JURNAL INFORMATIKA
PROXIES

Vol.9, No.1, July, 2025

Jawline Detection for Skin Tone Identification to Determine Foundation Recommendation Using K-Means Clustering.

Eveline Aurelia Mulyono, Tan[1], Rosita Herawati S.T., M.I.T.[2]

[1] Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata, 21k10015@student.unika.ac.id
[2] Teknik Informatika Fakultas Ilmu Komputer, Universitas Katolik Soegijapranata, rosita@unika.ac.id

Corresponding Author Email: rosita@unika.ac.id

https://doi.org/10.24167/proxies.v9i1.13251

Received: 2025-02-17
Revised: 2025-08-23
Accepted: 2025-10-22
Available online: 2025-10-23

Keywords:
Foundation, OpenCV, K-Means, Euclidean

ABSTRACT

Many people now use makeup to cover up their facial imperfections, but some seem much worse after wearing makeup. One of the causes is the wrong choice of foundation shade. Choosing the wrong foundation can cause your face to appear grayer. To reduce errors in choosing a foundation shade, researchers created a system by inputting a facial image which will then be processed with the help of OpenCV, K-Means, and Euclidean. The use of OpenCV is to determine the jawline because the skin color around the jawline will increase the accuracy of choosing a suitable foundation. Then K-Means find the dominant color of the skin around the jawline and make it in the form of a hex color. Then Euclidean finds the foundation shade that best matches the user's skin tone results by comparing the hex color of the skin tone obtained with the hex color in the dataset. The dataset used comes from Kaggle, which is around 600 hex colors from various global product foundations. The result you will get is the foundation shade that best matches the input face image.

1. INTRODUCTION

Background

Often women don't feel confident about the condition of their face. Sometimes it's due to scars or puberty hormones. However, it cannot be denied that humans look at someone based on their physical appearance first. So it is common for women to cover their facial imperfections by using make-up [1]. But it's rather difficult to find a color shade of makeup that fits each user because when the color used does not match our skin, it can cause the skin to look gray. Researcher Dina from Research and Development Zoya Cosmetics found that it took a lot of trial and error to find a shade that suited the respondents' various skin tones.

Matching the skin tone to the shade color can actually avoid trial and error. To address this, shades are categorized by their hex codes. Euclidean enables the recommendation of shades that complement the dominant skin color obtained through the process by K-Means[2][3]. The hex color dataset taken from Kaggle is a list of makeup names along with the color hex of each makeup product.

The result of this research is to detect the user's skin color and match it with the most suitable makeup shade. This research can be used by buyers to buy shades that match their skin color. It can also be used by makeup companies to improve services. Companies can provide a live interactive experience for customers, where their faces are automatically scanned to give appropriate product recommendations. As a result, it will show the dominant skin color and also the color of the suitable foundation shade as an image so that it can be compared and also display the distance between the two hexes.

also display the distance between the two hexes.

Problem Formulation

The problems formulation for this research are:

1. How does OpenCV detect the jawline?
2. Can the K-Means algorithm determine the dominant color based on an image of human skin?
3. Can Euclidean be used to calculate the distance between skin color and shade representations, to find a sufficient minimum distance to be utilized as a shade recommendation?

Scope

This research will only discuss foundation shades that are adjusted to the dominant facial skin color. The dominant facial skin color will be taken from the jawline area. The jawline is often a good reference for determining the right foundation color because there tends to be less redness than other areas of the face. While capturing the jawline, the photograph must use a front view with only one face and no obstructions. This research used globally available products like Maybelline, L'Oréal, Revlon, Dior, and many more as the dataset. The data was taken from Kaggle.com in 2021. To match the dataset with the subject, the K-Means algorithm is used to find the dominant color of the skin. Then, it will be compared with existing data using Euclidean Distance.

Objective

This research aims to help people who want to buy facial makeup so they don't make the wrong choice of color. This can be realized by using OpenCV to detect facial images, especially the jawline. After the jawline is detected, the dominant color is searched using K-Means. Then, Euclidean Distance is used to find the best color between the dominant color in the jawline and the foundation color in the data. With this research, it is hoped that purchasing the wrong foundation shade can be reduced.

LITERATURE STUDY

OpenCV needs to be the first step to getting started. OpenCV can be useful in detecting faces and features you want to detect. Patel et al [11] proved that it could be used to detect faces and their features such as eyes, nose, eyebrows, jawline, lips, and forehead. His research uses OpenCV to detect faces so that the faces of two people can be swapped. The data used uses videos containing people's faces. However, this research has a weakness, namely the lack of texture identification.

