PENYEMBUNYIAN DATA YANG *REVERSIBLE* MENGGUNAKAN METODE MODIFIKASI HISTOGRAM

REVERSIBLE DATA HIDING USING THE HISTOGRAM MODIFICATION BASED METHODS

Muhammad Arly Gunawan[1], Ledya Novamizanti[2], Rustam[3]
[123]Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom
[1]**arlygnwn@student.telkomuniversity.ac.id,** [2]**ledyaldn@telkomuniversity.ac.id,**
[3]**rustamtelu@telkomuniversity.ac.id**

Abstrak

Tugas akhir ini mengusulkan skema *reversible data hiding* menggunakan metode modifikasi histogram. Proses *embedding* yang dilakukan menggunakan *threshold* yang ditentukan sebagai zona '1' dan '0' dengan pergeseran nilai blok. Proses *extraction* yang dilakukan dengan mengekstraksi *payload* pada zona '1' dan '0' sesuai dengan *threshold* yang ditentukan dan memulihkan gambar asli dengan pergeseran nilai blok. Ketika *stego image* terkena serangan yang dilakukan hanya memulihkan *payload* tanpa gambar asli, pemulihan *payload* menggunakan jumlah nilai blok pada zona '1' dan '0'.

Berdasarkan pengujian yang telah dilakukan, telah diperoleh hasil performansi skema tanpa serangan dengan rata-rata *PSNR* dengan nilai 45.207 dB dan *BER* dengan nilai 0 pada ukuran blok 16×16. Hasil performansi skema terhadap serangan kompresi *JPEG* dengan rata-rata nilai *BER* <0.1 ketika faktor kualitas ≥ 60. Selain itu hasil performansi skema terhadap serangan *salt and pepper*, *awgn, gaussian filter*, dan *contrast adjusment* dengan batas parameter tertentu dapat memulihkan *payload* tanpa ada kesalahan dengan nilai *BER*<0.1.

Kata Kunci: ***reversible data hiding, modifikasi histogram***

Abstract

This final task analyzes and implements reversible data hiding using the histogram modification. The embedding process is carried out using the specified threshold as zones '1' and '0' with a shift in block value as data embedding. The extraction process is performed by extracting the payload in zones '1' and '0' according to the specified threshold and restoring the original image with a shift in block value. When the stego image is attacked, it only extracts the data without recovering the original image completely, the payload extraction uses a new threshold that is adjusted to the number of block values contained in zone '1' and zone '0'.

Based on the experiments that have been carried out, the results of performance scheme without attacks have been obtained with an average PSNR of 45.207 dB and a BER of 0 at a block size of 16×16. The results of performance scheme against JPEG compression with an average BER value <0.1 when the quality factor ≥ 60. The performance results of scheme against salt and pepper noise, awgn noise, gaussian filter, and contrast adjusment with a certain parameter limit can recover the payload without any error with a value of BER <0.1.

Keywords: ***reversible data hiding, histogram modification***

1. Pendahuluan

Penyebaran media *digital* yang semakin mudah menyebabkan tindakan seperti pencurian, duplikasi, distribusi data secara ilegal tanpa memperhatikan hak cipta. Dalam hal ini perlindungan data sangat dibutuhkan untuk menjaga keaslian data dan hak cipta pada dunia *digital*. Pada *digital data hiding* konvensional, data yang terdistorsi tidak dapat dipulihkan kembali menjadi keadaan asli. *Reversible data hiding* dapat menyembunyikan informasi pada suatu gambar dan memulihkan kembali tanpa adanya informasi yang hilang.

Penelitian yang terkait pada *reversible data hiding* menggunakan metode modifikasi histogram diantaranya yaitu Ni, dkk mengusulkan skema penyembunyian data gambar *lossless* yang *robust* menggunakan perbedaan rata-rata aritmatika blok, jika *bit* '1' disisipkan maka menggeser nilai selisih rata-rata aritmatika menjauh dari 0 dengan kuantitas pergeseran, jika bit '0' disisipkan maka blok tidak berubah [1]. Zeng,dkk. mengusulkan skema untuk meningkatakan performansi dari skema Ni dengan menambah dua *threshold* untuk ruang ekstra pada penyisipan data dan membuat aturan pergeseran aritmatika yang baru [2]. Li, dkk. mengusulkan skema baru *histogram shifting* yang *robust* dan *reversible* mengusulkan skema untuk meningkatkan performansi dari skema Zeng dengan cara menggunakan seluruh blok pada proses *embedding* [3].

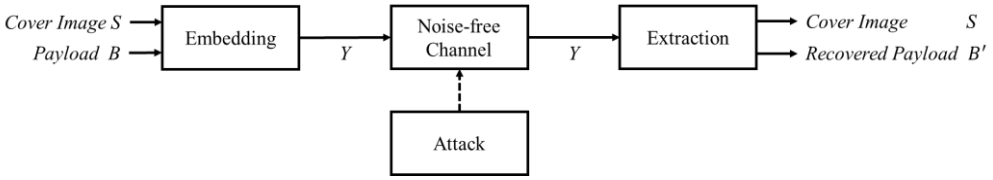
Tugas Akhir ini dilakukan penyembunyian data yang *reversible* menggunakan metode modifikasi histogram. Pada proses penyisipan dan ekstraksi digunakan skema Zeng dengan tambahan yaitu menambahkan zona bits '0'

dan menggunakan seluruh blok aritmatika saat menyisipkan data dengan hasil meningkatkan nilai *PSNR* dan ketahanan terhadap kompresi *JPEG* ataupun beberapa serangan lainnya.

