Color matching images with unknown non-linear encodings

Raquel Gil Rodríguez, Javier Vazquez-Corral, and Marcelo Bertalmío

***Abstract*—We present a color matching method that deals with different non-linear encodings. In particular, given two different views of the same scene taken by two cameras with unknown settings and internal parameters, and encoded with unknown non-linear curves, our method is able to correct the colors of one of the images making it look as if it was captured under the other camera's settings. Our method is based on treating the in-camera color processing pipeline as a concatenation of a matrix multiplication on the linear image followed by a non-linearity. This allows us to model a color stabilization transformation among the two shots by estimating a single matrix -that will contain information from both of the original images- and an extra parameter that complies with the non-linearity. The method is fast and the results have no spurious colors. It outperforms the state-of-the-art both visually and according to several metrics, and can handle HDR encodings and very challenging real-life examples.**

***Index Terms*—Color stabilization, color matching, logarithmic encoded images, gamma-corrected images, HDR encoding, PQ, HLG.**

I. Introduction

Color matching techniques aim to map the colors of one image, defined as source, to those of a second image, defined as reference. A particular case is color stabilization, where the two pictures are taken from the same scene and differ in terms of color. These differences in color may be caused either by the use of different camera models, which follow different internal procedures tailored by the manufacturer or by the use of the same camera but under different settings like white balance, exposure time, etc.

Digital cameras perform a number of image processing steps, including demosaicing, white balance, color correction (from RGB camera sensor to device independent color space), and encoding standard. Bianco *et al.* [1] proposed a generic model of the color processing pipeline of digital cameras

$$I_{out} = (A \cdot I_{lin})^{1/\gamma}, \tag{1}$$

where I_{lin} is the linear image read by the camera sensor after demosaicing, I_{out} is the output image, A is a 3×3 matrix which carries color information and white balance and the value γ defines a power law function (usually known as gamma correction). This is a simplified version of the pipeline, since other processing techniques, like denoising, contrast enhancement, etc. are also applied. Nonetheless, this approximation is quite accurate for those pixels not laying close to the boundary of the color gamut.

Although gamma correction has been the most used encoding technique, it fails when working with high dynamic range (HDR) imaging, mostly due to quantization issues. Current professional cinema cameras are able to capture a wide range of light intensities, and therefore, a compression of this range is needed for storage, while preserving all the details and appearance. For this reason, cinema cameras replace gamma correction with a logarithmic function whose general form (common to the most popular log-encoding approaches [2], [3]) can be expressed as:

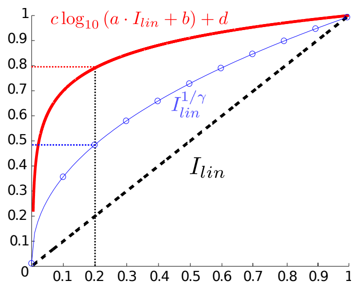

Fig. 1. Linear response (dashed) versus gamma-corrected (circle) and logarithmic response (continuous). The gamma correction was defined as $\gamma = 2.2$, and the logarithmic curve is an instance of an ARRI Log C curve.

$$I_{out} = c\log_{10}(a \cdot A \cdot I_{lin} + b) + d, \tag{2}$$

where I_{out} and I_{lin} are defined as above, and the parameters $a, b, c,$ and d are constant real values (varying for different camera manufacturers and camera settings). Figure 1 shows the plot of linear (dashed), gamma-corrected (circle) and logarithmic (continuous) responses to linear values. Notice that gamma correction and logarithmic curve assign respectively 50% and 80% of the output range to the first 20% of the linear intensity values.

More recently, other non-linear encoding curves devised for HDR content have appeared. The most well-known of these curves are Perceptual Quantizer (PQ) [4] and Hybrid Log-Gamma (HLG) [5]. They were designed to reduce quantization errors in the storage and coding of HDR scenes. Both PQ and HLG are mathematically well-defined. For more details on the curves definition, we refer the reader to [4] and [5].

In the industry, solutions for bringing consistency across image shots usually involve very skilled manual work, done by colorists during color grading in movie post-production and by technicians using camera control units (CCU) [6] in live TV broadcasts. They may also require a proper characterization of the cameras used and their settings like with the ACES framework [7], or the presence of color-charts in the shots.

In image processing and computer vision research, it is an open problem to color match a pair of pictures. We can differentiate between two different cases: i) the image pair does not necessarily share any content (color transfer), or ii) the image pair is acquired from the same scene (color stabilization). The latter can be understood as a constrained color transfer problem.

The aim of color transfer methods is to transfer the colors presented in the reference image to the source image. A seminal work in color transfer was proposed by Reinhard *et al.* [8], where the pair of RGB images are first converted to a decorrelated color space and then the mean and variance from the reference are transferred to the source. Pitié *et al.* [9] [10] defined the images as probability density functions, and then match them through an iterative non linear process. It is worth mentioning color transfer as an application of optimal transport, which minimizes the cost of transferring probability density distributions of the source image into the reference one, as in Rabin *et al.* [11] and Ferradans *et al.* [12] works. Pouli and Reinhard [13] performed histogram matching along different scales given images of different dynamic ranges. The method presented by Kotera [14] proposed to compute the principal components of the 3D color distributions, in order to match the principal axes of the source to the reference image by a matrix multiplication (rotation and scaling). Xiao and Ma [15] worked with color statistics, and in [16] they proposed a gradient preserving color transfer technique, and an evaluation metric for color transfer methods. Nguyen *et al.* [17] presented a color transfer method that first applies color constancy to the input images, then it performs luminance matching, and finally the color gamuts are aligned by a linear transformation. Hwang *et al.* [18] suggested to use moving least squares for color transfer, by also incorporating a probabilistic measure to ensure robustness against noise and outliers. Gong *et al.* [19], [20] proposed a color transfer method based on a projective transformation and a mean intensity mapping. All the above mentioned methods are global, although some local approaches also exist. The work of Tai *et al.* [21] segmented images into regions, and then it used Gaussian Mixture Models (GMM) to represent color distributions before the matching is performed. Xiang *et al.* [22] also followed a GMM representation of color areas, before matching them.

Color stabilization tackles the situation where some regions or objects appear in both the reference and the source images. HaCohen *et al.* [23] presented a method to compute dense correspondences between the images, combined with a global color mapping model. Vazquez-Corral and Bertalmío [24] proposed a color stabilization algorithm that consists of estimating a power law (γ value) for each of the images, and a single 3×3 matrix, to color match the source image to the reference. It is built on the assumption that in digital cameras the color encoding can be expressed as a matrix multiplication followed by a power law (gamma correction). Frigo *et al.* [25] presented a method to color stabilize video sequences, based on the estimation of a non-linearity and channel-based scaling. To the best of the authors knowledge, there are only two color stabilization works for logarithmic images. One is the method of Vazquez-Corral and Bertalmío [26], that relies on finding a sufficiently large number of achromatic matches among source and reference. Let us note that the need of detecting achromatic matches may be a challenging limitation in some cases. The other method [27] is an earlier version of the current work, with a different algorithm that consistently underperforms the one we will introduce here as it is shown in the Results section.

