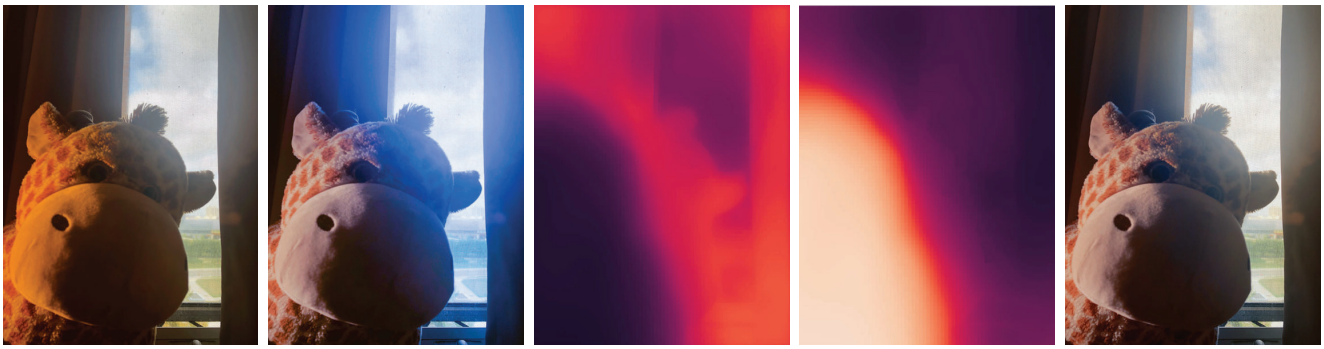
Auto White-Balance Correction for Mixed-Illuminant Scenes

Mahmoud Afifi[1] Marcus A. Brubaker[1,2] Michael S. Brown[1,2]

[1]York University [2]Vector Institute

{mafifi,mab,mbrown}@eecs.yorku.ca



(A) Camera WB correction for the outdoor light | (B) Camera WB correction for the indoor light | (C) Our weighting map for the outdoor correction | (D) Our weighting map for the indoor correction | (E) Our AWB result

Figure 1: Camera AWB correction corrects colors for only a single illumination present in a captured scene. In the mixed-illuminant scenario, this traditional WB correction often results in undesirable color tint (typically, reddish or bluish) in the final sRGB image, as shown in (A) and (B). Our method can effectively deal with such mixed-illuminant scenes by learning a set of local weighting maps to blend between different WB settings, as shown in (C) and (D), producing more compelling AWB correction compared to traditional camera AWB, as shown in (E).

Abstract

Auto white balance (AWB) is applied by camera hardware at capture time to remove the color cast caused by the scene illumination. The vast majority of white-balance algorithms assume a single light source illuminates the scene; however, real scenes often have mixed lighting conditions. This paper presents an effective AWB method to deal with such mixed-illuminant scenes. A unique departure from conventional AWB, our method does not require illuminant estimation, as is the case in traditional camera AWB modules. Instead, our method proposes to render the captured scene with a small set of predefined white-balance settings. Given this set of rendered images, our method learns to estimate weighting maps that are used to blend the rendered images to generate the final corrected image. Through extensive experiments, we show this proposed method produces promising results compared to other alternatives for single- and mixed-illuminant scene color correction.

1. Introduction

Auto white balance (AWB) is an essential procedure applied by a camera's dedicated image signal processor (ISP) hardware. The ISP "renders" the sensor image to the final output through a series of processing steps, such as white balance (WB), tone-mapping, and conversion of the final standard RGB (sRGB) output image. WB is performed early in the ISP pipeline and aims to remove undesirable color casts caused by the environment lighting [25]. AWB mimics the human visual system's ability to perform color constancy, which allows us to perceive objects as the same color regardless of scene lighting.

AWB consists of two steps. First, the scene illuminant color as observed by the camera's sensor is estimated using an illuminant estimation algorithm (e.g., [57, 44, 10, 12]). This first step assumes there is a single (i.e., global) illuminant in the scene. Second, the captured image is corrected based on the estimated illumination. These two steps are applied early in the camera's ISP to the raw image [34].

The majority of prior work focuses on the illuminant estimation step, while the WB procedure is performed using

a simple diagonal-based correction process [36]. Global illumination estimation is a well-studied topic and recent work (e.g., [60, 41, 49]) achieves impressive results of less than 2° angular errors[1] across all available single-illuminant datasets.

The assumption of a single global illuminant is well known to be an oversimplification of the problem, as many captured scenes have multiple light sources. Performing single-illuminant AWB on scenes with multiple illuminations often produces unsatisfactory results [17]. Figure 1 shows a typical example of a scene captured with a mixed lighting condition. The colors of the captured scene are corrected by the camera AWB module, which applies a single-illuminant WB correction after running an illuminant estimation algorithm on the ISP. The white-balanced image is then processed through a set of color rendering functions typical of an ISP to produce the final sRGB image, shown in Figure 1-(A). Because the captured scene has two different light sources – outdoor lighting on the right side of the scene and indoor lighting on the left side of the scene – traditional camera AWB correction results in either a reddish or bluish tint in the final rendered image.

Contribution In this paper, we present an AWB method to deal with both single- and mixed-illuminant scenes. Unlike traditional camera AWB modules, our method *does not* require illuminant estimation; instead, we propose to render the captured scene multiple times with a few fixed WB settings. Given these rendered images of the same scene, we outline a method to learn suitable pixel-wise blending maps to generate the final sRGB image. Our generated spatially varying weighting maps allow us to correct for different lighting conditions in the captured scene, as shown in Figure 1-(C–E). As a part of this effort, we propose a synthetic test set of mixed-illuminant scenes with pixel-wise ground truth. We use this set, along with other datasets ([9, 22]), to validate our method through an extensive set of experiments and comparisons. Code and trained models are available at https://github.com/mahmoudnafifi/mixedillWB.

