S. B. JAIN INSTITUTE OF TECHNOLOGY, MANAGEMENT & RESEARCH, NAGPUR

Session: 2025-2026 ODD

Project Based Learning Report
On

"RGB to LAB Color Conversion for Digital Painting Tools"

Subject: - Image Processing

Semester/Year: III/V

Details:

Sr. No	Name of Students	Problem Understanding & Objective[2M]	Implementation & Functionality[4 M]	Output Quality & Analysis[2 M]	Report/Presentation Quality[2 M]	Total [10M]
1.	Ashwini K. Barapatre					

Date of Submission:

1. ABSTRACT

In digital painting and image processing, accurate color representation is essential for achieving visually consistent and realistic artwork. Most digital painting tools rely on the RGB color model, which is device-dependent and does not accurately reflect human visual perception. This creates challenges in maintaining color uniformity and consistency across different devices and editing environments.

To address this problem, the project focuses on the conversion of images from RGB to LAB color space, where the LAB model is device-independent and perceptually uniform. The conversion process involves two major steps—first transforming RGB to XYZ color space and then converting XYZ to LAB using the standard CIE formulas. This transformation allows for more natural and accurate color adjustments, particularly in lightness, tone, and contrast, without distorting the hue or saturation.

2. INTRODUCTION

Currently, the need to use digital images in everyday life is increasing. These digital images are recorded through various devices such as digital cameras, cell phone cameras, and scanners. The need for digital images is widely used to make copies of documents stored and back up physical files into digital documents. Digital images that have been obtained can also be further processed, such as in the process of recognition or detection of characters hand geometry detection and face recognition for the authentication process. However, the resulting digital images sometimes have relatively similar foregrounds and backgrounds, so they experience difficulties when further processing. Therefore, object segmentation is needed to separate the object from the background. Segmentation is a process to separate an object from the background so that the object can be processed for further purposes.

Digital images that have been obtained can also be further processed, such as in the process of recognition or detection of characters, hand geometry detection, and face recognition for the authentication process. However, the resulting digital images sometimes have relatively similar foregrounds and backgrounds, so they experience difficulties when further processing.

Researchers and color scientists developed the CIE LAB (or simply LAB) color space, which is device-independent and designed to be perceptually uniform. Introduced by the *International Commission on Illumination (CIE)* in 1976, the LAB model separates color information into three components:

- **L** represents lightness, ranging from black to white.
- **a** represents the green–red axis.
- **b** represents the blue–yellow axis.

3. AIM AND OBJECTIVES

Aim:

To study and implement the RGB to LAB color conversion process for achieving perceptually accurate and device-independent color representation in digital painting and image processing tools.

Objectives:

1. To understand the theoretical concepts and differences between RGB and LAB color models.
2. To analyze and implement the mathematical conversion process from RGB → XYZ → LAB.
3. To evaluate the advantages of LAB color space in digital painting applications, focusing on improved color accuracy, tone adjustment, and artistic control.

4. PROBLEM STATEMENT

Most digital painting and image editing tools rely on the RGB color model, which is device-dependent and does not align with human visual perception. This results in inconsistent color appearance, difficulty in precise tone adjustment, and loss of color balance across different devices and editing environments. To address this issue, it is essential to implement RGB to LAB color conversion, as LAB is perceptually uniform and device-independent, allowing for accurate, consistent, and visually realistic color manipulation in digital painting and image processing workflows.

5. LITERATURE REVIEW

Study	Year	Focus	Key Findings
Zhang (2023)	2023	Color space conversion toolbox	Developed a MATLAB toolbox for RGB to LAB conversion, emphasizing perceptual uniformity in color representation.
Sharma (2021)	2021	OpenCV LAB color space applications	Discussed the advantages of LAB color space in image processing, highlighting its perceptual uniformity and applications in color segmentation.
Patel and Joshi (2019)	2019	Digital painting color correction	Applied LAB color conversion for tone correction and artistic color grading, achieving improved realism and device-independent results in digital artwork.

Tan et al. (2018)	2018	Palette-based image decomposition	Introduced a framework for color composition using LAB color space, enabling efficient color harmonization in digital images.
Reinhard et al. (2001)	2001	Color transfer between images	Introduced a method using LAB color statistics for transferring color styles between images, demonstrating LAB's strength in perceptual color editing.
Luo et al. (2006)	2006	CIE color difference evaluation	Refined the ΔE*ab formula to improve perceptual accuracy in LAB color space, supporting its reliability in digital image analysis.

