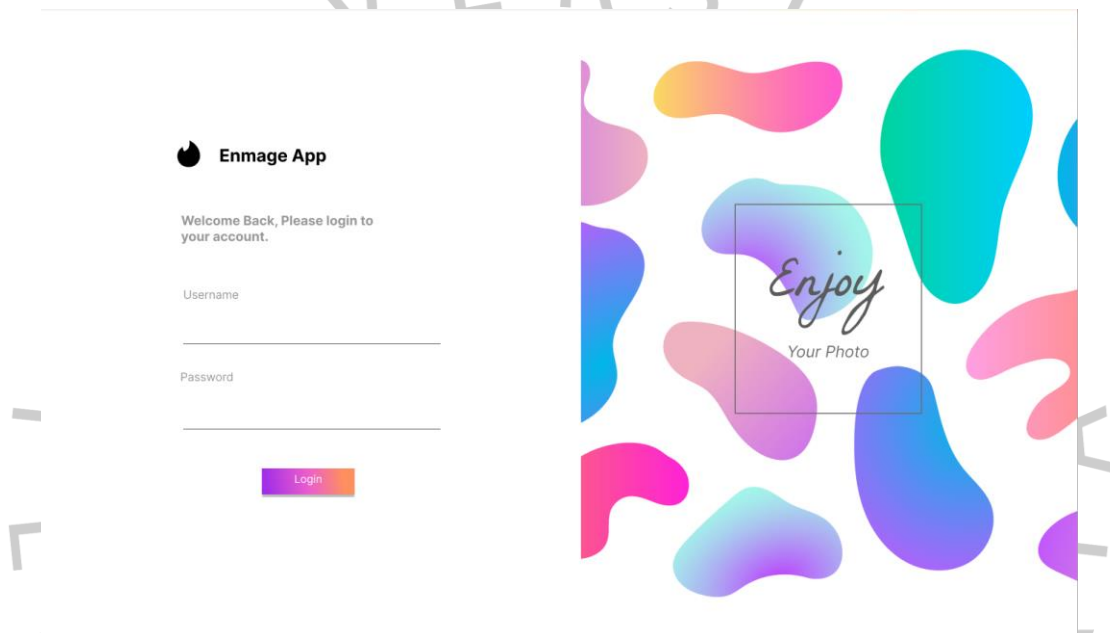


BAB V

HASIL DAN PEMBAHASAN

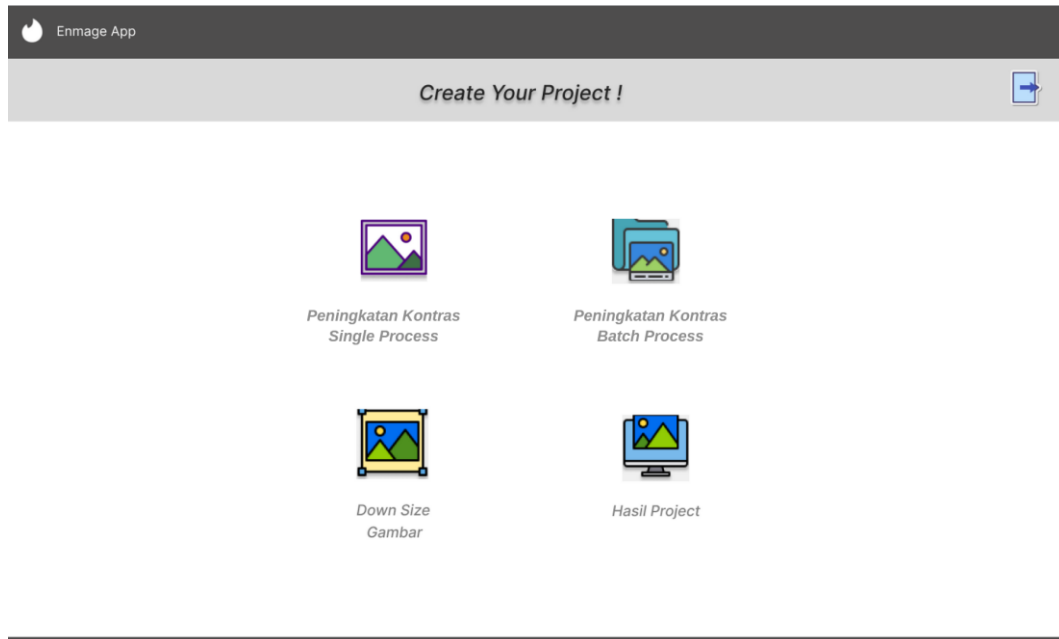
5.1 Hasil Perancangan Aplikasi

Aplikasi Meningkatkan Kontras Gambar dibuat untuk memudahkan pengguna untuk mampu meningkatkan kontras gambar agar lebih baik dari setiap piksel gambar. Berikut hasil lebih rinci aplikasi penelitian ini.



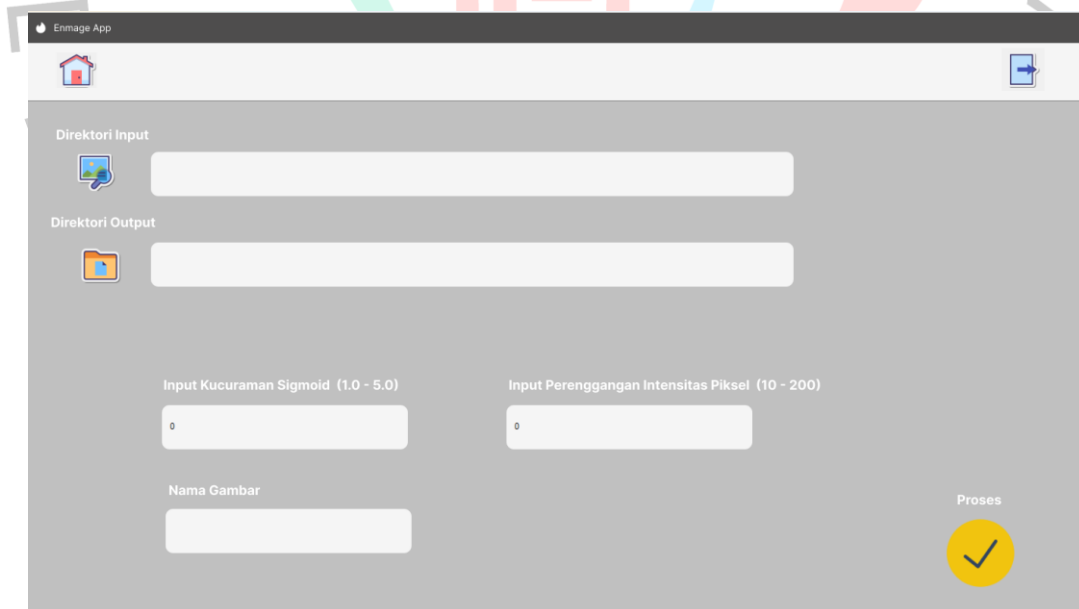
Gambar 5.1 Halaman *Login* Aplikasi

Terlihat pada Gambar 5.1 halaman *Login* yakni halaman pertama ketika *user* membuka aplikasi. *User* akan melakukan *Login* dengan menulis *username* dan *password* yang sudah ditetapkan, kemudian nantinya akan diverifikasi oleh sistem.



Gambar 5.2 Halaman *Dashboard* Aplikasi

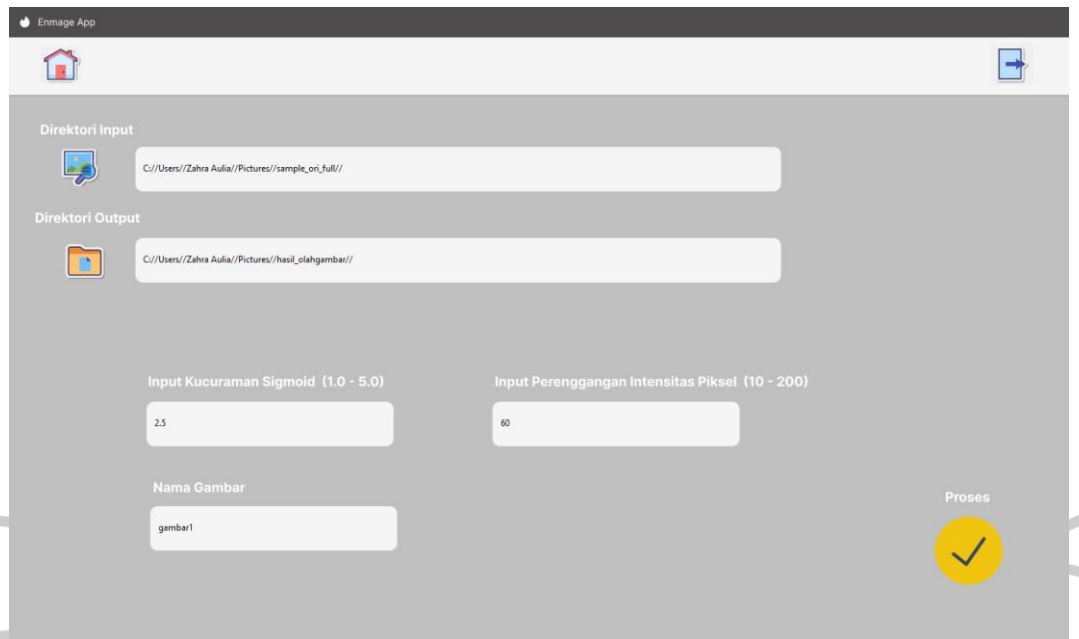
Pada Gambar 5.2 halaman *Dashboard*. Pengguna akan dihadapkan oleh empat menu yang bisa digunakan, yakni peningkatan kontras *Single Process*, peningkatan kontras *Batch Process*, *Down Size* dan hasil *Project*.



Gambar 5.3 Halaman Menu Peningkatan Kontras *Single Process*

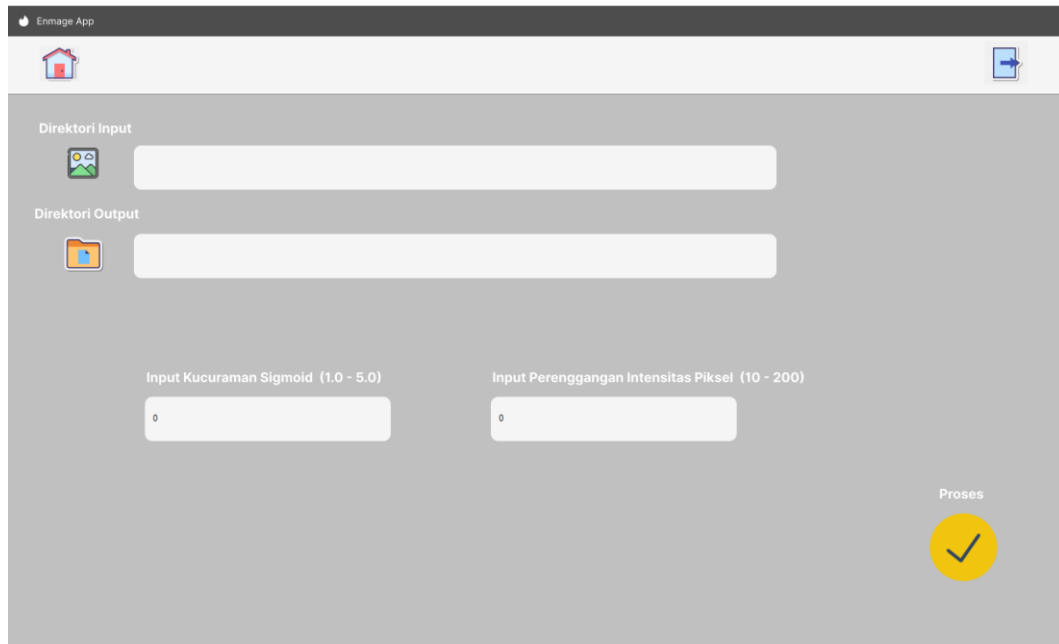
Pada Gambar 5.3 halaman menu Peningkatan Kontras *Single Process* untuk meningkatkan kontras gambar. *User* dapat *input* alamat direktori file gambar dengan mengisi *text box* yang disediakan dan *input* lokasi penyimpanan hasil

gambar diproses sistem pada *text box*. Pengguna juga harus *input* nama gambar pada kolom *text box* Nama Gambar. *User* juga memasukan nilai S pada algoritma sigmoid nantinya dan klik Proses untuk memproses gambar. *Range* pengguna memasukkan nilai S untuk persamaan *Modified Sigmoid Transfer Function* adalah minimal 0,1.



