Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

Analisis Perbandingan Algoritma Elias Gamma Codes Dengan Rice Codes Pada Kompresi File Gambar BMP

Ronal Marco

[1] Fakultas Ilmu Komputer dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Jalan Sisingamanganraja No. 338, Medan, Sumatera Utara, Indonesia
Email: [1]marcojoned8@gmail.com
(*: Coressponding Author)

Abstrak-Kompresi *File* merupakan salah satu aspek penting dalam perkembangan teknologi informasi. Tuntutan Penyediaan informasi dalam waktu singkat dengan jumlah dan ukuran *File* yang banyak menjadikan teknik kompresi menjadi penting. *File* gambar adalah *File* yang mempunyai banyak karakter sehingga selalu menyebabkan masalah pada penyimpanan memori. Ada banyak Algoritma yang dikembangkan untuk kompresi *File* yaitu Algoritma Huffman, Algoritma Goldbach Codes, Elias Delta Codes, Elias Gamma Codes, dan Rice Codes. Pada penelitian ini akan dilakukan analisis perbandingan antara kedua algoritma tersebut pada kompresi citra. Analisis yang dilakukan akan membandingkan rasio tiap algoritma pada citra. Hasil analisis yang didapatkan pada penelitian ini dari file bitmap yaitu algoritma Elias Gamma Codes memiliki rasio kompresi terbaik dengan nilai Ratio of Compression (RC) sebesar 4,17 , Compression Ratio (CR) sebesar 28,5%, Redudancy sebesar 71,4%, dan Space Saving (Ss) Sebesar 72,5% dibandingkan dengan Algoritma Rice Codes dengan nilai Ratio of Compression (RC) sebesar 2 , Compression Ratio (CR) sebesar 54,5%, Redudancy sebesar 46,7%, dan Space Saving (Ss) Sebesar 45,5%.

Kata Kunci: Kompresi; File; Gambar; Elias Gamma Codes; Rice Code.

Abstract-*File compression is an important aspect in the development of information technology. The demand for providing information in a short time with a large number and size of files makes compression techniques important. Image files are files that have a lot of characters so they always cause problems with memory storage. There are many algorithms developed for file compression, namely the Huffman Algorithm, Goldbach Codes Algorithm, Elias Delta Codes, Elias Gamma Codes, and Rice Codes. In this research, a comparative analysis will be carried out between the two algorithms for image compression. The analysis carried out will compare the ratio of each algorithm in the image. The analysis results obtained in this research from bitmap files are that the Elias Gamma Codes algorithm has the best compression ratio with a Ratio of Compression (RC) value of 4.17, Compression Ratio (CR) of 28.5%, Redundancy of 71.4%, and Space Saving (Ss) of 72.5% compared to the Rice Codes Algorithm with a Ratio of Compression (RC) value of 2, Compression Ratio (CR) of 54.5%, Redundancy of 46.7%, and Space Saving (Ss) of 45.5%.*

Keywords: *Compression; File; Image; Elias Gamma Codes; Rice Code.*

1. PENDAHULUAN

Kemajuan teknologi informasi akhir-akhir ini berdampak pada perubahan kebiasaan manusia dalam pertukaran data dan informasi, sehingga kebutuhan akan data digital semakin meningkat. Dengan ruang penyimpanan yang terbatas, ini merupakan tantangan besar. Proses pertukaran data sangat dipengaruhi oleh ukuran data itu sendiri semakin besar ukuran *file*, semakin lama waktu yang dibutuhkan untuk bertukar data seperti halnya *file* gambar, semakin besar ukuran piksel gambar semakin besar juga ukuran file gambar yang perlu disimpan ruang disk ini terutama benar jika file yang digunakan adalah format bitmap.

File berektensi Bitmap merupakan file dikembangkan oleh *Microsoft.* Terdiri dari beberapa titik (*pixel*) pada gambar, tersimpan di ruang penyimpanan pada komputer. *File* ekstensi .bmp termasuk ekstensi file yang minim akan kompresi sehingga ukuran dari *file* bitmap tersebut terlalu besar. Walaupun ekstensi dari bitmap memiliki ukuran terbesar dibandingkan ekstensi pada umumnya tetapi kualitas dari gambar .bmp jernih dan bagus sehingga di zoom tidak pecah gambarnya. Kompresi data adalah proses yang mengkonversi sebuah masukan berupa aliran data(data asli) menjadi suatu data aliran lain (data yang sudah di kompresi) yang memiliki ukuran lebih kecil[1]. Tujuan dari melakukan kompresi ini adalah mengurangi tempat penyimpanan memori.

Berbagai algoritma digunakan dalam teknik kompresi data, namun dalam penelitian ini penulis menggunakan dua algoritma Elias Gamma Code dan Rice Codes dengan membandingkan kinerja dari kedua algoritma yang digunakan untuk mengkompresi *File* Gambar Tersebut. Ada beberapa aspek yang perlu diperhatikan dari hasil perbandingan kedua algoritma yaitu *Ratio of Compression* (RC), Compression Ratio (CR), *Redudancy* (Rd) dan, *Space Saving* (SS), sehingga didapatkan yang lebih efisien digunakan dalam mengompresi *file* gambar. Algoritma yang digunakan adalah *Elias Gamma Codes* dan *Rice Codes*. Kedua algoritma tersebut cukup efesien digunakan untuk kompresi data. Namun algoritma memiliki kelebihannya masing-masing, sehingga kedua algoritma tersebut harus dianalisis dan dibandingkan untuk mengetahui efisiensi dan efektivitas dalam mengompresi file gambar.

