Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KI141502

IMPLEMENTASI TEMU KEMBALI CITRA MENGGUNAKAN FITUR WARNA BERBASIS HISTOGRAM DAN FITUR TEKSTUR BERBASIS BLOK

SANI PUJI RAHAYU
NRP 5113 100 153

Dosen Pembimbing
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Dini Adni Navastara, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR - KI141502

IMPLEMENTASI TEMU KEMBALI CITRA MENGGUNAKAN FITUR WARNA BERBASIS HISTOGRAM DAN FITUR TEKSTUR BERBASIS BLOK

SANI PUJI RAHAYU
5113 100 153

Dosen Pembimbing I
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

Dosen Pembimbing II
Dini Adni Navastara, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – KI141502

IMPLEMENTATION OF IMAGE RETRIEVAL USING COLOR FEATURE BASED ON HISTOGRAM AND TEXTURE FEATURE BASED ON BLOCK

SANI PUJI RAHAYU
5113 100 153

Supervisor I
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.

Supervisor II
Dini Adni Navastara, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
Sepuluh Nopember Institute of Technology
Surabaya, 2017

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

Implementasi Temu Kembali Citra Menggunakan Fitur Warna Berbasis Histogram dan Fitur Tekstur Berbasis Blok

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Cerdas Dan Visi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh :
SANI PUJI RAHAYU
NRP : 5113 100 153

Disetujui oleh Dosen Pembimbing Tugas Akhir :

Dr.Eng. NANIK SUCIATI, S.Kom., M.Kom. ..... (pembimbing 1)
NIP: 197104281994122001

DINI ADNI NAVASTARA, S.Kom., M.Sc. ..... (pembimbing 2)
NIP: 198510172015042001

SURABAYA
MEI 2017

[Halaman ini sengaja dikosongkan]

Implementasi Temu Kembali Citra Menggunakan Fitur Warna Berbasis Histogram dan Fitur Tekstur Berbasis Blok

Nama Mahasiswa : Sani Puji Rahayu
NRP : 5113 100 153
Jurusan : Teknik Informatika FTIf-ITS
Dosen Pembimbing 1 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Dosen Pembimbing 2 : Dini Adni Navastara, S.Kom., M.Sc.

ABSTRAK

Citra digital biasa digunakan masyarakat dalam berbagai bidang seperti kesehatan, perdagangan, dan hiburan. Hal ini menyebabkan meningkatnya citra digital yang dihasilkan setiap harinya. Citra digital yang dihasilkan kemudian disimpan dalam suatu tempat penyimpanan seperti *database*. Banyaknya citra digital yang disimpan dalam *database* menyebabkan sulitnya pengelolaan file-file citra terutama dalam menemukan konten citra yang diinginkan. *Content based image retrieval* (CBIR) merupakan sebuah metode pencarian citra dengan melakukan perbandingan antara citra *query* dengan citra yang ada di *database* berdasarkan informasi yang ada pada citra tersebut.

Pada tugas akhir ini, dibangun suatu sistem temu kembali citra menggunakan fitur warna berbasis histogram dan fitur tekstur berbasis blok. Metode histogram warna digunakan untuk ekstraksi fitur warna dan ekstraksi *Block Difference of Inverse Probabilities* dan *Block Variation of Local Correlation Coefficients* digunakan untuk mengekstraksi fitur tekstur. Metode *Square Chord Distance* digunakan untuk menghitung jarak citra. Hasil pencarian citra mirip dengan rata-rata *precision* terbaik didapatkan dari perpaduan ekstraksi fitur warna dan tekstur warna dengan rata-rata *precision* 93.71% dan rata-rata waktu komputasi 0.2281 detik. Sedangkan untuk perpaduan ekstraksi dengan hasil rata-rata waktu komputasi terbaik adalah menggunakan

perpaduan ekstraksi fitur warna dan tekstur *brightness* dengan rata-rata *precision* 92.22% dan rata-rata waktu komputasi 0.1468 detik.

Kata kunci: Temu Kembali Citra, Color Histogram, Block Difference of Inverse Probabilities, Block Variation Of Local Correlation Coefficients

Implementation of Image Retrieval Using Color Feature Based on Histogram and Texture Feature Based on Block.

Student Name : Sani Puji Rahayu
NRP : 5113 100 153
Major : Teknik Informatika FTIf-ITS
Advisor 1 : Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Advisor 2 : Dini Adni Navastara, S.Kom., M.Sc.

ABSTRACT

Digital imagery is commonly used by people in the fields of health, commerce, and entertainment. Digital images are usually stored in a storage place such as a database. The large number of digital images stored in the database causes the difficulty of managing image files, especially in finding the desired image content. Content based image retrieval (CBIR) is an image search method by performing a comparison between the image of the query and the image in the database based on the information contained in the image.

In this final project, an image retrieval system were built using histogram-based color features and block-based texture features. Color histogram method is used for color feature extraction. Block Difference of Inverse Probabilities and Block Variations of Local Correlation Coefficients are used to extract texture features. Square Chord Distance method is used to calculate the distance of the image. The best precision of image retrieval were obtained from the combination of color extraction and color texture extraction with average precision 93.71% and the average computation time 0.2281 seconds. As for the best average computation time of image retrieval were obtained from combination of color feature and brightness texture with average precision 93.71% and the average computation time 0.1468 seconds.

Keywords: Image Retrieval, Color Histogram, Block Difference of Inverse Probabilities, Block Variation Of Local Correlation Coefficients

KATA PENGANTAR

Puji syukur kepada Allah Yang Maha Esa atas segala karunia dan rahmat-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul :

"*Implementasi Temu Kembali Citra Menggunakan Fitur Warna Berbasis Histogram dan Fitur Tekstur Berbasis Blok*"

Melalui lembar ini, penulis hanya ingin menyampaikan ucapan terima kasih dan penghormatan yang sebesar-besarnya kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Keluarga tercinta yang telah memberikan dukungan, doa, motivasi, dan perhatian yang luar biasa tanpa henti selama penulis mengerjakan Tugas Akhir ini.
3. Ibu Dr.Eng. Nanik Suciati, S.Kom., M.Kom. dan Ibu Dini Adni Navastara, S.Kom., M.Sc. selaku pembimbing I dan II yang telah membimbing dan membantu penulis serta memberikan motivasi dalam menyelesaikan Tugas Akhir ini.
4. Teman-teman TC13 yang telah bersama-sama berjuang di kampus perjuangan ini.
5. Serta semua pihak yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Surabaya, Mei 2017

Sani Puji Rahayu

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dijabarkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, rumusan, batasan permasalahan, tujuan, metodologi pembuatan Tugas Akhir, dan sistematika penulisan.

1.1 Latar Belakang

Seiring dengan perkembangan teknologi dan tren, citra digital menjadi suatu barang yang lumrah dalam masyarakat. Citra digital biasa digunakan masyarakat dalam berbagai bidang seperti pemerintahan, pendidikan, kesehatan, perdagangan, dan hiburan. Kemajuan teknologi telepon genggam berkamera juga membuat masyarakat dapat dengan mudah membuat/menciptakan citra digital. Hal ini menyebabkan meningkatnya citra digital yang dihasilkan setiap harinya. Citra digital yang dihasilkan kemudian disimpan dalam suatu tempat penyimpanan seperti *database*. Banyaknya citra digital yang disimpan dalam *database* menyebabkan sulitnya pengelolaan file-file citra terutama dalam menemukan konten citra yang diinginkan. Oleh karena itu dibutuhkan suatu cara yang cepat dan akurat dalam menemukan citra yang diinginkan.

Dalam menemukan citra dapat menggunakan nama file dari citra yang dicari sebagai *keyword*. Namun cara tersebut seringkali kurang efektif ketika nama file yang digunakan tidak tepat. Selain menggunakan nama file sebagai masukan, pencarian gambar/citra dapat menggunakan isi dari citra atau *content based image retrieval* (CBIR). CBIR merupakan sebuah metode pencarian citra dengan melakukan perbandingan antara citra *query* dengan citra yang ada di *database* berdasarkan informasi yang ada pada citra tersebut (*Query by Example*) [1]. Untuk dapat menemukan citra yang sesuai, citra *query* diekstraksi dengan menggunakan metode tertentu. Ekstraksi citra *query* akan

menghasilkan karakteristik dari citra seperti fitur warna, fitur tekstur ataupun fitur bentuk.

Temu kembali citra berdasarkan isi terbagi ke dalam dua metode secara umum, yaitu metode secara global dan metode secara lokal. Metode global adalah melakukan ekstraksi fitur dari keseluruhan citra yang merepresentasikan keseluruhan karakteristik citra. Secara komputasi, metode global dinilai efisien dan kuat terhadap *noise* citra. Pada umumnya, metode global bersifat tidak fleksibel terhadap ukuran dan orientasi gambar. Biasanya, metode ini menghasilkan hasil fitur dengan dimensi yang rendah sehingga meningkatkan efisiensi temu kembali citra. Namun kekurangan metode global ini tidak dapat mengatasi keadaan citra yang berkaitan dengan perubahan sudut pandang, perubahan gelap terang, dan karakteristik bentuk gambar lokal. Citra dengan keadaan seperti ini dapat diatasi dengan menggunakan metode lokal yang lebih efektif daripada fitur global.

Pada penelitian yang dilakukan oleh Chandan Singh dan Kanwal Preet Kaur [2], telah mengumpulkan informasi tentang berbagai algoritma ekstraksi fitur yang pernah dilakukan. Fitur warna menjadi salah satu fitur visual yang paling dominan dan banyak diimplementasikan dalam pencarian citra. Adapun fitur tekstur dan fitur bentuk yang biasa digunakan dalam temu kembali citra. Fitur tekstur adalah deskriptor berbasis wilayah yang sangat baik untuk temu kembali citra. Kebanyakan citra dapat dibedakan berdasarkan teksturnya, fitur tekstur dapat mengukur karakteristik citra seperti kehalusan dan kekasaran citra serta keteraturan citra. Fitur bentuk juga memberikan informasi karakteristik citra dengan baik. Fitur bentuk direpresentasikan ke dalam dua cara, representasi tepi bentuk pada citra dan representasi wilayah bentuk pada citra. Namun fitur ini memiliki kelemahan ketika mengekstraksi citra berwarna terutama citra natural yang karakteristik citranya kebanyakan tidak berbentuk dan memiliki banyak tekstur. Pada penelitian sebelumnya telah digunakan ekstraksi fitur tekstur dengan metode *local binary*

pattern (LBP) [3]. LBP merupakan metode ekstraksi lokal yang dinilai sangat baik dalam mengekstraksi fitur tekstur. Namun hasil ekstraksi metode LBP memiliki dimensi yang besar sehingga akan mempersulit dalam perbandingan fitur.

Tugas akhir ini, mengimplementasikan sebuah aplikasi temu kembali citra menggunakan fitur warna berbasis histogram dan fitur tekstur berbasis blok untuk menemukan kembali citra natural. Untuk mengekstraksi warna citra, digunakan metode global yaitu histogram warna sedangkan untuk ekstraksi tekstur menggunakan metode lokal yaitu gabungan dari *Block Difference of Inverse Probabilities* (BDIP) *dan Block Variation of Local correlation Coefficients* (BVLC). Metode BDIP dan BVLC dinilai lebih efektif dari metode LBP dalam ekstraksi tekstur saat dipadukan dengan fitur warna [2]. Perpaduan kedua metode BDIP dan BVLC dalam ekstraksi fitur tekstur akan menghasilkan nilai karakteristik kecerahan dan kehalusan citra dengan pendekatan fitur lokal yang dinilai lebih efektif dalam temu kembali citra. Ekstraksi fitur warna menggunakan metode histogram warna yang sederhana sehingga mudah dipadukan dengan kedua metode ekstraksi fitur tekstur. Metode *Square-Chord Distance* digunakan untuk membandingkan citra atau mengukur jarak citra *query* dengan citra dalam *database*. Kombinasi dari metode-metode ini diharapkan dapat menghasilkan temu kembali citra yang akurat dengan waktu komputasi yang optimal.

1.2 Rumusan Permasalahan

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana mengekstraksi fitur warna citra menggunakan histogram warna?
2. Bagaimana mengekstraksi fitur tekstur citra menggunakan metode BDIP dan BVLC?
3. Bagaimana menemukan citra yang sesuai dengan masukan citra dengan *square-chord distance*?

4. Bagaimana mengevaluasi aplikasi temu kembali citra yang telah dibangun?

1.3 Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Citra yang diuji menggunakan citra RGB yang diambil dari data set Corel-5K, Corel-10K
2. Sistem yang dibuat akan menggunakan kakas bantu matlab

1.4 Tujuan Tugas Akhir

Tujuan dari pembuatan tugas akhir adalah membuat sistem temu kembali citra dengan metode histogram warna sebagai ekstraksi warna, BDIP dan BVLC sebagai ekstraksi tekstur.

1.5 Manfaat Tugas Akhir

Manfaat dari tugas akhir ini adalah menghasilkan sistem untuk temu kembali citra natural. Sistem temu kembali citra dapat digunakan dalam pengelolaan file citra dalam *database*, terutama dalam menemukan citra yang serupa.

1.6 Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

1. **Studi Literatur**

   Pada tahap ini dilakukan pencarian informasi dan studi literatur yang diperlukan untuk pengumpulan data dan desain sistem yang akan dibuat. Informasi didapatkan dari buku, artikel, internet, dan materi-materi lain yang berhubungan dengan:

   a. Histogram warna
   b. *Block Difference of Inverse Probabilities* (BDIP)
   c. *Block Variation of Local correlation Coefficients*

(BVLC)
d. *Square-chord distance*
e. *Precision* and *recall*.

2. Implementasi dan Pembuatan Perangkat Lunak

Pada tahap ini dilakukan implementasi proses ekstraksi fitur citra dan perhitungan kemiripan citra. Perincian tahap ini adalah sebagai berikut:

a. Pra-proses citra
b. Ekstraksi fitur warna dengan Histogram warna
c. Ekstraksi fitur tekstur dengan *Block Difference of Inverse Probabilities* (BDIP) dan *Block Variation of Local correlation Coefficients* (BVLC)
d. Menghitung kemiripan citra menggunakan *square-chord distance*
e. Membuat tampilan sistem.

3. Pengujian dan Evaluasi

Pada tahap ini dilakukan uji coba dengan menggunakan citra masukan untuk mencoba jalannya sistem temu kembali citra telah sesuai dengan rancangan dan desain implementasi yang dibuat, juga untuk mencari kesalahan-kesalahan program yang mungkin terjadi. Citra keluaran kemudian akan diteliti dengan menghitung nilai rata-rata *precision*, *recall*, mAP dan lama komputasi untuk mengevaluasi kinerja sistem temu kembali citra.

4. Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang berisi dasar teori, dokumentasi dari perangkat lunak, dan hasil-hasil yang diperoleh selama pengerjaan Tugas Akhir.

1.7 Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab yang berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

Bab II Tinjauan Pustaka

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

Bab III Desain dan Perancangan

Bab ini berisi penjelasan mengenai desain, perancangan dan data yang digunakan untuk memenuhi Tugas Akhir, serta urutan pelaksanaan percobaan.

Bab IV Implementasi

Bab ini berisi implementasi temu kembali citra yang dibangun menggunakan MATLAB sesuai dengan permasalahan dan batasan yang telah dijabarkan pada Bab 1.

Bab V Pengujian dan Evaluasi

Bab ini berisi penjelasan mengenai data hasil percobaan atau pengukuran, dan pembahasan mengenai hasil percobaan yang telah dilakukan.

Bab VI Kesimpulan dan Saran

Bab ini berisi hasil penelitian yang menjawab permasalahan atau yang berupa konsep, program,

dan karya rancangan. Selain itu, pada bab ini diberikan saran-saran yang berisi hal-hal yang masih dapat dikerjakan dengan laebih baik dan dapat dikembangkan lebih lanjut, atau berisi masalah-masalah yang dialami pada proses pengerjaan Tugas Akhir.

Bab VII Daftar Pustaka

Bab ini berisi daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

[Halaman ini sengaja dikosongkan]

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan implementasi perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap sistem yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 Sistem Temu Kembali Citra

Temu kembali citra adalah suatu kegiatan mencari dan menemukan citra dari database gambar digital yang besar menggunakan sistem komputer [4]. Cara temu kembali citra yang paling tradisional adalah dengan menambahkan metadata seperti kata kunci, deskripsi citra ataupun keterangan pada citra sehingga temu kembali citra dapat dilakukan dengan melakukan anotasi citra. Melakukan anotasi citra secara manual memerlukan waktu, tenaga dan biaya yang banyak, untuk itu telah dilakukan sejumlah besar penelitian untuk melakukan anotasi secara otomatis menggunakan komputer.

Ada beberapa cara yang dapat digunakan untuk mencari dan menemukan citra digital yaitu temu kembali citra berdasarkan teks dan temu kembali citra berdasarkan isi. Temu kembali citra berdsarkan teks yaitu mencari citra berdasarkan metadata terkait seperti kata kunci, teks, dll. Sedangkan temu kembali citra berdasarkan isi adalah mencari citra berdasarkan kemiripan isi citra.

2.2 *Content Based Image Retrieval* (CBIR)

Temu kembali citra berdasarkan isi atau *content based image retrieval* adalah temu kembali citra yang bergantung pada penggalian jumlah karakteristik yang sesuai dan menggambarkan isi citra yang diinginkan [5]. Karakteristik suatu citra didapatkan dengan melakukan ekstraksi fitur. Citra akan diindeks sesuai

intensitas citra seperti warna, tekstur dan bentuk. Isi dari citra diubah ke dalam bentuk numerik yang disebut dengan fitur.

Ekstraksi fitur dalam temu kembali citra berdasarkan isi digolongkan ke dalam dua metode secara umum, yaitu metode secara global dan metode secara lokal. Metode global adalah melakukan ekstraksi fitur dari keseluruhan citra yang merepresentasikan keseluruhan karakteristik citra. Biasanya, metode ini menghasilkan hasil fitur dengan dimensi yang rendah sehingga meningkatkan efisiensi temu kembali citra. Namun kekurangan metode global ini adalah tidak dapat mengatasi keadaan citra yang berkaitan dengan perubahan sudut pandang, perubahan gelap terang, dan karakteristik bentuk gambar lokal. Citra dengan keadaan seperti ini dapat ditangani dengan menggunakan metode secara lokal yang lebih efektif daripada metode secara global.

