Supplementary Materials for What Else Can Fool Deep Learning? Addressing Color Constancy Errors on Deep Neural Network Performance

Mahmoud Afifi[1]
[1]York University, Toronto
mafifi@eecs.yorku.ca

Michael S Brown[1,2]
[2]Samsung AI Center, Toronto
mbrown@eecs.yorku.ca

The supplemental material is organized as follows. Sec. S1 provides a brief explanation of a standard camera imaging pipeline, including where white balance (WB) is applied. Sec. S2 provides qualitative results of the proposed method. The training details are provided in Sec. S3. Sec. S4 provides more failure examples of pre-trained deep neural network (DNN) models. Lastly, we provide additional results of our experiments in Sec. S5.

S1. Camera Imaging Pipeline

Digital cameras apply a series of processing routines to convert a captured raw-RGB sensor image to the final sRGB output image. These routines are part of the image signal processor (ISP) hardware on the camera. While each camera manufacturer has its own customized ISP, researchers have developed reasonable representative ISP that includes the main components of typical camera pipeline [15, 26].

Fig. S1 provides a high-level diagram of the common steps applied onboard a camera's ISP. Stages in the pipeline that are nonlinear in nature have yellow boxes around them.

Input raw-RGB image The input starts with a minimally processed sensor image referred to as the raw-RGB image. The RGB values are with respect to the spectral sensitivity of the R, G, B color filters on the sensors color filter array.

Demosaicing The first step is to reconstruct three R, G, B values for each pixel based on the incomplete color samples provided by the color filter array.

Noise reduction Many cameras include a routine to reduce sensor noise.

WB and Colorimetric transform The next step is to apply WB correction based on the scene illumination. This is done by applying a single 3×3 diagonal matrix to each RGB color value. The entries of this diagonal matrix are related to the scene illumination. This can be either estimated directly from the image (i.e., auto white balance), or based on a manually selected pre-set matrix (i.e. the user selects the illumination type from a menu). Based on the estimated correlated color temperature of the WB, a colorimetric conversion matrix is then applied to map the white-balanced raw-RGB values to a perceptual color space – namely, the CIE 1932 XYZ space [6]. Note that if the WB is applied incorrectly, the image will have a strong color cast and the resulting CIE XYZ values will be wrong. An example is shown in Fig. S1, where both an incorrect and a correct WB images in their intermediate CIE XYZ image-states are shown after the WB and CIE XYZ transform step.

Hue/saturation Manipulation The hue/saturation manipulation is applied to make the rendered images more appealing [13]. This process is usually implemented as a 3D lookup table (LUT).

Exposure Compensation Beside the physical exposure control (i.e., shutter speed and aperture size), digital exposure may be applied to the pixel intensities using a simple linear gain.

General Color Manipulation Each camera has its own color manipulation function that is usually represented by a 3D LUT and applied in order to get more visually pleasing colors.

sRGB Color Space Conversion At this stage, a 3×3 full matrix is used to convert pixel values from the previous stage to the final sRGB color space. During this stage, a gamut mapping operation is performed to map the out-of-gamut pixels to the sRGB gamut. The simple approach is gamut clipping [21], while gamut compression can be used for a better mapping [12].

Tone Curve Application Before producing the final sRGB image, a camera-specific tone map operation is applied. This tone map is specified based on the current selected camera photo-finishing style. It is worth noting that tone mapping operations may include local image processing that are dynamically changed based on the context of the captured scene [4, 13, 22].

The entire processing chain that "renders" the final sRGB image, as described above can be expressed the following equation [1]:

$$\mathbf{I}_{\text{sRGB}} = f_{\text{raw}\rightarrow\text{sRGB}}(\mathbf{I}_{\text{raw}}), \tag{S1}$$

where $\mathbf{I}_{\text{raw}}$ and $\mathbf{I}_{\text{sRGB}}$ are $3 \times n$ matrices containing sensor raw-RGB and final sRGB rendered values of the image, re-

