Survey of Object Classification in 3D Range Scans

Allan Zelener

Abstract

Modern 3D cameras allow for the acquisition of large 3D scenes containing objects of interest that can be detected and classified for scene understanding. This survey focuses on the task of object classification in urban range scans and indoor RGB-D images. Each approach will be compared based on several factors including classification accuracy, utilization of the 3D data, and performance on key object categories as well as the techniques used for segmentation, feature generation, and classification. Major trends that appear in the literature include the incorporation of contextual cues in classification and the challenge of generating sophisticated 3D feature representations. We conclude that modeling contextual relations for structured prediction provides significant practical benefits and 3D shape-based descriptions can yield an improvement over only using 2D texture based features.

1 Introduction

Our world is composed of three-dimensional objects and in order for automated systems to accurately model and reason about these objects their inherent 3D nature must be incorporated in our representations. Using modern day range sensors it is possible to collect large volumes of 3D data to create these representations and utilize them for applications such as reconstructing architectural models of buildings, navigation of self-driving cars, preservation of at-risk historical sites, and recreating virtual environments for the film and video game industries. Time-of-flight based laser sensors mounted on both terrestrial and aerial vehicles have been used to acquire large-scale point clouds of urban scenes. In recent years the availability of low-cost sensors such as the Microsoft Kinect have enabled the acquisition of short-range indoor 3D data at the consumer level, and soon projects like Google's Tango will bring depth sensors to tablets and other mobile devices.

This 3D data introduces new challenges for developing efficient and accurate algorithms. The raw data itself is large with scenes containing millions of 3D points which should be organized into more efficient data structures. Despite having many points, the density may vary throughout a single scan based on the distance of scanned surfaces from the scanner. It is also possible to have missing

data due to occluded surfaces or materials with reflectance properties that are incompatible with the 3D sensor. Acquiring complete 3D models depends on multiple views of a given object due to self-occlusion, in which case all the views need to be accurately registered. Finally there is much variation between different 3D sensors, for example the effective range of the Kinect, which is designed for indoor user interface in a living room environment, is much shorter than LiDAR scanners designed for outdoor surveying applications. However there are also inherient advantages to 3D sensing as an additional modality for computer vision such as resolving ambiguity in scale, separating foreground from background by depth, and accurately estimating geometric shape from single views.

This survey will closely examine eight papers that describe systems for object classification and semantic segmentation in 3D scenes. First we will define these tasks and briefly summarize each of the eight highlighted approaches to solving these tasks. The selected papers show the trajectory of research for these tasks over the last decade. Sections 2 and 3 will examine the tasks in more detail for urban and indoor scenes respectively. Section 4 will compare the approaches for urban scenes together with those for indoor scenes as well as additional related papers and recent work. Section 5 provides conclusions on the state of research in object classification for 3D scenes and directions for future work.

1.1 Object Recognition and Classification

The first step to building interactive and sophisticated models of acquired 3D scenes is to identify the meaningful objects that reside in those scenes. In the self-driving car application this would include objects like other vehicles, pedestrians, signs, and traffic lights. For each 3D data point lying on the surface of some object of interest we would like to assign a label that is characterized by that object. This label may identify the object as the closest match to a specific object instance that the system contains in a database or it may designate the broader semantic class of that object. The task of identifying a specific object instance is known as object recognition, whereas the task of identifying the class is known as object classification. Additionally it is also possible to jointly label every scene point including foreground objects of interest and other background surfaces; this task is refered to as semantic labeling or semantic segmentation and by definition it simultaneously accomplishes object classification.

Some of the earliest systems in this domain performed object recognition by matching 3D features to reference objects in a database and either minimizing the error between matched features or the error in pose registration. Local features, centered on specific points in the scene, like the classic spin image [7] feature proved to be robust to occlusions and clutter by allowing matches based on small visible patches. Spin images acculumate a histogram of neighboring points using the support point's surface normal as the axis of reference, making the descriptor invariant to rotation. This allowed their system to recognize distinctive objects like a rubber duck among a collection of other toys. An example of a spin image can be seen in Figure 1.

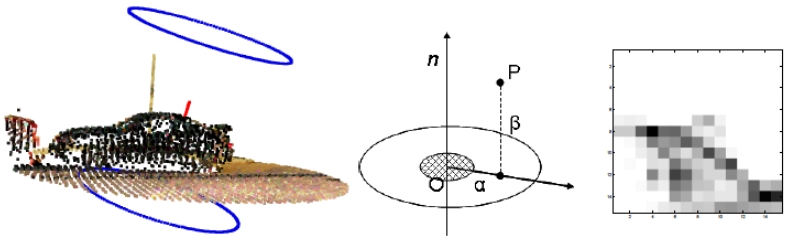

Figure 1: From Patterson [12]. Example spin image computation on real data. Left: Blue circles indicate the cylindrical support region and the red line is the surface normal of the support point. Middle: O is the support and n is its normal. α and β correspond to the projection of point P into the spin image's radial and elevation direction. Right: The complete spin image accumulating all points in the support region.

These types of features are still commonly used for recognition and classification, and the spin image itself is still a popular choice, used by many of the papers in this survey, despite many competing feature descriptors. This survey will not explicitly discuss the object recognition task where research has largely focused on engineering more reliable features for finding point correspondences, but it does include several papers that straddle the line between recognition and classification. Eventually approaches became more classification based because by reformulating object understanding as a classification task it becomes possible to generalize to new variations of the same class. It also allows for a more compressed representation than a database of object instances, such as simply the learned parameters of a machine learning classifier. Classification as a preprocessing step for recognition can also reduce the number of possible database entries that need to be matched.

Patterson et al. [12] classifies objects as vehicle versus background by identifying candidate vehicles using local spin image features and then verifies candidate matches using a global shape descriptor. Once vehicle objects have been isolated, the work of Huber et al. [6] decomposes them into parts and a vehicle class is inferred from the distribution of local features over the parts. Golovinskiy et al. [5] provide a complete pipeline to first localize and segment general object candidates using a graph cut approach and then classify using local 3D features and spatial relationships to other nearby objects.

1.2 Semantic Segmentation

From the earlier systems we see that segmentation of the object from the background is an essential step to achieving good classification accuracy. However errors in segmentation may propagate to the classification. Addressing this issue has lead to new approaches that jointly segment and classify all parts of a

scene.

In semantic segmentation it is possible to exploit additional information about an object from its local context by considering the properties of other nearby objects and background surfaces. Even using an extraordinarily simple hidden Markov model over single scanlines of a 3D scan, Stamos et al. [18] are able to label car objects among the simpler background categories of vertical and horizontal surfaces and vegetation. Xiong et al. [20] propose a system that makes a sequence of predictions over regions during training, allowing each region to build a feature representing the distributions of the possible class labels for neighboring regions determined by temporary classifiers that are discarded after training.

Going further, there seem to be two general approaches that have become popular for semantic segmentation. These methods have been adopted from 2D vision and are applied to RGB-D sensors which have good registration between color and depth channels. One is based on structured prediction and graphical models, which formulates semantic segmentation as assigning class labels to a set of semantic regions in the scan with dependencies described by the graphical model. Silberman and Fergus [14] formulate a conditional random field model using a combination of RGB and depth based features over superpixels for the unary and binary potentials. Anand et al. [1] similarly describe a graphical model with potentials based on color and depth features over regions that is learned using a structural SVM.

The other popular direction involves deep learning and neural networks, which eschew using manually engineered features like the spin image and instead assemble feature representations automatically by processing the low-level sensor readings through networks of simple learning units.

1.3 Survey Outline

Throughout this survey the systems will be compared based on how many different object categories can be distinguished, the nature of those categories, how they handle the 3D data that they use and what assumptions they make about their application domain. Where evaluations are reported using the same datasets then the systems can be compared directly in terms of classification accuracy. A summary of the highlighted approaches can be found in Table 1.

Section 2 focuses on systems for urban environments and includes a comparison based on recognition rates of the car class which is prominent in urban scenes. These systems will also be compared on how they utilize context, features of the environment in which the object resides, in order to improve classification performance. For the indoor approaches found in Section 3 there will be a comparison between systems that more formally incorporate context using probabalistic graphical models and between methods that construct learned feature representations instead of using manually engineered features.

In Section 4 we will compare the systems for urban and indoor environments together. We will look at the differences in the number of object classes, how the classes are defined, the differences in how point cloud and RGB-D data are

System	Section	Classes	Datatype	Domain
Patterson	2.1.1	1 - 3	Point cloud	Urban
Huber	2.1.2	8	Point cloud	Isolated Vehicles
Golovinsky	2.1.3	16	Point cloud	Urban
Stamos	2.1.4	4	Point cloud	Urban
Xiong	2.1.5	5 - 8	Point cloud	Urban
Silberman	3.1.1	12	RGB-D	Indoor
Couprie	3.1.2	14	RGB-D	Indoor
Anand	3.1.3	17	RGB-D	Indoor

Table 1: Summary of Object Classification Systems.

utilized, and how contextual cues are utilized. Section 5 will summarize the conclusions we draw from these comparisions and recommend future directions for research on object classification in 3D scenes.

