TEACHING IMPROVEMENTS ON HAAR BASED CLASSIFIERES

Szidónia LEFKOVITS

Petru Maior University Târgu-Mureş, Department of Mathematics and Computer Science
e-mail: szidonia.lefkovits@science.upm.ro

Abstract: Nowadays an increasing number of applications require fast and reliable object detection systems. The most efficient system presented in the publications of Viola-Jones [4,5,6] object detection framework and the open source implementation of their ideas creates a solid baseline for future detectors. This approach has been extensively used in Computer Vision research, particularly for detecting faces and facial features. The OpenCV community shares a collection of such classifiers. The analyses of such public classifiers define the basis of future work in the object detection domain. In this paper, the performance of cascade classifiers is analyzed. A series of ambiguities concerning the teaching process is also presented together with a few proposals how to solve them. It has been tried to discover and overtake the limitations of the OpenCV implementation and expanded it to set up the author's own classifier. Finally, an original algorithm is proposed to get 10^{-5} false alarm rate.

Keywords: face detection, AdaBoost, Haar features, training data set, supervised learning

1. Introduction

Face detection and recognition has become an increasingly researched area. The Viola and Jones method for face detection [3, 4, 5, 6] is an especially successful method, as it has a very low false positive rate. It can detect faces in real time and yet is very flexible in the sense that it can be trained for different level of computational complexity, speed and detection rate suitable for specific applications. The implementation offered by Intel in the OpenCV application made this algorithm more attractive. It is highly desirable to use this versatile method for anyone who might want to make research in this area. The OpenCV application has a poor tutorial in reference to the creation of classifiers. To exceed this, a lot of authors published their own experience on the internet [9, 10]. One can find a lot of comments, experiences and useful functionalities in order to create the training data set.

2. The facial detection system

The used detection system is a combination of geometrically-based and image-based methods. It is geometrical, because it uses general features of human faces: position of particular features among which the eyes, the nose and the mouth. The image properties are characterized by Haar functions (Section 2.2). The built face model is generated out of a training data set by a statistical learning algorithm, AdaBoost, which combines the most suitable selected Haar-functions.

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

2.1. The AdaBoost Algorithm

The AdaBoost algorithm was proposed by Freund and Shapire [1] as a training algorithm. It constructs an ensemble of classifiers and uses a voting mechanism for the classification. In a wide variety of classification problems, their weighting scheme and final classifier merge have proven to be an efficient method for reducing bias and variance, and improving misclassification rates.

The idea of boosting is to use the weak classifier to form a highly accurate prediction rule by calling the weak classifier repeatedly on different distributions over the training examples. Initially, all the weights of the training images are set equally, but in each iteration the weights of incorrectly classified examples are increased so that the images, which were poorly predicted by the previous classifier, will receive greater weight on the next iteration. Thus the weight represents the importance of the image in the learning process.

The most important theoretical propriety of AdaBoost concerns in its ability to reduce the training error. The AdaBoost converts a set of weak classifiers into a strong learning algorithm, which can generate an arbitrarily low error rate.

2.2. Haar functions

Many descriptive features could be used to train a classifier by boosting. In face detection, the feature based method seems to be quite efficient. The Haar wavelets are naturally set basis functions, which compute the difference of intensity in neighboring regions [5]. The value of the Haar function is the measure of likelihood between a specified subregion of an image and the graphical representation having the same size of the Haar function.

Significant is the very fast evaluation of this function by using a new image representation called Integral Image. Another important property is the fact that the value of a Haar function is the same if the picture is reduced by a factor, or the Haar function is increased by the same factor. This property decreases more the evaluation time.

A corresponding weak classifier can be built from each Haar function. For this, we need to determine the optimal threshold, which delimits in the best way the set of face and non-face images. Such a threshold is determined for each Haar function. This optimum is reached when the number of misclassified examples is the lowest weak classifier $h_j(x)$ consists of a feature $f_j(x)$, of a threshold value θj and a parity p_j to indicate the direction of inequality:

$$h_j(x) = \begin{cases} 1 \ ,if & p_j \cdot f_j(x) > p_j \cdot \theta_j \quad x - face - image \\ 0 \ ,if & p_j \cdot f_j(x) < p_j \cdot \theta_j \ x - non - face - image \end{cases} \tag{1}$$

