Exploring Histogram-Based Color Constancy

David R. Treadwell IV
treadwell.d@northeastern.edu
Yunxuan Rao
rao.yu@northeastern.edu
Daniel Bi
danielyhbi@gmail.com
Bruce A. Maxwell
b.maxwell@northeastern.edu

Northeastern University
Seattle, WA

Abstract

We present a lightweight fully convolutional network for color constancy (LHCC). The network uses multiple 2-D projections of the 3-D log RGB histogram of an image in order to predict the color correction coefficients.

In developing the network, we explored whether to use linear RGB or log RGB data, the network structure (width and depth), how to handle dark pixels, how to generate the 2-D mappings of the 3-D histogram, and how to normalize or transform the bin counts in order to preserve the fine histogram structure. Our results show that attention to each of these details makes a difference in overall performance.

Our most significant findings are that using log RGB outperforms linear RGB for this task, and that using a log transformation of the bin counts outperforms thresholding, a hyperbolic tangent, and linear normalization.

Our exploration resulted in a fully convolutional network with 0.5M parameters that sets a new performance standard on SimpleCube++ with a surprising 2.72° angular error on the worst 25%. It is also competitive on older data sets such as Gehler-Shi and NUS-8.

1 Introduction

Color constancy, or white balancing, is a core task that occurs early in the image processing pipeline. The goal of color constancy is to adjust the color channel gains to make surfaces with neutral reflection appear neutral in the resulting image. While not strictly necessary for computer vision, it is a necessary step for making images appear correct to people.

Color constancy is built on a model of the response of a camera I_c. I_c is the product of the color of the illumination $L(\lambda)$, the color of the reflectance $R(\lambda)$, and the sensitivity of the camera for color channel c, $S_c(\lambda)$ integrated over the visible wavelengths λ.

$$I_c = \int_\lambda S_c(\lambda)L(\lambda)R(\lambda)d\lambda \tag{1}$$

Most models of appearance assume sharp sensors, simplifying the integral to a product term for each color channel. Following the von Kries model [30], the task of color constancy is to estimate the relative values of $K_c = S_c L_c$ across color channels to create a scalar for each channel such that if R_c is uniform across channels then I_c is also uniform.

$$I_c = S_c L_c R_c = (S_c L_c) R_c = K_c R_c \tag{2}$$

The correction factor is T/K_c, where T is a constant that sets the brightness of the resulting image, such as $T = K_g$, which maintains the original green channel values.

Color constancy is one of the few vision tasks based on linear RGB data instead of sRGB. While there is some work in color constancy for processed imagery, most data sets and methods use the raw sensor data, taking into account both the sensor response function and the illuminant. Almost all color constancy algorithms use linear RGB as their input. Two exceptions are the Fast-Fourier Color Constancy (FFCC) approach [2], and Afifi and Brown [1], both of which use 2-D histograms of a log of chromaticity space (*e.g.* $\{u,v\} = \log(\{R/G, B/G\})$), first suggested and used by Finlayson [8]. These are not 2-D projections of 3-D spaces, but inherently 2-D spaces. It is not possible to estimate the original color values from multiple log of chromaticity values because the overall intensity of the colors is lost in the ratio calculation.

As far as we know, no color constancy method has yet used the plain logarithm of RGB (log RGB) as an input space, either for semantic analysis (image as input) or histogram-based analysis. The potential benefits of log RGB for image analysis were first noted in [20]. Under a Lambertian image model, taking the log RGB separates the reflectance and illumination into additive terms rather than multiplicative terms. As illumination varies in color or intensity on a surface, the direction of those changes is independent of the reflectance. This fact has been used in classic intrinsic imaging to enable linear optimization methods for separating reflectance and illumination (*e.g.* [11]). It has also used for shadow removal on road surfaces [21].

In addition to exploring semantic log RGB and linear RGB input spaces, we also explore 2-D projections of 3-D histograms. These 2-D projections maintain the intensity channel of the data, and 3-D information can be inferred from multiple 2-D projections.

Our primary hypothesis is that the separation of reflectance and illumination characteristics in log space will make identifying the primary illuminant direction simpler. We also hypothesize that histogram-based analysis has the potential to be faster and more robust than semantic approaches.

Our contributions include the following. (1) We evaluate the use of log RGB for color constancy using both semantic and histogram-based deep network architectures. (2) We present a novel architecture for histogram-based color constancy that takes as input multiple 2-D projections of 3-D histograms. (3) We provide a potential explanation for why log RGB makes the color constancy problem simpler. (4) We demonstrate new state-of-the-art results on the SimpleCube++ data set and competitive results on Gehler-Shi and NUS-8 with a small convolutional network.

