```
% start_gui_single_mode.m
% Skrip file untuk menghasilkan GUI

clear
FigWin = figure('Position',[50 -50 650 500],...
   'Name','Sistem Pengenalan dan Pencocokan Sidik Jari -
9922176',...
   'NumberTitle','off',...
   'Color',[ 0.827450980392157 0.815686274509804 0.776470588235294
]);

AxesHandle1 = axes('Position',[0.2 0.15 0.35 0.7],...
   'Box','on');
AxesHandle2 = axes('Position',[0.6 0.15 0.35 0.7],...
   'Box','on');

BackColor = get(gcf,'Color');
%[ 0.827450980392157 0.815686274509804 0.776470588235294 ]

%[ 0.741176470588235 0.725490196078431 0.658823529411765 ]

FrameBox = uicontrol(FigWin,...
   'Units','normalized', ...
   'Style','frame',...
   'BackgroundColor',[ 0.741176470588235 0.725490196078431
0.658823529411765 ],...
   'ForegroundColor',[ 0.741176470588235 0.725490196078431
0.658823529411765 ],...
    'Position',[0 0 0.15 1]);

%buat text diam
Text2 = uicontrol(FigWin,...
   'Style','text',...
   'Units','normalized', ...
   'Position',[0 0.95 1 0.05],...
   'FontSize',15,...
   'BackgroundColor',[ 0.741176470588235 0.725490196078431
0.658823529411765 ],...
   'HorizontalAlignment','right', ...
   'String','Pengenalan Sidik Jari - 9922176');

Text2 = uicontrol(FigWin,...
   'Style','text',...
   'Units','normalized', ...
   'Position',[0 0 1 0.05],...
   'FontSize',15,...
   'BackgroundColor',[ 0.741176470588235 0.725490196078431
0.658823529411765 ],...
   'HorizontalAlignment','right', ...
   'String','Pencocokan Sidik Jari - 9922176');

w=16;
textLoad='Input Sidik Jari';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,320,80,20],...
```

```
   'String','Pilih Sidik Jari',...
   'Callback',...
   ['image1=loadimage;'...
    'subplot(AxesHandle1);'...
    'imagesc(image1);'...
    'title(textLoad);'...
     'colormap(gray);']);

text_filterArea='Perkiraan Aliran Orientasi';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
    'Position',[0,240,80,20],...
    'String','Arah Orientasi',...
    'Callback',...

['subplot(AxesHandle2);[o1Bound,o1Area]=direction(image1,16);title
(text_filterArea);']);

text_ROI='Region Of Interest(ROI)';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
    'Position',[0,220,80,20],...
    'String','Area ROI',...
    'Callback',...

['subplot(AxesHandle2);[o2,o1Bound,o1Area]=drawROI(image1,o1Bound,
o1Area);title(text_ROI);']);

text_eq='Enhancement oleh Histogram Equalization';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,300,80,20],...
   'String','His-Equalization',...
   'Callback',...

['subplot(AxesHandle2);image1=histeq(uint8(image1));imagesc(image1
);title(text_eq);']);

text21='Adaptive Binarization setelah FFT';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,260,80,20],...
   'String','Ubah ke Biner',...
   'Callback',...
   [%'W=inputdlg(text);W=str2num(char(W));'...
      'subplot(AxesHandle1);'...
      'image1=adaptiveThres(double(image1),32);title(text21);']);

text='Masukkan FFT factor(0~1)';
text_fft='Enhancement oleh FFT';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,280,80,20],...
   'String','FFT',...
   'Callback',...
   ['W=inputdlg(text);W=str2double(char(W));'...
```

```
'subplot(AxesHandle1);image1=fftenhance(image1,W);imagesc(image1);
title(text_fft);']);

text31='Penipisan Minutia';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,200,80,20],...
   'String','Penipisan',...
   'Callback',...

['subplot(AxesHandle2);o1=im2double(bwmorph(o2,''thin'',Inf));imag
esc(o1,[0,1]);title(text31);']);

text41='Menghilangkan Patahan H';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,180,80,20],...
   'String','Hilangkan Patahan H',...
    'Callback',...

['subplot(AxesHandle2);o1=im2double(bwmorph(o1,''clean''));o1=im2d
ouble(bwmorph(o1,''hbreak''));imagesc(o1,[0,1]);title(text41);']);

textn1='Menghilangkan Spike';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
   'Position',[0,160,80,20],...
   'String','Hilangkan Spike',...
   'Callback',...

['subplot(AxesHandle2);o1=im2double(bwmorph(o1,''spur''));imagesc(
o1,[0,1]);title(textn1);']);

%% melokasikan minutia dan menampilkan semua minutia tersebut
text51='Minutia';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