Ada berbagai macam algoritma ekstraksi fitur yang dapat digunakan untuk temu kembali citra. Fitur warna menjadi salah satu fitur visual yang paling dominan dan banyak diimplementasikan dalam pencarian citra. Histogram warna adalah representasi fitur warna yang paling umum digunakan yang menggambarkan distribusi warna global dari sebuah gambar. Histogram warna ini menentukan probabilitas gabungan intensitas saluran warna. Fitur tekstur adalah deskriptor berbasis wilayah yang sangat baik untuk temu kembali citra. Kebanyakan citra dapat dibedakan berdasarkan teksturnya, fitur tekstur dapat mengukur karakteristik citra seperti kehalusan dan kekasaran citra dan keteraturan. Fitur bentuk juga memberikan informasi karakteristik citra dengan baik. Fitur bentuk direpresentasikan ke dalam dua cara, representasi tepi bentuk pada citra dan representasi wilayah bentuk pada citra. Namun fitur ini memiliki kelemahan ketika mengekstraksi citra berwarna terutama citra natural yang karakteristik citranya kebanyakan tidak berbentuk dan memiliki banyak tekstur.

Temu kembali citra berdasarkan isi dapat menggunakan beberapa kombinasi ekstraksi fitur citra, semakin banyak fitur

yang digunakan memungkinkan semakin baik hasil citra yang ditemukan. Namun, semakin banyak fitur yang digunakan menyebabkan semakin lama waktu proses yang diperlukan [2]. Untuk menemukan kembali citra natural, kombinasi fitur warna dan fitur tekstur dinilai lebih efektif karena karakteristik citra natural yang memiliki banyak tekstur dan kebanyakan tidak berbentuk. Tugas akhir ini akan mengimplementasikan temu kembali citra natural menggunakan ekstraksi fitur warna dan tekstur. Ekstraksi fitur bentuk tidak digunakan karena dinilai kurang efektif dalam memproses citra natural dan untuk menghindari waktu proses yang lama.

2.3 Pengolahan Citra Digital

Pengolahan citra digital adalah suatu proses yang bertujuan untuk memanipulasi dan menganalisis citra dengan bantuan komputer. Ada dua jenis kegiatan pengolahan citra digital yaitu mengolah kualitas citra dan mengolah informasi yang terdapat dalam citra. Mengolah informasi citra berhubungan dengan *pattern recognition* atau pengenalan pola yang bertujuan untuk mengenali suatu objek dengan cara mengekstraksi informasi penting yang terdapat pada suatu citra. Pengolahan citra dan pengenalan pola menjadi bagian dari proses pengenalan citra. Kedua aplikasi ini akan saling melengkapi untuk mendapatkan ciri khas dari suatu citra yang hendak dikenali.

Pengolahan citra digital secara khusus menjadi satu-satunya teknologi yang digunakan dalam klasifikasi, pengenalan pola, proyeksi, analisis sinyal multi skala, dan ekstraksi fitur. Pada temu kembali citra pengolahan citra digital diperlukan untuk mengekstraksi fitur-fitur karakteristik pada citra.

2.4 Ruang Warna

Ruang warna adalah suatu organisasi dari warna tertentu. Berbagai warna dapat diciptakan oleh pigmen warna primer dan warna tersebutlah yang akan menentukan ruang warna tertentu. Ruang warna atau dikenal sebagai model warna adalah model

matematika abstrak yeng menggambarkan bagaimana warna dapat diwakili sebagai bilangan tupel [6]. Pada dasarnya, ruang warna merupakan penjabaran dari sistem koordinat dan sub-ruang. Setiap warna dalam sistem diwakili oleh satu titik. Ruang warna dapat hal ini memungkinkan untuk membuat suatu ruang warna secara bebas

Ruang warna dapat untuk dibuat secara bebas dengan aturan yang bebas pula. Namun ruang warna yang dibuat secara bebas akan sulit untuk dipahami oleh orang lain secara global. Maka dibutuhkan ruang warna yang disepakati oleh orang-orang secara global. Ada berbagai macam jenis ruang warna yang digunakan secara global, seperti ruang warna RGB, HSV, CMYK, L*a*b dan lain-lain. Diantara semua ruang warna, ruang warna RGB adalah ruang warna yang paling umum digunakan oleh orang-orang. Pada Tugas Akhir ini akan menggunakan RGB sebagai ruang warna dalam pemetaan warna dan proses-proses lainnya.

2.5 *Red, Green, Blue* (RGB)

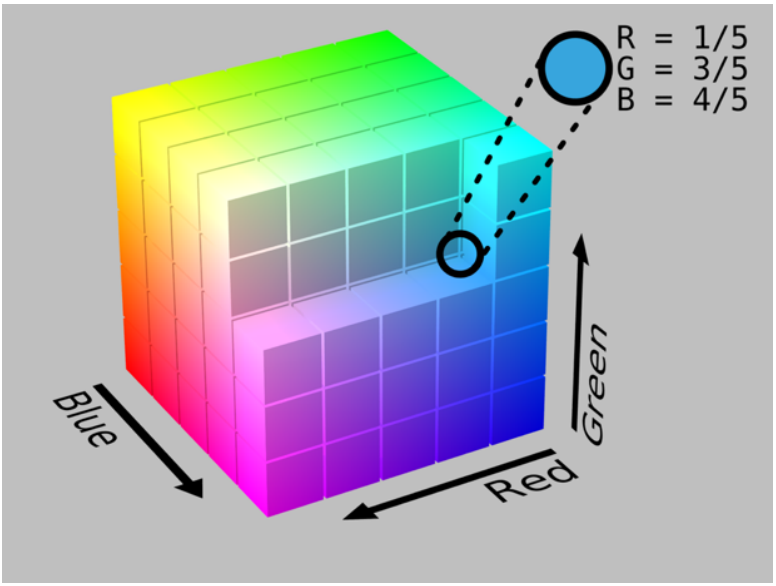
RGB kependekan dari *Red, Green*, dan *Blue*, adalah suatu ruang warna aditif yang menggunakan perpaduan cahaya merah, hijau dan biru dengan berbagai kombinasi untuk memproduksi bermacam-macam warna [7]. Tujuan utama ruang warna RGB adalah untuk penginderaan, representasi, dan tampilan gambar dalam sistem elektronik seperti komputer, telepon genggam, dan televisi.

Model warna RGB adalah model warna yang bergantung pada perangkat, setiap perangkat elektronik mendeteksi atau memproduksi nilai RGB secara berbeda, hal ini dikarenakan elemen warna seperti fosfor yang digunakan memiliki respons warna yang berbeda terhadap tingkat R, G, dan B. Sehingga nilai RGB pada perangkat satu dengan yang lain tidak dapat menentukan kesamaan warna.

Suatu warna pada ruang warna RGB digambarkan dengan menunjukkan berapa banyak masing-masing warna merah, hijau,

dan biru yang dikombinasikan. Masing-masing komponen warna merah, hijau, dan biru memiliki nilai intensitas antara 0 sampai 255. Ketika kombinasi ketiga warna adalah nilai terendah yaitu 0 semua, maka representasi warna akan menjadi hitam. Sebaliknya ketika kombinasi ketiga warna adalah nilai tertinggi yaitu 255 semua, maka representasi warna suatu citra akan menjadi putih. Sedangkan ketika kombinasi ketiga warna adalah nilai yang sama, maka representasi warna yang dihasilkan adalah abu-abu.

Pada citra digital, suatu citra berwarna dapat di bedakan menjadi masing-masing komponen ruang warna RGB. Hal ini biasa dilakukan untuk mengambil nilai intensitas pada masing-masing komponen ruang warna RGB. Kombinasi warna RGB direpresentasikan seperti kubus yang dapat dilihat pada Gambar 2.1.



Gambar 2.1 Kubus RGB
(Sumber: Wikipedia ”RGB Color Model” [7])

Tabel 2.1 Kuantisasi histogram warna

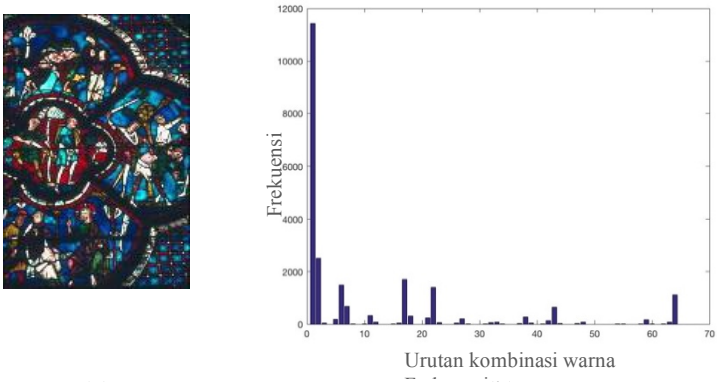
Kuantisasi	Intensitas
1	0-63
2	64-127
3	128-191
4	192-225

2.6 Histogram Warna

Citra terdiri dari piksel-piksel warna. Dalam pengolahan citra, intensitas citra dapat direpresentasikan ke dalam beberapa jenis ruang warna seperti RGB, HSV, L*a*b, dsb. Tugas akhir ini menggunakan pembagian warna RGB dalam pemrosesan ekstraksi fitur. RGB adalah pembagian warna citra ke dalam 3 ruang warna yaitu merah, hijau, dan biru. Setiap ruang warna RGB memiliki 256 nilai intensitas dengan rentang nilai dari 0-255.

Salah satu cara untuk mendapatkan karakteristik warna pada suatu citra adalah dengan menggunakan histogram warna. Histogram warna didapatkan dengan merepresentasikan distribusi warna dalam sebuah citra yaitu kemungkinan terjadinya warna dalam rentang warna yang telah ditentukan dalam citra [8]. Namun, banyaknya nilai intensitas warna pada RGB dapat mempersulit perhitungan perbedaan citra. Oleh karena itu, dilakukan kuantisasi warna dengan mengelompokkan intensitas warna pada masing-masing ruang warna menjadi 4 kelompok atau 4 bins dengan rentang intensitas 0-63, 64-127, 128-191, 192-255. Intensitas pada citra akan dirubah berdasarkan pembagian kelompok tersebut, angka 1 mewakilkan range intensitas 0-63, angka 2 mewakilkan *range* intensitas 64-127, angka 3 mewakilkan *range* intensitas 128-191 dan angka 4 mewakilkan *range* intensitas 192-255. Tabel 2.1 menunjukkan aturan pembagian intensitas untuk kuantisasi.

Histogram warna dibangun dengan menggunakan array 3 dimensi, masing-masing dimensi mewakilkan ruang warna merah, hijau, dan biru dengan indeks 1-4 yang merepresentasikan hasil



(a) (b)

Gambar 2.2 (a) Citra asli (b) Histogram warna

kuantisasi intensitas yaitu pengelompokkan intensitas menjadi 4 kelompok. Histogram warna berisi jumlah kombinasi dari ruang warna dengan indeks sesuai pengelompokkan intensitas dan menghasilkan 64 nilai histogram. Hasil histogram warna dapat dilihat pada Gambar 2.2. Pada gambar terlihat citra asli pada dan diagram batang yang menunjukkan histogram warna dari citra asil. Pada diagram batang terlihat frekuensi dari kombinasi nilai masing-masing RGB. Urutan kombinasi warna yang terdapat pada histogram dijelaskan pada Tabel 2.2. Kolom No merupakan nomor urutan pada tampilan histogram, sedangkan *Red, Green, Blue* merupakan nilai intensitas masing-masing komponen warna setelah dilakukan kuantisasi.

2.7 *Difference of Inverse Probabilities* (DIP)

Difference of inverse probabilities (DIP) atau perbedaan dari probabilitas invers adalah suatu operator yang digunakan untuk mengekstrak fitur sketsa yang terdiri dari lembah dan tepi yang berdasarkan pada intensitas lokal [9]. Pada DIP, rasio

Tabel 2.2 Konversi kombinasi nilai RGB

No	*Red*	*Green*	*Blue*
1	1	1	1
2	1	1	2
3	1	1	3
4	1	1	4
5	1	2	1
6	1	2	2
7	1	2	3
8	1	2	4
9	1	3	1
10	1	3	2
11	1	3	3
12	1	3	4
13	1	4	1
14	1	4	2
15	1	4	3
16	1	4	4
17	2	1	1
18	2	1	2
19	2	1	3
20	2	1	4
21	2	2	1
22	2	2	2
23	2	2	3
24	2	2	4
25	2	3	1
26	2	3	2
27	2	3	3
28	2	3	4
29	2	4	1
30	2	4	2
31	2	4	3
32	2	4	4

No	*Red*	*Green*	*Blue*
33	3	1	1
34	3	1	2
35	3	1	3
36	3	1	4
37	3	2	1
38	3	2	2
39	3	2	3
40	3	2	4
41	3	3	1
42	3	3	2
43	3	3	3
44	3	3	4
45	3	4	1
46	3	4	2
47	3	4	3
48	3	4	4
49	4	1	1
50	4	1	2
51	4	1	3
52	4	1	4
53	4	2	1
54	4	2	2
55	4	2	3
56	4	2	4
57	4	3	1
58	4	3	2
59	4	3	3
60	4	3	4
61	4	4	1
62	4	4	2
63	4	4	3
64	4	4	4

intensitas piksel pada jendela gambar terhadap jumlah dari semua intensitas piksel pada jendela dianggap sebagai probabilitas. Cara kerja DIP adalah menghitung perbedaan antara probabilitas invers dari piksel yang berada di tengah jendela dan piksel dengan intensitas maksimum pada jendela. DIP mengukur perubahan nilai intensitas yang drastis untuk mendeteksi tepi pada suatu daerah citra. Sedangkan deteksi lembah merupakan daerah yang terdiri dari intensitas lokal minimum [10]. Rumus dari *difference probability* (DP) dapat dilihat pada persamaan (2.1) berikut:

$$DP(x,y) = \frac{I_m(x,y)}{\overline{I(x,y)}} - \frac{I(x,y)}{\overline{I(x,y)}}, \quad (2.1)$$

sedangkan rumus DIP dapat dilihat pada persamaan (2.2) berikut:

$$DIP(x,y) = DP \frac{\overline{I(x,y)}\,\overline{I(x,y)}}{I_m(x,y)\,I(x,y)}, \quad (2.2)$$

dimana $I(x,y)$ adalah intensitas piksel *(x,y),* $\overline{I(x,y)}$ adalah jumlah intensitas dan $I_m(x,y)$ adalah nilai intensitas maksimum pada jendela.

2.8 *Block Difference of Inverse Probabilities* (BDIP)

Block difference of inverse probabilities (BDIP) atau blok perbedaan dari probabilitas invers merupakan metode ekstraksi fitur tekstur yang secara efisien mengukur variasi kecerahan citra. BDIP bekerja dengan cara menghitung perbedaan antara jumlah piksel dalam satu blok dan rasio jumlah intensitas piksel di blok sampai maksimum di blok. Sebelum melakukan perhitungan, akan diambil masing-masing komponen ruang warna RGB yaitu merah, hijau, dan biru pada citra *query*. Kemudian citra akan diproses dengan membagi citra kedalam blok-blok. Dalam proses perhitungan BDIP, tiap blok berukuran 2×2 piksel tidak tumpang tindih dari citra *I(x,y)* akan dicari nilai BDIP dengan rumus seperti pada persamaan (2.3).

Setiap blok-blok piksel akan menghasilkan 1 nilai, sehingga didapatkan matriks atau array 2 dimensi berukuran

seperempat dari ukuran citra asli. Hasil akhir ekstraksi BDIP ini merepresentasikan tepi dan batas dari wilayah citra.

Rumus BDIP dapat dilihat pada persamaan (2.3) berikut:

$$BDIP(x,y) = B \times B - \frac{\Sigma_{i=0}^{1}\Sigma_{j=0}^{1}\ I(2x+i,2y+j)}{I_{max}(x,y)}\ , \qquad (2.3)$$

$$x = 0,1,\ \ldots,\frac{M}{2}-1\ ,\quad y = 0,1,\ \ldots,\frac{N}{2}-1$$

dimana $I_{max}(x,y)$ adalah nilai intensitas terbesar pada blok, M adalah jumlah baris citra, N adalah jumlah kolom citra, dan B adalah ukuran blok yaitu 2.

2.9 Standar Deviasi

Standar deviasi adalah suatu pengukuran penyebaran dari suatu set data dari rata-ratanya [11]. Standar deviasi dihitung sebagai akar kuadrat dari varians dengan menentukan variasi antara setiap titik data relatif terhadap rata-rata. Jika titik data lebih jauh dari rata-rata, menunjukkan ada penyimpangan yang lebih tinggi dalam kumpulan data. Standar deviasi dihitung berdasarkan rata-rata. Jarak setiap titik data dari rata-rata dikuadrat, kemudian dijumlah dan dirata-ratakan untuk mendapatkan nilai variansnya. Dengan kata lain, Varians diturunkan dengan mengambil rata-rata titik data, mengurangi rata-rata dari setiap titik data satu per satu, dan mengkuadratkan masing-masing hasil ini dan kemudian mengambil titik lain dari kuadrat ini. Singkatnya, varians adalah rata-rata perbedaan kuadrat dari *mean* dan standar deviasi adalah akar kuadrat dari varians. dari Rumus standar deviasi dapat dilihat pada persamaan (2.4) berikut:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2}\ , \qquad (2.4)$$

dimana μ adalah rata-rata dari x, n merupakan jumlah sampel, dan x_i adalah nilai x ke-i.

2.10 *Covariance*

Covariance atau kovariansi adalah suatu pengukuran kekuatan korelasi antara dua atau lebih rangkaian variasi acak [12]. Kovariansi positif berarti bahwa kedua variabel bergerak bersama, sedangkan kovariansi negatif berarti hasil kedua variabel bergerak berlawanan. Kovariansi dihitung dengan menganalisa hasil standar deviasi atau dengan mengalikan korelasi antara kedua variabel dengan standar deviasi masing-masing variabel. Rumus kovariansi dapat dilihat pada persamaan (2.5) berikut:

$$Cov(x,y) = \sum_{i=1}^{N} \frac{(x_i-\overline{x})(y_i-\overline{y})}{N}, \qquad (2.5)$$

dimana x dan y adalah sekumpulan data masing-masing berjumlah *N*, x_i, dan y_i adalah nilai dua variabel yang akan dihitung, N adalah jumlah variabel, $\overline{x}$ adalah rata-rata nilai variabel x, $\overline{y}$ adalah nilai rata-rata variabel y.

