An Extended Set of Haar-like Features for Rapid Object Detection

Rainer Lienhart and Jochen Maydt
Intel Labs, Intel Corporation, Santa Clara, CA 95052, USA
Rainer.Lienhart@intel.com

ABSTRACT

Recently Viola et al. [5] have introduced a rapid object detection scheme based on a boosted cascade of simple features. In this paper we introduce a novel set of rotated haar-like features, which significantly enrich this basic set of simple haar-like features and which can also be calculated very efficiently. At a given hit rate our sample face detector shows off on average a 10% lower false alarm rate by means of using these additional rotated features. We also present a novel post optimization procedure for a given boosted cascade improving on average the false alarm rate further by 12.5%. Using both enhancements the number of false detections is only 24 at a hit rate of 82.3% on the CMU face set [7].

1 Introduction

Recently Viola et al. have proposed a multi-stage classification procedure that reduces the processing time substantially while achieving almost the same accuracy as compared to a much slower and more complex single stage classifier [5]. This paper extends their rapid object detection framework in two important ways: Firstly, their basic and over-complete set of haar-like feature is extended by an efficient set of 45° rotated features, which add additional domain-knowledge to the learning framework and which is otherwise hard to learn. These novel features can be computed rapidly at all scales in constant time. Secondly, we derive a new post-optimization procedure for a given boosted classifier that improves its performance significantly.

2 Feature Pool

The main purpose of using features instead of raw pixel values as the input to a learning algorithm is to reduce the in-class while increasing the out-of-class variability compared to the raw data and thus making classification easier. Features usually encode knowledge about the domain, which is difficult to learn from the raw and finite set of input data. A very large and general pool of simple haar-like features combined with feature selection therefore can increase the capacity of the learning algorithm.

The speed of feature evaluation is also a very important aspect since almost all object detection algorithms slide a fixed-size window at all scales over the input image. As we will see, our features can be computed at any position and any scale in the same constant time. Only 8 table lookups are needed.

2.1 Feature Family

Our feature pool was inspired by the over-complete haar-like features used by Papageorgiou *et al.* in [4,3] and their very fast computation scheme proposed by Viola *et al.* in [5], and is a generalization of their work.

Let us assume that the basic unit for testing for the presence of an object is a window of $W \times H$ pixels. Also assume that we have a very fast way of computing the sum of pixels of any upright and 45° rotated rectangle inside the window. A rectangle is specified by the tuple $r = (x, y, w, h, \alpha)$ with $0 \le x, x + w \le W$, $0 \le y, y + h \le H$, $x, y \ge 0$, $w, h > 0$, and $\alpha \in \{0°, 45°\}$ and its pixel sum is denoted by $RecSum(r)$. Two examples of such rectangles are given in Figure 1.

Our raw feature set is then the set of all possible features of the form

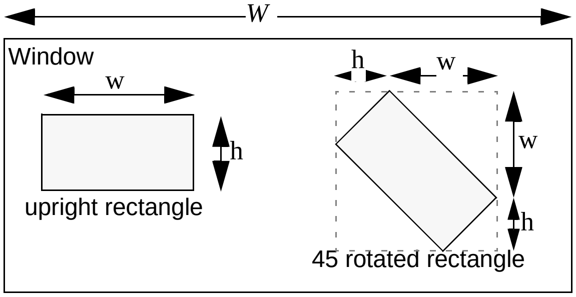

Figure 1. Examples of an upright and 45° rotated rectangle.

$$feature_I = \sum_{i \in I = \{1, \dots, N\}} \omega_i \cdot RecSum(r_i) ,$$

where the weights $\omega_i \in \Re$, the rectangles r_i, and N are arbitrarily chosen.

This raw feature set is (almost) infinitely large. For practical reasons, it is reduced as follows:

1. Only weighted combinations of pixel sums of two rectangles are considered (i.e., $N = 2$).
2. The weights have opposite signs, and are used to compensate for the difference in area size between the two rectangles. Thus, for non-overlapping rectangles we have $-w_0 \cdot Area(r_0) = w_1 \cdot Area(r_1)$. Without restrictions we can set $w_0 = -1$ and get $w_1 = Area(r_0) / Area(r_1)$.
3. The features mimic haar-like features and early features of the human visual pathway such as center surround and directional responses.

