Performance improvements of Haar Cascade Classifier for face detection.

Ludmila Natanzon, Algorithmic and SW Specialist,
(http://www.linkedin.com/profile/view?id=3597990)

Abstract

The focus of this paper is the improvement of the performance of the automatic detection of the human face/s on the image acquired by color photo or video camera. Algorithms for automatic feature detection are used more and more extensively in the fast developing field of mobile applications on different devices. Usually the standard video-camera application provides user with automatically detected face/s, and this detection implemented on hardware works effective enough to make an impression of instantaneous. The most commonly well-known approach for the problem of face detection is based on the multi-scale cascade filters of Haar type; the implementation of this algorithm called a Haar Cascade Classifier is available as a library function of OpenCV. Although, if one tries to use standard library functions of OpenCV for face detection in the applications running on devices with low processing power, the performance of this algorithm is far from being satisfactory. Therefore, the possible ways to improve the performance of Haar Cascade Classifier for face detection are considered in this paper. The innovative approach is suggested: to run the color analysis of the image prior to Haar Classifier in order to detect areas with skin-likely color and then subject only those areas, which also satisfy the restrictions of reasonable size and shape, to processing with Haar Cascade Classifier. Since the portion of such areas in the image is usually relatively small, this approach gives significant improvement of the performance of the face detection algorithm.

I. Introduction and prior art

Algorithms for feature detection and, as a special case, face detection are becoming widely required in various applications, from biometric authentication, surveillance to different interactive user interfaces, games, advertisement industry, entertainment services etc. The most natural task for a human brain, such as to localize face/s on the picture, which can be easily solved even by a 3-year old child, presents a real challenge for automatic solution, especially if one considers images of varying quality, size, resolution, various conditions of scene illumination, color intensity, shown person's pose, face 2D rotation on the plane of the image, expression, shadowing etc. Just notice, that all these varying factors will hardly confuse a human observer, but for a computer they can introduce insuperable obstacles.

Therefore, this practical and challenging problem caused a lot of research activities in the area. Early works were surveyed in [1], [2]. The basic work by Viola and Jones [4], [5] has made face detection practically feasible in real world applications such as digital cameras and photo organization software. Implementations of Viola-Jones framework, such as OpenCV and others, provide different face classifiers created by authors that used different datasets into their training. The performance and reliability of these classifiers vary a lot.

There was a lot of activity in the field of face detection in the past decade. In the report [3] the authors present a brief survey on the latest development in face detection techniques since the publication of [2], which shows that more attention was given to the improvements of the learning part of the Viola-Jones framework.

The use of color information has been introduced to the face-locating problem in the late 90-s [6], and it has gained increasing attention since then. One of two general approaches, described in [6], was to employ color as a feature for partitioning an image into a set of homogeneous regions: for instance, the color component of the image can be used in the region growing technique, as demonstrated in [7], or as a basis for a simple thresholding technique, as shown in [8]. The other approach makes use of color as a feature for identifying a specific object in an image: in this case, the skin color can be used to identify the human face. Some recent publications that have reported this study include [9]–[11].

II. Haar Cascade classifier for face detection and its performance

The description of the algorithm used in OpenCv implementation for face detection can be found in http://www.cognotics.com/

The implementation is based on a method that Paul Viola and Michael Jones published in 2001. Usually called simply the Viola-Jones method, or even just Viola-Jones, this approach to detecting objects in images combines four key concepts:

- Simple rectangular features, called Haar features
- An Integral Image for rapid feature detection
- The AdaBoost machine-learning method
- A cascaded classifier to combine many features efficiently

In order to detect the human face in the image the Haar Cascade Classifier (HCC) should be run on the sub region of the image defined by the rectangular frame of correct size and set in the correct location, as shown in Figure 3 on the left. To ensure that the required frame is tested one has to run HCC on the frames of all possible sizes and locations inside the image, which will result in the big amount of computations and long computation time. In practice, to reduce the amount of required computations the locations of the tested frames are incremented by more than one pixel, only square frames are tested, and the sizes of tested frames are also incremented by some reasonable factor and restricted by some limits. Although these and some additional performance improving steps like the applying of Integral Image, the amount of processing still presents a problem for running the implementations of the algorithm on devices with low CPU power. Since one path of the algorithm on the image of the standard 240x320 size on the device with 1GHz single core processor takes more than 1 sec, freezing the program makes it look non-responsive and almost impossible to use. Here, let us specify, that there are actually 2 different but related problems, originating from too long computation time of the single path of HCC on the whole image with the full set of parameters values: (1) freezing of the application during single path processing, if no special steps to run UI in a higher priority thread are taken; (2) too long time until the face(s) is (are) located. Further there will be suggested some ideas for solving both of the mentioned problems.