2. Dasar Teori

2.1 Reversible Data Hiding

Reversible data hiding merupakan salah satu teknik dalam *digital watermarking*, dimana teknik ini menyembunyikan informasi dengan memodifikasi nilai *pixel* pada gambar dan dapat memulihkan gambar aslinya setelah mengekstraksi informasi [4]. Teknik *reversible data hiding* memungkinkan bebas distorsi, *invertible* dan *lossless* atau terhapusnya informasi.



Gambar 1 Diagram Blok pada Skema *Reversible Data Hiding*

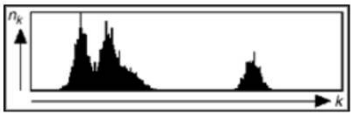
Pada Gambar 1 menunjukan blok diagram pada skema *reversible data hiding*. P *reversible data hiding* suatu *payload* (*B*) disembunyikan dalam *cover image* (*S*), yang menghasilkan *stego image* (*Y*) dan ditransmisikan melalui *channel* komunikasi. Pada proses *extraction* dibutuhkan *stego image* (*Y*) untuk merekontruksi *cover image* (*S*) dan memulihkan *payload* (B').

2.2 Histogram

Histogram adalah grafik yang menggambarkan penyebaran nilai intensitas *pixel* dari suatu gambar atau bagian tertentu di dalam gambar [5]. Gambar yang memiliki *L* derajat keabuan (*grey level*), yaitu nilai 0 sampai *L-1* secara matematis dapat dihitung menggunakan persamaan berikut.

$$h_i = \frac{n_i}{n}, i = 0,1, \dots, L-1 \tag{2}$$

dengan n_i adalah jumlah *pixel* yang memiliki derajat keabuan i dan n adalah jumlah seluruh *pixel* didalam gambar.

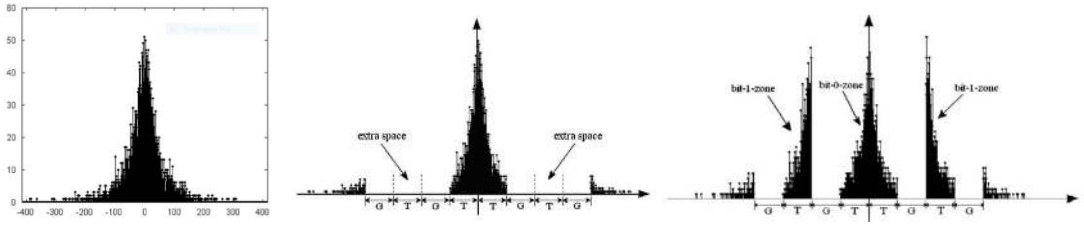


Gambar 2 Histogram

Pada Gambar 2 memperlihatkan contoh gambar histogram, dimana k adalah derajat keabuan, dan n_k adalah jumlah *pixel* yang memiliki keabuan k.

2.3 Modifikasi Histogram

Metode modifikasi histogram oleh Zeng dkk [2], menggunakan perbedaan aritmatika blok yang menghasilkan histogram dan menyisipkan data dengan aturan pergeseran pada histogram.



Gambar 3 Distribusi α pada Gambar (kiri) Sebelum; (tengah) Setelah eksplor ruang ekstra; (kanan) Setelah Penyisipan Data

Pertama, gambar dibagi menjadi *m*×*n* blok *non-overlapping* dan menghitung perbedaan aritmatika blok α. Matriks *M* dinyatakan dengan persamaan berikut.

$$M(i,j) = \begin{cases} 1 & mod(i,2) = mod(j,2) \\ -1 & mod(i,2) \neq mod(j,2) \end{cases} \tag{3}$$

Dengan $i \in [1,m], j \in [1,n]$, dan $mod(x,2)$ adalah *mod-2 function*.

$$\alpha^{(k)} = \sum_{i=1}^{m} \sum_{j=1}^{n} \left(C^{(k)}(i,j) \times M(i,j) \right) \tag{4}$$

Dengan *superscript* (k) menunjukan blok ke-kth dan $C^{(k)}(i,j)$ menunjukan nilai piksel dari titik (i,j) di blok ke-kth. Distribusi dari α ditunjukan pada Gambar 3 (kiri), dengan *X axis* menunjukan nilai α sedangkan *Y axis* jumlah α.

