PENERAPAN METODE *BOX COUNTING* UNTUK IDENTIFIKASI TELAPAK TANGAN

SKRIPSI

Oleh
Arum Andary Ratri
NIM 111810101001

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2015

PENERAPAN METODE *BOX COUNTING* UNTUK IDENTIFIKASI TELAPAK TANGAN

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan pendidikan Program Studi Matematika (S1)
dan mencapai gelar Sarjana Sains

Oleh
Arum Andary Ratri
NIM 111810101001

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2015

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Ibunda Siti Khoiriyah dan Ayahanda Ali Maskur yang senantiasa mendoakan, memberikan bimbingan, semangat serta kasih sayang yang tulus;
2. adik-adikku tersayang Bintang Pramono Timur dan Cundamanik;
3. seluruh guruku mulai dari taman kanak-kanak sampai perguruan tinggi yang senantiasa memberikan ilmu yang bermanfaat;
4. seluruh keluarga besar UKPKM Tegalboto yang telah memberikan wadah untuk berorganisasi dan belajar;
5. Almamater Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam, SMAN 1 Gambiran, SMPN 1 Genteng, SDN 6 Kembiritan, dan TK ABA III;
6. segenap keluarga KRAMAT'11 tercinta.

MOTO

Maka nikmat Tuhan-mu yang manakah yang kamu dustakan?

(Terjemahan Surat Ar-Rahman Ayat 13)[*)]

Aku menurut dugaan hambaKu, dan Aku bersamanya apabila ia ingat kepadaKu

(Hadist Qudsi)[**)]

[*)]Kementerian Agama Republik Indonesia. 2013. Al Quran Terjemah Perkata Asbabun Nuzul dan Tafsir Bil Hadis. Bandung. Penerbit Semesta Al-Quran.

[**)]Abu Hurairah ra. Hadits ditakhrij oleh Turmidzi. 2007. Himpunan Hadits Qudsi. Rembang. Setia Kawan.

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Arum Andary Ratri

NIM : 111810101001

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul "Penerapan Metode *Box Counting* untuk Identifikasi Telapak Tangan" adalah benar-benar karya sendiri, kecuali kutipan yang telah saya sebutkan sumbernya, belum pernah diajukan pada institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, Juni 2015

Yang menyatakan,

Arum Andary Ratri

NIM 111810101001

SKRIPSI

PENERAPAN METODE BOX COUNTING UNTUK IDENTIFIKASI TELAPAK TANGAN

Oleh

Arum Andary Ratri

NIM 111810101001

Pembimbing

Dosen Pembimbing Utama : Kosala Dwidja Purnomo, S.Si., M.Si.

Dosen Pembimbing Anggota : Kusbudiono, S.Si., M.Si.

PENGESAHAN

Skripsi berjudul "Penerapan Metode *Box Counting* untuk Identifikasi Telapak Tangan" telah diuji dan disahkan pada:

Hari, tanggal :

Tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Jember

Tim Penguji:

Ketua,	Sekretaris,
Kosala Dwidja Purnomo, S.Si., M.Si. NIP. 19690828 199802 1 001	Kusbudiono, S.Si., M.Si. NIP. 19770430 200501 1 001
Anggota I,	Anggota II,
Prof. Drs. Kusno, DEA., Ph.D. NIP. 19610108 198602 1 001	Ahmad Kamsyakawuni, S.Si., M.Kom. NIP. 19721129 199802 1 001

Mengesahkan

Dekan,

Prof. Drs. Kusno, DEA., Ph.D.
NIP. 19610108 198602 1 001

RINGKASAN

Penerapan Metode Box Counting untuk Identifikasi Telapak Tangan; Arum Andary Ratri, 111810101001; 2015; 37 halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Metode *box counting* adalah metode penghitungan dimensi fraktal dengan membagi citra menjadi kotak-kotak kecil dalam berbagai variasi ukuran. Fraktal sendiri berasal dari kata latin *fractus* yang artinya patah, pecah atau tidak teratur. Beberapa fraktal, apabila dipecah dan diambil beberapa bagian kecilnya jika diperbesar akan terlihat mirip dengan fraktal aslinya *(self similarity)*. Penelitian menggunakan metode *box counting* diterapkan pada identifikasi telapak tangan dengan mengambil ciri berupa garis-garis telapak tangan.

Tahapan pada identifikasi telapak tangan adalah proses awal pengolahan citra digital, ekstraksi ciri dimensi fraktal yang dihitung menggunakan metode *box counting*, dan pencocokan nilai dimensi fraktal menggunakan koefisien korelasi. Proses pengolahan citra yang diterapkan antara lain proses *cropping*, pengubahan menjadi citra *grayscale*, proses adaptif histogram ekualisasi, dan proses *thresholding*. Pada tahap pencocokan menggunakan koefisien korelasi, nilai-nilai dimensi fraktal yang dihasilkan dari proses penghitungan metode *box counting* dibandingkan menggunakan rumus koefisien korelasi.

Pada penelitian ini, dilakukan pengujian dua puluh citra telapak tangan uji yang lima belas diantaranya adalah citra milik orang yang sama dengan citra telapak tangan acuan, sedangkan lima lainnya berasal dari citra milik orang yang berbeda. Kemudian kedua puluh citra telapak tangan uji tersebut diidentifikasi. Selanjutnya pengujian menggunakan beberapa variasi ukuran kotak untuk mengetahui seberapa berpengaruh ukuran kotak pembagi pada metode *box counting.* Pengujian juga menggunakan beberapa nilai koefisien korelasi. Hal ini dilakukan untuk mengetahui nilai koefisien korelasi yang paling sesuai untuk digunakan pada program identifikasi telapak tangan.

Hasil yang didapatkan dari identifikasi dua puluh citra telapak tangan uji adalah memperoleh persentase keberhasilan sebesar 85%. Pada percobaan pengubahan variasi ukuran kotak didapatkan persentase keberhasilan 85% untuk variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \ldots, \frac{1}{4096}$ dan 65% untuk variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \frac{1}{128}$. Sedangkan pada percobaan pengubahan nilai koefisien korelasi didapatkan persentase keberhasilan program 65% untuk nilai koefisien korelasi 0,85 dan 0,95 serta 85% untuk nilai koefisien korelasi 0,99.

PRAKATA

Puji syukur kehadirat Allah SWT. atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Penerapan Metode *Box Counting* untuk Identifikasi Telapak Tangan". Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata satu (S1) pada jurusan Matematika Fakultas MIPA Universitas Jember.

Penyusunan skripsi ini tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Prof. Drs. Kusno, DEA., Ph.D. selaku Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam yang telah mengesahkan penulisan skripsi ini;
2. Kosala Dwidja Purnomo, S.Si., M.Si. selaku Dosen Pembimbing Utama dan Kusbudiono, S.Si., M.Si. selaku Dosen Pembimbing Anggota yang telah memberikan bimbingan serta arahan pada penulis sehingga skripsi ini dapat terselesaikan dengan baik;
3. Prof. Drs. Kusno, DEA., Ph.D. dan Ahmad Kamsyakawuni, S.Si., M.Kom. selaku Dosen Penguji yang telah memberikan kritik serta saran sehingga skripsi ini menjadi lebih baik;
4. Ibunda Siti Khoiriyah dan Ayahanda Ali Maskur yang tidak pernah berhenti mendoakan, mendidik, memberikan nasehat dan kasih sayang yang tulus;
5. seluruh staf pengajar Jurusan Matematika Fakultas MIPA Universitas Jember yang telah memberikan ilmu serta bimbingannya;
6. keluarga besar KRAMAT'11 yang telah memberikan persahabatan dan pengalaman-pengalaman yang luar biasa selama masa perkuliahan;
7. keluarga besar UKPKM Tegalboto yang telah memberikan wadah untuk berorganisasi dan berbagai pengalaman luar biasa;
8. teman-teman kos Assalam 2 dan teman-teman bimbingan belajar Rumah Cerdas yang selalu memberikan motivasi dan bantuannya;
9. semua pihak yang tidak dapat disebutkan satu persatu.

Penulis juga menerima segala kritik dan saran dari semua pihak demi kesempurnaan skripsi ini. Akhirnya penulis berharap, semoga skripsi ini dapat bermanfaat.

Jember, 2015 Penulis

DAFTAR ISI

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR LAMPIRAN

Halaman

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Sistem pengenalan diri (*personal recognition*) menjadi sesuatu yang sangat penting mengingat era modern ini berbagai aktivitas membutuhkan identitas diri untuk meningkatkan keamanan. Mulai dari pelayanan kesehatan, pengurusan rekening bank, pelayanan penerbangan, dan lain-lain. Sistem pengenalan diri bertujuan untuk mengetahui identitas seseorang. Terdapat dua tipe sistem pengenalan diri, yaitu verifikasi dan identifikasi. Sistem verifikasi bertujuan untuk menerima atau menolak identitas yang diklaim oleh seseorang, sedangkan sistem identifikasi bertujuan untuk memecahkan identitas seseorang (Putra, 2009).

Biometrika adalah pengembangan dari metode dasar identifikasi seseorang dengan menggunakan karakteristik alami manusia sebagai basisnya (Mulyadi, 2013). Salah satu cara untuk mengenali seseorang adalah dengan identifikasi telapak tangan (*palmprint*). Telapak tangan dapat digunakan sebagai biometrika pengenalan diri karena memiliki ciri khas yang unik. Setiap telapak tangan memiliki struktur yang detail dengan ciri garis-garis utama (*principle-line features*) dan ciri garis-garis kusut (*wrinkles features*). Bahkan tiap orang memiliki telapak tangan kanan dan kiri yang berbeda. Demikian juga, struktur garis pada telapak tangan tidak berubah-ubah atau stabil selama berpuluh-puluh tahun.

Salah satu cara yang dapat diterapkan untuk mengidentifikasi telapak tangan adalah dengan metode ekstraksi ciri berbasis dimensi fraktal. Tujuan dari ekstraksi ciri adalah untuk mendapatkan informasi-informasi penting dari tekstur telapak tangan. Metode yang dianggap sesuai untuk mengekstraksi ciri telapak tangan tersebut adalah dengan menghitung dimensi fraktal menggunakan metode *box counting*.

Metode *box counting* adalah metode penghitungan dimensi fraktal dengan membagi citra menjadi kotak-kotak kecil dalam berbagai variasi ukuran. Fraktal sendiri berasal dari kata latin *fractus* yang artinya pecah atau tidak teratur (Mulyadi, 2013). Jadi, fraktal adalah benda geometris yang kasar dan tidak teratur. Ketidak teraturan tersebut misalnya pada bangun-bangun di alam seperti gumpalan awan, garis pantai, dan ranting-ranting pohon. Layaknya bangun geometri yang lain, fraktal juga memiliki dimensi. Namun dimensi fraktal berbentuk pecahan.

Sebelumnya, telah dilakukan beberapa penelitian terkait topik tersebut. Diantaranya adalah Subiantoro (2005) yang menentukan dimensi objek fraktal menggunakan metode *box counting*. Subiantoro meneliti dimensi objek fraktal pada dua jenis objek, yaitu pada objek fraktal seperti *koch snowflake* dan fraktal yang ada di alam seperti daun paku-pakuan. Putra (2009) meneliti sistem verifikasi biometrika telapak tangan dengan metode dimensi fraktal dan *lacunarity*. Selanjutnya penelitian oleh Mulyadi (2013) yang membahas mengenai sistem identifikasi telapak tangan dengan ekstraksi ciri berbasis dimensi fraktal.

Pada skripsi ini, penulis mencoba membahas mengenai penerapan metode *box counting* untuk mengidentifikasi telapak tangan. Pada penelitian sebelumnya oleh Mulyadi tingkat keberhasilan dalam penelitian sangat dipengaruhi oleh akuisisi citra dan proses awal pengolahan citra. Kesalahan pada saat identifikasi disebabkan akuisisi citra dan proses pengolahan awal yang belum sempurna. Pada penelitian ini akan digunakan kamera yang dilengkapi dengan pengaturan terhadap fokus dan tingkat pencahayaan pada saat pengambilan citra. Sehingga diharapkan tekstur telapak tangan dapat terlihat jelas dan tidak terdapat kilatan cahaya.

1.2 Rumusan Masalah

Masalah yang akan dibahas adalah bagaimana penerapan metode *box counting* untuk mengidentifikasi telapak tangan.

1.3 Batasan Masalah

Adapun batasan masalah dalam penelitian ini antara lain:

a. Citra telapak tangan yang digunakan adalah citra berwarna yang diambil dalam keadaan diam;
b. Telapak tangan yang digunakan untuk penelitian haruslah telapak tangan normal (tidak cacat), berada pada kondisi bersih dan tidak terdapat coretan;
c. Ciri biometrika telapak tangan yang akan diproses adalah garis-garis telapak tangan;
d. Telapak tangan yang diuji harus sama dengan telapak tangan acuan, yaitu dalam hal ini dipilih telapak tangan kiri
e. Sudut pengambilan citra adalah tegak lurus dengan objek telapak tangan.

1.4 Tujuan

Tujuan dari penelitian ini adalah untuk mengetahui bagaimana penerapan metode *box counting* dalam mengidentifikasi telapak tangan seseorang.

1.5 Manfaat

Adapun manfaat yang dapat diperoleh dari penulisan skripsi ini antara lain:

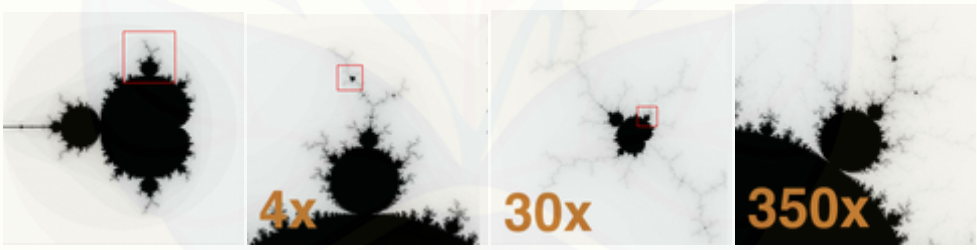
a. Memudahkan dalam sistem pengenalan diri seseorang
b. Dengan adanya suatu program pengenalan diri seseorang, diharapkan dapat semakin meningkatkan keamanan karena dapat memudahkan dalam hal identifikasi seseorang.

BAB 2. TINJAUAN PUSTAKA

2.1 Fraktal

Istilah fraktal diperkenalkan pertama kali oleh Benoit Mandelbrot pada tahun 1977 dalam bukunya yang berjudul "*The Fractal Geometry of Nature*". Penemuan Mandelbrot tentang geometri fraktal tersebut merupakan salah satu penemuan yang sangat berpengaruh terhadap perkembangan ilmu matematika pada pertengahan abad dua puluh. Konsep dasar fraktal sebenarnya sudah ditemukan jauh sebelum Mandelbrot. Ide dasar mengenai keserupaan diri sudah ditulis oleh Leibnizt, Kant, Lichtenberg, Cantor, dan Hausdorff (Evertsz, 1995).

Menurut Helja (2013) fraktal berasal dari kata latin *fractus* yang artinya patah, pecah atau tidak teratur. Beberapa fraktal, apabila dipecah dan diambil beberapa bagian kecilnya jika diperbesar akan terlihat mirip dengan fraktal aslinya *(self similarity)*. Fraktal dikatakan memiliki detail yang tak hingga dan pada tingkat perbesaran yang berbeda, ia memiliki struktur serupa diri dengan fraktal aslinya, (lihat Gambar 2.1).



Gambar 2.1 Ilustrasi perbesaran objek fraktal (Sumber: Wikipedia.org)

Fraktal menurut keserupaan dirinya terbagi menjadi tiga macam, yaitu serupa diri secara persis, serupa diri sebagian, dan serupa diri secara statistik. Serupa diri secara persis memiliki struktur fraktal yang sangat identik di segala skala. Karakteristik seperti ini biasanya terjadi pada bentuk fraktal yang terdefinisi

secara matematika seperti *koch snowflake* dan segitiga *sierpinski*. Karakteristik yang kedua adalah serupa diri sebagian. Fraktal jenis ini memiliki keserupaan diri yang tidak terlalu mirip jika skalanya diubah. Contoh dari fraktal jenis tersebut adalah himpunan Mandelbrot. Jenis yang ketiga adalah serupa diri secara statistik. Keserupaan dirinya bersifat statistik pada skala tertentu, jenis ini memiliki tingkat serupa diri yang paling lemah (Subiantoro, 2005).