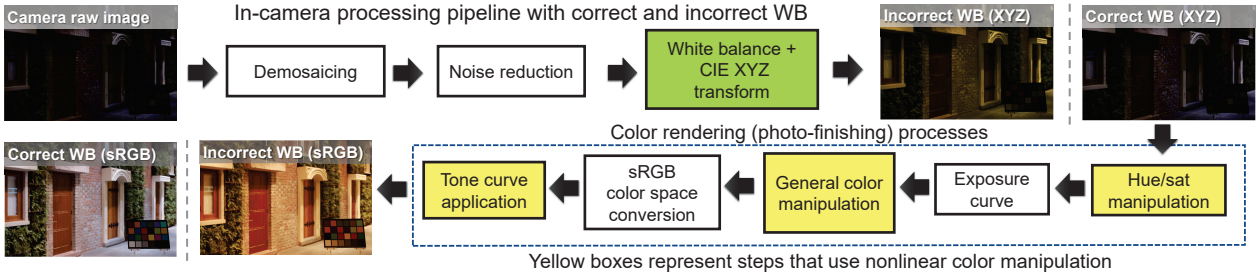


Figure S1. This figure is adapted from [15] and shows a typical camera ISP pipeline, where a correct and incorrect WB has been applied. As can be seen, the WB procedure (shown in green) is performed early in the pipeline. Afterwards, a number of nonlinear color manipulations (shown in yellow) are applied to obtain the sRGB output image. Applying the incorrect WB results incorrect colors in the final output. This examples show the same image rendered with the correct and an incorrect WB setting.

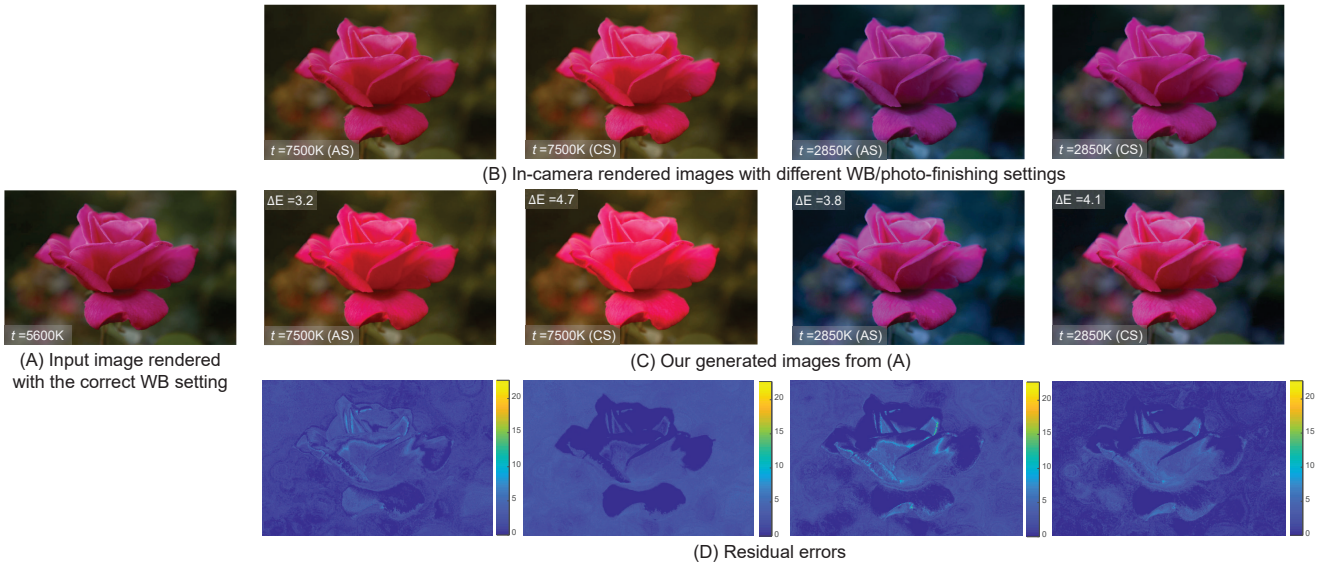


Figure S2. (A) Input image rendered with a correct WB. (B) The same image rendered with different in-camera color temperatures (denoted by t) and photo-finishing styles. (C) Our generated images. (D) Residual error between (B) and (C). The terms AS and CS refer to Adobe Standard and Camera Standard photo-finishing styles, respectively. The $\triangle$E 2000 error [27] is shown on top of each image in (C). The original raw-RGB image was taken from MIT-Adobe FiveK dataset [5].

spectively, and $f(\cdot)_{\text{raw}\rightarrow\text{sRGB}}$ represents the in-camera rendering operations.

