MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Fast Multi-view Face Detection

Jones, M.; Viola, P.

TR2003-96 August 2003

Abstract

This paper extends the face detection framework proposed by Viola and Jones 2001 to handle profile views and rotated faces. As in the work of Rowley et al. 1998, and Schneiderman et al. 2000, we build different detectors for different views of the face. A decision tree is then trained to determine the viewpoint class (such as right profile or rotated 60 degrees) for a given window of the image being examined. This is similar to the approach of Rowley et al. 1998. The appropriate detector for that viewpoint can then be run instead of running all detectors on all windows. This technique yields good results and maintains the speed advantage of the Viola-Jones detector.

Publication History:–

1. First printing, TR2003-96, July 2003

Fast Multi-view Face Detection

Michael J. Jones
mjones@merl.com
Mitsubishi Electric Research Laboratory
201 Broadway
Cambridge, MA 02139

Paul Viola
viola@microsoft.com
Microsoft Research
One Microsoft Way
Redmond, WA 98052

July 15, 2003

Abstract

This paper extends the face detection framework proposed by Viola and Jones 2001 to handle profile views and rotated faces. As in the work of Rowley et al 1998. and Schneiderman et al. 2000, we build different detectors for different views of the face. A decision tree is then trained to determine the viewpoint class (such as right profile or rotated 60 degrees) for a given window of the image being examined. This is similar to the approach of Rowley et al. 1998. The appropriate detector for that viewpoint can then be run instead of running all detectors on all windows. This technique yields good results and maintains the speed advantage of the Viola-Jones detector.

1. Introduction

There are a number of techniques that can successfully detect frontal upright faces in a wide variety of images [11, 7, 10, 12, 3, 6]. While the definition of "frontal" and "upright" may vary from system to system, the reality is that many natural images contain rotated or profile faces that are not reliably detected. There are a small number of systems which explicitly address non-frontal, or non-upright face detection [8, 10, 2]. This paper describes progress toward a system which can detect faces regardless of pose reliably and in real-time.

This paper extends the framework proposed by Viola and Jones [12]. This approach is selected because of its computational efficiency and simplicity.

One observation which is shared among all previous related work is that a multi-view detector must be carefully constructed by combining a collection of detectors each trained for a single viewpoint. It appears that a monolithic approach, where a single classifier is trained to detect all poses of a face, is unlearnable with existing classifiers. Our informal experiments lend support to this conclusion, since a classifier trained on all poses appears to be hopelessly inaccurate.

This paper addresses two types of pose variation: non-frontal faces, which are rotated out of the image plane, and non-upright faces, which are rotated in the image plane. In both cases the multi-view detector presented in this paper is a combination of Viola-Jones detectors, each detector trained on face data taken from a single viewpoint.

Reliable non-upright face detection was first presented in a paper by Rowley, Baluja and Kanade [8]. They train two neural network classifiers. The first estimates the pose of a face in the detection window. The second is a conventional face detector. Faces are detected in three steps: for each image window the pose of "face" is first estimated; the pose estimate is then used to de-rotate the image window; the window is then classified by the second detector. For non-face windows, the poses estimate must be considered random. Nevertheless, a rotated non-face should be rejected by the conventional detector. One potential flaw of such a system is that the final detection rate is roughly the product of the correct classification rates of the two classifiers (since the errors of the two classifiers are somewhat independent).

One could adopt the Rowley et al. three step approach while replacing the classifiers with those of Viola and Jones. The final system would be more efficient, but not significantly. Classification by the Viola-Jones system is so efficient, that derotation would dominate the computational expense. In principle derotation is not strictly necessary since it should be possible to construct a detector for rotated faces directly. Detection becomes a two stage process. First the pose of the window is estimated and then one of N rotation specific detectors is called upon to classify the window.

In this paper detection of non-upright faces is handled using the two stage approach. In the first stage the pose of each window is estimated using a decision tree constructed using features like those described by Viola and Jones. In the second stage one of N pose specific Viola-Jones detectors are used to classify the window.