2 Related Work

Classical color constancy methods are well summarized by Gijsenij *et al.* [13]. Of particular relevance to the later discussion is the grey-world method of Buchsbaum, which proposes that the mean color of a typical natural image should be grey [4].

2.1 Semantic-based Color Constancy

Prior work has included using standard network architectures that take an image as the input, such as GoogleNet [27]. Sidhorov used a GoogleNet pre-trained on ImageNet with an angular loss function to achieve competitive results on the SFU Greyball and the Gehler-Shi ColorChecker data sets [26]. Lo *et al*. used a SqeezeNet pre-trained on ImageNet to implement a contrastive learning approach to color constancy with strong results on the NUS-8 and ColorChecker data sets [19].

Hu, Wang, and Lin developed a fully convolutional network FC4, that estimates confidence maps for computing the correction coefficients [14]. Yu *et al*. enhanced performance by building a cascading convolutional network [32]. Both networks build on standard backbone architectures such as AlexNet or SqueezeNet. While both show good performance, the standard SqueezeNet architecture is 1.2M parameters, with C4 using a cascade of networks, making them less suitable for edge devices than smaller networks.

2.2 Histogram-based Color Constancy

Cardei *et al*. proposed the first ANN for color constancy using a binarized *rg* chromaticity histogram as input [5]. Bianco *et al*. demonstrated the first deep network approach to color constancy. While the input to the Bianco network was a 32x32 patch of the image, the first layer effectively learned a histogram with 240 bins, making it a histogram-like approach [3].

Fast Fourier Color Constancy (FFCC) by Barron and Tsai is a computationally lightweight and high performing histogram-based approach to color constancy [2]. It uses a log of chromaticity histogram with the axes $u = \log(B/R)$ and $v = \log(G/R)$. Its unique approach is to map the histogram onto a torus, compressing the analysis space and enabling fast complex filtering in the Fourier domain.

Afifi and Brown proposed an alternative histogram method that uses a pre-network to estimate an image-specific transformation of the color space to normalize the data across cameras, followed by a second network that estimates the color correction coefficients [1]. They use all six possible log of chromaticity histograms as a stacked input, but also use pixel intensity as a modifier on the weight of each pixel added to the histogram, emphasizing brighter pixels in the histogram building process. This approach discounts the contribution of darker materials, which may be important indicators of the actual color distribution.

More recently, Ershov *et al*. proposed a different approach to color constancy by re-annotating the Cube++ data set to take into account all of the colors on a chrome ball attached to the calibration cube [7]. Using the enhanced data, they learned to estimate a histogram of plausible illuminants instead of a single set of correction coefficients. Their network learns the conversion from a 128x128 2-D rg chromaticity histogram to a 2-D distribution of plausible illuminants, from which the user can select the most pleasing. The approach suggests a new definition of the meaning of color constancy for interactive editing.

Our CNN network design uses multiple 2-D projections of 3-D log RGB or linear RGB histograms. Since we use multiple views of a 3-D space, we explored several different approaches to combine the views and found that a slow-fusion style approach, as proposed by Karpathy *et al*. for video analysis, gave the best results [15].

2.3 Data Sets

SimpleCube++ is a subset of Cube++ [6] and contains images that have a single primary illuminant, defined by the two sides of the calibration cube having values within 1°. It contains 2,234 images with a recommended train/test split. SimpleCube++ offers a larger and more recent collection of images, although it has fewer results reported in the literature.

Gehler-Shi, or ColorChecker, contains 568 images [12]. This data set had a major revision by Shi and Funt, which provided a re-processed version [25]. Given the small size of this data set, papers typically report the results of 3-fold cross-validation (CV).

NUS-8 contains 1,736 total images taken by 8 cameras. Most reported results train a model per camera with 3-fold CV and report the geometric mean across all cameras. A few approaches have developed camera-independent models [1] that train on the whole data set.

3 Methods and Background

We chose to explore both linear and log RGB due to recent work showing it provided benefits for other computer vision tasks [10, 22, 23]. The basis for why log RGB is potentially interesting comes from the Bi-Illuminant Dichromatic Reflection (BIDR) Model, which models the appearance of surfaces and showed that the ambient term is essential to understanding appearance under varying illumination in natural environments [20].

In log RGB space–given reflectance R_b, ambient illuminant L_a, direct illuminant L_d, and direct illuminant visibility γ–the body reflection model in (3) shows that the appearance of the surface I from shadow to lit is an almost linear cylinder as γ goes from 0 to 1. Furthermore, the direction of the cylinders in log space is the same for all materials under the same ambient and direct illumination pair.

$$\log I = \log R_b + \log(L_a + \gamma L_d) \tag{3}$$

The plane defined by this vector, called the illumination spectral direction (ISD), enables an illumination invariant chromaticity space. Therefore, the distribution of points on the log space chromaticity plane defines the gamut of material colors within the scene, independent of how they are illuminated.

