LAPORAN TUGAS BESAR

IF2123 ALJABAR LINEAR DAN GEOMETRI

DIBUAT UNTUK MEMENUHI TUGAS BESAR IF2123 2023/2024

CBIR

CONTENT BASED IMAGE RETRIEVAL

Oleh

Kelompok Ottoman

13522123 - Jimly Nur A

13522141 - Ahmad Thoriq S

13522159 - Rafif Ardhinto I

Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung

Jl. Ganesha 10, Bandung 40132

DAFTAR ISI

Bab 1 - Deskripsi masalah

Spesifikasi lengkap tugas besar ini terdapat pada pranala berikut
https://docs.google.com/document/d/1HVDyywnUdNz9hStgx5ZLqHypK89hWH8qfERJOiDw6KA/edit?pli=1

Di zaman digital saat ini, jumlah gambar yang dibuat dan disimpan meningkat secara signifikan, baik dalam lingkup pribadi maupun profesional. Ini termasuk beragam jenis gambar, seperti foto pribadi, gambar medis, ilustrasi ilmiah, hingga gambar untuk keperluan komersial. Dalam situasi ini, pentingnya sistem pencarian gambar (image retrieval system) tidak dapat diabaikan. Sistem ini memudahkan pengguna dalam mencari, mengakses, dan mengelola gambar-gambar mereka. Fitur ini sangat berguna untuk berbagai kebutuhan, mulai dari pencarian gambar pribadi, analisis gambar medis, hingga pencarian produk komersial berdasarkan gambar. Contoh umum dari sistem ini adalah Google Lens.

Dalam Tugas Besar Algeo ini kami diminta untuk merealisasikan program CBIR yang berbasiskan warna dan tekstur kemudian menampilkan hasilnya pada website.

BAB 2 - Landasan Teori

a. Dasar teori

Content-Based Image Retrieval (CBIR) adalah teknik pengambilan gambar dari database berdasarkan konten visual gambar itu sendiri. CBIR menggunakan algoritma dan teknik untuk memungkinkan pencarian gambar yang berdasarkan pada fitur visual seperti warna, tekstur, bentuk, dan lainnya. Dua parameter utama yang sering digunakan dalam CBIR adalah warna dan tekstur, masing-masing dengan metodologi dan aplikasi khususnya.

CBIR dengan Parameter Warna

Konsep: Dalam CBIR berbasis warna, fitur utama yang digunakan untuk pencarian dan pengambilan gambar adalah warna. Sistem ini mengekstrak informasi warna dari gambar dan menggunakan informasi ini untuk mencari gambar yang serupa dalam database.

Metodologi:

Ekstraksi Fitur Warna: Menggunakan histogram warna atau metode lain untuk mengidentifikasi dan mengkuantifikasi distribusi warna dalam gambar.

Konversi Ruang Warna: Mengubah gambar dari satu ruang warna ke ruang lain (misalnya, dari RGB ke HSV) untuk analisis yang lebih efektif.

Pencocokan dan Pencarian: Menggunakan metrik kesamaan untuk membandingkan fitur warna gambar query dengan gambar dalam database.

Aplikasi: Cocok untuk aplikasi di mana warna adalah faktor penting, seperti dalam pencarian gambar seni, desain fashion, atau media digital.

RGB dinormalisasi menjadi range [0,1]

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

Mencari Cmax, Cmin, dan delta

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

Mencari Hue Saturation Value

$$H = \begin{Bmatrix} 0° & \Delta = 0 \\ 60° \times \left(\frac{G' - B'}{\Delta} mod 6\right) & , C' \max = R' \\ 60° \times \left(\frac{B' - R'}{\Delta} + 2\right) & , C' \max = G' \\ 60° \times \left(\frac{R' - G'}{\Delta} + 4\right) & , C' \max = B' \end{Bmatrix}$$

$$S = \begin{Bmatrix} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{Bmatrix}$$

$$V = Cmaks$$

Menentukan kemiripan

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

CBIR dengan Parameter Tekstur

Konsep: CBIR berbasis tekstur menggunakan karakteristik tekstur dari sebuah gambar untuk pencarian. Tekstur dapat mencakup pola, ketidakseragaman, kehalusan, atau aspek visual lain dari sebuah gambar.

Metodologi:

Ekstraksi Fitur Tekstur: Teknik seperti matriks co-occurrence, filter Gabor, dan analisis wavelet digunakan untuk menggambarkan tekstur gambar.

Deskripsi Tekstur: Mengidentifikasi fitur seperti kontras, homogenitas, energi, dan korelasi dari tekstur.