Algoritma *Elias Gamma Codes* yang kembangkan oleh Peter Elias di gunakan ketika mengkodekan bilangan bulat yang batas atasnya tidak dapat ditentukan sebelumnya. Pengkodean gamma ini bukan kode bilangan bulat nol atau negatif. Salah satu cara untuk menangani nol adalah dengan menambahkan 1 sebelum encoding dan mengurangi 1 setelah decoding.. Cara lain adalah setiap kode bukan nol untuk 1 dan kode nol sebagai 0 tunggal[2]. Disisi lain Algoritma *Rice Codes* di temukan oleh Robert F. Rice dan merupakan salah satu Teknik *entropy coding* Algoritma Rice Codes atau Rice Golomb Coding adalah salah satu algoritma yang dapat digunakan untuk mengkompresi data sehingga ukuran yang di hasilkan lebih kecil dari ukuran sebenarnya[3].

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

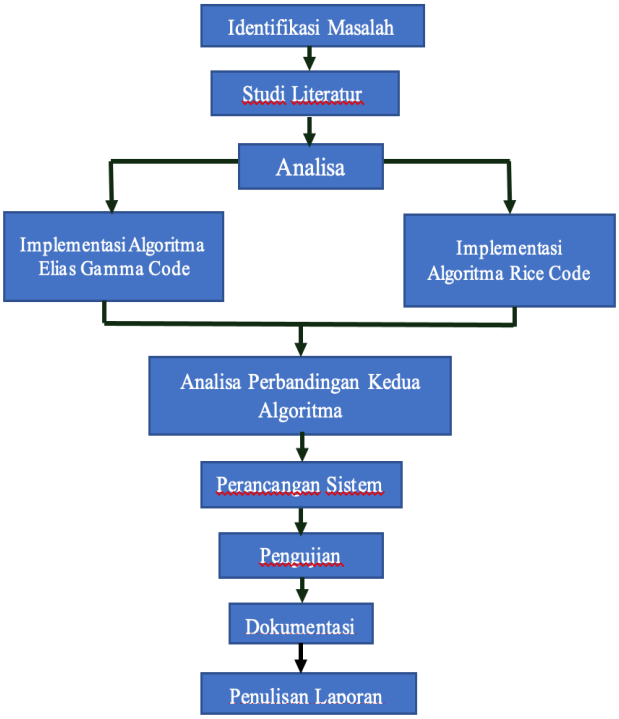
Berdasarkan penelitian sebelumnya yang berjudul "Penerapan Algoritma *Rice Code* pada aplikasi kompresi file gambar", yang dilakukan oleh Putri Fitria pada tahun 2020 disimpulkan bahwa file gambar yang di kompresi menghasilkan Ratio kompresi sebesar 50,9% Hasilnya bervariasi berdasarkan seberapa besar warna dominan pada citra hijau dominan, dan ketika di-blend memberikan rasio memori yang cukup tinggi di atas 50%[4].

Penelitian yang judul "Implementasi Algoritma Elias Gamma Kompresi Pada File Teks" yang dilakukan oleh Dina Cahyati Dkk, kita dapat menyimpulkan bahwa kinerja diukur dengan Ratio of Compression (RC), Compression Ratio (CR), Redundancy (Rd), waktu kompresi (detik) dan waktu dekompresi (detik) untuk file teks. Kompresi file teks bekerja dengan membaca string dalam file teks dengan mengkodekan string dengan Elias Gamma, kemudian mengompresinya. Hasil kompresi dalam file dengan ekstensi *.eg file ini berisi informasi karakter dan string bit hasil terkompresi yang dapat didekompresi. Algoritma Elias Gamma sensitif terhadap jumlah variasi karakter, selama proses kompresi pada string Elias Gamma Rasio Kompresi rata-rata sebesar 2.192%[5].

2. METODOLOGI PENELITIAN

2.1 Metodologi Penelitian

Metodologi penelitian adalah kerangka kerja atau tahapan yang digunakan dalam pengembangan suatu sistem untuk menentukan apa yang diperlukan sebagai solusi untuk memecahkan suatu masalah agar sistem dapat bekerja sesuai dengn tujun utamanya. Adapun Kerangka yang penulis lakukan yaitu:



Gambar 1. Kerangka penelitian

Berdasarkan informasi pada gambar 3.1 Kerangka penelitian diatas, maka tujuan dari langkah informasi dapat dijelaskan sebagai berikut:

a. Identifikasi Masalah
   Identifikasi masalah adalah analisis masalah untuk menentukan proses identifikasi sebab dan akibat menciptakan sistem yang dapat layak sehingga dapat bekerja sesuai dengan tujuan Utamanya. Permasalahan yang muncul dari penelitian ini adalah menggunakan metode perbandingan eksponensial untuk membandingkan Algoritma *Elias Gamma Codes* dengan Algoritma *Rice Codes* dan menemukan Algoritma kompresi file gambar terbaik diantara kedua algoritma tersebut.
b. Study Literatur
   Study literatur dilakukan dengan membaca dan mengumpulkan referensi penelitian yang ditemukan oleh penulis jurnal, makalah akademis, *e-book,* dan artikel lain yang terkait dengan kompresi File Gambar, Algoritma *Elias Gamma Codes,* Algoritma *Rice Codes,* dilakukan untuk memahami subjek penelitin dan pemrograman Microsoft Visual Basic 2008.
c. Analisa
   Sistem menerapkan Algoritma *Elias Gamma Codes* dan Algoritma *Rice Codes*. Pada tahap ini aplikasi kompresi bersifat *lossless,* dan data kompresi dapat didekompresi kembali. Apalagi data hasil kompresi data yang dihasilkan sama persis dengn data asli sebelum dikompresi.

