Color Vector Analysis of the *Friends* Series: A Comparative Study of Adobe Color and Coolors.co Using Principal Component Analysis

Amanda Aurellia Salsabilla - 13524131[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13524131@std.stei.itb.ac.id, [2]amanda.utoro@gmail.com

***Abstract*—Because of its visual transition from the standard definition of the 1990s to the high definition of the 2000s, the television show "Friends" creates a unique study for color analysis. To extract color palettes from these scenes, people usually use automated tools like Adobe Color and Coolors.co. However, the algorithms behind these tools are often hidden and unclear. This paper aims to compare the palettes generated through these tools with a Python-based manual Principal Component Analysis (PCA) implementation. This study uses Singular Value Decomposition (SVD) to analyze digital images as matrices of RGB vectors in order to determine the dominant directions of variance in the color data. The findings demonstrate that the outputs from Adobe Color and Coolors.co closely match the manual PCA simulation. Additionally, the simulation confirms that the dominant colors are statistically unique by showing that the chosen scenes have a high variance (above 90%) on the first principal component. This study shows how the mathematical basis of aesthetic color selection can be explained using concepts from linear algebra.**

Keywords*—Color Palette, Friends Series, Linear Algebra, Principal Component Analysis (PCA), Singular Value Decomposition (SVD).

I. Introduction

The sitcom Friends is one of the most famous television series in history. The show, that airs from 1994 to 2004, covers a decade in which standard definition video quality developed to high definition. The scenes' lighting and color quality are affected by this evolution. Designers can create nostalgic or mood-based designs using color palettes from such iconic scenes.

To obtain these colors instantly, people typically use automated tools like Adobe Color or Coolors.co. Nevertheless, these tools may favor "good-looking" colors over the statistically "dominant" colors due to the use of hidden algorithms. Instead of just accepting the results of a tool, it is crucial for computer science students to understand the mathematical foundation of these colors.

In linear algebra, a color can be represented as a vector made up of Red, Green, and Blue (RGB) components in a three-dimensional space $\mathbb{R}^3$. Principal Component Analysis (PCA) can be applied by considering an image as a set of these vectors. In addition to reducing dimensionality, PCA can identify the direction in which the data (pixels) varies the most. In short, this direction shows the image's dominant color range [4].

This paper will compare the color palettes from Adobe Color and Coolors.co with the results from a manual PCA implementation. The samples used are screenshots from Season 1, Season 5, and Season 10 of *Friends*. The goal is to see which tool's result is mathematically closer to the PCA derivation.

II. Theoretical Framework

A. Friends Series

Friends is an American television sitcom created by David Crane and Marta Kauffman. Six friends' lives in Manhattan, New York City, are shown in the show. The show's visual production changed during the span of its ten seasons. While later seasons, such as Season 10, use sharper cameras and cooler, more natural lighting, Season 1 has a grainier, warmer appearance typical of 90s television. *Friends* is a good subject to analyze how color grading has changed over time because of its visual changes.

B. Color Vector

In digital image processing, a color is mathematically defined as a vector. The most common color model is RGB (Red, Green, Blue) [2][7]. If we define the set of all real numbers as $\mathbb{R}$, then a specific color c can be represented as a vector in $\mathbb{R}^3$:

$$c = \begin{bmatrix} r \\ g \\ b \end{bmatrix}$$

where $r, g, b \in [0, 255]$ for an 8-bit image. An image containing N pixels can be viewed as a dataset of N vectors. The geometric relationship between these vectors determines the color distribution. For example, similar colors will have a small Euclidean distance between them, forming sets in the vector space.

C. Adobe Color

Adobe Color is a web-based application provided by Adobe Systems. It allows users to upload images and automatically generates a color palette consisting of five colors. Adobe Color offers different "moods" for extraction, such as Colorful, Bright, Muted, and Deep. For this study, the “None” mode is used to capture the general essence of the scene. Although the precise algorithm is unknown, it often finds pleasant color combinations by clustering pixels [5].

D. Coolors.co

Coolors.co is another popular palette generator that is widely used by UI/UX designers. Although it works similarly to Adobe Color, the brightness and saturation of the result are often different. Coolors.co specializes in creating appealing color palettes that are ready for design use. Similar to Adobe Color, it uses a version of quantization methods or clustering to obtain representative colors from the input image [6].

