

Tugas Besar 2 IF2123 Aljabar Linier dan Geometri
Aplikasi Aljabar Vektor dalam Sistem Temu Balik Gambar
Semester I Tahun 2023/2024

Revisi I : Minggu, 5 November 2023, Pukul 02.04 - Penambahan *ban-list*

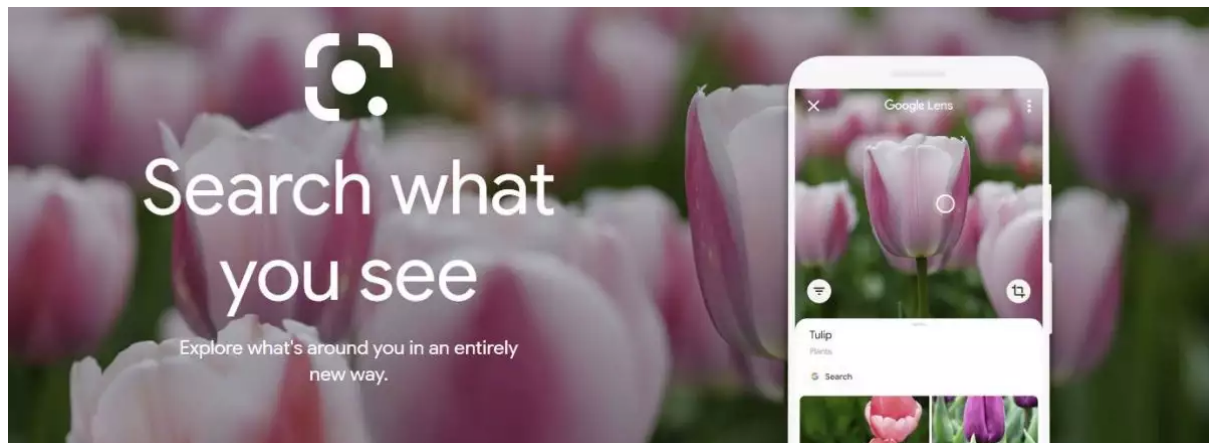
Revisi II : Senin, 6 November 2023, Pukul 21.15 - Penjelasan waktu eksekusi

Revisi III : Rabu, 8 November 2023, Pukul 22.02 - Kompresi pada CBIR tekstur dan nilai blok HSV

Revisi IV : Minggu, 19 November 2023, Pukul 19.33 - Lampiran pada Laporan

ABSTRAKSI

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. Salah satu contoh penerapan sistem temu balik gambar yang mungkin kalian tahu adalah Google Lens.



Gambar 1. Contoh penerapan *information retrieval system* (Google Lens)

Di dalam Tugas Besar 2 ini, Anda diminta untuk mengimplementasikan sistem temu balik gambar yang sudah dijelaskan sebelumnya dengan memanfaatkan Aljabar Vektor dalam bentuk sebuah *website*, dimana hal ini merupakan pendekatan yang penting dalam dunia pemrosesan data dan pencarian informasi. Dalam konteks ini, aljabar vektor digunakan untuk menggambarkan dan menganalisis data menggunakan 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.

CONTENT-BASED INFORMATION RETRIEVAL (CBIR)

Content-Based Image Retrieval (CBIR) adalah sebuah proses yang digunakan untuk mencari dan mengambil gambar berdasarkan kontennya. Proses ini dimulai dengan ekstraksi fitur-fitur penting dari gambar, seperti warna, tekstur, dan bentuk. Setelah fitur-fitur tersebut diekstraksi, mereka diwakili dalam bentuk vektor atau deskripsi numerik yang dapat dibandingkan dengan gambar lain. Kemudian, CBIR menggunakan algoritma pencocokan untuk membandingkan vektor-fitur dari gambar yang dicari dengan vektor-fitur gambar dalam dataset. Hasil dari pencocokan ini digunakan untuk mengurutkan gambar-gambar dalam dataset dan menampilkan gambar yang paling mirip dengan gambar yang dicari. Proses CBIR membantu pengguna dalam mengakses dan mengeksplorasi koleksi gambar dengan cara yang lebih efisien, karena tidak memerlukan pencarian berdasarkan teks atau kata kunci, melainkan berdasarkan kesamaan nilai citra visual antara gambar-gambar tersebut. Pada Tugas Besar kali ini, Anda diminta untuk mengimplementasikan 2 parameter CBIR yang paling populer, antara lain :

1. CBIR dengan parameter warna

Pada CBIR kali ini akan dibandingkan *input* dari sebuah *image* dengan *image* yang dimiliki oleh dataset, hal ini dilakukan dengan cara mengubah *image* yang berbentuk RGB menjadi sebuah metode histogram warna yang lebih umum.

