TUGAS AKHIR – KI141502

REVERSIBLE AUDIO STEGANOGRAPHY MENGGUNAKAN METODE *REDUCED DIFFERENCE EXPANSION* DAN *SAMPLE VALUE MODIFICATION*

MUHAMMAD GHULAM FAJRI
NRP 5113100094

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

Dosen Pembimbing II
Bagus Jati Santoso, S.Kom., Ph.D.

Departemen Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR – KI141502

REVERSIBLE AUDIO STEGANOGRAPHY MENGGUNAKAN METODE *REDUCED DIFFERENCE EXPANSION* DAN *SAMPLE VALUE MODIFICATION*

MUHAMMAD GHULAM FAJRI
NRP 5113100094

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

Dosen Pembimbing II
Bagus Jati Santoso, S.Kom., Ph.D.

Departemen Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESIS – KI141502

REVERSIBLE AUDIO STEGANOGRAPHY USING REDUCED DIFFERENCE EXPANSION AND SAMPLE VALUE MODIFICATION METHOD

MUHAMMAD GHULAM FAJRI
NRP 5112100094

First Advisor
Tohari Ahmad, S.Kom., MIT., Ph.D.

Second Advisor
Bagus Jati Santoso, S.Kom., Ph.D.

Department of Informatics
Faculty of Information Technology
Sepuluh Nopember Institute of Technology
Surabaya 2017

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

REVERSIBLE AUDIO STEGANOGRAPHY MENGGUNAKAN METODE *REDUCED DIFFERENCE EXPANSION* DAN *SAMPLE VALUE MODIFICATION*

TUGAS AKHIR

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

Oleh:
MUHAMMAD GHULAM FAJRI
NRP: 5113100094

Disetujui oleh Pembimbing Tugas Akhir:

1. Tohari Ahmad, S.Kom., MIT, Ph.D. ..... (Pembimbing 1)
   (NIP. 19750525 200312 1 002)

2. Bagus Jati Santoso, S.Kom., Ph.D. ..... (Pembimbing 2)
   (NIP. 051100116)

SURABAYA
JUNI, 2017

[Halaman ini sengaja dikosongkan]

REVERSIBLE AUDIO STEGANOGRAPHY MENGGUNAKAN METODE *REDUCED DIFFERENCE EXPANSION* DAN *SAMPLE VALUE MODIFICATION*

Nama Mahasiswa : MUHAMMAD GHULAM FAJRI
NRP : 5113100094
Departemen : Teknik Informatika FTIF-ITS
Dosen Pembimbing 1 : Tohari Ahmad, S.Kom., MIT., Ph.D.
Dosen Pembimbing 2 : Bagus Jati Santoso, S.Kom., Ph.D.

Abstrak

Steganografi adalah teknik penyisipan data rahasia kedalam suatu media, sehingga selain pengirim dan penerima tidak ada yang mengetahui dan menyadari adanya informasi yang disisipkan. Audio merupakan salah satu media yang dapat digunakan dalam penyisipan data rahasia. Dalam proses penyisipannya dibutuhkan teknik penyisipan yang dapat meningkatkan tingkat keamanan tanpa mengurangi kualitas, kapasitas serta performa dari proses penyisipan yang lebih efisien.

Tugas akhir ini mengimplementasikan dua metode penyisipan pesan rahasia kedalam media audio. Reduced Difference Expansion (RDE) dan Sample Value Modification (SVM), RDE memanfaatkan selisih sepasang sample untuk proses penyisipan pesan, sementara SVM memodifikasi nilai sepasang sample dalam proses penyisipan pesan.

Hasil dari uji coba yang dilakukan menunjukkan bahwa kombinasi metode RDE dan SVM dengan pengolahan pesan menggunakan Modulus Function dapat meningkatkan tingkat keamanan informasi, kualitas dan kapasitas dengan jenis penyisipan menggunakan Location Map dari pada tanpa Location Map. Sementara penyisipan pesan tanpa Location Map menghasilkan performa yang lebih baik dalam segi penggunaan memori yang lebih efisien dari pada menggunakan Location Map.

Kata kunci: Audio, Data hiding, Modulus function, Steganografi, Reversible, RDE, SVM.

REVERSIBLE AUDIO STEGANOGRAPHY USING REDUCED DIFFERENT EXPANSION AND SAMPLE VALUE MODIFICATION METHOD

Student's Name	**: MUHAMMAD GHULAM FAJRI**
Student's ID	**: 5113100094**
Department	**: Department of Informatics FTIF-ITS**
First Advisor	**: Tohari Ahmad, S.Kom., MIT., Ph.D.**
Second Advisor	**: Bagus Jati Santoso, S.Kom., Ph.D.**

Abstract

Steganography is the technique of insertion of confidential data into a medium, so other than the sender and recipient nobody knows and aware of the embedded information. Audio is one medium that can be used in the insertion of confidential data. In the insertion process required the insertion technique that can improve the level of security without decreasing on quality, capacity and insertion process more efficient.

This final project implements two methods of inserting confidential messages into audio media. Reduced Difference Expansion (RDE) and Sample Value Modification (SVM), RDE using differences pair of sample for the insertion process the message, while SVM modify a pair of sample values in the process of message insertion.

Based on experiments, the combination of RDE and SVM methods with message processing using modulus function can improve the level of information security, quality and capacity with insertion type using location map than without location map. While message insertion process without the location map results in better performance, in term of memory usage efficiency, than using the location map.

Keywords: Audio, Data hiding, Modulus function, Steganography, Reversible, RDE, SVM

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

Alhamdulillahirabbil'alamin, segala puji bagi Allah SWT, yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "***REVERSIBLE AUDIO STEGANOGRAPHY* MENGGUNAKAN METODE *REDUCED DIFFERENCE EXPANSION* DAN *SAMPLE VALUE MODIFICATION***". Tugas Akhir ini merupakan salah satu syarat dalam menempuh ujian sidang guna memperoleh gelar Sarjana Komputer. Dengan pengerjaan Tugas Akhir ini, penulis bisa belajar banyak untuk memperdalam dan meningkatkan apa yang telah didapatkan penulisa selama menempuh perkuliahaan di Teknik Informatika ITS.

Selesainya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan beberapa pihak, sehingga pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Orang tua penulis serta keluarga penulis yang selalu memberikan dukungan doa, moral, dan material yang tak terhingga kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
3. Bapak Tohari Ahmad, S.Kom., MIT., Ph.D. dan Bapak Bagus Jati Santoso, S.Kom., Ph.D. selaku dosen pembimbing penulis yang telah membimbing dan memberikan motivasi, nasehat dan bimbingan dalam menyelesaikan Tugas Akhir ini..
4. Bapak Darlis Herumurti, S.Kom., M.Kom. selaku kepala departemen Teknik Informatika ITS.
5. Bapak Dr. Radityo Anggoro, S.Kom., M.Sc. selaku koordinator Tugas Akhir.

6. Seluruh dosen dan karyawan Teknik Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa studi di ITS.
7. Ibu Eva Mursidah dan Ibu Sri Budiati yang selalu mempermudah penulis dalam peminjaman buku di RBTC.
8. Teman-teman seperjuangan RMK NCC/KBJ, yang telah menemani dan menyemangati penulis.
9. Teman-teman administrator NCC/KBJ, yang telah menemani dan menyemangati penulis selama penulis menjadi administrator.
10. Teman-teman angkatan 2013, yang sudah mendukung penulis selama perkuliahan.
11. Sahabat penulis yang tidak dapat disebutkan satu per satu yang selalu membantu, menghibur, menjadi tempat bertukar ilmu dan berjuang bersama-sama penulis.
12. Tim Futsal Teknik Informatika 2015 dan 2017 yang telah menemani penulis dalam keikutsertaan pada lomba IFC ITS.

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

Surabaya, Juni 2017

DAFTAR ISI

LEMBAR PENGESAHAN ..... **v**
Abstrak ..... **vii**
Abstract ..... **ix**
DAFTAR ISI ..... **xiii**
DAFTAR GAMBAR ..... **xvii**
DAFTAR TABEL ..... **xxi**
DAFTAR KODE SUMBER ..... **xxiii**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Batasan Permasalahan 2
1.4 Tujuan 3
1.5 Manfaat 3
1.6 Metodologi 3
1.6.1 Penyusunan Proposal Tugas Akhir 3
1.6.2 Studi Literatur 4
1.6.3 Implementasi Perangkat Lunak 4
1.6.4 Pengujian dan Evaluasi 4
1.6.5 Penyusunan Buku 5
1.7 Sistematika Penulisan Laporan 5
BAB II TINJAUAN PUSTAKA **7**
2.1 Steganografi 7
2.2 *Difference Expansion* 8
2.3 *Reduced Difference Expansion* 9
2.4 *Sample Value Modification* 11
2.5 *Correlation* 13
2.6 *SIM (Similarity Index Modulation)* 13
2.7 *MSE (Mean Square Error)* 14
2.8 *SNR (Signal to Noise Ratio)* 14
2.9 *PSNR (Peak Signal to Noise Ratio)* 15
2.10 Python 15
2.11 Anaconda 16
2.12 SciPy 16

2.13 NumPy 17
2.14 Matplotlib 17
2.15 PyQt 17
BAB III PERANCANGAN PERANGKAT LUNAK 19
3.1 Data 19
3.1.1 Data Masukan 19
3.1.2 Data Keluaran 19
3.2 Dekripsi Umum Sistem 20
3.3 Perancangan Modifikasi Metode *RDE* dengan *Location Map* 21
3.4 Perancangan Modifikasi Metode *SVM* dengan *Location Map* 24
3.5 Perancangan Penyisipan dan Ekstraksi Payload 28
3.5.1 Perancangan Penyisipan *Payload* 29
3.5.2 Perancangan *Location Map* 30
3.5.3 Perancangan Ekstraksi *Payload* dan *Reversible Audio* 30
3.6 Perancangan Modifikasi Metode *RDE* Tanpa *Location Map* 33
3.7 Perancangan Antarmuka 35
BAB IV IMPLEMENTASI 41
4.1 Lingkungan Implementasi 41
4.2 Implementasi 41
4.2.1 Kelas *Helper* 42
4.2.1.1 Implementasi Kelas *IOClass* 42
4.2.1.2 Implementasi Kelas *Sample* 45
4.2.1.3 Implementasi Kelas *Report* 46
4.2.2 Kelas *BASE3* 48
4.2.2.1 Implementasi Fungsi *toascii* 49
4.2.2.2 Implementasi Fungsi *tobase3* 49
4.2.2.3 Implementasi Fungsi *tochar* 50
4.2.2.4 Implementasi Fungsi *revbase3* 50
4.2.3 Kelas *RDE* 52
4.2.3.1 Implementasi Fungsi *NoLMRDE* 52
4.2.3.2 Implementasi Fungsi *RDE* 55

4.2.3.3 Implementasi Fungsi *DecNoLMRDE* 57
4.2.3.4 Implementasi Fungsi *DecodeRDE* 59
4.2.4 Kelas SVM..... 62
4.2.4.1 Implementasi Fungsi *NoLMSVM* 62
4.2.4.2 Implementasi Fungsi *DecLMSVM*..... 63
4.2.4.3 Implementasi Fungsi *SVM*..... 64
4.2.4.4 Implementasi Fungsi *DecodeSVM* 66
BAB V HASIL UJI COBA DAN EVALUASI 69
5.1 Lingkungan Pengujian..... 69
5.2 Dataset Pengujian..... 69
5.2.1 *Cover Audio (.wav)*..... 69
5.2.2 *Payload Data (.txt)* 70
5.3 Skenario Uji Coba 71
5.3.1 Skenario Uji Coba Fungsionalitas 71
5.3.2 Skenario Uji Coba Performa..... 72
5.3.3 Skenario Uji Coba Parameter *Cover* dan *Payload*..72
5.3.3.1 Skenario Uji Coba Parameter *Cover* 73
5.3.3.2 Skenario Uji Coba Parameter *Payload*..... 73
5.4 Uji Coba 73
5.4.1 Uji Coba Fungsionalitas..... 74
5.4.2 Uji Coba Performa..... 75
5.4.3 Uji Coba Parameter *Cover* dan *Payload*..... 80
5.4.3.1 Uji Coba Parameter *Cover*..... 80
5.4.3.2 Uji Coba Parameter *Payload*..... 82
5.5 Evaluasi Uji Coba..... 85
5.5.1 Evaluasi Uji Coba Fungsionalitas..... 85
5.5.2 Evaluasi Uji Coba Performa 89
5.5.3 Evaluasi Uji Coba Parameter *Cover* dan *Payload* ..98
5.5.3.1 Evaluasi Uji Coba Parameter *Cover*..... 98
5.5.3.2 Evaluasi Uji Coba Parameter *Payload* 108
5.6 Evaluasi Umum Skenario Uji Coba 118
BAB VI KESIMPULAN DAN SARAN..... 121
6.1 Kesimpulan..... 121
6.2 Saran..... 121
DAFTAR PUSTAKA 123

LAMPIRAN **125**
BIODATA PENULIS **131**

DAFTAR GAMBAR

Gambar 2.1 Skema Penyisipan Digit Payload Pada Metode SVM Tanpa Location Map 12
Gambar 3.1 Skema Penyisipan Digit Payload ke Dalam Cover Sample 20
Gambar 3.2 Skema Penyisipan Digit Payload Menggunakan Metode RDE 23
Gambar 3.3 Pemetaan Message Block 24
Gambar 3.4 Penyisipan Digit Payload ke Dalam Message block 0 25
Gambar 3.5 Penyisipan Digit Payload ke Dalam Message block 1 25
Gambar 3.6 Penyisipan Digit Payload ke Dalam Message block 2 26
Gambar 3.7 Skema Penyisipan Digit Payload Pada Metode SVM Menggunakan Location Map 27
Gambar 3.8 Skema Ekstraksi Payload dan Reversible Audio 32
Gambar 3.9 Skema Penyisipan Payload Pada Metode RDE Tanpa Location Map 34
Gambar 3.10 Desain Tampilan Antarmuka (Encode) 36
Gambar 3.11 Desain Tampilan Antarmuka (Decode) 37
Gambar 5.1 Nilai Correlation Cover-audio & Recovery-audio pada Masing Masing Genre Musik 86
Gambar 5.2 Nilai Correlation Payload & Recovery-payload pada Masing Masing Genre Musik 86
Gambar 5.3 Nilai MSE pada Masing Masing Genre Musik 87
Gambar 5.4 Nilai PSNR pada Masing Masing Genre Musik 88
Gambar 5.5 Nilai Correlation Cover-audio & Stego-audio pada Masing Masing Genre Musik 88
Gambar 5.6 Nilai MSE Uji Coba Performa pada Masing Masing Genre Musik 91
Gambar 5.7 Nilai PSNR Uji Coba Performa pada Masing Masing Genre Musik 91

Gambar 5.8 Nilai Correlation Cover-audio & Stego-audio pada Masing Masing Genre Musik 93
Gambar 5.9 Nilai Correlation Payload & Recovery-payload pada Masing Masing Genre Musik 93
Gambar 5.10 Waktu Penyisipan Payload Pada Masing Masing Genre Musik 95
Gambar 5.11 Waktu Ekstraksi Payload Pada Masing Masing Genre Musik 95
Gambar 5.12 Pemakaian Memori (Proses Penyisipan) pada Uji Coba Performa Pada Masing Masing Genre Musik 97
Gambar 5.13 Pemakaian Memori (Proses Ekstraksi) pada Uji Coba Performa Pada Masing Masing Genre Musik 98
Gambar 5.14 Nilai MSE pada Uji Coba Parameter Cover 100
Gambar 5.15 Nilai PSNR pada Uji Coba Parameter Cover 100
Gambar 5.16 Nilai Correlation Cover-audio & Stego-audio pada Uji Coba Parameter Cover 102
Gambar 5.17 Nilai Correlation Payload & Recovery-payload pada Uji Coba Parameter Cover 102
Gambar 5.18 Waktu Penyisipan Payload pada Uji Coba Parameter Cover 104
Gambar 5.19 Waktu Ekstraksi Payload pada Uji Coba Parameter Cover 105
Gambar 5.20 Pemakaian Memori saat Proses Penyisipan Payload pada Uji Parameter Cover 107
Gambar 5.21 Pemakaian Memori saat Ekstraksi Payload pada Uji Coba Parameter Cover 107
Gambar 5.22 Nilai MSE pada Uji Coba Parameter Payload 110
Gambar 5.23 Nilai PSNR pada Uji Coba Parameter Payload 110
Gambar 5.24 Nilai Correlation Cover-audio dan Stego-Audio Uji Coba Parameter Payload 112
Gambar 5.25 Nilai Correlation Payload dan Recovery-payload Uji Coba Parameter Payload 112
Gambar 5.26 Waktu Penyisipan Payload pada Uji Coba Parameter Payload 114

Gambar 5.27 Waktu Ekstraksi Payload pada Uji Coba Parameter Payload 115
Gambar 5.28 Pemakaian Memori saat Proses Penyisipan Payload pada Uji Parameter Payload 117
Gambar 5.29 Pemakaian Memori saat Ekstraksi Payload pada Uji Coba Parameter Payload 117

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

Tabel 4.1 Lingkungan Implementasi Perangkat Lunak.....41
Tabel 5.1 Spesifikasi Lingkungan Pengujian.....69
Tabel 5.2 Detail Data Audio.....70
Tabel 5.3 Detail Data Payload.....70
Tabel 5.4 Hasil Uji Coba Fungsionalitas dengan Location Map.74
Tabel 5.5 Hasil Uji Coba Fungsionalitas tanpa Location Map....75
Tabel 5.6 Hasil Uji Coba Performa dengan Kapasitas Maksimal (POP).....76
Tabel 5.7 Hasil Uji Coba Performa dengan Kapasitas Maksimal (METAL) 77
Tabel 5.8 Hasil Uji Coba Performa dengan Kapasitas Maksimal (ELECTRO) 78
Tabel 5.9 Hasil Uji Coba Performa dengan Kapasitas Maksimal (JAZZ).....79
Tabel 5.10 Uji Coba Parameter Cover Location Map (POP).....80
Tabel 5.11 Uji Coba Parameter Cover Tanpa Location Map (POP) 81
Tabel 5.12 Uji Coba Parameter Payload Location Map (POP)..82
Tabel 5.13 Uji Coba Parameter Payload Tanpa Location Map (POP).....84
Tabel 5.14 Nilai MSE pada Uji Coba Performa.....90
Tabel 5.15 Nilai PSNR pada Uji Coba Performa.....90
Tabel 5.16 Correlation Cover-audio dan Stego-Audio.....92
Tabel 5.17 Correlation Payload dan Recovery-payload.....92
Tabel 5.18 Waktu Penyisipan Payload pada Uji Coba Performa 94
Tabel 5.19 Waktu Ekstraksi Payload pada Uji Coba Performa...94
Tabel 5.20 Pemakaian Memori saat Proses Penyisipan Payload pada Uji Coba Performa.....96
Tabel 5.21 Pemakaian Memori saat Ekstraksi Payload pada Uji Coba Performa.....97
Tabel 5.22 Nilai MSE pada Uji Coba Parameter Cover.....99
Tabel 5.23 Nilai PSNR pada Uji Coba Parameter Cover.....99

Tabel 5.24 Correlation Cover-audio dan Stego-Audio Uji Coba Parameter Cover 101
Tabel 5.25 Correlation Payload dan Recovery-payload Uji Coba Parameter Cover 101
Tabel 5.26 Waktu Penyisipan Payload pada Uji Coba Parameter Cover 103
Tabel 5.27 Waktu Ekstraksi Payload pada Uji Coba Parameter Cover 103
Tabel 5.28 Pemakaian Memori saat Proses Penyisipan Payload pada Uji Parameter Cover 106
Tabel 5.29 Pemakaian Memori saat Ekstraksi Payload pada Uji Coba Parameter Cover 106
Tabel 5.30 Nilai MSE pada Uji Coba Parameter Payload 109
Tabel 5.31 Nilai PSNR pada Uji Coba Parameter Payload 109
Tabel 5.32 Correlation Cover-audio dan Stego-Audio Uji Coba Parameter Payload 111
Tabel 5.33 Correlation Payload dan Recovery-payload Uji Coba Parameter Payload 111
Tabel 5.34 Waktu Penyisipan Payload pada Uji Coba Parameter Payload 113
Tabel 5.35 Waktu Ekstraksi Payload pada Uji Coba Parameter Payload 114
Tabel 5.36 Pemakaian Memori saat Proses Penyisipan Payload pada Uji Parameter Payload 116
Tabel 5.37 Pemakaian Memori saat Ekstraksi Payload pada Uji Coba Parameter Payload 116

DAFTAR KODE SUMBER

Pseudocode 4.1 Fungsi openAudio 43
Pseudocode 4.2 Fungsi opentext 43
Pseudocode 4.3 Fungsi writeAudio 43
Pseudocode 4.4 Fungsi writeText 44
Pseudocode 4.5 Fungsi plot 44
Pseudocode 4.6 Fungsi rever_smple 45
Pseudocode 4.7 Fungsi irrever_smple 46
Pseudocode 4.8 Fungsi MSE 46
Pseudocode 4.9 Fungsi PSNR 47
Pseudocode 4.10 Fungsi Correlation 47
Pseudocode 4.11 Fungsi consoleprint 48
Pseudocode 4.12 Fungsi memory_usage 48
Pseudocode 4.13 Fungsi toascii 49
Pseudocode 4.14 Fungsi tobase3 50
Pseudocode 4.15 Fungsi tochar 50
Pseudocode 4.16 Fungsi revbase3 51
Pseudocode 4.17 Fungsi NoLMRDE 53
Pseudocode 4.18 Fungsi cal_h 53
Pseudocode 4.19 Fungsi h' 54
Pseudocode 4.20 Fungsi difference2 54
Pseudocode 4.21 Fungsi new_smp_x 54
Pseudocode 4.22 Fungsi new_smp_y 55
Pseudocode 4.23 Fungsi int_average 55
Pseudocode 4.24 Fungsi RDE 57
Pseudocode 4.25 Fungsi new_sample_x 57
Pseudocode 4.26 Fungsi new_sample_y 57
Pseudocode 4.27 Fungsi DecNoLMRDE 58
Pseudocode 4.28 Fungsi recovery_h" 59
Pseudocode 4.29 Fungsi DecodeRDE 60
Pseudocode 4.30 Fungsi recovery_I 61
Pseudocode 4.31 Fungsi recovery_h" 61
Pseudocode 4.32 Fungsi recovery_msg 61
Pseudocode 4.33 Fungsi recovery_sample_x 62

Pseudocode 4.34 Fungsi recovery_sample_y.....62
Pseudocode 4.35 Fungsi NoLMSVM 63
Pseudocode 4.36 Fungsi DecLMSVM.....63
Pseudocode 4.37 Fungsi SVM 65
Pseudocode 4.38 Fungsi BlockMsgSVM.....66
Pseudocode 4.39 Fungsi DecodeSVM 67
Pseudocode 4.40 Fungsi RevBlockMsgSVM 68

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pertukaran informasi melalui jaringan publik merupakan hal yang sering dilakukan masyarakat pada umumnya. Disisi lain permasalahan keamanan informasi menjadi sesuatu yang perlu diperhatikan. Seiring dengan perkembangan teknologi yang semakin pesat, maka dibutuhkan sistem keamanan yang aman untuk pengamanan informasi dari ancaman pencuri informasi. Hal ini sangat penting agar informasi yang dikirimkan tidak dapat diketahui dan disalah gunakan oleh pihak-pihak yang tidak diinginkan dan tidak bertanggung jawab.

Ada berbagai macam teknik dan teknologi yang digunakan untuk melakukan pengamanan informasi seperti enkripsi dan *data hiding* (steganografi). Steganografi berasal dari bahasa yunani yaitu "steganos" yang berarti "tersembunyi atau terselubung", dan "graphein" yang berarti "menulis". Dengan kata lain, steganografi adalah ilmu menyembunyikan informasi dengan suatu cara sehingga selain pengirim dan penerima tidak ada yang mengetahui dan menyadari adanya informasi rahasia yang disisipkan pada media[1].

Berbagai macam media dapat digunakan dalam penyisipan pesan rahasia, seperti teks, citra, audio maupun video. Salah satu media yang digunakan dalam menyembunyikan informasi adalah file audio. Penyisipan pesan rahasia dengan media audio sangat rentan mengalami kerusakan jika file tersebut mengalami perubahan seperti kompresi, *editing, converting* dan sebagainya. Oleh karena itu file audio cocok digunakan sebagai media penyisipan pesan rahasia. Karena dapat dengan mudah mengetahui kerusakan dan keaslian sebuah data.

Dalam tugas akhir ini, penulis mengusulkan sebuah teknik steganografi yang dapat mengoptimalkan proses penyisipan pesan dan keluaran hasil pengolahan dengan kombinasi metode *Reduced*

Difference Expansion[2]–[4] dan *Sample Value Modification* [5], [6]. Dengan menggunakan kombinasi kedua metode pada audio ini diharapkan dapat menigkatkan tingkat keamanan informasi, kualitas serta kapasitas dan proses penyisipan pesan yang lebih efisien sehingga didapatkan hasil yang optimal.

1.2 Rumusan Masalah

Tugas akhir ini mengangkat beberapa rumusan masalah sebagai berikut:

1. Bagaimana menyisipkan pesan serta meningkatkan kapasitas penyimpanan pesan dengan *Modulus Function* pada media audio?
2. Bagaimana menerapkan metode *Reduced Difference Expansion* dan *Sample Value Modification* pada media audio?
3. Bagaimana mengkombinasikan metode *Reduced Difference Expansion* dan *Sample Value Modification* sehingga dapat meningkatkan tingkat keamanan informasi, kualitas, kapasitas serta proses penyisipan pesan yang lebih efisien pada media audio?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada Tugas Akhir ini memiliki batasan sebagai berikut:

1. Bahasa pemrograman yang digunakan untuk menganalisa serangan *copy-move* ini adalah Python dengan distribusi *package* Anaconda.
2. Masukan dari implementasi ini adalah file audio berekstensi .wav.
3. Keluaran dari implementasi ini adalah file Stego audio.
4. Tidak dapat mendeteksi pesan rahasia pada file audio yang telah dilakukan proses *compressing*, *editing*, *converting* dan proses perubahan data lainnya.

5. Metode yang digunakan untuk proses penyisipan pesan rahasia adalah *RDE* dan *SVM*.
6. Perangkat lunak yang digunakan adalah PyCharm Professional 2016.2.3 sebagai IDE.

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah sebagai berikut:

1. Melakukan pengamanan data melalui penyisipan pesan rahasia ke dalam file audio.
2. Melakukan implementasi modifikasi metode *Reduced Difference Expansion* dan *Sample Value Modification* yang efisien sehingga hasil audio dari steganografi tidak mengalami perubahan yang signifikan.
3. 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. Memberikan manfaat di bidang *data hiding* terutama pada bidang *steganography* yang dapat digunakan untuk proses pengamanan informasi melalui media audio secara efisien.
2. Menghasilkan aplikasi steganografi dengan memanfaatkan metode steganografi yang diusulkan.

1.6 Metodologi

Tahapan-tahapan yang dilakukan dalam pengerjaan Tugas Akhir ini adalah sebagi berikut.

1.6.1 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.

1.6.2 Studi Literatur

Pada tahap ini dilakukan untuk mencari informasi dan studi literatur apa saja yang dapat dijadikan sebagai referensi untuk membantu pengerjaan Tugas Akhir ini. Tahap ini merupakan tahap untuk memahami semua metode yang akan dikerjakan, sehingga memberi gambaran selama pengerjaan Tugas Akhir. Informasi didapatkan dari buku dan literatur yang berhubungan dengan metode yang digunakan. Informasi yang dicari adalah metode *RDE, SVM*, dan metode-metode statistika seperti mean dan variance serta metode-metode untuk melakukan pengolahan audio.

1.6.3 Implementasi Perangkat Lunak

Implementasi merupakan tahap untuk mengimplementasikan metode-metode yang sudah diajukan pada proposal Tugas Akhir. Untuk membangun algoritma yang telah dirancang sebelumnya, implementasi dilakukan dengan menggunakan suatu perangkat lunak yaitu Python versi 2.7 sebagai bahasa pemrograman utama dengan distribusi *package* Anaconda dan PyCharm Edu 3.0.1 sebagai IDE.

1.6.4 Pengujian dan Evaluasi

Pada tahap ini metode yang telah diimplementasikan akan diuji coba dengan menggunakan data uji coba yang ada. Uji coba

dilakukan dengan menggunakan suatu perangkat lunak dengan tujuan mengetahui kemampuan metode yang digunakan dan mengevaluasi hasil Tugas Akhir dengan jurnal pendukung yang digunakan. Hasil evaluasi juga mencakup kompleksitas, parameter dan performa dari metode yang telah diimplementasikan dalam proses penyisipan pesan atau pengamanan informasi pada audio.

