IMPLEMENTASI METODE K-NN UNTUK KLASIFIKASI IMAGE BERDASARKAN TEKSTUR DAN CIRI FITUR WARNA

Nurul Fuad[1)],

Fakultas Teknik Informatika Universitas Islam Lamongan
Jl Veteran No 59 Lamongan
Nurulfuad@unisla.ac.id

Perkembangan teknologi akhir – akhir ini tidak bisa lepas dari pemanfaatan image sebagai salah satu obyek penelitian. *Content Based Image Retrieval* (CBIR) merupakan teknologi yang memungkinkan pencarian image berdasarkan *content*. CBIR bekerja dengan cara mengukur kemiripan image *query* dengan semua image yang ada dalam database sehingga *query cost* berbanding lurus dengan jumlah image dalam database. Membatasi *range* pencarian image dengan cara melakukan klasifikasi merupakan salah satu cara untuk mengurangi *query cost* pada CBIR.

Penelitian ini bertujuan mengimplementasikan *K-Nearest Neighbor* untuk klasifikasi image *landscape* serta mengukur tingkat akurasi dan waktu klasifikasinya. Dalam penelitian ini dibangun sebuah perangkat lunak yang dapat mengekstrak fitur warna dan tekstur dari sebuah image *landscape* dengan menggunakan metode *Color Histogram* dan *Edge Histogram Descriptor*. Hasil dari proses ekstraksi fitur kemudian digunakan oleh perangkat lunak dalam proses *learning* dan klasifikasi dengan metode *K-Nearest Neighbor*. Perangkat lunak dibangun dengan metode analisis dan perancangan terstruktur kemudian diimplementasikan dengan Microsoft Visual Basic.Net

Perangkat lunak yang dihasilkan kemudian diuji dengan parameter tingkat akurasi dan waktu klasifikasi. Hasil pengujian menunjukkan bahwa kombinasi fitur warna dan tekstur memberikan tingkat akurasi yang lebih tinggi dibandingkan dengan klasifikasi berdasarkan fitur warna saja atau tekstur saja namun membutuhkan waktu klasifikasi yang lebih lama.

Kata kunci : *klasifikasi Image, ekstraksi fitur, k-nearest neighbor*

Abstract

Content Based Image Retrieval (CBIR) is a novel technology that provide a 'search by image content' mechanism. CBIR works by measuring similarity between the query image and all images in database. This mechanism leads to a high query cost due to high numbers of image in database. Classifying images into classes in order to limit the query range could be a strategy to minimize CBIR's query cost.

This research's goal is to implement K-Nearest Neighbor to classify landscape images. The accuracy and time of this classifier are measured as well. In this final project, a software which can be used to extract color and texture features from a landscape image is developed. Color and texture features are extracted using Color Histogram and Edge Histogram Descriptor respectively. Afterwards, the result of this feature extraction process is used in learning and classification process where K-Nearest Neighbor classifier is implemented. This software is developed through structured software engineering and implemented using Microsoft Visual Basic.Net.

Software is tested to measure the accuracy and classification time of K-Nearest Neighbor classifier. Test result shows that the accuracy of color and texture based classification outperform the accuracy of color based or texture based classification despite of more time required for classification.

Keywords: *image classification, feature extraction, k-nearest neighbor*

1. Pendahuluan

Pencarian image dengan menggunakan *keyword* yang sering memberikan hasil yang ambigu mendorong lahirnya teknologi *Content Based Image Retrieval* (CBIR). CBIR dapat membantu proses pencarian image berdasarkan *content* image tesebut. CBIR bekerja dengan cara menerima masukan *query* berupa image dan membandingkan image *query* dengan semua image yang ada di dalam database kemudian menampilkan hasil *query* secara terurut berdasarkan ukuran kemiripan antara image *query* dengan image pada database [8].

Sayangnya, jumlah image dalam database yang sangat banyak akan meningkatkan *query cost* pada CBIR. Salah satu solusinya adalah membatasi *range* pencarian dengan melakukan klasifikasi image. Klasifikasi image tidak hanya meningkatkan akurasi, tetapi juga meningkatkan kecepatan *query* pada *image retrieval* [12].

Untuk melakukan klasifikasi terhadap image, harus ditetapkan terlebih dahulu ciri atau fitur yang digunakan sebagai pembeda antara image yang satu dengan image yang lain. Salah satu metode yang dapat digunakan untuk mengklasifikasi image berdasarkan fitur yang dimilikinya adalah *K-Nearest Neighbor*.

