Aplikasi Interaktif untuk Klasterisasi Warna Gambar Berbasis Streamlit dan K-Means

**Alifia Ananda Putri1, Febriyani Nurhida2**

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Pelita Bangsa

E-mail: [alifiaaa06@mhs.pelitabangsa.ac.id1,](mailto:alifiaaa06@mhs.pelitabangsa.ac.id1,%20) [febriyaninurhida@mhs.pelitabangsa.ac.id2](mailto:febriyaninurhida@mhs.pelitabangsa.ac.id2%20)

|  |  |  |
| --- | --- | --- |
| **Article Info** |  | **ABSTRACT** |
| ***Article history:***  Received month dd, yyyy  Revised month dd, yyyy  Accepted month dd, yyyy |  | Kebutuhan akan teknologi canggih untuk analisis data visual meningkat sebagai akibat dari pertumbuhan pesat pemrosesan citra digital. Studi ini memperkenalkan aplikasi interaktif untuk klasterisasi warna gambar yang menggunakan algoritma Streamlit dan K-Means. Algoritma K-Means mengelompokkan warna gambar berdasarkan kesamaan, sedangkan Streamlit membuat antarmuka pengguna lebih mudah digunakan. Peneliti, profesional pencitraan digital, dan desainer grafis dan fotografi akan menemukan aplikasi ini bermanfaat. Ada penjelasan tentang proses pengembangan aplikasi, penggunaan algoritma, dan evaluasi efektivitasnya. Tujuannya adalah untuk menyediakan alat klasterisasi warna yang efektif dan mudah digunakan untuk berbagai pengguna. |
| ***Keywords:***  Aplikasi Interaktif  Klasterisasi Warna  Streamlit  K-Means  Pemrosesan Citra Digital |
| *This is an open access article under the* [*CC BY-SA*](https://creativecommons.org/licenses/by-sa/4.0/) *license.* |
| ***Corresponding Author:***  Tanweer Alam  Department of Computer Science, Faculty of Computer and Information Technology  Islamic University of Madinah Abo Bakr Al Siddiq, Al Jamiah, Medina 42351, Saudi Arabia Email: tanweer03@iu.edu.sa | | |

1. **INTRODUCTION**

Pemrosesan citra digital merupakan salah satu bidang kecerdasan buatan yang berkembang pesat dalam beberapa dekade terakhir. Seiring bertambahnya jumlah data berupa gambar dan video, kebutuhan akan teknologi canggih untuk mengelola dan menganalisis data tersebut menjadi semakin penting. Salah satu teknik yang digunakan dalam pemrosesan gambar adalah pengelompokan warna, yang dapat membantu dalam berbagai aplikasi mulai dari segmentasi gambar hingga pengenalan objek.Tujuan dari aplikasi pengelompokan warna gambar interaktif berdasarkan Streamlit dan K-Means yang dikembangkan dalam penelitian ini adalah untuk menyediakan alat yang mudah digunakan kepada pengguna untuk analisis warna gambar. Sebagai platform yang memfasilitasi pembuatan aplikasi ilmu data berbasis web, Streamlit memungkinkan pengembangan antarmuka pengguna yang intuitif dan interaktif. Pada saat yang sama, algoritma K-Means yang merupakan salah satu algoritma clustering paling populer digunakan untuk mengelompokkan warna gambar menjadi beberapa kelompok berdasarkan kesamaan warna.[1]

.

Menggabungkan Streamlit dan K-Means, aplikasi ini diharapkan dapat memberikan solusi yang efektif dan efisien pada proses pengelompokan warna. Aplikasi ini tidak hanya berguna bagi para peneliti dan profesional pencitraan digital, tetapi juga dapat digunakan dalam bidang kreatif seperti desain grafis dan fotografi, dimana analisis dan manipulasi warna sangatlah penting.[1]. Kajian ini membahas tentang latar belakang teori pengolahan citra digital, pengelompokan warna dan algoritma K-Means. Selanjutnya dijelaskan proses pengembangan aplikasi dengan Streamlit dan implementasi algoritma K-Means[2].

