**ANALISIS FAKTOR *ROBUSTNESS* DAN *FIDELITY* PADA METODE *LEAST SIGNIFICANT BIT*, *DISCRETE COSINES TRANSFORM*, DAN *DISCRETE HAAR WAVELET TRANSFORM* DALAM IMPLEMENTASI STEGANOGRAFI PADA CITRA DIGITAL**

**TUGAS AKHIR**

Tugas Akhir ini sebagai salah satu syarat untuk memperoleh gelar sarjana

Informatika Universitas Pembangunan Nasional “Veteran” Yogyakarta

****

Disusun Oleh:

**Ardian Tri Kusuma**

**123160035**

**PROGRAM STUDI INFORMATIKA**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK INDUSTRI**

**UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”**

**YOGYAKARTA**

**2020**

# HALAMAN JUDUL

**ANALISIS FAKTOR *ROBUSTNESS* DAN *FIDELITY* PADA METODE *LEAST SIGNIFICANT BIT*, *DISCRETE COSINES TRANSFORM*, DAN *DISCRETE HAAR WAVELET TRANSFORM* DALAM IMPLEMENTASI STEGANOGRAFI PADA CITRA DIGITAL**

**TUGAS AKHIR**

Tugas Akhir ini sebagai salah satu syarat untuk memperoleh gelar sarjana

Informatika Universitas Pembangunan Nasional “Veteran” Yogyakarta

****

Disusun Oleh:

**Ardian Tri Kusuma**

**123160035**

**PROGRAM STUDI INFORMATIKA**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK INDUSTRI**

**UNIVERSITAS PEMBANGUNAN NASIONAL “VETERAN”**

**YOGYAKARTA**

**2020**

# HALAMAN PENGESAHAN PEMBIMBING

# HALAMAN PENGESAHAN PENGUJI

# SURAT KARYA ASLI TUGAS AKHIR

# PERNYATAAN BEBAS PLAGIAT

# HAL PERSEMBAHAN

# ABSTRAK

# KATA PENGANTAR

# DAFTAR ISI

[**HALAMAN JUDUL i**](#_Toc46259890)

[**HALAMAN PENGESAHAN PEMBIMBING ii**](#_Toc46259891)

[**HALAMAN PENGESAHAN PENGUJI iii**](#_Toc46259892)

[**SURAT KARYA ASLI TUGAS AKHIR iv**](#_Toc46259893)

[**PERNYATAAN BEBAS PLAGIAT v**](#_Toc46259894)

[**HAL PERSEMBAHAN vi**](#_Toc46259895)

[**ABSTRAK vii**](#_Toc46259896)

[**KATA PENGANTAR viii**](#_Toc46259897)

[**DAFTAR ISI ix**](#_Toc46259898)

[**DAFTAR GAMBAR x**](#_Toc46259899)

[**DAFTAR TABEL xi**](#_Toc46259900)

[**DAFTAR MODUL PROGRAM xii**](#_Toc46259901)

[**BAB I PENDAHULUAN** 1](#_Toc46259902)

[**12.1** **Latar Belakang** 1](#_Toc46259903)

[**1.2** **Perumusan Masalah** 3](#_Toc46259904)

[**1.3** **Batasan Masalah** 3](#_Toc46259905)

[**1.4** **Tujuan Penelitian** 3](#_Toc46259906)

[**1.5** **Manfaat Penelitian** 4](#_Toc46259907)

[**1.6** **Metodologi Penelitian dan Pengembangan Sistem** 4](#_Toc46259908)

[**1.6.1** **Metodologi Penelitian** 4](#_Toc46259909)

[**1.6.2** **Metode Pengembangan Sistem** 5](#_Toc46259910)

[**1.7** **Sistematika Penulisan** 7](#_Toc46259911)

[**BAB II TINJAUAN PUSTAKA** 8](#_Toc46259912)

[**13.1** **Citra Digital** 8](#_Toc46259913)

[**13.1.1** **Ruang Warna RGB** 9](#_Toc46259914)

[**13.2** **Steganografi** 11](#_Toc46259915)

[**13.2.1** ***Least Significant Bit*** 11](#_Toc46259916)

[**13.2.2** ***Discrete Cosine Transform*** 12](#_Toc46259917)

[**13.2.3** ***Discrete Haar Wavelet Transform*** 14](#_Toc46259918)

[**13.2.4** **PengujianKualitas MetodeSteganografi** 15](#_Toc46259919)

[**13.3** **Python** 18](#_Toc46259920)

[**13.4** **Tinjauan Literatur** 19](#_Toc46259921)

[**BAB III METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM** 21](#_Toc46259922)

[**14.1** **Metodologi Penelitian** 21](#_Toc46259923)

[**14.1.1** **Identifikasi Masalah** 22](#_Toc46259924)

[**14.1.2** **Studi Pustaka** 22](#_Toc46259925)

[**14.2** **Metodologi Pengembangan Sistem** 22](#_Toc46259926)

[**14.2.1** **Komunikasi** 24](#_Toc46259927)

[**14.2.2** **Quick Design** 26](#_Toc46259928)

[**BAB IV HASIL, PENGUJIAN, DAN PEMBAHASAN** 41](#_Toc46259929)

# DAFTAR GAMBAR

[**Gambar 2.1.** Contoh tata letak pixel dalam matriks 9](#_Toc46260623)

[**Gambar 2.2** Warna RGB 10](#_Toc46260624)

[**Gambar 2.3.** Ilustrasi penyisipan bit LSB 12](#_Toc46260625)

[**Gambar 2.4.** Dekomposisi Wavelet tiga tingkat 15](#_Toc46260626)

[**Gambar 2.5.** DWT dua dimensi 15](#_Toc46260627)

[**Gambar 3.1** Alur Tahap Penelitian 21](#_Toc46260628)

[**Gambar 3.2** Kerangka Kerja Tahap Pengembangan Sistem 23](#_Toc46260629)

[**Gambar 3.3** Perancangan Arsitektur Sistem 26](#_Toc46260630)

[**Gambar 3.4** Flowchart Utama Sistem 27](#_Toc46260631)

[**Gambar 3.5** Flowchart Proses Encode 28](#_Toc46260632)

[**Gambar 3.6** Flowchart sub proses encode LSB 29](#_Toc46260633)

[**Gambar 3.7** Flowchart sub proses DCT 30](#_Toc46260634)

[**Gambar 3.8** Flowchart sub proses Inverse DCT 32](#_Toc46260635)

[**Gambar 3.9** Flowchart sub proses DWT 33](#_Toc46260636)

[**Gambar 3.10** Ilustrasi subband DWT 34](#_Toc46260637)

[**Gambar 3.11** Flowchart sub proses Inverse DWT 35](#_Toc46260638)

[Gambar 3.12 Wireframe Tampilan Awal 35](#_Toc46260639)

[Gambar 3.13 Wireframe Tampilan Encode 36](#_Toc46260640)

[Gambar 3.14 Wireframe Tampilan Hasil Encode 37](#_Toc46260641)

[Gambar 3.15 Wireframe Tampilan Decode 38](#_Toc46260642)

[Gambar 3.16 Wireframe Tampilan Identifikasi 39](#_Toc46260643)

# DAFTAR TABEL

[**Tabel 2.1.** Tinjauan Literatur 19](#_Toc46260648)

[**Tabel 2.2.** Lanjutan Tinjauan Literatur 20](#_Toc46260649)

# DAFTAR MODUL PROGRAM

# BAB I PENDAHULUAN

## Latar Belakang

Pentingnya kerahasiaan suatu informasi telah menjadi suatu perhatian tersendiri dari masa ke masa. Manusia berusaha mencari cara bagaimana merahasiakan informasi terhadap pihak yang dianggap tidak berhak untuk mengetahuinya. Berbagai cara telah dilakukan oleh bangsa – bangsa kuno untuk merahasiakan informasi, karena informasi yang jatuh ke orang yang tidak berhak akan menimbulkan kerugian. Sebagai contoh Julius Caesar, kaisar romawi, telah menggunakan suatu metode untuk mengacak pesan sebelum pesan itu dikirimkan kepada penerimanya. Tujuannya adalah agar orang yang tidak berhak menerima pesan tersebut tidak dapat membacanya (Tarigan, 2015).

Zaman sekarang informasi tidak hanya dapat disandikan, tetapi dapat juga disisipkan kedalam media digital. Teknik menyisipkan pesan dikenal dengan nama steganografi. Steganografi sebagai ilmu dan seni untuk menyembunyikan informasi sehingga informasi yang bersifat rahasia tidak dapat diketahui oleh orang lain, kecuali pengirim dan penerima (Atoum, Ibrahim, & M. Ahmad, 2012). Proses steganografi biasanya melibatkan penyandian atau kriptografi. Proses yang dilakukan yaitu dengan enkripsi *plaintext* terlebih dahulu menjadi *Byte cipher* atau pesan rahasia. Kemudian *Byte cipher* disisipkan pada media digital berupa teks, audio, citra atau protocol.

Beberapa penelitian telah dilakukan berkaitan dengan pengamanan data dengan menerapkan beberapa algoritma. Salah satunya adalah algoritma *Least Significant Bit*. Dalam penelitian Danuputri & Kom (2018) menggunakan algoritma *Least Significant Bit* yang diperkuat dengan adanya algoritma *Vignere* *Key* untuk melakukan enkripsi terhadap dokumen berekstensi \*.doc, \*.docx, \*.xls, \*.xlsx, \*.txt, dan \*.pdf untuk dihasilkan plaintext yang kemudian menggunakan *Least Significant Bit* untuk menyisipkannya ke dalam gambar. Penelitian serupa terkait algoritma *Least Significant Bit* juga pernah dilakukan oleh Anselmus (2012) untuk melakukan penyisipan gambar lain yang bersifat identitas untuk melindungi hak cipta dari suatu gambar dengan menggabungkan algoritma *Least Significant Bit* dengan *Redundant Pattern Encoding.* Penelitian lain terkait algoritma *Least Significant Bit* juga pernah dilakukan oleh Devi Rahmadita, dkk (2016) yang menggunakan algoritma tersebut untuk menyembunyikan pesan ke dalam media video sebagai *cover object* dengan hasil uji yang memuaskan. Namun hasil tersebut akan terpengaruh jika *stego object* diserang atau dimanipulasi. Selain itu, terdapat juga penelitian oleh Garno (2017) yang mengkombinasikan metode *Discrete Cosines Transform* danInterpolasi *Bilinear* untuk menyembunyikan pesan. Pada penelitian tersebut Garno berhasil menyisipkan pesan ke dalam citra digital, namun terjadi distorsi yang cukup signifikan pada *stego object* sehingga nilai hasil uji kurang memuaskan.

Penelitian-penelitian tersebut telah menggunakan algoritma-algoritma steganografi yang mempunyai kelebihan masing-masing. Dengan banyaknya metode-metode yang dapat digunakan, maka akan sangat membantu pengguna dalam penyembunyian pesan. Namun, melihat dari kelebihan saja tidaklah cukup. Perlu diketahui bahwa stego object tidak selamanya aman dan tidak menimbulkan kecurigaan. Ketika stego object dikirimkan menggunakan perantara pihak ketiga, ada kemungkinan stego object tersebut rusak atau isinya telah dilihat oleh pihak lain tanpa sepengetahuan pembuat. Hal tersebut terjadi karena setiap metode memiliki tingkat *robustness* (ketahanan) dan *fidelity* (ketidaknampakan perbedaan) yang rendah.

Dari permasalahan yang telah dipaparkan, maka diperlukan perbandingan metode-metode tersebut sehingga dapat diketahui metode mana yang memiliki tingkat *robustness* dan *fidelity* yang baik untuk digunakan dalam steganografi maupun watermarking pada suatu objek. Dalam penelitian ini, metode yang akan diteliti adalah *Least* *Significant* *Bit*, *Discrete* *Cosines* *Transform*, *Discrete* *Haar* *Wavelet* *Transform*, serta gabungan dari algoritma tersebut yang menghasilkan sebuah *modified* *algorithm* yang diharapkan dapat meningkatkan tingkat *robustness* dan *fidelity* dari *stego* *object* yang dibuat.

Oleh karena itu disusunlah penelitian dengan judul “Analisis Faktor *Robustness* dan *Fidelity* pada Metode *Least Significant Bit*, *Discrete Cosines Transform*, dan *Discrete Haar Wavelet Transform* dalam Implementasi Steganografi pada Citra Digital”.

## Perumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, maka dapat diambil rumusan masalah dalam penelitian ini adalah bagaimana perbandingan kualitas *robustness* pada *stego object* setelah dilakukan serangan pada *stego* *object* yang menggunakan metode *Least Significant Bit*, *Discrete Cosine Transform*, dan *Discrete Haar Wavelet Transform* serta bagaimana perbandingan tingkat *fidelity* dari *stego object* setelah disisipkan *hidden object.*

## Batasan Masalah

Pada penelitian ini, permasalahan dibatasi sebagai berikut:

1. Citra yang digunakan menggunakan ekstensi yang umum digunakan yaitu \*.jpg, \*.png, dan \*.bmp.
2. *Hidden* *object* atau *embedded* *message* berupa teks dan citra lain.
3. Penelitian ini lebih berfokus pada tingkat *robustness* dan *fidelity* dari *stego object* sehingga *hidden object* tidak perludienkripsi*/*disandikan*.*

## Tujuan Penelitian

Tujuan pada penelitian ini adalah sebagai berikut:

1. Mengetahui hasil identifikasi uji kualitas *robustness stego object* setelah diimplementasikan metode *Least Significant Bit*, *Discrete Cosine Transform*, dan *Discrete Haar Wavelet Transform* dan manipulasi *stego object.*
2. Mengetahui hasil identifikasi tingkat *fidelity stego object* setelah diimplementasikan metode *Least Significant Bit*, *Discrete Cosine Transform*, dan *Discrete Haar Wavelet Transform* dan menentukan metode manakah yang paling baik dalam menyembunyikan *hidden object*.

