SLMRACE: A noise-free new RACE implementation with reduced computational time

Juliet Chauvin, Edoardo Provenzi

SLMRACE: A noise-free new RACE implementation with reduced computational time

Juliet Chauvin[a], **Edoardo Provenzi**[a]
[a]Université Paris Descartes, MAP5, 45 rue de Saints-Pères, Paris, France, 75006

Abstract. In this paper we present a faster and noise-free implementation of the RACE algorithm. RACE has mixed characteristics between the famous Retinex model of Land and McCann and the ACE color-correction algorithm. The original random spray-based RACE implementation suffers from two main problems: its computational time and the presence of noise. Here we will show that it is possible to adapt a technique recently proposed by Banić et al. to the RACE framework in order to drastically decrease the computational time and to avoid noise generation. The new implementation will be called SRACE, for smart-RACE.

Keywords: Color image enhancement, Retinex, RACE, SLRMSR, GIF, SRSR.

*Corresponding author: Edoardo Provenzi, edoardo.provenzi@parisdescartes.fr

1 Introduction

The Retinex theory of color vision proposed by Land and McCann[1] aimed at providing a computational model to understand some essential features of color vision such as locality or robustness with respect to illuminant changes. The Retinex theory has then been used for many other purposes, one of them is perceptual-based color correction: given an input picture taken under any illumination condition, one would like to transform it in order to induce a color sensation closer to what a human observer would experience in the scene depicted by the image. This is the Retinex use that will be considered in this paper.

The original version of Retinex used piecewise linear path to explore the image content around a point of an image. Several works have been dedicated to the analysis of path geometry which gives the best performances, see e.g.[2–6] The mathematical analysis performed in[7] allowed pointing out intrinsic problems related with the use of paths and this led to their replacement with 2-D sampling structures, called random sprays. This formulation of Retinex is known as RSR for 'Random Spray Retinex'.[8]

The spray technique has also been employed to reduce the computational time of another perceptually-based color correction algorithm called ACE for 'Automatic Color Equalization'.[9] The possibility to use sprays as sampling structure for both ACE and RSR allowed to mix the two algorithms into a single one with in-between features. The hybrid model is called RACE.[10]

Even if the spray-based technique is a noticeable improvement with respect to the path-based one, it is still affected by a high computational time and generation of noise, which affects, in particular, images characterized by large homogeneous areas.

To overcome these problems, some suitable methodologies has been proposed[11,12] and applied to RSR-based algorithms. In this paper we will show that these techniques can be also be adopted to provide a noise-free and faster implementation of RACE.

Tests and comparisons will show the efficiency of the new implementation with respect to the original one.

Before starting with the technical content of the paper, let us fix the notation. We will denote with $\Omega \subset \mathbb{Z}^2$ the spatial domain of a digital image of size $|\Omega|$ and with $x \equiv (x_1, x_2)$ and $y \equiv (y_1, y_2)$ the coordinates of two arbitrary pixels in Ω. We will always consider a *normalized dynamic range* in $[0, 1]$, so that a RGB image function will be denoted with $\vec{I} : \Omega \longrightarrow [0, 1] \times [0, 1] \times [0, 1]$, $x \mapsto (I_R(x), I_G(x), I_B(x))$, where each scalar component $I_c(x)$ defines the intensity level of the pixel $x \in \Omega$ in the red, green and blue channel, respectively.

We stress that we will perform every computation on the scalar components of the image, thus treating each chromatic component separately as in the original Retinex paper.[1] Therefore, we will avoid the subscript c and write simply $I(x)$ to denote the intensity of the pixel x in a given chromatic channel.

2 RSR, ACE and RACE

In this section we will describe the algorithms RSR, ACE and their fusion RACE. Let us start with RSR. Fixed a pixel x, N different sprays S_k, $k = 1, \dots, N$, as the one depicted in Fig. 1, are centered in x. For simplicity we will consider all the sprays to have the same number of pixels n. The generic pixel in the spray S_k, different from the center, will be denotes with y and the notation H_k will be used to indicate the pixel with highest intensity belonging to the spray S_k (notice that H_k can coincide with the center pixel x).

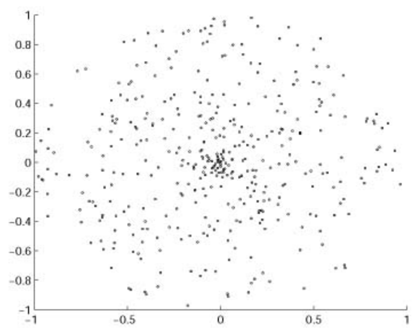

Fig 1 A random spray used in RSR.

