



INSTITUT TEKNOLOGI BANDUNG

PROGRAM STUDI TEKNIK ELEKTRO

JALAN GANESHA NO. 10 Gedung Labtek V Lantai 2 ☎ (022)2508135-36, 📠 (022)2500940
BANDUNG 40132

Dokumentasi Produk Tugas Akhir

Lembar Sampul Dokumen

Judul Dokumen **TUGAS AKHIR TEKNIK ELEKTRO:**
*Pengembangan Sistem Deteksi Dini Retinopathy
Diabetes dengan Smartphone*

Jenis Dokumen **IMPLEMENTASI**

Catatan: Dokumen ini dikendalikan penyebarannya oleh Prodi Teknik Elektro ITB

Nomor Dokumen **B400-01-TA1617.01.068**

Nomor Revisi **Versi 01**

Nama File **B400**

Tanggal Penerbitan **14 June 2017**

Unit Penerbit **Prodi Teknik Elektro - ITB**

Jumlah Halaman **47** (termasuk lembar sampul ini)

Data Pemeriksaan dan Persetujuan				
Ditulis Oleh	Nama	Amalia Lupitasari	Jabatan	Anggota
	Tanggal	10 Maret 2017	Tanda Tangan	
	Nama	Lutfi Bukhari	Jabatan	Anggota
	Tanggal	10 Maret 2017	Tanda Tangan	
	Nama	Yongky Purnomo	Jabatan	Anggota
	Tanggal	10 Maret 2017	Tanda Tangan	
Diperiksa Oleh	Nama	Dr. Hasballah Zakaria	Jabatan	Dosen Pembimbing
	Tanggal	10 Maret 2017	Tanda Tangan	
Disetujui Oleh	Nama	Dr. Hasballah Zakaria	Jabatan	Dosen Pembimbing
	Tanggal	10 Maret 2017	Tanda Tangan	

DAFTAR ISI

DAFTAR ISI.....	2
CATATAN SEJARAH PERBAIKAN DOKUMEN.....	3
1 PENGANTAR	4
1.1 RINGKASAN ISI DOKUMEN	4
1.2 TUJUAN PENULISAN DAN APLIKASI/KEGUNAAN DOKUMEN.....	4
1.3 REFERENSI	4
1.4 DAFTAR SINGKATAN.....	4
2 ISI DOKUMEN	5
2.1 IMPLEMENTASI HARDWARE	5
2.1.1 Pengimplementasian <i>Casing</i>	5
2.1.2 Implementasi Lensa Terhadap Pengambilan Citra.....	6
2.2 PENGOLAHAN CITRA RETINA.....	12
2.2.1 Diagram Blok Pengolahan Citra Retina.....	12
2.2.2 Algoritma Pengolahan Citra.....	16
2.2.2.1 <i>Pre-Processing</i>	16
2.2.2.2 Deteksi Pembuluh Darah.....	18
2.2.2.3 Deteksi <i>Optical Disk</i>	22
2.2.2.4 Deteksi Mikroaneurisme	23
2.2.2.5 Deteksi <i>Hemorrhage</i>	24
2.2.2.6 Klasifikasi.....	25
2.3 <i>GRAPHICAL USER INTERFACE (GUI) APLIKASI</i>	28
2.3.1 Tampilan Menu ' <i>Sign In</i> '	28
2.3.2 Tampilan Menu ' <i>Sign Up</i> '	31
2.3.3 Tampilan Menu ' <i>Utama</i> '	34
2.3.4 Tampilan Menu ' <i>Retinal Screening - Biodata</i> '	36
2.3.5 Tampilan Menu ' <i>Retinal Screening - Camera</i> '	38
2.3.6 Tampilan Menu ' <i>Log</i> '	46

Catatan Sejarah Perbaikan Dokumen

Tabel I – Catatan Sejarah Perbaikan

VERSI, TGL, OLEH	PERBAIKAN
1, 5 Mei 2017, Yongky Purnomo	Implementasi <i>Hardware</i>
1, 5 Mei 2017, Amalia Lupitasari	Pengolahan Citra Retina
1, 5 Mei 2017, Lutfi Bukhari	GUI Aplikasi

1 Pengantar

1.1 RINGKASAN ISI DOKUMEN

Dokumen B400 ini berisi implementasi dari desain yang telah dirancang sebelumnya. Implementasi ini mencakup *hardware*, pengolahan citra retina, serta aplikasi pada *smartphone* Android. Implementasi *hardware* melingkupi perhitungan terkait lensa yang digunakan beserta jarak yang sesuai serta pembuatan *hardware* itu sendiri. Untuk bagian pengolahan citra retina, implementasi yang dimaksud berupa algoritma yang digunakan dalam mengolah citra. Karena belum berhasil didapatkan gambar yang sesuai dalam implementasi *hardware*, gambar yang diolah dalam implementasi pengolahan citra retina diambil dari *database* Kaggle. Sedangkan untuk aplikasi, implementasi mencakup algoritma yang digunakan dalam pembuatan aplikasi serta tampilan sementara dari aplikasi tersebut.

1.2 TUJUAN PENULISAN DAN APLIKASI/KEGUNAAN DOKUMEN

Tujuan dari penulisan dokumen ini dijelaskan sebagai berikut:

- Menjelaskan implementasi yang telah dilakukan dalam membuat pengembangan sistem deteksi penyakit *retinopathy* diabetes dengan *smartphone*.
- Sebagai evaluasi terhadap desain yang telah dirancang pada dokumen sebelumnya.
- Pemenuhan persyaratan kelulusan mata kuliah EL4091 Tugas Akhir II (*Capstone Design*).

1.3 REFERENSI

- [1] Vidyasari, Rahmanita. *Algoritma Vessel Enhancement pada Filter Mikroaneurisma Citra Retina Digital untuk Klasifikasi Retinopati Diabetes Nonproliferatif*. Institut Teknologi Bandung, 2011, Bandung.
- [2] Choukikar, Prashant, Arun Kumar Patel, Ravi Shankar Mishra. *Segmenting the Optic Disc in Retinal Images Using Thresholding*. Internal Journal of Computer Applications, vol. 94 – no.11, May 2014.
- [3] Solanki, Mohit Singh. *Diabetic Retinopathy Detection Using Eye Images*. Indian Institute of Technology Kanpur, 2015.
- [4] *Database citra fundus retina*, <https://www.kaggle.com/c/diabetic-retinopathy-detection/data>

1.4 DAFTAR SINGKATAN

Berikut adalah tabel referensi dari singkatan-singkatan yang digunakan dalam dokumen ini.

Tabel II – Daftar Singkatan

SINGKATAN	ARTI
LED	<i>Light Emitting Diode</i>
CLAHE	<i>Contrast Limited Adaptive Histogram Equalization</i>
GUI	<i>Graphical User Interface</i>

2 ISI DOKUMEN

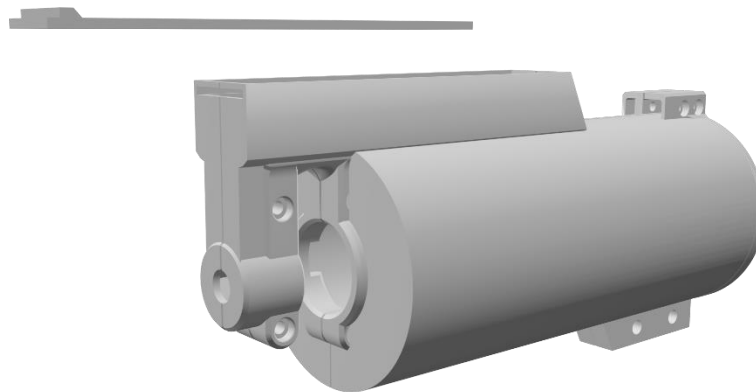
2.1 Implementasi Hardware

Berdasarkan desain-desain yang telah dibahas didokumen sebelumnya, pada dokumen ini akan dijelaskan pengimplementasian hardware perancangan. Pengimplementasian yang dilakukan diantaranya pengimplementasian casing, lensa, dan rangkaian.

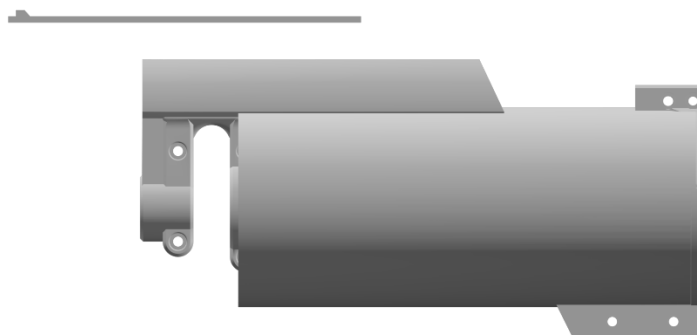
2.1.1 Pengimplementasian Casing

Desain-desain yang telah dirancang selanjutnya masuk kedalam proses pencetakan. Terdapat 2 jenis pencetakan objek yang sering digunakan yaitu bahan plastik 3D atau akrilik. Terdapat kelebihan dan kekurangan dari kedua jenis tersebut. Untuk desain menggunakan bahan akrilik kelebihannya adalah harga yang lebih murah. Sedangkan kekurangannya adalah sulit untuk pencetakan objek lengkung. 3D *printing* memiliki keunggulan dapat mengaplikasikan segala bentuk desain namun dengan harga yang cukup mahal. Dari desain yang telah dirancang, lebih banyak didominasi oleh bentuk lengkungan, sehingga jenis pencetakan 3D lebih dipilih dibandingkan dengan cetak akrilik. Takar satuan yang digunakan dalam pencetakan 3D adalah per gram dengan rentang harga Rp 3.500 – Rp 5.000.

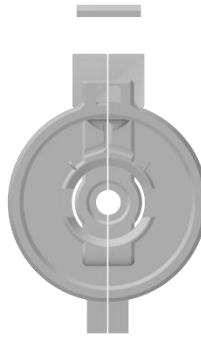
Berikut adalah implementasi dari desain *casing* yang dicetak menggunakan cetak 3D:



Gambar 2.1 *Casing Hardware*



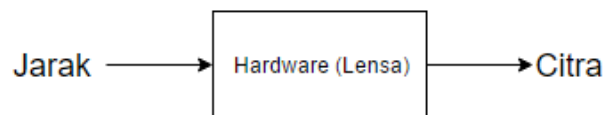
Gambar 2.2 *Casing Tampak Samping*



Gambar 2.3 *Casing* Tampak Depan

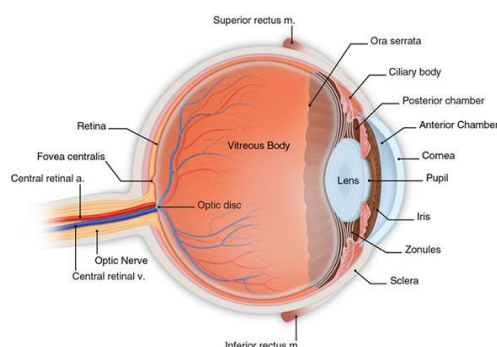
2.1.2 Implementasi Lensa Terhadap Pengambilan Citra

Pada bagian ini akan dilakukan pengimplementasian dari lensa fundus. Terdapat dua jenis lensa yang akan diuji dengan perbedaan kekuatan lensa. Lensa yang akan dibandingkan adalah lensa 20D dan 78D. Input dari pengujian ini adalah perhitungan jarak antara kamera *smartphone* ke lensa dan lensa ke objek. Sedangkan output yang akan diamati adalah citra yang didapatkan oleh *smartphone*.



Gambar 2.4 Diagram Blok Pengujian Lensa

Objek pengujian lensa terhadap citra dilakukan pada dua objek. Objek pertama adalah manusia sedangkan objek yang kedua adalah model mata. Model mata yang dibuat mengambil karakteristik bagian mata yang berkenaan dengan pengambilan data retinopathy diabetes. Karakteristik pertama bagian mata yang berkenaan dengan pengambilan citra retina adalah lensa mata. Lensa mata merupakan bagian pada mata yang berfungsi untuk meneruskan cahaya ke retina mata. Lensa mata manusia mempunyai kekuatan lensa sebesar 60 dioptri. Selain itu untuk membuat *modeling*, parameter lain yang perlu diperhatikan adalah jarak antara lensa dengan bidang tangkap adalah 2 cm. Parameter-parameter tersebut kemudian diimplementasikan pada sebuah model 3D yang ditunjukkan gambar 2.6 dan 2.7:



Gambar 2.5 Morfologi Mata Manusia

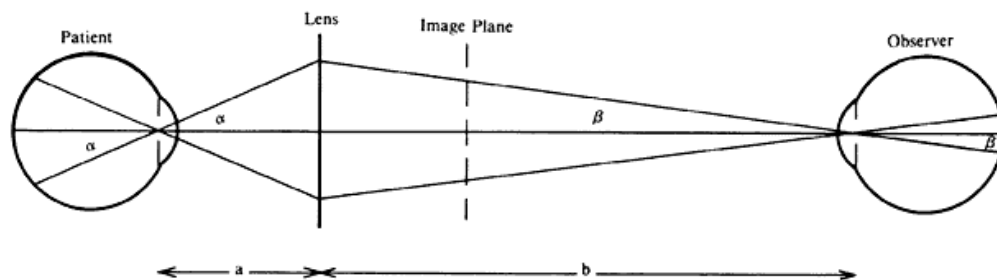


Gambar 2.6 Model Mata Tampak Serong



Gambar 2.7 Model Mata Tampak depan

Perhitungan jarak menggunakan perhitungan lensa tipis, dimana tidak terdapat pengaruh indeks bias medium dan indeks bias lensa.



