Semi-Supervised Raw-to-Raw Mapping

Mahmoud Afifi[1,2]
https://www.eecs.yorku.ca/~mafifi/
Abdullah Abuolaim[1,3]
https://www.eecs.yorku.ca/~abuolaim/

[1] York University, Canada
[2] Assiut University, Egypt
[3] Jordan University of Science and Technology, Jordan

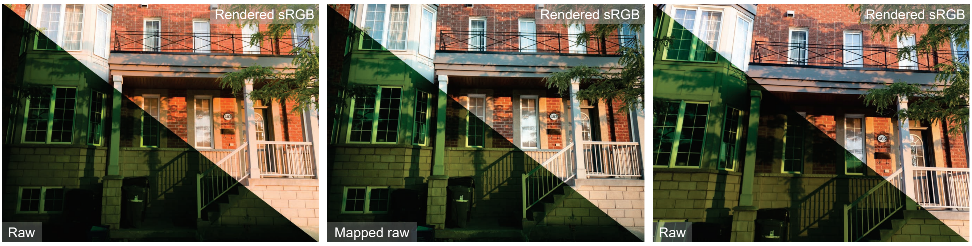


Image taken by iPhone X camera

Our mapping to Samsung S9 camera raw space

Reference image taken by Samsung S9 camera

Figure 1: We introduce a semi-supervised raw-to-raw mapping method. This figure shows a raw image captured by iPhone X smartphone camera and our mapping result to Samsung Galaxy S9 smartphone camera raw space, along with a reference image capturing the same scene by Samsung Galaxy S9. For each image, we show both raw and the camera-ISP rendered image. Shown images are from our dataset. Note that the original Bayer raw image is packed to `RGGB` channels. To aid visualization, the Green channels are averaged (i.e., three-channel `RGB` image), and a Gamma operation with $1/1.6$ encoding gamma is applied. This is applicable for raw visualization in the rest of this paper.

Abstract

The raw-RGB colors of a camera sensor vary due to the spectral sensitivity differences across different sensor makes and models. This paper focuses on the task of mapping between different sensor raw-RGB color spaces. Prior work addressed this problem using a *pairwise* calibration to achieve accurate color mapping. Although being accurate, this approach is less practical as it requires: (1) capturing pair of images by both camera devices with a color calibration object placed in each new scene; (2) accurate image alignment or manual annotation of the color calibration object. This paper aims to tackle color mapping in the raw space through a more practical setup. Specifically, we present a semi-supervised raw-to-raw mapping method trained on a small set of paired images alongside an unpaired set of images captured by each camera device. Through extensive experiments, we show that our method achieves better results compared to other domain adaptation alternatives in addition to the single-calibration solution. We have generated a new dataset of raw images from two different smartphone cameras as part of this effort. Our dataset includes unpaired and paired sets for our semi-supervised training and evaluation.

1 Introduction and Related Work

A camera image signal processor (ISP) applies a set of operations to render captured images from the camera's internal raw space into a standard display color space (e.g., standard RGB, or sRGB for short). While colors in the sRGB space can be different across cameras, it is also possible to observe color differences in the camera raw space when capturing the same scene by different camera devices. The reason behinds these color differences can be understood from the raw image formation. Mathematically, the construction of a raw image, $I = \{I_r, I_g, I_b\}$, can be described as follows [6]:

$$I_c(x) = \int_{\gamma} \rho(x,\lambda) R(x,\lambda) S_c(\lambda) d\lambda, \tag{1}$$

where $c = \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$, x refer to pixel location in I, and γ and $\rho(\cdot)$ are the visible light spectrum and the spectral power distribution of the scene illumination, respectively. The captured object spectral reflectance properties are denoted by $R(\cdot)$, and $S(\cdot)$ refers to the camera sensor sensitivity at wavelength λ. Note that we omit image noise in Eq. 1 for simplicity, as our focus in this paper is mainly related to raw colors.

From Eq. 1, it is clear that if camera A and camera B capture the same scene to produce images I_A and I_B (assuming they are perfectly aligned), the final colors in I_A and I_B would be similar *iff* the camera sensor sensitivities, S, of camera A and B are the same. Typically, this case rarely happens, especially when different vendors manufactured the camera sensors. As a result, I_A and I_B are likely to have different colors even when capturing the same scene under the same lighting conditions [14]. It is also clear that differences in colors produced by cameras A and B may have different levels of variations across scenes/lighting conditions due to the integral of $\rho(\cdot)R(\cdot)S_c(\cdot)$ on the visible light spectrum, γ, in Eq. 1.

Raw-to-raw mapping aims to reduce color differences in I_A and I_B, and it is useful for camera ISP manufacturing as elaborated next. First, raw-to-raw mapping is useful for data generation in any camera ISP learning-based module that utilizes raw images. For instance, prior work [3, 4, 11, 19] showed that color mapping for data augmentation purposes could improve the accuracy of color constancy in raw images.

