Journal of Applied Informatics and Computing (JAIC)
Vol.9, No.3, June 2025, pp. 868~875
e-ISSN: 2548-6861

Hamming Code in JPEG Image Steganography within the Discrete Cosine Transform Domain

Dian Hafidh Zulfikar [1]*, Hermanto [2]*
* Sistem Informasi, Fakultas Sains dan Teknologi UIN Raden Intan Lampung
dianhafidhzulfikar_uin@radenintan.ac.id [1], hermanto@radenintan.ac.id [2]

Article Info

Article history:

Received 2025-04-10
Revised 2025-05-01
Accepted 2025-05-06

Keyword:

Dicrete Cosine Transform,
Hamming Code,
JPEG,
Peak Signal to Noise Ratio (PSNR),
Structural Similarity Index (SSIM).

ABSTRACT

This study proposes a novel JPEG image steganography method that combines Least Significant Bit (LSB) embedding in the Discrete Cosine Transform (DCT) domain with *Hamming Code* ($2^k - 1, 2^k - k - 1$) to minimize the number of modified DCT coefficients. Experiments were conducted on images with varying resolutions (512×512, 1024×1024, 2048×2048) and JPEG quality factor of 75, where PSNR (*Peak Signal to Noise Ratio*) and SSIM (*Structural Similarity Index*) parameters are used to measure the quality and similarity between the original image and the stego image. The method achieved an embedding capacity of up to 524,288 bits, with an average PSNR of 39–41 dB and SSIM above 0.98. Compared to conventional techniques such as JSteg and F5, the proposed approach demonstrates improved embedding capacity, better visual quality, and higher resistance to statistical steganalysis, making it suitable for secure and efficient data hiding applications.

I. PENDAHULUAN

Steganografi adalah metode untuk mentransfer informasi rahasia melalui saluran komunikasi publik sehingga pihak ketiga (penyerang) tidak dapat mendeteksi adanya pengiriman informasi tersembunyi tersebut[1][2]. Tidak seperti kriptografi yang mengubah data menjadi format yang tidak dapat dibaca tanpa kunci dekripsi, steganografi berfokus pada penyembunyian keberadaan pesan itu sendiri. Dengan demikian, pihak ketiga tidak menyadari bahwa ada informasi tersembunyi dalam suatu media. Efektivitas deteksi informasi rahasia dalam sistem steganografi berkaitan erat dengan ukuran data rahasia yang disisipkan[3][4].

Sistem steganografi memanfaatkan media dengan ukuran tetap sebagai alat transportasi[5]. Oleh karena itu, salah satu tantangan utama adalah menentukan kapasitas maksimum informasi rahasia yang dapat disisipkan ke dalam media tersebut. Penentuan kapasitas ini bukanlah hal yang sederhana[6], bahkan untuk metode-metode paling dasar dalam steganografi berbasis gambar. Penelitian sebelumnya, seperti yang dilakukan oleh Chandramouli[7][8] menyajikan analisis teoretis untuk menentukan kapasitas informasi rahasia yang dapat disisipkan ke dalam gambar menggunakan algoritma LSB (Least Significant Bit) pada domain spasial. Dalam beberapa dekade terakhir, steganografi berbasis citra digital telah menjadi bidang penelitian yang berkembang pesat. Salah satu format gambar yang paling sering digunakan dalam steganografi adalah JPEG (Joint Photographic Experts Group). Metode penyisipan LSB menyembunyikan informasi rahasia dalam gambar JPEG tetapi membutuhkan perubahan banyak bagian gambar, yang dapat menjadi kerugian. Teknik ini, meskipun efektif, juga menghadapi tantangan dalam hal deteksi oleh perangkat lunak analisis gambar yang semakin canggih. Menurut Chandramouli Metode LSB bekerja dengan mengubah bit data gambar yang paling tidak signifikan. Secara sederhana, itu berarti mengubah bagian terkecil dari gambar sehingga perubahan tidak terlihat.

Jessica Fridrich[9] kemudian mengembangkan estimasi yang lebih akurat dengan menggunakan analisis statistik dua tahap. Dalam penelitiannya dia melihat berbagai cara untuk menganalisis gambar untuk melihat seberapa baik informasi tersembunyi dapat dideteksi atau diekstraksi.

Pilihan media penampung steganografi sangat penting[10], karena jenis media tersebut memengaruhi desain dan keamanan sistem secara signifikan[11]. Misalnya, format gambar BMP, yang bersifat lossless yang berarti bahwa ketika kita menyimpan gambar, tidak ada data yang hilang. Itu menjaga semua detail gambar tetap utuh, sering dianggap

ideal untuk implementasi steganografi berbasis gambar karena redundansi datanya[12]. Namun, redundansi tersebut juga menjadi kelemahan, karena format ini mudah mencurigakan dalam komunikasi publik[3].

Algoritma LSB juga dapat diterapkan pada media dengan kompresi data yang bersifat lossy[1], seperti format JPEG. Untuk meningkatkan efisiensi penyisipan data dalam gambar JPEG, berbagai metode telah dikembangkan. Salah satu metode yang banyak digunakan adalah penyisipan pada domain frekuensi menggunakan Discrete Cosine Transform (DCT) Beberapa sistem steganografi populer, seperti JSteg, Outguess, dan JPHide & JPSeek, memanfaatkan kompresi JPEG untuk menyisipkan informasi rahasia pada LSB dari koefisien DCT (Discrete Cosine Transform). Namun, kelemahan utama metode ini adalah kebutuhan untuk mengubah sebanyak mungkin koefisien DCT sesuai dengan jumlah bit informasi rahasia yang disisipkan[13]. Dalam metode ini, informasi rahasia disisipkan ke dalam koefisien DCT yang diperoleh setelah transformasi gambar dari domain spasial ke domain frekuensi. Meskipun metode ini dapat meningkatkan kapasitas penyisipan, perubahan yang terlalu signifikan pada koefisien DCT dapat meningkatkan kemungkinan terdeteksi oleh algoritma analisis statistik.

