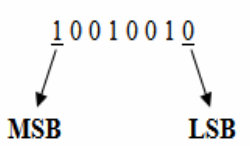
BAB II

LANDASAN TEORI

Bab ini berisi penjelasan mengenai teori – teori yang berkaitan dengan skripsi. Dasar teori yang akan dijelaskan meliputi penjelasan mengenai citra, penjelasan mengenai citra *GIF*, penjelasan mengenai steganografi dan penjelasan mengenai metode *LSB* dan Adaptif. Seluruh dasar teori yang dijelaskan akan menjadi dasar pengerjaan skripsi.

II.1. Metode *LSB*

Cara penyisipan pesan menggunakan metode ini adalah dengan cara mengganti *LSB* dari media penyisipan dengan *bit – bit* pesan yang akan disembunyikan. Metode ini biasanya dilakukan pada media digital yang tidak terkompresi, seperti citra *bitmap* 24-bit dan Video AVI. Metode yang digunakan untuk menyembunyikan pesan pada media digital tersebut berbeda-beda. Contohnya pada *file image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (*LSB*) pada data *pixel* yang menyusun *file* tersebut. Seperti kita ketahui untuk *file bitmap* 24 bit maka setiap *pixel* (titik) pada gambar tersebut terdiri dari susunan tiga warna yaitu merah, hijau, dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bit (*byte*) dari 0 sampai 255 atau dengan format biner
00000000 sampai 11111111.



Gambar II.1. *Most Significant Bit* (MSB) dan *Least Significant Bit* (LSB)
Sumber : Putri Alatas ; Universitas Gunadarma 2009

Bit yang cocok untuk diganti adalah bit *LSB*, sebab perubahan tersebut hanya mengubah nilai *byte* satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Misalkan *byte* tersebut menyatakan warna merah, maka perubahan satu bit *LSB* tidak mengubah warna merah tersebut secara berarti, dan mata manusia tidak dapat membedakan perubahan yang sangat kecil itu.

II.2. Metode Adaptif

Teknik penyisipan pesan pada steganografi secara teknik penyisipannya dapat dikategorikan ke dalam dua buah kategori, yaitu :

1. Teknik non adaptif

   Proses penyisipan pesan teknik non adaptif tidak mengkorelasikan fitur pada media penyisipan dengan pesan yang akan disisipkan, pada kasus citra ini. Contohnya adalah pada penyisipan dengan metode LSB yang menyisipkan bit – bit pesan secara acak di media penyisipan.

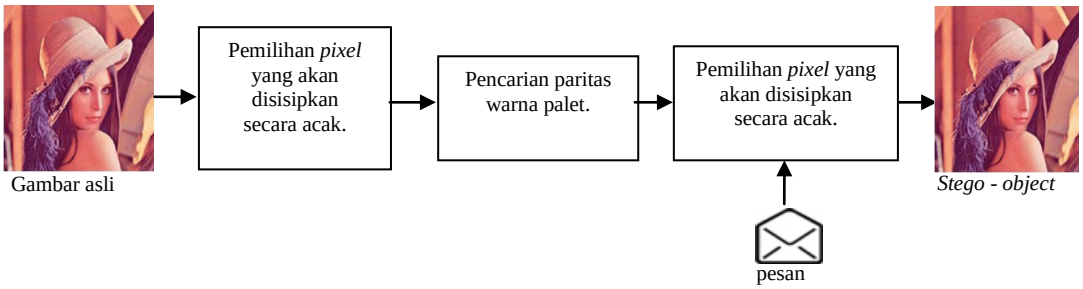
2. Teknik adaptif

   Proses penyisipan pesan pada teknik Adaptif modifikasi media penyisipan yang terjadi pada proses penyisipan pesan pada teknik

Adaptif modifikasi media penyisipan yang terjadi pada proses penyisipan pesan dikolerasikan dengan fitur dan konten citra. Teknik ini akan menganalisis dan memilih *pixel* yang akan disisipkan pesan, dan *pixel* mana yang akan disisipkan tergantung dengan media penyisipan. Contohnya teknik ini akan dapat menghindari daerah pada citra yang mempunyai warna sama (*solid color*) dan sehingga teknik ini akan memilih *pixel* berdasarkan nilai paritas dari *pixel* yang akan disisipkan oleh pesan dibandingkan dengan nilai paritas dari pesan yang akan disisipkan.

