Analisis Algoritma Sum-Delta pada Kriptografi Visual dan Perbandingannya dengan Algoritma XOR

Aliffiqri Agwar - 13517107[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13517107@std.stei.itb.ac.id

***Abstract*—Plagiasi dan pengakuan hak milik pada suatu karya yang bukan miliknya sering terjadi pada era digital. Tidak hanya plagiasi, pengguna teknologi juga mulai ditemukan melakukan pemalsuan gambar. Kriptografi visual dibutuhkan untuk mencegah hal-hal ini terus terjadi. Pengamanan pada kriptografi visual termasuk pengamanan yang mudah dilakukan. Kriptografi visual juga dapat menjangkau seluruh gambar baik hitam putih maupun berwarna. Algoritma XOR merupakan algoritma termudah dalam mengamankan gambar dari pihak yang tidak bertanggung jawab. Namun, terpikirkan bahwa algoritma XOR bersifat *reversible.* Dalam tulisan ini akan dijelaskan salah satu algoritma *reversible* lainnya dan perbandingannya dengan algoritma XOR.**

***Keywords*—kriptografi visual, algoritma XOR, algoritma sum-delta.**

I. PENDAHULUAN

Satu gambar memiliki sejuta cerita. Kalimat tersebut merupakan gambaran yang tepat untuk sebuah gambar. Tidak hanya cerita, dalam gambar juga menyimpan banyak informasi. Informasi yang disimpan tidak hanya bersifat visual, namun juga tersimpan hingga piksel terkecil. Informasi tidak hanya makna dari gambar, namun juga menyimpan informasi berupa ukuran gambar, warna, hingga nilai bit pada warna tersebut.

Pada era digital saat ini, akses teknologi sangat mudah didapatkan. Mulai dari mesin pencarian, pengiriman dan penerimaan data hingga manipulasi data. Hal ini tidak terkecuali pengaksesan dan manipulasi gambar. Salah satu contohnya adalah penggunaan Adobe Photoshop yang dapat mengubah gambar menjadi keiinginan pengguna.

Pengiriman dan penerimaan data juga rentan menjadi sasaran pihak peretas yang ingin mendapatkan data tersebut. Apalagi gambar yang dikirim berupa gambar penting, baik secara harga maupun informasi. Oleh karena itu dibutuhkan sebuah algoritma kriptografi yang dapat mengamankan gambar tersebut.

Kriptografi visual memiliki satu ciri khas. Gambar yang dienkripsi menghasilkan dua atau lebih gambar baru yang terlihat tidak memiliki makna. Namun apabila gambar-gambar tersebut ditumpuk, maka akan memberikan gambar aslinya. Hal ini memerlukan setidaknya dua orang yang dapat menyimpan hasil enkripsi.

Terdapat beberapa algoritma dalam kriptografi visual. Ada algoritma OR yang paling sederhana[1]. Ada pula kriptografi (k,n) [2] yang paling umum digunakan.Terkhusus pada citra berwarna, terdapat algoritma XOR yang penggunaannya mudah dan hasilnya yang akurat. Algoritma XOR ini yang akan menjadi pembanding dari algoritma Sum-Delta yang akan menjadi bahasan pada poin-poin berikutnya. Apabila algoritma XOR melakukan exclusive “or” logic pada setiap warna pada piksel, maka algoritma Sum-Delta melakukan plus minus pada warna tersebut pada sebuah angka pseudorandom.

II. KRIPTOGRAFI VISUAL

Kriptografi visual merupakan kriptografi yang khusus pada enkripsi dan dekripsi gambar. Hasil dari enkripsi sebuah gambar adalah dua atau lebih gambar yang hampir tidak memiliki makna. Namun, apabila gambar-gambar tersebut diolah (bahkan dekripsi yang paling sederhana adalah menumpuk gambar-gambar tersebut), akan menghasilkan gambar aslinya.

Kriptografi visual membutuhkan minimal dua orang yang *qualified* untuk memegang gambar hasil enkripsi tersebut. Sehingga apabila terjadi intervensi oleh pihak yang tidak bertanggung jawab, gambar asli tidak akan terbentuk.

