

Red Eye Detection through Bag-of-Keypoints Classification

Sebastiano Battiato¹, Mirko Guarnera², Tony Meccio¹, and Giuseppe Messina²

¹ Università degli Studi di Catania

Dipartimento di Matematica
Viale A. Doria 6, 95125 Catania

² STMicroelectronics
Advanced System Technology
Stradale Primosole 50, 95121 Catania

Abstract. Red eye artifacts are a well-known problem in digital photography. Small compact devices and point-and-click usage, typical of non-professional photography, greatly increase the likelihood for red eyes to appear in acquired images. Automatic detection of red eyes is a very challenging task, due to the variability of the phenomenon and the general difficulty in reliably discerning the shape of eyes.

This paper presents a method for discriminating between red eyes and other objects in a set of red eye candidates. The proposed method performs feature-based image analysis and classification just considering the bag-of-keypoints paradigm. Experiments involving different keypoint detectors/descriptors are performed. Achieved results are presented, as well as directions for future work.

Keywords: Red Eyes, Feature Classification, SIFT, GLOH, SVM.

1 Introduction

The Red Eye phenomenon is a well-known problem which happens when taking flash-lighted pictures of people, causing pupils to appear red instead of black. This is especially problematic in non-professional photography, where imaging devices are small in size. Thus, great need arises for a red eye removal system able to reliably detect and correct red eye artifacts without any user intervention [1].

Red eye artifacts are caused by direct reflection of light from the blood vessels of the retina through the pupil to the camera objective. High-end cameras often feature a separate flash with an extensible and steerable bracket (Fig. 1), which allows for more distance between the flash and the lens, thus reducing the probability for red eyes to appear. One preventive measure suitable to both high-end and low-end devices is to make additional flashes before actually taking the photograph. This method gives time to pupils to shrink in order to reduce the reflectance surface, thus making red eyes less likely. Despite the increased power consumption due to the additional flashes, this is a widespread feature of small digital cameras.

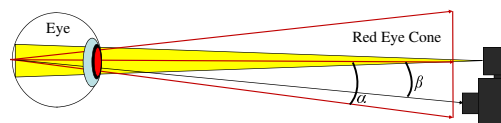


Fig. 1. Flash-gun light cone generated by reflection off the retina. If the angle α , representing the cone size, is greater than the angle β , between the camera lens and the flash-gun, then the red-eye artifact comes out.

In the easier cases, the small circle of the pupil is a clearly distinguishable red color disk instead of a normal black one. Usually a small white glint is also present, representing the direct reflection of the flash on the surface of the eye and giving the eye much more naturalness. However, red eye artifacts present a great degree of variability, as they may appear different in shape and color, and may also differ in position and size relative to the whole eye [2].

As the detection and the correction problems are well separable, lots of techniques exist which address only one of them. Typical red eye detection approaches involve extraction of “red” zones combined with template matching, face detection and/or skin extraction. Some approaches also make use of classifiers to better discriminate true eyes from false positives.

Patti et al. [3] used a nonstandard luminance-chrominance representation to enhance the regions affected by the red-eye artifact. After the detection of a block of maximal area, thresholding operation and a simple color replacement structure are employed. Most of the red eye detection algorithms apply some constraints to restrict possible red-eye regions in conjunction with reliable learning-based classifiers. Gaubatz and Ulichney [4] first applied a face detector and then searched for the eyes in the candidate face regions by using the constraints of color variance, red variance and glint. The results greatly depend on the face detector which only deals with frontal and upright faces.

Schildkraut and Gray [5] proposed an automatic algorithm to detect pairs of eyes, which is restricted to near-frontal face images. Ioffe [6] proposed a machine learning approach by combining two detectors. The first is the red-eye and non-red-eye classifier trained with boosting method on numerous training patches and the second is the face detector which is used to reduce the false positives. However, many photos have single red eyes (e.g., face partially screened). Zhang et al. [7] proposed a two-stage algorithm for red eye detection. At the first stage, red pixels are grouped and a cascade of heuristic algorithms based on color, size and highlight are used to decide whether the grouped region is red-eye or not. At the second stage, candidate red-eye regions are checked by the Adaboost classifier [8].

