

LAPORAN TUGAS BESAR 2
IF2123 ALJABAR LINEAR GEOMETRI

Kelompok HVTHFYGNSVD

HVTHFYGNSVD'S GROUP

Hana



Hansel



Yohana



Disusun oleh:

Hansel Valentino Tanoto 13520046

Hana Fathiyah 13520047

Yohana Golkaria Nainggolan 13520053

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2021

DAFTAR ISI

DAFTAR ISI.....	2
BAB 1 DESKRIPSI MASALAH	3
BAB 2 TEORI SINGKAT	6
BAB 3 IMPLEMENTASI PROGRAM.....	8
BAB 4 EKSPERIMEN	13
BAB 5 KESIMPULAN, SARAN, DAN REFLEKSI.....	18
DAFTAR REFERENSI.....	19

BAB 1 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 \(lifewire.com\)](http://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.

$$A_{m \times n} = U_{m \times n} S_{m \times n} V_{m \times n}^T$$

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 $A^T A$. 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.

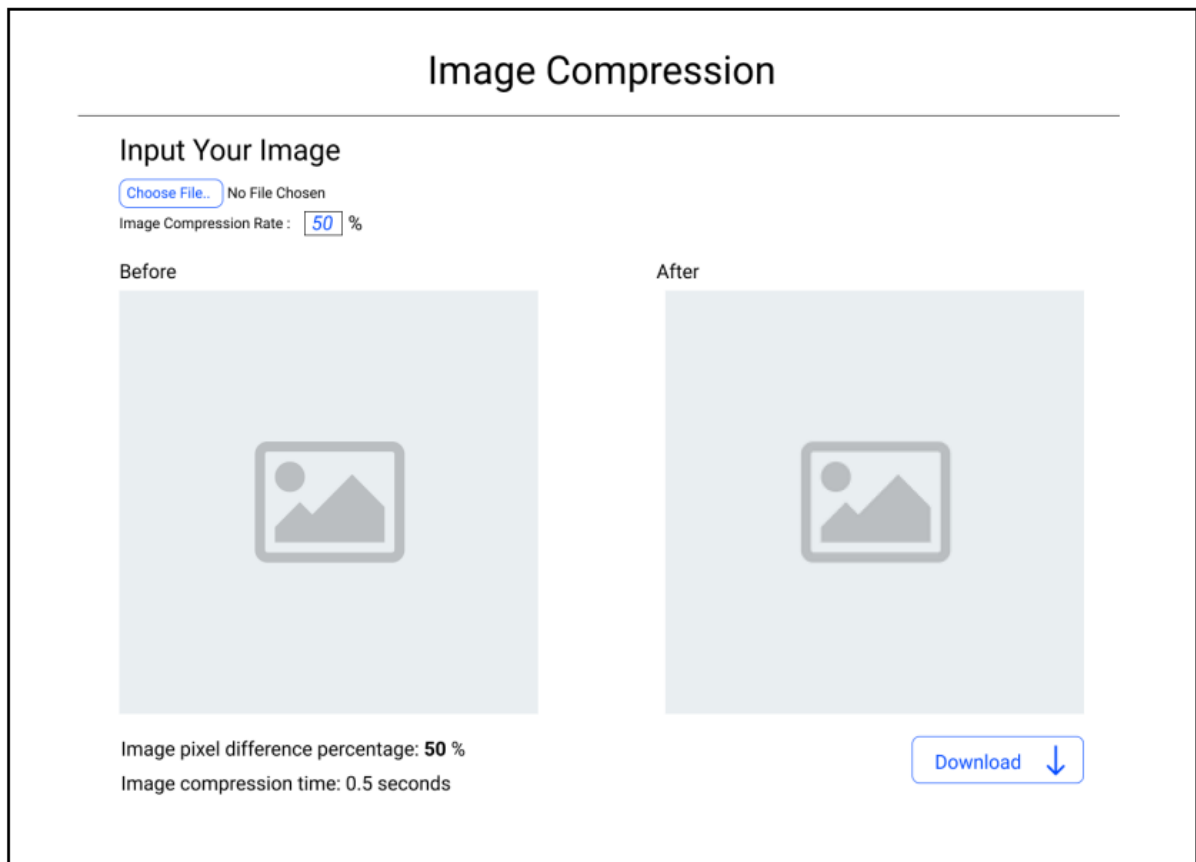
Pada kesempatan kali ini, kalian mendapatkan tantangan untuk membuat website kompresi gambar sederhana dengan menggunakan algoritma SVD.

PENGUNAAN 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.