Histogram warna adalah frekuensi dari berbagai warna yang ada pada ruang warna tertentu hal ini dilakukan untuk melakukan pendistribusian warna dari *image*. Histogram warna tidak bisa mendeteksi sebuah objek yang spesifik yang terdapat pada *image* dan tidak bisa mendeskripsikan posisi dari warna yang didistribusikan.

Pembentukan ruang warna perlu dilakukan dalam rangka pembagian nilai citra menjadi beberapa *range* yang lebih kecil. Hal itu dilakukan untuk membuat sebuah histogram warna yang setiap interval tiap *range* dianggap sebagai *bin*. Histogram warna dapat dihitung dengan menghitung piksel yang menyatakan nilai warna pada setiap interval. Fitur warna mencakup histogram warna global dan histogram warna blok.

Pada perhitungan histogram, warna global HSV lebih dipilih karena warna tersebut dapat digunakan pada kertas (*background* berwarna putih) yang lebih umum untuk digunakan. Maka dari itu, perlu dilakukan konversi warna dari RGB ke HSV dengan prosedur sebagai berikut.

1. Nilai dari RGB harus dinormalisasi dengan mengubah nilai *range* [0, 255] menjadi [0, 1]

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

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. Selanjutnya gunakan hasil perhitungan di atas untuk mendapatkan nilai HSV

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

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

$$V = C_{\max}$$

Setelah mendapatkan nilai HSV lakukanlah perbandingan antara *image* dari input dengan dataset dengan menggunakan *cosine similarity*

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

Dengan A dan B adalah vektor dan n adalah jumlah dimensi dari vektor. Tingkat kemiripan dihitung dari seberapa besar hasil dari *Cosine Similarity*.

Untuk melakukan pencarian histogram, blok *image* dibagi menjadi $n \times n$ blok. Setiap blok akan menjadi tidak terlalu signifikan jika blok-blok tersebut terlalu besar dan akan meningkatkan waktu dalam memprosesnya jika ukuran dari blok terlalu kecil. Pada pencarian blok ini agar lebih efektif disarankan menggunakan 4×4 blok. Nyatakan nilai representatif dari sebuah blok dengan melakukan kalkulasi **nilai rata-rata HSV dari blok terkait** (sedikit berbeda dengan yang disampaikan pada jurnal referensi untuk menyederhanakan perhitungan. Akan tetapi, jika sudah telanjur mengimplementasikan metode pada referensi [**rata-rata histogram HSV**], diperbolehkan).

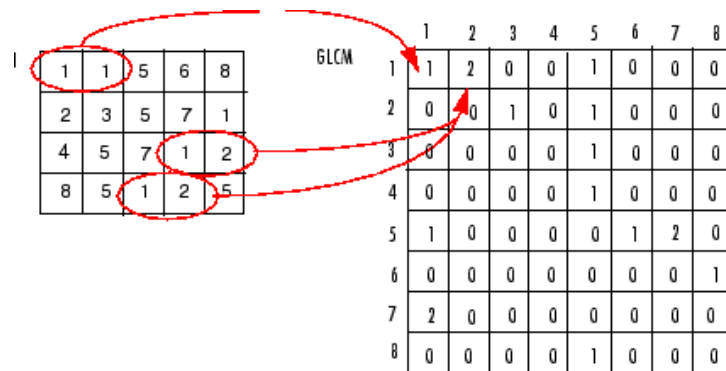
2. CBIR dengan parameter tekstur

CBIR dengan perbandingan tekstur dilakukan menggunakan suatu matriks yang dinamakan *co-occurrence matrix*. Matriks ini digunakan karena dapat melakukan pemrosesan yang lebih mudah dan cepat. Vektor yang dihasilkan juga mempunyai ukuran yang lebih kecil. Misalkan terdapat suatu gambar I dengan $n \times m$ piksel dan suatu parameter offset $(\Delta x, \Delta y)$, Maka dapat dirumuskan matriksnya sebagai berikut:

Dengan menggunakan nilai i dan j sebagai nilai intensitas dari gambar dan p serta q sebagai posisi dari gambar, maka *offset* Δx dan Δy bergantung pada arah θ dan jarak yang digunakan melalui persamaan di bawah ini.