Once N pose specific detectors are trained and available, an alternative detection process can be tested as well. In this case all N detectors are evaluated and the union of their de-

tections are reported. We have found that this simple try-all-poses system in fact yields a slightly superior receiver operating characteristics (ROC) curve, but is about $\frac{N}{2}$ times slower. This contradicts a conjecture by Rowley in his thesis, which claims that the false positive rate of the try-all-poses detector would be higher.

Both for the estimation of pose, and for the detection of rotated faces, we found that the axis aligned rectangle features used by Viola and Jones were not entirely sufficient. In this paper we present an efficient extension of these features which can be used to detect diagonal structures and edges.

In this paper we have also investigated detectors for non-frontal faces (which include profile faces). Using an identical two stage approach, an efficient and reliable non-frontal face detector can be constructed. We compare our results with those of Schneiderman et al. which is one of the few successful approaches to profile face detection. Li et al. [2] also built a profile detector based on a modification of the Viola-Jones detector, but did not report results on any profile test set.

The main contributions of this paper are:

- A demonstration that the Viola-Jones framework can be extended to rotated and profile faces.
- A more general two-stage approach to multi-view detection which can be applied to rotated faces as well as profile views.
- It shows that the advantage of the two or three stage approach is speed and not false positive rate as Rowley had hypothesized.
- A new set of rectangle features that are useful for rotated face detection.

The remainder of this paper i) reviews the face detection framework and our extensions; ii) describes the pose estimation classifier; iii) describes our experimental results; and iv) concludes with a discussion.

2. Face Detection Framework

As is typical for face detection, the input image is scanned across location and scale. At each location an independent decision is made regarding the presence of a face. This leads to a very large number of classifier evaluations; approximately 50,000 in a 320×240 image.

Viola-Jones use a boosted collection of features to classify image windows. Following the AdaBoost algorithm of Freund and Schapire [1] a set of weak binary classifiers is learned from a training set. Each classifier is a simple function made up of rectangular sums followed by a threshold. Since they can be visualized and interpreted, Viola and Jones call these classifiers features.

In each round of boosting one feature is selected, that with the lowest weighted error. The feature is assigned a weight in the final classifier using the confidence rated AdaBoost procedure of Schapire and Singer [9]. In subsequent rounds incorrectly labeled examples are given a higher weight while correctly labeled examples are given a lower weight.

In order to reduce the false positive rate while preserving efficiency, classification is divided into a cascade of classifiers. An input window is passed from one classifier in the cascade to the next as long as each classifier classifies the window as a face. The threshold of each classifier is set to yield a high detection rate. Early classifiers have few features while later ones have more so that easy non-faces are quickly discarded. Each classifier in the cascade is trained on a negative set consisting of the false positives of the previous stages. This allows later stages to focus on the harder examples.

In order to train a full cascade to achieve very low false positive rates, a large number of examples are required, both positive and negative. The number of required negative examples is especially large. After 5 stages the false positive rate is often well below 1%. Therefore over 99% of the negative data is rejected and is unavailable for training subsequent stages.

The main parts of the Viola-Jones framework are described in more detail below.

2.1. Filters and Features

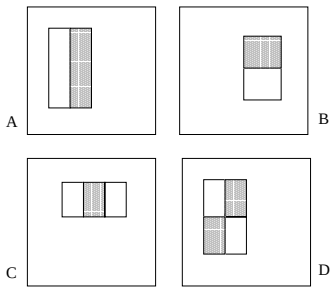

Figure 1: Example rectangle filters shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the gray rectangles.

Following [12] image features are called *Rectangle Features* and are reminiscent of Haar basis functions (see [4] for the use of Haar basis functions in pedestrian detection). Each rectangle feature, $h_j()$ is binary threshold function constructed from a threshold θ_j and a *rectangle filter* $f_j()$ which is a linear function of the image:

$$h_j(x) = \begin{cases} \alpha_j & \text{if } f_j(x) > \theta_j \\ \beta_j & \text{otherwise} \end{cases}$$

Here x is a 24×24 pixel sub-window of an image. Following Schapire and Singer [9], α_j and β_j are positive or negative votes of each feature set by Adaboost during the learning process.