Color consistency refers to the situation when a set of images from the same scene need to be color matched. HaCohen *et al.* [28] extended their previous approach in color stabilization problem [23], to the case of more than two images. In a recent work, Park *et al.* [29] proposed a model in which the parameters to be estimated are a gamma correction and a white balance constant. Xia *et al.* [30] presented a method to achieve color consistency in image stitching. On the overlapping regions among the shots, it computes parametric curves for each channel under color, gradient and contrast constraints. A review of the performance of color transfer methods is presented in Xu and Milligan [31] for image stitching.

Our main contribution in this work is a method able to color match pairs of images that were encoded with different non-linearities (gamma, logC, HLG, PQ). This work is an extension of the one presented in [27]. In this paper, we improve over the previous approach in different ways. First, we allow the matrix in our model to be a projective 4×4 matrix. This brings more flexibility to the model which enables allowing it to better deal with saturated pixels that have gone through different non-linear in-camera operations such as tone-mapping or gamut-mapping. Also, we introduce a new term in the minimization, which minimizes errors in the perceptual Lab color space. Furthermore, we show how our method can be used when images are encoded with HLG and PQ curves, the two current standards for HDR encodings. Lastly, in this paper we present a new dataset and framework for the problem, and perform larger comparisons both in terms of the methods and the metrics considered. Our results outperform the rest of the algorithms both quantitative and qualitatively.

II. Methodology

In this paper, we present a color stabilization method that takes as input an image pair encoded with unknown non-linear curves. For simplicity we explain the method for the case of gamma correction and logarithmic encodings, and at the end of this section, we show how to handle as well PQ and HLG encodings. The main steps of our method can be outlined as follows:

1) If source or reference are log-encoded, we transform them into gamma-corrected.
2) We color-stabilize the images by estimating a 4×4 matrix and two power law values.
3) Finally, we undo the correction made in the first step if necessary (in case the original reference image is log-encoded).

We refer the reader to the flowchart of the proposed model in Figure 2. We have made the code for our implementation available at http://ip4ec.upf.edu/ColorMatching.

A special case is when dealing with HLG and PQ encoded images. In this case, we proceed as if the images were log-encoded. Please note that this is an approximation, as these

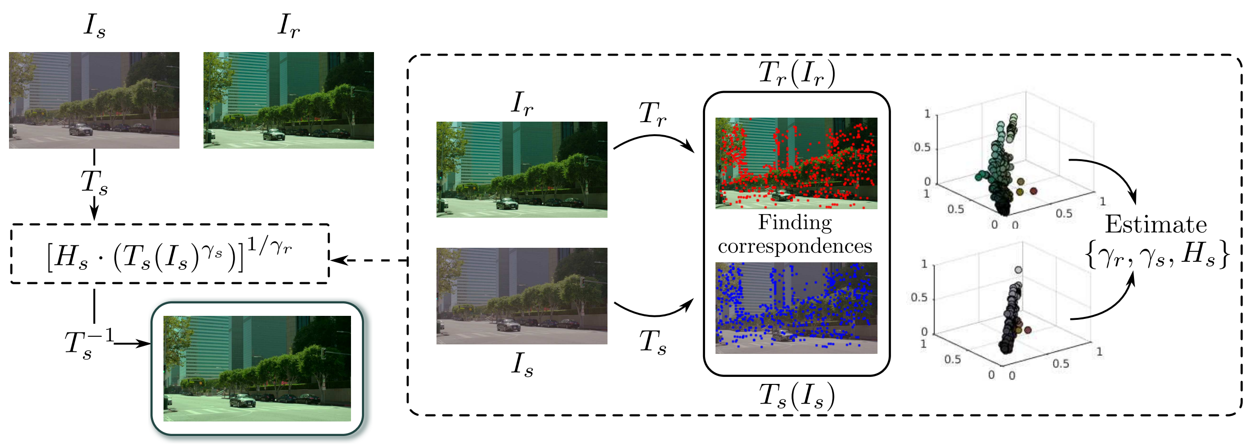

Fig. 2. Flowchart of the proposed color stabilization method. Given two non-linear encoded images, reference (I_r) and source (I_s), we apply the transformations T_r and T_s to the image pair. These transformations are defined as the power 10 function $10^{\times}$, in case of a given log-encoded image; and as the identity $\mathbb{1}$, in case of gamma corrected input. Then, we compute a set of correspondences pts_r and pts_s, using standard feature descriptor (e.g. in this article SIFT [32]). From this set of corresponding pixel locations, we estimate the parameters $\{\gamma_r, \gamma_s, H_s\}$ in the pixel values correspondences. The computed values are applied to the $T_s(I_s)$ image. Finally, T_s^{-1} function is applied to the color matched image.

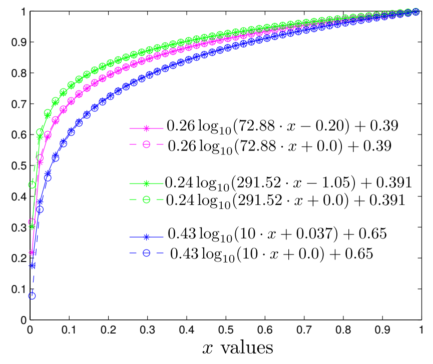

Fig. 3. Graph of 3 logarithmic encoded curves Log C ARRI of EI 320 (green), Log C ARRI of EI 1280 (magenta) [2], and S-Log [3] (blue) plotted in continuous lines. In addition, the same logarithmic curves by setting $b = 0$ in their definitions (dashed lines). Note that the distance between the dashed and continuous lines from the same color is negligible.

curves do not follow the definition in Equation (2). This said, this approximation works extremely well, as it is shown in section IV.

A. Conversion of log-encoding to gamma correction

Let us consider a log-encoded image as in Equation (2). If we apply a power 10 function to it (denoted as $T(\cdot)$), we obtain the following expression

$$\begin{aligned} T(I_{out}) = 10^{I_{out}} &= 10^{\log_{10}((a \cdot A \cdot I_{lin} + b)^c)} \cdot 10^d \\ &= (a \cdot A \cdot I_{lin} + b)^c \cdot 10^d. \end{aligned} \tag{3}$$

In logarithmic encoding curves, the value of parameter b is usually small. As Figure 3 shows, for the three different logarithmic curves (continuous lines), their equivalent curves fixing $b = 0$ (dashed lines) lie on top. Therefore, we can simplify Equation (3) by neglecting b,

$$T(I_{out}) = (a \cdot A \cdot I_{lin})^c \cdot 10^d = (K \cdot I_{lin})^c, \tag{4}$$

where $K = a \cdot A \cdot 10^{d/c}$ is a matrix with the same size as A. Notice how Equation 4 has the same form as Equation (1). Therefore, by applying the power 10 function to a log-encoded image, it can now be treated as a regular gamma-corrected picture.

