

# A Cluster-Based Statistical Model for Object Detection

Thomas D. Rikert

Michael J. Jones

Paul Viola

Artificial Intelligence Lab  
Mass. Inst. of Tech.  
545 Technology Sq  
Cambridge, MA 02139  
tdrikert@ai.mit.edu

Cambridge Research Lab  
Compaq Computer Corp.  
One Kendall Sq., Bldg 700  
Cambridge, MA 02139  
mjones@crl.dec.com

Artificial Intelligence Lab  
Mass. Inst. of Tech.  
545 Technology Sq  
Cambridge, MA 02139  
viola@ai.mit.edu

## Abstract

*This paper presents an approach to object detection which is based on recent work in statistical models for texture synthesis and recognition [7, 4, 23, 17]. Our method follows the texture recognition work of De Bonet and Viola [4]. We use feature vectors which capture the joint occurrence of local features at multiple resolutions. The distribution of feature vectors for a set of training images of an object class is estimated by clustering the data and then forming a mixture of gaussian model. The mixture model is further refined by determining which clusters are the most discriminative for the class and retaining only those clusters. After the model is learned, test images are classified by computing the likelihood of their feature vectors with respect to the model. We present promising results in applying our technique to face detection and car detection.*

## 1 Introduction

This paper is a response to a scientific question that has its root both in computational vision and visual psychology: “What underlies the human ability to recognize objects under *deformation* and changes in pose?” Take for example the recognition of a jacket casually tossed to the floor. Somehow the almost infinite variation in appearance is easily captured by the observer’s model of the jacket. It seems unlikely that recognition of the jacket’s image is based on purely geometric reasoning. It also seems unlikely that the observer has built a complex model of jacket deformation painstakingly acquired over many thousands of examples. The hypothesis addressed in this paper is that object recognition under these very difficult conditions relies on texture recognition.

Motivation for this experimental investigation has come from recent successes in texture representation and recognition [7, 4, 23, 17]. Each of these approaches

can be used both to recognize texture as well as generate novel images. Based on the quality of these generated textures, it is clear that these four approaches have gone far beyond the previous state of the art in texture modeling. Taken together these new results have destroyed classical distinctions between textures, noisy textures and highly structured patterns.

The Heeger and Bergen approach is perhaps the most efficient of the four, but it has some difficulty in modelling highly structured patterns. While Zhu, Wu and Mumford is the most formal and well grounded of the four, it currently lacks an efficient algorithm for learning and recognition. The De Bonet and Viola approach combines the efficiency and simplicity of the Bergen and Heeger model with the modeling power of Zhu, Wu and Mumford. In this paper we will extend the work of De Bonet and Viola in an effort to model object detection under challenging circumstances.

The main contribution of this paper is the investigation of a new framework for object class detection based on an extended texture model (e.g. face or car detection). The advantage of this approach is that it can handle larger variations in the appearance of the object class than previous techniques. Starting from the De Bonet and Viola texture model, we will extend it to allow learning from hundreds or thousands of images. In order to do this we must eliminate much of the complexity in the density model. Complexity is removed in two ways: i) using an unsupervised approach to cluster the features space; ii) using a supervised approach to maximize the discriminative ability of the density estimate.

## 2 Related Work

Recently there has been a surge in interest in statistical object recognition models as a means of handling, or perhaps ignoring, the wide variations that are observed in natural images. Many of the strongest re-

sults are in face detection and recognition: [20, 19, 13]. A statistical modelling approach can potentially provide a principled mechanism for learning which properties of images are important for recognition and which are not. One criticism of these approaches is that they attempt to model the entire image as a single rigid patch – making it difficult to model changes in pose and feature location. More recently these techniques have been generalized to include schemes for modelling deformations in the image plane [1, 8, 5, 10]. These techniques not only learn a set of allowed variations in the image values, but also a set of allowed variations in pixel location. Nevertheless reliable detection and recognition of faces across a wide variety of faces is not yet a solved problem.

