Red Eye Removal using Digital Color Image Processing

Jon Y. Hardeberg
Conexant Systems, Inc., Redmond, Washington

Abstract

The current paper provides methods to correct the artifact known as "red eye" by means of digital color image processing. This artifact is typically formed in amateur photographs taken with a built-in camera flash.

To correct red eye artifacts, an image mask is computed by calculating a colorimetric distance between a prototypical reference "red eye" color and each pixel of the image containing the red eye. Various image processing algorithms such as thresholding, blob analysis, and morphological filtering, are applied to the mask, in order to eliminate noise, reduce errors, and facilitate a more natural looking result. The mask serves to identify pixels in the color image needing correction, and further serves to identify the amount of correction needed. Pixels identified as having red-eye artifacts are modified to a substantially monochrome color, while the bright specular reflection of the eye is preserved.

Introduction

The effect of "red eyes" is a well-known problem in photography. Otherwise good flash photographs are often completely unacceptable because of that glowing red color that often appears in the eyes of people photographed with flash bulbs. Red eyes are caused by light entering the subject's eye through the pupil and reflecting from the retina. The light is usually coming from the flash used when taking the photograph. The reflection is red roughly because the blood in the retina absorbs all colors of the visible spectrum except red, see Figure 1.

Several attempts have been made to reduce this problem at the time the photograph is taken. Two main techniques are used: increasing the distance between the flash and the camera objective, and using one or several "pre-flashes" to reduce the size of the subject's pupils. Despite these efforts, "red eyes" are still, and will likely continue to be, a huge problem in amateur photography.

With the advent of digital imaging technology, new possibilities arise for solving this problem. By applying an innovative combination of color image processing algorithms, we wish to touch-up such photographs by replacing the unwanted red colors by dark neutral hues, in a manner that is simple to the user, and that results in an image that looks natural.

Several commercial imaging software packages propose the function of red eye removal in different forms and with different degrees of success. Examples of these include Microfrontier's Digital Darkroom (http://www.microfrontier.com/products/digital_darkroom10/resr.html), Photodex's CompuPic (http://www.photodex.com/products/cpic/photo.html#red-eye), Microsoft's PhotoDraw (http://mspress.microsoft.com/prod/books/sampchap/2316.htm#101), and Picture It! (http://www.home-publishing.com/PictureIt/default.asp), Adobe PhotoDeluxe (http://www.adobe.com), and Corel's CorelScan (http://www.corel.com).

These manufacturers have to our knowledge not published their algorithms. There is indeed little published material on this subject. The only reference that has come to our attention is a conference presentation[1] given in 1996 by researchers at the University of Toronto. Unfortunately, this abstract seems not to appear in the printed conference proceedings, and to date, we have not been able to consult this publication.

In this paper, we present our approach to this problem in the context of developing a function of a consumer imaging software application, the Photo Center, a component of Conexant's InternetDesktop software.[2] We present several aspects of the design process, such as the choice of a user interaction model and the development of the image processing algorithms.

General Methodology

In this section we first discuss what a "red eye" looks like, and how we would want it to look. Then we examine the question of how the user should interact with the software in order to touch up images, and finally we discuss some considerations related to color space.

Characteristic Features

The actual shape and color of a red eye vary a lot from image to image. The pupil may appear in different shades of red, depending on factors such as flash power, exposure time, camera sensitivities, and if conventional analog photography is used, film and paper type, photographic development process, and finally the digital scanning process including any digital color correction.

The shape of the red pupil is usually almost circular, but also here many variations are found, depending on factors such as image resolution, eye and eyebrow position, focus, and imaging geometry.

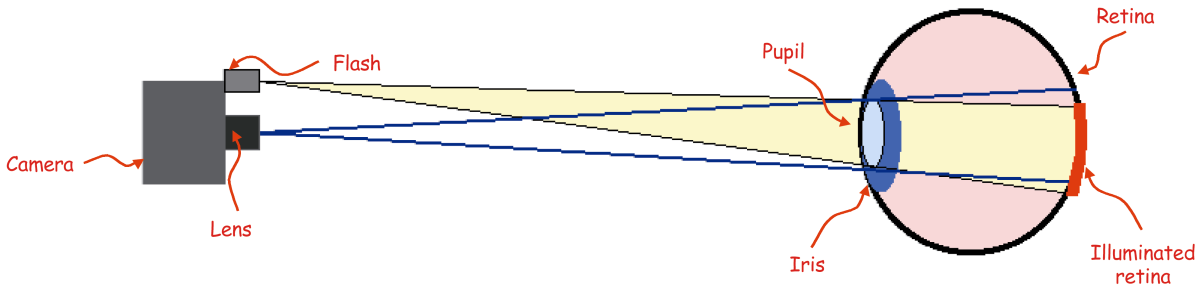

Figure 1. The "Red Eye" effect is caused by incident light, typically from a flash, being reflected from the retina back to the camera.