The screenshot shows a web application titled "Image Compression". Under the heading "Input Your Image", there is a blue button labeled "Choose File.." and the text "No File Chosen". Below this, a label "Image Compression Rate :" is followed by a blue input field containing the number "50" and a percentage sign "%". The interface is divided into two columns: "Before" and "After". Each column contains a large light blue square with a gray image placeholder icon in the center. At the bottom left, the text "Image pixel difference percentage: 50 %" and "Image compression time: 0.5 seconds" is displayed. At the bottom right, there is a blue button labeled "Download" with a downward arrow icon.

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.

BAB 2 TEORI SINGKAT

I. Perkalian Matriks

Perkalian dua buah matriks $C_{m \times n} = A_{m \times r} \times B_{r \times n}$

Dimisalkan sebuah matriks $A = [a_{ij}]$ dan $B = [b_{ij}]$. Jika kedua matriks tersebut dikalikan, diperoleh matriks C dengan $C = A \times B = [c_{ij}]$ dan $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$ dengan syarat jumlah kolom pada matriks A sama dengan jumlah baris pada matriks B

Dalam bahasa Python, perkalian dua buah matriks $C_{m \times n} = A_{m \times r} \times B_{r \times n}$ adalah sebagai berikut

```
for i in range(m):
    for j in range(n):
        c[i][j] = 0
        for k in range(r):
            c[i][j] = c[i][j] + a[i][k] * b[k][j]
```

II. Nilai Eigen

Jika A adalah matriks $n \times n$, vektor tidak nol \mathbf{x} di R^n disebut vektor eigen dari A jika $A\mathbf{x}$ sama dengan perkalian suatu skalar λ dengan \mathbf{x} , yaitu $A\mathbf{x} = \lambda\mathbf{x}$

Skalar λ disebut nilai eigen dari A dan \mathbf{x} dinamakan vektor eigen yang berkoresponden dengan λ .

Kata “eigen” berasal dari Bahasa Jerman yang artinya “asli” atau “karakteristik”. Dengan kata lain, nilai eigen menyatakan nilai karakteristik dari sebuah matriks yang berukuran $n \times n$.

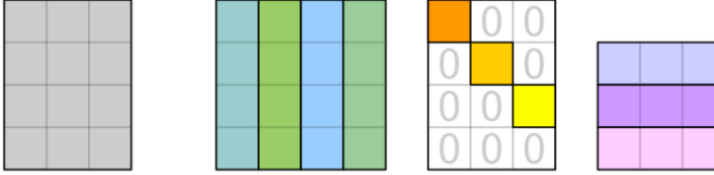
III. Vektor Eigen

Vektor eigen \mathbf{x} menyatakan vektor kolom yang apabila dikalikan dengan sebuah matriks $n \times n$ menghasilkan vektor lain yang merupakan kelipatan vektor itu sendiri. Dengan kata lain, operasi $A\mathbf{x} = \lambda\mathbf{x}$ menyebabkan vektor \mathbf{x} menyusut atau memanjang

dengan faktor λ dengan arah yang sama jika λ positif dan arah berkebalikan jika λ negatif.

IV. Matriks SVD

Suatu matriks M dapat difaktorkan menjadi hasil perkalian dari beberapa matriks sehingga dapat dituliskan sebagai $M = M_1 \times M_2 \times \dots \times M_n$. Salah satu metode untuk mendekomposisi matriks adalah SVD (*Singular Value Decomposition*) yang dapat digunakan baik untuk mendekomposisi matriks persegi maupun matriks non-persegi. Menggunakan metode SVD, matriks M berukuran $m \times n$ akan difaktorkan menjadi matriks U , Σ , dan V^T , yang masing masing berukuran $m \times m$, $m \times n$, dan $n \times n$.



$$\begin{matrix} \text{3x3 grid} & = & \text{3x3 grid} & \text{3x3 grid} & \text{3x3 grid} \\ \mathbf{M} & = & \mathbf{U} & \mathbf{\Sigma} & \mathbf{V}^* \\ m \times n & & m \times m & m \times n & n \times n \end{matrix}$$

Gambar 4. Ilustrasi dekomposisi matriks menggunakan metode SVD.

Matriks U (matriks singular kiri) merupakan matriks orthogonal yang kolomnya merupakan basis-basis ruang eigen dari matriks $A \times A^T$ yang sudah dinormalisasi menjadi vector satuannya. Sedangkan matriks V^T dan Σ diperoleh dari matriks $A^T \times A$, dengan matriks V^T (matriks singular kanan) merupakan matriks orthogonal yang baris-barisnya merupakan basis-basis ruang eigen $A^T \times A$ yang sudah dinormalisasi dan matriks Σ merupakan matriks diagonal dengan elemen diagonal utamanya merupakan nilai singular dari $A^T \times A$ yang tersusun terurut dari besar ke kecil. Nilai singular diperoleh dengan mengakarkan nilai-nilai eigen yang tidak nol. Salah satu pemanfaatan SVD adalah untuk kompresi gambar yang diterapkan dalam tugas besar ini.