Kriptografi visual memiliki keuntungan dapat dijalankan tanpa komputasi yang besar. Kemudian kriptografi visual memiliki dekripsi yang cepat. Tidak hanya itu, enkripsi dan dekripsinya dapat tidak menggunakan kunci. Namun, penggunaan kunci dapat diberikan untuk menambah kerumitan hasil enkripsi. Lalu, hasil enkripsi dari kriptografi visual sendiri merupakan kunci dari gambar aslinya. Hal yang perlu dilakukan hanya menggabungkan kedua kunci tersebut untuk dekripsi.

Setiap kelebihan pasti ada kekurangan. Begitu pula dengan kriptografi visual. Kriptografi visual walaupun dapat mengamankan gambar dari intervensi pihak yang tidak bertanggung jawab, nyatanya hasil dekripsi banyak yang tidak sama dengan gambar aslinya. Hal ini menjadi masalah apabila kualitas gambar aslinya benar-benar dibutuhkan. Oleh sebab itu, penggunaan kriptografi visual sering digunakan pada watermarking yang apabila terdapat sedikit kerusakan pada watermark, tidak menjadi masalah selama watermark masih dikenali. Selain itu, gambar hasil dekripsi juga dapat mengandung noise.

Gambar hasil dari enkripsi sendiri karena tidak memiliki makna, maka dapat menimbulkan kecurigaan dari pihak lain yang melihat gambar tersebut. Sehingga gambar tersebut bisa diintervensi dan ditahan hingga pihak tersebut menemukan *share* yang lain untuk membuka gambar asli. Oleh karena itu, dibutuhkan bantuan steganografi untuk menyembunyikan gambar *share* tersebut dan mengurangi kecurigaan pihak lain.

III. CITRA BERWARNA

Citra berwarna merupakan gambar yang pembagian setiap pikselnya memiliki tiga buah informasi yang tersimpan pada tiga tuple 256 bit. Informasi tersebut berupa warna yang ada pada piksel tersebut. Warna tersebut diwakili dengan RGB (Red-Green-Blue) yang biasa dikenal dengan warna adisi maupun CMYK (Cyan-Magenta-Yellow-Key) yang biasa dikenal dengan warna substraksi. Namun pada tulisan ini, seluruh proses enkripsi dan dekripsi menggunakan mode RGB. Pada mode RGB, semakin tinggi nilai bit maka semakin jelas warna tersebut. Contoh: merah terang akan memiliki tuple (255,0,0) dan ungu gelap memiliki tuple (125,0,125)

IV. ALGORITMA XOR

A. Pengertian

Algoritma XOR merupakan algoritma yang memecah gambar asli menjadi tepat dua gambar *share* (gambar yang akan dipegang oleh orang terpercaya). Berbeda dengan algoritma lain, algoritma XOR tidak melakukan pemecahan piksel sehingga ukuran hasil enkripsi tepat sama dengan gambar aslinya.

B. Algoritma Enkripsi

Algoritma enkripsi pada algoritma XOR menggunakan tahapan sebagai berikut:

1. Mengambil nilai RGB dari setiap piksel dan menyimpannya menjadi tuple (r,g,b)
2. Untuk setiap r, g dan b akan dipecah menjadi dua buah nilai r1,g1,b1 untuk share 1 dan r2,g2,b2 untuk share 2.
3. Nilai dari r1, g1 dan b1 di inisiasi dengan pseudorandom dengan kunci k yang berasal dari pengguna. Hasil random harus berada di antara 0 hingga 255.
4. Nilai dari r2, g2 dan b2 didapatkan dari XOR setiap bit dari warna yang ada pada piksel asli dan piksel hasil pseudorandom. Sehingga r2 = r XOR r1.
5. Ulangi step 3 dan 4 untuk setiap piksel yang ada pada gambar asli.
6. Setelah info dari kedua share selesai dibentuk, gabungkan piksel setiap share hingga didapatkan dua buah share hasil enkripsi.