Though highlight is a good constraint for red eye detection, red eyes with no highlight region may occur when the eye direction does not face toward the camera/flash light. Luo et al. [9] proposed an algorithm that uses multi-scale square concentric templates to assess the candidate red-eye regions, followed by an Adaboost classifier.

Volken et al. [10] detect the eye itself by finding the appropriate colors and shapes. They use the basic knowledge that an eye is characterized by its shape and the white color of the sclera. Combining this intuitive approach with the detection of “skin” around the eye they obtain good results. Safonov [2] suggested to take into account color information via 3D tables and edge information via directional edge detection filters. For classification a cascade of classifiers including AdaBoost have been used. The approach proposed by Ferman [11] uses a novel unsupervised method to discover red eye pixels. Analysis is performed primarily in the Hue-Saturation-Value (HSV) color space. A flash mask, which defines the regions where red-eye regions may be present, is first extracted from the brightness component. Subsequent processing on the other color components prunes the number of candidate regions that may correspond to red eyes.

In this paper we propose to attack the problem of red eye detection just using the bag-of-keypoints paradigm. It involves extraction of local image features, quantization of the feature space into a codebook through clustering, and extraction of codeword distribution histograms. A classifier is used to decide to which class each histogram, thus each image, belongs. Approaches of this kind have been shown to be able to recognize different kinds of images in a variety of applications [12] [13].

Our idea is to employ a classification technique to discriminate images representing red eyes from ones representing false candidates. In particular we propose to analyze the input dataset just considering a set of well-known keypoint detectors/descriptors [14] [15] [16] [17]. Support Vector Machine [18] [19] is used as final classifier. Such an approach is shape-based, thus robust to red image features which often cause false positives in color-based red eye detection algorithms, and is capable of detecting more complex features than most template-based methods. This, combined with a color-based red eye candidate extractor, and/or with a correction technique which leaves non-red zones untouched, may contribute to a full system robust to both color-based and shape-based false positives. Experiments confirm the effectiveness of the proposed pipeline.

This paper is organized as follows. Section 2 presents an overview of the proposed algorithm. Sections 3, 4 and 5 describe the feature extraction, quantization, and classification steps, respectively. Section 6 presents the results achieved, and in Section 7 suggestions for future work are given.

2 Algorithm Overview

The overall algorithm pipeline is depicted in Fig. 2. First, images are analysed with a local feature extraction method. This kind of methods scan the images for “interesting” points (keypoints) and compute the image information around the keypoints to associate to each a descriptor which is distinctive of the nature of the object(s) the keypoint belongs to. Keypoints are localized relative to their position, scale and orientation, and in some cases are normalized with respect to affine deformations, in order to make them invariant to as many variations as possible.

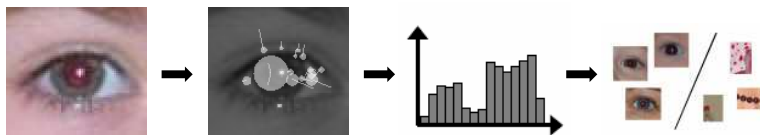


Fig. 2. The proposed algorithm pipeline. Left to right: Input image, feature extraction, feature quantization, classification.

In this application, the objects to describe are the various parts of the eye. Thus, it is fundamental to extract features distinctive of such parts, in order to well discriminate them from parts belonging to false candidates.

To compute a fixed-length vector from each image, local features extracted (which are variable in number) are counted into a histogram. Bins are distributed across the feature space in a way such that more populated portions of the space have more bins, in order to obtain more meaningful histograms.

Histograms are given as input to a classifier to separate the characteristics of the histograms of the two classes (eye and non-eye) and to discriminate them with high accuracy. A training set of labeled images is used to train the classifier and to find the optimal parameters for it.

