Classification of Objects in 3D Indoor Scenes Using Unsupervised Feature Learning

James A. Hill
augie@umich.edu

Arun Sundar Govindarajan
arunsg@umich.edu

Sharath Rathinakumar
sharath@umich.edu

Abstract

We present an exploration of methods for applying unsupervised feature learning to the problem of classifying objects in a three-dimensional scene, a problem at the core of autonomous robotics research. We utilize color point cloud data collected with a Microsoft Kinect sensor in two different settings: a work setting and a home setting. This work builds upon previous research which utilized a contextual graph of each scene and hand-crafted features, with good results. While we utilize a few of the features of the previous work, our work differs significantly in our use of color data and our use of appearance features, each learned in an unsupervised manner. In particular, we explore unsupervised appearance feature learning methods which rely upon standard two-dimensional scale-invariant feature transform (SIFT) descriptors, which are better studied, and recently-proposed three-dimensional SIFT descriptors, which are more appropriate for our setting. Our results show that the utilization of appearance features learned by unsupervised methods generally improve classification performance.

1 Introduction

The general problem we aim to solve is that of automatically classifying objects in a three-dimensional setting. This problem is at the core of autonomous robotics research, as being able to function in an environment, including manipulation of objects, requires the ability to understand what objects exist within the environment. Successfully solving this problem leads directly to autonomous robots that can be visually taught labels for objects, and when coupled with a semantic understanding of those labels, results in a robot that understands and productively interacts with its environment.

Traditionally, researchers in the field of computer vision have focused on using two-dimensional images in object classification tasks. With the advent of new depth-enabled devices like Microsoft's Kinect, both color and depth information (RGB+D) about scenes can be utilized to increase the fidelity of the classification input data. Scene point clouds, as used in simultaneous localization and mapping (SLAM), can be constructed from the data collected with the Kinect. Using three-dimensional scene images has various advantages and can help improve the result of object classification [1]. In this project, we aim to use the data collected from a Kinect sensor, converted into a point cloud representation, as input to an object classification algorithm which utilizes unsupervised feature learning methods.

This work is an extension of Koppula et al. [6, 7]. The authors of the previous study have kindly made available all of their source code and data, so the tedious data gathering step was not part of this study. In general, we aim to automatically label objects in color point cloud data collected from indoor scenes utilizing in part features learned in an unsupervised manner, then evaluate our learning method's performance and compare that performance with [6, 7].

Our primary hypothesis is that the utilization of appearance features learned in an unsupervised manner will perform better than the manually-crafted feature method utilized by Koppula et al. [6, 7].

We are able to directly compare the results of the two methods because we have obtained the same data that was collected and utilized by the previous research.

2 Related Work

2.1 Object Recognition and Scene Understanding

Object recognition and scene understanding have been an active research area in computer vision. Given a single two-dimensional image, object recognition is formulated as a classification task, which determines whether certain objects exist in a window of the image. The goal is thus to train a discriminative classifier from visual inputs. However, objects from the same class might have very different appearances. Felzenszwalb et al. [4] have adopted a deformable part-based model to capture the intra-class variability in the appearance of objects. In addition to the visual appearance, Dasai et al. [3] further incorporate 2D object spatial relationships and object co-occurrence statistics to refine the recognition result of a scene. Instead of characterizing objects and their relationship in 2D images, Bao and Savarese [1] claim that by jointly exploring 3D geometry of the scene, one could obtain more robust results for object recognition. Given multiple images, their method first establishes the correspondence between feature points and between object detection responses across images. They then jointly recognize objects and their spatial organizations in three dimensions.

The availability of inexpensive RGB+D sensors in recent years, particularly the Microsoft Kinect[1], has resulted in the proliferation of research towards scene understanding that utilizes RGB+D data. The use of RGB+D cameras have reduced the computational burden and increased the accuracy of 3D scene reconstruction from 2D images. One method by which RGB+D images are merged to obtain a 3D point cloud is by simultaneous localization and mapping (SLAM)[2]. Some recent works exploit the acquisition of such 3D information to recognize objects in a scene.