Metode Adaptif adalah metode yang menggunakan teknik Adaptif sebagai teknik penyisipan pesannya. Metode ini mempunyai keunggulan yaitu pada tingkat keamanan yang tinggi untuk steganografi pada citra berbasis palet (*palette – based image*), yaitu citra *GIF* dan *PNG*. Metode ini akan menganalisis dan memilih *pixel* yang tidak akan menghasilkan kecurigaan yang besar dengan perubahan nilai warna, contohnya metode ini akan menghindari penyisipan pesan pada sebuah daerah yang mempunyai warna yang sama pada sebuah citra. Pada citra *GIF*, metode ini disebut Adaptif karena pada proses penyisipan dan modifikasi *pixel*nya, metode ini akan memilih warna yang tersedia di tabel warna yang ada pada citra yang disisipkan pesan. Karena masing – masing citra *GIF* mempunyai konten warna tabel yang berbeda – beda.

Kapasitas penyisipan pesan metode ini bergantung pada media penyisipan, dan pada beberapa kasus, kapasitas dari pesan yang dapat disisipkan dalam sebuah citra dan dalam sekali penyisipan tidak dapat dihitung sebelum proses penyisipan dimulai, namun masih dapat dilakukan analisa mengenai kapasitas maksimum dari pesan yang disisipkan menggunakan metode Adaptif. Cara kerja metode Adaptif pada proses penyisipan dimulai dengan pemilihan *pixel* yang akan disisipkan oleh pesan secara semu acak. Proses penyisipan pesan pada citra *GIF* menggunakan metode Adaptif dapat di lihat pada gambar II.2.

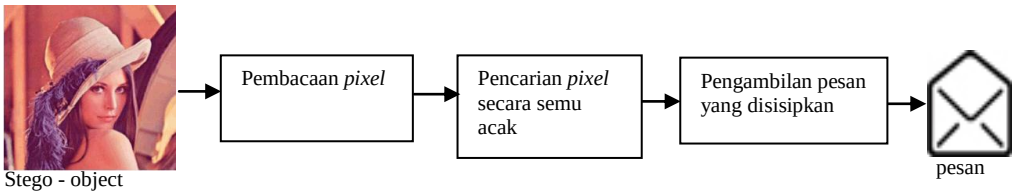


Gambar II.2 Penyembunyian Pesan dengan Menggunakan Metode Adaptif. Sumber : Andy Wicaksono ; ITB 2009

Pixel yang akan disisipkan pesan dipilih secara semu acak dari kumpulan *pixel* yang terdapat pada berkas citra. Lalu warna palet akan ditentukan paritas bitnya dengan menggunakan persamaan R + G + B *mod* 2. Setelah itu diadakan pengecekan pada setiap *pixel* dan membandingkan paritas dari setiap *pixel* dengan *bit* pesan. Jika dihasilkan persamaan *bit*, maka *pixel* tersebut tidak dimodifikasi dan pengecekan berlanjut pada *pixel* selanjutnya. Jika tidak terjadi

perbedaan *bit*, maka *pixel* warna tersebut di modifikasi dengan cara mencari warna tetangga terdekat yang mempunyai paritas yang sama pada palet warna.

Jika proses modifikasi selesai, maka pengecekan di lanjutkan pada *pixel* selanjutnya. Prosedur penyisipan pesan menjamin bahwa kumpulan blok pada berkas citra asli dan citra yang sudah di modifikasi adalah identik. Hal ini memungkinkan algoritma pendeteksi untuk mengembalikan pesan dari paritas warna dengan cara melakukan penelusuran pada *pixel – pixel* secara semu acak dengan pola yang sama dengan pemilihan *pixel* yang akan disisipkan pesan pada proses penyisipan. Proses ekstraksi pesan yang dapat dilihat pada Gambar II.3 adalah kebalikan dari urutan proses penyisipan pesan pada citra *GIF*. Pada awalnya cerita yang sudah disisipkan pesan akan dibaca dan kumpulan *pixel* akan dibaca dan dipilih secara semu acak, menggunakan pola semu acak yang sama dengan pola pada saat penyisipan pesan. Setelah *pixel* yang berisikan pesan diidentifikasi, maka pesan yang disisipkan dapat di ambil.



Gambar II.3 Proses Ekstraksi Pesan Menggunakan Metode Adaptif
Sumber : Andy Wicaksono ; ITB 2009

II.2.1. Pembangkit Bilangan Semu Acak

Dalam metode adaptif dibutuhkan tahap pembangkit bilangan semu acak untuk membantu memilih *pixel* mana yang akan disisipkan pesan. Bilangan acak yang dimaksud adalah bilangan yang tidak mudah di prediksi oleh pihak lain. Namun pada prakteknya sangat sulit mendapatkan bilangan acak, karena tidak ada prosedur komputasi yang dapat menghasilkan bilangan acak yang sempurna. Bilangan acak yang dihasilkan dengan rumus – rumus matematika adalah bilangan acak semu (*pseudo*), karena bilangan acak yang dibangkitkan dapat berulang secara periodik.

