TUGAS AKHIR – KI14502

PENERAPAN STEGANOGRAFI DENGAN ALGORITMA RDE PADA FILE CITRA DENGAN MEMANFAATKAN INTEREST POINT MENGGUNAKAN ALGORITMA SURF

RIPAS FILQADAR
NRP 5112100020

Dosen Pembimbing I
Tohari Ahmad, S.Kom., MIT., Ph.D.

Dosen Pembimbing II
Hudan Studiawan, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESES – KI1502

IMPLEMENTATION OF STEGANOGRAPHY RDE ON IMAGE FILE BASED ON INTEREST POINT IN SURF ALGORITHM

RIPAS FILQADAR
NRP 5112100020

Supervisor I
Tohari Ahmad, S.Kom., MIT., Ph.D.

Supervisor II
Hudan Studiawan, S.Kom., M.Kom.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2016

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

Penerapan Steganografi dengan Algoritma RDE pada *File* Citra dengan Memanfaatkan *Interest Point* Menggunakan Algoritma SURF

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Komputasi Berbasis Jaringan
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
RIPAS FILQADAR
NRP : 5112 100 020

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Tohari Ahmad, S.Kom., MIT, Ph.D. .....
   NIP: 19750525 200312 1 003 (Pembimbing 1)

2. Hudan Studiawan, S.Kom., M.Kom. .....
   NIP: 19870511 201212 1 003 (Pembimbing 2)

SURABAYA
JUNI, 2016

[Halaman ini sengaja dikosongkan]

Penerapan Steganografi dengan Algoritma RDE pada *File* Citra dengan Memanfaatkan *Interest Point* Menggunakan Algoritma SURF

Nama Mahasiswa : Ripas Filqadar
NRP : 5112100020
Jurusan : Teknik Informatika FTIF-ITS
Dosen Pembimbing 1 : Tohari Ahmad, S.Kom., MIT., Ph.D.
Dosen Pembimbing 2 : Hudan Studiawan, S.Kom., M.Kom.

Abstrak

Media penyimpanan data menggunakan steganografi masih sangat rentan terhadap perubahan. Ketika media penyimpanan data dilakukan perubahan, maka data yang sudah disimpan akan sulit untuk diektstraksi kembali. Hal ini dikarenakan kurangnya fleksibilitas terhadap media penyimpanan data. Penggunaan algoritma RDE pada steganografi, terdapat peta lokasi yang menunjukkan titik dari nilai piksel *yang dirubah. Peta lokasi ini menyimpan data berupa titik yang pikselnya dirubah serta status dari* piksel *tersebut. Karena data titik yang akan digunakan untuk menyimpan bit data telah ditentukan diawal, sehingga media menjadi rentan terhadap perubahan dari pengguna.*

Tugas akhir ini dibuat untuk meningkatkan ketahanan media penyimpanan steganografi dengan memanfaatkan interest point pada gambar keabuan sebagai media penyimpanan data dan menghilangkan data titik penyimpanan pada peta lokasi untuk mengurangi ukuran peta lokasi. Pendeteksian interest point menggunakan algoritma SURF yang sering digunakan untuk pencocokan gambar. Interest point yang terdeteksi akan digunakan sebagai titik-titik untuk penyimpanan data.

Hasil dari uji coba yang dilakukan menunjukkan bahwa interest point pada sebuah citra dapat digunakan

sebagai tempat untuk penyisipan data menggunakan steganografi dengan algoritma RDE sehingga bisa memperkuat ketahanan dari citra hasil steganografi.

Kata Kunci: Steganografi, Interest Point, SURF, Peta Lokasi

Implementation Of RDE On Image Using *Interest Point* In SURF Algorithm

Student's Name	**: Ripas Filqadar**
Student's ID	**: 5112100020**
Department	**: Teknik Informatika FTIF-ITS**
Supervisor 1	**: Tohari Ahmad, S.Kom., MIT., Ph.D.**
Supervisor 2	**: Hudan Studiawan, S.Kom., M.Kom.**

Abstract

The cover of steganography still vulnerable to change. When cover get action which change the cover like rotation, data that has been store will be difficult to extract. This is due to a lack of flexibility in cover. The use of RDE algorithm, there are location map which showing the locaton of point which the piksel *values are changed. Location map is storing data from a point which the* piksel *values are changed and status from it. Because point to be used to store bit data have been set, so the media be susceptible to changes of the pengguna.*

This Undergraduate Theses is designed to improve resilience of steganography by using interest point on keabuan *image as a cover and remove data point in location map to reduce the size of location map. Interest point detection using SURF algorithm which usually used for matching or compare similarity of two image or more. Interest point are detected will be used to embed data using RDE.*

Based on trials, interest point on an image can be used to a place for strore data using steganography with RDE algorithm. This can make the citra from steganography result more robust.

Keywords: Steganography, Interest Point, SURF, Location Map

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT, yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul **"Penerapan Steganografi dengan Algoritma RDE pada *File* Citra dengan Memanfaatkan *Interest Point* Menggunakan Algoritma SURF` ".** Dengan pengerjaan Tugas Akhir ini, penulis bisa belajar lebih banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan penulis selama menempuh perkuliahan di Teknik Informatika ITS. Dengan Tugas Akhir ini penulis juga dapat menghasilkan suatu implementasi dari apa yang telah penulis pelajari.

Selesainya Tugas Akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Keluarga penulis yang telah memberikan dukungan moral dan material serta do'a yang tak terhingga untuk penulis. Serta selalu memberikan semangat dan motivasi pada penulis dalam mengerjakan Tugas Akhir ini.
3. Bapak Tohari Ahmad, S.Kom.,MIT., Ph.D. dan Bapak Hudan Studiawan, S.Kom., M.Kom selaku dosen pembimbing penulis yang telah memberikan bimbingan dalam pengerjaan Tugas Akhir ini.
4. Bapak Arya Yudhi Wijaya, S.Kom.,M.Kom. selaku dosen wali penulis.
5. Bapak Dr.Eng Darlis Herumurti, S.Kom.,M.Kom. selaku Kepala Jurusan Teknik Informatika ITS, Bapak Dr. Radityo Anggoro , S.Kom.,M.Sc. selaku koordinator Tugas Akhir dan segenap dosen Teknik Informatika yang telah memberikan ilmunya.

6. Administrator Laboratorium Komputasi Berbasis Jaringan yang telah membantu dan berbagi ilmu serta selalu memberikan semangat kepada penulis dalam pengerjaan tugas akhir.
7. Teman-teman kontrakan, Ridho, Mumun, Dendi, Fikri, Anggi dan Yadi yang selalu menyemangati penulis.
8. Teman-teman *user* Tugas Akhir Laboratorium Komputasi Berbasis Jaringan yang selalu menemani dan mengerjakan Tugas Akhir bersama-sama.
9. Keluarga Departemen TAD Paguyuban KSE yang memberikan semangat dan motivasi penulis dalam mengerjakan Tugas Akhir ini.
10. Teman-teman angkatan 2012 yang sudah menemani, membagi ilmunya dan membantu penulis dalam masa perkuliahan hingga bisa menyelesaikan Tugas Akhir ini.
11. Teman-teman seperjuangan 2012 Admiral yang selalu menyemangati penulis.
12. Serta seluruh pihak-pihak lain yang belum sempat disebutkan satu per satu di sini yang sudah membantu dalam terselesaikannya Tugas Akhir ini,

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depannya.

Surabaya, Juni 2016

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

DAFTAR *PSEUDOCODE*

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1 Latar Belakang

Keamanan data adalah salah satu hal yang sangat penting dalam komunikasi berbasis jaringan. Seiring semakin dengan kemajuan teknologi yang semakin pesat, maka kebutuhan akan keamanan harus mengalami peningkatan. Ini sangat penting agar informasi yang dikirimkan tidak bisa digunakan oleh pihak-pihak yang tidak bertanggung jawab.

Berbagai macam teknik dan teknologi yang digunakan untuk melakukan pengamanan informasi dari orang yang tidak berhak. Salah satu teknik yang digunakan adalah steganografi. Steganografi berasal dari bahasa yunani yaitu "*steganos*" yang berarti "tersembunyi atau terselubung", dan "*graphein*" yang berarti "menulis". Steganografi adalah ilmu menyembunyikan informasi dengan suatu cara sehingga selain pengirim dan penerima tidak ada yang mengetahui dan menyadari bahwa ada informasi rahasia [1].

Salah satu media yang digunakan sebagai penyimpana data dalam steganografi adalah *file* citra. Penyimpanan data dilakukan dengan melakukan modifikasi pada nilai piksel dari *file* citra tersebut. Namun *file* citra yang sudah dimasuki data tersebut sangat rentan terhadap perubahan yang dilakukan oleh pengguna, sehingga data yang disimpan menjadi tidak bisa diekstrak kembali. Diperlukan metode baru untuk memilih titik-titik yang akan digunakan untuk menyimpan data, sehingga citra yang digunakan untuk menyimpan data memiliki toleransi terhadap perubahan minor yang terjadi.

Dalam pemrosesan citra, banyak metode yang digunakan dalam pencocokan gambar. Salah satu algoritma yang sering digunakan adalah SURF(*Speed Up Robustness Feature*) [2]. Algoritma SURF akan mendeteksi titik-titik yang sama yang disebut *interest point* walaupun dilakukan perubahan minor terhadap gambar.

Dalam tugas akhir ini, penulis mengusulkan sebuah teknik steganografi yang memiliki toleransi ketahanan terhadap perubahan gambar. Keadaan yang dapat di toleransi adalah rotasi, pencerminan

serta perubahan minor pada *file* yang menyimpan informasi. Dengan memanfaatkan algoritma SURF untuk mendeteksi titik-titik yang nantinya akan digunakan sebagai media untuk penyimpanan data., diharapkan permasalahan tersebut bisa diselesaikan.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana mendapatkan *interest point* yang akan digunakan untuk menyimpan data?
2. Bagaimana menerapkan teknik steganografi RDE menggunakan titik-titik yang didapat dari algoritma SURF?
3. Bagaimana mengurangi dampak modifikasi piksel karena steganografi RDE terhadap titik-titik dari *interest point* yang digunakan?

1.3 Batasan Masalah

Batasan dalam Tugas Akhir ini, yaitu:

1. Pesan yang disimpan adalah *file* multimedia, untuk *file* citra dikhususkan *file* dengan ekstensi .pgm.
2. Citra yang digunakan sebagai media penyimpanan merupakan citra dengan tipe warna keabuan.
3. Perubahan minor yang bisa ditolerir oleh citra hasil steganografi adalah rotasi, pencerminan dan pengaburan citra.

1.4 Tujuan

Tujuan dalam pembuatan tugas akhir ini adalah sebagai berikut:

1. Meningkatkan ketahanan *file* citra yang digunakan sebagai media penyimpanan data terhadap perubahan minoritas sesuai dengan yang diusulkan.
2. Membuat aplikasi yang bisa melakukan steganografi sesuai dengan metode yang diusulkan.

1.5 Manfaat

Manfaat dari hasil pembuatan tugas akhir ini adalah sebagai berikut:

1. Menghasilkan teknik steganografi di mana media penyimpanan memiliki ketahanan terhadap perubahan skala, rotasi dan pengaburan citra sehingga informasi masih dapat diekstrak kembali walau pun media penyimpanan sudah mengalami perubahan seperti yang disebutkan.
2. Menghasilkan aplikasi steganografi dengan memanfaatkan teknik yang diusulkan.

1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Penyusunan proposal Tugas Akhir.
   Proposal tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir.
2. Studi literatur
   Pada tahap ini dilakukan pemahaman informasi dan literatur yang diperlukan untuk tahap implementasi program. Tahap ini diperlukan untuk membantu memahami penggunaan komponen-komponen terkait dengan sistem yang akan dibangun seperti steganografi menggunakan algoritma RDE, Algoritma SURF

untuk pendeteksian *interest point*, *library* "Open CV" versi 2.4 sebagai *library* dalam pengolahan citra dan Python-CGI yang digunakan sebagai media untuk membuat antarmuka dari aplikasi.

3. Analisis dan perancangan perangkat lunak
   Tahap ini meliputi perancangan sistem berdasarkan studi literatur dan pembelajaran konsep teknologi dari perangkat lunak yang ada. Tahap ini mendefinisikan alur dari implementasi. Langkah-langkah yang dikerjakan juga didefinisikan pada tahap ini. Pada tahapan ini dilakukan perancangan teknik untuk memilih titik yang akan digunakan serta cara penyimpanan data terhadap titik yang sudah dipilih. Pada tahapan ini dirancang fungsi untuk mendeteksi *interest point*, memilih titik yang bisa digunakan untuk menyimpan data, teknik menyimpan data ke *file* citra dan fungsi untuk mengekstrak kembali data yang sudah disimpan.
4. Implementasi perangkat lunak
   Implementasi perangkat lunak merupakan tahap membangun rancangan program yang telah dibuat. Pada tahap ini akan direalisasikan mengenai rancangan apa saja yang telah didefinisikan pada tahap sebelumnya. Fungsi yang ada pada tahap ini merupakan implementasi dari fungsi yang dirancang sebelumnya. Implementasi menggunakan bahasa Python versi 2.7 dengan IDE "PyCharm" versi 5.04 serta *library* "Open CV" versi 2.4
5. Pengujian dan evaluasi
   Pada tahap ini dilakukan uji coba pada data yang telah dikumpulkan. Tahap ini digunakan untuk mengevaluasi kinerja program serta mencari masalah yang mungkin timbul saat program dievaluasi serta melakukan perbaikan jika terdapat kesalahan pada program.
6. Penyusunan buku Tugas Akhir.
   Pada tahap ini disusun buku yang memuat dokumentasi mengenai perancangan, pembuatan serta hasil dari implementasi perangkat lunak yang telah dibuat.

1.7 Sistematika Penulisan

Buku Tugas Akhir ini terdiri dari beberapa bab, yang dijelaskan sebagai berikut:

1. BAB I Pendahuluan
   Bab ini berisi latar belakang masalah, rumusan dan batasan permasalahan, tujuan dan manfaat pembuatan Tugas Akhir, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.
2. BAB II Tinjauan Pustaka
   Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan Tugas Akhir ini.
3. BAB III Analisis Dan Perancangan
   Bab ini membahas perancangan dan pengembangan metode yang akan dikerjakan.
4. BAB IV Implementasi
   Bab ini membahas implementasi dari rancangan metode yang sudah dibuat pada tahap perancangan.
5. BAB V Pengujian Dan Evaluasi
   Bab ini akan membahas uji coba menggunakan metode yang diusulkan. Setiap percobaan akan dihitung PSNR, kapasitas dan *bit error rate*.
6. BAB VI Penutup
   Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan selanjutnya dari metode yang diusulkan.

[Halaman ini sengaja dikosongkan]

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan algoritma yang diajukan pada pengimplementasian program. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap program yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak

2.1 Steganografi

Steganografi berasal dari bahasa yunani yaitu "*steganos*" yang berarti "tersembunyi atau terselubung", dan "*graphein*" yang berarti "menulis". Jadi steganografi adalah ilmu menyembunyikan informasi dengan suatu cara sehingga selain pengirim dan penerima tidak ada yang mengetahui dan menyadari bahwa ada informasi rahasia [3]. Proses penyembunyian data informasi menggunakan steganografi adalah dengan memodifikasi *bit* pada medium penyimpanan.

Dalam steganografi terdapat beberapa aspek yang dijadikan parameter dalam menentukan bagus atau tidaknya sebuah algoritma steganografi. Parameter yang dimaksud adalah [1] .

- Ketahanan medium penyimpanan
- Kapasitas data yang bisa disembunyikan
- Kualitas citra hasil steganografi yang biasa diukur menggunakan parameter PSNR.
- *Interceptibility*, yaitu seberapa tidak terlihat pengaruh data yang disisipkan terhadap citra.

2.2 Difference Expansion

Metode DE (*Difference Expansion)* merupakan metode steganografi yang melakukan penyisipan data dengan mengubah nilai jarak antara dua buah piksel. Kelebihan metode DE adalah, citra steganografi bisa dikembalikan kembali ke bentuk semula setelah dilakukan ekstraksi data.

Untuk sepasang citra keabuan 8 *bit (x,y)*, di mana $x, y \in Z, 0 \leq x, y \leq 255$, maka nilai rata-rata *m* dan selisihnya *d* dapat dihitung dengan Rumus (2-1) [4].

$$m = \left\lfloor \frac{x+y}{2} \right\rfloor, d = x - y \quad (2\text{-}1)$$

Di mana $\lfloor z \rfloor$ adalah operasi pembulatan nilai ke bawah dari bilangan desimal. Rumus yang digunakan untuk mendapatkan nilai piksel awal dari x dan y ketika proses ekstraksi dijelaskan pada Rumus (2-2)(2-2 [4].

$$x = m + \left\lfloor \frac{d+1}{2} \right\rfloor, y = m - \frac{d}{2} \quad (2\text{-}2)$$

Penyisipan satu *bit* data pada selisih *x* dan *y* dengan metode *Difference Expansion* didefinisikan pada Rumus (2-3) [4].

$$d' = 2\,x\,d + b \quad (2\text{-}3)$$

Di mana *d'* merupakan hasil penyisipan dan *b* adalah *bit* yang akan disisipkan. Nilai *b* adalah 1atau 0. Dalam proses penyisipan, terkadang nilai baru dari *x* dan *y* melebihi batas nilai piksel normal. Nilai *x* dan *y* mengalami *overflow* ketika nilai *x* dan *y* melebihi 255 dan mengalami *underflow* ketika nilai *x* dan *y* kurang dari 0. Untuk mencegah itu maka *d'* harus memenuhi Persamaan (2-4) [1].

$$\begin{cases} |d'| \leq 2 \times (255 - m) & jika\ 128 \leq m \leq 255 \\ |d'| \leq m + 1 & jika\ 0 \leq m \leq 255 \end{cases} \quad (2\text{-}4)$$

Jika nilai dari *d'* memenuhi Persamaan (2-4), maka *d* dapat diekspansi, namun jika tidak maka *d* tidak dapat diekspansi. Nilai dari selisih yang dapat diekspansi, nantinya bisa digunakan untuk penyisipan sebuah *bit*.

Sebagai contoh, terdapat pasangan piksel *x* dan *y*. Nilai *x* dan *y* yaitu *x*=205 dan *y*=201. Berikut langkah untuk melakukan penyisipan data 1 *bit* pada pasangan piksel tersebut.

1. Hitung nilai rata-rata *m* dan nilai selisih *d* menggunakan Rumus (2-1)

$$m = \left\lfloor \frac{x+y}{2} \right\rfloor = \left\lfloor \frac{205+201}{2} \right\rfloor = \left\lfloor \frac{406}{2} \right\rfloor = 203$$

dan

$$d = 205 - 201 = 4$$

2. Sisipkan nilai *bit* ke LSB dari selisih *d* dan buat nilai baru *d'* menggunakan Rumus (2-3).

$$d' = 2 \times d + b = 2 \times 4 + 1 = 9$$

3. Hitung nilai piksel baru (*x',y'*) dengan menggunakan nilai selisih *d'* yang baru dan nilai rata-rata *m*.

$$x' = m + \left\lfloor \frac{d'+1}{2} \right\rfloor = 203 + \left\lfloor \frac{9+1}{2} \right\rfloor$$

$$x' = 203 + 5 = 208$$

$$y' = m - \left\lfloor \frac{d'}{2} \right\rfloor = 203 - \left\lfloor \frac{9}{2} \right\rfloor$$

$$y' = 203 - 4 = 199$$

Proses ekstraksi dari data tersembunyi hampir sama dengan proses penyisipannya. Langkah-langkahnya adalah sebagai berikut.

1. Hitung nilai rata-rata *m'* dan nilai selisih *d'* pada nilai piksel yang baru

$$m = \left\lfloor \frac{x'+y'}{2} \right\rfloor = \left\lfloor \frac{208+199}{2} \right\rfloor = \left\lfloor \frac{407}{2} \right\rfloor = 203$$

dan

$$d = 208 - 199 = 9$$

2. Lakukan ekstraksi pada LSB dari *d'* untuk mendapatkan *bit b* yang disisipkan.

$$b = LSB(d') = LSB(9) = 1$$

$$d = \left\lfloor \frac{d'}{2} \right\rfloor = \left\lfloor \frac{9}{2} \right\rfloor = 4$$

Di mana LSB(*i*) merupakan fungsi ekstraksi LSB yang mengembalikan nilai *bit* terakhir dari *i*.

