**LAPORAN TUGAS MATA KULIAH KEAMANAN INFORMASI ENKRIPSI MENGGUNAKAN ALGORITMA CTC (COLUMNAR TRANSPOSITION CIPHER) DAN RSA (RIVEST SHAMIR ADLEMAN**



Disusun Oleh :

|  |  |
| --- | --- |
| **Dwian Luisana P.** | **(167006030)** |
| **Ali Miftahul R.** | **(167006033)** |
| **Ahmad Gymnastiar** | **(167006065)** |
| **Cahyo Iriyanto** | **(167006082)** |

# JURUSAN INFORMATIKA FAKULTAS TEKNIK UNIVERSITAS SILIWANGI

**TASIKMALAYA 2019**

* 1. DEFINISI
     1. DEFINISI STEGANOGRAFI

Steganografi (steganography) adalah ilmu dan seni menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui. Steganografi berasal dari Bahasa Yunani, yaitu “steganos” yang artinya “tulisan tersembunyi (covered writing)” Steganografi termasuk ke dalam security through obscurity. Steganografi biasa digunakan oleh teroris, intelijen, atau militer dalam menyampaikan pesan sehingga tidak diketahui orang lain.

Penggunaan steganografi antara lain bertujuan untuk menyamarkan eksistensi (keberadaan) data rahasia sehingga sulit dideteksi dan melidungi hak cipta suatu produk. Steganografi dapat dipandang sebagai kelanjutan kriptografi. Jika pada kriptografi, data yang telah disandikan (ciphertext) tetap tersedia, maka dengan steganografi ciphertext dapat disembunyikan sehingga pihak ketiga tidak mengetahui keberadaannya. Data rahasia yang disembunyikan dapat diekstraksi kembali persis sama seperti keadaan aslinya.

Keuntungan steganografi dibandingkan dengan kriptografi adalah bahwa pesan yang dikirim tidak menarik perhatian sehingga media penampung yang membawa pesan tidak menimbulkan kecurigaan bagi pihak ketiga. Ini berbeda dengan kriptografi dimana ciphertext menimbulkan kecurigaan bahwa pesan tersebut merupakan pesan rahasia.

Pesan rahasia yang akan disembunyikan akan disisipkan pada suatu media penampung seperti citra, suara, video dan sebagainya yang terlihat tidak mencurigakan untuk menyimpan pesan rahasia. Pesan rahasia akan memerlukan sebuah kunci rahasia yang dinamakan stego-key agar hanyak pihak yang berhak saja yang dapat membuka atau mengekstak pesan rasahia tersebut.

Jenis-jenis Steganografi :

1. Injection

Merupakan teknik menanamkan pesan rahasia secara langsung terhadap suatu media, kekurangan dari teknik ini adalah media yang diinjeksi akan menjadi lebih besar dari ukuran normalnya sehingga mudah terdeteksi. Teknik ini juga disebut embedding.

1. Substitution

Teknik ini mengubah data normal menjadi data rahasia, hasil dari teknik ini biasanya tidak akan terlalu mengubah ukuran data asli tetapi tergantung pada data yang akan disembunyikan. Teknik ini akan menurunkan kualitas media yang ditumpangi.

1. Transformasi Domain

Teknik ini sangat efektif. Pada dasarnya, transformasi domain menyembunyikan data pada transform space.

1. Spread Spectrum

Merupakan teknik pentransmisian menggunakan pseudo-noise code, yang independen terhadap data informasi sebagai modulator bentuk gelombang untuk menyebarkan energi sinyal dalam sebuah jalur komunikasi (bandwith) yang lebih besar daripada sinyal jalur komunikasi informasi. Oleh penerima, sinyal dikumpulkan kembali menggunakan replika pseudo-noise code tersinkronisasi.

1. Statistical Method

Teknik ini disebut juga skema steganographic 1 bit. Skema tersebut menanamkan satu bit informasi pada media tumpangan dan mengubah statistik walaupun hanya 1 bit. Perubahan statistik ditunjukkan dengan indikasi 1 dan jika tidak ada perubahan, terlihat indikasi 0. Sistem ini bekerja berdasarkan kemampuan penerima dalam membedakan antara informasi yang dimodifikasi dan yang belum.

1. Distortion

Teknik ini menciptakan perubahan atas benda yang ditumpangi oleh data rahasia.

1. Cover Generation

Tenik ini lebih unik daripada metode lainnya karena cover objek dipilih untuk menyembunyikan pesan.

* + 1. DEFINISI ALGORITMA Discrete Wavelet Transform (DWT)

Discrete Wavelet Transform (DWT) adalah salah satu metode yang digunakan dalam pengolahan citra digital. DWT dapat digunakan untuk transformasi citra dan kompresi citra. Selain untuk pengolahan citra (gambar), metode DWT dapat juga diterapkan pada bidang steganografi. Dekomposisi citra digital menggunakan Discrete Wavelet Transform dilakukan dengan cara mengambil koefisien wavelet dari citra tersebut.

