Jurnal Ilmiah Komputer dan Informatika (KOMPUTA)
Edisi. .. Volume. ... Bulan 20.. ISSN : 2089-9033

IMPLEMENTASI ALGORITMA K-MEANS PADA PENDETEKSIAN WARNA UNTUK MEMBANTU PENDERITA BUTA WARNA

Marcel Eka Putra

[1] Teknik Informatika – Universitas Komputer Indonesia
Jl. Dipatiukur 112-114 Bandung
E-mail : zellkoeman@gmail.com

ABSTRAK

K-Means merupakan salah satu algoritma *clustering* dimana setiap bagian pengelompokkannya diwakili oleh rata-rata dari anggota kelompoknya. K-Means mempunyai kelemahan pada inisialisasi titik pusat *cluster* yang bersifat random sehingga seringkali menyebabkan terjebaknya pada optimasi lokal sehingga hasil pengelompokannya tidak optimal. Pada kasus pengelompokkan warna sistem pendeteksian warna hal tersebut dapat berakibat pada kurang tepatnya hasil pendeteksian warna. Berdasarkan permasalahan tersebut maka diperlukan sebuah algoritma sebagai solusi yang dapat meningkatkan hasil *clustering* menuju optimasi global. *Fast Genetic K-Means Algorithm* (FGKA) merupakan sebuah algoritma *clustering* yang menggabungkan efisiensi algoritma K-Means dan kekuatan algoritma Genetika.

Dalam penelitian ini dilakukan sejumlah percobaan terhadap algoritma FGKA dengan melakukan kombinasi parameter inputan berupa jumlah kromosom, rasio mutasi, dan juga iterasi K-Means Operator (KMO). Kombinasi parameter FGKA yang optimal didapatkan pada penelitian ini sehingga dapat meningkatkan akurasi pendeteksian warna dengan waktu pengelompokkan warna yang paling efisien.

Kata kunci : *Fast Genetic K-Means Algorithm*, optimasi global, buta warna

1. PENDAHULUAN

K-Means [1] merupakan sebuah metode *clustering* yang bersifat efisien dan cepat dalam mengelompokkan data akan tetapi K-Means tidak terlalu efektif jika dilihat secara output dari proses pengelompokkannya. Hal ini dikarenakan K-Means hanya mengelompokkan data yang mengarah pada lokal optimum saja sehingga pengelompokkan data yang dihasilkan oleh K-Means seringkali kurang akurat. Berdasarkan hasil studi literatur K-Means dapat dioptimalkan dengan algoritma genetika yang merupakan algoritma yang baik dalam mendekati global optimum yang disebut dengan *Fast Genetic K-Means Algorithm* [3]. Dengan algoritma genetika hasil pengelompokkan dari K-Means akan menjadi lebih fit dan secara otomatis dapat meningkatkan akurasi pengelompokkan warna. Akan tetapi kekurangan dari algoritma FGKA adalah mahalnya biaya komputasi untuk mendekati kondisi konvergen.

Buta warna [2] merupakan salah satu kelainan atau penyakit pada indra penglihatan manusia yang menyebabkan ketidakmampuan penderitanya dalam mengenali setiap warna (buta warna total) ataupun warna tertentu saja (buta warna parsial). Pada kasus buta warna parsial, buta warna dapat dibagi menjadi beberapa golongan yaitu buta warna protanopia, buta warna deuteranopia, dan buta warna tritanopia dimana setiap golongan buta warna tersebut memiliki kekurangan dalam mengenali warna yang berbeda. Pada dunia medis sampai saat ini buta warna dikategorikan sebagai penyakit yang tidak dapat diobati karena buta warna merupakan penyakit yang disebabkan oleh faktor keturunan yaitu kekurangan satu atau lebih sel kerucut pada mata.

Pendeteksian warna merupakan suatu usaha dalam membantu penderita buta warna dalam mengenali warna dengan mengimplementasikan algoritma K-Means yang dapat dioptimalkan oleh algoritma genetika (FGKA) pada teknologi komputer. Aplikasi ini bekerja dengan cara melakukan pengelompokkan warna pada citra yang dipilih oleh pengguna penderita buta warna melalui sistem kemudian melakukan penandaan tepi oleh operator sobel.