Pencocokan dan Pencarian: Menerapkan algoritma untuk membandingkan fitur tekstur gambar query dengan gambar yang ada dalam database.

Aplikasi: Berguna dalam domain di mana tekstur gambar adalah kunci, seperti pengenalan pola dalam citra medis, klasifikasi material dalam industri, dan pencarian gambar alam atau arsitektur.

Content-Based Image Retrieval (CBIR) dengan perbandingan tekstur melibatkan penggunaan matriks co-occurrence. Proses ini dimulai dengan konversi gambar ke grayscale, karena warna tidak penting dalam analisis tekstur. Kemudian, nilai grayscale diquantifikasi ke dalam matriks 256 × 256 piksel. Dalam CBIR, tekstur gambar dinilai berdasarkan kekasaran teksturnya.

Matriks co-occurrence dibuat menggunakan parameter offset (Δx,Δy), yang bergantung pada arah (θ) dan jarak. Nilai θ yang digunakan adalah 0°, 45°, 90°, dan 135°. Dari matriks co-occurrence, diperoleh symmetric matrix dengan menjumlahkannya dengan hasil transpose-nya, dan kemudian dilakukan normalisasi matriks.

Ekstraksi tekstur dari co-occurrence matrix menghasilkan tiga komponen: contrast, homogeneity, dan entropy. Setiap komponen ini dihitung menggunakan persamaan khusus, dengan P sebagai matriks co-occurrence. Dari ketiga komponen ini, dibuat vektor yang digunakan untuk mengukur tingkat kemiripan antara dua gambar.

Pengukuran kemiripan antara dua gambar dilakukan dengan menggunakan Teorema Cosine Similarity. Vektor yang mewakili gambar dibandingkan, dan semakin besar nilai Cosine Similarity, semakin tinggi tingkat kemiripan antara kedua gambar tersebut.

Constrast:

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

Homogenity:

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

Entropy

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

Cosine similarity

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

CBIR dengan parameter warna dan tekstur masing-masing memiliki kelebihan dan kegunaan tertentu, dan sering kali keduanya digabungkan untuk mencapai hasil pencarian yang lebih akurat dan efisien. Teknik ini menjadi semakin penting seiring dengan pertumbuhan besar dalam jumlah dan variasi gambar digital yang tersedia.

b. Penjelasan singkat mengenai pengembangan sebuah *website*

Website program CBIR kami menggunakan react sebagai framework, kemudian kami menggunakan flask sebagai penghubung kode python kami dengan website. Serta guna meningkatkan estetika kami menggunakan css dan tailwind sebagai tools untuk mempercantik tampilan website. Kemudian untuk menjalankan program kami mengubah dataset menjadi format histogram dan vektor yang kemudian akan digunakan untuk pencarian gambar.

Pranala dokumentasi Flask
https://flask.palletsprojects.com/en/3.0.x/

Pranala dokumentasi React JS

https://react.dev/learn

Pranala dokumentasi Tailwind

https://tailwindcss.com/docs/installation

BAB 3 - Analisis Pemecahan Masalah

a. Langkah-langkah pemecahan masalah

Program CBIR memerlukan pengubahan dari RGB(Red, Green, Blue) gambar menjadi HSV(Hue, Saturation, Value) agar lebih mudah diproses. Kemudian untuk meningkatkan efisiensi penggunaan dataset maka kami menyimpan cache dari nilai yang kita simpan untuk setiap image dari dataset sehingga dataset cukup diolah satu kali bila menggunakan dataset yang sama.

b. Proses pemetaan masalah menjadi elemen-elemen pada aljabar geometri

Dalam kuliah Aljabar Linier dan Geometri dijelaskan bahwa untuk mencari kemiripan suatu vektor kita dapat mencari nilai cosinus antara dua vektor tersebut, kemudian operasi cosinus similarity menggunakan teorema dot product di mana rumusnya sebagai berikut

$$a \cdot b = \sum_{i=1}^{n} a_i b_i$$

a = 1st vector

b = 2nd vector

n = dimension of the vector space

a_i = component of vector a

b_i = component of vector b

c. Contoh ilustrasi kasus dan penyelesaiannya

misal kita ingin mencari dot product antara dua vektor

A = (2,-3,1) dan B = (-1,4,2) maka

$\mathbf{A} \cdot \mathbf{B} = 2 \times (-1) + (-3) \times 4 + 1 \times 2$

misal kita ingin mencari cosine similarity

$$\text{cosine similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \times \|\mathbf{B}\|}$$

