Penerapan Analisis Algoritma Elias Omega Code Pada Citra Digital

Liskedame Yanti Sipayung[1], Chaston Ramotto Sinaga[2], Angelica Claudia Sagala[3].
[1,2,3,] Program Studi Teknik Informatika, STMIK Pelita Nusantara, Indonesia

ARTICLE INFORMATION

Received :February 2024
Revised: Februari 2024
Available online: April 2024

KEYWORDS

Elias Omega Code algorithm, digital image compression, compression ratio, image quality

CORRESPONDENCE

Phone: +62 895 2964 0003
E-mail: liskedamesipayung@email.com, chaston278@gmail.com, angelclaudia011000@gmail.com

ABSTRACT

The large size of digital image files is an obstacle in their storage, transmission, and processing. Digital image compression is a solution to overcome this problem. The Elias Omega Code algorithm is one of the efficient and easy-to-implement lossless compression algorithms. This research aims to analyze the Elias Omega Code algorithm on digital images. The analysis includes: implementation of the Elias Omega Code algorithm for digital image compression, evaluation of the performance of the Elias Omega Code algorithm in terms of compression ratio and image quality, comparison of the performance of the Elias Omega Code algorithm with other image compression algorithms. The results of the study show that the Elias Omega Code algorithm can produce a high compression ratio with preserved image quality. This algorithm has better performance than the Huffman and LZW algorithms, but lower than the JPEG algorithm..

PENDAHULUAN

Citra digital merupakan bagian penting dalam berbagai bidang, seperti fotografi, medis, penginderaan jauh, dan desain grafis. Namun, ukuran file citra digital seringkali besar, sehingga membutuhkan ruang penyimpanan yang besar dan waktu transfer yang lama. Hal ini menjadi kendala dalam penyimpanan, transmisi, dan pengolahan citra digital.

Kompresi citra digital adalah solusi untuk mengatasi masalah tersebut. Kompresi citra bertujuan untuk mengurangi ukuran file citra tanpa mengurangi kualitasnya secara signifikan. Berbagai teknik kompresi citra telah dikembangkan, salah satunya adalah dengan menggunakan algoritma Elias Omega Code.

Algoritma Elias Omega Code adalah algoritma kompresi lossless yang efisien dan mudah diimplementasikan. Algoritma ini bekerja dengan mengkodekan nilai-nilai numerik dengan cara yang optimal, sehingga dapat menghasilkan rasio kompresi yang tinggi.

Algoritma Elias Omega merupakan sebuah metode kompresi data yang digunakan untuk mengompresi bilangan bulat menjadi urutan bit yang efisien. Algoritma ini dikembangkan sebagai pengembangan dari algoritma Elias Gamma, dengan tujuan untuk mencapai kompresi yang lebih baik terutama pada bilangan bulat yang besar. Penggunaan kode-prefix dalam algoritma Elias Omega memungkinkan representasi bilangan bulat menggunakan jumlah bit yang lebih sedikit daripada representasi biner biasa.

Proses kompresi dengan algoritma Elias Omega dimulai dengan mengonversi bilangan bulat positif nn ke dalam bentuk biner. Langkah pertama adalah menentukan panjang biner dari bilangan nn (disebut l=⌊log⁡2(n)⌋l=⌊log2(n)⌋). Selanjutnya, bagian kiri dari biner nn (tanpa bit paling signifikan) diambil sebagai representasi gamma dari nn (disebut γ=n−2$l$$\gamma$=n−2l). Kemudian, representasi gamma ini diikuti oleh representasi biner ll dalam bentuk unary.

Algoritma Elias Omega sangat efisien untuk bilangan bulat yang besar karena panjang representasi biner dari bilangan nn berbanding logaritmik dengan nilai nn. Hal ini membuat algoritma ini cocok untuk aplikasi kompresi data di mana bilangan bulat menjadi komponen penting untuk direpresentasikan secara efisien.