1.1 Warna

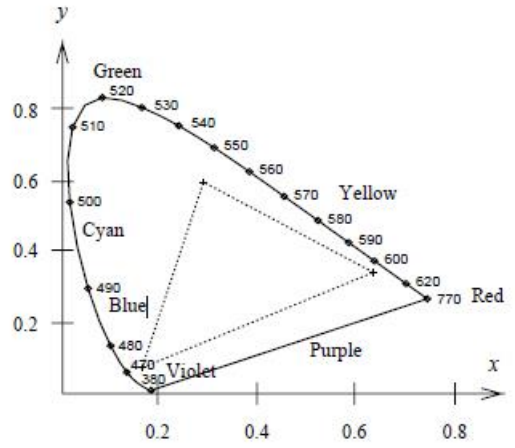
Warna [5] merupakan elemen terpenting pada sebuah objek berupa citra dimana elemen warna sangat dipengaruhi oleh intensitas cahaya. Warna sendiri dapat dilihat oleh indera penglihatan manusia dikarenakan adanya pantulan cahaya pada suatu permukaan benda dan dipengaruhi oleh pigmen yang terdapat pada permukaan benda tersebut.

Pada sebuah citra digital, warna dapat dibagi menjadi beberapa model warna yang digambarkan pada sistem koordinat berdimensi tiga. Model warna

yang umum dipakai adalah RGB (digunakan pada monitor), CMY (digunakan pada printer berwarna), YIQ (digunakan pada siaran televisi berwarna), HSV, CIEXYZ, CIE LUV, CIE Lab, dan Munsell. Model warna dibedakan berdasarkan sistem koordinat maupun gamut warna.

1.2 Model Warna RGB

Model warna RGB [5] digunakan oleh kebanyakan monitor berwarna. Monitor berwarna dapat menghasilkan 16 juta warna berbeda berdasarkan perpaduan warna dari ketiga chanel warna RGB yaitu merah, hijau dan biru. Gamut dari monitor berwarna RGB seringkali digambarkan sebagai unit kubus sehingga warna monitor sepenuhnya dikombinasikan oleh warna merah jenuh, kuning, hijau, cyan, biru, dan magenta. Pada sudut-sudut kubus terdapat warna hitam dan putih dimana secara diagonal menjadi degradasi warna putih menuju keabu-abuan hingga menjadi hitam. Sesuai dengan kromatisitas triad, gamut monitor dapat digambarkan sebagai segitiga kromatisitas tapal kuda yang dapat dilihat pada gambar 1.



Gambar 1 Kromatisasi warna

1.3 Algoritma K-Means

K-means [1] merupakan salah satu metode *clustering* non hirarki yang berusaha mempartisi data yang ada ke dalam bentuk satu atau lebih *cluster*. Metode ini mempartisi data ke dalam *cluster* sehingga data yang memiliki karakteristik yang sama dikelompokkan ke dalam satu *cluster* yang sama dan data yang mempunyai karateristik yang berbeda di kelompokan ke dalam *cluster* yang lain. Secara umum algoritma dasar dari K-Means adalah sebagai berikut :

1) Tentukan jumlah *cluster*
2) Alokasikan data ke dalam *cluster* secara random
3) Hitung *centroid*/rata-rata dari data yang ada di masing-masing *cluster*
4) Alokasikan masing-masing data ke *centroid*/rata-rata terdekat
5) Kembali ke Step 3, apabila masih ada data yang berpindah *cluster* atau apabila perubahan nilai *centroid* ada yang di atas nilai *threshold* yang ditentukan.

1.4 Distance Space

Distance space digunakan untuk menghitung jarak antara data dan *centroid*. Adapun persamaan yang dapat digunakan salah satunya yaitu *Euclidean Distance Space*. *Euclidean distance space* sering digunakan dalam perhitungan jarak, hal ini dikarenakan hasil yang diperoleh merupakan jarak terpendek antara dua titik yang diperhitungkan. Adapun persamaannya adalah sebagai berikut :

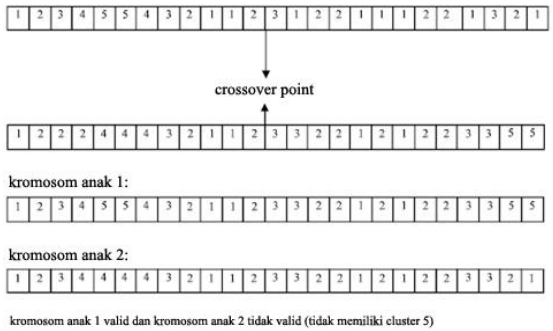
$$d_{ij} = \sqrt{\sum_{k=1}^{p} \left\{ x_{ik} - x_{jk} \right\}^2} \quad (1)$$

dimana,
d_{ij} = jarak *euclidean* antara data dan *centroid*
x_{ik} = data
x_{jk} = *centroid*/titik pusat *cluster*
p = dimensi data
k = jumlah *centroid*