1.6.5 Penyusunan Buku

Pada tahap ini disusun buku sebagai dokumentasi dari pelaksanaan Tugas Akhir yang mencangkup seluruh konsep, teori, implementasi, serta hasil yang telah dikerjakan.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan Tugas Akhir adalah sebagai berikut:

1. Bab I. Pendahuluan
   Bab ini berisikan penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan Tugas Akhir.
2. Bab II. Tinjauan Pustaka
   Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Tugas Akhir ini. Secara garis besar, bab ini berisi tentang metode *DE, RDE, SVM, Correlation, Similarity Index Modulation, MSE, SNR* dan *PSNR.*
3. Bab III. Perancangan Perangkat Lunak
   Bab ini berisi pembahasan mengenai perancangan dari metode *RDE* dan *SVM* yang digunakan untuk proses penyisipan pesan atau informasi dan perancangan *GUI* atau antarmuka menggunakan Qt.
4. Bab IV. Implementasi
   Bab ini menjelaskan implementasi yang berbentuk *Pseudocode* yang berupa *Pseudocode* dari metode *RDE* dan *SVM*

5. Bab V. Pengujian dan Evaluasi
   Bab ini berisikan hasil uji coba dari metode *RDE* dan *SVM* yang digunakan untuk proses penyisipan pesan yang sudah diimplementasikan. Uji coba dilakukan dengan menggunakan dataset citra yang memiliki kualitas rendah. Hasil evaluasi mencakup perbandingan kompleksitas, parameter dan performa dari masing-masing data masukan.
6. Bab VI. Kesimpulan dan Saran
   Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.
7. Daftar Pustaka
   Bab ini berisi daftar pustaka yang dijadikan literatur dalam Tugas Akhir.
8. Lampiran
   Dalam lampiran terdapat tabel-tabel data hasil uji coba dan kode program secara keseluruhan.

BAB II
TINJAUAN PUSTAKA

Bab ini berisi pembahasan mengenai teori-teori dasar yang digunakan dalam Tugas Akhir. Teori-teori tersebut diantaranya adalah metode *DE, RDE, SVM* dan beberapa teori lain yang mendukung pembuatan Tugas Akhir. Penjelesan ini bertujuan untuk memberikan gambaran secara umum terhadap program yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 Steganografi

Steganografi merupakan salah satu cara dalam pengamanan informasi untuk mencegah deteksi pada informasi yang disembunyikan. Steganografi berasal dari bahasa yunani yaitu "steganos" yang berarti "tersembunyi atau terselubung", dan "graphein" yang berarti "menulis". Dengan kata lain, steganografi adalah ilmu menyembunyikan informasi dengan suatu cara sehingga selain pengirim dan penerima tidak ada yang mengetahui dan menyadari adanya informasi rahasia yang disisipkan pada media[1].

Berbicara tentang steganografi belum lengkap tanpa kriptografi. Keduanya bisa disebut "*cousin of spycraft family"*[1]. Kriptografi mengacak sebuah pesan sehingga tidak dapat dimengerti isi dari pesan tersebut. Sedangkan steganografi menyembunyikan pesan pada medium *cover* sehingga tidak terlihat dan tidak menyadari keberadaan pesan didalamnya Steganografi sendiri sebenarnya tidak menjamin keamanan data, namun bukan berarti metode enkripsi yang sederhana. Jika steganografi dan enkripsi dikombinasikan maka akan menghasilkan metode enkripsi yang lebih aman. Jika pesan yang akan disisipkan dilakukan proses enkripsi terlebih dahulu kemudian dilakukan proses penyisipan pada medium *cover*, maka akan menghasilkan sebuah metode pengamanan yang lebih aman. Pada kriptografi, bila pesan yang

telah dienkripsi dicuri oleh pencuri pesan maka pencuri pesan akan mengetahui bahwa pesan tersebut telah dienkripsi sebelumnya. Namun dengan Steganografi pencuri pesan tidak akan menyadari adanya pesan rahasia pada medium *cover* tersebut.

2.2 *Difference Expansion*

Algoritma *Difference Expansion (DE)* memanfaatkan pasangan piksel yang saling bertetangga. Metode ini mampu menyediakan kapasitas yang lebih banyak pada medium penyimpanan informasi dengan tetap menjaga kualitas medium yang dihasilkan dengan perhitungan kompleksitas yang kecil[3]. *Difference expansion* menggunakan metode "*reversible integer transform"*[3]. Adapun kekurangan dari metode ini, diantaranya sulitnya menemukan pasangan pixel yang memiliki selisih yang kecil. Sehingga kualitas file steganografi yang dihasilkan kurang baik. Permasalahan ini nantinya akan diselesaikan dengan metode *Reduce Diffrence Expansion (RDE)*[2].

Sebelum data *payload* (pesan rahasia) disisipkan pada *cover* (file yang akan disisipkan pesan) terlebih dahulu dilakukan perhitungan-perhitungan. Adapun tahap-tahap dari penanaman *payload* ke dalam *cover* sebagai berikut:

1. Hitung *average* (*l*) dan *difference (h)* antara sepasang pixel *x* dan *y* menggunakan Persamaan 2.1.

$$l = \left\lfloor \frac{x+y}{2} \right\rfloor , \quad h = x - y \tag{2.1}$$

2. Masukkan data *payload* yang sudah diubah dalam binary ke dalam nilai *h* menggunakan Persamaan 2.2

$$h' = 2 \times h + b \tag{2.2}$$

3. Nilai *h'* harus memenuhi Persamaan 2.3 untuk mengetahui apakah sepasang pixel itu tidak *underflow* atau *overflow.*

$$h' = \begin{cases} |h'| \leq 2 \times (255 - l) & if \ 128 \leq l \leq 255 \\ |h'| \leq 2 \times l + 1 & if \ 0 \leq l \leq 127 \end{cases} \quad (2.3)$$

4. Jika memenuhi salah satu Persamaan 2.3 maka dapat diperoleh sepasang *pixel* baru lewat Persamaan 2.4

$$x' = l + \left\lfloor \frac{h' + 1}{2} \right\rfloor , \ y' = l - \left\lfloor \frac{h'}{2} \right\rfloor \quad (2.4)$$

2.3 *Reduced Difference Expansion*

Reduce Diffrence Expansion (RDE) adalah metode *reversible* pada *data hiding* dalam steganografi[2]. *Reversible* berarti citra dari proses *RDE* ini dapat kembali ke bentuk aslinya setelah terjadi proses ekstraksi data[4]. *RDE* merupakan pegembangan dari metode *DE* (*Difference Expansion*)[3]. Adapun proses dari penyisipan *RDE*[2] sebagai berikut:

1. *RDE* melakukan reduksi sebelum melakukan proses penyisipan bit dengan menggunakan Persamaan 2.5

$$h' = \begin{cases} h & if \ h' < 2 \\ h - 2^{\lfloor \log h \rfloor - 1} & otherwise \end{cases} \quad (2.5)$$

2. Untuk melakukan proses pengembalian kedalam *cover* atau citra awal, maka setiap blok piksel yang telah direduksi akan ditandai dalam sebuah *location map*. Penentuan *location map* dapat dilihat pada Persamaan 2.6

$$LM = \begin{cases} 0 , & if \ 2^{\lfloor \log h' \rfloor} = 2^{\lfloor \log h \rfloor} \ or \ h' = h \\ 1 , & if \ 2^{\lfloor \log h' \rfloor} \neq 2^{\lfloor \log h \rfloor} \end{cases} \quad (2.6)$$

3. Selanjutnya sepasang *pixel* baru dapat diperoleh dengan Persamaan 2.7

$$x' = l + \left\lfloor \frac{h'+1}{2} \right\rfloor, \; y' = l - \left\lfloor \frac{h'}{2} \right\rfloor \tag{2.7}$$

Sedangkan untuk proses ekstraksi dan pengembalian nilai piksel pada *RDE* dapat dibagi kedalam 5 tahap, yaitu:

1. Dapatkan kembali nilai *h'* dengan Persamaan 2.8

$$h' = \begin{cases} h' + 2^{\lfloor \log h' \rfloor - 1}, if\ Location\ map = 0 \\ h' + 2^{\lfloor \log h' \rfloor}, \quad if\ Location\ map = 1 \end{cases} \tag{2.8}$$

2. Hitung nilai *l* dan *h''* dengan Persamaan 2.9

$$l = \left\lfloor \frac{x' + y'}{2} \right\rfloor, \; h" = x' - y' \tag{2.9}$$

3. Selanjutnya ekstrak data *(b)* dengan Persamaan 2.10

$$b = LSB(h"), \quad h' = \left\lfloor \frac{h"}{2} \right\rfloor \tag{2.10}$$

4. Pengembalian sepasang *pixel* dapat dilakukan dengan menghitung nilai *h* terlebih dahulu melalui Persamaan 2.11

$$h = h' - 2^{\lfloor \log h \rfloor - 1} \tag{2.11}$$

5. Barulah didapatkan nilai sepasang *pixel* melalui Persamaan 2.12

$$x = l + \left\lfloor \frac{h+1}{2} \right\rfloor, y = l - \left\lfloor \frac{h}{2} \right\rfloor \tag{2.12}$$

2.4 *Sample Value Modification*

SVM (Sample Value Modification) adalah metode steganografi yang memanipulasi sebuah nilai *sample* (data binary pada audio) untuk menyisipkan data rahasia kedalam *sample* audio memanfaatkan *modulus function* [5].

Adapun tahap-tahap penyisipan data menggunakan SVM diantaranya.

1. Ekstrak file audio kedalam *sample* 16bit.
2. Ubah data rahasia *(b)* kedalam binary data *(0,1)*
3. Hitung *reminder (SR)* dengan Persamaan 2.13.

$$SR = mod(S, 2) \tag{2.13}$$

4. Bandingkan nilai *reminder (SR)* dengan nilai binary dari data rahasia *(b)* dengan Persamaan 2.14 untuk mendapatkan nilai *S'*(nilai sample yang sudah disisipkan data rahasia) dan *location map*[5].

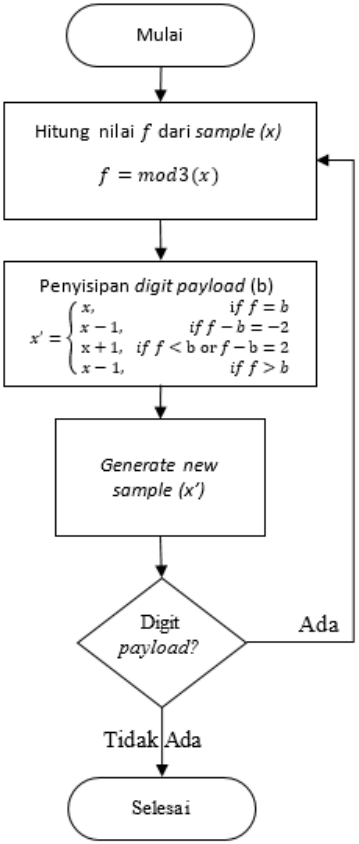
$$S' = \begin{cases} S \ and \ LM = 0, & if \ SR = b \\ S + 1 \ and \ LM = 2, & if \ SR \neq b \ and \ S = 0 \\ S - 1 \ and \ LM = 1, & if \ SR > b \ and \ S \neq 0 \\ S - 1 \ and \ LM = -1, & if \ SR < b \ and \ S \neq 0 \end{cases} \tag{2.14}$$

5. Penyusunan *stego audio* (audio yang telah disisipi data rahasia) kebentuk semula.

Berbeda dengan metode *SVM* yang telah diusulkan oleh Hatem dkk[5], metode *SVM* tanpa *location map* dapat menggunakan metode yang diusulkan oleh V. Nagaraj, V. Vijayalakshmi, dan G. Zayaraz [6] yang sebelumnya telah banyak diimplementasikan pada steganografi citra, metode *PVM (Pixel Value Modification)*. Diketahui $f = mod3(x)$, untuk lebih jelasnya dapat dilihat pada Persamaan 2.15. Skema peyisipan *digit*

payload dapat dilihat pada Gambar 2.1 Skema Penyisipan *Digit Payload* Pada Metode SVM Tanpa *Location Map*

$$x' = \begin{cases} x, & \text{if } f = b \\ x - 1, & if\ f - b = -2 \\ \text{x} + 1, & if\ f < \text{b or } f - \text{b} = 2 \\ x - 1, & if\ f > b \end{cases} \quad (2.15)$$



Gambar 2.1 Skema Penyisipan *Digit Payload* Pada Metode SVM Tanpa *Location Map*

Pada Proses ekstraksi *digit payload* pada metode *SVM* tanpa *Location Map*, didapatkan melalui perhitungan pada Persamaan 2.16, dimana x' adalah nilai *sample* dari *stego-audio.*

$$b = mod3(x') \tag{2.16}$$

2.5 *Correlation*

Correlation adalah suatu nilai yang didapat dari hasil pengujian tingkat kemiripan antara pesan yang disisipkan dengan pesan hasil ekstraksi dari file *stego* audio [5]. Nilai dari *correlation* berkisar antara 0 sampai 1, dimana semakin mendekati 1 semakin tinggi tingkat kemiripan begitu pula sebaliknya. Selain itu *correlation* juga dapat digunakan dalam pengujian tingkat kemiripan antar file. Nilai dari *correlation* dapat dihitung melalui Persamaan 2.17.

$$Correlation = \frac{\sum_{i=1}^{N}(X_i - \overline{X}) * (Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{N}(X_i - \overline{X})^2 * (Y_i - \overline{Y})^2}} \tag{2.17}$$

Dimana X_i adalah *cover-sample* atau *payload* yang akan disisipkan, Y_i adalah *stego-sample* atau hasil ekstraksi *payload* dari *stego-audio*. Sementara $\overline{X}$ dan $\overline{Y}$ adalah nilai *mean* atau rata-rata dari dari keduanya.

2.6 *SIM (Similarity Index Modulation)*

SIM adalah nilai yang didapat dari hasil pengujian tingkat tingkat kemiripan antara file *cover* audio (sebelum proses penyisipan data) dengan file *stego* audio (setelah proses penyisipan) [5]. Output nilai yang dihasilkan antara 0 sampai 1, dimana semakin mendekati 1 maka tingkat kemiripan semakin tinggi begitu pula sebaliknya. Nilai dari *SIM* dapat dihitung melalui Persamaan 2.18.

$$SIM = \frac{\sum_{i=1}^{N} S_i * S_i'}{\sum_{i=1}^{N} (S_i)^2} \quad (2.18)$$

Dimana *N* adalah jumlah dari *sample*, S_i adalah *cover sample* dari audio dan S_i' adalah *stego sample* dari audio.

2.7 *MSE (Mean Square Error)*

MSE adalah nilai yang didapat dari hasil pengujian nilai error kuadrat rata-rata antara file *cover* audio dengan file *stego* audio [5]. Semakin besar nilai MSE, semakin tinggi distorsi yang dihasilkan dan sebaliknya. Nilai dari *MSE* dapat dihitung melalui Persamaan 2.19.

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (S_i - S_i')^2 \quad (2.19)$$

Dimana *N* adalah jumlah dari *sample*, S_i adalah *cover sample* dari audio dan S_i' adalah *stego sample* dari audio.

2.8 *SNR (Signal to Noise Ratio)*

SNR adalah nilai yang didapat dari hasil pengujian perbandingan rasio antara kekuatan sinyal dan kekuatan derau (noise)[7]. Semakin besar nilai *SNR* yang dihasilkan semakin baik kualitas dari *stego* audio yang dihasilkan. Nilai dari *SNR* dapat dihitung melalui Persamaan 2.20.

$$SNR = 10 * log_{10} \left(\frac{\frac{1}{N} \sum_{i=1}^{N} S_i^2}{MSE} \right) \quad (2.20)$$

Dimana N adalah jumlah dari *sample* dan S_i adalah *cover sample* dari audio.

2.9 *PSNR (Peak Signal to Noise Ratio)*

PSNR adalah nilai yang didapat dari hasil pengujian perbandingan antara nilai maksimum dari sinyal dan kekuatan derau (noise)[7]. Semakin besar nilai *PSNR* yang dihasilkan semakin baik kualitas dari *stego* audio yang dihasilkan. Nilai *PSNR* bawah 30 dB menandakan rendahnya kualitas *stego* audio. Sedangkan nilai *PSNR* diatas 40 dB untuk audio 8 bit menandakan tingginya kualitas dari stego audio. Sementara nilai *PSNR* untuk audio 16 bit dengan kualitas audio yang tinggi berkisar diatas 60dB[8]. Nilai dari *PSNR* dapat dihitung melalui Persamaan 2.21.

$$PSNR = 10 * log_{10} \frac{\left(2^b - 1\right)^2}{MSE} \tag{2.21}$$

Dimana b adalah nilai bit audio yang digunakan (8bit, 16bit dan sebagainya). Jika menggunakan audio 16 bit, maka dapat dihitung $2^{16} - 1$ senilai dengan 65535. Nilai ini senilai dengan nilai maksimal dari suatu *sample.*

2.10 Python

Python merupakan Bahasa pemrograman yang *powerfull* dan cepat, dapat dipadu padankan dengan bahasa pemrograman yang lainnya dan dapat beroperasi di hampir semua *platform* seperti UNIX, Mac ataupun Windows [9]. Selain itu python juga merupakan bahasa pemrograman yang mudah dipelajari dan *opensource*. Ada beberapa alasan mengapa banyak *programmer* menggunakan bahasa ini. Diantaranya, python merupakan bahasa yang mudah dipahami dan dipelajari meskipun oleh *programmer* pemula ataupun *programmer* yang telah memiliki dasar bahasa pemrograman yang lain[9].

Python juga menyediakan *Python Package Index (PyPI)* yang berisi kumpulan *module* dari perangkat lunak untuk bahasa pemrograman python[10]. Jumlah *package* yang ada saat ini sejumlah 111162 *package*. Dengan banyaknya *package* yang telah disediakan, *programmer* dapat dengan mudah memakai dan mengimplementasikan idenya dengan python. Python juga dapat diaplikasikan pada *web & internet development*, akses basis data *(database access)*, desktop *GUIs*, *Scientific & Numeric*, pemrograman jaringan, perangkat lunak dan pengembangan game.

2.11 Anaconda

.

Anaconda adalah *open data science platform* yang dibangun menggunakan bahasa pemrograman python[11]. Lebih dari 4,5 juta pengguna, Anaconda menjadi *data science ecosystem* paling terkenal dan terpercaya di dunia. Anaconda menyediakan lebih dari 1000 paket *data science* untuk mengelolah data menjadi informasi yang diperlukan[11]. Versi Anaconda *open-source* menyediakan distribusi dengan performa tingkat tinggi dari bahasa pemrograman Python[9] dan R[12] yang berisikan lebih dari 100 paket paling popular dari Python, R dan Scala[13] untuk *data science*. Dengan Anaconda pengguna dapat mengakses lebih dari 720 paket yang dapat dengan mudah dipasang dengan conda. Anaconda dapat digunakan di berbagai *platform* seperti Windows, Linux dan Mac [14].

2.12 SciPy

SciPy merupakan singkatan dari *Scientific Python*. SciPy adalah pustaka perangkat lunak bahasa pemrograman Python bersifat *Open Source* untuk matematika, sains dan teknik[15]. SciPy mempunyai beberapa paket diantaranya adalah NumPy untuk paket basis ke N *dimensional array*, SciPy Library untuk pustaka dasar untuk komputasi ilmiah, Matplotlib untuk plot 2D,

IPython untuk penyempurnaan konsol interaktif, Sympy untuk simbol matematis dan pandas untuk struktur data dan analisis.

2.13 NumPy

NumPy atau *Numeric Python* merupakan pustaka perangkat lunak bahasa pemrograman Python bersifat *open source* yang digunakan untuk pengolahan ilmiah matematika. Selain digunakan untuk hal ilmiah, NumPy juga bisa digunakan untuk *container* multidimensi yang efisien untuk data generik. Tipe data *arbitrary* (tipe data yang dapat disesuaikan dengan keinginan) dapat didefinisikan yang memungkinkan NumPy secara lancar dan cepat mengintegrasikan dengan berbagai tipe basis data[17].

2.14 Matplotlib

Matplotlib merupakan sebuah library Python yang mendukung untuk menghasil plot 2D dari berbagai bentuk format dan jenis data. Matprotlib dapat digunakan dengan bahasa pemrograman python yang berlisensi pada *Python Sotfware Foundation (PSF)*[18]. Matplotlib mencoba membuat sesuatu yang mudah menjadi mudah dan sesuatu yang sulit menjadi mungkin diterapkan. Matplotlib dapat membuat plot, histogram, *power spectra, bar chart, errorcharts, scatterplots* dan sebagainya hanya dengan beberapa kode baris[19]. Matplotlib bersifat *open source* yang banyak digunakan untuk pengolahan ilmiah matematika dan ditampilkan dalam bentuk grafik.

2.15 PyQt

Qt merupakan sebuah IDE (*Integrated Development Environment*) yang dibuat pada tahun 1996. Qt adalah *toolkit* untuk membantu pengembangan aplikasi yang memiliki tampilan grafis yang dapat berjalan di berbagai *platform*. Sedangkan PyQt merukapan pustaka perangkat lunak Qt yang dapat membantu mengembangkan aplikasi yang memiliki tampilan grafis dengan

menggunakan bahasa pemrograman Python [20]. PyQt berisi lebih dari 620 kelas yang mencakup antarmuka pengguna grafis, penanganan XML, komunikasi jaringan, database SQL, penjelajahan web dan teknologi lainnya yang tersedia di Qt.

BAB III
PERANCANGAN PERANGKAT LUNAK

Bab ini membahas mengenai perancangan dan pembuatan sistem perangkat lunak. Sistem perangkat lunak yang dibuat pada Tugas Akhir ini adalah proses pengamanan pesan rahasia yang disisipkan kedalam media audio menggunakan penggabungan metode *RDE* dan *SVM*.

3.1 Data

Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan perangkat lunak untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan.

3.1.1 Data Masukan

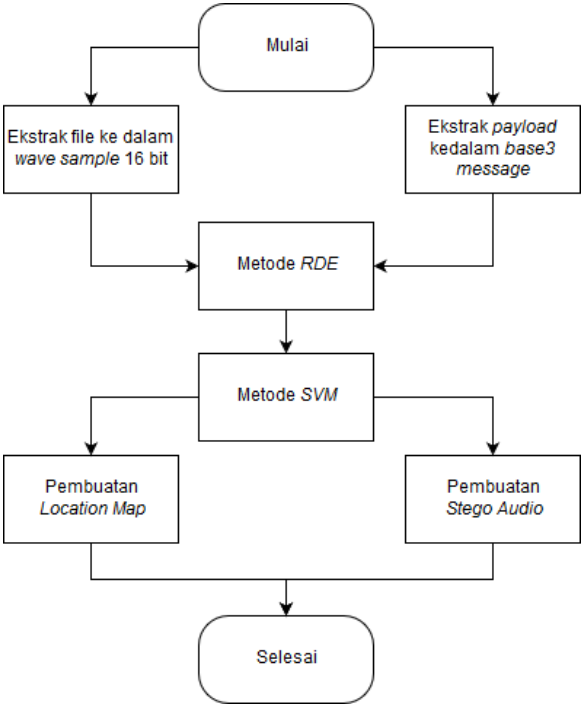
Data masukan adalah data yang digunakan untuk proses penyisipan pesan baik itu data *cover audio* maupun *payload.* Data *cover* yang digunakan berupa data audio dengan ekstensi file .wav 16 bit. Sedangkan data *payload* berupa data teks berekstensi .txt.

3.1.2 Data Keluaran

Setelah data masukan diproses menggunakan metode *RDE* dan *SVM*, program mengeluarkan 2 buah data yaitu *stego-audio* dan *location map. Stego-audio* merupakan data hasil penyisipan data file *payload* yang telah tertanam pesan rahasia didalamnya dengan ekstensi file yang sama seperti data *cover* (.wav). Sedangkan *location map*, merupakan peta lokasi berekstensi .txt dimana pada sample keberapakah informasi disisipkan.

3.2 Dekripsi Umum Sistem

Metode yang akan dikembangkan pada tugas akhir ini adalah kombinasi antara algoritma *RDE*[2] dan *SVM*[5]. Dimana sepasang *sample* dari file audio yang telah dilakukan proses *sampling* akan disisipkan digit *payload (b)* menggunakan metode *RDE*, setelah itu sepasang *sample* tersebut disisipi lagi pesan rahasia menggunakan metode *SVM*. Hal ini dilakukan untuk menghasilkan kualitas file *stego audio* yang lebih baik dan menambah kapasitas penyimpanan pesan rahasia pada file *cover audio* yang akan disisipkan. Untuk lebih jelasnya lihat Gambar 3.1 Skema Penyisipan *Digit Payload* ke Dalam *Cover Sample*.



Gambar 3.1 Skema Penyisipan *Digit Payload* ke Dalam *Cover Sample*

Berbeda dalam penelitian sebelumnya[5] data rahasia atau *payload* pada tugas akhir ini diproses dengan modifikasi *modulus function* dengan nilai modulus 3. Dimana hasil ekstraksi *payload* yang didapat akan menghasilkan pesan *base3* (0,1,2). Hal ini dilakukan untuk menekan jumlah data *payload* yang akan disisipkan. Misalnya diketahui nilai ASCII untuk karakter *"a"* adalah 97. Jika menggunakan modulus 2 maka didapatkan nilai data *payload, b*= [0, 1, 1, 0, 0, 0, 0, 1]. Sedangkan menggunakan modulus 3 didapatkan nilai data *payload, b*= [1, 0, 1, 2, 1]. Jika dibandingkan, jumlah data *payload* menggunakan mod2 *n(b)* = 8 sedangkan dengan mod3 *n(b)* = 5. Sehingga dapat disimpulkan pengolahan data *payload* dengan modulus 3 jauh lebih efektif dibandingkan dengan modulus 2. Dengan berkurangnya jumlah *payload* dapat meningkatkan kapasitas penyimpanan data pada file cover audio.

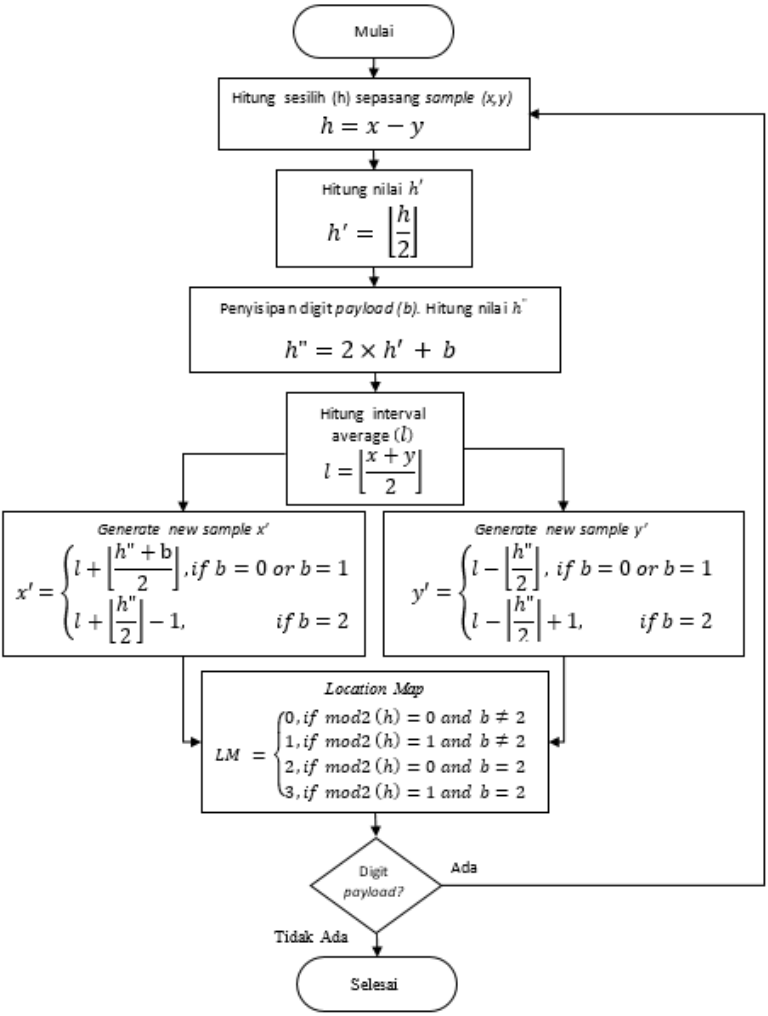
3.3 Perancangan Modifikasi Metode *RDE* dengan *Location Map*

Metode yang digunakan pada tugas akhir ini berbeda dengan metode yang diusulkan oleh Lou dkk[4]. Perbedaan terletak pada pengolahan data *payload* yang akan disisipkan. Metode yang digunakan pada tugas akhir ini menggunakan *base3* sedangkan metode yang diusulkan oleh Lou dkk[4] menggunakan *base2* (*binary bit*).

Dengan adanya perbedaan penggunaan data *payload* mengakibatkan perbedaan pengolahan metode *RDE* pada tugas akhir ini. Berbeda dengan sebelumnya[2], metode *RDE* pada tugas akhir ini menggunakan Persamaan 3.1 untuk menghasilkan nilai sepasang *sample* baru.

$$x' = \begin{cases} l + \left\lfloor \frac{h'' + \mathrm{b}}{2} \right\rfloor, & if\ b = 0\ or\ b = 1 \\ l + \left\lfloor \frac{h''}{2} \right\rfloor - 1, & if\ b = 2 \end{cases}$$
$$y' = \begin{cases} l - \left\lfloor \frac{h"}{2} \right\rfloor, & if\ b = 0\ or\ b = 1 \\ l - \left\lfloor \frac{h"}{2} \right\rfloor + 1, & if\ b = 2 \end{cases} \quad (3.1)$$

Untuk lebih jelasnya dapat dilihat pada Gambar 3.2 Skema Penyisipan *Digit Payload* Menggunakan Metode *RDE*.



