

Automatic Detection and Correction of Red-Eye Effect

Tauseef Ali, Asif Khan and Intaek Kim

Signal Processing Lab, Dept. of Communication Engineering,
Myongji University, Yongin Korea.

{tauseefcse,asifkhan2k5}@yahoo.com, kit@mju.ac.kr

Abstract

In this paper, we address the problem of automatic red-eye detection and correction. A four step, novel approach is presented where the first step consists of face detection in the input image. In the second step the face image is converted to gray-scale image in such a way that facilitates the detection of iris pair and its corresponding radius values. In third and fourth steps iris pair is located, and the central points of irises and their corresponding radius values are utilized to desaturate the redness inside iris regions. The desaturation scheme is adaptive to the severity of redness. Images with different severity level and size of redness are used to test the robustness of the proposed scheme. We also compare our correction schemes with two existing automatic methods.

1. Introduction

The red-eye artifact is one of the most common problems in digital photographs. It is caused by the flash light used to illuminate the subject in case of insufficient natural light. Some available softwares such as Adobe PhotoshopTM allow for manual correction of red-eye. But these are not easy to use and don't give satisfactory results. Recently, some research has been done on automatic red-eye correction but outside the patent literature [1,2], there is not much published about this subject. In this paper, we present a robust method to correct red-eye artifact. The algorithm first detects face in an input image. Face image is converted to gray-scale image in such a way which facilitates eye detection. Our algorithm for eye detection assumes that iris is black compared to its surrounding (sclera of eye). However, this assumption is not true for red-eye case. So in step 2, we convert image into gray-scale in such a way that red color inside iris become black which facilitate and aids in eye detection step. In step 3 we propose a precise and robust method to locate iris pair and radius of each iris. The center points of iris and its corresponding radius value is then used in step 4 to correct redness inside

iris. Our red-eye correction is adaptive to the severity of redness and produce perceptually good results with almost no false positives. In Fig.1 the steps of the proposed method are illustrated using a test image.

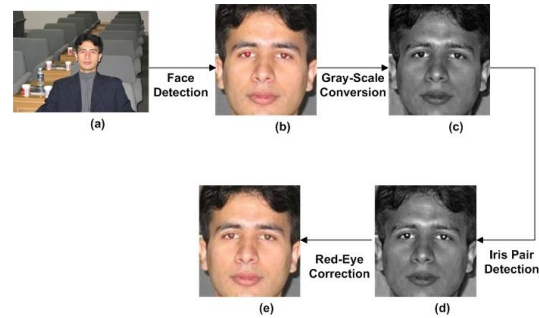


Fig. 1: Steps of Red-Eye Correction Process

2. Face Detection

We first detect the face in the input image. Then problem is simplified due to the background being restricted to the face. It saves searching time and improves accuracy. For face detection, Viola's method [3] is used. A robust face classifier is obtained by supervised AdaBoost learning. Given a sample set of training data $\{x_i, y_i\}$, the AdaBoost algorithm selects a set of weak classifiers $\{h_j(x)\}$ from a set of Haar-like rectangle features and combine them into a strong classifier. The strong classifier $g(x)$ is defined as follow:

$$g(x) = \begin{cases} 1 & \sum_{k=1}^{k_{\max}} \alpha_k h_k(x) \geq \theta \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where θ is the threshold that is adjusted to meet the detection rate goal. The Haar-like rectangle features are easily computed using "integral image" representation. The "cascade" method quickly filters out non-face image areas. More details can be found in [3]. Training data is selected from different well-known face databases. All of the face images are first processed using gray-scale and size normalization to 20*20

pixels. Negative sample are obtained using bootstrapping method.

