JISKA (Jurnal Informatika Sunan Kalijaga)
Vol. 11, No. 1, JANUARI, 2026, Pp. 56 – 69
ISSN: 2527 – 5836 (print) | 2528 – 0074 (online)

Uji Efektifitas Kompresi Golomb-Rice dan Huffman untuk Metadata EXIF dalam File JPEG

Yasir Hasan
Departemen Informatika, STMIK Mulia Darma, Labuhanbatu, Indonesia
e-mail : yasirhasan.kom@gmail.com.
* Penulis korespondensi.
Artikel ini diajukan 6 Maret 2025, direvisi 29 Mei 2025, diterima 17 Juni 2025, dan dipublikasikan 25 Januari 2026.

Abstract

Compression algorithms are now called modern compression algorithms. This improvement is characterized by the combination of various classical techniques and is even based on machine learning and AI. However, the important part of compression is not only the algorithm, but also knowledge of the internal structure and metadata of the file is required. Like JPEG has a file structure that can be changed, cannot be changed, every marker (header), and EXIF metadata. Lack of knowledge of the file structure can cause data damage and file corruption. This study evaluates the compression of EXIF metadata of JPEG files using the Golomb-Rice and Huffman algorithms. Golomb-Rice can produce compression that affects the k parameter, while Huffman is optimal based on symbol frequency, but requires a code table. This study measures the effectiveness of both algorithms based on the compression ratio (CR). The test results of Golomb-Rice are more effective than those of Huffman. So, it can be concluded that the Golomb-Rice algorithm is superior in the context of compressing EXIF JPEG metadata, while Huffman shows lower efficiency in the tested scenarios.

Keywords: Golomb-Rice, Huffman, EXIF, JPEG, Compression Ratio

Abstrak

Algoritma kompresi sekarang disebut algoritma kompresi modern. Peningkatan ini dicirikan dengan penggabungkan berbagai teknik klasik dan bahkan berbasis machine learning dan AI. Namun, bagian penting kompresi bukan hanya algoritmanya, tetapi pengetahuan tentang struktur internal dan metadata file harus diketahui. Seperti JPEG memiliki struktur file yang bisa diubah, tidak bisa diubah, setiap penanda-penanda (header) dan metada EXIF. Kurangnya pengetahuan struktur file dapat menyebabkan kerusakan data dan file korup. Penelitian ini mengevaluasi kompresi metadata EXIF file JPEG menggunakan algoritma Golomb-Rice dan Huffman. Golomb-Rice dapat menghasilkan kompresi yang berpengaruh pada parameter k, sementara Huffman optimal berdasarkan frekuensi simbol, namun membutuhkan tabel kode. Penelitian ini mengukur efektivitas kedua algoritma berdasarkan rasio kompresi (CR). Hasil pengujian Golomb-Rice lebih efektif dibandingkan Huffman. Sehingga dapat disimpulkan algoritma Golomb-Rice lebih unggul dalam konteks pengompresian metadata EXIF JPEG, sementara Huffman menunjukkan efisiensi yang lebih rendah dalam skenario yang diuji.

Kata Kunci: Golomb-Rice, Huffman, EXIF, JPEG, Rasio Kompresi

1. PENDAHULUAN

Teknik kompresi yang salah dapat menyebabkan hilangnya data atau bahkan kerusakan file yang tidak dapat diperbaiki. Dalam banyak masalah, kompresi yang tidak memperhitungkan struktur file akan mengubah susunan bit secara tidak semestinya, menyebabkan file menjadi korup atau tidak dapat dibaca oleh perangkat lunak standar (Liu et al., 2022). Hal ini sering terjadi ketika kompresi dilakukan dengan algoritma yang tidak sesuai dengan karakteristik data yang dikompresi, seperti menggunakan metode lossy pada informasi yang seharusnya tetap utuh (Cai et al., 2024; Cappello et al., 2025; Mahmood & Wagner, 2023). Selain itu, beberapa teknik kompresi dapat menyebabkan metadata penting dalam file hilang, yang mengakibatkan informasi seperti tanggal pengambilan gambar atau lokasi geografis tidak lagi tersedia. Kesalahan dalam memahami format penyimpanan metadata juga dapat menyebabkan penghapusan atau

pengubahan bagian dari file yang penting untuk kompatibilitas (Doménech Fons & Pegueroles Vallés, 2021; Zheng et al., 2023). Oleh karena itu, pemilihan metode kompresi yang tepat sangat penting untuk memastikan bahwa file tetap dapat digunakan dengan baik setelah dikompresi (Martini, 2025).

File gambar memiliki struktur yang kompleks, dengan bagian-bagian yang yang dapat diubah (*editable*) dan yang tidak dapat diubah (*non-editable*). Metadata yang dapat diubah, seperti tanggal pengambilan gambar, informasi hak cipta, atau komentar pengguna, dapat dimodifikasi tanpa memengaruhi kualitas visual gambar (Mani et al., 2022; Stoilov, 2022). Namun, metadata yang tidak dapat diubah, seperti tabel kuantisasi, tabel Huffman, atau *marker* SOF (*Start of Frame*), bersifat kritis untuk merekonstruksi gambar. Kesalahan dalam memodifikasi metadata non-editable dapat menyebabkan file JPEG menjadi rusak atau tidak dapat dibaca (Itier et al., 2022; Mills, 2018). Oleh karena itu, teknik kompresi yang digunakan harus memperhatikan struktur file JPEG secara keseluruhan, termasuk metadata EXIF, untuk memastikan integritas data tetap terjaga. File gambar JPEG sebenarnya merupakan hasil kompresi dari format lain yang lebih besar, dengan menggunakan metode *Discrete Cosine Transform* (DCT) (Kumar & Kumar, 2021). Selain itu, komposisi warna dalam JPEG dikompresi dengan konversi ke ruang warna YCbCr. Meskipun telah terkompresi, file JPEG masih dapat dikompresi lebih lanjut, terutama jika mengandung metadata EXIF yang besar. Metadata ini tidak termasuk dalam kompresi utama JPEG dan sering kali menggunakan format teks atau biner yang kurang optimal dalam hal penyimpanan namun sangat berguna jika membutuhkan informasi khusus tersendiri (Acharya R et al., 2023; Ardiansyah et al., 2020; Wijayanto et al., 2016).

Teknik kompresi dapat dibagi menjadi dua kategori utama, yaitu kompresi yang mengubah format file dan kompresi yang mempertahankan format aslinya. Kompresi yang mengubah format file, seperti mengonversi JPEG ke format lain yang lebih efisien seperti PNG, WebP, atau HEIF, dapat menghasilkan pengurangan ukuran yang signifikan tetapi sering kali menyebabkan hilangnya kompatibilitas dengan perangkat lunak yang hanya mendukung format asli (Dhawan, 2011; Jamil, 2024). Sebaliknya, kompresi yang mempertahankan format awal, seperti kompresi *lossless* pada JPEG, memungkinkan metadata EXIF tetap utuh. Namun, teknik kompresi *lossless* seperti *Deflate* atau LZW tidak selalu efisien untuk metadata EXIF, yang seringkali memiliki distribusi data yang unik. Dalam konteks metadata EXIF, kompresi yang ideal adalah yang dapat mengurangi ukuran metadata tanpa mengubah format file JPEG secara keseluruhan (Agnihotri et al., 2024; Hwang et al., 2021). Dengan demikian, pengguna tetap dapat mengakses dan menggunakan gambar seperti biasa tanpa perlu mengonversi kembali ke format asli.

