Color Mixer

Tianbo Xiong
Department of Computer Science
Georgia Institute of Technology
txiong33@gatech.edu

Xiaowen Ma
Department of Computer Science
Georgia Institute of Technology
xma335@gatech.edu

ABSTRACT:
Color is a fundamental aspect of human perception, playing a pivotal role in both natural processes and human design. Humans have developed and utilized color theory as a guide to create various colors and achieve different visual effects. For computational purposes, colors are often encoded in different mathematical models for computers to compute, then render the result to screen for the colors that humans can perceive. However, we recognize that color theory, while rich and insightful, can be daunting for many individuals due to its complexity. Similarly, existing computational models are robust but lack user-friendliness. Our proposal seeks to develop an accessible and intuitive visualization tool for the color mixing process.

Keywords: *color model, color theory, color model application, visualization, digital painting.*

INTRODUCTION:
In the world of graphical design and digital artistry, artistic design tools such as Photoshop, Adobe Illustrator are indispensable for creating stunning visuals. However, most tools utilize the existing mechanism such as a color panel which primarily facilitates the overlaying of colors with different opacities, restricting the creative freedom of users and leaving out the process of color mixing. On the other hand, there are also various color models such as RGB and CMYK that different programs adopted. These mechanisms are used for calculating the result color, which could be pretty dissimilar to what real color would look like. As an example, the conventional blending of RGB colors differs significantly from the process of mixing colors for traditional paintings, as it involves the mixing of light rather than the mixing of pigments. This disparity can often lead to results that are misleading for users who are not familiar with the intricacies of different color-mixing mechanisms. While some models attempt to achieve color addition by adjusting transparency[4], manually changing opacity and overlapping colors remains an unintuitive and inconsistent process when compared to the way humans perceive color mixing. In contrast, traditional artists and painters employ a step-by-step approach to craft a vast spectrum of colors. With a palette of fundamental colors of their choice, they choose a color and add a drop from the palette, stir, and perhaps try it on paper. They need to observe how the color changes and adjust the proportions until they achieve the exact shade desired. Both proportion and opacity matter in this natural process and this process does not require the understanding of the computational models, but achieves the goal by only using visual effects.

Unfortunately, there is not a definitive applicable working model that actually reflects the complexity of “real-world color mixing”. The process of obtaining color from a palette proves to be considerably more intricate than the basic color theory taught in school. However, there are some novel techniques developed in recent years that can better mimic color mixing in the real world using the Kubelka-Munk theory to predict realistic color behavior[19][6].

We want to make both the procedure and result interactive so that users can better visualize how the colors are mixed from different mixing models and understand what the resulting colors are and how they are related on a gamut. Our project will thus offer a hands-on simulation of the color mixing process, allowing users to actively experiment with color blending, rather than simply selecting colors and adjusting the parameters. In this project, we will first study the 3 different color models, and then we propose a color mixer project that provides an interactive, side-by-side visualization using different models, and shows the mixture history as well as the

transition on the color gamut. This project could be both a design and an educational tool, and the project will allow users to be more interactive and engaging on our digital platform, and to utilize the tool to tackle similar challenges in their physical settings.

BACKGROUND AND RELATIVE WORK:

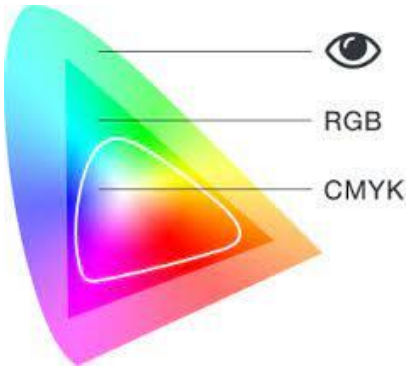

Figure 1.1: the color region of RGB and CMYK

Color mixing needs to mimic how it would in real life for it to function smoothly in digital painting tools. The interaction between incident light and pigment particles within the paint is what gives the paint we see as it comes out of the tube its color. Each wavelength of light is selectively absorbed and scattered by the pigment particles in a particular way. Pigments have unique colors due to differences in their absorption and scattering characteristics.

Various models have been proposed for color mixing[18]. Perhaps the most basic mixing models are the additive color model for digital arts and the subtractive color model for painting. We will then introduce the Kubelka-Munk model that better mimics color mixing in real life.

