

PROJECT SVD IMAGE COMPRESSION

ALJABAR LINEAR



DISUSUN OLEH: KELOMPOK 8



Cholif Bima A.

L0123040



Lutfiyah Istiana

L0124022



Shafa Rifkika N.F.

L0124031

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA

UNIVERSITAS SEBELAS MARET

2025

DAFTAR ISI

DAFTAR ISI.....	2
BAB I.....	3
DESKRIPSI MASALAH.....	3
1.1. Abstrak.....	3
1.2. Penggunaan Program.....	5
1.3. Spesifikasi Tugas.....	5
BAB II.....	7
TEORI SINGKAT.....	7
2.1. Perkalian Matriks.....	7
2.2. Nilai Eigen.....	7
2.3. Vektor Eigen.....	7
2.4. Matriks SVD.....	8
Dalam matriks SVD setiap matriks A dapat diuraikan sebagai:.....	8
BAB III.....	9
IMPLEMENTASI PROGRAM.....	9
BAB IV.....	16
EKSPERIMEN/EKSEKUSI PROGRAM.....	16
4.1. Eksperimen.....	16
4.2. Kesimpulan Eksperimen.....	21
BAB V.....	22
KESIMPULAN DAN SARAN.....	22
5.1. Kesimpulan.....	22
5.2. Saran.....	22
5.3. Refleksi.....	23
DAFTAR PUSTAKA.....	24

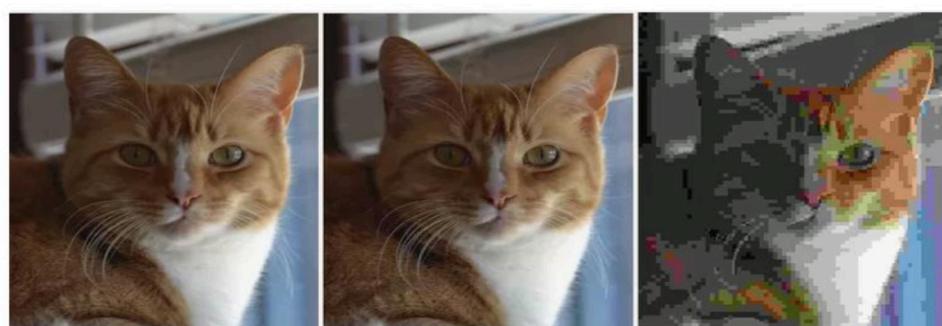
BAB I

DESKRIPSI MASALAH

1.1. Abstrak

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

Kompresi image merupakan suatu tipe kompresi data yang dilakukan pada image digital. Dengan kompresi image, suatu file Image digital dapat dikurangi ukuran filenya dengan baik tanpa mempengaruhi kualitas image secara signifikan. Terdapat berbagai metode dan algoritma yang digunakan untuk kompresi Image pada zaman modern ini. Image 1 menunjukkan contoh kompresi image dengan berbagai tingkatan.



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.

Image 1, Contoh kompresi image dengan berbagai tingkatan

<https://www.lifewire.com/the-effect-of-compression-on-photographs>
[.com/the-effect-of-compression-on-photographs](https://www.lifewire.com/the-effect-of-compression-on-photographs)

Salah satu algoritma yang dapat digunakan untuk kompresi Image 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 (1) berikut

$$A_{m \times n} = U_{m \times m} S_{m \times n} V^T_{n \times n} \quad (1)$$

Matriks U adalah matriks yang kolomnya terdiri dari vektor eigen ortonormal dari matriks AAT. 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 ATA. Matriks ini menyimpan informasi yang penting terkait kolom-kolom matriks awal, dengan informasi terpenting disimpan dalam baris pertama.

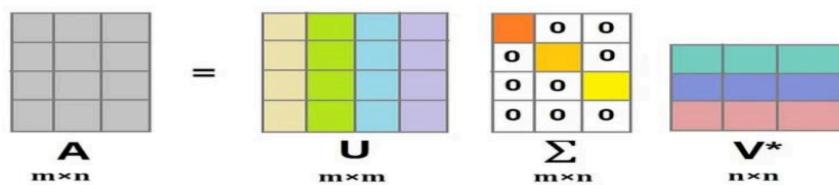


Image 2. Ilustrasi Algoritma SVD dengan rank k

Dapat dilihat di image di atas bahwa dapat direkonstruksi Image 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 dapat dilihat dalam Image 2. Kita dapat mengaproksimasi suatu Image yang mirip dengan image 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 Image yang terkompresi dengan ukuran yang lebih kecil dibanding Image awal.

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

1.2. Penggunaan Program

Berikut ini adalah input yang dimasukkan sebagai user untuk eksekusi program dapat dilihat dalam Image 3.

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

Tampilan layout dari aplikasi web yang akan dibangun kurang lebih dapat dilihat dalam Image 2. Anda dapat mengubah layout selama layout masih terdiri dari komponen yang sama.