Beberapa teknik steganografi yang telah dikembangkan, seperti JSteg, OutGuess, dan F5, memiliki kelemahan dalam hal kapasitas penyisipan dan resistensi terhadap deteksi oleh alat steganalisis. Oleh karena itu, diperlukan teknik baru yang tidak hanya dapat meningkatkan kapasitas penyisipan tetapi juga mempertahankan kualitas gambar agar tidak mudah terdeteksi.

Metode Least Significant Bit (LSB) adalah salah satu teknik yang paling sederhana dalam steganografi. Metode ini bekerja dengan mengganti bit-bit paling tidak signifikan dalam piksel gambar dengan bit-bit pesan rahasia. Namun, penerapan metode LSB pada citra JPEG cukup sulit karena kompresi lossy dapat mengubah bit-bit tersebut dan menyebabkan distorsi yang tidak dapat diprediksi.[11]

Penelitian ini bertujuan untuk mengembangkan dan mengevaluasi metode steganografi JPEG yang mengintegrasikan LSB di domain DCT dengan Hamming Code $(2k-1, 2k-k-1)$ untuk (i) memaksimalkan kapasitas penyisipan, (ii) meminimalkan modifikasi koefisien, dan (iii) mempertahankan kualitas visual serta resistensi terhadap deteksi statistik.

Penelitian ini adalah usulan baru yang mengkombinasikan (a) embedding LSB pada koefisien DCT terkuantisasi, (b) pengkodean Hamming untuk efisiensi, dan (c) kompresi ZIP sebelum embedding. Ini berbeda dari JSteg/F5 karena penggunaan Hamming Code untuk meminimalkan perubahan per blok.

II. METODE

A. *Jumlah dan Ukuran dataset Citra*

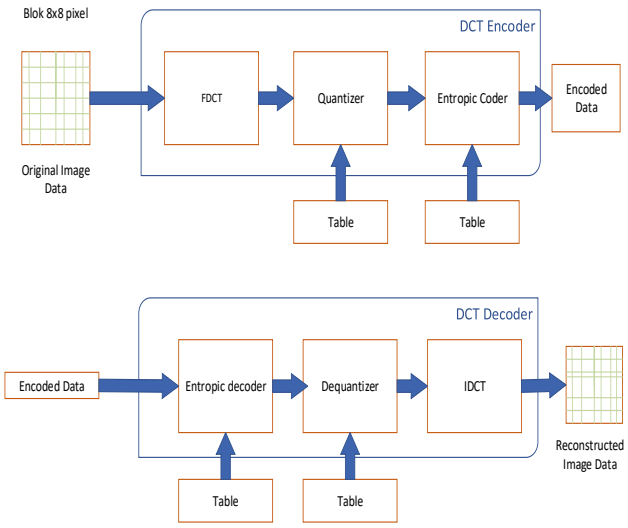
Eksperimen dalam penelitian ini dilakukan pada total 30 citra cover yang terdiri dari 10 citra beresolusi 512×512 piksel, 10 citra beresolusi 1024×1024 piksel, dan 10 citra beresolusi 2048×2048 piksel. Seluruh citra tersebut dikonversi ke format JPEG dengan kualitas yang dikendalikan secara terukur. Hasil eksperimen berupa kapasitas penyisipan dan metrik evaluasi untuk masing-masing resolusi disajikan pada Tabel 4.

Jumlah bit yang disisipkan pada setiap resolusi adalah sebesar 32.768 bit untuk citra 512×512 piksel, 131.072 bit untuk citra 1024×1024 piksel, dan 524.288 bit untuk citra 2048×2048 piksel. Penyebaran bit pesan serta bit parity Hamming didistribusikan secara merata ke seluruh koefisien DCT yang dipilih dengan menggunakan pseudorandom number generator (PRNG) yang diinisialisasi oleh kunci rahasia. Penelitian ini menggunakan tabel kuantisasi standar JPEG (ISO/IEC 10918-1) pada komponen luminance dengan quality factor sebesar 75. Tidak dilakukan modifikasi khusus terhadap matriks kuantisasi, sehingga kompatibilitas dengan pembaca JPEG umum tetap terjamin.

B. *File Gambar JPEG*

Format JPEG menggunakan kompresi data bersifat lossy, di mana informasi grafis yang dianggap tidak diperlukan akan dibuang tanpa memberikan dampak signifikan terhadap kualitas visual citra. Dengan pendekatan ini, rasio kompresi yang jauh lebih tinggi dapat dicapai dibandingkan dengan metode kompresi lossless. Standar JPEG mencakup serangkaian operasi yang harus dilakukan pada data visual [14], termasuk subsampling warna, Discrete Cosine Transform (DCT), kuantisasi koefisien DCT, dan pengkodean panjang kata variabel (Variable Length Code atau VLC).

Subsampling warna dilakukan dengan mengurangi resolusi data warna (chrominance) tanpa mengurangi resolusi data kecerahan (luminance), sehingga ruang penyimpanan dapat dihemat dengan membuang sebagian informasi warna yang tidak terlalu penting bagi persepsi visual manusia. Transformasi DCT digunakan untuk mengubah blok piksel gambar ke dalam domain frekuensi.