Previously Viola and Jones used three types of rectangle filters. The value of a *two-rectangle filter* is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent (see Figure 1 A and B). A *three-rectangle filter* computes the sum within two outside rectangles subtracted from twice the sum in a center rectangle (see C). Finally a *four-rectangle filter* computes the difference between diagonal pairs of rectangles (see D).

Given that the base resolution of the classifier is 24 by 24 pixels, the exhaustive set of rectangle filters is quite large, over 100,000, which is roughly $O(N^4)$ where N=24 (i.e. the number of possible locations times the number of possible sizes). The actual number is smaller since filters must fit within the classification window. Note that unlike the Haar basis, the set of rectangle features is overcomplete[1].

Computation of rectangle filters can be accelerated using an intermediate image representation called the integral image [12]. Using this representation any rectangle filter, at any scale or location, can be evaluated in constant time.

2.2. Diagonal Filters

Experiments showed that the three types of filters given above were not sufficient to detect non-upright faces and non-frontal faces with sufficiently high accuracy. To address this we created a fourth type of rectangle filter that focuses on diagonal structures in the image window. These diagonal filters are illustrated in figure 2. They consist of four overlapping rectangles that combine to yield the blocky diagonal areas shown in the figure. They operate in the same way as the previous filters. The sum of the pixels in the dark gray shaded region is subtracted from the sum in the light gray shaded region. Using an integral image, the diagonal filter can be computed by only looking at the 16 corner pixels.

2.3. Classifier

The form of the final classifier returned by Adaboost is a perceptron - a thresholded linear combination of features. The weights on each feature are encoded in the α_j and β_j votes described earlier. The classifier is described by the following equation:

[1]A complete basis has no linear dependence between basis elements and has the same number of elements as the image space, in this case $24 \times 24 = 576$. The full set of filters is many times over-complete.

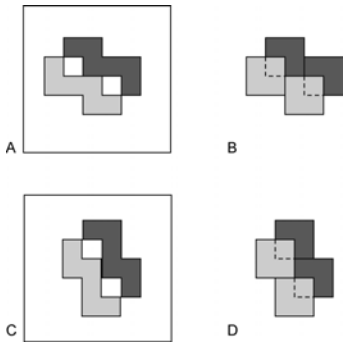

Figure 2: Example diagonal filters shown relative to the enclosing detection window. The sum of the pixels that lie within the light gray area is subtracted from the sum of pixels in the dark gray area. The two basic types of diagonal filters are shown in A and C. A diagonal filter is constructed from 4 rectangles as illustrated in B and D. The base rectangles can be any aspect ratio and any size that fits within the detector's query window.

$$C(x) = \begin{cases} 1 & \text{if } \sum_j h_j(x) > T \\ 0 & \text{otherwise} \end{cases}$$

2.4. Cascade

In order to greatly improve computational efficiency and also reduce the false positive rate, Viola and Jones use a sequence of increasingly more complex classifiers called a cascade. An input window is evaluated on the first classifier of the cascade and if that classifier returns false then computation on that window ends and the detector returns false. If the classifier returns true then the window is passed to the next classifier in the cascade. The next classifier evaluates the window in the same way. If the window passes through every classifier with all returning true then the detector returns true for that window. The more a window looks like a face, the more classifiers are evaluated on it and the longer it takes to classify that window. Since most windows in an image do not look like faces, most are quickly discarded as non-faces. Figure 3 illustrates the cascade.