1.5 Fast Genetic K-Means Algorithm (FGKA)

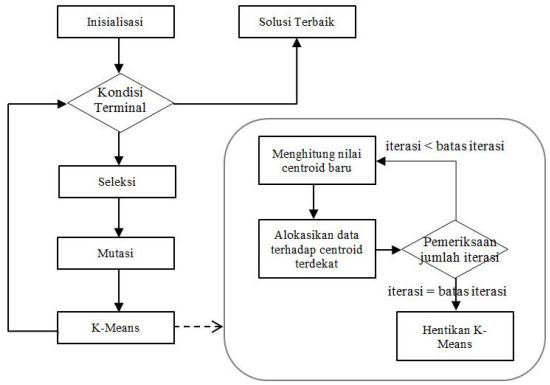
Fast Genetic K-Means Algorithm (FGKA) [3] merupakan pengembangan dari *Genetic K-means Algorithms* (GKA) yang diusulkan oleh Yi Lu pada tahun 2004. Algoritma ini selalu mampu menghasilkan konvergensi pada global optimal. FGKA mampu menghindari lokal optimal akan tetapi FGKA berjalan 20 kali lebih cepat dibandingkan GKA.

Algoritma GKA sangat baik dalam menemukan optimasi global akan tetapi membutuh waktu yang cukup besar dalam mencapai kondisi konvergensi akibat adanya operator genetika *crossover*. *Crossover* sangat mahal secara biaya komputasi untuk mendapatkan kromosom yang valid (kromosom dengan *cluster* tidak kosong) dan bahkan terkadang dapat menghasilkan kromosom yang tidak valid seperti yang diperlihatkan pada gambar 2. Kromosom 1 sebagai hasil *crossover* adalah contoh yang valid karena memiliki setiap *cluster* pada kromosomnya tetapi kromosom 2 adalah contoh yang tidak valid karena *cluster* 5 tidak memiliki anggota pada kromosom tersebut.



Gambar 2 Operator crossover pada GKA

Pada Gambar 3 diilustrasikan diagram alir algoritma FGKA. Algoritma dimulai dari fase inisialisasi, dimana pada fase ini dibangkitkan populasi awal P_0. Populasi dalam generasi berikutnya P_{i+1} didapatkan menggunakan operator genetika dari populasi sebelumnya P_i. Evolusi akan berhenti jika kondisi akhir telah dicapai yaitu pada saat setiap kromosom memiliki angka *fitness* yang konvergen pada suatu generasi. Operator genetika yang digunakan dalam algoritma FGKA ini adalah : seleksi, mutasi dan operator K-means.



Gambar 3 Diagram alir FGKA

1.5.1 Inisialisasi

Tahap inisialisasi [3] adalah membangkitkan populasi awal P0 yang berisi Z solusi, dimana Z adalah parameter yang ditentukan oleh pengguna. Masing-masing gen dalam kromosom menunjukkan nomor *cluster* yang dipilih secara *random* dengan distribusi uniform dalam jangkauan {1, ..., K}. Dari inisialisasi awal ini dimungkinkan didapatkannya string yang tidak legal. Untuk populasi awal string yang tidak legal dihindari, artinya dalam populasi awal harus didapatkan string yang legal. Apabila string yang tidak legal dapat dimasukkan dalam FGKA, maka sebagian besar solusi yang tidak legal dengan memberikan nilai TWCV yang tinggi dan nilai *fitness* yang rendah sehingga sebuah solusi yang tidak legal akan mempunyai probabilitas yang rendah untuk bertahan. Kompleksitas dari proses inisialisasi awal ini adalah O(Z).

1.5.2 Nilai Fitness

Permasalahan *clustering* [3] terdiri dari N gen yang merepresentasikan N data(obyek). Tujuan dari algoritma FGKA adalah membagi N data ke dalam K *cluster*, dimana *cluster* yang baik adalah *cluster* yang memiliki anggota dengan jarak total yang kecil terhadap masing-masing centroid *cluster*. Itu berarti *cluster* yang memiliki nilai *fitness* yang tinggi akan memiliki nilai *Total Within-Cluster Variation* (TWCV) yang rendah dan disebut sebagai masalah minimasi. Untuk mengkonversi masalah minimasi menjadi masalah maksimasi perlu dilakukan *inverse* terhadap TWCV dalam mencari nilai *fitness* suatu cluster dengan rumusan:

$$f(i) = 1/\sum_{j=1}^{n} \begin{matrix}(\text{CON} + \text{difference of allele j in chromosome} \\ \text{i from the corresponding cluster centre})\end{matrix} \quad (2)$$

Dimana CON adalah konstanta yang bernilai 1. Konstanta ini disertakan ke dalam persamaan untuk menangani kasus ketika nilai dari TWCV adalah nol. Jika TWCV bernilai nol dan tidak adanya konstanta CON maka nilai *fitness* akan menjadi ∞ (tidak terhingga).

