Differential Geometry boosts Convolutional Neural Networks for Object Detection

Chu Wang Kaleem Siddiqi
School of Computer Science and Centre for Intelligent Machines,
McGill University, Canada
{chuwang,siddiqi}@cim.mcgill.ca

Abstract

Convolutional neural networks (CNNs) have had dramatic success in appearance based object recognition tasks such as the ImageNet visual recognition challenge [8]. However, their application to object recognition and detection thus far has focused largely on intensity or color images as inputs. Motivated by demonstrations that depth can enhance the performance of CNN-based approaches [2][5], in this paper we consider the benefits of also including differential geometric shape features. This elementary idea of using zeroth order (depth), first-order (surface normal) and second-order (surface curvature) features in a principled manner boosts the performance of a CNN that has been pretrained on a color image database. Notably, in an object detection task involving 19 categories we achieve 39.30% accuracy on the NYUv2 dataset, which is a 10.4% improvement over the current state-of-the-art accuracy of 35.6% using the method in [5]. In the simpler scenario of turntable style object recognition, our experiments on the University of Washington (UW) RGB-D dataset yield an accuracy of 88.7% correct recognition over 51 object categories, where the best competing result is 87.5% [2]. Taken together, our results provide strong evidence that the abstraction of surface shape benefits object detection and recognition.

1. Introduction

The emergence of Kinect like sensors has the potential to revolutionize computer vision applications, since they provide registered appearance (color) and depth (RGB-D) images. Unlike color images, depth maps are robust to variations in lighting, and they provide strong shape cues along with information to separate object boundaries via depth discontinuities. Recent work in pattern recognition with 3D point cloud data demonstrates the promise of using depth information along with appearance for object detection and recognition [2][5][9][13][14].

A parallel trend in the computer vision community has been the use of deep learning with convolutional neural networks (CNNs) [10], for category level object recognition and detection, with demonstrated success on 2D image databases such as ImageNet [8] and on CIFAR-10 [6]. By combining region proposal algorithms with CNNs, a R-CNN algorithm by Girshick et al. [3] achieved the best reported accuracy thus far on the ImageNet detection challenge. Gupta et al. have extended this R-CNN to incorporate depth data from point clouds using hand designed depth features [5], yielding the best reported accuracy to date in an object detection task on the NYUv2 RGB-D dataset [11].

While depth information is informative, the manner in which it has been used thus far is somewhat ad-hoc. For example, Socher et al. [13] tackle object recognition on the UW dataset [9] using recursive CNNs combining appearance and depth, but with no additional shape cues. Gupta et al.'s approach in [5] uses hand crafted (HHA) features such as the angle between surface normals and an estimated direction of gravity assumed to be orthogonal to the ground plane. Bo et al. apply a K-SVD method on depth and surface normal features in the UW dataset [2], but no curvature features are incorporated. The sliding shape method of [14] utilizes eigenvalues from a point cloud covariance matrix along with surface normals as shape features for an SVM classifier.

In the present article we abstract surface differential geometry for 3D object detection in a principled manner, i.e., we use zeroth order (depth), first-order (surface normal) and second-order (curvature) estimated from point cloud data, as inputs to the state-of-art R-CNN for object detection on the NYUv2 RGB-D dataset in [5]. Though elementary, this addition of differential geometric features allows us to fine tune CNNs pretrained on appearance (color) images in a straightforward manner. Our experiments show that this improves upon the current best reported object detection results in [5]. We also report on leave-object-out recognition experiments on the easier problem of turntable style object recognition on the UW dataset, where we achieve competi-

