

Implementasi Algoritma Ternary Comma Code (TCC) Dalam Mengkompresi File Audio Pada Aplikasi Perekam Suara

Ayu Nita Irwanda

Fakultas Ilmu Komputer dan Teknologi Informasi, Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia

Email: ¹ayunitairwanda03@gmail.com

Abstrak—Teknik Dalam mengkompresi File Audio biasa digunakan untuk proses transmisi dan penyimpanan yaitu antara lain untuk mengurangi bottleneck pada proses input ataupun output, penyimpanan data lebih hemat ruang serta mempermudah distribusi data dan file dengan media elektronik maupun media removable. Perekam suara merupakan suatu perangkat elektronik yang mengambil suara atau gelombang frekuensi dan dirancang sesuai bentuk sebuah aplikasi yang bisa di simpan dalam format digital atau jaringan lainnya . Aplikasi perekam suara ini banyak sekali digunakan di era modern saat ini juga sangat membantu dalam kehidupan sehari hari kita misal aplikasi menulis, game, video maupun suara namun harus di sesuaikan dengan kebutuhan. Berdasarkan implementasi Algoritma yang di pakai adalah Ternary Comma Code (TCC). Metode ini merupakan suatu bilangan biner (baris2) yang didasarkan pada 2 nomor bit 0 dan 1. Dalam penggunaan metode algoritma TCC ini memungkinkan data file audio pada aplikasi perekam suara dapat mengurangi redundansi dari file untuk menyimpan atau mengirim dalam bentuk yang efisien serta dengan begitu kita bisa mengompres file penting tersebut lebih hemat ruang memori, Karena sebuah sistem penyimpanan memerlukan ruang yang cukup besar. Penggunaan ukuran data semakin lama semakin besar jadi proses yang dapat mengubahnya dengan cara mamampatkan data asli.

Kata Kunci: Kompresi; File Audio; Ternary Comma Code (TCC); Perekam Suara

Abstract—Techniques in compressing audio files are commonly used for transmission and storage processes, namely to reduce bottlenecks in the input or output process, save data storage space and facilitate the distribution of data and files with electronic media and removable media. A voice recorder is an electronic device that picks up sound or frequency waves and is designed to fit an application that can be saved in digital or other network formats. This voice recorder application is widely used in the modern era today is also very helpful in our daily life, for example writing applications, games, video and voice, but must be adjusted as needed. Based on the implementation of the algorithm used is the Ternary Comma Code (TCC). This method is a binary number (line2) which is based on 2 bit numbers 0 and 1. In the use of the TCC algorithm method, it allows audio file data in voice recording applications to reduce the redundancy of files to save or send in an efficient form and so we can compress these important files to save more memory space, because a storage system requires a large enough space. The use of the data size is getting bigger and bigger so the process can change it by compressing the original data.

Keywords: Compression; Audio File; Ternary Comma Code (TCC); Voice Recorder

1. PENDAHULUAN

Kompresi merupakan suatu proses perubahan informasi atau sekumpulan data menggunakan bit atau information dalam bentuk kode untuk menghemat kebutuhan sesuai dengan tempat penyimpanan dan waktu dalam mentransmisi data. Kompresi berarti menempatkan atau mengecilkan ukuran. Pengiriman hasil kompresi *file* dapat dilakukan jika dari pihak pengirim yang melakukan kompresi *file* dan dari pihak penerima mempunyai aturan yang sama dalam hal kompresi data.

Berdasarkan hasil dari pengamatan sebelumnya dilakukan oleh Maydina Sari Nasution tahun 2018 telah terbukti mampu mengkompresi *file* citra digital dengan menggunakan metode algoritma *Ternary Comma Code (TCC)* dengan cukup baik dan dapat diimplementasikan kedalam sistem tanpa adanya kesalahan dalam penggunaan sistem baik pada saat proses kompresi maupun dekompresi *file* citra digital[1]. Penulis memilih metode *Ternary Comma Code (TCC)* berdasarkan dari penelitian sebelumnya yang dilakukan oleh Desi Husni dalam skripsinya yang berjudul “Implementasi dan Analisis Algoritma Kompresi *Punctured Elias Codes* dan *Ternary Comma Code* Pada *File Doc*” berdasarkan pengujian kompresi terhadap *file Doc*, dapat disimpulkan bahwa kompresi dengan Metode *Ternary Comma Code* waktu kompresi dan dekompresinya yang di hasilkan jauh lebih besar dibandingkan dengan algoritma *Punctured Elias Codes*[2].

