Jurnal Coding, Rekayasa Sistem Komputer ISSN 2338-493X
Volume 06, No. 03(2018), hal 75-86

ANALISIS KOMPRESI *STEGANOGRAPHY* PADA CITRA DIGITAL DENGAN MENGGUNAKAN METODE *LEAST SIGNIFICANT BIT* BERBASIS *MOBILE* ANDROID

[1]Muhammad Ricky, [2]Fatma Agus Setyaningsih, [3]Muhammad Dipenogoro.
[1] [2] [3]Jurusan Rekayasa Sistem Komputer, Fakultas MIPA Universitas Tanjungpura Jalan Prof. Dr. H. Hadari Nawawi, Pontianak
Telp./Fax.: (0561) 577963
e-mail :
[1]muhammadricky.sk@gmail.com [2]fatmasetyaningsih@siskom.ac.id
[3]muhammad.dipenogoro@siskom.untan.ac.id

ABSTRAK

Saat ini teknologi informasi berkembang dengan sangat pesat dan tingkat kejahatan dengan memanfaatkan teknologi informasi juga semakin tinggi. Pada masa sekarang ini penggunaan smartphone merajai telekomunikasi dunia, penyadapan kerap terjadi pada komunikasi berupa suara, maupun teks. Untuk menghindari terjadinya penyadapan maka berkirim pesan harus dilakukan dengan mempetimbangkan keamanan berkirim pesan, salah satu caranya adalah dengan melakukan teknik steganografi untuk dapat menyembunyikan pesan kedalam gambar agar tidak terbaca atau diketahui oleh pihak ketiga dalam hal ini adalah penyadap. Dalam penelitian ini teknik yang digunakan adalah teknik steganografi dengan menggunakan metode least significant bit. Dengan metode ini memungkinkan untuk menyembunyikan bit-bit pesan kedalam bit-bit akhir pada setiap pixel citra gambar. Untuk mengetahui efekstifitas dari metode yang digunakan maka dalam penelitian ini akan dilakukan analisa untuk mengetahui kekuatan encoding pesan kedalam gambar serta keberhasilan decoding pesan terhadap teks. Analisa yang dilakukan antara lain adalah: analisa rasio kompresi, faktor kompresi, mean square erorr dan peak signal to noise ratio. Dari analisa tersebut maka diketahui nilai mse gambar semakin bernilai besar apabila nilai bit yang ditambahkan semakin besar dan nilai psnr semakin kecil apabila nilai bit teks yang ditambahkan semakin besar.

Kata Kunci : citra, steganografi, *peak signal to noise ratio*.

1. PENDAHULUAN

Saat ini teknologi informasi berkembang dengan sangat pesat dan tingkat kejahatan dengan memanfaatkan teknologi informasi juga semakin tinggi. *User* juga membutuhkan keamanan dalam penggunaan teknologi ini

Untuk dapat berkirim pesan yang aman dari penyadapan maka dibutuhkan penerapan cryptography dan *steganography* untuk dapat menjaga keamanan pesan yang akan dikirim. Keamanan pesan dapat dijaga dengan proses enkripsi dan pesan dapat dibaca hanya dengan proses dekripsi. Pesan akan lebih aman jika pesan tersebut di*encode* dengan menggunakan metode steganografi kedalam sebuah media data, dalam penelitian ini penulis akan menggunakan *file* gambar sebagai media untuk menyisipkan pesan yang akan di *decode*. penelitian sejenis juga sudah pernah dilakukan dengan judul "Perbandingan Steganografi spread spectrum dan *least significant bit* (LSB) Antara Waktu Proses dan Ukuran *File* Gambar". Pada penelitian ini proses *encode* pesan dilakukan dengan menggunakan dua pilihan metode yaitu metode spread spectrum dan *least significant bit*. dari kedua metode tersebut akan dibandingkan hasil dari *encode-decode* pesan yang disisipkan dan dianalisa *file* gambar yang dijadikan *file* stegano untuk mengetahui derau atau noise serta menghitung ukuran *file* setelah dan sebelum di *encode*. Hasil dari penelitian ini berupa aplikasi desktop *encode* dan *decode* dengan menggunakan bahasa pemograman java dan memiliki dua metode yang diuji [2]Pada penelitian ini digunakan metode *least significant bit* (LSB), dimana bit LSB pada citra akan diganti dengan bit data dengan mengubah

satu byte lebih tinggi atau lebih rendah dari nilai sebelumnya. Untuk memperkuat teknik penyembunyian data, bit data rahasia tidak digunakan untuk mengganti byte yang berurutan namun dipilih secara acak. Selain itu proses *encoding* dengan menggunakan metode *least significant bit* cenderung lebih cepat dan mudah diimplementasikan.

Penulis akan membangun sebuah aplikasi yang dapat menyembunyikan pesan agar tidak dapat dibaca oleh orang yang tidak memiliki hak untuk membaca selain itu penulis juga akan menganalisa pengujian ketahanan *Peak signal to noise ratio* (PSNR). Selain analisa yang telah disebutkan penulis juga akan menganalisa faktor kompresi dan rasio kompresi pada gambar yang telah disisipkan pesan. Oleh karena itu penulis mengajukan penelitian dengan judul "Analisis Kompresi *Steganography* Pada Citra Digital dengan Menggunakan Metode *Least significant bit* Berbasis *Mobile* Android".

2. Tinjauan Pustaka

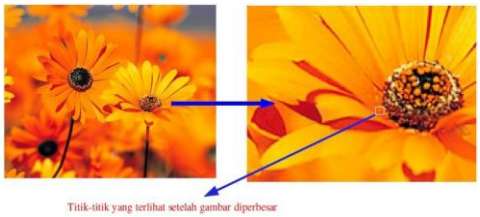
Penelitan ini menggunakan referensi-referensi mengenai teori, konsep dan metode yang digunakan.

2.1. Citra Digital

Citra adalah suatu representasi , kemiripan, atau imitasi dari suatu objek. Citra sebagai keluaran suatu sistem perekaman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal-sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu media penyimpanan. Citra digital adalah citra yang dapat diolah komputer. Beberapa format citra digital yang banyak ditemui adalah BMP, JPEG, GIF, PNG, dan lain-lain[3].

