PENCARIAN CITRA MENGGUNAKAN METODE HISTOGRAM *WAVELET* TERURUT

SKRIPSI

Oleh:
BAMBANG MULYO UTOMO
(0410960012-96)

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG
2011

UNIVERSITAS BRAWIJAYA

PENCARIAN CITRA MENGGUNAKAN METODE HISTOGRAM *WAVELET* TERURUT

SKRIPSI

Sebagai salah satu syarat untuk memperoleh gelar
Sarjana dalam bidang Ilmu Komputer

Oleh:
BAMBANG MULYO UTOMO
(0410960012-96)

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2011

UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN SKRIPSI

PENCARIAN CITRA MENGGUNAKAN METODE HISTOGRAM *WAVELET* TERURUT

Oleh:
Bambang Mulyo Utomo
0410960012-96

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 11 Agustus 2011

dan dinyatakan memenuhi syarat untuk memperoleh gelar Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I,	Pembimbing II,
Drs. Marji, MT	Candra Dewi, S.Kom., MSc
NIP. 196708011992031001	NIP. 197711142003122001

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, M.Sc
NIP.196709071992031001

UNIVERSITAS BRAWIJAYA

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama	: Bambang Mulyo Utomo
NIM	: 0410960012-96
Jurusan	: Matematika
Program Studi	: Ilmu Komputer
Penulis skripsi berjudul	: Pencarian Citra Menggunakan Metode Histogram *Wavelet* Terurut

Dengan ini menyatakan bahwa :

1. Isi dari Skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Skripsi ini.
2. Apabila dikemudian hari ternyata Skripsi yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 11 Agustus 2011
Yang menyatakan,

Bambang Mulyo Utomo.
NIM. 0410960012-96

UNIVERSITAS BRAWIJAYA

ABSTRAK

Pencarian citra merupakan proses pencocokan antara ciri –ciri citra *query* dengan ciri – ciri citra *database* . Hasil yang diharapkan tentunya harus sesuai dengan keinginan *user* . Pencarian citra pada umumnya berdasarkan teks atau teks *query* yang dimasukkan oleh *user*. Namun seiring dengan perkembangan ilmu pengetahuan saat ini, telah banyak di kembangkan dan dikenalkan teknik pencarian citra yang didasarkan pada kemiripan citra yang dimasukkan oleh *user*. Citra tersebut merupakan citra inputan atau *image query* yang digunakan sebagai parameter masukan untuk proses pencocokan pada citra yang telah ada di dalam database.

Sistem ini menjelaskan tentang suatu teknik yang digunakan untuk pencarian setumpukan citra. Teknik yang dimaksud adalah teknik transformasi *haar wavelet* dan metrika Lq. Jenis transformasi *wavelet* yang digunakan adalah Haar wavelet dan untuk pencocokannya menggunakan metrika Lq. Selanjutnya dilakukan proses pemotongan koefisien pada data koefisien serta dilakukan proses kuantisasi sehingga data koefisien menjadi lebih seragam. Selanjutnya akan dibuat suatu histogram dari nilai koefisien dan frekuensi kemunculannya. Proses pencocokan citra dilakukan dengan menghitung selisih frekuensi antara data *image database* dengan citra uji. Semakin kecil selisih antara dua histogram tersebut maka semakin mirip citra tersebut. Pengujian dilakukan dengan mengambil 3 citra dari semua kelompok citra yang ada pada database. Citra yang digunakan adalah citra asli, *blur*, *edge* dan *noise*. Hasil pengujian menunjukkan bahwa metode memberikan hasil tingkat kesuksesan untuk citra asli 98,65 %, citra *blur* 94,31 %, citra *edge* 46,19 %, dan citra *noise* 78,28 %.

Kata kunci : pencarian citra, *wavelet*, *haar wavelet*, metrika Lq, citra *query*

UNIVERSITAS BRAWIJAYA

ABSTRACT

Image searching is a process that matched image query characteristic feature with database image characteristic feature. The result expected with is the one that correspond with user requirement. Image searching in general are based on user inputted text or query text. But with the advanced technology and science nowadays, image searching technique that based on similarity with image by user input has been developed. The image is an inputted image or an image query that used as input parameter to process the adjustment matching with the image that has been stored in the database

This system explains about the technique that is used to searching process in a pile of images. The technique is the sorted wavelet histogram. The wavelet transformation that has been used is the haar wavelet for the pre processing, and then the result is sorted again to gain a significant image characteristic. Further process is coefficient cutting in the coefficient data, and the normalization process so that the coefficient data become uniformed. The next step is the histogram making process from the coefficient value and the frequency of occurrence. The process of image matching is done by calculating the frequency deviation between image database and the test image. The smaller the deviation between the histogram, then the two image is considered to be more similar. The testing process is been done by processing 3 image from all the image groups in the database. Using the real image, blur, edge and noise, the test result showed that this method gives the success rate of 98,65% for the original image, 94,31% for the blur image, 46,19% for edge image, and 78,28% for the noise image.

Keywords : image searching, wavelet, haar wavelet, metrika Lq, image query

UNIVERSITAS BRAWIJAYA

KATA PENGANTAR

Alhamdulillahi Robbil alamin, puji syukur penulis panjatkan kepada Allah SWT atas segala limpahan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan tugas akhir ini sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dalam bidang Ilmu Komputer.

Tugas akhir ini bertujuan untuk melakukan perancangan dan implementasi pencarian citra menggunakan metode histogram *wavelet* terurut. Dengan mengetahui hasil pengenalan dari metode ini, diharapkan dapat dijadikan sebagai sebuah masukan untuk membuat aplikasi pengenalanwajah dengan metode yang lebih baik.

Pada penyusunan tugas akhir ini, penulis ingin mengucapkan terima kasih kepada :

1. Drs. Marji MT., selaku pembimbing utama penulisan tugas akhir sekaligus selaku pembimbing akademik dan Ketua Program Studi Ilmu Komputer, Jurusan Matematika, FMIPA Universitas Brawijaya.
2. Candra Dewi, S.Kom, MSc, selaku pembimbing pendamping dalam penulisan tugas akhir.
3. Dr. Abdul Rouf Alghofari, M.Sc., selaku Ketua Jurusan Matematika, FMIPA Universitas Brawijaya.
4. Segenap bapak dan ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penulis selama menempuh pendidikan di Program Studi Ilmu Komputer Jurusan Matematika FMIPA Universitas Brawijaya.
5. Segenap staf dan karyawan di Jurusan Matematika FMIPA Universitas Brawijaya yang telah banyak membantu penulis dalam pelaksanaan penyusunan tugas akhir ini.
6. Orang tua penulis atas dukungan materi dan doa restunya kepada Penulis.
7. Rekan-rekan di Program Studi Ilmu Komputer FMIPA Universitas Brawijaya yang telah banyak memberikan bantuannya demi kelancaran pelaksanaan penyusunan tugas akhir ini.
8. Dan semua pihak yang telah membantu dalam penyusunan tugas akhir ini yang tidak dapat penulis sebutkan satu per satu.

Penulis sadari bahwa masih banyak kekurangan dalam laporan ini, oleh karena itu penulis sangat menghargai saran dan kritik yang sifatnya membangun demi perbaikan penulisan dan mutu isi tugas akhir ini untuk kelanjutan penelitian serupa dimasa mendatang.

Semoga laporan tugas akhir ini dapat bermanfaat. Amin.

Malang, 11 Agustus 2011

Penulis

DAFTAR ISI

BAB III METODOLOGI DAN PERANCANGAN

BAB IV HASIL DAN PEMBAHASAN

UNIVERSITAS BRAWIJAYA

DAFTAR GAMBAR

DAFTAR TABEL

UNIVERSITAS BRAWIJAYA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Saat ini, banyak institusi yang menggunakan data citra untuk keperluan tertentu, salah satunya Institusi kepolisian sebagai Institusi penegak hukum. Institusi tersebut memiliki banyak citra sidik jari dan wajah,untuk itu dibutuhkan suatu sistem pengelolaan data yang cepat dan efisien yang digunakan untuk proses identifikasi pencarian data citra tersebut sehingga hasil yang diharapkan dari pencarian itu akurat dan memiliki waktu yang pendek. Salah satu sistem pengolahan citra yang banyak digunakan adalah sistem pengelolaan basis data citra (*image database*).

Proses pencarian data citra dalam suatu *database* adalah suatu proses pencocokan antara ciri – ciri citra *query* dengan ciri- ciri citra pada *database* atau disebut citra target. Hasil yang diharapkan tentunya harus sesuai dengan keinginan *user*. Pencarian pada umumnya berdasarkan pada teks atau *text query* yang dimasukkan oleh *user*. Namun seiring dengan perkembangan ilmu pengetahuan saat ini, telah banyak dikembangkan dan dikenalkan teknik pencarian yang didasarkan pada kemiripan citra yang dimasukkan oleh *user*. Citra tersebut merupakan citra inputan atau *image query* yang digunakan sebagai parameter masukan untuk proses pencocokan pada citra yang telah ada di dalam *database*

Intisari dari pencarian citra adalah suatu proses pengenalan suatu objek dengan menggunakan metode tertentu dimana dalam proses pengenalannya memiliki tingkat akurasi yang tinggi, artinya bahwa suatu objek yang secara manual tidak dapat dikenali oleh manusia, tetapi bila menggunakan salah satu metode yang diaplikasikan pada komputer masih dapat dikenali dan dicari. Pengenalan citra dimulai dengan melakukan ekstraksi ciri citra yaitu mengenali citra berdasarkan karateristik citra. Salah satu metode yang biasa digunakan untuk mengenali suatu citra yang tersimpan di dalam suatu *database* berdasarkan pada ciri – ciri karateristik citra tersebut dengan pencocokan pada citra inputan adalah *Content Based Image Retrieval* (CBIR). Ciri – ciri citra yang dimaksud adalah warna, tekstur dan bentuk. Pada skripsi ini, ciri citra yang akan digunakan adalah berdasarkan warna.

Dalam CBIR, salah satu metode yang biasa digunakan adalah *wavelet.* Metode *wavelet* ini digunakan untuk *preprocessing*, sedangkan metrika Lq digunakan untuk proses pencocokannya. Banyak macam dan jenis *wavelet*, di antaranya adalah *Haar Wavelet*, Mathieu *Wavelet*, Daubechis *Wavelet* dan lain –lain. Metode yang digunakan pada skripsi ini adalah transformasi *wavelet* untuk proses *preprocessing* dan metrika Lq, sedangkan jenis transformasi *wavelet* yang digunakan adalah *haar wavelet*. Ciri – ciri tersebut kemudian diurutkan lagi untuk mendapatkan ciri – ciri yang signifikan, dengan melakukan pemotongan pada data koefisien yang memiliki nilai *magnitude* terbesar, serta mengkuantisasi variasi data sehingga data lebih seragam. Selanjutnya akan dibuat suatu histogram dari nilai koefisen dan frekuensi kemunculannya Proses pencocokan citra dilakukan dengan menghitung selisih frekuensi antara dua histogram tersebut, semakin kecil selisih antara dua histogram tersebut, maka semakin mirip citra tersebut. Namun jika selisihnya semakin besar, maka kedua citra tersebut tidak mirip dan dapat disimpulkan kalu citra yang ada pada *image database* itu bukanlah citra yang dicari. Dengan metode ini, kedua data yang dicocokkan relatif sedikit sehingga diharapkan waktu pencarian data menjadi pendek.

Penerapan metode CBIR lainnya adalah pada riset dengan judul ” *Pencarian Citra Menggunakan Metode Transformasi Wavelet dan Metrika Histogram Terurut*. “ yang menggunakan transformasi *wavelet* dengan jenis *wavelet daubechis* untuk mendekomposisikan koefisien dari sebuah citra digital dengan menggunakan analisa warna RGB dan Metrika Lq untuk perhitungan pencocokannya. Diperoleh hasil akurasi sebesar 100 % untuk citra Asli, 99,83 % untuk citra *blur*,90,25 % untuk citra tepi dan 73,57 % untuk citra berderau (Widiartha, I.B.K., dan Wijaya, I.G.P.S., 2006).

Oleh karena itu judul yang akan di ambil dalam skripsi ini adalah “**Pencarian Citra Menggunakan Metode Histogram *Wavelet* Terurut**”.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan, maka rumusan masalah yang akan diambil adalah sebagai berikut :

- Bagaimana mengimplementasikan pencarian citra menggunakan metode Transformasi *Haar Wavelet* dan Metrika Lq
- Bagaimana besar persentase tingkat kesuksesan hasil pencarian berdasarkan koefisien pemotongan

1.3 Batasan masalah

Cakupan masalah dibatasi pada :

- Citra yang digunakan sebagai *image query* dan *image database* adalah citra *true color* 24 bit dengan ukuran 128x128 piksel
- Citra yang digunakan merupakan citra dengan format (*.bmp)
- Transformasi *Wavelet* yang digunakan adalah jenis wavelet *Haar Wavelet*
- Metode pencocokan yang digunakan adalah Metrika Lq.

1.4 Tujuan penelitian

Tujuan penelitian dari penulisan skripsi ini adalah sebagai berikut :

- Membangun dan mengimplementasikan sebuah aplikasi pencarian data berupa data citra digital layaknya pencarian data berupa data teks menggunakan metode *Haar Wavelet* dan Metrika Lq.
- Mengevaluasi presentase akurasi hasil pencarian data citra menggunakan metode metode *Haar Wavelet* dan Metrika Lq.

1.5 Manfaat penelitian

Manfaat yang diperoleh dari pengerjaan skripsi ini adalah untuk terbentuknya sebuah perangkat lunak yang mampu melakukan pencarian data citra di dalam *image database* menggunakan metode *haar wavelet* dan metrika Lq. Selain itu adalah didapatnya tingkat akurasi kemiripan citra yang dicari.

1.6 Sistematika penulisan

Untuk memberikan gambaran tentang tugas akhir ini, berikut disajikan garis besar pembahasan dari keseluruhan isi tugas akhir untuk setiap babnya

BAB I : PENDAHULUAN

Bab ini berisi latar belakang penulisan, rumusan permasalahan yang akan dibahas, batasan masalah, tujuan dan manfaat serta sistematika penulisan tugas akhir.

BAB II : TINJAUAN PUSTAKA

Bab ini berisi penjelasan tentang teori – teori yang digunakan dalam menyelesaikan perancangan dan pembuatan aplikasi tentang pencarian data citra.

BAB III : METODOLOGI DAN PERANCANGAN

Bab ini berisi metode – metode yang digunakan dalam perancangan dan alur kerja aplikasi pencarian data citra

BAB IV : HASIL DAN PEMBAHASAN

Bab ini beisi tentang penjelasan implementasi sistem dan hasil pengujian yang dilakukan.

BAB V : PENUTUP

Bab ini berisi kesimpulan yang diperoleh dari hasil pengujian dan pembahasan serta saran – saran untuk pengembangan lebih lanjut.

