

MODUL DEEP LEARNING
FAKULTAS ILMU KOMPUTER



FAKULTAS ILMU KOMPUTER
UNIVERSITAS LANCANG KUNING

I. Soal & Penyelesaiannya

Lakukan proses Image Retrieval terhadap beberapa image dibawah ini dengan menggunakan Algoritma Neural Network Backpropagation:

Berikut Data Image yang digunakan untuk uji coba:



Lingkaran.png



lingkaran3.png



segiempat.png

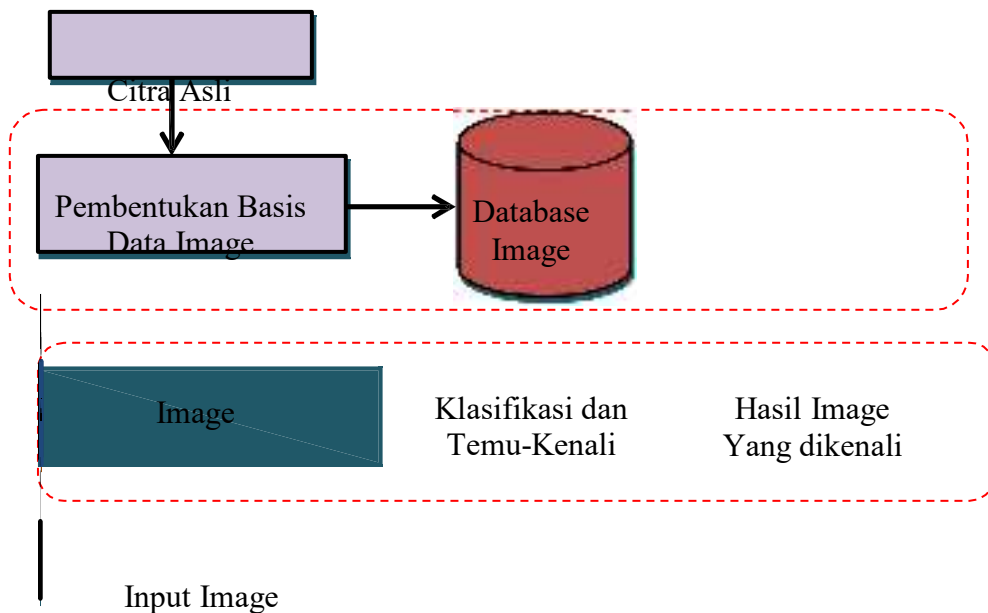


segitiga.png

Semua image diatas sudah tersedia dalam folder citra pada lampiran soal ini (*jangan mengubah dan mengurangi ukuran dan gunakan citra 4 diatas saja, citra yang lain digunakan untuk tugas kuliah*)

1.1 Tahapan Penelitian

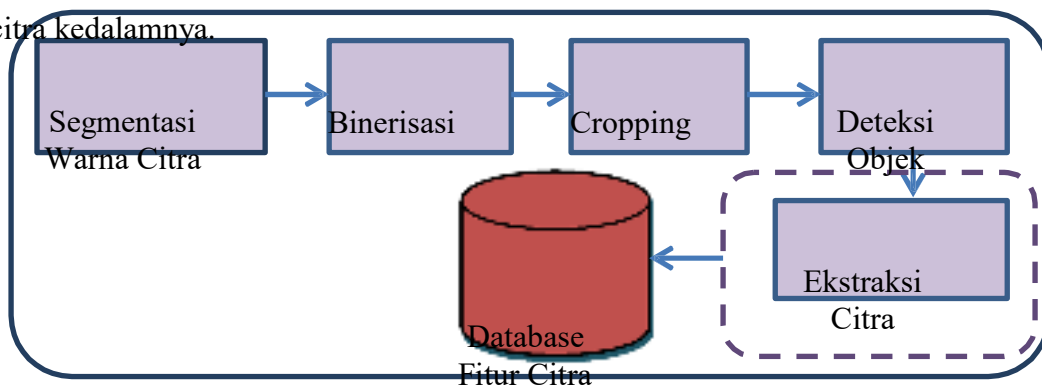
Metode penelitian terbagi dalam dua tahap dengan skema dapat dilihat pada Gambar 1. Tahap pertama adalah pembentukan basis data fitur pola citra yang digunakan untuk proses pelatihan pengenalan sebuah benda. Tahap kedua adalah proses klasifikasi dan temu-kenali. Tahap ini melakukan proses klasifikasi citra melalui proses *training* (pembelajaran) dan selanjutnya melakukan proses pengenalan citra yang terdapat pada *query* citra. Masing-masing tahap tersebut diuraikan secara rinci pada penjelasan tahap berikutnya.



Gambar 1. Tahapan Penelitian

1.2 Tahapan Pembuatan Database Motif Songket

Sebagaimana telah diuraikan pada skema Gambar 1, tentang tahap pembentukan basis data citra dengan tujuan untuk mendapatkan fitur citra referensi dan disimpan kedalam database. Tahap ini terdiri dari enam proses seperti yang diperlihatkan pada Gambar 2, yaitu: proses segmentasi warna citra, proses binerisasi, proses cropping, proses deteksi objek, proses ekstraksi fitur citra serta proses pembuatan database dan penyimpanan fitur citra kedalamnya.



Gambar 2. Bagan Pembuatan Database Citra

1.2.1 Segmentasi Motif Berdasarkan Warna Citra

Proses segmentasi citra didasarkan pada kesamaan warna antara warna setiap pixel terhadap warna background citra. Ruang warna RGB dan Euclidian distance digunakan untuk mengukur jarak antara dua warna tersebut. Rumus Euclidian distance warna RGB diberikan oleh persamaan:

$$D = \sqrt{(R_{ref} - R_p)^2 + (G_{ref} - G_p)^2 + (B_{ref} - B_p)^2} \quad (1)$$

dimana R_{ref} , G_{ref} dan B_{ref} , masing-masing adalah komponen warna merah, hijau dan biru referensi dari warna background citra. R_p , G_p dan B_p , masing-masing merupakan tiga komponen warna dari setiap pixel dalam citra.

Prinsip dasar segmentasi warna motif songket adalah menghapus semua pixel yang memiliki warna yang sama atau mendekati warna background citra ($D < Th$) dan mengambil semua pixel yang memiliki jarak warna $D > Th$. Nilai threshold Th dapat ditentukan berdasarkan hasil eksperimen. Mengacu pada prinsip dasar dan persamaan 1 di atas, dapat dibuat algoritma segmentasi seperti yang diberikan pada Algoritma 1.