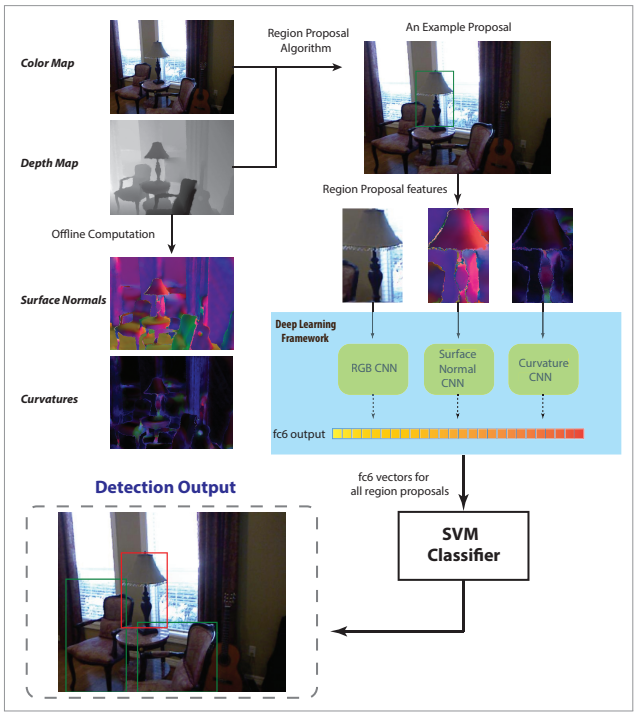


Figure 1: A schematic of the object detection framework. Given an input RGB and depth image pair, we use a pointclouid library (PCL) based implementation to compute the surface normals and curvatures. Then, following the steps used in [5], region proposal algorithms are applied to the color and depth data to generate proposal bounding boxes. Features are then extracted from each bounding box and are fed into CNNs. SVM classification is carried out on the CNNs' fc6 layer vector outputs. The system then generates object detection bounding boxes along with class labels for the RGB and depth image pair.

tive accuracy against the best reported current results in [2]. We describe the framework for detection and recognition, which we adopt from [5], in Section 2. We then describe Gaussian weighted depth attention maps, which we use as zeroth order shape features, in Section 3 and methods for estimating surface normals and principle curvatures from point cloud data in Section 4. Our experimental settings are described in Section 5 and extensive comparative detection and recognition results on the NYUv2 and the UW RGB-D datasets are presented in Section 6. We conclude with a discussion in Section 7.

2. Detection and Recognition Framework

We follow the R-CNN object detection approach of Gupta et al. in [5] since our main goal is to add differential geometric features and to evaluate their efficacy. The numerical implementation of this R-CNN is adopted from the GitHub repository of Gupta et al. [4] and for CNN training we use the Caffe Deep Learning Toolbox [7]. Figure 1 presents a schematic of the approach.

2.1. Region Proposals

To generate region proposals for evaluation we applied the multiscale combinatorial grouping (MCG) method of [1] with color and depth features computed from Gupta et al. [5]. For each 560×425 scene in the NYUv2 database the proposal algorithm typically generates around 2000 region candidates that cover the ground truth objects with high recall.

2.2. Fine tuning Imagenet CNNs for Multiple Feature Spaces

The region proposals for each scene were then fed into multiple CNNs, with different feature spaces as inputs. Pretrained ImageNet CNNs from the ILSVRC 2012 dataset [8] are used as the baseline CNNs, and these are fine tuned using different feature spaces, including color, depth, and shape. Details of the network structure and training parameters are in [8].

2.3. Feature Fusion and SVM Classification

The CNN outputs from the fully connected layers are then used as inputs for the final classifier, based on SVMs. The fully connected layer outputs from different CNNs can be concatenated in a manner of feature fusion. Binary SVMs for each object class are then trained on the fused feature space, and these generate final labels for each region proposal. For each scene non-maximum suppression is then applied on the region proposal boxes to generate sparse detections.

2.4. Recognition Framework

In the context of the simpler turntable style recognition experiments that we carry out on the UW database, we use essentially the same pipeline as that described above from [5] but without using region proposals. After SVM based classification, labels are assigned to each image. For this database each scene contains only one object of interest.

3. Depth based Mixture Attention Maps

Depth based features have been successfully used in previous pattern recognition work on point cloud data [2][5][9][13][14], with different encoding strategies for improving detection and recognition accuracy. In this section we introduce a zeroth order depth feature using Gaussian Mixture Attention Maps. The basic idea is to extend a raw depth map so that it has 3-channels: a foreground map, a mid-range map and a background map. The use of 3 channels is a design choice which we find works well in practice.

The assumption we make is that an object of interest is usually compact in space and is thus bounded by a finite volume which lies mostly in one of these channels. By doing this the CNN can learn features from each of the three depth ranges, and the manner in which they interact. For a given region proposal patch, to classify each point into one of the three categories: foreground, mid-range and background, we apply Gaussian Mixture Models (GMM).