Gambar 1. Diagram Blok proses JPEG Encoder dan Decoder

Selanjutnya, koefisien DCT dibagi dengan nilai pada tabel kuantisasi, yang menghasilkan proses kompresi bersifat lossy. Terakhir, data dikompresi menggunakan teknik pengkodean entropi melalui VLC. JPEG mendukung berbagai mode pemrosesan, termasuk sequential, progressive, hierarchical, dan lossless. Dari keempat mode tersebut, mode sequential merupakan yang paling umum digunakan serta didukung oleh sebagian besar aplikasi dan pustaka JPEG [15]. Mode ini mencakup penggunaan sampel warna 8-bit, transformasi DCT, kuantisasi, dan pengkodean entropi.

C. Proses Encoding JPEG

Proses encoding JPEG dimulai dengan representasi data gambar dalam bentuk piksel, di mana setiap piksel memiliki tiga komponen warna sesuai model RGB (Red, Green, Blue), yaitu merah, hijau, dan biru. Gambar dalam format RGB kemudian dikonversi ke format YCbCr, yang terdiri dari tiga komponen: Y sebagai komponen kecerahan (luminance), Cb sebagai perbedaan antara biru dan kecerahan, serta Cr sebagai perbedaan antara merah dan kecerahan. Setelah konversi ke YCbCr, dilakukan subsampling warna pada komponen Cb dan Cr, sementara komponen Y tetap dipertahankan sepenuhnya untuk menjaga detail kecerahan citra.

Subsampling warna ini dilakukan dengan beberapa jenis rasio [16], antara lain 4:4:4 tanpa subsampling (setiap piksel memiliki informasi lengkap Y, Cb, dan Cr sehingga memberikan kualitas terbaik namun ukuran file besar), 4:2:2 dengan subsampling horizontal (setiap dua piksel berbagi satu nilai Cb dan Cr sehingga mengurangi ukuran hingga 33% tanpa banyak kehilangan kualitas), 4:2:0 dengan subsampling horizontal dan vertikal (setiap blok 2×2 piksel berbagi satu nilai Cb dan Cr sehingga mengurangi ukuran hingga 50% dengan tetap menjaga detail kecerahan), serta 4:1:1 sebagai subsampling ekstrem (setiap empat piksel berbagi satu nilai Cb dan Cr yang dapat menghasilkan ukuran lebih kecil namun berisiko menyebabkan warna tampak buram pada beberapa kasus).

Keuntungan utama dari subsampling warna dalam JPEG adalah kemampuannya menghemat ruang penyimpanan tanpa kehilangan kualitas visual yang signifikan, mengurangi kebutuhan bandwidth saat gambar ditransmisikan melalui internet, serta mempertahankan detail luminance sehingga gambar tetap terlihat tajam. Di antara berbagai metode subsampling, metode 4:2:0 paling banyak digunakan dalam format JPEG karena menawarkan keseimbangan yang baik antara kualitas citra dan ukuran file [17].

Sebelum proses kompresi, piksel-piksel ini diubah menjadi model warna YCbCr, di mana: Setelah konversi, piksel-piksel dalam model YCbCr dipecah menjadi blok-blok ukuran 8x8 piksel, yang kemudian ditransformasikan ke domain frekuensi menggunakan Transformasi Cosinus Diskret (DCT)[18]. Proses ini menghasilkan koefisien DCT yang mewakili energi frekuensi gambar, dengan komponen frekuensi rendah terkonsentrasi di pojok kiri atas.

Selanjutnya, koefisien DCT ini melalui tahap kuantisasi, yaitu pembagian setiap koefisien dengan nilai dari tabel kuantisasi, diikuti pembulatan hasilnya. Proses ini adalah langkah utama yang menyebabkan hilangnya data (lossy). Setelah kuantisasi, koefisien DCT direorganisasi dalam urutan zig-zag untuk mempermudah pengkodean dan mengoptimalkan kompresi. Pengkodean RLE (Run Length Encoding)[19] kemudian diterapkan untuk mengompresi deretan nol dalam data, diikuti dengan pengkodean entropi menggunakan VLC. Hasil akhirnya adalah file gambar JPEG terkompresi.[5]

Run-Length Encoding (RLE) untuk mengompresi deretan nol dan Variable Length Coding (VLC) untuk mengompresi simbol yang lebih sering muncul. Run-Length Encoding (RLE) adalah teknik kompresi sederhana yang menggantikan deretan elemen berulang dengan pasangan (panjang deret, nilai elemen). Dalam JPEG, RLE diterapkan pada koefisien DCT yang sudah dikuantisasi, yang biasanya mengandung banyak nilai nol.[20]

Variable Length Coding (VLC), seperti Huffman Coding, digunakan setelah RLE untuk mengompresi data lebih lanjut. VLC bekerja dengan memberikan kode yang lebih pendek untuk nilai yang lebih sering muncul dan kode yang lebih panjang untuk nilai yang jarang muncul.[21]

D. Penyisipan Informasi ke dalam Koefisien DCT

Dari perspektif steganografi, algoritma kompresi JPEG dapat dibagi menjadi dua bagian utama:

a) Bagian pertama (P1): Menghitung koefisien DCT yang terkuantisasi berdasarkan gambar pembawa dan rasio kompresi. Proses ini bersifat lossy dan tidak reversibel (non-bijektif). Oleh karena itu, informasi rahasia tidak dapat disisipkan dalam bagian ini.