3. Hitung nilai asli pasangan piksel *(x,y)* dengan menggunakan nilai rata-rata *m'* dan nilai selisih piksel *d*.

$$x = m + \left\lfloor \frac{d+1}{2} \right\rfloor = 203 + \left\lfloor \frac{4+1}{2} \right\rfloor$$
$$x = 203 + 2 = 205$$
$$y = m - \left\lfloor \frac{d}{2} \right\rfloor = 203 - \left\lfloor \frac{4}{2} \right\rfloor$$
$$y = 203 - 2 = 201$$

Metode ini dapat diaplikasikan pada sebuah citra di mana untuk satu blok bisa digunakan untuk penyisipan lebih dari satu kali atau *multilayer*. Namun, jika menggunakan *multilayer* dalam penanaman data di citra, kualitas dari citra akan menurun jauh setelah penanaman data. Permasalahan ini yang nantinya akan diselesaikan dengan metode RDE (*Reduce Difference Expansion*).

2.3 Reduce Differenece Expansion (RDE)

RDE adalah metode steganografi yang diusulkan oleh Lou dkk [4] menggunakan sebuah fungsi transformasi untuk mereduksi nilai selisih dari pasangan piksel *x* dan *y*. Nilai selisih *d'* merupakan nilai selisih yang dapat direpresentasikan pada Rumus (2-5).

$$d' = \begin{cases} d, & d' < 2 \\ d - 2^{\lfloor \log_2 d \rfloor - 1}, & d' \geq 2 \end{cases} \quad (2\text{-}5)$$

Untuk mendapatkan kembali nilai pasangan piksel awal ketika proses ekstraksi, maka dibuat peta lokasi ketika melakukan reduksi terhadap nilai selisih awal. Jumlah peta lokasi harus sama dengan jumlah pasangan piksel yang digunakan. Ketika selisih *d* bernilai 0 atau 1, nilai piksel tidak diubah dan peta lokasi bernilai 0. Ketika selisih *d*=2 dan *d'*=1, peta lokasi bernilai 1. Nilai dari peta lokasi ditentukan dari Rumus (2-6) [4].

$$LM = \begin{cases} 0, & jika\ 2^{|\log_2 d'|} = 2^{|\log_2 d|}\ \text{atau}\ d' = d \\ 1, & jika\ 2^{|\log_2 d'|} \neq 2^{|\log_2 d|} \end{cases} \quad (2\text{-}6)$$

Sebagai contoh, terdapat pasangan piksel dengan nilai x = 205 dan y = 201. Nilai selisih d = 4 dan nilai rata-rata m = 203. Nilai d' dapat dihitung dengan

$$d' = d - 2^{|\log_2 d|-1} = 4 - 2^{|\log_2 4|-1}$$

$$d' = 4 - 2^{2-1} = 4 - 2 = 2$$

Berikutnya dilakukan penyisipan data 1 *bit* b =1 ke dalam selisih d'.

$$d'' = 2 \times d' + \mathrm{b} = 2 \times 2 + 1 = 5$$

$$x' = m + \left\lfloor \frac{d'' + 1}{2} \right\rfloor = 203 + \left\lfloor \frac{5+1}{2} \right\rfloor = 203 + 3 = 206$$

$$y' = m - \left\lfloor \frac{d''}{2} \right\rfloor = 203 - \left\lfloor \frac{5}{2} \right\rfloor = 203 - 2 = 201$$

Untuk mendapatkan kembali *bit* yang disisipkan, digunakan Rumus (2-7) [4].

$$d = \begin{cases} d' + 2^{|\log_2 d'|-1}, & \text{jika peta lokasi} = 0 \\ d' + 2^{|\log_2 d'|}, & \text{jika peta lokasi} = 1 \end{cases} \quad (2\text{-}7)$$

Untuk mendapatkan kembali nilai piksel awal dari pasangan piksel (206,201), pertama hitung nilai rata-rata m dan selisih d''.

$$m = \left\lfloor \frac{206 + 201}{2} \right\rfloor = \left\lfloor \frac{407}{2} \right\rfloor = 203,$$

dan

$$d'' = x - y = 206 - 201 = 5$$

Selanjutnya, ekstrak *bit* data yang tersembunyi b = 1 dan menghitung *d'* dengan persamaan berikut.

$$b = LSB(d'') = LSB(5) = 1$$

$$d' = \left\lfloor \frac{d''}{2} \right\rfloor = \left\lfloor \frac{5}{2} \right\rfloor = 2$$

Untuk mendapatkan selisih awal *d* maka digunakan Persamaan (2-7)

$$d = d' + 2^{|\log_2 d'|} = 2 + 2^{|\log_2 2|} = 2 + 2^1 = 2 + 2 = 4$$

Namun jika *d'* = [0,1] dan peta lokasi= 1, maka nilai *d=d'*. Selanjutnya untuk mendapatkan nilai awal piksel (*x,y*), gunakan selisih *d* dan rata-rata *m*

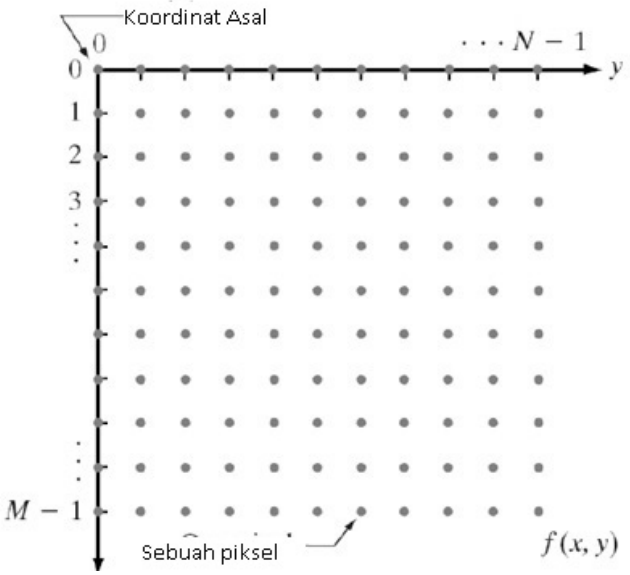
$$x = m + \left\lfloor \frac{d+1}{2} \right\rfloor = 203 + \left\lfloor \frac{4+1}{2} \right\rfloor$$
$$x = 203 + 2 = 205$$
$$y = m - \left\lfloor \frac{d}{2} \right\rfloor = 203 - \left\lfloor \frac{4}{2} \right\rfloor$$
$$y = 203 - 2 = 201$$

2.4 Citra Digital

Citra digital adalah sebuah array bilangan real positif dengan ukuran dua dimensi. Nilai dari setiap indek di array tersebut merepresentasikan warna pada piksel tersebut. Sebuah citra dapat direpresentasikan dengan ukuran *M* baris dan *N* kolom. Di mana *M*

dan *N* merupakan ukuran dari array dua dimensi yang menampung nilai piksel dari citra tersebut. Ukuran pada citra ditentukan dari banyak baris dan kolom piksel pada citra tersebut. Citra didefinisikan sebagai fungsi *f(x,y)*, di mana *x* dan *y* merupakan koordinat yang menunjukkan letak piksel tersebut.

Sebuah citra dapat direpresentasikan ke dalam bentuk sebuah matriks, seperti ditunjukkan gambar Gambar 2-1 [5].



Gambar 2-1 Representasi Citra Digital

Sebuah citra dapat juga direpresentasikan ke dalam bentuk matrik 2 dimensi dengan ukuran *M* x *N* di mana *M* adalah lebar dari citra dan *N* adalah tinggi dari citra. Matrik tersebut dapat ditulis seperti Rumus (2-8) [5].

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (2\text{-}8)$$

2.4.1 Citra Keabuan

Citra keabuan merupakan citra di mana nilai piksel pada setiap citra merepresentasikan tingkat kecerahan dari setiap piksel tersebut. Citra RGB dapat diubah menjadi citra keabuan dengan menggunakan Rumus (2-9).

$$\mathbf{F(x,y) = 0.21 * f(x,y,r) + 0.72 * f(x,y,g) + 0.07 * f(x,y,b)} \quad (2\text{-}9)$$

Dengan kata lain citra keabuan merupakan citra digital yang hanya memiliki satu nilai di setiap pikselnya. Citra keabuan sering digunakan untuk menyimpan data pada steganografi karena efek dari perubahan nilai piksel yang dilakukan pada citra tidak terlalu kelihatan.

Gambar 2-2 Konversi Citra RGB Menjadi Citra Keabuan

Warna yang dimiliki oleh citra keabuan adalah warna hitam, putih dan keabuan. Keabuan yang dimaksud adalah warna di antara warna putih yang menuju warna hitam. Salah satu jenis citra keabuan adalah citra keabuan yang memiliki kedalaman warna 8 *bit*, dengan rentang nilai piksel yang mungkin adalah dari 0 hingga 255. Contoh konversi citra RGB menjadi citra keabuan ditunjukkan pada Gambar 2-2.

2.4.2 *Euclidean Distance* dan *Chebyshev distance*

Euclidean adalah jarak antara dua titik [6] . Sebagai contoh nilai *euclidean E* dari dua buah titik p_1 dan p_2 dengan koordinat (x_1,y_1) dan (x_2,y_2) dapat dihitung menggunakan Persamaan (2-10).

$$E = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (2\text{-}10)$$

Chebyshev distance adalah nilai yang didapatkan dari nilai absolut selisih dari sebuah pasangan koordinat [7]. Sebagai contoh nilai *Chebyshev distance C* dari dua buah titik p_1 dan p_2 dengan koordinat (x_1,y_1) dan (x_2,y_2) dapat dihitung menggunakan Persamaan (2-11).

$$C = \max(|x_2 - x_1|, |y_2 - y_1|) \quad (2\text{-}11)$$

2.4.3 Piksel Tetangga

Piksel tetangga untuk sebuah piksel *p* pada koordinat *(x,y)* adalah sekumpulan piksel yang mengelilingi piksel *p* tersebut. Piksel tetangga dari sebuah piksel *p* memiliki nilai *euclidean* = 1. Satu piksel *p* memiliki 4 buah piksel tetangga yaitu (*x,y-1*), (*x-1,y)*, (*x+1,y*) dan (*x*, y+1).

	P1	
P2	P	P3
	P4	

Gambar 2-3 Piksel Tetangga Dari Piksel *P*

Gambar 2-3 menunjukkan piksel tetangga dari piksel *P* yang dinotasikan dengan $N_4(p)$.

2.4.4 *Moore Neighborhood*

Moore Neighborhood adalah piksel tetangga dari sebuah piksel *p* yaitu himpunan 8 titik yang saling bersinggungan dengan piksel *p* yang menjadi titik tengah dari himpunan tersebut. *Moore Neighborhood* dari sebuah piksel *p* adalah piksel-piksel dengan nilai dari *Chebyshev distance* adalah 1 [8].

Untuk satu piksel *p* pada koordinat *(x,y)*, terdapat delapan piksel tetangga pada koordinat (*x* - 1 , *y*-1), (*x*, *y*-1), (*x*+1, *y*-1), (*x*-1, *y*), (*x*+1, *y*), (*x*-1, *y*+1), (*x*, *y*+1) dan (*x*+1, *y*+1) dan dinotasikan seperti Persamaan (2-12) [8].

$$N^{M}_{(x0,y0)} = \langle (x,y): |x - x_0| \leq 1, |y - y_0| \leq 1 \rangle \qquad (2\text{-}12)$$

Detail dari piksel tetangga pada *Moore Neighborhood* dapat dilihat pada Gambar 2-4

P1	P2	P3
P4	C	P5
P6	P7	P8

Gambar 2-4 Kumpulan *Moore Neighborhood* Untuk Piksel *p*

2.4.5 PGM (*Portable Graymap Format*)

PGM adalah salah satu tipe dari penyimpanan citra. Untuk setiap *file* PGM, *file* diawali dengan dua karakter yang menunjukkan tipe dari PGM dan diikuti dengan informasi dari gambar tersebut, seperti lebar dan tinggi serta nilai dari setiap piksel [9]. Untuk *file* berekstensi PGM, *file* diawali dengan karakter P5 yang menunjukkan bahwa *file* memiliki ekstensi PGM.

2.5 Fitur Pada Citra

Citra adalah yang dihasilkan dari gambar analog dua dimensi yang kontinu menjadi gambar diskrit melalui proses sampling [10].

Citra merupakan sebuah data multimedia yang memiliki beberapa fitur sebagai data pengolahan. Fitur pada citra umumnya dapat digolongkan menjadi 3 golongan. Golongan yang pertama adalah komposisi warna atau histogram. Golongan kedua adalah tekstur dari citra dan golongan ketiga adalah *interest point*.

Histogram adalah diagram grafis yang merepresentasikan distribusi warna dalam sebuah citra. Domain warna berada pada rentang 0 sampai dengan 255. Banyak informasi dari sebuah citra yang bisa didapatkan dalam histogram, seperti peluang kemunculan dari sebuah piksel, intensitas piksel yang menonjol dan sebagainya. Tekstur dalam citra menyatakan pola dalam golongan nilai piksel.

Interest point atau yang biasa disebut juga dengan *keypoint* adalah sekumpulan point pada citra yang memiliki nilai nilai unik, di mana nilai dan posisinya tidak berubah walaupun dilakukan rotasi, perubahan ukuran serta perubahan warna minoritas pada sebuah citra. *Interest point* sering digunakan dalam menentukan kemiripan dan kecocokan antar citra [2]. Ada beberapa algoritma dalam pendeteksian *interest point*, beberapa algoritma yang terkenal adalah algoritma SIFT (*Scale-Invariant Feature Transform*) dan algoritma SURF (*Speeded-Up Robust Features*) [2]. Algoritma SURF memiliki kelebihan dari algoritma SURF yaitu waktu komputasi yang lebih cepat [11]. Hal ini dikarenakan adanya proses *integral image* pada algoritma SURF.

2.6 Algoritma SURF

SURF (*Speeded-Up Robust Features*) adalah sebuah algoritma yang cepat dan akurat untuk proses mendeteksi *descriptor* lokal dari kesamaan representasi citra *invariant* [11]. *Descriptor* adalah sebuah ciri-ciri dari suatu citra berdasarkan aturan tertentu dari suatu algoritma. SURF menggunakan citra integral untuk meningkatkan kecepatan komputasi [2]. Algoritma ini menggunakan teori *multi-scale space* dan deteksi fitur menggunakan "*hessian matrix"*. Algoritma ini didasarkan pada kerangka SURF dari hasil disertasi Herbert Bay [2]. Secara umum algoritma SURF memiliki 4 tahapan, yaitu:

1. Pendeteksian *Keypoint*

   Pendeteksian *Keypoint* menggunakan *matrik* Hessian. Matrik Hessian adalah matrik yang setiap elemennya dibentuk dari turunan parsial kedua dari suatu fungsi [2] .

   Keypoint merupakan *interest point* yang sudah dianggap valid karena sudah berada didalam batas nilai yang sudah ditentukan. Determinasi dari *matrik* Hessian digunakan untuk menentukan lokasi dan skala untuk menyeleksi kandidat *interest point*.

2. *Scale Space Representation*

   Menangani dalam perbedaan ukuran dengan menggunakan metode perbandingan skala. Metode ini menggunakan *scale-spaces* yaitu citra diimplementasikan kedalam sebuah *image pyramids*. Citra akan diperhalus secara berulang dengan *Gaussian* dan kemudian menggunakan *sub-sampled* untuk mecapai tingkat tertinggi dari piramida [2].

3. Pendeskripsian *Keypoint*

   Pendeskripsian keypoint dibagi menjadi dua tahap. Tahap pertama adalah menetapkan orientasi berdasarkan informasi dari daerah melingkar di sekitar *interest point* yang terdeteksi. Yang kedua adalah membuat *grid* sebesar 64 (8x8) yang digunakan untuk menampung *descriptor* yang berkorespondensi dengan histogram pada Haar Wavelet.

Gambar 2-5 Daerah *Interest Point*

Gambar 2-5 menunjukkan daerah dari *interest point* yang terdeteksi dengan batasan nilai *threshold* 500.

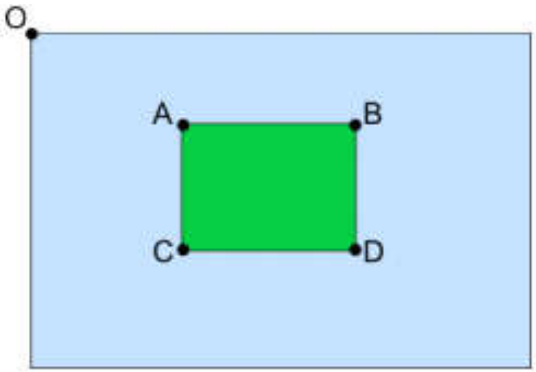
2.6.1 *Integral Image*

Integral image sangat berpengaruh dalam peningkatan performa algoritma SURF. *Integral image* digunakan untuk memepercepat perhitungan jumlah nilai-nilai piksel pada sebuah daerah persegi [12]. Untuk sebuah citra *I* dan titik (*x,y*) *integral image* $I_{\sum}$ adalah jumlah nilai piksel antara titik asal dan titik (*x,y)*, dapat dihitung dengan Persamaan (2-13) [2] [12].

$$I_{\sum(x,y)} = \sum_{i\leq 0}^{i\leq x}\sum_{j\leq 0}^{j\leq y} I(x,y) \tag{2-13}$$

Dengan mengggunakan *integral image*, untuk menentukan jumlah piksel dari persegi yang terdapat di citra dapat ditentukan hanya dengan empat operasi. Sebagai contoh, terdapat persegi dengan titik A, B, C dan D seperti pada Gambar 2-6 , jumlah piksel pada persegi dihitung dengan Rumus (2-14) [12] [2].

$$\sum = A + D - (C + B) \tag{2-14}$$



Gambar 2-6 Komputasi Untuk Menghitung Luas Persegi A-B-C-D

3	7	7	3
1	3	3	1
5	9	9	5
3	6	6	3

3	10	17	20
4	14	24	28
9	28	47	56
12	37	62	74

Gambar 2-7 Citra Awal (Kiri) dan Citra *Integral* (Kanan)

Untuk menghitung luas daerah citra awal yang ditandai dapat digunakan nilai dari citra *integral* seperti Persamaan (2-14).

$$\sum = 14 + 74 - (37 + 28) = 23$$

2.6.2 Fast Hessian Detector

Pendeteksian *interest point* menggunakan algoritma SURF ditentukan dengan determinan dari matriks *hessian*. Matriks *hessian* adalah turunan parsial dari fungsi f(*x,y*). Nilai dari determinan matriks *hessian* yang sudah dinormalisasi atau ketika skala dinormalkan menandakan kestabilan dari sebuah *interest point* [13].

2.6.3 Fitur Orientasi dan Skala pada *Interest Point*

Orientasi atau arah adalah salah satu fitur yang dimiliki oleh *interest point*. Orientasi akan sangat berguna untuk pencocokan citra ketika citra diputar. Ekstraksi fitur menggunakan respons dari *Harr Wavelet* dengan membentuk histogram dari setiap respons yang diberikan.

Penentuan orientasi dari respons *Harr Wavelet* adalah dengan merotasi sebuah bidang dengan sudut $\frac{\pi}{3}$ mengelingingi daerah dari *interest point*. Di setiap posisi dihitung *y-respon* dan *x-respon* kemudian jumlahkan ke dalam vektor [2]. Vektor dengan nilai terpanjang mewakili nilai dari orientasi di titik tersebut.

Skala adalah salah satu fitur yang digunakan untuk menangani perubahan skala pada *interest point*. Fitur skala menggambarkan luas daerah pada *interest point* tersebut. Perubahan

nilai piksel pada daerah *interest point* dengan jarak piksel yang diubah lebih kecil dari skala akan mempengaruhi stabilitas *interest point* tersebut.

2.7 *Peak Signal-Noise Ratio* (PSNR)

Untuk mengevaluasi kualitas citra hasil dari modifikasi citra setelah data disisipkan dengan steganografi dapat dengan menggunakan nilai MSE (*Mean Square Error*) dan PSNR (*Peak Signal-Noise Ratio*).

PSNR adalah salah satu indikator yang digunakan dalam komputasi citra untuk mengukur kualitas dari sebuah citra yang sudah dimanipulasi terhadap citra awal. Untuk menghitung PSNR maka harus ditentukan nilai dari MSE terlebih dahulu.

