

Laporan Tugas Besar II
IF2123 Aljabar Linier dan Geometri
Aplikasi Aljabar Vektor dalam Sistem Temu Balik Gambar



Disusun Oleh

Kelompok Cipung

13522009 Muhammad Yusuf Rafi

13522076 Muhammad Syarafi Akmal

13522105 Fabian Radenta Bangun

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

Semester I - 2023

Daftar Isi

Daftar Isi.....	1
Bab 1.....	2
Desripsi Masalah.....	2
Bab 2.....	2
Landasan Teori.....	2
2.1. Matrix.....	2
2.2. Cosine Similarity.....	3
2.3. Content-Based Image Retrieval (CBIR).....	3
2.4. CBIR Dengan Parameter Warna.....	4
2.5. CBIR Dengan Parameter Tekstur.....	4
2.6. Pengembangan Website.....	5
Bab 3.....	6
Analisis Pemecahan Masalah.....	6
3.1. Perbandingan Citra dengan Content-Based Image Retrieval (CBIR) dengan Parameter Warna.....	6
3.2. Perbandingan Citra dengan Content-Based Image Retrieval (CBIR) dengan Parameter Tekstur.....	7
Bab 4.....	8
Implementasi dan Uji Coba.....	8
4.1. Implementasi Program Utama.....	8
4.2. Struktur Program.....	10
4.3. Tampilan Awal Website.....	10
4.4. Pengujian Program.....	12
4.5. Analisis Hasil Uji.....	16
Bab 5.....	17
Penutup.....	17
5.1. Kesimpulan.....	17
5.2. Saran.....	17
5.3. Komentar & Refleksi.....	18
Bab 6.....	18
Daftar Referensi.....	18
Lampiran.....	19

Bab 1

Deskripsi Masalah

Dalam era digital, jumlah gambar yang dihasilkan dan disimpan semakin meningkat dengan pesat, baik dalam konteks pribadi maupun profesional. Peningkatan ini mencakup berbagai jenis gambar, mulai dari foto pribadi, gambar medis, ilustrasi ilmiah, hingga gambar komersial. Terlepas dari keragaman sumber dan jenis gambar ini, sistem temu balik gambar (image retrieval system) menjadi sangat relevan dan penting dalam menghadapi tantangan ini. Dengan bantuan sistem temu balik gambar, pengguna dapat dengan mudah mencari, mengakses, dan mengelola koleksi gambar mereka. Sistem ini memungkinkan pengguna untuk menjelajahi informasi visual yang tersimpan di berbagai platform, baik itu dalam bentuk pencarian gambar pribadi, analisis gambar medis untuk diagnosis, pencarian ilustrasi ilmiah, hingga pencarian produk berdasarkan gambar komersial.

Bab 2

Landasan Teori

2.1. Matrix

Matriks adalah suatu susunan bilangan real atau bilangan kompleks (atau elemen-elemen) yang disusun dalam baris dan kolom sehingga membentuk jajaran persegi panjang. *Ukuran* (ordo) suatu matriks dijelaskan dengan menyatakan banyaknya baris (garis horisontal) dan banyaknya kolom (garis vertikal) yang terdapat dalam matriks tersebut.

Notasi nama matriks biasanya menggunakan huruf kapital dan cetak tebal. Jika A adalah sebuah matriks, maka digunakan a_{ij} untuk menyatakan entri atau elemen yang terdapat di dalam baris i dan kolom j dari A. Jadi, matriks dengan ukuran $m \times n$ beserta entri-entrinya secara umum dapat dituliskan sebagai berikut :

$$A_{(m \times n)} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \text{ atau } [a_{ij}]_{m \times n}$$

2.2. Cosine Similarity

Cosine similarity merupakan konsep utama dalam menilai sejauh mana dua vektor non-nol mirip satu sama lain, yang didefinisikan dalam ruang produk dalam. Similaritas cosinus diperoleh dengan mengalikan titik (dot product) dari kedua vektor tersebut, lalu hasilnya dibagi dengan perkalian panjang (norm) masing-masing vektor. Artinya, similaritas cosinus mengukur seberapa searah kedua vektor berada dalam ruang vektor.

Proses ini dimulai dengan menghitung perkalian titik antara dua vektor, yaitu jumlah dari hasil perkalian elemen-elemen yang sesuai dari keduanya. Selanjutnya, panjang dari setiap vektor dihitung, dan hasil perkalian titik dibagi oleh hasil perkalian panjang vektor-vektor tersebut. Hasil perhitungan ini menghasilkan nilai antara -1 dan 1, di mana nilai 1 menunjukkan kesamaan sempurna, nilai -1 menunjukkan ketidakserupaan total, dan nilai 0 menunjukkan ketidakserupaan relatif.

Kelebihan dari cosine similarity adalah metrik ini tidak bergantung pada magnitudo vektor, melainkan hanya pada sudut antara vektor-vektor tersebut. Oleh karena itu, cosine similarity banyak digunakan dalam berbagai aplikasi, seperti sistem temu balik informasi, klasifikasi dokumen, dan analisis teks, di mana kesamaan antara dokumen atau teks dapat diukur tanpa memperhatikan magnitudo dari vektor representasi dokumen atau kata. Dengan memahami dasar-dasar ini, cosine similarity menjadi alat yang efektif dalam mengevaluasi hubungan antar vektor dalam konteks analisis data dan pemrosesan informasi.

2.3. Content-Based Image Retrieval (CBIR)

Content Based Image Retrieval (CBIR) atau temu kenali citra merupakan suatu metode yang digunakan untuk melakukan pencarian citra digital pada suatu basis data citra. Beberapa

konten aktual pada sebuah citra yang meliputi warna, bentuk, tekstur atau informasi lain yang didapatkan dari citra tersebut merupakan objek yang dianalisa dalam proses pencarian content based. CBIR merupakan suatu aplikasi dari *computer vision* yang mempunyai teknik pencarian gambar yang diambil dari basis data yang menyediakan gambar sebagai gambar uji. Proses query gambar dilakukan dengan mengekstraksi fitur yang meliputi histogram, nilai warna, tekstur, dan deteksi tepi.

