BAB III

ANALISIS DAN PERANCANGAN

Untuk mengidentifikasi permasalahan serta proses yang terjadi dalam membangun sebuah aplikasi maka dilakukan analisis mendalam. Analisis dilakukan terhadap beragam penelitian *demosaicing* yang umum digunakan dalam interpolasi gambar RAW digital pada kamera digital yang menggunakan *bayer color filter mosaic* pada sensornya. Analisis terhadap pengguna *(user)* yang membutuhkan aplikasi ini merupakan *point* utama tujuan dibuatnya aplikasi perbandingan algoritma *demosaicing* maupun sebagai bahan pembelajaran mengenai proses interpolasi. Kebutuhan perangkat keras dan perangkat lunak yang akan digunakan dalam membangun aplikasi adalah faktor penting lainnya agar pemanfaatan perangkat lunak yang digunakan memperoleh hasil yang maksimal.

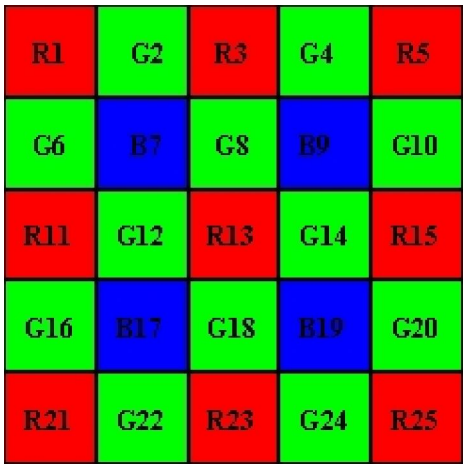
3.1. Analisis masalah

Sebuah gambar RAW yang diperoleh dari hasil pengambilan gambar sebuah kamera digital yang umumnya memiliki sebuah sensor yang dilengkapi *bayer color filter mosaic* tidak serta merta bisa dilihat dan digunakan pada sistem operasi dikarenakan format RAW tersebut tidak dikenali sebagai format RGB pada umumnya. Hal ini dikarenakan mekanisme *bayer color filter mosaic* yang hanya dapat menangkap sebuah warna pada sebuah *photo diode* berukuran satu *pixel* yaitu salah satu dari warna RGB (pengecualian warna hijau memiliki 2 pixel pada sensor kamera digital dikarenakan warna hijau memiliki *color range* yang

lebih kuat dari warna lainnya). Dengan demikian pixel yang lain tidak akan memiliki nilai karena telah melalui penyaringan warna. Pixel yang hilang tersebut dianalogikan sebagai kekosongan pixel warna dan harus di isi untuk mengkonstruksi sebuah gambar utuh dengan cara interpolasi.

3.2. Interpolasi menggunakan Algoritma Bilinear

Bilinear interpolation adalah interpolasi paling dasar dalam *demosaicing*, nilai pixel yang hilang diperoleh dari hasil rata-rata pixel terdekat disekelilingnya.



Gambar 3.1 Penomoran Pixel

3.2.1. Tahap-tahap penyelesaian masalah menggunakan Algoritma Bilinear

Berikut ini merupakan tahap-tahap secara matematis penyelesaian interpolasi dengan menggunakan algoritma Bilinear:

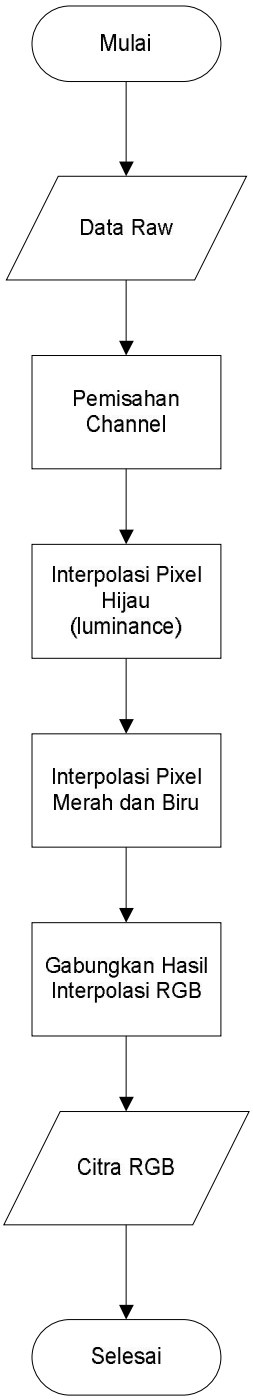
1. Nilai interpolasi pixel hijau (*green*, dengan notasi G) pada posisi pixel biru (B8) didapat dari rata-rata nilai pixel di bagian utara, barat, selatan, dan timur yang dijumlahkan kemudian dibagi empat , contoh :

```
G8 = (G3+G7+G13+G9) / 4
```

2. Nilai interpolasi pixel merah (*red*, dengan notasi R) pada posisi pixel hijau (G7) didapat dari rata-rata dua nilai pixel terdekatnya yang dijumlahkan lalu dibagi dua, contoh : `R7 = (R2+R12) / 2`
3. Nilai interpolasi pixel merah pada posisi pixel biru (B8) didapat dari rata-rata nilai pixel merah disekeliling pixel biru tersebut yang dijumlahkan lalu dibagi empat, contoh : `R8 = (R2+R4+R12+R14) / 4`
4. Nilai interpolasi pixel biru (*blue*, dengan notasi B) pada posisi pixel hijau (G7) didapat dari rata-rata dua nilai pixel terdekatnya yang dijumlahkan lalu dibagi dua, contoh : `B7 = (B6+B8) / 2`
5. Nilai interpolasi pixel biru pada posisi pixel merah (R12) didapat dari rata-rata nilai pixel biru disekeliling pixel merah tersebut yang dijumlahkan lalu dibagi empat, contoh : `B12 = (B6+B8+B16+B18) / 4`

3.2.2. Flowchart interpolasi menggunakan Algoritma Bilinear

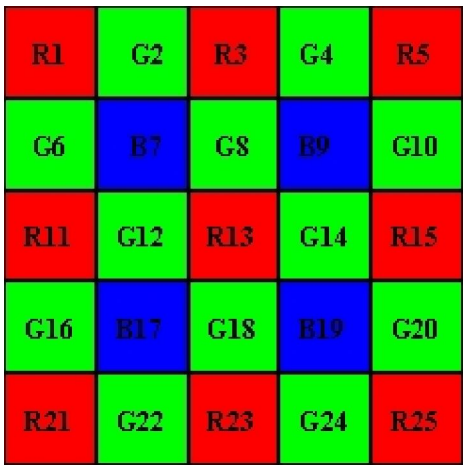
Berikut ini adalah flowchart interpolasi menggunakan Algoritma Bilinear. Data Raw mengalami pemisahan channel terlebih dahulu lalu mulai interpolasi pada piksel *green* sebagai luminance dilanjutkan dengan piksel *green, blue* dan *red.* Hasil interpolasi masing-masing piksel kemudian digabungkan menjadi citra RGB.



Gambar 3.2 Flowchart Algoritma Bilinear

3.3. Interpolasi menggunakan Algoritma Variable Number of Gradient

Algoritma ini dilakukan pada mosaic yang terdiri dari 5x5 pixel dengan pixel tengah sebagai acuan 8 penjuru arah interpolasi, yaitu gradient utara (N), timur (E), selatan (S), barat (W), timur laut (NE), barat laut (NW), tenggara (SE), barat daya (SW). Pada tiap gradient, sebuah nilai treshold dihitung untuk memilih beberapa subset gradient lainnya. Subset gradient ini digunakan untuk melacak daerah yg memiliki detail rendah.



