KOMPARASI ALGORITMA *LINEAR CONGRUENTIAL GENERATOR* DAN BLUM BLUM SHUB PADA IMPLEMENTASI *FRAGILE WATERMARKING* UNTUK VERIFIKASI CITRA DIGITAL

Tria Aprilianto[1], Yuliana Melita[2]
[1].. STMIK ASIA Malang, [2].. iSTTS
e-mail : raptorapril@gmail.com, ymp@stts.edu

ABSTRAKSI

Fragile watermarking merupakan salah satu aplikasi steganografi yang dapat menjadi solusi kebutuhan verifikasi citra digital. Metode yang digunakan untuk teknik watermarking ini adalah metode *Least Significant Bit* (LSB). Metode LSB ini mengganti bit-bit yang tergolong bit LSB pada setiap *byte* pada suatu piksel citra dengan bit-bit *watermark* yang akan disisipkan.

Untuk memperkuat teknik penyembunyian data, bit-bit *watermark* tidak digunakan mengganti bit-bit dari piksel awal sampai piksel terakhir secara berurutan, namun dipilih susunan piksel secara acak.. Hal ini dapat dilakukan dengan memanfaatkan algoritma *Linear Congruential Generator* (LCG) dan Blum Blum Shub (BBS) sebagai pembangkit bilangan acak semu (*Pseudo Random Number Generator* / PRNG). Bilangan acak yang dihasilkan akan digunakan sebagai posisi piksel sebagai tempat penyisipan watermark.

Penelitian ini dilakukan dengan membuat aplikasi *fragile watermarking* dan kemudian membandingkan histogram, nilai *Mean Squared Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR) dari setiap hasil penyisipan *watermark* (*embedding*) yang dilakukan. Aplikasi dibuat dengan menggunakan bahasa pemrograman Visual Basic 6.0.

Kata Kunci : *Steganografi, Fragile watermarkin,* watermarking, *Linear Congruential Generator*, Blum Blum Shub

ABSTRACT

Fragile watermarking is one of the steganography application that can be a solution to the needs of the digital image verification. The method used for the watermarking technique is the method of *Least Significant Bit* (LSB). The method replaces the LSB bits belonging LSB bits in each byte at a pixel image with watermark bits to be inserted.

To strengthen data hiding techniques, watermark bits are not used to change the bits from the beginning to the pixel last pixel in a sequence, but the arrangement of pixels selected at random. This can be done by using algorithms *Linear Congruential Generator* (LCG) and the *Blum Blum Shub* (BBS) as a pseudorandom number generator (*Pseudo Random Number Generator* / PRNG). Random number generated will be used as the position of pixels as the watermark insertion.

This research was done by creating fragile watermarking applications and then compare the histogram, the value of *Mean Squared Error* (MSE) and *Peak Signal to Noise Ratio* (PSNR) of each outcome watermark insertion (*embedding*) is done. Applications built using the Visual Basic 6.0 programming language.

Keywords : Steganography, Fragile watermarkin, watermarking, Linear Congruential Generator, Blum Blum Shub

PENDAHULUAN

Perkembangan teknologi serta internet yang begitu pesat dari tahun ketahun, member yang membuat data digital banyak digunakan, antara lain: mudah diduplikasi, mudah disimpan serta mudah didistribusikan. Kemudahan tersebut salah satunya merupakan pendorong adanya dampak negative perkembangan teknologi.

Data digital bias saja disalahgunakan oleh orang-orang yang tidak bertanggung jawab untuk tujuan yang komersil atau sekedar iseng. Untuk citra digital, pengguna seringkali melakukan manipulasi untuk mendapatkan tampilan citra digital baru sesuai dengan yang pengguna tersebut inginkan. Terkait dengan hal ini, banyak pemilik citra digital tidak ingin citra digital miliknya dapat berubah atau diubah, atau paling tidak mereka dapat mengetahui jika citra miliknya telah berubah atau termanipulasi, sehingga mereka bias menentukan apakah citra tersebut layak pakai atau tidak. Kebutuhan seperti ini disebut dengan kebutuhan verifikasi citra.

Watermarking dapat menjadi solusi untuk menyelesaikan permasalahan tersebut. *Watermarking* adalah teknik menyisipkan suatu informasi kedalam data multimedia. Informasi tersebut dapat berupa data teks, citra, audio, ataupun video yang menggambarkan kepemilikan suatu pihak.Informasi yang disisipkan tersebut dinamakan *watermark*. Penyisipan *watermark* dilakukan sedemikian rupa sehingga *watermark* tidak merusak data digital yang disisipi. Selain itu *watermark* yang telah disisipkan tidak dapat dipersepsi oleh indera manusia.