2. Related Work

This section briefly reviews prior work for illuminant estimation and WB correction for sRGB images.

2.1. Illuminant Estimation

Working directly on a raw sensor image, illuminant estimation methods aim to predict the global scene illumination color [36]. A large body of literature performs this estimation according to some statistical hypothesis on the image colors/edges. A common statistical hypothesis for illuminant estimation is the gray-world hypothesis [21], which assumes the mean of image irradiance is achromatic (i.e., "gray") and thus computing the mean image colors could give a rough estimation of the scene illuminant color. Other statistical methods include the white-patch hypothesis [20], the bright pixels [47], the shades-of-gray [31], the gray-edges [59], the weighted-gray-edges [37], the bright-and-dark colors PCA [24], the gray pixel [53], and the grayness index [54].

Despite being hardware-friendly, the accuracy of such statistical methods is not always satisfying. Learning-based methods offer more accurate results by relying on training examples to predict illuminant colors at inference time. Representative examples of learning-based illuminant estimation methods include: gamut-based methods [32, 26, 35, 30, 16], Bayesian methods [18, 19, 55, 34], bias correction methods [27, 28, 7], and neural network-based methods [33, 23, 50, 57, 44, 52, 15, 60, 41, 49, 2].

While the majority of prior work adopts the single-illuminant assumption, only a handful of attempts have been proposed to deal with multi-illuminant scenes (e.g., [29, 43, 38, 46, 14, 45]). These prior methods employ pixel-wise illumination estimation strategies to address the problem. In some cases, the number of different lights in the scene need to be known in advance. In contrast to prior work, our AWB method can deal with single- and mixed-illuminant scenes *without* an explicit illuminant estimation step on the camera imaging pipeline.

2.2. WB Correction for sRGB Images

As mentioned earlier, traditional camera ISPs use a simple diagonal correction to white-balance captured raw images, giving the estimated illuminant colors of the scenes. After white balancing, the camera ISP applies color rendering and photo-finishing steps to render the final sRGB image. If the AWB module had some errors during the camera ISP rendering – which typically is the case when the captured scenes have mixed lighting conditions – it becomes challenging to correct colors in post-capture. A few attempts were proposed recently to deal with such a scenario by replacing the diagonal correction with a non-linear correction function [5, 6, 4]. More recently, a deep neural network (DNN) was proposed to correct/edit the WB settings of camera-rendered sRGB images [3].

These sRGB methods treat the problem as a global illumination correction post image capture. In our approach, we employ a modified camera ISP that allows us to benefit from the raw image early in the camera rendering chain. With this modified ISP, our method can effectively correct colors of both single- and mixed-illuminant scenes without requiring any post-capture correction (e.g., [5, 3]) nor interaction from the user (e.g., [6, 4]).

[1] Perceptually, angular errors less than 2° are considered barely noticeable.

Figure 2: Proposed AWB method. Unlike traditional camera AWB, shown in (A), which corrects colors for only a single illuminant estimated by an illuminant estimation method, our AWB method, shown in (B), applies a single WB correction that corrects for a fixed color temperature (e.g., 5500 K) in addition to generating small images (384×384) with a predefined set of color temperatures (e.g., 2850K, 3800K) that correlate to common lighting conditions (e.g., incandescent, fluorescent). Given these images, our method predicts blending weights to produce our final AWB result.

3. Method

The overview of our method is depicted in Figure 2. As shown, we adopted a modified camera ISP, originally introduced in [6]. This modified camera ISP produces additional small images of the same captured scene, each of which is rendered with a different WB setting. Our approach works by observing the same scene object colors under a fixed set of WB settings. The correct WB is then generated by linearly blending amoung the different WB settings. That is, given a set of small sRGB images $\{I_{c_1\downarrow}, I_{c_2\downarrow}, ...\}$, each of which was rendered with a different WB setting (i.e., c_1, c_2, ...), the final spatially varying WB colors are computed as follows:

$$I_{\texttt{corr}\downarrow} = \sum_i W_i \odot I_{c_i\downarrow}, \tag{1}$$

where $I_{\texttt{corr}\downarrow}$ is the corrected small image, i is for indexing, $\odot$ is Hadamard product, and $W_i \in \mathbb{R}^{w\times h\times 3}$ is the weighting map for the i^{th} small image rendered with the c_i WB setting. Here, w and h refer to the width and height of our small images, respectively. For simplicity, we can think of W_i as a replicated monochromatic image to match the dimensions of I_i. In our experiments at inference time, we used small images of 384×384 pixels (i.e., $w = h = 384$).

Note that, we decided to learn the weighting maps, $\{W_i\}$, for the images rendered in the sRGB space – after a full rendering of demosaiced raw images – to make our method device-independent. With that said, our method can be easily adjusted to learn the weighting maps in the linear raw space.

Producing $\{I_{c_i\downarrow}\}$ can be attained by following the modification introduced in [6] (Sec. 3.1). Then, our main task is to learn the values of W_i, given $\{I_{c_i\downarrow}\}$ (Sec. 3.2). Lastly, we apply post-processing steps to compute the final high-resolution white-balanced image (Sec. 3.3).

3.1. Modified Camera ISP

We followed the modified ISP proposed in [6], which renders a few small images, each of which is rendered with one of a set of predefined WB settings, along with the high-resolution image rendered with the camera AWB correction. In contrast to [6], we propose to use a fixed WB setting to render the high-resolution image, and thus we *do not* need an illuminant estimation module in our pipeline. In the rest of this paper, the WB settings are interchangeably referred to by lighting source names or the correlated color temperatures.