Gambar 3.3 Penomoran Pada Setiap Pixel

Metode matematisnya sebagai berikut:

```
Gradient N = |G8 - G18| + |R3 - R13| + |B7 - B17| / 2 +
|B9 - B19| / 2 + |G2 - G12| / 2 + |G4 - G14| / 2 ;
Gradient E = |G14 - G12| + |R15 - R13| + |B9 - B7| / 2
+ |B19 - B17| / 2 + |G10 - G8| / 2 + |G20 - G18| / 2;
Gradient S = |G18 - G8| + |R23 - R13| + |B19 - B9| / 2
+ |B17 - B7| / 2 + |G24 - G14| / 2 + |G22 - G12| / 2;
```

```
Gradient W = |G12 - G14| + |R11 - R13| + |B17 - B19| /
2 + |B7 - B9| / 2 + |G16 - G18| / 2 + |G6 - G8| / 2;
Gradient NE = |B9 - B17| + |R5 - R13| + |G8 - G12| / 2
+ |G14 - G18| / 2 + |G4 - G8| / 2 + |G10 - G14| / 2;
Gradient SE = |B19 - B7| + |B25 - R13| + |G14 - G8| / 2
+ |G18 - G12| / 2 + |G20 - G14| / 2 + |G24 - G18| / 2;
Gradient NW = |B7 - B19| + |R1 - R13| + |G12 - G18| / 2
+ |G8 - G14| / 2 + |G6 - G12| / 2 + |G2 - G8| / 2;
Gradient SW = |B17 - B9| + |R21 - R13| + |G18 - G14| /
2 + |G12 - G8| / 2 + |G22 - G18| / 2 + |G16 - G12| / 2;
```

Contoh hasil yang didapat sebagai berikut

Tabel 3.1 Gradient Dan Nilai

Gradient	N	E	S	W	NE	SE	NW	SW
Nilai	12	13	7	8	4	7	12	14

1. Tentukan threshold (T) gradient dan pilih sebuah subset

$$T = k_1 * Min + k_2 * (Max - Min)$$

Min adalah nilai minimum gradient (4) dan Max adalah nilai maksimum gradient (14). k1 dan k2 ditetapkan dengan nilai masing-masing 1,5 dan 0,5 maka T = 11. Subset gradient dipilih yang memiliki nilai kurang dari 11, yaitu subset gradient = {S,W,NE,SE}.

2. Temukan lokasi pixel pada daerah yang berhubungan dengan subset gradient terpilih dan menggunakannya untuk menentukan perbedaan perbedaan warna antara center pixel dan warna yang akan di interpolasi.
3. Tentukan rata-rata nilai pixel G, B dan R pada subset gradient.

Tabel 3.2 Nilai Rata-rata Pixel Pada Subset Gradient

	G	B	R
S	G18	(B17 + B19) / 2	(R13 + R23) / 2
W	G12	(B7 + B17) / 2	(R11 + R13) / 2
NE	(G4 + G8 + G10 + G14) / 4	B9	(R13 + R5) / 2
SE	(G14 + G18 + G20 + G24) / 4	B19	(R13 + R25) / 2

G_{sum} = G18 + G12 + (G4 + G8 + G10 + G14) / 4 + (G14 + G18 + G20 G24) / 4;

B_{sum} = (B17 + B19) / 2 + (B7 + B17) / 2 + B9 + B19;

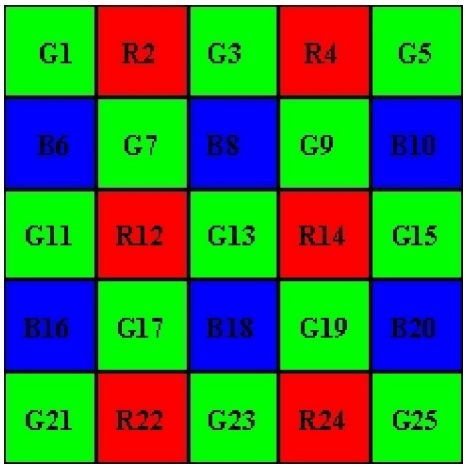
R_{sum} = (R13 + R23) / 2 + (R11 + R13) / 2 + (R13 + R5) / 2 + (R13 + R25) / 2;

4. Temukan perbedaan warna dengan membagi selisih dua nilai rata-rata (sum) dengan nomor gradient yang berada pada threshold subset, contohnya;

G13 = R13 + (G_{sum} - R_{sum}) / 4; B13 = R13 + (B_{sum} - R_{sum}) / 4

5. Gunakan hasilnya untuk menyusun dua komponen warna lainyang hilang.

6. Interpolasi pixel B atau R pada pixel G, saat ini kita akan mengestimasi R13 dan B13 pada G13 .



Gambar 3.4 Penomoran Warna Pixel

7. Dari 8 gradient, maka diperoleh perhitungan matematis sebagai berikut :

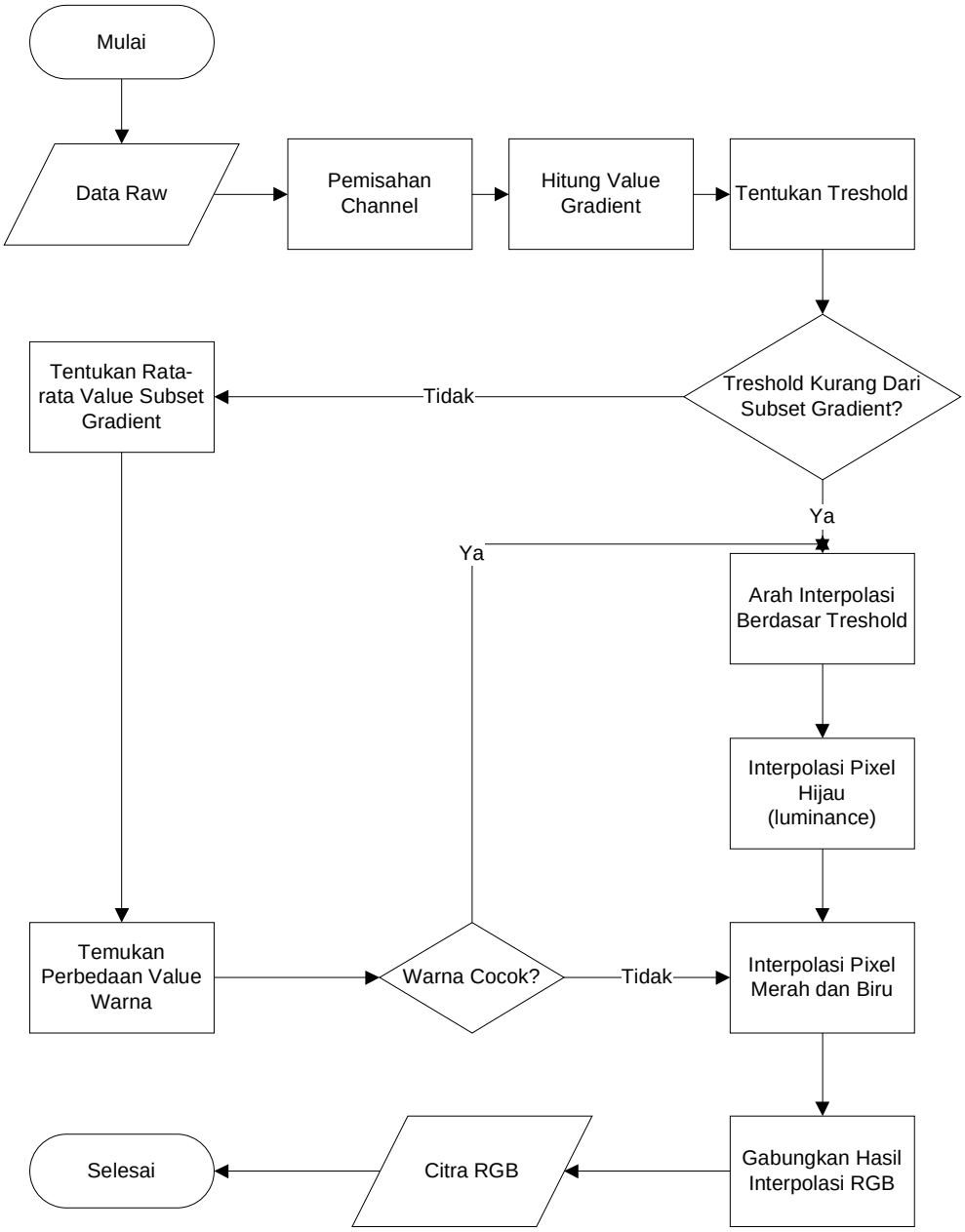
```
Gradient N = |G3 - G13| + |B8 - B18| + |G7 - G17| / 2 +
|G9 - G19| / 2 + |R2 - R12| / 2 + |R4 - R14| / 2 ;
Gradient E = |R14 - R12| + |G15 - G13| + |G9 - G7| / 2
+ |G19 - G17| / 2 + |B10 - B8| / 2 + |B20 - B18| / 2;
Gradient S = |B18 - B8| + |G23 - G13| + |G19 - G9| / 2
+ |G17 - G7| / 2 + |R24 - R14| / 2 + |R22 - R12| / 2;
Gradient W = |R12 - R14| + |G11 - G13| + |G17 - G19| /
2 + |G7 - G9| / 2 + |B16 - B18| / 2 + |B6 - B8| / 2;
Gradient NE = |G9 - G17| + |G5 - G13| + |R4 - R12| +
|B10 - B18|;
Gradient SE = |G19 - G7| + |G25 - G13| + |B20 - B8| +
|R24 - R12|;
Gradient NW = |G7 - G19| + |G1 - G13| + |B6 - B18| +
```

```
|R2 - R14|;
Gradient SW = |G17 - G9| + |G21 - G13| +|R22 - R14| +
|B16 - B8|;
```

