WBFlow: Few-shot White Balance for sRGB Images via Reversible Neural Flows

Chunxiao Li , Xuejing Kang , Anlong Ming*

School of Computer Science (National Pilot Software Engineering School), Beijing University of Posts and Telecommunications

{chunxiaol, kangxuejing, mal}@bupt.edu.cn

Abstract

The white balance methods for sRGB images (sRGB-WB) aim to remove their non-linear color cast without access to raw values. Although the existing sRGB-WB methods have achieved increasingly better white balance (WB) results, their generalization to the sRGB images from multiple cameras is still under-explored. In this paper, we propose an sRGB-WB network named WBFlow, which not only performs superior white balance for sRGB images but also generalizes to multiple cameras well. In detail, we take advantage of neural flow to ensure the reversibility of WBFlow, which allows it to losslessly render color-cast sRGB images back to pseudo-raw features for linear white balancing, thus achieving superior performance. Furthermore, inspired by the inter-camera approach, we design a camera transformation (CT) in the pseudo-raw feature space for generalizing the WBFlow to different cameras via few-shot learning. Given a few sRGB images from an untrained camera, our WBFlow can perform well on this camera by learning the camera-specific parameters of CT from these images. Extensive experiments show that WBFlow achieves state-of-the-art multi-camera generalization and WB accuracy for sRGB images on three public datasets and our rendered multi-camera sRGB dataset.

1 Introduction

The color cast in sRGB images is caused by improper color temperature settings in the white balance (WB) module and then exacerbated by the non-linear color renderings in the image signal processor (ISP) [Afifi *et al.*, 2019a]. White balance for sRGB images (sRGB-WB) aims to directly correct such color cast without access to raw values. It is an emerging direction and has an essential impact on high-level computer vision tasks, such as object recognition and image segmentation [Afifi and Brown, 2019].

To perform accurate sRGB-WB, an ideal way is to reverse the color-cast sRGB images to the unprocessed raw values for linear correcting and then re-render them to the white-balanced sRGB images [Afifi and Brown, 2020a]. However, it is challenging for the sRGB-WB methods to simulate this process because essential reversibility is difficult to achieve without the raw values [Chakrabarti *et al.*, 2014]. In fact, for achieving sRGB-WB, instead of the above ideal WB process, recent methods [Afifi *et al.*, 2019a; Afifi *et al.*, 2019b; Afifi and Brown, 2020b; Afifi and Brown, 2019; Afifi *et al.*, 2020] utilize *irreversible* polynomial kernel functions [Hong *et al.*, 2001] to train the mappings between color-cast sRGB images and ground truths (GTs). These trained mappings are not appropriate on untrained cameras due to the inherent color differences in the sRGB images from different cameras [Afifi, 2021]. This seriously limits their generalization to untrained cameras in real-world applications. In Figure 1(b), the representative sRGB-WB method, KNN-WB [Afifi *et al.*, 2019a], results in a noticeable greenish cast on the untrained camera.

Figure 1: Existing sRGB-WB methods are not effective in generalizing well to multiple cameras: KNN-WB [Afifi *et al.*, 2019a] has a negative WB performance for the sRGB image from the untrained camera, as in the obvious greenish output in the second row of (b). Although Deep WB [Afifi and Brown, 2020a] suppresses this color bias, this comes at the cost of worse WB performance for the sRGB image from the trained camera, as in the output with dark colors in the first row of (c). Comparatively, our WBFlow shows superior WB performance on both trained and untrained cameras, as shown in (d), it generates the outputs with similar color histograms to GTs in (e).

Few endeavors have been made to address this problem. The only method, Deep WB [Afifi and Brown, 2020a], proposes modeling the ideal WB process end-to-end via U-Net [Ronneberger *et al.*, 2015]. It is further generalized to different cameras by jointly training the multi-camera sRGB images end-to-end. However, due to the lossy structure of

the U-Net and the vanilla training strategy, Deep WB is *irreversible* and *suppresses camera specificity*. Thus, as exemplified in Figure 1(c), while improving the generalization to the untrained camera, Deep WB does so at the cost of degraded WB performance on the trained camera.

In this paper, we propose an sRGB-WB network called WBFlow that generalizes well to both trained and untrained cameras. Unlike the existing *irreversible* sRGB-WB methods, we take advantage of the neural flow [Kingma and Dhariwal, 2018] to ensure the *reversibility* of the WBFlow. Specifically, we design the Reversible Non-linear Rendering Transformation (RNRT) that simulates the ISP color renderings by the additive coupling layers [Kingma and Dhariwal, 2018]. Due to the inherent reversibility of additive coupling layers, the RNRT can render the color-cast sRGB image back to its pseudo-raw feature losslessly via forward propagation. Then, to correct the color cast of the pseudo-raw feature, we present the Reversible Linear Correction Transformation (RLCT) that simulates the white balance and color transformation matrices as reversible 1×1 convolutions [Kingma and Dhariwal, 2018]. By separating the color information and the content information, the RLCT corrects the color information of the pseudo-raw feature while ensuring the integrity of the content information, which helps to lossless reverse propagation of RNRT to generate the white-balanced sRGB image. Further, inspired by the fact that inter-camera transformation can be achieved by learning the mapping between raw color values from two cameras [Banić *et al.*, 2017], we generalize WBFlow to multiple cameras via inserting a camera transformation (CT) layer in the pseudo-raw space. In detail, by updating the weightings of CT from a few sRGB images of the input camera, we *enhance the camera specificity of the pseudo-raw feature*, which cooperates with RNRT and RLCT to generate the white-balanced sRGB image for the input camera. Our contributions are as follows:

(1) We propose the WBFlow that generalizes well to multiple cameras via reversible neural flows and few-shot learning.

