Ketahanan Steganografi Citra Metode LSB terhadap Perubahan Ukuran Citra dan Variasi Teknik Interpolasi

Ryan Dharma Chandra / 13217018[1]
Program Studi Teknik Elektro
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
[1]13217018@std.stei.itb.ac.id, [1]ryandchandra1@gmail.com

***Abstract*—Steganografi citra adalah penyembunyian pesan atau informasi pada citra agar tidak terdeteksi oleh pihak lain Salah satu metode steganografi citra digital yang paling sederhana adalah metode LSB. Salah satu operasi yang sering dilakukan pada citra adalah perubahan ukuran citra. Dalam makalah ini, akan diuji kebenaran pesan dari citra yang diskalakan dengan skala 10% - 300% dengan step 10% dan dikembalikan ke ukuran awal, dengan teknik interpolasi yang berbeda-beda pada library OpenCV Python. Berdasarkan hasil pengujian, beberapa teknik interpolasi menyebabkan pembesaran citra dengan skala bilangan bulat memberikan kebenaran pesan 100%. Pesan pada skala 200% benar untuk teknik interpolasi nearest neighbor dan area. Pesan pada skala 300% benar untuk semua teknik interpolasi. Performansi terbaik untuk skala bukan bilangan bulat diraih oleh interpolasi cubic dan lanczos4.**

Keywords—citra, interpolasi, LSB, pesan, steganografi

I. PENDAHULUAN

Tidak bisa dipungkiri bahwa dunia ini penuh dengan rahasia, termasuk rahasia antar manusia. Dalam pengiriman pesan, pesan-pesan rahasia kerap kali disembunyikan pada sesuatu yang lebih besar. Konsep ini disebut sebagai steganografi. Seiring dengan perkembangan teknologi, steganografi sudah mulai banyak diterapkan pada ranah digital, salah satunya adalah penyembunyian pesan pada sebuah citra digital (steganografi citra).

Terdapat banyak teknik untuk melakukan steganografi citra. Salah satu teknik yang paling sederhana adalah metode LSB, yang menyembunyikan bit-bit pesan pada bit terakhir dari setiap pixel pada citra. Akibat ketidakmampuan mata manusia untuk membedakan warna yang berbeda hanya 1 bit, manusia tidak dapat mendeteksi keberadaan pesan tersebut. Pihak yang dapat mendeteksi perbedaan tersebut hanyalah pihak yang dapat membaca bit-bit pixel seperti mesin.

Pengiriman pesan yang disembunyikan dengan steganografi utamanya memiliki suatu harapan, yaitu agar pesan dapat sampai ke tempat tujuan dengan tepat. Tepat disini berarti pesan yang diterima tersebut sama seperti pesan aslinya, tidak ada informasi yang terganti ataupun terbuang. Dalam steganografi citra, berarti pengiriman citra *stego image* harus sampai ke tempat tujuan dengan baik sedemikian sehingga pihak penerima dapat mengekstrak pesan yang disembunyikan dengan baik.

Salah satu operasi citra yang cukup sederhana tetapi banyak terjadi adalah perubahan ukuran citra. Perubahan ukuran citra berarti mengubah panjang dan lebar citra, dengan atau tanpa mengubah *aspect ratio*-nya. Secara digital, operasi ini berarti penambahan atau pengurangan pixel dalam citra. Dalam hal ini, pixel-pixel dari citra hasil operasi dipilih berdasarkan pixel citra awal sedemikian rupa sehingga tampilan dari citra hasil operasi adalah citra awal yang dibesarkan, dikecilkan, ataupun di-*stretch*. Pemilihan pixel-pixel citra hasil operasi berdasarkan pixel citra awal dilakukan berdasarkan teknik interpolasi citra.

Pada makalah ini, akan diuji ketahanan dari citra *stego image* dengan metode LSB yang ukurannya diubah dengan skala 10% - 300% dengan step 10%, kemudian dikembalikan ke ukuran awal lagi untuk diekstrak pesan dari citra tersebut. Dilakukan variasi teknik interpolasi dari proses tersebut, yaitu interpolasi *area*, interpolasi *linear*, interpolasi *nearest neighbor*, interpolasi *cubic*, dan interpolasi lanczos4 menggunakan library OpenCV pada Python. Untuk masing-masing pesan yang diekstrak, akan dicek ketahanan pesan dengan membandingkannya dengan pesan awal, pada level bit, level karakter (byte), dan errornya.

II. DASAR TEORI

A. Steganografi Citra Metode LSB

Steganografi adalah seni penyembunyian pesan tertentu sehingga tidak ada pihak yang mengetahui ataupun menyadari pesan tersembunyi tersebut. Sebagian besar dari teknik steganografi dilakukan dengan menyisipkan pesan tersebut ke berkas lain yang disebut sebagai *cover object*. [1]

Seiring dengan perkembangan teknologi digital, berkas *cover object* dapat menggunakan berkas digital seperti gambar, audio, ataupun video digital sebagai bentuk dari steganografi digital. Steganografi citra menyembunyikan pesan rahasia pada *cover object* berupa gambar digital yang disebut sebagai *cover image* untuk menghasilkan *stego image*. [1]

Salah satu teknik yang paling umum dalam steganografi citra digital adalah metode LSB. Pada metode ini, pesan disembunyikan dalam *least significant bit* (LSB) dari masing-

masing byte dari konten citra. [1] Sebagai contoh, berikut adalah ilustrasi penyembunyian beberapa bit data ke dalam representasi byte dari citra.

pesan	10101101			
citra	10110110	10101001	00101101	10101101
	00100010	10100011	01101100	01011000
pesan+citra	1011011**1**	1010100**0**	0010110**1**	1010110**0**
	0010001**1**	1010001**1**	0110110**0**	0101100**1**

Keberhasilan steganografi citra metode LSB ini disebabkan oleh manusia yang tidak dapat mendeteksi perbedaan antara *cover image* dan *stego image* secara visual. Fenomena ini muncul akibat ketidakmampuan mata manusia untuk mendeteksi perbedaan warna yang berbeda hanya sebanyak 1 bit [1], sehingga persepsi manusia terhadap *cover image* dan *stego image* secara umum sama.