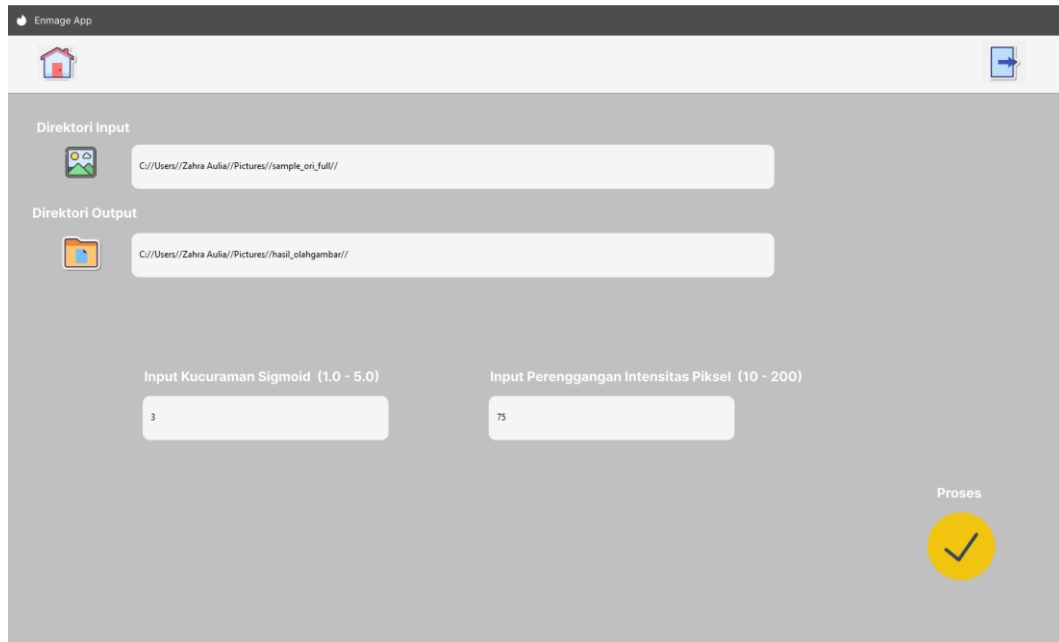
Gambar 5.4 Halaman Menu Peningkatan Kontras *Single Process* Terisi

Pada Gambar 5.4 terlihat alamat direktori untuk gambar yang dimasukan akan tertulis pada kolom *text box*. *User* juga dapat memasukkan nilai sigmoid sesuai kebutuhan.



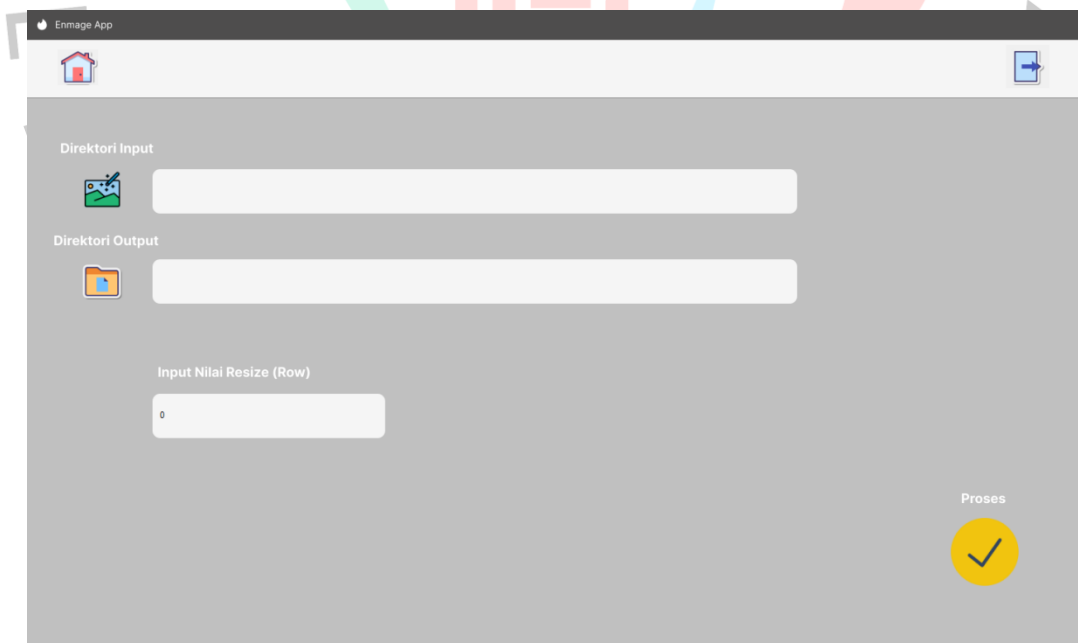
Gambar 5.5 Halaman Menu Peningkatan Kontras *Batch Process*

Pada Gambar 5.5 halaman menu Peningkatan Kontras *Batch Process* untuk meningkatkan kontras gambar secara *batch processing*. User akan dapat *input* gambar lokasi direktori folder gambar yang akan diproses sekaligus pada *text box* direktori *input*. Kemudian *input* lokasi penyimpanan hasil proses pada *text box* direktori *output*. User juga memasukkan nilai *Kk* pada algoritma sigmoid nantinya dan klik proses untuk memproses gambar.



Gambar 5.6 Halaman Menu Peningkatan Kontras *Batch Process* Terisi

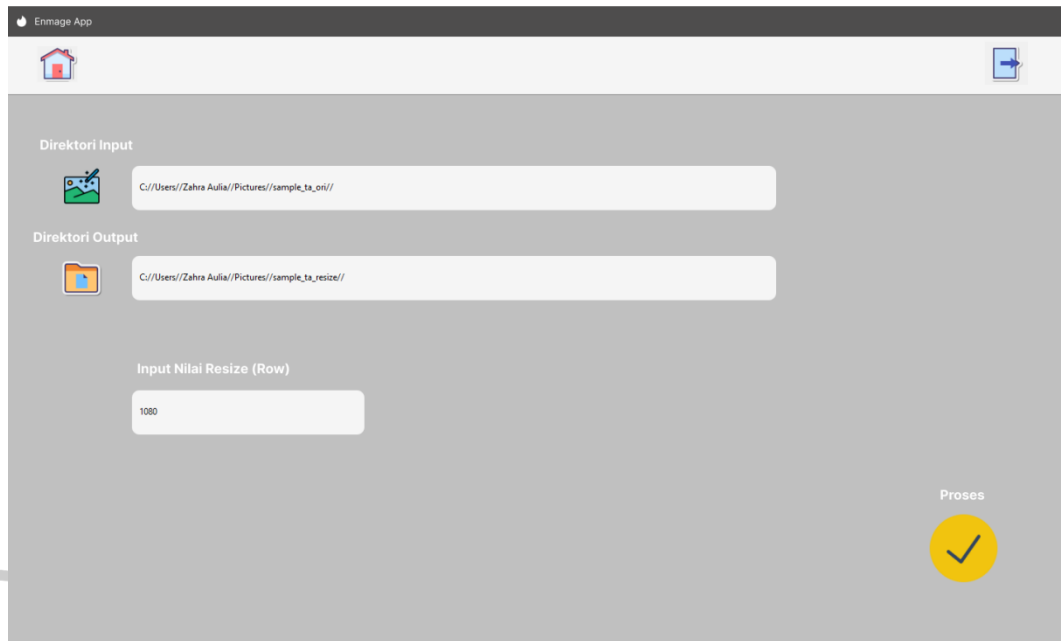
Pada Gambar 5.6 terlihat alamat direktori untuk gambar yang dimasukan akan tertulis pada kolom *text box*. *User* dapat memasukkan nilai sigmoid sesuai kebutuhan.



Gambar 5.7 Halaman Menu *Down Size* Gambar

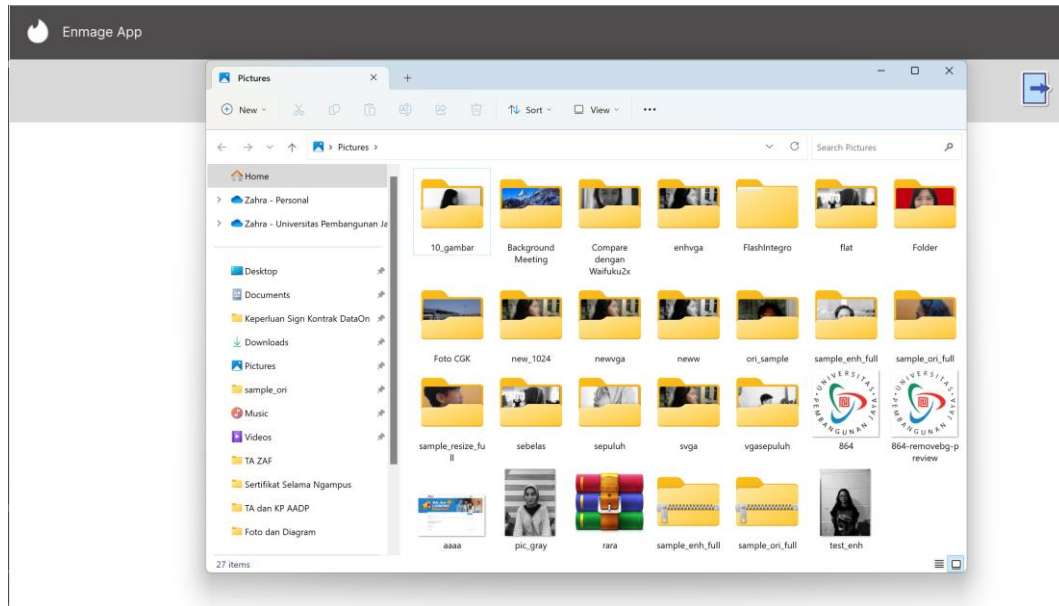
Pada Gambar 5.7 halaman menu *Down Size* Gambar untuk mengecilkan ukuran gambar dalam proses peningkatan kontras jika piksel gambar melebihi *full HD* 1920

x 1080 piksel. *User* dapat *input* alamat direktori folder gambar pada *text box* direktori *input* dan *input* lokasi penyimpanan hasil direktori yang akan diproses sistem pada *text box* direktori *output*. Proses *Down Size* gambar dilakukan secara *Batch Processing*.



Gambar 5.8 Halaman Menu *Down Size* Gambar Terisi

Pada Gambar 5.8 terlihat alamat direktori untuk *Path* gambar yang dimasukan akan tertulis pada kolom *text box*. *User* bisa langsung proses dan file hasil proses akan tersimpan di folder direktori komputer tujuan.



Gambar 5.9 Halaman Menu Hasil *Project*

Pada Gambar 5.9 halaman menu Hasil *Project* untuk melihat hasil olah gambar baik peningkatan kontras ataupun *Down Size* yang sudah dilakukan.

Pada penelitian ini aplikasi berjalan sesuai dengan kecepatan komputer pengguna. Peneliti menggunakan GUI dengan bahasa pemrograman Python sebagai pembuktian konsep *image processing* dan algoritma yang digunakan.



5.2 Pembahasan

5.2.1 Hasil *Down Size* Gambar

Sebelum pengguna melakukan peningkatan kontras pada gambar yang memiliki ukuran lebih dari *full HD* yakni 1080 x 1920 piksel, maka harus *resize* gambar dengan fitur yang tersedia pada aplikasi. Pengguna dapat memperkecil ukuran gambar dengan ukuran maksimal *row* gambar 1080 piksel. Berikut ini tabel hasil *Down Size* gambar dengan beberapa sampel.

Tabel 5.1 Hasil *Down Size* Gambar

No	Gambar Original	Gambar Hasil <i>Resize</i>
1	 <p data-bbox="384 1055 794 1189">Keterangan : Gambar original Nama file : Sample12.jpg Ukuran file :1,69 MB Ukuran Resolusi : 1836 x 3264 piksel</p>	 <p data-bbox="884 969 1281 1104">Keterangan : Gambar hasil <i>resize</i> Nama file : Sample12_rz.jpg Ukuran file :52,8 KB Ukuran Resolusi : 608 x 1080 piksel</p>
2	 <p data-bbox="384 1861 794 1995">Keterangan : Gambar original Nama file : Sample56.jpg Ukuran file : 3,02 MB Ukuran Resolusi : 3120 x 4160 piksel</p>	 <p data-bbox="884 1776 1281 1910">Keterangan : Gambar hasil <i>resize</i> Nama file : Sample12_rz.jpg Ukuran file :52,8 KB Ukuran Resolusi : 810 x 1080 piksel</p>

	
<p>Keterangan : Gambar original Nama file : Sample107.jpg Ukuran file : 70,7 KB Ukuran Resolusi : 1200 x 1600 piksel</p>	<p>Keterangan : Gambar hasil <i>resize</i> Nama file : Sample107_rz.jpg Ukuran file : 37,7 KB Ukuran Resolusi : 810 x 1080 piksel</p>

Tabel 5.1 merupakan hasil *resize* gambar dengan 3 sampel. Dalam proses *Down Size* gambar, jumlah *coloumn* (col) piksel akan mengikuti hitungan program yang mengacu pada besaran *row* piksel *input* pengguna. Program akan mengecek *input row* untuk memproses *Down Size* jika *row* > 1080 piksel, sedangkan program tidak memproses *Down Size* gambar dan mencetak informasi “*Sorry, File Smaller Than Resolution 480 Pixels!*” jika *row* <= 1080 piksel. Berikut contoh hitung piksel gambar *Down Size*.

Keterangan gambar:

row gambar = 4160 piksel dan *col* gambar = 3120 piksel.

Row input (kebutuhan pengguna) = 1080 piksel.

Maka proses selanjutnya:

resize_faktor = $4160 / 1080 = 3,8518$

coloumn resize = $3120 / 3,8518 = 810$ piksel.