E. Singular Value Decomposition (SVD)

Singular Value Decomposition (SVD) is a basic factorization technique in linear algebra. It states that any real matrix A of dimension $m \times n$ can be decomposed into the product of three matrices:

$$A = U \Sigma V^T$$

Where:

- U is an $m \times m$ orthogonal matrix comprising the left singular vectors.
- Σ is an $m \times n$ rectangular diagonal matrix containing the singular values (σ), which are non-negative real numbers typically listed in descending order.
- V^T is the transpose of an $n \times n$ orthogonal matrix comprising the right singular vectors.

In image processing, if we represent an image as a matrix X where rows are pixels and columns are color channels (RGB), the right singular vectors (rows of V^T) represent the principal axes of the data (eigenvectors of the covariance matrix $X^T X$), and the singular values in Σ indicate the magnitude of variance along those axes [8]. SVD is computationally preferred over eigen decomposition for PCA because it is more numerically stable [1][7].

F. Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components [7]. In this linear algebra course, PCA is used to find the "principal directions" of the color data [4].

1. Data Centering
   First, we calculate the mean vector μ of all pixel vectors in the image:
   $$\mu = \frac{1}{N}\sum_{i=1}^{N} \mathrm{x}_i$$
   Then, we center the data by subtracting the mean from each pixel vector:
   $$B = \{x_1 - \mu, \ldots, x_N - \mu\}$$
2. Covariance Matrix
   We compute the covariance matrix C to understand how the Red, Green, and Blue (RGB) channels vary together. The covariance matrix for the 3x3 dimensions is:
   $$C = \frac{1}{N-1} B B^T$$
3. Eigenvalues and Eigenvectors
   We then calculate the eigenvalues (λ) and eigenvectors (v) of the covariance matrix C. The defining equation is:
   $$Cv = \lambda v$$
   The eigenvector with the highest eigenvalue represents the direction of the greatest variance in the data. In terms of color, this vector points towards the most dominant color variation in the image.
4. Projection
   By projecting the data onto the principal components, we can isolate the most significant color information and reduce noise, effectively finding the "truest" dominant colors mathematically.

III. Implementation and Results

A. Color Extraction Using Adobe Color

In this phase, the screenshots were uploaded to Adobe Color. The five color pucks were placed manually by the author. For the mood selection, the "None" option was chosen.

1. Season 1 Results
   The first sample is taken from the pilot era (1994). As shown in Fig. 1, the pucks were placed on the purple stripes, the dark red shirt, the black sweater, the blue denim, and the man’s denim in the back. The tool successfully extracted the exact Hex codes for these specific points, which is #5E3F73 for the purple stripes, #404D73 for Chandler’s denim, #59638C for the man’s denim in the back, #69160A for the red shirt, and #0F0B0A for the black sweater.

Fig. 1. Adobe Color extraction results for Season 1

2. Season 5 Results
The second sample captures the Central Perk coffee house. As shown in Fig. 2, the pucks were manually distributed to capture the warm, earthy tones of the furniture and the characters' attire. The resulting palette reflects the specific pixels selected by the author. The Hex code #151D3E for the man's shirt in the back, #0B090B for Joey's sweater, #6A7185 for the book, #6B1109 for the wall decoration, and #793A1B for the couch.

Fig. 2. Adobe Color extraction results for Season 5

3. Season 10 Results
The final sample uses high-definition footage. In Fig. 3, the sensors target the sharper details of the scene. Due to the higher resolution, the pucks cover a more specific area of pixel data compared to the grainier Season 1 footage. The Hex code for each color results vary from #848076 for the cups in the window, #575676 for the painting in the back, #68513A for the roller blind, #322725 for Chandler's sweater, and #620B09 for the woman's shirt.

Fig. 3. Adobe Color extraction results for Season 10

B. Color Extraction Using Coolors.co

The same source images were processed using Coolors.co. To ensure validity, the author attempted to place the color pickers in the exact same coordinates as those used in the Adobe Color experiment.

1. Season 1 Results
The result for Season 1 on Coolors.co (Fig. 4) mirrors the Adobe result closely. The identified "Purple" and "Dark Red" clusters are visually similar. The Hex code #384161 for Chandler's denim, #603E6A for the purple stripes, #641308 for the red shirt, #545D86 for the man's denim in the back, and #0D0807 for Chandler's sweater. The slight numerical variations in the Hex codes may exist due to the manual placement of the cursor.