2 Urban Object Classification

2.1 System Overviews

Earlier efforts in 3D sensing have focused on urban and suburban scenes where the primary objects of interest are vehicles, road signs, and pedestrians. Background surfaces in these scenes include building facades, roads, and foliage. These systems have typically used long range LiDAR scanners in either stationary positions or mounted on moving platforms like trucks or airplanes. These scans may have variable density at different angles and distances, and poor registration with RGB imagery. Due to moving sensors or registration of multiple scans, the sensor viewpoint and raster scan order for any given scene point may not be available.

Despite these challenges, there has been great interest in collecting large scale 3D urban datasets for reconstruction and analysis. The task of object recognition in these scenes would be particularly useful for smart collision avoidance by self-driving cars and aerial delivery drones that would have to navigate these environments autonomously.

2.1.1 Object Detection Using Bottom-up and Top-down Descriptors

Patterson et al. [12] propose a system for detecting objects in large scale 3D datasets using a bottom-up and top-down process. In the bottom-up phase, local features are matched to database models to generate candidate object detections. These candidate objects are then verified in the top-down phase using a global feature descriptor computed over the whole candidate object. They evaluate their method on a large urban scene consisting of 200 million points with the task of detecting vehicles in the scene. Vehicles are one of the

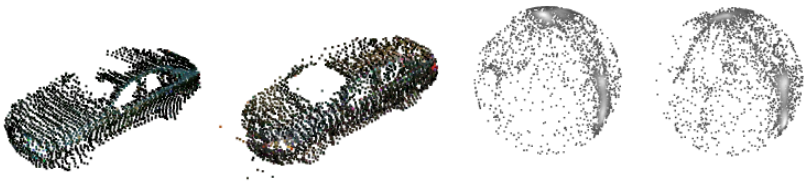

Figure 2: Patterson et al. [12]. Alignment of a database model (left car and left EGI) and a query (right car and right EGI).

Figure 3: Patterson et al. [12]. Left: Precision-recall curve as the sensitivity of the EGI comparison varies. Right: Detected vehicles with random colors, background with original colors.

most prominent and challenging objects to classify in urban environment and are the focus of many urban scene understanding projects.

Local spin image features are computed over a regular sampling of the original scene. These local features are relatively quick to compute and matched against spin images computed for the reference models in the database. Points with spin images that matched the database models are grouped together using simple region growing over a voxel grid of the scene to generate clusters of points representing the candidate objects.

For each candidate cluster an extended Gaussian image is computed, which is essentially a histogram of surface normal orientations. Unlike spin images the EGI is not rotation invariant and is more expensive to compare. To make the comparison more efficient the EGI is subsampled to include only points of maximal density, this is called a constellation EGI. Possible rotations to align two EGIs are made by attempting to align a pairs of points that subtend similar angles in the constellation EGIs. Only those candidate queries that match the EGIs of database models are considered as positive detections. An example alignment can be found in Figure 2.

In their experiment they select 17 vehicles as the training objects for their database. Although they use a variety of vehicles like sedans, SUVs and vans, they only report on the detection accuracy of objects as vehicles versus non-vehicles. After the bottom-up phase there were 2200 detections of which 1100 were correct. The top-down verification phase eliminated 1200 false positives

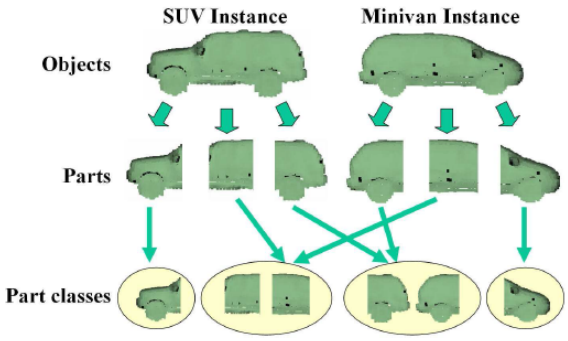

Figure 4: Huber [6]. Diagram of how objects are divided into parts and then each part is grouped into part classes. Here each car is split into a fixed number of parts. Parts are clustered together based on shape similarity. The middle and rear of SUVs and minivans may belong to the same part class but the shape of their fronts are more distinct.

but also 200 true positives. At the star indicated on the precision-recall curve in Figure 3, the system achieves 92.4% precision and 74.1% recall and there are 905 true positives, 74 false positives, and 316 false negatives. The main difficulty noted with this system is that it requires a linear number of comparisons in the size of its database and would probably not scale well to multiple classes.

2.1.2 Parts-based 3D Object Classification

When classifying objects of a globally similar set of classes it is helpful to model class variations using a part-based classification. The system of Huber et al. [6] segments objects into parts, classifies each part, and then determines the object class from the part classes. This approach is demonstrated by classifying point clouds of vehicles into eight vehicle classes.

Objects are segmented into parts and points on each part are densely sampled and a local feature, here the spin image is used, is computed at each point. Part classes are created in an unsupervised manner using agglomerative clustering where the distance between parts is determined by the probability that a feature in one part will be the closest match, among all features computed over the training set, to a feature in the other part. Each part class is then represented by the union of all the features belonging to all of the parts in the class. In order to reduce the complexity of matching and to generalize the features, the set of associated features is reduced to k prototype features using k-means clustering. An overview of this part class clustering procedure can be found in Figure 4.

Next, the part classes are used to determine the object class as follows. For each part class R_i and object class O_j the conditional probability $p(O_j|R_i)$ is estimated as

$$p(O_j|R_i) = \frac{p(R_i|O_i)p(O_j)}{p(R_i)},$$

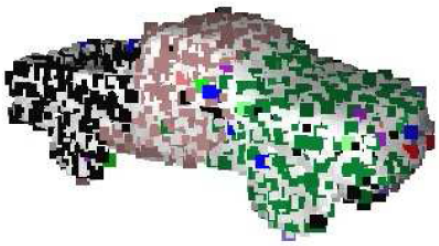

Figure 5: Huber [6]. An example query with features color coded by the most likely part in $\pi_{\mathcal{R}}$. Part classes tend to be consistently recognized with only a few outliers.

where $p(R_i|O_j)$, the distribution of parts given the object class, is determined empirically from the training set. The prior $p(O_j)$ may also be determined empirically but in these experiments is taken as a uniform distribution. $p(R_i)$ is determined using the total probability theorem: $p(R_i) = \sum_j p(R_i|O_j)p(O_j)$.

Now for each query containing $R_i \in \mathcal{R}$ parts, a distribution $\pi_{\mathcal{R}}$which corresponds to the probably of each part R_i belonging to each part class using the same method of determining distance as the agglomerative clustering step. This is essentially equivalent to a bag-of-words frequency vector corresponding to matches between spin image features and the k-means cluster centers for each part class. Then the class of each object is determined by the maximization,

$$j* = \max_j \sum_{R_i \in \mathcal{R}} \pi_{\mathcal{R}}(R_i)p(O_j|R_i).$$

In their experiments they segmented vehicles into three parts: front, middle, and back. Using the agglomerative clustering procedure they created 60 part classes, many corresponding to just one class but some part classes overlapped between different object classes, for example one part class was named minivan/SUV-back. These overlapping part classes are less useful for discriminating between the overlapping object classes. Each part class contained about 300 prototype features representing that part class.

In testing the object classifier achieved 96% accuracy with the correct class always contained in the top two results. However these experiments were based on simulated scans using synthetic models. While some measurement noise and self-occlusions were simulated, these experiments do not seem to incorporate the additional difficulty of variable density scans, ambient occlusions, moving sensors, and the effects of transparent and specular surfaces on the target object. Their chosen part segmentation also requires knowledge of the test object pose and they do not give a method for determining this.

2.1.3 Shape-based Recognition of 3D Point Clouds

Whereas the previous systems were specialized for detecting and then classifying vehicles, Golovinskiy et al. [5] present a more general system for classifying

Figure 6: Overview of Golovinskiy [5]. (a) Input point cloud colored by elevation. (b) Candidate object locations are marked with black vertical lines. (c) Candidate objects are segmented from the background. (d) Features describing shape and context are extracted. (e) Objects that have been classified.

multiple object classes within a point cloud. This system first localizes and segments objects before classification using a k-Nearest Neighbor graph over the point cloud weighted by the distances between points. An overview of this system can be found in Figure 6.

First, preprocessing steps remove the ground plane and buildings by removing large connected components with mostly horizontal or vertical surfaces as determined by surface normals at each point. Candidate objects are localized using an agglomerative clustering through the$k$-NN graph, with candidate locations at cluster centroids.

Using the candidate centroids, a min-cut graph segmentation is performed using the centroid as the source and a simulated background vertex as the sink. Every other point is connected to the background vertex with a weight based on the Euclidean distance from that point to the source and an expected background radius prior. This background radius prior can also be selected automatically by gradually increasing the prior until a sufficient number of points are contained in the segment and the cost of the cut is low. An example of this graph-based segmentation is show in Figure 7.