    'Position',[0,140,80,20],...
    'String','Ekstraksi',...
    'Callback',...

['[end_list1,branch_list1,ridgeMap1,edgeWidth]=mark_minutia(o1,o1B
ound,o1Area,w);'...

'subplot(AxesHandle2);show_minutia(o1,end_list1,branch_list1);titl
e(text51);']);

%Proses untuk menghilangkan minutia yang palsu
text61='Minutia Asli';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
    'Position',[0,120,80,20],...
    'String','Minutia Asli',...
   'Callback',...
```

```
['[pathMap1,real_end1,real_branch1]=remove_spurious_Minutia(o1,end
_list1,branch_list1,o1Area,ridgeMap1,edgeWidth);'...

'subplot(AxesHandle1);show_minutia(o1,real_end1,real_branch1);titl
e(text61);']);

%simpan template file, termasuk informasi posisi minutia,arah,dan
ridge
textSaveName='Nama File';
h=uicontrol(FigWin,...
   'Style','pushbutton',...
    'Position',[0,100,80,20],...
    'String','Simpan',...
   'Callback',...
   ['W=inputdlg(textSaveName);W=char(W);'...
    'save(W,''real_end1'',''pathMap1'',''-ASCII'');']);

%Meminta pengeluaran template file dan lakukan pencocokan
h=uicontrol('Style','pushbutton',...
   'String','Pencocokan',...
    'Position',[0,80,80,20],...
    'Callback',...
   ['finger1=fingerTemplateRead;'...
    'finger2=fingerTemplateRead;'...
    'percent_match=match_end(finger1,finger2,10);']);
```

```
% load_image.m

function image1=loadimage
% skrip untuk membuka file citra sidik jari

[imagefile1 , pathname]=
uigetfile('*.bmp;*.BMP;*.tif;*.TIF;*.jpg','Open An Fingerprint
image');
if imagefile1 ~= 0
cd(pathname);
image1=readimage(char(imagefile1));
image1=255-double(image1);

end;
```

```
% read_image.m

function b = readimage(w);

% baca citra w
% gunakan citra yang lengkap untuk proses selanjutnya

a=imread(w);
b = double(a);
```

```
% fftenhance.m

function [final]=fftenhance(image,f)

I = 255-double(image);

[w,h] = size(I);
%out = I;

w1=floor(w/32)*32;
h1=floor(h/32)*32;

inner = zeros(w1,h1);

for i=1:32:w1
   for j=1:32:h1
      a=i+31;
      b=j+31;
      F=fft2( I(i:a,j:b) );
      factor=abs(F).^f;
      block = abs(ifft2(F.*factor));

      larv=max(block(:));
      if larv==0
         larv=1;
      end;

      block= block./larv;
      inner(i:a,j:b) = block;
   end;
end;

%out(1:w1,1:h1)=inner*255;
final=inner*255;

%d=max(out(:)); %Cari nilai pixel maksimum di C.
%c=min(out(:));
%figure
%imshow(out,[c d]);

final=histeq(uint8(final));
%final=adaptiveThres2(final,32,0);

%imagesc(final);
%colormap(gray);
```

```
% adaptiveThres.m

function [o] = adaptiveThres(a,W,noShow);
%Adaptive thresholding dilakukan untuk segmentasi citra sidik jari

[w,h] = size(a);
o = zeros(w,h);

%bagi citra sidik jari menjadi blok-blok dengan ukuran W
%melangkah ke w dengan panjang W

for i=1:W:w
for j=1:W:h
mean_thres = 0;

%warna putih adalah ridge -> besar

if i+W-1 <= w & j+W-1 <= h
    mean_thres = mean2(a(i:i+W-1,j:j+W-1));
    %nilai threshold dipilih
      mean_thres = 0.8*mean_thres;
      %sebelum binerisasi
      %ridge berwarna hitam, nilai intensitas kecil -> 1 (ridge
putih)
      %latar dan valley berwarna putih , nilai intensitas besar ->
0 (hitam)
      o(i:i+W-1,j:j+W-1) = a(i:i+W-1,j:j+W-1) < mean_thres;
end;

end;
end;

if nargin == 2
imagesc(o);
colormap(gray);
end;
```

```
%direction.m

function [p,z] = direction(image,blocksize,noShow)
%   Menghitung arah  orientasi lokal pada tiap blok
%  dengan ukuran (blocksize x blocksize)
%
direction(grayScaleFingerprintImage,blocksize,graphicalShowDisable
Flag)
%  hasil p batas ROI
%  hasil z area ROI

%image=adaptiveThres(image,16,0);

[w,h] = size(image);
direct = zeros(w,h);
gradient_times_value = zeros(w,h);
gradient_sq_minus_value = zeros(w,h);
gradient_for_bg_under = zeros(w,h);

W = blocksize;
theta = 0;
sum_value = 1;
bg_certainty = 0;

blockIndex = zeros(ceil(w/W),ceil(h/W));
%directionIndex = zeros(ceil(w/W),ceil(h/W));

times_value = 0;
minus_value = 0;

center = [];

%Catat bahwa sistem koordinat citra adalah
%koordinat-x ke arah bawah dan koordinat-y ke arah kanan

filter_gradient = fspecial('sobel');
%untuk mendapatkan gradient x
I_horizontal = filter2(filter_gradient,image);