After obtaining the relevant facial features from OpenCV, the next step is to get the dominant color of that area. Dominant colors can be searched using K-Means. According to Chang and Mukai [2] research, they want to create a method to extract dominant colors automatically. Researchers used K-Means in the CIELAB color space and the graph cut of a region adjacency graph (RAG) to get the dominant color. The goal is to identify the dominant color from the image accurately. The image's 500 color values were used in the research. The results of the study resulted in dominant colors based on color contrast, area, and saturation on the image. However, they got 500 color data from images that were extracted manually.

The effectiveness of the K-Means algorithm for classifying skin color has also been proven by Pina and Ma [4] research. The study aims to provide objective, automatic, easy-to-use, and more accurate results in measuring skin color. They used face detection, skin segmentation, K-Means clustering, and then validation and testing to complete this research. Although the study did not mention a clear dataset because it aimed to explain more about skin color measurements, the researchers were finally able to show the effectiveness of the K-Means-based Classification Algorithm for Skin Color, or what they call CASCo. However, it may have limitations in accurately categorizing skin tones in certain conditions or with specific image characteristics. In the arts sector, it can also prove K-Means' ability to find dominant colors. Proven in Atram and Chawan [5] research in 2020. Researchers use K-Means to find dominant colors in artistic images in the form of painting images. They input artistic painting images, which are clustered into dominant colors. The results obtained show the potential of K-Means in color search and analysis. However, the drawback of the research is that it ignores the possibility of shadows, light, and so on.Not only that, K-Means can avoid shortages or excess stock caused by clothing trends. The K-Means are used to find trending colors. This has been proven by research conducted by Gladys and Olalekan [6]. They used 500 photos of people attending weddings and 500 photos of formal events such as seminars, office meetings, etc. They use accuracy and loss calculations as a way of their evaluation. However, in the next research, it is hoped that they will be able to visualize the results they get at the same time.

There is also the ability of K-Means combined with Euclidean to group types of mint leaves. Types of mint leaves are grouped based on shape, texture, and color. In the research of Harjanti et al. [7], K-Means is used to classify objects based on their attributes, while Euclidean is used to calculate the level of similarity between two images. The dataset used is 100 images of several types of mint, for example, peppermint, apple mint, chocolate mint, etc. In this research, precision, recall, and accuracy were used as evaluation methods. However, in this study, it had a 16% error rate.

Research on color segmentation of biological images supports the previous statement that Euclidean can be used to calculate similarity distance. To collect color-based data on species, camouflage, etc., Weller et al. [8] stated that Euclidean is able to improve color segmentation in biological research. Even though the data utilized in this study is not precisely stated, the recolorize package can be used to overcome the limitations of this research so that it can handle huge variations.

Euclidean's ability to calculate similarity distances has also been proven through research using hashing technology on content and internet security to limit malicious content. Although the type of dataset used is not explained in detail, this technology usually operates by generating hash values from multimedia files such as images, videos, and audio. Accuracy in detecting malicious content is used as an evaluation technique. However, this research has weaknesses. Hashing can be manipulated to circumvent detection. This result came from Farid's research in 2021 [9].

Furthermore, other relevant research is concerned with the use of Euclidean distance in different sectors. Euclidean Distance can estimate color distances for porcelain color, as shown by research by Jing et al. [3]. The goal of this study is to demonstrate how automation can improve efficiency and accuracy when choosing porcelain colors. The dataset used is made up of porcelain photographs with a wide range of glaze colors and patterns. The sung porcelain photos were first preprocessed to improve their quality. Next, a color segmentation algorithm was used. Finally, K-Means clustering was applied, and the outcomes were evaluated in regard to the color proportions of the overall color palette. Furthermore, the outcomes effectively demonstrate how automation can boost productivity and accuracy. This research's limitation is its inaccurate representation of complicated hues.

Euclidean have also found similarities in the field of human health care. This is proven by research by Chugh et al. [10] which obtains comparable photos from a set of images based on the similarity between the query image and images in the database. They compared the input image with the image in the database which was then extracted into several features and calculated the distance, one of which was using Euclidean. The results were evaluated using precision, recall, and F-measure. The dataset contains 100 photos of body parts such as the eyes, nose, ears, and hands. Because the research only uses photographs, the tiny dataset and lack of real-world validation are the research's weaknesses.

In this research, a combination of K-Means and Euclidean will be used to get foundation color recommendations. The various types of foundation that will be reviewed include liquid, cream, powder, and stick foundations with various colors suitable for various skin types, as explained by Saputra et al. [1]. This process will involve grouping data using the K-Means algorithm to identify the most dominant color groups. Next, the Euclidean distance will be used to measure the similarity between these colors and determine the most suitable recommendation for each skin tone. Of all the existing journals, none has combined OpenCV, K-Means, and Euclidean to make foundation recommendations. In this research, I plan to use these three techniques together to develop a system that can provide more accurate and personalized foundation color recommendations.

