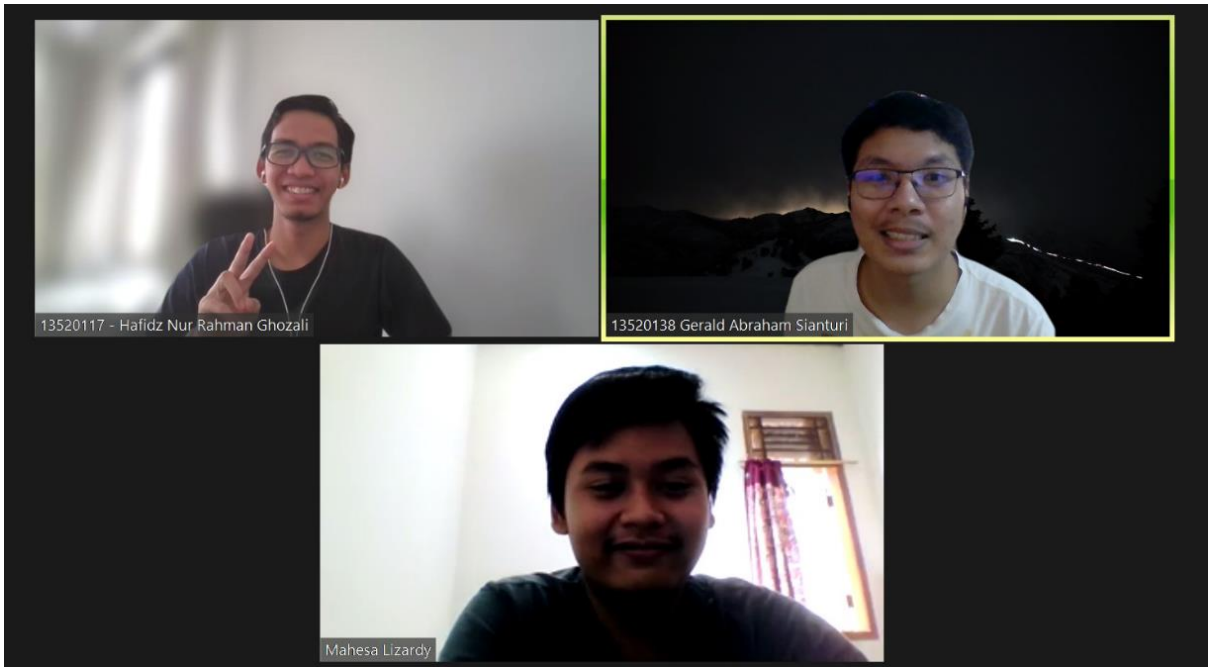
Tugas Besar 2 IF 2123 Aljabar Linier dan Geometri

Aplikasi Nilai Eigen dan Vektor Eigen dalam Kompresi Gambar

Semester I Tahun 2021/2022



KELOMPOK 3Maskethir

13520116 Mahesa Lizardy

13520117 Hafidz Nur Rahman Ghozali

13520138 Gerald Abraham Sianturi

BAB 1

DESKRIPSI MASALAH

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 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.

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 Tugas Besar kali ini, akan dibuat website kompresi gambar sederhana dengan menggunakan algoritma SVD.

BAB 2

TEORI SINGKAT

2.1 Matriks

Matriks adalah kumpulan bilangan yang disusun secara baris atau kolom. Baris adalah susunan bilangan-bilangan yang mendatar (horizontal), sedangkan kolom adalah susunan bilangan-bilangan yang tegak (vertikal). Ordo matriks adalah banyaknya elemen baris dan banyaknya elemen kolom dari suatu matriks. Matriks A berukuran $m\ x\ n$ dapat digambarkan sebagai berikut

$$\mathrm{A} = [a_{ij}] = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

jika $m = n$ maka dinamakan matriks bujursangkar orde n di dalam matriks terdapat beberapa operasi-operasi matriks yang dapat diterapkan seperti penjumlahan, perkalian, transpose dll.