The re-computation of pixel intensity performed by RSR is the following:

$$I(x) \longmapsto L^{\text{RSR}}(x) = I(x) \frac{1}{N} \sum_{k=1}^{N} \frac{1}{I(H_k)}, \tag{1}$$

for all pixel $x \in \Omega$, and for the three chromatic channels separately. Due to the quotient over the maximal intensity of each spray, RSR belongs to the category of white-patch color correction algorithms.

By contrast, the ACE method uses a different approach inspired by the gray-world approach.[13]

To describe ACE, we have to previously define the slope function $s_\alpha : [-1,1] \to [-1,1]$ as follows:

$$s_\alpha(t) = \begin{cases} -1 & \text{if } -1 \leq t \leq \frac{-1}{\alpha} \\ \alpha t & \text{if } \frac{-1}{\alpha} < t < \frac{1}{\alpha} \\ +1 & \text{if } \frac{1}{\alpha} \leq t \leq 1, \end{cases}$$

where $\alpha > 1$ is a slope parameter freely selected by a user. We must also consider a suitably normalized weight function w, i.e. $\sum_{x\in\Omega} w(x,y) = 1$, decreasing with the Euclidean distance $\|x-y\|$ and such that $w(x,x) = 0$.

We can now define the first stage of ACE as the following intensity re-computation:

$$I(x) \longmapsto R(x) = \sum_{y\in\Omega} w(x,y) s_\alpha(I(x) - I(y)), \tag{2}$$

thanks to the slope function, small intensity differences are amplified and large ones are saturated. The second, and final, step of the ACE computation is the following global re-mapping:

$$L^{\text{ACE}}(x) = \frac{1}{2} + \frac{R(x)}{2M} \tag{3}$$

where $M = \max_{x\in\Omega}\{R(x)\}$. This last step performs a global white-patch mapping, which sets at least one pixel to 1 in each separate chromatic channel.

The computational time of ACE formulated in this way is huge for high resolution images. However, if the computation 2 is performed over a localized spray instead of over the entire image domain, time computation can drastically decrease.

This is one of the reasons why in[10] ACE has been recast in the spray framework. The other reason is that this also allows to easily combine RSR and ACE in a single algorithm with intermediate features between the two models.

Let us show how to rewrite ACE by using localized sprays. First of all we re-define the slope function in order to incorporate the global white-patch mapping: $\tilde{s}_\alpha : [-1, 1] \to [0, 1]$:

$$\tilde{s}_\alpha(I(x) - I(y)) = \begin{cases} 0 & \text{if} \quad -1 \leq I(x) - I(y) \leq \frac{-1}{2\alpha} \\ \frac{1}{2} + \alpha(I(x) - I(y)) & \text{if} \quad \frac{-1}{2\alpha} < I(x) - I(y) < \frac{1}{2\alpha} \\ +1 & \text{if} \quad \frac{1}{2\alpha} \leq I(x) - I(y) \leq 1. \end{cases}$$

Then, the spray-version of ACE can be defined as follows:

$$L^{\text{ACE}}_{\text{spray}}(x) = \frac{1}{N} \sum_{k=1}^{N} \frac{1}{\tilde{n}_k} \sum_{y \in S_k(x) \setminus \{x\}} \tilde{s}_\alpha(I(x) - I(y)) \tag{4}$$

for all $x \in \Omega$, where $\tilde{n}_k$ is the number of spray points which actually fall in the image domain.

The combination of RSR and ACE into RACE is given simply by the arithmetic average of $L^{\text{RSR}}(x)$ and $L^{\text{ACE}}_{\text{spray}}(x)$:

$$I(x) \longmapsto L^{\text{RACE}}(x) = \frac{1}{N} \sum_{k=1}^{N} \frac{1}{2} \left(\frac{I(x)}{I(H_k)} + \frac{1}{\tilde{n}_k} \sum_{y \in S_k(x) \setminus \{x\}} \tilde{s}_\alpha(I(x) - I(y)) \right) \tag{5}$$

for all $x \in \Omega$.

Even if the computational time of spray-based algorithms RSR and spray-ACE is drastically inferior that of the original path-based Retinex and ACE, respectively, it is far from being acceptable for high resolution images. In the following section, we will recall a technique to accelerate spray-based algorithms recently proposed in,[11,12] which we will use in the subsequent section to accelerate RACE.