(2) We propose RNRT and RLCT to simulate the nonlinear color renderings and linear color transformations by reversible additive coupling and 1×1 convolutions respectively, which enables the reversibility of our WBFlow.

(3) We propose the CT that generalizes the WBFlow to multiple cameras by learning the camera-specific weights with a few sRGB images from those cameras.

(4) Extensive experiments on three public datasets and our rendered multi-camera sRGB dataset show that WBFlow achieves superior multi-camera generalization.

2 Related Works

White Balance for Raw Images. White balance for raw images falls into the research of computational color constancy (CCC). Most CCC methods [Gijsenij and Gevers, 2007; Foster, 2011; Hu *et al.*, 2017; Lo *et al.*, 2021; Song *et al.*, 2021; Xu *et al.*, 2021; Tang *et al.*, 2022; Zhang *et al.*, 2022] achieve the goal of correcting the color cast in raw images by estimating the illumination colors that form the linear WB matrix. They make up the camera ISP's WB module. However, none of these methods are intended to be applied to sRGB images [Afifi *et al.*, 2019a]. This is because, due to the non-linear ISP renderings, the linear color cast that the CCC methods are based on is broken. In this paper, we focus on correcting the non-linear color cast on sRGB images.

White Balance for sRGB Images. The sRGB-WB is an emerging research topic. Recent sRGB-WB methods corrected the color cast in sRGB images by optimizing the non-linear mappings (exemplar-based method) [Afifi *et al.*, 2019a; Afifi *et al.*, 2019b; Afifi and Brown, 2020b; Afifi and Brown, 2019; Afifi *et al.*, 2020] or modeling the ideal WB process (DNN-based method) [Afifi and Brown, 2020a]. In detail, the exemplar-based methods combined the trained non-linear mappings to correct the sRGB inputs. Since these mappings are irreversible, the specificities of training cameras are mixed and applied to the white-balanced sRGB image encodings, making it biased towards the trained camera characteristics rather than the test one. Inevitably, the performances of exemplar-based methods are inferior in practical applications where various untrained cameras exist (Figure 1(b)). To solve this problem, Deep WB proposed to render the input back to its pseudo-raw features by the encoder of U-Net, then correct the pseudo-raw features and re-render them to the white-balanced sRGB images by the decoder of U-Net. It also allowed WB manipulation in sRGB images by setting two additional decoders to render the sRGB images with 2850 Kelvin(K) and 7500K. However, due to the lossy pooling operations, Deep WB is irreversible, making the content of the pseudo-raw features less complete than the ideal raw image and further restricting the output accuracy. Further, the joint training strategy suppressed the specificity of different cameras [Huang *et al.*, 2020], and hence together with the irreversible structure, limited the improvement of multi-camera generalization (Figure 1(c)). Our WBFlow belongs to the DNN-based method and achieves superior generalization to multiple cameras via reversible neural flows and few-shot learning.

3 White Balance for sRGB Image via Reversible Flows and Few-shot Learning

In this section, we introduce our WBFlow which aims to generalize well to multiple cameras. We start with formulating the ideal WB process for the sRGB image. Then, we introduce the details of our WBFlow and how it generalizes to multiple cameras via few-shot learning.

3.1 Ideal White Balance Process for sRGB Images

An sRGB image $\mathbf{I}^n_{ct}$ that is captured by camera n and rendered by color temperature ct can be formed as [Afifi *et al.*, 2019a]:

$$\mathbf{I}^n_{ct} = f_n(\mathbf{T}_n \mathbf{W}_{ct} \mathbf{I}^n_{raw}), \tag{1}$$

where $\mathbf{I}^n_{raw}$ is the raw image, $\mathbf{W}_{ct}$ is the WB module with color temperature ct, $\mathbf{T}_n$ is the color transformation matrix for camera n, and $f_n(\cdot)$ is the non-linear rendering function, including color enhancement, tone manipulation, and gamma encoding *et al.*. The correct sRGB image $\mathbf{I}^n_{wb}$ is obtained by manually setting the values of $\mathbf{W}_{ct}$ according to the true

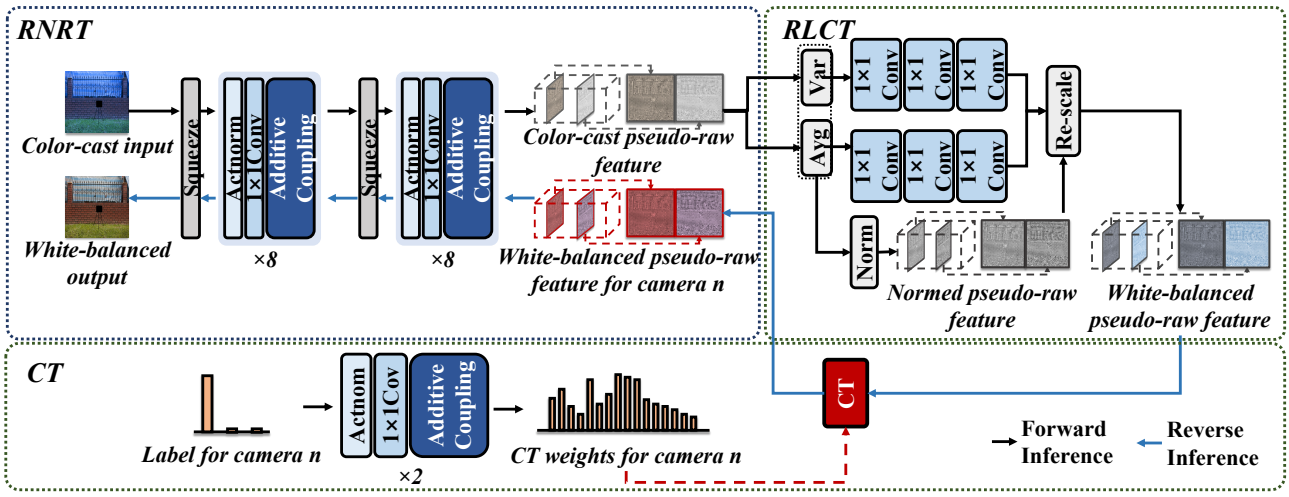

Figure 2: Illustration of our WBFlow. We propose the Reversible Non-linear Rendering Transformation and Reversible Linear Correction Transformation to ensure the reversibility of WBFlow, which significantly improves the floor of WB accuracy for multiple cameras. The Camera Transformation is then applied in the pseudo-raw space to generalize WBFlow to multiple cameras via few-shot learning.