RESEARCH METHODOLOGY

Research Methodology

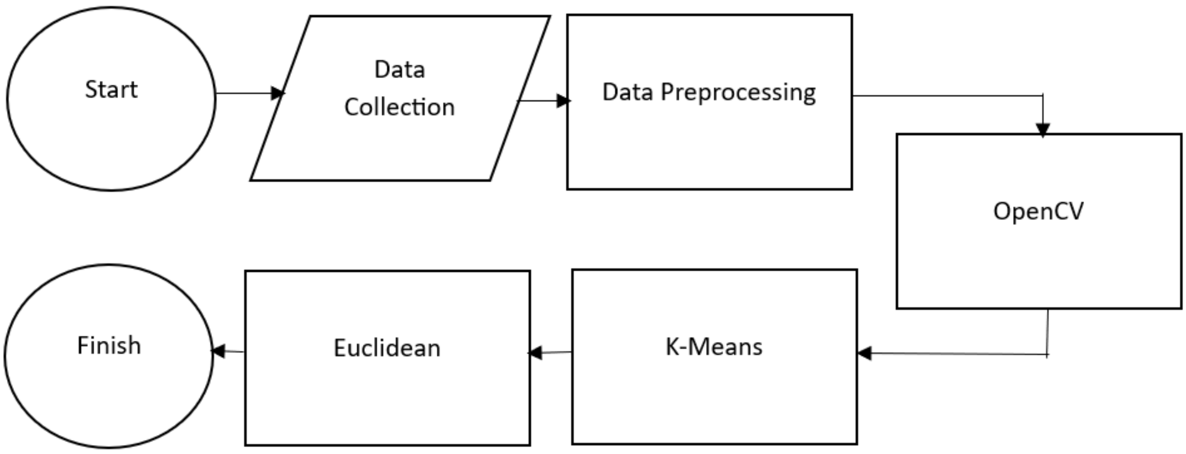


Figure 1. Methodology Flowchart

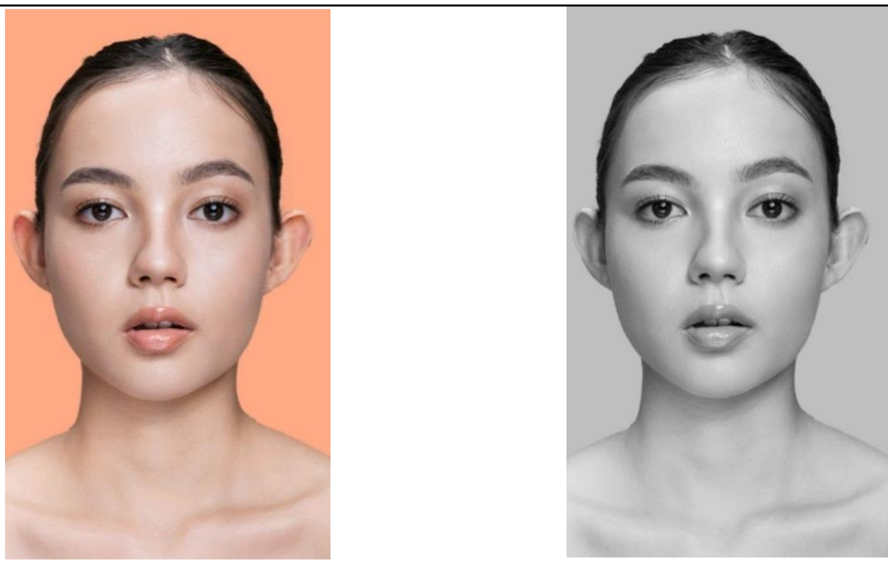
This research uses OpenCV, K-Means, and Euclidean. The flow of this research begins with data collection and data preprocessing. After data collection and data preprocessing, the data is implemented with OpenCV. OpenCV is used to handle various image processing operations, which helps in identifying and extracting important features from image data. From this implementation, it will be processed using K-Means. K-Means is a clustering algorithm that is used to group data into several groups based on feature similarity. And the last one is processed again using Euclidean. The Euclidean method is used to measure distances which makes it possible to determine similarities between data. The final result is a recommendation for the most suitable foundation.

Dataset Collection

The data set that will be used in this research is a data set that contains a list of foundation colors available in the US, Japan, Nigeria, and India. The brand's products and foundations are considered "best sellers" in each country. This data was obtained from Kaggle . Created in 2018 and last updated in 2021 with a total of 624 data. This dataset contains 10 variables, the variables are: (1) brand, (2) brand_short, (3) product, (4) product_short, (5) hex, (6) H, (7) S, (8) V, (9) L, and (10) group. Not all data in the dataset will be used. Only the brand, product names, and hex color will be used. By only using product names and hex color, this research can focus more on color comparison analysis which is the core of the research objective.

