Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

Penerapan Algoritma Punctured Elias Codes Dalam Kompresi Citra

Dwi Ayu Depika*, Surya Darma Nasution

Program Studi Teknik Informatika, STMIK Budi Darma, Medan, Indonesia
Email: 1,*ayudepikadwi4@gmail.com, 2darmashadow@gmail.com

Abstrak

Dalam proses untuk mengelola beberapa gambar atau gambar, dengan ukuran gambar atau gambar yang lebih besar dapat menampung penyimpanan lebih lama dan tempat penyimpanan yang lebih besar, dibutuhkan lebih sering terkait tidak dapat tertampung pada media penyimpanan dan tidak tersampaikan, sehingga akan membuka ruang kosong dalam memori media penyimpanan. Maka untuk membuat banyak ruang kosong dan memiliki ukuran data yang tidak besar pada media penyimpanan diperlukan metode kompresi yang memiliki arti untuk mempersingkat ukuran bit yang diperlukan. Algoritma Punctured Elias Kode dapat digunakan untuk mendapatkan informasi tentang kompresi yang dilakukan dengan mengkompresi gambar gambar untuk memberikan manfaat dalam penyimpanan serta membutuhkan ruang memori yang lebih banyak dibandingkan dengan gambar gambar yang tidak dikompresi

Kata Kunci: Citra Gambar; Kompresi; Algoritma; Punctured Elias Codes

Abstract

In the process of managing multiple images or images, with larger image sizes or images that can hold longer storage and larger storage, it is needed more often as it cannot be contained on storage media and is not conveyed, thus opening up free space in memory storage media. So to make a lot of free space and have a data size that is not large on the storage media, a compression method is needed which means to shorten the required bit size. Punctured Elias Algorithm Code can be used to obtain information about the compression performed by compressing images to provide benefits in storage and requires more memory space compared to uncompressed images.

Keywords: Images; Compression; Elias Code Punctured; Algorithm

1. PENDAHULUAN

Perkembangan dalam dunia teknologi informasi dan komunikasi telah berkembang pesat seiring kebutuhan masyarakat dalam memperoleh informasi secara cepat, selain memiliki potensi dalam menyaring data dan mengolah menjadi informasi,teknologi mampu menyimpan dengan jumlah kapasitas jauh lebih banyak, berbagai macam fasilitas teknologi telah dikembangkan agar dapat membantu masyarakat melakukan banyak pertukaran informasi dalalam bentuk citra atau gambar, teks, audio dan lain sebagainya, dengan kapasitas yang lebih baik. Citra (gambar) merupakan hal yang sangat vital dan menjadi bagian integral bagi kehidupan sehari-hari, citra (gambar) juga digunakan sebagai alat pengungkapan, pertimbangan (*reason*), interpretasi, ilustrasi, ingatan (*memorise*) dan lain sebagainya.

Berdasarkan keterbatasan media penyimpanan sering sekali manjadi kendala dalam proses untuk mengelola suatu citra atau gambar, dengan ukuran citra atau gambar yang lebih besar dapat memakan waktu penyimpanan lebih lama dan tempat penyimpanan yang lebih besar, terkadang sering terjadi resiko tidak dapat tertampung pada media penyimpanan dan terkadang tidak tersampaikannya, sehingga akan memperkecil ruang kosong dalam memori media penyimpanan, dibandingkan dengan ukuran citra atau gambar yang lebih kecil hanya membutuhkan waktu yang lebih singkat dan membutuhkan ruang memori dalam *storage* yang lebih sedikit.

Dari hasil penelitian sebelumnya yang dilakukan oleh Dedek Andri Yansyah pada tahun 2015 telah berhasil mengkompresi *file text* dengan menggunakan metode *punctured elias codes* yang berektensi *.doc hasil *output* dekompresi telah diatur dengan format perataan *text justify* dan jenis *font timesnew roman* dengan ukuran *font* 12[1]. Pada tahun 2016 Tri Rahmah Silviani dan Ayu Arfiana pernah meneliti tentang teknik kompresi citra menggunakan metode *huffman* bahwa hasil kompresi citra tidak merubah kualitas dari citra, hanya mengurangi bit dalam *file* tersebut. Begitu juga pada proses dekompresi, hasilnya akan sama dengan citra aslinya, sekalipun telah dilakukan iterasi baik sekali maupun dua kali iterasi. Jadi metode *huffman* merupakan salah satu alternatif dalam mengkompresi data citra[2].

Dengan hal tersebut penulis menggunakan algoritma *Punctured Elias Codes* untuk mengetahui bagaimana kinerja kompresi, apabila dilakukan dengan mengkompresi citra gambar untuk memberikan manfaat dalam penyimpanan serta membutuhkan ruang memori yang lebih sedikit dibandingkan dengan citra gambar yang tidak dikompresi.

2. METODE PENELITIAN

2.1 Citra

Citra adalah suatu representasi (*image*), kemiripan, atau imitasi dari suatu objek. Citra sebagai keluaran suatu sistem perekaman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal-sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu media penyimpanan[3].

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

2.2 Kompresi Citra

Kompresi citra adalah proses untuk meminimalisasi jumlah bit yang mempresentasikan suatu citra sehingga ukuran data citra menjadi lebih kecil. Namun seringsekali kualitas gambar yang diperoleh jauh lebih buruk dari aslinya karena keinginan untuk memperoleh rasio kompresi yang tinngi. Kompresi citra bertujuan untuk meminimalkan jumlah bit yang diperlukan untuk mempresintasikan citra[3].