Another important characteristic feature is the bright specular highlight, the so-called "catch light" which occur as the flashlight is reflected at the surface of the eye.

See Figure 2 for a few examples of details of photographs with red eye artifacts.[3]

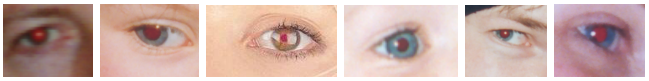

Figure 2. The color, size, and shape of red eyes are very image-dependent.[3]

Desired Appearance

In order to develop efficient algorithms to correct the red eye artifact, it is important to specify the desired appearance of the corrected image. We mention here some key features.

The corrected pupil should be essentially black. A common misconception is that the target color depends on the eye (iris) color of the person, this is obviously not the case.

If there is a specular highlight in the eye, a catch light, it is very important not to remove or significantly alter it. Catch lights represent an essential part of a good portrait, because they give the eyes a life-like quality and project the subject's expression and personality.[4]

A general goal is that the corrected image should look completely natural, one particular challenge to achieving this is to avoid visible boundaries between the corrected and non-corrected areas of the image.

Another important design goal is that elements of the image which is not a red eye artifact should not be affected. It is not acceptable to remove a bathing suit's red polka dots…

User Interface Model

We have identified four different possible user interface models:

1. Manual selection of the red pupil area, typically using tools such as "magic wand", "lasso", or "ellipse." This model is used e.g. in CorelScan and Microfrontier;s Digital Darkroom
2. Sweeping / brushing red areas. to neutralize red hues. This model is used in Photodex's CompuPic
3. Selecting a Region of Interest (ROI) rectangle around the red eye(s). This approach is used e.g. by Adobe PhotoDeluxe.
4. Completely automatic. If there are red eyes in the photo, they will be automatically corrected. We have not found any existing software using this model.

Although the fourth model would be extremely desirable from a user standpoint (if it's guaranteed never to fail), we have chosen model #3. The ROI model represents a reasonable tradeoff between algorithm complexity and user interaction. Figure 3 shows our implementation of a user interface using this model.

Color Space Considerations

As we describe in the following sections, important parts of the proposed image processing algorithm is performed using the CIELAB color space,[2] in particular because of its convenient ability to quantify color in terms of its perceptual attributes – lightness, chroma, and hue. For the understanding of this paper, basic familiarity with the CIELAB color space should be sufficient, in particular that a color is quantified by three values, the lightness L^* and the chromaticity coordinates a^* and b^*. More information about this color space can be found in any good book on color imaging and color management.[6, 7]

However, it is not common that the colors of an image are quantified using this device-independent color space. More typically the colors are quantified in an uncalibrated device-dependent RGB color space, and an exact formula for the colorspace conversion to CIELAB does not exist. Our approach to this problem is to assume that the images are quantified in a standardized RGB space, namely the sRGB color space[8], and use the known formulae to convert between sRGB and CIELAB. More and more imaging devices use the sRGB color space for unambiguous communication of color.[9]