scene illumination color. We denote it as $\mathbf{W}_{ideal}$ here. Therefore, the relationship between $\mathbf{I}^n_{wb}$ and $\mathbf{I}^n_{ct}$ can be modeled as:

$$\mathbf{I}^n_{wb} = f_n(\mathbf{T}_n \mathbf{W}_{ideal} \mathbf{W}^{-1}_{ct} \mathbf{T}^{-1}_n f^{-1}_n(\mathbf{I}^n_{ct})). \quad (2)$$

From Eq.2, the ideal WB process incorporates different kinds of reversible modules, which render the color-cast sRGB image toward its white-balanced version while preserving complete content. Specifically, the color-cast sRGB image $\mathbf{I}^n_{ct}$ is first reversed back to its linear values via $f^{-1}_n(\cdot)$; then, $\mathbf{T}_n$ transforms the linear values into the color cast raw image and $\mathbf{W}^{-1}_{ct}\mathbf{W}_{ideal}$ is applied to white balance it; after this, the correct sRGB image $\mathbf{I}^n_{wb}$ (GT) is generated via $f_n(\cdot)$ and $\mathbf{T}_n$.

As discussed in Sec.1 and 2, the existing sRGB-WB methods [Afifi *et al.*, 2019a; Afifi *et al.*, 2020; Afifi and Brown, 2020b; Afifi *et al.*, 2021; Afifi and Brown, 2020a] are *irreversible* and thus fail to meet the requirements of Eq.2, which limits their generalization to multiple cameras. In this paper, we propose an sRGB-WB network named WBFlow that satisfies the reversibility of Eq.2 and generalizes well to cameras via few-shot learning. Details will be introduced as follows.

3.2 White Balance for sRGB Images via Reversible Transformations

For achieving accurate sRGB-WB on multiple cameras, our primary goal is to model the ideal WB process appropriately. According to Eq.2, it has two types of modules: nonlinear color rendering function ($f_n(\cdot)$) and linear color transformations ($\mathbf{W}_{ct}$, $\mathbf{W}_{ideal}$, $\mathbf{T}_n$). Deep WB [Afifi and Brown, 2020a] models these two different modules via identical irreversible structures, which improves the multi-camera generalization only to a certain extent. To solve this problem, different from Deep WB, we model two types of modules by individual reversible functions:

$$\hat{\mathbf{I}}^n_{wb} = \mathcal{F}^{-1}(\mathcal{M}(\mathcal{F}(\mathbf{I}^n_{ct}))), \quad (3)$$

where $\mathcal{F}(\cdot)$ is the reversible non-linear function that renders the color-cast sRGB input $\mathbf{I}^n_{ct}$ back to the pseudo-raw feature losslessly, $\mathcal{M}(\cdot)$ is the reversible linear function that corrects the color cast of pseudo-raw feature and preserves its content, $\mathcal{F}^{-1}(\cdot)$ is the inverse of $\mathcal{F}(\cdot)$ that re-renders the white-balanced pseudo-raw feature to the corresponding sRGB image $\hat{\mathbf{I}}^n_{wb}$. Theoretically, due to $\mathcal{F}(\cdot)$ and $\mathcal{M}(\cdot)$, Eq.3 satisfies the reversibility of Eq.2, so that it can generate accurate white-balanced sRGB images for different cameras. Following are the implementation details.

Reversible Non-linear Rendering Transformation

As discussed, the reversible non-linear function $\mathcal{F}(\cdot)$ aims to produce the pseudo-raw feature with lossless content. Deep WB [Afifi and Brown, 2020a] cannot satisfy it due to its lossy pooling layers. Here, considering the color renderings included in $\mathcal{F}(\cdot)$ are independent [Ramanath *et al.*, 2005], we model them by a sequence of reversible bijective sub-functions: $\mathcal{F} = \mathcal{F}_1 \circ \mathcal{F}_2 \circ \cdots \circ \mathcal{F}_K$. This way, the relationship between the input $\mathbf{I}^n_{ct}$ and the intermediate feature $\mathbf{z}^n_{ct,k}$ is:

$$\mathbf{I}^n_{ct} \overset{\mathcal{F}_1}{\longleftrightarrow} \mathbf{z}^n_{ct,1} \overset{\mathcal{F}_2}{\longleftrightarrow} \mathbf{z}^n_{ct,2} \cdots \overset{\mathcal{F}_K}{\longleftrightarrow} \mathbf{z}^n_{ct,K}, \quad (4)$$

where $\mathbf{z}^n_{ct,K} = \mathbf{z}^n_{ct}$ is the pseudo-raw feature.