2.2 Dimensi Fraktal

Menurut kamus matematika, dimensi mengacu pada sifat-sifat yang dinamakan panjang, luas dan volume. Dimensi adalah bilangan yang menyatakan kebebasan untuk melakukan pergerakan di sebuah ruang. Pada umumnya, dimensi suatu objek adalah bilangan yang mendefinisikan bentuk dan ukuran objek tersebut. Konsep dimensi fraktal digunakan sebagai indikator kekasaran permukaan. Untuk suatu himpunan fraktal, dimensi Hausdorff- Besicovitch lebih besar dibandingkan dimensi topologi (Sankar, 2010).

Dimensi menurut Euclid berbeda dengan dimensi menurut fraktal. Seperti yang kita ketahui bersama, dalam dimensi Euclidean titik merupakan dimensi nol, garis merupakan dimensi satu, bidang merupakan dimensi dua, dan ruang merupakan dimensi tiga. Namun, pada dimensi fraktal kita mengenal dimensi pecahan, seperti dimensi 2,7 dan dimensi 1,5 (Subiantoro, 2005).

Terdapat dua konsep dimensi yang berkaitan dengan karakteristik objek fraktal, yaitu dimensi topologi dan dimensi Hausdorff. Dimensi topologi dilambangkan dengan D_T dan dimensi Hausdorff dilambangkan dengan D. Dimensi topologi ini sesuai dengan dimensi menurut Euclid, yakni nilainya selalu berupa bilangan bulat. Besarnya dimensi topologi suatu objek di ruang vector R^n adalah bilangan bulat antara 0 dan n (Mandelbrot, 1983).

Suatu fraktal didefinisikan sebagai himpunan yang mana dimensi Hausdorff Besicovitch lebih kuat dari pada dimensi topologinya. Dengan kata lain dimensi pada fraktal merupakan dimensi Hausdorff, yang memiliki nilai bukan bilangan bulat. Dimensi Hausdorff dilambangkan dengan D, banyaknya subunit atau subsegmen hasil iterasi dari suatu objek fraktal dilambangkan dengan N.

Sedangkan panjangnya subsegmen tersebut dilambangkan dengan r. Sehingga hubungan antara D, N, dan r dinyatakan dengan persamaan $N = \left(\frac{1}{r}\right)^D$ (Mandelbrot, 1983). Dengan mengambil logaritma dari kedua ruas persamaan tersebut, dimensi dapat dicari dengan persamaan (2.1):

$$D = \frac{\log(N)}{\log\left(\frac{1}{r}\right)} \tag{2.1}$$

2.3 Metode Box Counting

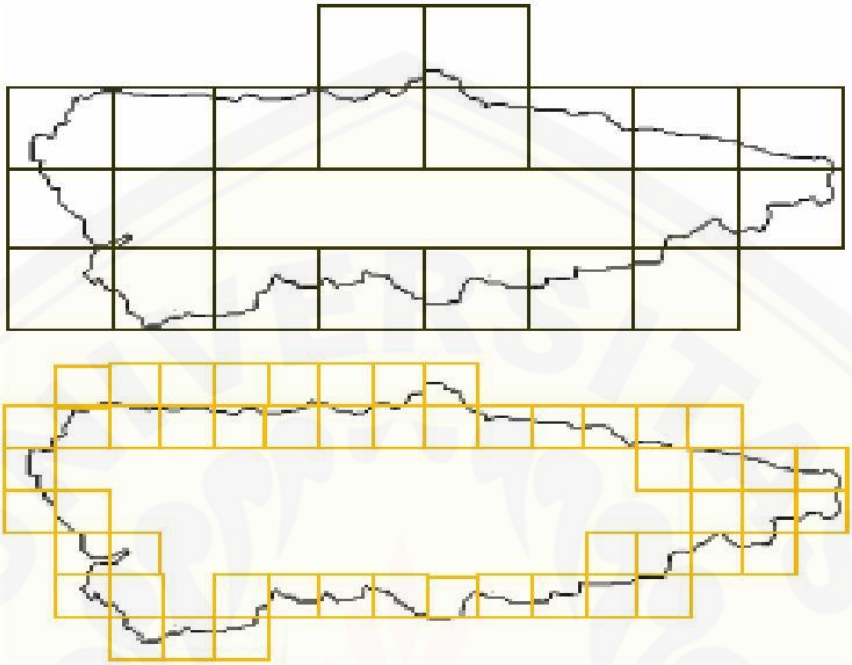
Terdapat beberapa metode untuk menentukan dimensi fraktal. Salah satu metode yang dapat digunakan untuk menentukan dimensi fraktal adalah metode *box counting*. Metode *box counting* sering dikenal sebagai metode penghitungan kotak. Metode ini membagi suatu objek menjadi beberapa bagian kotak dengan berbagai variasi ukuran. Langkah-langkah bekerja dengan metode *box counting* adalah sebagai berikut:

a. Mengambil citra suatu objek fraktal yang akan dihitung dimensinya
b. Membagi citra tersebut ke dalam kotak-kotak dengan variasi ukuran yang berbeda s
c. Menghitung banyaknya kotak yang berisi bagian objek pada citra N
d. Menghitung besarnya dimensi D dengan persamaan (2.2) (Putra, 2009):

$$D(s) = \frac{\log N(s)}{\log\frac{1}{s}} \tag{2.2}$$

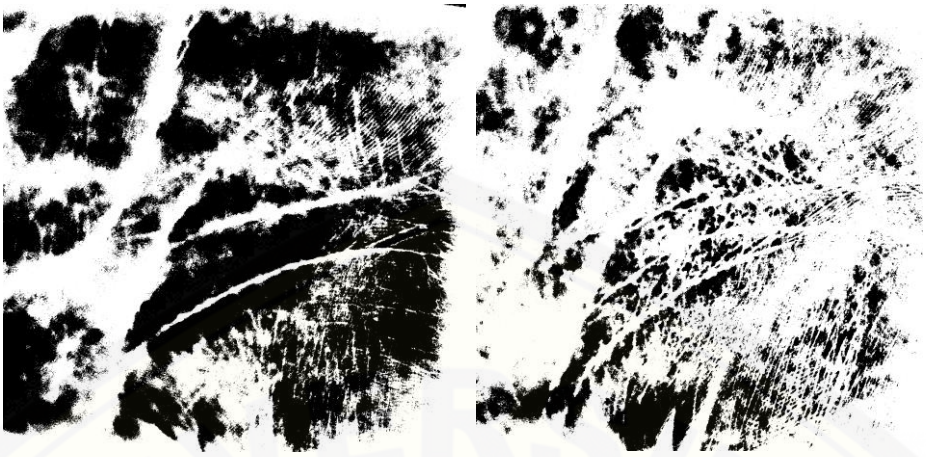
Gambar 2.2 merupakan ilustrasi untuk metode *box counting* dalam menghitung garis pantai. Gambar pertama menunjukkan pembagian kotak dengan s sebesar $\frac{1}{8}$ dari ukuran awal dan jumlah kotak terisi N sebanyak 22 kotak. Sehingga nilai dimensi fraktal dengan ukuran s sebesar $\frac{1}{8}$ adalah 1,486478. Sedangkan untuk gambar kedua, ukuran s adalah $\frac{1}{16}$ dengan jumlah kotak terisi N sebanyak 52 kotak. Berdasarkan pembagian kotak dengan ukuran tersebut, dihasilkan dimensi fraktal sebesar 1,42511. Nilai dimensi fraktal yang didapatkan akan bervariasi sesuai dengan penentuan nilai s. Nilai-nilai dimensi fraktal yang

berbeda tersebut, selanjutnya dijadikan sebagai vektor ciri suatu citra yang akan diproses lebih lanjut menggunakan metode pencocokan koefisien korelasi.



Gambar 2.2 Pembagian kotak pada metode *box counting* (Sumber: wordpress.com)

Mengenai penghitungan metode *box counting* pada citra telapak tangan, kategori kotak terisi adalah citra telapak tangan yang memiliki nilai derajat keabuan tidak nol, atau yang berwarna hitam. Setelah melalui tahapan pengolahan citra, garis-garis pada citra telapak tangan akan memiliki tingkat derajat keabuan yang berbeda. Gambar 2.3 merupakan citra telapak tangan yang telah diolah pada pengolahan citra. Kategori kotak terisi adalah kotak yang memuat objek berwarna hitam, sedangkan kotak yang tidak terisi adalah yang memuat objek berwarna putih atau yang memiliki derajat keabuan nol.



Gambar 2.3 Citra biner hasil proses pengolahan citra

2.4 Pengolahan Citra Digital

Secara harfiah, citra diartikan sebagai gambar pada bidang dwimatra atau bidang dua dimensi. Citra digital mengandung beberapa elemen dasar, antara lain kecerahan (*brightness*), kontras (*contrast*), kontur (*contour*), dan warna (*colour*). Kecerahan merupakan kata lain untuk intensitas cahaya. Kontras menyatakan sebaran terang dan gelap dalam sebuah citra. Kontur adalah keadaan yang ditimbulkan oleh perubahan intensitas pada *pixel-pixel* yang bertetangga. Sedangkan warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang.

Suatu citra memiliki histogram, histogram tersebut berguna untuk mengetahui informasi penting yang terdapat di dalam citra. Histogram citra adalah grafik yang menggambarkan nilai-nilai intensitas *pixel* atau bagian tertentu di dalam citra. Histogram dapat memberikan informasi mengenai kecerahan dan kontras dari sebuah citra. Cara mencari histogram dalam suatu citra adalah perbandingan jumlah *pixel* yang memiliki derajat keabuan tertentu dengan jumlah seluruh *pixel* di dalam citra. Kemudian nilai yang didapatkan diplotkan sehingga membentuk diagram batang. Jika suatu citra memiliki derajat keabuan L, maka histogram yang dicari adalah dari nilai derajat keabuan 0 sampai $L-1$. Secara matematis histogram citra dihitung dengan persamaan (2.3), yaitu

$$h_i = \frac{n_i}{n}, i = 0, 1, \dots, L-1 \qquad (2.3)$$

dimana n_i merupakan jumlah *pixel* yang memiliki derajat keabuan i, dan n adalah jumlah seluruh *pixel* di dalam citra (Munir, 2002).

Misalkan suatu citra memiliki 16 derajat keabuan, dengan ukuran matriks 8×8 yang dinyatakan dengan

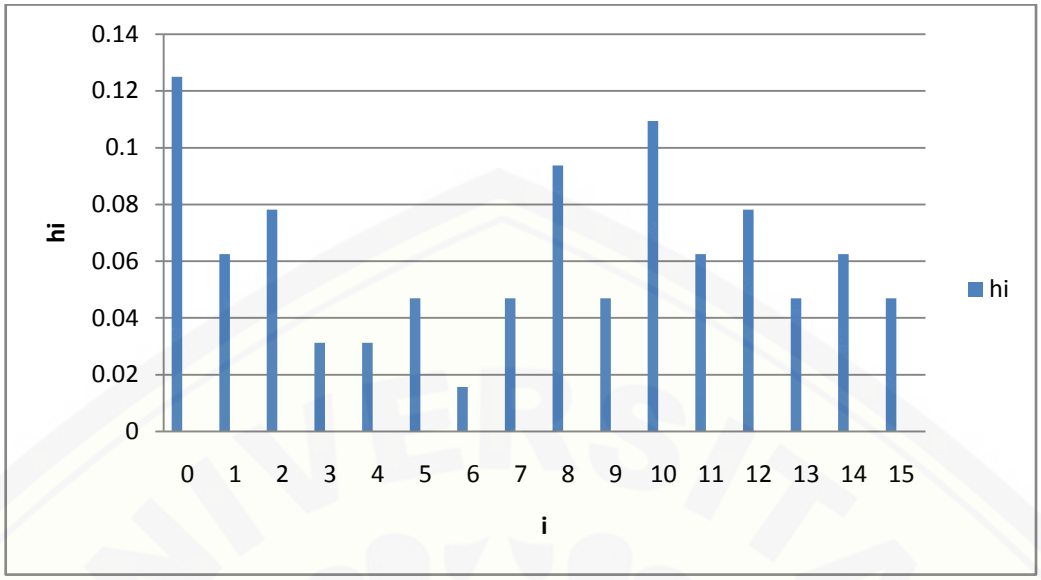
$$\begin{bmatrix} 3 & 7 & 7 & 8 & 10 & 12 & 14 & 10 \\ 2 & 0 & 0 & 0 & 1 & 8 & 15 & 15 \\ 14 & 6 & 5 & 9 & 8 & 10 & 9 & 12 \\ 12 & 12 & 11 & 8 & 8 & 10 & 11 & 1 \\ 0 & 2 & 3 & 4 & 5 & 13 & 10 & 14 \\ 4 & 5 & 0 & 0 & 1 & 0 & 2 & 2 \\ 15 & 13 & 11 & 10 & 9 & 9 & 8 & 7 \\ 2 & 1 & 0 & 10 & 11 & 14 & 13 & 12 \end{bmatrix}$$

Maka histogram yang dicari adalah derajat keabuan dari 0 sampai 15. Tabulasi perhitungan histogram disajikan pada Tabel 2.1.

Tabel 2.1 Tabulasi penghitungan histogram dari matriks 8×8

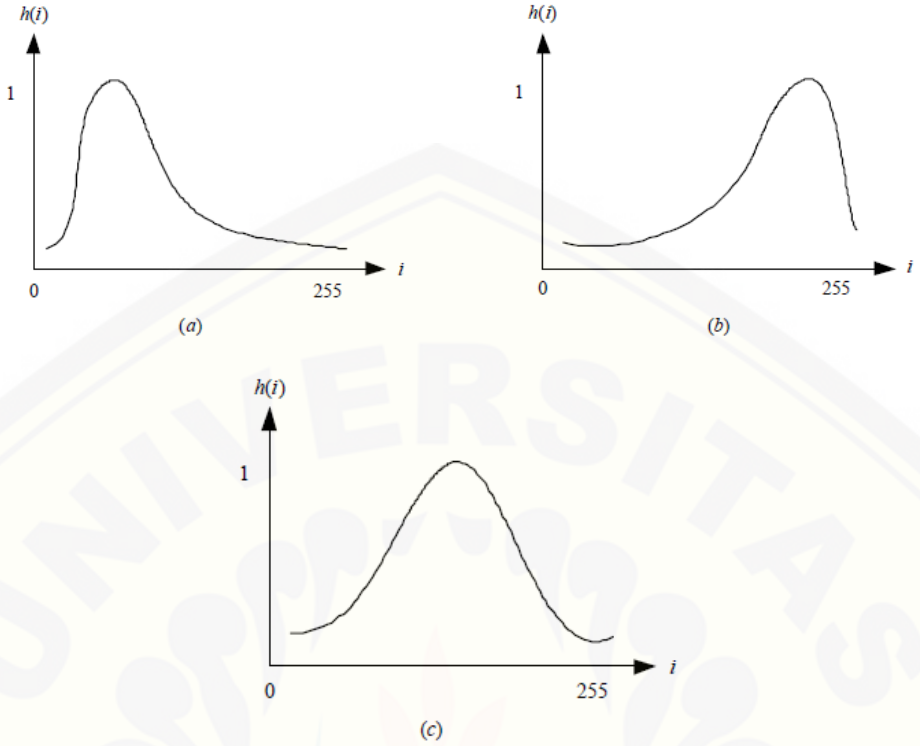
i	n_i	$h_i\ (n = 64)$
0	8	0,125
1	4	0,0625
2	5	0,078125
3	2	0,03125
4	2	0,03125
5	3	0,046875
6	1	0,015625
7	3	0,046875
8	6	0,09375
9	3	0,046875
10	7	0,109375
11	4	0,0625
12	5	0,078125
13	3	0,046875
14	4	0,0625
15	3	0,046875

Selanjutnya nilai-nilai tersebut diplotkan dalam suatu diagram batang nilai i terhadap nilai h_i. Sehingga histogram yang dihasilkan adalah sesuai dengan Gambar 2.4.



