

BAB V

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini dibahas mengenai implementasi perangkat lunak berdasarkan hasil yang telah didapatkan dari perancangan perangkat lunak yang telah dibuat. Implementasi merupakan proses transformasi hasil perancangan perangkat lunak ke dalam kode (*coding*) sesuai dengan sintaks dari bahasa pemrograman yang digunakan. Pembahasan terdiri dari penjelasan tentang Lingkungan Implementasi, Algoritma Implementasi, dan Implementasi Antarmuka Aplikasi.

5.1 Lingkungan Implementasi

Sistem dibuat dengan menggunakan aplikasi pemrograman Borland Delphi 7.0. Sistem diimplementasikan dengan menggunakan spesifikasi sebagai berikut:

5.1.1 Spesifikasi Perangkat Keras

- CPU : AMD Turion™ 64 RM-70 (2.0 GHz)
- Memory : 2 GB RAM
- Hard Disk : 160 GB
- Graphics Adapter : Nvidia GeForce 9100M G

5.1.2 Spesifikasi Perangkat Lunak

- Sistem operasi : Windows XP
- Aplikasi yang digunakan : Borland Delphi 7

5.2 Algoritma Sistem

Penyajian yang digunakan berupa urutan baris algoritma seperti kode pemrograman dan parameter yang digunakan dalam menggunakan class yang telah disediakan. Semua proses yang dijelaskan dalam bentuk potongan listing program dan parameternya. Tahap Implementasi ini berdasarkan perancangan sistem pada tahap perancangan.

5.2.1 Implementasi Input Gambar

Sebelum Melakukan Penginputan gambar perlu dipastikan bahwa gambar masukan berupa gambar graylevel 8 bit dengan format bitmap(*.bmp). Cara paling sederhana dalam melakukan penginputan gambar pada Delphi adalah dengan memanfaatkan komponen open dialog. Berikut ini adalah listing programnya:

```
procedure TForm1.Button2Click(Sender: TObject);
var
currentfile:string;
begin
if opendialog1.Execute then
begin
currentfile:=opendialog1.FileName;
image1.Picture.LoadFromFile(currentfile);
end;
end;
```

Currentfile merupakan variable bertipe string yang berfungsi untuk menampung nama file yang kita pilih saat dialog window terbuka. Sedangkan Image1 merupakan komponen Delphi yang berfungsi untuk menampilkan gambar pada form.

5.2.2 Implementasi Pencarian Nilai Pixel.

Untuk mendapatkan nilai intensitas suatu pixel pada Delphi, kita bisa menggunakan fungsi scanline. Scanline berfungsi untuk membaca nilai-nilai intensitas pixel dalam suatu baris dari array pixel gambar. Hasil dari scanline ini akan disimpan pada variable bertipe pbytearray (pointer to byte array). Pbytearray sendiri merupakan array 1 dimensi yang ukurannya bersifat dinamis dan berfungsi untuk menyimpan byte-byte informasi nilai pixel pada suatu baris pada gambar. Untuk implementasinya dalam Delphi adalah sebagai berikut:

```
procedure TForm1.Button1Click(Sender: TObject);
var
i,j,a:integer;
b:string;
buffer:pbytearray;
begin
memo1.Clear;
for i:=0 to image1.Picture.Height-1 do
begin
buffer:=image1.Picture.Bitmap.ScanLine[i];
b:='';
for j:=0 to (image1.Picture.Width*4)-1 do
begin
a:=buffer[j];
b=b+' '+inttostr(a);
end;
memo1.Lines.Add(b);
memo1.Lines.Add('#####');

end;
end;
```

Implementasi untuk Menampilkan nilai pixel suatu gambar

Setelah mendapatkan nilai intensitas tiap pixel yang disimpan pada variabel bernama buffer, selanjutnya akan ditampilkan pada form dengan menggunakan komponen memo. Komponen memo pada delphi berfungsi untuk menampilkan sejumlah data bertipe string dengan kapasitas yang lebih besar daripada komponen label.