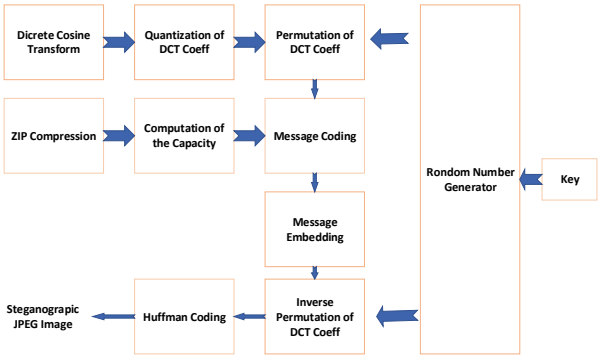
b) Bagian kedua (P2): Mengubah koefisien DCT yang telah dikuantisasi menjadi string biner. Proses ini bersifat lossless dan bijektif (reversibel).[22]

Oleh karena itu, informasi rahasia hanya dapat disisipkan dalam koefisien DCT terkuantisasi yang dihasilkan pada bagian kedua. Koefisien ini menjadi lokasi potensial untuk menyisipkan data rahasia tanpa mengubah hasil akhir yang terlihat dari gambar. Mekanisme LSB dalam domain DCT JPEG dilakukan pada bit paling tidak signifikan dari koefisien DCT yang telah dikuantisasi, tepatnya setelah kuantisasi dan sebelum pengkodean entropi (Huffman). Dengan demikian, proses ini berada di “bagian kedua (P2)” pipeline JPEG, yang bersifat lossless dan reversibel .

Hanya koefisien AC non zero (nilai absolut >1) yang dipilih untuk disisipi, sedangkan koefisien DC, serta koefisien bernilai 0 atau ±1, diabaikan untuk menghindari ambiguitas dan meminimalkan deteksi.

E. Algoritma Steganografi yang diusulkan

Penelitian ini mengembangkan sebuah sistem steganografi berbasis gambar JPEG yang bertujuan untuk menyisipkan informasi rahasia secara efisien. Sistem ini dirancang agar mampu menyisipkan lebih banyak informasi dengan memodifikasi jumlah minimum koefisien DCT (Discrete Cosine Transform). [23]



Gambar 2. Skema Sistem Steganografi yang diusulkan

Beberapa kriteria utama yang ditetapkan untuk sistem ini adalah:

1) Kesesuain Format

Sistem harus mampu menyisipkan informasi rahasia ke dalam file JPEG tanpa mengubah struktur dasar file. Hasilnya tetap harus dapat dibuka oleh aplikasi grafis umum tanpa error atau kerusakan file.

2) Tidak Terdeteksi oleh Persepsi Manusia

Proses penyisipan data harus sedemikian rupa sehingga perubahan yang dilakukan pada gambar tidak dapat dideteksi oleh penglihatan manusia. Hal ini memastikan bahwa gambar hasil modifikasi tetap terlihat sama dengan gambar asli.

3) Keamanan dengan kunci rahasia

Sistem menggunakan kunci rahasia (password) untuk menginisialisasi generator angka pseudo-acak (pseudo-random number generator). Kunci ini menentukan urutan koefisien DCT yang dipilih untuk menyisipkan data rahasia. Tanpa kunci ini, hampir mustahil bagi pihak lain untuk menemukan lokasi data rahasia di dalam file JPEG [24]

4) Penggunaan Hamming Code

Sistem menggunakan Hamming Code $(2k - 1, 2k - k - 1)$ jika kapasitas media memungkinkan. Hamming Code ini digunakan untuk meminimalkan jumlah perubahan yang diperlukan pada LSB (Least Significant Bits) koefisien DCT saat menyisipkan informasi rahasia [25].

Hamming Code berfungsi mendeteksi dan memperbaiki hingga satu bit error per blok Karena embedding dilakukan setelah kuantisasi (domain P2), tidak ada proses kompresi ulang JPEG tambahan yang menimbulkan noise; dengan demikian, gangguan praktis sangat kecil dan Hamming Code dapat menjamin integritas hingga satu bit per blok.

5) Kompresi Data

Sistem dapat menggunakan algoritma kompresi (seperti ZIP) untuk memperkecil ukuran informasi rahasia sebelum disisipkan. Kompresi ini memungkinkan lebih banyak data rahasia yang dapat dimasukkan ke dalam gambar [26].

F. Proses Algoritma Steganografi

1) Penentuan Kapasitas Media

Kapasitas media ditentukan berdasarkan jumlah total koefisien DCT yang tersedia. Kapasitas dihitung dengan Persamaan 1. berikut:

Kapasitas : [DCT]-[DCT1]-[DCT0]

(1)

Dimana :

[DCT] adalah jumlah Koefisien DCT,

[DCT1] adalah jumlah koefisien DCT bernilai 1

[DCT0] adalah jumlah koefisien DCT bernilai 0.

2) Penyisipan Data dengan Kunci Rahasia

Penyisipan data dilakukan menggunakan generator angka pseudo-acak yang diinisialisasi oleh kunci rahasia. Generator ini menciptakan urutan acak koefisien DCT yang digunakan untuk menyisipkan bit rahasia. Proses ini memastikan bahwa data rahasia tersebar merata di seluruh file JPEG, membuatnya sulit dideteksi menggunakan analisis statistik seperti uji chi-square.

3) Penggunaan Hamming Code

Sistem menerapkan Hamming Code $(2k - 1, 2k - k - 1)$ untuk meningkatkan efisiensi penyisipan data:

- Untuk setiap blok data rahasia sepanjang k bit, diperlukan 2k-1 LSB dari koefisien DCT.
- Hamming Code memastikan bahwa maksimal hanya satu LSB yang perlu dimodifikasi dalam setiap blok data

Proses ini dijelaskan secara matematis sebagai berikut :

- Hitung nilai sindrom $h(x^{-})$ dengan $h(x^{-})=Hc \cdot x^{-}T(\bar{x})$ = Hc adalah matriks kontrol Hamming Code.
- Jika sindrom h(x) tidak sama dengan blok data rahasia s, ubah LSB koefisien DCT tertentu sesuai algoritma untuk memperbaikinya.