It is important to emphasize that the arrangement and the details of the shown camera pipeline components in Fig. S1 may differ based on the camera manufacturer and model producing different colors by each camera capturing the same scene under the same conditions.

From this quick review of camera imaging pipeline, we can see that computing $f^{-1}_{\text{raw}\rightarrow\text{sRGB}}(\cdot)$ to invert the photo-finishing process is a challenging task. The current methods for linearization (i.e., undoing the nonlinearity applied on the rendered sRGB colors) simplify the color rendering procedure by a global tone mapping function, called the camera response function (e.g., [7, 11, 19, 20]), or a single gamma operation (e.g., [2, 9]). This simplifications, however, cannot properly linearize sRGB images captured by arbitrary cameras (e.g., images from the Internet) [1].

In order to undo such nonlinear operations, a careful camera-specific calibration is needed [16]. Even the existing methods for raw-RGB image reconstruction only can work under certain conditions, such as embedding metadata for the reconstruction process [23, 24], or using camera-specific DNN model [22]. Because of these challenges, we sought to find a solution that can work directly in the sRGB color space in order to emulate different WB effects.

S2. Results of WB Errors Emulation

Our WB emulation method produces images with different realistic color casts related to correlated color temperature. Here, we show some additional results of our method in Fig. S2. Specifically, we show that our results are similar

Figure S3. Unlike traditional color augmentation techniques, our generated images are similar to what real camera produces with different WB settings. (A) Input image captured with the correct WB setting. (B) Images generated by RGB color jittering (first row) and HSV jittering (second row). (C) Images generated by our method. (D) The same image rendered with different in-camera WB settings. The color temperatures (denoted by t) are written on each image in (A), (C), and (D). Except for the first image, original raw-RGB images were taken from MIT-Adobe FiveK dataset [5].

to what the actual in-camera sRGB rendered images would look like with the same color temperatures. Fig. S3 shows a qualitative comparison between our results and generated images by traditional color augmentation techniques—namely, RGB color and HSV jittering. We show the real in-camera rendered images with the same color temperatures used by our WB emulation method. We can see that our results are close to real camera rendered images with different WB settings.

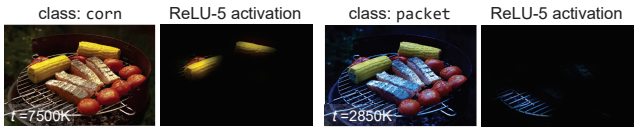


Figure S5. Image rendered with two in-camera color temperatures (denoted by t). The first image (left) is classified as `corn`, while the second image (right) is classified as `packet`. Classification results were obtained by AlexNet [18].

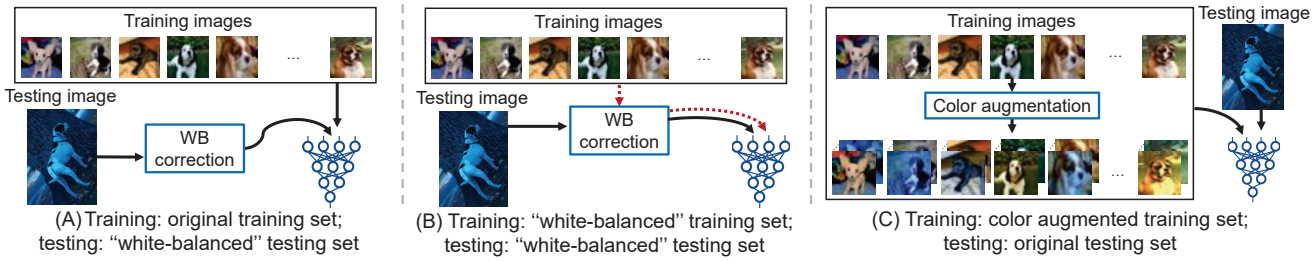


Figure S4. Different strategies to deal with improperly white-balanced images. (A) Traditional training process with unprocessed training set and applying a pre-processing WB correction step to each image in the inference phase. (B) Applying a pre-processing WB correction step to both training and testing images. (C) Color augmenting training images without any pre-processing applied during the inference phase. In (A) and (B), we used the WB-sRGB method [1] for the pre-processing WB correction.

