Digital Red Eye Removal

Jon Y. Hardeberg▲

Gjøvik University College. Gjøvik, Norway

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.

Journal of Imaging Science and Technology 46: 375–379 (2002)

Introduction

Red eye artifacts are a well-known problem in photography. Otherwise good flash photographs are often unacceptable because of a glowing red color that appears in the eyes of people photographed with flash lamps. This is particularly the case with photographs of babies and children. Red eye artifacts 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 because the retina absorbs all colors of the visible spectrum except red, see Fig. 1.

Several techniques can be used to reduce this problem at the time the photograph is taken. By increasing the level of light in the room, or by having the subject look at a bright light just before the picture is taken, the subject’s pupils will contract, and the reflective surface causing the red eye will be reduced. The same effect is also achieved by using one or several “pre-flashes” immediately before the picture is taken. These solutions, however, are not satisfactory because red eye artifacts, although usually less pronounced, typically remain. Furthermore, pictures taken with multiple pre-flashes consume more power than those taken with a single flash and require more time to take a picture.

Another, more effective solution is to increase the distance between the lens and the flash, see Fig. 2. However, because of the consumer market request for more compact cameras, the applicability of this solution is also limited. Therefore, red eye artifacts continue to be a significant problem in amateur photography.

With the advent of digital imaging technology, new possibilities arise for solving this problem, using digital image processing. Several commercial imaging software packages have offered the function of red eye removal in different forms and with different degrees of success.[1a] Outside of the patent literature, however, there is not much published on this subject. Prior to this author’s work,[1] only two conference presentations have been found,[2] one of which[2a] seems not to appear in the printed conference proceedings.

Possibly the first published image processing algorithm for red eye removal was proposed by Dobbs and Goodwin[3] of Eastman Kodak Company. In their algorithm, the user first has to zoom in on the picture and select the pixel that “best represents” the red eye artifact to be corrected. Then, neighboring pixels are modified to remove the red hue if they fall within a “discriminator” ellipse defined in the chrominance plane of the YIQ color space. The correction is weighed according to a function of the distance from the edge of the ellipse, so as to avoid sharp unwanted edges around the correction area. The size of the discriminator ellipse, as well as the target color for the correction, must be specified interactively by the user, if the default values do not give a satisfying result.

Later, Benati and co-workers[4,5] of the same company propose a new method, in which the level of user interactivity is reduced. Within a region of interest in the image defined by the user, a thresholding operation is first done in the HLS color space, in order to identify candidate pixels for correction. Then, the candidate pixels are grouped into one or more spatially contiguous groups. The group containing the red eye is then sought identified by a process where first all the pixels in the groups are given “scores” based on their color as well as on the shape of the group they belong to, then a region growth algorithm is applied to each group to improve

Original manuscript received May 24, 2001

▲ IS&T Member

Color Plates 1 through 6 are printed in the color plate section of this issue, pp. 380–381.

E-mail: jon.hardeberg@hig.no, jon@hardeberg.com

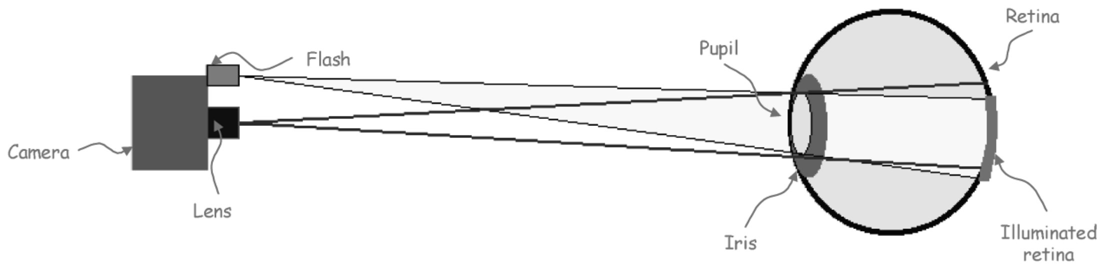


Figure 1. The "Red Eye" effect is caused by incident light, typically from a flash, being reflected from the retina back to the camera.

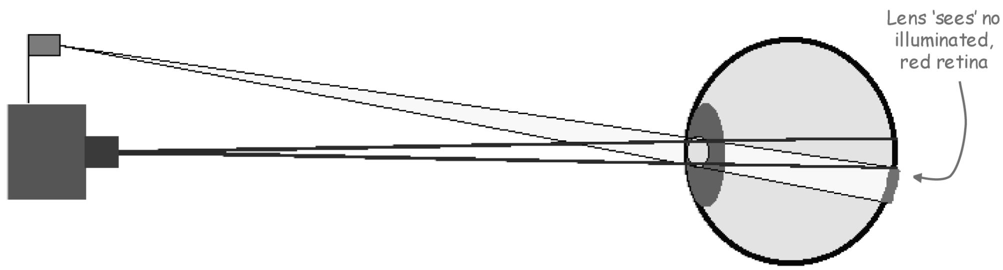


Figure 2. Illustration of two traditional techniques used to reduce the red eye effect: Increasing the distance between the lens and the flash, and reducing the size of the pupil, typically by issuing one or several "pre-flashes".

their shapes, and finally, a second similar score is calculated. The group with the highest score is assumed to contain the red eye, which is then corrected roughly by removing the chromaticity and darkening the luminance using the YCC color space.

More recently, Patti and co-workers[2b,6] of Hewlett-Packard Company propose a three-step algorithm to correct red eye artifacts. First, a binary mask is created using a segmentation algorithm, then the pupil is located in the mask based on a circularity criterion, and finally the color of the eye is changed. The mask creation step uses an adaptive thresholding algorithm applied to the Cr component in a modified YCrCb color space, followed by a morphological filtering operation applied to "clean up" the mask. The pupil location step applies an exhaustive search block-matching estimation scheme to the mask, in order to locate the pupil. Note that a square pupil model was preferred over a round one, in order to speed up calculations. The last step of their method first refines the shape of the mask to better match the actual red eye artifact. The refining is done by using a region growing approach, in which the mask region is expanded into a given neighboring pixel if both its red value and its red-green difference value do not change too much compared to the median value of the same quantity over the mask area. Finally, the pixels belonging to the mask is corrected by replacing them with a gray value of 0.8 times their original value.

In this paper, we present our approach to this problem in the context of developing a function of a consumer imaging software application.[7] 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. First, in the following section, we present our general methodology, including several aspects of the design process, such as the choice of a user interaction model. In the subsequent section we propose a color image processing algorithm to remove the red eye artifact. Then we report and discuss our experimental results, and finally, we provide some conclusions and ideas for future work on this subject.

General Methodology

In this section we first discuss the characteristic features of a red eye artifact, i.e., what it looks like, followed by a discussion of its desired appearance, i.e., 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 artifact vary considerably 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.

Another important characteristic feature is the bright specular highlight, the so-called "catch light" which occurs as the flashlight is reflected at the surface of the eye. See **Color Plate 5** (p. 380) for a few examples of details of photographs with red eye artifacts.

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.[8]

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 uncorrected areas of the image.

Another important design goal is that details of the image which are not red eye artifacts should not be modified. If this should occur, it could have dramatic negative implications on the image quality.

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 Corel Corporation's CorelScan™ and in 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 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 guaranteed never to fail), we have chosen model #3. The ROI model represents a reasonable tradeoff between algorithm complexity and level of user interaction.

Color Space Considerations

As we describe in the following sections, important parts of the proposed image processing algorithm are performed using the CIELAB color space.[9] The main rationale for using this color space is its convenient ability to quantify color in terms of its perceptual attributes (lightness, chroma, and hue), along with its pseudo-uniformity. Other color spaces, such as YIQ, HLS, and YCC, used for red eye removal by other authors,[2b,3–] are also claimed to have the ability to separate lightness information from chroma/hue information, but their success in doing so is much lower than that of CIELAB. Thus, concurring with several other researchers,[10–14] we have found the use of CIELAB to be advantageous for image processing applications.

For the understanding of this article, basic familiarity with the CIELAB color space should be sufficient, in particular that a color is quantified by three values; the lightness by L^* and the chromaticity by a^* and b^*. More information about this color space can be found in standard references on color and imaging, e.g., Giorgianni and Madden[15] or Hunt.[16]

However, it should be noted that 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,[17] and to use known formulae[18] to convert between sRGB and CIELAB. More and more imaging devices and systems use the sRGB color space for unambiguous communication of color images,[19] therefore this assumption makes sense. In a "color-managed" imaging environment compliant with the ICC standard,[20] a device profile associated with the image will describe the colorspace conversion to CIELAB.

Proposed Color Image Processing Algorithm

At this point we suppose that the user has selected a Region of Interest (ROI), that is, a rectangle that contains the red eye artifact 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 steps are described in more detail in the following subsections.

Initial Mask Computation

The first step of our algorithm is to compute a "redness mask" over the ROI, as illustrated by **Color Plate 1** (p. 380).

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" $\mathbf{i}_{TypicalRedEye}$ is calculated. By analyzing a significant number of images containing red eye artifacts, the value of $\mathbf{i}_{TypicalRedEye}$ has been set to $(R, G, B) = (150, 50, 50)$. 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:

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

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

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}.[9] The formula that turned out to give the best results was the chromaticity difference in CIELAB space, that is,

$$d(\boldsymbol{i}_2, \boldsymbol{i}_1) = \left[\left(a_2^* - a_1^*\right)^2 + \left(b_2^* - b_1^*\right)^2\right]^{1/2}. \tag{3}$$

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.

Mask Binarization

In the second step, the mask is binarized by a thresholding operation, as shown in **Color Plate 2** (p. 380). This segments the mask in background (black) and object (white), the goal being that the object is the red eye.

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.[21] These methods were not found to yield satisfactory results. Therefore we chose to use static thresholding, with a threshold level empirically set to 175 (assuming that the mask values have been normalized to values between 0 and 255).

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 as shown in **Color Plate 3** (p. 380). A combination of several image processing techniques are used, as described in the following paragraphs.

The morphological operations of opening and closing are first applied to the binary mask. The opening operation smoothes the contours of the mask, breaks narrow isthmuses, and eliminates thin protrusions. The closing operation smoothes sections and contours, fuses narrow breaks in long thin gulfs, eliminates small holes, and fills gaps in contours. In practice an opening with a structuring element of size 1/30th of the width of the ROI is performed, followed by a closing with a structuring element of size 1/10th of the ROI width.

A blob analysis technique is used to group the pixels of the mask into 8-connected components. After a complete set of blobs or connected components have been detected, the particular blob corresponding to the artifact is selected, and the remainder eliminated from the mask. Typically, a pattern recognition routine is used to choose the blob that has the highest probability of representing a red eye artifact, based on its size and shape. In our experience, the simple approach of selecting the blob having the width-to-height aspect ratio nearest to 1, and having a size at least 1/20th of the size of the ROI, has worked well.

Next, some degree of circularity may be imposed on the mask, for example by replacing the selected blob with a circular blob having a center corresponding to the center of the blob and a diameter equal to the greater of the blob's height and width.

Finally the mask is smoothed, or "fuzzified," to achieve a "softer" correction that appears to be more natural. Prior to the smoothing operation, the mask pixels are limited to two values, corresponding to black and white. The smoothing introduces intermediate gray levels back into the mask, so that the edges of the blob corresponding to the red eye artifact have varying shades of gray (8 bit representation). The gray levels moderate the extent to which various color characteristics of corresponding pixels in the original image are modified to correct the red eye artifact. We use a 2D averaging convolution filter with a kernel size of whichever is greater of 3 pixels and one fifth of the estimated pupil diameter. This process is shown in **Color Plate 4** (p. 380).

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 **Color Plate 5** (p. 380).

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 chrominance 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 within the mask area becomes black (or almost black) and its maximum value remains constant. This calculation can be represented by Eq. (4).

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

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 6** (p. 381). 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 6** (p. 381) more closely, we make the following observations.

- In several cases, PhotoDeluxe identifies two red eyes, even if there is only one. Although this clearly results in very objectionable results with PhotoDeluxe for these cases, it should be noted that this is rather attributable to a difference in the user interface model, than to a different performances of the image processing algorithms.
- Our method gives generally less edge artifacts around the corrected area, resulting in more natural appearance of the corrected images.
- Both algorithms fail on some images.
- For images of very low spatial resolution, our method generally outperforms PhotoDeluxe.

To further optimize the results, some fine tuning of the different parameters of the algorithm could be done. It would also be conceivable to replace one or several of the image processing steps with alternative algorithms of better performance, in order to increase the quality of the result, and/or to increase the processing speed.[1b]

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.

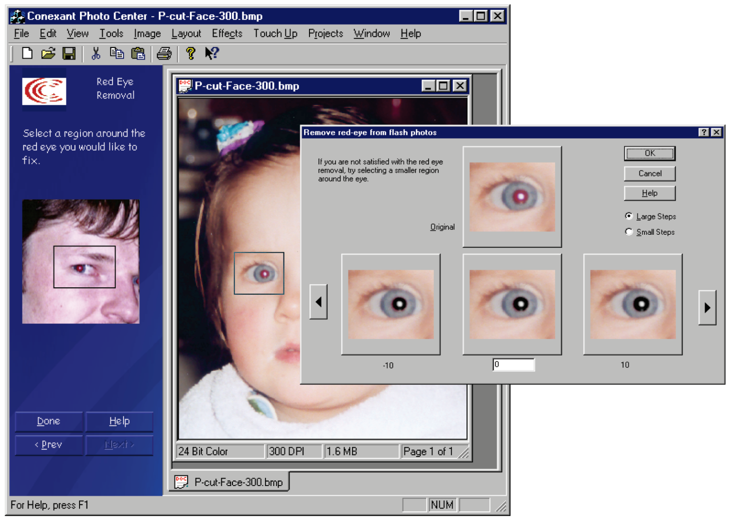
With only minor modifications, the proposed method could be used for correction of similar retinal-reflection artifacts found in pictures of various animals. Different pupil shapes and different artifact colors would have to be considered.

To go one step further in automating this process, it would be very interesting to combine the described method with known techniques of automatic face detection in digital images, such as for example those proposed in Refs. 22–25. This combination would result in a completely automatic removal of red eye artifacts, and could be of great potential, especially in the quickly growing market of digital photography. It would be conceivable to implement such algorithms embedded on the camera controller chip, in the software supporting online digital photofinishing services, and in digital mini-laboratories.

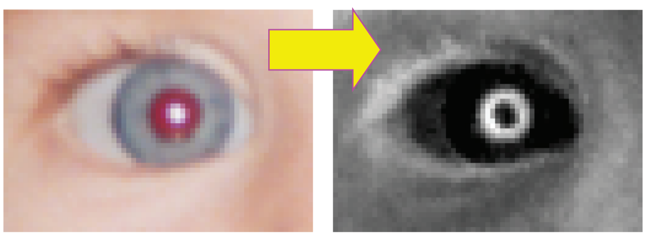
Acknowledgment. This work was performed while the author was employed with Conexant Systems/ViewAhead Technology. The participation of my former colleagues by providing images with red eye artifacts and being discussion partners in the development of the algorithms, and of the management by facilitating this paper, is highly appreciated.

References

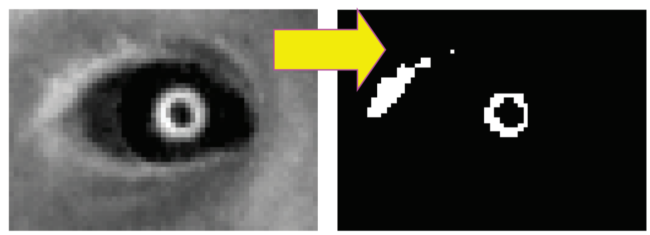
1. (a) J. Y. Hardeberg, Red Eye Removal using Digital Color Image Processing, in *Proceedings of IS&T's Image Processing, Image Quality, Image Capture, Systems Conference*, IS&T, Springfield, VA, 2001, pp. 283–287; (b) K. Czubin, B. Smolka, M. Szczepanski, J. Y. Hardeberg, and K. N. Plataniotis, On the red eye effect removal algorithm, *Proceedings of CGIV2002—First European Conference on Color in Graphics, Image and Vision*, IS&T, Springfield, VA, 2002, pp. 292–297.
2. (a) K. K. Y. Au, K. N. Plataniotis, and A. N. Venetsanopoulos, Red eye Removal by Colour Image Processing, *2nd World Congress of Nonlinear Analysts*, Athens, Greece, 1996, unpublished; (b) A. Patti, K. Konstantinides, D. Tretter, and Q. Lin, Automatic Digital Redeye Reduction, *Proceedings of ICIP,* IEEE, New York, NY, **3**, 1998, pp. 55-59.
3. C. M. Dobbs and R. M. Goodwin, Localized image recoloring using ellipsoid boundary function, U.S. Patent 5,130,789, (1992).
4. P. J. Benati, R. T. Gray, and P. A. Cosgrove, Automated detection and correction of eye color defects due to flash illumination, U.S. Patent 5,432,863 (1995).
5. P. J. Benati, R. T. Gray, and P. A. Cosgrove, Automated detection and correction of eye color defects due to flash illumination, US Patent 5,748,764 (1998).
6. Q. Lin, D.R. Tretter, K. Konstantinides, and A. Patti, Apparatus and a method for reducing red eye in a digital image, US Patent 6,016,354 (2000).
7. ViewAhead Photo Center is a component of ViewAhead's InternetDesktop software. For more information about ViewAhead's imaging software, refer to http://www.viewahead.com.
8. J. Zeltsman, *Zeltsman Approach to Formal Classic Portraiture*, Available online at http://www.zuga.net/zuga/contributors/zeltsman/body.htm.
9. International Commission on Illumination, *Colorimetry*, CIE Publications 15.2, Vienna, Austria, 1986.
10. C. Vertan, N. Boujemaa, and V. Buzuloiu, A Fuzzy Color Credibility Approach to Color Image Filtering, in *Proceedings of 2000 International Conference on Image Processing,* **2**, 808 (2000).
11. D. Paulus, K. Horecki, and K. Wojciechowski, Localization of Colored Objects, in *Proceedings of 2000 International Conference on Image Processing,* **3**, 492 (2000).
12. A. Trémeau, P. Colantoni and V. Lozano, Quality of color image filtering techniques, in *Proceedings of the Sixth Color Imaging Conference: Color Science, Systems, and Applications,* IS&T, Springfield, VA, 1998, pp. 135–139.
13. G. Dardier, J. Y. Hardeberg, and H. Brettel, Enhancement of Improperly Exposed Photographic Color Negatives, *SPIE Proceedings* **3648,** 411–423 (1999).
14. G. Paschos, Perceptually Uniform Color Spaces for Color Texture Analysis: An Empirical Evaluation, *IEEE Transactions on Image Processing*, **10**(6), 932–937 (2001).
15. E. J. Giorgianni and T. E. Madden, *Digital Color Management – Encoding Solutions*, Addison-Wesley Longman, Reading, Massachusetts, USA, 1998.
16. R. W. G. Hunt, *The Reproduction of Colour*, 5th ed., Fountain Press, Kingston-upon-Thames, England, 1995.
17. G. Starkweather, *Colorspace interchange using sRGB*, White Paper available online at http://www.srgb.com/sRGBColorSpacePaper.pdf, 1998.
18. M. Nielsen, *Structure of the sRGB profile*, available online at http://www.srgb.com/hpsrgbprof/index.htm, 1998.
19. J. Y. Hardeberg, Desktop scanning to sRGB, *SPIE Proc.* **3963,** 47–57 (2000).
20. International Color Consortium, *File Format for Color Profiles,* Specification ICC.1:2001-04, (2001), see http://www.color.org/.
21. J. R. Parker, *Algorithms for Image Processing and Computer Vision*, John Wiley & Sons, New York, 1997.
22. H. A. Rowley, S. Baluja, and T. Kanade, Neural Network-Based Face Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20,** 23–38 (1998).
23. K.-K. Sung and T. Poggio, Example-Based Learning for View-Based Human Face Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **20,** 39–51 (1998).
24. H. Schneiderman and T. Kanade, A Histogram-Based Method for Detection of Faces and Cars, in *Proceedings of 2000 International Conference on Image Processing,* **3,** 504 (2000).
25. T. Kawaguchi, D. Hidaka, and M. Rizan, Detection of Eyes from Human Faces by Hough Transform and Separability Filter, in *Proceedings of ICIP 2000,* **1,** 49 (2000).



Color Plate 1. Example of the Red Eye Removal user interface used in Conexant/ViewAhead's Photo Cnetersoftware application.[3] Ffirst, the Red Eye Removal function is chosen form 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. *(Hardeberg, pp. 375–379)*

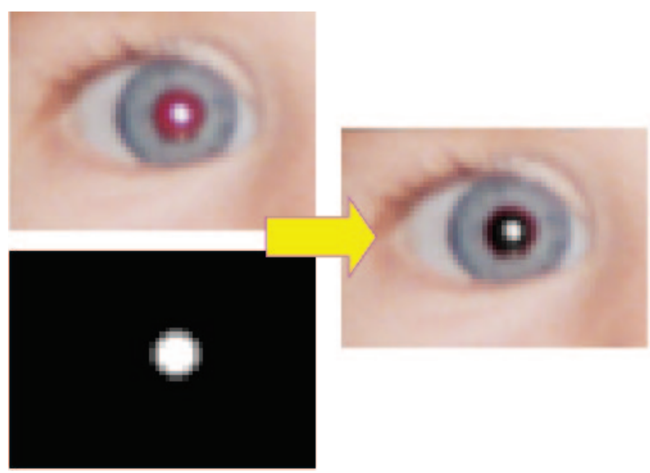


Color Plate 2. STEP 1 – Initial mask computation. The white and light gray areas identify areas that are likely to belong to a red-eye artifact, as judged by their color. *(Hardeberg, pp. 375–379)*

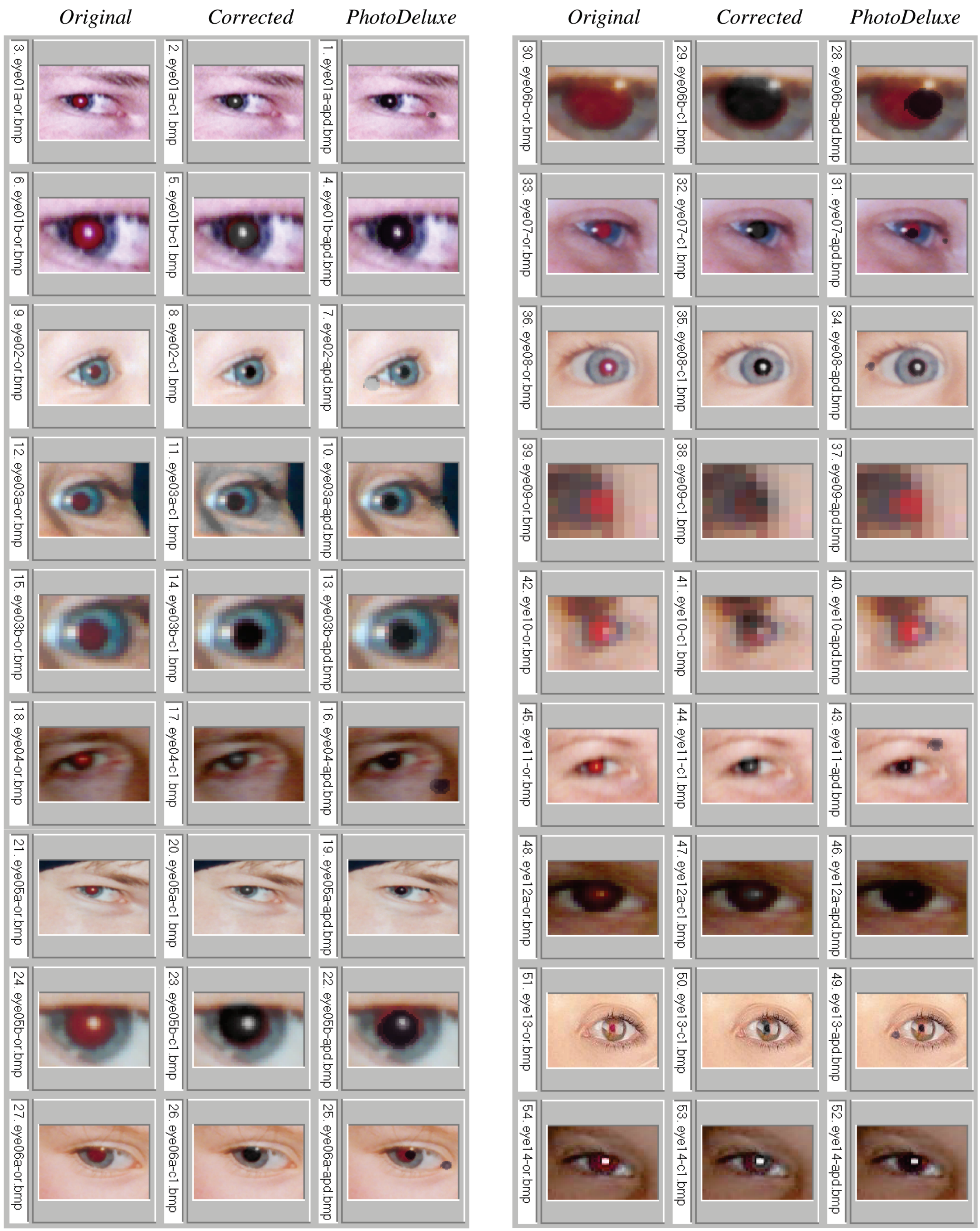


Color Plate 3. STEP 2 – Mask binarization. By a thresholding operation, the grayscale mask is reduced to a binary mask in which the color white represents areas being candidates for containing red-eye artifacts. *(Hardeberg, pp. 375–379)*

Color Plate 4. STEP 3 – Mask adjustment and fuzzification. Unwanted elements of the mask is removed by applying morphological operations and blob analysis techniques, some degree of circularity is imposed, and smoothing is applied. *(Hardeberg, pp. 375–379)*



Color Plate 5. STEP 4— Image correction. The values of the fuzzy mask weights the amount of applied correction, thereby achieving a soft edge between corrected and non-corrected pixels. The red-eye artifact is removed, giving the eye a natural appearance. *(Hardeberg, pp. 375–379)*



Color Plate 6. Experimental results of the proposed Red Eye Removal algorithm, compared to those obtained using Adobe PhotoDeluxe. *(Hardeberg, pp. 375–379)*