8. Lalu temukan lokasi pixel pada *region* yang sama dan gunakan untuk menentukan perbedaan warna antara *center pixel* dan warna yang akan di interpolasi, selanjutnya tambahkan warna yang didapat pada bagian pixel yang hilang.

3.4. Flowchart interpolasi algoritma Variable Number Of Gradient

Berikut ini adalah flowchart interpolasi menggunakan Algoritma VNG. Data Raw mengalami pemisahan channel terlebih dahulu lalu dihitung value gradient dari masing-masing piksel untuk menentukan treshold kemudian dilihat apakah treshold bernilai kurang dari subset gradient, jika ya maka arah interpolasi berdasar arah treshold dan berlanjut pada penggabungan masing-masing piksel menjadi citra RGB. Jika treshold bernilai lebih maka subset gradient dihitung rata-ratanya untuk dicocokan warna piksel dengan treshold. Jika warnanya cocok maka interpolasi berdasar arah treshold, jika tidak mulai interpolasi pada piksel *green* sebagai luminance dilanjutkan dengan piksel *green, blue* dan *red.* Hasil interpolasi masing-masing piksel kemudian digabungkan menjadi citra RGB.



Gambar 3.5 Flowchart Algoritma VNG

3.5. Interpolasi Algoritma Adaptive Homogeneity Directed

Sekilas algoritma Adaptive Homogeneity Directed ini merupakan penggabungan antara algoritma Bilinear dan algoritma VNG. Interpolasi secara

horizontal maupun interpolasi secara vertikal dari pixel warna R,G dan B. pertama, metode untuk interpolasi pixel G dari sampel pixel G dan R.

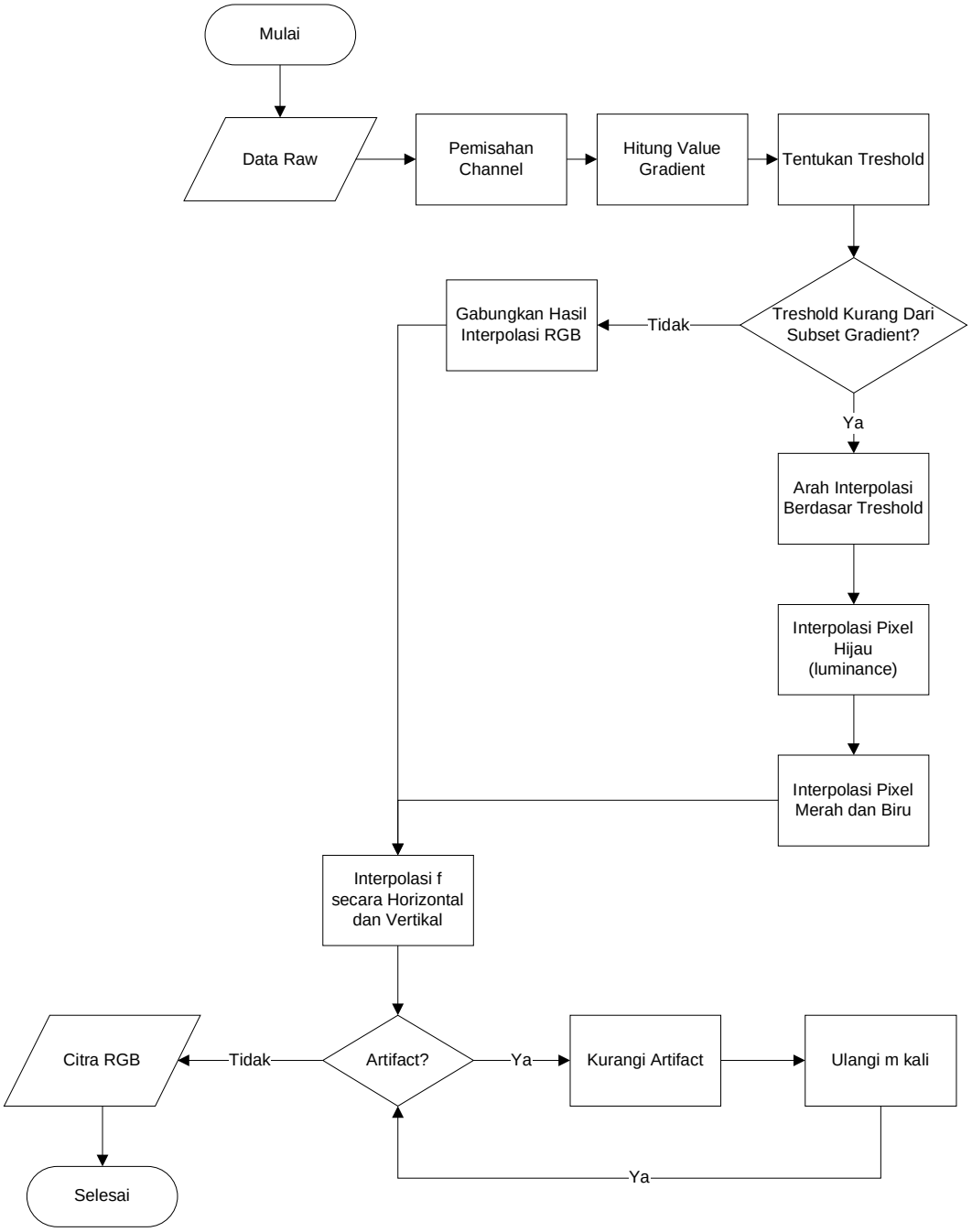
1. Interpolasikan f_H (horizontal) dan f_V (vertikal)
2. Gunakan teknik pengurangan *artifact* interpolasi ke f

Perbedaan mencolok ada pada penanganan *artifact* yang mungkin timbul dengan asumsi R-G dan B-G menggunakan treshold yang didapat dari median dan diulang *m* kali.

