Pemanfaatan Kode Huffman Dalam Optimasi Ukuran Citra Pada Sistem Steganografi Menggunakan Metode Least Significant Bit

Aufa Fadhlurohman and 13518009[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13518009@std.stei.itb.ac.id

Abstrak--Perkembangan teknologi informasi dan komunikasi saat ini semakin pesat. Internet menjadi media yang semakin beragam penggunaannya. Lalu lintas internet yang semakin padat membuat tingkat keamanan privasi dalam komunikasi di internet mengalami degradasi. Untuk itu diperlukan sebuah metode untuk mengamankan data yang dikirimkan melalui suatu jaringan. Salah satu teknik yang tergolong aman dalam meningkatkan keamanan pesan rahasia adalah steganografi. Steganografi adalah ilmu dan seni menyembunyikan pesan rahasia (hiding message) sedemikian sehingga keberadaan (eksistensi) pesan yang tidak terdeteksi oleh indera manusia.Salah satu media menyimpan pesan. Salah satu media untuk menyimpan pesan dengan teknik ini adalah citra atau gambar. Gambar yang berisi pesan dapat dikompres sebagi upaya optimasi ukuran citra. Salah satu teknik kompresi sederhana adalah memanfaatkan kode Huffman.

Kata Kunci—Steganografi, Kode Huffman, Kompresi Citra

I. PENDAHULUAN

Perkembangan internet yang semakin pesat menciptakan kemudahan dalam mengakses jalur komunikasi dan informasi. Kemudahan yang tercipta ini tentunya memberikan banyak manfaat bagi manusia dan kehidupannya, terutama dalam membangun dunia. Namun di balik perkembangan ini, beberapa lembaga yang membutuhkan akses pengiriman dan penerimaan data rahasia membutuhkan metode yang lebih aman. Dibutuhkan keamanan data yang lebih ketat lagi untuk mencegah terjadinya kejahatan-kejahatan pencurian informasi yang dilakukan oleh pihak-pihak yang tidak bertanggung jawab.

Terdapat beberapa usaha untuk menangani masalah keamanan data rahasia yang dikirimkan melalui internet, diantaranya adalah menggunakan teknik kriptografi dan steganografi. Kata kriptografi berasal dari bahasa Yunani, “kryptós” yang berarti tersembunyi dan “gráphein” yang berarti tulisan. Dari arti perkata tersebut, kriptografi dapat diartikan sebagai “tulisan tersembunyi”. Menurut Request for Comments (RFC), kriptografi merupakan ilmu matematika yang berhubungan dengan transformasi data untuk membuat artinya tidak dapat dipahami (untuk menyembunyikan maknanya), mencegahnya dari perubahan tanpa izin, atau mencegahnya dari penggunaan yang tidak sah. Kriptografi menekankan pada konsep menyembunyikan makna, sehingga hanya pembuat informasi dan penerimanya yang dapat mengakses makna dari suatu informasi. Sedangkan kata steganografi berasal dari bahasa Yunani “steganos”, yang artinya tersembunyi atau terselubung, dan “graphein”, yang artinya menulis. Steganografi adalah ilmu, teknik atau seni menyembunyikan pesan rahasia “hiding message” atau tulisan rahasia “covered writing” sehingga keberadaan pesan tidak terdeteksi orang lain kecuali pengirim dan penerima pesan tersebut [1]. Berbeda dengan kriptografi yang membuat orang tidak dapat memahami makna akan suatu pesan, steganografi menekankan pada menyembunyikan keberadaan pesan sehigga orang-orang tidak menyadari akan pesan tersebut.

Steganografi menjadi suatu hal yang unik karena hampir setiap media digital dapat menjadi media yang digunakan dalam menyembunyikan pesan. Media-media tersebut seperti berkas citra digital, berkas audio, dan berkas video. Penyembunyian ini dasarnya tidak terlihat, sehingga pesan cenderung menjadi lebih aman dikirimkan. Salah satu media yang paling mudah dan sering digunakan dalam menyimpan pesan rahasia adalah citra digital. Metode yang paling sederhana dalam menyematkan pesan rahasia adalah dengan menggunakan metode LSB (Least Significant Bit Modification). Metode ini menyematkan pesan rahasia dengan mengubah bit terakhir dari suatu dataatas beberapa bit. Bit terakhir diambil atas pertimbangan signifikansi perubahan yang terjadi dalam data yang digunakan sebagai media penyimpanan pesan.