3. Pose estimator

3.1. Purpose and Operation

Our approach to detecting faces from multiple views is to divide the space of poses into various classes and train different detectors for each pose class. In order to avoid the computational expense of having to evaluate every detector on every window in the input image, we use a two stage approach which first estimates the pose of the face in the window and then evaluates only the detector trained on that pose. When the pose estimator is evaluated on a non-face

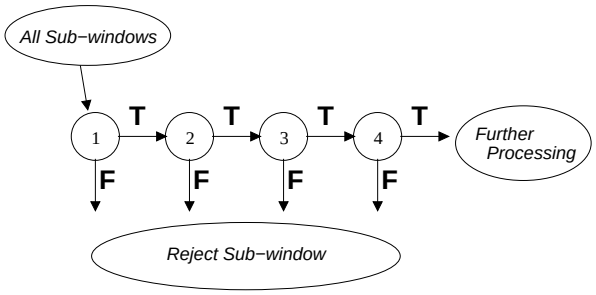

Figure 3: Schematic depiction of a the detection cascade. A series of classifiers are applied to every sub-window. The initial classifier eliminates a large number of negative examples with very little processing. Subsequent layers eliminate additional negatives but require additional computation. After several stages of processing the number of sub-windows have been reduced radically. Further processing can take any form such as additional stages of the cascade (as in our detection system) or an alternative detection system.

window, its output can be considered random. Any detector chosen to evaluate on a non-face window should return false. As mentioned earlier this approach is very similar to that of Rowley et al. [8].

3.2. Decision Tree

The pose estimator is thus a multi-class classifier. Furthermore, it needs to be about the same speed as a single face detector to make it advantageous to use over the try-all-poses approach. This speed constraint is a stringent one since a pose-specific face detector only evaluates about 8 features per window on average.

A decision tree classifier meets these constraints. Unlike a boosted classifier for multi-class problems, it is very straightforward to learn a multi-class decision tree. In addition a decision tree is quite efficient, since only one path from the root to a leaf is evaluated for each example. Even for very large and complex trees the number of features evaluated per window is logarithmic in the total number of nodes in the tree.

One criticism of decision trees is that they are somewhat brittle, and sometimes do not generalize well. This is one reason that the face detector uses boosting rather than decision trees. For this application we have found decision trees to be quite reliable.

In order to and maximize efficiency, rectangle features are used as the tests in the nodes of the tree (see Section 2.1). These features are fast and allow the reuse of the integral image.

3.3. Decision Tree Training

The decision tree learning algorithm follows the work of Quinlan [5]. In short the criteria for splitting nodes is mutual information, the node functions are rectangle features, and there is no pruning. Every node of the tree is split until a maximum leaf depth is attained or the leaf contains examples of only one node.

The training algorithm is almost identical to the boosting algorithm. The two main differences are the criteria for features selection (mutual information rather than Z criteria) and the splitting of the training set at each node.

4. Experiments

4.1. Non-upright Faces

The frontal face detector from Viola-Jones handles approximately ± 15 degrees of in-plane rotation. Given this, we trained 12 different detectors for frontal faces in 12 different rotation classes. Each rotation class covers 30 degrees of in-plane rotation so that together, the 12 detectors cover the full 360 degrees of possible rotations.

4.1.1 Detectors for each rotation class

The training sets for each rotation class consist of 8356 faces of size 24×24 pixels and over 100 million background (non-face) patches. The face examples are the same for each rotation class modulo the appropriate rotation. In practice, we only needed to train 3 detectors. One for 0 degrees (which covers -15 degrees to 15 degrees of rotation), one for 30 degrees (which covers 15 degrees to 45 degrees) and one for 60 degrees (which covers 45 degrees to 75 degrees). Because the filters we use can be rotated 90 degrees, any detector can also be rotated 90 degrees. So a frontal face detector trained at 0 degrees of rotation can be rotated to yield a detector for 90 degrees, 180 degrees and 270 degrees. The same trick can be used for the 30 degree and 60 degree detectors to cover the remaining rotation classes.