In practice, we adopt the additive coupling layer [Kingma and Dhariwal, 2018] to model $\{\mathcal{F}_k\}$. It is an efficient neural flow layer to simulate nonlinear reversible transformation [Kingma and Dhariwal, 2018; An *et al.*, 2021]. In our WBFlow, the forward computation of additive coupling is:

$$\begin{aligned} \mathbf{z}^{n,a}_{ct,k}, \mathbf{z}^{n,b}_{ct,k} &= Split(\mathbf{z}^n_{ct,k}), \\ \hat{\mathbf{z}}^{n,a}_{ct,k} &= Conv(\mathbf{z}^{n,a}_{ct,k}) + \mathbf{z}^{n,b}_{ct,k}, \\ \mathbf{z}^n_{ct,k+1} &= Concat(\hat{\mathbf{z}}^{n,a}_{ct,k}, \mathbf{z}^{n,b}_{ct,k}), \end{aligned} \quad (5)$$

where $Split(\cdot)$ splits the intermediate feature $\mathbf{z}^n_{ct,k}$ into two parts, $\mathbf{z}^{n,a}_{ct,k}$ and $\mathbf{z}^{n,b}_{ct,k}$, along the channel dimension. $Conv(\cdot)$ renders the colors of $\mathbf{z}^{n,a}_{ct,k}$ with 3×3 convolutions and relu functions. $Concat(\cdot)$ performs the reverse operation

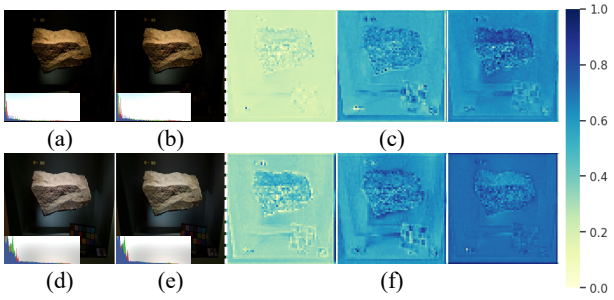

Figure 3: The pseudo-raw feature with lossless content in (c) is obtained from the color-cast input in (a) via the forward RNRT. The input in (b) is restored from the pseudo-raw feature in (c) via the reverse propagation of the RNRT. The pseudo-raw feature is corrected linearly in (f) by the RLCT and re-rendered back to the white-balanced sRGB image in (e) via the reverse RNRT.

of $Split(\cdot)$ that concatenates the rendered $\hat{\mathbf{z}}^{n,a}_{ct,k}$ and $\mathbf{z}^{n,b}_{ct,k}$ to form the rendered feature $\mathbf{z}^{n}_{ct,k+1}$. The reverse computation of additive coupling is easily derived from Eq.5. We can superimpose additive coupling layers to establish the reversible mapping between the input color-cast sRGB image and the pseudo-raw feature as Eq.4. However, as in Eq.5, the additive coupling leave some channels of the intermediate feature unchanged due to $Split(\cdot)$. To solve this problem, following [Kingma and Dhariwal, 2018], we use the reversible 1×1 convolution that integrates the information of all channels by fixing the channel numbers of input and output consistently. We also adopt the Actnorm [Kingma and Dhariwal, 2018] to normalize features with reversible affine function, which is an alternative to BatchNorm and accelerates the training times.

Our proposed RNRT is in Figure 2, consisting of the additive coupling layers, reversible 1×1 convolutions, and Actnorms. All the components of RNRT are reversible, making the information lossless during the forward and reverse propagations. As in Figure 3(c), we can obtain the pseudo-raw feature with lossless content from the color-cast input in Figure 3(a) via the forward RNRT. Simultaneously, in Figure 3(b), this input can be restored accurately from the pseudo-raw feature in Figure 3(c) via the reverse RNRT.

Reversible Linear Correction Transformation

In Eq.2 and Eq.3, the reversible linear function $\mathcal{M}(\cdot)$ is linear and reversible due to the white balance and color transformation matrices. To implement it appropriately while preserving lossless content, inspired by [Huang and Belongie, 2017], we propose the RLCT that segregates the color information from the pseudo-raw feature to correct it by reversible 1×1 convolutions [Kingma and Dhariwal, 2018]. The formation is:

$$\mathbf{z}^{n}_{wb} = \mathcal{T}_{\sigma}(\sigma_z)(\frac{\mathbf{z}^{n}_{ct} - \mu_z}{\sigma_z}) + \mathcal{T}_{\mu}(\mu_z), \tag{6}$$

where the channel-wise mean and variance, μ_z and σ_z, represents the color information of pseudo-raw feature $\mathbf{z}^{n}_{ct,k}$ [Stricker and Orengo, 1995]. They also help to keep the content information by normalizing $\mathbf{z}^{n}_{ct}$, *i.e.*, $(\mathbf{z}^{n}_{ct} - \mu_z)/\sigma_z$. For individual color information, we correct it by forwarding μ_z and σ_z into linear mappings $\mathcal{T}_{\sigma}(\cdot)$ and $\mathcal{T}_{\mu}(\cdot)$, which consist

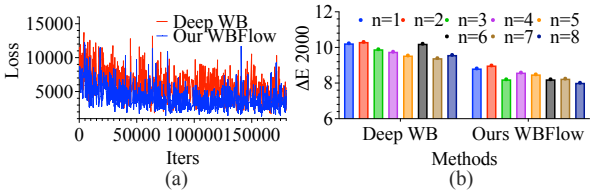

Figure 4: Comparison between Deep WB and our WBFlow: (a) multi-camera training loss and (b) generalizations on the new camera (PanasonicGX1) (n is the number of training cameras).

of three reversible 1×1 convolutions [Kingma and Dhariwal, 2018]. This way, as in Figure 3(f), the pseudo-raw feature is white-balanced by re-scaling the lossless content with learned channel-wise mean $\mathcal{T}_{\sigma}(\sigma_z)$ and variance $\mathcal{T}_{\mu}(\mu_z)$. Further, in Figure 3(e), the white-balanced sRGB image is successfully generated from the white-balanced pseudo-raw feature in Figure 3(f) via the reverse propagation of RNRT. It has an almost identical color histogram as the GT in Figure 3(d).