Koppula et al. [6, 7] proposed a method which poses the object recognition problem as a strongly contextual problem. Their method starts by segmenting the three-dimensional point cloud into clusters of points. They model the scene as a graphical model using Markov Random Fields (MRF), where a node represents a segment, and an edge represents a proximity between two segments. By combining a wide range of manually created features, the graphical model captures visual appearance, shape cues, geometric relationships, and object co-occurrence statistics. Finally, they train the model using a variation of a maximum-margin classifier (SVM). Their ultimate reported precision and recall are 56.81% and 54.80% for the home setting and 80.52% and 72.64% for the office setting. Designing the features used in this method was a complex and laborious task, no doubt, and may be prone to error due to its reliance on human design inputs. By utilizing automatically learned features, we hope to improve both the accuracy and generalization of object classification in color point cloud inputs.

2.2 Unsupervised Feature Learning

Methods for unsupervised feature learning were developed to solve the problem of the hand-crafting of features, which is typically a laborious task for even the most skilled designer. These methods generally involve the training of a model with copious amounts of example data, then the generation of a feature vector for each example from that model. We utilize this basic concept for unsupervised feature learning with the simplest of methods, K-means clustering.

Recent work has shown that the use of hierarchical models of the examples improves the abstraction and applicability of the learned features (e.g., convolutional deep belief networks [8]). We leave for future work the use of hierarchical unsupervised feature learning methods in our setting.

2.3 Scale-Invariant Feature Transform (SIFT)

The scale-invariant feature transform (SIFT) algorithm was introduced by David Lowe [9] in 1999 in part to automatically recognize the same or similar objects in different images. It does so by identifying the most important points of the image, the keypoints, and calculating 128-valued *descriptors*

[1]`http://www.xbox.com/en-US/kinect`
[2]`http://openslam.org/rgbdslam.html`

for each of those points. This algorithm has been widely applied to classification of objects in 2D images, but has been used little in three-dimensional space. Sehgal et al. [11] succesfully utilize SIFT descriptors in a 3D scene by using grey-scale distance measurements in place of the available RGB colors. We utilize a similar method.

An extension of the SIFT descriptor to three dimensions was presented by Scovanner et al. [10] in 2007, which have been utilized in preivous work on three-dimensional to recognize similar locations and join them into a single representation of the scene. We present in this paper a novel application of 640-valued 3D SIFT descriptors to the task of recognizing objects in a color point cloud scene.

3 Methods

Our work is based on the premise that there could be some features, picked by some unsupervised learning algorithm, which could yield better classification results. Thus, we explore this hypothesis by using simple unsupervised feature learning techniques to automatically learn features which are representative of the overall appearance of segments of the scene point cloud.

The input to our learning method is one of two collections of color point clouds made available by Koppula et al. [6, 7]. We utilize four categories of features: color, shape, context, and appearance. The 12 context and shape features were designed manually, and are the same or similar to some of the features used in the other study. The color and appearance features are learned by the following simple unsupervised feature learning method.

- Pool the categorical descriptions (color or SIFT descriptor) for all of the segments.
- Perform K-means clustering on the pool of descriptions.
- Use the K mean descriptions from the previous step as features. The feature vector for each of the segments is a histogram of descriptions which are nearest to the K mean descriptions.

3.1 RGB Features

Histogram-based color aggregation methods have been shown to work well in practice for image classification tasks, particularly with the support vector machine classifier, as shown in the study on classification of images by Chapelle et al. [2]. While Koppula et al. used HSV to represent their colors, we instead used RGB, as the results of Chapelle et al. showed that when using a Guassian kernel for SVM, which we use, representation by RGB tends to perform slightly better than HSV.

Each of the 3 values in an RGB descriptor may have a value in the range $[0, 255]$, a total of 256 possible values. Histogram features are calculated by binning along the three dimensions at some level of coarseness, determined by the *bins* hyperparameter.

As in [2], after binning the points of each of the segments by RGB color, we normalized the values by dividing by the sum of all histogram bin values for each example. We initialized each of the bins with a value of 1 to avoid any possible future issues with division by zero, as is possible when using the χ^2 distance measure. Thus, the values for each bin are in the range $(0, 1)$, which makes the segments more comparable by abstracting away the number of points in each segment.

3.2 Shape Features

We utilize 6 simple manually-designed shape features. The first 3 are derived from the scatter matrix of the segment point cloud, and the second 3 describe the bounding box of the segment.

The scatter matrix is 3×3, from which the 3 eigenvalues are calculated. The eigenvalues are then sorted from lowest to highest, resulting in the array $[\lambda_0, \lambda_1, \lambda_2]$. Using these eigenvalues, we then calculate the segment's *linearness* by $\lambda_1 - \lambda_0$, its *planarness* by $\lambda_1 - \lambda_2$, and utilize its *scatter* measure λ_0. The features described in this paragraph are exactly as utilized by Koppula et al. [6, 7]. The values of each of the measures is then individually scaled to be in the range $[0, 1]$.