1.5.3 Seleksi

Seleksi [3] dilakukan dengan maksud untuk menemukan pusat *cluster* global dengan mereduksi nilai dari TWCV setiap kromosom. Pada seleksi dilakukan operator elitism yang merupakan operator yang membiarkan kromosom dengan nilai *fitness* terbaik selalu dipertahankan.

Kromosom dengan nilai *fitness* terbaik sebagai hasil seleksi oleh operator elitism akan dibandingkan dengan nilai *fitness* kromosom-kromosom generasi berikutnya yang telah mengalami mutasi. Kromosom hasil pengkopian ini akan diurutkan ke dalam serangkaian kromosom pada generasi berikutnya sehingga kromosom dengan nilai *fitness* terburuk pada generasi berikutnya tidak akan bertahan. Dengan operator elitism ini maka populasi di dalam generasi terbaru akan memiliki serangkaian solusi Z kromosom yang merupakan kandidat-kandidat kromosom dengan nilai *fitness* terbaik.

1.5.4 Mutasi

Mutasi [3] digunakan untuk menghindari optimum lokal dan membuat pusat *cluster* untuk mendekati optimum global secara menyebar. Karena sifat inisialisasi yang acak, pusat *cluster* tidak akan tepat pada tahap awal sehingga dilakukan mutasi pada setiap kromosom Z selain kromosom terbaik pada setiap generasi tertentu sesuai dengan rasio mutasi yang ditetapkan. Untuk melakukan mutasi perlu dilakukannya pemeriksaan probabilitas dari suatu data X_{ij} pada kromosom Z_i yang didapatkan secara random.

$$D_m(X_{ij}) = (\text{abs}\ (X_{ij} - C_j) - \text{tot_min})$$
$$P_m(X_{ij}) = 0 \text{ if } D_m = 0$$
$$P_m(X_{ij}) = 1 \text{ otherwise} \quad (3)$$

Dimana tot_min merupakan jarak terkecil antara alel yang berada dalam *cluster* data Xij dan pusat *cluster* data Xij.

Jika Pm adalah 1 maka nilai dari alel terpilih akan dipetakan terhadap pusat *cluster* yang secara acak dipilih. Kemudian jika Pm adalah 0 maka kromosom tersebut tidak perlu dilakukan mutasi.

Mutasi ini didefinisikan sebagai berikut:

1) Xij dipilih secara random, hal ini bertujuan untuk mencapai optimal pada wilayah global.
2) Probabilitas dari Xij akan besar ketika jarak antara alel tersebut dengan pusat cluster adalah besar. Ini dimaksudkan agar pusat cluster baru yang terbentuk akibat mutasi tidak berjarak terlalu berdekatan atau bahkan merupakan nilai dari pusat cluster yang sudah eksis.

1.6 Operator Sobel

Operator Sobel [6] merupakan sebuah teknik deteksi tepi yang sederhana dan memiliki tingkat komputasi yang cepat dan akurat. Pada umumnya operator ini digunakan untuk citra *grayscale*. Operator Sobel dapat digambarkan dengan dua buah matriks berukuran 3 x 3 seperti pada gambar 4.

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

G_x G_y

Gambar 4 Matriks operator sobel

Matriks operator sobel dapat merespon tepian maksimal hingga 45º. Kedua matriks tersebut dapat diterapkan pada citra secara terpisah, baik untuk mendapatkan tepian horizontal maupun tepian vertikal. Kedua matriks ini juga dapat dikombinasikan untuk mendapatkan hasil *gradient* gabungan dengan menggunakan persamaan:

$$|G| = \sqrt{G_x^2 + G_y^2} \quad (4)$$

atau dengan persamaan,

$$|G| = |G_x| + |G_y| \quad (5)$$

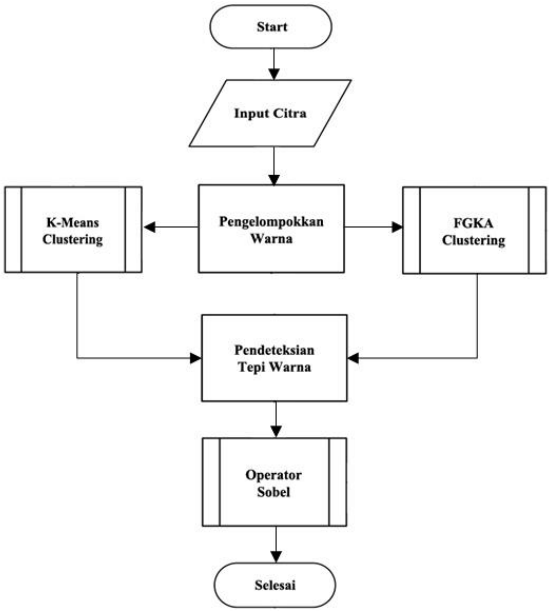
2. ISI PENELITIAN

2.1 Sistem Pendeteksian Warna

Sistem yang akan dibangun adalah sebuah sistem pendeteksian warna yang bertujuan untuk membantu penderita buta warna untuk dapat memilah warna dari suatu objek citra yang didapatkan dengan melakukan *capture* foto melalui kamera perangkat komputer ataupun melakukan *browse* terhadap citra yang telah eksis pada direktori hardisk komputer.

Citra inputan yang telah didapatkan oleh sistem komputer kemudian akan melalui tahap *pre-proccessing* dengan melakukan penskalaan citra menjadi ukuran 400 x 350 piksel. Tahap ini diharapkan dapat mereduksi beban proses sistem dalam melakukan pengelompokkan warna yang dilakukan pada tahap selanjutnya.

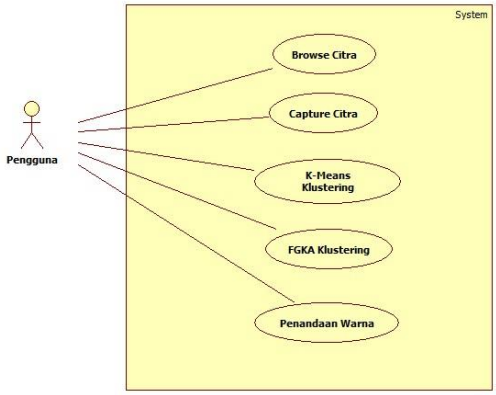
Setelah melakukan penskalaan ukuran citra, kemudian sistem mengelompokkan warna citra sekaligus menurunkan dimensi warna citra menjadi sejumlah titik pusat *cluster* K yang didefinisikan oleh pengguna sistem. Setelah warna citra berhasil dikelompokkan ke dalam setiap titik pusat *cluster*, maka setiap nilai titik pusat *cluster* adalah warna-warna terdeteksi pada citra yang dapat dipilih oleh pengguna sistem untuk kemudian dilakukan pendeteksian tepi pada setiap anggota *cluster* terpilih. Pendeteksian tepi dilakukan untuk memperjelas informasi warna yaitu batas-batas tepian objek berwarna yang dibutuhkan oleh pengguna sistem. Gambaran umum sistem pendeteksian warna dapat dilihat pada gambar 5.



Gambar 5 Gambaran umum sistem

2.2 Use Case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat oleh sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. Use case diagram sistem pendeteksian warna dapat dilihat pada gambar 6.



Gambar 6 Use Case Diagram

2.3 Hasil Pendeteksian Sistem

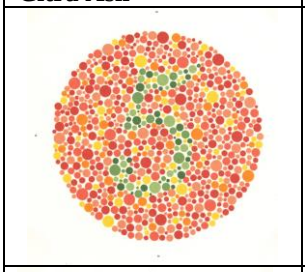
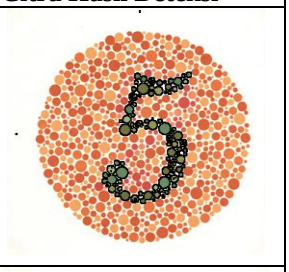
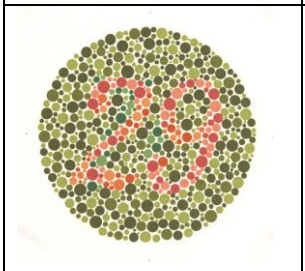
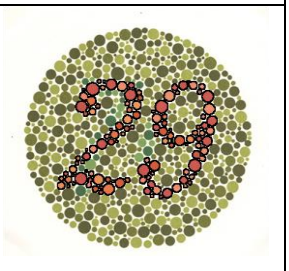
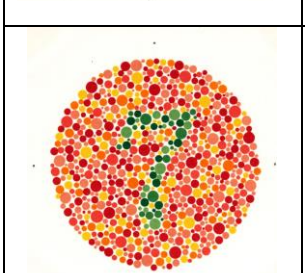
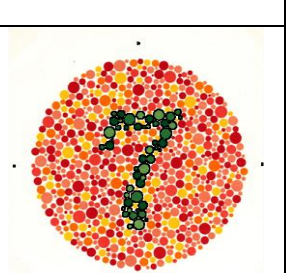
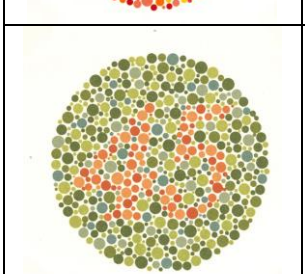
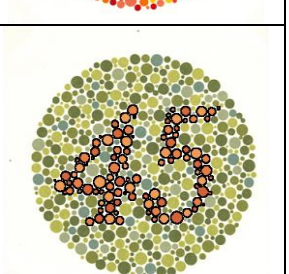
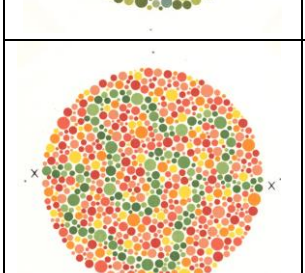
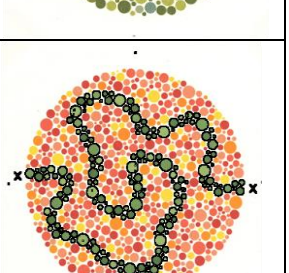
Penyebab buta warna adalah kurangnya atau tidak adanya pigmen pada sel kerucut di lapisan retina mata. Fungsi mata khususnya dari pigmen ini sendirilah yang memungkinkan orang dapat mengenali berbagai macam warna. Penyakit buta warna yang paling banyak terjadi dan umum adalah kekurangan pigmen merah dan juga hijau. Akibat yang ditimbulkan dari seseorang yang kekurangan pigmen tersebut maka seseorang akan mengalami kesulitan dalam hal mengidentifikasi dan mengenali akan warna merah dan hijau atau campuran keduanya.