Dalam penelitian penelitian ini, yang menjadi pokok pembahasan adalah bagaimana menerapkan *K-Nearest Neighbor* untuk klasifikasi image *landscape* serta bagaimana tingkat akurasi dan waktu klasifikasi dari *K-Nearest Neighbor classifier* tersebut. Agar penelitian tetap terfokus maka fitur yang akan diteliti adalah warna dan tekstur, metode ekstraksi fitur warna yang akan digunakan adalah *Color Histogram* dan untuk mengekstrak fitur tekstur digunakan metode *Edge Histogram Descriptor*.

2. K-Nearest Neighbor

2.1 Klasifikasi

Klasifikasi adalah proses mencari sekumpulan model atau fungsi yang mendeskripsikan dan membedakan kelas-kelas data untuk tujuan agar fungsi tersebut dapat digunakan untuk memprediksi label objek yang label kelasnya tidak diketahui. Model yang dihasilkan adalah berdasarkan hasil analisis terhadap sekumpulan data *learning* [4].

2.2 Ekstraksi Fitur

Fitur dari sebuah image adalah karakteristik atau atribut dari sebuah image yang dapat membedakannya dari image yang lain [11].

Pada referensi [5] disebutkan bahwa jika diinginkan sebuah sistem yang dapat membedakan objek-objek dengan tipe yang berbeda, maka pertama-tama harus ditetapkan karakteristik objek yang dapat diukur sebagai parameter yang mendeskripsikan objek. Karakteristik inilah yang disebut dengan fitur.

Dari beberapa definisi diatas maka ekstraksi fitur dapat didefinisikan sebagai proses mengekstrak karakteristik deskriptif dari sebuah image dengan menggunakan metode tertentu.

Fitur yang dipilih harus mampu mendeskripsikan objek yang diwakilinya dengan baik. Adapun kriteria fitur yang baik menurut [5] adalah sebagai berikut:

1. ***Discrimination***
   Fitur diharapkan mampu memberikan perbedaan nilai yang signifikan untuk objek-objek yang tidak berada dalam satu kelas.
2. ***Reliability***
   Fitur diharapkan mampu memberikan perbedaan nilai yang minimal untuk objek-objek yang berada dalam satu kelas yang sama.
3. ***Independence***
   Jika digunakan beberapa fitur, sebaiknya fitur-fitur tersebut tidak saling berkorelasi satu sama lain, artinya nilai fitur yang satu tidak mempengaruhi nilai fitur yang lain.
4. ***Small Number***
   Fitur diharapkan memiliki dimensionalitas yang kecil sehingga jika direpresentasikan sebagai vektor, maka vektor yang dihasilkan akan memiliki jumlah elemen yang sesedikit mungkin.

2.3 Color Histogram

Warna merupakan fitur yang paling ekspresif dibandingkan dengan fitur visual yang lain [1]. Warna juga merupakan fitur yang paling banyak digunakan dalam *image retrieval* [8]. Salah satu metode yang dapat digunakan untuk mengekstrak fitur warna dari sebuah image adalah dengan menggunakan metode *Color Histogram*.

Histogram dari sebuah image digital dengan *gray level* dalam rentang [0, L-1] merupakan fungsi

$$h(r_k) = n_k \tag{2.1}$$

dimana r_k adalah *gray level* ke-*k* dan n_k adalah jumlah *pixel* dalam image yang memiliki *gray level* ke-*k* [7]. Umumnya histogram mengalami proses normalisasi dengan cara membagi nilai tiap *bin* dengan jumlah seluruh *pixel* yang ada dalam image yang diwakili oleh variabel n. dengan demikian, histogram ternormalisasi dari sebuah image dapat dinyatakan dengan persamaan berikut:

$$p(r_k) = n_k / n \tag{2.2}$$

Ruang warna merepresentasikan warna dalam sistem koordinat sehingga tiap warna direpresentasikan sebagai titik dalam ruang warna tersebut [7]. Dalam ruang warna RGB tiap warna direpresentasikan oleh sebuah titik dalam ruang vektor 3 dimensi dengan sumbu R, G dan B yang menyatakan derajat warna merah, hijau dan biru yang menyusun warna tersebut.

Keuntungan menggunakan ruang warna RGB adalah umumnya image digital tersedia dalam format bitmap yang menggunakan ruang warna RGB. Oleh karena itu untuk membuat histogram dari image digital tidak harus melewati proses konversi ke ruang warna RGB terlebih dahulu.

Fitur warna yang diekstrak diharapkan memiliki dimensionalitas yang seminimal mungkin. Oleh karena itu, histogram dibentuk setelah melalui proses kuantisasi untuk mereduksi jumlah *gray level*.