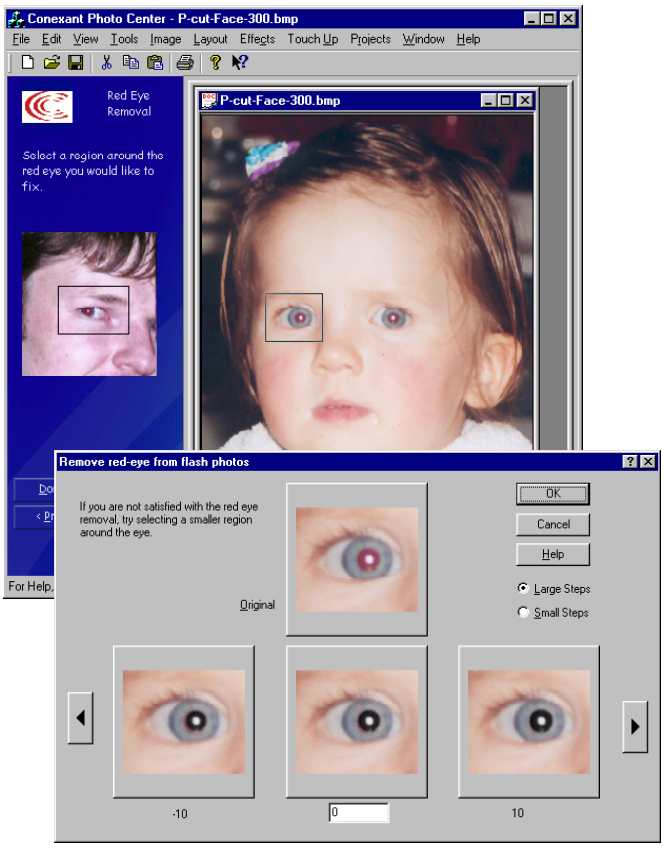

Figure 3. Example of the Red Eye Removal user interface used in our Photo Center software application.[2, 3] First, the Red Eye Removal function is chosen from the TouchUp menu, then the ROI is selected. A new window pops up showing a preview of the correction, and allowing for some parameter modification, in order to achieve the desired results.

Color Image Processing

At this point we suppose that the user has selected a Region of Interest (ROI), that is, a rectangle that contains the red eye to be corrected. Our problem is then to get from this image down to the actual pixels that should be modified, and to decide how these pixels should be modified. Because of the variations of color, size, and shape of the red eye artifact, this is not a trivial task.

The main idea of our approach is to use a fuzzy mask. The mask serves to identify precisely the pixels in the ROI needing correction, and further to identify the amount of correction needed in these pixels.

Our proposed color image processing algorithm method can be broken down into four steps. First, an initial mask is computed over the ROI. Then, this mask is binarized by a thresholding operation. In the third step, the mask is adjusted to fit more closely the actual location of the red eye, by a combination of several image processing techniques. The last step is to apply the actual correction to the areas of the ROI where the mask value is nonzero. These four stages are described in more detail in the following subsections.

Initial Mask Computation

The first step of our algorithm is to compute a 8-bit "redness mask" over the ROI, as illustrated by Figure 4.

For each pixel of the ROI a color difference between the actual pixel color $\mathbf{i}(i,j)$ and a predefined "typical red eye color" $\mathbf{i}_{\text{TypicalRedEye}}$ is calculated. This distance is then normalized such that the mask value $m(i,j)$ is white where the color is most likely to be a red eye, black where this is least likely, and different shades of gray in between. This computation can be expressed as follows:

$$x(i,j) = d(\mathbf{i}(i,j),\ \mathbf{i}_{\text{TypicalRedEye}}) \tag{1}$$

$$m(i,j)\ = \text{round}\left(255 \cdot \frac{\max(x) - x(i,j)}{\max(x) - \min(x)}\right), \tag{2}$$

where $d()$ is a function quantifying the color difference between two pixels. We have evaluated several possibilities for this color difference calculation, including simple Euclidean distance in RGB color space, and CIE ΔE_{ab}. The formula that turned out to give the best results was the chromaticity difference in CIELAB space, that is,

$$d(\mathbf{i}_2, \mathbf{i}_1) = [(a^*_2 - a^*_1)^2 + (b^*_2 - b^*_1)^2]^{1/2} \tag{3}$$

As shown above, the difference in luminance L^* of the pixel in question and that of the reference color are not factored into the color difference equation. Factoring out the luminance appears to have some advantage over calculating the three-dimensional Euclidean distance between the reference color and the color image points, because the luminance of a red-eye artifact typically varies depending on the proximity and intensity of the flash.

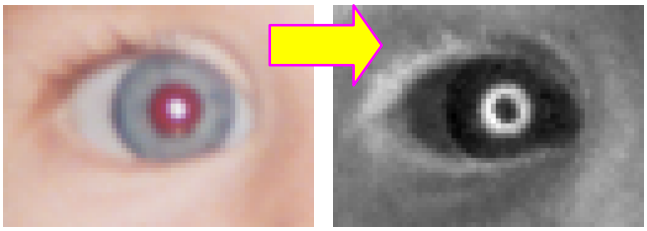

Figure 4. Initial mask computation. The light gray areas identify colors that are likely to be red-eye colors.

Mask Binarization

In the second step, the mask is binarized by a thresholding operation, as shown in Figure 5. This segments the mask in background (black) and object (white), the goal being that the object is the red eye.