Selanjutnya, dua *threshold,* masing-masing dilambangkan T dan G, keduanya bilangan bulat positif, dan eksplor ruang ekstra untuk menyisipkan data.

$$S_1^{(k)}(i,j) = \begin{cases} C^{(k)}(i,j) + \beta_1, & \alpha > T \text{ dan } mod(i,2) = mod(j,2) \\ C^{(k)}(i,j) + \beta_1, & \alpha < -T \text{ dan } mod(i,2) \neq mod(j,2) \\ C^{(k)}(i,j) & otherwise \end{cases} \quad (5)$$

Dengan $i \in [1,m], j \in [1,n]$, $\beta_1 = \lceil \frac{(2\times G+T)}{m\times n} \rceil$, dan symbol $\lceil . \rceil$ adalah *ceil function* yang artinya "ke bilangan bulat yang terdekat menuju tak terhingga". Distribusi dari α setelah eksplor ruang ekstra ditunjukan pada Gambar 3 (tengah).

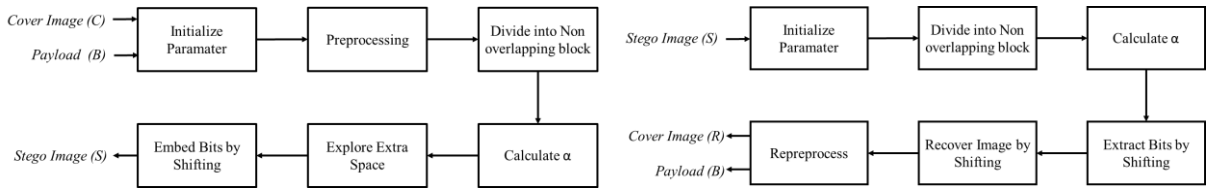
Cek setiap blok dan periksa selisih aritmatika α, jika nilai α berada dalam $[-T,T]$, satu bit dapat disisipkan ke dalam blok. Berikut ini proses penyisipan, Jika bit 0 akan disisipkan, blok ini tetap utuh, dan jika bit 1 akan disisipkan, dapat disisipkan ke dalam blok dengan menggeser selisih aritmatika α. Aturan pergeseran sebagai berikut.

$$S_1^{(k)}(i,j) = \begin{cases} C^{(k)}(i,j) + \beta_2, & \alpha \in [0,T] \text{ dan } mod(i,2) = mod(j,2) \\ C^{(k)}(i,j) + \beta_2, & \alpha \in [-T,0) \text{ dan } mod(i,2) \neq mod(j,2) \\ C^{(k)}(i,j), & otherwise \end{cases} \quad (6)$$

Dengan $i \in [1,m], j \in [1,n]$, $\beta_2 = \lceil \frac{(T+G)\times 2}{m\times n} \rceil$. Distribusi dari α setelah penyisipan data ditunjukan pada Gambar 3 (kanan).

3. Perancangan Sistem

Skema yang diusulkan untuk Tugas Akhir ini, secara umum skema terbagi menjadi dua proses yaitu proses *embedding* dan proses *extraction*. Berikut ini diagram blok dari kedua proses tersebut.



Gambar 4 Diagram Blok Proses (kiri) *Embedding*; (kanan) *Extraction*

3.1 Proses *Embedding*

Pada proses *embedding* dilakukan beberapa proses yang dilakukan. Berikut ini penjelasannya:

1. Masukan *cover image* dengan ukuran h×w dan maksimal ukuran *payload* $\left\lfloor \frac{h}{m} \right\rfloor \times \left\lfloor \frac{w}{n} \right\rfloor$. $\lfloor . \rfloor$ adalah *floor function* yang artinya "*integer* terbesar kurang dari atau sama".
2. Inisialisasi parameter yang digunakan yaitu nilai *threshold* T untuk jumlah histogram *bin* yang digeser agar menghasilkan ruang ekstra dan G untuk menentukan *robustness* pada algoritma, nilai m dan n sebagai pembagi ukuran blok dan St sebagai batas *pixel*.
3. *Preprocessing* ini dilakukan untuk mencegah *overflow* atau *underflow*. Pencegahan ini diawali dengan menentukan tipe gambar *histogram,* berikut ini persamaan untuk tipe gambar:

$$type = \begin{cases} Type\ A, & < 255 - St \text{ dan } > St \\ Type\ B, & < 255 - St \\ Type\ C, & > St \\ Type\ D, & < St \text{ dan } > 255 - St \end{cases} \quad (7)$$

Tipe gambar yang telah ditentukan maka mencari nilai δ menggunakan persamaan berikut:

$$\delta = \begin{cases} 1, & Type\ A\ or\ B \\ -1, & Type\ C \end{cases} \quad (8)$$

Khusus untuk tipe D nilai δ ditentukan dengan jumlah pixel berikut ini persamaan yang digunakan:

$$\delta = \begin{cases} 1, & jumlah\ pixel(< St) < jumlah\ pixel\big(> (255 - St)\big), \\ -1, & jumlah\ pixel(< St) > jumlah\ pixel\big(> (255 - St)\big), \end{cases} \quad (9)$$

dari persamaan diatas selain menentukan nilai δ adapun mengubah nilai pixel <St menjadi St atau nilai pixel >(255-St) diubah menjadi (255-St). nilai pixel asli disimpan dalam *payload* sebagai *overhead information*.

4. *Cover image* yang telah diproses dibagi menjadi *non-overlapping* block dengan masing-masing blok berukuran mxn dengan jumlah blok sebanyak *payload*.