2.3 Algoritma *Punctured Elias Codes*

Punctured Elias Codes adalah suatu metode yang dirancang oleh Peter Fenwick dalam sebuah percobaan untuk meningkatkan performa the Burrows–Wheeler transformyang sifatnya lossless. Istilah Punctured datang dari tempat pengawasan eror kode-kode (ECC). ECC terdiri dari data yang asli ditambah sejumlah bilangan dari checkbits. Jika beberapa check bits dihilangkan, untuk mempersingkat serangkaian kode itu, hasil kode ditujukan sebagai berikut :

1. Ambil bilangan biner dari n,
2. Reversed (balikkan bit-bitnya), dan siapkan flag untuk menunjukkan jumlah bit yang bernilai 1 di dalam n.
3. Untuk setiap bit 1 di dalam n kita siapkan flag dari 1 dan akhiri flag dengan 0.
4. Gabungkan flag dengan bilangan biner yang sudah dibalikkan (reversed).

Kode Punctured ini dinamakan kode P1 dan kode ini dimulai dengan 1 (paling sedikit terdapat satu flag, kecuali untuk P1 dengan n=0) dan juga diakhiri 1 (karena n yang asli, yaitu bit MSB (Most significant Bit) Adalah 1, telah dibalikan)[5].

3. ANALISA DAN PEMBAHASAN

Citra merupakan salah satu media yang digunakan dalam berkomunikasi biasanya sebagai alat pengungkapan, pertimbangan (*reason*), interpretasi, ilustrasi, ingatan (*memorise*) dan lain sebagainya. Masalah yang terjadi adalah terkadang ukuran citra yang terlalu besar kapasitasnya membuat sebagian orang merasa terganggu dalam proses penyimpanan di media *stroge* karena banyak memakan *space* dari media penyimpanan yang disediakan. Masalah lain adalah ketika sebuah citra dengan kapasitas besar didistribusikan melalui jalur internet seperti media sosial akan memakan waktu yang lama ketika proses *upload* dan *download*. Masalah-masalah tersebut dapat diatasi dengan aplikasi teknik kompresi.

Teknik kompresi merupakan sebuah teknik yang akan merubah ukuran dari suatu citra asli menjadi lebih kecil kapasitasnya dari ukuran sebelumnya. Perubahan ukuran saat sebuahcitra di kompres akan sedikit mempengaruhi kualitas daricitra tersebut. Salah satu metode yang digunakan dalam teknik kompresi adalah algoritma *Punctured Elias Codes*. Dalam penelitian ini citra yang akan kompresi adalah sebuah gambar citra warna. Citra warna akan dikompres menggunakan algoritma *Punctured Elias Codes* dengan mengambil nilai pixel citra tersebut. Hasil *output* yang dikeluarkan adalah sebuah citra dengan kapasitas yang lebih kecil dari kapasitas sebelumnya dengan kualitas yang tidak jauh berbeda jika dilihat dari mata manusia. Citra yang sudah dikompresi dapat dikembalikan semula kedalam bentuk dan kapasitas aslinya. Proses ini disebut dengan dekompresi.

3.1 Penerapan Algoritma *Punctured Elias Codes*

Diiambil sampel citra untuk keperluan hitungan manual menggunakan metode *Punctured Elias Codes*. Resolusi yang diambil adalah 5 x 5*pixel*. Citra sampel tersebut akan diambil nilai desimal warnanya dengan cara mengektraksi elemen nilai setiap *pixel*.

Gambar 1. Citra Sampel

Berdasarkan proses analisa yang dilakukan diambil sample gambar sebesar 5 x 5 pixel dengan kedalaman warna 24 bit. Proses kompresi yang akan dilakukan menggunakan algoritma Punctured elias codesuntuk mengkompresi citra. Dengan begitu dapat dihitung 5 x 5 x 24 bit =600 bit. Apabila diubah kedalam satuan byte menjadi 600/8 = 75 byte. Nilai desimal elemen warna pada *pixel* akan diekstraksi menggunakan *software matlab*. Sehingga diperoleh nilai desimal *Red Green Blue* (RGB) dari citra sampel resolusi 5 x 5.

Tabel 1. Nilai Desimal Pixel Sample

Pixel	warna	Nilai Desimal	Pixel	Warna	Nilai Desimal
	R	84		R	74
1.	G	82	5.	G	72

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

Pixel	warna	Nilai Desimal	Pixel	Warna	Nilai Desimal
	B	70		B	60
	R	87		R	82
2.	G	85	6.	G	70
	B	73		B	68
	R	82		R	78
	G	80	7.	G	85
3.	B	68		B	73
	R	75		R	87
	G	73	8.	G	85
4.	B	61		B	73
	R	79		R	83
	G	77		G	81
9.	B	65	18.	B	69
	R	80		R	78
	G	78		G	76
10.	B	66	19.	B	64
	R	81		R	80
	G	79		G	78
11.	B	67	20.	B	66
	R	86		R	83
	G	84		G	81
12.	B	72	21.	B	69
	R	84		R	87
13.	G	82	22.	G	85
	B	70		B	73
	R	79		R	84
14.	G	77	23.	G	82
	B	65		B	70
	R	81		R	80
15.	G	79	24.	G	78
	B	67		B	66
	R	83		R	84
16.	G	81	25.	G	82
	B	69		B	70
	R	87			
17.	G	85			
	B	73			