Gambar 2.4 Histogram citra

Histogram menunjukan tingkat kecerahan suatu citra dari nilai derajat keabuan yang diketahui. Semakin kecil derajat keabuan pada suatu *pixel*, semakin gelap tingkat derajat keabuannya dan begitu sebaliknya. Hal tersebut menunjukkan semakin histogram condong ke sebelah kiri, citra semakin gelap. Semakin condong ke kanan histogram yang dihasilkan, menandakan citra semakin terang. Citra dengan kecerahan yang baik adalah citra yang memiliki histogram di tengah-tengah. Gambar 2.4 berikut mencoba menjelaskan tingkat kecerahan suatu citra. Citra dengan tingkat kecerahan yang terlalu gelap, terlalu terang, dan citra yang memiliki tingkat kecerahan normal.



(a) citra gelap, (b) citra terang, (c) citra normal

Gambar 2.5 Perbandingan histogram citra (Sumber: Munir, 2002)

Sebuah citra, terkadang mengalami penurunan mutu (degradasi) seperti mengandung cacat atau derau (*noise*), warna terlalu kontras, kurang tajam, kabur (*blurring*), dan lain-lain. Citra yang demikian ini akan lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra menjadi berkurang. Oleh karena itu perlu adanya pengolahan citra, agar kualitas citra semakin baik dan menjadi lebih mudah untuk diinterpretasi (Munir, 2002).

Terdapat beberapa macam operasi dalam pengolahan citra, diantaranya histogram ekualisasi (HE), adaptif histogram ekualisasi (AHE), *cropping*, dan pengambangan (*thresholding*). Histogram ekualisasi adalah pengubahan histogram suatu citra menjadi seragam. Yaitu, setiap grafik batang pada histogram harus sama tinggi, atau dengan kata lain bahwa setiap level keabuan dalam citra harus memiliki frekuensi kemunculan yang sama. Histogram ekualisasi juga berguna untuk memperbaiki kontras pada citra (Sianipar, 2013).

Yang dimaksud dengan penyeragaman histogram atau pemerataan histogram adalah mengubah derajat keabuan suatu pixel r dengan derajat keabuan yang baru s dengan suatu fungsi transformasi T. Sehingga nilai s adalah $s = T(r)$. Nilai-nilai s diperoleh dari persamaan (2.4) berikut:

$$s_k = T(r_k) = \sum_{j=0}^{k} \frac{n_j}{n} = \sum_{j=0}^{k} P_r(r_j) \quad (2.4)$$

dengan $0 \leq r_k \leq 1$, $k = 0, 1, \ldots, L-1$ dan P_r adalah nilai derajat keabuan dibagi dengan total derajat keabuan dalam suatu citra (Munir, 2002).

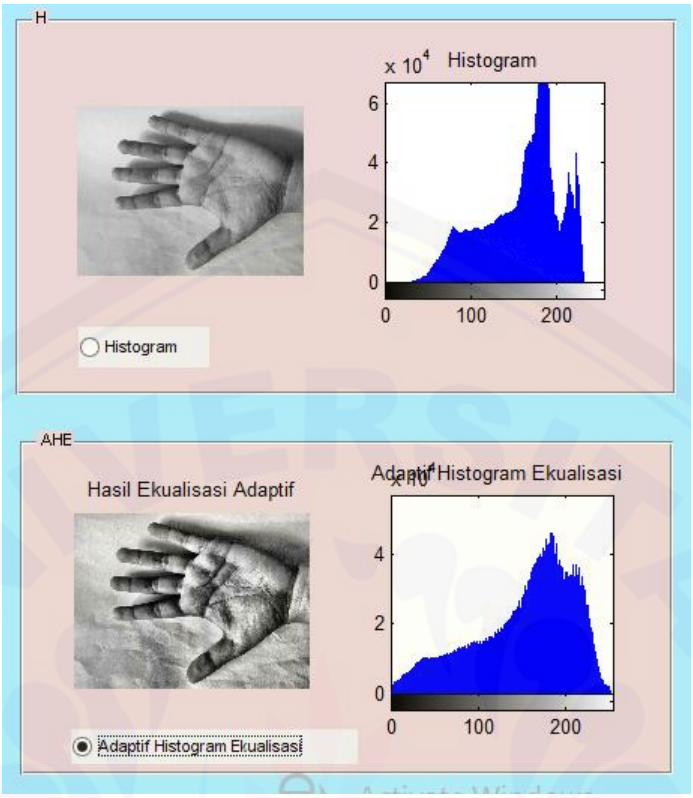
Adaptif histogram ekualisasi adalah teknik perbaikan kekontrasan citra dengan meningkatkan kontras lokal citra. Lokal citra didapatkan dengan membentuk beberapa *grid* simetris pada citra yang disebut dengan *region size*. Struktur regional citra dibagi menjadi tiga, yaitu bagian yang berada di sudut citra ditandai dengan *corner region* (CR), bagian tepi kecuali CR ditandai dengan *border region* (BR), dan bagian lainnya yang berada di tengah ditandai dengan *inner region* (IR).

Alasan dibedakannya struktur *region size* karena antara CR, BR, dan IR memiliki karakteristik ketetanggaan yang berbeda. Struktur *region size* ditunjukan dengan Gambar 2.6. Cara mendapatkan nilai *grey level* baru untuk tiap *region size* dilakukan dengan cara penghitungan sesuai persamaan (2.4). Perhitungan tersebut berlaku untuk setiap regional lokal (i, j) (Kanditami, 2014).

CR	BR	BR	CR
BR	IR	IR	BR
BR	IR	IR	BR
CR	BR	BR	CR

Gambar 2.6 Struktur *region size*

Untuk hasil proses AHE, ditunjukkan dalam Gambar 2.7. Gambar tersebut menunjukkan perbedaan citra awal dengan citra yang telah diolah menggunakan adaptif histogram ekualisasi.



Gambar 2.7 Citra hasil operasi AHE dan histogramnya.

Metode pengolahan citra yang sering dilakukan lainnya adalah *cropping* dan operasi pengambangan (*thresholding*). *Cropping* adalah pemotongan bagian tertentu dari citra menjadi matriks baru yang independen. *Thresholding* adalah operasi pengkonversian citra *grayscale* menjadi citra biner. Operasi *thresholding* mengelompokkan nilai derajat keabuan setiap *pixel* ke dalam dua kelas, hitam dan putih.

Pada proses pengambangan, setiap *pixel* di dalam citra dipetakan ke dua nilai 1 dan 0 dengan fungsi pengambangan berdasarkan persamaan (2.5).

$$f_B(i,j) = \begin{cases} 1, f_g(i,j) \leq T \\ 0, \text{lainnya} \end{cases} \quad (2.5)$$

Dalam hal ini, $f_g(i,j)$ adalah citra grayscale, $f_B(i,j)$ adalah citra biner, dan T adalah nilai ambang yang dispesifikan. Proses pengkonversian menjadi citra biner ini bertujuan untuk mengidentifikasi keberadaan objek yang direpresentasikan sebagai daerah (*region*) di dalam citra (Munir, 2002).

2.5 Koefisien Korelasi

Koefisien korelasi adalah nilai yang menunjukkan kuat atau tidaknya hubungan linier antara dua variabel. Tahapan ini digunakan untuk menentukan tingkat kesamaan (*similarity degree*) dan ketidaksamaan (*dissimilarity degree*). Tingkat kesamaan ditentukan dari suatu nilai. Dari nilai tersebut, dua vektor ciri yaitu vektor uji dari hasil ekstraksi ciri dan vektor acuan dapat dikatakan mirip atau tidak. Nilai tingkat kesamaan tersebut didapatkan dari perhitungan koefisian korelasi.

Koefisien korelasi nilainya berkisar antara -1 dan 1 $(-1 \leq S_{ij} \leq 1)$. Nilai yang mendekati -1 atau 1 menunjukkan hubungan yang kuat antara dua variabel yang dibandingkan. Sedangkan nilai yang mendekati 0 mengindikasikan lemahnya hubungan antara dua variabel tersebut. Tanda positif (+) dan negatif (-) menandakan arah hubungan antara kedua variabel. Jika nilai koefisien korelasi semakin besar atau bernilai positif, kedua variabel memiliki hubungan yang searah. Dengan kata lain, meningkatnya salah satu nilai variabel yang dibandingkan akan bersamaan dengan peningkatan nilai variabel yang lain. Menandakan tingkat kesamaan antara dua variabel semakin besar, data semakin mirip dan berlaku juga sebaliknya.

Perbandingan koefisien korelasi dinyatakan dengan persamaan (2.6) berikut (Mulyadi, 2013):

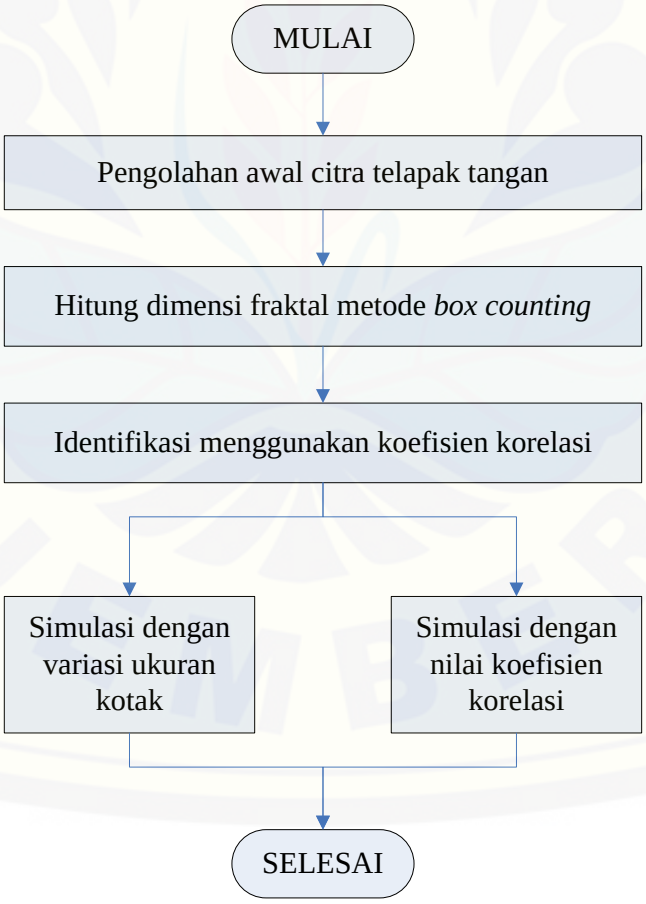
$$S_{ij} = \frac{\sum_{k=1}^{n}(X_{ik} - \bar{X}_i).(X_{jk} - \bar{X}_j)}{[\sum_{k=1}^{n}(X_{ik} - \bar{X})^2 . \sum_{k=1}^{n}(X_{jk} - \bar{X}_j)^2]^{\frac{1}{2}}} \quad (2.6)$$

dengan $\bar{X}_i = \frac{1}{n}\sum_{k=1}^{n} X_{ik}$ dan $\bar{X}_j = \frac{1}{n}\sum_{k=1}^{n} X_{jk}$.

BAB 3. METODE PENELITIAN

Pada bab ini akan dibahas mengenai langkah-langkah yang dilakukan untuk melakukan identifikasi terhadap telapak tangan menggunakan ekstraksi ciri berbasis dimensi fraktal. Dalam penelitian ini, *software* yang digunakan adalah matlab. Langkah-langkah yang dilakukan adalah proses awal pengolahan citra digital, ekstraksi ciri menggunakan dimensi fraktal, dan pencocokan nilai dimensi fraktal menggunakan koefisien korelasi.

Pada penelitian ini, dilakukan identifikasi telapak tangan dengan menggunakan beberapa variasi ukuran kotak, dan juga pengujian menggunakan beberapa nilai koefisien korelasi. Untuk lebih jelasnya, langkah-langkah tersebut digambarkan dalam diagram alir Gambar 3.1.



Gambar 3.1 Skema metodologi penelitian

3.1 Pengolahan Awal Citra Telapak Tangan

Proses pengolahan citra merupakan tahapan pertama dalam mengidentifikasi telapak tangan sebelum diekstraksi menggunakan dimensi fraktal. Tahapan-tahapan pada pengolahan citra antara lain:

a. Pembacaan berkas citra, yaitu dengan menginputkan citra telapak tangan yang akan diidentifikasi. Citra ini yang selanjutnya akan diproses pada pengolahan citra tahapan berikutnya
b. Pemotongan daerah pada citra yang berisi garis-garis telapak tangan dengan cara *cropping*
c. Mengubah citra hasil *cropping* bentuk *red-green-blue* (rgb) menjadi citra bentuk *grayscale*, dan menampilkan juga histogram yang dihasilkan
d. Melakukan proses adaptif histogram ekualisasi pada citra yang telah diubah dalam bentuk *grayscale*. Juga ditampilkan histogram hasil adaptif histogram ekualisasi
e. Menentukan nilai ambang (*threshold*) sebagai parameter untuk mengubah ke citra biner
f. Melakukan operasi *thresholding* untuk mengubah menjadi citra biner.

3.2 Ekstraksi Ciri Menggunakan Metode *Box Counting*

Setelah proses awal pengolahan citra, dilakukan proses ekstraksi ciri menggunakan dimensi fraktal. Ekstraksi ciri ini dilakukan untuk mencari informasi-informasi penting pada tekstur telapak tangan. Tahapan dari proses ekstraksi ciri menggunakan metode *box counting* adalah:

a. Menginputkan citra biner hasil dari proses awal pengolahan citra
b. Menentukan variasi ukuran kotak *s* yang digunakan, yaitu $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \cdots, \frac{1}{4096}$
c. Menghitung dimensi fraktal dari data acuan citra telapak tangan
d. Menghitung dimensi fraktal dari citra telapak tangan yang akan diuji.

3.3 Identifikasi Nilai Dimensi Fraktal Dengan Koefisien Korelasi

Proses identifikasi dilakukan untuk menentukan kepemilikan telapak tangan yang diuji. Proses yang dilakukan adalah dengan melakukan pencocokan vektor ciri dari citra telapak tangan yang diuji dengan citra telapak tangan acuan. Dalam hal ini, vektor ciri yang dimaksud adalah nilai-nilai dimensi fraktal yang telah dihasilkan dalam tahap sebelumnya. Berikut merupakan tahapan proses identifikasi nilai dimensi fraktal dengan koefisien korelasi:

a. Mengambil vektor ciri dari citra telapak tangan acuan
b. Mengambil vektor ciri dari citra telapak tangan yang akan diuji
c. Menghitung nilai-nilai dari vektor tersebut dengan koefisien korelasi pada persamaan (2.5)
d. Mengidentifikasi keanggotaan dari citra telapak tangan uji. Apakah citra uji merupakan milik dari orang yang terdapat dalam citra acuan atau tidak
e. Mengidentifikasi kepemilikan citra telapak tangan uji dari hasil perhitungan koefisien korelasi.

3.4 Pengubahan Variasi Ukuran Kotak

Pada metode *box counting* terdapat tahapan pembagian citra telapak tangan menjadi kotak-kotak kecil. Proses pengubahan variasi ukuran kotak dimaksudkan untuk mengetahui seberapa berpengaruh variasi ukuran kotak dapat menentukan hasil identifikasi. Sebelumnya pada subbab 3.2 dilakukan penentuan variasi ukuran kotak s adalah $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \dots, \frac{1}{4096}$. Pada tahapan ini dilakukan perubahan pada variasi ukuran kotak tersebut. Tahapan pada proses pengubahan variasi ukuran kotak adalah sebagai berikut:

a. Mengambil vektor ciri telapak tangan acuan
b. Mengambil vektor ciri telapak tangan uji
c. Menentukan variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \frac{1}{128}$
d. Menghilangkan vektor ciri telapak tangan acuan di luar variasi ukuran pada poin c
e. Menghilangkan vektor ciri telapak tangan uji di luar variasi ukuran pada poin c

f. Mengidentifikasi kepemilikan citra telapak tangan uji sesuai langkah pada subbab 3.3
g. Membandingkan hasil identifikasi kepemilikan telapak tangan uji dengan variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \ldots, \frac{1}{4096}$ dan $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \frac{1}{128}$.

