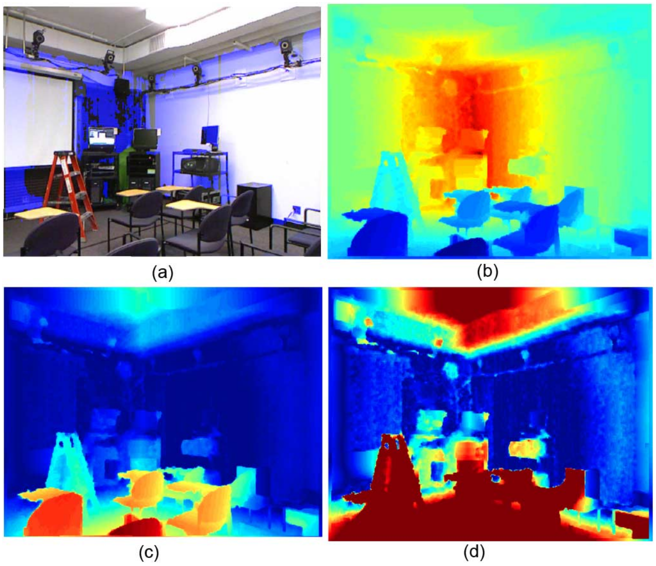

Fig. 1. Illustrating the distance-from-wall feature. (a) Color image with its blue channel replaced with the binary mask based on the bounding hull heuristic from [7]. The walls are detected from the bounding hull region. (b) Depth image, (c) and (d) show the minimum distance of each point in the scene from the walls before and after thresholding, respectively.

shown that semantic CNN features obtained by training classification tasks on large data sets can be transferred to related tasks, such as object detection [11], [12], subcategorization, domain adaptation, scene recognition [13], attribute detection, and instance retrieval [12]. This transfer of pretrained semantic features proved also to be useful for RGB-D object categorization, instance recognition and pose estimation [14], and for the task of RGB-D object class segmentation [5].

Our objective is to build a robust CNN architecture that predicts a semantic label for each pixel in the scene. We train CNN models with two different sets of pooling sizes end-to-end with the color image, the HHA encoding [9], and our proposed distance-from-wall feature. We transfer pre-trained semantic features from a CNN model trained on the ImageNet dataset[1]. The parameters of the networks are learned so that they minimize the pixelwise cross-entropy loss between the predicted labels and the ground truth. We demonstrate the effectiveness of our approach by evaluating each of the proposed modalities separately and also comparing it with the other state-of-the-art approaches on the widely used NYU v2 dataset [10].

The main contributions of this letter are:

- proposal of a new feature termed *distance-from-wall*,
- a novel CNN architecture using two different pooling sizes, yielding improved results, and
- evaluation and comparison with other state-of-the-art approaches, showing improved overall performance.

II. Related Work

The conventional approach to semantic labeling is carried out in multiple stages [4], [15]–[19]. This involves presegmenting the scene into smaller patches followed by feature extraction and classification. The final classification results are dependent on the results obtained at each stage of the approach. Another way is to train a deep CNN in an end-to-end fashion, i.e., directly from input pixels to semantic labels [5], [8], [20].

[4] performed a multiscale segmentation of image and point cloud followed by extraction of feature vectors. The feature vectors were classified separately per modality by a random forest (RF) and the classification results were fused and further refined using a pairwise CRF (Conditional Random Field) to enforce spatial consistency. Handcrafted features such as "area," "diameter" and "orientation" were used to identify different features. However, feature learning from combined raw data and hand-crafted features often yields better results as it exploits both the hidden cues and human knowledge. [19] is an example of such an approach, where the authors combine a height-above-ground feature with pixel-wise RF classification [21] and learn binary potentials between superpixels based on manually designed features.

[22] and [17] presegment a scene and afterwards build a model that exploits their semantic relations. [22] used a CRF-based model to relate pixel-wise and pair-wise observations to labels for hierarchical semantic labeling. [17] used a randomized decision forest for semantic segmentation, where the results were further refined using a dense CRF. Similarly, segmentation followed by a random forest classification to initialize the unary potentials of a CRF was proposed by [18].

