Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi
Vol. 1, No. 2, Juli 2005

ISSN 0216 - 0544

SISTEM IDENTIFIKASI OBYEK BERBASIS *FEATURE*

C.Very Angkoso

Jurusan Teknik Informatika
Universitas Trunojoyo

Email: cvery@trunojoyo.ac.id

ABSTRAK

Pemrosesan citra digital memerlukan dukungan perangkat yang canggih, sehingga mampu melakukan penghitungan dalam jumlah besar secara cepat, apalagi jika pemrosesan citra digital dilakukan untuk aplikasi yang *real time*. Namun dengan berkembangnya teknologi komputer sekarang ini, pemrosesan citra digital bahkan dapat dilakukan pada komputer personal atau PC. Dalam jurnal penelitian ini dibahas suatu aplikasi pemrosesan citra digital untuk pendeteksian obyek, dengan metode berbasis *feature*. Citra yang diperoleh dari stream sebuah *frame grabber* atau dari kamera digital, mengandung informasi tentang suatu obyek. Dalam penelitian ini obyek akan didefinisikan atau dikenali dengan memanfaatkan *feature* yang dimiliki oleh obyek tersebut. *Feature* ini meliputi warna, luas daerah pada citra, centroid obyek, dan orientasi obyek. .

Kata kunci : pemrosesan citra digital, pengenalan obyek berbasis *feature* obyek, *adaptive thresholding*

1. PENDAHULUAN

Dengan penglihatan, manusia dapat bernavigasi dan mendapatkan informasai perihal lingkungan sekitar. Mengenali wajah seseorang, mengidentifikasi sebuah obyek, memahami mood atau emosi seseorang lewat ekspresi wajah hanyalah sebagian kecil dari tugas sehari-hari penglihatan mata manusia. Walaupun seolah-olah tugas-tugas tersebut kita lakukan dengan usaha yang tidak seberapa berat, analisa dan simulasi dari proses-proses tersebut sangatlah rumit dan kompleks. Hanya dengan bantuan komputer dengan kemampuan dan kecepatan proses cukup tinggi, simulasi tentang sistem visual dimungkinkan untuk dilakukan.

Secara garis besar pemrosesan citra digital akan melalui tahap-tahap sebagai berikut :

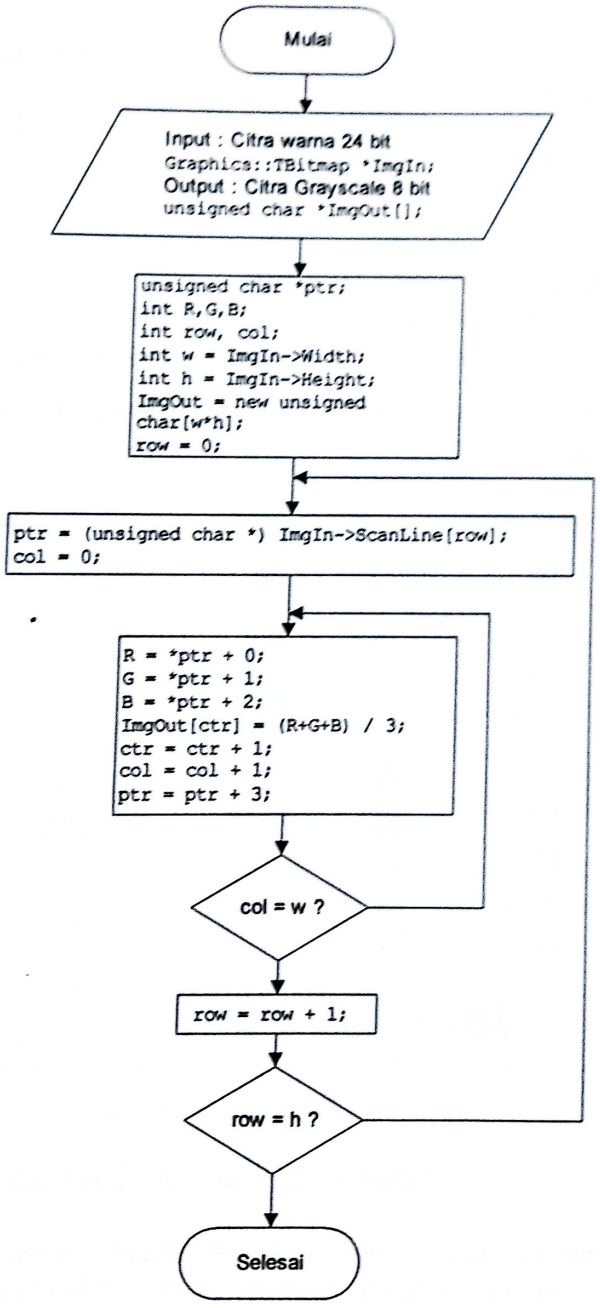
- Akuisi citra (dengan kamera digital, atau stream dari *frame grabber*)
- Proses pengkondisian citra, berupa proses *filtering* untuk menghilangkan *noise* dan memperbaiki kualitas citra.
- Proses *thresholding* untuk memisahkan obyek yang dianalisa dalam citra dari lingkungan obyek dalam citra.
- Proses *labeling* untuk mendapatkan informasi mengenai obyek dalam citra, informasi ini mencakup posisi obyek dalam citra, jumlah obyek dalam citra serta area obyek dalam citra.
- Proses pengenalan obyek yang telah didapat dalam proses *labeling*