Gambar 2.8 Ilustrasi Perhitungan Lensa yang digunakan

Persamaan lensa tipis yang digunakan adalah:

$$\frac{1}{a} + \frac{1}{b} = \frac{1}{f}$$

Keterangan :

a : Jarak antara mata dengan lensa fundus

b : Jarak antara lensa dengan

f : Jarak fokus lensa

Dari persamaan diatas, nilai yang diketahui hanyalah jarak fokus lensa. Nilai tersebut didapatkan dari persamaan kekuatan lensa. Persamaan kekuatan lensa:

$$P = \frac{1}{f}$$

Keterangan :

P : Kekuatan lensa (dioptri)

f : jarak fokus lensa (meter)

Perhitungan dilakukan dengan menentukan salah satu jarak menjadi sebuah nilai tetap. Selanjutnya dengan menggunakan persamaan yang ada maka kita dapat menentukan jarak lainnya. Penentuan nilai harus bersesuaian dengan prinsip-prinsip lensa yang digunakan. Sebagaimana yang telah dibahas pada dokumen sebelumnya, lensa yang digunakan

merupakan lensa bikonveks (konvergen). Sifat-sifat lensa konvergen yang sangat berpengaruh dalam implementasi ini adalah sifat bayangan yang dihasilkan berupa diperbesar pada peletakan di titik fokus, di antara fokus dengan jari-jari, dan di antara titik pusat dengan titik fokus.

2.1.2.1 Lensa 20D

Lensa 20D merupakan jenis lensa fundus dengan dioptri terkecil yang sering digunakan dalam oftalmologi. Lensa 20D memiliki spesifikasi sebagai berikut:

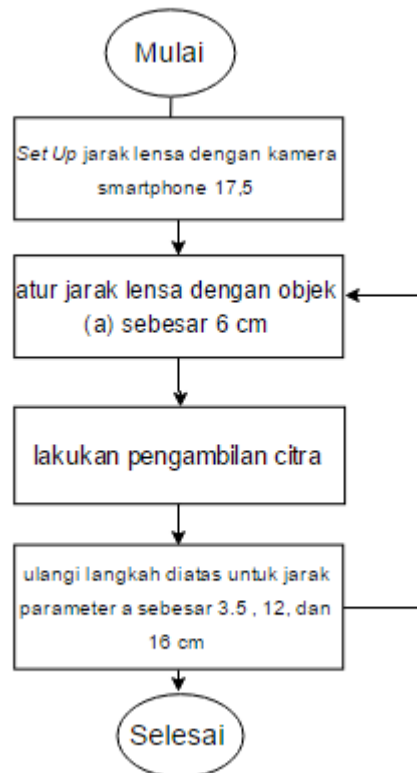
<i>Lens</i>	<i>20D</i>
<i>Field of View</i>	46°/60°
<i>Magnification</i>	3.13x
<i>Laser Spot</i>	0.32x
<i>Working Distance</i>	50 mm

Tabel 2.1 Spesifikasi 20D

Perhitungan dimulai dengan penentuan rentang jarak salah satu parameter agar prinsip lensa terpenuhi. Jika kita ingin mendapatkan bayangan diperbesar maka kita harus meletakkan benda pada rentang jarak 5 – 10 cm. Berdasarkan perhitungan yang telah dilakukan sebelumnya maka rentang jarak untuk parameter b adalah 11 – 16 cm, dimana semakin tinggi nilai parameter b maka nilai parameter a akan semakin kecil. Pada perancangan ini nilai yang ditentukan untuk parameter a adalah 7 cm. Pertimbangan yang diambil untuk menguatkan penentuan nilai ini adalah jarak kenyamanan untuk melakukan pemeriksaan kepada pasien. Jarak ini termasuk jarak yang tidak terlalu jauh dan tidak terlalu dekat dengan mata pasien. Perhitungan yang dilakukan adalah sebagai berikut:


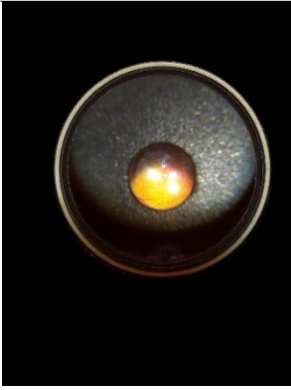
$$\begin{aligned}
 P &= P_a + P_b & P_b &= 20 - 14.28 \\
 P_a &= \frac{100 \text{ cm}}{a} & P_b &= 5,72 \text{ D} \\
 P_a &= \frac{100}{7} = 14,28 \text{ D} & P_b &= \frac{100}{b} \\
 P_b &= P - P_a & b &= \frac{100}{P_b} \\
 & & b &= \frac{100}{5,72} = 17,48 \text{ cm}
 \end{aligned}$$

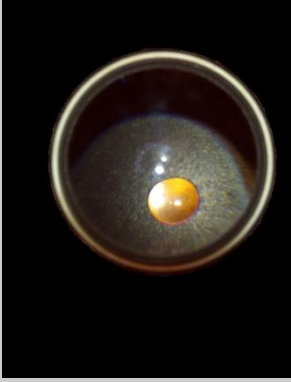

Setelah mendapatkan nilai-nilai parameter yang dibutuhkan, selanjutnya dilakukan pengujian pengaruh jarak dengan citra yang didapatkan. Pengujian dilakukan dengan melakukan koreksi-koreksi pada jarak parameter a guna mengamati citra yang didapatkan oleh kamera *smartphone*.



Gambar 2.9 *Flowchart* Pengujian Pengambilan Jarak Terhadap Pengambilan Citra Menggunakan Lensa 20D

Hasil data pengambilan data disajikan pada tabel 4.2:

No	Jarak	Citra model
1	6 cm	
2	3,5 cm	

3	12 cm	
4	16 cm	

Tabel 2.2 Pengaruh Jarak Terhadap Pengambilan Citra pada Lensa 20D

Hasil citra yang didapatkan berdasarkan jarak yang telah diperhitungkan. Dari koreksi-koreksi jarak objek dengan lensa terlihat bahwa ketika jarak antara lensa dengan objek diperpendek mendekati lensa maka yang terlihat hanya terjadi perbesaran lensa objek saja. sedangkan untuk jarak diluar jari-jari tidak terjadi pembesaran.

2.1.2.2 Lensa 78D

Pada bagian ini akan dijelaskan pengimplementasian desain lensa dengan menggunakan lensa fundus 78D. Spesifikasi lensa 78D sebagai berikut :

<i>Lens</i>	<i>78D</i>
<i>Field of View</i>	46°/60°
<i>Magnification</i>	3.13x
<i>Laser Spot</i>	0.32x
<i>Working Distance</i>	50 mm

Tabel 2.3 Spesifikasi 78D

Sama halnya dengan pengujian lensa 20D. Pertama kita akan meninjau rentang jarak fokus dengan jari-jari. Berdasarkan perhitungan yang telah dilakukan pada dokumen sebelumnya. Rentang nilai untuk parameter a lensa 78D adalah 1,4 – 2,4 cm. Pada pengujian kali ini dipilih besar jarak antara lensa fundus dengan objek adalah 1,6 cm. Sehingga akan didapatkan parameter jarak lainnya melalui persamaan berikut.

$$P = P_a + P_b$$

$$P_b = 78 - 62,5$$

$$P_b = 15,5 D$$

$$P_a = \frac{100 \text{ cm}}{a}$$

$$P_a = \frac{100}{1,6} = 62,5 \text{ D}$$



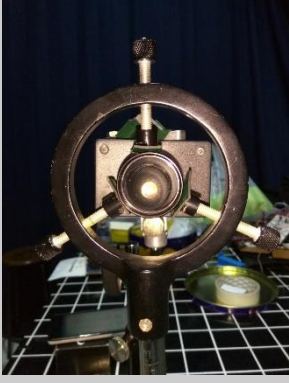
$$P_b = P - P_a$$

$$P_b = \frac{100}{b}$$

$$b = \frac{100}{P_b}$$

$$b = \frac{100}{15,5} = 6,45 \text{ cm}$$

Hasil Pengambilan data menggunakan lensa 78D

No	Jarak	Citra model
1	1,5 cm	
2	10,2 cm	
3	29,1 cm	

Tabel 2.4 Pengaruh Jarak Terhadap Pengambilan Citra pada Lensa 78D

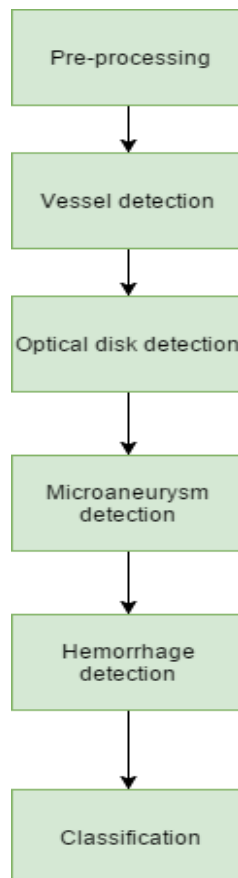
Perbedaan besar kekuatan lensa sangat berpengaruh pada perbesaran dan bidang pandang (*Field of View*). Terlihat bahwa citra model yang didapatkan lebih luas dibandingkan dengan lensa 20D. Namun besar dari citra yang didapatkan lebih kecil dibandingkan

dengan 20D. Hal ini disebabkan karena spesifikasi diameter lensa 78D lebih kecil dibandingkan dengan 20D.

2.2 Pengolahan Citra Retina

2.2.1 Diagram Blok Pengolahan Citra Retina

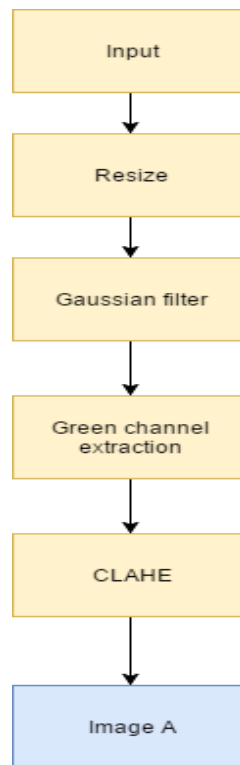
Pada dasarnya, pengolahan citra retina untuk deteksi dini penyakit retinopathy diabetes memanfaatkan dua fungsi utama, yaitu deteksi *optical disk* dan pembuluh darah. Deteksi kelainan pada retina seperti mikroaneurisma dan *hemorrhage* (*hemorrhages*) dilakukan dengan memanfaatkan kedua fungsi deteksi ini. Secara umum, proses keseluruhan deteksi dini retinopathy diabetes ini dapat dilihat pada gambar 2.10.