Dalam praktiknya, algoritma Elias Omega dapat diterapkan pada berbagai konteks, termasuk kompresi citra digital di mana representasi efisien dari nilai intensitas piksel atau koefisien transformasi menjadi kunci untuk penghematan ruang penyimpanan dan percepatan transfer data. Selain itu, algoritma ini juga memfasilitasi implementasi dekompresi yang efisien, yang memungkinkan pemulihan nilai asli dari representasi bit yang terkompresi.

Dengan demikian, algoritma Elias Omega menjadi sebuah kontribusi penting dalam bidang kompresi data, terutama untuk pengolahan data numerik yang membutuhkan representasi biner yang efisien dari bilangan bulat besar.

METODOLOGI PENELITIAN

2.1 Citral Digital

Citra digital adalah representasi visual dari suatu objek, gambar, atau scene dalam bentuk digital yang terdiri dari himpunan piksel (titik gambar) yang tersusun dalam suatu grid atau matriks. Setiap piksel memiliki nilai numerik yang menunjukkan warna atau tingkat keabuan dari bagian gambar yang direpresentasikannya. Penjelasan tentang citra digital dapat dibagi menjadi beberapa poin utama:

1. **Piksel**: Piksel (atau pixel) merupakan unit dasar yang membentuk sebuah citra digital. Setiap piksel memiliki lokasi (koordinat) dan nilai yang menunjukkan warna atau intensitas cahaya pada posisi tersebut. Nilai piksel dapat berupa kombinasi warna dasar (misalnya, merah, hijau, dan biru dalam citra berwarna) atau tingkat keabuan (dalam citra grayscale).
2. **Resolusi**: Resolusi citra mengacu pada jumlah piksel yang membentuk citra tersebut. Resolusi diukur dalam jumlah piksel per baris dan jumlah baris per citra (misalnya, 1920x1080 piksel untuk citra HD). Resolusi yang lebih tinggi menawarkan detail yang lebih halus karena lebih banyak piksel digunakan untuk merepresentasikan gambar.
3. **Format dan Kedalaman Warna**: Citra digital dapat berupa citra berwarna (RGB) atau citra keabuan (grayscale). Citra berwarna menggunakan kombinasi tiga warna dasar (merah, hijau, dan biru) untuk menciptakan berbagai warna, sedangkan citra keabuan hanya menggunakan tingkat kecerahan untuk setiap piksel tanpa informasi warna. Kedalaman warna (bit depth) menunjukkan jumlah bit yang digunakan untuk merepresentasikan setiap piksel, yang mempengaruhi jumlah warna atau tingkat keabuan yang dapat direpresentasikan.
4. **Format File**: Citra digital disimpan dalam format file seperti JPEG, PNG, GIF, BMP, dan lainnya. Setiap format memiliki cara tersendiri untuk menyimpan informasi piksel dan metadata tambahan seperti informasi warna, kompresi, dan lainnya.
5. **Prosesing dan Pengolahan**: Pengolahan citra digital melibatkan manipulasi piksel untuk tujuan tertentu seperti penyempurnaan gambar, pengenalan pola, analisis, dan aplikasi lainnya. Prosesing dapat melibatkan filter, segmentasi, penggabungan, transformasi, dan teknik lainnya untuk memanipulasi citra.
6. **Aplikasi**: Citra digital digunakan dalam berbagai aplikasi termasuk fotografi, sinematografi, desain grafis, ilmu kedokteran (misalnya, citra medis), ilmu forensik, pemrosesan gambar satelit, pengenalan wajah, dan banyak lagi. Penggunaan citra digital terus berkembang seiring dengan kemajuan teknologi digital.
7. **Kompresi dan Penyimpanan**: Kompresi citra digital penting untuk mengurangi ukuran file tanpa mengorbankan kualitas gambar. Teknik kompresi seperti JPEG, PNG, atau metode khusus lainnya digunakan untuk mengompresi citra digital agar dapat disimpan dan ditransmisikan secara efisien.

Citra digital memiliki peran yang sangat penting dalam dunia digital modern, memungkinkan representasi visual dari dunia nyata dalam format yang dapat diolah, disimpan, dan dibagikan dengan mudah. Pemahaman yang baik tentang citra digital memungkinkan pengembangan aplikasi yang inovatif dan solusi yang efektif dalam berbagai bidang.