Proses Transformasi :

Konsep Transformasi Wavelet cukup sederhana. Citra awal (asli) ditransformasi dibagi (didekomposisi) menjadi 2 atau 4 sub-image baru. Setiap sub-image citra transformasi berukuran atau kali dari citra asli. Sub-image pada posisi kanan atas, kiri bawah, dan kanan bawah nampak seperti versi kasar dari citra asli dikarenakan berisi komponen frekuensi tinggi dari citra asli. Sedangkan untuk sub-image pada posisi kiri atas nampak seperti citra asli dan nampak lebih halus, dikarenakan berisi komponen frekuensi rendah dari citra asli.

Sedangkan untuk 1 sub-image atas kiri tampak seperti citra asli dan tampak lebih halus (smooth) karena berisi komponen frekuensi rendah dari citra asli. Karena mirip dengan citra asli, maka sub-image kiri atas dapat digunakan untuk melakukan aproksimasi terhadap citra asli. Sedangkan nilai piksel (koefisien) 3 sub-image yang lainnya cenderung bernilai rendah dan terkadang bernilai nol (0) sehingga mudah dikompresi.

* + 1. TAHAPAN METODE

1. Pengumpulan Data

Metode Penelitian yang digunakan adalah dengan studi literatur yang merupakan penelitian dengan mengumpulkan informasi dari sejumlah buku – buku, jurnal, artikel penelitian, dan lain sebagainya. Sehingga akan didapat kesimpulan dan referensi yang relevan dengan rumusan masalah.

Pada penelitian ini, digunakan alat penelitian berupa perangkat keras dan juga perangkat lunak. Kebutuhan perangkat keras yang digunakan di dalam penelitian ini adalah dua buah laptop dengan spesifikasi minimal sebagai berikut:

1. Processor Intel Core i3 @ 2.53 GHz

2. RAM 4 Gb

Adapun kebutuhan perangkat lunak yang digunakan pada penelitian ini adalah sebagai berikut:

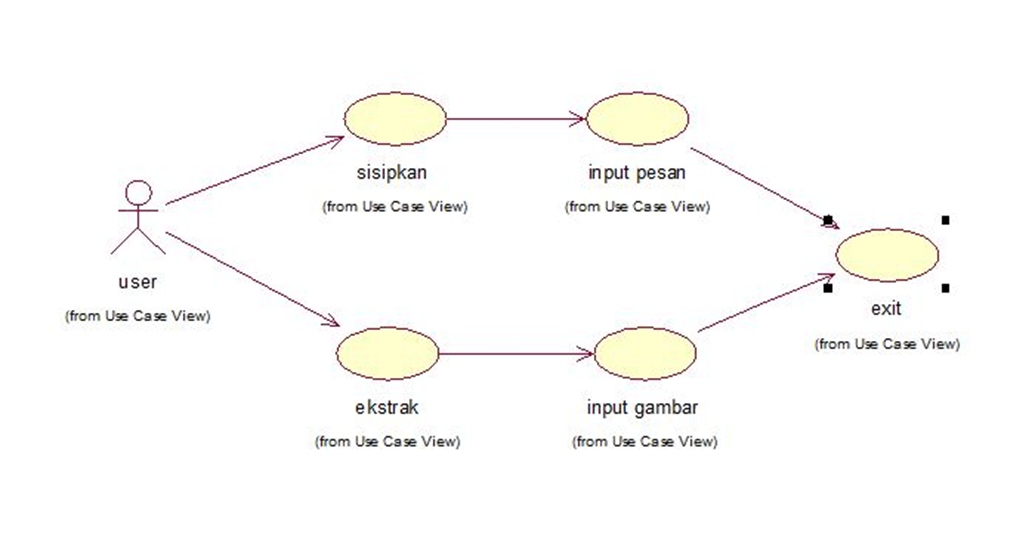
1. Sistem Operasi Windows 10

2. NetBeans IDE 8.2

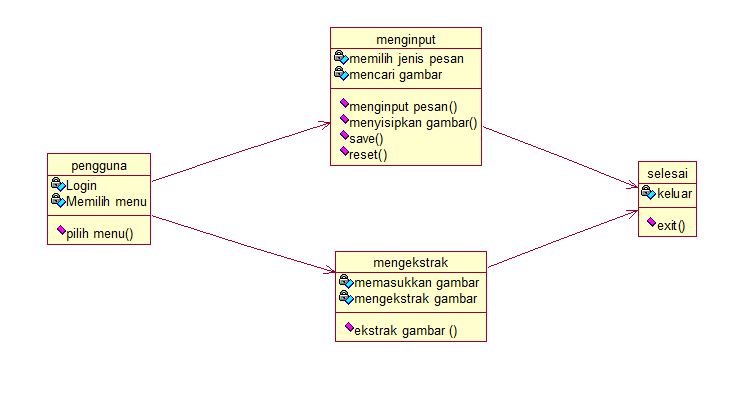
Bahan penelitian yang digunakan di dalam penelitian ini adalah berupa paper, skripsi, dan dokumentasi lainnya yang diperoleh dari world wide web.

