BAB I
PENDAHULUAN

1.1 Latar Belakang

Teknologi informasi saat ini telah berkembang pesat termasuk dalam bidang kedokteran. Salah satu contoh yang dapat diterapkan pada bidang kedokteran yaitu penghitungan trombosit, dimana hasil penghitungan akan digunakan sebagai data oleh tenaga medis untuk mengidentifikasi penyakit Demam Berdarah Dengue atau DBD, LUPUS atau penyakit lainnya yang membutuhkan hasil analisis trombosit secara berkala

Beberapa rumah sakit di Pontianak Kalimantan Barat telah menggunakan alat penghitung trombosit otomatis. Alat penghitung trombosit tersebut relatif mahal sehingga pengadaannya tidak dapat dipenuhi pada instansi kesehatan dengan skala kecil seperti puskesmas dan beberapa laboratorium.

Dengan cara konvensional, tenaga medis dapat menghitung jumlah trombosit dengan mengamati langsung citra trombosit yang telah diambil diatas mikroskop. Ketepatan penghitungan sangat dipengaruhi oleh ketelitian tenaga medis yang menganalisa citra trombosit tersebut. Oleh karena itu diperlukan waktu yang lama dan ketelitian yang tinggi untuk menghitung trombosit jika jumlah pasien yang diperiksa cukup banyak.

Teknologi informasi dengan pengolahan citra digital dapat mempermudah penghitungan trombosit dimana waktu penghitungan dapat dipersingkat dan memiliki ketelitian yang sama walaupun data yang dianalisa dalam jumlah banyak. Pengolahan citra digital merupakan pemrosesan citra yang bertujuan untuk memanipulasi dan menganalisis citra dua dimensi dengan bantuan komputer. Citra trombosit yang telah difoto diatas mikroskop akan diproses dengan pengolahan citra digital.

Secara umum, tahapan pengolahan citra digital yang akan diterapkan dalam penghitungan trombosit meliputi akusisi citra, peningkatan kualitas citra (*pre-processing*) dan analisis citra.

Akusisi citra merupakan proses pengambilan citra digital trombosit. Pengambilan citra dilakukan oleh tenaga medis berpengalaman karena tingkat

ketelitian dan kualitas citra sangat dipengaruhi oleh proses pemotretan. Pada tahap *pre-processing* akan dilakukan proses mengkonversi citra *RGB* menjadi *grayscale*, meningkatan kualitas citra digital trombosit, proses morfologi, deteksi tepi dan segmentasi citra. Pada tahap analisis citra, semua objek dalam citra akan ditandai (dilabeli) dengan metode pelabelan komponen dan dihitung luasnya. Objek yang luasnya sesuai dengan luas trombosit akan dihitung.

1.2 Perumusan Masalah

Berdasarkan latar belakang permasalahan, maka dapat dibuat rumusan masalah yaitu bagaimana proses dalam pengolahan citra digital mengolah citra hingga dapat dilakukan penghitungan trombosit.

1.3 Tujuan Penelitian

Tujuan Tugas Akhir ini adalah menghasilkan proses-proses pengolahan citra digital khususnya metode pelabelan komponen yang dapat diterapkan dalam penghitungan trombosit secara otomatis sehingga mencapai tingkat keakuratan perhitungan yang diinginkan dan efisiensi waktu dalam proses penghitungan trombosit.

1.4 Pembatasan Masalah

Pembatasan masalah dari penelitian yang akan dilakukan adalah:

1. Program yang dirancang hanya sebatas simulasi, tidak sampai pada tahap implementasi di lapangan.
2. Tidak membahas proses akusisi citra trombosit.
3. Citra yang digunakan berformat *.*BMP* dengan ukuran (800 x 800) piksel.
4. Penelitian ini tidak berfokus untuk menghitung trombosit yang berhimpit.
5. Pembesaran Mikroskop dengan okuler 10 dan objektif 40.
6. Simulasi menggunakan *Delphi 7.0*.

1.5 Sistematika Penulisan Skripsi

Bab I : PENDAHULUAN

Berisi latar belakang, perumusan masalah, tujuan penelitian, pembatasan masalah dan sistematika penulisan.

Bab II : TINJAUAN PUSTAKA

Berisi landasan teori yang ada hubungannya dengan penelitian yang akan dilakukan dan uraian sistematis tentang hasil-hasil penelitian yang didapat oleh peneliti terdahulu.

Bab III : METODOLOGI PENELITIAN

Berisi tentang Bahan Penelitian, Alat yang Dipergunakan, Metode Penelitian, Variabel atau Data, Perancangan Sistem, Analisis Hasil serta Diagram Alir Penelitian.

Bab IV: HASIL DAN ANALISIS APLIKASI

Berisi data hasil percobaan, pengamatan, survei, dan sebagainya yang telah dirancang pada Bab III. Setiap hasil yang disajikan akan dilakukan analisis untuk mengarah kepada suatu kesimpulan.

Bab V : PENUTUP

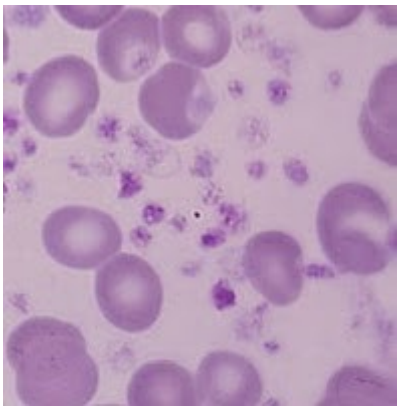
Berisi kesimpulan dari penelitian yang telah dilakukan dan saran/rekomendasi untuk perbaikan, pengembangan atau kesempurnaan/kelengkapan penelitian yang telah dilakukan.

BAB II
TINJAUAN PUSTAKA

2.1 Trombosit

2.1.1 Definisi Trombosit

Trombosit adalah *fragmen* atau kepingan-kepingan tidak berinti dari *sitoplasma megakariosit* yang berukuran 1-4 *mikron* dan beredar dalam sirkulasi darah selama 10 hari. Pada gambaran mikroskopik dengan pewarnaan *Wright – Giemsa*, trombosit tampak sebagai sel kecil, tak berinti, bulat dengan sitoplasma berwarna biru keabu-abuan pucat yang berisi granula merah-ungu yang tersebar merata.



Gambar 2.1 Gambar mikroskopik trombosit dengan pewarnaan *Wright-Giemsa*

Trombosit memiliki peran dalam sistem hemostasis yaitu suatu mekanisme tubuh untuk melindungi diri terhadap kemungkinan perdarahan atau kehilangan darah. Fungsi utama trombosit adalah melindungi pembuluh darah terhadap kerusakan *endotel* akibat trauma-trauma kecil yang terjadi sehari-hari dan mengawali penyembuhan luka pada dinding pembuluh darah. Trombosit membentuk sumbatan dengan jalan *adhesi* (perlekatan trombosit pada jaringan sub-*endotel* pada pembuluh darah yang luka) dan *agregasi* (perlekatan antar sel trombosit).

Orang-orang dengan kelainan trombosit, secara kualitatif maupun kuantitatif sering mengalami perdarahan-perdarahan kecil di kulit dan permukaan *mukosa* yang disebut *ptechiae*, dan tidak dapat menghentikan perdarahan akibat

luka. Agar dapat berfungsi dengan baik, trombosit harus memadai dalam kuantitas (jumlah) dan kualitasnya. Pembentukan sumbat hemostatik akan berlangsung dengan normal jika jumlah trombosit memadai dan kemampuan trombosit untuk beradhesi dan beragregasi dengan baik.

Beberapa uji laboratorium yang digunakan untuk menilai kualitas trombosit adalah agregasi trombosit, retensi trombosit, retraksi bekuan, dan antibodi anti trombosit. Sedangkan uji laboratorium untuk menilai kuantitas trombosit adalah masa perdarahan (*bleeding time*) dan hitung trombosit. Jumlah trombosit normal adalah 150.000 – 400.000 per mm3 darah. Dikatakan trombositopenia ringan apabila jumlah trombosit antara 100.000 – 150.000 per mm^3 darah. Apabila jumlah trombosit kurang dari 60.000 per mm^3 darah maka akan cenderung terjadi perdarahan. Bila jumlah trombosit kurang dari 40.000 per mm^3 darah, biasanya terjadi perdarahan spontan dan bila jumlahnya kurang dari 10.000 per mm^3 darah, perdarahan akan lebih berat. Dilihat dari segi klinik, penurunan jumlah trombosit lebih memerlukan perhatian daripada kenaikannya (trombositosis) karena adanya resiko perdarahan. (Ratnaningsih, 2003).

Metode untuk menghitung trombosit telah banyak dibuat. Hal ini disebabkan oleh tingkat kesulitan yang tinggi untuk menghitung sel-sel trombosit yang merupakan partikel kecil, mudah aglutinasi dan mudah pecah sehingga sukar membedakan trombosit dengan kotoran.

Bahan pemeriksaan yang dianjurkan untuk pemeriksaan hitung trombosit adalah darah EDTA. Antikoagulan ini mencegah pembekuan darah dengan cara mengikat kalsium dan juga dapat menghambat agregasi trombosit.

Hitung trombosit dapat dilakukan secara langsung dan tidak langsung. Metode secara langsung dengan menggunakan kamar hitung yaitu dengan mikroskop fase kontras dan mikroskop cahaya *(Rees-Ecker)* maupun secara otomatis. Metode yang dianjurkan adalah penghitungan dengan mikroskop fase kontras *(Rees-Ecker)*.

Hitung trombosit secara langsung menggunakan kamar hitung yaitu dengan mikroskop cahaya. Pada hitung trombosit cara *Rees-Ecker*, darah diencerkan ke dalam larutan yang mengandung *Brilliant Cresyl Blue* sehingga trombosit tercat biru muda. Sel trombosit dihitung dengan menggunakan kamar hitung standar dan mikroskop. Secara mikroskopik trombosit tampak refraktil dan mengkilat berwarna biru muda/lila lebih kecil dari eritrosit serta berbentuk bulat, lonjong atau koma tersebar atau bergerombol. Cara ini memiliki kesalahan sebesar

16-25% yang disebabkan oleh faktor teknik pengambilan sampel yang menyebabkan trombosit bergerombol sehingga sulit dihitung, pengenceran tidak akurat dan penyebaran trombosit yang tidak merata.

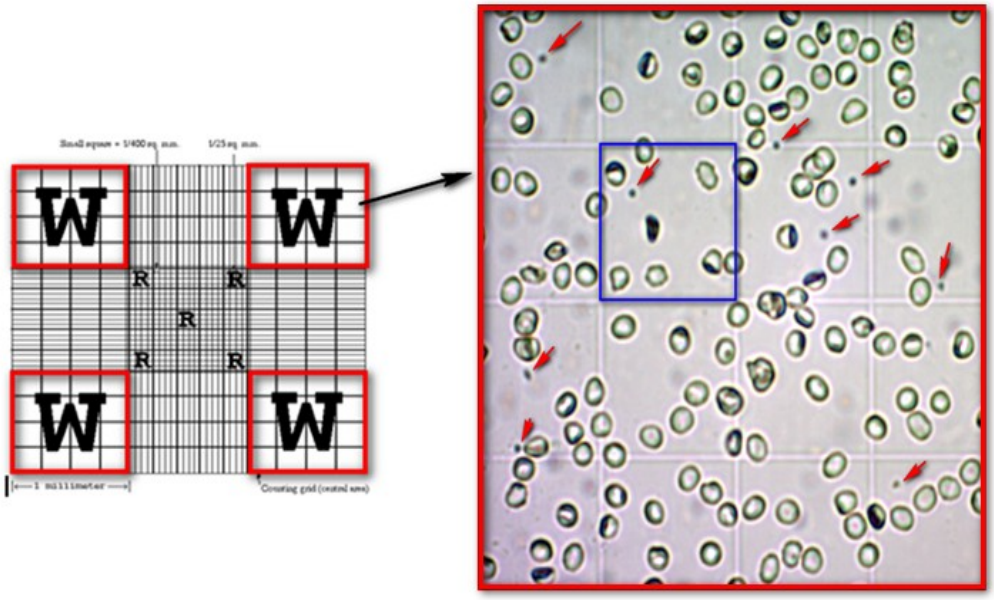
Hitung trombosit secara tidak langsung yaitu dengan menghitung jumlah trombosit pada sediaan apus darah yang telah diwarnai. Cara ini cukup sederhana, mudah dikerjakan, murah dan praktis. Keunggulan cara ini adalah dalam mengungkapkan ukuran dan morfologi trombosit, tetapi kekurangannya adalah bahwa perlekatan ke kaca obyek atau distribusi yang tidak merata di dalam apusan dapat menyebabkan perbedaan yang mencolok dalam perhitungan konsentrasi trombosit. Sebagai petunjuk praktis bahwa hitung trombosit akurat apabila apusan mengandung satu trombosit per dua puluh eritrosit, atau dua sampai tiga trombosit per lapang pandang besar (minyak imersi). Pemeriksaan apusan harus selalu dilakukan apabila hitung trombosit rendah karena penggumpalan trombosit dapat menyebabkan hitung trombosit rendah.

2.1.2 Penghitungan Trombosit

Penghitungan trombosit secara langsung (konvensional dengan metode *Rees Ecker*) diawali dengan mencampur darah dengan antikoagulan. Darah yang telah tercampur sempurna dengan antikoagulan akan dihisap dengan pipet eritrosit dan diencerkan dengan larutan *Rees Ecker*.

Setelah dikocok selama kurang lebih tiga menit, larutan dibuang sekitar dua atau tiga tetes dan dimasukkan ke dalam kamar hitung. Penghitungan dilakukan dengan cara berikut:

$$Jumlah_{trombosit}/mm^3 = \left(\frac{Jumlah\,titik\,trombosit}{Jumlah\,kotak\,pandang} \right) * 80 * 1000\,mm^3$$



Gambar 2.2 Gambar mikroskopik kamar hitung dan trombosit dengan campuran *Rees-Ecker*

Trombosit adalah gambar titik hitam kehijau-hijauan yang ditunjukkan oleh panah merah. Kotak merah merupakan kotak besar kamar hitung yang terdiri dari 16 kotak kecil yang ditunjukkan di sebelah kanan gambar. Kotak biru merupakan salah satu contoh kotak kecil kamar hitung.

Jika pengamatan dilakukan pada 16 kotak kecil seperti pada gambar dan di dalam citra darah terdapat 9 titik trombosit maka

$$Jumlah_{trombosit}/mm^3 = \left(\frac{9}{16}\right) * 80 * 1000\, mm^3 = 45.000\, mm^3$$

2.2 Pengolahan Citra Digital

2.2.1 Definisi Citra Digital

Citra *(image)* didefinisikan sebagai fungsi dua dimensi f(x,y) di mana x dan y adalah koordinat spasial dan amplitudo f pada setiap pasang (x,y) disebut intensitas (*gray level*) citra pada titik tersebut. Jika x dan y berhingga (*finite*) dan diskrit (tdk kontinyu) maka disebut citra digital. Secara umum, citra digital didefinisikan sebagai representasi dari citra dua dimensi sebagai himpunan nilai digital, yang disebut sebagai picture element atau *pixels*. Citra digital terdiri dari sejumlah elemen berhingga yang masing-masing mempunyai lokasi dan nilai.