The von Kries model assumes that color correction requires a single scalar multiplier per color channel [30]. In linear space, this represents a scaling of the RGB dimensions. Scaling changes the relative locations of colors in linear RGB. In log space, however, color balance is a translation of the RGB values and, therefore, the relative positions of the image colors do not change. The goal of color constancy in log space is to center the distribution of colors in the correct location, not to scale it to best fit a particular shape.

The center of a color distribution is dependent on the illumination color unless there is an illumination-invariant method of computing the distribution. If we define the proper correction to be **the one that makes a fully-lit neutral surface appear neutral**, then including shadowed pixels in the statistical computation–*e.g.* the mean value of each color channel–implies that the resulting correction is biased by the ambient illumination color.

Figure 1A and D show an uncorrected scene of a neutral surface with and without a shadow. Figure 1B and C show the corresponding rg chromaticity histograms ($r = R/(R+G+B)$ and $g = G/(R+G+B)$). Note the increased spread of colors that occurs in the half-shadowed image. Figure 1C exhibits two overlapping ellipses which correspond to the distribution of colors for each of the two illuminants in the scene (yellow direct and blue

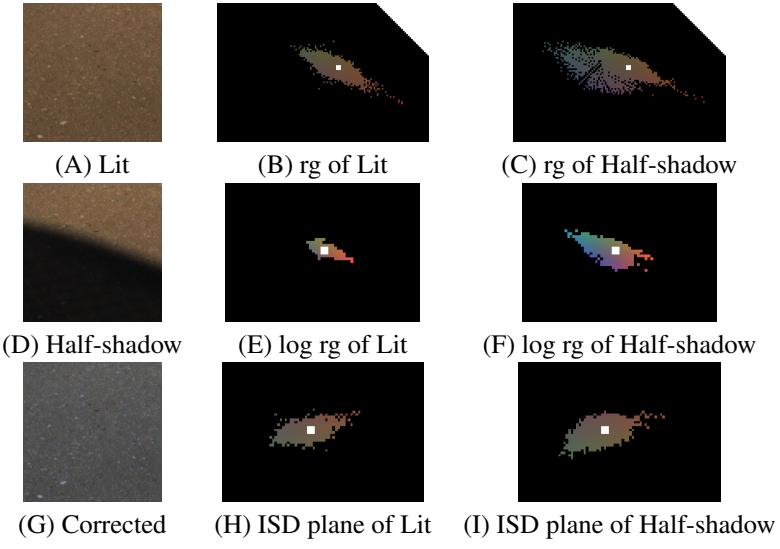

Figure 1: Left column: uncorrected images of the same neutral material (A) fully Lit, (D) Half-shadow, and (G) corrected Lit image. Middle Column: histograms of the Lit image. Right Column: histograms of the Half-shadow image

ambient). The histograms also show the average color as a white dot. The average colors of the lit rg histogram and the half-shadow rg histogram are over 6° degrees apart.

Figure 1E and F show histograms of the same images projected onto the (1, 1, 1) plane in log RGB space. These histograms are more compact, but they also show a bigger spread of colors for the half-shadow image. The log space histogram averages are also over 6° apart.

Gamut-based color constancy methods assume that there is a wide distribution of colors in an image, and that the lack of certain colors provides a guide as to the illuminant color [9]. The presence of shadows or shading causes the gamut of linear colors to expand beyond what would occur under only the direct illuminant, as can be clearly seen in figure 1C.

In log space, shadows and shading also bias the distribution of points in a scene. However, the bias occurs along a single dimension: the ISD. Any statistic computed on the plane defined by the ISD will be illuminant invariant and represent only the gamut of material colors in the image. The challenge with using the ISD is that it is unique to the direct-ambient pair in the image: it is image-specific. However, the information necessary to compute an ISD is in the log RGB histogram and available to a network learning with this input space.

For our example images, we can build a histogram on the plane perpendicular the ISD (computed manually), shown in figure 1H and I. Note the similarity of the log space chromaticity histograms for the fully lit and half-shadowed images. Furthermore, the difference in the means for the two ISD log chromaticity planes is only 0.6°, an order of magnitude smaller than the difference using the other two projections.

The visualizations and differences in the means demonstrate the impact of a shadow on statistical and gamut distributions. If a network can estimate the spectral ratio of the scene, perhaps by identifying structure in the histogram, it may be able to better estimate the color correction factors independently of shadows and shading in the image. We believe this constitutes at least part of the value of using log RGB data as an input to color constancy.