Before delving into the intricacies of different color mixing modes, it is important to establish a foundational understanding of the concept of gamut.

Gamut:

A gamut refers to the complete range or scope of something. In digital and print design, a color gamut is the range of colors that can be achieved using different color combinations like CMYK and RGB[14]. An RGB color gamut is much larger than a CMYK color gamut, meaning you can achieve many more colors in RGB than you can in CMYK. This is why a design may look great on screen, but when printed it becomes muddied and dark. Some colors cannot be reproduced using CMYK, some colors will lose their vibrancy when converted from CMYK to RGB.

Additive Color Model:

The Additive Color Model is a color model that describes how light produces color. The additive mixing predicts the appearance of color made by coincidence of component lights, which means the perceived color can be predicted by summing the numeric representation of the component light. Additive colors start with black, and add three additive colors(red, blue, green) light to produce the visible spectrum of colors. The color mixing model is predominately used in digital devices such as cameras, televisions, phones and computer monitors.

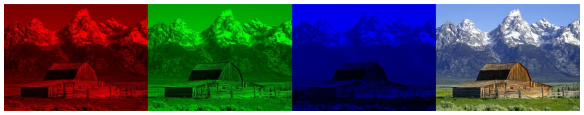

Figure 1.2 Additive Color Mixing[2]

In digital devices, an electrical charge activates a red, green, or blue element making them glow. We refer to these components as sub-pixels. One pixel can provide the desired hue by blending the three colors. After that, the pixels are assembled into tiny mosaics to produce a picture. When all three colors are combined equally, the result is the white light. PPI (Pixels Per Inch) is therefore the unit of measurement for digital graphics. Here are two ways to implement the additive model:

```
first_way:
new Color =
(min(Color1.r+Color2.r,
255),min(Color1.g+Color2.g,
255),min(Color 1b+Color 2.b, 255))

second_way:
new              Color              =
Color1*Color1.volume/(Color1.volume+C
olor2.volume)+Color2*Color2.volume(Co
lor1.volume+Color2.volume)
```

The first method simply sums of the input color with a constraint to prevent the color channel value from overflowing beyond the limit. The second method results in a blend where the contribution of each color

is proportional to its volume. For this project, we adopted the second approach to simulate the resulting color by blending two colors with unequal volumes. This approach can be interpreted as "Additive-Averaging Mixing" [9], according to David Briggs. Compared to simple additive mixing, which corresponds to the first implementation, the averaging method intermixes the incoming components over the total area, resulting in an intermediate brightness that is between the brightest and dimmest input colors.

Subtractive Color Model:

Additive color alone does not predict the appearance of mixtures of printed color inks, dye layers in color photographs or paint mixtures. Instead, it adheres to subtractive mixing principles. In subtractive mixing, the color model predicts the spectral power distribution of light after it passes through successive layers of partially absorbing media. Thus, the primary colors are those that absorb red, green, and blue light, specifically cyan, magenta, and yellow, respectively. Each layer partially absorbs some wavelengths of light from the illumination spectrum while letting others pass through, resulting in a colored appearance.

Figure 1.3 Subtractive Color Mixing[2]

A relationship was sought between the concentrations of cyan, magenta and yellow inkjet printouts and the screen XYZ values, based on the connection between additive and subtractive primaries[5]. A fourth color, black (K, which stands for key) is added to make four-color printing (CMYK). Due to imperfections in those ink colors, if we simply employed cyan, magenta, and yellow to generate black, we would obtain a brownish hue. The addition of density to the shadows and the neutralization of images and designs by the black ink. Such a model is essential for color printing and photography as dyes and pigments are simulated using such a model.

Though CMYK is known as a subtractive model that is widely used in printed works, it is hard to implement "color mixing" with this model. In this project, we used RGB to implement a subtractive model to compare the differences between various models. There are also two ways to implement subtractive color mixing algorithm:

```
The first way:
new Color = (Color1*Color2)/255

The second way:
new Color = 255- SQRT(((255-Color1)^2
+(255-Color2)^2)/2)
```

The first mixing algorithm used reverse-bayes formula, while the second algorithm allowed the system to be more diluting. The second algorithm will produce dark gray instead of black. We used reverse-bayes formula to calculate the resulting color .We decided not to consider volume for the implementation of the subtractive color mixing model.