Algoritma 1. Algoritma Segmentasi Warna Citra

1. Baca citra asli
2. Tampilkan citra yang akan disegmentasi
3. Pembacaan posisi warna *pixel*
4. Penentuan warna referensi ruang RGB
5. Penentuan nilai batas ambang (*threshold*) = 50
6. Pembacaan ukuran citra RGB
7. Inisialisasi matriks citra hasil RGB
8. Perhitungan jarak warna RGB
9. Melakukan pemisahan objek dengan latar belakang
10. Tampilkan citra hasil proses
11. Simpan citra hasil proses *segmentasi*

Petunjuk :

Buat sebuah folder dengan nama 2.citra hasil segmentasi lalu copykan kedalam folder ini file image lingkaran.png, lingkaran3.png segiempat.png, segitiga.png, selanjutnya buka matlab dan copy source code program dibawah ini dan simpan dengan nama segmentasi.m dalam folder 2.citra hasil segmentasi. Program ini tidak menggunakan GUI oleh sebab itu pertama robah nama citra asli dengan lingkaran.png pada coding dan ganti namanya menjadi segementasi_lingkaran.png pada simpan citra segmentasi. Pada saat program sudah jalan dan muncul Citra Asli maka double klik pada warna putih (background) sehingga akan muncul Citra Hasil Segmentasi dan otomatis tersimpan sebuah file dengan nama segmentasi_lingkaran.png, lakukan juga hal yang sama untuk lingkaran3.png, segiempat.png dan segitiga.png.

Coding segmentasi.m:

```
function SegmentasiCitra()
clear all;

% Baca citra asli
Im=double(imread('lingkaran.png'));

% Tampilkan citra RGB yang akan disegmentasi
figure(1), imshow(uint8(Im));title('Citra Asli');

% Pembacaan posisi warna piksel sebagai warna area untuk segmentasi
p = impoint(gca,[]);
p = wait(p);
x=round(p(1,1));
y=round(p(1,2));

% Penentuan warna referensi ruang RGB
RGB=Im(y,x,:);

% Penentuan nilai threshold
Th=50; % tentukan nilai threshold

% Pembacaan ukuran citra RGB
[m,n,l]=size(Im);

% Inisialisasi matriks citra hasil RGB
cit_hasilRGB(1:m,1:n,1:l)= uint8(0);

% Perhitungan jarak warna RGB
for i=1:m
    for j=1:n
        dR=(RGB(1,1,1)-Im(i,j,1))^2;
        dG=(RGB(1,1,2)-Im(i,j,2))^2;
        dB=(RGB(1,1,3)-Im(i,j,3))^2;
        % Pemisahan objek dengan latarbelakangnya & simpan dalam
        % cit_hasilRGB
        if sqrt(dR + dG + dB)>=Th
            cit_hasilRGB(i,j,:)= Im(i,j,:);
        end
    end;
end;

% Tampilkan citra hasil proses
figure(2), imshow(cit_hasilRGB);title('Citra Hasil');
%imwrite(cit_hasilRGB,'Citra Menggunakan Ruang RGB')

% Simpan citra hasil Segmentasi
imwrite(cit_hasilRGB,'segmentasi_lingkaran.png');
end
```

1.2.2 Binerisasi

Pada tahap ini citra hasil proses segmentasi dilakukan proses binerisasi dengan tujuan untuk mengubah citra motif berwarna menjadi citra keabuan dan kemudian menjadi citra biner. Hal ini bertujuan untuk memperjelas batas area setiap objek dan memudahkan proses analisis berikutnya. Hasil proses binerisasi kadang masih mengandung noise berupa bintik-bitik kecil berwarna putih, sehingga dibutuhkan proses penghilangan noise. Dalam penelitian ini digunakan morfologi matematika opening untuk menghilangkan noise tersebut. Berikut diberikan algoritma proses binerisasi dan penghilangan noise:

Algoritma 2. Algoritma Binerisasi

1. Baca citra songket hasil segmentasi
2. Tampilkan citra hasil segmentasi
3. Konversi citra RGB ke *grayscale* dan tampilkan citra *grayscale*
4. Konversi citra *grayscale* ke *biner* dan tampilkan citra *biner*
5. Simpan citra *biner*
6. Hilangkan *noise* pada citra *biner* dan tampilkan citra hasil
7. Simpan citra hasil pembersihan *noise*

Petunjuk :

Buat sebuah folder dengan nama 3.citra hasil biner lalu copykan kedalam folder ini semua file citra hasil segmentasi, selanjutnya buka matlab dan copy source code program dibawah ini dan simpan dengan nama biner.m dalam folder 3.citra hasil biner. Program ini tidak menggunakan GUI oleh sebab itu pertama robah nama baca citra segmentasi dengan segmentasi_lingkaran.png pada coding dan ganti namanya menjadi biner_lingkaran.png pada simpan citra biner serta bersih_biner_lingkaran.png pada hasil pembersihan noise. Pada saat program sudah jalan dan muncul Citra Hasil Segmentasi, Citra Abu-abu, Citra Biner, tersimpan dua buah file dengan nama biner_lingkaran.png dan bersih_biner_lingkaran. Lakukan juga hal yang sama untuk 3 citra berikutnya. (*perhatikan hasilnya dimana kelihatan mirip tapi tidak*)

Coding biner.m:

```
function biner()
clear all;

% Baca citra hasil segmentasi
cit_hasilRGB=imread('segmentasi_lingkaran.png');

% Tampilkan citra hasil disegmentasi
figure(1), imshow(uint8(cit_hasilRGB));title('Citra Hasil Segmentasi');

% Konversi citra RGB ke grayscale dan tampilkan citra grayscale
Gray = rgb2gray(cit_hasilRGB);
figure(3), imshow(Gray);title('Citra Abu-abu');

% Konversikan citra Grayscale ke biner dan tampilkan citra Biner
thresh=graythresh(Gray);
Biner=im2bw(Gray,thresh);
figure(4), imshow(Biner);title('Citra Biner');

% Simpan citra biner
imwrite(Biner,'biner_lingkaran.png');

% Hilangkan noise pada citra biner dan tampilkan hasil
BW=bwareaopen(Biner,50);
figure(5), imshow(BW);

% Simpan hasil pembersihan noise
imwrite(BW,'biner_bersih_lingkaran.png');
end
```

1.2.3 Cropping

Pada tahap ini citra biner hasil pembersihan di cropping untuk mengambil citra sesuai kebutuhan saja (*jika dirasa cukup maka boleh tidak melukan cropping, pada contoh ini tidak dilakukan karena citra sudah cukup kecil*), berikut algoritmanya:

Algoritma 3. Algoritma Cropping

1. Baca citra hasil pembersihan biner
2. Tampilkan citra hasil hasil pembersihan biner
3. Pilih yang mau di cropping
4. Simpan citra hasil cropping

Petunjuk :

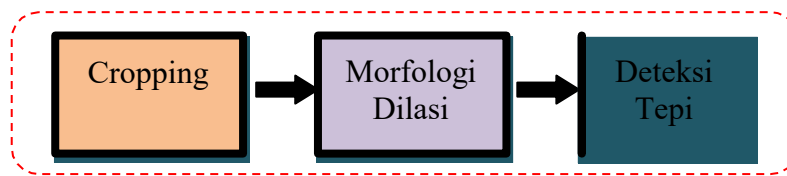
Buat sebuah folder dengan nama 4.citra hasil crop lalu copykan kedalam folder ini semua file citra hasil pembersihan biner, selanjutnya buka matlab dan copy source code program dibawah ini dan simpan dengan nama crop.m dalam folder 4.citra hasil crop. Program ini tidak menggunakan GUI oleh sebab itu pertama robah nama tampilan citra biner dengan biner_bersih_lingkaran.png pada coding dan ganti nama simpan hasil crop menjadi crop_lingkaran.png pada simpan citra biner. Pada saat program sudah jalan dan muncul Citra Hasil Binner lalu pilih koordinat yang akan di crop maka muncul hasil crop, simpan hasil crop. Lakukan juga hal yang sama untuk 3 citra berikutnya

Coding crop.m:

```
function crop()
clear all;
%tampilkan citra
Im=imread('biner_bersih_lingkaran.png');
% menampilkan citra asli
figure(1); imshow(Im);
% judul dari citra yang ditampilkan
title('Cropping [klik 1:Titik kiri atas] - [klik 2:Titik kanan bawah]');
p = ginput(2);
x0 = min(floor(p(1)), floor(p(2)));           %xmin
y0 = min(floor(p(3)), floor(p(4)));           %ymin
x1 = max(ceil(p(1)), ceil(p(2)));             %xmax
y1 = max(ceil(p(3)), ceil(p(4)));             %ymax
%hasil cropping Ipot=Im(y0:y1,x0:x1);
figure(2);imshow(Ipot);title('Cropping Image');
% Simpan hasil crop
imwrite(Ipot, 'crop_lingkaran.png');
end
```

1.2.4 Deteksi Objek

Operasi morfologi dapat digunakan sebagai algoritma pendeteksian tepi citra. Metode morfologi yang dipilih dalam tahapan ini yaitu proses morfologi dilasi dan dilanjutkan dengan proses thinning untuk deteksi tepi objek. Bagan dari proses deteksi objek yang dilakukan dapat dilihat pada Gambar 3:



Gambar 3. Bagan Proses Deteksi Objek

Proses *morfologi dilasi* ditujukan untuk memperlebar area tepian objek, sehingga bagian citra dapat terlihat lebih jelas. Proses berikutnya dilakukan *thinning* untuk deteksi tepi. Tahapan deteksi objek dapat dilihat pada algoritma 4.

Algoritma 4. Algoritma Deteksi Objek

1. Baca citra hasil *cropping*
2. Tampilkan citra hasil *cropping*
3. Proses *morfologi dilasi*
4. Tampilkan motif hasil morfologi dilasi
5. Proses deteksi tepi
6. Simpan Hasil Deteksi Tepi

Petunjuk :

Buat sebuah folder dengan nama 5.citra hasil deteksi lalu copykan kedalam folder ini semua file citra hasil cropping, selanjutnya buka matlab dan copy source code program dibawah ini dan simpan dengan nama deteksi.m dalam folder 5.citra hasil deteksi. Program ini tidak menggunakan GUI oleh sebab itu pertama robah membaca hasil hasil cropping dengan crop_lingkaran.png pada coding dan ganti nama menyimpan proses deteksi tepi menjadi deteksi_lingkaran.png. Lakukan juga hal yang sama untuk 3 citra berikutnya.

Coding deteksi.m:

```
% Proses Morfologi Dilasi
function dilasi()
clear all;

% Membaca hasil hasil cropping
MP=imread('crop_lingkaran.png');

% Menampilkan citra hasil cropping
figure(1), imshow(MP);

% Proses morfologi dilasi
B=[1 1 1;1 1 1;1 1 1];
Dilasi1=imdilate(MP,B);
Dilasi2=imdilate(Dilasi1,B);

% Menampilkan hasil proses morfologi dilasi
figure(2), imshow(Dilasi2);

% Proses deteksi tepi
bw=bwmorph(Dilasi2,'remove');

% Menampilkan proses deteksi tepi
figure(3), imshow(bw);

% Menyimpan proses deteksi tepi
imwrite(bw,'tepi_lingkaran.png')
end
```

1.2.5 Ekstraksi Motif

Pada tahap ini citra hasil proses *deteksi objek* dilakukan proses *ekstraksi citra* untuk mendapatkan nilai yang akan dijadikan sebagai ciri dari sebuah citra. Pada sebuah citra bisa terdapat satu atau banyak objek didalamnya, untuk mendapatkan kontur yang telah terurutkan digunakan pelacakan kontur algoritma *moore* (*yang belum dikembangkan, jadi pada tugas akhir saudara akan diminta untuk mengembangkan algoritma moore ini sehingga ada kebaruan/novelty disini*). Setelah memperoleh kontur yang telah terurutkan maka selanjutnya dicari nilai rantai dengan algoritma *chain code* (*yang belum, jadi pada tugas akhir saudara akan diminta untuk mengembangkan algoritma chain code ini sehingga ada kebaruan/novelty disini*). Hasil proses dari algoritma *chain code* diperoleh nilai akhir berupa nilai probabilitas kemuculan masing-masing rantai (0-7) dengan membuat rumus probabilitas sendiri (dalam hal ini

sudah tersedia), selanjutnya disimpan dalam database sebagai nilai ciri / karakteristik dari citra, dengan tahapan dapat dilihat pada algoritma 5:

Algoritma 4. Algoritma Ekstraksi Motif

1. Baca citra hasil deteksi tepi
2. Tampilkan citra hasil deteksi tepi
3. Tracing citra menggunakan pelacakan kontur algoritma *moore* yang belum dikembangkan
4. Mencari masing-masing nilai rantai objek yang terdapat pada citra dengan algoritma *chain code* yang belum dikembangkan
5. Hitung nilai probabilitas masing-masing objek pada citra
6. Simpan nilai probabilitas hasil proses dalam database

Petunjuk :

Buat sebuah folder dengan nama 6.citra hasil ekstraksi lalu copykan kedalam folder ini semua file citra hasil deteksi tepi, selanjutnya buka matlab dan copy source code program dibawah ini dan simpan dengan nama ekstraksi.m, program ini akan membaca function inbound tracing guna membaca kontur pada citra dan simpan program dibawah dengan nama inbound_tracing.m (program ini belum dikembangkan, ada pada bahan 8.pdf) pada folder

6.citra hasil ekstraksi. Program ekstraksi.m juga membaca function chain code (program ini belum dikembangkan, ada pada bahan 8.pdf), fungsi dari program ini adalah untuk menghitung nilai rantai pada kontur citra, Program ini tidak menggunakan GUI oleh sebab itu pertama robah membaca hasil deteksi tepi dengan tepi_lingkaran.png pada coding ekstraksi. Lakukan juga hal yang sama untuk 3 citra berikutnya.