B. Color stabilization

If I_r or I_s (or both) are log-encoded, we transform them into gamma-corrected images I'_r and I'_s, as explained in the previous Section. Then we compute a set of correspondences pts_r and pts_s; we use SIFT [32] for this purpose, although it can be replaced by any other method. It is important to note that we compute the correspondences between $I'_r \leftrightarrow I'_s$, and $I'_s \leftrightarrow I'_r$, and select those that appear in both directions. This allows us to discard some potentially incorrect correspondences. Let us now define the pixel values in the corresponding locations of I'_r and I'_s as

$$\{(R'_r, G'_r, B'_r)^t\}_i, \text{ and } \{(R'_s, G'_s, B'_s)^t\}_i, \tag{5}$$

where $i = 1, \ldots, N$ denotes the number of correspondences. We follow the idea from the color stabilization model proposed in [24],

$$H_s \cdot {I'}_s^{\gamma_s} \sim {I'}_r^{\gamma_r}, \tag{6}$$

where H_s was a 3×3 matrix that transforms colors from the source to match the ones of the reference, and γ_r, γ_s are inverse gamma correction values. In this work, we extend the matrix H_s as a projective transformation with size 4×4 (inspired by color homography [19], [20]). By doing this, the model can deal not only with pixels in the core of the color gamut, but also with those values that appear on the border, which are the most affected by gamut mapping and tone mapping. Then, from the set of correspondences, we can

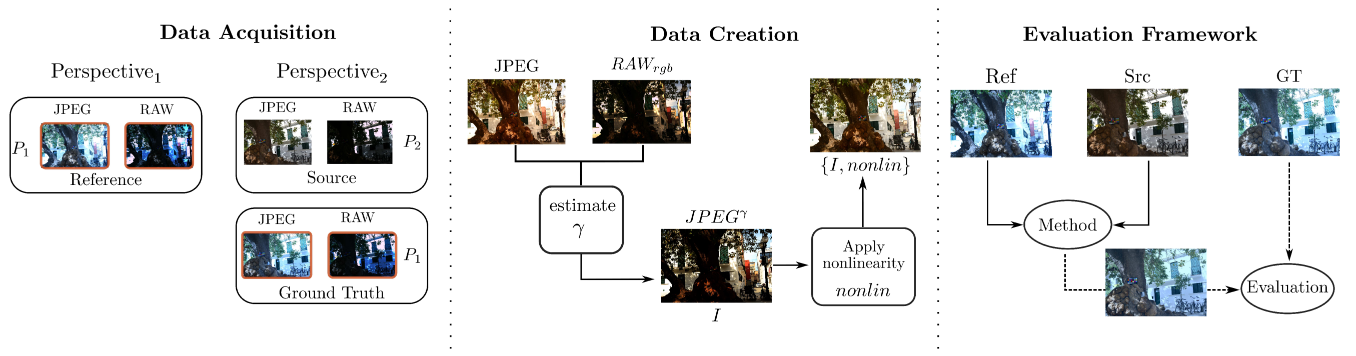

Fig. 4. Evaluation framework. On the left, data acquisition is described. Pictures are taken from the same scene, and from two different points of view Perspective$_1$ and Perspective$_2$. From the first one, the reference image is taken, and from the second, the source and the ground truth. Images are stored in RAW and JPEG format, and we chose different camera settings P_1, for reference and ground truth, and P_2 parameters for the source. On the middle, data is created by linearizing the JPEG image, i.e undoing the gamma correction I. Once linearized, a random non-linearity is applied, and the new image and the non-linearity are stored $\{I, nonlin\}$. Finally, the reference Ref and source Src become the input images for the color matching methods, and the corresponding output is evaluated against the GT.

build a system of equations considering matrix size 4×4 and homogeneous coordinates,

$$H_s \cdot \begin{bmatrix} R_s^{'} \\ G_s^{'} \\ B_s^{'} \\ 1 \end{bmatrix}^{\gamma_s} - \begin{bmatrix} R_r^{'} \\ G_r^{'} \\ B_r^{'} \\ 1 \end{bmatrix}^{\gamma_r} = 0, \tag{7}$$

$$H_r \cdot \begin{bmatrix} R_r^{'} \\ G_r^{'} \\ B_r^{'} \\ 1 \end{bmatrix}^{\gamma_r} - \begin{bmatrix} R_s^{'} \\ G_s^{'} \\ B_s^{'} \\ 1 \end{bmatrix}^{\gamma_s} = 0,$$

where $\{\gamma_s, \gamma_r, H_s, H_r\}$ are the unknowns. We perform a single optimization process, where the only constraint is $H_r \cdot H_s \sim \mathbb{1}$ (the identity). This constraint assures that the transformation H_s has an inverse, and that is represented by H_r (which corresponds to the matrix that would transfer the colors of the source into the reference). The objective function considers the 3×1 non-homogeneous coordinates. This function considers the differences on (R,G,B) points, plus the differences on Lab color space. In this way, we bring the corresponding point clouds (color matched and reference) as close as possible, both in terms of the display RGB color space, and also regarding perceptual color differences in the Lab space.

$$E(\mathcal{V}) = E_{RGB}(\mathcal{V}) + E_{Lab}(\mathcal{V}), \tag{8}$$

where $\mathcal{V} = \{\gamma_r, \gamma_s, H_r, H_s\}$ is the set of unknowns, and E_{RGB} and E_{Lab} are the errors in the RGB and Lab color spaces, respectively. These terms are defined as

$$E_{RGB}(\mathcal{V}) = \sum_{\substack{RGB_s \\ RGB_r}} \left\| RGB_r - g_{\mathcal{V}}^1(RGB_s) \right\|_2 + \left\| RGB_s - g_{\mathcal{V}}^2(RGB_r) \right\|_2, \tag{9}$$

$$E_{Lab}(\mathcal{V}) = \sum_{\substack{RGB_s \\ RGB_r}} \left\| Lab(RGB_r) - Lab\left(g_{\mathcal{V}}^1(RGB_s)\right) \right\|_2 + \left\| Lab(RGB_s) - Lab\left(g_{\mathcal{V}}^2(RGB_r)\right) \right\|_2, \tag{10}$$

where RGB_s, RGB_r are the (R, G, B) values of corresponding pointspts_s and pts_r, $Lab(\cdot)$ corresponds to the color transformation from RGB to Lab, and finally the functions $g_{\mathcal{V}}^1(\cdot)$ and $g_{\mathcal{V}}^2(\cdot)$ are defined as

$$g_{\mathcal{V}}^1(RGB_s) = \left(H_s \cdot RGB_s^{\gamma_s}\right)^{1/\gamma_r} \quad \text{and} \tag{11}$$

$$g_{\mathcal{V}}^2(RGB_r) = \left(H_r \cdot RGB_r^{\gamma_r}\right)^{1/\gamma_s}. \tag{12}$$

Finally, the matrices and non-linearities are applied to the entire images as in Equation (6), and we obtain the color matched image:

$$I''_s = \left(H_s \cdot I'^{\gamma_s}_s\right)^{1/\gamma_r}. \tag{13}$$

C. Conversion back to log-encoded images

If I_s was log-encoded, we apply a $\log_{10}$ function, denoted as $T^{-1}(\cdot)$, to the result of the previous step so as to undo the power 10 transform we applied at the beginning.

III. Experiments with Gamma and Logarithmic Encoding Non-Linearities

This section is divided into 3 different parts. First, we describe how we have created an image dataset for evaluation. Second, we compare our approach with seven popular color matching methods. Third, we evaluate the performance of the rest of methods when the proposed power 10 is applied in the case of log-encoded images.

A. Dataset

Our data is composed of different scenes, where each of them contains a reference image Ref, a source image Src and a ground truth image GT. In order to acquire our data, we work in camera manual mode to have full control over exposure time, white balance, ISO value, and aperture. We stored RAW and JPEG formats for each image. In that way, we have the linear information read by the camera sensor (RAW), as well as the final compressed image (JPEG). Images were taken

TABLE I
RESULTS FROM THE COMPARISON AMONG 35 IMAGE PAIRS FOR: 1) TWO γ-ENCODED IMAGES, 2) TWO LOG-ENCODED IMAGES, 3) REFERENCE LOG-ENCODED AND SOURCE γ-CORRECTED, AND 4) REFERENCE γ-CORRECTED AND SOURCE LOG-ENCODED.