[8] trained a deep CNN using image patches as input to the network, where the patch size was adjusted according to the measured depth of the patch center. This increased scale invariance and led to improved object class segmentation results.

In order to increase scale invariance in deep CNNs, [20] and [5] used input at multiple scales. A simplified version of the histogram of oriented gradient (HOG) descriptor applied to the depth channel provided depth information to the CNN of [23].

To increase the spatial accuracy of semantic segmentation, [5], [6] proposed two different CNN models. [5] divided a CNN into three sub-networks which gradually predicted output from a coarse to fine level. The network is initialized with ImageNet-trained AlexNet [24]. Additionally, loosely related computer vision tasks—estimating depths and surface normals—were optimized by adjusting the loss function. [6] combined upsampled predictions from intermediate layers with the final layer which lead to more refined results. A single-image classification network was adapted to a fully convolutional network and fine tuned for semantic segmentation.

Another line of research is to exploit temporal integration from RGB-D video sequences. For example, [25] perform RGB-D SLAM to estimate the camera motion and aggregate semantic segmentations from multiple views in 3D. [26] directly train hierarchical recurrent convolutional neural networks on object class segmentation from RGB-D video. In this work, we do not address temporal integration and process only individual RGB-D frames.

[1] http://www.image-net.org/

III. Problem Formulation

Given a color image and a dense depth map X of a scene, our goal is to obtain a label $\hat{y}_p \in \mathscr{C}$ for each pixel location $\mathrm{x}_p \in X$ that corresponds to the object class at the pixel location.

In our task, we deal with natural indoor scenes which are usually unbalanced with respect to the size and number of objects. For example, the number of pixels belonging to the floor class is much greater than that of those in the furniture class. Hence we use a weighted, multiclass cross entropy loss function [5]:

$$L = -\sum_{i \in X} \sum_{b \in \mathscr{C}} \alpha_b c_{i,b} \ln(\hat{c}_{i,b}),$$

where

$$\alpha_b = \text{median-freq}(\mathscr{C})/\text{freq}(b),$$

$\hat{c}_{i,\cdot}$ is the predicted class distribution at location i, and $c_{i,\cdot}$ is the respective ground truth class distribution. The factor α weighs each class according to its frequency with which it appears in the training set, and $\text{median-freq}(\mathscr{C})$ is the median of all class frequencies.

IV. Approach

We use a convolutional neural network in two stages, as illustrated in Figs. 2 and 3. In the first stage, we train the network with two different sets of pooling-operator sizes. In the second stage, we concatenate the feature maps of the last layer of the two networks from the previous stage and train the network again. This two-stage approach yields refined results (see Sec. IV-A). Finally, we enrich the set of concatenated features with geometric wall proximity information from the distance-from-wall feature (Sec. IV-B).

A. Network Architecture

Our network takes three inputs, i.e., color image, HHA encoding, and distance-from-wall feature. The color image is passed through a stack of five convolutional layers. The HHA encoding and distance-from-wall feature are processed by only four and three convolutional layers, respectively. This scheme is illustrated in Fig. 2. The weights of the first two layers for color image and the first layer for HHA encoding are transferred from the OverFeat network [27]. This network was designed by the CILVR Lab at NYU[2] and was trained on the ImageNet dataset for color image categorization. The weights of these two layers are kept fixed during our training and serve as semantic feature extractor. The network also contains pooling layers, a dropout layer and is divided into four configurations as described in Table I. We train the network with two different pool sizes after the first convolutional layer. Hence, the final feature map size in Configuration A is either four times or eight times smaller, depending on the pool size. The same applies to configuration B. Configurations A and B do not need training as the filters are transferred from the OverFeat network. Configuration C contains only the distance-from-wall feature. The outputs from Configurations A, B and C are concatenated and fed to Configuration D, which we train using the labeled training examples.

