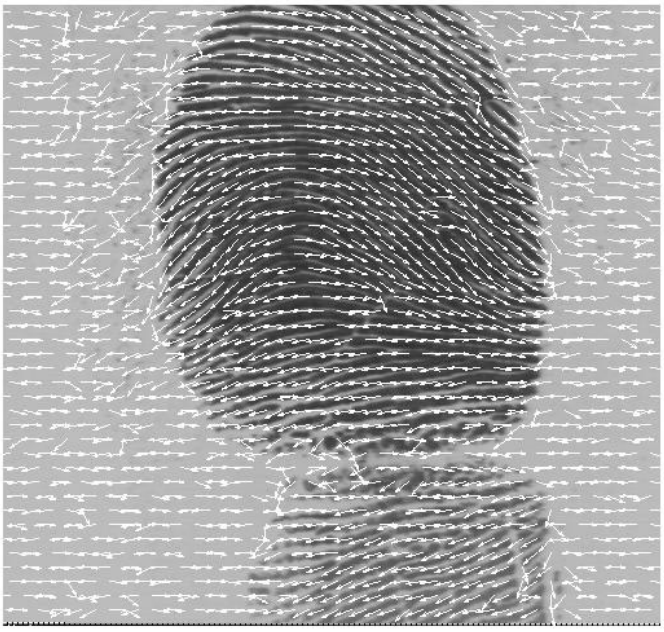
Project 2

CSE 491 - Selected Topics in Biometrics
Instructor: Dr. Arun Ross
Due Date: November 17, 2016
Total Points: 50

Name : Shirley Li
Pid: A47225481

Note:
(a) While you may discuss this project with others, the final submission must be your own effort. Any indication to the contrary will be considered an act of academic dishonesty.
(b) Your zipped code along with the results (labeled as proj02_yourname.zip) should be sent to rossarun at cse.msu.edu with the subject line “cse491: proj02_yourname".
(c) A hard-copy showing the output of your work along with the source code should also be turned in. Failure to submit the hard-copy will result in 0 points for this assignment.

1. [20 points] Based on the algorithm discussed in class write a program to compute the orientation field of a fingerprint image. The orientation should be computed for each pixel location. (So the number of rows and columns in the orientation field matrix should be the same as that of the image). Use the Sobel Operator to compute the x and y gradient value at each pixel location. Use a window size of 9 × 9 when computing the orientation field value associated with a pixel location. Run your program on the set of 10 fingerprint images available here. Store the orientation field of each fingerprint in an image file. Use the drawOrientation.m program to display the orientation field as an overlay on the original fingerprint image. Include these overlay images in your submission.



2. [20 points] Recall that a minutiae set, M, is a set of 3-tupled values M = {(xi, yi,θi)}, i = 1,2...NM, where (xi, yi) is the location of minutiae i, θi is its orientation, and NM is the total number of minutiae in M. Implement the minutiae matching method discussed in class (RANSAC method) that compares two minutiae sets M1 and M2, and outputs the transformation parameters Δx, Δy and Δθ relating M2 with M1, along with the number of matching minutiae pairs.
You are given a set of 10 minutiae files pertaining to 5 different fingers (the orientation value is denoted in “degrees"). Run your program on all possible pairings of the minutiae files and present a table listing theresultsinthefollowingformat:
FirstFileName(M1),SecondFileName(M2),Δx,Δy,Δθ,Number of Matching Minutiae Pairs (s). For example, the first line of the table will be:
user001_1.minpoints user001_2.minpoints -65 -6 0.052360 19