Gambar 3.2 Skema Penyisipan *Digit Payload* Menggunakan Metode *RDE*

3.4 Perancangan Modifikasi Metode *SVM* dengan *Location Map*

Pada Metode *SVM* yang digunakan pada tugas akhir ini berbeda pada metode yang diusulkan oleh Hatem dkk[5]. Pada metode kali ini menggunakan pemetaan *sample* ke-dalam *message block yang* dilakukan pada sepasang *sample(x,y)* dengan menggunakan fungsi modulus 3. Adapun pemetaan *message block* dapat dilihat pada Gambar 3.3 Pemetaan *Message Block*.

Message Block 0

0	0
0	2
2	0

Message Block 1

0	1
1	0
1	1

Message Block 2

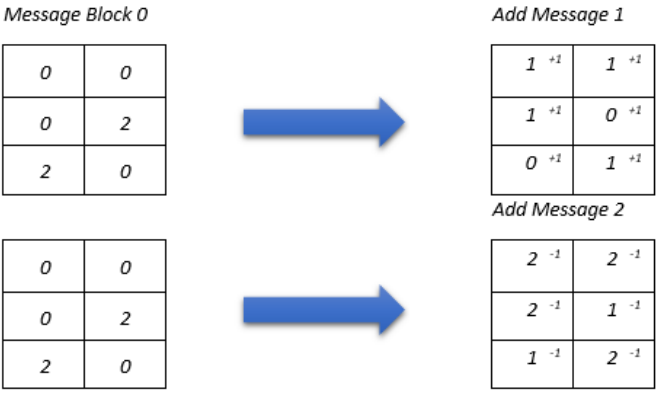
1	2
2	1
2	2

Gambar 3.3 Pemetaan *Message Block*

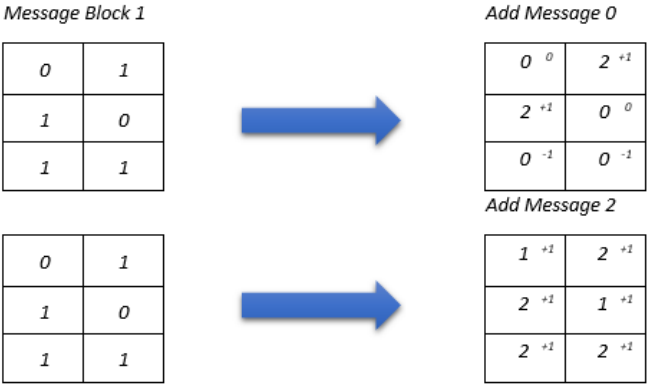
Pemetaan dari message block diatas dikelompokkan berdasarkan pada Persamaan 3.2.

$$Message\ block = \begin{cases} 0, if\ (x + y \neq 1)\ and\ (x\ or\ y = 0) \\ 1, if\ (x + y < 3)\ and\ (x\ or\ y = 1) \\ 2, \qquad\qquad if (x + y > 2) \end{cases} \tag{3.2}$$

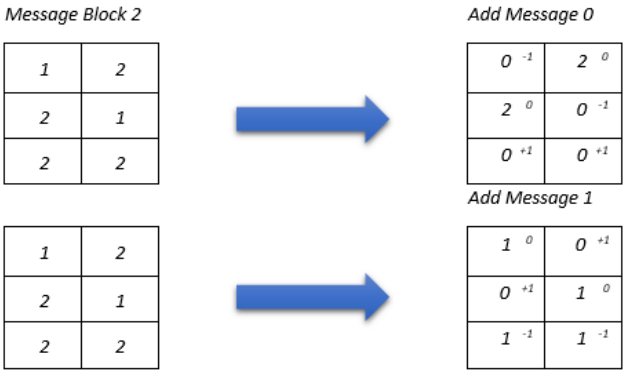
Perubahan sepasang *sample* yang akan disisipi *digit payload* bergantung pada nilai *b* dan *message block* dari sepasang *sample.* Adapun perubahan sepasang *sample* tersebut dapat dilihat pada Gambar 3.4 Penyisipan *Digit Payload* ke Dalam *Message block 0*, Gambar 3.5 Penyisipan *Digit Payload* ke Dalam *Message block 1* dan Gambar 3.6 Penyisipan *Digit Payload* ke Dalam *Message block 2.*



Gambar 3.4 Penyisipan *Digit Payload* ke Dalam *Message block 0*

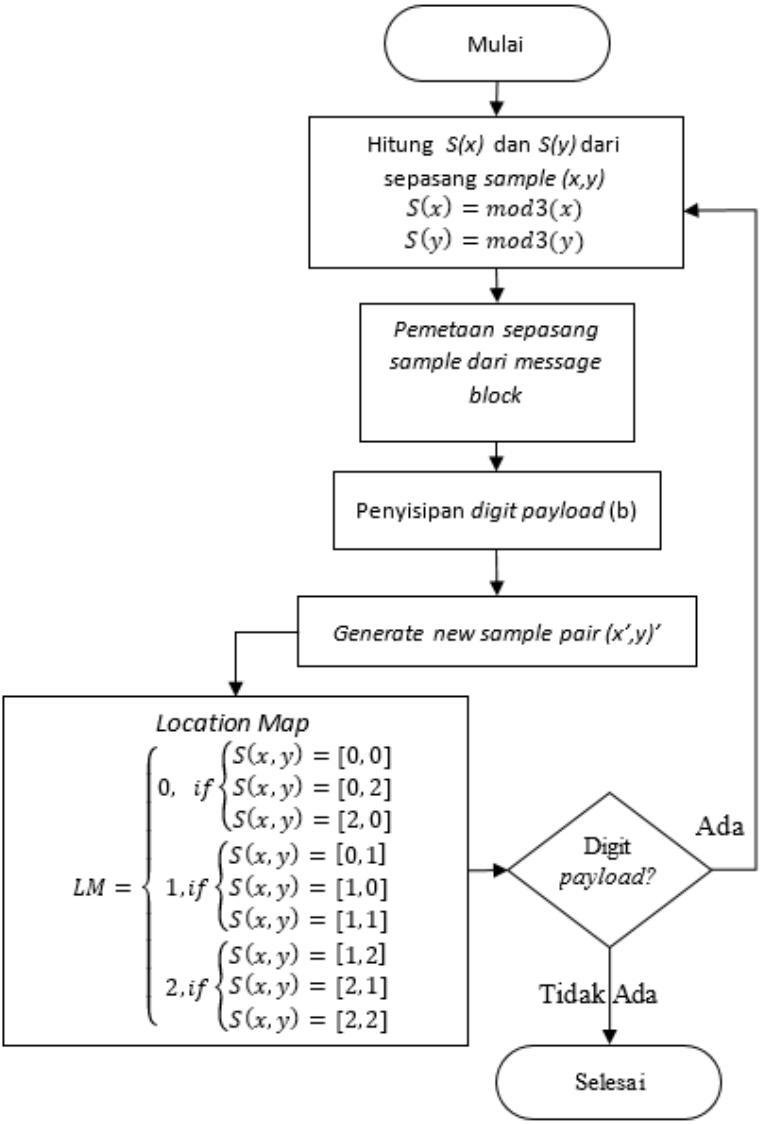


Gambar 3.5 Penyisipan *Digit Payload* ke Dalam *Message block 1*



Gambar 3.6 Penyisipan *Digit Payload* ke Dalam *Message block 2*

Untuk lebih jelasnya, skema penyisipan *digit* payload dapat dilihat pada Gambar 3.7 Skema Penyisipan *Digit Payload* Pada Metode *SVM* Menggunakan Location Map.



Gambar 3.7 Skema Penyisipan *Digit Payload* Pada Metode *SVM* Menggunakan Location Map.

Sebagai contoh, diketahui sepasang *sample(x,y)* masing-masing bernilai, *x = 15788; y = 15127 dan b=1*

Step1: Hitung nilai $S'(x) = mod3(x)$ $S'(y) = mod3(y)$.

$$S'(x) = mod3(15788) = 2$$
$$S'(y) = mod3(15127) = 2$$

Step2: Berdasarkan nilai S'(x) dan S'(y), [2,2] dapat dipetakan, bahwa sepasang *sample(x,y)* termasuk kedalam *message block 2* (Gambar 3.3 Pemetaan *Message Block*).

Step3: Dapatkan nilai sepasang *sample* baru berdasarkan hasil dari pemetaan sepasang *sample (message block 2)* dan nilai *b=1* pada Gambar 3.6 Penyisipan *Digit* Payload ke Dalam *Message block 2*. Berdasarkan Gambar 3.6 Penyisipan *Digit* Payload ke Dalam *Message block 2* maka didapatkan nilai sepasang *sample* sebagai berikut:

$$x' = 15788 - 1 = 15787$$
$$y' = 15127 - 1 = 15126$$

3.5 Perancangan Penyisipan dan Ekstraksi Payload

Metode penyisipan menggunakan teknik *RDE* dan *SVM*, dimana nilai selisih dihitung dari sepasang sample untuk menerapkan metode *RDE*. Setelah metode *RDE* diterapkan, *location map* dibuat dan selanjutnya diterapkanlah metode *SVM*. Pada metode *SVM* sepasang sample dipetakan berdasarkan *message block* yang sudah dijelaskan diatas. Setelah itu barulah *location map* kembali dibuat.

Dengan menggunakan kombinasi dari kedua metode tersebut, *digit* payload yang akan disimpan pada file *cover audio* dapat lebih efisien. Sepasang sample dapat menyimpan dua *digit*

payload. Sehingga dapat meningkatkan kapasitas penyimpanan *digit payload* pada file *cover audio*.

Pada proses ekstraksi bit *payload*, dilakukan secara bergantian menggunakan metode *SVM* dan *RDE*. Setelah proses ekstraksi menggunakan metode *SVM*, barulah proses ekstraksi menggunakan metode *RDE*. Hal ini dilakukan agar nilai dari sepasang *sample stego audio* dapat kembali ke nilai *sample* pada saat penyisipan menggunakan metode *RDE*. Kemudian barulah dilakukan proses pengembalian nilai sample seperti semula *(cover audio)* menggunakan proses ekstraksi dari metode *RDE*.

3.5.1 Perancangan Penyisipan *Payload*

Perancangan penyisipan *digit payload* dilakukan pada sepasang *sample(x,y)*, disisipkan secara bergantian menggunakan metode *RDE* dan *SVM*. Sebelum proses penyisipan, *payload* akan diproses menggunakan fungsi modulus 3, dimana pada metode sebelumnya [2], [5], menggunakan fungsi modulus 2 *(binary bit)*. Sebagai contoh, nilai ASCII karakter ‘a’ adalah 97, maka untuk merubah karakter tersebut ke dalam *base 3 message* digunakan cara sebagai berikut,

97 dibagi 3	=	32 sisa **1**
32 dibagi 3	=	10 sisa **2**
10 dibagi 3	=	3 sisa **1**
3 dibagi 3	=	1 sisa **0**
1 dibagi 3	=	0 sisa **1**

Berdasarkan sisa hasil bagi, maka bilangan base 3 untuk karakter ‘a’ adalah [1, 0, 1, 2, 1].

3.5.2 Perancangan *Location Map*

Perancangan *location map* dibuat secara bergantian pada saat proses penyisipan *digit payload* pada kedua metode *(RDE dan SVM).* Pembuatan *location map* pada metode *RDE* dibuat berdasarkan Persamaan 3.3:

$$LM = \begin{cases} 0, if\ mod2(h) = 0\ and\ b \neq 2 \\ 1, if\ mod2(h) = 1\ and\ b \neq 2 \\ 2, if\ mod2(h) = 0\ and\ b = 2 \\ 3, if\ mod2(h) = 1\ and\ b = 2 \end{cases} \tag{3.3}$$

Sedangkan pembuatan *location map* pada metode *SVM* berdasarkan pemetaan *message block* dari nilai $S'(x, y)$ sepasang *sample(x,y)* dapat dilihat pada Persamaan 3.4.

$$LM = \begin{cases} 0, if \begin{cases} S'(x,y) = [0,0] \\ S'(x,y) = [0,2] \\ S'(x,y) = [2,0] \end{cases} \\ 1, if \begin{cases} S'(x,y) = [0,1] \\ S'(x,y) = [1,0] \\ S'(x,y) = [1,1] \end{cases} \\ 2, if \begin{cases} S'(x,y) = [1,2] \\ S'(x,y) = [2,1] \\ S'(x,y) = [2,2] \end{cases} \end{cases} \tag{3.4}$$

3.5.3 Perancangan Ekstraksi *Payload* dan *Reversible Audio*

Proses ekstraksi *payload* dilakukan menggunakan metode *SVM* dan *RDE* secara berurutan. Pada metode *SVM*, nilai $S'(x', y')$ dihitung terlebih dahulu untuk mengekstrak *digit payload* yang tersimpan pada stego audio. Sebagai contoh sepasang *stego sample(x,y)* masing-masing bernilai, $x' = 15787,\ y' = 15126$. Dengan menggunakan nilai modulus 3, maka didapatkan nilai

$S'(x', y') = [\ 1, 0\ ]$. Setelah itu barulah di petakan berdasarkan *message block* (Gambar 3.3 Pemetaan *Message Block*). Berdasarkan Gambar 3.3 Pemetaan *Message Block* maka untuk nilai $S'(x', y') = [\ 1, 0\ ]$ termasuk kedalam *message block 1*, maka nilai *digit payload b=1*.

Untuk pengembalian nilai sepasang *sample* dilakukan dengan melihat *Location Map* dan nilai $S'(x', y')$. Setelah kedua nilai tersebut didapatkan maka pengembalian nilai sepasang *sample* dapat dilakukan dengan pemetaan berdasarkan Gambar 3.4 Penyisipan *Digit Payload* ke Dalam *Message block 0* Gambar 3.5 Penyisipan *Digit Payload* ke Dalam *Message block 1* dan Gambar 3.6 Penyisipan *Digit Payload* ke Dalam *Message block 2*.

Pada metode *RDE*, proses ekstraksi *digit payload* dilakukan dengan cara melihat *location map* dan nilai $h"$. Untuk lebih jelasnya dapat dilihat pada Persamaan 3.5.

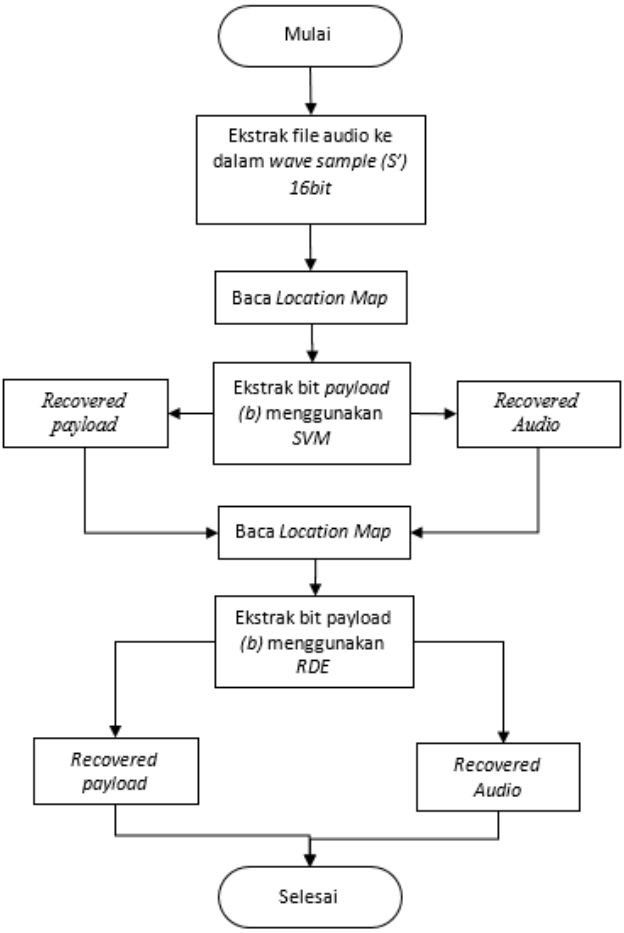
$$b = \begin{cases} 2, & if\ LM > 1 \\ 1, & if\ mod2(h") = 1 \\ 0, & if\ mod2(h") = 0 \end{cases} \tag{3.5}$$

Sedangkan untuk proses penyembalian nilai *sample (Reversible audio)* dapat dilihat pada Persamaan 3.6

$$x = \begin{cases} l + \left\lfloor \frac{h"}{2} \right\rfloor, & if\ mod2(LM) = 0 \\ l + \left\lfloor \frac{h"}{2} \right\rfloor + 1, & if\ mod2(LM) = 1 \end{cases} \tag{3.6}$$

$$y = l - \left\lfloor \frac{h"}{2} \right\rfloor$$

Untuk lebih jelasnya skema ekstraksi *payload* dan *Reversible audio* dapat dilihat pada Gambar 3.8 Skema Ekstraksi *Payload* dan *Reversible Audio*.



Gambar 3.8 Skema Ekstraksi *Payload* dan *Reversible Audio*

Step1: Ekstrak stego audio kedalam sample S' dan baca *location map.*

Step2: Hitung nilai $S(x') = mod3(x')$ $S(y') = mod3(y')$ dari sepasang *stego sample (x',y').*

Step3: Petakan termasuk kedalam *message block* mana dan dapatkan nilai bit *payload (b)* yang telah disisipkan pada *stego audio*. Kembalikan sepasang *stego sample* ke *sample* aslinya dengan memetakan berdasarkan *Location Map* dan Gambar 3.4 Penyisipan *Digit Payload* ke Dalam *Message block 0*Gambar 3.5 Penyisipan *Digit Payload* ke Dalam *Message block 1* dan Gambar 3.6 Penyisipan *Digit Payload* ke Dalam *Message block 2*.

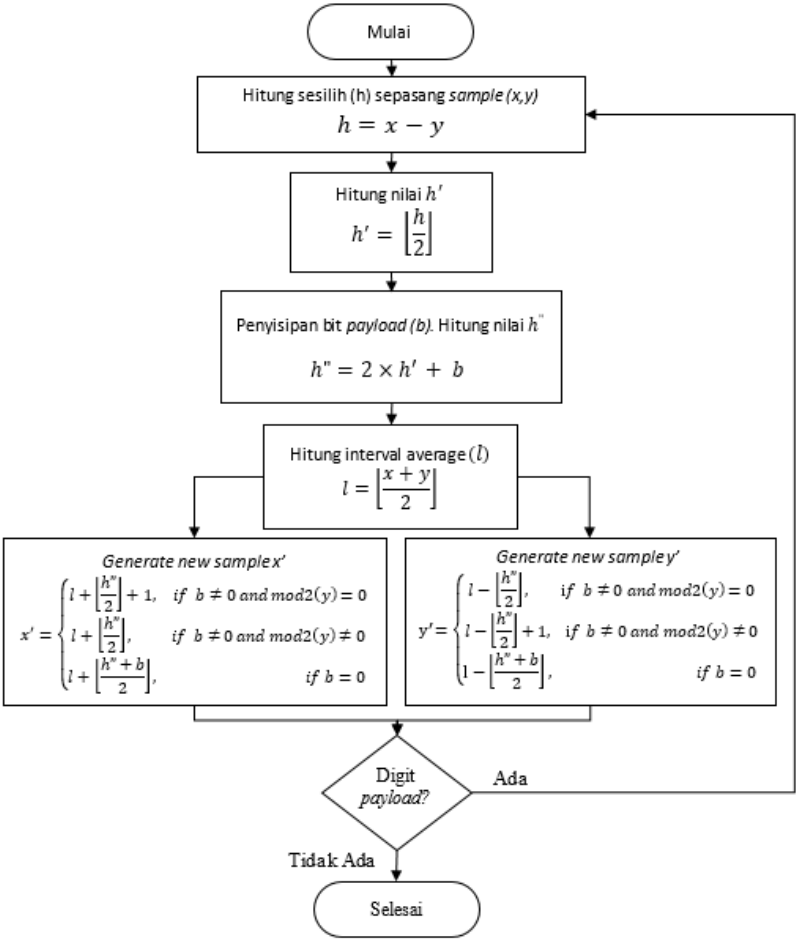
Step4: Baca *location map* dan lakukan ekstraksi bit *payload (b)* menggunakan metode *RDE*. Kembalikan nilai *stego sample* ke *sample* aslinya.

3.6 Perancangan Modifikasi Metode *RDE* Tanpa *Location Map*

Proses penyisipan tanpa *location map* pada metode ini hampir sama dengan proses penyisipan dengan *location map*. Letak perbedaan terdapat pada penyisipan digit untuk menghasilkan nilai baru dari sepasang *sample(x,y).* Untuk lebih jelasnya dapat dilihat pada Persamaan 3.7. Skema penyisipan *payload* dapat dilihat pada Gambar 3.9 Skema Penyisipan *Payload* Pada Metode *RDE* Tanpa *Location Map*.

$$x' = \begin{cases} l + \left\lfloor \frac{h''}{2} \right\rfloor + 1, & \text{if } b \neq 0 \text{ and } mod2(y) = 0 \\ l + \left\lfloor \frac{h''}{2} \right\rfloor, & \text{if } b \neq 0 \text{ and } mod2(y) \neq 0 \\ \mathrm{l} + \left\lfloor \frac{h'' + b}{2} \right\rfloor, & if\ b = 0 \end{cases} \quad (3.7)$$

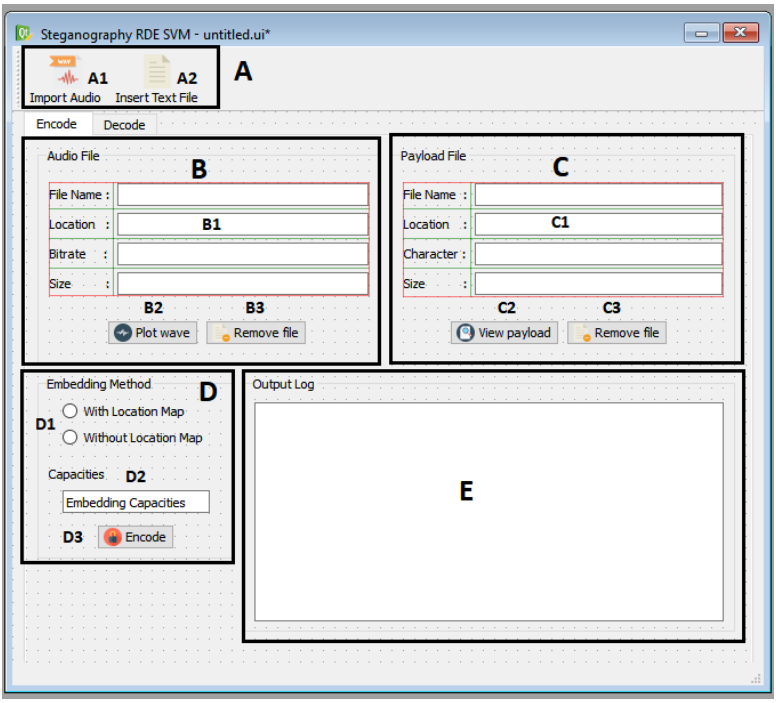
$$y' = \begin{cases} l - \left\lfloor \frac{h''}{2} \right\rfloor, & if \ \ b \neq 0 \ and \ mod2(y) = 0 \\ l - \left\lfloor \frac{h''}{2} \right\rfloor + 1, & if \ \ b \neq 0 \ and \ mod2(y) \neq 0 \\ l - \left\lfloor \frac{h'' + b}{2} \right\rfloor, & if \ b = 0 \end{cases}$$



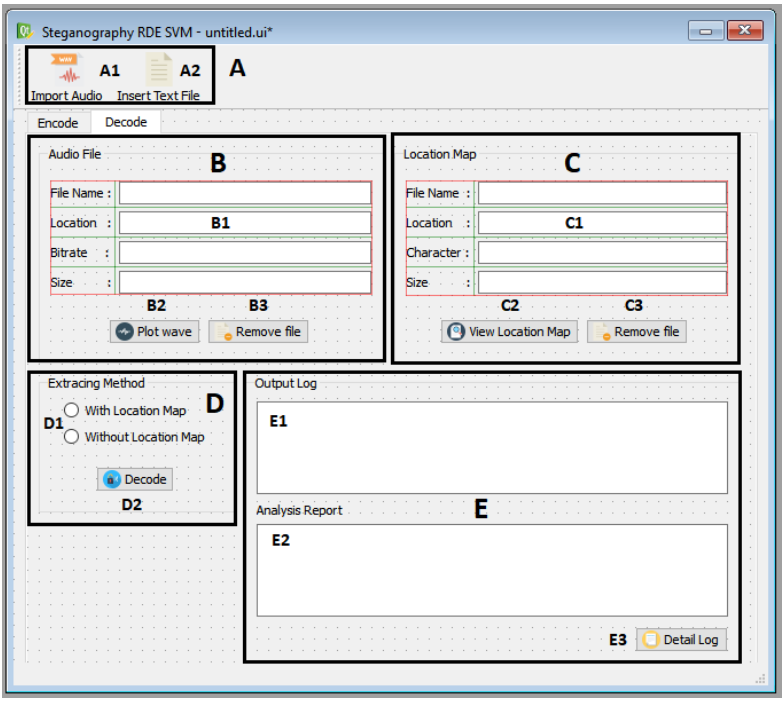
Gambar 3.9 Skema Penyisipan *Payload* Pada Metode *RDE* Tanpa *Location Map*

3.7 Perancangan Antarmuka

Aplikasi antarmuka digunakan untuk mempermudah pengguna dalam melakukan penerapan steganografi pada file audio menggunakan metode *RDE* dan *SVM*. Aplikasi antarmuka yang dibuat pada Tugas Akhir ini menggunakan pustaka Qt pada bahasa pemrograman Python dengan ditribusi pustaka PyQt4. Aplikasi antamuka yang dibangun menggunakan pustaka Qt, merupakan aplikasi *multiplatform* yang dapat berjalan di berbagai sistem operasi seperti Windows, Linux, dan lain-lain. Aplikasi *Qt-Designer* adalah aplikasi yang digunakan untuk mendesain tampilan antarmuka yang dapat dibuat dengan *drag-and-drop* beberapa menu *Widget* kedalam *Main Window* untuk membangun sebuah tampilan antarmuka. Sehingga memudahkan untuk menempatkan *toolbar, menubar, layout*, *button*, *label, tab widget, textviews, form*, *dropdown, textearea,* dan lain sebagainya Keluaran dari aplikasi ini berupa file dengan ekstensi *.ui* yang nantinya dapat dipanggil melalui file Python sebagai program tampilan antarmuka. Gambar 3.10 Desain Tampilan Antarmuka *(Encode)* dan Gambar 3.11 Desain Tampilan Antarmuka *(Decode)*merupakan desain tampilan antarmuka yang dibuat pada Tugas Akhir ini.



Gambar 3.10 Desain Tampilan Antarmuka *(Encode)*



Gambar 3.11 Desain Tampilan Antarmuka *(Decode)*

Adapun komponen yang terdapat dalam Gambar 3.10 Desain Tampilan Antarmuka *(Encode)* dan Gambar 3.11 Desain Tampilan Antarmuka *(Decode)*antara lain:

A. Bagian A merupakan menu *toolbar* untuk memasukkan file audio *(cover* atau *stego audio)* dan file teks. Menu toolbar A1 digunakan untuk memasukkan file audio yang nantinya detail dari audio akan ditampilkan pada bagian B1. Sedangkan pada A2 digunakan untuk memasukkan file teks yang nantinya juga akan ditampilkan detail dari teks yang dipilih pada bagian C1.
B. Bagian B merupakan *groupBox* berupa *formview* dan beberapa tombol yang berfungsi untuk menampilkan detail audio, plot audio dan menghapus file audio. Pada saat awal

aplikasi dijalankan tidak ada detail file audio di bagian B1, setelah file audio dimasukkan dengan menekan tombol A1 dan menampilkan *file explorer*, maka bagian B1 secara otomatis akan menampilkan detail dari file audio yang telah terpilih. Pada bagian B2 merupakan tombol untuk menampilkan hasil plot file audio yang telah dimasukkan sebelumnya. Sedangkan pada bagian B3 merupakan tombol untuk menghapus file audio yang akan digunakan sebagai file cover pada proses penyisipan pesan *(encode)* ataupun file *stego audio* pada proses ekstraksi pesan *(decode)*.

C. Bagian C merupakan *groupBox* berupa *formview* dan beberapa tombol yang berfungsi untuk menampilkan detail file teks, melihat isi dari file teks dan menghapus file teks. Pada saat awal aplikasi dijalankan tidak ada detail file teks di bagian C1, setelah file teks dimasukkan dengan menekan tombol A2 dan menampilkan *file explorer*, maka bagian C1 secara otomatis akan menampilkan detail dari file teks yang telah terpilih. Pada bagian C2 merupakan tombol untuk melihat isi dari file teks yang telah dimasukkan sebelumnya. Sedangkan pada bagian C3 merupakan tombol untuk menghapus file teks yang akan digunakan sebagai *payload* pada proses penyisipan pesan *(encode)* ataupun *location map* pada proses ekstraksi pesan *(decode)*.