Coding ekstraksi.m

```
% Membaca hasil deteksi tepi
bw=imread('tepi_segitiga.png');

% Menampilkan hasil deteksi tepi
figure(1), imshow(bw);

% Inbound Tracing dengan kontur moore
cc=inbound_tracing(bw);

% Proses mencari nilai rantai dengan chain code
[kode,x,y]=chain_code(cc);

% Proses menghitung nilai probabilitas masing-masing objek pada motif
prob (1,1:8) = 0;
k = 1; count
= 0;
length(kode)
for i = 1 : length(kode)
    Num=str2num(kode(i));
    if (Num<=7) && i<length(kode)
        prob(k,Num+1) = prob(k,Num+1)+1;
        count = count + 1;
    else
        count
        prob(k,1:8) = prob(k,1:8)/count
        count = 0;
        k = k + 1;
        prob (k,1:8) = 0;
    end
end
end
```

Coding Inbound_tracing.m

```
function [Kontur] = inbound_tracing(BW)
% INBOUND_TRACING Memperoleh kontur yang telah terurutkan
% dengan menggunakan algoritma pelacakan kontur Moore

[jum_baris, jum_kolom] = size(BW);
BW1(1:jum_baris+5, 1:jum_kolom+5)=0;
BW1(3:jum_baris+2, 3:jum_kolom+2)=BW;
BW=BW1;
[jum_baris, jum_kolom] = size(BW);
% Peroleh piksel awal
selesai = false;
```

```
for p = 1 : jum_baris
    for q = 1 : jum_kolom
        if BW(p, q) == 1
            b0.y = p;
            b0.x = q;
            selesai = true;
            break;
        end
    end

    if selesai
        break;
    end
end

c0 = 4; % Arah barat

% Periksa 8 tetangga dan cari piksel pertama yang bernilai 1
for p = 1 : 8
    [dy, dx] = delta_piksel(c0);
    if BW(b0.y + dy, b0.x + dx) == 1
        b1.y = b0.y + dy;
        b1.x = b0.x + dx;
        c1 = sebelum(c0);
        break;
    else
        c0 = berikut(c0);
    end
end

Kontur=[];
Kontur(1, 1) = b0.y;
Kontur(1, 2) = b0.x;
Kontur(2, 1) = b1.y;
Kontur(2, 2) = b1.x;

%Kontur

n = 2; % Jumlah piksel dalam kontur

b = b1;
c = c1;
```

```
% Ulang sampai berakhir
while true
    for p = 1 : 8
        [dy, dx] = delta_piksel(c);
        if BW(b.y + dy, b.x + dx) == 1
            b.y = b.y + dy;
            b.x = b.x + dx;

            c = sebelum(c);

            n = n + 1;
            Kontur(n, 1) = b.y;
            Kontur(n, 2) = b.x;

            break;
        else
            c = berikut(c);
        end
    end
end

% Kondisi pengakhir pengulangan

if (b.y == b0.y) && (b.x == b0.x)
    break;
end

end

return

function [b] = berikut(x)
if x == 0
    b = 7;
else
    b = x - 1;
end

function [s] = sebelum(x)
if x == 7
    s = 0;
else
    s = x + 1;
end
```



```
if s < 2
    s = 2;
elseif s < 4
    s = 4;
elseif s < 6
    s = 6;
else
    s = 0;
end

function [dy, dx] = delta_piksel(id)
if id == 0
    dx = 1; dy = 0;
elseif id == 1
    dx = 1; dy = -1;
elseif id == 2
    dx = 0; dy = -1;
elseif id == 3
    dx = -1; dy = -1;
elseif id == 4
    dx = -1; dy = 0;
elseif id == 5
    dx = -1; dy = 1;
elseif id == 6
    dx = 0; dy = 1;
elseif id == 7
    dx = 1; dy = 1;
end
```

Coding Chain_code.m

```
function [kode_rantai, xawal, yawal] = chain_code(U)
% CHAIN_CODE Digunakan untuk mendapatkan titik awal (x, y) dan
% kode rantai dari kontur U yang datanya telah terurutkan
% misalnya melalui get_contour

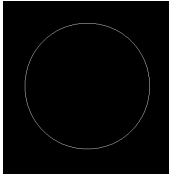
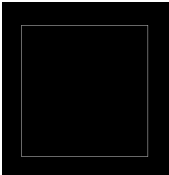
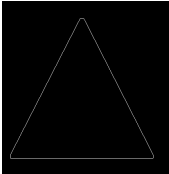
% Kode      1      2      3      4      5      6      7      8      9      10
Kode =      ['3', '2', '1', '4', '0', '0', '5', '6', '7', ' '];

xawal = U(1,2);
yawal = U(1,1);
kode_rantai = '';

for p=2: length(U)
    deltay = U(p, 1) - U(p-1, 1);
    deltax = U(p, 2) - U(p-1, 2);
    indeks = 3 * deltay + deltax + 5;
    kode_rantai = strcat(kode_rantai, Kode(indeks));
end
end
```


3. Kalau ingin melihat nilai probabilitas silakan buka prob (nilai probabilitas adalah diperoleh dari panjang rantai lalu dibuatkan rumus kemungkinan muncul masing-masing rantai (0-7), nilai probabilitas inilah yang nanti dijadikan nilai ciri pada citra, supaya jangan bertanya darimana datangnya rumus ini, maka jawabannya telah dicari sebelumnya), Cara mengcopy nilai probabilitas adalah buka prob lalu copy isinya dan paste di excel dan lakukan pembulatan 4 digit dibelakang koma, seperti dibawah ini: (jika proses sama maka saudara akan mendapatkan nilai yang sama seperti tabel dibawah)

