Color Stabilization Along Time and Across Shots of the Same Scene, for One or Several Cameras of Unknown Specifications

Javier Vazquez-Corral and Marcelo Bertalmío

***Abstract*—We propose a method for color stabilization of shots of the same scene, taken under the same illumination, where one image is chosen as reference and one or several other images are modified so that their colors match those of the reference. We make use of two crucial but often overlooked observations: firstly, that the core of the color correction chain in a digital camera is simply a multiplication by a 3x3 matrix; secondly, that to color-match a source image to a reference image we don't need to compute their two color correction matrices, it's enough to compute the operation that transforms one matrix into the other. This operation is a 3x3 matrix as well, which we call H. Once we have H, we just multiply by it each pixel value of the source and obtain an image which matches in color the reference. To compute H we only require a set of pixel correspondences, we don't need any information about the cameras used, neither models nor specifications or parameter values. We propose an implementation of our framework which is very simple and fast, and show how it can be successfully employed in a number of situations, comparing favourably with the state of the art. There is a wide range of applications of our technique, both for amateur and professional photography and video: color matching for multi-camera TV broadcasts, color matching for 3D cinema, color stabilization for amateur video, etc.**

I. Introduction

We expect two pictures of the same scene, taken under the same illumination, to be consistent in terms of color. But if we have used different cameras to take the pictures, or just a single camera with automatic white balance (AWB) and/or automatic exposure (AE) correction, then the most common situation is that there are objects in the scene for which the color appearance is different in the two shots.

This is problematic in many contexts. With a single camera, the only way to ensure that all pictures of the same scene are color consistent would be to save images in the RAW format, or to use the same set of manually fixed parameters for all the shots. These are not common choices for amateur users, but even professional users face the same challenges: the most popular DSLR cameras for shooting HD video don't have the option of recording in RAW [4]; and while in cinema the exposure and color balance values are always kept constant for the duration of a take (i.e. AE and AWB are never used), the shooting conditions may require to change these values from shot to shot. With different cameras the problem is aggravated, because using the same parameter values in all cameras is not enough to guarantee the stability of color across shots [16]. In many professional situations several cameras are used at the same time (e.g. large photo shoots, many mid-scale and most large-scale cinema productions), and in some cases the multi-camera set-up is required, not optional. For instance, TV broadcasts employ devices called camera control units, operated by a technician called Video Controller or Technical Director (TD): while each camera operator controls most of his/her camera functions such as framing or focus, the TD controls the color balance and shutter speed of a set of cameras so as to ensure color consistency across them [26].

Authors are with the Departament de Tecnologies de la Informació i les Comunicacions, Universitat Pompeu Fabra, Barcelona, Spain, {javier.vazquez,marcelo.bertalmio}@upf.edu

We can see then that an automatic color stabilization procedure would be both a very useful tool for amateur users and a key asset for the industry. Our contribution is to propose a framework for color stabilization of shots of the same scene, taken under the same illumination, where one image is taken as reference and one or several other images are modified so that their colors match those of the reference. This framework is very simple and does not require calibrated cameras nor any information about the cameras used, neither specifications nor camera parameters. We make use of two crucial but often overlooked observations: firstly, that the core of the color correction chain in a digital camera is simply a multiplication by a 3x3 matrix; secondly, that to color-match a source image to a reference image we don't need to compute their two color correction matrices, it's enough to compute the operation that transforms one matrix into the other, and this only requires a set of pixel correspondences. Although we rely on the estimation of a 3x3 matrix, what we propose is *not* a color constancy method: we do not try to recover reflectances neither illuminants, but to remove color fluctuations, making all images look the same in terms of color (even if those colors correspond to an incorrect white balance procedure). We show how our approach can be successfully employed in a number of situations, comparing favourably with the state of the art: single-camera stills or video with AE and AWB, single-camera with changing color temperature or *type of scene* presets, twin-camera shots for 3D cinema, multi-camera shots.

II. Related Work

A pioneering color stabilization method for video is presented in [5] by Farbman and Lischinksky: some frames are designated as anchors, and the rest are color-adjusted to them. This adjustment is performed with a dense map containing color differences, computed from a set of corresponding pixels and

after interpolation in color space, treating channels independently and not using any imaging model for the camera. The shown results are excellent, but the method is based on a strong temporal coherence assumption so the authors report it can't be used when there are considerable changes from one video frame to the next. For this same reason, it wouldn't appear to be suited for color matching two still pictures with wide-baseline views of the same scene, and in any case the authors restrict the application of their method to video.

In [13] Kim et al. propose an in-camera imaging model from which they derive camera calibration procedures to determine the camera response function, color transformation matrix and so on. This information allows them to perform a number of tasks with outstanding results, including color transfer between pictures taken with the same or different, *but known and calibrated,* cameras. Therefore, the main limitation of this approach is that it requires a camera calibration stage, it just can't be applied to pictures taken by cameras of which we know nothing: for each camera model a number of training images, each recorded both in JPG and RAW formats, is needed, and for cameras without RAW support they have to use a RAW image of the same scene but taken with another camera as reference. Also, another limitation of this technique is that correction of differences due to AWB requires user assistance.

In the academic literature there is abundant work on the more general problem of color transfer among images, and in a recent article [20] Reinhard surveys methods for this problem, mentioning essentially four types of approaches: warping clusters or probability distributions in a three-dimensional color space, as in Pitié et al. [18]; matching means and standard deviations in an appropriate, de-correlated color space, as in Reinhard et al. [21]; aligning and scaling principal axes of the color distributions, as in the work [14] by Kotera; and straightforward histogram matching, performed independently on each color channel of an original or de-correlated color space. These are all *global* methods, i.e. they modify the value of each pixel ignoring its location and hence also ignoring the values of the neighboring pixels. Recent methods use pixel correspondences to identify similar regions between the image pair and then refine the color matching transform which still is global, as in the work by Kagarlitsky et al. [12]. Local methods have been proposed by Tai et al. [24], where the images are segmented into regions in which color distributions are represented as Gaussian mixtures and then matched, and by Huang and Chen [11], where colors are transferred among corresponding landmarks and then extended to the rest of the image. While these methods provide very good results in a variety of situations, they are also much more involved than global approaches and rely on solutions to challenging tasks such as image segmentation. The state of the art in color transfer is the technique by HaCohen et al. [9], a remarkable work where a dense correspondence map is computed and applied to a variety of situations, including global and local color transfer; recently, this approach has been extended to improve color consistency in photo albums [10]. The results are excellent, although no quantitative evaluation is performed for the color transfer application and the authors report some limitations of their approach, including the difficulty of finding reliable correspondences in very large smooth regions or handling scenes with strong lighting changes because they fit one single color model to the whole scene.