2.4. CBIR Dengan Parameter Warna

Content-Based Image Retrieval (CBIR) menggunakan parameter warna melibatkan sejumlah konsep mendasar untuk mencapai pencarian gambar berbasis konten yang efektif. Dalam CBIR, perhatian utama diberikan pada pengenalan gambar berdasarkan ciri-ciri warna yang dimiliki. Warna merupakan atribut visual yang kuat dan dapat digunakan untuk membedakan gambar-gambar dalam suatu dataset. Proses dimulai dengan mengekstraksi fitur-fitur warna dari gambar, yang mencakup representasi distribusi warna dalam ruang warna tertentu, seperti RGB atau HSV. Setelah fitur-fitur warna berhasil diekstraksi, algoritma pencocokan digunakan untuk membandingkan vektor-fitur dari gambar yang sedang dicari dengan gambar dalam dataset. Hasil dari proses pencocokan tersebut digunakan untuk mengurutkan dan menampilkan gambar-gambar yang memiliki kesamaan dalam hal warna. Dengan menerapkan teori dasar ini, CBIR dengan mempertimbangkan parameter warna dapat menjadi alat yang efektif untuk melakukan pencarian dan manajemen koleksi gambar berdasarkan karakteristik visual warna, memberikan kemudahan dalam menjelajahi dan mengakses informasi visual.

2.5. CBIR Dengan Parameter Tekstur

Content-Based Image Retrieval (CBIR) dengan memanfaatkan parameter tekstur melibatkan sejumlah konsep kunci untuk mencapai efektivitas dalam pencarian gambar berbasis konten. Dalam CBIR, perhatian utama tertuju pada pengidentifikasi gambar berdasarkan ciri-ciri tekstur yang dimilikinya. Tekstur mencakup pola-pola spasial dalam citra yang dapat diidentifikasi dan dianalisis untuk membentuk deskripsi numerik, yang nantinya dapat dibandingkan dengan gambar-gambar lain dalam dataset. Proses ini dimulai dengan ekstraksi fitur tekstur menggunakan metode-metode seperti matrix co-occurrence atau transformasi

wavelet. Matrix co-occurrence digunakan untuk mengukur distribusi intensitas piksel dalam konteks spasial, sementara transformasi wavelet digunakan untuk memecah informasi frekuensi tinggi dan rendah dalam citra. Setelah fitur tekstur berhasil diekstraksi, algoritma pencocokan dapat diterapkan untuk membandingkan vektor-fitur dari gambar yang sedang dicari dengan gambar dalam dataset. Hasil dari proses pencocokan ini kemudian digunakan untuk mengurutkan dan menampilkan gambar-gambar yang memiliki kesamaan dalam hal tekstur. Dengan mengimplementasikan teori dasar ini, CBIR dengan mempertimbangkan parameter tekstur menjadi alat yang efektif untuk melakukan pencarian dan manajemen koleksi gambar berdasarkan ciri-ciri tekstur visualnya.

2.6. Pengembangan *Website*

Pengembangan website melibatkan serangkaian prinsip dan tahapan esensial guna menciptakan situs web yang berfungsi dan menarik. Awalnya, digunakan HTML (Hypertext Markup Language) untuk membentuk kerangka dasar halaman web dan menentukan elemen seperti heading, paragraph, dan gambar. CSS (Cascading Style Sheets) kemudian diterapkan untuk mengelola tata letak dan estetika visual elemen-elemen HTML, dengan tujuan memisahkan presentasi dari struktur halaman. Adopsi JavaScript memungkinkan penambahan elemen interaktif pada halaman web, memberikan tanggapan terhadap input pengguna, serta mengembangkan fitur-fitur yang dinamis. Prinsip-prinsip desain UI (User Interface) dan UX (User Experience) menjadi kunci utama dalam mencapai antarmuka yang menarik dan pengalaman pengguna yang memuaskan. Selain itu, responsif dan desain "mobile-first" menjadi fokus untuk memastikan situs dapat diakses dengan baik di berbagai perangkat. Pengembang web juga perlu memahami aspek-aspek seperti server, hosting, database, keamanan web, serta praktik pemeliharaan dan pengoptimalan agar pengembangan web dapat mencapai keberhasilan secara keseluruhan. Pemahaman protokol HTTP/HTTPS, sebagai dasar komunikasi web, juga menjadi hal yang penting. Dengan memperhatikan semua elemen ini, pengembang web dapat menciptakan situs yang tidak hanya berkinerja optimal tetapi juga memberikan pengalaman pengguna yang unggul.

Bab 3

Analisis Pemecahan Masalah

3.1. Perbandingan Citra dengan Content-Based Image Retrieval (CBIR) dengan Parameter Warna

Untuk membandingkan citra dengan metode ini, yang pertama dilakukan adalah mengkonversi warna dari RGB ke HSV. Berikut adalah prosedurnya :

1. Normalisasi nilai RGB

$$R' = \frac{R}{255} \quad G' = \frac{G}{255} \quad B' = \frac{B}{255}$$

2. Mencari C_{max} , C_{min} , dan Δ

$$C_{max} = \max(R', G', B')$$

$$C_{min} = \min(R', G', B')$$

$$\Delta = C_{max} - C_{min}$$

3. Dapatkan nilai HSV dengan persamaan berikut :

$$H = \begin{cases} 0^\circ & , C' \text{ max} = R' \\ 60^\circ \times \left(\frac{G' - B'}{\Delta} \bmod 6 \right) & , C' \text{ max} = G' \\ 60^\circ \times \left(\frac{B' - R'}{\Delta} + 2 \right) & , C' \text{ max} = B' \\ 60^\circ \times \left(\frac{R' - G'}{\Delta} + 4 \right) & \end{cases}$$
$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases}$$
$$V = C_{maks}$$