Hasil dan analisis penggunaan aplikasi ini disajikan untuk menunjukkan efektivitas dan manfaat aplikasi yang dikembangkan..Dengan adanya penelitian ini, kami berharap dapat berkontribusi dalam bidang pengolahan citra digital, khususnya dengan mengembangkan aplikasi pengelompokan warna gambar yang mudah digunakan dan efisien. Hasil penelitian ini diharapkan dapat memberikan wawasan baru dan alat yang berguna bagi para peneliti, profesional industri, dan pengembang untuk penerapan kelompok warna yang efektif dan efisien[2].

1. **METHOD**
2. **Pengumpulan Data**

Pengumpulan data Tahap pertama penelitian ini adalah pengumpulan data dari gambar digital. Data yang digunakan adalah gambar yang diambil dengan kamera digital dari berbagai sumber. Setiap gambar diproses menggunakan algoritma K-Means untuk pengelompokan warna. Proses akuisisi ini meliputi pencahayaan, latar belakang dan subjek serta posisi kamera untuk menjamin kualitas gambar yang baik[3].

1. **Preprocessing**

Setelah data citra digital diperoleh, langkah selanjutnya adalah preprocessing. Langkah ini meliputi:

* Konversi warna: Gambar yang diperoleh dalam format RGB dikonversi ke format HSV untuk memudahkan proses pengelompokan warna.
* Normalisasi: Nilai piksel gambar dinormalisasi untuk memastikan konsistensi dalam analisis lebih lanjut.

1. **Menerapkan algoritma K-Means**

* Algoritma K-Means digunakan untuk mengelompokkan warna pada setiap gambar. Langkah ini meliputi:
* Inisialisasi centroid: Menetapkan jumlah cluster (k) dan mengatur ulang asal centroid secara acak.
* Iterasi K-Means: Mengulangi proses pengelompokan warna berdasarkan jarak Euclidean dari pusat massa tertentu hingga menyatu, yaitu ketika posisi pusat massa tidak lagi berubah secara signifikan.
* Penentuan cluster akhir: Hasil akhir dari iterasi digunakan untuk menentukan warna dominan pada gambar[3].

1. **Mengembangkan aplikasi dengan Streamlit**

Aplikasi interaktif dikembangkan dengan Streamlit untuk memudahkan pengguna dalam mengelompokkan warna pada gambar. Langkah-langkah pengembangan aplikasi meliputi:

* Desain Antarmuka Pengguna: Membuat antarmuka pengguna yang intuitif dan mudah digunakan. Mengintegrasikan K-Means dengan Streamlit: Integrasikan algoritma K-Means dengan Streamlit sehingga pengguna dapat mengunggah gambar dan melihat hasil pengelompokan secara real-time.
* Pengujian dan Debugging: Menjalankan pengujian untuk memastikan aplikasi berfungsi dengan baik dan melakukan debug jika terjadi kesalahan[3].

1. **Evaluasi dan analisis hasil**

Setelah aplikasi dikembangkan, langkah selanjutnya adalah mengevaluasi dan menganalisis hasil kelompok warna. Langkah ini meliputi:

* Pengujian aplikasi: menggunakan beberapa set gambar untuk menguji keakuratan dan konsistensi hasil pengelompokan warna.
* Analisis hasil cluster: Evaluasi kualitas cluster berdasarkan warna dominan yang dihasilkan dan bandingkan dengan warna asli gambar[3].

1. **RESULTS AND DISCUSSION**

Pengelompokan warna dengan algoritma gambar K-Means. Proses ini melibatkan beberapa langkah yang meliputi membaca dan memproses gambar terlebih dahulu, menerapkan algoritma K-Means, dan memvisualisasikan hasil clustering. Berikut adalah penjelasan rinci dari setiap langkah yang digunakan dalam kode yang diberikan.

**3.1. Pembacaan dan Konversi Warna Gambar**

Pertama, gambar dibaca menggunakan OpenCV dan kemudian dikonversi dari format BGR (default OpenCV) ke RGB untuk keperluan visualisasi yang lebih tepat di matplotlib.