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

Melalui persamaan tersebut, digunakan nilai θ adalah 0° , 45° , 90° , dan 135° . Sebagai gambaran, berikut diberikan contoh cara pembuatan *co-occurrence matrix* dan [link cara pembuatannya](#) (Contoh ini dapat digunakan sebagai referensi, bukan acuan sebagai acuan utama).



Gambar 2. Cara Pembuatan Matrix Occurrence

Setelah didapat *co-occurrence matrix*, buatlah *symmetric matrix* dengan menjumlahkan *co-occurrence matrix* dengan hasil transpose-nya. Lalu cari *matrix normalization* dengan persamaan.

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

Langkah-langkah dalam CBIR dengan parameter tekstur adalah sebagai berikut :

1. Konversi warna gambar menjadi *grayscale*, ini dilakukan karena warna tidaklah penting dalam penentuan tekstur. Oleh karena itu, warna RGB dapat diubah menjadi suatu warna *grayscale* Y dengan rumus:

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

2. Lakukan kuantifikasi dari nilai *grayscale*. Karena citra *grayscale* berukuran 256 piksel, maka matriks yang berkoresponden akan berukuran 256×256 . Berdasarkan penglihatan manusia, tingkat kemiripan dari gambar dinilai berdasarkan kekasaran tekstur dari gambar tersebut. ~~Grayscale semula dari suatu gambar akan dikompresi untuk mengurangi operasi perhitungan sebelum dibentuknya *co-occurrence matrix*~~ (Bagian ini dihapus karena kompresi berada diluar *scope* Tugas Besar ini, tetapi jika sudah ada yang mengimplementasikan kompresi **secara mandiri (menggunakan algoritma kompresi gambar)**, diperbolehkan. Anda juga diperbolehkan untuk melakukan kompresi menggunakan *library* kompresi yang sudah ada).

3. Dari *co-occurrence matrix* bisa diperoleh 3 komponen ekstraksi tekstur, yaitu *contrast*, *entropy* dan *homogeneity*. Persamaan yang dapat digunakan untuk mendapatkan nilai 3 komponen tersebut antara lain :

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 :

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

Keterangan : *P* merupakan matriks *co-occurrence*

Dari ketiga komponen tersebut, dibuatlah sebuah vektor yang akan digunakan dalam proses pengukuran tingkat kemiripan.