3 Feature Extraction

The problems of keypoint detection and description can be treated independently. Experiments were made combining different detectors and descriptors, totalling 11 alternatives for image analysis.

As discussed in [15], the Harris corner detector [20] can be adapted to detect corners at multiple scales. To select only the most significant scale(s) for each feature, only corners for which the Laplacian-of-Gaussian operator attains a maximum over scale are selected (Harris-Laplace detector (HarLap)).

The Hessian-Laplace (HesLap) detector [16] looks for keypoints for which the determinant of the Hessian matrix, computed at different scales, attains a local maximum over space. In the same way as Harris-Laplace, the Laplacian-of-Gaussian operator is used to select the features at the most significant scale(s).

Since the Harris-Laplace and the Hessian-Laplace detectors are somewhat complementary to each other (corner-like vs. blob-like structures), it is possible to use both to obtain a richer representation (Harris-Hessian-Laplace detector (HarHesLap)).

Both the Harris-Laplace and Hessian-Laplace detectors can be further refined to detect affine invariant regions. To accomplish this, an iterative method is used to estimate the optimal shape adaptation matrix for each keypoint, thus extracting elliptical keypoints instead of circular ones (Harris-Affine (HarAff) and Hessian-Affine (HesAff) detectors). In these cases, the region underlying each keypoint is normalized to a circular/square one prior to computation of descriptors.

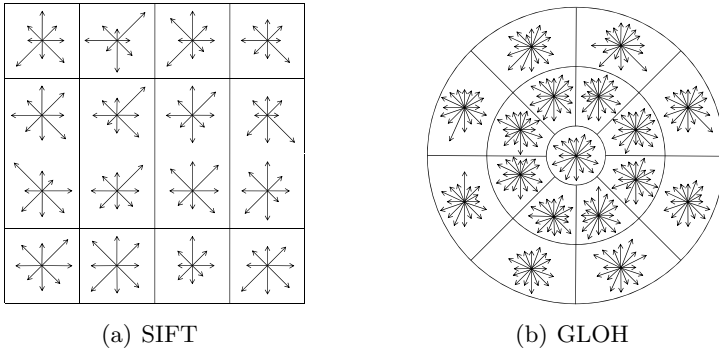


Fig. 3. Schematic representation of the two descriptors

One of the descriptors used to describe the extracted keypoints is the Scale Invariant Feature Transform (SIFT) descriptor [14]. It has been originally introduced for the SIFT image analysis technique. This method selects keypoints at different scales by computing a band-pass gaussian pyramid using the Difference-of-Gaussians (DoG) operator (an approximation of the Laplacian-of-Gaussian (LoG)), and local maxima and minima of the pyramid are selected as keypoints at the corresponding scale. Keypoints are filtered according to stability measures based on local contrast and “cornerness”, and then orientations are assigned to them according to local image gradients in order to achieve rotational invariance. It is shown in [16] that the Hessian-Laplace detector responds to similar structures as the maxima of the Difference-of-Gaussians, but has high responses near corners and low responses near edges, making it unnecessary to filter keypoints on local cornerness. The SIFT descriptor is composed by a 4-by-4 grid of 8-bin local gradient orientation histograms. The grid is centered on the keypoint, sized accordingly to its scale and rotated accordingly to its orientation. The resulting 128-dimensional vector is normalized to unit Euclidean length in order to compensate for contrast variations (see Fig.3(a)).

The other employed descriptor is the Gradient Location and Orientation Histogram (GLOH) (see Fig.3(b)), a variant of the SIFT descriptor. Like the latter, it is a set of gradient orientation histograms, but instead of being a grid, it has a “crosshair”-like shape, with a central circular region and two external concentric rings each divided in the angular direction in 8 regions. This adds to 17 histograms, each consisting of 16 bins. The 272-dimensional vector is then reduced to 128 dimensions with PCA.

