https://doi.org/10.2352/EI.2024.36.16.COLOR-165

Camera Color Correction using Splines

Joni Suominen[1,2] and Karen Egiazarian[2]
[1] Axon Enterprise, Inc; Tampere, Finland
[2] Tampere University; Tampere, Finland

Abstract

Color correction involves converting device-dependent RGB values into a device-independent color space, like XYZ or sRGB. This paper introduces a novel method for deriving a matrix-based transformation using tensor product B-splines, extending previous polynomial approaches. The proposed spline-based model offers enhanced adaptability in color correction, consistently surpassing traditional polynomial methods across various cameras. Splines, being piecewise polynomials, offer increased flexibility in modeling, adapting to different spectral characteristics of cameras. Performance comparison of the spline-based model against previous methods was conducted using simulated data of natural scenes on two different cameras in both training and testing phases.

Introduction

Color correction is an ever-growing area of interest due to advances in capture and display technologies. In the future, it is fair to assume that reproducing the scene as it was initially seen becomes more critical with emerging technologies such as VR and AR. Otherwise, reduced color fidelity might result in diminished immersion. Although most devices only support color spaces with limited gamuts, such as sRGB [1], in the future, we can expect more devices to adopt larger color spaces, such as Rec.2020 [2].

The ability of humans and cameras to sense color is based on photosensitive cells and sensors, respectively. The photosensitive cells, known as cones, are located in the eye's retina. There are three different types of cones, namely long (L), medium (L), and short (S). The naming is related to the ability to sense long (blueish), medium (greenish), and short (reddish) wavelengths. Image sensors commonly mimic this behavior by designing color filter arrays (CFA), such as the Bayer filter, which primarily senses red, green, and blue wavelengths.

The reference point for a camera is the response an average human would perceive for the same stimulus. For this purpose, the International Commission on Illumination (CIE) has published the spectral sensitivity curves for the average observer, from which the responses can be calculated [5]. Typically, the CIE XYZ color space is used, as the spectral sensitivities are all positive, and the Y component corresponds to perceived brightness. Similar functions can be obtained for an imaging sensor in a controlled laboratory environment using a monochromator or a similar device.

Having information about the spectral sensitivity of the average observer, it is possible to compute the responses for an infinite amount of scenes knowing the associated reflectances and illuminants. The response for light incident on the eye, after reflecting from an object is then given by the following:

$$X = \int_{380}^{780} E(\lambda)R(\lambda)S_X(\lambda)\,d\lambda \tag{1a}$$

$$Y = \int_{380}^{780} E(\lambda)R(\lambda)S_Y(\lambda)\,d\lambda \tag{1b}$$

$$Z = \int_{380}^{780} E(\lambda)R(\lambda)S_Z(\lambda)\,d\lambda, \tag{1c}$$

where $E(\lambda)$ is the spectral power distribution (SPD) of the illuminant, $R(\lambda)$ is the spectral reflectance of the object, and $S_T(\lambda)$ is the sensitivity of the tristimulus value T (either X, Y, or Z). Similarly, we can calculate the value recorded by the imaging sensor by replacing cone sensitivities S_X, S_Y, S_Z with the camera sensitivities S_R, S_G, S_B and the responses X, Y, Z with R, G, B.

The need for color correction then arises from differences in spectral sensitivities of standard observer cones and imaging sensors. Figure 1 shows the spectral sensitivities of Nikon D5100 and Sigma SD Merill cameras measured at the National Physics Laboratory [3] compared with the ones of an ordinary observer. They are different and, thus, do not produce the same responses to a given stimulus. Suppose the camera sensitivities cannot be represented as a linear transformation of the cone sensitivities. In that case, it violates the **Luther-Ives** condition [14], [17] and is said not to be **colorimetric**.