Gambar 2.10 Diagram Blok Deteksi Retinopathy Diabetes

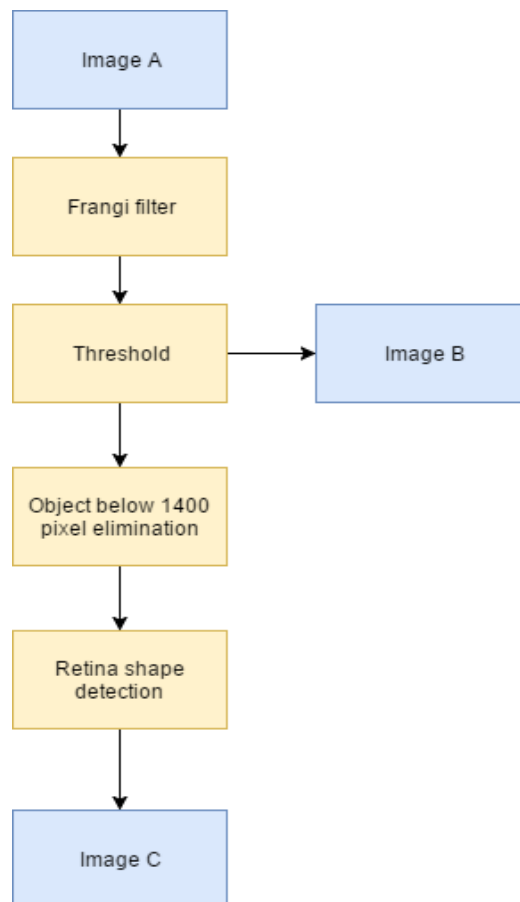
Adapun diagram blok dari masing-masing fungsi yang tertera pada gambar 2.10 adalah sebagai berikut:

- *Pre-processing*



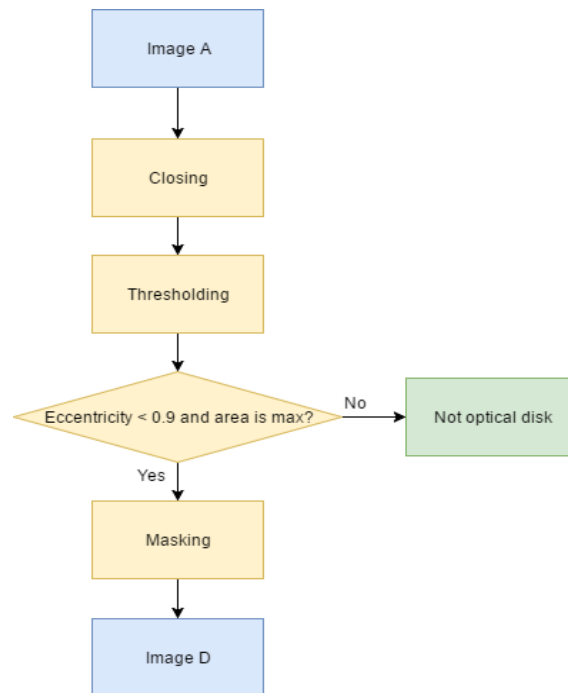
Gambar 2.11 Diagram Blok *Pre-Processing*

- Deteksi pembuluh darah



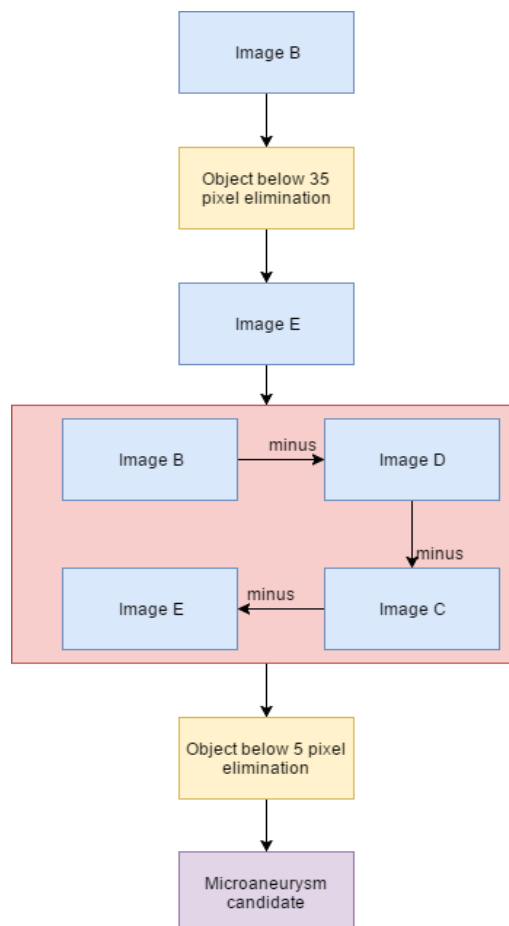
Gambar 2.12 Diagram Blok Deteksi Pembuluh Darah

- Deteksi *optical disk*



Gambar 2.13 Diagram Blok Deteksi *Optical Disk*

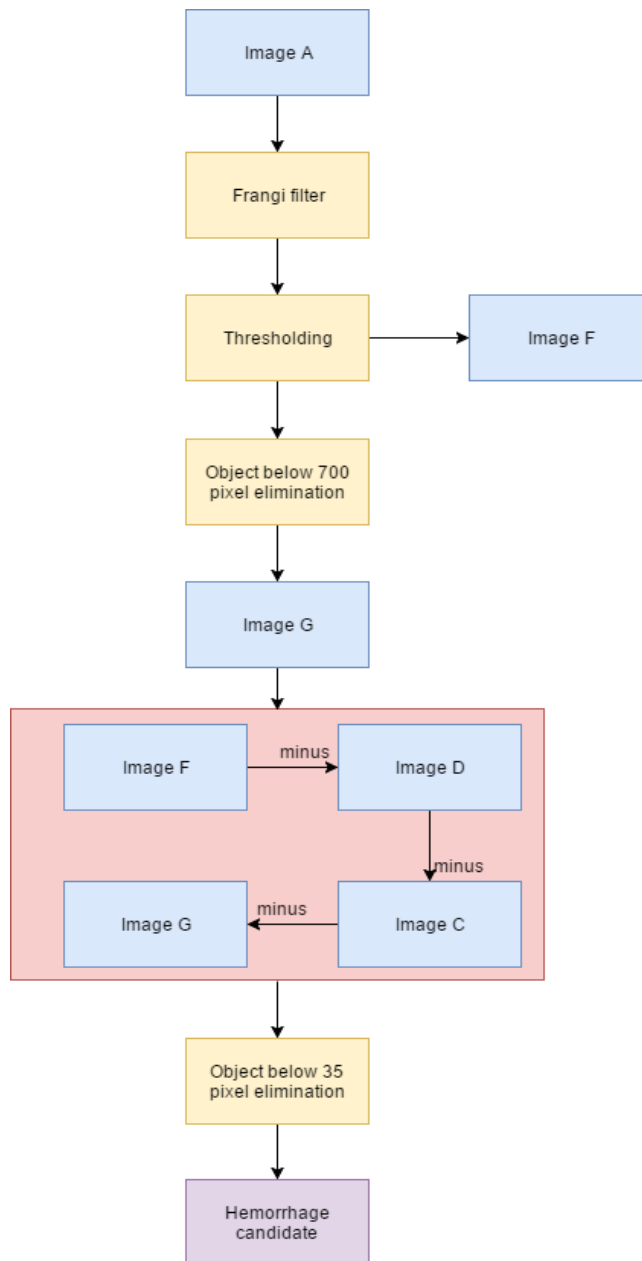
- Deteksi mikroaneurisme



Gambar 2.14 Diagram Blok Deteksi Mikroaneurisme

- Deteksi *hemorrhage*

Pada deteksi *hemorrhage*, filter Frangi yang digunakan memiliki parameter yang berbeda dari filter Frangi yang telah digunakan sebelumnya. Hal ini dikarenakan ukuran *hemorrhage* yang lebih besar dibandingkan mikroaneurisme.

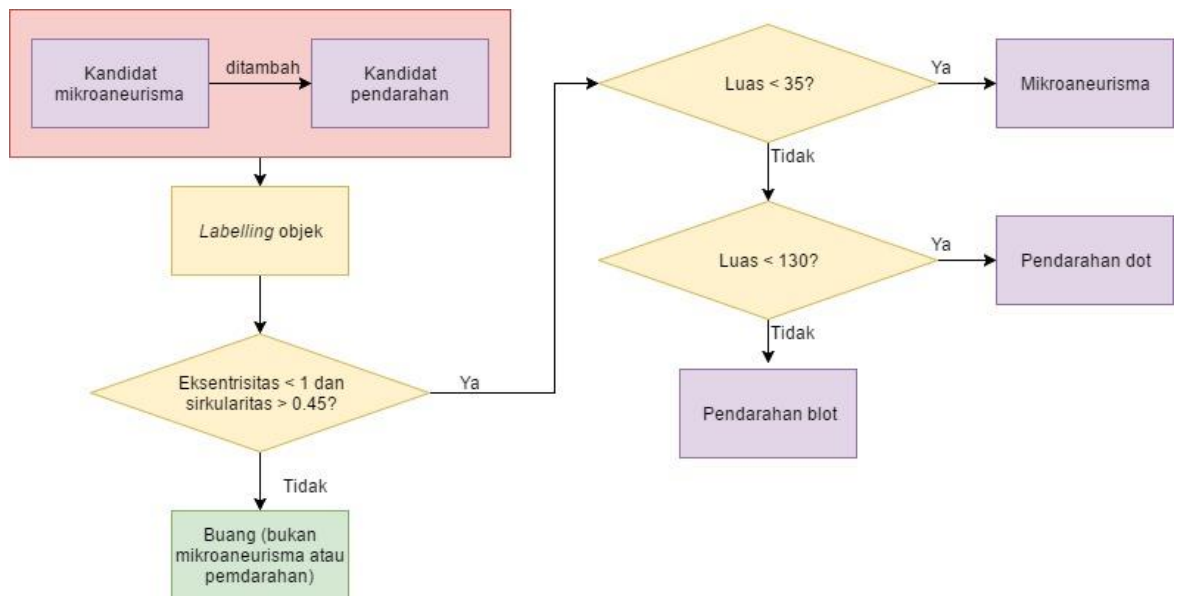


Gambar 2.15 Diagram Blok Deteksi *Hemorrhage*

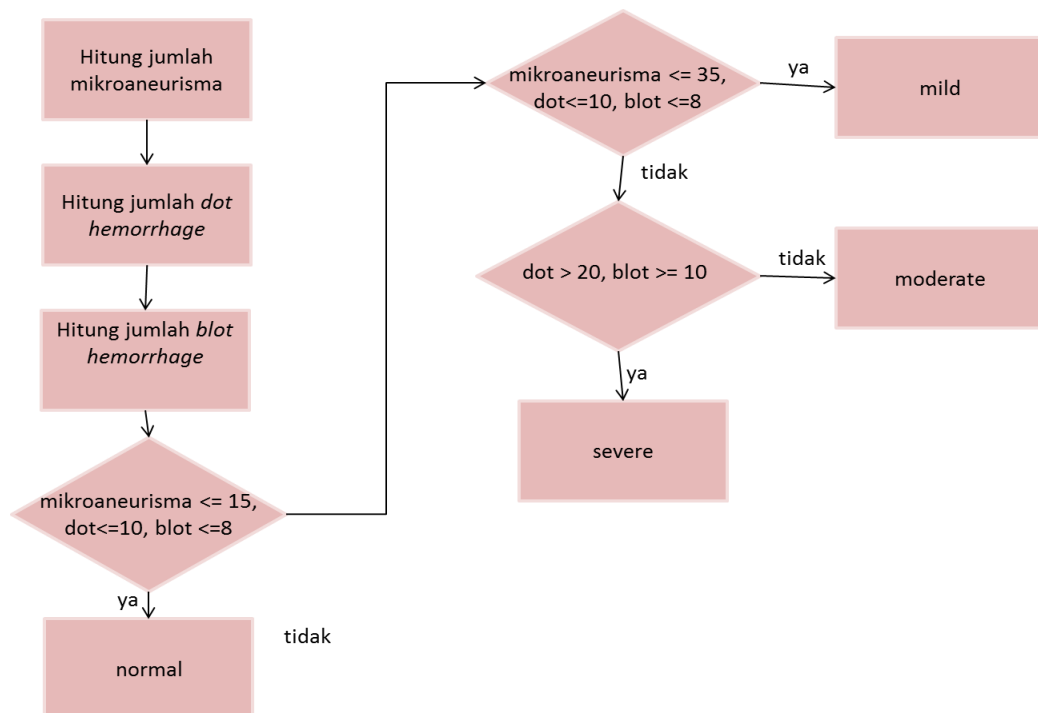
- Klasifikasi

Sebelum penyakit diklasifikasikan, terlebih dahulu dilakukan klasifikasi objek untuk membedakan mana mikroaneurisme dan *hemorrhage*. Setelah objek diklasifikasikan, dilakukan klasifikasi tingkat penyakit berdasarkan jumlah objek yang terdeteksi.

Diagram blok untuk masing-masing klasifikasi tersebut dapat dilihat pada gambar 2.16 dan 2.17 berikut.



Gambar 2.16 Diagram Blok Klasifikasi Objek



Gambar 2.17 Diagram Blok Klasifikasi Tingkat Penyakit

2.2.2 Algoritma Pengolahan Citra

Sebelum diterapkan pada OpenCV for Android, pengolahan citra retina dilakukan menggunakan Matlab R2016a sehingga hasil deteksi dapat dilihat terlebih dahulu. Pada implementasi pengolahan kali ini, citra yang diolah diambil dari *database* citra fundus retina Kaggle.