4. Ukurlah kemiripan dari kedua gambar dengan menggunakan Teorema *Cosine Similarity*, yaitu:

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

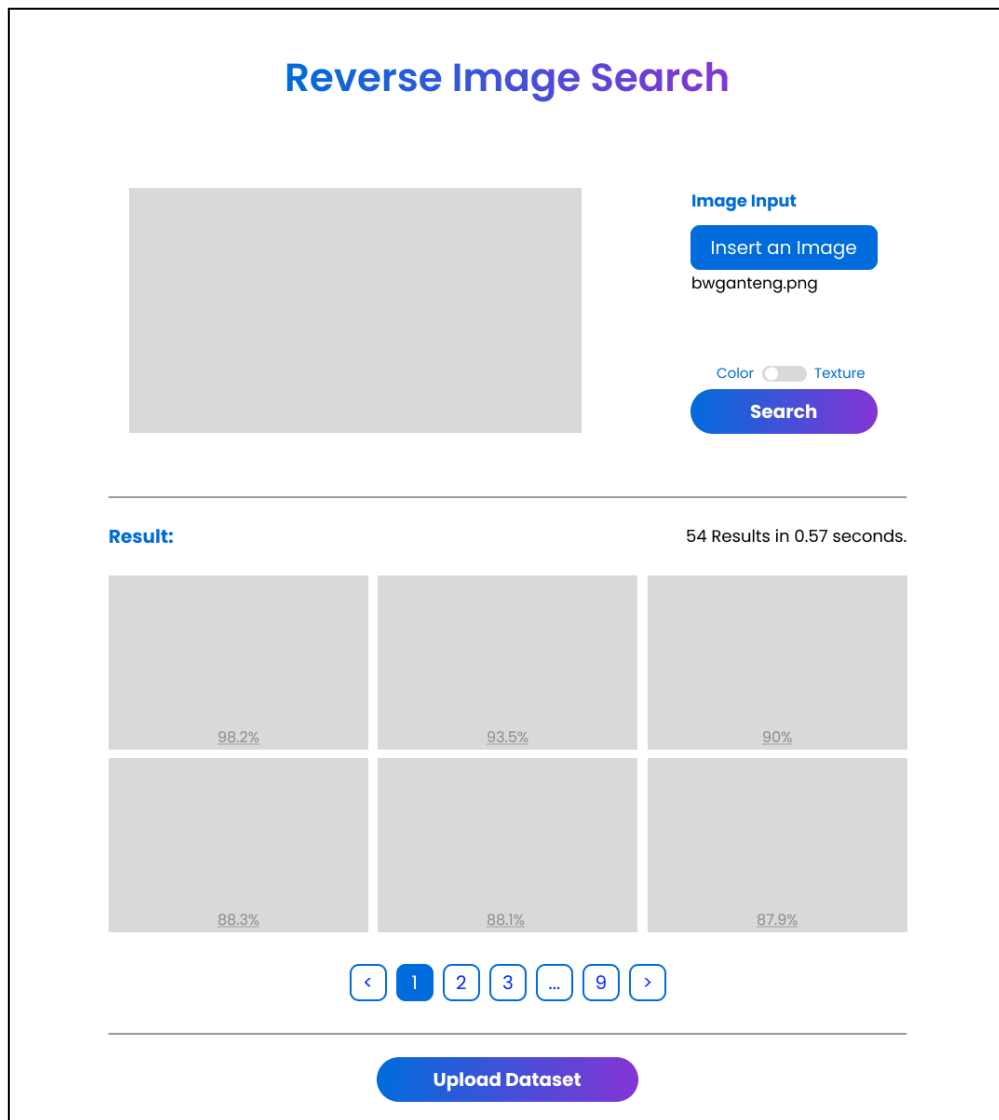
Disini *A* dan *B* adalah dua vektor dari dua gambar. Semakin besar hasil *Cosine Similarity* kedua vektor maka tingkat kemiripannya semakin tinggi.

PENGUNAAN PROGRAM

Berikut ini adalah masukan yang akan diberikan oleh pengguna untuk eksekusi program.

1. *Image query*, berisi gambar yang akan digunakan untuk melakukan pencarian gambar.
2. Kumpulan gambar (dataset), dilakukan dengan cara mengunggah *multiple image* dalam bentuk *folder* ke dalam *web browser*. Setelah memasukkan *image query*, kumpulan gambar inilah yang akan diseleksi menjadi *result*.

Tampilan *layout* dari aplikasi *website* yang akan dibangun adalah sebagai berikut.



Gambar 3. Ilustrasi tampilan layout dari aplikasi *website*

Anda dapat menambahkan menu lainnya seperti gambar, logo, dan sebagainya. Tampilan *Front-End* dari *website* tidak harus sama persis dengan *layout* yang diberikan di atas, tetapi dibuat semenarik mungkin dan wajib mencakup komponen-komponen berikut:

- Judul Website
- Tombol Insert Gambar, beserta Display Gambar yang ingin dicari
- Toggle Button untuk memilih *searching* berdasarkan Warna atau Tekstur
- Tombol Search untuk memulai pencarian

- Kumpulan Gambar (*result*) yang didapat dari hasil pencarian
- Informasi mengenai Jumlah *Result* yang didapat dan Waktu Eksekusi
- Tingkat Kemiripan Setiap Gambar (*result*) dengan gambar yang ingin dicari, dalam persentase (%)
- Tombol *Upload Dataset* untuk mengunggah kumpulan gambar (*dataset*) dalam suatu folder
- Page-page tambahan: Konsep singkat search engine yang dibuat, *How to Use*, dan *About us*

Secara umum, berikut adalah cara umum penggunaan program:

1. Pengguna terlebih dahulu memasukkan *dataset* gambar dalam bentuk *folder* yang berisi kumpulan gambar. *Dataset* gambar ini diperlukan sebelum proses *searching* agar ada perbandingan untuk gambar yang ingin dicari.
2. Setelah *dataset* sudah dimasukkan, pengguna memasukkan sebuah gambar yang ingin di-*search* dari *dataset*.
3. Pilih opsi pencarian, ingin melakukan pencarian berdasarkan warna atau tekstur.
4. Tekan tombol *search*, program kemudian akan memproses, mencari gambar-gambar dari *dataset* yang memiliki kemiripan dengan gambar yang dimasukkan tadi.
5. Program akan menampilkan kumpulan gambar yang mirip, diurutkan dari yang memiliki kemiripan paling tinggi ke yang paling rendah. Setiap gambar yang muncul diberi persentase kemiripannya.
6. Terdapat informasi terkait jumlah gambar yang muncul, dan waktu eksekusi programnya.