To improve the algorithm performance one can try to further reduce the number of tested frames. The first obvious way to do it is to run HCC on more sparse set of frames, increasing the distance between their locations and/or the factor between two consequent sizes.

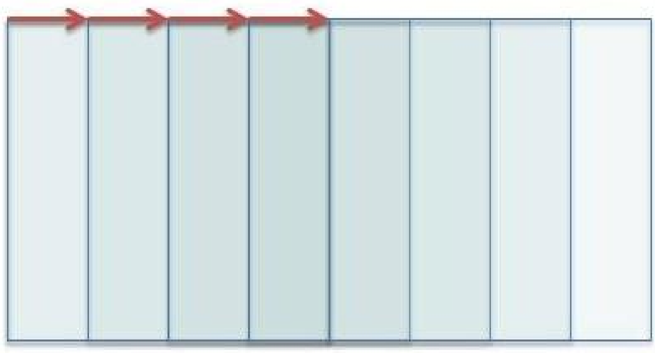

Figure 1. Frame shifted by ¼ of its size

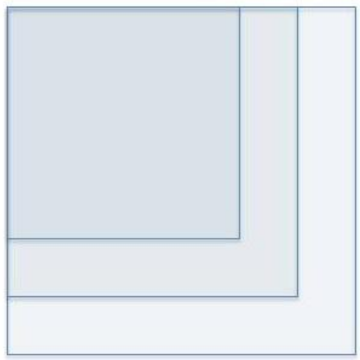

Figure 2. Frame size is increased by factor 1.2

However, if the frames are located on the image too sparse or their sizes change too fast, the face detection can fail because the HCC is sensitive to the correct position and the size of the tested frame, while the optimal frame location and size will be skipped, as shown on Figure 3. In order

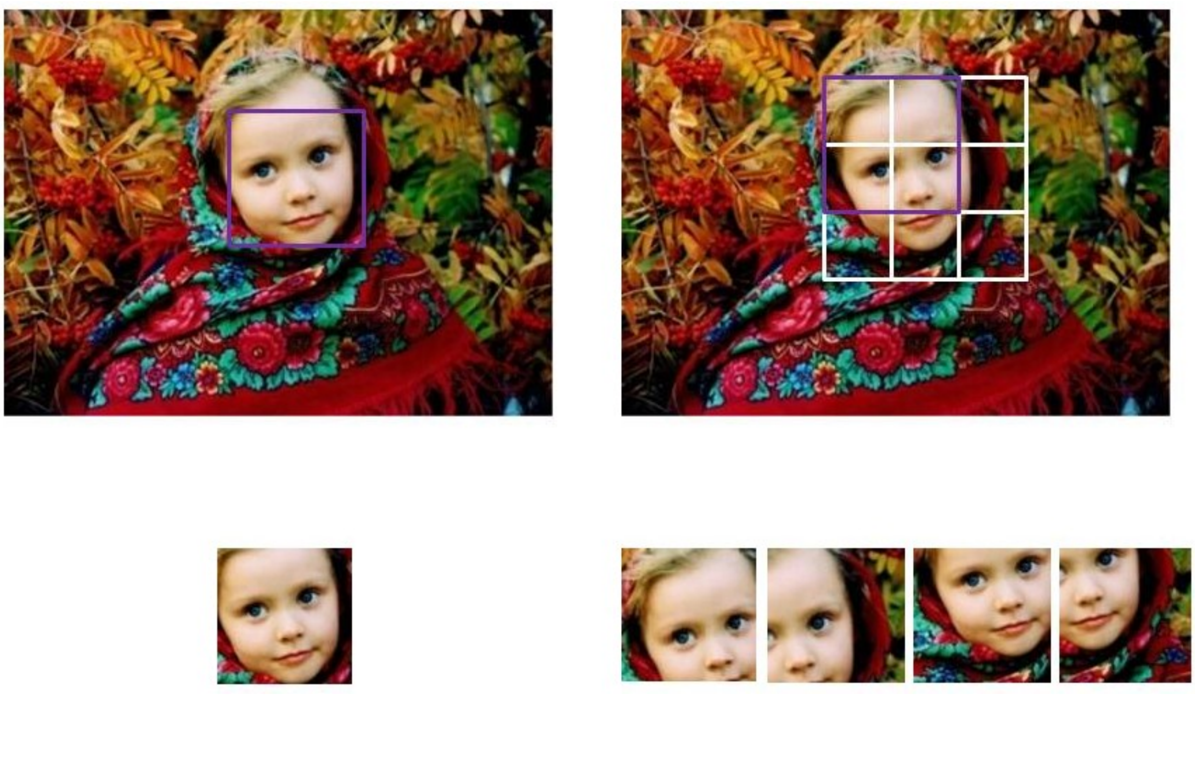

Figure 3. On the left: the frame of correct size and location, for which HCC will return positive result. On the right: too sparse locations of the tested frames, HCC on neither one of them will return positive result.

