Determine RGBW LED PWM from CIE Chromaticity

CHINZEI, Kiyoyuki

June 20, 2020

Abstract

This article is a private note to develop kch–rgbw–lib, available in github and npm. kch–rgbw–lib is written in typescript, its main functions include color space conversions between HSV, RGB, XYZ and xyY, and calculation of color mixing. This article gives a general solution of multi-number (more than RGB) LEDs to represent different colors used in kch–rgbw–lib. It is not intended to carry new, accurate, or most efficient ideas. This document is granted under MIT License.

1 Define the Problem

1.1 Past works

Obtaining accurate color by mixing RGB color sources has been utilized as color displays since 1950s. As various colors are available by LED, recent topics are solution to expand additional colors or light source typically for OLED applications [1, 2]. Usually while light sources are used as an additional light source [3, 1, 2]. For display purposes colors other than RGB have been also used to expand the possible color ranges[4]. Sharp once added yellow in Aquos flat-panel TV, but they researched 5-primary color display [5]. Amber [6], turquoise, and violet can be other colors to expand the gamut.

1.2 Given parameters and assumptions

We have $n \geq 3$ color sources (LEDs) with chromaticity (x_i, y_i) and maximum luminosity Y_i, where $i = 1 \ldots n$. Our problem is to find the optimum PWM output $\boldsymbol{\alpha} = [\alpha_1 \ldots \alpha_n]^T$ where $0 \leq \alpha_i \leq 1$ to represent a given color input with chromaticity (x, y) and luminosity Y.

Here, we set our goal of optimization as the following:

1. Minimize the error of color to represent,
2. If (x, y) is outside the gamut of (x_i, y_i), nearest color in the gamut is used,

3. When possible, minimize energy consumption,
4. When possible, set α_i to null when it's very small,
5. Under the physical constraint $Y \leq \alpha_1 Y_1 + \ldots + \alpha_n Y_n$.

- Condition 2 can be achieved by projecting the input to the contour of the gamut.
- Condition 3, energy consumption is obtained as

$$E = \sum_1^n \alpha_i W_i \tag{1.2.1}$$

  where W_i is the power (W) of each LED at the maximum luminosity Y_i.

- Condition 4 is preferable to avoid gitter of low PWM output, and high sensitivity of human eyes against such gitter.

It is a typical linear programming (LP) problem. When $n = 3$, e.g. R-G-B color sources only, it's a deterministic and not an optimization problem. And when $n = 4$, e.g. R-G-B-W LEDs, there is only one parameter to optimize, which makes the problem as simple as we don't need to use sophisticated LP solver. We first derive a general description of the problem and solve it for $n = 3$, $n = 4$ and $n \geq 5$ cases.

1.3 Description of problem

Composite of color source in XYZ color space (X, Y, Z) can be obtained as a simple sum of each term. Therefore we use XYZ color space. In XYZ color space, our problem is to determine $\boldsymbol{\alpha} = [\alpha_1 \ldots \alpha_n]$ which gives an equation between input color $[X, Y, Z]^T$ and color source $[X_i, Y_i, Z_i]^T, i = 1 \ldots n$;

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \alpha_1 \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \end{bmatrix} + \ldots + \alpha_n \begin{bmatrix} X_n \\ Y_n \\ Z_n \end{bmatrix} \tag{1.3.2}$$