1. R = median (R - G) + G
2. B = median (B - G) + G
3. G + ½ (median(G-R) + median (G – B) + R + B)

3.6. Flowchart interpolasi menggunakan algoritma Adaptive Homogeneity Directed

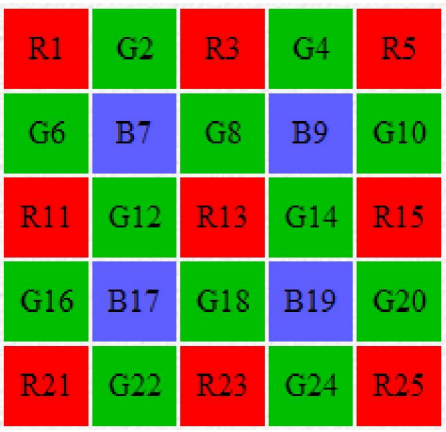
Berikut ini adalah flowchart interpolasi menggunakan Algoritma AHD. Data Raw mengalami pemisahan channel terlebih dahulu lalu dihitung value gradientnya untuk menentukan treshold. Jika treshold kurang dari subset gradient maka arah interpolasi berdasar arah treshold. mulai interpolasi pada piksel *green* sebagai luminance dilanjutkan dengan piksel *green, blue* dan *red.* Jika value treshold lebih dari subset gradient maka langsung mengalami interpolasi pada piksel green sebagai luminance dilanjutkan dengan piksel green, blue dan red secara vertikal dan horizontal. Pada algoritma AHD, tahap ini dilanjutkan dengan pemeriksaan apakah terdapat indikasi *artifact* pada gambar, jika ya maka akan terjadi pengurangan *artifact* sampai tidak ditemukan indikasi tersebut. Hasil interpolasi masing-masing piksel kemudian digabungkan menjadi citra RGB.



Gambar 3.6 Flowchart Algoritma AHD

3.7. Interpolasi menggunakan Algoritma Patterned Pixel Group

Algoritma ini merupakan modifikasi dari algoritma Variable Number of Gradient dengan tujuan mengurangi *artifact* yang bisa muncul pada algoritma VNG. Sebuah gambar biasanya terdiri dari kumpulan pixel group, contohnya sebuah huruf A berwarna biru pada backround merah terdiri dari group pixel biru dan group pixel merah. Pixel yang berada pada satu group memiliki tingkat kecerahan dan warna yang sama. Dengan mengidentifikasi pixel pada group yang sama kita bisa memperkirakan pixel yang hilang dan meningkatkan kualitas gambar terutama pada ketajaman tepi. *Channel* warna hijau di interpolasi melalui identifikasi pixel group sedangkan *channel* lainnya yaitu merah dan biru dihasilkan melalui *hybrid color correction* dan *smooth hue-transition* seperti pada algoritma VNG.



Gambar 3.7 Penomoran Setiap Pixel

2. Interpolasi nilai G pada pixel R atau B dengan pixel acuan G13 (R13) kalkulasikan gradient ke arah atas (N), kanan (E), kiri (W) dan bawah (S) dengan perhitungan matematis sebagai berikut;

```
ΔN  =    |R3-R13|×2+|G8-G18|
ΔE  =    |R13-R15|×2+|G12-G14|
ΔW  =    |R11-R13|×2+|G12-G14|
ΔS  =    |R13-R23|×2+|G8-G18|
```

3. Temukan ke arah mana gradient terkecil dan gunakan untuk memperkirakan G13 sebagai berikut;

```
ΔN is minimum:G13   = (G8×3+R13+G18-R3)/4
ΔE is minimum:G13   = (G14×3+R13+G12-R15)/4
ΔW is minimum:G13   = (G12×3+R13+G14-R11)/4
ΔS is minimum:G13   = (G18×3+R13+G8-R23)/4
```

4. Selanjutnya interpolasi pixel biru dan merah pada pixel hijau, contohnya B8 dan R8 pada G8

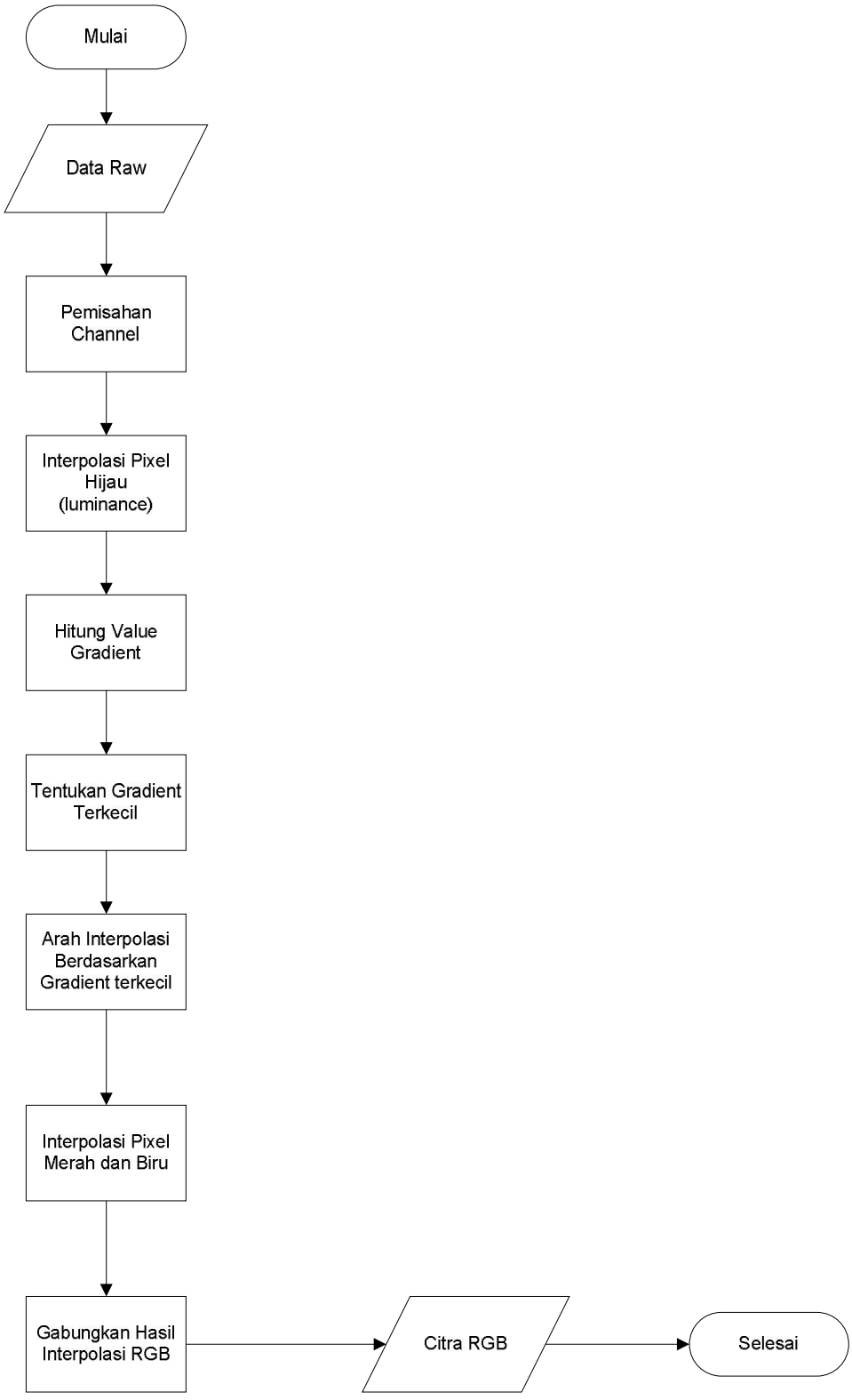
```
B8 = hue_transit(G7, G8, G9, B7, B9)
R8 = hue_transit(G3, G8, G13, R3, R13)
function hue_transit (l1, l2, l3, v1, v3) =
 if (l1<l2<l3 or l1>l2>l3)
 then return v1+(v3-v1)×(l2-l1)/(l3-l1)
 else return (v1+v3)/2+(l2×2-l1-l3)/4
```