3.5 Pengubahan Nilai Koefisien Korelasi

Proses pengubahan nilai koefisien korelasi dilakukan untuk mencari nilai yang paling sesuai. Jika nilai koefisien korelasi tidak sesuai, terlalu rendah atau terlalu tinggi nantinya akan mempengaruhi pada hasil identifikasi. Tahapan proses pengubahan nilai koefisien korelasi adalah sebagai berikut:

a. Tetapkan nilai koefisien korelasi pertama adalah 0,85
b. Tetapkan nilai koefisien korelasi kedua adalah 0,95
c. Tetapkan nilai koefisien korelasi ketiga sesuai pada subbab 3.3 yaitu 0,99
d. Lakukan proses identifikasi kepemilikan telapak tangan seperti pada langkah subbab 3.3 dengan besar nilai koefisien korelasi sesuai poin a
e. Hitung persentase keberhasilan identifikasi
f. Lakukan proses identifikasi kepemilikan telapak tangan seperti pada langkah subbab 3.3 dengan besar nilai koefisien korelasi sesuai poin b
g. Hitung persentase keberhasilan identifikasi
h. Lakukan proses identifikasi kepemilikan telapak tangan seperti pada langkah subbab 3.3 dengan besar nilai koefisien korelasi sesuai poin c
i. Hitung persentase keberhasilan identifikasi
j. Bandingkan persentase hasil identifikasi antara ketiga nilai koefisien korelasi tersebut

BAB 4. HASIL DAN PEMBAHASAN

Pada bab ini dibahas mengenai hasil dan pembahasan dari proses identifikasi telapak tangan dengan penghitungan dimensi fraktal menggunakan metode *box counting*. Terlebih dahulu dipaparkan langkah-langkah identifikasi telapak tangan, kemudian tampilan program yang dihasilkan. Program yang digunakan adalah menggunakan bantuan software matlab.

4.1 Identifikasi Telapak Tangan

Dalam kajian skripsi ini, yang akan dikerjakan adalah mengidentifikasi telapak tangan dengan mengambil nilai dimensi fraktal telapak tangan tersebut. Langkah-langkah yang dilakukan sesuai pada metode penelitian adalah dengan mengolah citra telapak tangan secara digital, kemudian dari hasil olah citra tersebut dihitung dimensi fraktalnya menggunakan metode *box counting*. Selanjutnya mencocokkan nilai dimensi fraktal telapak tangan acuan dan telapak tangan yang akan diidentifikasi.

Pada pengolahan citra telapak tangan, terlebih dahulu diambil citra beberapa telapak tangan yang nantinya akan digunakan sebagai data acuan, yaitu data pembanding yang disimpan dalam database program untuk mengidentifikasi telapak tangan. Citra yang digunakan haruslah citra berwarna dan diambil dalam keadaan diam. Telapak tangan yang digunakan adalah telapak tangan yang normal, dan tidak terdapat gangguan dari luar seperti coretan, kotoran, dan lain-lain. Dalam pengambilan citra telapak tangan, sudut pengambilan gambar haruslah tegak lurus dari objek telapak tangan tersebut. Setelah pengambilan citra telapak tangan acuan selesai, dilakukan pengambilan gambar beberapa telapak tangan uji, yaitu telapak tangan yang akan diidentifikasi kepemilikannya. Telapak tangan uji diambil dari beberapa telapak tangan dari individu yang sama dengan telapak tangan acuan, serta beberapa diambil dari individu yang berbeda. Hal ini dilakukan untuk mengetahui seberapa akurat program yang dihasilkan nanti.

Dari dua puluh citra telapak tangan yang diujikan, lima belas diantaranya merupakan telapak tangan dari orang yang sama dengan telapak tangan acuan. Sedangkan lima telapak tangan yang lain berasal dari orang yang belum dimasukkan ke dalam data telapak tangan acuan. Gambar 4.1 adalah contoh beberapa gambar telapak tangan yang telah dihasilkan.

Gambar 4.1 Citra telapak tangan acuan dan telapak tangan uji

Setelah proses pengambilan citra selesai, kemudian dilakukan beberapa proses pengolahan citra. Proses pengolahan citra yang dilakukan yaitu menginputkan citra telapak tangan acuan yang sudah didapatkan sebelumnya ke dalam program. Setelah masuk ke program, dilakukan proses *cropping* terlebih dahulu untuk mengambil bagian yang penting dan membuang bagian dari citra yang tidak diperlukan. Karena yang diekstraksi cirinya pada tiap-tiap telapak tangan adalah garis-garis pada telapak tangan, maka di *crop* bagian telapak tangan yang mengandung garis-garis telapak tangan saja. Gambar 4.2 berikut merupakan hasil dari citra telapak tangan yang telah melalui proses *cropping*.

Gambar 4.2 Citra telapak tangan hasil *cropping*

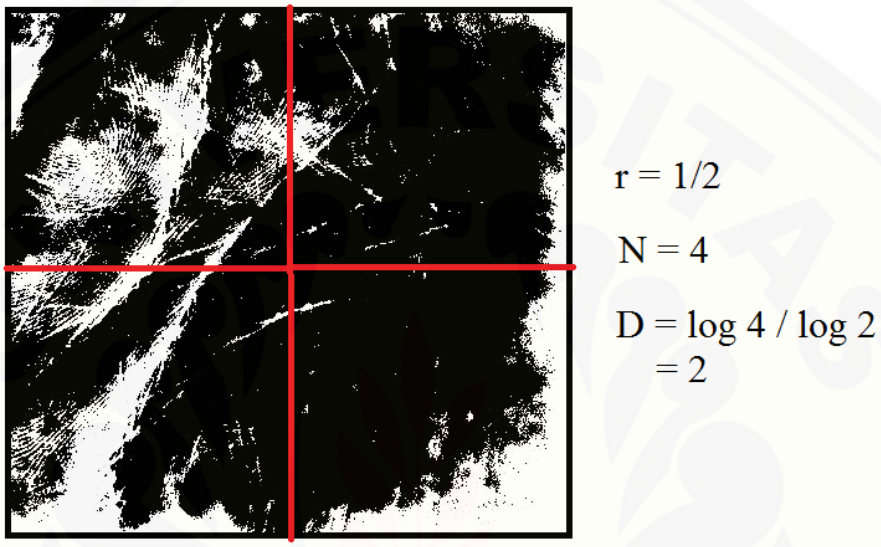
Proses selanjutnya, citra hasil *cropping* disimpan terlebih dahulu dan kemudian diolah menggunakan beberapa tahapan olah citra. Tahapan pengolahan citra yang dilakukan adalah mengubah citra berwarna (bentuk rgb) menjadi bentuk *grayscale*. Kemudian dilakukan proses adaptif histogram ekualisasi (AHE) untuk memperbaiki kontrasnya. Yang terakhir adalah mengubah menjadi citra biner dengan operasi *thresholding* dengan nilai ambang yang telah ditetapkan sebesar 128. Diambil nilai tersebut karena nilai derajat keabuan adalah 0 sampai 255. Jika terlalu dekat ke nilai 0, maka citra terlalu gelap atau terlalu hitam. Sebaliknya jika nilai ambang yang diambil terlalu dekat ke nilai 255, maka citra yang dihasilkan terlalu terang. Jadi diambil nilai 128 yang berada di tengah-tengah antara 0 dan 255. Gambar 4.3 adalah gambar proses pengubahan menjadi citra *grayscale*, proses AHE, dan *thresholding*.

(a) citra *grayscale*, (b) citra hasil AHE, (c) citra hasil *threshoding*

Gambar 4.3 Rangkaian proses olah citra

Setelah didapatkan citra biner dari hasil proses *thresholding*, kemudian dilakukan proses penghitungan nilai dimensi fraktal menggunakan metode *box counting*. Pembagian citra telapak tangan biner menjadi kotak-kotak kecil ditentukan dengan ukuran $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \dots, \frac{1}{4096}$ yang dilambangkan dengan r. Setelah itu dihitung berapa jumlah kotak terisi berdasarkan ukuran kotak-kotak kecil masing-masing yang dilambangkan dengan N. Kategori kotak terisi sesuai dengan yang telah dipaparkan sebelumnya pada bab 2, yaitu yang nilai derajat keabuannya tidak nol.

Misalkan pada salah satu citra telapak tangan yang telah diubah menjadi citra biner, ilustrasi penghitungan manual menggunakan metode *box counting* sesuai Gambar 4.4. Pada Gambar 4.4, citra biner dengan menggunakan ukuran kotak (r) adalah $\frac{1}{2}$, banyaknya kotak terisi adalah 4. Sehingga nilai dimensi yang didapatkan adalah $D = \frac{\log 4}{\log \frac{1}{1/2}} = \frac{\log 4}{\log 2} = 2.$



Gambar 4.4 Ilustrasi penghitungan manual metode *box counting*

Dari semua data telapak tangan yang didapatkan dihitung dimensi fraktalnya. Sehingga didapatkan nilai-nilai dimensi fraktal yang bervariasi dari tiap-tiap telapak tangan. Nilai-nilai tersebutlah yang nantinya akan digunakan sebagai vektor ciri pada tahap pencocokan menggunakan koefisien korelasi.

Pada tahap pencocokan menggunakan koefisien korelasi, nilai-nilai dimensi fraktal telapak tangan uji dicocokan satu persatu dengan nilai-nilai dimensi fraktal telapak tangan acuan menggunakan rumus pada persamaan 2.5. Dalam hal ini, terdapat dua macam proses identifikasi. Identifikasi pertama untuk mengetahui apakah data telapak tangan yang diujikan merupakan telapak tangan milik orang yang terdapat pada data telapak tangan acuan. Sedangkan identifikasi kedua, adalah untuk mengenali kepemilikan citra telapak tangan yang diujikan.

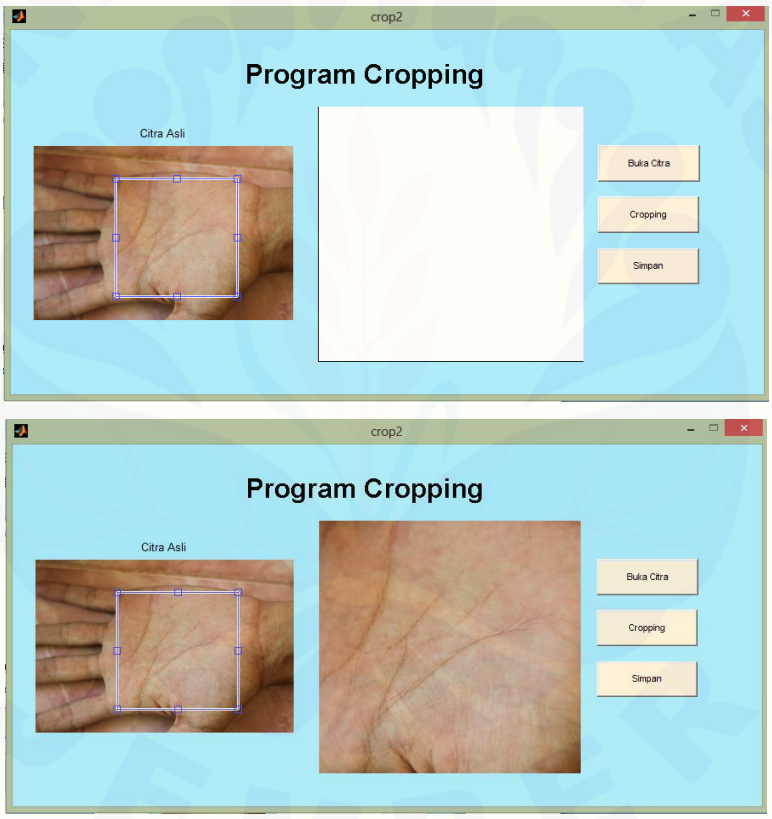
Pada proses identifikasi pertama, hasil identifikasi dikategorikan menjadi dua macam. Kategori yang dimaksud antara lain data ada dan data tidak ada. Kategori data ada adalah data telapak tangan uji merupakan data milik orang yang ada dalam telapak tangan acuan, sedangkan data tidak ada merupakan kebalikannya. Pengkategorian tersebut berdasarkan nilai hasil koefisien korelasi. Jika nilai koefisien korelasi di atas 0,99, maka data ada. Sedangkan jika nilai koefisien korelasi di bawah 0,99, maka data tidak ada.

4.2 Tampilan Program

Program identifikasi telapak tangan memiliki beberapa menu yang dapat dipanggil pada tampilan awal program. Menu tersebut antara lain *cropping* citra, olah citra, metode *box counting*, dan koef korelasi. Jika ingin memanggil fungsi tersebut, arahkan kursor ke salah satu menu dan klik menu tersebut. Gambar 4.5 adalah gambar tampilan awal program identifikasi telapak tangan.

Gambar 4.5 Tampilan awal program

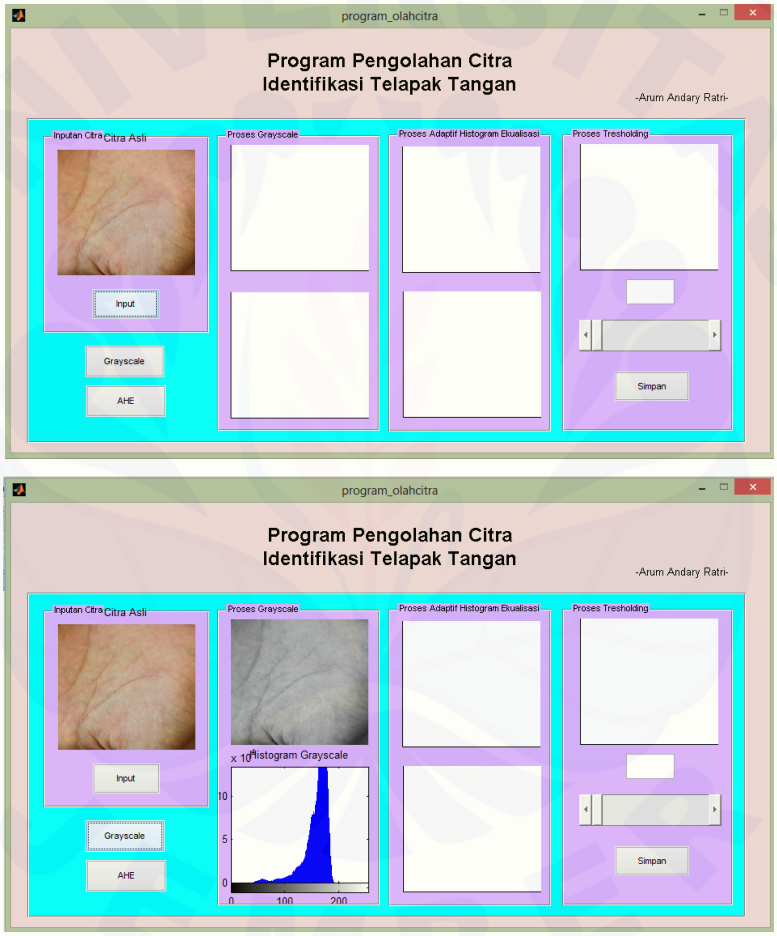
Menu *cropping* citra berfungsi untuk memotong citra inputan asli, kemudian diambil bagian penting dari telapak tangan. Cara bekerja dengan program *cropping* adalah dengan klik tombol buka citra pada program, maka citra telapak tangan akan tampil pada salah satu axes. Kemudian klik tombol *cropping* untuk memotong citra tersebut. Arahkan kursor pada axes yang telah menampilkan citra telapak tangan. Tentukan daerah yang akan dipotong menggunakan kursor, kemudian klik dua kali daerah tersebut. Citra hasil *cropping* akan tampil pada axes2. Selanjutnya simpan citra hasil *cropping* dengan cara mengklik tombol simpan. Gambar 4.6 adalah gambar menu *cropping* dan proses menjalankannya.