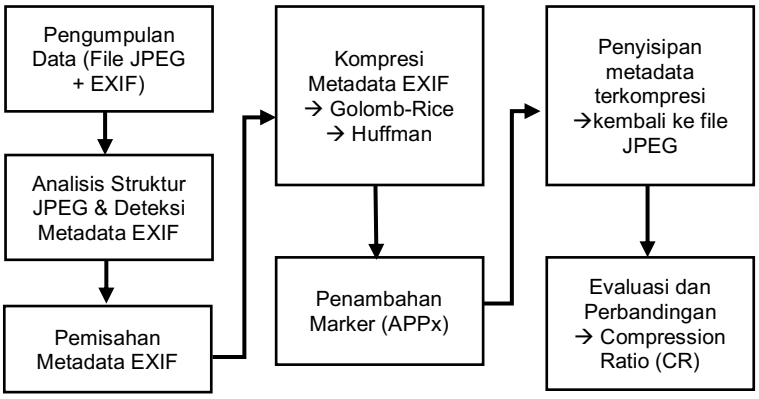
Huffman Coding dan Golomb-Rice Coding adalah dua metode kompresi data yang memiliki prinsip kerja berbeda. Golomb-Rice Coding mengandalkan pembagian nilai dengan parameter $m = 2^k$, di mana data dikodekan berdasarkan *quotient* dan *remainder*, sehingga lebih efisien untuk data dengan pola eksponensial yang sering mengandung nilai kecil berulang (Nousheen & Kumar, 2023; Žalik et al., 2021). Namun, keefektifan Golomb-Rice sangat bergantung pada pemilihan parameter k, yang jika tidak optimal, justru dapat menghasilkan ukuran kompresi yang lebih besar dibandingkan data aslinya (Himalyan & Gupta, 2023). Sebaliknya, Huffman Coding selalu memberikan kompresi paling optimal untuk data dengan distribusi acak (Kadhim et al., 2024). Huffman Coding bekerja dengan membangun pohon Huffman berdasarkan frekuensi kemunculan simbol, di mana simbol yang lebih sering muncul akan diberikan kode yang lebih pendek, sehingga cocok untuk data dengan distribusi yang tidak merata, seperti dalam format JPEG, MP3, dan ZIP. Oleh karena itu, pemilihan metode kompresi bergantung pada pola distribusi data yang dikompresi (Kumar & Kumar, 2021).

Ide untuk menerapkan kompresi Golomb-Rice dan Huffman pada metadata EXIF dalam file JPEG muncul untuk melihat performa metode klasik tersebut mengurangi ukuran file tanpa mengubah struktur file atau merusak metadata serta memperhatikan struktur file JPEG. Pendekatan ini memastikan bahwa metadata *non-editable*, yang kritis untuk integritas file, tetap tidak tersentuh. Selain itu, kompresi Golomb-Rice dan Huffman dapat diintegrasikan dengan teknik kompresi JPEG yang ada, sehingga menghasilkan gambaran yang komprehensif untuk optimasi ukuran

file dan penerapan metode klasik (Jamil, 2024). Urgensi penelitian ini terletak pengujian kompresi Golomb-Rice dan Huffman untuk mengoptimalkan ukuran file JPEG tanpa mengorbankan metadata EXIF. Oleh karena itu, penelitian ini menguji dan membandingkan efektivitas kedua metode ini dalam mengompresi metadata JPEG. Studi ini menjadi penting untuk mengetahui apakah pendekatan tradisional masih relevan atau apakah diperlukan modifikasi dan adaptasi algoritma agar sesuai dengan karakteristik metadata yang lebih spesifik. Dengan memahami keunggulan dan keterbatasan masing-masing metode.

2. METODE PENELITIAN

Penyajian diagram alur metode penelitian yang menggambarkan proses secara sistematis mulai dari pengumpulan data hingga evaluasi hasil. Penelitian ini diawali dengan pengumpulan file JPEG yang memiliki variasi ukuran metadata EXIF. Selanjutnya dilakukan analisis terhadap struktur internal file JPEG untuk mengidentifikasi letak metadata. File JPEG memiliki struktur yang terdiri dari berbagai segmen yang diawali dengan *marker* FF diikuti oleh *Byte* lain yang menentukan jenis segmen. Metadata EXIF terletak di antara *marker* FF D8 (*Start of Image*, SOI) dan sebelum *marker* FF DB (*Start of Quantization Table*, DQT) (Azeem & Fatima, 2022; Taha et al., 2022). Metadata EXIF kemudian diekstraksi dan dikompresi menggunakan dua algoritma kompresi klasik berbeda, yaitu Golomb-Rice dan Huffman. Hasil kompresi disisipkan kembali ke dalam file JPEG dengan penambahan marker khusus masing-masing algoritma untuk menandai metode kompresi yang digunakan. Langkah terakhir adalah evaluasi efektivitas kompresi berdasarkan rasio kompresi (*Compression Ratio*) untuk menilai keunggulan masing-masing metode. Diagram berikut memvisualisasikan keseluruhan proses tersebut secara runtut.



Gambar 1 Tahapan Penelitian

2.1 Identifikasi file Input

Identifikasi file input dilakukan pada tahap awal, dimulai dari pengumpulan data, analisis struktur JPEG, dan deteksi metadata EXIF hingga pemisahan metadata EXIF, sebagaimana terlihat pada Gambar 1. Bagian ini bertujuan untuk memastikan bahwa hanya metadata EXIF yang akan dikompresi, sedangkan bagian lain dari file JPEG tetap utuh agar kompatibilitas format tidak terganggu.

Langkah-langkah identifikasi file input adalah sebagai berikut:

1. Membaca file JPEG dalam format heksadesimal.
2. Mencari marker dari FF D8 hingga FF DB yang pertama muncul.
3. Mendeteksi penanda kompresi Golomb-Rice atau Huffman. Jika marker sudah ada, proses kompresi tidak dilakukan; jika marker belum ada, maka kompresi dilanjutkan.
4. Menyalin data yang berada di antara FF D8 hingga sebelum FF DB yang pertama.

2.2 Kompresi Terhadap Metadata EXIF

Setelah metadata *editable* diekstrak, langkah selanjutnya adalah menerapkan kompresi menggunakan algoritma Golomb-Rice atau Huffman. Proses kompresi Golomb-Rice terdiri dari beberapa tahapan (Boddu & Mandal, 2022). Pertama, metadata dikonversi ke dalam bentuk biner. Selanjutnya, ditentukan parameter m, seperti pada Pers. (1), di mana k merupakan parameter yang dioptimalkan. Data kemudian dibagi menjadi *quotient* (q) dan *remainder* (r) menggunakan rumus pada Pers. (2) dan (3). *Quotient* di-*encode* menggunakan *unary code*, sedangkan *remainder* di-*encode* menggunakan *binary code* sepanjang k bit. Setelah encoding selesai, data diintegrasikan kembali, diberikan penanda APP dan atribut kompresi Golomb-Rice pada awal metadata, dan disimpan sebagai metadata terkompresi.

$$m = 2^k \tag{1}$$

$$q = \lfloor n/m \rfloor \tag{2}$$

$$r = n \bmod m \tag{3}$$

Proses kompresi Huffman melibatkan beberapa langkah (Aria & Sanjaya, 2018). Pertama, dilakukan perhitungan frekuensi kemunculan setiap simbol pada metadata. Berdasarkan frekuensi tersebut, dibangun pohon Huffman untuk menentukan kode unik bagi setiap simbol. Metadata kemudian di-*encode* menggunakan kode Huffman yang telah ditetapkan. Hasil encoding diberikan penanda APP dan atribut kompresi Huffman pada awal metadata, kemudian disimpan sebagai data terkompresi.