MSE adalah nilai kesalahan kuadrat rata-rata antara citra asli dengan citra hasil modifikasi (citra steganografi). Untuk menentukan MSE dapat ditentukan oleh Rumus(2-15) [14].

$$MSE = \frac{1}{MN} * \sum_{x=1}^{M} \sum_{y=1}^{N} \left(S_{xy} - C_{xy}\right)^2 \qquad (2\text{-}15)$$

Di mana
M = lebar dari citra awal
N = tinggi dari citra awal
S_{xy} = nilai piksel pada citra hasil steganografi pada titik x,y
C_{xy} = nilai piksel pada citra asli pada titik x,y

Setelah nilai dari MSE didapatkan, maka nilai PSNR dapat dihitung menggunakan Rumus (2-16) [14].

$$PSNR = 10 * \log_{10}\left(\frac{Cmax^2}{MSE}\right) \qquad (2\text{-}16)$$

Di mana *Cmax* adalah nilai piksel maksimal yang mungkin dari citra *cover*. Untuk citra dengan rentang nilai piksel antara 0 hingga 255, maka *Cmax* bernilai 255. Sedangkan untuk citra dengan rentang nilai piksel antara 0 hingga 1, maka *Cmax* bernilai 1.

Semakin tinggi nilai PSNR dari sebuah citra hasil steganografi, maka menandakan bahwa kualitas dari citra tersebut semakin bagus. Bagusnya kualitas citra menunjukkan bahwa perubahan terhadap citra setelah dimodifikasi hampir tidak terlihat

2.8 *Bit Error Rate* (BER)

BER adalah parameter untuk mengukur persentase kegagalan dalam proses ekstraksi data. Bit yang eror dari data hasil ekstraksi akan dibandingkan dengan bit data masukan. Untuk menghitung nilai BER dapat menggunakan Persamaan (2-17).

$$BER = \frac{jumlah\ bit\ error}{jumlah\ bit\ inputan} \qquad (2\text{-}17)$$

2.9 Open CV

Open CV adalah *library* yang digunakan untuk pemrosesan citra. Open CV tersedia dalam banyak bahasa seperti C, C++, Python, Java dan mendukung sistem operasi Windows, Linux, Mac OS, iOS dan Android.

Open CV yang digunakan dalam pengerjaan tugas akhir ini adalah Open CV versi 2.4. Hal ini dikarenakan Open CV versi berikutnya fungsi untuk algoritma SURF sudah dihilangkan.

2.10 Python CGI

Python CGI adalah salah satu modul pada bahasa pemrograman Python yang memungkinkan *file* Python tersebut diakses melalui browser. Pada Python CGI *web server* Apache digunakan agar pengguna bisa mengakses melalui *web browser*. Python CGI kode HTML dicetak oleh *file* Python beserta *header response* ketika pengguna mengakses *file* tersebut.

BAB III
ANALISIS DAN PERANCANGAN

Pada bab ini akan dijelaskan mengenai deskripsi umum metode, perancangan pendeteksian *interest point*, perancangan penyisipan dan ekstraksi *bit* dan perancangan metode yang diusulkan.

3.1 Deskripsi Umum Metode

Metode pengembangan yang diusulkan terdapat pada mekanisme pemilihan blok yang akan digunakan untuk penyisipan *bit* data. Blok yang bisa digunakan adalah daerah dari *interest point* yang terdeteksi. Setiap blok hanya digunakan untuk sekali penyisipan data. Setiap satu titik *interest point* bisa menghasilkan banyak blok yang bisa digunakan, tergantung dari luasnya daerah yang dibentuk oleh *interest point* tersebut.

Blok yang digunakan adalah kumpulan piksel dari setiap daerah yang dibentuk oleh *interest point* di mana luas daerah ditentukan dari nilai skala untuk setiap *interest point*. Untuk *interest point I* di titik (*x,y*) memiliki skala *s*. Banyak blok $\sum p$ yang bisa digunakan untuk penyisipan data dengan nilai maksimum *chebyshev distance* c_{max} dapat dihitung menggunakan Rumus (3-1).

$$\sum p = \left\lfloor \left(2 \times \left\lfloor \frac{s}{2} \right\rfloor + 1\right)^2 - (c_{max} + 1)^2 \right\rfloor - 1 \qquad (3\text{-}1)$$

Dengan batasan, setiap daerah yang dibentuk oleh *interest point* tidak boleh saling beririsan. Pengembangan metode juga terdapat dalam pemilihan urutan titik yang disisipi. *Interest point* akan diurutkan berdasarkan respons dari setiap *interest point*.

Nilai respons dari sebuah *interest point* merepresentasikan kestabilan dari *interest point* tersebut. Pemilihan titik untuk setiap daerah pada *interest point* ditentukan berdasarkan nilai orientasi dari setiap *interest point* tersebut sehingga ketika citra mengalami rotasi, urutan titik yang diperoleh tetap sama.

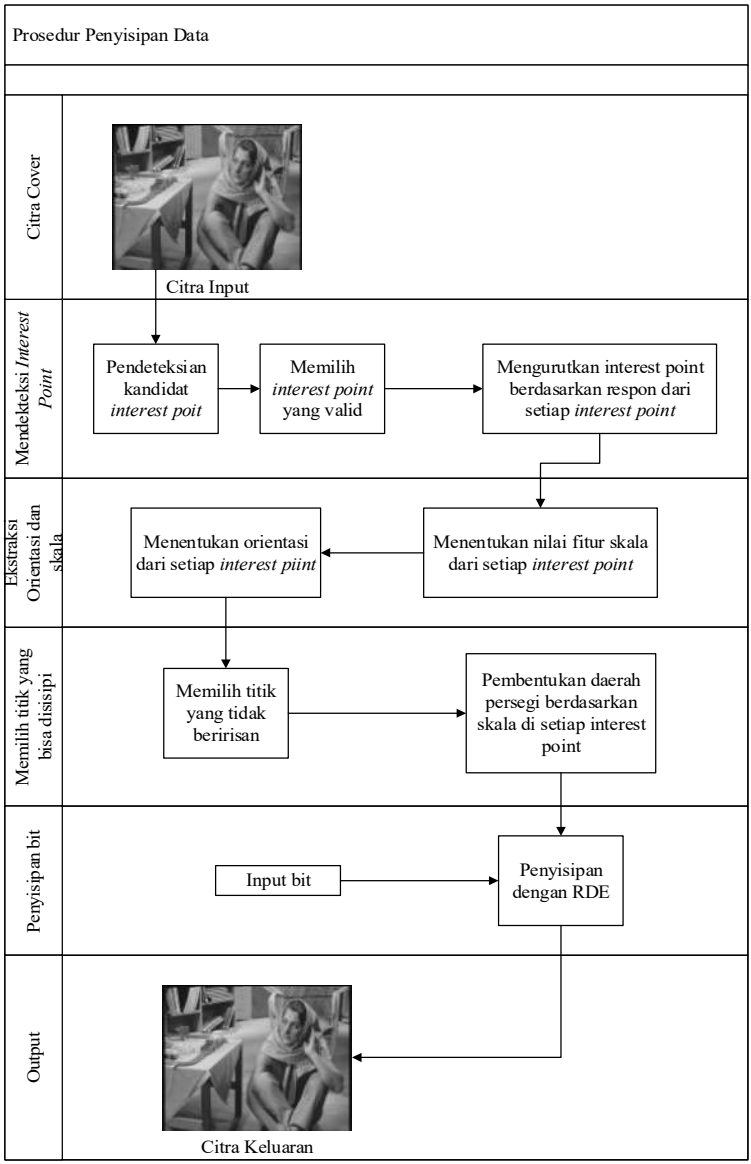
3.2 Perancangan Metode Keseluruhan

Metode steganografi RDE yang digunakan sama dengan metode yang diusulkan oleh Lou dkk [4]. Namun pada metode yang diusulkan lebih mementingkan ke dalam pemilihan titik yang akan disisipi *bit* data. Metode didasarkan pada sifat algoritma SURF dalam mendeteksi *interest point*. Dari hasil percobaan, selanjutnya dipelajari sifat *interest point* terhadap perubahan rotasi pada citra.

Pada metode yang diusulkan oleh Lou dkk [4] penelusuran piksel dilakukan pada keseluruhan citra untuk menemukan pasangan piksel yang akan digunakan untuk membentuk blok penyisipan data. Berbeda dari metode yang diusulkan, pada metode yang diusulkan pasangan piksel yang digunakan untuk membentuk blok penyisipan didapatkan dengan penelusuran pada daerah yang dibentuk oleh *interest point* di dalam citra tersebut.

Pada bagian mendeteksi *interest point*, algoritma SURF mendeteksi piksel yang nilainya memenuhi nilai *threshold* yang diberikan. Pemilihan titik dimaksudkan, agar tidak ada titik yang digunakan berulang kali, walaupun ada daerah yang beririsan. Ini ditujukan untuk menjaga kualitas citra.

Diagram penjelasan langkah kerja secara umum dari metode yang diusulkan dalam penyisipan data menggunakan steganografi dapat dilihat pada Gambar 3-1. Diagram tersebut menjelaskan proses utama dimulai dari masukan citra sebagai media penyimpanan data, hingga keluaran dari proses yaitu citra yang sudah dimodifikasi untuk tempat penyimpanan data beserta peta lokasi. Penjelasan mengenai setiap langkah yang terdapat dalam Gambar 3-1 akan dijelaskan lebih mendalam pada sub bab berikutnya.



Gambar 3-1 Metode Penyisipan Secara Umum

3.3 Perancangan Pendeteksian *Interest Point*

Pendeteksian *interest point* menggunakan algoritma SURF (*Speeded-Up Robust Features*). Algoritma ini didasarkan pada kerangka SURF dari hasil disertasi Herbert bay [2]. *Interest Point* yang digunakan adalah *interest point* yang memenuhi nilai dari *threshold* yang diberikan.

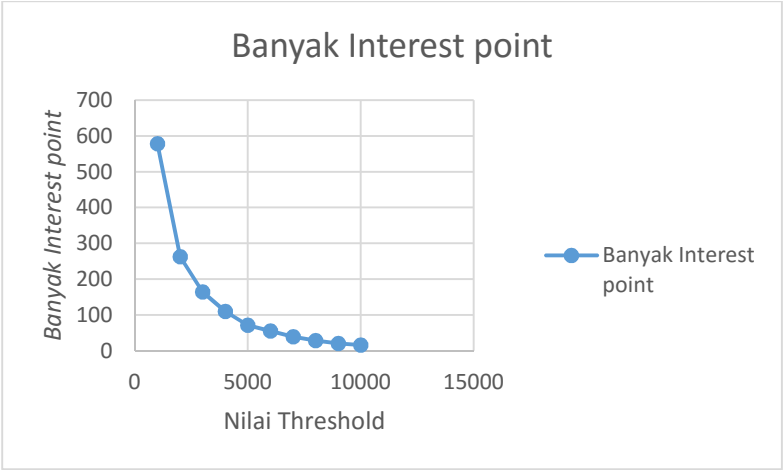
3.3.1 Perancangan Pemilihan *Threshold*

Threshold adalah nilai batasan untuk menentukan suatu kriteria. *Threshold* dalam algoritma digunakan untuk menentukan apakah titik tersebut termasuk *interest point* atau tidak. Sebuah titik akan dikategorikan sebagai *interest point* ketika nilai dari determinan dari *hessian matriks* melebihi nilai *threshold* [2]. *Interest point* akan dibandingkan dengan titik sekitar untuk menentukan *interest point* yang akurat.

Threshold akan sangat menentukan terhadap kualitas dan kuantitas dari *interest point* yang didapatkan. *Threshold* berbanding terbalik terhadap kuantitas *interest point*, berarti semakin besar nilai dari *threshold* semakin sedikit *interest point* yang didapatkan dan sebaliknya.

Threshold berbanding lurus terhadap kualitas *interest point*. Kualitas *interest point* ditentukan seberapa ketahanan *interest point* terhadap perubahan pada citra. *Interest point* yang baik adalah ketika *interest point* masih terdeteksi walaupun sudah dilakukan perubahan pada citra. Semakin besar nilai dari *threshold* maka semakin bagus kualitas *interest point* yang didapatkan.

Pemilihan *threshold* dilakukan dengan pengujian seberapa bagus kualitas dan kuantitas *interest point* untuk nilai *threshold* 1000-10000. Untuk data *interest point* pada *threshold* tertentu di citra *barbara.bmp* dapat dilihat pada Gambar 3-2.



Gambar 3-2 Grafik Pengaruh *Threshold* Terhadap *Interest Point*

3.3.2 Perancangan Penentuan Jumlah Lapisan dan Filter

Dalam algoritma SURF terdapat proses *Scale Space Representation* [12]. Dalam proses ini, citra akan direpresentasikan kedalam sebuah piramida. Dalam setiap lapisan *l* terdapat beberapa filter *f* yang akan digunakan untuk menentukan nilai dari LoG (*Laplacian of Gaussian*). Nilai filter untuk setiap lapisan ditentukan dengan Rumus (3-2) [12].

$$F = 3 \times (2^l \times interval + 1), \quad 0 \leq interval \leq f \qquad (3\text{-}2)$$

Jika kita menggunakan dua lapisan dan terdapat empat filter untuk setiap lapisan, maka nilai dari filter dapat direpresentasikan seperti Tabel 3-1 berikut. Filter akan mempengaruhi hasil dari *interest point*, karena semakin kecil nilai filter maka semakin sedikit *interest point* yang terdeteksi. Ukuran filter digunakan dalam menentukan nilai respons dari titik tersebut.

Tabel 3-1 Nilai Filter di Setiap Lapisan

Lapisan 1	9	15	21	27
Lapisan 2	15	27	39	51
Lapisan 3	39	51	75	99
Lapisan 4	75	99	147	195

Semakin banyak jumlah lapisan yang digunakan, semakin banyak *interest point* yang didapatkan. Hal ini dikarenakan, setiap titik akan dilakukan pengecekan nilai respons terhadap setiap lapisan, sehingga kemungkinan *interest point* semakin besar.

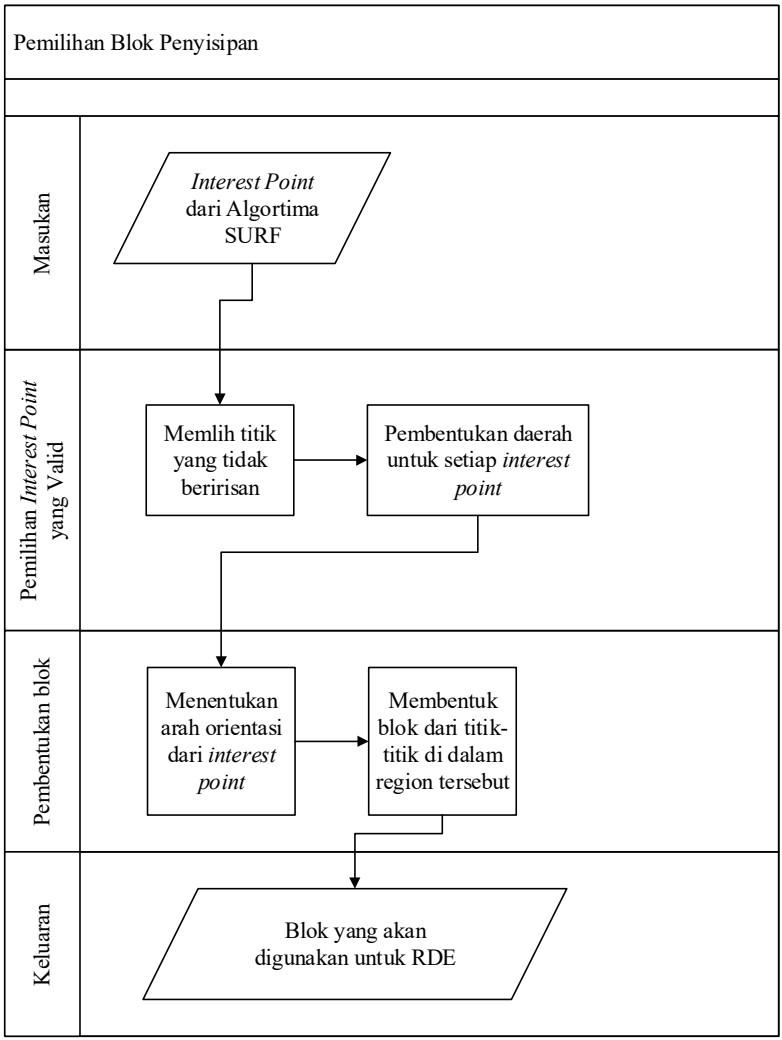
Semakin banyak jumlah filter yang digunakan, semakin bagus hasil dari *interest point* yang didapatkan. Hal ini dikarenakan setiap titik akan dilakukan pengecekan nilai respons menggunakan berbagai filter.

3.4 Perancangan Pembentukan Nilai Blok

Interest point yang terdeteksi tidak akan digunakan secara keseluruhan. Pada tahap ini akan dilakukan penyeleksian *interest point* yang dianggap valid untuk digunakan dalam proses RDE. *Interest point* yang terdeteksi tidak akan digunakan seluruhnya untuk proses steganografi. *Interest point* yang akan digunakan adalah *interest point* yang dianggap valid. Sebuah *interest point* dianggap valid jika memenuhi syarat berikut.

1. Nilai respons dari *interest point* tersebut melebihi *threshold* yang diberikan
2. Nilai *euclidean* dari *interest point* tersebut terhadap *interest point* lainnya memenuhi Persamaan (3-3) . Hal ini dikarenakan untuk menghilangkan *interest point* yang beririsan. Jika terdapat *interest point* yang beririsan, akan diambil *interest point* dengan nilai respons tertinggi.

$$E \geq d\sqrt{2} \tag{3-3}$$

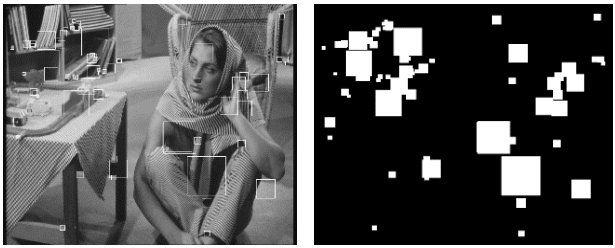


Gambar 3-3 Diagram Alir Pembentukan Blok RDE

3.4.1 Pemilihan Interest *Point* yang Valid

Interest point yang terdeteksi dalam proses algoritma SURF tidak akan digunakan keseluruhan untuk proses RDE. Dalam proses pemilihan, setiap *interest point* akan dibentuk persegi dengan *interest point* sebagai titik tengah dari persegi tersebut. Panjang dari persegi ditentukan oleh ukuran skala dari titik *interest point* tersebut.

Untuk setiap *interest point I* dengan skala *s*, akan dibentuk persegi dengan ukuran $(2 \times \lfloor s/2 \rfloor + 1)\text{^}2$. Dengan pembentukan daerah tersebut, akan terdapat daerah yang saling beririsan. Untuk daerah yang beririsan, akan diambil daerah dengan nilai kestabilan *interest point* yang paling besar. Proses ini dimaksudkan untuk mengurangi efek dari proses RDE terhadap *interest point* tersebut. Setelah dilakukan proses tersebut, daerah untuk penyisipan bisa ditentukan.



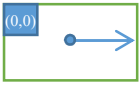
Gambar 3-4 Daerah yang Digunakan Untuk Menyisipkan Data

Gambar 3-4 menunjukkan daerah-daerah yang nantinya bisa digunakan untuk proses penyisipan data. Dari daerah yang terbentuk akan dibentuk blok-blok untuk penyisipan data.

3.4.2 Pembentukan dan Penyusunan Blok

Blok dibentuk dari daerah yang sudah ditentukan pada tahap sebelumnya. Pembentukan blok dimulai dari daerah dengan *interest point* dengan nilai kestabilan tertinggi sampai nilai

kestabilan terendah. Untuk setiap blok, pembentukan blok dipengaruhi oleh orientasi dari *interest point* tersebut, agar efek akibat dari rotasi bisa diatasi. Pemilihan blok ditentukan dari nilai orientasi. Hal ini dikarenakan, nilai orientasi akan selalu berubah ketika dilakukan rotasi pada gambar. Ketika orientasi bernilai 0^o, arah orientasi ditunjukkan oleh Gambar 3-5.



Gambar 3-5 Arah Orientasi 0 Derajat

Sebagai contoh, untuk sebuah *interest point I* dengan nilai orientasi $o = 60^o$, ketika Gambar 3-5 dilakukan rotasi 90 derajat searah jarum jam, maka nilai orientasi di *interest point* tersebut akan menjadi $60 + 90 = 150^o$ seperti yang ditunjukkan pada Gambar 3-6.