Source Code 4.1. Algoritma Enkripsi XOR

```
r, g, b = datum
  red = genbitarray(r)
  green = genbitarray(g)
  blue = genbitarray(b)

    share1_red = []
    share1_green = []
    share1_blue = []
    for i in range(8) :
        share1_red.append(random.randint(0, 100) %
2)
        share1_green.append(random.randint(0, 100)
% 2)
        share1_blue.append(random.randint(0, 100)
% 2)

    share2_red = xor_arr(red, share1_red)
    share2_green = xor_arr(green, share1_green)
    share2_blue = xor_arr(blue, share1_blue)

    xor_share1 = (array_to_int(share1_red),
array_to_int(share1_green),
array_to_int(share1_blue))
    xor_share2 = (array_to_int(share2_red),
array_to_int(share2_green),
array_to_int(share2_blue))

    return xor_share1, xor_share2
```

C. Algoritma Dekripsi

Algoritma dekripsi pada algoritma XOR menggunakan tahapan sebagai berikut:

1. Mengambil nilai RGB dari setiap piksel pada kedua buah share dan menyimpannya menjadi tuple (r1,g1,b1) untuk share 1 dan (r2,g2,b2) untuk share 2.
2. Nilai dari RGB hasil dekripsi merupakan XOR kedua tuple RGB dari share.
3. Sehingga nilai r dekripsi adalah r1 XOR r2.
4. Lakukan XOR untuk setiap RGB pada setiap piksel.
5. Setelah seluruh piksel telah di XOR, lakukan penggabungan untuk menampilkan gambar asli dari kedua share tersebut.

Source Code 4.2. Algoritma Dekripsi XOR

```
    red1, green1, blue1 = share1
    red2, green2, blue2 = share2

    red = array_to_int(xor_arr(genbitarray(red1),
genbitarray(red2)))
    green =
array_to_int(xor_arr(genbitarray(green1),
genbitarray(green2)))
    blue =
array_to_int(xor_arr(genbitarray(blue1),
genbitarray(blue2)))

    return red, green, blue
```

V. ALGORITMA SUM-DELTA

A. Pengertian

Algoritma Sum-Delta adalah algoritma rancangan penulis yang menggunakan kunci tambahan untuk menentukan seed pada pseudorandom yang dilakukan. Algoritma Sum-Delta

bersifat reversible. Hal ini serupa dengan algoritma XOR. Perbedaan yang paling mencolok dari algoritma XOR adalah algoritma Sum-Delta menggunakan penjumlahan nilai piksel dengan nilai pseudorandom untuk share pertama dan hasil pengurangan nilai piksel dengan nilai pseudorandom untuk mendapatkan share kedua.

B. Algoritma Enkripsi

Algoritma enkripsi pada algoritma Sum-Delta menggunakan tahapan sebagai berikut:

1. Mengambil nilai RGB dari setiap piksel dan menyimpannya menjadi tuple (r,g,b)
2. Inisiasi nilai kunci key_r, key_g, key_b dengan pseudorandom berkunci k yang berasal dari input pengguna.
3. Nilai dari r1 merupakan (r + key_r) mod 256. Begitu pula dengan g1 dan b1.
4. Nilai dari r2 merupakan (r - key_r) mod 256. Begitu pula dengan g2 dan b2.
5. Lakukan tahap 2 hingga 4 pada seluruh piksel pada gambar asli.
6. Setelah tahap 5 selesai, simpan kedua share tersebut.