2.3 Evaluasi

Untuk menilai efektivitas metode yang digunakan, evaluasi dilakukan dengan menggunakan perbandingan ukuran metadata file awal dan file hasil kompresi yaitu *Compression Ratio* (CR). Metrik ini mengukur seberapa banyak data dikompresi dibandingkan dengan ukuran aslinya, seperti yang ditunjukkan pada Pers. (4) (Boddu & Mandal, 2022; Hwang et al., 2021). Jika CR > 1 menunjukkan kompresi efektif (file lebih kecil setelah dikompresi) dan jika CR < 1 berarti ukuran file hasil kompresi malah lebih besar, yang menunjukkan kompresi tidak efektif.

$$CR = \frac{Ukuran\ Data\ Asli}{Ukuran\ Data\ Terkompresi} \tag{4}$$

3. HASIL DAN PEMBAHASAN

Hasil evaluasi kompresi menggunakan algoritma Golomb-Rice dan Huffman pada tiga file gambar yang diuji, dengan daftar file ditunjukkan pada Tabel 1, adalah sebagai berikut:

1) File AMAN_O.JPG
   a) Hasil kompresi Golomb-Rice dari 18 *Byte* → 25 *Byte* justru lebih besar dibandingkan ukuran aslinya. Ini menunjukkan bahwa Golomb-Rice kurang efisien untuk ukuran data yang sangat kecil atau data dengan distribusi nilai yang tidak sesuai untuk skema kompresi ini.
   b) Hasil Kompresi Huffman dari 18 *Byte* → 34 *Byte* tidak efektif, karena ukuran file setelah kompresi justru bertambah. Hal ini menunjukkan bahwa data awal mungkin sudah sangat terkompresi atau memiliki struktur yang tidak menguntungkan untuk Huffman.
2) File ME_O.JPG
   a) Hasil kompresi Golomb-Rice dari 560 *Byte* → 172 *Byte* cukup signifikan, mengurangi ukuran file menjadi sekitar 30.7% dari ukuran aslinya. Ini menunjukkan bahwa file memiliki pola yang dapat dimanfaatkan oleh Golomb-Rice untuk menghilangkan redundansi.

b) Hasil Kompresi Huffman dari 560 *Byte* → 347 *Byte* efektif, karena ada pengurangan ukuran sekitar 22%. Ini menunjukkan bahwa algoritma Huffman mampu mengoptimalkan representasi data dengan baik pada file.

3) File KUNO_O.JPG
   a) Hasil kompresi Golomb-Rice dari 616 *Byte* → 167 *Byte*. Setelah dikompresi hanya sekitar 27.1% dari ukuran awal. Ini menunjukkan bahwa data dalam file ini sangat sesuai untuk dikompresi dengan metode Golomb-Rice.
   b) Hasil Kompresi Huffman dari 616 *Byte* → 385 *Byte* efektif, dengan pengurangan sekitar 40%. Ini menunjukkan bahwa file ini memiliki banyak redundansi yang bisa dikompresi dengan Huffman.

Tabel 1 File Gambar JPEG yang diuji

No.	Nama file	File Gambar	Size file	Size EXIF
1	AMAN_O.JPG		59.8 KB	18 Byte
2	ME_O.JPG		3.45 KB	560 Byte
3	KUNO_O.JPG		42.2 KB	616 Byte

Compression Ratio (CR) dari ketiga file gambar yang telah dikompresi menggunakan Golomb-Rice dan Huffman terdapat perbedaan hasil dari panjang metadata yang digunakan untuk kompresi. Hasil CR dari ketiga file gambar yang diuji dapat dilihat pada Tabel 2, sebagai berikut.

1) Golomb-rice:
   AMAN_O.JPG → $CR = \frac{18}{25} = 0.72$ → (tidak efektif)
   ME_O.JPG→ $CR = \frac{560}{172} = 3.26$ → (efektif)
   KUNO_O.JPG→ $CR = \frac{616}{167} = 3.69$ → (efektif)
2) Huffman:
   AMAN_O.JPG → $CR = \frac{18}{34} = 0.53$ → (tidak efektif)
   ME_O.JPG→ $CR = \frac{560}{347} = 1.61$ → (efektif)
   KUNO_O.JPG→ $CR = \frac{616}{385} = 1.60$ → (efektif)

Tabel 2 Perbandingan *Compression Ratio*

Kompresi Golomb-Rice	Kompresi Huffman	CR Golomb-Rice	CR Huffman	Selisih CR
25 Byte	34 Byte	0.72	0.53	0.19
172 Byte	347 Byte	3.26	1.61	1.65
167 Byte	365 Byte	3.69	1.60	2.09

Dari hasil ini terlihat bahwa Golomb-Rice lebih unggul dibandingkan Huffman, terutama pada data dengan CR tinggi seperti ME_O.JPG dan KUNO_O.JPG. Namun pada file Aman_O.JPG, kedua metode tidak memberikan hasil yang baik karena metadata yang digunakan sangat kecil jumlahnya sehingga kompresi menjadi tidak efisien. Berdasarkan tabel perbandingan

Compression Ratio pada Tabel 2, maka dapat dijabarkan kelebihan dan keterbatasan dari metode Golomb-Rice dan Huffman pada Tabel 3.

Tabel 3 Kelebihan dan Keterbatasan Golomb-Rice dan Huffman

No.	Algoritma	Kelebihan	Keterbatasan
1	Golomb-Rice	- Efisiensi tinggi pada data berukuran sedang hingga besar. - Struktur sederhana	- Kurang efektif pada metadata kecil - Sensitif terhadap distribusi data
2	Huffman	- Kompresi stabil di berbagai ukuran data - Optimal berdasarkan frekuensi simbol	- Kurang efektif untuk data kecil atau terbatas jenis simbol - Membutuhkan tambahan header/tabel

3.1 Kelayakan Kompresi Terhadap Hasil Uji

Untuk menentukan kelayakan pemanfaatan media penyimpanan dengan kompresi pada file gambar, dapat melihat rasio antara ukuran file asli dan ukuran metadata EXIF. Perhitungan persentase metadata terhadap ukuran file dituliskan pada Pers. (5).

$$\text{Persentase Metadata} = \frac{Metada\ Exif}{Ukuran\ File} * 100\% \quad (5)$$

1) AMAN_O.JPG 59.8 KB → Metadata: 18 *Byte* → $= \frac{18}{61235} * 100\% = 0.03\%$
2) ME_O.JPG 3.45 KB → Metadata: 560 *Byte* → $= \frac{560}{3532} * 100\% = 15.86\%$
3) KUNO_O.JPG 42.2 KB → Metadata: 616 *Byte* →$= \frac{616}{43264} * 100\% = 1.42\%$

Pada gambar dengan ukuran besar AMAN_O.JPG (59.8 KB) dan KUNO_O.JPG (42.2 KB), metadata hanya mengambil porsi kecil dari total file (0.03% dan 1.42%), sehingga kompresi metadata saja tidak memberikan penghematan signifikan. Pada gambar kecil ME_O.JPG (3.45 KB), metadata diambil 15.86% dari ukuran file, sehingga kompresi metadata bisa lebih berarti dalam menghemat ruang penyimpanan.