5.2.3 Implementasi Penghitungan Integral Image

Untuk melakukan penghitungan integral image, pada baris pertama ($y=0$) cukup dengan melakukan penambahan pixel-pixel yang dilewati pada saat melakukan scanline. Sedangkan untuk mencari nilai integral image pada baris-baris berikutnya diperlukan satu variable lagi untuk menyimpan hasil penambahan pada baris tersebut (untuk lebih jelasnya dibahas pada bab perancangan). Sedangkan implementasinya pada Delphi adalah sebagai berikut:

```

procedure TForm1.Button5Click(Sender: TObject);
var
i,j,temp,absol,jmlh_array:longint;
a,b:string;
ptr:PByteArray;
begin
if image1.picture.Height>image1.Picture.Width
then
jmlh_array:=image1.Picture.Height
else
jmlh_array:=image1.Picture.Width;
SetLength(sum,jmlh_array,jmlh_array);
SetLength(img,jmlh_array,jmlh_array);
SetLength(threshold,jmlh_array,jmlh_array);
for i:=0 to (Image1.picture.Height-1) do
begin
temp:=0;
a:='';
b:='';
ptr:=Image1.Picture.Bitmap.ScanLine[i];
for j:=0 to (Image1.picture.Width-1) do
begin
img[i,j]:=ptr[4*j];
if i>0
then sum[i,j]:=sum[i-1,j]+ptr[4*j]+temp
else
sum[i,j]:=ptr[4*j]+temp;
temp:=temp+ptr[4*j];
b:=b+' '+inttostr(sum[i,j]);
a:=a+' '+inttostr(ptr[4*j]);
end;
memo1.Lines.Add(a);
memo1.Lines.Add('#####');
memo2.Lines.Add(b);
memo2.Lines.Add('#####');
end;
end;
end;

```

Implementasi Penghitungan Integral Image

5.2.4 Implementasi Thresholding

Implementasi thresholding dalam aplikasi ini meliputi 2 sub proses, yaitu penghitungan nilai threshold dan thresholding itu sendiri (konversi gambar menjadi citra biner). Penghitungan nilai threshold dilakukan dengan menggunakan metode integral image sedangkan thresholdingnya sendiri dengan melakukan perbandingan antara nilai yang diperoleh dari penghitungan integral image dengan nilai pixel itu sendiri yang dikalikan dengan jumlah pixel pada luasan yang telah ditentukan.

Proses penentuan nilai threshold diawali dengan menentukan luas daerah yang akan dibandingkan dengan suatu pixel. Luas ini kita notasikan dengan $s \times s$ yang berarti panjang sisi luasan tersebut adalah $s + 1$ karena ditambahkan dengan pixel itu sendiri. Kemudian setelah ditentukan luasannya adalah melakukan penghitungan total intensitas pixel pada luasan tersebut dengan menggunakan metode integral image. Hasil penghitungan tersebut selanjutnya kita simpan pada variable bertipe integer bernama 'th'.

Proses thresholding dilakukan dengan membandingkan suatu pixel dengan nilai pada variable th. Karena nilai pada variable th merupakan hasil penjumlahan dari nilai intensitas pixel pada luasan yang telah ditentukan, maka nilai piksel yang akan dibandingkan dengan variable tersebut harus dikalikan dengan jumlah piksel pada luasan yang sama ($(s+1) \times (s+1)$). Berikut ini adalah tampilan code explorer pada Delphi untuk implementasi thresholding.

```
procedure TForm1.Button6Click(Sender: TObject);
var
  i,j,x1,x2,y1,y2,s,count,th:integer;
  a,b,c,d:integer;
  ptr:PByteArray;
begin
  s:=40;
  for i:=0 to (Image1.picture.Height-1) do
  begin
    ptr:=Image1.Picture.Bitmap.ScanLine[i];
    for j:=0 to (Image1.picture.Width-1) do
    begin
      th:=0;
      x1:=round(i-s/2);
      x2:=round(i+s/2);
      y1:=round(j-s/2);
      y2:=round(j+s/2);

      if x1< 0 then x1:=0;
      if y1 < 0 then y1:=0;
      if x2 > (Image1.picture.height-1) then x2:=(Image1.picture.height-1);
      if y2 > (Image1.picture.Width-1) then y2:=(Image1.picture.Width-1);
      count:=(x2-x1)*(y2-y1);
      if x1-1 < 0 then
        begin
          if y1-1 < 0 then th:=sum[x2,y2]
          else th:=sum[x2,y2]-sum[x2,(y1-1)];
        end
      else
        begin