Source Code 5.1. Algoritma Enkripsi Sum-Delta

```
    r,g,b = datum
    r_key, g_key, b_key = key
    share1_red = (r + r_key) % RGB_SCALE
    share1_green = (g + g_key) % RGB_SCALE
    share1_blue = (b + b_key) % RGB_SCALE

    share2_red = mod_delta(r - r_key, RGB_SCALE)
    share2_green = mod_delta(g - g_key, RGB_SCALE)
    share2_blue = mod_delta(b - b_key, RGB_SCALE)

    rgb_share1 = (share1_red, share1_green,
share1_blue)
    rgb_share2 = (share2_red, share2_green,
share2_blue)

    return rgb_share1, rgb_share2
```

C. Algoritma Dekripsi

Algoritma dekripsi pada algoritma Sum-Delta menggunakan tahapan sebagai berikut:

1. Ambil nilai RGB dari piksel kedua buah share dan simpan sebagai (r1, g1, b1) dan (r2, g2, b2).
2. Inisiasi tuple kunci berdasarkan pseudorandom dengan kunci k dan disimpan sebagai (key_r, key_g, key_b).
3. Hasil dekripsi warna merah dari share 1 (r1`) menggunakan (r1 - key_r) mod 256. Lakukan pula pada warna hijau dan biru.
4. Hasil dekripsi warna merah dari share 2 (r2`) menggunakan (r2 + key_r) mod 256. Lakukan pula pada warna hijau dan biru.
5. Setelah kedua share telah terdekripsi, bandingkan kedua r1` dan r2`. Apabila terdapat perbedaan, dapat memilih salah satu baik memilih nilai maksimum atau minimum dari keduanya.

Source Code 5.2. Algoritma Dekripsi Sum-Delta

```
    red1, green1, blue1 = rgb1
    red2, green2, blue2 = rgb2
    r_key, g_key, b_key = key

    red_a = mod_delta(red1 - r_key, RGB_SCALE)
    green_a = mod_delta(green1 - g_key, RGB_SCALE)
    blue_a = mod_delta(blue1 - b_key, RGB_SCALE)

    red_b = mod_delta(red2 + r_key, RGB_SCALE)
    green_b = mod_delta(green2 + g_key, RGB_SCALE)
    blue_b = mod_delta(blue2 + b_key, RGB_SCALE)

    red = max(red_a, red_b)
    green = max(green_a, green_b)
    blue = max(blue_a, blue_b)

    rgb = (red, green, blue)
    return rgb
```

VI. PEAK SIGNAL-TO-NOISE RATIO

Peak Signal-To-Noise Ratio (PSNR) merupakan metrik pengukur kualitas gambar yang telah dimanipulasi dibandingkan dengan gambar aslinya. Nilai PSNR didapatkan dengan persamaan:

$$PSNR = 20\ x\ log_{10}(\tfrac{255}{rms})$$

$$rms = \sqrt{\frac{1}{MN}\sum_{i=1}^{N}\sum_{j=1}^{M}(X_{ij} - Y_{ij})^2}$$

Satuan dari PSNR adalah desibel (dB). Nilai PSNR berbanding terbalik dengan nilai rms. Sehingga apabila nilai PSNR semakin besar, maka nilai rms semakin kecil. Semakin kecil nilai rms, maka perbedaan antara piksel pada gambar asli (X) dan gambar manipulasi (Y) semakin kecil. Oleh karena itu, batas minimal untuk sebuah gambar adalah 30 dB. Apabila di atas 30 dB, maka gambar manipulasi dapat ditoleransi.

Source Code 6.1. Algoritma PSNR

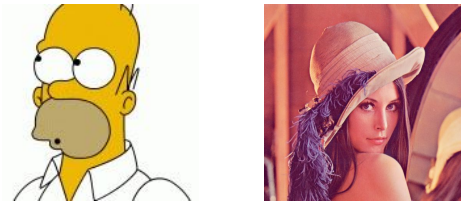
```
from math import log10, sqrt
import cv2
import numpy as np


def PSNR(original, compressed):
    mse = np.mean((original - compressed) ** 2)
    if (mse == 0):  # MSE is zero means no noise
is present in the signal .
        # Therefore PSNR have no importance.
        return 100
    max_pixel = 255.0
    psnr = 20 * log10(max_pixel / sqrt(mse))
    return psnr
```

VII. PERCOBAAN

Percobaan algoritma XOR dan Sum-Delta menggunakan tahapan sebagai berikut:

1. Melakukan enkripsi gambar asli dan menyimpan kedua gambar share hasil enkripsi. Terkhusus Sum-Delta menggunakan kunci.
2. Dari kedua gambar tersebut, dilakukan dekripsi sesuai algoritma masing-masing.
3. Setelah kedua tahap tersebut, akan dilakukan perbandingan PSNR.

