Department of Digital Business
Journal of Artificial Intelligence and Digital Business (RIGGS)
Homepage: https://journal.ilmudata.co.id/index.php/RIGGS
Vol. 5 No. 2 (2026) pp: 3529-3539
P-ISSN: 2963-9298, e-ISSN: 2963-914X

Pengamanan Informasi Rahasia dan Peta Koordinat Militer Menggunakan Metode Steganografi Least Significant Bit (LSB) Berbasis Software Simulasi

M. Miftah Farid[1], Uvi Desi Fatmawati[2]
[1,2]Program Studi Teknik Elekro, Fakultas Teknik dan Teknologi Pertahanan, Universitas Pertahanan Republik Indonesia
miftah1616f@gmail.com

Abstrak

Penelitian ini membahas pentingnya keamanan informasi dalam ruang lingkup militer, khususnya pada proses pengiriman data strategis yang bersifat rahasia. Dalam komunikasi militer, kebocoran informasi dapat berdampak langsung terhadap keberhasilan operasi, sehingga diperlukan metode pengamanan yang tidak hanya mampu melindungi isi pesan, tetapi juga menyamarkan keberadaan pesan tersebut. Penelitian ini menerapkan teknik steganografi dengan metode Least Significant Bit (LSB) untuk menyisipkan pesan rahasia dan data koordinat operasional ke dalam citra digital. Metode LSB dipilih karena memiliki kemampuan menyembunyikan informasi pada bit terakhir piksel gambar tanpa menimbulkan perubahan visual yang signifikan. Untuk meningkatkan tingkat keamanan, sistem steganografi ini diintegrasikan dengan algoritma enkripsi dan fungsi chaos logistic map, sehingga data yang disisipkan menjadi lebih sulit dikenali maupun diekstraksi oleh pihak yang tidak berwenang. Implementasi dilakukan melalui software simulasi dengan tahapan penyisipan pesan, pembentukan citra stego, ekstraksi data, dan pengujian kualitas citra. Hasil pengujian menunjukkan bahwa citra stego yang dihasilkan tetap memiliki kualitas visual yang baik, ditunjukkan melalui nilai PSNR yang tinggi. Selain itu, proses ekstraksi mampu mengembalikan pesan rahasia secara akurat tanpa merusak isi informasi. Berdasarkan hasil tersebut, metode steganografi LSB yang dikombinasikan dengan enkripsi dan logistic map dinilai efektif sebagai alternatif pengamanan informasi strategis. Penelitian ini menunjukkan bahwa steganografi dapat menjadi solusi pendukung dalam menjaga kerahasiaan komunikasi militer secara aman, tersembunyi, dan sulit dideteksi oleh pihak eksternal.

***Kata kunci**: Steganografi, Least Significant Bit, Enkripsi, Logistic Map, Keamanan Informasi.*

1. Latar Belakang

Di era digital yang serba cepat, keamanan informasi menjadi kebutuhan penting dalam berbagai bidang, terutama pada lingkungan militer yang sangat bergantung pada kerahasiaan, kecepatan, dan ketepatan komunikasi. Informasi yang berkaitan dengan strategi, instruksi, identitas personel, lokasi, serta peta koordinat operasional harus dijaga agar tidak diketahui oleh pihak yang tidak berwenang. Kebocoran informasi dalam konteks militer dapat menimbulkan risiko serius, mulai dari gagalnya suatu rencana, terganggunya koordinasi lapangan, hingga munculnya ancaman terhadap keselamatan personel. Oleh karena itu, sistem pengamanan informasi tidak cukup hanya mengandalkan pengiriman data secara biasa, tetapi perlu dilengkapi dengan metode yang mampu menjaga kerahasiaan sekaligus menyamarkan keberadaan pesan tersebut [1].

Salah satu pendekatan yang dapat digunakan untuk mengamankan informasi rahasia adalah steganografi. Steganografi merupakan teknik menyembunyikan pesan ke dalam media tertentu, seperti citra digital, audio, video, atau dokumen, sehingga keberadaan pesan tidak mudah diketahui oleh pihak luar. Berbeda dengan kriptografi yang mengacak isi pesan agar tidak dapat dibaca, steganografi berusaha menyembunyikan fakta bahwa pesan tersebut sedang dikirim. Dalam konteks komunikasi strategis, pendekatan ini memiliki keunggulan karena pesan rahasia dapat disisipkan ke dalam media yang tampak biasa, misalnya sebuah gambar, sehingga tidak menimbulkan kecurigaan. Dengan demikian, steganografi dapat menjadi lapisan tambahan dalam pengamanan data sensitif.

Penelitian ini mengusulkan penggunaan metode Least Significant Bit atau LSB sebagai teknik dasar dalam proses penyisipan pesan rahasia ke dalam citra digital. Metode LSB bekerja dengan cara mengganti bit paling tidak signifikan pada piksel gambar dengan bit pesan yang ingin disembunyikan. Perubahan pada bit terakhir tersebut umumnya tidak memberikan pengaruh besar terhadap kualitas visual gambar, sehingga citra hasil

penyisipan atau citra stego masih terlihat hampir sama dengan citra aslinya. Keunggulan inilah yang membuat metode LSB banyak digunakan dalam simulasi steganografi, terutama karena prosesnya relatif sederhana, mudah dipahami, dan dapat diimplementasikan menggunakan perangkat lunak simulasi [2].