6. PROBLEM DEFINITION & METHODOLOGY

Input:

- A digital image I_{RGB}in **RGB color space**, where each pixel $P(x, y) = [R, G, B]$and $R, G, B \in [0,255]$.

Output:

- A converted image I_{LAB}in **LAB color space**, where each pixel $P'(x, y) = [L^*, a^*, b^*]$, with:
  - $L^* \in [0,100]$
  - a^*, b^*scaled appropriately for storage/display.

Constraints & Assumptions:

1. Input image uses sRGB with a D65 reference white.
2. Conversion must maintain perceptual uniformity.
3. Output L* values range from 0–100; a*, b* values can be scaled appropriately.
4. Conversion should be efficient for practical use in digital painting tools.

II. COLOR CONVERSION METHODOLOGY

The color conversion method is discussed in details by Reinhard et al[9]. The color conversion from RGB color model to ℓαβ color space is done in three steps. In first step, the conversion between RGB to LMS space is done using **Eq. 1**.

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.3811 & 0.5783 & 0.0402 \\ 0.1967 & 0.7244 & 0.0782 \\ 0.0241 & 0.1288 & 0.8444 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

In second step, LMS is converted to **LMS** color space by taking the logarithmic values of them as shown in **Eq. 2 to Eq. 4**.

$$\mathbf{L}=\log(L) \quad (2)$$
$$\mathbf{M}=\log(M) \quad (3)$$
$$\mathbf{S}=\log(S) \quad (4)$$

Finally, the ℓαβ color space is achieved using another conversion from **LMS** space using **Eq. 3**.

$$\begin{bmatrix} \ell \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 0.5774 & 0.5774 & 0.5774 \\ 0.4082 & 0.4082 & -0.8165 \\ 0.7071 & -0.7071 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{L} \\ \boldsymbol{M} \\ \boldsymbol{S} \end{bmatrix} \quad (5)$$

Now, to ease the designing of the system architecture for this color conversion, **Eq. 1** is decomposed into smaller equations which are shown in **Eq. 6 to Eq. 17**.

$L_1=0.3811\times R$	(6)	$L_2=0.5783\times G$	(7)	$L_3=0.0402\times B$	(8)	$L=L_1+L_2+L_3$	(9)
$M_1=0.1967\times R$	(10)	$M_2=0.7244\times G$	(11)	$M_3=0.0782\times B$	(12)	$M=M_1+M_2+M_3$	(13)
$S_1=0.0241\times R$	(14)	$S_2=0.1288\times G$	(15)	$S_3=0.8444\times B$	(16)	$S=S_1+S_2+S_3$	(17)

Further, **Eq. 5** is decomposed into smaller equations as shown in **Eq. 18 to Eq. 25**.

$T_1=\mathbf{L}+\mathbf{M}$	(18)	$T_2=T_1+\mathbf{S}$	(19)	$\ell=0.5774\times T_2$	(20)
$T_3=0.4082\times T_1$	(21)	$T_4=0.8165\times \mathbf{S}$	(22)	$\alpha=T_3-T_4$	(23)
$T_5=\mathbf{L}-\mathbf{M}$	(24)	$\beta=0.7071\times T_5$	(25)		

7. MATHEMATICAL EQUATION/ MODELING

Step 1: RGB to XYZ Conversion

First, normalize the RGB values to the range [0, 1]:

$$R_n = \frac{R}{255}, \quad G_n = \frac{G}{255}, \quad B_n = \frac{B}{255}$$

Then apply the **inverse gamma correction** for sRGB:

$$C_{lin} = \begin{cases} \dfrac{C_n}{12.92}, & C_n \leq 0.04045 \\ \left(\dfrac{C_n + 0.055}{1.055}\right)^{2.4}, & C_n > 0.04045 \end{cases}$$

where $C \in \{R, G, B\}$.

