Shape-based Recognition Of Wiry Objects

Owen Carmichael and Martial Hebert
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA, 15213

Abstract

We present an approach to the recognition of complex-shaped objects in cluttered environments based on edge cues. We first use example images of the desired object in typical backgrounds to train a classifier cascade which determines whether edge pixels in an image belong to an instance of the object or the clutter. Presented with a novel image, we use the cascade to discard clutter edge pixels. The features used for this classification are localized, sparse edge density operations. Experiments validate the effectiveness of the technique for recognition of complex objects in cluttered indoor scenes under arbitrary out-of-image-plane rotation. [1]

1. Introduction

This paper addresses the recognition of objects which consist mainly of elongated, thin, stick-like components connected together into complex structures; we will refer to these as *wiry* objects. Man-made objects in this category are common; for example the chair, cart, tables, and lamps in Figure 1 contain a significant amount of wiry structure. We focus on the problem of recognizing specific instances of objects, for example the specific chair in Figure 1, rather than the more general problem of detecting chairs of all different shapes and sizes.

In recent years, several authors have made significant progress toward the recognition of certain objects, such as faces, buildings, and cars; examples include [21] and [24]. Typically, these approaches formalize the recognition problem as one of modeling the appearance of rectangular image patches circumscribing the object or its parts, across changes in pose[17], lighting[4], or other conditions. This reduces the recognition problem to examining a rectangular image template and using its appearance to determine whether or not it is the image of some section of the target object.

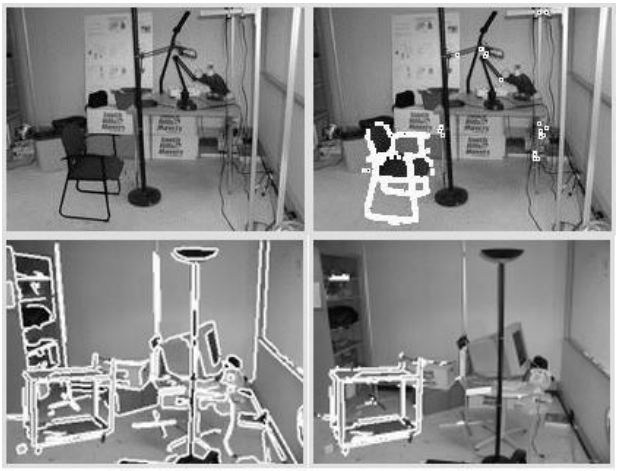


Figure 1. We address the recognition of objects like chairs(top) and carts(bottom) based on edge cues. Top Row: Example input image (left) and edge filtering result (right). Bottom Row: Example image with detected edges overlaid (left) and edge filtering result (right). See Section 1 for an overview and Section 3 for details on experiments.

Since appearance-based techniques formulate the problem in terms of rectangular image windows, they tend to work well when applied to target objects (or object parts) whose projection into the image fills a rectangular region. The objects we consider produce images that are poorly approximated by rectangles; for objects such as the chair, table, and lamps in Figure 1, a bounding box around the object or any substantial section of it will contain a high percentage of pixels which map to the background or other objects. Most successful appearance-based approaches can handle the variation in template appearance induced by a small number of background pixels in the patch, but when most of the template consists of clutter its appearance can vary widely due to a modification of the background or object pose. This in turn can make it difficult to model the appearance of the object or object part based on the entire template.

Furthermore, since most appearance-based recognition

[1]This research was supported by a grant from Honda Corporation.

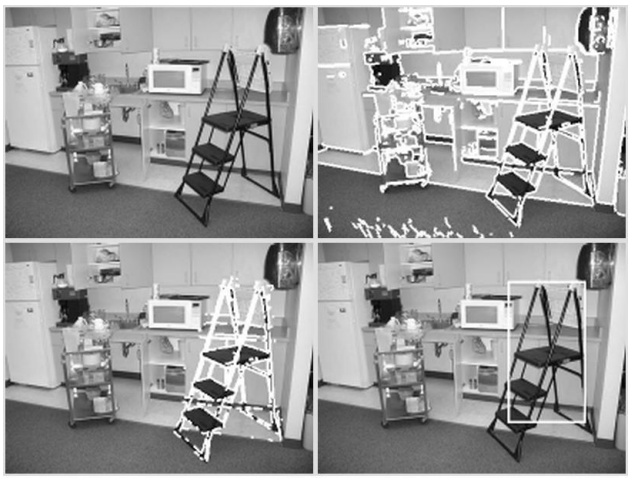


Figure 2. Input image of the ladder in the kitchen environment (top left), edges detected in the scene (top right), result of the edge filtering technique described in Section 2(bottom left), results of edge grouping described in Section 3.3(bottom right).