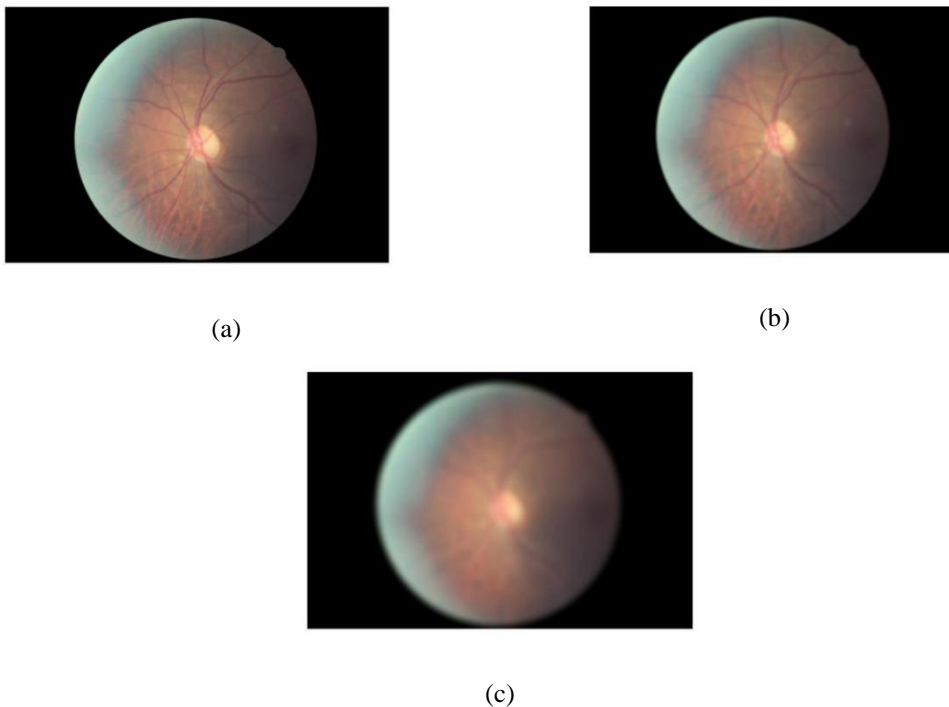
2.2.2.1 Pre-processing

Proses ini dilakukan untuk memperbaiki kualitas citra yang diolah. Pada tahap awal, dilakukan *resize* sehingga ukuran gambar lebih kecil untuk mengurangi beban komputasi. Setelah itu, diterapkan filter Gaussian untuk mengurangi *noise* pada gambar. Selanjutnya,

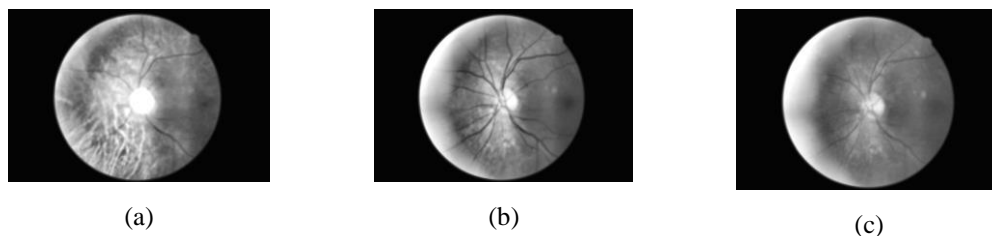
dilakukan ekstraksi pada kanal hijau. Kanal hijau dipilih karena hasil pemrosesan citra terlihat paling baik jika dilakukan pada kanal ini dibanding kanal-kanal lainnya. Kemudian diterapkan *Contrast Limited Adaptive Histogram Equalization* (CLAHE) sehingga dihasilkan citra yang lebih tajam. Potongan kode untuk bagian *pre-processing* ini adalah sebagai berikut.

```
Baca citra, ekstraksi kanal hijau, CLAHE
const = 0.35;
I = imresize(retinaimg, const);
imgblur = imgaussfilt(I, 5);
figure, imshow(imgblur)
green = imgblur(:, :, 2);
clahe = adapthisteq(green);
```

Penentuan parameter sigma Gaussian serta pemilihan kanal hijau untuk ekstraksi dapat dilihat pada gambar 2.18 dan gambar 2.19.



Gambar 2.18 Penerapan *Filter* Gaussian dengan Nilai Sigma: (a) 2; (b) 5; (c) 15. Untuk nilai sigma yang terlalu kecil efek filter tidak terlalu terlihat, sedangkan untuk nilai yang terlalu besar terlihat gambar terlalu *blur*.



Gambar 2.19 Peningkatan Kontras Citra pada Gambar dengan: (a) Ekstraksi Kanal Merah; (b) Ekstraksi Kanal Hijau; (c) Ekstraksi Kanal Biru. Terlihat kontras paling baik ketika dilakukan ekstraksi di kanal hijau.

2.2.2.2 Deteksi Pembuluh Darah

Pada citra hasil *pre-processing* diterapkan filter *vesselness* (Frangi Filter) kemudian dibinerisasi dengan *threshold* 0,1. Algoritma filter *vesselness* yang digunakan diambil dari Copyright © 2009, Dirk-Jan KroonAll, dengan pembahasan sebagai berikut.

Pada bagian awal, didefinisikan nilai parameter yang digunakan yaitu nilai *range* skala sigma (FrangiScaleRange), nilai rasio (FrangiScaleRatio), nilai β (FrangiBetaOne) dan nilai c (FrangiBetaTwo). Pada deteksi pembuluh darah, nilai parameter yang digunakan adalah sebagai berikut.

```
defaultoptions = struct('FrangiScaleRange', [1 4], 'FrangiScaleRatio',  
2, 'FrangiBetaOne', 0.5, 'FrangiBetaTwo', 15,  
'verbose',true,'BlackWhite',true);
```

Nilai β dan c didefinisikan sebagai berikut.

```
beta = 2*options.FrangiBetaOne^2;  
c = 2*options.FrangiBetaTwo^2;
```

Setelah itu, dibuat matriks nol untuk menampung hasil filter. Kemudian dilakukan pemrosesan berupa perhitungan secara berulang hasil filter *vesselness* pada setiap nilai sigma. Proses ini mencakup pembuatan matriks Hessian, penghitungan nilai Eigen, nilai anisotropi dan *structureness*, pendefinisian filter, serta penyimpanan hasil filter pada matriks nol yang telah didefinisikan sebelumnya. Nilai *structureness* didapat dari hasil akar penjumlahan kuadrat tiap nilai eigen, sedangkan nilai anisotropi didapat dari rasio antara kedua nilai eigen ($R_b = \lambda_1/\lambda_2$). Pada prinsipnya, untuk pembuluh darah, nilai $\lambda_1 < \lambda_2$, bahkan $\lambda_1 \approx 0$.

```
% Make matrices to store all filtered images  
ALLfiltered=zeros([size(I) length(sigmas)]);  
ALLangles=zeros([size(I) length(sigmas)]);  
  
% Make 2D hessian  
[Dxx,Dxy,Dyy] = Hessian2D(I,sigmas(i));  
  
% Correct for scale  
Dxx = (sigmas(i)^2)*Dxx;  
Dxy = (sigmas(i)^2)*Dxy;  
Dyy = (sigmas(i)^2)*Dyy;  
  
% Calculate (abs sorted) eigenvalues and vectors  
[Lambda2,Lambda1,Ix,Iy]=eig2image(Dxx,Dxy,Dyy);  
  
% Compute the direction of the minor eigenvector  
angles = atan2(Ix,Iy);  
  
% Compute some similarity measures  
Lambda1(Lambda1==0) = eps;  
Rb = (Lambda2./Lambda1).^2;  
S2 = Lambda1.^2 + Lambda2.^2;  
  
% Compute the output image  
Ifiltered = exp(-Rb/beta) .* (ones(size(I))-exp(-S2/c));  
  
% see pp. 45  
if(options.BlackWhite)  
    Ifiltered(Lambda1<0)=0;  
else
```

```

        Ifiltered(Lambda1>0)=0;
    end
    % store the results in 3D matrices
    ALLfiltered(:,:,i) = Ifiltered;
    ALLangles(:,:,i) = angles;

```

Setelah diterapkan filter Frangi, gambar dibinerisasi dengan *threshold* 0,1. Selanjutnya, dilakukan pembuangan terhadap bagian-bagian non pembuluh darah, yaitu bagian dengan ukuran kurang dari 1.400 pixel. Pembuangan ini dilakukan dengan cara *opening*.

```

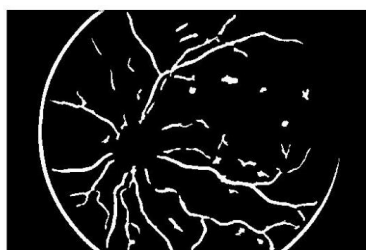
% Pembuangan bagian kecil (non pembuluh darah)
% Pembuangan bagian di bawah 1400 pixel
vessel_bw = bwareaopen(Ivessel_bw, 1400);

```

Dipilih angka 1400 piksel karena berdasarkan ujicoba, terlihat objek non-pembuluh darah yang ikut terdeteksi (seperti pendarahan) tidak muncul pada nilai ini. Untuk memudahkan pengamatan, gambar yang diujicoba adalah gambar retina dengan tingkat yang sangat parah.



Gambar 2.20 Gambar Retina yang Diujicoba



(a)



(b)



(c)



(d)

Gambar 2.21 Dilakukan *Opening* untuk Proses Deteksi Pembuluh Darah dengan Batas: (a) 300 piksel; (b) 500 piksel; (c) 700 piksel; (d) 1400 piksel

Tahap selanjutnya dalam proses ini adalah deteksi bentuk retina. Pertama-tama, dilakukan pembalikan terhadap citra yang telah diekstraksi kanal hijaunya, kemudian dilakukan binerisasi. Setelah itu, dicari bentuk bagian luar retina dengan cara dilasi menggunakan *structure element* berbentuk *disk* dengan jari-jari 15 piksel. Untuk mengetahui hasil deteksi pembuluh darah yang benar-benar bersih, hasil yang terlihat seperti pada gambar 2.21 dikurangi bentuk retina yang akan dicari. Proses ini dilakukan dengan cara *closing* (dilakukan dilasi dahulu diikuti dengan erosi) menggunakan *structure element* berukuran radius 2 piksel.

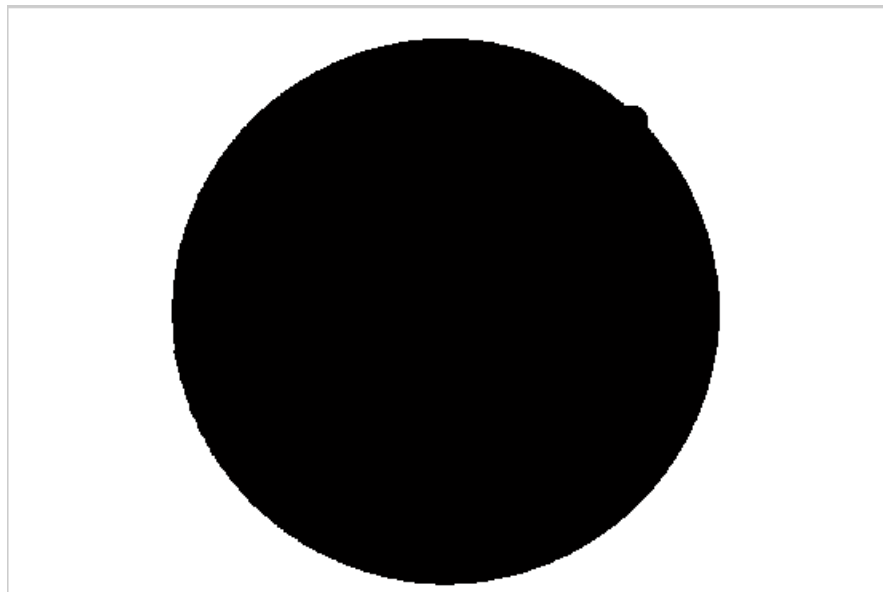
```
% Postprocessing
% Awal: komplemen kanal hijau, binerisasi
green_adjust = imadjust(green);
green_comp = imcomplement(green_adjust); % Pembalikan
green_bw = im2bw(green_comp, 0.999);

% Dilasi - mendapatkan bagian luar retina yg tdk ikut terdeteksi Frangi
% Filter
se = strel('disk', 15);
black_filled = imdilate(green_bw, se);

% Mendapatkan kandidat pembuluh darah
% Mengurangkan hasil pembuangan non pembuluh darah dengan hasil deteksi
% bentuk retina utk membuang lingkaran terluar retina
% Dilakukan dengan closing
vessel_min = imsubtract(vessel_bw, black_filled);
se = strel('disk', 2);
vessel_dilate = imdilate(vessel_min, se);
vessel_filled = imerode(vessel_dilate, se);

vessel_filled = (vessel_filled > 0);
```

Hasil deteksi bentuk retina dapat dilihat pada gambar 2.22 berikut.

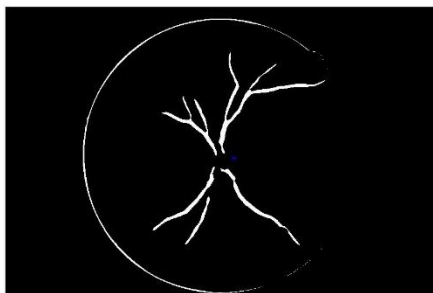


Gambar 2.22 Hasil Deteksi Bentuk Retina

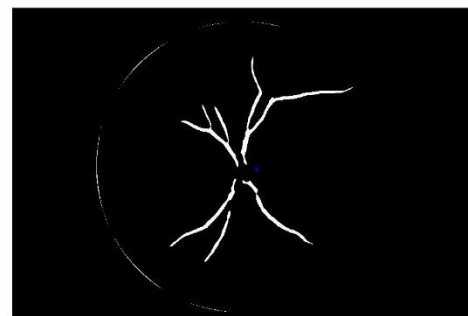
Penentuan radius *structuring element* dilakukan berdasarkan ujicoba terhadap gambar retina pada gambar 2.23.