Berdasarkan pada tabel di atas di dapati nilai desimal RGB setiap *pixel* citra sampel. Adapun nilai desimal pada citra sampel 5 x 5 tersebut adalah84, 82, 70, 87, 85, 73, 82, 80, 68, 75, 73, 61, 74, 72, 60, 82, 80, 68, 87, 85, 73, 87, 85, 73, 79, 77, 65, 80, 78, 66, 81, 79, 67, 86, 84, 72, 84, 82, 70, 79, 77, 65, 81, 79, 67, 83, 81, 69, 87, 85, 73, 83, 81, 69, 78, 76, 64, 80, 78, 66, 83, 81, 69, 87, 85, 73, 84, 82, 70, 80, 78, 66, 84, 82, 70.

1. Kompresi Berdasarkan Algoritma *Punctured Elias Codes*
   Pada algoritma *punctured elias codes* terdapat dua kode yaitu kode P1 dan P2 dan yang akan dibahas dipenelitian ini hanya P1 saja. Pada proses kompresi citra, langkah awal adalah membaca nilai pixel citra kemudian membuat tabel nilai *pixel* yang diurutkan dari nilai frekuensi terbesar (nilai *pixel* yang sama) ke terkecil. Urutan nilai pixel dapat dilihat pada tabel di bawah ini :

Tabel 2. Urutan Nilai Bit Pixel Sampel

Nilai		Bit	Frek	Bit x Frek
Des	Biner			
82	01010010	8	7	56
84	01010100	8	6	48
81	01010001	8	5	40
80	01010000	8	5	40
73	01001001	8	5	40
70	01000110	8	5	40
87	01010111	8	4	32
85	01010101	8	4	32
79	01001111	8	4	32

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

Nilai		Bit	Frek	Bit x Frek
Des	Biner			
78	01001110	8	4	32
83	01010011	8	3	24
69	01000101	8	3	24
66	01000010	8	3	24
77	01001101	8	2	24
72	01001000	8	2	24
67	01000011	8	2	24
68	01000100	8	2	24
65	01000001	8	2	24
75	01001011	8	1	8
86	01010110	8	1	8
74	01001010	8	1	8
76	01001100	8	1	8
61	00111101	8	1	8
60	00111100	8	1	8
64	01000000	8	1	8
Total				640

Proses selanjutnya adalah membangkitkan kode Punctured nilai P1 berdasarkan banyaknya nilai desimal pixel pada tabel di atas atau nilai N. Kemudian melakukan kompresi berdasarkan tabel nilai bit *pixel* sampel dengan tabel kode P1 yang telah dibangkitkan. Proses pembangkitan nilai P1 dapat dilihat pada tabel di bawah ini :

Tabel 3. Tabel Kode *Punctured Elias Codes*

N	Binary of n	Reserver	Flag	Flag|Reserver	P1
0	0	-	-	-	0
1	1	1	10	10|1	101
2	10	01	10	10|01	1001
3	11	11	110	110|11	11011
4	100	001	10	10|001	10001
5	101	101	110	110|101	110101
6	110	011	110	110|011	110011
7	111	111	1110	1110|111	1110111
8	1000	0001	10	10|0001	100001
9	1001	1001	110	110|1001	1101001
10	1010	0101	110	110|0101	1100101
11	1011	1101	1110	1110|1101	11101101
12	1100	0011	110	110|0011	1100011
13	1101	1011	1110	1110|1011	11101011
14	1110	0111	1110	1110|0111	11100111
15	111	111	1110	1110|111	1110111
16	10000	00001	10	10|00001	1000001
17	10001	10001	110	110|10001	11010001
18	10010	01001	110	110|01001	11001001
19	10011	11001	1110	1110|11001	111011001
20	10100	00101	110	110|00101	11000101
21	10101	10101	1110	1110|10101	111010101
22	10110	01101	1110	1110|01101	111001101
23	10111	11101	11110	11110|11101	1111011101
24	11000	00011	110	110|00011	11000011

Berdasarkan pada tabel di atas nilai P1 diambil dari gabungan nilai *Flag* dan *Reserver*. Nilai biner *Flag* didapat dengan menambahkan nilai 0 dibelakang jumlah nilai 1 pada nilai biner N. Contoh jika nilai biner N adalah 101, maka nilai *flag*nya adalah 110. Untuk nilai *reserver*didapat di kebalikan nilai biner N.Proses selanjutnya adalah melakukan kompresi nilai dari citra *pixel* sampel dengan nilai kode P1 algoritma *Punctured* yang di dapat dari tabel 3. di atas. Adapun Proses kompresi citra sampel dapat dilihat pada tabel di bawah ini :