3.2. Perbandingan Citra dengan Content-Based Image Retrieval (CBIR) dengan Parameter Tekstur

Langkah pertama pada metode ini adalah mengubah warna gambar menjadi *grayscale*. Hal ini dapat dilakukan dengan persamaan :

$$Y = 0.29 \times R + 0.587 \times G + 0.114 \times B$$

Kemudian, nilai *grayscale* tersebut dikuantifikasi untuk memperoleh matrix *co-occurrence*. Karena citra *grayscale* berukuran 256 piksel, maka matriks yang berkoresponden akan berukuran 256×256 . Berikut adalah persamaan untuk mendapatkan matrix *co-occurrence* :

$$C_{\Delta x, \Delta y}(i, j) = \sum_{p=1}^n \sum_{q=1}^m \begin{cases} 1, & \text{jika } I(p, q) = i \text{ dan } I(p + \Delta x, q + \Delta y) = j \\ 0, & \text{jika lainnya} \end{cases}$$

dengan keterangan i dan j sebagai nilai intensitas dari gambar dan p serta q sebagai posisi dari gambar. Offset Δx dan Δy bergantung pada arah θ dan jarak yang digunakan. Pada program ini digunakan jarak sebesar 1 dan arah 0° . Setelah didapatkan matrix *co-occurrence*, dibuat *symmetric matrix* dengan menjumlahkan matrix *co-occurrence* dengan hasil transpose-nya. Kemudian cari *matrix normalization* dengan persamaan :

$$\text{MatrixNorm} = \frac{\text{MatrixOcc}}{\sum \text{MatrixOcc}}$$

Kemudian bentuk vektor dengan komponen ekstraksi tekstur *contrast*, *homogeneity*, dan *entropy*. Persamaan yang digunakan adalah :

Contrast:

$$\sum_{i,j=0}^{\text{dimensi}-1} P_{i,j} (i - j)^2$$

Homogeneity :

$$\sum_{i,j=0}^{\text{dimensi}-1} \frac{P_{i,j}}{1 + (i - j)^2}$$

Entropy :

$$-\left(\sum_{i,j=0}^{\text{dimensi}-1} P_{i,j} \times \log P_{i,j} \right)$$

Lalu langkah terakhir adalah hitung persentase kemiripan dari kedua gambar dengan menggunakan *Cosine Similarity*, yaitu :

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

A dan B menyatakan vektor dari dua buah gambar. Semakin besar nilainya maka semakin mirip kedua gambar tersebut.

Bab 4

Implementasi dan Uji Coba

4.1. Implementasi Program Utama

Pseudocode fungsi untuk membandingkan gambar dengan parameter tekstur pada tekstur.py

```
function compare_images(image1, image2) -> float
    grayscale1 <- image_to_grayscale(image1)
    grayscale2 <- image_to_grayscale(image2)
```

```

comatrix1 <- calculate_cooccurrence_matrix(grayscale1)
comatrix2 <- calculate_cooccurrence_matrix(grayscale2)

contrast1 <- get_contrast(comatrix1)
contrast2 <- get_contrast(comatrix2)
homogeneity1 <- get_homogeneity(comatrix1)
homogeneity2 <- get_homogeneity(comatrix2)
entropy1 <- get_entropy(comatrix1)
entropy2 <- get_entropy(comatrix2)

vectorA <- np.arrayx(contrast1, homogeneity1, entropy1)
vectorB <- np.arrayx(contrast1, homogeneity1, entropy1)

magnitude_of_A <- np.linalg.norm(vectorA)
magnitude_of_B <- np.linalg.norm(vectorB)

result <- np.dot(vectorA, vectorB)/(magnitude_of_A*magnitude_of_B)
-> result

```

Pseudocode fungsi untuk membandingkan gambar dengan parameter warna pada warna.py

```

function process_color_dataset(input_image, dataset_folder) -> list
height, width, _ <- input_image.shape
similarity_scores <- []

input_hue, input_saturation, input_value <- calculate_hsv_histogram(input_image)
i <- 1
filename traversal (dataset_folder):
if filename.endswith(".jpg") or filename.endswith(".png") then
    dataset_image_path <- os.path.join(dataset_folder, filename)
    dataset_image <- cv2.imread(dataset_image_path)

    resized_image <- cv2.resize(dataset_image, (width, height))

    dataset_hue, dataset_saturation, dataset_value <-
    calculate_hsv_histogram(resized_image)
    similarity <- calculate_block_histogram(input_image, input_hue, input_saturation,
    input_value, dataset_hue, dataset_saturation, dataset_value)

    similarity <- similarity*100
    # Simpan hasil pencocokan sebagai dictionary dalam endpoint
    if(similarity > 60.0) then

```

```

similarity_scores.append({
    "id": i,
    "persentase": round(similarity,3),
    "img": 'http://127.0.0.1:8000/media/dataset/'+filename,
    "durasi": 0 #nilai durasi di default ke-0
})
i <- i+1

sorted_results <- sorted(similarity_scores, key=-lambda x: x['persentase'], reverse=True)

-> sorted_results