Selanjutnya program menyiapkan kanvas perhitungan *Down Size* dengan ukuran 810 x 1080 piksel dan menyalin piksel gambar original ke kanvas hasil *Down Size*.

5.2.2 Hasil Gambar Peningkatan Kontras

Pada penelitian ini peneliti mengubah piksel RGB pada gambar menjadi *gray* atau hitam putih, guna untuk melihat lebih jelas perbedaan kontras yang dialami pada gambar. Dalam proses visi komputer gambar dengan keadaan hitam putih juga diperlukan sebagai pra oleh visi komputer. Berikut adalah salah satu hasil gambar yang telah diproses pada sistem algoritma *Modified Sigmoid Transfer Function* dengan nilai $kk = 2,1$ dan $v = 60$.



Gambar 5.10 Sampel Pertama Original



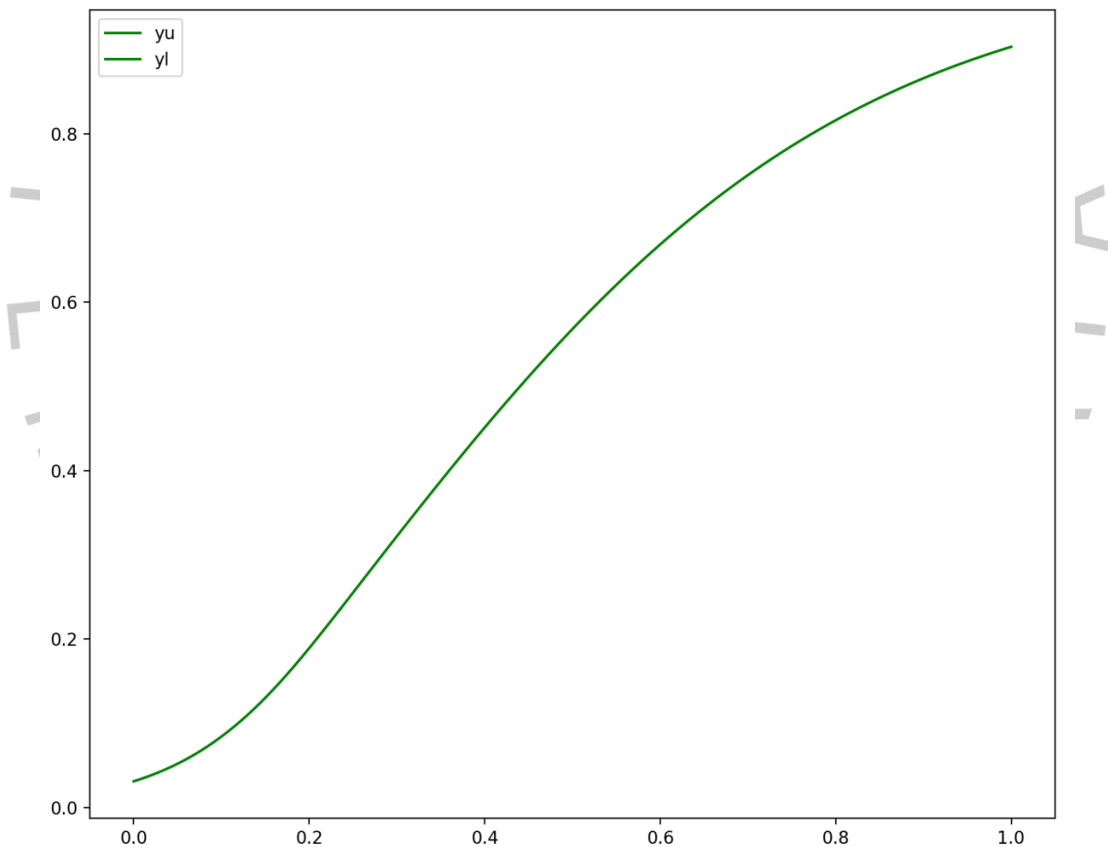
Gambar 5.11 Sampel Pertama Sebelum Peningkatan Kontras



Gambar 5.12 Sampel Pertama Sesudah Peningkatan Kontras

Pada Gambar 5.10 merupakan foto awal yang belum diproses. Kemudian pada Gambar 5.11 merupakan hasil perubahan RGB menjadi gambar hitam putih dengan mengubah piksel RGB 3 dimensi ke piksel 2 dimensi hitam putih dengan cara dimensi RGB pada gambar kali dengan nilai ilmiah untuk *Gray Scale*.

Pada Gambar 5.12 merupakan hasil final gambar setelah diproses dengan menggunakan algoritma *Modified Sigmoid Transfer Function*. Terlihat pada titik-titik tertentu pada kontras gambar menjadi lebih tajam dan terang. Dengan *input* $V = 60$ maka perhitungan perenggangan intensitas piksel menjadi sebagai berikut. Bisa dilihat bahwa hasil peningkatan kontras terjadi dengan baik sehingga membuat gambar menjadi lebih tajam dengan perenggangan intensitas piksel $= 60 (v) / 255 = 0,235$.



Gambar 5.13 Hasil Grafik Sigmoid Termodifikasi Sampel Gambar Pertama

Nilai 0,235 akan menjadi batas penentu piksel akan lebih digelapkan atau diterangkan sesuai dengan intensitasnya. Jika piksel gambar memiliki intensitas $> 0,235$ maka akan semakin meningkat atau warna pada piksel gambar akan semakin terang. Namun, jika piksel gambar memiliki intensitas $\leq 0,235$ akan menurun atau

warna piksel semakin gelap. Proses akan berlangsung sampai dengan piksel terakhir pada gambar.

Kondisi intensitas < lebih dari *threshold* (V)

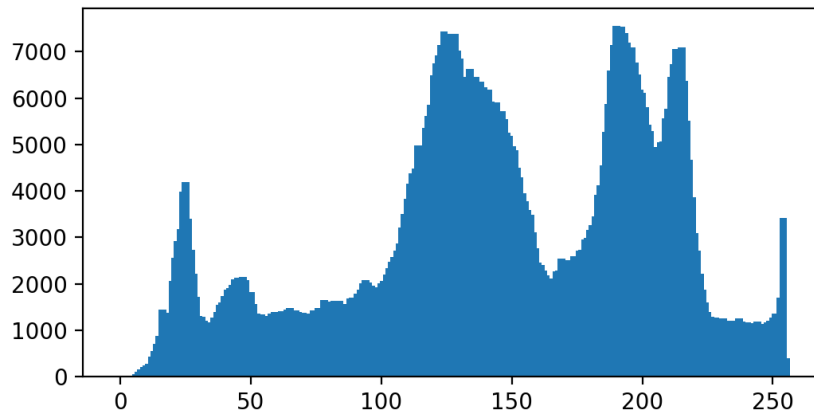
$$K = kk \times 9^{(1-V)} \quad (6)$$

$$S = \frac{v}{0,5} \quad (7)$$

Contoh perhitungan :

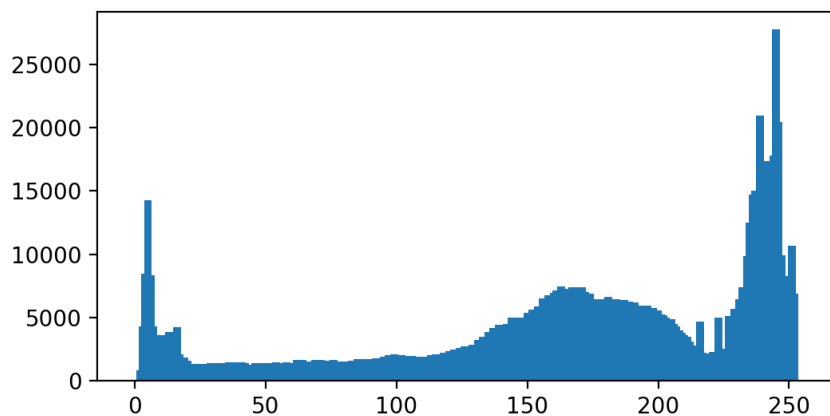
$$K = 2,1 \times 9^{0,235} = 3,519380994826$$

$$S = \frac{0,235}{0,5} = 0,47$$



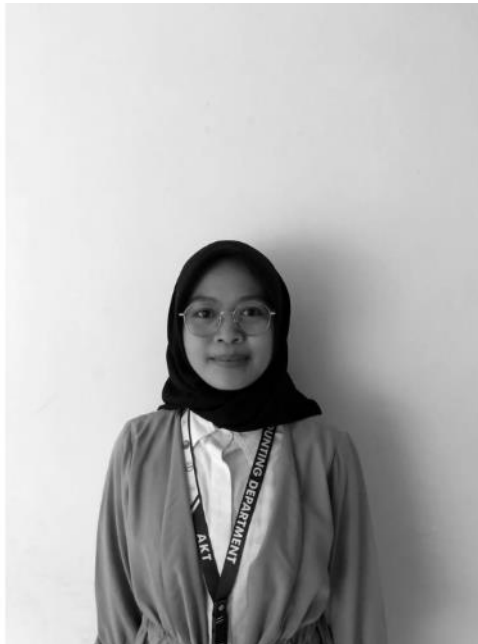
Gambar 5.14 Diagram Histogram Sampel Pertama Kondisi Original

Pada Gambar 5.14 merupakan histogram untuk frekuensi piksel saat gambar masih dalam RGB. Terlihat pada *value* warna 123-125 dan 188-192 memiliki frekuensi lebih dari lebih dari 7.000 piksel pada gambar.

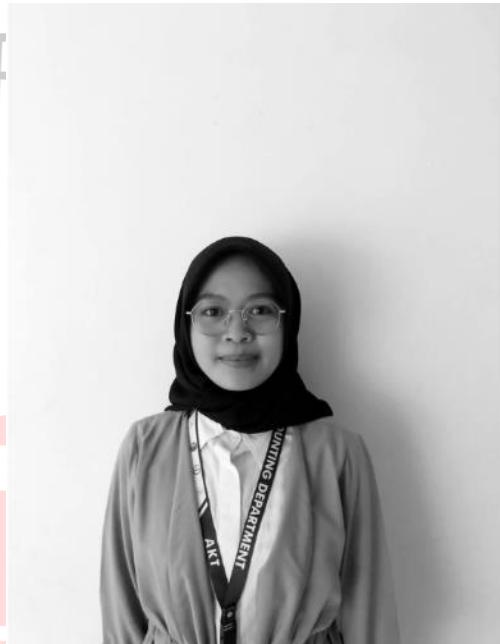


Gambar 5.15 Diagram Histogram Sampel Pertama Hasil Peningkatan Kontras

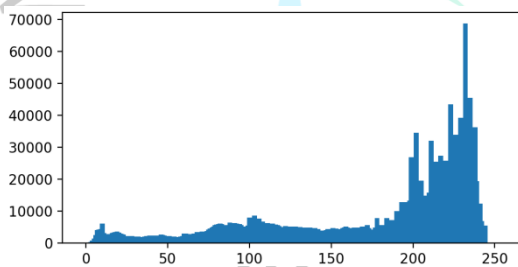
Pada Gambar 5.15 merupakan histrogram untuk frekuensi piksel saat telah diubah menjadi hitam putih dan diproses dengan *Modified Sigmoid Transfer Function*. Terjadi perubahan frekuensi pada *value* warna gambar secara keseluruhan, namun terlihat meningkat cukup signifikan frekuensi *value* warna pada 243-245 yakni mencapai lebih dari 25.000.