1. Bitmap

Gambar bitmap merupakan duplikat atau tiruan persis dari gambar asli dalam bentuk gambar digital. Gambar jenis ini tersusun dari sejumlah titik *pixel*/dot/point/titik koordinat yang ditempatkan pada lokasi-lokasi tertentu dengan nilai warna ter*send*iri sehingga membentuk pola tertentu di layar komputer. Pola yang terbentuk itulah yang menghasilkan atau menimbulkan kesan gambar. *Pixel* merupakan elemen terkecil citra digital yang dapat dilihat mata. Semakin banyak jumlah *pixel*, berarti semakin tinggi tingkat kerapatannya dan semakin halus gambar yang terbentuk. Akibatnya, semakin besar pula ukuran *file* gambar tersebut. Banyaknya titik dalam satu inchi dikenal dengan dpi (dot per inchi). Anda dapat mengenali gambar bitmap dari *file* komputer yang berekstensi .bmp, .jpg, .tfi, .gif, .png, .pix, .pcx, dan sebagainya.



Gambar 1. Contoh Gambar Bitmap

2. Vektor

Vektor adalah gambar yang tersusun oleh sekumpulan garis, kurva, dan bidang tertentu dengan menggunakan serangkaian instruksi yang masing-masing didefinisikan secara matematis. Setiap garis, kurva, dan bidang tertentu tersebut mempunyai property atau atribut masing-masing berupa *fill*, *stroke*, dan node. Gambar vektor tidak dipengaruhi oleh resolusi gambar atau titik *pixel* (dpi) seperti pada gambar bitmap.

Gambar vektor dapat dibuat dengan program aplikasi desain grafis untuk gambar vektor seperti *CorelDraw, Macromedia Freehand, Macromedia Flash, Dia*, dan *Inkscape*. Aplikasi-aplikasi tersebut masing-masing mempunyai kelebihan dan kekurangan. Dia dan Inkscape merupakan program aplikasi pengolah gambar vektor yang bersifat gratis, namun kemampuannya tidak kalah dengan aplikasi berbayar.

2.2 Steganografi

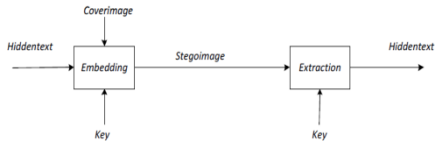
Steganografi merupakan seni untuk menyembunyikan pesan di dalam media digital sedemikian rupa sehingga orang lain tidak menyadari ada sesuatu pesan didalam media tersebut.

Steganografi menggunakan dua properti, yaitu wadah penampung dan data rahasia yang akan disembunyikan. Steganografi digital menggunakan media

digital sebagai wadah penampung, misalnya citra, audio, teks dan video[2].

Terdapat beberapa istilah yang berkaitan dengan steganografi:

1. *Hiddentext* atau *embedded message*: pesan yang disembunyikan.
2. *CoverImage*: citra yang digunakan untuk menyembunyikan embedded message.
3. *StegoImage*: citra yang sudah berisi embedded message.
4. *Stego*key: kunci rahasia.
5. *Embedding* : proses menyisipkan pesan pada citra sebagai medium penyisipan pesan.
6. *Extract* : mengambil pesan yang terdapat pada citra atau medium penyisipan.



Gambar 2. Diagram Penyisipan dan Ekstraksi pesan

Gambar 2 menunjukkan proses penyisipan *(embedding)* dan ekstrasi *(extraction)* pesan. Untuk menyisipkan pesan (hiddentext) dilakukan proses penyisipan *(embedding)* pada sebuah citra *(coverImage)* memerlukan kunci rahasia *(stego key)* untuk menghasilkan *stegoImage* atau pesan yang sudah disisipakan pesan rahasia. Untuk melakukan extraction dibutuhkan kunci rahasia untuk mengambil pesan yang terdapat pada *cover image*.

2.3 *Least significant bit*

Metode yang digunakan untuk menyembunyikan pesan pada media digital tersebut berbeda-beda. Contohnya, pada berkas *Image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data *pixel* yang menyusun *file* tersebut. Pada berkas bitmap 24 bit, setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna merah, hijau dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bit *(byte)* dari 0 sampai 255.

Kekurangannya dapat diambil kesimpulan dari contoh 8 bit *pixel*, menggunakan LSB Insertion dapat secara drastis mengubah unsur pokok warna dari *pixel*. Ini dapat menunjukkan perbedaan yang nyata dari *cover Image* menjadi *stego Image*, sehingga tanda tersebut menunjukkan keadaan dari steganografi. Variasi warna kurang jelas dengan 24 bit *Image*, bagaimanapun *file* tersebut sangatlah besar. Antara 8 bit dan 24 bit *Image* mudah diserang dalam pemrosesan *Image*, seperti *cropping* (kegagalan) dan *compression* (pemampatan).

Keuntungan yang paling besar dari algoritma LSB ini adalah cepat dan mudah. Dan juga algoritma tersebut memiliki *software* steganografi yang mendukung dengan bekerja di antara unsur pokok warna LSB melalui manipulasi *pallete*[4].

Metode *least significant bit*, digunakan citra digital sebagai media penyamar atau *cover-object*. Pada setiap *byte* dari *pixel* citra, terdapat bit yang paling kecil bobotnya (*Least significant bit* atau LSB)[3].

Metode ini bekerja dengan cara mengganti bit terakhir dari masing-masing *pixel* dengan pesan yang akan disisipkan. LSB mempunyai kelebihan yakni ukuran gambar tidak akan jauh berubah. Sedangkan kekurangannya adalah pesan/data yang akan disisipkan terbatas, sesuai dengan ukuran citra[3].

Sebagai contoh akan disisipkan A kedalam sebuah sebuah citra. Nilai A dalam ASCII adalah 65. Kemudian nilai ASCII A diubah menjadi biner yaitu 01000001. *Pixel* citra yang akan disisipkan teks yaitu 00000001, 00000110, 00000101, 00000011, 00000111, 00000100, 00000111, 00000100.