Watermarking merupakan cabang dari steganografi, yang membedakannya yaitu apabila pada *watermarking*, media (*cover*) menjadi focus utama, sedangkan pada steganografi yang menjadi focus adalah data yang disisipkan. Untuk kebutuhan verifikasi citra, digunakan *watermark* yang bersifat *fragile* (*fragile watermark*), yaitu *watermark* yang rentan terhadap perubahan/ manipulasi. Sehingga ketika suatu citra yang sudah disisipi *fragile watermark* dimanipulasi kemudian diekstrak, akan menyebabkan *watermark* tidakbisa di ekstraksi atau hasil ekstraksi tidak sama dengan *watermark* asli.

Salah satu metode *watermarking* yang cocok untuk aplikasi *fragile watermarking* adalah metode *Least Significant Bit* (LSB). Penyisipan *watermark* dilakukan dengan mengganti bit-bit LSB dari suatu piksel citra dengan bit-bit data *watermark*. Untuk memperkuat teknik penyembunyian data, bit-bit *watermark* tidak digunakan mengganti bit-bit dari piksel awal sampai piksel terakhir secara berurutan, namun dipilih susunan piksel secara acak. Piksel- piksel acak tersebut dapat dihasilkan dengan memanfaatkan algoritma *Linear Congruential Generator* (LCG) dan Blum Blum Shub (BBS) yang efektif dan sederhana secara kompleksitas teoritis sebagai pembangkit bilangan acak semu (*Pseudo Random Number Generator/* PRNG*)*(Munir, 2006).

Untuk itu pada Penelitian ini akan dibahas mengenai komparasi algoritma LCG dan BBS yang digunakan sebagai PRNG pada implementasi *fragile watermarking* untuk verifikasi citra digital. Dari hasil komparasi tersebut nantinya akan didapatkan algoritma PRNG mana yang lebih baik untuk diterapkan pada permasalahan ini

METODOLOGI PENELITIAN

Tahapan-tahapan yang dilakukan dalam penelitian ini antara lain :

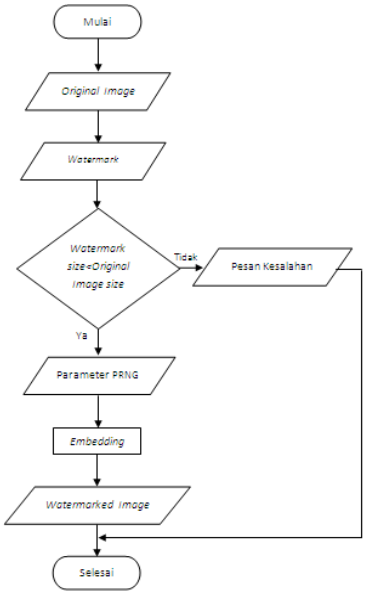
A. Analisa Sistem

Penelitian ini berupaya untuk membandingkan algoritma *Linear Congruential Generator* (LCG) dan *Blum Blum Shub* (BBS) yang digunakan

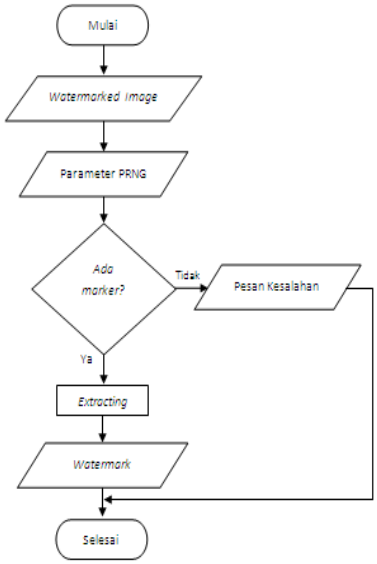
sebagai *Pseudo Random Number Generator* (PRNG) pada implementasi *fragile watermarking* untuk verifikasi citra digital. Dari hasil perbandingan tersebut nantinya akan didapatkan algoritma PRNG mana yang lebih baik untuk diterapkan pada permasalahan tersebut. Metode *watermarking* yang dipakai adalah metode *Least Significant Bit* (LSB) yang tidak kokoh terhadap perubahan yang dilakukan pada citra ber-*watermark* sehingga cocok untuk aplikasi *fragile watermarking*.

Ada 2 alur program pada aplikasi ini, yaitu alur program *embedding* dan alur program *extracting*.