These restrictions lead us to the 14 feature prototypes shown in Figure 2:

- Four edge features,
- Eight line features, and
- Two center-surround features.

These prototypes are scaled independently in vertical and horizontal direction in order to generate a rich, over complete set of features. Note that the line features can be calculated by two rectangles only. Hereto it is assumed that the first rectangle r_0 encompasses the black and white rectangle and the second rectangle r_1 represents the black area. For instance, line feature (2a) with total height of 2 and width of 6 at the top left corner (5,3) can be written as

$$feature_I = -1 \cdot RecSum(5,3,6,2,0°) + 3 \cdot RecSum(7,3,2,2,0°) .$$

Only features (1a), (1b), (2a), (2c) and (4a) of Figure 2 have been used by [3,4,5]. In our experiments the additional features significantly enhanced the expressional power of the learning system and consequently improved the performance of the object detection system. Feature (4a) was not used since it is well approximated by feature (2g) and (2e).

NUMBER OF FEATURES. The number of features derived from each prototype is quite large and differs from prototype to prototype and

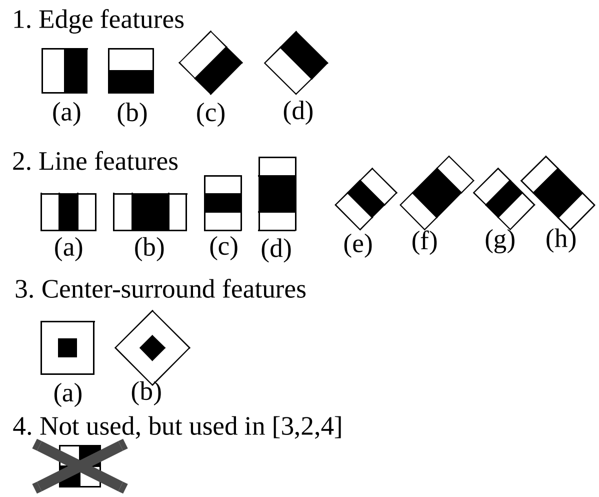

Figure 2. Feature prototypes of simple haar-like and center-surround features. Black areas have negative and white areas positive weights.

can be calculated as follows. Let $X=\lfloor W/w \rfloor$ and $Y=\lfloor H/h \rfloor$ be the maximum scaling factors in x and y direction. A upright feature of size wxh then generates

$$XY \cdot \left(W+1-w\frac{X+1}{2}\right) \cdot \left(H+1-h\frac{Y+1}{2}\right)$$

features for an image of size WxH, while a 45° rotated feature generates

$$XY \cdot \left(W+1-z\frac{X+1}{2}\right) \cdot \left(H+1-z\frac{Y+1}{2}\right) \text{ with } z=w+h.$$

Table 1 lists the number of features for a window size of 24x24.

Feature Type	w/h	X/Y	#
1a ; 1b	2/1 ; 1/2	12/24 ; 24/12	43,200
1c ; 1d	2/1 ; 1/2	8/8	8,464
2a ; 2c	3/1 ; 1/3	8/24 ; 24/8	27,600
2b ; 2d	4/1 ; 1/4	6/24 ; 24/6	20,736
2e ; 2g	3/1 ; 1/3	6/6	4,356
2f ; 2h	4/1 ; 1/4	4/4	3,600
3a	3/3	8/8	8,464
3b	3/3	3/3	1,521
Sum			117,941

Table 1: Number of features inside of a 24x24 window for each prototype.