Berikut ini penerapan tahap-tahap di atas serta hal yang melatar belakangi pemilihan satu metode tertentu dalam suatu tahap berdasarkan pertimbangan efisiensi algoritma dan kualitas hasil yang diperoleh.Dalam penelitian ini, citra diperoleh *stream* suatu *frame grabber* yang telah disimpan dalam bentuk *file* citra digital bertipe citra warna 24-bit dengan resolusi 320 x 240. Pemilihan resolusi yang optimal akan sangat berpengaruh pada kecepatan pemrosesan.

2. PENGKONDISIAN CITRA

2.1 *Color Conversion*

Pengubahan warna pada titik-titik gambar pada suatu citra diperlukan sebagai pengkondisian citra untuk dapat diolah lebih lanjut. Citra yang didapat dari *frame grabber* mempunyai tingkat kedalaman warna 24-bit, dengan segmentasi warna per-1 *byte*. Kombinasi dari tiga komponen warna menghasilkan variasi warna sejumlah $2^{kedalaman} = 2^{24} = 16777216$ warna. Setelah dari *frame grabber* citra dikonversikan menjadi bentuk yang lebih sederhana. Dari kedalaman warna 24 bit diubah menjadi kedalaman warna 8 bit, hal ini berfungsi untuk mempermudah proses pengolahan citra. Citra dengan kedalaman warna 8 bit mempunyai dua tipe yang berbeda. Yang pertama adalah citra kedalaman warna 8 bit bertipe gambar berwarna, tipe seperti ini akan mempunyai karakteristik citra dengan jumlah warna 256. Tiap kenaikan nilai *pixel* menandakan warna yang berubah secara gradual. Jenis yang kedua adalah berupa citra 8 bit dengan tipe *grayscale*. Pada jenis yang kedua ini masing-masing intensitas *pixel* menandakan tingkatan keabu-abuan. Sebagaimana terlihat pada Gambar 2A. Dalam konvensi tipe atau format citra digital, 2 tipe citra di atas hanya dibedakan dengan data *header* dalam file citra yang akan memberikan informasi tipe citra. *Database* berupa *look_up table* warna-warna menurut tingkat intensitas diberikan di bagian akhir dari sebuah *file* citra. Tabel *look_up* sendiri merupakan kombinasi dari data RGB 24 bit.