As introduced above, the RNRT and RLCT are reversible and thus cooperate to ensure complete reversibility, which satisfies the requirement of Eq.2. Thus, our WBFlow can significantly improve the floor of WB accuracy for multiple cameras. As in Figure 4(a), with the same multi-camera training strategy of Deep WB, our WBFlow converges on the much lower l_1 losses for multiple cameras than it.

3.3 Camera Generalization via Few-shot Learning

According to Eq.2 and Eq.3, the ideal pseudo-raw features should be specific to various cameras. However, the existing methods suppressed such specificities due to the combined trained mappings [Afifi *et al.*, 2019a; Afifi and Brown, 2020b] or joint multi-camera training strategy [Afifi and Brown, 2020a]. Inevitably, this will result in a limited multi-camera generalization. To enhance the camera specificity of the pseudo-raw feature, we introduce few-shot learning into WBFlow, which is based on the fact that inter-camera transformation can be achieved by learning the mapping between raw color values from two cameras [Gao *et al.*, 2017]. In Figure 2, we set the camera transformation named CT after the RLCT. For camera n, Eq.6 can be re-written as:

$$\hat{\mathbf{z}}^{n}_{wb} = \mathbf{w}^{n}_{cam} * \left\{ \mathcal{T}_{\sigma}(\sigma_z)(\frac{\mathbf{z}^{n}_{ct} - \mu_z}{\sigma_z}) + \mathcal{T}_{\mu}(\mu_z) \right\}, \tag{7}$$

where $*$ is the group convolution operator, $\mathbf{w}^{n}_{cam}$ is a set of weights learned from the camera labels by two reversible neural flow combinations. This strategy ensures that the learned weights are unique to the different cameras. To adapt the WBFlow to camera n, we fix the pre-trained parameter of the RNRT, then train the new weights in Eq.7 by minimizing the l_1 loss with a few samples. This way, the specificity for camera n is enhanced in the pseudo-raw feature to improve the generalization. In Figure 4(b), our WBFlow's $\Delta E2000$ on the new camera, PanasonicGX1, are always lower than that of Deep WB, regardless of the number of training cameras.

Methods	$\Delta E2000 \downarrow$				Mean Angle Error (MAE) $\downarrow$				Inference Time (s) $\downarrow$
	Mean	Q1	Q2	Q3	Mean (°)	Q1 (°)	Q2 (°)	Q3 (°)	
Set1–Test (trained cameras)									
KNN-WB[Afifi *et al.*, 2019a]	3.58	2.07	3.09	4.55	3.06	1.74	2.54	3.76	**0.78**
MixedWB[Afifi *et al.*, 2021]	4.55	3.00	4.15	5.63	4.07	2.64	3.68	5.16	1.43
Deep WB[Afifi and Brown, 2020a]	3.77	2.16	3.30	4.86	3.12	1.88	2.70	3.84	1.10
Our WBFlow	**3.13**	**1.92**	**2.79**	**3.94**	**2.67**	**1.73**	**2.39**	**3.24**	1.09
Set2 (untrained cameras)									
KNN-WB[Afifi *et al.*, 2019a]	5.60	3.43	4.90	7.06	4.48	2.26	3.64	5.95	**0.82**
MixedWB[Afifi *et al.*, 2021]	6.05	3.45	4.92	7.20	4.92	2.69	4.10	6.37	1.55
Deep WB[Afifi and Brown, 2020a]	4.90	**3.13**	4.35	6.08	3.75	2.02	3.08	4.72	1.09
Our WBFlow	**4.64**	3.16	**4.07**	**5.56**	**3.51**	**1.93**	**2.92**	**4.47**	1.08
Rendered Cube Dataset (untrained cameras)									
KNN-WB[Afifi *et al.*, 2019a]	5.68	3.22	4.61	6.70	4.12	1.96	3.17	5.04	**0.81**
MixedWB[Afifi *et al.*, 2021]	5.03	2.07	3.12	7.19	4.20	1.39	2.18	5.54	1.52
Deep WB[Afifi and Brown, 2020a]	4.59	**2.68**	3.81	5.53	3.45	**1.87**	2.82	4.26	1.08
Our WBFlow	**4.28**	2.71	**3.77**	**5.21**	**3.34**	1.94	**2.82**	**4.11**	1.07

Table 1: Quantitative results of our WBFlow and state-of-the-art sRGB-WB methods on three public datasets. The top results are in bold.

4 Experiments

4.1 Datasets

Public Datasets. Following [Afifi and Brown, 2020a], we randomly select 12000 sRGB images from the first fold of Set1 [Afifi *et al.*, 2019a; Cheng *et al.*, 2014] to train our WBFlow. We use the remaining two folds (Set1-Test), Set2 [Afifi *et al.*, 2019a] and rendered cube dataset [Afifi *et al.*, 2019a; Banić *et al.*, 2017] for testing.

Rendered Multi-camera sRGB Dataset. We collected a multi-camera sRGB dataset to evaluate the multi-camera generalization effect. Specifically, we selected and compiled 184 groups of raw images from the NUS dataset [Cheng *et al.*, 2014]. In each group, the raw images are consistent in the scenes and differ in the cameras: Canon1DsMkIII, Canon600D, FujifilmXM1, NikonD5200, OlympusEPL6, PanasonicGX1, SamsungNX2000, and SonyA57. To obtain the color-cast sRGB versions of these images, following [Afifi *et al.*, 2019a], we use the Adobe Camera Raw in Photoshop to render them with five common color temperatures (2850 K, 3800 K, 5500 K, 6500 K, and 7500 K) and camera standard photo finishing. We obtain the corresponding GTs by manually selecting the correct color temperature from the middle gray patches in the color checker of each raw image. The rest of the operations remain unchanged. In total, our multi-camera sRGB dataset contains 7360 sRGB images with 184 scenes, five color temperatures, and eight cameras.