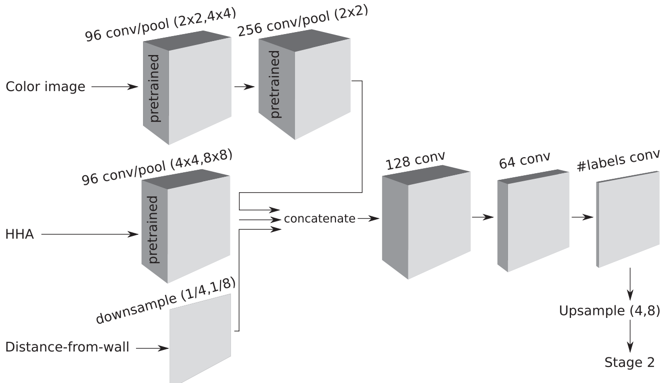

Fig. 2. First stage of our proposed model architecture for object class segmentation. Inputs are the color image, the HHA encoding [9], and the distance-from-wall feature (Sec. IV-B). Layer 1 (96 feature maps) and Layer 2 (256 feature maps) for color image and Layer 1 for HHA (96 feature maps) have filters pretrained on the ImageNet dataset which are not changed during training. Afterwards, the feature maps together with the distance-from-wall feature are concatenated and fed to a 3-layer trainable CNN. The output layer has the same number of feature maps as the number of object class labels. The output maps for the two different pooling sizes are used as input for Stage 2.

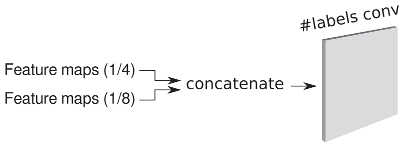

Fig. 3. Second stage of our proposed CNN architecture for object class segmentation. The output feature maps obtained after training the CNN in the first stage (Fig. 2) are used as input. The pooling operation in Stage 1 is done with two sets of sizes. Hence, two sets of output feature maps are obtained which are here referred to as (1/4) and (1/8). These maps are concatenated and used as an input for training a single-layer network. The output layer for this network has the same number of feature maps as the number of object class labels.

TABLE I
Convolutional Neural Network Configurations

Layer	Filter Size	Stride	No. of maps	Map size
Configuration A				
Input (color)	-	-	3	image_size
Conv1	11×11	1	96	image_size
Pool1	2×2, 4×4	2,4	96	image_size/(2,4)
Conv2	11×11		256	image_size/(2,4)
Pool2	2×2	2	256	image_size/(4,8)
Configuration B				
Input (HHA)	-	-	3	image_size
Conv1	11×11	1	96	image_size
Pool1	4×4, 8×8	4,8	96	image_size/(4,8)
Configuration C				
Input (Dist-from-wall)	-	-	1	image_size/(4,8)
Configuration D				
Input (Conf. A+B+C)	-	-	256+96+1	image_size/(4,8)
Conv1	11×11	1	128	image_size/(4,8)
Dropout (0.25)	-	-	-	-
Conv2	11×11	1	64	image_size/(4,8)
Conv3	11×11	1	No. of classes	image_size/(4,8)
Upsample	-	-	No. of classes	image_size

[2] http://cilvr.nyu.edu/doku.php

The networks are trained for two different sets of pool sizes. The final output from Configuration D is concatenated and trained again as shown in Fig. 3 before the upsampling. A single convolutional layer with the number of filters equal to the number of object classes and a size of $(3{\times}3)$ is learned. By concatenating networks with different pool sizes, we exploit invariance to local deformations while preserving information on spatial location, which increases the segmentation accuracy.

B. Distance-From-Wall

We have devised a simple yet robust heuristic to detect the walls at the outermost region of an indoor scene. This involves segmenting the point cloud and picking up those segments that lie within the outermost boundary region. Afterwards, the two largest segments are selected and a planar surface model is used to compute the distance from each point to those planar surfaces. The steps are illustrated in Fig. 4 and are detailed below.

1) Segment the Point Cloud: We partition the point cloud into surface segments using two different approaches. The first approach fits planes to the point cloud using RANSAC (see Fig. 4(c)). The second approach segments the 3D points based on quadratic surface fitting as described by [28] (see Fig. 4(d)). We use these two different approaches, because depending on the complexity of the scene, one approach performed better than the other. The method by [28] performed better in complex scenes containing mostly non-planar surfaces, whereas RANSAC performed better when the scene was dominated by planar surfaces.