techniques rely on greylevel or color texture patterns as cues, they may face exceptional difficulty with wiry objects. The ladder (Figure 2), for example, has very little in the way of appreciable visual texture. Therefore, while appearance-based approaches to recognition are effective for certain objects, we suggest that for wiry objects, shape can be a more illuminating cue. It may ultimately be appropriate to recognize these objects from a combination of cues; here, however, we concentrate on the use of shape features for recognition.

In an earlier paper [8], we described a discriminative technique for using shape cues to filter clutter edge pixels from images containing a wiry target object. First, we train a cascade of classifiers based on example images in which edge pixels have been labeled as belonging to the target object or the background. Given a novel image, we apply our classifier cascade to each edge pixel in the image to determine which of those edge points project onto the target object. The classifiers in the cascade calculate simple, localized edge density features called "edge probes" to make their decisions. Experiments showed that in composite images of target objects in cluttered scenes (Figure 1), it is possible to use the classifier cascade to discard most edge pixels on the clutter, resulting in a set of edge points which isolate the target object from the background.

However, this preliminary study raised a number of questions regarding the applicability of the technique to the recognition of arbitrary wiry objects in real-world scenes. First, since the experiments used composite images of the object superimposed on independent images of a background, it is unclear whether the technique would be effective on real images of objects in their environment. Second, while filtering clutter pixels is certainly a useful step in the recognition process, it does not perform "recognition" of the overall object per se since the output of filtering is a set of individual edge pixels. In particular, it is not clear how our scheme for clutter edge pixel removal relates to a higher-level determination of the 3D pose or 2D image location of the overall object. Third, since our algorithm for filtering edge pixels is tuned to specific background images, we would like to know how well it performs when presented with a test image whose background, lighting, or other environmental characteristics vary from those of the training images. Fourth, our experiments applied the technique to views of the target object at arbitrary out-of-image-plane rotation, but with only small variations in scale and in-plane rotation; for real applications we wish to recognize the object over a wider range of poses.

This paper addresses the first three of these points empirically. Specifically, in Section 3 we supplement our initial results on recognition of a chair and cart with a quantitative set of experiments on the filtering of clutter edge pixels from images of a common wiry object (a ladder) in a variety of real indoor environments. Furthermore, in Section 3.3 we show examples of how a simple post-processing step can group individual edge pixels into object-level descriptions of the contents of the image. Also, in Section 3.2 we present an experiment in which a classifier cascade is applied to test images taken in an environment whose characteristics vary significantly from those of training images. Finally, in Sections 3.5 and 3.4 we describe complexity issues and show a sensitivity analysis of our approach with respect to a user-set parameter of our algorithm. Related approaches to edge-based object recognition are discussed in 1.1, our algorithm for recognition is reviewed in Section 2, and closing comments are in Section 4.

1.1. Related Work

While appearance-based approaches to object recognition are dominant in the current computer vision literature, techniques based on analysis of edges have a long history, and were studied extensively in the 1980s. Algorithms developed during this period, such as interpretation trees[11], could recognize occluded, 2D, non-convex shapes from noisy, binary edge images. Unfortunately, many of these approaches rely on a tree search through the space of all possible correspondences between edge features in the image and edge features on an object model, and thus can become computationally expensive if the image or the model contains a large number of features. Indexing techniques such as geometric hashing[16] bypass the tree search by having each k-tuple of image features cast votes for the identities and/or poses of objects in the image; however if the image contains significant noise[12] or clutter, the votes cast by sets of clutter features will overwhelm the votes cast by

the object, making it difficult to draw any conclusions about what objects are there.