Dalam penelitian ini, citra digital digunakan sebagai media penampung pesan rahasia dan peta koordinat operasional. Citra dipilih karena memiliki jumlah piksel yang banyak, sehingga mampu menampung data tersembunyi dalam kapasitas tertentu. Pesan yang disisipkan dapat berupa teks, kode koordinat, atau informasi pendukung lain yang berkaitan dengan kebutuhan komunikasi strategis. Proses penyisipan dilakukan dengan mengubah sebagian kecil data piksel pada gambar, tanpa mengubah tampilan gambar secara mencolok. Dengan demikian, citra stego tetap dapat dikirim atau disimpan seperti gambar biasa, sementara informasi penting di dalamnya tetap terlindungi.

Meskipun metode LSB memiliki kelebihan dari segi kesederhanaan dan kualitas visual, metode ini juga memiliki kelemahan jika digunakan secara tunggal. Pesan yang disisipkan dengan LSB dasar berpotensi ditemukan apabila pihak luar mengetahui pola penyisipan yang digunakan. Oleh karena itu, penelitian ini tidak hanya menerapkan LSB secara sederhana, tetapi juga mengembangkan pendekatan hibrida dengan menggabungkan steganografi dan kriptografi. Melalui kriptografi, pesan rahasia terlebih dahulu dienkripsi sebelum disisipkan ke dalam citra digital. Dengan cara ini, apabila pesan berhasil diekstraksi oleh pihak yang tidak berwenang, isi pesan tetap tidak dapat dipahami karena telah diubah menjadi bentuk yang terenkripsi.

Penggabungan antara steganografi dan kriptografi memberikan perlindungan ganda terhadap informasi rahasia. Steganografi menyembunyikan keberadaan pesan, sedangkan kriptografi melindungi isi pesan. Kombinasi ini menjadi penting karena sistem pengamanan informasi yang baik tidak hanya bergantung pada satu mekanisme perlindungan. Dalam konteks militer, perlindungan berlapis dapat meningkatkan keamanan komunikasi, terutama ketika data yang dikirim memiliki nilai strategis. Pendekatan hibrida juga dapat mengurangi risiko penyalahgunaan informasi apabila media citra berhasil diakses oleh pihak lain.

Selain enkripsi, penelitian ini juga memanfaatkan fungsi chaos logistic map untuk meningkatkan keamanan proses penyisipan. Logistic map merupakan salah satu fungsi chaos yang menghasilkan pola bilangan yang sulit diprediksi. Dalam sistem steganografi, pola ini dapat digunakan untuk menentukan posisi penyisipan bit pesan secara lebih acak. Dengan demikian, pesan tidak selalu disisipkan secara berurutan pada piksel gambar, tetapi mengikuti pola tertentu yang hanya dapat diketahui oleh pihak yang memiliki parameter atau kunci yang sesuai. Penggunaan logistic map diharapkan dapat mempersulit proses deteksi dan ekstraksi oleh pihak eksternal.

Penerapan fungsi chaos dalam steganografi memberikan keuntungan karena karakteristiknya yang sensitif terhadap nilai awal. Perubahan kecil pada parameter awal dapat menghasilkan urutan yang berbeda secara signifikan. Hal ini dapat dimanfaatkan sebagai kunci tambahan dalam proses penyisipan dan ekstraksi pesan. Apabila pihak yang tidak berwenang tidak mengetahui nilai awal atau parameter yang digunakan, maka proses pengambilan pesan akan menjadi lebih sulit. Dengan demikian, logistic map dapat berperan sebagai mekanisme pengacakan yang memperkuat keamanan metode LSB.

Implementasi penelitian ini dilakukan menggunakan software simulasi agar setiap tahapan dapat diamati secara sistematis. Tahapan utama meliputi persiapan citra asli, persiapan pesan rahasia, proses enkripsi, proses penyisipan menggunakan LSB, pembentukan citra stego, ekstraksi pesan, dekripsi pesan, serta pengujian kualitas citra. Penggunaan software simulasi memungkinkan peneliti untuk menguji efektivitas sistem sebelum diterapkan pada lingkungan yang lebih luas. Selain itu, simulasi juga membantu dalam mengevaluasi apakah pesan dapat disisipkan dan diambil kembali secara akurat tanpa merusak citra digital.

Kualitas citra stego menjadi salah satu aspek penting dalam penelitian ini. Citra stego yang baik harus memiliki kualitas visual yang mendekati citra aslinya agar tidak menimbulkan kecurigaan. Salah satu ukuran yang umum digunakan untuk menilai kualitas citra adalah Peak Signal-to-Noise Ratio atau PSNR. Nilai PSNR yang tinggi menunjukkan bahwa perbedaan antara citra asli dan citra stego relatif kecil. Dengan demikian, semakin tinggi nilai PSNR, semakin baik kualitas citra hasil penyisipan. Selain PSNR, akurasi ekstraksi pesan juga menjadi indikator penting untuk memastikan bahwa data rahasia dapat dikembalikan secara utuh.

2. Metode Penelitian

Bagian ini menguraikan secara rinci perancangan sistem, prinsip kerja, serta aspek perangkat keras dan perangkat lunak yang terlibat dalam penelitian ini. Konsep dan teori dasar yang telah dibahas sebelumnya akan diimplementasikan di sini untuk mencapai tujuan penelitian. Fokus utama adalah pada desain yang tergambar dalam diagram blok sistem.