The second set of 3 features in this category describe the *minimum bounding box proportions* of the segment. The segment may initially be in any orientation, so there may be much empty space within the initial bounding box. In order to best assess the true dimensions of the segment itself, the segment is rotated about each of the 3 axes so that the length of each of the 3 dimensions is minimized. This process is guaranteed result in a bounding box which measures the canonical dimensions of the segment itself. The segment is now in one of 6 orientations, assuming it is generally a box shape. For comparison between segments, the dimensions are sorted by length and the measure used is the proportion of the length of each bounding box dimension divided by the sum of the length of all bounding box dimensions.

3.3 Context Features

6 features were utilized which describe the size and position of the segment to be labeled relative to the scene in which it exists. The first set of 3 features relate the dimensions of the segment to the corresponding dimensions of the scene by simple division. The second set of 3 features relate the center of the segment to its relative position within the scene, also by simple division.

3.4 Appearance Features

3.4.1 2D SIFT Descriptors

We first explored the use of standard 2D SIFT descriptors. The first portion of this task was to extract 2D SIFT descriptors from each of the segments. In order to utilize 2D SIFT-based algorithms, the 3D segments must be converted into a series of 2D images. The following method describes the steps for converting each segment into a series of 180×180 images.

- Center and rotate the segment so that the size of the segment's bounding box is minimized.
- Move the segment into a 90 degree high and wide field of vision.
- Assign each segment point to a pixel of the resulting image, then average the value of the image pixel over all pixels assigned to it. This is the resulting image.
- Perform the above steps for each of the 6 faces of the segment (we are assuming that the segment is generally a box shape), resulting in 6 images.

SIFT descriptors are extracted and pooled for keypoints in each of those 6 images. The keypoints were restricted to those entirely surrounded by non-blank pixels, that is, interior points. With descriptors for the entire data set, K-means clustering is then utilized to search for a set of K descriptors which best represent the set of all descriptors. We used $K = 50$ and allowed the clustering algorithm to run until convergence.

The final step is to assign the descriptors of each segment to one of the 50 automatically-learned descriptors. The end result of the method is a histogram. Thus, we have a total of 50 features for each segment, each corresponding to the number of SIFT descriptors closest to each of the 50 cluster centroids.

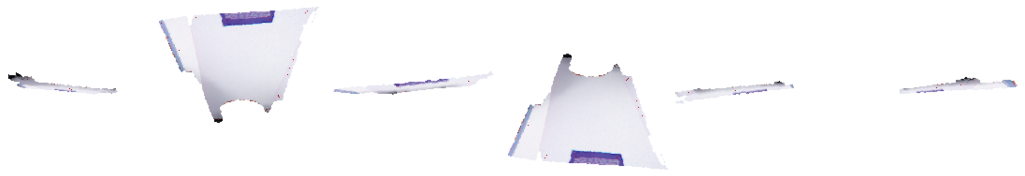

Figure 1: A series of 6 images of a wall segment from which 2D SIFT descriptors (indicated by the red dots) are extracted.

This method is applied in two different manners. The first method is the standard color-based approach. The second method, based on the work of Sehgal et al. [11], represents the three-dimensional nature of the segment in a two-dimensional image by assigning to each image pixel the distance from the origin to the point.

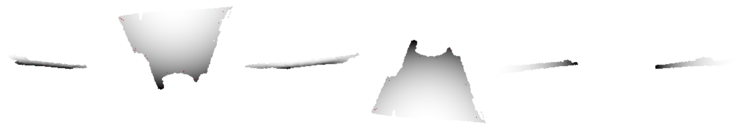

Figure 2: Images of the same wall segment with color based on distance from the origin.

3.4.2 3D SIFT Descriptors

We next devised a method to use 3D SIFT descriptors. In this method, segments must first be reduced to a voxel representation, as opposed to the more free-form coordinate system of the point cloud. Each segment is converted to a $75 \times 75 \times 75$ 3D image, where the value of each pixel is the average RGB values of the pixels assigned to that pixel.

The approach to SIFT descriptor extraction is different in this section than in the previous section. In the previous section, descriptors for only the most distinctive keypoints were used. We found through the experimentation process that this produced relatively few descriptors, which likely does not generalize well to categorical classification because it produces sparse histograms. Thus, in this section we attempt a different method. Here, we subsample 40 non-blank pixels uniformly at random from the 3D image, and calculate the 3D descriptors for each of those pixels. With this method, we hope to produce more distinguishable appearance feature histograms. We recognize that 40 may have been a low number given the size of the segments, so future work should consider increasing the proportion of the image sampled.