Early indexing techniques computed very simple descriptors from sets of 3, 4, or 5 image features; more recently, several authors have proposed the use of more rich descriptions of local image shape in conjunction with indexing for edge-based recognition. Conspicuous groupings of edge segments [3], edges in a rectangular image patch [22], and histograms of local edge distributions [5] are all examples of more advanced local edge features which have been applied to shape-based recognition problems. In each of these approaches, various parameters controlling the characteristics of the edge descriptor (histogram bin sizes, grouping thresholds, and so on) must be provided by the user; the goal of our approach is to use training examples of the object in typical backgrounds to estimate feature parameters in such a way that the edge features computed on a test image effectively discriminate the target object from the background.

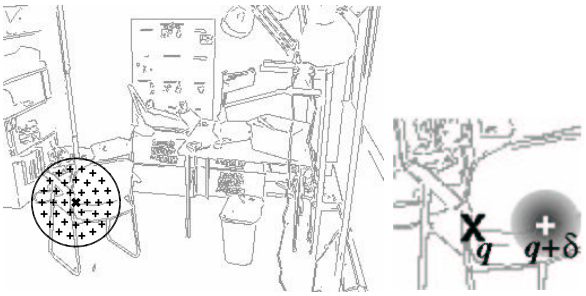
Belongie *et al*[5] calculate a histogram, or "shape context," at each edge point in an image; each bin in the histogram counts the number of edge pixels in a neighborhood near the point. Our approach is similar in that both use the distribution of edges in a local section of the image surrounding a point (which we call the *aperture* around the point) as the fundamental feature for recognition. However, the shape context uses a "dense" set of edge features for recognition; in other words, the bins in the histogram exhaustively cover the aperture. Our approach only computes edge features at isolated image locations deemed likely to discriminate the edge point in question as object or clutter, so the features we use are spatially "sparse." Section 3.5 contains an experiment which helps to quantify how sparse our shape features are in practice.

2. Approach

Given a novel image of the target object in a cluttered background, we extract edges and wish to apply a filter to the image to determine which edge pixels belong to an instance of the target object and which edge pixels belong to the clutter. To optimize this filter, we assume we possess a set of example images containing the object in typical scenes; edges in the example images have been marked as belonging to the object or to the background. This section explains our approach for training and testing in detail.

Our edge filter computes localized edge features at image locations near the edge pixel under consideration. An *edge probe* at *probe center* p over a list of edge pixels L is defined as

$$EP(p, L) = \sum_{t \in L} \exp\left(-\frac{\|p - t\|^2}{\sigma^2}\right)$$



(a) (b)

Figure 3. 3(a): Edge probes are evaluated in a circular region surrounding a query edge point. The query edge point is marked "X," and edge probes are evaluated at locations marked "+." 3(b) Each edge probe measures edge density in some image neighborhood. Here an edge probe is evaluated at shifted probe center $q + \delta$ for a query edge point q.

where t and p are 2-vectors of $[x, y]$ image coordinates. An edge probe can be thought of as a Gaussian receptive field with variance σ^2, centered at point p in an edge image whose edge pixels are contained in the list L. Edge probes measure the density of edge pixels in some neighborhood in the image; in this sense, each edge probe is analogous to a bin in a shape context histogram[5]. The variance σ^2 is a user-set parameter; however an experiment reported in Section 3.4 suggests that our edge filtering results are not highly sensitive to its setting.

Consider a set of *relative probe centers* $\Delta = \{\delta_1, \delta_2, \cdots, \delta_k\}$, $\delta_i = [x_{\delta_i}, y_{\delta_i}]$, laid out over a 2D grid centered at the origin. To classify a novel edge pixel q, we shift the relative probe centers so that they surround q and compute a subset of edge probes $EP(q, \Delta, L) = \{EP(q + \delta_1, L), \cdots EP(q + \delta_k, L)\}$ at *shifted probe centers* $\{q + \delta_1, \cdots, q + \delta_k\}$.An illustration is shown in Figure 3.

Given a fixed σ, we space the relative probe centers evenly over a circular aperture as in Figure 3(a) so that each pixel in the aperture contributes to one or more edge probes. But how large should the aperture be?[2] We want the shifted probe centers to cover a large enough neighborhood surrounding q that the edge probes will contain sufficient information to discriminate object pixels from background pixels. At the same time, however, if the aperture is too large (covering the entire image, for example), an unfeasible amount of computation will be required at training time to evaluate edge probes that might not be crucial for classification. Worse, if the aperture is so large that most of the edge probes at shifted probe centers are totally irrelevant to the category of the query edge point, error-prone classifiers could be trained[14][1]. Thus, we are presented with "the aperture problem" which appears in many computer vision

[2]We emphasize that σ determines the spatial support of a *single* edge feature while the aperture size controls the size of the neighborhood over which *all* edge features are computed for a given query point.