		ΔE^*_{00}		PSNR L		CPSNR		CID		RMSE	
		μ	$\hat{\mu}$	μ	$\hat{\mu}$	μ	$\hat{\mu}$	μ	$\hat{\mu}$	μ	$\hat{\mu}$
Ref γ-Src γ	**Kotera**	11.111	7.686	21.122	23.877	19.786	21.040	0.458	0.394	0.145	0.089
	Pitie	3.567	3.394	26.162	25.946	25.696	25.769	0.174	0.157	0.055	0.051
	Reinhard	4.777	4.652	25.525	25.162	23.904	23.571	0.205	0.191	0.068	0.066
	Xiao	4.377	4.232	25.940	26.077	25.183	25.270	0.196	0.160	0.059	0.055
	Ferradans	5.522	5.308	23.715	23.874	23.028	22.560	0.260	0.237	0.078	0.074
	Park	3.428	**3.020**	27.604	**27.381**	26.595	26.384	0.157	0.134	0.051	0.048
	Gil	3.726	3.554	27.420	27.228	26.116	24.965	0.164	0.149	0.054	0.056
	Ours	**3.263**	3.092	**27.650**	27.271	**26.907**	**26.576**	**0.145**	**0.125**	**0.049**	**0.047**
Ref log-Src log	**Kotera**	14.234	8.381	18.586	21.081	17.615	19.676	0.551	0.481	0.179	0.104
	Pitie	3.978	4.044	25.797	25.369	25.119	25.099	0.207	0.201	0.059	0.056
	Reinhard	7.878	7.916	22.656	22.512	19.899	19.369	0.364	0.368	0.107	0.108
	Xiao	5.632	5.599	24.330	23.910	23.199	23.190	0.272	0.264	0.072	0.069
	Ferradans	8.587	7.047	19.351	20.831	18.925	20.250	0.395	0.325	0.128	0.097
	Park	6.768	4.548	26.217	26.162	23.961	24.196	0.296	0.210	0.083	0.062
	Gil	4.057	3.644	27.027	26.689	25.665	25.379	0.193	0.155	0.057	0.054
	Ours	**3.400**	**3.022**	**27.446**	**27.158**	**26.587**	**26.479**	**0.161**	**0.135**	**0.050**	**0.047**
Ref log-Src γ	**Kotera**	15.704	12.405	17.017	18.864	15.970	16.625	0.631	0.586	0.199	0.148
	Pitie	3.909	3.830	25.796	25.498	25.225	25.020	0.200	0.201	0.059	0.056
	Reinhard	7.928	7.516	21.260	21.056	18.883	18.687	0.393	0.392	0.117	0.116
	Xiao	7.926	7.554	21.446	20.539	20.438	20.059	0.403	0.416	0.100	0.099
	Ferradans	8.578	7.954	19.654	19.163	19.172	18.518	0.381	0.369	0.122	0.119
	Park	5.895	5.242	24.038	23.352	22.972	22.294	0.305	0.290	0.078	0.077
	Gil	4.066	3.667	27.102	26.847	25.741	25.627	0.188	0.178	0.058	0.052
	Ours	**3.377**	**3.140**	**27.571**	**27.606**	**26.712**	**26.632**	**0.157**	**0.129**	**0.050**	**0.047**
Ref γ-Src log	**Kotera**	12.658	9.202	18.629	20.748	17.893	20.089	0.538	0.430	0.162	0.099
	Pitie	3.752	3.903	25.957	25.538	25.378	25.217	0.184	0.173	0.057	0.055
	Reinhard	6.438	6.246	22.861	22.642	21.776	21.666	0.291	0.291	0.084	0.083
	Xiao	6.794	5.734	23.023	22.770	22.097	22.215	0.322	0.314	0.081	0.077
	Ferradans	6.317	6.165	22.222	21.826	21.577	21.357	0.318	0.298	0.089	0.086
	Park	12.808	9.620	20.746	22.593	18.779	19.351	0.510	0.454	0.147	0.108
	Gil	3.863	3.476	27.197	26.672	25.889	25.341	0.173	0.162	0.054	0.054
	Ours	**3.444**	**3.313**	**27.395**	**26.922**	**26.563**	**25.684**	**0.152**	**0.144**	**0.050**	**0.052**

using two camera models, Nikon D3100 (12-bits) and Canon EOS80D (14-bits). Let us explain the steps we follow from acquisition to the final triplet Ref, Src, and GT images for each scene:

- Set the parameters of the camera P_1 (exposure time, white balance, ISO value, and aperture), the camera position Perspective$_1$, and acquire the Reference (Ref) set RAW and JPEG.
- Use the same camera parameters (P_1) as in the Ref set, and change the camera position to Perspective$_2$, then we acquire the Ground-Truth (GT) set RAW and JPEG.
- From the last camera position Perspective$_2$, vary the camera settings to a different configuration P_2 to acquire the Source (Src) set.
- For each pair (RAW, JPEG) obtain $\{I, nonlin\}$:
  1) Preprocess the RAW input to obtain an RGB linear image using *DCRAW* [33] open source code, we refer to this image as RAW_{rgb}.
  2) Estimate the γ correction curve, between the preprocessed RAW_{rgb} and the JPEG using [24].
  3) Undo γ from the JPEG image in order to obtain a linear image called I with the camera color processing still in.
  4) Apply a random generated non-linearity to I. In case of gamma correction, we set values to be selected in the range $[1.7, 2.7]$, and for logarithmic curves, we select the definitions from Log C ARRI (a total of 11 curves) [2], and S-Log from Sony [3]. We name the applied non-linearity $nonlin$.
  5) In case of GT, the same $nonlin$ as the one selected for the reference is applied.

B. *Results and comparisons*

We evaluate our approach against seven state-of-the-art methods for color transfer, stabilization and consistency: Reinhard *et al.* [8] (Reinhard), Kotera [14] (Kotera), Xiao and Ma [15] (Xiao), Pitié *et al.* [10] (Pitie), Ferradans *et al.* [12] (Ferradans), Park *et al.* [29] (Park), and Gil Rodríguez *et al.* [27] (Gil). We want to emphasize that for Pitié *et al.* [10], we focus only on the global part of the method. We studied all possible combinations of applied non-linearities to the reference and source image:

1) two gamma-corrected images,
2) two log-encoded images,
3) one gamma-corrected as reference and one log-encoded as source,
4) one log-encoded as reference and one gamma-corrected as source.