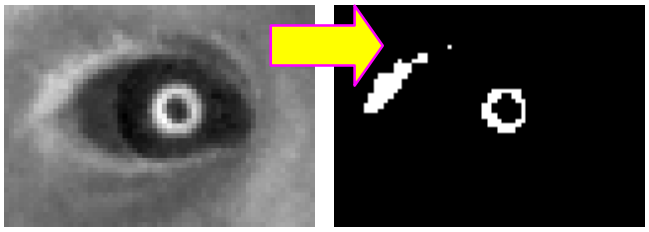

Figure 5. Mask binarization.

The important question in this step is how to determine the threshold level. Several automatic thresholding methods have been evaluated, such as the Mean value, Histogram peaks, Iterative selection, Pun, and Fuzzy methods, described by Parker.[10] These methods were not found to yield satisfactory results. Therefore we chose to simply use static thresholding, with a threshold level empirically set to 175.

Mask Adjustment and Fuzzification

Typically, at this point, the mask does not correctly identify the location of the red eye. The third step of our algorithm is therefore a very important one, in which the binary mask is adjusted to fit more closely the actual location of the red eye (Figure 6). A combination of several image processing techniques are used.

The morphological operations opening and closing are used to remove unwanted "noise" pixels, and to fill "holes," respectively.

A "blob analysis" technique is used to group the pixels of the mask into objects, and the object most likely to be a red eye artifact is selected, based on features such as size and shape.

Some degree of circularity may be imposed on the mask, for example by replacing the object by a circular disc.

Finally the mask is smoothed, or "fuzzified," to achieve a "softer" correction that appears to be more natural. In this operation the mask is converted into an 8-bit representation.

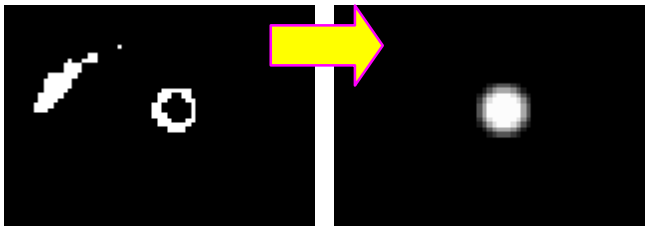

Figure 6. Mask adjustment and fuzzification.

Image Correction

The last step of our algorithm is to apply the actual correction to the areas of the ROI where the mask value is nonzero, as illustrated by Figure 7.

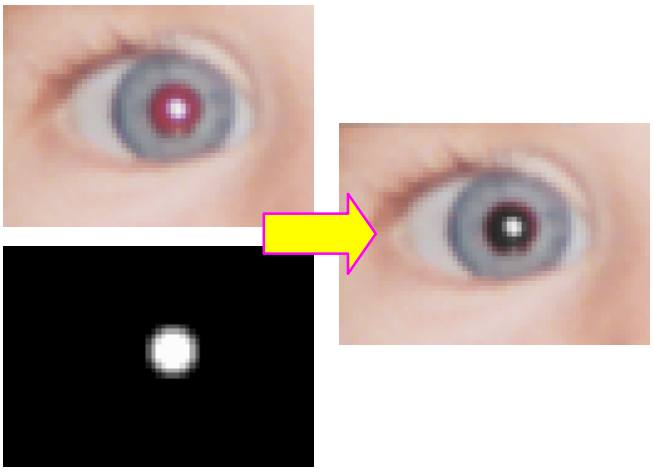

Figure 7. Image correction.

The first step of this correction is to identify the target color for the corrected red eye using the CIELAB color space as follows. The color components a^* and b^* determining the hue and saturation are set to zero, that is, shades of gray. The lightness component L^* is then determined by "stretching" out the lightness of the original image such that its minimum value becomes black (or almost black) and its maximum value remains constant. This calculation can be represented by Eq. (4).

$$\begin{aligned} L^* &\leftarrow \frac{\max L^*}{\max L^* - \min L^*}\left(L^* - \min L^*\right) \\ a^* &\leftarrow 0 \\ b^* &\leftarrow 0 \end{aligned} \tag{4}$$

Finally, the new color for each pixel is determined as a combination of the original color and the target color, by weighting with the fuzzy mask values. Denoting the target CIELAB values of Eq. (4) as $\mathbf{t}(i,j)$, the correction is represented by the following equation:

$$\mathbf{i}(i,j) \leftarrow \mathbf{t}(i,j)m(i,j) + \mathbf{i}(i,j)(1 - m(i,j)) \tag{5}$$

Note that the calculation of Eq. (5) is carried out in CIELAB space, therefore, to complete the correction, the pixels need to be converted back to an RGB representation.