Kedua algoritma diuji dengan dua gambar berwarna (yang tidak gray-scale) yang berbeda. Pada gambar 7.1, gambar kiri (Homer) merupakan gambar berwarna yang memiliki lebih banyak space putih dibandingkan gambar kanan (Lenna). Gambar kiri juga memiliki dimensi lebih kecil dibandingkan gambar kanan.



Gambar 7.1. Dua gambar yang akan menjadi variabel uji: Homer (kiri) dan Lenna (kanan)

A. Lingkungan Pengujian

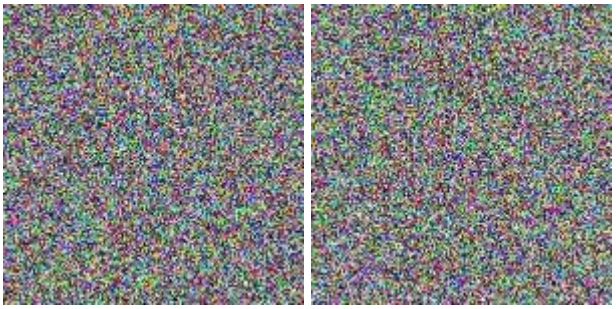
Kedua gambar uji dienkripsi dan dekripsi dengan lingkungan sebagai berikut:

Tabel 7.1. Lingkungan Pengujian

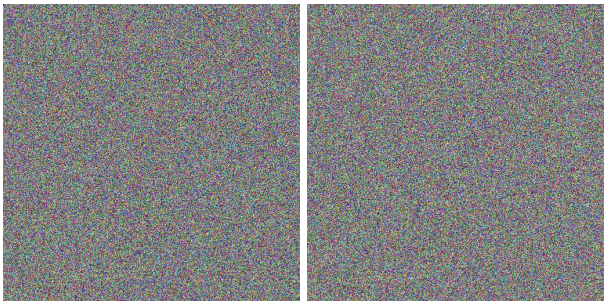
OS	Windows 10
GPU	Nvidia GTX 1080Ti
RAM	4 GB
CPU	Intel i5 780

B. Hasil Enkripsi XOR

Pengujian enkripsi algoritma XOR menghasilkan buah share sebagai berikut:



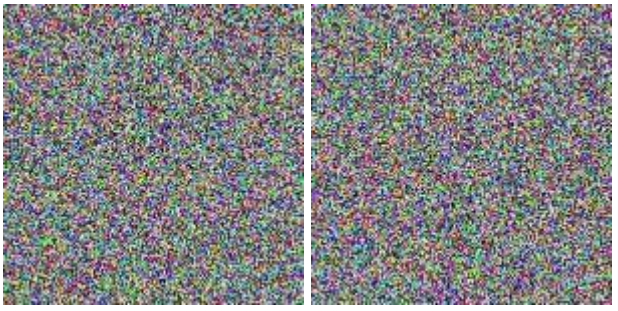
Gambar 7.2. Dua buah share hasil enkripsi dari gambar Homer dengan algoritma XOR



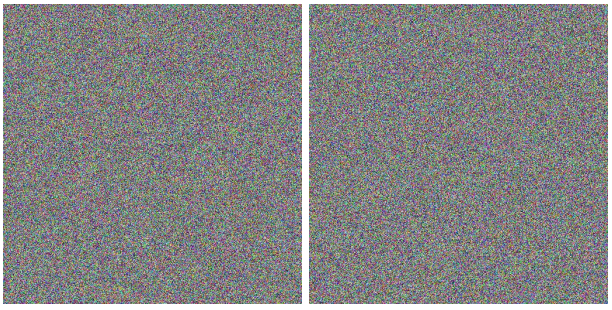
Gambar 7.3.Dua buah share hasil enkripsi dari gambar Lenna dengan algoritma XOR