3 LRSR (Light Random Spray Retinex) and SLMRSR (Smart Light Random Memory Spray Retinex).

In this section, we will discuss two techniques proposed in[11,12] that have been applied to RSR with the aim of reducing the noise in the output image and of decreasing the computational time.

First of all we notice that, if we use small values of N and n within the RSR algorithm to reduce the computational time, the output images are affected by a lot of noise, as it can be seen in Fig. 2.

Fig 2 *Left*: original image. *Right*: output of RSR with $N = 4$ sprays and $n = 5$ pixels per spray.

Let us start by describing how to avoid noise formation when using small values of n and N by using the technique proposed in.[11] The corresponding method is called LRSR for 'Light Random

Sprays Retinex'. Consider an arbitrary input image I and apply RSR to it, obtaining the image R. The ratio $C = \frac{I}{R}$ is called intensity change image.

In LRSR, the noise is reduced through a convolution with a kernel function k. This can be done after the computation of C, obtaining $C'_k = (C * k)(x)$, $\forall x \in \Omega$, or before it, i.e. applying the blurring on I and R, obtaining $C''_k(x) = \frac{(I*k)(x)}{(R*k)(x)}$, $\forall x \in \Omega$. By combining the two approaches, i.e filtering before and after calculation with two (possibly identical) kernels k_1 and k_2, respectively, one can define the new intensity change matrix: $C^*_{k_1,k_2}(x) = (C''_{k_1}(x) * k_2)(x)$.

Finally, the output image O of LRSR is calculated via the formula:

$$O(x) = \frac{I(x)}{C^*_{k_1,k_2}(x)} \quad \forall x \in \Omega, \tag{6}$$

the size of kernels k_1 and k_2 is 25×25 in.[11]

Now that we have seen how to reduce the noise, let us proceed by explaining how to reduce the computation complexity with the method proposed in[12] and called 'SLRMSR' for *Smart Light Random Memory Sprays Retinex*.

The basic concept is that of 'spray memory', which consists in creating a single spray that will be gradually modified while we browse the image. First of all, we extend the image by mirror symmetry. This procedure is needed to correctly treat pixels that lie near the edge of the image domain. We then consider the first pixel $x \equiv (1, 1) \in \Omega$ of the input image and we build a spray $S(x)$ with n pixels centered in x. We then pass to the subsequent pixel in the same row, i.e. $x' \equiv (2, 1)$, and we *modify just one pixel* of $S(x)$, by randomly selecting a pixel that lies in the neighborhood of x'.

By iterating this procedure, we will gradually and smoothly modify the content of each spray.

At a computational level, if we store the pixels of the spray in a hyper-matrix, for each new pixel we will just modify the last element of this matrix. Thereby, the spray will be totally renewed every n pixels.

Notice that, thanks to this idea, the computational complexity passes from $\mathcal{O}(nN|\Omega|)$ to $\mathcal{O}(n|\Omega|)$ because now we use only one spray.

When this technique is applied directly without the filtering strategy quoted above, it produces results affected by horizontal lines, as those in Fig. 3 (middle). The main reason is to be found in the great deal of information redundancy in natural images, i.e. the fact that nearby pixels are very likely to have similar values, unless they lie in the proximity of a sharp edge, thus, if we change only one pixel spray at the time and we browse the image horizontally, this kind of problem is to be expected.

Fig 3 *Left*: original image. *Middle*: output of spray memory RSR without applying the filters of LRSR. *Right*: output of SLMRSR.

4 SLMRACE (Smart Light Memory RACE) and its performances

If, instead of applying the light and spray memory techniques to RSR, we consider them in association with RACE, then we obtain an algorithm called SLMRACE.

Let us notice that, since only one spray with n points will be used in SLMRACE, formula (5) must be replaced with the following:

$$I(x) \longmapsto L^{\text{RACE}}(x) = \frac{1}{2}\left(\frac{I(x)}{\max\limits_{y\in S(x)}\{I(y)\}} + \frac{1}{n-1}\sum_{y\in S(x)\setminus\{x\}} \tilde{s}_\alpha(I(x)-I(y))\right) \quad \forall x \in \Omega. \tag{7}$$

Notice also that, due to mirror symmetry, all spray pixels correspond to an actual pixel value.

In Fig.4 we show a comparaison between the results of SLMRSR and SLMRACE with the same kernel size of 25×25. It can be seen that images filtered with SLMRACE show much more details than those filtered with SLMRSR without corrupting colors.

