Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri
Aplikasi Nilai Eigen dan Vektor Eigen dalam Kompresi Gambar

Semester I Tahun 2021/2022



DISUSUN OLEH

Safiq Faray	**13519145**
Daniel Salim	**13520008**
Kevin Roni	**13520114**

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
PROGRAM STUDI TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
BANDUNG

2021

BAB I
DESKRIPSI MASALAH

Abstraksi

Gambar adalah suatu hal yang sangat dibutuhkan pada dunia modern ini. Kita seringkali berinteraksi dengan gambar baik untuk mendapatkan informasi maupun sebagai hiburan. Gambar digital banyak sekali dipertukarkan di dunia digital melalui file-file yang mengandung gambar tersebut. Seringkali dalam transmisi dan penyimpanan gambar ditemukan masalah karena ukuran file gambar digital yang cenderung besar.

Kompresi gambar merupakan suatu tipe kompresi data yang dilakukan pada gambar digital. Dengan kompresi gambar, suatu file gambar digital dapat dikurangi ukuran filenya dengan baik tanpa mempengaruhi kualitas gambar secara signifikan. Terdapat berbagai metode dan algoritma yang digunakan untuk kompresi gambar pada zaman modern ini.

Three levels of JPG compression. The left-most image is the original. The middle image offers a medium compression, which may not be immediately obvious to the naked eye without closer inspection. The right-most image is maximally compressed.

Gambar 1. Contoh kompresi gambar dengan berbagai tingkatan
Sumber : Understanding Compression in Digital Photography (lifewire.com)

Salah satu algoritma yang dapat digunakan untuk kompresi gambar adalah algoritma SVD (Singular Value Decomposition). Algoritma SVD didasarkan pada teorema dalam aljabar linier yang menyatakan bahwa sebuah matriks dua dimensi dapat dipecah menjadi hasil perkalian dari 3 sub-matriks yaitu matriks ortogonal U, matriks diagonal S, dan transpose dari matriks ortogonal V. Dekomposisi matriks ini dapat dinyatakan sesuai persamaan berikut.

$$A_{m \times n} = U_{m \times m} \; S_{m \times n} \; V^{T}_{nxn}$$

Gambar 1. Algoritma SVD

Matriks U adalah matriks yang kolomnya terdiri dari vektor eigen ortonormal dari matriks AA^T. Matriks ini menyimpan informasi yang penting terkait baris-baris matriks awal,

dengan informasi terpenting disimpan di dalam kolom pertama. Matriks S adalah matriks diagonal yang berisi akar dari nilai eigen matriks U atau V yang terurut menurun. Matriks V adalah matriks yang kolomnya terdiri dari vektor eigen ortonormal dari matriks A^TA. Matriks ini menyimpan informasi yang penting terkait kolom-kolom matriks awal, dengan informasi terpenting disimpan dalam baris pertama.

Gambar 2. Ilustrasi Algoritma SVD dengan rank k

Dapat dilihat di gambar di atas bahwa dapat direkonstruksi gambar dengan banyak *singular values* **k** dengan mengambil kolom dan baris sebanyak k dari U dan V serta *singular value* sebanyak **k** dari S atau Σ terurut dari yang terbesar. Kita dapat mengaproksimasi suatu gambar yang mirip dengan gambar aslinya dengan mengambil **k** yang jauh lebih kecil dari jumlah total *singular value* karena kebanyakan informasi disimpan di *singular values* awal karena singular values terurut mengecil. Nilai **k** juga berkaitan dengan rank matriks karena banyaknya *singular value* yang diambil dalam matriks S adalah *rank* dari matriks hasil, jadi dalam kata lain **k** juga merupakan rank dari matriks hasil. Maka itu matriks hasil rekonstruksi dari SVD akan berupa informasi dari gambar yang terkompresi dengan ukuran yang lebih kecil dibanding gambar awal.