to find the best compromise between requirement to run the HCC at least once on the correct frame location and size and the tendency to reduce as possible the number of the tested frames, the following scheme is suggested: in the single path of the HCC on the image use sparse system of tested frames, but ensure that consequent runs will use the shifted frames so, that after few runs the examination of all needed locations and sizes will be provided. This approach will solve the problem of freezing of the application, but not the problem of the long total computational time. The good thing about this approach is that we don't need to work hard to optimize the parameters of the algorithm. Instead, we can use pretty sparse system of frames on every single run of HCC, but if we shuffle thorough enough the initial position, after few runs it will by sure hit the frame with correct position and size.

On the other hand, the statistical improvement of overall performance will yet be achieved by using the sparse system of frames with offsets. The reason for this is the following: to ensure that the positively tested frame will be included in the thick system of frames one has to use the overlap parameter with redundancy. In case of the sparse system with offsets this redundancy will cause that the average index of the path, in which the positively tested frame will be found for the first time, will be less than the half of the total number of the paths, needed to cover all frames from the thick set. Thus the average number of tested frames will be less than in case of testing all frames from the thick set straight away.

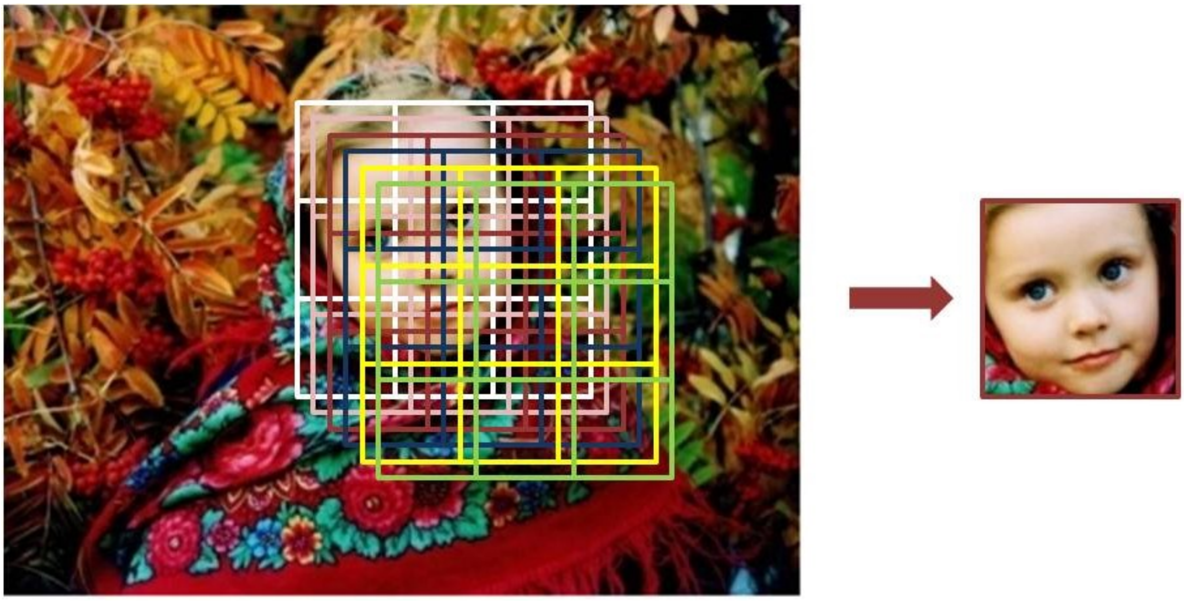

Figure 4. Few sparse systems of frames with offsets. The third system includes the frame, for which Cascade Classifier returns positive result

The frame size used in HCC is limited from below by the minimal number of pixels that allow reliable computation of Haarlike features and from above by the size of the image, or, in the case of non-square image, the size of its minimal side. If the minimal and the maximal size of frame are respectively 20 and 240 and the factor between two consequent values of frame sizes is 1.2, as in OpenCV implementation, the total amount of various frame sizes is ~14, which results in huge amount of tested frames. If one uses factor 1.5 instead of 1.2, the number of needed frame size increments is ~6, which will more than twice reduce the amount of computations. But, according to the same logic as for frame location, the correct frame size can be missed. The same logic will also give the solution of the problem: one can use bigger factor in single path on the image, but the initial frame size for single path should differ from the initial size of the previous path by the small factor, so that after few iterations all sizes from the sequence obtained by incrementing the size by the small factor will be explored. This will further reduce the amount of computations in the single path.