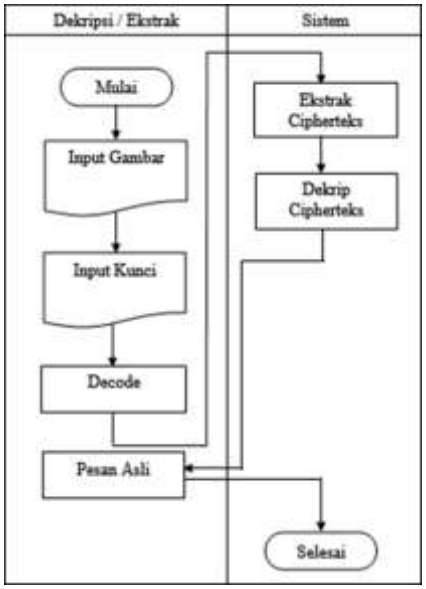
DOI: https://doi.org/10.31004/riggs.v5i2.8845

Implementasi Metode Steganografi

Untuk penerapan steganografi dalam domain militer, perpaduan antara keamanan yang ketat dan kapasitas yang memadai adalah esensial. Oleh karena itu, pendekatan hibrida yang mengombinasikan LSB dengan algoritma enkripsi (seperti AES dan Blowfish), serta potensi penggunaan peta logistik untuk pengacakan, dianggap sebagai pilihan yang paling optimal.

Algoritma Penyisipan Pesan Teks (Adaptasi LSB 1- bit)

Ini adalah adaptasi kode MATLAB dari dokumen PDS2_KLOMPOK 4_(LSB).docx yang digunakan untuk menyisipkan pesan teks ke dalam gambar.



Program Utama (Encoder)

```
% Encoder (Penyisipan Pesan Teks)
% Membersihkan workspace, menutup semua jendela gambar, dan membersihkan command window.
close all; clear; clc;
% Membaca gambar sampul. Penting untuk menggunakan format lossless seperti TIFF atau PNG
% untuk menjaga integritas data setelah penyisipan.
% Ganti 'tes1.tif' dengan jalur gambar yang relevan, misalnya 'peta_militer.tif'.
gambar = imread('tes1.tif');
gambarasli = gambar; % Menyimpan salinan gambar asli untuk tujuan perbandingan.
% Meminta pengguna untuk memasukkan teks rahasia yang akan disisipkan.
pesan = input('Masukkan teks yang hendak disisipkan = ', 's');
pesan = uint8(pesan); % Mengonversi setiap karakter pesan ke representasi unsigned 8-bit integer (nilai ASCII).
% Menyiapkan struktur pesan dengan menyematkan panjangnya di bagian awal.
[bp, kp] = size(pesan); % Mendapatkan dimensi (baris, kolom) dari pesan.
pesan2 = zeros(bp, kp + 1); % Membuat array baru dengan kolom tambahan.
pesan2(1, 1) = kp; % Menyimpan panjang asli pesan di elemen pertama array.
pesan2(1, 2:end) = pesan; % Menyalin seluruh pesan ke sisa elemen array.
pesan = pesan2; % Menggunakan array yang sudah diperbarui sebagai pesan utama.
```

DOI: https://doi.org/10.31004/riggs.v5i2.8845

```
[bp, kp] = size(pesan); % Memperbarui dimensi pesan.
% Menginisialisasi 'counter' untuk melacak posisi piksel terakhir yang disisipi.
% Ini membantu melanjutkan penyisipan dari lokasi yang tepat.
counter = [1 1];
% Melakukan iterasi untuk menyisipkan setiap karakter pesan ke dalam piksel gambar.
for i = 1:kp
% Mengubah karakter pesan dari format desimal (ASCII) ke representasi biner 8-bit.
hrfpsn = dec2bin(pesan(i), 8);
% Memanggil fungsi 'mskknkgbr' untuk melakukan proses penyisipan bit ke dalam gambar.
% Fungsi ini mengembalikan gambar yang telah dimodifikasi dan posisi 'counter' yang baru.
[gambar, counter] = mskknkgbr(gambar, counter, hrfpsn); end
% Menyimpan gambar hasil steganografi ke dalam file TIFF.
imwrite(gambar, 'gambarhsl.tif');
% Menampilkan gambar asli di jendela Figure 1 untuk referensi.
disp('Gambar yang asli Figure 1'); figure(1);
imshow(gambarasli);
% Menampilkan gambar yang telah disisipi pesan di jendela Figure 2.
disp('Gambar yang telah disisipi Figure 2'); figure(2);
imshow(gambar);
Fungsi mskknkgbr.m
% Fungsi mskknkgbr.m: Ini adalah fungsi pembantu yang dipanggil oleh skrip encoder.
% Fungsinya untuk menyisipkan bit-bit pesan ke dalam piks_el gambar.
function [gbr, counter] = mskknkgbr(gambar, counter, hrfpsn)
% gbr: Variabel output yang akan berisi gambar yang telah dimodifikasi.
% counter: Posisi piksel saat ini dalam gambar (format: [baris, kolom]).
% hrfpsn: String biner 8-karakter ('0' atau '1') yang mewakili satu karakter pesan.
% Mendapatkan dimensi gambar (jumlah baris, kolom, dan kanal warna).
[baris, kolom, rgb] = size(gambar);
% Asumsi: Penyisipan hanya dilakukan pada kanal Merah (Red) (kanal pertama).
% Jika diperlukan penyisipan pada semua kanal RGB, loop tambahan akan dibutuhkan di sini.
r = 1;
lokasi = counter; % Menyimpan lokasi awal 'counter' untuk referensi
% Memulai loop untuk mengambil 8 piksel dari gambar sampul.
% Setiap 8 piksel ini akan menampung 8 bit dari satu karakter pesan.
for i = 1:8
% Mengambil nilai piksel dari gambar pada posisi yang ditunjuk oleh 'counter'.
var(i) = gambar(counter(1), counter(2), r);
% Menggeser 'counter' ke kolom berikutnya untuk pemrosesan piksel selanjutnya.
```

