Journal of Electronic Imaging 13(1), 48–57 (January 2004).

Retinex in MATLAB™

Brian Funt
Florian Ciurea
Simon Fraser University
School of Computing Science
Burnaby, British Columbia, V5A 1S6
Canada

John McCann
McCann Imaging
Belmont, Massachusetts 02478

Abstract. *Many different descriptions of Retinex methods of lightness computation exist. We provide concise MATLAB™ implementations of two of the spatial techniques of making pixel comparisons. The code is presented, along with test results on several images and a discussion of the results. We also discuss the calibration of input images and the postRetinex processing required to display the output images.* © *2004 SPIE and IS&T.* [DOI: 10.1117/1.1636761]

1 Introduction

The Retinex model for the computation of lightness was introduced by Land and McCann.[1] McCann refers to these models as ratio-product-reset average, but for simplicity, we call these operations the Retinex model. Frankle and McCann provide complete FORTRAN code for their algorithm, with extensive discussion of image processing steps that follow spatial comparisons. Since that time, Land and his colleagues have described several variants on the original method.[2–6] The variants on Retinex mainly aim to improve the computational efficiency of the model, while preserving its basic underlying principles.

Retinex calculations aim to predict the sensory response of lightness. It is important to distinguish between physical reflectance, the sensation of lightness, and perceived reflectance, which are three distinct entities. A single model can attempt to calculate only one of the three: the Retinex goal is to calculate the sensation of lightness. Consider the case of two faces of a white cube, one in direct sunlight and the other in shadow. Physical reflectance is a measure of a property of the cube's surface relating its radiance to its irradiance.

The reflectances of the two faces are identical. Sensations, on the other hand, are the appearances of the faces of the cube in the sun and shade. To create the same appearances in a painting, a fine art painter would mix white with a little yellow to make the sunny face, but use white with blue and a little black to reproduce the appearance of the shadowed face. These samples of different colored paints are measures of sensation. Here the two faces are different.[7] In comparison, the question of the perceived reflectances of the cube's surfaces involves cognition. It asks the observer to recognize the paint on the cube. Asked to repaint the cube, the observer is not confused by sun and shade, and would simply apply white paint. In terms of perception, the two faces of the cube are identical. In contrast, Retinex calculates lightness sensations: it cannot be used to calculate physical reflectances or perceived reflectances.

The first model designed to calculate lightness was described in Land's Ives Medal Address to the Optical Society of America in 1968 and later published.[1] This lecture included a working demonstration of a primitive electronic Retinex camera. This was followed by publications and patents with additional details and improvements.[4,8,9] McCann, McKee, and Taylor[3] described a study of human color constancy that included color-matching experiments, the details of the lightness model, and successful results of modeling the experimental data. This result was further developed to show that there is no effect of cone pigment adaptation in color constancy.[10] The Retinex operators were selected for simplicity to mimic biological operators that sum, difference, and rectify input signals to obtain spatial interactions.

Dynamic range compression of real images was described in a patent by Frankle and McCann.[2] This implementation used specialized hardware (International Imaging Systems I^2S image processor with scrollable 8-bit image planes) for efficient image calculation. It described the idea that information from 2^n pixels is accumulated after n steps of the process. This patent also described the multiresolution approach to Retinex calculation used for computer applications.[6,11]

2 Appropriate Input Data

For quantitative testing of the Retinex model, it is crucial that the data be calibrated in the sense that the image digital values must be a logarithmic function of scene radiance, and they must be represented with sufficient precision. McCann[12] used slope 1.0 photographic film to capture real

Paper RTX-04 received Mar. 4, 2003; revised manuscript received Sep. 15, 2003; accepted for publication Oct. 13, 2003.
1017-9909/2004/$15.00 © 2004 SPIE and IS&T.

Table 1 This table describes the care one must take in preparing input images. The data comes from the image of two test targets: one in sun, the other in shade (see Fig. 1). The shade reduced the illumination such that white paper in the shade sends the same radiance to the eye as the black paper in the sun. The left box demonstrates the digitization of raw image data as equally spaced $\log_{10}$ increments. In other words, convert the scene into log radiance and then quantize to 8-bit (0 to 255) digits (log then quantize). The first column specifies either sun or shade illumination. The second column describes the papers in the grayscale. The third column lists the scene radiance from the two identical grayscales in sun and in shade. Note that the radiance from the black in the sun is equal to that from the white in the shade. The fourth column, in the left box, lists log radiances of scene radiance values (column 3). The fifth lists the 8-bit Quantized Log Digits for the values in column 4. Quantizing the log image makes equal log increments with equal differences 0.45 log units in radiance. That means each digit represents radiance ratio steps of 1.0321. The right box demonstrates problems arising from quantizing before converting to log. The sixth column (right box) lists the 8-bit quantized linear digit. The seventh column lists the log quantized digit. This segments the image into equal linear increments, namely equal radiance differences of 13.3971. The consequence of this is that all radiance values for Black, Dark gray 4, and Mid gray 3 are all represented by the same digit, 0. In other words, quantizing the input image to digits shows poor use of digits. Following quantization with a $\log_{10}$ transform does not improve the image. Representing radiances of the input image as log quantized digit (log then quantize) makes a suitable input image for studying high dynamic range images. Using log quantized digits (quantize then log) makes a highly undesirable input image.

	Paper	Scene radiance	Log radiance	Quantized log digit	Quantized linear digit	Log quantized digit
Sun	White	3162	3.50	255	255	255
	Light gray 1	1412	3.15	229	114	229
	Gray 2	631	2.80	204	51	204
	Mid gray 3	282	2.45	178	23	179
	Dark gray 4	126	2.10	153	10	152
	Black	56	1.75	127	5	131
Shade	White	56	1.75	127	5	131
	Light gray 1	25	1.40	102	2	102
	Gray 2	11	1.05	76	1	80
	Mid gray 3	5	0.70	51	0	0
	Dark gray 4	2	0.35	25	0	0
	Black	1	0.00	0	0	0

images (Ektachrome 5071 slide duplicating film). He was able to measure an in-camera dynamic range of 3.5 log units. The importance of the logarithmic function follows from Wallach's experiments on appearance.[13] He showed that equal radiance ratios generate equal lightness differences. A pair of papers, i.e., a 20% gray paper and a 100% white paper, have the same lightness difference in sun and shade. The pair also has a $\log_{10}$ edge difference of 0.7, regardless of illumination. If the input image data deviates from logarithmic, then the $\log_{10}$ edge difference for these papers will change with illumination, and the calculated lightness difference of the pair will change. For Retinex to work well, edge ratios, or $\log_{10}$ differences, within an object must be independent of illumination. Accurate logarithmic calibration guarantees this to be the case.