2.2 Algoritma Elias Omega

Elias Omega Code merupakan salah satu algoritma kompresi berjenis lossless yang diperkenalkan oleh Peter Elias pada tahun 1975. Terdapat 3 kode Elias, yaitu Elias Gamma, Elias Delta, dan Elias Omega . yang mana Algoritma Elias Omega Code ini berbeda dengan algoritma Elias lainnya, algoritma ini mempunyai cara rekusif untuk pengkodean awal dengan (prefix) untuk itu algoritma ini disebut sebagai kode Elias rekursif[5].

Tabel 1. Kode Elias Omega Code

Nilai n	Kode *Elias Omega*	Nilai n	Kode *Elias Omega*
1	0	9	1110010
2	100	10	1110100
3	110	11	1110110
4	101000	12	1111000
5	101010	13	1111010
6	101100	14	1111100
7	101110	15	1111110
8	1110000	16	10100100000

HASIL DAN PEMBAHASAN

3.1 Analisa

Gambar merupakan sebuah media yang biasa digunakan untuk berkomunikasi juga digunakan sebagai alat pengungkapan, ilustrasi, ingatan (memorize) dan lain sebagainya. Masalah yang kerap kali terjadi adalah gambar yang terkadang memiliki ukuran yang besar, oleh sebab itu membutuhkan penyimpanan yang besar. Pada penelitian iniakan dilakukan pengkompresian pada file citra berjenis bmp dengan menerapkan algoritma Elias Omega Code. Yang mana hasil dari penelitian ini di harapkan dapat bermanfaat dalam pemanfatan ukuran dari suatu file citra. Algoritma Elias Omega Code merupakan salah satu algoritma kompresi yang akan mengubah ukuran suatu file menjadi ukuran lebih kecil dari ukuran sebelumnya.

3.1.1 Study Kasus

1. **Konversi Citra ke Data Piksel**:

Langkah pertama adalah memilih salah satu gambar untuk di olah dan dianalisis dengan ukuran 4160 /3120 pixel.

Berdasarkan gambar diatas, maka untuk melakukan hitungan manual kompresi dengan menerapkan algoritmaElias Omega Code resolusi yang diambil ialah 4 × 4 pixel

Pertama, citra grayscale harus diubah menjadi kumpulan nilai intensitas piksel yang mewakili citra. Setiap nilai piksel biasanya merupakan bilangan bulat antara 0 hingga 255 (dalam citra 8-bit grayscale).

Dengan nilai pixel file citra bmp yang digunakan sebanyak 16 karakter untuk dilakukan hitungan manual. Adapun nilai pixel tersebut adalah 210, 172, 165, 180, 182, 100, 130, 180, 112, 21, 42, 83, 97, 63, 65, 79. Pembacaan total frekuensi pada tabel dibawah ini:

Tabel.2 Pembacaan Nilai Pixel

Pixel	Frekuensi
210	1
172	1
165	1
180	1
182	1
100	1
130	1
180	1
112	1
21	1
42	1
83	1
97	1
63	1
65	1
79	1
Total Pixel	16

Dari tabel diatas merupakan nilai pixel yang sama. Sebelum kompresi file citra, proses awal ialah

membaca nilai pixel file citra yang membuat nilai tabel pixel. Dapat diurutkan dari nilai frekuensi terbesar (dengan nilai yang sama) hinggayang terkecil. Adapun tabel pixel dibawah ini:

Tabel 3. Pixel Awal

Nilai Pixel	Binary	Bit	Frek	$Bit \times Frek$
210	11001000	8	1	8
172	10101011	8	1	8
165	10101000	8	1	8
180	10111001	8	1	8
100	10110111	8	1	8
130	01100101	8	1	8
112	10001000	8	1	8
181	10110101	8	1	8
114	10001101	8	1	8
21	00010101	8	1	8
42	00101010	8	1	8
83	01010011	8	1	8
97	01100001	8	1	8
63	00111111	8	1	8
65	01000001	8	1	8
79	01001111	8	1	8
Total				128