Besides the method for choosing the pixels for which to obtain descriptors, the remainder of the method is the same as the 2D SIFT descriptor method. We obtain 50 representative descriptors by K-means clustering using a uniformly randomly selected 10% of the total set of descriptors (since there are very many descriptors), then obtain feature values for each of the segments by generating a histogram of descriptors, assigning each descriptor to the nearest basis descriptor.

4 Data

For our experiments, we utilized the labeled 3D point cloud scenes[3] from the study by Koppula et al. [6, 7]. The collection used from that set of data was the *Stitched Pointclouds*, which are divided into two separate data sets–home and office scenes. As did Koppula et al., we treated the home and office collections as independent data sets. Figure 3 shows an example scene from each of the home and office collections.

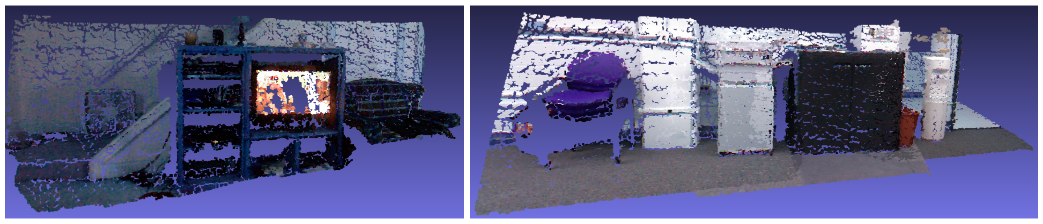

Figure 3: Two example 3D scenes made available by Koppula et al., one from the collection of home scenes (left) and one from the collection of office scenes (right).

Koppula et al. collected the 3D scenes with the use of a Microsoft Kinect sensor, which captures color as well as depth information for each of the points it observes. They then used simultaneous localization and mapping (SLAM) to combine many independent Kinect views into a single point cloud which is the final representation of the scene (see Figure 3). Each scene was then over-segmented based on the smoothness and continuity of surfaces [7], and segments were hand-labeled as one of 129 different labels. The additional default ”no-label” label is 0. The segments and their

[3]`http://pr.cs.cornell.edu/sceneunderstanding/data/data.php`

corresponding labels are the ground truth for the Koppula et al. study, and are so for our study, as well.

The point clouds were stored in the binary PCD file format, but did not strictly conform to the official specification. The extraction of the point cloud data from the files was thus quite difficult, but we were able to do so by guessing and checking their abnormal file format methods, and eventually were able to use all of the supplied data. Each scene point cloud contains about one million points., and each point of the point cloud has the following features: (x, y, z) coordinates, (r, g, b) values, distance from the camera to the point, the segment identifier, and the segment classification.

There are 24 office scenes, which are divided into $1,108$ labeled segments and $51,139$ unlabeled segments. There are 28 home scenes, which are divided into $1,387$ labeled segments and $55,665$ unlabeled segments. Only the labeled segments are used in the classification training and testing, as in the Koppula et al. study. The unlabeled segments tend to be very small. Of the unlabeled segments in the office scenes, only 1.5% of the files are greater than 100 kilobytes in size, while 89% of the files are less than or equal to 2 kilobytes in size. The unlabeled segments of the home scenes have a similar file size distribution. Compare this with the office segments labeled *wall*, of which 85% of the files are greater than 100 kilobytes in size. This indicates that unlabeled segments are mostly segmentation noise, and are thus safely ignored during the classification task.

Our study differs significantly from that of Koppula et al. in that we are not utilizing features which capture the nearby segments within a scene, abstracting away the objects in the scene in which the segment was observed. Thus, whereas they necessarily split their training and testing data by scene, we split ours by segment, perhaps with beneficial effects.