Figure 4. Results of edge filtering and edge grouping for cascades trained on various environments. For each pair of images, the left-hand image shows results of the edge filtering operation described in Section 2, and the right-hand image shows results of edge grouping as described in 3.3. Top row: The lab (left) and cubicle (right). Middle row: The classroom (left) and conference room (right). Bottom row: the warehouse (left) and living room (right).

problems– when attempting to induce information about a particular location in the image we want to incorporate image data from a large enough surrounding area that the information can be induced, but not so large that we introduce irrelevant data or useless computation.

Consider a set of relative probe centers Δ which cover a circular aperture as in Figure 3(a). Define $A(\Delta)$ to be the radius of the circle. Our approach is to train a series of classifiers $f_1, f_2, \cdots, f_k$ which evaluate edge probes according to sets of relative probe centers $\Delta_1, \Delta_2, \cdots, \Delta_k$ such that $A(\Delta_1) < A(\Delta_2) < \cdots < A(\Delta_k)$. The first classifier in the series, f_1, is trained to classify edge points based on edge probes taken from a small radius surrounding them; f_2 classifies based on edge probes over a slightly larger radius, and so on. Edge points labeled "object" by $f_1$are classified by f_2; points labeled "background" by $f_1$are discarded. Edge points labeled "object" by f_2 are passed to f_3, and so on.

Thus, we solve our aperture problem in phases– we first identify those edge points whose class is discriminable based on very nearby features, then identify points that are made discriminable by features slightly farther away, and continue to do so until the aperture covers the entire object in question.

Besides providing a solution to our aperture problem, the classifier cascade allows fast screening of image locations that are easily discriminable from the object of interest based on information in a small window, leaving the bulk of the computation to more ambiguous sections of the image. Similar cascade strategies have recently achieved significant speedups for template-based approaches to recognition[24][13].

The classifiers in our cascade are decision trees trained using a two-step process of tree generation and pruning, following the reduced-error pruning approach of Quinlan[19]. In this framework, the training data is split into two subsets, which we will refer to as the *tree-growing set* and the *holdout set*. The tree-growing set is recursively partitioned based on the values of features selected by a greedy information-theoretic criterion; the resulting tree has high classification accuracy on the tree-growing set but is prone to overfitting. Subtrees are then pruned from the tree when doing so improves a global accuracy criterion on the holdout set[19][6][7]. Our pruning criterion is shaped by the fact that the classifiers are applied in a cascade. Specifically, consider an edge pixel q which corresponds to a point on the object. If a classifier mistakenly classifies q as clutter (*i.e.* a "false negative"), then the edge point is permanently removed from consideration by further classifiers in the cascade; however, if a clutter edge pixel q is mistakenly classified as belonging to the object (a "false positive"), then the edge point is passed on to later phases in the cascade, which may in turn re-classify it correctly based on edge information in a larger aperture. We therefore optimize a Neyman-Pearson criterion [9] during pruning; specifically, we prune

Environment	# train	# test	mean TP	mean FP	var TP	var FP
Classroom	120	48	0.778	0.102	0.008	0.008
Kitchen	120	43	0.712	0.111	0.038	0.038
Cubicle	120	54	0.718	0.080	0.039	0.039
Conf. Rm	120	63	0.775	0.089	0.018	0.018
Warehouse	120	51	0.755	0.094	0.021	0.021
Living Rm	120	54	0.779	0.064	0.017	0.017
Lab	110	12	0.739	0.210	0.010	0.010

Table 1. Edge pixel filtering results for classifier cascades trained on individual environments. First column: Each row corresponds to edge filtering results for a classifier cascade trained on images taken in the environment indicated in the first column. Second column: Total number of images in the tree-growing and holdout sets together. Third column: Number of independent test images used for evaluation. Fourth and fifth columns: Average true positive (TP) and false positive (FP) rates over all test images. Sixth and seventh columns: Variance in TP and FP rates across all test images.