All of the resulting face detectors coincidentally turned out to have 35 layers of classifiers. They all took advantage of diagonal features (although for the frontal, upright detector, the added diagonal features did not improve the accuracy of the detector over previously trained versions). Training was stopped after new cascade layers ceased to significantly improve the false positive rate of the detector without significantly reducing its detection rate. This happened to be after 35 layers in all three cases.

4.1.2 Pose estimator

For the pose estimator we trained a decision tree with 1024 internal nodes (11 levels) to classify a frontal face into one

of the 12 rotation classes. The decision tree was trained using 4000 faces (also of size 24×24 pixels) for each of the 12 rotation classes. The set of faces for a particular rotation class used to train the decision tree was a subset of the faces used to train a single face detector (4000 of the 8356 faces). The limit of 4000 was imposed by memory and speed constraints of the decision tree training algorithm.

The resulting decision tree achieves 84.7% accuracy on the training set and 76.6% accuracy on a test set (consisting of the remaining 4356 faces for each rotation class). Because the pose estimator has multiple chances to predict the rotation class for each face when scanning, the loss in detection rate per face is much less than 23.4% (100% - 76.6%).

4.1.3 Two-stage rotated face detector

A two stage detector for rotated faces was created as described in section 3. It first runs the decision tree classifier on each window in the image and then runs the single face detector corresponding to the rotation class predicted by the decision tree. The decision tree takes approximately the same time to run as a single detector, so the running time of the two stage detector is about twice as long as the running time of a single detector. On a 320×240 pixel image, the two stage rotated face detector takes about 0.14 seconds on a 2.8 GHz Pentium 4. The try-all-rotations detector takes about 0.66 seconds (4.7 times longer). This is a little less than the 6 times speed-up for the two stage detector that you would expect because the decision tree is actually a little slower than the average rotated face detector.[2]

4.1.4 Results on the non-upright test set

This two stage rotated face detector was tested on the non-upright (also called "tilted") test set from Rowley et al. [8]. The non-upright test set consists of 50 images with 223 total faces. Some example detections are shown in figure 5. In the figure, the white bars on the detected boxes indicate the top of the head. A ROC curve for this two stage detector is given in figure 4. It plots the number of false positives on the non-upright test set versus the correct detection rate[3]. The number of false positives is plotted as opposed to the false positive rate to allow comparision with previous results. In our case, we examined 10,515,781 image windows in the test set so the number of false positives for a particular detection rate can be divided by this number to yield the false positive rate.

[2]These speeds are a little slower than one might expect from Viola and Jones's previous results. However, the running time of the average rotated face detector is about 2.2 times slower than the frontal upright face detector reported in Viola, Jones 2001.

[3]Note: when multiple detections significantly overlap, they are merged into a single detection.

Figure 4 also shows the ROC curve for the try-all-rotations detector. The try-all-rotations detector is only slightly more accurate but as noted above is nearly 5 times slower.

It is not obvious how to manipulate the detectors to get different points on the ROC curve since each detector is a cascade of classifiers each of which has a threshold and we have multiple such detectors. We use the same strategy as in Viola-Jones [12]. To decrease the false positve rate (and decrease the detection rate), the threshold of the final classifier is increased. To increase the detection rate (and increase the false positive rate), classifier layers are removed from the end of the cascade. This is done simultaneously for all of the detectors.

4.1.5 Comparison to Rowley et al.

Rowley et al's results on this test set were very similar. Unfortunately they only computed one point on their ROC curve. Rowley et al. reported a detection rate of 89.2% with 221 false positives using a three stage detector that determines the rotation class for the current image window then derotates the window and then evaluates the frontal upright detector on it. This compares with our result of 89.7% correct detections with 221 false positives. When testing all rotation classes on each window (there were 18 rotation classes in their case), Rowley et al. report a detection rate of 96.0% with 1345 false positives. This compares with our result of 95% with 1345 false positives. Their conclusion was that using the pose estimator lowered the detection rate as well as the false positive rate as compared to the try-all-rotations detector. However, this conclusion is not supported by their data because they only found a single point on the ROC curve for each of their detectors and these points are not comparable. In our case, by plotting ROC curves, one can see that the two approaches are very close with the try-all-rotations approach being slightly better at most places on the ROC curve. The big win for the two-stage detector therefore is not in false positive rate but rather only in speed.