SPESIFIKASI TUGAS

Buatlah program sistem temu balik gambar (*image retrieval system*) dengan spesifikasi sebagai berikut:

1. Program menerima input *folder dataset* dan sebuah citra gambar.
2. *Dataset* gambar dapat diunduh secara mandiri melalui pranala [berikut](#). Akan tetapi peserta diperbolehkan untuk menggunakan *dataset* lain yang telah dipersiapkan oleh kelompok masing-masing.
3. Program menampilkan gambar citra gambar yang dipilih oleh pengguna.
4. Program dapat memberikan kebebasan pada pengguna untuk memilih parameter pencarian yang hendak digunakan (warna atau tekstur) melalui *toggle*. *Default* parameter yang digunakan di awal dibebaskan kepada masing-masing kelompok.
5. Program mulai melakukan perhitungan nilai kecocokan antara *image* masukan dengan *dataset image* berdasarkan parameter yang telah dipilih (warna atau tekstur).
6. Program dapat menampilkan nilai kecocokan antara *image* masukan dengan setiap gambar dalam *dataset*.
7. Program menampilkan hasil luaran dengan melakukan *descending sorting* berdasarkan nilai kecocokan tiap gambar. Cukup tampilkan seluruh gambar yang **memiliki tingkat kemiripan > 60%** dengan gambar masukan.
8. Program mengimplementasikan *pagination* agar jumlah gambar dapat dibatasi dengan halaman-halaman tertentu. Jumlah gambar dalam *dataset* yang akan digunakan oleh asisten saat penilaian mungkin berbeda dengan jumlah gambar pada *dataset* yang kalian gunakan, sehingga pastikan bahwa *pagination* dapat berjalan dengan baik.
9. Program dapat menampilkan **jumlah gambar** yang memenuhi kondisi tingkat kemiripan serta **waktu eksekusi**.

BONUS

Bagian ini hanya boleh dikerjakan apabila spesifikasi wajib dari Tugas Besar telah berhasil dipenuhi. Anda **tidak diharuskan untuk mengerjakan keseluruhan bonus**, tetapi semakin banyak bonus yang dikerjakan, maka akan semakin banyak tambahan nilai yang diperoleh.

Bagian A (Kamera)

1. Terdapat fitur kamera yang dapat melakukan penangkapan gambar secara *real-time* menggunakan *webcam* ketika program dijalankan.
2. Dilarang menambahkan *button* untuk *trigger* pada fitur kamera. **HINT:** Gunakan interval waktu untuk melakukan penangkapan gambar.
3. Fitur kamera merupakan fitur **tambahan**, sehingga fitur utama *upload* gambar melalui *website* tetap harus ada.

Bagian B (Image Scraping)

1. Implementasikan fitur *scraping* untuk melakukan ekstraksi gambar dari sebuah *website* tertentu.
2. Hasil dari *scraping* akan digunakan oleh program sebagai input dataset gambar.
3. Fitur *scraping* merupakan fitur **tambahan**, sehingga fitur utama *upload* gambar melalui *website* tetap harus ada.

Bagian C (Video)

1. Video penjelasan algoritma dan aplikasi program yang diunggah ke *youtube*. Referensi dapat kalian peroleh dengan kata kunci : Tubes 2 Algeo, Tugas Besar 2 Aljabar Linear dan Geometri, dan lain-lain. Berikut ini beberapa contoh referensi video yang disarankan:
 - [Face Recognition using EigenFace Method - YouTube](#)
 - [Video ALGEO02 - 21109 - Eigenface - YouTube](#)
 - [Tubes 2 IF2123 Algeo: Face Recognition with Eigenface \[YUKBISAYUK\] - YouTube](#)
 - [Tubes Algeo Face Recognition - YouTube](#)
2. Video dibuat sekreatif mungkin dengan target untuk mensosialisasikan ilmu yang kalian sudah pelajari dan terapkan pada program ini. Bukan hanya video mengenai penggunaan aplikasi.
3. **Dilarang membuat video yang secara keseluruhan hanya dilakukan melalui online video conference, seperti: Google Meet, Zoom, dll.** Penggunaan *video conference* masih diperbolehkan pada saat **melakukan demonstrasi penggunaan program**. Hal ini dilakukan untuk mendorong interaksi antar masing-masing anggota kelompok. ~~"Kan sudah offline hehe"~~ - Michael Leon.
4. Tautan video yang dikumpulkan tidak boleh menggunakan *url shortener* seperti bit.ly, tinyurl.com, dll.