Pesan dalam steganografi disisipkan dalam bentuk biner. Untuk memaksimalkan penyimpanan, dapat diterapkan teknik kompresi. Kode Huffman yang memanfaatkan teori *tree* dapat digunakan unuk mempersingkat banyak bit atas pesan yang ingin disimpan dalam steganografi ini.

II. LANDASAN TEORI

2.1 Pengertian Pohon

Pohon adalah graf tak-berarah terhubung yang tidak mengandung sirkuit. Dalam teori graf, konsep pohon dapat dianggap konsep yang paling penting, karena memiliki terapan yang luas dalam berbagai bidang ilmu pengetahuan. [2]

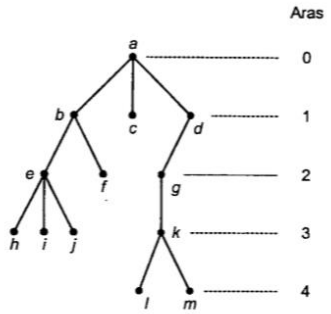
Pada dasarnya pohon merupakan salah satu bentuk graf yang memiliki aturan atau batasan tertentu. Misalkan $G = (V, E)$

adalah sebuah graf sederhana dan memiliki jumlah simpul n. Maka, semua pernyataan do bawah ini adalah ekivalen :

1. G adalah pohon.
2. Setiap pasang simpul di dalam G terhubung dengan lintasan tunggal.
3. G terhubung dan memiliki $m = n - 1$ buah sisi.
4. G tidak mengandung sirkuit dan $m = n - 1$ buah sisi.
5. G tidak mengandung sirkuit dan penambahan satu sisi pada graf akan membuat satu sirkuit.
6. G terhubung dan semua sisinya adalah jembatan (jembatan adalah sisi yang bila dihapus menyebabkan graf terpecah menjadi dua komponen).

Dalam teori pohon, seringkali digunakan terminologi tertentu. Pohon yang sebuah simpulnya diperlakukan sebagai akar dan sisi-sisinya diberi arah menjauh dari akar dinamakan pohon berakar (*rooted tree*). Selain itu terdapat beberapa terminologi atau istilah umum dari teori pohon ini, di antaranya:

1. Anak (*child* atau *children*) dan orang tua (*Parent*)
   Misalkan Xx adalah sebuah simpul di dalam pohon berakar. Simpul y dikatakan anak dari simpul x jika ada sisi dari simpul x ke y. Dengan demikian, x disebut orangtua dari y.
2. Lintasan (*path*)
   Lintasan dari simpul v_i ke simpul v_k adalah runtunan simpul-simpul v_i, v_{i+1}, ... , v_k sedemikian sehingga v_i adalah orang tua dari v_{i+1} untuk $1 \leq i \leq k$.
3. Keturunan (*descendant*) dan leluhur (*ancestor*)
   Simpul y disebut keturunan dari simpul x dan x adalah leluhur dari simpul y jika terdapat lintasan dari x ke y di dalam pohon.
4. Saudara kandung (*sibling*)
   Simpul x dan simpul y dikatakan bersaudara kandung satu sama lain jika orang tua dari simpul x sama dengan orang tua dari simpul y.
5. Upapohon (*subtree*)
   Misalkan x adalah simpul dalam pohon T, maka upapohon dengan x sebagai akar merupakan upagraf $T' = (V', E')$ sedemikian sehingga V' mengandung x dan semua keturunan x dan E' mengandung sisi-sisi dalam semua lintasan yang berasal dari x.
6. Derajat (*degree*)
   Derajat dari simpul x merupakan banyaknya anak dari simpul x itu sendiri. Meskipun pohon merupakan sebuah graf dengan ketentuan tertentu, definisi derajat dalam teori pohon dan graf tidak sama.
7. Daun (*leaf*)
   Simpul x disebut daun jika derajat dari simpul x adalah 0, artinya simpul x tidak memiliki anak pohon.
8. Simpul dalam (*internal node*)
   Simpul dalam merupakan simpul yang memiliki anak atau derajat simpul lebih besar dari nol.
9. Aras (*level*) atau tingkat
   Aras simpul x didefinisikan sebagai tingkat keturunan simpul x jika diukur dari orangtua yang teratas. Akar memiliki aras = 0. Ilustrasi aras atau tingkat dapat dilihar dari ilustrasi berikut:

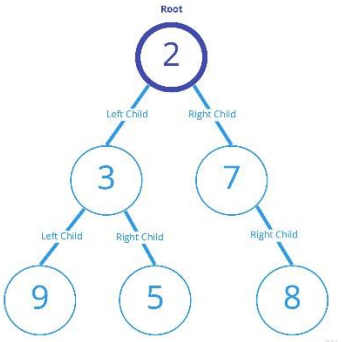


Gambar 2.1.1 Ilustrasi Aras Pohon

Sumber : Matematika Diskrit Jilid 3 2010 (Dr. Rinaldi Munir)

10. Tinggi (*Height*) atau kedalaman
    Tinggi pohon T merupakan nilai maksimum yang mungkin dari aras x untuk semua x yang merupakan simpul dari pohon T. [2]

Salah satu jenis pohon yang sering digunakan dalam komputasi adalah pohon biner. Pohon biner adalah pohon yang hanya memiliki maksimal dua anak untuk setiap orangtuanya. Umumnya digunakan istilah anak kiri dan anak kanan untuk kedua anak pada pohon biner. Jika ada penambahan dilakukan penelusuran ke kiri dan ke kanan yang ditetapkan sebagai subpohon. Gambar di bawah ini adalah contoh darin pohon biner :



Gambar 2.1.2 Pohon Biner

Sumber : https://medium.com/better-programming/introduction-to-binary-trees-deaabd5832bd

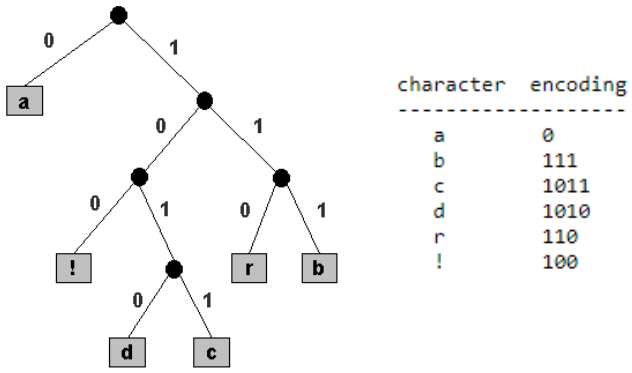
2.2 Kode Awalan

Kode awalan (*prefix code*) adalah himpunan kode, misalnya kode biner, sedemikian sehingga tidak ada anggota kumpulan yang merupakan awalan dari anggota yang lain [2]. Kode awalan biasanya direpresentasikan dengan pohon biner. Berikut adalah contoh kombinasi yang tergolong kode awalan :

$$KA = \{ 000, 001, 01, 10, 11 \}$$

Pada himpunan KA tidak ada anggota yang merupakan subbagian terdepan dari anggota lainnya. Misalkan dalam suatu himpunan kode terdapat anggota 00 dan juga terdapat 0001, maka himpunan tersebut bukan kode awalan. Dalam membangun pohon biner awalan, terdapat ketentuan-ketentuan tertentu. Sisi dari pohon dilabeli dengan angka 0 atau 1 saja. Pelabelan ini juga harus konsisten, jika sisi kiri diberi 0 maka setiap sisi kiri pun harus diberikan dengan label 0, dan jika sisi kanan dilabeli 1 maka demikian sisi kanan lainnya. Barisan sisi-

sisi yang dilewati dari akar menuju daun tertentu adalah representasi dari info dari daun tersebut. Berikut adalah contoh dari pohon yang membangun kode awalan beserta dengan kode awalan yang terbentuk.