4 Feature Space Quantization

Clustering is used to select a meaningful subdivision of the feature space into histogram bins. Descriptors from the training set are clustered with the k-means algorithm [21], with $k=50$. The set of centroids found is used as a “codebook”

of representative features, and one bin is assigned to each, thus obtaining a finer quantization in more populated regions of the feature space. Each descriptor contributes to the bin relative to the closest centroid.

In some cases, no keypoints are detected in a given candidate image. Since it almost always happens for false candidates, e.g., for blurry background objects, images without keypoints are considered non-eyes and are discarded from further consideration.

Prior to classification, histograms are normalized in order to make them less dependant to the number of keypoints detected. Then, since the classifier used (see below) considers euclidean distance between vectors, i.e. the 2-norm of the difference, a trasformation is performed to make this distance more meaningful: namely, histograms are transformed by taking the square root of each bin. This converts 1-norm normalized vectors into 2-norm normalized vectors. The dot product between two of these vectors, which is equivalent to the cosine of the angle between them, is the Bhattacharyya coefficient between the original (1-norm normalized) vectors [22]. This coefficient is a common measure of similiarity between frequency distributions, and it can be shown that the euclidean distance calculated this way is proportional to the metric commonly associated to the coefficient:

$$BC(\mathbf{v}, \mathbf{w}) = \sum_{i=1}^n \sqrt{v_i} \cdot \sqrt{w_i} . \quad (1)$$

$$\begin{aligned} d(\overline{\mathbf{v}}, \overline{\mathbf{w}}) &= \sqrt{\sum_{i=1}^n (\overline{v}_i - \overline{w}_i)^2} = \sqrt{\sum_{i=1}^n \overline{v}_i^2 + \sum_{i=1}^n \overline{w}_i^2 - 2 \cdot \sum_{i=1}^n \overline{v}_i \cdot \overline{w}_i} = \\ &= \sqrt{\sum_{i=1}^n v_i + \sum_{i=1}^n w_i - 2 \cdot \sum_{i=1}^n \sqrt{v_i} \cdot \sqrt{w_i}} = \\ &= \sqrt{2 - 2 \cdot BC(\mathbf{v}, \mathbf{w})} = \sqrt{2} \cdot \sqrt{1 - BC(\mathbf{v}, \mathbf{w})} . \quad (2) \end{aligned}$$

In the above formulas, BC is the Bhattacharyya coefficient, d is the Euclidean distance, \mathbf{v}, \mathbf{w} are 1-norm normalized vectors and $\overline{\mathbf{v}}, \overline{\mathbf{w}}$ are the corresponding 2-norm normalized vectors.

5 SVM Classification

Histograms are classified using a Support Vector Machine classifier [18] [19]. This classifier works by finding an optimal separation hyperplane between two different classes of labeled training vectors and specifying a small number of weighted vectors which lie on the boundaries of the classes (these vectors are called “support vectors”). Since a linear separation usually isn’t meaningful, vectors are usually projected into a higher-dimensional space and then linearly classified in that space. However, computing the projected vectors explicitly

can be expensive or even impossible. Instead, the problem can be expressed in a form where the projected vectors only appear in dot products between two of them. Thus, a common practice is to employ a function which, given two vectors in the original space, computes the dot product of their projections in the higher-dimensional space (usually in a much simpler way). This function is named “kernel”, and this practice is named “kernel trick”. It is proven that any continuous, symmetric, semidefinite positive function from \mathbf{R}^n to \mathbf{R} can be used as a classification kernel.

The kernel used in the experiments is the Radial Basis Function (RBF) kernel, which is a multidimensional non-normalized gaussian:

$$K(\mathbf{v}, \mathbf{w}) = e^{-\gamma \cdot \|\mathbf{v} - \mathbf{w}\|_2^2}, \quad \gamma > 0. \quad (3)$$

As shown above, the aperture of the gaussian function is controlled by a parameter γ . This is one of the parameters which must be adjusted in order to obtain the most accurate classification for each given training set. The other parameters to find, called C_1 and C_2 , are penalty factors for outliers in the two classes of the training set. It is important to carefully adjust them in order to find an optimal tradeoff between accuracy of the classification and tolerance to outliers, which is important to prevent overfitting. The two parameters are adjusted independently to achieve more generality.