Kubelka–Munk Model :

However, in contrast to how light interacts with actual pigment mixtures, where both absorption and scattering occur, subtractive mixing solely takes into account the absorption of light. As a result, the subtractive mixture seems darker and has a different color than in real life[19]. In the initial stages of exploration to simulate pigment mixture in real life, D.R. Duncan[7], like other practitioners, assumed that within a pigment mixture, the resultant colors in a specific medium could be determined through formulas incorporating two constants per pigment. These constants, contingent upon the wavelength of the incident light, represent the pigment's absorbing power for light and its scattering power.

The Kubelka-Munk model[22] is a fundamental approach to modeling the appearance of paint films. The model, published in a paper[16] called "An Article on Optics of Paint Layers" by two German scientists Paul Kubelka and Franz Munk, addresses the question of how the color of a substrate is changed by the application of a coat of paint of specified composition and thickness, and especially the thickness of paint needed to obscure the substrate[18]. The model involves just two paint-dependent constants and uses a two-stream approximation for light diffusing through a coating whose absorption and remission coefficients are known. Their work provides a valuable systematic methodology for color mixing matching, by solving

the Kubelka-Munk equation, one can determine the absorption to scatter ration, or "remission function"[10][8]:

$$F(R_\infty) \equiv \frac{(1-R_\infty)^2}{2R_\infty} = \frac{a_0}{r_0}.$$

We can also define K and S, the absorption and back-scattering coefficients respectively to replace the absorption and remission fractions a_0, r_0 from the equation above. If we then assume the separate additivity of absorption and back-scattering coefficients for each concentration component c_i, where i represent the individual components, the equation takes on the form outlined below:

$$\frac{(1-R_\infty)^2}{2R_\infty} = \frac{a_0}{r_0} = \frac{K}{S} = \frac{\Sigma(c_i K_i)}{\Sigma(c_i S_i)} \approx \frac{\Sigma(c_i K_i)}{S}.$$

The model would then return the reflectance of light for a layer of a pigment on a surface Unlike both additive or subtractive color mixing, where colors tend to lose saturation when mixed, the Kubelka-Munk model is more complicated to implement since it requires more terms, but it mimics the color mixing in real life[17][21].

Figure 1.4 Kubelka-Munk model by Mixbox[19]

In 2021, Sochorová and Jamriška ingeniously applied the Kubelka–Munk paint-mixing algorithm directly to the RGB color model. Their innovative "Mixbox" methodology involves the conversion of inputs into a modified CMYK format (utilizing phthalo blue, quinacridone magenta, Hansa yellow, and titanium white) alongside a residue component to address gamut variations. The K–M mixing process is then executed within this latent space before the final output is generated in RGB[19](see Figure 1.4). We use the Mixbox extension on Python for the Kubelka-Munk model implementation in our case.

Color Model Comparison:

As different color models utilize different algorithms, they may also have different resulting colors and effects. The computational differences and transformation have been studied[1][15], however, there lacks a way to perceive the difference.

PROJECT IMPLEMENTATION:

Framework:

In this project, we used Kivy as our main GUI framework. Below are the dependencies other than kivy we use in this project: color-science, numpy, matplotlib and backend_kivyagg from the kivy garden sub-module.

Layout:

There are two main views in the app, a main view for users to mix and visualize the colors as well as mixing history, and a side view for color transformation visualization on a gamut (Figure 2.1).

Color Mixer View:

The Mixer View on the left is the main interface where users interact with the app. The users are given a default color palette (bottom-left) with white, black, red, green, blue, cyan, magenta and yellow. The user can add custom colors to the palette by adjusting r, g, b values (Figure 2.2) or remove any undesired color from the palette (Figure 2.3). A reset button is provided for users to revert the current palette to the default palette.

By clicking or holding on a color, the volume in the pipet will be increased; the user can then transform the color into the mixing bucket (top-left). Three buttons are provided for the user to toggle between the history view and mixing view, to reset the current result and to save the current bucket color to the palette.

Color History View:

The Color History View (Figure 2.4) shows the current blending history as equations of input colors, input volumes and output colors. The color history is automatically updated when the bucket color is recalculated. If the current bucket is empty, the bucket will be updated with the current palette color; otherwise, the bucket will be updated by a new mixture and cumulative volume, and a record of the

current mixing process will be added to the history view. For example, in the second row, 1 unit of blue is mixed with 1 unit of cyan, resulting in another blue with r, g, b, a = 128, 255, 255, 255, and the current volume of the bucket would be 1 + 1 = 2.