In the quantitative evaluation we select the following color metrics:

- *PSNR* of luminance channel (PSNR L),

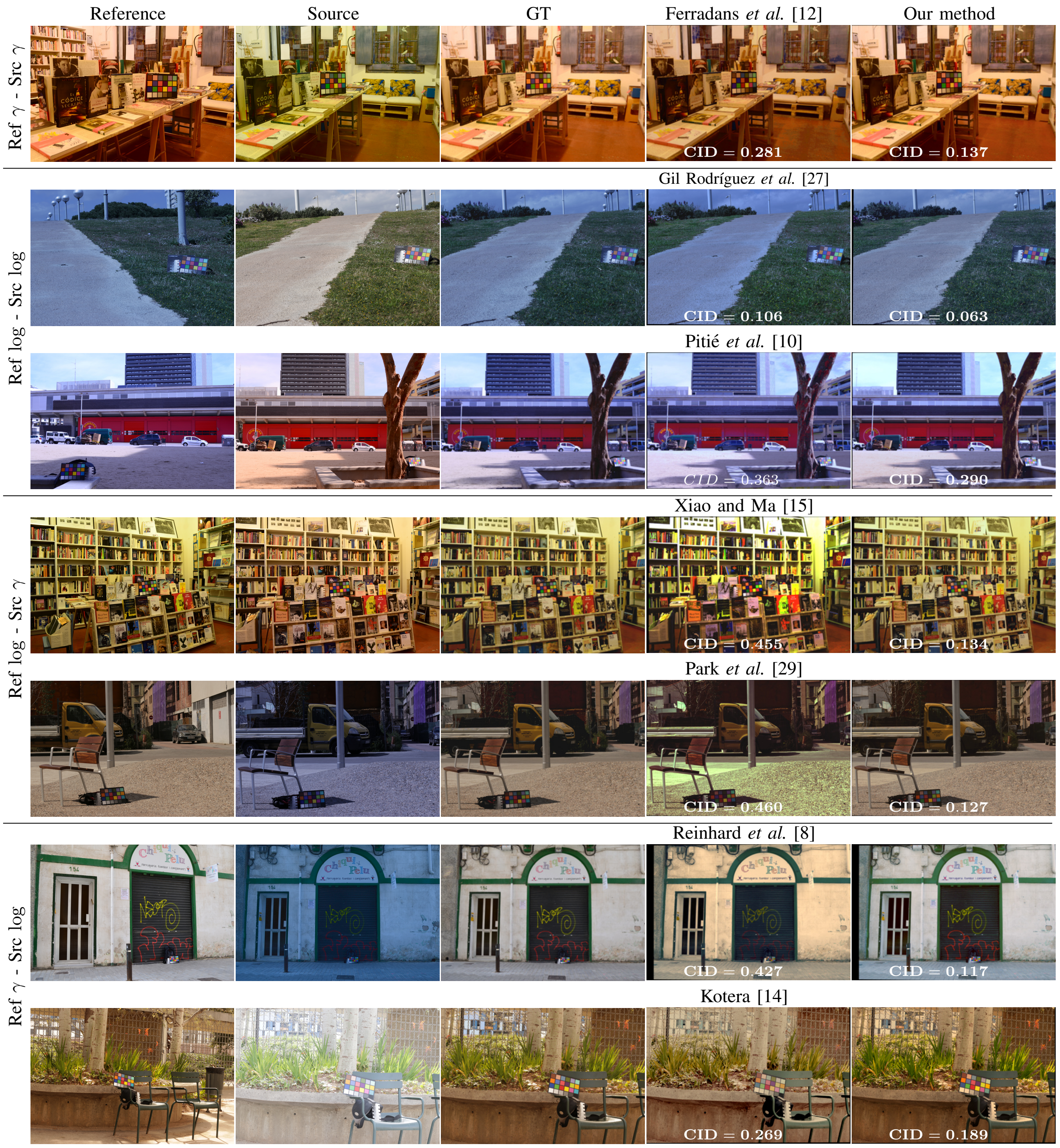

Fig. 5. Results of all the methods for the four comparisons. Each block represents: 1) gamma-corrected image pair, 2) log-encoded input images, 3) log-encoded reference and gamma-corrected source and 4) gamma-corrected reference and log-encoded source. The first column presents the reference, the second shows the source, the third the GT, the fourth the methods result, and the last our result. We present for 1) Ferradans *et al.* [12], 2) Gil Rodríguez *et al.* [27] and Pitié *et al.* [10] methods and 3) Xiao and Ma [15] and Park *et al.* [29] methods, and 4) Reinhard *et al.* [8] and Kotera [14] methods.

- color *PSNR* defined as *CPSNR* is the mean of the PSNR among the three color channels,
- *root mean squared error* (RMSE),
- ΔE^*_{00} [34] accounts for 'perceptually uniform' differences in the CIELAB color space,
- *CID* [35] is the color extension of SSIM [36], and it is therefore supposed to capture errors more perceptually than PSNR.

For each metric we show the mean (μ) and the median ($\hat{\mu}$). Notice that in order to compare the color-stabilized and the GT in case of log-encoded images, we first undo the ground-truth non-linearity (since it is known) from the result and GT, and then we apply a gamma-correction of value $1/2.2$. We use the data computed as described in Section III-A, which consists of 35 image pairs. In all the Tables for the quantitative comparisons we show in green the best results, and then in blue and orange, the second and third, respectively. In Figures 5 and 6, the log-encoded images are shown in sRGB for display purposes (as before, we first discount the ground-truth non-linearity, and then we apply the sRGB gamma).

1) Gamma-corrected inputs: The first block in Table I presents the results of pairs encoded using gamma correction. In most of the metrics our method outperforms the rest of algorithms, except for median values of ΔE^*_{00} and PSNR L, in which Park *et al.* [29] obtains better results.

In the first row of Figure 5, we show the results for one scene. The first column shows the reference, the second the source, and the third the GT. In this example, we compare our algorithm (last column) against the method of Ferradans *et al.* [12] (fourth column). We can see that [12] loses color saturation in general, and introduces gray colors in the output of the floor.

2) Log-encoded inputs: In the second block of Table I, it is shown that our method outperforms the rest of algorithms in all the metrics.

From Figure 5 (second and third rows), Gil Rodríguez *et al.* [27] is not able to darken the green color of the grass, which it is closer to the vivid look in the source image. In the second scene, the output from Pitié *et al.* [10] cannot recover the red color of the garage in the background, and it presents a purplish color in one of the doors, and it makes appear some clouds on the sky.

3) Log-encoded reference and gamma-corrected source: In the third block of Table I, our proposed method outperforms the rest of the algorithms in all the metrics.

Figure 5 shows the results from Xiao and Ma [15] (fourth row), and Park *et al.* [29] (fifth row), for this case. In the first scene, notice that [15] enhances yellow and red colors, and it saturates the upper right corner of the wall. The method in [29] shows a color shift in the floor, and intensifies the purple color on the right side.

4) Gamma-corrected reference and log-encoded source: The last block in Table I presents the results where the reference is a gamma-corrected image, and the source is log-encoded. For all the metrics, our method outperforms the rest of algorithms.

Figure 5 shows the results from Reinhard *et al.* [8], and Kotera [14] (last two rows), for this case. The result from [8] shows a yellowish cast on the wall. The method of [14] presents washed out colors, e.g. the chair and the wall behind it.

C. Results with power 10

In this subsection we analyse the performance of the rest of methods when also applying a power 10 function to them. More in detail, we first apply the power 10 function to the log-encoded inputs, we then apply the selected method to the new images, and finally we undo the power 10 if necessary. From now on, we refer to this process as $method_{10}$. The results for all the comparisons and methods are presented in Table II. Our results and the results of Gil Rodríguez *et al.* from the previous Table are also included for comparative purposes.