2.2. The monolithic classifier

The monolithic classifier for face detection is built by using the AdaBoost algorithm and the weak classifier based on Haar functions [5]. We also need a set of training examples consisting of all significant human faces (5,000) and various non face images (10,000) for the learning process. The first step is to build the classifier-image table, which contains the binary decision value of each classifier for every image of the

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

training set. The table is quite large, and we have to use each value of it to compute the weighted error in every iteration and for each weak classifier. The minimum error determines the selected weak classifier for one iteration and the weight for it in the final classifier. Thus, this error modifies also the weight of each picture. While the weight of the misclassified pictures increases, the weight of the well-classified ones decreases. The algorithm cycle stops when one of the learning conditions is satisfied; these conditions can be:

- the maximum number of T epochs.
- the error condition for the weak classifier
- the desired performances are reached.

2.3. The cascade classifier

The computation time of a monolithic classifier is the same for every image. In order to reduce this time, it is necessary to evaluate one part of the images with few weak classifiers and evaluate only complex images with the whole classifier. The idea is to reject rapidly the utmost part of negative images. Instead of one stage a cascade classifier built of multiple cascaded stages was proposed. The cascade design process is driven by a set of detection performances [4]. If each stage classifier is taught for low performances (f<0.5, false detection rate/stage and d>0.999, hit rate/stage), then the whole cascade will have the same performances as a monolithic classifier, but 10 times faster.

Each stage is taught with the remaining images from previous stage. The stop condition of the learning process is given by the reached performance. For this reason it is necessary to measure this performance with a validation set of images.

If we build a classifier with 20 stages, each with the above performances, then we will obtain both the global false positive error rate F $F < f^{20} = 0.5^{20} = 9.6 \cdot 10^{-7}$, and the error detection rate D $D \geq (1-d)^{20} = (1-0.001)^{20} = 0.98$ (Figure. 1).

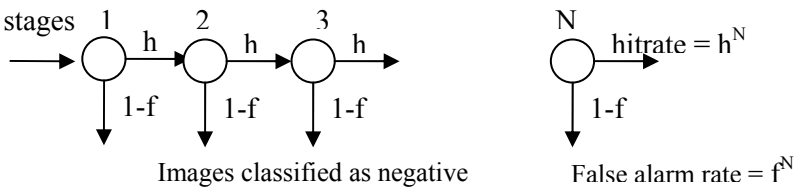


Figure 1. Cascade classifier with N stages [3]

To obtain the given performances, the global threshold has to be modified until the current cascade has a detection rate of at least d_k. This task is reachable by decreasing the threshold; then the detection rate increases, and the false detection rate increases, too. If this parameter is too big and does not fit the learning conditions, the algorithm takes another weak classifier in the current stage.

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

The overall training process involves two types of tradeoffs. In most cases, classifiers with more features will achieve higher detection rates and lower false positive rates. At the same time classifiers with more features require more time to compute.

3. Building a classifier

This section presents the different results obtained by the face detectors that have been developed. A few results of various authors will be discussed and interpreted and then the conclusions drawn regarding our experimental results.

Only a few classifiers can be found which work with this principle of boosting [7] in the public domain. The OpenCV community shares their collection of classifiers (see Table. 1). Only a few authors, disclosed their classifiers, but none of them published their training sets and training methodology. In this domain there are several unsuccessful attempts to train classifiers with the presented algorithm.

3.1 Building the training set

We propose to create a classifier for face detection with the cvHaarTraining program. It seems to be easy enough to follow the procedures described in OpenCV tutorial. In fact, there are a lot of problems to be solved in order to obtain an efficient classifier.

There are a lot of basic questions with the training set.

1. What is the best input pattern size?
2. How to crop the face images?
3. What is the background image size?
4. Which are the significant images?
5. What is the necessary training set size?
6. How to train the classifier to obtain 5×10^{-6} false alarm rate?

These questions are further debated.

1. The input size of the image determines the number of used features in the learning process. For a pattern of 24×24 pixels size, there are 84848 (BASE) features in the basic set and 111360 (CORE) in the extended set, and 138694(ALL) in the entire set features to evaluate.

Larger images are more detailed and need more memory and more features to evaluate. That means a larger feature-image table. Experimental analysis can conclude the size of image pattern depending on each application. According to the experiments [5], the images' pattern size 24×24 is the best in face detection, because it has the lowest false alarm rate at the same hit rate.

According to our experimental results, the optimal pattern size is 18×24. We concluded that the optimal pattern size depends on the variety of the data base used for training. Other approaches of face detectors have obtained other optimal dimensions for the training image pattern.