## Manfaat Penelitian

Manfaat yang dapat diperoleh dari penelitian ini adalah:

* + 1. Dapat mengetahui kelebihan serta kelemahan pada metode *Least Significant Bit*, *Discrete Cosine Transform*, dan *Discrete Haar Wavelet Transform* berdasarkan faktor *robustness* dan *fidelity* pada *stego* *object*.
    2. Dapat menjadi referensi dalam pemilihan metode steganografi yang dapat digunakan dalam kehidupan sehari-hari sehingga tidak perlu khawatir mengirimkan citra stego melalui perantara pihak ketiga.

## Metodologi Penelitian dan Pengembangan Sistem

Metodologi penelitian berisi langkah-langkah yang digunakan dalam penelitian ini agar terstruktur dengan baik. Pengembangan system merupakan bagian dari metode penelitian yang digunakan pada penelitian ini.

### Metodologi Penelitian

Metode yang digunakan dalam penelitian ini terdapat tiga tahap utama, yaitu tahap pendahuluan, tahap perancangan dan pembuatan prototype sistem, dan tahap penyelesaian. Tahap pendahuluan terdiri dari identifikasi masalah dan tinjauan pustaka. Tahap perancangan (*Quick Design*) dan pembuatan *prototype* sistem dijelaskan terpisah pada sub bab 1.6.2 mengenai metode pengembangan sistem. Tahap penyelesaian berisi penyusunan laporan tugas akhir.

### Metode Pengembangan Sistem

Metode yang digunakan dalam pengembangan sistem ini adalah metode *Prototype* (purwarupa). Tahapan dalam metode *prototype* (Pressman, 2010) dapat dilihat pada gambar 1.1.



**Gambar 1.1** Metode Prototyping

Metode *prototype* cocok digunakan untuk mengembangkan sebuah perangkat yang akan dikembangkan kembali. Metode ini dimulai dengan pengumpulan kebutuhan pengguna, dalam hal ini pengguna dari perangkat yang dikembangkan adalah peserta didik. Kemudian membuat sebuah rancangan kilat yang selanjutnya akan dievaluasi kembali sebelum diproduksi secara besar.

*Prototype* bukanlah merupakan sesuatu yang lengkap, tetapi sesuatu yang harus dievaluasi dan dimodifikasi kembali. Segala perubahan dapat terjadi pada saat *prototype* dibuat untuk memenuhi kebutuhan pengguna dan pada saat yang sama memungkinkan pengembang untuk lebih memahami kebutuhan pengguna secara lebih baik.

Berdasarkan model *prototype* yang telah digambarkan diatas, maka dapat diuraikan pembahasan masing-masing tahap dalam model tersebut adalah sebagai berikut:

1. *Communication*/Komunikasi

Perancang perangkat lunak melakukan analisis untuk menentukan kebutuhan perangkat lunak yang saat itu diketahui dan untuk menggambarkan area-area dimana definisi lebih jauh untuk iterasi selanjutnya.

1. *Quick Plan and Modeling Quick Design*/Pemodelan Rancangan Cepat

Pada bagian ini akan membuat perencanaan dan pemodelan secara cepat berdasarkan kebutuhan yang telah diidentifikasi pada tahap sebelumnya. Perencanaan yang akan dilakukan dapat berupa pembagian tugas dan rencana kerja. Sedangkan dari sisi pemodelan akan membuat desain model yang merepresentasikan aspek-aspek yang diinginkan dari pelanggan.

1. *Construction of Prototype*/Pembuatan Prototipe

Dalam pembuatan rancangan cepat berdasarkan pada representasi aspek-aspek perangkat lunak yang akan terlihat oleh *end user* ( tampilan antarmuka ). Rancangan cepat merupakan dasar untuk memulai konstruksi pembuatan prototipe.

1. *Deployment Delivery and Feedback*/Penyerahan dan Pemberian Umpan Balik

Prototipe kemudian dipresentasikan untuk mengevaluasi prototipe yang telah dibuat sebelumnya dan memberikan umpan balik yang akan digunakan untuk memperbaiki spesifikasi kebutuhan. Iterasi akan terjadi saat pengembang melakukan perbaikan prototipe tersebut.

## Sistematika Penulisan

Sistematika penulisan yang digunakan dalam menyusun laporan penelitian ini adalah sebagai berikut :

**BAB I Pendahuluan**

Pada bagian ini membahas tentang latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

**BAB II Tinjauan Pustaka**

Tinjauan pustaka memuat tentang dasar teori yang digunakan untuk analisis dan perancangan sistem serta implementasi pada penelitian ini. Selain itu juga sebagai bahan referensi dan pondasi untuk memperkuat argumentasi dalam penelitian ini. Teori-teori yang sesuai dengan penelitian ini antara lain steganografi*,* *Least Significant Bit*, *Discrete Cosines Transform ,*dan *Discrete Haar Wavelet Transform*, dan *Peak Signal to Noise Ratio*.

**BAB III Metode Penelitian dan Pengembangan Sistem**

Pada bagian ini akan membahas mengenai analisa dan perancangan sistem hingga implementasi system yang akan dibangun.

**BAB IV Hasil, Pengujian dan Pembahasan**

Pada bab ini akan menyajikan hasil penelitian yang berisi hasil implementasi dari perancangan yang telah dibuat pada bab sebelumnya. Selain itu berisi pengujian terhadap hasil penelitian beserta pembahasannya.

**BAB V Penutup**

Pada bagian ini berisi kesimpulan dari hasil penelitian dan saran yang diajukan oleh penulis untuk pengembangan penelitian selanjutnya.

# BAB II TINJAUAN PUSTAKA

## Citra Digital

Secara harafiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). Ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh oleh alat-alat optik, misalnya mata pada manusia, kamera, pemindai (*scanner*), dan sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam (Munir, 2004).

Agar dapat diolah dengan dengan komputer digital, maka suatu citra harus direpresentasikan secara numerik dengan nilai-nilai diskrit. Representasi citra dari fungsi malar (kontinu) menjadi nilai-nilai diskrit disebut digitalisasi. Citra yang dihasilkan inilah yang disebut citra digital (*digital* *image*). Pada umumnya citra digital berbentuk empat persegi panjang, dan dimensi ukurannya dinyatakan sebagai tinggi x lebar (atau lebar x panjang). Citra digital yang berukuran NxM lazim dinyatakan dengan matriks yang berukuran N baris dan M kolom sebagai berikut:

|  |  |  |
| --- | --- | --- |
|  | ………………… | ( 2.1 ) |

Indeks baris (i) dan indeks kolom (j) menyatakan suatu koordinat titik pada citra, sedangkan f(i, j) merupakan intensitas (derajat keabuan) pada titik (i, j). Masing-masing elemen pada citra digital (berarti elemen matriks) disebut *image* *element*, *picture* *element* atau *pixel* atau pel. Jadi, citra yang berukuran NxM mempunyai NM buah *pixel*. Sebagai contoh, misalkan sebuah berukuran 256x256 *pixel* dan direpresentasikan secara numerik dengan matriks yang terdiri dari 256 buah baris (di-indeks dari 0 sampai 255) dan 256 buah kolom (di-indeks dari 0 sampai 255) seperti contoh berikut:



**Gambar 2.1.** Contoh tata letak pixel dalam matriks

*Pixel* pertama pada koordinat (0, 0) mempunyai nilai intensitas 0 yang berarti warna *pixel* tersebut hitam, *pixel* kedua pada koordinat (0, 1) mempunyai intensitas 134 yang berarti warnanya antara hitam dan putih, dan seterusnya.

### Ruang Warna RGB

Warna pada dasarnya merupakan hasil persepsi dari cahaya dalam spektrum wilayah yang terlihat oleh retina mata, dan memiliki Panjang gelombang antara 400nm sampai dengan 700nm. Ruang warna atau sering disebut model warna merupakan sebuah cara atau metode untuk mengatur, membuat, dan memvisualisasikan warna (Ford & Roberts, 1998).

Salah satu dari ruang warna adalah RGB. RGB sering digunakan didalam sebagian besar aplikasi komputer karena dengan ruang warna ini, tidak diperlukan transformasi untuk menampilkan informasi di layar monitor. Alasan diatas juga menyebabkan RGB banyak dimanfaatkan sebagai ruang warna dasar bagi sebagian besar aplikasi. Model warna RGB adalah model warna berdasarkan konsep penambahan kuat cahaya primer yaitu *Red*, *Green* dan *Blue*. Dalam suatu ruang yang sama sekali tidak ada cahaya, maka ruangan tersebut adalah gelap total. Tidak ada signal gelombang cahaya yang diserap oleh mata kita atau RGB (0,0,0). Apabila ditambahkan cahaya merah pada ruangan tersebut, maka ruangan akan berubah warna menjadi merah misalnya RGB (255,0.0), semua benda dalam ruangan tersebut hanya dapat terlihat berwarna merah. Demikian juga apabila cahaya diganti dengan hijau atau biru.

Apabila diberikan 2 macam cahaya primer dalam ruangan tersebut seperti (merah dan hijau), atau (merah dan biru) atau (hijau dan biru), maka ruangan akan berubah warna masing-masing menjadi kuning, atau magenta, atau cyan. Warna-warna yang dibentuk oleh kombinasi dua macam cahaya tersebut disebut warna sekunder. Warna tersier adalah warna yang hanya dapat terlihat apabila terdapat tiga cahaya primer, jadi apabila dinonaktifkan salah satu cahaya, maka benda tersebut berubah warna. Contoh warna tersier adalah abu-abu, putih.



**Gambar 2.2** Warna RGB

Pada perhitungan dalam program-program komputer model warna direpresentasi dengan nilai komponennya, seperti dalam RGB (rgb) masing-masing nilai antara 0 hingga 255 sesuai dengan urusan masing-masing yaitu pertama *red*, kedua *green* dan ketiga adalah nilai *blue* dengan demikian masing-masing komponen ada 256 tingkat. Apabila dikombinasikan maka ada 256 x 256 x 256 atau 16777216 kombinasi warna RGB yang dapat dibentuk.

## Steganografi

Steganografi berasal dari bahasa Yunani yaitu kata *stegos* yang berarti sembunyi dan *graphia* yang berarti tulisan. Steganografi secara umum memiliki arti ilmu dan seni menyembunyikan suatu fakta untuk berkomunikasi. Dengan menggunakan steganografi, pesan rahasia dapat disisipkan ke dalam sebuah informasi yang tidak mencurigakan dan mengirimkannya tanpa ada yang mengetahui keberadaan dari pesan rahasia tersebut (Krenn, 2004). Dalam era digital ini, steganografi berarti penyisipan pesan dalam bentuk digital ke dalam media digital yang ada. Untuk steganografi terbentuk dari dua macam yaitu pesan digital atau sering disebut dengan *message* yang akan disisipkan dan media tempat penyisipan akan dilakukan. Media tempat penyisipan dapat berupa teks, gambar, suara, dan video. Untuk media gambar, gambar yang dijadikan sebagai citra penampung disebut dengan *cover* *image*. Penyembunyian pesan yang berupa teks maupun gambar ke dalam citra digital akan mempengaruhi kualitas citra tersebut.

### Least Significant Bit

Bit atau *binary digit* adalah unit dasar penyimpanan data di dalam komputer, nilai bit suatu data adalah 0 (nol) atau 1 (satu). Semua data yang ada pada komputer disimpan ke dalam satuan bit ini, termasuk gambar, suara, ataupun video hanya saja penerjemahan representasi bit pada masing-masing media yang tentunya akan berbeda. Seperti penjelasan pada bagian sebelumnya bahwa format pewarnaan pada media gambar juga menggunakan satuan bit dalam penyimpanannya. Sebagai contoh pewarnaan *monochrome* menggunakan 1 bit untuk merepresentasikan warna hitam atau putih, pewarnaan *grayscale* menggunakan 8 bit untuk merepresentasikan tingkat keabuan dan pada pewarnaan RGB menggunakan 24 bit (8 bit untuk *Red*, 8 bit untuk *Green*, 8 bit untuk *Blue*).

*Least Significant Bit* (LSB) adalah bagian dari barisan data biner yang mempunyai nilai paling tidak berarti atau paling kecil. Bit LSB letaknya di paling kanan pada barisan biner, karena nilai 1 bit LSB pada barisan biner hanya merepresentasikan nilai 1 desimal, maka bit ini dianggap tidak berarti. Sehingga jika terjadi perubahan pada nilai bit LSB maka tidak akan terjadi perubahan secara signifikan.



**Gambar 2.3.** Ilustrasi penyisipan bit LSB

*Modified Least Significant Bit* (MLSB) atau modifikasi dari Algoritma LSB digunakan untuk meng-*encode* sebuah identitas ke dalam citra asli. MLSB menggunakan manipulasi beberapa bit-bit penyisip sebelum meng-encode pesan tersebut (Zaher, 2011).

Modifikasi pesan dengan algoritma MLSB dimana bit pesan yang seharusnya 1 karakter memiliki nilai 8 bit ASCII akan dimodifikasi menjadi 5 bit (31 desimal). Pada algoritma ini karakter dan angka direpresentasikan dalam 5 bit sebelum disisipkan ke dalam citra asli dengan teknik LSB.

### Discrete Cosine Transform

*Discrete Cosine Transform* (DCT) Transformasi DCT merupakan salah satu *transform* *coding* yang akan merubah *byte* dari domain spasial menjadi domain frekuensi dan memisahkan *byte* data tersebut menjadi dua bagian, yaitu frekuensi tinggi (koefisien DC) dan frekuensi rendah (koefisien AC). Pada DCT, koefisien DC digunakan sebagai tempat penyisipan pesan. Hal ini dikarenakan koefisien DC memiliki kapasitas persepsi yang lebih tinggi dari pada koefisien AC sehingga proses penyisipan tidak akan mengubah kualitas gambar secara visual. Selain itu, sinyal proses dan distorsi gambar memiliki pengaruh yang lebih rendah terhadap koefisien DC daripada koefisien AC (Reva, Susilo, & Purwandari, 2016).