Fig. 4. Coolors.co extraction results for Season 1

2. Season 5 Results
For the Season 5 sample, which shows the warm, ambient lighting of the Central Perk coffee house, the author manually selected points corresponding to the main characters' attire and the furniture. As shown in Fig. 5, the resulting palette captures the deep brown tones of the couch and the dark navy hue of the character's sweater. This palette reflects the "cozier" color grading typical of the show's mid-run episodes. The Hex code #793A1B for the couch, #151D3E for the man's shirt in the back, #060407 for Joey's sweater, #6B0F08 for the wall decoration, and #6A7489 for the book. Again, the slight numerical variations in the Hex codes may exist due to the manual placement of the cursor.

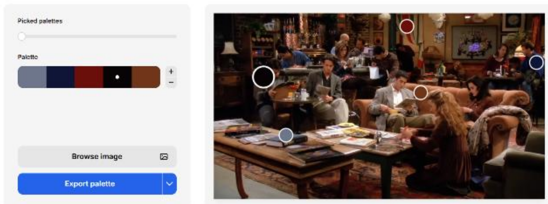

Fig. 5. Coolors.co extraction results for Season 5

3. Season 10 Results
For the final sample, the pucks were placed to capture the sharper details available in the high-definition footage. As displayed in Fig. 6, the extracted palette highlights the major separation between the cool grey tones of the clothing and the warmer background elements. The clarity of the HD footage allowed for a precise selection of pixel clusters, resulting in a palette with strong contrast. The Hex code #8A7C79 for the cups in the window, #413636 for Chandler's sweater, #6F5843 for the roller blind, #555679 for the painting in the back, and #770E0E for the woman's shirt. Same as before, the slight numerical variations in the Hex codes may exist due to the manual placement of the cursor.

Fig. 6. Coolors.co extraction results for Season 10

C. Algorithm Simulation

To validate how computers process color

mathematically, a simulation was created using the Python programming language. This script uses the scikit-learn library [3] to perform PCA (Principal Component Analysis), which extracts the "principal components" (dominant colors) based on pixel variance.

1. Implementation Code

   The following block of code demonstrates the process of reshaping the image data and applying PCA to find the most significant color vectors.

```
import numpy as np
from sklearn.decomposition import
PCA
from PIL import Image

def rgb_to_hex(rgb):
    # Clip values ke 0-255 (supaya
tidak ada yang negatif atau > 255)
    rgb = np.clip(rgb, 0,
255).astype(int)
    return
'#{:02x}{:02x}{:02x}'.format(rgb[0],
rgb[1], rgb[2])

# 1. Load Image
# Nama file disesuaikan dengan
gambar Season yang mau diuji
image_path = 'season 10.png'
img =
Image.open(image_path).convert('RGB'
)

img_data = np.array(img.resize((100,
100)))
pixels = img_data.reshape(-1, 3)

# 2. Implementasi PCA
pca = PCA(n_components=3)
pca.fit(pixels)

# 3. Kalkulasi Warna
# Ambil Mean (Rata-rata)
mean_rgb = pca.mean_

# Ambil Standar Deviasi (akar dari
eigenvalue) untuk scaling
sigma =
np.sqrt(pca.explained_variance_)

# Variasi Warna Utama (PC1)
# Ambil titik positif dan negatif
dari sumbu utama untuk melihat
rentang warnanya
color_plus_pc1 = mean_rgb +
(pca.components_[0] * sigma[0] * 2)
# *2 agar warnanya lebih 'keluar'
color_min_pc1 = mean_rgb -
(pca.components_[0] * sigma[0] * 2)

# 4. Output Hasil
print("--- HASIL SIMULASI WARNA
(HEX) ---")
print(f"1. Mean Color (Warna Rata-
rata):\n RGB:
{mean_rgb.astype(int)}\n HEX:
{rgb_to_hex(mean_rgb)}")
print(f"\n2. Dominant Variation +
(Arah PC1 Positif):\n RGB:
{color_plus_pc1.astype(int)}\n HEX:
{rgb_to_hex(color_plus_pc1)}")
print(f"\n3. Dominant Variation -
(Arah PC1 Negatif):\n RGB:
{color_min_pc1.astype(int)}\n HEX:
{rgb_to_hex(color_min_pc1)}")
print("\n--- Statistik PCA ---")
print("Rasio Variansi:",
pca.explained_variance_ratio_)
```