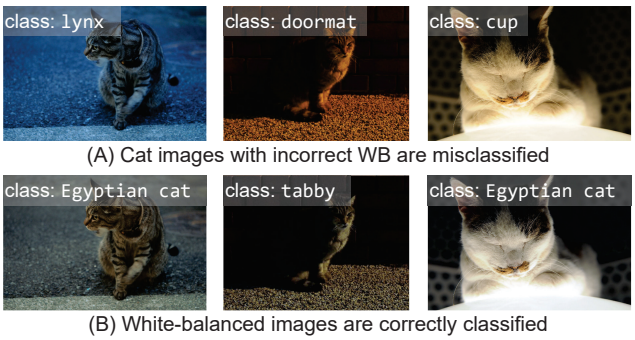


Figure S6. This figure shows the effect of incorrect WB on DNN models for image classification. (A) Three cat images rendered with incorrect WB settings are misclassified by ResNet-50 [14], trained on ImageNet [8]. (B) After applying a pre-processing WB [1], the images are correctly classified. Photo credit: Hisashi Flickr (CC BY-SA 2.0).

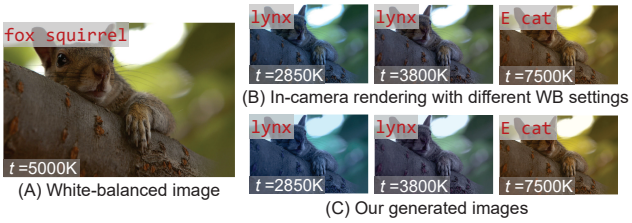


Figure S7. Our proposed method generates images similar to what camera produces with different WB settings which can negatively affect pre-trained DNN models for image classification. In this example, we show predicted classes by AlexNet [18] in top of each image. (A) Correctly white-balanced image. (B) Rendered images with different in-camera WB settings. (C) Our generated images. The color temperature (denoted by t) is shown in bottom of each image. The term E stands to `Egyptian`.

S3. Training Details

In order to examine the potential improvement of the strategies discussed in the main paper (summarized in Fig. S4), we trained SmallNet [25] on CIFAR-10 training set [17]. We also fine-tuned AlexNet [18] to recognize the

Figure S8. Examples from ImageNet validation set [8] (first row) and ADE20K validation set [29] (second row). (A) Original images. (B) Images with different WB settings produced by our method.

Figure S9. A pre-trained DNN for image recoloring detection classifies improperly white-balanced images as " `recolored images`"—which is incorrect. In this example, we used a recent DNN model proposed in [28].

new classes in CIFAR-10/100 datasets [17]. SmallNet and AlexNet were used to evaluate the potential improvement on image classification results. Additionally, we fine-tuned SegNet [3] on the training set of ADE20K dataset [29] for image semantic segmentation. The detailed results obtained by each model were discussed in the main paper. Now, we provide more details about the training process.

As CIFAR dataset contains 32×32 pixels images, SmallNet was implemented to accept images with these dimensions. In order to fine-tune AlexNet, we rescale all images to 227×227 pixels to fit with the input size of the architecture. For SegNet, the input size was 360×480 pixels.

Training was performed using mini-batch stochastic gradient descent with momentum. In our experiments, we used 0.9 momentum. The `L2` regularization factor was set to 0.0005. The mini-batch size was 512 images for SmallNet

and AlexNet. For SegNet, the mini-batch size was 4 images due to the GPU memory limitation.

The cross entropy loss was used for image classification (i.e., SmallNet and AlexNet). For image semantic segmentation (i.e., SegNet), we adopted the weighted pixel-wise entropy loss as be suggested by [3]. The assigned weights for each class were computed using the median frequency balancing [10].

The learning rate λ was as follows. For AlexNet's conv1–fc7 layers, we used $\lambda = 10^{-4}$. For AlexNet's fc8 layer, we used $\lambda = 10^{-4} \times 20$. SmallNet and SegNet were trained using $\lambda = 10^{-3}$. We trained SmallNet and AlexNet for 300 and 30 epochs without and with color augmentations, respectively. SegNet was trained for 110 and 11 epochs without and with color augmentations, respectively. Each model was trained for less than 1 day, ∼15 days , and ∼25 days for SmallNet, AlexNet, and SegNet, respectively.

As demonstrated in the main paper, the reason behinds adjusting the number of epochs for training with and without color augmentation is our seek for unbiased comparisons between models trained on original training sets and those trained on color augmented training sets—color augmented training set is 10 times the original training set. As a result, each model was trained on the same number of mini-batches (i.e., the same number of iterations).