Data Preprocessing

Some data will be changed to improve the results of this study. What will be done includes Data Transformation. The data transformation that goes on is transforming the image to Grayscale. The following is a further explanation



(a) **(b)**
Figure 2 (a) Before Grayscale and (b) After Grayscale

The face image input by the user will be converted to grayscale. Converting an image to grayscale is an important step in image processing because it can increase the accuracy of finding the jawline. This process removes color information and retains only brightness information, which is useful in various image processing applications.

OPENCV

Images that have been converted to grayscale will be processed again. The first thing to do is find a face in an image. Then look for the important parts, for example, the jaw. After that, the jaw's coordinate positions are saved and attached to a line to indicate that the line is the jawline. The evaluation method will be tested using many facial images and the success rate will be calculated. The method used to validate the success of jawline detection on the face will be measured using standard deviation. Standard deviation is a value that shows the level or degree of group variation or a standard measure of deviation from the mean. This measure is related to variance, which describes how scattered the quantitative data is. With data distribution measures, we can see how the data is spread from the smallest to the largest data or how the data is far from the center of the overall data distribution. The following is the formula for standard deviation.

[1] https://www.kaggle.com/datasets/shivamb/makeup-shades-dataset/data

$$\sigma = \sqrt{\frac{\sum \quad (X_i - \underline{X})^2}{n}} \tag{1}$$

on function (1), σ is standard deviation, X_i is the data value, $\underline{X}$ is the average of the data, n is the total amount of data. The overall function (1) explains the standard deviation formula. [12]

K-MEANS

After finding the jawline, the next step is to find the area around the jawline. The area around the jawline will be changed to RGB color. After that, K-means is applied to find the color with the highest RGB count. K-Means uses the clustering method. The results will be determined as the dominant color of the user's skin. For the evaluation method, it will be tested using the Silhouette Score. The silhouette score method, which measures the quality of a cluster, is used to find the average silhouette coefficient of all samples for a different number of clusters. A higher score on the silhouette score, which goes from -1 to 1, means that the sample is poorly matched to nearby clusters and well-matched to its own cluster. A silhouette score near 0 indicates overlapping clusters, while a negative value indicates that samples may have been assigned to the incorrect cluster. The following is the formula for the Silhouette Score.

$$Silhouette\ Score = \frac{b - a}{max(a, b)} \tag{2}$$

on function (2), the explanation of the silhouette score formula is that a is the average distance between a sample and all other points in the same cluster. And b is the average distance between a sample and all points in the nearest cluster to which the sample does not belong. [13]

EUCLIDEAN

Before processing the dominant color that was previously obtained, the existing dataset will be changed first. The hex in the dataset will be converted to RGB to proceed to the next step. Then the dominant RGB color that has been found is compared with the RGB in the dataset. Euclidean distance functions to calculate the distance between the dominant skin color and all colors from the dataset. Then look for the color index with the closest distance. This is the result of the foundation color closest to the user's skin tone. As an evaluation method, a color block will be displayed to show the comparison of the two colors. The evaluation will be conducted independently by visually observing the color similarity and matching their hex codes. The Euclidean distance formula used is as follows.

$$d = \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2} \tag{3}$$

r, g, and b each represent the colors found in the hex, r is red, g is green, and b is blue. and then d represents the result of the Euclidean calculation. Number 1 is the first hex, namely the dominant skin color, and number 2 is the second hex, namely the color found in the dataset. [9]

IMPLEMENTATION AND RESULT

Experiment Setup

This study was carried out with Python 3.10.12 on Google Colab with standard configurations. In this Google Colab environment, an Intel(R) Xeon(R) @2.20GHz CPU with about 12.7 GB of RAM was used, despite the lack of a GPU. Several libraries, including os, cv2, dlib, numpy, and pandas, were used to process the data. After the data is processed, it is clustered using KMeans and evaluated using silhouette_score. The distance between the data score is calculated using euclidean_distances, and the resulting image is displayed using cv2_imshow.

Implementation

To begin this research, a library must be downloaded in order to detect the location of the jawline. It has been tested with three libraries to achieve better results: Mediapipe, Dlib, and Face Alignment. Based on the findings, Dlib can deliver the greatest results. Shape_predictor_68_face_landmarks.dat is the Dlib model file that was used in this research. The file was downloaded and extracted in lines 1 and 2 of the code below.

```
1.  !wget http://dlib.net/files/shape_predictor_68_face_landmarks.dat.bz2
2.  !bzip2 -d shape_predictor_68_face_landmarks.dat.bz2
```

The process begins by reading all the images in the image folder one by one. As in the first to sixth lines, each image is read and converted to grayscale. This step aims to optimize face detection to be faster and more efficient. After that, face detection is performed, and since there is only one face per image, the first detected face is directly taken as the detection target.