III. Color stabilization

A. In-camera color processing chain

Before we introduce our approach, let us review the elements of a digital camera's color processing chain [2]:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_{out} = \left(\alpha ECD \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{in} \right)^{\gamma} \tag{1}$$

where RGB_{in} is the camera raw triplet at a given pixel location, to which a diagonal white balance matrix D is applied, followed by the colorimetric matrix C (that transforms an (R, G, B) triplet into its corresponding (X, Y, Z) tristimulus value), followed by the color encoding matrix E (that converts (X, Y, Z) tristimulus values into a standard RGB color space like sRGB, for display purposes), followed by a gain constant α, and finally a power function of exponent γ is applied. All three matrices D, C and E are 3x3. In [17] it is pointed out that the RGB_{in} camera raw triplet is usually not the original sensor signal but a corrected version of it, where the original nonlinear camera exposures have been linearized through a LUT; thus, we can assume that all RGB_{in} triplets are actually proportional to the exposures. The final gamma correction power function is also implemented with a LUT, and since the color transformation matrices can be cascaded into a single 3x3 matrix, the whole color processing pipeline can be expressed as a LUT-matrix-LUT sequence [17]:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_{out} = \left(A \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{in} \right)^{\gamma} \tag{2}$$

This model is an approximation: in practice the curve of the gamma correction is slightly modified so that it doesn't have infinite gain near black [19], and therefore it's not exactly a power law, and also all RGB_{out} values are clipped or non-linearly mapped into a predefined range (e.g. [0,255] in sRGB) which introduces non-linearities in the color correction procedure, as Kim et al. point out [13]. Furthermore, RGB_{out} is the output of this color correction pipeline but not the actual triplet value recorded by the camera, because there still remain some image processing operations in the full camera pipeline, e.g. edge enhancement, denoising, compression, which will alter the final values.

The colorimetric matrix is a function of the scene illuminant, so ideally a different matrix should be used for each different scene illuminant the camera is working with [17]. Many cameras come with several pre-set matrices computed under different illuminations. For instance, using the matrix for

fluorescent lighting removes a noticeable green cast that would otherwise be present if we used a matrix computed with a standard illuminant like D65 or D50; other pre-sets may correspond for instance to a "film look" (with de-saturated colors), or may give a very vivid color palette.

These pre-set matrices can also be adjusted manually so as to achieve a certain image look, since changing the colorimetric matrix affects hue and saturation. But for our problem there is a stunning, key observation that we can make from learning what *the original purpose* was of modifying a colorimetric matrix: to allow cameras to be "color matched" so that there were no color shifts when cutting between multiple cameras during live broadcasts [1].

That is, changing the nine elements of a broadcast camera's color matrix is enough to match its shots to those taken by another camera, and for this the camera operators didn't need to determine which were the internal matrices E, C, D of either camera. This means two fundamental things for us, which will constitute the basis of our approach:

1) Color matching can be achieved with a 3x3 matrix.
2) We don't need to know the actual values of the matrices involved in the color processing chain.

B. Overview of proposed framework

Let's start by considering the case of a single camera. We take a picture, $\hat{I}_1$, change its settings, change its position, and take a second picture, $\hat{I}_2$. We are also assuming that the illumination doesn't change, so if p is a scene point appearing in both images then it has produced in the sensor the same triplet $(R,G,B)_p$ in both pictures. But, given that we have changed both camera position and settings, point p will very probably appear at different locations and with different pixel values, $(R,G,B)_{p1}$ and $(R,G,B)_{p2}$, in the two images:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix}_{p1} = \left(A_1 \begin{bmatrix} R \\ G \\ B \end{bmatrix}_p \right)^{\gamma_1} ; \quad \begin{bmatrix} R \\ G \\ B \end{bmatrix}_{p2} = \left(A_2 \begin{bmatrix} R \\ G \\ B \end{bmatrix}_p \right)^{\gamma_2} \tag{3}$$

Although the parameter γ is a known industry standard (e.g. $\gamma = \frac{1}{2.2}$ in sRGB, $\gamma = \frac{1}{2.0}$ in BT.709 for HDTV, $\gamma = \frac{1}{2.6}$ in digital cinema, etc.) in practice it is usually changed, e.g. by the camera itself depending on the picture style [13], or by the technical director in charge of a camera control unit at a live broadcast [23]. But it is important to note that, as Thomas et al. point out in [23], processing of gamma-corrected (instead of linear or linearized) images isn't usually problematic. In practice we don't need to very accurately estimate the values γ_1 and γ_2: assuming for both the industry standard value already gives a reasonable result for our problem. This result can be improved using an estimate of the ratio $\frac{\gamma_1}{\gamma_2}$ (and we will introduce a novel procedure to perform this estimate,) but no gain seems to be obtained by using the actual values for both γ_1 and γ_2: just setting the smaller of the two values at 2.2, and using the estimated ratio to compute the other gamma value, is enough to give results that compare favorably with the state of the art and which also do not appear to improve if more accurate estimates of γ_1 and γ_2 are used, as we shall see in more detail in the implementation and experiments sections.