Dalam *Uniform Quantization*, tiap sumbu dalam ruang warna dikuantisasi secara terpisah. Tiap sumbu dibagi-bagi menjadi beberapa segmen yang ukurannya sama [10]. *Uniform Quantization* dinyatakan dengan persamaan sebagai berikut [6]:

$$i = \left\lfloor \frac{y - y_L}{y_U - y_L} (N-1) \right\rfloor \tag{2.3}$$

Salah satu keuntungan menggunakan metode *Uniform Quantization* yaitu waktu yang dibutuhkan untuk proses kuantisasi adalah yang paling cepat dibandingkan dengan menggunakan metode kuantisasi yang lain. [10]

2.4 Edge Histogram Descriptor

Edge Histogram Descriptor (EHD) mengimagekan distribusi spasial *edge.* Distribusi *edge* merupakan karakteristik tekstur yang berguna dalam proses *image matching* bahkan dengan kondisi dimana image tidak memiliki tekstur yang homogen [1]. Langkah-langkah untuk mengekstrak fitur tekstur dengan menggunakan metode EHD adalah sebagai berikut [3]:

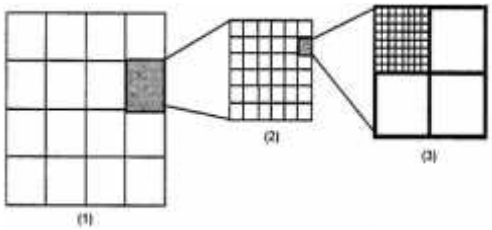

Image 2-1: Ilustrasi ekstraksi fitur tekstur dengan menggunakan EHD

1. Bagi image menjadi 4x4 region yang sama besar
2. Bagi tiap region menjadi *sub-block* dengan ukuran yang sama
3. Tiap *sub-block* dibagi menjadi 2x2 partisi
4. Untuk setiap *sub-block*, nilai dalam tiap partisi dirata-ratakan sehingga tiap *sub-block* dapat diperlakukan sebagai image 2x2 *pixel*
5. Terapkan *edge detector* pada tiap *sub-block*. Sebuah *sub-block* dinyatakan sebagai *edge block* jika hasil operasi *sub-block* dengan *edge detector* melebihi nilai *edge treshold* yang telah ditetapkan sebelumnya.
6. Buat histogram yang mengimagekan distribusi *edge block.* Histogram berukuran 240 bin, diperoleh dari 16 region x 5 *edge orientation* x 3 komponen warna (R,G,B)

Adapun *edge detector* standar yang digunakan pada MPEG-7 ISO/IEC terdiri dari 5 operator, masing-masing untuk mendeteksi *edge* dengan orientasi vertikal, horisontal, 45 derajat, 135 derajat dan *isotropic* atau tak berarah.

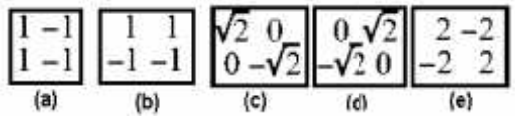

Image 2-2: Edge detector standar MPEG-7 ISO/IEC

Beberapa parameter EHD dapat mengalami penyesuaian tergantung pada kebutuhan. Salah satu contohnya adalah dalam [2] image tidak dibagi menjadi 4x4 region, *edge detector* yang digunakan adalah *Sobel operator*, sedangkan nilai *edge treshold* ditetapkan sama dengan 100. Pada penelitian ini, proses ekstraksi fitur dengan metode EHD dilakukan terhadap image yang dibagi menjadi 4x4, 3x3 dan 2x2 region. *Edge detector* yang digunakan adalah *edge detector* standar MPEG-7 ISO/IEC dengan nilai *edge treshold* = 100.

2.5 Prinsip Kerja K-Nearest Neighbor

K-Nearest Neighbor adalah *supervised learning algorithm* dimana sebuah objek diklasifikasikan berdasarkan kelas mayoritas dari *k* buah tetangga terdekatnya. Klasifikasi memanfaatkan mekanisme *voting* dari *k* buah objek terdekat dan bila hasil *voting* seri, maka label untuk objek akan dipilih secara acak. [9]

K-Nearest Neighbor berdasarkan konsep '*learning by analogy*'. Data *learning* dideskripsikan dengan atribut numerik *n*-dimensi. Tiap data *learning* merepresentasikan sebuah titik dalam ruang *n*-dimensi [4].