Secara umum penderita buta warna memiliki kesulitan dalam melihat angka pada saat tes buta warna ishihara. Ishihara sendiri merupakan sebuah *plate* yang di uji cobakan terhadap kontestan uji buta warna pada saat tes kesehatan untuk memasuki dunia kerja, perkuliahan, pembuatan SIM dan lain sebagainya. Tes ini pertama kali dipublikasi pada tahun 1917 di Jepang. Sejak saat itu, tes ini terus digunakan di seluruh dunia, sampai sekarang.

Tes buta warna Ishihara terdiri dari lembaran yang didalamnya terdapat titik-titik dengan berbagai warna dan ukuran. Titik berwarna tersebut disusun sehingga membentuk lingkaran. Warna titik itu dibuat sedemikian rupa sehingga orang buta warna tidak akan melihat perbedaan warna seperti yang dilihat orang normal (*pseudo-isochromaticism*)

Dengan menggunakan sistem pendeteksian warna maka penderita buta warna dapat berlatih secara mandiri dalam melihat *plate* ishihara. Beberapa hasil pendeteksian warna oleh sistem pendeteksian warna terhadap *plate* ishihara dapat dilihat pada tabel 1.

Tabel 1 Pendeteksian Warna Pada Plate Ishihara

Citra Asli	Citra Hasil Deteksi
	
	
	
	
	

2.4 Pengujian Simulasi Sistem

Pengujian pendeteksian warna dilakukan dengan menggunakan ketetapan jumlah *centroid* sebanyak 7 *centroid*. Kemudian akan dilakukan inputan parameter jumlah kromosom, rasio mutasi, dan jumlah iterasi K-Means Operator (KMO) yang berbeda-beda terhadap algoritma FGKA. Hal ini diterapkan untuk mengetahui kombinasi parameter yang paling tepat sehingga dapat meningkatkan optimasi pengelompokkan warna yang dilakukan oleh algoritma FGKA.

Jumlah kromosom merupakan banyaknya kromosom Z pada setiap generasi. Jumlah kromosom akan berpengaruh terhadap waktu yang dibutuhkan FGKA dalam mencapai kondisi konvergen sehingga semakin banyak jumlah kromosom maka akan semakin lama FGKA dalam mencapai kondisi konvergen. Akan tetapi dengan semakin banyaknya jumlah kromosom, FGKA akan memiliki kemungkinan yang lebih besar dalam mencapai kondisi optimasi global.

Rasio mutasi merupakan parameter yang menentukan pada setiap kelipatan generasi tertentu akan dilakukan mutasi. Semakin sering mutasi dilakukan maka akan semakin memperbesar kemungkinan kromosom untuk keluar dari optimasi lokal menuju optimasi global.

Iterasi KMO merupakan parameter yang menyatakan jumlah iterasi clustering algoritma K-Means pada FGKA dalam melakukan pengelompokkan warna. Semakin besar angka iterasi KMO maka akan semakin cepat konvergensi dicapai, akan tetapi semakin besar pula waktu yang dibutuhkan FGKA dalam mencapai kondisi optimal.