BAB 3 IMPLEMENTASI PROGRAM

I. Struktur Program

Salah satu implementasi dari *Singular-Value-Decomposition* (SVD) adalah proses *image-compression*. Kami membuat sebuah website menggunakan front-end berupa HTML dan CSS, serta back-end menggunakan Python dengan *framework* Flask.

A. Front-End

Pada front-end, kami menggunakan HTML dan juga CSS untuk memperindah tampilan. Ada beberapa *class* yang kami buat di dalam HTML kami. Pertama, *class Settings* kami buat untuk bagian memilih gambar hingga sebelum *before after*. Kedua, *class container* yang kami buat untuk menuliskan *before after*. Ketiga, *class contain*, tempat area menampilkan gambar sebelum dan sesudah kompresi. Keempat, *class download_button*, yaitu area apabila user ingin mengunduh gambar yang telah dikompresi. HTML dan CSS ini diintegrasikan dengan Python Flask dalam menerima *input*, mengolah *input*, dan mengeluarkan *output*.

B. Back-End

Pada backend, kami menggunakan beberapa library pendukung berupa *Numpy* untuk melakukan operasi-operasi terhadap matriks, *PIL* dan *OpenCV* untuk pemrosesan gambar (konversi gambar-matriks), *timeit* untuk menghitung waktu jalannya program, *werkzeug.utils* untuk fungsi *secure_filename()*, *os* untuk fungsi *join()*, *python-dotenv* untuk mengatur FLASK-APP dan FLASK-ENV, serta *Flask* untuk menghubungkan front-end dan back-end program. Bagian back-end program ini terdiri dari 4 file program python yaitu *app.py* yang merupakan program utama yang akan dijalankan di server, *img_compress.py* yang berisi fungsi utama untuk mengompresi gambar, *svd.py* yang berisi fungsi/prosedur untuk melakukan dekomposisi matriks dengan metode SVD, dan *vektor.py* yang berisi fungsi untuk menghasilkan nilai dan vector eigen dari suatu matriks. Fungsi/prosedur tersebut adalah sebagai berikut:

- *home()*: Fungsi ini digunakan untuk merender ke template *web.html* ke browser.
- *allowed_extension(filename)*: Fungsi ini digunakan untuk memvalidasi jenis ekstensi file yang dapat diterima yaitu hanya file gambar.

- `form()`: Fungsi ini digunakan untuk mendapatkan input dari form pada `web.html` yang akan diolah dan mengembalikan lagi hasil pengolahannya.
- `display(filename)`: Fungsi ini digunakan untuk menampilkan gambar *before* dan *after* dari input gambar yang dikompresi.
- `main(file_name, c_rate)`: Fungsi ini merupakan fungsi utama yang akan mengompresi gambar input sesuai `c_rate` (*compression rate*) dengan menggunakan fungsi yang sudah didefinisikan di file `svd.py` lalu mengembalikan ukuran awal, ukuran akhir, persentase pixel, waktu jalannya program, dan gambar hasil kompresi.
- `matrixSVD(M)`: Fungsi ini menghasilkan matriks U , S , dan V^T hasil dekomposisi matriks M menggunakan metode SVD.
- `compress(U, s, V_T, rate)`: Fungsi ini digunakan untuk mengambil hanya k kolom dari matriks U , k nilai singular (s), dan k baris dari matriks V^T , dengan nilai k dihitung sesuai dengan input `rate` (skala kompresi) yang dimasukkan.
- `multiplySVD(nU, ns, nV_T)`: Fungsi ini digunakan untuk menghitung kembali matriks M dengan mengalikan matriks U , S , dan V^T yang telah dikompresi menggunakan fungsi `compress()` di atas.
- `svdCompression(M, c_rate)`: Fungsi ini merupakan penggabungan dari fungsi `matrixSVD()`, `compress()`, dan `multiplySVD()` di atas.
- `svdColor(M, c_rate)`: Fungsi ini digunakan untuk melakukan kompresi gambar menggunakan fungsi `svdCompression()` untuk gambar berwarna (RGB).
- `svdGrayscale(M, c_rate)`: Fungsi ini digunakan untuk melakukan kompresi gambar menggunakan fungsi `svdCompression()` untuk gambar grayscale atau hitam putih.
- `svdColorPNG(M, c_rate)`: Fungsi ini digunakan untuk melakukan kompresi gambar menggunakan fungsi `svdCompression()` untuk gambar berwarna yang memiliki transparansi (RGBA).
- `eigen(M)`: Fungsi ini digunakan untuk menghitung nilai eigen dan vector eigen menggunakan algoritma *Simultaneous Power Iteration / Orthogonal Iteration* yang memanfaatkan dekomposisi QR.