3. Gray-Scale Conversion

If the original RGB image is converted to gray-scale by eliminating the Hue and saturation information while retaining the luminance, the red-eye pixels are converted to white pixels in gray-scale image. This makes it difficult to detect the center of eyes by the algorithm that we use. To avoid this problem we develop a method to convert the color RGB image to gray-scale in such a way that the intensities of red pixels are decreased proportionally to their redness. We proposed the following equation for converting RGB face image into gray-scale image.

$$Gray(x, y) = \frac{G(x, y) \times B(x, y)}{R(x, y)} \quad (2)$$

The resultant image is shown in Fig 1(b). This conversion facilitates eye detection in the next section.

4. Iris Pair Detection

Fig.2 shows the steps of iris pair detection in gray-scale image. The input to this step is the gray image obtained in section 3 as shown in Fig 2(a).

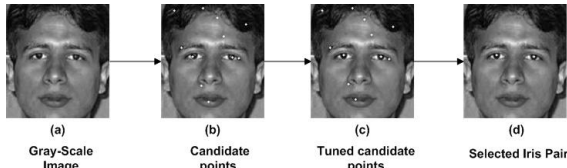


Fig. 2 Steps of iris pair detection

4.1. Eye Candidates Detection

By changing the training data and increasing the false-positive rate of the algorithm in section 2, we build an eye candidate detector. The training data of [4] is used to detect several eye candidate points in face region. A total of 7000 eye samples are used with the eye center being the center of the image and resize to 16*8 pixels. Because in this step face region is already detected, so the negative samples are taken only from the face images. We set low threshold and accept more false positive according to eq.1. On the average, we get 15 eye candidates out of the detector. Fig 2(b) shows the candidate points obtained on a test image.

4.2. Tuning Candidates Points

The candidate points generated by AdaBoost in section 4.1, contains two points which represent eyes. But in most cases AdaBoost can't point out the center of irises. So we need to shift the candidate points within a small size of neighborhood so that two of the candidate points are exactly in center of irises. This tuning process ultimately improves the accuracy of the final detected eye points and gives the radius value associated with each candidate point. These radius values are later used in next section for filtering candidate points and also for correction step. The separability filter proposed by Fukui and Yamaguchi [17] is utilized in an efficient and novel way to shift the candidate points within a small size of neighborhood. If we form two concentric circles around the center of the irises and measure the separability between small circle and outer part of the large circle as shown in template of Fig.3, separability value is greater for iris as compared to other candidates. For a neighborhood of size $m \times n$, we have $m \times n$ new candidate points for each of original candidate point. By using the template in Fig.3, the separability value (η) is computed for each point in the neighborhood by the following equation.

$$\eta = \frac{B}{A} \quad (3)$$

$$A = \sum_{i=1}^N (I(x_i, y_i) - \bar{P}_m)^2$$

$$B = n_1 (\bar{P}_1 - \bar{P}_m)^2 + n_2 (\bar{P}_2 - \bar{P}_m)^2$$

where n_k ($k = 1; 2$) is the number of pixels in R_k ; $N = n_1 + n_2$; P_k ($k=1; 2$) is the average intensity in R_k ; P_m is the average intensity in the union of R_1 and R_2 , and $I(x_i; y_i)$ the intensity values of pixels $(x_i; y_i)$ in the union of R_1 and R_2 .

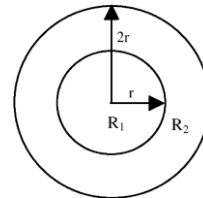


Fig. 3: An eye template (R1 is the inside region of the smaller circle and R2 is the region between the two concentric circles).

Using equation 3 and separability filter shown in Fig. 3, we repeat the process of forming neighborhood and selecting a new candidate for each of original candidate point. Separability values for each of the point in the neighborhood are determined by varying the radius in a range $\{R_L, \dots, R_U\}$. In our experiments we used $R_L = 3$ and $R_U = 6$. The point in the neighborhood which gives maximum separability is considered as the new candidate point. In our experiment we checked for several sizes of neighborhoods, and concluded that a 9×9 neighborhood is most efficient and correct. We also find the separability values for each new candidate point and its corresponding optimal radius R among $\{R_L, \dots, R_U\}$. These separability values and radius for new candidate points are used later. Fig 2(c) shows the new candidate points.

