Machine Vision and Applications (2013) 24:1555–1560
DOI 10.1007/s00138-013-0494-0

SHORT PAPER

A linear system form solution to compute the local space average color

Joaquin Salas · Carlo Tomasi

Received: 21 September 2012 / Revised: 4 February 2013 / Accepted: 13 February 2013 / Published online: 1 March 2013

Abstract In this document, we present an alternative to the method introduced by Ebner (Pattern Recognit 60–67, 2003; J Parallel Distrib Comput 64(1):79–88, 2004; Color constancy using local color shifts, pp 276–287, 2004; Color Constancy, 2007; Mach Vis Appl 20(5):283–301, 2009) for computing the local space average color. We show that when the problem is framed as a linear system and the resulting series is solved, there is a solution based on LU decomposition that reduces the computing time by at least an order of magnitude.

Keywords Color constancy · Gray-world assumption · Local space average color

1 Introduction

Color constancy refers to the outstanding capability of humans to recognize the color of the objects under a wide variety of illumination conditions. Due to the many benefits such a capability would bring to computer vision systems, much research has been carried out in this field (see [13] for a recent account). Although many different algorithms have been developed [1,3] and a good level of understanding has been achieved [15,17,20], the problem remains largely unsolved [16]. A recent trend in the field [4] is to either employ several color constancy algorithms, and combine the results, or to analyze the image to assess which algorithm will perform better. For instance, for scenarios with ample gamut of colors, some researchers [6,12,19] have chosen to take into account the gray-world assumption to develop their methods [5], which state that the average surface reflectance is gray. Furthermore, a widespread practice is to assume that the illuminant is uniformly distributed through space, and only a few methods explicitly consider the presence of multiple light sources [2,10,11]. Ebner [6–10] has proposed an iterative formulation aimed at computing the color of the illuminant locally to tackle the more realistic situation of multiple illuminants coexisting in a scenario using the gray-world assumption. Ebner called *space average color* the local illuminant thus computed.

The gray-world is too bold an assumption about how the world looks in general. However, there may be the need to make assumptions to gain intuition to build more comprehensive models. In this document, we present an alternative form solution to Ebner's iteration scheme. In our proposal, we formulate the problem as a linear system and solve the resulting series. We show that the resulting matrices are sparse and hence admit a compact representation and a fast solution via LU factorization [14]. Furthermore, we show that the computing requirements are reduced at least by an order of magnitude. Also, in this document, we are only concerned with the computational aspects of the problem. A method to compute a constant descriptor for color, including chroma and lightness, is out of the scope of the present research, and it is still one of the most fascinating open problems in the field of computer vision.

The rest of the paper is organized as follows. In the next section, we review the gray-color assumption and the referred iterative formulation. Then, in Sect. 3, after framing the problem as a linear system, we derive an alternative solution. There, we discuss the use of sparse matrices to reduce the

J. Salas (✉)
Instituto Politecnico Nacional, Cerro Blanco 141,
76090 Colinas del Cimatario, Queretaro, Mexico
e-mail: jsalasr@ipn.mx

C. Tomasi
Department of Computer Science, Duke University,
Box 90129, Durham, NC 27708-0129, USA
e-mail: tomasi@cs.duke.edu

space requirements and LU matrix decomposition to speed up the solution of the linear system. Next, in Sect. 4, we describe some experiments. A concluding section discusses the results and suggests directions for future research.

2 The gray-world assumption

In the gray-world assumption [5], it is expected that the observed color components of a given scenario will average to gray. As a consequence, any observed deviation with respect to gray may be attributed to the illuminant. That is, let the normalized representation of a color image and its observed space average color be $C(\mathbf{x}) = \{C_i(\mathbf{x})\}$ and $A(\mathbf{x}) = \{A_i(\mathbf{x})\}$, respectively, for $i = \{r, g, b\}$, $0 \leq A_i(\mathbf{x}) \leq 1$ and $0 \leq C_i(\mathbf{x}) \leq 1$. We are going to assume that A_i and C_i are represented by matrices with dimensions h rows by w columns, where $\kappa = hw$. Furthermore, suppose that $\tilde{\mathbf{c}} = (c_r, c_g, c_b)^T$ and $\tilde{\mathbf{a}} = (a_r, a_g, a_b)^T$ are particular pixels of the color image C and space average color image A, respectively. Under the gray-world assumption, when the scene is lit with an illuminant different than white, $\tilde{\mathbf{a}}$ will undergo a deviation with respect to the gray line, represented by the unitary vector $\mathbf{w} = (1, 1, 1)^T/\sqrt{3}$, that can be expressed as

$$\tilde{\mathbf{a}}_\perp = \tilde{\mathbf{a}} - (\tilde{\mathbf{a}}^T \mathbf{w})\mathbf{w}. \tag{1}$$