For the above reasons, then, let's assume that we have an estimate of the gamma values so we can undo the gamma correction:

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p1} = A_1 \begin{bmatrix} R \\ G \\ B \end{bmatrix}_p ; \quad \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p2} = A_2 \begin{bmatrix} R \\ G \\ B \end{bmatrix}_p \tag{4}$$

Now we can see that:

$$ {A_1}^{-1} \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p1} = \begin{bmatrix} R \\ G \\ B \end{bmatrix}_p = {A_2}^{-1} \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p2} , \tag{5}$$

therefore:

$$ A_1 {A_2}^{-1} \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p2} = \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p1} . \tag{6}$$

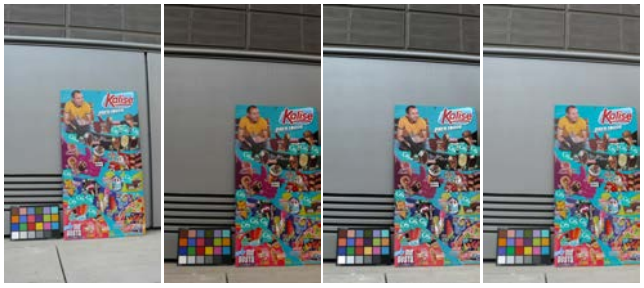
Let H be the unknown 3x3 matrix $H = A_1 {A_2}^{-1}$, then we have:

$$ \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p1} - H \begin{bmatrix} r \\ g \\ b \end{bmatrix}_{p2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} , \tag{7}$$

and this relationship holds for all pairs $p1$, $p2$ of corresponding pixels. Therefore, each pair of pixel correspondences gives us an equation of the form of (7), and we can find H by solving a system of equations. We only have 9 unknowns for H but typically we have many pixel correspondences, so the system is overdetermined and the solution can be found by an optimization procedure (since the simplest one, least squares minimization, is sensitive to outliers, we use a more robust technique as we explain later).

Once H has been found, we simply need to apply it to each pixel in $\hat{I}_2$ so that it looks as in $\hat{I}_1$. And this holds for all pixels, not just those for which we have found correspondences: the correspondences are used to estimate H, but H is a global transform and when we have it we have all we need to make the images color consistent.

It is easy to see that the analysis above holds as well for the multi-camera case. One could argue that with different sensors and different CFA (Color Filter Array) filters we no longer can say that the registered value $(R,G,B)_p$ is the same in both cameras. This is true, but the main purpose of the colorimetric matrix, transforming (R,G,B) triplets into (X,Y,Z) values, is to standardize the captured colors, making them device independent. Therefore we can assume



(a) (b) (c) (d)

Fig. 1: *(a) Reference. (b) Source. (c) Our result assuming $\gamma_1 = \gamma_2 = 2.2$, error RMSE=9.31. (d) Our result after estimating the gamma values, error RMSE=5.08.*

that these sensor differences are being taken care of by the color processing chain, and for two cameras we have the same working model, that of equation (3).

Summarizing, our proposed framework consists of the following three steps:

1) Set values for γ_1 and γ_2.
2) Find pixel correspondences among $\hat{I}_1$ and $\hat{I}_2$.
3) Compute the color matching matrix H.

This is a framework based on an imaging model which is definitely limited in some regards, e.g. it isn't completely accurate to linearize the camera response using a simple gamma correction, and the 3x3 transformation matrix is only valid for colors in the center of the mapping function. But in the following sections we will show that, with an example and simple implementation of each of the three abovementioned steps, the proposed approach is capable of solving our problem. Alternative implementations of our framework, with more accurate techniques for some or all of the three required steps, could improve the results.

C. Implementation

1) Set values for γ_1 and γ_2: As mentioned earlier, a working solution is to assume the standard values for γ_1 and γ_2, such as 2.2 for $sRGB$, but we obtain better results if we estimate the actual values. This is shown in Figure 1: the error, measured as the RMSE (Root Mean Squared Error) over the color chart, decreases from 9.31 to 5.08 when we use values for γ_1 and γ_2 estimated with the method that we will now describe.

An accurate estimation of γ from a single image is an ill-defined problem and existing solutions (e.g. [6]) are lacking. But in our case we have two images of the same scene, and we have used this fact to devise a novel method to estimate simultaneously γ_1 and γ_2. Our method is the following:

- Find pixel matches (P_i, Q_i) between $\hat{I}_1$ and $\hat{I}_2$, i.e. $\hat{I}_1(P_i) \simeq \hat{I}_2(Q_i) \forall i$. For this we use SIFT [15]. We perform this procedure on the original images, ignoring the differences in gamma values and color appearance that they have, so the final set of pixel matches is just a rough approximation, good enough at this current stage. Later on, for the second step of our framework, we will perform a more accurate computation of pixel correspondences, see section III-C2.
- Choose a particular value for γ_1 and another for γ_2, and use them to linearize $\hat{I}_1(P_i)$ and $\hat{I}_2(Q_i)$. Now we are in the case of eq. 4, and we can compute the color matching matrix H by solving eq. 7 (for this we use the same method explained below for the third step of our color stabilization framework). We denote this matrix as H_{γ_1,γ_2} since it depends on the values for γ_1 and γ_2.
- We perform the above step for many (γ_1, γ_2) pairs, and choose the one that minimizes the error $|a_{\gamma_1,\gamma_2} - 1|$, where a_{γ_1,γ_2} is given by $\hat{I}_1(P_i)^{\frac{1}{\gamma_1}} = a_{\gamma_1,\gamma_2} H_{\gamma_1,\gamma_2} \hat{I}_2(Q_i)^{\frac{1}{\gamma_2}} + b_{\gamma_1,\gamma_2}$ (a_{γ_1,γ_2} is found by correlation over all pixel matches (P_i, Q_i)).

Figure 2 plots the error $|a_{\gamma_1,\gamma_2} - 1|$ (in log-scale) as a function of γ_1 and γ_2 for two images where one is taken with AWB on and the other has a "daylight scene" preset. We can see that the error is clearly much smaller over a certain line $\gamma_1 = k\gamma_2$, so in practice we set the smaller of the two gamma values to 2.2 and use the ratio k to determine the other gamma.