C. Hasil Enkripsi Sum-Delta

Pengujian enkripsi kedua gambar uji dengan kunci k=126. Hasil enkripsi adalah sebagai berikut:



Gambar 7.4. Dua buah share hasil enkripsi dari gambar Homer dengan algoritma Sum-Delta



Gambar 7.5. Dua buah share hasil enkripsi dari gambar Lenna dengan algoritma Sum-Delta

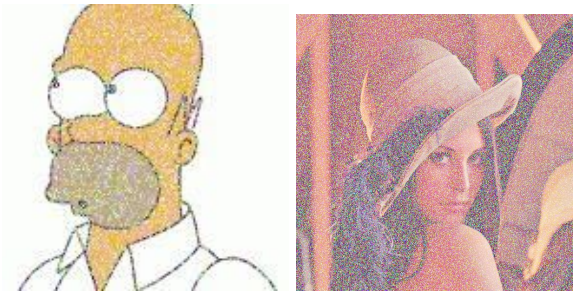
D. Hasil Dekripsi XOR

Pengujian dekripsi algoritma XOR menghasilkan buah share sebagai berikut:

Gambar 7.6. Gambar Homer dan Lenna hasil dekripsi dengan algoritma XOR dari share hasil enkripsi dengan algoritma yang sama

E. Hasil Dekripsi Sum-Delta

Pengujian dekripsi kedua gambar uji dengan kunci k=126. Dekripsi menggunakan mode max dan min. Hasil dekripsi adalah sebagai berikut:



Gambar 7.7. Gambar Homer dan Lenna hasil dekripsi dengan mode max dan algoritma Sum-Delta dari share hasil enkripsi dengan algoritma yang sama.

Gambar 7.8. Gambar Homer dan Lenna hasil dekripsi dengan mode min dan algoritma Sum-Delta dari share hasil enkripsi dengan algoritma yang sama.

F. Pengujian PSNR

Untuk membandingkan kedua algoritma, maka diadakan uji kualitas dengan PSNR. Hasil uji dari PSNR dari kedua algoritma tersebut adalah sebagai berikut:

Tabel 7.2. Hasil pengujian PSNR

Algoritma - Mode	Hasil PSNR (dB)	
	Homer	Lenna
XOR	34.81	100
Sum-Delta Max	32.05	30.94
Sum-Delta Min	28.02	30.85

VIII. UJI KEAMANAN

Pada algoritma Sum-Delta, dibutuhkan key untuk membangkitkan bilangan acak. Hal ini untuk mencegah peretas yang telah mendapatkan kedua share untuk langsung mendapatkan gambar asli dengan melakukan XOR. Oleh karena itu, perlu diuji hasil gambar apabila kunci yang dimasukkan salah. Hasil uji tersebut dilakukan sebanyak tiga kali. Pertama, tepat di bawah kunci asli. Kedua, tepat di atas kunci asli. Ketiga, nilai yang sangat jauh dari kunci asli. Dengan k=126, maka hasil uji adalah sebagai berikut:

Gambar 8.1. Uji keamanan gambar Homer dengan k=125, k=127 dan k=6000

IX. ANALISIS

Berdasarkan hasil percobaan, terdapat beberapa poin yang dapat dianalisis:

1. Hasil enkripsi
   Hasil enkripsi dari algoritma XOR dan Sum-Delta memiliki kualitas yang serupa. Dimana gambar benar-benar tersembunyi dan tidak memiliki makna bagi yang melihatnya. Hal ini membuktikan bahwa Sum-Delta memiliki kualitas yang sama dengan algoritma XOR dalam hal enkripsi.

2. Hasil dekripsi
   Hasil dekripsi dari algoritma XOR dan Sum-Delta memiliki hasil yang mirip dengan aslinya. Namun, Sum-Delta sangat dipengaruhi dengan mode yang dipakai. Mode maksimum menyebabkan hasil dekripsi terlihat lebih terang dibandingkan gambar aslinya. Begitu pula sebaliknya, mode minimum menyebabkan hasil dekripsi terlihat lebih gelap. Sementara algoritma XOR memberikan hasil dekripsi dengan intensitas terang dan kejelasan yang sangat baik.