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

d. Implementasi Algoritma *Elias Gamma Codes* dan *Rice Codes*
Tahapan analisis perbandingan Algoritma *Elias Gamma Codes* dan *Rice Codes* kompresi File Gambar. Pada tahap ini dilakukan proses untuk membandingkan apakah algoritma Elias Gamma Codes atau Algoritma Rice Codes lebih efektif dalam memperkecil ukuran data.
e. Analisa Perbandingan
Pada tahap ini semua tentang memahami algoritma mana yang bekerja paling baik saat mengompresi *file* gambar.
f. Perancangan Sistem
Fase ini adalah merupakan proses perancangan alur kerja, yang di buat dengan mengimplementasikan metode-metode yang telah diproses sebelumnya.
g. Pengujian Sistem
Hal ini dilakukan untuk memeriksa apakah sistem yang dirancang dapat bekerja baik dan perlu adanya pengembangan lebih lanjut
h. Dokumentasi
Pada tahap terakhir ini, Investigasi yang dilakukan didokumentasikan dari awal hingga pengujian sistem, setelah itu surat dikompilasi penelitian dilakukan dengan menarik kesimpulan dan laporan dari penelitin yang dilakukan, yang pada akhirnya dihasilkan dalam tesis.
i. Analisa Laporan
Menulis laporan untuk mendokumentasikan semua kegiatan penelitian dalam bentuk tesis. Ini jug tersedia dalam bentuk artikel ilmiah yang diterbitkan.

2.2 Algoritma Elias Gamma Codes

Algoritma *Elias Gamma Codes* dikembangkan oleh Peter Elias. Tabel kode elias gamma, elias dibuat dengan menambah panjang kode dalam *Unary* (u). Kode berikutnya, Eγ ditambahkan ke panjang kode (M) dalam biner (β). Kode Elias Gamma Codes, yang juga berlaku untuk bilangan bulat positif karena itu agak rumit untuk dibangun[10]. Aturan untuk penyandian angka dengan menggunakan *Elias Gamma Codes* adalah sebagai berikut:

a. Mengkoversi nomor kode ke biner.
b. Kurangi 1 dari jumlah bit yang dipilih pada langkah pertama dan tambahkan sesuai jumlah nol. Prosedur ekuivalen untuk menyatakan proses yang pada angka 2 dalah sebagai berikut:
   1. Dibagi menjadi pangkat 2 yang dapat menampung bilangan bulat (2*N*), meninggalkan bilangan biner N dari bilangan bulat tersebut
   2. Encode N dalam bentuk *unary*, jika N adalah nol maka diikuti oleh 1. Jumlahkan sisa bilangan biner N yang dihasilkan.

Proses kompresi/encoding bilangan bulat berdasarkan elias gamma dilakukan dengan cara:

a. Carilah nilai N untuk pangkat yang paling dekat nilai n yang ditulis sebagai β(n). Nilai ini disebut sebagai *unary code*, dan ditulis jumlah nilai N sebagai 0 dan diakhiri dengan angka
b. Kurangi nilai n dengan 2 untuk mendapatkan nilai L. nilai yang ditemukan konversi ke biner sebagai contoh, kita dapat mengambil pembentukan kode elias gamma. N=9 kemudian cari N bilangan bulat terbesar sehingga 2N < 2+1 = 23 9 <, <; 2 3+1 Setelah menemukan kode N terbesar, ubah nilai n menjadi biner dan hilangkan 1 bit ke kiri 9 = 1001 → 001 0 diikuti dengan 1 untuk menghasilkan unary 3 → 0001 . Kemudian tambahkan sisa bilangan biner n setelah kode unary yang dihasilkan 0001001 . Untuk lebih jelasnya, berikut hasil pengkodean kode gamma Elias ditunjukkan pada Tabel 1.

Tabel 1. Tabel Algoritma *Elias Gamma Codes*

Index	Proses	Elias Gamma Codes
1	2^0+0	1
2	2^1+0	010
3	2^1+1	011
4	2^2+2	00100
5	2^2+1	00101
6	2^2+2	00111
7	2^3+3	00111
8	2^3+0	0001000
9	2^3+1	0001001
10	2^3+2	0001010
11	2^3+3	0001011
12	2^3+4	0001100
13	2^3+5	0001101
14	2^3+6	0001110
15	2^3+7	001111
16	2^4+0	000010000
17	2^4+1	000010001

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

Index	Proses	Elias Gamma Codes
18	2^4+2	000010010

Langkah selanjutnya adalah konversi karakter sesuai kode *Elias Gamma Codes*, sebelum dikonversi terlebih dahulu mengecek panjang bit string pada langkat dibawah ini:

a. Tambahkan 00000001 jika sisa panjang string bit 8 adalah 0
b. Jika sisa bagi panjang string bit 8 adalah n (1, 2, 3, 4, 5, 6, 7) tambahkan 0 hingga 7 – n + "1" di akhir string bitd diwakilkan L, lalu menambahkn digit *biner* dari 9 hingga n. Nyatakan dengan bit akhir.

Proses Dekompresi/*decoding* dalam kode *Elias Gamma Codes* dapat dilakukan langkah-langkah adalah:

a. baca 8 bit terakhir, hasilnya adalah angka desimal. Nyatakan hasil pembacaan dengan n.
b. Hapus bit tambahan 7 + n.