1) Log-encoded inputs: Results show a considerable improvement between the original methods and after applying power 10, see first block. The only exceptions are the algorithms of Pitié *et al.* [10] and Ferradans *et al.* [12], which have a similar performance with and without power 10.

2) Log-encoded reference and gamma-corrected source: Results show a considerable difference between the original method and after applying the power 10 preprocessing. Notice the boosting of Park *et al.*, which improves significantly versus its original version. It is ranked second after our approach in most of the metrics, and in median PSNR L and CPSNR it gets the best results, see Table II (second block). In Figure 6, the Park_{10} method presents no color shift on the floor, although it cannot completely recover the yellow color of the truck.

3) Gamma-corrected reference and log-encoded source: In this case, although Park *et al.* improves their previous results, it is not as noticeable as in the previous comparison. The method of Pitie_{10} outperforms Park_{10} as opposed to the previous case; Pitie_{10} in this context shows a more consistent performance in both cases.

The results of our experiments show that the proposed framework (applying a power 10 function to log-encoded images) boosts the performance of the majority of the methods we compare with. This is true both in terms of quantitative metrics and image quality. The exceptions are the algorithms of Pitié *et al.* [10] and Ferradans *et al.* [12]. These two methods define the input images as probability density functions in order to match them. This fact allows these methods to better adapt for modifications present in the range of the input images.

IV. Experiments with HDR encodings

In this section, we color match a pair of images encoded using different transfer functions: PQ, HLG and Log C ARRI curves.

A. Dataset

The dataset we use for experiments is the one provided by ARRI. This data contains HDR videos. The linear RAW data is obtain by using *ARRIRAW Converter* [39]. We select three different scenes, and for each scene we set a reference image encoded with one of the 3 different options {PQ, HLG, Log C} (a random Log C ARRI curve). Then, we build the data by selecting all the possible combination pairs for each image reference (total of 9 pairs). We add an extra pair comparing two different Log C ARRI curves. Therefore, we have a total of 10 image pairs.

B. Results and comparisons

We compare our method, described in Section II, with the algorithms presented in the previous experiments in subsection III-B: Reinhard *et al.* [8] (Reinhard), Kotera [14] (Kotera),

TABLE II
RESULTS FROM THE COMPARISON AMONG 35 IMAGE PAIRS FOR: 1)) TWO LOG-ENCODED IMAGES, 2) REFERENCE LOG-ENCODED AND SOURCE γ-CORRECTED, AND 3) REFERENCE γ-CORRECTED AND SOURCE LOG-ENCODED. IN THIS CASE, WE APPLIED POWER 10 TO THE INPUTS (IF NECESSARY) FOR THE REST OF ALGORITHMS, EXCEPT GIL.

		ΔE^*_{00}		PSNR L		CPSNR		CID		RMSE	
		μ	$\hat{\mu}$	μ	$\hat{\mu}$	μ	$\hat{\mu}$	μ	$\hat{\mu}$	μ	$\hat{\mu}$
Ref log-Src log	**Kotera$_{10}$**	11.919	8.285	20.771	22.503	19.286	20.719	0.487	0.403	0.148	0.092
	Pitie$_{10}$	3.790	4.074	25.961	25.871	25.377	25.332	0.197	0.188	0.058	0.054
	Reinhard$_{10}$	5.238	5.216	24.824	23.975	23.190	23.178	0.231	0.211	0.072	0.069
	Xiao$_{10}$	4.758	4.507	25.623	25.002	24.646	24.542	0.219	0.216	0.063	0.059
	Ferradans$_{10}$	9.885	7.208	18.695	19.915	18.318	19.464	0.419	0.316	0.145	0.106
	Park$_{10}$	4.137	3.843	26.858	26.733	25.626	24.901	0.204	0.170	0.057	0.057
	Gil (as in previous Table)	4.057	3.644	27.027	26.689	25.665	25.379	0.193	0.155	0.057	0.054
	Ours (as in previous Table)	**3.400**	**3.022**	**27.446**	**27.158**	**26.587**	**26.479**	**0.161**	**0.135**	**0.050**	**0.047**
Ref log-Src γ	**Kotera$_{10}$**	11.775	8.609	20.918	22.022	19.308	20.385	0.485	0.407	0.147	0.096
	Pitie$_{10}$	3.697	3.681	25.991	25.487	25.487	25.258	0.189	0.184	0.057	0.055
	Reinhard$_{10}$	5.379	5.082	25.034	25.086	23.259	23.384	0.235	0.216	0.072	0.068
	Xiao$_{10}$	4.905	4.565	25.556	25.031	24.599	24.649	0.227	0.205	0.064	0.059
	Ferradans$_{10}$	7.649	6.155	20.637	21.916	20.167	21.338	0.353	0.318	0.113	0.086
	Park$_{10}$	3.773	3.250	27.076	**27.734**	26.201	**26.897**	0.179	0.154	0.054	**0.045**
	Gil (as in previous Table)	4.066	3.667	27.102	26.847	25.741	25.627	0.188	0.178	0.058	0.052
	Ours (as in previous Table)	**3.377**	**3.140**	**27.571**	27.606	**26.712**	26.632	**0.157**	**0.129**	**0.050**	0.047
Ref γ-Src log	**Kotera$_{10}$**	11.564	8.588	20.466	22.328	19.164	20.831	0.480	0.379	0.148	0.091
	Pitie$_{10}$	3.721	3.895	26.064	25.783	25.477	25.508	0.183	0.165	0.056	0.053
	Reinhard$_{10}$	5.579	5.300	24.691	24.479	23.073	22.759	0.250	0.256	0.074	0.073
	Xiao$_{10}$	4.847	4.699	25.100	25.252	24.345	24.667	0.219	0.206	0.063	0.058
	Ferradans$_{10}$	6.280	6.010	22.433	22.802	21.903	22.115	0.311	0.312	0.084	0.078
	Park$_{10}$	6.262	4.023	26.168	26.592	24.278	24.857	0.269	0.192	0.080	0.057
	Gil (as in previous Table)	3.863	3.476	27.197	26.672	25.889	25.341	0.173	0.162	0.054	0.054
	Ours (as in previous Table)	**3.444**	**3.313**	**27.395**	**26.922**	**26.563**	**25.684**	**0.152**	**0.144**	**0.050**	**0.052**

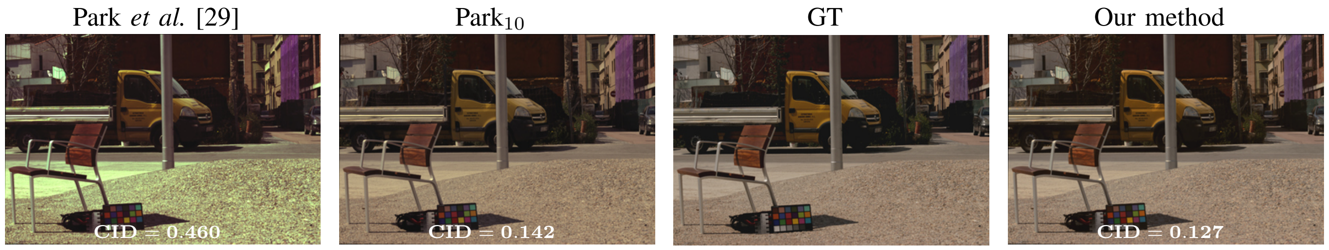

Fig. 6. Example of applying power 10 function to Park *et al.* [29]. The input images are a log-encoded reference and a gamma-corrected source. The first column presents the output of the original method from [29], the second shows the output of [29] applying power 10 (Park$_{10}$), the third shows the GT, and the last column shows our result.