Metode LSB ini juga dapat digunakan pada steganografi digital lainnya seperti steganografi audio dan video, namun tidak pada steganografi teks karena perubahan 1 bit akan mengubah karakter pada teks.

B. Perubahan Ukuran Citra dan Interpolasi

Salah satu operasi pada citra yang cukup sederhana tetapi banyak dilakukan adalah perubahan ukuran citra. Terdapat beberapa jenis perubahan ukuran citra, seperti

- Pembesaran / pengecilan ukuran citra dengan *aspect ratio* yang tetap (*rescaling*).
- Perubahan ukuran dan *aspect ratio* citra (*resizing*).

Secara visual, *rescaling* hanya mengubah ukuran dari citra sehingga citra hanya terlihat lebih besar atau lebih kecil. Akan tetapi, pada *resizing*, terjadi distorsi secara visual pada citra seperti *stretch* ke salah satu arah sehingga citra terlihat berbeda.

Secara digital, citra direpresentasikan dalam pixel. Operasi perubahan ukuran citra ini mengubah jumlah pixel yang terdapat pada citra awal dan citra hasil operasi. Untuk memperoleh visual citra output yang tidak jauh berbeda, pixel dari citra output harus diisi sesuai dengan pixel citra awal, yang tentu saja tidak dapat diambil dari pixel citra awal secara langsung. Untuk mengisi nilai dari pixel citra output, dilakukan interpolasi terutama pada pixel-pixel baru yang muncul (pada pembesaran ukuran citra) ataupun pixel yang tergabung (pada pengecilan ukuran citra).

Pada library OpenCV Python, proses perubahan ukuran citra dilakukan dengan fungsi resize dengan parameter [2]

- src : citra input yang akan diubah ukurannya
- dst : citra output hasil operasi
- dsize : ukuran baru citra
- fx : skala sumbu horizontal
- fy : skala sumbu vertical
- interpolation : teknik interpolasi

Pada library OpenCV Python juga, berikut adalah beberapa teknik interpolasi yang ada [2]

- INTER_NEAREST : interpolasi *nearest neighbor*
- INTER_LINEAR : interpolasi *bilinear*
- INTER_CUBIC : interpolasi *bicubic*
- INTER_AREA : interpolasi dengan hubungan pixel area
- INTER_LANCZOS4 : interpolasi lanczos4 8x8

III. RANCANGAN DAN IMPLEMENTASI

A. Metodologi

Alat dan bahan yang digunakan adalah

- Komputer
- Python
- Library OpenCV
- *Integrated Development Environment* (IDE)

Pengujian dilakukan dengan pembuatan citra *stego image* dengan metode LSB dari sebuah citra dan pesan rahasia berupa teks. Selanjutnya, citra *stego image* tersebut di-*rescale* dengan skala 10% - 300% dengan step 10%, kemudian dikembalikan lagi ke ukuran semulanya. Untuk masing-masing citra hasil operasi tersebut, akan diekstrak pesan dari citra tersebut. Pengujian dengan skala 100% berarti tidak melakukan apapun pada citra.

Terdapat tiga parameter performansi yang dihitung sebagai berikut,

1. Persentase jumlah kesamaan bit (beserta posisinya) dari pesan yang diekstrak dari citra hasil operasi dengan pesan awal, selanjutnya disebut sebagai BitScore.
2. Persentase jumlah kesamaan karakter (beserta posisinya) dari pesan yang diekstrak dari citra hasil operasi dengan pesan awal, selanjutnya disebut sebagai Score.
3. Kedekatan pesan yang diekstrak dari citra hasil operasi dengan pesan awal (berupa *mean square error* relatif terhadap nilai maksimum elemen (255)), selanjutnya disebut sebagai ErrorScore.

Citra yang digunakan sebagai *cover image* terdiri dari 768 x 480 pixel yang disimpan dalam format .bmp dengan ukuran 1.05 MB. Citra ini disimpan dengan nama citra_ori.bmp. Citra ini dapat dilihat sebagai berikut.

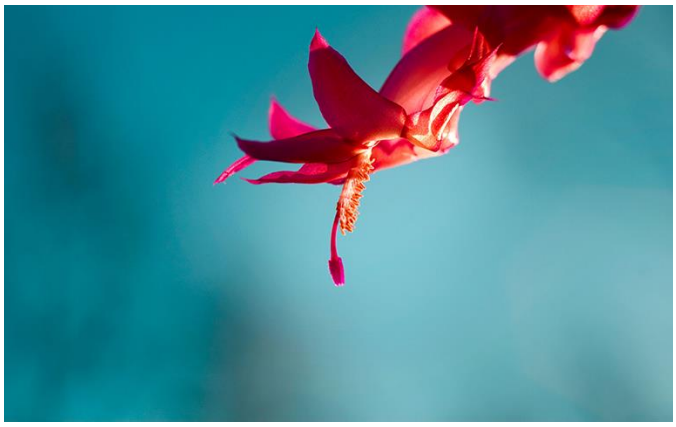

Fig. 1. Citra *cover image*

Pesan yang akan disembunyikan terdiri dari 705 karakter dengan yang terdiri dari beberapa *line* dengan karakter berupa huruf, angka, dan tanda baca yang masih berada dalam 255 karakter ASCII pertama. Pesan ini disimpan pada file eksternal message.txt. Berikut adalah pesan yang digunakan.

```
20200218
Please I don't want to scream
(Devil eyes come, eyes open, eyes open)
Please I don't want to scream
(Scream scream scream scream)
Spreading in the darkness, scream

20200817
The night grows deeper
Wrapped around in stormy clouds
Thunder, please come
Drop thunder hard
I'll lock up that BOCA

Where is the love
If those thorny words
Start to hurt you
I won't let them open that BOCA

20210126
See the hidden secret, odd eye
Fall even deeper inside
Covered by the sweetness
Of all of the lies

The eyes have worn down the border
Despair coming at you till the end
Will there ever be an end?
Back and forth, this isn't the place you've been looking for
No more Utopia
```