Second, camera ISPs include different carefully calibrated modules inherently tied to the camera sensor space used in designing such modules. When a camera manufacturer introduces a new sensor with a different spectral sensitivity, these tuned camera ISP modules should be adapted to the spectral sensitivity of this new sensor [13, 17, 23]. Needless to say, this adaptation process often requires collecting new labeled data with some corresponding ground-truth annotations. This process is tedious, and, as a result, deploying a new sensor in a camera device is still challenging and requires a lot of human effort.

To avoid generating new labeled data when employing a new sensor, recent work in [18] proposes to map color histogram of new sensor raw images to the original sensor space used to train the illumination estimation camera ISP module. Then, the estimated illumination color is projected back to the new sensor space. Building on top of the idea in [18], one could either: (1) map labeled training ISP images from the old sensor space to the new sensor space, and thus the retraining is practicable for all learning-based ISP modules; or (2) design a "universal camera ISP" by mapping all images to this specific sensor space. Fig. 1 shows an example the "universal camera ISP" idea, where we render an iPhone raw image – after mapping to the target sensor using our method – through an ISP designed for rendering Samsung raw images.

Despite its importance, the majority of prior work focuses on colorimetric calibration[1] (e.g., [5, 9, 10, 12]). There is a lack of prior work related to raw-to-raw mapping. Nguyen et al., [20], to the best of our knowledge, proposed the first attempt for raw-to-raw mapping and showed that classical color mapping approaches, originally proposed for colorimetric calibration, can be used for raw-to-raw mapping. Specifically, Nguyen et al., [20] proposed to compute a pairwise raw-to-raw calibration to map image I_A to image I_B, achieving very promising results. This mapping can be expressed as follows:

$$\hat{I}_B = g\left(M \quad \phi\left(r\left(I_A\right)\right)\right), \tag{2}$$

where $r(\cdot)$ and $g(\cdot)$ are reshaping functions that represent images as $3 \times n$ and $h \times w \times 3$, respectively. $n = hw$ is the total number of pixels in each image, M is a color mapping matrix, and $\phi(\cdot)$ is a kernel function. To compute M, Nguyen et al. [20] used color calibration charts captured in each scene and computed a scene-specific mapping matrix. The work in [20] studied different ways to compute this mapping, including polynomial and identity kernel functions. Despite being accurate, the applicability of such pairwise calibration methods in real scenarios is limited as it requires capturing and annotating a calibration object for each new scene. Moreover, the color chart used for mapping has a limited number of color samples (i.e., the typical 24-color checker chart).

Contribution This paper discusses a practical raw-to-raw mapping with an easy setup to use. In particular, we propose a semi-supervised training to learn, from a very small set of paired images, a reasonable raw-to-raw mapping that does not require computing per-scene calibration (see Fig. 1). In addition to this small paired set, we exploit another set of unpaired images (i.e., requires minimal capturing effort with no annotation) captured by each camera device to improve our mapping. As collecting such unpaired set is abundantly available, we believe that our method is the first to propose a *practical setup* for this problem. Through extensive evaluation, we show that our method achieves better results compared to other domain adaptation alternatives (e.g., [21, 26, 27]). To enable training and evaluation of our new approach, we have collected a new dataset of raw images captured by two different smartphone cameras, i.e., iPhone X and Samsung Galaxy S9.

2 Methodology

The overview of our method is shown in Fig. 2. As illustrated, we propose to map raw images from camera A to camera B through a deep learning framework. Our framework includes two encoder-decoder networks, each of which is dedicated to one of our cameras. We use two different training sets: (i) a small set of paired images, the so-called "anchor set" and (ii) a larger set of unpaired images taken by cameras A and B.

At each iteration, we optimize both networks using data taken from each of these two sets. We penalize our network on the standard encoder-decoder reconstruction task when training data is from the unpaired set. In contrast, when training data is from the anchor set, we apply a latent normalization step to encourage the encoder of both networks to produce similar latent representations for each pair of images taken by our two different cameras, A and B. The details of our loss functions are elaborated in Sec. 2.2. At the inference phase,

[1] Colorimetric calibration maps camera colors to corresponding device-independent tristimulus values in some canonical space (typically the CIE XYZ space).