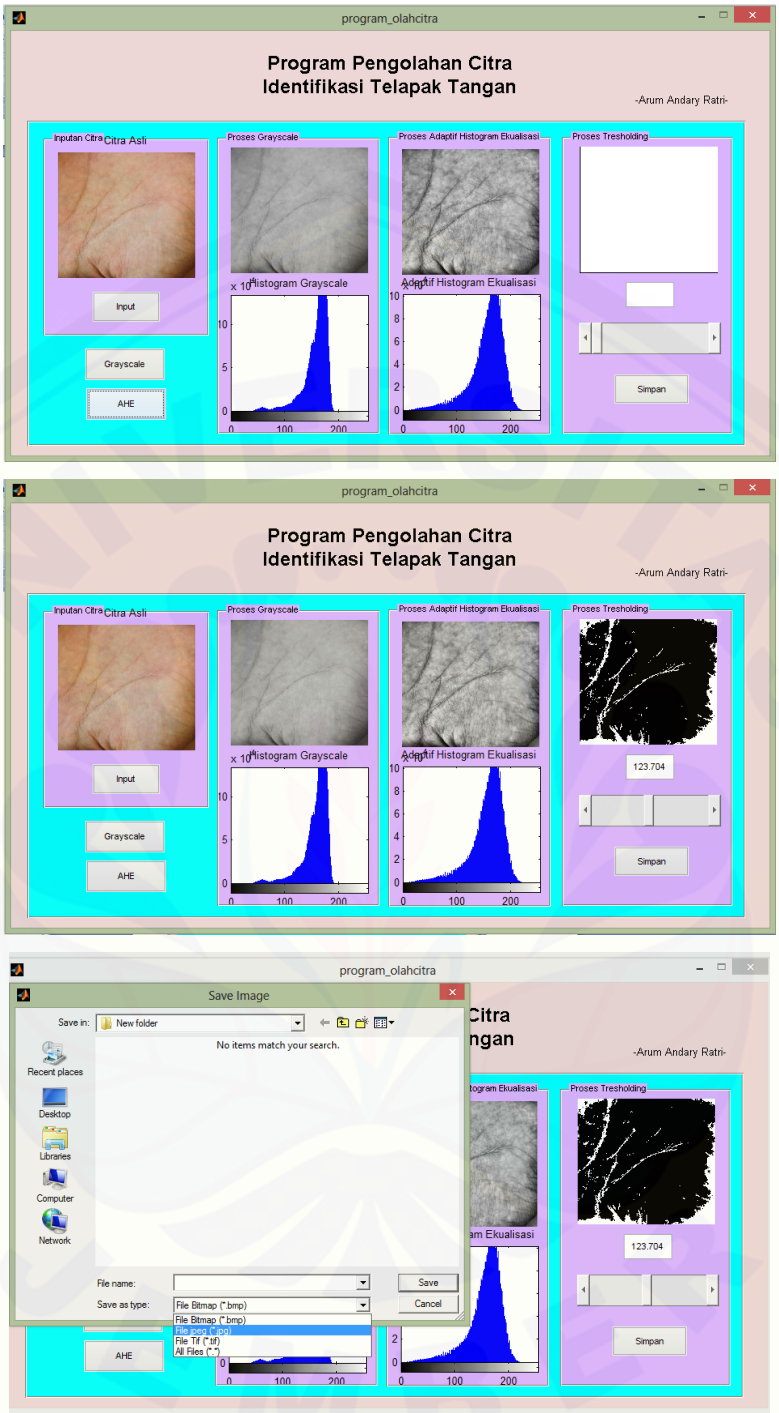
Gambar 4.6 Tampilan proses *cropping*

Menu olah citra berisi beberapa tahapan olah citra, yaitu pengubahan citra rgb menjadi citra *grayscale*, proses adaptif histogram ekualisasi, dan proses *thresholding*. Cara kerja menu tersebut adalah dengan terlebih dahulu

menginputkan citra telapak tangan hasil *cropping* ke dalam program ini. Kemudian ubah ke dalam bentuk *grayscale* dengan cara mengklik tombol *grayscale*. Selanjutnya untuk memperjelas kontras citra *grayscale* tersebut, klik tombol AHE agar diolah menggunakan proses AHE oleh program (contoh hitungan ada dalam lampiran G). Selanjutnya ubah citra hasil operasi AHE menjadi citra biner dengan cara meggeser-geser *slider*, dan tetapkan nilai ambang menjadi 123. Kemudian simpan citra yang telah diolah tersebut. Gambar 4.7 dan Gambar 4.8 mencoba mengilustrasikan tahapan olah citra pada program.

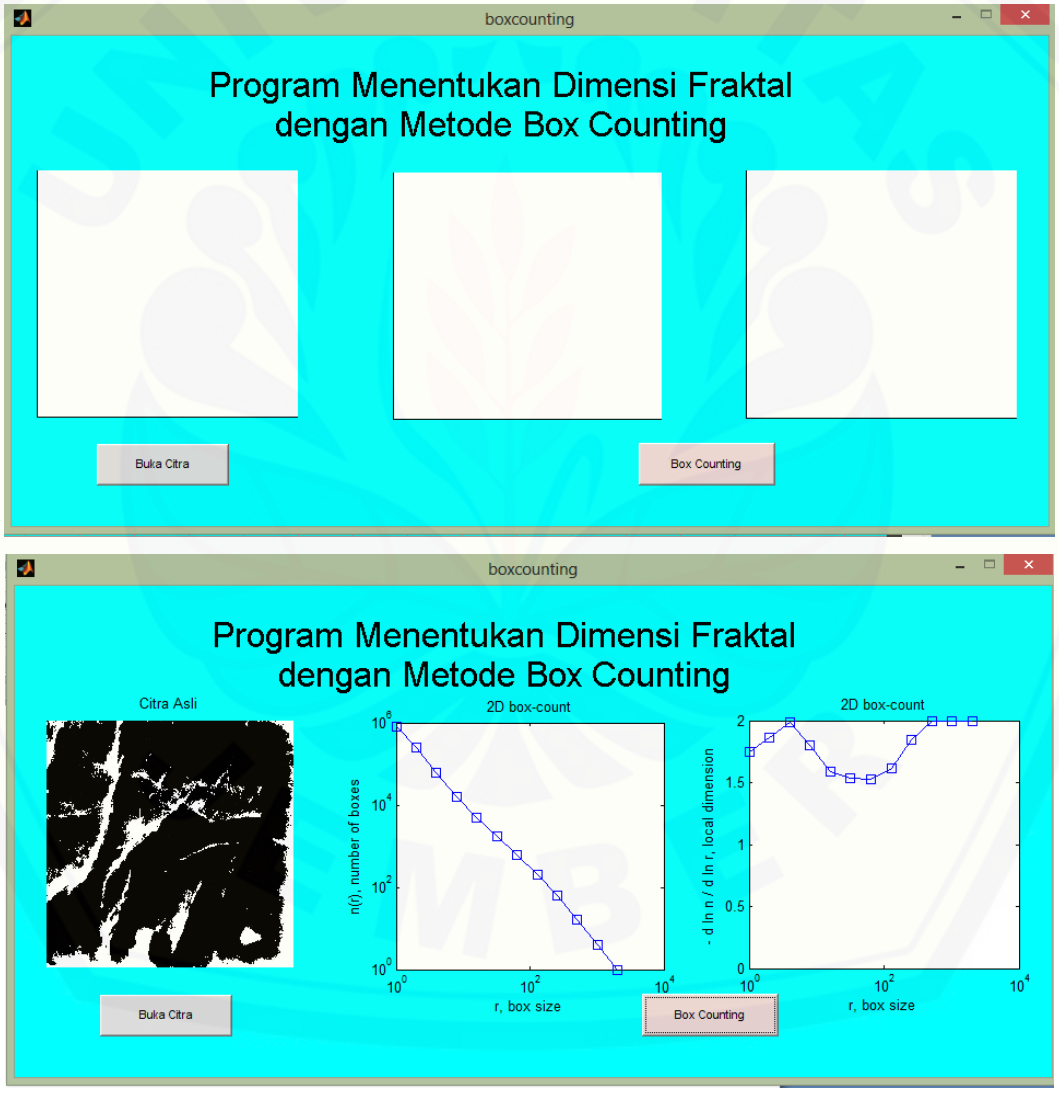


Gambar 4.7 Tampilan proses olah citra untuk input dan *grayscale*



Gambar 4.8 Tahapan proses AHE, *thresholding* dan penyimpanan citra

Selanjutnya adalah langkah penghitungan dimensi fraktal menggunakan metode *box counting*. Caranya adalah dengan menginputkan citra biner hasil proses olah citra ke dalam program *box counting*. Setelah itu klik tombol *box counting* untuk mencari ukuran kotak-kotak pembagi citra, jumlah kotak terisi, dan dimensi yang dihasilkan. Dalam program akan tampil dua kurva, kurva pertama menunjukkan hubungan antara ukuran kotak dan jumlah kotak, sedangkan kurva kedua menunjukkan plot antara ukuran kotak dan nilai dimensi fraktal. Tampilan program mengenai proses metode *box counting* disajikan dalam Gambar 4.9.



Gambar 4.9 Program metode *box counting*

Proses terakhir yaitu pencocokan nilai dimensi fraktal menggunakan koefisien korelasi. Program pencocokan koefisien korelasi berisi tempat untuk matriks inputan citra telapak tangan uji dan hasil identifikasi disajikan dalam bentuk nama dan juga foto wajah orang yang diidentifikasi tersebut. Tersedia dua axes untuk menampilkan citra telapak tangan uji dan juga citra wajah untuk menyatakan kepemilikan dari citra uji tersebut. Terdapat tiga tombol, yaitu identifikasi 1 untuk proses identifikasi yang pertama, identifikasi 2 untuk proses identifikasi yang kedua, dan reset untuk menyajikan bentuk awal program hasil identifikasi telapak tangan. Gambar 4.10 merupakan tampilan dari program identifikasi menggunakan koefisien korelasi dengan hasil proses identifikasi 1. Sedangkan Gambar 4.11 merupakan tampilan dari program pencocokan menggunakan koefisien korelasi dengan hasil identifikasi berupa foto dan nama orang yang berhasil diidentifikasi.

Gambar 4.10 Tampilan program hasil identifikasi pertama

Gambar 4.11 Tampilan program hasil identifikasi kedua

4.3 Hasil Identifikasi Telapak Tangan

Pada subbab ini akan dipaparkan hasil identifikasi telapak tangan pada dua puluh citra telapak tangan uji. Hasil identifikasi yang didapatkan pada program terbagi atas tiga kategori. Kategori pertama dilambangkan dengan BA, huruf B menyatakan benar dan A menyatakan ada. Jadi BA menyatakan program dapat mengidentifikasi dengan benar dan individu yang diklaim sesuai dengan data acuan. Kategori kedua adalah BT, yaitu B benar dan T tidak ada. Berarti program dapat mengidentifikasi dengan benar dan data tidak sesuai dengan citra acuan. Sedangkan kategori yang ketiga dilambangkan dengan S, yang mana kategori tersebut mewakili untuk identifikasi yang dilakukan salah.

Hasil yang didapatkan pada identifikasi telapak tangan adalah sebagai berikut. Dari dua puluh citra uji, yang berhasil diidentifikasi dengan benar adalah tujuh belas citra telapak tangan, dimana tiga belas diantaranya masuk pada kategori BA dan empat lainnya masuk di kategori BT. Sedangkan untuk tiga citra

telapak tangan uji sisanya diidentifikasi dengan salah. Persentase keberhasilan yang dihasilkan adalah 85%, yang didapatkan dari perhitungan $\frac{17}{20} \times 100\% = 85\%$. Tabel 4.1 merupakan hasil identifikasi program dari dua puluh citra telapak tangan uji. Dalam Tabel 4.1 tersebut terlihat telapak tangan mana yang berhasil diidentifikasi dengan benar, dan telapak tangan mana yang diidentifikasi dengan salah.

Tabel 4.1 Hasil identifikasi citra telapak tangan uji

No.	Nama	Hasil identifikasi
1	Telapak tangan uji 1	BA
2	Telapak tangan uji 2	BA
3	Telapak tangan uji 3	BT
4	Telapak tangan uji 4	S
5	Telapak tangan uji 5	BA
6	Telapak tangan uji 6	S
7	Telapak tangan uji 7	BA
8	Telapak tangan uji 8	BA
9	Telapak tangan uji 9	BA
10	Telapak tangan uji 10	BT
11	Telapak tangan uji 11	BA
12	Telapak tangan uji 12	BA
13	Telapak tangan uji 13	BA
14	Telapak tangan uji 14	BA
15	Telapak tangan uji 15	BA
16	Telapak tangan uji 16	BA
17	Telapak tangan uji 17	BT
18	Telapak tangan uji 18	S
19	Telapak tangan uji 19	BT
20	Telapak tangan uji 20	BA

4.4 Perbandingan Variasi Ukuran Kotak

Pada penghitungan nilai dimensi fraktal menggunakan metode *box counting*, ukuran kotak-kotak kecil turut menentukan hasil dari dimensi fraktal dan akan berpengaruh pada hasil identifikasi telapak tangan. Untuk membuktikan seberapa besar pengaruh ukuran kotak yang digunakan, berikut disajikan dua hasil identifikasi telapak tangan dengan variasi ukuran kotak yang berbeda. Percobaan pertama menggunakan variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \dots, \frac{1}{4096}$. Sedangkan percobaan kedua menggunakan variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \frac{1}{128}$.

Antara percobaan pertama dan percobaan kedua urutan data acuannya dibuat sama, nilai koefisien korelasinya pun disamakan, hanya berbeda pada variasi ukuran kotak yang digunakan. Hasil identifikasi pada percobaan pertama dengan variasi ukuran kotak $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \dots, \frac{1}{1024}$, persentase keberhasilan adalah 85%. Sedangkan untuk percobaan kedua yang variasi ukuran kotaknya adalah $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{32}, \frac{1}{64}, \frac{1}{128}$, persentase keberhasilannya menjadi 65%.

Nilai tersebut membuktikan bahwa variasi ukuran kotak yang ditentukan akan mempengaruhi pada hasil identifikasi telapak tangan. Pada percobaan pertama citra dibagi menjadi kotak-kotak kecil dengan ukuran sampai $\frac{1}{4096}$, dan percobaan kedua dibagi menjadi kotak-kotak kecil dengan ukuran terkecilnya adalah $\frac{1}{128}$. Hal tersebut menandakan semakin kecil ukuran kotak pembagi pada metode *box counting*, semakin tinggi tingkat ketelitiannya. Hasil identifikasi pada percobaan pertama dan kedua disajikan dalam Tabel 4.2 dan 4.3.

Tabel 4.2 Hasil identifikasi variasi kotak percobaan pertama

No.	Nama	Hasil identifikasi
1	Telapak tangan uji 1	BA
2	Telapak tangan uji 2	BA
3	Telapak tangan uji 3	BT
4	Telapak tangan uji 4	S
5	Telapak tangan uji 5	BA
6	Telapak tangan uji 6	S
7	Telapak tangan uji 7	BA
8	Telapak tangan uji 8	BA
9	Telapak tangan uji 9	BA
10	Telapak tangan uji 10	BT
11	Telapak tangan uji 11	BA
12	Telapak tangan uji 12	BA
13	Telapak tangan uji 13	BA
14	Telapak tangan uji 14	BA
15	Telapak tangan uji 15	BA
16	Telapak tangan uji 16	BA
17	Telapak tangan uji 17	BT
18	Telapak tangan uji 18	S
19	Telapak tangan uji 19	BT
20	Telapak tangan uji 20	BA

Tabel 4.3 Hasil identifikasi variasi kotak percobaan kedua

No.	Nama	Hasil identifikasi
1	Telapak tangan uji 1	BA
2	Telapak tangan uji 2	BA
3	Telapak tangan uji 3	S
4	Telapak tangan uji 4	S
5	Telapak tangan uji 5	BA
6	Telapak tangan uji 6	S
7	Telapak tangan uji 7	BA
8	Telapak tangan uji 8	BA
9	Telapak tangan uji 9	BA
10	Telapak tangan uji 10	BT
11	Telapak tangan uji 11	BA
12	Telapak tangan uji 12	BA
13	Telapak tangan uji 13	BA
14	Telapak tangan uji 14	BA
15	Telapak tangan uji 15	S
16	Telapak tangan uji 16	S
17	Telapak tangan uji 17	BT
18	Telapak tangan uji 18	S
19	Telapak tangan uji 19	BT
20	Telapak tangan uji 20	S

4.5 Perbandingan Nilai Koefisien Korelasi

Pada percobaan di subbab 4.3 dan 4.4, digunakan nilai koefisien korelasi sebesar 0,99. Nilai koefisien korelasi tersebut digunakan untuk mengidentifikasi apakah data masuk kategori data ada, atau data tidak ada. Tingginya pengambilan nilai untuk koefisien korelasi disebabkan tingkat kemiripan vektor ciri pada tiap-tiap telapak tangan sangat tinggi. Jika pengambilan nilai koefisien korelasi tidak tinggi, dikhawatirkan tingkat kesalahan identifikasi dalam program semakin tinggi. Maka diambil nilai koefisien korelasi yang tinggi pula.