DOI: https://doi.org/10.31004/riggs.v5i2.8845

counter(2) = counter(2) + 1;

% Logika untuk memindahkan 'counter' ke baris baru jika sudah mencapai akhir kolom.

if counter(2) > kolom

counter(2) = 1; % Kembali ke kolom pertama. counter(1) = counter(1) + 1; % Pindah ke baris

berikutnya. end

% Menyimpan koordinat piksel (baris, kolom) yang telah diambil.

lokasi(1, end + 1) = counter(1); lokasi(1, end + 1) = counter(2);

end

% Mengubah nilai-nilai piksel yang telah diambil dari desimal ke representasi biner 8-bit.

var1 = dec2bin(var(1), 8); var2 = dec2bin(var(2), 8); var3 = dec2bin(var(3), 8); var4 = dec2bin(var(4), 8); var5 = dec2bin(var(5), 8); var6 = dec2bin(var(6), 8); var7 = dec2bin(var(7), 8); var8 = dec2bin(var(8), 8);

% Proses utama penyisipan bit LSB (Least Significant Bit).

% Bit terakhir dari setiap variabel piksel (var1 hingga var8) diganti

% dengan bit yang sesuai dari string pesan biner 'hrfpsn'. var1(8) = hrfpsn(1);

var2(8) = hrfpsn(2); var3(8) = hrfpsn(3); var4(8) = hrfpsn(4); var5(8) = hrfpsn(5); var6(8) = hrfpsn(6); var7(8) = hrfpsn(7); var8(8) = hrfpsn(8);

% Mengubah kembali nilai biner yang sudah disisipi ke format desimal.

var1 = bin2dec(var1); var2 = bin2dec(var2); var3 = bin2dec(var3); var4 = bin2dec(var4); var5 = bin2dec(var5); var6 = bin2dec(var6); var7 = bin2dec(var7); var8 = bin2dec(var8);

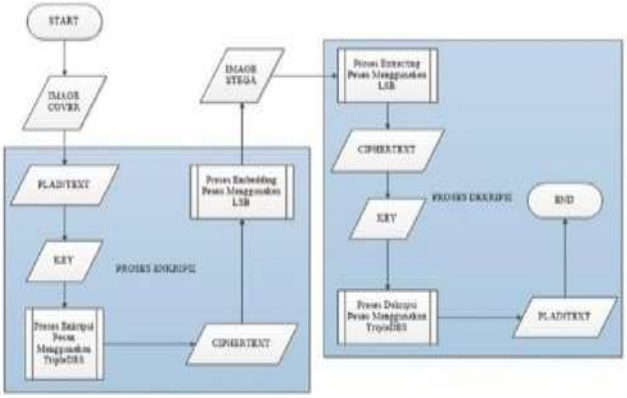
% Memasukkan kembali nilai-nilai piksel yang telah dimodifikasi

% ke lokasi aslinya dalam gambar. gambar(lokasi(1), lokasi(2), r) = var1; gambar(lokasi(3), lokasi(4), r) = var2; gambar(lokasi(5), lokasi(6), r) = var3; gambar(lokasi(7), lokasi(8), r) = var4; gambar(lokasi(9), lokasi(10), r) = var5; gambar(lokasi(11), lokasi(12), r) = var6; gambar(lokasi(13), lokasi(14), r) = var7; gambar(lokasi(15), lokasi(16), r) = var8;

gbr = gambar; % Mengembalikan gambar yang telah dimodifikasi.

End

Program Dekoder Pesan Teks



Gambar 1 Flowchart Proses Decoder LSB Teks

% Dekoder (Ekstraksi Pesan Teks)

% Membersihkan workspace, menutup semua jendela gambar, dan membersihkan command window.

close all; clear; clc;