The need for sufficient precision can be demonstrated by comparing two routes to the same scaling of an image. In one, we convert raw data to log radiance and then quantize to 8-bit $\log_{10}$ digits. This represents the image well (see Table 1). In the other example, we quantize raw data to 8-bit linear and then take the log. The 8-bit quantization stage truncates the information severely. Mid gray, dark gray, and black are all represented by the same digit (see Table 1, right columns). One cannot take an existing 8-bit image, apply a log to it, and have meaningful input image data for quantitative testing of the Retinex model.

Nevertheless, Retinex often enhances random images that have unknown and unknowable radiances for inputs.[2,14,15] The process improves the visibility of dark objects while maintaining the visual discrimination of the light areas. Unlike lookup tables, which improve one range of radiance at the expense of others, Retinex improves visual differentiation in all ranges of radiances. The danger is that artifacts such as noise create artificial edge information that is enhanced by Retinex processing. The ability to bring out shadow detail is limited by image noise.

3 Retinex Operators

The original Land and McCann work[1] described four steps for each iteration of a Retinex calculation: ratio, product, reset, and average.[16] With the exception of reset,[5] these operators have remained the same over the years. These operators are iteratively applied to an image, but the manner in which they are applied has varied. The focus of this work is to list specific details of how these four operators are applied to the image.

A fundamental concept behind Retinex computation of lightness at a given image pixel is the comparison of the pixel's value to that of other pixels. The main difference between the Retinex algorithms is the way in which the

other comparison pixels are chosen, including the order in which they are chosen. They use the same calculations but have dramatically different computational efficiencies in dealing with large real images. The original way of defining comparisons is by following a path, or set of paths, from pixel to neighboring pixel through the image.[1] Lightness estimates are accumulated along the path in a sequential product (SP). SP starts as 1 and then is modified by multiplying it with the ratio of the next pair of pixels along the path. In the case of the path following, path length affects the results substantially. Short paths mean the comparison is made only to others in a spatially localized group of pixels. Intermediate path lengths are to be used when modeling human vision. Infinite path lengths result in a degenerate case, in which the output image is simply a scaled version of the input image. Infinite path lengths should not be used to model vision.[17,18]

A reset step is a second important feature of Retinex. Each time a comparison is made, the SP is tested; if it exceeds 1.0, it is reset to 1.0. In this case, the value 1.0 becomes the current lightness estimate. A third aspect of Retinex is the way in which lightness estimates obtained from different paths to a pixel are combined. In earlier versions, Retinex also included a thresholding step. However, it is not included in later versions[6] and is not part of the MATLAB™ implementations shown later. The fourth-step averages present values of the product with previous ones.

4 Implementations

We have chosen two versions of Retinex to implement. The first is a computer-based version described by McCann,[6] which we refer to as McCann99 Retinex (see Fig. 6). The second is an older specialized-hardware version,[2] which we call Frankle-McCann Retinex. The two versions both replace the path following with more computationally efficient spatial comparisons. McCann99 Retinex creates a multiresolution pyramid from the input by averaging image data. It begins the pixel comparisons at the most highly averaged or top level of the pyramid. After computing lightness on the image at a reduced resolution, the resulting lightness values are propagated down, by pixel replication, to the pyramid's next level as initial lightness estimates at that level. Further pixel comparisons refine the lightness estimates at the higher resolution level, and then those new lightness estimates are again propagated down a level in the pyramid. This process continues until new products have been computed for the pyramid's bottom level.

In comparison, Frankle-McCann Retinex uses single pixel comparisons with variable separations. An important difference between this method and that described in Land and McCann[1] is that there are no paths. A single pixel eventually averages different products from all other pixels. The advantage of this structure, and also for the multiresolution approach, is that long-distance interactions are propagated with fewer comparisons.

4.1 *McCann99 Multilevel Retinex Details*

For this implementation, the input images must be of dimension $w \cdot 2^n \times h \cdot 2^n$, where $w \geq h$ and w and h are integers in the range [1,5]. This constraint arises from the fact that each level of the image pyramid differs from previous levels by a factor of 2 in each dimension. It is not a serious limitation in practice.

The algorithm assumes that input digits are proportional to the logarithm of scene radiance and are of meaningful precision. Using logarithms simplifies the computation of radiance ratios, which become simple differences. It also implies that when results from different spatial comparisons are averaged, the averaging is in log space and hence equivalent to a geometric mean.

In the first step, the log image is averaged down to the lowest resolution level, which, depending on the input dimensions, will be of the size 1×1, 1×2, 1×3, 2×3, 3×4, 3×5, 4×5, or 5×5. At each step, the resolution level will be doubled. The number of layers in the pyramid depends on the size of the input image. The number of layers will be the greatest power of 2, dividing both the width and height of the input images as calculated by the function ComputeSteps.

When the results (called new products) at one level of dimension $n\text{-}\times\text{-}m$ have been computed, the values are then replicated to form an old product image of dimension $2n\text{-}\times\text{-}2m$. In our implementation, we pad the old product image with zeroes to simplify handling boundary conditions. These extra pixels are discarded at the end of the computation.

At all levels, the new product, a precursor of calculated lightness, for each pixel is computed by visiting each of its eight immediately neighboring pixels in clockwise order. Each visit involves a ratio-product-reset-average operation,[6] which is implemented by the function CompareWithNeighbor. It subtracts the neighbor's log luminance (the ratio step), and then adds the result to the old product (the product step). If the result exceeds the maximum defined by Maximum, it is reset to Maximum (the reset step). Finally, the new product for the pixel obtained by comparison to its neighbor is averaged with the previous old product.