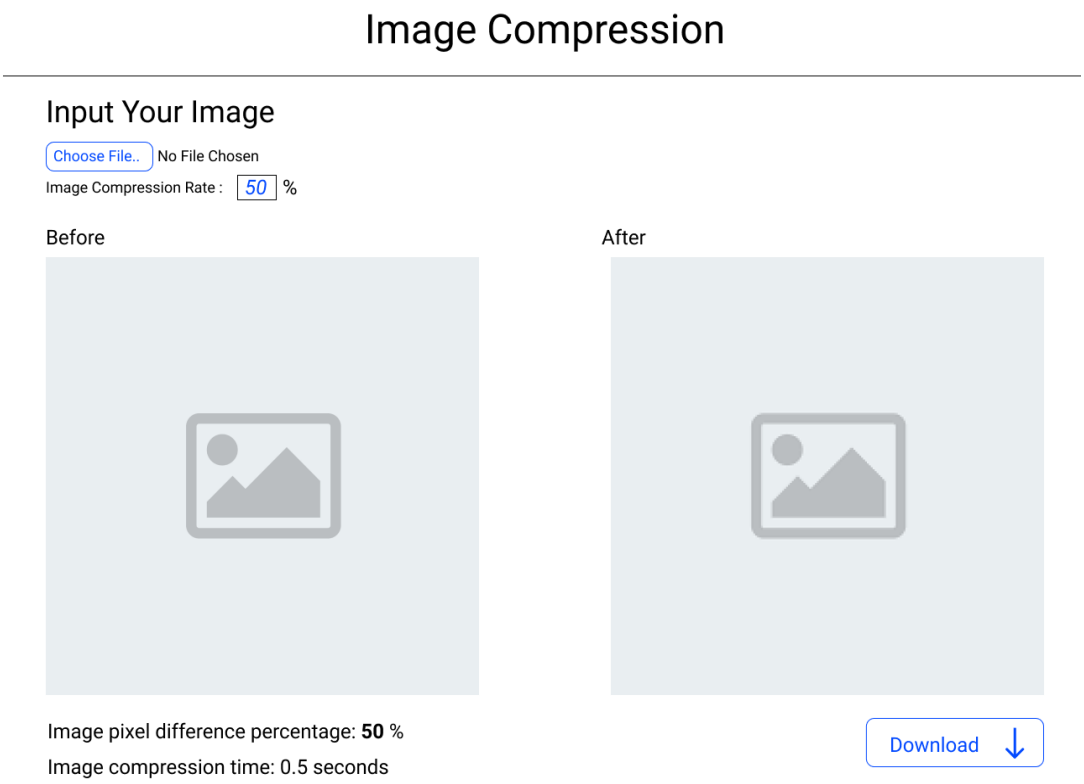
Pada kesempatan kali ini, kalian mendapatkan tantangan untuk membuat website kompresi gambar sederhana dengan menggunakan algoritma SVD.

Penggunaan Program

Berikut ini adalah input yang akan dimasukkan pengguna untuk eksekusi program:

1. **File gambar**, berisi *file* gambar input yang ingin dikompresi dengan format *file* yang bebas selama merupakan format untuk gambar.
2. **Tingkat kompresi**, berisi tingkat kompresi dari gambar (formatnya dibebaskan, cth: Jumlah *singular value* yang digunakan)

Tampilan *layout* dari aplikasi web yang akan dibangun kurang lebih adalah sebagai berikut. Anda dapat mengubah *layout* selama *layout* masih terdiri dari komponen yang sama.



Gambar 3. Contoh tampilan layout dari aplikasi web yang dibangun.

Catatan: Warna biru menunjukkan komponen yang dapat di klik.

Anda dapat menambahkan menu lainnya, gambar, logo, dan sebagainya. Tampilan *front end* dari *website* dibuat semenarik mungkin selama mencakup seluruh informasi pada layout yang diberikan di atas. Tampilan program merupakan bagian dari penilaian.

Saran Pengerjaan

Anda disarankan untuk membuat program *testing* pada **backend** terlebih dahulu untuk menguji keberhasilan dari proses kompresi dan rekonstruksi matriks gambar sebelum mengerjakan tampilan website.