Tabel 4. Kompresi Nilai Pixel Dengan P1 *Punctured*

NO	Nilai Des	Punctured Elias Code (P1)	Bit	Frekuensi	Bit x Frek
0	82	0	1	7	7

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

1	84	101	3	6	18
2	81	1001	4	5	20
3	80	11011	5	5	25
4	73	10001	5	5	25
5	70	110101	6	5	30
6	87	110011	6	4	24
7	85	1110111	7	4	28
8	79	100001	6	4	24
9	78	1101001	7	4	28
10	83	1100101	7	3	21
11	69	11101101	8	3	24
12	66	1100011	7	3	21
13	77	11101011	8	2	16
14	72	11100111	8	2	16
15	67	1110111	7	2	14
16	68	1000001	7	2	14
17	65	11010001	8	2	16
18	75	11001001	8	1	8
19	86	111011001	9	1	9
20	74	11000101	8	1	8
21	76	111010101	9	1	9
22	61	111001101	9	1	9
23	60	1111011101	10	1	10
24	64	11000011	9	1	9
		Total			433

Berdasarkan pada tabel di atas dapat dibentuk nilai bit baru hasil kompresi dari susunan nilai desimal pixel citra sampel awal sebelum kompresi yaitu 84, 82, 70, 87, 85, 73, 82, 80, 68, 75, 73, 61, 74, 72, 60, 82, 80, 68, 87, 85, 73, 87, 85, 73, 79, 77, 65, 80, 78, 66, 81, 79, 67, 86, 84, 72, 84, 82, 70, 79, 77, 65, 81, 79, 67, 83, 81, 69, 87, 85, 73, 83, 81, 69, 78, 76, 64, 80, 78, 66, 83, 81, 69, 87, 85, 73, 84, 82, 70, 80, 78, 66, 84, 82, 70.(tanpa tanda koma dan spasi) menjadi nilai bit biner :

"10101101011100111110111100010110111000001110010011000111100110111000101111001111111011
10101101110000011100111110111000111001111101111000110000111101011110100011101111010011
10001110011000011110111111011001101111001111010110101100001111010111101000110011000011
10111110010110011110110111001111101111000111001011001111011011101001111010101110000111
01111010011100011110010110011110110111001111101111000110101101011101111010011100011010
110101"

Sebelum di dapatkan hasil akhir kompresi dilakukan penambahan string bit itu sendiri yaitu padding bit dan *flag* bit. Penambahan (*padding*) string dilakukan jika hasil kompresi dibagi dengan 8 memiliki sisa. Jika hasil kompresi habis dibagi 8 dan tidak memiliki sisa atau nol maka tidak perlu adanya penambahan bit (*padding*). Sedangkan *flag* bit adalah nilai biner yang didapat dari nilai hasil padding. *Flag* bit selalu memiliki jumlah nilai 8 bit biner. Berdasarkan hasil kompresi nilai pixel citra sampel menggunakan algoritma *Punctured Elias Code* (P1) di dapati nilai bit sebanyak 433 bit. Jumlah string 433 bit tidak habis di bagi dengan 8 dan memiliki sisa 1. Karena panjang string tidak habis di bagi 8 maka dapat ditambahkan *padding* "0000000". Dengan demikian, *flag* bit yang merupakan nilai biner dari panjang dari *padding* bit yaitu 7 dalam biner menjadi "00000111". Sehingga string bit yang terbentuk adalah

"10101101011100111110111100010110111000001110010011000111100110111000101111001111111011
10101101110000011100111110111000111001111101111000110000111101011110100011101111010011
10001110011000011110111111011001101111001111010110101100001111010111101000110011000011
10111110010110011110110111001111101111000111001011001111011011101001111010101110000111
01111010011100011110010110011110110111001111101111000110101101011101111010011100011010
110101000000000000111"

Total panjang bit keseluruhan setelah ada penambahan bit dan flag bit adalah 433+7+8 = 448 bit.

10101101=173	01110011=115	11101111=239
00010110=22	11100000=224	11100100=228
11000111=199	10011011=155	10001011=139
11100111=231	11110111=241	01011011=91
10000011=131	10011111=159	01111000=120
11100111=231	11011110=222	00110000=48
11110101=245	11101000=232	11101111=239
01001110=78	00111001=57	10000111=135
10111111=191	01100110=102	11110011=243

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

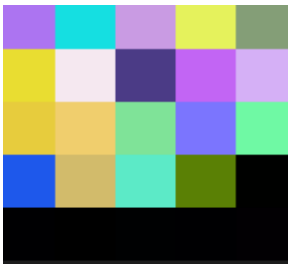
Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

11010110=214 10110000=176 11110101=245
1110100=232 11001100=204 00111101=61
1110010=242 11001111=207 01101110=110
01111101=125 11100011=227 10010110=150
01111011=123 01110100=116 11110101=254
01110000=112 11110111=247 10100111=167
00011110=30 01011001=89 11101101=237
11001111=207 10111100=188 01101011=107
01011101=93 11101001=233 11000111=199
01011010=90 10000000=128 00000111=7

Selanjutnya untuk mendapatkan hasil citra gambar terkompresi maka bit yang telah di susun 8 digit diubah ke decimal dan di susun kembali dalam bentuk 5x5 maka dapat dilihat seperti berikut:

Tabel 5. Hasil nilai decimal terkompresi

173 115 239	22 224 228	199 155 139	231 241 91	131 159 120
231 222 48	245 232 239	78 57 125	191 102 243	214 176 245
232 204 61	242 207 110	125 227 150	123 116 254	112 247 167
30 89 237	207 188 107	93 233 199	90 128 7	0
0	0	0	0	0



Gambar 2. Hasil Kompresi Dengan Penambahan Informasi Dimensi Citra Awal.