3.1 Implementation Details

Noise Thresholding: One issue with log RGB is that very dark pixels exhibit large changes in their log value for small changes in their linear value. For example, the difference between the log of 2 and the log of 1 is the same as the difference between the log of 20 and the log of 10. Therefore, camera noise is significant in log RGB when the values are very small. In linear RGB, the effect of camera noise is roughly constant across intensities. Based on measurements of dark areas across the cameras in the SimpleCube++ data set we build histograms from 8-bit data in the range [1,255], and for 16-bit data in [257, 65535].

Histogram Saturation: In image histograms, the difference between bin counts can be extremely large, hiding smaller structure. We compared three methods of reducing the range of the histogram bins: thresholding, a hyperbolic tangent, and a log transformation. We also tested applying no transform as a control. All three saturation methods demonstrated superior performance to to the control, with the log transformation of the bin counts producing the best results, as seen in our ablation study in table 4. Unless otherwise specified, all results use histograms with a log transformation applied to the bin counts before analysis.

Planar Projections: We explored two methods of capturing the 3-D structure of linear and log histograms using multiple 2-D mappings. The first method used 128x128 histograms of the 2-D projections onto the R-G, R-B, G-B, and (1, 1, 1) planes.

Eigenvector-Based Histogram Slicing: Our second method used slices of the 3-D histogram space. Using a randomly selected 100 RAW images from a personal image library of 6k diverse images, we calculated the eigenvectors of the aggregate histogram in both linear and log RGB space. In both cases, the primary eigenvector tracked intensity, and the results were similar across multiple subsets. We generated four slices perpendicular to the second eigenvector, projecting the bins within each slice onto the plane formed by the first and third eigenvectors. We examined the distribution of pixels along the second eigenvector to select slices with similar proportions of pixels. Given the central tendency of the distribution, this led to slice widths of (0.47, 0.03, 0.03, 0.47) for linear, and (0.45, 0.025, 0.025, 0.50) for log.

4 Experiments and Results

We developed our network design using the suggested training set for SimpleCube++, tuning the network structure and histogram processing methods mentioned in 3.1 using a randomly selected subset of 1,310 images for a training set and 462 images for a validation set. We then evaluated the network design on the 462 SimpleCube++ recommended test images, training the network using all 1,772 available SimpleCube++ recommended training images. We also evaluated the Gehler-Shi Color Checker and NUS-8 data sets using 3-fold cross-validation.

We used a linear, multi-pass tuning process to develop the LHCC structure. All structural tuning used the SimpleCube++ data set with four 2D projections, log RGB color space, dark thresholding, and a log transformation applied to the bin counts, with images downscaled to 0.75 size. The structural tuning explored the complexity of both the per-histogram blocks and the post-concatenation block, testing numbers of layers and kernel sizes.

The tuning process used stride-2 convolutions to reduce the spatial size of the layers. After tuning, we fixed the architecture to compare the use of 2x2 max-pooling layers to the stride-2 convolutions, but found that stride-2 convolutions produced better results. We then fine-tuned the number of channels per layer and the specific location of the second pooling layer in the per-histogram blocks. Our final network design is shown in figure 2.

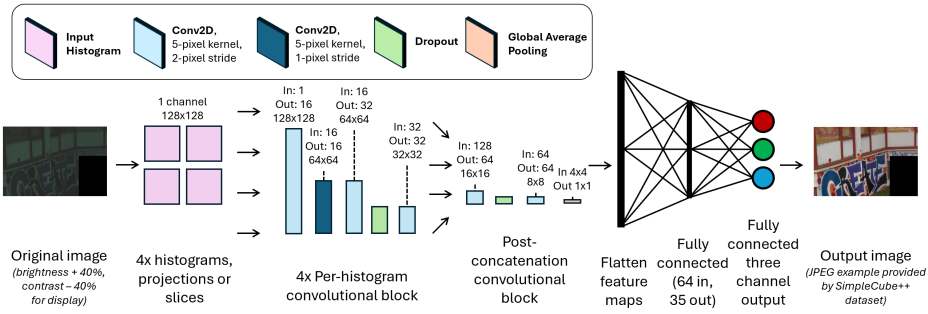

Figure 2: Diagram of the LHCC network architecture. The network contains four input convolution stacks, one per histogram. The structure concatenates the outputs of the per-histogram blocks to pass to the post-concatenation block. The corrected image is generated by scaling to the 8 bit range, dividing each color channel by its correction value (the "Fully connected three channel output"), multiplying by the pre-correction maximum green value divided by the maximum RGB value, and finally converting to 8 bit sRGB.