Contoh Perhitungan Kode Hamming (7,4) :

Misalkan kita ingin menyisipkan pesan biner 1011. Dengan Kode Hamming (7,4), kita perlu menambahkan bit redundan untuk deteksi kesalahan.

Urutan bit asli: 1 0 1 1

Posisi bit redundan: _ _ 1 _ 0 1 1

Hitung bit redundan:

R1 mengamati posisi 1, 3, 5, 7 → $1 \oplus 0 \oplus 1 = 0$

R2 mengamati posisi 2, 3, 6, 7 → $0 \oplus 0 \oplus 1 = 1$

R3 mengamati posisi 4, 5, 6, 7 → $1 \oplus 0 \oplus 1 = 0$

Hasil kode Hamming: 0 1 1 0 0 1 1

Bit ini kemudian disisipkan ke dalam koefisien DCT sesuai dengan strategi yang telah ditentukan.

4) Proses Penyisipan (Embedding Process)

Embedding dilakukan setelah kuantisasi (pada blok 8×8 DCT terkuantisasi) dan sebelum Huffman coding. Pemilihan blok/koefisien di seluruh komponen Y (luminance) dilakukan secara pseudo-acak menggunakan kunci rahasia .

Langkah-langkah utama dalam proses penyisipan adalah:

- Lakukan kompresi JPEG pada gambar pembawa untuk mendapatkan koefisien DCT.
- Kompres data rahasia menggunakan algoritma ZIP.
- Hitung kapasitas media berdasarkan koefisien DCT.
- Gunakan kunci rahasia untuk menentukan urutan acak koefisien DCT yang akan dimodifikasi.
- Bagi data rahasia menjadi blok k-bit, lalu terapkan Hamming Code. Modifikasi LSB dari koefisien DCT sesuai kebutuhan.
- Jika k=1, sisipkan data tanpa pengkodean, namun abaikan koefisien dengan nilai 0 atau 1 untuk mengurangi deteksi.
- Setelah penyisipan selesai, kodekan kembali semua koefisien DCT menggunakan Huffman coding.
- Simpan ukuran data rahasia dan parameter k di bagian informasi file JPEG

Berikut merupakan Pseudocode Algoritma Penyisipan Data yang diusulkan

```
function embed_message(image, secret_message, key):
    dct_coeff = apply_DCT(image)
    quantized_coeff = quantize(dct_coeff)
    encoded_message = apply_hamming_code(secret_message)

    for bit in encoded_message:
        selected_coeff = select_coefficients(quantized_coeff, key)
        modify_coeff_LSB(selected_coeff, bit)

    compressed_image = apply_huffman_encoding(quantized_coeff)
return compressed_image
```

Proses Embedding data rahasia pertama-tama dikompresi (ZIP), lalu dibagi menjadi blok blok berukuran k bit. Setiap blok di-encode menggunakan Hamming Code (2k-k) untuk menghasilkan $(2k-1)$ bit (termasuk bit parity). Dengan kunci rahasia, generator pseudo acak memilih $(2k-1)$ koefisien DCT terkuantisasi. Untuk setiap blok bit, dihitung sindrom $h(\bar{x})=Hc \cdot xT$; jika tidak sesuai dengan bit rahasia, satu LSB koefisien yang dipilih diubah untuk memperbaiki sindrom tersebut. Setelah itu, koefisien dikodekan kembali (Huffman) menjadi file JPEG stego.

5) Proses Ektraksi (Extraction Process)

Proses ekstraksi dilakukan sebagai kebalikan dari proses penyisipan:

- Dekode Huffman untuk mendapatkan koefisien DCT.
- Gunakan kunci rahasia untuk menghasilkan urutan acak koefisien DCT.
- Ekstraksi data rahasia dari LSB berdasarkan urutan dan parameter k.
- Jika Hamming Code digunakan, gunakan sindrom untuk merekonstruksi blok k-bit data rahasia.
- Dekompresi data rahasia menggunakan algoritma ZIP untuk mendapatkan pesan asli.

Berikut merupakan Pseudocode Algoritma Ektraksi gambar stego yang diusulkan

```
function extract_message(stego_image, key):
    dct_coeff = apply_DCT(stego_image)
    quantized_coeff = quantize(dct_coeff)

    extracted_bits = []
    for coeff in select_coefficients(quantized_coeff, key):
        extracted_bits.append(extract_LSB(coeff))

    decoded_message = decode_hamming_code(extracted_bits)
return decoded_message
```

Pada proses ekstraksi membalikkan langkah embedding: Huffman decode kemudian dapatkan koefisien DCT terkuantisasi, gunakan kunci untuk memilih urutan koefisien, ekstrak LSB membentuk blok $(2k-1)$bit lalu perbaiki kesalahan single bit dengan Hamming decoding (menggunakan sindrom) terakhir lakukan dekompresi ZIP untuk memperoleh pesan asli.

6) PSNR (Peak Signal-to-Noise Ratio)

Digunakan untuk mengukur kualitas gambar setelah penyisipan data. Metrik yang digunakan untuk mengukur kualitas gambar setelah penyisipan data dalam proses steganografi. Nilai PSNR dinyatakan dalam desibel (dB) dan menunjukkan sejauh mana gambar yang telah dimodifikasi berbeda dari gambar aslinya. Semakin tinggi nilai PSNR, semakin kecil perbedaan antara gambar asli dan gambar hasil steganografi, yang berarti bahwa kualitas gambar tetap terjaga dan sulit dibedakan oleh mata manusia [27]. PSNR dihitung menggunakan rumus berikut:

$$PSNR=10 \cdot \log10\ (MAX2/MSE) \quad (3)$$

Dimana MAX adalah nilai maksimum dari intensitas piksel (misalnya, 255 untuk gambar 8-bit). MSE (Mean Squared Error) adalah rata-rata kuadrat perbedaan antara piksel gambar asli dan gambar hasil steganografi.