Gambar 2.2.1 Pohon Kode Awalan

Sumber : https://www.cs.princeton.edu/courses/archive/spr01/cs126/assignments/prefix.html

Dari pohon di atas, dapat terlihat bahwa setiap karakter memiliki kode dengan rangkaian bit yang berbeda. Contohnya, pada pohon tersebut karakter 'a' direpresentasikan dalam sebuah bit, sedangkan karakter 'd' dengan empat bit. Sifat atau batasan ini merupakan skema untuk mempersingkat banyak bit dalam suatu pesan, dengan menyatakan karakter yang sering muncul dalam jumlah bit yang paling sedikit dan berlaku pula sebaliknya. Ini konsep dasar dari kode Huffman.

Kode awalan dibentuk dengan melakukan konkatenasi dari bit-bit setiap karakter. Misalnya kata "car" akan dibentuk :

c | a | r atau 1011 | 0 | 110 atau 10110110

Kode yang terbentuk dapat dibaca karena tidak ada kode yang merupakan prefix dari kode karakter lain.

1. Mulai dari akar pohon.
2. Lakukan aksi secara berulang hingga tercapai salah satu daun dari pohon:
    a. Terima satu bit dari string
    b. Cari pada anak kiri, jika bit bernilai 0. Dan jika bit bernilai 1, lakukan pencarian pada anak kanan.
3. Tulis/simpan karakter yang ada pada daun tersebut.
4. Ulangi algoritma hingga akhir string. [3]

2.3 Kode Huffman

Dalam komunikasi data, pesan dikirim dengan ukuran tertentu. Untuk mengirim data dalam jumlah banyak, diperlukan ukuran yang besar dan berimplikasi pada lamanya waktu pengiriman pesan tersebut. Pada penyimpanan juga berlaku konsep yang sama. Untuk itu salah satu yang dapat dilakukan untuk meningkatkan efisiensi data adalah dengan melakukan pemampatan. Data digital pada dasarnya terbentuk dari susunan bit. Perlu dilakukan pengkodean agar suatu susunan bit dari sebuah data menjadi lebih pendek. Salah satu metode yang dapat memperkecil ukuran data adalah metode kode Huffman. Kode Huffman adalah salah satu contoh kode awalan yang popular dan tergolong sederhana digunakan dalam teknik kompresi data. Metode kompresi data dengan kode Huffman ini adalah dengan mengonversikan simbol-simbol dari data yang ingin dikompresi menjadi rangkaian kode bit yang lebih pendek dengan memperhatikan jumlah kemunculan setiap simbol pada data. Proses konversinya menggunakan konsep pohon biner seperti pada kode awalan yang telah dijelaskan sebelumnya.

Salah satu bentuk representasi data yang paling sering adalah dalam bentuk kode ASCII. ASCII terbentuk dari 8 buah bit. Daftar dari asci telah ditetapkan melalui konvensi. Dengan ketentuan tersebut, misalkan terdapat sebuah data yang berisikan 11 karakter maka jika direpresentasikan dalam biner, data tersebut memiliki panjang $11\ x\ 8 = 88\ bit$. Untuk meminimumkan jumlah bit yang dibutuhkan, panjang kode untuk setiap karakter sedapat mungkin diperpendek, terutama untuk karakter dengan frekuensi kemunculan yang tinggi. Konsep ini yang menjadi dasar dari kode Huffman. Sebagai contoh jika terdapat string 'aaaaaaaaaabbc' yang terdiri dari 10 karakter 'a', 2 karakter 'b', dan 1 karakter 'c' dalam kode biasa akan memerlukan $13\ x\ 8 = 104\ bit$ untuk merepresentasikannya. Namun jika 'a' yang memiliki frekuensi tertinggi direpresentasikan dengan 1, 'b' dengan 10, dan 'c' dengan 11, maka diperlukan sebanyak $10 + 4 + 2 = 16\ bit$. Perbedaan ukuran sangat terlihat di sini karena frekuensi terbanyak memiliki jumlah bit paling maksimum.