Now convert the **linear RGB** values to **CIE XYZ** using the D65 reference white:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{bmatrix} \begin{bmatrix} R_{lin} \\ G_{lin} \\ B_{lin} \end{bmatrix}$$

Step 2: XYZ to LAB Conversion

Normalize the XYZ values using the **D65 reference white point**:

$$X_n = 95.047, \quad Y_n = 100.000, \quad Z_n = 108.883$$

$$X_r = \frac{X}{X_n}, \quad Y_r = \frac{Y}{Y_n}, \quad Z_r = \frac{Z}{Z_n}$$

Apply the nonlinear function:

$$f(t) = \begin{cases} t^{1/3}, & t > 0.008856 \\ 7.787t + \frac{16}{116}, & t \leq 0.008856 \end{cases}$$

Now compute **L*, a*, and b*:**

$$L^* = 116\, f(Y_r) - 16$$

$$a^* = 500\,[\, f(X_r) - f(Y_r)\,]$$

$$b^* = 200\,[\, f(Y_r) - f(Z_r)\,]$$

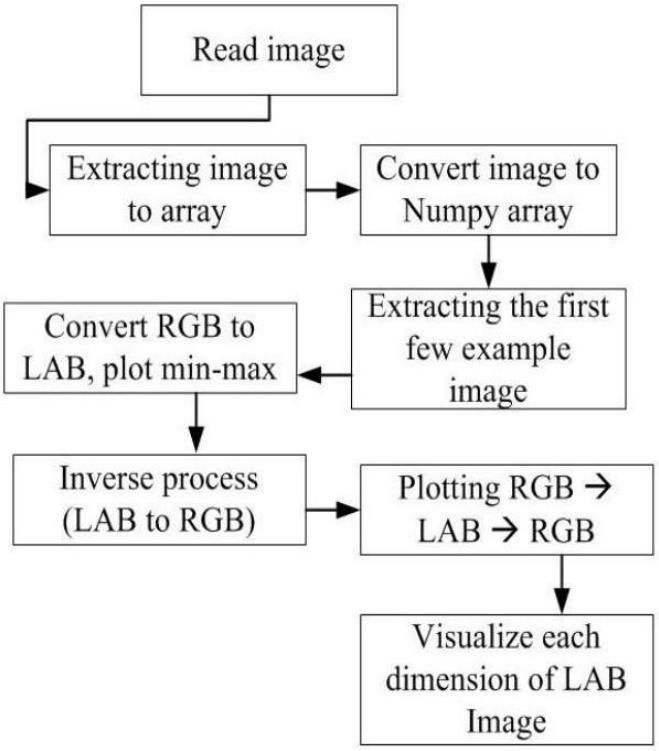
8. PSEUDOCODE/ALGORITHM

Implemented Algorithm:

Steps:

1. Read Image Data: Load the example images for processing.
2. Store Images: Read multiple images (e.g., five) and save them into a list for batch processing.
3. Convert to Numpy Array: Convert each image into a NumPy array for numerical operations.
4. Extract Color Layers:
    - RGB images have three channels: Red, Green, Blue.
    - Each channel has values from 0 to 255.
    - A pixel is black if all channels are 0; the absence of color is indicated if the channel value is 0.
5. Convert RGB → LAB:
    - Convert images from sRGB to CIE LAB color space under the given illuminant and observer settings.
    - Dimension ranges in skimage.color.rgb2lab and lab2rgb:
        - RGB → LAB: [0,1] x [0,1] x [0,1] → [0,100] x [-128,128] x [-128,128]
        - LAB → RGB: [0,100] x [-128,128] x [-128,128] → [0,1] x [0,1] x [0,1]
6. Inverse Conversion (LAB → RGB):
    - Use lab2rgb to convert back to RGB.
    - Ensure the resulting array has three channels (height, width, 3).
7. Validation:
    - Check the correctness of the RGB → LAB → RGB conversion by plotting the images.
8. Visualization of LAB Channels:
    - The 0th dimension (L*) represents brightness (lightness).
    - All three dimensions (L*, a*, b*) can be visualized separately to analyze lightness, green–red, and blue–yellow components.