```

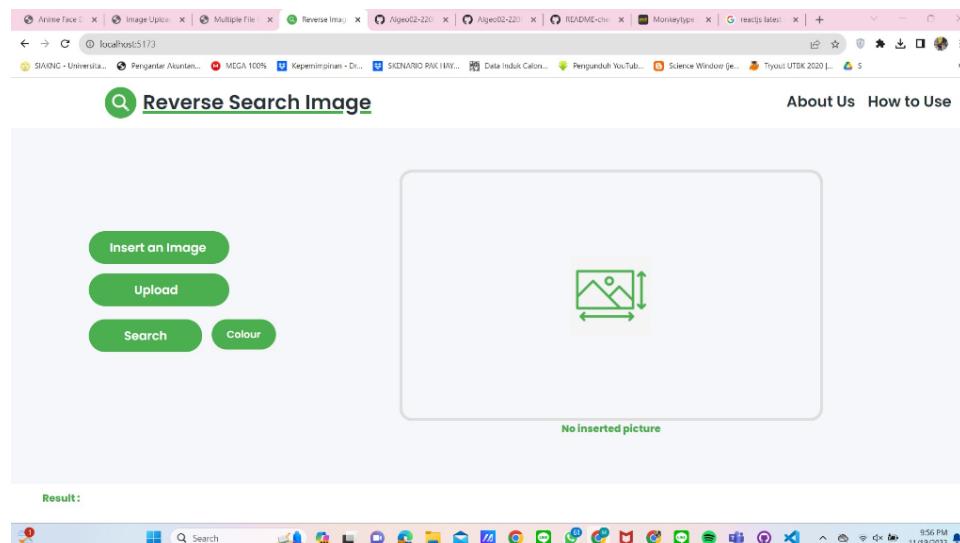
4.2. Struktur Program

```

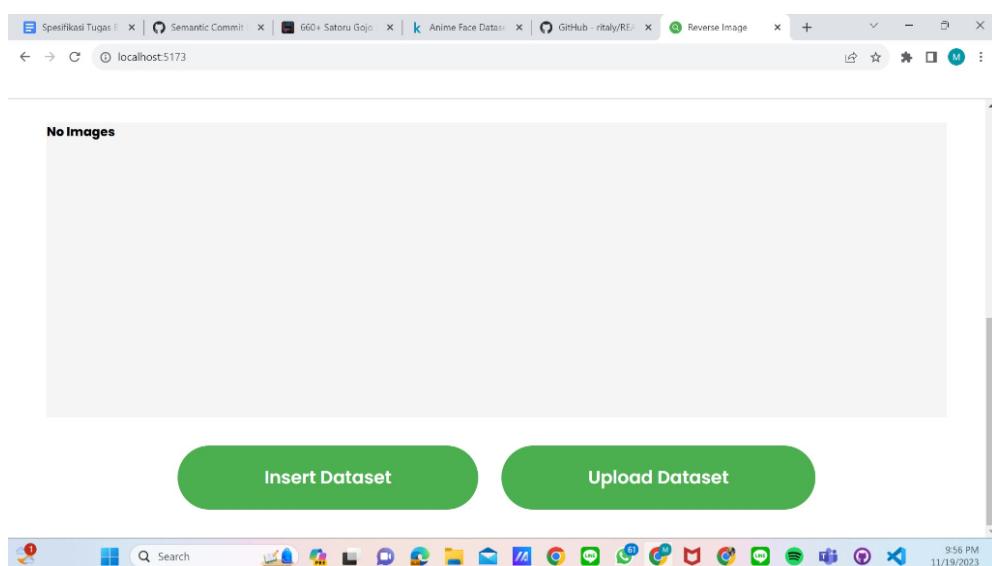
| README.md
|
|____doc          # Laporan
|____img
|____src          # Source code
|     |____Backend/project      # Backend django
|     |____FEALGEO            # Frontend react
|____test         # Gambar yang digunakan sebagai dataset

```

4.3. Tampilan Awal Website



Gambar 4.3.1 Tampilan awal website



Gambar 4.3.2 Tampilan awal website

Pada sebelah kiri atas gambar 4.3.1 ditampilkan judul web, yaitu Reverse Search Image. Kemudian di bawah judul ada 4 pilihan tombol. Yang pertama adalah “Insert image”, gunanya ialah memasukkan gambar yang akan dijadikan gambar referensi untuk dicari gambar yang serupa dengannya. Kemudian ada “upload” yang berfungsi untuk meng-upload gambar referensi yang sudah dipilih agar dapat diproses oleh *backend*. Lalu ada tombol “search” yang gunanya untuk mencari dan menampilkan gambar pada dataset yang persentase kemiripannya dengan

gambar referensi diatas 60%. Dan yang terakhir di sebelah tombol “search” ada tombol yang berguna untuk mengubah metode perhitungan yang digunakan, pilihannya antara “colour” atau “texture”. Hal ini menyatakan parameter apa yang digunakan saat proses pencarian gambar.