Perekam suara merupakan suatu perangkat elektronik yang mengambil suara atau gelombang frekuensi dan dirancang sesuai bentuk sebuah aplikasi yang bisa di simpan dalam format digital atau jaringan lainnya . Aplikasi perekam suara ini banyak sekali digunakan di era modern saat ini juga sangat membantu dalam kehidupan sehari hari kita misal aplikasi menulis, *game*, video maupun suara namun harus di sesuaikan dengan kebutuhan. Untuk membuat sebuah audio pada perekam suara di mulai dari proses *recording* yaitu suatu perekaman, pencatatan, ataupun pengabadian suara. Oleh karena itu Jika proses perekaman memiliki durasi waktu yang cukup lama, maka audio yang di gunakan akan mempunyai *file* yang makin besar sehingga ukuran tersebut harus di kompres atau dikurangi kualitasnya agar ukuran *filenya* kecil sehingga dapat memuat banyak ruang yang bisa digunakan sesuai kebutuhan.

Berdasarkan teknik atau metodologi kompresi maka metode yang paling tepat untuk mengkompresi *file* audio pada aplikasi perekam suara adalah metode *Ternary Comma Code (TCC)* yang menggunakan teknik lossless yaitu dimana data dari *file* audio hasil kompresi dapat dikembalikan ke *file* sebelum dikompres tanpa menghilangkan informasi pada data sebelumnya. *Ternary Comma Code* adalah suatu bilangan biner (baris2) yang didasarkan pada 2 nomor bit 0 dan 1. Dalam penggunaan metode algoritma TCC ini memungkinkan data *file* audio pada aplikasi perekam suara dapat mengurangi redundansi dari *file* untuk menyimpan atau mengirim dalam bentuk yang efisien serta dengan begitu kita bisa mengompres *file* penting tersebut lebih hemat ruang memori, Karena sebuah sistem penyimpanan memerlukan ruang yang cukup besar.

Penggunaan ukuran data semakin lama semakin besar jadi proses yang dapat mengubahnya dengan cara mamampatkan data asli.

2. METODOLOGI PENELITIAN

2.1 Kompresi

Kompresi yaitu suatu teknik memperkecil serta memadatkan *file* yang berukuran besar menjadi lebih kecil dan mengurangi kebutuhan ruang penyimpanan. Proses kompresi merupakan proses yang mendekati pada minimal seluruh bit pada cabang digital seperti gambar, audio, dan video, yang menghasilkan ukuran data yang lebih kecil namun tetap menjaga kuantitas informasi dalam data tersebut[3].

Kompresi audio adalah proses memperkecil atau meminimalisasi jumlah tiap bit yang merepresentasikan suatu audio dengan data yang menjadi lebih kecil. Kompresi juga dapat diartikan sebagai cara untuk memperkecil data agar diperoleh data dengan bentuk yang lebih kecil dari pada bentuk aslinya sehingga lebih efisien dalam menyimpannya atau mempersingkat waktu pertukaran data tersebut.