Jika sebuah data *query* yang labelnya tidak diketahui diinputkan, maka *K-Nearest Neighbor* akan mencari *k* buah data *learning* yang jaraknya paling dekat dengan data *query* dalam ruang *n*-dimensi. Jarak antara data *query* dengan data *learning* dihitung dengan cara mengukur jarak antara titik yang merepresentasikan data *query* dengan semua titik yang merepresentasikan data *learning* dengan rumus *Euclidean Distance*.

Diberikan 2 buah titik P dan Q dalam sebuah ruang vektor *n*-dimensi dengan P(p_1, p_2,…, p_n) dan Q(q_1, $q_{2,...}$,q_n), maka jarak antara P dan Q dapat diukur dengan menggunakan persamaan *Euclidean Distance* sebagai berikut [4]:

$$D_{euc}(P,Q) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2} \quad (2.4)$$

dimana P dan Q adalah titik pada ruang vektor *n* dimensi sedangkan p*i* dan q*i* adalah besaran skalar untuk dimensi ke *i* dalam ruang vektor *n* dimensi.

K buah data *learning* terdekat akan melakukan *voting* untuk menentukan label mayoritas. Label data *query* akan ditentukan berdasarkan label mayoritas dan jika ada lebih dari satu label mayoritas maka label data *query* dapat dipilih secara acak di antara label-label mayoritas yang ada.[9]

2.6 Akurasi Sistem

Tingkat akurasi sebuah *classifier* untuk sekelompok data uji adalah rasio perbandingan jumlah data uji yang dapat diklasifikasikan dengan benar dengan jumlah seluruh data uji. [4]. Dengan demikian, tingkat akurasi *K-Nearest Neighbor* untuk sekelompok data uji *T* dapat dinyatakan dalam persamaan berikut:

$$Acc(T) = {}^{T_{pos}}\!/_{n} \quad (2.5)$$

dimana T_{pos} adalah jumlah data uji yang diklasifikasikan dengan benar dan *n* adalah jumlah data uji seluruhnya.

Dengan demikian tingkat akurasi maksimum = 1 bila semua data uji dapat diklasifikasikan dengan benar dan tingkat akurasi minimum = 0 jika tidak satupun data uji dapat diklasifikasikan dengan benar. Akurasi dapat dinyatakan dalam satuan persen dengan mengalikan hasil perhitungan dengan 100%.

3. Framework Sistem

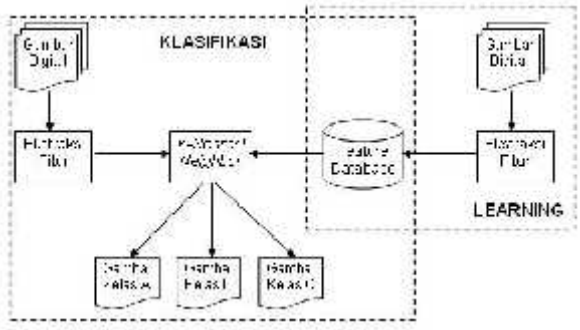

Image 3-1: Framework Sistem

Perangkat lunak akan terdiri dari 2 buah proses utama yang saling berkaitan, yaitu proses *learning* dan proses klasifikasi.

Input untuk proses *learning* adalah kumpulan *learning image* yang sudah diketahui label kelasnya. Adapun output yang dihasilkan adalah fitur image yang disimpan dalam sebuah *feature database*.

Input untuk proses klasifikasi adalah image yang akan diklasifikasikan dan fitur *learning image* yang tersimpan dalam *feature database.* Adapun outputnya adalah label kelas image yang diinputkan.

Dengan demikian jelaslah bahwa proses *learning* harus dilakukan sebelum proses klasifikasi sebab output dari proses *learning* menjadi salah satu input yang dibutuhkan dalam proses klasifikasi.

3.1 Langkah-langkah Tahap Learning

Pada proses *learning*, input perangkat lunak adalah kumpulan *learning image* yang telah diketahui label kelasnya. Semua *learning image* akan diekstrak fiturnya dengan menggunakan metode *Color Histogram* dan *Edge Histogram Descriptor*. Hasil dari proses ekstraksi fitur akan disimpan dalam sebuah *feature database.*

Berikut ini adalah langkah-langkah yang dilakukan dalam proses *learning*:

1. Membaca data bitmap *learning image*
2. Melakukan ekstraksi fitur wana dengan metode Color Histogram
3. Melakukan ekstraksi fitur tekstur dengan metode Edge Histogram Descriptor

3.2 Langkah-langkah Tahap Classification

Pada proses klasifikasi, input perangkat lunak adalah *testing image* yang belum diketahui label kelasnya. *Testing image* akan mengalami proses ekstraksi fitur. Hasil dari proses ekstraksi fitur akan dibandingkan dengan fitur yang tersimpan dalam *feature database* untuk menentukan label kelas *testing image.*