Dibawah ini adalah algoritma dekompresi untuk *Elias Gamma Codes* adalah:

a. Baca urutan string bit dari awal sampai ditemukan angka 1. Catat posisi angka 1 dan tuliskan sebagai p. Biarkan n menjadi jumlah nol.
b. Baca urutan string bit setelah angka 1 sampai n.
c. Ganti kode dengan karakter yang disesuai berdasarkan tabel elias gamma codes

2.3 Algoritma Rice Codes

Algoritma *Rice Codes* pertama kali ditemukan oleh Robert F. *Rice* pada tahun 1979. Algoritma *Rice Codes* memiliki nilai k yang mewakilkan jumlah angka dalam suffix dari kode terkompresi. Awalan dan akhiran dipisahkan selama penyandian. Selama Proses decoding, decoder membaca bit tanda, melompat ke 0 pertama sebelah kiri dan terus menambahkan bit ke k berikutnya [11].

Pengkodean dimana m adalah pangkat dua (m = 2k) menghasilkan Rice Codes. Juga dikenal sebagai Golomb, ditemukan Robert F. *Rice*, *Rice Codes* juga terkait dengan *Subexponential code*. *Rice Codes* untuk pilihan dasar.

$L = 0$; % $inisialisasi$
$N = 0$;
$m = 1$; % atau tanya pengguna %

Untuk setiap dari 0, r menggunakan m untuk menghasilkan Golomb Code untuk r.

$L = L + r$: % $perbarui\ L, N, and\ m$
$N = N + 1$; $p = L/(L + N)$;
m = $[-1/\ log2\ p + 0.5]$

Pada nilai k untuk proses kompresi, hal berikut:

a. Pisahkan bit tanda dari sisa nomor. Ini opsional dan bagian terpenting dari rice codes.
b. Pisahkan k LSB ke dalam LSB dari rice codes.
c. Sisa kode $j = [n/2\ k\ ]$ bit. j adalah nol diikuti oleh 1 atau j adalah 1 diikuti oleh sebuah 0 (mirip dengan kode unary). Ini adalah bagian tengah dari rice codes. Jadi, kode dihitung dengan beberapa operasi logis. Ini lebih cepat dari menghitung kode huffman, yang merupakan proses kompresi yang menggunakan pohon huffman untuk mengumpulkan bit individu dari kode. Fitur ini sangat penting untuk decoder, yang harus sederhana dan cepat. menunjukkan contoh kode ini untuk k = 2 , yang sesuai dengan m = 4 (kolom berlabel "jumlah 1 " mencantumkan nomor 1 ditengah kode). Perhatikan bahwa kode dalam tabel ini identik (kecuali bit tambahan opsional) ke kode pada baris 3 (baris dengan m = 4).

Tabel 2. Kode *Rice Codes*

No	Biner	Sign	LSB (List Sign Bit)	No. of ones	codeword
0	0	0	00	0	0000
1	1	0	01	0	0001
2	10	0	10	0	0010
3	11	0	11	0	0011
4	100	0	00	1	010000

3. ANALISA DAN PEMBAHASAN

3.1 Analisa

Tahap analisis merupakan penelitian tahap pertama yang bertujuan untuk menemukan permasalahan yang berkaitan dalam pembuatan sistem. Menganalisis sistem merupakan langkah penting untuk mengetahui proses yang dijalankannya. Proses yang dilakukan untuk mengolah data atau informasi *input* menggunakan metode langkah-langkah yang digunakan untuk memecahkn masalah dan output dari proses yang dilakukan**.** Dalam hal ini, penulis menganalisis kompresi dari algoritma Elias Gamma Codes dan algoritma Rice Code untuk *File* Gambar menggunakan Metode Eksponensial untuk membandingkan kedua algoritma tersebut.

Langkah pertama adalah menentukan komposisi karakter string pada nilai *pixel* pada *file* gambar. Setelah set karakter diperoleh, frekuensi kemunculan setiap karakter dihitung, kompresi dilakukan sesuai langkah-langkah algoritma

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

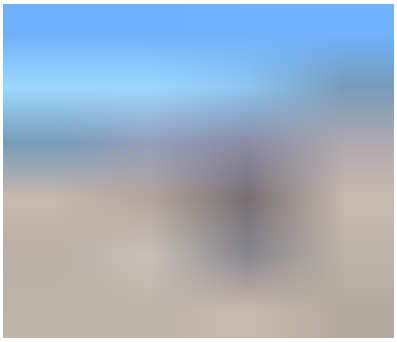
yang digunakan. Setelah menjalankan proses kompresi, hasil baru yang dihasilkan dapat diatur ulang ke nilai awalnya dengan proses dekompresi. Selanjutnya mengguakan metode eksponential untuk membandingkan kedua algoritma untuk menguji efisiensi dan efektivitasnya dalam mengompresi *file* gambar.

3.1.1 Penerapan Algoritma Elias Gamma Codes

Analisis proses kompresi file gambar dengan *Elias Gamma Codes* Langkah pertama dalam kompresi file gambar adalah dengan menyisipkan file gambar. Ukuran file gambar yang digunakan untuk proses kompresi adalah 3120x4160 pixel.