The most straightforward and commonly used algorithm for color correction is the simple linear transformation by a 3×3 color correction matrix (CCM). Here, an assumption is made that the camera satisfies the Luther-Ives condition, and acceptable results are often achieved even though the assumption is not valid in practice. Given a set of n corresponding camera and observer responses, the mapping is then defined as follows:

$$Y = MX, \tag{2}$$

where Y and X are matrices of size $3 \times n$, containing the observer and camera responses respectively, and M is the 3×3 color correction matrix.

There are multiple ways to find the CCM, with varying complexity. As the problem is to estimate the corresponding XYZ values given camera-recorded RGB values, it's natural to pose it as a regression problem. The simplest solution is then to find the minimum least-squares solution, which is given by the Moore-Penrose inverse [22] as follows:

$$M = (X^T X)^{-1} X^T Y = X^{\dagger} Y. \tag{3}$$

This formulation is known as the linear color correction model (LCC). It is often used as the initial calibration point for more complex schemes, such as the white-point preserving color correction [8] or saturation and noise balancing [15]. The optimization can also be done in perceptual color spaces like CIELAB [25] using non-linear optimization algorithms, as color spaces like XYZ are not perceptually uniform.

Existing Methods

Given that the Luther-Ives condition is frequently unfulfilled, approaches utilizing non-linearities have been proposed. A common method in regression problems is to transform the features with some non-linear function, such as different degree polynomials. Given an input RGB value, the transformation is the following for a 2nd-degree polynomial model:

$$(R, G, B, R^2, G^2, B^2, RG, RB, GR)^T.$$

The fundamental property of this polynomial color correction (PCC) model [13] is that it is still linear in its coefficients and thus allows us to find a closed-form solution. It also allows us to consider interactions as features for problems. The apparent downside is that the number of coefficients grows with the polynomial degree and that we have to compute the powers of our features at each pixel.

An issue that often prevents the practical application of the PCC is that it is not invariant to exposure. As the exposure or gain of a camera is varied by factor k, the output of a linear camera will also vary by a factor of k. It is easy to see why this is false for a polynomial model: if we scale the input R by factor k, the term will become $(kR)^2$ after applying the polynomial transformation of degree 2, and the function value increases quadratically.

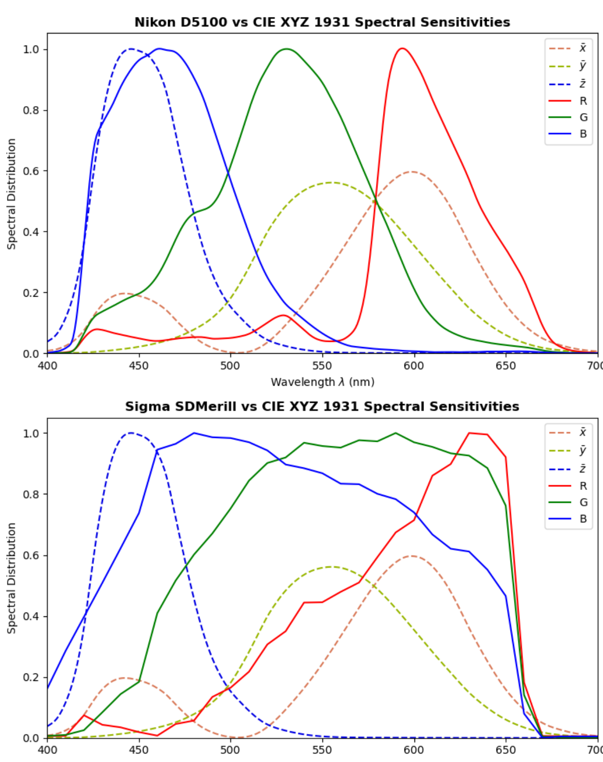

Figure 1. Comparison of Nikon D5100 (Top) and Sigma SDMerill (Bottom) spectral sensitivities with XYZ color matching functions

Non-linear exposure-invariant models have been proposed earlier in the literature, for example, Root Polynomial Color Correction (RPCC) [9], an expansion of PCC. In RPCC, the degree root is taken for all nth-degree polynomial models, resulting in the following terms for a 2nd-degree model:

$$(R, G, B, \sqrt{RG}, \sqrt{RB}, \sqrt{GB})^T.$$

All the features are of degree one, resulting in smoother surfaces, although not necessarily linear. Most importantly, interactions between terms are still considered. The reason for the massive emphasis on interaction terms is that there is an overlap in the spectral domain between channels and corresponding target sensitivities in the XYZ domain, as one can see in Figure 1. Both of the features then contribute to the same output features simultaneously.