1. Design dan Rancangan

* Use Case Diagram



* Class Diagram

****

1. Coding

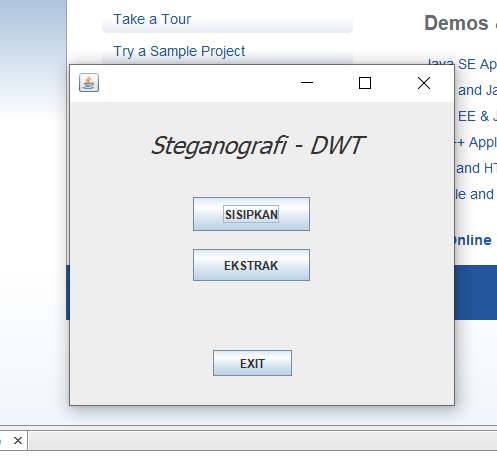
|  |
| --- |
| **DekripRGB.java** |
| package metode;  import java.awt.image.BufferedImage;  public class DekripRGB {    public int[] Memasukkanrgbtobitpesan(BufferedImage bf) {  int pixel;  int r, g, b;  int patokan = bf.getWidth() / 2;  int baris = 72, kolom = 72;  int[][] R = new int[baris][kolom];  int[][] G = new int[baris][kolom];  int[][] B = new int[baris][kolom];  int[][] buffer = new int[baris][kolom];  for (int i = 0, d = patokan; i < baris; i++, d++) {  for (int j = 0, e = patokan; j < kolom; j++, e++) {  buffer[i][j] = bf.getRGB(d, e);  pixel = buffer[i][j];  r = (pixel >> 16) & 0x000000FF;  g = (pixel >> 8) & 0x000000FF;  b = (pixel) & 0x000000FF;  R[i][j] = r;  G[i][j] = g;  B[i][j] = b;  }  }  int[] arraymaximals=new int[15552];  arraymaximals=membukabit\_toarray(R, 0, arraymaximals);  arraymaximals=membukabit\_toarray(G, 5184, arraymaximals);  arraymaximals=membukabit\_toarray(B, 10368, arraymaximals);  return arraymaximals;    }  private int[] membukabit\_toarray(int[][] masuk,int parameter, int[] arrayRGB) {  int inisialisasi=parameter;  for (int i = 0; i < masuk.length; i++) {  for (int j = 0; j < masuk.length; j++,inisialisasi++) {  arrayRGB[inisialisasi]=masuk[i][j];  }  }  return arrayRGB;  }  } |
| **RGB.java** |
| package metode;  import java.awt.image.BufferedImage;  import java.util.ArrayList;  public class RGB {  public int[] Memasukkanrgbtoarraylis(BufferedImage bf) {  int pixel;  int r, g, b;  int patokan = bf.getWidth() / 2;  int baris = 72, kolom = 72;  int[][] R = new int[baris][kolom];  int[][] G = new int[baris][kolom];  int[][] B = new int[baris][kolom];  int[][] buffer = new int[baris][kolom];  for (int i = 0, d = patokan; i < baris; i++, d++) {  for (int j = 0, e = patokan; j < kolom; j++, e++) {  buffer[i][j] = bf.getRGB(d, e);  pixel = buffer[i][j];  r = (pixel >> 16) & 0x000000FF;  g = (pixel >> 8) & 0x000000FF;  b = (pixel) & 0x000000FF;  R[i][j] = r;  G[i][j] = g;  B[i][j] = b;  }  }  int[] arraymaximal=new int[15552];  arraymaximal=memasukkanbit\_toarray(R, 0, arraymaximal);  arraymaximal=memasukkanbit\_toarray(G, 5184, arraymaximal);  arraymaximal=memasukkanbit\_toarray(B, 10368, arraymaximal);  return arraymaximal;  }  public void tampilarray(int[] arr) {    for (int i = 15552; i < 15555; i++) {  System.out.println(arr[i]+",");  }  }  private int[] memasukkanbit\_toarray(int[][] masuk,int parameter, int[] arrayRGB) {  int inisialisasi=parameter;  for (int i = 0; i < masuk.length; i++) {  for (int j = 0; j < masuk.length; j++,inisialisasi++) {  arrayRGB[inisialisasi]=masuk[i][j];  }  }  return arrayRGB;  }  public int [][] gettingR(int[] masuk){  int[][] tampung=new int[72][72];  int p = 0;  for (int i = 0; i < 72; i++) {  for (int j = 0; j < 72; j++, p++) {  tampung[i][j] = masuk[p];  }  }  return tampung;  }  public int [][] gettingG(int[] masuk){  int[][] tampung=new int[72][72];  int p = 5184;  for (int i = 0; i < 72; i++) {  for (int j = 0; j < 72; j++, p++) {  tampung[i][j] = masuk[p];  }  }  return tampung;  }  public int [][] gettingB(int[] masuk){  int[][] tampung=new int[72][72];  int p = 10368;  for (int i = 0; i < 72; i++) {  for (int j = 0; j < 72; j++, p++) {  tampung[i][j] = masuk[p];  }  }  return tampung;  }  } |
| **Classsisipke.java** |
| package metode;  import javax.swing.JOptionPane;  public class classsisipke {  public int[] masukkan\_teks\_kegambar(int[] lgambar, String pesanyangdisisipkan) {  //konversi menjadi byte  //byte temp\_byte\_gambar[] = byte\_gambar(gambar);  //mendapatkan jumlah byte gambar yang dipilih  byte temp\_byte\_pesan[] = pesanyangdisisipkan.getBytes();  //mendapatkan jumlah pbyte pesan yang akan dimasukkan  byte panjang\_data[] = konversi\_bit(temp\_byte\_pesan.length);  //mengkonversi jumlah pesan dalam 4 bit  try {  sisip\_teks(lgambar, panjang\_data, 0);  //memasukka jumlah teks ke dalam 16 bit pertama gambar  sisip\_teks(lgambar, temp\_byte\_pesan, 32);  //memasukkan teks ke dalam bit terakhir gambar  } catch (Exception e) {  