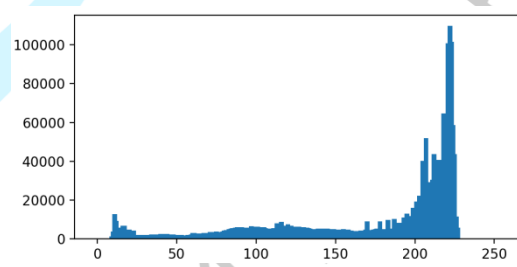
Gambar 5.16 Sampel Kedua Original



Gambar 5. 17 Sampel Kedua Hasil Peningkatan Kontras Sampel



Gambar 5.18 Diagram Histogram Sampel Kedua Original



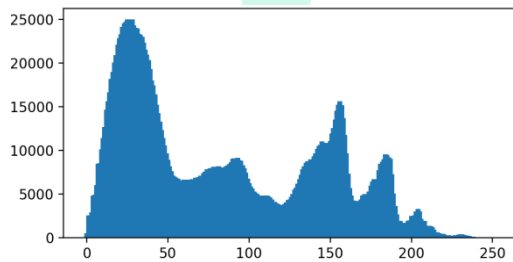
Gambar 5.19 Diagram Histogram Sampel Kedua Hasil Peningkatan Kontras



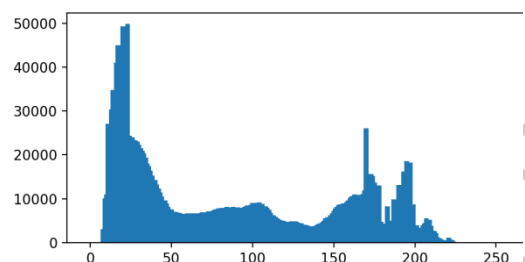
Gambar 5.20 Sampel Ketiga Original



Gambar 5.21 Sampel Ketiga Hasil Peningkatan Kontras



Gambar 5.22 Diagram Histogram Sampel Ketiga Original



Gambar 5.23 Diagram Histogram Sampel Ketiga Hasil Peningkatan Kontras



Gambar 5.24 Hasil Olah Gambar *Batch Process* Peningkatan Kontras 1-55



Gambar 5.25 Hasil Olah Gambar *Batch Process* Peningkatan Kontras 56-110



Gambar 5.26 Hasil Olah Gambar *Batch Process* Peningkatan Kontras 111-154



Gambar 5.27 Hasil Olah Gambar *Batch Process* Peningkatan Kontras 155-200



Gambar 5.28 Sampel Ketiga Original



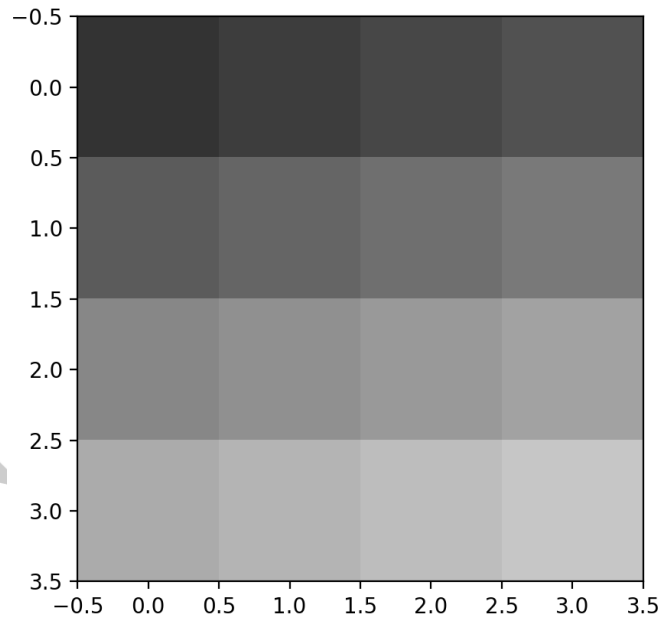
Gambar 5.29 Sampel Ketiga Ditingkatkan Kontras dengan Waifu2x



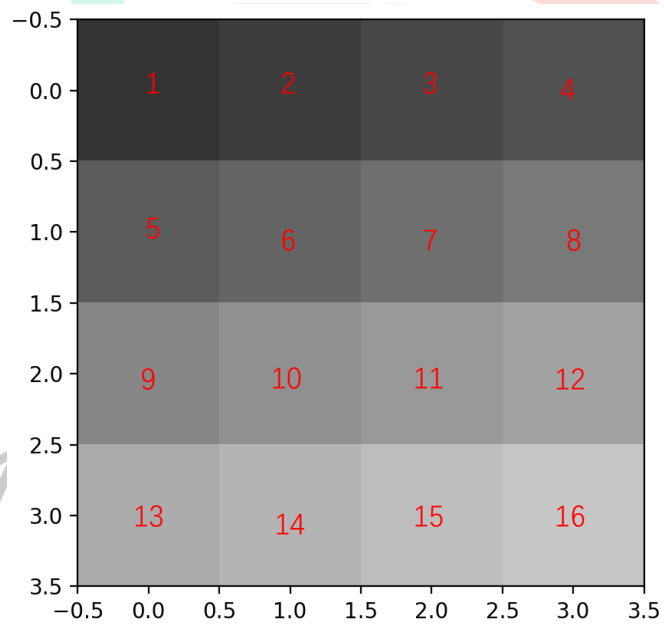
Gambar 5.30 Sampel Ketiga Ditingkatkan Kontras dengan Aplikasi Peneliti

Pada Gambar 5.28 merupakan gambar sampel original yang sudah menjadi *Gray Scale*. Peneliti menguji hasil gambar dengan menyandingkan tiga gambar sekaligus yakni original, hasil dari Waifu2x dan hasil aplikasi yang sudah dibuat oleh peneliti. Terlihat pada Gambar 5.29 bahwa hasil dari aplikasi Waifu2x lebih terang secara keseluruhan. Cahaya gambar yang meningkat secara merata membuat hasil objek gambar terlihat ketajaman gambar berkurang. Disisi lain terlihat Gambar 5.30 terlihat lebih kontras dan pencahayaan lebih baik.

5.2.3 Testing 16 Piksel



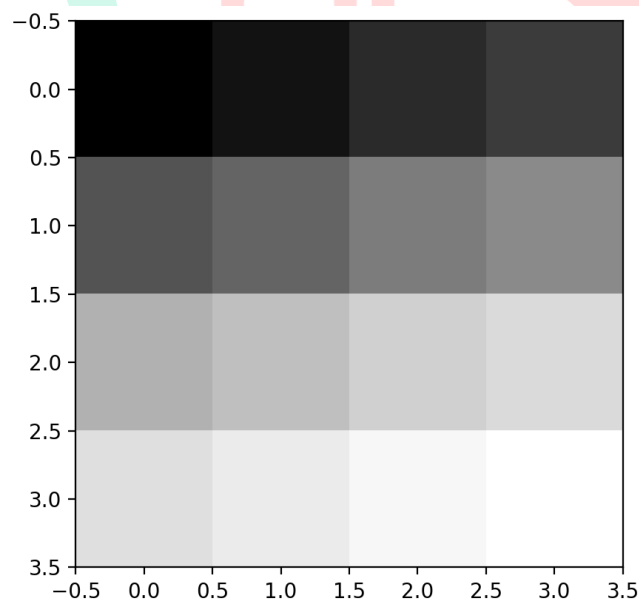
Gambar 5.31 Sampel 16 Piksel Original



Gambar 5.32 Penomoran Sampel 16 Piksel Original

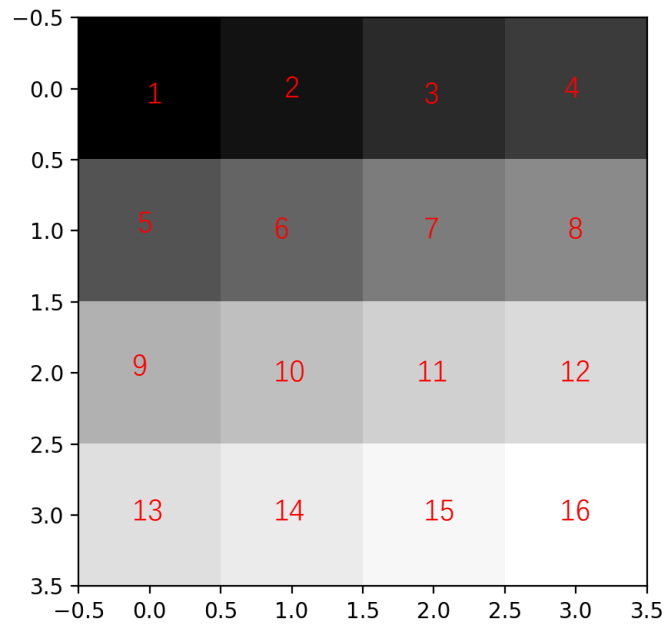
Tabel 5.2 Keterangan Nilai RGB Sampel 16 Piksel

Keterangan Piksel	
Piksel 1	RGB (51, 51, 51)
Piksel 2	RGB (61, 61, 61)
Piksel 3	RGB (71, 71, 71)
Piksel 4	RGB (81, 81, 81)
Piksel 5	RGB (91, 91, 91)
Piksel 6	RGB (101, 101, 101)
Piksel 7	RGB (111, 111, 111)
Piksel 8	RGB (121, 121, 121)
Piksel 9	RGB (135, 135, 135)
Piksel 10	RGB (144, 144, 144)
Piksel 11	RGB (153, 153, 153)
Piksel 12	RGB (162, 162, 162)
Piksel 13	RGB (171, 171, 171)
Piksel 14	RGB (180, 180, 180)
Piksel 15	RGB (189, 189, 189)
Piksel 16	RGB (198, 198, 198)



Gambar 5.33 Sampel 16 Piksel Hasil Peningkatan Kontras

Gambar 5.33 merupakan hasil piksel dari peningkatan kontras menggunakan nilai K_k sigmoid *input* 2,1 dan nilai V *input* 127.

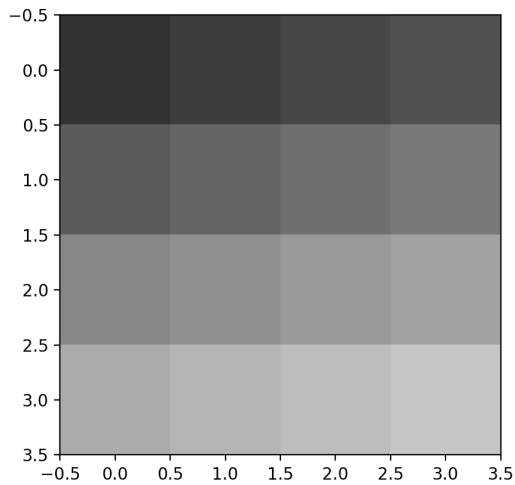


Gambar 5.34 Penomoran Sampel 16 Pixel Hasil Peningkatan Kontras

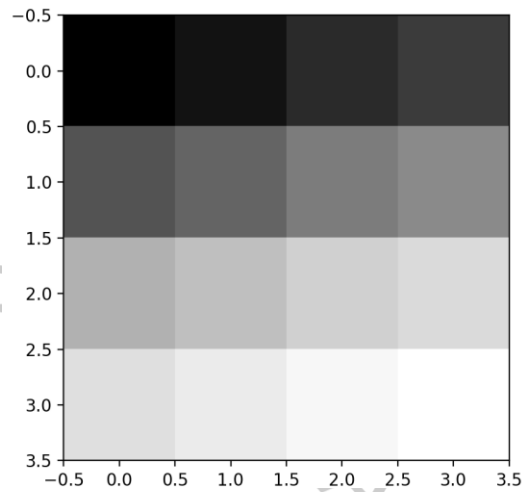
Tabel 5.3 Keterangan Hasil *Gray Scale* Sampel 16 Pixel

Keterangan Pixel	
Piksel 1	<i>Gray Scale</i> (36, 0)
Piksel 2	<i>Gray Scale</i> (42, 0)
Piksel 3	<i>Gray Scale</i> (53, 0)
Piksel 4	<i>Gray Scale</i> (60, 0)
Piksel 5	<i>Gray Scale</i> (74, 0)
Piksel 6	<i>Gray Scale</i> (85, 0)
Piksel 7	<i>Gray Scale</i> (101, 0)
Piksel 8	<i>Gray Scale</i> (111, 0)
Piksel 9	<i>Gray Scale</i> (144, 0)
Piksel 10	<i>Gray Scale</i> (156, 0)
Piksel 11	<i>Gray Scale</i> (172, 0)
Piksel 12	<i>Gray Scale</i> (182, 0)
Piksel 13	<i>Gray Scale</i> (185, 0)
Piksel 14	<i>Gray Scale</i> (196, 0)
Piksel 15	<i>Gray Scale</i> (207, 0)
Piksel 16	<i>Gray Scale</i> (214, 0)

Hasil sampel 16 piksel jika disandingkan bersama akan terlihat sebagai berikut.

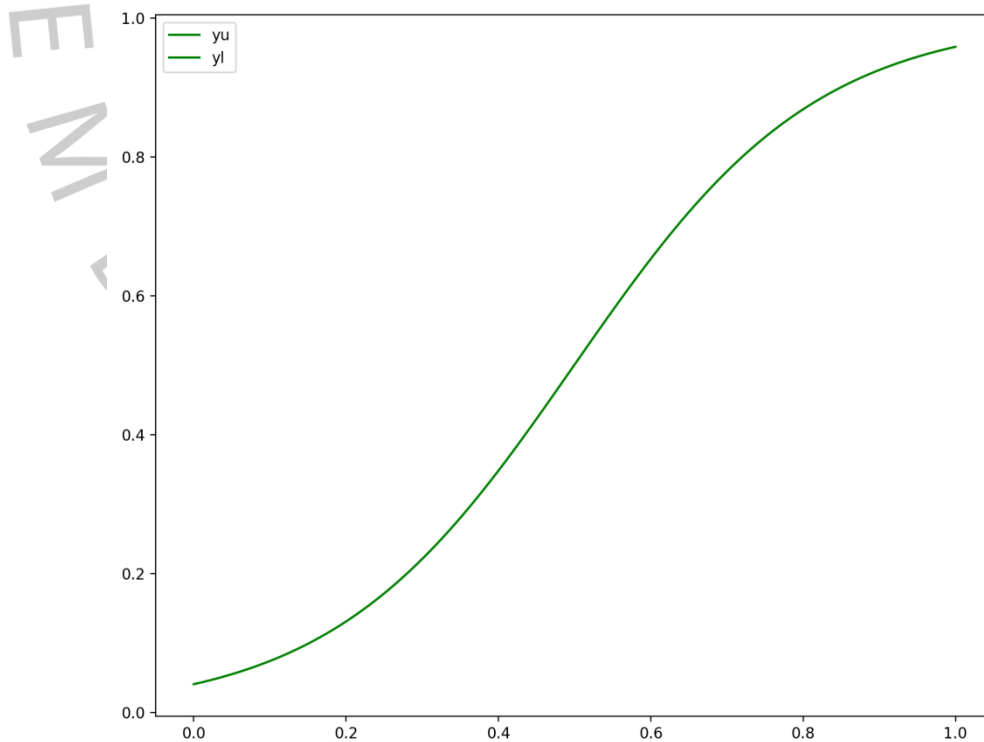


Gambar 5.35 Sampel 16 Piksel Original



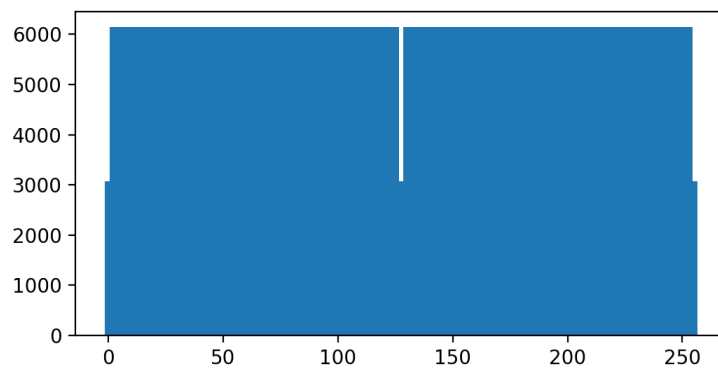
Gambar 5.36 Sampel 16 Piksel Hasil Peningkatan Kontras

Dapat dilihat pada Gambar 5.35 merupakan hasil uji coba pada 16 piksel dengan aplikasi peningkatan kontras. Dengan nilai *input* K_k sebesar 2,1 dan V 127 dapat menghasilkan peningkatan kontras yang signifikan. Kontras menjadi lebih baik dan tajam.