II.2.2. *Linear Congruential Generator (LCG)*

Linear congruential Generator (LCG) atau yang disebut juga dengan pembangkit bilangan acak kongruen – lanjar adalah salah satu pembangkit bilangan acak kongruen – lanjar adalah salah satu pembangkit bilangan acak tertua dan sangat terkenal. *LCG* dapat diterjemahkan kepada persamaan sebagai berikut :

$$x_{\mathrm{n}} = \left(a x_{\mathrm{n-1}} + b\right) \mod m$$

dimana

x_{n} : bilangan acak ke – n dari deretnya

$x_{\mathrm{n-1}}$: bilangan acak sebelumnya

a : faktor pengali

b : nilai *increment*

m : modulus (nilai a, b, dan m adalah konstanta)

Untuk memulai bilangan acak ini dibutuhkan sebuah bilangan x_0 yang di jadikan sebagai nilai awal (umpan). Bilangan acak pertama yang dihasilkan selanjutnya menjadi bibit pembangkitan bilangan acak selanjutnya. Jumlah bilangan acak yang tidak sama satu sama lain adalah sebanyak *m*. Semakin besar nilai *m*, maka semakin kecil kemungkinan dihasilkan nilai yang berulang.

II.3. Citra

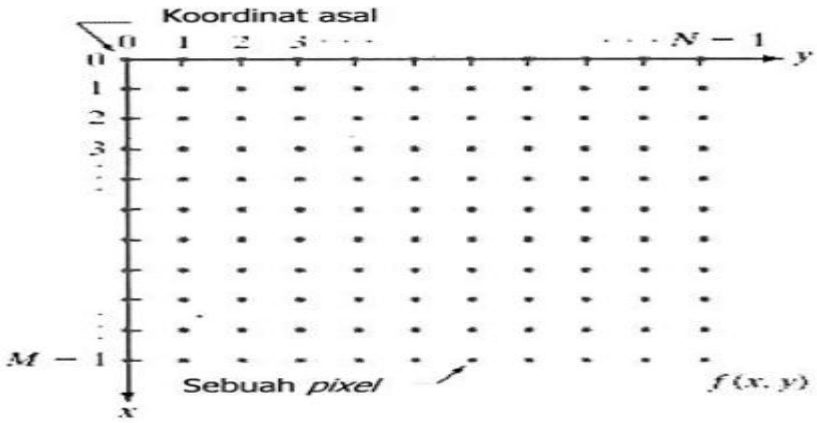
Citra adalah gambar pada bidang dwimatra (dua dimensi). Secara matematis, citra merupakan fungsi menerus dari intensitas cahaya pada bidang dwimatra. Terdapat dua jenis citra, yaitu :

1. Citra diam yaitu citra tunggal yang tidak bergerak. Contoh dari citra diam adalah foto.
2. Citra bergerak yaitu rangkaian citra diam yang ditampilkan secara beruntun sehingga memberi kesan pada mata sebagai gambar yang bergerak. Contoh dari citra bergerak adalah video.

II.3.1. Citra Digital

Secara umum, pengolahan citra digital menunjuk pada pemrosesan gambar 2 dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahan citra digital mengacu pada pemrosesan setiap data 2 dimensi. Citra digital merupakan sebuah larik (*array*) yang berisi nilai – nilai real maupun komplek yang direpresentasikan dengan deretan bit tertentu. Suatu citra dapat didefinisikan sebagai fungsi $f(x,y)$ berukuran M baris dan N kolom, dengan x dan y adalah

koordinat spasial, dan amplitudo *f* di titik koordinat (*x,y*) dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai x,y, dan nilai amplitudo f secara keseluruhan berhingga (*finite*) dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital. Gambar II.4 menunjukkan posisi koordinat citra digital.