Ada beberapa teknik yang dijadikan parameter untuk menunjukkan kualitas atau kinerja dari suatu metode kompresi, yaitu:

1. *Compression Ratio* (RC)

Compression ratio merupakan penguraian data atau *file* yang telah di kompresi di dapat dari reaksi perumpamaan antara bentuk data yang belum di kompresi dengan yang sudah di kompresi.

$$Cr = \frac{\text{Ukuran Data Setelah Di Kompresi}}{\text{Ukuran Data Sebelum Di Kompresi}} \times 100\% \quad (1)$$

2. *Redudancy*

Redundancy merupakan pengarsipan data dapat saja diartikan sebagai informasi yang bertindih, beberapa basis data yang sama, karakteristik yang mirip, struktur yang sama dan lainnya.

$$Rd = \frac{\text{file sebelum di kompresi} - \text{file setelah di kompres}}{\text{Ukuran file sebelum di kompresi}} \times 100 \quad (2)$$

2.2 Algoritma Ternary Comma Code (TCC)

Nilai *Binary* (baris2) merupakan dua bit yaitu 0 dan 1 sama dengan nilai *Ternary* (baris3) bersumber pada 3 digit (trit) 0, 1 dan 2. Pada Setiap trit (angka) biasa di kodekan dalam dua bit, namun dua bit tersebut mempunyai empat nilai. Jadi, hal ini terlihat nalar dalam bekerja dengan memakai sistem nilai *Ternary* setiap dimana trit (angka) mewakili dua bit dalam sebuah penilaian menjadi tiga trit (angka) ditemukan pada tanda keempat, yakni *Comma* (c). Namun Sekali kali letakkan tanda c, agar dapat memudahkan dalam merancang *Ternary Comma Code* mengikuti bilangan bulat. *Comma Code* pada n yaitu wujud biasa dari perwakilan Ternary dari $n - 1$ yang dibuat oleh c[1].

Rumus ini telah ada tapi berkembang. Sebab itu. Bertepatan pada aplikasi dimana bilangan bulat yang nilainya yaitu khusus. Rumus ini juga sangat gampang untuk di *-encode* dan di *-decode*- kan dan hal yang sangat prinsip ialah *Comma Symbol* (sebagai tanda akhir sebuah kode) yang dibutuhkan adalah dua bit. Ketidak efektifan ini tidak lah hal yang cukup besar, tapi akan menjadi lebih tinggi bagi *Comma Code* yang berlandaskan pada nilai dasar yang besar. Pada nilai basis 15 *Comma Code*, misalnya setiap 15 digit mendahulukan 4 bit dan *The Comma*-nya juga 4 bit. bentuk Setiap rumus di akhiri dengan 4 bit *Comma*, tidak tertulis dengan konsep minimal dari satu bit dan sifat ini membangun kode terlihat tidak efektif, bagaimana pun juga kerangkapan per simbol menurun untuk jumlah besar basis. pada sistem basis 7, salah satu dari delapan simbol dikorbankan untuk *Comma*, namun bebrapa dalam basis 15 itu merupakan salah satu dari 16 simbol. Dari kompresi dengan menggunakan algoritma *Ternary Comma Code*(TCC) ditemukan tabel basis 15 itu yaitu salah satu dari 16 simbol.

Tabel 1. Ternary Comma Code

Value	Code	L	Value	Code	L
0	C	2	11	101c	8
1	1c	4	12	102c	8
2	2c	4	13	110c	8
3	10c	4	14	111c	8
4	11c	6	15	112c	8
5	12c	6	16	120c	8
6	20c	6	17	121c	8
7	21c	6	18	122c	8
8	22c	6	19	200c	8
9	100c	6	20	201c	8
....			...		
64	2001c	10	1,000	1101000c	16
128	11201c	12	3,000	11010002c	18

256	100110c	14	10,000	111201100c	20
512	200221c	14	65,536	10022220020c	24

Sumber : M. S. Nasution, 2018[1]

Contoh:

$$CR = \frac{40}{75} \times 100\% = 0,5333 \times 100\% = 53,33 \%$$

Penjelasan:

40 adalah ukuran data setelah di kompres, 75 adalah ukuran data sebelum di kompresi. Untuk menghitung CR digunakan rumus 2.1 dimana hasil dari CR berbentuk persentase seperti contoh dimana persentase CR ialah 53,33%.