subtrees whenever doing so improves the false positive rate of the classifier while keeping the false negative rate below a low, fixed threshold θ. Depending on the arrangement of edges in the training images, it is possible that the decision tree trained for a particular cascade may not significantly reduce the number of false positives on the holdout set; in this case we simply skip this cascade phase and train a classifier for the next phase. For each edge pixel q in each image in the tree-growing set, we compute edge probes at all shifted probe centers $\{q + \delta_1, \cdots, q + \delta_k\}$ corresponding to the relative probe centers $\{\delta_1, \cdots, \delta_k\}$ in the smallest aperture Δ_1. Decision tree induction then iteratively splits the edge points into subsets according to the values of edge probes corresponding to selected relative probe centers. Each edge pixel in the holdout set images is then classified by the resulting tree, and subtrees are removed if the pruned tree reduces the number of background edge pixels classified as object edge pixels while keeping the percentage of object edge pixels correctly classified (the "true positive" rate) to 1-θ. Edge pixels from the tree-growing and holdout sets classified as object edge pixels by the pruned tree then pass to the training of the second phase in the classifier cascade: for each of these edge pixels, edge probes are computed at all shifted probe centers corresponding to relative probe centers in Δ_2, and so on.

Given a test image, we apply the trained classifiers to each of its edge pixels in turn. An edge pixel classified as "object" by the first classifier is passed to the second classifier; the second classifier classifies the point again, and so on until the point is labeled as "clutter" or the cascade ends.

3. Experiments

We took 1157 1600-by-1200 images of a ladder in 7 different indoor environments: a classroom, conference room, office, lab, living room, warehouse, and kitchen (Figures 2 and 4). For each image, the camera was approximately 3m from the objects in the scene; the elevation of the camera varied between 1.6m and 1.75m; the set of all images of a particular scene covered about 60 degrees of rotation with respect to the scene objects in the plane parallel to the floor. The camera was moved between each view, and once every five views the ladder was rotated to an arbitrary angle with respect to the ground and the poses and configurations of clutter objects were randomly modified. The depth of the ladder with respect to the camera varied by a total of approximately 20% across all views. Edges were detected in these images using the Vista line finder[18]; edges were hand-labeled as belonging to the ladder, or to the clutter objects.

Figure 5. Left: Example of scanning an edge density filter for the ladder over a raw edge image (left) and an edge image which has been filtered using the technique described in Section 2 (right). See section 3.3 for details.

For each experiment, we selected some number of images for tree-growing and holdout sets, and trained cascades of classifiers to filter out background edge pixels as described above. For the experiments in Sections 3.1 and 3.2, the edge probe variance parameter σ was set to 20 pixels; the decision tree pruning parameter θ was set to 2%, and a cascade phase was skipped if it failed to reduce the false positive rate by 5% or more. Trees were induced with the MLC++ package [15].

The relative probe centers were arranged as a set of concentric rings; specifically, the relative probe centers in the nth ring were positioned in a circle of distance $n * \sigma$ from the origin, with a σ-pixel spacing between adjacent relative probe centers on the circle. The set of relative probe centers Δ_n corresponding to the nth aperture in the cascade is the union of all relative probe centers in rings 1 through n. Note that uniformly tiling the aperture with a set of edge features of equal spatial support is in contrast to techniques which aim for a "foveal" layout of edge features, for example [5].

3.1. Training And Testing On A Single Environment

In our first set of experiments we considered training individual classifier cascades for each environment separately. For each environment, we randomly split the set of all images of the object in that environment into a tree-growing

Environment	# test	mean TP	mean FP	var TP	var FP
Classroom	169	0.661	0.072	0.050	0.050
Conf. Rm	183	0.779	0.110	0.013	0.013

Table 2. Edge pixel filtering results for classifier cascades trained on a different set of environments than the test images. See Table 1 for an explanation of notation.

set of 60 images, a holdout set of 60 images, and a test set containing the remainder of the images. Then the procedure described above was used to train a 20-phase classifier cascade, and run each of the test images through the resulting filter. For each test image, we measured the true positive (TP) and false positive (FP) rates, *i.e.* what percentage of edge pixels on our target object and background were ultimately classified as object edge pixels by the cascade. Results are summarized in Table 1; examples are shown in Figures 2 and 4. Each classifier cascade retained a high percentage of edge points on the object (roughly 70%-80%), while discarding most background edge pixels (roughly 90%). We emphasize that the reported true positive and false positive rates refer to the percentage of object *edge pixels* retained by the edge filter, not the percentage of times the overall object was correctly or incorrectly identified in all test images. However, as explained in Section 3.3 and indicated in Figures 2 and 4, the filtered edge image is an encouraging starting point for recognition processes that operate at the object level, such as 3D pose estimation or localizing the object in the image.