- Alur Program *Embedding*
  Proses *embedding* citra dilakukan pada gambar 2.1
- Alur Program *Extracting*
  Proses *Extracting* citra dilakukan pada gambar 2.2



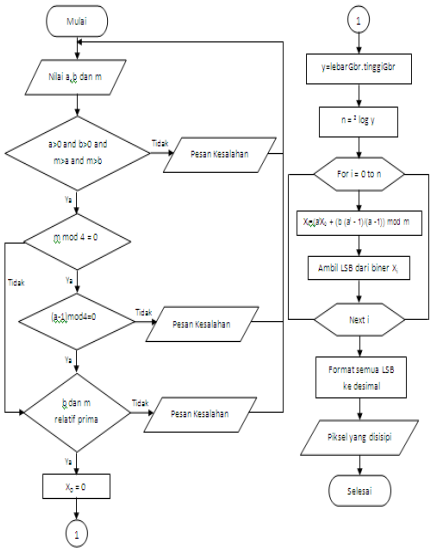
Gambar 2.1 Proses *embedding*



Gambar 2.2 Proses *Extracting*

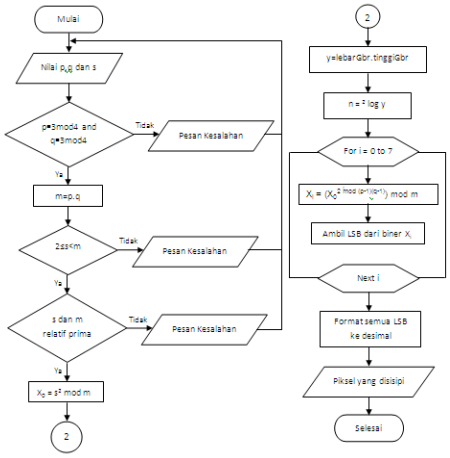
B. **Penerapan Algoritma dalam Permasalahan**

- *Flowchart* Algoritma *Linear Congruential Generator*



Gambar 2.3 Algoritma *Linear Congruential Generator*

- *Flowchart* Algoritma *Blum Blum Shub*



Gambar 2.3 Algoritma *Blum Blum Shub*

LANDASAN TEORI

1. Citra Digital

Citra merupakan representasi dua dimensi dari bentuk fisik nyata 3 dimensi. Mulai dari gambar hitam putih pada sebuah foto yang tidak bergerak sampai pada gambar berwarna yang bergerak pada sebuah pesawat televisi. Citra yang diolah menggunakan komputer digital yang mentransformasikan citra kedalam bentuk *array* numerik disebut citra digital. *Array* numerik tersebut merepresentasikan intensitas warna pada titik-titik elemen citra. Elemen-elemen citra digital apabila ditampilkan dalam layar monitor akan menempati sebuah ruang yang disebut dengan piksel (*pixel / picture elemen*).

2. Pengolahan Citra

Pengolahan citra (*image processing*) merupakan suatu sistem dimana proses dilakukan dengan masukan berupa citra (*image*) dan hasilnya juga berupa citra (*image)*. Citra digital diasumsikan dengan persamaan f(x,y) dimana x menyatakan nomor baris, y menyatakan nomor kolom, dan f menyatakan nilai derajat keabuan dari citra. Sehingga (x,y) adalah posisi dari piksel dan f adalah nilai derajat keabuan pada titik (x,y). Kecerahan setiap citra disimpan dengan cara pemberian nomor pada setiap piksel. Makin tinggi nomor piksel maka makin gelap (hitam) piksel tersebut. Begitu juga sebaliknya makin rendah nilai piksel tersebut maka makin terang. Sistem yang umum memiliki 256 tingkat kecerahan untuk setiap piksel, yang paling terang adalah 0 dan yang paling gelap adalah 255. Citra atau gambar terbagi dalam tiga tipe adalah sebagai berikut:

1) Citra biner, yaitu citra yang hanya memiliki dua nilai yaitu 1 dan 0. Dinyatakan dalam fungsi :
   f(x,y) Σ {0,1} (Persamaan 2.1 Citra biner)
2) Citra *grey-scale*, yaitu citra yang terdiri dari satu *layer* warna dengan derajat keabuan tertentu. Dinyatakan dalam fungsi :
   f(x,y) Σ [0…255] (Persamaan 2.2 Citra *grey-scale*)
3) Citra berwarna, yaitu citra yang terdiri dari tiga *layer* warna yaitu RGB (*Red-Green-Blue*) dimana R-*layer* adalah matrik yang menyatakan derajat kecerahan untuk warna merah, G-*layer* adalah matrik yang menyatakan derajat kecerahan untuk warna hijau, dan B-*layer* adalah matrik yang menyatakan derajat kecerahan untuk warna biru. Representasi dalam citra digital dinyatakan dalam persamaan :
   fR(x,y) Σ [0…255]
   fG(x,y) Σ [0…255] (Persamaan 2.3 Citra berwarna)
   fB(x,y) Σ [0…255]