A crucial parameter to the McCann99 algorithm is the number of times a pixel's neighbors are to be visited. In the code, this is set by nIterations. It controls the number of times the neighbors are cycled through, which, as a result, affects the distance at which pixels influence one another. This occurs because the new product values for all pixels are being computed in parallel, so that after one iteration, all neighboring pixels have had their new products values updated. Hence, in the second iteration, these new values involve information propagated from beyond a pixel's immediate neighbors. In the limiting case of an infinite number of iterations, the algorithm converges to produce an output image that is simply the input image scaled by the image's maximum value. A practical value for the number of iterations is 4. The final step is to scale the new product values to make an estimated lightness (see Sec. 6 on scaling of Retinex output to desired media and purpose). In the case of color images, the function retinex_mccann99 must be applied to each of the color channels independently.

The code is based on MATLAB™ 5 (Version 5.1.0.421). For the reader unfamiliar with MATLAB™, the statement IP(IP>Maximum)=Maximum, which sets all values in matrix IP that are greater than Maximum to Maximum, demonstrates an important feature of the language, namely,

that most of the functions and operators work on whole matrices applying the given function to all matrix elements.

4.2 *Frankle-McCann Retinex*

As in McCann99 Retinex, Frankle-McCann Retinex computes long-distance interactions between pixels first and then progressively moves to short-distance interactions. In Frankle-McCann, the spacing between the pixels being compared decreases with each step. The direction between pixels also changes at each step, in clockwise order. At each step, the comparison is implemented using the ratio-product-reset-average operation. The process continues until the spacing decreases to 1 pixel.

The original algorithm assumed the input image to be 512×512. This followed the hardware design of the I^2S. As a result, the initial spacing between pixels started at 256. We have generalized the algorithm slightly so that our implementation works on an image of arbitrary size. In this case, the initial spacing (as encoded by the variable shift) is computed as the largest power of 2, smaller than both of the input image dimensions.

The function CompareWith(s_row, s_col) updates the current lightness estimate for a pixel using the ratio-product-reset-average operation described before. In the case of Frankle-McCann, it is based on the pixel located at a distance of s_row, s_col. The square spiral path structure in this implementation means that when this function is called, one of the two parameters will always be zero. The original Frankle and McCann[2] implementation had the option of either square or 8-direction comparisons.

5 Retinex Parameters

All spatial operators use variable parameters to appropriately match their effects to input images. For example, this is true of unsharp masking, jpeg, and Retinex spatial operators.

The purpose of unsharp masking is to change the spatial content in the image, particularly in the high-spatial-frequency components. When successfully used, the image looks sharper and free of artifacts. With inappropriate parameters, the process will generate artifacts that are visible to the observer. If we compare the effects of a particular unsharp mask on same-size prints of a 256-×-256 digital image with the effects on an 2k-×-2k image, we see that they act very differently. A sharpening filter that is appropriate for the small image will have no effect on large images, while an appropriate filter for the large images will introduce artifacts in small ones. Given a print size and a viewing distance, one can optimize the shape of the filter kernel. The choice of sharpening kernel is selected so as to keep artifacts below visual threshold, which is a function of both spatial frequency,[19] size of the display,[20] and light intensity of the display.[21]

An analogous spatial dependence is found in jpeg compression, where knowledge of human sensitivity to spatial information is used to reduce the number of bits for rendering a visually similar image.[22] When we select a quality factor, we are controlling an underlying array of coefficients that filter the data, so as to reduce the data needed to recreate the image. To make two same-size prints from a 256-×-256 versus a 2k-×-2k image requires different jpeg coefficients. Any reduction in information will likely be visible in the small number-of-pixel image, while the larger image might well be compressed by factors of 10:1 or 20:1 without noticeable effect. The difference arises because the size and viewing distance control what information the observer can see in the final prints. Large digital files often contain more information than can be seen in a small print. This is the information that jpeg discards. As with unsharp masking, the user specifies the spatial parameters to optimize performance and avoid artifacts.

Retinex has parameters that are responsive to both spatial frequency and dynamic range of the input data. The number of iterations, as specified in the MATLAB™ code by nIterations, controls the amount of dynamic range compression and sets the stage for a different level of postprocessing by a post-lookup table (postLUT). The term postLUT derives from historical use of image processing hardware using a lookup table. PostLUT processing simply refers to the application of a function f applied uniformly to every image pixel, $I(x,y)=f[I(x,y)]$, for all image locations (x,y). The effect of the number of iterations can be seen in Fig. 1.

As we can see, the effect of the number of iterations (nIterations) is to reduce the contrast of the images, as demonstrated by the smaller range in the histograms. The process moves the entire image into a smaller dynamic range, with smaller digit differences representing edge ratios. With very few iterations, the range of output digits is small. The postLUT expansion (stretching of the image intensities) must be large to regenerate edge ratios appropriate for a print. With more iterations, the range of output digits is larger. The postLUT expansion will be moderate to regenerate edge ratios. With a very large number of iterations, the range of output digits is large, approaching that of the input image. The postLUT expansion must be small to none to regenerate edge ratios. The amount of postLUT expansion and its shape will vary with the amount of dynamic range compression.

The examples of unsharp masking and jpeg compression demonstrate the need for selecting the right parameters to match viewing size and distance. Analogously, the viewing distance, size, dynamic range and noise level of the input image, the number of iterations, and the postLUT are all important to make artifact-free Retinex images.