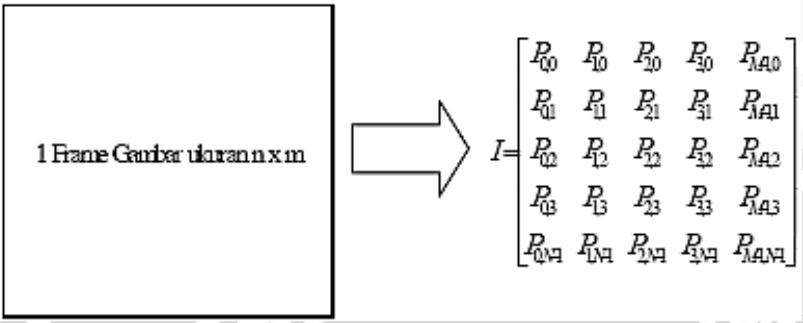
BAB II
TINJAUAN PUSTAKA

2.1 Definisi Citra Digital

Citra digital adalah sebuah citra yang telah di transformasikan ke dalam bentuk numeris dari tingkat keabuan pada piksel-piksel elemen citra. Kegiatan untuk mengubah informasi citra non digital menjadi citra digital disebut pencitraan (*imaging*) (Balza,A. dan Firdausy,K., 2005)

Dari sekian banyak citra hanya citra digital yang bisa diolah menggunakan komputer.Ada beberapa jenis citra lain yang dikelompokkan menjadi citra tampak dan citra tidak tampak. Contoh dari citra tampak dalam kehidupan sehari – hari, misalnya foto keluarga, sesuatu yang nampak pada televisi serta hologram.Sedangkan citra yang tidak tampak, misalnya data gambar dalam *file*, citra distribusi panas di kulit manusia.Untuk dapat dilihat oleh manusia, maka citra tidak tampak tadi harus diubah menjadi citra tampak, misalnya dengan menampilkannya di layar monitor, dicetak di kertas atau di tampilkan di layar televisi.

Berdasarkan dimensinya, citra dapat dibagi menjadi citra 2 dimensi maupun citra 3 dimensi.Proses pengolahan citra 3 dimensi adalah pengolahan yang cukup kompleks sehingga dalam pembahasan ini hanya citra digital 2 dimensi yang dibahas. Citra dapat dianggap sebagai matrik 2 dimensi ($f(x,y)$) dari bilangan yang dipresentasikan oleh sejumlah bit, dengan indeks baris dan kolomnya menyatakan koordinat (x,y) sebauh piksel pada citra tersebut. Nilai dari masing – masing elemennya menyatakan intensitas cahaya pada piksel tersebut.Pikselpada sebuah citra digital sering disebut sebagai elemen citra.Bentuk matrik citra digital dengan ukuran N x N piksel ditunjukkan pada gambar 2.1.



Gambar 2.1 Bentuk Matrik Citra Digital
(Sumber : Galih, 2011)

2.2 Komponen Citra Digital

Citra digital mempunyai beberapa karateristik, antara lain ukuran, resolusi, dan format nilainya. Bentuk dari citra digital pada umumnya adalah persegi panjang sehingga memiliki panjang dan lebar tertentu.Ukuran ini biasanya dinyatakan dalam banyaknya titik atau piksel, sehingga ukuran citra selalu bernilai bulat.

Ukuran citra dapat dinyatakan secara fisik dalam satuan panjang.Dalam hal ini harus ada hubungan antara ukuran piksel penyusun citra dengan satuan panjang. Hal ini dinyatakan dengan resolusi yang merupakan ukuran banyaknya piksel untuk setiap satuan panjang dalam satuan dpi (*dot per inch*) . Makin besar resolusi makin banyak piksel yang terkandung dalam citra dengan ukuran fisik yang sama, sehingga memberikan efek penampakan citra menjadi semakin halus. (Balza,A. dan Firdausy,K., 2005)

Citra digital memiliki format yang bermacam – macam, hal ini karena citra mempresentasikan informasi tertentu dan informasi tersebut dinyatakan secara bervariasi, maka citra yang mewakilinya dapat muncul dalam berbagai format.Informasi yang tersimpan pada citra digital berbentuk angka.Angka – angka tersebut mewakili warna.Citra digital tersusun atas piksel –piksel yang biasanya berbentuk persegi yang secara beraturan membentuk banyak baris dan kolom. Setiap piksel memiliki koordinat sesuai dengan posisinya dalam citra tersebut, biasanya dinyatakan dengan bilangan bulat positif yang dimulai dari 0 atau 1, piksel tersebut juga memiliki nilai

berupa angka digital yang merepresentasikan informasi pada piksel tersebut. Format nilai pikselsama dengan format citra keseluruhan.

2.3 Citra *Grayscale*

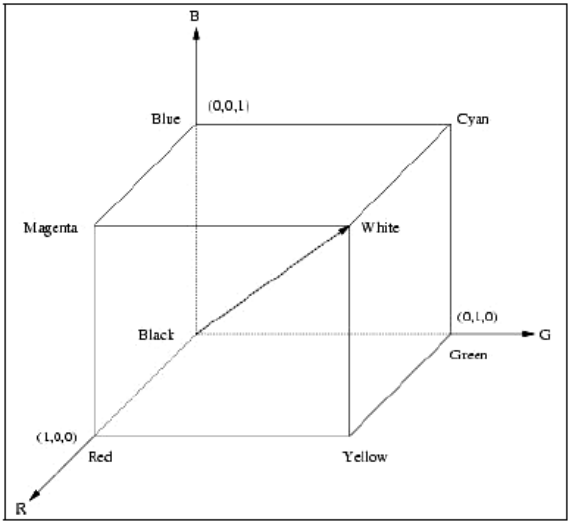
Citra *Grayscale* mempunyai kemungkinan warna lebih banyak daripada citra biner, karena terdapat nilai – nilai lain diantara nilai minimum dan maksimum. Banyaknya kemungkinan nilai minimum dan maksimumnya bergantung pada jumlah bit yang digunakan. Contoh untuk skala keabuan 4 bit, maka jumlah kemungkinan nilainya adalah $2^4 = 16$, dan nilai maksimumnya adalah $2^4-1 = 15$, sedangkan untuk skala keabuan 8 bit, maka jumlah kemungkinan nilainya adalah $2^8-1 = 255$.

Format citra ini disebut skala keabuan karena pada umumnya warna yang dipakai adalah antara hitam sebagai warna minimal dan puith sebagai warna maksimal, sehingga warna antaranya adalah abu – abu. Warna abu – abu merupakan warna yang memiliki nilai sama pada pada masing – masing warna yang membentuknya.

2.4 Citra Warna (*True Color*)

Pada citra warna, setiap titik mempunyai warna yang spesifik yang merupakan kombinasi dari 3 warna dasar, yaitu merah, hijau, dan biru.Format citra ini sering disebut sebagai citra RGB (*red-green-blue*). Setiap warna dasar memiliki nilai maksimum 255(8 bit), misalnya warna kuning merupakan kombinasi warna merah dan hijau sehingga nilai RGB-nya adalah 255,255,0. Dengan demikian setiap titik pada citra warna membutuhkan data 3 *byte*.(Balza, A. dan Firdausy, K., 2005).

Gambar 2.2 menunjukkan bentuk geometri dari model warna RGB untuk menspesifikasikan warna menggunakan sistem koordinat Cartesian.Spektrum *grayscale* (tingkat keabuan) yaitu warna yang dibentuk darigabungan tiga warna utama dengan jumlah yang sama,berada pada garisyang menghubungkan titik hitam danputih. (Suhendra, 2007)



Gambar 2.2 Koordinat Warna RGB
(Sumber : Suhendra, 2007)

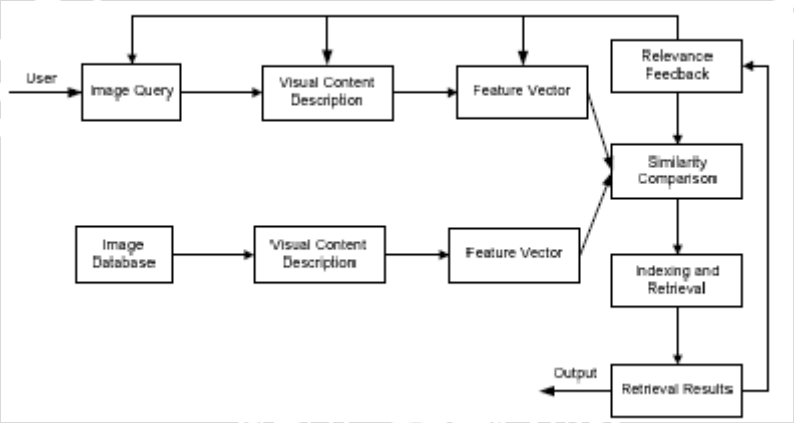
2.5 *Content Based Image Retrieval* (CBIR)

Sistem *Image Retrieval* pada awal pengembangannya pada tahun 1970-an masih menggunakan teks untuk menandai atau memberi keterangan pada citra. Pertama – tama citra diberi keterangan berbentuk teks kemudian untuk melakukan proses *retrieval* digunakan DBMS (*Database Management System*) berbasis teks. Kelemahan dari pemberian keterangan tersebut adalah jika koleksi citra memiliki jumlah yang sangat besar, maka menjadi tidak efisien karena proses dilakukan secara manual dan keterangan yang diberikan pada citra bersifat subyektif, sangat tergantung pada persepsi pemberi keterangan. Maka pada awal 1990-an mulai dikembangkan CBIR berdasarkan muatan visual. Muatan visual itu bisa berupa distribusi warna, tekstur atau bentuk yang dimiliki citra.

Muatan visual dalam basis data diekstrak, kemudian dideskripsikan sebagai vektor ciri dan disimpan dalam basis data ciri.Untuk mendapatkan kembali suatu citra, *user*menggunakan suatu sistem pengembalian yang menggunakan citra sebagai parameter pencariannya. Sistem tersebut akan mengekstrasi citra milik

*user*kedalam vektor fitur – fitur, kemudian akan dibandingkan dengan vektor fitur – fitur citra di dalam *image database* untuk menghitung derajat kemiripannya. Secara umum sistem CBIR dapat ditunjukkan pada gambar 2.3.

CBIR merupakan sebuah teknik yang menggunakan isi yang dikandung di dalam suatu citra untuk mencari citra – citra dari sekumpulan *image database* yang sesuai dengan permintaan *user* (Long, 2004).CBIR adalah suatu metode untuk pemanggilan kembali data citra berdasarkan ciri – ciri yang yang terkandung di dalam suatu citra. Teknik CBIR yang banyak digunakan adalah teknik berdasarkan warna, tekstur, dan bentuk (Wirawan ,2004).



Gambar 2.3 Diagram CBIR
(Sumber : Long, 2004)

2.6 Muatan Visual

Muatan suatu citra dapat diklasifikasikan menjadi dua, yaitu umum dan spesifik.Muatan visual yang bersifat umum meliputi fitur warna, tekstur dan bentuk. Muatan visual yang bersifat spesifik tergantung pada kebutuhan aplikasi, misalnya struktur wajah, sidik jari dan lain – lain. Karena persepsi terhadap citra bersifat subyektif, maka tidak ada satu fitur terbaik dari keseluruhan fitur yang dapat mempresentasikan citra.(Acharya, 2005).

2.6.1 Warna

Salah satu fitur yang paling banyak digunakan dalam sistem *image retrieval* adalah warna.Secara umum fitur ini hanya memperhatikan distribusi warna piksel –piksel pada suatu citra. Distribusi warna dapat di representasikan dalam bentuk histogram warna dan moment warna.Histogram warna merepresentasikan distribusi jumlah piksel untuk tiap intensitas warna dalam citra.Warna di kuantisasi menjadi beberapa level untuk mendefinisikan histogram kemudian untuk tiap level dihitung jumlah piksel yang nilainya sesuai. Dengan histogram dapat dicari citra yang memiliki kesamaan distribusi warna.Pengukuran tingkat kemiripan dilakukan dengan menghitung jarak antar histogram.

Selain dengan histogram warna, informasi distribusi warna secara umum dapat diketahui melalui tiga *low-order moment* warna, yaitu : *first-order moment* mewakili rata – rata warna, *second-order moment* mewakili standar deviasi warna dan *third-order moment* mewakili kecondongan warna (Acharya, 2005)

2.7 Transformasi Wavelet

Wavelet merupakan alat bantu matematis yang mampumelakukan dekomposisi terhadap sebuah fungsi secara terhirarkhi. *Wavelet*dapat digunakan untuk menggambarkan sebuah model atau citra asli ke dalam suatu fungsi matematis tanpa memperhatikan bentuk dari model merupakan citra, kurva, atau sebuah bidang.Transformasi *wavelet* merupakan sebuah fungsi konversi yang dapat membagi fungsi atau sinyal ke dalam komponen frekuensi atau skala yangberbeda, dan selanjutnya dapat dipelajari setiap komponennya tersebut dengan resolusi tertentu sesuaidengan skalanya.*Wavelet* mempunyai banyak jenis tergantung pada fungsi yang digunakannya seperti *Haar Wavelet*, *Symlet Wavelet*, *Deubechies Wavelet*, *Coifflet wavelet*, dan lain sebagainya. (Ramadijanti, 2006)

Transformasi *wavelet* telah digunakan sebagai ekstraksi ciri yang merupakan input bagi analisis tekstur dan penggolongan tekstur. Hal ini disebabkan karena wavelet mempunyai kemampuan membawa keluar ciri-ciri (*feature*) khusus pada suatu gambar yang diteliti. Pada transformasi *wavelet*, sebuah gambar didekomposisi

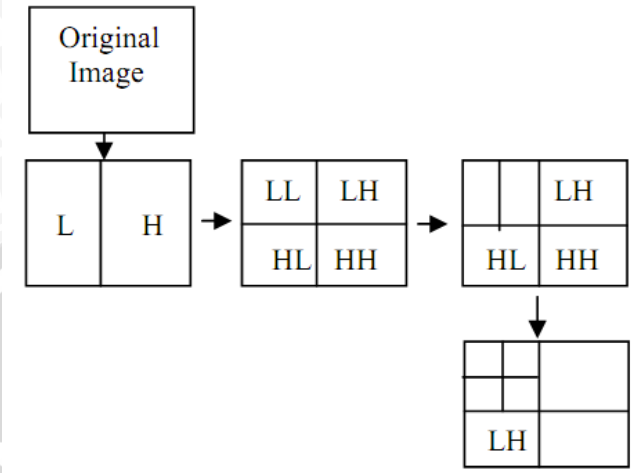
menjadi subgambar pada frekuensi dan orientasi yang berbeda, yaitu *low-low* (LL), *low-high* (LH), *high-low* (HL), dan *high-high* (HH).

Ada dua pendekatan yang dapat digunakan untuk mendekomposisi sebuah gambar tekstur dengan menggunakan transformasi *Wavelet* yaitu *Pyramid-Structured Wavelet Transform* (PWT) dan *Tree-Structured Wavelet Transform* (TWT). Pada*pyramid-structured wavelet transform*, proses dekomposisinya dilakukan hanya padasubband LL. Sedangkan pada *tree-structured wavelet transform*, proses dekomposisinya tidak dibatasi hanya pada subband LL, tetapi juga padasubband LH, HL atau HH. (Isa, 2007).

Transformasi *wavelet* merupakan proses mengubah sinyal ke dalam berbagai gelombang *wavelet* asli (*mother wavelet*) dengan berbagai pergeseran dan penyekalaan. Dengan demikian faktor skala memegang peranan yang sangat penting. Menurut Bruce dan Gao (Bruce dan Gao,1996), citra N × M merupakan data dua dimensi yang berbentuk matriks dengan elemennya berupa piksel–piksel penyusun citra. *Wavelet 2D* dapat dikonstruksikan dengan menggunakan *horisontal wavelet 1D* dan *vertikal wavelet 1D*.

Transformasi *wavelet* terhadap masing–masing piksel di dalam citra dapat dilakukan secara bergantian pada masing–masing kolom dan baris. Pertama kali dilakukan transformasi secara horisontal terhadap baris. Setelah itu dilakukan transformasi secara vertikal terhadap kolom. Langkah ini dilakukan secara bergantian sampai diperoleh koefisien aproksimasi dan koefisien detail dari citra. (Maimunah, 2007)

Transformasi *wavelet* dalam citra dapat diilustrasikan seperti pada gambar 2.4.