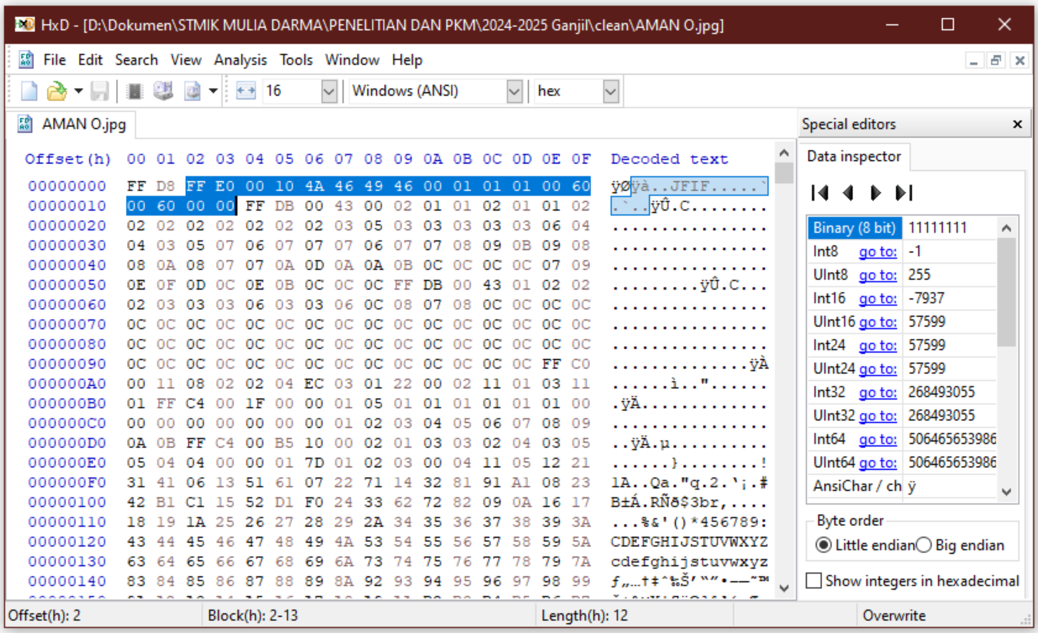
Jika tujuan utama adalah menghemat ruang penyimpanan secara keseluruhan, kompresi metadata saja tidak terlalu efektif untuk gambar berukuran besar. Lebih baik menerapkan kompresi ke seluruh gambar. Namun, untuk gambar kecil yang memiliki metadata besar, kompresi metadata bisa berkontribusi dalam mengurangi ukuran file. Jika metadata EXIF penting untuk dipertahankan, maka teknik kompresi metadata dapat berguna untuk mengurangi *overhead* penyimpanan. Implikasi praktis dari penelitian ini adalah potensi penerapan kompresi metadata EXIF menggunakan Golomb-Rice dan Huffman dalam sistem pengelolaan arsip foto digital, terutama untuk menghemat ruang penyimpanan tanpa memengaruhi kualitas visual file JPEG.

3.2 Analisis Data

Pada tahap awal, dilakukan ekstraksi metadata EXIF dari file JPEG menggunakan aplikasi HxD berguna untuk analisis Metadata Hex secara (Sunardi et al., 2020). File gambar yang digunakan untuk analisa sebanyak tiga file gambar, yaitu file yang bernama AMAN_O.JPG ukuran file 59.8 KB, ME_O.JPG ukuran file 3.45 KB, dan KUNO_O.JPG 42.2 KB. Struktur file dari semua file JPEG dianalisis untuk menentukan bagian metadata yang dapat diedit tanpa mengubah struktur utama file.

Berdasarkan hasil analisis, metadata yang dapat dikompresi terletak pada segmen dengan alamat offset 00000002h, yang diawali dengan FF E0, yaitu marker untuk segmen APP0 (JFIF header). Metadata yang dipilih dapat dilihat pada Gambar 2, dengan seleksi berwarna biru. Metadata ini berada setelah marker FF D8 (Start of Image) yang menandai awal file JPEG, hingga

marker FF DB (Define Quantization Table) yang menandai awal dari tabel kuantisasi yang bersifat tetap dan tidak dapat diubah (Hazel, 2008). Untuk perhitungan kompresi berikutnya, hanya digunakan metadata dari file AMAN_O.JPG, karena jumlah data heksadesimalnya lebih pendek dibandingkan file lainnya, yaitu 18 Byte atau 32 karakter heksadesimal. Metadata yang diperoleh dari setiap file JPEG input adalah sebagai berikut.



Gambar 2 Pembacaan file Gambar AMAN_O.JPG

1) File AMAN_O.JPG length 18 Byte:
   FF E0 00 10 4A 46 49 46 00 01 01 01 00 60 00 60 00 00
2) File ME_O.JPG length 560 Byte:
   FF E0 00 10 4A 46 49 46 00 01 01 01 00 60 00 60 00 00 FF E2 02 1C 49 43 43 5F 50 52 4F 46 49 4C 45 00 01 01 00 00 02 0C 6C 63 6D 73 02 10 00 00 6D 6E 74 72 52 47 42 20 58 59 5A 20 07 DC 00 01 00 19 00 03 00 29 00 39 61 63 73 70 41 50 50 4C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F6 D6 00 01 00 00 00 00 D3 2D 6C 63 6D 73 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 64 65 73 63 00 00 00 FC 00 00 00 5E 63 70 72 74 00 00 01 5C 00 00 00 0B 77 74 70 74 00 00 01 68 00 00 00 14 62 6B 70 74 00 00 01 7C 00 00 00 14 72 58 59 5A 00 00 01 90 00 00 00 14 67 58 59 5A 00 00 01 A4 00 00 00 14 62 58 59 5A 00 00 01 B8 00 00 00 14 72 54 52 43 00 00 01 CC 00 00 00 40 67 54 52 43 00 00 01 CC 00 00 00 40 62 54 52 43 00 00 01 CC 00 00 00 40 64 65 73 63 00 00 00 00 00 00 00 03 63 32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 74 65 78 74 00 00 00 00 46 42 00 00 58 59 5A 20 00 00 00 00 00 00 F6 D6 00 01 00 00 00 00 D3 2D 58 59 5A 20 00 00 00 00 00 00 03 16 00 00 03 33 00 00 02 A4 58 59 5A 20 00 00 00 00 00 00 6F A2 00 00 38 F5 00 00 03 90 58 59 5A 20 00 00 00 00 00 00 62 99 00 00 B7 85 00 00 18 DA 58 59 5A 20 00 00 00 00 00 00 24 A0 00 00 0F 84 00 00 B6 CF 63 75 72 76 00 00 00 00 00 00 00 1A 00 00 00 CB 01 C9 03 63 05 92 08 6B 0B F6 10 3F 15 51 1B 34 21 F1 29 90 32 18 3B 92 46 05 51 77 5D ED 6B 70 7A 05 89 B1 9A 7C AC 69 BF 7D D3 C3 E9 30 FF FF
3) File KUNO_O.JPG length 616 Byte:
   FF E0 00 10 4A 46 49 46 00 01 02 00 00 01 00 01 00 00 FF ED 00 36 50 68 6F 74 6F 73 68 6F 70 20 33 2E 30 00 38 42 49 4D 04 04 00 00 00 00 00 19 1C 02 67 00 14 73 62 4F 62 61 45 70 36 34 49 31 36 33 51 45 48 51 46 43 68 00 FF E2 02 1C 49 43 43 5F 50 52 4F 46 49

4C 45 00 01 01 00 00 02 0C 6C 63 6D 73 02 10 00 00 6D 6E 74 72 52 47 42 20 58 59 5A 20 07 DC 00 01 00 19 00 03 00 29 00 39 61 63 73 70 41 50 50 4C 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F6 D6 00 01 00 00 00 00 D3 2D 6C 63 6D 73 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A 64 65 73 63 00 00 00 FC 00 00 00 5E 63 70 72 74 00 00 01 5C 00 00 00 0B 77 74 70 74 00 00 01 68 00 00 00 14 62 6B 70 74 00 00 01 7C 00 00 00 14 72 58 59 5A 00 00 01 90 00 00 00 14 67 58 59 5A 00 00 01 A4 00 00 00 14 62 58 59 5A 00 00 01 B8 00 00 00 14 72 54 52 43 00 00 01 CC 00 00 00 40 67 54 52 43 00 00 01 CC 00 00 00 40 62 54 52 43 00 00 01 CC 00 00 00 40 64 65 73 63 00 00 00 00 00 00 00 03 63 32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 74 65 78 74 00 00 00 00 46 42 00 00 58 59 5A 20 00 00 00 00 00 00 F6 D6 00 01 00 00 00 00 D3 2D 58 59 5A 20 00 00 00 00 00 00 03 16 00 00 03 33 00 00 02 A4 58 59 5A 20 00 00 00 00 00 00 6F A2 00 00 38 F5 00 00 03 90 58 59 5A 20 00 00 00 00 00 00 62 99 00 00 B7 85 00 00 18 DA 58 59 5A 20 00 00 00 00 00 00 24 A0 00 00 0F 84 00 00 B6 CF 63 75 72 76 00 00 00 00 00 00 00 1A 00 00 00 CB 01 C9 03 63 05 92 08 6B 0B F6 10 3F 15 51 1B 34 21 F1 29 90 32 18 3B 92 46 05 51 77 5D ED 6B 70 7A 05 89 B1 9A 7C AC 69 BF 7D D3 C3 E9 30 FF