Rendering such small images through the camera ISP modules does not require major changes to standard camera ISPs. Specifically, this modification downsamples the captured raw demosaiced image into the target size of the small images. This small raw-image is then processed by the ISP color rendering modules to produce a small sRGB image. For each rendering pass, a predefined WB setting is used. Processing such small images though the ISP an additional few times requires negligible processing time [6], and thus it does not affect the real-time processing offered by the camera viewfinder.

In the photofinishing stage, we can employ these small images to map the colors of the high-resolution sRGB image, rendered with a fixed WB setting (e.g., daylight), to our set of predefined WB settings as described in the following equation:

$$\hat{I}_{c_i} = M_{c_i}\phi\left(I_{\texttt{init}}\right), \tag{2}$$

where $I_{\texttt{init}}$ is the initial high-resolution image rendered with the fixed WB setting, $\hat{I}_{c_i}$ is the mapped image to the target WB setting c_i, $\phi\left(\cdot\right)$ is a polynomial kernel function that projects the R, G, B channels into a higher-dimensional space [42], and M_{c_i} is a mapping matrix computed by fitting the colors of $I_{\texttt{init}}$, after downsampling, to the corresponding colors of $I_{c_i\downarrow}$. This fitting is computed by mini-

mizing the residual sum of squares between the source and target colors in both small images.

3.2. Learning Weighting Maps

So far, we have described the rendering process of our small images and the post-capture color mapping to generate high-resolution images with our predefined set of WB settings. Our task now is to estimate the values of W_i for a given set of our small images $\{I_{c_i\downarrow}\}$.

We employed a DNN to predict the values of $\{W_i\}$, where our network accepts the small images, rendered with our predefined WB settings, and learns to produce proper weighting maps $\{W_i\}$. Specifically, our network accepts a 3D tensor of concatenated small images and produces a 3D tensor of the weighting maps. The details of our network architecture are given in the supplementary materials.

To train our DNN model, we employed the Rendered WB dataset [5], which includes ∼65K sRGB images rendered with different color temperatures. Each image in the Rendered WB dataset [5] has the corresponding ground-truth sRGB image that was rendered with an accurate WB correction. From this dataset, we selected 9,200 training images that were rendered with "camera standard" photofinishing and with the following color temperatures: 2850 Kelvin (K), 3800 K, 5500 K, 6500 K, and 7500 K, which correspond to the following WB settings: tungsten (also referred to as incandescent), fluorescent, daylight, cloudy, and shade, respectively.We then train our network to minimize the following reconstruction loss function:

$$\mathcal{L}_r = \|P_{\texttt{corr}} - \sum_i \hat{W}_i \odot P_{c_i}\|_F^2, \tag{3}$$

where $\|\cdot\|_F^2$ computes the squared Frobenius norm, $P_{\texttt{corr}}$ and P_{c_i} are the extracted training patches from ground-truth sRGB images and input sRGB images rendered with the c_i WB setting, respectively, and $\hat{W}_i$ is the blending weighting map produced by our network for P_{c_i}.

To avoid producing out-of-gamut colors in the reconstructed image, we apply a cross-channel softmax operator to the output tensor of our network before computing our loss in Equation 3.

We apply the following regularization term to the produced weighting maps to encourage our network to produce smooth weights:

$$\mathcal{L}_s = \sum_i \|\hat{W}_i * \nabla_x\|_F^2 + \|\hat{W}_i * \nabla_y\|_F^2 \tag{4}$$

where ∇_x and ∇_y are 3×3 horizontal and vertical Sobel filters, respectively, and $*$ is the convolution operator. Thus, our final loss is computed as follows:

$$\mathcal{L} = \mathcal{L}_r + \lambda \mathcal{L}_s, \tag{5}$$

where λ is a scaling factor to control the contribution of $\mathcal{L}_s$ to our final loss. In our experiments, we used $\lambda = 100$. Minimizing the loss in Equation 5 was performed using the Adam optimizer [48] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for 200 epochs. We used a mini-batch size of 32, where at each iteration we randomly selected eight training images, alongside their associated images rendered with our predefined WB set, and extracted four random patches from each of the selected training images.

3.3. Inference

At inference time, we produce our small image with the predefined WB settings. These images are concatenated and fed to our DNN, which produces the weighting maps. To improve the final results, we advocate an ensembling strategy, where we feed the concatenated small images in three different scales: 1.0, 0.5, 0.25. Then, we upsample the produced weights to the high-resolution image size. Afterward, we compute the average weighting maps produced for each WB setting. This ensemble strategy produces weights with more local coherence. This coherence can be further improved by applying a post-processing edge-aware smoothing step. That is, we used our high-resolution images (i.e., the initially rendered one with the fixed WB setting and the mapped images produced by Equation 2) as a guidance image to post-process the generated weights. We utilized the fast bilateral solver for this task [11]. See the supplementary materials for an ablation study.

After generating our upsampled weighting maps $\{\hat{W}_{i\uparrow}\}$, the final image is generated as described in the following equation:

$$\hat{I}_{\texttt{corr}} = \sum_i \hat{W}_{i\uparrow} \odot \hat{I}_{c_i}. \tag{6}$$

4. Experiments

We conducted several experiments to evaluate our method. In all experiments, we set the fixed WB setting to 5500 K (i.e., daylight WB). We examined two different predefined WB sets to render our small images, which are: (i) : {2850 K, 3800 K, 6500 K, 7500 K} and (ii) {2850 K, 7500 K}. These two sets represent the following WB settings: {tungsten, fluorescent, cloudy, and shade} and {tungsten, shade}, respectively. Our network received the concatenated images, along with the downsampled daylight image. Thus, we will refer to the first predefined WB set as $\{\texttt{t}, \texttt{f}, \texttt{d}, \texttt{c}, \texttt{s}\}$ and the second set is referred to as $\{\texttt{t}, \texttt{d}, \texttt{s}\}$, where the terms t, f, d, c, and s refer to tungsten, fluorescent, daylight, cloudy, and shade, respectively.