%untuk mendapatkan gradient y
filter_gradient = transpose(filter_gradient);
I_vertical = filter2(filter_gradient,image);

gradient_times_value=I_horizontal.*I_vertical;
gradient_sq_minus_value=(I_vertical-
I_horizontal).*(I_vertical+I_horizontal);
gradient_for_bg_under = (I_horizontal.*I_horizontal) +
(I_vertical.*I_vertical);

for i=1:W:w
    for j=1:W:h
      if j+W-1 < h & i+W-1 < w
            times_value = sum(sum(gradient_times_value(i:i+W-1,
j:j+W-1)));
```

```
          minus_value = sum(sum(gradient_sq_minus_value(i:i+W-1,
j:j+W-1)));
          sum_value = sum(sum(gradient_for_bg_under(i:i+W-1,
j:j+W-1)));

          bg_certainty = 0;
          theta = 0;

          if sum_value ~= 0 & times_value ~=0
             %if sum_value ~= 0 & minus_value ~= 0 & times_value
~= 0
             bg_certainty = (times_value*times_value +
minus_value*minus_value)/(W*W*sum_value);

            if bg_certainty > 0.05
             blockIndex(ceil(i/W),ceil(j/W)) = 1;

             %tan_value = atan2(minus_value,2*times_value);
             tan_value = atan2(2*times_value,minus_value);

                theta = (tan_value)/2 ;
                theta = theta+pi/2;
            %now the theta is within [0,pi]

            %directionIndex(ceil(i/W),ceil(j/W)) = theta;
            %center = [center;[round(i + (W-1)/2),round(j + (W-
1)/2),theta,bg_certainty]];
             center = [center;[round(i + (W-1)/2),round(j + (W-
1)/2),theta]];
                end;
        end;
    end;
             times_value = 0;
          minus_value = 0;
          sum_value = 0;

   end;
end;

if nargin == 2
    imagesc(direct);

    hold on;
    [u,v] = pol2cart(center(:,3),8);
   quiver(center(:,2),center(:,1),u,v,0,'g');
   hold off;
end;

x = bwlabel(blockIndex,4);
%map = [0 0 0;jet(3)];
%figure
%imshow(x+1,map,'notruesize')

y = bwmorph(x,'close');
%figure
```

```
%imshow(y,map,'notruesize');

%z adalah daerah dari region of interest (ROI)
%dengan the index format

z = bwmorph(y,'open');
%figure
%imshow(z,map,'notruesize');

%p adalah batas dari ROI

p = bwperim(z);
%figure,
%imshow(p,map,'notruesize');

%directMap = directionIndex;
```

```
% drawROI.m

function [roiImg,roiBound,roiArea] =
drawROI(in,inBound,inArea,noShow)
%
drawROI(grayLevelFingerprintImage,ROIboundMap,ROIareaMap,flagToDis
ableGUI)
%  membangun segi empat ROI untuk masukan citra sidik jari dan
menghasilkan
%  segmentasi cita sidik jari
%  Dengan asumsi hanya satu daerah ROI untuk tiap citra sidik jari

[iw,ih]=size(in);
tmplate = zeros(iw,ih);
[w,h] = size(inArea);
tmp=zeros(iw,ih);
%ceil(iw/16) should = w
%ceil(ih/16) should = h

left = 1;
right = h;
upper = 1;
bottom = w;

le2ri = sum(inBound);
roiColumn = find(le2ri>0);
left = min(roiColumn);
right = max(roiColumn);

tr_bound = inBound';

up2dw=sum(tr_bound);
roiRow = find(up2dw>0);
upper = min(roiRow);
bottom = max(roiRow);

%potong area citra ROI

%show background,bound,innerArea with different gray
intensity:0,100,200

for i = upper:1:bottom
   for j = left:1:right
      if inBound(i,j) == 1
         tmplate(16*i-15:16*i,16*j-15:16*j) = 200;
         tmp(16*i-15:16*i,16*j-15:16*j) = 1;

      elseif inArea(i,j) == 1 & inBound(i,j) ~=1
         tmplate(16*i-15:16*i,16*j-15:16*j) = 100;
         tmp(16*i-15:16*i,16*j-15:16*j) = 1;

      end;
   end;
end;
```

```
in=in.*tmp;

roiImg = in(16*upper-15:16*bottom,16*left-15:16*right);

roiBound = inBound(upper:bottom,left:right);
roiArea = inArea(upper:bottom,left:right);

%area dalam
roiArea = im2double(roiArea) - im2double(roiBound);

if nargin == 3
    colormap(gray);
   imagesc(tmplate);
end;
```

```
% mark_minutia.m

function [end_list,branch_list,ridgeOrderMap,edgeWidth] =
mark_minutia(in, inBound,inArea,block);

[w,h] = size(in);

[ridgeOrderMap,totalRidgeNum] = bwlabel(in);

imageBound = inBound;
imageArea = inArea;
blkSize = block;

%innerArea = im2double(inArea)-im2double(inBound);

edgeWidth = interRidgeWidth(in,inArea,blkSize);

end_list    = [];
branch_list = [];

for n=1:totalRidgeNum
   [m,n] = find(ridgeOrderMap==n);
   b = [m,n];
   ridgeW = size(b,1);

   for x = 1:ridgeW
      i = b(x,1);
      j = b(x,2);