Thus, the correction $\tilde{\mathbf{o}}$ of the color $\tilde{\mathbf{c}}$ consists of compensating for the illuminant as

$$\tilde{\mathbf{o}} = \tilde{\mathbf{c}} - \tilde{\mathbf{a}}_\perp. \tag{2}$$

It is rarely the case that the illumination in the image is spatially uniform. Ebner [6–10] addressed this problem by computing a local average of the color before applying the gray-world assumption. That is, given an initial estimate of the local average of a color band a_i, and an observation of a color component c_i, Ebner proposed the following iterative relation to compute the local space average:

$$A_i^{'}(\mathbf{x}) = \frac{1}{\mid N(\mathbf{x}) \mid} \sum_{\mathbf{x}' \in N(\mathbf{x})} A_i(\mathbf{x}'), \tag{3}$$

where the average $A_i(\mathbf{x})$ is updated as

$$A_i(\mathbf{x}) = (1 - \rho)A_i^{'}(\mathbf{x}) + \rho C_i(\mathbf{x}), \tag{4}$$

and $0 < \rho \ll 1$ is a small positive constant. In this study, we assume absorbing conditions for points at the boundary, i.e., to compute the local average for pixels at the border, we use only the pixels inside the image limits. To compute the local space average, Ebner proposed either to build a resistive grid [10] or to execute (3) and (4) iteratively for a number of cycles in the order of the tens of thousands. In [10], Ebner also suggested the use of successive over relaxation (SOR) [22]. In the next section, we show that this formulation admits an alternative form of solution based on its representation as a linear system. Furthermore, we show that the alternative is faster and, for a given resolution, can be computed in a fixed number of operations.

3 A linear system alternative

Without loss of generality, the elements in A_i and C_i can be ordered in column-wise order, resulting respectively in the vectors $\mathbf{a} = \{a_j\}$ and $\mathbf{c} = \{c_j\}$. This way, (3) and (4) can be expressed as

$$\mathbf{a}^{(r)} = (1 - \rho)P\mathbf{a}^{(r-1)} + \rho\mathbf{c}, \tag{5}$$

where $\mathbf{a}^{(0)}$, for $r = 1$, corresponds to the initial estimate of the value of the local space average and $P = \{p_{ji}\}$ is a weighted connectivity matrix [21], with the property $\sum_{i=1}^{\kappa} p_{ji} = 1$. The second iteration can be expressed as

$$\mathbf{a}^{(2)} = (1 - \rho)P\mathbf{a}^{(1)} + \rho\mathbf{c}. \tag{6}$$

To solve for $\mathbf{a}^{(2)}$ we can plug-in (5), for $r = 1$, into (6). After some algebra, it results in

$$\mathbf{a}^{(2)} = (1 - \rho)^2 P^2\mathbf{a}^{(0)} + (1 - \rho)\rho P\mathbf{c} + \rho\mathbf{c}, \tag{7}$$

where P^r is the r-step weighted connectivity or the weight associated in taking into account any two pixels k and j in the rth iteration.

It is easy to appreciate that the pattern that is emerging has the general form

$$\mathbf{a}^{(n)} = (1 - \rho)^n P^n\mathbf{a}^{(0)} + \rho\left(\sum_{i=0}^{n-1}(1 - \rho)^i P^i\right)\mathbf{c}. \tag{8}$$

We are interested in the behavior of (8) as n tends to infinity. In these conditions, the first half vanishes because $(1 - \rho)^n$ becomes zero. Among other things, it means that the final local space average does not depend on the initial estimated value of it. As for the summation, it can be simplified by subtracting $\mathbf{a}^{(n)}$ from $(1-\rho)P\mathbf{a}^{(n)}$. That is, let the expansion of $\mathbf{a}^{(n)}$ be expressed as

