https://doi.org/10.2352/ISSN.2470-1173.2018.14.HVEI-541
© 2018, Society for Imaging Science and Technology

Colorizing Color Images

Ligeng Zhu, Brian Funt; School of Computing Science, Simon Fraser University, Vancouver, Canada

Abstract

This paper describes a method of improving the quality of the color in color images by colorizing them. In particular, color quality may suffer from improper white balance and other factors such as inadequate camera characterization. Colorization generally refers to the problem of turning a luminance image into a realistic looking color image and impressive results have been reported in the computer vision literature. Based on the assumption that if colorization can successfully predict colors from luminance data alone then it should certainly be able to predict colors from color data, the proposed method employs colorization to 'color' color images. Tests show that the proposed method quite effectively removes color casts—including spatially varying color casts—created by changes in the illumination. The colorization method itself is based on training a deep neural network to learn the connection between the colors in an improperly balanced image and those in a properly balanced one. Unlike many traditional white-balance methods, the proposed method is image-in-image-out and does not explicitly estimate the chromaticity of the illumination nor apply a von-Kries-type adaptation step. The colorization method is also spatially varying and so handles spatially varying illumination conditions without further modification.

Introduction

A popular application of deep learning has been to the problem of automatic colorization; namely, the generation of a plausible full color image from a greyscale (i.e., luminance) image. For example, Iizuka et al. [1] and Zhang et al.[2] have shown that systems employing a convolutional neural network (CNN) can produce quite realistic colorizations that fool observers at least some of the time. Our hypothesis is that with some minor modifications, the same techniques can be used to colorize a color image. The main goal of colorizing color images is to white balance images automatically, especially those with spatially varying scene illumination.

For white balancing, we adapt a modification of the colorization method proposed by Iizuka et al.[1]. Their network is very large with the pretrained model requiring 663MB. We use the modification of that network introduced by Johnson et al. [3] that leads to a much smaller network requiring only 1/20th the storage. In comparison to most existing white balancing methods, the proposed neural network approach requires neither pre-processing nor post-processing—everything is learned and applied in an end-to-end fashion. We train our model on the Microsoft COCO [4] image dataset, which includes 50,000 images for training and an additional 10,000 for validation.

Another term for white balancing in the context of digital images is color constancy. Hence the proposed method also addresses the color constancy problem; however, we avoid further use of that term since Logvinenko et al. [5] proved that color constancy defined as the determination of the intrinsic color of surfaces from trichromatic data simply does not exist in principle. Our 'color constancy' goal, therefore, is not to extract intrinsic surface properties but rather to remove color casts from images.

The vast majority of white balancing methods reported in the research literature use a two-step approach. The first step is to estimate the chromaticity of the scene illumination (often assumed to be constant) and the second is to adjust the image data by scaling the sharpened [6] color channels independently (i.e., von Kries rule or diagonal model) based on the estimated illuminant. In comparison, the colorization-based method proposed here forgoes the illumination-estimation step and directly predicts the output from the input on a pixel-by-pixel, spatially-varying manner.

Approach

The proposed approach is based on deep convolutional neural networks [7], which have been proven able to learn complex mappings from large amounts of training data. Previous work Iizuka et al. [1] has shown deep Convolutional Neural Networks perform colorization well. However, their proposed model is cumbersome, taking weeks to train and hundreds of megabytes to store. Thus we employ an alternative network structure that Johnson et al. [3] originally developed for transferring the style of one image to a second image. We find that its fully convolutional nature to be effective for the standard luminance-to-color colorization task as well. Therefore we adapt this network structure to the color-to-color colorization task.

Our network has several properties that are common to many neural network models. In particular, our model:

- Processes images of any resolution;
- Incorporates hierarchical features;
- Works directly end-to-end without requiring any human intervention.

An overview of our model is shown in Figure 1. Further details about its components are given in Table 1. As an encoder-decoder style of network, it consists of three main components: (1) an *Encoder* to compress the feature map size; (2) a sequence of *Shave Blocks* to distill the information passing through the network; and (3) a *Decoder* to restore the image from the distilled information. Given an input image with possibly poor colors, the output is a recovered image of improved color. All parameters are trained in an end-to-end fashion. No human interaction is required for either training or use.