B. *Rancangan Algoritma*

Algoritma yang dirancang akan bekerja sebagai berikut. Mula-mula, dilakukan pengambilan pesan dan citra *cover image* dari berkasnya masing-masing. Kemudian, pesan tersebut akan disembunyikan dalam *cover image* dengan metode LSB untuk menghasilkan *stego image*. Selanjutnya, *stego image* akan diperkecil/diperbesar dengan skala 10% - 300% dengan step 10% dan disimpan menjadi berkas citra baru. Lalu, masing-masing dari *stego image* tersebut akan dikembalikan ke ukuran aslinya. Untuk masing-masing citra *stego image* yang telah dikembalikan ke ukuran aslinya, akan diekstrak pesan dari LSB citra, kemudian dicek kesamaannya dengan pesan asli.

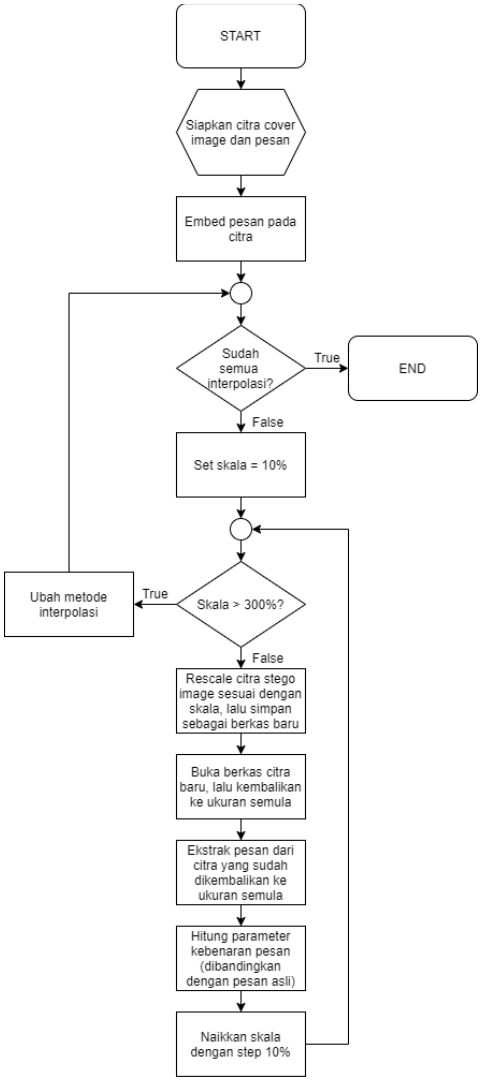

Fig. 2. *Flowchart* Algoritma

C. *Implementasi*

Implementasi dari algoritma yang dibuat dilakukan dalam kode Python. Pada kode ini, terdapat beberapa buah fungsi yang didefinisikan sebagai berikut.

Fungsi Embed yang menerima citra cover image RGB dan pesan yang akan disembunyikan dalam bentuk bit array, kemudian mengembalikan citra *stego image*.

```
def Embed(img, message):
    # Turn message into array of bits
    message_bit = StringToBitArray(message)

    msg_bit_idx = 0
```

```
    msg_bit_length = len(message_bit)

    # Embed message to image
    new_img = img
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            # Take each rgb pixel
            r, g, b = img[i][j]

            # Embed to LSB of r pixel
            new_r = (r&0xFE) +
message_bit[msg_bit_idx]
            msg_bit_idx = msg_bit_idx + 1

            if (msg_bit_idx==msg_bit_length):
                new_pixel = [new_r, g, b]
                new_img[i][j] = new_pixel
                break

            # Embed to LSB of g pixel
            new_g = (g&0xFE) +
message_bit[msg_bit_idx]
            msg_bit_idx = msg_bit_idx + 1

            if (msg_bit_idx==msg_bit_length):
                new_pixel = [new_r, new_g, b]
                new_img[i][j] = new_pixel
                break

            # Embed to LSB of b pixel
            new_b = (b&0xFE) +
message_bit[msg_bit_idx]
            msg_bit_idx = msg_bit_idx + 1

            if (msg_bit_idx==msg_bit_length):
                new_pixel = [new_r, new_g, new_b]
                new_img[i][j] = new_pixel
                break

            # Set as new pixel
            new_pixel = [new_r, new_g, new_b]
            new_img[i][j] = new_pixel

        if (msg_bit_idx==msg_bit_length):
            break

    return new_img
```

Fungsi Decode yang menerima citra *stego image* RGB dan panjang pesan yang akan diekstrak, kemudian mengembalikan pesan tersebut dalam bentuk string.

```
def Decode(img, msg_length):
    bit_array = []

    # Take each LSB from each rgb pixel and save
into array of bits
    for i in range(img.shape[0]):
        for j in range(img.shape[1]):
            r,g,b = img[i][j]

            # Take LSB from each rgb pixel
            bit_r = r & 0b1
            bit_g = g & 0b1
            bit_b = b & 0b1

            bit_array.append(str(bit_r))
            if (len(bit_array)>=(msg_length*8)):
                break

            bit_array.append(str(bit_g))
            if (len(bit_array)>=(msg_length*8)):
                break

            bit_array.append(str(bit_b))
            if (len(bit_array)>=(msg_length*8)):
                break

        if (len(bit_array)>=(msg_length*8)):
            break

    # Split array of bits into bytes
    byte_list = [bit_array[i:i+8] for i in
range(0, len(bit_array), 8)]

    # From each byte in list, convert into the
char value
    chr_list = [chr(int("".join(msg_byte),2)) for
msg_byte in byte_list]

    # Join the char into one string
    message = "".join(chr_list)

    return message
```