2.2 Operasi Matriks

A. Penjumlahan Matriks

Jika terdapat dua matriks, misal matriks A dan B, yang memiliki orde sama. Jumlah matriks A dan matriks B dapat dinyatakan dengan $c_{ij} = a_{ij} + b_{ij}$, dimana a_{ij} merupakan elemen dari matriks A dan b_{ij} merupakan elemen matriks B yang terletak pada kolom dan baris yang sama, sedangkan c_{ij} merupakan elemen matriks C hasil penjumlahan A dan B yang memiliki orde yang sama

B. Perkalian Matriks

Perkalian dua buah matriks A dan B memiliki syarat jika banyaknya kolom matriks A sama dengan banyaknya baris matriks B. Misal perkalian dua buah matriks $C_{m\times n} = A_{m\times r} \times B_{r\times n}$. Misal $A = [a_{ij}]$ dan $B = [b_{ij}]$. maka

$$C = A \times B = \left[c_{ij}\right], cij = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj}$$

C. Kombinasi linier Matriks

Perkalian matriks dapat dipandang sebagai kombinasi linier, misalkan

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \quad \text{and} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

$$A\mathbf{x} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \quad\quad \vdots \quad\quad\quad\quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{bmatrix} = x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix}$$

D. Transpose Matriks

Transpose matriks adalah matriks baru yang diperoleh dengan cara menukar elemen-elemen baris menjadi elemen kolom atau sebaliknya.

2.3 Nilai Eigen dan Vektor Eigen

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

$$A\boldsymbol{x} = \lambda\boldsymbol{x}$$

Skalar λ disebut nilai eigen dari A, dan x dinamakan vektor eigen yang berkoresponden dengan λ. Nilai eigen menyatakan nilai karakteristik dari sebuah matriks yang berukuran n x n. Sedangkan, Vektor eigen x menyatakan vektor kolom yang apabila dikalikan dengan sebuah matriks n x n menghasilkan vektor lain yang merupakan kelipatan vektor itu sendiri. operasi $Ax = \lambda x$menyebablam vector x menyusut atau memanjang denfan factor λ denggan arah yang sama atau berkebalikan. vector eigen dan nilai eigen dapat dihitung dengan mencari $det(\lambda I - A) = 0$ disebut persamaan karakteristik dari matriks A, dan akar-akar persamaan tersebut, yaitu λ, dinamakan akar-akar karakteristik atau nilai-nilai eigen

2.4 Matriks Ortogonal

Matriks ortogonal adalah matriks yang kolom-kolomnya adalah vector yang saling orthogonal satu sama lain (hasil kali titik sama dengan 0). Jika vektor-vektor kolom tersebut merupakan vektor satuan, maka matriks ortogonal tersebut dinamakan juga matriks ortonormal. Vektor satuan adalah vektor yang dinormalisasi dengan panjang atau magnitude-nya sehingga memiliki panjang atau magnitude = 1. Jika Q adalah matriks ortogonal m x n, dan kolom-kolom matriks Q adalah vektor-vektor satuan $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m$, maka $\boldsymbol{v}_i \,.\, \boldsymbol{v}_j = 0$ untuk $i \neq j$. Atau, dapat juga dikatakan bahwa Q adalah matriks ortogonal jika $Q^T Q = I$, dalam hal ini I merupakan matriks Identitas

2.5 Nilai Singular Matriks

Jika terdapat Matriks A berukuran m x n sehingga $\lambda_1, \lambda_2, \ldots, \lambda_n$ adalah nilai-nilai eigen dari $A^T A$, maka

$$\sigma_1 = \sqrt{\lambda_1}, \sigma_2 = \sqrt{\lambda_2}, \ldots, \sigma_n = \sqrt{\lambda_n}$$

disebut nilai nilai singular dari matriks A. dengan nilai eigen $A^T A$ nonegatif

2.4 Matriks SVD

Singular Value Decomposition (SVD) merupakan salah satu metode mendekomposisi matriks. Matriks bujursangkar A berukuran n x n dapat difaktorkan menjadi

$$A = EDE^{-1}$$

E = Matriks yang kolom-kolomnya adalah basis ruang eigen matriks

$E = (e_1|e_2| \dots |e_n)$

D = matriks diagonal

$D = E^{-1}$

Sedangkan untuk pemfaktoran matriks non-bujursangkar dapat menggunakan metode SVD. Jika terdapat matriks A berukuran m x n dapat difaktorkan menjadi matriks U, Σ, dan V sedimikian sehingga

$$A = U\Sigma V^T$$

U = matriks orthogonal m x m

V = matriks orthogonal n x n

Σ = matriks berukuran m x n yang elemen-elemen diagonal utamanya

adalah nilai-nilai singular dari matriks A dan elemen-elemen lainnya 0

Langkah-langkah SVD mendekomposisi A_{mxn} menjadi U, Σ, dan V :