Gambar 2.4 Ilustrasi Transformasi *Wavelet* Dalam Citra
(Sumber : Maimunah, 2007)

2.8 Transformasi *Haar Wavelet*

Salah satu keluarga *wavelet* adalah *wavelet Haar*.Transformasi menggunakan fungsi Haar sebagai fungsi basis merupakan transformasi *wavelet* yang paling sederhana. Fungsi Haar didefinisikan seperti rumus 2.1 berikut:

$$\Psi(x) = \begin{cases} 1, 0 \leq x < \frac{1}{2} \\ -1, \frac{1}{2} \leq x < 1 \\ 0, \ lainnya \end{cases} \quad (2.1)$$

Transformasi *wavelet Haar* merupakan transformasi *wavelet* paling sederhana. Dalam transformasi *wavelet* Haar pada suatu citra dilakukan dengan menggunakan penapis lolos rendah (*Low Pass Filter* / LPF) dan penapis lolos tinggi (*High Pass Filter* / HPF) sehingga diperoleh koefisisen *wavelet*. (Maimunah, 2007)

Haar Wavelet adalah metode *wavelet* yang pertama kali diajukan oleh Alfred Haar pada tahun 1909.*Haar wavelet* adalah metode *wavelet* yang paling sederhana dan mudah untuk diimplementasikan. Untuk mengekstrak ciri-ciri tekstur dengan transformasi *Haar Wavelet*, dilakukan proses *averaging* untuk mendapatkan bagian dari gambar yang berfrekuensi rendah dan

dilakukan proses *differencing* untuk mendapatkan bagian dari gambar yang berfrekuensi tinggi. (Ramadijanti, 2006)

Secara umum, rumus untuk menghitung nilai *averaging* dan *differencing* dapat dinyatakan pada rumus 2.2 dan 2.3 :

$$H_0 : f(n) = \frac{X_n + X_{n+1}}{\sqrt{2}} \quad (2.2)$$

$$H_1 : f(n) = \frac{X_n - X_{n+1}}{\sqrt{2}} \quad (2.3)$$

Dengan: H_0 = *nilai averaging coefficient*
H_1= *nilai differencing coefficient*

Pertama, dilakukan transformasi *Haar Wavelet* pada setiap baris matriks pada gambar.Kemudian, dilakukan transformasi *Haar Wavelet* pada setiap kolom. Transformasi ini akan menghasilkan suatu gambar yang seperti terbagi menjadi empat bagian atau empat subband, yaitu subband LL, LH, HL dan HH. Kemudian dilakukan transformasi *Haar Wavelet* pada subband LL. Subband LL dibagi berdasarkan kolom menghasilkan subband LLL dan LLH. Kemudian subband LL dibagi berdasarkan baris, sehingga menghasilkan empat subband lainnya, yaitu LLLL,LLLH, LLHL, dan LLHH. Totalnya, ada 7 buahsubband yang dihasilkan yaitu LL, LH, HL, LLLL, LLLH, LLHL dan LLHH.

2.9 Metrika Multiresolusi (Metrika Lq)

Tingkat kemiripan antara citra *query* dengan citra pustaka dihitung dengan menggunakan metrika pencarian citra (*image querying metrics*). Jenis metrika yang digunakan untuk menghitung tingkat kemiripan adalah metrika multiresolusi.Metrika multiresolusi sering disebut juga metrika Lq. Metrika ini diperkenalkan oleh Jacob, Finkelstein, dan Salesin pada tahun 1995. Bentuk metrika Lq sebagai berikut :

$$\|Q, T\| = w_{0,0}|Q[0,0] - T[0,0]| + \sum_{i,j} w_{i,j}|\bar{Q}[i,j] - \bar{T}[i,j]| \quad (2.4)$$

dengan Q[0,0] dan T[0,0] berturut turut menyatakan koefisien fungsi

penskalaan untuk citra *query* dan citra pustaka yang berhubungan dengankeseluruhan intensitas rata-rata untuk setiap sistem koordinat warnasedangkan Q[i,j] dan T[i,j] berturut-turut menyatakan koefisien hasil dari alih ragam *haar wavelet* yang telah dipotong (*truncated*) dan dikuantisasi (*quantized*) dari citra Q dan T untuk setiap posisi i,j. Faktor wi,j merupakan faktor bobot rata-rata untuk setiap titik citra pada posisi i,j dalam sistem koordinat warna.
Menurut Widiartha dan Wijaya (Widiartha, I.B.K., dan Wijaya, I.G.P.S., 2006), maka persamaan 2.4 dapat disederhanakan menjadi

$$\|Hp - Hq\|^2 = \left(\sum_{i=0}^{n}(Hp_i - Hq_i)^2\right) \quad (2.5)$$

hp = nilai koefisien *wavelet* pada *image database*
hq = nilai koefisien *wavelet* pada citra uji.
pi = frekuensi kemunculan koefisien *wavelet* pada *image database*
qi = frekuensi kemunculan koefisien *wavelet* pada citra uji.

2.10 Pengurutan (*Selection Sort*)

Selection sort mencari dari elemen yang berikutnya sampai dengan elemen yang terakhir, jika ditemukan elemen lain yang lebih kecil dari elemen sekarang maka elemen yang bersangkutan akan ditukar, demikian seterusnya. Oleh karena itu setiap langkah pengurutan ini selalu memilih (*select*) satu dari barisan bilangan berikutnya.Metode perngurutan yang digunakan pada skripsi ini adalah metode *Selection Descending*.

2.11 Kuantisasi (*Quantized*) dan Pemotongan (*Truncated*)

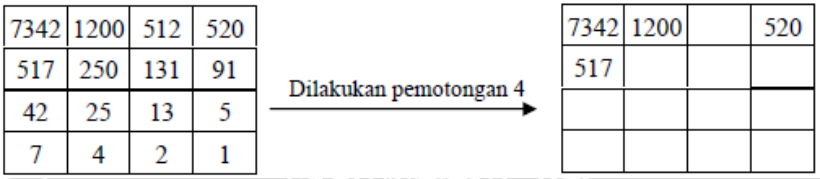
Dari istilahnya, kuantisasi merupakan pembulatan nilai tersampling kedalam level tertentu yang ditentukan oleh sistem. Dalam proses kuantisasi dilakukan proses normalisasi apabila nilai sampling diatas 1 atau dibawah 0, kemudian dilakukan pembulatan sesuai dengan nilai yang dikehendaki *user*. Proses normalisasi dapat dilihat pada persamaan 2.6.

$$x' = \frac{0.8\,(x-a)}{b-a} + 0{,}1 \quad (2.6)$$

Keterangan :

a = Nilai maksimum
b = Nilai minimum

Pada proses pemotongan,dilakukan pengambilan beberapa nilai besar untuk dilakukan proses berikutnya (Sugeng, 2004). Jika dalam satu citra dilakukan pemotongan 16, maka dari 16.384 nilai hasil transformasi akan diambil 16 nilai besar. Ilustrasi untuk proses pemotongan dapat dilihat pada gambar 2.5.

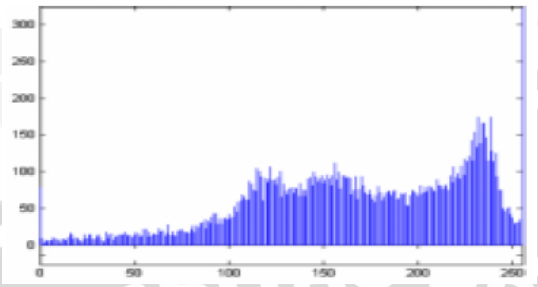


Gambar 2.5 Ilustrasi Pemotongan
(Sumber : Sugeng, 2004)

2.12 Histogram

Histogram merupakan suatu cara mempresentasikan data, dimana data dikelompokkan dalam bin – bin yang menunjukkan frekuensi kemunculan datanya, sehingga data menjadi sederhana dan mudah untuk mengambil informasi yang ada padanya.

Dari gambar 2.6 terlihat bahwa dalam suatu citra komponen penyusunan citra tersebut dikelompokkan atas bin – bin dengan frequensi kemunculannya. Histogram *wavelet* sedikit berbeda dengan histogram warna yang dilihat adalah frequensi kemunculan tiap warna dalam citra tersebut, sedangkan histogram *wavelet* tidak lagi menunjukkan warna melainkan nilai koefisien transformasi citra(Widiartha, I.B.K., dan Wijaya, I.G.P.S., 2006).



Gambar 2.6 Histogram warna suatu citra
(Sumber : Widiartha dan Wijaya, 2006)

2.13 Perhitungan Akurasi

Tingkat akurasi hasil pencarian dihitung menggunakan nilai ambang (*threshold value*) sesuai dengan persamaan 2.7 dan 2.8 (Widiartha, I.B.K., dan Wijaya, I.G.P.S., 2006).

$$Nilai\ ambang = \frac{1}{N*(N-p+1)}\ x\ 100\% \qquad (2.7)$$

Dengan **N** adalah ukuran basis data citra dan**p**merupakan posisi atau urutan citra yang ditemukan.Ini berarti bahwa citra yang ditemukan pada urutan pertama mempunyai tingkat kesuksesan 100 %, yang kedua mempunyai tingkat kesukesan 98 % dan seterusnya.

$$Nilai\ ambang = \frac{1}{N*(N-2+1)}\ x\ 100\% \qquad (2.8)$$

BAB III
METODOLOGI DAN PERANCANGAN SISTEM

3.1 Deskripsi Sistem

Perangkat lunak yang akan dibangun merupakan implementasi dari digital *image processing* menggunakan metode transformasi *Wavelet* dan Metrika Lq. Jenis transformasi *wavelet* yang digunakan adalah transformasi *Haar Wavelet*, berfungsi untuk mendapatkan fitur ciri citra, sedangkan metrika Lq digunakan pada proses pencocokan ciri antara citra uji dengan citra latih.

3.2 Sumber Data

Data yang digunakan sebagai pembuatan *image database* dan untuk pengujiannya adalah citra *true color* 24 bit berformat bitmap (*.bmp). Ukuran citra yang digunakan adalah 128 x 128 piksel. Data gambar didapat dari halaman http://wang.ist.psu.edu. Data asli yang diperoleh berupa citra berformat JPEG yang kemudian dikonversi ke format bitmap (*.bmp) agar lebih mudah pengolahan dalam proses selanjutnya.

Citra yang digunakan terdiri dari citra *true color*, antara lain :

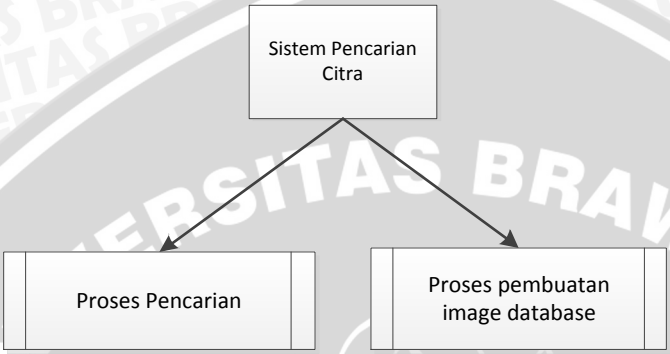
1. Citra Asmat 10 buah
2. Citra Beruang 10 buah
3. Citra Kupu - kupu 10 buah
4. Citra Kendaraan 10 buah
5. Citra Mawar 10 buah

3.3 Perancangan Proses

Pada perancangan sistem pencarian citra ini secara umum terdapat dua proses utama, yaitu proses pembuatan *image database* dan proses pencarian. Proses pembuatan *image database* adalah proses untuk memasukkan gambar yang akan menjadi rujukan. Tanpa adanya proses pembuatan *image database* maka sistem ini tidak akan bisa berjalan sesuai tujuannya.

Proses selanjutnya adalah proses pencarian. Pada proses ini dilakukan input gambar yang akan dicari kemiripannya dengan gambar – gambar yang ada dalam *database*.

Perancangan sistem secara umum dapat dilihat pada gambar 3.1.



Gambar 3.1 Diagram Alir Proses Perancangan Sistem Secara Umum

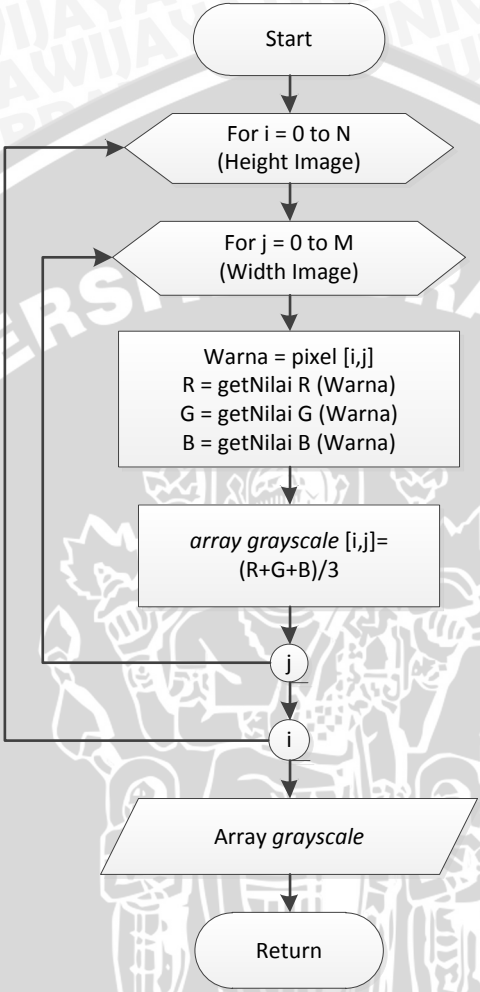
3.3.1 Pembuatan *Image Database*

Pada tahap ini dimulai dengan memasukkan citra latih (citra warna) yang kemudian akan dirubah formatnya citranya menjadi citra *grayscale*. Selanjutnya dilakukan proses dekomposisi *Haar Wavelet* sebanyak level dekomposisi dari citra tersebut, sehingga menghasilkan nilai koefisien citra. Diagram alir proses pembuatan *image database* dapat dilihat pada gambar 3.2.

Gambar 3.2 Diagram Alir Proses Pembuatan *Image Database*

3.3.1.1 Proses *Grayscale*

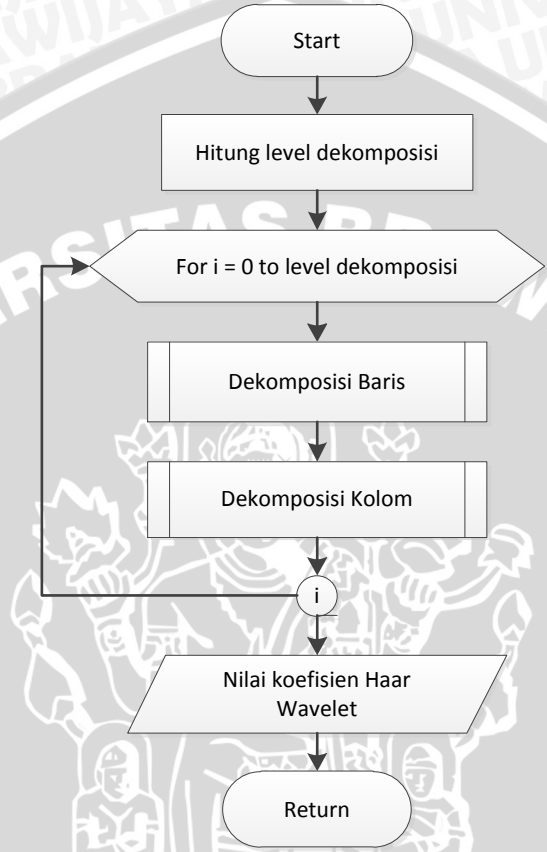
Pada tahap ini, citra warna dirubah formatnya menjadi citra *grayscale.* Proses *grayscale* dimulai dengan mengambil nilai warna (RGB) pada tiap – tiap piksel. Setelah itu nilai dari RGB tersebut dijumlah dan dirata – rata, serta disimpan pada *array*. Proses di atas dilakukan sebanyak jumlah piksel pada citra. Diagram alir untuk proses *grayscale* ditunjukan pada gambar 3.3.