2.4 Analisa Kualitas Citra

Di dalam pembuatan sebuah aplikasi steganografi penyisipan pesan kedalam sebuah media, akan dilakukan beberapa pengujian dan analisa hasil. Informasi yang hilang akibat kompresi seharusnya seminimal mungkin sehingga kualitas hasil kompresi bagus. Tetapi biasanya kualitas kompresi bagus bila

proses kompresi menghasilkan pengurangan memori yang tidak begitu besar, demikian sebaliknya. Dalam kompresi citra terdapat standar pengukuran error (galat) kompresi yaitu:

1. Faktor Kompresi dan Rasio Kompresi

Faktor kompresi *(compression factor)* merupakan invers dari nilai rasio kompresi *(compression ratio)* yang menunjukkan persentase ukuran berkas hasil pemampatan dibandingkan ukuran berkas sebelum dimampatkan.

$$(\mathrm{FK}) = \frac{Ukuran\ File\ Output}{Ukuran\ File\ Input} \quad (1)$$

Rasio Kompresi berfungsi mengetahui jumlah rasio citra gambar setelah dilakukan penyisipan data, rasio kompresi akan didapatkan dari hasil bagi *file* setelah dikompresi dan *file* sebelum dikompresi. semakin besar nilai faktor kompresi dari suatu algoritma pemampatan data, maka algoritma tersebut berarti semakin baik. Pada umumnya nilai faktor kompresi lebih sering digunakan sebagai standard ukuran mengingat secara alamiah nilainya menunjukkan tingkat keandalan dari suatu algoritma (semakin besar nilai ≡ semakin bagus kualitas).

$$(\mathrm{RK}) = \frac{Ukuran\ File\ Input}{Ukuran\ File\ Output} \quad (2)$$

Dari persamaan tersebut terlihat, bahwa rasio kompresi akan selalu bernilai kurang dari 1, jadi jika suatu algoritma kompresi memiliki nilai rasio kompresi 0,5 maka algoritma ini mampu memampatkan berkas hingga menjadi separuh (50%) dari ukuran semula. Jadi semakin kecil nilai rasio kompresi maka semakin baik hasil citra tersebut.

Berkebalikan dengan nilai rasio kompresi adalah nilai faktor kompresi. Sehingga semakin besar nilai faktor kompresi berarti semakin baik kualitas citra tersebut. Pada umumnya nilai faktor kompresi lebih sering digunakan sebagai standard ukuran mengingat secara alamiah nilainya menunjukkan tingkat keandalan dari suatu algoritma. [4]

2. MSE *(Mean Square Error*)

Yaitu sigma dari jumlah *erorr* antara citra hasil kompresi dan citra asli.

$$MSE = \frac{1}{MN} + \sum_{x=1}^{M} \sum_{y=1}^{N} \left(S_{xy} - C_{Xy}\right)^2 \quad (3)$$

Di mana:

a. Cxy adalah nilai *pixel* di citra asli.
b. Sxy adalah nilai *pixel* pada citra hasil kompresi.
c. M, N adalah dimensi citra.

3. PSNR (*Peak signal to noise ratio*)

Yaitu untuk mengukur kualitas hasil kompresi. Nilai b merupakan nilai maksimum dari *pixel* citra yang digunakan. Nilai b pada penelitian ini adalah 255. Nilai MSE yang semakin rendah akan semakin baik, sedangkan semakin besar nilai PSNR, semakin bagus kualitas kompresi. PSNR memiliki satuan decibel (dB)[3].

$$PSNR = 10 \log_{10} \left(\frac{C_{max}^2}{MSE}\right) \quad (4)$$

3. METODOLOGI PENELITIAN

Penelitian ini berjudul "Analisis kompresi *steganography* pada citra digital dengan menggunakan metode *least significant bit* berbasis *mobile* android". Penelitian ini dibuat dengan mengacu pada referensi yang ada dengan dilakukan pengembangan-pengembangan pada *system* dan analisa hasil, dan untuk dapat melakukan penelitian dengan baik maka dibutuhkan alur penelitian yang jelas, untuk itu penulis telah membuat *flowchart* alur penelitian yang akan dilakukan. Berikut adalah *flowchart* yang alur penelitian yang akan dilakukan.

3.1 Studi Literatur

Penelitian ini membutuhkan teori-teori pendukung untuk membuat sistem dan melakukan analisa agar dapat direalisasikan. Teori pendukung didapatkan dari telaah buku, data dari berbagai referensi yang berkaitan dengan penelitian ini. Seperti teori tentang steganografi, metode *least significant bit*, analisa faktor kompresi, rasio faktor kompresi, analisa

mean square erorr dan analisa *peak signal to noise ratio*.

3.2 Metode Pengumpulan data

Pada sub bab ini penjelasan terkait pengumpulan data akan dijelaskan berdasarkan studi literatur yang telah digunakan .

3.3 Analisa Kebutuhan

Pada analisa kebutuhan terdapat dua tahap yang akan dilakukan diantaranya adalah analisa kebutuhan fusngsional dan analisa non-fungsinonal yang dapat dilihat pada sub bab berikutnya.

3.3.1 Analisa kebutuhan fungsional

Analisa kebutuhan fungsional merupakan gambaran mengenai fitur-fitur yang akan dimasukkan kedalam aplikasi yang akan dibuat berdasarkan masalah yang didapat. Fitur-fitur tersebut dapat dilihat pada Tabel 1.

Tabel 1..Analisis Kebutuhan Fungsional

No	Modul	Keterangan
1	*Encoding* dan *Decoding*	*Layout Encode Decode*
2	*Encoding* Pesan	*Layout Encoding* Pesan
3	*Decoding* Pesan	*Layout Decoding* Pesan
4	Analisa	*Layout* Analisa

3.3.2 Analisa kebutuhan non-fungsional

Analisa kebutuhan non-fungsional merupakan kebutuhan yang menunjang dalam pembuatan dan pelaksanaan aplikasi. Kebutuhan non-fungsional ini dapat meliputi kebutuhan non-fungsional perangkat lunak (*software*), perangkat keras (*hardware*) dan pengguna (*user*).