2.11 *Correlation Coefficients*

Correlation coefficients atau koefisien korelasi adalah suatu pengukuran yang menentukan sejauh mana dua gerakan variabel berkaitan [13]. Hasil nilai dari koefisien korelasi adalah antara -1.0 sampai 1.0 sehingga jika hasil dari koefisien korelasi lebih besar dari 1.0 atau lebih kecil dari -1.0 dipastikan ada kesalahan dalam perhitungan. Nilai hasil koefisien korelasi sama dengan -1.0 menunjukkan bahwa kedua variabel bergerak ke arah yang berlawanan, nilai koefisien korelasi sama dengan 1.0 menunjukkan kedua variabel memiliki korelasi yang sempurna, sedangkan nilai koefisien korelasi yang bernilai 0 menunjukkan bahwa kedua variabel tidak memiliki korelasi sama sekali.

Nilai koefisien korelasi didapat dengan menghitung nilai *covariance* (persamaan 2.5) dari kedua variabel yang akan diuji dan membaginya dengan hasil kali nilai standar deviasi

(persamaan 2.4) dari kedua variabel yang akan diuji. Rumus koefisien korelasi dapat dilihat pada persamaan (2.6) berikut:

$$\rho_{(x,y)} = \frac{Cov(x,y)}{\sigma_x \sigma_y}, \quad (2.6)$$

dimana σ_x merupakan standar deviasi dari x dan σ_y merupakan standar deviasi dari *y*.

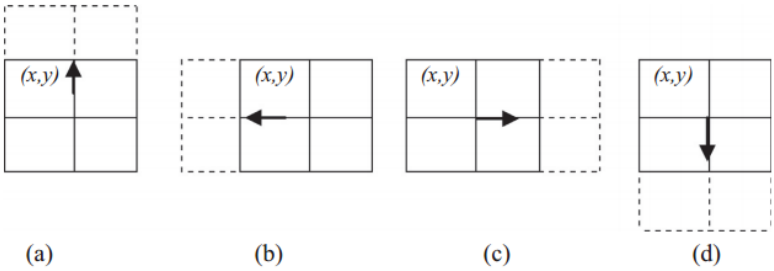
2.12 *Block Variation of Local Correlation Coefficients* (BVLC)

Block variation of local correlation coefficients atau variasi blok dari korelasi koefisien local adalah suatu fitur tekstur untuk mengukur variasi tekstur kehalusan lokal. BVLC bekerja dengan menghitung variasi atau perbedaan antara maksimum dan minimum dari korelasi koefisien local berdasarkan empat arah yaitu kanan, kiri, atas, dan bawah.

Sebelum melakukan perhitungan, akan diambil masing-masing komponen ruang warna RGB yaitu merah, hijau, dan biru pada citra *query*. Kemudian citra akan diproses dengan membagi citra kedalam blok-blok. BVLC merupakan metode versi blok dari VLCC (*variation of local correlation coefficients*) [14], yaitu setiap korelasi koefisien lokal didefinisikan sebagai lokal kovarians yang dinormalisasi dari lokal varians. Dalam proses perhitungan BVLC, tiap blok berukuran 2×2 piksel tidak tumpang tindih dari citra *I(x,y)* akan dicari nilai VLCC dengan rumus seperti pada persamaan (2.4) berikut:

$$\rho(p,q) = \frac{1}{\sigma(x,y)\sigma(p,q)} \times \left[\frac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1}[I(2x+i,2y+j)I(p,q) - \mu(x,y)\mu(p,q)]\right], \quad (2.4)$$

dimana $(p,q) \in \{(-1,0), (0,-1), (0,1), (1,0)\}$. Nilai-nilai (p,q) menunjukkan pergeseran horizontal dan vertikal yang terkait dengan empat orientasi seperti pada Gambar 2.3.



Gambar 2.3 Konfigurasi piksel pada 2×2 wilayah jendela dan pergeseran jendela pada setiap 4 arah, (a)(-1,0), (b)(0,-1), (c)(0,1), (d)(1,0)

Nilai-nilai *μ(x,y)* dan *σ(x,y)* adalah rata-rata dan standar deviasi dalam 2×2 wilayah lokal. *I(p,q), μ(p,q),* dan *σ(p,q)* menunjukkan intensitas, rata-rata dan standar deviasi dari intensitas ketika jendela lokal bergeser oleh *(p,q)* sepanjang empat arah. Rumus BVLC didefinisikan pada persamaan (2.5) berikut:

$$BVLC(x,y) = max_{(p,q)}[\rho(p,q)] - min_{(p,q)}[\rho(p,q)], \quad (2.5)$$

dimana $max_{(p,q)}$ adalah nilai VLCC tertinggi diantara semua nilai pergeseran VLCC dan $min_{(p,q)}$ adalah nilai VLCC terendah diantara semua nilai pergeseran VLCC.

Sama halnya seperti BDIP, BVLC menghasilkan matriks berukuran seperempat kali citra asli. Hasil akhir BVLC merepresentasikan tekstur dari keseluruhan citra termasuk wilayah dimana variasi intensitasnya rendah.

2.13 *Square Chord Distance*

Dalam Tugas Akhir ini *Square-chord distance* digunakan untuk mengukur kemiripan citra dengan mengukur jarak antara citra *query* dengan citra dalam database. Metode ini dinilai menjadi metode perhitungan jarak terbaik dalam kasus perhitungan fitur menggunakan metode histogram warna, BDIP,

dan BVLC [2]. Rumus *Square-chord distance* didefinisikan pada persamaan (2.6) berikut:

$$d(Q,T) = \sum_{i=0}^{L-1} (\sqrt{x_i(Q)} - \sqrt{y_i(T)})^2, \quad (2.6)$$

dimana $X_i(Q)$ dan $Y_i(T)$ mewakili masing-masing komponen ke-i dari fitur *query* vektor *x(Q)* dan target fitur vektor *y(T)*, sedangkan *L* merupakan jumlah data yang akan dihitung.

2.14 *Precision* dan *Recall*

Presision dan r*ecall* adalah pengukuran dasar yang digunakan dalam mengevaluasi strategi pencarian. *Precision* adalah persentase rasio jumlah *record* relevan yang ditemukan terhadap jumlah *record* yang tidak relevan dan relevan. Precision digunakan untuk mengetahui tingkat ketepatan antara citra masukan dan citra hasil keuaran. Sedangkan *recall* adalah persentase rasio jumlah *record* relevan yang ditemukan terhadap jumlah total *record* yang relevan dalam *Database* [15]. Perhitungan recall digunakan untuk mengetahui tingkat keberhasilan sistem dalam menemukan kembali citra. Variabel yang digunakan pada rumus *precision* dan *recall* dijelaskan pada Tabel 2.3. TP(*True Positives)* yaitu citra yang relevan dan ditemukan kembali. FP(*False Positives*) adalah citra yang tidak relevan namun ditemukan kembali. FN(*False Negatives*) merupakan citra yang tidak relevan namun ditemukan kembali. TN(True Negatives) adalah citra yang tidak relevan dan tidak ditemukan kembali. Rumus perhitungan *precision* dijelaskan pada persamaan (2.7) berikut:

$$\text{P(N)} = \frac{TP}{TP+FP} \times 100\%, \quad (2.7)$$

sedangkan perhitungan *recall* dijelaskan pada persamaan (2.8) berikut:

$$\mathrm{R(N)} = \frac{TP}{TP+FN} \times 100\%, \qquad (2.8)$$

dimana N_R adalah batas atas jumlah citra yang ditemukan, $P(N)$ adalah nilai *precision* dengan jumlah citra yang ditemukan *N*.

Dalam mengukur tingkat efektifitas temu kembali citra digunakan perhitungan rata-rata *precision*. Rata-rata *precision* dan *recall* dari suatu citra *query* adalah *mean* dari semua nilai *precision* maupun *recall* untuk setiap citra ditemukan sejumlah N_R teratas. Rumus dari rata-rata *precision* suatu citra *query* dapat dilihat pada persamaan (2.9) berikut:

$$\overline{P}(q) = \frac{1}{N_R}\sum_{N=1}^{N_R} P(N), \qquad (2.9)$$

sedangkan rumus dari rata-rata *recall* suatu citra *query* dapat dilihat pada persamaan (2.10) berikut:

$$\overline{R}(q) = \frac{1}{N_R}\sum_{N=1}^{N_R} R(N), \qquad (2.10)$$

dimana N_R adalah batas atas jumlah citra yang ditemukan, $R(N)$ adalah nilai *recall* dengan jumlah citra yang ditemukan N.

Pada persamaan (2.8) hasil maksimum perhitungan *recall* bergantung pada *N* yaitu jumlah citra yang ditemukan. Hal ini

Tabel 2.3 *Confusion matrix* perhitungan *precision* dan *recall*

	Relevant	Not Relevant
Retrieved	TP	FP
Not Retrieved	FN	TN

berpengaruh pada nilai rata-rata *recall* dengan nilai *N* yang berubah-ubah. Sehingga nilai maksimum rata-rata *recall* bergantung pada N_R yang digunakan. Sebagai contoh ketika N=20 dan semua keluaran citra relevan sehingga *TP*=20 dan *FN*=0 maka nilai *recall* maksimum adalah 100%, namun ketika *N*=10 dan semua keluaran citra relevan sehingga *TP*=10 dan *FN*=10 maka nilai *recall* maksimum adalah 50%. Oleh karena itu nilai maksimum untuk rata-rata *recall* bergantung pada nilai N_R. Nilai maksimum rata-rata *recall* dapat dihitung dengan rumus pada persamaan (2.11) berikut:

$$\text{maks}\overline{R}=\left(\frac{1}{N_R}\sum_{N=1}^{N_R}\frac{N}{N_R}\right)\times 100\%, \qquad (2.11)$$

dimana N_R adalah batas atas jumlah citra yang ditemukan.

Untuk mendapatkan rata-rata *precision* pada semua *query*, dilakukan perhitungan *mean* dari jumlah rata-rata *precision* semua *query* (*mean average precision* atau *mAP*). Rumus rata-rata *precision* secara keseluruhan dapat dilihat pada persamaan (2.12), berikut:

$$mAP = \frac{1}{Q}\sum_{q=1}^{Q}\overline{P}\,(q), \qquad (2.12)$$

dimana *Q* adalah jumlah banyaknya citra *query.*

Sedangkan untuk mendapatkan rata-rata *recall* pada setiap *query* dilakukan perhitungan *mean average recall* (*mAR*) yang perhitungannya mirip dengan *mAP*, rumus lengkapnya dapat dilihat pada persamaan (2.12) berikut:

$$mAR = \frac{1}{Q}\sum_{q=1}^{Q}\overline{R}\,(q), \qquad (2.13)$$

dimana *Q* adalah jumlah banyaknya citra *query* .

BAB III
DESAIN DAN PERANCANGAN

Pada bab ini diuraikan mengenai desain dan perancangan aplikasi agar dapat mencapai tujuan dari Tugas Akhir ini. Aplikasi yang dibuat pada Tugas Akhir ini berguna untuk melakukan temu kembali citra menggunakan kombinasi fitur warna dan tekstur. Proses perancangan aplikasi temu kembali citra ini terdiri dari perancangan dataset citra, perancangan proses dan diagram alir proses-proses yang dilakukan pada sistem.

3.1 Lingkungan Desain dan Implementasi

Subbab ini akan menjelaskan mengenai lingkungan desain dan implementasi perangkat lunak yang akan dibangun. Spesifikasi perangkat keras yang digunakan dalam desain dan implementasi perangkat lunak dijelaskan pada Tabel 3.1.

Tabel 3.1 Spesifikasi perangkat keras

Perangkat	Spesifikasi
Perangkat Keras	Prosesor: 2.7 GHz Intel Core i5 Memori : 8 GB
Perangkat Lunak	Sistem Operasi : macOS Sierra (10.12.1) Perangkat Pengembang : Ms. Word 2016 Matlab R2016b (9.1.0)

3.2 Perancangan Data

Pada subbab ini akan dijelaskan mengenai perancangan data yang dibutuhkan untuk membangun aplikasi Temu Kembali Citra menggunakan fitur warna berbasis histogram dan fitur tekstur berbasis blok. Perancangan data terdiri dari data masukan, data proses, dan data keluaran. Data masukan menjelaskan data yang diperlukan untuk menjalankan aplikasi. Data proses menjelaskan data-data yang dibutuhkan dan dihasilkan pada

proses-proses eksekusi aplikasi. Data keluaran menjelaskan hasil akhir data atau data citra yang mirip dengan data masukan.

3.2.1 Data Masukan

Data masukan adalah data awal yang dibutuhkan untuk menjalankan proses aplikasi. Data masukan terdiri dari satu data citra natural, pilihan angka antara 10, 12, dan 20 yang menyatakan berapa banyak citra mirip yang ingin ditampilkan dan dua pilihan ekstraksi tekstur yaitu ekstraksi fitur warna atau ekstraksi fitur tekstur *brightness*. Data citra natural dapat dipilih dari 500 citra natural yang tersedia (citra *query*). Lima ratus citra natural ini didapat dari dataset Corel-10k [16] yang berjumlah 10.000 citra dan telah dikelompokkan menjadi 100 kategori, seperti kembang api, bunga, mobil, kuda, bangunan, ikan, makanan, dll. Kemudian pada setiap kategori dipilih 5 citra secara acak untuk mendapatkan citra *query*.

(a)

(b)

(c)

Gambar 3.1 Contoh data citra masukan (a) Citra kategori harimau (b) Citra kategori gunung es (c) Citra kategori bus

Citra masukan kemudian akan diekstrak dan dibandingkan dengan hasil ekstrak 2.000 citra natural Corel-10k yang ada dalam database (citra *training*). Citra dalam database dipilih secara acak sebanyak 20 citra pada setiap kategori citra dataset Corel-10k, citra dalam database ini berbeda dengan citra yang digunakan untuk data citra masukan. Total citra masukan dan citra *training* berjumlah 2.500 citra dengan ukuran 192×128 piksel atau 128×192 piksel. Contoh data citra masukan dapat dilihat pada Gambar 3.1.

3.2.2 Data Proses

Data proses adalah data-data yang dihasilkan oleh suatu proses dan data yang digunakan sebagai masukan suatu proses selanjutnya. Dalam aplikasi ini, terdapat data proses yang berdasarkan subproses/fungsi yang ada. Pembagian subproses yang ada dapat dilihat pada Tabel 3.2. Pada setiap subproses akan menghasilkan data variabel yang menjadi masukan untuk proses berikutnya, data variabel tersebut dapat dilihat pada Tabel 3.3.

3.2.3 Data Keluaran

Data Keluaran adalah data akhir yang dihasilkan setelah melalui proses-proses pada aplikasi. Pada sistem temu kembali citra ini, data yang dihasilkan adalah citra-citra yang mirip dengan citra masukan dan ditampilkan pada panel hasil disebelah kanan (Gambar 3.2). Jumlah citra keluaran bergantung pada masukan angka yang dipilih antara 10, 12, dan 20. Pada data keluaran juga ditampilkan masing-masing nama file citra, kategori citra, dan jarak citra tersebut terhadap citra masukan yang ditampilkan pada atas dan bawah citra yang mirip. Terdapat juga kolom yang berisikan nilai *precision* dan *recall* serta waktu komputasi proses temu kembali citra pada kolom kiri bawah. Gambar tampilan hasil keluaran citra dapat dilihat pada Gambar 3.2.