Another famous family of color correction methods, which enable non-linear yet exposure-invariant color correction, are multi-dimensional look-up tables (MLUT). Here, a look-up table is first formed between some limited set of known input RGB and XYZ values, and at run-time, the rest of the values are computed using interpolation between the nearest values in the look-up table. This family of models is very flexible, as the transformation can be modified to be more colorimetric by using 3D look-up tables [26] or slightly less accurate yet exposure-invariant by using 2D look-up tables [20]. The downside of this method is the computational cost, as a compromise has to be made on which samples are stored in the look-up table since storing a large table in memory may not be feasible. Furthermore, interpolation has to be performed for each pixel not in the table, for which a trade-off between interpolation algorithm complexity and accuracy also exists.

The past decade has shown immense growth in the processing power of mobile devices. This has allowed us to deploy more data-intensive models, such as deep neural, to enhance images. This has also emphasized the importance of data, as neural networks require large and representative datasets to capture the underlying phenomena without overfitting. Previous regression-based methods in color correction have been modest in their dimensionality. Although higher-order polynomial models have been proposed, being global, they begin to overfit early and introduce ringing artifacts.

Splines for Color Correction

A natural continuum to previous polynomial models is to expand them into piecewise polynomials, namely splines. They allow us to include smooth non-linearities by enforcing continuity and controllability at control points. Here, we present a method to utilize penalized tensor product splines to produce smooth, interpretable surfaces.

Mathematically, B-splines are easy to formalize. All B-spline functions are defined as a linear combination of lower-order splines. Unlike piecewise polynomials, a spline of order M has a degree of $M-1$ due to the continuity condition at the knots. Increasing order of B-splines can then be computed recursively given the first-order spline formula:

$$B_{i,1}(x) = \begin{cases} 1 & \text{if } \tau_i \leq x < \tau_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

for i = 1, ..., K + 2M - 1, where K is the number of knots and τ_i and τ_{i+1} denote the interval over which the function is non-zero. Generally, a spline of order M has support over M knot intervals, and thus, for a first-order spline, we have that the constant function only has support over a single interval. [12, 186-189]

A B-spline of order M is defined by a sum of two shifted order $M-1$ B-splines at the current knot position i and the next knot position $i+1$. The formula is then given by:

$$B_{i,M}(x) = \frac{x-\tau_i}{\tau_{i+M-1}-\tau_i} B_{i,M-1}(x) + \frac{\tau_{i+M}-x}{\tau_{i+M}-\tau_{i+1}} B_{i+1,M-1}(x), \tag{5}$$

as is presented in [12, pp.186-189]. Other formulations exist for computing the B-spline basis functions, such as Cardinal B-Splines, which can be derived through successive convolutions and are very efficient to compute [4, pp. 89, 283].

B-splines have a few desirable properties that make them attractive for color correction. First, for a degree d spline, the formulation imposes a constraint that the derivatives are defined up to degree $d-1$, making them continuous at knot positions and enforcing smoothness. Consequently, this also results in the basis functions becoming smoother as the degree increases, at the cost of each spline taking up a larger domain and increased complexity. In practice, often third-order basis functions are used as a compromise for all things considered [6].