More recently a number of more general statistical models of recognition have been proposed: [21, 15, 2]. These attempt to model the appearance of localized features and then separately model feature location. While these are steps in the right direction, these approaches often require a great deal of computational time both for model learning and for image recognition.

Each of the above approaches attempts to model the distribution of images directly. Our approach instead attempts to model the distribution of multi-scale features. von der Malsburg and colleagues have shown excellent results on face recognition using a similar set of multi-scale features [9, 22]. Recently, Papageorgiou *et al.* have proposed multi-scale features for object class detection [11]. They use Haar wavelets computed at particular positions in the image to form feature vectors for a support vector machine classifier. Our approach is much more radical in its disregard for feature location.

Schiele and Crowley’s work on multidimensional receptive field histograms [14] also has some similarities to our work. They also look at the distribution of feature vectors formed from filter responses at different scales. Their focus differs in that they are looking at building models of single objects as opposed to object classes. Also their feature vectors do not use the parent vector structure that we use. Furthermore, they use fairly low dimensional feature vectors which makes the use of histograms practical.

### 3 The Statistical Model

This paper proposes a general statistical model for learning an object class from a set of example images and correctly classifying new images according to their membership in the class. Our framework follows De Bonet and Viola [4] in that it models the distribution of “parent vectors” in each training image.

The parent vectors are a collection of filter responses at different scales of a steerable image pyramid [16]. Figure 1 illustrates the structure of a parent vector. We estimate the distribution of these parent vectors from many training images using a mixture of gaussian model. We show results using this framework for face detection and car detection.

De Bonet and Viola’s texture recognition work modeled the distribution of parent vectors using a Parzen window density estimator. The Parzen density estimator, while quite flexible, requires time proportional to the quantity of training data. In our experiments we will use over a thousand training images. If evaluated in a naive fashion, the resulting density estimator would require many minutes to compute. Instead of the Parzen window model, we use a clustering algorithm to find a relatively small number of significant clusters of parent vectors from hundreds of training images. From these clusters a mixture of gaussian model is used to approximate this distribution of parent vectors.

The motivation for expecting parent vectors to make useful features for object detection comes from their use in texture synthesis [3, 4]. In that work, an example texture was first analyzed by finding its parent vectors. Then perceptually similar textures were synthesized by creating a new parent vector tree in which similar parent vectors were randomly interchanged. This representation captures the structure inherent in the example texture, but also allows for random variations which do not break the structure. What happens when we use face images as input to this texture synthesis framework? Figure 2 shows some examples. Each of these images was synthesized from a texture model built from many face images as described in [3]. Notice how features (such as eyes or nose) in one face are replaced by corresponding features from another face. From this figure it appears that the parent vector representation does capture some important structure even for face images.

### 4 Overview

The basis of our framework is using a mixture of gaussian model to estimate the distribution of parent vectors from a large set of example images of an object class. We present two datasets as examples: human faces and side views of cars.

Our method for building a model can be divided into four steps:

1. Apply multi-scale, multi-orientation filters to the training images to produce parent vectors.
2. Find clusters of similar parent vectors.

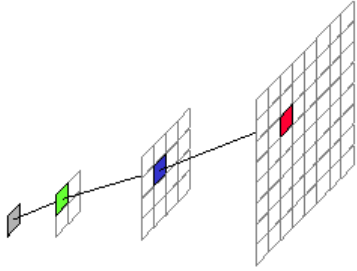


Figure 1: Illustration of parent vector structure. The grids represent an image pyramid. Each pixel is associated with a set of filter values capturing local features. The chain of line segments show the pixels whose filter values make up a single parent vector.



Figure 2: Texture synthesis examples using a set of face images as input textures. See text for an explanation.

3. Find an optimal subset of clusters for detection.
4. Build the final model as a mixture of multidimensional gaussian kernels centered on each cluster, with weight proportional to the population in the cluster and mean and variance proportional to the mean and variance of the cluster.