DOI: https://doi.org/10.31004/riggs.v5i2.8845

```
% Membaca gambar yang diduga mengandung pesan tersembunyi.
gambar = imread('gambarhsl.tif');
[baris, kolom, rgb] = size(gambar); % Mendapatkan dimensi gambar.
% Menginisialisasi 'counter' untuk melacak posisi piksel. counter = [1 1];
r = 1; % Asumsi: pesan disisipkan pada kanal Merah (Red). lokasi = counter;
% Proses pengambilan 8 piksel pertama dari gambar.
% 8 piksel ini menyimpan informasi mengenai panjang pesan yang disisipkan.
for i = 1:8
var(i) = gambar(counter(1), counter(2), r); % Mengambil nilai piksel.
counter(2) = counter(2) + 1; % Menggeser ke kolom berikutnya.
if counter(2) > kolom % Jika sudah mencapai akhir kolom, pindah ke baris berikutnya.
counter(2) = 1;
counter(1) = counter(1) + 1; end
lokasi(1, end + 1) = counter(1); % Menyimpan lokasi piksel yang diambil.
lokasi(1, end + 1) = counter(2); end
% Mengubah nilai piksel yang telah diambil menjadi representasi biner.
var1 = dec2bin(var(1), 8); var2 = dec2bin(var(2), 8); var3 = dec2bin(var(3), 8); var4 = dec2bin(var(4), 8); var5 = dec2bin(var(5), 8); var6 = dec2bin(var(6), 8); var7 = dec2bin(var(7), 8); var8 = dec2bin(var(8), 8);
% Mengambil bit terakhir (LSB) dari setiap 8 piksel pertama.
% Bit-bit ini akan digabungkan untuk membentuk nilai panjang pesan.
pjgpsn = zeros(1, 8, 'uint8'); % Menginisialisasi array untuk menyimpan bit panjang pesan.
pjgpsn(1) = str2double(var1(8)); pjgpsn(2) = str2double(var2(8)); pjgpsn(3) = str2double(var3(8)); pjgpsn(4) = str2double(var4(8)); pjgpsn(5) = str2double(var5(8)); pjgpsn(6) = str2double(var6(8)); pjgpsn(7) = str2double(var7(8));
pjgpsn(8) = str2double(var8(8));
% Mengonversi array bit panjang pesan menjadi string biner, lalu ke nilai desimal.
pjgpsn_bin_str = char(pjgpsn + '0'); % Konversi ke string karakter '0'/'1'.
pjgpsn = bin2dec(pjgpsn_bin_str); % Konversi string biner ke nilai desimal (panjang pesan).
% Proses pengambilan teks yang sebenarnya dari gambar, berdasarkan panjang pesan yang sudah didapat.
psn = zeros(pjgpsn, 8, 'uint8'); % Menginisialisasi array untuk menyimpan bit-bit pesan yang diekstrak.
for k = 1:pjgpsn % Melakukan loop sebanyak panjang pesan yang ditemukan.
for i = 1:8 % Mengambil 8 piksel untuk setiap karakter pesan.
var(i) = gambar(counter(1), counter(2), r); % Mengambil nilai piksel.
counter(2) = counter(2) + 1; if counter(2) > kolom
counter(2) = 1;
counter(1) = counter(1) + 1; end
lokasi(1, end + 1) = counter(1); lokasi(1, end + 1) = counter(2);
end
% Mengubah nilai piksel yang diambil menjadi representasi biner.
```

DOI: https://doi.org/10.31004/riggs.v5i2.8845

var1 = dec2bin(var(1), 8); var2 = dec2bin(var(2), 8); var3 = dec2bin(var(3), 8); var4 = dec2bin(var(4), 8); var5 = dec2bin(var(5), 8); var6 = dec2bin(var(6), 8); var7 = dec2bin(var(7), 8); var8 = dec2bin(var(8), 8);

% Mengambil bit terakhir (LSB) dari setiap piksel untuk merekonstruksi karakter pesan.

psn(k, 1) = str2double(var1(8)); psn(k, 2) = str2double(var2(8)); psn(k, 3) = str2double(var3(8)); psn(k, 4) = str2double(var4(8)); psn(k, 5) = str2double(var5(8)); psn(k, 6) = str2double(var6(8)); psn(k, 7) = str2double(var7(8)); psn(k, 8) = str2double(var8(8));

end

% Mengonversi bit-bit pesan yang diekstrak kembali ke nilai desimal, lalu ke karakter Unicode.

psn_bin_str = char(psn + '0'); % Konversi ke string biner. psn_char = bin2dec(psn_bin_str); % Konversi biner ke desimal.

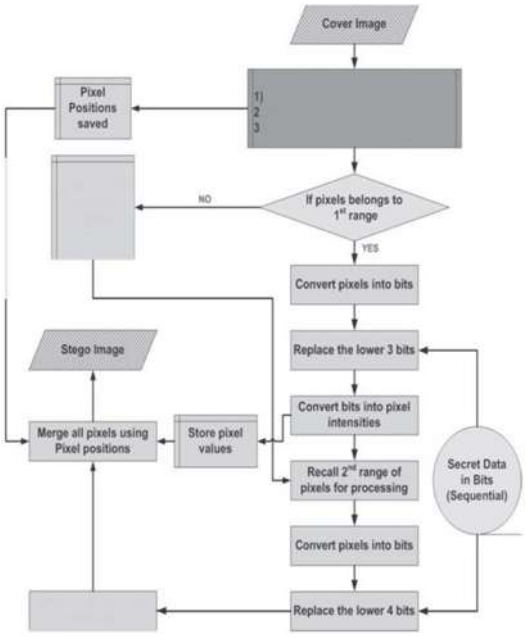
pesan_tersembunyi = native2unicode(psn_char); % Konversi desimal ke karakter Unicode.

disp('Pesan yang berhasil diekstraksi:'); disp(pesan_tersembunyi);

Algoritma Penyisipan Peta (Adaptasi LSB 4-bit)

Kode ini berfungsi untuk menyisipkan gambar (peta koordinat) ke dalam gambar sampul (peta militer) menggunakan 4 bit LSB dari setiap kanal warna RGB.