The likelihood of the depth data point d in a given point cloud $\mathcal{D}$ can be modelled with a GMM

$$p(d|\Theta) = \alpha_f \mathcal{N}_f(d; \mu_f, \sigma_f) + \alpha_m \mathcal{N}_m(d; \mu_m, \sigma_m) \quad (1)$$
$$+ \alpha_b \mathcal{N}_b(d; \mu_b, \sigma_b) \quad (2)$$

satisfying $\mu_f \leq \mu_m \leq \mu_b$, where $\mathcal{N}_m$ stands for the Gaussian Model for the mid-range depth data associated with mean μ_m and variance σ_m. The corresponding parameters with sub-index f and b represent the foreground and background ranges. Θ stands for the entire parameter set of the mixture model. We use Expectation Maximization to find the parameter set

$$\Theta^* = \arg\max_{\Theta} p(\mathcal{D}|\Theta) \quad (3)$$

that best separates the three depth ranges. Here we have

$$p(\mathcal{D}|\Theta) = \prod_{i=1}^{N} p(d_i|\Theta), \quad (4)$$

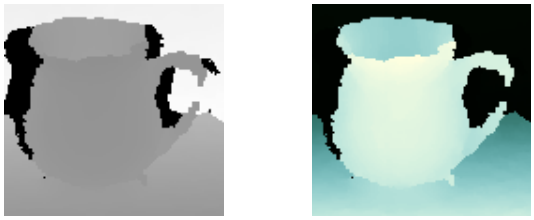
assuming that the observed depth samples are i.i.d. We then process the depth data using the foreground, mid-range, and background Gaussian models to create 3 attention maps. The raw depth $\mathcal{D}_{raw}$ measurements from the point cloud are fed into the trained GMM components $\mathcal{N}_i$ to acquire the Gaussian Mixture Weighted depth $\mathcal{D}_i$ as

$$\mathcal{D}_i = \mathcal{N}_i(\mathcal{D}_{raw}; \mu_i, \sigma_i), \quad (5)$$

where $i = f, m, b$. The final attention map is generated by combing the $\mathcal{D}_i$'s into a 3 channel image and then scaling each channel to the range $[0, 255]$, to make them commensurate with the range of the color channel features. We illustrate the depth based mixture attention map in comparison to the raw depth map for a mug from the UW database in Figure 2. For the NYUv2 dataset, this attention map is computed for each region proposal in a given scene.

4. Surface Normals and Curvatures

We now go beyond zeroth order depth features to include first-order derivatives of depth (surface normals), which are invariant to translations, and also second-order derivatives (surface curvatures), which are invariant to rotation as well. Our numerical estimates are based on use of the Point Cloud Library [12].



(a) Raw Depth Map (b) Mixture Attention Map

Figure 2: Depth based mixture attention map learned by GMM in comparison with a raw depth map for a mug.

4.1. Surface Normals

We acquire estimates of the surface normal at a point of interest by performing principle component analysis over the covariance matrix created from its nearest neighbours. For each 3D data point $p_i = (x_i, y_i, z_i)$ we construct its covariance matrix $\mathcal{C}$ as