3.3 Perancangan Proses

Perancangan proses dilakukan untuk memberikan gambaran alur algoritma yang digunakan pada tahap

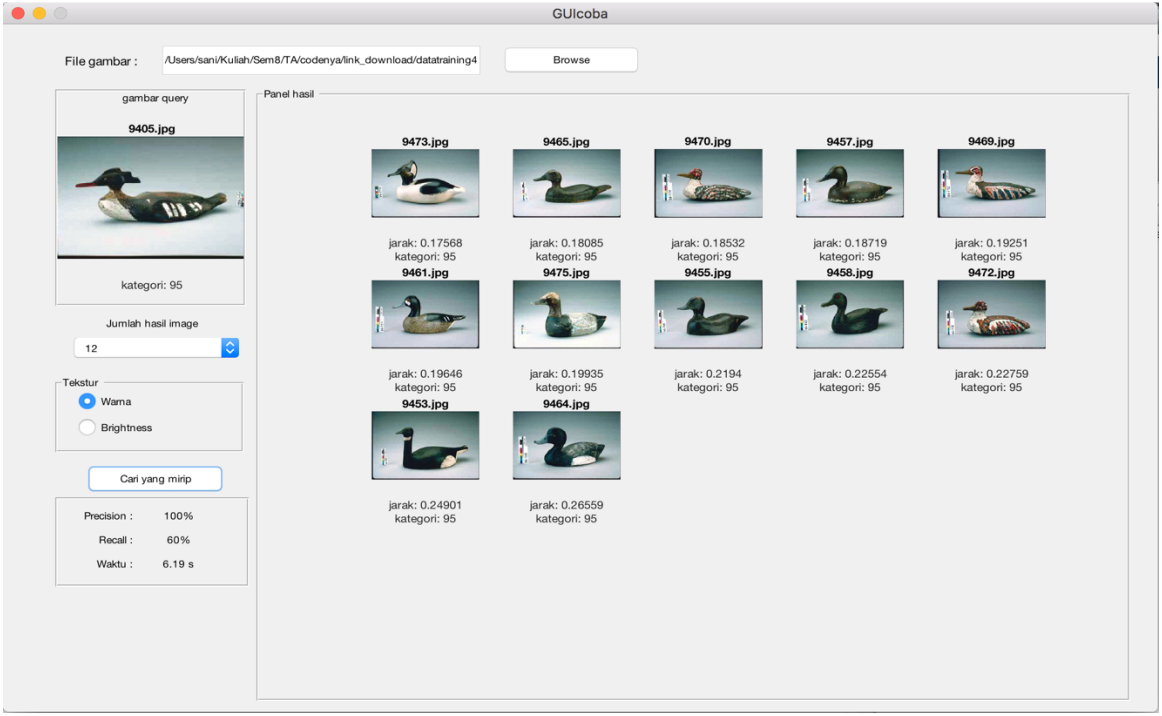
Tabel 3.2 Nama fungsi

No	Nama Fungsi	Keterangan
1	colorhistogram	Fungsi ekstraksi fitur warna
2	hitungBDIP	Fungsi ekstraksi fitur tekstur kecerahan BDIP
3	hitungBVLC	Fungsi ekstraksi fitur tekstur kecerahan BVLC
4	nilaipershift1	Fungsi pergeseran blok atas
5	nilaipershift2	Fungsi pergeseran blok kiri
6	nilaipershift3	Fungsi pergeseran blok kanan
7	nilaipershift4	Fungsi pergeseran blok bawah
8	BDIP_BVLC_brightness	Fungsi untuk mengekstrak fitur tekstur *brightness*
9	f_dividearray	Fungsi untuk mengklasifikasi fitur tekstur
10	Squarechord	Fungsi menghitung jarak citra
11	Disttekstur	Fungsi menghitung jarak tekstur citra
12	Precisionrecall	Fungsi untuk menghitung *precision* dan *recall*

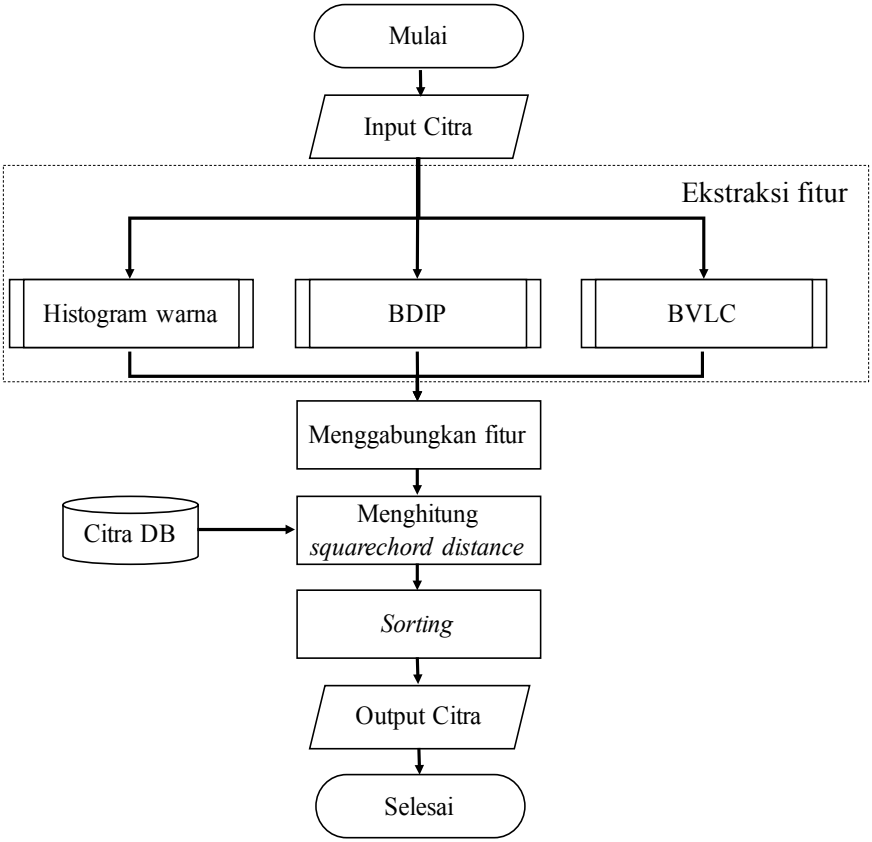
implementasi. Alur algoritma ini akan ditampilkan dalam diagram alir dari masing-masing proses. Sistem temu kembali citra ini digunakan dengan memasukkan citra natural yang telah dikategorikan sebagai *query*. Citra ini kemudian akan melalui serangkaian proses sehingga mendapatkan citra keluaran yang mirip. Terdapat tiga proses utama pada sistem temu kembali citra ini. Proses pertama adalah ekstraksi fitur. Proses kedua adalah menghitung jarak kemiripan citra dan proses terakhir adalah pengembalian citra yang mirip. Tahapan proses secara umum dijelaskan lebih lanjut dengan diagram alir pada Gambar 3.3.

Tabel 3.3 Data variabel

No	Nama Data	Tipe data	Keterangan
1	Q	uint8	Data citra masukan
2	ekstrak	struct	*H*asil ekstraksi fitur warna dan tekstur citra masukan
3	ekstrak.colorhis	array	*Array* 3 dimensi ekstraksi fitur warna
4	ekstrak.merahBDIP	struct	Hasil ekstraksi BDIP untuk warna merah
5	ekstrak.hijauBDIP	struct	Hasil ekstraksi BDIP untuk warna hijau
6	ekstrak.biruBDIP	struct	Hasil ekstraksi BDIP untuk warna biru
7	ekstrak.brightBDIP	struct	Hasil ekstraksi BDIP untuk *brightness*
8	ekstrak.merahBVLC	struct	Hasil ekstraksi BVLC untuk warna merah
9	ekstrak.hijauBVLC	struct	Hasil ekstraksi BVLC untuk warna hijau
10	ekstrak.biruBVLC	struct	Hasil ekstraksi BVLC untuk warna biru
11	ekstrak.brightBVLC	struct	Hasil ekstraksi BVLC untuk *brightness*
12	train	struct	Hasil ekstraksi fitur warna dan tekstur citra database
13	hasil	array	Hasil jarak citra masukan dengan citra *database*
14	hasil2	array	Hasil jarak citra *query* setelah diurutkan
15	mirip	array	20 citra keluaran hasil temu kembali citra
16	PR	struct	Berisi *precision* dan *recall*



Gambar 3.2 Contoh hasil keluaran citra



Gambar 3.3 Diagram alir sistem secara umum

3.3.1 Ekstraksi Fitur

Ekstraksi fitur dilakukan dengan merubah citra ke dalam ruang warna RGB dan menghasilkan tiga matriks yang mewakilkan warna merah, hijau, dan biru. Kemudian ketiga matriks tersebut akan melalui proses ekstraksi fitur warna dan tekstur. Pada ekstraksi fitur warna akan digunakan metode *color histogram* sedangkan ekstraksi fitur tekstur menggunakan metode BDIP dan BVLC. Terdapat dua macam ekstraksi tekstur yang

dilakukan yaitu ekstraksi tekstur kecerahan (*brightness*) dan ekstraksi tekstur warna (*color*). Perbedaan Ekstraksi fitur tekstur kecerahan dan Ekstraksi fitur tekstur warna adalah pada data citra yang diolah. Pada ekstraksi fitur tekstur *brightness*, intensitas citra didapatkan dari rata-rata intensitas warna merah, hijau, dan biru. Misalkan *R*, *G,* dan *B* masing-masing mewakilkan intensitas warna merah, hijau dan biru maka intensitas kecerahan didapat dengan $I(i,j)=(R(i,j)+G(i,j)+B(i,j))/3$ dengan (i,j) adalah indeks citra. Sedangkan Ekstraksi fitur tekstur warna mengolah ketiga data citra warna yaitu merah, hijau, dan biru secara langsung, sehingga hasil akhir ekstraksi fitur tekstur warna 3 kali lebih besar dari ekstraksi fitur tekstur kecerahan.

3.3.1.1 Histogram Warna

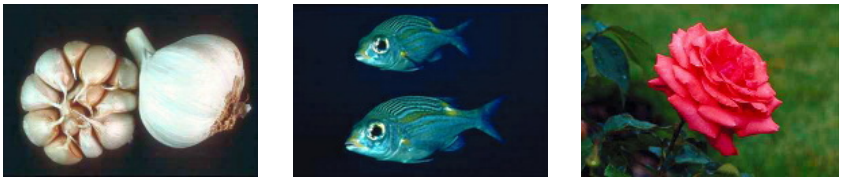
Histogram warna digunakan untuk mengekstrak fitur warna citra. *Color histogram* didapatkan dengan merepresentasikan distribusi warna dalam sebuah citra. Citra asli diubah menjadi matriks *channel* ruang warna RGB yaitu merah, hijau, dan biru. Kemudian masing-masing ruang warna dibagi ke dalam 4 bins, yaitu dengan melakukan kuantisasi pada masing masing komponen warna. Intensitas warna masing-masing komponen dikelompokkan menjadi 4 kelompok dengan rentang intensitas 0-63, 64-127, 128-191, 192-255. Intensitas pada citra akan diubah berdasarkan pembagian kelompok tersebut, angka 1 mewakilkan *range* intensitas 0-63, angka 2 mewakilkan *range* intensitas 64-127, angka 3 mewakilkan range intensitas 128-191 dan angka 4 mewakilkan *range* intensitas 192-255. Histogram warna dibangun dengan menggunakan *array* 3 dimensi, masing-masing dimensi mewakilkan ruang warna merah, hijau, dan biru dengan indeks 1-4 yang merepresentasikan hasil kuantisasi intensitas yaitu pengelompokkan intensitas menjadi 4 kelompok. Histogram warna berisi jumlah kombinasi dari ruang warna dengan indeks sesuai pengelompokkan intensitas dan menghasilkan 64 nilai histogram. Penjelasan lebih lanjut dapat dilihat pada potongan *pseudocode* pada Gambar 3.4.

1. ***Initialization:***
   for $index_r = 0$ *to* $bin_r - 1$
      for $index_g = 0$ *to* $bin_g - 1$
         for $index_b = 0$ *to* $bin_b - 1$
            $H(index_r, index_g, index_b) = 0$
         end for
      end for
   end for
2. ***Histogram updation:***
   for $i = 0$ *to* $M - 1$
      for $j = 0$ *to* $N - 1$
         $index_r = \lfloor R(i,j) \times bin_r/256 \rfloor$
         $index_g = \lfloor G(i,j) \times bin_g/256 \rfloor$
         $index_b = \lfloor B(i,j) \times bin_b/256 \rfloor$
         $H(index_r, index_g, index_b) = H(index_r, index_g, index_b + 1)$
      end for
   end for.

Gambar 3.4 *Pseudocode* histogram warna

3.3.1.2 Block Difference of Inverse Probability (BDIP)

Block Difference of Inverse Probability (BDIP) adalah salah satu metode ekstraksi fitur tekstur untuk kecerahan, metode ini biasanya dikombinasikan dengan metode BVLC yaitu metode ekstraksi fitur untuk kehalusan. Sebelum melakukan perhitungan, citra *query* diubah ke dalam masing-masing ruang warna RGB yaitu merah, hijau, dan biru. Kemudian citra akan diproses dengan membagi citra kedalam blok-blok. Dalam proses perhitungan BDIP, tiap blok berukuran 2×2 piksel tidak tumpang tindih dari citra *I(x,y)* akan dicari nilai BDIP dengan rumus seperti pada persamaan (2.3). Setiap blok-blok piksel akan menghasilkan satu nilai, sehingga didapatkan matriks 2 dimensi berukuran seperempat dari ukuran citra asli. Hasil akhir ekstraksi BDIP merepresentasikan tepi dan batas pada wilayah citra seperti pada Gambar 3.5. Pada Gambar 3.5, citra hasil ekstraksi tekstur BDIP diperbesar empat kali ukuran asli. Penjelasan lebih lanjut untuk proses-proses metode BDIP dapat dilihat pada diagram alir pada Gambar 3.6.



(a)

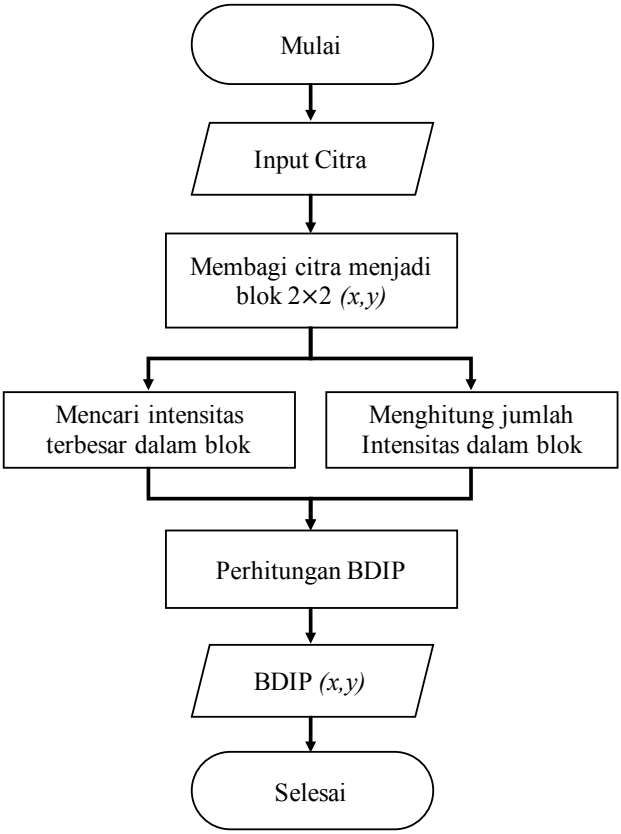
(b)

Gambar 3.5 Citra asli dan hasil ekstraksi tekstur BDIP
(a) Citra asli, (b) Citra BDIP

3.3.1.3 Block Variation of Local Correlation Coefficients (BVLC)

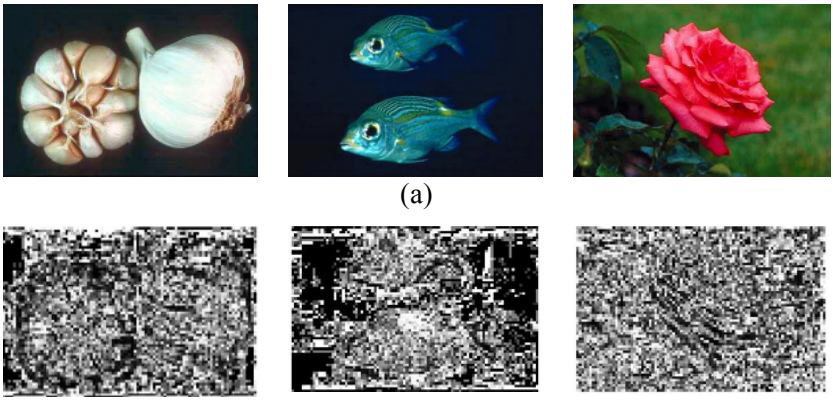
Block Variation of Local Correlation Coefficients (BVLC) merupakan metode ekstraksi fitur tekstur kehalusan. Metode BVLC adalah turunan dari metode VLCC (*variation of local correlation coefficients*) [14], yaitu setiap korelasi koefisien lokal didefinisikan sebagai lokal kovarians yang dinormalisasi dari lokal varians. Cara kerja BVLC adalah dengan menghitung variasi atau perbedaan antara maksimum dan minimum dari korelasi koefisien lokal berdasarkan empat arah yaitu kanan, kiri, atas, dan bawah. Sebelum melakukan perhitungan, citra *query* diubah ke dalam masing-masing ruang warna RGB yaitu merah, hijau, dan biru. Kemudian citra akan diproses dengan membagi citra kedalam blok-blok.

Dalam proses perhitungan BVLC, tiap blok berukuran 2×2 piksel yang tidak tumpang tindih dari citra $I(x,y)$ akan dicari nilai VLCC kemudian akan dihitung nilai maksimum dan minimum untuk mendapatkan nilai akhir BVLC seperti pada persamaan (3). Sama seperti BDIP, pada BVLC setiap blok-blok



Gambar 3.6 Diagram alir proses perhitungan BDIP

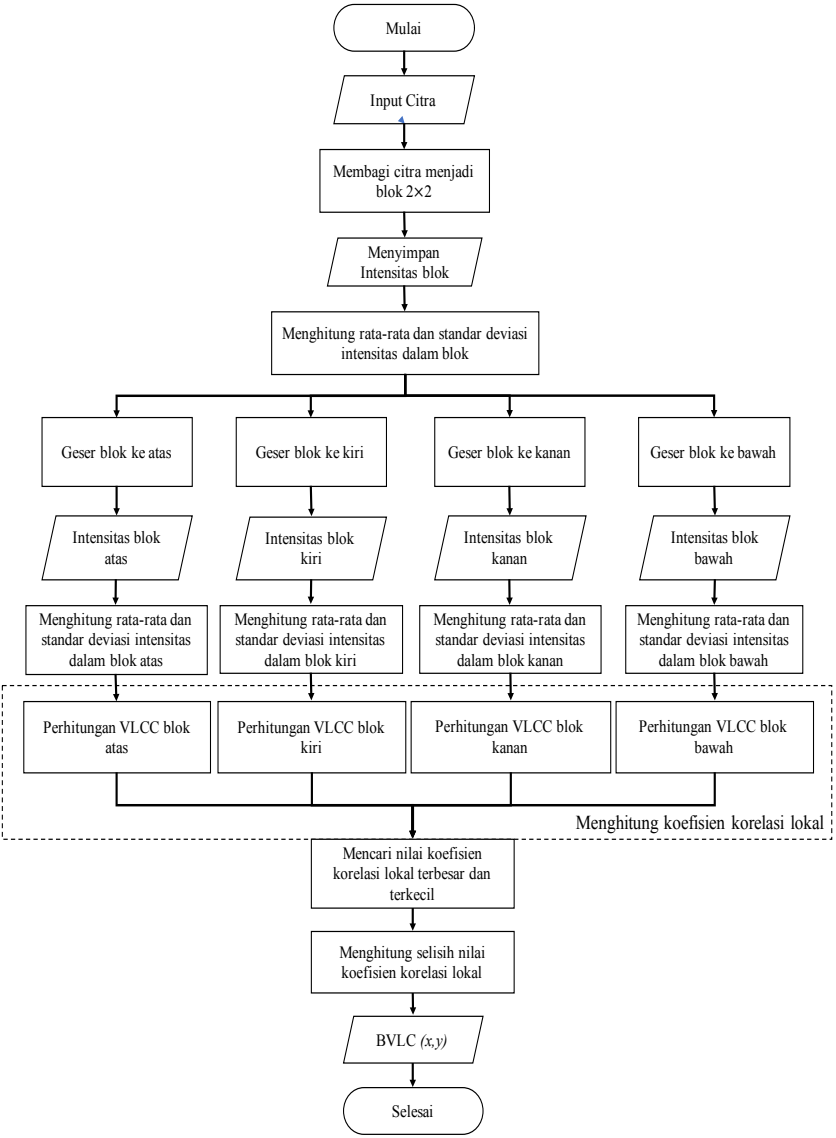
piksel akan menghasilkan 1 nilai, sehingga didapatkan matriks 2 dimensi berukuran seperempat dari ukuran citra asli. Hasil akhir ekstraksi BVLC merepresentasikan tekstur dari keseluruhan citra termasuk wilayah dimana variasi intensitasnya rendah seperti pada Gambar 3.7. Pada Gambar 3.7, citra hasil ekstraksi tekstur BVLC diperbesar empat kali ukuran asli. Penjelasan lebih lanjut untuk proses-proses metode BVLC dapat dilihat pada diagram alir pada Gambar 3.8.