Bagian D (Kreatifitas)

Segala bentuk kreatifitas fungsionalitas yang kalian buat akan mendapat apresiasi dari asisten. Beberapa fungsionalitas yang **disarankan** antara lain:

1. *Caching result* dari perhitungan sebuah dataset menggunakan *output file* tertentu, misalnya: csv, fvecs, dll. Hal ini akan meningkatkan efisiensi dan efektivitas ketika suatu dataset hendak digunakan berulang kali.
2. *Export* hasil luaran program dalam format PDF yang dapat diunduh ke *local storage* oleh pengguna. Hasil luaran dapat mencakup gambar masukan serta gambar dalam dataset dan nilai kecocokannya.

3. Dalam rangka menciptakan interpretasi parameter “tekstur” yang semakin mendekati riil, tambahkan komponen pembangun *co-occurrence matrix* selain yang disebutkan pada spesifikasi (*contrast*, *entropy* dan *homogeneity*). Berikan justifikasi alasan pemilihan komponen tambahan tersebut disertai penjelasan dan persamaan yang digunakan untuk mengimplementasikannya.
4. Gunakan *object detector* yang dapat melakukan proses pemotongan pada gambar masukan agar hasil yang diberikan dapat lebih maksimal. Sebagai contoh [haarcascade_frontalface_alt2.xml](#) yang hanya dapat digunakan sebagai *face detector*. Anda diperbolehkan untuk melakukan eksplorasi terhadap *object detector* yang lain.

PROSEDUR Pengerjaan

1. Tugas dikerjakan secara berkelompok yang terdiri dari 3 orang. Kelompok dipilih secara mandiri dan anggota kelompok diperbolehkan lintas kelas maupun lintas kampus. **Anggota kelompok tidak boleh ada yang sama dengan kelompok tugas besar sebelumnya.**
2. Daftarkan kelompok Anda via  Pendataan Tubes 2 Algeo 23/24 sebelum hari **Jumat, 3 November 2023 pukul 21.23 WIB**. Peserta yang tidak terdaftar setelah batas waktu yang ditentukan akan dikelompokkan secara acak oleh tim asisten.
3. Pemilihan asisten dilakukan melalui  Pendataan Tubes 2 Algeo 23/24 sebelum hari **Jumat, 3 November 2023 pukul 21.23 WIB**. Demo akhir dan penilaian tugas besar akan dilakukan dengan asisten yang dipilih. Jika terdapat kesulitan dalam pengerjaan tugas besar, Anda dapat dilakukan asistensi tugas besar kepada asisten yang telah dipilih dengan mekanisme dan batasan yang diumumkan kemudian.
4. Tugas ini dikumpulkan hari **Minggu, 19 November 2023** paling lambat **pukul 23.59 WIB**. Asisten akan mengumumkan jadwal demo program sesaat setelah pengumpulan selesai dilaksanakan.
5. **Dilarang keras menyalin program dari sumber lain** (buku, internet, program kakak tingkat, program kelompok lain).
6. Apabila ada pertanyaan yang ingin ditanyakan, diharapkan untuk bertanya melalui  Pendataan Tubes 2 Algeo 23/24 .

Catatan :

1. Diharapkan untuk **membaca aturan** yang telah disampaikan di *google sheets*.
2. Informasi tambahan mengenai tugas besar **akan disampaikan melalui bagian QnA** silahkan melakukan pemantauan secara rutin.
3. Asisten **berhak menolak melakukan asistensi** jika terdapat *conflict of interest* (Jadwal tidak cocok, asisten sedang sibuk/sakit, dan lain sebagainya).

LAPORAN

Berikut adalah sistematika Laporan Tugas Besar 2.

1. **Cover** : Cover laporan pada umumnya, namun terdapat foto anggota kelompok (foto bertiga sebagai anggota kelompok, bebas gaya) yang menggantikan logo “gajah” ganesha.
2. **Bab 1** : Deskripsi masalah (dapat meng-copy paste file tugas ini).
3. **Bab 2** : Landasan Teori
 - a. Dasar teori yang digunakan secara umum.
 - b. Penjelasan singkat mengenai pengembangan sebuah *website*.