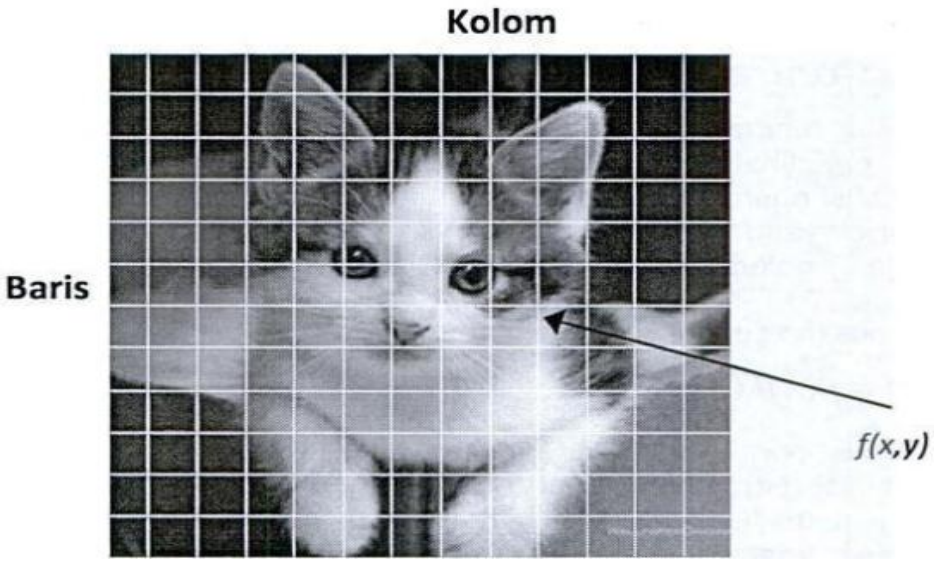


Gambar II.4 Koordinat Citra Digital
Sumber : Darma Putra : 2010 ; 20

Citra digital dapat ditulis dalam bentuk matrik sebagai berikut.

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad \text{..... (2.1)}$$

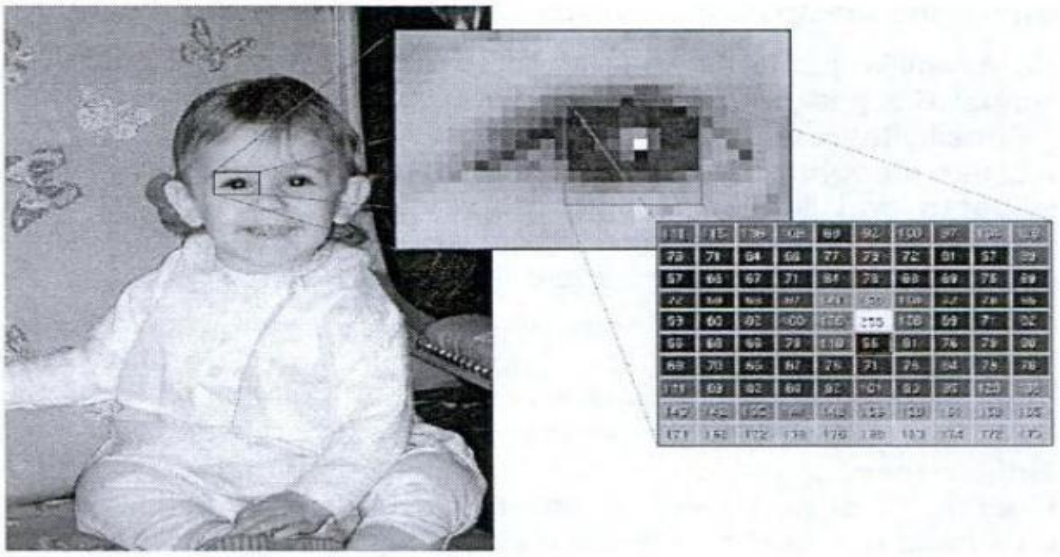
Nilai pada suatu irisan antara baris dan kolom (pada posisi x,y) disebut dengan picture elements, image elements, pels, atau pixels. Istilah terakhir (pixel) paling sering digunakan pada citra digital. Gambar II.5 menunjukkan ilustrasi digitalisasi citra dengan M = 16 baris dan N = 16 kolom. (Darma Putra ; 2010 : 20)



Gambar II.5 Ilustrasi Digitalisasi Citra

Sumber : Darma Putra : 2010 ; 21

Gambar II.6 menyajikan contoh lain dari suatu citra digital (citra *grayscale*), dengan nilai intensitas dari citra pada area tertentu.



Gambar II.6 Contoh Citra *Grayscale*, Cuplikan (*Croping*) pada Area Tertentu

Sumber : Darma Putra : 2010 ; 21

II.4. Penyembunyian Data

Teknik penyisipan data ke dalam *covertext* dapat dilakukan dalam dua macam ranah :

1. Ranah spasial (*spatial/time domain*)

   Teknik ini memodifikasi langsung nilai *byte* dari *covertext* (nilai *byte* dapat merepresentasikan intesitas/warna *pixel* atau amplitudo). Contoh metode yang tergolong kedalam teknik ranah spasial adalah metode *LSB.*

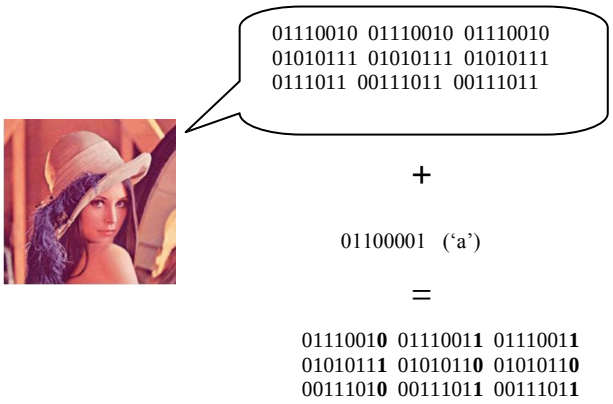
2. Ranah transform (*transform domain*)

   Teknik ini memodifikasi langsung hasil transformasi frekuensi sinyal. Contoh metode yang tergolong ke dalam teknik ranah frekuensi adalah *spread spectrum*.