The same procedure is used to build an out-of-class (or background) model. The out-of-class model is combined with the in-class model using Bayes' rule to yield the probability of in-class given a parent vector.

To classify a test image, the parent vectors of the test image are computed. Then the average probability of the parent vectors from the test image is computed. If this percentage is above some threshold, the test image is classified as in-class.

Each of these steps is described in detail below.

## 5 Computing Parent Vectors

Parent vectors are computed from a multiresolution wavelet transform as described by De Bonet and Viola [4]. We will use the same notation as in [4]. First, a

gaussian pyramid is created from an input image  $I$ :  $G_0 = I$ ,  $G_1 = 2 \downarrow (g \otimes G_0)$  and  $G_{i+1} = 2 \downarrow (g \otimes G_i)$ , where  $2 \downarrow$  downsamples an image by a factor of 2 in each dimension and  $g$  is a low pass filter. At each level of the pyramid, a series of filter functions are applied:  $F_j^i = f_i \otimes G_j$ , where the  $f_i$ 's are oriented derivative filters. For every pixel in an image define the *parent vector* of that pixel:

$$\vec{V}(x, y) = \left[ F_0^0(x, y), F_0^1(x, y), \dots, F_0^N(x, y), \right. \\ \left. F_1^0(\lfloor \frac{x}{2} \rfloor, \lfloor \frac{y}{2} \rfloor), F_1^1(\lfloor \frac{x}{2} \rfloor, \lfloor \frac{y}{2} \rfloor), \dots, F_1^N(\lfloor \frac{x}{2} \rfloor, \lfloor \frac{y}{2} \rfloor), \dots \right. \\ \left. F_M^0(\lfloor \frac{x}{2^M} \rfloor, \lfloor \frac{y}{2^M} \rfloor), F_M^1(\lfloor \frac{x}{2^M} \rfloor, \lfloor \frac{y}{2^M} \rfloor), \dots, \right. \\ \left. F_M^N(\lfloor \frac{x}{2^M} \rfloor, \lfloor \frac{y}{2^M} \rfloor) \right] \quad (1)$$

where  $M$  is the top level of the pyramid and  $N$  is the number of features.

As depicted in figure 1, each parent vector corresponds to exactly one pixel in the original image, and stores the wavelet coefficient for each scale and orientation at that pixel location. The high-pass and low-pass residual values are also included in the vector.

## 6 Estimating the distribution

The model we build to represent an object class is an estimate of the distribution of parent vectors from a set of example images of the object class. To estimate the distribution of parent vectors, a clustering algorithm is first run on the data and then the resulting clusters are used to build a mixture of gaussian model as in [12].

We have found the standard k-means algorithms [6] for clustering to be too computationally expensive for our purposes. The problem is we have very many data points and the data requires very many clusters to estimate it well. For example, we have a training set of over 1 million parent vectors in a 26 dimensional space that requires hundreds of thousands of clusters to represent it well. We can use our results from texture synthesis to give a good idea of the maximum distance that should be allowed between data points belonging to the same cluster. Using distance thresholds from synthesis, we can take a bottom-up approach to clustering which starts with every data point as its own cluster and then combine clusters which are close together according to our distance function. We use the function, `near()`, defined below to determine if two parent vectors are close.

```

near( $\mathbf{v}_1, \mathbf{v}_2$ ) {
  flag = 1
  for  $k=1$  to  $d$ 
    if  $D(\mathbf{v}_1[k], \mathbf{v}_2[k]) > T_k$  then
      flag = 0
  return flag
}

```

where  $d$  is the dimension of the parent vectors,  $\{T_k\}$  is a set of thresholds and

$$D(\mathbf{v}_1[k], \mathbf{v}_2[k]) = \frac{|\mathbf{v}_1[k] - \mathbf{v}_2[k]|}{|\mathbf{v}_1[k]| + |\mathbf{v}_2[k]| + 1}. \quad (2)$$

The intuition behind the distance function,  $D()$ , is that we want to look at the absolute difference between corresponding components of two parent vectors and scale that difference by the size of the components. We achieve the scaling by dividing by the sum of the magnitudes of the two components (plus one to avoid division by zero).