```

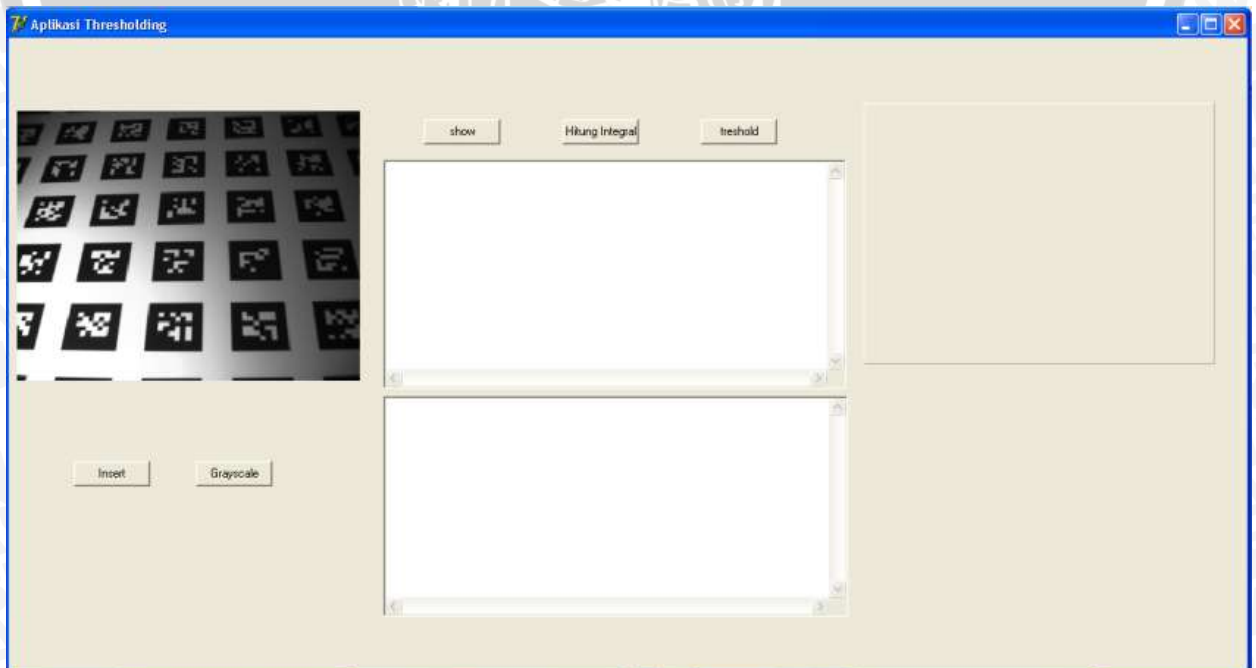
```
if y1-1 < 0 then th:=sum[x2,y2]-sum[(x1-1),y2]
else th:=sum[x2,y2]+sum[(x1-1),(y1-1)]-sum[(x1-1),y2]-sum[x2,(y1-1)]
end;

threshold[i,j]:=th;
if (ptr[4*j]*count)<=(th*85/100) then
begin
ptr[4*j]:=0;
ptr[4*j+1]:=0;
ptr[4*j+2]:=0;
end
else
begin
ptr[4*j]:=255;
ptr[4*j+1]:=255;
ptr[4*j+2]:=255;
end;
end;
end;
image2.Picture.Bitmap:=image1.Picture.bitmap;
showmessage('proses thresholding berhasil!');
end;
```

Implementasi Thresholding

5.3 Implementasi Antarmuka Aplikasi Thresholding

Berikut ini adalah tampilan antarmuka dari aplikasi thresholding dengan menggunakan metode integral image.



Gambar 5.1 Tampilan menu utama program


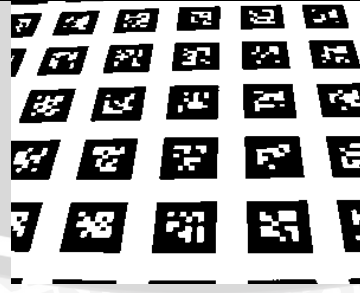
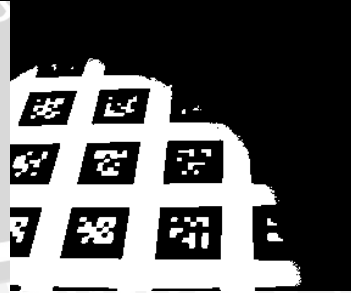
Terdapat 5 tombol pada form aplikasi yang fungsinya adalah sebagai berikut:

- Insert : Untuk memasukkan gambar Input.
- Grayscale : Sebagai fitur tambahan apabila gambar yang dimasukkan belum berupa gambar graylevel 8bit
- Show : Untuk menampilkan intensitas piksel gambar input.
- Hitung Integral: Untuk menghitung integral image dari gambar input dan menampilkannya pada form aplikasi.
- Threshold : Untuk melakukan proses thresholding.