2) Detect the Outermost Boundary: In order to detect the outermost boundary, we use the technique described by [7], i.e., if the depth of a point is within 4% of the maximum depth within each column of the image grid then it is marked as the boundary region. Fig. 4(a) shows the boundary region shaded in blue color. We only take the segments that lie in the outermost boundary as possible candidates for walls as shown in Figs. 4(e) and (f).

3) Select the Largest Two Segments When Viewed From the Top: To generate an approximate top view, we assume that the vertical camera-axis always points towards the upwards direction. We select at most two segments from the segmented point cloud that have the largest triangular width and are not coplanar, when viewed from the top. Afterwards we select the segments that have the largest sum of widths. Hence, among Figs. 4(g) and (h), the two segments from Fig. 4(g) are selected.

4) Computing the Distance From the Wall Feature: Once the two segments are selected, we compute for all scene points the minimum point-to-plane distance using a planar-surface equation. Since in each scene we have at most two planes yielding a point-plane distance value each (see Figs. 4(i) and (j)), we take the minimum to get the final distance-from-wall (see Fig. 4(k)),

$$D_{x\in\text{pixel}} = \min(\pi_x^1, \pi_x^2),$$

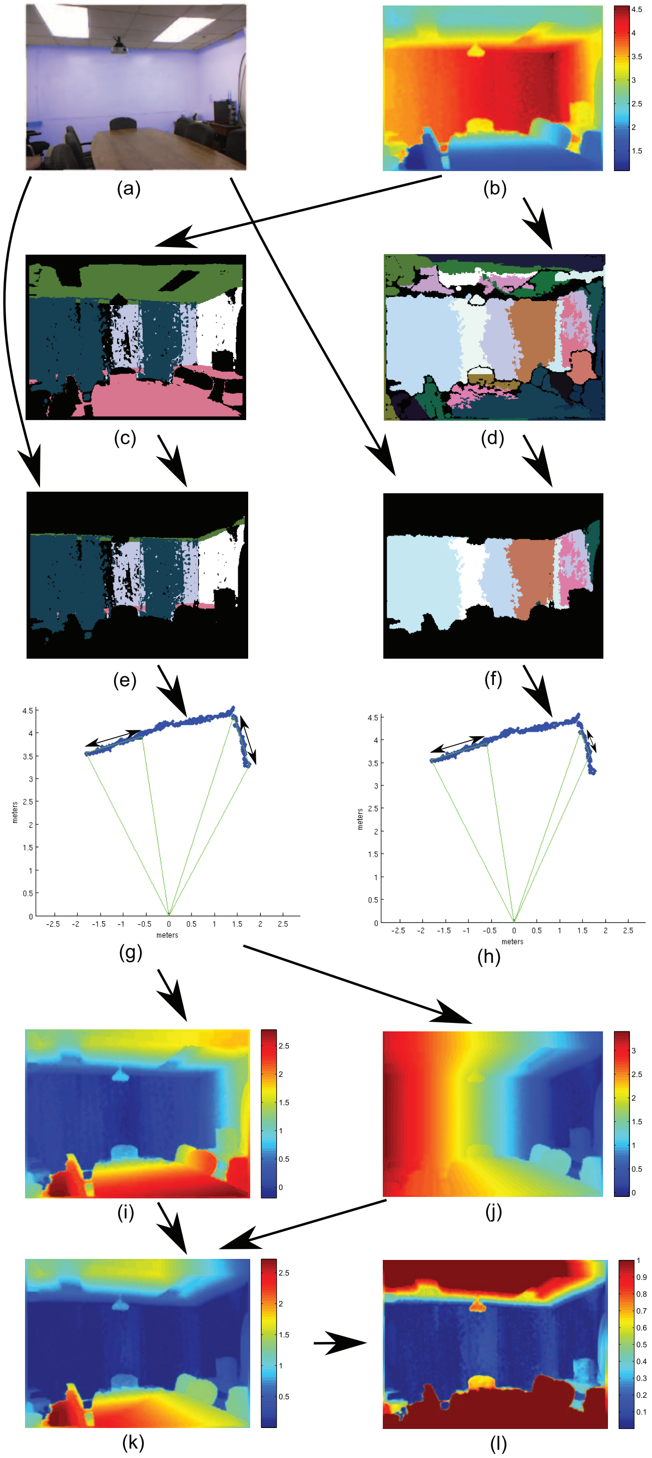
(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l)