3.2 Perapanan Algoritma Elias Omega

Setelah nilai pixel diurutkan berdasarkan frekuensi kemunculan dan didapatkan nilai biner serta total bitnya, maka selanjutnya menghitung bit menggunakan kode algoritma Elias Omega Code. Aturan dalam pembentukan kode Elias Omega Code dapat dilihat pada bab sebelumnya yaitu kajian pustaka sub bab 2.2. Adapun kode Elias Omega Code dibawah ini:

Tabel 4. Kode Elias Omega Code

No	Codeword	No	Codeword
1	0	9	1110010
2	100	10	1110100
3	110	11	1110110
4	101000	12	1111000
5	101010	13	1111010
6	101100	14	1111100
7	101110	15	1111110
8	1110000	16	10100100000

Setelah mencari nilai codeword ialah melakukan kompresi nilai pixel file citra demikian nilai kode Elias Omega Code yang telah ada di tabel 4.3, adapun proses perhitungan bit pada nilai pixel file citra tersebut dengan menggunakan kode Elias Omega Code sebagai berikut:

1. Untuk proses perhitungan kode pada nilai pixel file citra menggunakan algoritma Elias Omega Code pada tabel dibawah ini:

Tabel 5. Total Bit setelah dikompresi menggunakan Elias Omega Code

No	Nilai des pixel	Elias Omega Code	Bit	Frek	Bit x Frek
1	210	0	1	1	1
2	171	100	3	1	3
3	168	110	3	1	3
4	185	101000	6	1	6
5	183	101010	6	1	6
6	101	101100	6	1	6

No	Nilai des pixel	Elias Omega Code	Bit	Frek	Bit x Frek
7	136	101110	6	1	6
8	181	1110000	7	1	7
9	114	1110010	7	1	7
10	21	1110100	7	1	7
11	42	1110110	7	1	7
12	83	1111000	7	1	7
13	97	1111010	7	1	7
14	63	1111100	7	1	7
15	65	1111110	7	1	7
16	79	10100100000	11	1	11
Total Bit				98	

Langkah selanjutnya ialah menyusun kembali codeword masing–masing dengan nilai Pixel. Setelah disusun kembali codeword masing-masing nilai pixel, maka selanjutnya adalah menghubungkan semuacodeword menjadi string bit (tanpa tanda koma dan spasi). "01001101010001010101011001011101110000111001011101001110110111100011110101111001111110101001000000" dengan total 98 bit.

Setelah melakukan pembagian bit string menjadi per 8 bit maka, untuk mengubah kembali ke karakter baru, danper karakter adalah 8 bit panjangnya.

Tabel 6. Pembagian String

01001101	01000101	01010110	01011101
11000011	10010111	01001110	11011110
00111101	01111100	11111101	01001000
00			

Setelah dilakukan pembagian bit akan menyisahkan 0 (bisa dengan menggunakan 98 mod 8 = 0). Dan ditanyakandengan simbol n, maka n = 0.

Ada beberapa langkah untuk melakukan pemeriksaan string bit yaitu:

1. Jika sisa bagi = 0, maka ditambahkan **"00000001"** diakhiri bit dan dinyatakan sebagai bit akhir.
2. Jika sisa bagi adalah (1,2,3,4,5,6,7), maka ditambahkan "0" sebanyak $(7 - n + 1)$ diakhiri dengan string bit dan dinyatakan sebagai padding. Lalu tambahkan bilangan biner dari hasil **(9 – n)** diakhir bit dan dinyatakansebagai flagging.