5. Blok yang sudah terbagi dihitung nilai α dengan cara mengalikan dan menjumlahkan blok tersebut dengan matriks M×δ. Persamaan untuk matriks M adalah sebagai berikut:

$$M(i,j) = \begin{cases} 1 & mod(i,2) = mod(j,2) \\ -1 & mod(i,2) \neq mod(j,2) \end{cases} \quad (10)$$

dengan $i \in [1,m], j \in [1,n]$ dan $mod(x,2)$ adalah mod-2 function. Sedangkan persamaan untuk menghitung nilai α adalah sebagai berikut:

$$\alpha^{(k)} = \sum_{i=1}^{m}\sum_{j=1}^{n}(C^{(k)}(i,j) \times M(i,j)) \quad (11)$$

dengan *superscript* (k) menunjukan blok ke-kth dan $C^{(k)}(i,j)$ menunjukan nilai *pixel* dari titik (i,j) di blok ke-kth.

6. Ketika nilai α sudah dihitung maka eksplorasi ruang ekstra menggunakan persamaan berikut ini:

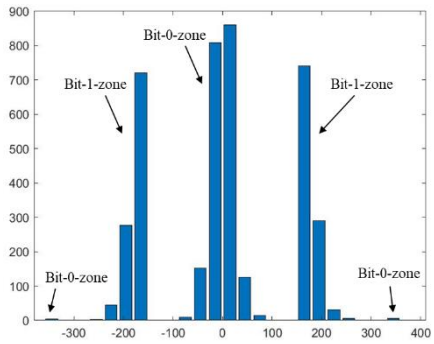
$$S^{(k)}(i,j) = \begin{cases} C^{(k)}(i,j) + \beta_1, & \alpha > T \text{ dan } mod(i,2) = mod(j,2) \\ C^{(k)}(i,j) + \beta_1, & \alpha < -T \text{ dan } mod(i,2) \neq mod(j,2) \\ C^{(k)}(i,j) & otherwise \end{cases} \quad (12)$$

dengan $i \in [1,m], j \in [1,n]$, $\beta_1 = \left\lceil \frac{(2\times G+T)\times 2}{m\times n} \right\rceil$, dan simbol $\lceil . \rceil$ adalah *ceil function* yang artinya "*integer* yang terdekat menuju tak terhingga".

7. Untuk menyisipkan bits *payload* pada *cover image* menggunakan persamaan berikut ini:

$$S^{(k)}(i,j) = \begin{cases} C^{(k)}(i,j) + \beta_2, & \alpha \in [0,T] \text{ dan } mod(i,2) = mod(j,2) \\ C^{(k)}(i,j) + \beta_2, & \alpha \in [-T,0) \text{ dan } mod(i,2) \neq mod(j,2) \\ C^{(k)}(i,j) + \beta_3, & \alpha > T \text{ dan } mod(i,2) \neq mod(j,2) \\ C^{(k)}(i,j) + \beta_3, & \alpha < -T \text{ dan } mod(i,2) = mod(j,2) \\ C^{(k)}(i,j), & otherwise, \end{cases} \quad (13)$$

dengan $i \in [1,m], j \in [1,n]$, $\beta_2 = \lceil \frac{(T+G)\times 2}{m\times n} \rceil$ dan $\beta_3 = \lceil \frac{(G)\times 2}{m\times n} \rceil$.



Gambar 5 Distribusi α pada Lena setelah *Embedding Data*

Pada Gambar 5 menunjukan distribusi α setelah menerapkan persamaan (3.7). Dari persamaan tersebut jika bit '1' disisipkan, nilai pixel digeser oleh β_2 dan ketika nilai α berada diluar $[-T,T]$ maka digeser oleh β_3. Jika bit '0' maka nilai α tidak berubah. Dengan pergeseran ini maka hasil dari nilai α yang berada di jarak $[-2\times T+2\times G,T], [T,2\times T+2\times G]$ adalah bit '1', lainnya adalah bit '0'. Nilai *threshold* T dapat ditentukan oleh nilai harus < α terbesar.

8. Keluaran dari proses ini adalah gambar *stego* (S)**.**

3.2 Proses *Extraction*

Proses ini merupakan proses kebalikan dari proses penyisipan. Berikut ini penjelasan dari proses ekstraksi tersebut:

1. Masukan gambar stego dengan ukuran h×w.
2. Inisialisasi parameter yang digunakan yaitu nilai *threshold* T dan G, nilai m dan n sebagai pembagi ukuran blok dan St sebagai batas *pixel*.
3. Gambar yang telah diproses dibagi menjadi *non-overlapping* block dengan masing-masing blok berukuran m×n dengan jumlah blok sebanyak ukuran *payload*
4. Menghitung nilai α pada masing-masing blok dengan cara yang sama seperti pada langkah 5 dalam proses *embedding*.
5. Untuk mengekstraksi bits menggunakan persamaan berikut:

$$Re\_B = \begin{cases} 1; & \alpha \in (T, 2 \times T + 2 \times G] \\ 1; & \alpha \in [-2 \times T + 2 \times G, -T) \\ 0; & otherwise \end{cases} \quad (14)$$

Dari persamaan (3.8) diatas jika α berada di jarak $[-2 \times T + 2 \times G, -T)$,$(T, 2 \times T + 2 \times G]$ maka nilai bits adalah '1', sedangkan lainnya adalah nilai bits '0'.