      %if imageArea(ceil(i/blkSize),ceil(j/blkSize)) == 1 &
imageBound(ceil(i/blkSize),ceil(j/blkSize)) ~= 1
if inArea(ceil(i/blkSize),ceil(j/blkSize)) == 1
      neiborNum = 0;
      neiborNum = sum(sum(in(i-1:i+1,j-1:j+1)));
      neiborNum = neiborNum -1;

   if neiborNum == 1
        end_list =[end_list; [i,j]];

   elseif neiborNum == 3
      %jika dua tetangga diantara tiga berhubung secara langsung
      %mungkin ada tiga percabangan yang dihitung didekat 2 sel
tersebut
      tmp=in(i-1:i+1,j-1:j+1);

      tmp(2,2)=0;
      [abr,bbr]=find(tmp==1);
      t=[abr,bbr];

      if isempty(branch_list)
         branch_list = [branch_list;[i,j]];
      else

      for p=1:3
```

```
         cbr=find(branch_list(:,1)==(abr(p)-2+i) &
branch_list(:,2)==(bbr(p)-2+j) );
         if ~isempty(cbr)
            p=4;
            break;
         end;
      end;

      if p==3
         branch_list = [branch_list;[i,j]];
      end;

    end;

    end;

    end;

end;
end;
```

```
% interRidgeWidth.m

function edgeDistance = interRidgeWidth(image,inROI,blocksize)

[w,h] = size(image);

a=sum(inROI);

b=find(a>0);

c=min(b);
d=max(b);
i=round(w/5);
m=0;

for k=1:4
   m=m+sum(image(k*i,16*c:16*d));
end;
e=(64*(d-c))/m;

a=sum(inROI,2);
b=find(a>0);

c=min(b);
d=max(b);

i=round(h/5);
m=0;

for k=1:4
   m=m+sum(image(16*c:16*d,k*i));
end;
m=(64*(d-c))/m;

edgeDistance=round((m+e)/2);
```

```
% remobe_spurious_minutia.m

function [pathMap, final_end,final_branch]
=remove_spurious_Minutia(in,end_list,branch_list,inArea,ridgeOrder
Map,edgeWidth)

[w,h] = size(in);

final_end = [];
final_branch =[];
direct = [];
pathMap = [];

end_list(:,3) = 0;
branch_list(:,3) = 1;

minutiaeList = [end_list;branch_list];
finalList = minutiaeList;
[numberOfMinutia,dummy] = size(minutiaeList);
suspectMinList = [];

for i= 1:numberOfMinutia-1
   for j = i+1:numberOfMinutia
      d =( (minutiaeList(i,1) - minutiaeList(j,1))^2 +
(minutiaeList(i,2)-minutiaeList(j,2))^2)^0.5;

      if d < edgeWidth
         suspectMinList =[suspectMinList;[i,j]];
      end;
   end;
end;

[totalSuspectMin,dummy] = size(suspectMinList);
%totalSuspectMin

for k = 1:totalSuspectMin
   typesum = minutiaeList(suspectMinList(k,1),3) +
minutiaeList(suspectMinList(k,2),3);

   if typesum == 1
      % cabang - akir pasangan
      if
ridgeOrderMap(minutiaeList(suspectMinList(k,1),1),minutiaeList(sus
pectMinList(k,1),2) ) ==
ridgeOrderMap(minutiaeList(suspectMinList(k,2),1),minutiaeList(sus
pectMinList(k,2),2) )
         finalList(suspectMinList(k,1),1:2) = [-1,-1];
          finalList(suspectMinList(k,2),1:2) = [-1,-1];
      end;

   elseif typesum == 2
      % cabang - cabang pasangan
      if
ridgeOrderMap(minutiaeList(suspectMinList(k,1),1),minutiaeList(sus
pectMinList(k,1),2) ) ==
```

```
ridgeOrderMap(minutiaeList(suspectMinList(k,2),1),minutiaeList(sus
pectMinList(k,2),2) )
         finalList(suspectMinList(k,1),1:2) = [-1,-1];
          finalList(suspectMinList(k,2),1:2) = [-1,-1];
      end;

   elseif typesum == 0
      % akhir - akhir pasangan
      a = minutiaeList(suspectMinList(k,1),1:3);
      b = minutiaeList(suspectMinList(k,2),1:3);

      if ridgeOrderMap(a(1),a(2)) ~=  ridgeOrderMap(b(1),b(2))

         [thetaA,pathA,dd,mm] = getLocalTheta(in,a,edgeWidth);
         [thetaB,pathB,dd,mm] = getLocalTheta(in,b,edgeWidth);

         %garis yang tersambung antara dua titik

         thetaC = atan2( (pathA(1,1)-pathB(1,1)), (pathA(1,2) -
pathB(1,2)) );

         angleAB = abs(thetaA-thetaB);
         angleAC = abs(thetaA-thetaC);

         if ( (or(angleAB < pi/3, abs(angleAB -pi)<pi/3 )) &
(or(angleAC < pi/3, abs(angleAC - pi) < pi/3)) )
            finalList(suspectMinList(k,1),1:2) = [-1,-1];
            finalList(suspectMinList(k,2),1:2) = [-1,-1];
         end;

         %hilangkan ridge yang pendek kemudian
      elseif  ridgeOrderMap(a(1),a(2)) ==
ridgeOrderMap(b(1),b(2))
         finalList(suspectMinList(k,1),1:2) = [-1,-1];
         finalList(suspectMinList(k,2),1:2) = [-1,-1];

      end;
   end;
end;

   for k =1:numberOfMinutia
      if finalList(k,1:2) ~= [-1,-1]
         if finalList(k,3) == 0