Currently, the history view only shows the result of the additive color model, and it only works with opaque color-blending calculation. The r, g, b, a values are presented for comparison with the results from the subtractive model and the K-M model.

Gamut View:

The Gamut View (Figure 2.5) will map and display each of the intermediate colors as a circle on the gamut, which provides an observable visualization of color transition on the complex gamut. The gamut mapping will be cleared after resetting the current bucket canvas. Currently, only additive colors are mapped and displayed on the gamut to avoid confusing users with various color models.

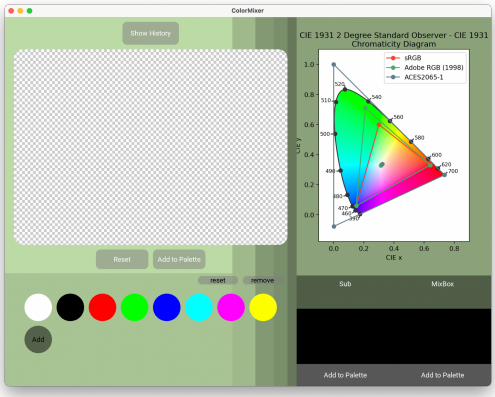

Figure 2.1: General App Layout.

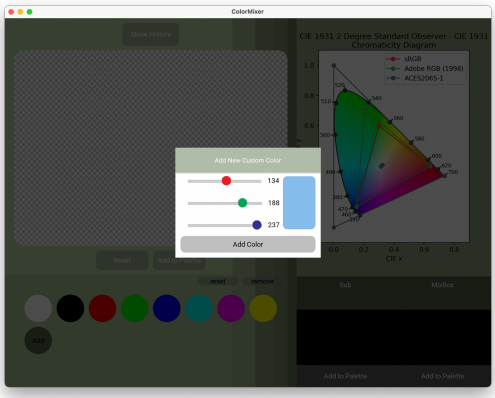

Figure 2.2: Customized pop up window that allows users to pick a desired color by adjusting the R,G,B sliders separatively.

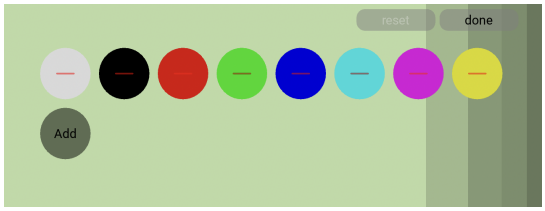

Figure 2.3: Remove mode is turned on for users to remove unwanted color from the current palette.

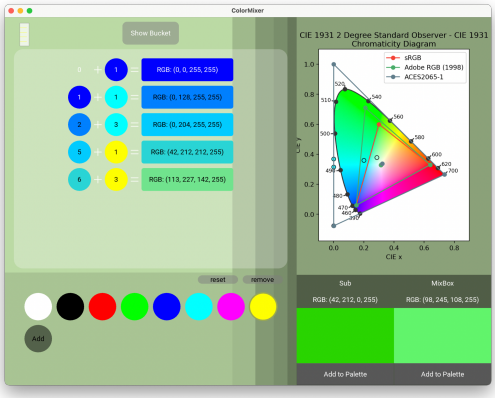

Figure 2.4: Color History View

Input:

Default Palette:

Users will be given eight default colors, including additive colors (red, green, blue) and subtractive colors (cyan, magenta, yellow, black) as well as white, simulating the simple situation artists are in when they start to draw. We include cyan, magenta and yellow to avoid creating only black with the default R, G, B palette when using the subtractive model.

Custom Color:

Users can also save their blended color to the palette, with a maximum number of 20. By clicking on the “Add” button in the color palette, a window for the user to add custom colors will pop up. Users can customize the color by adjusting the r, g, b sliders, on a scale from 0 to 255, respectively.

Functionality:

Step-by-Step Simulation and Real-Time Visualization:

The color mixer will simulate the color blending procedure with adjustable volumes of colors. Users are able to adjust the volume absorbed into the pipet and the volume released into the bucket.

History View:

The color mixer app will provide the user with the color blending history in a hierarchical structure (Figure 2.2). The color history function will allow users to visualize how a particular color is made from the basic color palette and it will also allow users to view the specific parameters for the color. This function is useful for both design and educational purposes[12].