Gambar 3.7 Citra asli dan hasil ekstraksi tekstur BVLC
(a) Citra asli (b) Citra BVLC

3.3.2 Menggabungkan Fitur

Setelah semua fitur di ekstrak, akan didapatkan *array* 3 dimensi berukuran 4×4×4 dan 6 matriks 2-D dari ekstraksi tekstur warna berukuran seperempat citra asli. Sebelum melakukan penggabungan fitur, nilai-nilai ekstraksi fitur warna akan dibagi dengan jumlah seluruh piksel pada citra asli, hal ini dilakukan untuk menormalisasi nilai histogram warna. Sedangkan pada ekstraksi fitur tekstur, 6 matriks ekstraksi fitur tekstur warna terdiri dari 3 matriks 2-D hasil ekstraksi BDIP dan 3 matriks 2-D hasil ekstraksi BVLC, sedangkan untuk ekstraksi fitur tekstur kecerahan akan didapatkan masing-masing 1 matriks 2-D hasil ekstraksi untuk BDIP dan BVLC. Semua matriks yang dihasilkan pada ekstraksi fitur tekstur ini berukuran seperempat dari citra asli. Kemudian, masing-masing matriks 2-D hasil ekstraksi tekstur akan diubah ke dalam *array* 1-D dan kemudian di urutkan berdasarkan nilai terkecil ke terbesar. Array 1-D yang telah diurutkan kemudian akan dipartisi menjadi 2 bagian dengan rata-rata sebagai *threshold*. Hal ini dilakukan terus menerus hingga array 1-D terpartisi menjadi 8 bagian. Kemudian setiap 8 bagian ini masing-masing akan dihitung rata-rata dan standar deviasi,



Gambar 3.8 Diagram alir proses perhitungan BVLC

sehingga akan didapatkan 8 nilai rata-rata dan 8 nilai standar deviasi untuk setiap hasil ekstraksi fitur tekstur. Jika dilakukan ekstraksi fitur tekstur warna maka akan didapatkan tiga kali 8 nilai rata-rata untuk BDIP yaitu rata-rata untuk ekstraksi fitur tekstur warna merah, hijau, biru dan tiga kali 8 nilai standar deviasi untuk warna merah, hijau, biru. Sama halnya dengan BDIP, dari hasil ekstraksi BVLC akan didapatkan tiga kali 8 nilai rata-rata untuk BVLC dan tiga kali 8 nilai standar deviasi. Total hasil ekstraksi fitur tekstur warna untuk BDIP dan BVLC masing-masing menjadi 96 nilai dengan 32 nilai untuk setiap warna merah, hijau, dan biru. Setelah didapatkan semua nilai, kemudian 96 nilai hasil klasifikasi ekstraksi fitur tekstur warna dan hasil ekstraksi fitur warna digabungkan ke dalam suatu *struct*.

3.3.3 Perhitungan Kemiripan

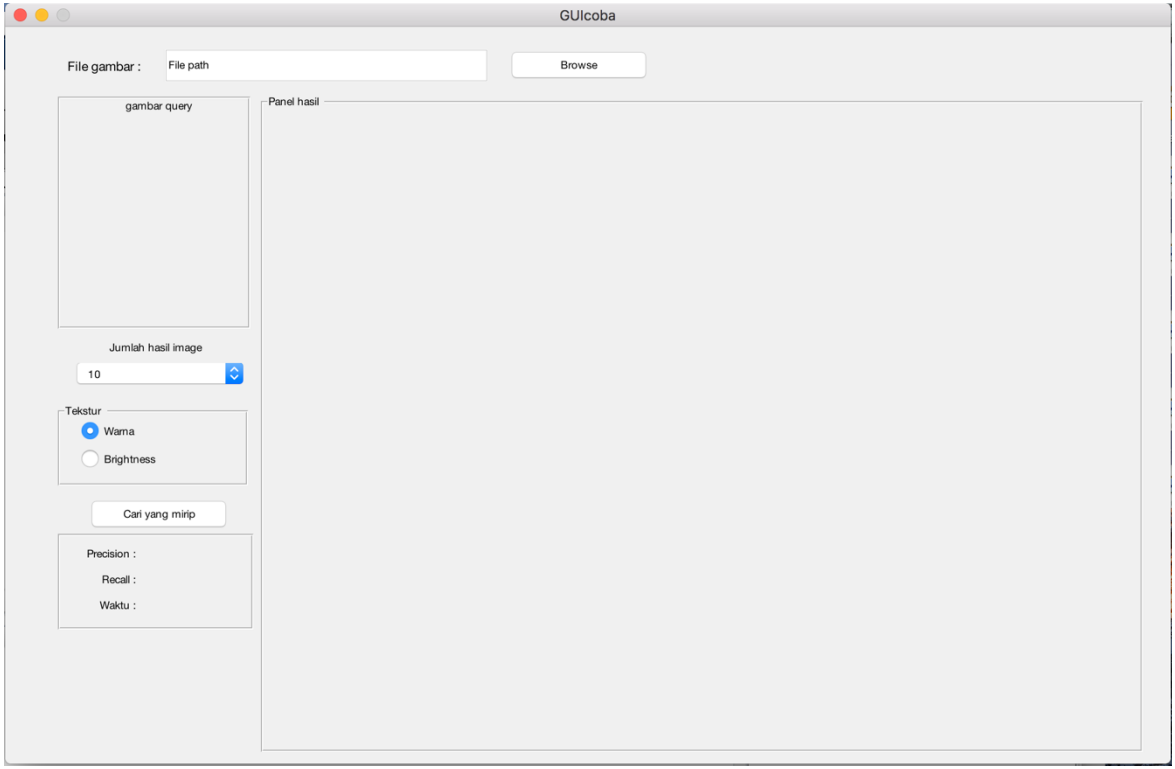
Perhitungan kemiripan bertujuan untuk mengukur jarak fitur-fitur citra *query* dengan citra dalam *database*. Tugas akhir ini menggunakan metode *square chord distance* dalam mengukur jarak fitur-fitur citra (persamaan 2.6). *Square chord distance* akan mengukur jarak antar vektor untuk tiap fitur. Metode ini dianggap menjadi metode pengukuran jarak yang baik untuk gabungan ekstraksi fitur menggunakan histogram warna, BDIP, dan BVLC [2].

3.4 Perancangan Antarmuka

Gambar 3.9 memperlihatkan rancangan antarmuka aplikasi temu kembali citra. Pengguna dapat melakukan hal-hal berikut pada aplikasi:

1. Memilih citra *query* yang ingin diakukan temu kembali citra (klik tombol ‘Browse’)
2. Memilih jumlah hasil citra mirip yang ingin ditampilkan (pilih pada ‘Jumlah hasil image’)
3. Memilih jenis fitur tekstur yang ingin dilakukan yaitu tekstur warna atau brightness (pilih *Radio Button* pada Panel ‘Tekstur’)

4. Memulai proses temu kembali citra (klik tombol ‘Cari yang mirip’).
5. Panel hasil adalah panel yang akan berisi hasil citra yang mirip.
6. Hasil nilai *precision, recall*, dan waktu komputasi ditampilkan pada panel kiri bawah.



Gambar 3.9 Rancanan antarmuka aplikasi

BAB IV
IMPLEMENTASI

Pada bab ini akan diuraikan mengenai implementasi perangkat lunak yang meliputi algoritma dan kode program yang terdapat dalam perangkat lunak terdapat dalam perangkat lunak. Pada tahap implementasi dari tiap fungsi, akan dijelaskan mengenai parameter masukan, keluaran, dan beberapa keterangan yang berhubungan dengan program dan teori.

4.1 Lingkungan Implementasi

Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam implementasi perangkat lunak untuk tugas akhir ini ditampilkan pada Tabel 4.4.

Tabel 4.1 Lingkungan implementasi perangkat lunak

Perangkat	Spesifikasi
Perangkat Keras	Prosesor: 2.7 GHz Intel Core i5 Memori : 8 GB
Perangkat Lunak	Sistem Operasi : macOS Sierra (10.12.1) Perangkat Pengembang : Ms. Word 2016 Matlab R2016b (9.1.0)

4.2 Implementasi Program

Pada bagian ini akan dijelaskan implementasi kode program yang terdapat dalam perangkat lunak. Pada tahap implementasi akan dijelaskan mengenai parameter masukan, keluaran, dan beberapa keterangan yang berhubungan dengan program dan teori.

4.2.1 Implementasi Histogram Warna

Histogram warna digunakan untuk mengekstraksi fitur warna. Histogram warna diimplementasikan pada fungsi colorhis, dapat dilihat pada potongan kode Kode Sumber 4.1. Masukan dari

```
1   for indexR=1:4
2       for indexG=1:4
3           for indexB=1:4
4            colorhis(indexR,indexG,indexB)=0;
5           end
6        end
7   end
8   for i=1:r
9     for j=1:c
10      indexR=floor(double(red(i,j))/64)+1 ;
11      indexG=floor(double(green(i,j))/64)+1;
12      indexB=floor(double(blue(i,j))/64)+1;
13      colorhis(indexR,indexG,indexB)=
        colorhis(indexR,indexG,indexB)+1;
14    end
17  end
18  end
```

Kode Sumber 4.1 Ekstraksi fitur warna (histogram warna)

fungsi histogram warna adalah citra yang telah dibagi ke dalam 3 komponen warna RGB yaitu merah, hijau, dan biru. Keluaran dari fungsi ini adalah array 3 dimensi yang berisi distribusi dari ketiga komponen warna RGB dengan setiap indeks dimensi mewaliki komponen warna.

Pada Kode Sumber 4.1, baris 1 sampai baris 7 merupakan potongan kode untuk inisialisasi array 3 dimensi. Pada baris 10 sampai 12 melakukan kuantisasi warna dengan 4 bin. Pada baris 13 menambahkan jumlah kombinasi warna RGB.

4.2.2 Implementasi *Block Difference of Inverse Probabilities* (BDIP)

Block difference of inverse probabilities (BDIP) digunakan untuk mengekstraksi fitur tekstur kecerahan. Metode BDIP diimplementasikan pada fungsi hitungBDIP yang dapat

```
1  I3=I2(x:x+1, y:y+1);
2  M=max(max(I3));
3  L=sum(sum(I3));
4
5  d=double(L/M);
6  f=double(4-d);
```

Kode Sumber 4.2 Ekstraksi fitur tekstur kecerahan (BDIP)

dilihat pada potongan kode pada Kode Sumber 4.2. Masukan dari fungsi hitungBDIP adalah potongan blok dan indeks blok pada citra asli. Keluaran dari fungsi ini adalah suatu nilai ekstraksi dari satu blok piksel berukuran 2x2. Nilai-nilai ekstraksi dari setiap blok kemudian digabungkan sehingga menjadi matriks citra yang berukuran seperempat citra asli.

Pada Kode Sumber 4.2, baris 1 dilakukan pemotongan citra menjadi blok dengan ukuran 2x2 piksel. Kemudian pada baris selanjutnya dihitung nilai maksimum dan jumlah dari nilai-nilai intensitas yang ada pada blok. Baris 9 adalah perhitungan nilai BDIP dari blok tersebut.

4.2.3 Implementasi *Block Variation of Local Correlation Coefficients* (BVLC)

Block Variation of Local Correlation Coefficients (BVLC) digunakan untuk mengekstraksi fitur tekstur kehalusan citra. Metode BVLC di implementasikan pada fungsi hitungBVLC yang dapat dilihat pada potongan kode pada Kode Sumber 4.3. Masukan dari fungsi hitungBVLC adalah potongan blok dan indeks blok pada citra asli. Pada hitungBVLC dipanggil fungsi nilaipershift1, nilaipershift2, nilaipershift3, dan nilaipershift4 yaitu fungsi yang digunakan untuk menghitung nilai pergesaran blok ke kanan, kiri, atas, dan bawah. Nilai BVLC blok suatu blok adalah selisih dari nilai pergeseran blok tertinggi dikurangi nilai pergeseran blok terendah. Keluaran dari fungsi ini adalah suatu nilai ekstraksi dari 1 blok piksel berukuran 2×2.

```
1  function hasil= hitungBVLC(Q,x,y)
2      shift(1)=nilaipershift1(Q,x,y);
3      shift(2)=nilaipershift2(Q,x,y);
4      shift(3)=nilaipershift3(Q,x,y);
5      shift(4)=nilaipershift4(Q,x,y);
6      maks=max(shift)
7      mini=min(shift)
8       hasil=maks-mini;
9  end
```

Kode Sumber 4.3 Ekstraksi fitur tekstur kehalusan (BVLC)

Nilai-nilai ekstraksi dari setiap blok kemudian digabungkan sehingga menjadi matriks citra yang berukuran seperempat citra asli. Potongan kode perhitungan pergeseran blok dapat dilihat pada Kode Sumber 4.4, 4.5, 4.6, dan 4.7.

Pada Kode Sumber 4.3, baris 2 sampai 5 adalah kode untuk memanggil fungsi pergeseran blok ke kanan, kiri, atas, dan bawah. Pada baris ke-6 dilakukan pencarian nilai hasil pergeseran blok tertinggi sedangkan pada baris 7 dilakukan pencarian nilai hasil pergeseran blok terkecil. Kemudian pada baris ke-8 dihitung selisih dari nilai pergeseran blok tertinggi dengan terendah, nilai selisih inilah yang menjadi nilai BVLC dari suatu blok 2×2 piksel.

Pada Kode Sumber 4.4, baris 2 dan 3 menghitung standar deviasi dan nilai mean blok sebelum pergeseran. Sedangkan pada baris 5 dan 6 menghitung standar deviasi dan nilai mean blok setelah pergeseran ke atas. Kode pada baris ke-12 sampai akhir adalah potongan kode untuk menghitung nilai *variation of local correlation coefficients*. Kemudian hasil perhitungan nilai *variation of local correlation coefficients* pergeseran blok atas akan digunakan pada fungsi hitung BVLC.

Pada Kode Sumber 4.5, baris 2 dan 3 menghitung standar deviasi dan nilai mean blok sebelum pergeseran. Sedangkan pada baris 5 dan 6 menghitung standar deviasi dan nilai mean blok setelah pergeseran ke kiri. Kode pada baris ke-12 sampai akhir

adalah potongan kode untuk menghitung nilai *variation of local correlation coefficients*. Kemudian hasil perhitungan nilai *variation of local correlation coefficients* pergeseran blok atas akan digunakan pada fungsi hitung BVLC.

Pada Kode Sumber 4.6, baris 2 dan 3 menghitung standar deviasi dan nilai mean blok sebelum pergeseran. Sedangkan pada baris 5 dan 6 menghitung standar deviasi dan nilai mean blok setelah pergeseran ke kanan. Kode pada baris ke-12 sampai akhir adalah potongan kode untuk menghitung nilai *variation of local*

```
1  function jum1=nilaipershift1(I,x,y)
2      s=std2(I(x:x+1, y:y+1));
3      m=mean(mean(I(x:x+1, y:y+1)));
4      I2= double(I(x-1:x, y:y+1));
5      s2=std2(I2);
6      m2=mean(mean(I2));
7      pembagi=4*s*s2;
8       if pembagi==0
9           jum1=0;
10       else
11          jum1=0;
12             for i=0:1
13              for j=0:1
14                 xy=double(I(x+i,y+j));
15                 kl=double(I(x+i-1,y+j+0
16                 jum1=double((xy*kl)+jum1
17               end
18             end
19           jum1=double(jum1-(4*m*m2));
20           jum1=double(jum1/pembagi);
21       end
22 end
```

Kode Sumber 4.4 Pergeseran atas blok untuk BVLC

```
1   function jum2=nilaipershift2(I,x,y)
2       s=std2(I(x:x+1, y:y+1));
3       m=mean(mean(I(x:x+1, y:y+1)));
4       I2= double(I(x:x+1, y-1:y));
5       s2=std2(I2);
6       m2=mean(mean(I2));
7       pembagi=4*s*s2;
8        if pembagi==0
9           Jum2=0;
10       else
11          Jum2=0;
12             for i=0:1
13              for j=0:1
14                 xy=double(I(x+i,y+j));
15                 kl=double(I(x+i+0,y+j-1));
16                 Jum2=double((xy*kl)+Jum2);
17               end
18             end
19          Jum2=double(Jum2-(4*m*m2));
20          Jum2=double(Jum2/pembagi);
21       end
22  end
```

Kode Sumber 4.5 Pergeseran kiri blok untuk BVLC

correlation coefficients. Kemudian hasil perhitungan nilai variation of local *correlation coefficients* pergeseran blok kanan akan digunakan pada fungsi hitung BVLC.