Experimental Results and Discussion

We have applied the proposed algorithm to a series of images containing red eyes, some of which are shown in Color Plate 1. In order to benchmark the performance, we also applied the red eye removal function of Adobe PhotoDeluxe 1.0 Business Edition to the same images.

Our proposed algorithm was found to give visually good results. It is judged to be very competitive in terms of the resulting image quality. Examining the results of Color Plate 1more closely, we make the following observations.

Color Plate 1: Experimental results of the proposed Red Eye Removal algorithm, compared to those obtained using Adobe PhotoDeluxe.

- In several cases, PhotoDeluxe identifies two red eyes, even if there is only one.
- Our method gives less edge artifacts around the corrected area.
- Both algorithms fail on some images.

To further optimize the results, some fine-tuning of the different parameters of the algorithm could be done.

Conclusions and Perspectives

A solution to the problem of removing "red eyes" from digital photographs was proposed. An innovative combination of image processing algorithms permits a robust determination of which pixels need to be modified, and how to modify them. The idea of using the CIELAB color space to specify the corrections enables to ensure that the specular reflection of the flash, which is often a very important feature of the image, remains at the same level of intensity in the corrected image, while the unwanted reddish hues are removed.

To go one step further in automating this process, it would be possible to combine the presented method with known techniques of automatic face detection in digital images, such as for example those proposed in the references 11-13. This combination would result in a completely automatic removal of red eyes, and could be of great potential, especially in the quickly growing market of amateur digital photography.

References

1. K. K. Y. Au, K. N. Plataniotis, and A. N. Venetsanopoulos, *Red-eye Removal by Colour Image Processing*, Proceedings of the 2nd World Congress of Nonlinear Analysts, Athens, Greece, July 10-17, 1996.
2. For more information about Conexant's imaging software, refer to http://www.conexant.com or send an email to rdc.info@conexant.com.
3. Note that most of the figures in this paper are best appreciated in color. Consult IS&T's digital library at http://www.imaging.org, or the author's web page at http://color.hardeberg.com in order to receive an electronic color version of this manuscript.
4. J. Zeltsman, *Zeltsman Approach to Formal Classic Portraiture,* http://www.zuga.net/zuga/contributors/zeltsman/body.htm
5. International Commission on Illumination, *Colorimetry*, CIE Publications 15.2, Vienna, Austria, 1986.
6. E, J, Giorgianni and T. E. Madden*, Digital Color Management – Encoding Solutions,* Addison-Wesley, 1998
7. R. W. G. Hunt, *The Reproduction of Colour,* Fountain Press, 5th Ed., 1995
8. G. Starkweather. *Colorspace interchange using sRGB*, 1998. This White Paper and other information about the sRGB colorspace is available at http://www.srgb.com.
9. J. Y. Hardeberg, *Desktop scanning to sRGB,* In Device Independent Color, Color Hardcopy and Graphic Arts V, Proc. SPIE 3963, pp. 47-57, 2000
10. J. R. Parker, *Algorithms for Image Processing and Computer Vision,* Wiley, 1997
11. H. A. Rowley, Shumeet Baluja, and Takeo Kanade, *Neural Network-Based Face Detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 20, number 1, pages 23-38, January 1998.
12. H. A. Rowley, *Neural Network-Based Face Detection*, Ph.D. dissertation, Carnegie Mellon University, CMU-CS-99-117, May 1999
13. K.-K. Sung and T. Poggio, *Example-Based Learning for View-Based Human Face Detection*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No. 1, January 1998

Biography

Jon Y. Hardeberg received his sivilingeniør (M.Sc.) degree in signal processing from the Norwegian Institute of Technology (Trondheim, Norway) in 1995. He received his Ph.D. from the Ecole Nationale Supérieure des Télécommunications (Paris, France) in February 1999. His Ph.D. research concerned color image acquisition and reproduction, with applications in facsimile, fine-art paintings, and multi-spectral imaging. He is currently employed with Conexant Systems, Inc., where he designs, implements, and evaluates color imaging system solutions for multifunction peripherals and other imaging devices. His professional memberships include if IS&T, SPIE, and ISCC.