5. Interpolasi berikutnya pixel biru pada pixel merah dan pixel merah pada pixel biru, dengan cara yang sama dengan algoritma VNG yaitu ke 8 penjuru yaitu gradient utara (N), timur (E), selatan (S), barat (W), timur laut (NE), barat laut (NW), tenggara (SE), barat daya (SW). Misalnya mencari nilai B13 pada R13.

```
ΔNE = |B9-B17|+|R5-R13|+|R13-R21|+|G9-G13|+|G13-G17|
ΔNW = |B7-B19|+|R1-R13|+|R13-R25|+|G7-G13|+|G13-G19|
if (ΔNE ≤ ΔNW)
then B13 = hue_transit(G9, G13, G17, B9, B17)
else B13 = hue_transit(G7, G13, G19, B7, B19)
```

3.8. Flowchart interpolasi menggunakan algoritma Patterned Pixel Group

Berikut ini adalah flowchart interpolasi menggunakan Algoritma PPG. Data Raw mengalami pemisahan channel terlebih dahulu lalu mulai interpolasi pada piksel *green* sebagai luminance kemudian dihitung value gradientnya untuk dideteksi gradient terkecil karena pada PPG arah interpolasi mengarah pada gradient terkecil. Interpolasi dilanjutkan dengan piksel *green, blue* dan *red.* Hasil interpolasi masing-masing piksel kemudian digabungkan menjadi citra RGB.



Gambar 3.8 Flowchart Algoritma PPG

3.9. Analisis kebutuhan non fungsional

Analisis kebutuhan non fungsional dalam pembangunan perangkat lunak ini, terdiri dari; analisis pengguna, perangkat keras dan perangkat lunak.

3.9.1. Analisis pengguna

Pengguna yang memiliki file RAW dari kamera digital dan ingin melihat dan atau mengkonversi RAW ke format lain yang lebih umum dan mengerti istilah-istilah grafis dan fotografi.

3.9.2. Analisis perangkat lunak

Sistem yang akan dibangun, diharapkan memiliki kemudahan dan fleksibilitas penggunaan. Selain digunakan pada *desktop computer* dapat pula digunakan pada *notebook.*

Perangkat lunak harus dapat digunakan untuk pembangunan aplikasi dan berbagai algoritma yang dianalisa serta mendukung pengembangan aplikasi ini kedepannya jika dibutuhkan. Perangkat lunak yang digunakan dalam pembangunan aplikasi ini adalah:

1. Sistem operasi Windows XP Service Pack 3.
2. Borland Delphi 7.0.

3.9.3. Analisis perangkat keras

Perangkat keras *(hardware)* adalah seluruh komponen peralatan yang digunakan untuk menunjang pembangunan sistem. Adapun perangkat keras yang digunakan untuk pemakaian perangkat lunak ini secara optimal memerlukan spesifikasi minimum sebagai berikut :

1. *Processor* dengan kecepatan minimum 1,4 GHz
2. RAM berkapasitas 512 MB
3. Harddisk berkapasitas 40 GB
4. VGA onboard
5. Monitor dengan resolusi WXGA (1280 x 800)
6. *Memory card reader*
7. USB port.

Perangkat keras yang digunakan sebagai implementasi atau simulasi dari program yang akan dibangun yaitu *notebook* dengan spesifikasi sebagai berikut:

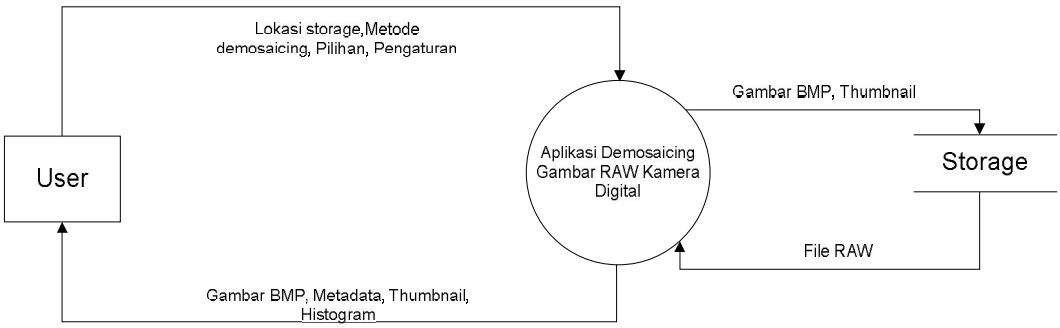
1. *Processor* Intel Core 2 Duo 2.0 GHz
2. Harddisk 160 GB
3. Monitor WXGA 14.1”
4. RAM 2048 MB
5. DVD RW
6. VGA Nvidia GeForce 8600M GS 256 MB
7. Port USB
8. *Memory card reader*

3.10. Perancangan sistem

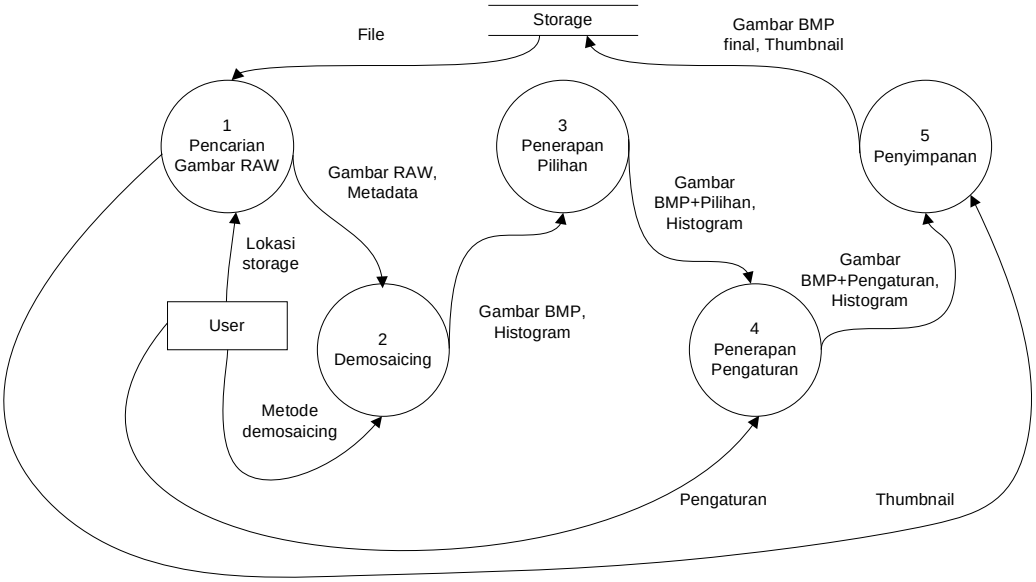
Setelah melakukan analisis sistem, maka tahapan selanjutnya adalah membuat suatu perancangan sistem. Tujuan dari pada perancangan sistem secara garis besar adalah untuk menghasilkan bentuk rancangan yang dapat memenuhi kebutuhan akan penyelesaian rumusan masalah secara cepat dan benar.