Spesifikasi Tugas

Buatlah program kompresi gambar dengan memanfaatkan algoritma SVD dalam bentuk website lokal sederhana. Spesifikasi website adalah sebagai berikut:

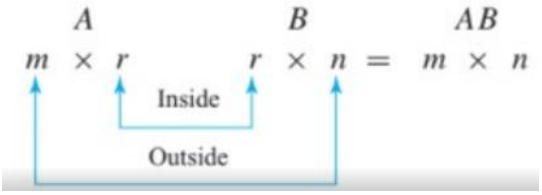
1. Website mampu menerima *file* gambar beserta *input* tingkat kompresi gambar (dibebaskan formatnya).
2. Website mampu menampilkan gambar *input*, *output*, *runtime* algoritma, dan persentase hasil kompresi gambar (perubahan jumlah pixel gambar).
3. File *output* hasil kompresi dapat diunduh melalui website.

4. Kompresi gambar tetap mempertahankan warna dari gambar asli.
5. **(Bonus)** Kompresi gambar tetap mempertahankan transparansi dari gambar asli, misal untuk gambar png dengan *background* transparan.
6. Bahasa pemrograman yang boleh digunakan adalah Python, Javascript, dan Go.
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. Kalian dapat menambahkan fitur fungsional lain yang menunjang program yang anda buat (unsur kreativitas diperbolehkan/dianjurkan).
9. Program harus modular dan mengandung komentar yang jelas.
10. Diperbolehkan menggunakan *library* pengolahan citra seperti OpenCV2, PIL, atau image dari Go.
11. **Dilarang** menggunakan *library* perhitungan SVD dan *library* pengolahan eigen yang sudah jadi.

BAB II
TEORI SINGKAT

2.1 Perkalian Matriks

Perkalian matriks memiliki syarat bahwa jumlah kolom matriks pertama A sama dengan jumlah baris matriks kedua B untuk membentuk produk AB. Jika kondisi ini tidak terpenuhi, produk tidak terdefinisi. Cara mudah untuk menentukan apakah produk dari dua matriks terdefinisi adalah dengan menuliskan ukuran matriks pertama dan di sebelah kanannya, tuliskan ukuran matriks kedua. Jika angka-angka di dalamnya sama, maka hasil kali terdefinisi dan angka-angka di luar memberikan ukuran produk.

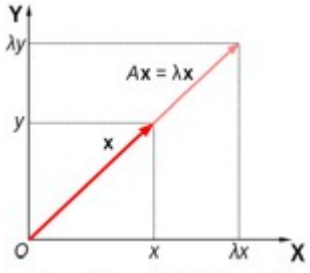


2.2 Nilai Eigen

Nilai Eigen adalah nilai karakteristik dari matriks yang berukuran n x n. Karena Eigen berasal dari bahasa jerman yang berarti ”asli” atau ”karakteristik”. Penggunaan nilai Eigen ini akan sangat mempengaruhi pada penentuan Vektor Eigen. Adapun, persamaan nilai Eigen yang dapat dipakai adalah seperti di bawah ini.

$$Ax = \lambda x$$

A menyatakan vektor yang ditentukan, x menyatakan vektor eigen, dan λ menyatakan nilai eigen. Nilai – nilai eigen ini bisa bernilai 1 angka atau 2 angka tergantung dengan vektor A. Operasi ini menyebabkan vektor Eigen menyusut ataupun memanjang bergantung dengan nilai eigen yang digunakan seperti pada ilustrasi di bawah.



2.3 Vektor Eigen

Vektor Eigen menyatakan sebuah vektor kolom pada formula yang sudah diberikan pada poin 2.2, yang apabila vektor ini dikalikan dengan vektorn n x n akan menghasilkan vektor yang merupakan kelipatan dari vektor itu sendiri. Sehingga setelah mendapatkan nilai-nilai Eigen, dapat dicari Vektor Eigen melalui formula di bawah ini:

$$\lambda I\text{-}Ax=0$$

Kemudian, untuk mencari terlebih dahulu nilai karakteristik nya dapat digunakan formula:

$$\det(\lambda I\text{-}Ax)=0$$

2.4 *Singular Value Decomposition* (SVD)

Matriks SVD adalah teorema dalam aljabar linier yang menyatakan bahwa sebuah matriks dua dimensi dapat dipecah menjadi hasil perkalian dari 3 sub-matriks yaitu matriks ortogonal U, matriks diagonal S, dan transpose dari matriks ortogonal V. Dekomposisi matriks ini dapat dinyatakan sesuai persamaan berikut.