The following pseudo code describes our clustering algorithm:

```

N = number of parent vectors
M = number of clusters
Let  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_N$  be the list of N
d-dimensional parent vectors

M = N;

Make a cluster for each parent vector with
mean  $\mu_i = \mathbf{v}_i$  and variance  $\sigma_i^2 = 0$ 

do {
  for  $i=1$  to  $M-1$ 
    for  $j=i+1$  to  $M$ 
      if near( $\mu_i, \mu_j$ ) then {
        combine cluster  $i$  and  $j$ 
        (keeping track of the
        mean, variance and
        population count)

        M = M - 1
      }
} until the decrease in M is insignificant

```

The resulting clusters are used to build a gaussian mixture model to generalize the distribution by placing a multidimensional gaussian kernel at the center of each cluster. Popat and Picard [12] use a similar procedure to model distributions obtained from images. The probability of vector  $\mathbf{v}$  in the model for class  $C$  is given by

$$P_C(\mathbf{v}) = \sum_{m=1}^M w_m \prod_{i=1}^d k_{m,i}(\mathbf{v}[i]) \quad (3)$$

where the kernel  $k$  is the one-dimensional gaussian

$$k_{m,i}(\mathbf{v}[i]) = \frac{1}{\sqrt{2\pi\hat{\sigma}_{m,i}^2}} e^{-\frac{(\mathbf{v}[i] - \mu_{m,i})^2}{2\hat{\sigma}_{m,i}^2}} \quad (4)$$

and  $w_m = N_m/N$  where  $N_m$  is the number of parent vectors in cluster  $m$ .

## 7 Using discriminative analysis to refine the distribution

The clustering algorithm eventually produces a set of feature clusters. If the number of parent vectors in the training set is large (say over one million) then the number of clusters will often also be large (sometimes hundreds of thousands). We wish to reduce the number of clusters for two reasons: 1) to improve the accuracy of the model by keeping only clusters which discriminate between in-class and out-of-class parent vectors and 2) to increase the speed of evaluating test images by having fewer gaussians to compute.

We use discriminative analysis to achieve this. The idea is to take a set of in-class and out-of-class training images and create two histograms showing how many times parent vectors from each set fall near an in-class cluster. Then we keep only those clusters which have a significantly larger count for in-class parent vectors than for out-of-class parent vectors. This reduces our in-class model to the most discriminative clusters. We can use the same method to reduce the number of clusters in the non-face model.

There are many possible tests to determine whether the count for in-class parent vectors is ‘‘significantly’’ more than the count for out-of-class vectors. The test we are currently using is as follows. Let  $hist_C[i]$  be the count of in-class parent vectors which are near cluster  $i$ . Let  $hist_{\bar{C}}[i]$  be the count of out-of-class parent vectors which are near cluster  $i$ . Then if

$$\frac{hist_C[i]}{hist_C[i] + hist_{\bar{C}}[i]} > \Theta \quad (5)$$

then cluster  $i$  is a discriminative cluster and therefore retained. The threshold  $\Theta$  can be any real number between 0 and 1. We used a value of 0.8 in the experiments that follow.

## 8 Classifying a test image

The previous sections have described using parent vectors as features and building models of the distributions of parent vectors from example in-class and

out-of-class images. The in-class and out-of-class mixture models for a class  $C$  give  $P(\mathbf{v}|C) = P_C(\mathbf{v})$  and  $P(\mathbf{v}|\bar{C}) = P_{\bar{C}}(\mathbf{v})$ . We can use Bayes rule to yield

$$P(C|\mathbf{v}) = \frac{P(\mathbf{v}|C)P(C)}{P(\mathbf{v}|C)P(C) + P(\mathbf{v}|\bar{C})P(\bar{C})}. \quad (6)$$

This equation gives the probability of class  $C$  given a single parent vector  $\mathbf{v}$ .