Gambar 3.3 Diagram Alir Proses *Grayscale*

3.3.1.2 Dekomposisi Citra

Proses dekomposisi citra dimulai dengan menghitung banyak level dekomposisi. Selanjutnya untuk setiap level dilakukan proses dekomposisi baris dan dekomposisi kolom. Diagram alir untuk proses dekomposisi citra ditunjukan pada gambar 3.4.

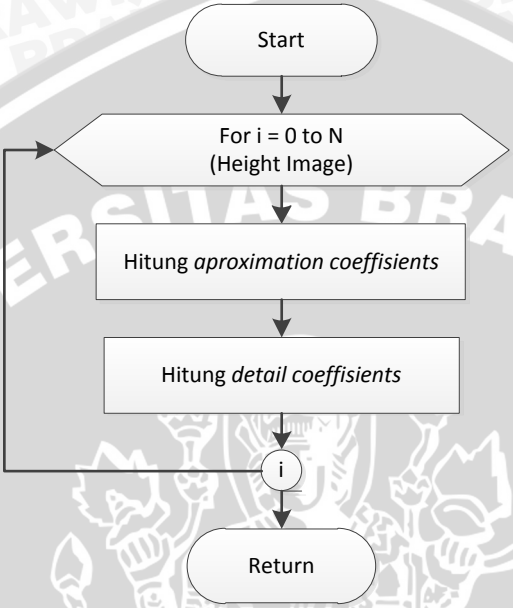


Gambar 3.4 Diagram Alir Proses Dekomposisi Citra

3.3.1.2.1 Dekomposisi Baris

Pada proses dekomposisi baris, data ditransformasi berdasarkan vektor barisnya. Langkah dekomposisi baris dimulai dengan menghitung nilai *approximation coefficients* menggunakan persamaan 2.2 dan menghitung nilai *detail coeffisients* menggunakan persamaan 2.3. Perhitungan dekomposisi dilakukan sebanyak jumlah baris pada citra.

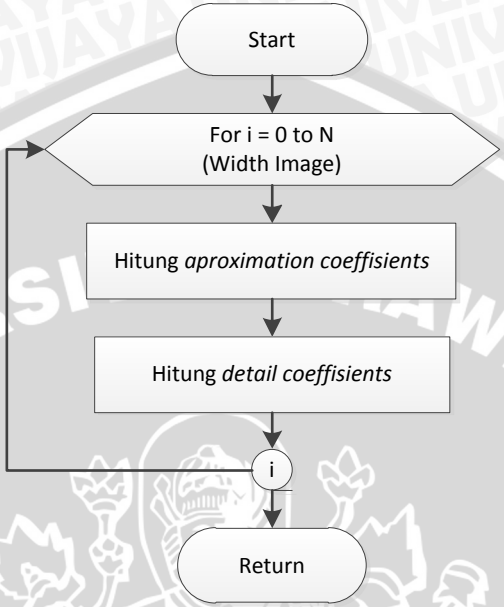
Diagram alir untuk proses dekomposisi baris ditunjukan pada gambar 3.5



Gambar 3.5 Diagram Alir Proses Dekomposisi Baris

3.3.1.2.2 Dekomposisi Kolom

Pada proses dekomposisi kolom data ditransformasi berdasarkan vektor kolomnya. Langkah dekomposisi kolom dimulai dengan menghitung nilai *approximation coefficients* menggunakan persamaan 2.2 dan menghitung nilai *detail coeffisients* menggunakan persamaan 2.3. Perhitungan dekomposisi dilakukan sebanyak jumlah kolom pada citra. Diagram alir untuk proses dekomposisi kolom ditunjukan pada gambar 3.6

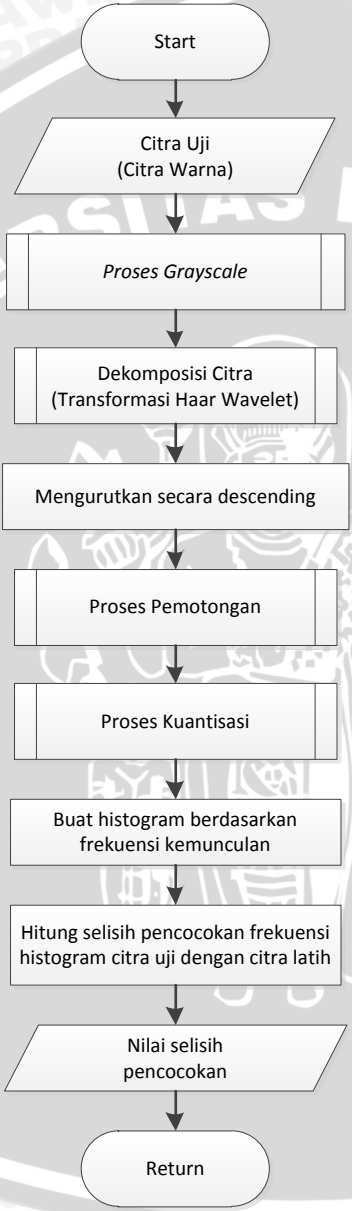


Gambar 3.6 Diagram Alir Proses Dekomposisi Kolom

3.3.2 Proses Pencarian

Pada proses pencarian dilakukan pencocokan citra uji dengan citra database berdasarkan cirinya. Proses pencarian dimulai menentukan ukuran koefisien pemotongan untuk *image database* dan citra uji. Kemudian nilai koefisien tersebut diurutkan secara *descending.* Hasil pengurutan kemudian dipotong sesuai dengan ukuran koefisien pemotongan dan dilakukan kuantisasi. Setelah itu hasil kuantisasi dikelompokkan (histogram) berdasarkan frekuensi kemunculannya. Selanjutnya dilakukan proses pada citra uji, Langkah pertama yang dilakukan adalah mengubah citra uji warna menjadi citra *grayscale*, proses tersebut bisa dilihat pada gambar 3.3. Kemudian dilakukan proses dekomposisi citra, proses tersebut bisa dilihat pada gambar 3.4. Selanjutnya dilakukan proses pemotongan dan kuantisasi, serta dibuat histogramnya. Langkah terakhir adalah menghitung selisih pencocokan frekuensi pencocokan histogram citra uji dengan

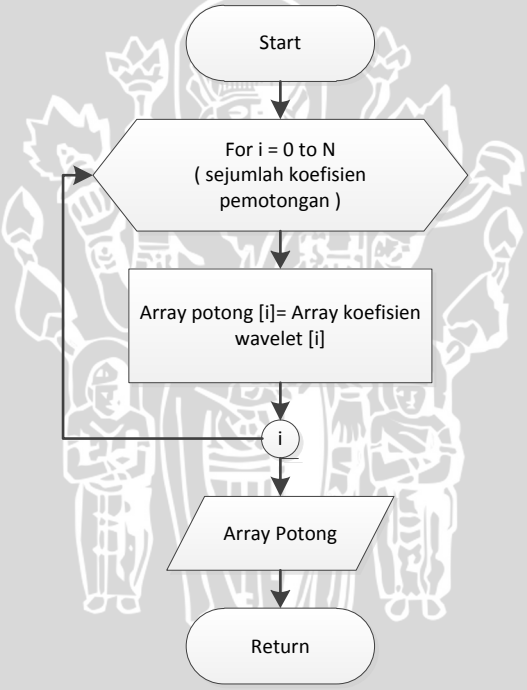
citra latih. Diagram alir proses pencarian dapat dilihat pada gambar 3.7.



Gambar 3.7 Diagram Alir Proses Pencarian

3.3.2.1 Pemotongan

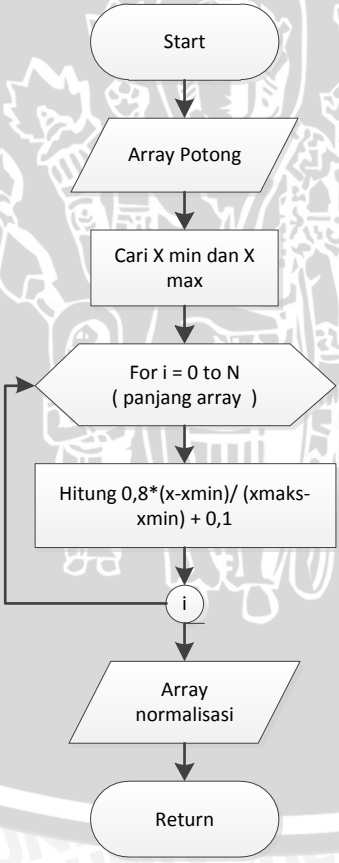
Proses pemotongan bertujuan untuk mendapatkan ciri – ciri citra yang dilakukan dengan cara memilih sebagian kecil (*m*) yang memiliki nilai magnitude terbesar saja, sehingga dapat mengurangi penggunaan ruang dan juga mempercepat proses pencarian. Diagram alir untuk proses pemotongan ditunjukan pada gambar 3.8.



Gambar 3.8 Diagram Alir Proses Pemotongan

3.3.2.2 Kuantisasi

Pada tahap ini, data yang telah di potong perlu dilakukan proses kuantisasi. Tujuannya adalah untuk mengurangi variasi data hasil transformasi. Setelah di kuantisasi diharapkan variasi data hanya akan berkisar antara 0-1 saja. Langkah pertama yang dilakukan pada proses kuantisasi yaitu menentukan nilai maksimum dan nilai minimum dari data hasil pemotongan. Kemudian menghitung kuantisasi untuk setiap indeks data menggunakan persamaan 2.4. Diagram alir untuk proses kuantisasi ditunjukan pada gambar 3.9.



Gambar 3.9 Diagram Alir Proses Kuantisasi

3.4 Struktur Data

Data pada *image database* diperoleh dari proses perhitungan yaitu berupa nilai koefisien dan frekuensi kemunculannya. Data hasil perhitungan tersebut akan di simpan dalam *file* berformat (*.txt) sesuai dengan nama *file* citra. Struktur tabel dapat dilihat pada tabel 3.1.

Tabel 3.1 Nilai Koefisien dan Frekuensi Kemunculannya

Layer	Fitur-1	Fitur-2	Fitur-3	Fitur-4	....	Fitur-N
Nilai koefisien						
Frekuensi						

Kumpulan *file-file* hasil ekstraksi tersebut merupakan sebuah *image database* yang akan digunakan sebagai data yang akan dicocokkan ketika *user* mencari citra yang dicari. Contoh file dapat dilihat pada Gambar 3.10.

file : image.txt

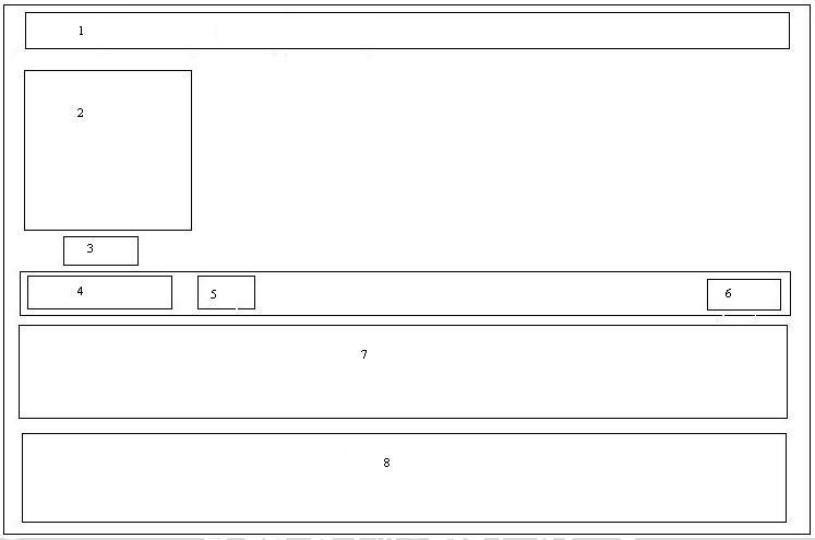
```
Citra.bmp
14535.1 5533.4 …
2     3
22234.3 8739.4 …
5     1
18073.5 615.8 …
4     2
```

Gambar 3.10 Nilai koefisien dan frekuensi kemunculan citra

3.5 Perancangan Antarmuka

Antarmuka pada sistem pencarian citra menggunakan transformasi Haar Wavelet dan Metrika Histogram Terurut ini terdiri 8 bagian yang ditunjukkan pada gambar 3.11, yaitu

1. Menu Navigasi
   Berisi opsi proses yang dapat dipilih oleh pengguna.
2. Daftar *File* Citra
   Berisi daftar *file-file* citra uji yang digunakan.
3. Tombol *Browse*
   Digunakan untuk memenentukan sumber data dari daftar citra yang akan diujikan kedalam sistem.
4. *Textbox* Koefisien Pemotongan
   Digunakan untuk menentukan besar koefisien yang akan digunakan pada proses pemotongan.
5. *ComboBox Wavelet*
   Digunakan untuk menentukan jenis koefisien wavelet yang akan digunakan
6. Tombol Mulai Pengujian
   Digunakan untuk memulai proses pencarian kemiripan file citra (*image query*) dengan citra di dalam *image database*
7. Daftar Hasil Pencarian
   Berfungsi untuk menampilan hasil pencarian citra
8. Bagian Proses Perhitungan
   Berfungsi untuk menampilan hasil perhitungan koefisien dari proses transformasi *Haar Wavelet file* citra uji dan citra *database.*



Gambar 3.11 Perancangan Antarmuka

3.6 Perancangan Uji Coba

Pengujian sistem ini bertujuan untuk menguji kelayakan sistem jika diterapkan secara langsung untuk pencarian citra. Pengujian dilakukan dengan memasukkan citra uji yang akan dicari hasil pencocokannya. Pengujian sistem menggunakan acuan Tabel 3.2, Tabel 3.3, dan Tabel 3.4.

Pengujian dilakukan untuk beberapa jenis citra uji yang berbeda, yaitu citra asli, citra *blur*, citra *edge*, dan citra *noise*. Koefisien pemotongan yang digunakan pada masing-masing pengujian, yaitu 64, 128, 256, 1024, 4096, 8192, dan 16384. Citra uji yang digunakan pada setiap pengujian mengambil 3 citra dari masing-masing kelompok citra.

Tabel 3.2 Hasil Pengujian Berdasarkan Jenis Citra

No	Citra	Jenis Citra								
		Nilai Potong Koefisien								
		64	128	256	512	1024	2048	4096	8192	16384

		%	%	%	%	%	%	%	%	%
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
	Rata-rata									

Pengujian data pada sistem pencarian ini dilakukan dengan menghitung tingkat akurasi pada setiap pengujian. Terdapat dua jenis perhitungan tingkat akurasi, yaitu tingkat akurasi berdasarkan jenis citra dan tingkat akurasi berdasarkan koefisien pemotongan. Perhitungan tingkat akurasi diperoleh berdasarkan persamaan 2.6.