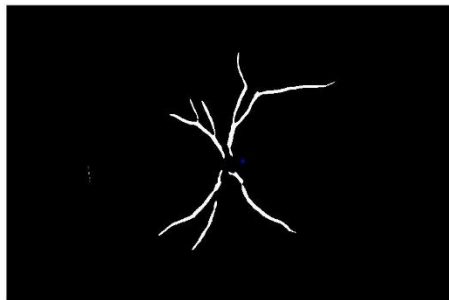
Gambar 2.23 Gambar Retina untuk Ujicoba



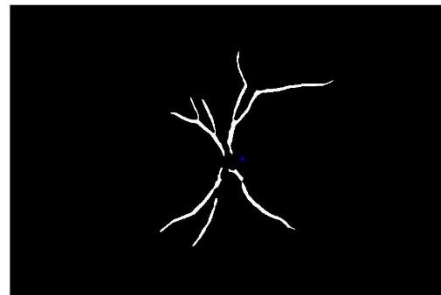
(a)



(b)



(c)



(d)

Gambar 2.24 Dilakukan *Closing* untuk Membuang Bagian Luar Retina dengan Radius *Structuring Element*: (a) 8 piksel; (b) 12 piksel; (c) 14 piksel; (d) 15 piksel

Dari keempat gambar tersebut, terlihat bahwa bentuk retina benar-benar hilang ketika *structuring element* berbentuk *disk* dengan radius 15 piksel. Bentuk retina ini dihilangkan untuk menyisakan pembuluh darah pada deteksi pembuluh darah.

2.2.2.3 Deteksi *Optical Disk*

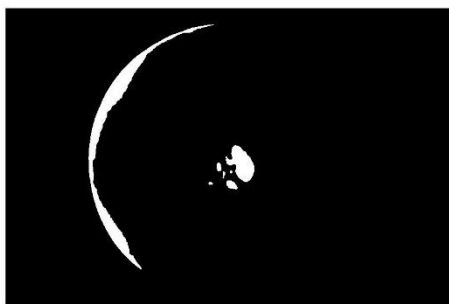
Untuk mendapatkan bagian *optical disc*, dilakukan proses *closing* kemudian *thresholding*. Pada proses *closing*, digunakan *structure element* berbentuk *disk* dengan jari-jari 10 piksel. Kemudian, dilakukan binerisasi terhadap gambar tersebut dengan nilai *threshold* 0,8. Hal ini dikarenakan bagian *optical disk* memiliki intensitas tertinggi sehingga perlu diambil bagian yang nilai intensitasnya mendekati 1.

```
%Closing dengan structure element berupa disk - jadi black and white
se = strel('disk',10);
c = imclose(c1, se);
bw = imbinarize(c, 0.80);
```

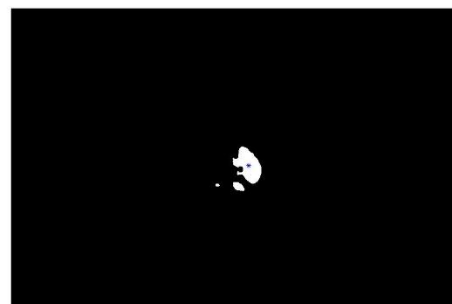
Setelah di-*threshold*, dilakukan pengambilan bentuk *optical disc* dengan mengecek eksentrisitasnya terlebih dahulu. Eksentrisitas merupakan nilai ukuran 'kebulatan' suatu bentuk. Eksentrisitas merupakan rasio antara jarak fokus elips dengan panjang sumbu utamanya. Nilai eksentrisitas 0 menunjukkan bentuk bulat sempurna sedangkan nilai 1 berbentuk garis. Nilai eksentrisitas di antara 0 dan 1 menunjukkan objek berbentuk elips. Berdasarkan perhitungan, nilai eksentrisitas mata manusia bernilai sekitar 0,9, sehingga daerah dengan nilai eksentrisitas di bawah nilai tersebut diasumsikan sebagai *optical disc*. Setelah itu, dicari bagian dengan luas terbesar.

```
%Mengukur eksentrisitas (ukuran kelengkungan) - mengambil optical disc
[binary_label, n] = bwlabel(bw);
stats = regionprops(binary_label, 'Eccentricity');
idx = find([stats.Eccentricity] < 0.9); %Optical disc memiliki
eksentrisitas 0.9
bw2 = ismember(binary_label,idx);
[binary_label, n] = bwlabel(bw2);
stats2 = regionprops(binary_label, 'Area');
[maxarea, index] = max([stats2.Area]);
OP = (binary_label==index); %menentukan optical disknya
```

Pengambilan nilai parameter dilakukan melalui ujicoba terhadap gambar retina dengan hasil seperti terlihat pada gambar 2.25 dan 2.26.

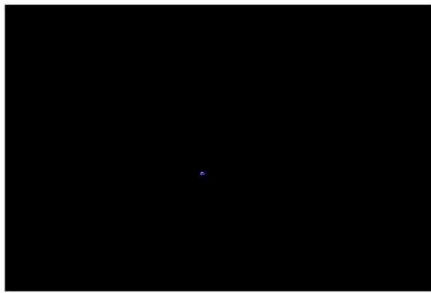


(a)

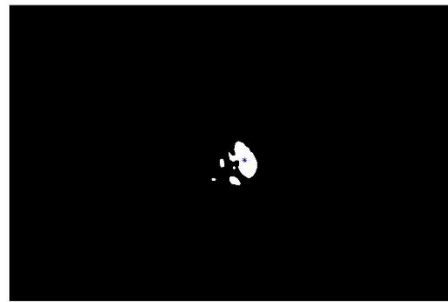


(b)

Gambar 2.25 Deteksi Daerah *Optical Disk* Setelah *Closing* dengan *Structuring Element* Berbentuk *Disk* dengan Radius: (a) 7 piksel; (b) 10 piksel.



(a)



(b)

Gambar 2.26 Deteksi Daerah *Optical Disk* Setelah Cek Eksentrisitas: (a) 0,8; (b) 0,9.

Setelah didapatkan daerah *optical disk*, digambarkan lingkaran pada daerah tersebut yang menunjukkan bahwa lingkaran tersebut adalah *optical disk*.

```
s = regionprops(OP, 'centroid'); %Mengecek centroid (titik tengah geometri)dari optical disk
centroids = cat(1, s.Centroid);
hold on
plot(centroids(:,1), centroids(:,2), 'b*')
hold off
a = round(centroids(:,1)); %Pembulatan
b = round(centroids(:,2));

%Untuk menggambar lingkaran
t = 0:pi/20:2*pi;
r=80;
xcc = r*cos(t)+a;
ycc = r*sin(t)+b;

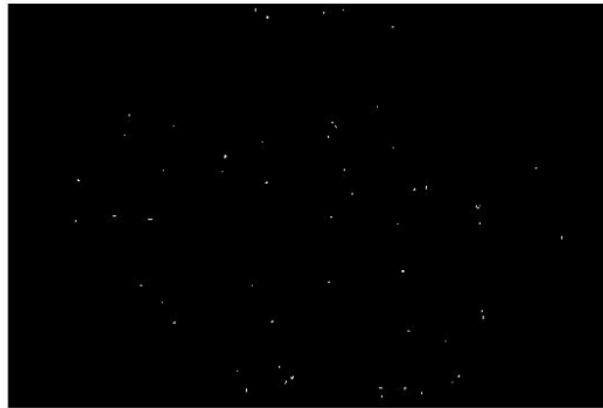
roimaskcc=poly2mask(double(xcc),double(ycc), size(gray,1),size(gray,2));
roi=gray;
roi(~roimaskcc)=0;
masking = imbinarize(roi, 0.5);
```

2.2.2.4 Deteksi Mikroaneurisma

Proses deteksi mikroaneurisma dilakukan dengan mengeliminasi bagian seukuran mikroaneurisma, yaitu di bawah 35 piksel. Nilai ini diambil mengikuti referensi pada tesis Rahmanita Vidyasari. Setelah itu, hasil *preprocessing* dikurangi hasil deteksi *optical disk*, bentuk retina, serta hasil penghilangan bagian di bawah 35 piksel tadi, sehingga pada akhirnya tersisa kandidat mikroaneurisma. Terakhir, hasil deteksi dieliminasi bagian yang dibawah 5 piksel untuk mengurangi *noise* yang sangat mungkin terdeteksi.

```
MA_open = bwareaopen(Ivessel_bw, 35);
MA_min = Ivessel_bw - black_filled - masking - MA_open;
MA_min = (MA_min > 0);
figure, imshow(MA_min)
MA_clean = bwareaopen(MA_min, 5);
```

Kandidat mikroaneurisma tergambar pada gambar 2.27 berikut.



Gambar 2.27 Kandidat Mikroaneurisma

2.2.2.5 Deteksi Hemorrhage

Tahap dalam deteksi *hemorrhage* sama persis dengan deteksi mikroaneurisma, hanya terdapat perbedaan pada nilai parameter filter *vesselness*, definisi filter, serta besarnya bagian yang dibuang. Pada deteksi *hemorrhage*, objek dieliminasi jika berukuran kurang 35 piksel. Hal ini dikarenakan definisi objek yaitu mikroaneurisma berarea kurang dari 35 piksel sedangkan pendarahan lebih besar dari itu (dibahas di bagian selanjutnya). Dilakukan cara yang sama dengan deteksi mikroaneurisma sehingga hasil yang tersisa di akhir adalah kandidat pendarahan dengan luas lebih dari 35 piksel.

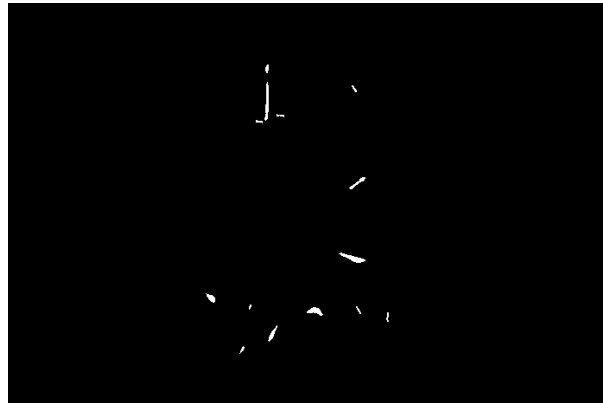
Parameter *filter* Frangi untuk deteksi pendarahan adalah sebagai berikut.

```
defaultoptions = struct('FrangiScaleRange', [4 10], 'FrangiScaleRatio',  
2, 'FrangiBetaOne', 0.5, 'FrangiBetaTwo', 15,  
'verbose',true,'BlackWhite',true);defaultoptions =  
struct('FrangiScaleRange', [3 8], 'FrangiScaleRatio', 1,  
'FrangiBetaOne', 0.5, 'FrangiBetaTwo', 15,  
'verbose',true,'BlackWhite',true);  
  
% Filter Hemorrhage  
  
Ifiltered = (Lambdal./beta) .* (ones(size(I))-exp(-S2/c));
```

Nilai parameter Frangi serta perumusan *filter* Frangi didasarkan pada tesis Rahmanita Vidyasari.

```
[hemorrhage, scalehemo, dirhemo] =  
FrangiFilter2D_hemorrhage(double(clahe));  
hemorrhage_bw = (hemorrhage > 1);  
  
HMA_open = bwareaopen(hemorrhage_bw, 700);  
HMA_min = hemorrhage_bw - black_filled - masking - HMA_open;  
HMA_min = (HMA_min > 0);  
HMA_clean = bwareaopen(HMA_min, 35);
```


Nilai 700 pada eliminasi bagian pendarahan didasarkan karena pada angka tersebut pendarahan masih terdeteksi (seperti terlihat pada gambar 2.21). Sedangkan eliminasi terhadap area kurang dari 35 piksel dikarenakan ukuran pendarahan yang lebih besar dari mikroaneurisma.