They limited their test segments to a set of 17 for each of the home and office data sets, as do we. We also changed to 0, as we believe they did as well, the label of everything other than those 17 labels, treating it as the ”everything else” label. Figure 4 shows the histogram of labeled segments for each of the 17 labels for each of the reduced sets. It is clear from these histograms that the first

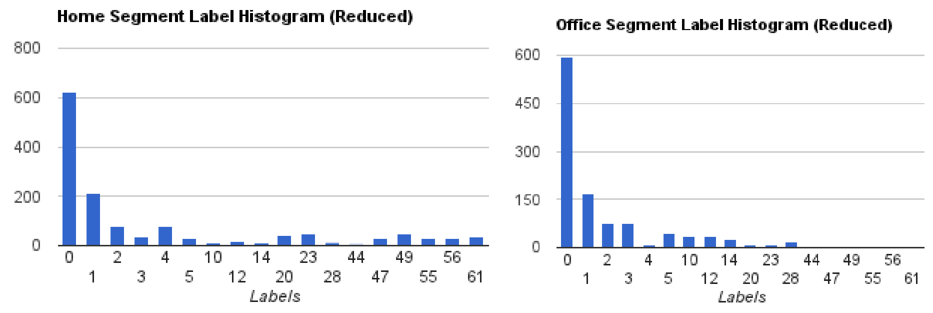

Figure 4: Histograms of labeled segments for each of the labels in the reduced label sets.

few labels will dominate the results, and that the difficulty of correctly labeling many of the objects will be very high since there are so few examples.

5 Experimental Results

5.1 Experimental Methods

We utilize the LibSVM library for our primary classification experiments. The C-SVC type of SVM of used with an Gaussian (RBF) kernel, with hyperparameters C and γ determined for each feature set by the grid search method, as specified by Hsu et al. [5], using 4-fold cross validation. For grid search, the stopping criteria (ϵ) was set to 0.1, and was decreased to 0.0001 for testing. All labels were assgined equal weight. In order to increase the speed of the classifier, the shrinking heuristic was enabled.

The Koppula et al. study uses micro and macro precision and recall to aggregate the results of their multi-class classification algorithm. So that we may be able to compare our results to those of the previous study, our results are shown in terms of macro precision and recall, and we additionally include an F-score in order to more clearly decide the relative scores between classification methods. Though not specified in their papers, we have assumed that the macro precision and recall are in fact aggregated by the *weighted* average of the precision and recall of the classes, so that the classes with very few examples, of which there are several, cannot significantly impact the overall results.

5.2 Control

We first take two control measurements. The first is a simple random guess of the label. After seeing the distribution of labels, it is clear that the best control classification measure would in fact be to always choose the ”everything else” label (0), which is denoted ”All 0” in Table 1. The F-scores for

	Office			Home		
	P	R	F	P	R	F
Random	15.64	5.23	7.84	17.24	4.87	7.59
All 0	15.20	38.99	21.87	20.18	44.92	27.84

Table 1: Results of classification by guess and by always choosing 0.

the *All* 0 classifier are the control measures for our experiments.

5.3 RGB, Shape, and Context Features

We considered three different values for the number of bins along each dimension: 4, 8, and 16, taking into account the precision, recall, and required training time when deciding which to use. Each value of *bins* results in 64, 512, and $4,096$ features, respectively. We ran n-fold k-nearest neighbors on both the office and home data sets, for $k \in \{1, 3\}$ and *bins* $\in \{4, 8, 16\}$. Results of these classifications are in Table 2, for which the distance function used was χ^2, which was found by Chapelle et al. to work particularly well for color histogram-based classification tasks. It appears

	$k =$		1			3			5	
	bins $=$	4	8	16	4	8	16	4	8	16
Office	P	36	39	41	35	36	40	33	33	41
	R	36	39	41	38	41	43	39	40	43
Home	P	36	42	43	34	41	42	35	42	42
	R	35	41	42	43	47	45	44	46	44

Table 2: Results for n-fold K-nearest neighbors, varying K and the number of RGB histogram bins.

that as k increases, results are generally the same or a little worse. This may indicate the classes are generally well-mixed. As expected, when the resolution of the histogram is higher, the results are generally better. We next evaluated the RGB, shape, and context feature sets with an SVM classifier.

	Office			Home		
	P	R	F	P	R	F
RGB (8 *bins*)	42.94	46.03	44.43	42.08	48.81	45.20
RGB (16 *bins*)	48.92	50.36	49.63	40.73	49.24	44.59
Shape	36.02	40.97	38.34	33.92	50.07	40.43
Context	36.17	46.48	40.68	40.29	53.14	45.83
RGB (16 *bins*) + Shape + Context	59.08	57.94	58.50	52.19	57.39	54.67

Table 3: Results of classification by 4-fold cross-validation using various feature sets.

Results are shown in Table 3. Comparing these results with our control, we see that the results are quite good for all features and all data sets.