3. HASIL DAN PEMBAHASAN

3.1 Analisa Masalah

Analisa adalah sebuah proses dalam mengambil sesuatu kedalam bagian bagian yang saling berhubungan satu sama lainnya secara detail pada suatu hal atau benda dan menyusun komponen tersebut untuk di kaji lebih lanjut. Pada tahap analisa ini sangat penting karna proses analisis yang akurat akan menghasilkan sebuah *software* yang dapat digunakan untuk pengguna.

Audio merupakan salah satu elemen yang sangat penting, karena ikut berperan dalam membangun sebuah sistem komunikasi dalam bentuk suara. Audio juga salah satu media yang populer di masyarakat, saat ini banyak masyarakat yang mengoleksi *file* audio dengan cara mengunduh di internet tanpa melihat kapasitas *file* audio tersebut dan tidak peduli sebesar apa media penyimpanan yang dimiliki. Jadi masyarakat terkadang harus perlu dalam kompresi *file* audio yang dimilikinya agar dapat menyimpan dan mengirim dengan jumlah yang banyak sehingga tidak memenuhi ukuran kapasitas dalam memori. Maka kegunaan kompresi *file* audio sangat dibutuhkan dalam menghemat ruang pada perangkat atau aplikasi yang di gunakan, semakin rendah tingkat kompresi yang dilakukan maka semakin kecil ukuran *file* yang di dapat.

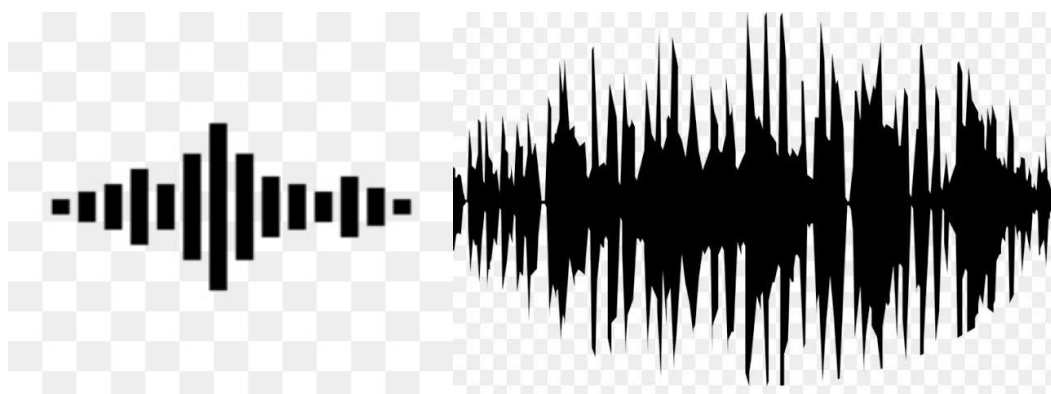
Pada bab ini akan dilakukan analisa tentang cara atau tingkatan yang akan diproses untuk mengkompresi *file* audio dengan menggunakan algoritma *Ternary Comma Code (TCC)* serta perancangan perangkat lunak kompresi *file* audio. Langkah awal dari proses kompresi yang akan dilakukan adalah mengambil nilai heksadesimal melalui media aplikasi *binary viewer*. Berikutnya nilai heksadesimal akan dikompresi dengan mengikuti langkah-langkah dari algoritma *Ternary Comma Code*. Selanjutnya *file* audio yang telah dikompresi dapat di kembalikan dengan melakukan proses dekompresi *file* audio.

3.1.1 Penerapan Algoritma Ternary Comma Code (TCC)

File audio yang berformat WAV merupakan *file* yang jarang sekali digunakan dan tidak begitu populer diantara *file* yang lainnya. Dalam menganalisa *file* WAV yang harus dilakukan adalah mengambil *sample file* WAV dengan melakukan pembacaan *file* WAV. Pembacaan *file* WAV dilakukan untuk mendapatkan nilai dari data pada sebuah *file* WAV yang berupa bilangan *hexadesimal*. Dalam proses hitungan manual nilai *sample file* WAV diambil menggunakan *software Binary Viewer*, kemudian data nilai *hexadesimal* WAV di kompresi menggunakan algoritma *Ternary Comma Code (TCC)*. Berikut informasi objek *file* WAV yang akan diambil *sample*nya sebelum dilakukan kompresi.