2.2 Fast Feature Computation

All our features can be computed very fast and in constant time for any size by means of two auxiliary images. For upright rectangles the auxiliary image is the *Summed Area Table* $SAT(x,y)$. $SAT(x,y)$ is defined as the sum of the pixels of the upright rectangle ranging from the top left corner at (0,0) to the bottom right corner at (x,y) (see Figure 3a) [5]:

$$SAT(x,y) = \sum_{x' \le x, y' \le y} I(x',y') .$$

It can be calculated with one pass over all pixels from left to right and top to bottom by means of

$$SAT(x,y) = SAT(x,y-1) + SAT(x-1,y) + I(x,y) - SAT(x-1,y-1)$$

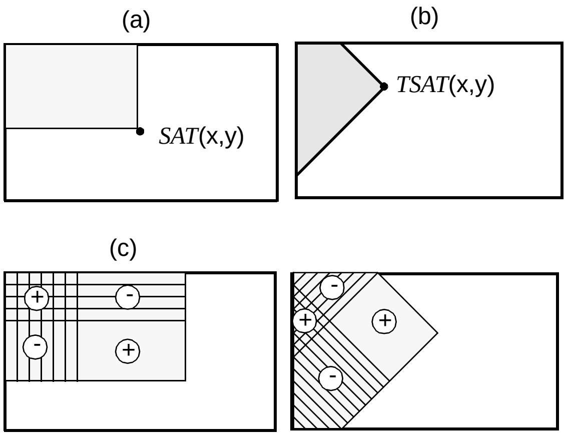

Figure 3. (a) Upright *Summed Area Table* (*SAT*) and (b) *Rotated Summed Area Table* (*RSAT*); calculation scheme of the pixel sum of upright (c) and rotated (d) rectangles.

with

$$SAT(-1,y) = SAT(x,-1) = 0$$

From this the pixel sum of any upright rectangle $r=(x,y,w,h,0)$ can be determined by four table lookups (see also Figure 3(c):

$$\begin{aligned} RecSum(r) &= SAT(x-1,y-1) + SAT(x+w-1,y+h-1) \\ &\quad - SAT(x-1,y+h-1) - SAT(x+w-1,y-1) \end{aligned}$$

This insight was first published in [5].

For 45° rotated rectangles the auxiliary image is defined as the *Rotated Summed Area Table* $RSAT(x,y)$. It gives the sum of the pixels of the rectangle rotated by 45° with the right most corner at (x,y) and extending till the boundaries of the image (see Figure 3b):

$$RSAT(x,y) = \sum_{x' \le x, x' \le x - |y-y'|} I(x',y') .$$

It can be calculated with two passes over all pixels. The first pass from left to right and top to bottom determines

$$RSAT(x,y) = RSAT(x-1,y-1) + RSAT(x-1,y) + I(x,y) - RSAT(x-2,y-1)$$

with

$$RSAT(-1,y) = RSAT(-2,y) = RSAT(x,-1) = 0 ,$$

whereas the second pass from the right to left and bottom to top calculates

$$RSAT(x,y) = RSAT(x,y) + RSAT(x-1,y+1) - RSAT(x-2,y)$$

From this the pixel sum of any rotated rectangle $r=(x,y,w,h,45°)$ can be determined by four table lookups (see also Figure 3(d) and Figure 4):

$$\begin{aligned} RecSum(r) &= RSAT(x+w,y+w) + RSAT(x-h,y+h) \\ &\quad - RSAT(x,y) - RSAT(x+w-h,y+w+h) \end{aligned} .$$

2.3 Fast Lighting Correction

The special properties of the haar-like features also enable fast contrast stretching of the form