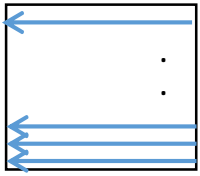
Gambar 3-6 Arah Orientasi 90 Derajat

Karena dalam metode ini mengatasi perubahan pada citra seperti rotasi dan pencerminan, di mana perubahan tersebut akan mempengaruhi nilai dari orientasi untuk setiap *interest point*. Untuk mengatasi hal tersebut, arah dari orientasi akan digunakan untuk menentukan arah penyusunan dan pengambilan titik pada setiap daerah.

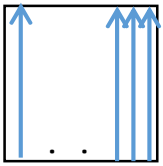
Akan terdapat delapan rentang nilai kemungkinan untuk orientasi pada setiap *interest point*. Hal ini dikarenakan setiap dilakukan rotasi nilai orientasi akan berubah. Rotasi yang bisa ditangani adalah rotasi sempurna, seperti 90^0, 180^0 dan 270^0 sehingga akan terdapat empat kemungkinan nilai orientasi. Selain rotasi citra juga memungkinkan untuk dilakukan pencerminan, di mana citra hasil pencerminan juga akan mempunyai empat kemungkinan nilai rentang orientasi. Sehingga untuk setiap *interest*

point dengan orientasi Φ akan memiliki delapan kemungkinan nilai orientasi.

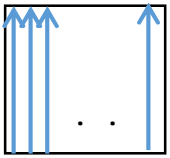
Gambar 3-7 hingga Gambar 3-14 menunjukkan urutan pengambilan titik untuk setiap daerah. Seperti yang ditunjukkan Gambar 3-7, proses penyusunan titik dimulai dari kanan bawah hingga kiri atas.



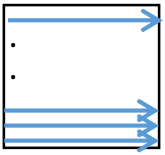
Gambar 3-7 Urutan Pengambilan Blok Untuk Orientasi Antara 0-44 Derajat



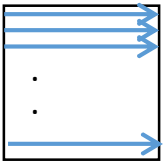
Gambar 3-8 Urutan Pengambilan Blok Untuk Orientasi Antara 45 - 89 Derajat



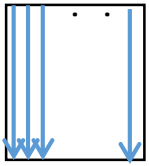
Gambar 3-9 Urutan Pengambilan Blok Untuk Orientasi Antara 90 – 134 Derajat



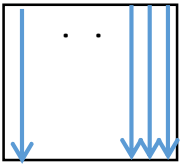
Gambar 3-10 Urutan Pengambilan Blok Untuk Orientasi Antara 135 – 180 Derajat



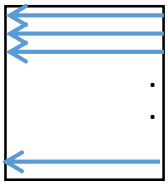
Gambar 3-11 Urutan Pengambilan Blok Untuk Orientasi Antara 180- 224 Derajat



Gambar 3-12 Urutan Pengambilan Blok Untuk Orientasi Antara 225 - 269 Derajat



Gambar 3-13 Urutan Pengambilan Blok Untuk Orientasi Antara 270 – 314 Derajat



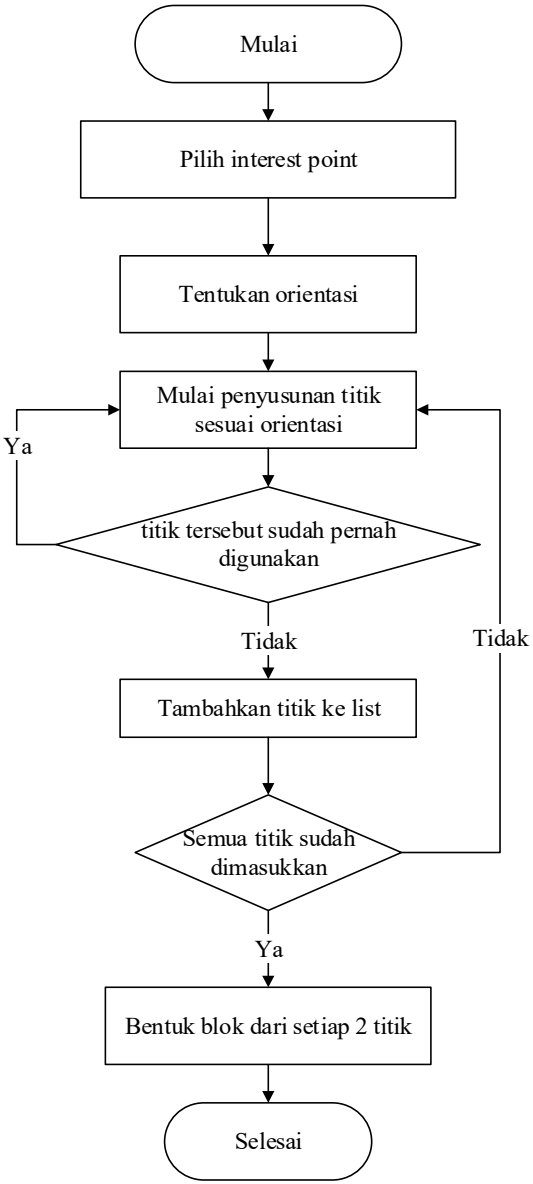
Gambar 3-14 Urutan Pengambilan Blok Untuk Orientasi Antara 315 – 369 Derajat

Untuk setiap daerah, tidak semua titik pada daerah tersebut akan digunakan. Untuk setiap daerah interest point, titik-titik yang memiliki nilai *chebyshev distance* lebih kecil sama dengan dari nilai *chebyshev distance* maksimum yang ditentukan terhadap titik tengah atau titik *interest point* pada daerah tersebut, maka titik tersebut tidak akan digunakan. Sehingga untuk *chebyshev distance* maksimum c_{max} akan terdapat $(2 \times c_{max} + 1)^2$ titik yang tidak digunakan.

Nilai c_{max} yang penulis sarankan adalah 2. Hal ini dikarenakan pada proses penyeleksian *interest point*, setiap *interest point* akan dibandingkan dengan 24 titik di sekitar interest point tersebut. Hal ini dimaksudkan untuk menjaga nilai dari *interest point* ketika proses ekstraksi.

Blok yang dihasilkan dari proses ini nantinya akan digunakan dalam proses RDE. Setiap *interest point I* akan membentuk sebuah daerah, di mana setiap daerah terdiri dari n blok dan setiap blok terdapat 1 pasangan nilai piksel. Jumlah blok *n* dari *interest point* tersebut dapat dihitung menggunakan Rumus (3-1).

Langkah penyusunan blok untuk satu daerah pada *interest point* dapat dilihat pada Gambar 3-15.


Mulai
Pilih interest point
Tentukan orientasi
Mulai penyusunan titik sesuai orientasi
Ya
titik tersebut sudah pernah digunakan
Tidak
Tidak
Tambahkan titik ke list
Semua titik sudah dimasukkan
Ya
Bentuk blok dari setiap 2 titik
Selesai

Gambar 3-15 Diagram Alir Penyusunan Blok Pada Suatu Daerah

3.5 Perancangan Penyisipan dan Ekstraksi *Bit*

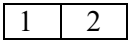
Metode penyisipan menggunakan teknik RDE yang diusulkan oleh Lou dkk [4]. Pada penyisipan dilakukan batasan nilai selisih *d* agar hasil dari hasil steganografi tidak memiliki nilai PSNR yang terlalu kecil sehingga kualitas citra tidak rusak. Nilai *d* atau nilai selisih dari dua piksel seperti Persamaan (2-1). Pada metode normal RDE, semua titik yang memenuhi Persamaan (2-4) bisa digunakan untuk menyembunyikan data.

Pada metode RDE yang diusulkan, ditambahkan batasan tambahan, di mana nilai selisih pasangan piksel *x* dan *y* pada sebuah blok harus lebih kecil dari nilai batasan tertentu. Nilai batasan ditentukan oleh pengguna. Semakin besar nilai selisih yang diperbolehkan, maka semakin besar kapasitas penyisipan yang diperoleh namun semakin besar kemungkinan kesalahan data ketika ekstraksi. Dan berlaku sebaliknya, jika nilai selisih yang diizinkan semakin kecil, maka semakin sedikit titik yang bisa disisipi namun kemungkinan kesalahan ketika ekstraksi semakin kecil.

Dengan metode RDE ini, citra *cover* yang sudah disisipi data, bisa dikembalikan seperti semula ketika proses ekstraksi. Citra *cover* bisa kembali jika nilai *bit error rate* dari hasil ekstraksi data bernilai 0% atau semua data yang disisipkan berhasil diekstrak secara sempurna.

3.5.1 Penentuan Blok

Blok yang digunakan berukuran 2x1 piksel. Gambar 3-16 menunjukkan blok yang terdiri dari 2 piksel. Data *bit* akan disisipkan ke dalam nilai selisih *d* dari pasangan piksel dalam blok tersebut.



Gambar 3-16 Urutan Blok RDE

Sebagai contoh terdapat kumpulan daerah *interest point* seperti Gambar 3-17. Dengan daerah tersebut, akan didapatkan 8 buah blok yang nantinya akan digunakan untuk proses steganografi. Blok yang terbentuk adalah (1,2), (2,3),(3,4),…,(8,9). Nilai dari blok yang saling tumpang tindih dimaksudkan untuk memaksimalkan data yang bisa disisipi pada sebuah citra *cover*.

Hal ini dimaksudkan untuk mengurangi efek perubahan nilai di sekitar *interest point* terhadap *interest point* tersebut. Pemilihan blok dengan titik-titik di antara blok saling tumpang tindih memungkinkan untuk dilakukan karena sifat dari RDE yang bersifat *lossless* sehingga nilai piksel dari citra masih bisa dikembalikan ketika proses ekstraksi.

Dalam sebuah daerah *interest point*, tidak semua piksel pada daerah tersebut yang digunakan sebagai blok untuk penyisipan data. Untuk piksel di sekitar *interest point* dengan nilai *chebyshev distance* kurang dari 5 tidak akan digunakan. Hal ini dikarenakan untuk mengurangi perubahan terhadap *interest point*. Nilai tersebut didapatkan dari ukuran filter yang paling minimum, di mana ukuran paling minimum yaitu 9.

1	2	3
4	5	6
7	8	9

Gambar 3-17 Contoh Kumpulan Piksel dengan Ukuran 3x3

3.5.2 Perancangan Penyisipan *Bit*

Proses penyisipan *bit* dilakukan dari daerah blok pertama hingga daerah blok terakhir. Dalam proses penyisipan, daerah blok dari setiap titik *interest point* diberikan sebagai masukan dari proses penyisipan.

Penyisipan akan dilakukan secara bertahap dimulai dari daerah dengan *interest point* yang mempunyai nilai respons tertinggi hingga daerah dengan *interest point* dengan nilai respons terendah atau hingga semua data sudah digunakan.

Untuk urutan dalam penyisipan blok, metode yang diusulkan adalah dengan memasukkan data dari daerah pertama hingga daerah terakhir dengan memenuhi daerah pertama terlebih dahulu. Untuk setiap 1 *byte* data atau 8 *bit* data, harus disisipkan di daerah yang sama. Ini dimaksudkan untuk mengurangi *byte* yang salah ketika ekstraksi.

Setiap blok akan memiliki nilai peta lokasi tersendiri yang menandakan status dari blok tersebut. Nilai dari peta lokasi ditentukan ketika proses penyisipan. Langkah penyisipan data pada setiap blok di setiap daerah berdasarkan metode yang diusulkan adalah sebagai berikut:

1. Tambahkan banyak *interest point* yang terdeteksi pada proses penyembunyian data di peta lokasi. Hal ini ditunjukkan untuk mengetahui berapa banyak *interest point* yang akan digunakan untuk proses ekstraksi.
2. Mulai lakukan penyisipan 8 *bit* dimulai dari blok pertama pada daerah pertama.
3. Tambahkan nilai dari skala dan nilai jumlah dari *moore neighborhood* dari *interest* point itu ke dalam peta lokasi.
4. Sebelum menyisipkan data, dipastikan terlebih dahulu bahwa daerah tersebut masih bisa menyembunyikan 8 *bit* data.
5. Langkah ketiga ditujukan untuk menghilangkan ketergantungan antara setiap daerah *interest point*, sehingga jika terdapat *interest point* yang salah tidak akan mempengaruhi terhadap daerah *interest point* berikutnya.
6. Jika daerah tersebut masih memungkinkan untuk menampung 8 *bit* data, lakukan penyisipan 8 *bit* data pada blok di daerah tersebut dimulai dari blok setelah blok terakhir yang disisipi.
7. Jika tidak memenuhi lanjut ke daerah berikutnya. Ulangi langkah tiga hingga enam.
8. Jika memenuhi sisipkan data menggunakan metode RDE.

9. Ulangi langkah empat hingga delapan sampai semua data sudah disembunyikan atau semua blok untuk setiap daerah sudah digunakan semua.
10. Simpan peta lokasi ke dalam *file* dan citra hasil modifikasi ke citra yang baru.

Untuk detail dari penyisipan data *bit* dapat dilihat pada Gambar 3-18. Hasil dari penyisipan adalah citra *cover* yang sudah disisipi data yang disembunyikan atau yang biasa disebut citra steganografi. Kualitas dari citra hasil steganografi dapat diukur menggunakan PSNR seperti yang dijelaskan pada sup bab 2.7.

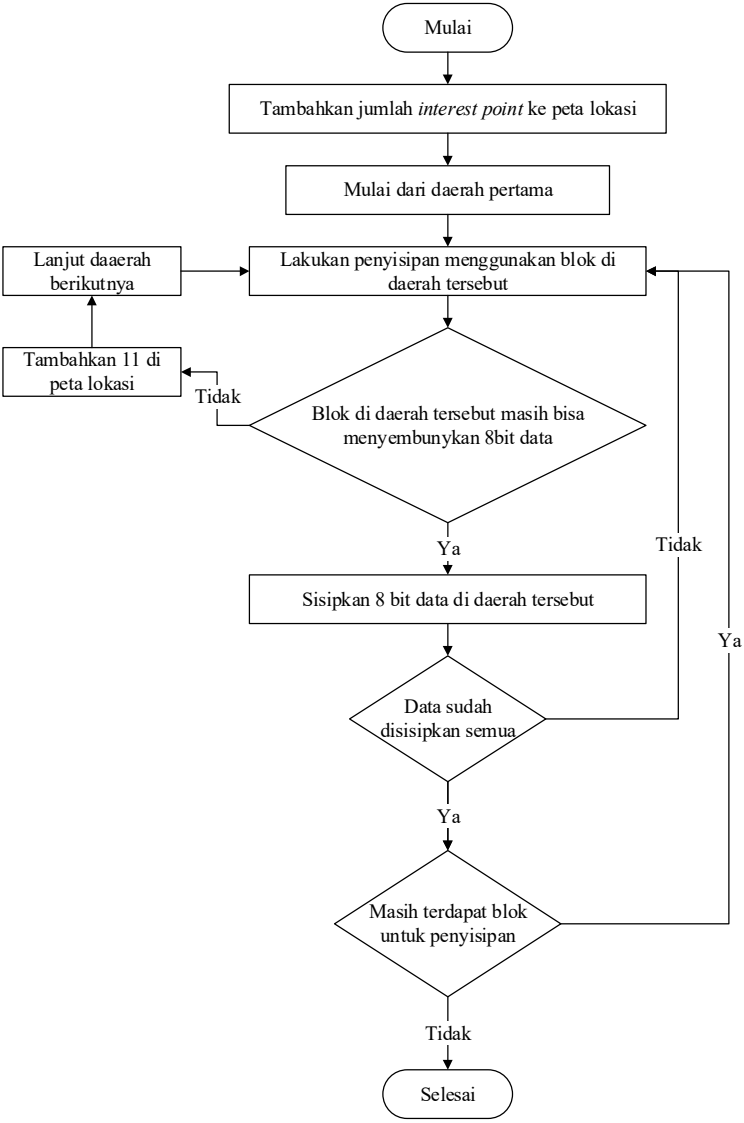
Selain citra hasil steganografi, terdapat juga *file* yang menyimpan nilai dari peta lokasi untuk setiap blok pada citra steganografi. Peta lokasi akan disimpan dalam *file* berekstensi txt. di mana peta lokasi hanya akan memiliki karakter 0 dan 1 serta tidak memiliki nilai posisi dari blok yang digunakan.

Hal ini menjadi keuntungan untuk proses RDE, di mana kita tidak perlu menyimpan nilai posisi dari blok yang digunakan. Hal ini dimungkinkan karena urutan dari blok-blok yang digunakan untuk menyimpan data didapatkan menggunakan metode SURF.

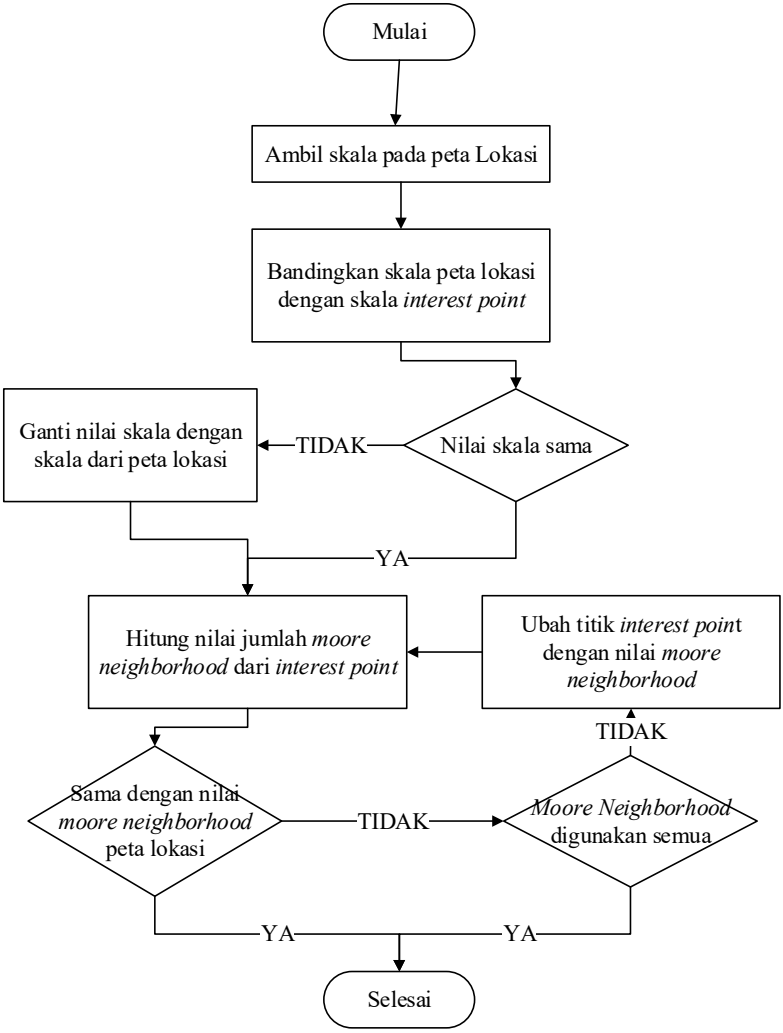
3.5.3 Validasi *Interest* Point

Ketika proses ekstraksi, *interest point* yang didapatkan dan dianggap valid akan divalidasi terlebih dahulu menggunakan informasi yang didapatkan dari peta lokasi. Informasi yang disimpan pada peta lokasi untuk setiap daerah seperti nilai jumlah dari *moore neighborhood* dan skala dari setiap *interest point* yang digunakan ketika proses penyisipan.

Nilai jumlah nilai *moore neighborhood* dan skala yang didapatkan dari peta lokasi akan dicocokkan dengan nilai *moore neighborhood* dan skala dari setiap *interest* point. Hal ini ditujukan untuk memvalidasi *interest point* yang terdeteksi. Untuk detail dari proses ini dapat dilihat pada Gambar 3-19.



Gambar 3-18 Diagram penyisipan data



Gambar 3-19 Diagram Alir Proses Validasi *Interest Point*

3.5.4 Perancangan Peta lokasi

Peta lokasi adalah nilai yang menyimpan keadaan sebuah blok ketika dilakukan penyisipan data. Peta lokasi digunakan ketika ekstraksi untuk menentukan sifat dari sebuah blok. Dalam RDE yang diusulkan oleh Lou dkk [4], peta lokasi menyimpan lokasi dan nilai dari keadaan tersebut. nilai yang diberikan adalah 0 atau 1. Nilai 0 atau 1 ditentukan berdasarkan Rumus (2-6). Selain menyimpan status dari titik, di metode yang diusulkan oleh Lou dkk [4], peta lokasi juga menyimpan nilai posisi dari blok yang digunakan untuk proses penyembunyian data.