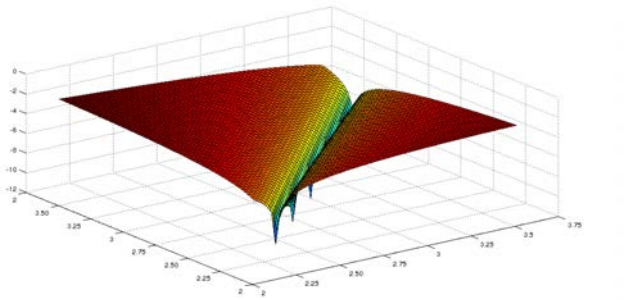


Fig. 2: *Error $|a_{\gamma_1,\gamma_2} - 1|$ (in log-scale) as a function of γ_1 and γ_2.*

We have run a test where we consider an input image pair and perform color transfer with our full method for many gamma values which vary over the line $\gamma_1 = k\gamma_2$. In this test we have the ground truth image so we can compute the Peak Signal to Noise Ratio (PSNR) of our result as a function of the gamma pair along the line, and this value is shown in figure 3: we can see that the PSNR is pretty stable on this optimum line.

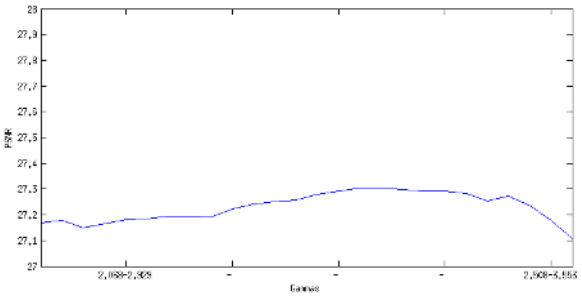


Fig. 3: *PSNR of the color stabilization result as a function of the gamma pair along the optimum line.*

We tested this method for estimating γ_1 and γ_2 over the multispectral image database of Foster et al. [8] using the 31 camera model specifications of Kim et al. [13], obtaining an extremely high correlation with the actual γ_1 and γ_2 values.

2) Find pixel correspondences among linearized $\hat{I}_1$ and $\hat{I}_2$: Once we have γ_1 and γ_2 we can undo the gamma correction, as explained in section III-B, obtaining the linearized images I_1 and I_2. Next we compute pixel correspondences with SIFT [15], using the implementation of [25] with their default choice of parameters. To avoid the possible inaccuracies of the SIFT matches we first compute smoothed versions of the original images, after convolution with a Gaussian of $\sigma = 5$. By doing this, we make the matches steady even if they have been found in textured regions or corners. We also refine these correspondences with RANSAC [7], removing matches that lead to a transformation matrix H that produces large color errors, e.g. if one pixel is on a diffuse surface and its match lies on the highlight of a specular reflection, or if there is a large difference between the dynamic range of the two cameras so one shot has for instance a completely overexposed sky which matches blue tones in the other image. These matches originate H matrices which are outliers, and therefore they are cancelled out by RANSAC. Our parameter values for RANSAC are: 5 points to fit the model, 1000 iterations, a threshold value of 0.09 for determining when a datum fits a model, and 0.001 as the minimum length of the solution vector. As a further refinement, we remove those correspondences that suppose an angular motion more than 20 degrees larger or smaller than the median average. The corrective effect provided by RANSAC (and by the angular motion procedure) over the original SIFT matches is presented in Figure 4, which shows how incorrect correspondences obtained by SIFT result in poor color stabilization results. In this example, by using only SIFT we get an incorrect yellow tone on the propeller (see detail), whereas this color is properly matched from the reference if we apply the RANSAC and angular motion corrections.

3) Compute the color matching matrix H: We look for H by solving a system of equations which is the same as that of eq. 7, but instead of the (r, g, b) values at the point matches we use more robust measures, that are less sensitive to variations in the localization of the keypoints found by SIFT and which therefore allow us to obtain a more robust estimate of H: mean average and first SVD-eigenvector of neighborhood values. We will now present this procedure in detail.

Firstly, for each keypoint pixel $p_1 = (p_{1x}, p_{1y})$ and its corresponding match $p_2 = (p_{2x}, p_{2y})$, eq. 7 can be written as:

$$I_1(p_1) - HI_2(p_2) = \overrightarrow{0}, \tag{8}$$

where $I_1(p_1)$ is the column vector with the (r, g, b) values of image I_1 at pixel p_1 (likewise for I_2, p_2). We consider the square neighborhoods of half-size w centered at pixels p_1, p_2,

$$I_i^w(p_i) = \{I_i(x, y)| \ \ |x - p_{ix}| \leq w, |y - p_{iy}| \leq w\}, i = 1, 2, \tag{9}$$

and arrange these patches as matrices of size $3 \times (2w + 1)$ so that each column of $I_i^w(p_i)$ has the (r, g, b) values of a neighbor of p_i. Assuming that the neighborhoods of matching pixels are also similar, we may write

$$I_1^w(p_1) - HI_2^w(p_2) = \overrightarrow{0}, \tag{10}$$

therefore the mean average of the columns of each matrix $((m_r, m_g, m_b)_{p_i}^T, i = 1, 2)$ must also satisfy eq. 7:

$$\begin{bmatrix} m_r \\ m_g \\ m_b \end{bmatrix}_{p1} - H \begin{bmatrix} m_r \\ m_g \\ m_b \end{bmatrix}_{p2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{11}$$

Clearly, the solution to eq. 11 will be less affected by noise and errors in the localization of pixel matches than the solution to eq. 8.