Fig. 4. Illustrating the procedure for the computation of the distance-from-wall feature. (a) Color image with the boundary region shaded in blue, (b) depth image, (c) segmentation using RANSAC, (d) segmentation of depth image using the method proposed by [28], (e) and (f) selecting only the segments from the boundary region, (g) and (h) are top views of the point clouds based on the two largest single-colored plane segments from (e) and (f), respectively, (g) is selected because it has the largest sum of the widths (the double sided arrows in (g) and (h)) of the enclosed segments, (i) and (j) are the distances of each point from the two planes detected based on the two segments in (g), (k) is the minimum distance for each point from both planes and (l) is the final distance feature after thresholding. The color image is shown for illustration only and not used in computing the distance feature.

where π_x^1 and π_x^2 are the point to plane distances of point x to plane π^1 and π^2, respectively. We use a threshold value of one meter. All points whose value exceeds the threshold are clipped as shown in Fig. 4(l). The rationale behind this particular value is that the objects within a 1 meter proximity of walls are likely to have depth readings differing slightly, but distinctly from the wall itself, such as "lamps", "wall hangings", and "cupboard". Such objects are highlighted from the rest of the scene after applying this threshold. Objects further away than one meter do not typically occur in specific distances to the wall and the feature becomes meaningless.

V. Experiments

In this section, we describe the evaluation results on the NYU v2 dataset. We use the four object classes [10] in Sec. V-A and the 13 object classes [29] in Sec. V-B.

The dataset contains a total of 1449 samples of different indoor scenes. We use the training/test split as provided by the dataset authors. The parameters for gradient descent, i.e., the learning rate, momentum and the number of iterations are adjusted by first separating 10% of the training examples and using them for validation.

In order to evaluate our approach, we use two common measures of performance:

- average pixel accuracy $\sum_i n_{ii} / \sum_i t_i$ and
- average class accuracy $(1/n_{cl}) \sum_i (n_{ii}/t_i)$,

where n_{ii} is the number of correctly classified pixels for class i, t_i is the total number of pixels for class i, and n_{cl} is the number of classes.

In order to get an insight on the benefits of our proposed models, we evaluate different aspects separately. This includes the network from first stage, referred to as net-ABCD and without the distance-from-wall feature as net-ABD, the network from second stage referred to as net-ABCD-combined and also without the distance-from-wall feature as net-ABD-combined. We distinguish between the two upsampling factors for the different pooling sizes in the first stage as U4 and U8.

A. NYU v2 With Four Classes

We present the results for the NYU v2 dataset, using four classes as defined by Silberman and Fergus [10]. Table II shows a comparison of the individual labeling accuracies for each of the four classes. We get better results after combining the two networks, in the struct class (81.9% vs. 81.6% and 79.9%) and furniture class (72.8% vs. 66.6% and 72.0%). The Tables III and V are ordered according to the increasing level of accuracy corresponding to the Tables II and IV respectively. The net-ABCD-combined shows improved overall results as compared to net-ABCD-U4 and net-ABCD-U8. Our results are competitive to the cascaded multi-scale CNNs approach [5]. However, because of cascading different networks, the latter model required training in two steps. Additionally, the feature learning at multiple scales approach in [5] seems to be beneficial, when compared to our single scale model.

TABLE II
Individual Classes of NYU v2 (Four Classes)

Method	Accuracy (%)			
	floor	struct	furniture	prop
Couprie et al. [29]	87.3	86.1	45.3	35.5
Khan et al. [30]	87.1	**88.2**	54.7	32.6
Stückler et al. [25]	90.7	81.4	68.1	19.8
Müller and Behnke [19]	94.9	78.9	71.1	42.7
Wolf et al. [18]	**96.8**	77.0	70.8	45.7
net-ABD-U4 (without dist-from-wall)	96.6	82.5	62.8	63.2
net-ABD-U8 (without dist-from-wall)	94.9	78.1	75.9	60.5
net-ABCD-U4	95.9	79.9	72.0	63.5
net-ABD-combined (w/o dist-from-wall)	94.8	78.2	69.2	69.9
net-ABCD-U8	94.8	81.6	66.6	**70.3**
Eigen and Fergus [5] (AlexNet)	93.9	87.9	**79.7**	55.1
net-ABCD-combined	95.0	81.9	72.8	67.2