Elemen-elemen x dan y disebut elemen citra/pels/pixel. Citra digital adalah citra dengan f(x,y) yang nilainya didigitalisasikan (dibuat diskrit) baik

dalam koordinat spasialnya maupun dalam gray levelnya. Digitalisasi dari koordinat spasial citra disebut dengan *image sampling*, sedangkan digitalisasi dari *gray level* citra disebut dengan *gray level quantization*. Citra digital dapat dibayangkan sebagai suatu matriks dimana baris dan kolomnya menunjukkan *gray level* di titik tersebut.

2.2.2 Definisi Pengolahan Citra Digital

Pengolahan citra *(image processing)* merupakan suatu sistem dimana proses dilakukan dengan masukan berupa citra *(image)* dan hasilnya juga berupa citra *(image).* Pada awalnya pengolahan citra ini dilakukan untuk memperbaiki kualitas citra, namun dengan berkembangnya dunia komputasi yang ditandai dengan semakin meningkatnya kapasitas dan kecepatan proses komputer, serta munculnya ilmu-ilmu komputasi yang memungkinkan manusia dapat mengambil informasi dari suatu citra, maka *image processing* tidak dapat dilepaskan dengan bidang *computer vision*.

Computer vision merupakan proses otomatis untuk mengintegrasikan sejumlah besar proses untuk persepsi visual seperti akusisi citra, pengolahan citra, klasifikasi, pengenalan *(recognition),* dan membuat keputusan. *Computer vision* terdiri dari teknik-teknik untuk mengestimasi ciri-ciri objek di dalam citra, pengukuran ciri yang berkaitan dengan geometri objek, dan menginterprestasikan informasi geometri tersebut. Pengolahan citra merupakan proses awal (*preprocessing*) pada *computer vision.* (Munir, 2004:7)

Minat terhadap bidang pengolahan citra secara digital dimulai pada awal tahun 1921, yaitu pertama kalinya sebuah foto berhasil ditransmisikan secara digital melalui kabel laut dari kota New York ke kota London (*Bartlane Cable Picture Transmission System).*

(a)

(b)

Gambar 2.3 (a) Citra digital yang dibuat tahun 1921 dari sebuah '*coded tape'* melalui printer telegraf. Gambar (b) citra digital yang dibuat tahun 1922 dari sebuah *'tape punched'* setelah sinyal (citra) melewati Atlantik dua kali.
Sumber: Marvin (2007:24)

Keuntungan utama yang dirasakan pada waktu itu adalah pengurangan waktu pengiriman foto dari sekitar 1 minggu menjadi kurang dari 3 jam. Foto tersebut dikirim dalam bentuk kode digital dan kemudian diubah kembali oleh printer telegraph. Sekitar tahun 1960 baru tercatat suatu perkembagan pesat seiring dengan munculnya teknologi komputer yang sanggup memenuhi suatu kecepatan proses dan kapasitas memori yang dibutuhkan oleh berbagai algoritma pengolahan citra. Sedangkan untuk bidang kedokteran, pengolahan citra digital mulai digunakan pada tahun 1970-an. Sejak itu berbagai aplikasi mulai dikembangkan, yang secara umum dapat dikelompokkan ke dalam dua kegiatan yaitu (Wijaya, 2007:24):

1. Memperbaiki kualitas citra sehingga dapat lebih mudah diinterpretasikan oleh mata manusia.
2. Mengolah informasi yang terdapat pada citra untuk keperluan pengenalan objek secara otomatis oleh suatu mesin.

2.2.3 Format Citra *BMP (Bitmap)*

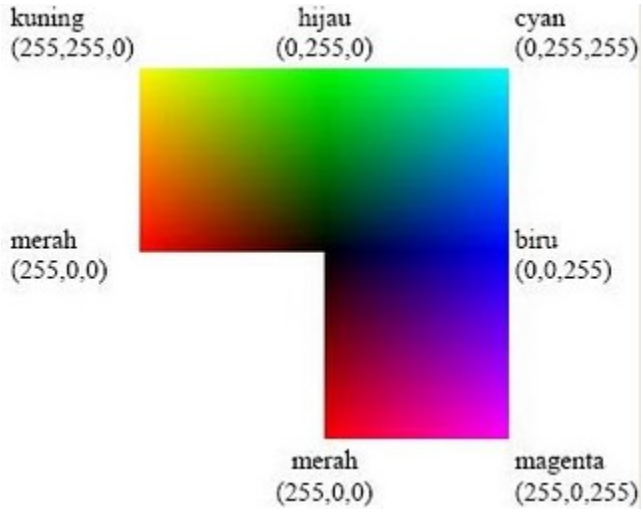
Tipe file BMP umum digunakan pada sistem operasi *Windows* dan *OS/2*. Kelebihan tipe file BMP adalah dapat dibuka oleh hampir semua program pengolah gambar. Baik file BMP yang terkompresi maupun tidak terkompresi, file BMP memiliki ukuran yang jauh lebih besar daripada tipe-tipe yang lain. File BMP tepat penggunaannya sebagai gambar sementara yang mau diedit ulang tanpa menurunkan kualitasnya.

2.2.4 Pengolahan Warna

2.2.4.1 Citra RGB *(True Color Image)*

Citra RGB (*Red, Green, Blue*) adalah suatu model warna yang terdiri dari merah, hijau, dan biru, digabungkan dalam membentuk suatu susunan warna yang luas. Setiap warna dasar misalnya merah, dapat diberi rentang-nilai. Untuk monitor komputer, nilai rentangnya paling kecil 0 dan paling besar 255. Pilihan skala 256 ini didasarkan pada cara mengungkap 8 digit bilangan biner yang digunakan oleh mesin komputer. Dengan cara ini, akan diperoleh warna campuran sebanyak 256 x 256 x 256 = 1677726 jenis warna. Sebuah jenis warna, dapat dibayangkan sebagai sebuah vektor di ruang 3 dimensi yang biasanya dipakai dalam matematika, koordinatnya dinyatakan dalam bentuk tiga bilangan,

yaitu komponen-x, komponen-y dan komponen-z. Misalkan sebuah vektor dituliskan sebagai r = (x,y,z). Untuk warna, komponen-komponen tersebut digantikan oleh komponen *R(ed), G(reen), B(lue).* Jadi, sebuah jenis warna dapat dituliskan sebagai berikut: warna = RGB(30, 75, 255). Putih = RGB (255,255,255), sedangkan untuk hitam= RGB(0,0,0).



Gambar 2.4 Gradiasi warna RGB

2.2.4.2 Derajat Keabuan Citra *(Grayscale Image)*

Grayscale adalah warna-warna *pixel* yang berada dalam rentang gradasi warna hitam dan putih. Proses ini adalah tahap penyederhanaan warna untuk mendapat citra grayscale dari representasi warna RGB (red, green, blue). Nilai *grayscale* diperoleh dengan menggunakan rumus konversi seperti berikut ini:

Y= a * R + b * G + c * B

Keterangan:
Y: nilai *pixel* yang baru pada mode *grayscale*
R: nilai *pixel* red
G: nilai *pixel* green
B: nilai *pixel* blue
a,b,c adalah variabel koefisien dimana nilai a+b+c=1.

2.2.4.3 Citra Biner *(Binary Image)*

Citra biner merupakan citra yang hanya memiliki dua intensitas nilai piksel. Citra biner diperoleh melalui proses pemisahan *pixel-pixel* berdasarkan derajat keabuan yang dimilikinya. *Pixel* yang memiliki derajat keabuan lebih kecil dari nilai batas yang ditentukan akan diberikan nilai 0, sementara *pixel* yang

memiliki derajat keabuan yang lebih besar dari batas akan diubah menjadi bernilai 1. Jadi untuk gambar yang biasanya berwarna hitam putih, nilainya 0 dan 1.

2.2.5 Hubungan Antar *Pixel*

2.2.5.1 *Pixel* Tetangga

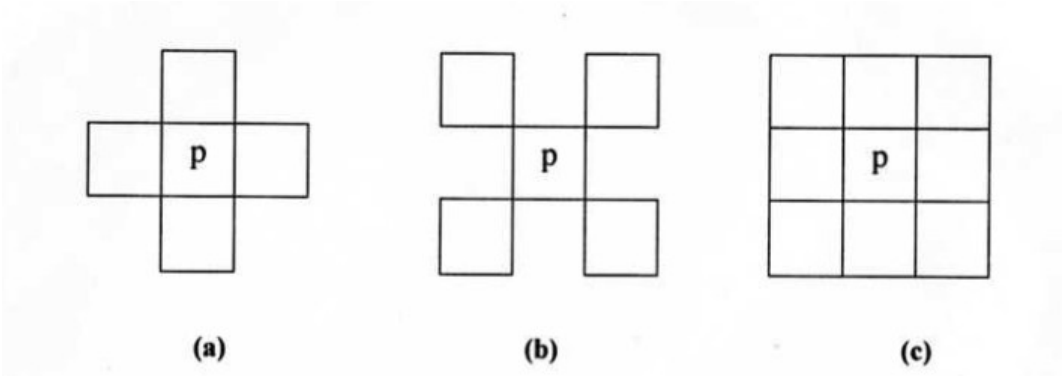
Suatu *pixel* p pada koordinat (x,y) mempunyai dua tetangga horizontal dan dua tetangga vertikal dengan koordinat :

(x+1,y),(x-1,y),(x,y+1),(x,y-1)

Jarak antara *pixel* dihitung dengan unit satuan. Untuk tetangga horizontal dan vertikal jarak antar *pixel*nya dihitung satu satuan, maka set *pixel* ini disebut koneksi 4 tetangga dari p yang dinotasikan dengan $N_4(p)$. Empat tetangga diagonal dari p mempunyai koordinat:

(x+1,y+1),(x+1,y-1),(x-1,y+1),(x-1,y-1)

Untuk tetangga diagonal, jarak antar *pixel*nya dihitung dua satuan dinotasikan dengan $N_D(p)$. Set ini bersama dengan $N_4(p)$ merupakan 8 tetangga dari *pixel* dan dinotasikan dengan $N_8(p)$.

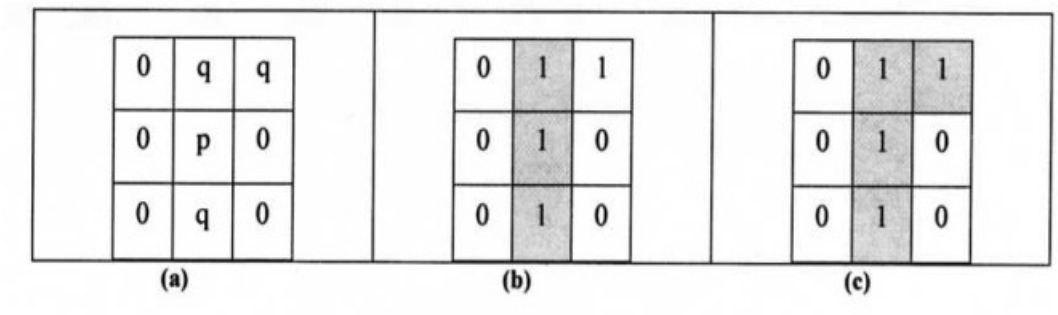


Gambar 2.5 (a) $N_4(p)$, (b) $N_D(p)$, (c) $N_8(p)$.

2.2.5.2 Konektivitas

Konektivitas antar *pixel* merupakan konsep fundamental agar dapat mempermudah definisi mengenai konsep citra digital yang banyak. Untuk menyatakan apakah 2 *pixel* terhubung, maka harus ditentukan apakah dua *pixel* tersebut bertetangga dengan level *grey* yang sama. Contoh untuk citra biner dengan nilai 0 dan 1, dua *pixel* mungkin saja merupakan 4 tetangga, tetapi dikatakan terkoneksi apabila dua *pixel* tersebut memiliki nilai yang sama.

Dua *pixel* p dan q dikatakan 4-*adjacent* jika $q \in N_4(p)$, demikian halnya p dan q dikatakan 8-*adjacent* jika $q \in N_8(p)$. Seperti pada gambar berikut :



Gambar 2.6 (a) Citra dengan *pixel* p dan q, (b) 4-*adjacent*, (c) 8-*adjacent*

Komponen terkoneksi merupakan lintasan *pixel* yang berdekatan *(adjacency pixel).* Hal ini mengimplikasikan bahwa sifat dari komponen terkoneksi tergantung dari *adjacency* yang dipilih. Namun yang umum digunakan adalah 4-*adjacent* dan 8-*adjacent*.

2.2.6 Perbaikan Kualitas Citra

Perbaikan citra bertujuan meningkatkan kualitas tampilan citra untuk pandangan manusia atau untuk mengkonversi suatu citra agar memiliki kualitas yang lebih baik sehingga citra tersebut menjadi lebih mudah diolah dengan mesin (komputer).

2.2.6.1 Peregangan Kontras (*Image Adjustment*)

Kontras menyatakan sebaran terang (*lightness*) dan gelap (*darkness*) di dalam sebuah gambar. Citra dapat dikelompokkan ke dalam 3 kategori kontras antara lain citra kontras rendah (*low contrast*), citra kontras bagus (*good contrast* atau *normal contrast*), dan citra kontras tinggi (*high contrast*). Ketiga kategori ini umumnya dibedakan secara intuitif.

Citra kontras rendah memiliki cirri-ciri dimana sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Dari histogram terlihat sebagian besar derajat keabuannya terkelompok (*clustered*) bersama atau hanya menempati sebagian kecil dari rentang nilai-nilai keabuan yang mungkin. Jika pengelompokan *pixel-pixel* berada disebelah kiri (yang berisi nilai keabuan yang rendah), citranya cenderung gelap. Jika pengelompokan nilai *pixel* berada disebelah kanan (yang berisi nilai keabuan yang tinggi), citranya cenderung terang. Tetapi mungkin saja suatu citra tergolong kontras rendah meskipun tidak

terlalu terang atau tidak terlalu gelap bila semua pengelompokan nilai keabuan berada di tengah histogram.

Citra kontras bagus memperlihatkan jangkauan nilai keabuan yang lebar tanpa ada suatu nilai keabuan yang mendominasi. Histogram citranya memperlihatkan sebaran nilai keabuan yang relatif seragam.

Citra kontras tinggi, seperti halnya citra kontras bagus, memiliki jangkauan nilai keabuan yang lebar, tetapi terdapat area yang lebar yang didominasi oleh warna gelap dana area yang lebar yang didominasi oleh warna terang.

Citra kontras rendah dan citra kontras tinggi dapat diperbaiki kualitasnya dengan operasi peregangan kontras. Melalui operasi ini, nilai keabuan *pixel* akan merentang dari 0 sampai 255 (pada citra 8-bit), dengan kata lain seluruh nilai keabuan *pixel* terpakai secara merata. (Munir, 2004:106)

2.2.6.2 Penajaman Citra (*Image Sharpening*)

Operasi penajaman citra bertujuan memperjelas tepi pada objek di dalam citra. Penajaman citra merupakan kebalikan dari operasi pelembutan citra karena operasi ini menghilangkan bagian citra yang lembut.

Operasi penajaman dilakukan dengan melewatkan citra pada penapis lolos tinggi (*high-pass filter*). Penapis lolos tinggi akan meloloskan (atau memperkuat) komponen yang berfrekuensi tinggi (misalnya tepi atau pinggiran objek) dan akan menurunkan komponen berfrekuensi rendah. Akibatnya, nilai pinggiran objek terlihat lebih tajam dibandingkan sekitarnya. (Munir, 2004:129)

Karena penajaman citra berpengaruh pada tepi (*edge*) objek, maka penajaman citra sering disebut juga dengan penajaman tepi (*edge sharpening*) atau peningkatan kualitas tepi (*edge enchancement*).