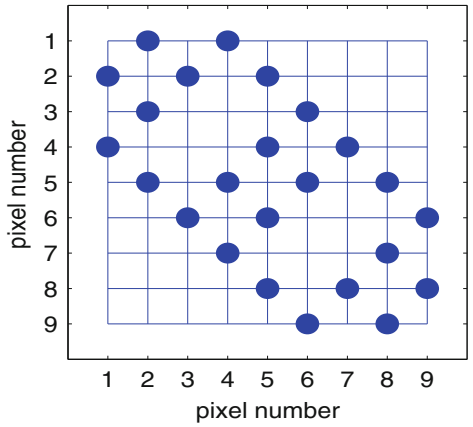
$$\lim_{n\to\infty} \mathbf{a}^{(n)} = \rho(I + (1-\rho)P + \cdots + (1-\rho)^{(n-1)}P^{(n-1)})\mathbf{c}. \tag{9}$$

Then, we multiply by $(1 - \rho)P$, resulting in

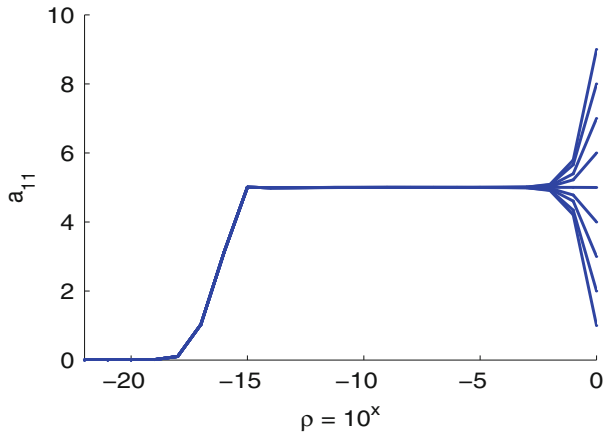
$$\lim_{n\to\infty} (1 - \rho)P\mathbf{a}^{(n)} = \rho((1-\rho)P + (1-\rho)^2P^2 + \cdots + (1-\rho)^nP^n)\mathbf{c}. \tag{10}$$

The subtraction of (9) from (10) will eliminate most of the terms except those at the extremes. Factoring out $\mathbf{a}^{(n)}$, this results in

(a) 3 × 3 image.



(b) Connectivity matrix P.



(c) Convergency toward the average.

Fig. 1 Local space average. In **a**, the number in the *middle of each square* represents the intensity value and the position in row-wise ordering. The connectivity used to compute the average is illustrated in **b**. Note that the central pixel is not used in this example. For instance, to compute the average of pixel *1*, the pixels *2* and *4* are used. The weights are uniform and have to add to 1. In **c**, we show the convergence for different values of ρ, in logarithmic scale. The *values on the right* are the starting values. At $\rho = 10^{-4}$, the largest difference with respect to 5 is smaller than 10^{-3}. For values below 10^{-12}, the values start to diverge from 5, and for values below 10^{-15}, the average breaks down due to numerical round off

$$\lim_{n\to\infty}(I-(1-\rho)P)\mathbf{a}^{(n)}=\rho(I-(1-\rho)^n P^n)\mathbf{c}=\rho\mathbf{c}, \tag{11}$$

where the term $(1-\rho)^n P^n$ vanishes when $n \to \infty$. Finally, we arrive at the expression

$$\lim_{n\to\infty}\mathbf{a}^{(n)} = \rho(I-(1-\rho)P)^{-1}\mathbf{c} = \rho K^{-1}\mathbf{c}. \tag{12}$$

Note that, in practice, the matrix K^{-1} is actually not computed. Instead, one can take advantage of the structure and sparsity of $K = \{k_{ij}\}$. As it can be appreciated in Fig. 1b, K is banded, i.e., $k_{ij} = 0$ for $j > i + q$ and $i > j + q$. Golub and van Loan [14] show that the LU factorization maintains the sparsity on L and U. They also demonstrate how to compute the factorization in $2\kappa q^2$ flops, and both the forward and backward substitution in $4\kappa q$ flops. On the other hand, to store the elements of the band requires $(2q+1)\times\kappa$ cells. Furthermore, it should be stressed that L and U need to be computed only once for a particular average definition and image resolution.