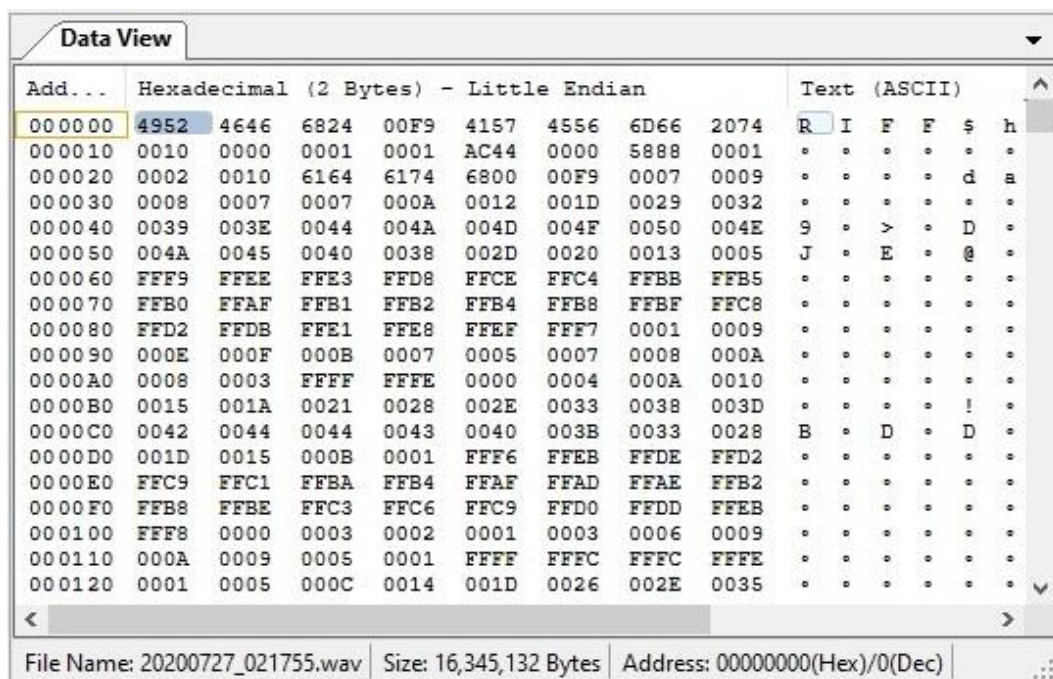
Tabel 2. Informasi File Wav Sample

Keterangan	
Jenis File	WAV
Judul	Rekaman Lagu Naomi Scott - Speechless
Ukuran	15,5 MB
Durasi	3,05 Menit



Gambar 1. Gelombang Suara WAV

Pada susunan ini akan melakukan proses bilangan heksadesimal adalah bilangan heksadesimal yang tertera pada *file* Audio. Untuk mendapatkan nilai heksadesimal dari *file* audio yang akan dikompresi ialah dengan menggunakan aplikasi *Binary Viewer*.



Gambar 2. Nilai heksadesimal *file* audio

Berdasarkan tabel di atas sebelum dikompres nilai heksadesimal harus dikelompokkan seperti tabel di bawah ini :

Tabel 3. Nilai Heksadesimal Sebelum Dikompresi

No	Heksadesimal	ASCII Binary	Frekuensi	Bit	Frekuensi x Bit
1	49	01001001	1	8	8
2	52	01010010	1	8	8
3	46	01000110	2	8	16
4	68	01101000	1	8	8
5	24	00100100	1	8	8
6	00	00000000	1	8	8
7	F9	11111001	1	8	8
8	41	01000001	1	8	8
9	57	01010111	1	8	8
10	45	01000101	1	8	8
11	56	01010110	1	8	8
12	6D	01101101	1	8	8
13	66	01100110	1	8	8
14	20	00100000	1	8	8
15	72	01110010	1	8	8
Jumlah					128