            [thetak,pathk,dd,mm] =
getLocalTheta(in,finalList(k,:),edgeWidth);
            if size(pathk,1) >= edgeWidth
                final_end=[final_end;[finalList(k,1:2),thetak]];
                [id,dummy] = size(final_end);
                pathk(:,3) = id;
                pathMap = [pathMap;pathk];
            end;
         else

            final_branch=[final_branch;finalList(k,1:2)];
```

```
            [thetak,path1,path2,path3] =
getLocalTheta(in,finalList(k,:),edgeWidth);

            if size(path1,1)>=edgeWidth & size(path2,1)>=edgeWidth
& size(path3,1)>=edgeWidth

            final_end=[final_end;[path1(1,1:2),thetak(1)]];
            [id,dummy] = size(final_end);
            path1(:,3) = id;
            pathMap = [pathMap;path1];

            final_end=[final_end;[path2(1,1:2),thetak(2)]];
            path2(:,3) = id+1;
            pathMap = [pathMap;path2];

            final_end=[final_end;[path3(1,1:2),thetak(3)]];
                path3(:,3) = id+2;
            pathMap = [pathMap;path3];

            end;

         end;
      end;
   end;

   %final_end
   %pathMap
   %edgeWidth
```

```
% getLocalTheta.m

function [theta,paths1,paths2,paths3] =
getLocalTheta(in,start_point,edgeWidth)

paths1 =[];
paths2 =[];
paths3 =[];

           a = start_point;
         pathA = [];
           pathA = a(1,1:2);

         theta = [];

if a(3) == 0
         for p=1:edgeWidth

            [cur,dummy] = size(pathA);
            i = pathA(cur,1);
            j = pathA(cur,2);

            window=in(i-1:i+1,j-1:j+1);

            window(2,2) = 0;

            if cur > 1
               window( 2 - pathA(cur,1) + pathA(cur-1,1) , 2-
pathA(cur,2) + pathA(cur-1,2) ) = 0;
            end;

            [q,r]=find(window);
            b=[q,r];
            [neighbors,dummy]=size(b);

            if neighbors == 1
                pathA(cur+1,1) = b(1,1)-2 + pathA(cur,1);
             pathA(cur+1,2) = b(1,2)-2 + pathA(cur,2);
            else
               break;
            end;
          end;

             [path_length, dddd] = size(pathA);
            paths1 = pathA;

                mean_x = 0;
                mean_y = 0;

               mean_value = sum(pathA);

            mean_x = mean_value(1) / path_length;
                mean_y = mean_value(2) / path_length;
```

```
                theta = atan2( (mean_x - pathA(1,1)),(mean_y -
pathA(1,2)) );

elseif a(3) == 1
            pathA = [];

            total_mx = 0;
            total_my = 0;
            i = a(1);
            j = a(2);

            pathA(1,:) = [i,j];

            window=in(i-1:i+1,j-1:j+1);
            window(2,2) = 0;
            [q,r]=find(window);
            b=[q,r];
            [neighbors,dummy]=size(b);

if neighbors == 3
   for s = 1:3

      pathA(2,1) = b(s,1)-2 + pathA(1,1);
        pathA(2,2) = b(s,2)-2 + pathA(1,2);

        for p = 1:edgeWidth

            [cur,dummy] = size(pathA);
            i = pathA(cur,1);
            j = pathA(cur,2);

            window=in(i-1:i+1,j-1:j+1);
            window(2,2) = 0;

            if cur > 1
               window( 2 - pathA(cur,1) + pathA(cur-1,1) , 2-
pathA(cur,2) + pathA(cur-1,2) ) = 0;
            end;

            [q,r]=find(window);
            c=[q,r];
            [neighbors,dummy]=size(c);

            if neighbors == 1
                pathA(cur+1,1) = c(1,1)-2 + pathA(cur,1);
             pathA(cur+1,2) = c(1,2)-2 + pathA(cur,2);
            else
               break;
            end;
        end;

           [path_length, dddd] = size(pathA);

            mean_x = 0;
            mean_y = 0;
```

```
          mean_value = sum(pathA);

         mean_x = mean_value(1) / path_length;
            mean_y = mean_value(2) / path_length;

         theta = [theta;atan2( (mean_x - pathA(1,1)),(mean_y -
pathA(1,2)) )];

         if s == 1
         paths1 = pathA(2:path_length,:);
        elseif s == 2
        paths2 = pathA(2:path_length,:);
        elseif s == 3
        paths3 = pathA(2:path_length,:);
        end;

         pathA(2:path_length,:) = [];

         %total_mx = total_mx + mean_x - pathA(1,1);
         %total_my = total_my + mean_y - pathA(1,2);