Tabel 1. Hasil Ekstraksi

Nama Citra	Citra Hasil Deteksi tepi	Panjang Kode	Nilai Probabilitas							
			0	1	2	3	4	5	6	7
Lingkaran		882	0,1453	0,1010	0,1510	0,1022	0,1453	0,1022	0,1510	0,1022
Lingkaran3		882	0,1453	0,1010	0,1510	0,1022	0,1453	0,1022	0,1510	0,1022
Segiempat		1264	0,2447	0,0000	0,2549	0,0000	0,2447	0,0000	0,2557	0,0000
Segitiga		1037	0,0077	0,1641	0,1631	0,0010	0,3349	0,0019	0,1622	0,1651

Pada gambar diatas ada tulisan dimerahkan pada Lingkaran3, kalau saudara jeli melihatnya maka nilainya adalah sama dengan Lingkaran (Pada tugas berikutnya saudara akan ditugaskan untuk mencari bagaimana pelacakan kontur diatas bisa membaca semua kontur yang ada, seperti Lingkaran3 ada didalamnya 3 buah lingkaran kecil lagi, tapi algoritma moore yang ada tidak bisa membacanya, begitu juga algoritma chain code tidak bisa membacanya juga sehingga hanya kontur bagian luar saja yang bisa dibaca, akibatnya dalam image retrieval nanti citra lingkaran.png dan citra Lingkaran3.png akan dianggap sama, tentu itu merupakan suatu kesalahan yang tidak bisa ditolerir. Jadi tugas saudara nanti untuk mengembangkan 2 algoritma tersebut, tapi jangan panik dulu karena tentu saya akan berikan saudara pseudocode hasil algoritma yang telah saya kembangkan (seperti ini untuk mendapatkan kebaruan/novelty dalam sebuah penelitian).

Mari kita lanjutkan tugas berikutnya, sesuai bagan pada Gambar 2 yaitu membuat database citra. Buat seperti tabel berikut ini pada matlab:

0,1453	0,1010	0,1510	0,1022	0,1453	0,1022	0,1510	0,1022	1
0,2447	0,0000	0,2549	0,0000	0,2447	0,0000	0,2557	0,0000	2
0,0077	0,1641	0,1631	0,0010	0,3349	0,0019	0,1622	0,1651	3

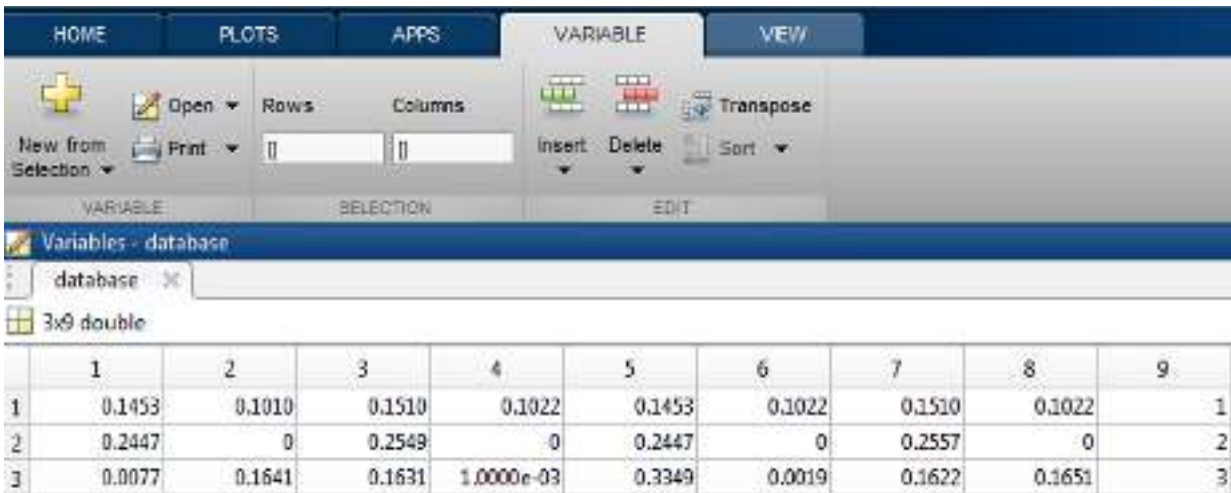
Gambar 5. Contoh Pengisian Database

Keterangan:

1. Probabilitas untuk lingkaran3 tidak usah dimasukan karena itu digunakan untuk melihatkan kepada saudara semuanya bahwa algoritma tersebut sebelum dikembangkan tidak bisa membaca kontur yang ada didalamnya, sehingga ciri dari citra tersebut tidak akurat maka berakibat dalam image retrieval sulit untuk mencari sebuah citra karena kemungkinan ciri yang sama sangat banyak didalam database.

Berikut langkah-langkah cara membuat database pada matlab:

1. Bisa melakukan dengan cara import data dari excel pada matlab
2. Cara yang mudah adalah buka saja file database.mat (data masih kosong) yang diberikan, lalu buka dimatlab dan isi seperti gambar dibawah ini:



The screenshot shows the MATLAB Variable Editor interface. The 'VARIABLE' tab is selected, displaying a variable named 'database' of type '3x9 double'. The variable is expanded to show its contents as a table with 9 columns and 3 rows. The columns are indexed 1 through 9. The data in the table is as follows:

	1	2	3	4	5	6	7	8	9
1	0.1453	0.1010	0.1510	0.1022	0.1453	0.1022	0.1510	0.1022	1
2	0.2447	0	0.2549	0	0.2447	0	0.2557	0	2
3	0.0077	0.1641	0.1631	1.0000e-03	0.3349	0.0019	0.1622	0.1651	3

Gambar 6. Database Hasil Ekstraksi yang Disimpan Pada Database Matlab

Pada kolom 9 ada angka 1, 2 dan 3, itu adalah kode urutan citra, 1. Lingkaran.png, 2. Segiempat.png, 3. Segitiga.png (jangan lupa simpan workspace → lihat disudut kanan atas matlab)

3.4. Tahapan Klasifikasi dan Temu-Kenali Motif Songket

Pada tahap ini hasil proses ekstraksi citra yang telah disimpan dalam database diuji dengan citra asli, untuk mengukur dan membandingkan tingkat kesamaan citra dengan data latih yang tersimpan dalam database. Seluruh tahapan dapat dilihat pada algoritma 5 dan flowchart pada Gambar 7.