Selain untuk mempertajam gambar, penapis lolos tinggi juga digunakan untuk mendeteksi keberadaan tepi (edge detection). Penapis lolos tinggi yang digunakan adalah

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

$$\sum ¿1$$

dengan jumlah koefisien adalah 1. Penapis lolos tinggi dengan koefisien 0 akan menurunkan nilai komponen yang berfrekuensi rendah. Sedangkan jika koefisien

sama dengan 1, komponen berfrekuensi rendah akan tetap sama dengan nilai semula.

Nilai koefisien yang besar di titik pusat penapis memainkan peranan kunci dalam proses konvolusi. Pada komponen citra dengan frekuensi tinggi (yang berarti perubahan besar pada nilai intensitasnya), nilai tengah ini dikalikan dengan *pixel* yang dihitung. Koefisien negatif yang lebih kecil di sekitar titik tengah penapis bekerja untuk mengurangi faktor pembobotan yang besar. Efek nettonya adalah *pixel-pixel* yang bernilai besar diperkuat, sedangkan area citra dengan intensitas *pixel* konstan tidak berubah nilainya.

Karena koefisien penapis mengandung nilai negatif, maka konvolusi mungkin saja menghasilkan *pixel* bernilai negatif. Meskipun intensitasnya bernilai negatif, tetapi kita tidak dapat menampilkannya. Untuk alasan tersebut, implementasi konvolusi men-set nilai negatif menjadi 0.

2.2.7 Deteksi Tepi (*Edge Detection*)

Tepi dari suatu citra mengandung informasi penting dari citra besangkutan. Tepian citra dapat merepresentasikan objek-objek yang terkandung dalam citra tersebut, bentuk dan ukurannya serta terkadang juga informasi tentang teksturnya. Tepian citra adalah posisi dimana intensitas *pixel* dari citra berubah dari nilai rendah ke nilai tinggi atau sebaliknya. Pada umumnya deteksi tepi merupakan langkah awal melakukan segmentasi citra.

Berikut adalah langkah-langkah dalam melakukan deteksi tepi :

1. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis Gaussian. Proses ini akan menghasilkan citra yang tampak sedikit buram. Hal ini dimaksudkan untuk mendapatkan tepian citra yang sebenarnya. Bila tidak dilakukan maka garis-garis halus juga akan dideteksi sebagai tepian. Berikut ini salah satu contoh tapi Gaussian dengan σ = 1.4 yang diperoleh dengan persamaan

   $$G(x,y)=\frac{1}{2\pi\sigma^2}\exp\left(-\left(x^2+y^2\right)/2\sigma^2\right)$$

   dengan posisi (x,y) piksel tengah adalah (0,0).

   $$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

atau dapat disederhanakan menjadi

$$\begin{bmatrix} 0{,}0173 & 0{,}0384 & 0{,}0453 & 0{,}0348 & 0{,}0173 \\ 0{,}0348 & 0{,}0782 & 0{,}1043 & 0{,}0782 & 0{,}0348 \\ 0{,}0435 & 0{,}1043 & 0{,}1304 & 0{,}1043 & 0{,}0435 \\ 0{,}0348 & 0{,}0782 & 0{,}1043 & 0{,}0782 & 0{,}0348 \\ 0{,}0173 & 0{,}0384 & 0{,}0453 & 0{,}0348 & 0{,}0173 \end{bmatrix}$$

2. Melakukan deteksi tepi dengan salah satu operator deteksi tepi yaitu Sobel dengan melakukan pencarian secara horizontal (G_x) dan secara vertikal (G_y).
3. Langkah terakhir adalah binerisasi dengan menerapkan *thresholding*.

2.2.8 Segmentasi Citra

Segmentasi citra merupakan suatu proses pemisahan latar depan (objek/*foreground*) dan latar belakang (*background*) pada suatu citra. Proses yang dilakukan dalam segmentasi citra adalah *thresholding* yang merupakan proses pengelompokan *pixel-pixel* dalam citra berdasarkan batas nilai intensitas tertentu pada suatu *pixel*. Pada operasi ini hasil proses suatu *pixel* tidak bergantung pada kondisi *pixel-pixel* tetangganya, hanya bergantung pada kondisi *pixel* itu sendiri. Dalam operasi *thresholding* suatu *pixel* pada citra asal akan dipetakan menjadi objek atau latar belakang pada citra hasil operasi, tergantung pada intensitas *pixel* itu sendiri pada citra asalnya. Bila intensitasnya sesuai dengan persyaratan intensitas objek maka ia akan dipetakan menjadi *pixel* objek pada citra hasil operasi, dan sebaliknya bila tidak memenuhi syarat, maka ia akan dipetakan menjadi *pixel* yang merupakan bagian latar belakangnya.

Metode yang digunakan untuk mendapatkan nilai *threshold* secara otomatis adalah metode otsu. Metode Otsu menghitung nilai ambang T secara otomatis berdasarkan citra masukan. Pendekatan yang dilakukan metode Otsu adalah dengan menganalisis diskriminan yaitu menentukan suatu variable yang dapat membedakan antara dua atau lebih kelompok yang muncul secara alami. Analisis diskriminan akan memaksimumkan variabel tersebut agar dapat memisahkan objek dengan latar belakang.

Misalkan nilai ambang yang akan dicari dinyatakan dengan k. Nilai k berkisar antara 1 sampai dengan L, dengan L=255. Probabilitas untuk *pixel* i dinyatakan dengan :

$$\text{Pi}=\frac{n_i}{N}$$

Dengan n_i menyatakan jumlah *pixel* dengan tingkat keabuan I dan N menyatakan banyak *pixel* pada citra.

Nilai moment kumulatif ke nol, moment kumulatif ke satu, dan nilai rata-rata berturut-turut dapat dinyatakan sebagai berikut :

$$\omega(k)=\sum_{i=1}^{k} P_i$$

$$\mu(k)=\sum_{i=1}^{k} i.P_i$$

$$\mu_T=\sum_{i=1}^{L} i.P_i$$

Nilai ambang k dapat ditentukan dengan memaksimumkan persamaan :

$\sigma_B^2¿$

dengan :

$\sigma_B^2(k)=¿¿$

Nilai k yang dipilih adalah nilai k yang memaksimumkan persamaan $\sigma_B^2¿$.

2.2.9 Operasi Morfologi

Morfologi mendesripsikan teknik pemrosesan citra yang berkaitan dengan analisa bentuk (morfologi) di dalam citra. Operasi morfologi menggunakan 2 input himpunan yaitu suatu citra (pada umumnya citra biner) dan satu kernel. Khusus dalam morphologi, istilah kernel biasa disebut dengan *structuring elements* (elemen pembentuk struktur). *SE* merupakan suatu matrik dan pada umumnya berukuran kecil. Ada 2 operasi dasar morfologi antara lain :

- ✓ Erosi pada citra *f* dengan *structuring element* s ditunjukkan dengan $f \ominus s$. Jika semua *pixel* pada *structuring element* ditutupi oleh pixel objek maka *pixel* baru diset nilainya sama dengan *pixel* objek, bila tidak maka *pixel* baru akan diset menjadi *pixel background.* Proses erosi akan memperkecil objek sehingga menghilangkan gangguan (*noise*) dan memisahkan objek yang berhimpit. *Structuring element* s ditempatkan pada posisi *(x, y)* dan nilai pixel baru ditentukan menggunakan aturan :

$$g(x,y)=\begin{cases}1 \text{ if } s \text{ fits } f\\ 0 \text{ otherwise}\end{cases}$$

- ✓ Dilasi pada citra f dilakukan dengan *structuring element* s ditunjukkan dengan $f \oplus s$. Jika paling sedikit ada 1 *pixel* pada *structuring element* yang ditutupi oleh *pixel* pada objek maka *pixel* baru diset nilainya dengan nilai *pixel* objek. Efek dilasi pada citra biner adalah memperbesar batas objek yang ada sehingga objek semakin besar dan lubang-lubang yang terdapat pada objek akan tampak mengecil. *Structuring element* s ditempatkan pada posisi *(x, y)* dan nilai baru pixel ditentukan dengan aturan :

$$g(x,y)=\begin{cases}1 \text{ if } s \text{ hits } f\\ 0 \text{ otherwise}\end{cases}$$

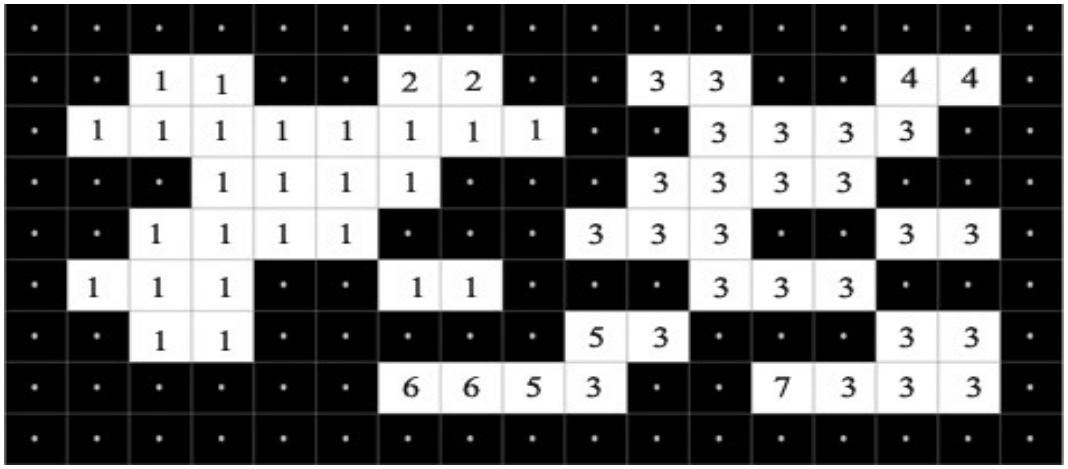
Operasi dilasi diikuti operasi erosi disebut dengan proses closing karena menutup atau menghilangkan celah-celah dari elemen penstruktur pada daerah yang terjangkau dan membiarkan yang lain tetap seperti semula.

Operasi erosi diikuti operasi dilasi disebut dengan proses opening dimana cenderung membuka atau memutus bentuk-bentuk tipis pada objek misalnya disebabkan oleh hasil *thresholding* yang kurang bersih. Proses opening menghapus noise dengan bentuk yang sesuai dengan elemen penstruktur yang digunakan. Proses opening dapat digunakan untuk memisahkan objek-objek yang menempel satu sama lainnya.

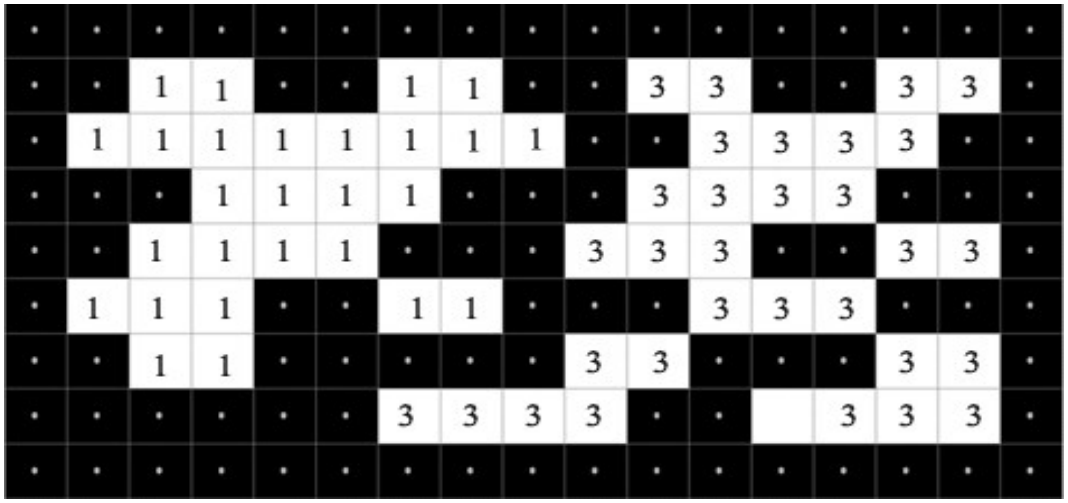
2.2.10 Pelabelan Komponen

Pelabelan (*labelling*) merupakan algoritma untuk menemukan komponen terkoneksi dalam sebuah citra dan menandainya. Operasi ini diperlukan untuk citra biner yang mempunyai banyak objek. Jika pada suatu citra mengandung sejumlah P objek yang akan ditandai, maka bila diasumsikan objek yang akan ditandai memiliki nilai 1 dan latar belakang memiliki nilai 0, *scanning* kita mulai dari sisi kiri atas baris pertama citra menuju arah kanan, kemudian pindah ke baris selanjutnya dan seterusnya. *Pixel* objek pertama yang ditemukan akan diberi label 1, jika *pixel* selanjutnya yang berada di sebelah kanan mempunyai nilai intensitas yang sama maka akan diberi label 1 juga dan begitu seterusnya. Bila *pixel*

selanjutnya pada baris ini ternyata bernilai 0 maka tidak akan diberi label karena merupakan latar belakang dan *pixel* selanjutnya setelah latar belakang yang bernilai 1 akan diberi label 2 sebagai tanda objek kedua dan begitu seterusnya. Ketika *scanning* pindah ke baris selanjutnya maka *pixel* yang terkoneksi dengan *pixel* yang telah diberi label sebelumnya akan diberi label yang sama, sedangkan untuk *pixel* yang tidak terkoneksi pada *pixel* yang telah diberi label akan ditandai dengan label baru sampai sejumlah P objek. Untuk *pixel* yang belum diberi label namun mempunyai dua tetangga yang berbeda label maka pelabelan akan mengikuti label yang lebih kecil. Seperti ditunjukkan pada gambar yang menggunakan koneksi *pixel* dengan 8 tetangga.



(a)



(b)

Gambar 2.7 Proses pelabelan objek pada citra. (a) hasil pelabelan sementara, (b) hasil pelabelan akhir**.**

2.3 Pengujian Perangkat Lunak

Pengujian adalah serangkaian aktifitas yang dapat direncanakan sebelumnya dan dilakukan secara sistematis. Pengujian perangkat lunak adalah satu elemen dari topik yang lebih luas yang sering diacu sebagai verifikasi dan validasi. Pengujian sistem adalah sederetan pengujian yang berbeda yang tujuan utamanya adalah sepenuhnya mempergunakan sistem berbasis komputer.

Adapun metode pengujian yang digunakan adalah Metode *Black-Box*. Metode *Black-Box* merupakan metode pengujian yang memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program.

Ujicoba *Black Box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

a. Fungsi yang salah atau hilang
b. Kesalahan *interface*
c. Kesalahan struktur data atau akses *database eksternal.*
d. Kesalahan performa
e. Kesalahan inisialisasi dan terminasi

Jika perangkat lunak komputer dipertimbangkan, maka pengujian *black-box* berkaitan dengan pengujian yang dilakukan pada antarmuka perangkat lunak. Meskipun didesain untuk mengungkapkan kesalahan, pengujian black-box digunakan untuk memperlihatkan bahwa fungsi-fungsi perangkat lunak adalah operasional; bahwa *input* diterima dengan baik dan output dihasilkan dengan tepat; dan integritas informasi eksternal (seperti file data) dipelihara (Al Bahra, 2006).