After the face is detected, the shape predictor from Dlib works to recognize 68 points on the face, such as the eyes, nose, and jawline, which are the focus of the initial research. After the facial points are identified, a looping process is carried out to take the first 17 points that represent the jawline position. According to Dlib documentation, point 0 is located on the right side of the jaw, point 8 is at the bottom of the chin, and point 16 is on the left side of the jaw. For each of these points, the x and y coordinates are recorded and then stored in a particular variable called jawline_point. In the next step, these points are connected to each other to form a line that indicates the position of the jawline. This is shown in lines seven to thirteen.

```
1.  for img_path in image_files:
2.      img = cv2.imread(os.path.join(image_folder, img_path))
3.      gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
4.      faces = detector(gray)
5.      face = faces[0]
6.      landmarks = predictor(gray, face)
7.      jawline_points = []
8.      for n in range(0, 17):
9.          x = landmarks.part(n).x
10.         y = landmarks.part(n).y
11.         jawline_points.append((x, y))
12.         if n > 0:
13.             cv2.line(img, jawline_points[n-1], jawline_points[n], (0, 255, 0), 2)
```

To obtain the skin color around the jawline, the area of the jawline that has been identified earlier is needed. Therefore, 10 pixels of the area around the jawline that has been drawn earlier are taken, as shown in the 5 lines of code below.

```
1.  margin = 10
2.  x_min = max(min([p[0] for p in jawline_points]) - margin, 0)
3.  x_max = min(max([p[0] for p in jawline_points]) + margin, img.shape[1])
4.  y_min = max(min([p[1] for p in jawline_points]) - margin, 0)
```

```
5.      y_max = min(max([p[1] for p in jawline_points]) + margin, img.shape[0])
```

The code below's first and second lines transform the image input into pixel format, giving each pixel an RGB value so the K-Means algorithm can process it. Next, the K-Means process begins by determining the ideal number of clusters, which is 2 clusters. This selection is based on prior tests with three and four clusters, with the best results obtained with two clusters. Once the number of clusters has been determined, the K-Means algorithm is used to identify the dominating color in the image.

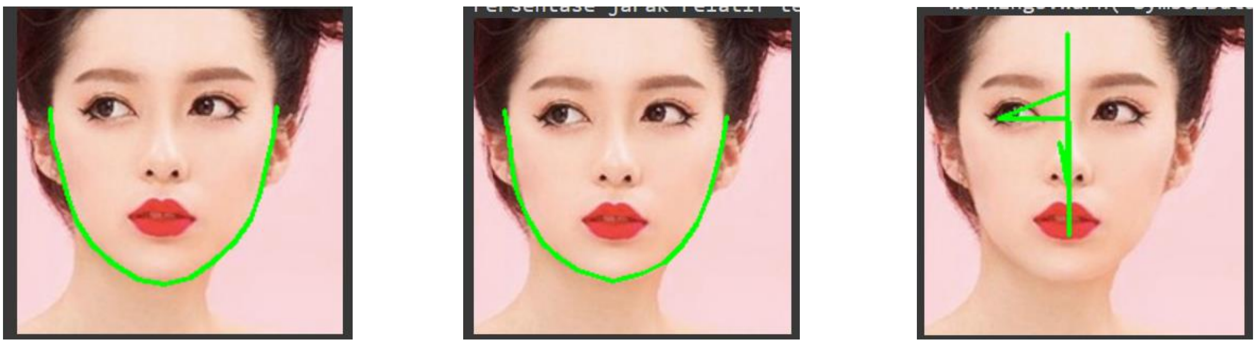
```
1.      jawline_area_reshaped = jawline_area.reshape((jawline_area.shape[0] * jawline_area.shape[1], 3))
2.      kmeans = KMeans(n_clusters=n_clusters_used, random_state=0).fit(jawline_area_reshaped)
```

This method, as seen in the four lines of code below, begins by converting the dominant color found in BGR format to RGB to make it more compatible with the following steps. After the conversion, the Euclidean Distance algorithm is used to calculate the distance between the dominant color and the colors in the foundation dataset, which helps determine how similar each color is to the dominant color. The results of this algorithm are used to arrange the colors according to their distance. From this order, the three colors with the closest distance are selected as the most suitable final result.

```
1.      dominant_rgb = dominant_color[::-1]
2.          distances = euclidean_distances([dominant_rgb], hex_values_converted)
3.          nearest_index = distances.argmin()
4.          matching_hex = hex_values.iloc[nearest_index]
```

Results

In addition to the findings of this study, several experiments have been conducted to see whether the applications built achieved good results. Image data was partially taken from the Roboflow` site to conduct the testing, with a total of 100 randomly selected images as a test sample. Based on the library used to detect the jawline, here is a comparison in terms of results and time.