2. There are a lot of possibilities to crop face images.
   a. cropping only the significant part of the images

We can define the lower and upper boundary of the face by adding the distance between the mouth and the nose to the height of the eye-line, and subtract from the height of mouth-line the same distance. We

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

define the left-right margins by adding the distance between the eyes to the right margin of the right eye and subtract it from the left margin of the left eye [9].

b. cropping images that include extra visual information, such as contours of the chin and cheeks and the hair line. It seems that additional information in larger sub-windows can be used to reject non-faces earlier in the detection cascades [5]. The second case included additional information of the faces. The evaluable features are also more numerous, which make the detection process more accurate.

3. The size of the background image does not seem to be so important, it is never explicitly specified. It can be taken to be the same size as the positive pattern size, namely 24×24 pixels.

The OpenCV HaarTraining program can read background images of any size and it crops from this various number of backgrounds by shifting the cropping window through the whole image by a step of width/2 and height/2. It takes the specified number of backgrounds generated from the same given images each time. Smaller images with different characteristics are recommended to crop as backgrounds. There are two possibilities to create your own classifier with OpenCV. One way is to create the whole classifier containing more stages at once. Probably, in this case the background images are filtered in each step and only the false alarm images are used in the further stages. The other way is to create each stage separately. One drawback of the OpenCV training process is the building of the background set. In this case, one should choose backgrounds randomly, in an other way than OpenCV, which uses the same image table for all the stages created separately. In their application, the background training set contains the same aggregation of images in the same order for each stage. This is the reason we propose to use different backgrounds in each stage.

4. The image-based learning method needs a number of significant positive and negative images. At this step, one should have a methodology to choose only the significant images. Practice proves that an amount of 5,000 face images would be enough, but the difficult question is the number of backgrounds. The positive images are usually cropped manually and each is verified by a human operator. To increase the insensibility of the built classifier, many images can be generated from one image by applying distortion functions (randomly little rotation, translation and resizing). This little variation can be applied to the whole image set in order to multiply the number of used images. The images of our own database were collected from public lebeled face databases FERET and Yale (about 1,800 pictures), and these were completed by a self marked cropped studio images (about 1,100).

The background images are generated automatically, in general randomly, from a set of images. The backgrounds were downloaded randomly from the internet, and besides we used the Corel Draw image set. To increase their variety and cardinality, several random operations were performed: rotation, translation, resizing. Because of the large variety of backgrounds the selection of significant patterns is a difficult task. One idea is to take the images filtered by the existing stages of the classifier as significant background pattern.

5. The training set size determines the learning time. Leinhart proposed a training set with 5,000 positive and 3,000 negative images [3]. Viola and Jones built their classifier with 4,916 faces and 10,000 non-faces selected randomly from a set of 9,500 images which did not contain faces [5].

Our first experience had 3,000 positive images and 27,000 negative ones. The scanning Windows sizes were 18x18, which contained 33,000 Haar features from the basic set. The dimension of feature-image table is 10^9, and used 1GB memory. The learning process reported 30s for the selection of one Haar feature. The

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

computer we used was an Intel Core 2 Duo, CPU E460 frequency 2.4GHz, Gigabyte motherboard FSB 1333MHz, Dual Channel DDR2 800 and 2GB of RAM. It needs about 30,000 seconds, more than 8 hours, in the selection of 1,000 features.

The functionality principle of the detector is to scan the image at multiple scales and locations. Good results are obtained by using a scale factor of 1.3 and a step size of s=1 pixel. With this scanning parameters, one image of 320×240 pixels size has 130,000 sub-windows of 24×24 pixels. The processing of the high number of sub-windows, suppose a false alarm rate lower than 10^{-5}.

6. The question is how to learn the classifier in order to obtain a 5×10^{-6} false alarm rate. In order to achieve this performance, millions of different background pictures are needed. [6]

If we use 2 million background pictures, the processing time of one feature selection increases dramatically. The feature-image table needs a huge amount of memory space which exceeds the usable RAM memories. This limitation can be solved by the usage of virtual memory created on HDD. Access time to virtual memory increases the computation time. It will become 100 times longer, so it will take at least one month to build a classifier containing 1,000 features.

The solution is probably behind a methodology of choosing the background images.