Pada Kode Sumber 4.7, baris 2 dan 3 menghitung standar deviasi dan nilai mean blok sebelum pergeseran. Sedangkan pada baris 5 dan 6 menghitung standar deviasi dan nilai mean blok setelah pergeseran ke bawah. Kode pada baris ke-12 sampai akhir

```
1   function jum3=nilaipershift3(I,x,y)
2       s=std2(I(x:x+1, y:y+1));
3       m=mean(mean(I(x:x+1, y:y+1)));
4       I2= double(I(x:x+1, y+1:y+2));
5       s2=std2(I2);
6       m2=mean(mean(I2));
7       pembagi=4*s*s2;
8        if pembagi==0
9            Jum3=0;
10        Else
11           Jum3=0;
12              for i=0:1
13               for j=0:1
14                  xy=double(I(x+i,y+j));
15                  kl=double(I(x+i+0,y+j+1));
16                  Jum3=double((xy*kl)+Jum3);
17                End
18              End
19            Jum3=double(Jum3-(4*m*m2));
20            Jum3=double(Jum3/pembagi);
21        End
22  End
```

Kode Sumber 4.6 Pergeseran kanan blok untuk BVLC

adalah potongan kode untuk menghitung nilai *variation of local correlation coefficients*. Kemudian hasil perhitungan nilai *variation of local correlation coefficients* pergeseran blok bawah akan digunakan pada fungsi hitung BVLC**.**

4.2.4 Implementasi *Square Chord Distance*

Square chord distance digunakan untuk menghitung jarak citra *query* dengan citra dalam database. Implementasi dari

```
1   function jum4=nilaipershift4(I,x,y)
2       s=std2(I(x:x+1, y:y+1));
3       m=mean(mean(I(x:x+1, y:y+1)));
4       I2= double(I(x:x+2, y:y+1));
5       s2=std2(I2);
6       m2=mean(mean(I2));
7       pembagi=4*s*s2;
8        if pembagi==0
9           Jum4=0;
10        Else
11          Jum4
12             for i=0:1
13              for j=0:1
14                 xy=double(I(x+i,y+j));
15                 kl=double(I(x+i+1,y+j+0));
16                 Jum4=double((xy*kl)+Jum4);
17               End
18             End
19           Jum4=double(Jum4-(4*m*m2));
20           Jum4=double(Jum4/pembagi);
21        End
```

Kode Sumber 4.7 Pergeseran bawah blok untuk BVLC

Square chord distance terdapat pada fungsi squarechord. Masukan dari fungsi ini adalah struct yang berisi hasil ekstraksi histogram warna, ekstraksi BDIP dan BVLC. Pada fungsi ini dipanggil fungsi disttekstur yaitu fungsi yang menghitung jarak untuk ekstraksi fitur tekstur. Keluaran dari fungsi ini adalah jumlah dari semua hasil jarak ekstraksi fitur warna dan tekstur. Potongan kode fungsi ini dapat dilihat pada Kode Sumber 4.8.

Pada Kode Sumber 4.8, baris 1 sampai dengan baris ke-11 merupakan kode yang digunakan untuk menghitung jarak fitur warna pada citra. Perhitungan jarak fitur citra mengkoresponden-

```
1     for i=1:4
2         for j=1:4
3             for k=1:4
4                 a=Q(1).colorhis(i,j,k);
5                 b=Q1(1).colorhis(i,j,k);
6                 sc=sqrt(a)-sqrt(b);
7                  sc=sc*sc;
8                  hasil=hasil+sc;
9         end
10    end
11    end
12    hasil=hasil+disttekstur
      (Q(1).merahBDIP,Q1(1).merahBDIP);
13    hasil=hasil+disttekstur
      (Q(1).merahBVLC,Q1(1).merahBVLC);
14    hasil=hasil+disttekstur
      (Q(1).hijauBDIP,Q1(1).hijauBDIP);
15    hasil=hasil+disttekstur
      (Q(1).hijauBVLC,Q1(1).hijauBVLC);
16    hasil=hasil+disttekstur
      (Q(1).biruBDIP,Q1(1).biruBDIP);
17    hasil=hasil+disttekstur
      (Q(1).biruBVLC,Q1(1).biruBVLC);
18    end
```

Kode Sumber 4.8 Perhitungan jarak

sikan 64 nilai histogram warna yang disimpan pada *array* 3 dimensi dengan ukuran array 4×4×4. Perhitungan jarak fitur tekstur dilakukan pada baris kode ke-12 sampai 17, perhitungan jarak fitur tekstur dilakukan dengan mengkorespondensi struct yang berisi 16 nilai BDIP dan BVLC per komponen warna RGB. Untuk menghitung nilai jarak fitur tekstur digunakan fungsi disttekstur. Perhitungan pada kode ini menggabungkan hasil jarak fitur warna dan fitur tekstur warna RGB.

Pada Kode Sumber 4.9, baris ke-2 sampai 5 adalah kode untuk menghitung jarak 8 nilai *mean* BDIP dan BVLC.

```
1   hasil=0;
2   for i=1:8
3   temp=sqrt(A(i).mean)-sqrt(B(i).mean);
4   temp=temp*temp;
5   hasil=hasil+temp;
6
7   temp=sqrt(A(i).deviasi)-sqrt(B(i).deviasi);
8   temp=temp*temp;
9   hasil=hasil+temp;
10  end
```

Kode Sumber 4.9 Perhitungan jarak fitur tekstur

Sedangkan pada baris ke-7 sampai 9 adalah kode untuk menghitung jarak 8 nilai standar deviasi dari BDIP dan BVLC.

4.2.5 Implementasi *Precision and Recall*

Precision dan *recall* digunakan untuk mengukur akurasi citra yang ditemukan. Perhitungan *precision* dan *recall* diimplementasikan pada fungsi precisionrecall. Masukan dari fungsi ini adalah *array* yang berisi nama citra dengan jarak yang paling dekat, jumlah citra yang ingin ditemukan, dan nama citra *query*. Keluaran fungsi ini adalah suatu *struct* yang berisi nilai *precision* dan *recall.* Potongan kode program fungsi precisionrecall dapat dilihat pada Kode Sumber 4.9.

Pada Kode Sumber 4.10, baris 8 sampai 14 adalah menghitung berapa jumlah citra *training* yang kategorinya sesuai dengan citra *query*. Sedangkan pada baris ke 16 merupakan perhitungan *precision* dan baris ke-17 adalah perhitungan *recall*.

```
1   function lol=
    precisonrecall(hasil2, batas, nama)
2   field1='dicari'; val1=[];
3   field2='precision'; val2=[];
4   field3='recall'; val3=[];
5   lol=struct
    (field1,val1,field2,val2,field3,val3);
6
7   jmlh=0;
8   for i=1:batas
9       a=fix(nama/100);
10      b=fix(hasil2(i,3)/100);
11      if a==b
12      jmlh=jmlh+1;
13      end
14  end
15  lol.dicari=nama;
16  lol.precision=jmlh/batas*100; %precision
17  lol.recall=jmlh/20*100; %recall
18  end
```

Kode Sumber 4.10 Perhitungan *precision* dan *recall*

[Halaman ini sengaja dikosongkan]

BAB V
UJI COBA DAN EVALUASI

Pada bab ini akan dijelaskan skenario uji coba dan evaluasi yang dilakukan. Hasil uji coba pada tahap ini akan dievaluasi dengan tujuan memperoleh jawaban dari rumusan masalah. Pembahasan yang dikemukakan meliputi data uji coba, hasil uji coba, dan evaluasi.

5.1 Lingkungan Uji Coba

Lingkungan uji coba yang digunakan dalam pembuatan Tugas Akhir ini meliputi perangkat lunak dan perangkat keras yang digunakan untuk Implementasi Temu Kembali Citra Menggunakan Fitur Warna Berbasis Histogram dan Fitur Tekstur Berbasis Blok. Lingkungan uji coba merupakan computer tempat uji coba perangkat lunak. Berikut adalah lingkungan uji coba yang digunakan pada Tugas Akhir ini.

Tabel 5.1 Lingkungan uji coba

Perangkat	Spesifikasi
Perangkat Keras	Prosesor: Intel(R) Core(TM) i3-2120 CPU @ 3.30GHz 3.30 GHz Memori : 6 GB
Perangkat Lunak	Sistem Operasi : Windows 8.1 Pro Perangkat Pengembang : Ms. Word 2016 Matlab R2013

5.2 Data Uji Coba

Data yang digunakan untuk uji coba implementasi temu kembali citra menggunakan fitur warna berbasis histogram dan fitur tekstur berbasis blok adalah dataset citra Corel-10k. Terdapat dua jenis data yang digunakan yaitu data tes atau data masukan dan data *training* atau data yang akan diuji dalam database. Data tes yang digunakan berjumlah 500 citra yang terdiri dari 5 citra

untuk setiap kategori dengan total 100 kategori. Sedangkan untuk data *training,* berjumlah 2000 citra yang terdiri dari 20 citra pada setiap kategori dengan total 100 kategori. Data tes citra dan data *training* citra tersebut dipilih secara acak.

5.3 Hasil Uji Coba Tiap Proses

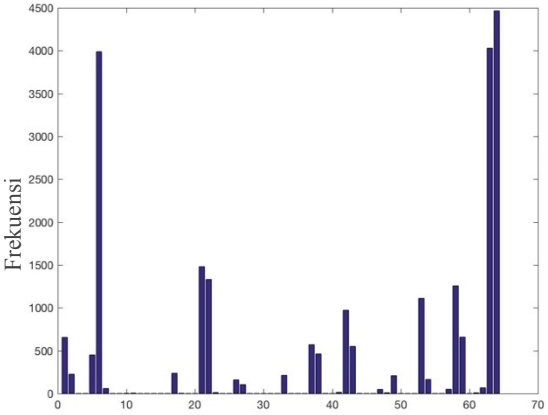
Pada bagian ini akan dijelaskan hasil uji coba pada tiap proses. Proses-proses yang dilakukan pada uji coba ini meliputi ekstraksi fitur warna menggunakan histogram warna, ekstraksi fitur tekstur warna yang menggunakan ketiga komponen RGB untuk mengekstraksi BDIP dan BVLC, dan ekstraksi fitur tekstur *brightness* yang menggunakan rata-rata intensitas dari ketiga komponen RGB untuk mengekstraksi BDIP dan BVLC.

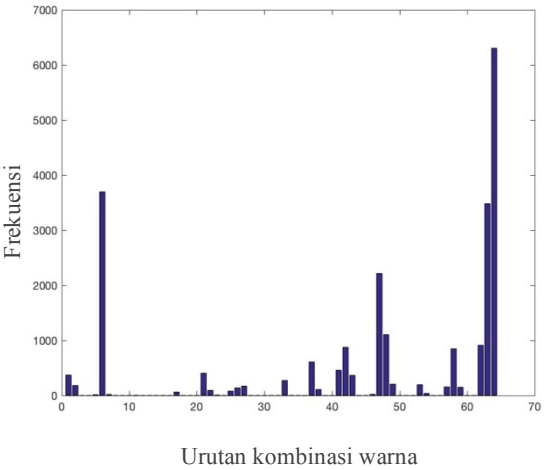
5.3.1 Ekstraksi Fitur Warna

Ekstraksi fitur warna dilakukan menggunakan metode histogram warna dengan melakukan kuantisasi 4 bins. Gambar 5.1 menunjukkan hasil histogram warna dalam bentuk diagram batang. Pada histogram terlihat frekuensi dari kombinasi ketiga warna RGB. Urutan kombinasi warna dapat dilihat pada Tabel 2.2. Hasil Ekstraksi selengkapnya dapat dilihat pada Lampiran A.

5.3.2 Ekstraksi Fitur Tekstur Warna

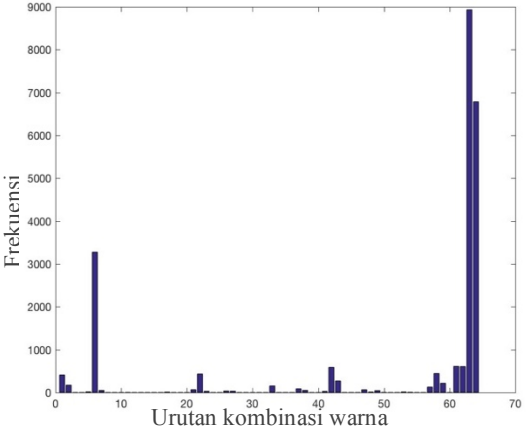
Ekstraksi fitur tekstur warna dilakukan menggunakan dua metode ekstraksi yaitu metode BDIP dan BVLC. Pada ekstraksi fitur tekstur warna, citra asli akan dibagi kedalam ketiga komponen RGB. Ketiga komponen RGB ini akan diproses untuk diekstrak menggunakan metode BDIP dan BVLC. Ekstraksi fitur tekstur warna menghasilkan 96 nilai ekstraksi yang terdiri dari 32 nilai hasil BDIP dan BVLC pada setiap komponen warna yaitu merah, hijau, dan biru. Tiga puluh dua nilai ini terdiri dari 16 nilai untuk BDIP dan 16 nilai untuk BVLC. Enam belas nilai ini terdiri dari 8 nilai mean dan 8 nilai standar deviasi. Gambar hasil ekstraksi BDIP warna dapat dilihat pada Gambar 5.2. Gambar hasil ekstraksi BVLC warna dapat dilihat pada Gambar 5.3.

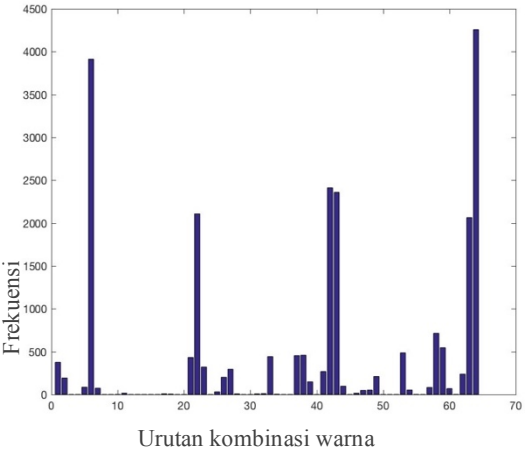




(a) (b)

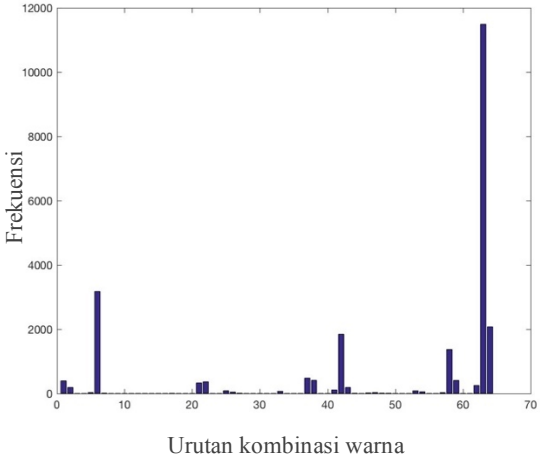
Gambar 5.1 Hasil ekstraksi fitur warna
(a) Citra asli (b) Hasil histogram





(a) (b)

Gambar 5.1 Hasil ekstraksi fitur warna
(a) Citra asli (b) Hasil histogram (Lanjutan)

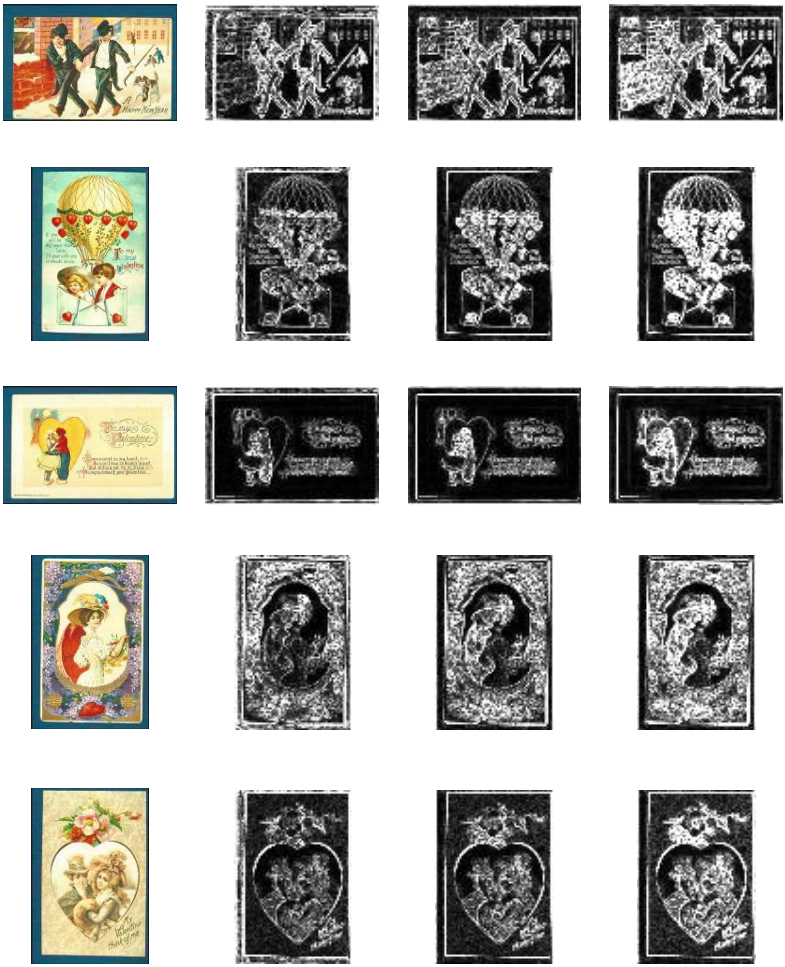


(a) (b)

Gambar 5.1 Hasil ekstraksi fitur warna
(a) Citra asli (b) Hasil histogram (Lanjutan)

5.3.3 Ekstraksi Fitur Tekstur *Brightness*

Ekstraksi fitur tekstur *brightness* dilakukan menggunakan dua metode ekstraksi yaitu metode BDIP dan BVLC. Pada ekstraksi fitur tekstur *brightness*, citra asli akan dibagi kedalam ketiga komponen RGB. Kemudian akan dihitung rata-rata dari intensitas ketiga komponen RGB. Misalkan R, G, dan B masing-masing mewakilkan intensitas warna merah, hijau, dan biru maka rata-rata intensitas *brightness* didapat dengan *I(i,j)=(R(i,j)+G(i,j)+B(i,j))/3* dengan *(i,j)* adalah indeks citra. Hasil rata-rata intensitas ini yang akan diproses untuk diekstrak menggunakan metode BDIP dan BVLC. Ekstraksi fitur tekstur *brightness* menghasilkan 32 nilai ekstraksi yang 16 nilai hasil BDIP dan 16 nilai hasil BVLC. Enam belas nilai ini terdiri dari 8 nilai mean dan 8 nilai standar deviasi. Gambar hasil ekstraksi BDIP *brightness* dapat dilihat pada Gambar 5.4. Hasil Ekstraksi selengkapnya dapat dilihat pada Lampiran.