3.11. Perancangan fungsional

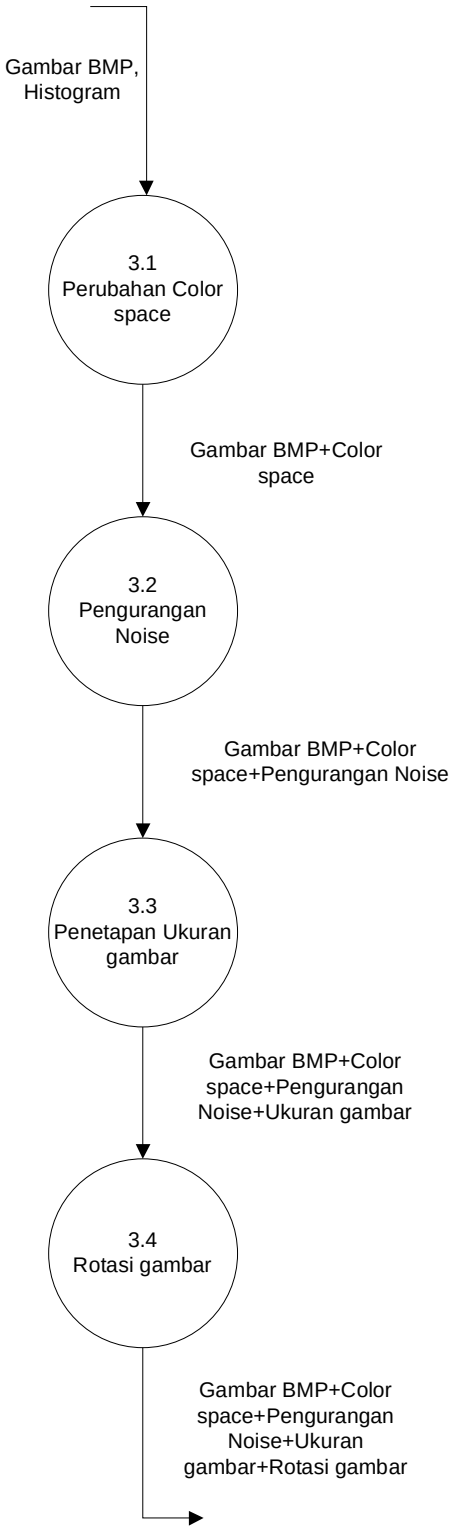
Perancangan fungsional ini meliputi pembuatan diagram konteks (*context diagram*), dan diagram aliran data (*data flow diagram*).



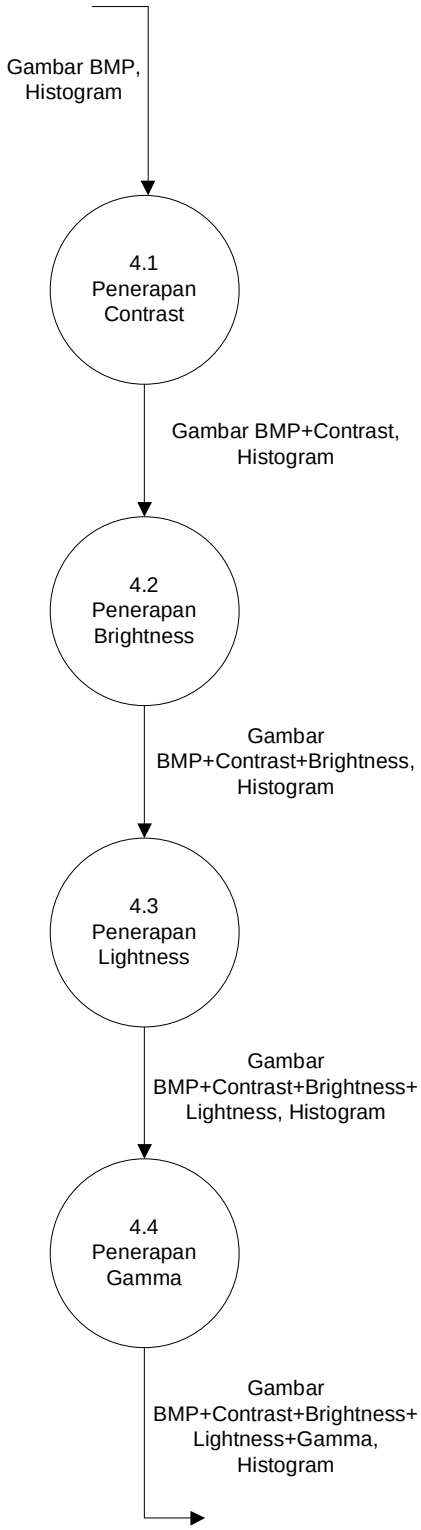
Gambar 3.9 Diagram Konteks



Gambar 3.10 Data Flow Diagram Level 1



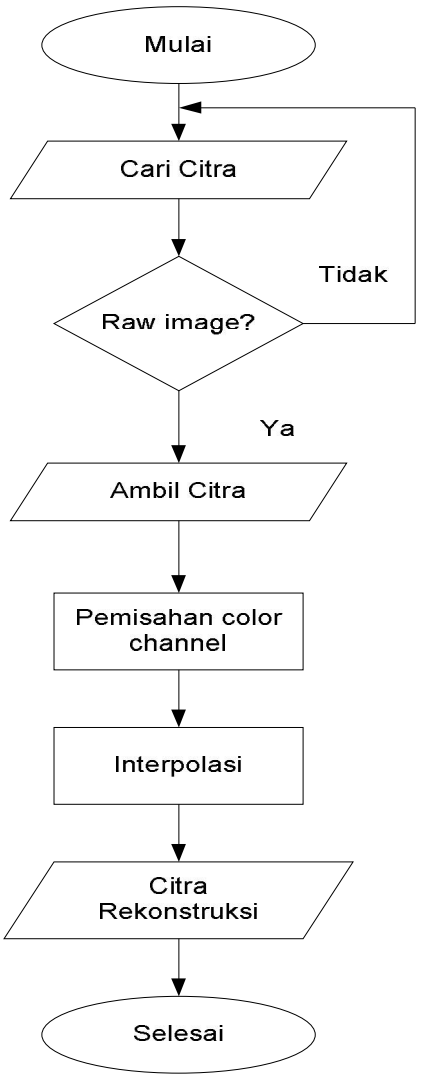
Gambar 3.11 Data Flow Diagram Level 2 Proses 3



Gambar 3.12 Data Flow Diagram Level 2 Proses 4

3.11.1. Flowchart sistem secara keseluruhan

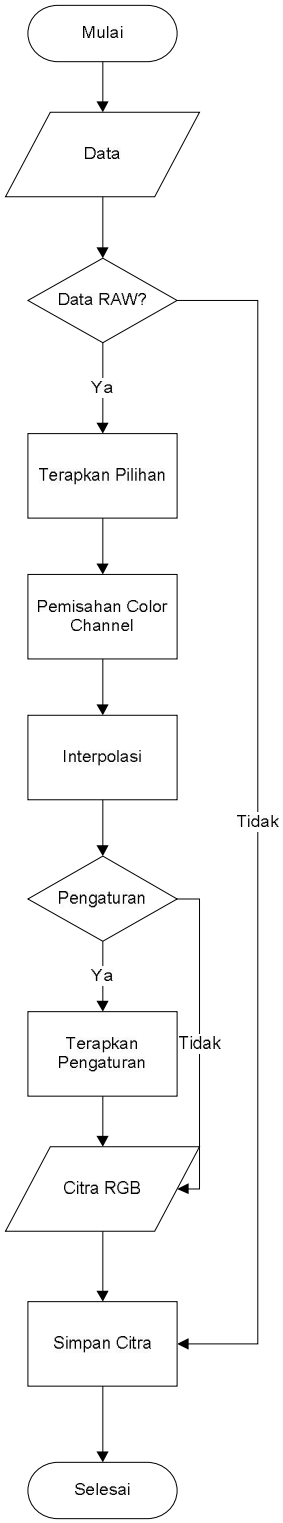
Berikut ini adalah flowchart sistem secara menyeluruh. Dimulai dengan pencarian manual citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat dan dilakukan demosaicing dengan proses pemisahan *channel* warna terlebih dahulu sebelum dilakukan algoritma interpolasi.