II. Garis Besar Program

Alur jalannya program kompresi gambar ini adalah sebagai berikut. Pertama-tama, program akan menerima input gambar dan skala kompresi yang dimasukkan pengguna dari web, lalu gambar tersebut akan disimpan ke directory src/static/upload apabila semua inputnya sudah valid. Setelah itu fungsi form() akan memanggil fungsi main() untuk melakukan kompresi gambar. Gambar tersebut akan diubah ke dalam bentuk matriks (M) menggunakan library PIL dan di-split menjadi bagian RGB menggunakan library OpenCV jika merupakan gambar RGB atau RGBA. Setelah itu dilakukan perhitungan nilai eigen dan vector eigen yang sudah ternormalisasi dari matriks MM^T untuk masing masing layer (R/G/B/A) menggunakan algoritma *simultaneous power iteration*.

Power iteration adalah sebuah metode aljabar linear untuk mengaproksimasi nilai eigen dan vektor eigen dari sebuah matriks. Dimisalkan A bersifat simetris, maka dekomposisi eigen dari A adalah $A = Q\Lambda Q^T$ dan $A^k = Q\Lambda^k Q^T$ dengan q_i adalah kolom dari Q. Maka λ_1 adalah nilai eigen dominan dari A jika $|\lambda_1| > |\lambda_i|$ untuk semua $i = 2, \dots$ dan vektor q_1 adalah vektor eigen dominannya. Untuk mencari vektor eigennya digunakan algoritma power method dengan menentukan aproksimasi awal vektor eigen x_0 dengan nilai apapun, lalu dilakukan perhitungan $x_1 = Ax_0$, $x_2 = AAx_0 = A^2x_0$, ..., $x_k = A^kx_0$ hingga diperoleh nilai aproksimasi vektor eigen (x_k) yang baik. Setelah diketahui vektor eigennya (v), nilai eigen (λ) dapat dicari menggunakan Rayleigh Quotient, yaitu:

$$\lambda = \frac{v^T A^T v}{v^T v} = \frac{(Av)^T v}{v^T v}$$

Simultaneous Power Iteration merupakan pengembangan dari power method di atas yaitu dibanding mencari vektor eigen satu persatu, dengan simultaneous power iteration, matriks A dikalikan dengan q_i, \dots, q_r (matriks Q). Pada setiap langkah, vektor tersebut dinormalisasi menggunakan dekomposisi QR. Algoritmanya adalah sebagai berikut:

Pilih Q_0 untuk aproksimasi awal (random), lalu hitung

$$Q_k = AQ_{k-1}, \quad Q_k R_k = Z_k \text{ hingga aproksimasi cukup baik}$$

Sehingga lama kelamaan R akan berbentuk matriks segitiga atas. Maka nilai eigen akan terdapat pada diagonal utama dari R dan vektor eigen merupakan kolom-kolom dari Q.

Dari nilai eigen dan vector eigen tersebut bisa diperoleh matriks U dan Σ sesuai perhitungan pada BAB 2 – Teori Singkat. Sedangkan nilai matriks V^T diperoleh dengan memanfaatkan invers yaitu $V^T = U^{-1}S^{-1}(M.M^T)$. Hal ini dilakukan karena dalam perhitungan vektor eigen, bisa saja didapatkan nilai vector eigen yang merupakan bentuk negatifnya, misalkan vector eigen (a, b, c) yang sebenarnya sama dengan vector eigen $(-a, -b, -c)$. Meskipun nilainya sama saja, namun dalam perhitungan SVD, nilai ini sangat berpengaruh karena ketika dilakukan perhitungan kembali matriks awal, hasil konversi ke gambarnya akan berubah atau berbeda.

Setelah diperoleh matriks U , S , dan V^T , hanya akan diambil k kolom dari matriks U , k nilai singular (s), dan k baris dari matriks V^T . Nilai k ini dihitung menggunakan rumus berikut:

$$\begin{aligned} compressed_size &= \frac{c_rate}{100} \times initial_size \\ (m \times k + k + k \times n) &= \frac{c_rate}{100} \times (m \times n) \\ k &= \frac{c_rate}{100} \times \frac{m \times n}{m + n + 1} \end{aligned}$$

Selanjutnya akan dihitung hasil perkalian matriks U , S , dan V^T yang sudah dikompresi tersebut dan matriks hasilnya akan digabungkan dengan layer (R/G/B/A) yang sudah dikompres juga jika ada dan dikonversi kembali membentuk gambar. Lalu gambar tersebut akan di-save di *directory* src/static/upload dan ditampilkan ke website beserta nilai ukuran awal, ukuran akhir, persentase perubahan pixel, dan waktu jalannya program.