TABLE III
Overall Performance on NYU v2 (Four Classes)

Method	Accuracy (%)	
	class avg.	pixel avg.
Couprie et al. [29]	64.5	63.5
Khan et al. [30]	69.2	65.6
Stückler et al. [25]	70.9	67.0
Müller and Behnke [19]	72.3	71.9
Wolf et al. [18]	72.6	74.1
net-ABD-U4 (without distance-from-wall)	76.3	74.6
net-ABD-U8 (without distance-from-wall)	77.4	76.4
net-ABCD-U4	77.8	76.5
net-ABD-combined (without distance-from-wall)	78.2	76.5
net-ABCD-U8	78.3	76.4
Eigen and Fergus [5] (AlexNet)	79.1	**80.6**
net-ABCD-combined	**79.2**	78.0

TABLE IV
Overall Performance for the NYU v2 (13 Classes)

Method	Accuracy (%)	
	class avg.	pixel avg.
Couprie et al. [29]	36.2	52.4
Hermans et al. [17]	48.0	54.2
net-ABD-U4 (without distance-from-wall)	50.9	58.3
net-ABD-U8 (without distance-from-wall)	54.9	61.1
net-ABCD-U4	56.2	62.5
Wolf et al. [18]	56.9	64.9
net-ABD-combined (without distance-from-wall)	58.1	64.1
Eigen and Fergus [5] (AlexNet)	59.4	**70.5**
net-ABCD-U8	59.5	65.7
net-ABCD-combined	**59.6**	66.4

Figure 5 shows selected examples from the test set of the NYU v2 dataset. Results obtained with and without the distance-from-wall feature are shown. It can be observed that the furniture (brown color) and the prop (pink) class are better segmented. For example, in Fig. 5(2c) it can be seen that the "tv" is easily distinguishable in the distance-from-wall feature and it is better segmented (see Fig. 5(2d) and (2e)) when this feature is used.