Color space $[X, Y, Z]^T$ is expressed by using (x_i, y_i, Y_i):

$$X_i = \frac{x_i}{y_i} Y_i \tag{1.3.3}$$

$$Y_i = Y_i \tag{1.3.4}$$

$$Z_i = \frac{1 - x_i - y_i}{y_i} Y_i \tag{1.3.5}$$

Using matrix representation, Eq. 1.3.2 is written as

$$[\boldsymbol{X}] = [\boldsymbol{A}]\,[\boldsymbol{\alpha}] \tag{1.3.6}$$

where

$$
\begin{aligned}
[\boldsymbol{X}] &= [X, Y, Z]^T && (1.3.7)\\
[\boldsymbol{A}] &= \begin{bmatrix} \frac{x_1}{y_1}Y_1 & \dots & \frac{x_n}{y_n}Y_n \\ Y_1 & \dots & Y_n \\ \frac{1-x_1-y_1}{y_1}Y_1 & \dots & \frac{1-x_n-y_n}{y_n}Y_n \end{bmatrix} && (1.3.8)\\
[\boldsymbol{\alpha}] &= [\alpha_1, \ \dots \ , \alpha_n]^T && (1.3.9)
\end{aligned}
$$

Our goal is to solve Eq. 1.3.6 for $\boldsymbol{\alpha} = [\alpha_1 \dots \alpha_n]^T$. To solve it, $\boldsymbol{A}^{-1}$, the pseudo-inverse matrix of $\boldsymbol{A}$ is obtained by the singular value decomposition (SVD) (Eq. 1.3.10) [7].

$$
\begin{bmatrix} \boldsymbol{A} \end{bmatrix} = \begin{bmatrix} \boldsymbol{U} \end{bmatrix} \begin{bmatrix} \omega_1 & & & \\ & \omega_2 & & 0 \dots 0 \\ & & \omega_3 & \end{bmatrix} \begin{bmatrix} \boldsymbol{V}^T \end{bmatrix} \tag{1.3.10}
$$

where $\boldsymbol{A}$ is $3 \times n$, $\boldsymbol{U}$ is 3×3, $[\omega_1 \ddots \omega_3, 0 \dots 0]$ is $3 \times n$, $\boldsymbol{V}^T$ is $n \times n$ matrixes In this specific case, since $\boldsymbol{A}$ is a $3 \times n$ matrix, there are upto 3 ω's. When $n \geq 4$, $[\omega_1 \ddots \omega_3]$ is null-padded in $3 \times (n-3)$. $\boldsymbol{A}^{-1}$ is obtained by

$$
\begin{bmatrix} \boldsymbol{A} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{V}_{1-3} \end{bmatrix} \begin{bmatrix} 1/\omega_1 & & \\ & 1/\omega_2 & \\ & & 1/\omega_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{U}^T \end{bmatrix} \tag{1.3.11}
$$

where $\boldsymbol{V}_{1-3}$ is the first 3 columns of $\boldsymbol{V}$, those correspond to $\omega_1 \dots \omega_3$. Using $\boldsymbol{A}^{-1}$, one obtains

$$
[\boldsymbol{\alpha}] = [\boldsymbol{A}]^{-1} [\boldsymbol{X}] \tag{1.3.12}
$$

By the way, what about the rest of columns in $\boldsymbol{V}$? They are null vectors of $\boldsymbol{A}$. A null vector $\boldsymbol{n}$ of $\boldsymbol{A}$ is such vector that satisfies $\boldsymbol{A}\boldsymbol{n} = [0]$. By denoting these columns as $\boldsymbol{n}_4 \dots \boldsymbol{n}_n$, Eq. 1.3.12 can be extended as

$$
[\boldsymbol{\alpha}] = [\boldsymbol{A}]^{-1} [\boldsymbol{X}] + \beta_4 \boldsymbol{n}_4 + \dots + \beta_n \boldsymbol{n}_n \tag{1.3.13}
$$

where $\beta_4 \dots \beta_n$ are arbitrary numbers. By choosing these using other constraint, one can obtain the optimum solutions.

2 Solution of $n = 3$ case

When $n = 3$, Eq. 1.3.12 gives a deterministic solution. No optimization. However, one should be careful that the obtained α_i are physically meaningful, i.e., $0 \leq \alpha_i \leq 1$. This can happen when the input color $[X]$ is out of the gamut defined by the color sources. When $\alpha_i < 0$, it should be truncated to 0. In this case, the color has certain error.