Dari hasil kompresi dengan algoritma *Punctured Elias Code* (P1) di atas dapat dihitung kinerja kompresinya menurut parameter yang sudah ditentukan yaitu :

a. *Compression Ratio* (C_R)

$$C_R = \frac{\text{Ukuran Data Setelah Dikompresi}}{\text{Ukuran Data Sebelum Dikompresi}} \times 100\%$$

$$C_R = \frac{448}{640} \times 100\%$$

$$C_R = 70\%$$

2. Dekompresi Berdasarkan Algoritma *Punctured Elias Code*

Pada proses dekompresi hal yang dilakukan adalah menganalisa keseluruhan bit hasil dari kompresi sebelumnya. Adapun bit keseluruhan hasil kompresi dapat dilihat pada tabel berikut :

Tabel 6. Nilai Biner dan Desimal hasil Kompresi keseluruhan

	Nilai			Nilai	
N	Biner	Desimal	N	Biner	Desimal
0	10101101	173	28	11110101	245
1	01110011	115	29	11101000	232
2	11101111	239	30	11001100	204
3	00010110	22	31	00111101	61
4	11100000	224	32	11110010	242
5	11000111	199	33	11001111	207
6	10011011	155	34	01101110	110
7	1000101	96	35	01111101	125

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

Nilai			Nilai		
N	Biner	Desimal	N	Biner	Desimal
8	11100111	231	36	11100011	227
9	11110111	247	37	10010110	150
10	01011011	91	38	01111011	123
11	10000011	131	39	01110100	116
12	10011111	159	40	11110101	245
13	01111000	120	41	01110000	112
14	11100111	231	42	11110111	247
15	11011110	222	43	10100111	167
16	00110000	48	44	00011110	30
17	11110101	245	45	01011001	89
18	11101000	232	46	11101101	237
19	11101111	239	47	11001111	207
20	01001110	78	48	10111100	188
21	00111001	57	49	01101011	107
22	10000111	135	50	01011101	93
23	10111111	191	51	11101001	233
24	01100110	102	52	11000111	199
25	11110011	243	53	01011010	90
26	11010110	214	54	10000000	128
27	10110000	176	55	00000111	7

Ambil keseluruhan bit dan gabungkan seperti berikut :

"10101101011100111110111100010110111000001110010011000111100110111000101111001111111011
10101101110000011100111110111000111001111101111000110000111101011110100011101111010011
10001110011000011110111111011001101111001111010110101100001111010111101000110011000011
10111110010110011110110111001111101111000111001011001111011011101001111010101110000111
01111010011100011110010110011110110111001111101111000110101101011101111010011100011010
11010100000000000111"

a. Tahap Awal

Proses awal dekompresi adalah membaca nilai flag bits dari keseluruhan bit tersebut dengan cara mengubah nilai 8 bit terakhir kedalam nilai desimal seperti di bawah ini :

"10101101011100111110111100010110111000001110010011000111100110111000101111001111111011
10101101110000011100111110111000111001111101111000110000111101011110100011101111010011
10001110011000011110111111011001101111001111010110101100001111010111101000110011000011
10111110010110011110110111001111101111000111001011001111011011101001111010101110000111
01111010011100011110010110011110110111001111101111000110101101011101111010011100011010
11010100000000000111".

Didapati nilai biner 8 bit terakhir sebagai berikut,00000111 dalam desimal adalah merupakan nilai 7. Nilai 7 tersebut menandakan bahwa hasil kompresi sebelumnya habis di bagi 8 sehingga ada penambahan *padding* dan *plag bits*. Selanjutnya hapus *padding* dan *Flags bits* dari nilai keseluruhan bit sehingga menjadi seperti di bawah ini:

"10101101011100111110111100010110111000001110010011000111100110111000101111001111111011
10101101110000011100111110111000111001111101111000110000111101011110100011101111010011
10001110011000011110111111011001101111001111010110101100001111010111101000110011000011
10111110010110011110110111001111101111000111001011001111011011101001111010101110000111
01111010011100011110010110011110110111001111101111000110101101011101111010011100011010
11010".

b. Pengecekan Bit

Selanjutnya adalah melakukan cek bit dari bit pertama dengan tabel kode *Punctured* P1 pada tabel 3.4 di atas. Jika ditemukan bit yang sesuai dengan tabel kode P1 di atas maka ubah nilai string yang sesuai. Sehingga didapati hasil seperti tabel di bawah ini:

Tabel 7. Dekompresi Nilai Pixel Citra Sampel

N	**Punstured P1 Code**	**Nilai Des**	**N**	**Punstured P1 Code**	Nilai Des
0	101	84	38	110101	70
1	0	82	39	100001	79
2	110101	70	40	11101011	77
3	110011	87	41	11010001	75
4	1110111	85	42	1001	81