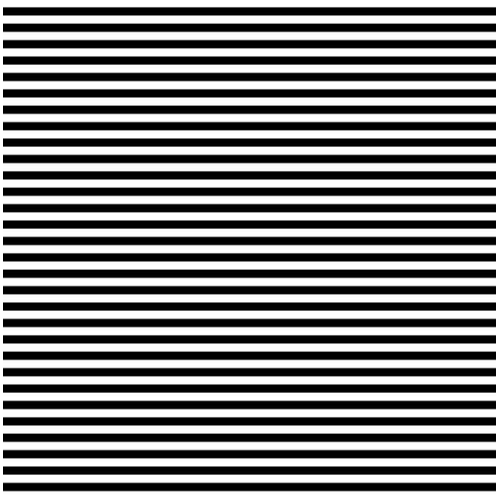
‘M1'	'M 2'	'x'	‘ y'	'theta'	's'
'user001_1.minpoints'	'user001_2.minpoints'	[-65]	[-6]	[0.0524]	[19]
'user001_1.minpoints'	'user002_1.minpoints'	[-195]	[-56]	[5.7596]	[12]
'user001_1.minpoints'	'user002_2.minpoints'	[-29]	[6]	[-1.1519]	[13]
'user001_1.minpoints'	'user003_1.minpoints'	[-96]	[117]	[-2.8972]	[14]
'user001_1.minpoints'	'user003_2.minpoints'	[-117]	[-247]	[4.5553]	[11]
'user001_1.minpoints'	'user004_1.minpoints'	[-241]	[-63]	[4.1364]	[12]
'user001_1.minpoints'	'user004_2.minpoints'	[-36]	[-46]	[-0.4887]	[13]
'user001_1.minpoints'	'user005_1.minpoints'	[65]	[1]	[0.9948]	[10]
'user001_1.minpoints'	'user005_2.minpoints'	[6]	[-241]	[3.2638]	[12]
'user001_2.minpoints'	'user002_1.minpoints'	[-86]	[-196]	[5.3233]	[13]
'user001_2.minpoints'	'user002_2.minpoints'	[-105]	[10]	[5.6549]	[12]
'user001_2.minpoints'	'user003_1.minpoints'	[-112]	[-113]	[-0.3142]	[13]
'user001_2.minpoints'	'user003_2.minpoints'	[39]	[-28]	[-1.5184]	[13]
'user001_2.minpoints'	'user004_1.minpoints'	[-96]	[30]	[-0.1222]	[12]
'user001_2.minpoints'	'user004_2.minpoints'	[28]	[-38]	[-0.4363]	[12]
'user001_2.minpoints'	'user005_1.minpoints'	[-85]	[-305]	[3.0892]	[10]
'user001_2.minpoints'	'user005_2.minpoints'	[-72]	[-56]	[3.5430]	[11]
'user002_1.minpoints'	'user002_2.minpoints'	[15]	[-2]	[0]	[38]
'user002_1.minpoints'	'user003_1.minpoints'	[15]	[-84]	[-0.2443]	[18]
'user002_1.minpoints'	'user003_2.minpoints'	[-9]	[33]	[1.2392]	[16]
'user002_1.minpoints'	'user004_1.minpoints'	[55]	[-76]	[0.7854]	[13]
'user002_1.minpoints'	'user004_2.minpoints'	[63]	[-14]	[-1.1868]	[15]
'user002_1.minpoints'	'user005_1.minpoints'	[125]	[-214]	[4.6600]	[14]
'user002_1.minpoints'	'user005_2.minpoints'	[-10]	[218]	[-1.2043]	[14]
'user002_2.minpoints'	'user003_1.minpoints'	[69]	[45]	[-2.7227]	[18]
'user002_2.minpoints'	'user003_2.minpoints'	[6]	[-94]	[0.0873]	[16]
'user002_2.minpoints'	'user004_1.minpoints'	[12]	[26]	[-4.2935]	[13]
'user002_2.minpoints'	'user004_2.minpoints'	[34]	[17]	[-0.6807]	[17]
'user002_2.minpoints'	'user005_1.minpoints'	[61]	[1]	[-0.0698]	[14]
'user002_2.minpoints'	'user005_2.minpoints'	[22]	[9]	[-5.2534]	[12]
'user003_1.minpoints'	'user003_2.minpoints'	[-1]	[17]	[0.0698]	[27]
'user003_1.minpoints'	'user004_1.minpoints'	[-208]	[-78]	[2.5133]	[16]
'user003_1.minpoints'	'user004_2.minpoints'	[232]	[-6]	[2.9322]	[20]
'user003_1.minpoints'	'user005_1.minpoints'	[259]	[203]	[-2.4435]	[14]
'user003_1.minpoints'	'user005_2.minpoints'	[31]	[86]	[0.5585]	[18]
'user003_2.minpoints'	'user004_1.minpoints'	[162]	[-161]	[-1.3788]	[15]
'user003_2.minpoints'	'user004_2.minpoints'	[142]	[-43]	[-5.6549]	[17]
'user003_2.minpoints'	'user005_1.minpoints'	[-21]	[75]	[-1.7628]	[15]
'user003_2.minpoints'	'user005_2.minpoints'	[54]	[-54]	[4.9044]	[13]
'user004_1.minpoints'	'user004_2.minpoints'	[64]	[-44]	[0]	[30]
'user004_1.minpoints'	'user005_1.minpoints'	[49]	[-3]	[-0.0524]	[11]
'user004_1.minpoints'	'user005_2.minpoints'	[127]	[42]	[1.2566]	[12]
'user004_2.minpoints'	'user005_1.minpoints'	[6]	[-327]	[-3.0020]	[13]
'user004_2.minpoints'	'user005_2.minpoints'	[-40]	[22]	[0.0524]	[13]
'user005_1.minpoints'	'user005_2.minpoints'	[-75]	[97]	[0]	[23]
'user001_1.minpoints'	'user001_2.minpoints'	[-65]	[-6]	[0.0524]	[19]
'user001_1.minpoints'	'user002_1.minpoints'	[-195]	[-56]	[5.7596]	[12]