When WB=$\{\texttt{t}, \texttt{d}, \texttt{s}\}$ is used, our DNN takes ∼0.6 seconds on average without ensembling, while it takes ∼0.9 seconds on average when ensembling is used. This processing time is reported using a single NVIDIA GeForce GTX 1080 graphics card. When WB=$\{\texttt{t}, \texttt{f}, \texttt{d}, \texttt{c}, \texttt{s}\}$ is used, our

(A) 3D scene in 3Ds Max (B) Rendered with 6500 Kelvin (K) (C) Rendered with 2850 K (D) Ground truth

Figure 3: Our mixed-illuminant evaluation set. (A) We began with 3D scenes modeled in Autodesk 3Ds Max [58]. (B-C) We set the WB setting of the virtual camera to different color temperatures and produced realistic scene images with multiple illuminations using Vray rendering [1]. (D) For each input image in our test set, a ground-truth image is provided.

DNN takes ∼0.8 seconds and ∼0.95 seconds on average to process with and without ensembling, respectively.

The average CPU processing time of the post-processing edge-aware smoothing for images with 2K resolution is ∼3.4 seconds and ∼5.7 seconds with WB=$\{\texttt{t}, \texttt{d}, \texttt{s}\}$ and WB=$\{\texttt{t}, \texttt{f}, \texttt{d}, \texttt{c}, \texttt{s}\}$, respectively. This post-processing step is optional (see the supplementary materials for comparisons) and its processing time can be reduced with a GPU implementation of the edge-aware smoothing step.

In the remaining part of this section, we elaborate the details of our experiments and report both qualitative and quantitative results of our method.

4.1. Evaluation Sets

We used three different datasets to evaluate our method. The scenes and camera models present in these evaluation sets were not part of our training data.

Cube+ dataset [9] As the Cube+ dataset [9] is commonly used by prior color constancy work, we used the linear raw images provided in the Cube+ dataset to evaluate our AWB method for single-illuminant scenes. Specifically, we used the raw images in order to generate the small images to feed them into our DNN.

MIT-Adobe 5K dataset [22] We selected a set of mixed-illuminant scenes from the MIT-Adobe 5K dataset [22] for qualitative comparisons. Similar to the Cube+ dataset, the MIT-Adobe 5K dataset [22] provides the linear raw DNG file of each image, which facilitates our small image rendering process. This set is used to compare our results against the traditional camera AWB results by using the "as-shot" WB values stored in each DNG raw file.

Our mixed-illuminant test set As a part of our contribution, we generated a synthetic testing set to quantitatively evaluate mixed-illuminant scene WB methods. To generate our test set, we started with 3D scenes modeled in Autodesk 3Ds Max [58]. We added multiple light sources in each scene, such that each scene has at least two types of light sources (e.g., indoor and outdoor lighting). For each scene, we set the virtual camera in different poses and used different color temperatures (e.g., 28500 K, 5500 K) for the camera AWB setting. Then, we rendered the final scene images using Vray rendering [1] to generate realistic photographs. In total, we rendered 150 images with mixed lighting conditions. Each image comes with its associated ground-truth image. To render the ground-truth images, we set the color temperature of our virtual camera's WB and all light sources in the scene to 5500 K (i.e., daylight). Note that setting the color temperature does not affect the intensity of the final rendered image pixels, but only changes the lighting colors; see Figure 3. In comparison with existing multi-illuminant datasets (e.g., [14, 13, 51, 40, 39]), our test set is the first set that includes images rendered with different WB settings along with the corresponding ground-truth images rendered with correctly white-balanced pixels in the rendering pipeline. See supplementary materials for further discussion.

4.2. Results

Figure 4 shows examples of the generated blending weights. In this figure, we compare our results, shown in Figure 4-(H), with the traditional camera AWB results, shown in Figure 4-(D). Figure 4 also shows the initial sRGB image rendered with the fixed WB setting (i.e., daylight), along with the mapped images to the predefined WB settings, as described in Equation 2. As can be seen, our method learned to produce proper blending weights to locally white-balance each region in the captured scene.

Table 1 shows the results of our AWB method on the single-illuminant Cube+ dataset [9]. In this table, we also show the results of a set of ablation studies conducted to show the impact of some training options. Specifically, we show the results of training our network on the following patch sizes (p): 256×256, 128×128, and 64×64 pixels.

We also show the results of our method when training without using the smoothing loss term ($\mathcal{L}_s$). The impact of $\mathcal{L}_s$ can also be shown in Figure 5. In addition, we show