Untuk membangun kode Huffman, terdapat beberapa langkah yang harus dilakukan, yaitu :

1. Buat tabel yang berisikan semua symbol yang muncul dalam suatu pesan, hitung jumlah kemunculannya, dan hitung peluang kemunculannya. Peluang kemunculan dapat dihitung melalui pembagian jumlah kemunculan symbol x pada string dengan panjang string tersebut.
2. Pilih dua symbol dengan peluang terkecil dari tabel yang telah dibangun. Kombinasikan kedua symbol sebagai simpul orangtua yang bernama konkatenasi antara dua simbol tersebut dan peluangnya merupakan penjumlahan dari kedua anaknya. Simbol baru ini diperlakukan sebagai simpul baru dan diperhitungkan dalam mencari simbol selanjutnya yang memiliki peluang paling kecil.
3. Selanjutnya, pilih dua simbol berikutnya, termasuk simbol baru yang mempunyai peluang terkecil. Setelah itu, lakukan hal yang sama dengan setelah kombinasi simbol pada langkah kedua sehingga terbentuk simbol baru.
4. Proses dilakukan hingga menghasilkan simpul yang telah mencakupi setiap simbol yang ada pada string atau jumlah peluangnya menjadi 1. [2]

Misalkan kita ingin membangun kode Huffman dari string "ABABABACDC".

1. Buat tabel frekuensi

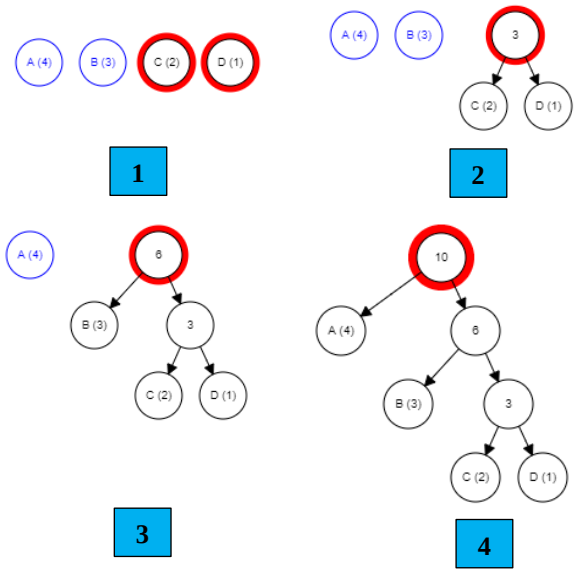
Tabel 2.1 Contoh Kode Huffman

Simbol	Kekerapan	Peluang
A	4	4/10
B	3	3/10
C	2	2/10
D	1	1/10

2. Selanjutnya, lakukan sesuai algoritma yang telah dijelaskan. Pada kasus ini, dua simbol dengan peluang terkecil adalah simbol C dan D. Terbentuk orangtua CD dengan peluang $\frac{2}{10} + \frac{1}{10} = \frac{3}{10}$. Simbol CD dengan

simbol A memiliki peluang terkecil, kombinasikan lagi keduanya sehingga terbentuk simbol ACD.

Berikut adalah ilustrasi dalam membangun pohon kode Huffman string di atas :



Gambar 2.3.1 Alur Kode Huffman

Sumber : https://people.ok.ubc.ca/ylucet/DS/Huffman.html

Dari proses tersebut didapatkan kode untuk tiap simbol yaitu 'A' adalah 0, 'B' adalah 10, 'C' adalah 110, dan 'D' adalah 111. Terlihat bahwa simbol yang kekerapannya lebih tinggi memiliki panjang biner yang lebih kecil.

2.4 Steganografi Pada Citra Digital

Steganografi adalah seni dan ilmu menulis informasi tersembunyi sedemikian rupa sehingga tak seorang pun, selain pengirim dan penerima yang dituju, mengetahui keberadaan informasi tersebut [4]. Pada umumnya informasi dititipkan pada media digital seperti foto, artikel, daftar belanja, atau lainnya. Secara garis besar metide steganografi terbagi menjadi dua bagian utama, yaitu proses penyembunyian data (*encode*) dan proses pengembalian data ke bentuk semula (*decode*). Hasil dari *encoding* atau *embedding* disebut *Stego Object* [2].