S4. Failure Cases of Pre-trained DNNs

In the main paper, we illustrated an example showing that different WB settings can change the attention of DNN models. In Fig. S5, we show another example illustrating the impact of color casts caused by different WB settings on the DNN's attention.

Fig. S6 shows images rendered with *real* in-camera incorrect WB settings. As shown, pre-trained ResNet-50 [14] misclassifies the original cat images, while they are correctly classified after applying a pre-processing WB correction. In this example, we used the WB-sRGB method [1].

As our generated WB effects are close to what real camera produces, both (real and synthetic) WB settings can fool pre-trained DNNs. Fig. S7 shows an example. In this example, we show the classification results of AlexNet [18].

In Sec. 3 of the main paper, we studied the effect of incorrectly white-balanced images on different pre-trained models for image classification and semantic segmentation. In this study, we used ImageNet validation set [8] and ADE20K validation set [29] after applying different WB settings on each image using our method. Fig. S8 shows examples of the generated images with different WB settings used in our study.

Interestingly, incorrectly white-balanced images can also affect DNN models for other tasks. As an example, Fig. S9 shows images classified as "`recolored images`" by a recent DNN model proposed in [28] for recoloring detection. The shown images are not recolored images; however, they were rendered with an incorrect WB setting which deceives the DNN model.

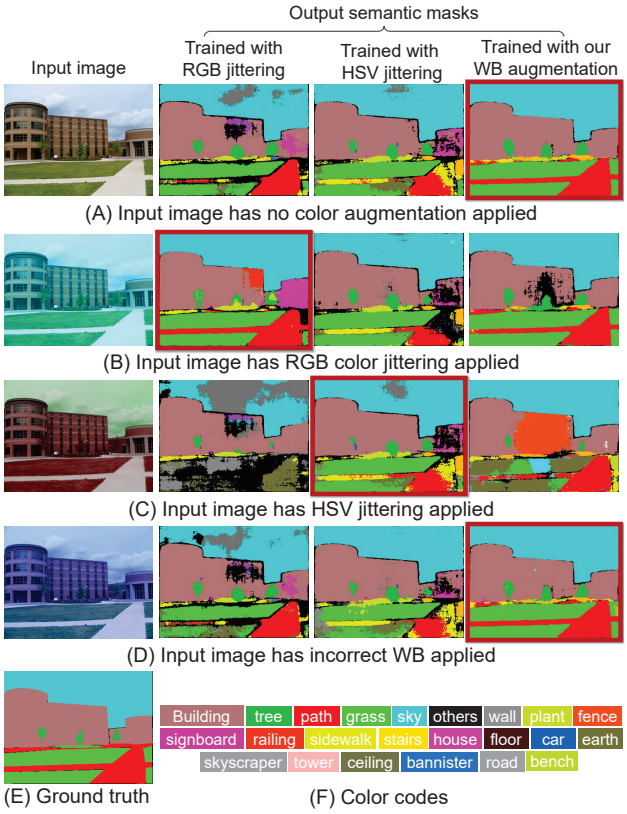


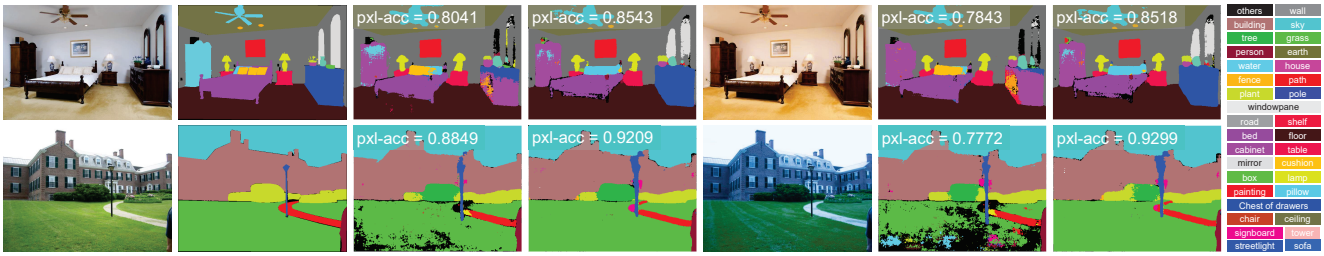
Figure S11. (A) Original image. (B) and (C) Generated by RGB and HSV jittering, respectively. (D) Generated by our WB emulation method. (E) Ground truth semantic mask. (F) color codes. Result masks are obtained by training on augmented data using RGB/HSV jittering and our WB emulation method. The best results are shown in red borders.