4.2 Implementation Details

Loss Function. Following [Afifi and Brown, 2020a], we apply l_1 loss to train WBFlow:

$$\arg\min_{\mathcal{F},\mathcal{M}} \sum_{ct} \sum_{n} \left\| \mathcal{F}^{-1}(\mathcal{M}(\mathcal{F}(\mathbf{I}^n_{ct}))) - \mathbf{I}^n_{wb} \right\|_1 . \quad (8)$$

Training and Testing Detail. We implemented the WBFlow on Pytorch with CUDA support and used the Adam [Kingma and Ba, 2014] with $\beta_1 = 0.9$ and learning rate 10^{-4} to optimize it. For the experiments with all training images, we trained our WBFlow for 340000 iterations with batch size 4. While for few-shot experiments, we train the CT for 15000 iterations. We applied color jittering, average/median blur, geometric rotation, and flipping as data augmentations. During testing, for a fair comparison, following [Afifi and Brown, 2020a], we resized all input images to a maximum dimension of 656 pixels and set a color mapping procedure to compute the final white-balanced sRGB images.

Error Metric. We used the same error metrics as the existing sRGB-WB methods [Afifi and Brown, 2020a; Afifi *et al.*, 2019a]: mean angle error (MAE) and $\Delta E2000$ [Sharma *et al.*, 2005]. We reported the mean, first quantile (Q1), second quantile (Q2), and third quantile (Q3) for evaluation. A lower error metric denotes a better sRGB-WB performance.

4.3 Multi-camera Generalization Evaluation

Table 1 shows the quantitative results of our WBFlow and state-of-the-art sRGB-WB methods on Set1-Test (trained cameras), Set2 (untrained cameras) and rendered cube dataset (untrained cameras). The results of MixedWB on Set1-Test and Set2 are computed by its code, and others are collected from [Afifi *et al.*, 2021] and [Afifi and Brown, 2020a].

Quantitate Comparison. From Table 1, due to the irreversible and biased mapping, the exemplar-based methods, KNN-WB and MixedWB, generalized badly to the untrained cameras. For example, their mean values of $\Delta E2000$ on Set2 and rendered cube dataset are 56.42% and 36.97% higher than those in Set1-Test. Deep WB alleviated this by modeling the ideal WB process by U-Net. However, since U-Net is irreversible, Deep WB performed inferiorly on the trained cameras. That is, although Deep WB reduced the performance of KNN-WB in the mean of $\Delta E2000$ about 12.50% and 19.19% in Set2 and rendered cube dataset, its mean of $\Delta E2000$ is 5.31% worse than KNN-WB in Set1-test. In contrast, in Set1-test (trained cameras), our WBFlow outperformed the KNN-WB (ranking second) and the Deep WB

Figure 5: Qualitative comparison (sRGB images and their color histograms) for multi-camera generalization.

Methods	OlympusEPL6 (△E2000↓)				PanasonicGX1 (△E2000↓)				SamsungNX2000 (△E2000↓)				SonyA57 (△E2000↓)			
	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3	Mean	Q1	Q2	Q3
Deep WB	7.12	5.43	6.94	8.46	6.69	5.50	6.48	7.69	6.87	5.49	6.71	7.76	6.53	4.99	6.28	7.87
WBFlow K=0	6.19	4.39	5.73	7.72	5.72	4.23	5.55	6.65	6.04	4.30	5.30	7.03	5.51	3.82	5.38	6.83
WBFlow K=5	6.13	4.33	5.68	7.75	5.65	4.12	5.24	6.90	6.02	4.23	5.26	7.37	5.50	3.79	5.12	6.76
WBFlow K=10	6.00	4.37	5.37	7.18	5.62	4.10	5.21	6.93	6.00	4.23	5.26	7.34	5.48	3.72	5.10	6.81
WBFlow K=20	5.97	4.25	5.39	7.36	5.61	4.08	5.08	6.92	5.95	4.13	5.16	6.97	5.45	3.70	5.07	6.62

Table 2: Few shot Evaluation of our WBFlow on the rendered multi-camera sRGB dataset (Deep WB:[Afifi and Brown, 2020a]).

(most related method) by about 12.57% and 16.98% in the Mean of $\Delta E2000$, respectively. Simultaneously, in Set2 and rendered cube dataset (untrained datasets), our WBFlow still outperformed all competing methods in most metrics, *e.g.*, our WBFlow greatly improved the accuracy of Deep WB in the mean of $\Delta E2000$ and MAE about 5.31% and 6.40%. Similar results appear in Set2. These results verify the effectiveness of our WBFlow in multi-camera generalization.

Inference Time Comparison. From Table 1, our WBFlow achieves superior WB performance on all datasets with almost the same inference time as Deep WB (the most related method). Compared with KNN-WB with the fastest inference time, our WBFlow outperforms it by about 12.57%, 17.14% and 24.65% for the mean of $\Delta E2000$ in three datasets.

Qualitative Comparison. To qualitatively compare the multi-camera generalization, we show the generated white-balanced sRGB images and their color histograms of KNN-WB, Deep WB, and WBFlow for eight cameras in Figure 5. From it, our WBFlow achieves the most similar colors as GTs for all cameras, consistent with the quantitate comparison.