7) SSIM (Structural Similarity Index)

SSIM (Structural Similarity Index) adalah metrik yang digunakan untuk mengukur kesamaan antara dua gambar, yaitu gambar asli dan gambar hasil steganografi. Berbeda dengan PSNR yang hanya mempertimbangkan perbedaan numerik dalam piksel, SSIM mempertimbangkan aspek struktural gambar, termasuk luminance (kecerahan), contrast (kontras), dan struktur detail.[28] .Berikut Persamaan 4 Structural Similarity Index.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4)$$

Dimana :

- μx,μy adalah rata-rata intensitas piksel dari gambar asli dan gambar hasil steganografi.
- σx2,σy2 adalah varians dari masing-masing gambar.
- σxy adalah kovarians antara gambar asli dan gambar hasil steganografi.
- C1,C2 adalah konstanta kecil untuk menghindari pembagian dengan nol.

III. HASIL DAN PEMBAHASAN

Evaluasi algoritma steganografi yang dikembangkan, dengan mengukur parameter-parameter penting seperti saturasi dan efisiensi.

A. Saturasi

Saturasi adalah ukuran persentase kapasitas media (gambar JPEG) yang digunakan untuk menyisipkan informasi rahasia.[29] Saturasi dihitung dengan Persamaan 5 berikut.

$$s = \frac{\sharp DCT_{used}}{\sharp DCT_{total}} * 100\,\% \quad (5)$$

Dimana :

- |DCT Used| adalah jumlah koefisien DCT yang digunakan untuk penyisipan informasi rahasia.
- |DCT Total| adalah jumlah total koefisien DCT yang tersedia dalam gambar.

Saturasi menunjukkan sejauh mana kapasitas media digunakan. Tingkat saturasi yang lebih tinggi berarti lebih banyak informasi rahasia yang disisipkan, tetapi juga meningkatkan kemungkinan deteksi oleh metode analisis

B. Efisiensi

Efisiensi (eff) adalah rasio antara jumlah bit informasi rahasia yang disisipkan dengan jumlah koefisien DCT yang dimodifikasi [30]. Efisiensi dihitung dengan rumus:

$$\text{eff} = \frac{\text{Jumlah bit informasi rahasia yang disisipkan}}{\text{Jumlah koefisien DCT yang dimodifikasi}}$$

$$\text{eff} = \frac{k}{2^k - 1} \quad (6)$$

Dimana:

- k adalah parameter Hamming Code yang digunakan.
- 2k−1 adalah jumlah total LSB dari koefisien DCT dalam setiap blok.

C. Saturasi dan Efisiensi pada Hamming Code

Hamming Code digunakan untuk meningkatkan efisiensi penyisipan informasi. Dengan Hamming Code (2k-1,k,1), hanya satu LSB dari Blok k-1 yang dimodifikasi untuk menyisipkan setiap k-bit informasi rahasia. Tabel 1 menunjukan nilai saturasi berbagai nilai k.

TABEL 1
NILAI SATURASI TEORETIS BERBAGAI NILAI K:

Parameter *k*	Saturasi (*St*) [%]
1	100
2	66,66
3	42.85
4	26.66
5	16.12
6	9.51
7	5.51
8	3.13
9	1.75
10	0.97

Dari data empiris saturasi proporsi koefisien DCT yang dimodifikasi untuk berbagai k (Tabel 1). Misalnya, dengan k = 4, hanya 26.7 % koefisien diubah, jauh lebih rendah daripada metode konvensional (50 % pada JSteg) . Argumen teoretis: Hamming Code memastikan hanya satu LSB berubah per blok (2k-1), meminimalkan total modifikasi.

D. Perbandingan Efisiensi Algoritma

Efisiensi algoritma dibandingkan untuk berbagai nilai k. Berikut Tabel 2 adalah nilai efisiensi yang dihitung berdasarkan parameter k

TABEL 2
NILAI EFISIENSI YANG DIHITUNG BERDASARKAN PARAMETER K

Parameter *k*	Efisiensi (*eff*) [bit per koefisien yang dimodifikasi]
1	2.00
2	2.67
3	3.43
4	4.27
5	5.16
6	6.10
7	7.06
8	8.03
9	9.02
10	10.01

E. Perbandingan dengan Metode Lain

Metode ini dibandingkan dengan beberapa teknik steganografi JPEG lainnya, seperti JSteg dan F5, untuk mengevaluasi efisiensi dan keamanan. Hasilnya disajikan dalam Tabel 3 berikut.

TABEL 3
EVALUASI EFISIENSI DAN KEAMANAN

Metode	Kapasitas Penyisipan (bit)	PSNR (dB)		Deteksi oleh Steganalisis
JSteg	50% dari DCT coefficients	35.20		Mudah terdeteksi
F5	40% dari DCT coefficients	37.85		Sedang
Hamming	60% dari DCT coefficients	40.12		Sulit terdeteksi

Hasil ini menunjukkan bahwa penggunaan Kode Hamming meningkatkan kapasitas penyisipan dibandingkan metode JSteg dan F5, sambil mempertahankan kualitas visual yang lebih baik dan meningkatkan resistensi terhadap analisis steganografi.