2.4 Telaah Pustaka

Adapun tahapan proses pengolahan citra pada simulasi penghitungan trombosit dengan metode pelabelan komponen pengolahan citra digital ini dimulai dengan akusisi citra, konversi citra ke *grayscale*, peregangan kontras, penajaman citra, deteksi tepi, operasi morfologi dan diakhiri dengan *labeling*. Operasi perbaikan citra diharapkan dapat menghasilkan citra dengan kualitas yang lebih baik untuk dianalisis. Proses morfologi dapat menambah ketepatan penghitungan karena dalam beberapa kasus dapat memisahkan objek yang berhimpit sehingga dapat diidentifikasi pada proses *labeling* dan penapisan luas.

Beberapa penelitian yang telah dibuat sebelumnya memiliki kesamaan dengan aplikasi yang akan dirancang. Penelitian ini memiliki persamaan dengan

penelitian Hartadi namun memiliki perbedaan antara lain : *input* citra, objek diteliti (sel darah merah) dan proses-proses pengolahan citra. Hartadi dalam penelitiannya telah membuat program penghitungan sel darah merah. Proses pengolahan citra dimulai dari akusisi data citra, penghapusan derau, pengambangan, penyapuan, hingga citra siap untuk dianalisis. Analisis citra yang dilakukan dalam hal ini adalah pencacahan jumlah sel darah merah. Program yang dibuat memiliki kemampuan untuk mengenali citra, menambahkan derau, menghapus derau, mereka-ulang (rekonstruksi), mengambangkan dan menyapu objek (penapisan berdasarkan luas) yang tidak diinginkan. Berbeda dengan masukan citra pada program simulasi yang akan dibangun pada penelitian ini, masukan citra pada penelitian Hartadi berupa citra hasil digitalisasi citra analog pemotretan sel darah merah pada preparat hapus diatas mikroskop sedangkan program simulasi yang akan dibangun menggunakan masukan berupa citra digital hasil pemotretan sel trombosit pada kamar hitung diatas mikroskop. Untuk mendapatkan hapusan sel darah pada preparat hapus membutuhkan waktu pengeringan kurang lebih 4 jam dan sebaran sel darah yang tidak merata jika dibandingkan dengan penghitungan sel darah pada kamar hitung yang tidak membutuhkan pengeringan. Walaupun objek yang akan dianalisis berbeda, metode yang digunakan untuk analisis citra memiliki kesamaan dengan metode yang akan dibangun yaitu metode pelabelan komponen.

Penelitian ini juga memiliki kesamaan dengan penelitian Aprilianti yaitu otomatisasi penghitungan sel darah merah berbasis pengolahan citra digital dengan metode analisis warna dan ukuran sel namun tetap berbeda pada objek yang dihitung dan metode analisis yang digunakan. Metode pada penelitian yang akan dibangun adalah pelabelan komponen pada citra biner sehingga analisis citra dengan informasi warna sel tidak dapat dilakukan. Analisis citra oleh Aprilianti dilakukan dengan cara membaca informasi warna sel darah merah dan ukuran selnya, kemudian melakukan perbandingan antara hasil penghitungan manual dan penghitungan otomatis dan waktu proses penghitungannya. Dari hasil analisis diperoleh bahwa error terjadi karena adanya kelainan pada sel darah merah dan juga karena adanya obyek sel darah merah yang berimpit.

Berbeda dengan penelitian Aprilianti, penelitian oleh Yulianti lebih menekankan pada penggunaan operasi morfologi untuk meningkatkan akurasi penghitungan sel darah merah berbasis pengolahan citra digital yang memiliki kesamaan dengan program yang akan dibangun pada proses morfologi dan

labelling. Untuk kriteria pengujian lebih lanjut digunakan citra tanpa *noise* dan citra dengan *noise*.

Pada penelitian yang memiliki persamaan objek yang dianalisis namun dengan metode yang berbeda telah dilakukan sebelumnya oleh Tobing yaitu penghitungan trombosit dengan metode analisis warna dan ukuran sel pengolahan citra digital. Proses pengolahan citra dimulai dari akuisisi data citra, penghapusan noise, *thresholding*, hingga citra siap untuk dihitung. Analisis citra dilakukan dengan cara membaca informasi warna sel trombosit dan ukuran selnya, kemudian melakukan perbandingan antara hasil penghitungan manual dan penghitungan otomatis dan waktu proses penghitungannya.

BAB III
METODOLOGI PENELITIAN DAN PERANCANGAN SISTEM

3.1 Alat dan Bahan yang Digunakan

3.1.1 Alat penelitian

Alat penelitian yang digunakan dalam penelitian ini adalah:

- *Flowchart* untuk menggambarkan sistem yang digunakan.
- *Data Flow Diagram (DFD)*, untuk menggambarkan aliran data pada sistem yang terdiri dari dua bagian utama yaitu sistem input data dan sistem pengolahan data.

3.1.2 Bahan penelitian

Bahan penelitian berupa citra sampel darah yang digunakan sebagai input pada program penghitungan trombosit yang akan dibangun. Selain itu digunakan beberapa referensi yang berhubungan dengan trombosit dan pengolahan citra digital untuk membantu dalam proses rancang bangun program.

3.1.3 Perangkat keras

Untuk membangun aplikasi penghitungan trombosit ini diperlukan beberapa perangkat keras. Perangkat keras yang digunakan dalam penelitian ini adalah:

- 1 unit *Personal Computer* (PC) dengan *processor* AMD Athlon(tm) 64 X2 *Dual Core Proccecor* 4600, memori 1GB, *hardisk* 80 GB HDD.
- Mikroskop Prima Star Zeiss dengan kamera AxioCam ERc 5S, P95-C ½ ” 0,5x.
- *LCD LED LG* 18”

3.1.4 Perangkat Lunak

Untuk membangun aplikasi penghitungan trombosit ini diperlukan beberapa perangkat lunak. Perangkat lunak yang digunakan dalam penelitian ini adalah :

- Sistem operasi Windows XP Service Pack 3.
- Borland Delphi 7.0.
- AlphaControls 2009 v6.68

3.2 Metode Penelitian

3.2.1 Studi Literatur

Literatur yang digunakan berupa buku, karya ilmiah maupun jurnal Internasional. Literatur digunakan pula untuk membandingkan aplikasi penghitungan trombosit yang akan dibuat dengan aplikasi penghitungan trombosit yang sudah ada agar dapat menjadi sebuah aplikasi penghitungan trombosit yang lebih baik. Studi literatur yang telah dipelajari antara lain:

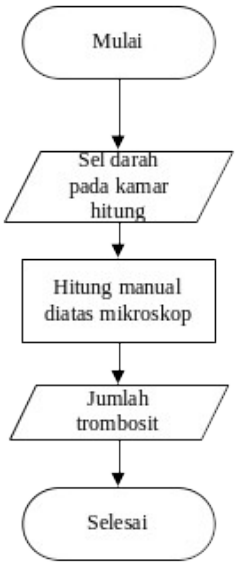
1. Proses-proses dalam pengolahan citra digital yaitu *image processing* (konversi gambar RGB ke *grayscale*, perbaikan kualitas citra, deteksi tepi, morflogi, dan *image analysis* (pelabelan komponen dan penapisan luas).
2. Formula untuk penghitungan jumlah trombosit.

3.2.2 Observasi Dan Pengumpulan Data

Pengumpulan data (sampel darah) sebagai sampel didapatkan dari Rumah Sakit Umum Sudarso Pontianak Kalimantan Barat. Pengambilan gambar darah dilakukan di laboratorium Kedokteran UNTAN dilakukan sehari setelah pengambilan sampel darah di RSU Sudarso. Gambar darah yang telah diambil kemudian dianalisis dan dihitung oleh tenaga medis. Hasil penghitungan trombosit dengan gambar tersebut akan digunakan sebagai indikator pembanding akurasi antara penghitungan manual dan penghitungan program yang akan dibangun.

3.2.3 Analisis Kebutuhan Sistem

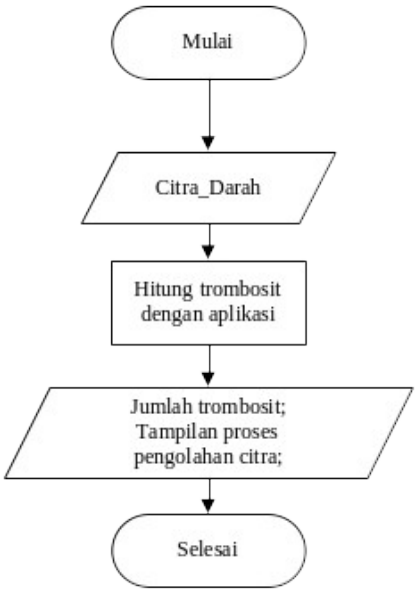
Secara singkat sistem penghitungan trombosit dengan cara konvensional dapat digambarkan sebagai berikut.



Gambar 3.1 Sistem penghitungan trombosit dengan cara konvensional (manual)

Dari analisis proses penghitungan trombosit dengan cara konvensional, maka diperlukan :

1. Akusisi citra menggunakan mikroskop digital.
2. Komputerisasi proses penghitungan trombosit dengan metode pelabelan komponen pengolahan citra digital.
3. Menampilkan jumlah trombosit dan proses pengolahan citra darah yang dapat dilihat oleh user.



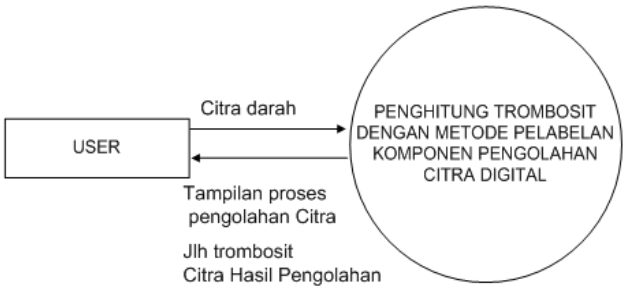
Gambar 3.2 Sistem penghitungan trombosit terkomputerisasi.

Input pada sistem penghitungan trombosit yang dibangun merupakan citra darah. Citra darah diperoleh dari proses akusisi citra (persiapan gambar) yang merupakan proses diluar sistem yang dibangun. Citra darah akan diproses dengan pengolahan citra digital. Output sistem adalah jumlah trombosit dan tampilan proses pengolahan citra digital.

3.2.4 Perancangan Konseptual dari Sistem yang Dibangun

3.2.4.1 *Data Flow Diagram Context Level* (Diagram Konteks)

Diagram konteks adalah diagram yang memberikan gambaran umum terhadap kegiatan yang berlangsung dalam sistem.

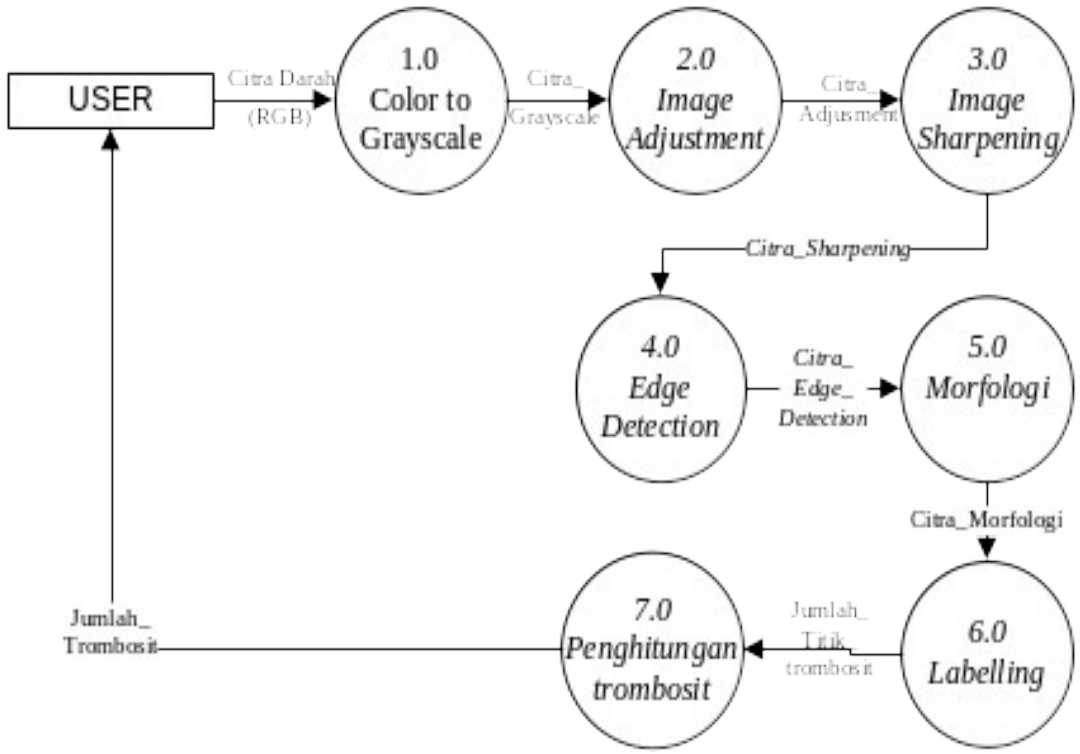


Gambar 3.3 *Data Flow Diagram Context* Aplikasi.

Diagram *overview* adalah diagram yang menjelaskan urutan-urutan proses dari diagram konteks. Sistem ini dibagi menjadi tujuh proses yaitu :

1. Proses 1.0 *color to grayscale* adalah proses mengubah warna citra menjadi abu-abu.
2. Proses 2.0 *Image Adjustment* adalah proses memperbaiki kontras citra.
3. Proses 3.0 *Image Sharpening* adalah proses memperjelas tepi pada objek di dalam citra dengan salah satu cara sharpening dengan mengkonvolusi menggunakan penapis lolos tinggi.
4. Proses 4.0 Edge Detection adalah proses menemukan tepian objek.
5. Proses 5.0 Morfologi adalah proses yang mendesripsikan teknik pemrosesan citra yang berkaitan dengan bentuk (morfologi) di dalam citra.
6. Proses 6.0 *Labeling* adalah proses untuk menemukan komponen terkoneksi dalam citra dan menandainya. Apabila jumlah piksel-piksel yang terlabeli memenuhi intensitas luas sebagai objek maka akan dihitung sebagai objek.

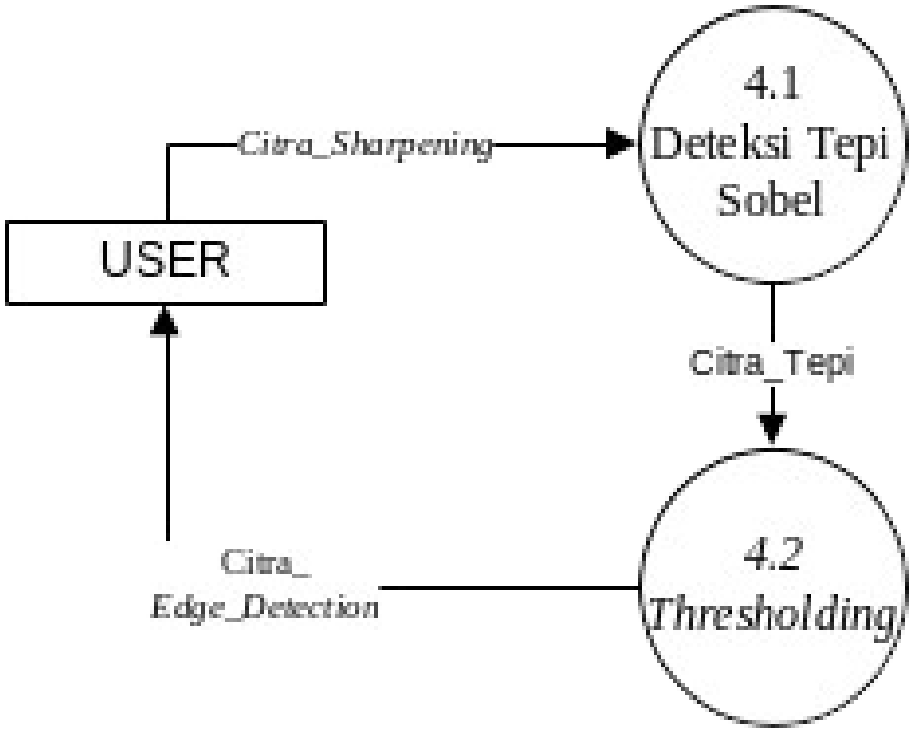
7. Proses 7.0 Penghitungan Trombosit adalah proses penghitungan trombosit dalam 1mm^3.