D. Bagian D merupakan *groupBox* berupa *radiobutton* dan tombol untuk proses penyisipan pesan ataupun proses ekstraksi pesan. Pada saat awal aplikasi dijalankan tidak ada metode penyisipan atau motode ekstraksi yang terpilih pada bagian D, dengan memilih *radiobutton* pada bagian D1 maka proses penyisipan atau ekstraksi pesan dapat dilakukan. Pada bagian D3 Gambar 3.10 Desain Tampilan Antarmuka *(Encode)* dan D2 Gambar 3.11 Desain Tampilan Antarmuka *(Decode)*merupakan tombol untuk mengeksekusi perintah penyisipan dan ekstraksi pesan

sesuai metode yang telah dipilih pada bagian D1. Bagian D2 pada Gambar 3.10 Desain Tampilan Antarmuka *(Encode)* merupakan *formview* berupa kapasitas pesan yang dapat ditampung pada file *cover* audio.

E. Bagian E merupakan *formview* untuk melihat *log* dari aplikasi setelah perintah-perintah dijalankan. Pada bagian E1 merupakan *log* dari perintah yang telah dijalankan. Sedangkan E2 merupakan *report* dari hasil ekstraksi pesan *(Decode)*. Pada bagian E3 merupakan *detail-log* dari awal proses menyisipan pesan hingga akhir ekstraksi pesan beserta waktu dan beberapa hasil Analisa dari proses steganografi.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi dari perancangan yang telah dijelaskan pada bab sebelumnya. Adapun lingkungan implementasi juga akan dijelaskan pada bab ini. Implementasi berupa *Pseudocode* fungsi-fungsi yang akan digunakan untuk membangun program.

4.1 Lingkungan Implementasi

Lingkungan pengujian pada uji coba steganografi menggunakan metode *RDE* dan *SVM* akan digunakan lingkungan pengujian dengan spesifikasi perangkat keras dan perangkat lunak yang ditunjukkan pada Tabel 4.1 Lingkungan Implementasi Perangkat Lunak.

Tabel 4.1 Lingkungan Implementasi Perangkat Lunak

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	Intel(R) Core(TM) i7-4700HQ CPU @ 2.4 GHz
	Memori	8 GB 1600 MHz DDR3
Perangkat Lunak	Sistem Operasi	Windows 10
	Perangkat Pengembang	PyCharm Edu 3.0.1 2016.2.3

4.2 Implementasi

Pada sub bab implementasi ini menjelaskan mengenai pembangunan perangkat lunak secara detail beserta *Pseudocode* yang digunakan mulai tahap *preprocessing* hingga penyisipan *payload* menggunakan metode *RDE* dan *SVM*. Pada Tugas Akhir

ini, implementasi menggunakan bahasa pemrograman Python. Program yang dibangun menggunakan konsep *Object-Oriented Programming* pada bahasa pemrograman Python.

4.2.1 Kelas *Helper*

Pada kelas Helper terdapat 3 sub kelas yang terdiri dari kelas *IOClass*, kelas *Sample* dan kelas *Report*. Didalam masing-masing kelas terdapat fungsi-fungsi yang akan dijelaskan pada sub-bab 4.2.1.1 , 4.2.1.2 dan 4.2.1.3

4.2.1.1 Implementasi Kelas *IOClass*

Pada subkelas ini menangani segala fungsi yang berkaitan dengan input output, diantaranya adalah pembacaan *cover audio*, pembacaan *location map*, pembacaan *stego audio*, penulisan *stego audio*, *penulisan location map* dan *plotting audio*. Proses implementasi masing-masing fungsi dapat dilihat pada Pseudocode 4.1, Pseudocode 4.2, Pseudocode 4.3, Pseudocode 4.4 dan Pseudocode 4.5

Fungsi *openAudio* digunakan untuk membuka file audio dan mengambil segala informasi yang berkaitan dengan file tersebut. Seperti namafile, *bitrate*, *samples*, nilai minimum dan maksimum *sample*.

Input	Path lokasi file audio
Output	Objek wave dari audio

```
1.  FUNCTION openAudio(path):
2.      wav <- scipy.io.wavfile.read(path)
3.      filename <- path.split("/")[len(path.split("/")
    )-1]
4.      waveObj <- Wave.Wave(filename)
5.      waveObj.bitrate <- wav[0]
6.      samples <- wav[1]
7.      samples <- numpy.array(samples,dtype =numpy.int
    16)
8.      waveObj.samples <- samples
```

```
9.      waveObj.max <- samples.max
10.     waveObj.min <- samples.min
11.     waveObj.shape <- samples.shape
12.     waveObj.type <- samples.dtype
13.     RETURN waveObj
14. ENDFUNCTION
```

Pseudocode 4.1 Fungsi *openAudio*

Fungsi *openText* digunakan untuk membuka file teks baik berupa *payload* maupun *location map*. Masukan dari fungsi ini berupa *path* lokasi file teks dan keluaran berupa *array* berisi *string* yang nantinya dapat digunakan dalam proses penyisipan maupun ekstraksi pesan.

Input	Path lokasi file teks
Output	Data file teks

```
1. FUNCTION openText(path):
2.     message <- open(path).read()
3.     RETURN message
4. ENDFUNCTION
```

Pseudocode 4.2 Fungsi *opentext*

Fungsi *writeAudio* digunakan untuk membuka file audio dan mengambil segala informasi yang berkaitan dengan file tersebut. Diantaranya adalah namafile, *bitrate, samples*, nilai minimum dan maksimum *sample.*

Input	Path lokasi file audio, bitrate dan sample audio
Output	File audio .wav

```
1. FUNCTION writeAudio(path,rate,data):
2.     data <- np.array(data,dtype =np.int16)
3.     scipy.io.wavfile.write(path,rate,data)
4. ENDFUNCTION
```

Pseudocode 4.3 Fungsi *writeAudio*

Fungsi *writeText* digunakan untuk menyimpan file teks baik berupa *location map* maupun pesan hasil ekstraksi dari *stego-*

audio. Masukan dari fungsi ini berupa teks bertipe *string* dan *path* lokasi file teks. Sedangkan keluaran berupa file teks berekstensi *.txt* yang akan disimpan sesuai *path* lokasi file.

Input	Path lokasi file teks
Output	File teks *.txt*

```
1. FUNCTION writeText(text,path):
2.     with open(path, "w") as text_file:
3.         text_file.write("{0}".format(text))
4. ENDFUNCTION
```

Pseudocode 4.4 Fungsi *writeText*

Fungsi *plot* digunakan untuk *plotting sample-audio.* Masukan dari fungsi ini berupa *path* lokasi file teks. Sedangkan keluaran berupa hasil *plotting sample-audio.*

Input	Path lokasi file audio
Output	Plot file audio

```
1.  FUNCTION plot(path):
2.      spf <- wave.open(path,'r')
3.      signal <- spf.readframes(-1)
4.      signal <- np.fromstring(signal, 'int16')
5.      fs <- spf.getframerate()
6.      IF spf.getnchannels() = 2:
7.          OUTPUT 'Just mono files'
8.          sys.exit(0)
9.      ENDIF
10.     Time=np.linspace(0, len(signal)/fs, num=len(sig
      nal))
11.     plt.figure(path)
12.     plt.title('Signal Wave '+path)
13.     plt.plot(Time,signal)
14.     plt.show()
15. ENDFUNCTION
```

Pseudocode 4.5 Fungsi *plot*

4.2.1.2 Implementasi Kelas *Sample*

Pada subkelas ini menangani fungsi yang berkaitan dengan proses *reversible sample* dan *irreversible sample*. Penggunaan fungsi *rever_smpl* digunakan pada saat penerapan steganografi menggunakan *Location Map*. Dimana didalam fungsi tersebut memanggil fungsi *decodeRDE* untuk pengembalian nilai *sample (reversible-sampel)* seperti sebelum penyisipan *payload*. Sedangkan fungsi *irrever_smpl* digunakan pada saat penerapan steganografi tanpa *Location Map.* Dimana didalam fungsi tersebut memanggil fungsi *DecNoLMRDE* untuk proses ekstraksi *payload*. Proses implernentasinya dapat dilihat pada Pseudocode 4.6 dan Pseudocode 4.7.

Fungsi *rever_smple* digunakan untuk pengembalian nilai *sample* setelah proses penyisipan pesan. Masukan dari fungsi ini berupa *path* lokasi file *stego-audio,* sedangkan keluaran berupa pengembalian *sample* dari proses ekstraksi dan pesan dalam bentuk *base3*.

Input	Path lokasi file audio
Output	*Sample recovery audio* dan pesan *base3*

```
1. FUNCTION rever_smple(path):
2.     rate, data <- Helper.IOClass.openWave(path)
3.     data <- np.array(data,dtype =np.int16)
4.     data2, b3_msg <- rde.RDE.DecodeRDE(data)
5.     RETURN data2, b3_msg
6. ENDFUNCTION
```

Pseudocode 4.6 Fungsi *rever_smple*

Fungsi *irrever_smple* digunakan untuk ekstraksi pesan dari *stego-audio*. Masukan dari fungsi ini berupa *path* lokasi file *stego-audio,* sedangkan keluaran berupa ekstraksi pesan dalam bentuk *base3*.

Input	Path lokasi file audio
Output	Pesan *base3*

```
1. FUNCTION irrever_smple(path):
2.     rate, data <- Helper.IOClass.openWave(path)
3.     data <- np.array(data,dtype =np.int16)
4.     b3_msg <- rde.RDE.DecNoLMRDE(data)
5.     RETURN b3_msg
6. ENDFUNCTION
```

Pseudocode 4.7 Fungsi *irrever_smple*

4.2.1.3 Implementasi Kelas *Report*

Pada subkelas ini menangani fungsi yang berkaitan dengan hasil perhitungan setelah proses penerapan metode steganografi diantaranya adalah perhitungan *MSE*, *PSNR*, *correlation* dan *memory usage*. Semakin besar nilai *PSNR*, semakin baik kualitas audio yang dihasilkan. Sedangkan *correlation* menandakan tingkat kemiripan antara dua file yang berkisar antara 1-0. Semakin mendekati 1, semakin tinggi tingkat kemiripan dari kedua file. Proses implementasinya dapat dilihat pada Pseudocode 4.8, Pseudocode 4.9, Pseudocode 4.10, Pseudocode 4.11 dan Pseudocode 4.12. Fungsi *MSE* digunakan untuk menghitung nilai *MSE* antara dua *sample-audio (cover-audio* dan *stego-audio).* Masukan dari fungsi ini berupa *path* lokasi kedua file audio *(cover-audio* dan *stego-audio)* sedangkan keluaran berupa nilai *MSE* dan *correlation.*

Input	Path lokasi file *cover-audio* dan file *stego-audio*
Output	*MSE* dan *Correlation*

```
1. FUNCTION MSE(path,path2):
2.     rate, data <- Helper.IOClass.openWave(path)
3.     data <- np.array(data,dtype =np.int16)
4.     rate2, data2 <- Helper.IOClass.openWave(path2)
5.     data2 <- np.array(data2,dtype =np.int16)
6.     mse <- np.sqrt(mean_squared_error(data, data2))
7.     corr <- Helper.Report.Correlation(data,data2)
8.     RETURN mse,corr
9. ENDFUNCTION
```

Pseudocode 4.8 Fungsi *MSE*

Fungsi *PSNR* digunakan untuk menghitung nilai *PSNR* antara dua *sample-audio (cover-audio* dan *stego-audio).* Masukan dari fungsi ini berupa nilai *MSE*, sedangkan keluaran berupa nilai *PSNR*.

Input	*MSE*
Output	*PSNR*

```
1. FUNCTION PSNR(MSE):
2.     psnr <- 10 * np.math.log((65535*65535)/MSE, 10)
3.     RETURN psnr
4. ENDFUNCTION
```

Pseudocode 4.9 Fungsi *PSNR*

Fungsi *Correlation* digunakan untuk menghitung nilai *correlation* antara dua *sample-audio (cover-audio* dan *stego-audio)* ataupun antara dua file teks (pesan dan elstraksi pesan). Masukan dari fungsi ini berupa dua buah file, sedangkan keluaran berupa nilai *correlation*.

Input	Path lokasi file 1 dan path lokasi file 2
Output	*Correlation*

```
1. FUNCTION Correlation(file, file2):
2.     Corr <- np.corrcoef(file, file2)[0, 1]
3.     RETURN Corr
4. ENDFUNCTION
```

Pseudocode 4.10 Fungsi *Correlation*

Fungsi *consoleprint* digunakan untuk menampilkan atau mencetak *log* hasil dari proses-proses yang telah berjalan. Masukan dari fungsi ini berupa *LogWidget* dan teks pesan yang akan ditampilkan, sedangkan keluaran berupa teks pesan.

Input	*LogWidget* dan teks pesan
Output	Teks pesan

```
1. FUNCTION consoleprint(logOutput,text):
2.     logOutput.moveCursor(QtGui.QTextCursor.End)
3.     logOutput.insertPlainText(text)
4.     sb <- logOutput.verticalScrollBar()
5.     sb.setValue(sb.maximum())
6. ENDFUNCTION
```

Pseudocode 4.11 Fungsi *consoleprint*

Fungsi *memory_usage* digunakan untuk menghitung total pemakaian memori yang digunakan pada saat suatu proses berjalan. Tidak ada masukan dari fungsi ini, sedangkan keluaran berupa total pemakaian memori.

Input	-
Output	Pemakaian memori

```
1. FUNCTION memory_usage_psutil():
2.     # RETURN the memory usage in MB
3.     process <- psutil.Process(os.getpid())
4.     RETURN mem
5. ENDFUNCTION
```

Pseudocode 4.12 Fungsi *memory_usage*

4.2.2 Kelas *BASE3*

Kelas ini menangani *preprocessing payload* yang akan disisipkan dan ekstraksi *payload* setelah proses penyisipan. *Payload* yang terdapat pada file berekstensi .txt akan diproses terlebih dahulu sebelum proses penyisipan *payload*. *Payload* nantinya akan diubah kedalam ASCII terslebih dahulu untuk diambil nilai dari sebuah karakter melalui fungsi *toassci*. Setelah mengetahui nilai ASCII dari suatu karakter barulah *payload* diubah kedalam bentuk basis 3 menggunakan fungsi *tobase3*. *Payload* base3 ini kemudian disimpan dalam *array* dan siap untuk dilakukan proses penyisipan. Sedangkan pada proses ekstraksi *payload*, fungsi *revbase3* digunakan untuk pengambilan pesan dari

stego audio. Didalam fungsi *revbase3* terdapat pemanggilan fungsi *tochar* yang digunakan untuk pengembalian pesan dari base3 ke karakter. Proses implementasinya dapat dilihat pada Pseudocode 4.13.

4.2.2.1 Implementasi Fungsi *toascii*

Fungsi *toascii* digunakan untuk mengambil nilai *ASCII* dari sebuah karakter. Didalam fungsi ini terdapat pemanggilan fungsi *tobase3* yang digunakan untuk mengubah nilai *ASCII* kedalam bentuk *base3*. Masukan dari fungsi ini berupa teks pesan, sedangkan keluaran berupa pesan dalam bentuk *base3*.

Input	Teks pesan
Output	Teks pesan *base3*

```
1.  FUNCTION toascii(message):
2.      asci <- [ord(c) for c in message]
3.      b <- []
4.      for x in asci:
5.          b.append(BASE3.tobase3(x))
6.      ENDFOR
7.      data <- []
8.      for x in b:
9.          for y in range(len(x)):
10.             data.append(x[y])
11.         ENDFOR
12.     ENDFOR
13.     RETURN data
14. ENDFUNCTION
```

Pseudocode 4.13 Fungsi *toascii*

4.2.2.2 Implementasi Fungsi *tobase3*

Fungsi *tobase* digunakan untuk mengubah nilai *ASCII* kedalam bentuk *base3*. Masukan dari fungsi ini berupa pesan *ASCII*, sedangkan keluaran berupa pesan dalam bentuk *base3*. Proses implementasinya dapat dilihat pada Pseudocode 4.14.

Input	Pesan *ASCII*
Output	Pesan *base3*

```
1.  FUNCTION tobase3(message):
2.      a <- []
3.      while message>=1:
4.          sisa <- message % 3
5.          message /= 3
6.          a.append(sisa)
7.      ENDWHILE
8.      while len(a) < 5:
9.          a.append(0)
10.     ENDWHILE
11.     a.reverse()
12.     RETURN a
13. ENDFUNCTION
```

Pseudocode 4.14 Fungsi *tobase3*

4.2.2.3 Implementasi Fungsi *tochar*

Fungsi *tochar* digunakan untuk mengembalikan pesan *base3* kedalam nilai *ASCII*. Masukan dari fungsi ini berupa pesan *base3*, sedangkan keluaran berupa pesan dalam bentuk karakter. Proses implementasinya dapat dilihat pada Pseudocode 4.15.

Input	Pesan *base3*
Output	Karakter

```
1. FUNCTION tochar(message):
2.     char <- [chr(c) for c in message]
3.     RETURN char
4. ENDFUNCTION
```

Pseudocode 4.15 Fungsi *tochar*

4.2.2.4 Implementasi Fungsi *revbase3*

Fungsi *revbase3* digunakan untuk mengembalikan pesan *base3* kedalam nilai *ASCII*. Didalam fungsi ini terdapat pemanggilan fungsi *tochar* yang digunakan untuk mengubah pesan

base3 kedalam nilai *ASCII*. Masukan dari fungsi ini berupa teks pesan *base3*, sedangkan keluaran berupa pesan dalam bentuk *ASCII*. Proses implementasinya dapat dilihat pada Pseudocode 4.16.

Input	Pesan *base3*
Output	Pesan *ASCII*

```
1.  FUNCTION revbase3(message):
2.      a <- []
3.      itr=4
4.      data=0
5.      for x in range(len(message)):
6.          temp=pow(3,itr)*int(message[x])
7.          itr-=1
8.          data+=temp
9.          IF itr==-1:
10.             a.append(data)
11.             temp=0
12.             itr=4
13.             data=0
14.         ENDIF
15.     ENDFOR
16.     decode=a
17.     char=BASE3.tochar(decode)
18.     out=[]
19.     buff=[]
20.     for x in char:
21.         buff.append(x)
22.     ELSE:
23.         IF buff:
24.             out.append(''.join(buff))
25.         ENDIF
26.     ENDFOR
27.     RETURN out
28. ENDFUNCTION
```

Pseudocode 4.16 Fungsi *revbase3*

4.2.3 Kelas *RDE*

Kelas ini menangani fungsi yang berkaitan dengan penerapan metode *RDE*. Terdapat 4 fungsi utama diantaranya: fungsi *NoLMRDE, DecNoLMRDE, RDE* dan *DecodeRDE*. Proses implementasinya dapat dilihat pada sub-bab 4.2.2.1, 4.2.2.2, 4.2.2.3 dan 0

4.2.3.1 Implementasi Fungsi *NoLMRDE*

Fungsi *NoLMRDE* digunakan untuk proses penyisipan pesan tanpa menggunakan *location map*. Didalam fungsi ini juga terdapat pemanggilan fungsi *NoLMSVM* yang digunakan untuk proses penyisipan pesan setelah metode *RDE* diterapkan. Masukan dari fungsi ini berupa *cover-sample* dan pesan *base3*, sedangkan keluaran berupa *stego-sample*. Proses implementasinya dapat dilihat pada Pseudocode 4.17.

Input	*Sample-audio* dan pesan *base3*
Output	*Stego-sample*

```
1.  FUNCTION NoLMRDE(sample,b3_msg):
2.      a=0
3.      i=0
4.      msg=len(b3_msg)
5.      stop=str(msg)+" "
6.      stop=base3.BASE3.toascii(stop)
7.      while(True):
8.          sample[i]=svm.SVM.NoLMSVM(sampl[i],stop[i]
9.          i+=1
10.         IF(i==len(stop)):break
11.         ENDIF
12.     ENDWHILE
13.     while(True):
14.         # calculate h
15.         h <- RDE.cal_h(sample[i],sample[i+1])
16.         # calculate I
17.         I <- RDE.int_average(sample[i],sample[i+1])
18.         # calculate h'
```

```
19.         h1 <- RDE.cal_h1(h)
20.         # calculate h"
21.         h11 <- RDE.difference2(h1,b3_msg[a])
22.                     ENDIF
23.         y=sample[i+1]%2
24.         new_x <- RDE.new_smp_x(h11,I,b3_msg[a],y)
25.         new_y <- RDE.new_smp_y(h11,I,b3_msg[a],y)
26.         sample[i] <- new_x;
27.         sample[i+1] <- new_y;
28.         IF(a < msg-1):
29.             a+=1
30.             sample[i+2]=svm.SVM.NoLMSVM(sample[i+2]
    ,b3_msg[a])
31.         ENDIF
32.         i+=3;a+=1
33.         IF(a==msg):break
34.         ENDIF
35.     ENDWHILE
36.     RETURN sample
37. ENDFUNCTION
```

Pseudocode 4.17 Fungsi *NoLMRDE*

Selain fungsi *NoLMSVM* ada beberapa fungsi lain yang dipanggil dalam fungsi *NoLMRDE* diantaranya adalah fungsi *cal_h, int_average, cal_h1, difference2, new_smp_x* dan *new_smp_y*. Beberapa fungsi tersebut digunakan dalam proses penyisipan pesan ke dalam *sample* untuk mendapatkan nilai *sample* baru. *Pseudocode* dari beberapa fungsi diatas dapat dilihat pada Pseudocode 4.18, Pseudocode 4.19, Pseudocode 4.20, Pseudocode 4.21, Pseudocode 4.22 dan Pseudocode 4.23

Input	Nilai sepasang *sample (sample-x, sample-y)*
Output	Nilai selisih *sample (h′)*

```
1. FUNCTION cal_h(sample_x, sample_y):
2.     RETURN (sample_x - sample_y)
3. ENDFUNCTION
```

Pseudocode 4.18 Fungsi *cal_h*

Input	Nilai h
Output	Nilai h'

```
1.  FUNCTION cal_h1(h):
2.      RETURN math.floor(h/2)
3.  ENDFUNCTION
```

Pseudocode 4.19 Fungsi *h'*

Input	Nilai h' dan *digit payload (b)*
Output	Nilai h"

```
1.  FUNCTION difference2(h1, b):
2.      RETURN 2*h1 + b
3.  ENDFUNCTION
```

Pseudocode 4.20 Fungsi *difference2*

Input	Nilai $h", l, b$ dan *sample y*
Output	Nilai *sample x* baru

```
1. FUNCTION new_smp_x(h11,int_avg,b,y):
2.     IF(b==0):RETURN int_avg+math.floor((h11+b)/2)
3.     ELSE:
4.         IF(y==0):RETURN int_avg +math.floor(h11/2)+1
5.         ELSE:RETURN int_avg + math.floor((h11)/2)
6.         ENDIF
7.     ENDIF
8. ENDFUNCTION
```

Pseudocode 4.21 Fungsi *new_smp_x*

Input	Nilai $h", l, b$ dan *sample y*
Output	Nilai *sample y* baru

```
1. FUNCTION new_smp_y(h11,int_avg,b,y):
2.     IF(b==0):RETURN int_avg - math.floor(h11/2)
3.     ELSE:
4.         IF(y==0):RETURN _avg - math.floor(h11/2)
5.         ELSE:RETURN int_avg-math.floor(h11/2)+1
6.         ENDIF
7.     ENDIF
8. ENDFUNCTION
```

Pseudocode 4.22 Fungsi *new_smp_y*

Input	Nilai sepasang *sample (sample-x, sample-y)*
Output	Nilai l

```
1. FUNCTION int_average(sample_x,sample_y):
2.     int_avg <- float(sample_x)/2 + float(sample_y)/2
3.     IF(int_avg < 0):
4.         int_avg <- math.floor(int_avg)
5.     ELSE:
6.         int_avg <- math.floor(int_avg)
7.     ENDIF
8.     RETURN int_avg
9. ENDFUNCTION
```

Pseudocode 4.23 Fungsi *int_average*

4.2.3.2 Implementasi Fungsi *RDE*

Fungsi *RDE* digunakan untuk proses penyisipan pesan dengan menggunakan *location map*. Didalam fungsi ini juga terdapat pemanggilan fungsi *SVM* yang digunakan untuk proses penyisipan pesan setelah metode *RDE* diterapkan. Masukan dari fungsi ini berupa *cover-sample* dan pesan *base3*, sedangkan keluaran berupa *stego-sample*. Proses implementasinya dapat dilihat pada Pseudocode 4.24.

Input	*Cover-sample* dan pesan *base3*
Output	*Stego-sample* dan *location map*

```
1.  FUNCTION RDE(sample,b3_msg):
2.      a=0
3.      i=0
4.      msg=len(b3_msg)
5.      LM <-""
6.      while(True):
7.          # calculate h
8.          h <- RDE.cal_h(sample[i],sample[i+1])
9.          IF(h%2!=0):
10.             IF(b3_msg[a]==2):LM+=str(3)
11.             ELSE:LM+=str(1)
12.             ENDIF
13.         ELSEIF(b3_msg[a]==2):LM+=str(2)
14.         ELSEIF(h%2==0):LM+=str(0)
15.         ELSE :LM+=str(2)
16.         ENDIF
17.         # calculate I
18.         I <- RDE.int_average(sample[i],sample[i+1])
19.         # calculate h'
20.         h1 <- RDE.cal_h1(h)
21.         # calculate h"
22.         h11 <- RDE.difference2(h1,b3_msg[a])
23.
24.         new_x <- RDE.new_sample_x(h11,I,b3_msg[a])
25.         new_y <- RDE.new_sample_y(h11,I,b3_msg[a])
26.         sample[i] <- new_x;
27.         sample[i+1] <- new_y;
28.         IF(a < msg-1):
29.             a+=1
30.             sample[i], sample[i+1], lmSVM <- svm.SVM
      .SVM(sample[i], sample[i+1], int(b3_msg[a]))
31.             IF(lmSVM<3): lmSVM=0
32.             ELSEIF(lmSVM<6): lmSVM=1
33.             ELSE: lmSVM=2
34.             ENDIF
35.             LM+=str(lmSVM)
36.         ENDIF
37.         i+=2;a+=1
38.         IF(a==msg): break
39.         ENDIF
```

```
    ENDWHILE
    RETURN sample, LM
ENDFUNCTION
```

Pseudocode 4.24 Fungsi *RDE*

Selain fungsi *SVM* ada beberapa fungsi lain yang juga dipanggil sama seperti dalam fungsi *NoLMRDE* diantaranya adalah fungsi *cal_h*, *int_average*, *cal_h1* dan *difference2*. Namun untuk mendapatkan nila baru dari sepasang *sample*, dilakukan pemanggilan fungsi *new_samlpe_x* dan *new_sample_y*. *Pseudocode* dari kedua fungsi tersebut dapat dilihat pada Pseudocode 4.25 dan Pseudocode 4.26.

Input	Nilai $h"$, l dan b
Output	Nilai *sample x* baru

```
FUNCTION new_sample_x(h11,int_avg,b):
    IF(b==2): RETURN int_avg + math.floor(h11/2)-1
    ELSE: RETURN int_avg + math.floor((h11+b)/2)
    ENDIF
ENDFUNCTION
```

Pseudocode 4.25 Fungsi *new_sample_x*

Input	Nilai $h"$, l dan b
Output	Nilai *sample y* baru

```
FUNCTION new_sample_y(h11,int_avg,b):
    IF(b==2): RETURN int_avg -math.floor(h11/2)+1
    ELSE: RETURN int_avg - math.floor(h11/2)
    ENDIF
ENDFUNCTION
```

Pseudocode 4.26 Fungsi *new_sample_y*

4.2.3.3 Implementasi Fungsi *DecNoLMRDE*

Fungsi *DecNoLMRDE* digunakan untuk proses ekstraksi pesan tanpa menggunakan *location map*. Didalam fungsi ini juga terdapat pemanggilan fungsi *DecNoLMSVM* yang digunakan untuk

proses ekstraksi setelah proses ekstraksi pesan menggunakan metode *RDE* dilakukan. Masukan dari fungsi ini berupa *stego-sample* dan *path* lokasi file *stego-audio*, sedangkan keluaran berupa *recovery-sample*. Proses implementasinya dapat dilihat pada Pseudocode 4.27.

Input	*Stego-sample*
Output	Ekstraksi pesan

```
1.  FUNCTION DecNoLMRDE(sample):
2.      msg <- []
3.      a=0
4.      i=0
5.      while(True):
6.          stop_parram=svm.SVM.DecNoLMSVM(sample[i])
7.          msg.append(stop_parram)
8.          IF(i%5==4):
9.              check=base3.BASE3.revbase3(msg[i-4:])
10.             IF(check[0]==" "):break
11.         ENDIF
12.             ENDIF
13.         i+=1
14.     ENDWHILE
15.     s=i
16.     stop <- (base3.BASE3.revbase3(msg[0:i-4]))
17.     stop <- int(stop[0])
18.     i+=1
19.     while(stop > 0):
20.         h11 <- RDE.recovry_h11(sampl[i],sampl[i+1])
21.         IF(h11%2==0):msg.append(0)
22.         ELSEIF(sample[i+1]%2==1):msg.append(2)
23.         ELSEIF(sample[i+1]%2==0):msg.append(1)
24.         ENDIF
25.         IF(stop!=1):msg.append(svm.SVM.DecNoLMSVM(s
      ample[i+2]))
26.         ENDIF
27.         i+=3;stop-=2
28.     ENDWHILE
29.     RETURN msg[s+1:]
30. ENDFUNCTION
```

Pseudocode 4.27 Fungsi *DecNoLMRDE*

Selain fungsi *DecNoLMSVM* ada fungsi lain yang dipanggil dalam fungsi *DecNoLMRDE* yaitu fungsi *recovery_h11*. Fungsi tersebut digunakan untuk menentukan nilai dari *digit payload* pada proses ekstraksi pesan. *Pseudocode* dari fungsi *recovery_h11* dapat dilihat pada Pseudocode 4.28.