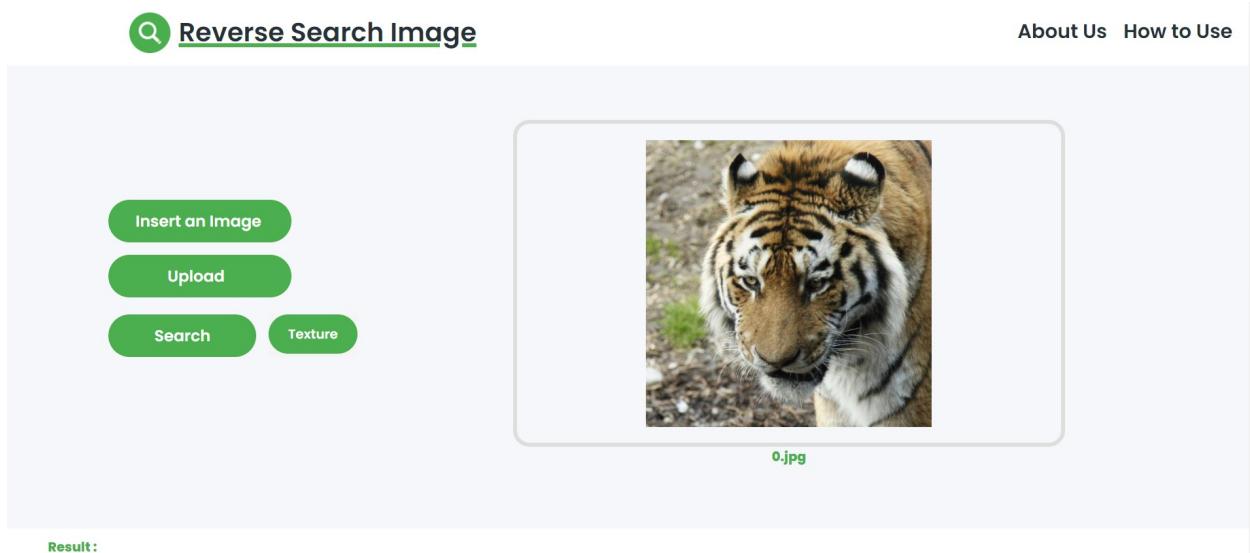
Jika di-scroll ke bagian bawah halaman awal website akan ditemukan kondisi seperti pada gambar 4.3.2. Di sebelah halaman tersebut ada dua tombol, yaitu yang pertama “Insert Dataset” untuk memasukkan dataset dan “Upload Dataset” untuk mengunggah dataset.

4.4. Pengujian Program

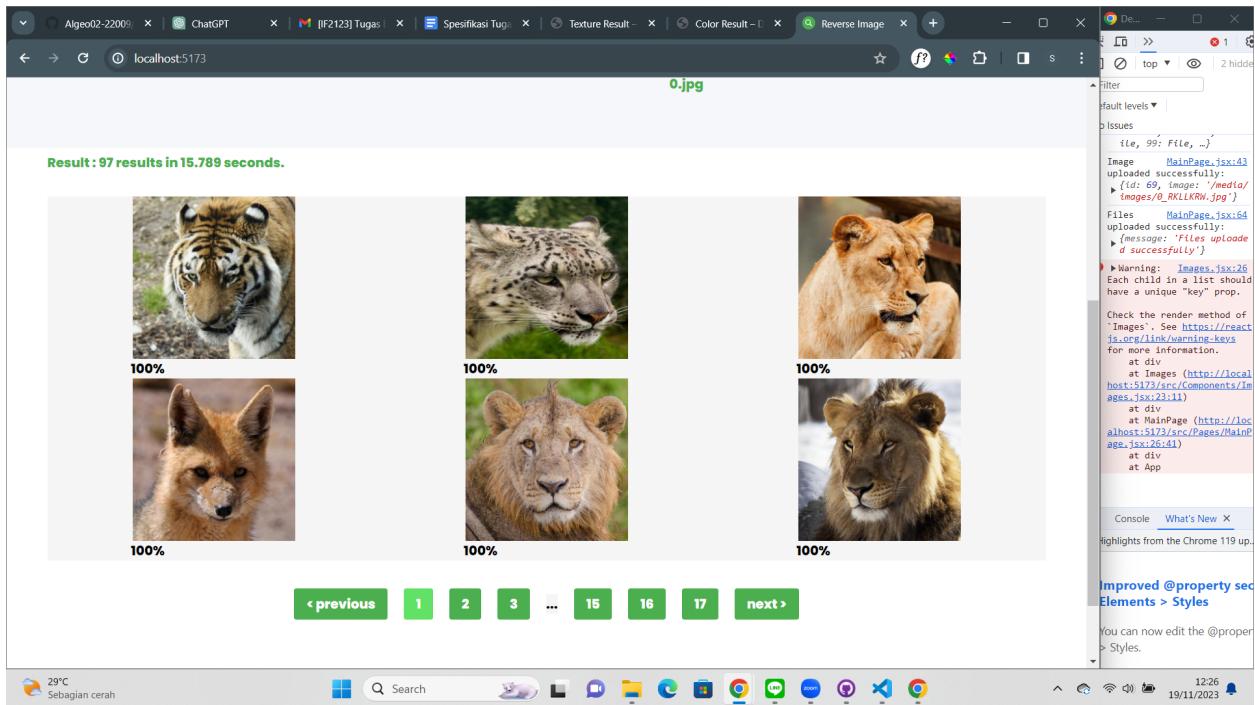
Dataset : media/dataset

Deskripsi : 100 gambar, dimensi 512 x 512

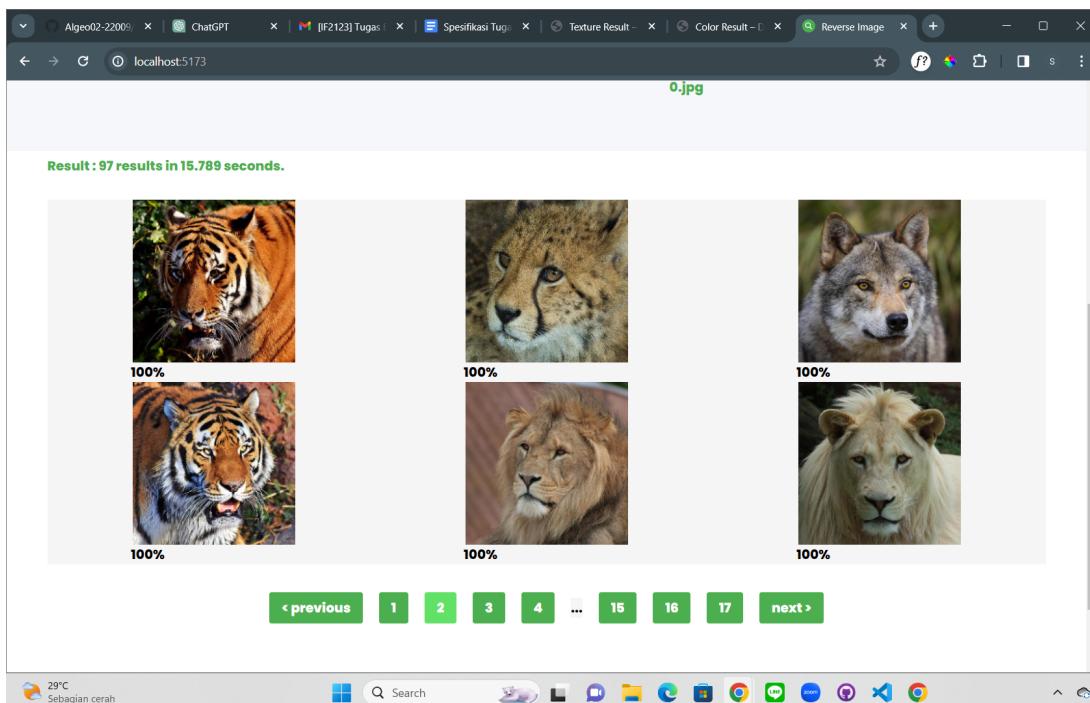
4.4.1. Set-up pengujian dengan menggunakan parameter tekstur