Gambar 3.4 Diagram o*verview* sistem.

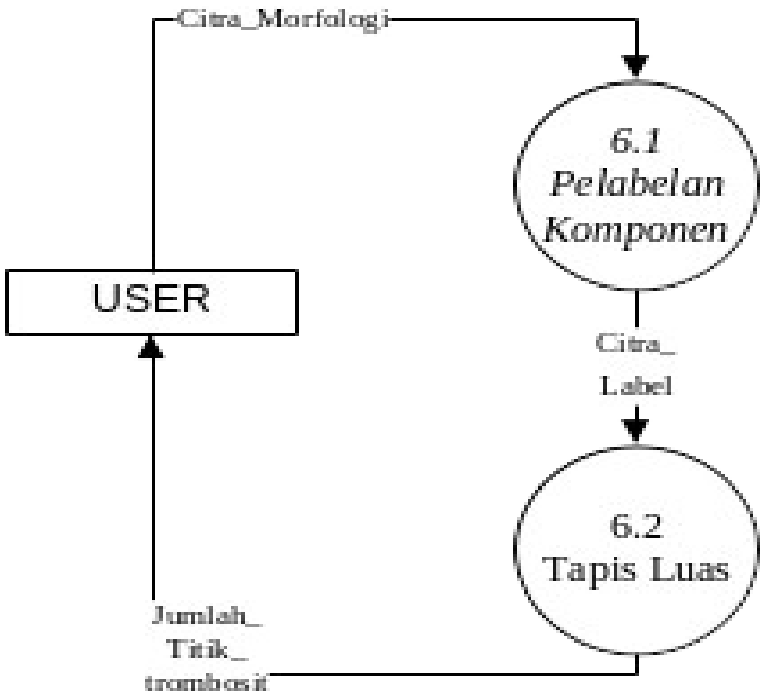
Diagram rinci menguraikan lebih lanjut mengenai proses dari diagram *overview*, yang memperlihatkan arus data masuk dan arus data keluar. Berikut adalah model-model diagram rinci.

1. Proses 4.0 *Edge Detection* menghasilkan citra biner dimana objek didalamnya hanya berupa titik-titik tepian objek dan *background*.
   a. Proses 4.1 Deteksi tepi Sobel adalah proses deteksi tepi menggunakan operator sobel.
   b. Proses 4.2 *Thresholding* adalah proses untuk membagi citra ke dalam segmen *background* dan *foreground* berdasarkan nilai *threshold* yang sesuai.



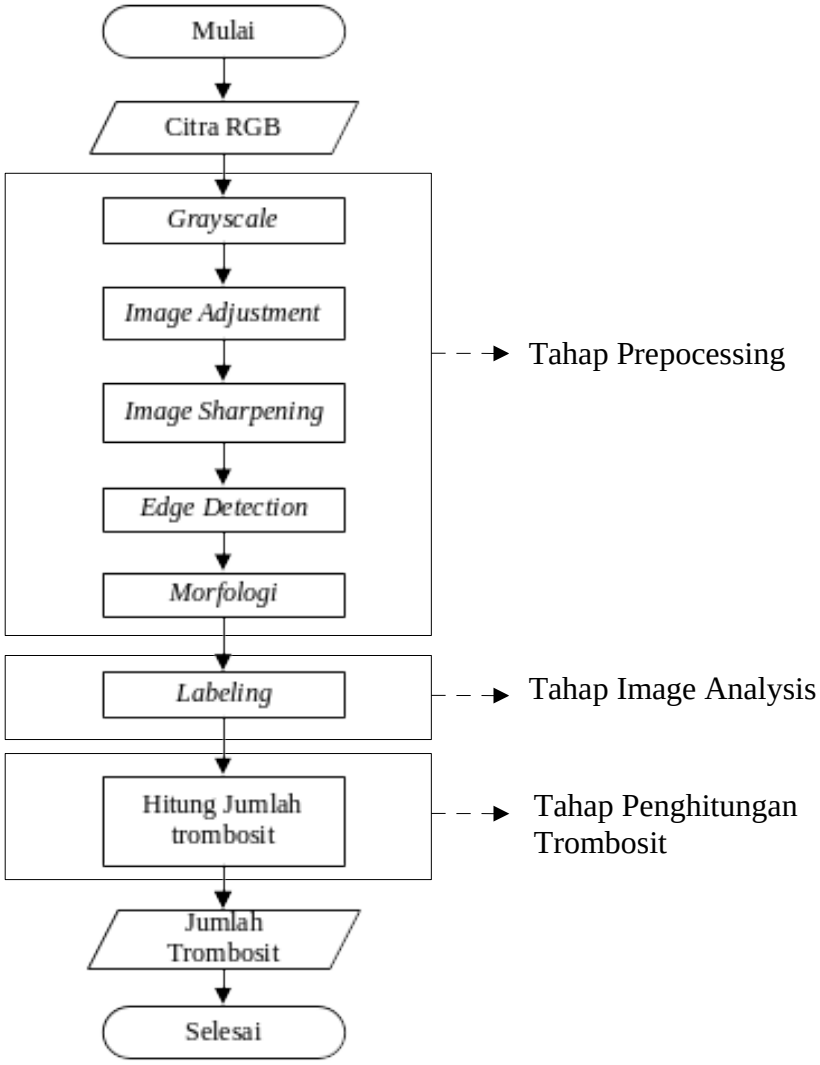
Gambar 3.5 Diagram rinci proses 4.0 (Proses *Edge Detection*).

2. Proses 6.0 *Labeling* adalah proses untuk menemukan komponen terkoneksi dalam citra dan menandainya.
   a. Proses 6.1 Pelabelan Komponen. Proses ini terdiri dari 2 tahap *labeling.*
   b. Proses 6.2 Tapis luas adalah proses untuk menghitung luas objek yang telah dilabeli dan menyeleksi objek yang sesuai dengan kriteria luas trombosit.



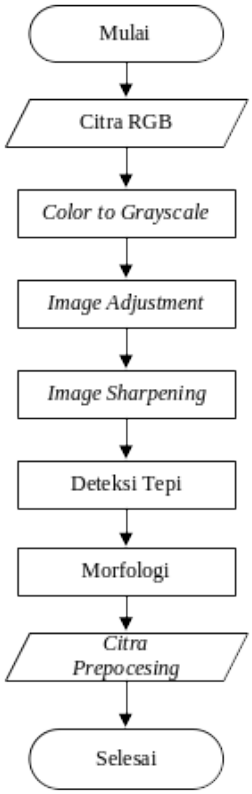
Gambar 3.6 Diagram rinci proses 6.0 (*Labeling*).

3.2.4.2 *Flowchart*



Gambar 3.7 *Flowchart* proses-proses sistem penghitungan trombosit.

3.2.4.2.1 Tahap-Tahap *Pre-Processing*



Gambar 3.8 *Flowchart Prepocessing*

3.2.4.2.1.1 *Grayscale*

Untuk konversi citra RGB ke *grayscale* menggunakan nilai *grayscale* yang diperoleh dengan menggunakan rumus konversi seperti berikut ini:

Y= 0,299 * R + 0,587 * G + 0,114 * B

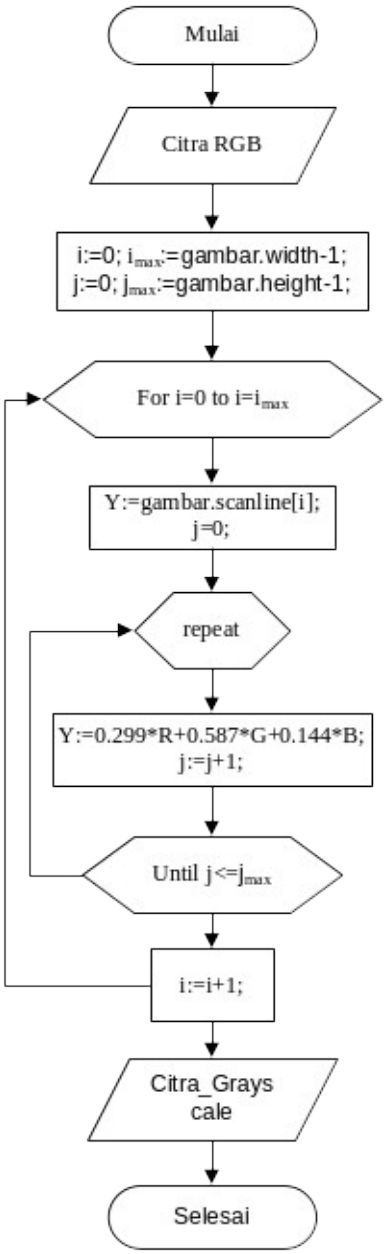
Keterangan:

Y: nilai piksel yang baru pada mode *grayscale*

R: nilai piksel red

G: nilai piksel green

B: nilai piksel blue



Gambar 3.9 *Flowchart Grayscale*

3.2.4.2.1.2 Peregangan Kontras (*Image Adjustment*)

Algoritma *Adjustment* untuk citra grayscale :

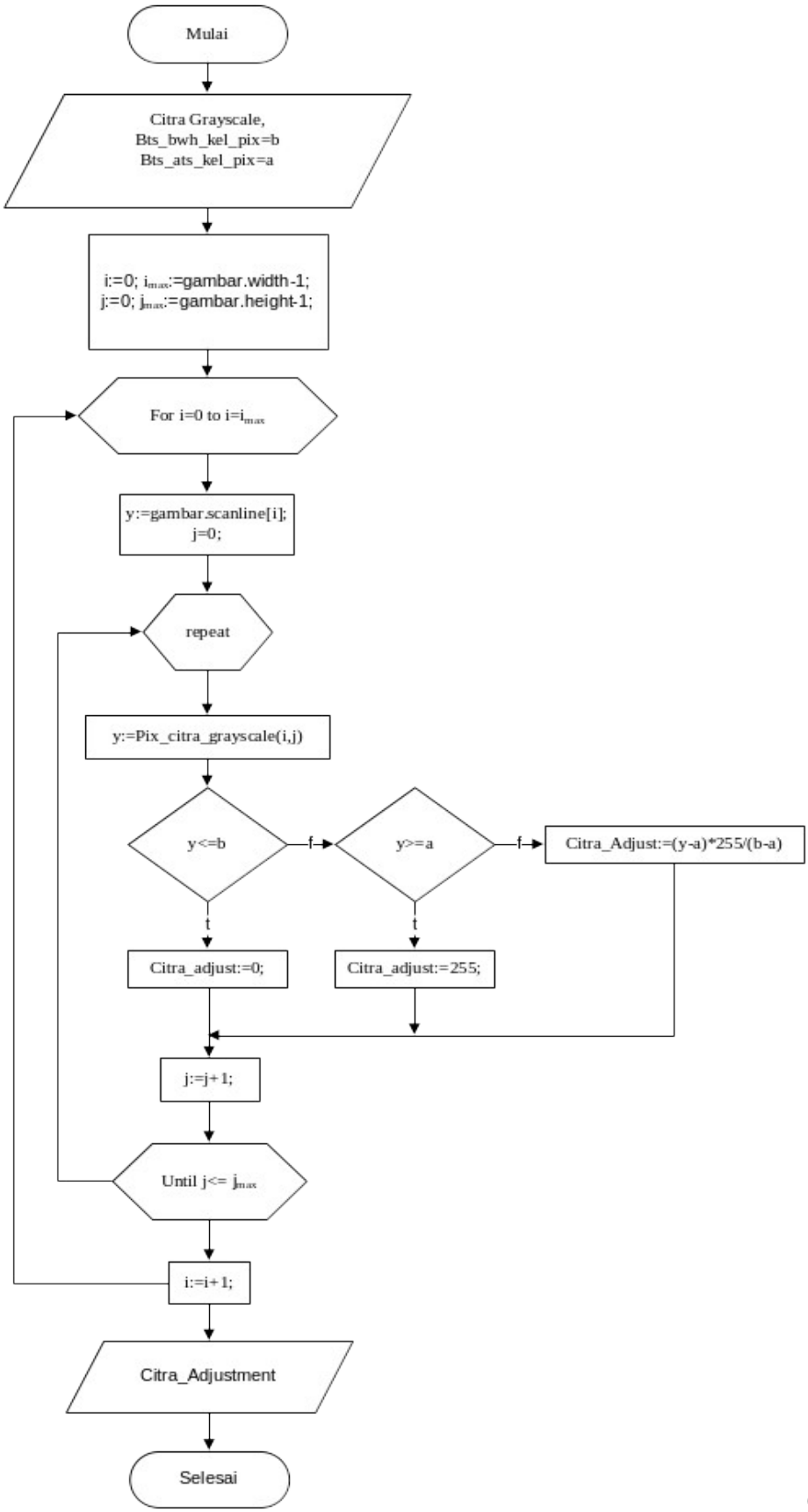
i. Nilai keabuan piksel yang berada di batas atas diset sama dengan 255

ii. Nilai keabuan piksel yang berada di batas bawah diset sama dengan 0

iii. Nilai yang berada diantara batas bawah dan batas atas dipetakan (diskalakan) untuk memenuhi rentang nilai keabuan yang lengkap (0 sampai dengan 255).

$$Citra\ Adjustment = \frac{y-a}{b-a} * 255$$

Dimana y adalah nilai keabuan semula, b adalah nilai keabuan terendah dari kelompok pixel, dan a adalah nilai keabuan tertinggi dari kelompok pixel.