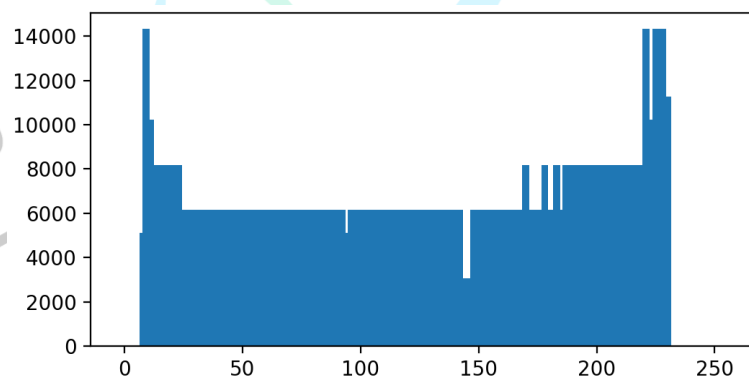


Gambar 5.37 Kurva Sigmoid Termodifikasi Sampel 16 Piksel

Gambar 5.37 merupakan grafik hasil sampel 16 piksel dengan menggunakan *Modified Sigmoid Transfer Function*. Terlihat bahwa jika intensitas piksel ≤ 0.5 maka nilai piksel akan semakin gelap dari pada intensitas piksel original. Contohnya terlihat intensitas piksel pada garis Y sebesar 0.4 yang seharusnya memiliki *output* 0.4, namun menjadi 0.3504 (semakin gelap). Sedangkan Gambar 5.28 akan meningkatkan intensitas piksel menjadi semakin terang jika perenggangan intensitas > 0.5 . Contohnya terlihat intensitas piksel pada garis Y sebesar 0.6 yang seharusnya memiliki *output* 0.6, namun menjadi 0.6540 (semakin terang). Seluruh perubahan kontras pada piksel terlihat lebih jelas pada Tabel 5.2 dan Tabel 5.3.

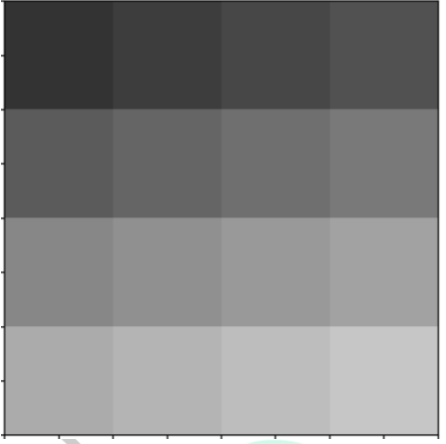
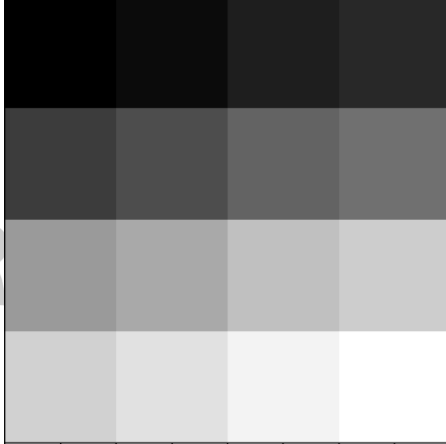
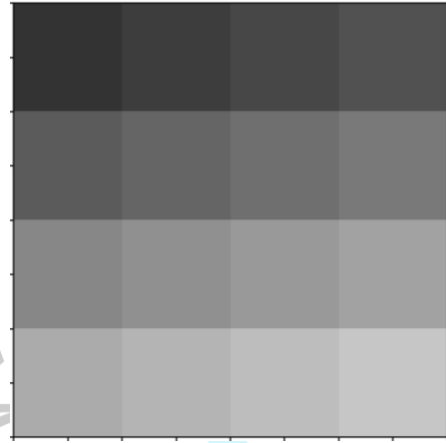
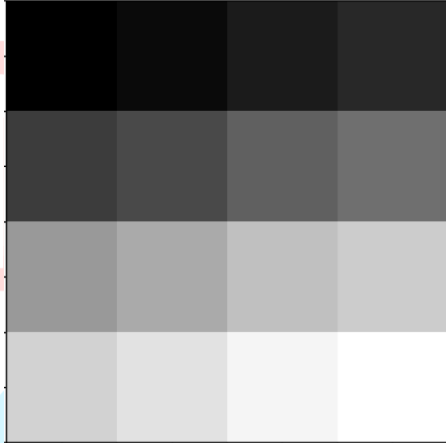
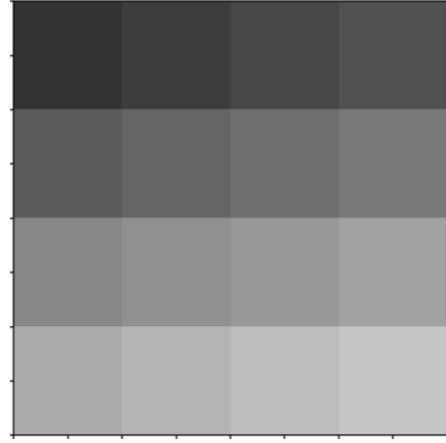
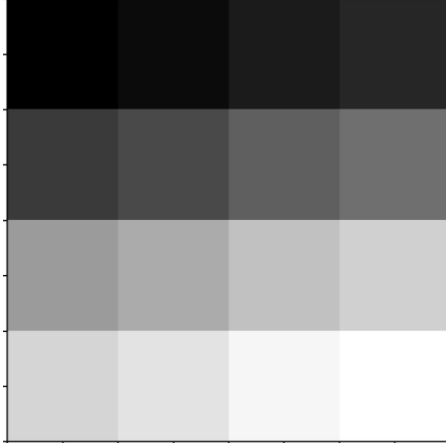


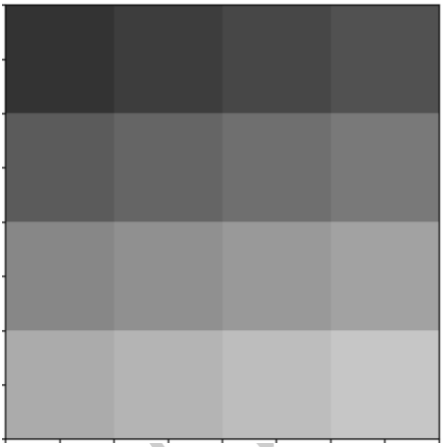
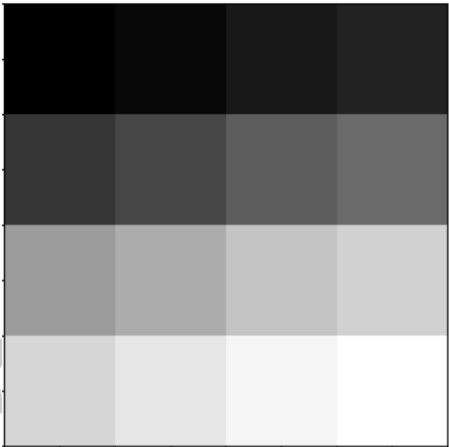
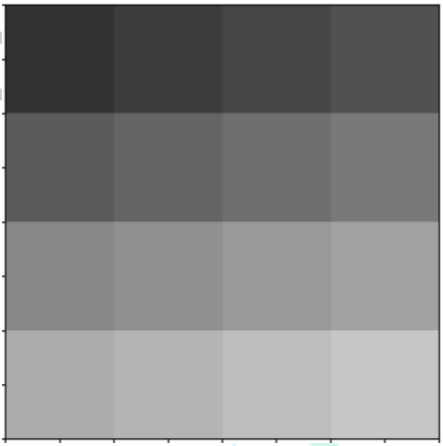
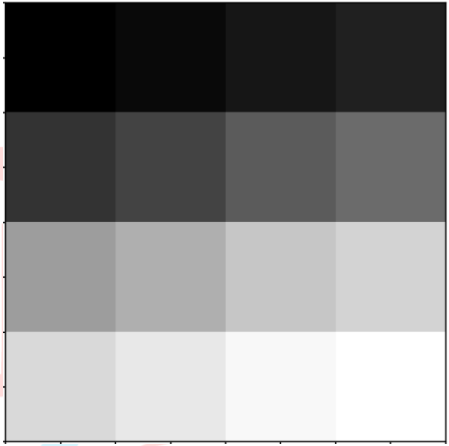
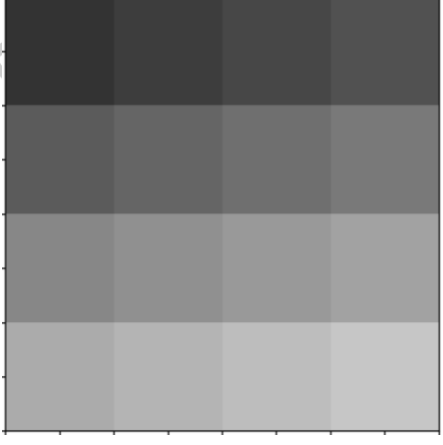
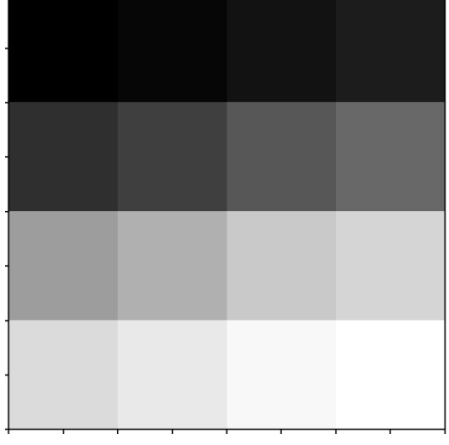
Gambar 5.38 Diagram Histogram Sampel 16 Piksel Origina

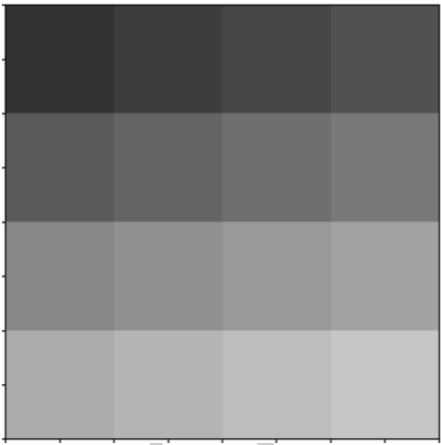
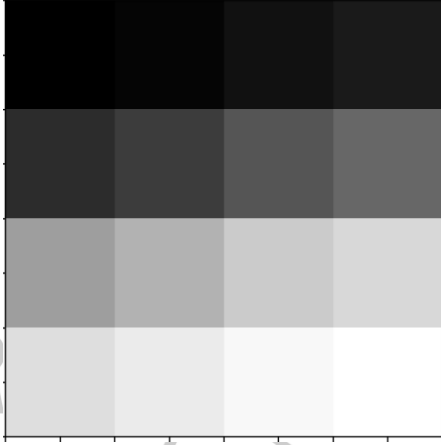
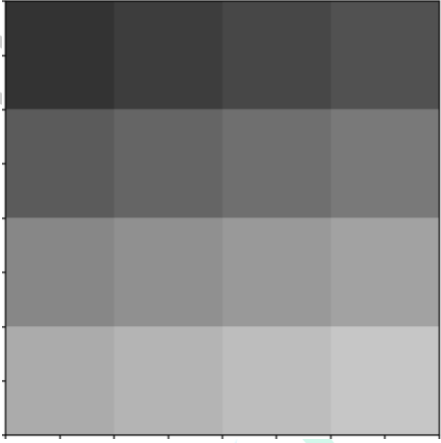