(a) (b) (c)

Figure 3. (a) Dlib result, (b) Mediapipe result, and (c) Face Alignment result

It is seen from the visualization results above that the Dlib and Face Alignment approaches detect faces, including the location of the jawline, with reasonable accuracy. However, because Mediapipe cannot precisely identify the jawline, the results are less than ideal. This suggests that Dlib and Face Alignment are better than Mediapipe in terms of jawline accuracy, even though all three libraries are capable of detecting faces.

[1] https://universe.roboflow.com/show-stopper/deglam/browse

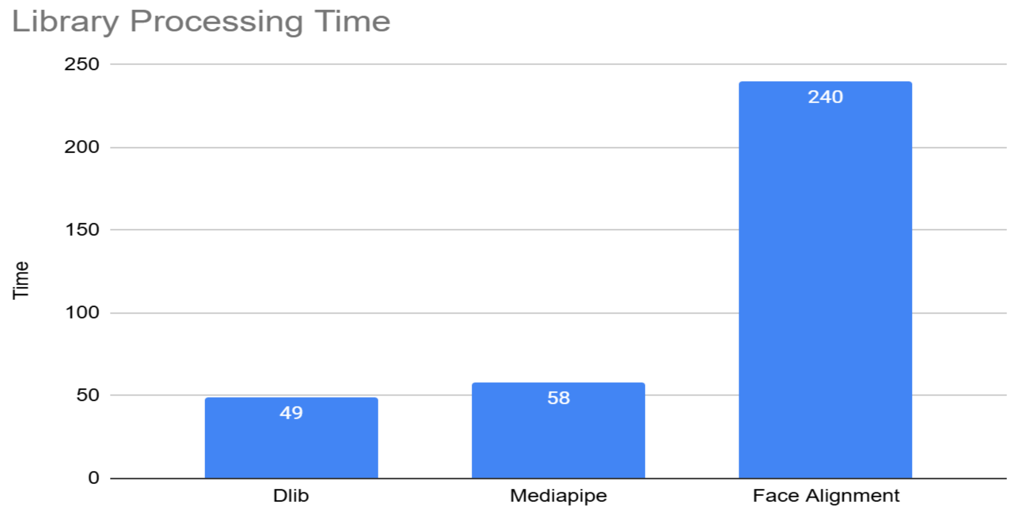


Figure 4. Library Processing Time

Here are the time results from testing with the Dlib, Mediapipe, and Face Alignment libraries on the same dataset. The results reveal that Dlib took 49 seconds to complete the facial detection process. Meanwhile, Mediapipe was slightly slower, taking 58 seconds. On the other hand, Face Alignment took much longer, approximately 4 minutes. In addition to using the library to detect the jawline, this research also attempts several changes in the number of clusters to determine the optimal cluster value. The outcomes of such experiments are shown here.