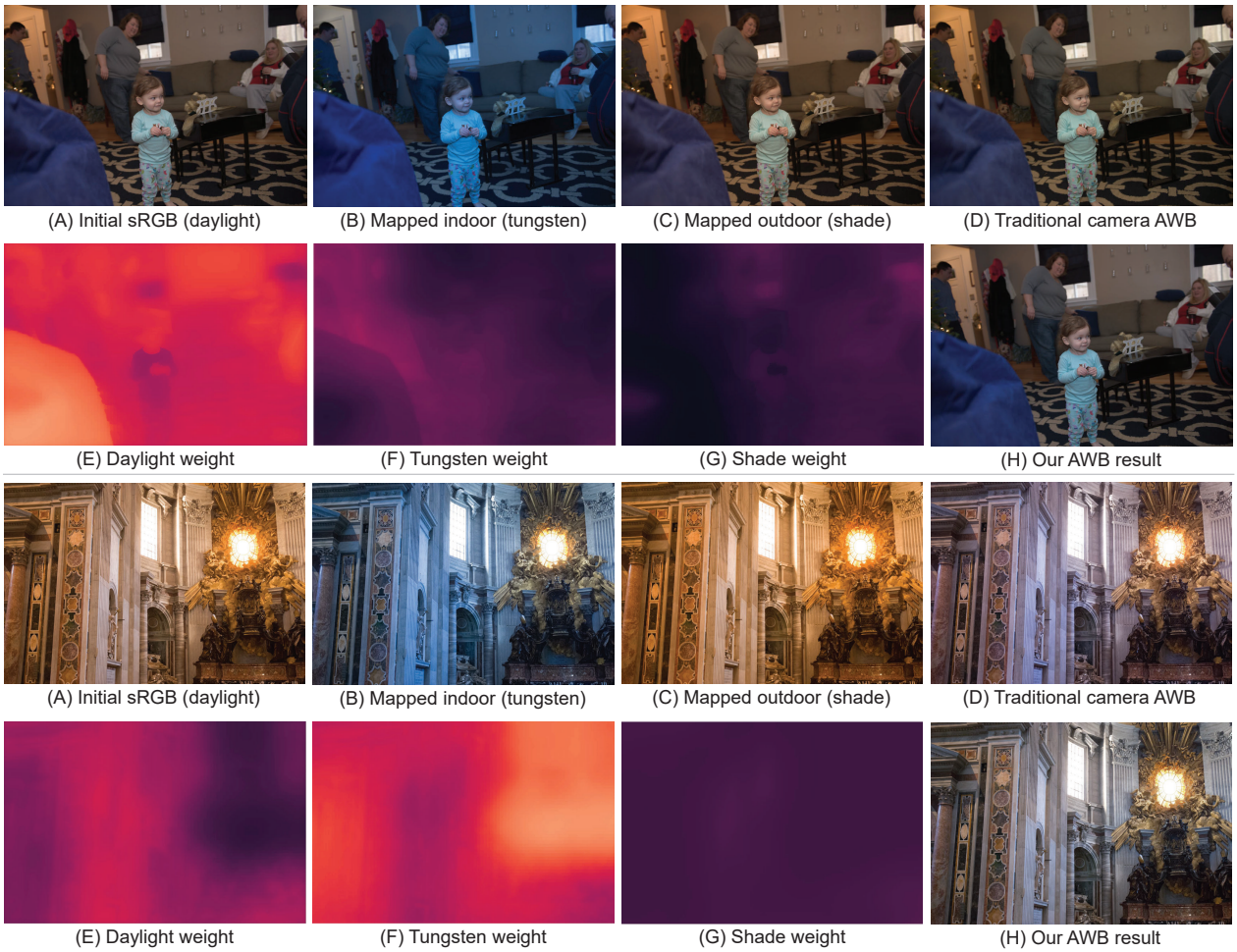


Figure 4: Predicted weighting maps. (A) Initial rendered image with daylight WB applied. (B-C) High-resolution images with indoor and outdoor WB settings after color mapping. (D) Traditional camera AWB. (E-G) predicted weights for each WB setting. (H) Our final AWB. Images are from the MIT-Adobe 5K dataset [22].

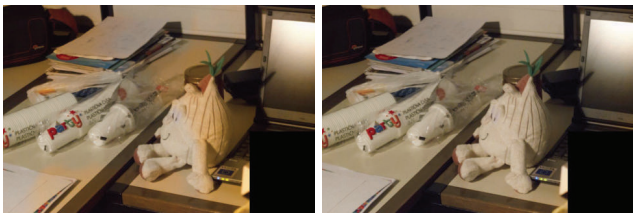


Figure 5: Impact of the smoothing loss term ($\mathcal{L}_s$). (A) Result without $\mathcal{L}_s$. (B) Results with $\mathcal{L}_s$. Input image are from the Cube+ dataset [9].

the results of training our method without keeping the same order of concatenated small images during training. That is, we shuffle the order of concatenated small images before feeding the images to the network during training to see if that would affect the accuracy. As shown in Table 1, using a fixed order achieves better results as that helps our network to build knowledge on the correlation between pixel colors when rendered with different WB settings and the generated weighting maps. Table 1 also shows that when comparing our method with recent methods for WB correction, our method achieves very competitive results, while requiring small memory overhead compared to the state-of-the-art methods (e.g., [3]).

In Table 2, we report the results of our method, along with the results of recently published methods for WB correction in sRGB space, on our synthetic test set. As shown, our method achieves the best results over most evaluation metrics. Table 2 shows the results when training our method using different predefined sets of WB settings.

Figure 6 shows qualitative comparisons on images from the MIT-Adobe 5K dataset [22]. We compare our results, shown in Figure 6-(D), against traditional camera AWB (Figure 6-[A]), after applying the KNN-WB correction [5] (Figure 6-[B]), and after applying the deep WB correction

Figure 6: Qualitative comparisons with other AWB methods on the MIT-Adobe 5K dataset [22]. Shown are the results of the following methods: gray pixel [53], grayness index [54], interactive WB [4], KNN WB [5], deep WB [3], and our method.

[3] (Figure 6-[C]). It is clear that our method provides better per-pixel correction compared to other alternatives.

Our method has the limitation of requiring a modification to an ISP to render the small images. To overcome this, one could employ one of the sRGB WB editing methods to synthesize rendering our small images with the target predefined WB set in post-capture time. In Figure 7, we illustrate this idea by employing the deep WB [3] to generate the small images of a given sRGB camera-rendered image taken from Flickr. As shown, our method produces a better result compared to the camera-rendered image (i.e., traditional camera AWB) and the deep WB result for post-capture WB correction.

Lastly, Figure 8 shows a failure example of our method, where it could not properly correct colors of each pixel in the image. In such cases, our method produces results that are very similar to correcting only for a single illuminant, as it is bounded by our predefined set of WB settings.