Pada metode ini, dilakukan modifikasi terhadap peta lokasi, di mana peta lokasi hanya menyimpan status dari sebuah blok tanpa menyimpan posisi titik-titik pada blok tersebut. Titik tersebut bisa dihilangkan, karena SURF akan memberikan urutan titik yang sama ketika ekstraksi dan penyisipan data.

Peta lokasi juga digunakan sebagai penanda untuk menentukan daerah blok yang digunakan. Ketika peta lokasi memiliki nilai 11, maka proses ekstraksi dilanjutkan ke daerah blok berikutnya.

Status dari setiap blok ditentukan ketika penyisipan data, di mana setiap blok ditandai dengan dua *bit* data peta lokasi. Nilai dari peta lokasi yang digunakan pada metode yang diusulkan adalah seperti berikut.

- Jika peta lokasi memiliki nilai '00' maka blok tersebut tidak digunakan ketika penyisipan data.
- Jika peta lokasi memiliki nilai '01' maka blok tersebut mengalami perubahan atau digunakan untuk penyisipan data dan memenuhi persamaan $d' = d$ atau $2^{\lfloor \log_2 d' \rfloor} = 2^{\lfloor \log_2 d \rfloor}$.
- Jika peta lokasi memiliki nilai '10' maka blok tersebut mengalami perubahan atau digunakan untuk penyisipan data dan memenuhi persamaan $d' \neq d$.

Untuk proses ekstraksi data, peta lokasi akan dibentuk menjadi *array* 2 dimensi dengan ukuran M x N, di mana M adalah

panjang dari daerah yang digunakan dan *N* adalah banyak titik yang digunakan untuk menyimpan data sebanyak 8 *bit*.

Langkah pemrosesan peta lokasi untuk proses ekstraksi adalah

1. Kelompokkan peta lokasi setiap 2 *bit*, contoh dari peta lokasi 011100 menjadi 01, 11 dan 00.
2. Lakukan pemisahan untuk setiap peta lokasi yang memiliki nilai "11".
3. Peta lokasi nantinya akan membentuk *array* 2 dimensi.

3.5.5 Perancangan Ekstraksi *Bit*

Proses ekstraksi dilakukan dari blok terakhir di daerah yang paling terakhir disisipi data. Urutan blok tersebut ditentukan oleh peta lokasi. Peta lokasi untuk setiap daerah akan dibentuk terlebih dahulu untuk mempermudah proses ekstraksi. Peta lokasi akan dibentuk menjadi *array* 2 dimensi. Setiap indeks peta lokasi akan menentukan status untuk 8 *bit* data pada urutan tersebut.

Setelah peta lokasi dibentuk menjadi *array* 2 dimensi, lakukan proses ekstraksi dari blok terakhir yang ditunjukkan oleh peta lokasi. Sebagai contoh, jika peta lokasi mempunyai 10 data, maka terdapat 10 daerah yang akan digunakan untuk proses ekstraksi. Proses ekstraksi dimulai dari daerah ke-10. Jika pada peta lokasi ke -10, peta lokasi tersebut memiliki 20 data, maka proses ekstraksi dimulai blok ke-20 pada daerah tersebut hingga blok pertama.

Proses ekstraksi dilakukan dari blok terakhir daerah terakhir hingga blok pertama daerah pertama. Hal ini dikarenakan titik-titik yang digunakan untuk setiap blok saling *overlap*, sehingga ekstraksi dan penyusunan data dimulai dari blok paling terakhir yang disisipi data.

Ilustrasi untuk perubahan dan ekstraksi kembali dari beberapa blok dapat dilihat pada Gambar 3-20 dan Gambar 3-21

A	B	C
A'	B'	C
A'	B''	C'

Gambar 3-20 Proses Penyisipan Untuk Titik-Titik yang *Overlap* Pada Beberapa Blok

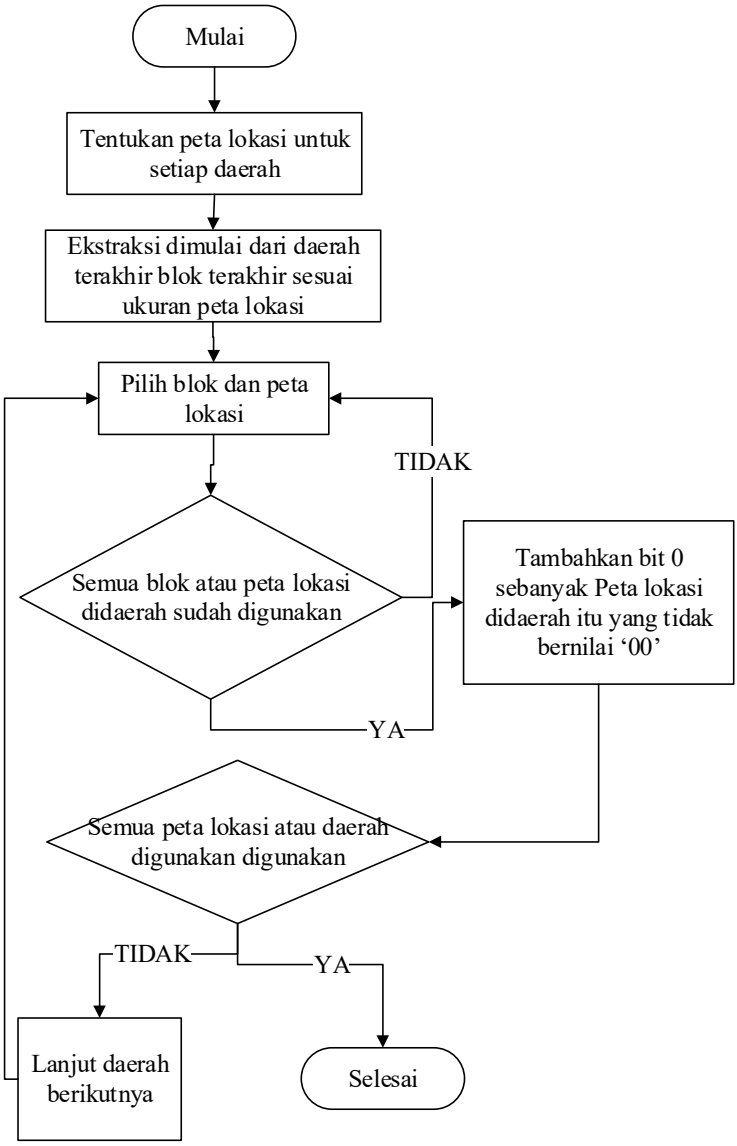
A’	B’’	C’
A’	B’	C
A	B	C

Gambar 3-21 Proses Ekstraksi Untuk Titik-Titik yang *Overlap* Pada Beberapa Blok

Langkah-langkah untuk proses ekstraksi pada metode ini adalah seperti berikut.

1. Lakukan ekstraksi pada daerah terakhir menggunakan peta lokasi pada blok yang ditunjukkan oleh peta lokasi tersebut.
2. Lakukan ekstraksi pada setiap blok pada daerah tersebut sesuai dengan status yang diberikan oleh peta lokasi
3. Lanjutkan ekstraksi data pada daerah tersebut sebanyak peta lokasi.
4. Gabungkan data *bit* hasil ekstraksi dengan *bit* data sebelumnya
5. Jika blok pada daerah tersebut sudah digunakan semua untuk menyisipkan data, namun masih terdapat peta lokasi yang belum digunakan, tambah *bit* 0 atau 1 sebanyak peta lokasi yang memiliki nilai “01” atau “10”.
6. Ulangi proses 2 hingga 4 sampai semua peta lokasi digunakan

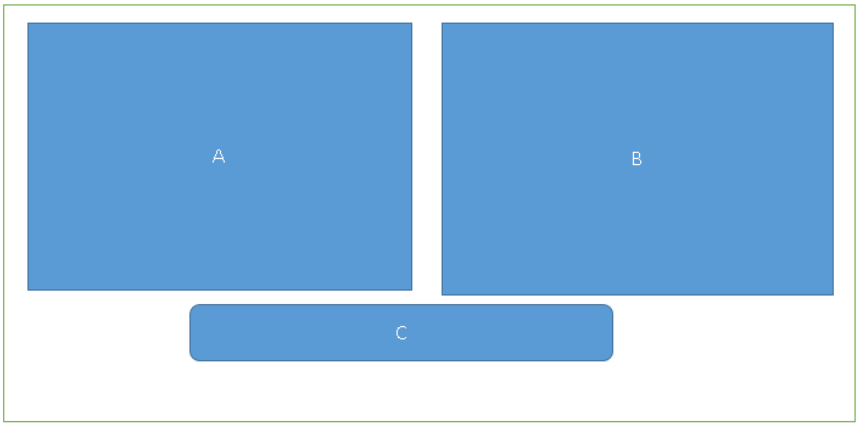
Untuk memperjelas mengenai proses ekstraksi data dapat dilihat pada Gambar 3-22.



Gambar 3-22 Diagram Alir Proses Ekstraksi Data

3.6 Perancangan Antarmuka

Program antarmuka digunakan untuk mempermudah pengguna dalam melakukan uji coba menggunakan metode yang diusulkan. Antarmuka dibuat menggunakan bahasa pemrograman HTML dan Javascript. Aplikasi nantinya akan bisa diakses melalui *web browser*.



Gambar 3-23 Rancang Bangun Antarmuka Aplikasi

Komponen yang terdapat dalam Gambar 3-23 antara lain:

A. *Form* untuk menyisipkan data menggunakan metode yang diusulkan. Pada *form* ini terdapat beberapa atribut yang harus dimasukkan, yaitu:
   1. Citra yang akan digunakan sebagai media penyimpanan.
   2. Tipe data yang akan disisipkan. Tipe data yang disediakan adalah teks atau *file*.
   3. Data yang akan disisipkan.
   4. *Threshold*, *layer*, *filter* untuk pendeteksian interest point.
   5. Selisih maksimum dan *chebyshev distance* maksimum yang digunakan untuk proses pembentukan daerah.

B. *Form* yang digunakan untuk mengekstraksi data menggunakan metode yang diusulkan. Pada *form* ini terdapat beberapa atribut yang harus dimasukkan yaitu:
C. Untuk proses penyisipan data, bagian ini berfungsi untuk menampilkan citra hasil modifikasi dan tombol untuk mengunduh peta lokasi dan citra hasil modifikasi. Sedangkan untuk proses ekstraksi data, bagian ini menampilkan data hasil ekstraksi atau tombol untuk mengunduh *file* data yang disembunyikan.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya. Sebelum penjelasan implementasi akan ditunjukkan terlebih dahulu lingkungan untuk melakukan implementasi..

4.1 Lingkungan Implementasi

Dalam implementasi metode yang diusulkan, digunakan perangkat-perangkat sebagai berikut:

4.1.1 Perangkat Keras

Spesifikasi dari perangkat keras yang digunakan adalah sebagai berikut:

6. Processor 3.3GHz Inter Core i3-2120
7. RAM sebesar 4GB
8. Ruang penyimpanan sebesar 200GB

4.1.2 Perangkat Lunak

1. Windows 10 pro 64 *bit*
2. Python 2.7
3. PyCharm Community Edition 5.0.4

4.2 Implementasi Kelas *Interest Point*

Kelas *interest point* digunakan sebagai objek yang nantinya akan menampung segala fitur dan atribut dari *interest point* yang nantinya akan digunakan. detail dari kelas *interest point* dapat dilihat pada *Pseudocode* 4-1 Kelas Ipoint

Kelas *interest point* nantinya tidak hanya untuk menampung objek dari *interest point* yang dideteksi, namun kelas ini digunakan untuk inisiasi semua titik yang terdapat pada sebuah *interest point*.

```
1. class IPoint ()
2.    Integer x
3.    Integer y
4.    Real scale
5.    Real respons
6.    Real orientation
7.    Integer posisi
```

Pseudocode 4-1 Kelas Ipoint

4.2.1 Detail Atribut kelas Ipoint

Untuk detail dari setiap atribut dari kelas Ipoint dapat dilihat pada Tabel 4-1.

Tabel 4-1 Atribut Kelas Ipoint

Nama Atribut	Keterangan
x	Nilai axis dari titik yang digunakan
y	Nilai absis dari titik yang digunakan
Scale	Fitur skala dari titik yang digunakan
Respons	Nilai yang merepresentasikan kekuatan dari yang digunakan
Posisi	Nilai yang merepresentasikan jumlah data yang bisa disisipi pada daerah ini hingga titik ini
Orientation	Nilai yang merepresentasikan nilai orientasi atau arah dari *interest point* tersebut

4.3 Implementasi Fungsi Pendeteksi *Interest Point*

Dalam pendeteksian *interest point* dilakukan pembuatan fungsi pendeteksian *interest point* dan menghilangkan *interest point* yang memiliki irisan daerah.

Fungsi ini digunakan untuk mencari *interest point* yang memiliki nilai respons lebih besar dari *threshold* yang diberikan. Pendeteksian *interest point* menggunakan *library* Open CV versi 2.4. Masukan dari fungsi ini adalah citra yang akan diproses dan

threshold sebagai nilai batasan yang akan digunakan untuk mengecek kevalidan *interest point* tersebut.

Masukan	Citra cover, *threshold*
Keluaran	*Interest Point*

```
1.  surf=cv2.SURF(thresh,4,2)
2.  interest_point, des =
    surf.detectAndCompute(self.img,None)
3.  return kp
```

Kode Sumber 4-1 Pendeteksian *Interest Point* Menggunakan Open CV

4.4 Implementasi Pembentukan Blok

Blok yang dibentuk akan digunakan untuk menyisipkan data pada proses RDE. Setiap blok memiliki dua buah titik yang akan digunakan untuk penyisipan 1 *bit* data. Pembentukan blok diawali dengan pembentukan daerah di setiap *interest point*, mengeliminasi daerah yang beririsan dan pembentukan blok pada setiap daerah.

4.4.1 Implementasi Eliminasi Daerah yang Beririsan

Interest point yang digunakan adalah *interest point* yang saling tidak beririsan. Hal ini dimaksudkan agar perubahan pada sebuah daerah *interest point* tidak akan mempengaruhi nilai respons dari *interest point* di daerah lainnya.

Untuk menentukan apakah dua daerah dari *interest point* saling beririsan digunakan nilai *euclidean* dari dua titik *interest point* tersebut. Jika nilai *euclidean* memenuhi Persamaan (3-3) maka dua daerah dari *interest point* tersebut tidak saling beririsan.

Implementasi dari fungsi untuk menentukan nilai dari *euclidean* antara dua titik *interest point* dapat dilihat pada *Pseudocode* 4-2

Masukan	*Interest point* 1, *interest point* 2
Keluaran	Nilai *Euclidean*, status beririsan atau tidak

```
1. point_a_x <- posisi interest point 1 (x)
2. point_b_x <- posisi interest point 2 (x)
3. point_a_y <- posisi interest point 1 (y)
4. point_b_y <- posisi interest point 2 (y)
5. r_a < - skala interest point 1
6. r_b < - skala interest point 2
7. Euc < - (point_a_x – point_b_x)^2 + (point_b_y-
   point_a_y)^2
8. Jarak <- (r_a+r_b)^2
9. If Euc >= jarak THEN
10.  Return True,euc
11. Else THEN
12.  Return false, euc
13. ENDIF
```

Pseudocode 4-2 Impelementasi Menentukan Nilai *Euclidean*

Jika terdapat daerah yang beririsan, akan diambil daerah dengan nilai respons tertinggi. Untuk pemilihan daerah yang tidak beririsan dapat dilihat pada

Masukan	List *interest point*
Keluaran	*Interest point* yang tidak beririsan

```
1.  Ipts <- list interest point
2.  Point <- []
3.  Flag_penuh <- list Null
4.  For x=0 to LEN(ipts) do
5.    If flag_penuh[x] not NULL THEN
6.      CONTINUE NEXT LOOP
7.    For y = x+1 to LEN(ipts) do
8.      If Euclidean(ipts[x], ipts[y]) is FALSE THEN
9.        CONTINUE NEXT LOOP
10.     Else
11.       Flag_penuh[y] = 1
12.   ENDIF
13.  ENDFOR
14.  Flag_penuh[x] = 1
15.  Add ipts[x] to point list
16. ENDFOR
17. Return point
```

Pseudocode 4-3 Mengeliminasi Titik yang Beririsan

4.4.2 Implementasi Pembentukan titik-titik di Daerah dari Setiap *Interest Point*

Fungsi ini digunakan untuk membentuk daerah yang akan digunakan untuk penyisipan data. Luas daerah yang terbentuk dipengaruhi oleh nilai skala *s* dari titik *interest point* tersebut. Luas daerah *L* dari sebuah *interest point* ditentukan Rumus (3-1). Pembentukan daerah juga dipengaruhi oleh orientasi dari *interest point*, karena orientasi menentukan arah pembentukan daerah dari sebuah titik *interest point*.

Ada delapan kemungkinan orientasi, hal ini dikarenakan setiap rotasi terdapat empat kemungkinan orientasi. Karena setiap citra terdapat kemungkinan bisa dilakukan pencerminan, sehingga citra hasil pencerminan juga memiliki empat kemungkinan rotasi, sehingga total kemungkinan rotasi ada 8, dan setiap kemungkinan memiliki nilai rentang 45^0. Pada awal pembentukan daerah, ditentukan dulu nilai dari setiap titik sudut pada daerah tersebut.

```
1. kp <- interest point
2. image_center <- kp.pt
3. orientation <- kp.angle
4. length <- kp.size
5. img <- citra
6. top=max(int(image_center[1]-length),0)
7. left=max(int(image_center[0]-length),0)
8. right=min(int(image_center[0]+length),len(img[0])
9. bottom=min(int(image_center[1]+length),len(img))
10. point <- []
```

Pseudocode 4-4 Penentuan Sudut Untuk Pembentukan Daerah

Pemilihan titik seperti yang dijelaskan sebelumnya adalah menggunakan pertimbangan nilai orientasi untuk mempertahankan nilai ketika dilakukan rotasi dan pencerminan. Untuk aturan urutan pemilihan titik dapat dilihat pada *Pseudocode* 4-5 hingga *Pseudocode* 4-12

```
1.  for y=bottom to top do
2.    for x = right to left do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

Pseudocode 4-5 Urutan Pemilihan Titik Orientasi 0^0 – 44^0

```
1.  for x=right to left do
2.    for y = bottom to top do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

Pseudocode 4-6 Urutan Pemilihan Titik Orientasi 45^0 – 89^0

```
1.  for x = left to right do
2.    for y=bottom to top do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

Pseudocode 4-7 Urutan Pemilihan Titik Orientasi 90^0 – 134^0

```
1.  for y = bottom to top do
2.    for x = left to right do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

Pseudocode 4-8 Urutan Pemilihan Titik Orientasi 135^0 – 179^0

```
1.  for y = top to bottom do
2.    for x = left to right do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

Pseudocode 4-9 Urutan Pemilihan Titik Orientasi 180^0 – 224^0

```
1.  for x = left to right do
2.    for y = top to bottom do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

Pseudocode 4-10 Urutan Pemilihan Titik Orientasi 225^0 – 269^0

```
1.  for x = right to left do
2.    for y = top to bottom do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

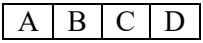
Pseudocode 4-11 Urutan Pemilihan Titik Orientasi 270^0 – 314^0