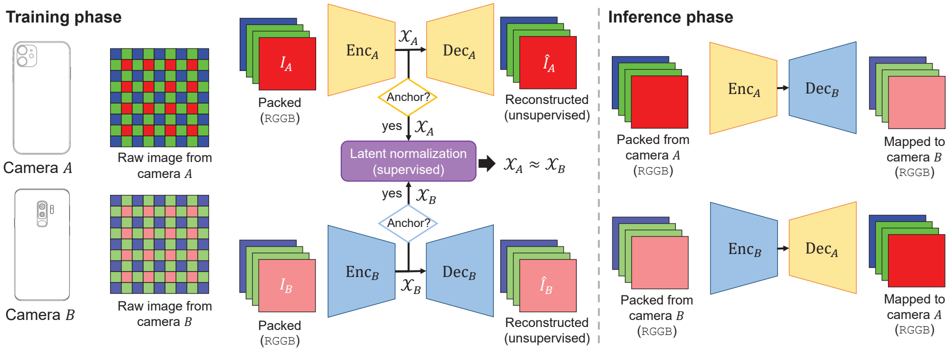

Training phase
Camera A
Raw image from camera A
Camera B
Raw image from camera B
I_A
Packed (RGGB)
Enc$_A$
$\mathcal{X}_A$
Dec$_A$
$\hat{I}_A$
Reconstructed (unsupervised)
Anchor?
yes $\mathcal{X}_A$
Latent normalization (supervised)
$\mathcal{X}_A \approx \mathcal{X}_B$
yes $\mathcal{X}_B$
Anchor?
I_B
Packed (RGGB)
Enc$_B$
$\mathcal{X}_B$
Dec$_B$
$\hat{I}_B$
Reconstructed (unsupervised)
Inference phase
Packed from camera A (RGGB)
Enc$_A$
Dec$_B$
Mapped to camera B (RGGB)
Packed from camera B (RGGB)
Enc$_B$
Dec$_A$
Mapped to camera A (RGGB)

Figure 2: Our semi-supervised raw-to-raw mapping. In training, we use a set of unpaired images taken by A and B cameras, alongside a small set of paired images (so-called "*anchor set*"). We train two encoder-decoder networks to learn reconstructing the unpaired images from each camera. Additionally, we use our small anchor set to encourage the encoder net in both networks to produce similar latent features when input images share the same scene. At inference, we swap the decoder nets to map between raw spaces.

we swap our decoders to map images taken by camera A to the raw space of camera B and vice versa.

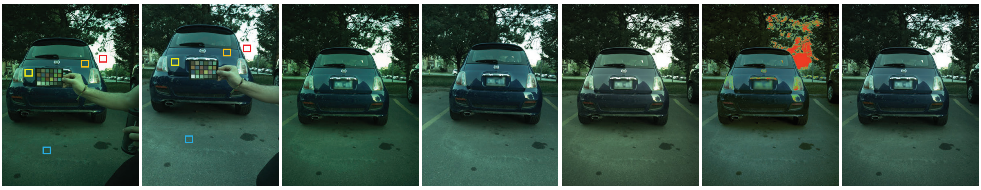
We use U-Net-like design [22] for each network that includes skip connections from different encoder blocks to the corresponding decoder blocks to improve the reconstruction accuracy. We used the same number of channels used in [2] for our networks, as well as the inference post-processing used in [2].

2.1 Anchor Set

The first step necessary for our training is to generate the anchor set. This paired anchor set must include perfectly aligned images taken by both cameras A and B. This is hard to achieve due to the drastic difference in optics and field of view between the two cameras. Even with careful capturing settings (e.g., [13]), perfect alignment is not guaranteed. Due to this reason, and as our interest is in the color mapping (neither to map image noise characteristics nor quality), we adopted the standard pairwise calibration approach proposed in [20] to generate our paired set.

Specifically, we capture paired unaligned images by our two cameras with a color calibration chart in each scene. Then, we manually select corresponding colors from the color chart patches taken by each camera and use these colors to compute a polynomial mapping matrix, M, to map from camera A to camera B. Afterward, we apply the computed matrix M to all pixels produced by camera A using Eq. 2 to get aligned corresponding pixels in the raw space of camera B. Similarly, we map images by camera B to the raw space of camera A.

When computing M, we found that relying solely on the colors of chart patches does not always produce accurate mapping and may result in noticeable out-of-gamut mapping. This is due to the fact that, in some cases, the polynomial transformations tend to overfit to colors in the calibration chart and produces inaccurate mapping for other colors. To fix this, we propose to improve this mapping by the following. For each scene, we manually select corresponding homogeneous patches from other objects in each pair of images captured by camera A and B in order to improve the generalization of the polynomial fitting. To do that,



iPhone w/ color chart (CC) | Samsung w/ CC | iPhone w/o CC | Samsung w/o CC | Linear mapping iPhone → Samsung | Poly. w/o aug. regions iPhone → Samsung | Poly. w/ aug. regions iPhone → Samsung

Figure 3: We propose to select other corresponding regions, in addition to the color chart's patches, to achieve better pairwise image calibration. As a comparison, we show calibration results using linear and polynomial mapping matrices w/ and w/o augmenting regions.

we designed a tool to assist a human annotator to manually select patches by a simple drag and drop operation. In particular, we asked a human annotator to manually select a correspondence variable-size square patch from A and B such that the patches are strictly homogeneous. Our tool allows selecting a variable number of correspondence patches depending on the given image pair (i.e., more patches are better). We use these extra corresponding colors along with the colors of our calibration chart to compute M. This introduces a noticeable improvement in our mapping, as shown in Fig. 3. Instead of applying the mapping to the image that contains the color chart, we capture another equivalent image (for each scene) without the color chart and apply the mapping to this chart-free image, as shown in Fig. 3.