3.1 Speeding up calculations

In [10], Ebner offers an alternative to speed up the computation of the local space average color based on the use of SOR [22]. SOR techniques aim to accelerate the rate of convergence by introducing a factor ω that increases the value of the estimate in the direction of the solution. Ebner's iterative scheme can be described by

$$A_i'(\mathbf{x}) = \frac{1}{\mid N(\mathbf{x})\mid}\sum_{\mathbf{x}'\in N(\mathbf{x})}\bar{A}_i(\mathbf{x}'), \tag{13}$$

where now $\bar{A}_i$ is the local space average and $A_i(\mathbf{x})$ is an auxiliary variable that is updated as

$$A_i(\mathbf{x}) = (1-\rho)A_i'(\mathbf{x}) + \rho C_i(\mathbf{x}), \tag{14}$$

and the SOR equation is defined as

$$\bar{A}_i(\mathbf{x}) = (1-\omega)\bar{A}_i(\mathbf{x}) + \omega A_i(\mathbf{x}). \tag{15}$$

After ordering A_i and $\bar{A}_i$ in column-wise form as $\mathbf{a}$ and $\bar{\mathbf{a}}$, respectively, (14) and (15) can be represented as

$$\mathbf{a}^{(r-1)} = (1-\rho)P\bar{\mathbf{a}}^{(r-1)} + \rho\mathbf{c}, \tag{16}$$

and

$$\bar{\mathbf{a}}^{(r)} = (1-\omega)\bar{\mathbf{a}}^{(r-1)} + \omega\mathbf{a}^{(r-1)}. \tag{17}$$

And the first iteration yields

$$\bar{\mathbf{a}}^{(1)} = ((1-\omega)I + \omega(1-\rho)P)\bar{\mathbf{a}}^{(0)} + \rho\omega\mathbf{c}. \tag{18}$$

We can continue the same process as before, expanding successive iterations and simplifying the expressions. At the end, nth iteration yields the form

$$\begin{aligned}\mathbf{a}^{(n)} &= ((1-\omega)I + \omega(1-\rho)P)^n\mathbf{a}^{(0)} \\ &\quad + \omega\rho\sum_{i=0}^{n-1}((1-\omega)I + \omega(1-\rho)P)^i\mathbf{c}.\end{aligned} \tag{19}$$

When we analyze (19) for $n \to \infty$, we notice that the first term is not guaranteed to fade unless $0 \leq \omega \leq 1$. Provided that, we can solve for $\mathbf{a}^{(n)}$ by subtracting it from $((1-\omega)I + \omega(1-\rho)P)\mathbf{a}^{(n)}$. Using the conditions for ω previously stated, this gives the solution

$$\begin{aligned}&\lim_{n\to\infty}\ (I - ((1-\omega)I + \omega(1-\rho)P))\mathbf{a}^{(n)} \\ &\qquad = \omega\rho(I - ((1-\omega)I + \omega(1-\rho)P)^n)\mathbf{c} = \omega\rho\mathbf{c}.\end{aligned} \tag{20}$$

In other words,

$$\lim_{n\to\infty}\mathbf{a}^{(n)} = (I - ((1-\omega)I + \omega(1-\rho)P))^{-1}\omega\rho\mathbf{c}. \tag{21}$$

For simplicity, we prefer (12) for all of our computations.

4 Experimental results

The toy example in Fig. 1 illustrates some considerations in the construction of the weighted connectivity matrix $P = \{p_{ji}\}$, specially for issues related to boundary conditions. A 3×3 matrix is filled up with values from 1 to 9, with 5 in the center. Although for this example the central pixel is not included, for subsequent tests, we use a 4-connected neighborhood, with the central pixel included in order to compute the average in (3). The following describes the conditions at the borders. For the computation of $P = p_{ji}$, and depending on the number of neighbors involved, there are three cases. For the pixels on the corners, the vertical and horizontal neighbors are used, and the weight for each element is 1/2. For the pixels in the horizontal borders at the top and at the bottom rows, three pixels are used: the two lateral pixels and the vertical neighbor, and so a weight of 1/3 is used. The same rule applies for the vertical borders, the leftmost and the rightmost columns. In this case, three pixels are used, the lateral and the two vertical neighbors. The weight for each element is also 1/3. Finally, for the pixels in the center, a 4-connected neighborhood is used and the weight for each element is 1/4.