Utilizing interactions between variables, as was done with polynomial models, is also straightforward with splines via tensor products. Assuming we have transformed our red and green channel features using n splines per feature, the interactions would be captured as follows:

$$\mathbf{r} \otimes \mathbf{g} = \mathbf{r}\mathbf{g}^T = \begin{pmatrix} r_1g_1 & r_1g_2 & r_1g_3 \\ r_2g_1 & r_2g_2 & r_2g_3 \\ r_3g_1 & r_3g_2 & r_3g_3 \end{pmatrix}, \tag{6}$$

Where $\mathbf{r}$ and $\mathbf{g}$ are $n \times 1$ vectors containing the values of the n B-spline basis evaluated at the position of the input pixel's intensity level for the red and green channels. All two-way interactions can then be considered using three tensor-product matrices: $\mathbf{r} \otimes \mathbf{g}$, $\mathbf{r} \otimes \mathbf{b}$ and $\mathbf{g} \otimes \mathbf{b}$. Combining these into one matrix forms our design (or basis) matrix for regression.

As one might guess, introducing tensor products to the model increases the dimensionality quadratically. It is well known that high dimensionality might lead to overfitting if the model is too complex for given training data. Often, regularization techniques are used, which impose a penalty on the model's coefficients to limit them from becoming too large, causing the model to become sensitive to small changes in the input.

It is possible that without regularization, the resulting model could have drastic differences in coefficients around some neighborhoods, violating our condition of smoothness. Eiler and Marx proposed a solution to problems of a similar nature by penalizing differences in adjacent coefficients of splines [7]. They appropriately named the method as Penalized Splines (P-splines). The penalty can be applied to the least-squares minimization problem as follows:

$$\|\mathbf{y} - \mathbf{M}\boldsymbol{X}\|^2 + \lambda \|\mathbf{D}\boldsymbol{M}\|^2. \tag{7}$$

When applied to the coefficients $\mathbf{M}$, the matrix $\mathbf{D}$ computes the difference between adjacent coefficients, and by taking the norm, we get a single representative number of the roughness. The term λ is a regularization parameter that helps balance the trade-off between fitting the model closely to the target values and promoting smoothness in the model output. Lower values of λ emphasize the goal of interpolating the target function, while higher values drive the function towards a linear fit.

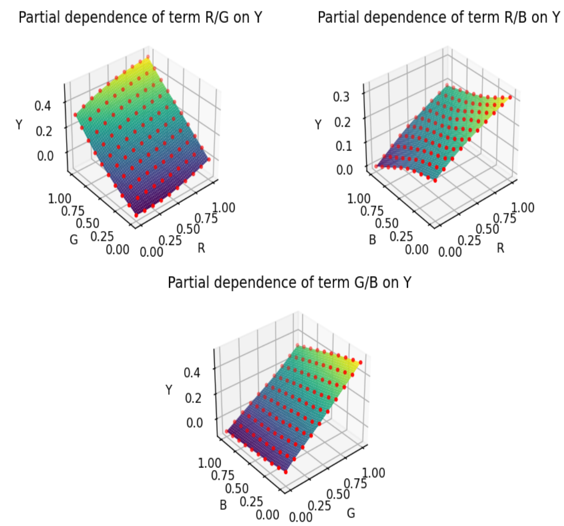

Figure 2. Partial dependencies of different terms on the predicted Y-component

As only two-way interactions are used, it is possible to visualize the partial dependencies of different terms of the fitted model. In Figure 2, the partial dependencies on the Y-predictor can be seen for a model with ten splines per feature and λ of 0.1. Each of the red dots indicates a knot of the tensor product spline. The figures show that all predictors form a smooth surface, as is desired in color correction, yet lots of freedom is available.

Experimental Results

The model performance was evaluated on the two cameras in Figure 1, Nikon D5100 and Sigma SDMerill. These were chosen as they possess very different spectral sensitivities, and one might expect that the transformations from device-dependent RGB values to device-independent XYZ values are quite different.

Dataset

The primary datasets used for training and testing in this study consisted of hyperspectral images of natural scenes. These images were recorded in the Minho Region, Portugal, and were collected by Foster et al. for their work in color constancy [21, 11]. The photos consist of nature and city landscapes, vegetation, and human-made objects, recorded using a hyperspectral camera at 10 nm intervals from 400 to 720 nm. To create a more diverse dataset, we merged the training set with images from the CAVE Multispectral Image Database [27], which consists of common household objects, skin, hair, and so on with varying saturation captured in a laboratory. This resulted in a diverse data set for training, with various reflectance spectra that we may run into daily. The reason for using a large dataset of hyperspectral images instead of, say, color checkers with precisely measured reflectances, is that training strictly on a small set of color patches does not guarantee that the mapping performs well in real-life scenes. Typical colors in real life are not very saturated, which is the opposite of what we see in typical color checker charts. The models should thus be trained and evaluated based on their target application.

The dataset of Foster et al. consisted of 58 hyperspectral images of sizes ranging from 336 × 256 to 819 × 812. As even one image comprised millions of reflectances when considering each pixel, the images were downsampled to 32 × 32 spatially. It is also clear that neighbouring pixels in an image are often very similar as most of the content is low-frequency. So, they do not add significant information during the training process. Similar processing was performed for the CAVE dataset, originally consisting of images of size 512 × 512. This resulted in 39935 samples for training and 204800 samples for testing.

The chromaticity gamut of the training and test sets is shown in Figure 3, with the white triangles displaying the edges of the sRGB color space gamut and red dots displaying the samples. Our training set occupies much of the visible spectrum, even more than the sRGB color space. The training and test sets, both representative, could switch places and produce even better results, but the training data was kept smaller for computational limitations.

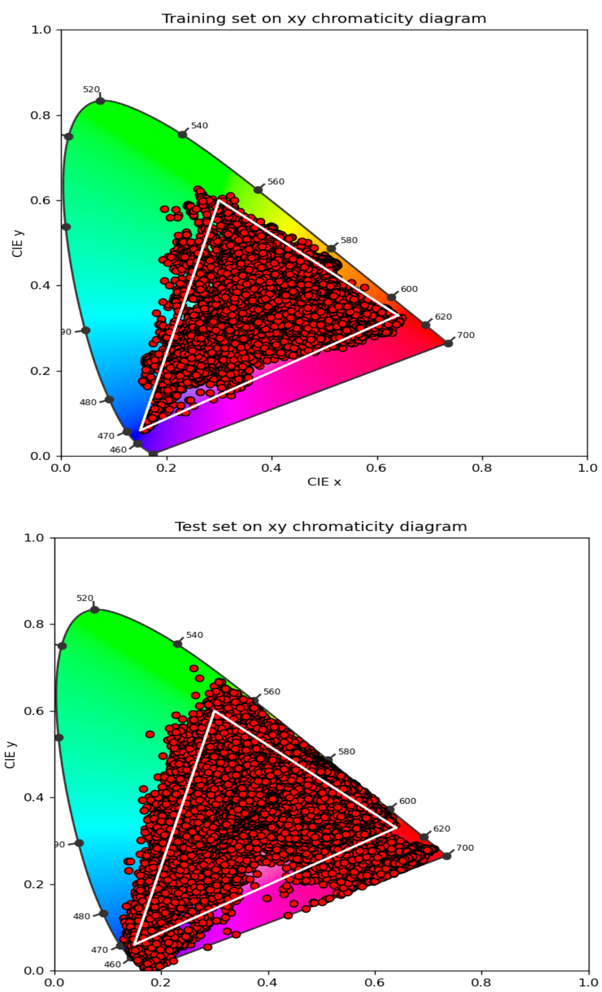

Figure 3. Top: training set. Bottom: test set

Selected Models

For these experimentations, we conducted comparisons against established models in the polynomial family, namely LCC and degrees two and three of PCC and RPCC. Higher-order polynomial models were not considered because they are known to be too prone to overfitting. Neural networks and MLUTs were also not considered in this paper, as the focus was kept on the expansion of polynomial models.

For the P-spline model, the smoothing parameter λ was found by hyperparameter tuning on the training dataset, and the best model was then retrained on the complete training dataset before testing. Other parameters that could have been tuned were the order of the spline basis functions and the number of them per input channel. As for the order, it was concluded that the 3rd order splines would produce a good compromise of smoothness, computational complexity, and support.

In prior research on P-splines, it has been emphasized that better performance can always be achieved using more spline basis functions as long as the smoothing parameter is chosen appropriately [6, p. 3]; we decided to present results at 5, 10, and 20 basis functions per input channel. The complexity of the resulting model can easily be computed: for each of the three output channels, three tensor product features are used, resulting in the total $3 \times 3 \times n^2 = 9n^2$ terms, where n is the number of basis functions per feature. The models then have 225, 900, and 3600 parameters respectively.

Results

The results are divided into two tables according to the camera model. For both cameras, the RGB responses were computed by numerical integration from 400 nm to 700 nm at 10 nm intervals under **D65** illuminant. Before finding the color correction matrix, white balance was applied by dividing each channel by response to a perfect reflecting diffuser (PRD), resulting in an input domain of $[0,1]$. The exposure is then equal to the situation where the image sensor would be just short of clipping for the PRD. Numerical integration was again performed to obtain the XYZ responses for modeling the human eye. Since it is often convenient to have the Y channel, corresponding to luminance, be treated as a percentage, all three channels were normalized by the Y channel response to a PRD so that Y has a $[0,1]$ range.

The mean, max, 95%, and 99% errors for the Nikon D5100 are seen in Table 1. The numbers 5, 10, and 20 after the name "P-splines" indicate the number of spline basis functions, while the numbers for LCC, PCC, and RPCC indicate the used polynomial order. Models ending with "LAB" (e.g., LCC-LAB) were trained using the CIEDE2000 [24] objective function within the CIELAB color space. Since a closed-form solution does not exist, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) [10] algorithm was used to find the best solution.

Our P-spline models lead in performance across all categories, albeit by a narrow margin. The mean error is under the just noticeable difference (JND, 1 ΔE), while typically, even the 95th percentile errors are around 2 ΔE. Less surprisingly, the model with the least spline features has the lowest maximum error, as with lower flexibility, the model is more likely to capture mean characteristics across control points.

Surprisingly, both LCC models produced results that were even with the more complex models. One hypothesis is that the transformation is close to linear. Due to their global nature, the more complex polynomial features might overshoot or undershoot the data at a mean level. Spline models can consider this by controlling the fit locally but are computationally much higher in demand than the basic LCC model. From a complexity-accuracy point of view, LCC will likely be the best choice for this camera model.

Algorithm	Mean	Max	95%	99%
P-Splines-5	0.94	**4.81**	2.08	2.48
P-Splines-10	0.86	5.16	1.94	2.45
P-Splines-20	**0.83**	5.77	**1.92**	**2.42**
LCC	1.00	5.56	2.18	2.93
LCC-LAB	0.94	5.51	2.04	2.66
PCC-2	1.13	5.58	2.43	3.05
PCC-3	1.07	5.40	2.37	2.98
PCC-3-LAB	0.91	7.07	2.03	2.49
RPCC-2	1.05	5.54	2.44	3.06
RPCC-3	1.06	5.69	2.43	3.00
RPCC-3-LAB	0.92	8.04	2.22	3.04

Table 1: CIEDE2000 Errors for Nikon D5100

The second set of experiments was conducted on Sigma SD Merrill due to its significantly different spectral sensitivities. Here, transforming from RGB to XYZ responses is much more complex, and we see much higher errors, especially in the outlier performance (Max, 95th, and 99th). Similar to the Nikon results, spline models beat most competing polynomial models. Surprisingly, the 3rd-order PCC models perform comparably to the spline models, with its maximum error being half that of the 3rd-order RPCC model. A hypothesis for the excellent performance is that the spectral sensitivities are relatively flat across the domain, and the RPCC is not flexible enough for such a complex transform.

Conclusion

In this paper, we proposed a novel method for color correction that can be seen as an extension of the previous polynomial models. Although narrowly, we noticed that the model produced the best results on two very different cameras. In contrast, for the other models, we saw that the rank order can change depending on the spectral characteristics. The flexibility of spline models can explain this, as the transformation can be tuned locally rather than globally.

A valid point of criticism for the proposed model is its increased computational complexity. The dimensionality grows quadratically with the number of spline basis functions per feature, so a compromise must be made on model accuracy versus performance. For example, the model with five splines has 225 coefficients in total, and in comparison, the model with 20 splines has 3600 coefficients. On the other hand, computing splines is simple, and the coefficients can be applied by a simple matrix multiplication. The model is thus similar in computational complexity to neural networks for color correction [18, 16].

Algorithm	Mean	Max	95%	99%
P-Splines-5	1.77	18.27	4.14	**5.90**
P-Splines-10	1.74	17.74	4.03	5.92
P-Splines-20	**1.73**	**16.95**	**4.00**	5.94
LCC	2.20	23.49	6.98	10.56
LCC-LAB	2.57	19.60	5.59	7.84
PCC-2	2.02	22.07	5.82	9.20
PCC-3	1.92	20.78	5.01	8.16
PCC-3-LAB	1.98	17.92	4.42	6.38
RPCC-2	2.17	25.38	6.84	10.04
RPCC-3	2.32	34.32	6.90	11.37
RPCC-3-LAB	2.02	43.49	5.55	9.71

Table 2: CIEDE2000 Errors for Sigma SDMerrill

Despite our model performing well, there is still room to improve. Here, we proposed a simple model based on uniform tensor product splines, but non-uniform or shifted knot placement could increase the performance depending on the transformation. We also optimized purely in XYZ color space, but further improvements will likely be obtained by fine-tuning in a perceptually uniform color space.

An important topic, exposure invariance, was not discussed in this work. We predict that the model we used here for comparison is not entirely exposure invariant, but a higher amount of penalization would likely achieve better results across exposure changes. This is because higher penalties drive the fitted surface towards a hyperplane. The penalty we applied here was equal for all input features. Still, an elegant way to impose a constraint on exposure invariance would be to use a higher penalty on control points near the achromatic axis.

We make the code and dataset publicly available at `https://github.com/JoniSuominen/SplineColorCorrection`.

We thank the authors of **pyGAM** [23] for their fantastic implementation of generalized additive models (GAMs), which we used extensively in this paper. In addition, we are grateful to the contributors of the Python package **Colour** [19] for their work in producing an amazing Python implementation of typical color science functions.

References

[1] Matthew Anderson, Ricardo Motta, Srinivasan Chandrasekar, et al. Proposal for a standard default color space for the internet—srgb. In *Color and imaging conference*, volume 1996, pages 238–245. Society for Imaging Science and Technology, 1996.

[2] ITU-R Recommendation BT et al. Parameter values for ultra-high definition television systems for production and international programme exchange. 2012.

[3] Maryam Darrodi, Graham Finlayson, Teresa Goodman, et al. Reference data set for camera spectral sensitivity estimation. *JOSA A*, 32, 03 2015.

[4] Carl. De Boor. *A practical guide to splines*. Applied mathematical

sciences ; 27. Springer, New York, rev. ed. edition, 2001.
[5] Commission Internationale de l'Éclairage. Commission internationale de l'eclairage proceedings, 1931.
[6] Paul Eilers and Brian Marx. Why p-splines? 6 2021. Retrieved from the authors' personal website.
[7] Paul HC Eilers and Brian D Marx. Flexible smoothing with b-splines and penalties. *Statistical science*, 11(2):89–121, 1996.
[8] Graham D Finlayson and Mark S Drew. White-point preserving color correction. In *Color Imaging Conference*, pages 258–261. Citeseer, 1997.
[9] Graham D Finlayson, Michal Mackiewicz, and Anya Hurlbert. Color correction using root-polynomial regression. *IEEE Transactions on Image Processing*, 24(5):1460–1470, 2015.
[10] Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2000.
[11] David H Foster and Adam Reeves. Colour constancy failures expected in colourful environments. *Proceedings of the Royal Society B*, 289(1967):20212483, 2022.
[12] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, New York, second edition edition, 2009.
[13] Guowei Hong, M Ronnier Luo, and Peter A Rhodes. A study of digital camera colorimetric characterization based on polynomial modeling. *Color Research & Application: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, 26(1):76–84, 2001.
[14] Herbert E. Ives. The transformation of color-mixture equations from one system to another. *Journal of the Franklin Institute*, 180(6):673–701, 1915.
[15] Brian Keelan, Robin Jenkin, and Elaine Jin. Quality versus color saturation and noise. *Proceedings of SPIE - The International Society for Optical Engineering*, 8299:14–, 01 2012.
[16] Abdullah Kucuk, Graham D Finlayson, Rafal Mantiuk, et al. An exposure invariant neural network for colour correction. In *Color and Imaging Conference*, volume 30, pages 176–181. Society for Imaging Science and Technology, 2022.
[17] R. Luther. Aus dem gebiet der farbreizmetrik. *Zeitschrift fur Technische Physik*, pages 540–558, 1927.
[18] Lindsay MacDonald and Katarina Mayer. Camera colour correction using neural networks. In *London Imaging Meeting*, volume 2021, pages 54–57. Society for Imaging Science and Technology, 2021.
[19] Thomas Mansencal, Michael Mauderer, Michael Parsons, et al. Colour 0.4.4, 12 2023.
[20] Jon S McElvain and Walter Gish. Camera color correction using two-dimensional transforms. In *Color and Imaging Conference*, volume 21, pages 250–256. Society for Imaging Science and Technology, 2013.
[21] Sérgio MC Nascimento, Flávio P Ferreira, and David H Foster. Statistics of spatial cone-excitation ratios in natural scenes. *JOSA A*, 19(8):1484–1490, 2002.
[22] Roger Penrose. A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press, 1955.
[23] Daniel Servén and Charlie Brummitt. pygam: Generalized additive models in python. *Zenodo. doi*, 10, 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: Endorsed by Inter-Society Color Council, The Colour Group (Great Britain), Canadian Society for Color, Color Science Association of Japan, Dutch Society for the Study of Color, The Swedish Colour Centre Foundation, Colour Society of Australia, Centre Français de la Couleur*, 30(1):21–30, 2005.
[25] CIE Standard et al. Colorimetry-part 4: Cie 1976 l* a* b* colour space. *International Standard*, 2007.
[26] Michael J. Vrhel. Use of simulated reflectance spectra in camera transform creation. *Color and Imaging Conference*, 24(1):306–306, 2016.
[27] Fumihito Yasuma, Tomoo Mitsunaga, Daisuke Iso, et al. Generalized assorted pixel camera: postcapture control of resolution, dynamic range, and spectrum. *IEEE transactions on image processing*, 19(9):2241–2253, 2010.

Author Biography

Joni Suominen received his B. Eng. from Tampere University of Applied Sciences in 2022 and is currently pursuing an MSc in Signal Processing at Tampere University. Along with his studies, he is also working for the imaging team at Axon in Tampere, Finland, where his work focuses on imaging software and sensor characterization.

Karen Egiazarian (Eguiazarian) received the M.Sc. degree in mathematics from Yerevan State University, Yerevan, Armenia, in 1981, the Ph.D. degree in physics and mathematics from Moscow State University, Moscow, Russia, in 1986, and the Doctor of Technology degree from the Tampere University of Technology (TUT), Tampere, Finland, in 1994. He is a Professor with the Signal Processing Department, Tampere University, Tampere, leading the Computational Imaging Group, and a Docent with the Department of Information Technology, University of Jyväskylä, Jyväskylä, Finland. His main research interests are in the fields of computational imaging, compressed sensing, efficient signal-processing algorithms, image/video restoration, and image compression. He has published more than 750 refereed journal articles and conference papers, books, and patents in these fields. Prof. Egiazarian was the Editor-in-Chief of the Journal of Electronic Imaging, and recipient of Service Award, IS&T, 2014