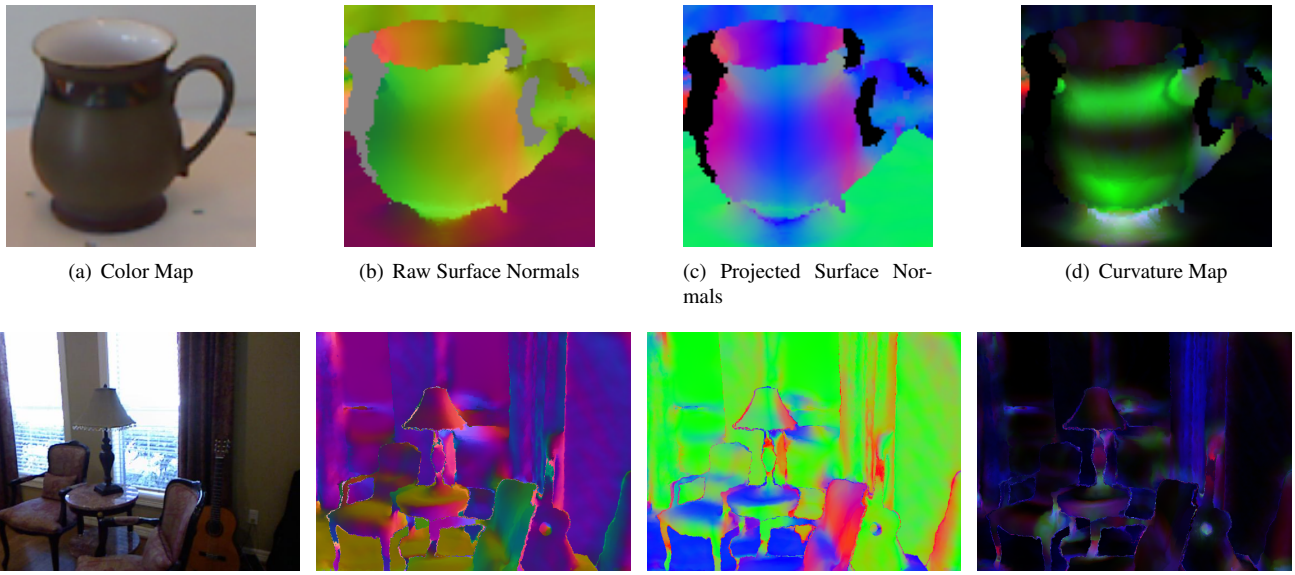
$$\mathcal{C} = \frac{1}{N} \sum_{i=1}^{N} (p_i - \overline{p})(p_i - \overline{p})^T, \quad (6)$$

where N is the number of points that are considered as neighbours of p_i within some radius and $\overline{p}$ is the centroid of the N neighbours. We increase the neighbourhood radius size linearly with increasing depth, in order to acquire a sufficient number of data points for numerical estimation. Eigen system analysis is then performed on the covariance matrix $\mathcal{C}$ and the eigenvector associated with the smallest eigenvalue is treated as the surface normal estimate at this point p_i.

Surface normals consist of unit vectors with each entry having values in the range $[-1, 1]$. To make surface normal features commensurate with the other features used as inputs to the CNNs, we followed two strategies for normalization:

1. We linearly scaled the surface normal map from $[-1, 1]$ to $[0, 255]$. This generates what we refer to as "raw" surface normal maps and we used these as inputs for fine tuning the CNN for detection experiments on the NYUv2 dataset.

2. We took the absolute value of each surface normal entry and multiplied it by 255: $s_{proj} = 255 * abs(s_{raw})$. This has the effect of projecting each surface normal to the positive orthant, which we refer to as "projected" surface normal maps. We used these as inputs for fine tuning the CNNs for recognition experiments on the UW dataset, since they highlight objects with symmetry.



(a) Color Map (b) Raw Surface Normals (c) Projected Surface Normals (d) Curvature Map

(e) Color Map (f) Raw Surface Normals (g) Projected Surface Normals (h) Curvature Map

Figure 3: An illustration of shape features for a coffee mug in the UW dataset and a sample scene from the NYUv2 dataset. In the surface normal maps, the x, y, z components of the vectors are mapped to the colors blue, red and green, respectively.

As illustrated by the example in Figure 3, the NYUv2 dataset is inherently more complex. It contains objects at many different depth ranges of many different sizes in cluttered scenes with complex surfaces. As such, we retain the "raw" surface maps for such scenes. The UW dataset, on the other hand, is quite a bit simpler. Each image in it consists of a single largish object, with well defined boundaries and very little clutter. For such scenes the "projected" surface normal maps have the advantage of emphasizing symmetry.