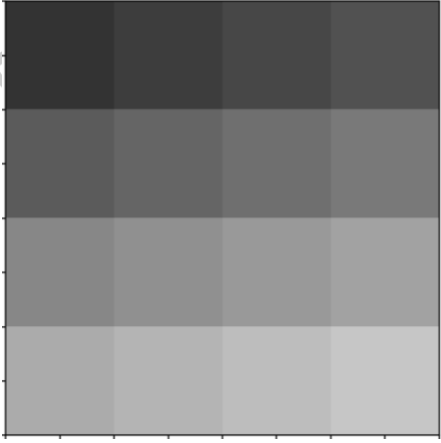
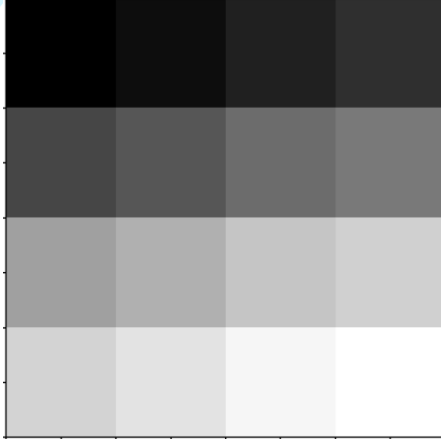


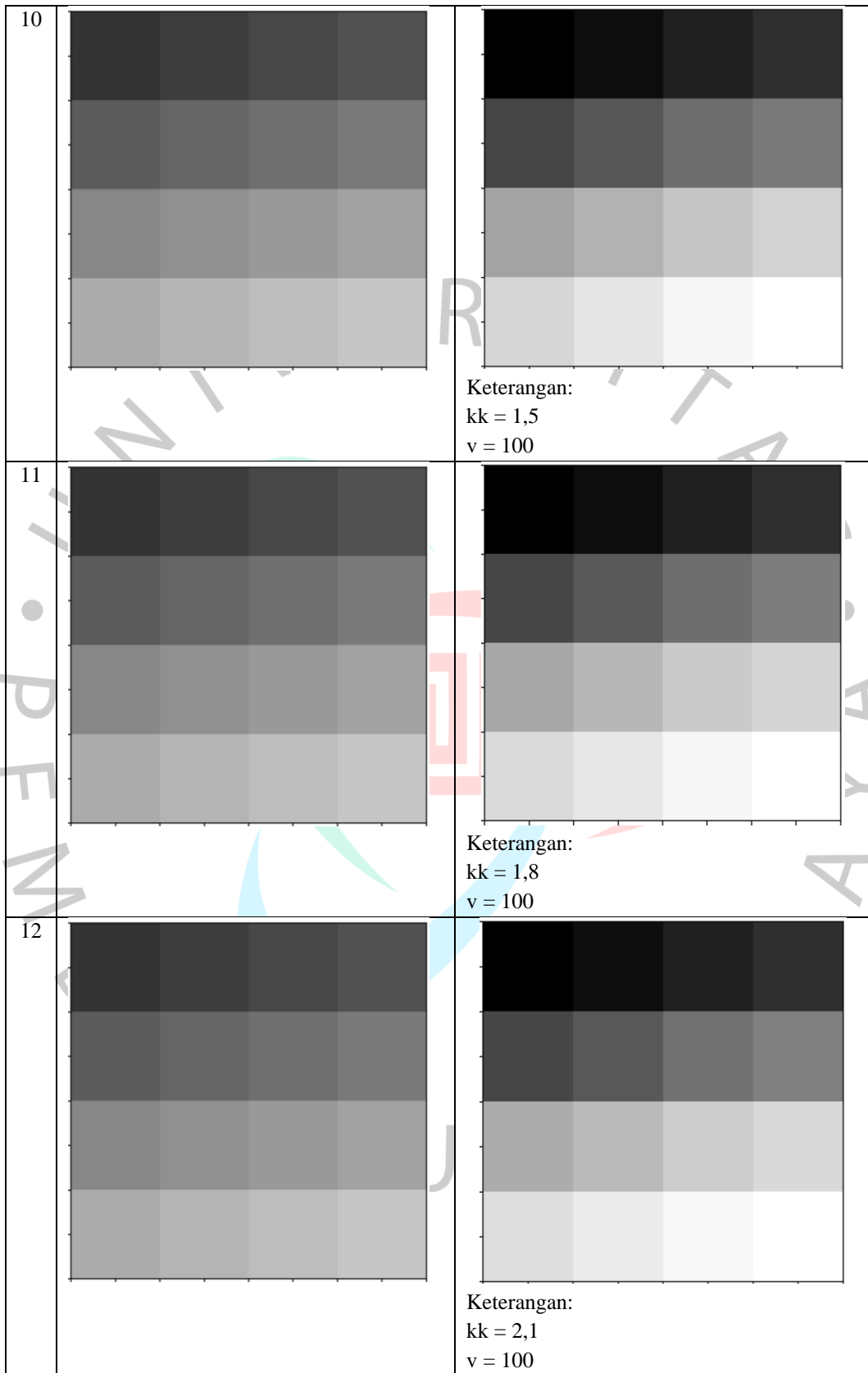
Gambar 5.39 Diagram Histogram Sampel 16 Piksel Peningkatan Kontras

Tabel 5.4 Hasil *Testing* 16 Piksel dengan Berbagai Nilai *Input* Kk dan V

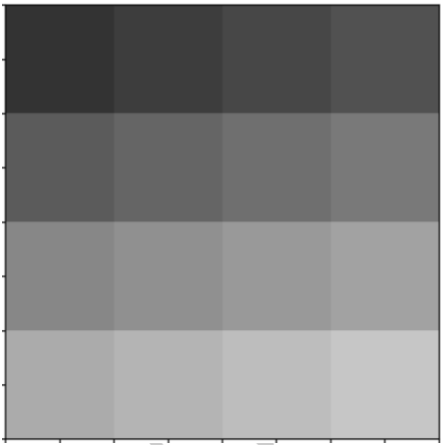
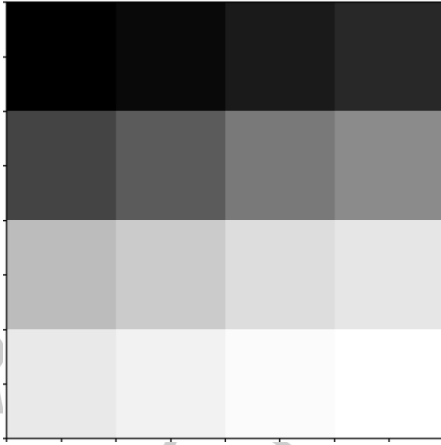
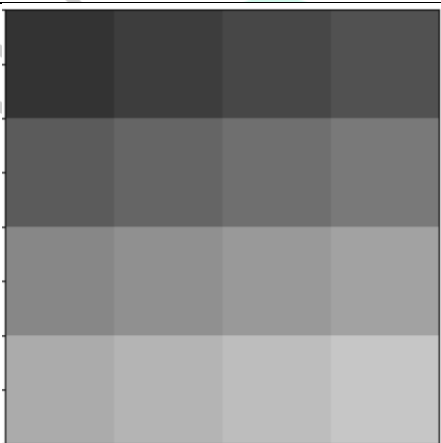
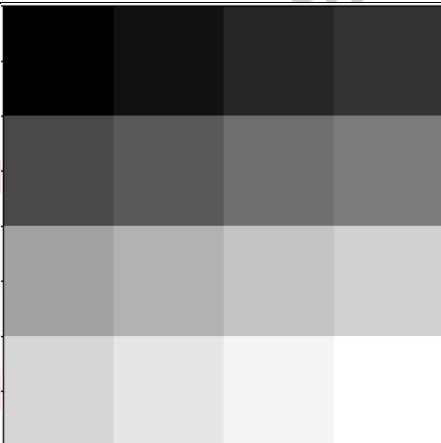
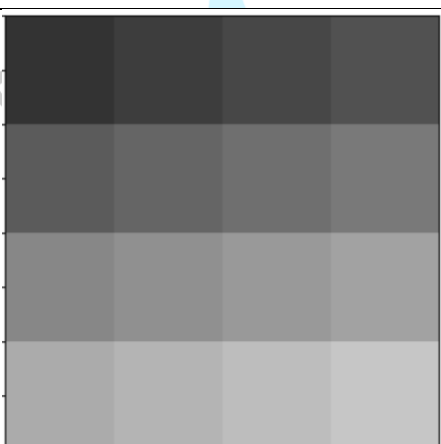
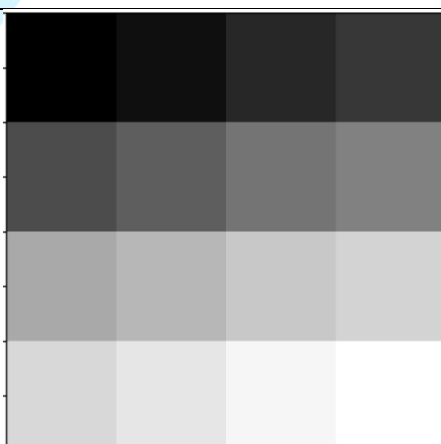
No	Original Gambar	Gambar Hasil Peningkatan Kontras
1		 <p data-bbox="866 790 1002 891">Keterangan: kk = 1,2 v = 127</p>
2		 <p data-bbox="866 1350 1002 1451">Keterangan: kk = 1,5 v = 127</p>
3		 <p data-bbox="866 1910 1002 2011">Keterangan: kk = 1,8 v = 127</p>

4			<p>Keterangan: $kk = 2,1$ $v = 127$</p>
5			<p>Keterangan: $kk = 2,4$ $v = 127$</p>
6			<p>Keterangan: $kk = 2,7$ $v = 127$</p>

7		 <p data-bbox="863 678 1002 779">Keterangan: kk = 3,0 v = 127</p>
8		 <p data-bbox="863 1236 1002 1337">Keterangan: kk = 3,3 v = 127</p>
9		 <p data-bbox="863 1794 1002 1890">Keterangan: kk = 1,2 v = 100</p>



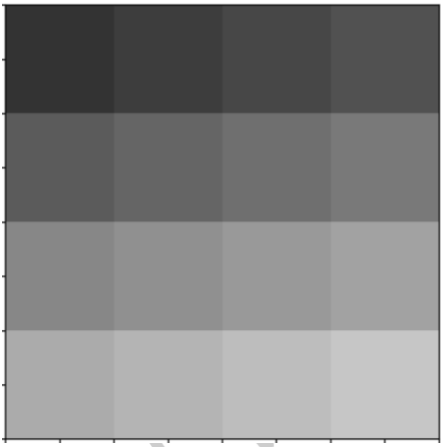
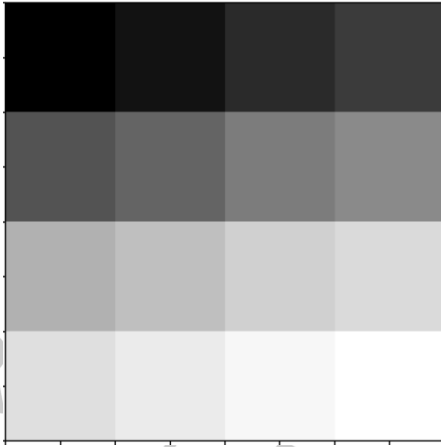
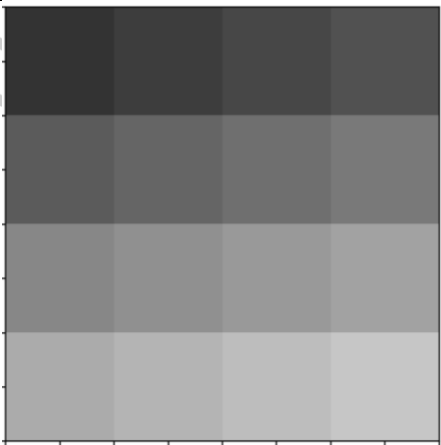
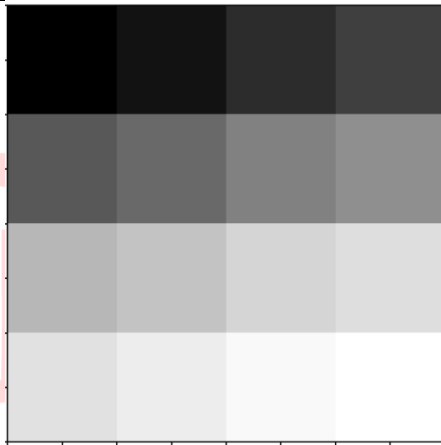
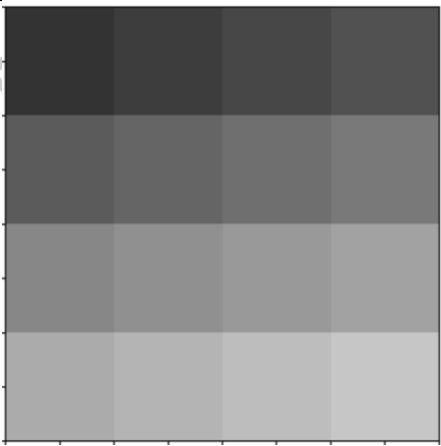
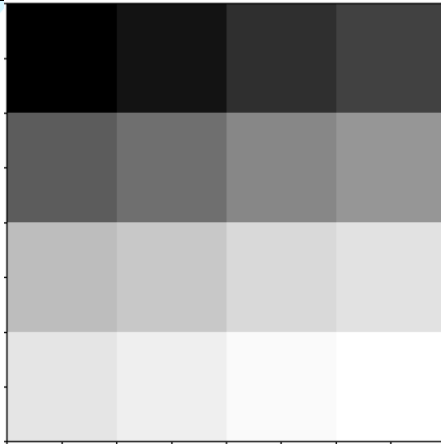
13			<p>Keterangan: $kk = 2,4$ $v = 100$</p>
14			<p>Keterangan: $kk = 2,7$ $v = 100$</p>
15			<p>Keterangan: $kk = 3,0$ $v = 100$</p>