Pada subbab ini, akan dibandingkan nilai beberapa koefisien korelasi yang berbeda untuk melihat seberapa berpengaruh perubahan nilai koefisien korelasi terhadap persentase keberhasilan identifikasi. Beberapa nilai koefisien korelasi yang dibandingkan adalah 0,85, 0,95 dan 0,99. Hasil identifikasi telapak tangan dengan tiga nilai koefisien korelasi yang berbeda disajikan dalam Tabel 4.4.

Pada Tabel 4.4 terlihat perbedaan hasil identifikasi telapak tangan dengan nilai koefisien korelasi yang berbeda. Pada saat nilai koefisien korelasi 0,85, dan nilai koefisien korelasi 0,95 persentase keberhasilan program adalah 65%. Sedangkan untuk nilai koefisien korelasi 0,99 tingkat keberhasilan identifikasi adalah 85%.

Pengambilan nilai koefisien korelasi ini berpengaruhpada hasil identifikasi yang data telapak tangan ujinya tidak termasuk dalam telapak tangan acuan. Karena dalam identifikasi telapak tangan, vektor ciri yang dihasilkan kemiripannya sangatlah tinggi. Maka dari ketiga nilai koefisien korelasi tersebut yang paling sesuai untuk menghasilkan persentase keberhasilan identifikasi yang paling tinggi adalah dengan nilai koefisien korelasi 0,99.

Tabel 4.4 Hasil identifikasi dengan tiga nilai koefisien korelasi yang berbeda

No	citra uji	0,85	0,95	0,99
1	Telapak tangan uji 1	BA	BA	BA
2	Telapak tangan uji 2	BA	BA	BA
3	Telapak tangan uji 3	S	S	BT
4	Telapak tangan uji 4	S	S	S
5	Telapak tangan uji 5	BA	BA	BA
6	Telapak tangan uji 6	S	S	BA
7	Telapak tangan uji 7	BA	BA	BA
8	Telapak tangan uji 8	BA	BA	BA
9	Telapak tangan uji 9	BA	BA	BA
10	Telapak tangan uji 10	S	S	BT
11	Telapak tangan uji 11	BA	BA	BA
12	Telapak tangan uji 12	BA	BA	BA
13	Telapak tangan uji 13	BA	BA	BA
14	Telapak tangan uji 14	BA	BA	BA
15	Telapak tangan uji 15	BA	BA	BA
16	Telapak tangan uji 16	BA	BA	BA
17	Telapak tangan uji 17	S	S	BT
18	Telapak tangan uji 18	S	S	S
19	Telapak tangan uji 19	S	S	BT
20	Telapak tangan uji 20	BA	BA	S

BAB 5 PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian yang dilakukan, dapat disimpulkan beberapa hal berikut:

a. Penerapan metode *box counting* untuk identifikasi telapak tangan adalah dengan melakukan pembacaan matriks berkas citra, dan dihitung dimensi fraktalnya menggunakan rumus metode *box counting* untuk dicari vektor cirinya lalu dicocokan dimensi fraktalnya menggunakan koefisien korelasi. Dalam penelitian ini, pencocokan dimensi fraktal tersebut menggunakan koefisien korelasi yang terbagi atas dua identifikasi, pertama identifikasi untuk mengetahui data ada jika nilai koefisien korelasi di atas 0,99 atau pun tidak ada jika koefisien korelasi di bawah 0,99. Sedangkan identifikasi kedua untuk mengetahui kepemilikan citra telapak tangan uji.
b. Dengan menggunakan data sampel dua puluh citra uji, lima belas diantaranya berasal dari data yang sama dengan telapak tangan acuan, sedangkan lima lainnya berasal dari data di luar telapak tangan acuan. Ukuran kotak maksimal yang digunakan adalah $\frac{1}{4096}$ dan nilai koefisien korelasi sebesar 0,99 yang memperoleh persentase keberhasilan hasil identifikasi 85%.

5.2 Saran

Identifikasi telapak tangan menggunakan metode *box counting* telah mencapai persentase yang cukup tinggi. Untuk penelitian selanjutnya, dapat diterapkan metode lain untuk lebih meningkatkan persentase keberhasilan identifikasi.

DAFTAR PUSTAKA

Evertsz, C. J. G., Peitgen H. O., dan Voss R. F. 1995. *Fractal Geometry and Analysis*. Singapore: World Scientific.

Helja, M., Nurhasanah, dan Sampurno, J. 2013. Analisis Fraktal Citra Mammogram Berbasis Tekstur Sebagai Pendukung Diagnosis Kanker Payudara. *POSITRON*. ISSN: 2301-4970. Vol. **3** (2): 35-38.

Kanditami, F., Deni S., dan Achmad R. 2014. Analisis Contrast Limited Adaptive Histogram Equalization (CLAHE) dan region Growing dalam Deteksi Gejala Kanker Payudara pada Citra Mammogram. Jurnal Elektro. Vol. **7** (1) : 15-28.

Mandelbrot, B. B. 1983. *The Fractal Geometry of Nature*. New York: W.H Freeman and Company.

Mulyadi, M. I., Isnanto, R. R., dan Hidayatno, A. 2013. Sistem Identifikasi Telapak Tangan Menggunakan Ekstraksi Ciri Berbasis Dimensi Fraktal. *TRANSIENT*. ISSN 2302-9927. Vol. **2** (3).

Munir, R. 2002. *Diktat Kuliah Pengolahan Citra, Edisi Kedua*. Bandung: Departemen Teknik Informatika ITB.

Putra, K. G. D. 2009. Sistem Verivikasi Biometrika Telapak Tangan dengan Metode Dimensi Fraktal dan Lacunarity. *Teknologi Elektro*. Vol. **8** (2) : *1-6.*

Sankar, D. & Thomas T. 2010. "Fractal Feature based on Differential Box Counting Method for the Categorization of Digital Mammograms". *International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM).* ISSN: 2150-7988. Vol. **2** : *011-019.*

Sianipar, R. H., Mangiri, H. S., dan Wiryajati, I. K. 2013. *Matlab Untuk Pemrosesan Citra Digital*. Bandung: Informatika Bandung.

Subiantoro, N. 2005. *Penentuan Dimensi Objek Fraktal dengan Metode Box Counting*. Skripsi. Jember. Jurusan Matematika Fakultas MIPA Universitas Jember.

Lampiran A. Data telapak tangan acuan

No.	Nama	Citra acuan 1	Citra acuan 2
1	Telapak tangan acuan 1		
2	Telapak tangan acuan 2		
3	Telapak tangan acuan 3		
4	Telapak tangan acuan 4		
5	Telapak tangan acuan 5		
6	Telapak tangan acuan 6		

7	Telapak tangan acuan 7		
8	Telapak tangan acuan 8		
9	Telapak tangan acuan 9		
10	Telapak tangan acuan 10		
11	Telapak tangan acuan 11		
12	Telapak tangan acuan 12		
13	Telapak tangan acuan 13		

14	Telapak tangan acuan 14		
15	Telapak tangan acuan 15		
16	Telapak tangan acuan 16		
17	Telapak tangan acuan 17		
18	Telapak tangan acuan 18		
19	Telapak tangan acuan 19		
20	Telapak tangan acuan 20		

Lampiran B. Data telapak tangan uji

No.	Nama	Citra uji
1	Telapak tangan uji 1	
2	Telapak tangan uji 2	
3	Telapak tangan uji 3	
4	Telapak tangan uji 4	
5	Telapak tangan uji 5	
6	Telapak tangan uji 6	
7	Telapak tangan uji 7	

8	Telapak tangan uji 8	
9	Telapak tangan uji 9	
10	Telapak tangan uji 10	
11	Telapak tangan uji 11	
12	Telapak tangan uji 12	
13	Telapak tangan uji 13	
14	Telapak tangan uji 14	

15	Telapak tangan uji 15	
16	Telapak tangan uji 16	
17	Telapak tangan uji 17	
18	Telapak tangan uji 18	
19	Telapak tangan uji 19	
20	Telapak tangan uji 20	

Lampiran C. Script program proses *cropping*

```
function varargout = crop2(varargin)
% CROP2 M-file for crop2.fig
%      CROP2, by itself, creates a new CROP2 or raises the
existing
%      singleton*.
%
%      H = CROP2 returns the handle to a new CROP2 or the handle
to
%      the existing singleton*.
%
%      CROP2('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in CROP2.M with the given input
arguments.
%
%      CROP2('Property','Value',...) creates a new CROP2 or raises
the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before crop2_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to crop2_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help crop2

% Last Modified by GUIDE v2.5 23-Mar-2015 16:40:48

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @crop2_OpeningFcn, ...
                   'gui_OutputFcn',  @crop2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```
% --- Executes just before crop2 is made visible.
function crop2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to crop2 (see VARARGIN)

% Choose default command line output for crop2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes crop2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = crop2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
[nama_file,nama_path]=uigetfile({'*.jpg';'*.bmp';'*.png';'*.tif'},
...
    'Buka Citra');
if~isequal(nama_file,0)
    handles.data1=imread(fullfile(nama_path,nama_file));
    guidata(hObject,handles);
    axes(handles.axes1)
    imshow(handles.data1)
    title('Citra Asli');
    [row,col,val]=size(handles.data1);
else
    return
end
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
h = imrect;
position = wait(h);
hasilCrop = imcrop(handles.data1,position);
axes(handles.axes2)
imshow(hasilCrop);
handles.data2=hasilCrop;
guidata(hObject,handles);
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider


% --- Executes during object creation, after setting all
properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double
```

```
% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
hasilCrop = handles.data2;
[name_file_save,path_save] = uiputfile(...
    {'*.bmp','File Bitmap (*.bmp)';...
    '*.jpg','File jpeg (*.jpg)';
    '*.tif','File Tif (*.tif)';
    '*.*','All Files (*.*)'},...
    'Save Image');
if ~isequal (name_file_save,0)
    imwrite(hasilCrop,fullfile(path_save,name_file_save));
else
    return
end
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

Lampiran D. Script program proses olah citra

```
function varargout = program_olahcitra(varargin)
% PROGRAM_OLAHCITRA M-file for program_olahcitra.fig
%      PROGRAM_OLAHCITRA, by itself, creates a new
PROGRAM_OLAHCITRA or raises the existing
%      singleton*.
%
%      H = PROGRAM_OLAHCITRA returns the handle to a new
PROGRAM_OLAHCITRA or the handle to
%      the existing singleton*.
%
%      PROGRAM_OLAHCITRA('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in PROGRAM_OLAHCITRA.M with the
given input arguments.
%
%      PROGRAM_OLAHCITRA('Property','Value',...) creates a new
PROGRAM_OLAHCITRA or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before program_olahcitra_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to
program_olahcitra_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
program_olahcitra

% Last Modified by GUIDE v2.5 23-Mar-2015 04:20:10

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn',
@program_olahcitra_OpeningFcn, ...
                   'gui_OutputFcn',  @program_olahcitra_OutputFcn,
...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
```

```
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before program_olahcitra is made visible.
function program_olahcitra_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to program_olahcitra (see
VARARGIN)

% Choose default command line output for program_olahcitra
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes program_olahcitra wait for user response (see
UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = program_olahcitra_OutputFcn(hObject,
eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
citra = handles.data1;
gray = rgb2gray(citra);
axes(handles.axes2)
imshow(gray);
handles.data2=gray;
guidata(hObject,handles);
axes(handles.axes3)
imhist(gray);
title('Histogram Grayscale');
% hObject    handle to pushbutton2 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
citra=rgb2gray(handles.data1);
adaptif=adapthisteq(citra);
axes(handles.axes4)
imshow(adaptif);
handles.data3=adaptif;
guidata(hObject,handles);
axes(handles.axes5)
imhist(adaptif);
title('Adaptif Histogram Ekualisasi ');
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
nilai_slider = get(handles.slider1,'value');
set(handles.edit1,'String',nilai_slider);
adaptif = handles.data3;
value = get(handles.slider1,'value');
thresh = (adaptif > 0) & (adaptif < value);
axes(handles.axes7);
imshow(thresh);
handles.data4=thresh;
guidata(hObject,handles);
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine
range of slider


% --- Executes during object creation, after setting all
properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double


% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
[nama_file,nama_path]=uigetfile({'*.jpg';'*.bmp';'*.png';'*.tif'},
...
    'Buka Citra');
if~isequal(nama_file,0)
    handles.data1=imread(fullfile(nama_path,nama_file));
    guidata(hObject,handles);
    axes(handles.axes1)
    imshow(handles.data1)
    title('Citra Asli');
    [row,col,val]=size(handles.data1);
else
    return
end
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
thresh = handles.data4;
[name_file_save,path_save] = uiputfile(...
    {'*.bmp','File Bitmap (*.bmp)';...
    '*.jpg','File jpeg (*.jpg)';
    '*.tif','File Tif (*.tif)';
    '*.*','All Files (*.*)'},...
    'Save Image');
if ~isequal (name_file_save,0)
    imwrite(thresh,fullfile(path_save,name_file_save));
else
    return
end
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

Lampiran E. Script program penghitungan metode *box counting*

```
function varargout = boxcounting(varargin)
% BOXCOUNTING M-file for boxcounting.fig
%      BOXCOUNTING, by itself, creates a new BOXCOUNTING or raises
the existing
%      singleton*.
%
%      H = BOXCOUNTING returns the handle to a new BOXCOUNTING or
the handle to
%      the existing singleton*.
%
%      BOXCOUNTING('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in BOXCOUNTING.M with the given
input arguments.
%
%      BOXCOUNTING('Property','Value',...) creates a new
BOXCOUNTING or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before boxcounting_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to boxcounting_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help boxcounting

% Last Modified by GUIDE v2.5 27-Mar-2015 11:09:33

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @boxcounting_OpeningFcn, ...
                   'gui_OutputFcn',  @boxcounting_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
```

```
% End initialization code - DO NOT EDIT


% --- Executes just before boxcounting is made visible.
function boxcounting_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to boxcounting (see VARARGIN)

% Choose default command line output for boxcounting
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes boxcounting wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = boxcounting_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
[n,r] = boxcount(handles.data1)
axes(handles.axes2)
boxcount(handles.data1)
axes(handles.axes3)
boxcount(handles.data1, 'slope')
df=-diff(log(n))./diff(log(r))
nilaiy = get(handles.pushbutton1,'value');



% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
```

```
[nama_file,nama_path]=uigetfile({'*.jpg';'*.bmp';'*.png';'*.tif'},
...
    'Buka Citra');
if~isequal(nama_file,0)
    handles.data1=imread(fullfile(nama_path,nama_file));
    guidata(hObject,handles);
    axes(handles.axes1)
    imshow(handles.data1)
    colormap gray
    axis square
    title('Citra Asli');
    [row,col,val]=size(handles.data1);
else
    return
end
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double



% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

Lampiran F. Script program pencocokan koefisien korelasi

```
function varargout = identifikasi2(varargin)
% IDENTIFIKASI2 M-file for identifikasi2.fig
%      IDENTIFIKASI2, by itself, creates a new IDENTIFIKASI2 or
raises the existing
%      singleton*.
%
%      H = IDENTIFIKASI2 returns the handle to a new IDENTIFIKASI2
or the handle to
%      the existing singleton*.
%
%      IDENTIFIKASI2('CALLBACK',hObject,eventData,handles,...)
calls the local
%      function named CALLBACK in IDENTIFIKASI2.M with the given
input arguments.
%
%      IDENTIFIKASI2('Property','Value',...) creates a new
IDENTIFIKASI2 or raises the
%      existing singleton*.  Starting from the left, property
value pairs are
%      applied to the GUI before identifikasi2_OpeningFcn gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to identifikasi2_OpeningFcn
via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help identifikasi2

% Last Modified by GUIDE v2.5 07-Jun-2015 09:32:56

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @identifikasi2_OpeningFcn,
...
                   'gui_OutputFcn',  @identifikasi2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
```

```
end
% End initialization code - DO NOT EDIT


% --- Executes just before identifikasi2 is made visible.
function identifikasi2_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to identifikasi2 (see
VARARGIN)