We propose to learn each stage of classifiers the same amount of 5,000 positive images and a number of 10,000 background images filtered by the previous stages. A simple program is needed, which randomly crops background patterns from a specified set of images. The background patterns are taken for each step from a different set of given images, thus the needed number of pictures will increase each time, because one needs 10,000 remaining images after the filtration by previous stages. Supposedly, this is a way to get more and more specialized stages. The process ends when we cannot get more significant images or the time of choosing background patterns increases over a given limit (Fig. 2).

The inner block cycle executes the selection of background images until the *nmax* cycle limit is reached, namely the desired number of non-face images. The outer block represents the training process with the face and previously chosen non-face images. This cycle ends if we achieve a given number of stages of the classifier or the set of backgrounds is not sufficient any more.

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

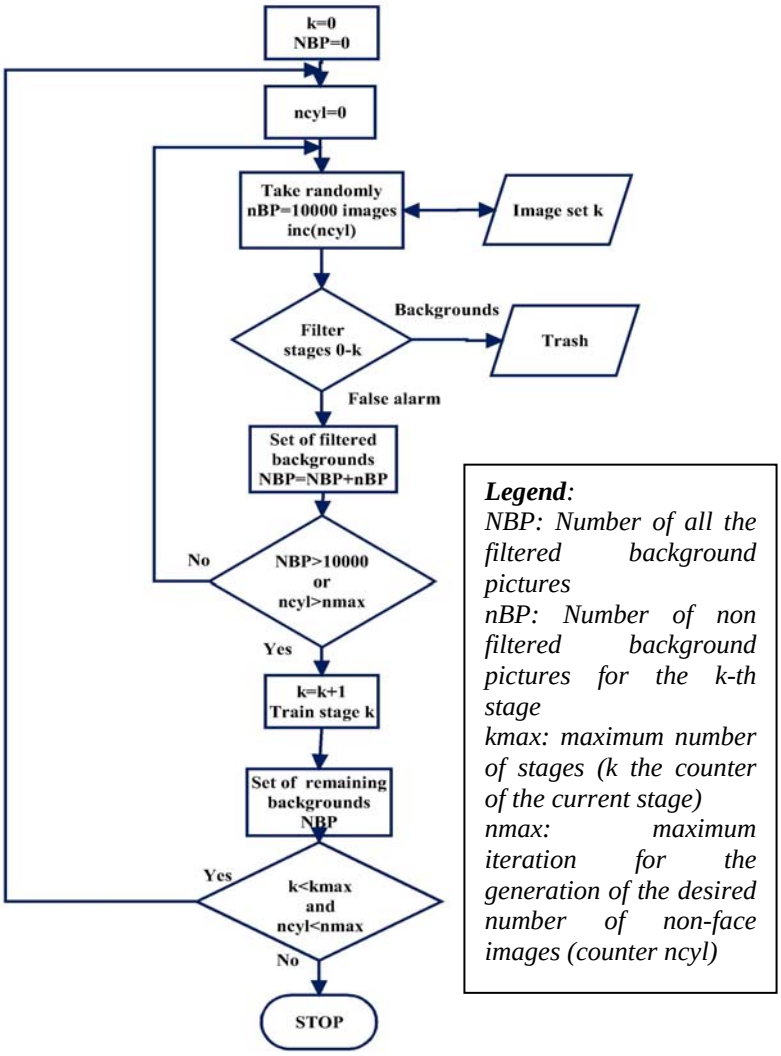


Figure. 2. Significant background generator

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

Algorithm:

1 Initialization: set the number of filtered background pictures NBP=0, counter of the current stage k=0
2 Repeat
2.1 Set number of generation cycle for the current stage (k) ncly=0
2.2 Repeat
2.2.1 Increment the counter of generation cycles
2.2.2 Take randomly 10,000 backgrounds
2.2.3 Filter backgrounds with previous stages 0..k
2.2.4 Add the current filtered images to the total number of filtered images
2.3 Until the desired number of filtered images are achieve or the generation cycle exceeds the set limit (nmax)
2.4 Increase stage counter
2.5 Train a new stage
3 Until the number of stages and maximum number of generation cycles are achieved

3.3 Performance analysis – Reported experiments

The classifier structure had to be deducted from the available public classifiers stored in xml files. Owing to this, a cascade classifier consists of several stages and a stage threshold. On one stage, the weak classifiers have to be cumulated according to the set of hit and false detection rate. Thus, on the first stages only few features have to be used with low error rate and on the further stages more, because the remainder less efficient classifiers should be combined.

