Improving Spatial Support for Objects via Multiple Segmentations

Tomasz Malisiewicz and Alexei A. Efros
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

Sliding window scanning is the dominant paradigm in object recognition research today. But while much success has been reported in detecting several rectangular-shaped object classes (i.e. faces, cars, pedestrians), results have been much less impressive for more general types of objects. Several researchers have advocated the use of image segmentation as a way to get a better spatial support for objects. In this paper, our aim is to address this issue by studying the following two questions: 1) how important is good spatial support for recognition? 2) can segmentation provide better spatial support for objects? To answer the first, we compare recognition performance using ground-truth segmentation vs. bounding boxes. To answer the second, we use the multiple segmentation approach to evaluate how close can real segments approach the ground-truth for real objects, and at what cost. Our results demonstrate the importance of finding the right spatial support for objects, and the feasibility of doing so without excessive computational burden.

1 Introduction

In the early days of computer vision, image segmentation fit neatly into the well-defined object recognition pipeline. First, in the image processing stage, low-level features (edgelets, corners, junctions) are detected. Next, in the segmentation stage, these features are used to partition the image into regions. Finally, in the recognition stage, the regions are labeled with object identities (and possibly even 3D object models!). Alas, it was soon evident that hopes of computer vision being solved in such a simple and elegant way were overly optimistic. Image segmentation in particular turned out to be a major disappointment in this regard – none of the many segmentation algorithms were able to partition a image into anything corresponding to objects. In fact, it was argued (rightly) that low/mid-level segmentation can't possibly be expected to know where one object ends and another one begins since, without object recognition, it doesn't know what objects are!

Meanwhile, the last decade saw an explosion of work in object recognition, most of it without any use of segmentation. Breakthrough results have been achieved in face detection [12, 18] using an exhaustive sliding window approach to predict the presence of a face at every location and scale in the image. A number of researchers have looked at using various visual features directly for recognition, either with a texture-like "bag-of-words" model, or using some spatial relationships (see [9] for an overview). These approaches have shown surprisingly good performance on a number of tasks, including over 65% recognition on the Caltech dataset of 101 object classes. Does this mean that, segmentation, even if it was more accurate, has nothing to contribute to object recognition?

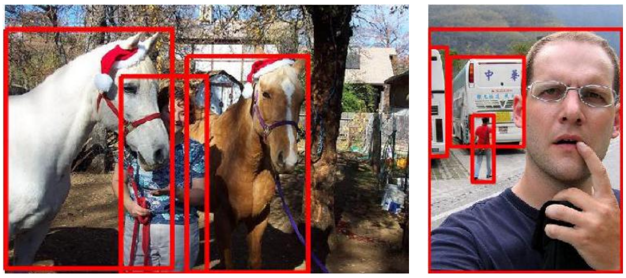


Figure 1: Methods representing objects by bounding boxes often find that up to half of the pixels don't belong to the object of interest (examples from PASCAL Challenge dataset [8]).

1.1 A case for better spatial support

When one considers carefully the recent successes in object recognition, a more nuanced picture emerges. It appears that the current methods are extremely good at a few selected recognition tasks, but quite miserable at most others. For instance, classic rectangular sliding window approaches are known for outstanding results on faces [12, 18], pedestrians [3], and front/side views of cars – all rectangular-shaped objects, but not much else. It seems likely that in cases when the bounding box doesn't cover an object well (e.g. Figure 1), window-based approaches have trouble distinguishing foreground from background.

On the other hand, feature-based methods demonstrate remarkable performance recognizing up to a hundred complicated object categories (grand pianos, anchors, umbrellas, etc) in the Caltech-101 dataset. However, with a single object per image (large and nicely centered), and relatively correlated backgrounds (airplanes in the sky, people in offices, etc) the problem is really more of image classification than object detection. Therefore, it has so far been a challenge to extend detectors trained on Caltech dataset to perform well on novel, cluttered data.

The 2006 PASCAL Visual Object Classes Challenge [8] offers a more realistic dataset with a few large-scale objects per image, although with only 10 object classes. The challenge consists of two tasks: image classification (does this image contain one or more objects of a given class?) and object detection (find and localize all instances of an object class). Tellingly, the contest results on the classification task were overwhelmingly good, especially using the bag-of-words techniques, whereas the performance on the detection task was quite poor. This disparity suggests that the current methods are having a hard time grouping image evidence into coherent objects. Therefore, it is unlikely they will be successful in a setting where a large number of object classes (like in Caltech-101) is presented in realistic images with multiple objects per image (like in PASCAL).