2.2 Loss Function

To optimize our framework, we used the following loss functions: (i) reconstruction loss, (ii) anchor loss, and (iii) mapping loss. The reconstruction loss $\mathcal{L}_r$ is described as follows:

$$\mathcal{L}_r = \frac{1}{N}\sum_{n=1}^{N} \left\| I_n - \hat{I}_n \right\|_F^2 , \tag{3}$$

where n is the mini-batch index, I and $\hat{I}$ are the input and reconstructed image, respectively, and $\|\cdot\|_F^2$ is squared Frobenius norm. Our mini-batch is designed to contain examples randomly from both unpaired and paired sets. During the training iteration, this loss is used for the samples from the unpaired set. For the samples from the anchor set, we encourage our encoders in both networks to normalize the latent features by using our anchor loss $\mathcal{L}_a$:

$$\mathcal{L}_a = \frac{1}{N}\sum_{e=1}^{E}\sum_{n=1}^{N} \left\| \mathcal{X}_{A_n}^{e} - \mathcal{X}_{B_n}^{e} \right\|_F^2 , \tag{4}$$

where e refers to encoder block indices; $\mathcal{X}_A^e$ and $\mathcal{X}_B^e$ are the latent output of the e^{th} encoder block for camera A's image and camera B's image, respectively. As shown in Eq. 4, instead of penalizing only on the output of the last block of our encoder, we aggregate the loss overall encoder blocks. This enables us to pass high-level latent representations from the encoder net to the decoder net through skip connections to help our reconstruction process. In addition to the anchor loss, we also incorporate our mapping loss:

$$\mathcal{L}_m = \frac{1}{2N}\sum_{n=1}^{N} \left\| I_{A_n} - \hat{I}_{A_n} \right\|_F^2 + \left\| I_{B_n} - \hat{I}_{B_n} \right\|_F^2 , \tag{5}$$

where $\hat{I}_{A_n}$, I_{A_n}, $\hat{I}_{B_n}$, and I_{B_n} are the mapped and corresponding *paired* ground-truth images by cameras A and B, respectively. Our final loss function is then computed as:

$$\mathcal{L} = \mathcal{L}_r + \mathcal{L}_a + \mathcal{L}_m \tag{6}$$

3 Experiments and Discussion

3.1 Datasets

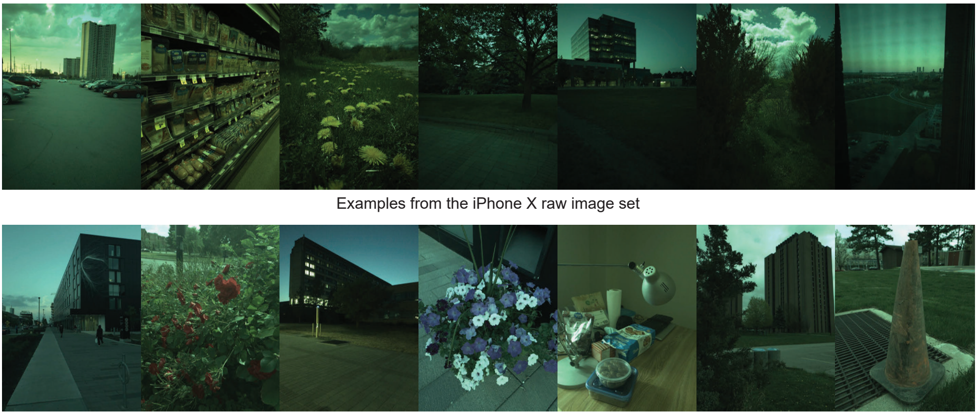
In this section, we provide a description of the datasets used in our proposed framework. Particularly, we use two datasets: (i) the NUS dataset [7], and (ii) our collected dataset.

NUS Dataset We used two sets captured by Canon EOS 600D and Nikon D5200 DSLR cameras. Note that although this dataset provides the same set of scenes, images in this dataset are not aligned and unpaired (i.e., the i^{th} image in the Canon's set does not capture the same scene of the i^{th} image in the Nikon's set). Thus, we manually selected seventeen corresponding images that capture the same scenes from each camera for testing. Additionally, we manually selected five corresponding images from each camera set as our anchor image set. We followed the same procedure explained in Sec. 2.1 to get aligned pairs for both test and anchor image sets. Thus, our final test set includes 34 images for each camera, and our anchor set includes ten images for each camera—the number of images is doubled after mapping all images in each camera set to the other camera. We used the rest of the unpaired images in each camera set to construct our unpaired training sets. In particular, we used 172 images in each camera set. The unpaired set along with the anchor set were used to train our method.