Encode peta



Gambar 2. Flowchart Proses Encode Peta LSB 4bit

% Encode (Penyisipan Peta dalam Peta)

% Membersihkan workspace, menutup semua jendela gambar, dan membersihkan command window.

close all; clear; clc;

% Memuat gambar sampul (cover image) dan gambar pesan (message image).

coverImage = imread('kalimantan.jpeg'); % Gambar yang akan digunakan sebagai penampung.

messageImage = imread('benuaa.jpg'); % Gambar rahasia yang akan disisipkan (peta koordinat).

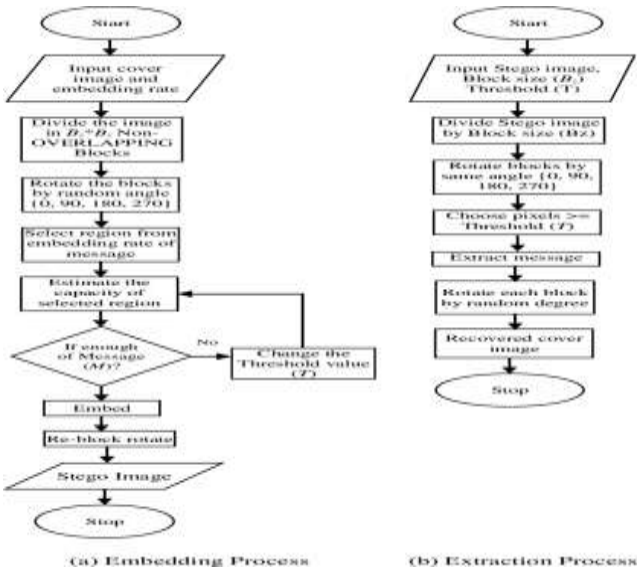
% Mengubah ukuran gambar pesan agar sesuai dengan dimensi gambar sampul.

% Ini penting untuk memastikan kompatibilitas dan kapasitas penyisipan.

DOI: https://doi.org/10.31004/riggs.v5i2.8845

```
messageImage = imresize(messageImage, [size(coverImage, 1), size(coverImage, 2)]);

% Menginisialisasi gambar yang akan disisipi. Awalnya, ini adalah salinan dari gambar sampul.

embeddedImage = coverImage;

% Melakukan iterasi untuk menyematkan 4 bit dari gambar pesan ke dalam gambar sampul

% untuk setiap kanal warna (Merah, Hijau, Biru). for channel = 1:3 % Loop untuk kanal R, G, B.

% Membersihkan 4 bit LSB dari 'coverImage' dan kemudian menyisipkan 4 bit MSB dari 'messageImage'.

% bitget(messageImage(:,:,channel), 5) mengambil bit ke- 5 dari kanan (LSB=1).

% bitset(..., 1, ...) menyisipkan bit ke posisi LSB (bit ke- 1).

% Baris-baris berikut menyisipkan bit ke-5 hingga ke-8 dari gambar pesan

% ke dalam bit ke-1 hingga ke-4 dari gambar sampul. embeddedImage(:,:,channel) =

bitset(coverImage(:,:,channel), 1, bitget(messageImage(:,:,channel), 5)); % Bit ke-1 LSB cover <-- Bit ke-5 message

embeddedImage(:,:,channel) = bitset(embeddedImage(:,:,channel), 2, bitget(messageImage(:,:,channel), 6)); % Bit ke-2 LSB cover <-- Bit ke-6 message

embeddedImage(:,:,channel) = bitset(embeddedImage(:,:,channel), 3, bitget(messageImage(:,:,channel), 7)); % Bit ke-3 LSB cover <-- Bit ke-7 message

embeddedImage(:,:,channel) = bitset(embeddedImage(:,:,channel), 4, bitget(messageImage(:,:,channel), 8)); % Bit ke-4 LSB cover <-- Bit ke-8 (MSB) message

end

% Menyimpan gambar hasil steganografi (embedded image) ke dalam file PNG.

imwrite(embeddedImage, 'embedded_image.png');

% Menampilkan gambar yang telah disisipi dan memberikan judul.

figure, imshow(embeddedImage), title('Embedded Image (4 bits LSB)');
```

Decode Peta



Gambar 3. Flowchart Proses Decode Peta LSB 4bit

DOI: https://doi.org/10.31004/riggs.v5i2.8845

```
% Decode (Ekstraksi Peta dari Peta)
% Membersihkan workspace, menutup semua jendela gambar, dan membersihkan command window.
close all; clear; clc;
% Memuat gambar yang telah disisipi (embedded image). embeddedImage = imread('embedded_image.png');
% Menginisialisasi matriks untuk menyimpan gambar pesan yang diekstrak.
% Ukurannya disesuaikan dengan gambar yang disisipi.
extractedImage = zeros(size(embeddedImage), 'uint8');

% Melakukan iterasi untuk mengekstrak 4 bit dari gambar yang disisipi
% untuk setiap kanal warna (Merah, Hijau, Biru). for channel = 1:3 % Loop untuk kanal R, G, B.
% Menggabungkan bit-bit yang diekstrak untuk merekonstruksi gambar pesan.
% bitget(embeddedImage(:,:,channel), 1) mengambil bit ke-1 (LSB).
% Bit-bit ini kemudian digeser ke posisi MSB yang sesuai (dikalikan dengan 16, 32, 64, 128)
% untuk membentuk kembali nilai piksel 8-bit. extractedImage(:,:,channel) =
uint8(bitget(embeddedImage(:,:,channel), 1) * 16 + ...
bitget(embeddedImage(:,:,channel), 2) * 32 +
...
... 128);
end
bitget(embeddedImage(:,:,channel), 3) * 64 +
bitget(embeddedImage(:,:,channel), 4) *
% Menyimpan gambar yang berhasil diekstrak ke dalam file PNG.
imwrite(extractedImage, 'extracted_image.png');
% Menampilkan gambar yang berhasil diekstrak dan memberikan judul.
figure, imshow(extractedImage), title('Extracted Image (4 bits LSB)');
```