4. **Bab 3 : Analisis Pemecahan Masalah**
 - a. Langkah-langkah pemecahan masalah.
 - b. Proses pemetaan masalah menjadi elemen-elemen pada aljabar geometri.
 - c. Contoh ilustrasi kasus dan penyelesaiannya.
5. **Bab 4 : Implementasi dan Uji Coba**
 - a. Implementasi program utama (*pseudocode* program yang mendukung fungsionalitas utama).
 - 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).
 - 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).
 - 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.
 - 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.
6. **Bab 5 :**
 - a. Kesimpulan
 - b. Saran
 - c. Komentar atau Tanggapan
 - d. Refleksi terhadap tugas besar ini.
 - e. Ruang Perbaikan atau Pengembangan
7. **Daftar Pustaka** serta **link menuju release terakhir dari repository** github dan video (jika mengerjakan bonus video). Tidak diperbolehkan menggunakan *url shortener* seperti bit.ly atau semacamnya.

PENGUMPULAN TUGAS

1. Program disimpan di dalam *repository github* dengan nama repository **Algeo02-22XXX** dengan XXX adalah tiga digit terakhir anggota dengan NIM terkecil. Di dalam *repository* tersebut terdapat empat folder: *src*, *test*, *doc*, dan *img* yang masing-masing berisi:
 - Folder *src* berisi *source code* program
 - Folder *test* berisi gambar masukan untuk pengujian
 - Folder *doc* berisi laporan dengan format PDF
 - Folder *img* berisi *screenshot* hasil ujicoba program yang ditampilkan juga *image* yang digunakan oleh READMESelain itu, pastikan **README** yang dibuat dapat mencakup seluruh informasi yang diperlukan untuk melakukan instalasi serta penggunaan *website*. Silahkan gunakan template [berikut](#), sebagai referensi untuk implementasi struktur dalam README.
2. Dalam melakukan *commit*, gunakan *template semantic commit* [berikut](#), agar pesan *commit* dapat lebih terstruktur. Pastikan juga *repository* bersifat **private** dan telah **mengundang asisten** yang pembagiannya akan diumumkan kemudian.
3. Format penamaan laporan adalah **Algeo02-22XXX.pdf** dengan XXX adalah tiga digit terakhir NIM anggota terkecil.

4. Buatlah **tag dan release** untuk tiap *commit* terakhir pengumpulan. Tag yang digunakan memiliki format v.1.X dengan X dimulai dari angka 0 dan penambahan 1 untuk tiap revisi yang dilakukan, misal: v.1.0 (untuk pengumpulan pertama) dan v.1.1 (untuk revisi pertama setelah pengumpulan). Pelajari prosedurnya pada referensi [berikut](#).
5. Laporan tugas besar dikumpulkan melalui pranala berikut [Kumpul Tugas Besar 2](#), dengan **link repository yang dilampirkan pada laporan**. Akan tetapi, pastikan bahwa pengumpulan *source code* via *form* dilakukan dengan menyertakan **link release terakhir BUKAN link repository**. (*Repository* dibuat **public** setelah **deadline** pengumpulan dengan batas maksimal **H + 1 deadline**).
6. Jika terdapat revisi yang menyebabkan pengumpulan harus dilakukan lebih dari sekali, silahkan kumpulkan kembali berkas dengan format penamaan **Algeo02-22XXX-RevisiY** dengan Y adalah nomor revisi.
7. Deadline pengumpulan adalah **Minggu, 19 November 2023 pukul 23.59 WIB**. Pengumpulan setelah waktu tersebut akan mendapatkan pengurangan nilai. Selain itu, pengurangan nilai juga dapat diberlakukan apabila terdapat ketidaksesuaian pengerjaan yang dilakukan dengan spesifikasi yang telah diberikan.

PENILAIAN

Komposisi penilaian umum adalah sebagai berikut :

1. Program : 80%
2. Laporan : 20%

Catatan tambahan, **waktu eksekusi** menjadi **salah satu aspek penilaian** yang akan dikonsiderasi. Oleh sebab itu, susunlah algoritma untuk memproses CBIR **semangkus dan sesangkil mungkin**. Sebagai estimasi, waktu normal untuk melakukan pemrosesan **per-gambar** berada pada rentang **kurang dari 1 detik hingga 1 detik** untuk menghasilkan performa yang maksimal.