3. Steganografi

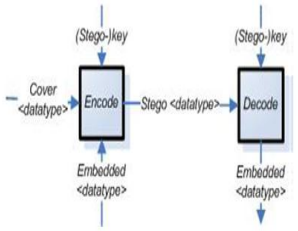
Steganografi (*steganography*) adalah ilmu dan seni menyembunyikan pesan rahasia (*hiding message*) sedemikian sehingga keberadaan (eksistensi) pesan tidak terdeteksi oleh indera manusia (Munir, 2004). Kata steganografi berasal dari Bahasa Yunani

yang berarti "tulisan tersembunyi" (*covered writing*).

Steganografi membutuhkan dua properti: wadah penampung dan data rahasia yang akan disembunyikan. Steganografi digital menggunakan media digital sebagai wadah penampung, misalnya citra, suara, teks, dan video. Data rahasia yang disembunyikan juga dapat berupa citra, suara, teks, atau video.

Penyembunyian data rahasia ke dalam media digital mengubah kualitas media tersebut. Kriteria yang harus diperhatikan dalam penyembunyian data diantaranya adalah (Munir, 2004):

1) *Imperceptibility.* Keberadaan pesan tidak diketahui secara langsung oleh penglihatan manusia.
2) *Fidelity*. Mutu citra penampung tidak jauh berubah.
3) *Recovery*. Pesan yang disembunyikan harus dapat diungkapkan kembali (*recovery*).



Gambar 3.1 *Diagram proses penyisipan dan ekstraksi*

1. *Embedded*: pesan yang disembunyikan
2. *Cover*: media yang digunakan untuk menyembunyikan pesan.
3. *Encode*: proses penyisipan *cover-object* ke dalam *stego-object*.
4. *Stego*: pesan yang sudah berisi pesan tersembunyi.
5. *Decode*: proses pengembalian *cover* dari *stego*.
6. *Stego-key*: kunci yang digunakan untuk menyisipan pesan dan mengekstraksi pesan dari *stegotext* (Munir, 2006).

4. Watermarking

Watermarking adalah teknik menyisipkan suatu informasi ke dalam suatu media penampung yang berupa data multimedia. Pada prinsipnya *watermarking* sama dengan steganografi karena *watermarking* adalah cabang dari steganografi sehingga untuk penyisipan *watermark* ke dalam media penampung sama halnya dengan konsep steganografi. Perbedaan *watermarking* dengan steganografi terletak pada media penampungnya. Jika pada steganografi informasi yang disisipkakan dalam media penampung, dimana media penampung tersebut tidak berarti apa-apa (hanya sebagai pembawa), maka pada *watermarking* media penampung tersebut yang dilindungi dari penyalahgunaan dengan pemberian informasi tersebut (*watermark)*.

5. Metode *Least Significant Bit*

Salah satu teknik *watermarking* yang dapat diterapkan pada citra digital adalah metode *Least Significant Bit.* Penyembunyian data dilakukan dengan mengganti bit-bit data di dalam segmen citra dengan bit-bit data *watermark*

6. *Pseudo Random Number Generator* (PRNG)

Pseudorandom Number Generator (PNRG) atau pembangkit bilangan acak semu adalah sebuah algoritma yang membangkitkan sebuah deret bilangan yang tidak benar-benar acak. Keluaran dari pembangkit bilangan acak semu ini hanya mendekati beberapa sifat yang dimiliki bilangan acak (Thung, 2008).

7. *Linear Congruential Generator*

Linear Congruential Generator (LCG) adalah PRNG yang berbentuk:

$$X_{n+1} = (aX_n + b) \bmod m$$

di mana :
X_{n+1}= bilangan acak ke-(n+1) dari deretnya
Xn = bilangan acak sebelumnya
a = faktor pengali
b = increment
m = modulus X_0 merupakan kunci pembangkit atau sering juga disebut umpan (*seed*).