$\mathbf{A} = (1, 2, 3)$ dan $\mathbf{B} = (4, 5, 6)$

maka A.B = 0.975

BAB 4 - Implementasi dan Uji Coba

a. Implementasi program utama
(pseudocode program yang mendukung fungsionalitas utama)

Program color.py

```
FUNCTION rgb_to_hsv_vectorized(image)
    CONVERT image to float32 array and normalize values
    INITIALIZE r, g, b channels from image
    CALCULATE max and min for each pixel
    CALCULATE difference between max and min (df)
    INITIALIZE h, s with zeros and v with max values
    HANDLE division by zero cases for df
    CALCULATE hue (h) values WHERE df is not zero
      IF red is max THEN calculate h for red
      IF green is max THEN calculate h for green
      IF blue is max THEN calculate h for blue
    CALCULATE saturation (s) values
    ADJUST h values to range 0 to 180
    COMBINE h, s, v to form HSV image
    RETURN HSV image

FUNCTION calculate_histogram(image)
    CONVERT image to HSV using rgb_to_hsv_vectorized
    DEFINE histogram bins for h and s
    CALCULATE 3D histogram for HSV image
    FLATTEN histogram to 1D array
    NORMALIZE histogram values
    RETURN normalized histogram

FUNCTION normalize_vector(image)
    CALCULATE histogram of image
    COMPUTE norm of histogram
    IF norm is zero THEN return original histogram
```

ELSE normalize histogram by norm and return

FUNCTION calculate_cosine_similarity(vector1, vector2)

CALCULATE dot product between vector1 and vector2

CLAMP similarity value between -1.0 and 1.0

RETURN similarity value

END FUNCTION

Program texture.py

FUNCTION processImg(image)

SPLIT image into r, g, b channels

ADJUST each channel with specific weights (0.29 for r, 0.587 for g, 0.114 for b)

COMBINE adjusted channels to form grayscale image

CONVERT grayscale image to integer type

RETURN grayscale image

FUNCTION coOccurrenceMat(image)

CONVERT image to grayscale using processImg function

INITIALIZE cooccurrence matrix with size 256x256

FOR EACH pixel in image

INCREMENT cooccurrence matrix based on pixel intensity and its neighbor

MAKE cooccurrence matrix symmetric by adding its transpose

NORMALIZE cooccurrence matrix by dividing with sum of all elements

RETURN normalized cooccurrence matrix

FUNCTION features(image)

COMPUTE co-occurrence matrix for image

CALCULATE contrast, entropy, and homogeneity from co-occurrence matrix

STORE calculated values in a vector

RETURN vector containing contrast, homogeneity, and entropy

```
FUNCTION cosineSimilarity(vector1, vector2)
    INITIALIZE variables for dot product, square sum of vector1, and square sum of vector2
    COMPUTE dot product and square sums
    CALCULATE square roots of square sums
    COMPUTE cosine similarity by dividing dot product by product of square roots
    RETURN cosine similarity

END FUNCTION
```

b. Penjelasan struktur program berdasarkan spesifikasi
(dapat membahas terkait arsitektur kode secara keseluruhan, struktur pembangunan website, atau hal-hal lain yang berkaitan dengan struktur program).

Struktur Program kami terdiri dari dua komponen utama, yakni SRC (color.py yakni program CBIR berbasiskan warna dan texture.py yakni program CBIR berbasiskan tekstur) dan Web

Kerangka folder Web berisi struktur khas struktur react JS. Kemudian terdapat folder dataset dan file-file csv untuk menyesuaikan dengan program kami. File csv tersebut memuat cache dari histrogram dan vektor dari image dataset, kemudian memuat data image mana saja yang mirip.

c. Penjelasan tata cara penggunaan program
(dapat membahas terkait interface program, fitur-fitur yang disediakan program, atau hal-hal lain yang berkaitan dengan cara penggunaan program).