3.3 Proses Kompresi dan Penyisipan Marker Golomb-Rice

Setelah metadata *editable* diekstrak, langkah berikutnya adalah melakukan kompresi menggunakan algoritma Golomb-Rice. Tahapan ini dimulai dengan konversi nilai heksadesimal ke biner. Sebagai contoh, metadata dari file gambar AMAN_O.JPG yang dipilih, yaitu FF E0 00 10 4A 46 49 46 00 01 01 01 00 60 00 60 00 00, dikonversi menjadi biner, dan hasilnya ditampilkan pada Tabel 4. Setelah data biner diperoleh, langkah penting berikutnya adalah memilih parameter k untuk menentukan nilai m. Pada Golomb-Rice, m biasanya dipilih sebagai pangkat dari 2 (Pers. 1) (Hamilton et al., 2023). Dalam perhitungan ini dipilih $k = 4$, sehingga $m = 2^4 = 16$. Nilai m ini digunakan untuk menghitung *Quotient* (q) dan *Remainder* (r) dengan rumus pada Pers. (2) dan (3). Misalnya, untuk nilai metadata pertama $n_1 = 255$, diperoleh $q = \left\lfloor \frac{255}{16} \right\rfloor = 15$ dan $r = 255 \bmod 16 = 15$.

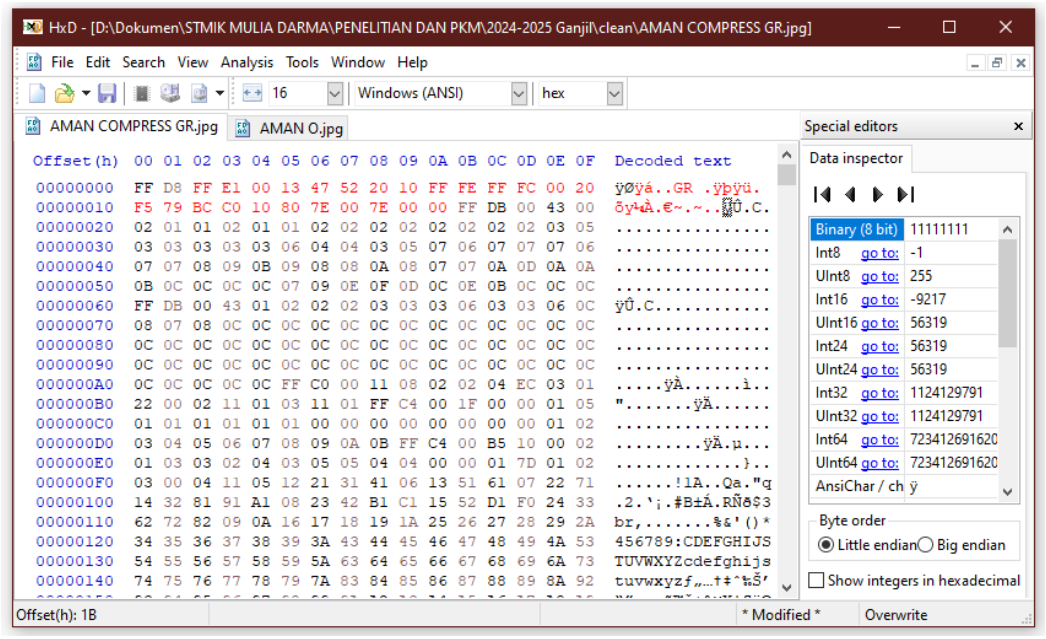
Tabel 4 Konversi Heksadesimal, Desimal ke Biner

No.	Hex	Decimal	Bin
1	FF	255	11111111
2	E0	224	11100000
3	00	0	00000000
4	10	16	00010000
5	4A	74	01001010
6	46	70	01000110
7	49	73	01001001
8	46	70	01000110
9	00	0	00000000
10	01	1	00000001
11	01	1	00000001
12	01	1	00000001
13	00	0	00000000
14	60	96	01100000
15	00	0	00000000
16	60	96	01100000
17	00	0	00000000
18	00	0	00000000

Proses yang sama diterapkan pada seluruh metadata. Nilai 255 (FF) dikompresi menjadi q = 15, yang di-*encode* dalam bentuk unary sebagai 1111111111111110, dan r = 15 di-*encode* dalam bentuk biner sebagai 1111. Kedua kode ini kemudian digabung menjadi 11111111111111101111. Encoding setiap pasangan (q, r) untuk seluruh metadata ditunjukkan pada Tabel 5. Seluruh *encoded bits* dari setiap pasangan digabungkan, dibagi menjadi 8-bit, dan dikonversi kembali ke bentuk heksadesimal. Jika terdapat kekurangan bit di akhir, ditambahkan padding 0. Hasil akhir penggabungan unary dan binary diubah menjadi heksadesimal: FF FE FF FC 00 20 F5 79 BC C0 10 80 7E 00 7E 00 00. Pada hasil akhir ini diberikan penanda (*marker*) Golomb-Rice agar file dapat dikenali untuk proses dekompresi selanjutnya, sebagaimana ditunjukkan pada Tabel 6. Proses penggantian metadata EXIF dengan hasil kompresi Golomb-Rice divisualisasikan pada Gambar 3 (Harvey, 2025).

Tabel 5 Pembentukkan Unary dan Binary

No.	q	Unary	r	Binary	Hasil
1	15	1111111111111110	15	1111	11111111111111101111
2	14	11111111111110	0	0000	11111111111100000
3	0	0	0	0000	00000
4	1	10	0	0000	100000
5	4	11110	10	1010	111101010
6	4	11110	6	0110	111100110
7	4	11110	9	1001	111101001
8	4	11110	6	0110	111100110
9	0	0	0	0000	00000
10	0	0	1	0001	00001
11	0	0	1	0001	00001
12	0	0	1	0001	00001
13	0	0	0	0000	00000
14	6	1111110	0	0000	11111100000
15	0	0	0	0000	00000
16	6	1111110	0	0000	11111100000
17	0	0	0	0000	00000
18	0	0	0	0000	00000



Gambar 3 Penggantian Metadata EXIF Dengan Hasil Kompresi Golomb-Rice

Tabel 6 Marker Golomb-Rice

Deskripsi	Heksadesimal
Marker APP1	FFE1
Panjang Data 19 Byte (2 Byte identifier + hasil kompresi)	0013
Identifier "GR " (Golomb-Rice, k=4, m=16)	47 52 20 10
Hasil kompresi Golomb-Rice	FF FE FF FC 00 20 F5 79 BC C0 10 80 7E 00 7E 00 00
Deskripsi	Heksadesimal

3.4 Proses Kompresi dan Penyisipan Marker Huffman

Langkah-langkah kompresi Huffman dimulai dengan mengonversi data heksadesimal FF E0 00 10 4A 46 49 46 00 01 01 01 00 60 00 60 00 00 menjadi bentuk biner. Selanjutnya, dilakukan perhitungan frekuensi kemunculan setiap byte, dan hasilnya disajikan pada Tabel 7. Tahap penting berikutnya adalah pembuatan pohon Huffman, yang membentuk pemetaan untuk proses kompresi. Byte dengan frekuensi paling kecil digabungkan, setiap simpul baru memiliki bobot total dari dua simpul digabungkan, dengan cabang kiri diberikan kode 0 dan cabang kanan diberikan kode 1. Hasil *encoding* dari setiap byte ditampilkan pada Tabel 8, kemudian setiap kode disusun per 8-bit. Jika kode kurang dari 8-bit, dilakukan *padding* 0 hingga mencukupi satu byte, kemudian dikonversi ke bentuk heksadesimal.