III. Skin regions localization based on color analysis and its use for performance improvements

As it was already described in the introduction, in many publications the color information is utilized for image segmentation. We here suggest using the color information for significant reducing of the amount of computations while applying HCC and therefore improving the performance of face detection.

For this purpose one can use the color analysis as the preliminary step prior to the running of HCC. After color analysis, the areas of the image where the skin-like colored pixels are detected in majority should be defined. As a third step, these areas should be tested by HCC, while the rest of the image should be skipped. The factor of the algorithm acceleration will be reciprocal to the portion of the areas with the majority of the skin-colored pixels. It's clear, that this factor will vary in the different images, but usually the portion of such areas is not exceeds 10-20% of the whole image area, which provides 10-5 times performance improvement.

As shown in [6] in the color image presented in YCrCb color space the location of the chrominance values of the color of the human skin are narrowly and consistently distributed not only for the individual image or person, or skin color of the same human race, but over all possible varieties of the listed. The apparent difference in skin color perceived by viewers is mainly due to the darkness or fairness of the skin; these features are characterized by the difference in the brightness of the color, which is governed by Y but not Cr and Cb. The ranges of the chrominance values can be obtained from the representative set of the tested images, and the ranges R (Cr) = [133, 173] and R (Cb) = [77, 127] are assumed in [6] to be very robust against different types of skin color.

Let's the input image is **Img**(x, y), and it's representation in YCrCb color space is **ImgYCrCb**(x, y). Then the skin bitmap computed with the predefined chrominance ranges, will be defined as

SkinBmp(x, y) =

In order to optimize the speed of computation of the **SkinBmp** only Cr and Cb components of the **ImgYCrCb** can be computed, as the value of luminance Y is of no use for this purpose.

The next step in the combined color analysis with HCC procedure of face detection is to define the areas to be tested with the classifier according to the skin bitmap. First, some morphological operations should be performed on the skin bitmap to avoid noise influences. Second, the maximal square areas with significant portion of pixels with values 1 in the skin bitmap should be outlined and subjected to the HCC. The significant portion of pixels can be defined by some threshold, which in our implementation was set to 0.75.

For the purpose of computation of the maximal areas with the skin color predominance we suggest the following algorithm: consequently test the square frames of the minimal size as it is defined in HCC and incremented as defined in HCC, going from left side to the right side and from the top to the bottom of the image. For the first frame, for which the portion of skin-colored pixels is bigger than set threshold, try to extend it as far as possible in the diagonal direction towards the bottom right corner of the image, keeping the quality of predominance of the skin-colored pixels. The resulting frame includes the sub region of the image to run HCC on it. The scan of the image for more areas with the skin color predominance should be continued on the rest of the image, not including already found areas, until the bottom right corner is reached.

Figure 5. SkinBmp and area with mostly skin-colored pixels

The example of **SkinBmp** can be seen on Figure 5. For the input **Img** shown on the left the **SkinBmp** is presented as a red overlay on the right. The red square presents the maximal area with mostly skin-colored pixels. In this example the ratio between the square included of the red frame and the whole image is close to 1/8; one would get the factor 8 of performance improvement of face detection algorithm, if it wouldn't be for the prior computations of **SkinBmp** and the areas with the skin color predominance. Those additional stages add some computation time and reduce the improvement factor, but since they are less time-consuming than the Haar Cascade, one still get significant performance improvement. In addition, the **SkinBmp** and the areas with the skin color predominance can be successfully computed on the decimated image, which will reduce even more the impact of these extra stages on the overall performance of the face detection.

IV. Results

Using the offset sparse systems of the tested frames and the prior detection of the areas with mostly skin-colored pixels we were able to improve significantly the performance of face detection by applying pre-processing and OpenCV implementation of HCC on the mobile device. The application performance was tested on the mobile phone by HTC Droid Incredible 2, 1 GHz processor and 768MB memory. Some pictures with results are presented on the Figure 6.