Our Dataset As one of our contributions, we propose a new dataset of raw images captured by two different smartphone cameras: Samsung Galaxy S9 and iPhone X. Fig. 4 shows example raw images from each camera. We opt in to use smartphone cameras for data collection due to the fact that smartphone cameras introduce large differences in spectral sensitivities compared to DSLR cameras [18], which makes raw-to-raw mapping more challenging using smartphone cameras. As far as we know, there is no dataset of raw images captured by two or more smartphone cameras that meet our setup (i.e., contains unpaired and paired raw image sets).

Our dataset consists of an unpaired and paired set of images for each smartphone camera. The unpaired set includes 196 images captured by each smartphone camera (total of 392). The paired set includes 115 pair of images used for testing (generated as described in Sec. 2.1). In addition to this paired set, we have another small set of 22 anchor paired images. Original DNG files of our raw images, in addition to the associated source code to extract raw images and metadata, are available online[2]. To train our method, we used x paired anchor images (we present an ablation study using $x = \{1, 7, 15, 22\}$ in Sec. 3.3). We used the unpaired set for each camera for the unsupervised part of our training.

[2] https://github.com/mahmoudnafifi/raw2raw



Examples from the iPhone X raw image set

Examples from the Samsung Galaxy S9 raw image set

Figure 4: Examples from each camera set. In the top row, we show examples from the iPhone X smartphone camera. In the bottom row, we show examples from the Samsung Galaxy S9 smartphone camera. To aid visualization, the Green channels are averaged (i.e., three-channel RGB image), and a Gamma operation with 1/1.6 encoding gamma is applied.

3.2 Implementation Details

All images are processed after black-level subtraction and image normalization. We examined demosaiced and mosaiced (Bayer) images in our experiments. For the experiments on the NUS dataset [7], we used sensor raw images that were minimally processed using the DCRAW converter as described in the dataset paper [7]. Accordingly, the network's first and last layers were modified to accept/output three-channel images instead of the four channels shown in Fig. 2. We used the metadata provided in the NUS dataset to get black and white levels. For the experiments in our dataset, we used RGGB images after packing each mosaiced image into four channels. For each image, we extracted the black and white levels from the associated DNG file [1]. We end-to-end optimized our network's weights using Adam algorithm [16] with a learning rate of 10^{-4} and beta values 0.9 and 0.999. We set the mini-batch size N to 16, and we trained on 256×256 patches randomly selected from each training image for 140 epochs.

3.3 Comparisons and Ablation Studies

We compared our method with global calibration methods used for raw-to-raw mapping [20]. Specifically, we compared our method with two different global calibration approaches: (i) the linear 3×3 mapping and (ii) the polynomial mapping (here, we used the same polynomial function used in [12, 20]). We first compute a single (either 3×3 or polynomial) calibration matrix from a single pair of color charts captured by cameras A and B. Then, we applied the computed calibration matrix to the testing images. We repeated this experiment multiple times to avoid any bias to the selected pair of images using all pairs in the anchor set. We reported the mean and standard deviation of the results. We further examined the recent Fourier domain adaptation (FDA) method proposed in [26]. To test the FDA method [26], we followed a similar setting, where we used one of the anchor images as a target image and apply the FDA between this image and all testing images. We repeated this process overall

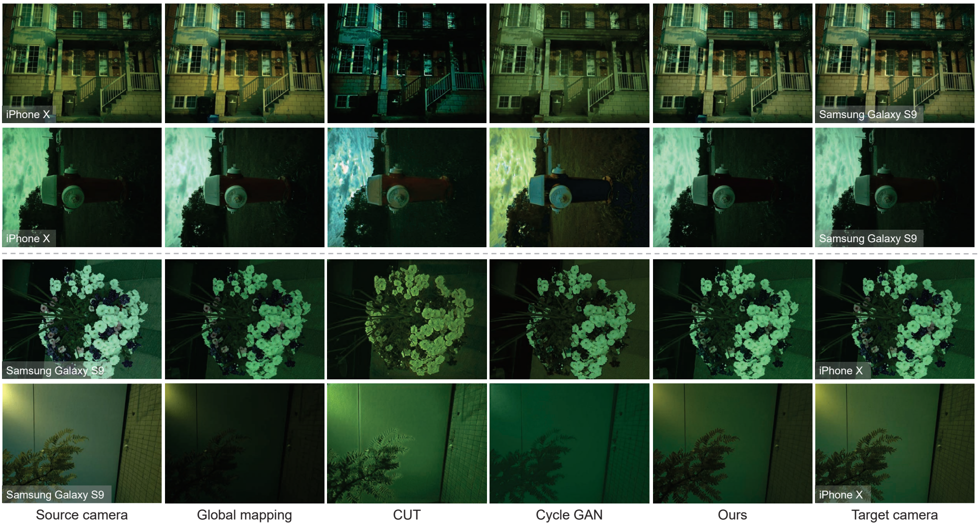


Figure 5: Qualitative comparisons on our dataset. We show our results alongside the results of: global mapping, CUT [21], and Cycle GAN [27]. For each example, we show the "ground-truth" image in the target camera's raw space.