5.4 Appearance Features

We next perform experiments upon the features learned by unsupervised methods, which in general describe the overall appearance of the segments. Tables 4 and 5 respectively show the results of classification using 2D and 3D SIFT descriptors. These results show that the learned features

	Office			Home		
	P	R	F	P	R	F
SIFT (RGB)	29.62	37.73	33.19	31.47	44.49	36.86
SIFT (Shape)	25.30	37.09	30.08	30.00	44.56	35.86

Table 4: Results of classification by 4-fold cross-validation using RGB and Shape SIFT descriptor histograms.

	Office			Home		
	P	R	F	P	R	F
3D SIFT ($K = 50$)	36.31	41.70	38.82	37.35	47.44	41.79
3D SIFT ($K = 100$)	37.95	42.60	40.14	34.75	46.43	39.75
3D SIFT ($K = 200$)	35.16	42.60	38.52	37.65	46.79	41.73

Table 5: Results of classification by 4-fold cross-validation using the 3D SIFT descriptor histograms.

were useful in distinguishing between the various object classes, performing significantly better than the control. It is also apparent that the method which utilizes 3D SIFT descriptors significantly outperforms the method which utilizes the standard 2D SIFT descriptors.

5.5 RGB, Shape, Context, and Appearance Features

For our ultimate experiments, we chose to combine the 16-bin RGB, shape, context, and $K = 50$ 3D SIFT descriptor features, the results of which are shown in Table 6. These results indicate that the

Office			Home		
P	R	F	P	R	F
59.59	56.41	57.95	56.83	58.47	57.64

Table 6: Results of classification by 4-fold cross-validation using all features.

addition of 3D SIFT descriptor features had a small effect on the ultimate classification performance, improving precision by half a percent for the office data set, although the F-score for the office data set was reduced due to lower recall. We can unambiguously say that the learned appearance features improved the classification for the home data set by 5% precision and 1% recall, for a 3% F-score improvement.

Comparing these results to those of Koppula et al. , we have improved upon their best score for the home data set by 0.02% precision and 3.67% recall. Unfortunately, the same is not true for the office data set, which is 10% to 20% worse than their best score. This perhaps raises the queston of whether their features were inadvertently hand-crafted to favor the office data set to the determinent of the home data set. Such an issue clearly does not arise when utilizing features learned in an unsupervised manner.

6 Conclusions and Discussion

The problem of automatically classifying objects visually is critical to the development of fully autonomous robots. Previous work utilized hand-crafted features as the method for automatic three-dimensional object classfication. Hand-crafting features is a laborious task that does not generalize well for all types of objects and contexts. We have proposed to instead utilize unsupervised feature learning for this task, and have generally shown that features learned in such a manner can be beneficial to the classification task. Because features are learned automatically, our method generalizes well across classes of objects and domains with little additional effort relative to the previous method.

Our results showed that the novel use of both 2D and 3D SIFT descriptors in a classification task were able to modestly improve the overall classifier performance. We presented a novel method for utilizing 2D SIFT descriptors for classification in a three-dimensional environment, and also presented a novel application of 3D SIFT descriptors to a classification task.

Upon reflection, it is clear that the use of a contextual graph of a scene, the method explored by Koppula et al. , is crucial to the performance of a classifier in this setting. The lack of fidelity on each of the segments individually plus the virtual guaranteed occlusions obscuring part of segments combine to make the observations of individual segments very obscure, even to a human. Future work should consider combining appearance features generated by unsupervised methods, as we have done in this paper, and the contextual graph of the scene, as demonstrated by Koppula et al. , to further improve classifification performance.

References

[1] S. Y. Bao and S. Savarese. Semantic structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2025–2032. IEEE, 2011.

[2] O. Chapelle, P. haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5):1055–1063, September 1999.

[3] C. Desai, D. Ramanan, and C. Fowlkes. Discriminative models for multi-class object layout. In *ICCV*, pages 229–236, 2009.

[4] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 1–8. IEEE, 2008.

[5] C. Hsu, C. Chang, and C. Lin. A practical guide to support vector classification. Technical report, National Taiwan University, April 2010.

[6] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. Labeling 3d scenes for personal assistant robots. In *R:SS workshop on RGB-D cameras*, 2011.

[7] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *NIPS*, 2011.

[8] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

[9] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, September 1999.

[10] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *MM*, 2007.

[11] A. Sehgal, D. Cernea, and M. Makaveeva. Real-time scale invariant 3d range point cloud registration. In *ICIAR*, 2010.