Image 3 Layout image compression dengan SVD

1.3. Spesifikasi Tugas

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

1. Website mampu menerima file image beserta input tingkat kompresi image (dibebaskan formatnya).

2. Website mampu menampilkan image input, output, runtime algoritma, dan persentase hasil kompresi image (perubahan jumlah pixel image).
3. File output hasil kompresi dapat diunduh melalui website.
4. Kompresi image tetap mempertahankan warna dari image asli.
5. Bahasa pemrograman yang boleh digunakan adalah Python, Javascript, dan Go.
6. Penggunaan framework untuk back end dan front end website dibebaskan. Contoh framework website yang bisa dipakai adalah Flask, Django, React, Vue, dan Svelte.
7. Kalian dapat menambahkan fitur fungsional lain yang menunjang program yang anda buat (unsur kreativitas diperbolehkan/dianjurkan).
8. Program harus modular dan mengandung komentar yang jelas.
9. Diperbolehkan menggunakan library pengolahan citra seperti OpenCV2, PIL, atau image dari Go.
10. Bahasa program yang digunakan adalah Python dengan menggunakan fungsi-fungsi yang sudah ada maupun dapat membuat fungsi sendiri.

BAB II

TEORI SINGKAT

2.1. Perkalian Matriks

Dalam SVD perkalian matriks digunakan untuk membangun kembali matriks dari dekomposisi. Perkalian dua matriks:

Jika A adalah $m \times n$ dan B adalah $n \times p$, maka hasil $C = A \cdot B$ adalah $m \times p$.

Setiap elemen C_{ij} adalah:

$$C_{ij} = \sum_{k=1}^n A_{il} \cdot B_{kj}$$

Sebelum kompresi:

$$A = U \cdot S \cdot V^T$$

Setelah kompresi:

$$A_k = U_k \cdot S_k \cdot V_k^t$$

2.2. Nilai Eigen

Untuk nilai matriks kuadrat A , nilai λ disebut *nilai eigen* jika:

$$Av^\rightarrow = \lambda v^\rightarrow$$

Artinya, vektor v^\rightarrow tetap pada arah yang sama setelah dikalikan matriks A .

Untuk menemukan λ :

$$\det(A - \lambda I) = 0$$

Dalam SVD, nilai eigen digunakan untuk menghitung nilai singular:

1. SVD menggunakan matriks $A^T A$ dan AA^T
2. Nilai eigen dari $A^T A$ atau AA^T adalah kuadrat dari nilai singular.

2.3. Vektor Eigen

Vektor eigen v^\rightarrow adalah solusi dari:

$$(A - \lambda I) v^\rightarrow = 0$$

Kolom matriks U dibentuk dari vektor eigen dari AA^T

Kolom matriks V dibentuk dari vektor eigen dari $A^T A$

Vektor-vektor ini menyusun transformasi orthonormal pada dominan (U) dan kodomain (V) dari matriks A .

2.4. Matriks SVD

Dalam matriks SVD setiap matriks A dapat diuraikan sebagai:

$$A = U \cdot S \cdot V^T$$

Dimana:

U adalah vektor eigen orthonormal dari AA^T ,

V adalah vektor eigen orthonormal $A^T A$

S adalah nilai singular (akar) dari nilai eigen dari $A^T A$

BAB III

IMPLEMENTASI PROGRAM

3.1. Kekas yang Digunakan

a. Python

Python adalah bahasa pemrograman tingkat tinggi yang populer karena sintaknya sederhana dan mudah dibaca. Dalam projek ini, Python digunakan sebagai dasar untuk mengembangkan seluruh logika *back-end*, termasuk proses kompresi gambar menggunakan algoritma SVD. Python sangat cocok untuk tugas-tugas matematis dan ilmiah karena memiliki banyak library yang kuat. Kemampuan dalam pengolahan matriks dan integrasi dengan framework web membuatnya ideal untuk projek ini.

b. Flask

Flask adalah framework web ringan berbasis python yang sangat cocok untuk membangun server aplikasi kecil hingga menengah. Dalam projek ini, flask digunakan untuk membangun server aplikasi yang menangani routing, file upload, pemrosesan SVD, dan menampilkan hasil kompresi. Flask bersifat minimalis namun fleksibel, sehingga memudahkan pengembang menambahkan fitur sesuai kebutuhan. Flask juga mendukung templating HTML menggunakan Jinja2, yang memudahkan integrasi antara back-end dan front-end.

c. NumPy

NumPy adalah library Python sangat populer untuk perhitungan numerik di aljabar linear. Dalam projek ini, NumPy digunakan untuk melakukan proses SVD dari matriks representasi gambar. Fungsi `np.linalg.svd()` digunakan untuk menghitung U , S , dan V^T yang dibutuhkan dalam proses kompresi. Tanpa NumPy,

perhitungan ini akan jauh lebih kompleks dan lambat. NumPy juga memungkinkan manipulasi array dan matriks gambar secara tepat.

d. HTML

HTML digunakan untuk struktur dasar halaman web. Dalam projek ini, HTML membentuk tampilan form upload, tombol, teks, dan layout web. HTML bekerja sama dengan flask melalui sistem templating, memungkinkan data dari back-end ditampilkan langsung di halaman web. HTML menjadi fondasi utama dari front-end, sebelum diberi gaya dan fungsi tambahan menggunakan CSS dan JavaScript. Tanpa HTML ini, antarmuka pengguna tidak bisa dibentuk.