BAN-LIST

Karena seringkali muncul pertanyaan di QnA yang memastikan apakah *library* boleh digunakan atau tidak, maka berikut disampaikan **batasan library** yang boleh dan tidak boleh digunakan selama pengerjaan Tugas Besar.

1. *Library* yang **berhubungan dengan pemrosesan gambar** (seperti OpenCV2 dan PIL pada python atau image pada Golang) **boleh** untuk digunakan.
2. *Library* yang diperlukan untuk **melakukan mekanisme pre-processing**, baik untuk dataset maupun gambar masukan (seperti *resizing* dan *interpolate*) untuk adaptasi dimensi gambar **boleh** untuk digunakan. Perhatikan bahwa tahap ini **dapat membawa beberapa konsekuensi** pada hasil akhir pemrosesan CBIR, sehingga berikan justifikasi yang jelas **mengapa harus dilakukan** dan **apa dampak yang mungkin timbul** pada laporan.
3. *Library* untuk melakukan **proses kalkulasi matematis kompleks** seperti *math* untuk kalkulasi logaritma atau operasi sigma (bila diperlukan) **boleh** untuk digunakan.
4. *Library* yang **mendukung fungsionalitas dan karakteristik matriks** (seperti *numpy* pada python) untuk operasi matematis matriks, seperti penambahan, pengurangan, dan lain-lain **boleh** untuk digunakan. Akan tetapi, untuk **membangun modul utama dari CBIR**, seperti pembentukan matriks *co-occurrence* harus **dibangun secara mandiri**.

5. *Library* untuk melakukan **proses-proses utama pada CBIR**, seperti *library* untuk menentukan nilai *cosine-similarity*, konversi RGB ke HSV, serta untuk mengekstrak fitur warna dan tekstur **tidak boleh** untuk digunakan.
6. Bahasa pemrograman yang digunakan untuk membangun algoritma **dibebaskan**, tetapi kami dari asisten **menyarankan** untuk menggunakan Python, Javascript (Node.js), atau Golang. “*Mau pake prolog juga boleh, ini [sumbernya](#) kali aja ada yang tertarik*” - Michael Leon
7. Penggunaan *framework* untuk *back-end* dan *front-end website* **dibebaskan**. **Contoh *framework website*** yang bisa dipakai adalah Flask, Django, React, Vue, dan Svelte.
8. *Library* atau modul lain yang tidak secara eksplisit termasuk dalam kategori diatas pada dasarnya tidak terlalu dibutuhkan. Jika Anda merasa ragu, tanyakan pada QnA atau kepada asisten masing-masing.

REFERENSI

Perhatikan bahwa referensi hanya ada sebagai sumber belajar, bukan sebagai standar apa yang kami ekspektasikan dari kode kalian. Banyak sumber lain, jangan menutup diri ke hanya sumber sumber di bawah ini.

“RGB to HSV color conversion”

<https://math.stackexchange.com/questions/556341/rgb-to-hsv-color-conversion-algorithm>

Proses konversi warna RGB ke HSV.

“Content-based image retrieval using color and texture fused features”

<https://www.sciencedirect.com/science/article/pii/S0895717710005352>

Penjelasan lebih jauh mengenai CBIR tekstur dan warna.

“Feature Extraction : *Gray Level Co-occurrence Matrix* (GLCM)”

<https://yunusmuhammad007.medium.com/feature-extraction-gray-level-co-occurrence-matrix-glcm-10c45b6d46a1>

Cara membuat GLCM

“Create APIs with python and flask” by programming historian

<https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask>

Tautan ini untuk belajar teori tentang API.

“How to create a react flask project” by Miguel Grinberg

<https://blog.miguelgrinberg.com/post/how-to-create-a-react--flask-project>

Contoh project kecil yang dibangun dengan react dan flask.

Selamat Mengerjakan!

“Udah baca spek trus bingung mau ngapain dulu? Sama sih, mending baca ulang”

- Leon -

“Kaget ga tiba2 bikin website? Iya sama wkoawkoakwo”

- Bewe -

“Capek, pusing, mual? Sama. Panadol mana ya”

- Alex -

“Siapkan pr*mag untuk mengerjakan berbagai tubes”

- Fahrian -

“Kiw-kiw cukuruk kugeru, raga ngantuk kurang turu”

- Rifqi -

“Sudah mulai keos dengan tubes? Algeo: `Sstt tenang, gue tambahin 1 yak`”

- Sulthan -