Arahkan ke direktori Web, pip install flask, install react, npm install, npm start

Langkah pertama: upload dataset

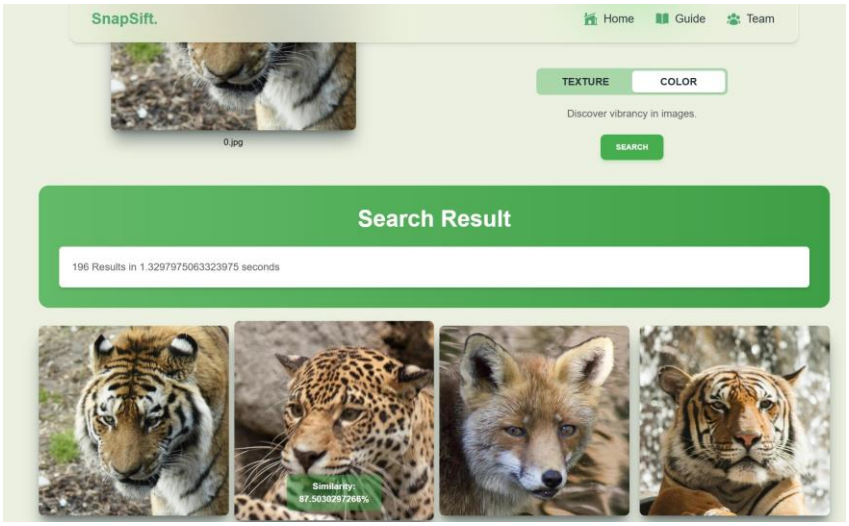
Langkah kedua: upload image query

Langkah ketiga: pilih metode untuk menentukan kemiripan (warna atau tekstur)

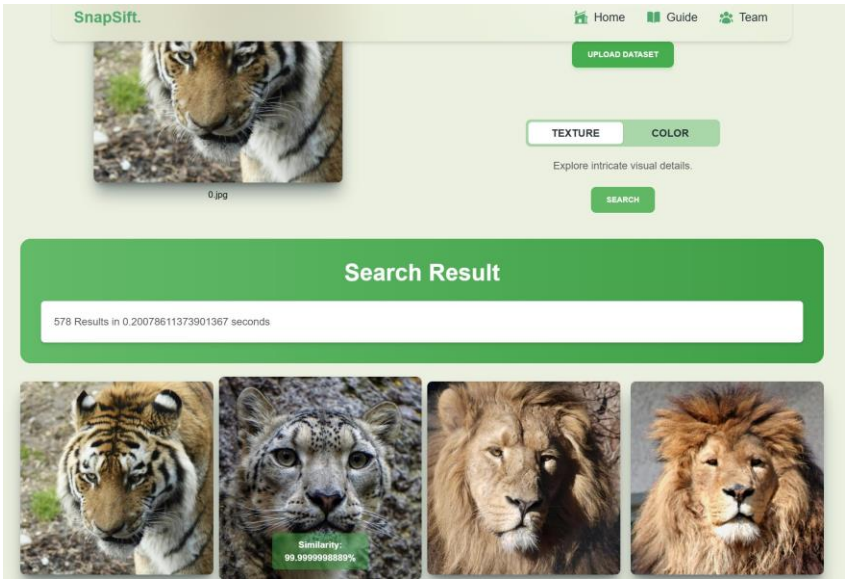
Langkah keempat: klik search button

Langkah kelima: voila! Muncul hasil image yang mirip disertai prosentase kemiripan dan runtime program.

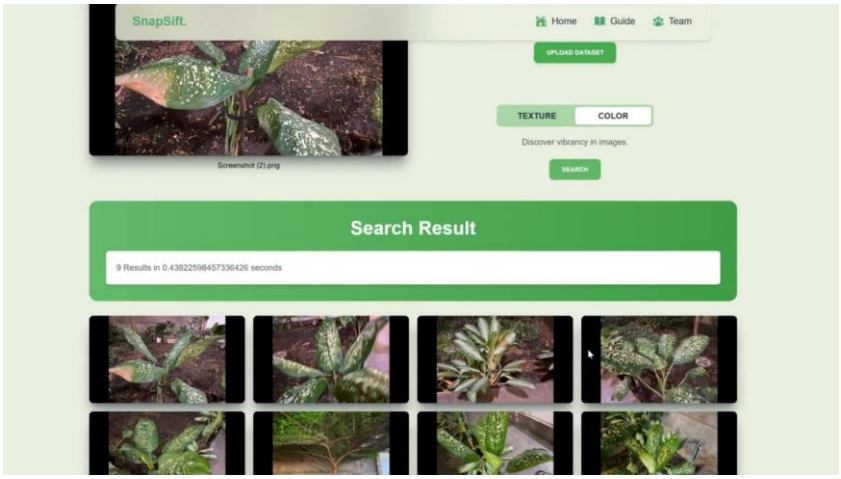
d. Hasil pengujian
(screenshot antarmuka program dan beberapa data uji beserta skenario pengujian). Diharapkan bahwa setiap kelompok menyampaikan hasil pengujian untuk beberapa test case yang berbeda, mulai dari variasi gambar masukan pencarian hingga dataset.