Fungsi Resize yang menerima citra yang akan di-resize beserta skala pembesarannya (dalam persen), kemudian mengembalikan citra yang sudah di-resize.

```
def Resize(img, new_size, method):
    return cv2.resize(img, new_size, interpolation
= method)
```

Fungsi Rescale yang menerima citra yang akan di-*rescale* beserta ukuran baru citra.

```
def Rescale(img, scale percent, method):
    # Rescale img using according to scale_percent
and method interpolation
    new_width = int(img.shape[1] * scale_percent /
100)
    new_height = int(img.shape[0] * scale_percent
/ 100)
    new size = (new width, new height)
    return Resize(img, new_size, method)
```

Fungsi OpenRGB yang menerima nama file dari citra dan mengembalikan objek citra tersebut dalam format RGB.

```
def OpenRGB(filename):
    # Open RGB image
    img = cv2.imread(filename)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    return img
```

Fungsi SaveRGB yang menerima objek citra dan nama file untuk menyimpan objek citra tersebut dengan nama file yang diberikan.

```
def SaveRGB(img, filename):
    # Save RGB image
    img = cv2.cvtColor(img, cv2.COLOR_RGB2BGR)
    cv2.imwrite(filename, img)
```

Fungsi Score yang menerima dua buah list, kemudian mengembalikan skor berupa persentase kesamaan elemen dari kedua list.

```
def Score(message, original_message):
    # Calculate the score of correctness (as
percentage) of the message
    # Metrics : number of same elements (and same
position) as the original message
    result = [message[i]==original_message[i] for
i in range(len(original_message))]
    return sum(result)/len(result)*100
```

Fungsi ErrorScore yang menerima dua buah list, kemudian mengembalikan skor berupa *mean square error* relatif terhadap nilai maksimum elemen (255) dari masing-masing elemen list.

```
def ErrorScore(message, original_message):
    # Calculate the mean square error / 255
between the message and original message
    result = [((ord(message[i])-
ord(original_message[i]))/255)**2 for i in
range(len(original_message))]
    return sum(result)
```

Fungsi StringToBitArray yang menerima sebuah string dan mengembalikan representasi array of bit dari string tersebut.

```
def StringToBitArray(message):
    # Convert string into array of bits
representation (no separator per char)
    message_bit = []
    for c in message:
        for i in [7,6,5,4,3,2,1,0]:
            message_bit.append((ord(c)&(1<<i))>>i)

    return message_bit
```

Fungsi RescaleTest yang menerima objek citra, pesan, dan metode interpolasi kemudian menguji ketahanan steganografi citra setelah di-*rescale* dengan skala 10%-300% dengan step 10%.

```
def RescaleTest(ori_img, original_message,
method):
    # Kode utama untuk mengecek ketahanan image
terhadap resize dengan interpolasi method
    print("Method : " + method)
    if (method=="AREA"):
        method = cv2.INTER_AREA
    elif (method=="LINEAR"):
        method = cv2.INTER_LINEAR
    elif (method=="CUBIC"):
        method = cv2.INTER_CUBIC
    elif (method=="NEAREST"):
        method = cv2.INTER_NEAREST
    elif (method=="LANCZOS4"):
        method = cv2.INTER_LANCZOS4
    else:
        print("Method not found")
        return

    for scale in range (10, 310, 10):
        stego_img = Embed(ori_img,
original_message)
        rescaled_stego_img = Rescale(stego_img,
scale, method)
        SaveRGB(rescaled_stego_img, "citra_stego_"
+ str(scale) + ".bmp")

        new_scaled_img = OpenRGB("citra_stego_" +
str(scale) + ".bmp")
        new_img = Resize(new_scaled_img,
(ori_width, ori_height), method)
        SaveRGB(new_img, "citra_stego_new_" +
str(scale) + ".bmp")

        message = Decode(new_img,
len(original_message))
        message_bit = StringToBitArray(message)
        print(scale, Score(message_bit,
original_message_bit),
Score(message,original_message),
ErrorScore(message,original_message), sep=',')

    print()
```

Program utama yang dibuat mengecek ketahanan steganografi citra dari beberapa teknik interpolasi yang dipanggil menggunakan fungsi RescaleTest sebagai berikut.

```
if (__name__ == "__main__"):
    # Get message from file
    message_file = open("message.txt","r")
    original_message = message_file.read()
    original_message_bit =
StringToBitArray(original_message)

    # Open cover image
    ori_img = OpenRGB("citra_ori.bmp")
    ori_width = ori_img.shape[1]
    ori_height = ori_img.shape[0]

    # Test
    RescaleTest(ori_img, original_message, method
= "AREA")
    RescaleTest(ori_img, original_message, method
= "LINEAR")
    RescaleTest(ori_img, original_message, method
= "CUBIC")
    RescaleTest(ori_img, original_message, method
= "NEAREST")
    RescaleTest(ori_img, original_message, method =
"LANCZOS4")
```

IV. Hasil dan Analisis

A. *Interpolasi Nearest Neighbor*

Pengujian pertama yang dilakukan adalah ketahanan steganografi citra dengan teknik interpolasi *nearest neighbor*. Hasil pengujian dengan step 20% dapat dilihat pada tabel berikut.

TABLE I. Hasil Pengujian Interpolasi Nearest Neighbor

Skala	BitScore	Score	ErrorScore
20%	50,73964497	0,443786982	94,17534794
40%	47,67011834	0,147928994	84,01897732

Skala	BitScore	Score	ErrorScore
60%	49,14940828	0,591715976	86,36418301
80%	51,27588757	0,295857988	79,03040369
100%	100	100	0
120%	52,29289941	0,147928994	69,80313725
140%	52,18195266	0,147928994	70,1919108
160%	52,27440828	0,295857988	69,6761707
180%	52,18195266	0,147928994	70,1919108
200%	100	100	0
220%	52,29289941	0,147928994	69,80313725
240%	52,18195266	0,147928994	70,1919108
260%	52,27440828	0,295857988	69,6761707
280%	52,21893491	0,147928994	70,05608612
300%	100	100	0

Dari tabel tersebut, diperoleh grafik sebagai berikut.