end;
end;

%com_theta = atan2(total_mx,total_my);

%tmp =abs(theta_b - com_theta);
%theta = min(tmp);
%pathA = path_b(find(tmp==theta));


end;
```

```
% show_minutia.m

function show_minutia(image,end_list,branch_list);

%tampilkan citra dari semua titik yang ada di list

%[x,y] = size(end_list);
%imag = zeros(200,200);
%imag = image;
%x
%for i=1:x
%   xx = end_list(i,1);
%   yy = end_list(i,2);
%
%   imag(xx-2:xx+2,yy-2:yy+2) = 1;
%   imag(xx,yy) = 0;
%end;

%[x,y] = size(branch_list);
%for i = 1:x
%   xx = branch_list(i,1);
%   yy = branch_list(i,2);

%   imag(xx-2:xx+2,yy-2:yy+2) = 1;
%   imag(xx-1:xx+1,yy-1:yy+1) = 0;
    %imag(xx,yy) = 0;
%end;

%figure;
colormap(gray);imagesc(image);
hold on;

if ~isempty(end_list)

plot(end_list(:,2),end_list(:,1),'*r');
if size(end_list,2) == 3
   hold on
    [u,v] = pol2cart(end_list(:,3),10);
    quiver(end_list(:,2),end_list(:,1),u,v,0,'g');
end;
end;

if ~isempty(branch_list)
hold on
plot(branch_list(:,2),branch_list(:,1),'+y');
end;
```

```
%fingerTemplateRead.m

function template=fingerTemplateRead
%skrip untuk membuka file template sidik jari

[templatefile , pathname]= uigetfile('*.dat','Open An Fingerprint
template file');
if templatefile ~= 0
cd(pathname);
template=load(char(templatefile));
end;
```

```
% match_end.m

function
[percent_match]=match_end(template1,template2,edgeWidth,noShow)
% MATCH_END Fingerprint Minutia Matcher Based on Ridge Alignment
%    match_end(template1,template2) menyetujui dua template file
%    dan menghasilkan kesamaan maksimum dari dua sidik jari
%       file template disimpan dalam matrix Nx3 dengan format :
%     ------------------------------------------------------------
-------
%       minutia_1_position_x  minutia_2_position_y
minutia_1_orientation
%       ...                   ...                   ...
%     minutia_n_position_x  minutia_n_position_y
minutia_n_orientation
%     ridge_1_point_1_posx    ridge_1_point_1_posy    ridge_ID(1)
%     ...                     ...                     ...
%     ridge_1_point_m_posx    ridge_1_point_m_posy    ridge_ID(1)
%     ridge_2_point_1_posx    ridge_2_point_1_posy    ridge_ID(2)
%     ...                     ...                     ...
%     ridge_n_point_1_posx    ridge_n_point_1_posy    ridge_ID(n)
%     ...                     ...                     ...
%     ridge_n_point_m_posx    ridge_n_point_m_posy    ridge_ID(n)
%
%       n menyatakan jumlah total minutia
%       m diautur ke nilai dari rata-rata inter ridge width
%     ------------------------------------------------------------
------
%    match_end(template1,template2,noShow) juga menyetujui tanda
'noShow' untuk meniadakan
%    pesan yang menampilkan persentase kecocokan. Nilai tsb dapat
diatur ke 0
%    Fungsi ini digunakan untuk beberapa proses.

%   Uraikan file tempalte ke dalam minutia dan ridge matrix secara
terpisah
if or(edgeWidth == 0,isempty(edgeWidth))
   edgeWidth=10;
end;

if or(isempty(template1), isempty(template2))
   percent_match = -1;
else
length1 = size(template1,1);
minu1 = template1(length1,3);
real_end1 = template1(1:minu1,:);
ridgeMap1= template1(minu1+1:length1,:);

length2 = size(template2,1);
minu2 = template2(length2,3);
real_end2 = template2(1:minu2,:);
ridgeMap2= template2(minu2+1:length2,:);

ridgeNum1 = minu1;
minuNum1 = minu1;
ridgeNum2 = minu2;
minuNum2 = minu2;
```

```
max_percent=zeros(1,3);

for k1 = 1:minuNum1
      %minuNum2

      %hitung kesamaan antara ridgeMap1(k1) and ridgeMap(k2)
      %pilih dua minutia yang sekarang sebagai yang asli dan atur
minutia yang lain
      %berdasarkan minutia asli.

      newXY1 = MinuOriginTransRidge(real_end1,k1,ridgeMap1);
   for k2 = 1:minuNum2

      newXY2 = MinuOriginTransRidge(real_end2,k2,ridgeMap2);

      %pilih panjang minumun ridge
      compareL = min(size(newXY1,2),size(newXY2,2));
      %bandingkan kesamaan dari dua ridge
      eachPairP = newXY1(1,1:compareL).*newXY2(1,1:compareL);
      pairPSquare = eachPairP.*eachPairP;
      temp = sum(pairPSquare);

      ridgeSimCoef = 0;

      if temp > 0
      ridgeSimCoef = sum(eachPairP)/( temp^.5 );