16		
17		
18		

Keterangan:
 $kk = 3,3$
 $v = 100$

Keterangan:
 $kk = 1,2$
 $v = 60$

Keterangan:
 $kk = 1,7$
 $v = 60$

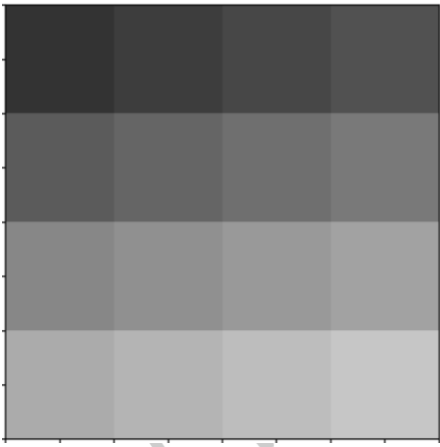
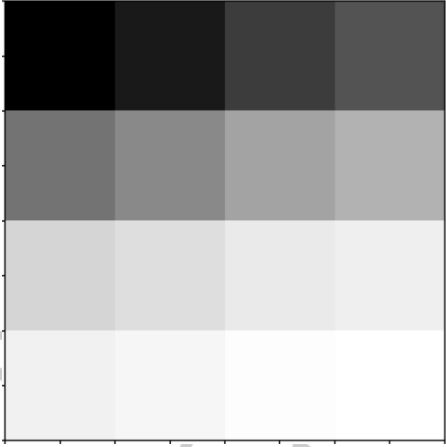
19		
20		
21		

Keterangan:
 $kk = 2,1$
 $v = 60$

Keterangan:
 $kk = 2,4$
 $v = 60$

Keterangan:
 $kk = 2,7$
 $v = 60$

22			<p>Keterangan: $kk = 3,0$ $v = 60$</p>
23			<p>Keterangan: $kk = 3,3$ $v = 60$</p>
24			<p>Keterangan: $kk = 3,6$ $v = 60$</p>

25		
		Keterangan: kk = 4,0 v = 60

5.2.4 Batch Processing Testing

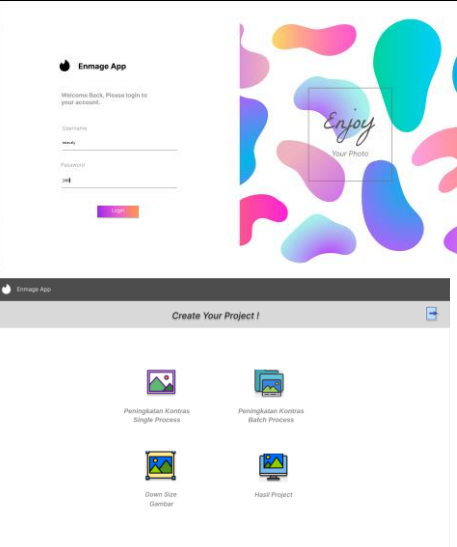


Tabel 5.5 Pengujian Waktu *Batch Processing* Aplikasi

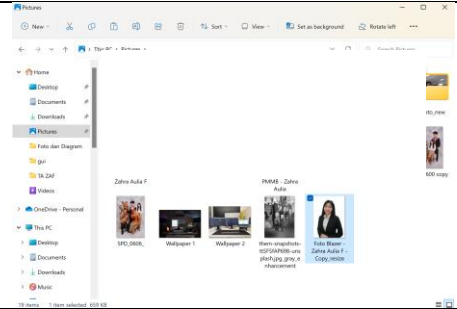
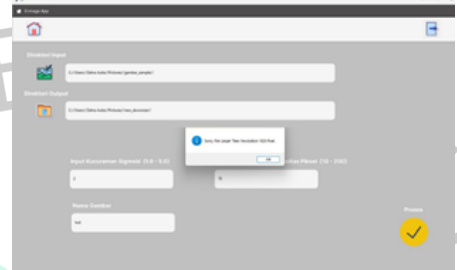
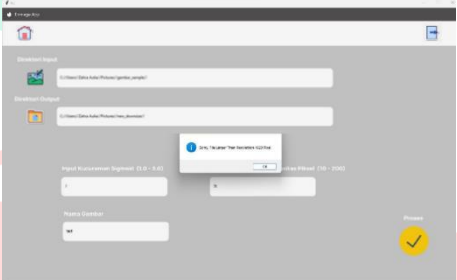
No	Percobaan	Banyak Gambar	VGA (row 640 x 480 piksel)	SVGA (800 x 600 piksel)	XGA (1024 x 768 piksel)	HD (1280 x 720 piksel)	Full HD (1920 x 1080 piksel)
1	Pertama	10 Gambar	8	11	12	15	18
	Kedua		7	13	12	14	18
	Ketiga		8	11	13	15	19
Rata-rata			7,66	11,66	12,33	14,66	18,33
2	Pertama	50 Gambar	61	57	56	65	69
	Kedua		62	60	58	63	69
	Ketiga		62	58	56	66	70
Rata-rata			61,33	58,33	56,66	64,66	69,33
3	Pertama	100 Gambar	136	323	123	136	139
	Kedua		138	325	124	138	141
	Ketiga		137	324	123	140	142
Rata-rata			136,66	324	123,33	138	140,66
4	Pertama	200 Gambar	271	567	511	547	560
	Kedua		279	571	513	545	562
	Ketiga		277	569	513	548	570
Rata-rata			275,66	569	512,33	546,66	564

Tabel 5.5 merupakan hasil pengujian waktu dengan beberapa jumlah pengelompokan dalam memproses gambar secara *Batch Processing* juga secara otomatis langsung menyimpan hasil gambar ke alamat direktori tujuan. Perlu menjadi catatan bahwa kecepatan proses aplikasi juga dipengaruhi spesifikasi laptop atau komputer pengguna sampai dengan gambar *full HD* 1080 piksel. Hasil dari pengujian waktu *batch processing* tabel dapat menjadi bahan perbandingan dari hasil proses menggunakan Waifu2x yakni memerlukan 7 detik untuk memproses 1 gambar. Dalam menggunakan Waifu2x pengguna juga harus secara satu per satu memilih dan *input* gambar untuk diproses.

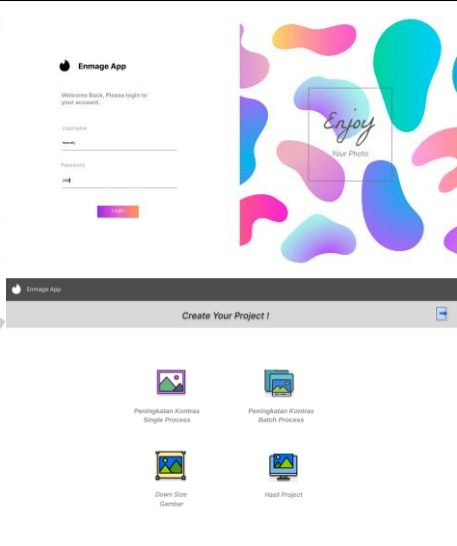
5.2.5 Black Box Testing

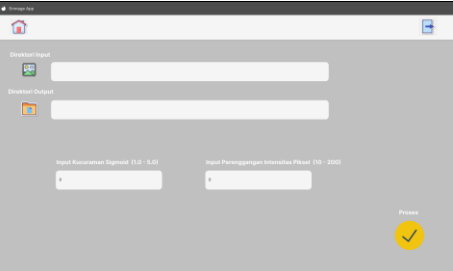
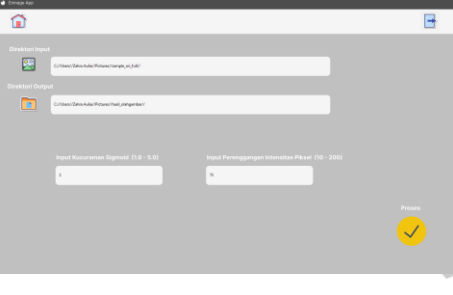
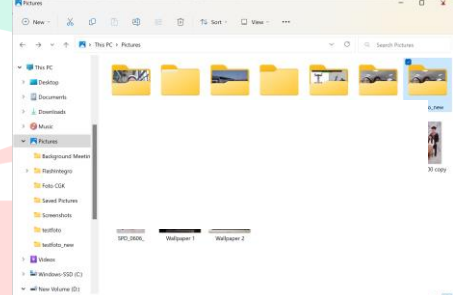
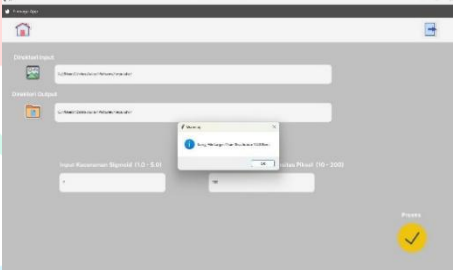
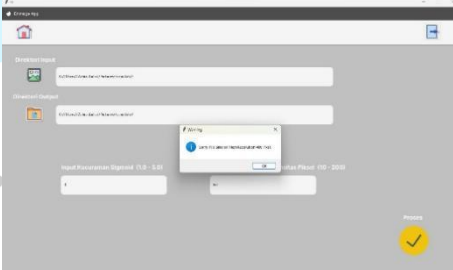
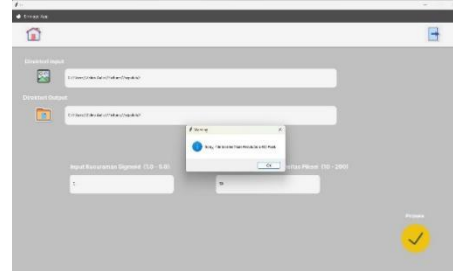
Tabel 5.6 *Black Box Testing* Fitur *Down Size* Gambar

No	Pengujian	Hasil yang diharapkan	Hasil Pengamatan	Kesimpulan
1	User membuka aplikasi Peningkatkan Kontras Gambar.	Menampilkan halaman <i>Login</i> . Dan berhasil <i>Login</i> untuk masuk ke <i>Dashboard</i> .		Berhasil.
2	Klik menu <i>Down Size</i> Gambar yang ada pada <i>Dashboard</i> .	Menampilkan halaman <i>Down Size</i> Gambar.		Berhasil.
3	<i>Input</i> direktori gambar direktori, hasil <i>output</i> , dan besaran <i>row down scale</i> gambar.	Menyimpan lokasi direktori file dan memproses gambar dengan nilai <i>row input</i> .		Berhasil.

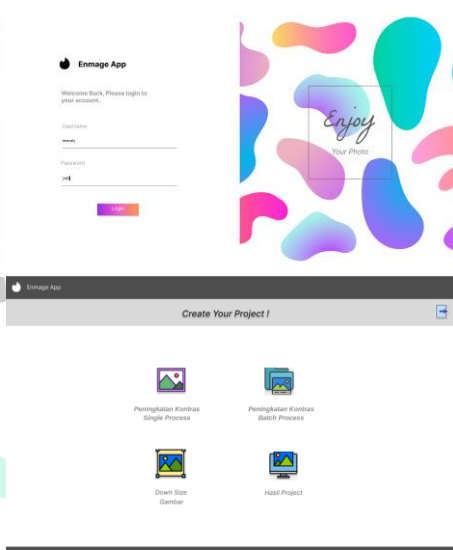
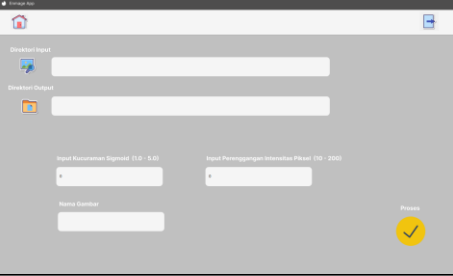
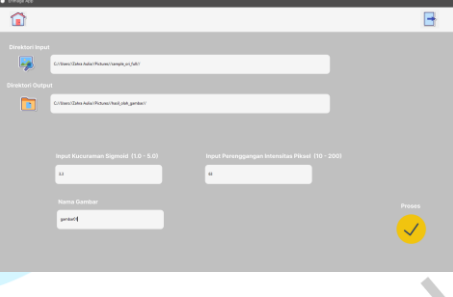
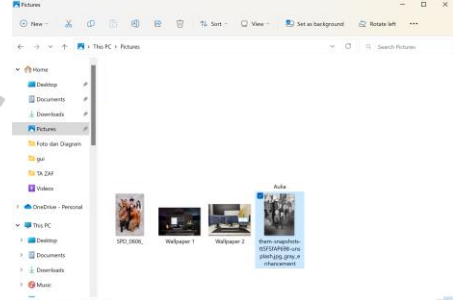
4	klik tombol <i>Process</i> .	Proses olah <i>Down Size</i> gambar dan menyimpan hasil gambar yang telah berhasil.		Berhasil.
5	Klik <i>Process</i> dengan kondisi ukuran gambar kurang dari 640 x 480 piksel.	Proses <i>Down Size</i> gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.
6	Klik <i>Process</i> dengan kondisi format gambar selain .jpg	Proses gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.