Optimal parameters are searched with a multi-level grid search using 8-fold cross-validation for training and testing: first, a grid of parameter triples, spaced evenly in a logarithmic scale, is tried, then a finer grid covering the parameters who gave the best results is tried, and so on, up to the fifth level of grid refinement.

6 Experimental Results

The proposed red-eye detection system has been trained on a data set of 4079 image patches, comprising 956 red eyes, and tested on a data set of 5797 image patches, comprising 234 red eyes. The sets has been collected from photographs taken with various sources, including DSLR images, compact cameras and Internet photos, and the image candidates have been extracted using an automatic red cluster detector [23]. This means that the “eye” class is mostly trained with red eye patches: however, since the method is shape-based, regular eyes are recognized as well, as red eye artifacts have little impact on the luminance. Training on red eyes allows to learn the slight differences in shape which occur (e.g. biggest and brightest pupils, different luminance distribution), while classifying regular eyes along with red eyes is not a problem, since regular eyes have no red pupils to correct, then a color-based correction algorithm can discard them easily, and a “blind” correction algorithm can desaturate them with almost no harm. Table 1 shows results achieved with the different detector+descriptor combinations. Hit rate is the percentage of eyes correctly classified out of the total number of eyes; False positives is the percentage of false candidates incorrectly classified out of

Table 1. Classification results for each detector/descriptor combination tested

Detector + Descriptor	Hit rate	False positives	Overall perf.	Accuracy
DoG + SIFT	83.3%	8.9%	87.20%	90.78%
HarLap + SIFT	78.2%	12.2%	83.00%	87.47%
HesLap + SIFT	71.8%	7.1%	82.35%	92.12%
HarHesLap + SIFT	82.9%	3.5%	89.70%	95.94%
HarAff + SIFT	70.1%	8.1%	81.00%	91.08%
HesAff + SIFT	82.9%	7.0%	87.95%	92.59%
HarLap + GLOH	75.2%	12.1%	81.55%	87.38%
HesLap + GLOH	80.3%	14.4%	82.95%	85.41%
HarHesLap + GLOH	76.9%	7.3%	84.80%	92.07%
HarAff + GLOH	73.5%	11.9%	80.80%	87.52%
HesAff + GLOH	69.2%	7.3%	84.80%	92.07%



Fig. 4. Classification examples obtained with Harris-Affine + SIFT. Top row: (a), (b), (c) are eye patches correctly classified; (d) has been incorrectly classified as non-eye. Bottom row: (e), (f), (g) have been correctly classified as non-eyes; (h) is a false positive.

the total number of false candidates; Overall performance is $(\text{Hit rate} + (100\% - \text{False positives}))/2$; Accuracy is the percentage of patches correctly classified out of the total number of patches.

It can be seen from the results that, while there are quite a significant amount of eyes that are missed, a restricted percentage of false candidates are misclassified as eyes (using the best performing detector+descriptor combinations). This is a remarkable result, since correction of false positives is one of the biggest problems in red eye removal, thus it is very important to keep false positives as low as possible. It is also evident how the blob-based detectors yielded better results than the corner-based detectors. This is not surprising, as many parts of the eye are more suitable to be extracted by a blob-like detector (especially the

iris and the pupil). Using both types of detector together, however, helps keep the false positives lower. SIFT performed better than GLOH.

Comparison with other red eye detectors cannot be made, since the proposed approach doesn't select candidates on its own. It cannot even be made with the classification phase of two-step detectors (candidate detection + classification), since in these cases the performance of the classifier is heavily influenced by the number of candidates extracted, and is also affected by how much the candidate detector and the classifier are "fine-tuned" for each other.