Berikut ini adalah langkah-langkah yang dilakukan dalam proses klasifikasi:

1. Membaca data bitmap *testing image*
2. Melakukan ekstraksi fitur warna dengan metode Color Histogram
3. Melakukan ekstraksi fitur tekstur dengan metode Edge Histogram Descriptor
4. Mengukur perbedaan antara hasil ekstraksi fitur *testing image* dengan fitur learning image yang tersimpan dalam *feature database*
5. Melakukan voting untuk menentukan label kelas image

3.3 Perancangan Sistem

Perancangan perangkat lunak dilakukan dengan metode terstruktur. Adapun model-model yang dihasilkan adalah sebagai berikut:

1. Diagram Konteks
2. *Data Flow Diagram* (DFD)
3. Spesifikasi Proses
4. Kamus Data.

Berikut ini adalah diagram konteks dan *data flow diagram* dari sistem yang telah dibangun:

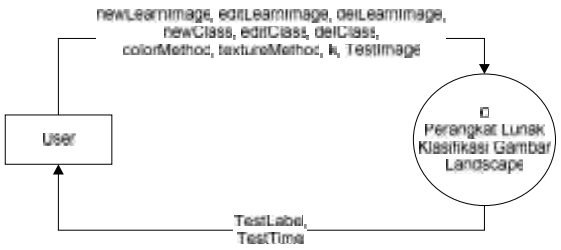

Image 3-2: Diagram Konteks

Diagram konteks hanya mengimagekan input dan output sistem secara keseluruhan. Adapun proses-proses inti yang ada di dalam sistem serta aliran data yang terjadi antar proses diimagekan dalam *data flow diagram* berikut ini:

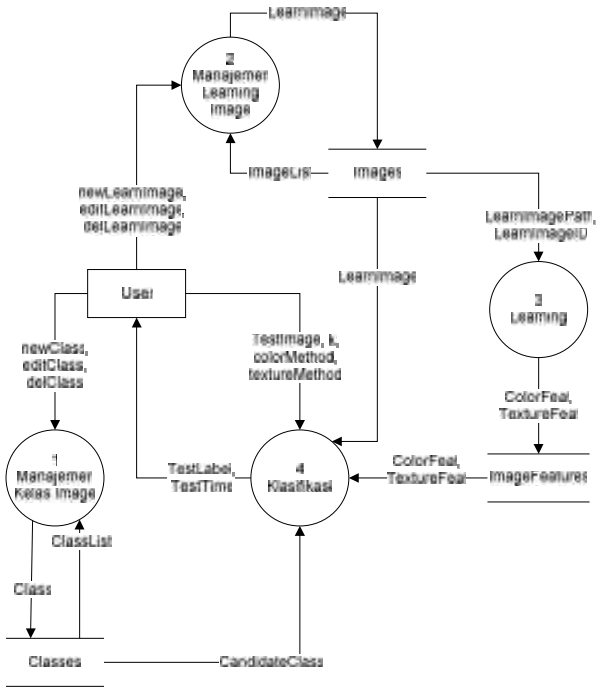

Image 3-3: DFD Level 1

Adapun untuk keperluan penyimpanan data, maka diperlukan sebuah basis data. Rancangan basis data dapat dilihat pada diagram Entity-Relationship berikut ini:

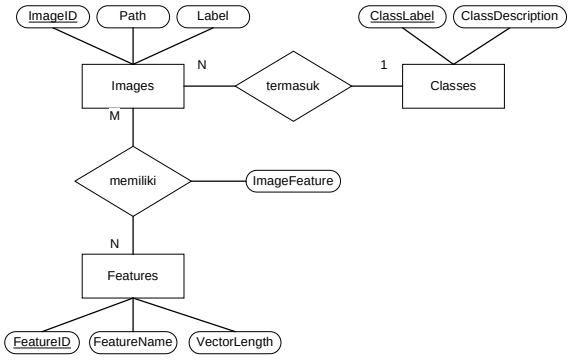

Image 3-4 Entity Relationship Diagram

Adapun hasil implementasi dari rancangan basis data pada diagram ER di atas dapat dilihat melalui struktur tabel berikut ini:

Tabel 3-1: Struktur tabel 'Classes'

No	Field	Tipe Data	Keterangan
1	ClassLabel	String	Primary key, label dari kelas
2	ClassDescription	String	Deskripsi kelas

Tabel 3-2: Struktur tabel 'Features'