4.4 Few Shot Evaluation

To validate the few-shot capacity, we compared the performances of WBFlow with the most related method, Deep WB[Afifi and Brown, 2020a], on four untrained cameras. We retrained Deep WB and our WBFlow by randomly selecting 5000 sRGB images from the first fold of Set1, and used the four untrained cameras in the last 84 groups of our rendered multi-camera sRGB dataset for evaluation: OlympusEPL6, PanasonicGX1, SamsungNX2000, and SonyA57. Training details are followed by [Afifi and Brown, 2020a]. For few-shot experiments, we randomly selected K images (0, 5, 10, 20) for the untrained cameras from the first 100 groups of our rendered multi-camera sRGB dataset to train the parameters

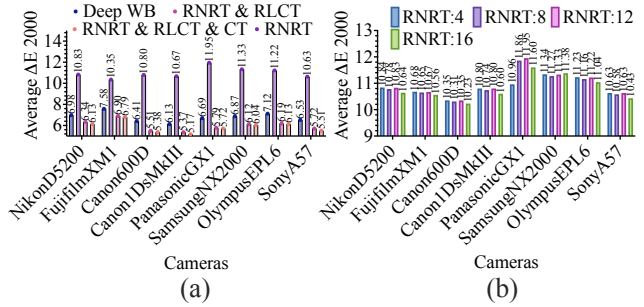

Figure 6: Ablation comparisons of our WBFlow on the sRGB images from eight cameras (fixed scenes and color temperatures): (a) different variants of the WBFlow and (b) flow numbers of the RNRT.

of the RLCT and CT, which are 5.82% of our WBFlow's. To avoid randomness and disturbance, we repeated the experiments 1000 times with randomly selected images and then computed the average of $\Delta E2000$ in Table 2.

From Table 2, due to the reversible structure, our WBFlow significantly outperforms Deep WB even with no shots (K=0). For example, the average $\Delta E2000$ values of our WBFlow are smaller than Deep WB's by about 17.44%, 23.09%, 21.68%, and 23.45% in Q2 for four untrained cameras. This superiority is enhanced when we learn the RLCT and CT with K=5, 10, 20 to enhance the camera specificity. Especially, the few-shot learning effectively improves the Q3 of average $\Delta E2000$ values for four untrained cameras. This indicates that our WBFlow with few-shot learning is superior in handling difficult scenes from untrained cameras.

4.5 Ablation Analysis

To verify the effectiveness of each part in the WBFlow, we conducted an ablation analysis on our rendered multi-camera sRGB dataset. Since Deep WB is the most related method, we report the average $\Delta E2000$ of Deep WB and three variants of our WBFlow in Figure 6(a). The training and testing settings are the same as the few-shot experiments. Further, we compared the influences of flow numbers for the RNRT in Figure 6(b) and the camera specificity verification for the CT in Figure 7. The separated color and content visualization for the RLCT is in supplementary materials.

Variants Comparison. From Figure 6(a), the average $\Delta E2000$ values of RNRT for eight cameras are much worse than Deep WB, which shows that only modeling the color renderings hardly improves the multi-cameras generalization. This phenomenon is significantly mitigated when we integrate RNRT and RLCT. Specifically, compared with only RNRT, RNRT&RLCT work together to improve the average $\Delta E2000$ values by about 50% for eight cameras. RNRT&RLCT also considerably outperforms Deep WB by 13% for eight cameras. This is because both the RNRT and RLCT are reversible to model the ideal WB process effectively. When we add the CT, the multi-camera generalization is further improved, as in the smallest average $\Delta E2000$ of RNRT&RLCT&CT for all cameras.

Flows Numbers Influence for RNRT. To explore the optimal flow numbers of RNRT, in Figure 6(b), we compared the

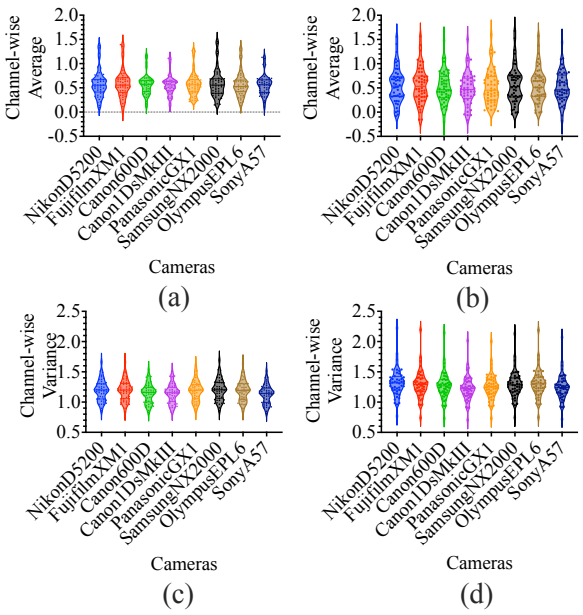

Figure 7: The distributions of the average and variance values of the white-balanced pseudo-raw features for sRGB images from eight cameras when WBFlow w/ and w/o CT: (a) and (c) are the distributions when WBFlow w/o CT, (b) and (d) are the distributions when WBFlow w/ CT.

average $\Delta E2000$ of the RNRT with 4, 8, 12, and 16 flows on eight cameras. As seen, when the flow number increases, the complexity of RNRT becomes larger to better model the ISP nonlinear renderings, so that the average $\Delta E2000$ of RNRT is minimized at a number of 16.

Camera Specificity Verification. Since the color information of the pseudo-raw feature can be represented by the channel-wise average and variance [Stricker and Orengo, 1995], we compute them for the white-balanced pseudo-raw features without and with the CT for eight cameras to verify the camera specificity in Figure 7. As seen, with the CT, the difference between the mean/variance for all channels of the white balance pseudo-raw feature becomes significantly larger for eight cameras. This verifies that our CT can significantly enhance the camera specificity for the pseudo-raw feature, thus improving the multi-camera generalization.