Flow Chart:



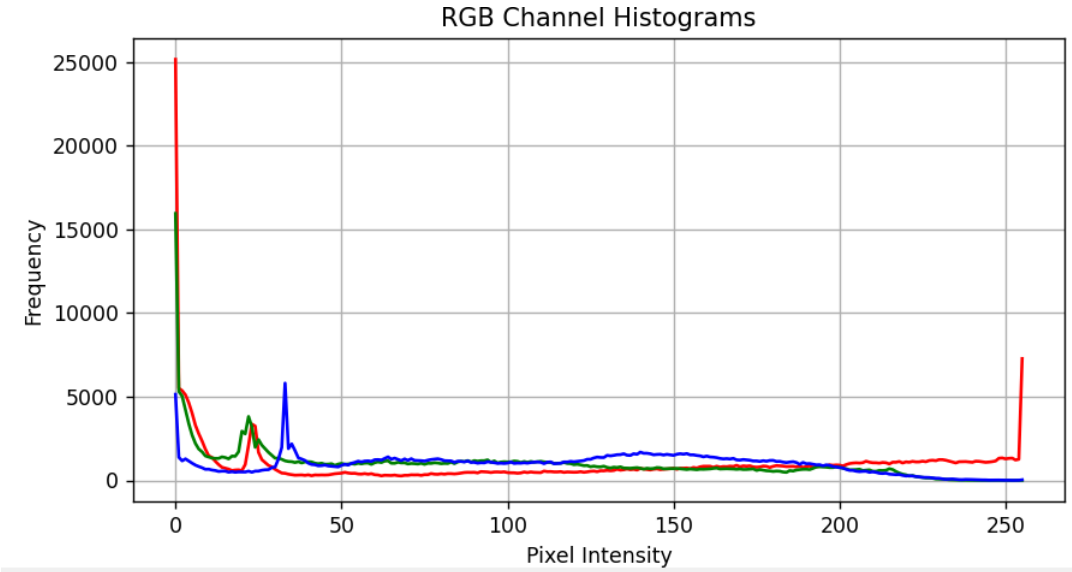
9. IMPLEMENTATION

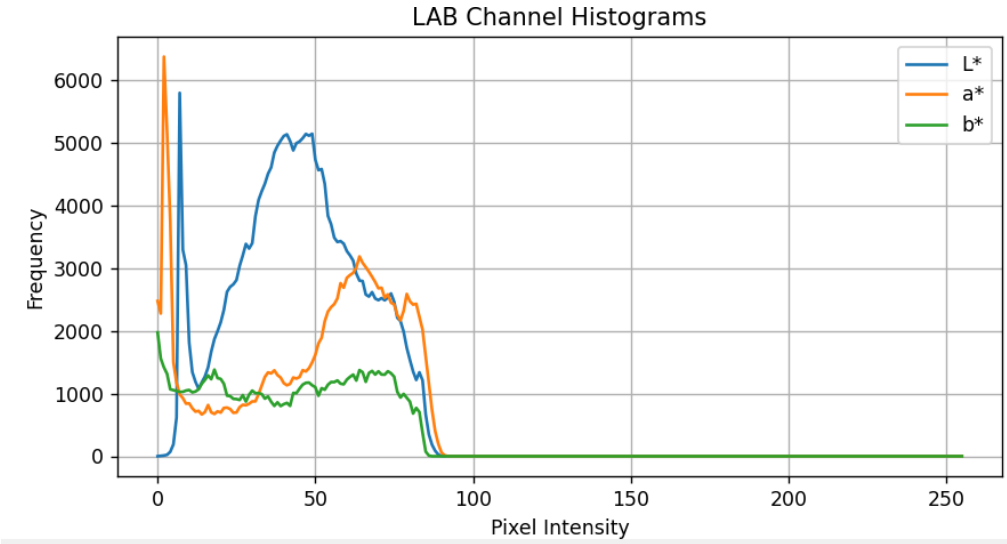
```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import os

os.makedirs("outputs", exist_ok=True)

image_path = "color.jpg"
rgb_image = cv2.imread(image_path)
if rgb_image is None:
    raise FileNotFoundError("Image not found.")

rgb_image = cv2.cvtColor(rgb_image, cv2.COLOR_BGR2RGB)
lab_image = cv2.cvtColor(rgb_image, cv2.COLOR_RGB2LAB)

r, g, b = cv2.split(rgb_image)
l, a, b_lab = cv2.split(lab_image)

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.imshow(rgb_image)
plt.title("Original RGB Image")
plt.axis("off")

plt.subplot(1, 2, 2)
plt.imshow(cv2.cvtColor(lab_image, cv2.COLOR_LAB2RGB))
plt.title("LAB Converted Image")
plt.axis("off")
```