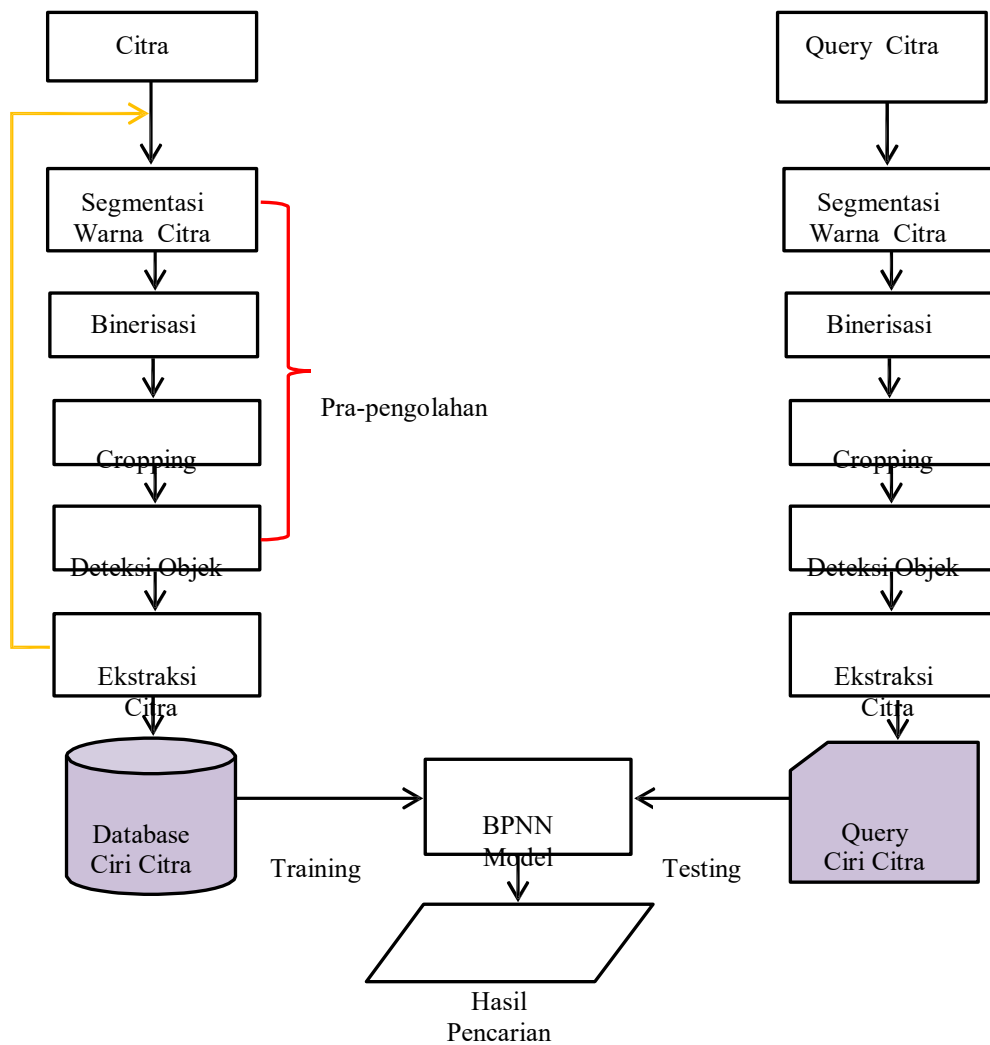
Algoritma 5. Algoritma Klasifikasi dan Temu-Kenali

Proses Pelatihan :

1. Input citra songket / *query* motif songket
2. Proses segmentasi warna citra
3. Proses *binerisasi*
4. Proses *cropping*
5. Proses deteksi objek songket
6. Proses ekstraksi motif songket
7. Ulangi proses 1 sampai dengan 6, sampai tidak ada lagi motif yang mau diproses.
8. Database Ciri Motif songket di latih dengan BPNN
9. Hasil Pelatihan

Proses Uji :

1. Input citra songket / *query* motif songket
2. Proses segmentasi warna citra
3. Proses *binerisasi*
4. Proses *cropping*
5. Proses deteksi objek songket
6. Proses ekstraksi motif songket
7. Database Ciri Motif songket di uji dengan net keluaran hasil proses BPNN
8. Hasil Uji



Gambar 7. Flowchart Image Retrieval (Backpropagation Neural Network)

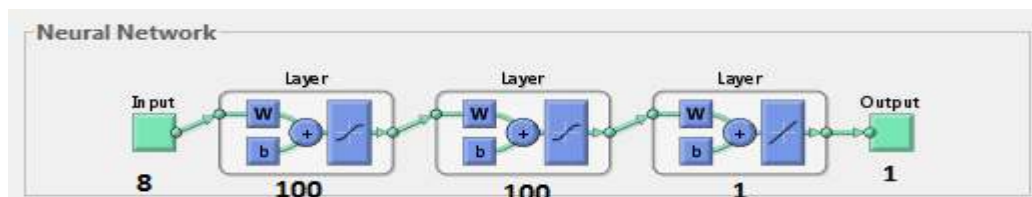
Pada algoritma 5 terdapat 2 bagian, bagian pertama algoritma proses pelatihan dan bagian kedua algoritma kedua proses pengujian. Pada proses pelatihan dan pengujian ini menggunakan algoritma jaringan syaraf tiruan propagasi balik (backpropagation neural network).

3.4.1 Pelatihan Jaringan Syaraf Tiruan Propagasi Balik

Adapun tujuan dari pelatihan jaringan syaraf tiruan propagasi balik adalah untuk membuat jaringan dapat beradaptasi terhadap pola dari citra yang digunakan sebagai data latih. sehingga bobot yang terdapat dalam jaringan bisa beradaptasi dan menghasilkan target keluaran yang diinginkan. Sebelum memulai latihan pada jaringan maka terlebih dahulu ditentukan parameter-parameter dan komponen yang akan dilibatkan dalam pembuatan jaringan tersebut. Hasil dari parameter tersebut digunakan untuk pelatihan dengan harapan pelatihan yang dilakukan membuat tiap bobot dapat beradaptasi dan beberapa parameter dapat diperbaiki sehingga terbentuk jaringan yang pintar yang dapat mengenali target keluaran yang dikehendaki dari data latih.

Parameter-parameter yang digunakan dalam pembangunan jaringan ini meliputi, iterasi (epochs), laju pembelajaran (learning rate), jumlah neuron pada setiap hidden layer serta goal. Epochs dan learning rate ditentukan dengan cara melihat mean square error (MSE) saat pelatihan dilakukan, semakin kecil MSE semakin bagus kerja dari jaringan syaraf tiruan.