```
1.  for y = top to bottom do
2.    for x = right to left do
3.      if point (x,y) notused before THEN
4.        Insert point(x,y) to list
5.       ENDIF
6.     ENDFOR
7.  ENDFOR
```

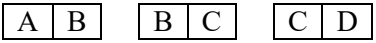
Pseudocode 4-12 Urutan Pemilihan Titik Orientasi 315^0 – 359^0

Setelah semua titik pada daerah tersebut dimasukkan ke dalam *list* yang menampung titik tersebut, dari titik-titik tersebut akan dibentuk blok-blok di mana setiap blok berukuran memiliki dua titik dan setiap blok memiliki titik yang *overlap* terhadap blok sebelahnya.

Semisal terdapat empat buah titik dengan urutan seperti Gambar 4-1 maka akan dihasilkan 3 buah blok dengan setiap blok memiliki dua titik seperti ditunjukkan Gambar 4-2. *Overlapping* memungkinkan untuk dilakukan karena dengan menggunakan metode RDE kita bisa mendapatkan kembali titik yang sudah dimodifikasi seperti berikut.



Gambar 4-1 Contoh Urutan Titik Pada *List*



Gambar 4-2 Urutan Blok yang dihasilkan

4.4.3 Implementasi Fungsi Pembentukan Blok pada Suatu Daerah

Fungsi ini digunakan untuk membentuk blok yang akan digunakan untuk proses steganografi. Setiap blok terdiri dari 2 piksel yang bersebelahan atau berurutan dari setiap daerah. Setiap blok akan ditentukan urutan berdasarkan pembentukan daerah di blok tersebut. Setiap blok akan mendapatkan nomor urutan yang merepresentasikan banyak data yang sudah tersimpan pada daerah tersebut mulai dari blok pertama hingga blok tersebut. Nomor urutan ini berfungsi agar data yang tersimpan pada setiap blok tersebut adalah 1 *byte*, ini digunakan untuk mengurangi efek kerusakan data pada saat proses pengembalian.

Untuk setiap titik-titik yang terdapat dalam blok, akan mengalami tumpang tindih dengan blok sebelahnya, karena setiap titik akan digunakan untuk 2 blok. Untuk implementasi dari fungsi ini dapat dilihat pada *Pseudocode* 4-13

Masukan	*List* titik pada daerah tersebut
Keluaran	*List* blok pada daerah tersebut

```
1. listTitik <- list titik di daerah tersebut
2. blok <- []
3. for i = 0 to LEN(listTitik)-1
4.    temp <- []
5.    for j = 0 to 2
6.      ipoint=IPoint()
7.      ipoint.x=listTitik[i+j][0]
8.      ipoint.y= listTitik[i+j][1]
9.    ENDOF
10.   blok[LEN(blok)] = temp
11. ENDFOR
12. Return blok
```

Pseudocode 4-13 Pembentukan Blok Untuk Setiap Daerah

4.5 Implementasi RDE

Pada sup bab ini akan dijelaskan mengenai implementasi dari algoritma RDE yang digunakan. Terdapat dua fungsi yaitu penyisipan *bit* ke dalam setiap blok dan fungsi ekstraksi data serta pengembalian nilai piksel seperti semula.

4.5.1 Implementasi Penyisipan Data RDE

Implementasi penyisipan 1 *bit* data pada 1 blok menggunakan RDE dapat dilihat pada *Pseudocode* 4-14.

Masukan	*Bit* yang akan disisipkan, piksel 1, piksel 2
Keluaran	Piksel 1 hasil modifikasi, piksel 2 hasil modifikasi, peta lokasi

```
1. bitEmbed <- input bit
2. x <- piksel 1
3. y <- piksel 2
4. average <-floor((x+y)/2)
5. difference <- abs(x-y)
6. differenceReduced <- difference
7. IF difference <2 THEN
```

```
8.      differenceReduced = difference
9.  ELSE
10.     differenceReduced =
    difference+2^(floor(log2(abs(difference)))-1)
11. ENDIF
12. newDifference <- differenceReduced*2+bitEmbed
13. IF ((average>=0 and average<=127) and
    (newDifference<=2*average+1)) or ( (average>=128
    and average<=255) and (newDifference<=2*(255-
    average) ))
14.     IF log2(differenceReduced) ==
    log2(difference)
15.             petaLokasi <- 01
16.     ELSE
17.             petaLokasi <- 10
18.     ENDIF
19.     IF x<y:
20.             x = average - floor(newDifference/2)
21.             y = average +
    floor((newDifference+1)/2)
22.     ELSE
23.             x = average + floor(newDifference/2)
24.             y = average - floor((newDifference+1
25.     ENDIF
26. ELSE
27.     petaLokasi = 00
28. ENDIF
29. Return(x,y,petaLokasi)
```

Pseudocode 4-14 Penyisipan Data Menggunakan RDE

4.5.2 Implementasi Fungsi Penyisipan Pesan

Fungsi ini digunakan untuk menyisipkan data ke dalam citra *cover* yang akan digunakan sebagai media penyimpanan. Implementasi untuk proses ini dapat dilihat pada *Pseudocode* 4-15.

Masukan	Data string yang akan disisipkan, citra cover, blok setiap daerah penyisipan data
Keluaran	Citra hasil modifikasi, petaLokasi

```
1. daerahBlok <- daerah penyisipan
2. petaLokasi <- ''
3. data <- string binary inputan
4. img <- citra cover
5. for k =0 to LEN(daerahBlok) do
6.   petaLokasi <- petaLokasi + '11'
7.   daerah = daerahBlok[k]
8.     for l =0 to LEN(daerah) do
9.       blok = daerah[l]
10.      if LEN(data)%8==0 then
11.        if daerah bisa menyimpan 8 bit data then
12.          pass
13.        else
14.          break
15.        ENDIF
16.      ENDIF
17.      if LEN(data)==0 then
18.        break
19.      ENDIF
20.      b=data[0]
21.      x = img[blok[0].y][blok[0].y]
22.      y = img[blok[1].y][blok[1].y]
23.      x,y,lm,status= Penyisipan data
         RDE(x,y,b)
24.      if status then
25.        petaLokasi=petaLokasi + lm
26.        img[blok[0].y][blok[0].y] = x
27.        img[blok[1].y][blok[1].y] = y
28.      ENDIF
29.    ENDFOR
30. ENDFOR
31. return img, petaLokasi
```

Pseudocode 4-15 Implementasi penyisipan keseluruhan pesan

4.5.3 Implementasi Fungsi Ekstraksi Data RDE

Fungsi ini digunakan untuk melakukan ekstraksi 1 *bit* data pada 1 blok dan mengembalikan nilai piksel yang sudah diubah

kembali seperti semula. Implementasi untuk fungsi ini dapat dilihat pada *Pseudocode* 4-16.

Masukan	Peta lokasi, piksel 1, piksel 2
Keluaran	Piksel 1 awal, piksel 2 awal, data *bit* yang disisipkan

```
1.  x <- piksel 1
2.  y <- piksel 2
3.  locationMap <- peta lokasi
4.  average <- floor((x+y)/2)
5.  difference <- abs(x-y)
6.  embedBit <- mod(difference,2)
7.  differenceReduced <- floor(difference/2)
8.  if locationMap == "00" THEN
9.      return False
10. ENDIF
11. If differenceReduced > 1 THEN
12.     If locationMap == "01" THEN
13.             differenceReduced = differenceReduced
    + 2^(2log differenceReduced - 1)
14.     Else if locationMap == "10"
15.             differenceReduced = differenceReduced
    + 2^(2log differenceReduced)
16.     ENDIF
17. ENDIF
18. if x<y THEN
19.     x = m - floor(differenceReduced/2)
20.     y = m + floor((differenceReduced+1)/2)
21. else
22.     x = m + floor((differenceReduced+1)/2)
23.     y = m - floor(differenceReduced/2)
24. ENDIF
25. return (x,y,embedBit)
```

Pseudocode 4-16 Ekstraksi Data Menggunakan RDE

4.5.4 Implementasi Menentukan Jumlah Nilai *Moore Neighborhood* dari *Interest Point*

Fungsi ini digunakan untuk menghitung *moore neighborhood* yang akan digunakan untuk validasi. Implementasi dari fungsi ini dapat dilihat pada *Pseudocode* 4-17.

Masukan	Titik tengah dari *Moore Neighborhood*, citra
Keluaran	Jumlah nilai dari *Moore Neighborhood*

```
1. center <- titik tengah (x,y)
2. img <- citra
3. total = 0
4. nilaiCenter = img[center.y][center.x]
5. for i = -1 to 1 DO
6.    for j = -1 to 1 DO
7.      if (i,j) != center THEN
8.        total=total+ (tengah -   img[center.y +
          i][center.x + j])
9.      ENDIF
10.   ENDFOR
11. ENDFOR
12. return total
```

Pseudocode 4-17 Menentukan Jumlah Nilai *Moore Neighborhood*

4.5.5 Implementasi Validasi *Interest* Point Ketika Ekstraksi

Fungsi ini digunakan untuk memvalidasi *interest point* yang diperoleh ketika proses ekstraksi. Implementasi dari fungsi ini dapat dilihat pada *Pseudocode* 4-18.

Masukan	List *interest point*, list nilai jumlah *moore neighborhood* dan list nilai skala dari peta lokasi
Keluaran	*Interest* point yang sudah divalidasi

```
1. Ipts <- list interest point
2. mooreNeigh <- list jumlah moore neighborhood
3. listSkala <- list nilai skala
```

```
4.  for x=0 to LEN(ipts) DO
5.    center = ipts[x].center
6.    nilaiNeigh <- neighboor(center)
7.    If nilaiNeigh != mooreNeigh[x] THEN
8.      for i=-1 to 1 DO
9.        for j=-1 to 1 DO
10.         nilaiNeigh <- neighboor((center.x +
            i, center.y +j))
11.          if nilaiNeigh == mooreNeigh[x] THEN
12.            ipts[x].center = ((center.x+i,
               center.y + j))
13.            break
14.          ENDIF
15.        ENDFOR
16.     ENDFOR
17.   ENDIF
18. ENDFOR
19. return ipts
```

Pseudocode 4-18 Validasi *Interest Point*

4.5.6 Implementasi Ekstraksi Pesan

Fungsi ini digunakan untuk mendapatkan data utuh dari citra steganografi dan peta lokasi yang diberikan. Implementasi untuk fungsi dari penerapan metode ini dapat dilihat pada *Pseudocode* 4-19.

Masukan	Citra steganografi, peta lokasi, blok setiap daerah
Keluaran	Pesan

```
1.  petaLokasi <- peta lokasi
2.  listDaerah <- daerah mengandung blok yang akan
    diembed
3.  img <- citra steganografi
4.  pesan <- ''
5.  for i = LEN(petaLokasi)-1 to 0 do
6.    petaDaerah = petaLokasi[i]
7.    Daerah = listDaerah[i]
8.    for j = len(petaDaerah) -1 to 0 do
9.      if j >= LEN(daerah) then
10.       if petaDaerah[j]!='00' THEN
```

```
11.          hasil='0'+hasil
12.         ENDIF
13.         continue next loop
14.      ENDIF
15.    ENDFOR
16.    x = nilai piksel untuk titik blok 1
17.    y = nilai piksel untuk titik blok 1
18.    Ekstrak data menggunakan RDE berdasarkan
        PetaDaerah[j]
19.    pesan = hasil ekstraksi RDE + pesan
20. ENDFOR
21. pesanString <- ubah pesan ke input string
22. simpan img awal
```

Pseudocode 4-19 Ekstraksi Pesan Secara Kesuluruhan

4.5.7 Implementasi StringTo*Bit*

Fungsi ini digunakan untuk mengubah inputan data dari string menjadi *bit*. Kode untuk proses ini dapat dilihat pada Kode Sumber 4-2.

Masukan	Data string yang akan disembunyikan
Keluaran	*Bit binary* dari string yang diinput

```
1. binary=' '.join('{0:08b}'.format(ord(x), 'b')
   for x in string)
2.     binary=binary.replace(' ','',)
3. Return binary
```

Kode Sumber 4-2 Implementasi Mengubah *String* Menjadi Binari

4.5.8 Implementasi *Bit*ToString

Fungsi ini digunakan untuk mengubah hasil ekstraksi data yang berupa *bit* menjadi *string* sehingga hasil ekstraksi bisa dibaca oleh pengguna. Implementasi untuk proses ini dapat dilihat pada *Pseudocode* 4-20.

Masukan	*Bit string* hasil ekstraksi RDE
Keluaran	*String* data yang disisipi

```
1. x <- 0
2. string <- NULL
3. while x < LEN(bit)
4.    bitString <- bit[x:x+8]
5.    ascii_bitstring <- ascii(bitString)
6.    string <- string + character(ascii_bitstring)
7.    x <- x+8
8. ENDWHILE
9. return string
```

Pseudocode 4-20 Mengubah Data Binari Menjadi Data *String*

4.5.9 Implementasi fungsi PSNR

PSNR digunakan untuk mengevaluasi kualitas citra yang telah disisipi data menggunakan RDE. Implementasi fungsi menentukan PSNR dapat dilihat pada *Pseudocode* 4-21.

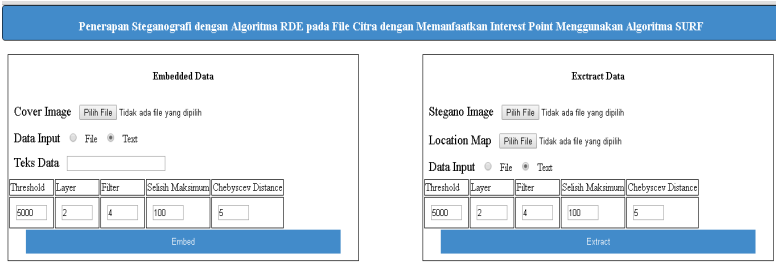
Masukan	Citra *cover* awal dan citra *cover* setelah disisipi data
Keluaran	PSNR