3.2. Distinct Training and Testing Environments

Next we address the address the training of a classifier cascade across a particular set of environments and applying the resulting filter to images of the same object in front of entirely distinct environments. To suggest that the performance of our classifier cascades degrades gracefully according to the deviation between training image characteristics and test image characteristics, we trained a classifier cascade on a set of images of the ladder in five of the environments (kitchen, cubicle, warehouse, living room, and lab), and tested it on images of the other two environments (classroom and conference room). We randomly selected a total of 120 images from the set of all images of the object in the training environments, using 60 of them for the tree-growing set and 60 for the holdout set. A classifier cascade was computed from these training images and applied to all images of the object in the test environments, i.e. environments *not* present in the training data. Edge pixel classification results are summarized in Table 2, using the same notation as Table 1. Comparing the corresponding lines in Tables 1 and 2, it appears that filtering performance decreases slightly in some aspects: the true positive rate on the classroom images drops somewhat when the classroom images

Sigma	mean TP	mean FP	var TP	var FP
15	0.827	0.094	0.009	0.009
20	0.775	0.089	0.018	0.018
25	0.715	0.074	0.017	0.017
30	0.775	0.125	0.014	0.014

Table 3. Edge pixel filtering on the conference room image set for various settings of σ.

are not present in the training data, and the false positive rate for the conference room images increases when the conference room images are absent from training. However, some decrease in performance is to be expected when conditions in training and test images vary significantly; the key point here is that the experiment suggests that edge filtering performance degrades gracefully with these variations.

3.3. Grouping Individual Edge Detections

As indicated in Section 1, the edge filtering procedure does not solve object recognition at an object level; instead it selects individual image pixels likely to project onto the object. To give an example of how filtered edge images may be used as an input to object-level recognition processes, we implemented a simple density filter which scans the filtered edge image with a rectangular template roughly the size of the target object. At each image location, the number of edge pixels falling inside the template is recorded, and image locations with a large number of edge pixels inside the template are noted as likely locations of the target object.

More specifically, we first run all training images through the classifier cascade described above. For each training image, we calculate the bounding box around the remaining edges of the target object. Taking an average over all resulting bounding box sizes gives us our characteristic bounding box for the object; we will assume that our object is well-approximated by a bounding box of this characteristic size in all test images. We then scan test images with this characteristic bounding box, recording the number of edge pixels inside the box at each image location. Boxes with a large amount of edge pixels are determined to be the bounding box around an instance of the target object, and non-maximum suppression is performed to arrive at final, isolated boxes. Examples of images on which this density filter have been applied are shown in Figures 2 and 4.

We emphasize that this density filter is in no way an optimal procedure for recovering an object-level description of the contents of the image; in particular, a variety of alignment algorithms[2][23] would be able to give a more precise correspondence between the test image and a reference image or 3D model. Still, this experiment illustrates that the edge filtering procedure can be a useful preprocessing step to higher-level edge-based recognition processes that may fail in extreme clutter. For example, Figure 5 shows a test

Phase	# features	# examples	time
1	6	316338	0:04
5	92	213282	0:16
10	340	141120	0:44
15	746	111674	1:05
20	1309	107848	1:20

Table 4. Number of relative probe centers (second column), number of training examples (third column), and decision tree induction times (fourth column, hours:minutes) for various cascade phases for the classroom image set.

image for which the density filter fails when applied to the raw edge image, but is able to localize the object once the clutter edges have been removed by the classifier cascade.