Transformasi citra dilakukan dengan menggunakan DCT (*Discrete Cosine Transform*), sehingga dapat dikatakan bahwa penyisipan dilakukan pada ranah DCT. Penyisipan dilakukan terhadap citra bitmap dengan kedalaman warna 24 bit. DCT digunakan untuk metransformasikan nilai intensitas blok 8x8 pikselnya yang berurutan dari *image* menjadi 64 koefisien DCT kedalam frekunesi dasarnya, diubah koefisien- koefisiennya dan kemudian ditransformasikan kembali dengan IDCT (*Inverse Discrete Cosine Transform* ). Setiap basis matriks dikarakteristikan olehfrekuensi spatial horizontal dan vertical. Dalam konteks citra, hal ini menunjukkan tingkat signifikasi secara perseptual, artinya basis fungsi dengan frekuensi rendah memiliki sumbangan yang lebih besar bagi perubahan penampakan citra dibandingkan basis fungsi yang memiliki frekuensi tinggi. Nilai konstanta basis fungsi yang terletak di bagian kiri atas sering disebut basis fungsi DC, dan DCT koefisien yang bersesuaian dengannya disebut koefisien DC (DC coeficient). Masukan proses DCT berupa matrik NxN. Persamaan DCT untuk matrik berukuran NxN dapat dituliskan sebagai berikut :

|  |  |  |
| --- | --- | --- |
|  | …….. | ( 2.2 ) |

Sedangkan persamaan untuk invers DCT (IDCT) adalah sebagai berikut:

|  |  |  |
| --- | --- | --- |
|  | …….. | ( 2.3 ) |

Dimana ;

T(i,j) = Data pada domain frekuensi

*pixel*(x,y) = Data pada domain ruang secara pixel

Output dari fungsi DCT adalah nilai komponen frekuensi tertentu dan output dari fungsi ini ditentukan oleh dua parameter, yaitu i dan j. Begitu pula pada proses IDCT, hanya saja proses tersebut untuk mengembalikan citra semula.

### Discrete Haar Wavelet Transform

Secara umum Transformasi Wavelet Diskrit merupakan dekomposisi citra pada frekuensi subband citra tersebut. Komponen subband transformasi wavelet dihasilkan dengan cara penurunan level dekomposisi. Implementasi Transformasi Wavelet Diskrit dapat dilakukan dengan melewatkan sinyal melalui sebuah *lowpass* *filter* dan *highpass* filter dan melakukan *downsampling* pada keluaran masing-masing filter. *Highpass* filter digunakan untuk menganalisis frekuensi tinggi dan *lowpass* *filter* digunakan untuk menganalisis frekuensi rendah (Reva, Susilo, & Purwandari, 2016). Proses dekomposisi dapat melalui satu atau lebih tingkatan. Dekomposisi satu tingkat dapat ditulis dengan persamaan sebagai berikut:

|  |  |
| --- | --- |
| …...……………………………………………… | ( 2.4 ) |
| ..……………………………………………..... | ( 2.5 ) |

Variabel ytinggi[k] dan yrendah[k] adalah hasil dari *highpass* *filter* dan *lowpass* filter, merupakan sinyal asal, adalah *highpass* *filter*, dan adalah *lowpass* *filter*.



**Gambar 2.4.** Dekomposisi Wavelet tiga tingkat

Dengan menggunakan koefisien Transformasi Wavelet Diskrit ini maka dapat dilakukan proses *Inverse* *Discrete* *Wavelet* *Transform* (IDWT) untuk merekonstruksi menjadi sinyal asal.

|  |  |
| --- | --- |
| ……….. | ( 2.6 ) |

Proses rekonstruksi merupakan kebalikan dari proses dekomposisi sesuai dengan tingkatan pada proses dekomposisi. Transformasi wavelet diskrit dimensi dua dapat digambarkan seperti gambar 2.3.



**Gambar 2.5.** DWT dua dimensi

### Pengujian Kualitas Metode Steganografi

Terdapat kriteria-kriteria yang harus diperhatikan dalam penyembunyian data yaitu:

1. *Impercebility* yaitu keberadaan pesan rahasia tidak dapat dipersepsi secara indrawi.

2. *Fidelity* yaitu mutu citra penampung tidak jauh berubah

3. *Robustness* yaitu data yang disembunyikan harus tahan terhadap manipulasi yang dilakukan pada citra penampung (seperti pengubahan kontras, penajaman, pemampatan, rotasi, perbesaran gambar, pemotongan, enkripsi, dan sebagainya)

4. *Recovery* yaitu pesan dapat diekstrak kembali.

#### Uji Tingkat Robustness

*Robustness* merupakan salah satu kriteria dalam steganografi yang kurang penting dalam steganografi dikarenakan selama objek stego tidak terdapat perbedaan mencolok dan tidak menimbulkan kecurigaan. Namun kriteria ini tetap dapat dibilang penting apabila menyangkut objek tersembunyi di dalamnya. *Robustness* tidak kalah penting dikarenakan jika terdapat serangan pada objek stego, *hidden object* harus tetap tidak rusak dan dapat memenuhi kriteria *Recovery*.

Dalam pengujiannya, kriteria *robustness* dapat dilakukan dengan beragam cara yang juga disebut sebagai *steganalysis*. Salah satu metode yang dapat diterapkan adalah metode serangan StirMark. StirMark merupakan metode steganalysis yang menggabungkan berbagai teknik dasar dalam manipulasi citra seperti rotasi, pemotongan, *resampling*, *resizing*, dan kompresi pada citra watermark. Pada umumnya skema watermarking berhasil lolos dari manipulasi-manipulasi tersebut dan pada umumnya pula, skema watermark tidak dapat bertahan dari gabungan atau kombinasi dari manipulasi-manipulasi dasar tersebut. Hal ini menjadi dasar pembangunan dari metode StirMark. (Ferdian, 2006)

Dapat dikatakan dalam versi yang paling sederhana, StirMark membangkitkan sebuah proses *resampling*. Proses *resampling* ini dilakukan secara digital. Proses ini dapat diumpamakan dengan cara mencetak sebuah citra ke kertas dan melakukan scanning terhadap citra tersebut, dalam hal ini scanner yang digunakan diasumsikan berkualitas sangat tinggi, sehingga hampir tidak ada perubahan jika hasil resampling dibandingkan dengan sample aslinya.

#### Uji Tingkat Fidelity

*Fidelity* merupakan kriteria paling umum yang sangat penting dalam steganografi. Kriteria *fidelity* menunjukan besaran mutu stego object tidak jauh berubah dari sebelum dimasukkan *hidden* *object* maupun sesudahnya. Hal ini berkaitan tentang kecurigaan pihak lain jika melihat *stego* *object* dan demi keamanan objek yang disembunyikan maka kriteria *fidelity* perlu diperhatikan. Jika objek stego mempunyai nilai *fidelity* yang sangat rendah, kriteria lain seperti *impercebility* juga akan terpengaruh dikarenakan distorsi yang ada tentunya akan disadari oleh inderawi.

Pengujian fidelity umumnya menggunakan penghitungan PSNR dan MSE. *Peak Signal to Noise Ratio* (PSNR) merupakan sebuah parameter yang penting untuk mengukur kualitas proses pengolahan citra. PSNR adalah rasio antara intensitas maksimum citra dengan *Mean Square Error* (MSE) dari citra. Persamaan untuk menghitung nilai PSNR adalah sebgai berikut:

|  |  |
| --- | --- |
| …………………………………………………….... | ( 2.7 ) |

Dengan persamaan MSE adalah sebagai berikut:

|  |  |
| --- | --- |
| ………………………………………... | ( 2.8 ) |

Dalam penghitungan kualitas dua buah citra makin besar nilai PSNR maka makin kecil nilai MSE. Nilai MSE yang kecil mengindikasikan dua buah citra mempunyai sedikit perbedaan. Dari pengalaman empiris, citra dengan PSNR≥30 masih dapat dianggap berkualitas bagus, tetpi jika PSNR < 30 dikatakan kualitas citra terdegradasi secara signifikan (Munir, 2019).

## Python

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Python juga didukung oleh komunitas yang besar.

Python mendukung multi paradigma pemrograman, utamanya pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada python adalah sebagai bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa skrip meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa skrip. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi.

Python dikembangkan oleh Guido van Rossum pada tahun 1990 di *Stichting Mathematisch Centrum* (CWI), Amsterdam sebagai kelanjutan dari bahasa pemrograman ABC. Versi terakhir yang dikeluarkan CWI adalah 1.2. (Venners, 2003)

Tahun 1995, Guido pindah ke CNRI di Virginia Amerika sambil terus melanjutkan pengembangan Python. Versi terakhir yang dikeluarkan adalah 1.6. Tahun 2000, Guido dan para pengembang inti Python pindah ke BeOpen.com yang merupakan sebuah perusahaan komersial dan membentuk BeOpen PythonLabs. Python 2.0 dikeluarkan oleh BeOpen. Setelah mengeluarkan Python 2.0, Guido dan beberapa anggota tim PythonLabs pindah ke *Digital Creations*.

Saat ini pengembangan Python terus dilakukan oleh sekumpulan pemrogram yang dikoordinir Guido dan *Python Software Foundation*. *Python Software Foundation* adalah sebuah organisasi non-profit yang dibentuk sebagai pemegang hak cipta intelektual Python sejak versi 2.1 dan dengan demikian mencegah Python dimiliki oleh perusahaan komersial. Saat ini distribusi Python sudah mencapai versi 2.7.14 dan versi 3.6.3

## Tinjauan Literatur

Adapun beberapa penelitian yang telah dilakukan dan dijadikan referensi dalam penelitian berikut ini:

**Tabel 2.1.** Tinjauan Literatur

| **No** | **Penulis** | **Judul** | **Metode** | **Hasil** |
| --- | --- | --- | --- | --- |
| 1 | Anselmus Krisma Adi Kurniawan. 2012 | *Digital Watermarking* pada Gambar Digital dengan Metode *Redundant Pattern Encoding* | *Standart Least Significant Bit* dan *Redundant Pattern Encoding* | Citra *watermark* berhasil diimplementasikan dan tahan terhadap serangan transformasi *cropping*. |
| 2 | Devi Rahmadita, Dr. Ir. Bambang Hidayat, DEA dan I Nyoman Apraz Ramatryana, S.T., M.T. 2016 | Steganografi pada *Frame* *Video* *Stationer* menggunakan Metode *Least Signification Bit* | *Standart Least Significant Bit* | Penghitungan kualitas dengan metode PSNR mendapatkan hasil yang maksimal dan tetap memenuhi syarat dianggap bagus sebagai *stego* *object*. Namun nilai PSNR akan menurun ketika mendapat serangan dengan metode *Salt and Pepper Noise.* |

**Tabel 2.2.** Lanjutan Tinjauan Literatur

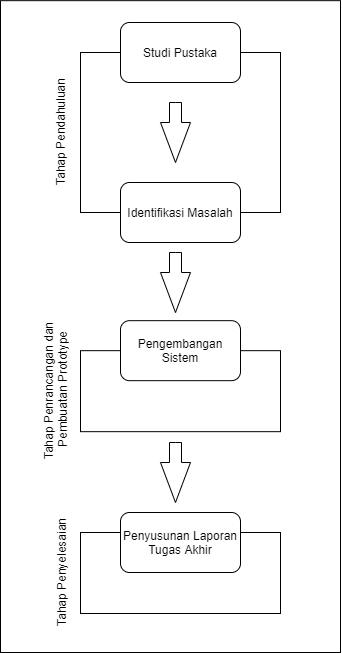
| 3 | Thomas Edyson Tarigan.  2015 | Algoritma MEoF (Modifikasi *End of File*) untuk Steganografi pada Citra Bitmap 24 Bit | *Modified End of File* | Algoritma Modifikasi *End of File* berhasil memperbaiki kelemahan algoritma *End of File*, yaitu dapat mengatasi peningkatan ukuran *file* dan perubahan kualitas citra *stego* dari citra asli |
| --- | --- | --- | --- | --- |
| 4 | Garno dan Arip Solehudin.  2017 | Teknik Steganografi dengan metode *Discrete Cosines Transform* (DCT) pada Citra Interpolasi *Bilinear* untuk Pengamanan Pesan | *Discrete Cosines Transform* danInterpolasi *Bilinear* | Dari penelitian yang telah dilakukan menghasilkan *stego* *object* dengan rata-rata belum mencapai nilai 40 db sehingga terbilang kurang bagus namun memiliki nilai MSE yang relatif kecil. |
| 5 | Chyquita Danuputri, M. Kom. 2018 | Pengamanan Data melalui *Cloud Computing* dengan Integrasi Steganografi *Least Significant Bit* dan Kriptografi *Vigenere Key* Berbasis *Android* | *Standart Least Significant Bit* dan enkripsi *Vigenere Key* | Pesan berhasil di sembunyikan pada citra digital dan diimplementasikan pada *cloud drive*. |

Penelitian ini akan menganalisis metode LSB, DCT, dan DWT dalam steganografi ditinjau dari faktor *robustness* dan *fidelity*. Hal yang membedakan dari penelitian sebelumnya adalah objek yang diteliti yang kebanyakan lebih fokus pada performa algoritma berdasarkan waktu eksekusi dan perbandingan visual hasil akhir.

# BAB III METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM

## Metodologi Penelitian

Metode penelitian berisi langkah-langkah yang digunakan dalam penelitian ini agar terstruktur dengan baik. Dengan sistematika ini proses penelitian dapat dipahami dan diikuti oleh pihak lain. Penelitian yang dilakukan untuk merancang sistem diperoleh dari pengamatan data-data yang ada. Metode pada penelitian ini terdapat tiga tahap, yaitu tahap pendahuluan, tahap pengembangan sistem, penyelesaian. Tahap pendahuluan terdiri dari identifikasi permasalahan dan tinjauan pustaka. Tahap perancangan dan pembuatan sistem berisi metode pengembangan sistem yang dibahas pada sub bab 3.2. Tahap penyelesaian berisi pembuatan laporan akhir. Adapun langkah-langkah yang tujuan dari penelitian ini dapat dilihat pada gambar 3.1.