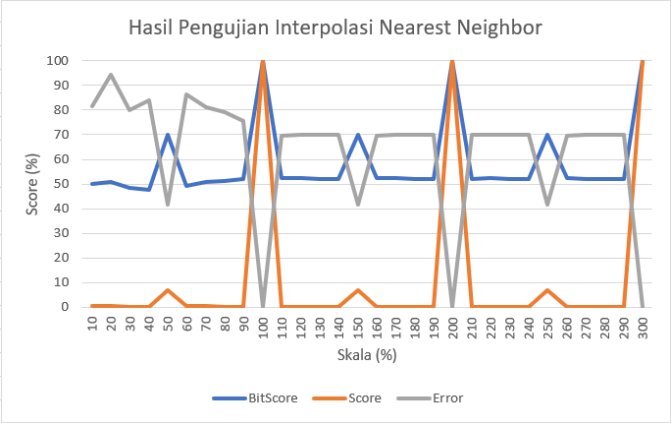

Fig. 3. Grafik Hasil Pengujian Interpolasi *Nearest Neighbor*

Berdasarkan hasil tersebut, terlihat bahwa kebenaran pesan 100% (seluruhnya benar) pada interpolasi *nearest neighbor* terjadi pada skala 100%, 200%, dan 300%. Nilai-nilai ini berarti pengubahan ukuran citra harus tepat kelipatan dari ukuran citra awal.

Untuk nilai-nilai skala yang berada di tengah kelipatan tersebut, terlihat bahwa hasilnya sangat kacau. Hanya sekitar 50% dari bit yang nilai dan posisinya benar, sehingga membuat isi pesan hampir seluruhnya salah. Ada beberapa kasus khusus pada skala dengan kelipatan 50% yang memberikan pesan yang "lebih benar" daripada skala lain. Perhatikan juga bahwa error (dan bentuk grafik) cukup *uniform* pada skala di atas 100%. Secara visual, berarti penggunaan skala di antara nilai-nilai kelipatan tersebut akan memberikan pesan yang sangat tidak tepat.

B. *Interpolasi Linear*

Pengujian kedua dilakukan dengan teknik interpolasi *linear*. Hasil pengujian dengan step 20% dapat dilihat pada tabel berikut.

TABLE II. HASIL PENGUJIAN INTERPOLASI *LINEAR*

Skala	BitScore	Score	ErrorScore
20%	52,40384615	0,887573964	77,72339869
40%	57,65532544	1,035502959	62,75813918
60%	64,01627219	4,289940828	56,39057286
80%	68,25073964	12,4260355	53,28602845
100%	100	100	0
120%	78,77218935	24,26035503	34,05411765
140%	80,45488166	32,10059172	34,80564398
160%	84,89275148	43,93491124	23,93574779
180%	86,1316568	52,36686391	25,64222991
200%	83,72781065	50	31,81783929
220%	88,44304734	60,94674556	20,37957709
240%	90,25517751	63,60946746	15,50685121
260%	90,36612426	64,05325444	16,16516724
280%	90,86538462	66,56804734	16,54515955
300%	100	100	0

Dari tabel tersebut, diperoleh hasil sebagai berikut.

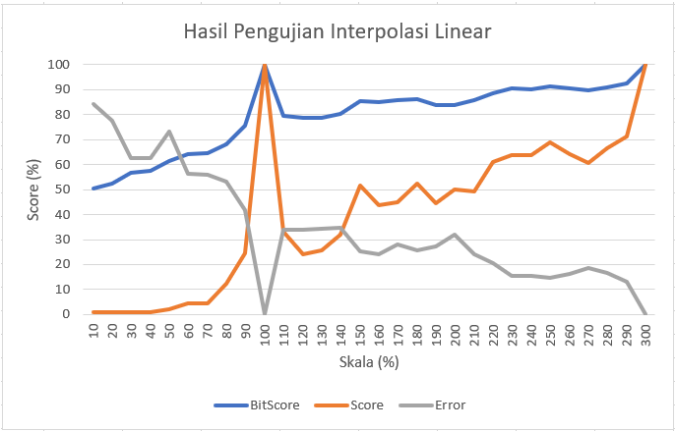

Fig. 4. Grafik Hasil Pengujian Interpolasi *Linear*

Dari grafik tersebut, terlihat bahwa kebenaran pesan 100% hanya terjadi pada skala 100% dan 300%. Mengingat penggunaan skala 100% tidak mengubah apapun pada citra, maka untuk kasus ini kebenaran pesan hanya terjadi pada skala 300%, atau tiga kali lipat dari ukuran citra awal.

Untuk nilai-nilai skala yang lain, terlihat bahwa skor dari parameter yang dihitung memiliki kecenderungan untuk naik. Penggunaan skala yang lebih besar memberikan kebenaran

pesan yang lebih baik, walaupun *trend* ini terkadang naik-turun. Walaupun kebenaran bit sudah cukup baik untuk kasus ini (saat skala mendekati 300%, skor sudah mencapai 90%), kebenaran karakter pesan masih cukup buruk (sekitar 60%). Dalam hal ini, pesan yang diekstrak akan sulit terbaca, namun masih ada beberapa karakter yang cocok.

C. *Interpolasi Cubic*

Pengujian ketiga dilakukan dengan teknik interpolasi *cubic*. Hasil pengujian dengan step 20% dapat dilihat pada tabel berikut.