II.5. Penyisipan dan Penyembunyian Pesan Citra Digital

II.5.1. Penyembunyian dan Penyisipan Metode LSB

Misalkan ada sebuah gambar dengan nilai *pixel*nya sebagai berikut :



Gambar II.7 Penyisipan Pesan Menggunakan Metode *LSB*
Sumber : Andy Wicaksono ; ITB 2009

Pada Gambar II.7 yang merupakan contoh *LSB insertion* pada citra *bitmap* 24 - *bit*, diambil sebuah daerah secara acak yang terdiri dari *pixel*, dimana tiap *pixel* memiliki 24 – *bit* data yang terbagi menjadi 8 – *bit* warnah merah, 8 – *bit* warna hijau, dan 8 – *bit* warna biru. Pesan yang akan disisipkan adalah karakter a, yang memiliki bilangan *ASCII* 97, atau 01100001 dalam format *biner*. *Bit* – *bit* pesan akan disisipkan pada tiga *pixel* yang diambil. *Bit* yang disisipkan pesan adalah *bit* yang bergaris bawah, dan *bit* yang diubah ditunjukkan dengan cara penulisan angka tebal.

Efek dari metode penyisipan ini adalah berubahnya nilai warna tertentu pada citra sebesar 1 – *bit*, namun karena perubahan nilai warna terjadi pada *LSB*, sehingga perubahan warna tidak cukup signifikan dan secara kasat mata media digital yang disisipkan tidak ditangkap mempunyai perubahan. Metode *LSB* adalah metode yang paling banyak digunakan, dikarenakan kemudahan pengimplementasiannya dan efek perubahan kualitas media penyisipan yang telah

disisipkan *bit – bit* pesan tidak signifikan. Akan tetapi metode ini mempunyai kekurangan, yaitu media digital yang telah disisipkan pesan tidak tahan akan dengan proses pemanipulasian yang akan mengubah struktur bit *pixel* media. Contoh pengubahan format citra dan penggantian ukuran (*resize*). Proses manipulasi citra ini akan merusak *bit – bit* pesan yang telah disisipkan, sehingga pesan yang telah disisipkan tidak dapat diekstraksi.

II.5.2. Ekstraksi Pesan

Ekstraksi pesan yaitu proses mengembalikan pesan yang telah disisipkan kedalam media citra. Proses ekstraksi merupakan kebalikan dari proses penyisipan pesan, hanya saja proses kerjanya hampir sama, yaitu merubah gambar *stego object* ke bilangan biner, dan setelah itu akan langsung terlihat kode biner dari pesan rahasia tersebut, karena nilai biner pada bit terakhir gambar telah di ubah menjadi nilai biner pesan rahasia.

(01110010 01110010 01110010)

(01010111 01010111 01010111)

(00111011 00111011 00111011) Segmen citra sebelum disisipkan.

(0111001**0** 0111001**1** 0111001**1**)

(0101011**0** 0101011**0** 0101011**0**)

(0011101**0** 0011101**0** 0011101**1**) Segmen citra sesudah disisipkan *a.*

Pada segmen citra setelah disisikan jelas terlihat bahwa bit terahir dari setiap segmen telah berubah, dan perubahan tersebut adalah proses penyisipan pesan. Dalam pengembalian pesan pada *stego object* nilai biner terahir di ambil lalu dikonversi kembali ke karakter dari kode biner tersebut. Bit-bit terahir dari

setiap segmen citra adalah 011000001, dimana 011000001 adalah kode biner untuk 97 yang merupakan kode ASCII karakter *a* Dalam proses ini pesan telah berhasil dikembalikan.

II.5.3. Penyembunyian dan Penyisipan Pesan Metode Adaptif

Pada proses penyisipan pesan, pesan yang akan disisipkan kedalam citra *GIF* dalam beberapa tahap. Pada awalnya citra *GIF* didekompresi dan diubah data rasternya menjadi kode *bitmap.* Dibawah ini adalah ilustrasi penyisipan pesan pada citra *GIF*.