Pengujian terhadap sistem pendeteksian warna untuk membantu penderita buta warna dilakukan dengan inputan citra berukuran 400x350 piksel seperti pada gambar 7.



Gambar 7 Citra inputan pengujian sistem

Rencana pengujian sistem terhadap beberapa kombinasi inputan parameter FGKA yang dilakukan dapat dilihat pada tabel 2.

Tabel 2 Rencana pengujian sistem

Kasus Uji	Jumlah Kromosom	Rasio Mutasi	Iterasi KMO
Kasus Uji 1	4	3	2
Kasus Uji 2			3
Kasus Uji 3			4
Kasus Uji 4			5
Kasus Uji 5		5	2
Kasus Uji 6			3
Kasus Uji 7			4
Kasus Uji 8			5
Kasus Uji 9		7	2
Kasus Uji 10			3
Kasus Uji 11			4
Kasus Uji 12			5
Kasus Uji 13	6	3	2
Kasus Uji 14			3
Kasus Uji 15			4
Kasus Uji 16			5
Kasus Uji 17		5	2
Kasus Uji 18			3
Kasus Uji 19			4
Kasus Uji 20			5
Kasus Uji 21		7	2
Kasus Uji 22			3
Kasus Uji 23			4
Kasus Uji 24			5

Hasil pengujian simulasi terhadap sistem dapat dilihat pada tabel 3. Berdasarkan pengujian tersebut didapatkan nilai *fitness* dan waktu pengelompokkan warna citra oleh algoritma FGKA. Sistem dieksekusi pada perangkat komputer dengan spesifikasi Processor : Intel Pentium Dual Core P6100, 2.00Ghz. Memory : 5GB.

Tabel 3 Hasil pengujian sistem

Kasus Uji	Nilai Fitness	Waktu
Kasus Uji 1	8.487994×10^{-9}	35.5 detik
Kasus Uji 2	8.488081×10^{-9}	1 menit 38.3 detik
Kasus Uji 3	8.487951×10^{-9}	48.1 detik
Kasus Uji 4	8.487951×10^{-9}	1 menit 22.3 detik
Kasus Uji 5	8.487953×10^{-9}	1 menit 42.7 detik
Kasus Uji 6	8.487962×10^{-9}	1 menit 9.5 detik
Kasus Uji 7	8.487962×10^{-9}	1 menit 29.9 detik
Kasus Uji 8	8.487952×10^{-9}	1 menit 32.2 detik
Kasus Uji 9	8.487156×10^{-9}	1 menit 8.6 detik
Kasus Uji 10	8.488081×10^{-9}	1 menit 52.2 detik

Kasus Uji 11	8.487184 x 10^{-9}	1 menit 28.8 detik
Kasus Uji 12	8.487952 x 10^{-9}	1 menit 52.2 detik
Kasus Uji 13	8.488089 x 10^{-9}	5 menit 53.1 detik
Kasus Uji 14	8.488089 x 10^{-9}	1 menit 47 detik
Kasus Uji 15	8.488089 x 10^{-9}	2 menit 20.2 detik
Kasus Uji 16	8.487241 x 10^{-9}	3 menit 8.8 detik
Kasus Uji 17	8.487951 x 10^{-9}	1 menit 46.6 detik
Kasus Uji 18	8.488087 x 10^{-9}	3 menit 6.9 detik
Kasus Uji 19	8.488088 x 10^{-9}	1 menit 56.8 detik
Kasus Uji 20	8.488089 x 10^{-9}	3 menit 18.6 detik
Kasus Uji 21	8.488089 x 10^{-9}	1 menit 46.3 detik
Kasus Uji 22	8.488089 x 10^{-9}	2 menit 39.7 detik
Kasus Uji 23	8.488089 x 10^{-9}	2 menit 48.6 detik
Kasus uji 24	8.488089 x 10^{-9}	2 menit 30.3 detik

Sebagai hasil pengujian keseluruhan bahwa algoritma FGKA mencapai optimasi global dengan nilai *fitness* tertinggi yang dapat dicapai hingga kondisi konvergen yaitu 8.488089 x 10^{-9}. Pada pengelompokkan oleh parameter jumlah kromosom 4, FGKA belum dapat konvergen mencapai kondisi optimasi global. Sedangkan pada jumlah kromosom 6 didapatkan kondisi konvergen pada optimasi global pada rasio mutasi 3, 5, dan 7 namun tidak selalu mutlak menuju optimasi global. Angka rasio mutasi yang dapat meningkatkan performansi dengan selalu mencapai kondisi global optima adalah pada rasio mutasi 7 sehingga dapat disimpulkan bahwa parameter jumlah kromosom 6 dan rasio mutasi 7 merupakan parameter inputan yang baik.