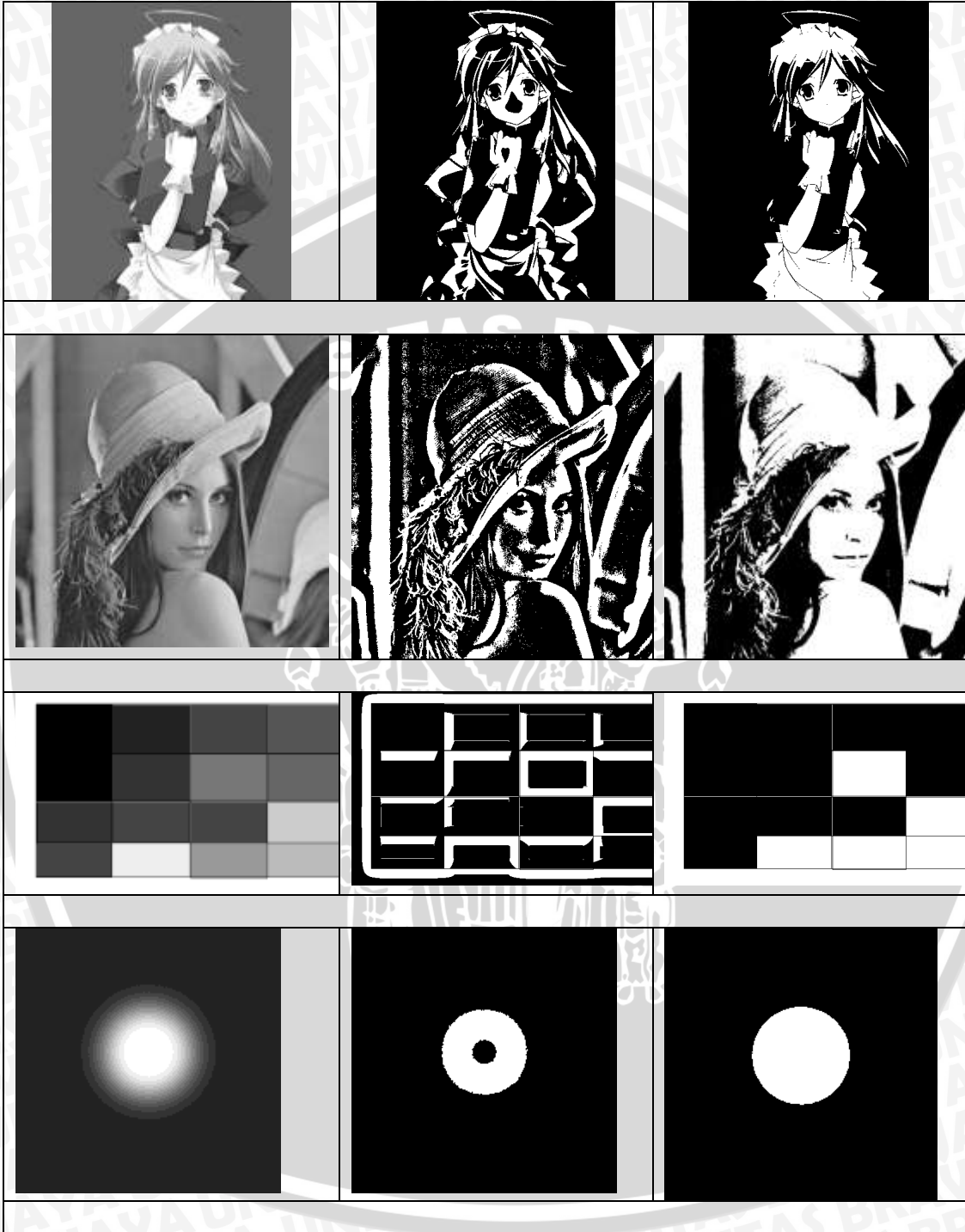
5.4 Pengujian

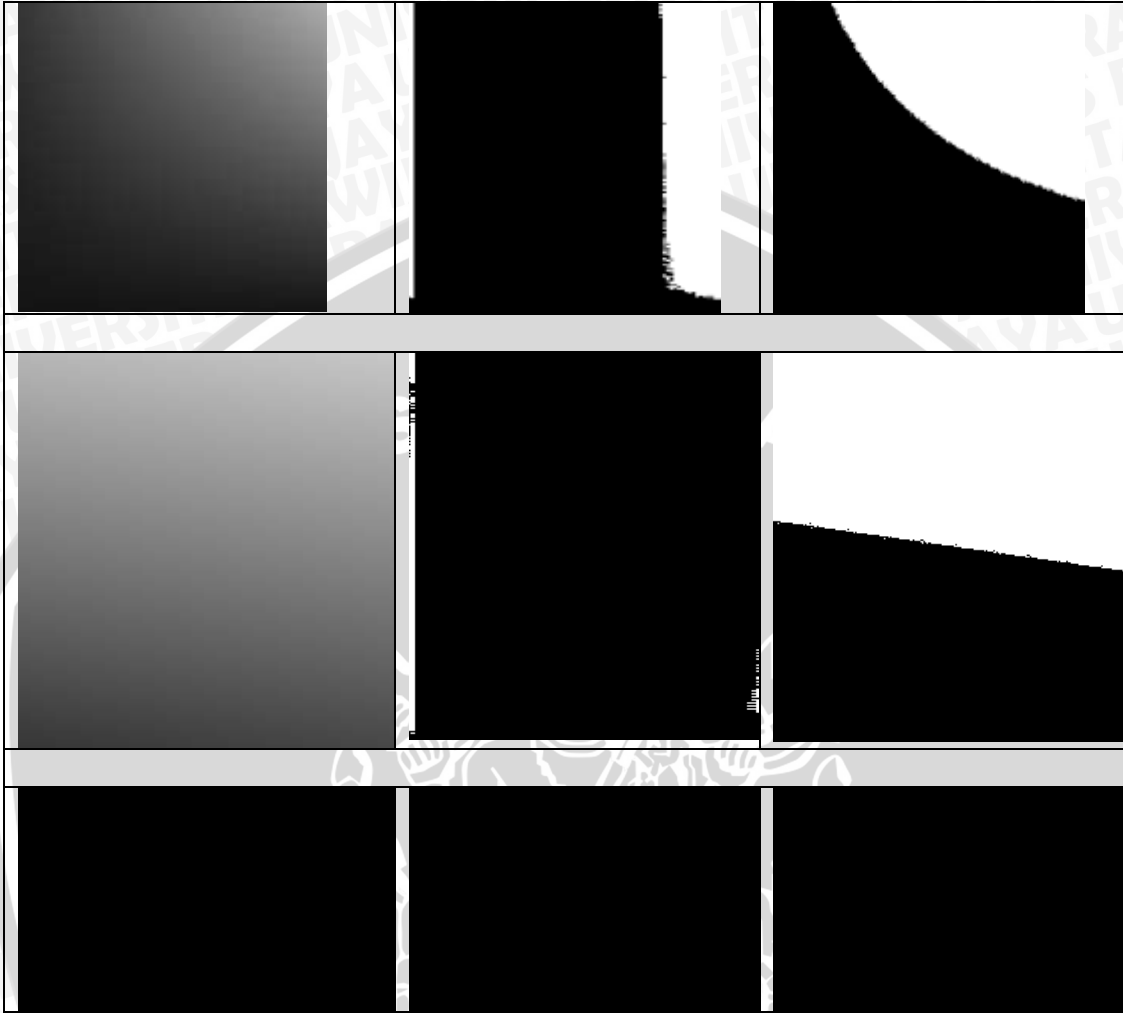
Pengujian aplikasi ini dilakukan dengan membandingkan hasil thresholding dari aplikasi ini dengan aplikasi pengolah citra yang umum digunakan. Dalam Hal ini aplikasi pengolah citra yang umum digunakan adalah GIMP versi 2.6 karena merupakan aplikasi freeware yang memiliki fitur untuk menghasilkan gambar hasil thresholding.

Berikut ini adalah hasil pengujian dengan menggunakan beberapa sampel dengan karakteristik yang berbeda:

Gambar Input	Gambar Output	
	Aplikasi Thresholding	GIMP
		







Gambar 5.2 Pengujian