4.2. Surface Curvatures

Our curvature estimates are also based on principle component analysis over a covariance matrix constructed by point normals in a neighbourhood defined by some radius parameter. For each point p_i, a projection matrix P_i is constructed for the tangent plane at p_i given by the surface normal s_i. Then, for every point p_j in the neighbourhood, each surface normal s_j is projected onto the tangent plane by multiplying it with the projection matrix P_i:

$$s'_j = P_i s_j. \tag{7}$$

Then a covariance matrix for surface normals $\mathcal{C}_s$ is built on top of the projected surface normals, which is given by

$$\mathcal{C}_s = \frac{1}{N}\sum_{j=1}^{N}(s'_j - \overline{s'})(s'_j - \overline{s'})^T. \tag{8}$$

Finally eigen system analysis is performed over the covariance matrix $\mathcal{C}_s$. The maximum eigenvalue is proportional to the first principle curvature κ_1 while the minimum eigenvalue is proportional to the magnitude of the second principle curvature κ_2. We take the average of the two to represent a type of mean curvature:

$$\kappa_m = \frac{\kappa_1 + \kappa_2}{2}. \tag{9}$$

Strictly speaking, of course, covariance based estimation from a point cloud does not capture the sign of the second principal curvature estimate. One would need a more refined procedure for doing this, e.g., based on fitting surfaces.

We construct a curvature map κ with following the 3 channels $\kappa = (\kappa_1, \kappa_2, \kappa_m)$ and scale each channel in it linearly to the range $[0, 255]$, again, to make its values commensurate with those of the other features (color, depth and surface normals). Examples of shape feature maps (surface normals and curvatures) used for finetuning the CNNs are presented in Figure 3.

4.2.1 PCL based Implementation

Our numerical implementations are based on the convenient Point Cloud Library (PCL) [12]. Our code takes a *pcd* file containing the 3D coordinates of each scene point as input and outputs a *pcd* file with each point's surface normal and curvature information. The use of PCL does add computational overhead since PCL is not optimized, e.g., on an Intel i7 Quad core CPU a 560×425 *pcd* image takes around 30

seconds to process. However, since this is done offline in our pipeline this step does not adversely affect recognition or detection time.

5. Experiments

We now present details of the NYUv2 dataset [11], which we use for object detection, and the UW RGB-D dataset [9], which we use for turntable style object recognition. We also present details of the training and testing phases.

5.1. Datasets

For object detection in the NYUv2 RGB-D dataset [11], there are 1449 scenes including everyday indoor environments such as bedrooms, offices, living rooms and kitchens. Within the densely labelled object categories, both Gupta et al. [5] and the sliding shape approach [14] focus on the 19 most common represented furniture object categories including chairs, beds, sofas, tables, and toilets. To provide a fair comparison with the present state-of-the-art work on this dataset [5], we use the same *train*, *val*, and *test* splits they used. In our experiments, all training processes are carried out on the *train* and *val* sets, and the final detection accuracy is then reported on the *test* set, using the trained R-CNN model. The *train* + *val* set contains 795 scenes for development and training and the *test* set contains 654 scenes for testing.

For turntable style object recognition on the UW RGB-D dataset [9] we work on a total of 300 instances spanning 51 different categories of everyday objects, with several images of each instance. The RGB-D dataset provides 10 leave-object-out training/testing splits, which have been used for the baseline recognition experiments in [9]. For each split, one random object within each category is selected as the test object and images of it are not used in the training process. The use of the provided splits, as in [2, 13], is standard when reporting performance on the UW RGB-D dataset for object recognition. The data set contains roughly 42,000 RGB-D images, as well as the point cloud data. The splits give about 35,000 images for training and 7,000 images for testing.

5.2. Experimental Settings

Since the recognition framework is directly adopted from the detection framework, we focus on the experimental settings for the detection task on the NYUv2 dataset.

5.2.1 Training Process

During the training phase, we have to

- Generate region proposals for each scene using the MCG algorithm [1].
- Fine tune the pretrained ImageNet CNNs from the ILSVRC 2012 dataset [8], using the region proposals from the *train* + *val* set. For the CNN training, we used the Caffe CNN library [7] and applied the same training parameters as in [5]. During the training, a proposal patch is labelled as the class corresponding to the maximum overlapping ground truth instance, if this overlap is larger than 50%, and is labelled background otherwise. Multiple CNNs have to be fine tuned for color images, HHA feature maps (used in [5]), surface normal maps and curvature maps. The feature maps are computed before the training process. With an NVIDIA Tesla K40 card, four CNNs can be trained in parallel and it takes around 6 hours for 30k iterations with a batch size of 64, which is a half of the applied batch size in [5].
- Train binary SVMs for each class on top of the CNNs' fully connected layer 6 features. The process and parameters in this phase are the same as in [5].