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

N	Punstured P1 Code	Nilai Des	N	Punstured P1 Code	Nilai Des
5	10001	73	43	100001	79
6	0	82	44	1110111	67
7	11011	80	45	1100101	83
8	1000001	68	46	1001	81
9	11001001	75	47	11101101	69
10	10001	73	48	110011	87
11	111001101	61	49	1110111	85
12	11000101	74	50	10001	73
13	11100111	72	51	1100101	83
14	1111011101	60	52	1001	81
15	0	82	53	11101101	69
16	11011	80	54	1101001	78
17	1000001	68	55	111010101	76
18	110011	87	56	11000011	64
19	1110111	85	57	11011	80
20	10001	73	58	1101001	78
21	110011	87	59	1100011	66
22	1110111	85	60	1100101	83
23	10001	73	61	1001	81
24	100001	79	62	11101101	69
25	11101011	77	63	110011	87
26	11010001	65	64	1110111	85
27	11011	80	65	10001	73
28	1101001	78	67	101	84
29	1100011	66	68	0	82
30	1001	81	69	110101	70
31	100001	79	70	11011	80
32	1110111	67	71	1101001	78
33	111011001	86	72	1100011	66
34	101	84	73	101	84
35	11100111	72	74	0	82
36	101	84	75	110101	70
37	0	82			

Berdasarkan hasil dekompresi di atas didapati nilai desimal awal pixel citra sampel sebelum kompresi sebagai berikut :84, 82, 70, 87, 85, 73, 82, 80, 68, 75, 73, 61, 74, 72, 60, 82, 80, 68, 87, 85, 73, 87, 85, 73, 79, 77, 65, 80, 78, 66, 81, 79, 67, 86, 84, 72, 84, 82, 70, 79, 77, 65, 81, 79, 67, 83, 81, 69, 87, 85, 73, 83, 81, 69, 78, 76, 64, 80, 78, 66, 83, 81, 69, 87, 85, 73, 84, 82, 70, 80, 78, 66, 84, 82, 70.Nilai desimal pixel hasil dekompresi keseluruhan dapat dilihat pada tabel di bawah ini :

Tabel 8. Nilai Desimal Pixel Hasil Dekompresi

Pixel	warna	Nilai Desimal	Pixel	warna	Nilai Desimal
1.	R	84	5.	R	74
	G	82		G	72
	B	70		B	60
2.	R	87	6.	R	82
	G	85		G	80
	B	73		B	68
3.	R	82	7.	R	87
	G	80		G	85
	B	68		B	73
4.	R	75	8.	R	87
	G	73		G	85
	B	61		B	73
9.	R	79	18.	R	83
	G	77		G	81
	B	65		B	69
10.	R	80	19.	R	78
	G	78		G	76
	B	66		B	64

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

Pixel	warna	Nilai Desimal	Pixel	warna	Nilai Desimal
	R	81		R	80
11.	G	79	20.	G	78
	B	67		B	66
	R	86		R	83
12.	G	84	21.	G	81
	B	72		B	69
	R	84		R	87
13.	G	82	22.	G	85
	B	70		B	73
	R	79		R	84
14.	G	77	23.	G	82
	B	65		B	70
	R	81		R	80
15.	G	79	24.	G	78
	B	67		B	66
	R	83		R	84
16.	G	81	25.	G	82
	B	69		B	70
	R	87			
17.	G	85			
	B	73			

Berdasarkan hasil dekompresi pada tabel 8. didapati nilai hasil dekompresi sama dengan nilai sebelum citra dikompresi, sehingga dapat diambil kesimpulan bahwa algoritma punctured elias codes memiliki jenis kompresi *lossless*.Hasil citra sample dapat dilihat seperti gambar berikut:

Gambar 3. Hasil gambar yang telah di dekompresi

3.2 Pengujian Program

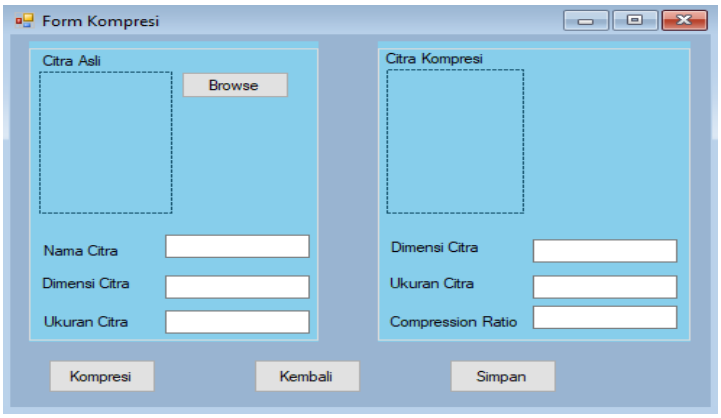
Tampilan program merupakan tampilan dari aplikasi perancangan aplikasi kompresi citra dengan menerapkan algoritma *punctured elias codes*. Aplikasi yang akan dijalankan telah dibangun dengan menggunakan aplikasi *Microsoft Visual Studio* 2008 dengan bahasa pemograman *visual basic*. Tampilan aplikasi program yang dibutuhkan diantaranya yaitu tampilan *input*, tampilan proses dan tampilan *output*. Tampilan *input* terdiri dari *interface* aplikasi, tampilan *output* terdiri dari hasil proses kompresi. *Form menu* utama merupakan form yang pertama kali muncul saat aplikasi dijalankan. *Form menu* utama memiliki beberapa sub *menu* diantaranya adalah *menu form* kompresi, *menu form* dekompresi, *menu form* bantuan, *menu form about me*. Adapun tampilan halaman menu utama pada aplikasi dapat dilihat pada gambar 3.