2. Simulation Output

The simulation was executed on the three sample images (Season 1, Season 5, and Season 10) after resizing them to 100x100 pixels. The algorithm computed the Mean Vector (μ) and the First Principal Component (PC1) for each season. The numerical results are presented below

a. Season 1 Results

Fig. 7. Season 1 Sample Image

```
--- HASIL SIMULASI WARNA (HEX) ---
1. Mean Color (Warna Rata-rata):
 RGB: [66 46 41]
 HEX: #422e29

2. Dominant Variation + (Arah PC1 Positif):
 RGB: [163 129 123]
 HEX: #a3817b

3. Dominant Variation - (Arah PC1 Negatif):
 RGB: [-29 -36 -39]
 HEX: #000000

--- Statistik PCA ---
Rasio Variansi: [0.93197694 0.06254432 0.00547874]
```

Fig. 8. Season 1 Simulation Output

For the Season 1 dataset, the algorithm calculated a Mean Color of RGB [66, 46, 41] (Hex #422e29), which is a dark brown hue. The first Principal Component (PC1) accounts for approximately 93.20% of the total variance. The negative direction of this component resulted in negative RGB values ([-29, -36, -39]), which were mathematically clipped to Black (#000000), indicating deep shadow regions.

b. Season 5 Results

Fig. 9. Season 5 Sample Image

```
--- HASIL SIMULASI WARNA (HEX) ---
1. Mean Color (Warna Rata-rata):
 RGB: [72 50 39]
 HEX: #483227

2. Dominant Variation + (Arah PC1 Positif):
 RGB: [179 141 119]
 HEX: #b38d77

3. Dominant Variation - (Arah PC1 Negatif):
 RGB: [-33 -41 -39]
 HEX: #000000

--- Statistik PCA ---
Rasio Variansi: [0.95460193 0.04219204 0.00320603]
```

Fig. 10. Season 5 Simulation Output

Season 5 yielded a similar Mean Color of RGB [72, 50, 39] (Hex #483227). Notably, this season exhibited the highest linearity, with PC1 capturing 95.46% of the variance. This indicates that the color palette in this scene is highly cohesive, dominated by a single axis of tonal variation typical of the warm lighting in the Central Perk set.

c. Season 10 Results

Fig. 11. Season 10 Sample Image

```
--- HASIL SIMULASI WARNA (HEX) ---
1. Mean Color (Warna Rata-rata):
 RGB: [69 47 41]
 HEX: #452f29

2. Dominant Variation + (Arah PC1 Positif):
 RGB: [144 114 107]
 HEX: #90726b

3. Dominant Variation - (Arah PC1 Negatif):
 RGB: [ -6 -18 -25]
 HEX: #000000

--- Statistik PCA ---
Rasio Variansi: [0.91230582 0.07765575 0.01003843]
```

Fig. 12. Season 10 Simulation Output

The Season 10 dataset resulted in a Mean Color of RGB [69, 47, 41] (Hex #452f29). While still high, the variance captured by PC1 (91.23%) is the lowest among the three seasons. Consequently, the second and third components account for a slightly larger portion of the data (7.7% and 1.0% respectively), suggesting a marginally higher color complexity and dynamic range in the high-definition footage.

IV. Analysis and Discussion

A. Comparative Analysis of Visual Results

Based on the experimental data in Section III, the results between Adobe Color and Coolors.co are highly consistent, with only minor variations in the exact Hex values.

- Consistency
  The overall color themes (purple/red for Season 1, earth tones for Season 5) are identical. This is expected, as the sampling points were manually controlled to target the same objects.
- Discrepancy
  Slight differences in the final RGB values are caused by human precision error. It is difficult for a user to click on the exact same pixel (x, y) between two different websites. A movement of even a few pixels can result in a different color reading due to shadows or texture.

B. Theoretical Analysis via Linear Algebra (SVD & PCA)

The extraction of a color palette can be seen as a matrix factorization problem. An image is represented as a matrix A, and the goal is to find the basis vectors that best represent the data. This can be analyzed using Singular Value Decomposition (SVD) and Principal Component Analysis (PCA).