For object recognition, region proposal algorithms are not applied, since the images contain a single object.

5.2.2 Testing Process

During the testing phase, region proposals for each test scene are first generated using the MCG algorithm [1]. For each region proposal, forward propagation through the CNNs is applied for different feature maps and the fully-connected layer 6 (fc6) features are extracted. These are then fed into binary SVMs followed by per-scene non-maximum suppression in each region proposal box. The above process generates sparse detections for a test scene. For object recognition on a test image, the label associated to the maximum response over the binary SVMs is selected as the predicted label.

6. Results

We present comparative results against Gupta et al. [5] for object detection on the NYUv2 dataset and against [2] [13] for recognition on the UW dataset. For the tables in both Section 6.1 and 6.2, we apply the following abbreviations. RGB stands for the color image. HHA is the feature designed by Gupta et al. [5] which contains for each channel: horizontal disparity, height above the ground and the angle between the surface normal at a scene point and the direction of gravity. GaussD stands for the depth based mixture attention map introduced in Section 3. Shape stands for a feature space containing both the surface normal maps and the curvature maps, as introduced in Section 4.

Table 1: NYUv2 Results

	A	B	C	D	E	F	G	H
input feature	RGB	HHA	GaussD	Shape	RGB + HHA [5]	RGB + Shape	RGB + GaussD + Shape	RGB + HHA + Shape
bathtub	16.22	30.25	9.82	21.34	37.80	29.51	37.86	41.77
bed	41.0	66.34	62.62	68.44	69.90	70.64	73.30	75.03
bookshelf	27.96	19.84	21.81	21.19	33.89	34.37	38.19	36.42
box	0.69	0.90	0.21	0.74	1.48	1.66	1.78	2.17
chair	27.44	35.69	27.33	35.21	43.18	41.63	43.45	46.87
counter	34.56	35.29	25.54	38.13	44.97	43.86	45.26	46.39
desk	8.44	6.53	2.41	5.03	15.73	13.66	14.01	15.75
door	15.19	6.78	3.46	7.36	20.54	22.45	22.65	23.88
dresser	16.92	25.24	11.18	18.41	32.93	31.57	36.65	37.92
garbage-bin	16.51	18.39	5.89	22.17	32.88	34.48	33.64	39.88
lamp	25.88	22.03	23.83	29.93	33.65	35.58	37.16	37.46
monitor	38.42	32.87	4.93	16.74	50.86	44.44	45.35	52.95
night-stand	12.09	23.25	18.05	25.84	31.56	32.63	39.58	41.67
pillow	14.97	31.52	7.56	33.84	37.33	39.06	40.50	43.98
sink	27.5	31.67	14.55	38.53	38.97	46.83	43.81	44.43
sofa	28.16	46.48	27.65	48.36	49.00	45.80	48.44	51.79
table	10.55	19.18	21.77	22.16	22.94	24.28	26.09	26.85
television	24.94	13.39	2.99	9.04	32.23	31.60	30.50	34.51
toilet	44.83	40.00	30.83	38.26	44.91	45.39	45.10	46.96
mean	22.75	26.61	16.97	26.35	35.51	35.23	37.02	39.30

6.1. Detection accuracies on the NYUv2 dataset

We provide detection accuracies for the same 19 classes using the same test set from the NYUv2 dataset as in [5]. Following the standard metric they apply, which is PASCAL VOC box detection average precision (AP), we present APs when using different feature spaces in the R-CNN framework, for individual object classes as well as for overall performance. These results are in Table 1, with the best result for each category, as well as for overall accuracy (bottom row), highlighted with a bounding box. This table's style is similar to that in [5], allowing us to carry out a direct comparison. In these experiments we repeat the experiments of [5] using code from their GitHub repository [4], but without data augmentation using synthetic examples. As reported in [4] [5] such augmentation can improve overall performance by around 1%, but this implementation has not yet been released.