Table 1: Results of mapping between Canon EOS 600D and Nikon D5200 DSLR cameras from the NUS dataset [7]. This table also provides the results of our ablation studies on the effect of our loss terms ($\mathcal{L}_m$, $\mathcal{L}_a$, and $\mathcal{L}_r$). Best results are highlighted in yellow and boldfaced.

Method	Canon → Nikon				Nikon → Canon			
	PSNR↑	SSIM↑	MAE↓	Δ E↓	PSNR↑	SSIM↑	MAE↓	Δ E↓
Global calibration (3×3)	26.99 ± 1.66	0.84 ± 0.04	0.042 ± 0.01	8.66 ± 1.89	27.84 ± 2.28	0.84 ± 0.04	0.04 ± 0.01	7.76 ± 1.87
Global calibration (poly)	21.08 ± 4.58	0.77 ± 0.07	0.087 ± 0.05	12.57 ± 2.45	22.43 ± 7.00	0.78 ± 0.09	0.10 ± 0.05	11.82 ± 4.21
Cycle GAN [27]	27.33	0.83	0.030	15.31	26.85	0.83	0.031	11.49
CUT [21]	26.06	0.71	0.040	16.24	29.85	0.88	0.022	10.35
Ours (w/o $\mathcal{L}_a$and $\mathcal{L}_r$)	30.17	0.91	0.022	6.38	29.69	0.91	0.024	6.23
Ours (w/o $\mathcal{L}_r$)	29.94	0.91	0.023	**6.21**	30.19	0.90	0.024	6.10
Ours (w/o $\mathcal{L}_a$)	29.85	0.90	0.023	6.27	29.10	0.89	0.025	6.03
Ours (w/o $\mathcal{L}_m$)	31.15	0.91	0.022	7.01	29.81	0.91	0.026	6.67
Ours	**32.36**	**0.93**	**0.020**	**6.21**	**30.81**	**0.93**	**0.023**	**5.95**

anchor images and reported the mean and standard deviation of the results.

In addition, we compared our method with two unsupervised image-to-image translation methods. In particular, we reported the results of the Cycle GAN method [27] and the contrastive unpaired translation (CUT) method [21] after training the network of each method using the same training settings used to train our network, with the exception that both Cycle GAN and CUT networks were found to require more epochs to converge. Thus, we increased the training duration to 450 epochs (∼3 times our training epochs). When reporting results on packed Bayer images, the network architectures of Cycle GAN [27] and the CUT [21] methods were modified to accept and output four-channel images.

We also conducted a set of ablation studies to evaluate the improvement gained by our introduced loss terms (i.e., $\mathcal{L}_m$, $\mathcal{L}_a$, and $\mathcal{L}_r$). Note that when using only the mapping loss term, $\mathcal{L}_m$, this is equivalent to train two separate networks on the paired anchor set.

Results of our method, including our ablation studies, and competitive methods on the NUS dataset [20] are reported in Table 1. We show the PSNR score, SSIM score [25], and

Table 2: Results of mapping between Samsung Galaxy S9 and iPhone X cameras on our dataset. The symbol x refers to the number of paired raw images used in the anchor set. Best results are highlighted in yellow and boldfaced.

Method	Samsung → iPhone				iPhone → Samsung			
	PSNR↑	SSIM↑	MAE↓	Δ E↓	PSNR↑	SSIM↑	MAE↓	Δ E↓
Global calibration (3×3)	24.52 ± 3.13	0.71 ± 0.16	0.049 ± 0.02	10.22 ± 3.60	17.03 ± 7.13	0.51 ± 0.30	0.16 ± 0.12	18.76 ± 11.32
Global calibration (poly)	24.88 ± 2.82	0.72 ± 0.16	0.048 ± 0.02	10.08 ± 3.71	16.88 ± 7.18	0.50 ± 0.30	0.16 ± 0.12	19.26 ± 11.23
FDA [26]	20.95 ± 0.28	0.48 ± 0.03	0.06 ± 0.002	21.31 ± 0.86	19.18 ± 0.50	0.47 ± 0.02	0.090 ± 0.004	22.55 ± 1.09
Cycle GAN [27]	24.52	0.71	0.043	14.64	24.35	0.75	0.043	12.84
CUT [21]	22.24	0.71	0.051	15.16	22.79	0.74	0.046	14.94
Ours ($x = 1$)	28.42	0.87	0.031	**5.95**	26.52	0.86	0.039	6.95
Ours ($x = 7$)	28.64	0.88	0.028	6.21	26.70	0.87	0.039	6.93
Ours ($x = 15$)	29.24	0.88	**0.027**	6.66	**28.59**	**0.90**	**0.032**	6.61
Ours ($x = 22$)	**29.65**	**0.89**	**0.027**	6.32	28.58	**0.90**	0.033	**6.53**