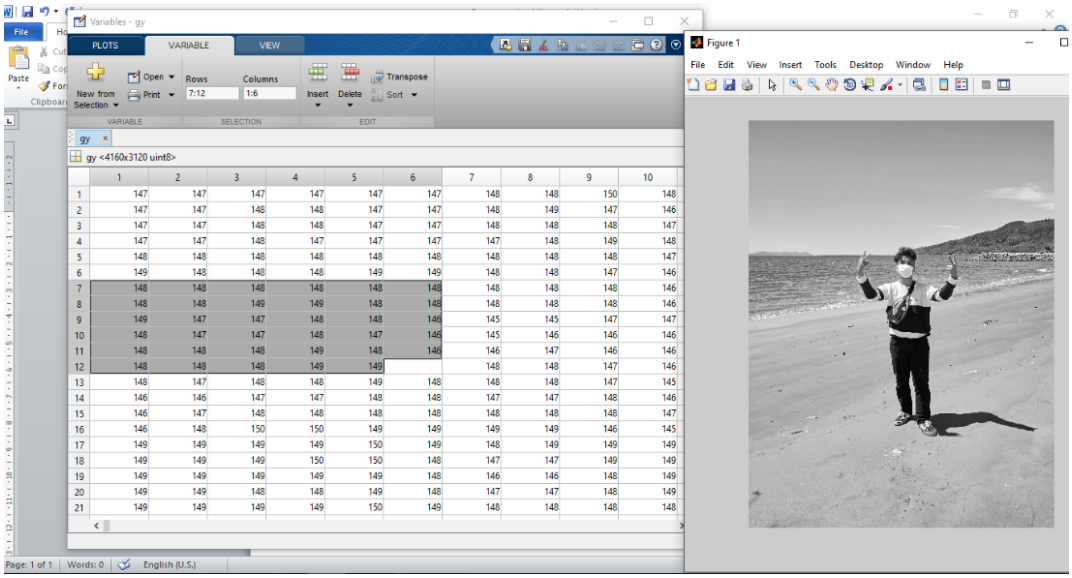
Gambar 2. Citra Asli

Berdasarkan proses analisis yang dilakukan citra sample gambar sebesar 6 x 6 pixel dengan kedalaman warna 24-bit direkam. Proses kompresi dilakukan menggunakan algoritma Elias Gamma Codes untuk kompresi *file* gambar. Dengan begitu dapat dihitung 6 x 6 x 24 bit = 864 bit. Saat dikonversi ke byte menjadi 864/8 = 108 byte. Di bawah ini adalah contoh gambar untuk dikompres.



Gambar 3. sampel gambar 6 x 6 pixel

Langkah selanjutnya adalah membaca isi file gambar. Setiap file gambar terdapat nilai pixel. Nilai pixel tersebut diolah dalam bentuk 6 x 6 sehingga menghasilkan 36 pixel sebagai pola citra yang akan dianalisis. Adapun nilai 36 pixel diperoleh menggunakan aplikasi MATLABR2010a dan dapat dilihat pada gambar di bawah ini:



Gambar 4. Tampilan Nilai Pixel Matlab

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

Berdasarkan pada gambar 4 di atas diperoleh nilai pixel karakter sebagai berikut ; 146, 147, 148, 149. Nilai pixel ini dimasukan kedalam tabel untuk dilakukan pembacaan frekuensi. Pembacaan frekuensi dilakukan dengan menghitung jumlah nilai identik di setiap nilai pixel yang ditampilkan. Nilai frekuensi diperoleh pada tabel di bawah ini:

Tabel 1. Pembacaan Nilai Pixel

No	Nilai Pixel	Frekuensi
1	146	3
2	147	5
3	148	22
4	149	5
	Total	45

Dapatkan nilai piksel yang sama berdasarkan tabel diatas. Sebelum proses kompresi langkah pertama adalah membaca nilai piksel pada citra dan membuat tabel nilai piksel yang diurutkan dari nilai yang paling sering (nilai piksel yang sama) hingga yang paling sedikit. Urutan nilai dapat dilihat pada tabel di bawah ini :

Tabel 2. Nilai Piksel Sebelum kompresi Elias Gamma Codes

No	Pixel	Binary	Value (ASCII)	Frekuensi	Frekuensi x bit
1	146	10010010	8 bit	3	24
2	147	10010011	8 bit	5	40
3	148	10010100	8 bit	22	176
4	149	10010101	8 bit	5	40
				Jumlah	280

Berdasarkan tabel di atas, nilai desimal (tanda) setara dengan 8 bit dalam biner. Untuk mengonversi satuan ke byte, bagi jumlah total bit dengan 8, menghasilkan 280/8 = 35 byte. Sebelum melakukan kompresi data, maka terlebih dahulu mencari codeword dari algoritma *Elias Gamma Codes*.

Tabel 3. Tabel Algoritma *Elias Gamma Codes*

Index	Proses	Elias Gamma Codes
1	2^0+0	1
2	2^1+0	010
3	2^1+1	011
4	2^2+2	00100
5	2^2+1	00101
6	2^2+2	00111
7	2^3+3	00111
8	2^3+0	0001000
9	2^3+1	0001001
10	2^3+2	0001010
11	2^3+3	0001011
12	2^3+4	0001100
13	2^3+5	0001101
14	2^3+6	0001110
15	2^3+7	001111
16	2^4+0	000010000
17	2^4+1	000010001
18	2^4+2	000010010

Kemudian kompresi *file* gambar tersebut menggunakan algoritma *Elias Gamma Codes*. Langkah pertama membentuk karakter set pada sebuah tabel algoritma *Elias Gamma Codes*, setiap set karakter string bit kemudian digantikan dengan codeword *Elias Gamma Codes* sesuai dengan tabel 4.3 di atas. Setelah diganti, hitung jumlah bit pada setiap karakter pada nilai piksel citra dengan menggunakan *Elias Gamma Codes* yang ada pada tabel di bawah ini :