5. Conclusion

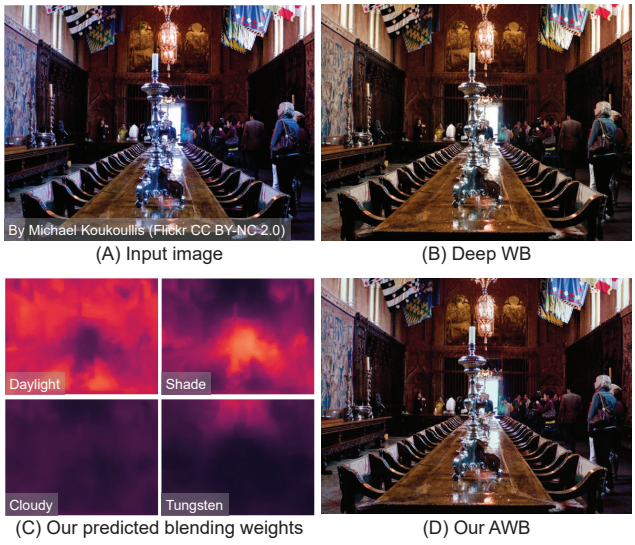
We have presented an AWB method for mixed-illuminant scenes. Our method achieves local WB correction by producing weighting maps that blend between the same input image rendered with different WB settings. Our proposed method learns to produce these local weighting maps through a DNN. Our method is fast

Table 1: Comparison on the single-illuminant Cube+ dataset [9]. For our method, we report the results of several ablation studies, where we trained our model with different WB settings and patch sizes (p). The terms `t`, `f`, `d`, `c`, and `s` refer to tungsten, fluorescent, daylight, cloudy, and shade, respectively. We followed prior work [5] and reported the mean, first, second (median), and third quantile (Q1, Q2, and Q3) of mean square error (MSE), mean angular error (MAE), and △E 2000 [56]. For all diagonal-based methods, gamma linearization [8, 25] is applied. The top results are indicated with yellow and bold, while second and third best results are indicated with green and red, respectively.

Method	MSE				MAE				△E 2000				Size
	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3	
FC4 [44]	371.9	79.15	213.41	467.33	6.49°	3.34°	5.59°	8.59°	10.38	6.6	9.76	13.26	5.89 MB
Quasi-U CC [15]	292.18	15.57	55.41	261.58	6.12°	1.95°	3.88°	8.83°	7.25	2.89	5.21	10.37	622 MB
KNN WB [5]	194.98	27.43	57.08	118.21	4.12°	1.96°	3.17°	5.04°	5.68	3.22	4.61	6.70	21.8 MB
Interactive WB [4]	159.88	21.94	54.76	125.02	4.64°	2.12°	3.64°	5.98°	6.2	3.28	5.17	7.45	**38 KB**
Deep WB [3]	**80.46**	15.43	33.88	**74.42**	**3.45°**	1.87°	2.82°	**4.26°**	**4.59**	2.68	3.81	**5.53**	16.7 MB
Our results													
$p = 256$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$	235.07	54.42	83.34	197.46	6.74°	4.12°	5.31°	8.11°	8.07	5.22	7.09	10.04	5.10 MB
$p = 128$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$	176.38	16.96	35.91	115.50	4.71°	2.10°	3.09°	5.92°	5.77	3.01	4.27	7.71	5.10 MB
$p = 64$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$	161.80	9.01	**19.33**	90.81	4.05°	1.40°	**2.12°**	4.88°	4.89	2.16	**3.10**	6.78	5.10 MB
$p = 64$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$, w/o $\mathcal{L}_s$	189.18	11.10	23.66	112.40	4.59°	1.57°	2.41°	5.76°	5.48	2.38	3.50	7.80	5.10 MB
$p = 64$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$, w/ shuff.	197.21	24.48	55.77	149.95	5.36°	2.60°	3.90°	6.74°	6.66	3.79	5.45	8.65	5.10 MB
$p = 64$, WB=$\{\texttt{t},\texttt{d},\texttt{s}\}$	168.38	**8.97**	19.87	105.22	4.20°	**1.39°**	2.18°	5.54°	5.03	**2.07**	3.12	7.19	5.09 MB

Table 2: Comparison on our mixed-illuminant evaluation set. Highlights and symbols are the same as in Table 1.

Method	MSE				MAE				△E 2000			
	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3
Gray pixel [53]	4959.20	3252.14	4209.12	5858.69	19.67°	11.92°	17.21°	27.05°	25.13	19.07	22.62	27.46
Grayness index [54]	1345.47	727.90	1055.83	1494.81	6.39°	4.72°	5.65°	7.06°	12.84	9.57	12.49	14.60
KNN WB [5]	1226.57	680.65	1062.64	1573.89	5.81°	4.29°	5.76°	6.85°	12.00	9.37	11.56	13.61
Interactive WB [4]	1059.88	616.24	896.90	1265.62	5.86°	4.56°	5.62°	6.62°	11.41	8.92	10.99	12.84
Deep WB [3]	1130.598	621.00	886.32	1274.72	**4.53°**	**3.55°**	4.19°	**5.21°**	10.93	**8.59**	**9.82**	11.96
Ours ($p = 64$, WB=$\{\texttt{t},\texttt{d},\texttt{s}\}$)	**819.47**	655.88	**845.79**	1000.82	5.43°	4.27°	4.89°	6.23°	**10.61**	9.42	10.72	**11.81**
Ours ($p = 64$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$)	938.02	757.49	961.55	1161.52	4.67°	3.71°	**4.14°**	5.35°	12.26	10.80	11.58	12.76
Ours ($p = 128$, WB=$\{\texttt{t},\texttt{d},\texttt{s}\}$)	830.20	**584.77**	853.01	**992.56**	5.03°	3.93°	4.78°	5.90°	11.41	9.76	11.39	12.53
Ours ($p = 128$, WB=$\{\texttt{t},\texttt{f},\texttt{d},\texttt{c},\texttt{s}\}$)	1089.69	846.21	1125.59	1279.39	5.64°	4.15°	5.09°	6.50°	13.75	11.45	12.58	15.59



(A) Input image (B) Deep WB

(C) Our predicted blending weights (D) Our AWB

Figure 7: Post-capture WB correction. By employing an off-the-shelf method for global WB editing (e.g., [3]), we can employ our method to correct mixed-illuminant scene colors in post-capture time. Input image is from Flickr.

and thus can be deployed on board a camera ISP hardware. As a part of our contribution, we have proposed a synthetic test set of mixed-illuminant scenes with pixel-wise ground truth. Compared with other alternatives, we showed that our method produces promising results through both qualitative and quantitative evaluations. Our source code, trained models, and synthetic test set are available at https://github.com/mahmoudnafifi/mixedillWB.