4.3. Iris Pair Selection

We combine three metrics to measure the fitness of each candidate point with iris and select iris pair.

4.3.1. Mean crossing function. A rectangular subregion is formed around each iris candidate. The size of the subregion is depicted in Fig. 4, where R is the radius of the candidate determined in section 4.2. A subregion of the form shown in Fig. 4 is formed around each candidate point. The subregion is scanned horizontally and the mean crossing function [6] for pixel (i, j) is computed as follows:

$$\mu C(i, j) = \begin{cases} 1 & ; \text{If } I(i, j) \geq \mu + A \text{ then if } I(i, j+1) \leq \mu - A \\ 1 & ; \text{If } I(i, j) \leq \mu - A \text{ then if } I(i, j+1) \geq \mu + A \\ 0 & ; \text{otherwise} \end{cases} \quad (4)$$

where A is a constant. The horizontal mean crossing value for the subregion is determined as

$$\mu C_{subregion} = \sum_{i=1}^M \sum_{j=1}^N \mu C(i, j) \quad (5)$$

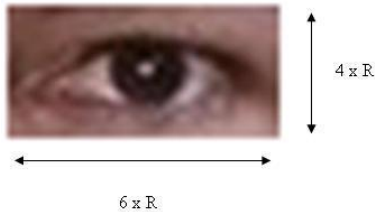


Fig. 4: Subregion for mean crossing function

In a similar way, vertical mean crossing function is evaluated by scanning vertically the subregion. To find the final mean crossing value for the subregion, we linearly add both mean crossing numbers.

4.3.2. Convolution with edge image subregion.

First we find the edge image of the subregion around the candidate point. The size of the subregion is the same as the mask in Fig. 5. The subregion is convolved with the convolution kernel shown in Fig. 5 with the edge image of the subregion. The radius of the template is equal to the radius of the candidate determined in section 4.2. The center of the template is placed on the candidate point and the value of convolution is determined. The process is repeated for each of the candidate. The resultant signal from convolution is summed up and a single value is obtained.

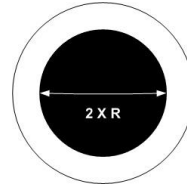


Fig. 5: Convolution template

Now, we define fitness of an iris candidate by the following equation.

$$fitness(C_x) = \frac{\mu C_{subRegion}(C_x)}{\sum_{i=1}^N \mu C_{subRegion}(C_i)} + \frac{Conv(C_x)}{\sum_{i=1}^N Conv(C_i)} + \frac{Seperability(C_x)}{\sum_{i=1}^N Seperability(C_i)} \quad (6)$$

where N is the total number of candidate points, $\mu C_{subRegion}(C_j)$, $Conv(C_j)$, are the mean crossing value and convolution result for candidate C_j and $Seperability(C_j)$ is the separability value for candidate C_j computed in section 4.2.

Now, candidate pair with the maximum fitness is taken as the iris pair according to the following equation.

$$fitness(Pair(C_x, C_y)) = fitness(C_x) + fitness(C_y) \quad (7)$$

5. Red-Eye Correction

In section 4.2, we determine the radius of each candidate point. After finding the iris pair, we use the iris pair radius and their centers to determine the iris

region. For some images, these regions are shown in Fig 6(c). Now the correction process is only applied to these iris regions. We correct the red-eye artifact in YC_bC_r color space. First the pixels inside iris circle are converted to monochrome in which the red-eye pixels are highlighted as a bright spot. For this the approach of [7] is followed.

$$T(x, y) = \frac{[R(x, y) - \max\{G(x, y), B(x, y)\}]}{R(x, y)} \quad (8)$$

In [12] this conversion is applied to the whole image to detect red areas in the image.

Now the YC_bC_r color space the detected iris circle is modified as follows:

$$Cr(x, y)_{Corrected} = (1 - T(x, y)) \times Cr(x, y) \quad (9)$$

The correction is proportional to the redness of the pixel and don't have hard correction boundaries.