mean absolute error (MAE) achieved by each method. Additionally, we measure the visual perception errors of each method using Δ E 2000 [24]. As computing Δ E is performed in the CIE Lab space, we first transform each of the mapped raw images and corresponding ground-truth images to the CIE XYZ space, then we mapped the XYZ images to the CIE Lab space. Mapping images to the CIE XYZ space was performed using the calibration matrices provided by each camera manufacturer. The illuminant vectors – which are required for the CIE XYZ mapping [4, 15] – of the mapped images were obtained using the state-of-the-art sensor-independent illuminant estimation method [4], while we used the achromatic patches in the color calibration chart to determine the illuminant vectors of ground-truth images.

Table 2 shows the results on our proposed dataset. In Table 2, we also report the results of using different number of images in the anchor set ($x = \{1, 7, 15, 22\}$). As shown, even when using a single pair of anchor images, our method achieves superior results to other competitors. We acknowledge that studying the number of anchor images does not give enough insights about the characteristics of anchor data to improve the results (in comparison to, for example, studying the impact of lighting conditions of the anchor set); however, one could study this research question in future work. Note that when we used $x < 22$ in the ablation study, we randomly selected the anchor images from our anchor set, which includes 22 images for each camera.

We provide qualitative comparisons in Fig. 5. Additionally, we examined the idea of "universal camera ISP" mentioned in Sec. 1. Specifically, we trained a deep neural network on our Samsung S9 unpaired set to learn rendering raw images to the sRGB color space. We adopted the same architecture and training setting of AWNet [8]. Then, we feed raw images from iPhone X with and without our raw-to-raw mapping to this trained deep ISP. As shown in Fig. 6, our mapped raw gives reasonable sRGB that perceptually look similar to the actual output from Samsung S9.

3.4 Limitations and Future Work

Given the practical setup used in our experiments, we achieve state-of-the-art results compared to other alternatives. However, our results still need more improvement to be able to achieve more robust raw-to-raw mapping. Specifically, our method fails in some cases, as shown in Fig. 7, where it could not produce good results in dark scenes or scenes with challenging lighting conditions.

Capturing conditions and settings can provide useful cues for better raw-to-raw mapping. To further guide the training process, one could incorporate raw-image metadata (e.g., exposure time, ISO, noise level, and focus distance) to improve the accuracy and robustness.

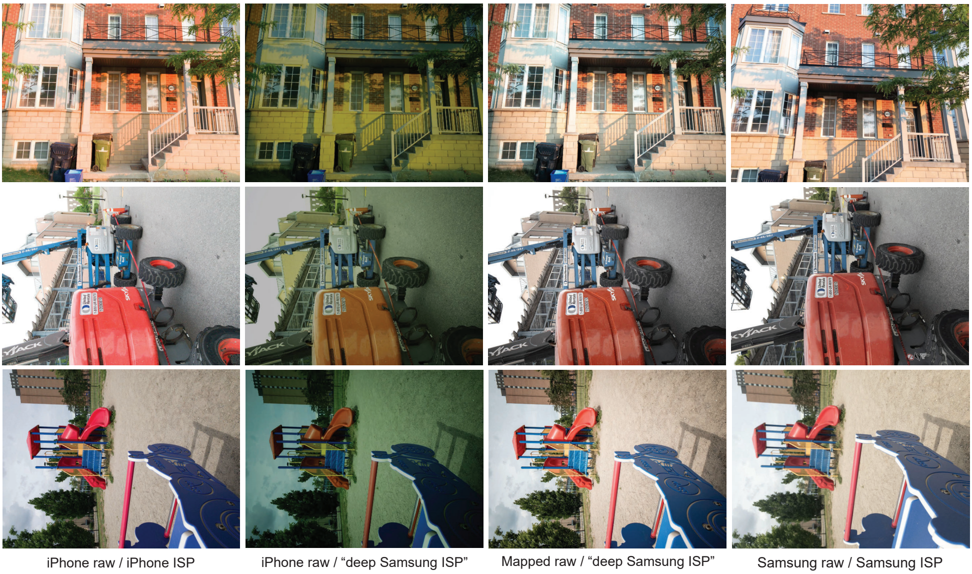

iPhone raw / iPhone ISP
iPhone raw / “deep Samsung ISP”
Mapped raw / “deep Samsung ISP”
Samsung raw / Samsung ISP

Figure 6: This figure shows example sRGB images captured and rendered by two different cameras (iPhone [1st column] and Samsung [4th column]). The 2nd and 3rd columns show results of a deep learning-based camera ISP model trained on Samsung raw/sRGB images. We show examples of feeding iPhone raw images to this trained deep ISP in the 2nd column. We also show in the 3rd column the results of rendering the raw images (mapped from iPhone to Samsung using our raw-to-raw mapping) through the trained Samsung ISP.