Encode/Input: The Encoder layer consists of three consecutive *conv* convolutional layers followed by a BatchNorm [8] layer and a ReLU layer that together compress the input image data. The final output of each layer is the result of applying various image filters that extract learned features from the previous layer. During the Encoder stage, the size of the feature maps is reduced

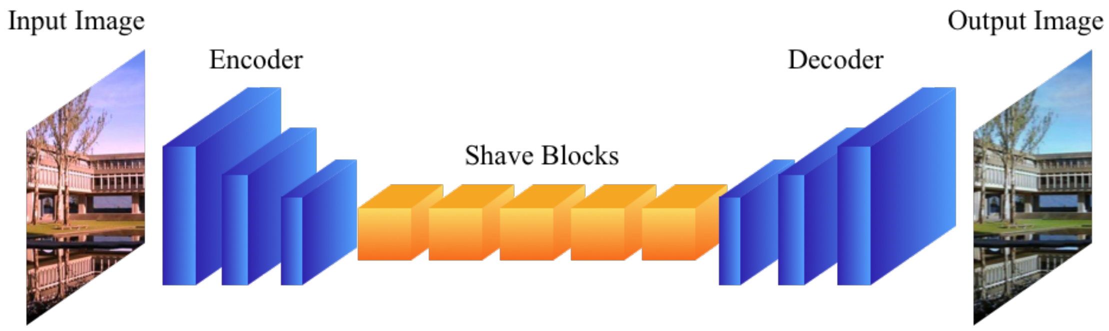

Figure 1: Overview of the proposed color-by-colorization network for automatic white balancing of color images. Blue indicates a composition of a convolutional (conv) layer, a batch normalization (BN) layer, and a rectified linear unit (ReLU). Orange indicates a block that includes a shave layer in parallel to a composite Conv-BN-ReLU-Conv-BN convolutional layer. The results from the parallel paths are then summed.

Encode

type	kernel	stride	output
conv	9x9	1x1	32
conv	3x3	3x3	64
conv	3x3	3x3	128

Shave Block

type	kernel	stride	output
conv	3x3	1x1	128
conv	3x3	1x1	128
shave	-	-	128
sum	-	-	128

Decode

type	kernel	stride	output
deconv	3x3	2x2	64
deconv	3x3	2x2	32
conv	9x9	1x1	3

Table 1: Specifications of the different components used in the color-by-colorization network. *Kernel* indicates the size of the convolution (conv) kernel. *Stride* controls the subsampling of the input data. *Output* refers to the number of convolution filters of the given size and stride used in the respective convolutional layer.

by a factor of 64, while the number of feature maps increases from 3 to 128.

Shave Block: A typical neural network simply stacks layers with connections occurring only between adjacent layers. However, recently Kaiming He et al. [9] showed the advantage of adding 'skip connections' between non-adjacent layers. Given the apparent effectiveness of this strategy, we include in the network design a *Shave Block* having two branches, one branch including two *Conv* layers and the other including one *Shave* layer. The *Shave* layer is similar to an *identity* layer but has its output rescaled to match the size of the output of the other branch. This allows the network to skip *conv* layers even when the dimensions of the layers' outputs do not match.

Decode/Output: After the last Shave block finishes distilling information, two *deconv* layers, which are followed by BatchNorm [8] and ReLU, expand the resulting feature maps back to the original image dimensions. A *conv* layer then further processes the result. A *Tanh* activation function follows the last layer to ensure the results are in the range $[-1,1]$.

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \in (-1,1)$$

Implementation Details

Since the goal of the network is to predict plausible colors from an imperfect input color image, the network's loss function is defined as

$$Loss(I_{predict}, I_{true}) = \frac{\sum_{i,j \in M, c \in \{R,G,B\}} (I_{predict}(i,j,c) - I_{true}(i,j,c))^2}{3 * N}$$

where M is the matrix storing the RGB values from the N image pixels.