TABLE III. HASIL PENGUJIAN INTERPOLASI *CUBIC*

Skala	BitScore	Score	ErrorScore
20%	51,68269231	0,887573964	78,17479431
40%	57,69230769	1,627218935	64,00461361
60%	64,20118343	3,99408284	55,35066513
80%	74,13091716	18,93491124	45,14466744
100%	100	100	0
120%	91,90088757	65,38461538	16,56641292
140%	94,39718935	77,21893491	9,569334871
160%	96,28328402	85,0591716	6,232233756
180%	97,37426036	87,72189349	4,796601307
200%	97,30029586	87,13017751	5,381622453
220%	98,29881657	90,82840237	3,17668589
240%	98,37278107	90,82840237	2,683306421
260%	98,35428994	91,27218935	3,171672434
280%	98,22485207	91,12426036	2,656239908
300%	100	100	0

Dari tabel tersebut, diperoleh grafik sebagai berikut.

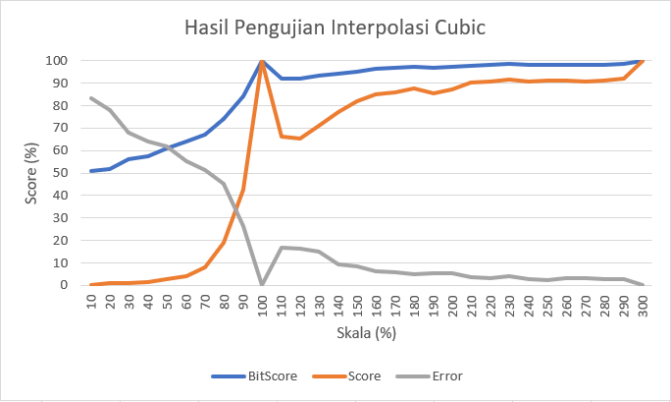

Fig. 5. Grafik Hasil Pengujian Interpolasi *Cubic*

Sama seperti pengujian dengan interpolasi *linear*, untuk interpolasi *cubic*, kebenaran pesan hanya terjadi pada skala 100% dan 300%, yang berarti fokus nilai yang akan dilihat adalah skala 300% saja.

Untuk nilai-nilai skala lain yang salah, terlihat bahwa terdapat *trend* naik yang sama seperti pada interpolasi *linear*. Namun, dapat dilihat bahwa skor pada interpolasi *cubic* ini lebih baik daripada interpolasi linear, yang mencapai kebenaran bit 98% dan kebenaran karakter 90% pada skala di atas 200%. Nilai ini menunjukkan bahwa kebenaran pesan cukup baik sehingga pesan yang diekstrak untuk nilai-nilai ini mungkin saja sebagian besar dapat dibaca.

D. *Interpolasi Area*

Pengujian keempat dilakukan dengan teknik interpolasi *area*. Hasil pengujian dengan step 20% dapat dilihat pada tabel berikut.

TABLE IV. HASIL PENGUJIAN INTERPOLASI *AREA*

Skala	BitScore	Score	ErrorScore
20%	52,5147929	0,73964497	91,43566321
40%	57,80325444	1,331360947	73,62359093
60%	63,40606509	4,733727811	61,09176471
80%	69,15680473	12,5739645	50,76492118
100%	100	100	0
120%	78,6612426	29,8816568	35,66352941
140%	83,0806213	44,37869822	31,37502499
160%	85,13313609	48,22485207	24,34437524
180%	87,13017751	55,0295858	22,82262207
200%	100	100	0
220%	90,16272189	65,0887574	16,95947712
240%	90,68047337	69,37869822	18,54291426
260%	91,49408284	73,37278107	14,71443291
280%	92,30769231	75,14792899	14,88535179
300%	100	100	0

Dari tabel tersebut, diperoleh grafik sebagai berikut.

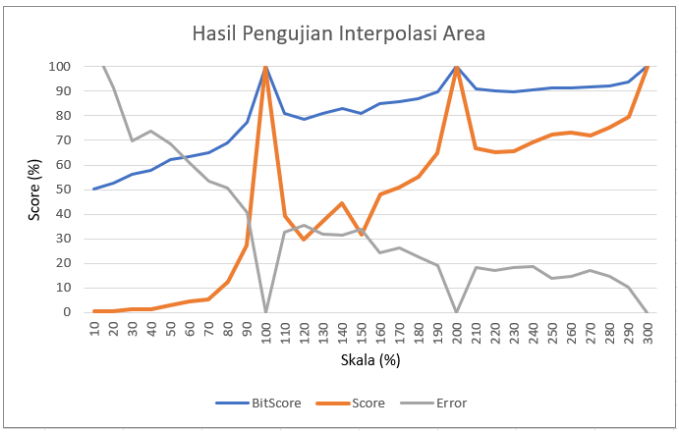

Fig. 6. Grafik Hasil Pengujian Interpolasi *Area*

Pada pengujian dengan interpolasi *area* ini, ada tiga buah nilai skala yang memberikan pesan yang benar, yaitu 100%, 200%, dan 300%. Perolehan hasil ini sama seperti pengujian dengan interpolasi *nearest neighbor*, yaitu skala-skala yang tepat kelipatan bilangan bulat.

Sama seperti hasil pengujian pada interpolasi lainnya, terjadi *trend* naik pada nilai-nilai skala di antara kelipatan tersebut. Namun, nilai skor pada skala-skala tersebut cukup rendah bila dibandingkan dengan hasil interpolasi sebelumnya pada interpolasi *linear*, yang hanya mencapai kebenaran pesan 75% pada skala mendekati 300%. Secara visual, nilai ini menunjukkan bahwa hanya tiga perempat pesan saja yang dapat terbaca.

E. Interpolasi Lanczos4

Pengujian terakhir dilakukan dengan teknik interpolasi lanczos4. Hasil pengujian dengan step 20% dapat dilihat pada tabel berikut.