Another robust measure is given by the first SVD left-singular vector of the neighborhood values, because it represents the principal axis of the 3D color histogram at the patch: in regions with several dominant colors, this axis remains stable under minor changes in keypoint location. Performing the singular value decomposition of $I_1^w(p_1)$, eq. 10 becomes:

$$USV^T - HI_2^w(p_2) = \overrightarrow{0}, \tag{12}$$

where U has dimensions 3×3, S is a rectangular diagonal $3 \times (2w + 1)$ matrix containing the singular values of $I_1^w(p_1)$ and V^T is a $(2w+1) \times (2w+1)$ matrix. Rearranging equation 12, we get

$$U = HI_2^w(p_2)(V^T)^{-1}S^+ \tag{13}$$

where $+$ represents the pseudo-inverse. This equation has the form

$$U = HW \tag{14}$$

where the 3 matrices U, H, W are of dimension 3×3. If the first singular value in S is somewhat larger than the second one we can assume that there are several dominant colors in the patch, therefore the first left-singular vector of the SVD will be robust under the presence of noise or small displacements of the patch (given by changes in the position of the keypoint). So, if the ratio of the first two singular values $\frac{S_{11}}{S_{22}}$ is larger than some threshold T, we relate the first column of U and the first column of W in eq. 14 again as in eq. 7:

$$\begin{bmatrix} U_{11} \\ U_{21} \\ U_{31} \end{bmatrix}_{p1} - H \begin{bmatrix} W_{11} \\ W_{21} \\ W_{31} \end{bmatrix}_{p2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{15}$$

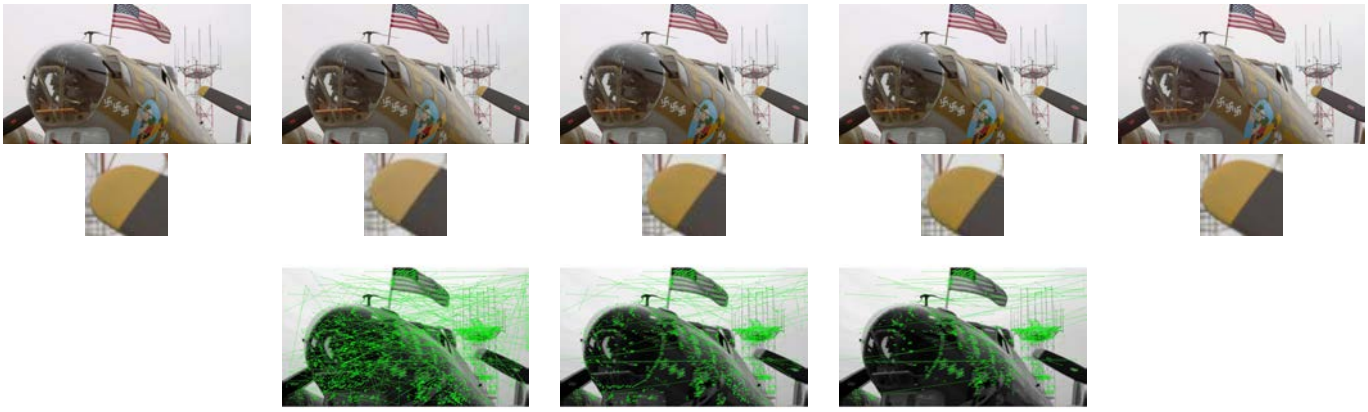


Fig. 4: *Effect of the use of RANSAC in our approach. Top row, from left to right: source, our result using only SIFT, our result using SIFT and RANSAC, our results using SIFT and RANSAC and angular motion, reference. Middle row: cropped detail from top row. Bottom row, from left to right: correspondences by using SIFT, by using SIFT and RANSAC, and by using SIFT, RANSAC and angular motion.*

Each pair of pixel correspodences will give us a set of equations as in eq. 11, and possibly one set of equations as in eq. 15 (if $\frac{S_{11}}{S_{22}} > T$). With all the resulting equations, obtained from all pixel matches, we build a system and solve for H. As we mentioned in section III-B, computing matrix H through least squares minimization is sensitive to outliers, so in order to make it more robust we may employ RANSAC as in section III-C2, now using a more restrictive threshold value of 0.05.

After finding H we apply it to all the pixels in the source image with the gamma correction un-done, and finally re-do the gamma correction. Values outside the $[0, 255]$ range are clipped. We can see that the only two parameters required by our algorithm are the patch half-width size w and the threshold ratio T.

For the numerical implementation we have chosen SIFT and RANSAC because they are simple, well known and very fast techniques. In our prototype Matlab code more than 85% of the execution time is due to SIFT and RANSAC, and both have real-time implementations [22], so this suggests the possibility that our method could be applied in real-time scenarios like live TV broadcasts.

IV. Results

We start this section with example results of our method for different situations, then we compare with the state of the art, and finally we present some figures that show the limitations of our method and possible improvements to it.

A. Seven different scenarios

In order to highlight the effectiveness of our approach, for all the examples in this section we have used the same parameter values: $w = 1, T = 4$.

1) *Color matching for professional 3D cinema*. We can apply our method to color correction in 3D cinema, as Fig. 5 shows. Despite the use of professional cameras of the same model and with the same parameters, still color differences appear and our method eliminates most of them.
2) *Using two known cameras*. In Figure 6, (a) to (c), we are matching the colors of a picture taken with a mobile phone camera (Samsung Galaxy Note, 8Mpixels) to those of an image taken with a photographic camera (Panasonic DMC-FZ45). Although the cameras are known we do not make use of their specifications or parameter settings, so this scenario is for us exactly the same as the following one.
3) *Using two unknown cameras*. In Figure 6, (d) to (f), (g) to (i), and (j) to (l) we are making consistent the colors of a pair of pictures of the same location taken in different moments, most probably violating our assumption that the illumination is the same for both images, but nevertheless the result is convincing. The original images (d),(e),(g),(h),(j),(k) were simply freely available web pictures from different individuals, taken with unknown camera models, so it wouldn't be possible to use for this case the state-of-the-art color transfer method based on the camera calibration procedure of Kim et al. [13].
4) *Varying color temperature*. In Figure 7, (a) to (c), our method successfully stabilizes colors after a change in the color temperature setting (manual white balance) of the camera.
5) *Varying "type of scene" camera-setting*. In Figure 7, (a), (d), and (e), our method is able to correctly match the colors to the reference, which has a different "type of scene" setting than the source image. Typical scene settings in consumer cameras include "sunny", "cloudy", "lightbulb", "fluorescent lighting", "sunset", etc.
6) *Amateur video with AWB and AE*. Figure 8 shows an example of a video recorded with AE and AWB on. Top: original. Bottom: our result, where all frames were color-matched to the leftmost frame. Notice how we are

able to stabilize the brightness of the wall on the right and its surroundings. Also notice that our method does not suffer from accumulation errors, since we select one frame as reference and we keep using it as long as there are enough correspondences. We must note that in this result the stabilization has caused some dark regions to become brighter (because they are brighter in the reference) and this may make the noise more apparent.