$$\bar{I}(x,y) = \frac{I(x,y)-\mu}{c\sigma} ,\ c \in R^{+} .$$

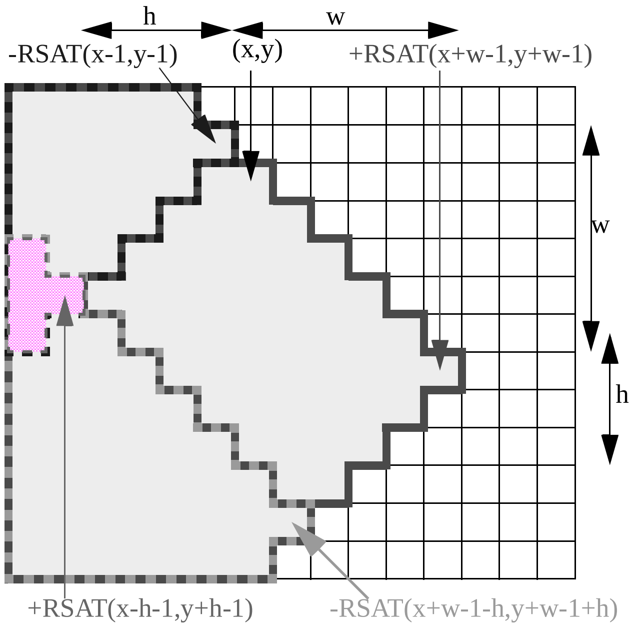

Figure 4. Calculation scheme for rotated areas.

μ can easily be determined by means of *SAT*(x,y). Computing σ, however, involves the sum of squared pixels. It can easily be derived by calculating a second set of *SAT* and *RSAT* auxiliary images for $I^2(x,y)$. Then, calculating σ for any window requires only 4 additional table lookups. In our experiments *c* was set to 2.

3 Cascade of Classifiers

A cascade of classifiers is degenerated decision tree where at each stage a classifier is trained to detect almost all objects of interest (frontal faces in our example) while rejecting a certain fraction of the non-object patterns [5] (see Figure 5). For instance, in our case each stage was trained to eliminated 50% of the non-face patterns while falsely eliminating only 0.2% of the frontal face patterns; 13 stages were trained. In the optimal case, we can expect a false alarm rate about $0.5^{13} \approx 1.2e-04$ and a hit rate about $0.998^{13} \approx 0.97$.

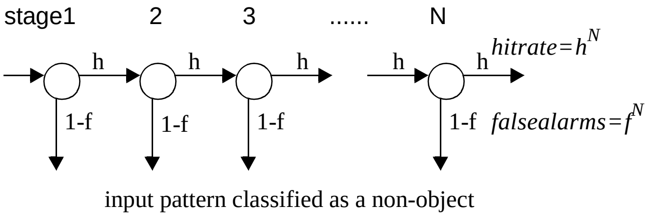

Figure 5. Cascade of Classifiers with *N* stages. At each stage a classifier is trained to achieve a hit rate off *h* and a false alarm rate of *f*.

Each stage was trained using the Discrete Adaboost algorithm [1]. Discrete Adaboost is a powerful machine learning algorithm It can learn a strong classifier based on a (large) set of weak classifiers by re-weighting the training samples. Weak classifiers are only required to be slightly better than chance. Our set of weak classifiers are all classifiers which use one feature from our feature pool in combination with a simple binary thresholding decision. At each round of boosting, the feature-based classifier is added that best classifies the weighted training samples. With increasing stage number the number of weak classifiers, which are needed to achieve the desired false alarm rate at the given hit rate, increases (for more detail see [5]).

4 Stage Post-Optimization

Given a discrete AdaBoost stage classifier

$$c(x) = sign\left(\sum_{m=1}^{M_c} \alpha_m \cdot f_m(x|t_m) + b\right) \text{ with } b = 0$$

we can easily construct an non-optimal ROC by smoothly varying offset *b* (see Figure 6). While this stage classifiers is designed to yield a low error rate (= miss + false alarms) on the training data, it in general performs unfavorable for *b*!=0, specially in our case where we want to achieve the miss rate close to zero.

However, any given stage classifier can be post-optimized for a given hit rate. The free parameters are t_i, while α_i must be chosen according to the AdaBoost loss function to preserve the properties of AdaBoost. We use the iterative procedure shown in Figure 7 for optimization, where step 4.2.1. is implemented in a gradient decent-like manner: Starting with the original t_i value, t_i is first slowly increased then decreased as long as the performance does not degrade. A true gradient decent cannot be implemented since the c(x) is not continuos. Note that any change in the threshold t_n requires recomputation of α_j, $w_{j+1,i}$ for $j \geq n$.