3.4. Sensitivity

The edge probe variance parameter σ is a free parameter that critically affects the size of the spatial support region for our basic edge features. Thus, it is natural to wonder how the performance of our approach depends on the choice of σ. To address this issue, we trained a set of 4 classifier cascades on the conference room image set, corresponding to σ values of 15,20,25, and 30 pixels. As in Sections 3.1 and 3.2, the relative probe centers were arranged in concentric rings at distances $\sigma, 2 * \sigma, \cdots$ pixels from the origin. However, in order to increase the aperture size at a similar rate for all values of σ, we added two rings of relative probe centers per cascade phase for $\sigma = 15$. 60 images of the conference room were randomly selected as a tree-growing set, 60 images made up the holdout set,and the remaining 63 images were used for evaluation. Figure 3 summarizes the results, using the same notation as described in Table 1. While the true positive and false positive rates do vary across settings of the parameter, in each case the filter retains a high percentage of edge pixels on the object (roughly 70% to 80%) while removing roughly 90% of all background edge pixels. Exactly how the setting of σ affects true positive and false positive rates is complex and depends on characteristics of the viewing environment, the objects present, the classifiers in the cascade, and the policy for arranging the relative probe centers in the aperture. Nonetheless, these results suggest that the classifier cascade performs well over a range of reasonable values for this parameter.

3.5. Complexity

This section addresses concerns about the time and space complexity of our approach, in terms of both the training and test phases. In particular, we demonstrate that although a potentially large number of edge features is considered by our algorithm, and although we employ numerous classifiers at training and test time, the amount of computation

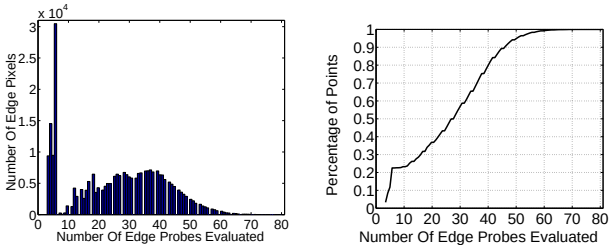


Figure 6. Left: histogram of the number of edge probes evaluated per edge point in 54 test images. Right: cumulative distribution function for this histogram.

required to train the cascade and evaluate a novel image is feasible.

Training Training the cascade of classifiers involves computing edge probes and inducing a decision tree for each phase of the cascade. At a particular cascade phase, an exhaustive set of edge probes is computed over all edge points in the training set; thus, space requirements at training time will be determined by the number of edge pixels in the training set at each cascade phase, along with the number of relative probe centers in the aperture at each phase. In Table 4 we show these numbers for a few phases of the classifier cascade trained on the classroom image set as described in Section 3.1, along with the approximate running times of the decision tree inducer on a 1.67 GHz Athlon for those cascade phases. On one hand, due to our strategy of evenly spacing the relative probe centers in the aperture, the dimensionality of the training data increases as the aperture is grown; on the other hand, since we filter training examples out of the training set at each phase of cascade training, the number of training examples decreases as the aperture is grown. The total time required to train one classifier cascade on a 1.67 GHz machine, including all decision tree induction and edge probe calculation, is approximately one day.

Testing When evaluating a novel image, some number of edge probes are evaluated at each edge point in the image until either the point is filtered out or the last phase of the cascade is reached. Thus, the time complexity of evaluating a novel image will in large part be determined by the number of edge probes required at its edge points. To get a sense of the total number of edge probes computed at a typical image point, we took the classifier cascade trained on the living room image set, described in Section 3.1, and counted the number of edge probes evaluated at each edge pixel in each test image of the living room, over all cascade phases. Figure 6 summarizes these edge probe counts in a histogram and cumulative distribution. Note that roughly 23% of all edge pixels are classified from ten or fewer distinct edge probes, and 95% of all points in the test images

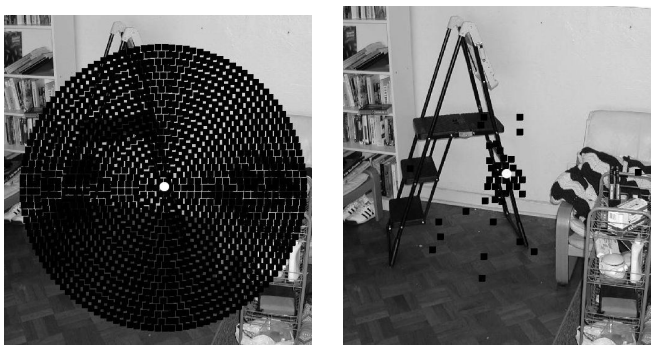


Figure 7. Left: The set of all relative probe centers for the 20th cascade phase, shifted to the point at the white circle, are shown as black dots. Right: In order to classify the point, edge probes are only evaluated at the probe centers shown in black.

require evaluation of 50 or fewer edge probes. This is significant since the total number of relative probe centers in the largest aperture (Table 4, last row) is 1309. Thus, while the training phase selects features from a large set of potential features, relatively few of these features are evaluated at any given image point at run time. As an illustration, Figure 7 shows the set of all shifted probe centers for the 20th cascade phase at a typical edge pixel (left) and the set of 65 shifted probe centers at which edge probes were computed during the classification of the point by all phases of the cascade. Note also that since an edge probe essentially consists of convolution of a portion of the image with a small gaussian kernel, it is fast to compute.