$$A_{m\times n} = U_{m\times m}\ S_{m\times n}\ V^{T}_{nxn}$$

Matriks U adalah matriks yang kolomnya terdiri dari vektor eigen ortonormal dari matriks (AA)^T. Matriks ini menyimpan informasi yang penting terkait baris-baris matriks awal, denganinformasi terpenting disimpan di dalam kolom pertama. Matriks $ adalah matriks diagonal yang berisi akar dari nilai eigen matriks U atau V yang terurut menurun. Matriks V adalah matriks yang kolomnya terdiri dari vektor eigen ortonormal dari matriks A^TA. Matriks ini menyimpan informasi yang penting terkait kolom-kolom matriks awal, dengan informasi terpenting disimpan dalam baris pertama.

BAB III
IMPLEMENTASI PROGRAM

3.1 Kompresi Gambar

Fungsi dan Prosedur tercantum hanya fungsi utama saja

1. Function qrmethod(m: Matriks) → eigenvalue, eigenvector
   Deskripsi : mencari nilai eigen dan vector eigen dengan algoritma QR. Idenya adalah melakukan iterasi terhadap dekomposisi QR, lalu menulis matriks sebagai dot product dari matriks orthogonal dan segitiga atas, lalu mengalikannya faktornya. Dalam beberapa jurnal ditemukan algoritma QR akan *converge* ketika melewati iterasi ke-5.

2. Function svdmethod(m: Matriks) → U, sigma, Vt
   Deskripsi : Mencari nilai matriks U, sigma, dan Vt. Idenya adalah mencari eigenvalue dan eigenvector dengan metode qr yang dilakukan pada vektor singular kanan. Sigma dicari dengan mencari nilai singular. Lalu untuk mencari U, dihitung berdasarkan matriks V, hal ini disebabkan oleh nilai-nilai eigen yang tidak unik sehingga perhitungan tidak boleh terpisah.

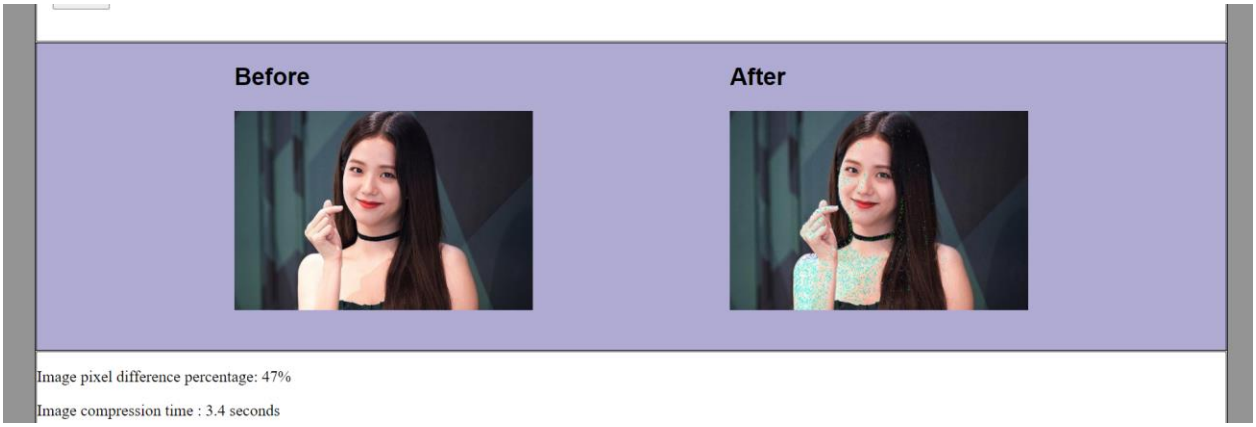
3. Function percentage_convert(img_mat: Matriks, k: integer) → Matriks Rekonstruksi
   Deskripsi : Mengubah pixel pada gambar sehingga terdekomposisi menjadi U, sigma, Vt menggunakan fungsi svdmethod lalu matriks gambar akan direkonstruksi berupa hasil dari dot product U, sigma, dan Vt.