Tabel 3.3 Tingkat Keakuratan berdasarkan Jenis Citra

Nilai Potong Koefisien	Tingkat Kesuksesan *Query* Citra (%)			
	Asli	Blur	Edge	Noise
64				
128				
256				
512				
1024				
2048				

4096				
8192				
16384				
Rata - Rata				

Tabel 3.4 Tingkat Keakuratan berdasarkan Koefisien Pemotongan

Nilai Potong Koefisien	Tingkat Kesuksesan *Query* Citra (%)				
	Asli	Blur	Edge	Noise	Rata - Rata
64					
128					
256					
512					
1024					
2048					
4096					
8192					
16384					

3.7 Contoh Perhitungan Manual

3.7 Perhitungan Tranformasi *Haar Wavelet*

Berikut ini akan dijelaskan contoh perhitungan manual transformasi *Haar Wavelet* pada tugas akhir ini. Diilustrasikan mempunyai citra warna, format bmp, dengan ukuran 8x8. Setelah diubah menjadi citra *grayscale*, data piksel awal yang dimiliki citra ini dimisalkan sebagai berikut

9	55	54	12	13	51	50	16
64	2	3	61	60	6	7	57
32	34	35	29	28	38	39	25
49	15	14	52	53	11	10	56
8	58	59	5	4	62	63	1
20	21	43	42	24	3	61	60

32	34	35	29	28	38	40	26
41	23	22	44	45	19	18	48

Dekomposisi matrik tersebut dengan transformasi *Haar Wavelet*. Pertama dilakukan penghitungan pada bagian baris dari matrik tersebut. Yakni dengan dilakukan terhadap fungsi basis matrik *Haar Wavelet* menggunakan Persamaan 2.2 dan 2.3. Dimisalkan perhitungan baris pertama

9	55	54	12	13	51	50	16

Maka perhitungan *average* dan *differencing* sebagai berikut
Average = (9+55)/√2;(54+12)/√2;(13+51)/√2;(50+16)/√2
Differencing = (9-55)/√2;(54-12)/√2;(13-51)/√2;(50-16)/√2

Karena ukuran matrik adalah 8x8 dimana $8 = 2^4$, maka *iterasi* kedalaman *level* dihitung mulai *level* 1 sampai *level* ke 4.

Hasil dari dekomposisi baris ditunjukkan sebagai berikut

32	33	32	33	-23	21	-19	17
33	32	33	32	31	-29	27	-25
33	32	33	32	-1	3	-5	7
32	33	32	33	17	-19	21	-23
33	32	33	32	-25	27	-29	31
20.5	42.5	13.5	60.5	-0.5	0.5	10.5	0.5
33	32	33	33	-1	3	-5	7
32	33	32	33	9	-11	13	-15

Langkah selanjutnya adalah menghitung dekomposisi kolom dari hasil matrik dekomposisi baris. Hasil ini merupakan hasil akhir dari perhitungan transformasi *Haar Wavelet*. Rumus yang digunakan sama dengan rumus pada dekomposisi baris yaitu mengacu pada persamaan 2.2 dan 2.3. Hasil dari dekomposisi kolom ditunjukkan sebagai berikut

32.5	32.5	32.5	32.5	4	-4	4	-4

32.5	32.5	32.5	32.5	8	-8	8	-8
26.75	37.25	23.25	46.25	-12.8	13.75	-9.25	15.75
32.5	32.5	32.5	33	4	-4	4	-4
-0.5	0.5	-0.5	0.5	-27	25	-23	21
0.5	-0.5	0.5	-0.5	-9	11	-13	15
6.25	-5.25	9.75	-14.3	-12.3	13.25	-19.75	15.25
0.5	-0.5	0.5	0	-5	7	-9	11

Data hasil perhitungan wavelet kemudian diubah menjadi *array* 1 dimensi, seperti berikut

Indeks ke	1	2	...	...	63	64
Nilai Koefisien	32.5	32.5	.....	....	7	-9

3.7 Proses Pengurutan

Pada tahap ini, koefisien hasil transformasi *Haar Wavelet* yang telah dijadikan *array* 1 dimensi diurutkan dari nilai terbesar hingga nilai terkecil, sehingga diperoleh hasil sebagai berikut

Indeks ke	1	2	3	4	5	.....	63	64
Nilai Koefisien	46	37.25	33	32.5	32.5	.....	-23	-27

3.7 Proses Pemotongan

Pada tahap ini, data yang telah diurutkan kemudian dipotong sesuai dengan koefisien pemotongan yang telah ditentukan. Sebagai simulasi, misalnya kita telah menentukan koefisien pemotongan 5, artinya data yang diambil adalah 5 data dari indeks ke 1 sampai indeks ke 5.

Indeks ke	1	2	3	4	5	.....	63	64
Nilai Koefisien	46	37,25	33	32,5	32,5	.....	-23	-27

3.7 Perhitungan Proses Kuantisasi

Pada tahap ini, data setelah dipotong tentu mempunyai variasi yang banyak. Untuk mengurangi variasi tersebut perlu dilakukan normalisasi. Rumus yang digunakan untuk normalisasi sesuai dengan persamaan 2.6. Misal data yang akan di normalisasi menggunakan data hasil proses pemotongan, sehingga perhitungan normalisasi dapat dilihat sebagai berikut

Untuk x' ke 1 = $x' = \frac{0.8(46 - 32,5)}{46 - 32,5} + 0.1 = 0.9$

Untuk x' ke 2 = $x' = \frac{0.8(37,25 - 32,5)}{46 - 32,5} + 0.1 = 0.3$

Untuk x' ke 3 = $x' = \frac{0.8(33 - 32,5)}{46 - 32,5} + 0.1 = 0.1$

Untuk x' ke 4 = $x' = \frac{0.8(32,5 - 32,5)}{46 - 32,5} + 0.1 = 0.1$

Untuk x' ke 5 = $x' = \frac{0.8(32,5 - 32,5)}{46 - 32,5} + 0.1 = 0.1$

Maka diperoleh hasil dari normalisasi seperti berikut ini

X	1	2	3	4	5
Nilai Koefisien	0,9	0,3	0,1	0,1	0,1

3.7 Histogram

Pada tahap ini, koefisien hasil normalisasi dikelompokkan sehingga kita bisa mengetahui berapa frekuensi kemunculan dari masing – masing koefisien histogram. Misal menggunakan data hasil normalisasi sehingga frekuensi dari histogram terurutnya menjadi seperti berikut

Nilai Koefisien	0.9	0.3	0.1

Frekuensi Kemunculan	1	1	2

Dari hasil diatas, dapat diketahui bahwa frekuensi kemunculan nilai koefisien 0,9 adalah 1 kali, 0,3 adalah 1 kali dan 0,1 adalah 2 kali.

3.7 Perhitungan Pencarian Citra

Pada tahap ini, proses pencocokan merupakan tahap terakhir dari proses pencarian citra. Proses pencocokan dilakukan dengan cara mencari selisih antara hasil koefisien dan frekuensi kemunculan *image Query* dengan *image database*. Rumus yang digunakan untuk menghitung selisih tersebut adalah rumus metrika LQ yang telah di jelaskan pada persamaan 2.5.

Misalkan data awal dari *image query* dan *image database* ditunjukkan seperti berikut ini

Data Awal *Image Database*

Nilai Koefisien	0.9	0.3	0.1
Frekuensi Kemunculan	1	1	2

Data Awal *Image Query*

Nilai Koefisien	0.9	0.3	0.1
Frekuensi Kemunculan	3	2	1

Maka proses perhitungan pencocokan adalah :

$$|| Hp - Hq ||^2 = \left(\sum_{i=0}^{n} (hpi - hqi)^2 \right)$$

$$|| Hp - Hq ||^2 = (1 * 0.9 - 3 * 0.9)^2 + (1 * 0.3 - 2 * 0.3)^2 + (2 * 0.1 - 1 * 0.1)^2$$

$$|| Hp - Hq ||^2 = 3,2 + 0,09 + 0.01 = 3,22$$

Sehingga selisih hasil koefisien histogramnya adalah 3,22 yang selanjutnya disebut nilai *error*. Semakin kecil nilai *error* (nilai *error* mendekati 0) maka semakin mirip *image query* (citra uji) dengan *image database* (citra latih). Sehingga dari hasil perhitungan pencocokan dapat disimpulkan bahwa *image* tersebut berbeda dengan *image* yang dimaksud.

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

4.1 Lingkungan Implementasi

Implementasi merupakan proses transformasi representasi rancangan ke dalam bahasa pemrograman yang dapat dimengerti oleh komputer. Pada bab ini, lingkungan implementasi yang akan dijelaskan meliputi lingkungan implementasi perangkat keras dan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan sistem CBIR citra *true color* menggunakan Haar *Wavelet Transform* ini adalah sebuah Laptop dengan spesifikasi sebagai berikut :

1. *Processor* Intel® Core(TM) 2 Duo T5750 @ 2.0 GHz
2. 2048 MB RAM
3. 320 GB HDD
4. *Monitor* 14.1”
5. *Keyboard*
6. *Mouse*

Perangkat keras ini akan difungsikan sebagai tempat perangkat lunak untuk penelitian dijalankan.

4.1.2 Lingkungan Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan sistem penghitungan jumlah kendaraan ini adalah :

1. Sistem operasi *Microsoft Windows XP Professional Edition Service Pack 2* sebagai tempat aplikasi dijalankan.
2. *Borland Delphi 7.0* sebagai *software development* dalam mengembangkan aplikasi untuk menghasilkan data citra hasil pencarian dengan metode ini.
3. *MS Access* sebagai aplikasi database untuk menyimpan data *file image database*

4.2.4 Implementasi Program

Berdasarkan analisa dan perancangan proses yang terdapat pada bab 3, maka pada subbab ini akan dijelaskan implementasi proses-proses tersebut.

4.2.1 Implementasi Pembuatan *Image Database*

Tahap awal dari pembuatan *image database* adalah memilih citra yang akan digunakan sebagai data latih, kemudian disimpan informasinya pada suatu database. Proses ini berada pada form master. Pada proses ini ada beberapa procedure yang digunakan, antara lain prosedur untuk *load* citra, penyimpanan dan hapus data. Prosedur untuk pembuatan data awal bisa dilihat pada gambar 4.1.

```
// prosedur pembuatan image database

procedure TfomMasterImage.btnOpenClick(Sender: TObject);
begin
     if OpenPictureDialog.Execute then
     begin
          img_open.Picture.LoadFromFile(OpenPictureDialog.FileName);
          if img_open.Picture.Graphic <> nil then
          begin
               eNama.Text :=
ExtractFileName(OpenPictureDialog.FileName);
               i_lebar := img_open.Picture.Bitmap.Width;
               i_tinggi := img_open.Picture.Bitmap.Height;
               eResolusi.Text := IntToStr(i_lebar) + ' x ' +
IntToStr(i_tinggi);
               i_ukuran :=
Round(FileSizeByName(OpenPictureDialog.FileName) / 1000);
               eUkuran.Text := FloatToStr(i_ukuran);
               eKet.Clear;
               btnSimpan.Caption := 'Simpan';

               s_source := OpenPictureDialog.FileName;
               s_image := path_image +
ExtractFileName(OpenPictureDialog.FileName);
          end;
     end;
end;
```

```
// procedure untuk memanggil citra yang telah tersimpan pada
database.

procedure TfomMasterImage.LoadImage();
var
   sqlLoadImage : TADOQuery;
   i : Integer;
```

```
begin
     sqlLoadImage := TADOQuery.Create(nil);
     sqlLoadImage.Connection := DM.ADOConnection;

     sqlLoadImage.Close;
     sqlLoadImage.SQL.Text := 'SELECT * FROM tImage';
     sqlLoadImage.Open;

     dbGridMasterImage.ClearRows;
     img_database.Clear;

     if sqlLoadImage.FieldByName('id_image').AsInteger <> 0 then
     begin
          i := 0;

          while not sqlLoadImage.Eof do
          begin
               with dbGridMasterImage do
               begin
                    AddRow();
                    BeginUpdate;

                    if FileExists(path_image +
sqlLoadImage.FieldByName('file_name').AsString) then
                    begin
                         img_temp.Picture.LoadFromFile(path_image +
sqlLoadImage.FieldByName('file_name').AsString);
                         img_database.Insert(i,
ImgResize(img_temp).Picture.Bitmap,nil);

//img_database.Insert(sqlLoadImage.FieldByName('id_image').AsInteger
- 1, ImgResize(img_temp).Picture.Bitmap,nil);

                         Cell[1,LastAddedRow].AsInteger :=
sqlLoadImage.FieldByName('id_image').AsInteger;
                         Cell[2,LastAddedRow].AsInteger := i;
                         //Cell[2,LastAddedRow].AsInteger :=
sqlLoadImage.FieldByName('id_image').AsInteger - 1;
                         Cell[3,LastAddedRow].AsString := 'Nama ' +
#32#32#32#32#32#32#32#32#32#32 + ': ' +
sqlLoadImage.FieldByName('file_name').AsString;
                         Cell[4,LastAddedRow].AsString := 'Resolusi '
+ #32#32#32#32#32#32 + ': ' +
sqlLoadImage.FieldByName('lebar').AsString +
                                                        ' x ' +
sqlLoadImage.FieldByName('tinggi').AsString + ' Pixels';
                         Cell[5,LastAddedRow].AsString := 'Ukuran ' +
#32#32#32#32#32#32#32#32 + ': ' +
sqlLoadImage.FieldByName('ukuran').AsString + ' KB';
                         Cell[6,LastAddedRow].AsInteger :=
sqlLoadImage.FieldByName('lebar').AsInteger;
                         Cell[7,LastAddedRow].AsInteger :=
sqlLoadImage.FieldByName('tinggi').AsInteger;
```

```
                        Cell[8,LastAddedRow].AsFloat :=
sqlLoadImage.FieldByName('ukuran').AsFloat;
                        Cell[9,LastAddedRow].AsString := 'Keterangan
' + #32 + ': ' + sqlLoadImage.FieldByName('ket').AsString;
                        Cell[10,LastAddedRow].AsInteger := 2;
                    end;
                    EndUpdate;
                end;
                sqlLoadImage.Next;
                Inc(i);
            end;
        end;
        FreeAndNil(sqlLoadImage);
```

Gambar 4.1 Prosedur *Pembuatan Image Database*

Pada proses ini, data citra yang telah tersimpan pada *database* diambil untuk selanjutnya diproses sebagai sumber data pada tahap *training*. Semua citra yang telah dipilih di *load* dan ditampilkan pada form *training*.

4.2.2 Implementasi Proses *Grayscale*

Tahap ini adalah proses pengambilan warna tiap *pixel* dari suatu citra warna. Selanjutnya citra tersebut diubah menjadi citra *gray*. Prosedure untuk mengubah citra RGB ke citra Gray ditunjukan pada gambar 4.2

```
// procedure untuk mendapatkan nilai RGB

function Gray(rgb : TColor) : Byte;
var
   r, g, b : Byte;
begin
     r := GetRValue(rgb);
     g := GetGValue(rgb);
     b := GetBValue(rgb);

     Result := Round(r * PERCENT_RED + g * PERCENT_GREEN + b *
PERCENT_BLUE);
end;

// prosedur untuk convert RGB ke Grayscale

function ConvertToGrayScale(img_rbg : TImage) : TImage;
var
   img_gray : TImage;
   x, y : Integer;
```

```
   rgb_gray : Byte;
begin
     img_gray := TImage.Create(nil);
     img_gray.Width := img_rbg.Picture.Bitmap.Width;
     img_gray.Height := img_rbg.Picture.Bitmap.Height;

     for x := 0 to img_rbg.Picture.Bitmap.Height do
     begin
          for y := 0 to img_rbg.Picture.Bitmap.Width do
          begin
               rgb_gray := Gray(img_rbg.Canvas.Pixels[x,y]);
               img_gray.Canvas.Pixels[x,y] :=
RGB(rgb_gray,rgb_gray,rgb_gray);
          end;
     end;
     Result := img_gray;
end;
```

Gambar 4.2 Prosedur Ubah Citra Warna Ke Citra *Grayscale*.