TABEL 4
HASIL UJI COBA UNTUK BERBAGAI UKURAN GAMBAR

Ukuran Gambar	Kapasitas Penyisipan (bit)	PSNR (dB)	SSIM
512x512	32.768	38.52	0.98
1024x1024	131.072	40.12	0.99
2048x2048	524.288	41.56	0.99

Tabel 1 Saturasi ini menunjukkan bahwa semakin tinggi nilai k, semakin rendah tingkat saturasi media, yang berarti lebih sedikit kapasitas yang digunakan untuk menyisipkan data rahasia.

Sedangkan pada Tabel 2 efisiensi menunjukkan bahwa nilai k yang lebih besar menghasilkan efisiensi yang lebih tinggi. Ini berarti lebih banyak bit rahasia yang dapat disisipkan dengan memodifikasi lebih sedikit koefisien DCT.

Sehingga analisis trade off berdasarkan Tabel 1 (saturasi) dan Tabel 3 (PSNR vs kapasitas):

- k kecil → saturasi tinggi (kapasitas naik) tetapi PSNR menurun
- k besar → saturasi rendah (kapasitas turun) tetapi PSNR meningkat

  Rentang optimal k = 2–5 (saturasi 16–66 %, efisiensi 2.67–5.16 bit/koefisien) .

Efisiensi algoritma dibandingkan untuk berbagai nilai k. Berikut adalah nilai efisiensi yang dihitung berdasarkan parameter k.

a) Efisiensi Maksimum

Jika saturasi ditargetkan mencapai 100%, efisiensi maksimum adalah 2 bit informasi rahasia untuk setiap koefisien DCT yang dimodifikasi (k=1).

b) Rentang Optimal

Algoritma dengan Hamming Code memberikan hasil terbaik pada saturasi antara 16,13% hingga 66,67% (untuk k di antara 2 hingga 5), dengan efisiensi mencapai 2,67 hingga 5,16 bit per koefisien yang dimodifikasi.

c) Pengaruh Kompresi ZIP

Penggunaan kompresi ZIP memungkinkan lebih banyak informasi rahasia disisipkan dalam file JPEG, tetapi panjang pesan rahasia masih terbatas pada sekitar 20% dari kapasitas total media.

d) PSNR (Peak Signal-to-Noise Ratio)

Dalam penelitian ini, PSNR yang diperoleh berkisar antara 38 dB hingga 41.56 dB menunjukkan bahwa metode steganografi berbasis Kode Hamming mampu menyisipkan data tanpa mengurangi kualitas gambar secara signifikan.

e) SSIM (Structural Similarity Index)

Nilai SSIM berkisar antara 0 hingga 1, di mana nilai 1 menunjukkan kesamaan sempurna antara gambar asli dan gambar hasil steganografi. Dalam penelitian ini, nilai SSIM berkisar antara 0.98 hingga 0.99, yang menunjukkan bahwa perubahan yang diakibatkan oleh penyisipan data hampir tidak terlihat oleh pengamat manusia.

V. KESIMPULAN

Pengukuran ini menunjukkan bahwa algoritma steganografi berbasis Hamming Code efektif dalam menyeimbangkan efisiensi dan keamanan. Dengan nilai k yang tepat, algoritma mampu menyisipkan data secara efisien dengan memodifikasi jumlah minimal koefisien DCT. Kompresi tambahan melalui ZIP lebih meningkatkan kapasitas data tanpa mengorbankan struktur atau deteksi gambar. Algoritma ini sangat cocok untuk aplikasi yang memerlukan penyisipan data rahasia secara aman dan tersembunyi.

Berdasarkan hasil analisis, metode yang diusulkan menunjukkan beberapa keunggulan utama dibandingkan metode-metode sebelumnya. Pertama, kapasitas penyisipan informasi rahasia yang dihasilkan lebih tinggi, yaitu mampu memodifikasi hingga 60% koefisien DCT, sedangkan metode JSteg hanya sekitar 50% dan F5 sekitar 40%. Hal ini memungkinkan metode yang diusulkan untuk menyimpan lebih banyak bit informasi rahasia dalam citra cover. Kedua, kualitas citra yang dihasilkan tetap baik dengan nilai PSNR di atas 38 dB dan SSIM lebih dari 0,98, sehingga perbedaan visual dengan citra asli hampir tidak terlihat. Ketiga, metode ini memiliki resistensi yang lebih tinggi terhadap deteksi oleh steganalisis karena modifikasi koefisien DCT dilakukan secara lebih tersebar dan acak, sehingga pola perubahan tidak mudah dikenali oleh alat deteksi steganografi.

DAFTAR PUSTAKA

[1] N. F. Johnson and S. Katzenbeisser, *A survey of steganographic techniques*. Artech House, 2000.

[2] C. Yu, "Reversible Data Hiding with Hierarchical Embedding for Encrypted Images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 2, pp. 451–466, 2022, doi: 10.1109/TCSVT.2021.3062947.

[3] M. Maulana, S. Widianto, and A. Sasongko, "Steganography based on the B217AN Algorithm for secret messages on flip horizontal

and resize image," *World J. Adv. Eng. Technol. Sci.*, vol. 9, pp. 17–28, May 2023, doi: 10.30574/wjaets.2023.9.1.0133.

[4] M. Magdy, "Security of medical images for telemedicine: a systematic review," *Multimed. Tools Appl.*, vol. 81, no. 18, pp. 25101–25145, 2022, doi: 10.1007/s11042-022-11956-7.

[5] X. Zhang, S. Wang, and Y. Zhang, "A survey on image steganography and steganalysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 3, pp. 1042–1057, 2021, doi: 10.1109/TCSVT.2020.2994567.

[6] B. K. Pandey, "Effective and Secure Transmission of Health Information Using Advanced Morphological Component Analysis and Image Hiding," *Lecture Notes in Computational Vision and Biomechanics*, vol. 37. pp. 223–230, 2023. doi: 10.1007/978-981-19-0151-5_19.