Input	Nilai sepasang *stego-sample (sample-x,sample-y)*
Output	Nilai *recovery h*"

```
1.  FUNCTION recovery_h11(sample_x,sample_y):
2.      RETURN (int(sample_x - sample_y))
3.  ENDFUNCTION
```

Pseudocode 4.28 Fungsi *recovery_h"*

4.2.3.4 Implementasi Fungsi *DecodeRDE*

Fungsi *DecodeRDE* digunakan untuk proses ekstraksi pesan dan pengembalian nilai *sample (Reversible-sample)* seperti saat semula sebelum proses penyisipan pesan menggunakan *location map*. Didalam fungsi ini juga terdapat pemanggilan fungsi *DecodeSVM* yang digunakan untuk proses ekstraksi dan pengembalian *sample* sebelum proses *recovery-sample* menggunakan metode *RDE* dilakukan. Masukan dari fungsi ini berupa *stego-sample* dan *path* lokasi file *stego-audio*, sedangkan keluaran berupa *recovery-sample* dan ekstraksi pesan. Proses implementasinya dapat dilihat pada Pseudocode 4.29.

Input	*Stego-sample* dan path lokasi file *location map*
Output	*Recovery-sample* dan ekstraksi pesan

```
31. FUNCTION DecodeRDE(sample,path):
32.     msg <- []
33.     lm=hp.Helper.IOClass.openText(path);
34.     a=0
35.     i=0
36.     l_msg= len(lm)
37.     SVM_flag=0
38.     while(True):
```

```
39.         IF(a < l_msg-1):
40.             sample[i],sample[i+1],SVM_msg <- svm.SV
      M.DecodeSVM(sample[i],sample[i+1],lm[a+1])
41.             SVM_flag =1
42.         ENDIF
43.         I <- RDE.recovery_I(sample[i],sample[i+1])
44.         h11 <- RDE.recovery_h11(sample[i],sample[i+
      1])
45.
46.         msg.append(RDE.recovery_msg(h11,int(lm[a]))
      )
47.         IF(SVM_flag):
48.             msg.append(SVM_msg)
49.         ENDIF
50.         recov_sample_x <- RDE.recovery_sample_x(I,h
      11,lm[a])
51.         recov_sample_y <- RDE.recovery_sample_y(I,h
      11)
52.         sample[i] <- recov_sample_x
53.         sample[i+1] <- recov_sample_y
54.         i+=2; a+=2; SVM_flag=0
55.         IF(a>=l_msg):break
56.         ENDIF
57.     ENDWHILE
58.     RETURN sample, msg
59. ENDFUNCTION
```

Pseudocode 4.29 Fungsi *DecodeRDE*

Selain fungsi *DecodeSVM* ada beberapa fungsi lain yang dipanggil dalam fungsi *DecodeRDE* diantaranya adalah fungsi *recovery_I*, *recovery_h11*, *recovery_msg*, *recovery_sample_x* dan *recovery_sample_y*. Namun untuk mendapatkan nilai baru dari sepasang *sample*, dilakukan pemanggilan fungsi *new_samlpe_x* dan *new_sample_y*. *Pseudocode* dari beberapa fungsi tersebut dapat dilihat pada Pseudocode 4.30, Pseudocode 4.31, Pseudocode 4.32, Pseudocode 4.33 dan Pseudocode 4.34.

Input	Nilai sepasang *stego-sample (sample-x,sample-y)*
Output	Nilai *recovery l*

```
1. FUNCTION recovery_I(sample_x,sample_y):
2.     RETURN (math.floor(float((sample_x))/2 + float((
      sample_y))/2))
3. ENDFUNCTION
```

Pseudocode 4.30 Fungsi *recovery_I*

Input	Nilai sepasang *stego-sample (sample-x,sample-y)*
Output	Nilai *recovery h*"

```
4.  FUNCTION recovery_h11(sample_x,sample_y):
5.      RETURN (int(sample_x - sample_y))
6.  ENDFUNCTION
```

Pseudocode 4.31 Fungsi *recovery_h"*

Input	Nilai *h*" dan *location map*
Output	Ekstraksi pesan

```
1.  FUNCTION recovery_msg(h11,LM):
2.      IF(LM==3):
3.          msg =2
4.      ELSEIF(LM==2):
5.          msg =2
6.      ELSEIF(h11 % 2 = 0):
7.          msg <- 0
8.      ELSE: msg <- 1
9.      ENDIF
10.     RETURN msg
11. ENDFUNCTION
```

Pseudocode 4.32 Fungsi *recovery_msg*

Input	Nilai *l, h"* dan *location map*
Output	*Recovery-sample-x*

```
1. FUNCTION recovery_sample_x(I,h11,LM):
2.     LM=int(LM)
3.     IF(LM%2): RETURN int(I + math.floor(h11/2) + 1)
4.     ELSE: RETURN int(I + math.floor(h11/2))
5.     ENDIF
6. ENDFUNCTION
```

Pseudocode 4.33 Fungsi *recovery_sample_x*

Input	Nilai *l* dan *h"*
Output	*Recovery-sample-y*

```
1. FUNCTION recovery_sample_y(I,h11):
2.     RETURN int(I - float(math.floor(h11/2)))
```

Pseudocode 4.34 Fungsi *recovery_sample_y*

4.2.4 Kelas SVM

Kelas ini menangani fungsi yang berkaitan dengan penerapan metode *SVM*. Terdapat 4 fungsi utama diantaranya: fungsi *NoLMSVM*, *DecNoLMSVM*, *SVM* dan *DecodeSVM*. Proses implementasinya dapat dilihat pada *sub-bab* 4.2.4.1, 4.2.4.2, 4.2.4.3 dan 4.2.4.4

4.2.4.1 Implementasi Fungsi *NoLMSVM*

Fungsi *NoLMSVM* digunakan untuk proses penyisipan pesan tanpa menggunakan *location map.* Masukan dari fungsi ini berupa *cover-sample* dan pesan dalam bentuk *base3*, sedangkan keluaran berupa *stego-sample.* Proses implementasinya dapat dilihat pada Pseudocode 4.35.

Input	*Cover-sample* dan pesan *base3*
Output	*Stego-sample*

```
1.  FUNCTION NoLMSVM(sample,b3_msg):
2.      s_mod=sample % 3
3.      IF(s_mod==b3_msg):
4.          sample=sample
5.      ELSEIF(s_mod-b3_msg==-2):
6.          sample=sample-1
7.      ELSEIF(s_mod<b3_msg):
8.          sample=sample+1
9.      ELSEIF(s_mod-b3_msg==2):
10.         sample=sample+1
11.     ELSEIF(s_mod>b3_msg):
12.         sample=sample-1
13.     ENDIF
14.     RETURN sample
15. ENDFUNCTION
```

Pseudocode 4.35 Fungsi *NoLMSVM*

4.2.4.2 Implementasi Fungsi *DecLMSVM*

Fungsi *DecLMSVM* digunakan untuk proses ekstraksi pesan tanpa menggunakan *location map*. Masukan dari fungsi ini berupa *stego-sample*, sedangkan keluaran berupa hasil ekstraksi *digit payload*. Proses implementasinya dapat dilihat pada Pseudocode 4.36.

Input	*Stego-sample*
Output	Ekstraksi pesan

```
1. FUNCTION DecNoLMSVM(sample):
2.     recov_msg <- sample % 3
3.     RETURN recov_msg
4. ENDFUNCTION
```

Pseudocode 4.36 Fungsi *DecLMSVM*

4.2.4.3 Implementasi Fungsi *SVM*

Fungsi *SVM* digunakan untuk proses penyisipan pesan dengan menggunakan *location map*. Masukan dari fungsi ini berupa sepasang *cover-sample (x, y)* dan pesan dalam bentuk *base3*, sedangkan keluaran berupa sepasang *stego-sample (x,y)* dan katagori blok pesan. Proses implementasinya dapat dilihat pada Pseudocode 4.37.

Input	*Cover-sample x* & *y* dan pesan *base3*
Output	*Stego-sample x* & *y* dan blok pesan

```
1.  FUNCTION SVM(sample_x,sample_y,msg):
2.      num_block <- SVM.BlockMsgSVM(sample_x,sample_y)
3.      IF(num_block <3):
4.          IF(msg==0):
5.              RETURN sample_x,sample_y,num_block
6.          ELSEIF(msg==1):
7.              RETURN sample_x+1,sample_y+1,num_block
8.          ELSEIF(msg==2):
9.              RETURN sample_x-1,sampl_y-1,num_block
10.         ENDIF
11.         OUTPUT "msg block 0"
12.     ELSEIF(num_block <6):
13.         IF(msg==1):
14.             RETURN sample_x,sample_y,num_block
15.         ELSEIF(num_block==3):
16.             IF(msg==0):
17.                 RETURN sample_x,sampl_y+1,num_bloCk
18.             ELSEIF(msg==2):
19.                 RETURN sampl_x+1,sampl_y+1,num_blok
20.             ENDIF
21.         ELSEIF(num_block==4):
22.             IF(msg==0):
23.                 RETURN sample_x+1,sample_y,num_blok
24.             ELSEIF(msg==2):
25.                 RETURN sampl_x+1,sampl_y+1,num_blok
26.             ENDIF
27.         ELSEIF(num_block==5):
28.             IF(msg==0):
```

```
29.                 RETURN sample_x-1,sample_y-
    1,num_block
30.             ELSEIF(msg==2):
31.                 RETURN sampl_x+1,sampl_y+1,num_blok
32.             ENDIF
33.         ENDIF
34.         OUTPUT "msg block 1"
35.     ELSEIF(num_block <9):
36.         IF(msg==2):
37.             RETURN sample_x,sample_y,num_block
38.         ELSEIF(num_block==6):
39.             IF(msg==0):
40.                 RETURN sample_x-
    1,sample_y,num_block
41.             ELSEIF(msg==1):
42.                 RETURN sample_x,sampl_y+1,num_block
43.             ENDIF
44.         ELSEIF(num_block==7):
45.             IF(msg==0):
46.                 RETURN sample_x,sample_y-
    1,num_block
47.             ELSEIF(msg==1):
48.                 RETURN sample_x+1,sampl_y,num_block
49.             ENDIF
50.         ELSEIF(num_block==8):
51.             IF(msg==0):
52.                 RETURN sampl_x+1,sampl_y+1,num_blok
53.             ELSEIF(msg==1):
54.                 RETURN sample_x-1,sample_y-
    1,num_block
55.             ENDIF
56.         ENDIF
57.         OUTPUT "msg block 2"
58.     ELSE: "block not registered"
59.     ENDIF
60. ENDFUNCTION
```

Pseudocode 4.37 Fungsi *SVM*

Fungsi lain yang dipanggil dalam fungsi *SVM* adalah fungsi *BlockMsgSVM*. Fungsi tersebut digunakan dalam proses penentuan blok pesan, termasuk dalam katagori manakah sepasang

sample yang akan melalui tahap penyisipan pesan. Berikut merupakan *Pseudocode* dari fungsi *BlockMsgSVM*. Proses implementasinya dapat dilihat pada Pseudocode 4.38.

Input	*Cover-sample x & y*
Output	*Message-block*

```
1.  FUNCTION BlockMsgSVM(sample_x,sample_y):
2.      a=[]
3.      msgBlc=[]
4.      a.append(sample_x % 3)
5.      a.append(sample_y % 3)
6.      msgBlc.append(a)
7.      i=0
8.      BlcMsg <- [[0,0],[0,2],[2,0], [0,1],[1,0],[1,1]
      , [1,2],[2,1],[2,2]]
9.      while(True):
10.         IF(msgBlc[0]==BlcMsg[i]):
11.             break
12.         ENDIF
13.         i+=1
14.     ENDWHILE
15.     RETURN i
16. ENDFUNCTION
```

Pseudocode 4.38 Fungsi *BlockMsgSVM*

4.2.4.4 Implementasi Fungsi *DecodeSVM*

Fungsi *DecodeSVM* digunakan untuk proses ekstraksi pesan dengan menggunakan *location map*. Masukan dari fungsi ini berupa sepasang *stego-sample(x,y)* dan *location map*, sedangkan keluaran berupa sepasang *recovery-sample(x,y)* dan hasil ekstraksi *digit payload*. Proses implementasinya dapat dilihat pada Pseudocode 4.39.

.

Input	*Stego-sample x & y* dan *location map*
Output	*recovery -sample x & y* dan ekstraksi pesan

```
1.  FUNCTION DecodeSVM(sample_x,sample_y,LM_msg):
2.      Block=SVM.BlockMsgSVM(sample_x,sample_y)
```

```
3.      IF(Block<3):real_block <- 0
4.      ELSEIF(Block<6):real_block <- 1
5.      else:real_block <- 2
6.      ENDIF
7.      IF(int(LM_msg)==real_block):
8.          RETURN sample_x,sample_y,real_block
9.      ELSE:
10.         num_block <- SVM.RevBlockMsgSVM(sample_x,sa
      mple_y,LM_msg)
11.         IF(num_block<3):
12.             RETURN sample_x-1,sample_y-1, 1
13.         ELSEIF(num_block<6):
14.             RETURN sample_x+1,sample_y+1, 2
15.         ELSEIF(num_block==6):
16.             RETURN sample_x,sample_y-1, 0
17.         ELSEIF(num_block==7):
18.             RETURN sample_x-1,sample_y, 0
19.         ELSEIF(num_block==8):
20.             RETURN sample_x+1,sample_y+1, 0
21.         ELSEIF(num_block<12):
22.             RETURN sample_x-1,sample_y-1, 2
23.         ELSEIF(num_block==12):
24.             RETURN sample_x+1,sample_y, 0
25.         ELSEIF(num_block==13):
26.             RETURN sample_x,sample_y+1, 0
27.         ELSEIF(num_block==14):
28.             RETURN sample_x-1,sample_y-1, 0
29.         ELSEIF(num_block==15):
30.             RETURN sample_x,sample_y-1, 1
31.         ELSEIF(num_block==16):
32.             RETURN sample_x-1,sample_y, 1
33.         ELSEIF(num_block==17):
34.             RETURN sample_x+1,sample_y+1, 1
```

Pseudocode 4.39 Fungsi *DecodeSVM*

Fungsi lain yang dipanggil dalam fungsi *SVM* adalah fungsi *BlockMsgSVM* dan *RevBlockMsgSVM*. Sama seperti sebelumnya fungsi *BlockMsgSVM* digunakan untuk proses penentuan blok pesan, sedangkan fungsi *RevBlockMsgSVM* digunakan dalam proses pengembalian nilai *sample (Reversible-sample)* dan ekstraksi pesan. Berikut merupakan *Pseudocode* dari

fungsi *RevBlockMsgSVM*. Proses implementasinya dapat dilihat pada Pseudocode 4.40.

Input	*Stego-sample x & y*
Output	*Message-block*

```
1.  FUNCTION RevBlockMsgSVM(sample_x,sample_y,LM_msg):
2.      a=[]
3.      msgBlc=[]
4.      a.append(sample_x % 3)
5.      a.append(sample_y % 3)
6.      msgBlc.append(a)
7.      i=0
8.      IF(int(LM_msg)==0):i=0
9.      ELSEIF(int(LM_msg)==1):i=6
10.     ELSEIF(int(LM_msg)==2):i=9
11.     ENDIF
12.     BlcMsg <- [[1,1],[1,0],[0,1], [2,2],[2,1],[1,2]
      , [0,2],[2,0],[0,0], [1,2],[2,1],[2,2], [0,2],[2,
      0],[0,0], [1,0],[0,1],[1,1]]
13.     while(True):
14.         IF(msgBlc[0]==BlcMsg[i]):
15.             break
16.         ENDIF
17.         i+=1
18.     ENDWHILE
19.     RETURN i
20. ENDFUNCTION
```

Pseudocode 4.40 Fungsi *RevBlockMsgSVM*

BAB V
HASIL UJI COBA DAN EVALUASI

Bab ini berisi penjelasan mengenai skenario uji coba dan evaluasi dari metode yang diusulkan.

5.1 Lingkungan Pengujian

Lingkungan pengujian pada uji coba steganografi menggunakan metode *RDE* dan *SVM* akan digunakan lingkungan pengujian dengan spesifikasi perangkat keras dan perangkat lunak yang ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Pengujian

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras	Prosesor	Intel(R) Core(TM) i7-4700HQ CPU @ 2.4 GHz
	Memori	8 GB 1600 MHz DDR3
Perangkat Lunak	Sistem Operasi	Windows 10
	Perangkat Pengembang	PyCharm Edu 3.0.1 2016.2.3

5.2 Dataset Pengujian

Sub bab ini mejelaskan mengenai data yang digunakan untuk uji coba Adapun data pengujian dalam Tugas Akhir ini adalah sebagai berikut:

5.2.1 *Cover Audio (.wav)*

Cover Audio merupakan data set dari data masukan yang digunakan dalam proses penyisipan pesan. Data *cover* audio terdiri

dari empat buah file audio dengan tipe atau *genre* musik yang berbeda. Hanya genre musik *POP* yang akan dipotong sehingga memiliki 5 buah durasi audio yang berbeda (dua detik, empat detik, enam detik, delapan detik dan sepuluh detik). Adapun detail dari empat buah audio tersebut ditunjukan pada Tabel 5.2.

Tabel 5.2 Detail Data Audio

No	Nama *Cover Audio*	Durasi Audio (detik)	Kapasitas Pesan *(Dengan LM) (digit)*	Kapasitas Pesan *(Tanpa LM) (digit)*	Ukuran Audio (KB)
1	*Pop.wav*	2	88200	58770	172,27
		4	176400	117570	344,53
		6	264600	176370	516,80
		8	352800	235170	689,06
		10	441000	293970	861,33
2	*Metal.wav*	10	441000	293970	861,33
3	*Electro.wav*	10	441000	293970	861,33
4	*Jazz.wav*	10	441000	293970	861,33

5.2.2 *Payload Data (.txt)*

Payload Data merupakan data masukan berupa pesan yang akan disisipkan pada saat proses penyisipan menggunakan metode *RDE* dan *SVM*. *Payload* diambil dari *www.lipsum.com* dengan menghasilkan data teks secara acak. *Payload* yang digunakan sebagai uji coba terdiri dari tujuh buah *payload* dengan kapasitas yang berbeda-beda. Untuk lebih jelasnya dapat dilihat pada Tabel 5.3.

Tabel 5.3 Detail Data *Payload*

No	Nama *Payload*	Kapasitas *(kB)*	*Payload Base 3 (digit)*
1	10000 bytes.txt	10	50.480
2	20000 bytes.txt	20	100.650

3	30000 bytes.txt	30	150.820
4	40000 bytes.txt	40	200.990
5	50000 bytes.txt	50	251.110
6	Max(LM).txt	88.200	441.000
7	Max.txt	58.793	293.965

5.3 Skenario Uji Coba

Skenario uji coba diperlukan untuk menguji kebenaran dari metode yang diusulkan. Skenario uji coba juga dapat digunakan dalam proses pengujian performa dari metode yang diusulkan. Sebelum melakukan uji coba, perlu ditentukan skenario yang akan digunakan dalam uji coba. Melalui skenario ini, program akan diuji apakah sudah berjalan dengan benar dan bagaimana performa pada masing-masing skenario dan membandingkan skenario manakah yang memiliki hasil yang lebih baik. Pada uji coba parameter yang diukur adalah nilai *MSE, PSNR, correlation* antara *cover-audio* dan *stego-audio* serta *correlation* antara *payload* dan hasil ekstraksi *payload.*

5.3.1 Skenario Uji Coba Fungsionalitas

Skenario Uji coba fungsionalitas dilakukan untuk menguji kebenaran dari metode yang diusulkan pada bab 4. Uji fungsionalitas meliputi dua buah proses pada steganografi, yaitu pada proses penyisipan dan proses ekstraksi data yang telah disisipkan pada proses penyisipan.

Pada uji coba fungsionalitas digunakan data masukan berupa empat *cover-audio* dengan ukuran yang sama (861 KB) dan *payload* 10000.txt. Uji coba dilakukan pada kedua tipe penyisipan yaitu, penyisipan menggunakan *location map* dan penyisipan tanpa menggunakan *location map.*

Terdapat beberapa parameter yang akan digunakan dalam proses evaluasi uji coba fungsionalitas diantaranya adalah *MSE, PSNR, correlation* antara *cover-audio* dan *stego-audio, correlation*

antara *cover-audio* dan *recovery-audio* pada proses penyisipan dengan *location map* serta *correlation* antara *payload* dan hasil ekstraksi *payload.*

5.3.2 Skenario Uji Coba Performa

Skenario Uji coba performa dilakukan untuk menguji performa dari metode yang diusulkan pada bab 4. Uji coba performa meliputi dua tipe penyisipan dari kedua metode yang diusulkan yaitu, penyisipan menggunakan *location map* dan penyisipan tanpa menggunakan *location map.*

Pada uji coba performa akan dilakukan penyisipan dengan kapasitas maksimal dari audio Pada masing masing genre musik dengan dua tipe penyisipan (dengan dan tanpa *location map*). Terdapat beberapa parameter yang akan digunakan dalam proses evaluasi uji coba performa diantaranya adalah *MSE, PSNR, correlation* antara *cover-audio* dan *stego-audio, correlation* antara *payload* dan hasil ekstraksi *payload,* waktu penyisipan dan ekstraksi *payload* serta total pemakaian memori.

5.3.3 Skenario Uji Coba Parameter *Cover* dan *Payload*

Pada uji coba parameter *cover* dan *payload* digunakan semua dataset yang ada. Data masukan berupa *cover-audio* pada sub-bab 5.2.1 dan *payload* pada sub-bab 5.2.2.

Uji coba dilakukan dengan 2 skenario uji coba. Pertama, penggunaan kapasitas *cover* tetap dengan ukuran *payload* yang berbeda. Kedua, penggunaan kapasitas *payload* tetap dengan ukuran *cover* yang berbeda. Skenario diatas dilakukan pada genre music *POP*. Sehingga total percobaan yang dilakukan terdiri dari empat tabel percobaan (dua tabel uij coba parameter cover dan dua tabel uij coba parameter *payload*). Untuk lebih jelasnya akan dijelaskan lebih lanjut pada subbab 5.3.3.1 dan 5.3.3.2.

Terdapat beberapa parameter yang akan digunakan dalam proses evaluasi uji coba parameter diantaranya adalah *MSE, PSNR,*

correlation antara *cover-audio* dan *stego-audio, correlation* antara *payload* dan hasil ekstraksi *payload,* waktu penyisipan dan ekstraksi *payload* serta total pemakaian memori.

5.3.3.1 Skenario Uji Coba Parameter *Cover*

Pada uji coba parameter *cover* dilakukan penggunaan kapasitas *cover* yang berubah-ubah dengan ukuran *payload* yang tetap. Terdapat lima buah *cover* untuk setiap genre musik dengan ukuran *payload* tetap

Terdapat beberapa parameter yang akan digunakan dalam proses evaluasi uji coba parameter *cover* diantaranya adalah *MSE, PSNR, correlation* antara *cover-audio* dan *stego-audio, correlation* antara *payload* dan hasil ekstraksi *payload,* waktu penyisipan dan ekstraksi *payload* serta total pemakaian memori.

5.3.3.2 Skenario Uji Coba Parameter *Payload*

Pada uji coba parameter *payload* dilakukan penggunaan ukuran *payload* yang berubah-ubah dengan kapasitas *cover* yang tetap. Terdapat lima buah *payload* untuk setiap genre musik dengan kapasitas *cover* tetap

Terdapat beberapa parameter yang akan digunakan dalam proses evaluasi uji coba parameter *payload* diantaranya adalah *MSE, PSNR, correlation* antara *cover-audio* dan *stego-audio, correlation* antara *payload* dan hasil ekstraksi *payload,* waktu penyisipan dan ekstraksi *payload* serta total pemakaian memori.

5.4 Uji Coba

Pada uji coba akan dilakukan uji coba fungsionalitas dan uji coba performa yang telah dijelaskan pada sub-bab 5.3.1 dan 5.3.2.

5.4.1 Uji Coba Fungsionalitas

Pada uji coba fungsionalitas digunakan data masukan berupa empat *cover-audio* (genre musik yang berbeda) dengan ukuran yang sama yaitu 861 KB dan *payload* 10000 bytes.txt. Uji coba dilakukan pada kedua tipe penyisipan yaitu, penyisipan menggunakan *location map* dan penyisipan tanpa menggunakan *location map.*

Hasil uji coba fungsionalitas dapat dilihat pada Tabel 5.4 dan Tabel 5.5.

Tabel 5.4 Hasil Uji Coba Fungsionalitas dengan *Location Map*

Nama *Cover Audio*	*Pop.wav*	*Metal.wav*	*Electro.wav*	*Jazz.wav*
Ukuran Audio (MB)	861,33	861,33	861,33	861,33
Kapasitas Pesan *(digit)*	441000	441000	441000	441000
Nama *Payload*	10000 bytes.txt			
Ukuran *Payload (Byte)*	10096			
Jumlah *Payload Base3 (digit)*	50480			
MSE	0.29762	0.29643	0.29805	0.29747
PSNR (dB)	101.59302	101.61043	101.58675	101.59513
Correlation (cover dan stego)	0.999999999094	0.999999999714	0.999999999440	0.999999995140
Corr. (cover & recov. audio)	1.0	1.0	1.0	1.0
Corr. (payload & ekstraksi payload)	1.0	1.0	1.0	1.0

Tabel 5.5 Hasil Uji Coba Fungsionalitas tanpa *Location Map*

Nama *Cover Audio*	*Pop.wav*	*Metal.wav*	*Electro.wav*	*Jazz.wav*
Ukuran Audio (MB)	861,33	861,33	861,33	861,33
Kapasitas Pesan *(digit)*	441000	441000	441000	441000
Nama *Payload*	10000 bytes.txt			
Ukuran *Payload (Byte)*	10096			
Jumlah *Payload Base3 (digit)*	50480			
MSE	0.31377	0.42842	0.42974	0.43002
PSNR (dB)	101.36354	100.01092	99.99750	99.99475
Correlation (cover dan stego)	0.999999998992	0.999999999403	0.999999998837	0.999999989852
Corr. (cover & recov. audio)	-	-	-	-
Corr. (payload & ekstraksi payload)	1.0	1.0	1.0	1.0

5.4.2 Uji Coba Performa

Pada uji coba performa akan dilakukan penyisipan dengan kapasitas maksimal dari audio pada masing masing genre musik dengan dua tipe penyisipan (dengan dan tanpa *location map*).

Hasil uji coba performa dengan kapasitas maksimal dapat dilihat pada Tabel 5.6, Tabel 5.7, Tabel 5.8 dan Tabel 5.9.