Gambar 4. Tampilan Form Menu Utama

Form kompresi merupakan *form* yang digunakan untuk melakukan proses kompresi. Pada *form* ini disediakan tombol *button browse* untuk membuka file atau gambar yang akan diproses, *picturebox* untuk menampilkan gambar yang dipilih, *textbox* untuk menampilkan informasi nama, dimensi dan ukuran pada gambar yang dikompresi, serta *button* dan beberapa *textbox* informasi hasil kompresi. Tampilan *form* kompresi dapat dilihat pada gambar 5.

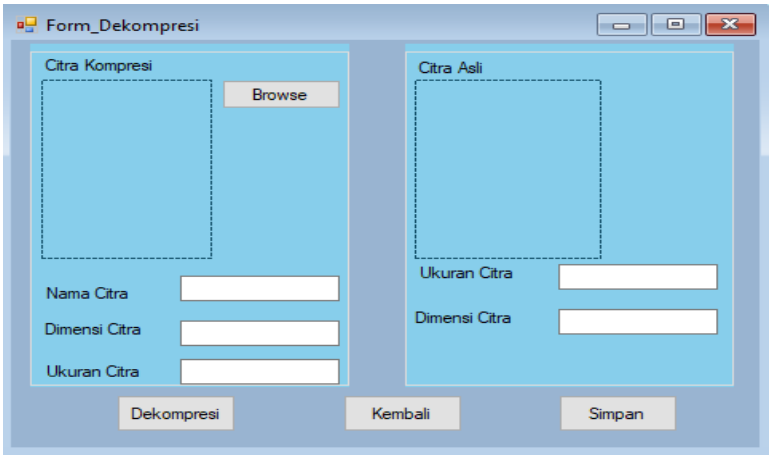
Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)



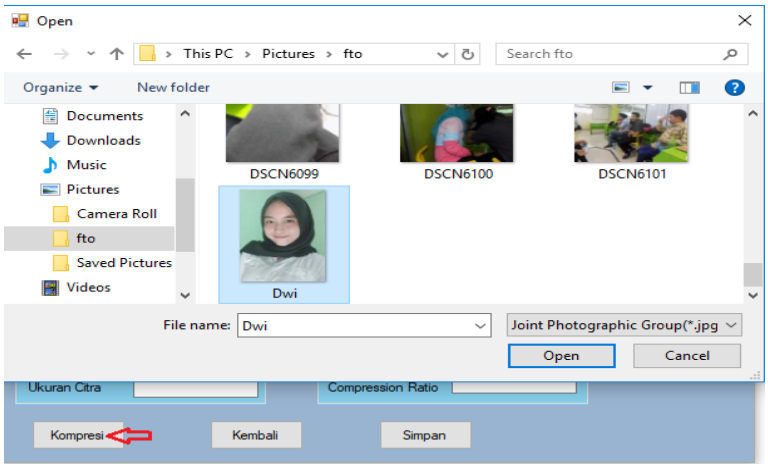
Gambar 5. Form Kompresi

Form dekompresi merupakan *form* yang digunakan untuk melakukan proses dekompresi. Pada *form* ini disediakan tombol *button browse* untuk memilih atau mencari gambar yang akan didekompresi, *picturebox* untuk menampilkan gambar yang telah didekompresi, *textbox* untuk menampilkan informasi nama, dimensi dan ukuran pada gambar yang didekompresi, *button* dekompresi untuk mendekompresi gambar, serta beberapa textbox untuk informasi hasil dekompresi. Tampilan *form* dekompresi dapat dilihat pada gambar 6.



Gambar 6. Form Dekompresi

Memilih gambar kompresi dilakukan ketika *user* akan melakukan proses kompresi. Adapun proses pemilihan gambar dapat dilihar pada gambar 7.

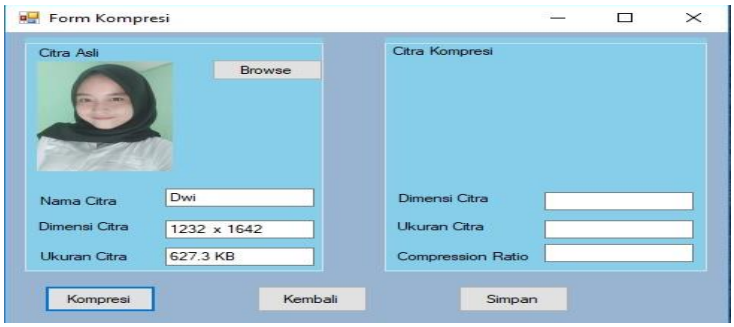


Gambar 7. Tampilan ketika memilih gambar kompresi

Berdasarkan pada gambar 7. proses yang harus dilakukan ketika pengguna (*user*) melakukan kompresi gambar adalah memilih tombol *browse* dimenu aplikasi kompresi. Tampilan direktori komputer akan terbuka dan *user* yang sudah memilih gambar dapat menekan tombol *open*, kemudian gambar yang dipilih akan tampil pada *picturebox* aplikasi, dapat dilihat pada gambar 8.