1. Proses Kompresi

Proses Kompresi untuk algoritma *Ternary Comma Code* bisa dilihat menurut tabel di bawah ini:

Tabel 4. Nilai Heksadesimal yang telah dikompresi dengan algoritma *Ternary Comma Code*

No	Heksadesimal	Ternary Comma Code	Biner	Frekuensi	Bit	Frekuensi x Bit
1	46	1c	10	2	1	2
2	49	2c	11	1	2	2
3	52	10c	100	1	2	3
4	68	11c	101	1	3	3
5	24	12c	110	1	3	3
6	00	20c	111	1	3	3
7	F9	21c	1000	1	3	4
8	41	22c	1001	1	4	4

No	Heksadesimal	Ternary Comma Code	Biner	Frekuensi	Bit	Frekuensi x Bit
9	57	100c	1010	1	4	4
10	45	101c	1011	1	4	4
11	56	102c	1100	1	4	4
12	6D	110c	1101	1	4	4
13	66	111c	1110	1	4	4
14	20	112c	1111	1	4	4
15	72	120c	10000	1	4	5
Jumlah						53

Proses mendapatkan kode *ternary* yaitu bilangan bulat dari n-1 yang diikuti oleh c. Sebagai contoh diambil dari tabel 4. dengan nilai n=13. Jika n-1, maka 13-1 = 12 lalu dilakukan pembangian 3 dan dihasilkan nilai *ternary* 1101 dan masukkan simbol *comma code* (c). Sehingga nilai *ternary* dari n=13=1101c.

Berdasarkan nilai heksadesimal sebelum dikompresi yaitu “49, 52, 46, 46, 68, 24, 00, F9, 41, 57, 45, 56, 6D, 66, 20, 74”, dihasilkan nilai heksadesimal yang telah terkompresi dengan algoritma *ternary comma code* “1c 2c 10c 11c 12c 20c 21c 22c 100c 101c 102c 110c 111c 112c 120c”.

Dalam sebuah perhitungan komputer tidak mengenali basis 3 (0, 1 dan 2), maka untuk itu mencari nilai bit di atas digunakan rumus n-1 dan diubah ke basis 2 (0 dan 1). Misalnya nilai n =13-1=12 maka nilai bit dari 13=1101. Setelah mendapatkan nilai bit, masukkan nilai bit dari *comma code* (c). Nilai c didapat dari jumlah n terakhir dari tabel 3.3. Diketahui nilai terakhir dari tabel 3.3 (n=15) maka bit dari c (n=16) sehingga nilai bit dari c=10000. Gabungkan nilai bit dari setiap nilai heksadesimal tambahkan nilai bit dari c. Dari penggabungan bit dihasilkan bit seperti berikut “10100001 11000010 01000010 11000011 01000011 11000010 00100001 00110000 10101000 01011100 00110010 00011011 00001110 10000111 11000010 00010000”.

Sebelum ditulis ke sebuah *file* hasil kompresi dilakukan penambahan *string* bit itu sendiri apakah habis dibagi 8 dan berapa sisanya jika tidak habis dibagi 8. Jumlah *string* bit setelah dikompresi adalah 53 ditambah *string* bit dari c. *string* bit dari c yaitu 5 lalu dikali dengan jumlah nilai heksadesimal sebanyak 15 maka didapatkan *string* bit dari c sebanyak 75. Jadi total keseluruhan bit 53+75=128 dibagi 8 dihasilkan 16 bit dan tidak memiliki sisa hasil bagi. Lakukan pemisahan bit menjadi beberapa kelompok yang terdiri dari 8 bit serupa tabel yang ada di bawah ini :

Tabel 5. Pembagian Bit

10100001	11000010	01000010	11000011
01000011	11000010	00100001	00110000
10101000	01011100	00110010	00011011
00001110	10000111	11000010	00010000

Berdasarkan pembagian nilai biner dihasilkan 16 kelompok nilai biner baru yang sudah terkompresi. Nilai biner yang telah dikelompokkan dirubah ke dalam suatu karakter. Sebelum dirubah terlebih dahulu mencari nilai desimal dari *string* bit tersebut dengan menggunakan kode ASCII. Karakter dari hasil kompresi dirubah dengan menggunakan excel agar lebih memudahkan untuk mencari karakter tersebut.