```
1.  img <- citra awal
2.  img_stegano <- citra modifikasi
3.  selisih <- 0
4.  for y <- 0 to LEN(img)
5.      for x <- 0 to LEN(img[0])
6.            beda_piksel= (img[y][x] -
    img_stegano[y][x]
7.      beda_piksel <- beda_piksel*beda_piksel
8.            selisih <- selisih+beda_piksel
9.     ENDFOR
10. ENDFOR
11. MSE <- selisih/LEN(img)/LEN(img[0])
12. PSNR <- 10*log(255*255/MSE)
13. return PSNR
```

Pseudocode 4-21 Menentukan Nilai PSNR

4.6 Implementasi Antarmuka

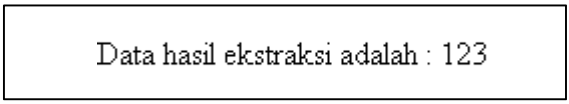
Antarmuka diimplementasikan menggunakan Python CGI yang sudah dijelaskan pada sub bab 2.10. Antarmuka dibuat menggunakan *form* HTML dan Javascript. HTML dan Javascript serta *header respons* untuk pengguna yang mengakses adalah keluaran dari program Python yang dipanggil. Untuk implementasi antarmuka utama dapat dilihat pada Gambar 4-3.



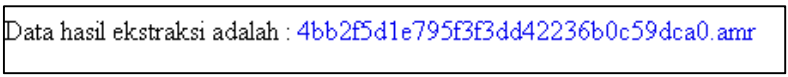
Gambar 4-3 Implementasi Antarmuka Utama

Gambar 4-4 Antarmuka Ketika Proses Penyisipan Data

Pada Gambar 4-3, terdapat 2 *form* yang digunakan untuk ekstraksi dan penyisipan data. Antarmuka ini adalah implementasi dari perancangan yang sudah dijelaskan pada sub bab 3.6. Ketika proses penyisipan data dilakukan, hasil dari proses tersebut dapat dilihat pada Gambar 4-4. Sedangkan antarmuka untuk hasil proses ekstraksi data dapat dilihat pada Gambar 4-5 dan Gambar 4-6. Pada Gambar 4-5 data yang disembunyikan pada citra langsung ditampilkan di aplikasi. Namun pada Gambar 4-6 data yang disembunyikan adalah berupa *file*, sehingga pengguna harus mengunduh *file* tersebut terlebih dahulu.



Gambar 4-5 Antarmuka Hasil Ekstraksi Untuk Data Teks



Gambar 4-6 Antarmuka Hasil Ekstraksi Untuk Data *File*

BAB V
PENGUJIAN DAN EVALUASI

Pada bab ini akan dijelaskan uji coba yang dilakukan pada aplikasi yang telah dikerjakan serta analisa dari uji coba yang telah dilakukan. Pembahasan pengujian meliputi lingkungan uji coba, skenario uji coba yang meliputi uji kebenaran dan uji kinerja serta analisa setiap pengujian.

5.1 Lingkungan Uji Coba

Lingkungan uji coba menjelaskan lingkungan yang digunakan untuk menguji program penyisipan steganografi menggunakan algoritma RDE. Lingkungan uji coba meliputi perangkat keras dan perangkat lunak yang dijelaskan sebagai berikut:

1. Perangkat keras
   a. Processor 3.3GHz Inter Core i3-2120
   b. RAM sebesar 4GB
   c. Ruang penyimpanan sebesar 200GB.
2. Perangkat lunak
   a. Windows 10 pro 64 *bit*
   b. Python 2.7
   c. PyCharm Community Edition 5.0.4
   d. Xampp versi 3.2.2

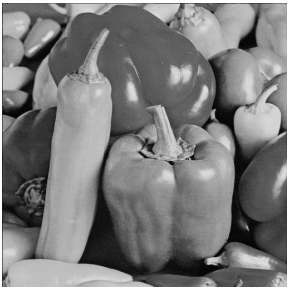
5.2 Dataset Uji Coba

Terdapat enam buah citra [15] yang digunakan sebagai citra *cover* dalam proses uji coba. Kelima citra tersebut dapat dilihat pada Gambar 5-1 Girlface sampai dengan Gambar 5-5 Cablecar. Untuk detail dari setiap citra dapat dilihat Tabel 5-1. Dataset citra yang digunakan adalah *file* citra yang mempunyai ekstensi bmp. Citra yang digunakan adalah citra dengan tipe warna keabuan. Jika pengguna memasukkan citra bertipe warna lain, maka akan diubah terlebih dahulu ke dalam bentuk citra keabuan.

Gambar 5-1 Citra Girlface.bmp

Gambar 5-2 Citra Girl.bmp

Gambar 5-3 Citra Barbara.bmp



Gambar 5-4 Pepper.bmp

Gambar 5-5 Citra cablecar.bmp

Tabel 5-1 Detail Dataset Citra *Cover*

Nama File	Ukuran (kB)	Dimensi
Girlface.bmp	257	512 x 512
Girl.bmp	406	720 x 576
Barbara.bmp	406	720 x 576
Pepper.bmp	257	512 x 512
Cablecar.bmp	241	512 x 480

Untuk data rahasia yang disisipkan adalah berupa *file* citra dengan format pgm dan data audio dengan format amr serta wav. *File* audio yang digunakan adalah *file* hasil rekaman menggunakan *Smartphone*. Data rahasia [16] yang akan disisipkan adalah *file* multimedia dengan detail seperti pada Tabel 5-2. Data rahasia terdiri dari *file* citra dengan ekstensi pgm dan file audio dengan

berbagai ekstensi. Untuk data rahasia citra yang disisipkan dapat dilihat seperti pada Gambar 5-6 [17].

Gambar 5-6 Citra Inputan

Tabel 5-2 Detail Data yang Akan Disisipkan

Nama File	Ekstensi	Ukuran (kB)
Citra.pgm	.pgm	4.01
Recorder.amr	.amr	4.81
Shortwave.wav	.wav	2.50
Alarm_Antelope	.aac	5.03
test9_v1_Poly_8chan_9track_8polys	.mid	2.97
test5_v1_Poly_1chan_7track_TextInc_GNMIDI-GM2	.mid	1.44
AMRaudio1	.amr	1.73
AMRaudio2	.amr	3.29
test10_v0_Mono_Mono	.mid	0.42

5.3 Skenario Uji Coba

Skenario uji coba diperlukan untuk menguji kebenaran dari metode yang diusulkan. Skenario uji coba juga dapat digunakan untuk menguji performa dari metode yang diusulkan. Pada uji coba terdapat beberapa skenario yang digunakan untuk menguji kebenaran fungsionalitas dari metode yang diusulkan dan untuk menguji performa dari metode yang diusulkan. Untuk detail dari skenario pengujian dijelaskan pada sub bab berikutnya. Pada uji coba parameter yang diukur adalah BER (*bir error rate*) yang merepresentasikan persentase kegagalan data yang di ekstraksi seperti yang sudah dijelaskan pada sub bab 2.8.

5.3.1 Skenario Uji Coba Fungsionalitas

Uji coba dilakukan untuk menguji kebenaran dari implementasi aplikasi dan metode yang diusulkan pada Bab 4. Uji fungsionalitas meliputi dua buah proses pada steganografi, yaitu pengujian pada proses penyisipan data dan proses ekstraksi data.

Pada uji coba, digunakan parameter *bit eror rate* yang dijelaskan pada sub bab 2.8 sebagai parameter untuk mengevaluasi performa dari metode yang diusulkan.

Namun setiap citra *cover* memiliki kapasitas yang berbeda. Sehingga ada kemungkinan tidak semua data berhasil disisipkan ke dalam citra *cover*. Jika pesan yang akan disisipkan melebihi kapasitas dari citra *cover* maka pesan hanya akan disisipkan sebanyak kapasitas dari *cover* tersebut.

Pada uji coba, data yang disisipkan yaitu Gambar 5-6 akan disisipkan pada setiap citra *cover*. Citra *cover* yang sudah disisipi nantinya akan dilakukan perubahan. Perubahan yang diperbolehkan adalah, rotasi 90^0, rotasi 180^0, rotasi 270^0 , pencerminan terhadap sumbu *x*, pencerminan terhadap sumbu *x* dan *blur* dengan skala *Gaussian* 1. Setiap citra *cover* yang sudah dimodifikasi, akan dilakukan perubahan seperti yang disebutkan dan dilakukan ekstraksi data.

5.3.2 Skenario Uji Coba Performa

Uji coba performa dilakukan untuk membandingkan efektivitas dan performa dari penyusunan blok yang dilakukan. Metode yang dibandingkan adalah dengan penyusunan blok secara *overlap* dan tidak *overlap*. Pada uji coba performa juga dilakukan penyisipan sebanyak kapasitas maksimal dari sebuah citra. Data yang disisipkan adalah *string* random sepanjang *X* data.

5.4 Uji Coba

Dalam uji coba akan dilakukan uji coba fungsionalitas dan uji coba performa.

5.4.1 Uji Coba Fungsionalitas

Hasil uji coba fungsionalitas dapat dilihat pada Tabel 5-3. Data parameter untuk uji coba dapat dilihat pada Tabel 5-3.

Tabel 5-3 Hasil Uji Coba Fungsionalitas

Cover	Perubahan	PSNR (dB)	BER (%)
pepper.bmp	-	46.47	1.36
	Rotasi 90^0		1.36
	Rotasi 180^0		1.36
	Rotasi 270^0		1.36
	Flip Horizontal		1.36
	Flip Vertikal		1.36
	Blur		71.4
girl.bmp	-	50.56	0.34
	Rotasi 90^0		0.34
	Rotasi 180^0		0.34
	Rotasi 270^0		0.34
	Flip Horizontal		0.34
	Flip Vertikal		0.34
	Blur		78.13
barbara.bmp	-	45.15	22.8
	Rotasi 90^0		22.8
	Rotasi 180^0		22.8
	Rotasi 270^0		22.8
	Flip Horizontal		22.8
	Flip Vertikal		22.8
	Blur		36.20
cablecar.bmp	-	44.29	14.11
	Rotasi 90^0		14.11
	Rotasi 180^0		14.11
	Rotasi 270^0		14.11
	Flip Horizontal		14.11
	Flip Vertikal		14.11

	Blur		95.24
girlface.bmp	-	47.73	0.77
	Rotasi 90^0		0.77
	Rotasi 180^0		0.77
	Rotasi 270^0		0.77
	Flip Horizontal		0.77
	Flip Vertikal		0.77
	Blur		100.0

5.4.2 Uji Coba Performa

Dalam uji coba performa akan digunakan semua data set dengan ukuran dan tipe data berbeda seperti pada Tabel 5-2. Hasil dari untuk uji coba dengan penyisipan data tersebut dapat dilihat pada

Dalam uji coba performa, juga akan dilakukan penyisipan maksimal ke setiap citra *cover*. Dalam hasil uji coba ini akan didapatkan kondisi terburuk untuk sebuah citra, jika kita menggunakan semua titik di setiap daerah *interest point*. Hasil untuk uji coba penyisipan maksimal ini bisa dilihat pada Tabel 5-4

Tabel 5-4 Hasil Uji Coba Performa Maksimal

Citra	*Interest point* sama (%)	PSNR (dB)	Kapasitas (Kb)	BER (%)
pepper.bmp	85	43.45	47,11	25.60
girl.bmp	83	44.59	53,42	0.95
barbara.bmp	94	41.46	52,35	18.75
cablecar.bmp	76	44.13	24,24	17.32
girlface.bmp	100	45.83	21,66	0.70

5.5 Evaluasi Uji Coba

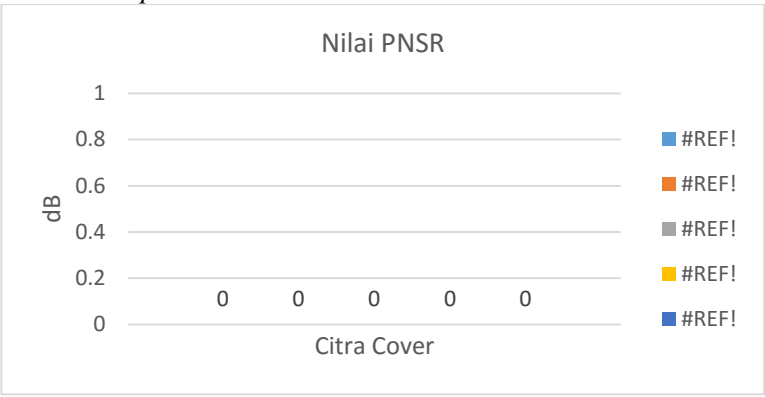
Pada bagian ini akan dijelaskan mengenai evaluasi berdasarkan hasil uji coba yang telah dijelaskan pada subbab 5.4.

Evaluasi uji coba terdiri dari dua bagian, yaitu evaluasi untuk uji coba fungsionalitas dan evaluasi untuk uji coba performa.

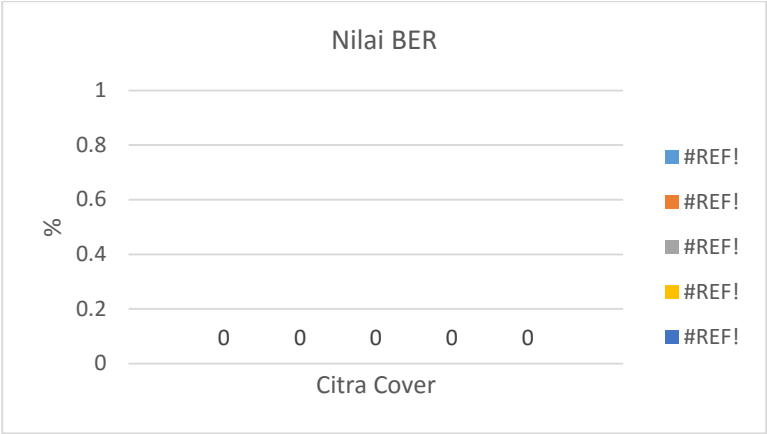
5.5.1 Evaluasi Uji Coba Fungsionalitas

Dari hasil uji coba fungsionalitas terhadap metode yang dikembangkan, maka dilihat metode bekerja dengan cukup baik, di mana rasi *bit error rate* (BER) paling besar adalah 1,48%. BER atau *bit error rate* adalah representasi kegagalan ekstraksi ketika dibandingkan dengan masukan yang sudah dijelaskan pada sub bab 2.8. Ketika citra dirotasi, data akan tetap tetap bisa diekstraksi dengan rasio BER yang sama. Hal ini dikarenakan rotasi yang dilakukan sempurna dan filter *laplacian* ketika proses *scale space representation* adalah berukuran *nxn* sehingga nilai respons untuk setiap *interest point* tidak akan berubah.

Gambar 5-7 menunjukkan nilai PSNR dan data yang gagal dikembalikan ketika proses ekstraksi. Kegagalan ekstraksi data ini disebabkan oleh terdapatnya titik *interest point* yang tidak terdeteksi atau titik tersebut memiliki urutan stabilitas yang berbeda. Perubahan nilai piksel di sekitar daerah *interest point* akan mempengaruhi nilai dari *interest point* tersebut. Sehingga terdapat kemungkinan urutan pergeseran nilai respons atau pergeseran titik dari *interest point* tersebut.



Gambar 5-7 Diagram Perbandingan Antar Citra



Gambar 5-8 Diagram BER Untuk Penyisipan Data Maksimum

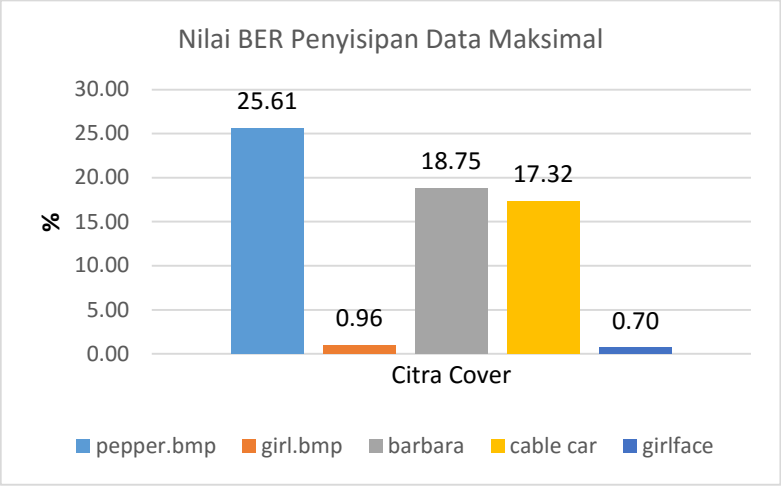
5.5.2 Evaluasi Uji Coba Performa

Dalam uji coba performa, rasio BER terbesar memiliki nilai 12%. Walaupun pada rasio ini, rasio keberhasilan pengembalian *interest point* cukup besar, namun rasio BER bisa menjadi besar ketika *interest point* yang gagal diekstrak kembali memiliki nilai luas daerah yang sangat besar.

Nilai rata-rata PSNR yang diperoleh dari penyisipan data ke semua titik yang bisa digunakan rata-rata adalah 42.36 dB. Nilai BER paling besar diperoleh adalah 12,57%. Hal ini dikarenakan sedikitnya titik yang bisa dikembalikan dan titik yang gagal dikembalikan memiliki luas daerah yang besar.

Gambar 5-10 menunjukkan nilai perbandingan untuk setiap gambar dengan menggunakan data *input* seperti yang ditunjukkan pada Tabel 5-2. Grafik menunjukkan bahwa nilai PSNR hasil uji coba terhadap 5 citra *cover* menggunakan 3 *file input* dengan tipe dan ukuran berbeda. Ukuran *file* yang akan disisip mempengaruhi *bit error rate* (BER) atau kesalahan data ketika proses ekstraksi. Hal ini dipengaruhi karena semakin banyaknya titik yang digunakan untuk penyimpanan data.

Hubungan ukuran data dan BER dapat dilihat pada Gambar 5-10 yang merupakan data BER untuk *file* "Cablecar.bmp" untuk 3 *file* inputan tersebut.

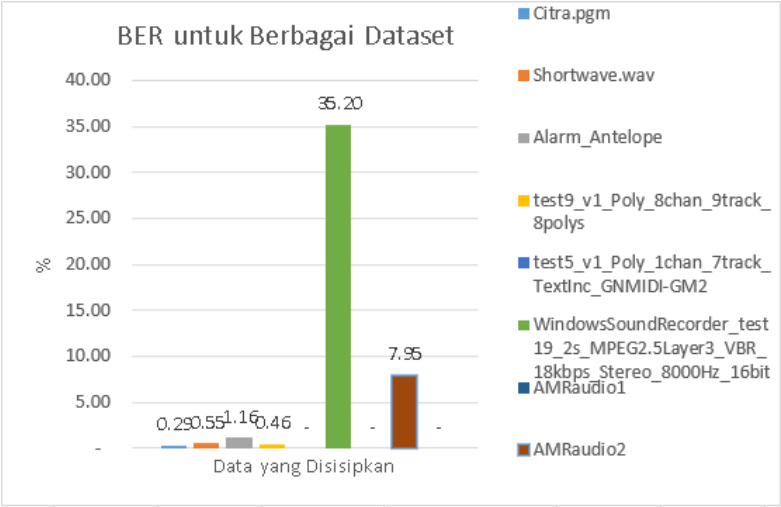


Gambar 5-9 Diagram BER Untuk Penyisipan Maksimal

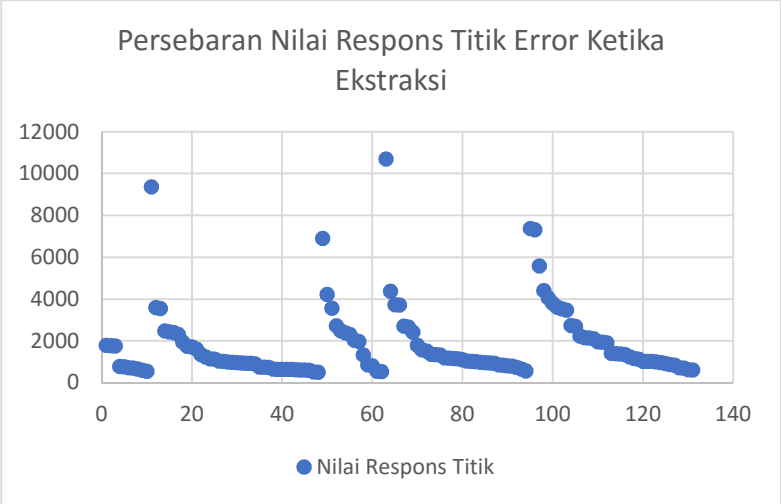
Gambar 5-11 menunjukkan nilai persebaran respons yang gagal atau *error* ketika di ekstraksi. Pada diagram ini ditunjukkan *interest point* dengan nilai respons kurang dari 5000 cenderung mengalami kesalahan ketika ekstraksi. Berdasarkan dari grafik pada Gambar 5-11 tersebut, nilai *threshold* yang digunakan untuk proses uji performa dan fungsionalitas adalah 5000.

Dalam semua uji tersebut, digunakan *threshold* sebagai batasan nilai respons adalah 5000. Hal ini untuk mengurangi nilai *error* ketika ekstraksi.

Selain nilai *threshold* banyaknya *layer* dan jumlah filter juga akan mempengaruhi hasil dari ekstraksi. Pengaruh dari jumlah *layer* dan filter dijelaskan pada subbab 3.3.2. Pada uji coba ini akan dibandingkan BER untuk beberapa nilai *layer* dan filter. Nilai jumlah *layer* yang dibandingkan adalah mulai dari 1 *layer* hingga 8 *layer*. Nilai default dari *library* adalah 4 seperti dokumentasi *library* Open CV [18].

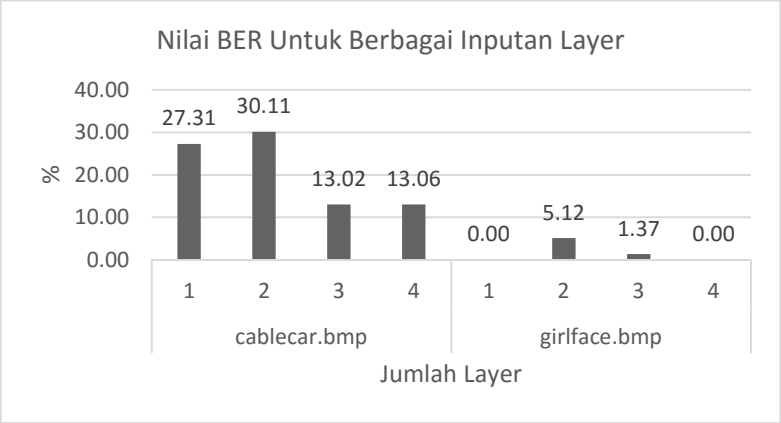


Gambar 5-10 Grafik Perbandingan Kapasitas Terhadap BER

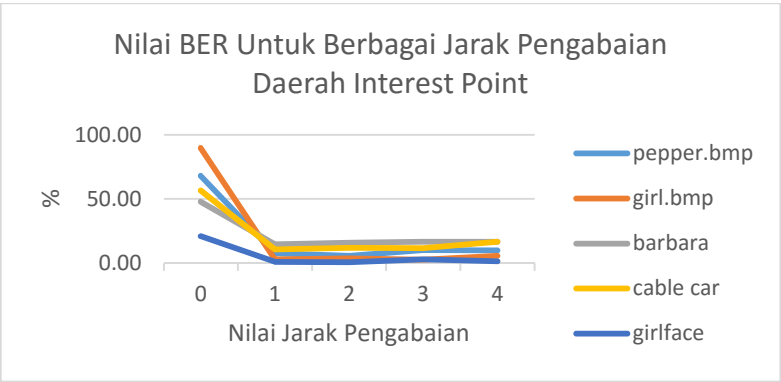


Gambar 5-11 Persebaran Nilai Respons Titik *Error* Ketika Ekstraksi

Gambar 5-12 menunjukkan pengaruh *layer* terhadap kapasitas dan BER. Dari grafik tersebut menunjukkan semakin besar jumlah *layer*, maka kapasitas yang didapatkan akan semakin besar kapasitas yang didapatkan. Hal ini dikarenakan semakin besar nilai *layer*, semakin banyak *interest point* yang didapatkan dan semakin kecil nilai BER yang diperoleh.



Gambar 5-12 Grafik Pengaruh *Layer* Terhadap Kapasitas dan BER



Gambar 5-13 Perbandingan BER Terhadap Pengabaian Daerah Sekitar *Interest Point*

Gambar 5-13 menunjukkan pengaruh dari perubahan daerah di sekitar *interest point*. Daerah dari *interest* point yang dimodifikasi akan mempengaruhi nilai respons *interest point*. Pada Gambar 5-13 menunjukkan pengaruh dari seberapa luas daerah yang tidak diubah. Di Gambar 5-13 menunjukkan daerah yang tidak diubah dari dimulai dari semua titik dengan *chebyshev distance* bernilai dua hingga. Pada Di Gambar 5-13 menunjukkan bahwa hasil terbaik ketika *chebyshev distance* bernilai 2.

[Halaman ini sengaja dikosongkan]

LAMPIRAN

Lampiran 1. Kumpulan Kode Sumber

```
1.  def decodeBelakang(self,img, daerahBlok,locationMapPath):
2.          temp=[]
3.          for x in range(0,len(self.locationMap)):
4.              temp.append(re.findall('..?', self.locationMap[x][2]))
5.          self.locationMap=temp
6.          hasil=''
7.          index=-1
8.          titik_hasil=[]
9.          while True:
10.             if len(self.locationMap)>len(daerahBlok):
11.                 self.locationMap.pop()
12.             else:
13.                 break
14.         for index in range(len(self.locationMap)-1,-1,-1):
15.             petaDaerah=self.locationMap[index]
16.             daerah=daerahBlok[index]
17.             hasil_perdaerah=''
18.             if len(daerah)==0:
19.                 for lm in petaDaerah:
20.                     if lm!='00':
21.                         hasil=hasil+'0'
22.                 continue
```

```
            for index_lm in range(len(petaDaerah)-1,-1,-1):
                locationMap=petaDaerah[index_lm]
                len(petaDaerah)
                if index_lm>=len(daerah):
                    if locationMap!='00':
                        hasil='0'+hasil
                    continue
                blok=daerah[index_lm]
                x=int(img[blok[0].y][blok[0].x])
                y=int(img[blok[1].y][blok[1].x])
                if locationMap=='00':
                    pass
                else:
                    m=int((x/2.0+y/2.0))
                    d=abs(int(x)-int(y))
                    b=str(d%2)
                    hasil=b+hasil
                    hasil_perdaerah=b+hasil_perdaerah
                    d_new=int(d/2)
                    if d_new>=1:
                        if locationMap=='01':
                            d=d_new+pow(2,(int(math.log(d_new,2))-1))
                        else:
                            d=d_new+pow(2,(int(math.log(d_new,2))))
                    else:
                        d=d_new
                    if(x<y):
```

```
                            y_new=m+int((d+1)/2)
                            x_new=m - int(d/2)
                        else:
                            x_new=m+int((d+1)/2)
                            y_new=m - int(d/2)
                        img[blok[0].y][blok[0].x]=x_new
                        img[blok[1].y][blok[1].x]=y_new
        return self.bitToString(hasil)
```

Kode Sumber 7-1Fungsi Ekstraksi Data

```
def encode(self, img, daerahBlok, data):
    banyak_titik=0
    diff=0
    indexBlok=0
    data_sisa=''
    i=0
    data_bit=''
    self.locationMap=''
    beda=0
    input=''
    panjang_data=len(data)
    banyak_data=0
    for region in daerahBlok:
        temp=panjang_data-len(data)

```