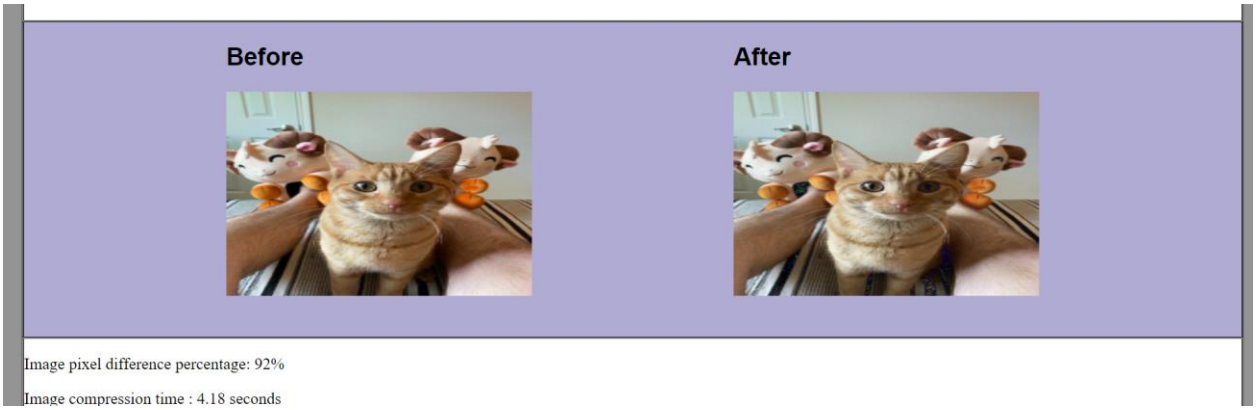
4. Procedure/function img_compress(Input filename: string, input ratio: integer, output compressed: image, output runtime: time)
   Deskripsi : Gambar diinput sebagai array dengan elemen float. Lalu setiap channel warna dipisahkan (RGB). Kemudian matriks dikompres dengan nilai k pada masing-masing warna. Setelah itu matriks yang sudah dikompres disatukan Kembali.

3.2 Website

Kami menggunakan flask sebagai bantuan untuk menghubungkan frontend dan backend dari website ini. Inputan pengguna kurang lebih sama dengan spesifikasi minimum yang diberikan pada spek tugas, seperti upload file dan persentase konversi yang diiginkan pengguna. Lalu tampilan outputnya menyediakan gambar before-after, runtime, pixel difference, serta pilihan untuk mendownload gambar.

BAB IV
EKSPERIMEN





BAB V
KESIMPULAN

5.1 Kesimpulan

Algortima yang kami buat sudah dapat mengompresi gambar dan membuat sebuah website sederhana untuk mengeksekusi program yang telah kami buat. Setelah serangkaian test, algoritma yang kami buat telah berhasil mengurangi ukuran file menjadi lebih kecil.

5.2 Saran

Jika bisa masa pengerjaannya diperlama atau dihindarkan dengan tugas-tugas besar lainnya, hal ini disebabkan materi yang kami pelajari di kelas hanya sedikiti yang bisa digunakan pada pengerjaan tugas besar ini karena tidak cepat sehingga butuh waktu eksplorasi terhadap algoritma yang ada.

5.3 Refleksi

Kesulitan paling sulit adalah ketika melakukan eksplorasi terhadap algoritma yang digunakan untuk mengaproksimasi eigenvalue dan eigenvector. Selain itu, untuk mendekomposisi matriks menggunakan metode svd juga sempat mengalami kesulitan karena harus mencari algoritma yang cepat. Kesulitan lainnya adalah ketika mencoba untuk membuat frontend dari website karena minimnya pengalaman yang kami miliki.

DAFTAR REFERENSI

https://www.quantstart.com/articles/QR-Decomposition-with-Python-and-NumPy/
https://www.youtube.com/watch?v=H7qMMudo3e8
https://www.lyndonduong.com/orthogonal-iteration/
https://math.stackexchange.com/questions/1762613/connection-between-power-iterations-and-qr-algorithm
https://www.youtube.com/watch?v=rYz83XPxiZo&ab_channel=MITOpenCourseWare
https://people.inf.ethz.ch/arbenz/ewp/Lnotes/chapter4.pdf