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

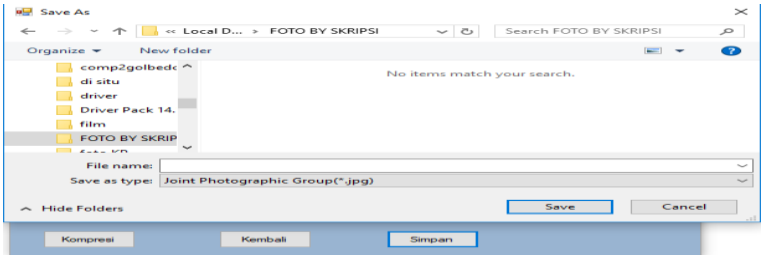


Gambar 8. Tampilan Gambar Kompresi Yang Dipilih

Berdasarkan pada gambar 8 aplikasi menampilkan sebuah gambar yang sudah dipilih oleh pengguna (*user*) serta nama, dimensi dan ukuran dari gambar yang dipilih. Proses seanjutnya yaitu melakukan kompresi dengan memilih tombol kompresi. Tampilan kompresi citra yaitu *form* yang menampilkan hasil dari kompresi citra. Ada tampilan hasil kompresi citra dapat dilihat pada gambar 9.

Gambar 9. Tampilan Gambar Hasil Kompresi

Berdasarkan gambar 9 untuk memulai proses kompresi dapat dilakukan dengan cara menekan tombol kompresi, hasil kompresi gambar sebelumnya mengalami perubahan pada dimensi gambar dan ukuran gambar serta menampilkan rasio kompresi. Pada proses ini akan menampilkan *output* yang akan keluar ketika gambar hasil kompresi yang akan disimpan. Adapun tampilan simpan gambar hasil kompresi dapat dilihat pada gambar 10.



Gambar 10. Tampilan Simpan Gambar Hasil Kompresi

Berdasarkan pada gambar 10 proses simpan dapat dilakukan ketika gambar sudah dikompresi dengan menekan tombol simpan. Tampilan direktori penyimpanan pada komputer akan tampil kemudian pengguna (*user*) dapat memilih tombol *save*. Proses dekompresi yaitu proses pengembalian ukuran gambar pada ukuran semula sebelum kompresi. Proses penginputan gambar dilakukan dengan cara yang sama saat proses kompresi. Adapun proses dekompresi dapat dilihat pada gambar 11.

Gambar 11. Tampilan Proses Dekompresi

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**

Building of Informatics, Technology and Science (BITS)
Volume 2, No 2, December 2020 Page 176-187
ISSN 2684-8910 (media cetak)
ISSN 2685-3310 (media online)

Berdasarkan pada gambar 11 proses dekompresi dimulai dengan memilih tombol dekompresi. Hasil dekompresi adalah gambar awal dengan ukuran dan dimensi yang kembali seperti semula saat sebelum proses kompresi. Hasil gambar dekompresi dapat simpan dengan memilih tombol simpan.

4. KESIMPULAN

Berdasarkan hasil analisa yang telah dilakukan, maka penulis mengambil kesimpulan prosedur kompresi citra dengan menerapkan algoritma *punctured elias codes* dapat dilakukan dengan melakukan langkah-langkah untuk melakukan proses kompresi dan dekompresi, sehingga aplikasi yang diharapkan dapat berjalan sesuai dengan keinginan. Algoritma yang digunakan dalam penelitian ini adalah algoritma *punctured elias codes*, dengan algoritma ini dapat mengetahui kinerja kompresi apabila dilakukan dengan mengkompresi citra untuk memberikan manfaat penyimpanan serta ruang memori yang lebih sedikit.

REFERENCES

[1] D. A. Yansyah and I. Pendahuluan, “PERBANDINGAN METODE PUNCTURED ELIAS CODE DAN,” vol. 2, no. 6, pp. 33–36, 2015.
[2] T. R. Silviani and A. Arfiana, “Teknik Kompresi Citra Menggunakan Metode Huffman,” pp. 93–100, 2016.
[3] D. T. Sutoyo, S.Si., M.Kom., Teori pengolahan citra digital. Semarang: ANDI yogyakarta, 2009.
[4] A. Yogyakarta, Pengolahan Citra Digital. yogyakarta: CV.ANDI OFFSET, 2010.
[5] G. M. David Salomon, Handbook of Data Compression. London: Spinger - Verlag London Limited 2010, 2010.
[6] A. K. dan A. Susanto, Teori dan Aplikasi Pengolahan Citra. Yogyakarta: CV.ANDI OFFSET, 2013.
[7] Y. Sugiarti, Analisa Dan Perancangan UML (Unified Modeling Language). yogyakarta: Ed. Yogyakarta, 2013.
[8] M. S. Rosa A. S., Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek. Bandung: INFORMATIKA bandung, 2014.
[9] WAHANA KOMPUTER, Membangun Aplikasi Toko Dengan Visual Basic 2008. Yogyakarta: C.V ANDI OFFSET, 2009.
[10] P. Hidayatullah, Visual Basic. Net membuat aplikasi database dan program kreatif. Bandung: INFORMATIKA Bandung, 2014.
[11] S. Sari, "Penerapan Metode Median Filter untuk Mereduksi Noise Speckle dan Salt & Pepper pada Citra Ortokromatik," Building Informatics, Technology and Science (BITS), vol. 1, no. 1, pp. 34-41, 2019.
[12] A. Novian, "Perancangan Aplikasi Denoise Citra Dengan Menerapkan New Daptive Based Median Filter," Building Informatics, Technology and Science (BITS), vol. 1, no. 1, pp. 42-47, 2019.

Submitted: **09/06/2020**; Accepted: **10/12/2020**; Published: **10/12/2020**