JOptionPane.showMessageDialog(null,  "gambar tidak dapat diberikan sebuah teks", "Error", JOptionPane.ERROR\_MESSAGE);  }  return lgambar;  }  //memasukan file teks yang telah di olah untuk dimasukkan kedalam gambar  public int[] sisip\_teks(int[] gambar, byte[] data, int offset) {  //melakukan pengecekan apakah data gambar lebih besar dari data pesan setelah ditambah 32 bit  if (data.length + offset > gambar.length) {  throw new IllegalArgumentException("File terlalu besar!");  }  //perulangan untuk perubahan bit di setiap byte nya pesan  for (int i = 0; i < data.length; ++i) {  //memindahkan panjang data ke variabel baru di setiap bitnya  int add = data[i];  //perulangan untuk mengganti 8bit si setiap byte nya  for (int bit = 7; bit >= 0; --bit, ++offset) {  //data ke-i di shift right zero oleh bit kemudian di geser 1 bit  int b = (add >>> bit) & 1;  //mengganti bit terakhir dari byte gambar  gambar[offset] = (byte) ((gambar[offset] & 0xFE) | b);  //nilai byte yang lama dan 254 atau bit yang akan di tambahkan  }  }  return gambar;  }  //untuk mengubah nilai integer jumlah kata ke dalam 8 byte  public byte[] konversi\_bit(int i) {  byte byte3 = (byte) ((i & 0xFF000000) >>> 24);  byte byte2 = (byte) ((i & 0x00FF0000) >>> 16);  byte byte1 = (byte) ((i & 0x0000FF00) >>> 8);  byte byte0 = (byte) ((i & 0x000000FF));  return (new byte[]{byte3, byte2, byte1, byte0});  }  public byte[] ekstrak\_teks(int[] gambar) {  int panjang = 0;  int offset = 32;  //mengambil panjang pesan yang ada di 32 bit pertama  for (int i = 0; i < 32; ++i) {  panjang = (panjang << 1) | (gambar[i] & 1);  }  byte[] result = new byte[panjang];  System.out.println(result.length);  //perulangan untuk membaca setiap byte dari teks  for (int b = 0; b < result.length; ++b) {  //perulangan untuk membaca setiap bit didalam byte teks  for (int i = 0; i < 8; ++i, ++offset) {  //bit di masukkan ke dalam byte teks, dari bit baru di shift 1 atau byte text di geser 1 bit  //gambar dari 32 ke atas di dankan dengan 1 kemudian di OR kan dengan jumlah bit teks yang didapatkan dari atas  result[b] = (byte) ((result[b] << 1) | (gambar[offset] & 1));  }  }  return result;  }  public void tampilkango(byte[] data){  for(int i=0;i<data.length;i++){  System.out.print(data[i]+",");  if(i==8 && i==16 && i== 24){  System.out.println("\n");  }  }  }  public void tampilkan32(int[] data){  for(int i=32;i<=50;i++){  System.out.print(data[i]+",");  }  }  } |
| **Membentukgambar.java** |
| package metode;  import java.awt.Graphics2D;  import java.awt.image.BufferedImage;  import java.awt.image.WritableRaster;  public class membentukGambar {    public BufferedImage setGambarintrgb(BufferedImage gambar) {  BufferedImage gambar\_baru = new BufferedImage(gambar.getWidth(), gambar.getHeight(), BufferedImage.TYPE\_INT\_RGB);  //menjadikan sebuah gambar dalam tipe byte blue green red  Graphics2D graphics = gambar\_baru.createGraphics();  graphics.drawRenderedImage(gambar, null);  graphics.dispose();  return gambar\_baru;  }  public BufferedImage settnewrgb(BufferedImage gambarbaru, int[][] red, int[][] green, int[][] blue) {  int patokan = gambarbaru.getWidth() / 2;  WritableRaster raster = gambarbaru.getRaster();    for (int i = 0,d=patokan; i < 72; i++,d++) {  for (int j = 0,e=patokan; j < 72; j++,e++) {    int R = red[i][j];  int G = green[i][j];  int B = blue[i][j];  //System.out.println(B);  int gray = R << 16 | G << 8 | B;  gambarbaru.setRGB(d, e, gray);  }  }  return gambarbaru;  }  } |
| **saveGambar.java** |
| package metode;  import java.awt.image.BufferedImage;  import java.io.File;  import javax.imageio.ImageIO;  import javax.swing.JOptionPane;  public class saveGambar {  public boolean menciptakan\_gambar(BufferedImage gmbr\_bmp, File file, String ext) {  try {  file.delete();  // menghapus file yang ada dengan nama yang sama  ImageIO.write(gmbr\_bmp, ext, file);  //menciptakan sebuah file sesuai dengan gambar,ekstensi,dan file  return true;  } catch (Exception e) {  JOptionPane.showMessageDialog(null,  "File tidak berhasil di simpan!", "Error", JOptionPane.ERROR\_MESSAGE);  return false;  }  }  } |