***Gambar 1.** Fungsi Graying, color conversion,untuk mengkonversi citra warna ke citra grayscale*

2.2 *Adaptive Threshold*

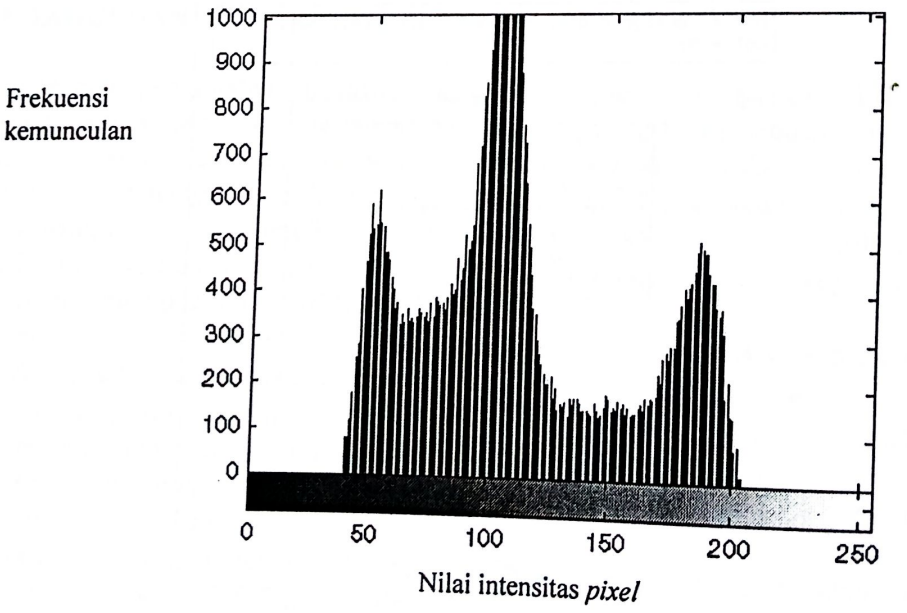
Thresholding diperlukan untuk memisahkan obyek utama pada citra dari obyek lain pada citra. Dengan *thresholding* diharapkan citra hanya mempunyai dua kondisi intensitas nilai *pixel*, intensitas *high* menandakan obyek dan intensitas *low* untuk menandakan yang bukan obyek. Proses *thresholding* pada umumnya dilakukan dengan pemfilteran citra dan pemberian suatu nilai ambang intensitas. Jika nilai intensitas suatu *pixel* berada di bawah nilai ambang tersebut maka pada keluaran citra hasil *thresholding* titik yang bersangkutan akan diubah menjadi *low*. Berlaku sebaliknya juga untuk nilai intensitas *pixel* yang berada di atas nilai ambang. Pemberian nilai ambang yang tepat sangat penting peranannya untuk menghasilkan penafsiran citra yang benar.

Sistem *adaptive thresholding* merupakan proses *thresholding* dengan nilai ambang mengacu kepada kondisi citra yang bersangkutan. Citra yang menginginkan informasi suatu obyek cenderung akan menempatkan obyek pada proporsi yang lebih banyak dibandingkan dengan yang bukan obyek, untuk mendapatkan detail. Atas asumsi inilah metode *adaptive thresholding* dipakai, yaitu memanfaatkan histogram citra sebagai sumber informasi proporsi jumlah *pixel* yang merupakan bagian dari obyek yang diinginkan. Dengan informasi dari histogram citra, nilai intensitas ambang dapat diketahui. Nilai ambang adalah nilai batas intensitas dimana suatu kelompok intensitas obyek dipisahkan dengan kelompak intensitas bukan obyek. Dalam persamaan matematis, operasi *thresholding* dapat dinyatakan sebagai berikut :

$$g(x,y) = \begin{cases} G_o, f(x,y) > T & .....(2.1) \\ G_b, f(x,y) \le T & .....(2.2) \end{cases}$$

dengan:

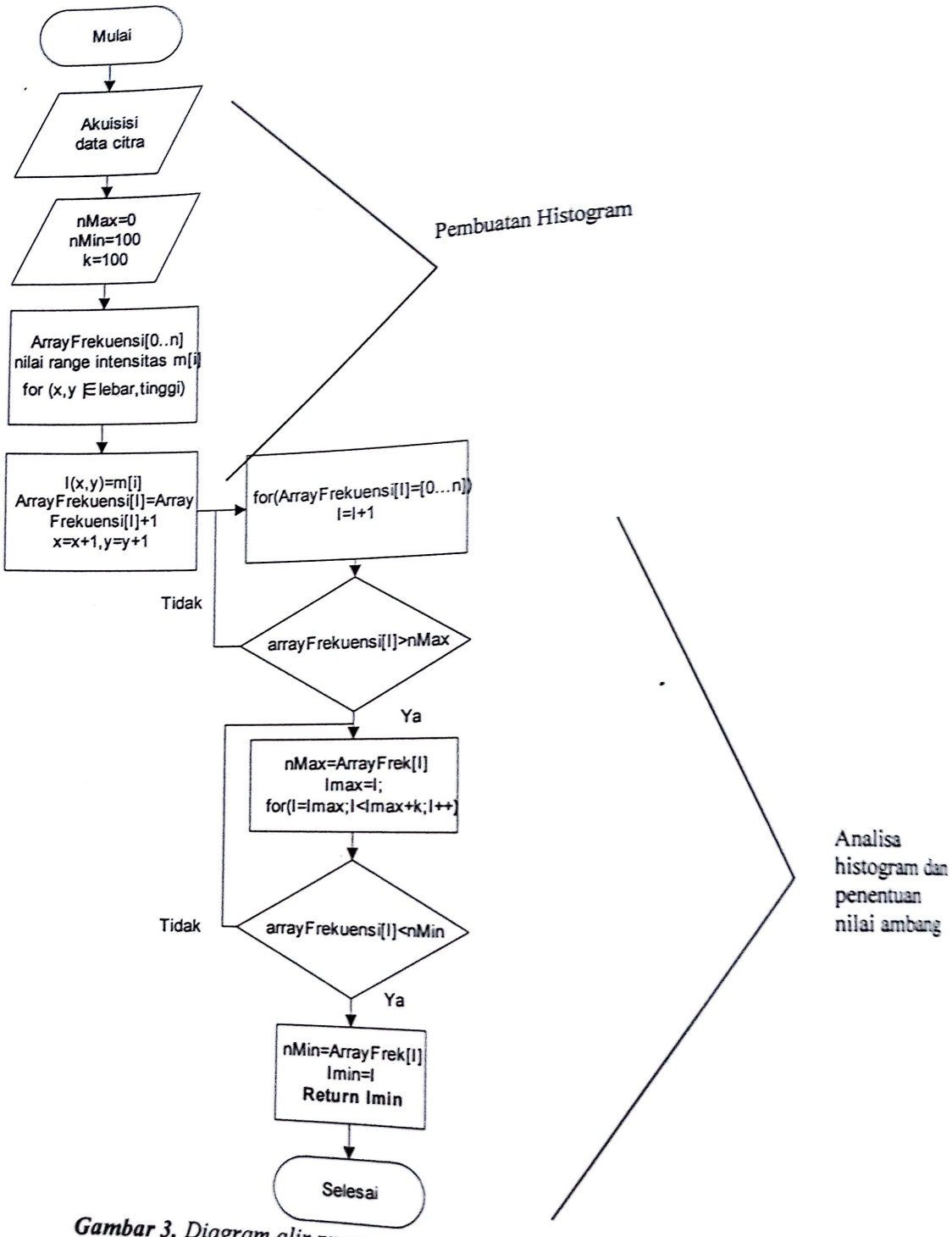
f(x,y) : nilai *greylevel* citra sebenarnya
g(x,y) : nilai *greylevel* citra hasil operasi *threshold*
G_o : nilai *greylevel* obyek citra setelah operasi *threshold*
G_b : nilai *greylevel* latar belakang citra setelah operasi *threshold*
T : nilai *threshold*



Gambar 2. Histogram sebuah citra

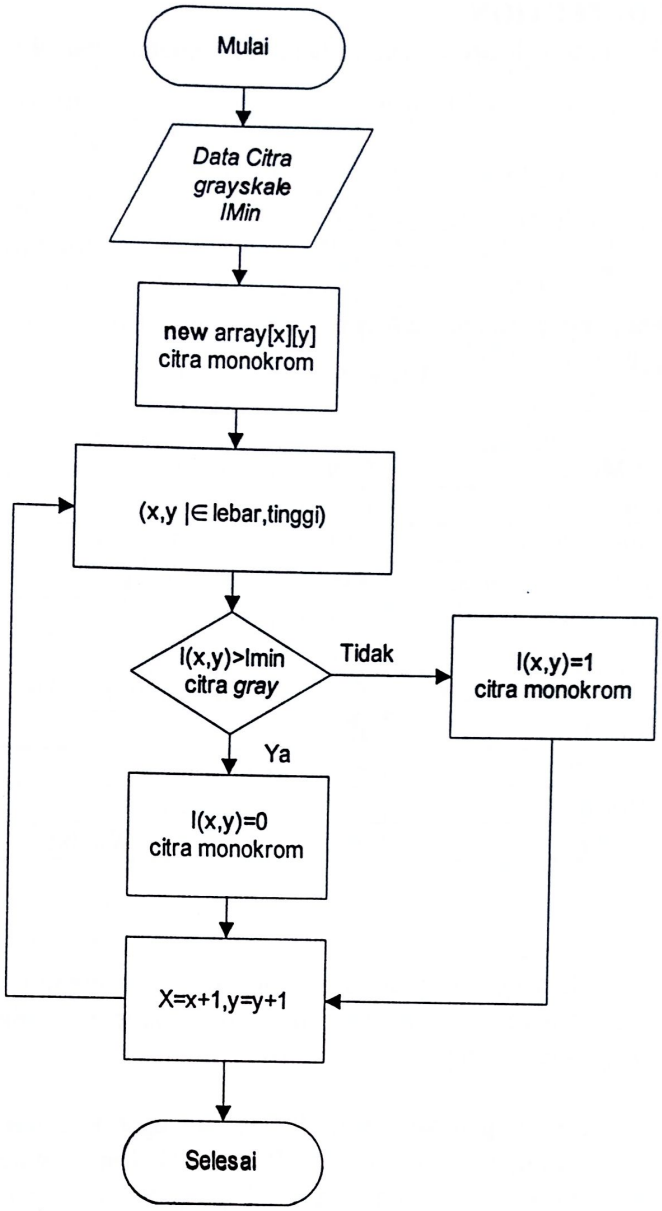
Dengan informasi dari histogram di atas dapat ditentukan nilai batas intensitas antara kerumunan intensitas tertentu yang menandakan obyek dan kerumunan yang lain yang menandakan lingkungan. Cara kerja proses *adaptive thresholding* meliputi tiga prosedur utama yaitu prosedur pembuatan histogram citra, dilanjutkan dengan proses pembacaan dan analisa histogram untuk