When $\alpha_i > 1$, All αs should be normalized by the largest α. This way the color will be correctly obtained, but it will be darker than expected.

3 Solution of $n = 4$ case

When $n = 4$, Eq. 1.3.13 is modified as

$$[\boldsymbol{\alpha}] = [\boldsymbol{A}]^{-1}[\boldsymbol{X}] + \beta_4 \boldsymbol{n}_4 \tag{3.0.1}$$

Parameter β_4 will be determined by the assumptions in section 1.2. Solving for β (hereafter omitting '4'), we obtain the following conditions.

$$\beta \geq -\frac{b_i}{n_i} \quad i = 1 \ldots 4, \text{ if } n \neq 0 \tag{3.0.2}$$

$$\beta \leq \frac{1 - b_i}{n_i} \quad i = 1 \ldots 4, \text{ if } n \neq 0 \tag{3.0.3}$$

$$E = \sum_1^4 (\beta n_i + b_i) W_i \to min \tag{3.0.4}$$

where $[b_1, \ldots, b_4]^T = [\boldsymbol{A}]^{-1}[\boldsymbol{X}]$, n_i are the elements of $\boldsymbol{n}$. Eq. 3.0.4 is from Eq. 1.2.1. Finding the largest and smallest values of the right hand side of Eqs. 3.0.2 and 3.0.3, denoted as β and β_{max}, Eqs. 3.0.2 and 3.0.3 are rewritten as

$$\beta_{min} \leq \beta \leq \beta_{max} \tag{3.0.5}$$

Since Eq. 3.0.4 is rewritten as $E = s_1\beta + s_2$, here s_1 and s_2 are constants determined by calculating the sums in Eq. 3.0.4, optimized β is determined as

$$\beta = \begin{cases} \beta_{min} & \text{if } s_1 = \sum_0^4 n_i W_i > 0 \\ \beta_{max} & \text{else} \end{cases} \tag{3.0.6}$$

To implement the assumption 4 in section 1.2, you introduce allowance of small α, α_ε, and exchange Eq. 3.0.2 as

$$\beta \geq \frac{\alpha_\varepsilon - b_i}{n_i} \tag{3.0.7}$$

But this may result in $\beta_{min} > \beta_{max}$, where no possible answer found. When this happens, you need to relax the constraint by α_ε. The easiest is to go back to Eq. 3.0.2.

4 Solution of $n > 4$ case

You need to optimize Eq. 1.2.1 under constraints of $0 \leq \alpha_i \leq 1$ and Eq. 1.3.13 using a linear programming solution. We will implement it in future.

References

[1] Chi Can and Ian Underwood. A compact and efficient method of RGB to RGBW data conversion for oled microdisplays. In *EURODISPLAY 2011*, pages 59–62, 2011.

[2] Chul Lee and Vishal Monga. Power-constrained RGB-to-RGBW conversion for emissive displays. In *IEEE International Conference on Acoustic, Speech and Signal Processing*, pages 1214–1218, 2014.

[3] Brian Tompson, Stephen Allen, and Microchip Technology Inc. High resolution RGB LED color mixing application note. Technical Report AN1562, Microchip Technology Inc., 2014.

[4] Wikipedia contributors. Multi-primary color display, 2020. [Online; accessed 14-June-2020].

[5] 冨沢 一成. 多原色ディスプレイの意義. In **フラットパネルディスプレイの人間工学シンポジウム** *2011*. JEITA, March 2011.

[6] Swathi Sridhar, Ashutosh Tiwari, Namrata Dalvi, and Microchip Technology Inc. RGBA color mixing with bluetooth ® low energy communication. Technical Report AN2026, Microchip Technology Inc., 2016.

[7] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Singular Value Decomposition*, chapter 2.6, pages 59–70. Cambridge University Press, Cambridge, MA, USA, second edition, 1992.

MIT License