In Fig. 5 we present the influence of three kernel sizes on the results. It can be seen that a large kernel size allows us to see more details and obtain an output image with more contrast. However, with a large kernel size we can see a halo effect near the edges. Our tests have shown that the intermediate choice of kernel size of 25×25 is a good trade-off.

5 Conclusions

In this paper we have recalled the so-called spray-based implementations of some Retinex-inspired color correction algorithms and pointed out their limits: high computational time and noise generation. We have also discussed a recent proposal to avoid this kind of problems and showed that it can be applied to RACE, the algorithm obtained from the fusion of RSR, the random spray Retinex implementation, and the perceptually-inspired color correction model ACE. Finally, we have presented tests and comparisons to corroborate the good performances of this new RACE implementation, called SLMRACE.

Fig 4 *Left*: original image. *Middle*: output of SLMRSR. *Right*: output of SLMRACE with a slope $\alpha = 2$ and a number of spray pixel n equal to the integer part of the image diagonal of each image.

Acknowledgments

The authors would like to thank Nikola Banić for his help and advise during the implementation of SLMRACE.

Fig 5 Output of SLMRACE with different kernel sizes s *Left*: $s = 5 \times 5$. *Middle*: $s = 25 \times 25$. *Right*: $s = 50 \times 50$.

References

1 E. Land and J. McCann, "Lightness and Retinex theory," *Journal of the Optical Society of America* **61**, 1–11 (1971).

2 J. McCann, S. McKee, and T. Taylor, "Quantitative studies in Retinex theory: A comparison

between theoretical predictions and observer responses to the 'color mondrian' experiments," *Journ. of Vision Res.* **16**, 445–458 (1976).

3 J. Frankle and J. J. McCann, "Method and apparatus for lightness imaging." United States Patent, US 4,348,336 (1983).

4 D. Marini and A. Rizzi, "A computational approach to color adaptation effects," *Image and Vision Computing* **18**, 1005–1014 (2000).

5 T. J. Cooper and F. A. Baqai, "Analysis and extensions of the Frankle-McCann Retinex algorithm," *Journal of Electronic Imaging* **13**, 85–92 (2004).

6 J. J. McCann and et al., "Special session on Retinex at 40," *Journal of Electronic Imaging* **13**, 6–145 (2004).

7 E. Provenzi, L. De Carli, A. Rizzi, and D. Marini, "Mathematical definition and analysis of the retinex algorithm," *Journal of the Optical Society of America A* **22**, 2613–2621 (2005).

8 E. Provenzi, M. Fierro, A. Rizzi, L. De Carli, D. Gadia, and D. Marini, "Random spray retinex: A new retinex implementation to investigate the local properties of the model," *IEEE Transactions on Image Processing* **16**, 162–171 (2007).

9 A. Rizzi, C. Gatta, and D. Marini, "A new algorithm for unsupervised global and local color correction," *Pattern Recognition Letters* **24**, 1663–1677 (2003).

10 E. Provenzi, C. Gatta, M. Fierro, and A. Rizzi, "Spatially variant white patch and gray world method for color image enhancement driven by local contrast," *IEEE Trans. on Pattern Analysis and Machine Intelligence* **30**, 1757–1770 (2008).

11 N. Banić and S. Lončarić, "Light random sprays retinex: Exploiting the noisy illumination estimation," *IEEE Signal Processing Letters* **20**, 1240–1243 (2013).

12 N. Banić and S. Lončarić, "Smart light random memory sprays retinex: a fast retinex implementation for high-quality brightness adjustment and color correction," *JOSA A* **32**(11), 2136–2147 (2015).

13 G. Buchsbaum, "A spatial processor model for object colour perception," *Journal of the Franklin Institute* **310**, 337–350 (1980).

List of Figures

1 A random spray used in RSR.

2 *Left*: original image. *Right*: output of RSR with $N = 4$ sprays and $n = 5$ pixels per spray.

3 *Left*: original image. *Middle*: output of spray memory RSR without applying the filters of LRSR. *Right*: output of SLMRSR.

4 *Left*: original image. *Middle*: output of SLMRSR. *Right*: output of SLMRACE with a slope $\alpha = 2$ and a number of spray pixel n equal to the integer part of the image diagonal of each image.

5 Output of SLMRACE with different kernel sizes s *Left*: $s = 5 \times 5$. *Middle*: $s = 25 \times 25$. *Right*: $s = 50 \times 50$.

List of Tables