Color Metrics Transformation:

For each input, in-progress, and output color, a panel with detailed metrics computed by 3 different methods (additive, subtractive, Kubelka–Munk) will be shown. The difference between these three models can be visualized.

Gamut Mapping:

After calculating and displaying the resulting colors, we will calculate and display the transition states on the gamut. For this project, we will try to adopt the traditional algorithm[11][14] for RGB gamut mapping[20], which helps users visualize the change and understand the mechanism of the gamut.

User Interaction:

Single Click:

a) Upon clicking on a color widget in the palette, the pipet will be updated to the current color; each click will add one unit of current color to the pipet;

b) Upon clicking on the color bucket, the entire volume of the current color will be added to the current bucket; in the callback function, the three color models are called to compute the updated color.

Tap and Hold:

a) Tap and hold on a color in the palette will increase the amount of color that is absorbed into the pipet or released into the bucket.

Gesture Recognition:

a) Swipe from right to left on the bucket in "mixing mode" to switch to the history view;

b) Swipe from left to right on the history view to switch to the "mixing mode".

RESULTS AND COMPARISON:

For this project, we aim to provide different visualizations for users to see the resulting colors and to learn about color gamut, as well as the different color models through an intuitive process. In the following example, we blended Red (255, 0, 0, 255), Green (0, 255, 0, 255) and Blue (0, 0, 255, 255) with a ratio of 1:1:1, and the resulting colors calculated by the three models are shown below(Figure 3.1.).

The additive model gives a result of gray color instead of white, which is consistent with the algorithm described in the previous section. The Kubelka-Munk model results in a similar but warmer gray. However, the subtractive model would generate the color black.

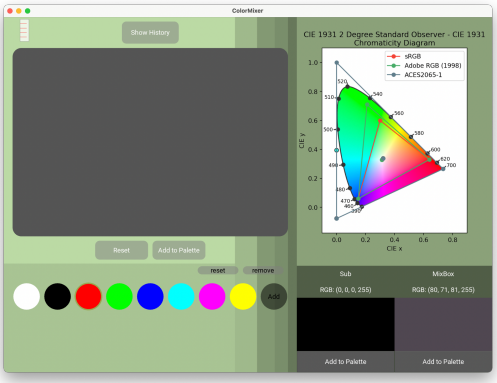

Figure 3.1: Difference between additive, subtractive and Kubelka-Munk color models.

To compare additive and subtractive models, we did another example by blending Cyan and Yellow by a 1:1 ratio (Figure 3.2). As shown below, the additive color model generates a white-ish green color, while the subtractive mode results in a pure green.

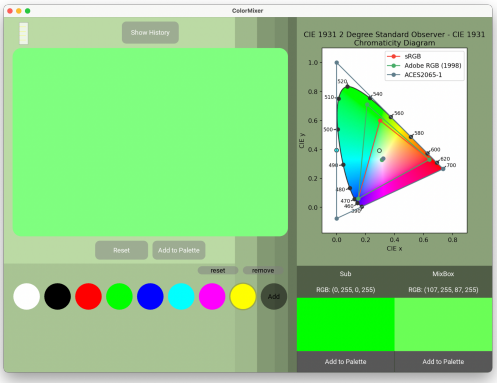

Figure 3.2: Difference between additive and subtractive color models.

LIMITATIONS AND FUTURE WORK:

Our current implementation only works with opaque color mixing, where the alpha value is always 255 on a scale of 0 to 255. For additive color models, we consider changing the alpha value as analogous to adding water to blended pigments in real life; we can add another button for alpha blending. Another limitation is that Mixbox does not support alpha blending since it only takes into account the homogenous mixing of opaque paints. The model treats both RGB layers as if they consisted of thick wet paint when compositing two RGB layers using alpha-blending. We could possibly extend this by implementing the model ourselves, and add support for the Kubelka-Munk layer-compositing model, which would allow for a more accurate simulation of transparent layers and handle effects like watercolor.

Another limitation is that, in our current implementation, we recalculate the updated bucket color and consider the current bucket color as a fixed, in-progress color; thus each individual step would be the mixture of two colors, and the entire history is composed of multiple 2-color blending subprocess, and it holds for all three models to match the blending history equations. However, for the Kubelka-Munk model, calculation based on the entire history would result in a different result and with current implementation, the result of the K-M model is dependent on mixing order. To calculate blended color based on the entire history in each step using the K-M model, we would need to re-design the history widget and layout to make it work for different models and growing sizes.