Padding merupakan penambahan bit data yang telah dikompresi agar seluruh jumlah bit data tersebut habis di bagi 8 sedangkan flagging merupakan penambahan 8 bit bilangan biner setelah padding yang dimana untukmempermudah dalam membaca bit–bit hasil kompresi pada saat proses dekompresi. Berdasarkan hasilperhitungan sisa bagi sebelumnya di dapat hasil n = 2, maka untuk pemeriksaan string bit menggunakan langkahkedua, yang mana jika sisa bagi = 1,2,3,4,5,6 atau 7 maka ditambahkan "0" sebanyak 7 - n + "1". Maka 7 – 2 = 5,artinya tambahkan "0" sebanyak 5 di akhir bit dan tambahkan nilai "1" dan dinyatakan sebagai padding. Makasting bit sekarang menjadi :"01001101010001010101011001011101110000111001011101001110110111100011110101111001111110101001000000000001" maka setelah dilakukan penambahan bit string, total bit sekarang menjadi 104 bit. Lalutambahkan bilangan biner dari hasil (9 – n). Maka 9 – 2 = 7 dan biner dari 7 adalah 00000111 dan dinyatakansebagai flagging.

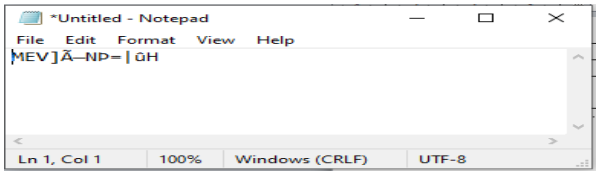
Maka bit string sekarang menjadi :
"01001101010001010101011001011101110000111001011101001110110111100011110101111001111110101001000000000100000111".

Maka setelah dilakukan penambahan bit string, total bit sekarang menjadi 112 bit. Langkah selanjutnya ialah membagi string bit yang telah di tambah padding dan flagging menjadi per 8 bit.

Tabel 7. Pengelompokan Bi

01001101	01000101	01010110	01011101	11000011
10010111	01001110	11011110	00111101	01111100
11111101	01001000	00000001	00000111	

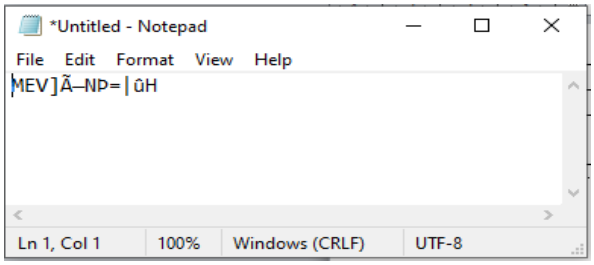
Setelah niai desimal diketahui, maka nilai akan diubah menjadi suatu karakter. Karakter hasil dari proses kompresiyang dihasilkan, disimpan dalam suatu file dengan ekstensi ".eoc", apabila file tersebut dibuka dengan aplikasi notepad, maka akan tampil karakter seperti gambar di bawah ini:



Gambar . Hasil Karakter Terkompresi

Untuk mengetahuin tingkat kinerja algoritma Elias Omega Code maka penulis akan menghitung rasio kompresi menggunakan rumus Compression (Cr) untuk mengetahui berapa rasio kompresi pengurangan ukuran yang di dapat.

2. Dekompresi berdasarkan algoritmaElias Omega Code
   Proses dekompresi pada file dokumen yang telah dikompresi dapat dilakukan dengan mengembalikan karakter– karakter aneh menjadi bilangan pixel kemudian diubah menjadi string bit semula. File hasil kompresi Elias OmegaCode yang apabila di buka menggunakan aplikasi notepad, maka akan terlihat seperti pada gambar di bawah ini:



Gambar . Hasil Karakter Terkompresi

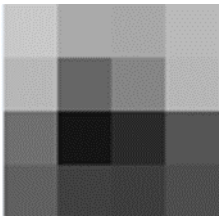
.Adapun bit hasil analisa pada karakter–karakter tersebut dapat dilihat pada tabel dibawah ini: Berdasarkan padatabel diatas, gabungkan seluruh nilai biner menjadi string bit (tanpa tanda koma dan spasi).
"0100110101000101010101100101110111000011100101110100111011011110001111010111110011111101010010000000000100000111" Dengan total 112 bit. Selanjutnya lakukan pengurangan bit untuk kembali menjadi bit string semua dengan menghilangkan padding dan flagging dengan melakukan pembacaan 8 bit akhir lalu ubah kedesimal dan dinyatakan dengan n. kemudian gunukan rumus 7 + n untuk menghilangkan padding dan flagging agar kembali ke string bit awal. "0100110101000101010101100101110111000011100101110100111011011110001111010111110011111101010010000000000100000111" (bit akhir = 00000111)
N = 00000111 = 7. Lalu gunakan rumus 7 + n untuk menghilangkan padding dan flagging 7 + n = 7 + 7 = 14.