The priors  $P(C)$  and  $P(\bar{C})$  can be chosen arbitrarily such that  $P(C) + P(\bar{C}) = 1$ . The choice does not effect the receiver operating characteristics curve which expresses the relationship between correct detections and false positives for classifying parent vectors [6].

The question remains of how to use this probability model to determine if a *set* of parent vectors from a test image are more likely to come from an in-class image than an out-of-class image. De Bonet and Viola [4] suggest comparing the distribution of parent vectors from a test image against the in-class model distribution using the Kullback-Liebler (KL) divergence. The problem with this idea in our case is that the distribution from a single test image will probably not look like the distribution from a large set of training images. The reason is the parent vectors from an image are not independent of each other. They form a tree structure. The model distribution is more like a collection of these trees than like a single tree. Thus, the two distributions will probably not be similar.

Instead of comparing distributions, we have used the simple idea of calculating the average probability of  $C$  given each of the parent vectors in a single test image and then thresholding this value to classify the image. Thus, if

$$\frac{\sum_{j=1}^N P(C|\mathbf{v}_j)}{N} > t \quad (7)$$

for some threshold  $t$  then the image is classified as belonging to  $C$ , otherwise it is classified as  $\bar{C}$ .

The average probability can be viewed as one of many possible statistics we could calculate. If we estimate the distribution of this statistic for the training data, we could then evaluate the likelihood of this statistic computed for a test image. Using multiple such statistics to evaluate test images could produce more accurate tests for a classifier. We leave this idea for future work.

## 9 Results

### 9.1 Face Detection

We have built a model of faces from a set of 1060 face images cropped from photographs found on the

World Wide Web. The faces were chosen to be approximately frontal. There were small variations in scale as well as large variations in lighting and image quality. Each face image was scaled to be 32 by 32 pixels, converted to gray scale and then histogram equalized to reduce the variations from lighting. Some example face images are shown in figure 3 (shown before histogram equalization).



Figure 3: Example face images from the training set.

To build a background model of non-face images, we selected windows of random size and position from a set of Web images which did not contain people. The windows were then scaled to be 32 by 32 pixels and histogram equalized as with the faces. We used 1016 such non-face examples.

The parent vectors for each of the example faces and non-faces were computed as described in section 5. This yielded 1,085,440 ( $= 1060 \times 32 \times 32$ ) face parent vectors and 1,040,384 ( $= 1016 \times 32 \times 32$ ) non-face parent vectors. We used parent vectors with 4 scales and 6 oriented edge filters per scale. Including the high and low pass residuals, this resulted in parent vectors with 26 components.

To construct face and non-face models we first applied the clustering algorithm described in section 6 to the two sets of parent vectors separately. For the face model we used a threshold of 0.7 for all  $T_i$  in the function `near()`. For non-faces we used a threshold of 0.8 for all  $T_i$ . The clustering algorithm yielded 351,615 different clusters for the face parent vectors and 155,222 different clusters for the non-face parent vectors.

Next we applied discriminative analysis as discussed in section 7 to reduce the number of clusters and improve the models. This analysis yielded a face model with 1447 clusters and a non-face model with 483 clusters. A mixture of gaussian model was built from these clusters as described in section 6.

We can improve our intuition for the cluster-based model by verifying that some clusters correspond to

particular perceptual features in a face. In other words, does a cluster encode a feature in the image such as an eye or lip that is useful for face detection? To test this hypothesis, we collected a new set of 266 face images and computed the parent vectors for these images. We then selected a parent vector cluster from the model and highlighted test image regions containing parent vectors from within this cluster. Cluster membership was determined using the distance function and thresholds defined before. Figure 4 shows the location where a “lip cluster” responds in several face images. While this cluster only responds to 6% of the *test* faces, in almost every case the response is localized near the lip region. In each of the remaining faces there is no response. Although this is a small percentage of faces, together the 1447 clusters provide enough coverage to detect most faces.