TABLE V. HASIL PENGUJIAN INTERPOLASI LANCZOS4

Skala	BitScore	Score	ErrorScore
20%	52,68121302	0,887573964	72,88147636
40%	57,13757396	1,479289941	73,05688581
60%	63,36908284	3,698224852	51,82415994
80%	73,50221893	17,01183432	47,44213764
100%	100	100	0
120%	95,37721893	75,29585799	10,34122261
140%	96,52366864	80,17751479	8,219730873
160%	97,50369822	83,28402367	5,5327797
180%	97,91050296	87,27810651	4,957570165
200%	89,51553254	44,08284024	21,23866205
220%	97,85502959	86,53846154	4,483613995
240%	98,40976331	89,20118343	2,502760477
260%	98,00295858	88,31360947	4,585851596
280%	97,83653846	86,68639053	4,546543637
300%	100	100	0

Dari tabel tersebut, diperoleh grafik sebagai berikut.

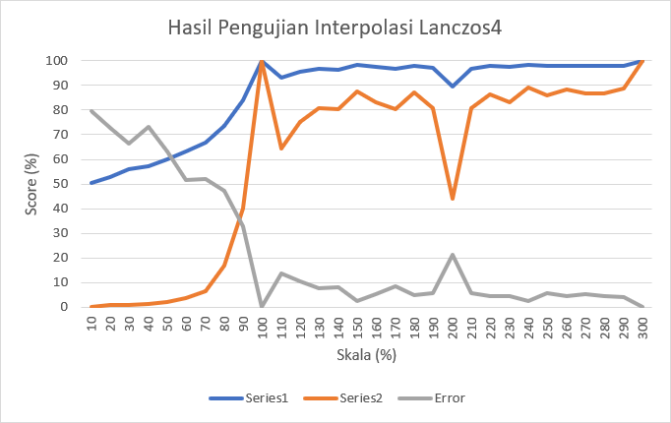

Fig. 7. Grafik Hasil Pengujian Interpolasi Lanczos4

Mirip seperti beberapa hasil sebelumnya, kebenaran skala terjadi pada skala 100% dan 300%. Mirip seperti hasil-hasil sebelumnya, terjadi kecenderungan naik dari skor terhadap kenaikan skala. Nilai ini juga cukup baik, yang mencapai kebenaran pesan lebih dari 85% pada skala di atas 200%, yang menunjukkan bahwa pesan masih cukup terbaca.

Namun, ada hal menarik dari hasil ini. Bila beberapa hasil pengujian lain memberikan nilai yang cukup tinggi pada nilai kelipatan bilangan bulat, nilai yang diperoleh oleh skala 200% justru jatuh bila dibandingkan dengan nilai-nilai di sekitarnya.

F. Analisis Lanjutan

Selanjutnya, akan dilakukan analisis dari masing-masing parameter terhadap teknik interpolasi. Grafik yang akan dilihat dapat dilihat sebagai berikut.

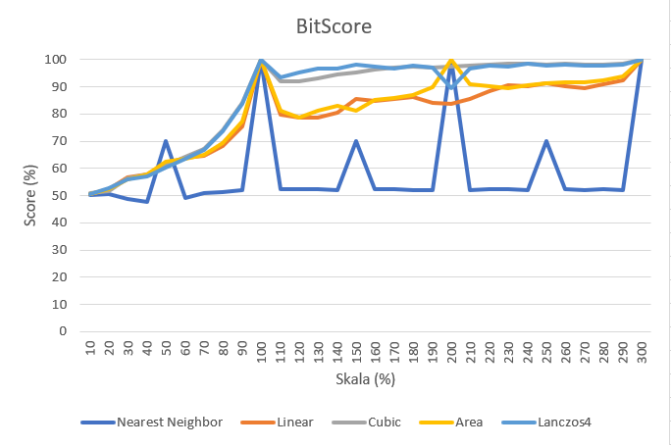

Fig. 8. Grafik BitScore terhadap Teknik Interpolasi

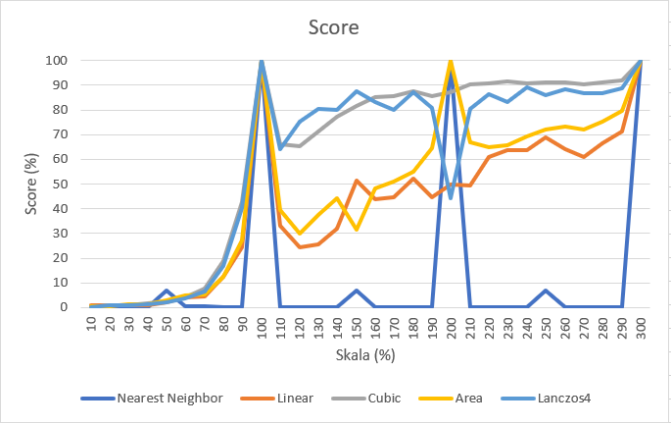

Fig. 9. Grafik Score terhadap Teknik Interpolasi

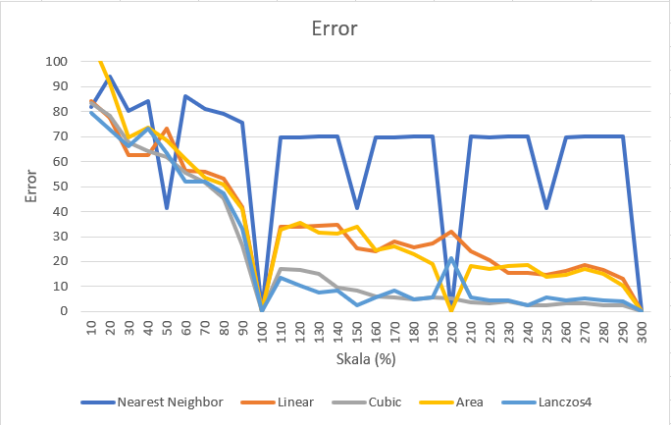

Fig. 10. Grafik Error terhadap Teknik Interpolasi

Berdasarkan grafik-grafik tersebut, terlihat bahwa terjadi *trend* yang hampir sama yaitu kenaikan skor dan penurunan error pada kenaikan skala terlihat pada hampir semua teknik interpolasi, kecuali pada interpolasi *nearest neighbor*. Pada interpolasi *nearest neighbor*, *trend* yang sama dengan teknik interpolasi lain terlihat pada skala di bawah 100%, namun pada skala di atas 100%, terjadi kemonotonan pada parameter-parameter interpolasi *nearest neighbor*.