The matrix K can be stored compactly in a $5 \times \kappa$ array, i.e., the central diagonal and two off-diagonal elements on each side. However, L and U require additional non-zero entries of size $\kappa \times (h+1)$ [18]. This is important because carelessly defined, an $\kappa = h \times w$ image will require as much as $\kappa \times \kappa$ cells to hold the weighted connectivity matrix P, i.e., four times as much space as the original image. Nonetheless, note that the vast majority of the entries of the linear system are zero and the non-zero entries are in or close to the main diagonal. The problem of space can be solved using sparse representation for matrices. As for the value of ρ, Ebner [10] points out that it determines the radius over which local space average color is computed. Larger values of ρ use small neighborhoods whereas smaller values use larger ones. Figure 1c shows that for our toy example, the ratio of convergence spans between 10^{-2} and 10^{-15}. Out of this range, the numerical stability essentially breaks down. In a more realistic exercise, we obtained permission to use the images in [8] and applied our method. The color images have resolution 188 × 293 (rows × cols) `pixels`. We implemented Ebner's methods to compute the local space average defined by (3), (4) and (13)–(15) in Matlab. In both cases, we used $\rho = 10^{-4}$ and a maximum number of iterations equal 10,000. Convergency was defined by the sum of the squared difference between the current and previous estimate of the local space average, with a value of $\epsilon = 10^{-7}$. In the case of the SOR version, we tried several values of ω. In Fig. 2a, we ran 30 times the three algorithms. The average time for (3) and (4) was 95.95 s. For (13)–(15), the experimental results are more complicated. Depending on the particular values of ρ and ω, the expressions may or may not converge. For instance, at $\rho = 10^{-4}$, 28,613 iterations have to be executed for $\omega = 0.1$ and 9,364 iterations at $\omega = 1.3$ to reach convergency (although Ebner recommends $\omega > 1$ to achieve faster convergence). On the other hand, for $\rho = 10^{-3}$, the method requires 12,458 iterations for $\omega = 0.1$ and 2,333 iterations for $\omega = 1.2$. For values of $\omega > 1.3$, the method diverged for both values of ρ. As for our method, the average computing time was 1.97 s with $\rho = 10^{-4}$. In Fig. 2, we show the results for other images. To evaluate quantitatively the relative change from the input to the output, we computed the root mean squared error (RMSE), in CIE La*b* color space, for the images in Fig. 2. For the input, the first column in Fig. 2, the RMSE between the first image and the second one, the first one and the third one, and the second one and the third one is 6.32, 10.3, and 33.74, respectively. As for the third column, the output, the RMSE is correspondingly 1.16, 2.20, and 1.04. As expected, the resulting images are more similar to each other.

5 Conclusion

In this document, we present an alternative solution to the iterative formulation introduced by Ebner [6–10] to compute