      end;

if ridgeSimCoef > 0.8
        %pindahkan semua minutia dalam dua sidik jari berdasarkan
        %referensi dari pasangan minutia
         fullXY1=MinuOrigin_TransAll(real_end1,k1);
         fullXY2=MinuOrigin_TransAll(real_end2,k2);

         minuN1 = size(fullXY1,2);
         minuN2 = size(fullXY2,2);
         xyrange=edgeWidth;
         num_match = 0;

         %jika dua minutia dalam sebuah kotak dengan lebar 20 dan
tinggi 30,
         %mempunyai variasi arah yang kecil pi/3
         %maka nyatakan mereka sebagai pasangan yang cocok

for i=1:minuN1
   for j=1:minuN2
      if (abs(fullXY1(1,i)-fullXY2(1,j))<xyrange &
abs(fullXY1(2,i)-fullXY2(2,j))<xyrange)
         angle = abs(fullXY1(3,i) - fullXY2(3,j) );
         if or (angle < pi/3, abs(angle-pi)<pi/6)
         num_match=num_match+1;
         break;
         end;
```

```
      end;
   end;
end;


% dapatkan nilai kecocokan terbesar
current_match_percent=num_match;
if current_match_percent > max_percent(1,1);
   max_percent(1,1) = current_match_percent;
   max_percent(1,2) = k1;
   max_percent(1,3) = k2;
end;
num_match = 0;

end;
end;
end;

percent_match = max_percent(1,1)*100/minuNum1;
end;

%jika fungsi dipanggil dalam GUI mode, keluarkan kotak pesan
%untuk hasil akhir
if nargin == 3
   text=strcat('Persen Kecocokan adalah
',num2str(percent_match,3),' %');
msgbox(text);
end;
```

```
% MinuOriginTarnsRidge.m

function [newXY] = MinuOriginTransRidge(real_end,k,ridgeMap)
%  MinuOrigin(real_end,k,ridgeMap)
%   atur k-th minutia sebagai yang asli dan sejajarkan arahnya ke
0 (sepanjang x)
%  dan kemudian sesuaikan semua titik ridge yang lain yang
tersambung ke minutia ke
%  sistem koordinat yang baru
%
%  Catat bahwa koordinat sistem dan sudut berbeda :
%  ---------------------->y
%  |\
%  | \
%  |  \
%  |   \
%  |thet\a
%  x
%  nilai posisi ke arah bawah, ke kanan posirif.
%  nilai sudut adalah berlawanan arah jarum jam dari bawah ke atas
di koordinat x sebelah kanan, dalam [0,pi]
%          dan searah jarum jam dari bawah ke atas di koordinat x
sebelah kiri,dalam [0,-pi]

        %buat persamaan transform matrix
        % cos(theta)  -sin(theta)
        % sin(theta)   cos(thea)
        % untuk merotasikan sudut theta

      theta = real_end(k,3);
      if theta <0
        theta1=2*pi+theta;
        end;

        theta1=pi/2-theta;

      rotate_mat=[cos(theta1),-
sin(theta1);sin(theta1),cos(theta1)];

      %lokasikan semua titik ridge yang terhubung ke minutia
      %dan transspose seperti format :
      %x1 x2 x3...
      %y1 y2 y3...
      pathPointForK = find(ridgeMap(:,3)== k);
      toBeTransformedPointSet =
ridgeMap(min(pathPointForK):max(pathPointForK),1:2)';

      %ubah posisi minutia (x,y) ke (0,0)
      %ubah semua titik ridge yang lain menurut basis
      tonyTrickLength = size(toBeTransformedPointSet,2);
      pathStart = real_end(k,1:2)';
      translatedPointSet = toBeTransformedPointSet -
pathStart(:,ones(1,tonyTrickLength));

      %rotasikan kumpulan titik
      newXY = rotate_mat*translatedPointSet;
```

```
% MinuOrigin_TransAll.m

function [newXY] = MinuOrigin_TransAll(real_end,k)
%  MinuOrigin_all(real_end,k)
%   atur k-th minutia sebagai yang asli dan sejajarkan arahnya ke
0 (sepanjang x)
%  dan kemudian sesuaikan semua titik minutia yang lain pada sidik
jari ke
%  asli yang baru
%
%  Lihat juga MinuOrigin
%  Perbedaan antara MinuOrigin and MinuOrigin_all adalah bahwa
orientasi
%  dari setiap minutia juga diatur dengan minutia asli

theta = real_end(k,3);

if theta <0
    theta1=2*pi+theta;
end;

theta1=pi/2-theta;

rotate_mat=[cos(theta1),-
sin(theta1),0;sin(theta1),cos(theta1),0;0,0,1];

      toBeTransformedPointSet = real_end';

      tonyTrickLength = size(toBeTransformedPointSet,2);

      pathStart = real_end(k,:)';

      translatedPointSet = toBeTransformedPointSet -
pathStart(:,ones(1,tonyTrickLength));

      newXY = rotate_mat*translatedPointSet;

      %pastikan arah ada di dalam domain [-pi,pi]

      for i=1:tonyTrickLength
         if or(newXY(3,i)>pi,newXY(3,i)<-pi)
            newXY(3,i) = 2*pi - sign(newXY(3,i))*newXY(3,i);
         end;
      end;
```