1. Untuk vektor singular kiri, hitung nilai-nilai eigen dari AA^T dan nilai-nilai singular dari A.
2. Bentuklah matriks Σ berukuran m x n dengan elemen-elemen diagonalnya adalah nilai-nilai singular dari matriks A dengan susunan dari besar ke kecil.
3. Tentukan vektor-vektor eigen $u_1, u_2, \dots, u_m$ yang berkoresponden dengan nilai-nilai eigen dari AA^T. Normalisasi $u_1, u_2, \dots, u_m$ dengan cara setiap komponen vektornya dibagi dengan panjang vektor. Diperoleh matriks U.
4. Untuk vektor singular kanan, hitung nilai-nilai eigen dari A^TA.
5. Tentukan vektor-vektor eigen $v_1, v_2, \dots, v_n$ yang berkoresponden dengan nilai-nilai eigen dari A^TA. Normalisasi $v_1, v_2, \dots, v_n$ dengan cara setiap komponen vektornya dibagi dengan panjang vektor. Diperoleh matriks V. Transpose-kan matriks V sehingga menjadi V^T.
6. Maka $A = U\Sigma V^T$.

BAB 3

IMPLEMENTASI PROGRAM

Kakas yang digunakan

Pada program kompresi gambar ini, kami menggunakan modul Flask pada Python untuk pembuatan API dan struktur *backend* pada website. Untuk tampilan website, kami memanfaatkan kombinasi antara HTML, CSS, dan Javascript. Selain itu, kami juga menggunakan modul opencv-python dan numpy untuk mengolah gambar dan melakukan kompresi.

Berikut adalah algoritma-algoritma yang kami gunakan dalam program kompresi gambar ini.

1. Algoritma pencarian nilai eigen dan vektor eigen

```python
def simultaneous_power_iteration(A, k):
    # fungsi untuk mencari nilai eigen dan eigen vector dengan metode QR Decomposition
    n, m = A.shape
    Q = np.random.rand(n, k)
    Q, _ = np.linalg.qr(Q)

    for i in range(200):
        Z = A.dot(Q)
        Q, R = np.linalg.qr(Z)

    return np.diag(R), Q
```

Dalam algoritma ini, kami memanfaatkan pemecahan matriks dengan metode QR. Hal ini bertujuan untuk memudahkan pencarian nilai eigen dan vektor eigen. Kami menggunakan perulangan sebanyak dua ratus kali untuk mendapatkan tingkat presisi bagus terhadap nilai-nilai eigen yang sebenarnya. Terdapat dua parameter dalam fungsi *simultaneous power* iteration ini, yaitu A yang merupakan matriks berukuran $n \times m$ dan k yang merupakan banyaknya nilai eigen yang dicari. Nilai eigen yang dicari adalah nilai-nilai eigen yang terbesar. Fungsi ini mengembalikan list nilai eigen dan matriks vektor eigen yang bersesuaian dengan nilai eigennya.

2. Algoritma pemrosesan kompresi gambar

```python
def main(img, persen):
    transparent = False
    # split tiap channel warna
    if (len(img[0][0]) == 4) :
        transparent = True
        b,g,r,a = cv.split(img)
    else:
        b,g,r = cv.split(img)

    # proses setiap channel secara terpisah
    afterB = process(b, persen)
    print("Channel Blue selesai")
    afterG = process(g, persen)
    print("Channel Green selesai")
    afterR = process(r, persen)
    print("Channel Red selesai")

    # menggabungkan kembali hasil pemrosesan tiap channel
    if (transparent):
        after = cv.merge([afterB, afterG, afterR, a])
    else :
        after = cv.merge([afterB, afterG, afterR])

    after = after.astype(np.uint8)
    return after
```

Fungsi ini merupakan inti dari pemrosesan kompresi gambar. Fungsi ini menerima dua parameter, yaitu matriks dari gambar dan persentase kompresi. Pertama, matriks dari gambar dipisah menjadi matriks-matriks terpisah berdasarkan *channel* warna yang ada di dalam matriks. Setelah dipisah, semua matriks tersebut diproses secara individu. Setelah diproses, matriks-matriks tersebut digabungkan kembali menjadi matriks yang merepresentasikan gambar yang sudah dikompresi. Fungsi ini mengembalikan matriks gambar yang sudah dikompresi.

```python
def process(Mat, k):
    U,S,Vt = svd(Mat, k)
    if (min(Mat.shape) > 200):
        k *= 2
    else:
        k = k*min(Mat.shape)//100

    # proses pemotongan matriks
    U = U[:,:k]
    S = S[:k,:k]
    Vt = Vt[:k,:]

    MatRes = U @ S @ Vt
    MatRes = np.clip(MatRes,0,255)
    MatRes = MatRes.astype(np.uint8)

    return MatRes
```