7) *Video with large temporal variation*. Our proposed method also works when the temporal coherence between frames is low: Figure 9 shows two stills from a video where the objects in the scene have moved substantially from one image to the other. With our technique we are able to restore the background tree to its reference colors.

B. Comparison with the state of the art

We have compared our technique with the state of the art methods of Kotera [14], Pitié et al. [18] and HaCohen et al. [9].

The PCA-based color transfer method by Kotera [14] is worth mentioning because in cinema post-production there are software packages where color grading is performed via PCA, with user assistance. This PCA method computes, like ours, a 3x3 matrix and applies it globally to the source image, but the principal components are estimated on 3D histograms and often there are problems due to mismatches of principal axes, like in the example shown in Figure 10.

Next we compare qualitatively our results with those of Pitié et al. [18] and HaCohen et al. [9], in Figures 11 and 12. In Figure 11 the differences are especially noticeable in the sky, which lacks contrast in the result of [9] and shows some pink/orange tones in the result of [18]. In Figure 12 we point to the colors of the woman's shirt and trousers, of the cupboards and of other objects in the back. These image regions do not appear in the reference, and hence they are more challenging for a method like [18] that relies on the statistics of the overlapping regions, and also for [9] which fits a single color model, computed on the overlapping regions, to the whole image.

We perform quantitative comparisons versus the methods of HaCohen et al. [9], Pitié et al. [18], and Kotera [14]. Visual results are presented in Figures 13, 14 and 15. In Figures 13 and 15 we have available the ground truth, which was obtained by taking a photo of the scene with the same camera and settings as those of the reference image, while for Figure 14 the quantitative evaluation is computed using the result by Kim et al. [13] as ground truth. Therefore, for each image in these figures we can compute the PSNR between the ground truth image and the output of each color transfer method:

$$PSNR = 10log_{10}\left(\frac{255^2}{MSE(I_{GT}, I_{CT})}\right), \qquad (16)$$

where I_{GT} is the ground truth image, I_{CT} is the color transfer result, and the MSE is averaged over the three color channels:

$$MSE(I, I') = \frac{1}{3}\sum_{c=R,G,B}\frac{1}{MN}\sum_{\substack{1\le i\le M\\ 1\le j\le N}}(I^c(i,j) - I'^c(i,j))^2, \qquad (17)$$

where I^c $(c = R, G, B)$ denotes channel c of image I.

We have fixed the parameters of our method in the following way: we have used the parameter values $w = 5$ and $T = 16$ for the odd rows of Figure 15 that correspond to a situation where we must match an image taken with "type of scene = sunset" to a reference taken with "type of scene = portrait; illuminant = sunny", and also for Figure 13 which represents as well a somewhat dark scene. In the even rows of Figure 15 where one image was taken with AWB on, and the other with "illuminant = sunny", and for Figure 14 which also represents a well-lit scene, we have used the parameter values $w = 9$ and $T = 2$ and run RANSAC only once, to discard incorrect matches.

Table I shows that, with the default choice of parameters, our results for the images from Figures 13, 14 and 15 are on average above the state of the art. We can vary w and T on a per-image basis to obtain even better PSNR numbers. For this we generate results for a fixed range of (w, T) values, find the three image results which are optimum according to three different measures, and select the best image among these three by visual inspection. We have chosen to use the following measures, computed as differences (accumulated over the keypoint neighborhoods) between the reference image and the result: difference in color, Euclidean histogram distance, and ratio between the mutual information of the two images and the entropy of the result image.

Finally, let us focus again on Figure 14. We recall how Kim et al. [13] perform camera calibration by taking test images with a given camera, after which they are able to do color matching, automatically if there has been no AWB, or with manual user assistance otherwise. Our result is very similar to the one of Kim et al. [13], with the advantage that our color matching process is "blind" (i.e. we don't need to know anything about the camera with which the picture was taken).

TABLE I: *Average PSNR for the images of Figures 13, 14 and 15*

Method	PSNR			
	Fig.12	Fig.13	Fig.14	Average
Proposed: changing parameters	**34.69**	27.09	**27.37**	**27.94**
Proposed: parameters fixed	**33.71**	27.05	27.17	**27.70**
[HaCohen et al. 2011]	32.12	22.88	27.31	27.34
[Pitié et al. 2007]	23.20	**27.70**	24.63	24.76
[Kotera 2005]	8.72	26.10	23.51	22.49

C. Limitations and possible improvements

Figure 16 shows how our results may improve if we use, instead of SIFT, a method which gives a more populated set of pixel correspondences between I_1 and I_2 (in this example we

have applied to the ground truth image a state-of-the-art optical flow computation algorithm [3]), at the expense of increasing the computational cost.

Figure 17 shows some limitations of our method. For the wide angle shots on the top our result is poor, probably due to the lack of enough SIFT matches caused by the great disparity among the shots; a more sophisticated method than SIFT for finding correspondences, like [9], could improve the results. For the stereo shots on the bottom our results are also lacking, and we think that for this example a global approach like ours isn't enough, because the color differences may be due to color aberrations of the beam-splitter and therefore local instead of global, not an uncommon scenario in 3D cinema [16].

Another limitation of our method is related to highly saturated colors, which tend to fall outside the color gamut of the output format and therefore are usually clipped. The result is that for pixels with these colors the in-camera color processing can no longer be modeled as a linear transformation, as stated by Kim et al. [13]. Since these colors do not fulfill our basic assumption of a linear model for the color pipeline, as stated in Eq. 4, they might be transferred incorrectly, as Figure 18 shows: notice the results for the yellow marker and the red cup.