Tabel 4. Nilai pixel Sesudah Dikompresi Elias Gamma Codes

No	Pixel	Frekuensi	Biner	2^N+L	Kurang 1 bit di kiri	Unary (N)	codeword	Bit	Frek x Bit
1	148	22	11	2^0+0		1	1	1	22
2	147	5	10	2^1+0	0	01	010	3	15

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

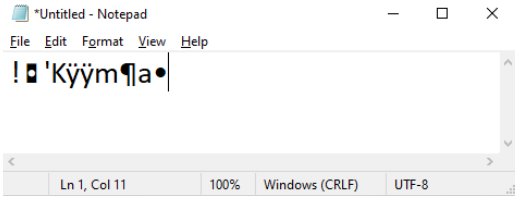
No	Pixel	Frekuensi	Biner	2^N+L	Kurang 1 bit di kiri	Unary (N)	codeword	Bit	Frek x Bit
3	149	5	11	2^1+1	1	01	011	3	15
4	146	3	100	2^2+0	00	001	00100	5	15
Total									67 bit

Berdasarkan Tabel 4 di atas, nilai bit baru dapat dibentuk dari pengurutan himpunan karakter. Nilai string biner terkompresi tersebut kemudian diurutkan menggunakan algoritma Alias Gamma Code. Bit kompresi total menunjukkan nilai 67-bit. Langkah berikutnya membagi string bit menjadi per 8 bit lalu merubahnya ke desimal dan mengubahnya ke karakter. Proses mengubah string bit menjadi sebuah karakter pada kompresi dapat dilihat pada tabel berikut:

Tabel 5. String Bit Karakter Hasil Kompresi Elias Gamma Codes

Biner	Nilai Desimal	Karakter
00100001	33	!
00001000	8	◘
10010010	146	'
01001011	75	K
11111111	255	ÿ
11111111	255	ÿ
11110110	109	m
11011011	182	¶
01100001	97	a
00000110	7	•

Setelah mengetahui nilai desimal setiap karakter, ubah nilai desimal menjadi karakter. Karakter terkompresi disimpan dalam file, dan ketika Anda membuka file dengan aplikasi Notepad, Anda akan melihat karakter seperti ini:



Gambar 5. Hasil Karaker Kompresi *Elias Gamma Codes*

Berdasarkan penambahan jumlah bit setelah dilakukannya kompresi dengan algoritma *Elias Gamma Codes,* didapatkan hasil dari kompresi yang bernilai 80 bit. Konversi ke byte menghasilkan 80/8 = 10 byte. Kemudian Anda dapat memperkirakan ukuran sampel file gambar sebelum kompresi dengan algoritma *Elias Gamma Codes* adalah bernilai 35 byte, sedangkan ukuran sampel *file* gambar yang sudah dikompresi dengan algortima *Elias Gamma Codes* adalah bernilai 10 byte setelah dilakukannya pembagian. Berdasarkan hasil Kompresi dengan Algoritma *Elias Gamma Codes* dapat dihitung kinerjanya sebagai berikut:

a. Ratio of Compression (RC)

$$RC = \frac{\text{Jumlah bit sebelum dikompresi}}{\text{Jumlah bit sesudah dikompresi}}$$

$$RC = \frac{280\ \text{bit}}{67\ \text{bit}} = 4{,}17$$

b. Compression Ratio (CR)

$$CR = \frac{\text{Jumlah bit sesudah dikompresi}}{\text{Jumlah bit sebelum dikompresi}} \text{ X } 100\%$$

$$CR = \frac{10\ \text{byte}}{35\ \text{byte}} \text{ X } 100\% = 28{,}5\%$$

c. Redudancy (Rd)

$$Rd = \frac{\text{File sebelum dikompresi} - \text{File Setelah dikompresi}}{\text{Ukuran file Sebelum Dikompresi}} \text{ x } 100\%$$

$$Rd = \frac{35\ byte - 10\ byte}{35\ \text{byte}} \text{ x } 100\%$$

$$= 71{,}4\%$$

d. Space Saving (Ss)

$$Ss = 100\% - CR =$$

$$Ss = 100\% - 28{,}5\% = 72{,}5\%$$

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

3.1.2 Penerapan Algoritma Rice Codes

Dalam penerapan algoritma *Rice Code*s ini, *File Gambar* yang berisi Beberapa nilai pixel yaitu dengan sampel pixel 6 x 6 pixel memberikan data yang ingin dikompresi yaitu sebesar 120 bit. Berdasarkan pada tabel 2 di atas, didapatkan beberapa nilai frekuensi dan biner yang sama. Sebelum proses kompresi data teks dengan algoritma *Rice Codes*, langkah awal adalah membaca nilai biner kemudian membuat tabel nilai desimal yang diurutkan dari nilai frekuensi terbesar (nilai Binner yang sama) hinggs ke terkecil. Urutan nilai Desimal dapat dilihat pada tabel di berikut:

Tabel 6. Nilai Pixel sebelum Dikompresi Rice Codes

No	Nilai Pixel	Binary	Bit (ASCII)	Frekuensi	Frekuensi x bit
1	146	10010010	8 bit	3	24
2	147	10010011	8 bit	5	40
3	148	10010100	8 bit	22	176
4	149	10010101	8 bit	5	40
Jumlah					280 Bit