4.1 Results

We used the Adam optimizer, running for 200 epochs, for all reported results [17]. We tuned learning rate and batch size separately for linear RGB and log RGB. We also scaled colors to 16-bits prior to processing and downscaled all input images to 0.75 their original size.

Table 1 shows the results on SimpleCube++. The LHCC network achieves a new state-of-the-art, with a mean angular error of 1.03° and a worst 25% error under 2.75°. Both the LHCC using linear RGB and the LHCC using slices of log RGB produce comparatively excellent results, but do not perform as well as log RGB data with histogram projections.

SimpleCube++ Method	Mean	Med	Tri	25% Best	25% Worst	Parameters
Gray World [31]	3.18	2.00	2.37	-	-	-
Grey World 2OE [31]	3.06	1.75	2.08	-	-	-
FFCC (train/test)	2.64	1.75	1.85	0.52	6.35	12,288
FFCC (3-fold)	1.26	0.59	0.69	**0.18**	3.55	12,288
FC4 (log) [14] [24]	2.83	1.17	1.51	0.35	8.31	1,705,284
FC4 (default) [14] [24]	1.65	1.03	1.15	0.32	4.02	1,705,284
Afifi & Brown (SIIE, **Cube**) [1]	1.98	1.36	-	0.40	4.64	1,008,044
Afifi & Brown (SIIE, **Cube+**) [1]	2.14	1.44	-	0.44	5.06	1,008,044
CCMNet (**Cube+**) [16]	1.68	1.16	1.26	0.38	3.89	-
CauNet (**Cube+**) [18]	1.61	0.97	1.08	-	4.08	-
GoogLeNet (linear, ours)	1.53	0.95	1.05	0.25	3.90	6,797,700
GoogLeNet (log, ours)	1.35	0.75	0.88	0.22	3.52	6,797,700
LHCC (log, slices, ours)	1.12	0.63	0.69	0.19	2.93	490,895
LHCC (linear, proj, ours)	1.11	0.60	0.66	**0.18**	2.99	490,895
LHCC (log, proj, ours)	**1.03**	**0.57**	**0.64**	**0.18**	**2.72**	490,895

Table 1: Results on SimpleCube++. All results are given in degrees. We used available code to test FFCC and FC4, since the authors did not originally report results for SimpleCube++. Afifi & Brown, CCMNet, and CauNet are shown for Cube or Cube+, which are similar to SimpleCube++

To evaluate the impact of log RGB on a semantic network, we also trained a GoogLeNet on linear and log data [27]. The GoogLeNet achieves excellent performance (with 6M+ parameters), and log inputs work better than linear inputs (even with a network pre-trained on sRGB ImageNet). But it does not match the performance of the smaller LHCC.

One interesting result is that FFCC showed very different performance between 3-fold cross-validation and the train/test set splits for SimpleCube++. The result suggests that FFCC may be easily tunable but not as good at generalizing as the LHCC network.

Gehler-Shi Method	Mean	Med	Tri	25% Best	25% Worst	Parameters
Grey World [4]	6.36	6.28	6.28	2.33	10.58	
Grey World 2OE [29]	5.13	4.44	4.62	2.11	9.26	
Afifi & Brown (SIIE) [1]	2.77	1.93	-	0.55	6.53	1,008,044
Bianco [3]	2.63	1.98	-	-	-	
GoogLeNet [26]	2.55	1.69	1.92	0.39	6.03	6,797,700
CCMNet [16]	2.23	1.53	1.62	0.36	5.46	-
FFCC [2]	1.61	**0.86**	1.02	**0.23**	4.27	61,480
FC4 (SqueezeNet-FC) [14]	1.65	1.18	1.27	0.38	3.78	1,705,284
TLCC [28]	1.51	0.98	1.07	0.33	3.52	6,406,549
CLCC [19]	1.44	0.92	1.04	0.27	3.48	-
C4 (SqNet) [32]	**1.35**	0.88	**0.99**	0.28	**3.21**	1.2M+
GoogLeNet (log, ours)	2.32	1.64	1.79	0.45	5.27	6,797,700
LHCC (pretrained, linear, ours)	2.26	1.70	1.81	0.46	5.15	490,895
LHCC (pretrained, log, ours)	1.96	1.31	1.42	0.36	4.74	490,895

Table 2: Results Gehler-Shi. Our log GoogLeNet results are on a randomly selected 70% training set and 30% test set. All other results are the averages of 3-fold cross-validation.

For Gehler-Shi and NUS-8, due to the small amount of data in each data set, we initialized the LHCC using the pre-trained best network from the SimpleCube++ data set and trained an additional 200 epochs. The pre-trained network matched the network's color space (*e.g.* the log RGB data results used a SimpleCube++ network pre-trained on log RGB data, and linear data results used a SimpleCube++ network pre-trained on linear data).