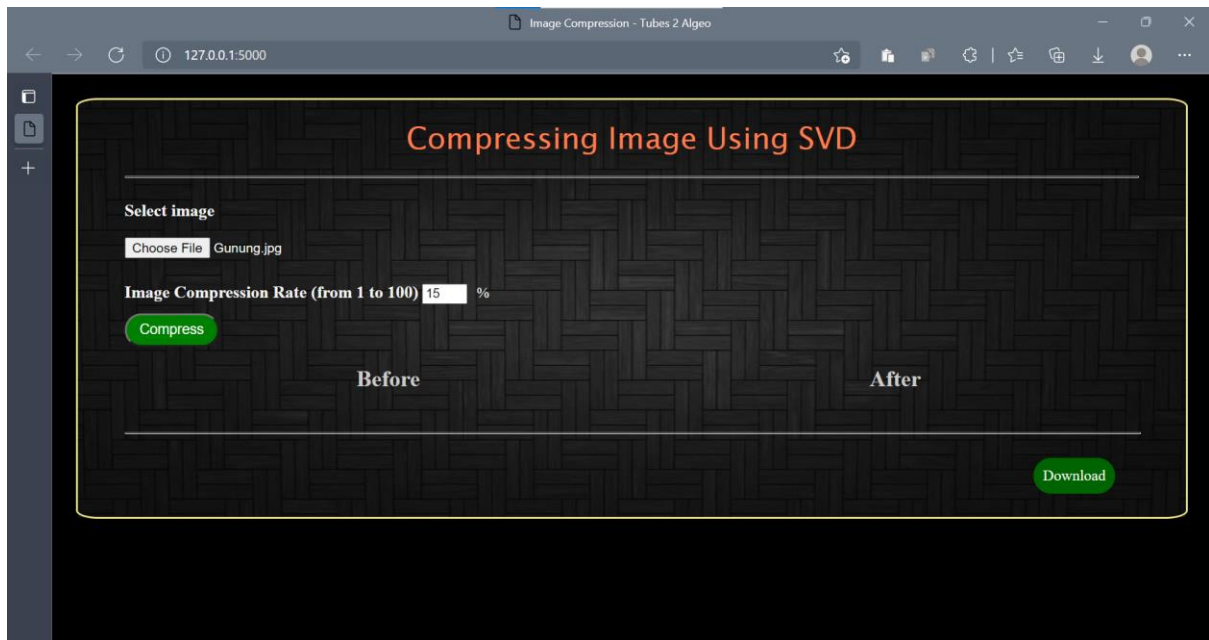
Langkah-langkah untuk menjalankan program adalah sebagai berikut:

1. Buka terminal (cmd), lalu *install* library yang diperlukan yaitu Pillow, opencv-python, flask, numpy, dan python-dotenv dengan cara ketik: `pip install nama_library`
2. Ubah directory terminal ke folder program yaitu Algeo02-20046/src
3. Ketik `flask run` pada terminal untuk menjalankan program
4. Buka <http://localhost:5000/> atau <http://127.0.0.1:5000/> di browser
5. Akan muncul tampilan web dengan title “Image Compression – Tubes 2 Algeo”

6. Upload gambar yang ingin di-*compress* dengan klik tombol *choose file* lalu pilih gambarnya atau *drag & drop* gambar ke area tombol *choose file*.
7. Masukkan skala kompresi yang diinginkan dan klik tombol *Compress* untuk memproses
8. Akan muncul tanda loading di atas webpage, silakan tunggu hingga gambar hasil kompresinya muncul.
9. Untuk mendownload gambar klik tombol *Download* di kanan bawah
10. Untuk menghentikan program, ketik ctrl+c pada terminal (cmd)

BAB 4 EKSPERIMEN

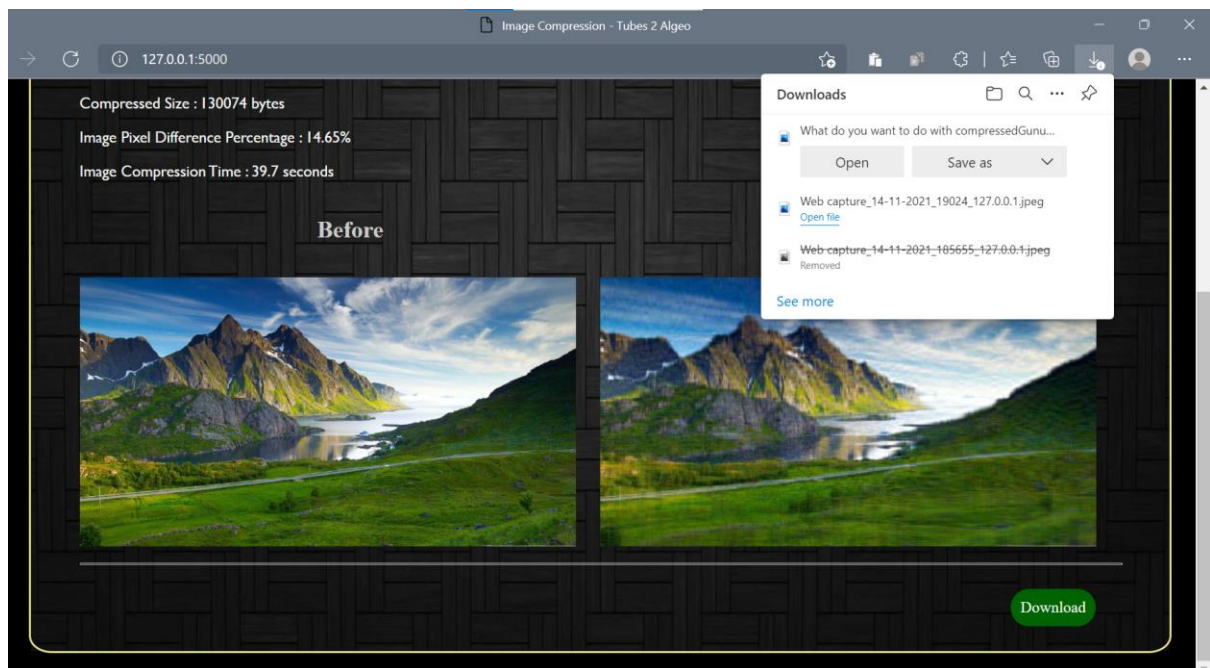
- Tampilan saat memasukkan input file gambar dan skala kompresi



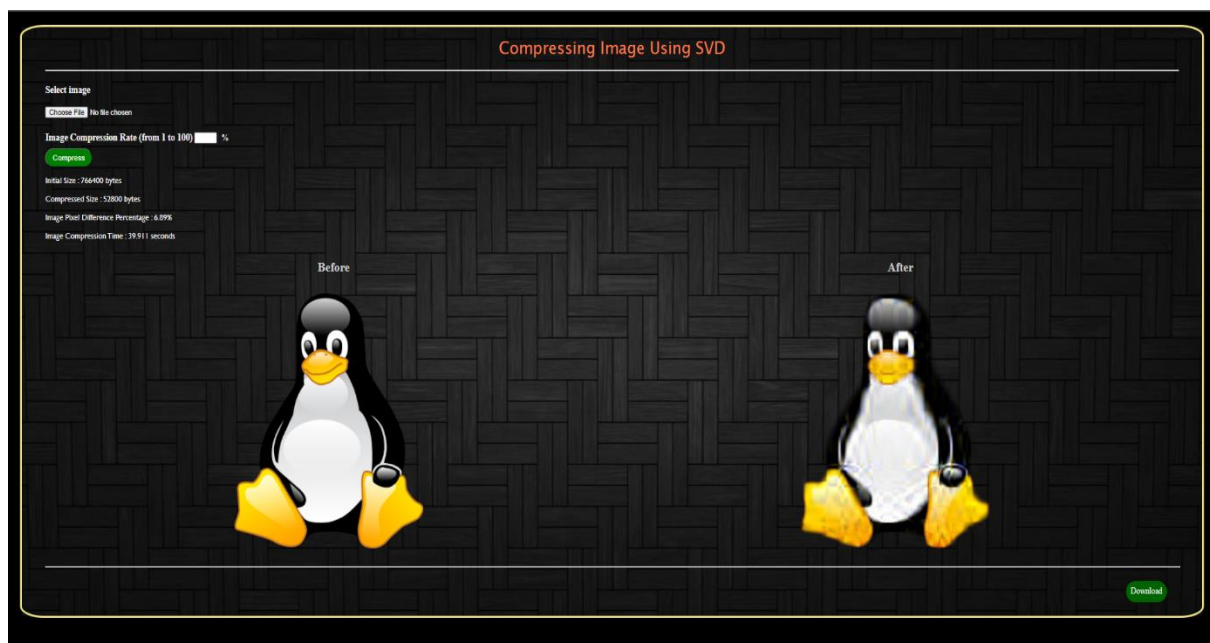
- Tampilan gambar hasil kompresi (15%)



- Tampilan saat akan mengunduh gambar hasil kompresi



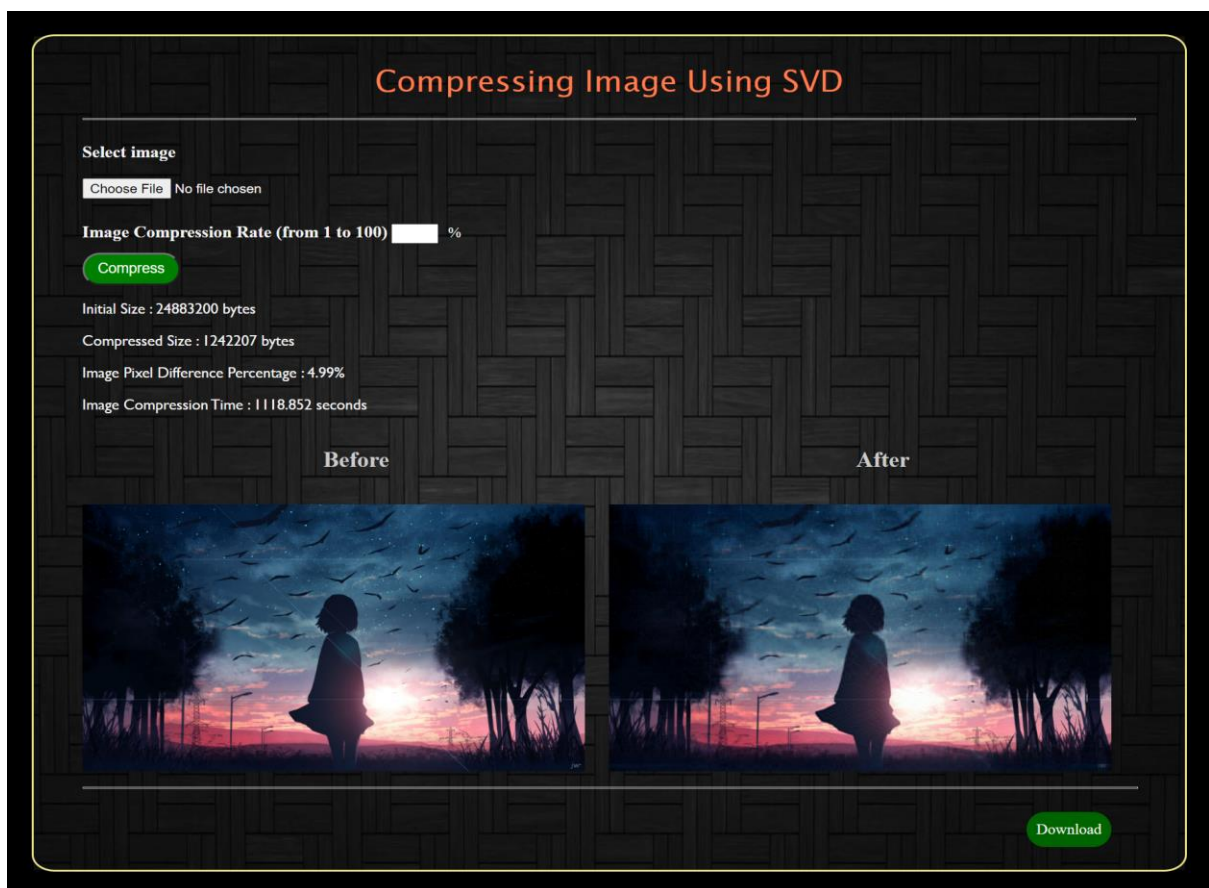
- Tampilan hasil kompresi gambar PNG (7%)



- Tampilan hasil kompresi gambar grayscale (30%)



- Tampilan hasil kompresi gambar beresolusi tinggi (4K) (5%)