mendapatkan nilai ambang yang cocok untuk citra yang bersangkutan, dan yang terakhir proses *thresholding*, dengan acuan nilai ambang yang telah didapatkan pada prosedur sebelumnya. Pada prosedur pembuatan histogram citra dirunut nilai intensitas *pixel*-nya satu persatu, dan kemudian nilai yang diperoleh dimasukkan dalam suatu matriks frekuensi nilai intensitas. Setiap kali menemukan nilai intensitas tertentu algoritma ini akan menambahkan nilai 1 pada kolom matriks yang bersesuaian. Setelah didapatkan histogram, analisa dilakukan dengan mencari nilai frekuensi maksimum untuk setiap nilai intensitas *pixel* yang ada. Nilai ini dijadikan patokan sebagai kisaran nilai *pixel* lingkungan terhadap obyek yang diamati. Selanjutnya dilakukan pencarian nilai frekuensi minimum pada rentang nilai intensitas di atas nilai intensitas dengan frekuensi paling tinggi. Rentang yang diberikan mencapai sedikit dibawah nilai intensitas maksimum.

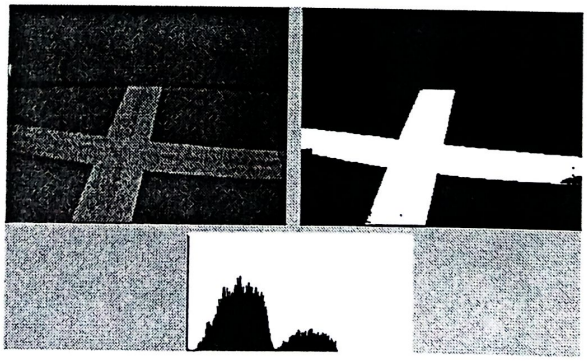


***Gambar 3.** Diagram alir proses pembuatan histogram dan penentuan nilai ambang*

Dari proses dalam diagram alir di atas didapatkan keluaran berupa IMin, yang merupakan nilai ambang yang cocok untuk diterapkan dalam proses *thresholding*.



Gambar 4. Diagram alir proses thresholding



Gambar 5. Threshold memisahkan lintasan (warna putih) dengan lingkungan dan histogram yang dibentuk

2.3. FEATURE DETECTION

Perangkat keras untuk percobaan ini berupa komputer sebagai berikut :

Tabel 1. Spesifikasi teknis komputer pengolah citra

Processor	Intel Celeron-II 2,4 GHz
Motherboard	GygaByte
Memory	Spectek 256MB DDR
VGA	Nvidia Riva TNT2 M64 Pro 32 MB
OS	Windows XP Professional

Perangkat pengakuisisi citra berupa kamera beserta *frame grabber* :