6. Untuk mengembalikan *cover image* menggunakan persamaan berikut:

$$R^{(k)}(i,j) = \begin{cases} S^{(k)}(i,j) - \beta_3, & \alpha \in ((2 \times T + G), (2 \times T + 2 \times G)] \\ & \text{dan } mod(i,2) \neq mod(j,2) \\ S^{(k)}(i,j) - \beta_3, & \alpha \in [-(2 \times T + 2 \times G), -(2 \times T + G)) \\ & \text{dan } mod(i,2) = mod(j,2) \\ S^{(k)}(i,j) - \beta_2, & \alpha \in (T, 2 \times T + G] \\ & \text{dan } mod(i,2) = mod(j,2) \\ S^{(k)}(i,j) - \beta_2, & \alpha \in [-2 \times T + G, -T) \\ & \text{dan } mod(i,2) \neq mod(j,2) \\ S^{(k)}(i,j) - \beta_1, & \alpha > (2 \times T + G) \\ & \text{dan } mod(i,2) = mod(j,2) \\ S^{(k)}(i,j) - \beta_1, & \alpha < -(2 \times T + G) \\ & \text{dan } mod(i,2) \neq mod(j,2) \\ S^{(k)}(i,j), & otherwise \end{cases} \quad (15)$$

Dari persamaan diatas nilai pixel digeser oleh β_3 ketika nilai α berada di jarak $[-(2 \times T + 2 \times G), -(2 \times T + G))$ dan $((2 \times T + G), (2 \times T + 2 \times G)]$, ketika nilai α berada di jarak $[-(2 \times T + G), -T)$ dan $(T, (2 \times T + G)]$ maka digeser β_2, ketika nilai $\alpha > (2 \times T + G)$ dan $< -(2 \times T + G)$ maka digeser β_1, lainnya nilai α tidak berubah.

7. *Repreprocessing* ini dilakukan untuk mengubah koordinat dan nilai *pixel* pada *cover image* (R) yang terdapat pada *payload*. untuk tipe A, B dan C tidak perlu dilakukan *re-processing* sedangkan tipe D perlu dilakukan *re-processing*. *Payload* yang telah dipulihkan dikomputasi agar mendapatkan koordinat dan nilai *pixel* lalu pada *cover image* (R) diubah sesuai dengan koordinat dan nilai *pixel* yang telah didapatkan.
8. Output dari proses ini adalah *cover image* (*R*) dan *payload* (*B*).

3.3 Ekstraksi Data untuk *Stego image* yang Terkena Serangan

Stego-image yang telah berubah karena serangan, *cover image* tidak dapat dipulihkan kembali dengan tepat sehingga difokuskan untuk ekstraksi data yang disembunyikan. Untuk mengekstraksi data yang disembunyikan maka disesuaikan jumlah dari '0' dan '1' (N_0, N_1, dan N_2) yang telah dicatat pada proses *embedding*. Proses *extraction* dapat dilakukan dengan langkah berikut ini:

1. Menghitung nilai α pada seluruh blok. tentukan *threshold* V_0 sehingga jumlah α pada jarak $[-T, T]$ berjumlah N_0.
2. Setelah *threshold* V_0 ditentukan maka tentukan *threshold* V_1 sehingga jumlah α pada jarak $[-V_1, -V_0)$ dan $(V_0, V_1]$ berjumlah N_1.
3. Yang terakhir, menentukan *threshold* V_2 sehingga jumlah α pada jarak $[-V_2, -V_1)$ dan $(V_1, V_2]$ berjumlah N_2.

Dari langkah diatas, nilai *threshold* dari 3 daerah dapat diperoleh dan *payload* dapat diekstraksi sesuai dengan jarak α.

4. Pengujian dan Analisa Sistem

4.1 Pengujian Tanpa Serangan

Dalam pengujian ini dilakukan terhadap perubahan ukuran blok dan kapasitas *payload*. Gambar yang digunakan untuk pengujian yaitu lima gambar *gray-scale* yang ditunjukan pada Gambar 4.1 dengan ukuran 512 x 512.