NUS-8 Method	Mean	Med	Tri	25% Best	25% Worst	Parameters
Grey World [4]	4.59	3.46	3.81	1.16	9.85	-
Grey World 2OE [3]	3.36	2.70	2.80	0.89	7.14	-
CCMNet [16]	2.32	1.71	1.83	0.53	5.18	
FC4 (SqueezeNet-FC) [14]	2.23	1.57	1.72	0.47	5.15	1,705,284
Afifi & Brown (SqNet) [1]	2.05	1.50	-	0.52	4.48	1,008,044
FFCC [2]	1.99	1.31	1.43	**0.35**	4.75	61,480
C4 (SqNet) [32]	1.96	1.42	1.53	0.48	4.40	1.2M+
TLCC [28]	**1.61**	**1.27**	**1.33**	0.44	**3.35**	6,406,549
LHCC (pretrained, linear, ours)	2.09	1.51	1.63	0.49	4.68	490,895
LHCC (pretrained, log, ours)	2.05	1.46	1.58	0.46	4.64	490,895

Table 3: Results on NUS-8. All results are the averages of a geometric mean of a 3-fold cross-validation per-camera.

LHCC returns competitive results for both data sets as seen in tables 2 and 3, with a mean angular error near 2° and a worst 25% error under 5°. Notably, we achieve competitive

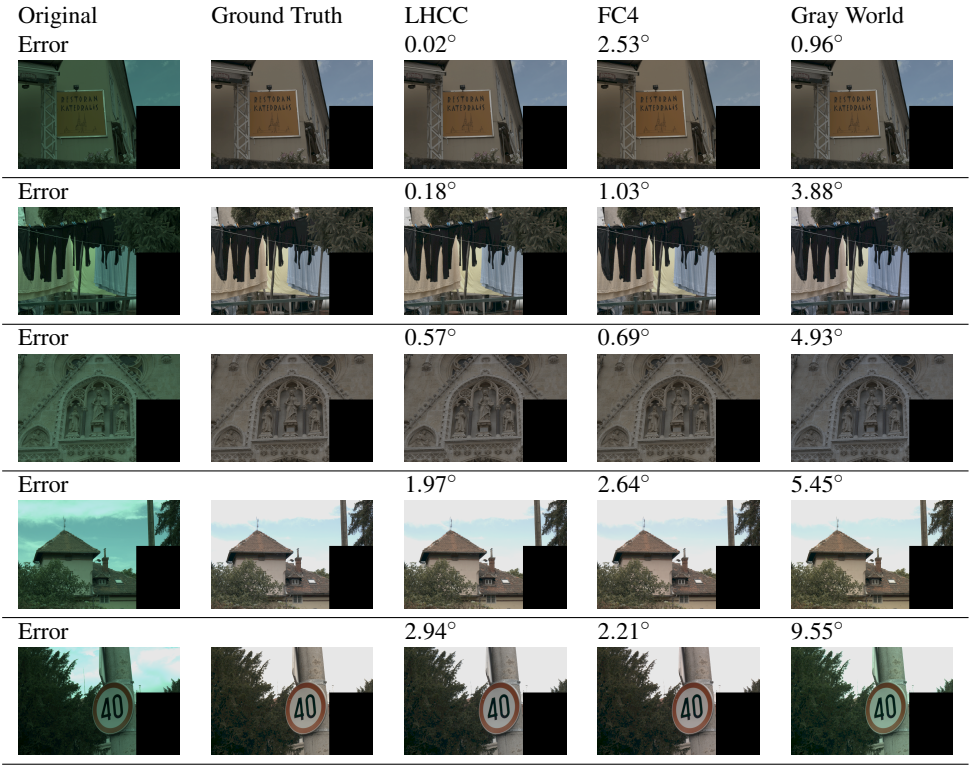

Figure 3: Qualitative results for three images in the SimpleCube++ data set, with angular errors from various models' estimated corrections. Originally linear images are corrected to sRGB for easier visualization.

results with fewer parameters than any model besides FFCC, with half or fewer parameters compared to other state-of-the-art models.

Importantly, across all data sets we see that an LHCC network trained on log RGB data outperforms an equivalent LHCC network that uses linear data, suggesting the value of a log RGB transformation before training.

4.2 Ablation Study

As mentioned in section 3.1, in histograms the difference between bin counts can vary greatly, hiding smaller structures. To address this, we compared five methods of reducing the range of histograms bins: thresholding/clipping the upper bound of the bin counts, a hyperbolic tangent, and a log transformation. We also tested applying no transform as a control (the "linear" test). All tests used the same learning parameters and log RGB input data from SimpleCube++.