3.1 Kebutuhan Perangkat Lunak (*software*)

Berikut ini komponen perangkat lunak yang akan digunakan dalam pembangunan aplikasi:

a. Sistem operasi window 7.
b. Java JRE 8.1.
c. Android Studio 2.1.3

3.2 Kebutuhan Perangkat Keras (*hardware*)

Berikut ini beberapa spesifikasi komputer atau personal computer (PC) perangkat keras yang akan digunakan dalam pembangunan aplikasi:

a. Intel(R) Core(R) i3 CPU M350 @ 2.27.GHz.
b. Memory (RAM) 6GB dan hard disk drive (HDD) 300GB.

3.3 Kebutuhan Pengguna (*user*)

Selain kebutuhan perangkat lunak dan perangkat keras, Aplikasi ini juga membutuhkan pengguna (*user*) dalam menjalankan aplikasi *encoding* dan *decoding* pesan.

a. Pengguna dapat memililh gambar yang akan disisipkan pesan.
b. Pengguna dapat melakukan input pesan.
c. Pengguna dapat melakukan ekstraksi pesan yang tersisip didalam gambar.
d. Pengguna dapat melakukan pengiriman *file* gambar.

3.4 Rancangan penelitian

Perancangan penelitian dilakukan untuk dapat melakukan proses penelitian menjadi lebih sistematis dan terstruktur dengan baik.

3.4.1 Rancangan Sistem Kerja

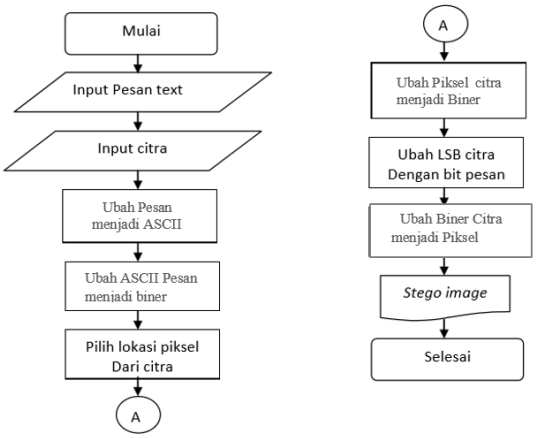
Perancangan dalam penelitian ini dimulai dengan Penentuan sistem kerja aplikasi. Sistem kerja ini meliputi kesuluruhan cara kerja aplikasi yang dibuat, yaitu menentukan apakah aplikasi dapat melakukan proses *encode decode* pesan kedalam gambar..Berikut adalah proses penyisipan pesan.

Proses embedding pada aplikasi steganografi ini adalah sebagai berikut:

1. Input pesan yang akan disisipkan.
2. Input citra yang akan disisipi pesan.
3. Jika ukuran teks lebih besar daripada ukuran citra, maka akan diminta untuk memasukan kembali citra yang lebih besar. Jika tidak, proses akan berlanjut ke langkah selanjutnya.
4. Ubah pesan menjadi kode-kode biner. Untuk mempermudah dapat terlebih dulu diubah menjadi ASCII, kemudian biner.
5. Pilih lokasi *pixel* dari citra.
6. Telah didapatkan masing-masing *pixel* citra.

7. Ubah *pixel* citra tersebut menjadi kode-kode biner.
8. Ganti bit terakhir kode biner citra dengan bit pesan.
9. Ubah biner citra yang telah mengandung bit pesan ke *pixel*.
10. Ubah *pixel* menjadi citra, maka akan diperoleh citra baru yang mengandung pesan (*stego-Image*).

Berikut adalah *flowchart* pada proses encode pesan, dapat dilihat pada Gambar 3.



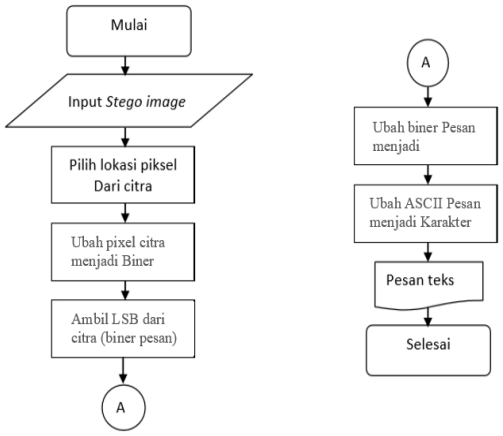
Gambar 3. Flowchart *Encode* Pesan

Dari hasil *encoding* pesan akan dihasilkan citra baru yang merupakan citra hasil *encoding* pesan yang disebut dengan *stego Image* yang sudah mengandung pesan yang jika diperhatikan dengan mata manusia tidak akan bisa ditemukan perbedaan. Hal ini yang menjadi kelebihan dari metode *Least significant bit*, dan jika ingin kembali mengambil pesan yang sudah disisipkan pada gambar maka diperlukan yang namanya ekstraksi pesan atau bisa disebut dengan *decoding* pesan

Berikut adalah langkah-langkah dalam *decoding* pesan dengan menggunakan algoritma *least significant bit*:

1. Masukan citra yang mengandung pesan (*stego-Image*)
2. Pilih lokasi *pixel* dari citra.
3. Ubah *pixel* citra ke dalam biner.
4. Ambil LSB citra sehingga menghasilkan biner pesan.
5. Ubah biner pesan ke dalam kode ASCII.
6. Ubah kode ASCII pesan menjadi karakter, sehingga menghasilkan pesan.

Berikut adalah *flowchart* pada proses *decode* pesan, dapat dilihat pada Gambar 4.