According to Viola and Jones's writing [5]:

"Training time for entire 32 layers was on the order of weeks on a single 466MHz AlphaStation XP800. During this laborious training process several improvements to the learning algorithm were discovered. This improvements which will be described elsewhere, yield to 100 fold decrease in training time." These improvements were never published and remain the secret of the learning process.

The following can be concluded by analyzing the published classifiers: the number of used stages of a classifier varies between 16 and 46, the mean value is about 22-23 stages. The first stages contain 2 - 10 features, whereas the last stages contain 100-200 features. It can be asserted that a good classifier should have more than 1,000 features (see Table 1). The table contains the measured parameters of public classifiers FD (frontal default), FA1 (frontal alternate 1), FAT (frontal alternate tree), FA2 (frontal alternate 2) and the original proposed classifiers (Class_04, Class_05, Class_06).

BDD-A23672 © 2009 Editura Universităţii „Petru Maior"
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

	Detection	Missed	False	Stages	Features	DetTime(s)
FD	344	24	192	25	2913	16.77
FA1	337	31	106	22	2135	20.47
FAT	325	43	61	47	8468	18.99
FA2	344	24	143	20	1047	17.73
Class_04	293	75	666	6	677	17.53
Class_05	329	39	1190	12	1632	26.58
Class_06	279	90	150	16	1319	17.87

Table 1. Measured performances of classifiers

Most authors created their classifiers and published their results on face detection in tables containing the performance parameters, usually a representation of the detection rate and the amount of false detections. A table, that is, one measurement, represents only a single point on the ROC (Receiver Operating Characteristic) curve. In order to be able to compare the detectors, we need the ROC that represents the variation of the detection rate depending on the number of false alarms [7]. The ROC curves would be very helpful if the publishers appended the used database and the measure methodology. In their absence, we took some measurements on each stage, and based on these results we draw up the ROC curve for classifiers. There are two types of detectors: the first one has a good (90%) hit rate, which implies a huge amount of false detections. The second has much less false alarms, but the detection rate and in the mean the the detection time also decreases. The authors found their optimum between these contradictory requirements of the above described two types of classifiers. If each stage is taught separately in the learning process, then the threshold of each stage can be modified by the required performance values. Referring to this, we could observe the difference between theory and OpenCV implementation. OpenCV implementation only uses the training data set for testing performances. But the theoretical algorithm requires the result of the performance on the test data set and, accordingly, modifies the stage threshold. Depending on the result of the modified stage threshold, one decides to continue the learning process until the required parameters are reached.

In order to evaluate and compare detectors, we used the performance evaluation of OpenCV, which resulted in as follows:

- A table containing the tested pictures and the number of hits and false alarms. The last row is the sum of the results, out of which we can calculate the necessary percentage for the ROC.
- A second table contains the points of the ROC (not suitable for comparing classifiers).

The face detected regions are directly marked on tested images for visual empiric performance analysis.

One needs a test data set in order to test detector performance. Fortunately, this is available on the CMU's (Carnegie Mellon University) internet sites [11]. The description of data set does not correspond to the input image format request of OpenCV program; consequently, we modified it according to the face positions. This set contains 105 images with 368 faces. The performance results are given in Table 1.

Based on the fact that the detection rate of the next stage is greater or equal to the previous stage rate and only the false detection rate decreases (i.e. every stage corrects the number of false images), we propose to build our own stage by stage measured ROC curve for classifiers comparison (Table 2 and Figure 3). We consider that this curve is more suitable for the comparison of classifiers.

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

We can conclude from the mentioned table (Table 2), that our classifier presents a lower false detection rate, beginning form the earlier stages. This result is due to the proposed algorithm.

stage no.	10	11	12	13	14	15	16	17	18	19	20
hit rate	91,84	94,29	96,1	95,9	96,1	95,6	94,5	94	93,4	93,75	94,3
no.false det	4458	3412	2805	2157	1685	1199	832	596	346	219	143

stage no.	6	7	8	9	10	11	12	13	14	15	16
hit rate	85	89,4	89,9	90,4	88,85	87,8	85,32	83,96	80,16	75,8	75,5
no.false det	4661	3588	2646	2230	1722	1363	1188	846	390	168	150