Dalam mencapai optimasi algoritma FGKA perlu adanya perbandingan efisiensi terhadap kandidat-kandidat kombinasi parameter terbaik. Setelah melakukan perbandingan efisiensi berdasarkan waktu pengelompokkan warna pada kasus uji ke-21, kasus uji ke-22, kasus uji ke-23, dan kasus uji ke-24 yang merupakan kombinasi parameter inputan iterasi K-Means didapatkan bahwa pada iterasi K-Means bernilai 2 menghasilkan waktu pengelompokkan yang paling efisien.

3. PENUTUP

Berdasarkan hasil pengujian yang telah dilakukan ditemukan bahwa semakin besar jumlah kromosom maka semakin besar pula kemungkinan FGKA dalam mencapai konvergensi pada global optima. Namun perlu adanya pertimbangan pada nilai rasio mutasi, semakin rendah nilai rasio mutasi maka semakin sulit FGKA dalam mencapai kondisi konvergen. Iterasi K-Means menjadi penyebab mahalnya biaya komputasi FGKA dalam mencapai konvergensi.

Algoritma FGKA dapat mencapai kondisi pengelompokkan warna yang optimal dengan kombinasi parameter inputan jumlah kromosom sebanyak 6, rasio mutasi bernilai 7 dan iterasi K-Means dilakukan sebanyak 2 iterasi. Kombinasi parameter inputan tersebut dapat selalu menghasilkan pengelompokkan warna dengan kondisi global optima dan memiliki waktu pengelompokkan selama 1 menit 46.3 detik.

Untuk lebih meningkatkan kinerja dari sistem yang dibuat, maka diusulkan beberapa saran sebagai berikut :

1. Perlu adanya otomatisasi sistem dalam menentukan jumlah centroid berdasarkan kompleksitas warna yang dimiliki oleh citra inputan ke dalam sistem sehingga keuntungan yang didapatkan adalah untuk citra yang memiliki kompleksitas warna rendah akan memiliki waktu pengelompokkan warna yang lebih efisien, dan untuk citra yang memiliki kompleksitas warna tinggi akan memiliki akurasi pendeteksian warna yang lebih optimal.
2. Perlu adanya usaha dalam meningkatkan efisiensi pengelompokkan warna oleh operator K-Means pada algoritma FGKA. Dengan usaha tersebut maka waktu pengelompokkan warna dalam setiap iterasi operator K-Means yang dilakukan oleh setiap kromosom pada suatu generasi FGKA dapat diminimalkan.
3. Untuk kasus pendeteksian warna secara real-time seperti pada sistem berbasis mobile ataupun web, pengelompokkan warna tanpa memperhatikan optimasi nilai fitness lebih disarankan karena kebutuhan efisiensi waktu dan memori pada kasus tersebut lebih diutamakan.

DAFTAR PUSTAKA

[1] Prasetyo, E. (2013). Data Mining : Konsep Dan Aplikasi Menggunakan Matlab. Yogyakarta: Andi Publisher.
[2] Sunar Prasetyono Dwi (2013). Tes Buta Warna untuk Segala Tujuan. Yogyakarta: Saufa.

[3] Yi Lu, Shiyong Lu, Farshad Fotouhi, Youping Deng, Susan Brown , "FGKA: A Fast Genetic K-means Algorithm", Discourse Processes, 2004.
[4] Suyanto, ST., Msc. (2008). Evolutionary Computation - Komputasi Berbasis Evolusi dan Genetika. Bandung: Informatika.
[5] Supriyono, R. (2010). Desain Komunikasi Visual. Yogyakarta: Andi Publisher.
[6] Hermawati, F. A. (2013). Pengolahan Citra Digital. Yogyakarta: Andi Publisher.
[7] K. Venkatalakshmi, P. Anisha Praisy, R. Maragathavalli, S. Mercy Shalinie (2007), Multispectral Image Clustering Using Enhanced Genetic k-Means Algorithm, Information Technology Journal 6 (4) : 554-560.
[8] Wiharto, Y.S. Palgunadi, Muh Aziz Nugroho (2013), Analisis Penggunaan Algoritma Genetika untuk Perbaikan Jaringan Syaraf Tiruan Radial Basis Function, ISSN: 2089-9815.
[9] Frery, A. C.; Melo, C. A. S. & Fernandes, R. C. (13 October 2000). Web-based Interactive Dynamics for Color Models Learning, Color Research and Application 25 (6): 435–441.