Figure 4: A cluster was chosen from the face model and compared to all parent vectors in each test image. The white boxes show the position in each image where a parent vector was near the cluster. This cluster apparently represents a lip-corner feature.

To test the model we used the 266 face images mentioned before and a new set of 2500 non-face images. For each image, the parent vectors were computed and then the test in equation 7 was applied to classify the image.

The results are shown as a receiver operating characteristics curve (ROC curve) in figure 5. The ROC curve shows promising preliminary results. For example, we get 90% correct detection with a false positive rate of 1.2% or 70% correct detection with 0.28% false positives.

It is difficult to compare these results with those of Sung and Poggio or Rowley and Kanade. These systems process entire images; first decomposing them into a set of overlapping patches at multiple scales, and then classifying each patch. There are ten’s of thousands of such patches in each image, most containing only background. While it is a daunting task to reject each and every background patch, it is im-

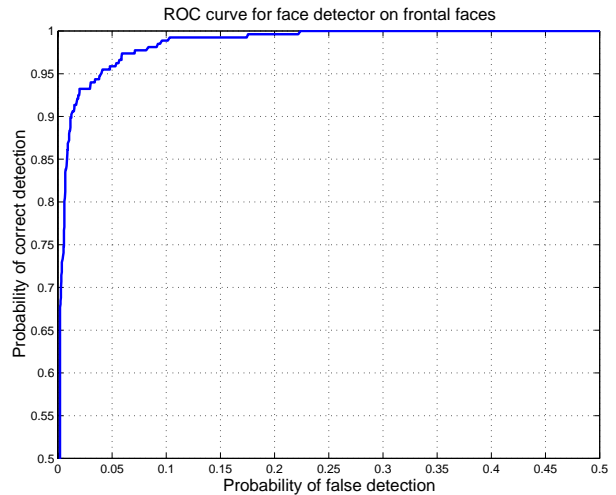


Figure 5: Receiver operating characteristics curve for a test set of 266 frontal faces and 2500 non-faces. Note that to focus on the interesting part of the graph, the correct detection rate axis begins at 50%.

portant to point out that since they overlap many of the patches are highly redundant. A patch that is correctly rejected because it does not contain a face is very likely to be rejected if it is shifted by a single pixel. A similar issue arises in the detection of faces. Since a single face will appear in many overlapping patches, there will be many opportunities to detect it.

The above ROC curve demonstrates performance on independently selected patches. There is no redundancy in this dataset. We believe that when placed in an end-to-end system this detector will demonstrate improved detection and rejection rates. One obvious refinement is to let the system bootstrap its non-face training set by adding false positive images into the non-face training set and relearning the model. This work is currently underway.



Figure 6: Example non-frontal face images which the face detector correctly classified.

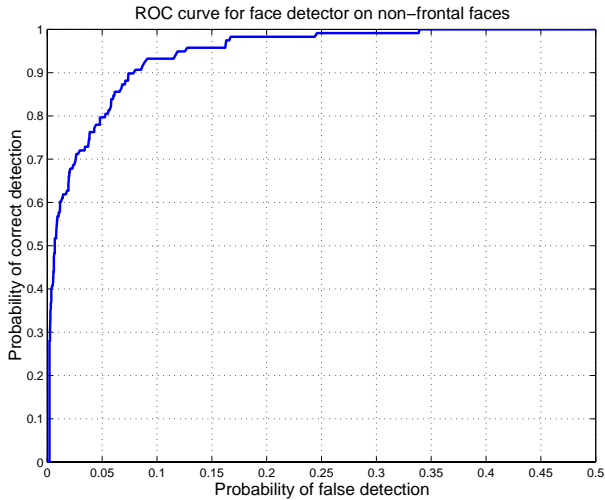


Figure 7: Receiver operating characteristics curve for a test set of 266 non-frontal faces and 2500 non-faces.