4.2. Profile Faces

4.2.1 Detectors for each profile view

We trained a right profile detector using 2868 manually cropped 24×24 pixel profile faces and over 100 million background (non-profile) patches. All profile faces were derotated so that the faces were looking approximately straight right. Artificial rotations of these faces of ± 15 degrees and ± 30 degrees were then created to generate variations in in-plane rotations. This should make the resulting detector more robust to such rotations. In-plane rotations

Figure 5: Example detections for two stage rotated detector.

are fairly common in images of profile faces. Some example training data are shown in figure 6.

The right profile detector uses diagonal filters as well as two, three and four rectangle filters. The resulting cascade has 38 layers of classifiers with the first six classifiers having 2, 10, 20, 20, 25 and 35 features, respectively.

To create a left profile detector, we simply flipped all the features of the right profile detector.

The resulting profile detectors handle out-of-plane rotations from about 3/4 view to full profile. Since the frontal upright detector handles poses from about left 3/4 view to right 3/4 view, these three detectors combine to handle the fulll range of upright poses from left profile to right profile.

4.2.2 Pose estimator

We also trained a decision tree to classify an image window as left or right profile. We used the same positive training data used for training the right profile detector and flipped versions of each image for examples of left profile faces. The trained decision tree has 256 internal nodes and 9 levels. It achieves 95.4% accuracy on the training set.

Since there are only two classes in this case, the decision tree is only useful to test its effect on the detection rate and false positive rate and not to improve the detector's speed. We wanted to confirm that the ROC curves for the two stage detector and the try-both-profiles detector were close as in the non-upright face case.

4.2.3 Results on profile test set

We obtained a profile face test set from Schneiderman and Kanade at CMU. The test set contains 208 images and according to Schneiderman and Kanade it has 347 profile faces (faces between 3/4 view and full profile). We, however, counted 355 profile faces. This difference is minor and our results should still be comparable to their results reported in [10]. In this test we are only trying to detect profile faces although some of the images also contain frontal faces.

We examined 48,303,529 image windows over the entire profile test set.

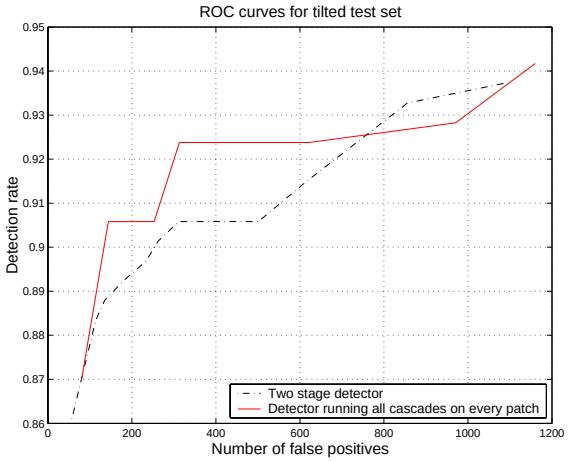

Figure 4: ROC curve showing the performance on the non-upright test set for both the two stage detector (which is much faster) and a detector that runs all 12 cascades on every image window.

The results of running the two-stage profile detector on some of the test images are shown in figure 8. A ROC curve is plotted in figure 7 showing the detection rate for various numbers of false positives. The figure shows the ROC curves for the two-stage detector using the decision tree as well as the try-both-profiles detector. As with the rotated face detector, the two systems are very close in accuracy with a slight edge going to the try-both-profiles detector. For the two-stage detector we detect 70.4% of the profile faces with 98 false positives and 83.1% with 700 false positives.