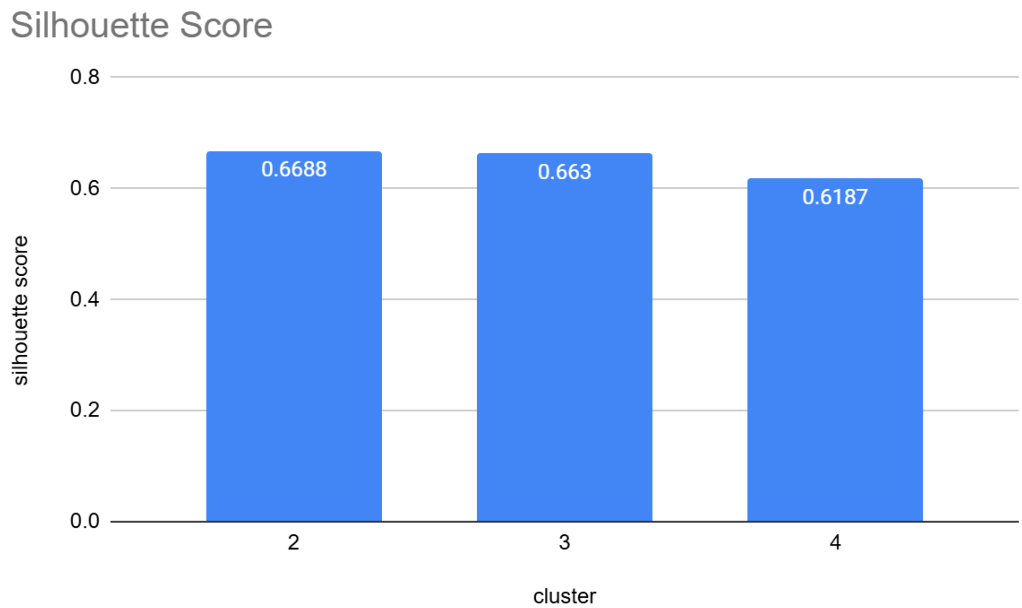


Figure 5. Silhouette score each cluster

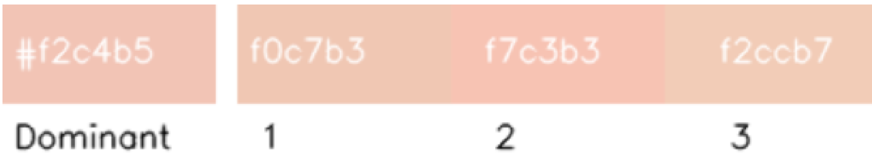
The ideal number of clusters was determined by clustering evaluates using the Silhouette Score. In the trial with two clusters, the Silhouette Score was 0.6688. The score dropped to 0.663 when the number of clusters was raised to three. Similarly, the experiment with four clusters obtained a Silhouette Score of 0.6187. This result summarizes how variations in the number of clusters effect the Silhouette Score in each case.

The position of the jawline in each image will be in a similar location because the faces in the photos are facing forward. With this consistent position, the calculation of each point across all images is done using standard deviation to ensure that each line in all images has a similar position.

```
Standar deviasi untuk titik jawline 0: (x: 7.53, y: 6.21)
Standar deviasi untuk titik jawline 1: (x: 6.79, y: 5.60)
Standar deviasi untuk titik jawline 2: (x: 6.37, y: 5.19)
Standar deviasi untuk titik jawline 3: (x: 6.24, y: 5.02)
Standar deviasi untuk titik jawline 4: (x: 5.67, y: 4.89)
Standar deviasi untuk titik jawline 5: (x: 4.89, y: 4.68)
Standar deviasi untuk titik jawline 6: (x: 3.72, y: 4.29)
Standar deviasi untuk titik jawline 7: (x: 2.31, y: 3.73)
Standar deviasi untuk titik jawline 8: (x: 1.84, y: 3.59)
Standar deviasi untuk titik jawline 9: (x: 2.31, y: 3.54)
Standar deviasi untuk titik jawline 10: (x: 3.34, y: 4.17)
Standar deviasi untuk titik jawline 11: (x: 4.74, y: 4.87)
Standar deviasi untuk titik jawline 12: (x: 5.92, y: 5.41)
Standar deviasi untuk titik jawline 13: (x: 6.63, y: 5.59)
Standar deviasi untuk titik jawline 14: (x: 6.94, y: 5.79)
Standar deviasi untuk titik jawline 15: (x: 6.99, y: 6.05)
Standar deviasi untuk titik jawline 16: (x: 7.27, y: 6.53)
```

Figure 6 Standart Deviation result

Because the jawline search uses 16 points, the standard deviation is calculated for each point on all photos. For example, for point 0, the calculation results reveal a standard deviation of 7.53 for x and 6.21 for y, indicating that point 0's position does not exceed 10 pixels. This process continues till point 16. The study results demonstrate that the difference in face line positions across all photos is no more than 10 pixels, showing that jawline placements are highly consistent. With little variance, it can be determined that the jawline location in each photo is nearly identical, allowing for more consistent and precise detection results. This research produces three dominant colors that reflect skin tones and colors found in the dataset. The colors are displayed in the form of color blocks, as shown in the example below.



The visualization above displays four color blocks, each with a hex color code written inside. The color block on the left shows the dominant skin color in a facial image, while the three on the right show the most suitable foundation color, chosen based on Euclidean Distance calculations. The result indicates that both colors appear similar and have nearly identical hex codes.

The image below shows the final result after the matching color has been successfully found in the dataset. After the closest color is obtained, the system will display product data that corresponds to that color. The displayed product data provides information about the recommended foundation.

```
3 recommendations for thumb (51).jpg:
1. HEX: f0c7b3, Brand & Product: L'Oréal True Match, Distance: 4.1231
2. HEX: f7c3b3, Brand & Product: Make Up For Ever Ultra HD, Distance: 5.4772
3. HEX: f2ccb7, Brand & Product: Make Up For Ever Ultra HD, Distance: 8.2462
```

Figure 8. Brand result

Based on the research results, it was concluded that the use of the Dlib library yielded the best results compared to the other two libraries, both in terms of jawline detection accuracy and the fastest processing time. The success of this detection is evident from the standard deviation of the jawline position not exceeding 10 pixels, indicating the consistency of the jawline position across all images.

To obtain the dominant color of the jawline that has been achieved, this study used 2 clusters selected based on the highest Silhouette Score compared to clusters 3 and 4. The final results are visualized in the form of color blocks. These color blocks are then compared with the color blocks of the foundation found through Euclidean distance calculations, which find the closest distance between the dataset and the dominant color. This aims to show that the dominant color is similar to the recommended color.