Gambar 4.3.3 Set-up awal pada pengujian dengan parameter tekstur

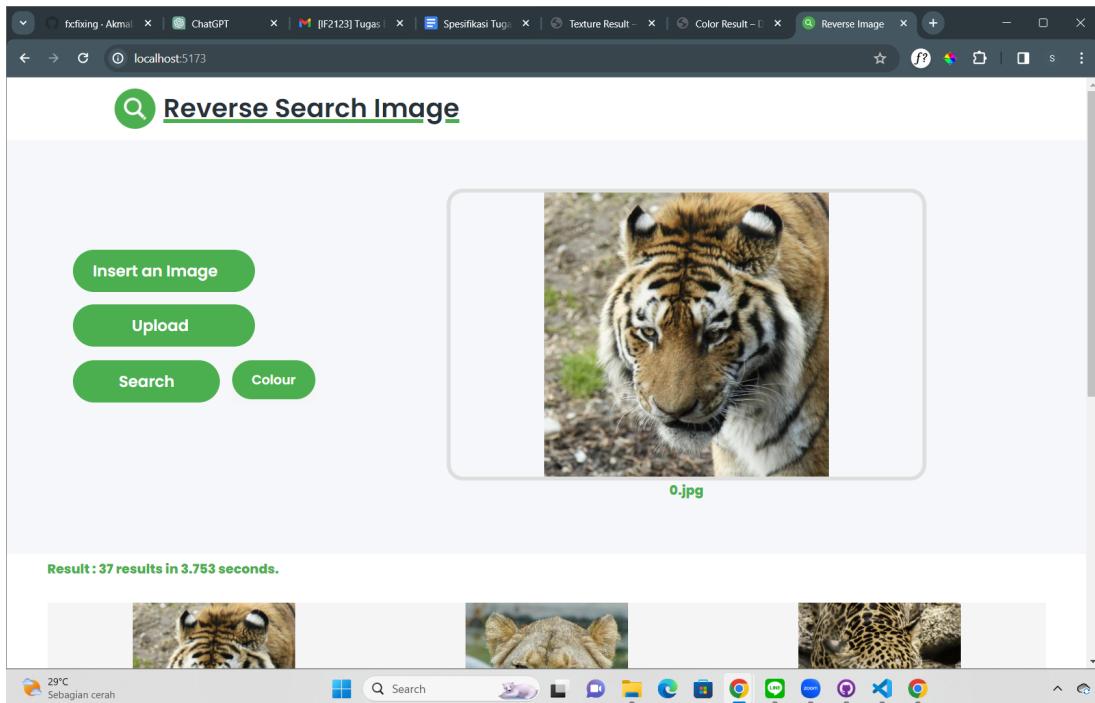


Gambar 4.3.4 Halaman pertama hasil pengujian dengan parameter tekstur



Gambar 4.3.4 Halaman kedua hasil pengujian dengan parameter tekstur

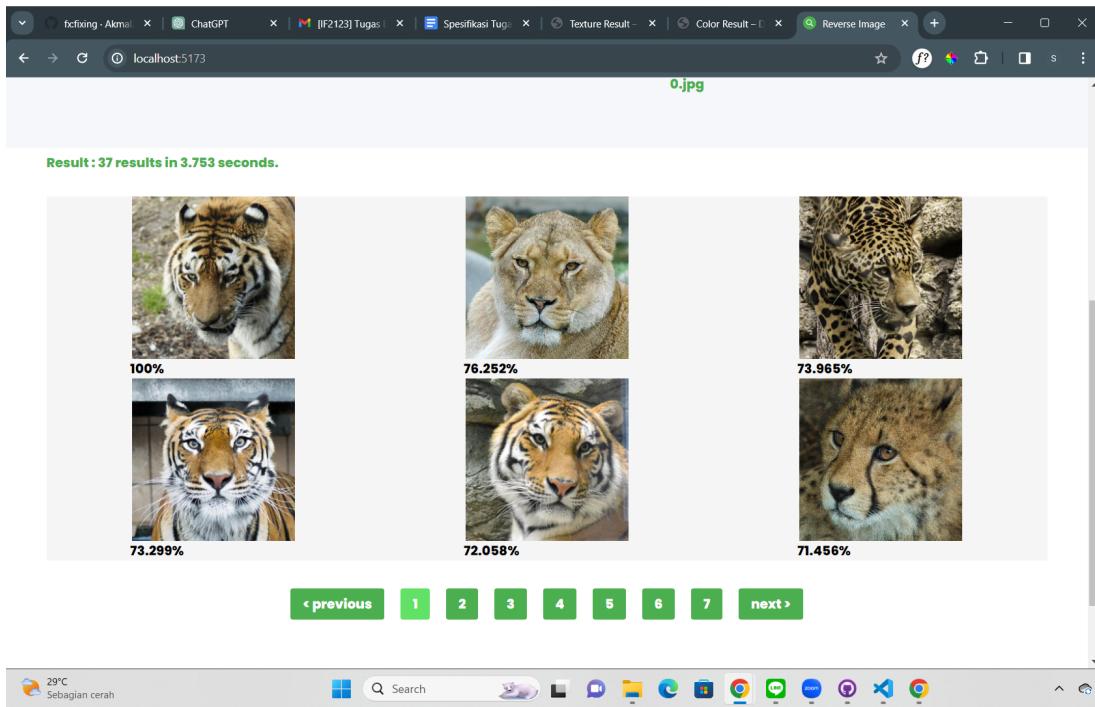
4.4.2. Set-up pengujian dengan menggunakan parameter warna