4. Conclusion

In this paper we have presented a discriminative technique for compensating for highly cluttered backgrounds in edge-based object recognition. We substantiated its feasibility and accuracy through detailed tests on a real object in a variety of cluttered scenes. We show that since edge feature extraction is automatically tuned to the object and environments present at training time, we can effectively address the discrimination of object edge points from background edge points while leaving few arbitrary parameters for the user to estimate. Our experiments demonstrate that the technique can be robust to out-of-image-plane rotation of objects in the scene, variations between training and test data, and changes in parameter settings. Also we give an example of how our approach can serve as an effective preprocessing step for object-level recognition processes in cluttered environments.

Future experiments will extend the variability between training and test data to explore the robustness of the technique. Also, we will compare our approach to similar classifier cascades employed in other object recognition work, for example [24].

References

[1] H. Almuallim and T. G. Dietterich. Learning with many irrelevant features. In *Proc. AAAI*, 1991.

[2] R. Basri and D. Jacobs. Projective alignment with regions. *IEEE Trans. PAMI*, 23(5):519–527, May 2001.

[3] J. Beis and D. Lowe. Indexing without invariants in 3d object recognition. *IEEE Trans. PAMI*, 21(10):1000–1015, 1999.

[4] P. Belhumeur and D. Kriegman. What is the set of images of an object under all possible illumination conditions? *IJCV*, 28(3):245–260, 1998.

[5] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. PAMI*, 24(4):509–522, April 2002.

[6] J. P. Bradford, C. Kunz, R. Kohavi, C. Brunk, and C. E. Brodley. Pruning decision trees with misclassification costs. In *Proc. ECML*, pages 131–136, April 1998.

[7] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth International, Belmont, CA, 1984.

[8] O. Carmichael and M. Hebert. Object recognition by a cascade of edge probes. In *Proc. BMVC*, 2002.

[9] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, 2 edition, 2001.

[10] W. Grimson. *Object recognition by computer : the role of geometric constraints*. MIT Press, 1990.

[11] W. Grimson and D. Huttenlocher. On the sensitivity of geometric hashing. In *Proc. ICCV*, pages 334–338, 1990.

[12] D. Karen, M. Osadchy, and C. Gotsman. Antifaces: A novel, fast method for image detection. *IEEE Trans. PAMI*, 23(7):747–781, July 2001.

[13] K. Kira and L. A. Rendell. A practical approach to feature selection. In *Proc. ICML*, pages 249–256, 1992.

[14] R. Kohavi, D. Sommerfield, and J. Dougherty. Data mining using MLC++: A machine learning library in C++. In *Tools with Artificial Intelligence*. IEEE Computer Society Press, 1996. `http://www.sgi.com/tech/mlc`.

[15] Y. Lambdan and H. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proc. ICCV*, pages 238–249, 1988.

[16] H. Murase and S. Nayar. Visual learning and recognition of 3-d objects from appearance. *IJCV*, 14:5–24, 1995.

[17] A. Pope and D. Lowe. Vista: A software environment for computer vision research. In *Proc. CVPR*, 1994.

[18] J. Quinlan. *C4.5 : programs for machine learning*. Morgan Kaufmann Publishers, 1993.

[19] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *Proc. CVPR*, 1998.

[20] A. Selinger and R. C. Nelson. A perceptual grouping hierarchy for appearance-based 3d object recognition. Technical Report 690, University of Rochester Computer Science Department, May 1998.

[21] P. Viola and W. M. W. III. Alignment by maximization of mutual information. *IJCV*, 24(2):137–154, 1997.

[22] P. Viola and M. Jones. Robust real-time object detection. Technical Report CRL 2001/01, Compaq Cambridge Research Laboratory, 2001.