Maka hilangkan string bit sebanyak 8 bit terakhir, sehingga string bit sekarang menjadi: "0100110101000101010101100101110111000011100101110100111011011110001111010111110011111101010 0100000" dengan total 98 bit. Berdasarkan hasil dekompresi diatas di dapati hasil akhir dari pembacaan string bitmenggunakan algoritma Elias Omega Code kembali ke string awal yaitu "200, 171, 168, 185, 183, 101, 136, 181,114, 21, 42, 83, 97, 63, 65, 79". Nilai desimal hasil dekompresi keseluruhan dapat dilihat pada gambar dibawahini:

Tabel 10. Matriks nilai pixel dekompresi

200	171	168	185
183	101	136	181
114	21	42	83
97	63	65	79

Pada gambar yang diatas hasil dekompresi mengembalikan nilai pixel sampel citra sebanyak 16 pixel tanpa adanyaperubahan nilai dari proses sebelum dikompresi, setelah nilai pixel tersebut dirubah menjadi sebuah citra sesuai file citra asli.



Gambar 8. Hasil dekompresi sampel file citra

KESIMPULAN

Berdasarkan hasil analisa yang telah dilakukan, maka kesimpulan pada penelitian ini ialah Proses awal yang dilakukan dalam mengkompresi file citra menggunakan algoritma Elias Omega Code yaitu dengan mencari file citra yang akan dikompresi, kemudian mengubah kenilai pixel dengan bantuan software matlab, dan nilai pixel tersebut akan dilakukan proses kompresi dan diperoleh hasil kompresi, Berdasarkan hitungan manual dengan menggunakan algoritma Elias Omega code telah berhasil melakukan proses kompresi file citra berekterensi *.bmp, dengan rasio kompresi sebesar 87,5%..

DAFTAR PUSTAKA

1 Tetti. P.S, dkk,"Penerapan Algoritma Levenstein Pada Aplikasi Kompresi File MP3",KOMIK, vol.2, No.1, pp.334-342, 2018

2 Michael. S,"Perbandingan Algoritma Elias Delta Code dan Unary Coding Dalam Kompresi Citra Forensik"djuornal, Vol.1, No.1, pp.18-26, 2020

3 Marlina. L, dkk, "Data Compression Using Elias Delta Code" International Journal of Recent Trends in Engineering & Research (IJRTER), Vol. 3, pp.210-217, 2017

4 Subada. M.A," Analisis Perbandingan Algoritma Even-Rodeh Code dan Algoritma Fibonacci Code untuk Kompresi File Teks", Skripsi, 2018

5 Shanmusgasundaram, S dan Lourdusamy. R,"A Comparative Study Of Text Compression Algorithms. International Journal of Wisdom Based Computing", Vol. 1 (3), pp.68-76, 2011.

6 Budiman. M.A dan Rachmawati. D, "On Using Goldbach GO Codes and EvenRodeh Codes for Text Compression". Material Science and Engineering 180, 2017

7 Rani. M dan Singh. V, "An Enhanced Text Compression System Based on ASCII Values and Huffman Coding". International Journal of Computer Science Trends and Technology (IJCST), Vol.4, Issue 3 , 2016.

8 Kodituwakku. S.R,"Comparison Of Lossless Data Compression Algorithms For Text Data". Indian Journal of Computer Science and Engineering, Vol 1, pp.416-426, 2011

9 Salomon. D. 2007. Variable-Length Codes for Data Compression. Springer: USA.

10 Wikipedia, (2019, Jun28), Fibonacci Coding [online] Available: https://en.wikipedia.org/wiki/Fibonacci_coding.