% Choose default command line output for identifikasi2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes identifikasi2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = identifikasi2_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
y=str2num(get(handles.edit1,'String'));

a=[1.714709831592495   1.985215884236913   1.988387716292948
1.545903380454021   1.511224115097718   1.388940617322489
1.396517411308046   1.541460862084569 1.751320887143276
1.736965594166206   1.169925001442313   2.000000000000002];
aa=[1.680804548613436   1.984498414423310   1.987111650133199
1.531585685052522   1.428058551072216   1.375990595147031
1.394003799406646   1.662965012722430 1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
b=[1.724121594682216   1.991856017075994   1.999491531347807
1.639266902903264   1.597687567042348   1.542457772206621
1.603699063302743   1.672425341971495 1.999999999999999
2.000000000000000   2.000000000000000   2.000000000000000];
```

```
bb=[1.743179482634955   1.987236819229915   1.980483741840276
1.655963024588020   1.636620554512218   1.507769354822956
1.446831581857664   1.675565049502063 1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
c=[1.703238900624159   1.992434563660651   1.989449547434532
1.707290239746542   1.731130885598217   1.750061251777890
1.792088165404294   1.814968106167479   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
cc=[1.744005309565677   1.992209167501118   1.994699954624086
1.720339814857377   1.730922259691220   1.771492787448191
1.808805595926329   1.809897116620057   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
d=[1.700429738823049   1.985118074831997   1.994833779749484
1.679521311657352   1.718298184133028   1.760812336120575
1.842943678828665   1.820021333893764   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
dd=[1.698142222171443   1.987648656884767   1.989573153231548
1.675534757283430   1.751051350469744   1.755934830732796
1.824428435416545   1.830074998557687   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
e=[1.820146896444131   1.992819525586395   1.993681594047846
1.847715307678173   1.903400774831065   1.925539851763811
1.950653946621794   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
ee=[1.828739472218434   1.993108070851051   1.996680258633330
1.858995695342933   1.915696111798566   1.949587787701524
1.953924129452026   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
f=[1.726906122679991   1.989662758305645   1.992030624891654
1.755820942579289   1.802547348130153   1.825252126584163
1.857835098880201   1.921774832694498   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
ff=[1.736691763223036   1.991142517988212   1.995482910761763
1.753456119202412   1.800548613682515   1.837457962076376
1.896164189015461   1.999999999999999   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
g=[1.732739631820818   1.990937820274016   1.994591423787620
1.734078405437883   1.732980246215027   1.762823762087233
1.815821014877446   1.890770930245242   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
gg=[1.737880960312639   1.991151051915334   1.995076668591741
1.740837455893761   1.745508626095503   1.778934749608061
1.888539569511457   1.901537361114311   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
h=[1.717677313804416   1.990066895308413   1.995159452152207
1.728880641521062   1.807104171202554   1.856446096390769
1.919657891682397   1.906890595608519   1.999999999999999
2.000000000000000   2.000000000000000 2.000000000000000];
hh=[1.735852520396205   1.990103107826112   1.991380302365319
1.767352369166357   1.847880235571838   1.871113036051689
1.880627910794917   1.824428435416545   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
i=[1.776704001803229   1.990492375818113   1.999799549536921
1.773930300468217   1.820234405226769   1.866800477731109
1.956015106717136   1.839959587489531   2.000000000000000
1.473931188332412   1.169925001442313   2.000000000000002];
```

```
ii=[1.789266160802318   1.991256913509331   1.996380159467360
1.804121461519321   1.860936382228666   1.899711943099106
1.924105150960719   1.851998837112445   2.000000000000000
1.473931188332412   1.169925001442313   2.000000000000002];
j=[1.782082217055803   1.991234323454376   1.995218423000029
1.750422389626583   1.795519401093348   1.780645078454589
1.860966306361995   1.974733241389570   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
jj=[1.795120645442478   1.991395198390710   1.995371113339872
1.774037759563524   1.792336032025250   1.809622122693223
1.851707680339209   1.898554736440454   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
k=[1.729533048752311   1.984635300459115   1.992463482348632
1.694722845282534   1.662036936112174   1.597074315824087
1.570996321557153   1.679740725473256   1.754887502163469
2.000000000000000   2.000000000000000 2.000000000000000];
kk=[1.709949604870203   1.984625222807601   1.990029459535633
1.649162958079002   1.629765806948468   1.595570600287138
1.630937929521542   1.696323609668861   1.754887502163469
2.000000000000000   2.000000000000000 2.000000000000000];
l=[1.899296387847347   1.995574282968009   1.993601506534549
1.940845389749728   1.963563259005707   1.935416293789809
1.908553644831163   1.835075679616054   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
ll=[1.902647505790608   1.995974607888407   1.996856872306022
1.939365059480141   1.960466778246123   1.968519252383328
1.954196310386875   1.917537839808027   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
m=[1.715715003010463   1.993344867604171   1.991666943116695
1.801885507643253   1.868298037690107   1.884445015550917
1.880079086258725   1.906890595608517   1.807354922057605
2.000000000000000   2.000000000000000 2.000000000000000];
mm=[1.711687455588567   1.991316463148920   1.996391502139241
1.819377450306816   1.884854489992118   1.920489706944341
1.980045393624355   1.893084796083488   1.614709844115208
2.000000000000000   2.000000000000000 2.000000000000000];
n=[1.736411691575842   1.988469750539394   1.994386775379061
1.712428602052544   1.725923799878829   1.790466288564510
1.862763617263861   1.906890595608519   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
nn=[1.701030335610312   1.989356097729621   1.999411649173818
1.699381290918727   1.743518088289007   1.800759976374798
1.923327485419193   2.000000000000000   1.999999999999999
2.000000000000000   2.000000000000000 2.000000000000000];
o=[1.778434530442391   1.989205208282247   1.991792971731509
1.755804687704438   1.791510526098849   1.827163403137786
1.842993748067265   1.904751684852183   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
oo=[1.789302031351435   1.990818918820490   1.992005650533764
1.760774319258891   1.797233299469315   1.865577121668803
1.856953382233836   1.913288366806669   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
p=[1.954602012142333   1.995597626490796   1.994661240673803
1.971724089505926   1.979778299295946   1.977828002823029
1.920883867688018   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
```

```
pp=[1.941515497965074   1.996681699472327   1.994587296918386
1.966329574089206   1.980985504754503   1.976979275448286
1.920883867688018   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
q=[1.721929949858536   1.987708155897873   1.993620204861778
1.677492902594451   1.701101454277252   1.742393467271303
1.868616391670703   1.778442230142366   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
qq=[1.709343344255338   1.989599659654585   1.994118913736704
1.673419558278687   1.739463849643773   1.792103063712340
1.897526378750206   1.888743248898260   1.999999999999999
2.000000000000000   2.000000000000000 2.000000000000000];
u=[1.795821334744344   1.992045178894024   1.999417884026666
1.797429034513890   1.829594864674772   1.855205481829366
1.921748554335632   1.839063781784943   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
uu=[1.812878646853647   1.990182668635562   1.999296037817012
1.813097199933766   1.846478521659413   1.849903972319609
1.963817663693354   1.843537258364250   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
s=[1.772237411517107   1.991579438720636   1.994063590129466
1.712315579597298   1.718463320170308   1.742645201066739
1.708537185839688   1.825056923791063   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
ss=[1.779393707568362   1.991267975501484   1.999713241101881
1.724156518259327   1.708917585736363   1.701512132426938
1.796548167101022   1.726239187573147   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
t=[1.719342597126365   1.991162467827313   1.995485097628615
1.714151286781141   1.798849064004123   1.863039461206056
1.883441928080349   1.917537839808026   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
tt=[1.727209179165093   1.989120225530005   1.991515223547192
1.715192860090790   1.782743864466708   1.867951924888537
1.897139211044802   1.994981925233376   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];


ra=corrcoef(y,a);
raa=corrcoef(y,aa);
rb=corrcoef(y,b);
rbb=corrcoef(y,bb);
rc=corrcoef(y,c);
rcc=corrcoef(y,cc);
rd=corrcoef(y,d);
rdd=corrcoef(y,dd);
re=corrcoef(y,e);
ree=corrcoef(y,ee);
rf=corrcoef(y,f);
rff=corrcoef(y,ff);
rg=corrcoef(y,g);
rgg=corrcoef(y,gg);
rh=corrcoef(y,h);
rhh=corrcoef(y,hh);
ri=corrcoef(y,i);
rii=corrcoef(y,ii);
```

```
rj=corrcoef(y,j);
rjj=corrcoef(y,jj);
rk=corrcoef(y,k);
rkk=corrcoef(y,kk);
rl=corrcoef(y,l);
rll=corrcoef(y,ll);
rm=corrcoef(y,m);
rmm=corrcoef(y,mm);
rn=corrcoef(y,n);
rnn=corrcoef(y,nn);
ro=corrcoef(y,o);
roo=corrcoef(y,oo);
rp=corrcoef(y,p);
rpp=corrcoef(y,pp);
rq=corrcoef(y,q);
rqq=corrcoef(y,qq);
ru=corrcoef(y,u);
ruu=corrcoef(y,uu);
rs=corrcoef(y,s);
rss=corrcoef(y,ss);
rt=corrcoef(y,t);
rtt=corrcoef(y,tt);

m1=max(ra,raa); m2=max(rb,rbb); m3=max(rc,rcc); m4=max(rd,rdd);
m5=max(re,ree);
m6=max(rf,rff); m7=max(rg,rgg); m8=max(rh,rhh); m9=max(ri,rii);
m10=max(rj,rjj);
m11=max(rk,rkk); m12=max(rl,rll); m13=max(rm,rmm);
m14=max(rn,rnn); m15=max(ro,roo);
m16=max(rp,rpp); m17=max(rq,rqq); m18=max(ru,ruu);
m19=max(rs,rss); m20=max(rt,rtt);
m21=max(m1,m2); m22=max(m3,m4); m23=max(m5,m6); m24=max(m7,m8);
m25=max(m9,m10);
m26=max(m11,m12); m27=max(m13,m14); m28=max(m15,m16);
m29=max(m17,m18); m30=max(m19,m20);
m31=max(m21,m22); m32=max(m23,m24); m33=max(m25,m26);
m34=max(m27,m28); m35=max(m29,m30);
m36=max(m31,m32); m37=max(m33,m34); m38=max(m36,m37);
m39=max(m35,m38);

if re >= m39
    axes(handles.axes2);
    imshow('uji5.jpg');
    axes(handles.axes1);
    imshow('phaki.jpg');
    set(handles.text2,'String','Nur Ahmad Baihaki');
elseif ree >= m39
    axes(handles.axes2);
    imshow('uji5.jpg');
    axes(handles.axes1);
    imshow('phaki.jpg');
    set(handles.text2,'String','Nur Ahmad Baihaki');
elseif rm >= m39
    axes(handles.axes2);
    imshow('uji13.jpg');
    axes(handles.axes1);
```

```
    imshow('prista.jpg');
    set(handles.text2,'String','Rista Nurjanah');
elseif rmm >= m39
    axes(handles.axes2);
    imshow('uji13.jpg');
    axes(handles.axes1);
    imshow('prista.jpg');
    set(handles.text2,'String','Rista Nurjanah');
elseif ro >= m39
    axes(handles.axes2);
    imshow('uji15.jpg');
    axes(handles.axes1);
    imshow('pria.jpg');
    set(handles.text2,'String','Komariyah');
elseif roo >= m39
    axes(handles.axes2);
    imshow('uji15.jpg');
    axes(handles.axes1);
    imshow('pria.jpg');
    set(handles.text2,'String','Komariyah');
elseif rf >= m39
    axes(handles.axes2);
    imshow('uji6.jpg');
    axes(handles.axes1);
    imshow('pemil.jpg');
    set(handles.text2,'String','Emil Gufron');
elseif rff >= m39
    axes(handles.axes2);
    imshow('uji6.jpg');
    axes(handles.axes1);
    imshow('pemil.jpg');
    set(handles.text2,'String','Emil Gufron');
elseif ru >= m39
    axes(handles.axes2);
    imshow('uji18.jpg');
    axes(handles.axes1);
    imshow('pcenul.jpg');
    set(handles.text2,'String','Khusnul Khowatim');
elseif ruu >= m39
    axes(handles.axes2);
    imshow('uji18.jpg');
    axes(handles.axes1);
    imshow('pcenul.jpg');
    set(handles.text2,'String','Khusnul Khowatim');
elseif rg >= m39
    axes(handles.axes2);
    imshow('uji7.jpg');
    axes(handles.axes1);
    imshow('paan.jpg');
    set(handles.text2,'String','Aan Ageng Wibowo');
elseif rgg >= m39
    axes(handles.axes2);
    imshow('uji7.jpg');
    axes(handles.axes1);
    imshow('paan.jpg');
    set(handles.text2,'String','Aan Ageng Wibowo');
```

```
elseif rt >= m39
    axes(handles.axes2);
    imshow('uji20.jpg');
    axes(handles.axes1);
    imshow('ponne.jpg');
    set(handles.text2,'String','Onne Hena Sugianto');
elseif rtt >= m39
    axes(handles.axes2);
    imshow('uji20.jpg');
    axes(handles.axes1);
    imshow('ponne.jpg');
    set(handles.text2,'String','Onne Hena Sugianto');
elseif rl >= m39
    axes(handles.axes2);
    imshow('uji12.jpg');
    axes(handles.axes1);
    imshow('pivan.jpg');
    set(handles.text2,'String','Ivan Syaikhul Hadi');
elseif rll >= m39
    axes(handles.axes2);
    imshow('uji12.jpg');
    axes(handles.axes1);
    imshow('pivan.jpg');
    set(handles.text2,'String','Ivan Syaikhul Hadi');
elseif rc >= m39
    axes(handles.axes2);
    imshow('uji3.jpg');
    axes(handles.axes1);
    imshow('phani.jpg');
    set(handles.text2,'String','Haniah Zakin');
elseif rcc >= m39
    axes(handles.axes2);
    imshow('uji3.jpg');
    axes(handles.axes1);
    imshow('phani.jpg');
    set(handles.text2,'String','Haniah Zakin');
elseif ri >= m39
    axes(handles.axes2);
    imshow('uji9.jpg');
    axes(handles.axes1);
    imshow('ppotty.jpg');
    set(handles.text2,'String','Achmad Baichaki Yunirahman');
elseif rii >= m39
    axes(handles.axes2);
    imshow('uji9.jpg');
    axes(handles.axes1);
    imshow('ppotty.jpg');
    set(handles.text2,'String','Achmad Baichaki Yunirahman');
elseif rp >= m39
    axes(handles.axes2);
    imshow('uji16.jpg');
    axes(handles.axes1);
    imshow('pputri.jpg');
    set(handles.text2,'String','Putri Rizky H. P.');
elseif rpp >= m39
    axes(handles.axes2);
```

```
    imshow('uji16.jpg');
    axes(handles.axes1);
    imshow('pputri.jpg');
    set(handles.text2,'String','Putri Rizky H. P.');
elseif rd >= m39
    axes(handles.axes2);
    imshow('uji4.jpg');
    axes(handles.axes1);
    imshow('pkikil.jpg');
    set(handles.text2,'String','Dyah Kiki Langit');
elseif rdd >= m39
    axes(handles.axes2);
    imshow('uji4.jpg');
    axes(handles.axes1);
    imshow('pkikil.jpg');
    set(handles.text2,'String','Dyah Kiki Langit');
elseif rq >= m39
    axes(handles.axes2);
    imshow('uji17.jpg');
    axes(handles.axes1);
    imshow('pkarin.jpg');
    set(handles.text2,'String','Karinda Rizky Aprilia');
elseif rqq >= m39
    axes(handles.axes2);
    imshow('uji17.jpg');
    axes(handles.axes1);
    imshow('pkarin.jpg');
    set(handles.text2,'String','Karinda Rizky Aprilia');
elseif rk >= m39
    axes(handles.axes2);
    imshow('uji11.jpg');
    axes(handles.axes1);
    imshow('pwafi.jpg');
    set(handles.text2,'String','Wafiatul Husna');
elseif rkk >= m39
    axes(handles.axes2);
    imshow('uji11.jpg');
    axes(handles.axes1);
    imshow('pwafi.jpg');
    set(handles.text2,'String','Wafiatul Husna');
elseif rj >= m39
    axes(handles.axes2);
    imshow('uji10.jpg');
    axes(handles.axes1);
    imshow('pciten.jpg');
    set(handles.text2,'String','Fatkurotin');
elseif rjj >= m39
    axes(handles.axes2);
    imshow('uji10.jpg');
    axes(handles.axes1);
    imshow('pciten.jpg');
    set(handles.text2,'String','Fatkurotin');
elseif rb >= m39
    axes(handles.axes2);
    imshow('uji2.jpg');
    axes(handles.axes1);
```