* + 1. HASIL DAN PEMBAHASAN

1. Implementasi Program

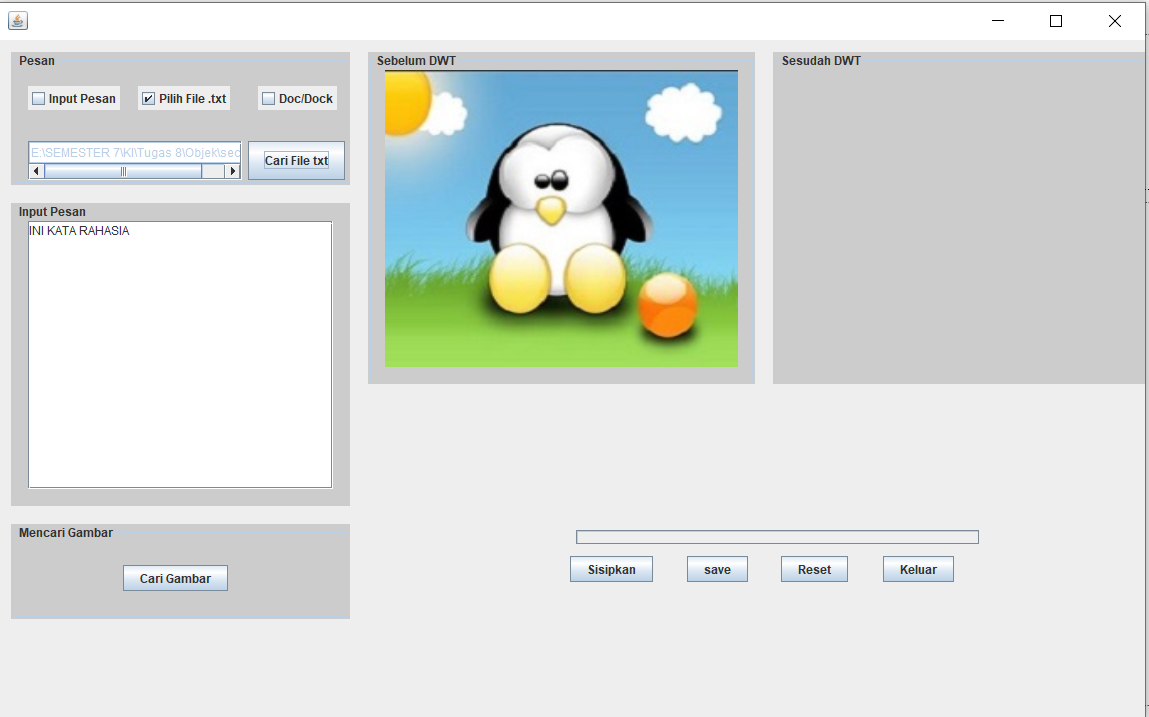
Aplikasi Steganografi untuk penyisipan sebuah teks dalam sebuah gambar dengan menggunakan metode Discrete Wavelet Transform (DWT) ini dikembangkan dengan menggunakan bahasa pemrograman java dan menggunakan NetBeans IDE 8.2. berikut adalah hasil dari aplikasi Steganografi dengan metode Discrete Wavelet Transform (DWT):