```
        banyak_data=panjang_data-len(data)
        if len(data)==0:
            break
        self.locationMap=self.locationMap+'\n'+self.stringToBit(str(region[0][0].scale)
)+'\n'+self.stringToBit(str(region[0][0].jumlahSelisih))+'\n'
        beda=0
        for k in range(0,len(region)) :
            locationMap='ripas'
            blok=region[k]
            if len(data)%8==0:
                selisih=abs(blok[0].posisi - region[len(region)-1][0].posisi )
                if  selisih<=8:
                    break
            if len(data)==0: break
            x=img[blok[0].y][blok[0].x]
            y=img[blok[1].y][blok[1].x]
            m=int(x/2.0+y/2.0)
            d=abs(int(x)-int(y))
            b=int(data[0])
            if d>=self.selisih:
                locationMap='00'
                self.locationMap=self.locationMap+'00'
            else:
                temp1=0
                temp2=0
                if d>=2:
                    temp= int(math.log(d,2))-1
```

```
                    d_new= int(d- pow(2,temp))
                    temp1=int(math.log(int(d),2))
                    temp2=int(math.log(d_new,2))
                else:
                    d_new=d
                d= abs(2*d_new+b)

                if( ( (m>=0 and m<=127) and (d<=2*m+1) ) or ( (m>=128 and m<=255) and ( d<=2*(255-m)) ) ):
                    beda=beda+d
                    data_bit=data_bit+str(data[0])
                    input=input+data[0]
                    data.remove(data[0])
                    i=i+1
                    if(x<y):
                        y_new=m+int((d+1)/2)
                        x_new=m - int(d/2)
                    else:
                        x_new=m+int((d+1)/2)
                        y_new=m - int(d/2)
                    img[blok[0].y][blok[0].x]=x_new
                    img[blok[1].y][blok[1].x]=y_new
                    if temp1==temp2:
                        locationMap='01'
                        flag2='01'
                    else:
                        locationMap='00'
```

```
                        flag2='10'
                    self.locationMap=self.locationMap+str(flag2)

                    diff=diff+math.pow((x_new-x),2)+math.pow((y_new-y),2)
                    beda=beda+abs(x-x_new)+abs(y-y_new)
                    banyak_titik=banyak_titik+1
                else:
                    locationMap='00'
                    self.locationMap=self.locationMap+'00'
    return (img,banyak_titik)
```

Kode Sumber 7-2 Fungsi Penyisipan Data

```
def disjoint(self):
        temp=[]
        flag_penuh={}
        for x in range(0,len(self.ipts)):
            flag_penuh[x]=0
            flag_embed=1
        for x in range(0,len(self.ipts)):
            if flag_penuh[x]!=0: continue
            for y in range(x+1,len(self.ipts)):
                    embed=self.euclidean(self.ipts[x],self.ipts[y])
                    if embed: continue
                    else:
```

```
                        flag_penuh[y]=1
                flag_penuh[x]=1
                temp.append(self.ipts[x])
            self.ipts=temp
            return self.ipts
```

Kode Sumber 7-3Pemisahan Daerah *Interest Point*

```
#!/Python27/python
import cgi, cgitb
import sys
import os
import cv2
import sys
import zipfile
from TA.surfPraProcessing import Surf
from TA.RDEBlok import RDEBlok

cgitb.enable()

sys.path.append('C:\\xampp\\cgi-bin\\TA\\TA')

base_path="/cgi-bin/TA/WEB"
try: # Windows needs stdio set for binary mode.
    import msvcrt
    msvcrt.setmode (0, os.O_BINARY) # stdin  = 0
```

```
    msvcrt.setmode (1, os.O_BINARY) # stdout = 1
except ImportError:
    pass

#get input from form Input
form = cgi.FieldStorage()

if not form.has_key('cover'):
    pass
else:
    image=form['cover']
tipe=form['tipe_data'].value
data=form['data_'+tipe].value
thresh=int(form['threshold'].value)
layer=int(form['layer'].value)
filter=int(form['filter'].value)
selisih=int(form['selisih'].value)
jarak=int(form['jarak'].value)


fn = os.path.basename(image.filename)
print "Content-Type: text/html"     # HTML is following
#fn='barbara.bmp'                              # blank line, end of headers
files=open('image_cover/'+fn, 'wb+')
while 1:
    chunk=image.file.read()
    if not  chunk:
```

```
46.         break
47.     files.write(chunk)
48. files.close()
49. 
50. 
51. img=cv2.imread('image_cover/'+fn,0)
52. img_awal=cv2.imread('image_cover/'+fn,0)
53. surf=Surf(img)
54. rde=RDEBlok()
55. 
56. surf.selisih=selisih
57. surf.jarak=jarak
58. surf.InterestPoint(thresh, filter ,layer)
59. surf.disjoint()
60. daerahBlok,td,total_titik=surf.findTitik()
61. 
62. 
63. #bikin data jadi list
64. binary=' '.join('{0:08b}'.format(ord(x), 'b') for x in data)
65. binary=binary.replace(' ','',)
66. binary=list(binary)
67. 
68. img_stegano,banyak_titik=rde.encode(surf.img,daerahBlok,binary)
69. psnr=rde.PSNR(img_awal,img_stegano)
70. cv2.imwrite('image_embedded/'+fn,img_stegano)
71. files=open('image_embedded/'+fn+'.txt','w+')
72. files.write(rde.locationMap)
```

```
73. files.close()
74. 
75. zipf=zipfile.ZipFile('/xampp/htdocs/TA/'+fn+'.zip',mode='w')
76. zipf.write('image_embedded/'+fn)
77. zipf.write('image_embedded/'+fn+'.txt')
78. zipf.close()
79. 
80. if tipe=="file":
81.     open("/xampp/htdocs/TA/"+form['data_'+tipe].filename,'w+').write(data)
82. data_uri = open('image_embedded/'+fn, 'rb').read().encode('base64').replace('\n', '')

83. img_tag = '<img src="{0}" >'.format(data_uri)
84. 
85. print "Content-type: text/html; charset=utf-8\r\n\r\n"
86. print open("index.html",'r').read()
87. print """<div style="width:100%">"""
88. print """
89. <div>
90. <table style="margin-left:15px">
91.   <tr>
92.     <td>PSNR (dB)</td>
93.     <td>Titik Terpakai(%)</td>
94.   </tr>
95.   <td>"""+str(psnr)+"""</td>
96.   <td>"""+str(banyak_titik*100.0/total_titik)+"""</td>
97. </table>
98. """
```

```
print """

    <a href='/TA/"""+fn+""".zip' style="text-
decoration:none;width:30px;color:white; " >
    <div class="button" style="background:#8bc34a;width:300px;margin:10px;padding:15p
x;text-align:center;font-size:20px">Download gambar dan location Map</div>


    </a>"""

    print  "<div>"+img_tag+"</div>"
    print "</div>"
```

Kode Sumber 7-4Embedded Data

```
#!/Python27/python
import cgi, cgitb
import sys
import os
import cv2
import hashlib

import time

from TA.RDEBlok import RDEBlok
from TA.surfPraProcessing import Surf
```

```
12. cgitb.enable()
13.
14. sys.path.append('C:\\xampp\\cgi-bin\\TA\\TA')
15.
16. base_path="/cgi-bin/TA/WEB"
17. try: # Windows needs stdio set for binary mode.
18.     import msvcrt
19.     msvcrt.setmode (0, os.O_BINARY) # stdin  = 0
20.     msvcrt.setmode (1, os.O_BINARY) # stdout = 1
21. except ImportError:
22.     pass
23.
24. form = cgi.FieldStorage()
25. #get Input from form web
26. image=form['cover']
27. locationMap=form['locationmap']
28. ext=form['ext'].value
29. thresh=int(form['threshold'].value)
30. layer=int(form['layer'].value)
31. filter=int(form['filter'].value)
32. selisih=int(form['selisih'].value)
33. jarak=int(form['jarak'].value)
34.
35. #simpan image ke tmp folder
36. image_name = os.path.basename(image.filename)
37. files=open('tmp/'+image_name, 'wb')
38. while 1:
```

```
39.     chunk=image.file.read()
40.     if not  chunk:
41.         break
42.     files.write(chunk)
43. files.close()
44. 
45. 
46. locationMap_name=os.path.basename((locationMap.filename))
47. files=open('tmp/'+locationMap_name,'wb+')
48. while 1:
49.     chunk=locationMap.file.read()
50.     if not  chunk: break
51.     files.write(chunk)
52. files.close()
53. 
54. 
55. 
56. 
57. img=cv2.imread('tmp/'+image_name,0)
58. surf=Surf(img)
59. surf.InterestPoint(thresh,filter,layer)
60. surf.disjoint()
61. rde=RDEBlok()
62. rde.locationMap=open('tmp/'+locationMap_name,'r').read()
63. rde.buildLM()
64. 
65. surf.validasiKP(rde.locationMap)
```

```
daerahBlok,td,total_titik=surf.findTitik()
data_hidden=rde.decodeBelakang(surf.img,daerahBlok,rde.locationMap)
print "Content-type: text/html; charset=utf-8\r\n\r\n"

print open("index.html",'r').read()
if ext!="":
    file_name=hashlib.md5(str(time.time())).hexdigest()
    open("/xampp/htdocs/TA/tmp_hasil/"+file_name+"."+ext,'wb+').write(data_hidden)
    print """ <div>
    Data hasil ekstraksi adalah : <a href='/TA/tmp_hasil/"""+file_name+""".""""+ext+"""
'>"""+file_name+""".""""+ext+""" </a>
    </div>
    """
else:
    print """ <div>
    Data hasil ekstraksi adalah : """+data_hidden+"""
    </div>
    """
```

Kode Sumber 7-5 Ekstraksi Data

```
#!/Python27/python
import cgi
base_path="/cgi-bin/TA/WEB"

print "Content-type: text/html\r\n\r\n"
```

```
6.
7.  print open('index.html','r').read()
```

Kode Sumber 7-6 Menampilkan Halaman Utamas

7.1 Lampiran Hasil Uji Coba

Tabel 7-1 Lampiran Hasil Uji Coba Penyisipan Berbagai Data Rahasia

Citra *Cover*	Data Rahasia	PSNR (dB)	*Interest* Point Awal	*Interest* Point Akhir	*interest point* sama	Panjang Data (kb)	Titik Terpakai (%)	byte of error(%)	intest point sama(%)
pepper.bmp	Citra.pgm	43.28008	30	30	29	57	4.109	1.38719883 2	96.66667
	Short wave gelombang.wav	45.69944	30	30	29	28	2.562	1.09289617 5	96.66667

	Alarm_Antelope	42.12251	30	31	14	438	5.16	8.48837209 3	46.66667
	test9_v1_Poly_8chan_9track_8polys	44.96438	30	30	28	85	3.046	2.79054497 7	93.33333
	test5_v1_Poly_1chan_7track_TextInc_GNMI	48.10454	30	30	29	28	1.475	1.89830508 5	96.66667

	DI-GM2								
	WindowsSoundRecorder_test19_2s_MPEG2.5Layer3_VBR_18kbps_Stereo_8000Hz_16bit	44.56476	30	30	28	84	3.31	2.53776435	93.33333

	AMR audio 1	47.32194	30	30	30	0	1.773	0	100
	AMR audio 2	44.7279	30	30	28	85	3.373	2.52001185 9	93.33333
	test10_v0_Mono_Mono	55.78968	30	30	30	0	0.402	0	100
girl.bmp	Citra.pgm	47.70459	30	30	29	12	4.109	0.29204185 9	96.66667
	Shortwave.wav	49.13489	30	30	29	14	2.562	0.54644808 7	96.66667

	Alarm_Antelope	46.52856	30	30	27	60	5.16	1.16279069 8	90
	test9_v1_Poly_8chan_9track_8polys	48.90634	30	30	29	14	3.046	0.45961917 3	96.66667
	test5_v1_Poly_1chan_7track_TextInc_GNMI	51.5528	30	30	30	0	1.475	0	100

	DI-GM2								
	WindowsSoundRecorder_test19_2s_MPEG2.5Layer3_VBR_18kbps_Stereo_8000Hz_16bit	48.33421	30	31	14	1165	3.31	35.196374622	46.66667

	AMR audio 1	51.07484	30	30	30	0	1.773	0	100
	AMR audio 2	48.5235	30	30	28	268	3.373	7.944549155	93.33333
	test10_v0_Mono_Mono	56.56333	30	30	30	0	0.402	0	100
barbara	Citra.pgm	39.56143	31	31	29	1159	4.109	28.2063762 5	93.54839
	Shortwave.wav	43.49077	31	32	29	952	2.562	37.1584699 5	93.54839

	Alarm_Antelope	38.29926	31	32	28	1168	5.16	22.6356589 1	90.32258
	test9_v1_Poly_8chan_9track_8polys	43.20085	31	32	26	1257	3.046	41.2672357 2	83.87097
	test5_v1_Poly_1chan_7track_TextInc_GNMI	44.57742	31	32	30	0	1.475	0	96.77419

	DI-GM2								
	WindowsSoundRecorder_test19_2s_MPEG2.5Layer3_VBR_18kbps_Stereo_8000Hz_16bit	41.95846	31	32	28	1164	3.31	35.166163 14	90.32258

	AMR audio 1	44.21037	31	32	29	277	1.773	15.623237445	93.54839
	AMR audio 2	41.8302	31	32	26	1278	3.373	37.8891194 8	83.87097
	test10_v0_Mono_Mono	54.5805	31	32	30	0	0.402	0	96.77419
	Citra.pgm	40.14151	37	37	34	333	4.109	8.10416159 7	91.89189
cable car	Short wave .wav	42.69874	37	37	37	0	2.562	0	100

	Alarm_Antelope	39.9008	37	37	32	399	4.309	9.259689023	86.48649
	test9_v1_Poly_8chan_9track_8polys	41.48433	37	37	34	213	3.046	6.992777413	91.89189
	test5_v1_Poly_1chan_7track_TextInc_GNMI	45.62439	37	37	37	0	1.475	0	100

	DI-GM2								
	WindowsSoundRecorder_test19_2s_MPEG2.5Layer3_VBR_18kbps_Stereo_8000Hz_16bit	40.79333	37	37	34	317	3.31	9.577039275	91.89189

	AMR audio 1	44.55599	37	37	37	0	1.773	0	100
	AMR audio 2	40.79881	37	37	34	325	3.373	9.63533946	91.89189
	test10_v0_Mono_Mono	52.24316	37	37	37	0	0.402	0	100
girlface	Citra.pgm	44.31927	14	14	12	53	3.867	1.37057150 2	85.71429
	Shortwave.wav	46.65023	14	14	14	0	2.562	0	100
	Alarm_A	44.34376	14	14	13	21	3.862	0.54375971	92.85714

	ntelope								
	test9_v1_Poly_8chan_9track_8polys	45.73248	14	14	14	0	3.046	0	100
	test5_v1_Poly_1chan_7track_TextInc_GNMIDI-GM2	50.30666	14	14	14	0	1.475	0	100

	WindowsSoundRecorder_test19_2s_MPEG2.5Layer3_VBR_18kbps_Stereo_8000Hz_16bit	45.16761	14	14	13	30	3.31	0.90634441	92.85714
	AMR audio 1	49.16103	14	14	14	0	1.773	0	100

	AMR audio 2	45.26181	14	14	13	31	3.373	0.91906314 9	92.85714
	test10_v0_Mono_Mono	53.03429	14	14	14	0	0.402	0	100

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari tujuan pembuatan perangkat lunak dan hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, terdapat pula saran yang ditujukan untuk pengembangan perangkat lunak lebih lanjut.

6.1. Kesimpulan

Dalam proses pengerjaan Tugas Akhir mulai dari tahap analisis, desain, implementasi, hingga pengujian didapatkan kesimpulan sebagai berikut:

1. *Interest point* bisa dimanfaatkan dalam meningkatkan ketahanan dalam steganografi.
2. Terdapat *interest point* yang gagal di ekstraksi ketika proses ekstraksi data dikarenakan dilakukan perubahan nilai piksel di sekitar *interest point* tersebut.
3. Mempertahankan nilai piksel di sekitar *interest point* bisa mempertahankan nilai dari *interest point*. Nilai piksel yang dipertahankan adalah minimal 1.
4. *Interest point* bisa dimanfaatkan untuk menyisipkan data di proses steganografi karena bisa meningkatkan ketahanan data.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang, berdasarkan pada hasil perancangan, implementasi dan uji coba yang telah dilakukan.

1. Diperlukan pengujian lebih lanjut untuk menentukan pengaruh perubahan dari setiap piksel pada suatu daerah *interest point* terhadap nilai dari *interest point* tersebut.
2. Menggunakan metode steganografi lain yang membuat perubahan atau modifikasi dari piksel semakin sedikit.

3. Penelitian lebih lanjut dengan menggunakan metode pendeteksian *interest point* yang lain

DAFTAR PUSTAKA

[1] D.-C. Lou, M.-C. Hu and J.-L. Liu, "Multiple layer data hiding scheme for medical images," *Computer Standards & Interfaces,* vol. 31(2), pp. 329-335, 2009.

[2] Bay, Herbert, T. Tuytelaars and L. V. Gool, "Surf: Speeded up robust features," *Computer Vision and Image Understanding,* vol. 110, no. 3, pp. 404-417, 2006.

[3] Johnson, N. F and S. Jajodia, "Exploring steganography: Seeing the unseen.," *Computer 31.2,* pp. 26-34, 1998.

[4] Liu, Chiang-Lung, Lou, Der-Chyuan and C.-C. Lee, "Reversible data embedding using reduced difference expansion," *Intelligent Information Hiding and Multimedia Signal Processing,* vol. I, pp. 433-436, 2007.

[5] D. Putra, Pengolahan Citra Digital, Yogyakarta: Andi, 2010.

[6] Hamid, N. Yahya, A. Ahmad and Al-Qershi, "Characteristic Region Based Image Steganography Using Speeded-Up Robust Features Technique," *International Conference on Future Communication Networks,* pp. 141-146, 2012.

[7] C. D. Cantrell, Modern Mathematical Methods for Physicists and Engineers. Cambridge University Press, 2000.

[8] M. Neighborhood, "Moore Neighborhood," [Online]. Available: http://mathworld.wolfram.com/MooreNeighborhood.html. [Accessed 8 Juni 2016].

[9] "General Documentation," [Online]. Available: http://www.ffmpeg.org/general.html#Image-Formats. [Accessed 6 juni 2016].

[10] C. Baldick, in *The Oxford Dictionary of Literary Terms*, 2008.

[11] P. M. Panchal, S. R. Panchal and S. K. Shah, "A Comparison of SIFT and SURF," *International Journal of Innovative*

Research in Computer and Communication Engineering , vol. 1, no. 2, pp. 323-327, 2013.

[12] Evans and Christopher, Notes on the OpenSURF Library, 2009.

[13] "OpenCV," [Online]. Available: http://docs.opencv.org/2.4/modules/features2d/doc/common_interfaces_of_feature_detectors.html. [Accessed 25 May 2016].

[14] Hamid and Nagham, "A Comparison between using SIFT and SURF for characteristic region based image steganography," *International Journal of Computer Science,* pp. 110-116, 2012.

[15] [Online]. Available: http://www.hlevkin.com/TestImages/. [Accessed 13 Juni 2016].

[16] [Online]. Available: http://download.wavetlan.com/. [Accessed Juni 15 2016].

[17] " LogoEPS.com," [Online]. Available: http://www.logoeps.com/abc-logo-vector-eps-format/5660/. [Accessed 6 Juni 2016].

[18] O. CV, "Open CV," [Online]. Available: http://docs.opencv.org/2.4/modules/nonfree/doc/feature_detection.html. [Accessed 8 Juny 2016].

BIODATA PENULIS

Ripas Filqadar dilahirkan di Inuman 22 tahun silam. Penulis menempuh pendidikan mulai dari SDN 1 Inuman, (2000-2006), SMPN 1 Inuman, (2006-2009), SMAN Plus Propinsi Riau(2009-20012), dan S1 Teknik Informatika ITS (2012-2016).

Selama kuliah, penulis aktif dalam organisasi Himpunan Mahasiswa Teknik Computer-Informatika ITS (HMTC). Diantaranya adalah sebagai Staf Departemen Kewirausahaan dan Minat Bakat HMTC (2013-2015). Selain aktif di organisasi HMTC, penulis juga pernah manjadi Staf DPM BEM-FTIf (2014-2015). Penulis juga aktif dalam kegiatan kepanitiaan Schematics dengan menjadi staf Dana dan Sponsorhip pada Schematics 2014. Penulis juga merupakan seorang administrator di Laboratorium Komputasi Berbasis Jaringan dan pernah menjadi asisten mata kuliah Sistem Operasi, Jaringan Komputer dan Pemrograman jaringan

Kritik dan saran sangat diharapkan guna peningkatan kualitas dan penulisan selanjutnya. Untuk itu, silakan kirim kritik dan saran ke : **ripas.filqadar@gmail.com**