3. Noise
   Noise hampir tidak terlihat pada hasil dekripsi dengan algoritma XOR. Namun, pada algoritma Sum-Delta noise yang ditemukan sangat banyak. Terutama pada piksel yang terang dengan mode minimum (lihat gambar 7.8 pada Homer) serta piksel yang gelap dengan mode maksimum (lihat gambar 7.7 pada Lenna).

4. Hasil PSNR
   Hasil dari kedua algoritma menunjukkan hasil yang sangat kontras di antara kedua gambar uji. Pada algoritma XOR dan gambar uji Lenna yang memiliki detail yang lebih tinggi, nilai PSNR dapat mencapai 100 dB. Hal ini sulit dijangkau oleh algoritma Sum-Delta mode apapun.
   Namun, dengan gambar Homer yang memiliki persebaran piksel yang sederhana, nilai PSNR dari kedua algoritma menghasilkan selisih yang tipis.

5. Hasil uji keamanan
   Pengujian dengan kunci yang berbeda pada algoritma Sum-Delta memberikan hasil yang cukup berbeda dengan gambar aslinya. Namun, pada k=125 garis tepi dan pemetaan warna pada gambar Homer hampir terlihat seperti aslinya. Pada nilai kunci lainnya, gambar benar-benar tidak sama dengan aslinya sehingga peretas akan kesulitan mendapatkan gambar aslinya kecuali dengan metode brute-force.

X. KESIMPULAN

Berdasarkan hasil percobaan dan analisis, dapat disimpulkan bahwa algoritma Sum-Delta dapat dipertimbangkan menjadi algoritma kriptografi visual. Menimbang hasil enkripsi yang kualitasnya mirip dengan algoritma XOR serta hasil dekripsi yang masih berada pada batas toleransi.

Terdapat beberapa kelebihan pada algoritma Sum-Delta. Kelebihan tersebut adalah terdapat lapisan tambahan yang membantu menjaga gambar dari peretas. Lapisan tambahan berupa key yang membangkitkan bilangan acak dapat menghambat proses peretasan.

Namun, terdapat kekurangan pada algoritma ini. Seperti banyaknya noise yang terjadi apabila menggunakan mode minimum. Serta hasil PSNR yang berada di ambang batas toleransi.

UCAPAN TERIMA KASIH

Melalui tulisan ini, saya mengucapkan terima kasih kepada Allah Yang Maha Esa. Berkat bantuan-Nya, saya masih diberikan kesempatan untuk menyelesaikan studi ini hingga akhir semester.

Tidak lupa ucapan terima kasih juga saya berikan kepada Bapak Rinaldi Munir selaku dosen pengajar IF4020 - Kriptografi yang telah memberikan ilmu mengenai cara untuk mengamankan sebuah data. Ilmu tersebut benar-benar berguna untuk saya di kemudian hari nanti.

Ucapan terima kasih pula saya berikan kepada teman-teman sekelas IF4020 - Kriptografi yang telah membantu saya memahami mata kuliah ini.

REFERENSI

[1] Rinaldi M., "Kriptografi Visual (Bagian 1)", http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Kriptografi-Visual-Bagian1.pdf (diakses pada 19 Desember 2020)

[2] http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Kriptografi-Visual-Bagian2.pdf (diakses pada 19 Desember 2020)

[3] M., Karolin & Meyyapan, T. (2015). RGB Based Secret Sharing Scheme in Color Visual Cryptography. 4. 151-155. 10.17148/IJARCCE.2015.4734. (diakses pada 19 Desember 2020)

[4] Rinaldi M., "Steganography (Bagian 2)", http://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/Steganografi-Bagian2-2020.pdf (diakses pada 20 Desember 2020)

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 21 Desember 2020

Aliffiqri Agwar - 13517107