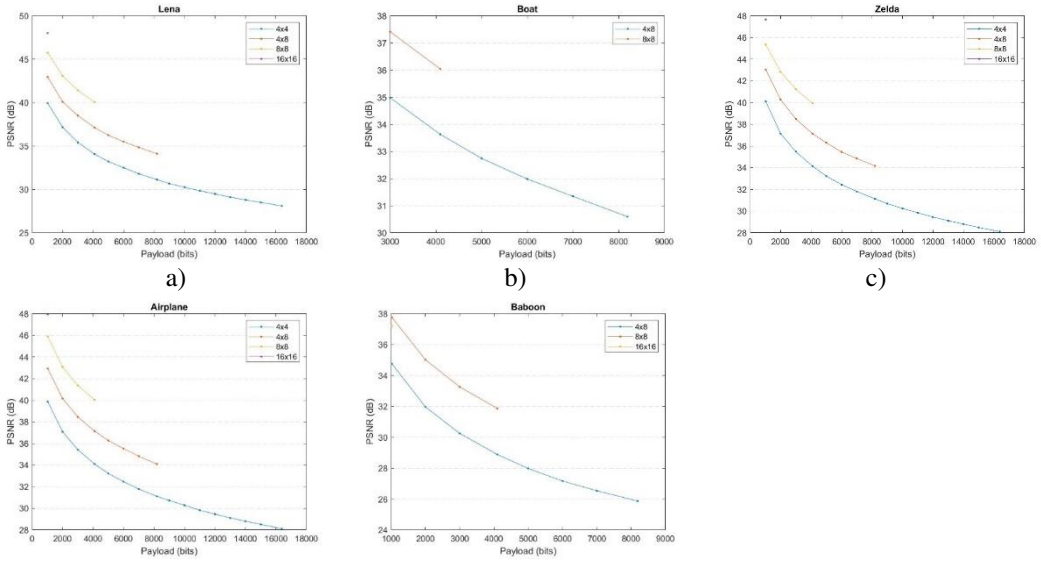
a) Lena b) Boat c) Zelda d) Airplane e) Baboon

Gambar 6 Gambar *Dataset*

Pengujian ini dilakukan menggunakan beberapa parameter yang digunakan. Berikut ini adalah parameter tersebut. Kapasitas *payload* setiap ukuran blok diawali dengan 1000 *bits* hingga maksimum dengan setiap peningkatan sebesar 1000 *bits*. Nilai T dan G dengan masing-masing nilai 80 dan 80. Pengujian ini dinyatakan berhasil ketika *PSNR* pada *Recovery Image* bernilai *inf* dan *BER* bernilai 0.

Gambar 7 Gambar Hasil Pengujian Tanpa Serangan pada Lena (*Payload*=4096 bits)

Pada Gambar 7 menunjukan perbandingan antara *cover image* dan *stego image* dengan ukuran blok 8×8 dan ukuran *payload* yang disembunyikan berjumlah 4096 *bits*. Secara kasat mata pada *stego image* yang menyembunyikan *payload* tidak terjadi perubahan yang signifikan dengan nilai *PSNR* 40.061 dB. *Recovery image* menghasilkan *PSNR* yaitu *inf* yang artinya gambar sama dengan *Cover image* dan *BER* yaitu 0 yang artinya tidak memiliki kesalahan pada *payload* yang diekstraksi.

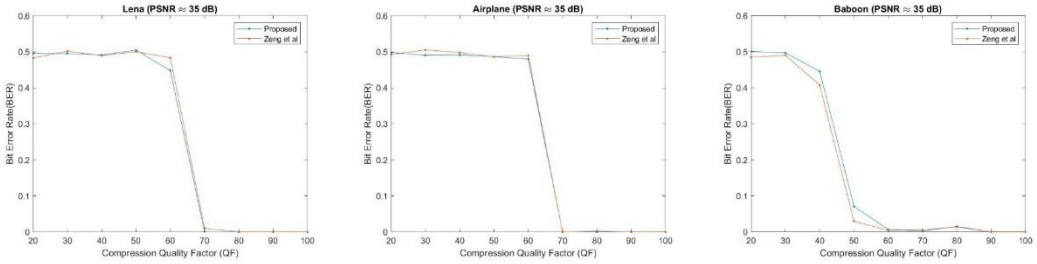


Gambar 8 Hasil Pengujian Blok

Pada Gambar 8 dapat dilihat bahwa kapasitas *payload* menentukan besar atau kecilnya nilai *PSNR*, semakin besar kapasitas *payload* maka semakin kecil nilai *PSNR*, dan sebaliknya. Selain itu ukuran blok menetukan pula besar atau kecilnya nilai *PSNR*, semakin besar ukuran blok maka semakin kecil nilai *PSNR*, dan sebaliknya. Untuk masing-masing nilai *PSNR* ketika *payload* maksimum dengan urutan sesuai label ukuran blok adalah sebagai berikut: Lena (28.09, 34.10, 40.06, 48.01) dB, Boat (30.05, 36.04) dB, Zelda (28.11, 34.17, 39.94, 47.67) dB, Airplane (28.11, 34.09, 40.06, 47.93) dB, dan Baboon (25.87, 31.86, 37.20) dB.