S5. Additional Results

In this section, we provide additional results obtained during our experiments discussed in the main paper.

Fig. S10 shows additional examples of SegNet results. As shown, the trained model using our WB augmentation is more stable against changes in WB settings compared to the model trained on the original training set.

It is worth pointing out that when we utilize a certain augmentation technique, we implicitly help the model to expect inputs with similar conditions to what the color augmenter generates. When the testing images having *unrealistic* color manipulations generated by RGB/HSV jittering, we found that trained models on augmented data by these techniques (i.e., RGB/HSV jittering) are more robust than models trained on other types of images (e.g., original or WB augmented training images). However, for images with color casts caused by different WB settings, the trained



(A) Original validation images | (B) Ground truth semantic masks | (C) Results w/o color augmentation | (D) Results w/ WB augmentation | (E) Images with different WB settings | (F) Results w/o color augmentation | (G) Results w/ WB augmentation | (H) Color codes

Figure S10. Additional results of SegNet [3] on ADE20K validation set [29]. (A) Original validation image. (B) Ground truth semantic mask. (C) & (D) Results of trained model wo/w color augmentation using image in (A), respectively. (E) Image with a different WB. (F) & (G) Results w/o and with color augmentation using image in (E), respectively. (H) Color codes. The term 'pxl-acc' refers to pixel-wise accuracy.

model with our WB augmentation has more resistance than other models; Fig. S11 shows an example.

In the main paper, we discussed the results acquired using our collected external set (Cat-2). This set includes 15,098 images rendered with different real in-camera WB settings. This set was generated by rendering raw-RGB images of CIFAR-10 object classes in order to test the trained models on images rendered with in-camera WB settings. Fig. S12 shows examples from our external testing set.

In Fig. S13, we show additional examples, rendered with different in-camera WB settings, which are misclassified by AlexNet trained on the original CIFAR-10 training set. Note that all the shown misclassified images are correctly classified by AlexNet model trained on the WB augmented set generated by our method.

References

[1] Mahmoud Afifi, Brian Price, Scott Cohen, and Michael S Brown. When color constancy goes wrong: Correcting improperly white-balanced images. In *CVPR*, 2019. 1, 2, 4, 5, 7

[2] 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. 2

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017. 4, 5, 6

[4] Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T Barron. Unprocessing images for learned raw denoising. *arXiv preprint arXiv:1811.11127*, 2018. 1

[5] 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. 2, 3

[6] Hakki Can Karaimer and Michael S Brown. Improving color reproduction accuracy on cameras. In *CVPR*, 2018. 1

[7] A. Chakrabarti, Ying Xiong, Baochen Sun, T. Darrell, D. Scharstein, T. Zickler, and K. Saenko. Modeling radiometric uncertainty for vision with tone-mapped color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(11):2185–2198, 2014. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 4, 5

[9] Marc Ebner. *Color Constancy*. John Wiley & Sons, 2007. 2

[10] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 5

[11] M.D. Grossberg and S.K. Nayar. What is the space of camera response functions? In *CVPR*, 2003. 2

[12] Lin Haiting. *A new in-camera color imaging model for computer vision*. PhD thesis, National University of Singapore, 2013. 1

[13] Samuel W Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Transactions on Graphics*, 35(6):192, 2016. 1

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4, 5

[15] Hakki Can Karaimer and Michael S Brown. A software platform for manipulating the camera imaging pipeline. In *ECCV*, 2016. 1, 2

[16] Seon Joo Kim, Hai Ting Lin, Zheng Lu, Sabine Süsstrunk, Stephen Lin, and Michael S Brown. A new in-camera imaging model for color computer vision and its application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2289–2302, 2012. 2