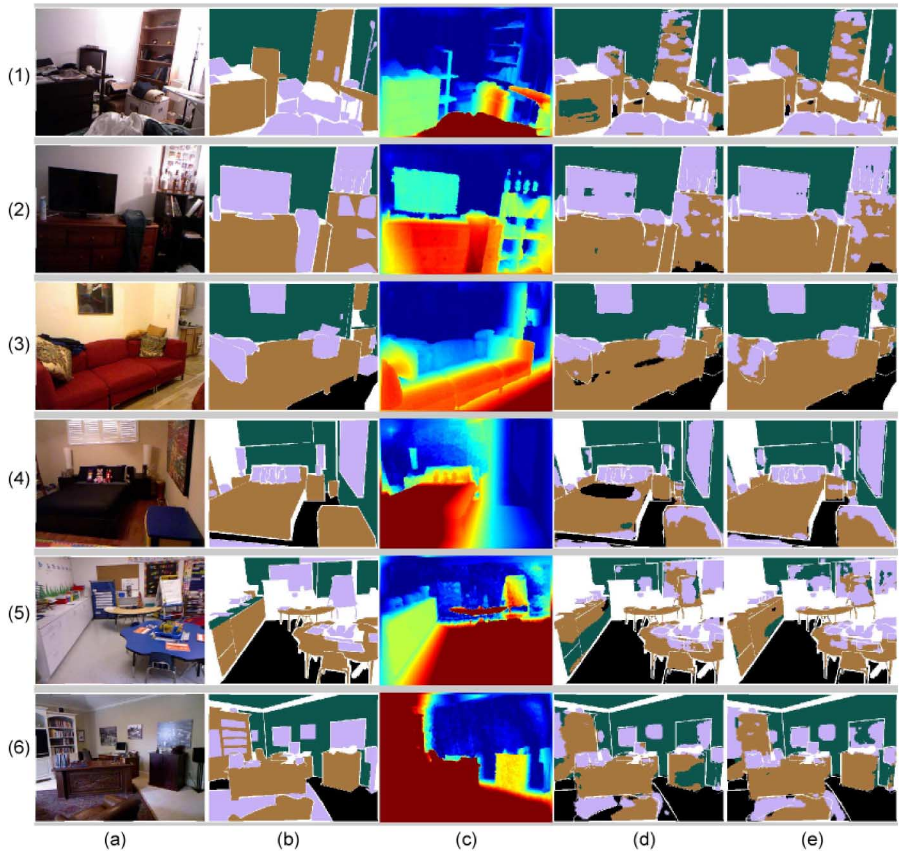

Fig. 5. Selected test set examples showing the improved labeling after using our proposed feature, (a) color image, (b) ground truth labeling, (c) distance-from-wall, (d) predicted labels without distance-from-wall and (e) predicted labels with distance-from-wall. White color in Figs. (b), (d) and (e) represents the unknown label.

TABLE V
INDIVIDUAL CLASS LABELING ACCURACY FOR NYU V2 (13 CLASSES)

	Accuracy (%)												
Method	bed	books	ceiling	chair	floor	furniture	objects	picture	sofa	table	tv	wall	window
Couprie et al. [29]	30.3	31.7	33.2	44.4	68.0	28.5	10.9	38.5	25.8	18.0	18.8	**89.4**	37.8
Hermans et al. [17]	**68.4**	45.4	83.4	41.9	91.5	37.1	8.6	35.8	28.5	27.7	38.4	71.8	46.1
net-ABD-U4 (without distance-from-wall)	21.1	22.3	94.4	19.5	93.2	69.9	49.2	68.9	27.3	41.6	39.5	62.3	53.4
net-ABD-U8 (without distance-from-wall)	48.0	32.3	93.3	37.1	94.7	77.4	48.0	45.1	44.3	41.7	30.6	58.3	71.2
net-ABCD-U4	52.5	42.5	**93.3**	32.1	93.3	**67.2**	48.4	63.5	**53.2**	34.1	25.8	66.2	58.0
Wolf et al. [18]	58.2	45.3	92.8	57.7	**97.5**	57.3	37.4	32.3	49.8	**51.8**	26.4	74.4	43.2
net-ABD-combined (without distance-from-wall)	42.2	37.6	92.9	34.3	94.7	63.9	50.8	**70.0**	41.5	53.6	42.5	70.4	61.0
Eigen and Fergus [5] (AlexNet)	57.7	39.9	77.6	**71.1**	95.9	64.1	**54.9**	49.4	45.8	45.0	25.2	87.9	57.6
net-ABCD-U8	51.9	**46.5**	91.8	41.0	94.8	66.7	44.9	61.7	49.4	49.0	41.6	73.5	60.5
net-ABCD-combined	44.1	42.9	92.7	38.5	95.2	66.6	53.9	63.5	46.8	49.4	**42.6**	74.7	**63.5**

B. *NYU v2 With 13 Classes*

We present the results for the NYU v2 dataset, using 13 classes as defined by Couprie et al. [29]. Table V shows a comparison of the individual labeling accuracies for each of the 13 classes. It can be observed that our approach performs better for the smaller object categories that are usually closer to the wall such as “picture”, “tv”, and “window”. Table IV shows a comparison of the average class accuracies and the average pixel accuracies corresponding to the results in Table V. Similar behavior can be observed, as was the case of four classes, i.e., better results with net-ABCD-combined as compared to net-ABCD-U4 and net-ABCD-U8.

VI. CONCLUSIONS

We have proposed a new model based on deep learning for the task of semantic object class segmentation. The model employs a multi-pooling architecture and takes the color image, the HHA encoding, and a novel distance-from-wall feature as input. The distance-from-wall feature is able to successfully highlight objects that are in close vicinity to the walls. This enables the deep learning model to learn from a more detailed representation of a scene. Our extensive evaluation on the NYU v2 dataset demonstrates the effectiveness of our proposed feature by showing better overall object class segmentation results. This letter shows that features of relative position are helpful for semantic segmentation. In the future, we plan to increase invariance to camera position even further by taking into account additional properties such as 3D room layout.

REFERENCES

[1] A. Dragan, N. Ratliff, and S. Srinivasa, "Manipulation planning with goal sets using constrained trajectory optimization," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2011, pp. 4582–4588.