(A) Camera AWB (B) Ours

Figure 8: Failure examples. Our method fails in some cases, where it could not properly choose the correct WB setting for some regions in the input image. Input image is from the MIT-Adobe 5K dataset [22].

Acknowledgement

This study was funded in part by the Canada First Research Excellence Fund for the Vision: Science to Applications (VISTA) programme and an NSERC Discovery Grant. The authors would like to thank Mohammed Hossam for the help in generating our synthetic test set.

References

[1] V-ray for 3ds max. https://www.chaosgroup.com/3d-rendering-software. Accessed: 2021-07-20.

[2] Mahmoud Afifi, Jonathan T Barron, Chloe LeGendre, Yun-Ta Tsai, and Francois Bleibel. Cross-camera convolutional color constancy. In *ICCV*, 2021.

[3] Mahmoud Afifi and Michael S Brown. Deep white-balance editing. In *CVPR*, 2020.

[4] Mahmoud Afifi and Michael S Brown. Interactive white balancing for camera-rendered images. In *Color and Imaging Conference*, 2020.

[5] Mahmoud Afifi, Brian Price, Scott Cohen, and Michael S Brown. When color constancy goes wrong: Correcting improperly white-balanced images. In *CVPR*, 2019.

[6] Mahmoud Afifi, Abhijith Punnappurath, Abdelrahman Abdelhamed, Hakki Can Karaimer, Abdullah Abuolaim, and Michael S Brown. Color temperature tuning: Allowing accurate post-capture white-balance editing. In *Color and Imaging Conference*, 2019.

[7] Mahmoud Afifi, Abhijith Punnappurath, Graham Finlayson, and Michael S Brown. As-projective-as-possible bias correction for illumination estimation algorithms. *JOSA A*, 36(1):71–78, 2019.

[8] Matthew Anderson, Ricardo Motta, Srinivasan Chandrasekar, and Michael Stokes. Proposal for a standard default color space for the Internet - sRGB. In *Color and Imaging Conference*, 1996.

[9] Nikola Banić and Sven Lončarić. Unsupervised learning for color constancy. *arXiv preprint arXiv:1712.00436*, 2017.

[10] Jonathan T Barron. Convolutional color constancy. In *ICCV*, 2015.

[11] Jonathan T Barron and Ben Poole. The fast bilateral solver. In *ECCV*, 2016.

[12] Jonathan T Barron and Yun-Ta Tsai. Fast Fourier color constancy. In *CVPR*, 2017.

[13] Shida Beigpour, Mai Lan Ha, Sven Kunz, Andreas Kolb, and Volker Blanz. Multi-view multi-illuminant intrinsic dataset. In *BMVC*, 2016.

[14] Shida Beigpour, Christian Riess, Joost Van De Weijer, and Elli Angelopoulou. Multi-illuminant estimation with conditional random fields. *IEEE Transactions on Image Processing*, 23(1):83–96, 2013.

[15] Simone Bianco and Claudio Cusano. Quasi-unsupervised color constancy. In *CVPR*, 2019.

[16] Simone Bianco and Raimondo Schettini. Color constancy using faces. In *CVPR*, 2012.

[17] Michael Bleier, Christian Riess, Shida Beigpour, Eva Eibenberger, Elli Angelopoulou, Tobias Tröger, and André Kaup. Color constancy and non-uniform illumination: Can existing algorithms work? In *ICCV Workshops*, 2011.

[18] David H Brainard and William T Freeman. Bayesian method for recovering surface and illuminant properties from photosensor responses. In *Human Vision, Visual Processing, and Digital Display V*, volume 2179, pages 364–377, 1994.

[19] David H Brainard and William T Freeman. Bayesian color constancy. *Journal of the Optical Society of America A*, 14(7):1393–1411, 1997.

[20] David H Brainard and Brian A Wandell. Analysis of the retinex theory of color vision. *Journal of the Optical Society of America A*, 3(10):1651–1661, 1986.

[21] Gershon Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin Institute*, 310(1):1–26, 1980.

[22] Vladimir Bychkovsky, Sylvain Paris, Eric Chan, and Frédo Durand. Learning photographic global tonal adjustment with a database of input / output image pairs. In *CVPR*, 2011.

[23] Vlad C Cardei, Brian Funt, and Kobus Barnard. Estimating the scene illumination chromaticity by using a neural network. *Journal of the Optical Society of America A*, 19(12):2374–2386, 2002.

[24] Dongliang Cheng, Dilip K Prasad, and Michael S Brown. Illuminant estimation for color constancy: Why spatial-domain methods work and the role of the color distribution. *Journal of the Optical Society of America A*, 31(5):1049–1058, 2014.

[25] Marc Ebner. *Color Constancy*, volume 6. John Wiley & Sons, 2007.

[26] Graham Finlayson and Steven Hordley. Improving gamut mapping color constancy. *IEEE Transactions on Image Processing*, 9(10):1774–1783, 2000.