Gambar 4. *flowchart Decode* Pesan

4. IMPLEMENTASI PENGUJIAN

4.1 Implementasi

Bab ini berisi tentang pembahasan mengenai pengujian dan implementasi dari sistem aplikasi yang telah dibuat dengan menggunakan metode *least significant bit* pada penyisipan pesan pada citra digital.untuk mengimplementasikan aplikasi yang telah dibangun dibutuhkan suatu perangkat keras dan perangkat lunak agar aplikasi tersebut dapat dijalankann.berikut adalah spesifikasi perangkat keras dan perangkat lunak yang digunakan:

1. *Processor Intel Core i3*
2. *RAM 4GB*
3. *Harddisk 500GB*
4. Sistem Operasi Windows 10
5. Android SDK
6. Android NDK
7. JAVA JRE
8. Android Studio 3.0

Implementasi dari aplikasi ini terdiri dari empat bagian yaitu:

1. Proses untuk menampilkan halaman utama *(Main Activity)*.
2. Proses untuk menyisipkan teks kedalam citra (*encode*).

3. Proses untuk ekstraksi pesan yang telah berisi pesan (*decode*).
4. Proses untuk melakukan analisa kualitas gambar setelah dilakukan proses *encoding* pesan.

4.2 Tampilan Antarmuka Aplikasi

Implementasi aplikasi yang telah dibangun dapat dilihat pada gambar-gambar berikut:

4.2.1 Halaman Menu Utama Aplikasi

Berikut adalah halaman menu utama aplikasi ketika aplikasi dibuka.

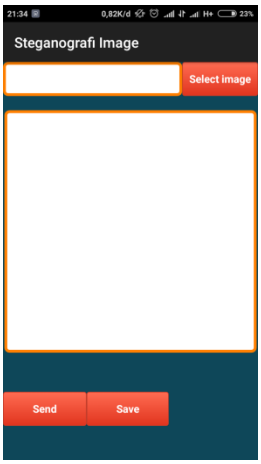
Gambar 5. Tampilan Menu Utama

Pada tampilan menu utama ini *user* dapat memilih akan melakukan proses yang diinginkan dengan menekan salah satu tombol, pada menu utama ini terdapat tiga tombol yang dapat digunakan yaitu tombol *decode Image*, *encode Image* dan Analisa.

4.2.2 Halaman *Encode* Pesan

Halaman *encode* pesan adalah halaman dimana segala proses terjadinya *encoding*/penyisipan pesan terjadi. Pada halaman ini terjadi tiga proses pada *button* diantaranya adalah Select *Image* , pada *button* ini *user* akan memilih gambar yang akan disisipkan pesan ketika menekan tombol tersebut, *user* akan diarahkan ke direktori penyimpanan gambar yang ada pada ponsel *user*, ketika gambar telah dipilih maka *user* akan memasukkan teks yang diinginkan kedalam *textbox* yang telah tersedia.

untuk lebih jelasnya dapat dilihat pada Gambar 6 dibawah ini.



Gambar 6. Tampilan *Encode Image*

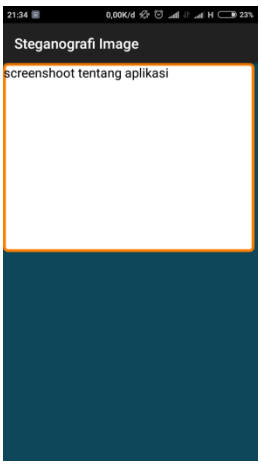
4.2.3 Halaman *Decode* Pesan

Pada halaman ini *user* akan mengekstraksi pesan yang telah disisipkan kedalam gambar, dan keluarannya berupa teks yang disisipkan sebelumnya.

Pada halaman ini *user* akan mengembalikan plainteks yang telah disisipkan kedalam gambar, dan keluarannya berupa teks utuh yang disisipkan sebelumnya.

Untuk lebih jelasnya dapat dilihat pada gambar *screenshoot* aplikasi yang telah dibuat.

Berikut adalah gambar tampilan *decode* pesan, dapat diihat pada Gambar 7.



Gambar 7. Halaman *Decode* Pesan

4.2.4 Halaman Analisa

Halaman ini merupakan halaman untuk melakukan proses analisa hasil dari encoding pesan kedalam gambar/citra.

Berikut adalah *layout* halaman analisa, dapat dilihat pada Gambar 8.

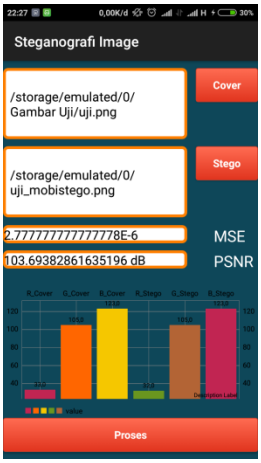
 ISSN 2338-493X

Gambar 8. Halaman Analisa PSNR dan MSE

Dan ketika aplikasi telah berhasil dijalankan maka akan terlihat nilai dari *mean square erorr* dan nilai dari *peak signal to noise ratio* gambar yang telah dianalisa.

Hasil dari analisa akan ditampilkan pada *textbox cover* dan *textbox stego.*

Berikut adalah *screenshoot* aplikasi setelah melakukan analisa *mean square erorr* dan *peak signal to noise ratio.* Dapat dilihat pada Gambar 9.



Gambar 9. Halaman Analisa saat Melakukan Analisa

5. Analisa Hasil

5.1 Analisa

Pada bab ini penulis telah melakukan analisa terhadap gambar hasil penyisipan teks dan gambar yang belum dilakukan penyisipan teks. Untuk mengetahui jumlah besaran *erorr* gambar terhadapa penyisipan yang dilakukan serta untuk mengetahui perbandingan serta perbedaan nilai gambar yang telah dilakukan proses *encode* pesan maka dibutuhkan analisa hasil gambar untuk mengetahuinya.