* Menu Awal



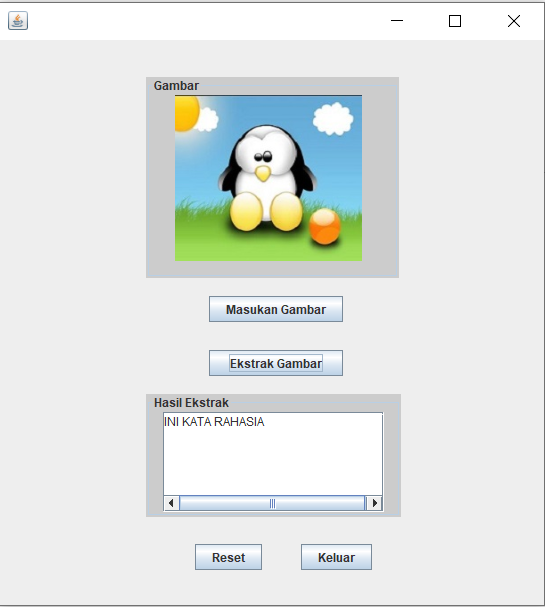
Pada aplikasi ini menu utama nya terdiri dari menu sisipkan dan ekstrak. Menu sisipkan digunakan untuk menyisipkan text pada gambar, kemudian menyimpan gambar yang telah disisipkan teks tersebut. Pada menu Ekstrak digunakan untuk mengekstrak sebuah gambar yang telah disisipkan suatu teks dan kemudian menampilkan teks yang telah disisipkan.

* Menu Sisipkan



Pada menu sisipkan ini pertama dapat menyisipkan pesan berupa sebuah teks yang diinputkan langsung atau sebuah file berupa file .txt atau file .docx. kemudian selanjutnya memilih gambar yang akan dimasukan sebuah pesan, lalu gambar tersebut akan muncul pada kolom “sebelum DWT’. Lalu selanjutnya terdapat button sisipkan untuk menyisipkan pesan yang telah diataur sebelumnya. Kemudian gambar yang telah disipkan pesan ini akan tampil pada kolom “sesudah DWT’. Setelah disispkan teks selanjutnya gambar tersebut dapat di disimpan.

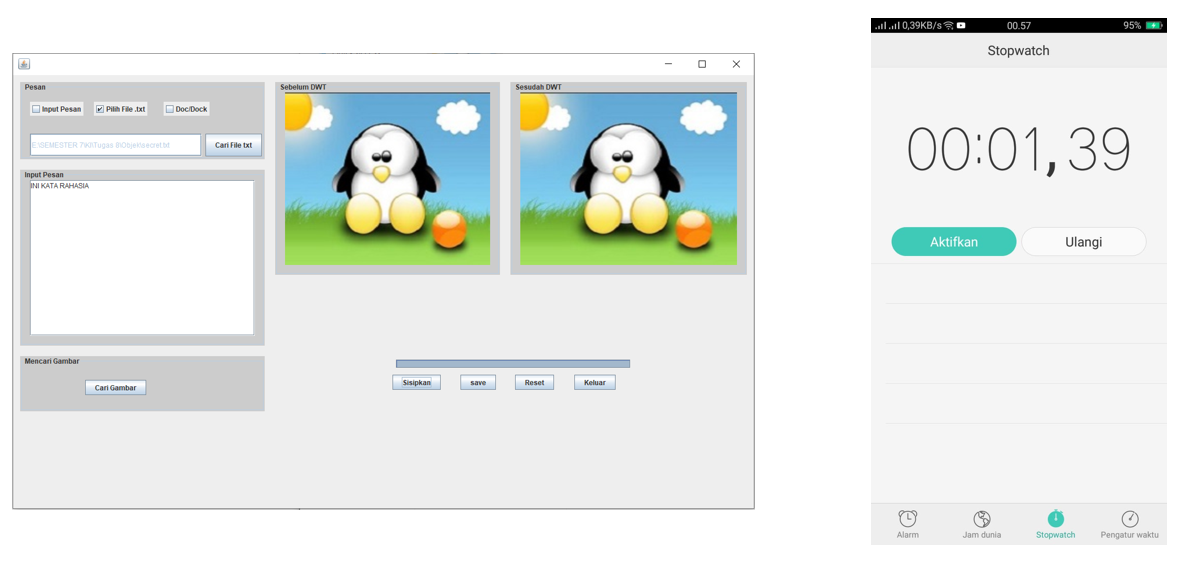
* Menu Ekstrak



Pada menu ekstrak ini digunakan untuk mengektrak teks yang telah di sisipkan pada gambar. Jadi jika gambar yang dimasukan terdapat teks yang disisipkan maka akan muncul pada kolom hasil ekstrak.