Gamba

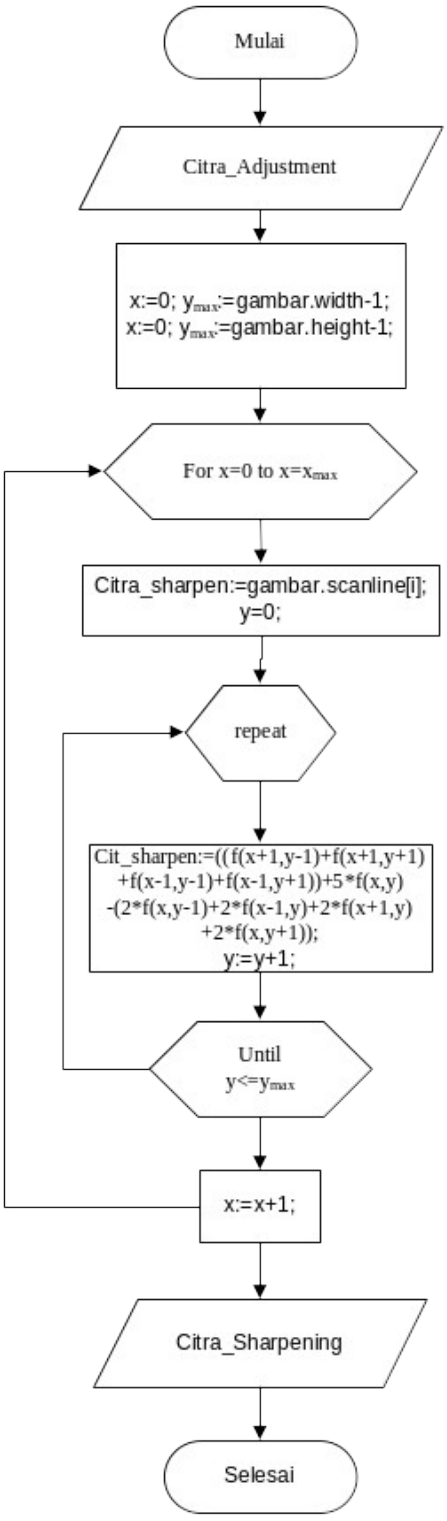
r 3.10 *Flowchart Image Adjustment*

3.2.4.2.1.3 *Image Sharpening*

Dibawah ini merupakan *flowchart image sharpening* yang menggunakan penapis lolos tinggi seperti berikut.

$$\begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

$$\sum ¿1$$



Gambar 3.11 *Flowchart Image Sharpening*

3.2.4.2.1.4 *Edge Detection*

Algoritma deteksi tepi :

i. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis Gaussian. Tapis Gaussian yang digunakan

$$\frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

atau

$$\begin{bmatrix} 0{,}0173 & 0{,}0384 & 0{,}0453 & 0{,}0348 & 0{,}0173 \\ 0{,}0348 & 0{,}0782 & 0{,}1043 & 0{,}0782 & 0{,}0348 \\ 0{,}0435 & 0{,}1043 & 0{,}1304 & 0{,}1043 & 0{,}0435 \\ 0{,}0348 & 0{,}0782 & 0{,}1043 & 0{,}0782 & 0{,}0348 \\ 0{,}0173 & 0{,}0384 & 0{,}0453 & 0{,}0348 & 0{,}0173 \end{bmatrix}$$

dengan $\sigma = 1.4$.

ii. Melakukan deteksi tepi dengan operator Sobel dengan melakukan pencarian secara horizontal (G_x) dan secara vertikal (G_y). Tinjauan pengaturan pixel disekitar pixel (x,y).

$$\begin{bmatrix} a_0 & a_1 & a_2 \\ a_7 & (x,y) & a_3 \\ a_6 & a_5 & a_4 \end{bmatrix}$$

$$\begin{bmatrix} a_0=f(x-1,y+1) & a_1=f(x,y+1) & a_2=f(x+1,y+1) \\ a_7=f(x-1,y) & f(x,y) & a_3=f(x+1,y) \\ a_6=f(x-1,y-1) & a_5=f(x,y-1) & a_4=f(x+1,y-1) \end{bmatrix}$$

Operator sobel adalah magnitude dari gradient yang dihitung dengan

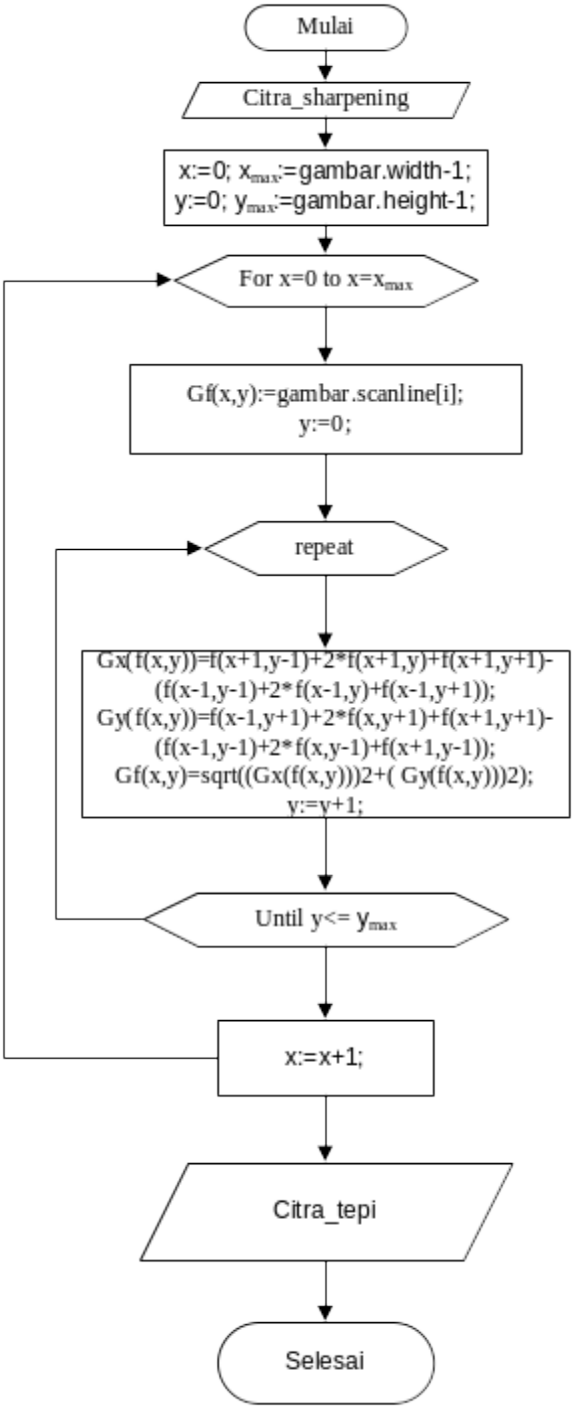
$M=\sqrt{G_x^{\,2}+G_y^{\,2}}$

Yang dalam hal ini, turunan parsial dihitung dengan

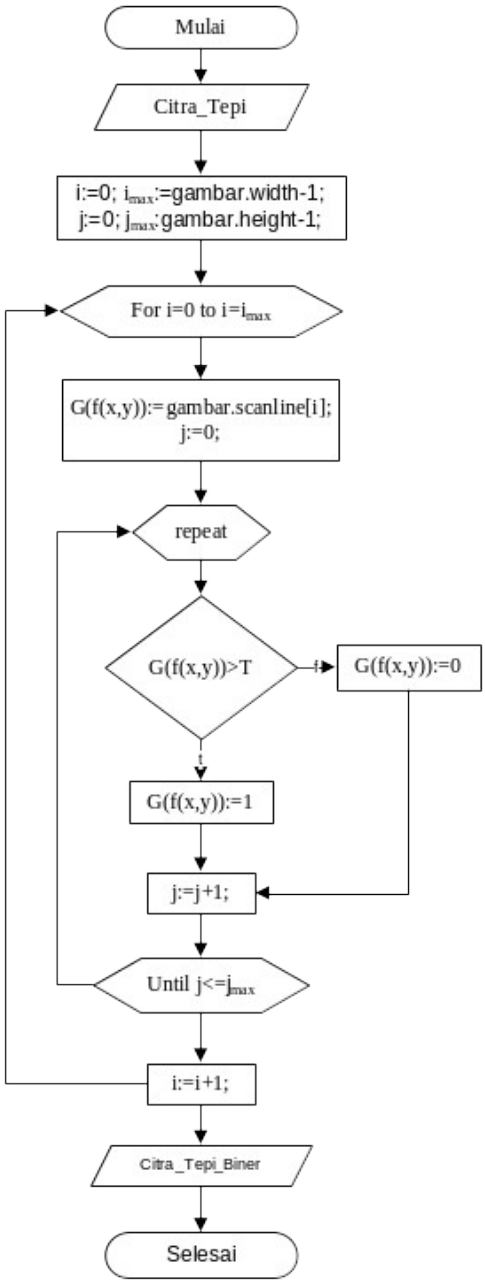
$G_x=(a_2+2a_3+a_4)-(a_0+2a_7+a_6)$

$G_y=(a_0+2a_1+a_2)-(a_6+2a_5+a_4)$

iii. Binerisasi dengan menerapkan *thresholding*.



Gambar 3.12 *Flowchart* Deteksi Tepi



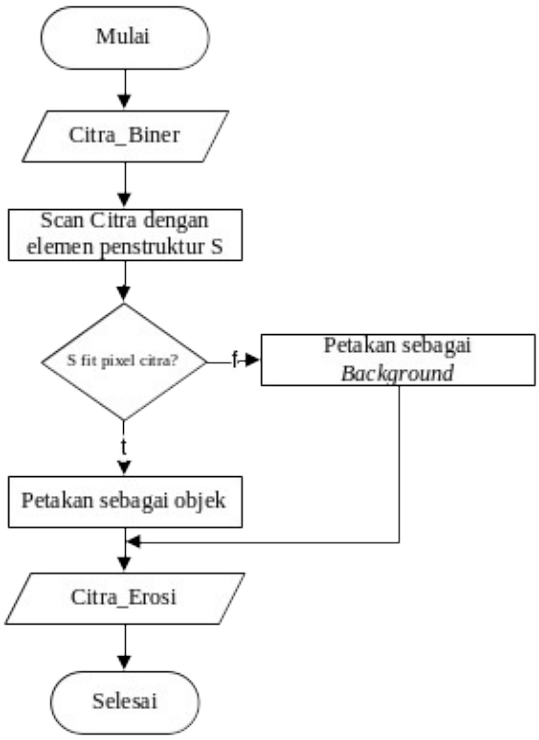
Gambar 3.13 *Flowchart Threshold*

3.2.4.2.1.5 Morfologi

Elemen penstrukstur S yang digunakan adalah

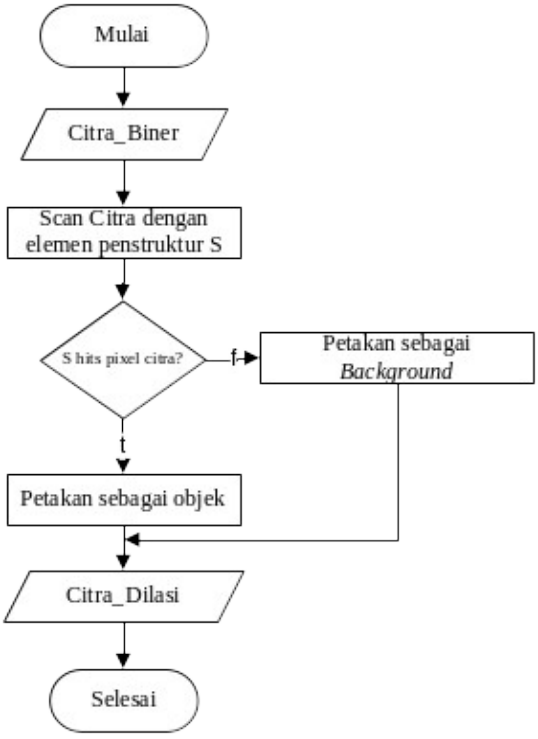
0
0

1. Erosi



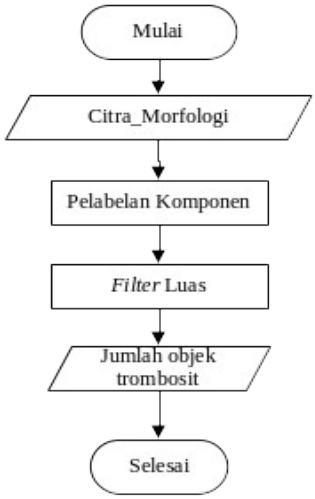
Gambar 3.14 *Flowchart* Erosi

2. Dilasi



Gambar 3.15 *Flowchart* Dilasi

3.2.4.2.2 Tahap-tahap *Image Analysis*

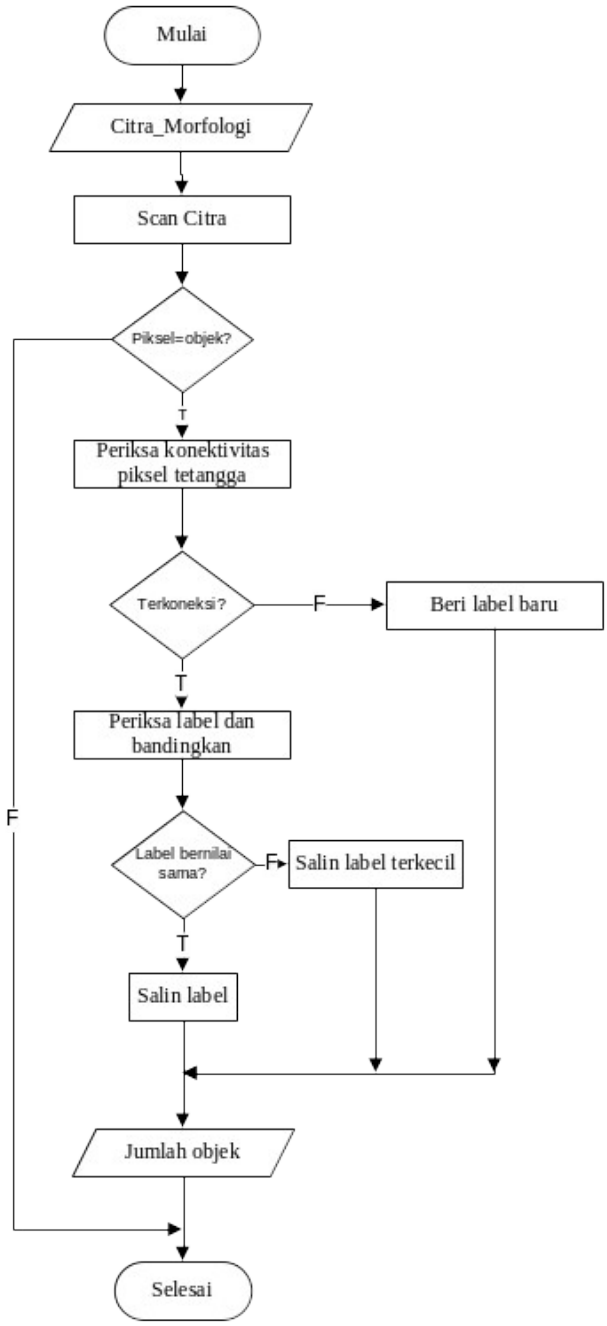


Gambar 3.16 *Flowchart Image Analysis*

3.2.4.2.2.1 Pelabelan Komponen

Algoritma pelabelan komponen :

1. Scanning citra dari baris sudut kiri atas ke sudut kanan atas
2. Jika piksel yang dibaca adalah milik objek, maka :
   - ✓ Periksa konektivitas piksel tetangga, jika terkoneksi maka periksa labelnya dan bandingkan
   - ✓ Jika label piksel tetangga yang terkoneksi bernilai sama, maka salin labelnya.
   - ✓ Jika label piksel tetangga yang terkoneksi mempunyai nilai yang berbeda, maka salin label terkecil.
   - ✓ Selain itu, beri label baru pada piksel objek tersebut.
3. Jika masih ada piksel yang perlu diperiksa, ulangi langkah 2.