6 Scaling of Retinex Output to Desired Media and Purpose

As shown in Fig. 1, the contrast of the output is controlled by the number of iterations. This parameter can vary the output from radical to no dynamic range compression. The input data also plays a major role. The total dynamic range of input data determines the magnitude of radiance ratio associated with each digit. The final parameter is the postLUT that matches the final new product with the output media. That media can be a printer, a monitor, a LCD display, a system profile, a 3-D plot of output at each pixel (output equal height), or a pseudocolor image. The essential idea is that the input calibration controls the correlation between digital differences and radiances in the world. The number of iterations controls the degree of compression. The postLUT controls the rendition of new product digital differences in the output media. All three parameters (input

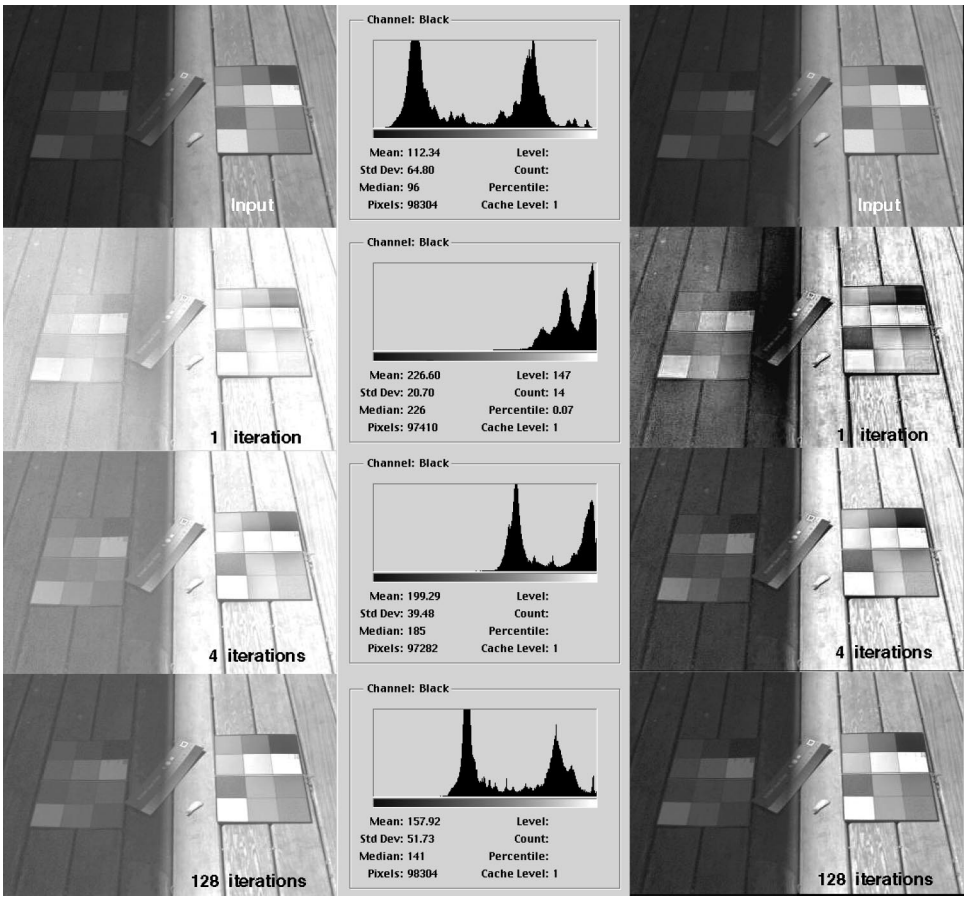


Fig. 1 This figure demonstrates the role of the number of iterations and postLUTs. The first column shows the effect of spatial comparisons (ratio product reset average). The second column is the histogram of the images in the first column. The third column shows images that have been stretched by a different postLUT for each number of iterations. The first row shows the input $\log_{10}$ image scaled so that 3.5 $\log_{10}$ units covers 0 to 255. The sun half of the image is on the right and the shade half is on the left. The shade image is a lower radiance copy of the sun image. The histogram of this image is in the second column. The third column image is the same as the first column, illustrating that it has a slope 1.0 postLUT. Output equals input. The second row shows an output image using one iteration, with its histogram. Here the output dynamic range has been compressed into the top 25% of the 0 to 255 digit range. A slope 4.0 linear postLUT will stretch the first column image to render contrast in the sun properly. It is very steep and generates artifacts. The third row shows the output for four iterations, and its histogram. Here the range data has been compressed from 3 log units to 1.5. A slope 2.0 postLUT has only to expand the data from 128 to 0. The fourth row shows the output for 128 iterations and its histogram. There is only a 25% compression. A slope 1.5 postLUT will be very gentle; however, the improvement of the shadow detail in the third column output image is minimal. In this figure, we used simple linear postLUTs to illustrate how calibration, number of iterations, and postLUT work together. To optimize the image, these postLUTs should be shaped so as to take into account the response of the output device and the tone reproduction curve desired. (See Appendices 2 and 3 of Frankle and McCann for details.[2])

dynamic range, number of iteration, and postLUT) are crucial to the process. All three share the control of the output image. They can be used only as well designed sets. They are not randomly interchangeable.

7 Results on Test Images

Figures 2–5 illustrate the behavior of the two algorithms. Figure 2 shows the behavior when the input is a simple square at the very center of the image. A slight asymmetry can be seen in both the McCann99 (using four iterations comparing eight nearest neighbors) and Frankle-McCann (using four iterations of four directions) outputs.

These calculations used the same pattern of spatial comparisons for each layer of comparisons. The McCann99 output shows the effect of processing the 8-pixel neighbors in clockwise order. No postLUT has been applied to these images. This enhances the visibility of the effect.

The calculations in Fig. 2 used the same pattern of spatial comparisons for each size of comparisons. The original Frankle and McCann calculation changed the order of the direction of comparisons in each size of spatial separation. This sequence of the spatial process was controlled by a LUT of comparison directions. Such randomization of the comparison process minimizes the directional gradients

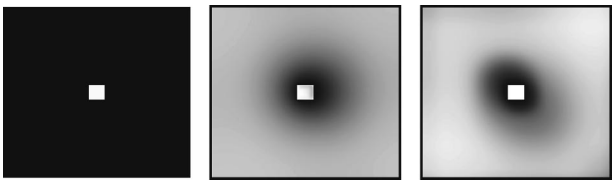


Fig. 2 Effect of McCann99 and Frankle-McCann processing (without postLUT) on input of a single bright square against a black background. In the limiting case of the square being a single pixel, this is analogous to the point spread function for the algorithm. It must be noted that because of the reset step, the shape of this function varies depending on the direction of individual comparisons of the image content. Frankle-McCann used different papers of spatial comparisons to minimize these effects. From left to right we have: input image, McCann99 four-iteration output, and Frankle-McCann four-iteration output.