The network minimizes the loss function $Loss(I_{predict}, I_{true})$. Since all pixel values are normalized to $[-1,1]$ in experiments, we use *Tahn* as the activation function in the output layer.

Dataset

CNNs require a large volume of training data, usually 5,000 images or more. Unfortunately, existing color constancy datasets (NUS Color Constancy [10], SFU-Gray-Ball [11] and Gehler Colorchecker dataset [12] are not that large. As a result, we start with the Microsoft COCO [4] dataset (a large-scale dataset originally developed for object detection, segmentation, and captioning) and modify it. COCO includes 80,000 images for training, 40,000 images for validation and 40,000 images for testing.

We generate training data by perturbing the COCO images using various different transformations. For training, a perturbed image and its corresponding original image are taken as the input and ground truth, respectively. The perturbation transformations used are as follows.

1. Global von Kries Transformation: For numbers R_1, R_2 chosen randomly from the interval $(0.6, 1.4)$

$$\begin{aligned} I_R(x,y) &= I_R(x,y) * R_1 \\ I_B(x,y) &= I_B(x,y) * R_2 \end{aligned} \tag{1}$$

2. Linearly Interpolated von Kries Transformation in the x-direction or the y-direction.

$$\begin{aligned} I_R(x,y) &= I_R(x,y) * interp(0.6, 1.4) \\ I_B(x,y) &= I_B(x,y) * interp(0.6, 1.4) \end{aligned} \tag{2}$$

The interpolation can be either increasing or decreasing.