Xiao and Ma [15] (Xiao), Pitié *et al.* [10] (Pitie), Ferradans *et al.* [12] (Ferradans), Park *et al.* [29] (Park), and Gil Rodríguez *et al.* [27] (Gil). The last method also considers the inputs as log-encoded images. In order to compute the quantitative results, we undo the non-linearity (since it is known) of the resulting color matched image and the GT, and then apply a γ correction of $1/2.2$, as done in the previous experiments. From the data in Table III, it is apparent that our method is accurate when working with real data and common situations.

Figure 7 presents the image results, where we show the GTs and our results after applying the tone mapping operator (TMO) from [37]. The reference and the source are presented without tone mapping, in order to appreciate the differences between applying the different curves. As it can be seen in the last column in Figure 7, our method recovers the colors and appearance of the reference image, in different input situations. We show for 3 different scenes (rows), and for each scene: the reference (first column), the source (second column), the GT (third column) and our result (last column). Notice that on the last row, where the reference is PQ-encoded and the source HLG-encoded, our result (last column) is not able to completely recover the blue on the t-shirt on the left upper corner. In our output, the blue appears brighter than in the GT. This is due to the fact that no correspondences are available in this particular hue, thus the recovery is not perfect.

V. CONCLUSION

In this paper we have presented a method for the color matching of different image views encoded with unknown non-linear curves. The method is based on the modification of logarithmic-encoded images so that they behave as gamma-corrected ones. In this way, we can color stabilize the images by estimating a 4×4 matrix and a power law value. Our results show that our method outperforms state-of-the-art algorithms quantitatively and qualitatively. In a future work, we would like to explore the more general case, when no content is shared among the input images.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant

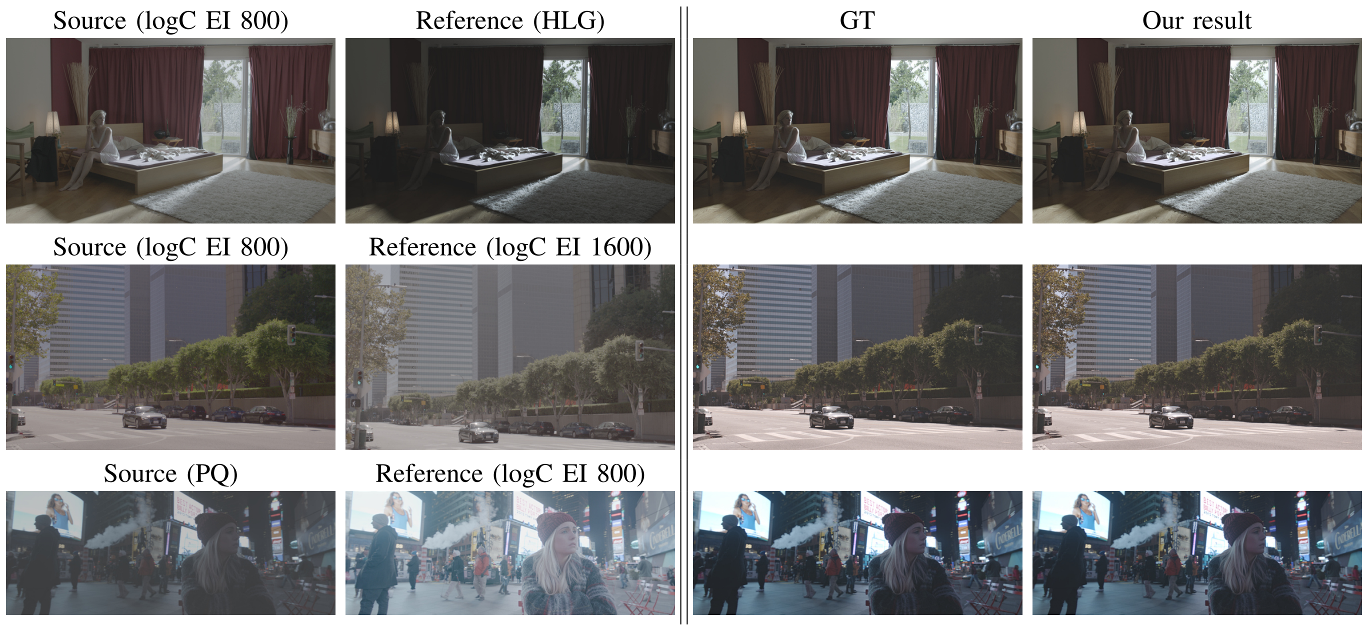

Fig. 7. Examples with PQ and HLG. From left to right, source, reference, GT and our result. Each row represents a different comparison and scenario. The GTs and our results are tone mapped using [37]. Images from ARRI [38]. In the case of PQ curve, we set up the absolute luminance of the display to 1000 cd/m^2.

TABLE III
RESULTS SHOW MEAN (μ) AND MEDIAN ($\hat{\mu}$) AVERAGES OVER 10 PAIRS, WHERE REFERENCE AND SOURCE IMAGES ARE ENCODED USING HLG, PQ AND LOGARITHMIC CURVES. IN THE CASE OF PQ CURVE, WE SET UP THE ABSOLUTE LUMINANCE OF THE DISPLAY TO 1000 cd/m^2.

		ΔE^*_{00}	PSNR L	CPSNR	CID	RMSE
Kotera	μ	3.344	32.505	30.567	0.110	0.045
	$\hat{\mu}$	4.117	30.379	29.061	0.114	0.052
Pitie	μ	1.022	40.047	40.134	0.035	0.021
	$\hat{\mu}$	0.582	43.069	42.568	0.004	0.006
Reinhard	μ	1.861	35.311	35.020	0.062	0.040
	$\hat{\mu}$	2.058	32.023	31.408	0.042	0.037
Xiao	μ	1.891	32.965	32.789	0.061	0.032
	$\hat{\mu}$	2.066	30.379	30.244	0.054	0.033
Ferradans	μ	4.820	24.692	24.624	0.183	0.073
	$\hat{\mu}$	3.865	25.364	25.485	0.145	0.052
Park	μ	1.624	38.250	36.795	0.044	0.029
	$\hat{\mu}$	1.418	37.730	37.142	0.018	0.017
Gil	μ	1.775	36.086	35.636	0.068	0.029
	$\hat{\mu}$	1.659	31.674	31.295	0.046	0.022
Ours	μ	**0.310**	**48.324**	**47.649**	**0.002**	**0.005**
	$\hat{\mu}$	**0.239**	**49.435**	**49.131**	**0.001**	**0.004**

agreement number 761544 (project HDR4EU) and under grant agreement number 780470 (project SAUCE), and by the Spanish government and FEDER Fund, grant ref. TIN2015-71537-P (MINECO/FEDER,UE).