Gambar 3.14 Flowchart Sistem

3.11.2. Flowchart interpolasi gambar

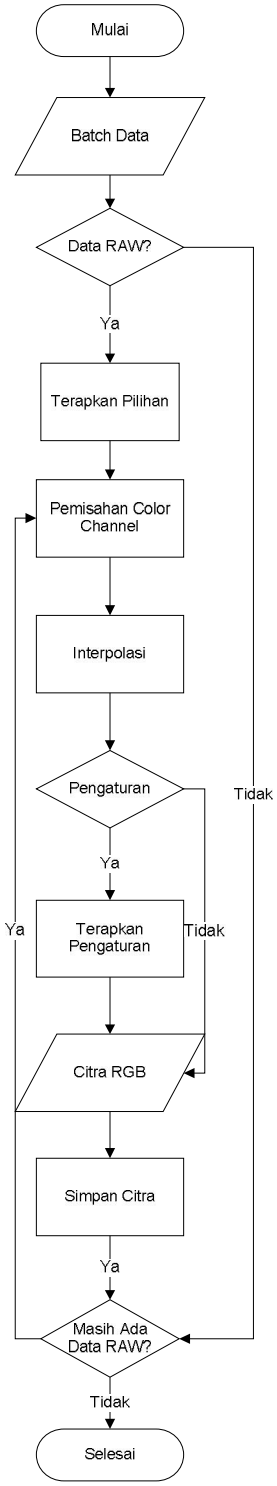
Berikut ini adalah flowchart interpolasi gambar. Dimulai dengan pencarian manual citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat. User dapat terlebih dahulu memilih salah satu algoritma demosaicing yang disediakan. Setelah proses interpolasi selesai user dapat melakukan manipulasi gambar berupa *contrast, lightness, brightness* dan *gamma* untuk kemudian menghasilkan citra .BMP untuk disimpan.



Gambar 3.15 Flowchart Interpolasi Gambar

3.11.3. Flowchart batch folder

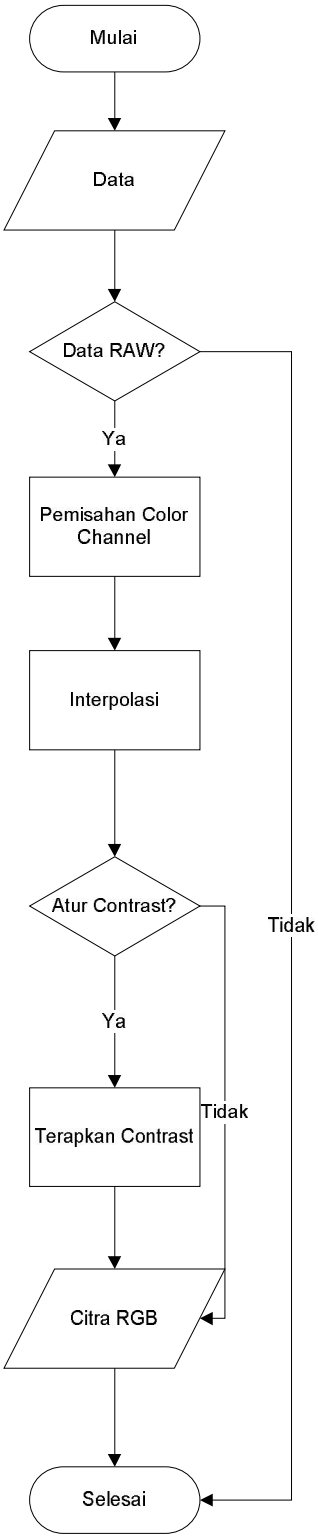
Berikut ini adalah flowchart interpolasi gambar secara batch, yaitu memproses beberapa file Raw sekaligus dalam sebuah folder. Dimulai dengan pencarian manual folder yang memuat citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat. User dapat terlebih dahulu memilih salah satu algoritma demosaicing yang disediakan. Sebelum proses demosaicing secara batch berlangsung user dapat melakukan manipulasi gambar berupa *contrast, lightness, brightness* dan *gamma.* Aplikasi akan melakukan demosaicing satu persatu file Raw di folder tersebut hingga tidak ditemukan lagi file Raw yang belum di interpolasi. Proses ini menghasilkan citra .BMP yang otomatis tersimpan di folder yang sama.



Gambar 3.16 Flowchart Batch Folder

3.11.4. Flowchart pengaturan *contrast*

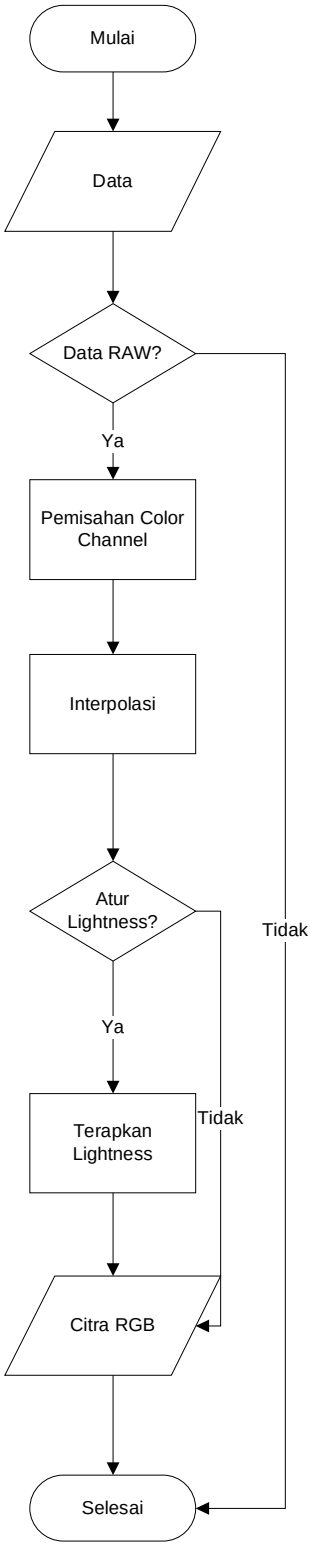
Berikut ini adalah flowchart pengaturan contrast setelah dilakukan interpolasi gambar. Dimulai dengan pencarian manual citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat. User dapat terlebih dahulu memilih salah satu algoritma demosaicing yang disediakan. Setelah proses interpolasi selesai user dapat melakukan manipulasi gambar berupa *contrast* untuk kemudian menghasilkan citra .BMP yang telah dimanipulasi untuk disimpan.



Gambar 3.17 Flowchart Pengaturan Contrast

3.11.5. Flowchart pengaturan *lightness*

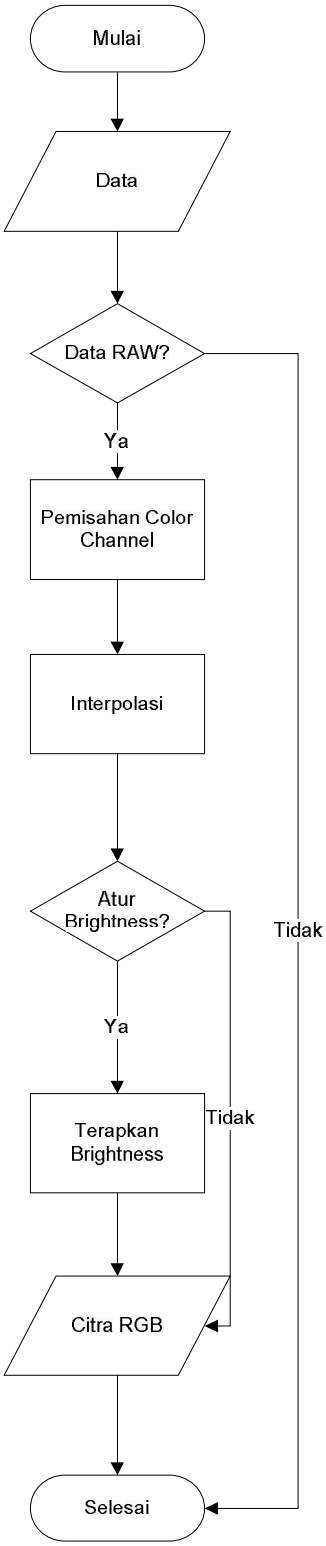
Berikut ini adalah flowchart pengaturan *lightness* setelah dilakukan interpolasi gambar. Dimulai dengan pencarian manual citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat. User dapat terlebih dahulu memilih salah satu algoritma demosaicing yang disediakan. Setelah proses interpolasi selesai user dapat melakukan manipulasi gambar berupa *lightness* untuk kemudian menghasilkan citra .BMP yang telah dimanipulasi untuk disimpan.