Tabel 7 Frekuensi Kemunculan *Byte*

Byte	FF	E0	00	10	4A	46	49	01	60
Frek	1	1	6	1	1	2	1	3	2

Tabel 8 Simbol Kode

Byte	Huffman Code	Panjang
0	0	1-bit
1	10	2-bit
46	1100	4-bit
60	1101	4-bit
FF	11100	5-bit
E0	11101	5-bit
10	11110	5-bit
4A	111110	6-bit
49	111111	6-bit

Tabel 9 Penyusunan data Huffman dalam APP1 (FFE1)

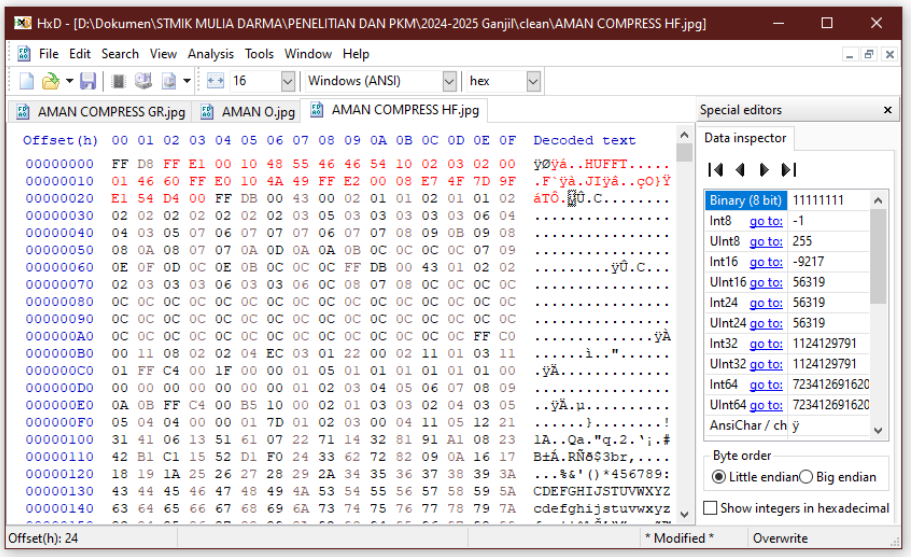
Deskripsi	Heksadesimal
Marker APP1	FFE1
Panjang Data (18 Byte)	0012
Identifier "HUFF"	48554646
Jenis Huffman DC Chrominance → 010100	54
Jumlah kode untuk setiap panjang → 010000	10
Daftar panjang kode	020302
Daftar simbol sesuai panjang kode	00 01 46 60 FF E0 10 4A 49

Untuk memungkinkan metadata JPEG dikompresi dan didekompresi dengan benar, penyisipan hasil kompresi Huffman dan tabel Huffman dilakukan ke dalam bagian metadata JPEG. Marker khusus digunakan untuk menandai bagian metadata, yaitu FF E1 sebagai Marker APP1 untuk tabel Huffman (Tabel 9) dan FF E2 sebagai Marker APP2 untuk data Huffman yang telah dikompresi (Tabel 10) (Accusoft, 2021; Harvey, 2025). Hasil penyusunan byte Huffman setelah proses pengurutan berjumlah 12 Byte, yaitu FF E2 00 0A E7 4F 7D 9F E1 54 D4 00. Hasil akhir

setelah digabungkan dengan struktur metadata JPEG menjadi 34 Byte, yaitu FF D8 FF E1 00 12 48 55 46 46 54 10 02 03 02 00 01 46 60 FF E0 10 4A 49 FF E2 00 08 E7 4F 7D 9F E1 54 D4 00 FF DB. Visualisasi proses penggantian metadata EXIF dengan hasil kompresi Huffman dapat dilihat pada Gambar 4.

Tabel 10 Hasil Kompresi ke APP2 (FFE2)

Deskripsi	Heksadesimal
Marker APP2 2 Byte	FFE2
Panjang Data (Hasil kompresi + 2 Byte panjang)	000A
Data hasil kompresi 8 Byte	E7 4F 7D 9F E1 54 D4 00



Gambar 4 Penggantian Metadata EXIF Dengan Hasil Kompresi Huffman

3.5 Hasil Kompresi ME_O.JPG DAN KUNO_O.JPG

Berikut ditampilkan hasil dari file gambar ME_O.JPG dan KUNO_O.JPG yang telah disiapkan untuk proses kompresi menggunakan algoritma Golomb-Rice dan Huffman. Pada hasil kompresi Golomb-Rice, marker ditandai dengan warna merah untuk mempermudah identifikasi lokasi marker dalam metadata. Sedangkan pada hasil kompresi Huffman, marker APP1 ditandai dengan warna merah dan marker APP2 ditandai dengan warna hijau. Pemberian warna ini bertujuan untuk memudahkan pemahaman mengenai penambahan marker pada metadata setelah kompresi, sehingga mempermudah identifikasi dan proses dekompresi pada tahap berikutnya.

1) Kompresi Golomb-Rice

File KUNO_O.JPG 8 + 164 = 172 *Byte* kompresi Golomb-Rice

FF E1 00 A6 47 52 20 10 00 44 32 14 E8 52 D8 F8 24 96 6A 29 AA BB 30 61 C9 A7 6E EE 0E 1E 2E 3E 4E 6E 8E 9E BE FF 07 87 C5 E3 F2 F9 BC FE 8F 4F AB D9 ED F7 FC 1F 0F C5 F2 7D 1F 4F D5 F6 7D BF 77 DF F8 7F 17 E3 FC 9F 97 F3 FE 8F D3 FA FF 67 ED FD DF BF F8 3F 8B F8 FF 93 F9 7F 9B F9 FF A3 FA BF B3 FB 7F C9 FE 5F F4 FF C1 FF 17 FD 3F F5 7F E0 FF C5 FF 93 FF 67 FF 0F FF 37 FF 3F FF 47 FF 7F FF 8F FF D3 FF EB FF F6 7F FB FF FE 3F FF 9B FF EA FF FB 3F FF 07 FF E2 FF FD 3F FF B7 FF F8 7F FF 97 FF F9 BF FF B3 FF FB 03

File ME_O.JPG 8 + 159 = 167 *Byte* Kompresi Golomb-Rice

FF E1 00 A1 47 52 20 10 00 44 32 9D 0A 5B 1F 04 92 CD 45 35 57 66 0C 39 34 ED E0 E2 E3 E4 E8 E9 EB EF F0 78 7C 5E 3F 2F 9B CF E9 F5 7B 3D FF 07 C3 F1 7C 9F 47 D3 F5 7D 9F 6F DD F7 FE 0F C3 F8 BF 1F E4 FC BF 9F F4 7E 9F D7 FB 3F 6F EE FD FF C1 FC 5F C7 FC 9F CB FC DF CF FD 1F D5 FD 9F DB FE 4F F2 FF A7 FE 0F F8 BF E9 FF AB FF 07

FE 2F FC 9F FB 3F F8 7F F9 BF F9 FF FA 3F FB FF FC 7F FE 9F FF 5F FF B3 FF DF FF F1 FF FC DF FF 57 FF D9 FF F8 3F FF 17 FF E9 FF FD BF FF C3 FF FC BF FF CD FF FD 9F FF 6F