REFERENCES

[1] S. Bianco, A. Bruna, F. Naccari, and R. Schettini, "Color space transformations for digital photography exploiting information about the illuminant estimation process," *Journal of the Optical Society of America (JOSA)*, vol. 29, no. 3, pp. 374–384, 2012.
[2] H. Brendel, "ALEXA Log C Curve- Usage in VFX," ARRI, Tech. Rep., 2011.
[3] Sony Corporation, "S-Log White Paper, S-Log within Digital Intermediate workflow designed for cinema release," SONY, Tech. Rep., 2009.
[4] SMPTE, "ST 2084:2014 - SMPTE Standard - High Dynamic Range Electro-Optical Transfer Function of Mastering Reference Displays," *SMPTE ST 2084:2014*, pp. 1–14, Aug 2014.
[5] T. Borer and A. Cotton, "A Display-Independent High Dynamic Range Television System," *SMPTE Motion Imaging Journal*, vol. 125, no. 4, pp. 50–56, 2016.
[6] MediaCollege, *CCU (Camera Control Unit) Operations*, 2012. [Online]. Available: https://www.mediacollege.com/video/production/camera-control/
[7] P. Postma and B. Chorley, "Colour Grading with Colour Management," in *SMPTE15: Persistence of Vision - Defining the Future*, 2015, pp. 1–8.
[8] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley, "Color Transfer Between Images," *IEEE Computer Graphics and Applications*, vol. 21, no. 5, pp. 34–41, 2001. [Online]. Available: http://dx.doi.org/10.1109/38.946629
[9] F. Pitié, R. Dahyot, and A. C. Kokaram, "N-Dimensional Probablility Density Function Transfer and its Application to Colour Transfer," in *IEEE International Conference on Computer Vision (ICCV)*, vol. 2, 10 2005, pp. 1434–1439.
[10] F. Pitié, A. C. Kokaram, and R. Dahyot, "Automated colour grading using colour distribution transfer," *Computer Vision and Image Understanding*, vol. 107, no. 1, pp. 123 – 137, 2007.
[11] J. Rabin, S. Ferradans, and N. Papadakis, "Adaptive color transfer with relaxed optimal transport," in *IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 4852–4856.
[12] S. Ferradans, N. Papadakis, G. Peyr, and J. Aujol, "Regularized Discrete Optimal Transport," *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1853–1882, 2014.
[13] T. Pouli and E. Reinhard, "Progressive color transfer for images of arbitrary dynamic range," *Computers & Graphics*, vol. 35, pp. 67 – 80, 2011.
[14] H. Kotera, "A scene-referred color transfer for pleasant imaging on display," in *IEEE International Conference on Image Processing (ICIP)*, vol. 2, Sept 2005, pp. 5–8.
[15] X. Xiao and L. Ma, "Color Transfer in Correlated Color Space," in *PACM International Conference on Virtual Reality Continuum and Its Applications*, ser. VRCIA. New York, NY, USA: ACM, 2006, pp. 305–309.
[16] ——, "Gradient-Preserving Color Transfer," *Computer Graphics Forum*, vol. 28, no. 7, pp. 1879–1886, 2010.
[17] R. M. H. Nguyen, S. J. Kim, and M. S. Brown, "Illuminant Aware Gamut-Based Color Transfer," *Computer Graphics Forum*, vol. 33, no. 7, pp. 319–328, Oct. 2014.
[18] Y. Hwang, J. Y. Lee, I. S. Kweon, and S. J. Kim, "Color Transfer Using Probabilistic Moving Least Squares," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014, pp. 3342–3349.
[19] H. Gong, G. Finlayson, and R. Fisher, "Recoding Color Transfer as

A Color Homography," in *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, September 2016, pp. 17.1–17.11.

[20] H. Gong, G. D. Finlayson, R. B. Fisher, and F. Fang, "3D color homography model for photo-realistic color transfer re-coding," *The Visual Computer*, pp. 1–11, 2017.

[21] Y.-W. Tai, J. Jia, and C.-K. Tang, "Local Color Transfer via Probabilistic Segmentation by Expectation-Maximization," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 747–754.

[22] Y. Xiang, B. Zou, and H. Li, "Selective color transfer with multi-source images," *Pattern Recognition Letters*, vol. 30, no. 7, pp. 682–689, May 2009.

[23] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Non-rigid Dense Correspondence with Applications for Image Enhancement," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4, pp. 70.1–70.10, Jul. 2011. [Online]. Available: http://doi.acm.org/10.1145/2010324.1964965

[24] J. Vazquez-Corral and M. Bertalmío, "Color Stabilization Along Time and Across Shots of the Same Scene, for One or Several Cameras of Unknown Specifications," *IEEE Transactions on Image Processing (TIP)*, vol. 23, no. 10, pp. 4564–4575, Oct 2014.

[25] O. Frigo, N. Sabater, J. Delon, and P. Hellier, "Motion Driven Tonal Stabilization," *IEEE Transactions on Image Processing (TIP)*, vol. 25, no. 11, pp. 5455–5468, Nov 2016.

[26] J. Vazquez-Corral and M. Bertalmío, "Log-encoding Estimation for Color Stabilization of Cinematic Footage," in *IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3349–3353.

[27] R. Gil Rodríguez, J. Vazquez-Corral, and M. Bertalmío, "Color-matching Shots from Different Cameras Having Unknown Gamma or Logarithmic Encoding Curves," in *SMPTE Annual Technical Conference & Exhibition*, 2017, pp. 1–15.

[28] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Optimizing Color Consistency in Photo Collections," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, pp. 85.1 – 85.9, 2013.

[29] J. Park, Y. W. Tai, S. N. Sinha, and I. S. Kweon, "Efficient and robust color consistency for community photo collections," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 430–438.

[30] M. Xia, J. Y. Renping, X. M. Zhang, and J. Xiao, "Color Consistency Correction Based on Remapping Optimization for Image Stitching," in *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017, pp. 2977–2984.

[31] W. Xu and J. Mulligan, "Performance evaluation of color correction approaches for automatic multi-view image and video stitching," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010, pp. 263–270.

[32] D. G. Lowe, "Object recognition from local scale-invariant features," in *IEEE International Conference on Computer Vision (ICCV)*, vol. 2, 1999, pp. 1150–1157.

[33] D. Coffin, "Decoding raw digital photos in Linux," http://www.cybercom.net/~dcoffin/dcraw/, 2010, accessed: 2010.

[34] G. Sharma, W. Wu, and E. N. Dalal, "The CIEDE2000 color-difference formula: implementation notes, supplementary test data, and mathematical observations," *Color research and application*, vol. 30, no. 1, pp. 21–30, 2005.

[35] I. Lissner, J. Preiss, P. Urban, M. S. Lichtenauer, and P. Zolliker, "Image-Difference Prediction: From Grayscale to Color," *IEEE Transactions on Image Processing (TIP)*, vol. 22, no. 2, pp. 435–446, 2013.

[36] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing (TIP)*, vol. 13, no. 4, pp. 600–612, April 2004.

[37] P. Cyriac, D. Kane, and M. Bertalmío, "Optimized Tone Curve for In-Camera Image Processing," in *IS&T Electronic Imaging Conference*, vol. 2016, no. 13, 2016, pp. 1–7.

[38] S. Andriani, H. Brendel, T. Seybold, and J. Goldstone, "Beyond the Kodak image set: A new reference set of color image sequences," in *IEEE International Conference on Image Processing (ICIP)*, Sept 2013, pp. 2289–2293.

[39] ARRI, "ARRIRAW Converter," http://www.arri.com/camera/alexa/tools/ar-riraw_converter/, 2018, accessed 2018.