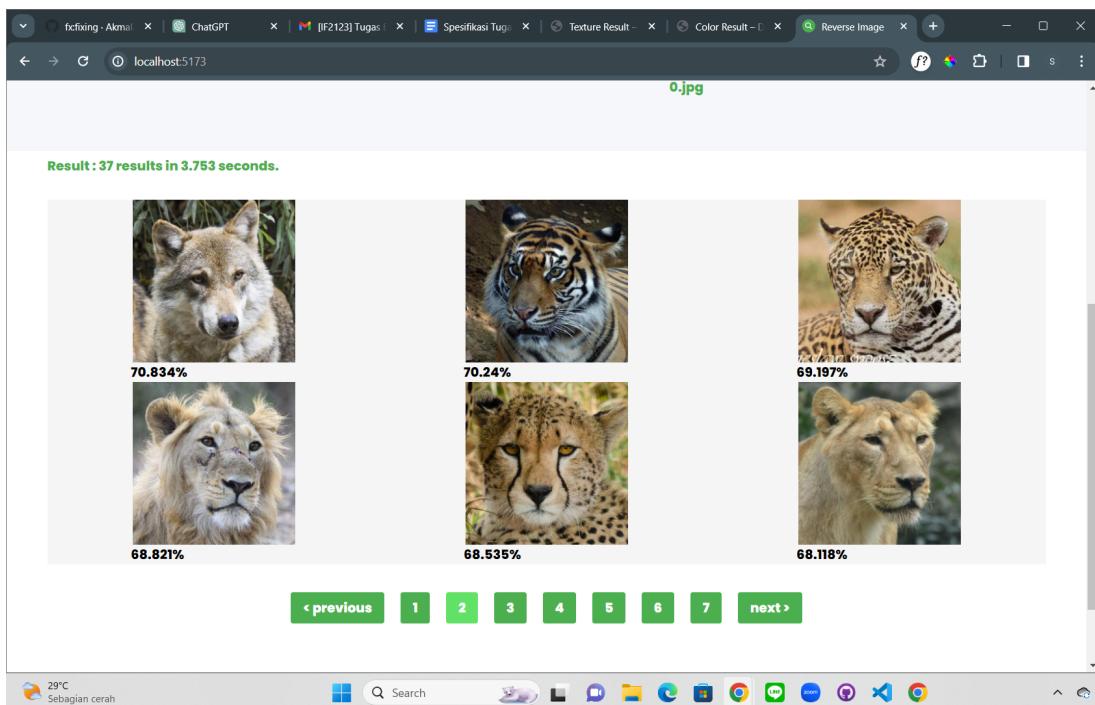
Gambar 4.3.3 Set-up awal pada pengujian dengan parameter warna

Cara penggunaan website ini adalah dengan memasukkan gambar referensi dengan tombol “Insert an Image”. Kemudian unggah gambar referensi dengan tombol “Upload”. Setelah itu pilih dan unggah dataset dengan 2 tombol di bagian bawah halaman. Pastikan parameter yang dipilih adalah warna pada pengujian kali ini. Kemudian klik tombol “Search” untuk mendapatkan hasil pengujian.