4.2 *Robustness* Terhadap Kompresi JPEG

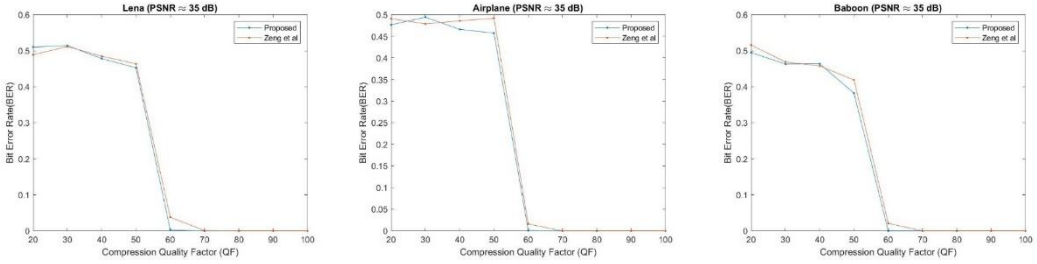
Pengujian ini dilakukan dengan mengompresi kualitas gambar pada *stego image* dan mengukur katahanannya dengan cara menghitung nilai BER. Parameter pada pengujian ini adalah kualitas faktor 20 hingga 100, nilai *threshold* T dan G diatur agar nilai *PSNR* ≈ 35 dB. Ukuran blok yang digunakan yaitu 8×8 dan 16×16. Jika ukuran blok 8×8 *payload* yang digunakan pada Lena dan Airplane adalah 2500 *bits*, dan Baboon adalah 1200 *bits*. Jika ukuran blok 16×16 *payload* yang digunakan adalah 500 bits.



a) b) c)

Gambar 9 Hasil Kompresi JPEG ukuran 8×8

Dari hasil pengujian perbandingan skema usulan dan Zeng saat dapat memulihkan *payload* tanpa ada kesalahan pada Gambar 9 a) skema usulan ketika faktor kualitas ≥ 70 sedangkan pada skema Zeng ketika faktor kualitas ≥ 80, b) skema usulan ketika faktor kualitas ≥ 70 sedangkan pada skema Zeng ketika faktor kualitas = 70 dan ≥ 90, c) kedua skema ketika faktor kualitas ≥ 90, selain itu ketika faktor kualitas ≥ 50 nilai *BER* yang dihasilkan <0.1.



a) b) c)

Gambar 10 Hasil Kompresi JPEG ukuran 16×16

Dari hasil pengujian yang telah dilakukan dapat dilihat pada Gambar 10 a) dan b) skema usulan dan Zeng dapat memulihkan *payload* tanpa ada kesalahan ketika faktor kualitas ≥ 70, akan tetapi pada skema usulan ketika faktor kualitas = 60 lebih mendekati tanpa adanya kesalahan dibanding dengan skema Zeng. Pada Gambar 10 c) skema usulan dapat memulihkan *payload* tanpa ada kesalahan ketika faktor kualitas ≥ 60 sedangkan pada skema Zeng ketika faktor kualitas ≥ 70.

Dari hasil pengujian perbandingan skema usulan dan Zeng saat dapat memulihkan *payload* tanpa ada kesalahan pada Gambar 4.4 a) kedua skema ketika faktor kualitas ≥ 70 akan tetapi skema usulan lebih mendekati tanpa adanya kesalahan, b) dan c) skema usulan ketika faktor kualitas ≥ 60 sedangkan pada skema Zeng ketika faktor kualitas ≥ 70.

4.3 Penilaian Kualitas Gambar

Pada pengujian ini dilakukan dengan melihat kualitas dan struktur gambar terhadap serangan kompresi *JPEG* yang menghasilkan nilai *PSNR* dan *SSIM* yang. Parameter yang ditentukan yaitu; ukuran blok 8×8, *payload* 200 *bits* hingga 1000 bits dengan 200 *bits* setiap peningkatan, dan faktor kualitas 90. Pengujian ini menggunakan gambar Baboon, Lena dan Airplane.

Tabel 1 Hasil *PSNR* dan *SSIM* pada Baboon

Payload (*bits*)	Skema (*PSNR* / *SSIM*)		
	Li et al's	Zeng et al's	Skema Usulan
200	35.7073 / 0.9883	37.5885 / 0.9745	42.4871 / 0.9742
400	35.3941 / 0.9875	35.0130 / 0.9748	39.7406 / 0.9744
600	34.9500 / 0.9865	33.2997 / 0.9751	38.1791 / 0.9745
800	34.6881 / 0.9861	32.0678 / 0.9753	36.9707 / 0.9746
1000	34.4127 / 0.9854	31.1469 / 0.9754	36.0125 / 0.9747

Dari Tabel 1 dapat dilihat ketika *payload* yang disembunyikan sebesar 1000 *bits,* skema usulan menghasilkan *PSNR* yang lebih tinggi yaitu 36.01 dB dibanding dengan skema Li dan Zeng dengan masing-masing nilai 34.41 dB dan 31.14 dB. Untuk struktur kesamaan, skema Li lebih tinggi dengan nilai *SSIM* yaitu 0.9854 dibanding untuk skema usulan dan Zeng yaitu 0,9747 dan 0.9754.

4.4 *Robustness* Terhadap Serangan Lain

Serangan yang diujikan antara lain adalah serangan terhadap *salt and pepper noise*, *AWGN*, *Gaussian filter, Median Filter*, *Histogram Equalization*, dan *Contrast Adjusment*. Parameter yang digunakan dalam pengujian ini dengan ukuran blok 8×8 dengan *payload* yang digunakan adalah 200 dan 500 *bits* nilai *threshold* T dan G diatur

agar nilai *PSNR* ≈ 35 dB**.** Pada serangan *gaussian filter* dan *median filter*, *stego image* diberikan serangan *AWGN* dengan σ=10.