Tabel 2. Spesifikasi teknis kamera

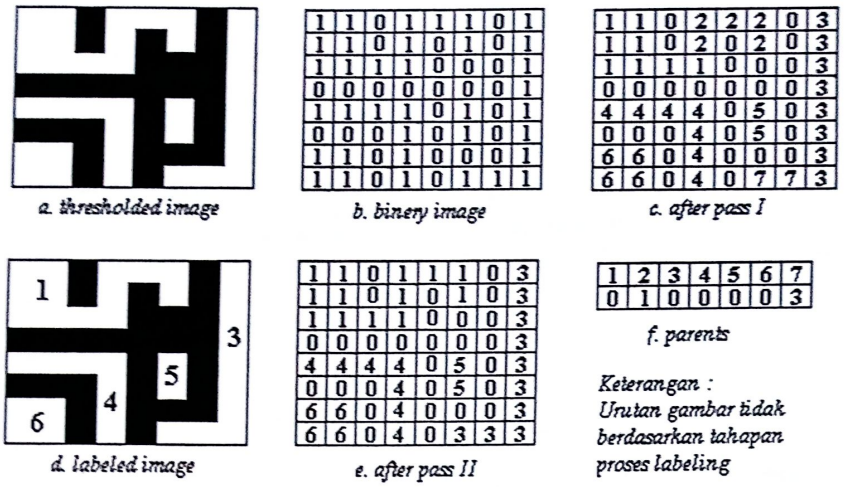
No	Spesifikasi	Keterangan
1.	Tipe/Merk	Genius VideoCAM Slim USB2
2.	Antar muka	USB 1.1 atau USB 2.0
3.	Resolusi video	640x480, true 300K pixels
4.	Resolusi interpolasi	Sampai 1,3 Mega pixel
5.	Transfer rate	Automatic
6.	Kedalaman warna	Maksimum 32 bit
7.	Kemampuan *capturing*	*Frame rate* maksimum /USB2.0 CIF:30fps,VGA: 30 fps
8.	Format file	JPEG
9.	Tipe lensa	Fokus manual
10.	Platform	Notebook atau PC
11.	Dimensi	40x76,9x15,1 mm (WxHxD)

3.1 *Labeling*

Label digunakan agar program dapat mengenali dan memproses obyek tertentu diantara sekian banyak obyek. Proses pemberian label disebut sebagai labeling. Labeling merupakan salah satu tahap penting dalam image processing.

Sebelum labeling, gambar harus dithreshold agar menjadi binery image. Misalkan, binery image ini disebut sebagai *SourceImage*. Proses labeling memerlukan gambar lain katakanlah *DestinationImage*, untuk menampung hasil labeling. Proses labeling memerlukan sebuah variabel *Label* dan sebuah aray *Parent*. Posisi pixel dinyatakan oleh pixel(x,y), dengan x dan y adalah kolom dan baris image. Untuk mempermudah uraian, pixel-pixel tertentu diberi sebutan sebagai berikut :

Pada SourceImage : pixel yang diamati adalah pixel(a,b) dan disebut Spixel. Pixel(a-1,b) disebut SpixelL (karena berada di kiri Spixel), pixel(a,b-1) disebut SpixelT (berada di atas Spixel). Aturan ini berlaku juga bagi DestinationImage sehingga akan terdapat Dpixel, DpixelL, dan DpixelT.