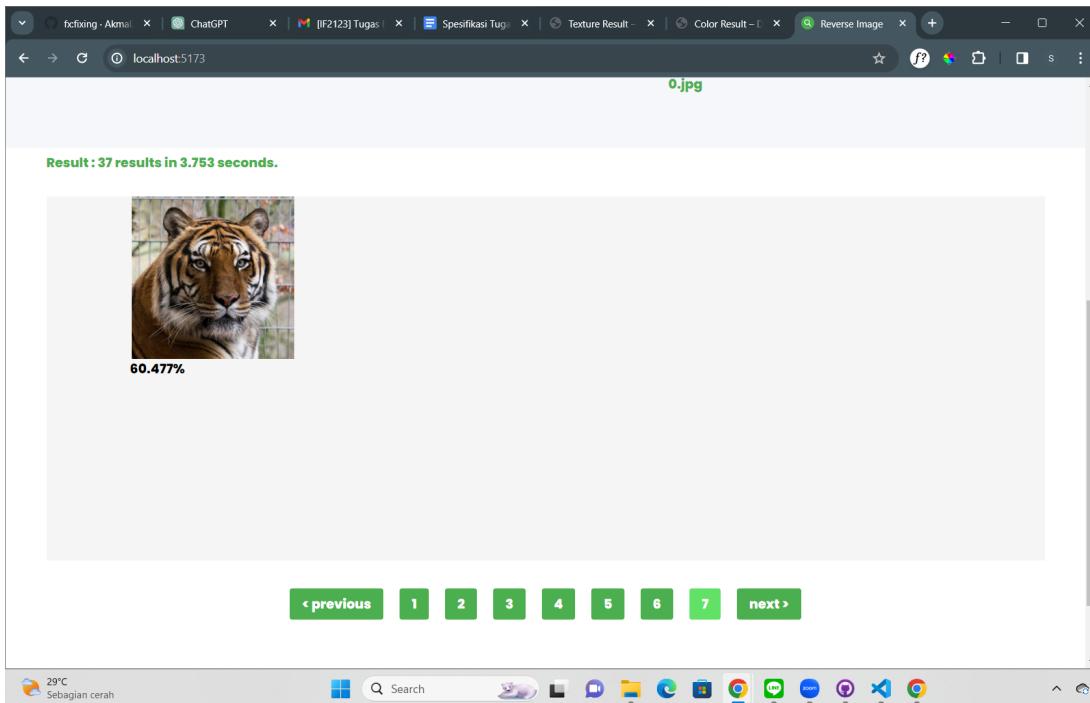
4.4.3. Hasil Uji



Gambar 4.3.4 Halaman pertama hasil pengujian dengan parameter warna



Gambar 4.3.5 Halaman kedua hasil pengujian dengan parameter warna



Gambar 4.3.6 Halaman terakhir hasil pengujian dengan parameter warna

4.5. Analisis Hasil Uji

Dari hasil pengujian yang telah dilakukan, didapati bahwa waktu eksekusi program ketika menguji gambar dengan parameter warna lebih cepat daripada dengan parameter tekstur. Hal ini terjadi karena dalam perhitungannya, vektor yang dibentuk ketika menggunakan parameter warna hanya memiliki 1 komponen, sementara pada pengujian dengan parameter tekstur vektor yang dibentuk memiliki 3 komponen.

Kemudian juga didapati pengujian gambar dengan parameter warna lebih akurat hasilnya. Hal ini disebabkan karena program untuk membandingkan gambar dengan parameter tekstur pada pengerjaan tugas ini hanya menggunakan 3 komponen ekstraksi tekstur, yaitu *contrast*, *homogeneity*, dan *entropy*. Padahal masih ada komponen ekstraksi lain yang dapat digunakan dalam perhitungan sehingga hasil perhitungannya lebih akurat. Namun, solusi ini dapat menyebabkan waktu eksekusi program menjadi lebih lama.

Bab 5

Penutup

5.1. Kesimpulan

Kesimpulan yang bisa ditarik dari tugas besar kedua Mata Kuliah Aljabar Linear dan Geometri ini adalah sebagai berikut :

- 5.1.1. Konsep Content-Based Information Retrieval dapat digunakan untuk menyatakan persentase kemiripan dua buah citra dengan pendekatan klasifikasi berbasis konten (*Content-Based Image Retrieval* atau CBIR), di mana sistem temu balik gambar bekerja dengan mengidentifikasi gambar berdasarkan konten visualnya, seperti warna dan tekstur.
- 5.1.2. Library Numpy dan OpenCV dapat membantu penyelesaian fungsi-fungsi yang dibutuhkan dalam penggerjaan *face recognition* ini.
- 5.1.3. Hasil persentase perbandingan dan waktu eksekusi program dapat bervariasi tergantung dimensi gambar, banyak gambar, dan pengaruh performansi serta *cache* yang berbeda-beda pada setiap komputer.

5.2. Saran

- 5.2.1. Untuk mendapatkan waktu eksekusi program yang lebih cepat pada perbandingan citra dengan CBIR menggunakan parameter tekstur dapat digunakan fungsi cvtColor yang disediakan pada library OpenCV untuk mengubah *image* ke *grayscale*.
- 5.2.2. Untuk mendapatkan hasil yang lebih akurat pada perbandingan citra dengan CBIR menggunakan parameter tekstur dapat ditambahkan komponen ekstraksi tekstur lainnya, tidak hanya *contrast*, *homogeneity*, *entropy*.

5.3. Komentar & Refleksi

Tugas besar ini membuat kami lebih memahami implementasi dari pelajaran yang kami dapat di Mata Kuliah Aljabar Linear dan Geometri. Tugas besar ini juga melatih keterampilan kami dalam mencari cara untuk menulis algoritma seefektif mungkin demi mendapatkan waktu eksekusi program yang cepat. Dari tugas ini juga kami belajar untuk mengatur waktu lebih baik lagi mengingat banyaknya tugas mata kuliah lain yang harus kami kerjakan.

Bab 6

Daftar Referensi

1. <https://yunusmuhammad007.medium.com/feature-extraction-gray-level-co-occurrence-matrix-glcm-10c45b6d46a1>
2. <https://dspace.uji.ac.id/bitstream/handle/123456789/5376/Laporan%20Skripsi.pdf?sequence=1&isAllowed=y>
3. <https://www.geeksforgeeks.org/what-is-information-retrieval/>
4. [https://www.baeldung.com/cs/cbir-tbir#:~:text=Content%2DBased%20Image%20Retrieval%20\(CBIR\)%20is%20a%20way%20of,image%20to%20the%20database%20images.](https://www.baeldung.com/cs/cbir-tbir#:~:text=Content%2DBased%20Image%20Retrieval%20(CBIR)%20is%20a%20way%20of,image%20to%20the%20database%20images.)
5. <http://eprints.uny.ac.id/25779/2/BAB%202.pdf>
6. <https://www.sciencedirect.com/topics/computer-science/cosine-similarity#:~:text=Cosine%20similarity%20measures%20the%20similarity.document%20similarity%20in%20text%20analysis.>
7. https://en.wikipedia.org/wiki/Cosine_similarity

Lampiran

Link Repository : <https://github.com/Akmal2205/Algeo02-22009/>

Link Video

https://drive.google.com/file/d/19ntNnKF2herUT_vJa6usV5AF6o_IWn1r/view?usp=drivesdk