Awal pelatihan ini ditetapkan parameter dengan arsitektur jaringan 8 masukan (input), dan 2 lapisan tersembunyi (hidden layer) dengan masing-masing 100 neuron pada lapisannya dan 1 keluaran (output). Fungsi aktivasi yang digunakan adalah fungsi aktivasi tansig untuk lapisan input dan fungsi aktivasi purelin untuk lapisan output serta traingdx untuk fungsi pelatihan jaringan dengan metode penurunan tercepat dengan variabel laju pemahaman ditambah momentum. Data latih yang digunakan adalah vektor matriks seperti yang dapat dilihat pada Tabel 4.7 dengan inisialisasi bobot dan bias dilakukan secara acak dengan jangkauan -0,5 sampai dengan +0,5 (Prasetyo, 2012).



Gambar 8. Arsitektur JST Propagasi Balik Data Latih Citra

Gambar 8 merupakan gambaran singkat terhadap jaringan yang digunakan pada latihan ini dengan 8 input, 100 neuron masing-masing pada 2 hidden layer dan 1 output, berikut langkah-langkah yang digunakan dalam mencari parameter-parameter yang tepat untuk jaringan ini.

Tabel 2. Konfigurasi Akhir Parameter Pelatihan

No.	Nama Parameter	Jumlah
1	Arsitektur Jaringan	8-100-100-1
2	Neuron Input	8
3	Neuron pada hidden layer 1	100
4	Neuron pada hidden layer 2	100
5	Neuron Output	1
6	Learning rate	0,1
7	Goal	0,0001
8	Maksimum epoch	800
9	Momentum	0,95

Bagaimana cara memperoleh Parameter Pelatihan Pada Backpropagation Neural Network tidak dijelaskan secara detil disini karena kalau dijelaskan terlalu luas dan panjang penjelasannya, jika ingin lebih tahu tentang BPNN maka dapat melihat Disertasi saya di Pustaka UPI YPTK Padang atau membaca buku yang berjudul *Jaringan Syaraf Tiruan & Pemrogramannya Menggunakan Matlab*, penulis Siang, Jong Jek (2009).

3.4.2 Pemrograman Matlab untuk Image Retrievalnya

Agar bisa memahami dengan baik tentang BPNN (*Backpropagation Neural Network*), disarankan untuk banyak belajar lagi, karena latihan diberikan agar saudara mengetahui bagaimana sebuah proses dalam pengolahan citra mulai dari awal sampai image retrieval, akan tetapi fokus kita disini adalah untuk mempelajari tentang mendapatkan ciri dari sebuah citra dengan proses mulai: Citra – Segmentasi – Biner – Cropping – Ekstrak – Simpan Database. Selanjutnya kita masuk ke tahap pemrograman Image Retrieval:

1. BPNN dalam prosesnya harus melalui tahapan pelatihan :

- Buat sebuah folder dengan nama 7. JST testing
- Copy folder database yang sudah diisi kedalam folder 7. JST testing
- Copy folder bobot_awal kedalam folder 7. JST testing (*bobot_awal ini sudah melalui proses pencarian, jika ingin mempeleajari lebih jauh silakan baca buku sesuai petunjuk sebelumnya, ini bisa juga digunakan untuk kasus lain tapi sesuaikan jumlah neuronnya*)
- Copy folder data_uji kedalam folder 7. JST testing (data uji untuk sementara diisi sama dengan database, mestinya adalah data uji, didalam BPNN ada 2 jenis data yaitu: data latih dan data uji)
- Selanjutnya copy program dibawah ini dan masukkan kedalam folder 7. JST testing

Coding pelatihan_jst.m

```
clc; clear; close all; warning off all;

load database
X = database;
load data_uji
Y = uji;

data_latih = X(:,1:8)';
target = Y(:,9)';
n = numel(target);

% Pembuatan JST
load bobot_awal.mat
net = newff(minmax(data_latih),[100 100
1],{'tansig','tansig','purelin'},'traingdx');
net.IW{1,1} = bobot_hidden;
net.LW{2,1} = bobot_keluaran;
net.LW{3,2} = bobot_keluaran2;

net.b{1,1} = bias_hidden;
net.b{2,1} = bias_hidden2;
net.b{3,1} = bias_keluaran;

% Memberikan nilai untuk mempengaruhi proses pelatihan
```

```
net.performFcn = 'mse';
net.trainParam.goal = 0.0001;
net.trainParam.show = 20;
net.trainParam.epochs = 1000;
net.trainParam.mc = 0.95;
net.trainParam.lr = 0.1;

% Proses training
[net_keluaran,tr,Y,E] = train(net,data_latih,target);

% Hasil setelah pelatihan
bobot_hidden = net_keluaran.IW{1,1};

bobot_keluaran = net_keluaran.LW{2,1};
bobot_keluaran2 = net_keluaran.LW{3,2};

bias_hidden = net_keluaran.b{1,1};
bias_hidden2 = net_keluaran.b{2,1};

bias_keluaran = net_keluaran.b{3,1};

jumlah_iterasi = tr.num_epochs;
nilai_keluaran = Y;
nilai_error = E;
error_MSE = (1/n)*sum(nilai_error.^2);

save net_keluaran.mat net_keluaran

% Performa jaringan
hasil_uji = sim(net_keluaran,data_latih);
output = round(hasil_uji);

total_images = numel(output);
[m,n] = find(output==target);
akurasi = sum(m)/total_images*100
```

2. Setelah pelatihan dilakukan maka dianggap BPNN sudah pintar dan sudah bisa mengenali, maka lakukan proses image retrieval:
- Buat sebuah folder dengan nama 8. Program Image Retrieval
 - Copy database, net_keluaran, bobot_awal hasil pelatihan kedalam folder 8. Program Image Retrieval
 - Copy program inbound_tracing.m dan chain_code.m kedalam folder 8. Program Image Retrieval
 - Copy file crop_lingkaran.png, crop_segiempat.png, crop_segitiga.png, lingkaran.png, segiempat.png, segitiga.png, unknown.png kedalam folder 8. Program Image Retrieval
 - Selanjutnya copy program dibawah ini dan masukkan kedalam folder 8. Program Image Retrieval