The main advantage of our framework for object detection is that it should be able to handle more variability than most other classifier-based methods. With faces this means that we expect parent vectors for frontal face images to be similar to parent vectors for non-frontal faces. To test this, we evaluated our face detector on a set of non-frontal faces, some of which are shown in figure 6. We used the same face detector just described which was trained on frontal faces. The preliminary results support the robustness of our framework. The ROC curve for a test set of 118 non-frontal faces and the same set of 2500 non-faces is shown in figure 7. The results are surprisingly good considering that non-frontal faces were not used in the training set. For example, we get a correct detection rate of 80% with a false positive rate of 5% or a correct detection rate of 60% with a false positive rate of 1.2%. These results should be improved by including non-frontal faces in the training set.

## 9.2 Car Detection

As a second example, we also learned a model for side views of cars. We only had a database of 48 car images available for this experiment. However, since all the cars were photographed under similar conditions, our intuition was that this should be sufficient to learn a model. We split the set into 24 training images and 24 test images. We also used a set of 1024 non-car images to build a background model. The image size used was 52 pixels in width by 16 pixels in height. Figure 8 shows a few example cars. All of the car images were acquired by taking photographs of toy cars.

The parent vectors consisted of the values from 6 oriented edge filters computed over 3 scales. Including the high and low resolution residuals, the parent vectors contained 20 components each. Initial clustering of the car parent vectors yielded 12,295 clusters using a threshold of  $T_i = 0.5$  in the distance function of equation 2. Initial clustering of the parent vectors from the 1024 non-car images yielded 32,405 clusters using a threshold of  $T_i = 0.8$ . After discriminative analysis, the car model contained 1229 clusters and the non-car model contained 1001 clusters.



Figure 8: Eight of the 24 example cars used to train the car detector.

This model was then tested on a test set of 24 car and 200 non-car images. The resulting ROC curve is shown in figure 9. The results are very good. Only 1 false positive was made with 100% correct detections.

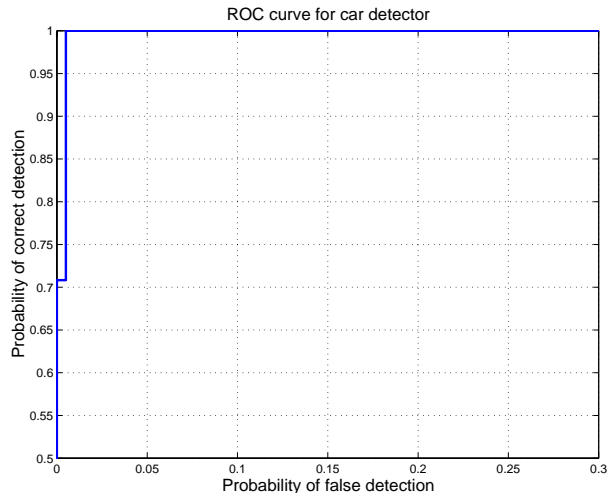


Figure 9: Receiver operating characteristics curve for a test set of 24 side views of cars and 200 non-cars.

## 10 Conclusions and future work

We have presented a new approach to detecting classes of objects which is based on De Bonet and Viola's recent technique for representing textures. The results we have obtained for face detection and car detection are very promising. They show that object detection for an object class with large variations can rely on texture recognition.

Our future work in this area will focus on implementing a face detector that searches over positions and scales. We also intend to improve the computational expense of the method to make it a practical solution for object detection.

### Acknowledgments

This research was supported by Compaq Computer Corporation's Cambridge Research Laboratory. Thanks to L.J. Ruell and Gene Preble for collecting the face database used in this research.