V. Summary

We have presented a method to remove color fluctuations among images of the same scene, taken with a single camera or several cameras of the same or different models. No information about the cameras is needed. The method works for still images and for video. It is based on the observation that the color correction operations performed in-camera (apart from gamma correction) can be cascaded into a single 3x3 matrix, and color matching among images only requires to transform one matrix into another, it's not necessary to actually estimate the matrices. This is precisely what motivated TV engineers to allow for manual modification of the colorimetric matrices of broadcast cameras, so that their colors could be matched and the transitions among them were smooth.

We have shown applications of our method in a variety of settings, as well as comparisons with the state of the art in the color transfer literature. We have chosen to implement our method using reliable, classical techniques, for which there exist hardware accelerated implementations, but we also show how better numerical methods can further improve the results.

References

[1] G. Adcock. Charting your camera. *Creative COW Magazine*, November 2011.

[2] S. Bianco, A. Bruna, F. Naccari, and R. Schettini. Color space transformations for digital photography exploiting information about the illuminant estimation process. *JOSA A*, 29(3):374–384, 2012.

[3] Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40(1):120–145, 2011.

[4] S. Devaud. *Tourner en vidéo HD avec les reflex Canon: EOS 5D Mark II, EOS 7D, EOS 1D Mark IV*. Eyrolles, 2010.

[5] Z. Farbman and D. Lischinski. Tonal stabilization of video. *ACM Transactions on Graphics (TOG)*, 30(4):89, 2011.

[6] Hany Farid. Blind inverse gamma correction. *Image Processing, IEEE Transactions on*, 10(10):1428–1433, 2001.

[7] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[8] David H. Foster, Kinjiro Amano, Sérgio M. C. Nascimento, and Michael J. Foster. Frequency of metamerism in natural scenes. *J. Opt. Soc. Am. A*, 23(10):2359–2372, Oct 2006.

[9] Y. HaCohen, E. Shechtman, D.B. Goldman, and D. Lischinski. Non-rigid dense correspondence with applications for image enhancement. *ACM Transactions on Graphics (TOG)*, 30(4):70, 2011.

[10] Yoav HaCohen, Eli Shechtman, Dan B Goldman, and Dani Lischinski. Optimizing color consistency in photo collections. *ACM Transactions on Graphics (TOG)*, 32(4):38, 2013.

[11] T.W. Huang and H.T. Chen. Landmark-based sparse color representations for color transfer. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 199–204. IEEE, 2009.

[12] S. Kagarlitsky, Y. Moses, and Y. Hel-Or. Piecewise-consistent color mappings of images acquired under various conditions. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2311–2318. IEEE, 2009.

[13] S. Kim, H. Lin, Z. Lu, S. Susstrunk, S. Lin, and M. Brown. A new in-camera imaging model for color computer vision and its application. *IEEE Trans PAMI*, 2012.

[14] Hiroaki Kotera. A scene-referred color transfer for pleasant imaging on display. In *Proceedings of IEEE ICIP 2005*, pages 5–8, 2005.

[15] D.G. Lowe. Object recognition from local scale-invariant features. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[16] B. Mendiburu. *3D movie making: stereoscopic digital cinema from script to screen*. Focal Press, 2009.

[17] K. Parulski and K. Spaulding. Color image processing for digital cameras. *Digital Color Imaging Handbook*, pages 727–757, 2003.

[18] F. Pitié, A.C. Kokaram, and R. Dahyot. Automated colour grading using colour distribution transfer. *Computer Vision and Image Understanding*, 107(1):123–137, 2007.

[19] C. Poynton. *Digital Video and HD: Algorithms and Interfaces*. Morgan Kaufmann, 2003.

[20] E. Reinhard. Example-based image manipulation. In *6th European Conference on Colour in Graphics, Imaging, and Vision (CGIV 2012)*, Amsterdam, May 2012.

[21] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley. Color transfer between images. *Computer Graphics and Applications, IEEE*, 21(5):34–41, 2001.

[22] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[23] O. Schreer, J-F Macq, O. Niamut, J. Ruiz-Hidalgo, B. Shirley, G. Thallinger, and G. Thomas (editors). *Media Production, Delivery and Interaction for Platform Independent Systems: Format-Agnostic Media*, chapter 2. Wiley, 2014. ISBN:978-1-118-60533-2.

[24] Y.W. Tai, J. Jia, and C.K. Tang. Local color transfer via probabilistic segmentation by expectation-maximization. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 747–754. IEEE, 2005.

[25] A. Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the international conference on Multimedia*, pages 1469–1472. ACM, 2010.

[26] WavelengthMedia. CCU Operations. Technical report, http://www.mediacollege.com/video/production/camera-control/, 2012.

Fig. 5: *Left: Reference image. Middle: Source image. Right: Our result. Images property of Mammoth HD Inc.*

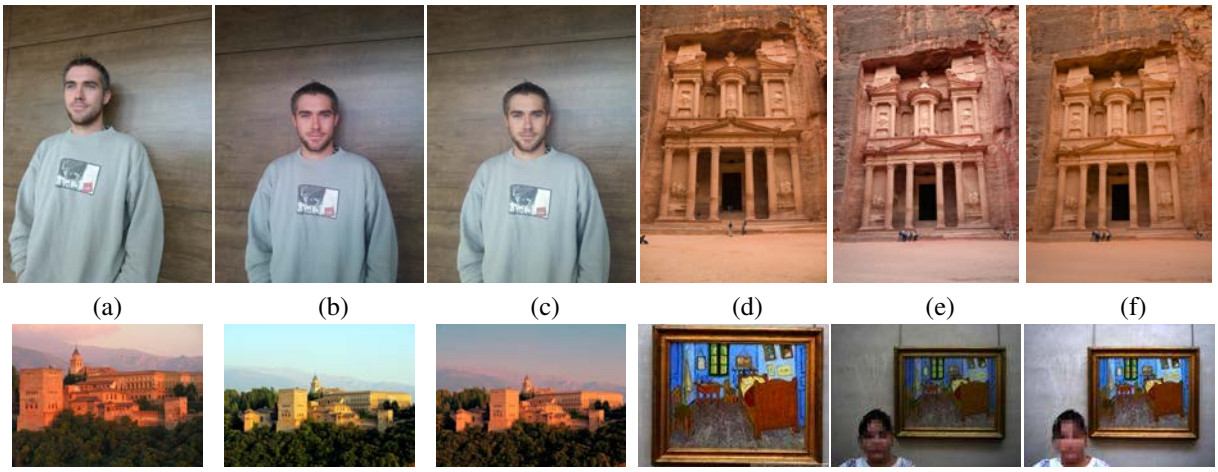


Fig. 6: *(a) Picture taken with a photographic camera. (b) Picture taken with a mobile phone. (c) Our result of matching (b) to (a). (d) Reference. (e) Source. (f) Our result, matching (e) to (d). (g) Reference. (h) Source. (i) Our result, matching (h) to (g). (j) Reference. (k) Source. (l) Our result, matching (k) to (j).*