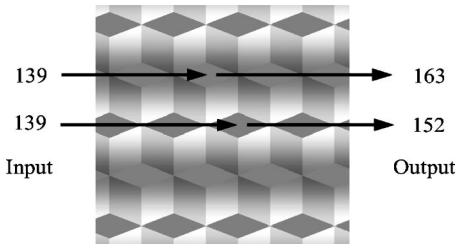


Fig. 3 Logvinenko cubes pattern illusion. As shown on the left, the input values of the cube tops are equal despite the fact that we see them as unequal. McCann99 four-iteration Retinex output values are shown on the right.

shown in Fig. 2. Alternatively, one can change the averaging process controlling the old product. If all the reports from different directions were averaged before changing the value of the old product, then these calculated spatial asymmetries are not observed. The use of postLUTs and more complex sequences of spatial comparisons all contribute to reducing the magnitude visibility of asymmetries.

Figure 3 shows Logvinenko's gradient experiment, which generates a large lightness change between the diamonds. A vertical sinusoidal gradient in nondiamond areas creates the illusion. The numbers on the left side of Fig. 3 show that the input digits for the light and dark diamond faces are both 139. The numbers on the right show the output from the corresponding faces to be 152 and 163 after McCann99 four-iteration processing. McCann[6] reports that, ''Retinex models can predict appearances that were previously attributed to cognitive behavior.'' Figure 4 shows pseudocolor renditions of input (left) and output (right) of the Logvinenko illusion. The diamond-shaped tops of the cubes are equal on the left and unequal on the right. Note that the upper faces of the output cubes are not uniform.

Figure 5 shows the effect of McCann99 applied to a color image with a substantial blue color cast. The algorithm has been applied to each of the color channels independently. Clearly, in this case the color cast has been removed. Retinex differs from many color constancy methods, in that it does not aim to find a single chromaticity for the scene illumination, as is the case, for example, in the neural network[23] and color by correlation[24] methods. Retinex instead adjusts the image colors in a nonglobal manner as is necessary, since the model attempts to match the human visual response. Some effects of this can be seen in the way that some of the green bleeds into the white area surrounding the C in Compiler, and the way the blue is darkened near the white lettering on the right-hand blue book in Fig. 5(b).

8 Discussion

This work describes the basic Retinex algorithms in MATLAB™ code. It provides the starting point for many different implementations for many possible variations. This code is the basis of making spatial comparisons in a very efficient manner. In carefully calibrated situations, it can be used as the basis for a model of human color appearance. This requires accurate calibration in both the luminance and spatial frequency domains. Numbers of iterations for each pixel separation or level of pyramid processing must match human spatial frequency data.[3,6,25] Alternatively, it can be used to enhance images of unknown calibrations in digitization. In an uncalibrated mode, it is more limited. The system works by enhancing edges. If poor calibration introduces edges from noise, the process will enhance the noise. Nevertheless, uncalibrated input images generally appear better with Retinex processing than without it.

As in many image-processing operations,[26] there are three sequential steps:

1. taking the raw input and transforming the information into an image space appropriate for the process
2. performing the process
3. scaling the output process into a space appropriate for the end use. In this particular case, ideally the input transforms convert the captured digits into a space in which constant, scene edge ratios have constant differences in digits. This property can be used by the process to render pairs of objects in different illumination as equally different in appearance.

The process assumes that the visual system uses edges to synthesize appearance. The Retinex algorithms provide an image processing engine that synthesizes sensation images from spatial comparisons of radiance inputs. The meaningful parameters in McCann99 are the pyramid level and the number of iterations. In Frankle and McCann, it is separation and the number of iterations. In McCann, McKee, and Taylor, it is path length and the number of paths. A number of studies experimentally measured the appearance of a variety of achromatic and color constancy experiments. Using this quantitative data, it is possible to experimentally optimize the parameters of the model.[2,27–29] The details of this work are summarized by Ciurea, Funt, and McCann[29] and McCann and Savoy.[30] All of these studies indicate that the human visual system is neither local nor global, with regard to spatial interactions. Neither local center-surround operators, nor global gray-world models can account for psychophysical results.[29] The spatial frequency filter applied by human vision is image dependent.[16] The effect of maxima have an effect over large distances, but varies with distance and enclosure.[31,32]

In the examples described, we used constant values for the number of iterations for all levels of a pyramid. Although efficient, this is not the best set of parameters for

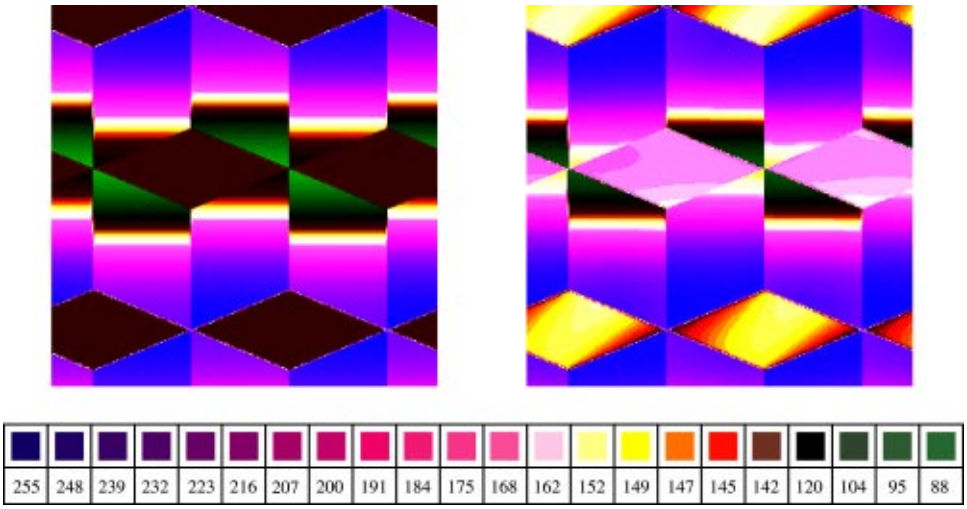


Fig. 4 Pseudocolor representation of a portion of the Logvinenko cubes input (left) and McCann99 four-iteration output (right). Note that despite the fact the upper cube faces on alternating rows appear to differ in intensity, the top faces of all the cubes are, in fact, both uniform and equal. In the output, however, the top faces of the cubes are no longer equal nor are they completely uniform.