Tabel 2 Hasil *Robustness* Terhadap Serangan Lain

Skema	Salt and pepper noise (*Density*)	AWGN (σ)	Gaussian Filter (*size*)	Median Filter (*size*)	Histogram Equalization (*bin*)	Contrast Adjusment (%)
	0.001	[10,20,30]	5×5	3×3	64	[+20,+10,-10,-10]
Li et al.s	0	0	0.435	0.020	-	-
Zeng et al.'s	0	0	0.385	0.103	0.050	0.095
Usulan	0	0	0.389	0.071	0.007	0

Pada Tabel 2 dapat dilihat bahwa skema usulan dapat memulihkan *payload* tanpa kesalahan terhadap serangan *salt and pepper, AWGN*, dan *Contrast Adjusment*, selain itu terhadap serangan *median filter* dan *histogram equalization* menghasilkan nilai BER<0.1 yaitu 0.071 dan 0.007 dibandingkan dengan skema zeng menghasilkan nilai BER>0.1 yaitu 0.103 dan 0.050 akan tetapi terhadap serangan *gaussian filter* ketiga skema menghasilkan nilai *BER*>0.1 dengan skema usulan dan Zeng lebih resistan dengan nilai BER=0.389 dan 0.385 di bandingkan dengan skema Li yang menghasilkan nilai *BER*=0.435.

5. Kesimpulan dan Saran

5.1 Kesimpulan

Pada Tugas Akhir ini suatu skema penyembunyian data yang *reversible* diusulkan menggunakan metode histogram modifikasi. *cover image* dan *payload* dapat dipulihkan kembali tanpa ada kesalahan ketika *stego image* tidak mengalami perubahan. Hasil *robustness* terhadap kompresi *JPEG* dapat memulihkan *payload* tanpa kesalahan dengan ukuran blok dan faktor kualitas sampai batas. Penilaian kualitas gambar dan kesamaan struktural menghasilkan nilai *PSNR* > 39 dB dengan nilai *SSIM* = 0.9708. *Robustness* terhadap serangan lainnya yaitu *salt and pepper noise, awgn, gaussian filter* dan *contrast adjustment* dapat memulihkan kembali *payload* sampai batas tertentu dengan nilai BER < 0.1.

5.2 Saran

Saran dari penulis adalah dengan adanya *threshold* untuk pembagian zona *bits,* dapat mengembangkan pembagian zona *bits* selain itu karena skema usulan menggunakan spasial domain dapat dikembangkan menggunakan frekuensi *domain* ataupun pengembangan peraturan pergeseran dalah penyembunyian data.

Daftar Pustaka:

[1] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, Q. Sun and X. Lin, "Robust Lossless Image Data Hiding," *2004 IEEE International Conference on Multimedia and Expo (ICME),* vol. 3, pp. 2199-2202, 2004.

[2] X. T. Zeng, X. Z. Pan, L. D. Ping and Z. Li, "A lossless robust data hiding scheme," *Journal of Zhejiang University SCIENCE C,* vol. 11, no. 2, pp. 101-110, 2010.

[3] Q. Li, X. Wang and Q. Pei, "A robust reversible watermarking scheme overcomes the misalignment problem of generalized histogram," *Multimedia Tools and Applications,* vol. 82, no. 5, pp. 7207-7227, 2023.

[4] A. Menendez Ortiz, C. Feregrino Uribe, R. Hasimoto Beltran and J. J. Garcia Hernandez, "A Survey on Reversible Watermarking for Multimedia Content: A Robustness Overview," *IEEE Access,* vol. 7, pp. 132662-132681, 2019.

[5] A. K. Arasu, M. Nizar and D. Prabakaran, "Review of Image Contrast Enhancement Techniques," *INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT),* vol. 2, no. 11, 2013.

[6] L. Novamizanti, A. B. Suksmono, D. Danudirdjo and G. Budiman, "Robust Reversible Watermarking Using Stationary Wavelet Transform and Multibit Spread Spectrum in Medical Images," *International Journal of Intelligent Engineering & System,* vol. 15, no. 3, 2022.

[7] M. I. Rabbani, G. Budiman and L. Novamizanti, "Perancangan Teknik CS dan Sinkronisasi pada Audio Watermarking Stereo Berbasis Lwt dengan Metode Hybrid Cepstrum dan Histogram," *ReTII,* 2017.

[8] F. Adhanadi, L. Novamizanti and G. Budiman, "DWT-SMM-based audio steganography with RSA encryption and compressive sampling," *TELKOMNIKA (Telecommunication Computing Electronics and Control),* vol. 18, no. 2, pp. 1095-1104, 2020.

[9] H. Harahap, G. Budiman and L. Novamizanti, "Implementasi Teknik Watermarking menggunakan FFT dan Spread Spectrum Watermark pada Data Audio Digital," *ELKOMIKA: Jurnal Teknik Energi Elektrik, Teknik Telekomunikasi & Teknik Elektronika,* vol. 4, no. 1, p. 98, 2016.