3. Hasil dan Diskusi

Pengujian Steganografi Teks (LSB 1-bit)

Pada percobaan ini, kami menyelidiki proses LSB 1-bit dengan menyisipkan karakter "a" ke dalam matriks piksel gambar berukuran 3x3. Analisis ini memberikan gambaran jelas tentang bagaimana perubahan terjadi pada tingkat fundamental.

Rincian Langkah Analisis:

1. Konversi Matriks Gambar ke Biner: Gambar host (sampul) terlebih dahulu diubah ke dalam representasi biner. Ini adalah langkah awal untuk mempersiapkan data gambar agar bisa dimanipulasi pada tingkat bit.
2. Konversi Huruf 'a' ke Biner (ASCII): Karakter 'a' dikonversi menjadi nilai desimal ASCII-nya, yaitu 97, yang kemudian diubah ke dalam bentuk biner: 01100001. Ini adalah pesan rahasia yang akan disembunyikan

4. Kesimpulan

Penelitian ini berhasil menunjukkan aplikasi steganografi menggunakan metode Least Significant Bit (LSB) berbasis MATLAB untuk mengamankan informasi dan peta koordinat. Penyisipan Teks (1-bit LSB) Penyisipan Bit LSB: Bit-bit biner dari pesan (dalam hal ini, dari huruf 'a') disisipkan ke dalam Least Significant Bit (LSB) dari setiap piksel gambar host. Karena karakter 'a' terdiri dari 8 bit, minimal 8 piksel diperlukan untuk menampung seluruh pesan ini. Perubahan pada nilai piksel setelah menyisipan ternyata sangatlah minimal rata-rata hanya selisih 1 nilai desimal. Akibatnya, secara visual, perbedaan antara gambar asli dan gambar yang telah disisipi pesan hampir tidak dapat dibedakan, mengurangi kemungkinan kecurigaan. Proses Dekripsi: Untuk

DOI: https://doi.org/10.31004/riggs.v5i2.8845

mengungkapkan pesan tersembunyi, proses dekripsi dilakukan sebagai kebalikan dari enkripsi. Ini melibatkan konversi gambar stego kembali ke matriks biner, kemudian mengekstrak bit LSB dari setiap piksel. Bit-bit yang diekstrak ini kemudian dikelompokkan menjadi 8 bit untuk membentuk setiap karakter, dikonversi kembali ke nilai desimal, dan akhirnya diubah menjadi karakter ASCII yang dapat dibaca. Dalam upaya menyisipkan gambar peta ke dalam gambar peta lainnya, kami melakukan serangkaian percobaan iteratif hingga mencapai hasil yang optimal. Setiap iterasi memberikan pemahaman lebih lanjut tentang tantangan dan solusi dalam steganografi gambar berkapasitas tinggi. Pada awalnya, hasil ekstraksi gambar hanya menampilkan seperempat bagian kiri atas, dan gambarnya tidak sesuai dengan citra asli. Permasalahan ini muncul akibat kesalahan dalam penentuan dimensi gambar pesan; variabel baris_m dan kolom_m secara keliru membaca nilai dari piksel yang sama (carrier(1,1,1)). Kondisi ini membatasi dimensi gambar pesan yang diekstrak menjadi maksimum 255x255 piksel, jauh lebih kecil dari ukuran sebenarnya. Percobaan Kedua: Kami mengidentifikasi bahwa akar masalah pada percobaan sebelumnya adalah cara dimensi gambar pesan diambil dari dua byte pertama gambar carrier (carrier(1,1,1) dan carrier(1,1,2)). Meskipun ada sedikit perbaikan, hasilnya tetap terbatas karena nilai piksel uint8 memiliki batas maksimum 255. Namun, ada peningkatan signifikan pada kejernihan warna gambar yang diekstrak. Hal ini terjadi karena 2 bit LSB yang diekstrak digeser ke posisi Most Significant Bit (MSB), yang secara efektif meningkatkan intensitas warna dan membuat gambar terlihat lebih jelas. Percobaan Ketiga: Percobaan ketiga berhasil mencapai hasil yang optimal: gambar yang dihasilkan sempurna, mencakup seluruh bagian, dan sangat sesuai dengan citra asli. Keberhasilan ini disebabkan oleh penentuan dimensi gambar yang diekstrak secara tepat, di mana ukurannya disesuaikan dengan dimensi gambar carrier (embeddedImage), tidak lagi dibatasi oleh pembacaan *byte* tunggal. Lebih lanjut, proses ekstraksi 4 bit LSB dari setiap kanal RGB gambar carrier dan penggabungan kembali bit-bit ini untuk membentuk nilai intensitas 8-bit penuh berperan krusial. Pendekatan ini menghasilkan resolusi warna yang jauh lebih tinggi dan rekonstruksi gambar yang sangat akurat, mendekati kualitas aslinya. Metode ini sederhana dan efektif untuk menyembunyikan pesan teks. Meskipun kapasitas penyimpanannya rendah dan rentan terhadap modifikasi gambar, perubahan visual pada gambar sampul nyaris tidak terlihat. Untuk pengembangan, disarankan menggunakan lebih dari 1 bit LSB dan menambahkan mekanisme enkripsi. Penyisipan Peta (4-bit LSB): Metode ini menawarkan kapasitas penyimpanan yang lebih tinggi dan ketahanan