Fig. 7: *(a) Reference. (b) Source, different color temperature. (c) Our result of matching (b) to (a). (d) Source, different "type of scene" camera setting. (e) Our result of matching (d) to (a).*

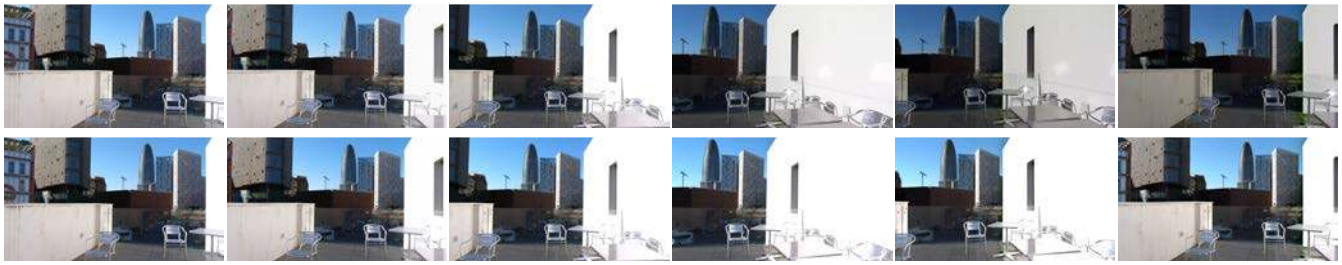


Fig. 8: *Top: Original video, with AE and AWB. Bottom: Our result.*

Fig. 9: *Left: Reference image. Middle: Source image. Right: our result. The original images are stills from a video.*

(a) (b) (c) (d)

Fig. 10: *(a) Reference. (b) Source. (c) Result from Kotera [14]. (d) Our result.*

(a) (b) (c) (d) (e)

Fig. 11: *(a) Reference. (b) Source. (c) Result from Pitié et al. [18]. (d) Result from HaCohen et al. [9]. (e) Our result.*

(a) (b) (c) (d) (e)

Fig. 12: *(a) Reference. (b) Source. (c) Result from Pitié et al. [18]. (d) Result from HaCohen et al. [9]. (e) Our result.*

(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 13: *(a) Reference. (b) Source. (c) Ground truth. (d) Result from Kotera [14] . (e) Result from Pitié et al. [18]. (f) Result from HaCohen et al. [9]. (g) Our result with fixed parameters.(h) our result changing parameters*

(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 14: *(a) Reference. (b) Source. (c) Result of Kim et al. [13], used as ground truth. (d) Result from Kotera [14] . (e) Result from Pitié et al. [18]. (f) Result from HaCohen et al. [9]. (g) Our result with fixed parameters.(h) our result changing parameters.*

Fig. 15: *From left to right: reference, source, ground-truth, Kotera [14], Pitié et al. [18], HaCohen et al. [9], our result with fixed parameters, our result changing parameters.*

Fig. 16: *From left to right: reference, source, our result changing parameters (31.04 dB), our result with fixed parameters (30.85 dB), our result using optical flow to compute the matches (31.86 dB).*

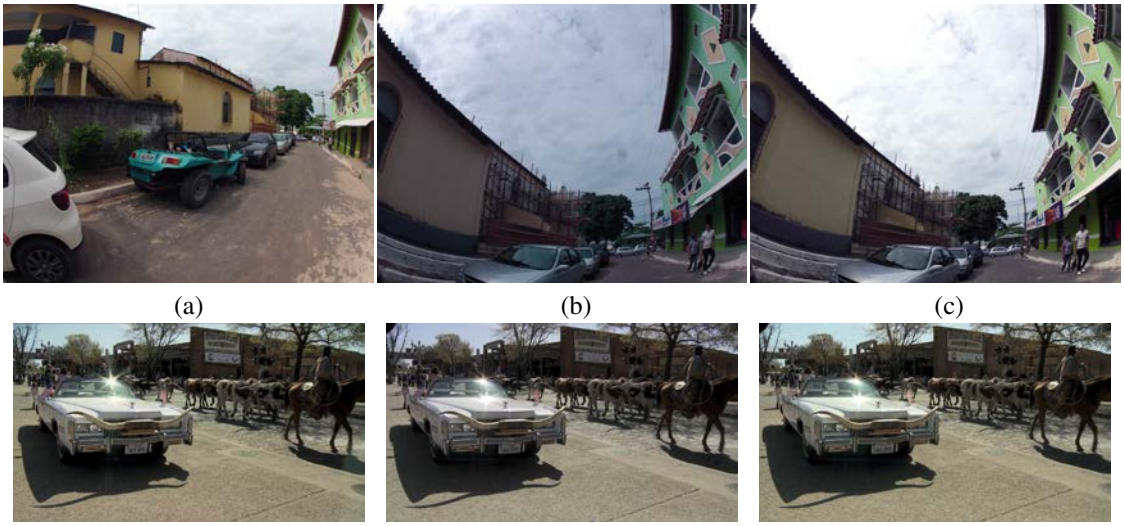


Fig. 17: *(a) Reference. (b) Source. (c) Our result, matching (b) to (a). (d) Reference. (e) Source. (f) Our result, matching (e) to (d).*

Fig. 18: *Effect of saturated pixels in our approach. From left to right: source, reference, our result.*