6. Experimental Results

We tested our approach on 50 images with varying size and severity of redness. All the images consist of one subject. Subjects have different pose, ethnicity and facial expression. Fig 6 shows some of the examples of the correction procedure. As can be seen in Fig 6(d) the corrected red-eye has improved visual appearance and natural color of the subject. We compare the efficiency of the proposed algorithm with two automatic methods available online: Hewlett Packard Redbot [8] and Stoik Red Eye Autofix. Table 1 shows the quantitative results for each of the method when tested on the same 50 images. As can be seen in table 1, our algorithm performs better than existing methods.

7. Conclusion

We have proposed an algorithm which can correct red-eye images with high correction rate as well as natural visual appearance. The algorithm is based on robust iris pair localization. After iris pair is localized the iris radius and central points of irises are used to correct the redness inside iris. The correction is proportional to the severity of redness. The algorithm is tested and compared with state-of-art works available and shows improved correction rate as well as better quality recovery of eyes.

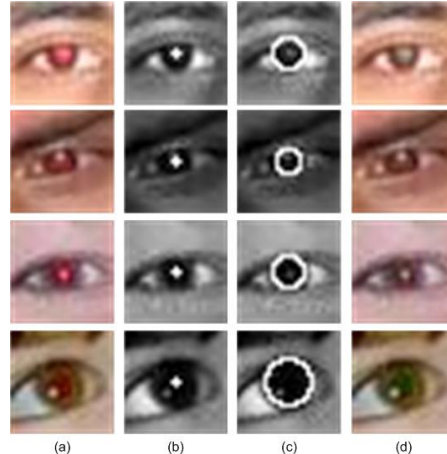


Fig. 6: Some examples of red-eye correction by proposed method. (a) Test image of the red-eye region. (b) detected iris center. (c) iris region detected using iris center and iris radius. (d) corrected red-eye region.

Table 1: Comparison of the proposed method with other automatic methods using 50 test images

Result / Method	No red-eye corrected	One red-eye corrected	False Positives	Both Red-eye corrected
Proposed Algorithm	7	2	4	41
HP	8	11	8	31
Stoik	23	8	2	19

8. References

- [1] Benati, P.J., Gray, R.T., Cosgrove, P.A., 1998. "Automated detection and correction of eye color defects due to flash illumination" *US Patent 5,748,764*, May.
- [2] Dobbs, C.M., Goodwin, R.M., 1992. "Localized image recoloring using ellipsoid boundary function" *US Patent 5,130,789*, July.
- [3] Viola, P., Jones, M.: "Rapid Object Detection Using a Boosted Cascade of Simple Features" In *Computer Vision and Pattern Recognition Conference 2001*, 1 (2001) 511-518.
- [4] Modesto Castrillón-Santana, Javier Lorenzo-Navarro, Oscar D'éniz-Suárez, José Isern-González, and Antonio Falcón-Martel, "Multiple Face Detection at Different Resolutions for Perceptual User Interfaces". 2nd Iberian Conference on *Pattern*

Recognition and Image Analysis, LNCS 3522, pp. 445–452, 2005.

[5] K. Fukui, O. Yamaguchi, “Facial feature point extraction method based on combination of shape extraction and pattern matching”, *Trans. IEICE Japan J80-D-II* (8) (1997) 2170–2177(in Japanese).

[6] Chun-Hung Lin and Ja-Ling, “Automatic facial feature extraction by genetic algorithms”, *IEEE Trans. Image Process.* 8 (6) (1999)1057–7149.

[7] B. Smolka, K. Czubin, J. Y. Hardeberg, K. N. Plataniotis, M. Szczepanski, K. Wojciechowski, “Towards Automatic Red Eye Effect Removal”, *Pattern Recognition Letters*, vol. 24, no. 11, pp.1767-1785, 2003.

[8] Redbot, Hewlett-Packard Labs, “RedBot automatic red eye correction”,<http://redbot.net/>.