Berdasarkan tabel diatas, satu nilai decimal (karakter) bernilai 8 bit bilangan biner. Untuk mengubah satuan menjadi byte maka jumlah keseluruhan bit dibagikan 8, maka dihasilkan 280/8 = 35 byte. Sebelum melakukan kompresi data, maka terlebih dahulu mencari codeword dari algoritma *Rice Codes*. Terdapat beberapa tahap dalam pencarian codeword pada algoritma ini, adapun aturan dari langkah langkah dalam mencari codeword algoritma *Rice Codes* yaitu:

a. Pisahkan bit tanda dari sisa nomor. Ini opsional dan bit menjadi bagian terpenting dari rice codes.
b. Pisahkan k LSB ke dalam LSB dari rice codes.
c. Sisa kode $j = [n/2\ k\ ]$ bit. J adalah nol diikuti oleh 1 atau j adalah 1 diikuti oleh 0 (mirip dengan kode unary). Ini adalah bagian tengah dari rice codes. Jadi kode dihitung dengan beberapa logis. Ini lebih cepat daripada menghitung huffman kode, yang merupakan proses kompresi yang menggunakan pohon huffman untuk mengumpulkan bit individu dari kode. Fitur ini sangat penting untuk decoder, yang harus sederhana dan cepat. menunjukkan contoh kode ini untuk k = 2 , yang sesuai dengan m = 4 (kolom berlabel "jumlah 1" mencantumkan nomor 1 dibagian tengah kode). Perhatikan bahwa kode tabel ini identik (kecuali opsional tambahan) sama dengan kode pada baris ke 3 (baris dengan1 m = 4

Tabel 7. Kode *Rice Codes*

No	Biner	Sign	LSB (List Sign Bit)	No. of ones	codeword
0	0	0	00	0	0000
1	1	0	01	0	0001
2	10	0	10	0	0010
3	11	0	11	0	0011
4	100	0	00	1	010000

Selanjutnya mengkompresi *File* Gambar menggunakan algoritma *Rice Codes*. Tahapan pertama adalah dengan membentuk karakter set pada sebuah tabel algoritma *Elias Gamma Codes*, setelah itu pada tiap – tiap set karakter string bit tersebut, diganti dengan codeword *Rice Codes* sesuai pada tabel 4.9 di atas. Setelah substitusi,gunakan kode rice pada tabel di bawah untuk menghitung jumlah bit pada setiap karakter dalam file gambar sebagai berikut:

Tabel 8. Nilai Pixel yang Kompresi Rice Codes

No	Nilai Pixel	Frekuensi	biner	Sign	LSB (List Sign Bit)	No. of ones	codeword	bit	Frek x Bit
1	148	22	0	00	0	0	0000	4	88
2	147	5	0	01	01	0	0001	4	20
3	149	5	0	01	10	0	0010	4	20
4	146	3	0	11	11	00	0011	4	12
Total									140 bit

Berdasarkan pada tabel di atas dapat dibentuk nilai bit baru dari susunan set karakter, setelah itu mengurutkan nilai string biner yang sudah dikompresi dengan algoritma *Rice Codes.* Langkah berikutnya membagi string bit menjadi per 8 bit lalu merubahnya ke decimal dan mengubahnya ke karakter. Proses mengubah string bit menjadi sebuah karakter pada kompresi dapat dilihat pada tabel berikut:

Tabel 9. String Bit Karakter Kompresi Rice Codes

Biner	Desimal	Karakter
00110011	51	3
00110001	49	1
00010001	17	◄
00000000	17	◄
00000000	0	

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

Biner	Desimal	Karakter
00000000	0	
00000000	0	
00000000	0	
00000000	0	
00000000	0	
00000000	0	
00000000	0	
00000000	0	
00000000	0	
00100010	34	“
00100010	34	“
00100001	33	!
00000101	5	

Berdasarkan penambahan jumlah bit setelah dilakukannya kompresi dengan algoritma *Rice Codes,* didapatkan hasil dari kompresi yang bernilai 152 bit. Jika diubah menjadi byte maka 152/8= 19 byte. Maka dapat disimpulkan bahwa ukuran sampel *data teks* sebelum dikompresi dengan algoritma *Elias Gamma Codes* adalah bernilai 35 byte, sedangkan ukuran sampel data teks yang sudah dikompresi dengan algortima *Rice Codes* adalah bernilai 19 byte setelah dilakukannya pembagian. Berdasarkan hasil Kompresi dengan Algoritma *Rice Codes* dapat dihitung kinerjanya sebagai berikut ini:

1. Ratio of Compression (RC)

$$RC = \frac{\text{Jumlah bit sebelum dikompresi}}{\text{Jumlah bit sesudah dikompresi}}$$
$$RC = \frac{280\ \text{bit}}{140\ \text{bit}} = 2$$

2. Compression Ratio (CR)

$$CR = \frac{\text{Jumlah bit sesudah dikompresi}}{\text{Jumlah bit sebelum dikompresi}} \text{ X } 100\%$$
$$CR = \frac{19\ \text{byte}}{35\ \text{byte}} \text{ X } 100\% = 54{,}2\%$$

3. Redudancy (Rd)

$$Rd = \frac{\text{File sebelum dikompresi} - \text{File Setelah dikompresi}}{\text{Ukuran file Sebelum Dikompresi}} \text{ x } 100\%$$
$$Rd = \frac{35\ byte - 19\ byte}{35\ \text{byte}} \text{ x } 100\%$$
$$= 45{,}7\%$$