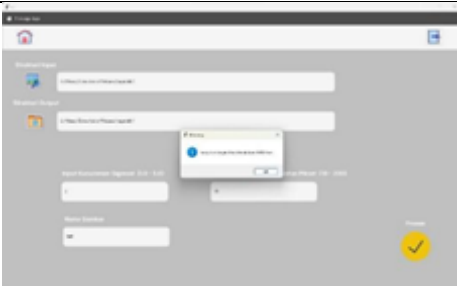
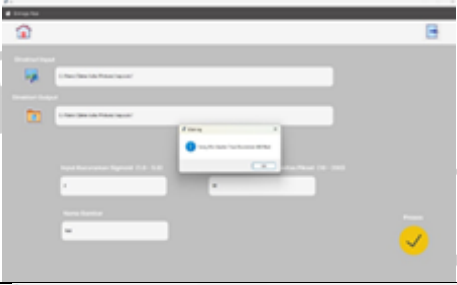
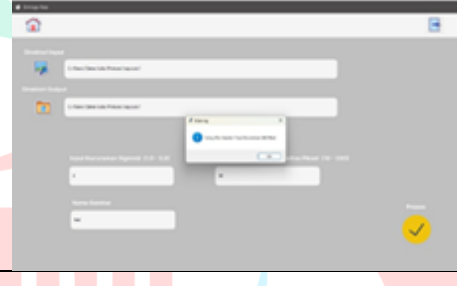
Tabel 5.7 *Black Box Testing* Fitur Peningkatan Kontras *Batch Process*

No	Pengujian	Hasil yang diharapkan	Hasil Pengamatan	Kesimpulan
1	User membuka aplikasi Peningkatan Kontras Gambar.	Menampilkan halaman <i>Login</i> . Dan berhasil <i>Login</i> untuk masuk ke <i>Dashboard</i> .		Berhasil.

2	Klik menu Peningkatan Kontras <i>Batch Process</i> untuk tingkatan gambar kontras secara berkelompok.	Menampilkan halaman Peningkatan Kontras <i>Batch Process</i> .		Berhasil.
3	<i>Input</i> direktori gambar, direktori hasil <i>output</i> , kecuraman sigmoid. Dan perenggangan piksel.	Menyimpan lokasi direktori file dan memproses gambar dengan nilai sigmoid dan perenggangan piksel <i>input</i> .		Berhasil.
4	Klik tombol <i>Process</i> .	Proses olah gambar dan menyimpan hasil gambar berhasil.		Berhasil.
5	Klik tombol <i>Process</i> dengan kondisi ukuran gambar melebihi 1080 x 1920 piksel.	Proses olah gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.
6	Klik tombol <i>Process</i> dengan kondisi ukuran gambar kurang dari 640 x 480 piksel.	Proses olah gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.
7	Klik tombol <i>Process</i> dengan format gambar selain .jpg	Proses olah gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.

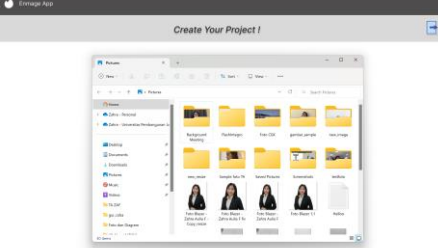
Tabel 5.8 *Black Box Testing* Fitur Peningkatan Kontras *Single Process*

No	Pengujuan	Hasil yang diharapkan	Hasil Pengamatan	Kesimpulan
1	<p>User membuka aplikasi Enmage App. Peningkatan Kontras Gambar.</p>	<p>Menampilkan halaman <i>Login</i>. Dan berhasil <i>Login</i> untuk masuk ke <i>Dashboard</i>.</p>		<p>Berhasil.</p>
2	<p>Klik menu Peningkatan Kontras <i>Single Process</i> untuk tingkatan gambar kontras secara berkelompok.</p>	<p>Menampilkan halaman Peningkatan Kontras <i>Single Process</i>.</p>		<p>Berhasil.</p>
3	<p>Input direktori gambar, direktori hasil <i>output</i>, kecuraman sigmoid, perenggangan piksel, dan nama gambar.</p>	<p>Menyimpan lokasi direktori file dan memproses gambar dengan nilai sigmoid dan perenggangan piksel <i>input</i>.</p>		<p>Berhasil.</p>
4	<p>Klik tombol <i>Process</i>.</p>	<p>Proses olah gambar dan menyimpan hasil gambar berhasil.</p>		<p>Berhasil.</p>

5	Klik tombol <i>Process</i> dengan kondisi ukuran gambar melebihi 1080 x 1920 piksel.	Proses olah gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.
6	Klik tombol <i>Process</i> dengan kondisi ukuran gambar kurang dari 640 x 480 piksel.	Proses olah gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.
7	Klik tombol <i>Process</i> dengan format gambar selain .jpg	Proses olah gambar tidak terjadi dan mengeluarkan <i>alert</i> .		Berhasil.

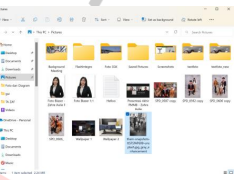


Tabel 5.9 *Black Box Testing* Fitur Hasil Project

No	Pengujian	Hasil yang diharapkan	Hasil Pengamatan	Kesimpulan
1	User membuka aplikasi Peningkatkan Kontras Gambar.	Menampilkan halaman <i>Login</i> . Dan berhasil <i>Login</i> untuk masuk ke <i>Dashboard</i> .		Berhasil.

2	Klik menu Hasil <i>Project</i> untuk melihat file <i>explorer</i> dari aplikasi.	Membuka file <i>explorer</i> penyimpanan komputer.		Berhasil.
---	----------------------------------------------------------------------------------	----------------------------------------------------	------------------------------------------------------------------------------------	-----------

5.2.6 White Box Testing

Tabel 5.10 White Box Testing

No	Program	Keterangan	Hasil Tampilan
1	<pre> def info(): messagebox.showinfo("Info", "Successfully!") def Program_Sigmoid(path_input, path_output, sigmoid, Intensitas): kk = sigmoid # Steepness coefficient of the tranfer function v = Intensitas / 255 # Pixel intensity of which maximum stretching happens # -----DEVELOPING FUNCTIONS----- # # FUNCTION TO CREATE A GRAYSCALE IMAGE FROM A COLOR IMAGE def RGB_to_gray(p): # p is an array of an RGB image R, G, B = p[:, :, 0], p[:, :, 1], p[:, :, 2] gray = np.round(0.2989 * R + 0.5870 * G + 0.1140 * B) gray = gray.astype(int) return gray # FUNCTION TO FIND MIN AND MAX OF ALL PIXEL INTENSITIES OCCUR IN AN IMAGE def minmax_color(p): # p is an array of a color image min = np.amin([np.amin(p[:, :, 0]), np.amin(p[:, :, 1]), np.amin(p[:, :, 2])]) max = np.amax([np.amax(p[:, :, 0]), np.amax(p[:, :, 1]), np.amax(p[:, :, 2])]) return (min, max) def minmax_gray(p): # p is an array of a grayscale image min = np.amin(np.amin(p[:, :])) max = np.amax(np.amax(p[:, :])) return (min, max) # CREATING THE TRANSFER FUNCTION FOR PIXELS HAVING INTENSITY > THRESHOLD (v) def TFU(x, v): # Transfer function for upper pixel intensities k = kk * 9 ** v s = (1 - v) / 0.5 e = np.e return s * 1 / (1 + e ** (-k * (x - 0.5 + 0.5 - v))) - 0.5 * s + v # CREATING THE TRANSFER FUNCTION FOR PIXELS HAVING INTENSITY < THRESHOLD (v) def TFL(x, v): # Transfer function for lower pixel intensities k = kk * 9 ** (1 - v) s = v / 0.5 e = np.e return s * 1 / (1 + e ** (-k * (x + 0.5 - 0.5 - v))) - 0.5 * s + v # FUNCTION TO CREATE THE HISTOGRAM OF A GRAYSCALE PICTURE def histogram(p): # p is an array of a grayscale picture # Preparing an array containing all intensity values (0, 1, 2, ... 255) values = np.zeros(256) values = values.astype(int) for i in range(0, 256): values[i] = 1 # Preparing an array to store the occurance frequency of each intensity value freq = np.zeros(256) freq = freq.astype(int) </pre>	<p>Algoritma pengolahan gambar dengan <i>Modified Sigmoid Transfer Function</i>.</p> <p>berhasil.</p>	<p>Gambar berhasil dirposes dan tersimpan di direktori tujuan.</p>    <p>Untuk hasil gambar lebih jelas dapat dilihat pada Gambar 5.9 dan Gambar 5.11.</p>

```

row, col = p.shape
for m in range(0, row):
    for n in range(0, col):
        measure = p[m, n]
        index = measure
        freq[index] = freq[index] + 1
plt.figure(figsize=(6, 3))
return plt.bar(values, freq, width=3, bottom=None, align='center', data=None);
plt.show()

#-----#
#-----MAIN PROGRAM-----#
#-----#
# READING A COLOR IMAGE
# path_start = "Gambar/"
# path = path_start + file + ".jpg"
# pic = np.round(imread(path))
# pic = pic.astype(int)

def EnhancementImage(image, folder, kk, v):
    pic = np.round(imread(folder + "/" + image))
    pic = pic.astype(int)
    # PREPARING THE INPUT IMAGE, NORMALIZATION AND THE TEMPLATE FOR OUTPUT IMAGE
    # BEFORE THE CONTRAST ENHANCEMENT
    pic_gray = RGB_to_gray(pic) # Creating grayscale image from the color image
    pic_gray_ = pic_gray.astype(int)
    pic_gray_float = pic_gray / 255 # Normalization
    pic_gray_float_new = pic_gray_float # Creating the template for the new image
    # APPLYING THE TRANSFER FUNCTION FOR CONTRAST ENHANCEMENT
    row, col = pic_gray_float.shape
    for m in range(0, row):
        for n in range(0, col):
            x = pic_gray_float[m, n]
            if x > v:
                pic_gray_float_new[m, n] = TFU(x, v)
                # print(x)
            else:
                pic_gray_float_new[m, n] = TFL(x, v)
                # print(x)

    # DENORMALIZATION OF THE OUTPUT IMAGE
    pic_gray_new = np.round(pic_gray_float_new * 255)
    pic_gray_new = pic_gray_new.astype(int)
    # SHOWING THE MIN AND THE MAX OF PIXEL INTENSITIES
    a, b = minmax_color(pic)
    print('Shape, min and max of pic: ', pic.shape, a, b)
    c, d = minmax_gray(pic_gray)
    print('Shape, min and max of pic_gray: ', pic_gray.shape, c, d)
    e, f = minmax_gray(pic_gray_new)
    print('Shape, min and max of pic_gray_new: ', pic_gray_new.shape, e, f)
    # Saving the processed image
    path = path_output + image + "_gray_enhancement" + ".jpg"
    imsave(path, pic_gray_new)

# Baca Seluruh Data yang ada di Folder
folder_dir = path_input
for images in os.listdir(folder_dir):

    # Cek format file sesuai ketentuan
    if (images.endswith(".jpg")):
        print(images)
        EnhancementImage(images, folder_dir, kk, v)
        # plt.figure(1); plt.imshow(pic); plt.show()

info()

```

```

def info():
    messagebox.showinfo("Info", "Successfully!")

def Program_Resize(path_input, path_output, size):
    path = path_input
    files = os.listdir(path)
    for file in files:
        pic = imread(os.path.join(path, file))
        row, col, ch = pic.shape
        row_res = size # ukuran gambar yang ditargetka
        if (row < row_res + 1):
            print("The image is small")
            pic_res = pic
            # if the image size <= row_res, do not resize it if
        if row > row_res:
            res_factor = row / row_res
            col_res = int(round(col / res_factor))
            # Creating a template Array (canvas)
            pic_res = np.zeros(shape=(row_res, col_res, 3), dtype=np.uint8)

            for m_res in range(0, row_res - 1):
                print(m_res)
                for n_res in range(0, col_res - 1):
                    m = int(round(res_factor * m_res))
                    n = int(round(res_factor * n_res))
                    pic_res[m_res, n_res, :] = pic[m, n, :]

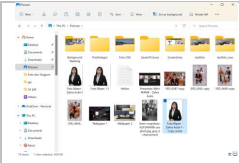
    plt.imshow(pic_res)
    patho = path_output + file
    plt.imsave(patho, pic_res)
    # plt.show()

```

```
info()
```

Algoritma
downscale
gambar

berhasil.



Untuk hasil gambar lebih
jelas dapat dilihat pada
Tabel 5.1.