Gambar 3.18 Flowchart Pengaturan Lightness

3.11.6. Flowchart pengaturan *brightness*

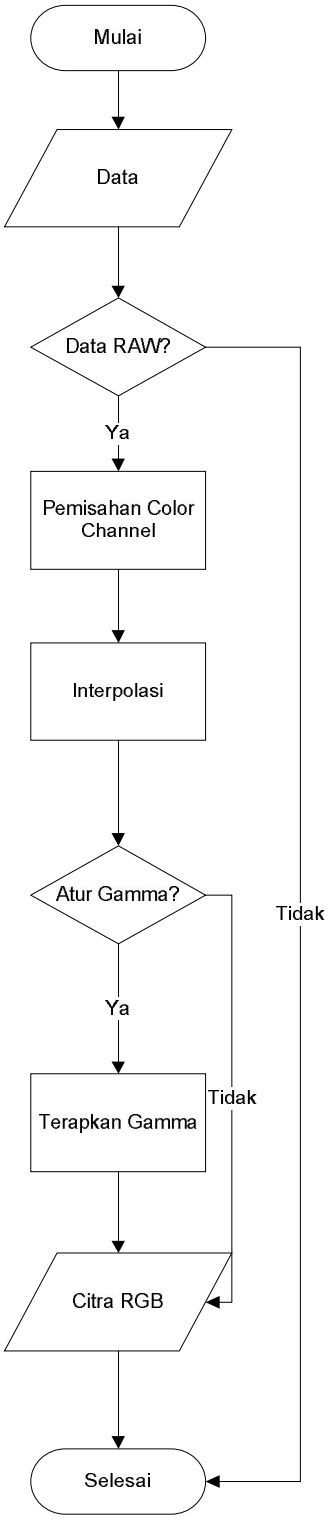
Berikut ini adalah flowchart pengaturan *brightness* setelah dilakukan interpolasi gambar. Dimulai dengan pencarian manual citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat. User dapat terlebih dahulu memilih salah satu algoritma demosaicing yang disediakan. Setelah proses interpolasi selesai user dapat melakukan manipulasi gambar berupa *brightness* untuk kemudian menghasilkan citra .BMP yang telah dimanipulasi untuk disimpan.



Gambar 3.19 Flowchart Pengaturan Brightness

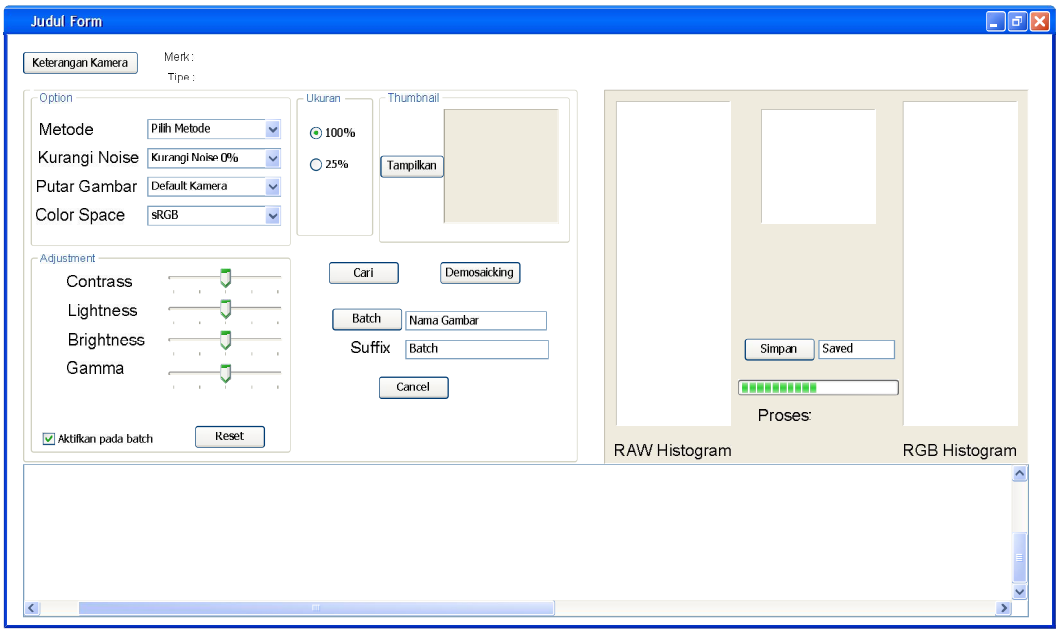
3.11.7. Flowchart pengaturan *gamma*

Berikut ini adalah flowchart pengaturan *gamma* setelah dilakukan interpolasi gambar. Dimulai dengan pencarian manual citra oleh user, aplikasi akan mendeteksi apakah citra merupakan format Raw atau bukan. Hanya format Raw yang bisa dimuat. User dapat terlebih dahulu memilih salah satu algoritma demosaicing yang disediakan. Setelah proses interpolasi selesai user dapat melakukan manipulasi gambar berupa *gamma* untuk kemudian menghasilkan citra .BMP yang telah dimanipulasi untuk disimpan.



Gambar 3.20 Flowchart Pengaturan Gamma

3.11.8. Perancangan Antar Muka Utama



Gambar 3.21 Perancangan Menu Utama

Tabel 3.3 Keterangan tampilan antar muka utama

No	Nama	Keterangan	Navigasi
1	Keterangan kamera	Tombol untuk menampilkan keterangan kamera	Jika di-klik maka merk dan tipe kamera akan ditampilkan
2	Merk	Label1	
3	Tipe	Label2	
4	Pilih metode	ComboBox	Jika di-klik maka combo box menampilkan pilihan metode/algortima demosaicing

5	Kurangi noise	ComboBox	Jika di-klik maka combo box menampilkan pilihan pengurangan noise
6	Color space	ComboBox	Jika di-klik maka combo box menampilkan pilihan *color space*
7	Putar gambar	ComboBox	Jika di-klik maka combo box menampilkan pilihan putar gambar
8	contrast	slider	Jika digeser akan mengubah contrast
9	lightness	slider	Jika digeser akan mengubah lightness
10	brightness	slider	Jika digeser akan mengubah brightness
11	gamma	slider	Jika digeser akan mengubah gamma
12	Aktifkan pada batch	CheckBox	Jika ditandai akan mengaktifkan slider adjustment
13	Reset	Tombol untuk reset slider.	Jika di-klik akan mereset slider ke posisi default
14	Ukuran gambar	Radiobutton untuk memilih ukuran yang akan di save	Jika dipilih salah satu, file bisa di save dengan ukuran tersebut.
15	thumbnail	Gambar tampilan thumbnail RAW	
16	tampilkan	Tombol untuk	Jika di-klik akan menampilkan

		menampilkan thumbnail data gambar RAW	thumbnail data gambar RAW
17	Cari	Tombol untuk mencari data RAW	Jika di-klik akan membuka windows explorer
18	Demosaicing	Tombol untuk demosaicing	Klik untuk proses demosaicing
19	Batch	Tombol untuk memulai batch	Jika di-klik membuka batch folder
20	Nama gambar	Menampilkan file yang di batch	
21	Batch suffix	Menamai batch	
22	Histogram RAW	Menampilkan histogram RAW 16bit	
23	Histogram RGB	Menampilkan histogram RGB 8bit	
24	Image 2	Menampilkan gambar ukuran kecil	
25	Simpan sebagai	Tombol simpan	Jika di-klik gambar yang telah di interpolasi disimpan
26	saved	Menamai file	
27	Progress bar	Menampilkan proses yang berlangsung	
28	resolusi	Label menampilkan resolusi gambar	