8. *Blum Blum Shub*

Blum Blum Shub merupakan algoritma pembangkit bilangan yang cukup sederhana dan efektif. *Blum Blum Shub* diperkenalkan pada tahun 1986 oleh Lenore Blum, Manuel Blum, dan Michael Shub (Munir 2006). *Blum Blum Shub* berbentuk sebagai berikut:

$$X_{n+1} = (X_n)^2 \bmod M$$

di mana :
X_{n+1} = bilangan acak ke-n+1 dari deretnya
X_n = bilangan acak sebelumnya
M = modulus, di mana M merupakan hasil kali dari dua

PERANCANGAN DAN IMPLEMENTASI

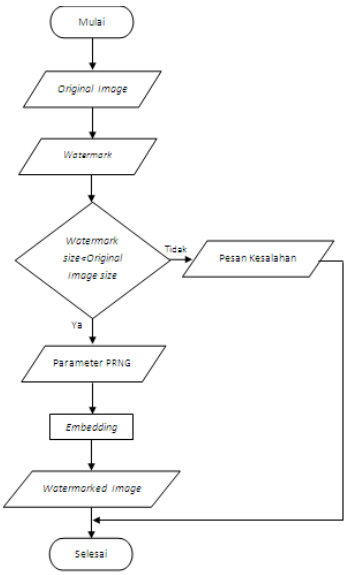
1. Analisa Sistem

Tugas akhir ini berupaya untuk membandingkan algoritma *Linear Congruential Generator* (LCG) dan *Blum Blum Shub* (BBS) yang digunakan sebagai *Pseudo Random Number Generator* (PRNG) pada implementasi *fragile watermarking* untuk verifikasi citra digital. Dari hasil perbandingan tersebut nantinya akan didapatkan algoritma PRNG mana yang lebih baik untuk diterapkan pada permasalahan tersebut. Metode *watermarking* yang dipakai adalah metode *Least Significant Bit* (LSB) yang tidak kokoh terhadap perubahan yang dilakukan pada citra ber-*watermark* sehingga cocok untuk aplikasi *fragile watermarking*.

Ada 2 alur program pada aplikasi ini, yaitu alur program *embedding* dan alur program *extracting*.

A Alur Program *Embedding*

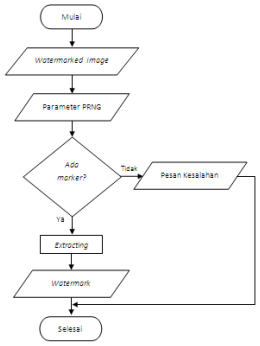
Inputkan citra yang akan diberi *watermark* (*original image*) dan *watermark* yang akan disisipkan ke citra tersebut terlebih dahulu. Setelah itu *watermark* akan dicek apakah ukurannya lebih kecil dari *original image* jika iya maka *user* akan diminta untuk memasukkan parameter-parameter yang diperlukan dalam pembangkitan bilangan acak, sedangkan jika tidak maka pesan kesalahan akan ditampilkan. Setelah semua parameter diisi, proses *embedding* citra dilakukan. Hasil dari proses tersebut adalah file citra yang mempunyai *watermark* di dalamnya (*watermarked image*).



Gambar 4.1*Diagram alur embedding*

B. Alur Program *Extracting*

Inputkan *watermarked image* yang telah di *embed* sebelumnya menggunakan aplikasi ini. Kemudian inputkan parameter yang sesuai dengan PRNG yang dipilih. Dari parameter-parameter tersebut maka penanda (*marker*) dapat dicek apakah ada atau tidak, apabila ada berarti citra tersebut berisi *watermark* didalamnya sehingga proses *extracting* akan dilakukan dan sebaliknya jika tidak.



Gambar 4.2 *Diagram alur extracting*

2. Penerapan Algoritma *Blum Blum Shub*

Untuk algoritma BBS, dimisalkan nilai parameter p yang digunakan adalah 31, parameter q-nya 19, dan *seed*-nya 13. Untuk mendapatkan jumlah LSB yang diperlukan dilakukan perhitungan logaritma dimensi citra yaitu: round (^{2} log 240) = 7 (240 merupakan dimensi gambar, 15 x 16 piksel), sehingga bilangan acak akan dibangkitkan adalah sebanyak 7 kali. Berikut bilangan acak yang dihasilkan beserta kode binernya:

1) $X_1 = 13^2$ Mod 589= 169 = 10101001
2) $X_2 = 169^2$ Mod 589 = 289 = 00100001
3) $X_3 = 289^2$ Mod 589 = 472 = 11011000
4) $X_4 = 472^2$ Mod 589 = 142 = 10001110
5) $X_5 = 142^2$ Mod 589 = 138 = 10001010
6) $X_6 = 138^2$ Mod 589 = 196 = 11000100
7) $X_7 = 196^2$ Mod 589 = 131 = 10000011

Dari 7 *byte* kode biner yang dihasilkan oleh bilangan acak tersebut dihasilkan 1 *byte* kode biner baru dengan mengambil 1 LSB dari tiap *byte*. Satu *byte* kode biner baru tersebut harus diubah ke dalam bentuk desimal untuk mendapatkan piksel awal yang akan disisipkan *watermark*, sehingga $1100001_2 = 97_{10}$. Karena piksel pada citra dimulai dari 0 maka piksel awal untuk proses penyisipan adalah 97 -1 = 96.