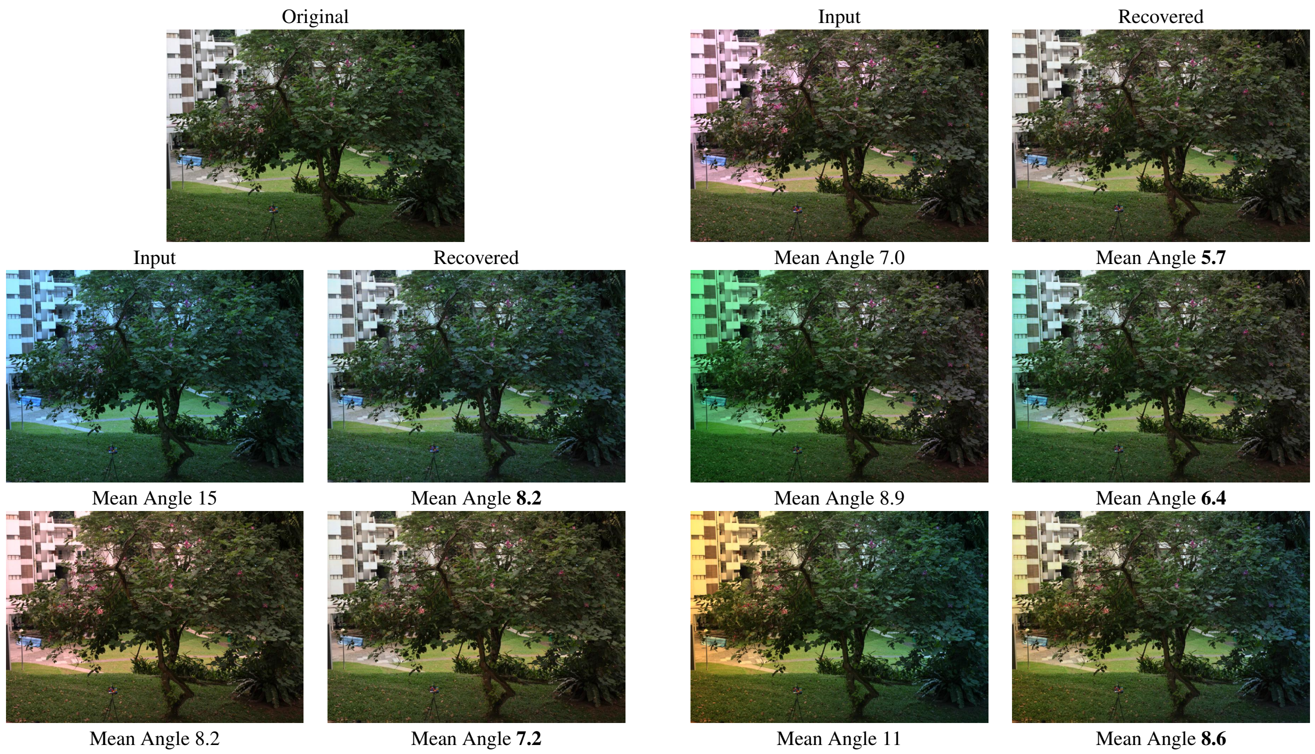

Figure 2: Examples of the proposed color by colorization processing. The lefthand column shows the input images and the righthand column shows the images recovered via the proposed colorization method. The first three rows are examples of a global, spatially uniform color shift as per Eq. 1. The bottom two rows are examples of a spatially varying color shift as per Eq. 2. The Mean Angle is the mean angular difference between the given input or recovered image and the original image across all non-dark (i.e., $(R+G+B) > 150$) pixels. The recovered images are visually much more similar to the original image than the input images. In other words, they demonstrate that the simulated illumination change has been substantially removed.

Thus we have 5 different transformations simulating various degrees of variation in the chromaticity of the illumination across the scene.

Training

All tests are run on an Intel i7-6950k at 2.4 Gigahertz with an NVIDIA GTX 1080Ti GPU. Training is based on $83,000$ images of the COCO 2014 training dataset with perturbed versions added. Images are perpetuated 4 different ways for each epoch. The Adam [13] optimizer is used for training. The learning rate is initialized to 10^{-3} and then reduced by half (i.e., $\eta' = 0.5 * \eta$) every $3,000$ iterations. The batch size is set to 30 and we train the model for 30 epochs (i.e., 30 full passes over the training set). Training takes roughly 16 hours.

Results

For testing, we perturbe a normal color image with the same type of random synthetic illumination variation as used during training. For generality, test images were collected from three sources

1. MS COCO validation set [4]
2. NUS Color dataset [10]
3. Random images from the Internet

It is worth noting that none of these images are seen during training process, especially the NUS Color dataset and random images, which are from completely different sources. In other words, the set of test images is completely separate from the set of training images. The results indicate that the proposed color-by-colorization method is good at generalization.

Examples of the proposed method's results on an image from NUS Color Dataset [10] with five different perturbed versions as input are shown in Figure 2. Four additional examples using perturbed versions of images from the Internet are shown in Figure 3.

Unfortunately, it is not possible to make a direct performance comparison to existing illumination estimation methods because they are based on the assumption that the illumination's chromaticity is constant throughout the scene. In addition, the proposed method corrects the image colors without explicitly estimating the illuminant chromaticity, even locally. As a result, we evaluate the method's performance by comparing the angular difference between the input and processed images on a pixel-by-pixel basis. In the case of a black pixel (R=G=B=0) the angle is undefined. Similarly, the 'color' of very dark pixels can be quite variable without creating a noticeable difference in the image. Hence, we calculate the mean and median angular differences only over pixels for which $R+G+B > 150$. The mean and median per-pixel-thresholded-angular-differences are tabulated in Table 2, where the mean and median are over all the per-image mean errors for each image in the entire test set.