**Gambar 3.1** Alur Tahap Penelitian

### Identifikasi Masalah

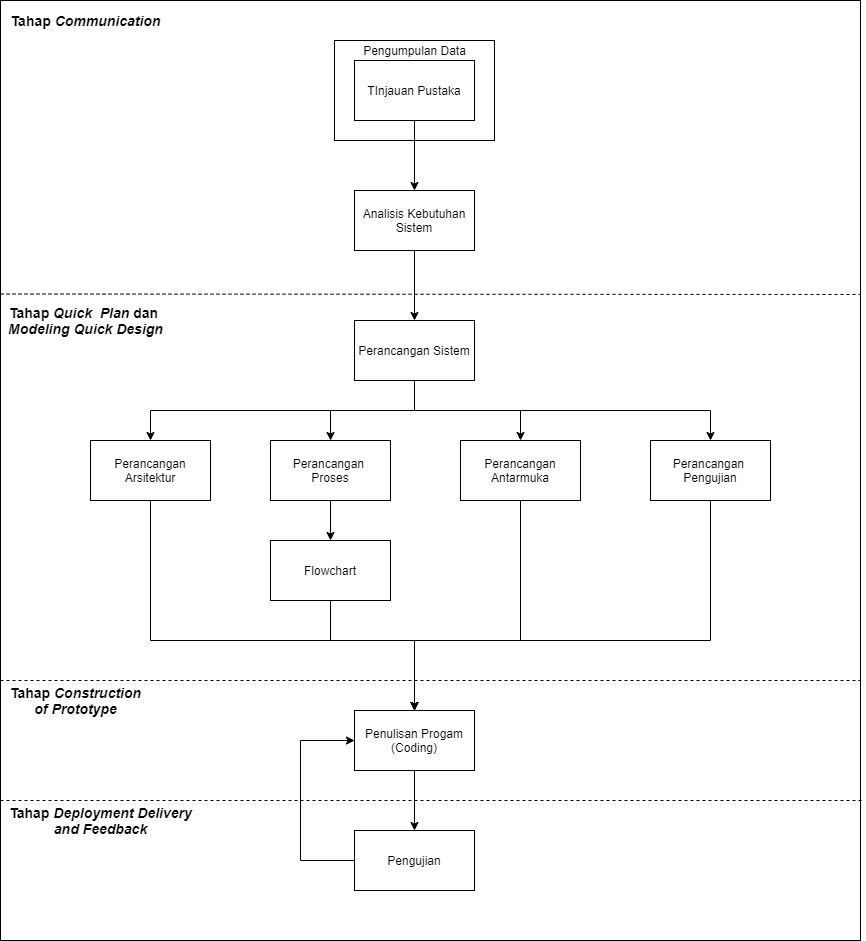
Tahap awal dalam penelitian ini adalah mengidentifikasi masalah yang akan dijadikan sebagai objek penelitian. Objek dari penelitian ini adalah citra digital yang merupakan salah satu informasi penting di era digital seperti ini. Permasalahan keamanan informasi yang terdapat pada citra digital adalah mengenai hak kepemilikan Citra digital. Watermarking merupakan salah satu teknik yang dapat digunakan untuk menyelesaikan permasalahan tersebut. Watermarking sendiri merupakan salah satu bagian dari implementasi seni menyembunyikan objek atau biasa disebut steganografi. Steganografi mempunyai banyak metode yang bisa digunakan. Dari metode-metode tersebut terdapat kelebihan dan kekurannya masing-masing.

### Studi Pustaka

Studi pustaka dilakukan untuk mempelajari teori-teori dan hasil penelitian sebelumnya yang berkaitan dengan masalah pada penelitian ini. Studi pustaka bertujuan untuk memperkuat materi pembahasan maupun rumus-rumus tertentu dalam menganalisa dan mendesain suatu sistem. Sumber-sumber yang di resensi seperti buku-buku, jurnal penelitian, artikel, thesis atau diktat perkuliahan. Tapi ini dilakukan untuk pencarian informasi mengenai steganografi, metode yang digunakan dalam steganografi, dan informasi lainnya untuk mendukung pengembangan sistem.

## Metodologi Pengembangan Sistem

Metode pengembangan sistem yang digunakan dalam penelitian ini adalah metode *Prototype* yang telah dikembangkan oleh Pressman. Metode *prototyping* menggunakan pendekatan pengembangan system dengan empat tahap utama yaitu komunikasi, *quick* *design*, pembuatan *prototype*, dan *deployment delivery and feedback* (umpan balik kepada pengguna). Pada penelitian ini hanya dilakukan sampai dengan tahap keempat yaitu tahap pembentukan prototype dan dilanjutkan dengan pengujian unit.



**Gambar 3.2** Kerangka Kerja Tahap Pengembangan Sistem

### Komunikasi

Pada tahap komunikasi, didefinisikan format seluruh perangkat lunak, identifikasi kebutuhan, dan garis besar aplikasi steganografi yang akan dibangun. Tahap ini dibagi menjadi tiga bagian yaitu identifikasi masalah, studi pustaka dan pengumpulan data, dan analisis kebutuhan sistem. Penjelasan tentang identifikasi masalah dan studi pustaka telah dijelaskan pada sub bab metode penelitian.

#### Pengumpulan Data

Pengumpulan data merupakan proses untuk mengetahui lebih dalam mengenai metode yang akan digunakan untuk dianalisis beserta metode pengujian yang akan dilakukan pada *output* metode-metode tersebut. Tujuan dari pengumpulan data tersebut untuk mendapatkan informasi mengenai hal-hal yang dibutuhkan untuk mendukung pembuatan sebuah sistem. Pengumpulan data pada sistem ini hanya dilakukan dengan studi pustaka untuk mengetahui pengembangan sistem yang digunakan untuk teknik steganografidan gambar yang digunakan untuk proses penyembunyian objek dalam penelitian ini.

#### Analisis Kebutuhan Sistem

Analisis kebutuhan sistem pada penelitian ini mencakup analisis kebutuhan fungsional dan non-fungsional. Kebutuhan fungsional merupakan kebutuhan yang berisikan proses-proses yang mampu dilakukan oleh sistem. Sedangkan kebutuhan non-fungsional merupakan kebutuhan yang menitikberatkan pada properti perilaku yang dimiliki oleh sistem. Kebutuhan fungsional dalam pengembangan sistem ini diantaranya sebagai berikut :

User dapat memasukkan *cover image* dan *hidden object*.

User dapat melihat *stego image* dan hasil ekstraksi dari *stego image*.

User dapat melihat hasil PSNR dan MSE baik dari *cover image* dan *stego image* maupun *stego image original* dengan *stego image* yang telah diserang.

Kebutuhan non-fungsional melibatkan beberapa perangkat yang mendukung sistem. Perangkat tersebut berupa perangkat keras (*hardware*) dan perangkat lunak (*software*). Kebutuhan perangkat tersebut diantaranya sebagai berikut :

Analisis Kebutuhan Perangkat Keras

Perangkat keras (*hardware*) yang dibutuhkan untuk pengembangan sistem ini adalah Laptop Asus X456UR yang memiliki spesifikasi sebagai berikut :

*Processor* Intel® Core™ i5 6200U *Processor* @2.30 GHz

*RAM* 8 GB

*Internal Drive* 1 TB HDD+ 512 GBSSD

*Display* 13.3" (1920x1080)

*Graphic* NVIDIA® GeForce® 930MX dengan 2GB DDR3 VRAM

Analisis Kebutuhan Perangkat Lunak

Perangkat lunak (*software*) yang dibutuhkan untuk pengembangan sistem ini diantaranya sebagai berikut :

OS Microsoft Windows 10 Pro

Python v3.8.1

OpenCV v4.2.0.32

Numpy v1.18.1

PyWavelets v1.1.1

Flask v1.1.1

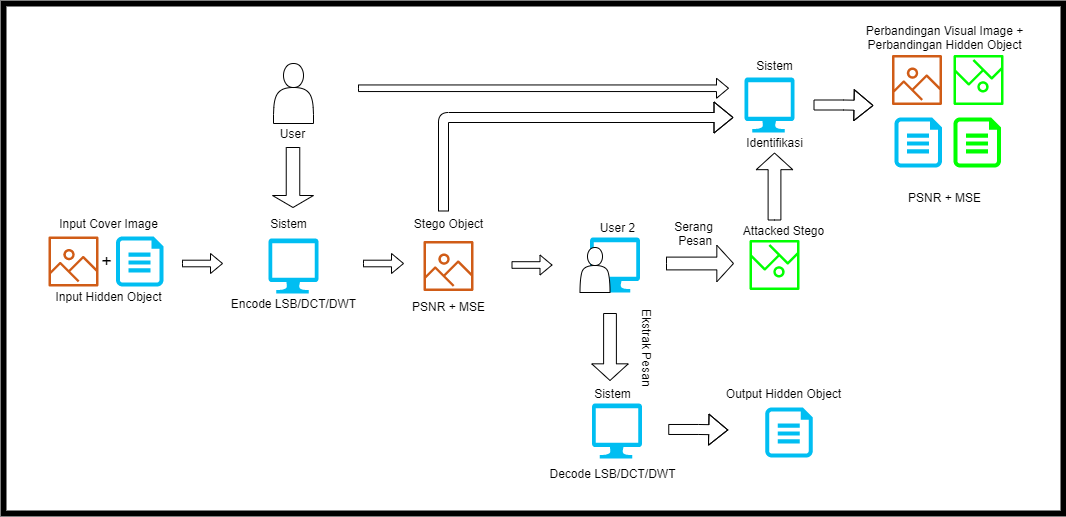
Jinja2 v2.11.1

### Quick Design

Pada tahap *quick* *design*, akan dibuat desain aplikasi steganografi secara umum untuk dikembangkan kembali. Tahap ini dibagai menjadi empat yaitu perancangan arsitektur, perancangan proses, perancangan antarmuka, dan perancangan pengujian.

#### Perancangan Arsitektur

Perancangan asitektur merupakan gambaran proses dari sistem yang akan dikembangkan. Arsitektur dari sistem yang akan dibangun dapat dilihat pada gambar 3.3.

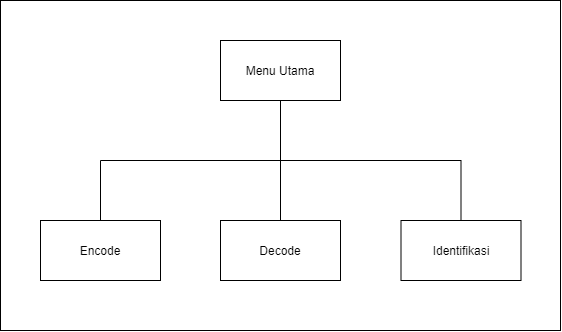


**Gambar 3.3** Perancangan Arsitektur Sistem

Pada arsitektur di atas, *user* akan memasukkan memiliki tiga pilihan yang dapat dilakukan pada sistem. Utamanya user akan memasukkan citra yang akan dijadikan citra *cover* dan *hidden object* dengan prosedur *encode*. Kemudian sistem akan memproses citra digital tersebut menjadi sebuah citra stego. Selanjutnya citra stego tadi dapat dikirimkan ke *user* 2 atau tetap digunakan oleh *user* pembuat citra stego tersebut. *User* pembuat maupun *user* 2 dapat melakukan ekstraksi pesan dari citra stego dengan prosedur *decode*. Prosedur ini akan mengekstrak pesan sesuai dengan algoritma yang digunakan didalam citra stego. *User* 2 dapat berlaku sebagai penerima sah dari pesan citra stego atau berlaku sebagai pihak ketiga yang akan menyerang pesan. Jika citra stego diserang maka *user* pembuat maupun *user* 2 dapat melakukan identifikasi pada sistem untuk menganalisa secara visual, tingkat perbedaan citra melalui PSNR dan MSE, dan ketahanan pesan pada citra stego yang ada.

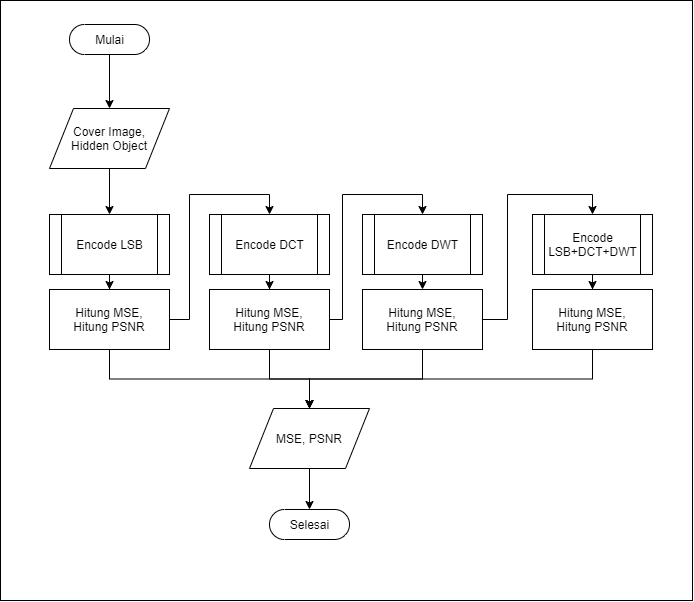
#### Perancangan Proses

Tahapan ini terdiri dari *flowchart* dari sistem yang akan dikembangkan yang terdiri dari *flowchart* proses steganografidan *flowchart* sub proses yang menggambarkan langkah-langkah pada steganograficitra digital dengan metode LSB, DCT, dan DWT.