[27] Graham D Finlayson. Corrected-moment illuminant estimation. In *ICCV*, 2013.

[28] Graham D. Finlayson. Colour and illumination in computer vision. *Interface Focus*, 8(4):1–8, 2018.

[29] Graham D Finlayson, Brian V Funt, and Kobus Barnard. Color constancy under varying illumination. In *ICCV*, pages 720–725, 1995.

[30] Graham D Finlayson, Steven D Hordley, and Ingeborg Tastl. Gamut constrained illuminant estimation. *International Journal of Computer Vision*, 67(1):93–109, 2006.

[31] Graham D Finlayson and Elisabetta Trezzi. Shades of gray and colour constancy. In *Color and Imaging Conference*, 2004.

[32] David A Forsyth. A novel algorithm for color constancy. *International Journal of Computer Vision*, 5(1):5–35, 1990.

[33] Brian Funt, Vlad Cardei, and Kobus Barnard. Learning color constancy. In *Color and Imaging Conference*, 1996.

[34] Peter Vincent Gehler, Carsten Rother, Andrew Blake, Tom Minka, and Toby Sharp. Bayesian color constancy revisited. In *CVPR*, 2008.

[35] Arjan Gijsenij, Theo Gevers, and Joost Van De Weijer. Generalized gamut mapping using image derivative structures for color constancy. *International Journal of Computer Vision*, 86(2):127–139, 2010.

[36] Arjan Gijsenij, Theo Gevers, and Joost Van De Weijer. Computational color constancy: Survey and experiments. *IEEE Transactions on Image Processing*, 20(9):2475–2489, 2011.

[37] Arjan Gijsenij, Theo Gevers, and Joost Van De Weijer. Improving color constancy by photometric edge weighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):918–929, 2012.

[38] Arjan Gijsenij, Rui Lu, and Theo Gevers. Color constancy for multiple light sources. *IEEE Transactions on Image Processing*, 21(2):697–707, 2011.

[39] Xiangpeng Hao and Brian Funt. A multi-illuminant synthetic image test set. *Color Research & Application*, 45(6):1055–1066, 2020.

[40] Xiangpeng Hao, Brian Funt, and Hanxiao Jiang. Evaluating colour constancy on the new mist dataset of multi-illuminant scenes. In *Color and Imaging Conference*, volume 2019, 2019.

[41] Daniel Hernandez-Juarez, Sarah Parisot, Benjamin Busam, Ales Leonardis, Gregory Slabaugh, and Steven McDonagh. A multi-hypothesis approach to color constancy. In *CVPR*, 2020.

[42] Guowei Hong, M Ronnier Luo, and Peter A Rhodes. A study of digital camera colorimetric characterization based on polynomial modeling. *Color Research & Application*, 26(1):76–84, 2001.

[43] Eugene Hsu, Tom Mertens, Sylvain Paris, Shai Avidan, and Frédo Durand. Light mixture estimation for spatially varying white balance. In *SIGGRAPH*. 2008.

[44] Yuanming Hu, Baoyuan Wang, and Stephen Lin. FC4: Fully convolutional color constancy with confidence-weighted pooling. In *CVPR*, 2017.

[45] Md Akmol Hussain and Akbar Sheikh Akbari. Color constancy algorithm for mixed-illuminant scene images. *IEEE Access*, 6:8964–8976, 2018.

[46] Hamid Reza Vaezi Joze and Mark S Drew. Exemplar-based color constancy and multiple illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(5):860–873, 2013.

[47] Hamid Reza Vaezi Joze, Mark S Drew, Graham D Finlayson, and Perla Aurora Troncoso Rey. The role of bright pixels in illumination estimation. In *Color and Imaging Conference*, 2012.

[48] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[49] Yi-Chen Lo, Chia-Che Chang, Hsuan-Chao Chiu, Yu-Hao Huang, Chia-Ping Chen, Yu-Lin Chang, and Kevin Jou. CLCC: Contrastive learning for color constancy. In *CVPR*, 2021.

[50] Zhongyu Lou, Theo Gevers, Ninghang Hu, Marcel P Lucassen, et al. Color constancy by deep learning. In *BMVC*, 2015.

[51] Lukas Murmann, Michael Gharbi, Miika Aittala, and Fredo Durand. A dataset of multi-illumination images in the wild. In *ICCV*, 2019.

[52] Seoung Wug Oh and Seon Joo Kim. Approaching the computational color constancy as a classification problem through deep learning. *Pattern Recognition*, 61:405–416, 2017.

[53] Yanlin Qian, Ke Chen, Jarno Nikkanen, Joni-Kristian Kämäräinen, and Jiri Matas. Revisiting gray pixel for statistical illumination estimation. In *VISAPP*, 2019.

[54] Yanlin Qian, Joni-Kristian Kämäräinen, Jarno Nikkanen, and Jiri Matas. On finding gray pixels. In *CVPR*, 2019.

[55] Charles Rosenberg, Alok Ladsariya, and Tom Minka. Bayesian color constancy with non-gaussian models. In *NeurIPS*, 2004.

[56] Gaurav Sharma, Wencheng Wu, and Edul N Dalal. The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30, 2005.

[57] Wu Shi, Chen Change Loy, and Xiaoou Tang. Deep specialized network for illuminant estimation. In *ECCV*, 2016.

[58] Sham Tickoo. *Autodesk 3Ds Max 2021: A comprehensive guide*. Cadcim Technologies, 2020.

[59] Joost Van De Weijer, Theo Gevers, and Arjan Gijsenij. Edge-based color constancy. *IEEE Transactions on Image Processing*, 16(9):2207–2214, 2007.

[60] Bolei Xu, Jingxin Liu, Xianxu Hou, Bozhi Liu, and Guoping Qiu. End-to-end illuminant estimation based on deep metric learning. In *CVPR*, 2020.