5.1.1 Analisa Faktor kompresi

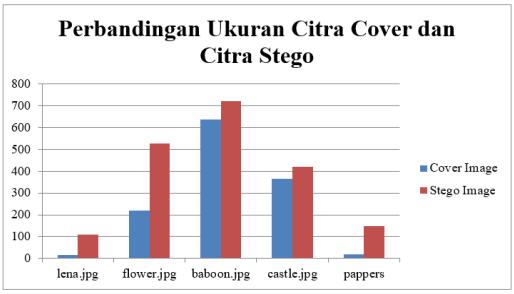
Pada analisa faktor kompresi merupakan perbandingan antara citra gambar yang telah dilakukan penyisipan pesan dan citra gambar yang belum dilakukan penyisipan pesan dengan rumus sebagai berikut pada persamaan (1).

Berikut ini adalah tabel nilai dari hasil pengujian untuk analisa faktor kompresi:

Tabel 2 Tabel Analisa Hasil Faktor Kompresi Gambar

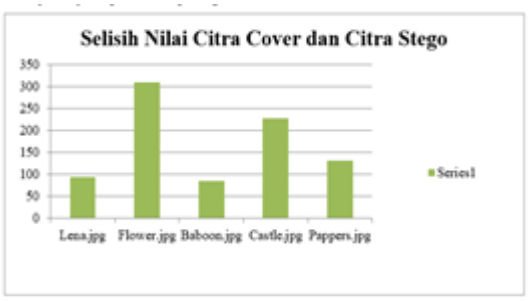
Nama *file* Citra	Ukuran Citra *Cover*	Ukuran Citra *Stego*	Ukuran *Pixel* Citra	Resolusi Gambar	Selisih	Nilai Faktor kompresi
Lena.jpg	15.6 Kb	108 Kb	256x256 px	72x72 dpi	92.2 Kb	6.9
Flower.jpg	213 Kb	515 Kb	512x512 px	72x72 dpi	302 Kb	2.4
Baboon.jpg	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	1.5
Castle.jpg	226 Kb	454 Kb	500x500 px	72x72 dpi	228 Kb	2.0
Pappers.jpg	16.6 Kb	146 Kb	298x298 px	72x72 dpi	129.4 Kb	8.7

Dari data hasil analisa di atas terlihat bahwa nilai faktor kompresi pada citra lena.jpg dan pappers.jpg memiliki nilai faktor kompresi yang paling tinggi, karena nilai citra *cover* yang kecil berubah jauh lebih besar ketika dilakukan penyisipan pesan, dan nilai pappers.jpg memiliki nilai faktor kompresi terbesar dengan nilai 8.7. Berikut adalah grafik dari hasil analisa pada Tabel 3.



Gambar 10. Grafik Perbandingan ukuran *file* Citra *Cover* dan Citra *Stego*

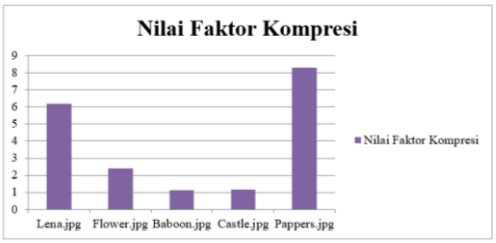
Dari grafik di atas ukuran *file* paling besar sebelum dilakukan penyisipan adalah citra baboon.jpg dan ukuran terkecil adalah lena.jpg begitu pula setelah dilakukan proses *encoding* pesan ukuran *file* yang paling besar adalah baboon.jpg dan yang paling kecil adalah lena.jpg.



Gambar 11. Grafik Selisih Ukuran Citra *Cover* dan Citra *Stego*

Pada grafik di atas dapat diperhatikan bahwa nilai selisih terbesar antara citra *cover* dan citra *stego* dimiliki oleh flower.jpg dengan selisih 302 Kb dan yang paling kecil dimiliki oleh lena.jpg dengan selisih 92.2 Kb.

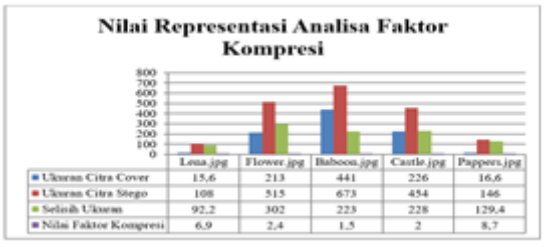
Dan dari perbandingan tersebut telah didapatkan nilai faktor kompresi dari setiap citra yang telah di analisa sebelumnya. Berikut adalah grafik dari hasil analisa faktor kompresi untuk nilai faktor kompresi.



Gambar 12. Grafik Perbandingan Nilai Faktor Kompresi Citra Uji

Dari grafik di atas terlihat nilai faktor kompresi terbesar dimiliki oleh citra pappers.jpg dengan nilai faktor kompresi 8.7 Kb dan nilai faktor kompresi yang terkecil dimiliki citra baboon.jpg 1.5 Kb.

Dari data pada Tabel 3 maka akan dipeoleh grafik dengan nilai setiap analisa yang dilakukan. Berikut adalah grafik nilai



Gambar 13. Grafik Nilai Representasi Analisa Faktor Kompresi

Berdasarkan hasil di atas terjadi perubahan yang signifikan terhadap nilai gambar citra asli dan citra *stego*, pada gambar dengan selisih rata-rata sebesar 527 kb.

Karena terjadi pemampatan atau penambahan bit pada setiap bit miskin, sehingga ukuran citra *stego* semakin besar. Penambahan/pemampatan bit dengan jumlah 400 bit pada setiap citra membuat citra *stego* menampung bit dengan jumlah yang lebih besar.

Faktor kompresi penyebab membesarnya jumlah citra *stego* karena telah mendapatkan pemampatan yang signifikan pada setiap citra *cover* yang diuji.