No	Field	Tipe Data	Keterangan
1	FeatureID	Number	Primary key
2	FeatureName	String	Nama fitur
3	VectorLength	Number	Jumlah elemen vektor

Tabel 3-3: Struktur tabel 'Images'

No	Field	Tipe Data	Keterangan
1	ImageID	Number	Primary key
2	Label	String	Label dari *learning image*, foreign key yang mengacu ke field 'ClassLabel' pada tabel 'Classes'
3	Path	String	Path absolute ke file *learning image*

Tabel 3-4: Struktur tabel 'ImageFeatures'

No	Field	Tipe Data	Keterangan
1	ImageID	Number	Foreign key yang mengacu ke field 'ImageID' pada tabel 'Images'
2	FeatureID	Number	Foreign key yang mengacu ke field 'FeatureID' pada tabel 'Feature'
3	ImageFeature	String	Fitur yang diekstrak dari image dan disimpan dalam bentuk string.

4. Pengujian

Data yang digunakan untuk pengujian berasal dari *Best Photo Collection CD* yang terdiri dari 200 image yang terbagi dalam 4 kelas yaitu *canyon, mountain, sunset* dan *waterfall.*

Tabel 3-1: Data untuk pelatihan dan pengujian

Kelas	Jumlah data learning	Jumlah data testing
Canyon	25 image	25 image
Mountain	25 image	25 image
Sunset	25 image	25 image
Waterfall	25 image	25 image
Total	**100 image**	**100 image**

Parameter yang diuji adalah tingkat akurasi dan waktu klasifikasi. Tingkat akurasi dihitung dengan menghitung persentase image yang dapat diklasifikasikan dengan benar dalam satuan persen sedangkan waktu klasifikasi dihitung mulai proses pembacaan data bitmap sampai menghasilkan label kelas dan diukur dalam satuan detik per image.

Total jumlah percobaan yang dilakukan adalah 375 percobaan yang terdiri dari:

1. 75 percobaan klasifikasi berdasarkan fitur warna yang diekstrak dengan metode *Color Histogram* 256, 4 dan 32 bin untuk nilai *k* masing-masing 1 sampai 25.
2. 75 percobaan klasifikasi berdasarkan fitur tekstur yang diesktrak dengan metode *Edge Histogram Descriptor* 4x4, 3x3 dan 2x2 region untuk nilai *k* masing-masing 1 sampai 25.
3. 225 percobaan klasifikasi berdasarkan fitur warna dan tekstur yang diekstrak dengan 9 kombinasi metode ekstraksi fitur warna dan tekstur untuk nilai k masing-masing 1 sampai 25.

Tabel 4-2: Skenario Pengujian

Variabel Klasifikasi			Jumlah Percobaan
Fitur	**Metode Ekstraksi Fitur**	**Nilai k**	
Warna	Color Histogram 256 bin	1 sampai 25	25
	Color Histogram 64 bin	1 sampai 25	25
	Color Histogram 32 bin	1 sampai 25	25
Tekstur	Edge Histogram 4x4 Region	1 sampai 25	25
	Edge Histogram 3x3 Region	1 sampai 25	25
	Edge Histogram 2x2 Region	1 sampai 25	25
Warna dan Tekstur	Color Histogram 256 bin + Edge Histogram 4x4 Region	1 sampai 25	25
	Color Histogram 256 bin + Edge Histogram 3x3 Region	1 sampai 25	25
	Color Histogram 256 bin + Edge Histogram 2x2 Region	1 sampai 25	25
	Color Histogram 64 bin + Edge Histogram 4x4 Region	1 sampai 25	25
	Color Histogram 64 bin + Edge Histogram 3x3 Region	1 sampai 25	25
	Color Histogram 64 bin + Edge Histogram 2x2 Region	1 sampai 25	25
	Color Histogram 32 bin + Edge Histogram 4x4 Region	1 sampai 25	25
	Color Histogram 32 bin + Edge Histogram 3x3 Region	1 sampai 25	25
	Color Histogram 32 bin + Edge Histogram 2x2 Region	1 sampai 25	25
Total Jumlah Percobaan			**375**

Dalam tiap percobaan digunakan 100 image testing. Adapun hasil pengujian dapat dilihat dalam grafik berikut:

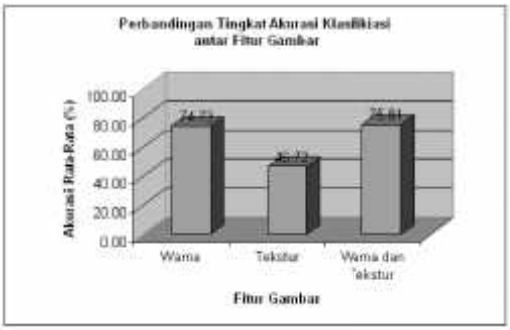

Image 4-1: Grafik perbandingan tingkat akurasi klasifikasi antar fitur

Fitur image yang digunakan dalam klasifikasi akan mempengaruhi tingkat akurasi klasifikasi. Klasifikasi dengan berdasarkan fitur warna dan tekstur memiliki tingkat akurasi rata-rata yang paling tinggi yaitu sebesar 75,81%.