Tabel 5.6 Hasil Uji Coba Performa dengan Kapasitas Maksimal *(POP)*

Nama *Cover Audio*	*Pop.wav*	
Ukuran Audio (KB)	861.33	
Tipe Penyisipan	*Location Map*	Tanpa *LM*
Kapasitas Pesan *(digit)*	441000	293965
Nama *Payload*	Max(LM).txt	Max.txt
Ukuran *Payload (Byte)*	88200	58793
Jumlah *payload Base3 (digit)*	441000	293965
MSE	0.87705	1.14847
PSNR (dB)	96.89935	95.72841
Correlation (cover dan stego)	0.999999992238	0.99999998676
Correlation (payload & ekstraksi payload)	1.0	1.0
Waktu *Penyisipan (detik)*	3.587	1.489
Pemakaian Memori (Penyisipan) (MB)	150.453	145.281
Waktu *Ekstraksi (detik)*	4.280	0.770
Waktu *Recovery Audio (detik)*	4.115	-
Pemakaian Memori (Ekstraksi) (MB)	169.020	159.348

Tabel 5.7 Hasil Uji Coba Performa dengan Kapasitas Maksimal *(METAL)*

Nama *Cover Audio*	Metal.wav	
Ukuran Audio (KB)	861.33	
Tipe Penyisipan	Location Map	Tanpa LM
Kapasitas Pesan *(digit)*	441000	293965
Nama *Payload*	Max(LM).txt	Max.txt
Ukuran *Payload (Byte)*	88200	58793
Jumlah *payload Base3 (digit)*	441000	293965
MSE	0.878666274	1.150437164
PSNR (dB)	96.89135904	95.72096958
Correlation (cover dan stego)	0.999999998	0.99999999577
Correlation (payload & ekstraksi payload)	1.0	1.0
Waktu *Penyisipan (detik)*	3.702000141	1.544000149
Pemakaian Memori (Penyisipan) (MB)	142.5976563	145.8671875
Waktu *Ekstraksi (detik)*	4.410999775	0.765999794
Waktu *Recovery Audio (detik)*	4.246999979	-
Pemakaian Memori (Ekstraksi) (MB)	162.25	156.1796875

Tabel 5.8 Hasil Uji Coba Performa dengan Kapasitas Maksimal *(ELECTRO)*

Nama *Cover Audio*	Electro.wav	
Ukuran Audio (KB)	861.33	
Tipe Penyisipan	Location Map	Tanpa LM
Kapasitas Pesan *(digit)*	441000	293965
Nama *Payload*	Max(LM).txt	Max.txt
Ukuran *Payload (Byte)*	88200	58793
Jumlah *payload Base3 (digit)*	441000	293965
MSE	0.878683049	1.149820059
PSNR (dB)	96.89127613	95.72329981
Correlation (cover dan stego)	0.99999999520	0.9999999918
Correlation (payload & ekstraksi payload)	1.0	1.0
Waktu *Penyisipan (detik)*	3.674999952	1.518000126
Pemakaian Memori (Penyisipan) (MB)	143.2617188	147.5195313
Waktu *Ekstraksi (detik)*	4.373999834	0.762000084
Waktu *Recovery Audio (detik)*	4.205999851	-
Pemakaian Memori (Ekstraksi) (MB)	163.96875	160.5

Tabel 5.9 Hasil Uji Coba Performa dengan Kapasitas Maksimal *(JAZZ)*

Nama *Cover Audio*	Jazz.wav	
Ukuran Audio (KB)	861.33	
Tipe Penyisipan	Location Map	Tanpa LM
Kapasitas Pesan *(digit)*	441000	293965
Nama *Payload*	Max(LM).txt	Max.txt
Ukuran *Payload (Byte)*	88200	58793
Jumlah *payload Base3 (digit)*	441000	293965
MSE	0.877696687	1.147711911
PSNR (dB)	96.89615402	95.73126973
Correlation (cover dan stego)	0.999999958	0.999999929
Correlation (payload & ekstraksi payload)	1.0	1.0
Waktu *Penyisipan (detik)*	3.670000076	1.501999855
Pemakaian Memori (Penyisipan) (MB)	145.1445313	146.6523438
Waktu *Ekstraksi (detik)*	4.292999983	0.75999999
Waktu *Recovery Audio (detik)*	4.128000021	-
Pemakaian Memori (Ekstraksi) (MB)	163.125	160.3945313

5.4.3 Uji Coba Parameter *Cover* dan *Payload*

Pada uji coba parameter *cover* dan *payload* digunakan semua dataset *cover* dan *payload* yang ada. Data masukan berupa *cover-audio* pada sub-bab 5.2.1 dan *payload* pada sub-bab 5.2.2. Untuk lebih jelasnya akan dijelaskan pada sub-bub 5.4.3.1 dan 5.4.3.2

5.4.3.1 Uji Coba Parameter *Cover*

Pada uji coba parameter c*over* akan dilakukan penyisipan dengan beberapa parameter kapasitas *cover* dengan memanfaatkan panjang durasi dari file *cover*. Hasil uji coba parameter *cover* dapat dilihat pada Tabel 5.10 dan Tabel 5.11

Tabel 5.10 Uji Coba Parameter *Cover Location Map (POP)*

Nama Payload (Byte)	1000 bytes.txt				
Ukuran Payload	10096				
Jumlah payload Base3 (digit)	50480				
Durasi Audio (detik)	2	4	6	8	10
Ukuran Audio (KB)	172.265625	344.53125	516.796875	689.0625	861.328125
Kapasitas Pesan (digit)	88200	176400	264600	352800	441000
MSE	0.66549	0.47057	0.38422	0.33275	0.29762
PSNR (dB)	98.09817	99.60332	100.48378	101.10847	101.59302
Correlation (cover dan stego)	0.999999995634	0.999999997757	0.999999998430	0.999999998838	0.999999999094

Correlation(payload & ekstraksi payload)	1.0	1.0	1.0	1.0	1.0
Waktu Penyisipan (detik)	0.400	0.391	0.390	0.382	0.381
Pemakaian Memori (Penyisipan) (MB)	137.390	137.999	137.629	138.637	141.551
Waktu Ekstraksi (detik)	0.967	0.982	0.983	0.988	1.019
Waktu Recovery Audio (detik)	0.474	0.482	0.482	0.484	0.501
Pemakaian Memori (Ekstraksi) (MB)	137.863	138.038	138.118	138.770	140.980

Tabel 5.11 Uji Coba Parameter *Cover Tanpa Location Map (POP)*

Nama Payload (Byte)	1000 bytes.txt				
Ukuran Payload	10096				
Jumlah payload Base3 (digit)	50480				
Durasi Audio (detik)	2	4	6	8	10
Ukuran Audio (KB)	172.265625	344.53125	516.796875	689.0625	861.328125
Kapasitas Pesan (digit)	58770	117570	176370	235170	293970
MSE	0.96082	0.67940	0.55473	0.48041	0.31377

PSNR (dB)	96.50318	98.00833	98.88879	99.51348	101.36354
Correlation (cover dan stego)	0.999999999093 9	0.999999999533 6	0.999999999673 3	0.999999999758 1	0.999999999899 2
Correlation(payload & ekstraksi payload)	-	-	-	-	-
Waktu Penyisipan (detik)	0.227	0.226	0.223	0.224	0.221
Pemakaian Memori (Penyisipan) (MB)	137.709	138.334	137.676	139.023	141.752
Waktu Ekstraksi (detik)	0.128	0.129	0.130	0.130	0.140
Pemakaian Memori (Ekstraksi) (MB)	137.476	137.687	137.519	138.450	140.909

5.4.3.2 Uji Coba Parameter *Payload*

Pada uji coba parameter *payload* akan dilakukan penyisipan dengan beberapa parameter *payload* dengan memanfaatkan ukuran dari *payload*. Hasil uji coba parameter *payload* dapat dilihat pada Tabel 5.12 dan Tabel 5.13

Tabel 5.12 Uji Coba Parameter *Payload Location Map (POP)*

Nama Cover Audio	Pop.wav
Ukuran Audio (KB)	861.33

Kapasitas Pesan (digit)	441000				
Nama Payload (Byte)	1000 bytes.txt	2000 bytes.txt	3000 bytes.txt	4000 bytes.txt	5000 bytes.txt
Ukuran Payload	10096	20130	30164	40198	50222
Jumlah payload Base3 (digit)	50480	100650	150820	200990	251110
MSE	0.29762	0.41997	0.51337	0.59296	0.66258
PSNR (dB)	101.59302	100.09745	99.22533	98.59933	98.11721
Correlation (cover dan stego)	0.999999999094	0.999999999819	0.999999999731	0.999999999642	0.999999999554
Correlation(payload & ekstraksi payload)	1.0	1.0	1.0	1.0	1.0
Waktu Penyisipan (detik)	0.526	0.937	1.262	1.642	2.019
Pemakaian Memori (Penyisipan) (MB)	140.10938	140.24219	141.21094	141.57813	141.68750
Waktu Recovery Audio (detik)	0.501	0.999	1.404	1.876	2.345
Waktu Ekstraksi (detik)	1.019	2.023	2.865	3.826	4.845
Pemakaian Memori (Ekstraksi) (MB)	147.48828	148.06641	148.62891	155.39453	156.65625

Tabel 5.13 Uji Coba Parameter *Payload Tanpa Location Map (POP)*

Nama Cover Audio	Pop.wav				
Ukuran Audio (KB)	861.33				
Kapasitas Pesan (digit)	441000				
Nama Payload (Byte)	1000 bytes.txt	2000 bytes.txt	3000 bytes.txt	4000 bytes.txt	5000 bytes.txt
Ukuran Payload	10096	20130	30164	40198	50222
Jumlah payload Base3 (digit)	50480	100650	150820	200990	251110
MSE	0.31377	0.44148	0.74184	0.85705	0.95580
PSNR (dB)	101.36354	99.88046	97.62650	96.99952	96.52592
Correlation (cover dan stego)	0.999999999899 2	0.999999999800 6	0.999999999440 2	0.999999999254 8	0.999999999075 5
Correlation(payload & ekstraksi payload)	-	-	-	-	-
Waktu Penyisipan (detik)	0.256	0.556	0.751	1.013	1.250
Pemakaian Memori (Penyisipan) (MB)	140.25000	141.49609	140.70313	140.12891	142.30469
Waktu Recovery Audio (detik)	-	-	-	-	-
Waktu Ekstraksi (detik)	0.140	0.272	0.376	0.497	0.631

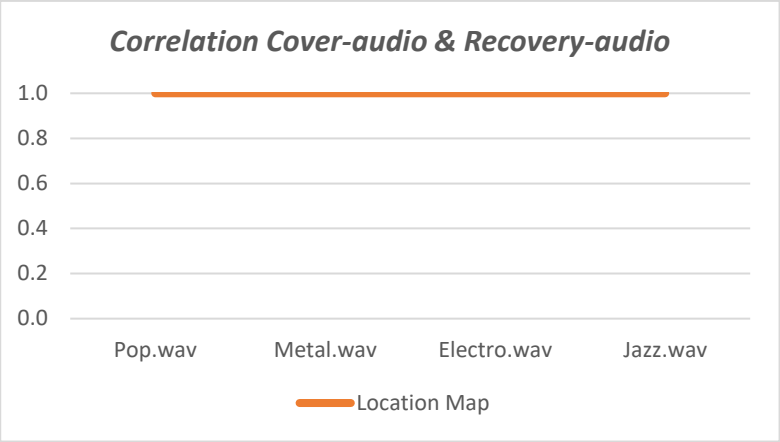
Pemakaian Memori (Ekstraksi) (MB)	146.65234	148.03906	148.04688	149.94531	153.13281

5.5 Evaluasi Uji Coba

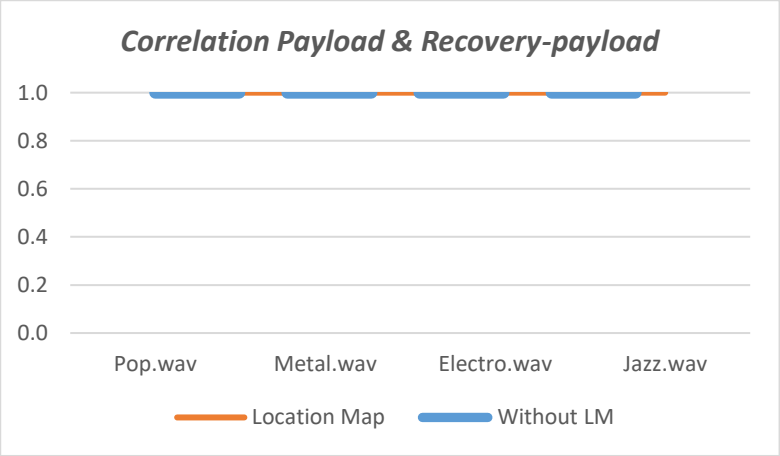
Pada bagian ini akan dijelaskan mengenai evaluasi berdasarkan hasil uji coba pada sub-bab 5.4. Evaluasi uji coba terdiri dari tiga bagian, yaitu evaluasi untuk uji coba fungsionalitas, evaluasi uji coba performa dan evaluasi untuk uji coba parameter *cover* dan *payload.*

5.5.1 Evaluasi Uji Coba Fungsionalitas

Dari hasil uji coba fungsionalitas dari metode yang diusulkan, maka dapat dilihat metode berjalan dengan baik. Hasil uji coba menunjukkan bahwa baik proses penyisipan, proses ekstraksi dan pengembalian nilai *sample* dari audio *(recovery-audio)* berhasil dilakukan. Seperti yang telah dijelaskan pada sub-bab 2.11, dimana nilai *correlation* semakin mendekati 1 semakin tinggi tingkat kemiripan antara kedua file begitu pula sebaliknya. Dari hasil uji coba, nilai *correlation* baik antara *cover-audio* dan *recovery-audio* serta antara *payload* dan hasil ekstrasi *payload*, keduanya (Gambar 5.1 & Gambar 5.2) bernilai 1. Hal ini menunjukkan bahwa proses penyisipan, ekstraksi dan pengembalian audio berhasil dilakukan.



Gambar 5.1 Nilai *Correlation Cover-audio & Recovery-audio* pada Masing Masing Genre Musik

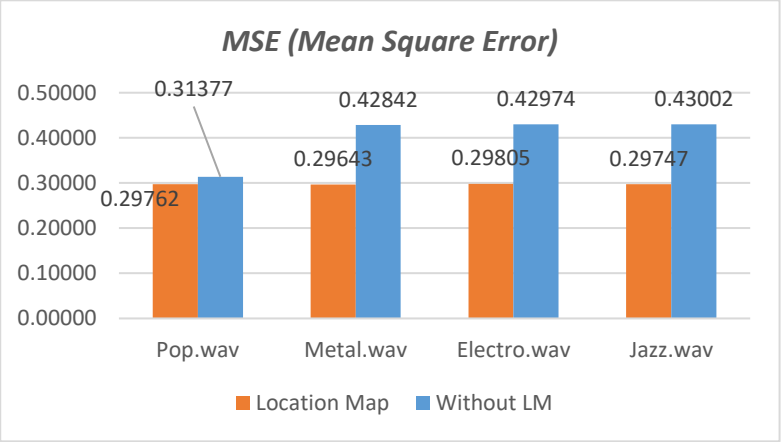


Gambar 5.2 Nilai *Correlation Payload & Recovery-payload* pada Masing Masing Genre Musik

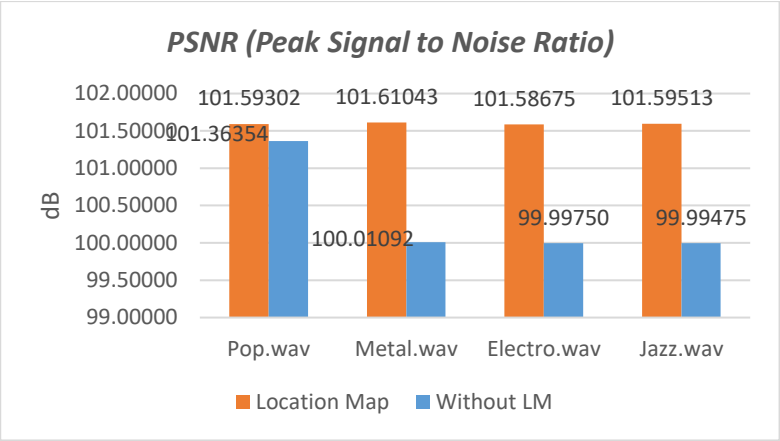
Dilihat dari hasil *MSE* yang didapatkan, menunjukkan bahwa kualitas audio yang dihasilkan sangat baik. Nilai *MSE*

menunjukkan bahwa, semakin besar nilai *MSE* semakin tinggi pula distorsi yang dihasilkan dan sebaliknya. Dari hasil uji coba (Gambar 5.3) didapatkan nilai *MSE* yang sangat kecil. Hal ini menunjukkan tingkat distorsi yang kecil pada saat proses penyisipan diterapkan.

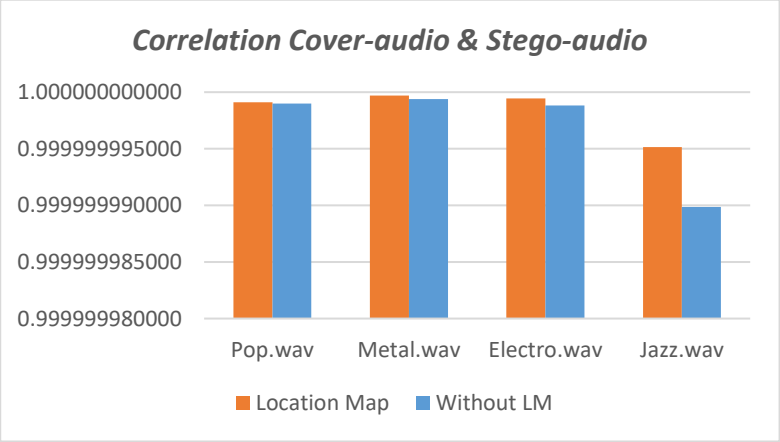
Nilai *PSNR* yang didapatkan juga menunjukkan hasil yang sangat baik. Dengan mengacu pada[8], kualitas steganografi yang baik pada audio memiliki nilai *PSNR* diatas 50dB[8]. Dari hasil uji coba (Gambar 5.4) didapatkan nilai rata-rata *PSNR* diatas 101 dB untuk penyisipan menggunakan *location map* dan diatas 99 dB untuk penyisipan tanpa menggunakan *location map* pada masing-masing genre musik. Hal ini juga dibuktikan dengan hasil dari nilai *correlation* yang dihasilkan antara *cover-audio* dan *stego-audio* (Gambar 5.5) yang dapat dikatakan mencapai nilai 1.0, hanya selisih $1\ x\ 10^{-8}$.



Gambar 5.3 Nilai *MSE* pada Masing Masing Genre Musik



Gambar 5.4 Nilai *PSNR* pada Masing Masing Genre Musik



Gambar 5.5 Nilai *Correlation Cover-audio & Stego-audio* pada Masing Masing Genre Musik

5.5.2 Evaluasi Uji Coba Performa

Dari hasil uji coba performa dari metode yang diusulkan, nilai *MSE* terendah didapatkan dari jenis penyisipan menggunakan *location map*. Tabel 5.14 menunjukkan nilai *MSE* dari masing-masing genre musik pada uji coba performa. Rata-rata nilai *MSE* yang didapatkan dari hasil uji coba adalah 0.87802 untuk proses peyisipan menggunakan *location map*, sedangkan untuk proses penyisipan tanpa *location map* didapatkan nilai *MSE* sebesar 1.1491. Dari nilai *MSE* yang didapatkan, menunjukkan proses penyisipan menggunakan *location map* menghasilkan *stego*-audio dengan distorsi yang lebih kecil dari pada penyisipan tanpa menggunakan *location map.* Hal ini disebabkan, pada penyisipan menggunakan *location map* memanfaatkan dua *sample* untuk dua metode penyisipan sekaligus. Penggunaan dua *sample* inilah yang menyebabkan peluang kemungkinan dari nilai *sample* kembali ke posisi awal seperti sebelum metode *RDE* diterapkan pada saat metode *SVM* berjalan. Selain itu pada proses penyisipan dengan *location map* menggunakan dua s*ample* untuk dua *digit payload*, sedangkan pada proses penyisipan tanpa *location map* menggunakan tiga s*ample* untuk dua *digit payload.* Hal ini menyebabkan banyaknya perubahan *sample* pada proses penyisipan tanpa menggunakan *location map.* Sehingga nilai distorsi pada proses penyisipan tanpa *location map* lebih besar.

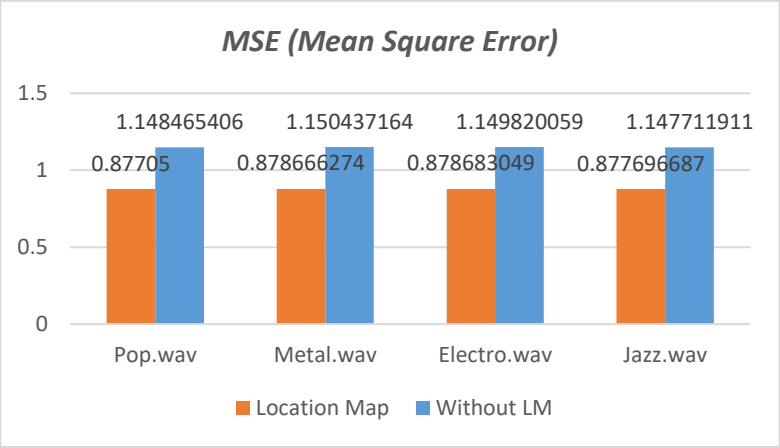
Nilai *PSNR* berbanding terbalik dengan nilai *MSE*, semakin kecil nilai *MSE* maka semakin besar nilai *PSNR* yang dihasilkan. Pada Tabel 5.15 menunjukkan nilai *PSNR* terbesar juga didapatkan pada proses penyisipan menggunakan *location map.* Untuk lebih jelasnya dapat dilihat pada Gambar 5.6 dan Gambar 5.7.

Tabel 5.14 Nilai *MSE* pada Uji Coba Performa

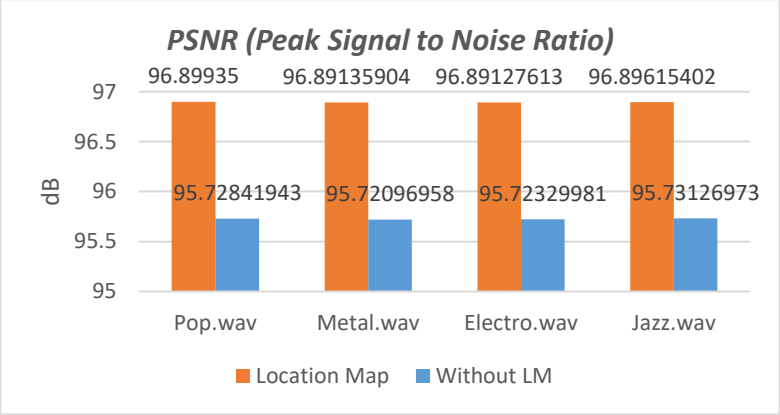
Genre musik	*Location Map*	*Tanpa LM*
POP	0.87705	1.14847
Metal	0.87867	1.15044
Electro	0.87868	1.14982
Jazz	0.87770	1.14771
Rata-rata	**0.87802**	**1.14911**

Tabel 5.15 Nilai *PSNR* pada Uji Coba Performa

Genre musik	*Location Map*	*Tanpa LM*
POP	96.89935 dB	95.72842 dB
Metal	96.89136 dB	95.72097 dB
Electro	96.89128 dB	95.72330 dB
Jazz	96.89615 dB	95.73127 dB
Rata-rata	**96.894535 dB**	**95.72599 dB**



Gambar 5.6 Nilai *MSE* Uji Coba Performa pada Masing Masing Genre Musik



Gambar 5.7 Nilai *PSNR* Uji Coba Performa pada Masing Masing Genre Musik

Tabel 5.16 menunjukkan nilai correlation antata cover-audio dan stego-audio. Sedangkan Tabel 5.17 menunjukkan nilai *correlation* antara *payload* dan *recovery-payload* atau hasil

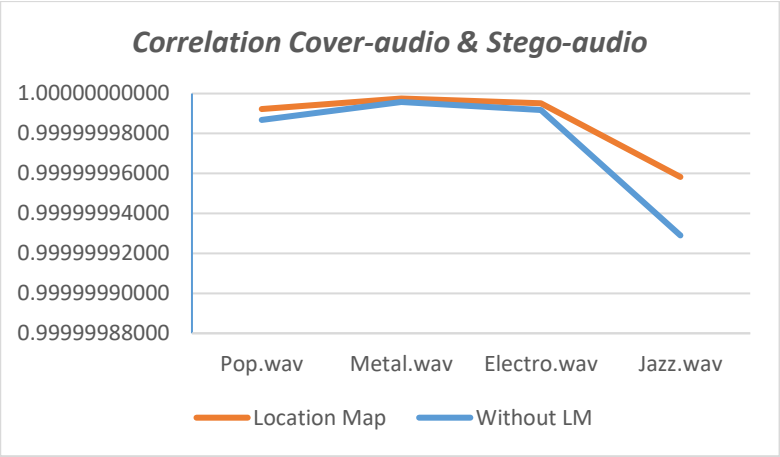
ekstraksi pesan. Tingkat kemiripan antara *cover-audio* dan *stego-audio* menunjukkan kuaitas yang sangat tinggi. Hal ini ditunjukan pada nilai *correlation* yang mendekati 1.0 hanya selisih $1\ x\ 10^{-8}$. dari nilai sempurna. Sedangkan nilai *correlation* antara *payload* dan *recovery-payload* menunjukkan nilai yang sempurna 1.0. Hal ini menunjukkan bahwa proses ekstraksi pesan berhasil dilakukan. Untuk lebih jelasnya dapat dilihat pada Gambar 5.8 Nilai *Correlation Cover-audio & Stego-audio* pada Masing Masing Genre Musik dan Gambar 5.9 Nilai *Correlation Payload & Recovery-payload* pada Masing Masing Genre Musik.

Tabel 5.16 *Correlation Cover-audio* dan *Stego-Audio*

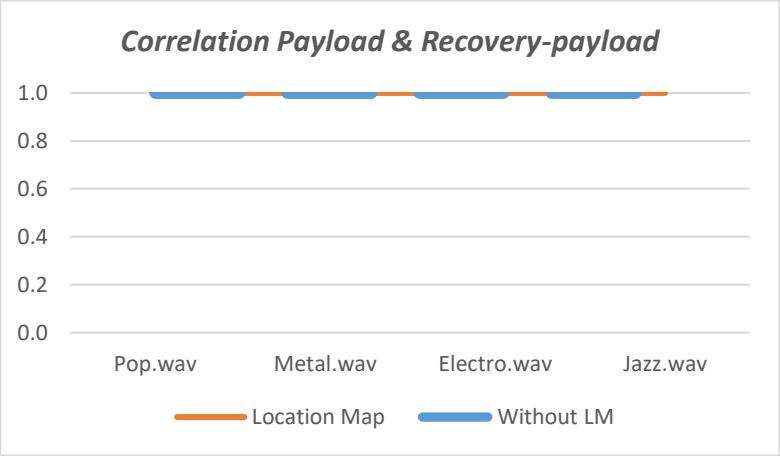
Genre musik	*Location Map*	*Tanpa LM*
POP	0.999999992	0.999999987
Metal	0.999999998	0.999999996
Electro	0.999999995	0.999999992
Jazz	0.999999958	0.999999929
Rata-rata	**0.999999986**	**0.999999976**

Tabel 5.17 *Correlation Payload* dan *Recovery-payload*

Genre musik	*Location Map*	*Tanpa LM*
POP	1.0	1.0
Metal	1.0	1.0
Electro	1.0	1.0
Jazz	1.0	1.0



Gambar 5.8 Nilai *Correlation Cover-audio & Stego-audio* pada Masing Masing Genre Musik



Gambar 5.9 Nilai *Correlation Payload & Recovery-payload* pada Masing Masing Genre Musik

Tabel 5.18 menunjukkan waktu yang diperlukan dalam proses penyisipan pesan, sedangkan Tabel 5.19 menunjukkan waktu yang diperlukan dalam proses ekstraksi *payload*. Waktu tercepat baik pada penyisipan dan ekstraksi *payload* didapatkan pada proses penyisipan tanpa menggunakan *location map.* Rata-rata waktu penyisipan menggunakan *location map* adalah 3,659 detik dan tanpa menggunakan *location map* adalah 1,513 detik. Sedangkan rata-rata waktu ekstraksi menggunakan *location map* adalah 4,340 detik dan tanpa menggunakan *location map* adalah 1,513 detik. Hal ini disebabkan karena pada proses penyisipan dan ekstraksi menggunakan *location map* dilakukan perhitungan yang lebih kompleks jika dibandingkan dengan penyisipan dan ekstraksi tanpa menggunakan *location map.* Untuk lebih jelasnya dapat dilihat pada Gambar 5.10 dan Gambar 5.11.

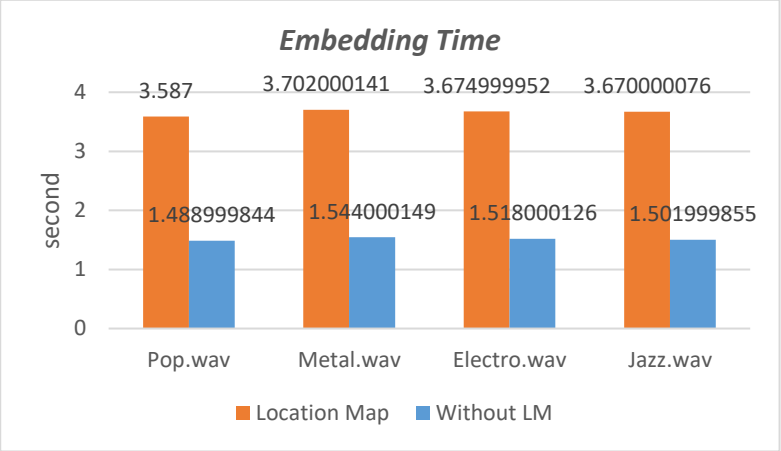
Tabel 5.18 Waktu Penyisipan *Payload* pada Uji Coba Performa

Genre musik	*Location Map*	*Tanpa LM*
POP	3,587 detik	1,489 detik
Metal	3,702 detik	1,544 detik
Electro	3,675 detik	1,518 detik
Jazz	3,670 detik	1,502 detik
Rata-rata	**3,659 detik**	**1,513 detik**

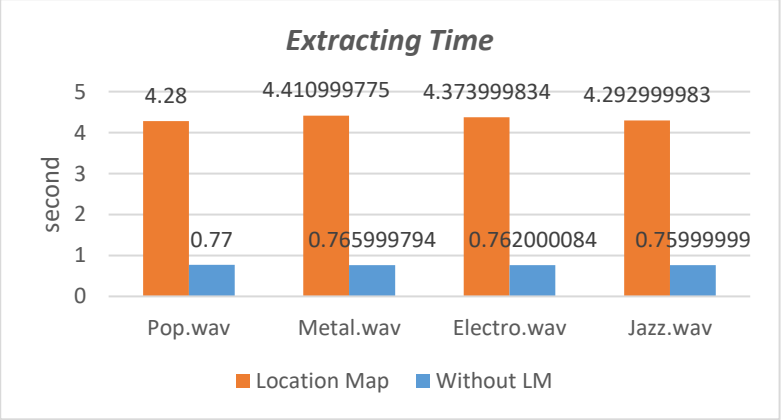
Tabel 5.19 Waktu Ekstraksi *Payload* pada Uji Coba Performa

Genre musik	*Location Map*	*Tanpa LM*
POP	4,280 detik	0,770 detik
Metal	4,411 detik	0,766 detik
Electro	4,374 detik	0,762 detik

Jazz	4,293 detik	0,760 detik
Rata-rata	**4,340 detik**	**0,765 detik**



Gambar 5.10 Waktu Penyisipan *Payload* Pada Masing Masing Genre Musik



Gambar 5.11 Waktu Ekstraksi *Payload* Pada Masing Masing Genre Musik

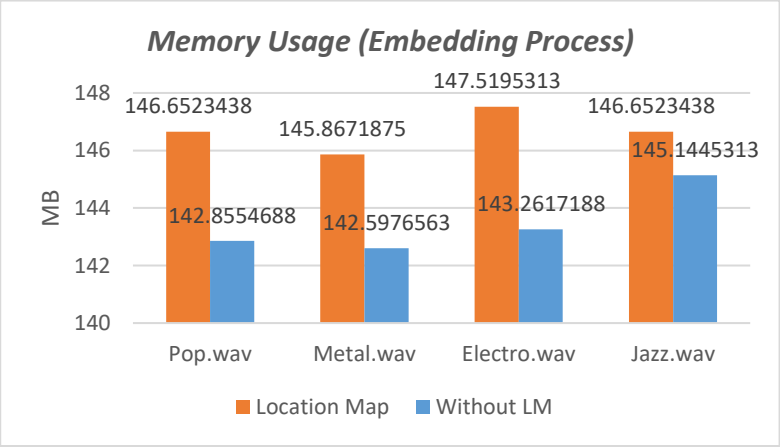
Tabel 5.20 menunjukkan total pemakaian memori yang digunakan dalam proses penyisipan pesan, sedangkan Tabel 5.21 menunjukkan total pemakaian memori yang digunakan dalam proses ekstraksi *payload.* Pemakaian memori terkecil baik pada penyisipan dan ekstraksi *payload* didapatkan pada proses penyisipan tanpa *location map.* Rata-rata pemakaian memori pada saat proses penyisipan menggunakan *location map* adalah 146.673 MB dan tanpa menggunakan *location map* adalah 143.465 MB. Sedangkan rata-rata pemakaian memori pada saat proses ekstraksi menggunakan *location map* adalah 163.296 MB dan tanpa menggunakan *location map* adalah 159.106 MB. Hal ini disebabkan karena pada proses penyisipan dan ekstraksi menggunakan *location map* dilakukan perhitungan yang lebih kompleks jika dibandingkan dengan penyisipan dan ekstraksi tanpa menggunakan *location map.* Selain itu dengan adanya pemakaian *location map* maka bertambah pula pemakaian memori yang diperlukan saat penyisipan dan ekstraksi *payload.* Untuk lebih jelasnya dapat dilihat pada Gambar 5.12 dan Gambar 5.13.