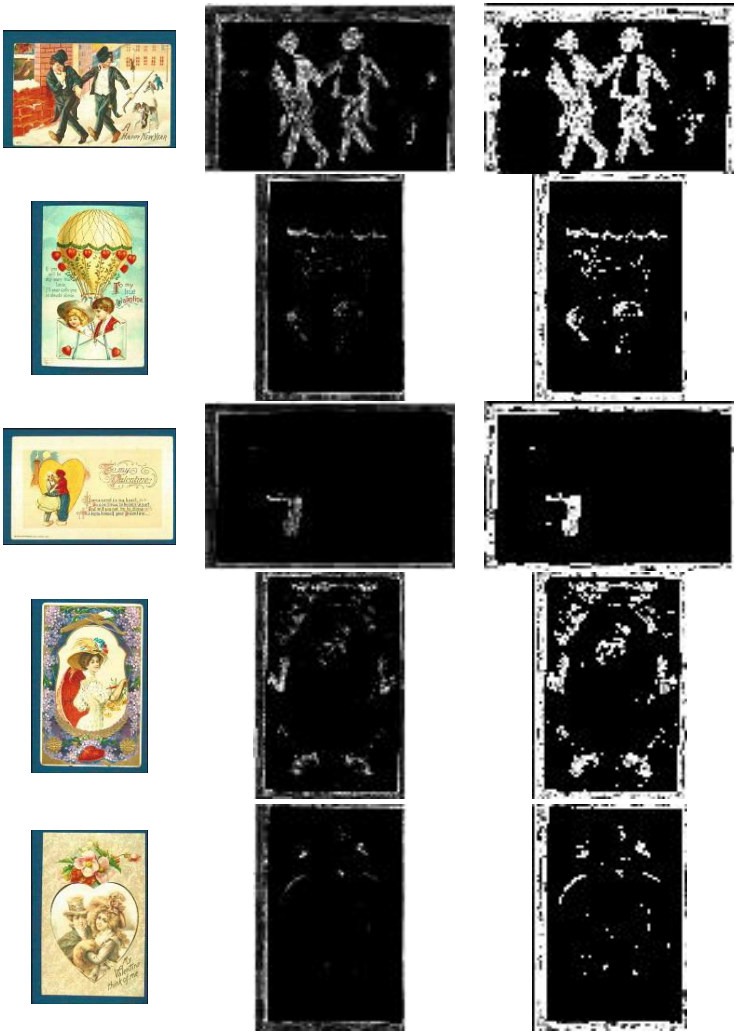


(a) (b) (c) (d)

Gambar 5.2 Hasil ekstraksi tekstur BDIP warna
(a) Citra asli (b) BDIP merah (c) BDIP hijau (d) BDIP biru

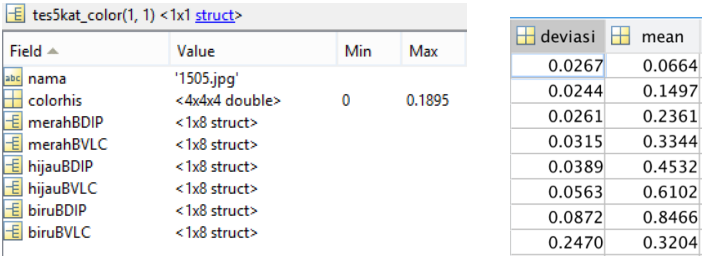
(a) (b) (c) (d)

Gambar 5.3 Hasil ekstraksi tekstur BVLC warna
(a) Citra asli (b) BVLC merah (c) BVLC hijau (d) BVLC biru



(a) (b) (c)

Gambar 5.4 Hasil ekstraksi tekstur BDIP dan BVLC *brightness* (a) Citra asli (b) BDIP *brightness* (c) BVLC *brightness*



tes5kat_color(1, 1) <1x1 struct>

Field	Value	Min	Max
nama	'1505.jpg'		
colorhis	<4x4x4 double>	0	0.1895
merahBDIP	<1x8 struct>		
merahBVLC	<1x8 struct>		
hijauBDIP	<1x8 struct>		
hijauBVLC	<1x8 struct>		
biruBDIP	<1x8 struct>		
biruBVLC	<1x8 struct>		

(a)

deviasi	mean
0.0267	0.0664
0.0244	0.1497
0.0261	0.2361
0.0315	0.3344
0.0389	0.4532
0.0563	0.6102
0.0872	0.8466
0.2470	0.3204

(b)

Gambar 5.5 (a) Hasil *struct* penggabungan fitur (b) isi *struct* hasil ekstraksi tekstur tiap komponen warna

5.3.4 Menggabungkan Fitur

Setelah didapatkan ekstraksi fitur warna dan tekstur, hasil ekstraksi akan digabungkan dalam suatu *struct.* Sebelum menggabungkan fitur, nilai-nilai ekstraksi fitur warna akan dibagi dengan jumlah seluruh piksel pada citra asli, hal ini dilakukan untuk menormalisasi nilai histogram warna. Sedangkan untuk ekstraksi fitur tekstur, hasil ekstraksi akan diubah ke dalam *array* 1-D dan kemudian di urutkan berdasarkan nilai terkecil ke terbesar. *Array* 1-D yang telah diurutkan kemudian akan dipartisi menjadi 2 bagian dengan rata-rata sebagai *threshold*. Hal ini dilakukan terus menerus hingga *array* 1-D terpartisi menjadi 8 bagian. Kemudian setiap 8 bagian ini masing-masing akan dihitung rata-rata dan standar deviasi, sehingga akan didapatkan 8 nilai rata-rata dan 8 nilai standar deviasi untuk setiap hasil ekstraksi fitur tekstur. Kemudian ekstraksi fitur warna dan tekstur digabungkan dalam suatu struct. Hasil penggabungan fitur dapat dilihat pada Gambar 5.5.

5.4 Skenario Uji Coba

Pada bagian ini dijelaskan bagaimana skenario uji coba yang telah dilakukan. Uji coba dilakukan untuk mendapatkan nilai mAP, mAR, dan lama komputasi untuk evaluasi aplikasi temu kembali citra. Perhitungan mAP dan mAR masing-masing

bertujuan untuk mengetahui rata-rata *precision* dan *recall* untuk setiap *query* dengan N_R=20, dimana N_R yang merupakan batas atas jumlah citra ditemukan sama dengan jumlah citra per kategori dalam *database*. Nilai mAP maksimum yang dapat dihasilkan adalah 100%, sedangkan nilai mAR maksimum yang dapat dihasilkan adalah 52.5% dihitung menggunakan rumus pada persamaan (2.11). Hal ini dikarenakan nilai rata-rata *recall* bergantung pada nilai *N* yang berubah-ubah. Rumus mAP dan rumus mAR dapat dilihat pada persamaan (2.12) dan (2.13) Terdapat beberapa jenis uji coba yang dilakukan, yaitu :

a. Uji coba berdasarkan kategori yaitu uji coba ekstraksi fitur warna dan fitur tekstur warna pada dataset dengan jumlah kategori yang berbeda-beda yaitu 100 kategori, 50 kategori, 25 kategori, dan 5 kategori.
b. Uji coba berdsasarkan kombinasi fitur warna dan tekstur warna yaitu uji coba menggunakan ekstraksi masing-masing fitur secara terpisah dan perpaduan beberapa fitur, yaitu fitur warna, fitur tekstur kecerahan (BDIP) dan kehalusan (BVLC), fitur tekstur kecerahan (BDIP), fitur tekstur kehalusan (BVLC), perpaduan fitur warna dan fitur tekstur kecerahan (BDIP), perpaduan fitur warna dan tekstur kehalusan (BVLC), dan perpaduan fitur warna dengan fitur tekstur kecerahan (BDIP) dan tekstur kehalusan (BVLC),.
c. Uji coba perbandingan fitur tekstur warna dan fitur tekstur *brightness* yaitu uji coba dengan membandingkan hasil ekstraksi fitur warna dan fitur tekstur *brightness*.

5.4.1 Uji Coba Berdasarkan Kategori

Pada uji coba pertama, proses ekstraksi fitur yang dilakukan adalah kombinasi dari ekstraksi fitur warna menggunakan histogram warna dan ekstraksi fitur tekstur warna menggunakan BDIP dan BVLC yang diimplementasikan pada ketiga komponen warna RGB. Ekstraksi fitur akan dilakukan pada dataset dengan 100 kategori, 50 kategori, 25 kategori, dan 5 kategori, dengan masing-masing jumlah data *training* 2000, 1000,

Tabel 5.2 Hasil uji coba menggunakan kategori yang berbeda

Jumlah Kategori	mAP (%)	mAR (%)	Waktu (detik)
100	41.29	19.45	4.8147
50	52.28	24.95	2.2821
25	78.71	38.75	1.1572
5	**93.71**	**48.13**	**0.2281**

500, dan 100 citra. Jumlah citra hasil temuan yang digunakan adalah 1-20 citra (sejumlah citra pada database per kategori). Hasil uji coba pertama dapat dilihat pada Tabel 5.2. Pada uji coba pertama didapatkan bahwa semakin sedikit kategori yang digunakan semakin besar nilai mAP dan mAR yang dihasilkan. Hal ini karena tingkat kemiripan antar kategori yang tinggi.

5.4.2 Uji Coba Berdasarkan Kombinasi Fitur Warna dan Fitur Tekstur

Pada uji coba kedua akan dibandingkan hasil kombinasi masing-masing ekstraksi fitur. Dataset yang digunakan adalah dataset dengan 5 kategori karena memiliki nilai mAP dan mAR terbaik dan tingkat kemiripan antar kategori yang relatif lebih kecil. Hasil uji coba kedua dapat dilihat pada Tabel 5.3.

Tabel 5.3 Hasil uji coba kombinasi fitur ekstraksi

Fitur	mAP (%)	mAR (%)	Waktu (detik)
Histogram warna	90.13	46.07	**0.1062**
BDIP	76.65	38.02	0.1593
BVLC	52.83	25.56	0.1587
BDIP+BVLC	86.83	43.40	0.2165
Histogram warna +BDIP	92.35	47.37	0.1668
Histogram warna +BVLC	92.68	47.46	0.1731
Histogram warna +BDIP+BVLC	**93.71**	**48.13**	0.2281

Tabel 5.4 Hasil uji coba komponen brightness dan RGB

Fitur	mAP (%)	mAR (%)	Waktu (detik)
BDIP+BVLC (color)	86.83	43.40	0.2165
BDIP+BVLC (brightness)	83.50	41.93	**0.1387**
Histogram+BDIP+ BVLC (color)	**93.71**	**48.13**	0.2281
Histogram+BDIP+ BVLC (brightness)	92.22	47.55	0.1468

5.4.3 Uji Coba Perbandingan Fitur Tekstur Warna dan Fitur Tekstur *Brightness*

Pada uji coba ketiga akan dibandingkan hasil temu kembali citra menggunakan kombinasi histogram warna dan BDIP, BVLC pada tiga komponen warna dengan histogram warna dan BDIP, BVLC pada komponen warna *brightness*. Hasil uji coba ketiga dapat dilihat pada Tabel 5.4. Pada tabel terlihat nilai mAP, mAR, dan rata-rata waktu komputasi citra.

5.5 Evaluasi

Pada subbab ini akan dijelaskan hasil serangkaian uji coba yang dilakukan dan kendala yang dihadapi selama proses pengerjaan. Evaluasi yang dilakukan adalah pada nilai hasil mAP, mAR, dan waktu komputasi pada percobaan menggunakan jumlah kategori yang berbeda, evaluasi pada percobaan kombinasi fitur warna dan tekstur, dan evaluasi pada perbandingan fitur tekstur warna dan fitur tekstur *brightness*.

5.5.1 Evaluasi Uji Coba Berdasarkan Jumlah Kategori

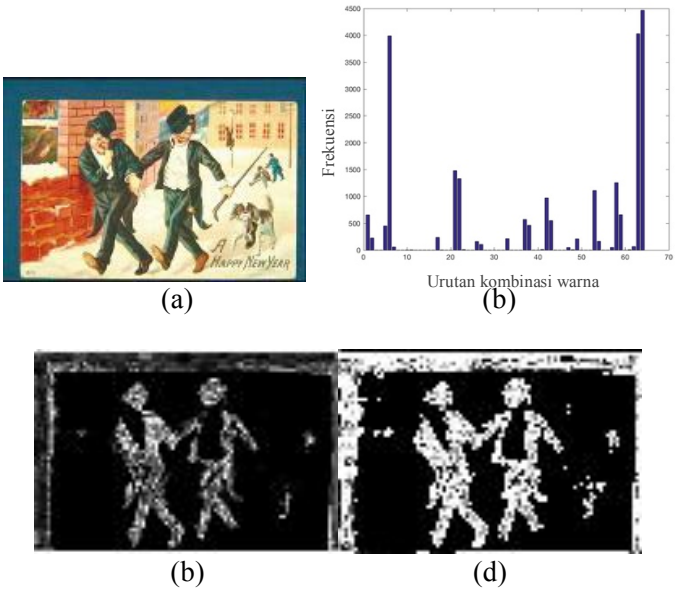
Pada uji coba berdasarkan jumlah kategori didapatkan percobaan dengan nilai mAP terbesar adalah pada percobaan menggunakan 5 kategori. Begitu pula untuk hasil nilai mAR dan

waktu komputasi terbaik didapatkan pada percobaan dengan 5 kategori yaitu 93.71% untuk mAP, 48.13% untuk mAR, dan 0.2281 detik untuk waktu komputasi. Hal ini dikarenakan pada kategori 100, 500, 50, 25 terdapat banyak citra yang mirip namun berada pada kategori yang berbeda seperti kategori citra binatang yang berada di alam bebas dibedakan kategorinya berdasarkan nama binatang tersebut.

5.5.2 Evaluasi Uji Coba Berdasarkan Kombinasi Fitur Warna dan Fitur Tekstur

Pada uji coba berdasarkan kombinasi fitur warna dan fitur tekstur didapatkan hasil nilai rata-rata *precision* terbaik berada pada percobaan dengan kombinasi fitur histogram warna, BDIP, dan BVLC warna dengan nilai mAP 93.71%. Namun percobaan ini memerlukan rata-rata waktu komputasi yang paling lama dibandingkan percobaan lainnya yaitu 0.2281 detik. Pada Tabel 5.3 dapat dilihat persentase mAP yang paling mendominasi adalah mAP untuk percobaan menggunakan histogram warna yaitu 90.13%. Dengan menambahkan fitur tekstur warna menggunakan BDIP dan BVLC persentase mAP meningkat sebanyak 3.71%. Histogram warna juga menjadi fitur dengan waktu komputasi tercepat yaitu 0.1062 detik.

Pada ekstraksi fitur tekstur warna, nilai mAP dan mAR untuk percobaan menggunakan metode BDIP lebih tinggi daripada metode BVLC dengan perbedaan waktu komputasi yang tidak terlalu jauh. Nilai mAP dan mAR untuk BDIP adalah masing-masing 76.65% dan 38.02%, sedangkan untuk BVLC adalah 52.83% dan 25.56%. Namun ketika metode ekstraksi fitur tekstur dikombinasikan dengan ekstraksi fitur warna, kombinasi metode histogram warna dengan BVLC lebih unggul dari kombinasi histogram warna dengan BDIP. Nilai mAP dan mAR untuk kombinasi histogram warna dengan BVLC adalah masing-masing 92.68% dan 47.46%, sedangkan Nilai mAP dan mAR untuk kombinasi histogram warna dengan BDIP adalah masing-masing 92.35% dan 47.37%.



Gambar 5.6 Hasil ekstraksi citra dengan *precision* dan *recall* terbaik (a) Citra asli (b) Histogram warna (c) BDIP (d) BVLC

5.5.3 Evaluasi Uji Coba Perbandingan Fitur Tekstur Warna dan Fitur Tekstur *Brightness*

Pada uji coba perbandingan fitur tekstur warna dan fitur tekstur brightness didapatkan nilai mAP dan mAR tertinggi adalah pada percobaan kombinasi fitur histogram warna dengan fitur tekstur warna yaitu 93.71%. Namun waktu komputasi tercepat didapatkan pada percobaan menggunakan fitur tekstur *brightness* yaitu 0.1387 detik. Hasil terbaik dengan mAP, mAR, dan waktu komputasi terbaik didapatkan dengan percobaan kombinasi fitur histogram warna dengan fitur tekstur *brightness* yaitu 92.22% dan waktu komputasi 0.1468 detik. Perbedaan lama komputasi ini dikarenakan jumlah nilai hasil ekstraksi pada fitur

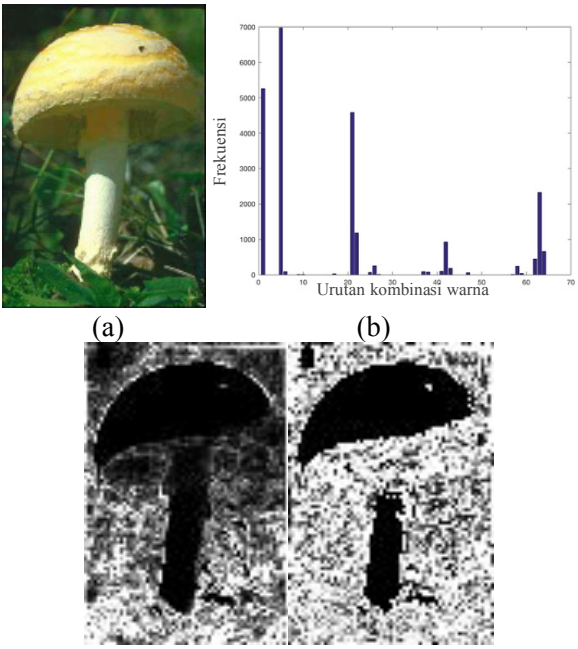
tekstur brightness jauh lebih sedikit dibandingkan fitur tekstur warna, yaitu 32 nilai ekstraksi untuk fitur tekstur *brightness* dan 96 nilai ekstraksi untuk fitur tekstur warna.

5.5.4 Evaluasi Hasil Temu Kembali Citra

Evaluasi hasil temu kembali citra dilakukan pada dataset citra natural dengan 500 citra sebagai datates dan 2000 citra sebagai data training. Metode yang digunakan adalah kombinasi histogram warna dengan BDIP dan BVLC untuk warna brightness. Pada Gambar 5.6 dapat dilihat hasil kombinasi ekstraksi histogram warna dan tekstur *brightness* dengan hasil *precision* terbaik yaitu 100%.