Table 4 shows that all of the proposed methods of reducing the range of bin counts improve training when compared to our linear control (without thresholding). The best results by far come from using a log transformation on the bin counts, which results in superior performance to any of the alternative methods.

SimpleCube++ Method	Mean	Med	Tri	25% Best	25% Worst
linear	1.28	0.64	0.74	0.21	3.46
clipped (1023)	1.26	0.68	0.76	0.20	3.41
clipped (255)	1.18	0.64	0.71	0.22	3.09
tanh	1.12	0.68	0.73	0.20	2.89
log	**1.03**	**0.57**	**0.64**	**0.18**	**2.72**

Table 4: Ablation study on normalizing histogram bin values. The values next to "clipped" indicate the maximum allowed bin value, if relevant.

5 Discussion and Summary

In this paper, we present a novel, fully convolutional, histogram-based neural network that uses multiple 2-D projections of the 3-D log RGB histogram of an image to generate state-of-the-art color constancy results.

As seen in tables 1, 2, and 3, both semantic and histogram-based networks using log RGB inputs outperform an equivalent network that learns using linear inputs. This is the first time that result has been demonstrated for the task of color constancy. The LHCC network also outperforms the other two histogram-based networks that use log of chromaticity.

In log space, the reflectance anchors the body reflection cylinder, and illumination change occurs in a fixed direction for all materials. Our results support the hypothesis that giving a network direct access to log RGB data enables it to more effectively learn the relationship between the image or histogram properties and the proper color correction factors.

The benefits of using log RGB data support our second primary result, which is that our simple LHCC network is outperforming much larger models. Compared to networks with more than 2-10x the number of parameters, the LHCC network delivers either state-of-the-art or competitive error metrics at a much smaller size.

Finally, in table 4 we show the importance of details such as using an appropriate saturation transform on the histogram bin counts in order to preserve and highlight smaller details in the histogram. Using a log transform on the bin counts was a significant part of the improvement in performance achieved by LHCC.

Anyone working on color constancy should consider the potential benefits of using the information accessible in log RGB histograms, especially vision researchers and developers working on edge devices with easy access to linear sensor data. If using histograms, they should also carefully consider how to handle the range of values in the histogram buckets to ensure that the potentially large differences in bin counts don't overwhelm the fine histogram structures that represent important information about the image.

6 Acknowledgements

This research was supported in part by the Khoury Teaching Faculty Research Funding Program at Northeastern University.

References

[1] Mahmoud Afifi and Michael S Brown. Sensor-independent illumination estimation for dnn models. In *British Machine Vision Conference (BMVC)*, 2019.

[2] Jonathan T. Barron and Yun-Ta Tsai. Fast fourier color constancy. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6950–6958, Los Alamitos, CA, USA, July 2017. IEEE Computer Society. doi: 10.1109/CVPR.2017.735. URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2017.735.

[3] Simone Bianco, Claudio Cusano, and Raimondo Schettini. Color constancy using cnns. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[4] G. Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin Institute*, 310(1):1–26, 1980. ISSN 0016-0032. doi: https://doi.org/10.1016/0016-0032(80)90058-7. URL https://www.sciencedirect.com/science/article/pii/0016003280900587.

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

[6] Egor Ershov, Alexey Savchik, Illya Semenkov, Nikola Banić, Alexander Belokopytov, Daria Senshina, Karlo Kočević, Marko Subašić, and Sven Lončarić. The cube++ illumination estimation dataset. *IEEE Access*, 8:227511–227527, 2020.

[7] Egor Ershov, Vasily Tesalin, Ivan Ermakov, and Michael S. Brown. Physically-plausible illumination distribution estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12928–12936, October 2023.

[8] G. D. Finlayson, S. D. Hordley, and M. H. Brill. Illumination invariance at a single pixel. In *The 8th Color Imaging Conf.*, Scottsdale, Arizona, 2000.

[9] David Forsyth. A novel algorithm for color constancy. *Int'l J. of Computer Vision*, 5 (1):5–36, 1990.

[10] Brian Funt and Ligeng Zhu. Laplacian of logarithm as illumination-invariant input space. In *Proc. of 30th Color and Imaging Conference*, 2022.

[11] Elena Garces, Adolfo Munoz, Jorge Lopez-Moreno, and Diego Gutierrez. Intrinsic images by clustering. *Computer Graphics Forum*, 31(4):1415–1424, 2012. ISSN 1467-8659. doi: 10.1111/j.1467-8659.2012.03137.x. URL http://dx.doi.org/10.1111/j.1467-8659.2012.03137.x.