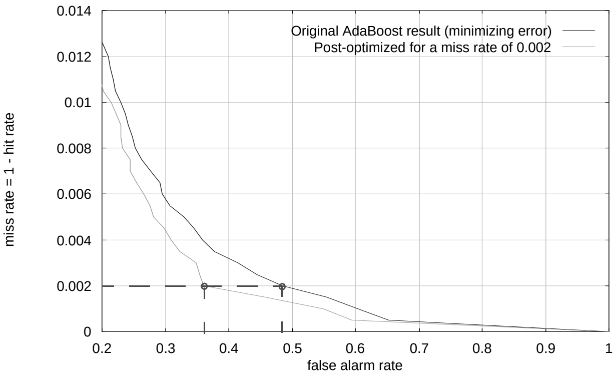

Figure 6. Comparison of the ROCs of a discrete Adaboost classifier with 11 features at stage 0 without and with stage post-optimization.

5 Experimental Results

5.1 Basic vs. Extended Haar-like Features

Two face detection systems were trained: One with the basic and one with the extended haar-like feature set. On average the false alarm rate was about 10% lower for the extended haar-like feature set at comparable hit rates. Figure 7 shows the ROC for both classifiers using 12 stages. At the same time the computational complexity was comparable. The average number of features evaluation per patch was about 31.

These results suggest that although the larger haar-like feature set usually complicates learning, it was more than paid of by the added domain knowledge. In principle, the center surround feature would have been sufficient to approximate all other features, however, it is in general hard for any machine learning algorithm to learn joint behavior in a reliable way.

5.2 Stage Post-Optimization

A third face detection system was trained using the extended feature

```
1. Define
       $F_M(x) = \sum_{m=1}^{M} \alpha_m \cdot f_m(x;t_m)$   and   $F_M^j(x) = \sum_{m=1, m \neq j}^{M} \alpha_m \cdot f_m(x;t_m)$
2. Given
   2.1. Positive and negative examples $(x_1^p, y_1^p), \ldots, (x_{N_p}^p, y_{N_p}^p)$ and
        $(x_1^n, y_1^n), \ldots, (x_{N_n}^n, y_{N_n}^n)$ where $y_i^p = 1$ and $y_i^n = -1$
   2.2. Stage classifier $c(x) = sign(F_M + b)$
   2.3. Desired target hit rate $h$
3. Initialize
   3.1. $err \leftarrow E_w^n[1_{y_i^n \neq c(x_i^n)}]$ with $b$ subject to $E_w^p[1_{y_i^p = c(x_i^p)}]/E_w^p[1] \geq h$
   3.2. $errOld = err + 1$
4. While ($err < errOld$)
   4.1. $errOld = err$
   4.2. Repeat for $j$=1, 2, ..., $M$:
        4.2.1 Find combination $\{t_j, b\}$ that minimizes the expected
              weighted false alarm rate at target hit rate $h$:
                  $\{\tilde{t}_j, \tilde{b}_j\} \leftarrow \arg\min_{t_j, b}(err^n(j, t_j, b))$
              with
                  $err^n(j, t_j, b) = E_w^n[1_{y_i^n \neq sign(F_M^j + \alpha_j \cdot f_j(x_i^n; t_j) + b)}]/E_w^n[1]$
              subject to
                  $E_w^p\left([1_{y_i^p = sign(F_M^j + \alpha_j \cdot f_j(x_i^p; t_j) + b_j)}]/E_w^p[1] \geq h\right)$
              and
                  $\alpha_j$ , $w_{j,i}$
              set according to Adaboost rule. The superscript $p$ and
              $n$ denotes that the expectation value and error is calculated with respect to the weighted positive and negative samples only.
   4.3. Determine $\{\tilde{t}_j, \tilde{b}_j\}$ combination with smallest expected
        weighted false alarm rate at given hit rate:
            $\tilde{j} \leftarrow \arg\min_j(err^n(j, \tilde{t}_j, \tilde{b}_j))$
   4.4. $t_{\tilde{j}} \leftarrow \tilde{t}_{\tilde{j}}, b_{\tilde{j}} \leftarrow \tilde{b}_{\tilde{j}}$ , update $\alpha_j$ and $w_{t,j}$ according to Adaboost
        rule, $err \leftarrow err^n(\tilde{j}, \tilde{t}_{\tilde{j}}, \tilde{b}_{\tilde{j}})$
```

Figure 7. Post-optimization procedure of a given boosted classifier for a given target hit rate.