Fungsi ini bertujuan untuk melakukan kompresi gambar. Fungsi ini menerima 2 parameter masukan, yaitu matriks Mat yang berupa matriks salah satu channel warna dan tingkat kompresi (k). Pertama, matriks dipecah menggunakan fungsi SVD. Setelah itu dilakukan pemotongan matriks bergantung pada nilai k yang diberikan. Matriks-matriks tersebut disusun kembali agar menjadi matriks channel yang sudah terkompresi. Fungsi ini mengembalikan matriks yang telah dikompresi.

3. Algoritma pemecahan matriks menggunakan metode SVD

```python
def svd(A):
    A = A.astype('float64')
    m,n = A.shape
    At = np.transpose(A)

    Right = At @ A
    # Mengambil 200 nilai eigen yang paling besar
    EV, V = simultaneous_power_iteration(Right, 200)
    V = np.pad(V,((0,0),(0,n-V.shape[1])))
    # Membuat matriks Sigma
    singularValue = np.sqrt(EV)
    Sigma = np.diag(singularValue)
    Sinv = np.linalg.inv(Sigma)
    Sigma = np.pad(Sigma, ((0,n-Sigma.shape[0]),(0,m-Sigma.shape[1])))
    Sinv = np.pad(Sinv, ((0,n-Sinv.shape[0]),(0,m-Sinv.shape[1])))

    # Mencari matriks U
    U = A @ V @ Sinv

    return U, Sigma, np.transpose(V)
```

Dalam algoritma SVD ini, kami hanya mengambil 200 nilai eigen saja. Hal ini bertujuan untuk memangkas waktu jalannya program. Fungsi ini menerima sebuah parameter berupa matriks berukuran $m \times n$ yang akan dipecah dengan metode SVD. Pertama, kami mencari nilai-nilai eigen dan vektor eigen dari matriks $A^T A$. Nilai eigen yang sudah didapatkan kemudian disusun menjadi matriks singular dan ukurannya disesuaikan. Setelah itu kami mencari matriks U dengan informasi informasi yang sudah ada, yaitu matriks A, V, dan S^{-1}. Fungsi ini mengembalikan matriks U yaitu matriks singular kiri, matriks singular yang berisi diagonalnya berisi nilai nilai singular, dan transpose dari matriks V yang merupakan matriks singular kanan.

BAB 4

EKSPERIMEN

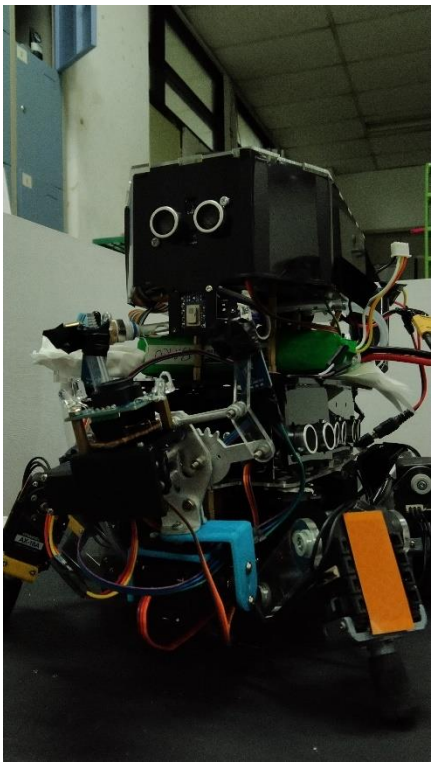
1. Kompresi pada gambar berukuran persegi
   Gambar asli
   Ukuran : $4724\ x\ 4724$ pixels

Gambar hasil kompresi

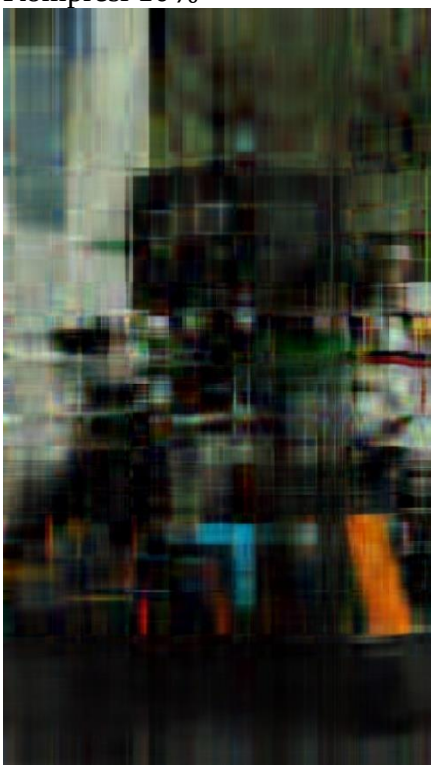
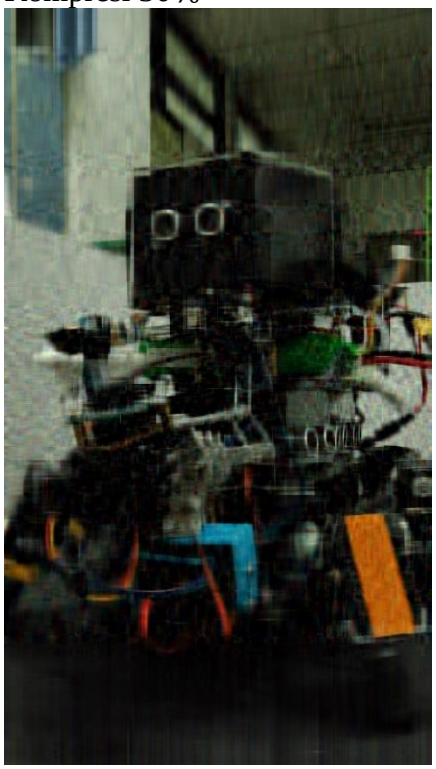
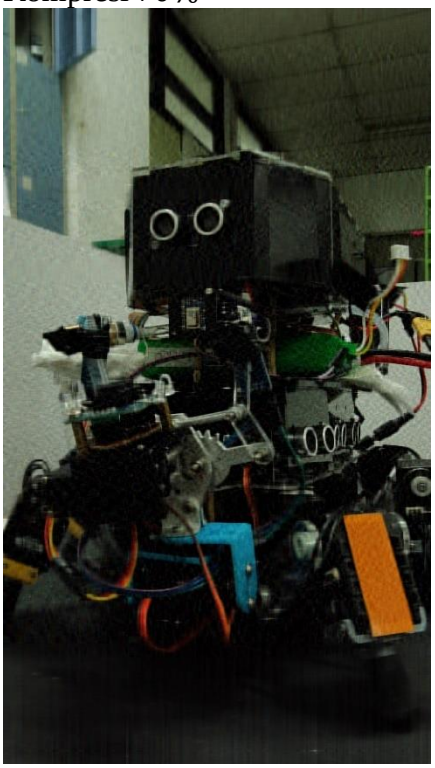
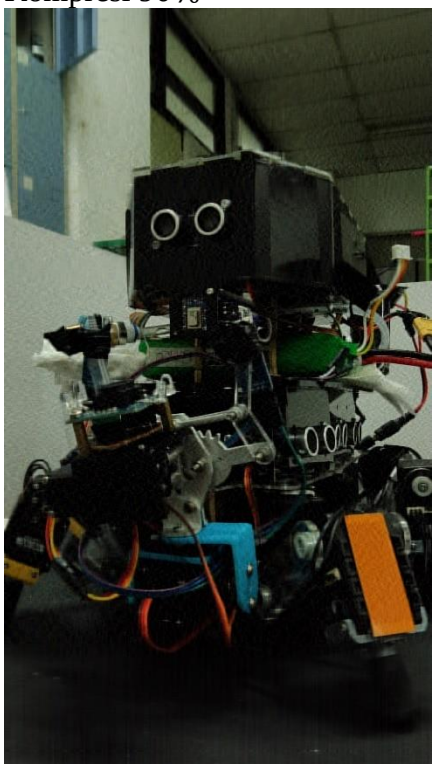
Kompresi 10% 	Kompresi 30% 
Kompresi 70% 	Kompresi 90% 