Salah satu metode steganografi sederhana adalah Metode LSB atau *Least Significant Bit*. Kemudahan dalam proses menginput data maupun mengambil data membuat metode ini menjadi salah satu yang sering digunakan. Metode ini bekerja dengan cara mengganti bit terakhir dari representasi susunan bit objek pembangun media. Misalnya jika digunakan media citra digital berukuran $80\ x\ 120$ piksel, artinya terdapat 9600 piksel yang terdapat pada gambar tersebut. Tiap piksel terdiri dari kombinasi tiga warna yaitu *red*, *green*, dan *blue* (RGB), yang tiap warnanya dibangun atas 8 bit. Dari tiap 8 bit tersebut, bit terakhir akan diganti untuk menyinpan pesan yang ingin kita kirimkan. Tiap piksel dapat menampung 3 bit pesan rahasia. Penggunaan LSB bertujuan untuk sesuai namanya untuk mengurangi perbedaan yang timbul. Pada sebuah gambar, apabila bit terakhir warna piksel diganti rasanya tidak membuat perbedaan terasa jelas pada mata. Untuk lebih jelasnya misal terdapat sebuah gambar yang memiliki kode biner 3 piksel pertama sesuai data berikut ini :

Tabel 2.2 Tabel Biner

R	G	B
11110010	10101000	00001101
10110110	11111111	00000101
01000101	01011000	10101011
LSB : 001 011 101		

Jika ingin disisipkan pesan yang memiliki biner 111 111 111 pada ketiga piksel tersebut maka tiap bit terakhir dari sel harus disesuaikan menjadi seperti berikut :

Tabel 2.3 Tabel Biner Hasil

R	G	B
11110011	10101001	00001101
10110111	11111111	00000101
01000101	01011001	10101011
LSB : 111 111 111		

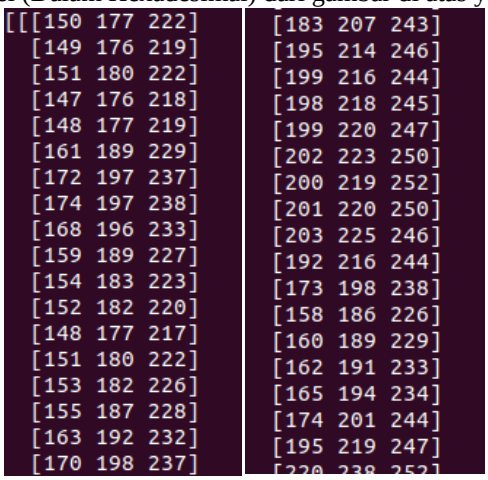
III. ANALISIS PERMASALAHAN

Pemanfaatan media gambar sebagai tempat menyimpan pesan cenderung tidak membuat kecurigaan sehingga efektif terhadap tujuan pengiriman pesan. Perubahan yang dilakukan tidak terlihat secara laingsung. Sebagai contoh berikut terdapat sebuah gambar berukuran $120\ \times 80\ piksel$. (Format .PNG)

Gambar 3.1 Citra Digital Kampus ITB

Sumber : https://cdn.sindonews.net/dyn/620/content/2019/08/19/144/1431208/itb-terbaik-nonvokasi-pens-peringkat-teratas-untuk-vokasi-gOA.jpg

Dari gambar tersebut dapat dilihat bentuk binernya menggunakan berbagai *tools*, salah satunya menggunakan *library* PIL, opencv, dan base64 pada bahasa pemrograman python 3. Setelah dikompilasi, didapatkan list kode RGB dari tiap piksel (Dalam Hexadesimal) dari gambar di atas yaitu :



Gambar 3.1 Cuplikan RGB Citra Digital Kampus ITB

```
import sys
import numpy as np
from PIL import Image

np.set_printoptions(threshold=sys.maxsize)
img = Image.open('Pixel.jpg')
array = np.array(img)
print(array)
```

Gambar 3.2 Cuplikan Program Binary Picture RGB

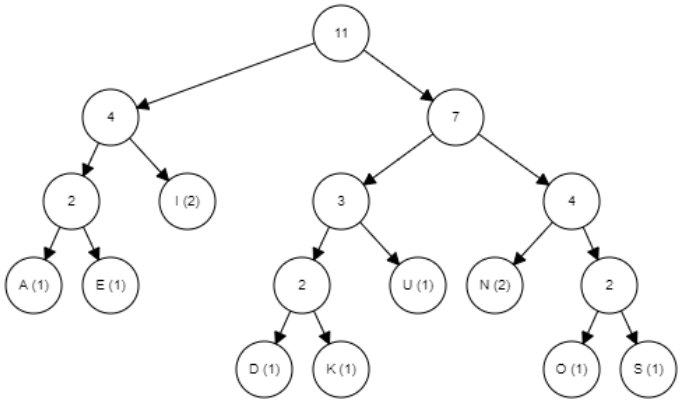
Untuk menyimpan sebuah pesan pada gambar tersebut, terlebih dahulu lakukan kompresi biner sesuai yang telah jelaskan sebelumnya yaitu menggunakan prinsip kode Huffman. Misalkan kata yang ingin disisipkan adalah "INDONESIAKU".