Global features are computed for each segment: the number of points, estimated volume, average height, standard deviation in height, and standard deviation in the two principal horizontal directions. Also a spin image is used as a global descriptor by placing it at the object center, using a large 2m radius, and orienting it using the vertical axis. Multiple segmentations are performed using different background radius priors and the features for each segmentation are concatenated together.

Additionally contextual features are computed based on geolocation of the acquired dataset using OpenStreetMap. One simple contextual feature is the distance of the object to the nearest street. Another contextual feature models where objects are likely to be with respect to each other. This is established using a 2D overhead grid oriented based on the closest street location, this grid is then populated using the class labels of other nearby objects. At test time an initial classification using all of the other features is used to create this grid feature for the final classification. Classification was tested using nearest neighbor, random forest, and SVM classifiers.

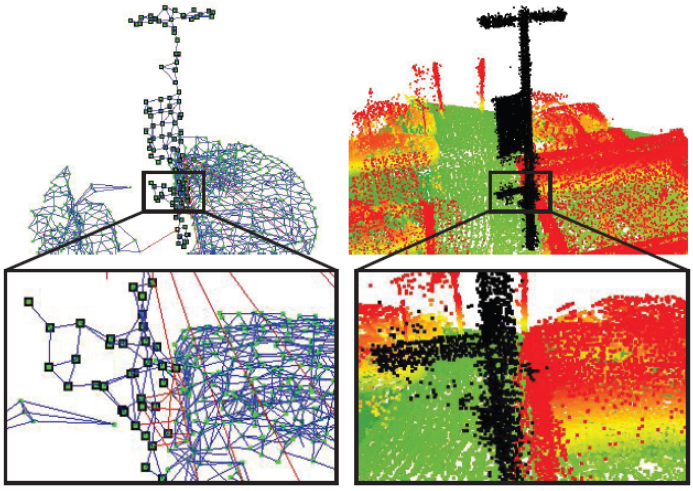

Figure 7: Golovinskiy [5]. Example of graph-cut segmentation. The k-NN graph is shown on the left with foreground object nodes bordered in black, edges in blue, and cut edges in red. On the right the extracted object is shown in black.

In experiments the localization and segmentation components achieve high recall but the classification only achieves 65% accuracy. This is likely due to the poor discriminative power of the selected features which for the most part are simple global statistics. A denser sampling of local features over each object as done in the other methods described may have increased the performance of this system.

2.1.4 Online Algorithms for Classification

Unlike the other systems in this section, the work of Stamos et al. [18] uses scans that are acquired in a raster order from a stationary LiDAR scanner, one vertical scanline at a time. The classification is performed purely on single scanlines using hidden Markov models and a quickest detection framework based on the cumulative sum (CUSUM) statistic to detect transitions from one HMM to another. The CUSUM statistic is simply defined as $S_0 = 0$ and $S_{n+1} = \max(0, S_n + x_n - \omega_n)$ where x_n is the nth observation and ω_n is the likelyhood of that observation given no change has occured. The CUSUM statistic becomes large when it's likely that a change has occured.

This method uses two simple features. One called a signed vector which takes the vector between two consecutives points k and $k+1$ in scanline i, denote this $D_{i,k}$, calculates the angle between it and the vertical axis, and determines a sign based on whether the vector is pointing away from or towards the scan origin. Based on this feature it becomes possible to classify points into vertical,

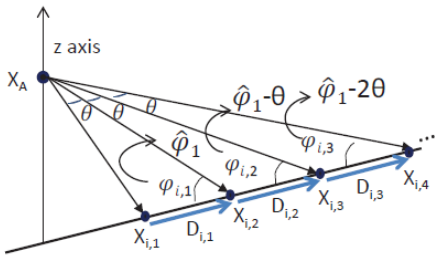

Figure 8: Stamos [18]. Four colinear points with line angels $\phi_{i,k} = \hat{\phi}_1 - (k-1)\theta$

horizontal, or scattered surfaces with scattered surfaces generally corresponding to vegetation. The HMM for a vertical or horizontal surfaces describes the probability of the next signed vector being 0 or 90 degrees as very high and anything else as low. For vegetation because the laser hits leaves at various depths it becomes equally likely that any angle could be observed and that the vector could point either towards or away from the scanner.

The other feature is called a line angle which has a consistent behavior when consecutive vectors between points are colinear; this feature is used to more reliably detect ground regions which tend to be colinear. Let X_A be the position of the scanner and $X_{i,k}$ be the position of the kth point in the ith scanline. Let θ be the angle the scanner's laser moves between consecutive points. Compute $\phi_{i,k}$ as the angle between vectors $D_{i,k}$ and $X_{i,k+1} - X_A$. If $X_{i,k}$and $X_{i,k+1}$ are colinear then $\phi_{i,k+1} = \phi_{i,k} - \theta$ and in general $\phi_{i,k+n} = \phi_{i,k} - n\theta$. This constant difference between angles for consecutive points is used for the line angle statistic $\hat{\phi}_k = \phi_{i,k} + (j-k)\theta$ for $j = k, k+1, ..., n$. See Figure 8 for an illustration of this concept.

The system uses these features to run several online algorithms for sequential classification for all of the scanlines. First a coarse classification into horizontal, vertical, and vegetation is obtained using the signed angles as described earlier. Then algorithms for ground, curb, and car detection are performed. Each of these can be seen as extensions of the coarse classification method that incorporates more prior knowledge to test additional alternate hypothesis. For car detection this is based on the observations that cars are located at a certain height above the ground plane, they should not appear behind the curbs of sidewalks or walls of buildings, and that there is a maximum height and width for a car and maximum height for a curb.

In the ground detection algorithm, points are labeled as ground using the line angle statistic. First colinear horizontal points near the beginning of each scanline are labeled as potential ground points and the height of the lowest point in each scanline z_i is recorded. The recorded heights in the first 50 scanlines are used to estimate the height of the ground z_g by using the mean-shift algorithm to find the dominant mode. Potential ground points are revisited to make

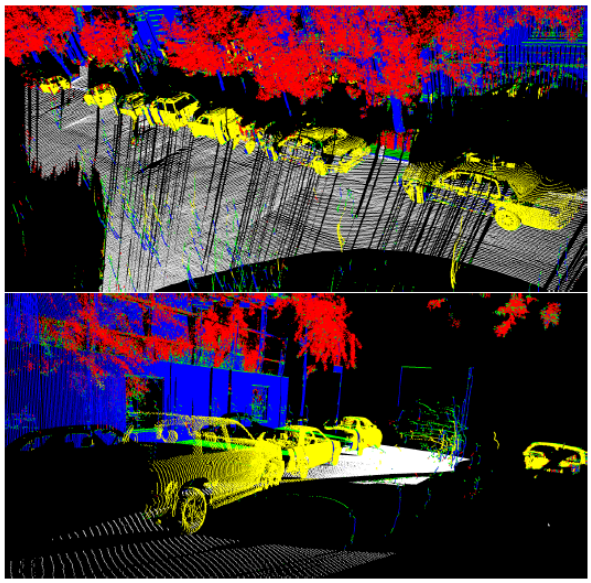

Figure 9: Stamos [18]. Classification results. Detected car points are displayed in yellow. Horizontal surfaces are green, vertical are blue, and vegetation is red. Ground points are shown in white.

sure they are close to the height z_g and are accepted as ground points if so. Successive horizontal points that are almost colinear may also be relabeled as ground if their height is close to z_g, growing the ground region at each scanline. For each scanline beyond the 50th, z_g is reestimated using the new z_i.

The curb detection algorithm finds potential curb points as the first vertical points that follow ground points and that have no points preceeding them in the scanline that are further away from the scanner. For example the points on the side of a car may be the first vertical points found in a scanline but it should be the case that some of the ground underneath the car was scanned and those points are further away from the scanner than the side of the car. Potential curb points are verified by first performing sequential region growing on potential curb points with vertical points. For each of these regions containing potential curb points, they must not exceed the expected height of a curb and either must span at least three scanlines or have small vertical curvature.