Gambar 5.7 merupakan 12 citra yang ditemukan dan memiliki kategori yang sama dengan citra *query*. Sedangkan Gambar 5.8 merupakan hasil ekstraksi fitur warna dan tekstur *brightness* data citra query dengan hasil *precision* terburuk yaitu 0%. Hasil citra yang ditemukan dapat dilihat pada Gambar 5.9.

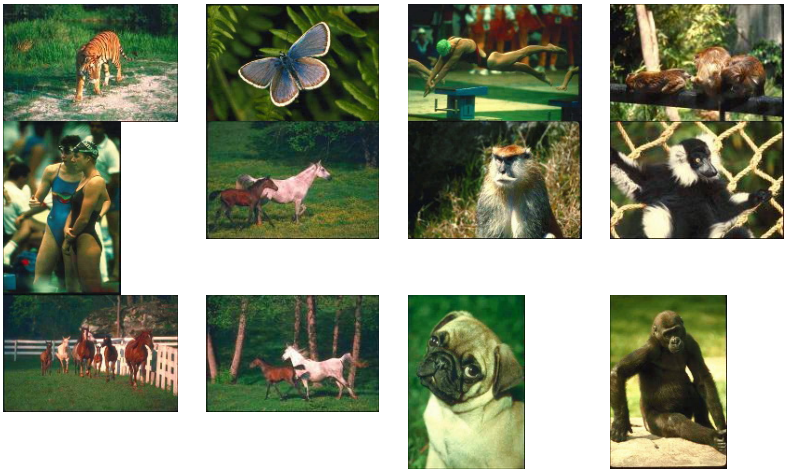
Gambar 5.7 Citra yang ditemukan dengan *precision* terbaik (berdasarkan Gambar 5.6 a)



Gambar 5.8 Hasil ekstraksi citra dengan *precision* dan *recall* terburuk (a) Citra asli (b) Histogram warna (c) BDIP (d) BVLC

Pada citra yang ditemukan (Gambar 5.9) terlihat kemiripan gambar yaitu latar belakang gambar yang terdapat di alam. Hal ini menunjukkan bahwa kategori citra terlalu spesifik dalam mengkategorikan citra dan mengakibatkan nilai precision menjadi rendah. Pada Gambar 5.10 menunjukkan perbandingan ekstraksi citra *query* (kiri) dengan hasil ekstraksi citra (kanan) yang ditemukan dengan nilai jarak terdekat. Meskipun nilai *precision* dan *recall* rendah, citra hasil temu kembali tetap memiliki kemiripan baik dari segi warna ataupun tekstur. Pada Gambar 5.10 terlihat kemiripan frekuensi kombinasi warna RGB pada histogram warna dan pada ekstraksi fitur tekstur terlihat kemiripan tekstur latar belakang citra. Hal ini membuktikan

bahwa kombinasi metode histogram warna, BDIP, dan BVLC mampu menghasilkan citra keluaran yang mirip, namun hasil yang didapatkan belum sesuai dengan kategori citra yang spesifik pada bentuk objek dalam citra.



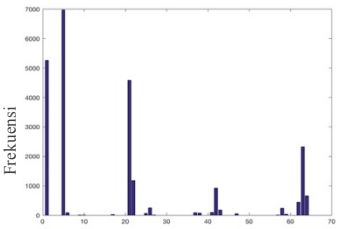
Gambar 5.9 Citra yang ditemukan dengan *precision* terburuk (berdasarkan Gambar 5.8 a)

Citra masukan

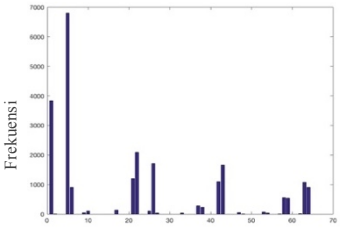
Citra keluaran

(a)

Gambar 5.10 Perbandingan citra *query* dengan citra ditemukan (a) Citra asli (b) Histogram warna (c) BDIP (d) BVLC



Histogram warna citra masukan



Histogram warna citra keluaran

(b)

BDIP citra masukan

BDIP citra keluaran

(c)

BVLC citra masukan

BVLC citra keluaran

(d)

Gambar 5.10 Perbandingan citra *query* dengan citra ditemukan (a) Citra asli (b) Histogram warna (c) BDIP (d) BVLC (Lanjutan)

BAB VI
PENUTUP

Pada bab ini dijelaskan mengenai kesimpulan yang didapatkan setelah melakukan serangkaian uji coba. Selain itu, dijelaskan juga saran pengembangan Sistem Temu Kembali Citra menggunakan Fitur Warna Berbasis Histogram dan Fitur Tekstur Berbasis Blok selanjutnya.

6.1 Kesimpulan

Berdasarkan kajian, uji coba, dan evaluasi yang telah dilakukan, dapat disimpulkan beberapa hal penting sebagai berikut:

1. Ekstraksi fitur warna menggunakan histogram warna berhasil dilakukan dengan hasil mAP tertinggi dibanding ekstraksi fitur tekstur yaitu 90.13% dan waktu komputasi tercepat 0.1062 detik.
2. Ekstraksi fitur tekstur warna menggunakan BDIP berhasil dilakukan dengan hasil mAP 76.65% dan waktu komputasi 0.1593 detik.
3. Ekstraksi fitur tekstur warna menggunakan BVLC berhasil dilakukan dengan hasil mAP 52.83% dan waktu komputasi 0.1587 detik.
4. Akurasi tertinggi diperoleh dari kombinasi ekstraksi fitur warna dan ekstraksi fitur tekstur warna dengan mAP 93.71%, mAR 48.13%, dan waktu komputasi 0.2281 detik.
5. Uji coba dengan ekstraksi fitur warna dan fitur tekstur *brightness* menghasilkan nilai akurasi yang lebih rendah dibanding kombinasi ekstraksi fitur warna dan fitur tekstur warna yaitu mAP 92.22% dan mAR 47.55%, namun memiliki waktu komputasi yang lebih cepat 49% yaitu 0.1468 detik.
6. Hasil uji coba dengan kombinasi fitur terbaik adalah dengan menggunakan kombinasi ekstraksi histogram warna dan ekstraksi tekstur brightness dengan hasil mAP 92.22% dan dengan rata-rata waktu komputasi 0.1468 detik.

6.2 Saran

1. Mengelompokkan kembali kategori dalam dataset, citra yang serupa dikelompokkan menjadi satu. Hal ini perlu dilakukan karena kemiripan citra antara satu kategori dengan kategori yang lain mempengaruhi nilai hasil *precision* dan *recall*.
2. Membandingkan metode jarak dengan metode lain seperti *Manhattan distance* dan *Canberra* untuk mengetahui metode apa yang paling baik untuk digunakan dalam kasus ini.
3. Membandingkan hasil ekstraksi menggunakan komponen RGB dan menggunakan komponen HSV dalam pembagian komponen warna untuk mengetahui ruang warna apa yang paling baik dengan menggunakan metode ini.
4. Melakukan reduksi dimensi menggunakan PCA untuk mengurangi jumlah fitur yang digunakan dan meningkatkan kecepatan komputasi.
5. Menambahkan fitur bentuk untuk menghasilkan keluaran citra yang mirip dengan hasil *precision* dan *recall* yang lebih tinggi.

DAFTAR PUSTAKA

[1] A. I, "Content Based Image Retrieval (CBIR)," *MARKIJAR.Com*. [Online]. Available: http://www.markijar.com/2015/05/content-based-image-retrieval-cbir.html. [Accessed: 13-Dec-2016].

[2] C. Singh and K. Preet Kaur, "A fast and efficient image retrieval system based on color and texture features," *J. Vis. Commun. Image Represent.*, vol. 41, pp. 225–238, Nov. 2016.

[3] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognit.*, vol. 29, no. 1, pp. 51–59, Jan. 1996.

[4] "Image retrieval," *Wikipedia*, 28-Mar-2017. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Image_retrieval&oldid=772705475. [Accessed: 08-Jun-2017].

[5] S. Min, *Handbook of Research on Text and Web Mining Technologies*. IGI Global, 2008.

[6] "Color space," *Wikipedia*, 03-May-2017. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Color_space&oldid=778420050. [Accessed: 08-Jun-2017].

[7] "RGB color model," *Wikipedia*, 08-Jun-2017. [Online]. Available: https://en.wikipedia.org/w/index.php?title=RGB_color_model&oldid=784499411. [Accessed: 08-Jun-2017].

[8] M. J. Swain and D. H. Ballard, "Color indexing," *Int. J. Comput. Vis.*, vol. 7, no. 1, pp. 11–32, Nov. 1991.

[9] Y. J. Ryoo and N. C. Kim, "Valley operator for extracting sketch features: DIP," *Electron. Lett.*, vol. 24, no. 8, pp. 461–463, Apr. 1988.

[10] Sang-Hyun Kim and Gil-Ja So, "The Research Of Depth Discontinuity Detection Using Adaptive DIP," *Asia-Pac. J.*

Multimed. Serv. Converg. Art Humanit. Sociol., vol. 5, no. 3, pp. 37–46, Jun. 2015.

[11] I. Staff, "Standard Deviation," *Investopedia*, 26-Nov-2003. [Online]. Available: http://www.investopedia.com/terms/s/standarddeviation.asp. [Accessed: 08-Jun-2017].

[12] E. W. Weisstein, "Covariance." [Online]. Available: http://mathworld.wolfram.com/Covariance.html. [Accessed: 08-Jun-2017].

[13] I. Staff, "Correlation Coefficient," *Investopedia*, 19-Nov-2003. [Online]. Available: http://www.investopedia.com/terms/c/correlationcoefficient.asp. [Accessed: 28-May-2017].

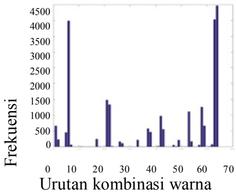
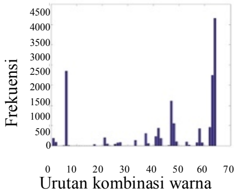
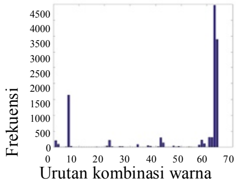
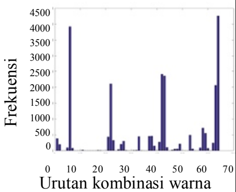
[14] Y. D. Chun, S. Y. Seo, and N. C. Kim, "Image retrieval using BDIP and BVLC moments," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 9, pp. 951–957, Sep. 2003.

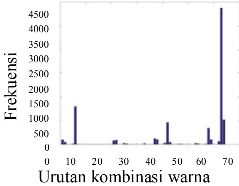
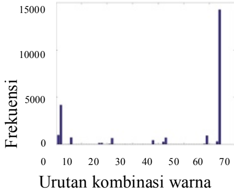
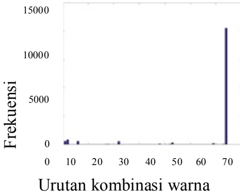
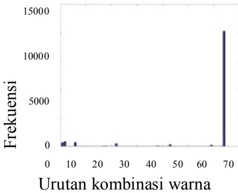
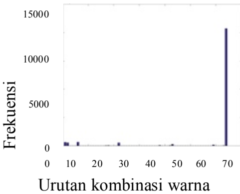
[15] "Precision and recall," *Wikipedia*, 03-Dec-2016. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Precision_and_recall&oldid=752795201. [Accessed: 13-Dec-2016].

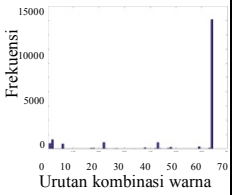
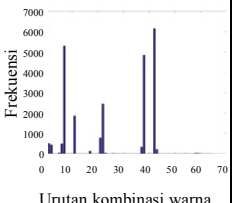
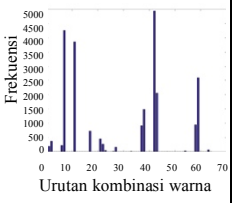
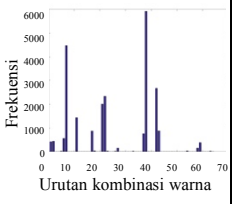
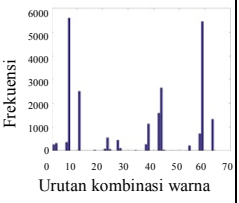
[16] Guang-Hai Liu, "Corel-5K and Corel-10K Datasets." [Online]. Available: http://www.ci.gxnu.edu.cn/cbir/Dataset.aspx. [Accessed: 13-Dec-2016].

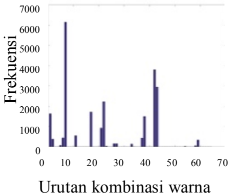
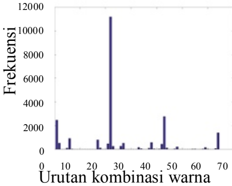
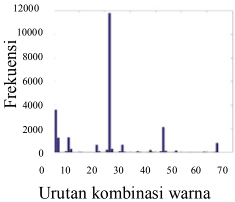
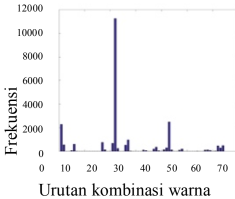
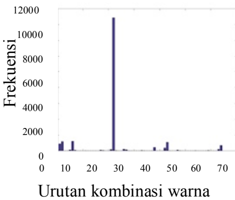
LAMPIRAN

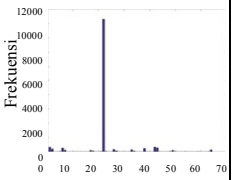
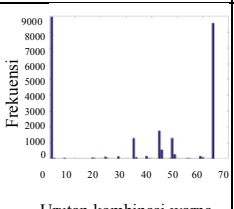
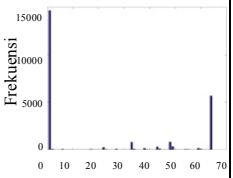
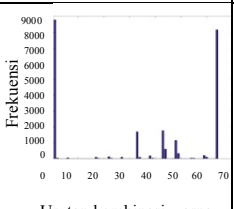
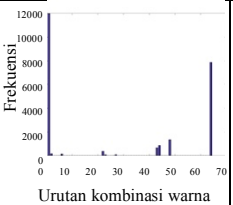
A. HASIL EKSTRAKSI DATA TES 5 KATEGORI

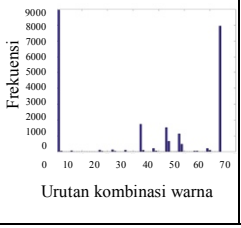
Citra asli	Histogram warna	BDIP	BVLC
			
			
			
			

<table>
<tr><th>Citra asli</th><th>Histogram warna</th><th>BDIP</th><th>BVLC</th></tr>
<tr><td>  </td><td> 
</td><td>  </td><td>  </td></tr>
<tr><td>  </td><td> 
</td><td>  </td><td>  </td></tr>
<tr><td>  </td><td> 
</td><td>  </td><td>  </td></tr>
<tr><td>  </td><td> 
</td><td>  </td><td>  </td></tr>
<tr><td>  </td><td> 
</td><td>  </td><td>  </td></tr>
</table>

Citra asli	Histogram warna	BDIP	BVLC
			
			
			
			
			

Citra asli	Histogram warna	BDIP	BVLC
			
			
			
			
			

Citra asli	Histogram warna	BDIP	BVLC
			
			
			
			
			

Citra asli	Histogram warna	BDIP	BVLC
			

B. RATA-RATA *PRECISION* DAN *RECALL* UNTUK 100 KATEGORI

Kategori Citra	*Precision* (%)	*Recall* (%)
1	66.67	20
2	66.67	20
3	66.67	20
4	68.33	20.5
5	65	19.5
6	68.33	20.5
7	71.67	21.5
8	71.67	21.5
9	66.67	20
10	60	18
11	50	15
12	38.33	11.5
13	33.33	10
14	26.67	8
15	26.67	8
16	25	7.5
17	30	9
18	31.67	9.5
19	28.33	8.5
20	26.67	8
21	30	9
22	25	7.5
23	25	7.5
24	25	7.5
25	28.33	8.5
26	28.33	8.5
27	33.33	10
28	31.67	9.5
29	36.67	11
30	35	10.5
31	35	10.5
32	35	10.5
33	31.67	9.5

Kategori Citra	***Precision* (%)**	***Recall* (%)**
34	33.33	10
35	31.67	9.5
36	38.33	11.5
37	43.33	13
38	46.67	14
39	46.67	14
40	55	16.5
41	53.33	16
42	43.33	13
43	38.33	11.5
44	35	10.5
45	21.67	6.5
46	13.33	4
47	13.33	4
48	21.67	6.5
49	26.67	8
50	25	7.5
51	30	9
52	30	9
53	23.33	7
54	16.67	5
55	20	6
56	11.67	3.5
57	16.67	5
58	18.33	5.5
59	21.67	6.5
60	33.33	10
61	45	13.5
62	38.33	11.5
63	43.33	13
64	48.33	14.5
65	36.67	11
66	26.67	8
67	28.33	8.5
68	20	6
69	8.333	2.5

Kategori Citra	***Precision* (%)**	***Recall* (%)**
70	5	1.5
71	3.333	1
72	21.67	6.5
73	40	12
74	55	16.5
75	73.33	22
76	93.33	28
77	86.67	26
78	81.67	24.5
79	81.67	24.5
80	75	22.5
81	66.67	20
82	66.67	20
83	60	18
84	53.33	16
85	60	18
86	61.67	18.5
87	68.33	20.5
88	76.67	23
89	86.67	26
90	83.33	25
91	90	27
92	70	21
93	66.67	20
94	50	15
95	45	13.5
96	33.33	10
97	53.33	16
98	60	18
99	76.67	23
100	86.67	26

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Sani Puji Rahayu, lahir di Kebumen, Jawa Tengah, pada tanggal 1 Mei 1995, merupakan anak tengah dari tiga bersaudara. Jenjang pendidikan formal yang ditempuh mulai dari SDN Lidah Kulon 1 Surabaya, SMPN 16 Surabaya, hingga SMAN 5 Surabaya. Setelah menyelesaikan sekolah menengah atas, penulis melanjutkan pendidikan tinggi jenjang Strata 1(S1) di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya mulai semester gasal tahun ajaran 2013-2014. Dalam prosesnya, penulis pernah menjadi asisten pengajar di Jurusan Teknik Informatika ITS. Penulis dapat dihubungi via email di sanipujirahayu@gmail.com.