1. Tabel frekuensi
   Berdasarkan string tersebut, dapat dibuat tabel frekuensi sebagi berikut:

Tabel 3.1 Tabel Simbol Kode Huffman "INDONESIAKU"

Simbol	Kekerapan	Peluang
I	2	2/11
N	2	2/11
D	1	1/11
O	1	1/11
E	1	1/11
S	1	1/11
A	1	1/11
K	1	1/11
U	1	1/11

2. Pohon *Huffman Coding*
   Dengan melakukan proses dengan langkah-langkah seperti yang dijelaskan pada teori dasar, dapat dibangun pohon Huffman sebagai berikut atas string "INDONESIAKU" :



Gambar 2.3.1 Kode Huffman "INDONESIAKU"

Sumber : https://people.ok.ubc.ca/ylucet/DS/Huffman.html

3. Tabel Kode Huffman

Tabel 3.2 Tabel Hasil Kode Huffman "INDONESIAKU"

Simbol	Kode Huffman
A	000
E	001
I	01
D	1000
K	1001
U	101
N	110
O	1110
S	1111

Dari tabel tersebut, terlihat string "INDONESIAKU" dapat direpresentasikan dengan kode biner berikut:

I	N	D	O	N	E	S	I	A	K	U
01	110	1000	1110	110	001	1111	01	000	1001	101

Kode 01110100011101100011111010001001101 memiliki panjang 35 bit, padahal jika menggunakan konsep ASCII pada umumnya, untuk merepresentasikan kata INDONESIAKU dibutuhkan $8\ x\ 11 = 88\ bit$. Perbandingan ukuran data setelah dan sebelum kompresi dapat dilihat dari *compression ratio (CR)* dari hasil pengkodean yaitu:

$$CR = \frac{panjang\ bit\ awal}{panjang\ bit\ kompresi} = \frac{88}{35} = 2{,}514$$

Perbandingan yang cukup signifikan padalah dari sisi frekuensi kemunculan berulang tidak terlalu banyak.

Setelah didapat kode Huffman dari string tersebut, ganti setiap LSB pada gambar 3.1. Karena ukuran bit yang akan diinjeksi sebanyak 36 maka hanya diperlukan $\frac{35}{3}\ atau\ 12\ piksel$ dari gambar 3.1. Berikut adalah data pengubahan bit pada gambar :

Tabel 3.3 Perubahan LSB piksel pada gambar

Sebelum	Sesudah
10010110	10010110
10110001	10110001
11011110	11011111
10010101	10010101
10110000	10110000
11011011	11011011
10010111	10010110
10110100	10110100
11011110	11011110
10010011	10010011
10110000	10110001
11011010	11011011
10010100	10010100
10110001	10110001
11011011	11011011
10100001	10100000
10111101	10111100
11100101	11100100
10101100	10101101
11000101	11000101
11101101	11101101
10101110	10101111
11000101	11000101
11101110	11101110
10101000	10101001
11000100	11000100
11101001	11101000
10011111	10011110
10111101	10111101
11100011	11100010
10011010	10011010
10110110	10110111
11011100	11011101
10010100	10010100
10110001	10110001
11011001	11011001

Setelah bit gambar 3.1 disisipi dengan pesan, tidak sama sekali terlihat adanya perubahan pada gambar, berikut gambar yang telah disisipi pesan:

Gambar 3.3 Citra Digital Kampus ITB yang telah disisipi pesan

Sumber : https://cdn.sindonews.net/dyn/620/content/2019/08/19/144/1431208/itb-terbaik-nonvokasi-pens-peringkat-teratas-untuk-vokasi-gOA.jpg

Jelas perubahan tidak terlihat karena yang diubah hanya 12 piksel pertama di baris pertama. Namun lebih banyak piksel dibutuhkan jika kode tidak dikompres terlebih dahulu.