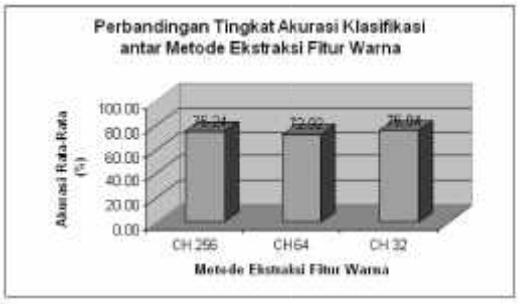

Image 4-2: Grafik perbandingan tingkat akurasi klasifikasi antar metode ekstraksi fitur warna

Metode ekstraksi fitur warna berpengaruh terhadap tingkat akurasi klasifikasi. Adapun metode ekstraksi fitur warna yang memberikan tingkat akurasi klasifikasi tertinggi adalah *Color Histogram 32 bin* (CH 32) yaitu sebesar 76,04 %.

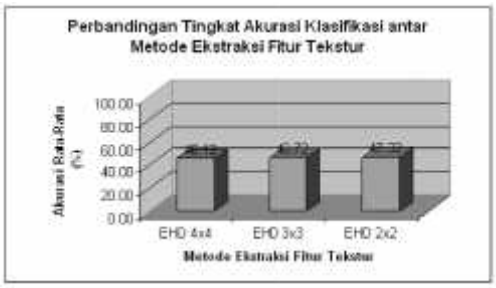

Image 4-3: Grafik perbandingan tingkat akurasi kaslifkasi antar metode ekstraksi fitur

Metode ekstraksi fitur tekstur berpengaruh terhadap tingkat akurasi klasifikasi. Adapun metode ekstraksi fitur tekstur yang memberikan tingkat akurasi klasifikasi tertinggi adalah *Edge Histogram Descriptor 2x2 Region* (EHD 2x2) yaitu sebesar 47,32 %.

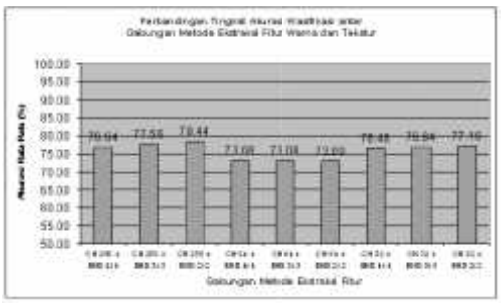

Image 4-4: Grafik perbandingan tingkat akurasi klasifikasi antar gabungan metode ekstraksi fitur warna dan tekstur

Dari grafik di atas terlihat bahwa gabungan metode ekstraksi fitur warna dan tekstur yang memberikan tingkat akurasi klasifikasi paling tinggi adalah

Color Histogram 256 bin (CH 256) dan *Edge Histogram Descriptor 2x2 Region* (EHD 2x2). Gabungan kedua metode ini memberikan tingkat akurasi rata-rata sebesar 78,44%.

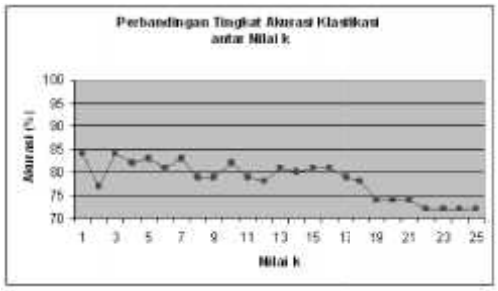

Image 4-5: Grafik perbandingan tingkat akurasi klasifikasi antar nilai k

Grafik di atas menunjukkan bahwa tingkat akurasi klasifikasi berfluktuasi tergantung nilai *k* yang diinputkan namun cenderung menurun seiring naiknya nilai *k*. Untuk klasifikasi berdasarkan fitur warna dengan metode *ColorHistogram 256 bin* dan fitur tekstur dengan metode *Edge histogram 2x2 Region*, tingkat akurasi paling tinggi diperoleh dengan nilai *k*=1 dan *k*=3 yaitu sebesar 84 %.