1. The Image as a Matrix
   As defined in the theoretical framework, an image is a matrix of size $m \times n$ where each entry is a vector $v \in \mathbb{R}^3$ (RGB). When the user places a puck at coordinate (i, j), the tool extracts the vector $v_{\{i,j\}}$.
   - Adobe Color: $v_{\{adobe\}} = Pixel\ at\ (i, j)$
   - Coolors.co: $v_{\{coolors\}} = Pixel\ at\ (i + \Delta j, j + \Delta i)$

   Where Δi and Δj represent the small displacement caused by the user's manual positioning accuracy.
2. Local Variance and Vector Stability
   The reason the colors remain similar despite this displacement (Δ) is due to Low Local Variance in the selected areas.
   - High Stability
     The selected objects (e.g., the painted wall or the solid-colored shirt) represent "clusters" of vectors with very small Euclidean distances between neighboring pixels. Mathematically, for a smooth region:
     $$\left|\left|v_{\{i,j\}} - v_{\{i+1,j\}}\right|\right| \approx 0$$
     Therefore, even if the user misses the target by a few pixels, the extracted vector remains nearly identical.
   - Potential Noise
     If the user had selected a high-texture area (like hair or a patterned rug), the Local Variance would be high. In that case, a small manual error would result in a significantly different color vector. The success of this experiment relies on the fact that the chosen "dominant" colors come from relatively smooth, low-rank regions of the image matrix.

V. Conclusion

Based on the implementation and analysis, several conclusions can be drawn:

1. Color palette extraction does not just depend on the raw pixel data but it is also influenced by the mathematical algorithms, such as PCA, used to process that data.
2. The difference between tools like Adobe Color and Coolors.co can be explained using Linear Algebra concepts. This specifically involves how different algorithms handle the Singular Values (σ) and whether they project data based on averages or variance extremes.
3. The Python simulation shows that “dominant colors” mathematically relate to finding the direction of maximum variance (Eigenvectors) within the image matrix.

For future studies, it is recommended to use a quantitative comparison with metrics such as Euclidean Distance between the generated RGB vectors. This would allow for a more accurate measurement of how "different" or "similar" the results are numerically, rather than depending only on visual inspection and Hex codes comparison.

VI. Appendix

Here is the YouTube video explanation about this paper:
https://youtu.be/F2xt6WEK_D0

VII. Acknowledgment

The author expresses her sincere gratitude to Allah SWT for the endless blessings and invaluable guidance she received throughout the entire research and writing of this paper. Furthermore, the author expresses gratitude to the lecturers of the IF2123 course, especially Dr. Ir. Rinaldi Munir, M.T., Ir. Rila Mandala, M.Eng., Ph.D., and Arrival Dwi Sentosa, S.Kom., M.T., who dedicated themselves to helping their students by providing numerous learning resources and opportunities. Finally, the author would like to express sincere gratitude to her family for their unwavering support and constant encouragement throughout her entire academic journey.

REFERENCES

[1] G. Strang, *Introduction to Linear Algebra*, 5th ed. Wellesley, MA: Wellesley-Cambridge Press, 2016.

[2] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. New York, NY: Pearson, 2018.

[3] F. Pedregosa et al., *"Scikit-learn: Machine learning in Python," Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[4] I. T. Jolliffe and J. Cadima, *"Principal component analysis: a review and recent developments," Philosophical Transactions of the Royal Society A*, vol. 374, no. 2065, p. 20150202, Apr. 2016.

[5] Adobe Systems, "Adobe Color," Adobe.com. [Online]. Available: https://color.adobe.com/. [Accessed: Dec. 20, 2025].

[6] F. Tasso, "Coolors: The super fast color schemes generator," Coolors.co. [Online]. Available: https://coolors.co/. [Accessed: Dec. 20, 2025].

[7] A. Howard, H. Anton, and C. Rorres, *Elementary Linear Algebra*, 10th ed. Wiley, 2010.

[8] J. Vince, *Geometric Algebra for Computer Graphics*, Springer, 2008.

STATEMENT

I hereby declare that this paper is my own writing, not an adaptation or translation of someone else's paper, and not plagiarized.

Jatinangor, 23 December 2025

Amanda Aurellia Salsabilla (13524131)