*import numpy as np*

*import matplotlib.pyplot as plt*

*import cv2*

*%matplotlib inline*

*# Membaca gambar sesuai dengan yang dimiliki*

*image = cv2.imread('images/monarch.jpg')*

*# Mengubah warna menjadi RGB (dari BGR)*

*image = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)*

*plt.imshow(image)*

Langkah berikutnya adalah mengubah gambar menjadi susunan piksel 2D dengan masing-masing piksel memiliki 3 nilai warna (RGB).

*# Membentuk ulang gambar menjadi susunan piksel 2D dengan 3 nilai warna (RGB)*

*pixel\_vals = image.reshape((-1, 3))*

*# Mengkonversikan ke tipe float*

*pixel\_vals = np.float32(pixel\_vals)*

Menentukan kriteria untuk algoritma K-Means agar berhenti berjalan setelah mencapai jumlah iterasi tertentu atau tingkat akurasi yang diinginkan, kemudian melaksanakan klasterisasi dengan jumlah klaster yang ditetapkan.

*# Menentukan kriteria untuk algoritme berhenti berjalan*

*criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 100, 0.85)*

*# Melakukan k-means clustering dengan jumlah cluster yang ditetapkan sebagai 3*

*k = 3*

*retval, labels, centers = cv2.kmeans(pixel\_vals, k, None, criteria, 10, cv2.KMEANS\_RANDOM\_CENTERS)*

Setelah klasterisasi selesai, data dikonversi kembali ke nilai 8-bit dan hasil klasterisasi divisualisasikan dengan membentuk ulang data ke dimensi gambar asli.

*# Mengonversi data menjadi nilai 8-bit*

*centers = np.uint8(centers)*

*segmented\_data = centers[labels.flatten()]*

*# Membentuk ulang data menjadi dimensi gambar asli*

*segmented\_image = segmented\_data.reshape((image.shape))*

*plt.imshow(segmented\_image)*

Melalui langkah-langkah di atas, gambar telah berhasil diklasterisasi menjadi tiga kelompok warna menggunakan algoritma K-Means. Proses ini melibatkan pembacaan gambar, pra-pengolahan, penerapan K-Means, dan visualisasi hasil.

**3.2. Implementasi project streamlite**

Aplikasi ini diimplementasikan menggunakan Streamlit untuk antarmuka pengguna dan OpenCV untuk pemrosesan gambar. Pengguna dapat mengatur jumlah cluster (kkk) menggunakan slider dan menekan tombol untuk menjalankan segmentasi. Hasil segmentasi ditampilkan dalam bentuk gambar yang telah dikelompokkan berdasarkan warna.

3.2.1. Build App Klasterisasi Warna pada gambar

import streamlit as st

import numpy as np

import cv2

import matplotlib.pyplot as plt

from PIL import Image

def segment\_image(image, k, max\_iter=100, epsilon=0.85):

# Ubah warna gambar dari RGB ke BGR untuk cv2

image = cv2.cvtColor(image, cv2.COLOR\_RGB2BGR)

# Membentuk ulang gambar menjadi susunan piksel 2D dan 3 nilai warna (RGB)

pixel\_vals = image.reshape((-1, 3))

# Mengkonversikan ke tipe float

pixel\_vals = np.float32(pixel\_vals)

# Menentukan kriteria agar algoritme berhenti berjalan

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, max\_iter, epsilon)

# Melakukan k-means clustering

retval, labels, centers = cv2.kmeans(pixel\_vals, k, None, criteria, 10, cv2.KMEANS\_RANDOM\_CENTERS)

# Mengonversi data menjadi nilai 8-bit

centers = np.uint8(centers)

segmented\_data = centers[labels.flatten()]

# Membentuk ulang data menjadi dimensi gambar asli

segmented\_image = segmented\_data.reshape((image.shape))

# Ubah kembali warna gambar dari BGR ke RGB untuk ditampilkan

segmented\_image = cv2.cvtColor(segmented\_image, cv2.COLOR\_BGR2RGB)

return segmented\_image

st.title("Segmentasi Gambar Menggunakan K-Means Clustering")

st.write("Unggah gambar dan sesuaikan parameter untuk segmentasi gambar.")