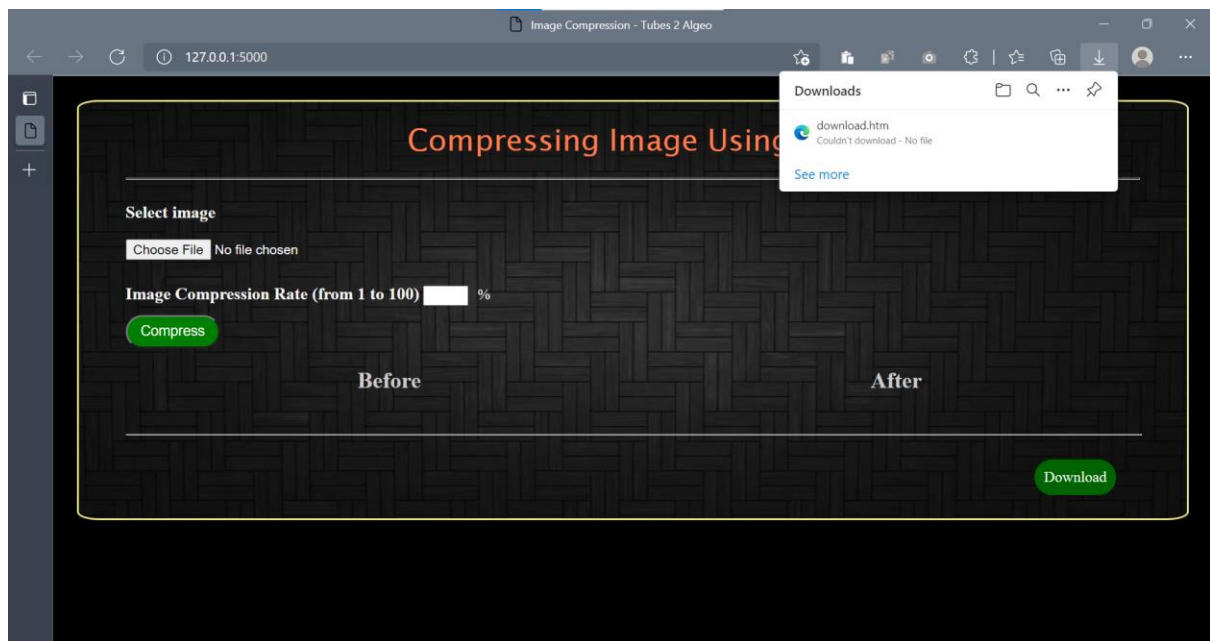
- Tampilan ketika input invalid (skala kompresi)

The screenshot shows a web application titled "Compressing Image Using SVD". It features a "Select image" section with a "Choose File" button and a file name "4K.jpg". Below this is an "Image Compression Rate (from 1 to 100)" input field with a value of "-3" and a percentage sign. A green "Compress" button is present. A tooltip message states: "Value must be greater than or equal to 1." Below the input field are two columns labeled "Before" and "After". At the bottom right is a green "Download" button.

- Tampilan ketika input invalid (gambar)

The screenshot shows the same web application titled "Compressing Image Using SVD". In the "Select image" section, the "Choose File" button is active, and the text "No file chosen" is displayed. Below this, the "Image Compression Rate" input field is empty, and a tooltip message states: "Please select a file." The "Compress" button is green. Below the input field are two columns labeled "Before" and "After". At the bottom right is a green "Download" button.

➤ Tampilan ketika input invalid (download)



Dari hasil eksperimen, kompresi gambar dapat dilakukan dengan baik, tetapi masih membutuhkan waktu yang cukup lama karena menggunakan algoritma yang kurang efisien. Program kami berhasil mempertahankan transparansi gambar yang memiliki format RGBA, RGB, dan grayscale. Untuk gambar PNG yang matriksnya berupa gabungan matriks RGBA, transparansi masih bisa dipertahankan, namun untuk gambar PNG yang hanya terdiri dari 1 matriks, warna dan transparansinya belum bisa dipertahankan.

BAB 5 KESIMPULAN, SARAN, DAN REFLEKSI

Berdasarkan tugas besar yang telah kami buat, dapat disimpulkan bahwa algoritma SVD dapat membantu proses kompresi gambar. Pada dasarnya, sebuah gambar dapat dimodelkan dengan sebuah matriks, dan dari matriks tersebut, dapat diolah menjadi bentuk matriks lainnya dalam waktu tertentu sebagai proses kompresi gambar. Informasi terpenting dari gambar tersebut tersimpan pada k kolom pertama matriks U , k nilai singular pertama, dan k baris pertama dari matriks V , sehingga dengan hanya mengambil data tersebut, dapat direkonstruksi matriks awal dengan resolusi yang lebih kecil.

Algoritma yang kami gunakan pada program ini masih membutuhkan waktu yang cukup lama dalam prosesnya sehingga untuk pengembangan selanjutnya disarankan untuk mencari algoritma lain yang lebih efisien.

DAFTAR REFERENSI

Anton, Howard dan Chris Rorres. 2014. *Elementary Linear Algebra 11th Edition*. Canada: Anton Textbooks, Inc.

Budai, Adam. 2018. *QR Algorithm: Finding Eigenvalues*.

<https://adamtiger.github.io/ai/math/2018/01/02/qr.html>

Mathews, Brady. 2014. *Image Compression using Singular Value Decomposition (SVD)*.

http://www.math.utah.edu/~goller/F15_M2270/BradyMathews_SVDImage.pdf

Matrix Computations (CS 6210). 2009. *Orthogonal Iteration*.

<https://www.cs.cornell.edu/~bindel/class/cs6210-f09/lec27.pdf>