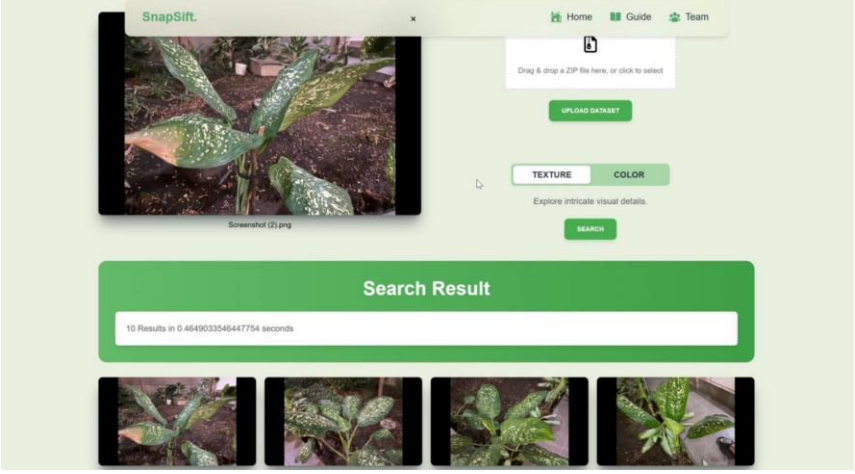
Gambar(1) di atas adalah pengujian hewan dengan CBIR menggunakan color



Gambar(2) di atas adalah pengujian hewan dengan CBIR menggunakan texture



Gambar(3) di atas adalah pengujian tanaman dengan CBIR menggunakan color



Gambar(4) di atas adalah pengujian tanaman dengan CBIR menggunakan texture

e. Analisis dari desain solusi algoritma pencarian

--Yang diimplementasikan pada setiap pengujian yang dilakukan. Misalnya, apakah pembangunan CBIR berdasarkan fitur warna lebih baik dari tekstur pada kasus-kasus tertentu beserta analisis mengenai mengapa hal itu bisa terjadi jika benar--

Menurut kami di banyak kasus, pembangunan CBIR berdasarkan fitur tekstur akan lebih baik daripada fitur warna, ini dikarenakan tekstur akan cenderung menunjukkan kepresisian yang lebih baik dibanding dengan warna, terlebih fitur warna sensitif terhadap pencahayaan. Akan tetapi tidak dapat dipungkiri fitur warna juga memiliki keunggulan tersendiri yakni mampu membedakan warna.

BAB 5 - Penutup

a. Kesimpulan

Pembangunan CBIR baik berdasarkan warna maupun tekstur keduanya memiliki kelebihan masing-masing. Menurut kami untuk tindak lanjut bila ingin menggarap CBIR maka akan sangat bagus kalau kedua fitur ini dikombinasikan , misalnya image output harus memenuhi threshold minimum 60% dari fitur warna dan tekstur

b. Saran

untuk menggunakan website CBIR kami, kami menyarankan agar:

a. Gambar dan dataset yang diinput memiliki pencahayaan yang baik.
b. Menggunakan kedua fitur (warna dan tekstur) untuk menentukan similarity agar kepresisian meningkat.
c. Memasukkan dataset dalam format zip agar lebih ringan untuk diolah

c. Komentar atau Tanggapan

Komentar atau tanggapan kami untuk tugas besar ini adalah kami menjadi paham bagaimana implementasi mata kuliah Aljabar Linear dan Geometri dan kami menjadi paham implementasi kode python serta penggunaan framework reactJs, tailwind, flask, dan juga css pada website.

d. Refleksi terhadap tugas besar ini.

Menurut kami tugas besar matakuliah Aljabar Linear dan Geometri untuk membuat algoritma CBIR berbasiskan warna dan tekstur yang diimplementasikan di website ini membutuhkan pemahaman vektor yang cukup baik dan juga pemahaman framework website.

e. Ruang Perbaikan atau Pengembangan

Menurut kami aspek Spesifikasi Tugas sudah dipenuhi, namun demikian bila program ini ingin di-deploy atau dikomersialisasi maka baiknya threshold fitur warna dan tekstur digabung agar kepresisian meningkat, serta website harus dihosting agar dapat diakses melalui internet.

BAB 6 – DAFTAR PUSTAKA

a. Pranala repository Github: https://github.com/thoriqsaputra/Algeo02-22123
b. Pranala Bonus Video: https://youtu.be/fsf0D43lMIQ?si=f-i9THXBL1zlax_o