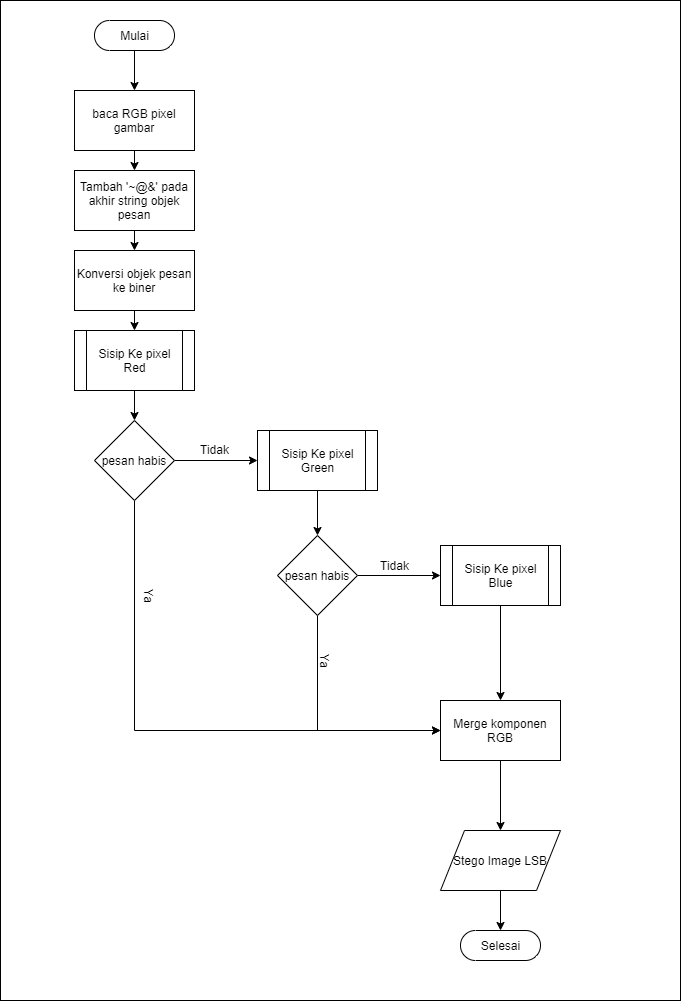
**Gambar 3.4** Struktur Menu Utama Sistem

Pada gambar 3.4 menunjukan sistem yang akan dibuat memiliki tiga pilihan proses utama. Pilihan pertama berupa menu *encode*. Pada pilihan *encode* ini *user* akan memasukkan gambar *cover* dan objek *hidden* lalu akan melakukan proses penyisipan objek dengan semua metode yang ada. Proses ini menghasilkan empat citra berbeda yang meliputi citra yang telah disisipi dengan metode LSB, DCT, dan DWT, serta kombinasi ketiga metode tersebut. Pada pilihan kedua terdapat proses *decode* atau ekstraksi objek dari objek stego. Pada pilihan ketiga adalah proses identifikasi dimana *user* akan memasukkan citra stego asli dan citra stego yang sudah dimodifikasi/serang untuk dibandingkan. Pada proses ini *output* yang dihasilkan berupa komparasi kedua citra tersebut secara visual, ekstraksi objek *hidden*, dan nilai PSNR.



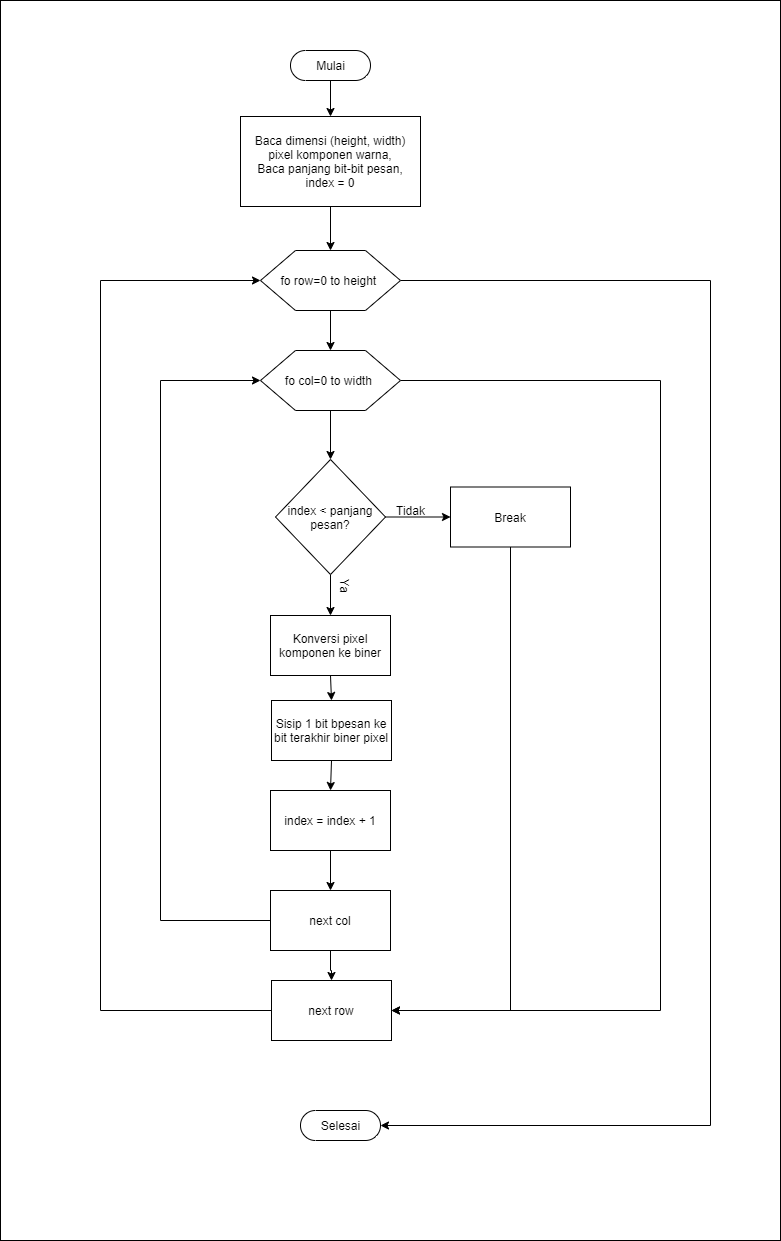
**Gambar 3.5** Flowchart Proses Encode

Pada gambar 3.5 ditunjukkan proses *encode* atau penyematan *hidden* *object* kedalam citra *cover*. Proses tersebut melibatkan sub proses *encode* LSB, DCT, dan DWT. Hasil dari sub proses tersebut merupakan sebuah citra stego dari masing-masing metode yang akan diidentiikasi kedepannya serta nilai PSNR dan MSE yang akan dijadikan acuan dalam menganalisis *fidelity* citra stego tersebut.



**Gambar 3.6** Flowchart sub proses encode LSB

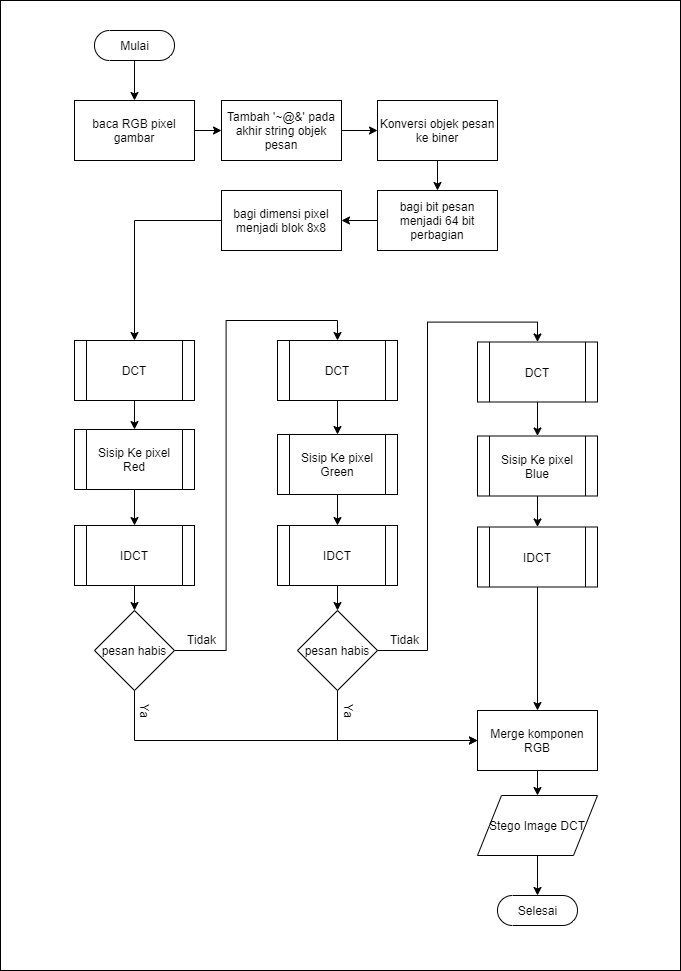
Pada sub proses *encode* LSB yang ditunjukkan gambar 3.6, citra *cover* akan dibaca panjang dan lebarnya dan akan dijadikan acuan ukuran untuk batas memasukkan *hidden* *object*. Hidden object berupa teks akan langsung ditambahkan tanda stop berupa ‘~@&’ yang sekiranya 3 karakter tersbut jarang dikombinasikan dalam kehidupan. Sementara *hidden object* berupa citra akan diubah menjadi *string* dengan *encode* base64() barulah ditambahkan tanda stop. Hal ini dilakukan agar penyematan juga berupa *string*. Pada pilihan *hidden* *object* teks merupakan representasi objek yang pendek dan tidak padat dan *hidden object* citra yang dijadikan string merupakan representasi objek yang panjang dan padat. Kemudian *hidden* *object* yang akan disematkan akan diubah menjadi bit-bit biner. Bit-bit biner tadi akan digunakan untuk penyisipan pada *pixel* R, G, dan B. Komponen-komponen tersebut didapatkan dengan fungsi split() dari *library* Open-CV pada python. Metode penyematan yang digunakan menggunakan keseluruhan tiga komponen warna pada citra. Ketika pesan yang sudah dikonversi menjadi biner belum habis disematkan pada komponen warna *red*, maka akan dilanjutkan penyematan pada komponen warna *green*. Begitu pula seterusnya pada komponen warna *blue*. Pada saat proses penyisipan, sesuai dengan namanya, bit yang diubah merupakan bit terakhir dari 8-bit nilai warna dari *pixel* tersebut seperti yang ditunjukan pada *flowchart* gambar 3.7.



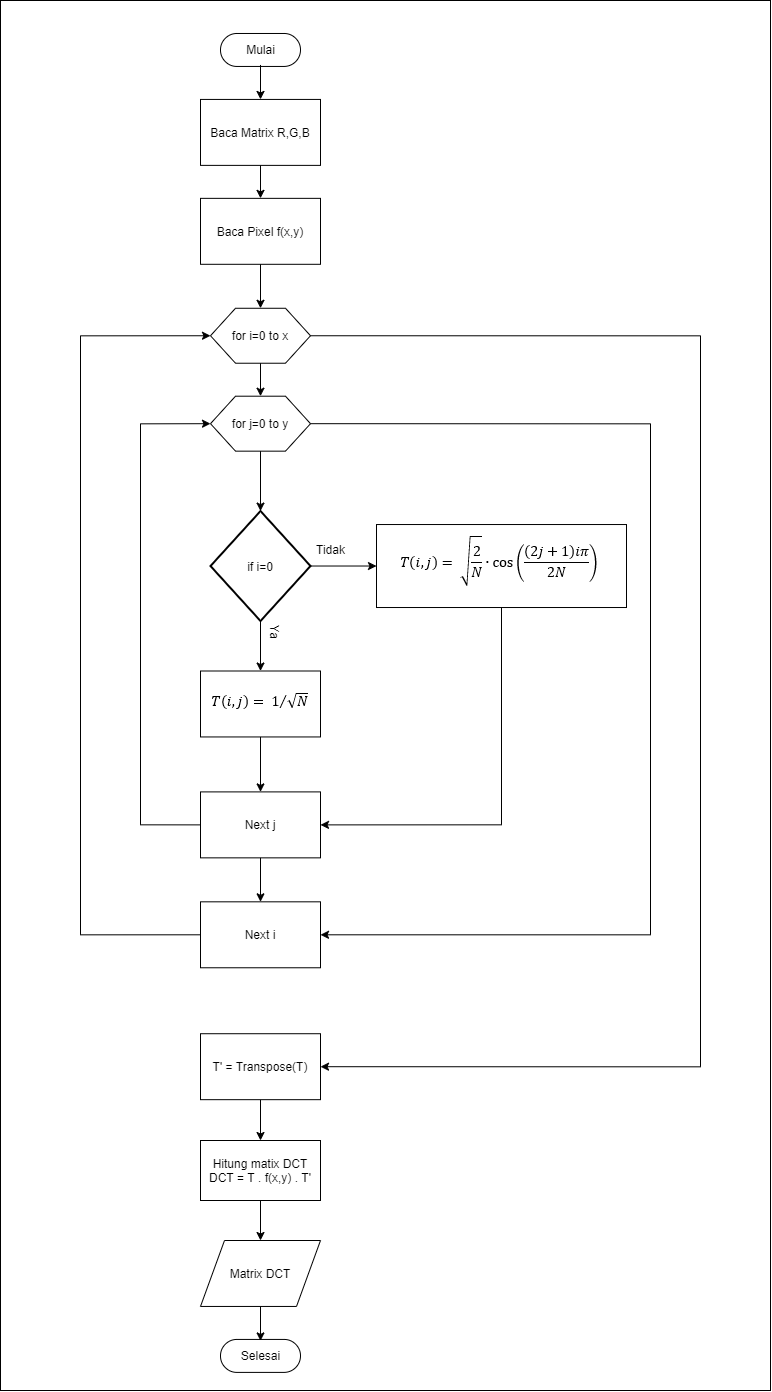
**Gambar 3.7** Flowchart sub proses penyisipan pada pixel komponen warna

Setelah proses penyisipan selesai maka komponen R, G, dan B akan ditata ulang dengan fungsi merge() dari *library* Open-CV. Hal ini bertujuan menggabungkan komponen RGB menjadi satu kesatuan citra yang utuh. Hal ini juga akan dilakukan pada proses penyisipan dengan metode yang selain LSB.

Pada penyisipan pesan menggunakan metode DCT, proses penyisipan hampir sama dengan LSB dikarenakan steganografi ranah frekuensi tetap membutuhkan proses penyisipan ke bit lokasi pixel dengan penyisipan ranah spasial seperti LSB. Hal tersebut dapat dilihat pada gambar 3.8.

**Gambar 3.8** Flowchart sub proses encode DCT

Sebelum dilakukan penyisipan, bit-bit pesan dibagai menjadi beberapa bagian yang terdiri dari 64-bit perbagiannya. Hal ini dikarenakan citra akan dibagi menjadi blok 8x8 sehingga *pixel*-*pixel* yang akan diubah dengan DCT adalah berukuran 8x8 dari masing-masing blok pada citra. Karena ukuran tersebut maka jika pada satu *pixel* mengandung 8-bit, maka pada 64-*pixel* masing-masing menyisipkan 1-bit pada akhir 8-bit mereka.