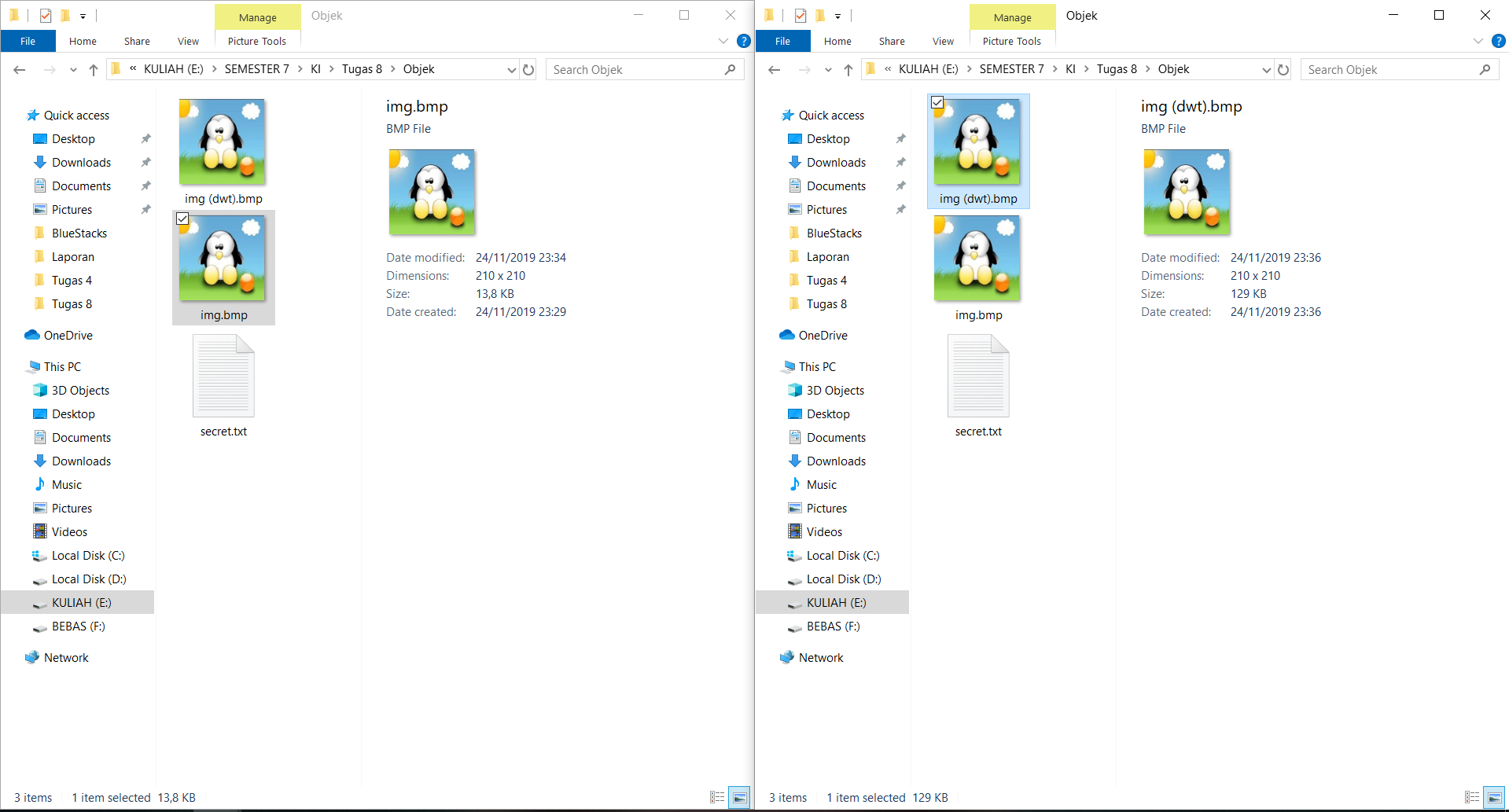
1. Percobaan Paramater Waktu, Size, dan Warna

* Parameter Waktu



Analisis : Pengujian pada parameter waktu dilakukan menggunakan stopwatch pada smartphone. Hal tersebut dilakukan pada waktu penyisipan teks ke dalam gambar pada aplikasi steganografi dwt.