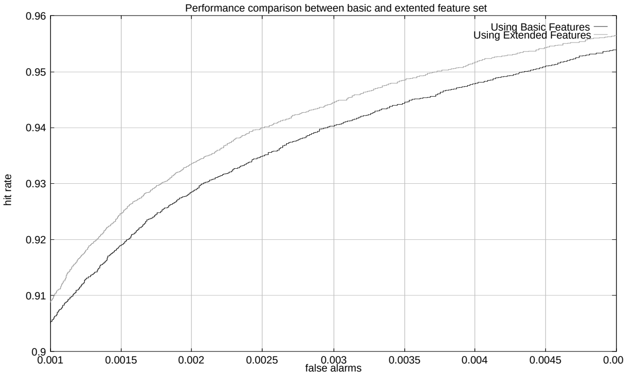

Figure 7. Basic versus extended feature set: On average the false alarm rate of the face detector exploiting the extended feature set was about 10% better at the hit rate.

set as well as our novel post-optimization procedure for each completed stage classifier. On average the false alarm rate was about 12.5% lower for the post-optimized classifier at comparable hit rates. Figure 8 shows the ROC for both classifiers[1]. At the same time the computational complexity was also comparable. The average number of features evaluation per patch was about 28.

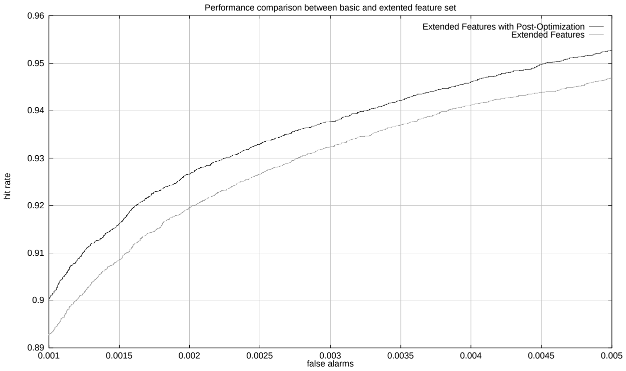

Figure 8. Stage post-optimization improves performance of the boosted detection cascade by about 12.5%.

Frontal faces are detected in CIF images (320x240) at 5fps on a Pentium®-4 2Ghz while searching at all scales with a rescaling factor of 1.2 using a pure C++-based implementation.

6 Conclusion

The paper introduced an novel and fast to compute set of rotated haar-like features as well as a novel post-optimization procedure for boosted classifiers. It was shown that the overall performance could be improved by about 23.8% of which 10% could be constributed to the rotated features and 12.5% to the stage post-optimization scheme.

7 REFERENCES

[1] Freund, Y. & Schapire, R. E. (1996b). Experiments with a new boosting algorithm. In Machine Learning: Proceedings of the Thirteenth International Conference, Morgan Kauman, San Francisco, pp. 148-156, 1996.

[2] Chulhee Lee and David A. Landgrebe. Fast Likelihood Classification. IEEE Transactions on Geoscience and Remote Sensing, Vol. 29, No. 4, July 1991.

[3] A. Mohan, C. Papageorgiou, T. Poggio. Example-based object detection in images by components. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 23, No. 4, pp. 349 -361, April 2001.

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

[5] Paul Viola and Michael J. Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE CVPR, 2001.

[6] P. Pudil, J. Novovicova, S. Blaha, and J. Kittler. Multistage pattern recognition with reject option. 11th IAPR International Conference on Pattern Recognition, Vol.2, pp. 92 -95, 1992.

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

[1] In Figure 8 only 9 stages were used. In the final paper, the data for a 13 stage classifier will be given.