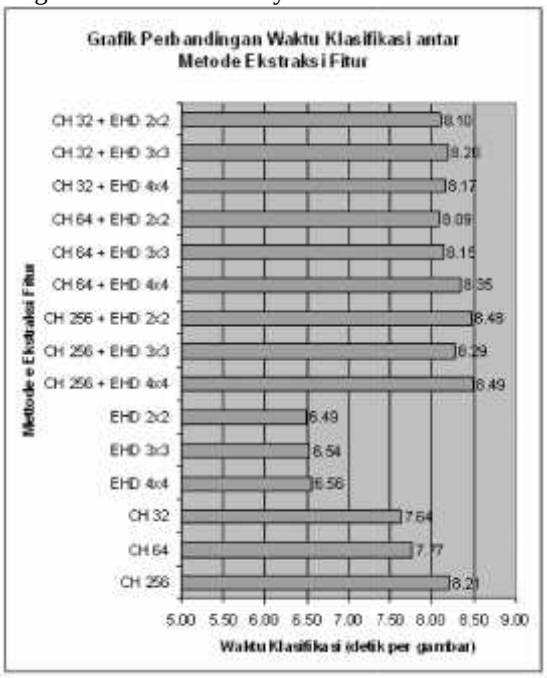

Image 4-6: Grafik perbandingan waktu klasifikasi

Waktu klasifikasi tersingkat adalah klasifikasi berdasarkan fitur tekstur yang diekstrak dengan metode *Edge Histogram Descriptor 2x2 Region* (EHD 2x2) yaitu selama 6.49 detik per image sedangkan yang membutuhkan waktu paling lama adalah klasifikasi berdasarkan warna dan tekstur yang diekstrak dengan metode *Color Histogram 256 bin* dan *Edge Histogram Descriptor 4x4 Region* (CH 256 + EHD 4x4) yaitu selama 8.49 detik per image.

5. Kesimpulan

Dari hasil perancangan sistem ini serta dari hasil uji coba yang telah dilakukan, dapat ditarik kesimpulan bahwa fitur image, metode ekstraksi fitur dan jumlah neighbor yang melakukan voting mempengaruhi tingkat akurasi klasifikasi image dengan metode *K-Nearest Neighbor*.

Pada bagian Fitur yang memberikan tingkat akurasi paling tinggi adalah fitur warna dan tekstur sedangkan metode ekstraksi fitur yang memberikan tingkat akurasi tertinggi adalah kombinasi *Color Histogram 256 bin* dan *Edge Histogram Descriptor* 2x2. Tingkat akurasi klasifikasi berfluktuasi tergantung jumlah *neighbor* yang melakukan *voting* namun cenderung menurun seiring bertambahnya jumlah *neighbor* yang melakukan *voting*.

Sedangkan klasifikasi yang membutuhkan waktu paling singkat adalah klasifikasi dengan fitur tekstur, disusul kemudian oleh klasifikasi dengan fitur warna dan yang membutuhkan waktu paling lama adalah klasifikasi berdasarkan fitur warna dan tekstur.

Daftar Pustaka:

[1] B. S. Manjunath et al, 2001, "Color and Texture Descriptors", IEEE Transcations on Circuits and Sistem for Video Technology
[2] CBIR:Features, http://www.ee.columbia.edu/~xlx/courses/vis-hw3/page2.html, didownload pada April 2007
[3] CBIR: Texture Features, 2007, www.cs.auckland.ac.nz/compsci708s1c/, didownload pada tanggal 11 April 2007
[4] Jiawei Han, Micheline Kamber, 2002, "Data Mining Concept and Techniques", Academic Press
[5] Kenneth R. Castleman, 1996, "Digital Image Processing", Prentice Hall
[6] Maher A. Sid Ahmed, 1995, "Image Processing: Theory, Algorithm and Architecture", McGrawHill
[7] Rafael C. Gonzales, Richard E. Woods, 2002, "Digital Image Processing", Pentice Hall
[8] Sundaram RMD, "Image Mining, Intricacies and Innovations", http://www.amrita.edu/cde/, didownload pada tanggal 11 April 2007
[9] Teknomo, Kardi. K-Nearest Neighbors Tutorial, 2006, http://people.revoledu.com/kardi/tutorial/KNN/, didownload pada tanggal 11 Desember 2006
[10] Uniform Quantization, 2007, http://www.cs.wpi.edu/~matt/courses/cs563/, didownload pada tanggal 11 April 2007
[11] William K. Pratt, 1991, "Digital Image Processing", Wiley-Interscience Publication