Langkah selanjutnya adalah mengambil nilai piksel dari citra *grayscale* dan menyimpannya pada sebuah *array* 2 dimensi. Prosedure yang digunakan ditunjukkan pada gambar 4.3.

```
function ImageToArray(img_gray : TImage) : ar_image2D;
var
   x, y : Integer;
   ar_2D : ar_image2D;
begin


SetLength(ar_2D,img_gray.Picture.Bitmap.Height,img_gray.Picture.Bitma
p.Width);
     SetLength(ar_temp,img_gray.Picture.Bitmap.Height,
img_gray.Picture.Bitmap.Width);

     for x := 0 to img_gray.Picture.Bitmap.Height - 1 do
      begin
          for y := 0 to img_gray.Picture.Bitmap.Width - 1 do
           begin

ar_2D[x,y]:=GetRValue(img_gray.Canvas.Pixels[x,y]);
          end;
     end;
     Result := ar_2D;
end;
```

Gambar 4.3 Prosedur Pengambilan Nilai Piksel Citra *Grayscale*

4.2.3 Implementasi Haar *Wavelet Transform*

Tahap ini terdapat tiga buah prosedur yang saling terkait untuk mendekompisi nilai *array* pada masing-masing jenis warna RGB. Prosedur awal merupakan prosedur dekomposisi yang digunakan untuk menghitung nilai *approximation coefficients* dan *detail coefficients* pada tiap pasang nilai pada masing-masing *array* warna. Proses ini menggunakan Persamaan 2.2 dan 2.3. Pada proses ini dibagi dua jenis, yakni pada proses penghitungan pada baris dan kolom. Prosedur yang digunakan ditunjukkan pada Gambar 4.4

```
procedure DecompositionStep(img_size : Integer; index : Integer;
level : Integer; posisi : String);
var
    i : integer;
    h : Integer;
begin

// dekomposisi pada baris

      h := level;
      if (posisi = 'row') then
      begin

// membagi tiap baris menjadi 2 bagian

            for i := 0 to (Round(h/2) - 1) do
            begin
                  //TempArray[index, i] := RoundTo((ArrayRed[index,
2*i] + ArrayRed[index, 2*i+1])/Sqrt(2),-1);
                  //TempArray[index, round(h/2)+i] :=
RoundTo((ArrayRed[index, 2*i ] - ArrayRed[index, 2*i+1])/Sqrt(2),-1);

                  ar_temp[index, i] := RoundTo((ar_pixel[index, 2*i]
+ ar_pixel[index, 2*i+1]) / Sqrt(2), -2);
                  ar_temp[index, Round(h/2) + i] :=
RoundTo((ar_pixel[index, 2*i] - ar_pixel[index, 2*i+1]) / Sqrt(2), -
2);
            end;

// penyalinan hasil perhitungan dari array sementara ke array awal
            for i := 0 to img_size - 1 do
            begin
                  ar_pixel[index, i] := ar_temp[index, i];
                  //ArrayRed[index, i] :=TempArray[index,i];
            end;
      end

// dekomposisi pada kolom
```

```
      else if (posisi = 'col') then
      begin

// membagi tiap kolom menjadi 2 bagian

            for i := 0 to (Round(h/2) - 1) do
            begin
                  ar_temp[i, index] := RoundTo((ar_pixel[2*i, index]
+ ar_pixel[2*i+1, index]) / Sqrt(2), -2);
                  ar_temp[Round(h/2) + i, index] :=
RoundTo((ar_pixel[2*i, index] - ar_pixel[2*i+1, index]) / Sqrt(2), -
2);
            end;
// penyalinan hasil perhitungan dari array sementara ke array asal
            for i := 0 to img_size - 1 do
            begin
                  ar_pixel[i, index] := ar_temp[i, index];
                  //ArrayRed[i, index] := TempArray[i, index];
            end
      end;
end
```

Gambar 4.4 Prosedur Dekomposisi

Proses selanjutnya adalah proses pengulangan proses dekomposisi untuk tiap yaitu dari *level* 2^1 sampai 2^7 sesuai dengan ukuran citra yang digunakan. Pada kasus ini dimulai pada nilai *level* 1 sampai *level* 7 karena citra berukuran 128 (2^7). Prosedur yang digunakan ditunjukkan pada Gambar 4.5.

```
procedure Decomposition(level : Integer; index : Integer; posisi :
String);
var
   h : integer;
begin
     h := level;
 // perulangan dekomposisi tiap level dari 2^0 sampai 2^n
     while  h > 1 do
     begin
          DecompositionStep(level, index, h, posisi);
          h := Round(h/2);
     end;
end;
```

Gambar 4.5 Prosedur Perulangan Level Dekomposisi

Semua proses dekomposisi dilakukan pada tiap baris dan kolom sebanyak jumlah baris dan kolom pada citra. Prosedur iterasi pada baris dan kolom ditunjukkan pada Gambar 4.6.

```
procedure StandartDecomposition(img_size : Integer);
var
   i, j, row, col : integer;
begin

     for i := 0 to img_size - 1 do
     begin
          row := i;
          Decomposition(img_size, row, 'row');
     end;

     for i := 0 to img_size - 1 do
     begin
          col := i;
          Decomposition(img_size, col, 'col');
     end;
end;
```

Gambar 4.6 Prosedur Perulangan Dekomposisi Baris dan Kolom

Hasil perhitungan nilai koefisien *wavelet* ditampilkan pada form training dan disimpan pada sebagai data *training* pada database. Prosedure untuk penyimpanan pada *database* ditunjukkan pada gambar 4.7.

```
sqlValid.SQL.Text := 'SELECT * FROM tTrainning WHERE id_image = ' +
                                        IntToStr(img_id);
                    sqlValid.Open;

                    if sqlValid.FieldByName('id_trainning').AsInteger
= 0 then
                    begin
                         sqlMaxId.Close;
                         sqlMaxId.SQL.Text := 'SELECT
MAX(id_trainning) As max_id FROM tTrainning';
                         sqlMaxId.Open;

                         if sqlMaxId.FieldByName('max_id').AsInteger
= 0 then
                            max_id := 0
                         else
                            max_id :=
sqlMaxId.FieldByName('max_id').AsInteger;
```

```
                        sqlInsert.Close;
                        sqlInsert.SQL.Text := 'INSERT INTO
tTrainning VALUES ' +
                                              '(' +
                                              IntToStr(max_id + 1) +
',' + IntToStr(img_id) +
                                              ',' + QuotedStr(value)
+
                                              ')';
                        sqlInsert.ExecSQL;
                    end
                    else
                    begin
                        sqlInsert.Close;
                        sqlInsert.SQL.Text := 'UPDATE tTrainning SET
feature = ' +
                                              QuotedStr(value) + '
WHERE id_image = ' +
                                              IntToStr(img_id);
                        sqlInsert.ExecSQL;
                    end;
                    Application.ProcessMessages;
                    Inc(j);
                end;
            end;
        end;
        FreeAndNil(sqlInsert);
        FreeAndNil(sqlValid);
        FreeAndNil(sqlMaxId);
end;
```

Gambar 4.7 Penyimpanan Hasil Perhitungan Koefisien *Wavelet*

4.2.4 Implementasi Pencocokan

Pada proses pencocokan, langkah awal yang dilakukan adalah memanggil citra uji yang akan dijadikan citra *query*, kemudian citra tersebut di ubah menjadi citra *grayscale*, hasil dari proses ini adalah sebuah image *grayscale*. Fungsi untuk mengubah citra uji menjadi citra *grayscale* bisa dilihat pada gambar 4.8.

```
function ConvertToGrayScale(img_rbg : TImage) : TImage;
var
   img_gray : TImage;
   x, y : Integer;
   rgb_gray : Byte;
begin
     img_gray := TImage.Create(nil);
     img_gray.Width := img_rbg.Picture.Bitmap.Width;
     img_gray.Height := img_rbg.Picture.Bitmap.Height;
```

```
    for x := 0 to img_rbg.Picture.Bitmap.Height do
    begin
         for y := 0 to img_rbg.Picture.Bitmap.Width do
         begin
              rgb_gray := Gray(img_rbg.Canvas.Pixels[x,y]);
              img_gray.Canvas.Pixels[x,y] :=
RGB(rgb_gray,rgb_gray,rgb_gray);
         end;
    end;
    Result := img_gray;
end;
```

Gambar 4.8 Fungsi Mengubah Citra Warna ke Citra *Grayscale*

Selanjutnya ambil nilai piksel dari citra tersebut. Fungsi yang digunakan untuk mengambil nilai masing – masing piksel dapat dilihat pada gambar 4.9.

```
function ImageToArray(img_gray : TImage) : ar_image2D;
var
   x, y : Integer;
   ar_2D : ar_image2D;
begin

     SetLength(ar_2D, img_gray.Picture.Bitmap.Height,
img_gray.Picture.Bitmap.Width);
     SetLength(ar_temp, img_gray.Picture.Bitmap.Height,
img_gray.Picture.Bitmap.Width);

     for x := 0 to img_gray.Picture.Bitmap.Height - 1 do
     begin
          for y := 0 to img_gray.Picture.Bitmap.Width - 1 do
          begin
               ar_2D[x,y]:=GetRValue(img_gray.Canvas. Pixels[x,y]);
          end;
     end;
     Result := ar_2D;
end;
```

Gambar 4.9 Fungsi Pengambilan Nilai Piksel Pada Citra Uji.

Selanjutnya lakukan proses transformasi *wavelet* pada citra uji. Prosedur yang digunakan untuk menghitung nilai koefisien wavelet tersebut dapat dilihat pada gambar 4.10.

```
procedure StandartDecomposition(img_size : Integer);
var
   i, j, row, col : integer;
```

```
begin

     for i := 0 to img_size - 1 do
     begin
          row := i;
          Decomposition(img_size, row, 'row');
     end;

     for i := 0 to img_size - 1 do
     begin
          col := i;
          Decomposition(img_size, col, 'col');
     end;
end;

procedure Decomposition(level : Integer; index : Integer; posisi :
String);
var
   h : integer;
begin
     h := level;
     while  h > 1 do
     begin
          DecompositionStep(level, index, h, posisi);
          h := Round(h/2);
     end;
end;


procedure DecompositionStep(img_size : Integer; index : Integer;
level : Integer; posisi : String);
var
    i : integer;
    h : Integer;
begin
      h := level;
      if (posisi = 'row') then
        begin
            for i := 0 to (Round(h/2) - 1) do
            begin
                ar_temp[index, i] := RoundTo((ar_pixel[index, 2*i] +
                ar_pixel[index, 2*i+1]) / Sqrt(2), -2);
                ar_temp[index, Round(h/2) + i] :=
                RoundTo((ar_pixel[index, 2*i] - ar_pixel[index,
                2*i+1]) / Sqrt(2), -2);
            end;
        for i := 0 to img_size - 1 do
                begin
                  ar_pixel[index, i] := ar_temp[index, i];
            end;
end
      else if (posisi = 'col') then
      begin
```

```
            for i := 0 to (Round(h/2) - 1) do
            begin
                ar_temp[i, index] := RoundTo((ar_pixel[2*i, index] +
                ar_pixel[2*i+1, index]) / Sqrt(2), -2);
                ar_temp[Round(h/2) + i, index] :=
                RoundTo((ar_pixel[2*i, index] - ar_pixel[2*i+1,
                index]) / Sqrt(2), -2);
            end;

            for i := 0 to img_size - 1 do
            begin
                  ar_pixel[i, index] := ar_temp[i, index];
                  //ArrayRed[i, index] := TempArray[i, index];
            end
      end;
end;
```

Gambar 4.10 Prosedur Perhitungan *Wavelet* Pada Citra Uji.

Setelah itu dilakukan proses pengurutan secara *descending*. Pada kasus ini, gunakan pengurutan dengan *selection sort* Prosedur untuk tahap ini bisa dilihat pada gambar 4.11.

```
procedure SortArray(var ar_1D : ar_image1D);
var
   i, j : Integer;
   max, temp : Real;
   index : Integer;
begin
     for i := Length(ar_1D) - 1 downto 0 do
     begin
          max := ar_1D[i];
          index := i;

          for j := i-1 downto 0 do
          begin
               if ar_1D[j] < max then
               begin
                    max := ar_1D[j];
                    index := j;
               end;
          end;

          if index <> i then
          begin
               temp := ar_1D[index];
               ar_1D[index] := ar_1D[i];
               ar_1D[i] := temp;
          end;
     end;
end;
```

Gambar 4.11 Prosedur Proses Pengurutan

Setelah itu kita memilih koefisien pemotongan. Proses ini bertujuan untuk memilih ciri – ciri suatu citra yang dilakukan dengan memilih sebagian kecil nilai koefisien yang memiliki nilai magnitude terbesar saja. Proses ini ditunjukan pada gambar 4.12

```
SetLength(ar_feature_use, StrToInt(cbKoefisien.Text));
// set array

     for i := 0 to StrToInt(cbKoefisien.Text) - 1 do
// get array just used
     begin
          ar_feature_use[i] := ar_feature[i];
     end;
```

Gambar 4.12 Proses Pemilihan Nilai Pemotongan Koefisien oleh *User*.

Setelah nilai ditentukan, maka dilakukan proses kuantisasi. Prosedure yang digunakan ditunjukkan pada gambar 4.13.

```
procedure Normalisasi(var arTemp : ar_image1D);
var
   min, max : Real;
   i : Integer;
begin
     min := getMinimum(arTemp);
     max := getMaximum(arTemp);

     for i := 0 to Length(arTemp) - 1 do
     begin
          arTemp[i] := RoundTo(((0.8*(arTemp[i] - Min)) / (Max -
Min)) + 0.1, -1);
     end;
```

Gambar 4.13 Proses Kuantisasi

Data dari proses kuantisasi tersebut merentang antara 0,9 sampai 0,1. Selanjutnya kita membuat suatu record untuk menyimpan data dan frekuensi. *Record* tersebut bisa dilihat pada gambar 4.14.

```
Type
```

```
    PFrekFeature = ^FrekFeature;
    FrekFeature = record
      value : Real;
      frek : Integer;
    end;
```

Gambar. 4.14 *Record* Untuk Menyimpan Data Koefisien Dan Frekuensinya.

Setelah itu mulai menghitung frekuensi histogramnya. Hasil perhitungan tersebut disimpan pada suatu *templist*. Prosedur yang digunakan bisa dilihat pada gambar 4.15.

```
function TfoActionRecognize.FrekFeature(arTemp : ar_image1D): TList;
var
   temp_list : Tlist;
   temp_a : PFrekFeature;
   i, j, index : Integer;
   bSama : Boolean;
begin
     temp_list := TList.Create;

     temp_list.Clear;

     for i := 1 to 9 do
     begin
          New(temp_a);
          temp_a^.value := i/10;
          temp_a^.frek := 0;
          temp_list.Add(temp_a);
     end;

     for i := 0 to Length(arTemp) - 1 do
     begin
          for j := 0 to temp_list.Count - 1 do
          begin
               temp_a := temp_list.Items[j];
               if arTemp[i] = temp_a^.value then
                  temp_a^.frek := temp_a^.frek + 1;
          end;
     end;
     Result := temp_list;
end;
```

Gambar 4.15 Prosedur Menghitung Frekuensi Histogram Citra.