Gambar 2.28 Kandidat Pendarahan

2.2.2.6 Klasifikasi

Sebelum klasifikasi dilakukan, dilakukan pendefinisian terhadap masing-masing objek. Pendefinisian ini dilakukan berdasarkan luasnya, eksentrisitas, serta *circularity*-nya. Untuk melakukan pendefinisian ini, pertama-tama dilakukan penggabungan hasil deteksi mikroaneurisme dengan deteksi *hemorrhage* lalu pelabelan. Setelah itu, dibuat tiga buah matriks kosong untuk menampung hasil deteksi mikroaneurisme, *dot hemorrhage*, serta *blot hemorrhage*. *Hemorrhage* dibagi menjadi dua yaitu *dot hemorrhage* yang berukuran lebih kecil dan *blot hemorrhage* yang ukurannya lebih besar. Kemudian dilakukan seleksi terhadap objek berdasarkan eksentrisitas dan *circularity*-nya. *Circularity* merupakan ukuran ‘kebulatan’ suatu objek yang bernilai 0 sampai 1, di mana semakin mendekati angka 1 berarti semakin berbentuk lingkaran. *Circularity* sendiri didapatkan dengan rumus:

$$\frac{4\pi Area}{Keliling^2}$$

Pada dasarnya, mikroaneurisme dan *hemorrhage* berbentuk menyerupai oval atau lingkaran (bukan garis), sehingga objek yang terlihat seperti garis akan dieliminasi. Setelah dilakukan eliminasi terhadap bentuk seperti garis, mikroaneurisme dan *hemorrhage* didefinisikan berdasarkan luasnya. Mikroaneurisme didefinisikan berukuran kurang dari 35 pixel, *dot hemorrhage* memiliki luas antara 35 – 150 pixel, sedangkan *blot hemorrhage* sisanya.

```
%% Combining Microaneurism and Hemorrhage Detection

% Klasifikasi mikroaneurisma dan pendarahan - penjumlahan hasil deteksi
% mikroaneurisma dan pendarahan (sebagai kandidat)
MA_HMA_cand = MA_clean + HMA_clean;
MA_HMA_cand = (MA_HMA_cand > 0);
%figure, imshow(MA_HMA_cand) %masi kebentuk dari HMA_clean

% Pelabelan objek
[label, jum_lab] = bwlabel(MA_HMA_cand, 8);
L = bwlabel(label);

% Matriks 0 untuk menampung hasil deteksi mikroaneurisma, dot
```

```

hemorrhages,
% blot hemorrhages
Fin_MA = zeros(size(L));
Fin_hemodot = zeros(size(L));
Fin_hemoblot = zeros(size(L));

% Pendefinisian masing-masing deteksi berdasarkan luas objek yang telah
% diberi label
for i = 1:jum_lab
    [x, y] = find(L==i);
    lab = zeros(size(L));
    for j = 1:size(x,1)
        lab(x(j), y(j)) = 1;
    end
    %Pengabaian yang berbentuk garis
    perim = regionprops(lab, 'Perimeter');
    objecc = regionprops(lab, 'Eccentricity');
    objarea = regionprops(lab, 'Area');
    Circularity=(4*pi*objarea.Area)./(perim.Perimeter^2);
    if (objecc.Eccentricity < 1 && Circularity > 0.45)
        if (objarea.Area < 35)
            Fin_MA = Fin_MA + lab;
        elseif (objarea.Area < 130)
            Fin_hemodot = Fin_hemodot + lab;
        else
            Fin_hemoblot = Fin_hemoblot + lab;
        end
    end
end
Fin_MA = (Fin_MA > 0);
Fin_hemodot = (Fin_hemodot > 0);
Fin_hemoblot = (Fin_hemoblot > 0);
figure, imshow(Fin_MA)
figure, imshow(Fin_hemodot)
figure, imshow(Fin_hemoblot)

```

Proses tersebut mendefinisikan hasil akhir deteksi mikroaneurisma dan pendarahan. Proses terakhir yaitu melakukan klasifikasi berdasarkan jumlah dan lokasi mikroaneurisma dan pendarahan. Memang seharusnya deteksi tergolong normal jika tidak terdeteksi kelainan apapun. Namun pada kenyataannya masih sering terdeteksi *noise* sehingga dilakukan semacam ‘toleransi’ terhadap objek yang terdeteksi. Klasifikasi retinopati diabetes dapat dilihat pada kode berikut.

```

% Klasifikasi NPDR %
if (jumlah_mikro <= 15 && jumlah_blot <= 6 && jumlah_dot <= 8)
    result=fprintf('Normal \n');
elseif (jumlah_mikro<=35 && jumlah_blot<=8 && jumlah_dot <= 15)
    result=fprintf('Mild \n');
elseif (jumlah_dot>=20 && jumlah_blot>=10 )
    result = fprintf('Severe \n');
else
    result = fprintf('Moderate \n');
end

```

Seperti yang telah disampaikan, kemungkinan terdeteksi *noise* pada masing-masing deteksi objek sangat besar, sehingga walaupun proses deteksi menunjukkan adanya mikroaneurisme dan *hemorrhage*, hasil klasifikasi menunjukkan bahwa mata tersebut normal.

Setelah dilihat hasil melalui Matlab, proses deteksi tersebut diimplementasikan pada OpenCV untuk Android, sehingga bahasa yang digunakan adalah Java. OpenCV mendukung pengolahan gambar sehingga terdapat banyak fungsi *built in* dalam OpenCV untuk mengolah gambar. Jikapun tidak tersedia, Android Studio menyediakan kemampuan untuk mengaplikasikan kode C++ pada Android Studio dengan memanfaatkan NDK.

Untuk pengembangan sistem ini, digunakan OpenCV 3.1.0 pada Android Studio 2.3.1.

```
package com.example.amalia.testopencv;

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import org.opencv.android.Utils;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.core.MatOfPoint;
import org.opencv.core.MatOfPoint2f;
import org.opencv.core.Point;
import org.opencv.core.Scalar;
import org.opencv.core.Size;
import org.opencv.core.Core;
import org.opencv.imgproc.CLAHE;
import org.opencv.imgproc.Imgproc;
import org.opencv.imgproc.Moments;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

import static java.lang.Math.sqrt;

public class MainActivity extends AppCompatActivity {

    // Used to load the 'native-lib' library on application startup.
    static {
        System.loadLibrary("opencv_java3");
        System.loadLibrary("native-lib");
    }

    // Definisi variabel

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Pemanggilan gambar
    }

    //Deteksi pembuluh darah
    public void deteksiVessel() {
        //Pre-processing
        //Ekstraksi kanal hijau

        //Frangi Filter

        //Penghilangan bagian yang luasnya kurang dari 1400 pixel
    }
}
```

```

        //Deteksi bentuk retina

//Deteksi optical disk
public void deteksiOD(){
    //Closing, thresholding

    //Cek eksentrisitas: OD bila < 0.9

    //Berdasarkan area: cari lingkaran yang dengan luas maks

//Deteksi mikroaneurisma
public void deteksiMA(){
    //Eliminasi di bawah 35 pixel
    //Pengurangan hasil filter Frangi dengan optical disk, pembuluh
darah, bentuk retina, hasil eliminasi
    //Eliminasi bagian yg kurang dari 5 pixel

//Deteksi hemorrhage
public void deteksiHemo(){
    //Frangi Filter untuk hemorrhage
    //Eliminasi di bawah 700 pixel
    //Pengurangan hasil filter Frangi dengan optical disk, pembuluh
darah, bentuk retina, hasil eliminasi
    //Eliminasi bagian yang kurang dari 35 pixel

//Definisi mikroaneurisma dan hemorrhage dan klasifikasi penyakit
public void klasifikasi(){
    //Labelling objek
    //Definisi objek berdasarkan area

    //Klasifikasi penyakit berdasarkan jumlah objek
}

/**
 * A native method that is implemented by the 'native-lib' native
library,
 * which is packaged with this application.
 */
public native void frangimikro(long addrinp, long addrout);
public native void frangihemo(long addrinp, long addrout);
}

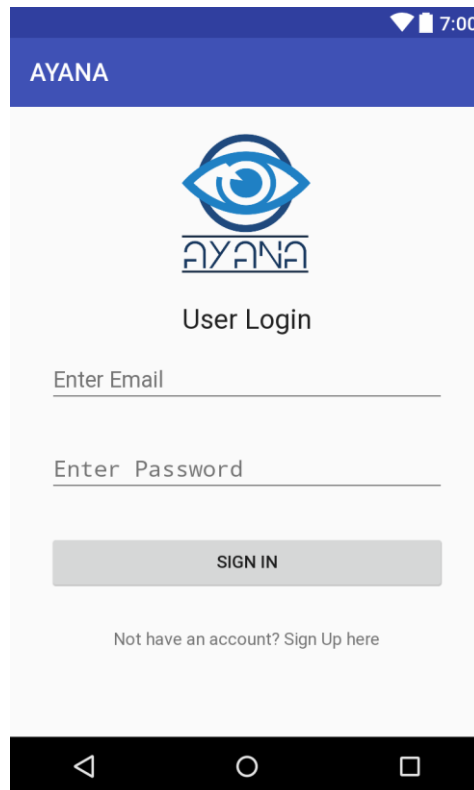
```

2.3 Graphical User Interface (GUI) Aplikasi

GUI dirancang menggunakan *software* Android Studio 2.2.

2.3.1 Tampilan Menu 'Sign In'

Tampilan aplikasi saat pertama kali dibuka adalah sebagai berikut.



Gambar 2.29 Tampilan untuk Melakukan *Sign In*

Pada saat pertama kali meng-*install* aplikasi, pengguna akan berada pada menu *Sign In* dan diberikan pilihan dalam memasukkan identitas untuk dapat mengakses masuk ke dalam aplikasi atau melakukan *Sign Up* atau pendaftaran. Source code dalam java yang digunakan untuk menghasilkan tampilan seperti ini adalah sebagai berikut.

```
package com.example.userpc.mojojo;

import android.app.ProgressDialog;
import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class login extends AppCompatActivity implements
View.OnClickListener {

    //defining views
    private Button buttonSignIn;
    private EditText editTextEmail;
    private EditText editTextPassword;
    private TextView textViewSignup;

    //firebase auth object
```

```

private FirebaseAuth firebaseAuth;

//progress dialog
private ProgressDialog progressDialog;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.login);

    //getting firebase auth object
    firebaseAuth = FirebaseAuth.getInstance();

    //if the objects getCurrentuser method is not null
    //means user is already logged in
    if(firebaseAuth.getCurrentUser() != null) {
        //close this activity
        finish();
        //opening profile activity
        startActivity(new Intent(getApplicationContext(),
MainActivity.class));
    }

    //initializing views
    editTextEmail = (EditText) findViewById(R.id.editTextEmail);
    editTextPassword = (EditText)
findViewById(R.id.editTextPassword);
    buttonSignIn = (Button) findViewById(R.id.buttonSignin);
    textViewSignUp = (TextView)
findViewById(R.id.textViewSignUp);

    progressDialog = new ProgressDialog(this);

    //attaching click listener
    buttonSignIn.setOnClickListener(this);
    textViewSignUp.setOnClickListener(this);
}

//method for user login
private void userLogin() {
    String email = editTextEmail.getText().toString().trim();
    String password =
editTextPassword.getText().toString().trim();

    //checking if email and passwords are empty
    if(TextUtils.isEmpty(email)) {
        Toast.makeText(this, "Please enter
email", Toast.LENGTH_LONG).show();
        return;
    }

    if(TextUtils.isEmpty(password)) {
        Toast.makeText(this, "Please enter
password", Toast.LENGTH_LONG).show();
        return;
    }

    //if the email and password are not empty
    //displaying a progress dialog
    progressDialog.setMessage("Registering Please Wait...");
    progressDialog.show();
}

```

```

        //logging in the user
        firebaseAuth.signInWithEmailAndPassword(email, password)
            .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult>
task) {

                progressDialog.dismiss();
                //if the task is successfull
                if(task.isSuccessful()){
                    //start the profile activity
                    finish();
                    startActivity(new
Intent(getApplicationContext(), MainActivity.class));
                }
            }
        });
    }

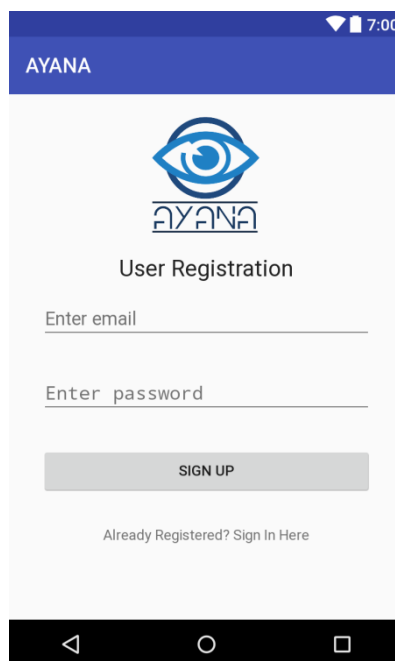
    @Override
    public void onClick(View view) {
        if(view == buttonSignIn){
            userLogin();
        }

        if(view == textViewSignup){
            finish();
            startActivity(new Intent(this, signup.class));
        }
    }
}

```

2.3.2 Tampilan Menu 'Sign Up'

Menu ini berfungsi untuk melakukan pendaftaran sebelum dapat mengakses ke dalam aplikasi.