5.1.2 Analisa Rasio Kompresi

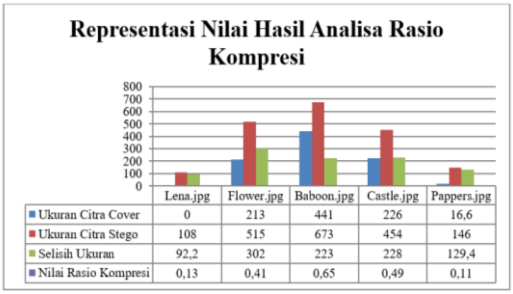
Analisa yang telah dilakukan berikutnya adalah analisa untuk menentukan rasio kompresi gambar yang dihasilkan setelah gambar disisipi pesan. berikut adalah rumus yang telah dijelaskan pada dasar teori yaitu pada persamaan (2)

Berikut ini adalah tabel nilai dari hasil pengujian untuk analisa rasio kompresi:

Tabel 3 Tabel Analisa Hasil Rasio Kompresi Gambar

Nama *file* Citra	Citra *Cover*	Citra *Stego*	Ukuran *Pixel* Citra	Resolusi Gambar	Selisih	Rasio kompresi
Lena.jpg	15.6 Kb	108 Kb	256x256 px	72x72 dpi	92.2 Kb	0.13
Flower.jpg	213 Kb	515 Kb	512x512 px	72x72 dpi	302 Kb	0.41
Baboon.jpg	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.65
Castle.jpg	226 Kb	454 Kb	500x500 px	72x72 dpi	228 Kb	0.49
Pappers.jpg	16.6 Kb	146 Kb	298x298 px	72x72 dpi	129.4 Kb	0.11

Pada grafik dibawah ini akan lebih jelas nampak nilai dari analisa rasio kompresi.



Gambar 13. Grafik Representasi Nilai Hasil Analisa Rasio Kompresi

Sama halnya dengan analisa faktor kompresi pada analisa rasio kompresi yang diperhatikan adalah nilai rasio kompresi gambar yang terjadi setelah gambar/citra dimampatkan teks berjumlah 400 bit maka nilai rasio kompresi terhadap masing-masing gambar akan berbeda karena adanya pemampatan teks sehingga nilai gambar atau citra *stego* akan memiliki jumlah nilai yang lebih besar dengan jumlah rasio pemampatan yang berbeda-beda pula dari masing-masing citra.

5.1.3 Analisa *Mean square erorr* (MSE)

MSE adalah nilai *erorr* kuadrat rata-rata antara citra asli dan citra hasil sisipan . nilai ini berguna untuk melakukan analisa perbandingan kualitas citra sebelum dan sesudah dilakukan penyisipan pesan dengan mengukur besarnya derau yang ditimbulkan.

Untuk mendapatkan nilai MSE diperlukan pengujian nilai hasil dari pengolahan citra *cover* dan citra *stego* dengan menggunakan rumus pada persamaan (3).

Karena jumlah *pixel* gambar yang sangat besar perhitungan pengujian untuk melakukan analisa dari MSE ini tidak bisa dilakukan dengan cara manual, untuk itu penulis melakukan analisa dengan menggunakan Aplikasi yang telah dibuat. Untuk melakukan pengujian maka akan dilakukan pengambilan data uji, untuk melakukan pengujian ini citra yang akan digunakan adalah citra baboon.jpg dengan ukuran citra 512x512 dan resolusi gambar sebesar 72x72 dpi.

Berikut adalah tabel dari hasil pengujian nilai *mean square erorr*:

Tabel 4. Tabel Pengujian Citra Baboon.jpg

Nama *file* Citra	Ukuran Citra *Cover*	Ukuran Citra *Stego*	Ukuran *Pixel* Citra	Resolusi Gambar	Selisih	Nilai *mean square erorr*
Baboon.jpg (1)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.00878
Baboon.jpg (2)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.01098
Baboon.jpg (3)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.02134
Baboon.jpg (4)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.03955

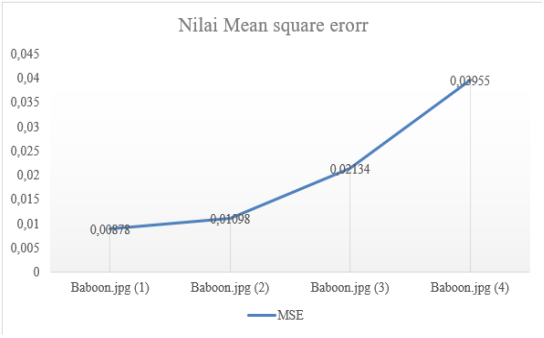
Dari hasil data di atas untuk lebih jelas agar dapat membandingkan dapat dilihat pada grafik representasi nilai *mean square erorr*.

Pada halaman ini *user* akan mengekstraksi pesan yang telah disisipkan kedalam gambar, dan keluarannya berupa teks yang disisipkan sebelumnya.

Berikut adalah grafik hasil pengujian dari Tabel 4, diperoleh grafik nilai *MSE* yang dihasilkan dari pengujian perhitungan *MSE* yang dilakukan.

Dari nilai tabel di atas dapat dilihat nilai represenasi pembentuk nilai MSE dan pada grafik dibawah ini dapat dilihat nilai dari MSE berdasarkan masing-masing gambar yang disisipkan teks dengan jumlah bit yang berbeda.

Pada halaman ini *user* akan mengekstraksi pesan yang telah disisipkan kedalam gambar, dan keluarannya berupa teks yang disisipkan sebelumnya.



Gambar 14. Grafik Nilai *MSE*

Pada grafik di atas dapat disimpulkan bahwa nilai bit yang ditambahkan pada penyisipan pesan dapat mempengaruhi nilai citra *stego* dan nilai MSE yang dihasilkan. Terlihat dari grafik di atas bahwa nilai semakin bertambah seiring tingginya nilai bit yang disisipkan. Semakin rendah nilai MSE yang dihasilkan maka akan semakin baik nilai citra yang penyisipan dihasilkan.

5.1.4 Analisa *Peak signal to noise ratio* (PSNR)

Analisa ini didapat dari hasil analisa MSE sebelumnya, dari hasil PSNR akan diketahui besaran nilai *erorr* gambar setelah dilakukan penyisipan pesan.

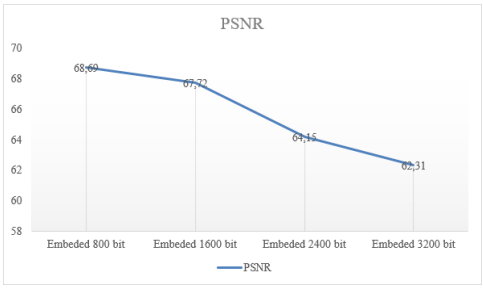
Hasil analisa PSNR dapat dilihat pada tabel dibawah ini.