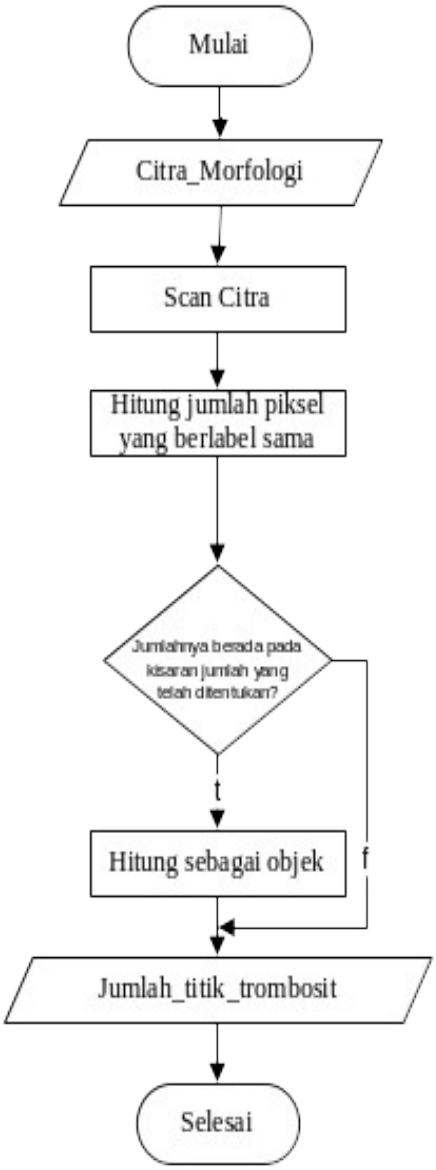


Gambar 3.17 *Flowchart* Pelabelan Komponen

3.2.4.2.2.2 *Filter* Luas

Algoritma filter luas :

1. Baca nilai label piksel objek
2. Hitung jumlah piksel yang berlabel sama.
3. Jika jumlah piksel yang berlabel sama bukan merupakan jumlah rata-rata piksel yang telah ditentukan sebagai objek maka hapus objek tersebut.

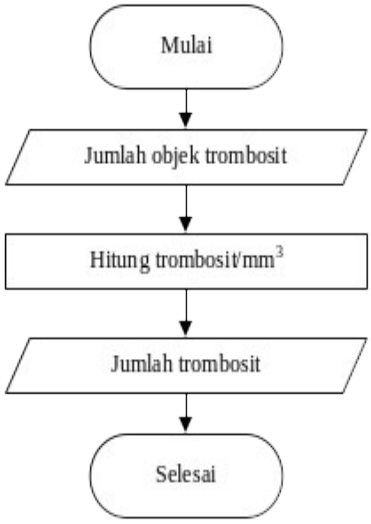


Gambar 3.18 *Flowchart* Filter Luas

3.2.4.2.3 Penghitungan Trombosit

Jumlah trombosit atau objek pada proses segmentasi akan dimasukkan pada proses penghitungan trombosit/mL. Jumlah trombosit merupakan jumlah objek dibagi jumlah kotak hitung dan dikalikan dengan jumlah kotak seluruhnya untuk 1ml darah.

$$Jumlah_{trombosit} / mm^3 = \left(\frac{Jumlah\,titik\,trombosit}{Jumlah\,kotak\,pandang} \right) * 80 * 1000\,mm^3$$



Gambar 3.19 Penghitungan Trombosit mm^3

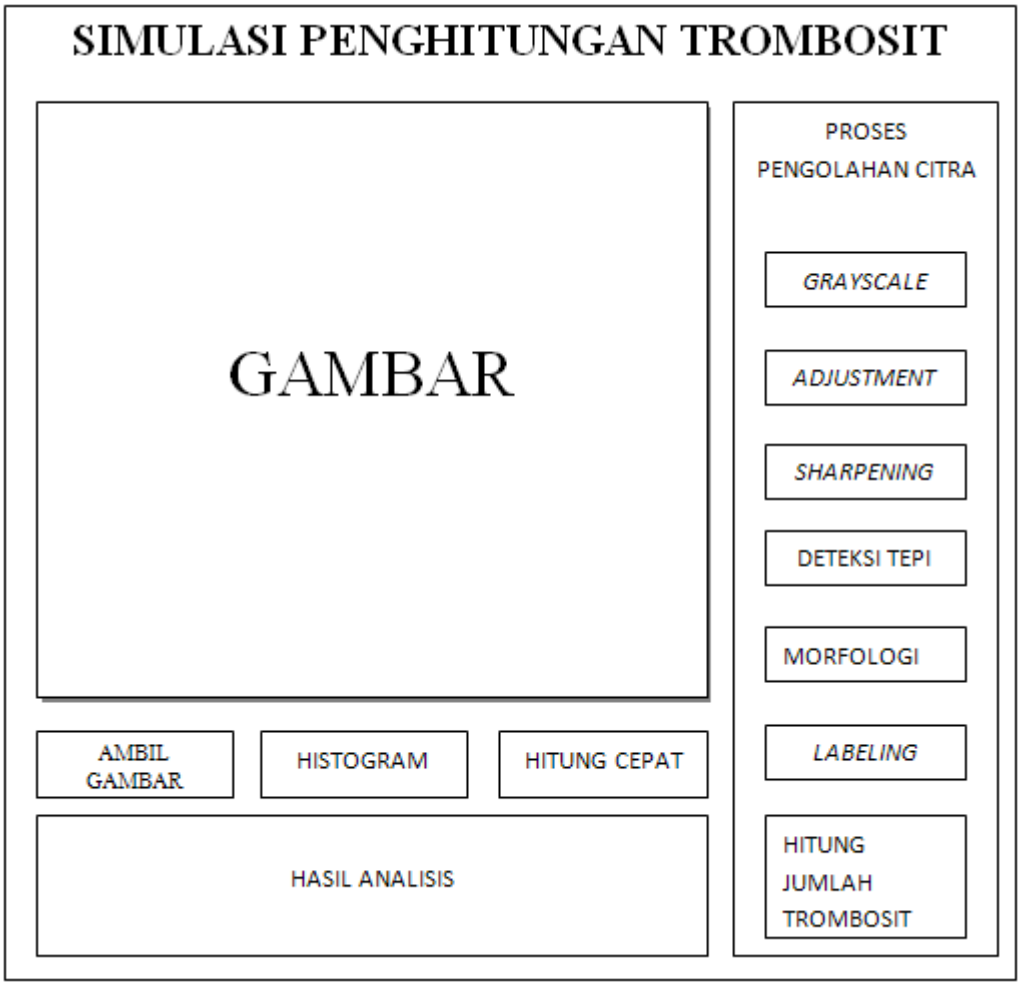
3.2.4.3 Perancangan Antarmuka Aplikasi

3.2.4.3.1 Perancangan Struktur Antarmuka Aplikasi

Aplikasi simulasi penghitungan trombosit ini merupakan aplikasi desktop yang dibangun menggunakan Delphi. Struktur antarmuka aplikasi yang dirancang hanya terdiri dari satu *form*, dimana terdapat tombol-tombol navigasi proses-proses yang akan diterapkan pada citra masukan.

3.2.4.3.2 Perancangan Layout dan Komponen Antarmuka Aplikasi

Pada halaman terdapat beberapa bagian yaitu bagian header yang bertuliskan “SIMULASI PENGHITUNGAN TROMBOSIT” sebagai nama aplikasi. Sebelah kiri terdapat beberapa menu tombol fungsi-fungsi antara lain penginputan citra, pengolahan citra dan penghitungan trombosit. Perancangan aplikasi digambarkan sebagai berikut:



Gambar 3.20 *Layout* Antarmuka

3.2.5 Implementasi

Implementasi dituangkan dalam baris kode dengan menggunakan bahasa pemrograman Borland Delphi 7.0. Hasil implementasi dapat dipergunakan untuk menghitung jumlah trombosit dengan masukan berupa citra darah hasil pemotoan diatas mikroskop sehingga dapat memaksimalkan efisiensi waktu dan ketepatan penghitungan. Aplikasi juga mempunyai fitur bagi *user* untuk melihat perubahan citra pada tiap proses pengolahan citra digital hingga proses penghitungan trombosit.

3.2.6 Pengujian Dan Validasi Aplikasi

Pengujian dilakukan sebanyak dua belas pengujian dengan menginputkan citra darah dan menerapkan semua fungsi yang ada. Analisis dilakukan dalam

tahap pengujian dan validasi untuk mengetahui karakteristik sistem dan mengidentifikasi jika terdapat ketidakkonsistenan sistem. Hasil analisis juga digunakan sebagai dasar perbaikan. Pada tiap pengujian, waktu yang diperlukan untuk penghitungan trombosit, *error*, hasil penghitungan trombosit dengan aplikasi akan dicatat dan dibandingkan hasil penghitungan trombosit dengan penghitungan manual. Perbandingan hasil penghitungan akan dicatat sebagai persentase keberhasilan penghitungan.

Berikut adalah contoh tabel pengujian performa aplikasi:

Tabel 3.1 Contoh tabel hasil pengujian performa aplikasi

No	Gambar	Jumlah titik trombosit hitung program	Jumlah titik trombosit hitung manual	Jumlah trombosit hitung program	Jumlah trombosit hitung manual	Persentase Keberhasilan (%)	Waktu penghitungan (detik)	Keterangan
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…

BAB IV
HASIL DAN ANALISIS APLIKASI

4.1. Hasil Perancangan

4.1.1. Antarmuka Aplikasi Simulasi Penghitungan Trombosit

Aplikasi Simulasi Penghitungan Trombosit hanya terdiri dari 1 menu form. Form ini memiliki tombol-tombol yang masing-masing tombol merupakan proses yang akan diterapkan pada gambar *input.* Untuk menampilkan proses tiap langkah maka setelah mengambil gambar maka dipilih trombol *grayscale.* Sedangkan untuk melakukan proses sekaligus, maka setelah meng-*input* gambar dipilih tombol hitung cepat.

Gambar 4.1 Antarmuka menu utama *Platelet Count* (Hitung Trombosit)

Tabel 4.1 Daftar Menu *Platelet Count (*Hitung Trombosit*)* dan Fungsinya

Menu	Fungsi
Ambil Gambar	Menampilkan gambar dalam format *.BMP*
Histogram RGB	Menampilkan histogram citra *RGB*
Hitung Cepat	Menjalankan semua proses yang ada dan secara khusus digunakan untuk mengurangi kejenuhan pengguna pada tombol navigasi dalam jumlah banyak.
Grayscale	Mengubah gambar *RGB* menjadi *grayscale*
Image Adjustment	Memperbaiki kontras citra
Image Sharpening	Memperjelas objek
Deteksi Tepi	Mendapatkan tepian objek sekaligus mengkonversi citra menjadi citra biner
Morfologi	Memperbaiki bentuk suatu objek dalam citra
Labeling	Menemukan komponen terkoneksi dan menandainya. Disisipkan pula fungsi untuk penapisan luas untuk menentukan jumlah objek yang sesuai dengan kriteria yang diinginkan
Hitung Trombosit	Menghitung jumlah trombosit dalam mm^3
Help ?	Memberikan panduan penggunaan aplikasi

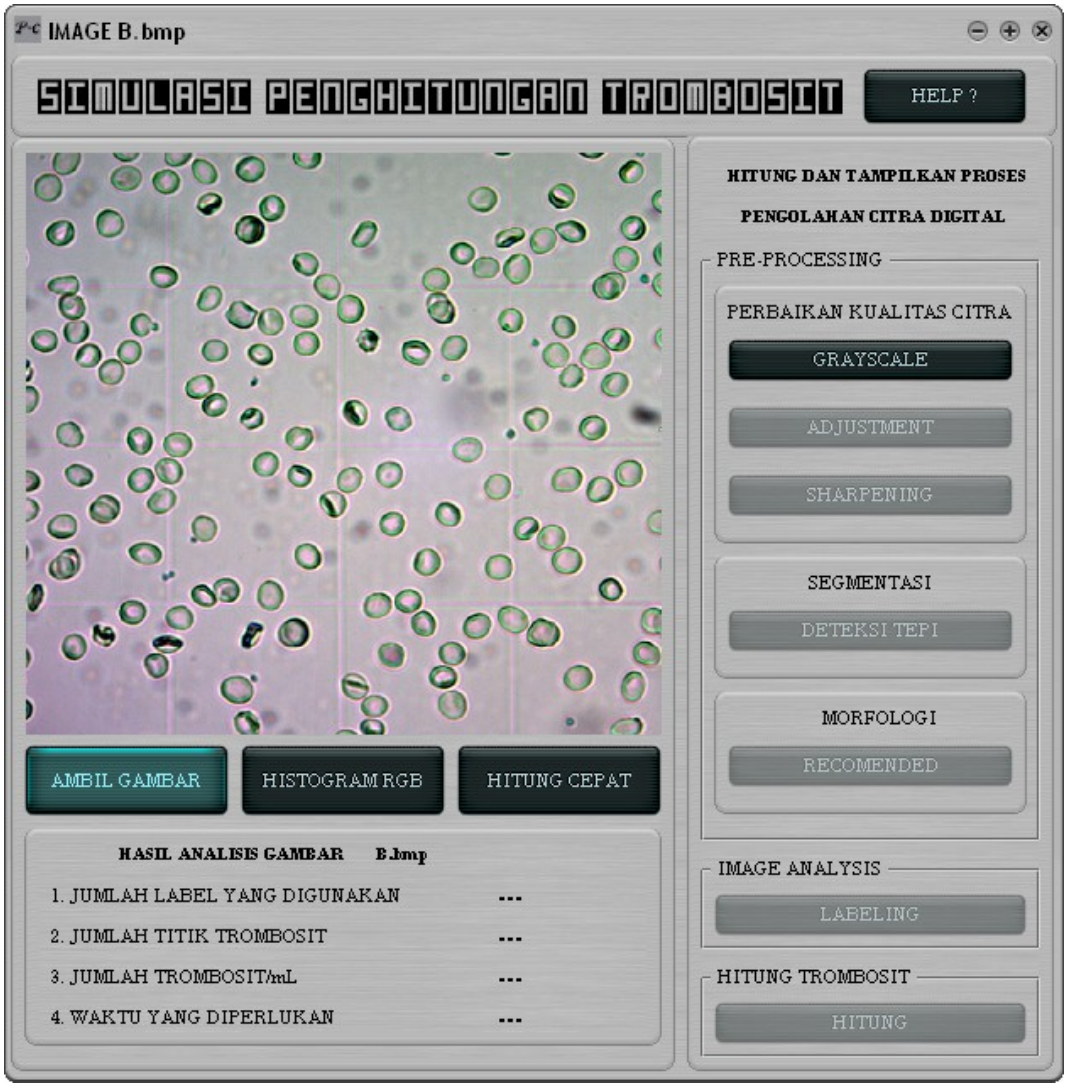
4.2 Pengujian

Pengujian ini dilakukan untuk mengetahui keberhasilan aplikasi dalam menghitung jumlah trombosit dalam citra. Pengujian dilakukan dengan cara menghitung jumlah trombosit pada 12 citra input. Adapun ketentuan yang digunakan pada pengujian ini adalah :

1. Operasi yang direkomendasikan yaitu 1 kali dilasi, 2 kali closing dan 1 kali opening.
2. Rata-rata luas objek yang dipetakan menjadi objek adalah 75 sampai dengan 120 piksel.