Watermark akan disisipkan perblok, 1 blok berisi 3 karakter. Karena *watermark* yang akan disisipkan berjumlah 3 karakter, maka hanya diperlukan 1 blok. Satu blok memerlukan 8 piksel sehingga nantinya proses penyisipan akan dimulai dari piksel ke 96 sampai dengan piksel ke 103. Untuk mendapatkan letak kolom dan baris piksel dalam citra maka dilakukan perhitungan sebagai berikut:

1) kolom = piksel Mod lebarCitra
   = 96 mod 16
   = 0
2) baris = Int (piksel / lebarCitra)
   = Int(96 /16)
   = 6

Watermark disisipkan mulai dari piksel yang terletak pada kolom 0 baris 6. Satu piksel citra dapat menampung 3 bit *watermark*. Ketiga bit *watermark* tersebut akan ditambahkan pada byte warna merah, hijau dan biru pada piksel

tersebut. Pada piksel 0,6 nilai warna merahnya = 192 (11000000_2), warna hijaunya = 192 (11000000_2), dan warna birunya = 192 (11000000_2). Tiga bit watermark pertama yang akan disisipkan adalah 010, sehingga menjadi:

11000000 11000001 11000000 = 192_{10} 193_{10} 192_{10}

Dengan cara yang sama maka nilai warna merah, hijau dan biru piksel selanjutnya dapat dihitung. Berikut tabel untuk nilai warna merah (R), hijau (G), dan biru (B) sebelum dan sesudah penyisipan.

Piksel	Kolom,baris	Sebelum Penyisipan			Sesudah Penyisipan		
		R	G	B	R	G	B
117	5,7	255	255	255	254	255	254
118	6,7	255	0	0	254	0	0
119	7,7	255	255	255	254	254	254
120	8,7	255	255	255	254	255	254
121	9,7	255	0	0	254	1	0
122	10,7	255	255	255	254	254	254
123	11,7	255	255	255	255	255	254
124	12,7	255	0	0	254	1	0

Tabel 4.1 *Nilai warna RGB sebelum dan sesudah penyisipan*

UJI COBA DAN ANALISA HASIL

Data yang digunakan untuk uji coba ada 2 macam yaitu *watermark* berupa teks berformat .txt dan media citra berformat .jpg dan .bmp.

1. *Input Watermark*

   Input *watermark* adalah *watermark.txt* yang berisi seperti di bawah ini:

   Tegar in action ©juni2011

A. Original Image

Original Image yang digunakan ada enam gambar yaitu *Tegar.jpg*, *Tegar2.jpg* (citra *grayscale* dari *Tegar.jpg*), *Tegar 3.jpg* (citra biner dari *Tegar.jpg*), *Guntur.bmp*, *Guntur2.bmp* (citra *grayscale* dari *Guntur.bmp*), *dan Guntur3.bmp* (citra biner dari *Guntur.bmp*).

Gambar 5.1.1.1 *Tegar.jpg*, *Tegar2.jpg,* dan *Tegar3.jpg*

Gambar 5.1.1.2 *Guntur.bmp*, *Guntur2.bmp,* dan *Guntur3.bmp*

2. Kombinasi Kunci

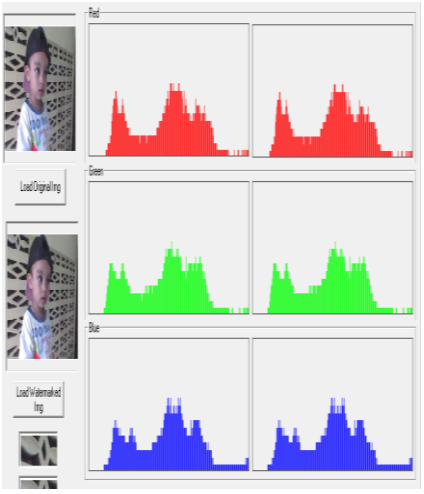
dilihat pada tabel 4.1 di bawah Kombinasi kunci yang digunakan untuk setiap *original image* dapat ini.