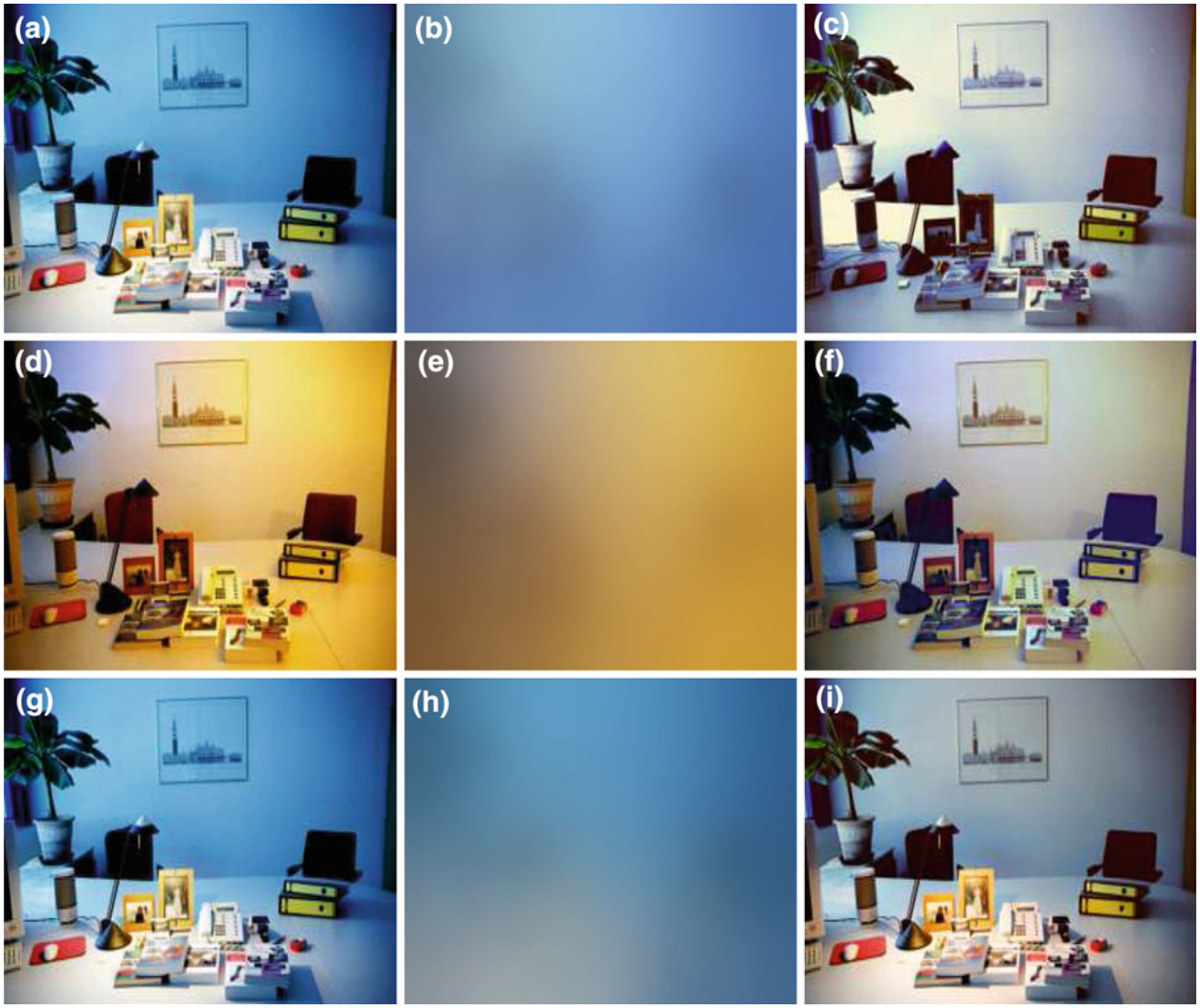


Fig. 2 Office environment. Results of the algorithm for the same set of images used by Ebner [8]. The *columns* show the original, the local space color average, and the corrected image. In the experiments, $\rho = 0.0001$. With kind permission from Springer Science and Marc Ebner

the local space average color in an image. We referenced the requirements of the solution in terms of the space needed in a sparse matrix representation, and the computing complexity, in terms of the number of flops. Our results show that our implementation requires at least an order of magnitude less time to execute.

The gray-world assumption is a bold statement about the nature of the world that clearly may not hold up in some typical scenarios. Nevertheless, as our knowledge about how the physical world operates advances and more complete computational models are developed, these assumptions may prove useful in gaining the necessary intuition to solve the problem. Despite this, the algorithm presented here shall only be used with the necessary careful assessment of the characteristics of the particular scenario.

Acknowledgments Thanks to Marc Ebner for providing the original images for Fig. 2, to Mary Masterman for editing the document, and to the reviewers whose comments improved greatly the quality of our exposition. This work was partially supported by the Fomix CONACYT-DF under Grant No. 189005 and the Instituto Politecnico Nacional under Grant No. 20131832 for Joaquin Salas, and the National Science Foundation under Grant No. IIS-1017017 and by the Army Research Office under Grant No. W911NF-10-1-0387 for Carlo Tomasi.