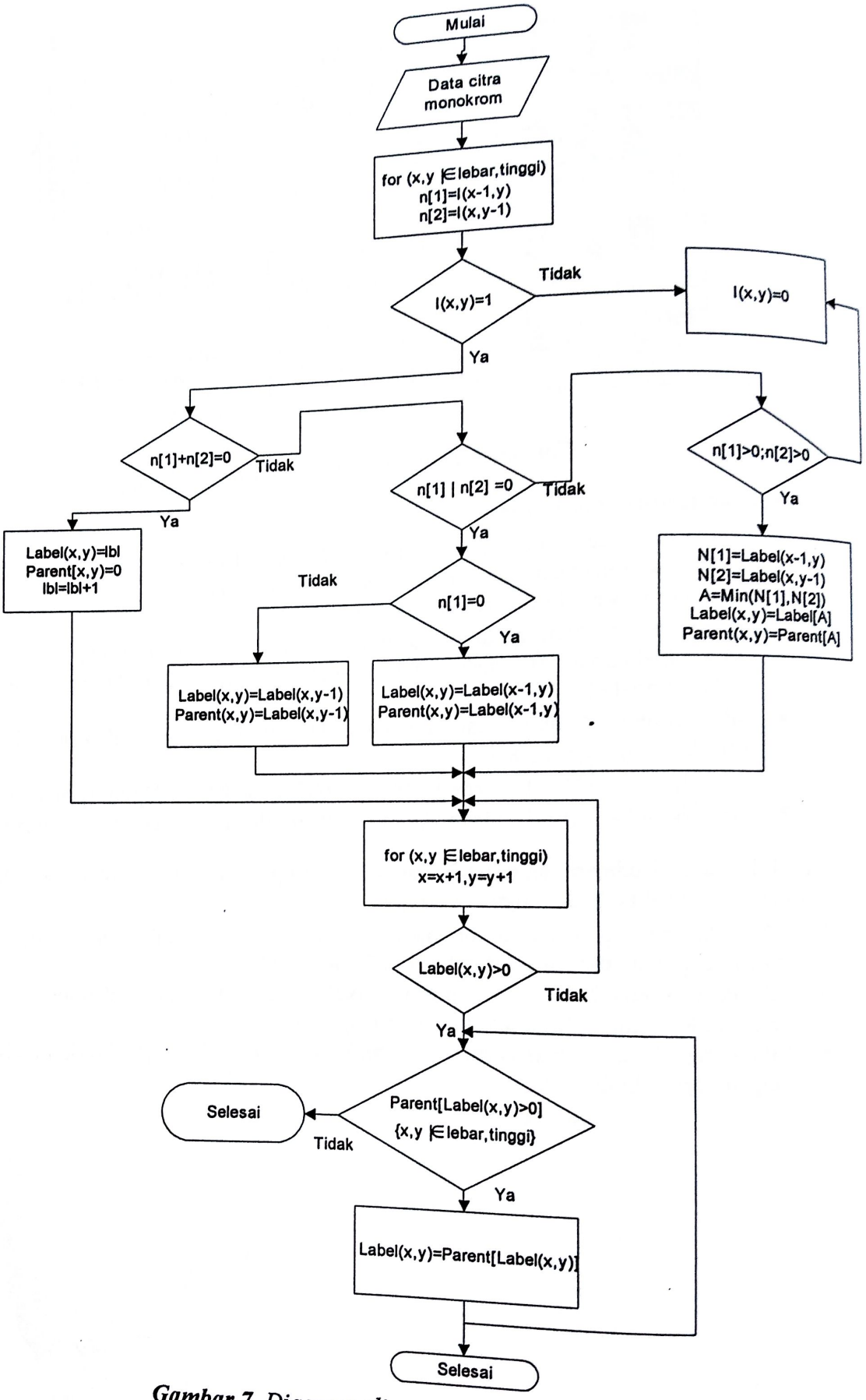


Gambar 6.Tahap-tahap dalam proses labeling

Proses labeling terdiri atas 2 tahap :

1. **Pass 1** : Mula-mula, set Label=1. Lalu, proses dilakukan dengan membaca Spixel dari (0,0). Arah pembacaan dari kiri kekanan dan dari atas kebawah. Pada suatu Spixel :
   - Bila Spixel=0 (berarti tidak ada obyek), maka Dpixel=0.
   - Bila Spixel=1, program perlu mengetahui harga DpixelL dan DpixelT.
     - Jika keduanya 0 maka Dpixel=Label, dan Parent[Dpixel]=Label. Kemudian Label=Label+1.
     - Jika keduanya tidak nol dan sama besar, Dpixel diisi dengan salah satu diantara keduanya, misalnya Dpixel=DpixelL
     - Jika DpixelL>DpixelT maka Dpixel=DpixelT, dan Parent[DpixelL]=DpixelT.
     - Jika DpixelL<DpixelT maka Dpixel=DpixelL, dan Parent[DpixelT]=DpixelL

   Hasil dari pass 1 adalah gambar 4.9c dan 4.9f. Pada gambar 4.9c ini, obyek sudah dilabel namun terjadi adanya satu obyek dengan beberapa label.
2. **Pass 2** : Pass 2 digunakan untuk menghindari adanya dua label dalam satu obyek. Proses dilakukan dengan membaca seluruh Dpixel. Pada suatu Dpixel :
   - Bila Parent[Dpixel] tidak nol, maka Dpixel = Parent[Dpixel]. Sekarang harga Dpixel telah berubah. Kemudian dicek lagi harga Parent[Dpixel] .
   - Bila Parent[Dpixel] masih tidak nol maka Dpixel = Parent[Dpixel], demikian seterusnya hingga Parent[Dpixel]=0.