Pesan yang disisipkan : 1010

Citra yang disisipkan pesan :

144	44	198	163	54	198	163	147
188	205	202	188	211	192	157	211
218	163	64	218	173	75	218	182
192	116	218	192	126	218	192	157
188	218	221	198	225	163	54	225
225	221	198	231	163	33	238	173
231	208	238	163	33	238	173	238
64	238	192	75	238	202	75	238

Sumber : Prasetyo Andy Wicaksono ; ITB 2009

Proses penyisipan pesan dilakukan dalam beberapa tahap, yaitu :

1. Pemilihan *pixel* yang akan disisipkan secara semu acak.

   Dari sekumpulan *pixel* yang telah diidentifikasikan pada tahap sebelumnya, akan dilakukan pemilihan *pixel* yang akan disisipkan pesan secara semu acak. Pengguna akan memasukkan *seed* (*m*) yang akan menghasilkan rangkaian angka semu acak tertentu. Nilai *m* ditentukan minimum sebesar ukuran minimum sebesar ukuran dari

gambar yang akan disisipkan, agar fungsi pencarian angka semu acak tidak menghasilkan angka yang berulang. Setelah itu berdasarkan angka semu acak yang dibangkitkan, akan dipilih *pixel* yang akan disisipkan pesan. Pertama – tama setiap *pixel* akan diberikan indeks penomoran dengan terurut dari kiri ke kanan, lalu atas ke bawah.

144_0	44_1	198_2	163_3	54_4	198_5	163_6	147_7
188_8	205_9	202_{10}	188_{11}	211_{12}	192_{13}	157_{14}	211_{15}
218_{16}	163_{17}	64_{18}	218_{19}	173_{20}	75_{21}	218_{22}	182_{23}
192_{24}	116_{25}	218_{26}	192_{27}	126_{28}	218_{29}	192_{30}	157_{31}
188_{32}	218_{33}	221_{34}	198_{35}	225_{36}	163_{37}	54_{38}	225_{39}
225_{40}	221_{41}	198_{42}	231_{43}	163_{44}	33_{45}	238_{46}	173_{47}
231_{48}	208_{49}	238_{50}	163_{51}	33_{52}	238_{53}	173_{54}	238_{55}
64_{56}	238_{57}	192_{58}	75_{59}	238_{60}	202_{61}	75_{62}	238_{63}

Sumber : Prasetyo Andy Wicaksono ; ITB 2009

Setelah setiap *pixel* diberi nomor, angka semu acak dihitung dengan *m* minimal ukuran gambar. Pada contoh ini diambil nilai $m = 64$, $a = 5$, $x_0 = 1$ dan $b = 1$. Jika dimasukkan kedalam persamaan LCG, maka akan didapat angka semu acak yang akan digambarkan pada proses perhitungan dibawah ini :

$X_i = (5x_{i-1} + 1)$ *mod* 64

$X_1 = 6$ *mod* $64 = 6$

$X_2 = 31$ *mod* $64 = 31$

$X_3 = 156$ *mod* $64 = 28$

$X_4 = 141$ *mod* $64 = 13$

$X_5 = 66$ *mod* $64 = 2$

X_6 = 11 *mod* 64 = 11

X_7 = 56 *mod* 64 = 56

X_8 = 281 *mod* 64 = 25

Setelah perhitungan dilakukan didapat himpunan angka semu acak untuk memilih *pixel* mana yang akan menjadi kandidat *pixel* yang akan disisipkan pesan.

2. Pencarian nilai paritas dari warnet palet

Warna palet akan ditentukan nilai paritasnya menggunakan persamaan (R + G + B) *mod* 2. Diambil sebuah contoh warna palet 611c0d, 6f2f17, 752f0d, 752f17, 7c7c74, daa336, b87c2c dan fcb363. Setiap warna palet dicari nilai paritasnya dengan memasukan nilai warna ke dalam persamaan :

611c0d → R : 97, G : 28, B : 13 → (97 + 28 + 13) *mod* 2 = 0

6f2f17 → R : 111, G : 47, B : 23 → (111 + 47 + 23) *mod* 2 = 1

752fod → R : 117, G : 47, B : 13 → (117 + 47 + 13) *mod* 2 = 1

752f17 → R : 117, G : 47, B : 23 → (117 + 47 + 23) *mod* 2 = 1

7c7c74 → R : 117, G : 124, B : 116 → (124 + 124 + 116) *mod* 2 = 0

daa336 → R : 218, G : 163, B : 64 → (218 + 163 + 64) *mod* 2 = 1

b87c2c → R : 184, G : 124, B : 44 → (184 + 124 + 44) *mod* 2 = 0

fcb636 → R : 252, G : 182, B : 54 → (252 + 182 + 54) *mod* 2 = 0

3. Pengecekan paritas dan penyisipan pesan

Diadakan pengecekan dan membandingkan nilai paritas dengan bit pesan pada setiap *pixel*. Jika dihasilkan persamaan *bit*, maka *pixel* tersebut tidak dimodifikasi dan pengecekan berlanjut pada *pixel* selanjutnya. Jika terjadi perbedaan *bit*, maka *pixel* warna tersebut dimodifikasi dengan cara mencari warna tetangga terdekat pada palet warnanya yang mempunyai paritas yang sama pada citra. Cara pencarian warna tetangga terdekat dengan menggunakan persamaan jarak :

$$d = \sqrt{(R_2 - R_1)^2 + (G_2 - G_1)^2 + (B_2 - B_1)^2}.$$

Dari ilustrasi citra *GIF* yang sudah diberi nomor, didapat bahwa *pixel* nomor 6, 31, 28, 13, 2, 11, 56, 25 yang akan dicek paritasnya dan disisipkan pesan di dalamnya. Sedangkan pesan yang akan disisipkan bernilai 1010. Proses pengecekan paritas dan penyisipan pesan dijelaskan lebih lanjut pada tabel II.1.