Table. 2 Stage by stage measurements for FA2 and Class_05

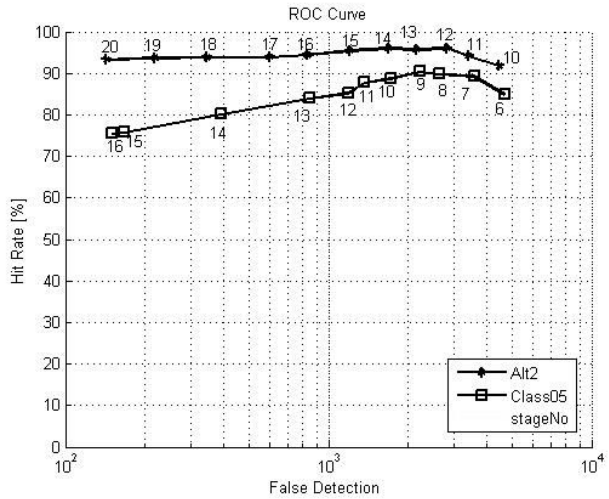


Figure. 3 Stage by stage ROC curve

Theoretically, the ROC has to be a decreasing curve. In the early stages the curve increases because of the numerous false alarms. So this modifies the center coordinate of the resulting detected object.

But it is evident that the detection rate is less than the value of it compared to the known best classifiers. The maximum hit rate (95%, respectively 90%) shows the properness of the face database used. This is due to the prepared pictures which do not contain sufficient various faces. They predominantly present young European people without beard or moustache, and very few of them wear glasses. Thus, without these lacks, the detection rate would be also over 90%, closer to the best known results.

In conclusion, today's known most efficient classifiers for frontal face detection are those of OpenCV source, which were trained by Leinhart, Kuranov and Pisarevsky [3].

4. Conclusion

BDD-A23672 © 2009 Editura Universităţii „Petru Maior”
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)

In several cases, under more natural conditions, our classifier presents fewer false alarms at the same detection rate than the known ones. It is necessary to enlarge the face database and, simultaneously, create an algorithm for detection of significant faces, in order to eliminate the most part of the previously supervised human-classification. The efficiency of the detector depends on the training data set and the used methodology, but this remains the secret of the authors. The building of the training data set is very laborious and drudgery. The inconveniences of the OpenCV program can be avoided by the ability and knowledge of the user.

References

[1] Y *Y. Freund and R.E Schapire. "A decision-theoretic generalization of on-line learning and an application to boosting", Journal of Computer and System Sciences, vol. 55, pp. 119-139, Art. No. SS971504, 1997*

[2] *C. Huang, B. Wu, H. Ai, S. Lao – "Omni-directional Face Detection Based on Real AdaBoost", Proceedings of International Conference on Image Processing, vol. 1, pp. 593-596, 2004*

[3] *R. Leinhart, A. Kuranov, V. Pisarevsky "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection", 25th DAGM Pattern Recognition Symposium, pp. 1-x1, 2003*

[4] *R. Leinhart, L.Liang, A. Kuranov "A Detector Tree of Boosted Classifiers for Real-time Object Detection and Tracking", Proceedings of International Conference on Multimedia and Expo (ICME'03), vol. 2, pp. 277-280, 2003*

[5] *P. Viola, M. Jones „Robust Real-time Object Detection", International Journal of Computer Vision, vol. 57, no. 2, pp. 137-154, 2004*

[6] *P. Viola, M. Jones „Fast Multi-view Face Detection" Mitsubishi Electric Research Laboratories, Cambridge, Technical Report TR2003-096, July 15, 2003*

[7] *M. Castrillón-Santana, O. Déniz-Suárez, L. Antón-Canalís, J. Lorenzo-Navarro, "Face and Facial Feature Detection Evaluation - Performance Evaluation of Public Domain Haar Detectors for Face and Facial Feature Detection", Proceedings of 3rd International Conference on Computer Vision Theory and Applications (VISAPP'2008), pp. 167-172, 2008.*

[8] *http://www.intel.com/research/mrl/research/opencv*

[9] *Naotoshi Seo "Tutorial: OpenCV haartraining" http://note.sonots.com/SciSoftware/haartraining.html*

[10] *Florian Adolf "How-to Built a Cascade of Boosted Classifiers Based on Haar-like Features" http://robotik.inflomatik.info/other/opencv/OpenCV_ObjectDetection_HowTo.pdf*

[11] *CMU/VACS image data base: http://www.cs.cmu.edu/~cil/v-images.html*

BDD-A23672 © 2009 Editura Universităţii „Petru Maior"
Provided by Diacronia.ro for IP 216.73.216.143 (2026-11-19 08:46:21 UTC)