**Gambar 3.9** Flowchart sub proses DCT

Metode DCT menggunakan pendekatan nilai kosinus yang akan mengubah detail warna dari gambar. Namun karena keterbatasan indra penglihatan manusia, maka perubahan yang terjadi tidak begitu terlihat. Pada gambar 3.9 ditunjukkan proses yang dijalankan berdasarkan prosedur DCT tipe 2 untuk objek dua dimensi. Proses tersebut dimulai dengan membaca nilai RGB dengan *library* Open-CV dengan fungsi split() yang akan memisahkan matriks R, G dan B. Kemudian pada setiap matriks akan dibaca posisinya mulai dari koordinat awal sampai akhir dan menjalankan algoritma yang telah ditentukan dalam metode DCT. Untuk contoh perhitungan manual akan dijeaskan sebagai berikut:

1. Mengambil *pixel* dari komponen R/G/B citra. Sebagai contoh perhitungan ini akan menggunakan ukuran 4x4 *pixel*. Diibaratkan matriks pixel(x,y) = matriks A.

A =

1. Mencari matriks T dengan persamaan 2.2. Berdasarkan matriks A, diketahui bahwa:

N = 4

Representasi posisi pixel (x, y) = (i, j)

Jika x sama dengan 0, maka

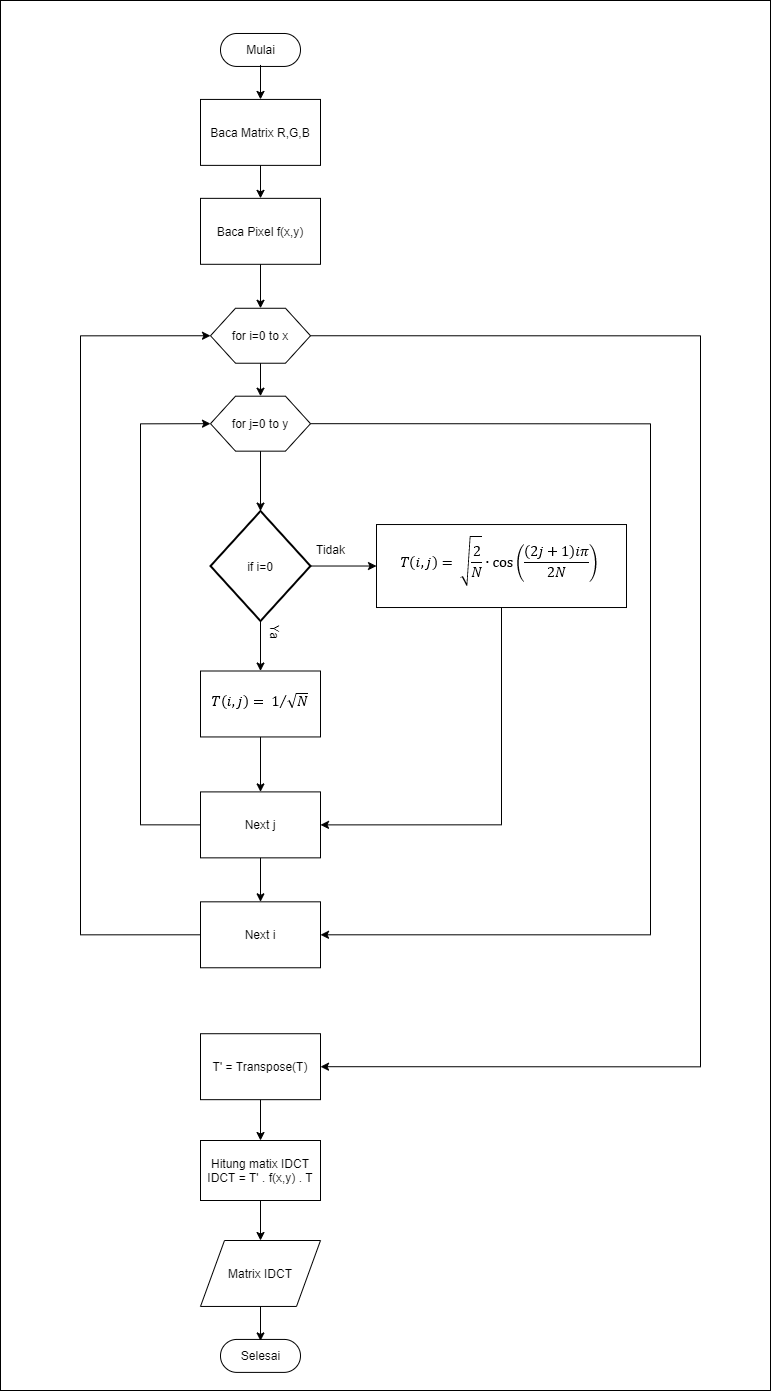
Jika x tidak sama dengan 0, maka

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Sehingga, nilai matriks T secara keseluruhan adalah:

1. Menghitung matriks akhir DCT dengan mengalikan matrik T, matriks A, dan matriks *Transpose* T secara berurutan.
2. Mengubah setiap elemen matriks DCT sehingga bertipe data *integer* agar dapat dikonversi menjadi biner.

Setelah mendapat matriks DCT dengan tipe data *integer*, maka selanjutnya elemen-elemen matriks tersebut dapat disematkan pesan melalui metode LSB. Elemen pada matriks harus bersifat *integer* sedangkan hasil dari perhitungan DCT bertipe data *float*. Maka dari itu harus diubah terlebih dahulu tipe data dari elemen matriks karena untuk konversi biner menggunakan fungsi *integer* dari python yang diwajibkan nilai bersifat *integer* dan bukan *float*. Proses penyisipan menyesuaikan metode yang digunakan. Karena pada DCT dimensi *pixel* dibagi menjadi blok 8x8 maka pesan yang disisipkan pada blok tersebut merupakan 8-bit atau satu karakter. Hal tersebut berlanjut sampai pesan habis disisipkan. Setelah pesan disisipkan pada matriks DCT tersebut, maka matriks tersebut perlu dikembalikan lagi ke keadaan awal melalui proses IDCT. Proses IDCT (*Inverse Discrete Cosine Transform*) merupakan proses untuk membalikkan matriks hasil DCT ke keadaan awal. Untuk lebih penjelasan lebih lanjut terdapat pada gambar 3.10.



**Gambar 3.10** Flowchart sub proses Inverse DCT

Dalam sub proses IDCT yang membedakan prosesnya dengan *Forward* DCT adalah perkalian matriks yang dilakukan pada akhir algoritma. Jika pada *forward* DCT perkalian melibatkan secara urut hasil perhitungan matriks T, matriks *pixel(x,y)* komponen warna (R/G/B), dan *transpose* dari matriks T (T’), sedangkan pada *inverse* DCT perkalian melibatkan secara urut *transpose* dari matriks T (T’), matrix DCT, dan hasil perhitungan matriks T. Untuk contoh perhitungan manual akan dijeaskan sebagai berikut:

1. Mengambil *pixel* dari komponen R/G/B citra. Sebagai contoh perhitungan ini akan menggunakan ukuran 4x4 *pixel*. Diibaratkan matriks DCT = matriks B.

B =

1. Mencari matriks T dengan persamaan 2.2. Berdasarkan matriks A, diketahui bahwa:

N = 4

Representasi posisi pixel (x, y) = (i, j)

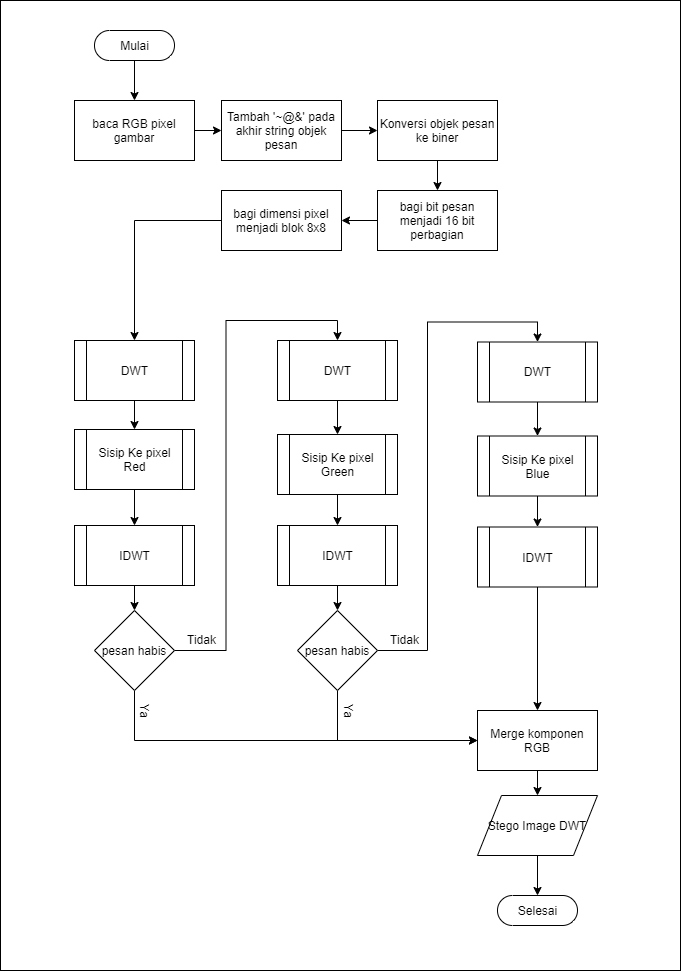
Jika x sama dengan 0, maka

Jika x tidak sama dengan 0, maka

Karena dimensi matriks sama dengan perhitungan DCT, maka hasil matriks T pada perhitungan IDCT merupakan hasil yang sama matriks T pada perhitungan DCT. Sehingga, nilai matriks T secara keseluruhan adalah:

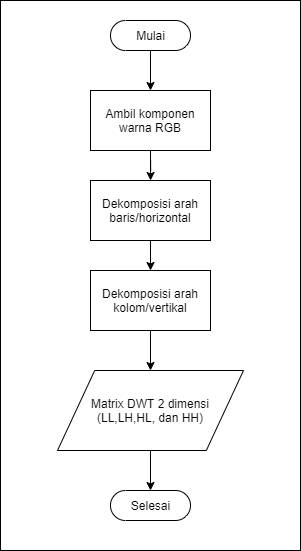
1. Menghitung matriks akhir IDCT dengan mengalikan matrik *Transpose* T, matriks B, dan matriks T secara berurutan. Selanjutnya elemen dijadikan integer untuk proses penggabungan komponen RGB menjadi citra utuh.

Setelah melalui proses IDCT, maka elemen-elemen pada matriks IDCT dikonversi menjadi *integer*. Hal ini dilakukan karena dalam proses penggabungan komponen R, G, dan B harus dimasukkan dalam bentuk *integer*. Terkait dengan penyisipan pesan, pada metode DCT juga menerapkan tiga ruang warna R, G, dan B. Hal ini dilakukan sehingga kapasitas pesan bisa diperbanyak. Ketika pesan belum habis maka penyisipan akan menggunakan komponen warna selanjutnya. Ketika pesan sudah habis disisipkan barulah citra dapat dibentuk kembali melalui proses penggabungan komponen RGB. Sama seperti pada penggabungan komponen citra pada proses LSB, penggabungan ini juga menggunakan fungsi merge() dari *library* Open-CV milik python.



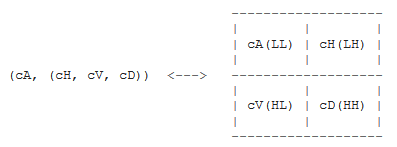
**Gambar 3.11** Flowchart sub proses Encode DWT

Metode *encode* DWT tidak berbeda jauh dengan proses *encode* DCT. Yang membedakan hanyalah jumlah ruang penyematan bit pesan dan metode DWT itu sendiri. Jika pada proses *encode* DCT pesan dibagi menjadi masing-masing bagian mendapat 64-bit maka pada DWT pesan hanya dibagi menjadi masing-masing 16-bit. Hal ini dikarenakan pada metode DWT nantinya citra juga akan dibagi menjadi blok 8x8, lalu blok tersebut akan mengalami dekomposisi sehingga dibagi menjadi 4 bagian yang akan dijelaskan kemudian. Hasil akhir dekomposisi tersebut adalah blok matriks *pixel* berukuran 4x4.



**Gambar 3.12** Flowchart sub proses DWT

Metode DWT (*Discrete Wavelete Transform*) dimulai dengan mengambil salah satu komponen warna RGB pada citra. Kemudian komponen warna tersebut diolah dengan dekomposisi ke arah baris dan ke arah kolom. Dari hasil dekomposisi tersebut akan terbentuk 4 matriks utama yaitu matriks cA, cH, cV, dan cD. Dekomposisi tersebut memanfaatkan fungsi dwt2() milik *PyWavelete*. Fungsi tersebut merupakan fungsi untuk menerapkan DWT pada matriks dua dimensi dengan metode *Haar*. Matriks-matriks tersebut mewakili *subband* dari hasil DWT yaitu *Low-Low*(LL), *Low*-*High*(LH), *High*-*Low*(HL), dan *High*-*High*(HH) seperti pada ilustrasi gambar 3.13.



**Gambar 3.13** Ilustrasi subband DWT

Untuk contoh perhitungan manual akan dijelaskan sebagai berikut:

1. Mengambil *pixel* dari komponen R/G/B citra. Sebagai contoh perhitungan ini akan menggunakan ukuran 4x4 *pixel*. Diibaratkan matriks pixel(x,y) = matriks A.