4. Space Saving (Ss)

$$Ss = 100\% - CR =$$
$$\text{Ss} = 100\% - 54{,}5\% = 45{,}5\%$$

3.1.3 Perbandingan Algoritma Elias Gamma Codes Dan Rice Codes

Analisis perbandingan dari dua algoritma memerlukan langkah-langkah untuk menghitung dan membandingkan dua algoritma. Berikut langkah-langkahnya:

a. Tentukan dasar untuk membandingkan dua algoritma. Kriterianya adalah Compression Ratio (RC), Compression Ratio (CR), Redundancy (RD), Space Saving (SS).
b. Melaporkan nilai untuk setiap kriteria yang ditentukan. Nilai ini didasarkan pada analisis algoritma Elias Gamma Codes dan Rice Codes untuk mengompresi file gambar BMP.
c. Menentukan hasil atau penentuan prioritas berdasarkan nilai masing-masing kriteria. Hasil keputusan tersebut dapat dilihat pada tabel di bawah ini:

Tabel 10. Analisa Hasil Perbandingan Algoritma Kompresi

Algoritma Elias Gamma Code	4,17	28,5%	71,4%	72,5%	1
Algoritma Rice Code	2	54,5%	46,7%	45,5%	2

Berdasarkan analisis perbandingan pada keterangan di atas, algoritma *Elias Gamma Codes* mendapatkan perangkingan peringkat 1 dibandingkan *Rice Codes* yang menduduki peringkat ke 2. Maka dapat disimpulkan bahwa algortima *Elias Gamma Codes* sangat efektif dan efesiensi dalam mengkompresi File Gambar dibandingkan dengan Algoritma *Rice Codes.*

Jurnal Sains Dan Teknologi Informasi
Vol 4, No 1, Desember 2024, Hal 15-24
ISSN 2809-610X (Media Online)
DOI 10.47065/jussi.v4i1.7702
Website https://ejurnal.seminar-id.com/index.php/jussi

4. KESIMPULAN

Berdasarkan pada analisis, kesimpulan dari perbandingan kinerja kompresi algoritma *Elias Gamma Codes* dan algoritma *Rice Codes* yaitu Pada proses pengkompresian File Gambar Bmp menggunakan algoritma *Elias Gamma codes* serta algoritma *Rice Codes* sudah memastikan jika *File Gambar* yang mempunyai ukuran besar bisa dikompresi menjadi ukuran yang lebih kecil. Pada analisis serta perbandingan kompresi file gambar, hasil pengujian algoritma *Elias Gamma Codes* lebih baik dari algoritma *Rice Code.* Dimana algoritma *Elias Gamma Codes* mendapatkan hasil rata - rata perbandingan *Ratio of Compression(RC)* bernilai 4,17, *Compression of Ratio(CR)* bernilai 28,5% , Redundancy(RD) bernilai 71,4%, dan *Space Saving*(SS) yang bernilai 72,5% dibandingkan dengan algoritma *Rice Codes* yang hasil rata – rata perbandingan *Ratio of Compression(RC)* bernilai 2, *Compression of Ratio(CR)* bernilai 54,5% , Redundancy(RD) bernilai 46,7%, dan Space Saving(SS) yang bernilai 45,5%.

REFERENCES

[1] D. A. Yansyah and I. Pendahuluan, “Perbandingan Metode Punctured Elias Code Dan,” vol. 2, no. 6, pp. 33–36, 2015.
[2] R. Siregar, “Penerapan Algoritma Elias Gamma Code Pada Aplikasi Kumpulan Resep Makanan Berbasis Android,” vol. 2, no. 5, pp. 284–291, 2021.
[3] M. A. Latif, S. D. Nasution, I. Pendahuluan, A. K. Data, and B. A. R. Codes, “Analisa Perbandingan Algoritma Rice Codes Dengan Algoritma Goldbach Codes Pada Kompresi File Text Menggunakan Metode Exponential,” vol. 5, pp. 113–117, 2018.
[4] P. Fitria, “Penerapan Algoritma Rice Codes Pada Aplikasi Kompresi File Gambar,” vol. 1, no. 3, pp. 158–165, 2020.
[5] D. Cahayati, A. M. H. Pardede, and H. Khair, “Implementasi Algoritma Elias Gamma Kompresi Pada File Teks,” vol. 6341, no. April, pp. 159–166, 2022.
[6] E. Prayoga and K. M. Suryaningrum, “Implementasi Algoritma Huffman Dan Run Length Encoding Pada Aplikasi Kompresi Berbasis Web,” vol. IV, no. 2, pp. 92–101, 2018.
[7] E. Surahman, P. Studi, and T. Informatika, “Audio Wav Dengan Menggunakan.”
[8] F. Riandari and T. Informatika, “Rancang Bangun Aplikasi Kompresi Text,” vol. 1, no. 2, pp. 130–133, 2017.
[9] “No Title,” vol. 3, pp. 1–32, 2013.
[10] H. Sartika, T. Zebua, and R. Parapat, “Perancangan Dan Implementasi Algoritma Elias Gamma Code Untuk Mengkompresi Record Database Pada Aplikasi Rangkuman,” vol. 3, pp. 259–265, 2019, doi: 10.30865/komik.v3i1.1600.
[11] K. Ramayani, “Penerapan Algoritma Rice Codes Untuk Mengkompresi File Video,” vol. 5, pp. 186–192, 2021, doi: 10.30865/komik.v5i1.3670.