References

1. Barnard, K., Cardei, V., Funt, B.: A comparison of computational color constancy algorithms. I: methodology and experiments with synthesized data. IEEE Trans. Image Process. **11**(9), 972–984 (2002)
2. Barnard, K., Finlayson, G., Funt, B.: Colour constancy for scenes with varying illumination. In: ECCV, pp. 1–15 (1996)

3. Barnard, K., Martin, L., Coath, A., Funt, B.: A comparison of computational color constancy algorithms. II. Experiments with image data. IEEE Trans. Image Process. **11**(9), 985–996 (2002)
4. Bianco, S., Ciocca, G., Cusano, C., Schettini, R.: Automatic color constancy algorithm selection and combination. Pattern Recognit. **43**(3), 695–705 (2010)
5. Buchsbaum, G.: A spatial processor model for object colour perception. J. Frankl. Inst. **310**(1), 1–26 (1980)
6. Ebner, M.: Combining white-patch Retinex and the gray world assumption to achieve color constancy for multiple illuminants. In: Pattern Recognition. Lecture Notes in Computer Science, vol 2781, pp 60–67 (2003)
7. Ebner, M.: A parallel algorithm for color constancy. J. Parallel Distrib. Comput. **64**(1), 79–88 (2004)
8. Ebner, M.: Color constancy using local color shifts. In: ECCV, pp. 276–287 (2004)
9. Ebner, M.: Color Constancy. Wiley, West Sussex (2007)
10. Ebner, M.: Color constancy based on local space average color. Mach. Vis. Appl. **20**(5), 283–301 (2009)
11. Finlayson, G., Funt, B., Barnard, K.: Color constancy under varying illumination. In: ICCV, p. 720 (1995)
12. Finlayson, G., Schiele, B., Crowley, J.: Comprehensive colour image normalization. In: ECCV, pp. 475–490 (1998)
13. Gijsenij, A., Gevers, T., van de Weijer, J.: Computational color constancy: survey and experiments. IEEE Trans. Image Process. **20**(9), 2475–2489 (2011)
14. Golub, G., van Loan, C.: Matrix Computations. Johns Hopkins University Press, Baltimore (1996)
15. Livingstone, M.: Vision and Art: the Biology of Seeing. Abrams, New York (2008)
16. McCann, J.: HDR imaging and color constancy: two sides of the same coin? In: SPIE, vol. 7866, p. 22 (2011)
17. Palma-Amestoy, R., Provenzi, E., Bertalmío, M., Caselles, V.: A perceptually inspired variational framework for color enhancement. IEEE Trans. Pattern Anal. Mach. Intell. **31**(3), 458–474 (2009)
18. Press, W., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical Recipes, vol. 3. Cambridge University Press, New York (2007)
19. Provenzi, E., Gatta, C., Fierro, M., Rizzi, A.: A spatially variant white-patch and gray-world method for color image enhancement driven by local contrast. IEEE Trans. Pattern Anal. Mach. Intell. **30**, 1757–1770 (2007)
20. Wandell, B.: Foundations of Vision. Sinauer Associates, Sunderland (1995)
21. West, D.: Introduction to Graph Theory, vol. 1. Prentice Hall (2001)
22. Young Jr., D.: Iterative Methods for Solving Partial Difference Equations of Elliptic Type. Tech. rep. Harvard University (1950)

Author Biographies

Joaquin Salas is professor at the Instituto Politécnico Nacional. His research focuses in computer vision, where he has published 50 articles in journals and international conferences. He has been visiting scholar at the École Nationale Supérieure des Télécommunications de Bretagne, Stanford University, Oregon State University, the Universitat Autónoma de Barcelona, Xerox PARC, and Duke University. He has led several research and applied projects for industry. For his professional activity, he received the Lázaro Cárdenas del Río medal from the President of Mexico.

Carlo Tomasi is professor of computer science and department chair at Duke University. He has worked in computer vision for about 25 years at Carnegie Mellon, Cornell, Stanford, Duke, and in industry. According to Google Scholar, his top five publications have been cited more than 12,000 times in total. He has been principal investigator or co-investigator on 28 peer-reviewed grants. He has supervised many graduate and undergraduate students on a variety of research projects in image motion analysis, stereo vision, image retrieval, medical imaging, and general image understanding. He has developed and taught courses in computer vision, applied mathematics, mathematical modeling, discrete mathematics, and artificial intelligence.