Tabel II.1 Proses Pengecekan Paritas dan Penyisipan Pesan

Nomor *pixel*	**Indeks warna**	**Warna palet**	**Paritas**	**Pesan yang akan disisipkan**	**Keterangan**
6	163	611c0d	0	1	Bisa disisipkan, *pixel* dimodifikasi
31	157	6f2f17	1	0	Bisa disisipkan, *pixel* dimodifikasi
28	126	752f0d	1	1	Bisa disisipkan, tanpa memodifikasi *pixel*
13	192	752f17	1	0	Bisa disisipkan, *pixel* di modifikasi

2	198	7c7c74	0	-	
11	188	Daa336	1	-	

Sumber : Prasetyo Andy Wicaksono ITB 2009

II.5.4. Ekstraksi Pesan

Proses ekstraksi pesan adalah proses pengambilan pesan dari berkas citra yang telah disisipkan pesan. Pada proses ekstraksi pesan, pesan diambil dari *pixel* dengan cara menggunakan angka semu acak yang telah ditentukan sebelumnya saat penyisipan pesan. Berikut adalah rincian proses ekstraksi pesan.

1. Penghitungan angka semu acak

   Angka semu acak dihitung menggunakan nilai *seed* yang dipakai saat proses penyisipan pesan, pada ilustrasi ini digunakan nilai m = 64. Nilai ini akan dimasukkan kedalam persamaan LCG dimana a = 5, x_0 = 1, dan b = 1, maka akan didapat angka semu acak yang akan digambar pada proses berikut.

   $X_i = (5x_{i-1} + 1)\ mod\ 64$

   $X_1 = 6\ mod\ 64 = 6$

   $X_2 = 31\ \ mod\ 64 = 31$

   $X_3 = 156\ mod\ 64 = 28$

   $X_4 = 141\ mod\ 64 = 13$

   $X_5 = 66\ \ mod\ 64 = 2$

   $X_6 = 11\ mod\ 64 = 11$

$X_7 = 56 \ mod \ 64 = 56$

$X_8 = 281 \ mod \ 64 = 25$ ....dst

Dari perhitungan ini didapat himpunan angka semu acak A = {6,31,28,13,2,11,56,25,....}. Angka semu acak yang dihasilkan dari perhitungan menentukan *pixel* mana yang akan disisipkan pesan.

2. Pengambilan pesan

Pixel yang sudah diidentifikasi sebagai *pixel* yang disisipkan pesan lalu dicek nilai paritas dari warna palet yang ditunjukan pada *pixel* tersebut. Nilai paritas warna palet dari *pixel* tersebut menjadi bit pesan yang disisipkan. Proses pengambilan pesan dapat dilihat pada tabel II.2.

Tabel II.2 Proses Pengambilan Pesan dari *Pixel*

Nomor *pixel*	**Indeks warna**	**Paritas**	**Pesan yang disisipkan**
6	163	1	1
31	157	0	0
28	126	1	1
13	192	0	0

Sumber : Prasetyo Andy Wicaksono ; ITB 2009

Setelah proses pengambilan pesan dilakukan, dihasilkan pesan yang disisipkan pada citra tersebut, yaitu 1010.

II.6. STEGANOGRAFI

Steganografi berasal dari Bahasa Yunani, yaitu *steganos* yang artinya tulisan tersembunyi. Secara umum, steganografi adalah ilmu dan seni

menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui. Steganografi dapat dianggap sebagai kelanjutan dari kriptografi dan mempunyai hubungan yang erat, namun eksistensi dari pesan tersebut masih ada. Sedangkan steganografi merahasia isi pesan dengan cara menutupi keberadaan pesan. Pesan yang diacak namun masih terlihat eksistensinya pada kriptografi akan menimbulkan kecurigaan, dan hal ini tidak akan terjadi pada steganografi.