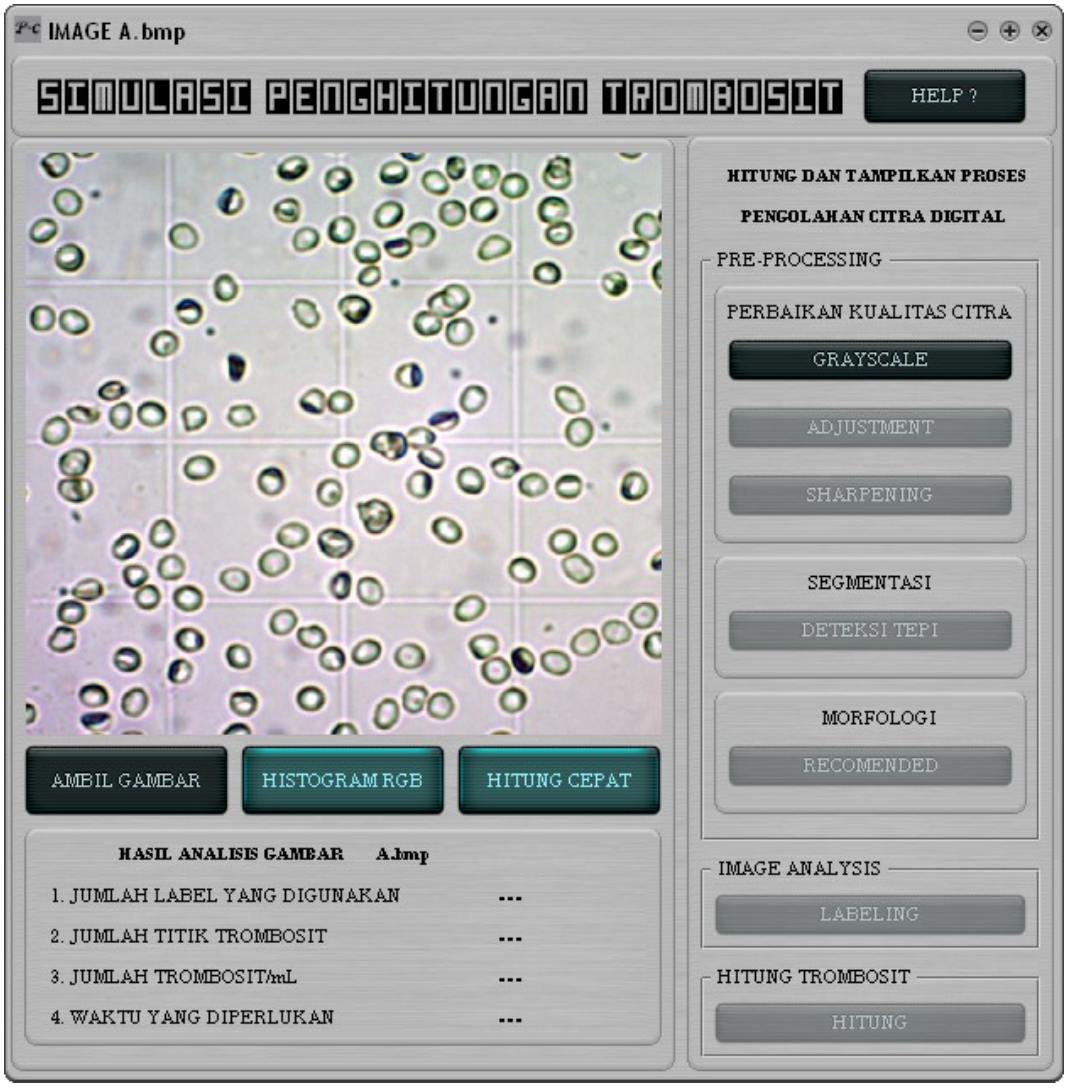
Berikut adalah langkah-langkah melakukan pengujian penghitungan trombosit :

1. Input citra dengan menekan tombol ‘AMBIL GAMBAR’.



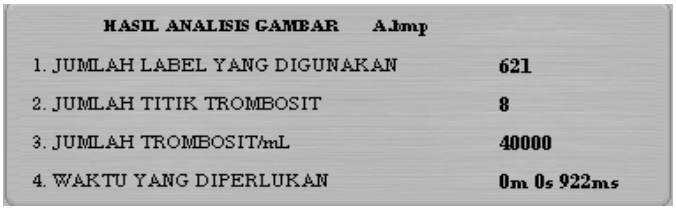
Gambar 4.2 Input citra *.RGB*

2. Tekan tombol '*HITUNG CEPAT*' untuk mengeksekusi semua fungsi yang ada.



Gambar 4.3 Tekan tombol '*HITUNG CEPAT*'.

3. Selanjutnya hasil analisis akan ditampilkan pada kolom di bagian bawah.



Gambar 4.4 Hasil analisis citra.

4. Langkah pertama sampai tiga diulangi pada pengujian berikutnya dengan meng-*input* 11 gambar trombosit yang lainnya.

Data hasil pengujian dapat dilihat pada Tabel 4.2.

Tabel 4.2 Hasil Pengujian Penghitungan Trombosit

No	Gambar	Jumlah titik trombosit hitung program	Jumlah titik trombosit hitung manual	Jumlah trombosit hitung program	Jumlah trombosit hitung manual	Persentase Keberhasilan (%)	Waktu penghitungan (detik)	Keterangan
1.	A	8	9	40.000	45.000	88,9	0,954	
2.	B	10	11	50.000	55.000	90,9	0,985	-
3.	C	17	17	85.000	85.000	100	1,31	-
4.	D	9	8	45.000	40.000	88,9	1,453	-
5.	E	7	7	35.000	35.000	100	1,250	-
6.	F	15	15	75.000	75.000	100	1,265	-
7.	G	17	21	85.000	105.000	80,9	1,265	Terdapat ukuran trombosit tidak normal, patahan objek, noise
8.	H	22	23	110.000	115.000	95,65	1,516	-
9.	I	24	20	120.000	100.000	83,3	1,250	Terdapat ukuran trombosit tidak normal, patahan objek, noise
10.	J	14	14	70.000	70.000	100	0,922	-
11.	K	22	20	110.000	100.000	90,9	1,15	-
12.	L	8	8	40.000	40.000	100	1,219	

Keterangan penghitungan :

- Jumlah titik trombosit hitung program merupakan jumlah titik trombosit yang dihitung otomatis oleh aplikasi sedangkan jumlah titik trombosit hitung manual merupakan jumlah trombosit dalam citra yang dihitung oleh tenaga medis.
- Citra *input* pada progam menggunakan 16 kotak kamar hitung. Jumlah trombosit hitung program atau manual diperoleh dari persamaan

$$Jumlah_{trombosit}/mm^3 = \left(\frac{Jumlah\ titik\ trombosit}{Jumlah\ kotak\ pandang} \right) * 80 * 1000\ mm^3$$

Contoh penghitungan jumlah trombosit hitung program gambar A :

$$Jumlah_{trombosit}/mm^3 = \left(\frac{8}{16} \right) * 80 * 1000\ mm^3 = 40.000\ mm^3$$

Contoh penghitungan jumlah trombosit hitung manual gambar A :

$$Jumlah_{trombosit}/mm^3 = \left(\frac{9}{16} \right) * 80 * 1000\ mm^3 = 45.000\ mm^3$$

- Jumlah titik trombosit penghitungan manual diasumsikan sebagai penghitungan dengan tingkat ketelitian 100 persen. Persentase keberhasilan diperoleh dari:

$$Persentase\ keberhasilan = \left(\frac{Jumlah\ titik\ trombosit\ hitung\ manual - (x)}{Jumlah\ titik\ trombosit\ hitung\ manual} \right) * 100\,\%$$

Jika jumlah titik trombosit hitung program lebih besar dari jumlah titik trombosit hitung manual maka

$$x = jumlah\ titik\ trombosit\ hitung\ program - jumlah\ titik\ trombosit\ hitung\ manual$$

Jika jumlah titik trombosit hitung manual lebih besar dari jumlah titik trombosit hitung program maka

$$x = jumlah\ titik\ trombosit\ hitung\ manual - jumlah\ titik\ trombosit\ hitung\ program$$

Contoh penghitungan persentase keberhasilan pada gambar A adalah :

$$Persentase\ keberhasilan = \left(\frac{9 - (1)}{9} \right) * 100\,\% = 88{,}9\,\%$$

Karena jumlah titik trombosit hitung manual lebih besar dari jumlah titik trombosit hitung program maka

$$x = jumlah\ titik\ trombosit\ hitung\ manual - jumlah\ titik\ trombosit\ hitung\ program$$

$$x = 9 - 8 = 1$$

- Rata-rata persentase keberhasilan perhitungan pada 12 gambar diperoleh dari persamaan

$$Rata-rata\ persentase\ keberhasilan = \frac{\sum_{i=1}^{12} Persentase_{keberhasilan}[i]}{12}$$

$$¿ \frac{(88{,}9+90{,}9+100+88{,}9+100+100+80{,}9+95{,}65+83{,}3+100+90{,}9+100)\%}{12}$$

$$¿ 93{,}2875\%$$

Berdasarkan hasil pengujian, rata-rata persentase keberhasilan aplikasi dalam menghitung trombosit pada 12 gambar adalah 93,2875%

- Rata-rata waktu yang dibutuhkan untuk menghitung trombosit pada 12 gambar diperoleh dari persamaan

$$Rata-rata\ waktu\ perhitungan\ trombosit\ dengan\ program = \frac{\sum_{i=1}^{12} waktu_{penghitungan}[i]}{12}$$

$$¿ \frac{(0{,}954+0{,}985+1{,}31+1{,}453+1{,}250+1{,}265+1{,}265+1{,}516+1{,}250+0{,}922+1{,}15+1{,}219)\ detik}{12}$$

$$¿ 1{,}2115\ detik$$

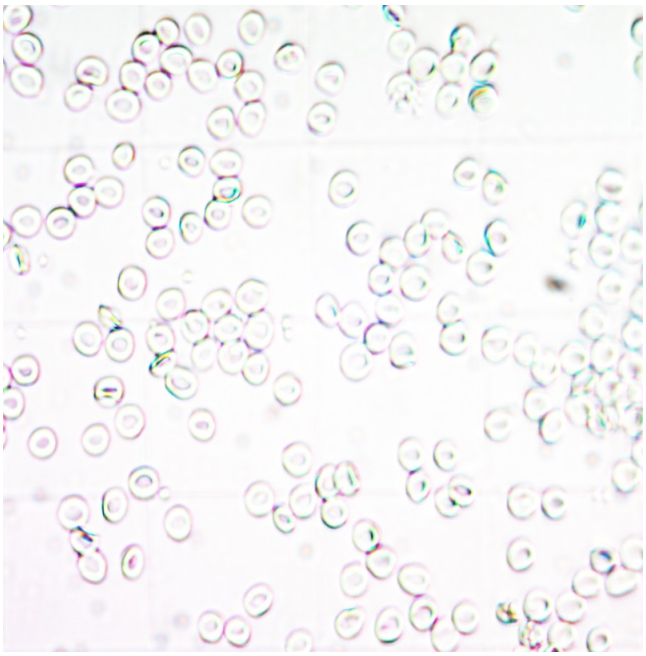
maka rata-rata waktu yang dibutuhkan untuk menghitung trombosit adalah 1,2115 detik.

4.3. Analisis Hasil Perancangan dan Pengujian Aplikasi

Berikut ini adalah analisis hasil perancangan dan pengujian aplikasi *platelet count* antara lain :

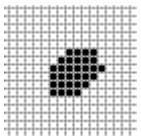
1. Rata-rata persentase keberhasilan aplikasi dalam menghitung trombosit pada 12 gambar adalah 93,2875%. Beberapa hal yang menyebabkan menurunnya tingkat keberhasilan program antara lain :
   a. Kualitas citra yang sangat rendah sebagai akibat kesalahan pada saat akusisi dapat menyebabkan banyak citra hasil akusisi tidak dapat digunakan. Kesalahan tingkat akusisi yang sangat berpengaruh pada aplikasi adalah pencahayaan. Pencahayaan yang tinggi dapat menyebabkan objek dalam gambar memiliki nilai piksel yang tinggi,

sedangkan nilai piksel yang tepat dipetakan menjadi objek adalah piksel yang bernilai rendah.



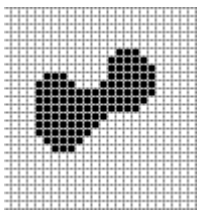
Gambar 4.5 Citra dengan kualitas rendah akibat kesalahan akusisi.

b. Trombosit yang ukurannya tidak normal. Trombosit yang terlalu kecil atau terlalu besar tidak akan dipetakan menjadi objek.



Gambar 4.6 Trombosit dengan luas dibawah rata-rata

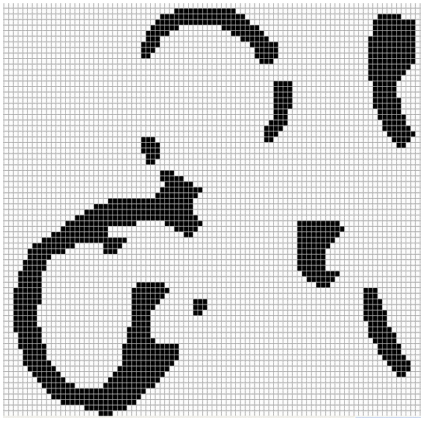
c. Trombosit yang berhimpit. Apabila luas trombosit yang berhimpit tersebut masih dalam *range* luas objek, maka akan dihitung hanya 1. Jika luasnya tidak memenuhi maka tidak akan dihitung.



Gambar 4.7 Trombosit berhimpit

d. Semua objek yang sudah dilabeli dan memiliki luas dalam *range* luas trombosit, akan dipetakan sebagai objek. Hal ini menyebabkan *noise dan*

patahan objek lain yang memiliki luas dalam *range* luas trombosit akan dihitung.



Gambar 4.8 Patahan Objek lain yang memiliki kemungkinan dipetakan menjadi objek trombosit

2. Aplikasi dapat meningkatkan efisiensi waktu dalam perhitungan trombosit dimana rata-rata waktu yang dibutuhkan untuk menghitung trombosit pada 12 gambar adalah 1,2115 detik.
3. Aplikasi dapat melakukan proses pelabelan komponen pada gambar yang bukan citra darah namun masih berformat *BMP*.
4. Metode pelabelan komponen dapat digunakan untuk menghitung banyak objek dalam citra dan dapat pula digunakan sebagai data awal untuk menyeleksi citra berdasarkan ukuran.

BAB V
PENUTUP

5.1. Kesimpulan

Setelah dilakukan analisis dan pengujian terhadap Aplikasi penghitungan trombosit dengan metode pelabelan komponen ini, dapat disimpulkan bahwa:

1. Aplikasi penghitungan trombosit yang menerapkan metode pelabelan komponen dapat meningkatkan keakuratan penghitungan trombosit dengan rata-rata persentase keberhasilan pada 12 gambar adalah 93,2875%.
2. Aplikasi penghitungan trombosit dengan metode pelabelan komponen dapat meningkatkan efisiensi waktu dalam proses penghitungan.
3. Akusisi citra sangat mempengaruhi hasil penghitungan.
4. Aplikasi penghitungan trombosit dengan metode pelabelan komponen hanya dapat mengenali objek berdasarkan kisaran luas atau jumlah piksel.

5.2. Saran

Hal-hal yang menjadi saran dalam pengembangan Aplikasi penghitungan trombosit dengan metode pelabelan komponen ini agar menjadi lebih baik adalah sebagai berikut:

1. Akusisi sebaiknya dilakukan oleh tenaga medis yang telah berpengalaman dalam proses akusisi sehingga citra darah hasil akusisi memiliki kualitas yang baik.
2. Aplikasi penghitungan trombosit dengan metode pelabelan komponen dapat dijadikan referensi untuk pengolahan atau analisis citra tingkat lanjut.
3. Aplikasi penghitungan trombosit dengan metode pelabelan komponen diharapkan dapat dikembangkan untuk mengenali objek dengan metode lain sehingga semua objek dalam citra dapat dikenali (bukan hanya berdasarkan jumlah piksel objek) sehingga meningkatkan keakuratan penghitungan dan pengenalan trombosit.