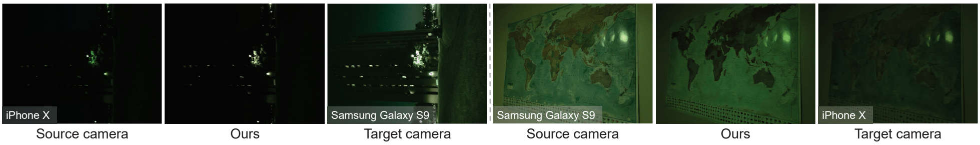


Figure 7: Failure cases. Our method fails in some cases, especially with dark scenes or scenes with challenging light conditions.

Another potential improvement could be achieved by considering an adaptive and sensor-specific amplification for low-light scenes.

4 Conclusion

We have presented a semi-supervised raw-to-raw mapping method. Our work presents a practical way to achieve this mapping with a limited set of paired images required to train the model. Under this practical scenario, we demonstrated state-of-the-art results on two different datasets of DSLR and a new proposed smartphone camera dataset. Our method is the first step towards having practical and accurate raw-to-raw mapping to assist camera ISP manufacturing.

References

[1] Digital negative (DNG) specification. Technical report, Adobe Systems Incorporated, 2012. Version 1.4.0.0.

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

[3] Mahmoud Afifi, Abdelrahman Abdelhamed, Abdullah Abuolaim, Abhijith Punnappurath, and Michael S Brown. CIE XYZ Net: Unprocessing images for low-level computer vision tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–12, 2021.

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

[5] Casper Find Andersen and David Connah. Weighted constrained hue-plane preserving camera characterization. *IEEE Transactions on Image Processing*, 25(9):4329–4339, 2016.

[6] Ronen Basri and David W Jacobs. Lambertian reflectance and linear subspaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):218–233, 2003.

[7] 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.

[8] Linhui Dai, Xiaohong Liu, Chengqi Li, and Jun Chen. AWNet: Attentive wavelet network for image isp. *arXiv preprint arXiv:2008.09228*, 2020.

[9] Graham Finlayson, Han Gong, and Robert B Fisher. Color homography: theory and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(1): 20–33, 2017.

[10] Graham D. Finlayson, Michal Mackiewicz, and Anya Hurlbert. Color correction using root-polynomial regression. *IEEE Transactions on Image Processing*, 24(5):1460–1470, 2015.

[11] Damien Fourure, Rémi Emonet, Elisa Fromont, Damien Muselet, Alain Trémeau, and Christian Wolf. Mixed pooling neural networks for color constancy. In *ICIP*, 2016.

[12] 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.

[13] Andrey Ignatov, Luc Van Gool, and Radu Timofte. Replacing mobile camera ISP with a single deep learning model. In *CVPR Workshops*, 2020.

[14] Jun Jiang, Dengyu Liu, Jinwei Gu, and Sabine Süsstrunk. What is the space of spectral sensitivity functions for digital color cameras? In *WACV*, 2013.

[15] Hakki Can Karaimer and Michael S Brown. Improving color reproduction accuracy on cameras. In *CVPR*, 2018.

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

[17] Zhetong Liang, Jianrui Cai, Zisheng Cao, and Lei Zhang. CameraNet: A two-stage framework for effective camera ISP learning. *IEEE Transactions on Image Processing*, 30:2248–2262, 2021.

[18] Orly Liba, Kiran Murthy, Yun-Ta Tsai, Tim Brooks, Tianfan Xue, Nikhil Karnad, Qiurui He, Jonathan T Barron, Dillon Sharlet, Ryan Geiss, et al. Handheld mobile photography in very low light. *ACM Transactions on Graphics (TOG)*, 38(6):1–16, 2019.

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

[20] Rang Nguyen, Dilip K Prasad, and Michael S Brown. Raw-to-raw: Mapping between image sensor color responses. In *CVPR*, 2014.

[21] Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. In *ECCV*, 2020.

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

[23] Eli Schwartz, Raja Giryes, and Alex M Bronstein. DeepISP: Toward learning an end-to-end image processing pipeline. *IEEE Transactions on Image Processing*, 28(2): 912–923, 2018.

[24] 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.

[25] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

[26] Yanchao Yang and Stefano Soatto. FDA: Fourier domain adaptation for semantic segmentation. In *CVPR*, 2020.

[27] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.