Tabel 4.1 Kombinasi kunci

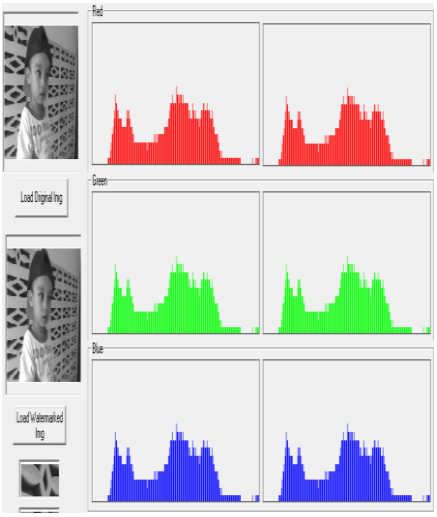
Original Image	Kombinasi Kunci LCG	Kombinasi Kunci BBS
Tegar.jpg	106-1283-6075	31-19-13
Tegar2.jpg	211-1663-7875	131-11-71
Tegar3.jpg	421-1663-7875	227-107-211
Guntur.bmp	106-1283-6075	31-19-13
Guntur2.bmp	211-1663-7875	131-11-71
Guntur3.bmp	421-1663-7875	227-107-211

A. Perbandingan Histogram

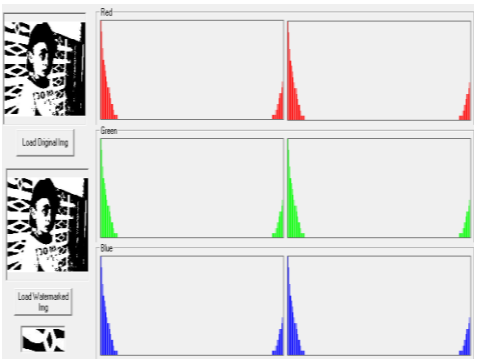
Histogram *original image* dan *watermarked image* yang telah dihasilkan dapat dilihat seperti di bawah ini:



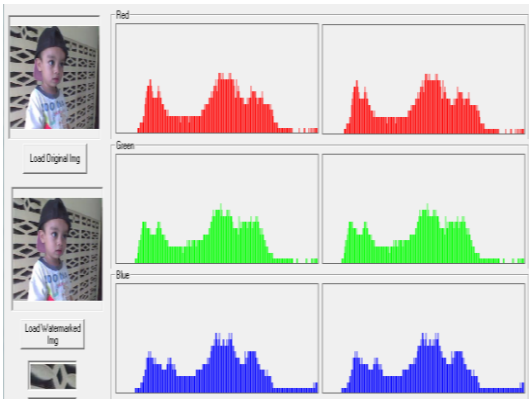
Gambar 5.2.1.1 Histogram *Tegar.jpg* (kiri) dan *TegarLCG.jpg* (kanan)



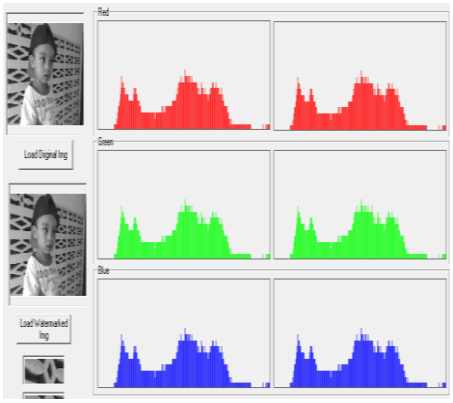
5.2.1.2 Histogram *Tegar2.jpg* (kiri) dan *Tegar2LCG.jpg* (kanan)



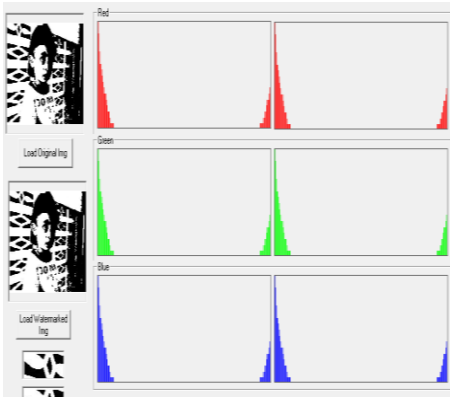
Gambar 5.2.1.3 Histogram *Tegar3.jpg* (kiri) dan *Tegar3LCG.jpg* (kanan)



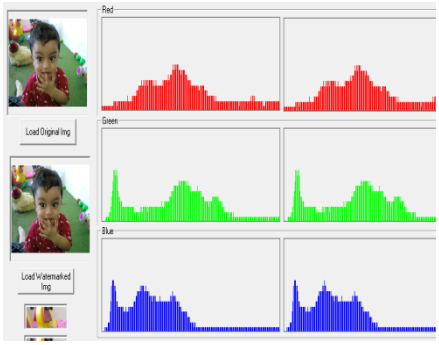
Gambar 5.2.1.4 Histogram *Tegar.jpg* (kiri) dan *TegarBBS.jpg* (kanan)