Ukuran berkas gambar utamanya ditentukan oleh banyak pikselnya, jika ingin mengirim pesan rahasia yang disisipkan ke gambar dalam ukuran banyak, kompresi adalah hal yang sangat penting, karena berkorelasi dengan ukuran *file* yang dikirimkan. Dan ukuran file akan berpengaruh pada waktu proses pengiriman.

Steganografi adalah metode yang penting bagi jalur komunikasi penting dan rahasia. Di masa yang memiliki perkembangan jaringan global yang luar biasa ini, keamanan menjadi hal yang patut dikembangkan. Keterbukaan informasi pada beberapa pihak adalah hal yang membahayakan, dan seni atau ilmu menyembunyikan pesan ini berguna bagi pihak-pihak tertentu tersebut. Ukuran berkas dapat dioptimasi melalui kompresi dengan metode-metode yang dapat dilakukan, salah satu metode kompresi sederhana adalah Kode Huffman.

IV. KESIMPULAN

Prinsip Kode Huffman sangat bermanfaat dalam menekan ukuran atas suatu berkas digital. Salah satu pemanfaatannya adalah untuk mempersingkat kode biner atas suatu pesan. Salah satu pemanfaatannya terdapat pada kompresi pesan dalam teknik steganografi. Dengan memanfaatkan prinsip ini maka jumlah pesan yang dapat ditampung dari suatu gambar akan lebih banyak. Prinsip ini meningkatkan efisiensi pengiriman, karena untuk menyimpan pesan yang tinggi secara kuantitas dalam suatu citra digital, piksel yang dibutuhkan semakin sedikit dan berimplikasi pada lebih kecilnya ukuran gambar yang dibutuhkan.

Proses dari konsep kode Huffman, yaitu menghitung frekuensi kemunculan setiap karakter yang terdapat dalam sebuah file. Kemudian karakter tersebut disusun dalam sebuah tabel yang memuat peluang kemunculan simbol atau karakter tersebut, lalu membuat pohon biner berdasarkan urutan karakter dari yang peluangnya terkecil ke yang terbesar. Dan selanjutnya kode dapat dilihat dari pohon untuk setiap karakternya. Kode dari suatu string dapat didapat dengan melakukan konkatenasi biner hasil metode Huffman sesuai dengan karakter yang ada.

V. UCAPAN TERIMA KASIH

Puji syukur kehadirat Allah SWT, atas limpahan Rahmat dan Karunia-Nya, sehingga penulis dapat menyelesaikan pembuatan makalah ini. Terima kasih penulis ucapkan kepada Bu Fariska Zakhralativa, S.T, M.T sebagai dosen pengampu dalam mata kuliah IF 2120 Matematika Diskrit ini. Terima kasih juga penulis ucapkan kepada kedua Orang Tua penulis yang selalu memberikan dukungan moral dalam setiap aktivitas penulis.

REFERENSI

[1] Suhendra, Adhe. Steganografi Pada Citra Terkompresi Metode Huffman. Jurnal Media Informasi Analisa dan Sistem, Volume 1, 2016. Diambil dari *https://media.neliti.com/media/publications/282448*

[2] Munir, Rinaldi. Matematika Diskrit, Bandung: Informatika, 2010, edisi ketiga.

[3] http://www.cs.princeton.edu/courses/archive/spr01/cs126/assignments/prefix.html, diakses pada 4 Desember 2019

[4] Yaddarabullah. Pemanfaatan Kompresi Huffman Untuk Optimasi. Jurnal TELEMATIKA MKOM, Volume 7, 2015.. Diambil dari *https://www.researchgate.net/* publication/325859618

[5] Steven Pigeon, Yoshua Bengio — Memory-Efficient Adaptive Huffman Coding — Doctor Dobb's Journal, n° 290, 1998, p. 131–135

[6] Steganography Tutorial: Least Significant Bit (LSB). *BoiteAKlou*, 12 Aug. 2018, *https://www.boiteaklou.fr/Steganography-Least-SignificantBit.html*

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 5 Desember 2019

Aufa Fadhlurohman
13518009