Isi coding pengenalan_citra_jst.m (bagian function)

```
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)
[filename,pathname] = uigetfile({'*.jpg'});

if ~isequal(filename,0);
    Img = imread(fullfile(pathname,filename));
    axes(handles.axes1)
    cla reset
    set(gca,'XTick',[])
    set(gca,'YTick',[])
    imshow(Img)
    title('Motif Input')
    handles.Img = Img;
    guidata(hObject,handles)
    set(handles.pushbutton5,'Enable','on')
else
    return
end
```

```
% --- Executes on button press in pushbutton5.
function pushbutton5_Callback(hObject, eventdata, handles)
% 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)

cropped_im = handles.Img; B=[1
1 1;1 1 1;1 1 1];
Dilasi1=imdilate(cropped_im,B);
Dilasi2=imdilate(Dilasi1,B);

[pixel_labels,~] = bwlabel(Dilasi2);
max_labels = max(max(pixel_labels));
area_px = zeros(1,max_labels);

for n = 1:max_labels
    area_px(n) = sum(find(pixel_labels==n));
end

[~,cluster_fish] = max(area_px);
bw = (pixel_labels==cluster_fish);
bw=bwmorph(bw, 'remove');

axes(handles.axes2)
imshow(bw);
title('Hasil Morfologi')

handles.bw = bw; guidata(hObject,
handles);
set(handles.pushbutton6, 'Enable', 'on')
```

```
% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject      handle to pushbutton6 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
% Proses menghitung probabilitas

bw = handles.bw;
cc=inbound_tracing(bw);
[kode,~,~]=chain_code(cc);
prob (1,1:8) = 0;
k = 1;
count = 0;
for i = 1 : length(kode)
    Num=str2double(kode(i));
    if (Num<=7) && i<length(kode)
        prob(k,Num+1) = prob(k,Num+1)+1;
        count = count + 1;
    else
        prob(k,1:8) = prob(k,1:8)/count;
        count = 0;
        k = k + 1;
        prob (k,1:8) = 0;
    end
end

set(handles.edit1, 'String',prob(1,1))
set(handles.edit2, 'String',prob(1,2))
set(handles.edit3, 'String',prob(1,3))
set(handles.edit4, 'String',prob(1,4))
set(handles.edit5, 'String',prob(1,5))
set(handles.edit6, 'String',prob(1,6))
set(handles.edit7, 'String',prob(1,7))
set(handles.edit8, 'String',prob(1,8))

load net_keluaran
data_uji = prob(1,:);
hasil_uji = sim(net_keluaran,data_uji);
output = round(hasil_uji);

axes(handles.axes3)

switch output
    case 1
        citra = 'lingkaran';
        imshow('lingkaran.png')
    case 2
        citra = 'segiempat';
        imshow('segiempat.png')
    case 3
        citra = 'segitiga';
```

```
imshow('segitiga.png')
```

```
        otherwise
            citra = 'Unknown';
            imshow('unknown.png')
    end

title(strcat(['Nama Citra Hasil Pencarian: ',citra]))

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject      handle to pushbutton7 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
axes(handles.axes1)
cla reset
set(gca,'XTick',[])
set(gca,'YTick',[])

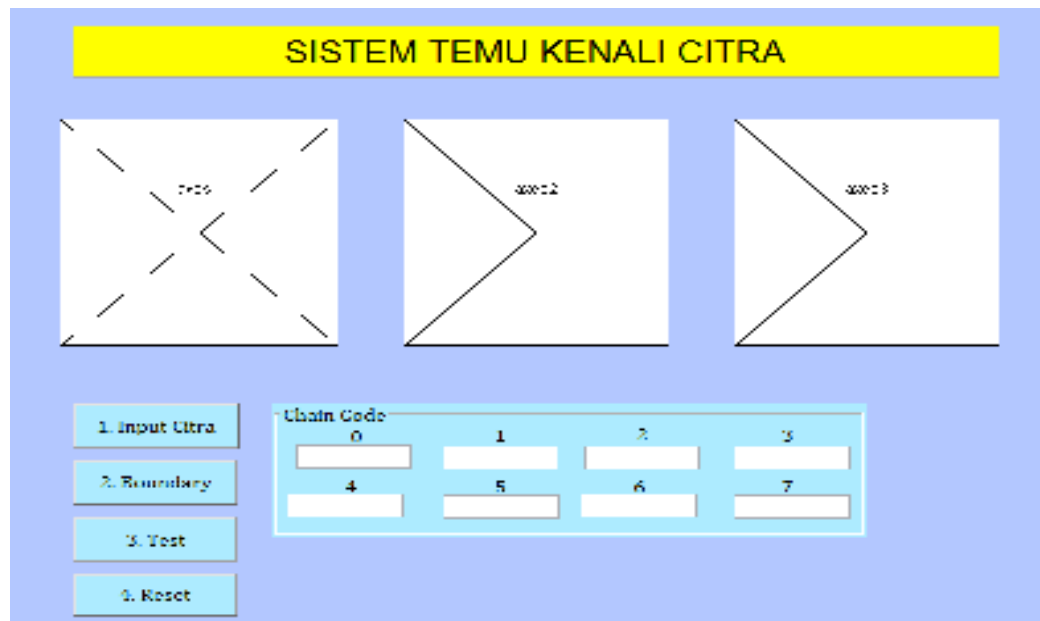
axes(handles.axes2)
cla reset
set(gca,'XTick',[])
set(gca,'YTick',[])

axes(handles.axes3)
cla reset
set(gca,'XTick',[])
set(gca,'YTick',[])

set(handles.edit1,'String','')
set(handles.edit2,'String','')
set(handles.edit3,'String','')
set(handles.edit4,'String','')
set(handles.edit5,'String','')
set(handles.edit6,'String','')
set(handles.edit7,'String','')
set(handles.edit8,'String','')

set(handles.pushbutton5,'Enable','off')
set(handles.pushbutton6,'Enable','off')
```


- Untuk program GUI agar disain sesuai dengan tampilan ini, agar jangan terlalu banyak maka proses di GUI mulai dari Input File Crop – Deteksi Tepi – Ekstraksi – Image Retrieval, jika saudara ada waktu kalau mau membuat GUI dari citra awal sampai akhir disilakan



Terdiri dari :

- Axes1, axes2, axes3, input citra = pushbutton1, Boundary = pushbutton5, Test = pushbutton6, Reset = pushbutton7, 0 = edit1, 1 = edit2, 2 = edit3, 3 = edit4, 4 = edit5, 5 = edit6, 6 = edit7, 7 = edit8,

Cara menjalankannya :

- Klik Input Citra, pilih crop_lingkaran.png
- Klik Boundary
- Klik Tes maka hasil akan muncul di axes3
- Klik Reset untuk menghapus
- Ulangi lagi untuk crop_segiempat, crop_segitiga

Catatan:

Pada kasus diatas pengujian belum menggunakan citra uji tapi masih menggunakan citra latih

Selamat Belajar