# Unggah gambar

uploaded\_file = st.file\_uploader("Pilih file gambar", type=["jpg", "jpeg", "png"])

if uploaded\_file is not None:

image = np.array(Image.open(uploaded\_file))

st.image(image, caption='Gambar Asli', use\_column\_width=True)

# Slider untuk memilih jumlah cluster (k)

k = st.slider("Jumlah Cluster (k)", min\_value=2, max\_value=10, value=3)

# Tombol untuk menjalankan segmentasi

if st.button("Segmentasikan Gambar"):

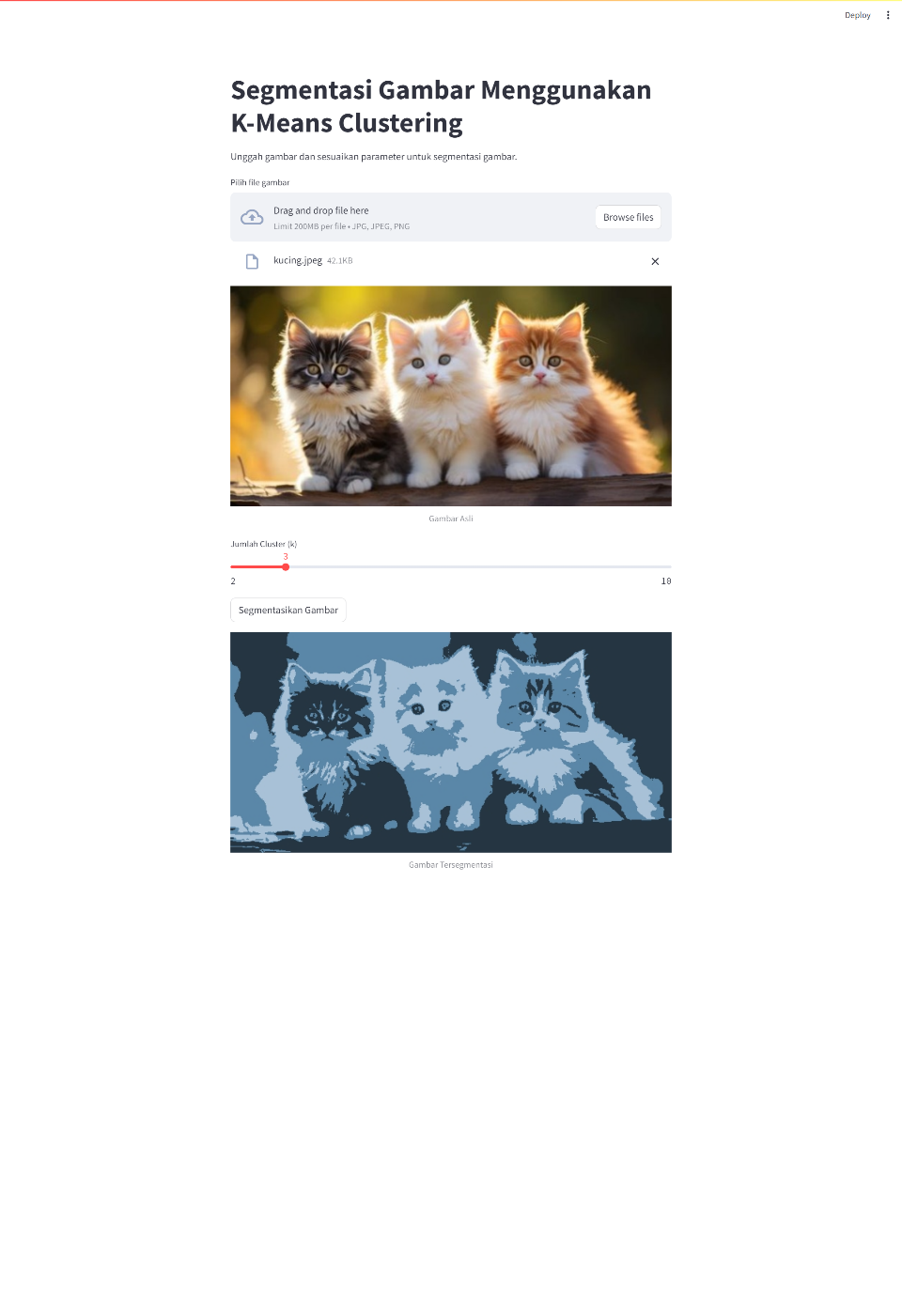
segmented\_image = segment\_image(image, k)