modeling human vision. An obvious variation is to have different numbers of iteration for each size. Franke and McCann used different numbers of iterations for each size of separation. They also changed the pattern of directions to remove the pattern found in Fig. 2.[2] The 1, 4, and 128 iteration images in Fig. 1 could be described by their spatial-frequency content. The difference between the input and output images describes a spatial filter. That filter can be resolved into a 2-D spatial filter, or set of spatial filters. Since the work of Campbell and Robson, and Hubel and Wiesel, human visual processing has been regarded as sets of spatial channels.[33] As demonstrated in Fig. 1, the number of iterations controls the strength of the filter. The greater the number of iterations, the weaker the filter. The size of the separation or the pyramid level controls the spatial frequency of the response. The number of iterations at that level controls the strength of the filter at that frequency. Just as human vision requires models using multichannels with different filter strengths, the Retinex models should have the same spatial frequency tuning.

Sobol[32] has described variations to the Retinex process that uses LUTs to control the magnitude and shape of edges at different spatial separations. This algorithm produces dramatic images. The ability to control different spatial frequencies adds considerable power to the algorithm. In addition, it makes the model more like human vision.

An important final variation is the use of the spatial comparison engine for gamut mapping problems. Examples

Fig. 5 (a) Input with blue color cast created by scene illumination, for which the camera was not balanced. The image also has extended dynamic range obtained by frame averaging. (b) Output from the McCann99 four-iteration. (c) Output from the Frankle-McCann four-iteration. The results here can be compared with those of Barnard.[14] Note that both the input and output images have been adjusted with postLUTs for printing. The actual Retinex input image is in log space.

Matlab implementation of McCann99 retinex

Notation:

L:	logarithmic input image
nIterations:	number of iterations for each pixel
nLayers:	number of pyramid layers
OP:	matrix of Old Products for all pixels
RR:	input radiance
NP:	matrix of New Products for all pixels
OPE:	OP padded with zeros for doing all computations at once
RRE:	RR padded with two additional columns and two additional rows
IP:	Intermediate Product computed with the Ratio-Product

```
function Retinex = retinex_mccann99(L, nIterations)
global OPE RRE Maximum
[nrows ncols] = size(L);                              % get size of the input image
nLayers = ComputeLayers(nrows, ncols);                % compute the number of pyramid layers
nrows = nrows/(2^nLayers);                            % size of image to process for layer 0
ncols = ncols/(2^nLayers);
if (nrows*ncols > 25)                                 % not processing images of area > 25
  error('invalid image size.')                        % at first layer
end
Maximum = 1;                                          % maximum color value in the image
OP = Maximum*ones([nrows ncols]);                   % initialize Old Product
for layer = 0:nLayers
   RR = ImageDownResolution(L, 2^(nLayers-layer));    % reduce input to required layer size

   OPE = [zeros(nrows,1) OP zeros(nrows,1)];          % pad OP with additional columns
   OPE = [zeros(1,ncols+2); OPE; zeros(1,ncols+2)];   % and rows
   RRE = [RR(:,1) RR RR(:,end)];                      % pad RR with additional columns
   RRE = [RRE(1,:); RRE; RRE(end,:)];                 % and rows

   for iter = 1:nIterations
     CompareWithNeighbor(-1, 0);                      % North
     CompareWithNeighbor(-1, 1);                      % North-East
     CompareWithNeighbor(0, 1);                       % East
     CompareWithNeighbor(1, 1);                       % South-East
     CompareWithNeighbor(1, 0);                       % South
     CompareWithNeighbor(1, -1);                      % South-West
     CompareWithNeighbor(0, -1);                      % West
     CompareWithNeighbor(-1, -1);                     % North-West
   end
   NP = OPE(2:(end-1), 2:(end-1));
   OP = NP(:, [fix(1:0.5:ncols) ncols]);              %%% these two lines are equivalent with
   OP = OP([fix(1:0.5:nrows) nrows], :);              %%% OP = imresize(NP, 2) if using Image
   nrows = 2*nrows; ncols = 2*ncols;                  % Processing Toolbox in MATLAB
end
Retinex = NP;

function CompareWithNeighbor(dif_row, dif_col)
global OPE RRE Maximum

% Ratio-Product operation
IP = OPE(2+dif_row:(end-1+dif_row), 2+dif_col:(end-1+dif_col)) + ...
     RRE(2:(end-1),2:(end-1)) - RRE(2+dif_row:(end-1+dif_row), 2+dif_col:(end-1+dif_col));

IP(IP > Maximum) = Maximum;                           % The Reset step

% ignore the results obtained in the rows or columns for which the neighbors are undefined
if (dif_col == -1) IP(:,1) = OPE(2:(end-1),2); end
if (dif_col == +1) IP(:,end) = OPE(2:(end-1),end-1); end
if (dif_row == -1) IP(1,:) = OPE(2, 2:(end-1)); end
if (dif_row == +1) IP(end,:) = OPE(end-1, 2:(end-1)); end
NP = (OPE(2:(end-1),2:(end-1)) + IP)/2;               % The Averaging operation
OPE(2:(end-1), 2:(end-1)) = NP;

function Layers = ComputeLayers(nrows, ncols)
power = 2^fix(log2(gcd(nrows, ncols)));               % start from the Greatest Common Divisor
```

Fig. 6 Matlab implementation of McCann99 Retinex (*continued on next page*).

are in another article in this issue.[16] The principle is straightforward. If displays and printers had the same color spaces, then Tristimulus matches would be able to successfully transform display/print images. However, they occupy only half of their combined physical color space. Using strict colorimetric matches creates problems with extra-gamut colors. All of the variations between the gamut of the smaller space are represented by the gamut value. This clipping of local detail produces undesirable artifacts. Many algorithms systematically distort the colorimetric matches to achieve an image with a better appearance. All the transforms increase the colorimetric errors.[16,34]

The Retinex approach uses two different sets of RGB input images. One image (Goal) has digits representing the color space values of the large gamut desired image. The other image (Best) has digits representing the color space values of the best colorimetric reproduction possible in the smaller gamut media. The RGB Goal images are used to supply the ratios. The Best image is used to supply the reset values. The rest of the process is the same as described before. The color gamut calculation provides an excellent example of using the Retinex spatial-comparison process to generate new sensation images that have very similar appearances with different radiances at each pixel. Experiments have shown that human spatial processing is key to understanding color constancy, high dynamic range sensations, and transparency.[35] Further, spatial comparisons can be used to simplify gamut mapping algorithms. As long as spatial comparisons are constant, near-constant appearances can be made from very different stimuli.

9 Conclusions

We present new, very concise MATLAB™ implementations of two of the main practical Retinex algorithms. (The MATLAB™ code and figures are available at http://

```
while(power > 1 & ((rem(nrows, power) ~= 0) | (rem(ncols, power) ~= 0)))
   power = power/2;                                  % and find the greatest common divisor
end                                                  % that is a power of 2
Layers = log2(power);

function Result = ImageDownResolution(A, blocksize)
[rows, cols] = size(A);                              % the input matrix A is viewed as
result_rows = rows/blocksize;                        % a series of square blocks
result_cols = cols/blocksize;                        % of size = blocksize
Result = zeros([result_rows result_cols]);
for crt_row = 1:result_rows                          % then each pixel is computed as
   for crt_col = 1:result_cols                       % the average of each such block
      Result(crt_row, crt_col) = mean2(A(1+(crt_row-1)*blocksize:crt_row*blocksize, ...
                                       1+(crt_col-1)*blocksize:crt_col*blocksize));
   end
end
```

Matlab implementation of Frankle-McCann retinex

```
function Retinex = retinex_frankle_mccann(L, nIterations)
global RR IP OP NP Maximum
RR = L;
Maximum = 1;                                         % maximum color value in the image
[nrows, ncols] = size(L);

shift = 2^(fix(log2(min(nrows, ncols)))-1);          % initial shift
OP = Maximum*ones(nrows, ncols)                      % initialize Old Product

while (abs(shift) >= 1)
   for i = 1:nIterations
      CompareWith(0, shift);                         % horizontal step
      CompareWith(shift, 0);                         % vertical step
   end
   shift = -shift/2;                                 % update the shift
end
Retinex = NP;

function CompareWith(s_row, s_col)
global RR IP OP NP Maximum
IP = OP;
if (s_row + s_col > 0)
   IP((s_row+1):end, (s_col+1):end) = OP(1:(end-s_row), 1:(end-s_col)) + ...
   RR((s_row+1):end, (s_col+1):end) - RR(1:(end-s_row), 1:(end-s_col));
else
   IP(1:(end+s_row), 1:(end+s_col)) = OP((1-s_row):end, (1-s_col):end) + ...
   RR(1:(end+s_row),1:(end+s_col)) - RR((1-s_row):end, (1-s_col):end);
end
IP(IP > Maximum) = Maximum;                          % The Reset operation
NP = (IP + OP)/2;                                    % average with the previous Old Product
OP = NP;                                             % get ready for the next comparison
```

Fig. 6 (*Continued.*)

www.cs.sfu.ca/research/groups/Vision/.) Our hope is that this will eliminate much of the variability in what is meant when different researchers refer to Retinex and thereby facilitate further rigorous testing and discussion of the method. For modeling human vision, these MATLAB™ programs depend on calibrated input data. Although these MATLAB™ programs provide the details of how pixels are compared and processed during the ratio-product-reset-average steps of Retinex processing, they do not provide details on the selection of an appropriate postLUT for a particular output device. The postLUT must be provided by the reader.

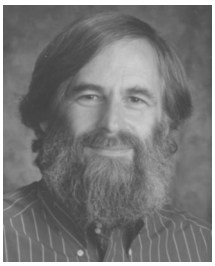
Acknowledgments

The authors gratefully acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada and Hewlett Packard Incorporated.

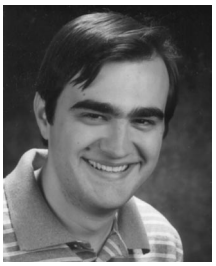
References

1. E.H. Land and J.J. McCann, ''Lightness and Retinex theory,'' *J. Opt. Soc. Am.* **61**, 1–11 (1971).
2. J. Frankle and J. McCann, ''Method and apparatus for lightness imaging,'' US Patent No. 4,384,336 (1983).
3. J.J. McCann, S.P. McKee, and T.H. Taylor, ''Quantitative studies in Retinex theory,'' *Vision Res.* **16**, 445–458 (1976).
4. E.H. Land, ''The Retinex theory of color vision,'' *Sci. Am.* **237**, 108–129 (1977).
5. E.H. Land, ''An alternative technique for the computation of the designator in the Retinex theory of color vision,'' *Proc. Nat. Academy Sci.* **83**, 3078–3080 (1986).
6. J.J. McCann, ''Lessons learned from Mondrians applied to real images and color gamuts,'' *Proc. IS&T/SID 7th Color Imag. Conf.*, pp. 1–8 (1999).
7. J.J. McCann and K.L. Houston, ''Color sensation, color perception and mathematical models of color vision,'' in *Colour Vision*, J.D. Mollon and L.T. Sharpe, Eds., pp. 891–894, Academic Press, London (1983).
8. E.H. Land and J.J. McCann, ''Method and system for image reproduction based on significant visual boundaries of original object,'' US Patent No. 3553360 (1971).
9. E.H. Land, *et al.*, ''Image reproduction system which detects subject by sensing intensity ratios,'' US Patent No. 3651252 (1972).
10. J.J. McCann, ''Color Mondrian experiments without adaptation,'' *Proc. AIC*, pp. 159–162 (1997).
11. W.R. Wray, ''Method and apparatus for image processing with field portions,'' US Patent No. 4750211 (1988).
12. J.J. McCann, ''Calculated color sensations applied to color image reproduction,'' *Proc. SPIE* **901**, 194–204 (1988).
13. H. Wallach, ''Brightness constancy and the nature of achromatic colors,'' *J. Exp. Psychol.* **38**, 310–324 (1948).
14. K. Barnard and B. Funt, ''Investigations into multi-scale retinex (MSR),'' in *Colour Imaging: Vision and Technology*, L.W. Macdonald and M.R. Luo, Eds., John Wiley and Sons, New York, pp. 17–36, (1999).
15. D. Marini and A. Rizzi, ''Color appearance approach to image database visual retrieval,'' *Proc. SPIE* **3964**, 186–195 (2000).
16. J.J. McCann, ''Capturing a black cat in shade: The past and present of Retinex color appearance models,'' *J. Electron. Imaging* **13**(1), 28–34 (2004) (in this issue).
17. D.A. Brainard and B.A. Wandell, ''Analysis of the Retinex theory of color vision,'' *J. Opt. Soc. Am. A* **3**(10), 1651–1661 (1986).
18. J.J. McCann, ''The role of nonlinear operations in modeling human color sensations,'' *Proc. SPIE* **1077**, 355–363 (1989).
19. F.W. Campbell and J.G. Robson, ''Application of Fourier analysis to the visibility of gratings,'' *J. Physiol.* **197**, 551–566 (1968).
20. J.J. McCann, *et al.*, ''Visibility of continuous luminance gradients,'' *Vision Res.* **14**, 917–927 (1974).
21. R.L. Savoy, ''Low spatial frequencies and low number of cycles at low luminances,'' *J. Photographic Sci. Eng.* **22**(2), 76–79 (1978).

22. M. Rabbani and P.W. Jones, *Digital Image Compression Techniques*. SPIE Press, Bellingham, WA (1991).
23. V.C. Cardei, B. Funt, and K. Barnard, ''Estimating the scene illumination chromaticity by using a neural network,'' *J. Opt. Soc. Am. A* **19**(12), 2374–2385 (2002).
24. G.D. Finlayson, S. Hordley, and P.H. Hubel, ''Color by correlation: a simple unifying framework for color constancy,'' *IEEE Trans. Pattern Anal. Mach. Intell.* **23**(11), 1209–1221 (2001).
25. J.J. McCann, ''Calculating lightnesses in a single plane,'' *J. Electron. Imaging* **10**(1), 110–122 (2001).
26. M. Abdulwahab, J.L. Burkhardt, and J.J. McCann, ''Method and apparatus for transforming color image data on the basis of an isotropic and uniform colorimetric space,'' US Patent No. 4839721 (1984).
27. J.J. McCann, E.H. Land, and S.M.V. Tatnall, ''A technique for comparing human visual responses with a mathematical model for lightness,'' *Am. J. Optom. Arch. Am. Acad. Optom.* **47**, 845–855 (1970).
28. B. Funt and F. Ciurea, ''Control parameters for Retinex,'' *Proc. 9th Congress Intl. Color Assoc.*, pp. 287–290 (2001).
29. F. Ciurea, B. Funt, and J.J. McCann, ''Tuning Retinex parameters,'' *J. Electron. Imaging* **13**(1), 21–27 (2004) (in this issue).
30. J.J. McCann and R.L. Savoy, ''Measurement of lightness: Dependence on the position of a white in the field of view,'' *Proc. SPIE* **1453**, 402–411 (1991).
31. F. Ciurea and B. Funt, ''A large database for color constancy research,'' in *11th Color Imag. Conf.* (2003).
32. R. Sobol, ''Improving the Retinex algorithm for rendering wide dynamic range photographs,'' *Proc. SPIE* **4662**, 341–348 (2002).
33. J. Cowan, ''Visual cortex and the Retinex algorithm,'' *Proc. SPIE* **4662**, 279–285 (2002).
34. J.J. McCann, ''A spatial color gamut calculation to optimize color appearance,'' in *Colour Imaging: Vision and Technology*, L.W. Macdonald and M.R. Luo, Eds., pp. 213–233, Wiley and Son Ltd., Chichester, UK (2002).
35. S. Westland, P.O. Da Pos, and C. Ripamonti, ''Conditions for perceptual transparency,'' *Proc. SPIE* **4662**, 315–323 (2002).



Brian Funt is a professor of computing science at Simon Fraser University. After receiving his PhD from the University of British Columbia in 1976, he spent two years at Stanford as a postdoctoral researcher followed by two years as a professor at State University New York, Buffalo. He has been at Simon Fraser University since 1980. His work on computational models of color perception began in the early 1980s and has covered color constancy, color object recognition, high dynamic range color imaging, and color in computer vision. He is a Marr Prize recipient. This year he is the General Co-Chair of the Eleventh Color Imaging Conference.



Florian Ciurea received his BSc in computer science from Politehnica University of Bucharest, Romania. He is currently a PhD student in computing science at Simon Fraser University. He is interested in color imaging, with a focus on color correction and computational models for color constancy.

John McCann received his BA degree in biology from Harvard University in 1964. He managed the Vision Research Laboratory at Polaroid from 1961 to 1996. His work concentrated on research in human color vision, large format instant photography, and the reproduction of fine art. He is a Fellow of the IS&T. He is a past president of IS&T and the Artists Foundation, Boston. In 2003, he received the IS&T/OSA Edwin Land Medal. He is currently consulting and continuing his research on color vision.