5 Conclusion

In this paper, we propose an sRGB-WB network named WBFlow, which not only performs superior white balance for sRGB images but also generalizes to multiple cameras well. Unlike the existing methods that are irreversible and fail to model the ideal WB process, WBFlow models it successfully by the reversible RNRT and RLCT. Furthermore, we generalize the WBFlow to multiple cameras by enhancing the camera characteristic of the pseudo-raw features via few-shot learning. Extensive experiments indicate the superiority of our WBFlow in multi-camera generalization and few-shot sRGB-WB task. The ablation analysis shows the effectiveness of each part of WBFlow and their optimal combinations.

Acknowledgements

This work was supported by the national key R & D program intergovernmental international science and technology innovation cooperation project (2021YFE0101600).

References

[Afifi and Brown, 2019] Mahmoud Afifi and Michael S Brown. What else can fool deep learning? addressing color constancy errors on deep neural network performance. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 243–252, 2019.

[Afifi and Brown, 2020a] Mahmoud Afifi and Michael S Brown. Deep white-balance editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1397–1406, 2020.

[Afifi and Brown, 2020b] Mahmoud Afifi and Michael S Brown. Interactive white balancing for camera-rendered images. *arXiv preprint arXiv:2009.12632*, 2020.

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

[Afifi *et al.*, 2019b] 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*, volume 2019, pages 1–6. Society for Imaging Science and Technology, 2019.

[Afifi *et al.*, 2020] Mahmoud Afifi, Abdelrahman Abdelhamed, Abdullah Abuolaim, Abhijith Punnappurath, and Michael S Brown. Cie xyz net: Unprocessing images for low-level computer vision tasks. *arXiv preprint arXiv:2006.12709*, 2020.

[Afifi *et al.*, 2021] Mahmoud Afifi, Marcus A Brubaker, and Michael S Brown. Auto white-balance correction for mixed-illuminant scenes. *arXiv preprint arXiv:2109.08750*, 2021.

[Afifi, 2021] Mahmoud Afifi. Image color correction, enhancement, and editing. *arXiv preprint arXiv:2107.13117*, 2021.

[An *et al.*, 2021] Jie An, Siyu Huang, Yibing Song, Dejing Dou, Wei Liu, and Jiebo Luo. Artflow: Unbiased image style transfer via reversible neural flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 862–871, 2021.

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

[Chakrabarti *et al.*, 2014] Ayan Chakrabarti, Ying Xiong, Baochen Sun, Trevor Darrell, Daniel Scharstein, Todd Zickler, and Kate Saenko. Modeling radiometric uncertainty for vision with tone-mapped color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2185–2198, 2014.

[Cheng *et al.*, 2014] 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. *JOSA A*, 31(5):1049–1058, 2014.

[Foster, 2011] David H Foster. Color constancy. *Vision research*, 51(7):674–700, 2011.

[Gao *et al.*, 2017] Shao-Bing Gao, Ming Zhang, Chao-Yi Li, and Yong-Jie Li. Improving color constancy by discounting the variation of camera spectral sensitivity. *JOSA A*, 34(8):1448–1462, 2017.

[Gijsenij and Gevers, 2007] Arjan Gijsenij and Theo Gevers. Color constancy using natural image statistics. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[Hong *et al.*, 2001] Guowei Hong, M Ronnier Luo, and Peter A Rhodes. A study of digital camera colorimetric characterization based on polynomial modeling. *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, 26(1):76–84, 2001.

[Hu *et al.*, 2017] Yuanming Hu, Baoyuan Wang, and Stephen Lin. Fc4: Fully convolutional color constancy with confidence-weighted pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4085–4094, 2017.

[Huang and Belongie, 2017] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.

[Huang *et al.*, 2020] Xinwei Huang, Bing Li, Shuai Li, Wenjuan Li, Weihua Xiong, Xuanwu Yin, Weiming Hu, and Hong Qin. Multi-cue semi-supervised color constancy with limited training samples. *IEEE Transactions on Image Processing*, 29:7875–7888, 2020.

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

[Kingma and Dhariwal, 2018] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

[Lo *et al.*, 2021] 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. *arXiv preprint arXiv:2106.04989*, 2021.

[Ramanath *et al.*, 2005] Rajeev Ramanath, Wesley E Snyder, Youngjun Yoo, and Mark S Drew. Color image processing pipeline. *IEEE Signal Processing Magazine*, 22(1):34–43, 2005.

[Ronneberger *et al.*, 2015] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks

for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[Sharma *et al.*, 2005] 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.

[Song *et al.*, 2021] Zeyu Song, Dongliang Chang, Zhanyu Ma, Xiaoxu Li, and Zheng-Hua Tan. Cc-loss: Channel correlation loss for image classification. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 7601–7608. IEEE, 2021.

[Stricker and Orengo, 1995] Markus Andreas Stricker and Markus Orengo. Similarity of color images. In *Storage and retrieval for image and video databases III*, volume 2420, pages 381–392. SPiE, 1995.

[Tang *et al.*, 2022] Yuxiang Tang, Xuejing Kang, Chunxiao Li, Zhaowen Lin, and Anlong Ming. Transfer learning for color constancy via statistic perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2361–2369, 2022.

[Xu *et al.*, 2021] Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A fourier-based framework for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14383–14392, 2021.

[Zhang *et al.*, 2022] Zhifeng Zhang, Xuejing Kang, and Anlong Ming. Domain adversarial learning for color constancy. In Lud De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, pages 1693–1699. International Joint Conferences on Artificial Intelligence Organization, 7 2022. Main Track.