e. Tailwind CSS

Tailwind CSS adalah framework CSS modern berbasis utility-first yang memudahkan styling HTML secara langsung dengan class, dalam projek ini digunakan untuk membuat tampilan web yang bersih, responsif dan estetik. Developer dapat menambahkan gaya langsung di dalam elemen HTML tanpa menulis file CSS sepanjang. Tailwind mempercepat proses pengembangan antarmuka.

f. Git

Git adalah sistem version control yang digunakan untuk mencatat dan mengelola perubahan dalam projek. Dengan Git, setiap pengembang dapat dicatat dalam bentuk commit, dan kolaborasi antar anggota tim menjadi lebih mudah. Git juga memungkinkan rollback ke versi sebelumnya jika terjadi kesalahan. Dalam projek ini, Git digunakan untuk mengelola source code serta mengintegrasikan projek ke GitHub untuk deployment otomatis. Pengguna Git memastikan pekerjaan lebih terstruktur dan terdokumentasi dengan baik.

g. GitHub

GitHub adalah platform hosting untuk repository Git juga menyediakan layanan kolaborasi dan integrasi. Dalam projek ini, GitHub berfungsi sebagai pusat penyimpanan source code sekaligus tempat mengatur workflow GitHub Actions. Setiap kali code di *push* ke *branch main*, GitHub secara otomatis menjalankan deployment ke Azure Web App. GitHub juga memungkinkan pembagian tugas dan dokumentasi projek dengan lebih mudah, integrasi GitHub menjadikan projek lebih profesional dan siap produksi.

h. GitHub Actions

GitHub Actions adalah fitur CI/CD dari GitHub yang memungkinkan workflow otomatis seperti build, test, dan deploy. Dalam projek ini, GitHub Actions digunakan untuk membuat pipeline otomatis yang membangunprojek Python dan mendistribusikannya ke Azure Web Ap. Setiap kali terjadi perubahan pada branch main, workflow ini akan membuat zip file dan mengirimkannya ke server Azure. Hal ini meminimalisir kesalahan manual dan mempercepat proses deployment. GitHub Actions memberikan efisiensi tinggi dan profesionalitas pada proses pengembangan.

i. Azure

Azure Web App adalah layanan hosting dari Microsoft Azure untuk aplikasi web. Dalam projek ini, Azure Web App menjadi tempat men-deploy aplikasi Flask agar bisa diakses melalui internet. Setelah GitHub Actions selesai membangun dan mengirimkan kode, Azure akan menjalankan aplikasi secara otomatis. Azure mendukung aplikasi Python dengan konfigurasi dan performa tinggi.

j. **Venv (Virtual Environment)**

Venv adalah fitur Python untuk membuat virtual environment, yaitu lingkungan isolasi untuk package dan library projek tertentu. Dalam projek ini, venv digunakan untuk memastikan semua dependency (seperti Flask, NumPy, Pillow) tidak bercampur dengan projek lain. Dengan venv, kamu bisa menghindari konflik versi library dan menjaga projek tetap rapi. Ini juga sangat penting saat deploy ke server karena hanya menginstal dependensi yang tertulis.

k. **NPM (Node Package Manager)**

NPM adalah pengelola package yang memungkinkan developer JavaScript menemukan dan menginstal package kode ke aplikasi jaringan atau server-side. Dalam projek ini NPM digunakan untuk menginstal dan membangun Tailwind CSS, karena Tailwind bekerja paling optimal ketika dipasang melalui sistem build berbasis [Node.js](#). NPM memungkinkan developer mengelola versi Tailwind, plugin, serta file konfigurasi seperti ini. Dengan penggunaan NPM, pengelolaan file CSS jadi lebih cepat dan efisiensi.

l. **Tailwind Css**

Tailwind CSS adalah framework CSS yang mengutamakan utilitas yang memudahkan pembuatan desain custom tanpa harus menulis CSS custom. Framework ini memungkinkan tim penulis menerapkan utilitas individual langsung di HTM, yang mampu menciptakan tata letak yang responsif dan menarik.

3.2. Algoritma yang Digunakan

a. **Algoritma Singular Value Decomposition (SVD)**

Svd adalah algoritma utama yang digunakan untuk melakukan dekomposisi matriks menjadi tiga komponen: $A = U \cdot S \cdot V^T$. Dalam konteks gambar, matriks A merepresentasikan piksel citra (biasanya dalam grayscale atau tiap kanal warna RGB). Dengan SVD, gambar bisa direkonstruksi kembali menggunakan sebagian nilai singular terbesar, sehingga menghasilkan versi gambar yang dikompresi. Proses ini sangat efisien karena sebagian besar informasi dalam gambar disimpan di nilai singular awal. Rekonstruksi gambar hasil kompresi dilakukan dengan mengalikan kembali $U_k \cdot S_k \cdot V_k^T$ untuk rank k tertentu. SVD cocok untuk kompresi karena sifatnya yang mampu mengurangi dimensi informasi tanpa perlu banyak kehilangan kualitas visual.

b. Algoritma Perkalian Matriks

Perkalian matriks merupakan proses penting dalam merekonstruksi gambar setelah kompresi dengan SVD. Matriks $U_k \cdot S_k \cdot V_k^T$ dikalikan satu per satu untuk menghasilkan matriks baru sebagai citra hasil kompresi. Perkalian ini mengikuti aturan aljabar linier, yaitu $m \times k \cdot k \times k \cdot k \times n = m \times n$. Perkalian ini memungkinkan kita mendapatkan matriks aproksimasi dari gambar asli dengan dimensi dan informasi yang lebih ringan. Ini sangat penting dalam menjaga ukuran file gambar tetap kecil namun tetap terlihat jelas.

c. Algoritma konversi Gambar ke Matriks

Untuk menerapkan SVD pada gambar, pertama-tama gambar harus diubah menjadi bentuk matriks numerik. Algoritma ini melibatkan pembacaan gambar menggunakan Pillow, kemudian dikonversi menjadi array NumPy. Setelah hasil kompresi diperoleh, array tersebut dikonversi kembali menjadi gambar dengan `image.fromarray` dan disimpan ke file. Proses ini memastikan

format file tetap terjaga, serta hasil kompresi bisa langsung digunakan dan diunduh. Tanpa algoritma konversi ini, gambar tidak bisa diolah secara matematis maupun ditampilkan kembali.

d. Algoritma Upload dan Menyimpan File

Dalam projek ini juga menggunakan algoritma untuk menerima gambar dari user melalui form HTML. File yang diupload akan disimpan sementara ke server. File diamankan menggunakan `secure_filename()` agar tidak terjadi injeksi nama atau error sistem file. Setelah file disimpan, gambar siap untuk diproses lebih lanjut menggunakan SVD. Setelah kompresi selesai, file hasilnya juga disimpan di server dan bisa diunduh kembali oleh user.

e. Algoritma Penerimaan dan Penanganan Input User

Dalam aplikasi web, *input* dari *user* berupa file gambar dan nilai kompresi ditangani oleh *Flask*. *Flask* akan melakukan validasi dasar seperti apakah file ada, apakah ekstensinya valid, dan apakah nilai kkk berada dalam rentang yang logis. Jika valid, proses selanjutnya dilakukan jika tidak, *user* akan diarahkan kembali atau ditampilkan pesan *error*. Algoritma ini memastikan aplikasi berjalan aman dan tidak *crash* karena input yang tidak sesuai.

3.3. Alur Program

1. Menerima file dari pengguna.
2. Memvalidasi jenis file (apakah .jpg, .jpeg, atau .png).
3. Mengecek apakah gambar sudah ada di cache. Jika sudah ada, maka hasil kompresi akan diambil dari cache sehingga lebih menghemat waktu.
4. Jika tidak ada di cache, maka gambar akan dikompresi terlebih dahulu menggunakan SVD dan kemudian disimpan hasilnya ke dalam cache.
5. Menghitung waktu kompresi, rasio kompresi, dan dimensi gambar.

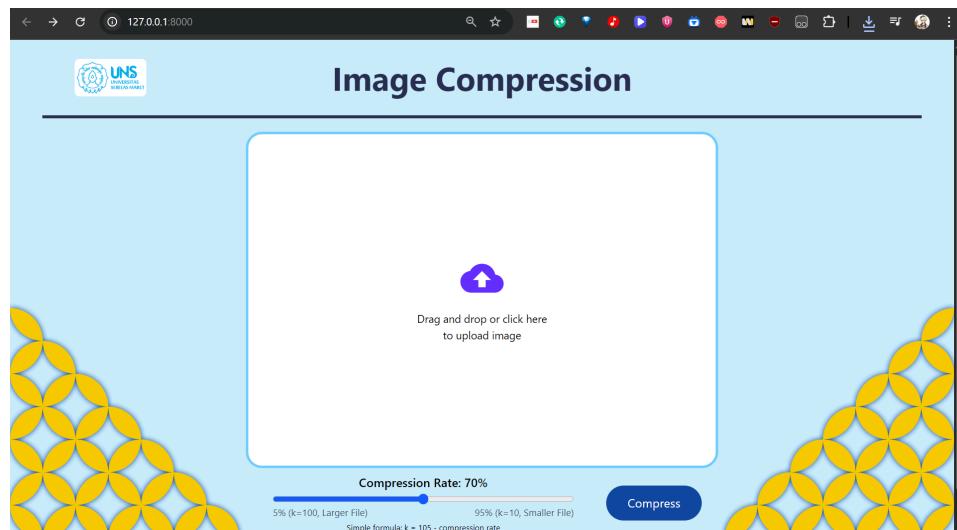
6. Menampilkan hasil kompresi kepada pengguna beserta informasi-informasi mengenai hasil dari proses kompresi tersebut.

BAB IV

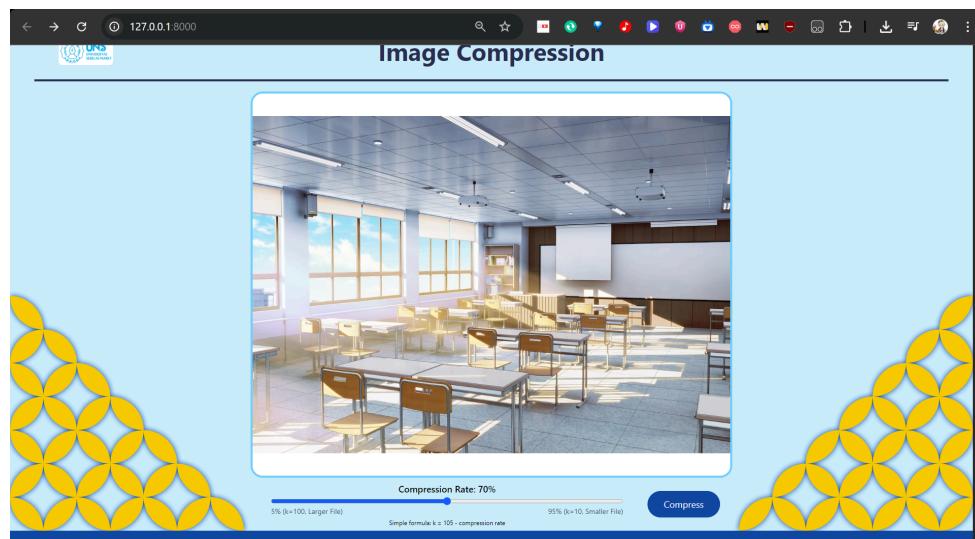
EKSPERIMENT/EKSEKUSI PROGRAM

4.1. Eksperimen

a. Tampilan Awal Program

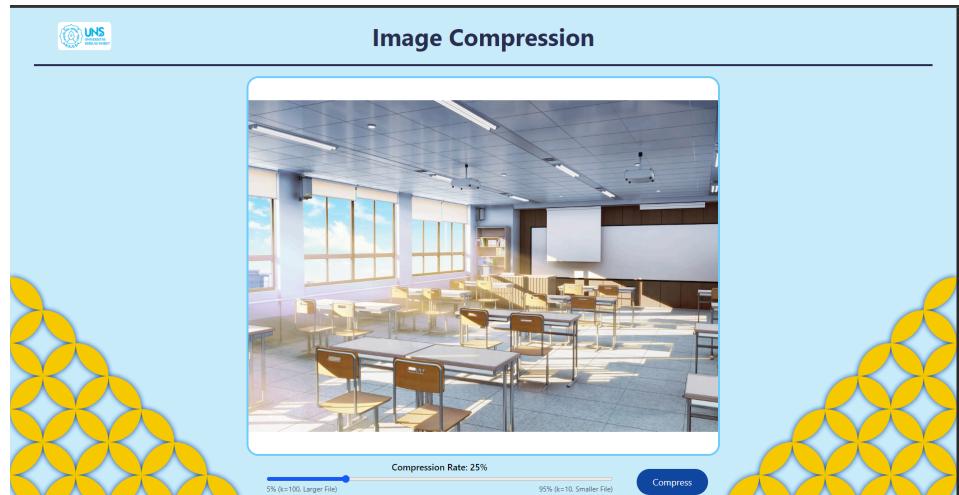


Disini user akan membuka tampilan utama program ketika dijalankan, user bisa mengupload gambar yang ingin dikompress dan juga bisa *adjust compression rate* dalam persen (%) untuk mengatur nilai k-value yang akan menjadi penentu seberapa besar gambar akan dikompres, semakin besar *compression rate %* , maka akan kecil nilai k-value maka juga akan semakin ter-kompres gambar.



Jika user sudah memasukan gambar, maka gambar akan di-*preview* dulu di home page ini sebelum user menekan tombol *Compress*.

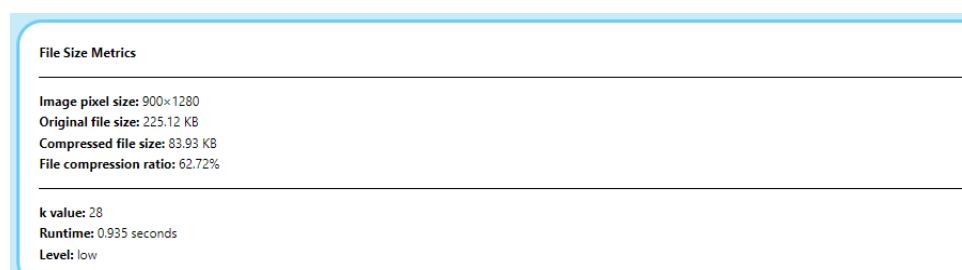
b. Uji Coba 1 : *Compression Rate 25%*



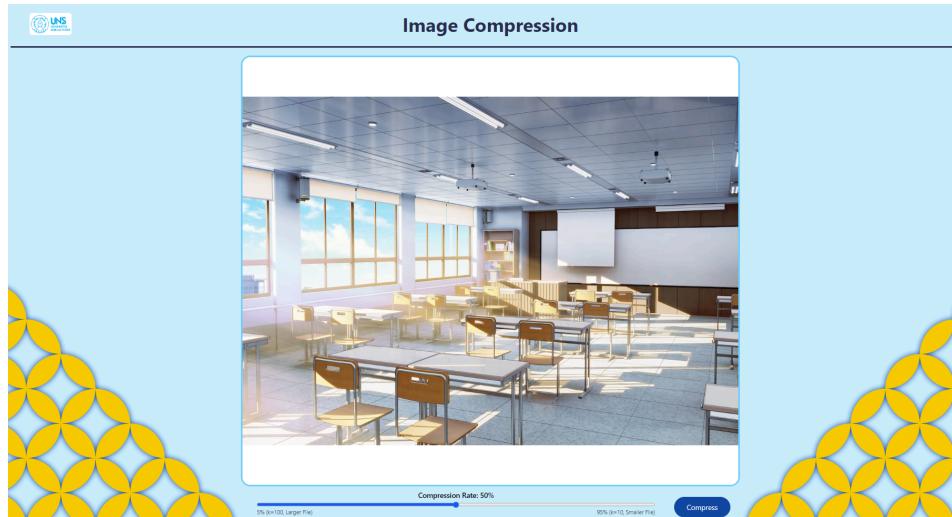
Uji coba pertama ini kita akan coba dengan *Compression Rate 25%* dengan nilai k-value = 80. Ukuran asli gambar adalah 225.15 kb dengan resolusi 900 x 1280.



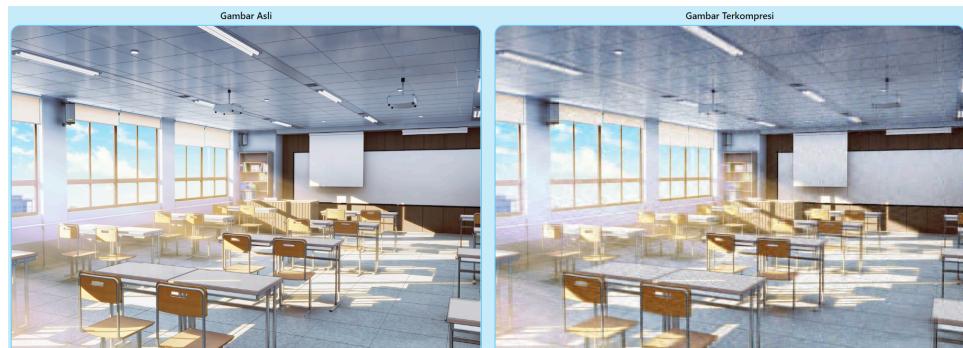
Bisa dilihat, untuk *compression rate 25%* tidak terlalu terlihat untuk pengurangan kualitas gambarnya, akan tetapi terlihat jelas untuk pengurangan ukuran file sedikit dari ukuran asli



c. Uji Coba 2 : *Compression Rate 50%*



Pada uji coba kedua ini kita akan coba dengan *Compression Rate 50%* dengan nilai k-value = 55. Ukuran asli gambar tetap sama yaitu 225.15 kb dengan resolusi 900 x 1280.



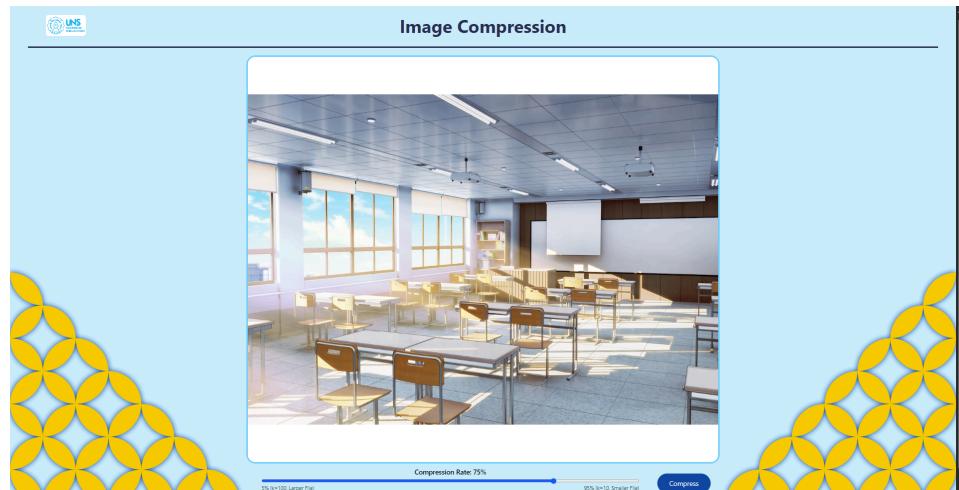
Bisa dilihat, untuk *compression rate 50%* sudah lumayan terlihat untuk pengurangan kualitas gambarnya, dan juga terlihat jelas untuk pengurangan ukuran file sedikit dari ukuran asli

File Size Metrics

Image pixel size: 900x1280
Original file size: 225.12 KB
Compressed file size: 92.06 KB
File compression ratio: 59.11%

k value: 55
Runtime: 0.920 (cached) seconds
Level: low

d. Uji Coba 3 : *Compression Rate 75%*



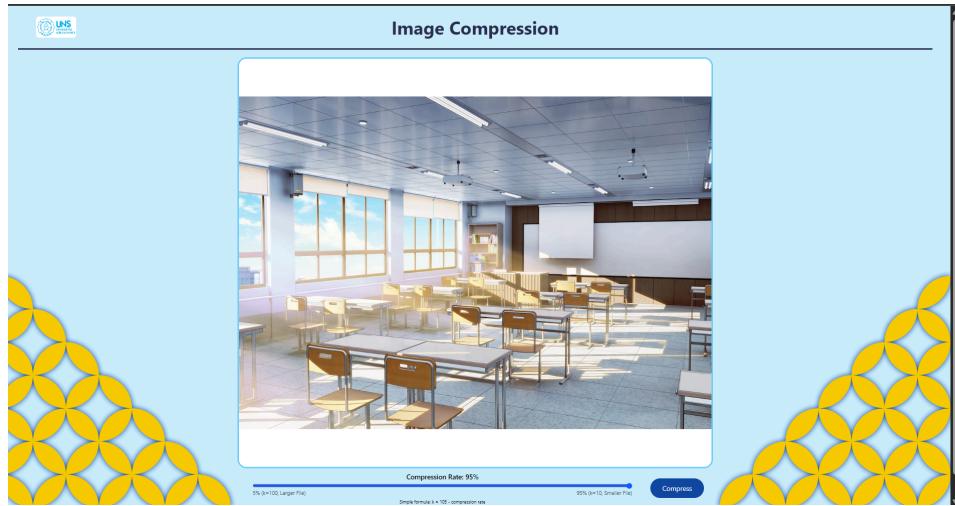
Pada uji coba ketiga ini, sudah cukup besar untuk *compression rate* 75% dengan nilai k-value = 30, ini sudah cukup membuat gambar seolah-olah terdistorsi akibat hasil kompresi yang sudah cukup tinggi.



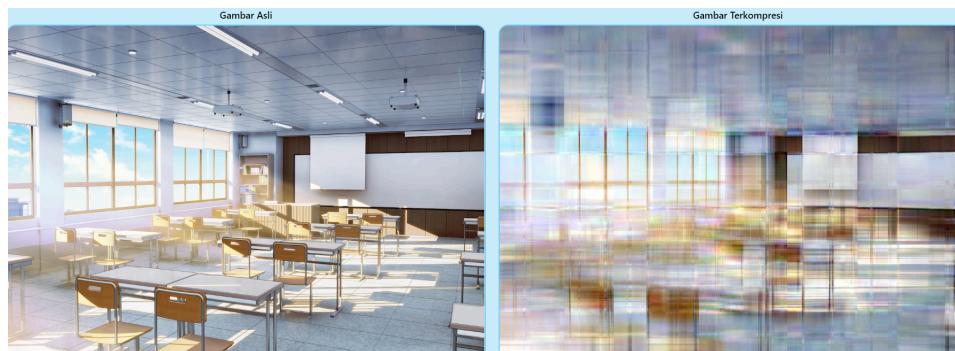
Bisa dilihat, pada gambar kompresi 75% (kanan), sudah terlihat distorsi pixelnya terlihat buram, karena nilai k-value yang semakin kecil akan mengakibatkan gambar semakin buram, tetapi juga ukuran akan semakin mengecil.

File Size Metrics	
Image pixel size:	900x1280
Original file size:	225.12 KB
Compressed file size:	84.56 KB
File compression ratio:	62.44%
k value:	30
Runtime:	0.879 seconds
Level:	low

e. Uji Coba 4 : *Compression Rate 95%*



Pada percobaan terakhir ini, kita akan mencoba kompresi paling tinggi di program ini yaitu 95%, ini sudah membuat gambar tidak jelas karena dengan nilai k-value yang sangat kecil yaitu $k = 10$.



Bisa dilihat dengan jelas yaitu gambar yang sudah di kompresi (kanan) begitu sangat buram dan tidak jelas, terdistorsi sangat tinggi karena nilai $k = 10$ yang berarti sangat kecil. Ukuran file juga akan berkurang drastis.

File Size Metrics	
Image pixel size:	900x1280
Original file size:	225.12 KB
Compressed file size:	70.28 KB
File compression ratio:	68.78%
k value:	10
Runtime:	1.063 seconds
Level:	low

4.2. Kesimpulan Eksperimen

a. Kesimpulan

Pada program ini, jika nilai compression rate % semakin tinggi -> nilai k-value semakin kecil -> gambar semakin buram dan ukuran file semakin kecil.

b. Hasil Uji Coba lain

- *Compression Rate 85%*



File Size Metrics		
Image pixel size:	900x1280	
Original file size:	175.00 KB	
Compressed file size:	74.36 KB	
File compression ratio:	57.51%	
k value:	20	
Runtime:	0.952 seconds	
Level:	low	

- *Compression Rate 65%*



File Size Metrics		
Image pixel size:	900x1280	
Original file size:	230.46 KB	
Compressed file size:	73.60 KB	
File compression ratio:	68.06%	
k value:	40	
Runtime:	0.957 seconds	
Level:	low	

BAB V

KESIMPULAN DAN SARAN

5.1. Kesimpulan

Dalam makalah ini tim penulis dapat mengimplementasikan algoritma Singular Value Decomposition (SVD) sebagai metode kompresi gambar dalam bentuk web. Projek ini menunjukkan bahwa SVD sangat efektif dalam mengurangi ukuran file gambar tanpa mengorbankan kualitas secara signifikan. Proyek ini mengintegrasikan berbagai kakas seperti Flask, NumPy, Pillow, dan Tailwind CSS dengan baik. Selain itu, sistem deployment menggunakan GitHub Actions dan Azure Web App menunjukkan kemampuan untuk meng-online-kan aplikasi. Program ini, mencakup alur upload program, pemrosesan kompresi, hingga tampilan hasil. Hasilnya membuktikan bahwa pendekatan berbasis aljabar linear seperti SVD dapat diaplikasikan secara langsung dalam bidang teknologi informasi. Aplikasi ini dibuat mampu mengolah berbagai format gambar umum dengan kecepatan cukup baik. Tampilan antarmuka pun dibuat responsif dan sederhana agar user experience tetap optimal. Program ini membuktikan manfaat dari teori aljabar linear dalam komputasi nyata. Selain itu, juga menekankan pentingnya keterhubungan antara matematika, logika pemrograman, dan desain aplikasi modern. Terakhir, melalui projek ini, tim penulis memperoleh pemahaman tentang peran algoritma dalam efisiensi penyimpanan data visual.

5.2. Saran

Dalam projek ini, tim penulis menyadari masih banyak kekurangan, namun terdapat saran yang akan dituliskan untuk pengembangan projek di masa kini dan masa depan, antara lain: menambahkan fitur visualisasi perbandingan antara gambar asli. Program juga dapat dilengkapi dengan indikator kualitas kompresi berbasis metrik seperti PSNR (Peak Signal-to-Noise Ratio) atau SSIM (Structural Similarity Index). Selain itu, disarankan untuk menambahkan dukungan kompresi untuk gambar berwarna secara menyeluruh, yaitu dengan memisahkan dan mengompresi kanal RGB

secara paralel. Akan sangat berguna juga jika pengguna bisa memilih sendiri tingkat kompresi melalui slider interaktif, bukan hanya input angka manual. Dokumentasi kode juga sebaiknya ditingkatkan agar pengembang lain dapat lebih mudah memahami struktur proyek. Penambahan fitur preview hasil kompresi secara real-time juga akan meningkatkan nilai guna program ini.

5.3. Refleksi

Melalui projek ini, tim penulis menyadari bahwa menggabungkan konsep aljabar linear seperti dekomposisi matriks dengan pengembangan aplikasi nyata adalah hal yang menantang namun sangat memberikan pengalaman. Tim penulis belajar bahwa teori yang selama ini hanya dipahami, tetapi sangat aplikatif jika dikaitkan dengan kehidupan nyata. Pengalaman membangun aplikasi dari sisi back-end hingga tampilan front-end membuka wawasan baru bahwa penguasaan algoritma saja tidak cukup tanpa didukung pemahaman tools pendukung. Tim penulis juga belajar pentingnya merancang antarmuka yang user-friendly agar dapat dinikmati oleh pengguna dengan baik. Tidak hanya memahami materi tetapi juga cara mengelola waktu, pembagian tugas, dan diskusi tim. Refleksi ini juga membuat saya lebih menghargai pentingnya keterpaduan antara teori, coding, dan komunikasi tim. Tim penulis merasa bangga telah menyelesaikan projek ini dengan sangat baik walaupun masih terdapat kendala dan jauh dari kata sempurna. Dalam projek ini tim penulis merasa bahwa bukti nyata dari belajar tidak hanya tentang nilai, tetapi juga kemampuan untuk menciptakan sesuatu yang berguna dan dapat diaplikasikan secara nyata yang memberikan dampak positif.

DAFTAR PUSTAKA

- Apa Itu npm (Node Package Manager): Panduan untuk Pemula.* (2023, April 19). Hostinger. Diakses pada 24 Juni 2025, dari <https://www.hostinger.com/id/tutorial/apa-itu-npm>
- Jain, S. (2025, April 8). *Tailwind CSS Tutorial*. GeeksforGeeks. Diakses pada 24 Juni 2025, dari <https://www.geeksforgeeks.org/css/tailwind-css/>
- Singular Value Decomposition (SVD)*. (n.d.). Informatika. Diakses pada 17 Juni 2025, dari <https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-19b-Singular-value-decomposition.pdf>
- Wasnik, A. (2020, November 30). *Singular Value Decomposition (SVD) in Python*. AskPython. Diakses pada 17 June 2025, dari <https://www.askpython.com/python/examples/singular-value-decomposition>