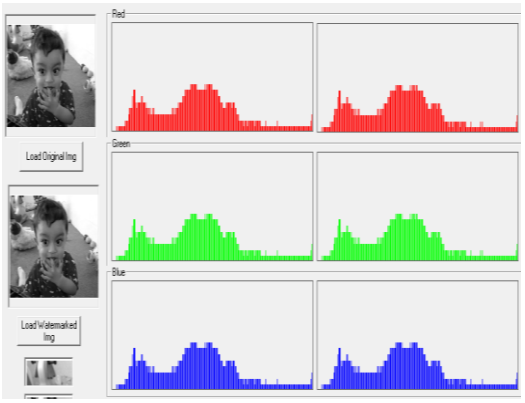
Gambar 5.2.1.5 Histogram *Tegar2. jpg* (kiri) dan *Tegar2BBS. jpg* (kanan)



Gambar 5.2.1.6 Histogram *Tegar3. jpg* (kiri) dan *Tegar3BBS. jpg* (kanan)



Gambar 5.2.1.7 Histogram *Guntur.bmp* (kiri) dan *GunturLCG.bmp* (kanan)



Gambar 5.2.1.8 Histogram *Guntur2.bmp* (kiri) dan *Guntur2LCG.bmp* (kanan)

Dari semua histogram di atas dapat kita lihat bahwa hampar tidak ada perubahan pada citra sebelum dan disisipi *watermark*. Hal ini di karenakan metode LSB hanya mengubah satu bit LSB dari satu *byte* warna dan perubahan satu bit ini tidak mengubah warna tersebut secara berarti.

Secara visual kedua PRNG bekerja sangat baik dengan menghasilkan citra ber-*watermark* yang terlihat sama dengan citra aslinya.

KESIMPULAN

Dari pengujian dan analisa hasil yang telah dilakukan, diperoleh beberapa kesimpulan, yaitu:

1. *Pseudo Random Number Generator* (PRNG) *Linear Congruential Generator* (LCG) bekerja sama baiknya dengan PRNG Blum Blum Shub (BBS).
2. Untuk penyisipan *watermark* pada citra biner dan *grayscale* nilai PSNR yang dihasilkan lebih kecil dan nilai MSE yang dihasilkan lebih besar hal ini dikarenakan terjadinya penumpukan jumlah piksel pada *range* tertentu pada citra *grayscale* (0-4 untuk citra *grayscale* 2 bit, 0-8 untuk citra *grayscale* 3 bit, dst) dan

untuk citra biner hanya pada nilai 0 atau 255.

3. Kualitas *watermarked image* dipengaruhi oleh ukuran *original image,* format gambar dan jumlah karakter *watermark* yang disisipkan.

DAFTAR PUSTAKA

1. Basuki, A., Josua F. Palandi, dan Fatchurrohman (2005). Pengolahan Citra Digital menggunakan Visual Basic. Graha Ilmu Yogyakarta.
2. Curran K, Bailey K (2003). An Evaluation of Image Based Steganography Methods 2: 16. http://www.ijde.org (di akses 21 November 2010).
3. Groves, J (2005). Notes for 620-351: Number Theory. Department of Mathematics and Statistics. University of Melbourne.
4. Johnson dan Jajodia S (1998). Exploring Steganography: Seeing the Unseen. George Mason University. http://www.jjtc.com/pub/r2026.pdf (di akses 21 November 2010).
5. Manaf AA, Zeki AM. (2006). Watermarking of Digital Images. 1st ENGAGE European-South East Asia ICT Research Collaboration. Malaysia, 29-31 Mar 2006. Malaysia: University Technology Malaysia.
6. Mulopulos GP, Hernandez AA, Gasztonyi LS. (2003). Peak Signal to Noise Ratio Performance Comparison of JPEG and JPEG 2000 for Various Medical Image Modalities. Compressus Inc. http://www.debugmode.com/imagecmp/ SCAR CompressionRatio - Jun 2003.pdf [07 Juni 2007].
7. Munir, R. (2004). Diktat Kuliah IF5054 Kriptografi: Steganografi dan Watermarking. Institut Teknologi Bandung.
8. Munir, R. (2005). Matematika Diskrit Edisi ketiga. Informatika Bandung.
9. Munir, R. (2006). Kriptografi. Informatika Bandung.
10. Octovhiana, K. (2003). Cepat Mahir Visual Basic 6.0. http://www.ilmukomputer.com (di akses 4 Desember 2010).
11. Sutoyo, T (2009). Teori Pengolahan Citra Digital. Andi Yogyakarta.
12. Thung, F. (2008). Aplikasi Teori Bilangan Bulat dalam Pembangkitan Bilangan Acak Semu. Makalah Program Studi Teknik Informatika ITB.