A =

1. Menentukan matriks hasil dekomposisi baris dari matriks A. Diibaratkan matrik baru tersebut adalah matriks T.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Sehingga didapatkan matrik baru sebagai berikut:

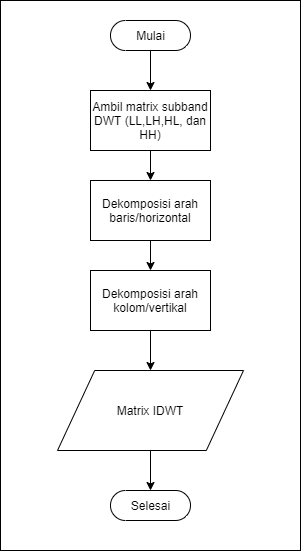
1. Menentukan matriks hasil dekomposisi kolom dari matriks hasil dekomposisi baris. Diibaratkan hasil dari matriks dekomposisi kolom adalah matriks T.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Sehingga matriks baru sebagai berikut:

1. Membagi matriks hasil dekomposisi kolom menjadi empat bagian, yaitu cA, (cH, cV, cD).

Contoh di atas merupakan gambaran penghitungan manual dekomposisi DWT. Bila terjadi pada matriks berukuran 8x8, maka *subband* akan menjadi 4x4 sehingga pesan yang dapat disisipi merupakan 16-bit dari keseluruhan pesan yang ada. Metode penyisipan pesan masih menggunakan cara yang sama, yaitu dengan memanfaatkan metode LSB yang memanfaatkan keseluruhan komponen warna RGB sebagai metode penyisip pesan, hanya saja melalui dekomposisi matriks terlebih dahulu. Tempat yang digunakan untuk metode ini ialah *subband Low-High*(LH) atau cH yang diubah tipe data elemennya menjadi bertipe *integer*. Setelah semua pesan berhasil dimasukkan maka proses selanjutnya adalah melakukan IDWT (*Inverse Discrete Wavelete Transform*) pada *subband-subband* yang telah dimasukkan pesan.



**Gambar 3.14** Flowchart sub proses Inverse DWT

Metode IDWT merupakan kebalikan dari proses DWT. Proses ini berfungsi mirip dengan proses IDCT, yaitu mengembalikan matriks dekomposisi menjadi matriks semula. Singkatnya matriks subband-subband akan disatukan kembali dan akan mengulang proses dekomposisi Untuk contoh perhitungan manual akan dijelaskan sebagai berikut:

1. Mengambil *subband-subband* hasil dari proses penyisipan pesan. lalu digabungkan menjadi posisi semula seperti pada gambar 3.13. Diibaratkan matriks tersebut adalah matriks A.

A =

1. Menentukan matriks hasil dekomposisi baris dari matriks T. Diibaratkan matrik baru tersebut adalah matriks T.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Sehingga didapatkan matrik baru sebagai berikut:

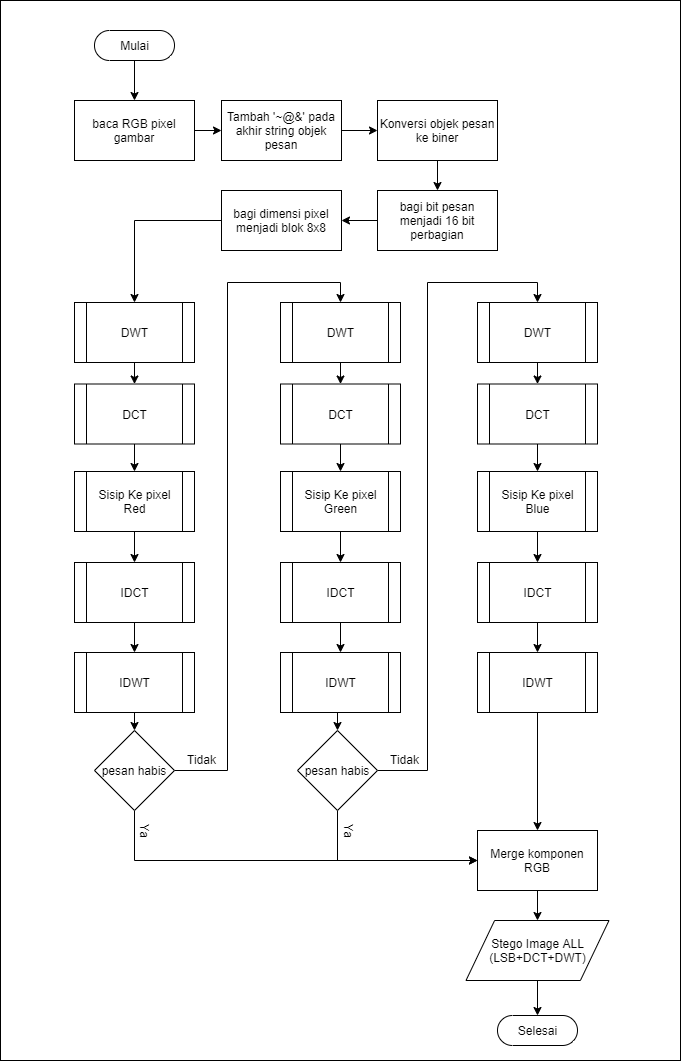
1. Menentukan matriks hasil dekomposisi kolom dari matriks A. Diibaratkan hasil dari matriks dekomposisi kolom adalah matriks T.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Sehingga matriks baru sebagai berikut:

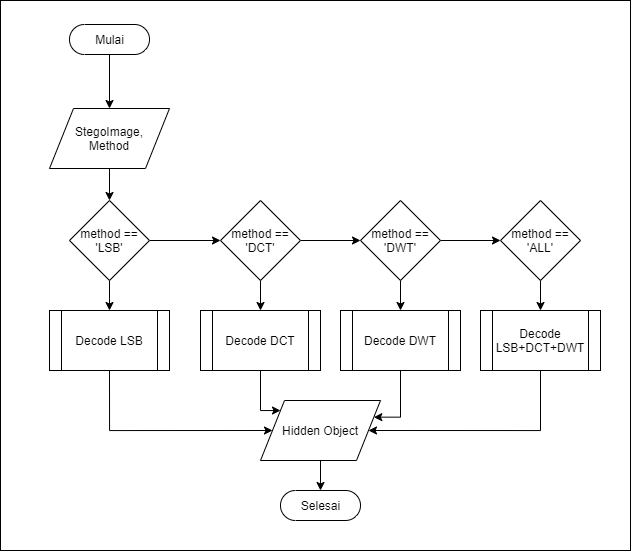
1. Mengubah tipe data setiap elemen matriks T menjadi *integer*.

Setelah melewati proses IDWT maka komponen-komponen ruang warna akan digabungkan kembali menggunakan fungsi merge(). Proses penggabunggan RGB membutuhkan satu kesatuan komponen citra yang utuh dan sudah tidak terdekomposisi. Jika tidak, maka komponen-komponen tersebut tidak dapat disatukan menjadi citra utuh.



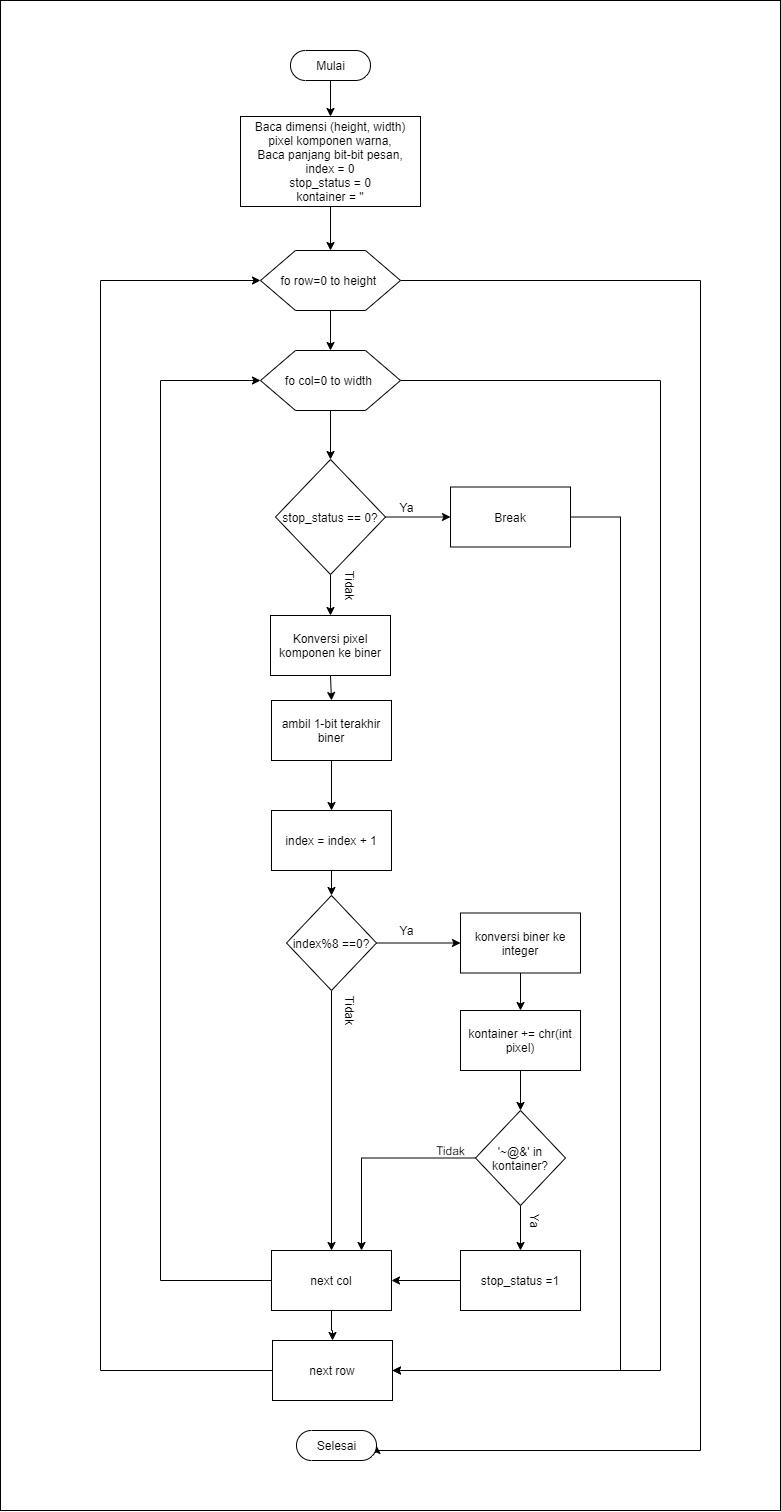
**Gambar 3.15** Flowchart sub proses encode Kombinasi (LSB+DCT+DWT)

Metode kombinasi yang dilakukan adalah menggunakan semua metode yang diajukan pada penelitian ini menjadi satu. Pada sub proses ini, citra akan mengikuti pembagian blok 8x8 dan pembagian pesan 16-bit perbagiannya. Mula-mula citra akan diambil komponen warnanya menggunakan fungsi split() seperti metode lain untuk mendapat data ruang warna RGB pada citra tersebut. Selanjutnya komponen ruang warna tersebut akan di dekomposisi DWT menggunakan fungsi dwt2() terlebih dahulu menghasil kan subband cA, cH, cV, cD. Kemudian subband cH akan dipilih untuk melakukan DCT dengan fungsi dct(). Hasil DCT tersebut akan disispkan pesan dengan metode LSB dengan panjang pesan masing-masing 16-bit karena matriks yang tersedia merupakan hasil pembagian blok 8x8 yang didekomposisi oleh DWT sehingga hanya tersedia matriks 4x4 yang berisi 16 elemen. Setelah pesan berhasil disisipkan maka akan dilakukan IDCT menggunakan fungsi idct() dan selanjutnya akan dilakukan idwt2() sehingga menjadi komponen warna utuh seperti semula. Terakhir maka seluruh komponen warna akan digabung menggunakan fungsi merge() untuk menjadi satu matriks citra utuh kembali.



**Gambar 3.16** Flowchart Proses Decode

Setelah semua metode melakukan penyisipan, maka untuk mendapat pesan tersembunyi di dalam sebuah citra perlu adanya ekstraksi pesan dari dalam citra tersebut. Untuk proses ektraksi pesan tersebut proses yang dijalankan tidak jauh berbeda dari proses encode atau penyisipan yang telah dilakukan. Yang membedakan hanyalah pemanggilan fungsi encode() diubah menjadi fungsi decode(). Keduanya memiliki perbedaan pada fungsinya. Apabila fungsi encode adalah menyisipkan pesan pada komponen warna maka fungsi decode() adalah sebaliknya, yaitu mengambil pesan yang berada pada komponen warna yang telah disisipi pesan. Fungsi decode() digambarkan dalam *flowchart* seperti yang terlihat pada gambar 3.16.



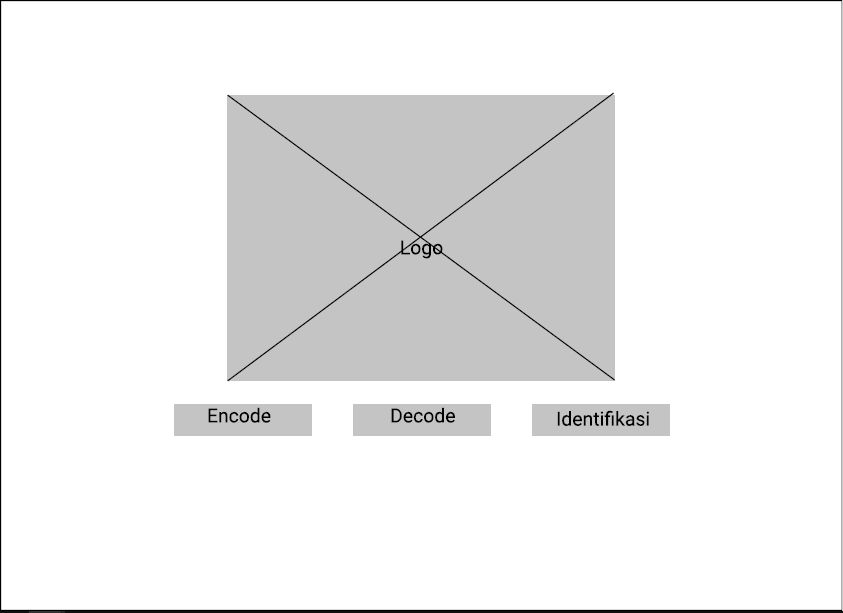
**Gambar 3.17** Flowchart sub proses ekstraksi pada pixel komponen warna

Metode ektraksi di atas pada dasarnya hampir sama dengan metode penyisipan pesan. namun pada proses ekstraksi panjang bit tidak diketahui sehingga perlu mencari *stopper* berupa string ‘~@&’ untuk menghentikan ekstraksi pesan. Dalam proses di atas proses tidak akan berhenti sebelum menemukan *stopper*. *Stopper* akan diketahui ketika di kontainer mengandung ‘~@&’ pada string tersebut. Kontainer akan menerima karakter dari penyatuan bit-bit ketika index merupakan kelipatan delapan, maka bit-bit tersebut akan dikonversi menjadi karakter dan dimasukkan kedalam kontainer. Apabila dalam kontainer tidak ditemukan *stopper*, maka perulangan akan terus berjalan sampai pada komponen warna terakhir di akhir matriksnya. Namun apabila *stopper* ditemukan, maka perulangan akan berhenti dan pesan dapat ditampilkan.