7 Conclusions and Future Work

In this work we showed the effectiveness of Bag-of-Keypoints-based classification in discriminating red eyes from false red eye candidates. Results are interesting, but improvement is needed. Further studies will be performed with different keypoints detectors/descriptors and other kind of image analyses, including spatial domain-based features.

Acknowledgments. The authors would like to thank Giovanni Maria Farinella and Giovanni Puglisi for their precious advice about image analysis, scene classification and SVM.

References

1. Gasparini, F., Schettini, R.: Automatic Red-Eye Removal for digital photography. In: *Single-Sensor Imaging: Methods and Applications for Digital Cameras*, pp. 429–457 (2008)
2. Safonov, I.V.: Automatic red-eye detection. In: *GraphiCon., International conference on the Computer Graphics and Vision, Moscow, Russia* (2007)
3. Patti, A.J., Konstantinides, K., Tretter, D., Lin, Q.: Automatic digital redevye reduction. In: *IEEE International Conference on Image Processing (ICIP 1998)*, Chicago, Illinois, vol. 1, pp. 55–59 (1998)
4. Gaubatz, M., Ulichney, R.: Automatic red-eye detection and correction. In: *IEEE International Conference on Image Processing (ICIP 2002)*, Rochester, New York, USA, vol. 1, pp. 804–807 (2002)
5. Schildkraut, J.S., Gray, R.T.: A fully automatic redevye detection and correction algorithm. In: *IEEE International Conference on Image Processing (ICIP 2002)*, Rochester, New York, USA, vol. 1, pp. 801–803 (2002)
6. Ioffe, S.: Red eye detection with machine learning. In: *IEEE International Conference on Image Processing (ICIP 2003)*, Barcelona, Catalonia, Spain, vol. 2, pp. 871–874 (2003)
7. Zhang, L., Sun, Y., Li, M., Zhang, H.: Automated red-eye detection and correction in digital photographs. In: *IEEE International Conference on Image Processing (ICIP 2004)*, pp. 2363–2366 (2004)
8. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 55, 119–139 (1997)

9. Luo, H., Yen, J., Tretter, D.: An efficient automatic re-eye detection and correction algorithm. In: IEEE International Conference on Pattern Recognition (ICPR 2004), pp. 883–886 (2004)
10. Volken, F., Terrier, J., Vandewalle, P.: Automatic red-eye removal based on sclera and skin tone detection. In: Third European Conference on Color in Graphics, Imaging and Vision (CGIV 2006), Society for Imaging Science and Technology, pp. 359–364 (2006)
11. Ferman, A.M.: Automatic detection of red-eye artifacts in digital color photos. In: IEEE International Conference on Image Processing (ICIP 2008), San Diego, California, vol. 1, pp. 617–620 (2008)
12. Battiato, S., Farinella, G.M., Gallo, G., Ravi, D.: Scene categorization using bags of textons on spatial hierarchy. In: International Conference on Image Processing, ICIP 2008 (2008)
13. Bosch, A., Zisserman, A., Munoz, X.: Scene classification using a hybrid generative/discriminative approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (2008)
14. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60, 91–110 (2004)
15. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. *International Journal of Computer Vision* 60, 63–86 (2004)
16. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.* 27, 1615–1630 (2005)
17. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L.J.: A comparison of affine region detectors. *International Journal of Computer Vision* 65, 43–72 (2005)
18. Boser, B.E., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: COLT, pp. 144–152 (1992)
19. Cortes, C., Vapnik, V.: Support-vector networks. *Machine Learning* 20, 273–297 (1995)
20. Harris, C., Stephens, M.: A combined corner and edge detection. In: Proceedings of The Fourth Alvey Vision Conference, pp. 147–151 (1988)
21. Hartigan, J.A., Wong, M.A.: A K-means clustering algorithm. *Applied Statistics* 28, 100–108 (1979)
22. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25, 564–575 (2003)
23. Battiato, S., Farinella, G.M., Guarnera, M., Messina, G., Ravi, D.: Red-eyes detection through cluster based linear discriminant analysis (to appear)