Gambar 2.30 Tampilan untuk Melakukan *Sign Up*

Source code dalam bahasa java yang digunakan untuk menghasilkan tampilan seperti di atas adalah sebagai berikut.

```
package com.example.userpc.mojojo;

import android.app.ProgressDialog;
import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class signup extends AppCompatActivity implements
View.OnClickListener {

    //defining view objects
    private EditText editTextEmail;
    private EditText editTextPassword;
    private Button buttonSignup;

    private TextView textViewSignin;

    private ProgressDialog progressDialog;

    //defining firebaseauth object
    private FirebaseAuth firebaseAuth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.signup);

        //initializing firebase auth object
        firebaseAuth = FirebaseAuth.getInstance();

        //if getCurrentUser does not returns null
        if(firebaseAuth.getCurrentUser() != null){
            //that means user is already logged in
            //so close this activity
            finish();

            //and open profile activity
            startActivity(new Intent(getApplicationContext(),
MainActivity.class));
        }

        //initializing views
        editTextEmail = (EditText) findViewById(R.id.editTextEmail);
```



```

        editTextPassword = (EditText)
findViewById(R.id.editTextPassword);
        textViewSignin = (TextView) findViewById(R.id.textViewSignin);

        buttonSignup = (Button) findViewById(R.id.buttonSignup);

        progressDialog = new ProgressDialog(this);

        //attaching listener to button
        buttonSignup.setOnClickListener(this);
        textViewSignin.setOnClickListener(this);
    }

    private void registerUser(){

        //getting email and password from edit texts
        String email = editTextEmail.getText().toString().trim();
        String password =
editTextPassword.getText().toString().trim();

        //checking if email and passwords are empty
        if(TextUtils.isEmpty(email)){
            Toast.makeText(this, "Please enter
email", Toast.LENGTH_LONG).show();
            return;
        }

        if(TextUtils.isEmpty(password)){
            Toast.makeText(this, "Please enter
password", Toast.LENGTH_LONG).show();
            return;
        }

        //if the email and password are not empty
        //displaying a progress dialog

        progressDialog.setMessage("Registering Please Wait...");
        progressDialog.show();

        //creating a new user
        firebaseAuth.createUserWithEmailAndPassword(email, password)
            .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult>
task) {

                //checking if success
                if(task.isSuccessful()){
                    finish();
                    startActivity(new
Intent(getApplicationContext(), MainActivity.class));
                }else{
                    //display some message here
                    Toast.makeText(signup.this, "Registration
Error", Toast.LENGTH_LONG).show();
                }
                progressDialog.dismiss();
            }
        });
    }
}

```

```

@Override
public void onClick(View view) {

    if(view == buttonSignup){
        registerUser();
    }

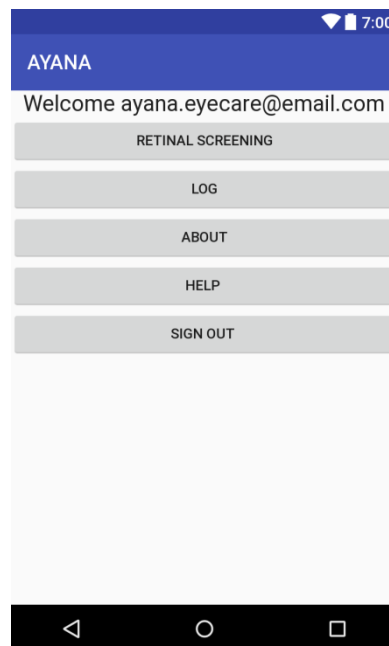
    if(view == textViewSignin){
        //open login activity when user taps on the already
        registered textview
        startActivity(new Intent(this, login.class));
    }

}
}

```

2.3.3 Tampilan Menu 'Utama'

Pilihan menu ini berfungsi menentukan hasil diagnosa dengan memasukkan identitas serta gambar.



Gambar 2.31 Tampilan untuk Menu Utama

Pada *activity* ini, pengguna akan berada pada menu utama dan diberikan pilihan dalam menggunakan aplikasi yaitu diantaranya *Retinal Screening*, *Log*, *About*, *Help* dan *Sign Out*. Ketika menekan tombol *Sign Out* maka akan kembali ke tampilan *Sign In*. Source code dalam *java* yang digunakan untuk menghasilkan tampilan seperti ini adalah sebagai berikut.

```

package com.example.userpc.mojojo;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.content.DialogInterface;
import android.app.AlertDialog;

```

```

import android.widget.TextView;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public class MainActivity extends AppCompatActivity implements
View.OnClickListener{

    //firebase auth object
    private FirebaseAuth firebaseAuth;
    //view objects
    private TextView textViewUserEmail;
    private Button buttonImg, buttonAbt, buttonHlp, buttonLogout;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //initializing firebase authentication object
        firebaseAuth = FirebaseAuth.getInstance();
        //if the user is not logged in
        //that means current user will return null
        if(firebaseAuth.getCurrentUser() == null) {
            //closing this activity
            finish();
            //starting login activity
            startActivity(new Intent(this, login.class));
        }

        //getting current user
        FirebaseUser user = firebaseAuth.getCurrentUser();
        //initializing views
        textViewUserEmail = (TextView)
findViewById(R.id.textViewUserEmail);
        buttonImg = (Button) findViewById(R.id.button2);
        buttonAbt = (Button) findViewById(R.id.button3);
        buttonHlp = (Button) findViewById(R.id.button4);
        buttonLogout = (Button) findViewById(R.id.button5);
        //displaying logged in user name
        textViewUserEmail.setText("Welcome "+user.getEmail());
        //Here MainActivity.this is a Current Class Reference
(context)
        buttonImg.setOnClickListener(this);
        buttonAbt.setOnClickListener(this);
        buttonHlp.setOnClickListener(this);
        buttonLogout.setOnClickListener(this);
    }

    @Override
    public void onClick(View view) {
        //assign for intent
        Intent f = null;
        //if image is pressed
        if(view == buttonImg){
            f = new Intent(this, biodata.class);
        } //if about is pressed
        else if(view == buttonAbt) {
            f = new Intent(this, about.class);
        } //if help is pressed
        else if(view == buttonHlp) {
            f = new Intent(this, help.class);
        } //if logout is pressed
    }

```

```

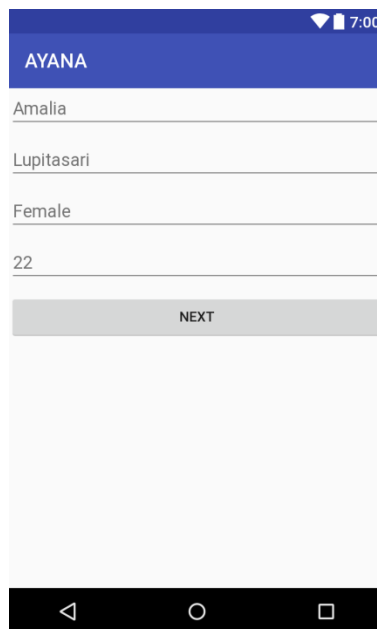
else if(view == buttonLogout){
    //logging out the user
    firebaseAuth.signOut();
    f = new Intent(this, login.class);
} finish(); //closing activity
startActivity(f); //starting login activity
}

public void onBackPressed() {
    new AlertDialog.Builder(this)
        .setMessage("Apa anda yakin ingin menutup aplikasi?")
        .setCancelable(false)
        .setPositiveButton("Ya", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int
id) {
                MainActivity.this.finish();
            }
        })
        .setNegativeButton("Tidak", null)
        .show();
}
}

```

2.3.4 Tampilan Menu 'Retinal Screening - Biodata'

Pilihan menu *Retinal Screening* akan memberi tampilan tentang biodata yang diambil sebagai data yang diperiksa.



Gambar 2.32 Tampilan untuk Menu 'Biodata'

Source code dalam bahasa java yang digunakan untuk menghasilkan tampilan seperti di atas adalah sebagai berikut.

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

```

```

<EditText
    android:id="@+id/firstName"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="First name" />

<EditText
    android:id="@+id/lastName"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Last name"
    android:layout_marginTop="10dp"
    android:layout_below="@+id/firstName"
    android:layout_alignParentStart="true" />

<EditText
    android:id="@+id/gender"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Gender"
    android:layout_marginTop="10dp"
    android:layout_below="@+id/lastName"/>

<EditText
    android:id="@+id/age"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="Age"
    android:layout_marginTop="10dp"
    android:layout_below="@+id/gender"/>

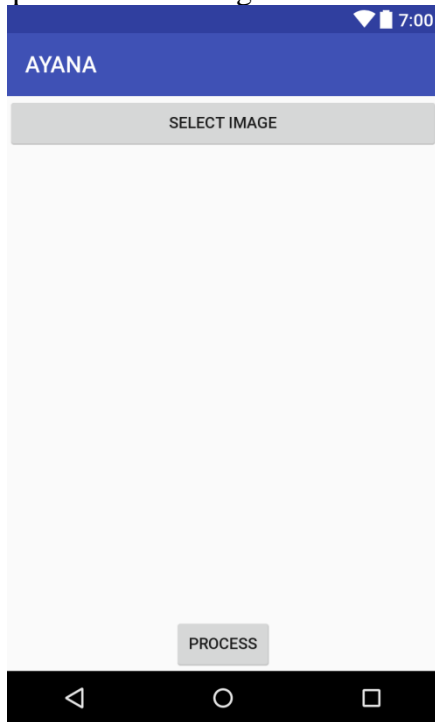
<Button
    android:id="@+id/buttonNext"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Next"
    android:layout_marginTop="10dp"/>

</LinearLayout>

```

2.3.5 Tampilan Menu 'Retinal Screening - Camera'

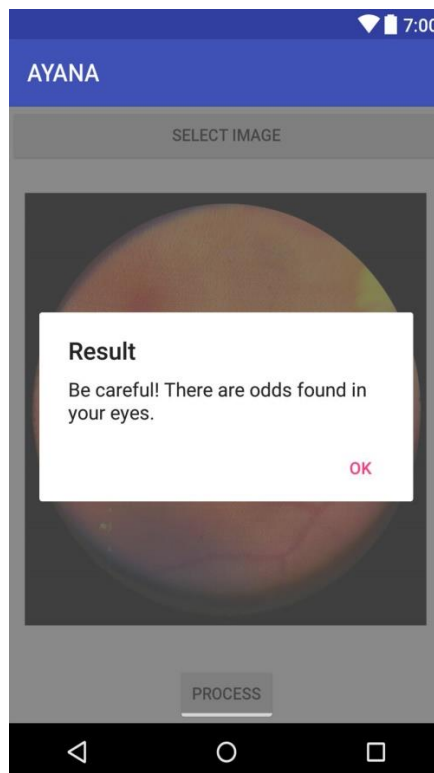
Tampilan ketika biodata yang diperiksa telah diisi dan menekan tombol *Next*, maka ditampilkan menu sebagai berikut.



Gambar 2.33 Tampilan untuk Menu *Camera(1)*



Gambar 2.34 Tampilan untuk Menu *Camera(2)*



Gambar 2.35 Tampilan untuk Menu *Camera(3)*

Dalam menu ini, dilakukan pengambilan citra retina dari yang diperiksa sehingga dapat diproses kemudian. Pengambilan citra dapat dilakukan dengan dua cara, yaitu

menggunakan kamera atau mengambil dari *Gallery*. Kemudian dapat melakukan pemrosesan dengan menekan tombol *Process*’ sehingga akan muncul pemberitahuan bahwa yang diperiksa dalam keadaan sehat atau tidak pada matanya. *Source code* dalam bahasa *java* yang digunakan untuk menghasilkan tampilan seperti di atas adalah sebagai berikut.