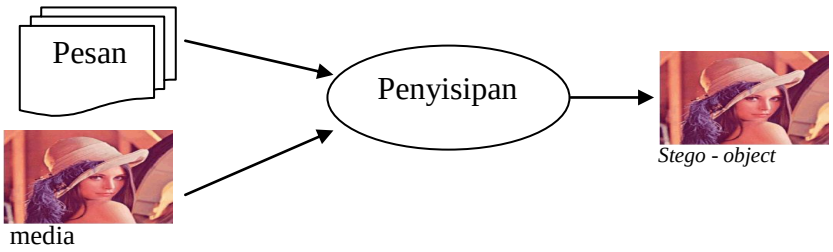
II.6.1. Sejarah Steganografi

Menurut catatan sejarah, teknik steganografi sudah digunakan sejak tahun 440 SM. Salah satu contoh steganografi pada masa lalu yaitu pada saat Demaratus, seorang prajurit Yunani, hendak menyerang Yunani, hendak mengirimkan pesan ke Sparta yang berisi peringatan bahwa Xerxes, raja Persia, hendak meyerang Yunani. Teknik yang digunakan adalah dengan menggunakan meja yang telah diukir kemudian diberi lapisan lilin untuk menutupi pesan tersebut, dengan begitu pesan dalam meja dapat disampaikan tanpa menimbulkan kecurigaan oleh para penjaga. Contoh lainnya adalah saat pengiriman pesan dilakukan dengan cara mencukur habis kepala seorang budak, lalu menuliskan pesan tersebut pada kulit kepalanya. Kemudian budak tersebut dikirim setelah pesan tertutupi oleh rambut yang tumbuh. Pesan dapat dibaca dengan mencukur kembali rambut mereka.

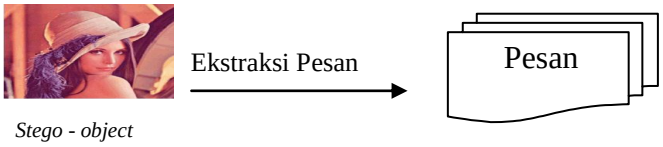
II.6.2. Terminologi Steganografi

Pada masa kini, steganografi banyak dilakukan pada data digital dengan menggunakan media digital, seperti teks, citra, audio, dan video. Steganografi digital ditekankan pada kinerja dan teknik penyisipan pesan agar sedapat mungkin pesan yang disisipkan kedalam media digital tidak mengubah kualitas media digital tersebut.

Proses steganografi mirip dengan *digital watermarking*, namun keduanya memiliki perbedaan. Pada steganografi pesan rahasia disembunyikan di dalam media penampung dimana media penampung tersebut tidak berarti apa – apa (hanya menjadi pembawa), sedangkan pada *watermarking* media penampung tersebut dilindungi kepemilikannya dengan pemberian label hak cipta (*watermark*). Selain itu, jika pada steganografi kekokohan data tidak terlalu penting, maka pada watermarking kekokohan *watermark* merupakan properti utama sebab *watermark* tidak boleh rusak atau hilang meskipun media penampung dimanipulasi.



Gambar II.8 Proses Penyisipan Pesan
Sumber : Prasetyo Andy Wicaksono ; ITB 2009



Gambar II.9 Proses Ekstraksi Pesan
Sumber : Prasetyo Andy Wicaksono ; ITB 2009

Steganografi mencakup dua buah proses, yaitu penyisipan, contohnya dapat dilihat pada Gambar II.8 dan ekstraksi pesan, contohnya dapat dilihat pada Gambar II.9. Proses penyisipan pesan membutuhkan dua buah masukan, yaitu pesan yang akan disembunyikan dan media penyisipan. Proses penyisipan pesan akan menghasilkan *stego – object*, yaitu suatu media digital yang telah disisipkan pesan tersembunyi, namun kualitas *stego – object* mirip dengan media penyisipan yang menjadi masukan proses ini.

II.7. Microsoft Visual Studi 2010

Visual Studio merupakan sebuah lingkungan kerja (IDE – *Integrated Development Environtment*) yang digunakan untuk pemrograman .Net yang dapat digunakan untuk beberapa bahasa pemrograman, seperti Visual Basic (VB), C# (baca C Sharp), Visual C++, J# (baca J Sharp), F# (baca F Sharp), dan lain-lain. Bahasa pemrograman Visual Basic Merupakan salah satu bahasa yang sangat populer hingga kini dan merupakan salah satu solusi untuk menciptakan aplikasi pada sistem operasi *windows*, baik *windows* 7, *windows server* 2008, dan *windows mobile* 6.1. hal ini dikarenakan kemudahan yang diberikan visual basic dan IDE visual studio yang digunakan untuk menciptakan sebuah aplikasi.

Visual Basic 2010 adalah inkarnasi dari bahasa Visual Basic yang sangat popoler dan telah dilengkapi dengan fitur serta fungsi yang setara dengan bahasa tingkat tinggi lainnya seperti C++. Anda dapat menggunakan Visual Basic 2010 untuk membuat aplikasi Wondows, mobile, Web, dan Office yang kompleks dengan menggunakan kode yang Anda tulis, atau kode yang telah ditulis oleh orang lain dan kemudian dimasukkan kedalam program Anda. (Christopher Lee : 1).