Tabel 5. Tabel Nilai Representasi *Peak signal to noise ratio*

Nama *file* Citra	Ukuran Citra *Cover*	Ukuran Citra *Stego*	Ukuran *Pixel* Citra	Resolusi Gambar	Selisih	Nilai *mean square erorr*	Nilai *peak signal to noise ratio*
Baboon.jpg (1)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.00878	68.69 dB
Baboon.jpg (2)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.01098	67.72 dB
Baboon.jpg (3)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.02134	64.15 dB
Baboon.jpg (4)	441 Kb	673 Kb	512x512 px	72x72 dpi	223 Kb	0.03955	62.31 dB

Dari nilai yang telah dianalisa hasil nilai PSNR berada pada rentang yang baik. Seperti pada parameter quality hasil kompresi JPEG yang baik.

Berikut adalah nilai grafik PSNR berdasarkan Tabel 5



Gambar 15. Grafik Nilai PSNR

Pada grafik terjadi penurunan nilai setiap penambahan bit pada proses penyisipan teks. Nilai PSNR berada pada kisaran 60-63 dB. Nilai PSNR gambar berada pada rentang nilai yang baik.

Tabel 5.5 Tabel Nilai PSNR

PSNR(dB)	Picture Quality
60	Excellent, no noise apparent
50	Good, a small amount of noise but picture quality good
40	Reasonable, fine grain or snow in the picture, some fine detail lost
30	Poor picture with a great deal of noise
20	Unusable

Rentang pada nilai 30-60 adalah rentang nilai yang baik untuk sebuah gambar yang telah dilakukan penyisipan pesan sementara pada hasil pengujian nilai yang dihasilkan berada pada rentang 40-50 sehingga dapat dikatakan nilai yang dihasilkan pada gambar masih berada pada kualitas gambar yang baik.

6. KESIMPULAN DAN SARAN

6.1 Kesimpulan

Berdasarkan pada hasil pengujian, pembahasan serta analisa yang telah dilakukan maka didapatkan kesimpulan sebagai berikut.

1. Penyisipan pesan kedalam gambar dapat diimplementasikan dengan menggunakan metode *least significant bit*.
2. MSE digunakan untuk mengetahui *erorr* dari metode yang digunakan, semakin kecil nilai MSE maka makin kecil pula *erorr* yang terjadi pada metode yang digunakan, dan pada pengujian yang dilakukan nilai MSE yang dihasilkan dominan kecil.
3. Penyisipan gambar dengan menggunakan metode least significant pada aplikasi Android ini memenuhi kriteria steganografi yang baik. Yakni memiliki *imperceptibility* (keberadaan citra embeded di dalam citra *stego* tidak dapat dideteksi secara kasat mata), *fidelity* (mutu citra tidak berubah walaupun setelah mengalami proses penyisipan) dan memiliki *recovery* (citra *stego* dapat diekstrak kembali).

6.2 Saran

Berikut adalah saran untuk penelitian berikutnya.

1. Penelitian ini dapat dikembangkan dengan menambahkan input password sebagai penguat pada keamanan serta memberikan noise tambahan untuk dilakukan analisa ketahanan pada citra gambar terhadap serangan noise pada gambar.
2. Penelitian ini dapat dikembangkan dengan mengunakan media lain sebagai media penyimpanan pesan teks seperti audio dan video.

DAFTAR PUSTAKA

[1] Aditya, Y., Pratama, A., Nurlifa, Alfian. 2010. Studi Pustaka Untuk Steganografi Dengan Beberapa Metode. Seminar Nasional Aplikasi Teknologi Informasi 2010 (SNATI 2010) ISSN: 1907-5022, Yogyakarta, 19 Juni 2010.

[2] Munir, R. 2004. Pengolahan Citra Digital dengan Pendekatan Algoritmik. Bandung: Penerbit Informatika.

[3] Munir, R. 2004. Steganografi dan Watermarking. Bahan Kuliah ke-7 IF5054 Kriptografi. Departemen

Teknik Informatika Institut Teknologi Bandung 2004..

[4] Pakereng, I. 2010. Perbandingan Steganografi Metode Spread Spectrum dan *Least significant bit* (LSB) Antar Waktu Proses dan Ukuran *File* Gambar. Jurnal Informatika Fakultas Teknik Informatika Universitas Kristen Duta Wacana Yogyakarta. Vol.6 No.1 April 2010.

[5] Prasetyo, E. 2011. Pengolahan Citra Citra Digital dan Aplikasinya Menggunakan Matlab. Yogyakarta. Penerbit: ANDI.

[6] Prihanto, A., Djanali, S., Husni, Muchammad. 2010. Peningkatan Kapasitas Informasi Tersembunyi pada *Image* Steganografi Menggunakan Teknik Hybrid. Seminar Nasional Pascasarjana X – ITS, Surabaya 4 Agustus 2010 ISBN No. 979-545-0270-1.

[7] Putra, D. 2010. Pengolahan Citra Citra Digital. Yogyakarta. Penerbit: ANDI.

[8] Suryani, E., Martini, Titin Sri. 2008. Kombinasi Kriptografi Dengan Hillcipher Dan Steganografi Dengan LSB Untuk Keamanan Data Teks. Prosiding Seminar Nasional Teknoin 2008 Bidang Teknik Informatika

[9] Sutoyo, T. 2009. Teori Pengolahan citra digital. Yogyakarta. Penerbit ANDI.

[10] Zaher, Mazen Abu. 2011. Modified *Least significant bit* (MLSB). Jurnal Computer and Information Science Vol. 4, No. 1; January 2011. www.ccsenet.org/cis.

[11] *Steganography*, "LSB *Steganography* Using Java", *Steganography Image* Using LSB Methode, https://github.com/lsb-*steganography*-java/ (24 Juli 2015)