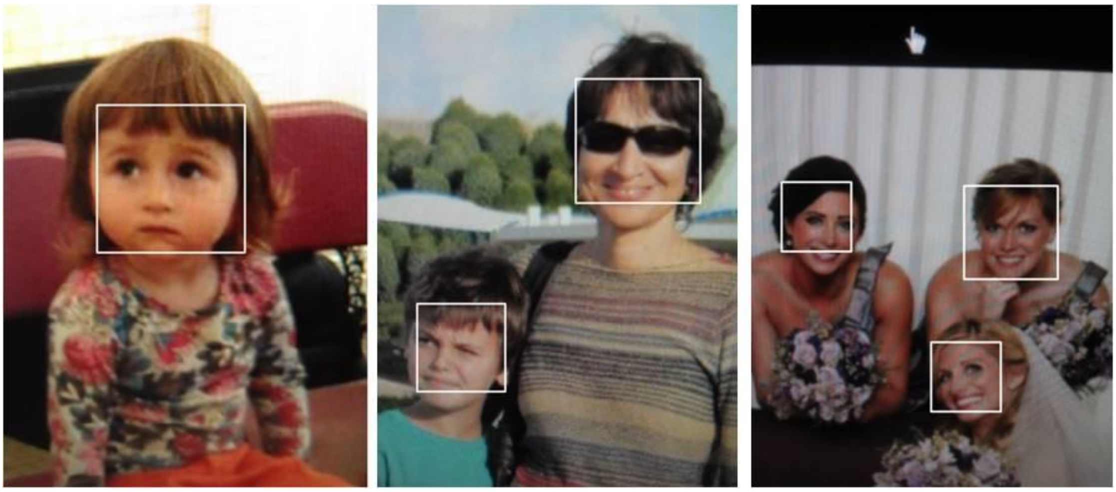

Figure 6. Results of the face detection on color images with one, two or three human faces.

The measurements of the processing times for these pictures are shown in the following table. The achieved performance improvement changes the perception of the application from the poorly responding while detection subroutine is operated to the fully interactive.

Images with N faces	Skin Area Detection, ms	HCC on skin area, ms	Total time, ms	HCC on the whole image, ms	Performance improvement , %
1	28	143	171	1233	86
2	20	213	233	1177	80
3	24	289	313	1175	73

Average	24	215	239	1195	80

In general, the idea of the reducing of the amount of the frames tested by HCC turned out to be very productive for the performance improvements in face detection. Besides suggested and tested in this work ways of reducing of this amount, other ideas could be implemented for this purpose.

REFERENCES

1. Hjelmas, E., Low, B.K.: Face Detection: A Survey, Computer Vision and Image Understanding, Volume 83, Issue 3, 236-274 (2001)

2.Yang, M.H., Kriegman, D.J., Ahuja, N.: Detecting faces in images: A survey, IEEE Transactions on Pattern analysis and Machine intel., Vol. 24, Issue 1, 34-58 (2002)

3. Cha Zhang and Zhengyou Zhang, A Survey of Recent Advances in Face Detection, Microsoft Research Microsoft Corporation, Technical Report MSR-TR-2010-66

4. P. Viola and M. Jones, "Robust real-time object detection," International Journal of Computer Vision, 2002 vol. 57, no. 2, pp. 137-154.

5. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Proc. of CVPR, 2001.

6. D. Chai, K.N. Ngan, Locating facial region of a head-and-shoulders color image Conference on Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International 14-16 Apr 1998

7. Y. J. Zhang, Y. R. Yao, and Y. He, “Automatic face segmentation using color cues for coding typical videophone scenes,” Proc. SPIE Visual Commun. and Image Processing, San Jose, CA, Feb. 1997, vol. 3024, pp. 468–479.

8. M. J. T. Reinders, P. J. L. van Beek, B. Sankur, and J. C. A. van der Lubbe, “Facial feature localization and adaptation of a generic face model for model-based coding, ”Signal Processing Image Community, vol.7, no. 1, pp. 57–74, Mar. 1995.

9. Sanjay Kr. Singh, D. S. Chauhan, Mayank Vatsa, Richa Singh, A Robust Skin Color Based Face Detection Algorithm, Tamkang Journal of Science and Engineering, Vol. 6, No. 4, pp. 227-234 (2003)

10. Andrew W Yip, Pawan Sinha, Contribution of color to face recognition, Perception, 2002, volume 31, pages 995 – 1003

11. Ali Atharifard, Sedigheh Ghofrani, Robust Component-based Face Detection Using Color Feature, Proceedings of the World Congress on Engineering 2011 Vol II WCE 2011, July 6 - 8, 2011, London, U.K