Tabel 6. Biner ke karakter

10100001 = i	11000010 = Â	01000010 = B	11000011 = Ã
01000011 = C	11000010 = Â	00100001 = !	00110000 = 0
10101000 = "	01011100 = \	00110010 = 2	00011011 =
00001110 =	10000111 = ‡	11000010 = Â	00010000 =

Berdasarkan hasil kompresi dengan algoritma *ternary comma code* dan penambahan nilai *string* sebesar 32 bit, maka perhitungan kinerja kompresi sesuai parameter yang sudah ditentukan yaitu sebagai berikut :

a. *Ratio Of Compression* (Rc)

$$Rc = \frac{\text{ukuran data sebelum dikompresi}}{\text{ukuran data setelah dikompresi}}$$

$$Rc = \frac{128 \text{ bit}}{53 \text{ bit}}$$

$$Rc = 2,41$$

b. *Compression ration* (Cr)

$$Cr = \frac{\text{Ukuran Data Setelah di Kompresi}}{\text{Ukuran Data Sebelum di Kompresi}} \times 100\%$$

$$Cr = \frac{53 \text{ bit}}{128 \text{ bit}} \times 100\%$$

$$Cr = 41,40\%$$

2. Proses Dekompresi

Proses dekompresi yaitu mengembalikan karakter menjadi nilai desimal dan dirubah menjadi menjadi biner (bit). Dalam proses pengembalian nilai biner digunakan aplikasi microsoft excel.

Tabel 7. Merubah karakter ke nilai biner

$\dot{\imath}$ = 10100001	$\hat{A}$ = 11000010	B = 01000010	$\tilde{A}$ = 11000011
C = 01000011	$\hat{A}$ = 11000010	! = 00100001	0 = 00110000
" = 10101000	\ = 01011100	2 = 00110010	= 00011011
= 00001110	$\ddagger$ = 10000111	$\hat{A}$ = 11000010	= 00010000

Berdasarkan pada tabel di atas maka diambil seluruh nilai seperti berikut "10100001 11000010 01000010 11000011 01000011 11000010 00100001 00110000 10101000 01011100 00110010 00011011 00001110 10000111 11000010 00010000". Proses pengembalian bit akhir yaitu dengan cara menyatukan seluruh biner dan memisahkan nilai biner dengan biner *ternary* yaitu sebagai berikut "10 10000 11 10000 100 10000 101 10000 111 10000 1000 10000 1001 10000 1010 10000 1011 10000 1100 10000 1101 10000 1110 10000 1111 10000 10000 10000".

Berdasarkan hasil *string* bit di atas nilai biner dari "10000" dirubah dengan simbol *comma code* (c) dan masing-masing nilai biner dirubah menjadi nilai *ternary*. Maka perubahan itu akan dihasilkan nilai dari *ternary* "1c 2c 10c 11c 12c 20c 21c 22c 100c 101c 102c 110c 111c 112c 120c". Susun kembali nilai *ternary* dalam bentuk tabel.

Tabel 8. Nilai Ternary Comma Code

No	Heksadesimal	Ternary Comma Code	Biner
1	46	1c	10
2	49	2c	11
3	52	10c	100
4	68	11c	101
5	24	12c	110
6	00	20c	111
7	F9	21c	1000
8	41	22c	1001
9	57	100c	1010
10	45	101c	1011
11	56	102c	1100
12	6D	110c	1101
13	66	111c	1110
14	20	112c	1111
15	72	120c	10000

Kembalikan nilai bit tersebut kebentuk semula berdasarkan tabel angka dan tabel kode yang telah didapat dari hasil proses kompresi dan bentuk nilai semula, rubah menjadi "49, 52, 46, 46, 68, 24, 00, F9, 41, 57, 45, 56, 6D, 66, 20, 74".