[12] P.V. Gehler, C. Rother, A. Blake, T. Minka, and T. Sharp. Bayesian color constancy revisited. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

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

[14] Yuanming Hu, Baoyuan Wang, and Stephen Lin. Fc4: Fully convolutional color constancy with confidence-weighted pooling. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[15] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. doi: 10.1109/CVPR.2014.223.

[16] Dongyoung Kim, Mahmoud Afifi, Dongyun Kim, Michael S. Brown, and Seon Joo Kim. CCMNet: Leveraging calibrated color correction matrices for cross-camera color constancy, 2025. URL http://arxiv.org/abs/2504.07959.

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL https://api.semanticscholar.org/CorpusID:6628106.

[18] Zhihao Li and Zhan Ma. Robust white balance estimation using joint attention and angular loss optimization. In Wolfgang Osten, Dmitry P. Nikolaev, and Jianhong Zhou, editors, *Thirteenth International Conference on Machine Vision*, volume 11605, page 116051E. International Society for Optics and Photonics, SPIE, 2021. doi: 10.1117/12.2586930. URL https://doi.org/10.1117/12.2586930.

[19] Yi-Chen Lo, Chia-Che Chang, Hsuan-Chao Chiu, Yu-Hao Huang, Chia-Ping Chen, Yu-Lin Chang, and Kevin Jou. CLCC: Contrastive Learning for Color Constancy . In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8049–8059, Los Alamitos, CA, USA, June 2021. IEEE Computer Society. doi: 10.1109/CVPR46437.2021.00796. URL https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00796.

[20] Bruce A Maxwell, Richard M Friedhoff, and Casey A Smith. A bi-illuminant dichromatic reflection model for understanding images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[21] Bruce A. Maxwell, Casey A. Smith, Maan Qraitem, Ross Messing, Spencer Whitt, Nicolas Thien, and Richard M. Friedhoff. Real-time physics-based removal of shadows and shading from road surfaces. In *CVPR Workshop on Autonomous Driving*. IEEE / CVF, 2019.

[22] Bruce A. Maxwell, Sumegha Singhania, Heather Fryling, and Haonan Sun. Log rgb images provide invariance to intensity and color balance variation for convolutional networks. In *British Machine Vision Conference*, 2023.

[23] Bruce A. Maxwell, Sumegha Singhania, Avnish Patel, Rahul Kumar, Heather Fryling, Sihan Li, Haonan Sun, Ping He, and Zewen Li. Logarithmic Lenses: Exploring Log RGB Data for Image Classification . In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17470–17479, Los Alamitos, CA, USA, June 2024. IEEE Computer Society. doi: 10.1109/CVPR52733.2024.01654. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52733.2024.01654.

[24] Matteo Rizzo. matteo-rizzo/fc4-pytorch, 2025. URL https://github.com/matteo-rizzo/fc4-pytorch. original-date: 2021-02-16T13:55:19Z.

[25] W. Shi and Brian Funt. Re-processed version of the gehler color constancy dataset of 568 images, 2024. URL http://www.cs.sfu.ca/colour/data/.

[26] Oleksii Sidorov. Artificial color constancy via googlenet with angular loss function. *Applied Artificial Intelligence*, 34(9):643–655, 2020. doi: 10.1080/08839514.2020.1730630. URL https://doi.org/10.1080/08839514.2020.1730630.

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[28] Yuxiang Tang, Xuejing Kang, Chunxiao Li, Zhaowen Lin, and Anlong Ming. Transfer learning for color constancy via statistic perspective. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(2):2361–2369, Jun. 2022. doi: 10.1609/aaai.v36i2.20135. URL https://ojs.aaai.org/index.php/AAAI/article/view/20135.

[29] Joost van de Weijer, Theo Gevers, and Arjan Gijsenij. Edge-based color constancy. *IEEE Transactions on Image Processing*, 16(9):2207–2214, 2007. doi: 10.1109/TIP.2007.901808.

[30] J. von Kries. Chromatic adaptation. *Festschrift der Albrecht-Ludwigs-Universität*, pages 145–158, 1902.

[31] Mengda Xie, Peng Sun, and Yubo Lang. Gray region extension via white patch assumption for color constancy. *Journal of Computer-Aided Design & Computer Graphics*, 36, 2024. ISSN 1003-9775. URL https://www.jcad.cn/en/article/id/7d3c79a7-7e7b-4361-989f-e643c0b2e273.

[32] Huanglin Yu, Ke Chen, Kaiqi Wang, Yanlin Qian, Zhaoxiang Zhang, and Kui Jia. Cascading convolutional color constancy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12725–12732, 2020.