Figures 2 and 3 illustrate the kind of improvement in the im-

Dataset and Setting		Mean	Median	Minimum	Maximum	Best 25%	Worst 25%
NUS Canon EOS-1Ds Mark III	Test v.s. Original	8.3	7.9	**0.64**	21	4.5	13
	Recovered v.s. Original	**6.5**	**6.1**	1.2	**14**	**3.7**	**9.8**
NUS Fujifilm X-M1	Test v.s. Original	7.6	7.3	**0.52**	16	3.5	12
	Recovered v.s. Original	**5.1**	**4.8**	1.1	**13**	**3.0**	**7.7**
NUS Nikon D5200	Test v.s. Original	7.6	7.2	**0.25**	**19**	3.6	12
	Recovered v.s. Original	**4.7**	**4.4**	1.0	16	**2.4**	**11**
NUS Samsung NX2000	Test v.s. Original	7.7	7.0	**0.49**	18	3.6	13
	Recovered v.s. Original	**4.3**	**3.8**	1.0	**13**	**2.1**	**7.1**
NUS Sony SLT-A57 (*)	Test v.s. Original	7.5	7.2	**0.29**	17	3.4	12
	Recovered v.s. Original	**5.0**	**4.7**	1.2	**13**	**2.7**	**7.9**
MS COCO	Test v.s. Original	7.5	7.0	**0.46**	19	3.5	12
	Recovered v.s. Original	**4.3**	**3.9**	0.43	**16**	**2.1**	**7.1**

Table 2: Statistics of the pixel-wise angular difference relative to the original images. The mean, median, maximum and minimum are taken over the per-image-mean-thresholded-angular-differences across all images in the given dataset. Best 25% and Worst 25% are the means taken over the 25% of the images with the lowest and highest mean errors, respectively.

age color that the proposed method provides. In terms of a numerical evaluation, Table 2 shows that the proposed method on average reduces the mean angular error to 65% of its initial value.

Conclusion

Inspired by the success in colorizing greyscale images reported by researchers in the computer vision field, we investigate applying the same general techniques to the problem of 'colorizing' color images. Since it is impossible to know at this point exactly what it is that a deep neural network learns, we can only speculate as to precisely why they are successful in colorizing greyscale images. However, it would appear that the networks are learning something about image content since colorization methods always manage to color the sky blue, beaches a sandy color, faces a flesh tone, and so on. In terms of color correcting or color balancing color images, the majority of illumination estimation methods used for those purposes do not exploit knowledge of the image content. The fact that colorization methods (likely) do is therefore an important advantage to using them.

Acknowledgement

Funding was provided by the Natural Sciences and Engineering Research Council of Canada.

References

[1] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *Proc. of SIGGRAPH 2016*, 35(4):110:1–110:11, 2016.

[2] R Zhang, P Isola, and A A Efros. Colorful image colorization. In *ECCV*, 2016.

[3] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

[4] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[5] Alexander D. Logvinenko, Brian Funt, Hamidreza Mirzaei, and Rumi Tokunaga. Rethinking colour constancy. *PLOS ONE DOI*, 10.1371/journal.pone.0135029, 09 2015.

[6] G Finlayson, M Drew, and B.V. Funt. Spectral sharpening: Sensor transformations for improved color constancy. *Journal of the Optical Society of America A*, 1994.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] D Cheng, D K. Prasad, and M S. Brown. Illuminant estimation for color constancy: Why spatial domain methods work and the role of the color distribution. In *JOSA A*, volume 31, pages 1049–1058, 2014.

[11] F. Ciurea and B Funt. A large image database for color constancy research. In *Proceedings of the Eleventh Color Imaging Conference*, 2003.

[12] Peter Vincent Gehler, Carsten Rother, Andrew Blake, Tom Minka, and Toby Sharp. Bayesian color constancy revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[13] D P. Kingma and J Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