```
    imshow('pdewi.jpg');
    set(handles.text2,'String','Dewi Evianti');
elseif rbb >= m39
    axes(handles.axes2);
    imshow('uji2.jpg');
    axes(handles.axes1);
    imshow('pdewi.jpg');
    set(handles.text2,'String','Dewi Evianti');
elseif rn >= m39
    axes(handles.axes2);
    imshow('uji14.jpg');
    axes(handles.axes1);
    imshow('pfahma.jpg');
    set(handles.text2,'String','Fahma Hilviah');
elseif rnn >= m39
    axes(handles.axes2);
    imshow('uji14.jpg');
    axes(handles.axes1);
    imshow('pfahma.jpg');
    set(handles.text2,'String','Fahma Hilviah');
elseif rh >= m39
    axes(handles.axes2);
    imshow('uji8.jpg');
    axes(handles.axes1);
    imshow('plusi.jpg');
    set(handles.text2,'String','Lusi Wahyu');
elseif rhh >= m39
    axes(handles.axes2);
    imshow('uji8.jpg');
    axes(handles.axes1);
    imshow('plusi.jpg');
    set(handles.text2,'String','Lusi Wahyu');
elseif rs >= m39
    axes(handles.axes2);
    imshow('uji19.jpg');
    axes(handles.axes1);
    imshow('ptrian.jpg');
    set(handles.text2,'String','Trian Ghofarul M');
elseif rss >= m39
    axes(handles.axes2);
    imshow('uji19.jpg');
    axes(handles.axes1);
    imshow('ptrian.jpg');
    set(handles.text2,'String','Trian Ghofarul M');
elseif ra >= m39
    axes(handles.axes2);
    imshow('uji1.jpg');
    axes(handles.axes1);
    imshow('prafi.jpg');
    set(handles.text2,'String','Rafiulfath R. Riwansia');
elseif raa >= m39
    axes(handles.axes2);
    imshow('uji1.jpg');
    axes(handles.axes1);
    imshow('prafi.jpg');
    set(handles.text2,'String','Rafiulfath R. Riwansia');
```

```
end
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double


% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
set(handles.edit1,'String','');
set(handles.text2,'String','');
axes(handles.axes2);
imshow('reset.jpg');
axes(handles.axes1);
imshow('reset.jpg');
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
y=str2num(get(handles.edit1,'String'));
```

```
a=[1.714709831592495   1.985215884236913   1.988387716292948
1.545903380454021   1.511224115097718   1.388940617322489
1.396517411308046   1.541460862084569 1.751320887143276
1.736965594166206   1.169925001442313   2.000000000000002];
aa=[1.680804548613436   1.984498414423310   1.987111650133199
1.531585685052522   1.428058551072216   1.375990595147031
1.394003799406646   1.662965012722430 1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
b=[1.724121594682216   1.991856017075994   1.999491531347807
1.639266902903264   1.597687567042348   1.542457772206621
1.603699063302743   1.672425341971495 1.999999999999999
2.000000000000000   2.000000000000000   2.000000000000000];
bb=[1.743179482634955   1.987236819229915   1.980483741840276
1.655963024588020   1.636620554512218   1.507769354822956
1.446831581857664   1.675565049502063 1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
c=[1.703238900624159   1.992434563660651   1.989449547434532
1.707290239746542   1.731130885598217   1.750061251777890
1.792088165404294   1.814968106167479   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
cc=[1.744005309565677   1.992209167501118   1.994699954624086
1.720339814857377   1.730922259691220   1.771492787448191
1.808805595926329   1.809897116620057   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
d=[1.700429738823049   1.985118074831997   1.994833779749484
1.679521311657352   1.718298184133028   1.760812336120575
1.842943678828665   1.820021333893764   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
dd=[1.698142222171443   1.987648656884767   1.989573153231548
1.675534757283430   1.751051350469744   1.755934830732796
1.824428435416545   1.830074998557687   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
e=[1.820146896444131   1.992819525586395   1.993681594047846
1.847715307678173   1.903400774831065   1.925539851763811
1.950653946621794   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
ee=[1.828739472218434   1.993108070851051   1.996680258633330
1.858995695342933   1.915696111798566   1.949587787701524
1.953924129452026   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
f=[1.726906122679991   1.989662758305645   1.992030624891654
1.755820942579289   1.802547348130153   1.825252126584163
1.857835098880201   1.921774832694498   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
ff=[1.736691763223036   1.991142517988212   1.995482910761763
1.753456119202412   1.800548613682515   1.837457962076376
1.896164189015461   1.999999999999999   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
g=[1.732739631820818   1.990937820274016   1.994591423787620
1.734078405437883   1.732980246215027   1.762823762087233
1.815821014877446   1.890770930245242   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
gg=[1.737880960312639   1.991151051915334   1.995076668591741
1.740837455893761   1.745508626095503   1.778934749608061
1.888539569511457   1.901537361114311   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
```

```
h=[1.717677313804416   1.990066895308413   1.995159452152207
1.728880641521062   1.807104171202554   1.856446096390769
1.919657891682397   1.906890595608519   1.999999999999999
2.000000000000000   2.000000000000000 2.000000000000000];
hh=[1.735852520396205   1.990103107826112   1.991380302365319
1.767352369166357   1.847880235571838   1.871113036051689
1.880627910794917   1.824428435416545   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
i=[1.776704001803229   1.990492375818113   1.999799549536921
1.773930300468217   1.820234405226769   1.866800477731109
1.956015106717136   1.839959587489531   2.000000000000000
1.473931188332412   1.169925001442313   2.000000000000002];
ii=[1.789266160802318   1.991256913509331   1.996380159467360
1.804121461519321   1.860936382228666   1.899711943099106
1.924105150960719   1.851998837112445   2.000000000000000
1.473931188332412   1.169925001442313   2.000000000000002];
j=[1.782082217055803   1.991234323454376   1.995218423000029
1.750422389626583   1.795519401093348   1.780645078454589
1.860966306361995   1.974733241389570   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
jj=[1.795120645442478   1.991395198390710   1.995371113339872
1.774037759563524   1.792336032025250   1.809622122693223
1.851707680339209   1.898554736440454   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
k=[1.729533048752311   1.984635300459115   1.992463482348632
1.694722845282534   1.662036936112174   1.597074315824087
1.570996321557153   1.679740725473256   1.754887502163469
2.000000000000000   2.000000000000000 2.000000000000000];
kk=[1.709949604870203   1.984625222807601   1.990029459535633
1.649162958079002   1.629765806948468   1.595570600287138
1.630937929521542   1.696323609668861   1.754887502163469
2.000000000000000   2.000000000000000 2.000000000000000];
l=[1.899296387847347   1.995574282968009   1.993601506534549
1.940845389749728   1.963563259005707   1.935416293789809
1.908553644831163   1.835075679616054   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
ll=[1.902647505790608   1.995974607888407   1.996856872306022
1.939365059480141   1.960466778246123   1.968519252383328
1.954196310386875   1.917537839808027   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
m=[1.715715003010463   1.993344867604171   1.991666943116695
1.801885507643253   1.868298037690107   1.884445015550917
1.880079086258725   1.906890595608517   1.807354922057605
2.000000000000000   2.000000000000000 2.000000000000000];
mm=[1.711687455588567   1.991316463148920   1.996391502139241
1.819377450306816   1.884854489992118   1.920489706944341
1.980045393624355   1.893084796083488   1.614709844115208
2.000000000000000   2.000000000000000 2.000000000000000];
n=[1.736411691575842   1.988469750539394   1.994386775379061
1.712428602052544   1.725923799878829   1.790466288564510
1.862763617263861   1.906890595608519   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
nn=[1.701030335610312   1.989356097729621   1.999411649173818
1.699381290918727   1.743518088289007   1.800759976374798
1.923327485419193   2.000000000000000   1.999999999999999
2.000000000000000   2.000000000000000 2.000000000000000];
```

```
o=[1.778434530442391   1.989205208282247   1.991792971731509
1.755804687704438   1.791510526098849   1.827163403137786
1.842993748067265   1.904751684852183   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
oo=[1.789302031351435   1.990818918820490   1.992005650533764
1.760774319258891   1.797233299469315   1.865577121668803
1.856953382233836   1.913288366806669   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
p=[1.954602012142333   1.995597626490796   1.994661240673803
1.971724089505926   1.979778299295946   1.977828002823029
1.920883867688018   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
pp=[1.941515497965074   1.996681699472327   1.994587296918386
1.966329574089206   1.980985504754503   1.976979275448286
1.920883867688018   1.925999418556222   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
q=[1.721929949858536   1.987708155897873   1.993620204861778
1.677492902594451   1.701101454277252   1.742393467271303
1.868616391670703   1.778442230142366   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];
qq=[1.709343344255338   1.989599659654585   1.994118913736704
1.673419558278687   1.739463849643773   1.792103063712340
1.897526378750206   1.888743248898260   1.999999999999999
2.000000000000000   2.000000000000000 2.000000000000000];
u=[1.795821334744344   1.992045178894024   1.999417884026666
1.797429034513890   1.829594864674772   1.855205481829366
1.921748554335632   1.839063781784943   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
uu=[1.812878646853647   1.990182668635562   1.999296037817012
1.813097199933766   1.846478521659413   1.849903972319609
1.963817663693354   1.843537258364250   1.847996906554950
1.473931188332412   1.169925001442313   2.000000000000002];
s=[1.772237411517107   1.991579438720636   1.994063590129466
1.712315579597298   1.718463320170308   1.742645201066739
1.708537185839688   1.825056923791063   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
ss=[1.779393707568362   1.991267975501484   1.999713241101881
1.724156518259327   1.708917585736363   1.701512132426938
1.796548167101022   1.726239187573147   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
t=[1.719342597126365   1.991162467827313   1.995485097628615
1.714151286781141   1.798849064004123   1.863039461206056
1.883441928080349   1.917537839808026   1.695993813109901
1.473931188332412   1.169925001442313   2.000000000000002];
tt=[1.727209179165093   1.989120225530005   1.991515223547192
1.715192860090790   1.782743864466708   1.867951924888537
1.897139211044802   1.994981925233376   1.847996906554950
1.736965594166206   1.584962500721156   1.000000000000001];


ra=corrcoef(y,a);
raa=corrcoef(y,aa);
rb=corrcoef(y,b);
rbb=corrcoef(y,bb);
rc=corrcoef(y,c);
rcc=corrcoef(y,cc);
```

```
rd=corrcoef(y,d);
rdd=corrcoef(y,dd);
re=corrcoef(y,e);
ree=corrcoef(y,ee);
rf=corrcoef(y,f);
rff=corrcoef(y,ff);
rg=corrcoef(y,g);
rgg=corrcoef(y,gg);
rh=corrcoef(y,h);
rhh=corrcoef(y,hh);
ri=corrcoef(y,i);
rii=corrcoef(y,ii);
rj=corrcoef(y,j);
rjj=corrcoef(y,jj);
rk=corrcoef(y,k);
rkk=corrcoef(y,kk);
rl=corrcoef(y,l);
rll=corrcoef(y,ll);
rm=corrcoef(y,m);
rmm=corrcoef(y,mm);
rn=corrcoef(y,n);
rnn=corrcoef(y,nn);
ro=corrcoef(y,o);
roo=corrcoef(y,oo);
rp=corrcoef(y,p);
rpp=corrcoef(y,pp);
rq=corrcoef(y,q);
rqq=corrcoef(y,qq);
ru=corrcoef(y,u);
ruu=corrcoef(y,uu);
rs=corrcoef(y,s);
rss=corrcoef(y,ss);
rt=corrcoef(y,t);
rtt=corrcoef(y,tt);

m1=max(ra,raa); m2=max(rb,rbb); m3=max(rc,rcc); m4=max(rd,rdd);
m5=max(re,ree);m6=max(rf,rff); m7=max(rg,rgg); m8=max(rh,rhh);
m9=max(ri,rii); m10=max(rj,rjj);m11=max(rk,rkk); m12=max(rl,rll);
m13=max(rm,rmm); m14=max(rn,rnn); m15=max(ro,roo);m16=max(rp,rpp);
m17=max(rq,rqq); m18=max(ru,ruu); m19=max(rs,rss);m20=max(rt,rtt);
m21=max(m1,m2); m22=max(m3,m4); m23=max(m5,m6); m24=max(m7,m8);
m25=max(m9,m10); m26=max(m11,m12); m27=max(m13,m14);
m28=max(m15,m16); m29=max(m17,m18); m30=max(m19,m20);
m31=max(m21,m22); m32=max(m23,m24); m33=max(m25,m26);
m34=max(m27,m28); m35=max(m29,m30); m36=max(m31,m32);
m37=max(m33,m34); m38=max(m36,m37); m39=max(m35,m38);

if m39 > 0.99
    set(handles.text2,'String','DATA ADA');
else m39 < 0.99
    set(handles.text2,'String','DATA TIDAK ADA');
    axes(handles.axes1);
    imshow('pnihil.jpg');
end
% hObject    handle to pushbutton3 (see GCBO)
```

Lampiran G. Perhitungan manual proses adaptif histogram ekualisasi

Misalkan suatu citra yang diketahui matriksnya, kemudian citra tersebut dibagi-bagi menjadi beberapa grid. Misal matriks untuk salah satu gridnya adalah matriks berukuran 64×64 yang memiliki 8 derajat keabuan. Tabel nilai derajad keabuan disajikan dalam tabel:

k	rk	nk	nk/n	sk	sk
0	0/7 = 0	790	0,19	0,19	1/7
1	1/7 = 0,14	1023	0,25	0,44	3/7
2	2/7 = 0,29	850	0,21	0,65	5/7
3	3/7 = 0,43	656	0,16	0,81	6/7
4	4/7 = 0,57	329	0,08	0,89	6/7
5	5/7 = 0,71	245	0,06	0,95	1
6	6/7 = 0,86	122	0,03	0,98	1
7	7/7 = 1	81	0,02	1	1

k = derajat keabuan

$$r_k = \frac{k}{L-1} \qquad r_0 = \frac{0}{7}$$

n_k = jumlah nilai derajat keabuan dalam matriks

$$P_r(r_k) = \frac{n_k}{n}$$

$$s_k = T(r_k) = \sum_{j=0}^{k} P_r(r_j)$$

$s_0 = 0{,}19$ lebih dekat ke nilai $\frac{1}{7}$ $(= 0{,}14)$, maka $s_0 = 1/7$

$s_1 = 0{,}19$ lebih dekat ke nilai $\frac{3}{7}$ $(= 0{,}14)$, maka $s_0 = 3/7$

$s_2 = 0{,}19$ lebih dekat ke nilai $\frac{5}{7}$ $(= 0{,}14)$, maka $s_0 = 5/7$

$s_3 = 0{,}19$ lebih dekat ke nilai $\frac{6}{7}$ $(= 0{,}14)$, maka $s_0 = 6/7$

$s_4 = 0{,}19$ lebih dekat ke nilai $\frac{6}{7}$ $(= 0{,}14)$, maka $s_0 = 6/7$

$s_5 = 0{,}19$ lebih dekat ke nilai $\frac{7}{7}$ $(= 0{,}14)$, maka $s_0 = 7/7$

$s_6 = 0{,}19$ lebih dekat ke nilai $\frac{7}{7}$ $(= 0{,}14)$, maka $s_0 = 7/7$

$s_7 = 0{,}19$ lebih dekat ke nilai $\frac{7}{7}$ $(= 0{,}14)$, maka $s_0 = 7/7$

s_k	n_k	$P_s(s_k)$
$1/7$	790	0,19
$3/7$	1023	0,25
$5/7$	850	0,21
$6/7$	656+329=958	0,23
$7/7$	245+122+81=448	0,11

Berikut merupakan histogram sebelum perataan dan setelah perataan: