

LAPORAN TUGAS KECIL 02
IF2211 STRATEGI ALGORITMA

Kompresi Gambar Dengan Metode Quadtree



Disusun oleh:

Danendra Shafi Athallah

13523136

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
JL. GANESA 10, BANDUNG 40132

2024

BAB I

ALGORITMA DIVIDE AND CONQUER

Algoritma divide and conquer adalah paradigma pemecahan masalah yang memecah persoalan menjadi beberapa bagian kecil (divide), menyelesaikan masing-masing bagian secara independen (conquer), kemudian menggabungkan solusi dari bagian-bagian tersebut menjadi solusi persoalan awal (combine). Pendekatan ini sangat efektif untuk menyelesaikan berbagai jenis permasalahan komputasi yang kompleks, termasuk kompresi gambar yang menjadi fokus tugas kecil ini.

Dalam implementasi teknis, sebuah Quadtree direpresentasikan sebagai simpul (*node*) dengan maksimal empat anak (*children*). Simpul daun (*leaf*) merepresentasikan area gambar yang seragam, sementara simpul internal menunjukkan area yang masih membutuhkan pembagian lebih lanjut. Setiap simpul menyimpan informasi seperti posisi (x, y), ukuran (width, height), dan nilai rata-rata warna atau intensitas piksel dalam area tersebut. Struktur ini memungkinkan pengkodean data gambar yang lebih efisien dengan menghilangkan redundansi pada area yang seragam. QuadTree sering digunakan dalam algoritma kompresi lossy karena mampu mengurangi ukuran file secara signifikan tanpa mengorbankan detail penting pada gambar. Dalam konteks kompresi gambar dengan metode quadtree, algoritma divide and conquer diterapkan sebagai berikut:

- Divide: Gambar dibagi menjadi empat bagian (kuadran) yang sama ukurannya. Proses ini dilakukan secara rekursif, sehingga setiap kuadran dapat dibagi lagi menjadi empat kuadran yang lebih kecil.
- Conquer: Untuk setiap kuadran, dihitung nilai variansi atau error berdasarkan metode yang dipilih (Variance, MAD, Max Pixel Difference, Entropy, atau SSIM).
- Combine: Berdasarkan nilai error yang dihitung yaitu jika error kurang dari threshold yang ditentukan, kuadran tersebut akan direpresentasikan dengan satu warna rata-rata. Sedangkan bila error melebihi threshold dan ukuran kuadran masih lebih besar dari ukuran blok minimum, kuadran tersebut akan dibagi lagi menjadi empat bagian dan proses dilanjutkan secara rekursif.

Spesifikasi wajib dalam tugas ini meliputi implementasi algoritma divide and conquer untuk kompresi gambar berbasis quadtree, implementasi metode perhitungan error (Variance, MAD, Max Pixel Difference, Entropy), pembagian blok secara rekursif berdasarkan threshold dan ukuran minimum, normalisasi warna untuk blok yang tidak lagi dibagi, serta rekonstruksi gambar dari struktur quadtree. Hasil yang ditampilkan berupa gambar kompresi, persentase kompresi, waktu eksekusi, ukuran gambar sebelum dan sesudah, kedalaman pohon, dan jumlah simpul. Terdapat pula spesifikasi bonus yang dapat diimplementasikan yaitu ada mode persentase kompresi otomatis yang menyesuaikan threshold untuk mencapai target kompresi tertentu, implementasi *Structural Similarity Index* (SSIM) sebagai metode pengukuran error tambahan, dan visualisasi GIF proses kompresi yang menunjukkan pembagian gambar secara dinamis. Proses ini menghasilkan struktur *quadtree*, dimana setiap node merepresentasikan sebuah kuadran dalam gambar. Node leaf merepresentasikan area yang dianggap homogen sehingga dapat diwakili oleh satu warna. Dengan pendekatan ini, area gambar yang homogen direpresentasikan lebih efisien dengan sedikit node, sementara area dengan detail tinggi memiliki lebih banyak node. Algoritma kompresi gambar dengan quadtree dapat dijabarkan dalam langkah-langkah berikut:

1. Inisialisasi: Masukkan gambar sebagai matriks piksel RGB dengan parameter metode variansi, threshold, ukuran blok minimum, dan target kompresi.
2. Pembagian Blok: Dari gambar keseluruhan, periksa error blok terhadap threshold. Jika melebihi threshold dan ukuran memungkinkan untuk dibagi, bagi menjadi empat sub-blok sama besar.
3. Perhitungan Error: Hitung nilai error setiap blok menggunakan metode yang dipilih.
4. Keputusan Pembagian: Bandingkan error dengan threshold. Jika error di atas threshold dan ukuran masih memadai, bagi menjadi empat. Jika tidak, representasikan dengan warna rata-rata.

5. Penggabungan (Combine): Rekonstruksi gambar berdasarkan struktur quadtree, dimana setiap leaf node memberikan warna pada area yang diwakilinya.

BAB II

SOURCE CODE PROGRAM

2.1. Library

Dalam pengembangan program kompresi gambar dengan metode quadtree, beberapa library digunakan untuk mendukung berbagai fungsionalitas yang diperlukan:

- **opencv2/opencv.hpp**: Library utama untuk pemrosesan gambar yang menyediakan fungsi-fungsi untuk membaca, menulis, dan memanipulasi gambar, termasuk operasi pada piksel, transformasi gambar, dan I/O format gambar.
- **iostream, string, iomanip**: Library standar C++ untuk operasi input/output, manipulasi string, dan pemformatan output yang digunakan untuk interaksi dengan pengguna dan tampilan hasil.
- **chrono, thread**: Library untuk pengukuran waktu dan penanganan thread. Digunakan untuk mengukur performa algoritma, menerapkan batas waktu eksekusi, dan sinkronisasi proses.
- **limits, filesystem**: Library untuk mengakses nilai-nilai batas dan memanipulasi sistem file. Filesystem (C++17) digunakan untuk validasi path, cek eksistensi file, dan operasi direktori.
- **cmath, vector, algorithm**: Library matematika, struktur data vektor, dan algoritma standar yang digunakan untuk perhitungan dalam algoritma kompresi.
- **map, set, future, atomic**: Struktur data dan primitive sinkronisasi untuk pengelolaan data dan komputasi paralel.
- **fstream, sstream**: Library untuk operasi file stream dan string stream, digunakan untuk pembacaan/penulisan file dan manipulasi string.
- **random**: Library untuk menghasilkan bilangan acak, digunakan dalam beberapa operasi optimasi.
- **FFmpeg (eksternal)**: Program eksternal yang dipanggil melalui sistem untuk pembuatan animasi GIF dari rangkaian frame. Library ini tidak langsung di-include tetapi dieksekusi melalui system call.
- **Windows.h (khusus Windows)**: API Windows untuk fungsionalitas spesifik platform, termasuk pengaturan konsol untuk tampilan warna.
- **termios.h (khusus UNIX/Linux)**: API UNIX/Linux untuk pengaturan terminal, digunakan untuk implementasi input non-blocking.
- **interface.hpp, Quadtree.hpp**: Header file kustom yang mendefinisikan antarmuka pengguna dan implementasi struktur data quadtree, mengenkapsulasi fungsionalitas inti program.

Program ini juga memanfaatkan beberapa fitur C++17 seperti filesystem, structured bindings, dan inline variables untuk implementasi yang lebih efisien dan mudah dibaca.

2.2. Struktur Program

Program dibuat ke dalam beberapa class dan file utama:

1. **Quadtree.hpp** dan **Quadtree.cpp**: Mendefinisikan struktur data dan algoritma quadtree untuk kompresi gambar. Class utama yaitu:

- QuadtreeNode: Merepresentasikan node dalam struktur quadtree, menyimpan informasi posisi, ukuran, dan warna rata-rata.
 - Quadtree: Mengelola operasi-operasi pada struktur quadtree, termasuk pembentukan, kompresi, dan rekonstruksi gambar.
2. **interface.hpp**: Menyediakan antarmuka pengguna dengan tampilan visual informatif, termasuk progress bar, dialog interaktif, rekomendasi, tampilan status, dan tabel hasil statistik kompresi. Komponen ini dirancang untuk memberikan pengalaman pengguna yang menyenangkan dan informatif.
 3. **main.cpp**: Titik masuk program, menangani interaksi dengan pengguna, validasi input, dan menjalankan proses kompresi.

Program menggunakan pendekatan berorientasi objek dengan pemisahan yang jelas antara logika aplikasi (Quadtree), antarmuka pengguna (QuadtreeInterface), dan koordinasi program keseluruhan (main). Struktur ini memungkinkan pengembangan dan pemeliharaan modular pada komponen-komponen yang berbeda.

2.3. Program Utama

Berikut ini adalah potongan dalam pseudocode untuk algoritma utama dalam program kompresi gambar menggunakan quadtree. Algoritma ini mengimplementasikan pendekatan divide and conquer untuk membagi gambar menjadi blok-blok sesuai dengan tingkat homogenitas warnanya.

2.3.1. Pseudocode Quadtree

```

1. FUNCTION ConstructQuadtree(image, threshold, minBlockSize,
   method)
2.     root ← CreateNode(0, 0, image.width, image.height)
3.     DivideNode(image, root, threshold, minBlockSize, method)
4.     RETURN root
5. END FUNCTION
6.
7. FUNCTION DivideNode(image, node, threshold, minBlockSize,
   method)
8.     IF node.width ≤ minBlockSize OR node.height ≤ minBlockSize
     THEN
9.         node.calculateAverageColor(image)
10.        node.isLeaf ← true
11.        RETURN
12.    END IF
13.
14.    error ← CalculateError(image, node, method)
15.    IF error < threshold THEN
16.        node.calculateAverageColor(image)
17.        node.isLeaf ← true
18.    RETURN

```

```

19.      END IF
20.
21.      halfWidth ← node.width / 2
22.      halfHeight ← node.height / 2
23.      node.isLeaf ← false
24.
25.      node.children[0] ← CreateNode(node.x, node.y, halfWidth,
   halfHeight)
26.      node.children[1] ← CreateNode(node.x + halfWidth, node.y,
   node.width - halfWidth, halfHeight)
27.      node.children[2] ← CreateNode(node.x, node.y +
   halfHeight, halfWidth, node.height - halfHeight)
28.      node.children[3] ← CreateNode(node.x + halfWidth, node.y
   + halfHeight, node.width - halfWidth, node.height - halfHeight)
29.
30.      FOR i ← 0 TO 3 DO
31.          DivideNode(image, node.children[i], threshold,
   minBlockSize, method)
32.      END FOR
33.  END FUNCTION
34.
35.  FUNCTION CalculateError(image, node, method)
36.      region ← image.getRegion(node.x, node.y, node.width,
   node.height)
37.      SWITCH method
38.          CASE VARIANCE:
39.              RETURN CalculateVariance(region)
40.          CASE MAD:
41.              RETURN CalculateMAD(region)
42.          CASE MAX_PIXEL_DIFF:
43.              RETURN CalculateMaxPixelDiff(region)
44.          CASE ENTROPY:
45.              RETURN CalculateEntropy(region)
46.          CASE SSIM:
47.              averageColor ← CalculateAverageColor(region)
48.              uniformRegion ← CreateUniformRegion(averageColor,
   region.width, region.height)
49.              RETURN 1 - CalculateSSIM(region, uniformRegion)
50.      END SWITCH
51.  END FUNCTION

```

Algoritma constructQuadtree membuat struktur pohon quadtree dengan membagi gambar secara rekursif menjadi empat bagian. Pembagian dilakukan berdasarkan nilai error yang dihitung dengan metode yang dipilih pengguna. Proses pembagian akan berhenti jika salah satu dari dua kondisi terpenuhi yaitu ukuran blok mencapai ukuran minimum yang ditentukan pengguna atau nilai error sudah di bawah threshold yang ditentukan. Setiap node daun (leaf) menyimpan warna rata-rata dari region yang diwakilinya, yang akan digunakan untuk merekonstruksi gambar hasil kompresi.

2.3.2. Pseudocode Metode Perhitungan Error

```
1. FUNCTION CalculateVariance(region)
2.     meanR, meanG, meanB ← CalculateMeanColors(region)
3.     varR, varG, varB ← 0, 0, 0
4.     pixelCount ← region.width * region.height
5.
6.     FOR EACH pixel IN region DO
7.         varR ← varR + (pixel.R - meanR)^2
8.         varG ← varG + (pixel.G - meanG)^2
9.         varB ← varB + (pixel.B - meanB)^2
10.    END FOR
11.
12.    varR ← varR / pixelCount
13.    varG ← varG / pixelCount
14.    varB ← varB / pixelCount
15.
16.    RETURN (varR + varG + varB) / 3
17. END FUNCTION
18.
19. FUNCTION CalculateMAD(region)
20.     meanR, meanG, meanB ← CalculateMeanColors(region)
21.     madR, madG, madB ← 0, 0, 0
22.     pixelCount ← region.width * region.height
23.
24.     FOR EACH pixel IN region DO
25.         madR ← madR + |pixel.R - meanR|
26.         madG ← madG + |pixel.G - meanG|
27.         madB ← madB + |pixel.B - meanB|
28.     END FOR
29.
30.     madR ← madR / pixelCount
31.     madG ← madG / pixelCount
32.     madB ← madB / pixelCount
33.
```

```

34.      RETURN (madR + madG + madB) / 3
35. END FUNCTION
36.
37. FUNCTION CalculateMaxPixelDiff(region)
38.     minR, minG, minB ← 255, 255, 255
39.     maxR, maxG, maxB ← 0, 0, 0
40.
41.     FOR EACH pixel IN region DO
42.         minR ← MIN(minR, pixel.R)
43.         minG ← MIN(minG, pixel.G)
44.         minB ← MIN(minB, pixel.B)
45.
46.         maxR ← MAX(maxR, pixel.R)
47.         maxG ← MAX(maxG, pixel.G)
48.         maxB ← MAX(maxB, pixel.B)
49.     END FOR
50.
51.     diffR ← maxR - minR
52.     diffG ← maxG - minG
53.     diffB ← maxB - minB
54.
55.     RETURN (diffR + diffG + diffB) / 3
56. END FUNCTION
57.
58. FUNCTION CalculateEntropy(region)
59.     histR, histG, histB ← CreateHistograms(region)
60.     entropyR, entropyG, entropyB ← 0, 0, 0
61.     pixelCount ← region.width * region.height
62.
63.     FOR i ← 0 TO 255 DO
64.         IF histR[i] > 0 THEN
65.             probR ← histR[i] / pixelCount
66.             entropyR ← entropyR - (probR * LOG2(probR))
67.         END IF
68.
69.         IF histG[i] > 0 THEN
70.             probG ← histG[i] / pixelCount
71.             entropyG ← entropyG - (probG * LOG2(probG))
72.         END IF
73.
74.         IF histB[i] > 0 THEN

```

```

75.           probB ← histB[i] / pixelCount
76.           entropyB ← entropyB - (probB * LOG2(probB))
77.       END IF
78.   END FOR
79.
80.   RETURN (entropyR + entropyG + entropyB) / 3
81. END FUNCTION
82.
83. FUNCTION CalculateSSIM(region1, region2)
84.     C1 ← (0.01 * 255)^2
85.     C2 ← (0.03 * 255)^2
86.     wR ← 0.299
87.     wG ← 0.587
88.     wB ← 0.114
89.
90.     mean1R, mean1G, mean1B ← CalculateMeanColors(region1)
91.     mean2R, mean2G, mean2B ← CalculateMeanColors(region2)
92.     var1R, var1G, var1B ←
         CalculateVariancePerChannel(region1)
93.     var2R, var2G, var2B ←
         CalculateVariancePerChannel(region2)
94.     covRGB ← CalculateCovariance(region1, region2)
95.
96.     ssimR ← ((2 * mean1R * mean2R + C1) * (2 * covRGB.R +
    C2)) /
97.                 ((mean1R^2 + mean2R^2 + C1) * (var1R + var2R +
    C2))
98.
99.     ssimG ← ((2 * mean1G * mean2G + C1) * (2 * covRGB.G +
    C2)) /
100.                ((mean1G^2 + mean2G^2 + C1) * (var1G + var2G +
    C2))
101.
102.     ssimB ← ((2 * mean1B * mean2B + C1) * (2 * covRGB.B +
    C2)) /
103.                ((mean1B^2 + mean2B^2 + C1) * (var1B + var2B +
    C2))
104.
105.     RETURN wR * ssimR + wG * ssimG + wB * ssimB
106. END FUNCTION

```

Fungsi-fungsi ini mengimplementasikan berbagai metode pengukuran error untuk menentukan tingkat homogenitas suatu blok gambar:

1. **Variance**: Menghitung deviasi standar kuadrat dari nilai piksel untuk setiap kanal warna, kemudian dirata-ratakan. Metode ini sensitif terhadap perbedaan intensitas warna yang signifikan.
2. **Mean Absolute Deviation (MAD)**: Menghitung rata-rata perbedaan absolut antara nilai piksel dengan nilai rata-ratanya untuk setiap kanal warna. Pendekatan ini lebih tahan terhadap outlier dibandingkan variance.
3. **Max Pixel Difference**: Menghitung selisih antara nilai piksel maksimum dan minimum untuk setiap kanal warna. Metode ini sangat efektif untuk mendeteksi kontras tinggi dalam suatu blok.
4. **Entropy**: Menghitung entropi informasi dari distribusi nilai piksel dalam suatu blok. Nilai tinggi menunjukkan keacakan atau variasi warna yang tinggi.
5. **SSIM (Bonus)**: Mengukur kemiripan struktural antara blok gambar asli dengan blok rata-rata. Metode ini mempertimbangkan aspek perceptual manusia terhadap kualitas gambar.

2.3.3. Pseudocode Rekonstruksi Gambar

```
1. FUNCTION ReconstructImage(originalImage, root)
2.     resultImage ← CreateEmptyImage(originalImage.width,
   originalImage.height)
3.     ReconstructNode(resultImage, root)
4.     RETURN resultImage
5. END FUNCTION
6.
7. FUNCTION ReconstructNode(image, node)
8.     IF node.isLeaf THEN
9.         FillRegion(image, node.x, node.y, node.width,
   node.height, node.avgColor)
10.    ELSE
11.        FOR i ← 0 TO 3 DO
12.            ReconstructNode(image, node.children[i])
13.        END FOR
14.    END IF
15. END FUNCTION
```

Fungsi reconstructImage merekonstruksi gambar dari struktur quadtree yang telah dibuat. Untuk setiap node daun (leaf), region yang sesuai pada gambar hasil diisi dengan warna rata-rata yang telah dihitung sebelumnya. Rekonstruksi dilakukan secara rekursif dari root hingga ke semua node daun, menghasilkan gambar terkompresi.

2.3.4. Pseudocode Interface

```
1. FUNCTION Main()
2.     interface ← NEW QuadtreeInterface()
```

```

3.    interface.ShowLogo()
4.    interface.ShowIntro()
5.
6.    interface.ShowSectionHeader("INPUT GAMBAR")
7.    inputImagePath ← GetUserInput("Masukkan path file gambar
   input: ")
8.    ValidateImageFile(inputImagePath)
9.
10.   interface.ShowSectionHeader("METODE PENGUKURAN ERROR")
11.   method ← GetUserMethodChoice()
12.
13.   interface.ShowSectionHeader("NILAI THRESHOLD")
14.   threshold ← GetUserThreshold(method)
15.
16.   interface.ShowSectionHeader("UKURAN BLOK MINIMUM")
17.   minBlockSize ← GetUserBlockSize()
18.
19.   interface.ShowSectionHeader("PERSENTASE KOMPRESI TARGET
   [BONUS]")
20.   targetCompressionPct ← GetUserTargetCompression()
21.
22.   interface.ShowSectionHeader("GAMBAR OUTPUT")
23.   outputImagePath ← GetUserOutput()
24.
25.   interface.ShowSectionHeader("VISUALISASI GIF [BONUS]")
26.   visualizeGif ← GetUserGifChoice()
27.   IF visualizeGif THEN
28.       gifOutputPath ← GetUserGifOutput()
29.   END IF
30.
31.   interface.ShowSectionHeader("PEMROSESAN")
32.   startTime ← GetcurrentTime()
33.
34.   interface.ShowLoading("Membuat quadtree")
35.   quadtree ← NEW Quadtree(image, threshold, minBlockSize,
   method, targetCompressionPct, visualizeGif)
36.
37.   interface.ShowLoading("Mengompresi gambar")
38.   quadtree.CompressImage()
39.
40.   interface.ShowLoading("Merekonstruksi gambar")

```

```

41.     compressedImage ← quadtree.ReconstructImage()
42.
43.     endTime ← GetCurrentTime()
44.     executionTime ← endTime - startTime
45.
46.     SaveCompressedImage(compressedImage, outputPath)
47.
48.     IF visualizeGif THEN
49.         SaveGifAnimation(quadtree, gif outputPath)
50.     END IF
51.
52.     ShowCompressionStats(quadtree, executionTime,
53.     originalImageSize, compressedImageSize)
54.     interface.ShowThankYou()
55. END FUNCTION

```

Fungsi main merupakan titik masuk program yang menangani interaksi dengan pengguna melalui antarmuka yang telah didefinisikan. Fungsi ini memandu pengguna melalui berbagai tahapan konfigurasi parameter kompresi, menjalankan proses kompresi, dan menampilkan hasilnya.

2.3.5. Pseudocode Visualisasi GIF

```

1. Function CaptureFrameForGif(currentImage)
2.     If visualizeGif is false
3.         Return
4.     EndIf
5.
6.     Increment frameCounter
7.     If gifFrames.size() < 5 OR frameCounter % 10 == 0 AND
       gifFrames.size() < 30 OR frameCounter % 30 == 0 AND
       gifFrames.size() < 60
8.         Set targetWidth = 640
9.         Set targetHeight = 480
10.        Calculate scaleX and scaleY based on targetWidth and
          targetHeight
11.
12.        If scale < 1.0
13.            Resize currentImage to new size
14.        Else
15.            Use currentImage as is

```

```

16.      EndIf
17.
18.      Create visImage as clone of currentImage
19.      Draw quadtree visualization on visImage
20.      Add infoText to visImage with current frame number
21.      Add visImage to gifFrames
22.  EndIf
23. EndFunction

```

2.3.5. Pseudocode Target Kompresi

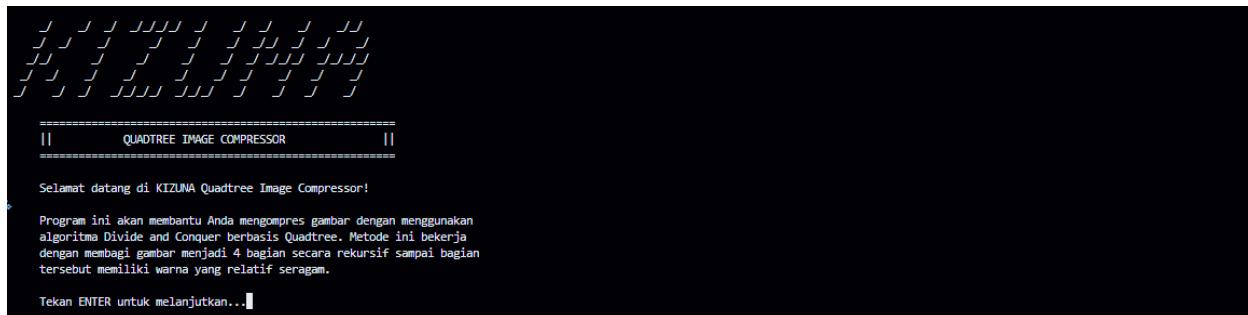
```

1. Function AdjustThresholdForTargetCompression(image)
2.   If targetCompressionPct <= 0.0
3.     Output "Target compression is disabled. Using standard
   threshold-based compression."
4.   Return
5.
6.   Set targetPct = targetCompressionPct
7.   If targetPct < 20.0
8.     Output "Using high precision compression"
9.     Set threshold based on errorMethod (for example, 5.0
   for Variance)
10.    Calculate gridSize for blocks
11.    Adjust minBlockSize based on gridSize
12.    Return
13.  ElseIf targetPct < 75.0
14.    Output "Using fixed-grid compression"
15.    Set threshold based on previous thresholds
16.    Calculate targetLeafNodes based on targetPct
17.    Adjust minBlockSize and maxDepth for better
   compression
18.    Return
19.  Else
20.    Output "Using adaptive compression"
21.    Set high and low ranges based on errorMethod
22.    Start iterative process to adjust threshold
23.    Adjust threshold using the best-fit method
24.  EndIf
25. EndFunction

```

2.3. Alur Program

Program Quadtree Image Compressor alias KIZUNA memiliki alur program yang didesain secara terstruktur dengan antarmuka pengguna yang intuitif, memungkinkan pengguna mengonfigurasi parameter kompresi dengan mudah dan melihat hasil secara visual. Program dimulai dengan menampilkan tampilan awal berupa logo "KIZUNA Quadtree Image Compressor" yang memberikan identitas visual pada aplikasi. Pengguna kemudian disambut dengan penjelasan singkat mengenai cara kerja program, menekankan penggunaan algoritma divide and conquer yang membagi gambar menjadi empat bagian secara rekursif hingga mencapai tingkat homogenitas warna yang ditentukan.



Setelah itu, program meminta input berupa alamat file gambar yang akan dikompresi. Setelah file dipilih, program memvalidasi keberadaan dan format file, kemudian menampilkan informasi dimensi dan kanal warna gambar untuk referensi pengguna. Proses ini memastikan bahwa gambar yang dimasukkan valid dan dapat diproses oleh algoritma. Selanjutnya, pengguna diminta memilih metode pengukuran error yang akan digunakan dalam proses kompresi. Lima opsi tersedia: Variance (variansi statistik antar kanal warna), Mean Absolute Deviation (MAD), Max Pixel Difference (perbedaan maksimum dalam blok), Entropy (pengukuran keacakan warna), dan Structural Similarity Index (SSIM) sebagai fitur bonus. Setiap metode menawarkan karakteristik berbeda dalam menentukan homogenitas area gambar.

Setelah memilih metode error, pengguna diminta memasukkan nilai threshold yang menentukan agresivitas kompresi. Program menyediakan panduan rentang nilai yang direkomendasikan untuk setiap metode, membantu pengguna memilih nilai yang sesuai. Threshold rendah menghasilkan kualitas lebih tinggi dengan kompresi minimal, sedangkan threshold tinggi menghasilkan kompresi lebih agresif dengan potensi penurunan kualitas. Pengguna juga diberikan rentang yang direkomendasikan berdasarkan metode yang dipilih, contohnya untuk Variance direkomendasikan antara 10-1000.

```

+-----+
| INPUT GAMBAR |
+-----+
Format: Silakan berikan path lengkap ke file gambar.
Contoh: /home/user/images/sample.jpg atau C:\Users\user\Pictures\sample.png
Format yang didukung: JPG, JPEG, PNG, WEBP, BMP, TIFF, PPM/PGM

Masukkan path file gambar input: C:\Users\DAWENDRA\OneDrive\Documents\ITB\SEMESTER 4\IF2211 Strategi Algoritma\Tucil 2\test\tes1.png
[BERHASIL] File gambar berhasil diperiksa.
Memuat gambar... Selesai!
[INFO] Dimensi gambar: 360x360 piksel
[INFO] Kanal warna: 3

+-----+
| METODE PENGUKURAN ERROR |
+-----+
Pilih metode pengukuran error:
1. Variance - Variansi statistik antar kanal warna
2. Mean Absolute Deviation (MAD) - Rata-rata perbedaan dari warna rata-rata
3. Max Pixel Difference - Perbedaan warna maksimum dalam blok
4. Entropy - Pengukuran keacakan warna dari teori informasi
5. Structural Similarity Index (SSIM) - Metrik kesamaan perceptual [BONUS]

Masukkan pilihan (1-5): 1
[BERHASIL] Metode terpilih: Variance

+-----+
| NILAI THRESHOLD |
+-----+
Threshold menentukan seberapa agresif gambar akan dikompresi.
- Threshold rendah = kualitas lebih tinggi, kompresi lebih sedikit
- Threshold tinggi = kualitas lebih rendah, kompresi lebih banyak

Rentang threshold yang direkomendasikan untuk Variance:
- Minimum yang direkomendasikan: 10 (kompresi minimal)
- Nilai tengah: 100.0 (keseimbangan kualitas/kompresi)
- Maksimum yang direkomendasikan: 1000 (kompresi maksimal)
- Nilai tipikal: 30-200 untuk sebagian besar gambar

Masukkan nilai threshold (rentang yang direkomendasikan: 10 sampai 1000): 50
[BERHASIL] Threshold diatur ke: 50.000000

```

Parameter berikutnya adalah ukuran blok minimum yang menentukan batas granularitas pembagian gambar. Nilai yang lebih kecil mempertahankan detail lebih banyak tetapi mengurangi efisiensi kompresi, sementara nilai lebih besar meningkatkan kompresi dengan potensi kehilangan detail. Sebagai fitur bonus, program memungkinkan pengguna menentukan target persentase kompresi. Jika fitur ini diaktifkan, algoritma akan secara dinamis menyesuaikan threshold untuk mencapai target kompresi yang diinginkan, terlepas dari nilai threshold yang dimasukkan sebelumnya. Pengguna dapat memasukkan nilai 0 untuk menonaktifkan fitur ini. Setelah konfigurasi parameter, pengguna diminta memasukkan path output untuk menyimpan gambar hasil kompresi dan file visualisasi GIF.

```

+-----+
| UKURAN BLOK MINIMUM |
+-----+
Ukuran blok minimum menentukan blok terkecil yang akan dibuat oleh Quadtree.
- Nilai lebih kecil (mis., 2, 4) mempertahankan detail lebih banyak tapi mengurangi kompresi
- Nilai lebih besar (mis., 8, 16, 32) meningkatkan kompresi tapi detail bisa hilang

Praktik terbaik:
- Gunakan pangkat dari 2 (2, 4, 8, 16, 32) untuk kinerja optimal
- Untuk gambar dengan detail tinggi, gunakan nilai lebih kecil (2-4)
- Untuk gambar lebih sederhana, nilai lebih besar (8-16) bisa lebih baik
- Nilai antara 2 dan 16 biasanya paling berguna

Berdasarkan ukuran gambar Anda (360x360):
- Minimum yang direkomendasikan: 2
- Maksimum yang direkomendasikan: 16

Masukkan ukuran blok minimum: 4
[BERHASIL] Ukuran blok minimum diatur ke: 4

+-----+
| PERSENTASE KOMPRESI TARGET [BONUS] |
+-----+
Fitur BONUS ini memungkinkan algoritma menyesuaikan threshold secara otomatis
untuk mencapai rasio kompresi tertentu, terlepas dari threshold yang Anda atur sebelumnya.

Panduan:
- 0.0 = Nonaktifkan penyesuaian otomatis (gunakan nilai threshold sebelumnya)
- 1-30% = Kompresi rendah, kualitas tinggi
- 30-60% = Kompresi sedang, kualitas baik
- 60-80% = Kompresi tinggi, kualitas berkurang
- 80-95% = Kompresi sangat tinggi, kualitas turun signifikan
- Nilai di atas 95% mungkin sulit dicapai tanpa penurunan kualitas yang parah

Catatan: Algoritma akan mencoba mendekati target Anda sedekat mungkin, tetapi
persentase yang persis mungkin tidak dapat dicapai untuk semua gambar.

Masukkan persentase kompresi target (0.0 untuk nonaktifkan, mis., 50.0 untuk 50%): 0.0
[INFO] Target kompresi dinonaktifkan. Menggunakan kompresi berbasis threshold saja.

+-----+
| GAMBAR OUTPUT |
+-----+
Masukkan path file gambar output (kosongkan untuk default): 1
[INFO] Menambahkan ekstensi file: 1.png

+-----+
| SIMPAN GAMBAR OUTPUT |
+-----+
Simpan gambar terkompresi? [1-Ya, 0-Tidak]: 1

```

```

+-----+
| VISUALISASI GIF [BONUS] |
+-----+
Buat visualisasi GIF? [1-Ya, 0-Tidak]: 1
Masukkan path file output GIF (kosongkan untuk default): gif1
[INFO] Menggunakan ekstensi file .gif: gif1.gif

+-----+
| PEMROSESAN |
+-----+

Ringkasan Parameter Kompresi:
- Gambar Input: C:\Users\DAMENDRA\OneDrive\Documents\ITB\SEMESTER 4\IF2211 Strategi Algoritma\Tucil 2\test\tes1.png
- Metode Error: Mean Absolute Deviation
- Threshold: 15
- Ukuran Blok Minimum: 4
- Kompresi Target: Dinonaktifkan
- Gambar Output: 2.png
- Buat GIF: Ya
- Output GIF: gif1.gif

Lanjutkan dengan kompresi? [/n]: y

```

```

Buat visualisasi GIF? [1-Ya, 0-Tidak]: 0

+-----+
| PEMROSESAN |
+-----+

Ringkasan Parameter Kompresi:
- Gambar Input: C:\Users\DANENDRA\OneDrive\Documents\ITB\SEMESTER 4\IF2211 Strategi Algoritma\Tucil 2\test\tes1.png
- Metode Error: Variance
- Threshold: 50
- Ukuran Blok Minimum: 4
- Kompresi Target: Dinonaktifkan
- Gambar Output: 1.png
- Buat GIF: Tidak

Lanjutkan dengan kompresi? [y/n]: y

```

Program kemudian menampilkan ringkasan parameter yang telah dikonfigurasi dan meminta konfirmasi untuk melanjutkan proses. Saat pemrosesan dimulai, program menampilkan progress bar yang menunjukkan kemajuan kompresi, memberikan umpan balik visual kepada pengguna. Proses kompresi terdiri dari tiga tahap utama: membuat quadtree, mengompresi gambar berdasarkan parameter yang ditentukan, dan merekonstruksi gambar dari struktur quadtree yang telah dibuat. Setelah kompresi selesai, program menampilkan hasil statistik detail yang mencakup waktu eksekusi, ukuran gambar sebelum dan sesudah kompresi, persentase kompresi yang dicapai, threshold akhir yang digunakan, kedalaman quadtree, dan jumlah node dalam struktur. Pengguna juga diberikan opsi untuk membuka gambar asli dan terkompresi secara berdampingan untuk perbandingan visual.

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##                                     ] 0.0%
[###                                    ] 5.0%
[####                                   ] 10.0%
[#####                                  ] 15.0%
[######                                 ] 20.0%
[#######                                ] 25.0%
[########                               ] 30.0%
[#########                              ] 35.0%
[##########                             ] 40.0%
[###########                            ] 45.0%
[############                           ] 50.0%
[#############                          ] 55.0%
[##############                         ] 60.0%
[###############                        ] 65.0%
[################                       ] 70.0%
[#################                      ] 75.0%
[##################                     ] 80.0%
[###################                    ] 85.0%
[####################                   ] 90.0%
[#####################                  ] 95.0%
[######################                 ] 100.0%
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 50.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 50.0 in 11 ms
Merekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 5751 bytes
Compressed: 10075 bytes
Percentase kompresi: -75.24%

+-----+
| HASIL KOMPRESI
+-----+
| Waktu eksekusi | 547.021200 ms
| Ukuran gambar asli | 5751 bytes
| Ukuran gambar terkompresi | 10075 bytes
| Persentase kompresi | -75.186924%
| Threshold akhir | 50.00
| Kedalaman Quadtree | 8
| Jumlah node dalam Quadtree | 6813
+-----+
Kompresi berhasil diselesaikan!
Gambar terkompresi disimpan ke: 1.png

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sombras tombol pada jendela gambar untuk menutupnya.

```

Terima kasih telah menggunakan KIZUNA Quadtree Image Compressor!

Sampai jumpa kembali...

BAB III

HASIL PROGRAM

3.1. tes1.png

Parameters :

- Image: tes1.PNG
- Method: Variance
- Threshold: 50
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes1.png via terminal:

```
Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[   ] 0.0%
[##] 5.0%
[#####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0% [BERHASIL] Gambar terkompresi berhasil disimpan.

Kemudian pindah ke direktori... Selesai!
Mengompresi gambar... Selesai!
Starting compression with threshold: 50.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 50.0 in 11 ms
Merestrukturasi gambar... Selesai!
Memimpin gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 5751 bytes
Compressed: 10075 bytes
Persentase kompresi: -75.2%
```

-----| HASIL KOMPRESI |-----

Muktu eksekusi	547.023200 ms
Ukuran gambar asli	5751 bytes
Ukuran gambar terkompresi	10075 bytes
Persentase kompresi	-75.186924%
Threshold akhir	50.00
Kedalaman Quadtree	8
Jumlah node dalam Quadtree	6813

-----| Kompresi berhasil diselesaikan! |-----

Gambar terkompresi disimpan ke: 1.png

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.



Parameters :

- Image: tes1.PNG
- Method: MAD
- Threshold: 15

- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes1.png via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Mulai proses kompresi...
[##          ] 0.0%
[###         ] 5.0%
[####        ] 10.0%
[#####       ] 15.0%
[######      ] 20.0%
[#######     ] 25.0%
[########    ] 30.0%
[#########   ] 35.0%
[##########  ] 40.0%
[########### ] 45.0%
[############] 50.0%
[############] 55.0%
[############] 60.0%
[############] 65.0%
[############] 70.0%
[############] 75.0%
[############] 80.0%
[############] 85.0%
[############] 90.0%
[############] 95.0%
[############] 100.0%
[############] Selesai!
Kompreksi gambar menggunakan Quadtree...
Starting compression with threshold: 15.0
Method: Mean Absolute Deviation
Quadtree compression completed successfully
Compression complete with threshold: 15.0 in 7 ms
Rekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 5951 bytes
Compressdi: 9712 bytes
Persentase kompresi: -68.9%
-----+-----+-----+
| HASIL KOMPRESI |
-----+-----+-----+
| Waktu eksekusi | 515.018700 ms
| Ukuran gambar asli | 5951 bytes
| Ukuran gambar terkompresi | 9712 bytes
| Persentase kompresi | -68.94978%
| Threshold akhir | 15.00
| Kedalaman Quadtree | 8
| Jumlah node dalam Quadtree | 3499
-----+-----+-----+
Kompresi berhasil diselesaikan!
Gambar terkompresi disimpan ke: 2.png

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

Parameters:

- Image: tes1.PNG
- Method: Max Pixel Difference
- Threshold: 35
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes1.png via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##] 0.0%
[###] 5.0%
[####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0% [=====]
Menbuat quadtree.. Selesai!
Menyimpan gambar.. Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 35.0 in 5 ms
Method: Max Pixel Difference
Quadtree compression completed successfully
Compression complete with threshold: 35.0 in 5 ms
Menyimpan gambar.. Selesai!
Menyimpan gambar terkompresi.. Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 5751 bytes
Compressed: 9908 bytes
Percentase kompresi: -72.3%
```

HASIL KOMPRESI	
Waktu eksekusi	516.692300 ms
Ukuran gambar asli	5751 bytes
Ukuran gambar terkompresi	9908 bytes
Percentase kompresi	-72.283881%
Threshold akhir	35.00
Kedalaman Quadtree	8
Jumlah node dalam Quadtree	4261

Kompresi berhasil diselesaikan!

Gambar terkompresi disimpan ke: 3.png

Apakah Anda ingin membuat gambar asli dan terkompresi untuk perbandingan? [y/n]: y

[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

Parameters:

- Image: tes1.PNG
- Method: Entropy
- Threshold: 1.0
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes1.png via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##] 0.0%
[###] 5.0%
[###] 10.0%
[####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0%
[#####] Membuat quadtree... Selesai!
[#####] Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 1.0
Method: Entropy
Quadtree compression completed successfully
Compression complete with threshold: 1.0 in 3 ms
    Merenkonstruksi gambar... Selesai!
    Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 5751 bytes
Compressed: 8297 bytes
Persentase kompresi: -44.3%
+-----+
| HASIL KOMPRESI |
+-----+
| Waktu eksekusi | 521.880600 ms
| Ukuran gambar asli | 5751 bytes
| Ukuran gambar terkompresi | 8297 bytes
| Persentase kompresi | -44.270562%
| Threshold akhir | 1.00
| Kedalaman Quadtree | 8
| Jumlah node dalam Quadtree | 2517
+-----+
Kompresi berhasil diselesaikan!
Gambar terkompresi disimpan ke: 4.png

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

Parameters :

- Image: tes1.PNG
- Method: SSIM
- Threshold: 0.15
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes1.png via terminal:

```

Lanjutkan dengan kompresi? [./n]: y
[INFO] Memulai proses kompresi...
[
  [##] 0.0%
  [###] 5.0%
  [####] 10.0%
  [#####] 15.0%
  [#####] 20.0%
  [#####] 25.0%
  [#####] 30.0%
  [#####] 35.0%
  [#####] 40.0%
  [#####] 45.0%
  [#####] 50.0%
  [#####] 55.0%
  [#####] 60.0%
  [#####] 65.0%
  [#####] 70.0%
  [#####] 75.0%
  [#####] 80.0%
  [#####] 85.0%
  [#####] 90.0%
  [#####] 95.0%
  [#####] 100.0%
]
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing Image using Quadtree...
Starting compression with threshold: 0.1
Method: SSIM
Quadtree compression completed successfully
Compression complete with threshold: 0.1 in 16 ms
Rekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 5751 bytes
Compressed: 9537 bytes
Persentase kompresi: -65.8%
HASIL KOMPRESI
+-----+
| Waktu eksekusi | 542.348800 ms |
| Ukuran gambar asli | 5751 bytes |
| Ukuran gambar terkompresi | 9537 bytes |
| Persentase kompresi | -65.832029% |
| Threshold akhir | 0.15 |
| Kedalaman Quadtree | 8 |
| Jumlah node dalam Quadtree | 8829 |
+-----+
[compress berhasil diselesaikan]
Gambar terkompresi disimpan ke: 5.png

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [./n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

3.2. tes2.jpg

Parameters :

- Image: tes2(JPG)
- Method: Variance
- Threshold: 10.0
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes2.jpg via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##] 0.0%
[##] 5.0%
[####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0%
Membuat quadtree... Selesai!
Mengompres gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 10.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 10.0 in 13 ms
Rekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 173485 bytes
Compressed: 115906 bytes
Persentase kompresi: 33.7%
+-----+
| HASIL KOMPRESI |
+-----+
| Waktu eksekusi | 546.813600 ms |
| Ukuran gambar asli | 173485 bytes |
| Ukuran gambar terkompresi | 115906 bytes |
| Persentase kompresi | 33.708390% |
| Threshold akhir | 10.00 |
| Kedalaman Quadtree | 9 |
| Jumlah node dalam Quadtree | 47829 |
+-----+
[Versiensi libjpeg-turbo disebutkan]
Gambar terkompresi disimpan ke: 6.jpg

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

Parameters :

- Image: tes2.JPG
- Method: Variance
- Threshold: 50.0
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes2.jpg via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##] 0.0%
[###] 5.0%
[####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0%
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 50.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 50.0 in 15 ms
Merekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BEWARE!] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 173485 bytes
Compressed: 102023 bytes
Persentase kompresi: 41.2%
+-----+
| HASIL KOMPRESI |
+-----+
| Waktu eksekusi | 544.813000 ms |
| Ukuran gambar asli | 173485 bytes |
| Ukuran gambar terkompresi | 102023 bytes |
| Persentase kompresi | 41.192834% |
| Threshold akhir | 50.00 |
| Kedalaman Quadtree | 9 |
| Jumlah node dalam Quadtree | 34769 |
+-----+
Kompresi berhasil diselesaikan!
Gambar terkompresi disimpan ke: 7.jpg

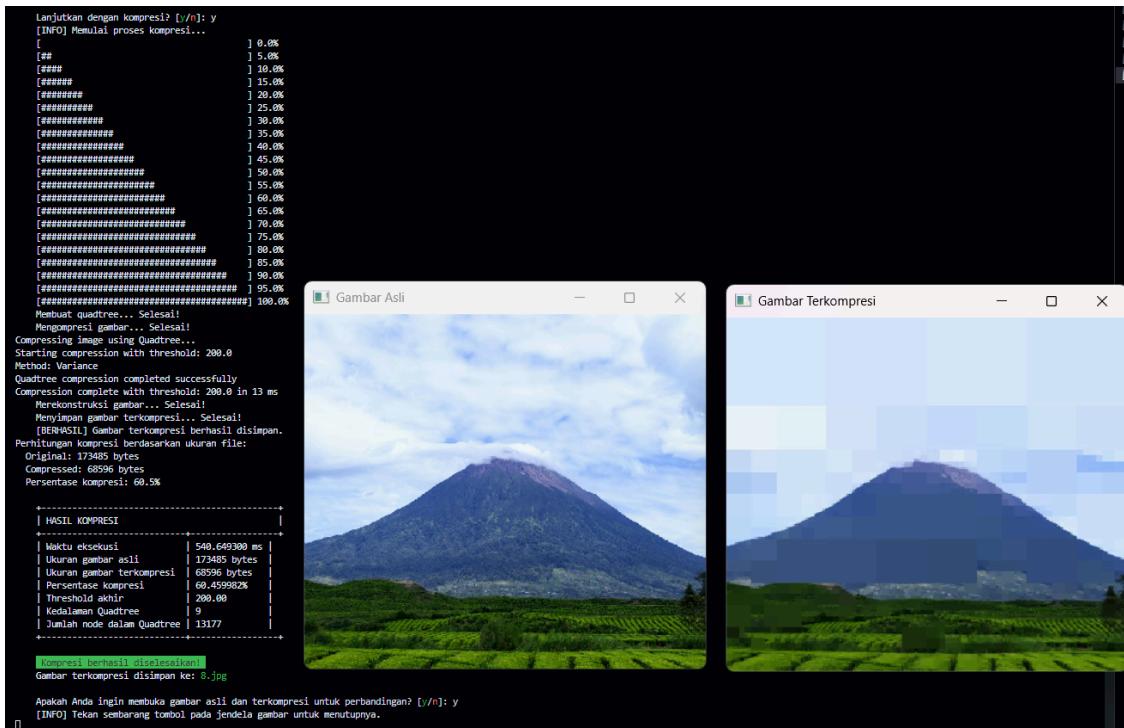
Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

Parameters:

- Image: tes2.JPG
- Method: Variance
- Threshold: 200.0
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes2.jpg via terminal:



3.3. tes3.jpg

Parameters:

- Image: tes3.jpg
- Method: Variance
- Threshold: 50.0
- Min Block Size: 2
- Target Compression: Disabled

Berikut merupakan hasil output tes3.jpg via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[          ] 0.0%
[##       ] 5.0%
[###     ] 10.0%
[####   ] 15.0%
[##### ] 20.0%
[###### ] 25.0%
[#######] 30.0%
[#######] 35.0%
[#######] 40.0%
[#######] 45.0%
[#######] 50.0%
[#######] 55.0%
[#######] 60.0%
[#######] 65.0%
[#######] 70.0%
[#######] 75.0%
[#######] 80.0%
[#######] 85.0%
[#######] 90.0%
[#######] 95.0%
[#######] 100.0%
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 50.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 50.0 in 51 ms
    Merekonstrui gambar... Selesai!
    Menyimpan gambar terkompresi... Selesai!
    [BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 854565 bytes
Compressed: 284453 bytes
Persentase kompresi: 66.7%

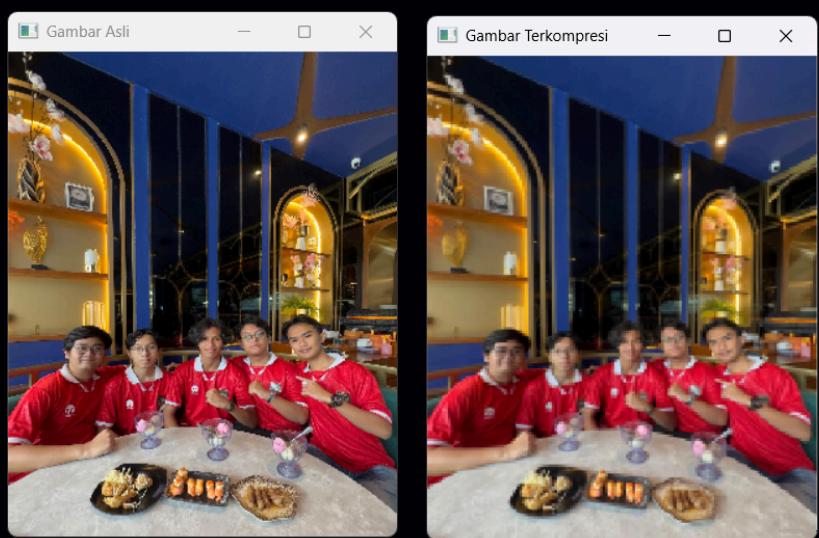

+-----+
| HASIL KOMPRESI |
+-----+
| Waktu eksekusi      | 606.872600 ms |
| Ukuran gambar asli  | 854565 bytes |
| Ukuran gambar terkompresi | 284453 bytes |
| Persentase kompresi  | 66.713708% |
| Threshold akhir    | 50.00 |
| Kedalaman Quadtree | 11 |
| Jumlah node dalam Quadtree | 150017 |
+-----+



Kompresi berhasil diselesaikan
Gambar terkompresi simpan ke: 9.jpg

Apakah Anda ingin membuka gambar asli dan terkompresi?
[INFO] Tekan sembarang tombol pada jendela gambar untuk

```



Parameters:

- Image: tes3(JPG)
 - Method: Variance
 - Threshold: 50.0
 - Min Block Size: 8
 - Target Compression: Disabled

Berikut merupakan hasil output tes3.jpg via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[
  ##          ] 0.0%
  #####      ] 5.0%
  #######    ] 10.0%
  #########  ] 15.0%
  ########## ] 20.0%
  ###########] 25.0%
  ###########] 30.0%
  ###########] 35.0%
  ###########] 40.0%
  ###########] 45.0%
  ###########] 50.0%
  ###########] 55.0%
  ###########] 60.0%
  ###########] 65.0%
  ###########] 70.0%
  ###########] 75.0%
  ###########] 80.0%
  ###########] 85.0%
  ###########] 90.0%
  ###########] 95.0%
  ###########] 100.0%
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 50.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 50.0 in 51 ms
Rekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 854565 bytes
Compressed: 1000619 bytes
Persentase kompresi: -17.1%
+-----+
| HASIL KOMPRESI
+-----+
| Waktu eksekusi | 684.467100 ms |
| Ukuran gambar asli | 854565 bytes |
| Ukuran gambar terkompresi | 1000619 bytes |
| Persentase kompresi | -17.091035% |
| Threshold akhir | 50.00 |
| Kedalaman Quadtree | 10 |
| Jumlah node dalam Quadtree | 141565 |
+-----+
[Kompresi berhasil diselesaikan!]
Gambar terkompresi disimpan ke: 10.jpg

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```




Parameters :

- Image: tes3.JPG
- Method: Variance
- Threshold: 50.0
- Min Block Size: 16
- Target Compression: Disabled

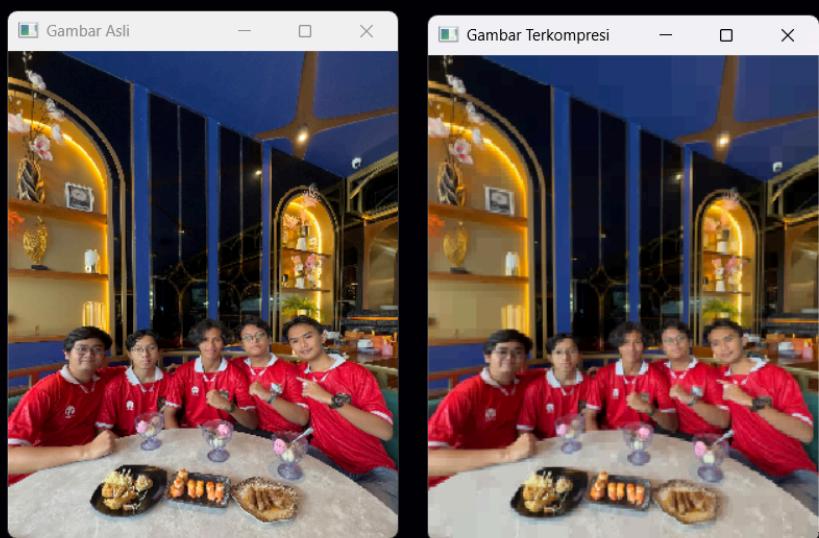
Berikut merupakan hasil output tes3.jpg via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[
[##] 0.0%
[###] 5.0%
[####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0%
Membuat quadtree... Selesai!
Mengompres gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 50.0
Method: Variance
Quadtree Compression completed successfully
Compression complete with threshold: 50.0 in 45 ms
Merekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 854565 bytes
Compressed: 849536 bytes
Persentase kompresi: 0.6%
```

+-----+ HASIL KOMPRESI +-----+	
Waktu eksekusi 601.527500 ms	
Ukuran gambar asli 854565 bytes	
Ukuran gambar terkompresi 849536 bytes	
Persentase kompresi 0.588487%	
Threshold akhir 50.00	
Kedalaman Quadtree 9	
Jumlah node dalam Quadtree 48537	

Kompresi berhasil diselesaikan!
 Gambar terkompresi disimpan ke: 11.jpg



3.4. tes4.jpg

Parameters:

- Image: tes4.jpg
 - Method: Variance
 - Threshold: 50.0
 - Min Block Size: 4
 - Target Compression: 30

Berikut merupakan hasil output tes4.jpg via terminal:

```

[
  [ 0.0%
  [## 5.0%
  ##### 10.0%
  ##### 15.0%
  ##### 20.0%
  ##### 25.0%
  ##### 30.0%
  ##### 35.0%
  ##### 40.0%
  ##### 45.0%
  ##### 50.0%
  ##### 55.0%
  ##### 60.0%
  ##### 65.0%
  ##### 70.0%
  ##### 75.0%
  ##### 80.0%
  ##### 85.0%
  ##### 90.0%
  ##### 95.0%
  ##### 100.0%
  Membuat quadtree... Selesai!
  Mengompresi gambar... Selesai!
  Compressing image using Quadtree...
  Target kompresi 50.0%. Menggunakan pendekatan fixed-grid
  - Target leaf nodes: 72352 dari 103360 pixels
  - Menggunakan mode Fixed-grid dengan ukuran blok 1x1
  - Prediksi kompresi: 0.0%
  - Batas kedalaman: 9
  - Region tengah: 163x228 dengan detail tinggi
  Starting compression with threshold: 50.0
  Method: Variance
  Quadtree compression completed successfully
  Compression complete with threshold: 50.0 in 8 ms
  Merekonstruksi gambar... Selesai!
  Menyimpan gambar terkompresi... Selesai!
  [BERHASIL] Gambar terkompresi berhasil disimpan.
  Perhitungan kompresi berdasarkan ukuran file:
  Original: 35262 bytes
  Compressed: 22197 bytes
  Persentase kompresi: 37.1%
  +-----+
  | HASIL KOMPRESI |
  +-----+
  | Waktu eksekusi | 511.510600 ms |
  | Ukuran gambar asli | 35262 bytes |
  | Ukuran gambar terkompresi | 22197 bytes |
  | Persentase kompresi | 37.051217% |
  | Threshold akhir | 50.00 |
  | Kedalaman Quadtree | 9 |
  | Jumlah node dalam Quadtree | 12065 |
  +-----+
  Kompresi berhasil diselesaikan!
  Gambar terkompresi disimpan ke: 12.jpg

  Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
  [INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

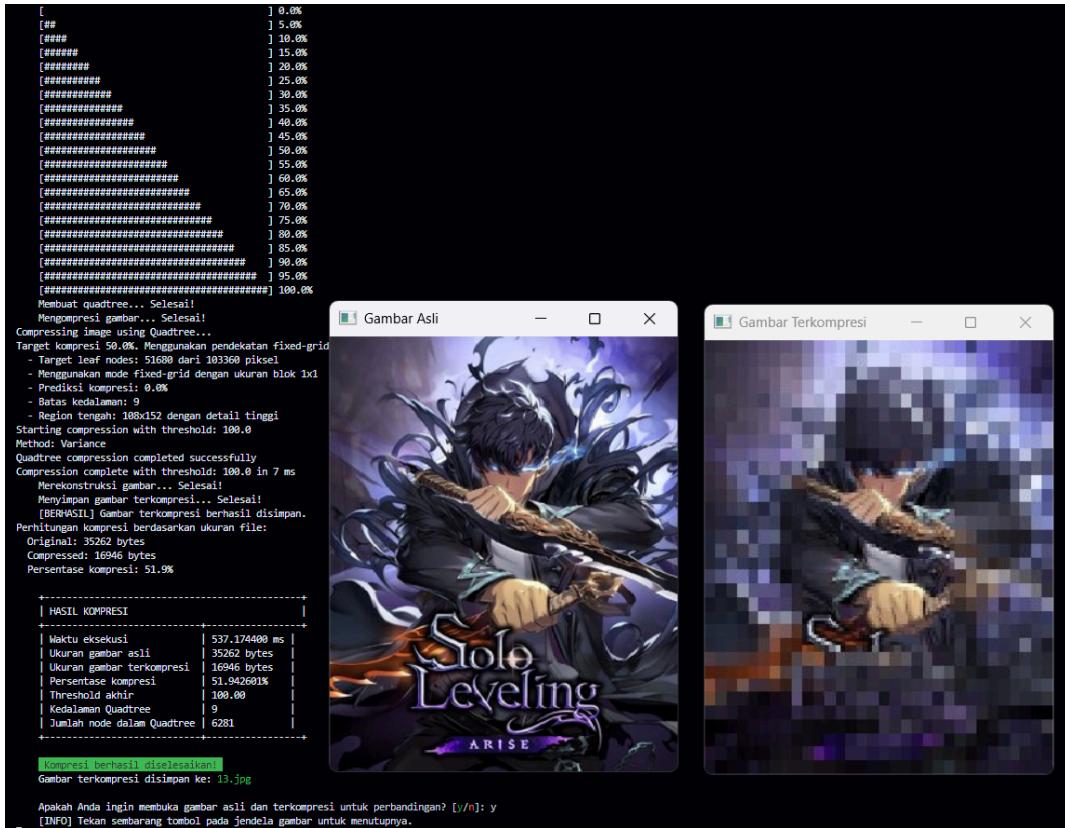
```



Parameters:

- Image: tes4.jpg
- Method: Variance
- Threshold: 100.0
- Min Block Size: 4
- Target Compression: 50

Berikut merupakan hasil output tes4.jpg via terminal:



Parameters :

- Image: tes4.jpg
- Method: Variance
- Threshold: 200.0
- Min Block Size: 4
- Target Compression: 100

Berikut merupakan hasil output tes4.jpg via terminal:

```

Iteration 4: Testing threshold = 495.9
Compressing image using Quadtree...
Starting compression with threshold: 495.9
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 495.9 in 3 ms
    Current compression: 96.7%
Iteration 5: Testing threshold = 498.8
Compressing image using Quadtree...
Starting compression with threshold: 498.8
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 498.8 in 3 ms
    Current compression: 96.7%
Iteration 6: Testing threshold = 499.6
Compressing image using Quadtree...
Starting compression with threshold: 499.6
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 499.6 in 3 ms
    Current compression: 96.7%
Fine-tuning with threshold = 516.0
Compressing image using Quadtree...
Starting compression with threshold: 516.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 516.0 in 3 ms
Using best threshold = 516.0
Estimated final compression: within 3.3% of target
Starting compression with threshold: 516.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 516.0 in 39 ms
    Merekonstruksi gambar... Selesai!
    Menyimpan gambar terkompresi... Selesai!
    [BERHASIL] Gambar terkompresi berhasil disimpan.
Perintungan kompresi berdasarkan ukuran file:
Original: 35262 bytes
Compressed: 15300 bytes
Persentase kompresi: 56.6%

+-----+
| HASIL KOMPRESI |
+-----+
| Waktu eksekusi | 556.802300 ms |
| Ukuran gambar asli | 35262 bytes |
| Ukuran gambar terkompresi | 15300 bytes |
| Persentase kompresi | 56.610516% |
| Threshold akhir | 515.97 |
| Kedalaman Quadtree | 8 |
| Jumlah node dalam Quadtree | 4517 |
+-----+
[Kompresi berhasil diselesaikan]
Gambar terkompresi disimpan ke: 14.jpg

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [Y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

3.5. tes5.jpg

Parameters :

- Image: tes5.jpg
- Method: Variance
- Threshold: 50.0
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes5.jpg via terminal:

```

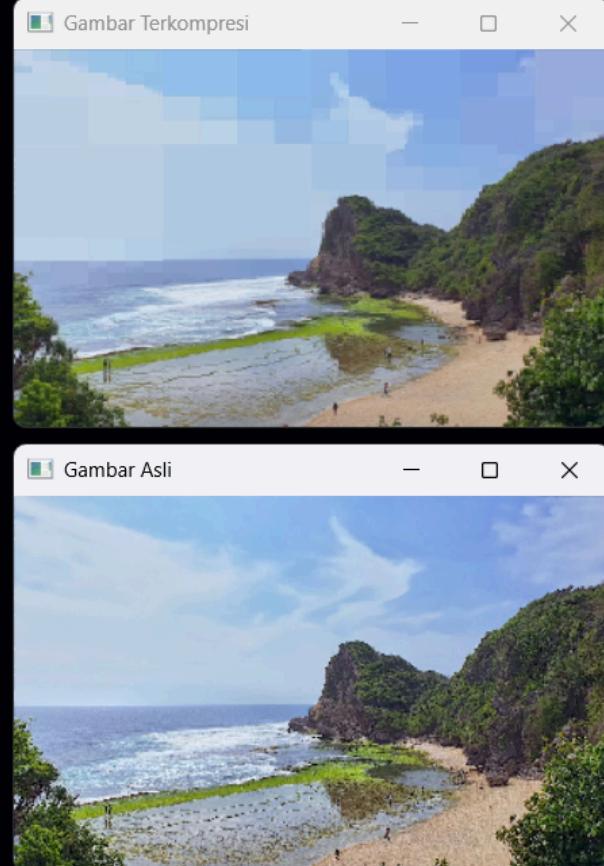
Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##] 0.0%
[###] 5.0%
[####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0%
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 50.0
Method: Variance
Quadtree compression completed successfully
Compression complete with threshold: 50.0 in 12 ms
Merekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 165890 bytes
Compressed: 112288 bytes
Persentase kompresi: 32.3%

+-----+
| HASIL KOMPRESI
+-----+
| Waktu eksekusi | 539.203500 ms |
| Ukuran gambar asli | 165890 bytes |
| Ukuran gambar terkompresi | 112288 bytes |
| Persentase kompresi | 32.311773% |
| Threshold akhir | 50.00 |
| Kedalaman Quadtree | 9 |
| Jumlah node dalam Quadtree | 44117 |
+-----+

Kompresi berhasil diselesaikan!
Gambar terkompresi disimpan ke: 15.jpg

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```



Parameters :

- Image: tes5.jpg
- Method: SSIM
- Threshold: 0.1
- Min Block Size: 4
- Target Compression: Disabled

Berikut merupakan hasil output tes5.jpg via terminal:

```

Lanjutkan dengan kompresi? [y/n]: y
[INFO] Memulai proses kompresi...
[##] 0.0%
[###] 5.0%
[####] 10.0%
[#####] 15.0%
[#####] 20.0%
[#####] 25.0%
[#####] 30.0%
[#####] 35.0%
[#####] 40.0%
[#####] 45.0%
[#####] 50.0%
[#####] 55.0%
[#####] 60.0%
[#####] 65.0%
[#####] 70.0%
[#####] 75.0%
[#####] 80.0%
[#####] 85.0%
[#####] 90.0%
[#####] 95.0%
[#####] 100.0%
Membuat quadtree... Selesai!
Mengompresi gambar... Selesai!
Compressing image using Quadtree...
Starting compression with threshold: 0.1
Method: SSIM
Quadtree compression completed successfully
Compression complete with threshold: 0.1 in 50 ms
Merekonstruksi gambar... Selesai!
Menyimpan gambar terkompresi... Selesai!
[BERHASIL] Gambar terkompresi berhasil disimpan.
Perhitungan kompresi berdasarkan ukuran file:
Original: 165890 bytes
Compressed: 115893 bytes
Percentase kompresi: 30.1%

+-----+
| HASIL KOMPRESI |
+-----+
| Waktu eksekusi | 586.466100 ms |
| Ukuran gambar asli | 165890 bytes |
| Ukuran gambar terkompresi | 115893 bytes |
| Persentase kompresi | 30.138646% |
| Threshold akhir | 0.10 |
| Kedalaman Quadtree | 9 |
| Jumlah node dalam Quadtree | 48265 |
+-----+

[Kompresi berhasil diselesaikan]
Gambar terkompresi disimpan ke: 16.jpg

Apakah Anda ingin membuka gambar asli dan terkompresi untuk perbandingan? [y/n]: y
[INFO] Tekan sembarang tombol pada jendela gambar untuk menutupnya.

```

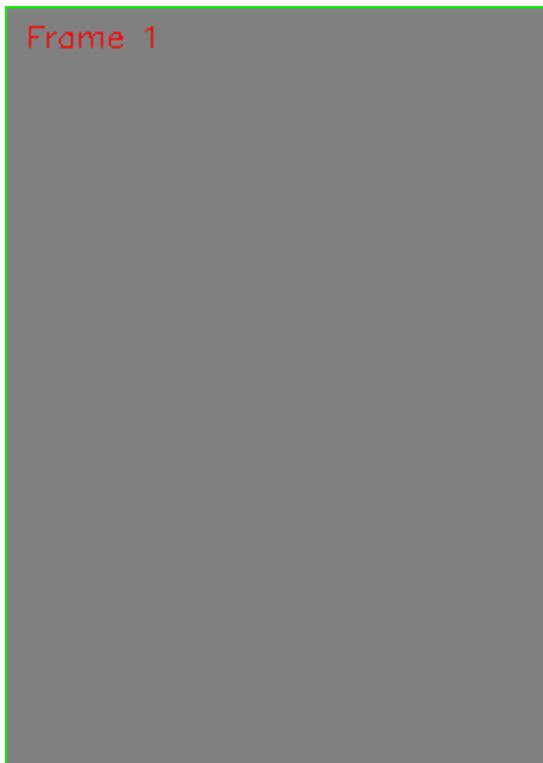
The terminal window shows the compression process of a coastal landscape image. The progress bar indicates the compression is at 100.0%. The command output details the compression parameters (threshold: 0.1, method: SSIM) and the resulting file size (original: 165890 bytes, compressed: 115893 bytes, percentage: 30.1%). A table summarizes the execution time and memory usage. Two windows titled "Gambar Terkompresi" and "Gambar Asli" show the original and compressed versions of the same coastal scene respectively.

3.6 Gif

Parameters:

- Image: tes4.jpg
- Method: Variance
- Threshold: 50.0
- Min Block Size: 4
- Target Compression: Disabled
- Gif: Ya

Berikut merupakan hasil output tes4.jpg via terminal:



BAB IV

PENUTUP

4.1. Kesimpulan

Algoritma divide and conquer yang diterapkan dalam kompresi gambar dengan metode Quadtree terbukti efektif dalam mengurangi ukuran gambar dengan memanfaatkan struktur data yang efisien. Kompresi yang dihasilkan sangat bergantung pada parameter seperti threshold dan ukuran blok minimum. Meskipun kompleksitas waktu dan ruangnya dapat meningkat untuk gambar dengan banyak detail, metode ini tetap menjadi pilihan yang baik untuk kompresi gambar yang besar dengan area homogen yang signifikan.

4.2. Source Code Repository

Link repository dapat diakses sebagai berikut: https://github.com/danenftyessir/Tucil2_13523136.git

4.1. Checklist Penyelesaian Tugas Kecil

No	Poin	Ya	Tidak
1	Program berhasil dikompilasi tanpa kesalahan	✓	
2	Program berhasil dijalankan	✓	
3	Program berhasil melakukan kompresi gambar sesuai parameter yang ditentukan	✓	
4	Mengimplementasi seluruh metode perhitungan error wajib	✓	
5	[Bonus] Implementasi persentase kompresi sebagai parameter tambahan	✓	
6	[Bonus] Implementasi Structural Similarity Index (SSIM) sebagai metode pengukuran error	✓	

7	[Bonus] Output berupa GIF Visualisasi Proses pembentukan Quadtree dalam Kompresi Gambar	✓	
9	Program dan laporan dibuat sendiri	✓	