Referensi

1. Regyna, Tatya Fara, Dian Agustina, and FIRANIA NAZZILLA PRAMADISTA. 2022. "'SISTEM MANAJEMEN KEAMANAN INFORMASI.'" doi:10.31219/osf.io/t7keb.
2. Permana, Angga Aditya, and Habib Amna. 2022. "IMPLEMENTASI STEGANOGRAFI FILE CITRA DIGITAL MENGGUNAKAN METODE LEAST SIGNIFICANT BIT." Jurnal Teknik 11(1). doi:10.31000/jt.v11i1.6161.
3. Al Maki, Wikky Fawwaz, Indra Bayu Muktyas, Samsul Arifin, Suwarno, and Mohd Khairul Bazli Mohd Aziz. 2023. "Implementation of a Logistic Map to Calculate the Bits Required for Digital Image Steganography Using the Least Significant Bit (LSB) Method." Journal of Computer Science 19(6): 686–93. doi:10.3844/jcssp.2023.686.693.
4. Alanzy, May, Razan Alomrani, Bashayer Alqarni, and Saad Almutairi. 2023. "Image Steganography Using LSB and Hybrid Encryption Algorithms." Applied Sciences 13(21): 11771. doi:10.3390/app132111771
5. Kelompok 4 (Eliana Maharani, M. Miftah Farid, Rangga Taqwa, Ria Aprilianingsih). (2024). *LAPORAN PRAKTIKUM PENGOLAHAN SINYAL DASAR 2"Least Significant Bit
6. Laksono, A. W., Suhada, S., & Zakaria, A. (2024). Implementasi Metode Least Significant Bit (Lsb) Dalam Teknik Steganografi Pada Citra Digital Menggunakan Matlab. *Diffusion: Journal of Systems and Information Technology*, *4*(1).
7. Santiko, I. Implementasi Model Steganografi Dalam Mengelola Kerahasiaan Informasi Dengan Metode LSB (Least Significant Bit).
8. Nur'aini, S. (2019). Steganografi Pada Digital Image Menggunakan Metode Least Significant Bit Insertion. *Walisongo Journal of Information Technology*, *1*(1), 75-90.
9. TRIANA, F. (2020). *IMPLEMENTASI CAESAR CIPHER CRYPTOGRAPHY DAN LEAST SIGNIFICANT BIT-2 (LSB-2) STEGANOGRAPHY UNTUK KEAMANAN DATA BERBASIS ANDROID* (Doctoral dissertation, POLITEKNIK NEGERI SRIWIJAYA).
10. Insan, R. K. (2022). *Implementasi Steganografi Untuk Pengamanan Citra Digital Menggunakan Metode Bit-Plane Complexity Segmentation (BPCS)= Implementation of Steganography for Digital Image Security Using the Bit-Plane Complexity Segmentation (BPCS) Method* (Doctoral dissertation, Universitas Hasanuddin).
11. IBRAHIM, A. (2022). *MENGGABUNGKAN TEKNIK STEGANOGRAFI DISCRETE WAVELET TRANSFORM DUA DIMENSI (2-D) DAN ALGORITMA KRIPTOGRAFI RSA PADA PERANCANGAN DAN ANALISIS KEAMANAN PESAN* (Doctoral dissertation, UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU).
12. Marditya, R. D. (2016). *Implementasi Algoritma Rijndael dalam Teknik Steganografi Citra Digital Menggunakan Metode Least Significant Bit* (Doctoral dissertation, UII).

DOI: https://doi.org/10.31004/riggs.v5i2.8845

13. MARTOFORI, E. (2013). *PENGAMANAN PENGIRIMAN PESAN DALAM GAMBAR MENGGUNAKAN STEGANOGRAFI DENGAN DISCRETE COSINE TRANSFORM (DCT) DAN KRIPTOGRAFI RIVEST CODE 6 (RC6)* (Doctoral dissertation, UNIVERSITAS ISLAM NEGERI SULTAN SYARIEF KASIM RIAU).
14. Terapan, R. K. PROTOTIPE VALIDASI DAN PROTEKSI DATA PADA FILE IMAGE DENGAN MENGGUNAKAN ADVANCED ENCRYPTION STANDARD DAN LEAST SIGNIFICANT BIT BERBASIS ANDROID.
15. Anwar, N. (2018). Perancangan Steganografi Hidden Message Dengan Metode Least Significant Bit Insertion (Lsb) Berbasis Matlab. *Jurnal Algoritma, Logika Dan Komputasi*, *1*(1).

DOI: https://doi.org/10.31004/riggs.v5i2.8845