Ukuran gambar asli	Tingkat Kompresi	Waktu kompresi	Ukuran setelah kompresi
3.50 MB	10%	1 menit 51 detik	1.50 MB
3.50 MB	30%	2 menit 26 detik	1.71 MB
3.50 MB	70%	3 menit 11 detik	1.98 MB
3.50 MB	90%	3 menit 33 detik	2.11 MB

2. Kompresi pada gambar bertipe *potrait*
   Gambar asli
   Ukuran : $2352\ x\ 4160\ pixels$



Gambar Hasil Kompresi

Kompresi 10% 	Kompresi 30% 
Kompresi 70% 	Kompresi 90% 

Ukuran gambar asli	Tingkat Kompresi	Waktu kompresi	Ukuran setelah kompresi
1.91 MB	10%	18 detik	1.19MB
1.91MB	30%	26 detik	1.47MB
1.91MB	70%	42 detik	1.72MB
1.91MB	90%	51 detik	1.79MB

3. Kompresi pada gambar bertipe *landscape*

Gambar Awal
Ukuran : $3000\ x\ 1700$ pixels

Gambar Hasil Kompresi

Kompresi 10%	Kompresi 30%
	

Kompresi 70%	Kompresi 90%
	

4. Kompresi pada gambar transparan

   Gambar Asli

   Ukuran : 1920 x 1440 pixels

Kompresi 10%	Kompresi 30%

	
Kompresi 70% 	Kompresi 90% 

Ukuran asli	Tingkat kompresi	Waktu Kompresi	Ukuran setelah kompresi
2.01 MB	10%	12 detik	1.63 MB
2.01 MB	30%	25 detik	1.92 MB
2.01 MB	70%	45 detik	2.23 MB
2.01 MB	90%	63 detik	2.35 MB

BAB 5

KESIMPULAN

5.1 KESIMPULAN

Metode *Singular Value Decomposition* (SVD) telah terbukti memiliki banyak aplikasi dalam bidang sains dan teknik. Dalam *paper* ini, penulis menggunakan metode ini untuk melakukan pemrosesan gambar statis, khususnya melakukan pengompresian gambar. Format gambar yang dapat diolah melalui fungsi-fungsi yang dibentuk dibatasi pada format .jpg, .png, dan .jpeg. Melalui percobaan yang didapatkan, diperoleh gambar hasil kompresi dengan tingkat kompresi 10%, 30%, 70%, dan 90%. Pada tiga kasus percobaan, semakin kecil tingkat kompresi, semakin besar waktu yang diperlukan untuk kompresi. Diperoleh juga bahwa secara umum, ukuran file gambar hasil kompresi mengalami penurunan. Namun, terdapat anomali pada percobaan keempat, yakni file kompresi mengalami kenaikan ukuran file.

5.2 SARAN

Pada pengerjaan fungsi-fungsi yang ada, ada hal yang tidak dalam perhatian kami yang menyebabkan anomali dalam ukuran file hasil kompresi. Untuk pihak yang hendak menciptakan fungsi dengan fungsionalitas yang sama, kami sarankan untuk melakukan analisa kembali terhadap anomali tersebut.

5.3 REFLEKSI

Melalui tugas besar ini, kami memperoleh lebih banyak ilmu, mulai dari segi alur berpikir menyelesaikan permasalahan matematis, hingga penambahan pemahaman terkait penggunaan html, js, dan penggunaan library tertentu untuk pengembangan suatu website. Kami juga semakin terbiasa bekerja sama dalam tim, dengan menyamakan persepsi terhadap suatu permasalahan yang ada.

DAFTAR REFERENSI

[1] "Power Iteration - ML Wiki." http://mlwiki.org/index.php/Power_Iteration.

[2] "List of numerical analysis topics - Wikipedia." https://en.wikipedia.org/wiki/List_of_numerical_analysis_topics.

[3] "A Very Fast Algorithm for Finding ... - Computer Science." https://www.cs.unc.edu/techreports/96-043.pdf.

[4] "Inverse Power Method Shifted Power Method ... - The Citadel." http://www.macs.citadel.edu/chenm/344.dir/08.dir/lect4_2.pdf.

[5] "Layout · Bootstrap v5.1." https://getbootstrap.com/docs/5.1/forms/layout/.

[5] “Slide Bahan Ajar Mata Kuliah IF2123 Aljabar Linier dan Geometri oleh Rinaldi Munir” https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2021-2022/algeo21-22.htm