CONCLUSION

According to the findings, a suitable foundation color based on skin tone in the jawline area may be calculated using OpenCV, K-Means, and Euclidean Distance. OpenCV helps to properly detect the jawline position using the Dlib library, allowing for speedy and exact identification in that area. This is demonstrated by the jawline coordinates in 100 front-facing image datasets, which had no deviations greater than 10 on either the X or Y axis, and was done in 49 seconds.

After determining the location of the jawline, the dominant skin tone was obtained from the area surrounding the line. The dominating color is extracted by dividing the color into two clusters using K-Means, which produces the highest Silhouette Score when compared to other cluster numbers. Next, the skin color in the jawline area is compared to 625 foundation color data, and foundation color recommendations are made depending on the closest distance. The foundation color with the least Euclidean distance is deemed the best match for the user's skin tone in the jawline area.

For future researchers, it is expected that the data can be expanded by adding various types of base makeup products available domestically, not limited to just foundation. By including other products such as cushions, loose powders, compact powders, and other types of base makeup, the color recommendation system can become more comprehensive and relevant for users with diverse product preferences. This will help consumers find base makeup products that match their skin tone and simplify the selection process, thereby increasing the benefits of this application for users in Indonesia.

REFERENCES

[1] A. W. Saputra, S. Munawaroh, L. Sulkah, R. Siwi, and A. R. Putri, “Makeup Training for Teenagers as a Form of Self-Confidence,” vol. 2, no. 1, 2024.

[2] Y. Chang and N. Mukai, “Color Feature Based Dominant Color Extraction,” IEEE Access, vol. 10, no. September, pp. 93055–93061, 2022, doi: 10.1109/ACCESS.2022.3202632.

[3] J. Jing, X. Pinghua, S. Xiaowan, C. Jingwen, and L. Ruibing, “Automatic coloration of pattern based on color parsing of Sung porcelain,” Text. Res. J., vol. 92, no. 23–24, pp. 5066–5079, 2022, doi: 10.1177/00405175221113088.

[4] R. A. Rejón Piña and C. Ma, “Classification Algorithm for Skin Color (CASCo): A new tool to measure skin color in social science research,” Soc. Sci. Q., vol. 104, no. 2, pp. 168–179, 2023, doi: 10.1111/ssqu.13242.

[5] P. S. Atram, P. P. M. Chawan, and K. Means, “Finding Dominant Color in the Artistic Painting using Data Mining Technique,” no. May, pp. 262–266, 2020.

[6] A. Oyinkansola Gladys and A. Solomon Olalekan, “A machine learning model for predicting colour trends in the textile fashion industry in south-west Nigeria,” Int. J. Integr. Educ., vol. 4, no. 2, pp. 174–188, 2021, [Online]. Available: www.masterclass.com

[7] T. W. Harjanti, H. Setiyani, J. Trianto, and Y. Rahmanto, “Classification of Mint Leaf Types Based on the Image Using Euclidean Distance and K-Means Clustering with Shape and Texture Feature Extraction,” Tech-E, vol. 5, no. 2, pp. 115–124, 2022, doi: 10.31253/te.v5i1.940.

[8] H. I. Weller, S. M. Van Belleghem, A. E. Hiller, and N. P. Lord, “Flexible color segmentation of biological images with the R package recolorize,” bioRxiv, p. 2022.04.03.486906, 2022, [Online]. Available: https://www.biorxiv.org/content/10.1101/2022.04.03.486906v3%0Ahttps://www.biorxiv.org/content/10.1101/2022.04.03.486906v3.abstract

[9] H. Farid, “An Overview of Perceptual Hashing,” J. Online Trust Saf., vol. 1, no. 1, 2021, doi: 10.54501/jots.v1i1.24.

[10] H. Chugh et al., “Image Retrieval Using Different Distance Methods and Color Difference Histogram Descriptor for Human Healthcare,” J. Healthc. Eng., vol. 2022, 2022, doi: 10.1155/2022/9523009.

[11] J. Patel, A. Shetty, A. Vishwakarma, and V. Shelke, “Automated Face Detection And Swapping In Video,” pp. 2–9, 2021, doi: 10.4108/eai.16-5-2020.2303961.

[12] A. B. Afnenda, S. Sukoriyanto, and I. N. Parta, “Miskonsepsi Mahasiswa dalam Menyelesaikan Masalah Standar Deviasi Ditinjau dari Tipe Kepribadian Influence,” J. Cendekia J. Pendidik. Mat., vol. 7, no. 2, pp. 1469–1481, 2023, doi: 10.31004/cendekia.v7i2.2190.

[13] K. R. Shahapure and C. Nicholas, “Cluster quality analysis using silhouette score,” Proc. - 2020 IEEE 7th Int. Conf. Data Sci. Adv. Anal. DSAA 2020, pp. 747–748, 2020, doi: 10.1109/DSAA49011.2020.00096