Secara umum, performansi paling baik dapat diamati pada teknik interpolasi lanczos4 ataupun *cubic*, dan paling buruk pada interpolasi *nearest neighbor*. Untuk teknik interpolasi *linear* dan *area* berada di tengah-tengah.

Karena salah satu fokus utama dari steganografi adalah penyampaian pesan dengan baik, maka akan dilihat parameter kebenaran pembacaan karakter pesan yaitu Score pada Fig. 9. Berdasarkan grafik tersebut, pilihan skala yang memberikan pesan yang benar adalah pada kelipatan bilangan bulat yaitu 100% (tidak melakukan apapun), 200%, ataupun 300%. Pada skala 300%, seluruh teknik interpolasi memberikan kebenaran pesan 100%. Namun, kebenaran pesan 100% pada skala 200% hanya ditemui pada beberapa teknik interpolasi. Secara umum, berikut adalah rangkuman dari teknik interpolasi pada skala tersebut.

TABLE VI. RANGKUMAN SKALA DAN INTERPOLASI KEBENARAN PESAN

Skala	Nearest Neighbor	Linear	Cubic	Area	Lanczos4
200%	benar	salah	salah	benar	salah
300%	benar	benar	benar	benar	benar

Namun, mengingat *trend* kenaikan skor pada kenaikan skala pada beberapa teknik interpolasi (kecuali interpolasi *nearest neighbor*), mungkin saja penggunaan skala yang lebih besar lagi (hingga 400%, 500%, dan seterusnya) dapat memberikan skor yang mendekati 100% pada nilai-nilai skala di antara kelipatan bilangan bulat. Dalam hal ini, walaupun mungkin pesan yang diekstrak tidak terlalu tepat, sebagian besar informasi pada pesan dapat diterima.

Dari informasi tersebut, dapat ditambahkan lapisan keamanan baru pada proses steganografi, berupa pengubahan ukuran citra dengan skala berupa kelipatan bilangan bulat. Kemudian, citra ini lah yang ditransaksikan, sehingga apabila terdapat penyadapan, stegoanalisis dari citra tersebut lebih sulit dipecahkan. Dalam hal ini, pihak penerima harus tau ukuran awal citra (atau skala yang digunakan) serta teknik interpolasi yang digunakan untuk mengekstrak pesan tersebut. Namun, kekurangan dari metode ini adalah ukuran pesan yang ditransaksikan menjadi jauh lebih besar.

Di lain pihak, hasil pengujian ini juga memberitahukan bahwa pesan yang ada di dalam citra *stego image* akan rusak bila diskalakan dengan skala yang tidak tepat. Oleh karena itu, apabila diterima sebuah citra yang dicurigai mengandung pesan rahasia di dalamnya, citra tersebut dapat diskalakan untuk merusak pesan tersebut.

V. KESIMPULAN

Pada makalah ini, dilakukan pengujian ketahanan steganografi citra metode LSB terhadap perubahan ukuran citra dan variasi teknik interpolasi. Perubahan ukuran citra yang dimaksud adalah penskalaan 10% - 300% dengan step 10% dari citra, kemudian dikembalikan ke ukuran semula. Teknik interpolasi yang digunakan adalah *nearest neighbor*, linear, cubic, area, dan lanczos4 dari library OpenCV Python.

Kebenaran pesan seluruhnya terjadi pada skala 200% untuk teknik interpolasi *nearest neighbor* dan area, dan skala 300% untuk seluruh teknik interpolasi. Pada skala lainnya, terjadi *trend* kecocokan pesan dengan kenaikan skala kecuali pada interpolasi *nearest neighbor*.

Dalam hal ini, steganografi citra dapat diberikan lapisan keamanan baru berupa pembesaran ukuran citra *stego image* dengan skala kelipatan bilangan bulat. Dengan mentransaksikan citra yang sudah diperbesar, stegoanalisis menjadi lebih sulit. Namun, pihak penerima harus tau ukuran citra awal dan teknik interpolasi, serta ukuran citra yang ditransaksikan menjadi lebih besar. Di lain pihak, pengubahan ukuran citra *stego image* dengan sembarang dapat merusak pesan di dalamnya.

VIDEO LINK AT YOUTUBE

https://www.youtube.com/watch?v=_fnr7NypfWU

REPOSITORI

https://github.com/ryandchandra/ii4031-makalah/

UCAPAN TERIMA KASIH

Ucapan terima kasih dan puji syukur dipanjatkan kepada Tuhan yang Maha Esa karena berkat rahmat-Nya, penulis dapat menyelesaikan makalah ini dengan baik dan lancar. Penulis juga ingin mengucapkan terima kasih kepada Bapak Dr. Ir. Rinaldi Munir, M.T. sebagai dosen pengajar II4031 Kriptografi dan Koding untuk ilmu yang diberikan selama satu semester ini sehingga penulis dapat mengaplikasikan dan menuliskan makalah ini. Tidak lupa juga, penulis mengucapkan terima kasih kepada pihak-pihak lain termasuk keluarga dan teman yang telah membantu penulis dan memberikan dukungan.

REFERENSI

[1] R. Munir, "Steganografi", 2021. https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi-dan-Koding/Steganografi-2021.pdf (diakses 23 Mei 2021).

[2] OpenCV Docs, "Geometric Image Transform". https://docs.opencv.org/3.4/da/d54/group__imgproc__transform.html (diakses 23 Mei 2021).

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, 26 April 2021

Ryan Dharma Chandra
13217018