#### Perancangan Antarmuka

Perancangan *interface* atau antarmuka merupakan suatu kebutuhan yang penting dalam pembuatan aplikasi dan merupakan mekanisme komunikasi antara *user* dengan sistem. Hal tersebut memiliki tujuan untuk mengkomunikasikan fitur-fitur yang ada didalam sebuah sistem tersebut sehingga dapat dengan mudah digunakan. Antarmuka pada sistem yang akan dibuat merupakan tampilan *web*-*based* yang diakses menggunakan *desktop* *browser*. Perancangan antarmuka dapat dilihat pada gambar 3.18 sampai dengan gambar 3.23.

##### ***Wireframe* Antarmuka Halaman *Home***



**Gambar 3.18** Wireframe Tampilan Awal

Pada perancangan antarmuka sistem ini terdapat tiga menu utama yang dapat diakses oleh pengguna. Adapun penjelasan mengenai tombol-tombol menu yang terdapat pada gambar dijelaskan sebagai berikut:

1. *Encode*

Menu untuk melakukan penyisipan steganografi

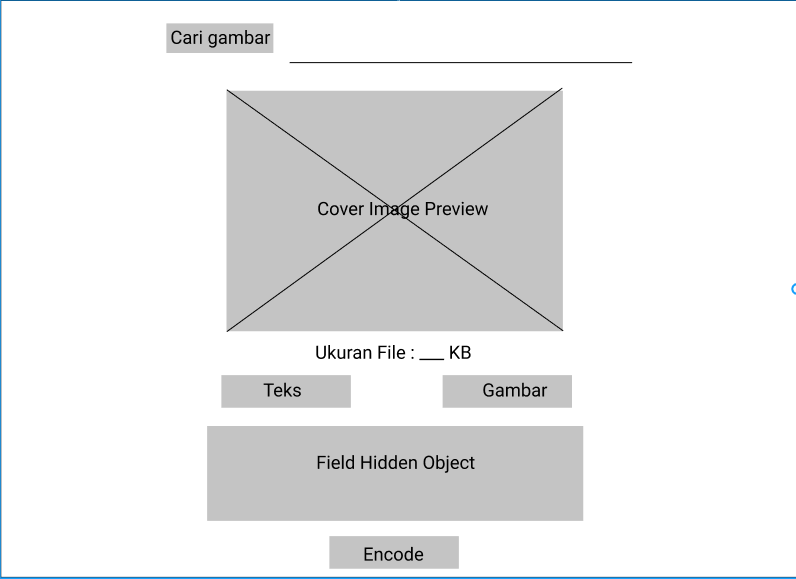
1. *Decode*

Menu untuk melakukan ekstraksi *hidden* *object*

1. Identifikasi

Menu untuk melakukan identifikasi pada dua citra. Baik citra *cover* dan citra stego maupun citra stego original dan citra stego yang diserang.

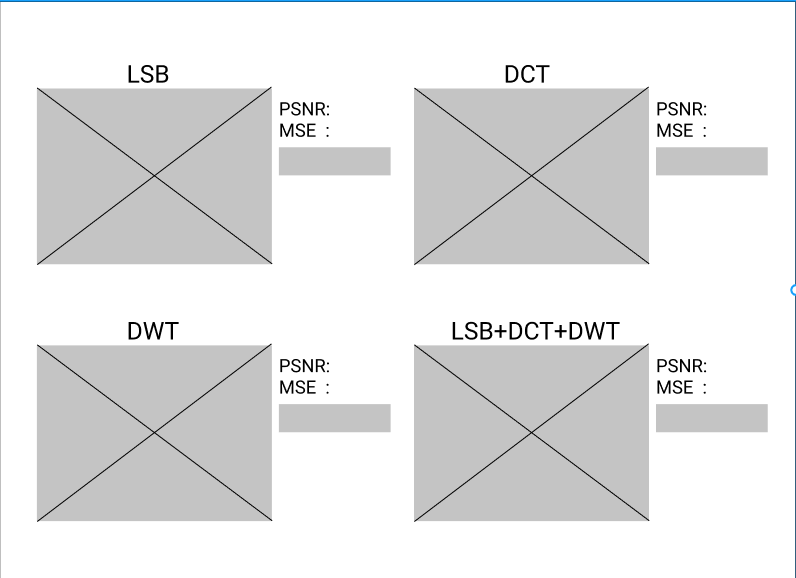
##### ***Wireframe* Antarmuka Halaman *Encode***



**Gambar 3.19** Wireframe Tampilan Encode

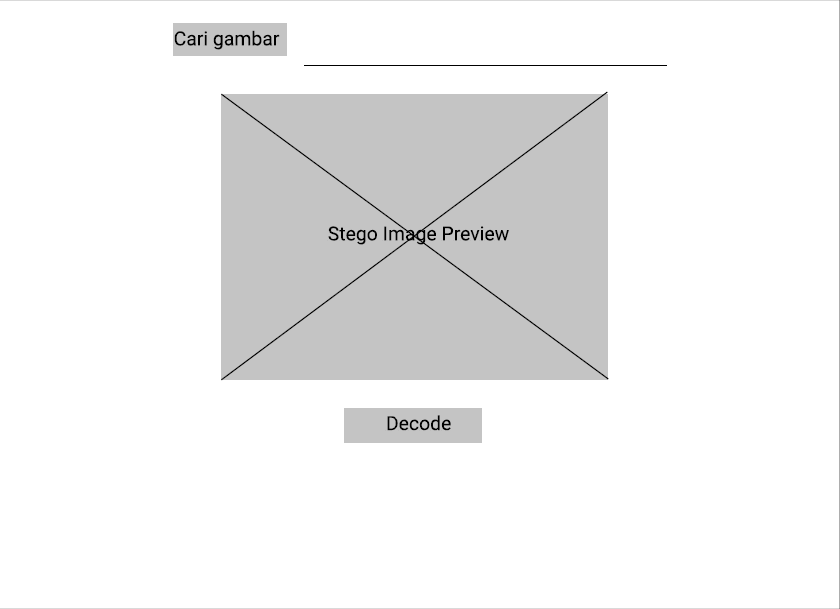
Pada halaman *encode* digunakan untuk melakukan penyisipan *hidden* *object* ke dalam *cover* *image* seperti pada gambar 3.19. Pada halaman tersebut terdapat bagian pencarian citra *cover* dan bagian *field* *hidden* *object.* Pada *field* *hidden* *object* dapat disesuaikan sesuai keinginan pengguna akan menyisipkan teks atau gambar. Setelah melakukan penyisipan maka akan terbuka sub halaman berupa kumpulan hasil citra stego beserta hasil perhitungan PSNR pada masing-masing citra stego.

##### ***Wireframe* Antarmuka Halaman Hasil *Encode***



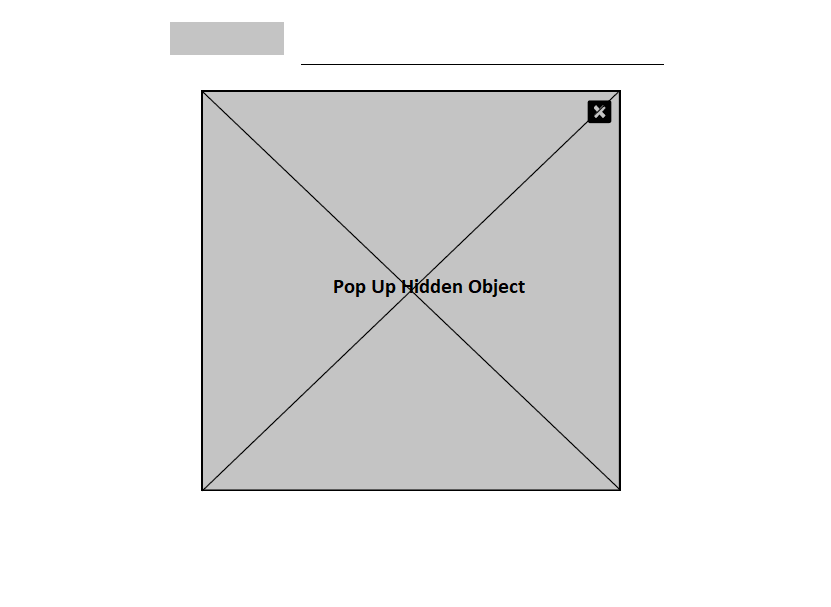
**Gambar 3.20** Wireframe Tampilan Hasil Encode

##### ***Wireframe* Antarmuka Halaman *Decode***



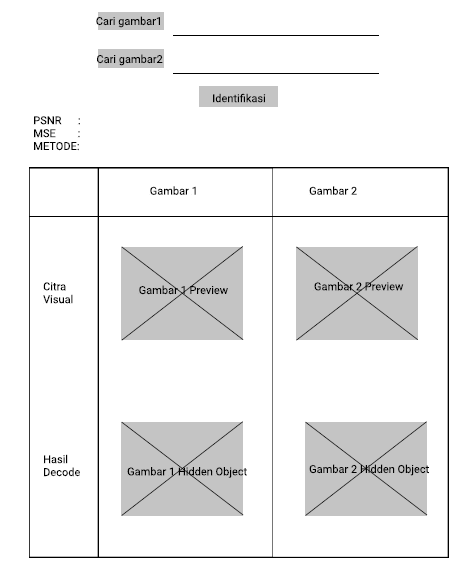
**Gambar 3.21** Wireframe Tampilan Decode

Pada halaman menu *decode* terdapat pencarian citra yang akan digunakan untuk mencari citra stego dan tombol *decode*. Ketika tombol ditekan maka sebuah *pop* *up* yang berisikan hasil dari ekstrak *hidden* *object* akan ditampilkan seperti pada gambar 3.22.



**Gambar 3.22** Wireframe Tampilan Pop Up Hasil Ekstraksi hidden object

##### ***Wireframe* Antarmuka Halaman Identifikasi**



**Gambar 3.23** Wireframe Tampilan Identifikasi

Pada halaman identifikasi akan ditampilkan bagian pencarian dua buah citra dan sebuah table perbandingan dari kedua citra yang telah dimasukkan. Kedua citra tersebut akan dibandingkan dari segi visual, hasil *decode* (*hidden* *object*), dan ukuran *file*. Serta sebelumnya terdapat tingkat PSNR dari kedua buah citra sebgai acuan tingkat perbedaan pada kedua citra.

#### Perancangan Pengujian

Pada penelitian ini pengujian yang dilakukan merupakan pengujian kualitas *robustness* *hidden object* dari citra stego dengan menggunakan *benchmark* *stirmark* serta pengujian *fidelity* citra stego dengan tolak ukur visual, selisih ukuran file, dan PSNR. Pengujian *stirmark* dilakukan setelah system dapat berjalan dan dapat digunakan. Pengujian ini akan memanfaatkan penyerangan pada citra stego berupa *geometric distortion attack* (serangan geometris) dan *compression* yang biasa dilakukan sebagai uji stirmark pada citra steganografi maupun *watermark*. Uji ini akan memanipulasi citra dengan cara dipotong (*crop*), diputar (*rotate*), dibalik (*flip*), diubah dimensinya (*resize*), dan ditekan ukuran file citra. Selanjutnya untuk menyelesaikan masalah utama pada penelitian ini citra stego akan diuji juga menggunakan media yang biasa digunakan pada kehidupan sehari-hari. Citra akan dikirim menggunakan fitur pesan pada aplikasi Whatsapp, Telegram, Line, E-Mail, Google Drive, serta citra juga akan dikirimkan menggunakan *pendrive*/*flashdisk*. Skenario pengujian untuk penelitian ini diantaranya sebagai berikut:

1. Skenario Uji *Robustness*

Skenario uji *robustness* pada citra stego meliputi tujuh tahap utama. Tujuh tahap tersebut diantaranya:

1. Tanpa Serangan
2. Dipotong (*Cropping*)
   1. 50% Horizontal Atas
   2. 50% Horizontal Bawah
   3. 50% Vertikal Kiri
   4. 50% Vertikal Kanan
3. Diputar (*Rotate*)
   1. 90° Ke Kiri
   2. 180° Ke Kiri
   3. 90° Ke Kanan
   4. 180° Ke Kanan
4. Dibalik (*Flip*)
   1. Horizontal
   2. Vertikal
5. Diubah dimensi (*Resize*)
   1. 25%
   2. 50%
   3. 75%
6. Ditekan ukuran file (*Compress*)
   1. 25%
   2. 50%
7. Pihak Ketiga
   1. WhatsApp
   2. Telegram
   3. Line
   4. Email
   5. Google Drive
   6. *Flashdisk*
8. Skenario Uji *Fidelity*

Skenario uji *fidelity* pada citra stego meliputi tiga komponen utama. Tiga komponen tersebut diantaranya:

1. Perbandingan secara visual
2. Perbandingan ukuran *file*
3. Perbandingan tingkat PSNR

# BAB IV HASIL, PENGUJIAN, DAN PEMBAHASAN

Hasil, pengujian, dan pembahasan merupakan tahap dimana sistem ini siap dioperasikan pada keadaan yang sebenarnya. Dari pembahasan ini akan diketahui apakah sistem yang dibuat dapat menghasilkan tujuan yang sesuai berdasarkan analisis dan perancangan yang telah dilakukan.

## Hasil Penelitian

# BAB V PENUTUP

## Kesimpulan

# DAFTAR PUSTAKA

# LAMPIRAN