### References

- [1] D. Beymer, A. Shashua and T. Poggio, "Example Based Image Analysis and Synthesis", *A.I. Memo 1431*, MIT, 1993.
- [2] M.C. Burl, T.K. Leung, P. Perona, "Face Localization via Shape Statistics", in *Int'l Conf. on Automatic Face and Gesture Recognition*, IEEE, 1995, pp. 154-159.
- [3] J. De Bonet, "Multiresolution sampling procedure for analysis and synthesis of texture images," in *Computer Graphics*. ACM SIGGRAPH, 1998.
- [4] J. De Bonet and P. Viola, "Texture recognition using a non-parametric multi-scale statistical model," in *CVPR*, 1998.
- [5] G.J. Edwards, C.J. Taylor and T.F. Cootes, "Interpreting Face Images using Active Appearance Models" in *Int'l Conf. on Automatic Face and Gesture Recognition*, IEEE, 1998, pp. 300-305.
- [6] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, Inc., San Diego, CA. 1990.
- [7] D. Heeger and J. Bergen, "Pyramid-based texture analysis/synthesis," In *Proc. of ACM SIGGRAPH*, August 1995, pp 229-238.
- [8] M. Jones and T. Poggio, "Multidimensional Morphable Models: A Framework for Representing and Matching Object Classes" in *IJCV*, Volume 29, No. 2, August 1998, pp. 107-131.
- [9] M. Lades, *et al.*, "Distortion Invariant Object Recognition in the Dynamic Link Architecture", *IEEE Trans. on Computers*, Vol 42, No 3, March 1993, pp. 300-311.
- [10] B. Moghaddam, C. Nastar and A. Pentland, "Bayesian Face Recognition using Deformable Intensity Surfaces" in *CVPR*, 1996.
- [11] C. Papageorgiou, M. Oren and T. Poggio, "A General Framework for Object Detection," in *ICCV*, January 1998, pp 555-562.
- [12] K. Popat and R. Picard, "Cluster-based probability model and its application to image and texture processing," *IEEE Trans. on Image Processing*, Vol. 6, No. 2, Feb 1997, pp. 268-284.
- [13] H. Rowley, S. Baluja, T. Kanade, "Human Face Detection in Visual Scenes," *Technical Report CMU-CS-95-158R*, CMU, 1995.
- [14] B. Schiele and J. Crowley, "Recognition without Correspondence using Multidimensional Receptive Field Histograms" *MIT Media Laboratory Perceptual Computing Section TR No. 453*, December 1997.
- [15] H. Schneiderman and T. Kanade, "Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition", in *CVPR*, 1998, pp. 45-51.
- [16] E. Simoncelli and W. Freeman, "The Steerable Pyramid: A flexible architecture for multi-scale derivative computation," In *IEEE Int'l Conf. on Image Processing*, pp. 444-447, Washington, D.C., October 1995.
- [17] E. Simoncelli and J. Portilla, "Texture Characterization via Joint Statistics of Wavelet Coefficient Magnitudes," in *c. IEEE Int'l Conf. on Image Processing*, 1998.
- [18] E. Simoncelli and R. Buccigrossi, "Embedded Wavelet Image Compression Based on a Joint Probability Model," in *IEEE Int'l Conf. on Image Processing*, 1997.
- [19] K. Sung and T. Poggio, "Example-based learning for view-based human face detection", *A.I. Memo 1521*, MIT, December 1994.
- [20] M. Turk and A. Pentland, "Face Recognition Using Eigenfaces" in *CVPR*, 1991, pp. 586-591.
- [21] P. Viola, "Complex Feature Recognition: A Bayesian Approach for Learning to Recognize Objects", *A.I. Memo 1591*, MIT, November, 1996.
- [22] L. Wiskott, *et al.*, "Face Recognition by Elastic Bunch Graph Matching", *IEEE Int'l Conf. on Image Processing*, Vol. I, 1997.
- [23] S. Zhu, Y. Wu and D. Mumford, "Filters, Random Fields and Maximum Entropy (FRAME) - Towards A Unified Theory For Texture Modeling" in *IJCV*, Vol 27, No 2, March 1998, pp. 107-126.