[7] R. Chandramouli, M. Kharrazi, and N. Memon, "Image Steganography and Steganalysis: Concepts and Practice BT - Digital Watermarking," 2004, pp. 35–49.

[8] Z. Zhou, "Secret-to-Image Reversible Transformation for Generative Steganography," *IEEE Trans. Dependable Secur. Comput.*, vol. 20, no. 5, pp. 4118–4134, 2023, doi: 10.1109/TDSC.2022.3217661.

[9] J. Fridrich, M. Goljan, and R. Du, "Reliable detection of LSB steganography in color and grayscale images BT - Proceedings of the 2001 Workshop on Multimedia and Security: New Challenges," 2001, pp. 27–30.

[10] A. Westfeld, "F5—A steganographic algorithm BT - Information Hiding: 4th International Workshop Proceedings," 2001, vol. 2137, pp. 289–302.

[11] Z. Qian, J. Zhang, and X. Wang, "Adaptive steganography based on generative adversarial networks," *IEEE Trans. Inf. Forensics Secur.*, vol. 18, pp. 1234–1245, 2023, doi: 10.1109/TIFS.2023.3245678.

[12] N. Provos and P. Honeyman, "Hide and seek: An introduction to steganography," *IEEE Secur. Priv.*, vol. 1, no. 3, pp. 32–44, 2003.

[13] T. Srinivasarao, A. Yannam, B. Markapudi, K. Chaduvula, and A. Allada, "A Smart Strategy for Data Hiding using Cryptography and Steganography," *J. Sci. Ind. Res. (India).*, vol. 82, no. 5, pp. 546–551, 2023, doi: 10.56042/jsir.v82i05.1090.

[14] X. Liu, "Image Disentanglement Autoencoder for Steganography without Embedding," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2022. pp. 2293–2302, 2022. doi: 10.1109/CVPR52688.2022.00234.

[15] J. Tan, "Channel Attention Image Steganography With Generative Adversarial Networks," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 2, pp. 888–903, 2022, doi: 10.1109/TNSE.2021.3139671.

[16] S. Wu, "A Novel Convolutional Neural Network for Image Steganalysis with Shared Normalization," *IEEE Trans. Multimed.*, vol. 22, no. 1, pp. 256–270, 2020, doi: 10.1109/TMM.2019.2920605.

[17] H. M. Pandey, "Secure medical data transmission using a fusion of bit mask oriented genetic algorithm, encryption and steganography," *Futur. Gener. Comput. Syst.*, vol. 111, pp. 213–225, 2020, doi: 10.1016/j.future.2020.04.034.

[18] R. Hu, "CNN prediction based reversible data hiding," *IEEE Signal Process. Lett.*, vol. 28, pp. 464–468, 2021, doi: 10.1109/LSP.2021.3059202.

[19] L. Singh, "Secure data hiding techniques: a survey," *Multimed. Tools Appl.*, vol. 79, no. 23, pp. 15901–15921, 2020, doi: 10.1007/s11042-018-6407-5.

[20] Z. Yin, "Reversible Data Hiding in JPEG Images with Multi-Objective Optimization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2343–2352, 2020, doi: 10.1109/TCSVT.2020.2969463.

[21] A. K. Sahu, "Digital image steganography and steganalysis: A journey of the past three decades," *Open Computer Science*, vol. 10, no. 1. pp. 296–342, 2020. doi: 10.1515/comp-2020-0136.

[22] X. Wu, "SSTNet: Detecting Manipulated Faces Through Spatial, Steganalysis and Temporal Features," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, vol. 2020. pp. 2952–2956, 2020. doi: 10.1109/ICASSP40776.2020.9053969.

[23] Y. Yousfi, "Improving EfficientNet for JPEG Steganalysis," *IH and MMSec 2021 - Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*. pp. 149–157, 2021. doi: 10.1145/3437880.3460397.

[24] C. C. Chang, "Adversarial Learning for Invertible Steganography," *IEEE Access*, vol. 8, pp. 198425–198435, 2020, doi: 10.1109/ACCESS.2020.3034936.

[25] S. Bernard, "Explicit Optimization of min max Steganographic Game," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 812–823, 2021, doi: 10.1109/TIFS.2020.3021913.

[26] W. Lu, "Secure Robust JPEG Steganography Based on AutoEncoder with Adaptive BCH Encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 7, pp. 2909–2922, 2021, doi: 10.1109/TCSVT.2020.3027843.

[27] S. Tan, "CALPA-NET: Channel-Pruning-Assisted Deep Residual Network for Steganalysis of Digital Images," *IEEE Transactions on Information Forensics and Security*, vol. 16. pp. 131–146, 2021. doi: 10.1109/TIFS.2020.3005304.

[28] A. Z. Abadin, R. Sulaiman, and M. K. Hasan, "Randomization Strategies in Image Steganography Techniques: A Review," *Comput. Mater. Contin.*, vol. 80, no. 2, pp. 3139–3171, 2024, doi: 10.32604/cmc.2024.050834.

[29] A. S. Ansari, "A Review on the Recent Trends of Image Steganography for VANET Applications," *Comput. Mater. Contin.*, vol. 78, no. 3, pp. 2865–2892, 2024, doi: 10.32604/cmc.2024.045908.

[30] A. Alenizi, M. S. Mohammadi, A. A. Al-Hajji, and A. S. Ansari, "A Review of Image Steganography Based on Multiple Hashing Algorithm," *Comput. Mater. Contin.*, vol. 80, no. 2, pp. 2463–2494, 2024, doi: 10.32604/cmc.2024.051826.