```
package com.example.userpc.mojojo;

import java.io.File;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;
import android.app.Activity;
import android.content.DialogInterface;
import android.database.Cursor;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.content.pm.ResolveInfo;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;
import android.widget.VideoView;

public class camera extends AppCompatActivity {

    private static final int ACTION_TAKE_PHOTO_B = 1;
    private static final int RESULT_LOAD_IMG = 2;
    private static final int REQUEST_CROP_ICON = 3;
    String imgDecodableString;

    private static final String BITMAP_STORAGE_KEY = "viewbitmap";
    private static final String IMAGEVIEW_VISIBILITY_STORAGE_KEY =
"imageviewvisibility";
    private ImageView mImageView;
    private Bitmap mImageBitmap;

    private static final String VIDEO_STORAGE_KEY = "viewvideo";
    private static final String VIDEOVIEW_VISIBILITY_STORAGE_KEY =
"videoviewvisibility";
    private VideoView mVideoView;
    private Uri mVideoUri;

    private String mCurrentPhotoPath;

    private static final String JPEG_FILE_PREFIX = "IMG_";
    private static final String JPEG_FILE_SUFFIX = ".jpg";
```

```

private AlbumStorageDirFactory mAlbumStorageDirFactory = null;

private static final String TAG = "GalleryUtil";

Button AlrtBtn, scBtn;
private Uri mImageCaptureUri;

private boolean backPressedToExitOnce = false;
private Toast toast = null;

/* Photo album for this application */
private String getAlbumName() {
    return getString(R.string.album_name);
}

private File getAlbumDir() {
    File storageDir = null;

    if
(Environment.MEDIA_MOUNTED.equals(Environment.getExternalStorageState())
) {

        storageDir =
mAlbumStorageDirFactory.getAlbumStorageDir(getAlbumName());

        if (storageDir != null) {
            if (!storageDir.mkdirs()) {
                if (!storageDir.exists()) {
                    Log.d("CameraSample", "failed to create
directory");
                    return null;
                }
            }
        }
    } else {
        Log.v(getString(R.string.app_name), "External storage is not
mounted READ/WRITE.");
    }

    return storageDir;
}

private File createImageFile() throws IOException {
    // Create an image file name
    String timeStamp = new
SimpleDateFormat("yyyyMMdd_HHmmss").format(new Date());
    String imageFileName = JPEG_FILE_PREFIX + timeStamp + "_";
    File albumF = getAlbumDir();
    File imageF = File.createTempFile(imageFileName,
JPEG_FILE_SUFFIX, albumF);
    return imageF;
}

private File setUpPhotoFile() throws IOException {

    File f = createImageFile();
    mCurrentPhotoPath = f.getAbsolutePath();
}

```



```

        return f;
    }

    private void setPic() {

        /* There isn't enough memory to open up more than a couple camera
        photos */
        /* So pre-scale the target bitmap into which the file is decoded
        */

        /* Get the size of the ImageView */
        int targetW = mImageView.getWidth();
        int targetH = mImageView.getHeight();

        /* Get the size of the image */
        BitmapFactory.Options bmOptions = new BitmapFactory.Options();
        bmOptions.inJustDecodeBounds = true;
        BitmapFactory.decodeFile(mCurrentPhotoPath, bmOptions);
        int photoW = bmOptions.outWidth;
        int photoH = bmOptions.outHeight;

        /* Figure out which way needs to be reduced less */
        int scaleFactor = 1;
        if ((targetW > 0) || (targetH > 0)) {
            scaleFactor = Math.min(photoW / targetW, photoH / targetH);
        }

        /* Set bitmap options to scale the image decode target */
        bmOptions.inJustDecodeBounds = false;
        bmOptions.inSampleSize = scaleFactor;
        bmOptions.inPurgeable = true;

        /* Decode the JPEG file into a Bitmap */
        Bitmap bitmap = BitmapFactory.decodeFile(mCurrentPhotoPath,
        bmOptions);

        /* Associate the Bitmap to the ImageView */
        mImageView.setImageBitmap(bitmap);
        mVideoUri = null;
        mImageView.setVisibility(View.VISIBLE);
        //mVideoView.setVisibility(View.INVISIBLE);
    }

    private void galleryAddPic() {
        Intent mediaScanIntent = new
        Intent("android.intent.action.MEDIA_SCANNER_SCAN_FILE");
        File f = new File(mCurrentPhotoPath);
        Uri contentUri = Uri.fromFile(f);
        mediaScanIntent.setData(contentUri);
        this.sendBroadcast(mediaScanIntent);
    }

    private void dispatchTakePictureIntent(int actionCode) {

        Intent takePictureIntent = new
        Intent(MediaStore.ACTION_IMAGE_CAPTURE);

        switch (actionCode) {
            case ACTION_TAKE_PHOTO_B:
                File f = null;

```

```

        try {
            f = setUpPhotoFile();
            mCurrentPhotoPath = f.getAbsolutePath();
            takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT,
Uri.fromFile(f));
        } catch (IOException e) {
            e.printStackTrace();
            f = null;
            mCurrentPhotoPath = null;
        }
        break;

        default:
            break;
    } // switch

    startActivityForResult(takePictureIntent, actionCode);
}

private void handleBigCameraPhoto() {

    if (mCurrentPhotoPath != null) {
        setPic();
        galleryAddPic();
        mCurrentPhotoPath = null;
        //scBtn.setVisibility(View.VISIBLE);
        //picBtn.setVisibility(View.INVISIBLE);
    }

}

Button.OnClickListener mTakePicOnClickListener =
    new Button.OnClickListener() {
        @Override
        public void onClick(View v) {
            dispatchTakePictureIntent(ACTION_TAKE_PHOTO_B);
        }
    };

/**
 * Called when the activity is first created.
 */
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.camera);

    mImageView = (ImageView) findViewById(R.id.imageView1);
    mImageBitmap = null;
    mVideoUri = null;

    //scBtn = (Button) findViewById(R.id.Scan);
    //setBtnListenerOrDisable(
    //    scBtn,
    //    mTakePicOnClickListener,
    //    MediaStore.ACTION_IMAGE_CAPTURE
    //);

    AlrtBtn = (Button) findViewById(R.id.button7);
    AlrtBtn.setOnClickListener(new View.OnClickListener() {

```

```

        @Override
        public void onClick(View view) {
            showDialog();
        }
    });

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.FROYO) {
        mAlbumStorageDirFactory = new FroYoAlbumDirFactory();
    } else {
        mAlbumStorageDirFactory = new BaseAlbumDirFactory();
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
    switch (requestCode) {
        case ACTION_TAKE_PHOTO_B: {
            if (resultCode == RESULT_OK) {
                handleBigCameraPhoto();
            }
            break;
        }
        case RESULT_LOAD_IMG: {
            if (resultCode == Activity.RESULT_OK && data != null &&
data.getData() != null) {
                try {
                    Uri selectedImage = data.getData();

                    String[] filePathColumn =
{MediaStore.Images.Media.DATA};
                    Cursor cursor =
getContentResolver().query(selectedImage,
filePathColumn, null, null, null);
                    cursor.moveToFirst();
                    int columnIndex =
cursor.getColumnIndex(filePathColumn[0]);
                    String picturePath =
cursor.getString(columnIndex);
                    cursor.close();

                    //return Image Path to the Main Activity
                    Intent returnFromGalleryIntent = new Intent();
                    returnFromGalleryIntent.putExtra("picturePath",
picturePath);

                    setResult(RESULT_OK, returnFromGalleryIntent);
                    finish();
                } catch (Exception e) {
                    e.printStackTrace();
                    Intent returnFromGalleryIntent = new Intent();
                    setResult(RESULT_CANCELED,
returnFromGalleryIntent);
                    finish();
                }
            } else {
                Log.i(TAG, "RESULT_CANCELED");
                Intent returnFromGalleryIntent = new Intent();
                setResult(RESULT_CANCELED, returnFromGalleryIntent);
                finish();
            }
            break;
        }
    }
}

```

```

    }
    } // switch
}

// Some lifecycle callbacks so that the image can survive
orientation change
@Override
protected void onSaveInstanceState(Bundle outState) {
    outState.putParcelable(BITMAP_STORAGE_KEY, mImageBitmap);
    outState.putParcelable(VIDEO_STORAGE_KEY, mVideoUri);
    outState.putBoolean(IMAGEVIEW_VISIBILITY_STORAGE_KEY,
(mImageBitmap != null));
    outState.putBoolean(VIDEOVIEW_VISIBILITY_STORAGE_KEY, (mVideoUri
!= null));
    super.onSaveInstanceState(outState);
}

@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    mImageBitmap =
savedInstanceState.getParcelable(BITMAP_STORAGE_KEY);
    mVideoUri = savedInstanceState.getParcelable(VIDEO_STORAGE_KEY);
    mImageView.setImageBitmap(mImageBitmap);
    mImageView.setVisibility(
savedInstanceState.getBoolean(IMAGEVIEW_VISIBILITY_STORAGE_KEY) ?
        ImageView.VISIBLE : ImageView.INVISIBLE
    );
}

/**
 * Indicates whether the specified action can be used as an intent.
This
 * method queries the package manager for installed packages that
can
 * respond to an intent with the specified action. If no suitable
package is
 * found, this method returns false.
 * http://android-developers.blogspot.com/2009/01/can-i-use-this-intent.html
 *
 * @param context The application's environment.
 * @param action The Intent action to check for availability.
 * @return True if an Intent with the specified action can be sent
and
 * responded to, false otherwise.
 */
public static boolean isIntentAvailable(Context context, String
action) {
    final PackageManager packageManager =
context.getPackageManager();
    final Intent intent = new Intent(action);
    List<ResolveInfo> list =
        packageManager.queryIntentActivities(intent,
            PackageManager.MATCH_DEFAULT_ONLY);
    return list.size() > 0;
}

private void setBtnListenerOrDisable(
    Button btn,

```

```

        Button.OnClickListener onClickListener,
        String intentName
    ) {
        if (isIntentAvailable(this, intentName)) {
            btn.setOnClickListener(onClickListener);
        } else {
            btn.setText(
                getText(R.string.cannot).toString() + " " +
btn.getText());
            btn.setClickable(false);
        }
    }

    // public void onBackPressed () {
    //     scBtn.setVisibility(View.INVISIBLE);
    //     picBtn.setVisibility(View.VISIBLE);
    //     if (backPressedToExitOnce) {
    //         super.onBackPressed();
    //     } else {
    //         this.backPressedToExitOnce = true;
    //         showToast("Press again to exit");
    //         new Handler().postDelayed(new Runnable() {

    //             @Override
    //             public void run() {
    //                 backPressedToExitOnce = false;
    //             }
    //         }, 2000);
    //     }
    // }

    /**
     * Created to make sure that you toast doesn't show multiple times,
     if user pressed back
     * button more than once.
     *
     * @param message Message to show on toast.
     */
    private void showToast(String message) {
        if (this.toast == null) {
            // Create toast if found null, it would be the case of first
call only
            this.toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);

        } else if (this.toast.getView() == null) {
            // Toast not showing, so create new one
            this.toast = Toast.makeText(this, message,
Toast.LENGTH_SHORT);

        } else {
            // Updating toast message is showing
            this.toast.setText(message);
        }

        // Showing toast finally
        this.toast.show();
    }

    /**
     * Kill the toast if showing. Supposed to call from onPause() of

```

```

activity.
    * So that toast also get removed as activity goes to background, to
    improve
    * better app experiance for user
    */
    private void killToast() {
        if (this.toast != null) {
            this.toast.cancel();
        }
    }

    @Override
    protected void onPause() {
        killToast();
        super.onPause();
    }

    public void loadImagefromGallery(View view) {
        // Create intent to Open Image applications like Gallery, Google
        Photos
        Intent galleryIntent = new Intent(Intent.ACTION_PICK,
        android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
        // Start the Intent
        startActivityForResult(galleryIntent, RESULT_LOAD_IMG);
    }

    private void showDialog(){
        AlertDialog.Builder alertDialogBuilder = new
        AlertDialog.Builder(
            this);

        // set title dialog
        alertDialogBuilder.setTitle("Result");

        // set pesan dari dialog
        alertDialogBuilder
            .setMessage("Be careful! There are odds found in your
        eyes.")
            .setCancelable(false)
            .setPositiveButton("OK", new
        DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                // jika tombol diklik, maka akan menutup
                activity ini
                camera.this.finish();
            }
        });

        // membuat alert dialog dari builder
        AlertDialog alertDialog = alertDialogBuilder.create();

        // menampilkan alert dialog
        alertDialog.show();
    }
}

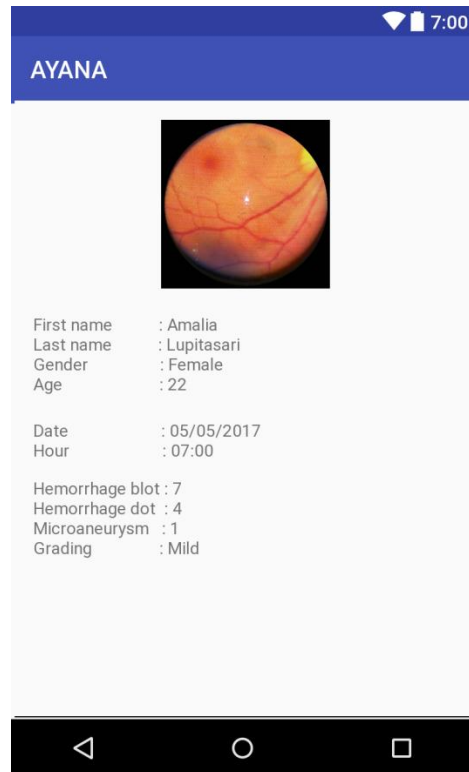
```

2.3.6 Tampilan Menu 'Log'

Tampilan dilakukan ketika tombol *Log* ditekan maka akan ditampilkan daftar dari data yang sudah diproses. Daftar ditampilkan secaraurut berdasarkan waktu dilakukan pemeriksaan.



Gambar 2.36 Tampilan untuk Menu *Log*



Gambar 2.37 Tampilan untuk data dalam Menu *Log*

Tampilan ini berfungsi untuk menampilkan hasil pemrosesan yang sudah dilakukan. Selain itu hasil pemrosesan berupa citra retina, biodata yang diperiksa, serta jumlah kelainan yang ada pada mata. Berikut merupakan *source code* .xml yang digunakan.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <TextView
        android:id="@+id/label"
        android:text="@+id/TextView01"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="30px"/>
</LinearLayout>
```