[2] D. Martínez, G. Alenyà, and C. Torras, "Planning robot manipulation to clean planar surfaces," *Eng. Appl. Artif. Intell.*, vol. 39, pp. 23–32, 2015.

[3] F. Husain, A. Colome, B. Dellen, G. Alenya, and C. Torras, "Realtime tracking and grasping of a moving object from range video," in *Proc. Int. Conf. Rob. Autom. (ICRA)*, 2014, pp. 2617–2622.

[4] R. Zhang, S. Candra, K. Vetter, and A. Zakhor, "Sensor fusion for semantic segmentation of urban scenes," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 1850–1857.

[5] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 2650–2658.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 3431–3440.

[7] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *Proc. ICCV Workshop 3D Represent. Recognit.*, 2011, pp. 601–608.

[8] H. Schulz, N. Höft, and S. Behnke, "Depth and height aware semantic RGB-D perception with convolutional neural networks," in *Proc. Eur. Conf. Neural Netw. (ESANN)*, 2015, pp. 463–468.

[9] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. Europ. Conf. Comput. Vis. (ECCV)*, 2014, vol. 8695, pp. 345–360.

[10] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2012, pp. 746–760.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," arXiv: 1311.2524, 2013.

[12] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. CVPR DeepVision Workshop*, 2014, pp. 806–813.

[13] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2014, pp. 647–655.

[14] M. Schwarz, H. Schulz, and S. Behnke, "RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features," in *Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 1329–1335.

[15] C. Cadena and J. Kosecka, "Semantic segmentation with heterogeneous sensor coverages," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2014, pp. 2639–2645.

[16] X. Xiong, D. Munoz, J. Bagnell, and M. Hebert, "3-D scene analysis via sequenced predictions over points and regions," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2011, pp. 2609–2616.

[17] A. Hermans, G. Floros, and B. Leibe, "Dense 3D semantic mapping of indoor scenes from RGB-D images," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2014, pp. 2631–2638.

[18] D. Wolf, J. Prankl, and M. Vincze, "Fast semantic segmentation of 3D point clouds using a dense CRF with learned parameters," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 4867–4873.

[19] A. Müller and S. Behnke, "Learning depth-sensitive conditional random fields for semantic segmentation of RGB-D images," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2014, pp. 6232–6237.

[20] C. Farabet, C. Couprie, L. Najman, and Y. Lecun, "Scene parsing with multiscale feature learning, purity trees, and optimal covers," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2012, pp. 575–582.

[21] H. Schulz, B. Waldvogel, R. Sheikh, and S. Behnke, "CURFIL: Random forests for image labeling on GPU," in *Proc. 10th Int. Conf. Comput. Vis. Theory Appl. (VISAPP)*, 2015, pp. 156–164.

[22] C. Wu, I. Lenz, and A. Saxena, "Hierarchical semantic labeling for task-relevant RGB-D perception," in *Robotics: Science and Systems (RSS)*, Berkeley, CA, USA, 2014, pp. 1–9.

[23] N. Höft, H. Schulz, and S. Behnke, "Fast semantic segmentation of RGB-D scenes with GPU-accelerated deep neural networks," in *Proc. Ger. Conf. Artif. Intell. (KI)*, 2014, vol. 8736, pp. 80–85.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.

[25] J. Stückler, B. Waldvogel, H. Schulz, and S. Behnke, "Dense real-time mapping of object-class semantics from RGB-D video," *J. Real-Time Image Process.*, vol. 10, 2015, pp. 599–609.

[26] M. S. Pavel, H. Schulz, and S. Behnke, "Recurrent convolutional neural networks for object-class segmentation of RGB-D video," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2015, pp. 1–8.

[27] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–16.

[28] F. Husain, B. Dellen, and C. Torras, "Consistent depth video segmentation using adaptive surface models," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 266–278, Feb. 2015.

[29] C. Couprie, C. Farabet, L. Najman, and Y. LeCun, "Indoor semantic segmentation using depth information," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2013, pp. 1–8.

[30] S. Khan, M. Bennamoun, F. Sohel, and R. Togneri, "Geometry driven semantic labeling of indoor scenes," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 679–694.