[17] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009. 4, 7

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3, 4, 5, 7

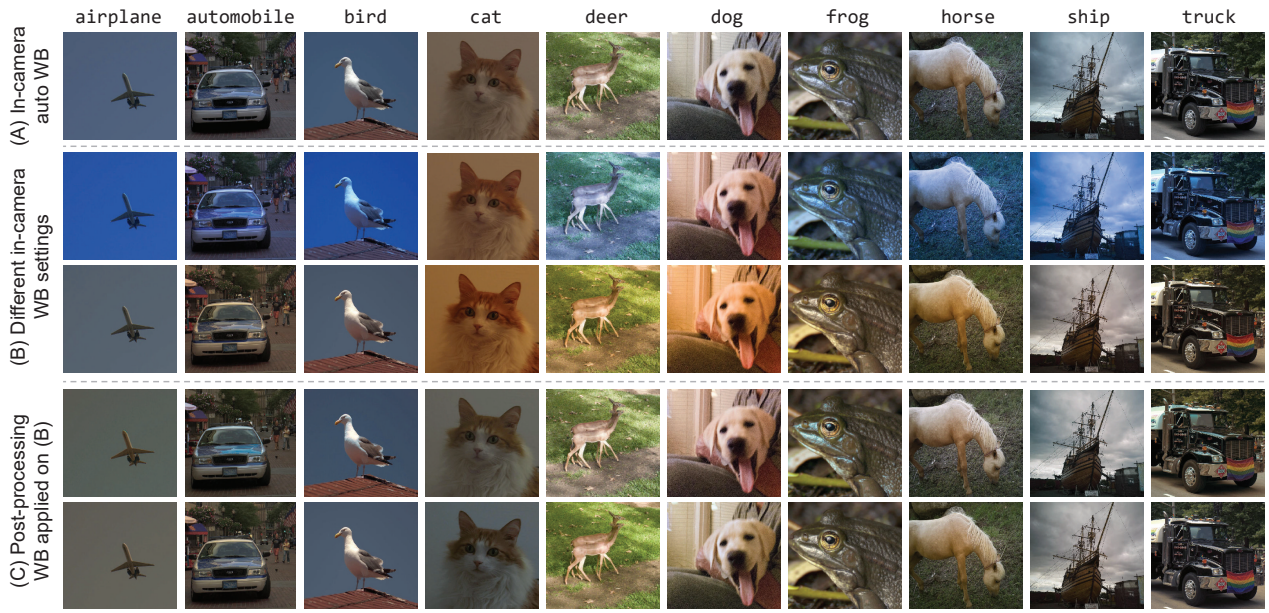


Figure S12. Examples of sRGB images used in Cat-2 (i.e., the external testing set of in-camera rendered images). We used this set to evaluate trained models on CIFAR-10 dataset [17]. Class labels of CIFAR-10 dataset are written on top of each column. (A) Images were rendered using the in-camera auto WB setting. (B) Images were rendered with different WB settings. (C) Pre-processing WB correction [1] is applied to images in (B).

Figure S13. Additional results obtained using Cat-2. (A) Correctly classified images rendered with in-camera auto WB. (B) Misclassified images rendered with *in-camera* different WB. Note that all images in (B) are correctly classified by the same model (AlexNet [18]) trained on WB augmented data.

[19] Chen Li, Stephen Lin, Kun Zhou, and Katsushi Ikeuchi. Radiometric calibration from faces in images. In *CVPR*, 2017. 2

[20] S. Lin, Jinwei Gu, S. Yamazaki, and Heung-Yeung Shum. Radiometric calibration from a single image. In *CVPR*, 2004. 2

[21] Ján Morovč. *To develop a universal gamut mapping algorithm*. PhD thesis, University of Derby, 1998. 1

[22] Seonghyeon Nam and Seon Joo Kim. Modelling the scene dependent imaging in cameras with a deep neural network. In *ICCV*, 2017. 1, 2

[23] Rang MH Nguyen and Michael S Brown. Raw image reconstruction using a self-contained sRGB–JPEG image with small memory overhead. *International Journal of Computer*

Vision, 126(6):637–650, 2018. 2

[24] R. M. H. Nguyen and M. S. Brown. Raw image reconstruction using a self-contained sRGB-JPEG image with only 64 KB overhead. In *CVPR*, 2016. 2

[25] Luis Perez and Jason Wang. The effectiveness of data augmentation in image classification using deep learning. *arXiv preprint arXiv:1712.04621*, 2017. 4

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

[27] 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. 2

[28] Yanyang Yan, Wenqi Ren, and Xiaochun Cao. Recolored image detection via a deep discriminative model. *IEEE Transactions on Information Forensics and Security*, 14(1):5–17, 2019. 4, 5

[29] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *CVPR*, 2017. 4, 5, 6