In considering further enhancements, we could enhance the interactivity of the application by exploring extending the origin function. Currently, users are constrained to working solely with the mixing color on the designated canvas. To introduce a new color into the workflow, users must save the current color, reset the main canvas, and then introduce the desired color for further interaction. This procedural approach is not only cumbersome but also lacks the intuitive fluidity as if to real-life color mixing using a palette. Furthermore, there is room for refinement in the representation of historical records on the chromaticity diagram. Although the current 2D representations suffice, it is noteworthy that color-science submodules enable the visualization of 3D gamut models, presenting an opportunity for enhanced depth and insight (see graph 4.1).

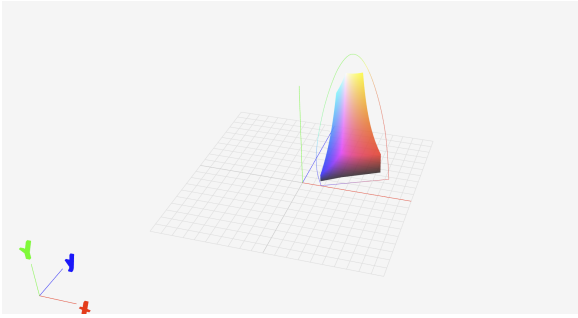

Figure 4.1:3D Gamut using color-science with 3JS.

Throughout the implementation phase, our attempts to incorporate RGB color mapping into the original function were met with several challenges. The existing function is incompatible with RGB color mapping, and the model exhibited lag when operated with the Kivy-supported backend provided by kivy garden library. The model is built using 3JS and Kivy is also grounded in OpenGL. We could potentially solve the issue by extending the original function to seamlessly integrate the interactive gamut history model into the Kivy application. This integration would be further enhanced by recording the model's history with animation, contingent on the successful implementation of the backend. For instance, once a new color is mixed, a new circle will be generated just like cell division, where the circle will move to the new color mapping on the 3D space. Such implementation would make it easier for users to track the latest color generated by the algorithm[3].

Another function that could significantly enhance user experience is the incorporation of a dedicated canvas where users can actively mix colors using a pen or stylus. This feature not only aligns more closely with real-life color mixing practices but also provides users with a hands-on and immersive experience. By allowing users to directly interact with a canvas using a pen, we create a more intuitive and artistically expressive environment, mirroring the traditional process of mixing colors on a physical palette. This addition not only caters to users seeking a more authentic color-mixing experience but also opens up opportunities for enhanced creativity and experimentation within the digital platform.

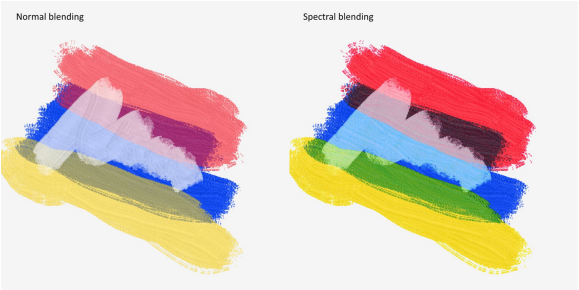

Figure 4.2:Color Mixing using pen with different blending method

CONCLUSION:

In conclusion, our Color Mixer project represents a significant step towards enhancing the accessibility and user-friendly nature of computational color models. By exploring and implementing three distinct color models: Additive, Subtractive, and Kubelka–Munk. We have developed an interactive and educational tool that empowers users to experiment with color blending in a hands-on simulation.

Our application provides real-time mixing in the mixer view, blending history tracking in the color history view, and color transition visualization on a complex gamut in the gamut view. While our project has made some notable progress, there are areas for improvement, such as incorporating alpha blending for transparent color mixing. The Color Mixer not only serves as a design tool but also as an educational resource, allowing users to gain insights into color science through an engaging digital platform. As we move forward, we aim to refine and expand the Color Mixer, offering a powerful and intuitive tool that can reach a broader audience.

REFERENCES:

[1] Alvy Ray Smith. 1978. Color gamut transform pairs. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques (SIGGRAPH '78)*. Association for Computing Machinery, New York, NY, USA, 12–19.

[2] Anon. Additive & subtractive color models. Retrieved November 24, 2023 from https://pavilion.dinfos.edu/Article/Article/2355687/additive-subtractive-color-models.