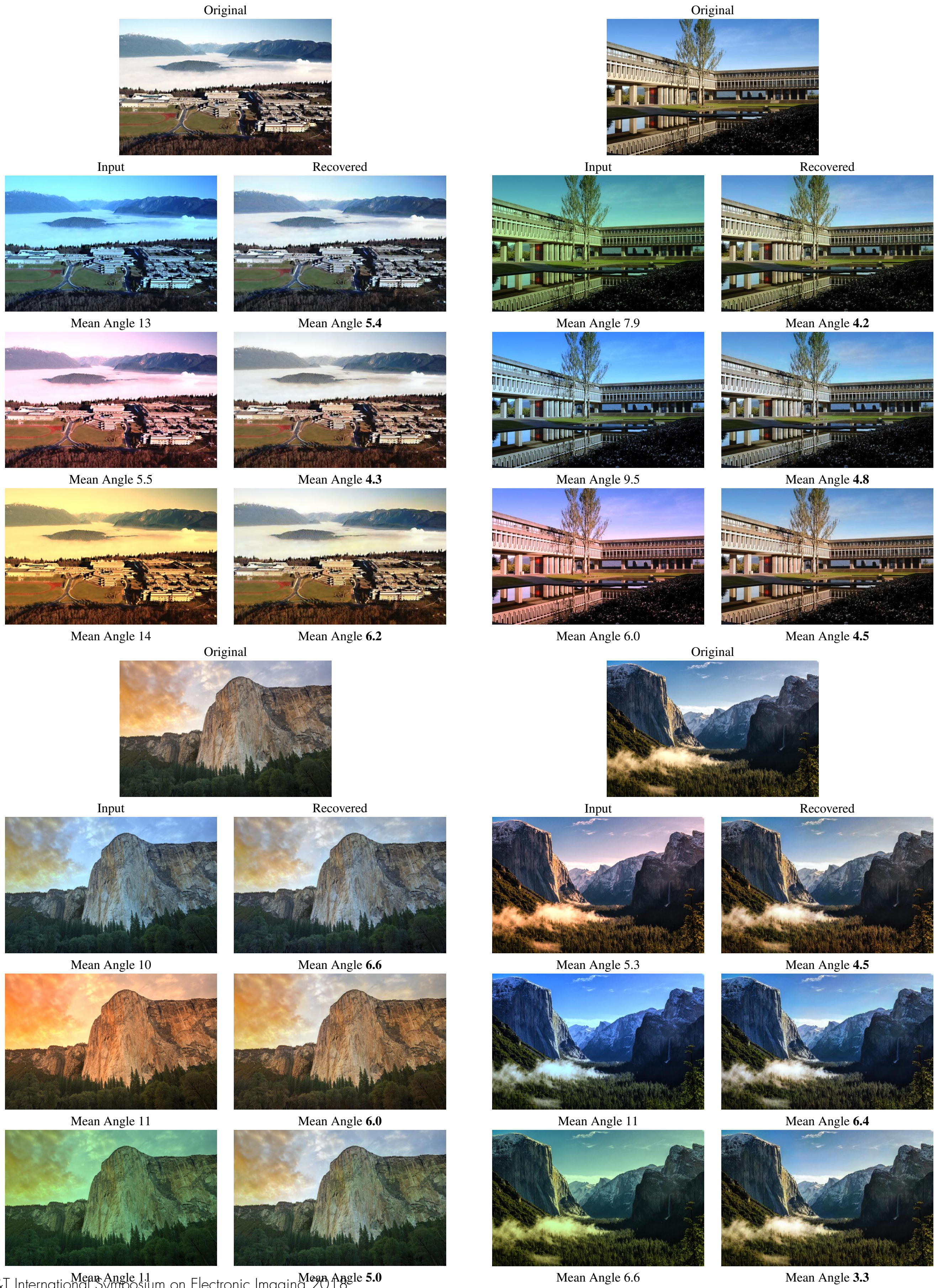

Figure 3: Results for images from the Internet that are completely separate from the COCO dataset. These examples show that as a general rule the color-by-colorization method performs better on images of natural scenes than for those primarily contaning manmade objects.

Author Biography

Ligeng Zhu plans to receive his B.Sc. in Computing Science from Simon Fraser University and B.Eng. in Computer Science from Zhejiang University in 2018. His research work is focused on applied statistics and machine learning. E-mail: lykenz@sfu.ca.

Brian Funt is Professor of Computing Science at Simon Fraser University where he has been since 1980. He obtained his Ph.D. in Computer Science from the University of British Columbia in 1976. His research focus is on computational approaches to modeling and understanding visual perception generally and color in particular. E-mail: funt@sfu.ca.