Tabel 5.20 Pemakaian Memori saat Proses Penyisipan *Payload* pada Uji Coba Performa

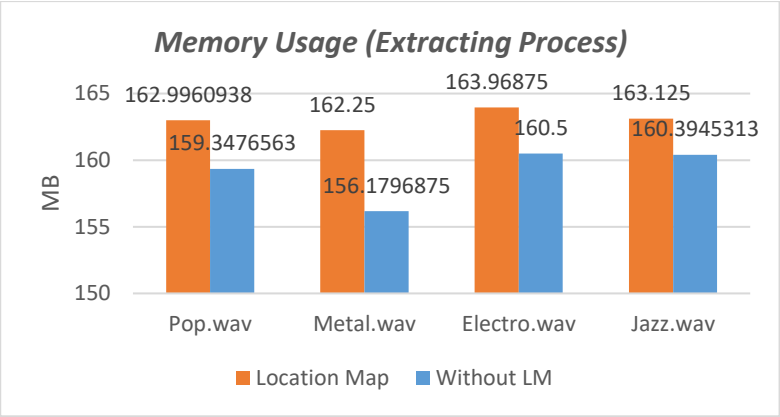
Genre musik	*Location Map*	*Tanpa LM*
POP	146.652 MB	142.855 MB
Metal	145.867 MB	142.598 MB
Electro	147.520 MB	143.262 MB
Jazz	146.652 MB	145.145 MB
Rata-rata	**146.673 MB**	**143.465 MB**

Tabel 5.21 Pemakaian Memori saat Ekstraksi *Payload* pada Uji Coba Performa

Genre musik	*Location Map*	*Tanpa LM*
POP	162.996 MB	159.348 MB
Metal	162.250 MB	156.180 MB
Electro	163.969 MB	160.500 MB
Jazz	163.969 MB	160.395 MB
Rata-rata	**163.296 MB**	**159.106 MB**



Gambar 5.12 Pemakaian Memori (Proses Penyisipan) pada Uji Coba Performa Pada Masing Masing Genre Musik



Gambar 5.13 Pemakaian Memori (Proses Ekstraksi) pada Uji Coba Performa Pada Masing Masing Genre Musik

5.5.3 Evaluasi Uji Coba Parameter *Cover* dan *Payload*

Terdapat dua evaluasi uij coba parameter dan payload. Masing-masing evaluasi uji coba akan dijelaskan pada sub-bab 5.5.3.1 dan 5.5.3.2.

5.5.3.1 Evaluasi Uji Coba Parameter *Cover*

Dari hasil uji coba parameter *cover* dari metode yang diusulkan, penggunaan parameter *cover* dari kecil ke besar menunjukkan bahwa semakin besar ukuran *cover* semakin besar pula nilai dari *PSNR.* Hal ini terjadi karena semakin besar ukuran dari *cover* semakin banyak *sample-sample* yang tidak digunakan untuk proses penyisipan pesan. Hal inilah yang membuat nilai dari *MSE* semakin kecil dan nilai *PSNR* semakin besar. Tabel 5.22 menunjukkan nilai *MSE* dari uji coba perameter *cover.* Dari nilai *MSE* yang didapatkan, menunjukkan proses penyisipan menggunakan *location map* menghasilkan *stego*-audio dengan distorsi yang lebih kecil dari pada penyisipan tanpa menggunakan

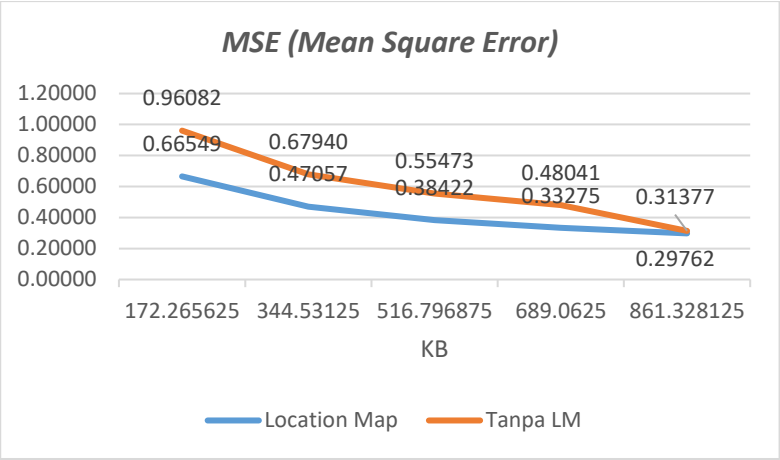
location map. Pada Tabel 5.23 menunjukkan nilai *PSNR* terbesar juga didapatkan pada proses penyisipan menggunakan *location map*. Untuk lebih jelasnya dapat dilihat pada Gambar 5.14 dan Gambar 5.15.

Tabel 5.22 Nilai *MSE* pada Uji Coba Parameter *Cover*

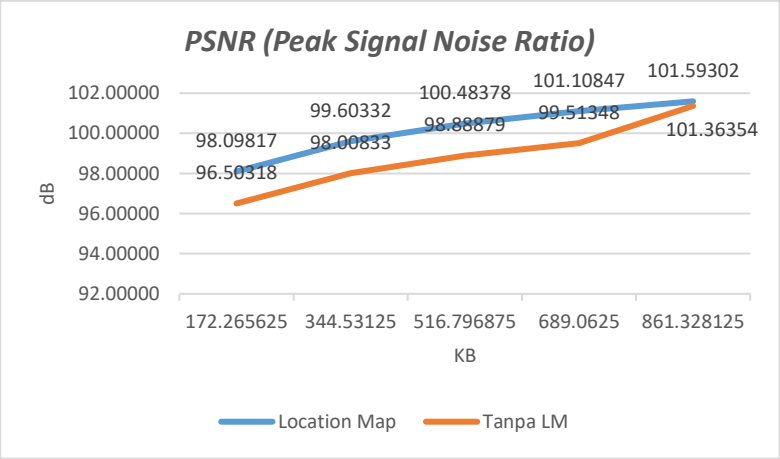
Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	0.66549	0.96082
344.53125	0.47057	0.67940
516.796875	0.38422	0.55473
689.0625	0.33275	0.48041
861.328125	0.29762	0.31377

Tabel 5.23 Nilai *PSNR* pada Uji Coba Parameter *Cover*

Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	98.09817 dB	96.50318 dB
344.53125	99.60332 dB	98.00833 dB
516.796875	100.48378 dB	98.88879 dB
689.0625	101.10847 dB	99.51348 dB
861.328125	101.59302 dB	101.36354 dB



Gambar 5.14 Nilai *MSE* pada Uji Coba Parameter Cover



Gambar 5.15 Nilai *PSNR* pada Uji Coba Parameter Cover

Tabel 5.24 menunjukkan nilai *correlation* antata *cover-audio* dan *stego-audio.* Sedangkan Tabel 5.25 menunjukkan nilai *correlation* antara *payload* dan *recovery-payload* atau hasil ekstraksi pesan. Tingkat kemiripan antara *cover-audio* dan *stego-*

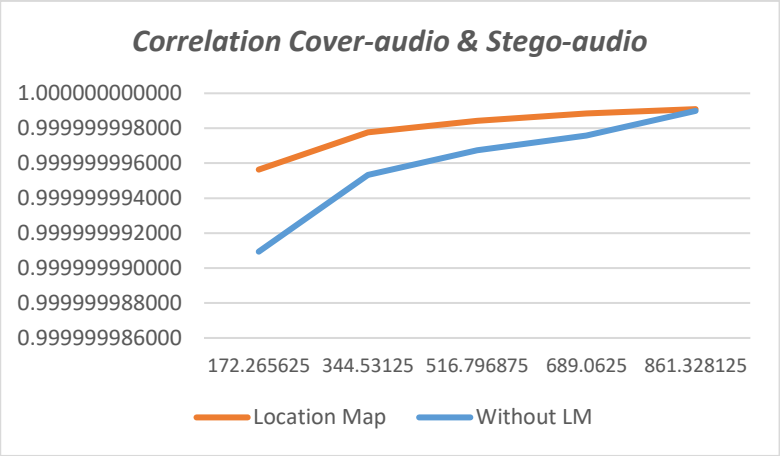
audio menunjukkan kuaitas yang sangat tinggi. Hal ini ditunjukan pada nilai *correlation* yang mendekati 1.0 hanya selisih $1\ x\ 10^{-8}$. dari nilai sempurna. Sedangkan nilai *correlation* antara *payload* dan *recovery-payload* menunjukkan nilai yang sempurna 1.0. Hal ini menunjukkan bahwa proses ekstraksi pesan berhasil dilakukan. Untuk lebih jelasnya dapat dilihat pada Gambar 5.16 Nilai *Correlation Cover-audio & Stego-audio* pada Uji Coba Parameter *Cover* dan Gambar 5.17 Nilai *Correlation Payload & Recovery-payload* pada Uji Coba Parameter *Cover*.

Tabel 5.24 *Correlation Cover-audio* dan *Stego-Audio* Uji Coba Parameter *Cover*

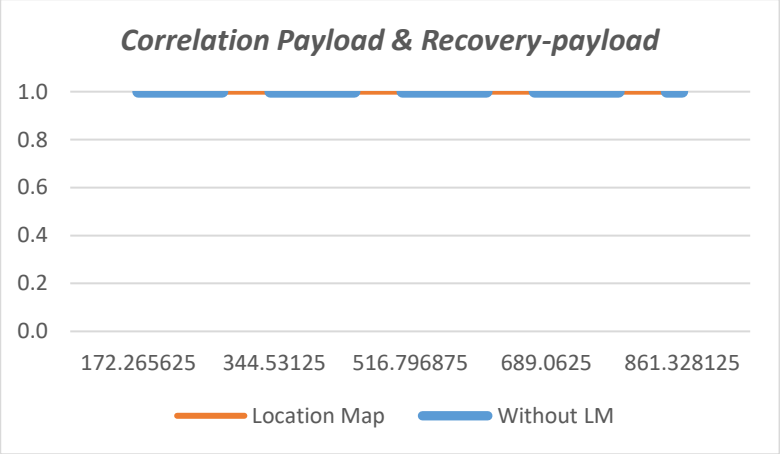
Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	0.999999999094	0.999999998992
344.53125	0.999999998199	0.999999998006
516.796875	0.999999997313	0.999999994402
689.0625	0.999999996422	0.999999992548
861.328125	0.999999995540	0.999999990755

Tabel 5.25 *Correlation Payload* dan *Recovery-payload* Uji Coba Parameter *Cover*

Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	1.0	1.0
344.53125	1.0	1.0
516.796875	1.0	1.0
689.0625	1.0	1.0
861.328125	1.0	1.0



Gambar 5.16 Nilai *Correlation Cover-audio & Stego-audio* pada Uji Coba Parameter *Cover*



Gambar 5.17 Nilai *Correlation Payload & Recovery-payload* pada Uji Coba Parameter *Cover*

Pada penggunaan parameter *cover*, semakin besar ukuran *cover* semakin lama waktu yang digunakan untuk proses penyisipan walaupun hanya terpaut 0.1 angka. Hal ini disebabkan karena penggunaan ukuran *sample* yang besar membutuhkan pengalokasian memori yang lebih besar sehingga membuat proses penyisipan dan ekstraksi sedikit lebih lama karena ukuran *cover* yang semakin besar. Tabel 5.26 menunjukkan waktu yang diperlukan dalam proses penyisipan pesan, sedangkan Tabel 5.27 menunjukkan waktu yang diperlukan dalam proses ekstraksi *payload.* Waktu tercepat baik pada penyisipan dan ekstraksi *payload* didapatkan pada proses penyisipan tanpa menggunakan *location map*. Untuk lebih jelasnya dapat dilihat pada Gambar 5.18 dan Gambar 5.19.

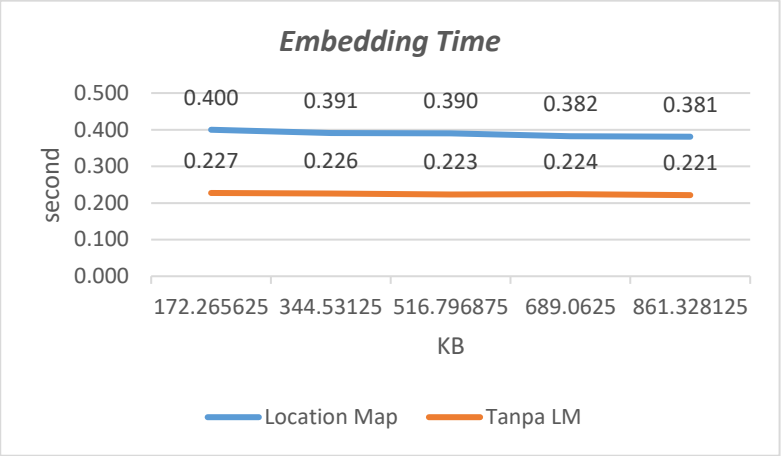
Tabel 5.26 Waktu Penyisipan *Payload* pada Uji Coba Parameter *Cover*

Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	0.381 detik	0.221 detik
344.53125	0.382 detik	0.223 detik
516.796875	0.390 detik	0.224 detik
689.0625	0.391 detik	0.226 detik
861.328125	0.400 detik	0.227 detik

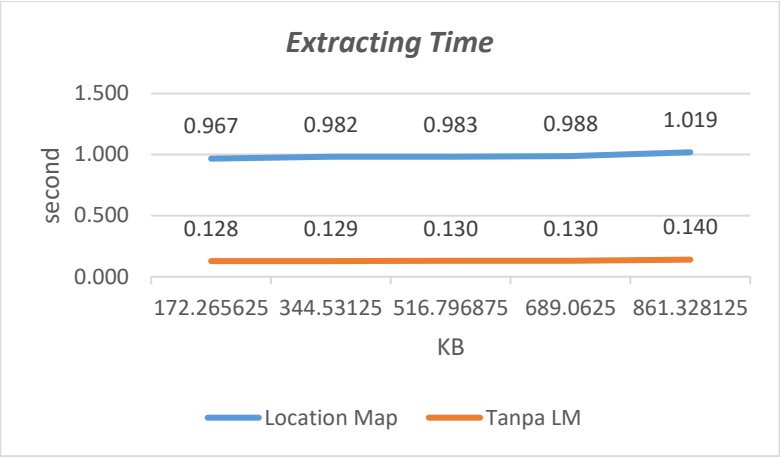
Tabel 5.27 Waktu Ekstraksi *Payload* pada Uji Coba Parameter *Cover*

Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	0.967 detik	0.128 detik
344.53125	0.982 detik	0.129 detik

516.796875	0.983 detik	0.130 detik
689.0625	0.988 detik	0.130 detik
861.328125	1.019 detik	0.140 detik



Gambar 5.18 Waktu Penyisipan *Payload* pada Uji Coba Parameter *Cover*



Gambar 5.19 Waktu Ekstraksi *Payload* pada Uji Coba Parameter *Cover*

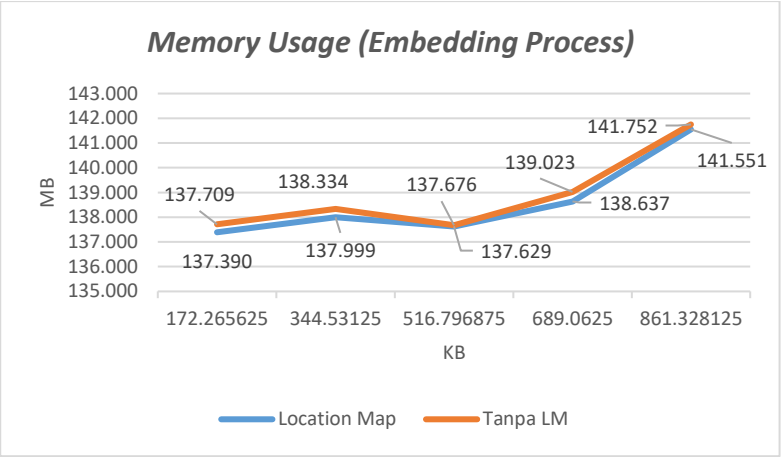
Pada penggunaan parameter *cover* juga menunjukkan bahwa semakin besar kapasitas *cover* semakin besar pula penggunaan memori yang digunakan baik pada proses penyisipan maupun proses ekstraksi *payload.* Tabel 5.28 menunjukkan total pemakaian memori yang digunakan dalam proses penyisipan pesan, sedangkan Tabel 5.29 menunjukkan total pemakaian memori yang digunakan dalam proses ekstraksi *payload.* Pemakaian memori terkecil baik pada penyisipan dan ekstraksi *payload* didapatkan pada proses penyisipan tanpa *location map.* Untuk lebih jelasnya dapat dilihat pada Gambar 5.20 dan Gambar 5.21.

Tabel 5.28 Pemakaian Memori saat Proses Penyisipan *Payload* pada Uji Parameter *Cover*

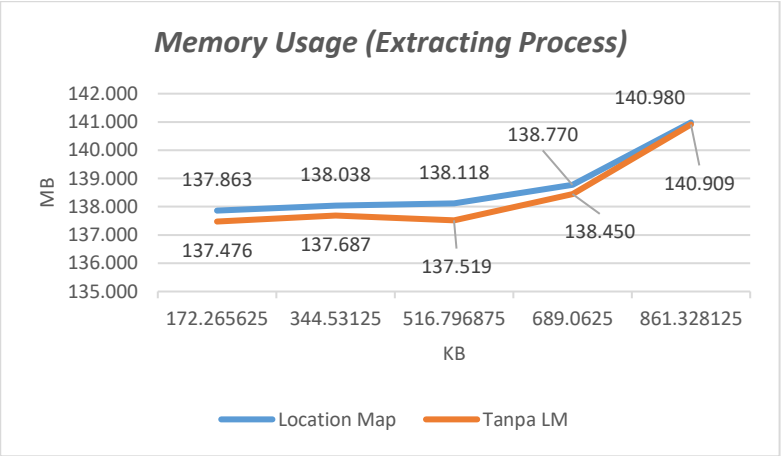
Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	137.390 MB	137.676 MB
344.53125	137.629 MB	137.709 MB
516.796875	137.999 MB	138.334 MB
689.0625	138.637 MB	139.023 MB
861.328125	141.551 MB	141.752 MB

Tabel 5.29 Pemakaian Memori saat Ekstraksi *Payload* pada Uji Coba Parameter *Cover*

Ukuran *Cover (Byte)*	*Location Map*	*Tanpa LM*
172.265625	137.863 MB	137.476 MB
344.53125	138.038 MB	137.687 MB
516.796875	138.118 MB	137.519 MB
689.0625	138.770 MB	138.450 MB
861.328125	140.980 MB	140.909 MB



Gambar 5.20 Pemakaian Memori saat Proses Penyisipan *Payload* pada Uji Parameter *Cover*



Gambar 5.21 Pemakaian Memori saat Ekstraksi *Payload* pada Uji Coba Parameter *Cover*

5.5.3.2 Evaluasi Uji Coba Parameter *Payload*

Dari hasil uji coba parameter *payload* dari metode yang diusulkan, penggunaan parameter *payload* dari kecil ke besar, menunjukkan bahwa semakin besar ukuran *payload* semakin besar pula nilai dari *MSE* yang didapatkan. Hal ini terjadi karena semakin besar ukuran dari *payload* semakin banyak *sample-sample* yang digunakan untuk proses penyisipan pesan. Hal inilah yang membuat nilai dari *PSNR* semakin kecil dan nilai *MSE* semakin besar.

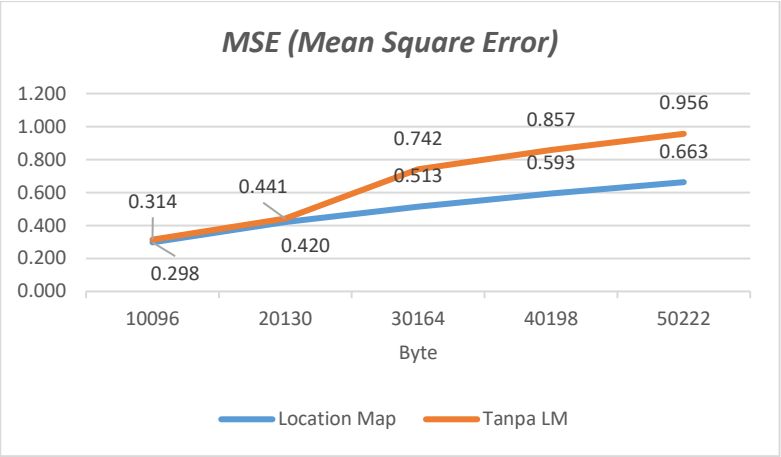
Nilai *MSE* terendah didapatkan dari jenis penyisipan menggunakan *location map*. Tabel 5.30 menunjukkan nilai *MSE* dari uji coba perameter *payload*. Dari nilai *MSE* yang didapatkan, menunjukkan proses penyisipan menggunakan *location map* menghasilkan *stego*-audio dengan distorsi yang lebih kecil dari pada penyisipan tanpa menggunakan *location map.* Sama seperti penjelasan sebelumnya, hal ini disebabkan pada penyisipan menggunakan *location map* memanfaatkan dua *sample* untuk dua metode penyisipan sekaligus. Penggunaan dua *sample* inilah yang menyebabkan peluang kemungkinan dari nilai *sample* kembali ke posisi awal sama seperti sebelum metode *RDE* diterapkan pada saat metode *SVM* berjalan. Nilai *PSNR* berbanding terbalik dengan nilai *MSE,* semakin kecil nilai *MSE* maka semakin besar nilai *PSNR* yang dihasilkan. Pada Tabel 5.31 menunjukkan nilai *PSNR* terbesar juga didapatkan pada proses penyisipan menggunakan *location map*. Untuk lebih jelasnya dapat dilihat pada Gambar 5.22 dan Gambar 5.23.

Tabel 5.30 Nilai *MSE* pada Uji Coba Parameter *Payload*

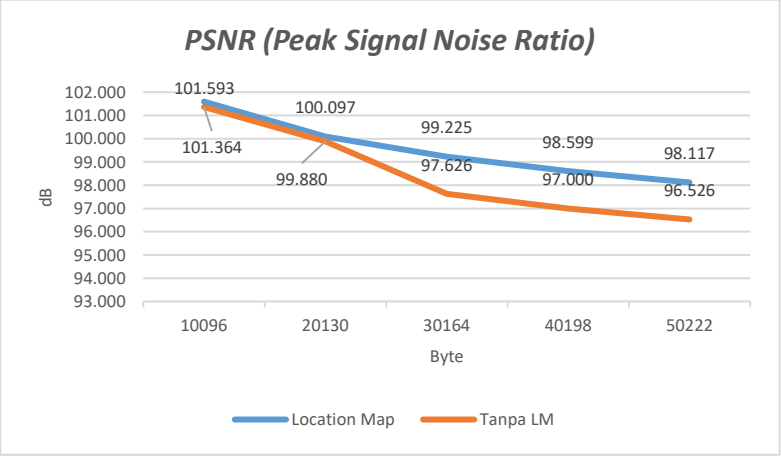
Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	0.29762	0.31377
20130	0.41997	0.44148
30164	0.51337	0.74184
40198	0.59296	0.85705
50222	0.66258	0.95580

Tabel 5.31 Nilai *PSNR* pada Uji Coba Parameter *Payload*

Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	101.59302 dB	101.36354 dB
20130	100.09745 dB	99.88046 dB
30164	99.22533 dB	97.62650 dB
40198	98.59933 dB	96.99952 dB
50222	98.11721 dB	96.52592 dB



Gambar 5.22 Nilai *MSE* pada Uji Coba Parameter *Payload*



Gambar 5.23 Nilai *PSNR* pada Uji Coba Parameter *Payload*

Tabel 5.32 menunjukkan nilai *correlation* antata *cover-audio* dan *stego-audio.* Sedangkan Tabel 5.33 menunjukkan nilai *correlation* antara *payload* dan *recovery-payload* atau hasil

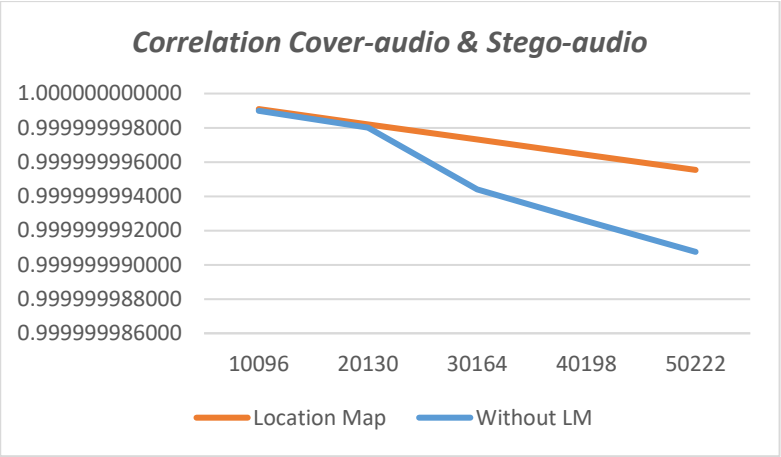
ekstraksi pesan. Tingkat kemiripan antara *cover-audio* dan *stego-audio* menunjukkan kuaitas yang sangat tinggi. Hal ini ditunjukan pada nilai *correlation* yang mendekati 1.0 hanya selisih $1\ x\ 10^{-8}$. dari nilai sempurna. Sedangkan nilai *correlation* antara *payload* dan *recovery-payload* menunjukkan nilai yang sempurna 1.0. Hal ini menunjukkan bahwa proses ekstraksi pesan berhasil dilakukan. Untuk lebih jelasnya dapat dilihat pada Gambar 5.24 dan Gambar 5.25

Tabel 5.32 *Correlation Cover-audio* dan *Stego-Audio* Uji Coba Parameter *Payload*

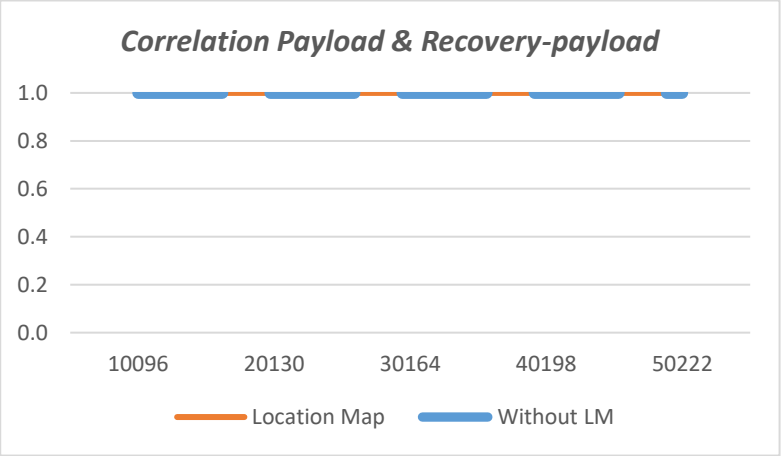
Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	0.999999999094	0.999999998992
20130	0.999999998199	0.999999998006
30164	0.999999997313	0.999999994402
40198	0.999999996422	0.999999992548
50222	0.999999995540	0.999999990755

Tabel 5.33 *Correlation Payload* dan *Recovery-payload* Uji Coba Parameter *Payload*

Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	1.0	1.0
20130	1.0	1.0
30164	1.0	1.0
40198	1.0	1.0
50222	1.0	1.0



Gambar 5.24 Nilai *Correlation Cover-audio* dan *Stego-Audio* Uji Coba Parameter *Payload*



Gambar 5.25 Nilai *Correlation Payload* dan *Recovery-payload* Uji Coba Parameter *Payload*

Dari penggunaan parameter *payload*, semakin besar ukuran *payload* semakin lama waktu yang digunakan untuk proses

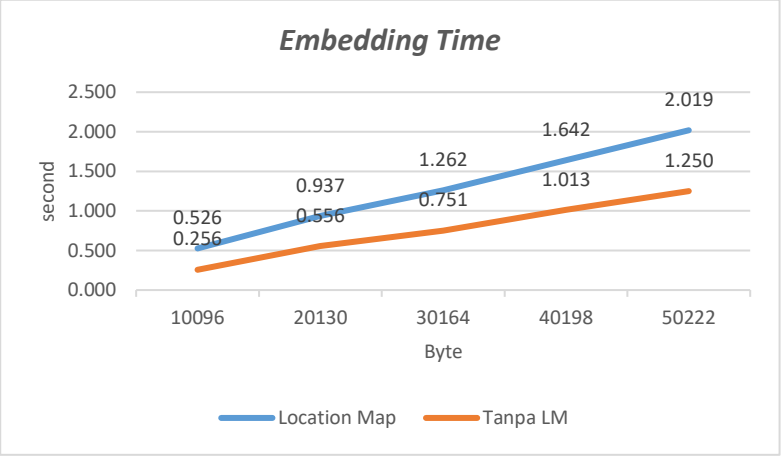
penyisipan. Hal ini disebabkan karena penggunaan ukuran *payload* yang besar membutuhkan pengalokasian waktu yang lebih lama pada proses penyisipan dan ekstraksi *payload*. Dengan semakin besarnya ukuran *payload* semakin banyak proses perhitungan yang dilakukan. Sehingga dibutuhkan waktu yang lebih lama untuk proses penyisipan dan ekstraksi *payload.* Tabel 5.34 menunjukkan waktu yang diperlukan dalam proses penyisipan pesan, sedangkan Tabel 5.35 menunjukkan waktu yang diperlukan dalam proses ekstraksi *payload.* Waktu tercepat baik pada penyisipan dan ekstraksi *payload* didapatkan pada proses penyisipan tanpa menggunakan *location map.* Untuk lebih jelasnya dapat dilihat pada Gambar 5.26 dan Gambar 5.27.