The car detection algorithm proceeds in several stages. First it uses a less sensitive version of the coarse classification algorithm that is less likely to label noisy parts of cars as vegetation. Then an online CUSUM statistic is used to find potential car points by comparing an HMM for mostly vertical or horizontal surfaces with one for cars that contain many transitions between vertical and horizontal due to their curvature, as well as missing data due to mirrors, windows, and reflective metallic surfaces. If a given scanline has potential car points then it is segmented into intervals and these intervals are tested for the

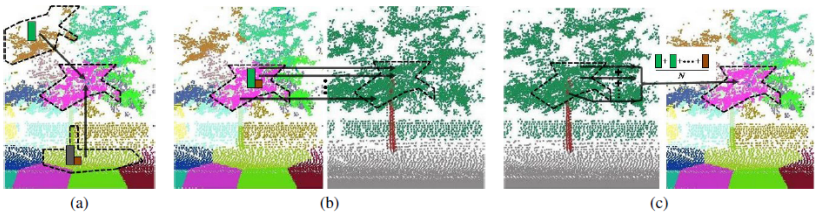

Figure 10: Overview of Xiong [20]. (a) Top level segments in different colors. The pink segment receives contextual information from its neighbors during stacking. (b) Information from the top level segment is sent to all of its consituent points in the bottom level. Points are colored by their class label for visualization. (c) After some rounds of stacking at the bottom level, contextual information from the points is averaged and sent to the top level segment.

presence of cars. Intervals that lie too far above the estimated ground height z_g, beyond a curb region, or beyond the horizontal distance from the scanner to large vertical regions that tend to correspond to walls, tree trunks, and street lights are eliminated from consideration. One final test seeks to distinguish low-lying vegetation from cars by taking the signed angle measurements in an interval and transforming them into the frequency domain with a Fourier transform. Intervals containing cars will have one dominant frequency whereas those with vegetation have more variation.

For car detection the system achieves a precision of 96% and a recall of 86%, classifying 7.3 million points across 11 urban scenes. Examples of classification results can be found in Figure 9. One common source of errors are spikes produced by moving cars and pedestrians that pass by too quickly for the scanner.

2.1.5 3D Analysis via Sequenced Predictions

While inspired by Markov random fields for modeling relationships and contextual information between segments but due to the intractability of doing exact inference over sophisticated MRFs, Xiong et al. [20] propose a system that models those relationships implicitly within the feature represenation. This system does this using a multi-stage inference procedure with a technique known as stacking. Similar to boosting, several classifiers are created during training but here they are used to generate contextual feature representations based on the distribution of class labels in neighboring regions. These intermediate classifier results are also used in further rounds of training instead of the ground truth labels in order to emulate the noisy conditions by which the inference procedure would be accomplished on data at testing time; the classifier model used here is logistic regression. Additionally there is a hierarchy of regions and each level in the hierarchy may engage in multiple rounds of stacking before sending the learned contextual features up or down the hierarchy.

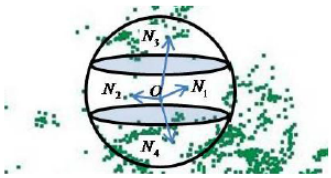

Figure 11: Xiong [20]. Neighborhood for computing contextual features.

In the described experiments there are two layers in the hierarchy, one corresponding to individual points and one corresponding to regions. The regions are defined using an oversegmentation by k-means clustering with k set to 1% of the number of points in the scene. Contextual features for a given point or a given region are then defined as the distribution of predicted labels within a fixed radius of the point or region centroid. This spherical neighborhood is further subdivided into a top, middle, and bottom slice to differentiate context by location. An overview of the procedure can be found in Figure 10 and an illustration of the spherical neighborhood within a region in Figure 11.

The stacking procedure generates multiple temporary classifiers for certain folds of the training data. Similar to cross-validation, the classifier for one fold is trained on the remaining data to prevent overfitting to that one fold. The various class probabilties of these classifiers for neighboring regions are then appended to the original feature vectors for those regions. This process can be repeated for multiple rounds, using the results from the previous round to further refine the predictions.

To pass contextual information through the hierarchy of regions, first contextual features are estimated for individual points at the bottom level using multi-round stacking. The average of the contextual features for the points within each region from the final round of stacking is used to initialize the multi-round stacking procedure for the top level region. After this the final contextual features at the top level can be passed back down to each point within that region and the stacking procedure can be reinitialized using these contextual features. This hierarchical parsing procedure can occur many times but in the experiments shown here there is one pass up from points to regions and one pass down from regions to points. An example of these learned contextual features may be found in Figure 12.

Additional non-contextual features are also used in their experiments. At the point level this includes scatter, linearity, and planarity which are based on the eigenvalues of the scatter matrix of a local neighborhood. Also the scalar projections of tangent and normal vectors on to the z-axis. Features for both points and regions include the dimensions of the bounding box in the principal component space, spin images about the z-axis, and relative elevations to high and low points in the surrounding area. For the top level segments there's an additional intra-region relative elevation feature as well as a histogram of all the contained point level features.

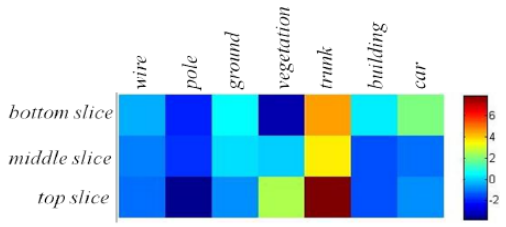

Figure 12: Xiong [20]. Contextual features for a tree-trunk region. It's most likely for a tree trunk to be adjacent to more tree trunks. It's also likely for part of the trunk to have vegetation above it or a car parked below it. The tree-trunk to vegetation relation is particularly useful since without color or texture it is the only way to distinguish thin trunks from poles.

The system is evaluated against both ground based and aerial datasets and achieves superior precision and recall scores for many classes compared to Markov network based approaches. The approach had more difficulty with a third dataset containing noisy labels and a background class that contained many different objects that could appear in a variety of contexts which contributed to confounding contextual features.

2.2 Comparison of Urban Object Classification

The urban detection systems described here use a variety of techniques to achieve their final classification result, but there are some common threads related to how they treat 3D data. Except for [18], none of these systems assume a raster order structure to the point cloud and utilize methods like kd-trees or clustering to efficiently organize the points for processing. Local features like the spin image are commonly utilized, with the exception of [5] which showed poor classification performance without them or another similar local feature. Global features tend to be aggregations of lower level features, although global shape descriptors that are challenging to align and compare like the EGI used in [12] are less common. Color is not used as a feature in any of these systems due to poor or non-existing alignment of the 3D sensors with color imagery.

Here we'll look more closely at how each system handles car detection, one of the most common classes for urban scenes. Also we'll compare the use of context, essentially nearby features that are not necessarily part of the object, and how this impacts classification.

2.2.1 Car Object Performance

Cars are one of the most commonly observed objects in urban datasets, their detection is important for safe navigation of urban environments but they introduce many challenges. Vehicles have complex varying geometry relative to most other objects which either have smooth horizontal or vertical surfaces or

scattered noise like vegetation. Highly specular surfaces on cars can deflect a sensor's laser resulting in missing points, and scans through a car's windows can result in noisy data on the interior surface. Self-occlusions can completely change the observed shape of the vehicle depending on the relative orientation to the scanner and there are often ambient occlusions due to other cars on the road.

The localization and segmentation efforts of Golovinskiy [5] encountered particular difficulty with cars, with estimated locations being in the more densly sampled hood or trunk areas and resulted in oversegmentation due to the uneven sampling and large shape of the vehicles. Xiong et al. [20] also observed difficulty based on sampling when comparing performance between ground-based and aerial datasets. In aerial datasets vehicles are more sparsely sampled and lose their distinctive geometry making them appear similar to low-lying vegetation. Even on ground-based scans their method does not perform as well for vehicles as the Exemplar SVM based approach of Lai and Fox [9] which utilizes domain adaptation from CAD models to provide additional training examples for cars, however the comparison is not entirely fair because the dataset was annotated for Exemplar SVMs in mind with many diverse hard negative examples contained in the background class.

Huber [6] focuses on subclasses of vehicles in isolation and achieves high accuracy on this task. A parts-based classification may be a good way to address the complex shape and variations of vehicles. However it's not clear if this specific approach would scale to include more different classes and how objects in those classes would be segmented into parts in a way that is comparable to the vehicle classes. This method would also be dependent on another localization and segmentation method for objects of interest as a preprocessing step.

At this point it seems that the methods specifically designed for car detection achieve the best performance for detecting this challenging object category. The EGI global shape descriptor used in Patterson [12] is able to model the unique shape properties of vehicles. Distinct shape properties and context of cars in urban scenes are also modeled explicitly in Stamos [18] for detecting boundary transitions from background classes to car. Although these methods may have difficulty scaling to more object classes, the more general classification algorithms need to be able to automatically learn these sophisticated structures that can be manually designed.

2.2.2 Use of Context

Although it may be expected for an ideal vision system to correctly detect and classify a visible object in many different environments, for modern practical systems the use of contextual cues within a target domain can disambiguate certain classes, eliminate false detections and improve overall performance. This is particularly useful in long-range LiDAR scans where sparse sampling and unreliable RGB alignment may make local features indistinct and noisy. Basic features like spin images do implicitly embed context by accumulating data points from nearby objects and the background, but some systems have shown improvement

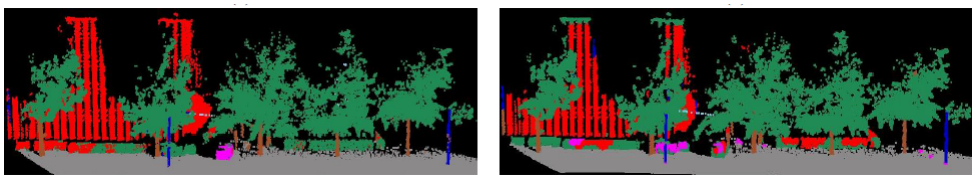

Figure 13: Left: Xiong [20]. Bushes at the base of the building are incorrectly classified as building. Right: The max-margin Markov network does better on those bushes but incorrectly classifies the sparsely scanned building tops as vegetation.

by more explicitly modeling contextual features and scene structure.

Two of the methods described here use preliminary classification to provide semantic context for a final classification. Although overall accuracy was low in Golovinskiy [5], they show a +0.05 increase in precision by using their contextual features. This may be further improved considering the contextual features that localize objects with respect to other objects being classified were likely noisy due to reliance on the poor classification results for those other objects using the non-contextual features. Xiong et al. [20] present a more sophisticated procedure for generating features for context using the stacking technique. This yielded a gain of +0.09 precision and +0.31 recall over the max-margin Markov network approach of Munoz et al. [11]that only enforced similar labels between neighboring points without explicitly modeling a semantic context using class labels.

The sequential detection algorithm of Stamos [18] also utilizes context explicitly in its rules for detecting the car class by eliminating impossible locations in the scan. Similarily some of the geolocation features used by Golovinskiy [5] also give explicit context to the possible locations of object classes within an urban scene.

While context gives strong cues for making correct predictions, it may also cause undesirable smoothing or hallucinations that conform with learned contexts. For example, as Xiong et al. show in Figure 13 that vegetation directly underneath a building may be misclassified as part of the building due to the strong prior that the most common neighboring region below a building region is another building region; this vegetation was correctly classified by Munoz et al. [11] with weaker use of context. We will look more closely at Markov network based approaches in the following section when we consider Silberman [14] and Anand [1].

3 Indoor Object Classification

3.1 System Overviews

In more recent years the availablility of fast and affordable RGB-D cameras like the Kinect have provided 3D data that is well aligned with traditional RGB images and maintains a similar image grid structure. The technologies of these cameras let them acquire data more quickly than older LiDAR scanners but have a much more limited effective range and so applications for these cameras are typically for indoor environments. Because of the similarity to traditional camera images, techniques that utilize colors and grid structure have been imported from the 2D computer vision literature on object classification and semantic segmentation.

3.1.1 Indoor Scene Segmentation using a Structured Light Sensor

Utilizing a manually operated Kinect camera with a counterweight for stabilization, Silberman et al. [14] acquired RGB-D images of indoor scenes such as living rooms, and offices, capturing both residential apartment spaces and workplace settings. Image frames were extracted every 2 - 3 seconds from the acquired RGB-D videos for a total of 2347 images across 64 different scenes. These images were densely annotated at each pixel by Amazon Mechanical Turk workers using the LabelMe [13] interface. Using the RGB-D data Silberman et al. present a conditional random field model that utilizes representations for depth information and a prior based on 3D location.

The RGB-D images are pre-processed to eliminate errors in alignment and noise introduced in reconstruction. The Kinect contains two cameras, a conventional VGA camera and an IR camera that reconstructs depth using a structured light pattern emitted by an infrared projector. Although these two cameras are factory calibrated, Silberman et al. recalibrate using checkboard images. Despite this there still remain artifacts in the depth images from the reconstruction process, the displacement between the IR camera and projector, and the reflectance of some surfaces. To correct these errors they apply an edge-aware bilateral filter to the depth maps that fills in missing depth values while respecting edges in the corresponding RGB image. Finally pitch and roll in images with respect to the direction of gravity are eliminated using the Kinect's accelerometer.

The conditional random field energy function $E(\mathbf{y})$ measures the cost of the label vector $\mathbf{y}$ where each component y_i labels pixel i in an image with a value in $\{1, ..., C\}$ where C is the number of class labels. Their energy is composed of three potentials: a unary cost function $\phi(x_i, i; \theta)$ that depends on pixel i with features x_i and learned parameters θ, a pairwise cost $\psi(y_i, y_j)$ between labels of adjacent pixels, and a spatial smoothness term $\eta(i, j)$ between adjacent pixels.

$$E(\mathbf{y}) = \sum_i \phi(x_i, i; \theta) + \sum_{i,j} \psi(y_i, y_j)\eta(i, j)$$

Before applying the CRF directly, a low level super-pixel segmentation is performed on both the RGB images, denoted S_{RGB}, and the combined RGB-D images, S_{RGBD}, and are used when computing the unary and spatial terms of the energy function.

The unary potential function ϕ is computed as the product of two terms, a local appearance model $P(y_i|x_i, \theta)$ and a location prior $P(y_i, i)$, with $\phi(x_i, i; \theta) = -\log(P(y_i|x_i, \theta))P(y_i, i)$.

The appearance model is discriminatvely trained using features extracted over a densely sampled grid and at various scales. The feature for a given grid point is the concatenation of the features extracted at that point at each scale. The appearance model parameters are trained using a neural network with a single hidden layer and a softmax output layer. After training, for each super-pixel s the probabilities $P(y_i|x_i, \theta)$ are averaged for each sampled descriptor x_i falling within s and this mean class probablity is assigned to every pixel in s. Silberman et al. test several combinations of features for the appearance model including SIFT for RGB images, SIFT computed on the depth map, and spin images from the depth map.

For the location prior $P(y_i, i)$, Silberman et al. present a prior based on 3D information that outperforms a standard 2D location prior that simply uses image pixel coordinates. Their 3D prior is based on relative depth of objects to the interior walls of the scene with an emphasis on objects that tend to be clustered near walls and some invariance to the position of the camera in the empty areas of the room. For each column of the depth map, the pixel at the greatest depth is assumed to lie on the bounding hull. All points within a column are rescaled such that the greatest depth $\tilde{z} = 1$. Within this reference frame a 3D histogram of labels are computed over all images with finer bins at relative depths close to boundary and coarser bins at depths close to the camera. Finally the histogram is normalized such that for bins$i$, $\sum_i P(y_i, i) = 1/C$, representing a uniform prior for each class within the image. Examples of how this prior varies over depths for four classes can be found in Figure 14.

The pairwise potenetials are determined using a simple Potts model where $\psi(y_i, y_j) = 0$ when $y_{i,} = y_j$ and $d = 3$ otherwise. This simple model is used to highlight the impact of depth information in the other potentials.

The spatial transition potential effects label transitions independent of proposed class labels using the image gradient $|I(i) - I(j)|$. The potential takes the form

$$\eta(i, j) = \eta_0 e^{-\alpha \max(|I(i)-I(j)|-t,0)},$$

where t is a gradient threshold and $\alpha$and η_0 are scaling factors. These parameters were varied as they were tested on gradients defined for the RGB image, the depth map, a convex combination of the two, and a restriction of gradients to superpixel boundaries.

To evaluate the CRF, the original annotations containing over 1000 different labels was reduced to 12 common categories using Wordnet synonym structure as well as a background class containing rare objects. These classes include room

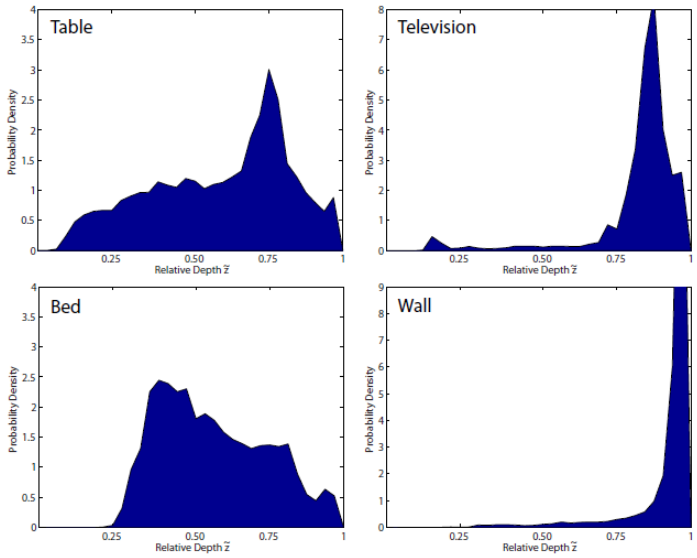

Figure 14: From Silberman [14]. Probability densities in relative depth direction for table, television, bed and wall classes. Walls are on the boundary and cluster tightly at relative depth 1. TVs are often placed near walls whereas beds and tables occupy more of a room's interior space.

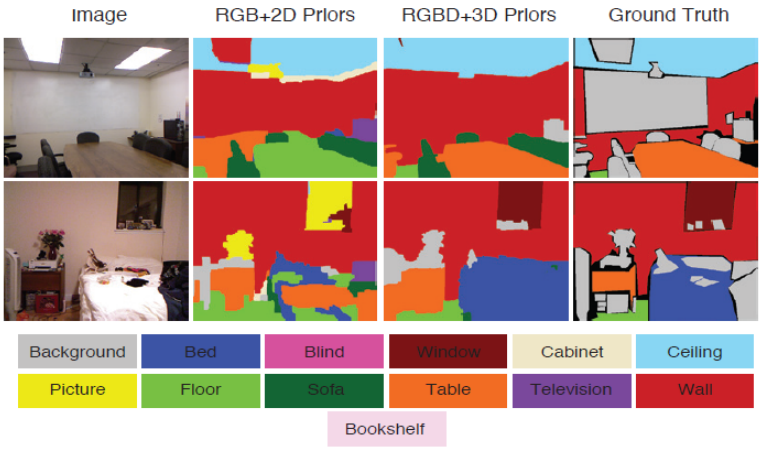

Figure 15: Example image labelings from Silberman[14]. Implausible errors such as cabinets on the ceiling and floors on the bed are corrected using 3D data.

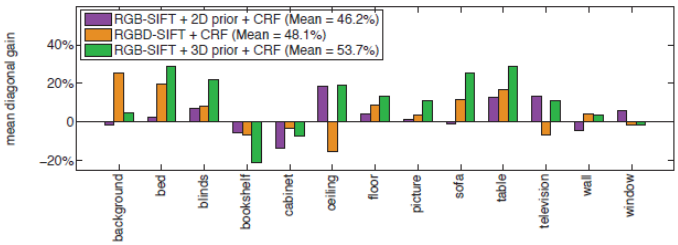

Figure 16: From Silberman [14]. Changes in performance for each class compared to a baseline CRF with no location prior. Several classes see an increase in accuracy using 3D features and the 3D location prior.

boundries such as wall, ceiling and floor; objects found on walls like picture, window, and cabinet; and large furniture like bed, table, and sofa. In total these common categories make up 47.2% of the pixels across all images. To measure performance of the CRF under different configurations they take the mean of the diagonal entries of the confusion matrix, essentially giving a mean accuracy across classes. Example labelings can be found in Figure 15.

Comparing different features for unary potentials the best performance was with SIFT using both RGB and depth maps with a mean accuracy of 48.1%, slightly outperforming the use of RGB SIFT with spin images for the depth map at 45%. This seems to indicate that there are advantages to the more sophisticated structure of the SIFT descriptor that can be utilized in 3D data. The 3D location prior further increases the mean accuracy to 53.7%. For some classes this achieves a 29% increase in accuracy over a baseline approach that does not use 3D features or priors. Figure 16 shows changes in performance for each class using different configurations of the CRF.

For the different choices of gradients for spatial transition priors the best results were with RGB edges achieving 56.6% accuracy and no significant impact from depth edges or the superpixel constraints. It's likely that edges in the depth map contribute little here because these edges only correspond to major spatial discontinuities that are likely already captured in the features for the unary potentials, whereas RGB edges can separate adjacent objects at similar depths.

3.1.2 Semantic Segmentation using Depth Information

The work of Couprie et al. [3] incorporates depth as an additional channel of a convolutional neural network (CNN) on an updated version of the dataset collected by Silberman et al. [14]. A CNN is a hierarchical architecture designed for taking an entire image as an input and producing a feature representation or classification as the output without requiring domain specific knowledge or feature engineering.

A CNN is composed of several stages, in practice typically between one to three stages, containing three layers each: a filter bank layer, a non-linearity

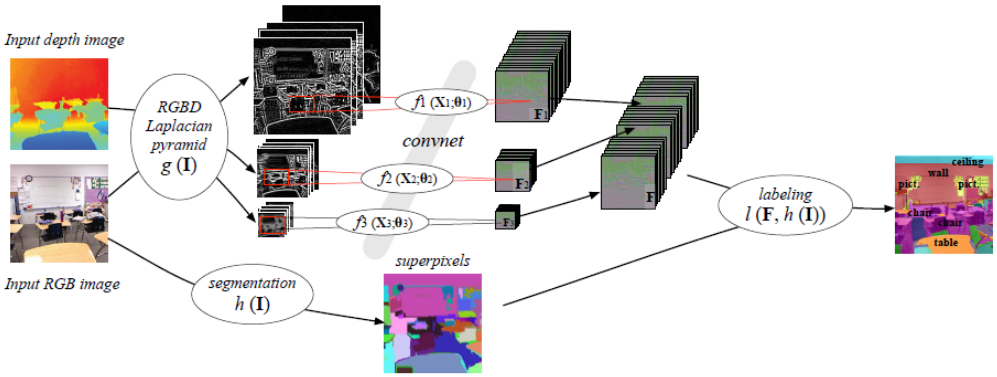

Figure 17: Overview of Couprie [3]. Given the input RGB-D image pair, a Laplacian pyramid generates normalized images at multiple scales. A CNN generates representations for each scale, finer scale representations are upsampled and all scales are concatenated for the final feature vectors for each pixel. A superpixel segmentation of the original color image enforces smooth labeling within similar regions.

layer, and a feature pooling layer. The filter banks act like a set of convolution filters for different regions of the input image for a given stage, these filters tend to become similar to edge detectors in the first stage. The non-linearity layer normalizes the output of the filter banks and transforms it in a way that the linear filter bank operations would not. The pooling layer subsamples the results of the previous layer to produce a more compact and general representation for future stages.

Due to the fixed size filter banks in the CNN, Couprie et al. utilize a multi-scale CNN that creates a Laplacian pyramid representation of the input RGB-D images at multiple scales. This allows the CNN to generate features at various scales of context. Similar to the unary features of Silberman [14], the CNN features produced at each scale are concatenated together for the final representation.

The final representation is classified using a two layer multilayer perceptron and then smoothed using a superpixel segmentation of the original RGB image by aggregating pixel labels into the superpixels that contain them. Figure 17 provides an overview of the whole procedure.

The authors evaluate the work on the NYU depth dataset using two different clusterings of object categories. The first set of experiments contains fourteen object categories, some example scenes that have been labeled with a multiscale CNN using only RGB and one also using depth information can be seen in Figure 18. The use of depth information improved accuracy for background surfaces like walls and floors, but with the exception of the furniture category the accuracy for more objects of interest decreased and the improvement in

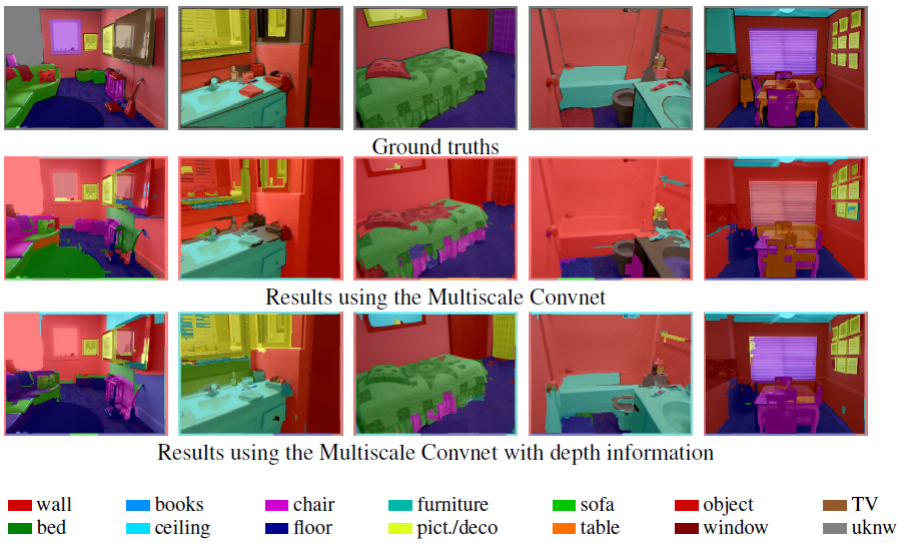

Figure 18: Couprie [3]. Sample results. The use of depth in the multiscale CNN seems to correct errors in classes that have consistent depth measurements across images such as floors, ceilings, and walls, but does not necessarily learn the shapes of objects of interest that can appear in various locations.

average class accuracy is small.

In a four class formulation of the dataset, the multiscale CNN with depth again finds improvement on background categories like ground and structural supports but loses accuracy compared to a baseline method attempting to detect props and furniture that are supported by the previous categories in these scenes. Here the furniture category is defined differently and includes chairs and beds but not desks and cabinets.

While the CNN is able to produce gradient based features similar to the depth map SIFT features that worked well in Silberman [14], a lack of depth normalization and interpreting depth values the same way as colors seems to have a negative impact on depth-variant object categories.

3.1.3 Contextually Guided Semantic Labeling

Anand et al. [1] present a Markov Random Field approach to detecting objects in RGB-D scenes that utilizes many features and contextual relationships. Unlike Silberman [14], this work uses additional pairwise features for inference. Here inference is done efficiently using mixed integer programming and efficient learning is accomplished with structural SVMs.

This method utilizes a complete point cloud registered from RGB-D images. The initial point cloud is oversegmented based on difference in surface normals and distances between points. These segments are the basic units that are

Node features for segment i.

Description	Count
Visual Appearance	**48**
N1. Histogram of HSV color values	14
N2. Average HSV color values	3
N3. Average of HOG features of the blocks in image spanned by the points of a segment	31
Local Shape and Geometry	**8**
N4. linearness (λ_{i0} - λ_{i1}), planarness (λ_{i1} - λ_{i2})	2
N5. Scatter: λ_{i0}	1
N6. Vertical component of the normal: $\hat{n}_{iz}$	1
N7. Vertical position of centroid: c_{iz}	1
N8. Vert. and Hor. extent of bounding box	2
N9. Dist. from the scene boundary (Fig. 5)	1

Edge features for edge (segment i, segment j).

Description	Count
Visual Appearance (associative)	**3**
E1. Difference of avg HSV color values	3
Local Shape and Geometry (associative)	**2**
E2. Coplanarity and convexity (Fig. 5)	2
Geometric context (non-associative)	**6**
E3. Horizontal distance b/w centroids.	1
E4. Vertical Displacement b/w centroids: $(c_{iz} - c_{jz})$	1
E5. Angle between normals (Dot product): $\hat{n}_i \cdot \hat{n}_j$	1
E6. Difference in angle with vertical: $\cos^{-1}(n_{iz})$ - $\cos^{-1}(n_{jz})$	1
E8. Distance between closest points: $\min_{u \in s_i, v \in s_j} d(u, v)$ (Fig. 5)	1
E9. Relative position from camera (*in-front-of/behind*). (Fig. 5)	1

Figure 19: Summary of features used in Anand [1].

classified.

The model used in this work attempts to capture three properties of objects in RGB-D scenes: visual apperance in terms of color and gradient; shape and geometry in terms of orientation, smoothness, and convexity; and geometric context in terms of the expected relative placement of objects such as on-top-of or in-front-of relations.

In the model, a discriminant function captures the dependencies between segment labels as defined by an undirected graph (V, E) where V is the set of segments and there is an edge between two segments if there exist two points in each segment that are within a distance threshold called the context range. The discriminant function is defined on segmented point cloud x, class labels y, weight vectors w, segment features $\phi_n(i)$ and edge features $\phi_t(i, j)$ as

$$f_w(x, y) = \sum_{i \in V} \sum_{k=1}^{K} y_i^k [w_n^k \cdot \phi_n(i)] + \sum_{(i,j) \in E} \sum_{T_t \in T} \sum_{(l,k) \in T_t} y_i^l y_j^k [w_t^{lk} \cdot \phi_t(i, j)].$$

Here there are multiple types t of edge feature maps $\phi_t(i, j)$ and each type has a graph over the K classes with edges T_t, defining relationships between those classes. We define an edge feature map t to be modeled by an associative edge potential if the corresponding graph only has self loops, that is $T_t = \{(k, k) | \forall k = 1..K\}$, and is non-associative if the corresponding graph is fully connected. Typically previous models have only used associative models to smooth labeling between visually similar points of the same class, but this excludes modeling semantic relationships like geometric context between different classes. But using a non-associative model increases the number of parameters to K^2. The authors introduce here the idea of a parsimonious model that uses both associative and non-associative potentials depending on the edge features being modeled.

A summary of all of the unary and pairwise features modeled by this approach is available in Figure 19. Many of the segment features we have already seen used previously by [20, 14], but here we also have pairwise features that

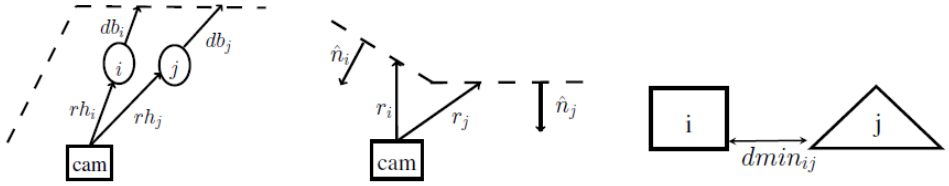

Figure 20: Feature illustrations, Fig 5 from Anand [1]. Left: N9 and E9, segment i is in front of segment j if $rh_i < rh_j$. Middle: E2, two segments form a convex shape if $(r_i - r_j) \cdot \hat{n_i} \geq 0$ and $(r_j - r_i) \cdot \hat{n_j} \geq 0$. Right: Feature E8.

Figure 21: Anand [1]. Classification results for office (top) and home (bottom) scenes. Left-most is the original point cloud, middle is the ground truth labeling, and right is predicted labels. Some visible errors include the monitor labeled as cpuFront in the office scene on the top right and the pillow labeled as bed in the home scene.

describe geometric relationships between objects in the scene. Illustrations of some of these features are show in Figure 20.

For their experiments Anand et al. collected 24 office and 28 home scenes with a total of 550 views with each domain having some different class labels present. Many of their self-selected categories correspond to planar components of semantic objects such as chairBack and chairBase instead of simply chair, it seems like having object part categories better exploits the shape and context features that they've designed and they indeed report a drop in performance when using object level categories for all parts.

Interestingly they also attribute some of their success to having registered 3D models versus single-view RGB-D images. Here they report implicit benefits in terms of more complete views reducing errors from single view acquisitions and larger registered scenes containing more interdependent context between segments.

Overall their system achieves high classification accuracy for the office setting

and nearly as good performance for the home domain. Example scenes can be found in Figure 21. The most common errors include objects being assigned the label of the object they're placed on top of, such as a book resting on a tableTop or a pillow on top of a bed. This is likely either due to poor segmentation of the object from the surface or undesirable smoothing from the contextual potentials.

3.2 Comparison of Indoor Object Classification

Recent interest in synthesizing 2D and 3D data for object classification has created many new directions for research in how to best utilize techniques from both domains. Unlike in the urban scene setting, there is no single class that is as ubiquitous and challenging as the car. For indoor settings the most challenging aspects are the wide variety of object categories and large variations in how they are arranged. Although relatively large 3D datasets have been collected, the limited availability of labeled data for training these systems seems to impact the rates of successful classification for rarely observed object classes.

Of the three indoor classification systems surveyed, the system of Anand [1] achieves the best accuracy on its evaluation sets and presents fewer errors in its example scenes than the other two methods and this seems largely due to the non-associative context relations between objects of different classes. Comparisons between the office and home domains in Anand [1] show that there are differences in difficulty between well structured scenes with regularly shaped objects and more cluttered scenes with irregular objects. The visually simpler classes and more structured environment of the office allow the system to make more reliable classification. The use of object-part classes skirts the difficulty of overly complicated 3D shape representation and they show it can be useful for simple robotic applications involving a robot that can search for object parts and for example open a desk drawer.

Silberman [14] does seem to achieve similar performance to Anand's home dataset for similarly cluttered indoor environments with more general object categories, although few of these would qualify as objects of interest for interaction. However just the SIFT based features used by Silberman do better than the baseline unary features of Anand and the 3D location prior used by Silberman seems more sophisticated than the simpler array of location features used by Anand and is responsible for closing the gap.

The system of Couprie [3] does not produce competitive results but presents a possible framework for automatic feature learning that can utilize depth information. It seems that because depth information was not normalized or directly related to room boundaries or viewer location as in the other methods, only structural elements that appeared at constant depth throughout the dataset benefited from this added information. In contrast, Silberman was able to apply the standard 2D SIFT feature to depth maps directly with greater success and further correct errors with a 3D location prior. However we see from the large collection of features used by Anand and previous approaches that researchers are accumulating many disparate features that may not give the best representation of the objects for classification; an automatic method

for generating optimized 3D representations like a convolutional neural network is a promising research direction.

4 Overall Comparison

Over time we see that the landscape of research in 3D object classification has changed significantly with changes in camera technology and as a result increased focus on indoor scenes. These scenes tend to be more cluttered with more potenital objects of interest that are difficult to detect in comparison to point clouds of urban scenes produced by LiDAR sensors. There's also an increased diversity of object classes through varying indoor scene domains.

The fundamental structure of the acquired 3D data has also changed. Depth mapped images have allowed the use of more traditional grid based features like SIFT whereas techniques for unstructured point clouds have relied more on geometric shape properties.

4.1 Number of Classes

Urban classification systems have largely focused on car detection in urban environments with other structural elements like buildings, signs, and vegetation having well-ordered properties that are easier to distinguish. As such the number of classes in these systems has been relatively small. Indoor scenes are more densely populated with objects that would have semantic meaning for interaction and would be of interest in practical applications. Unfortunately this leads to an explosion of possible classes that may be confused with each other and confound classification algorithms.

Silberman and Couprie attempt to cope with this variety by clustering ground truth labels, which vary widely because they were collected by many independent workers, into a limited set of semantic categories. But this results in a large set of objects relegated to a background or unknown object class. In experiments on a noisy dataset with a similar catch-all background class, Xiong reported difficulty due to the inability to learn correct contextual features from these labels compared to simpler datasets with cleaner labels.

The solution of Anand seems to use object-part categores that may be more easily distinguished but seems to work best in the office domain with more regularly structured objects. This is similar to Huber's approach to part-based classification of vehicles and so may benefit from some additional part class clustering and incorporation of a model between parts of the same object category. Perhaps a hierarchical model of parts, objects, and context would best explain and cope with the large number of relationships that would have to be modeled.