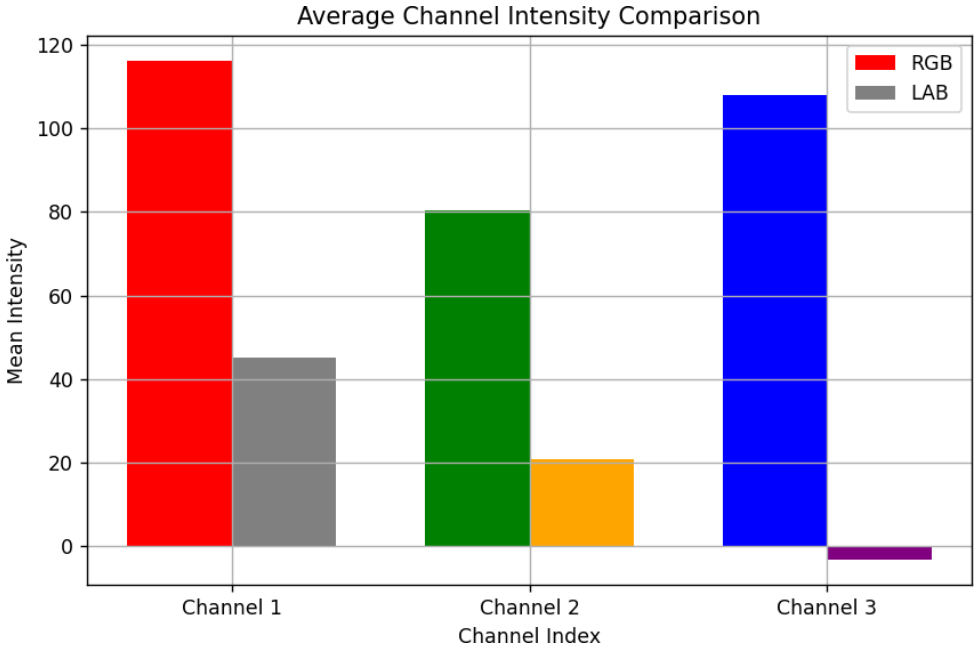
```
plt.savefig("outputs/1_RGB_vs_LAB.png", dpi=300)
plt.show()

plt.figure(figsize=(12, 4))
titles = ['L* Channel', 'a* Channel', 'b* Channel']
for i, channel in enumerate([l, a, b_lab]):
    plt.subplot(1, 3, i + 1)
    plt.imshow(channel, cmap='gray')
    plt.title(titles[i])
    plt.axis("off")
plt.savefig("outputs/2_LAB_channels.png", dpi=300)
plt.show()

plt.figure(figsize=(8, 4))
colors = ('r', 'g', 'b')
for i, col in enumerate(colors):
    hist = cv2.calcHist([rgb_image], [i], None, [256], [0, 256])
    plt.plot(hist, color=col)
    plt.xlim([0, 256])
plt.title("RGB Channel Histograms")
plt.xlabel("Pixel Intensity")
plt.ylabel("Frequency")
plt.grid(True)
plt.savefig("outputs/3_RGB_histogram.png", dpi=300)
plt.show()

plt.figure(figsize=(8, 4))
labels = ['L*', 'a*', 'b*']
for i, lab_channel in enumerate([l, a, b_lab]):
    plt.plot(cv2.calcHist([lab_image], [i], None, [256], [0, 256]), label=labels[i])
plt.title("LAB Channel Histograms")
plt.xlabel("Pixel Intensity")
plt.ylabel("Frequency")
plt.legend()
plt.grid(True)
plt.savefig("outputs/4_LAB_histogram.png", dpi=300)
plt.show()

rgb_means = [np.mean(r), np.mean(g), np.mean(b)]
lab_means = [np.mean(l), np.mean(a), np.mean(b_lab)]

x_labels = ['Channel 1', 'Channel 2', 'Channel 3']
x = np.arange(len(x_labels))
width = 0.35

plt.figure(figsize=(8, 5))
plt.bar(x - width/2, rgb_means, width, label='RGB', color=['red', 'green', 'blue'])
plt.bar(x + width/2, lab_means, width, label='LAB', color=['gray', 'orange', 'purple'])
plt.title("Average Channel Intensity Comparison (RGB vs LAB)")
plt.xlabel("Channel Index")
plt.ylabel("Mean Intensity Value")
plt.xticks(x, x_labels)
plt.legend()
```

```
plt.grid(True)
plt.savefig("outputs/5_Channel_Comparison.png", dpi=300)
plt.show()

def print_stats(name, channel):
    print(f"{name} → Mean: {np.mean(channel):.2f}, Std: {np.std(channel):.2f}, Min: {np.min(channel)}, Max: {np.max(channel)}")

print("\nRGB Channel Statistics:")
print_stats("Red", r)
print_stats("Green", g)
print_stats("Blue", b)

print("\nLAB Channel Statistics:")
print_stats("L*", l)
print_stats("a*", a)
print_stats("b*", b_lab)

lab_to_rgb = cv2.cvtColor(lab_image, cv2.COLOR_LAB2RGB)
diff = cv2.absdiff(rgb_image, lab_to_rgb)

plt.figure(figsize=(10, 4))
plt.subplot(1, 2, 1)
plt.imshow(lab_to_rgb)
plt.title("LAB → RGB Reconverted Image")
plt.axis("off")

plt.subplot(1, 2, 2)
plt.imshow(diff)
plt.title("Pixel Difference (RGB vs LAB)")
plt.axis("off")
plt.savefig("outputs/6_RGB_LAB_Difference.png", dpi=300)
plt.show()

print("\nINTERPRETATION SUMMARY:")
print("""
• RGB is device-dependent and less perceptually uniform.
• LAB separates luminance (L*) from chromaticity (a*, b*),
  allowing natural color adjustments.
• LAB provides better color manipulation for digital painting.
• Histograms show smoother distributions in LAB, enhancing tonal control.
• Pixel-difference image demonstrates LAB's robustness in preserving visual quality.
""")

print("\n All figures and results are saved in the 'outputs' folder successfully.")
```