st.image(segmented\_image, caption='Gambar Tersegmentasi', use\_column\_width=True)

# Menjalankan aplikasi Streamlit

if \_\_name\_\_ == '\_\_main\_\_':

st.set\_option('deprecation.showPyplotGlobalUse', False)



Gambar 1. Aplikasi Streamlite Klasterisasi Warna

3.2.2. Hasil project Streamlite Klaterisasi Warna

Hasil dari klasterisasi warna pada gambar anak kucing ini menghasilkan gambar yang dibagi menjadi beberapa area warna dominan. Pada gambar ini, kita dapat melihat bahwa warna-warna yang berbeda telah dikelompokkan menjadi kluster-kluster yang terpisah, yang membuat detail gambar menjadi lebih sederhana dan terfokus pada warna-warna utama.

** **

Gambar 1. Sesudah Klasterisasi Warna Gambar 2. Sesudah Klasterisasi Warna

1. **CONCLUSION**

Proyek ini berhasil mengimplementasikan algoritma K-Means Clustering untuk segmentasi gambar menggunakan Streamlit dan OpenCV. Dengan antarmuka yang intuitif, pengguna dapat dengan mudah mengunggah gambar dan menyesuaikan parameter segmentasi untuk melihat hasil yang berbeda. Proyek ini menunjukkan bahwa K-Means Clustering adalah metode yang efektif untuk memisahkan gambar menjadi beberapa bagian berdasarkan karakteristik warna.

**ACKNOWLEDGEMENTS**

Berdasarkan hasil dan pengalaman proyek ini, berikut beberapa rekomendasi untuk pengembangan lebih lanjut:

1. **Optimasi algoritma:** Pertimbangkan untuk mengoptimalkan parameter algoritma clustering K-Means untuk meningkatkan kecepatan dan akurasi segmentasi. Investigasi lebih lanjut terhadap metode pengoptimalan seperti Mini-Batch K-Means mungkin bisa membantu.
2. **Tambahkan metode segmentasi:** Anda dapat memberikan lebih banyak fleksibilitas kepada pengguna dengan menambahkan metode segmentasi lainnya, seperti rata-rata bergerak atau daerah aliran sungai, yang mungkin lebih efektif untuk jenis gambar tertentu.
3. **Peningkatan antarmuka pengguna:** Memperluas antarmuka pengguna dengan fitur tambahan seperti memperbesar dan menggeser gambar yang diunggah, serta menyesuaikan parameter lanjutan untuk segmentasi yang lebih akurat.

**REFERENCES**

[1] F. Dona Marleny, “Pengolahan Citra Digital Menggunakan Python.” [Online]. Available: https://www.researchgate.net/publication/358220979

[2] A. D. Goenawan, M. Bakhara, A. Rachman, and M. P. Pulungan, “Januari 2022 hal 68-74 Menara 165, Lantai 18 & 19,” *Jl. Tb. Simatupang Kav*, vol. 4, no. 1, [Online]. Available: https://jurnal.ugp.ac.id/index.php/jutei

[3] R. I. Armianti, A. Fanany, O. Gaffar, A. Bramanto, W. Putra, and P. Korespondensi, “PENERAPAN K-MEANS CLUSTERING UNTUK SELEKSI FRAME DOMINAN BERBASIS NTSC PADA OBYEK BERGERAK APPLICATION OF K-MEANS CLUSTERING FOR NTSC-BASED DOMINANT FRAME SELECTION ON MOVING OBJECTS,” vol. 7, no. 4, 2020, doi: 10.25126/jtiik.202072184.