2) Kompresi Huffman

File KUNO_O.JPG 32 + 405 = 437 *Byte* kompresi huffman

FF E1 00 12 48 55 46 46 54 10 02 03 02 01 02 03 04 05 06 07 08 09 0f 0A 0D 0C 0E 0B FF E2 01 97 42 D3 BE 8B A1 22 90 BB EE FB FB FC 21 68 B6 24 72 01 05 88 41 60 90 08 2C BA 22 E6 83 C4 08 E4 52 2D 4A 7F EF 53 DB B8 99 BD 18 24 E4 09 38 9E 86 C8 90 A2 28 5E 24 22 6F 43 40 1B D0 90 48 18 42 D1 D7 3D B4 52 0A 09 84 E6 E4 0A 12 29 1B 43 EF BF EE B6 4D 90 92 CC 1B 9F F2 58 9A 31 18 E3 72 31 1C E9 80 D4 6B BB 62 CD F7 EF 57 1D CA C5 44 F2 16 0B 28 F3 9D 1B FF FF FF FF FF FF F0 92 C5 DF FE B0 B9 62 6C 84 96 60 FF FF FF FF FF FF FF FF FF FF FF FF 5C 90 8D 82 43 F8 5B FC DA 10 B2 C7 31 3E F3 6F FB 56 31 88 CB 13 EF 20 7E F4 27 13 56 58 9F 7B 1B FD E8 C7 18 0D 46 BF DE AF EF 42 61 80 D4 6B FD EB A7 EF 42 71 80 D4 6B FD ED 41 FB D1 8E 34 37 20 FD ED B7 FA 49 86 86 E4 1F BD B6 FF 49 38 D0 DC 83 F7 B6 DF E9 24 23 60 90 FF FF 12 11 77 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF B1 08 D8 06 27 FA 12 3B CC 06 A3 5D DF FF 09 2C 5D FF EB 0B 96 30 1A 8D 77 7F FE 2F 2F 88 8F DC BA 18 0D 46 BB BF FE 41 5D DE 20 09 FC 55 30 1A 8D 77 7F FC 9C A5 7B 56 00 FD E0 59 73 01 A8 D7 77 FF DC 8B FE 10 27 B5 26 C2 42 C3 63 98 5F FF BD 7F DB 6B 7B 6A 89 0C D4 E8 09 AD A8 49 F1 0B CC DE F6 A2 8E 78 5E E5 2A 2E 78 05 AA 9C 85 33 7B 18 6B 34 58 9A B2 C5 E6 05 35 7A 97 63 6B B6 4A 6A 16 2C B0 B6 2D 14 30 84 21

File ME_O.JPG 32 + 353 = 385 *Byte* kompresi huffman

FF E1 00 12 48 55 46 46 54 10 02 03 02 01 02 03 04 05 06 07 08 09 0f 0A 0D 0C 0E 0B FF E2 01 63 21 3C E1 27 C8 C9 44 7C 21 0C F9 FC 42 78 DB 40 64 A2 71 39 04 94 69 08 8C 94 43 24 E1 0F B6 33 19 B8 D6 63 AD 0F 9A C6 F1 84 C6 A3 49 84 D6 D1 74 52 3E DD 8B 0F 86 0A EE DA AD CA 30 37 31 CC 90 25 24 3F FF FF FF FF FF FF 11 AC 7C 3F D6 73 56 31 9B 8D 66 3B FF FF FF FF FF FF FF FF FF FF FF FB E6 46 8C 71 BB F1 0F F4 78 6E 65 8D 61 7C 10 7F DE B1 8C 26 58 5F 03 5F F0 23 69 BD 65 85 F0 60 FF 02 63 51 74 52 3F F0 57 F0 23 62 2E 8A 47 FE 0F 97 E0 46 D4 5D 14 8F FC 1E AF F8 13 1A 84 8D 27 7C 06 3F CA 6C 42 46 93 BE 03 1F E5 36 A1 23 49 DF 01 8F F2 99 1A 31 C6 EF FF EE 37 39 BF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FE C2 34 62 EC 2F F2 32 6F A2 E8 A4 7D BF FF 11 AC 7C 3F D6 73 56 45 D1 48 FB 7F FF 70 1F DC E7 7D AF 92 2E 8A 47 DB FF F3 13 ED F7 2E 24 FB 95 45 D1 48 FB 7F FE 6D 52 BD EB 17 4F 05 D6 7D 17 45 23 ED FF FB 49 FF C4 B9 7B D3 18 8D CC 46 35 87 FF F8 3F F8 DE C0 6A B8 DD 45 36 B9 BD BD 11 85 C2 04 40 07 AE 26 80 81 AA 55 CD 05 DC F5 4D 23 A2 03 18 8B 3C 58 DE B2 C7 E8 BA 9E 82 9F 60 DF 19 A9 E8 98 B2 CE 1B 9E 29 D1 08 42

4. KESIMPULAN

Berdasarkan hasil kompresi yang diperoleh, metode Golomb-Rice menunjukkan rasio kompresi (CR) yang lebih tinggi dibandingkan metode Huffman pada sebagian besar file, terutama untuk file dengan ukuran metadata yang besar. Namun, untuk file dengan metadata kecil, kedua metode tidak memberikan kompresi yang signifikan. Hal ini menunjukkan bahwa efektivitas Golomb-Rice lebih unggul dalam kondisi tertentu, sedangkan Huffman masih memberikan hasil kompresi yang lebih stabil di berbagai skenario.

Metadata EXIF memiliki kontribusi yang berbeda terhadap ukuran file tergantung pada ukuran gambar. Pada gambar dengan ukuran besar, metadata hanya mengambil porsi kecil dari keseluruhan file, sehingga kompresinya tidak memberikan penghematan yang signifikan. Sebaliknya, pada gambar kecil, metadata dapat mencapai lebih dari 15% dari total ukuran file, sehingga kompresi metadata menjadi lebih relevan untuk mengurangi overhead penyimpanan. Selain itu pemanfaatan metadata EXIF sangat tepat untuk digunakan sebagai bagian kompresi yang mana dalam file JPEG memilki metadata yang tidak bisa diubah.

Dengan kemajuan teknologi penyimpanan dan kompresi gambar saat ini, metode Golomb-Rice dan Huffman masih memiliki manfaat dalam skenario tertentu. Namun, dengan meningkatnya efisiensi algoritma kompresi berbasis JPEG 2000, WebP, dan AVIF, kompresi metadata EXIF

saja kurang efektif untuk mengoptimalkan ruang penyimpanan secara keseluruhan. Oleh karena itu, dalam aplikasi modern, kombinasi metode kompresi metadata dengan algoritma kompresi gambar yang lebih canggih lebih disarankan untuk efisiensi penyimpanan yang optimal. Penelitian selanjutnya berdasarkan temuan ini dapat menjadi dasar untuk pengembangan metode hybrid yang menggabungkan kedua metode dan lebih adaptif dalam implementasi teknologi kompresi metadata JPEG dan file lainnya di masa depan.