1.2 The Return of Segmentation?

It seems apparent that what is missing from the above recognition efforts is a way to define better spatial support for objects in the image – exactly what the mythical "segmentation stage" is supposed to supply. If we did have perfect spatial support for each object, this would presumably make the recognition task easier in a number of ways. First, by including only the features *on the object* into whatever statistical learning method is being used should greatly reduce the amount of noise in the data, since the relevant features are not being contaminated by the irrelevant and conflicting ones. In fact, it is possible that the features themselves could be made much simpler, since less burden is being placed on them. Additionally, the shape of the object boundary can also be utilized for recognition. Finally, for general image parsing, the information about object-to-object boundaries is extremely useful not just for resolving object ambiguity via context, but for better modeling of the structure of the entire scene.

But how does one get good spatial support? Researchers have long realized that the original segment-then-recognize paradigm is flawed and that one must consider the two in tandem. As early as 1970, several papers developed image interpretation frameworks that combined bottom-up segmentation with top-down semantic knowledge [15, 7]. Recent efforts in that direction include work by Tu et al. [16], Yu and Shi [20], Bornstein et al. [1], and Shotton et al. [14, 19] among others.

Another direction, taken by Hoiem et al. [6] and Russell et al. [11], is to sample *multiple* segmentations from an image, treating them as hypotheses for object support rather than a single partitioning of the image. The motivation is that, while none of the segmentations are likely to partition the image correctly, *some* segments in *some* of the segmentations appear to provide good spatial support for objects. In this setting, the goal of recognition is no longer finding and localizing objects, but merely scoring the various segments by how likely they are to contain a particular object. The multiple segmentation approach is appealing because of its simplicity and modularity – segmentation and recognition are working in tandem, and yet don't need to know anything about each other, so almost any algorithm could be used for each. However, the approach has yet to be thoroughly evaluated, so it is not clear how much benefit it would provide in the context of finding objects. One of the contributions of this paper is providing just such an evaluation.

1.3 Our Aims

In this paper we will address some of the issues raised above by investigating the potential benefits of using segmentation to find better spatial support for objects. The aim of the paper is to answer the following two questions concerning the role of segmentation in recognition: 1) *Does spatial support matter?* That is, do we even need to segment out objects or are bounding boxes good enough for recognition? 2) *Can segmentation provide better spatial support for objects?* That is, even if spatial support is important, can segmentation deliver it?

1.4 Our Dataset

In this paper, we utilize the Microsoft Research Cambridge (MSRC) dataset [14] of 23[1] object classes. To our knowledge, this is the only object recognition dataset with dense labeling (almost every pixel in each image is labeled) and a large number of object categories. Therefore, this is a good dataset for evaluating segmentation algorithms as well as studying multi-class recognition techniques. To make it more suitable for evaluating segmentation performance, we also manually refined many object boundaries and eliminated "void" regions between objects. On the down side, the small size of the dataset (only 591 images) means that some categories contain only a handful of examples making training object detectors extremely difficult.

[1] Like in previous studies, the horse and mountain categories were removed for recognition purposes due to the small number of instances in each category

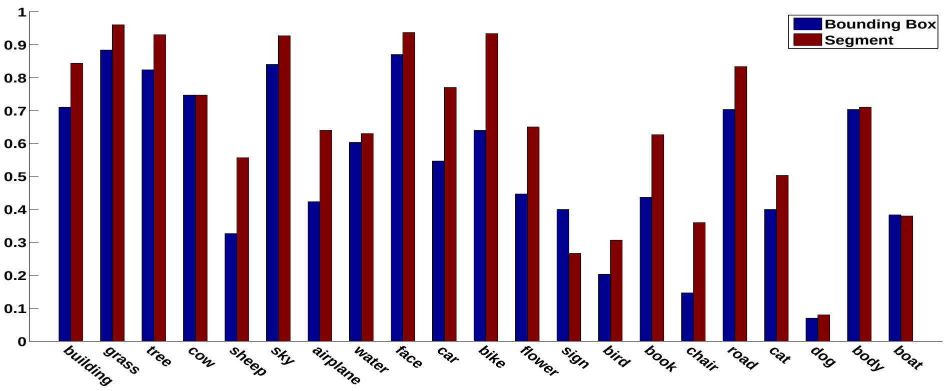


Figure 2: Is good spatial support really worth it? Here, we compare the recall for 21 object classes in the MSRC dataset. For each class, the left bar represents the performance of using ground-truth segments and the right bar is for bounding boxes. In most cases, using pixel-wise segmentations masks gives us a reasonable improvement in classification accuracy.

2 Does spatial support matter?

The first question that must be addressed is whether finding a good spatial support for objects is even a worthwhile goal. After all, there have been suggestions that a bounding box may be able to provide some degree of context and may actually be beneficial [21]. To evaluate this question, we designed a simple experiment on ground-truth segmented objects using a standard object recognition technique. For each object in the dataset, we estimate its class label given a) only the pixels inside the object's ground-truth support region, and b) all pixels in the object's tight bounding box. Because the MSRC dataset is so small, we have chosen the Boosted Decision Tree classifier approach of Hoiem et al. [6] as it appears to perform well under limited amounts of training data. We use exactly the same features as [6] (which measure texture, shape, location, and geometry) extracted from either the ground-truth segment or the tight bounding box of each object.

The results of the experiment, on a per-class basis, can be seen in Figure 2. The first thing to notice is that for 18 out of 21 object categories, the use of correct spatial support improves recognition results. Objects that are poorly approximated by rectangles, such as *sheep*, *bike*, and *airplane*, see the largest improvement (over 50%). Interestingly, recognition on *cars*, which have traditionally been modelled as rectangular, also improves substantially, probably due to the 45 degree views. Categories that don't show improvement with better spatial support are usually doing very well already, e.g. *cows, body*. One exception is *dogs* which is an extremely difficult object class, for which the current methods don't seem to be powerful enough, regardless of the spatial support. Overall, the recognition performance using ground-truth segments is 15% better than using the bounding boxes, increasing from .665 for bounding boxes to .765 for segments. Clearly, the issue of correct spatial support is very important for recognition and should not be ignored. The main question now is, can we actually obtain good spatial support in practice?

3 Can segmentation provide better spatial support?

We have demonstrated that using perfect spatial support can improve recognition, but how can we hope to obtain a good level of spatial support in a purely bottom-up fashion? While there exist many general purpose segmentation engines, studies have shown that none are particularly good at segmenting out individual objects [17, 5]. Instead, we will follow [6, 11] in using multiple different segmentations of the same image, in a way sampling the space of segmentations. In order to put various segmentation schemes on an equal footing, we simply treat the output of any region-generating process as returning a "soup of segments" $\Omega(I)$ for a particular image I.

$$\Omega(I) = (N, \{S_i\}_{i=1}^{N}) \tag{1}$$

When evaluating a segment S's degree of spatial support with respect to a ground truth region G, we compute a normalized overlap score $OS(S,G) \in [0,1]$ (we refer to this overlap score as the spatial support score). When evaluating a soup of segments with respect to G, we report the Best Spatial Support (BSS) score. The BSS is the maximum overlap score in the soup of segments and measures how well the *best* segment covers a ground-truth region.

$$OS(S,G) = \frac{|S \cap G|}{|S \cup G|} \tag{2}$$

For any ground truth region G and soup of segments $\Omega(I)$, we characterize the soup of segments by 2 numbers – the BSS score as well as the number of segments in the soup N. In theory, we want to obtain a small soup (low N) with very high spatial support scores. The performance of a segmentation algorithm across an entire dataset is obtained by averaging the BSS score across each ground truth region in the dataset. Interestingly, under this formulation any sliding window approach can also be seen as generating a soup of segments – namely overlapping rectangular regions at various scales (and possibly orientations).

3.1 The Segmentation Algorithms

In order to have a broad evaluation of bottom-up segmentation, we choose three of the most popular segmentation algorithms, Normalized Cuts (NCuts) [13], the Felzenszwalb and Huttenlocher (FH) algorithm [4] and Mean-Shift [2], to generate our "soup of segments". Following [11], we generate multiple segmentations by varying the parameters of each algorithm. For Normalized Cuts[2], we generate a total of 33 different segmentations per image by varying the number of segments $k = 3,6,9,12,15,18,21,24,27,30,50$ and the image scale to be 100%, 50%, 37% of the original image size. For the FH algorithm, we get 24 segmentations by letting $\sigma = .5,1,1.5,2$, $k = 200,500,1000$, and $min\_range = 50,100$. For the Mean-Shift segmentation, we get 33 segmentations, by fixing $min\_area = 500$ and varying $spatial\_band = 5,7,9$ and $range\_band = 1,3,5,7,9,11,13,15,17,19,21$. The number of segmentations for each algorithm was chosen such that size of the multiple segmentation soup was roughly the same.

However, even with multiple segmentations, it's not always possible to get an object as a single segment. Using the intuition that complicated objects often get over-segmented, we will also consider a larger collection of segments by merging up to 3 segments. To get contiguous segments, we only allow segments that are adjacent and come from the same

[2] The version of NCuts we use utilizes the intervening contour cue on top of the Berkeley Pb operator.

segmentation to be merged. Of course, the improvement in the spatial support is done at the cost of considerably increasing the size of the segment soup.

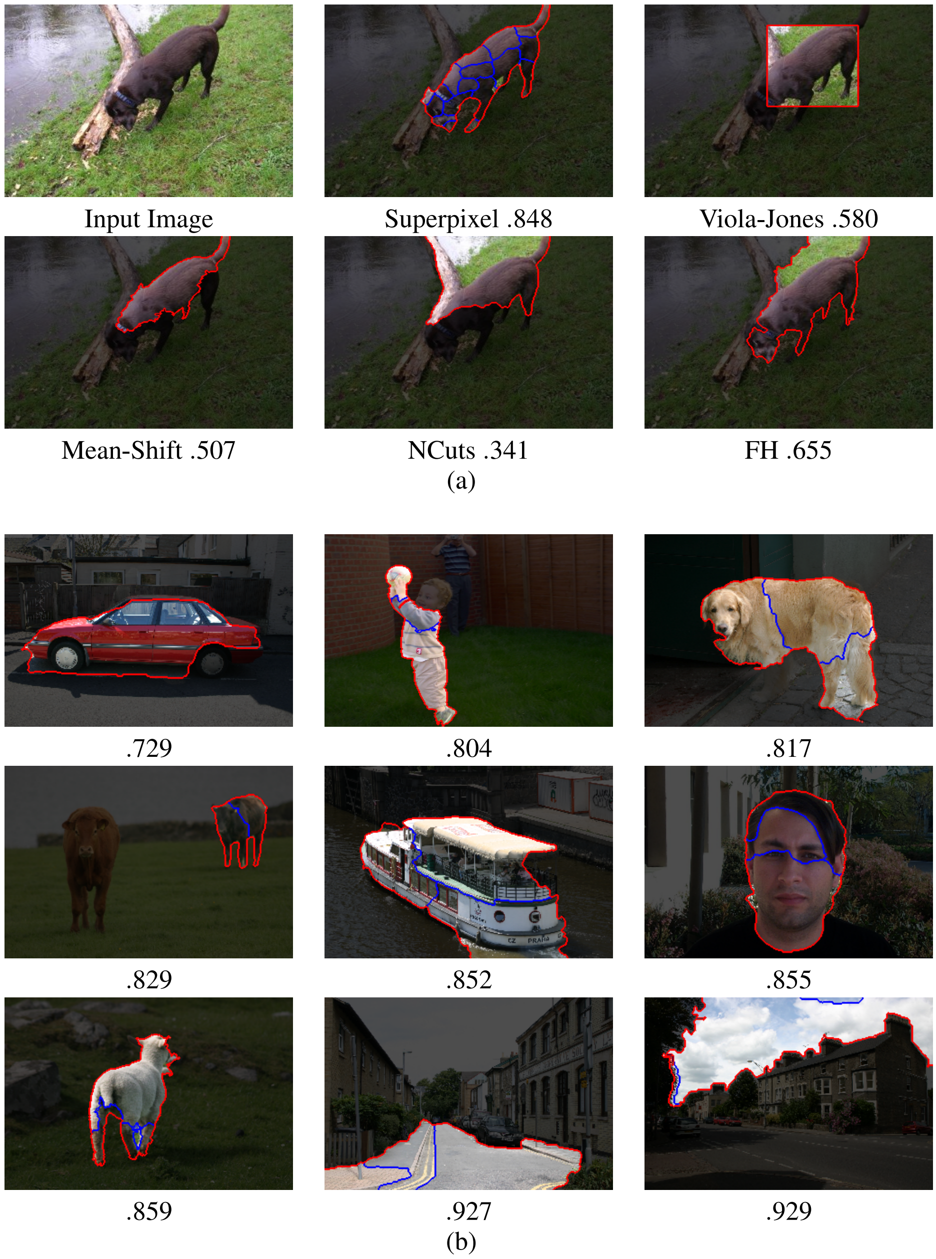
As a baseline, we also consider the soup of segments generated by Viola-Jones [18] style sliding window approach, measuring the best spatial support provided by a square window. Following [18], we choose the smallest window to be 24×24 pixels, and scan each image at 10 scales at $1.25\times$ magnification. The best segments from each algorithm for one example object are shown in Figure 3.

3.2 Experiments

First, we measure the benefits of using multiple segmentations over a single segmentation for the 23 object classes in the MSRC dataset. For each segmentation algorithm, we rank each of the multiple segmentations by their mean BSS score and find the best single segmentation. We then compare the segment soups generated by the best single segmentation, multiple segmentations, multiple segmentations with 1 merge, and multiple segmentations with 2 merges. Each algorithm is evaluated on a per-category basis and compared to Viola-Jones sliding windows (see Figure 4 for results using Mean-Shift). We find that for all algorithms considered, multiple segmentations drastically outperform the best single segmentation. Adding a single merge to the soup of segments also significantly improves the BSS scores, and we notice diminishing returns when adding more merges. We also create a superpixel-based upper bound (the SP Limit) on how well we expect a bottom-up segmentation algorithm to cover objects of a particular class. For the superpixel limit, we create superpixels [10] by over-segmenting each image into approximately 200 regions using Normalized Cuts and finding the optimal spatial support for each ground-truth object instance with respect to the superpixel map (example shown in Figure 3). The SP Limit is just the average performance of superpixels. To no surprise, the multiple segmentation approach is able to provide very good spatial support for object categories whose instances have a relatively homogeneous appearance such as grass, cow, and road while there is still plenty of room left for improving spatial support for complicated objects such as airplanes and boats.

This type of analysis suggests that merging adjacent segments always helps since we only consider the *best* spatial support score. However, in order give a fair evaluation we must also characterize each segment soup by its size. Thus, we also compare the mean BSS of each algorithm versus segment soup size, as seen in Figure 5 (here we use log segments, since the size of the soup quickly gets out of hand). For each of the three algorithms, we compute the mean BSS and the mean soup size for the single best segmentation, multiple segmentations, multiple segmentations with 1 merge, and multiple segmentations with 2 merges. While both FH and Mean-Shift have similar average performance, they both significantly outperform NCuts. To complement the superpixel upper bound, we also determine an upper-bound for the overlap score if using tight bounding boxes (the BB Limit). This is computed by finding the rectangular window with the best overlap score for each object instance in the MSRC dataset and averaging those overlap scores. (This is merely a limit, since in practice it is intractable to slide rectangular regions of *all* aspect ratios and *all* scales across an image.)

We also consider what happens when we concatenate the best single segmentation from each algorithm, the multiple segmentations from each algorithm, and so on. Both FH and Mean-Shift were only able to approach the BB Limit by considering the largest soup of segments (created from multiple segmentations with 2 merges); however, by concatenating the three different algorithms we were able to surpass the BB Limit with a much smaller number of segments. This suggests that the different segmentation algorithms are



Input Image

Superpixel .848

Viola-Jones .580

Mean-Shift .507

NCuts .341

FH .655

(a)

.729

.804

.817

.829

.852

.855

.859

.927

.929

(b)

Figure 3: (a) Best overlap using a single segmentation. Under each image is the segmentation algorithm from which the segment came from as well as the corresponding overlap score. Segment boundaries are represented in red and boundaries of merged segments are represented in blue. The first row of (a) shows the original input image, a superpixel approximation to the ground truth region, as well as the square with best spatial support (Viola-Jones). (b) Nine example results when using a soup of approximately 30k segments. For each object we display the segment with the best spatial support. Here we consider the segment soup created by concatenating all three segmentation algorithms and merging up to 2 segments.

complementary, with each algorithm providing better spatial support for different object categories.

We also note that when using Viola-Jones style window sliding, we generate on the average an order of magnitude more regions than the largest segment soup (created from the combination of the 3 algorithms and up to 2 merges). In addition, the performance of such a square-window based approach is not only far from the BB Limit, but significantly inferior to the multiple segmentation approach. Quantitatively, using Viola-Jones approach we obtain a mean BSS score of .426 while combining the output of the three segmentation algorithms with 2 merges gives a mean BSS of .855 – which is a 100% improvement in spatial support over Viola-Jones window sliding while considering an order of magnitude less segments.

4 Discussion

In this paper, our central goal was to carefully examine the issues involved in obtaining good spatial support for objects. With segmentation (and multiple segmentation approaches in particular) becoming popular in object recognition, we felt it was high time to do a quantitative evaluation of the benefits and the trade-offs compared to traditional sliding window methods. The results of this evaluation can be summarized in terms of the following "take-home" lessons:

Correct spatial support is important for recognition: We confirm that knowing the right spatial support leads to substantially better recognition performance for a large number of object categories, especially those that are not well approximated by a rectangle. This should give pause to researchers who feel that recognition can be solved by training Viola-Jones detectors for all the world's objects.

Multiple segmentations are better than one: We empirically confirm the intuition of [6, 11] that multiple segmentations (even naively produced) substantially improve spatial support estimation compared to a single segmentation.

Mean-Shift is better than FH or NCuts, but together they do best: On average, Mean-Shift segmentation appeared to outperform FH and NCuts in finding good spatial support for objects. However, for some object categories, the other algorithms did a better job, suggesting that different segmentation strategies are beneficial for different object types. As a result, combining the "segment soups" from all three methods together produced by far the best performance.

Segment merging can benefit any segmentation: Our results show that increasing the segmentation soup by merging 2 or 3 adjacent segments together improves the spatial support, regardless of the segmentation algorithm. This is because objects may contain parts that are very different photometrically (skin and hair on a face) and would never make a coherent segment using bottom-up strategies. The merging appears to be an effective way to address this issue without doing a full exhaustive search.

"Segment soup" is large, but not catastrophically large: The size of the segment soup that is required to obtain extremely good spatial support can be quite large (around 10,000 segments). However, this is still an order of magnitude less than the number of sliding windows that a Viola-Jones-style approach must examine. Moreover, it appears that using a number of different segmentation strategies together, we can get reasonable performance with as little as 100 segments per image!

In conclusion, this work takes the first steps towards understanding the importance of providing good spatial support for recognition algorithms, as well as offering the practitioner a set of concrete strategies for using existing segmentation algorithms to get the best object support they can.

References

[1] Eran Borenstein, Eitan Sharon, and Shimon Ullman. Combining top-down and bottom-up segmentation. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, page 46, Washington, DC, USA, 2004.

[2] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Patt. Anal. Mach. Intell.*, 24(5):603–619, 2002.

[3] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 2005.

[4] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *Int. Journal of Computer Vision*, 59(2):167–181, 2004.

[5] Feng Ge, Song Wang, and Tiecheng Liu. Image-segmentation evaluation from the perspective of salient object extraction. In *CVPR*, 2006.

[6] D. Hoiem, A. Efros, and M. Hebert. Geometric context from a single image. In *Proc. Int. Conf. Comp. Vision*, 2005.

[7] Y. Ohta, T. Kanade, and T. Sakai. An analysis system for scenes containing objects with substructures. In *IJCPR*, pages 752–754, 1978.

[8] The pascal object recognition database collection. Website, 2006. http://www.pascal-network.org/challenges/VOC/.

[9] J. Ponce, M. Hebert, C. Schmid, and A. Zisserman. *Toward Category-Level Object Recognition*. Springer-Verlag Lecture Notes in Computer Science, 2006. In press.

[10] Xiaofeng Ren and Jitendra Malik. Learning a classification model for segmentation. In *International Conference on Computer Vision*, pages 10–17, 2003.

[11] Bryan C. Russell, Alexei A. Efros, Josef Sivic, William T. Freeman, and Andrew Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *Proc. CVPR*, 2006.

[12] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *Proc. IEEE Conf. Comp. Vision Patt. Recog.*, Hilton Head, SC, 2000.

[13] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. PAMI*, 22(8):888–905, August 2000.

[14] Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *ECCV*, 2006.

[15] J.M. Tenenbaum and H.G. Barrow. Experiments in interpretation guided segmentation. *Artificial Intelligence*, 8(3):241–274, June 1977.

[16] Z. Tu, X. Chen, A. Yuille, and S.-C. Zhu. Image parsing: unifying segmentation, detection, and recognition. In *Proc. Int. Conf. Comp. Vision*, volume 1, pages 18–25, 2003.

[17] Ranjith Unnikrishnan, Caroline Pantofaru, and Martial Hebert. A measure for objective evaluation of image segmentation algorithms. In *Proceedings of the 2005 CVPR Workshop on Empirical Evaluation Methods in Computer Vision*, June 2005.

[18] P. Viola and M.J. Jones. Robust real-time face detection. *Int. J. of Comp. Vision*, 57(2):137–154, 2004.

[19] John Winn and Jamie Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. In *CVPR*, pages 37–44, 2006.

[20] Stella X. Yu and Jianbo Shi. Object-specific figure-ground segregation. In *CVPR*, 2003.

[21] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. of Comp. Vision*, 2007. in press.

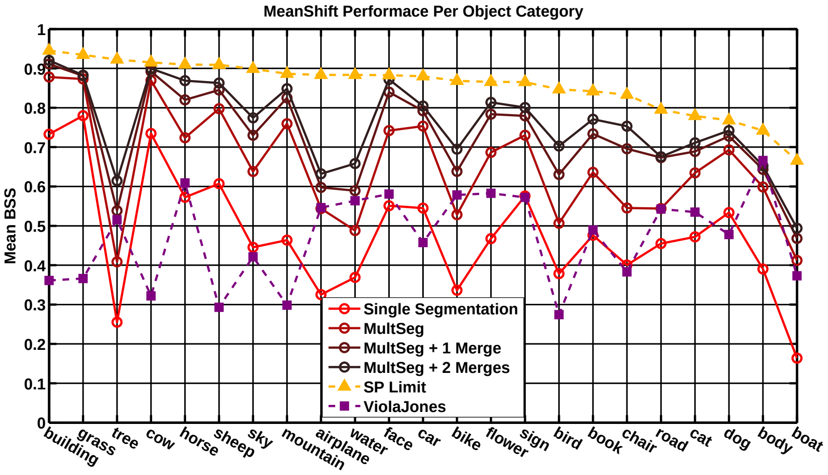


Figure 4: Performance across object category for the Mean-Shift algorithm. The performance of the segment soup created by the single best segmentation, multiple segmentations, multiple segmentations with 1 merge, and multiple segmentations with 2 merges is compared to a Viola-Jones window sliding approach as well as a superpixel-based uppper-bound on segmentation performance.

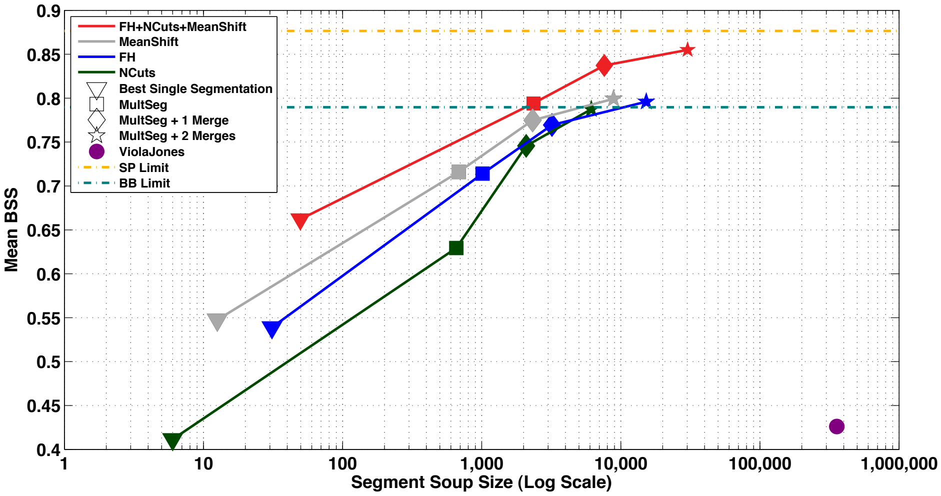


Figure 5: Mean BSS versus Segment Soup Size for different segment soups generated from each of the 3 segmentation algorithms compared to Viola-Jones window sliding. A log scale is used along the x-axis since segment soups get large very fast. A combination of the 3 algorithms is also evaluated as well as a superpixel-based upper bound and a bounding box upper bound on performance.