Setelah nilai data koefisien dan frekuensi citra uji diketahui. Selanjutnya *load* data training dan buat suatu *record* yang digunakan untuk menyimpan data gambar dan hasil perhitungan *wavelet* semua

citra yang telah disimpan. Data tersebut di ubah menjadi array 1 dimensi dan dilakukan proses normalisasi. Proses selanjutnya sama dengan yang dilakukan pada data citra uji. Hasil histogram dan frekuensinya disimpan. Hasil dari proses ini selanjutnya akan digunakan sebagai sumber pencocokan histogram antara citra uji dengan citra *training.* Prosedur yang digunakan dapat dilihat pada gambar 4.16.

```
procedure TfoActionRecognize.LoadDataTrainning;
var
   sqlLoadTrainning : TADOQuery;
   arTemp : ar_image1D;
begin
     sqlLoadTrainning := TADOQuery.Create(nil);
     sqlLoadTrainning.Connection := DM.ADOConnection;

     sqlLoadTrainning.Close;
     sqlLoadTrainning.SQL.Text := 'SELECT tTrainning.id_trainning,
tTrainning.id_image, tImage.file_name, tTrainning.feature FROM
tTrainning LEFT JOIN tImage ' +
                                  'ON tTrainning.id_image =
tImage.id_image ';
     sqlLoadTrainning.Open;

     if sqlLoadTrainning.FieldByName('id_trainning').AsInteger > 0
then
     begin
          list_db_feature.Clear;
          while not sqlLoadTrainning.Eof do
          begin
               New(temp_list);
               temp_list^.rec_id_image :=
sqlLoadTrainning.FieldByName('id_image').AsInteger;
               temp_list^.rec_file_image :=
sqlLoadTrainning.FieldByName('file_name').AsString;
               temp_list^.rec_feature :=
sqlLoadTrainning.FieldByName('feature').AsString;

               arTemp :=
ParsingFeature(Trim(sqlLoadTrainning.FieldByName('feature').AsString)
+ ' ', StrToInt(cbKoefisien.Text));

               Normalisasi(arTemp);

               temp_list^.rec_ar_feature := arTemp;

               temp_list^.rec_ar_feature_frek := FrekFeature(arTemp);

               list_db_feature.Add(temp_list);
```

```
                sqlLoadTrainning.Next;
            end;
        end;
        FreeAndNil(sqlLoadTrainning);
end;
```

Gambar 4.16 Prosedur *Load Data Training*

Selanjutnya dilakukan proses pencocokan. Pada tahap ini hitung perbedaan nilai dan frekuensi masing – masing data training dengan citra uji. Hasilnya kemudian diurutkan berdasarkan nilai error yang paling kecil sampai nilai error yang terbesar. Fungsi ini didasarkan pada persamaan di bab 2.11. Fungsi yang digunakan bisa dilihat pada gambar 4.17.

```
for i := 0 to list_db_feature.Count - 1 do
// proses similarity
     begin
          temp_list := list_db_feature.Items[i];
          temp_list^.rec_similarity :=
Similarity(temp_list^.rec_ar_feature_frek, list_query);
     end;

     for i := list_db_feature.Count - 1 downto 0 do

 // hasil dari proses tersebut diurutkan

 begin
          temp_list := list_db_feature.Items[i];
          min_error := temp_list^.rec_similarity;
          index := i;

          for j := i - 1 downto 0 do
          begin
               temp_list := list_db_feature.Items[j];
               error := temp_list^.rec_similarity;

               if error > min_error then
               begin
                    min_error := error;
                    index := j;
               end;
          end;

          if index <> i then
          begin
               list_db_feature.Exchange(index,i);
          end;
     end;
```

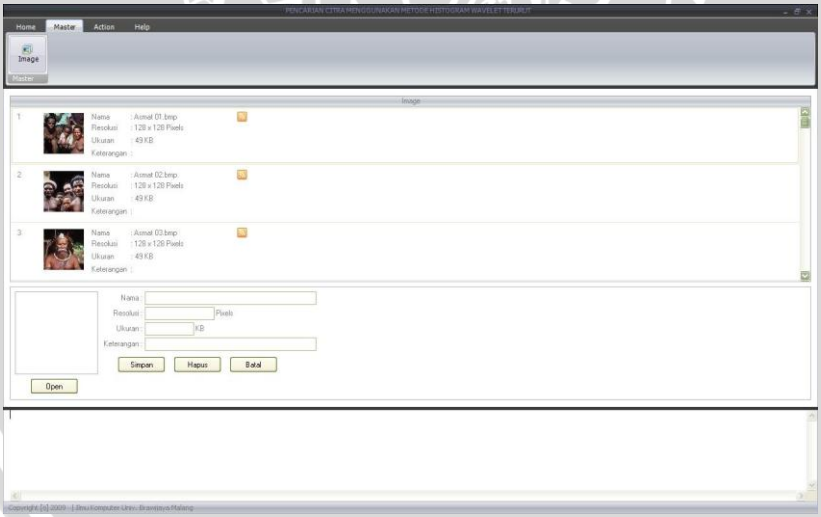
```
function TfoActionRecognize.Similarity(list_db : TList; list_img :
Tlist) : Real;
var
   i : Integer;
   temp : Real;
   temp_a, temp_b : PFrekFeature;
begin
     temp := 0;
     for i := 0 to list_img.Count - 1 do
     begin
          temp_a := list_db.Items[i];
          temp_b := list_img.Items[i];

          temp := temp + Power((temp_a^.value * temp_a^.frek) -
(temp_b^.value * temp_b^.frek),2)
     end;
     Result := temp;
end;
```

Gambar 4.17 Fungsi Pencocokan

4.3 Implementasi Antar Muka

Tampilan *form* antar muka awal dapat dilihat pada gambar 4.18

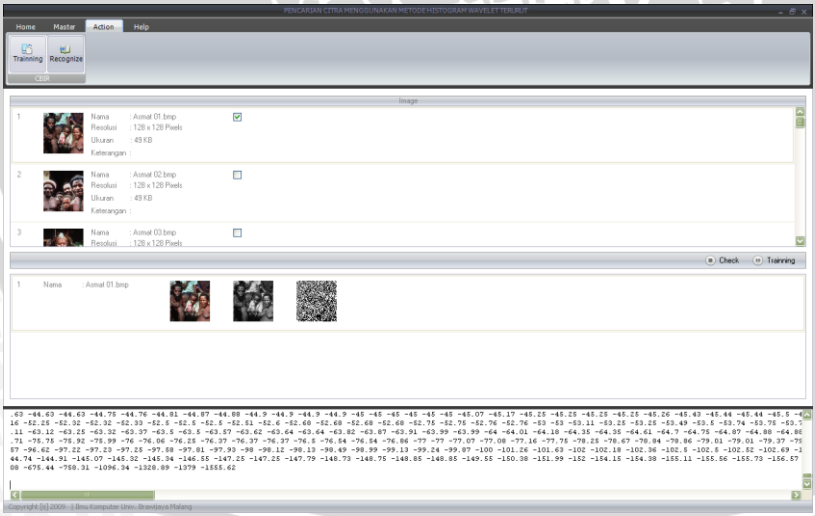


Gambar 4.18 Antar Muka Awal

Pada *Form* Master terdapat beberapa tombol, yang pertama adalah tombol *image*. Tombol ini berfungsi untuk memanggil semua citra

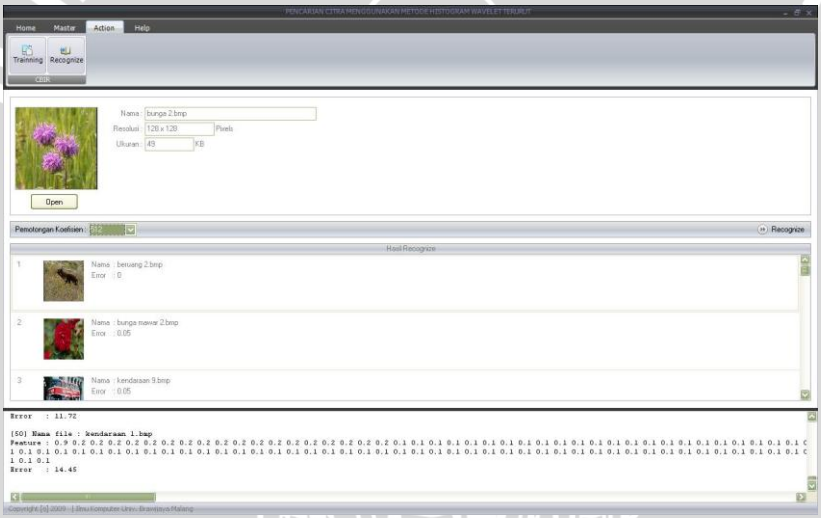
data *training* yang telah tersimpan pada *database*, citra – citra tersebut kemudian ditampilkan pada sebuah panel. Citra yang ditampilkan juga berisi informasi tentang nama, resolusi, dan ukurannya. Tombol *open* berfungsi untuk memanggil citra baru yang ingin dijadikan data training. Tombol *simpan* berfungsi untuk menyimpan citra baru yang ingin digunakan sebagai citra training. Tombol *hapus* berfungsi untuk menghapus salah satu citra yang ingin dihapus dan telah tersimpan pada *image database*. Tombol *batal* berfungsi untuk membatalkan semua aktivitas pada *form* master.

Saat *form training* pertama kali dibuka, ditampilkan semua citra yang telah tersimpan pada image database pada sebuah panel image. Citra yang ditampilkan disertai dengan informasi seperti nama, resolusi dan ukuran masing – masing citra. Tombol *check* pada *form* ini digunakan oleh user untuk menandai semua image data citra yang akan di*training*. Jika user ingin memilih citra tertentu, maka user bisa menekan tombol *cekbox* yang ada di sebelah kanan citra yang diinginkan. Tombol *training* digunakan untuk memproses *training* data citra yang telah dipilih. Hasil dari training tersebut ditampilkan dalam bentuk citra training serta perhitungan pengujiannya. *Form training* ditunjukkan pada gambar 4.19



Gambar 4.19 Antar Muka *Form Training*

Pada *form recognize* terdapat beberapa tombol yang bisa digunakan. Tombol *open* digunakan untuk membuka citra yang akan diujikan. Citra uji ini kemudian ditampilkan pada suatu panel beserta informasi nama, resolusi dan ukurannya. *Combo box* pemotongan koefisien digunakan oleh user untuk menentukan berapa koefisien yang akan digunakan pada proses pengujian ini. Pemilihan dilakukan dengan cara menekan *combo box* tersebut dan akan ditampilkan beberapa nilai seperti 64, 128 ... 16384. Tombol *recognize* digunakan user untuk melakukan proses pencocokan. Setelah proses selesai, hasilnya ditampilkan dalam bentuk hasil citra yang ditemukan dan diurutkan sesuai dengan nilai error terkecil sampai terbesar, serta hasil perhitungan pencocokannya.



Gambar 4.20 Antar Muka *Form* Pencocokan

4.4 Implementasi Uji Coba

Pada subbab ini akan dilakukan pembahasan mengenai pengujian yang telah dilakukan pada sistem dan hasil evaluasi dari ringkasan hasil sistem.

4.4.1 Skenario Pengujian

Pada pengujian ini digunakan sampel citra yang diperoleh dari situs http://wang.ist.psu.edu. Citra yang digunakan adalah citra *true color* 24 bit beresolusi 128 x 128 *pixel* dengan format bitmap (*.bmp). Wavelet yang digunakan adalah Haar Wavelet. Tiap citra dihitung nilai koefisien dan frekuensi histogramnya. Selanjutnya hitung tingkat akurasi citra berdasarkan nilai pemotongan koefisiennya mulai dari pemotongan 64, 128 … 16384. Pencocokan dilakukan dengan mencari selisih hasil histogram citra *query* dengan hasil histogram citra *database*. Jika selisih antara kedua hasil histogram tadi semakin kecil, maka semakin mirip kedua citra tersebut. Namun jika selisihnya semakin besar, maka kedua citra tersebut tidak mirip dan dapat disimpulkan kalau citra yang ada itu bukanlah citra yang dicari.

Citra yang digunakan terdiri dari citra *true color* :

1. Citra Asmat 10 buah
2. Citra Beruang 10 buah
3. Citra Mawar 10 buah
4. Citra Kendaraan10 buah
5. Citra kupu-kupu 10 buah

Citra yang digunakan untuk pengujian adalah citra asli, citra *blur*, citra *edge* dan citra *noise*. Citra yang digunakan untuk pengujian diambil masing – masing 3 buah dari masing – masing kelompok.

4.4.2 Hasil Pengujian

Berikut ini adalah hasil pencarian citra pada *image database* yang memiliki kemiripan dengan citra pada *image query*. Pengujian dilakukan terhadap 3(tiga) citra pada semua jenis citra yang ada. Ditampilkan hasil tingkat kesuksesan pencarian citra berdasarkan nilai koefisien pemotongannya. Pada Tabel 4.1 berikut disajikan hasil pengujian pada citra asli, sedangkan pada Tabel 4.2 adalah hasil pengujian dengan citra uji berupa citra *blur*. Pada Tabel 4.3 merupakan hasil pengujian dari citra uji berupa citra yang di beri efek *edge*. Pada Tabel 4.4 merupakan hasil pengujian dari citra yang diberi *noise*.

Tabel 4.1 Hasil pengujian citra asli berdasarkan nilai pemotongan koefisien.

No	Citra	CITRA ASLI								
		Nilai Potong Koefisien								
		64	128	256	512	1024	2048	4096	8192	16384
		%	%	%	%	%	%	%	%	%
1	Asmat 01	100	100	100	100	100	100	100	100	100
2	Asmat 03	96	96	96	96	96	96	100	100	100
3	Asmat 05	100	100	96	98	100	98	98	100	100
4	Beruang 04	100	98	100	100	100	100	100	100	100
5	Beruang 07	100	100	100	100	96	100	100	98	100
6	Beruang 09	98	98	98	100	96	98	96	98	100
7	Kupu 01	94	100	98	100	100	98	100	98	100
8	Kupu 03	100	100	100	100	100	100	100	100	100
9	Kupu 10	98	98	100	94	98	100	100	96	100
10	Kendaraan 01	100	100	100	100	100	100	100	100	100
11	Kendaraan 04	96	98	100	98	100	100	100	98	100
12	Kendaraan 09	98	100	100	98	98	98	98	100	100
13	Mawar 01	98	98	98	100	100	100	100	98	100
14	Mawar 03	98	100	100	100	100	100	100	100	100
15	Mawar 09	100	100	98	100	98	98	98	98	100
	Rata-rata	98.4	99.1	98.9	98.9	98.8	99.1	99.3	98.9	100

Tabel 4.2 Hasil pengujian citra *blur* berdasarkan nilai pemotongan koefisien

No	Citra	Citra BLUR								
		Nilai Potong Koefisien								
		64	128	256	512	1024	2048	4096	8192	16384
		%	%	%	%	%	%	%	%	%
1	Asmat 01	100	100	72	84	82	98	100	100	100
2	Asmat 03	96	96	96	96	96	96	100	100	98
3	Asmat 05	100	100	100	98	100	98	98	100	100
4	Beruang 04	100	66	100	100	92	100	100	100	100
5	Beruang 07	100	100	100	100	96	100	100	98	100
6	Beruang 09	98	98	98	100	82	84	86	98	100
7	Kupu 01	94	90	78	90	100	86	86	88	98

8	Kupu 03	98	100	100	100	100	100	100	100	88
9	Kupu 10	98	98	100	60	66	90	88	86	100
10	Kendaraan 01	98	100	100	98	100	100	100	100	100
11	Kendaraan 04	82	88	80	88	80	34	34	34	100
12	Kendaraan 09	98	100	100	98	98	98	98	100	98
13	Mawar 01	90	98	98	100	100	100	100	100	100
14	Mawar 03	98	100	96	100	100	100	100	100	94
15	Mawar 09	100	100	90	92	98	98	98	98	98
	Rata-rata	96.7	95.6	93.9	93.6	92.7	92.1	92.5	93.5	98.27