[3] Bay-Wei Chang and David Ungar. 1993. Animation: from cartoons to the user interface. In Proceedings of the 6th annual ACM symposium on User interface software and technology (UIST '93). Association for Computing Machinery, New York, NY, USA, 45–55. https://doi.org/10.1145/168642.168647

[4] Beck, J. Additive and subtractive color mixture in color transparency. *Perception & Psychophysics* 23, 265–267 (1978).

[5] C J Hawkyard. 1993. Synthetic reflectance curves by subtractive color mixing. *Journal of the Society of Dyers and Colourists* 109, 7–8: 246–251. http://doi.org/10.1111/j.1478-4408.1993.tb01568.x

[6] Chet S. Haase and Gary W. Meyer. 1992. Modeling pigmented materials for realistic image synthesis. *ACM Trans. Graph.* 11, 4 (Oct. 1992), 305–335.

[7] D R Duncan. 1940. The color of pigment mixtures. Proceedings of the Physical Society 52, 3: 390–401. http://doi.org/10.1088/0959-5309/52/3/310

[8] Daniel W. Dichter. 2023. Kubelka-Munk model of full-gamut oil colour mixing - AIC. *Kubelka-Munk model of full-gamut oil colour mixing*. Retrieved December 2, 2023 from https://aic-color.org/resources/Documents/jaic_v32_06.pdf

[9] David Briggs. 2007. Modern colour theory for traditional and digital painting media. *The Dimensions of Colour, modern colour theory*. Retrieved December 2, 2023 from http://www.huevaluechroma.com/

[10] Deane B. Judd. 1952. Color in business science and industry. *Art Education* 5, 4: 18. http://doi.org/10.2307/3183943

[11] Gamut Mapping-RGB http://www.brucelindbloom.com/index.html?Eqn_RGB_to_XYZ.html https://facelessuser.github.io/coloraide/gamut/

[12] Ghita Jalal, Nolwenn Maudet, and Wendy E. Mackay. 2015. Color Portraits: From Color Picking to Interacting with Color. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 4207–4216.

[13] Ibraheem, Noor A., Mokhtar M. Hasan, Rafiqul Z. Khan, and Pramod K. Mishra. "Understanding color models: a review." *ARPN Journal of science and technology* 2, no. 3 (2012): 265-275.

[14] Maureen C. Stone, William B. Cowan, and John C. Beatty. 1988. Color gamut mapping and the printing of Digital Color Images. *ACM Transactions on Graphics* 7, 4: 249–292. http://doi.org/10.1145/46165.48045

[15] Michael W. Schwarz, William B. Cowan, and John C. Beatty. 1987. An experimental comparison of RGB, YIQ, LAB, HSV, and opponent color models. *ACM Trans. Graph.* 6, 2 (April 1987), 123–158.

[16] P. Kubelka, and F. Munk, (1931) *Ein Beitrag Zur Optik Der Farbanstriche. Z. Techn. Phys.*, 12, 593 -601.

[17] Sarah Abraham and Donald Fussell. 2014. Smoke Brush. *In Proceedings of the Workshop on Non-Photorealistic Animation and Rendering (NPAR '14)*. Association for Computing Machinery, New York, NY, USA, 5–11. https://doi.org/10.1145/2630397.2630404

[18] Simonot, L. and Hébert, M. 2013. Between additive and subtractive color mixings: Intermediate mixing models. *Journal of the Optical Society of America* A 31, 1, 58.

[19] Sochorová, Š. and Jamriška, O. (2021) 'Practical pigment mixing for digital painting', *ACM Transactions on Graphics*, 40(6), pp. 1–11.

[20] Song Gang, Li Yihui and Li Hua, "A gamut extension algorithm based on RGB space for wide-gamut displays," *2011 IEEE 13th International Conference on Communication Technology*, Jinan, China, 2011, pp. 743-746.

[21] T. Ma, W.M. Johnston, and A. Koran. 1987. The color accuracy of the Kubelka-munk theory for various colorants in maxillofacial prosthetic material. *Journal of Dental Research* 66, 9: 1438–1444. http://doi.org/10.1177/00220345870660090601

[22] von Traubenberg, H.R. (1921) 'Ein Beitrag zur Kenntnis der Bremsung von α-Strahlen durch Elemente und Verbindungen', *Zeitschrift für Physik*, 5(5–6), pp. 396–403.