6.2. Recognition accuracies on the UW dataset

We present recognition results for the same system, but applied without region proposals, on the UW RGB-D dataset using different input features, in Table 2, and a comparison against previous work in Table 3. We provide results averaged over the first 4 of their 10 leave-object-out splits, due to the fact that we need to train multiple CNNs for each split which is time consuming. The numbers in both tables are recognition accuracies as percentages.

7. Discussion

We discuss our results and the impact of using differential geometric features in comparison with the present state-of-the-art detection and recognition results on these databases.

Table 2: UW Benchmark

	Mean Accuracy
RGB	79.46 ± 4.20
RGB + HHA	82.98 ± 1.46
RGB + GaussD	85.37 ± 3.54
RGB + Shape	88.40 ± 1.99
RGB + HHA + Shape	88.37 ± 2.17
RGB + GaussD + Shape	$\boxed{88.67 \pm 2.11}$

Table 3: UW results

	Mean Accuracy
Bo et al. [2]	87.5 ± 2.9
Socher et al. [13]	86.8 ± 3.3
Ours	$\boxed{88.7 \pm 2.1}$

7.1. Contribution of Surface Normals and Curvatures

From columns E and H in Table 1, we observe that shape features can boost the performance of RGB + HHA [5] by a non-trivial 10.4% to a new result of 39.3%. In addition, comparing columns E and F, we see that using RGB + Shape, but without zeroth order depth features, is still competitive against the state-of-the-art RGB + HHA. To further demonstrate the contribution of shape features, we provide qualitative examples in Table 4, where we compare the detection outputs between the following two scenarios: R-CNN with RGB + HHA, as used in [5], and the same setting but with the addition of shape features. The detection accuracy results for these two scenarios corresponds to columns E and H in Table 1. It is clear that when combined with shape features, the system from [5] works consistently better, across each object category.

7.2. HHA versus Mixture Attention Maps

The HHA features are comprised of horizontal disparity, height above ground, and the angle between the surface normal and gravity. Whereas this choice is ad-hoc, these features are engineered to work well with the NYUv2 dataset. An implicit assumption behind their use is that the camera viewing direction is parallel to the ground plane and thus the angle component builds a certain invariance to view direction. This assumptions holds true for the NYUv2 dataset and hence it is not surprising that the best results are achieved when combining color, HHA and surface normals and curvature.

HHA does not perform nearly as well in databases where the viewing direction varies relative to the ground plane, as is the case for the UW database. From the experiments in Table 2, it is evident that RGB + Shape outperforms RGB + HHA, while the opposite is true for the NYUv2 detection experiments. In addition, HHA brings down the overall performance when combined with color and shape features in the UW dataset. Another drawback of HHA is that whereas it builds invariance to the viewing angle, details in surface normal variation are lost. This is likely why shape features boost the use of RGB + HHA.

The mixture attention maps, which do not implicitly assume certain camera viewing directions, are successful in both the NYUv2 experiments and the UW RGB-D experiments. Columns E, F and G in Table 1 show that the use of attention maps boosts the performance of RGB + Shape and outperforms Gupta et al. [5], without use of the engineered HHA feature. Moreover, the results with mixture attention maps consistently yield higher recognition accuracy for the UW dataset, as shown in Table 2.

7.3. Conclusion

The use of elementary differential geometric features as inputs to a CNN based detection or recognition pipeline, has the potential to boost recognition performance. We have shown this experimentally by extensive benchmarking on two large databases - the NYUv2 dataset and the UW RGB-D dataset. For both the harder problem of object detection and the easier one of object recognition, we demonstrate a consistent improvement over the current state-of-the-art in terms of accuracy when such features are used. Our results suggest that higher order shape features including surface normals and curvature as independent input feature maps are critical for object recognition and detection. They do not appear to be easily abstracted or learned by CNNs from depth information alone.

Acknowledgments We are grateful to the Natural Sciences and Engineering Research Council of Canada for funding and to NVIDIA for donating a Tesla K40 graphics card to facilitate numerical computation.

References

[1] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 328–335, 2014.

[2] L. Bo, X. Ren, and D. Fox. Unsupervised feature learning for RGB-D based object recognition. In *Experimental Robotics*, pages 387–402. Springer, 2013.