Aat.bmp

Adit.bmp

Agil_1.bmp

Agil_2.bmp

Agil_3.bmp

Andi.bmp

Arif_1.bmp

Aryza.bmp

Ayah_1.bmp

Baby_1.bmp

Benny.bmp

Berly.bmp

Cong.bmp

Deden_1.bmp

Dila_1.bmp

Dito_1.bmp

Ela.bmp

Eni.bmp

Fajar.bmp

Gafar.bmp

Gundara.bmp

Gun.bmp

Gusmou.bmp

Hamid.bmp

Hanly.bmp

Hendrik_1.bmp

Heny.bmp

Husen.bmp

Ica.bmp

Ika.bmp

Indah_1.bmp

Ivan_1.bmp

Ivan_2.bmp

Ivan_3.bmp

Jay_1.bmp

Lisna.bmp

Lusi.bmp

Niko.bmp

Novi_1.bmp

Nur_1.bmp

Oni.bmp

Ratih.bmp

Selvi_1.bmp

Sidik_1.bmp

Sidik_1 (1).bmp

Sidik_1 (2).bmp

Sidik_1 (3).bmp

Sidik_1 (4).bmp

Sidik_1 (5).bmp

Sidik_1 (6).bmp

Sidik_1 (7).bmp

Sidik_2.bmp

Sidik_2 (1).bmp

Sidik_2 (2).bmp

Sidik_2 (3).bmp

Sidik_2 (4).bmp

Sidik_2 (5).bmp

Sidik_2 (6).bmp

Sidik_2 (7).bmp

Sidik_3.bmp

Sidik_3 (1).bmp

Sidik_3 (2).bmp

Sidik_3 (3).bmp

Sidik_3 (4).bmp

Sidik_3 (5).bmp

Sidik_3 (6).bmp

Sidik_3 (7).bmp

Sidik_4.bmp

Sidik_4 (1).bmp

Sidik_4 (2).bmp

Sidik_4 (3).bmp

Sidik_4 (4).bmp

Sidik_4 (5).bmp

Sidik_4 (6).bmp

Sidik_4 (7).bmp

Sidik_5.bmp

Sidik_5 (1).bmp

Sidik_5 (2).bmp

Sidik_5 (3).bmp

Sidik_5 (4).bmp

Sidik_5 (5).bmp

Sidik_5 (6).bmp

Sidik_5 (7).bmp

Sidik_6.bmp

Sidik_6 (1).bmp

Sidik_6 (2).bmp

Sidik_6 (3).bmp

Sidik_6 (4).bmp

Sidik_6 (5).bmp

Sidik_6 (6).bmp

Sidik_6 (7).bmp

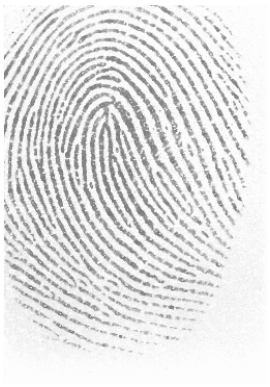
Sidik_7.bmp

Sidik_7 (1).bmp

Sidik_7 (2).bmp

Sidik_7 (3).bmp

Sidik_7 (4).bmp



Sidik_7 (5).bmp

Sidik_7 (6).bmp

Sidik_7 (7).bmp

Sidik_8.bmp

Sidik_8 (1).bmp

Sidik_8 (2).bmp

Sidik_8 (3).bmp

Sidik_8 (4).bmp

Sidik_8 (5).bmp

Sidik_8 (6).bmp

Sidik_8 (7).bmp

Sidik_9.bmp

Sidik_9 (1).bmp

Sidik_9 (2).bmp

Sidik_9 (3).bmp

Sidik_9 (4).bmp

Sidik_9 (5).bmp

Sidik_9 (6).bmp

Sidik_9 (7).bmp

Sidik_10.bmp

Sidik_10 (1).bmp

Sidik_10 (2).bmp

Sidik_10 (3).bmp

Sidik_10 (4).bmp

Sidik_10 (5).bmp

Sidik_10 (6).bmp

Sidik_10 (7).bmp

Yuyun. bmp