4.2 Point Clouds vs RGB-D

Difference in camera technology have impacted the selection of techniques for feature representation and modeling dependencies. Many of the systems that utilize unstructured point clouds or registered RGB-D scenes [1, 5, 20] have relied on an oversegmentation of the point cloud and nearest neighbor distance relations to establish context dependency between object classes and the background.

For uncolored point clouds there has also been a greater emphasis on shape based description such as the EGI used in Patterson or many of the features used in Xiong that were adopted by Anand like scatter, linearity, and planarity. The systems of Silberman and Couprie have relied more on color based features and have used depth based features largely as an additional color channel rather than as an explicit shape based description of objects.

The convolution neural network used by Couprie relies on the grid structure of the depth map to produce features so adapting this kind of feature learning to unstructured arbitrary 3D datasets may be difficult. However 3D shape feature descriptions have remained relatively primitive, like spin images, or difficult to compare, like EGIs, and so some recent work has focused on the development of more sophisticated 3D shape descriptions for object classification. Bo et al. [2] utilizes sparse coding and spatial pyramid max pooling to construct multi-scale features for color, depth, grayscale, and surface normal channels which achieve high recognition performance on the UWash RGB-D Objects dataset, a set of 300 common household objects placed on a turntable and recorded by a RGB-D camera. They note that while small scale color based features are useful for identifying object instances, for example different boxes of cereal, the shape based features help distinguish between object categories.

4.3 Related Works

In addition to the highlighted approaches there have been several recent related works that should be mentioned.

Kahler and Reid [8] present a similar MRF based approach to Anand but using Decision Tree Fields and Regression Tree Fields instead of structural SVM for the inference procedure leading to improved computational time while maintaining similar performance. Interestingly they also report the number of times each feature is used as a splitting rule in a decision tree, giving a rough estimate of the importance of each feature. Although this is given as a rough estimate, it seems that evaluating feature importance and producing representations with only relevant features will become increasingly important given the increasing number of available 3D shape features to choose from.

The work of Muller and Behnke [10] also utilizes a CRF approach where inference is still performed using structural SVM but unary potentials are learned using the results of a per-pixel random forest classifier as well as a rough estimate of pixel height. For the pairwise potentials between segments they use simple differences between color, depth, normal orientations, and alignment along the

vertical direction. Despite the simple design of their features they achieve competitive performance on the NYU Depth dataset, with a large improvement on the furniture category over Couprie.

Another feature learning based approach is that of Socher et al. [16] which utilizes a combination of convolutional and recursive neural networks to generate. Low level features are extracted from RGB and depth map images using a single CNN layer that has been trained in an unsupervised fashion. These features are recursively merged by the RNN into single feature vectors for color and depth that are concatenated and given to a softmax classifier. Interestingly they show that rather than training a single RNN using backpropagation, they achieve slightly superior performance by simply using an ensemble of randomly initialized RNNs and concatenating all of their feature vectors. They achieve competitive performance with the sparse coding method of Bo et al. [2] on the UWash RGB-D Objects dataset, which also relies on features learned in an unsupervised manner. The key difference is that Bo et al. also learn features over surface normals which are estimated using the depth information and that Socher et al. disregard as an additional input modality. Also because Socher et al. only utilize top level features they are able to achieve similar performance with lower dimensional features by an order of magnitude.

The sliding shapes approach of Song and Xiao [17] is more similar to prior work from traditional computer vision using a sliding template to detect object occurances but uses the Exemplar-SVM framework with depth based features for matching and CAD models for training. The template used also moves in the 3D space of the captured scene and uses a 3D occupancy mask based on each training exemplar to deal with occlusions. Using the purely 3D features they show superior performance on the NYU Depth dataset compared to two state-of-the-art 2D classification systems, deformable part models [4]and convolutional neural networks, for five object categories: chairs, beds, toilets, sofas, and tables. However the approach is computationally expensive due to the large number of exemplars that need to be matched, including multiple views rendered for each reference object, but may be parallelized to some extent.

Recently Silberman et al. [15] have also introduced the problem of separately labeling different instances of a given class within a scene, this is performed using a structured learning scheme for cutting a segmentation tree. In a segmentation tree smaller segments are merged such that each ancestor segment entirely contains its descendents and an optimal cut of this tree, selecting one node per branch, will provide a semantic segmentation of the given scene. Although this work does not focus on using RGB-D data, because it used the NYU Depth dataset which contains instance information it was available and they had best results using a combination of 2D CNN features and depth based features. The problem presented is also a natural extension of the object classification and semantic segmentation tasks.

Finally the work of Wu et al. [19] provides a CRF based semantic labeling but now with a hierarchy of semantic labels that can model is-part-of and is-type-of relations and similar to [15] this labeling is also performed over a segmentation tree. This approach addresses the problem found in Anand where

semantic labels of parts like chairBack did not give any relation to associated semantic labels like chair or furniture. In the segmentation tree they allow ancestor supersegments to take on ancestor labels in the semantic hierarchry, so for example a subsegment of a chair region may be labeled as chairBack. This is another natural extension of the semantic segmentation problem and may reduce errors by enforcing hierarchical contraints.

5 Conclusions

Throughout the survey we've examined systems for both urban and indoor object classification and semantic segmentation with both unstructured 3D data and color aligned depth maps. Features based on geometrical shape, context, and the relationships between object classes and parts have consistently provided strong improvements to overall classification accuracy.

We see future directions in this area including a structured hierarchy between classes, parts, and context to handle the number of different objects in more complex scenes, as well as learned representations for 3D shape features that are trained for the object classification task.

References

[1] Abhishek Anand, Hema Swetha Koppula, Thorsten Joachims, and Ashutosh Saxena. Contextually guided semantic labeling and search for three-dimensional point clouds. *The International Journal of Robotics Research*, 32(1):19–34, 2013.

[2] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Unsupervised feature learning for rgb-d based object recognition. In *Experimental Robotics*, pages 387–402. Springer, 2013.

[3] Camille Couprie, Clément Farabet, Laurent Najman, Yann Lecun, et al. Indoor semantic segmentation using depth information. In *Proceedings of the International Conference on Learning Representations*, pages 1–8, 2013.

[4] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.

[5] Aleksey Golovinskiy, Vladimir G Kim, and Thomas Funkhouser. Shape-based recognition of 3d point clouds in urban environments. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2154–2161. IEEE, 2009.

[6] Daniel Huber, Anuj Kapuria, Raghavendra Donamukkala, and Martial Hebert. Parts-based 3d object classification. In *Computer Vision and Pat-*

tern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, volume 2, pages II–82. IEEE, 2004.

[7] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5):433–449, 1999.

[8] Olaf Kahler and Ian Reid. Efficient 3d scene labeling using fields of trees. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3064–3071. IEEE, 2013.

[9] Kevin Lai and Dieter Fox. Object recognition in 3d point clouds using web data and domain adaptation. *The International Journal of Robotics Research*, 29(8):1019–1037, 2010.

[10] AC Muller and Sven Behnke. Learning depth-sensitive conditional random fields for semantic segmentation of rgb-d images. In *International Conference on Robotics and Automation*, 2014.

[11] Daniel Munoz, James A Bagnell, Nicolas Vandapel, and Martial Hebert. Contextual classification with functional max-margin markov networks. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 975–982. IEEE, 2009.

[12] Alexander Patterson IV, Philippos Mordohai, and Kostas Daniilidis. Object detection from large-scale 3d datasets using bottom-up and top-down descriptors. In *Computer Vision–ECCV 2008*, pages 553–566. Springer, 2008.

[13] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[14] Nathan Silberman and Rob Fergus. Indoor scene segmentation using a structured light sensor. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 601–608. IEEE, 2011.

[15] Nathan Silberman, David Sontag, and Rob Fergus. Instance segmentation of indoor scenes using a coverage loss. In *Computer Vision–ECCV 2014*, pages 616–631. Springer, 2014.

[16] Richard Socher, Brody Huval, Bharath Bath, Christopher Manning, and Andrew Ng. Convolutional-recursive deep learning for 3d object classification. In *Advances in Neural Information Processing Systems 25*, pages 665–673, 2012.

[17] Shuran Song and Jianxiong Xiao. Sliding shapes for 3d object detection in rgb-d images. In *European Conference on Computer Vision*, volume 2, page 6, 2014.

[18] Ioannis Stamos, Olympia Hadjiliadis, Hongzhong Zhang, and Thomas Flynn. Online algorithms for classification of urban objects in 3d point clouds. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 332–339. IEEE, 2012.

[19] Chenxia Wu, Ian Lenz, and Ashutosh Saxena. Hierarchical semantic labeling for task-relevant rgb-d perception. In *Robotics: Science and Systems (RSS)*, 2014.

[20] Xuehan Xiong, Daniel Munoz, J Andrew Bagnell, and Martial Hebert. 3-d scene analysis via sequenced predictions over points and regions. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2609–2616. IEEE, 2011.