4. KESIMPULAN

Sesuai dengan apa yang telah di bahas pada bab sebelumnya dan berdasarkan hasil analisa yang dilakukan, maka penulis dapat mengambil kesimpulan Kompresi *file* audio dilakukan dengan mengubah nilai kedalam bentuk heksadesimal agar dapat dilakukan perhitungan. Nilai bilangan heksa desimal yang didapat akan di proses menggunakan algoritma *Ternary Comma Code (TCC)* sehingga menghasilkan nilai bilangan heksadesimal baru yang akan merubah ukuran dari *file* audio tersebut semula besar menjadi kecil. Metode Ternary Comma Code(TCC) dapat diterapkan dalam mengkompresi *file* audio pada aplikasi perekam suara dimana *file* audio yang mempunyai ukuran besar dapat di kompres menjadi ukuran yang lebih kecil. Hasil dari aplikasi tersebut mampu mengkompresi *file* audio hingga dapat menghasilkan rasio rata-rata 41,40%.

REFERENCES

- [1] M. S. Nasution, "Analisis Perbandingan Algoritma Ternary Comma Code (TCC) dan Rice Code dalam Kompresi File Citra Digital (Bitmap)," Medan, 2018.
- [2] D. Husni, "Implementasi Dan Analisis Algoritma Kompresi Punctured Elias Codes dan Ternary Comma Code Pada File Doc," 2016.
- [3] D. Putra, *Pengolahan Citra Digital*. Yogyakarta: C.V Andi Offset, 2010
- [4] T. W. M. K. T. S. M.kom, E. M.kom, D.V.Suhartono, O. D. N. MT, *Pengolahan Citra Digital*. Yogyakarta: C.v Andi Offset, 2009.
- [5] Utari, "Implementasi Algoritma Run Lenght Encoding Untuk Perancangan Aplikasi Kompresi Dan Dekompresi File Citra," *J. Times*, vol. 2, p. 24, 2016.
- [6] E. F. Sigit Suryanto, *Pemograman Algoritma*. Yogyakarta: Cv Andi offset, 2012.
- [7] I. Binanto, *Multimedia Digital - Dasar Teori dan Pengembangannya*. Yogyakarta: Andi, 2010.
- [8] M. F. Heri Santoso, "Perancangan Aplikasi Keamanan File Audio Format WAV (WAVEFORM) Menggunakan Algoritma RSA," *J. ilmu Komput. dan Inform.*, vol. 2, p. 01, 2018.
- [9] R. : S. M. 2011 S, *Module Pembelajaran Rekayasa Perangkat Lunak*. 2010.
- [10] Hendrayudi, *Dasar Dasar Pemograman Microsoft Visual Basic 2008*. Bandung: PT. Sarana Tutorial Nurani Sejahtera, 2011.
- [11] Hendrayudi, *VB 2008 untuk Berbagai Keperluan Pemograman*. jakarta: PT .Elex Media Komputindo, 2009.
- [12] Ihsan and D. P. Utomo, "Analisis Perbandingan Algoritma Even-Rodeh Code Dan Algoritma Subexponential Code Untuk Kompresi File Teks," KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer), vol. 4, no. 1, 2020.
- [13] S. R. Saragih and D. P. Utomo, "Penarapan Algoritma Prefix Code Dalam Kompresi Data Teks," KOMIK (Konferensi Nas. Teknol. Inf. dan

- Komputer), vol. 4, no. 1, 2020.
- [14] Lamsah and D. P. Utomo, "Penerapan Algoritma Stout Codes Untuk Kompresi Record Pada Databade Di Aplikasi Kumpulan Novel," KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer), vol. 4, no. 1, 2020.