[3] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on*

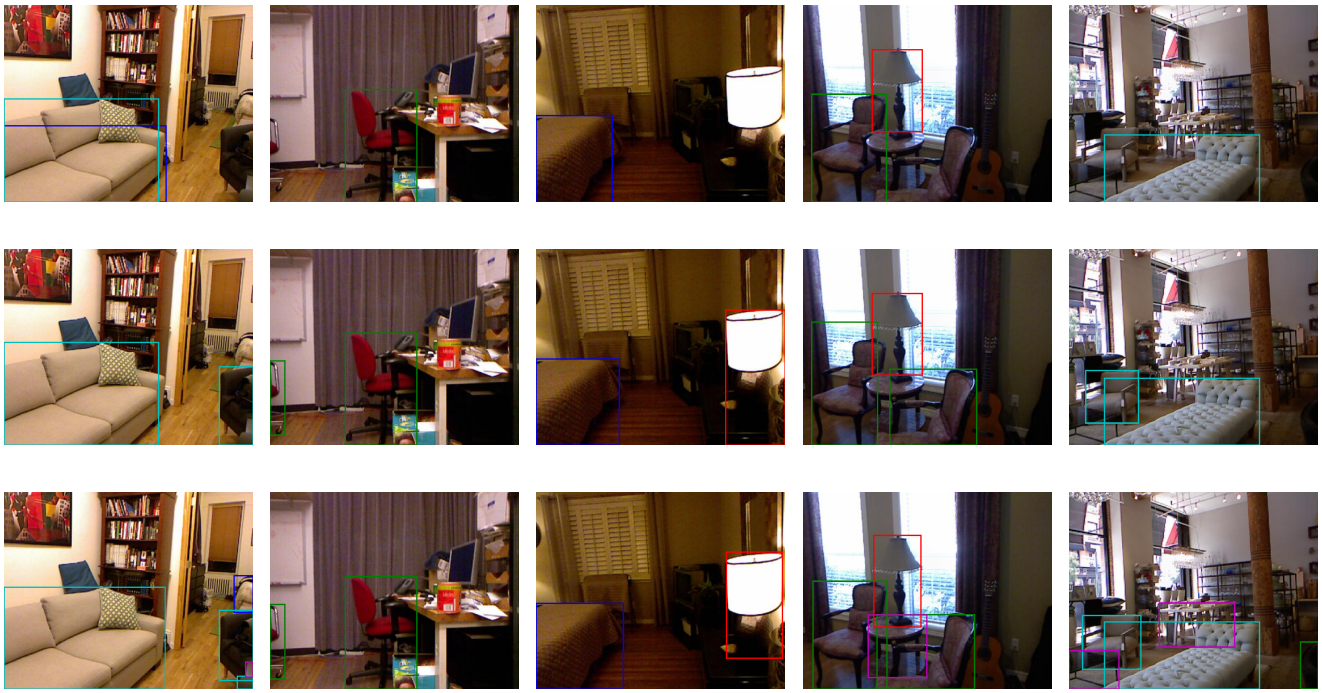


Table 4: Object detection examples focusing on the following 5 classes : *bed, chair, lamp, sofa, table*. These are bounded by boxes with color *blue, green, red, cyan* and *purple*, respectively. The first row contains detection results using [5], while the second row contains detection results when shape features are used in addition. The last row contains the labelled ground truth results. In the first 3 example scenes (columns 1 to 3), our system achieves almost 100% detection accuracy except for scene 1 where some ground truth boxes are too small even for a human to detect. In the last 2 examples (columns 4 and 5), we missed several ground truth objects while still outperforming [5].

computer vision and pattern recognition, pages 580–587, 2014.

[4] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Implementation for learning rich features from RGB-D images for object detection and segmentation. `https://github.com/s-gupta/rcnn-depth`, 2014. [Online; accessed 24-Mar-2016].

[5] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *Computer Vision–ECCV 2014*, pages 345–360. Springer, 2014.

[6] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[7] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011.

[10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[11] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[12] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[13] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Y. Ng. Convolutional-recursive deep learning for 3D object classification. In *Advances in Neural Information Processing Systems*, pages 665–673, 2012.

[14] S. Song and J. Xiao. Sliding shapes for 3D object detection in depth images. In *Computer Vision–ECCV 2014*, pages 634–651. Springer, 2014.