10. RESULTS AND DISCUSSION







Modified RGB (LAB edited)

Difference Image

Channel	Mean	Std	Min	Max
Red (R)	116.20	93.28	0	255
Green (G)	80.43	66.25	0	248
Blue (B)	108.01	60.24	0	255
L*	45.13	19.46	1.06	91.04
a*	20.99	43.92	-73.78	92.39
b*	-3.26	40.24	-73.20	86.53

11. CONCLUSION AND FUTURE SCOPE

The project successfully demonstrated the conversion of images from RGB to LAB color space, emphasizing the benefits of perceptual uniformity and device independence. By using LAB, it is possible to make precise adjustments to lightness, color tone, and contrast without compromising overall color integrity, offering improved control for digital painting and image editing. The implementation in Python with OpenCV proved the conversion to be efficient, reliable, and visually effective, enhancing consistency across devices and significantly improving the quality and realism of digital artwork.

Future Scope

1. **Real-time integration:** Optimize the conversion and editing pipeline for real-time digital painting applications, possibly using GPU acceleration.
2. **User interface enhancement:** Develop artist-friendly controls for LAB adjustments, including interactive sliders for L*, a*, and b* channels.
3. **Cross-device color management:** Extend the study to handle multiple device profiles, ensuring consistent color representation across monitors, tablets, and printers.
4. **3D and video applications:** Apply RGB→LAB conversion for video frames or 3D rendering pipelines, enabling consistent color correction in animations and simulations.

12. REFERENCES

1. RGB_to_Lab_Transformation_Using_Image_Set[1].pdf
2. A Study of Lab Color Space and Its Visualization.pdf
3. The CIEDE2000 color-difference formula: Implementation notes, supplementation.pdf
4. https://www.researchgate.net/publication/355472991_RGB_Lab_conversion
5. https://www.joyful-printing.com/info/rgb-to-lab-color-space-conversion-29939246.html

13. APPENDIX

- LAB color space allows for accurate color manipulation and consistent visual results across different devices and lighting conditions.
- It is particularly useful for recoloring, shading, and tone adjustments, as modifications can be made in the lightness channel (L*) without affecting color components (a*, b*).
- Many advanced photo editing and painting tools, such as Adobe Photoshop and Krita, utilize LAB for precise color correction and texture enhancement.
- LAB enables accurate color comparison and matching across various media formats, ensuring that digital paintings retain their intended look when printed or displayed on different screens.