Tabel 4.3 Hasil pengujian citra *edge* berdasarkan nilai pemotongan koefisien

No	Citra	Citra *Edge*								
		Nilai Potong Koefisien								
		64	128	256	512	1024	2048	4096	8192	16384
		%	%	%	%	%	%	%	%	%
1	Asmat 01	38	42	60	80	100	100	90	90	72
2	Asmat 03	34	50	62	72	94	36	28	28	12
3	Asmat 05	84	56	72	60	72	98	40	38	38
4	Beruang 04	36	54	32	86	64	96	96	68	32
5	Beruang 07	32	36	54	52	60	96	96	42	84
6	Beruang 09	32	32	38	46	78	98	52	36	64
7	Kupu 01	40	40	50	72	20	20	16	14	32
8	Kupu 03	4	18	100	94	90	92	92	92	84
9	Kupu 10	70	68	68	64	34	34	30	32	38
10	Kendaraan 01	6	2	2	2	80	96	98	98	90
11	Kendaraan 04	96	92	94	34	28	70	80	74	20
12	Kendaraan 09	12	6	4	4	6	4	4	4	82
13	Mawar 01	46	32	12	6	6	6	6	6	6
14	Mawar 03	52	10	10	14	12	12	12	12	48
15	Mawar 09	18	2	2	8	8	8	8	8	32
	Rata-rata	40	36	44	46.3	50.1	57.7	49.9	42.8	48.93

Tabel 4.4 Hasil pengujian citra *noise* berdasarkan nilai pemotongan koefisien

No	Citra	Citra Noise

		Nilai Potong Koefisien								
		64	128	256	512	1024	2048	4096	8192	16384
		%	%	%	%	%	%	%	%	%
1	Asmat 01	88	94	90	38	30	16	14	14	78
2	Asmat 03	34	54	96	96	96	96	100	100	98
3	Asmat 05	84	68	78	62	66	82	84	84	94
4	Beruang 04	100	98	52	68	56	66	66	66	96
5	Beruang 07	76	60	66	80	84	94	96	90	92
6	Beruang 09	98	74	62	76	78	80	80	94	96
7	Kupu 01	82	100	86	90	90	92	90	98	92
8	Kupu 03	16	16	88	74	82	84	80	90	22
9	Kupu 10	98	84	92	52	58	80	86	92	92
10	Kendaraan 01	18	8	22	30	70	58	90	98	92
11	Kendaraan 04	96	80	76	66	70	32	22	30	98
12	Kendaraan 09	98	100	100	98	98	98	98	100	48
13	Mawar 01	98	96	98	100	100	100	96	92	48
14	Mawar 03	86	92	94	88	90	90	92	96	72
15	Mawar 09	100	100	98	92	94	92	94	92	100
	Rata-rata	78.1	74.9	79.9	74	77.5	77.3	79.2	82.4	81.2

Dari tabel 4.1 diketahui bahwa tingkat akurasi yang didapatkan cenderung stabil yaitu berkisar antara 94% hingga 100 %. Pada tabel 4.2 diketahui bahwa tingkat akurasi yang didapatkan cenderung sedikit fluktuatif yaitu berkisar antara 80 – 100 %. Pada beberapa citra uji yang memiliki tingkat akurasi rendah pada beberapa nilai potong koefisien tertentu. Misal pada citra uji kendaraan 04 didapatkan hasil 82%, 88%, 80%, 88%, 80 %, 34%, 34%, 34% dan 100%.

Pada tabel 4.3, tingkat akurasi yang didapatkan sangat fluktuatif. Misal pada citra uji kendaraan 01 didapatkan hasil 6%, 2%, 2%, 2%, 80 %, 96%, 98%, 98% dan 90%.

Pada tabel 4.4 diketahui bahwa tingkat akurasi yang didapatkan cenderung fluktuatif terhadap nilai potong koefisien. Jika nilai potong koefisien semakin besar, maka tingkat akurasi yang didapat ada yang menurun, ada yang meningkat bahkan ada juga yang naik dan turun. Misal pada citra uji asmat 01, tingkat akurasinya semakin menurun. Pada citra uji asmat 03, tingkat

akurasinya semakin meningkat. Sedangkan pada citra uji beruang 09, tingkat akurasinya fluktuatif.

4.4.3 Analisa Hasil

Dari hasil pengujian maka dapat dilakukan beberapa analisa. Persentase tingkat akurasi berdasarkan koefisien pemotongannya dapat dilihat pada tabel 4.5.

Dari tabel 4.5 tersebut dapat dilihat bahwa tingkat akurasi tertinggi didapatkan pada jenis citra asli. Hal ini disebabkan ciri yang dihasilkan dari transformasi haar wavelet memiliki lebih banyak informasi sehingga berpengaruh pada saat pencocokan citra. Sedangkan untuk tingkat akurasi terendah didapatkan pada jenis citra uji edge. Hal ini terjadi karena ciri yang dihasilkan dari transformasi haar wavelet tidak memiliki banyak informasi sehingga hasil dari pencocokan rendah.

Tabel 4.5 Persentase Keakuratan Pencarian Citra

Nilai Potong Koefisien	Tingkat Kesuksesan Query Citra (%)			
	ASLI	BLUR	EDGE	NOISE
64	94.80	96.67	40.00	78.13
128	99.07	95.60	36.00	74.93
256	98.93	93.87	44.00	79.87
512	98.93	93.60	46.27	74.00
1024	98.80	92.67	50.13	77.47
2048	99.07	92.13	57.73	77.33
4096	99.33	92.53	49.87	79.20
8192	98.93	93.47	42.80	82.40
16384	100.00	98.27	48.93	81.20
Rata - Rata	98.65	94.31	46.19	78.28

Jumlah pemilihan nilai potong koefisien tidak terlalu mempengaruhi tingkat keakuratan pencarian secara signifikan. Hal ini disebabkan karena berapapun nilai potong koefisien yang digunakan, apabila ciri citra yang digunakan mengandung banyak informasi maka tingkat akurasi yang dihasilkan akan tetap tinggi. Sebaliknya, jika ciri citra yang digunakan rendah, maka untuk

berapapun nilai potong koefisien yang digunakan, tingkat akurasi yang dihasilkan akan tetap rendah.

Dari hasil analisa diketahui bahwa tingkat akurasi rata – rata dari masing – masing jenis citra adalah 98,65% untuk citra Asli, 94,31% untuk citra *blur*, 46,19% untuk citra *edge* dan 78,28% untuk citra *noise*.

BAB V
KESIMPULAN DAN SARAN

5.1 KESIMPULAN

Dari hasil analisis terhadap pengujian yang dilakukan, maka dapat ditarik kesimpulan sebagai berikut :

1. Pencarian citra menggunakan *Haar Wavelet* sebagai proses *preprocessing* dan Metrika Lq sebagai proses pencocokan memberikan hasil terbaik pada jenis citra uji asli. Tingkat akurasi rata-rata yang didapatkan untuk jenis citra asli sebesar 98,65%, citra *blur* 94,31%, citra *edge* 46,19% dan untuk citra *noise* 78,28%.
2. Jumlah pemilihan nilai potong koefisien tidak terlalu mempengaruhi tingkat keakuratan pencarian secara signifikan.
3. Tingkat akurasi dipengaruhi oleh ciri yang dimiliki oleh citra. Semakin banyak perlakuan yang diberikan pada citra (*noise , blur*, *edge*), maka ciri yang dimiliki akan semakin sedikit sehingga tingkat akurasi yang didapatkan semakin rendah.

5.2 SARAN

Pengembangan lebih lanjut yang dapat dilakukan dari penelitian ini antara lain :

1. Ukuran citra yang digunakan agar lebih bervariasi.
2. Penggunaan jenis transformasi *wavelet* yang lain agar mendapatkan tingkat akurasi yang lebih tinggi.
3. Pencarian citra ini dapat dikembangkan untuk citra uji dengan perubahan berbagai posisi, misal dirotasi 45^0.

DAFTAR PUSTAKA

Acharya T.,dan Ray A. K. 2005. *Image processing Principle and Applications*. John Wiley & Sons.

Balza,A. dan Firdausy.K. 2005. *Teknik Pengolahan Citra Digital Menggunakan Delphi*. Penerbit Ardi. Yogyakarta.

Long, F, dan Hongjiang,Z. 2004. *Fundamentals Of Content-Based Image Retrieval*.

Madhaveelatha,L.Y., Jinaga,B.C. dan Reddy, V.S.K.. 2007. *Content Based Color Image Retrieval Via Wavelet Transform*. IJCSNS International Journal of Computer Science and Network Security, Vol.7 No.12, December 2007.

Maimunah dan Harjoko.A. 2007. *Sistem Pengenalan Iris Mata Manusia Menggunakan Transformasi Wavelet*. Yogyakarta : Seminar Nasional Aplikasi Teknologi Informasi.

Medison, A.S.,.2007. *Sistem Pembanding Citra Pas Foto Dengan Metode Transformasi Wavelet*. Medan : Universitas Sumatera Utara.

Morton, Peggy, dan Arne Petersen.1997. *Image Compression Using the Haar Wavelet Transform*.

Nugraha,A.P. dan Mutiara,A.B. 2006. *Metode ekstraksi data untuk pengenalan huruf dan angka tulisan tangan dengan menggunakan jaringan syaraf buatan propagasi balik*. Jurusan Teknik Informatika, Fakultas Teknologi Industri, Universitas Gunadarma.

Porwik, Piotr, dan Agnieszka.L. 2004. *The Haar Wavelet Transform in Digital Image Processing : Its Status and Achievments.* Jurnal Machine graphics & Vision.

Ramadijanti,N. 2006. *Content Based Image Retrival Berdasarkan Ciri Tekstur Menggunakan Wavelet*. Yogyakarta : Seminar Nasional Aplikasi Teknologi Informasi.

Sugeng,.2004. *Pencarian Citra Dengan Metode Discrete Cosine Transform (DCT) dan Metrika Lq*, Jurusan Teknik Elektro Unram. Mataram.

Suhendra, A. 2007. *Catatan Kuliah Pengantar Pengolahan Citra*. Fakultas Teknik Informatika Universitas Kristen Duta wacana.

Stollnitz, dan Eric J. 2001. *Wavelets for Computer Graphics: A Primer Part 1*. University of Washington. United State.

Walidainy, Hubbul dan Nazlum. 2004. *Simulasi Menghapus Derau Pada Sinyal Suara*. Jurusan Elektro Fakultas Teknik Universitas Syiah Kuala.

Widiartha,I.B.K., dan Wijaya,I.G.P.S. 2006. *Pencarian Citra Menggunakan Metode Transformasi Wavelet dan Metrika Histogram Terurut*. Jurusan Elektronik Fakultas Teknik Universitas Mataram.

Wirawan,S. 2004. *Content Based Image Information Retrieval*. Universitas Gunadarma. Jakarta.

LAMPIRAN

Lampiran 1. Citra Latih

Citra Asmat










Citra Beruang









Citra Mawar










Citra Kendaraan








Citra Kupu - Kupu










Lampiran 2. Citra Uji

1. Citra Asli





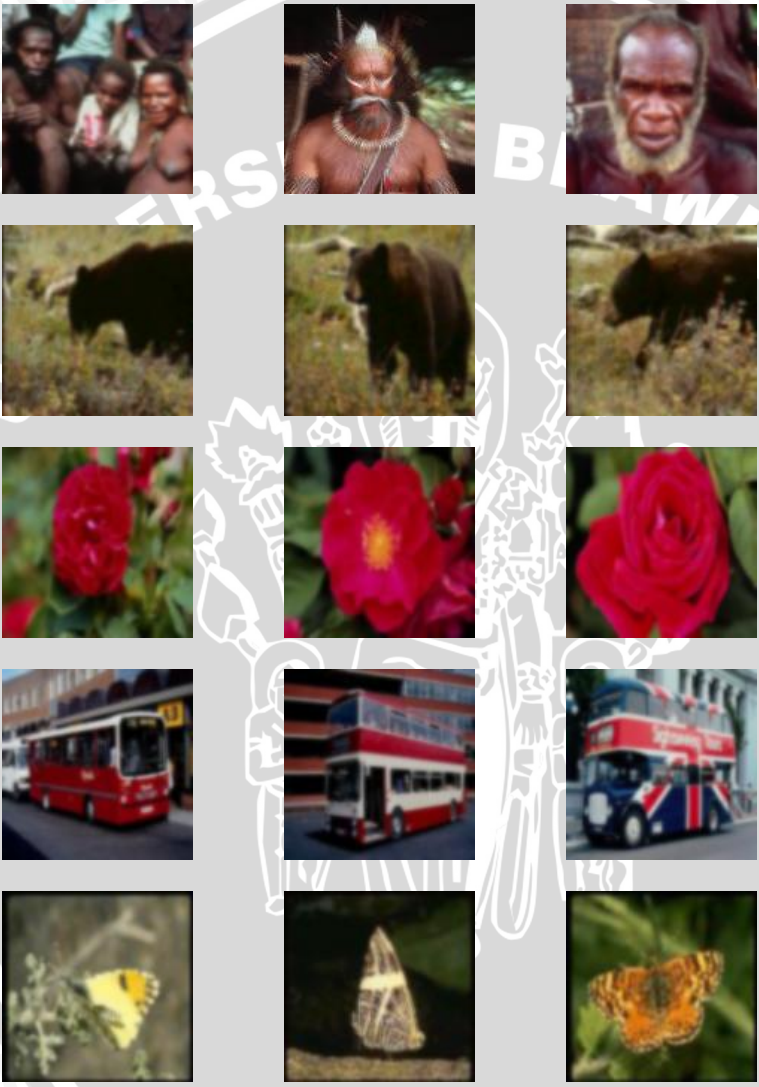




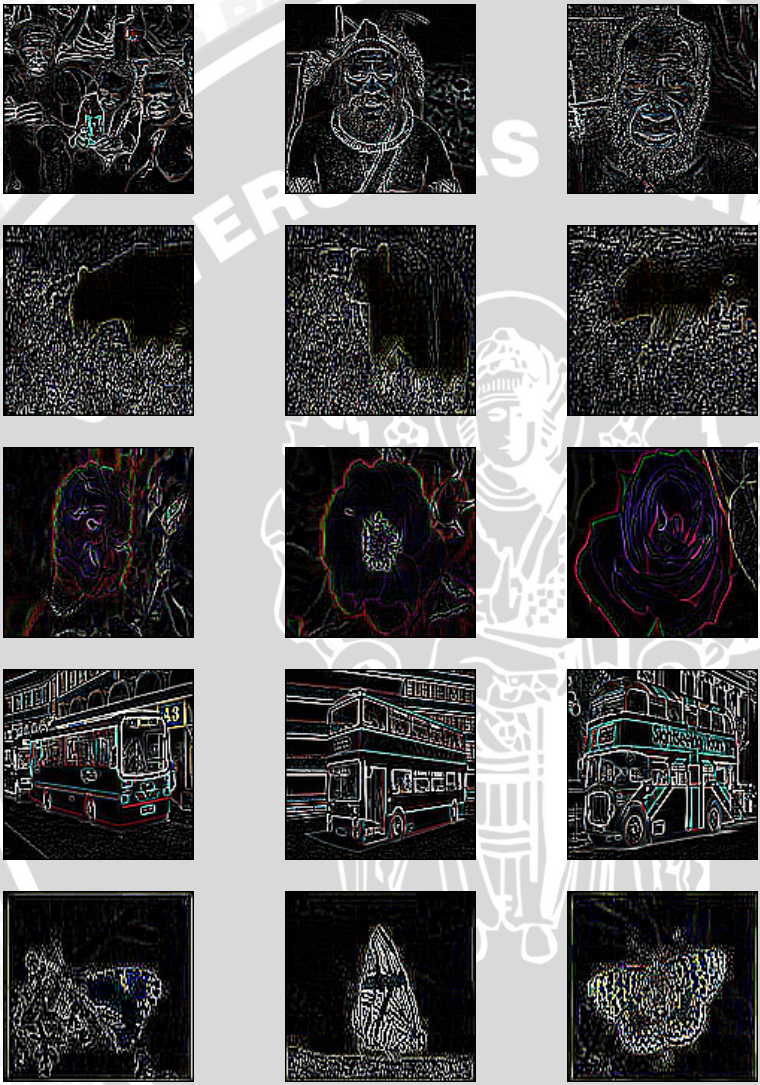





2. Citra *Blur*



3. Citra *Edge*



4. Citra *Noise*