Tabel 5.34 Waktu Penyisipan *Payload* pada Uji Coba Parameter *Payload*

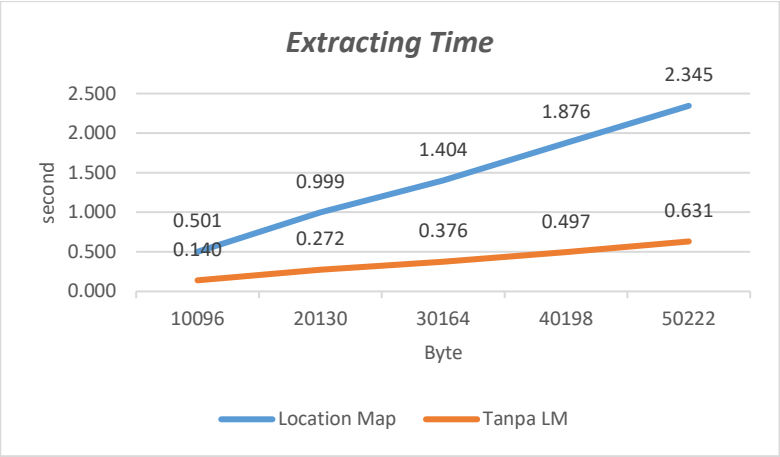
Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	0.526 detik	0.256 detik
20130	0.937 detik	0.556 detik
30164	1.262 detik	0.751 detik
40198	1.642 detik	1.013 detik
50222	2.019 detik	1.250 detik

Tabel 5.35 Waktu Ekstraksi *Payload* pada Uji Coba Parameter *Payload*

Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	0.501 detik	0.140 detik
20130	0.999 detik	0.272 detik
30164	1.404 detik	0.376 detik
40198	1.876 detik	0.497 detik
50222	2.345 detik	0.631 detik



Gambar 5.26 Waktu Penyisipan *Payload* pada Uji Coba Parameter *Payload*



Gambar 5.27 Waktu Ekstraksi *Payload* pada Uji Coba Parameter *Payload*

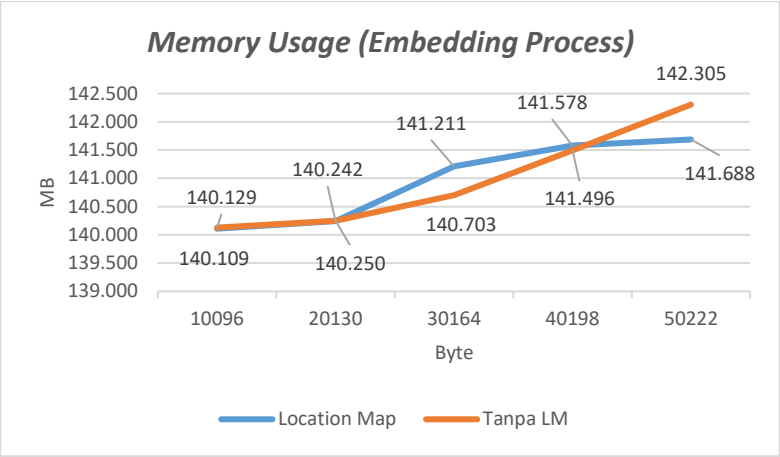
Penggunaan parameter *payload* juga berdampak pada penggunaan memori yang diperlukan. Semakin besar ukuran *payload* semakin besar pula memori yang dibutuhkan untuk proses penyisipan maupun proses ekstraksi pesan. Tabel 5.36 menunjukkan total pemakaian memori yang digunakan dalam proses penyisipan pesan, sedangkan Tabel 5.37 menunjukkan total pemakaian memori yang digunakan dalam proses ekstraksi *payload.* Untuk lebih jelasnya dapat dilihat pada Gambar 5.28 dan Gambar 5.28.

Tabel 5.36 Pemakaian Memori saat Proses Penyisipan *Payload* pada Uji Parameter *Payload*

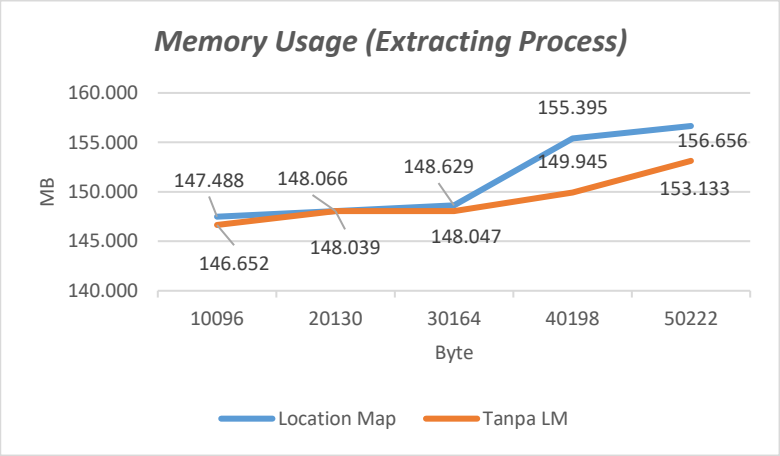
Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	140.109 MB	140.129 MB
20130	140.242 MB	140.250 MB
30164	141.211 MB	140.703 MB
40198	141.578 MB	141.496 MB
50222	141.688 MB	142.305 MB

Tabel 5.37 Pemakaian Memori saat Ekstraksi *Payload* pada Uji Coba Parameter *Payload*

Kapasitas *Payload*	*Location Map*	*Tanpa LM*
10096	147.488 MB	146.652 MB
20130	148.066 MB	148.039 MB
30164	148.629 MB	148.047 MB
40198	155.395 MB	149.945 MB
50222	156.656 MB	153.133 MB



Gambar 5.28 Pemakaian Memori saat Proses Penyisipan *Payload* pada Uji Parameter *Payload*



Gambar 5.29 Pemakaian Memori saat Ekstraksi *Payload* pada Uji Coba Parameter *Payload*

5.6 Evaluasi Umum Skenario Uji Coba

Dari beberapa uji coba yang telah dilakukan pada Sub bab 5.4 uji coba fungsionalita, performa dan kedua parameter, didapatkan beberapa hasil sebagai berikut:

1. Dari uji coba fungsionalitas didapatkan bahwa kombinasi metode steganografi *RDE* dan *SVM* dapat diterapkan pada media audio baik menggunakan *location map* maupun tanpa *location map.*
2. Dari uji coba performa didapatkan bahwa:
   a) Kapasitas maksimum penyisipan pesan pada proses penyisipan dengan *location map* lebih besar dari pada tanpa *location map* dengan kapasitas *payload* sebesar 441000 digit berbanding 293965 digit atau sebanding dengan 88200 *byte* berbanding 58793 *byte*.
   b) Nilai *MSE* pada proses penyisipan tanpa *location map* lebih besar dari pada menggunakan *location map* dengan rata-rata sebesar 1.15 berbanding 0.88.
   c) Nilai *PSNR* pada proses penyisipan dengan *location map* lebih besar dari pada tanpa *location map* dengan rata-rata sebesar 96.90 dB berbanding 95.73 dB.
   d) Nilai *correlation* antara *cover-audio* dan *stego-audio* pada proses penyisipan dengan *location map* lebih besar dari pada tanpa *location map* dengan rata-rata sebesar 0.999999986 berbanding 0.999999976. Sementara nilai *correlation* antara *payload* dan *recovery-payload* pada proses penyisipan dengan dan tanpa *location map* menunjukkan nilai yang sama, yaitu sebesar 1.0.
   e) Waktu penyisipan dan ekstraksi pesan pada proses penyisipan tanpa *location map* lebih cepat dari pada menggunakan *location map* dengan rata-rata waktu penyisipan sebesar 3.66 detik berbanding 1.51 detik dan waktu ekstraksi sebesar 4.34 detik berbanding 0.76 detik.

f) Penggunaan memori penyisipan dan ekstraksi pesan pada proses penyisipan menggunakan *location map* lebih besar dari pada tanpa *location map* dengan rata-rata penggunaan memori saat penyisipan sebesar 146.67 MB berbanding 143.46 MB dan penggunaan memori saat ekstraksi sebesar 163.08 MB berbanding 159.11 MB.

3. Dari uji coba parameter *cover* didapatkan:
   a. Semakin besar ukuran *cover* semakin kecil nilai *MSE*.
   b. Semakin besar ukuran *cover* semakin besar pula nilai *PSNR*.
   c. Semakin besar ukuran *cover* semakin besar nilai *correlation* antara *cover-audio* dan *stego-audio*.
   d. Semakin besar ukuran *cover* semakin cepat waktu yang diperlukan untuk proses penyisipan pesan.
   e. Semakin besar ukuran *cover* semakin lama waktu yang diperlukan untuk proses ekstraksi pesan.
   f. Semakin besar ukuran *cover* semakin besar memori yang diperlukan untuk proses penyisipan pesan.
   g. Semakin besar ukuran *cover* semakin besar memori yang diperlukan untuk proses ekstraksi pesan.
4. Dari uji coba parameter *payload* didapatkan:
   a. Semakin besar ukuran *payload* semakin besar nilai *MSE*.
   b. Semakin besar ukuran *payload* semakin kecil pula nilai *PSNR*.
   c. Semakin besar ukuran *payload* semakin kecil nilai *correlation* antara *cover-audio* dan *stego-audio*.
   d. Semakin besar ukuran *payload* semakin lama waktu yang diperlukan untuk proses penyisipan pesan.
   e. Semakin besar ukuran *payload* semakin lama waktu yang diperlukan untuk proses ekstraksi pesan.
   f. Semakin besar ukuran *payload* semakin besar memori yang diperlukan untuk proses penyisipan pesan.
   g. Semakin besar ukuran *payload* semakin besar memori yang diperlukan untuk proses ekstraksi pesan.

Dari segi tingkat keamanan informasi, kualitas dari *stego-audio* dan kapasitas, dapat disimpulkan bahwa proses penyisipan dengan *location map* lebih baik dari pada tanpa *location map.* Hal ini dapat dibuktikan dengan nilai *correlation* antara *cover-audio* dan *stego-audio* pada proses penyisipan menggunakan *location map* lebih besar dari pada tanpa *location map* dengan rata-rata sebesar 0.999999986 berbanding 0.999999976 dan kapasitas maksimum penyisipan *payload* 441000 digit berbanding 293965 digit atau sebanding dengan 88200 *byte* berbanding 58793 *byte*.

Sedangkan proses penyisipan tanpa *location map* lebih baik dari segi proses penyisipan pesan yang efisien. Hal ini dibuktikan dari waktu penyisipan dan ekstraksi pesan pada proses penyisipan tanpa *location map* lebih cepat dari pada menggunakan *location map* dengan rata-rata waktu penyisipan sebesar 3.66 detik berbanding 1.51 detik dan waktu ekstraksi sebesar 4.34 detik berbanding 0.76 detik serta penggunaan memori penyisipan dan ekstraksi pesan dengan penggunaan memori saat penyisipan sebesar 146.67 MB berbanding 143.46 MB dan penggunaan memori saat ekstraksi sebesar 163.08 MB berbanding 159.11 MB.

BAB VI
KESIMPULAN DAN SARAN

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

6.1 Kesimpulan

Kesimpulan yang didapatkan berdasarkan hasil uji coba metode steganografi yang diusulkan metode *RDE dan SVM* adalah sebagai berikut:

1. Penggunaan *Modulus Function (base3)* dapat dimanfaatkan untuk proses penyisipan pesan serta dapat meningkatkan kapasitas penyimpanan pesan pada media audio.
2. Metode *Reduced Difference Expansion* dan *Sample Value Modification* dapat diterapkan pada media audio dengan kombinasi antara keduanya.
3. Kombinasi metode *RDE* dan *SVM* dapat dimanfaatkan untuk meningkatkan tingkat keamanan informasi, kualitas, kapasitas serta proses penyisipan pesan yang efisien dengan proses penyisipan dengan dan tanpa *location map*.

6.2 Saran

Berikut merupakan beberapa saran untuk pengembangan perangkat lunak di masa yang akan datang saran diberikan terkait pengembangan pada Tugas Akhir ini adalah:

1. Diperlukan pengembangan lebih lanjut terhadap pengembalian nilai *sample audio* (*reversible-audio)* pada metode *RDE* dan *SVM* tanpa *location map.*
2. Diperlukan pengujian lebih lanjut untuk menentukan pengaruh genre musik terhadap peningkatan kualitas, serta proses

penyisipan pesan yang efisien dalam penerapan metode steganografi dalam Tugas Akhir ini.

3. Menggunakan metode steganografi yang lain untuk menentukan pengaruh genre musik terhadap peningkatan kualitas, serta proses penyisipan pesan yang efisien.

DAFTAR PUSTAKA

[1] N. F. Johnson dan S. Jajodia, "Exploring steganography: Seeing the unseen," *Computer*, vol. 31, no. 2, hal. 26–34, Feb 1998.

[2] D.-C. Lou, M.-C. Hu, dan J.-L. Liu, "Multiple layer data hiding scheme for medical images," *Comput. Stand. Interfaces*, vol. 31, no. 2, hal. 329–335, Feb 2009.

[3] Jun Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, hal. 890–896, Agu 2003.

[4] T. Ahmad, M. Holil, W. Wibisono, dan I. R. Muslim, "An improved Quad and RDE-based medical data hiding method," in *Computational Intelligence and Cybernetics (CYBERNETICSCOM), 2013 IEEE International Conference on*, 2013, hal. 141–145.

[5] M. Hatem Ali Al-Hooti, S. Djanali, dan T. Ahmad, "Audio Data Hiding Based on Sample Value Modification Using Modulus Function," *J. Inf. Process. Syst.*, 2016.

[6] V. Nagaraj, V. Vijayalakshmi, dan G. Zayaraz, "Color Image Steganography based on Pixel Value Modification Method Using Modulus Function," *IERI Procedia*, vol. 4, hal. 17–24, Jan 2013.

[7] G. Kasana, S. Kulbir, dan S. Singh Bhatia, "Data Hiding Algorithm for Images Using Discrete Wavelet Transform and Arnold Transform," *J. Inf. Process. Syst.*, 2015.

[8] S. T. Welstead, *Fractal and Wavelet Image Compression Techniques*. SPIE Press, 1999.

[9] "Welcome to Python.org," *Python.org*. [Daring]. Tersedia pada: https://www.python.org/. [Diakses: 14-Des-2016].

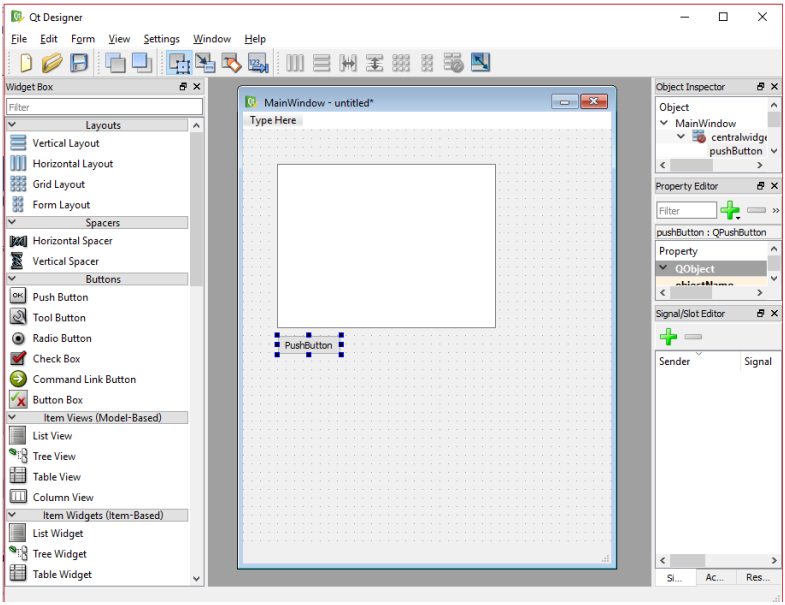
[10] "PyPI - the Python Package Index : Python Package Index." [Daring]. Tersedia pada: https://pypi.python.org/pypi. [Diakses: 30-Jun-2017].

[11] “Anaconda | Continuum.” [Daring]. Tersedia pada: https://www.continuum.io/anaconda-overview. [Diakses: 30-Jun-2017].
[12] “R: The R Project for Statistical Computing.” [Daring]. Tersedia pada: https://www.r-project.org/. [Diakses: 01-Jul-2017].
[13] “The Scala Programming Language.” [Daring]. Tersedia pada: https://www.scala-lang.org/. [Diakses: 01-Jul-2017].
[14] “Continuum,” *Continuum*. [Daring]. Tersedia pada: https://www.continuum.io/. [Diakses: 15-Des-2016].
[15] “SciPy.org — SciPy.org.” [Daring]. Tersedia pada: https://www.scipy.org/index.html. [Diakses: 01-Jul-2017].
[16] “Numpy and Scipy Documentation — Numpy and Scipy documentation.” [Daring]. Tersedia pada: https://docs.scipy.org/doc/. [Diakses: 15-Des-2016].
[17] “NumPy — NumPy.” [Daring]. Tersedia pada: http://www.numpy.org/. [Diakses: 01-Jul-2017].
[18] “History and License — Python 3.6.2rc1 documentation.” [Daring]. Tersedia pada: https://docs.python.org/3/license.html. [Diakses: 01-Jul-2017].
[19] “Matplotlib: Python plotting — Matplotlib 2.0.2 documentation.” [Daring]. Tersedia pada: https://matplotlib.org/. [Diakses: 01-Jul-2017].
[20] “PyQt/Tutorials - Python Wiki.” [Daring]. Tersedia pada: https://wiki.python.org/moin/PyQt/Tutorials. [Diakses: 15-Des-2016].
[21] “PyQt_first/pyqt_skeleton.py at master · shantnu/PyQt_first · GitHub.” [Daring]. Tersedia pada: https://github.com/shantnu/PyQt_first/blob/master/pyqt_skeleton.py. [Diakses: 11-Jul-2017].

LAMPIRAN

Lampiran A. User Interface

User interface pada tugas akhir ini dibangun menggunakan Qt Designer dimana modul-modul seperti *toolbar*, *menubar*, *layout*, *button*, *label*, *tab widget*, *textviews*, *form*, *dropdown*, *textearea* dan lain sebagainya dibuat dengan fitur *drag and drop*. Lampiran A 1 merupakan tampilan dari Qt Designer.



Lampiran A 1 Tampilan Antar Muka *Qt Designer*

Setelah tampilan berhasil dibuat, selanjutnya akan disimpan untuk proses selanjutnya. Hasil keluaran atau *output* dari Qt Designer adalah file berekstensi *.ui*. Dimana file *.ui* tersebut dibangun dengan bahasa pemprograman *xml*. Lampiran A 2

merupakan tampilan potongan kode program dari file berekstensi *.ui.*

```
1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <ui version="4.0">
3.   <class>MainWindow</class>
4.   <widget class="QMainWindow" name="MainWindow">
5.    <property name="geometry">
6.     <rect>
7.      <width>658</width>
8.      <height>560</height>
9.     </rect>
10.   </property>
11.   ...
12.   ...
13. </ui>
```

Lampiran A 2 Potongan Kode Program File berekstensi .ui

Lampiran A 3 merupakan kerangka kode awal untuk memulai sebuah projek PyQt. Dimana file berekstensi *.ui* hasil dari proses sebelumnya akan dimuat pada kerangka awal tersebut. File *.ui* dimuat pada baris ke empat dari kode program. Setelah memuat file *.ui* barulah kode program untuk melakukan fungsi-fungsi dari tombol-tombol atau *textview* ditambahkan. Pada baris ke delapan terdapat kelas *MyApp.* Pada kelas ini dilakukan proses inisialisasi dimana *MainWindow* pertama kali dipanggil. Dalam kelas ini nantinya juga akan dilakukan proses inisialisasi aksi. Contohnya pada saat tombol ditekan, maka akan memanggil fungsi untuk melakukan aksi seperti membuka *text dialog*, *openfile* dan lain sebagainya. Pada baris 14 hingga 18 dari kode program merupakan *main program* dimana pada baris ini, baris yang pertama kali dibaca pada saat program berjalan.

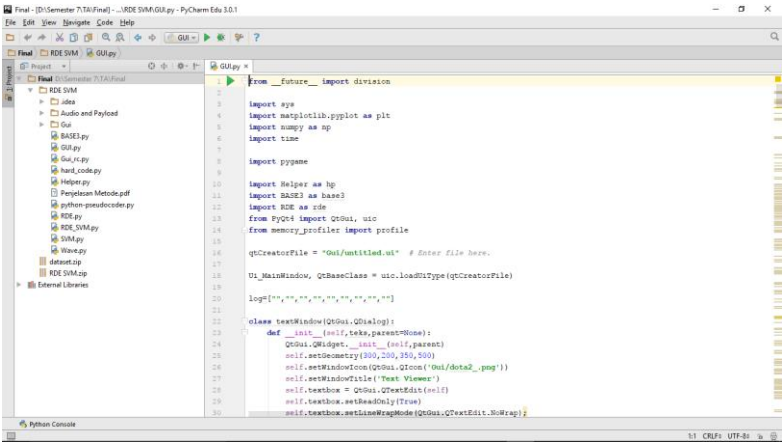
```
18 lines (14 sloc)  |  455 Bytes
```

```python
import sys
from PyQt4 import QtCore, QtGui, uic

qtCreatorFile = "" # Enter file here.

Ui_MainWindow, QtBaseClass = uic.loadUiType(qtCreatorFile)

class MyApp(QtGui.QMainWindow, Ui_MainWindow):
    def __init__(self):
        QtGui.QMainWindow.__init__(self)
        Ui_MainWindow.__init__(self)
        self.setupUi(self)

if __name__ == "__main__":
    app = QtGui.QApplication(sys.argv)
    window = MyApp()
    window.show()
    sys.exit(app.exec_())
```

Lampiran A 3 Kerangka Kode Awal *(skeleton) PyQt Project* [21]

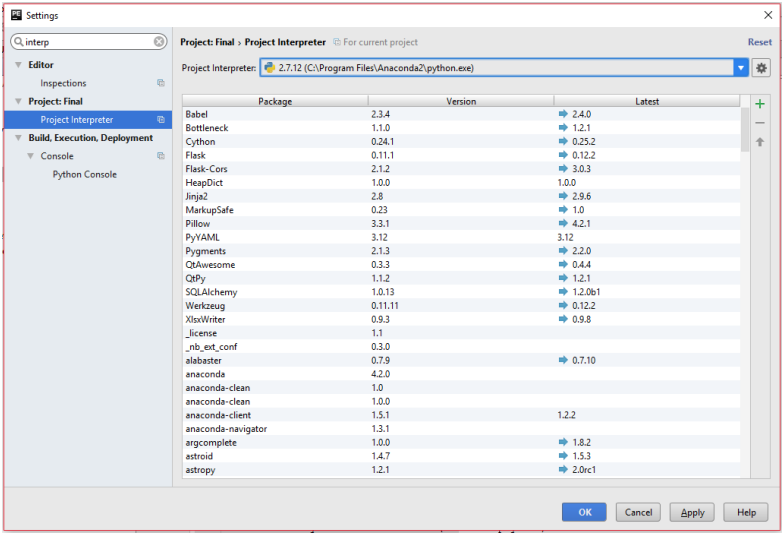
PyCharm merupakan IDE untuk bahasa pemprograman Python. PyCharm digunakan untuk menggabungkan file hasil dari Qt Designer dengan kode program agar tombol yang ada pada *user interface* dapat berfungsi. Lampiran A 4 merupakan tampilan dari PyCharm

.



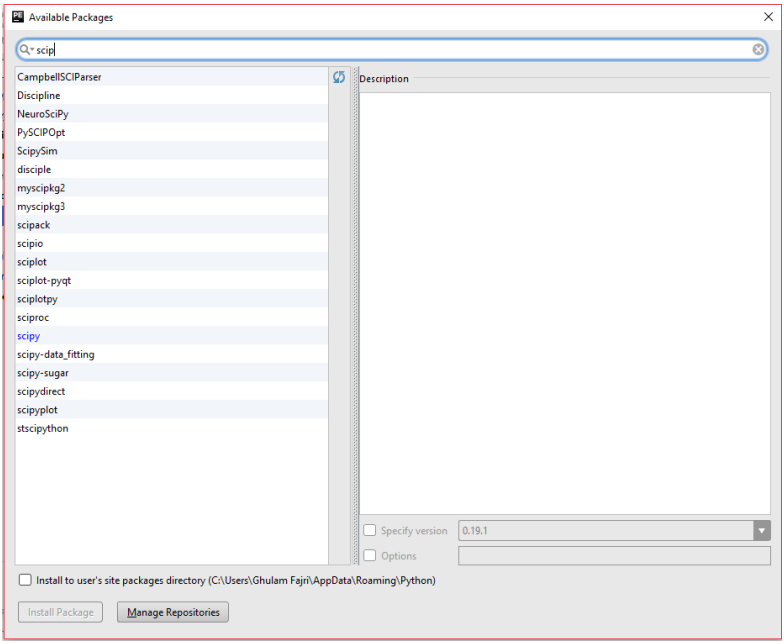
Lampiran A 4 Tampilan *PyCharm*

Sementara Lampiran A 5 merupakan tampilan menu interpreter. Anaconda dipilih sebagai interpreter, dimana didalamnya terdapat kumpulan paket yang digunakan untuk pengolahan audio. Paket-paket yang digunakan diantaranya SciPy, NumPy, Matplotlib dan PyQt.



Lampiran A 5 Menu Project Interpreter

Lampiran A 6 merupakan fitur pencarian paket melalui interpreter. Melalui menu ini pengguna dapat dengan mudah mencari paket yang dibutuhkan dari proses unduh paket hingga paket berhasil dipasang.



Lampiran A 6 Pencarian Paket

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Muhammad Ghulam Fajri dilahirkan pada tanggal 30 Agustus 1995 di Pasuruan. Saat ini sedang menempuh pendidikan perguruan tinggi di jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya.

Memiliki pengalaman organisasi sebagai Staf Departemen Hubungan Luar Himpunan Teknik Computer-Informatika 2014-2015 dan 2015-2016, Staf Ahli Departemen Hubungan Kelembagaan Keluarga Muslim Informatika 2015-2016. Selain itu, juga memiliki pengalaman kepanitiaan, diantaranya sebagai Staf National Logic Competiton Schematics 2014 dan Staf Perlengkapan-Transportasi Schematics 2015 serta turut menjadi Administrator di Laboratorium NCC (Net Centrict Computing) Teknik Informatika ITS.

Kritik dan saran sangat diharapkan guna peningkatan kualitas pada penulisan selanjutnya. Penulis dapat dihubungi melalui email: ghulamfajri13@gmail.com