Gambar 7. Diagram alir proses labelling 1 kali iterasi

3.2 Deteksi Warna (Intensitas Pixel)

Khusus untuk operasi pendeteksian warna, proses tidak melewati tahap *graying* karena operasi *color conversion* ini akan menghilangkan sebagian besar informasi warna obyek-obyek pada citra. Citra dalam format RGB (red, green, blue), terlebih dahulu dikonversikan ke dalam format HLS untuk membuat identifikasi lebih baik. Format RGB sangat rentan terhadap perubahan pencahayaan ruang pengambilan citra digital, berbeda dengan HLS (hue, lumination, dan saturation) akan baik dalam mendeteksi benda meskipun terjadi perubahan tingkat pencahayaan pada saat akuisisi citra, karena warna yang berbeda tingkat pencahayaannya hanya akan berbeda nilainya pada komponen "luminasi" saja, sedangkan komponen hue dan saturasi cenderung sama. Proses pengkonversian format citra dilakukan dengan bantuan fungsi dalam library IPL.

Gambar 8. Pendeteksian warna (intensitas pixel) citra bersistem HLS

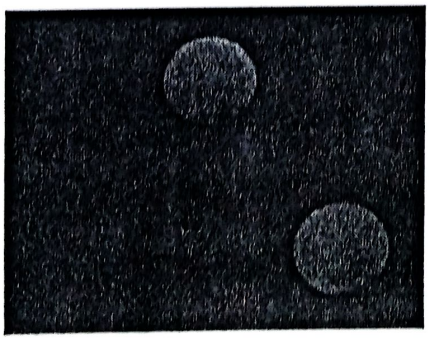
3.3 Deteksi luas daerah dan centroid

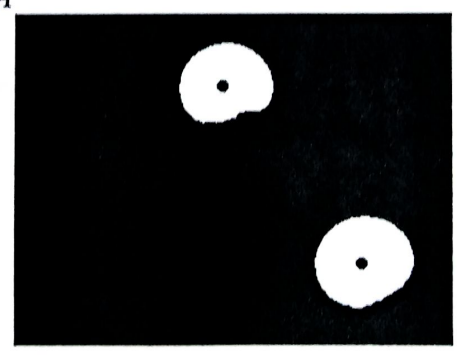
Deteksi luas daerah obyek dalam citra digunakan untuk mengkategorikan obyek berdasarkan ukuran relatif obyek satu terhadap obyek lainnya. Sedangkan penentuan centroid obyek bertujuan mengetahui posisi-posisi relatif obyek satu terhadap oyek yang lain. Setelah proses labeling dan filterisasi dilakukan, pada citra akan terdapat obyek-obyek yang telah dipisahkan. Masing-masing obyek akan dihitung luasnya berdasarkan jumlah pixel yang menghuni daerah pada obyek yang bersangkutan. Centroid dihitung dengan persamaan sebagai berikut:

$$\bar{y} = \frac{\int y dA}{A}$$

dan untuk sumbu x :

$$\bar{x} = \frac{\int x dA}{A}$$





Obyek	Label	Center X	Center Y	Area Pixel
1	1	254	60	3590
2	20	153	188	3060

Gambar 9. Proses *feature detection* dengan hasil posisi titik tengah/centroid obyek dan luas obyek

4. KESIMPULAN

Dari percobaan yang dilakukan metode-metode yang dipilih telah berhasil diterapkan dengan baik dan mampu memberikan hasil pemrosesan yang memuaskan, sehingga dapat dijadikan bahan acuan untuk penerapan lebih lanjut. Algoritma yang terdapat dalam library IPL sangat membantu dalam pemrosesan citra dengan tersedianya fungsi-fungsi dasar pengolahan citra digital berbasis komputer dengan bahasa pemrograman c/c++. Algoritma yang berhasil dikembangkan, (adaptive thresholding, advanced labeling, filter ambang area, pendeteksian berbasis warna dan orientasi (*orientation and color tracking*) telah berfungsi dengan baik.

5. REFERENSI

1. Intel Image Processing Library, *Refference Manual*, Intel Corporation 1997-2000
2. Linda Shapiro & George Stockman,2000, Computer Vision, Prentice-Hall
3. Kadir, Abdul. "Pemrograman C++". Penerbit Andi. Yogyakarta. 2003
4. Shah, Mubarak. "Fundamental of Computer Vision". Computer Science Department University of Central Florida. 1997