DAFTAR PUSTAKA

Acharya R, S., J, S., S, S., S Aithal, V., & G S, H. (2023). Geo-Locating an Image Using EXIF Data. *International Journal of Engineering Applied Sciences and Technology*, *8*(1), 43–47. https://doi.org/10.33564/IJEAST.2023.v08i01.007

Agnihotri, S., Rameshan, R. M., Ghosal, R., & Gupta, P. (2024). Lossless Binary Image Compression Using Learned Multi-Level Dictionaries. *2024 60th Annual Allerton Conference on Communication, Control, and Computing*, 1–8. https://doi.org/10.1109/Allerton63246.2024.10735272

Ardiansyah, A., Hardi, N., & Gata, W. (2020). Identifikasi dan Recovery File JPEG dengan Metode Signature-Based Carving dalam Model Automata. *Komputika: Jurnal Sistem Komputer*, *9*(1), 75–83. https://doi.org/10.34010/komputika.v9i1.2733

Aria, M., & Sanjaya, A. (2018). Teknik Kompresi pada Transmisi Data Citra Payload KOMURINDO. *Komputika: Jurnal Sistem Komputer*, *7*(2), 103–111. https://doi.org/10.34010/komputika.v7i2.1512

Azeem, E. A., & Fatima. (2022). The Data Carving - The Art of Retrieving Deleted Data as Evidence. *International Journal for Electronic Crime Investigation*, *6*(2), 8. https://doi.org/10.54692/ijeci.2022.0602101

Boddu, M., & Mandal, S. K. (2022). Quantum-Dot Cellular Automata Based Lossless CFA Image Compression Using Improved and Extended Golomb-Rice Entropy Coder. *International Journal of Intelligent Engineering and Systems*, *15*(2), 12–25. https://doi.org/10.22266/ijies2022.0430.02

Cai, S., Liang, X., Cao, S., Yan, L., Zhong, S., Chen, L., & Zou, X. (2024). Powerful Lossy Compression for Noisy Images. *2024 IEEE International Conference on Multimedia and Expo (ICME)*, 1–6. https://doi.org/10.1109/ICME57554.2024.10687468

Cappello, F., Acosta, M., Agullo, E., Anzt, H., Calhoun, J., Di, S., Giraud, L., Grützmacher, T., Jin, S., Sano, K., Sato, K., Singh, A., Tao, D., Tian, J., Ueno, T., Underwood, R., Vivien, F., Yepes, X., Kazutomo, Y., & Zhang, B. (2025). Multifacets of Lossy Compression for Scientific Data in the Joint-Laboratory of Extreme Scale Computing. *Future Generation Computer Systems*, *163*, 107323. https://doi.org/10.1016/j.future.2024.05.022

Dhawan, S. (2011). A Review of Image Compression and Comparison of Its Algorithms. *IJECT*, *2*(1). https://www.iject.org/vol2issue1/sachindhawan.pdf

Doménech Fons, J., & Pegueroles Vallés, J. R. (2021). *Study and Development of an Autopsy Module for Automated Analysis of Image Metadata* [Universitat Politècnica de Catalunya]. https://hdl.handle.net/2117/356815

Hamilton, A., El-Hajjar, M., & Maunder, R. G. (2023). Reordered Exponential Golomb Error Correction Code for Universal Near-Capacity Joint Source-Channel Coding. *IEEE Access*, *11*, 93619–93634. https://doi.org/10.1109/ACCESS.2023.3310824

Harvey, P. (2025). JPEG Tags. In *EXIFTool*. https://EXIFtool.org/TagNames/JPEG.html

Hazel, T. (2008, September). *JPEG Marker Codes*. TechStumbler. https://techstumbler.blogspot.com/2008/09/jpeg-marker-codes.html

Himalyan, S., & Gupta, V. (2023). Golomb–Rice Coder-Based Hybrid Electrocardiogram Compression System. *ECSA 2023*, *2023*, Article ID: 10. https://doi.org/10.3390/ecsa-10-16209

Hwang, I., Yun, J., Chung, W., Lee, J., Kim, C.-G., Kim, Y., & Park, W.-C. (2021). Lossless Compression Algorithm and Architecture for Reduced Memory Bandwidth Requirement with Improved Prediction Based on the Multiple DPCM Golomb-Rice Algorithm. *Journal of Web Engineering*, *20*(6), 1813–1828. https://doi.org/10.13052/jwe1540-9589.2065

Itier, V., Puteaux, P., & Puech, W. (2022). Image Crypto-Compression. In M. Security (Ed.), *Multimedia Security 2* (pp. 91–128). Wiley. https://doi.org/10.1002/9781119987390.ch4

Jamil, S. (2024). Review of Image Quality Assessment Methods for Compressed Images. *Journal of Imaging*, *10*(5), Article ID: 113. https://doi.org/10.3390/jimaging10050113

Kadhim, D. J., Mosleh, M. F., & Abed, F. A. (2024). Exploring Text Data Compression: A Comparative Study of Adaptive Huffman and LZW Approaches. *BIO Web of Conferences*, *97*, Article ID: 00035. https://doi.org/10.1051/bioconf/20249700035

Kumar, G., & Kumar, R. (2021). Analysis of Arithmetic and Huffman Compression Techniques by Using DWT-DCT. *International Journal of Image, Graphics and Signal Processing*, *13*(4), 63–70. https://doi.org/10.5815/ijigsp.2021.04.05

Liu, X., An, P., Chen, Y., & Huang, X. (2022). An Improved Lossless Image Compression Algorithm Based on Huffman Coding. *Multimedia Tools and Applications*, *81*(4), 4781–4795. https://doi.org/10.1007/s11042-021-11017-5

Mahmood, A., & Wagner, A. B. (2023). Lossy Compression with Universal Distortion. *IEEE Transactions on Information Theory*, *69*(6), 3552–3573. https://doi.org/10.1109/TIT.2023.3247601

Mani, R. G., Parthasarathy, R., Eswaran, S., & Honnavalli, P. (2022). A Survey on Digital Image Forensics: Metadata and Image Forgeries. *WAC-2022: Workshop on Applied Computing*, 22. https://ceur-ws.org/Vol-3142/PAPER_03.pdf

Martini, M. G. (2025). Measuring Objective Image and Video Quality: On the Relationship Between SSIM and PSNR for DCT-Based Compressed Images. *IEEE Transactions on Instrumentation and Measurement*, *74*, 1–13. https://doi.org/10.1109/TIM.2025.3529045

Mills, R. (2018). *The Metadata in JPEG Files*. https://dev.exiv2.org/projects/exiv2/wiki/The_Metadata_in_JPEG_files

Nousheen, M. S., & Kumar, T. V. (2023). Implementation of Lossless Image Compression Using Fuzzy Based Modified Golomb Rice Encoding. *Journal of Engineering Sciences*, *14*(02), 242–253. https://doi.org/10.15433.JES.2023.V14I2.43P.29

Stoilov, E. P. (2022). Discovery and Analysis of EXIF Data in Images. *61 St Annual Scientific Conference - University of Ruse and Union of Scientists*, 52–58. https://conf.uni-ruse.bg/bg/docs/cp22/bp/bp-6.pdf

Sunardi, S., Riadi, I., & Akbar, M. H. (2020). Steganalisis Bukti Digital pada Media Penyimpanan Menggunakan Metode Static Forensics. *Jurnal Nasional Teknologi dan Sistem Informasi*, *6*(1), 1–8. https://doi.org/10.25077/TEKNOSI.v6i1.2020.1-8

Taha, H., Luisa, S., Nasir, V., & Editors, M. (2022). *Multimedia Forensics* (H. T. Sencar, L. Verdoliva, & N. Memon, Eds.). Springer Singapore. https://doi.org/10.1007/978-981-16-7621-5

Wijayanto, H., Riadi, I., & Prayudi, Y. (2016). Encryption EXIF Metadata for Protection Photographic Image of Copyright Piracy. *IJRCCT*, *5*(5), 237–243.

Žalik, B., Mongus, D., Žalik, K. R., Podgorelec, D., & Lukač, N. (2021). Lossless Chain Code Compression with an Improved Binary Adaptive Sequential Coding of Zero-Runs. *Journal of Visual Communication and Image Representation*, *75*, Article ID: 103050. https://doi.org/10.1016/j.jvcir.2021.103050

Zheng, C., Shrivastava, A., & Owens, A. (2023). EXIF as Language: Learning Cross-Modal Associations Between Images and Camera Metadata. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6945–6956. https://doi.org/10.1109/CVPR52729.2023.00671