* Parameter Size

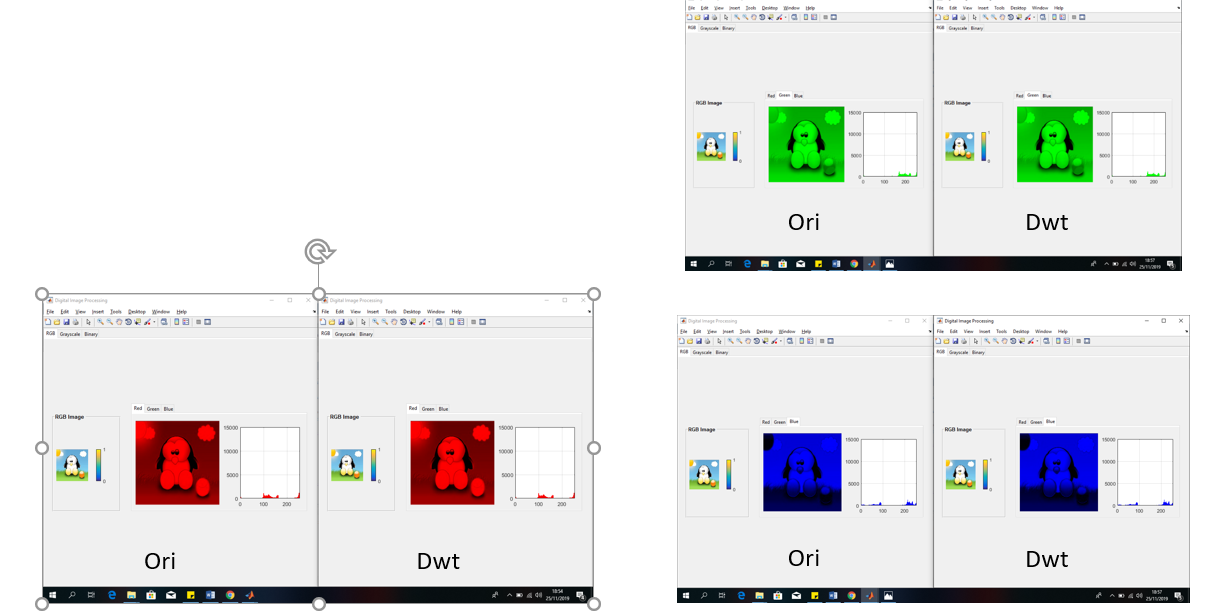


Analisis : Pada pengujian parameter size bisa dilihat pada gambar … bahwa perubahan ukuran terjadi pada saat menyisipkan teks kedalam gambar. Gambar original sebesar 13.8kb, setelah dilakukan penyisipan ukurannya menjadi 129kb.

* Parameter Warna

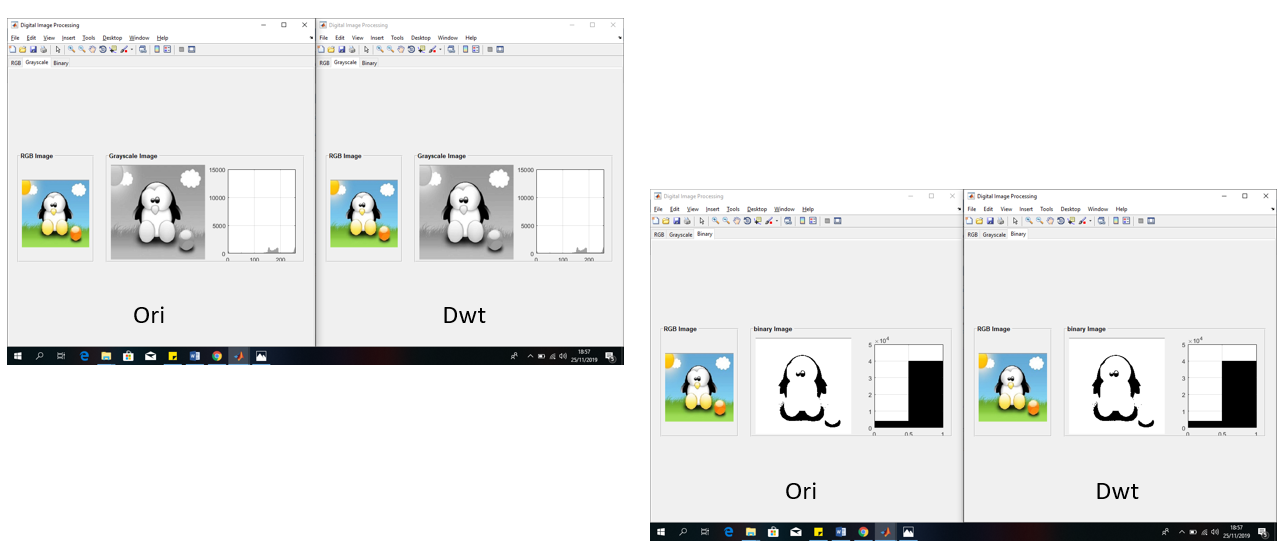
Pada pengujian parameter warna dilakukan pada tools matlab menggunakan Bahasa pemrograman …

1. RGB (Red, Green, Blue)



Analisis : Pada pengujian parameter warna RGB dengan menggunakan tools matlab dapat dilihat hasilnya grafik warna nya sama.

1. Grayscale dan Binary (Black White)



Analisis : Pada pengujian parameter warna Grayscale dan Binary (Black and White) dengan menggunakan tools matlab dapat dilihat hasilnya grafik warna nya sama.

**KESIMPULAN**

Steganografi (steganography) adalah ilmu dan seni menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui. Discrete Wavelet Transform (DWT) adalah salah satu metode yang digunakan dalam pengolahan citra digital. DWT dapat digunakan untuk transformasi citra dan kompresi citra. Selain untuk pengolahan citra (gambar), metode DWT dapat juga diterapkan pada bidang steganografi.pembuatan aplikasi Steganografi dengan penyisipan sebuah teks dalam sebuah gambar dengan menggunakan metode Discrete Wavelet Transform (DWT) ini bertujuan untuk memenuhi suatu aspek dalam keamanan informasi yaitu aspek Confidentiality dan privacy. Karena dengan menyisipkan suatu teks pada suatu gambar untuk memberikan suatu informasi maka pihak yang bukan ditujukan untuk menerima pesan tersebut tidak akan menyadari bahwa dalam sebuah gambar yang dianggap biasa terdapat suatu informasi yang telah disisipkan