Schneiderman and Kanade [10] give a few points on the ROC curve for their profile detector. They get a detection rate of 92.8% with 700 false positives or 86.4% with 91 false positives or 78.6% with 12 false positives. Schneiderman and Kanade's results are better than ours but with a large penalty in speed. Our detector processes a 320×240 pixel image in about 0.12 seconds on a 2.8 GHz Pentium 4 machine.

5. Conclusions

We have demonstrated detectors for in-plane rotated faces and for profile faces. Together these detectors handle most face poses encountered in real images. This work confirms that the Viola-Jones framework can handle non-frontal, non-upright faces despite some previous doubts along these lines.

We have also presented a general method for selecting among a set of detectors while scanning an input image. A decision tree is learned which can select the appropriate detector to run. This method works well and preserves the speed advantage of the Viola-Jones detectors.

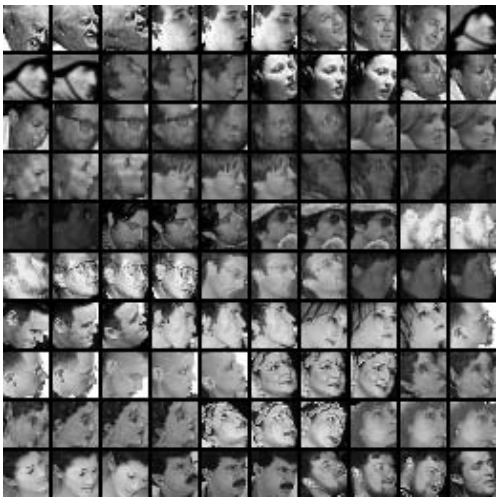

Figure 6: Some profile faces from the training set. They include artificially rotated versions of each face.

References

[1] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory: Eurocolt '95*, pages 23–37. Springer-Verlag, 1995.

[2] S. Li, L. Zhu, Z.Q. Zhang, A. Blake, H.J. Zhang, and H. Shum. Statistical learning of multi-view face detection. In *Proceedings of the 7th European Conference on Computer Vision*, Copenhagen, Denmark, May 2002.

[3] Edgar Osuna, Robert Freund, and Federico Girosi. Training support vector machines: an application to face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[4] C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *International Conference on Computer Vision*, 1998.

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

[6] D. Roth, M. Yang, and N. Ahuja. A snowbased face detector. In *Neural Information Processing 12*, 2000.

[7] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. In *IEEE Patt. Anal. Mach. Intell.*, volume 20, pages 22–38, 1998.

[8] H. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 38–44, 1998.

Figure 8: Example detections for two stage profile detector.

[9] R. Schapire and Y. Singer. Improving boosting algorithms using confidence-rated predictions, 1999.

[10] H. Schneiderman and T. Kanade. A statistical method for 3D object detection applied to faces and cars. In *International Conference on Computer Vision*, 2000.

[11] K. Sung and T. Poggio. Example-based learning for view-based face detection. In *IEEE Patt. Anal. Mach. Intell.*, volume 20, pages 39–51, 1998.

[12] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, Kauai, HI, December 2001.

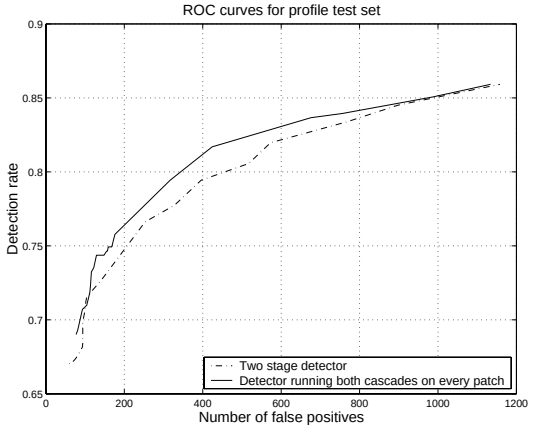

Figure 7: ROC curve showing the performance on the profile test set for both the two stage detector and a detector that runs both left and right profile cascades on every image window.