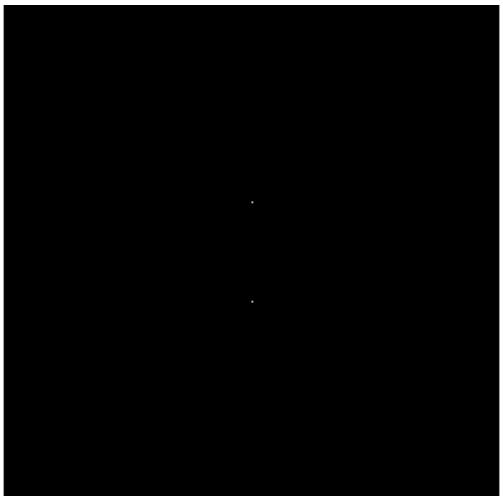
'user001_1.minpoints'	'user002_2.minpoints'	[-29]	[6]	[-1.1519]	[13]
'user001_1.minpoints'	'user003_1.minpoints'	[-96]	[117]	[-2.8972]	[14]
'user001_1.minpoints'	'user003_2.minpoints'	[-117]	[-247]	[4.5553]	[11]
'user001_1.minpoints'	'user004_1.minpoints'	[-241]	[-63]	[4.1364]	[12]
'user001_1.minpoints'	'user004_2.minpoints'	[-36]	[-46]	[-0.4887]	[13]
'user001_1.minpoints'	'user005_1.minpoints'	[65]	[1]	[0.9948]	[10]
'user001_1.minpoints'	'user005_2.minpoints'	[6]	[-241]	[3.2638]	[12]
'user001_2.minpoints'	'user002_1.minpoints'	[-86]	[-196]	[5.3233]	[13]
'user001_2.minpoints'	'user002_2.minpoints'	[-105]	[10]	[5.6549]	[12]
'user001_2.minpoints'	'user003_1.minpoints'	[-112]	[-113]	[-0.3142]	[13]
'user001_2.minpoints'	'user003_2.minpoints'	[39]	[-28]	[-1.5184]	[13]
'user001_2.minpoints'	'user004_1.minpoints'	[-96]	[30]	[-0.1222]	[12]
'user001_2.minpoints'	'user004_2.minpoints'	[28]	[-38]	[-0.4363]	[12]
'user001_2.minpoints'	'user005_1.minpoints'	[-85]	[-305]	[3.0892]	[10]
'user001_2.minpoints'	'user005_2.minpoints'	[-72]	[-56]	[3.5430]	[11]
'user002_1.minpoints'	'user002_2.minpoints'	[15]	[-2]	[0]	[38]
'user002_1.minpoints'	'user003_1.minpoints'	[15]	[-84]	[-0.2443]	[18]
'user002_1.minpoints'	'user003_2.minpoints'	[-9]	[33]	[1.2392]	[16]
'user002_1.minpoints'	'user004_1.minpoints'	[55]	[-76]	[0.7854]	[13]
'user002_1.minpoints'	'user004_2.minpoints'	[63]	[-14]	[-1.1868]	[15]
'user002_1.minpoints'	'user005_1.minpoints'	[125]	[-214]	[4.6600]	[14]
'user002_1.minpoints'	'user005_2.minpoints'	[-10]	[218]	[-1.2043]	[14]
'user002_2.minpoints'	'user003_1.minpoints'	[69]	[45]	[-2.7227]	[18]
'user002_2.minpoints'	'user003_2.minpoints'	[6]	[-94]	[0.0873]	[16]
'user002_2.minpoints'	'user004_1.minpoints'	[12]	[26]	[-4.2935]	[13]
'user002_2.minpoints'	'user004_2.minpoints'	[34]	[17]	[-0.6807]	[17]
'user002_2.minpoints'	'user005_1.minpoints'	[61]	[1]	[-0.0698]	[14]
'user002_2.minpoints'	'user005_2.minpoints'	[22]	[9]	[-5.2534]	[12]
'user003_1.minpoints'	'user003_2.minpoints'	[-1]	[17]	[0.0698]	[27]
'user003_1.minpoints'	'user004_1.minpoints'	[-208]	[-78]	[2.5133]	[16]
'user003_1.minpoints'	'user004_2.minpoints'	[232]	[-6]	[2.9322]	[20]
'user003_1.minpoints'	'user005_1.minpoints'	[259]	[203]	[-2.4435]	[14]
'user003_1.minpoints'	'user005_2.minpoints'	[31]	[86]	[0.5585]	[18]
'user003_2.minpoints'	'user004_1.minpoints'	[162]	[-161]	[-1.3788]	[15]
'user003_2.minpoints'	'user004_2.minpoints'	[142]	[-43]	[-5.6549]	[17]
'user003_2.minpoints'	'user005_1.minpoints'	[-21]	[75]	[-1.7628]	[15]
'user003_2.minpoints'	'user005_2.minpoints'	[54]	[-54]	[4.9044]	[13]
'user004_1.minpoints'	'user004_2.minpoints'	[64]	[-44]	[0]	[30]
'user004_1.minpoints'	'user005_1.minpoints'	[49]	[-3]	[-0.0524]	[11]
'user004_1.minpoints'	'user005_2.minpoints'	[127]	[42]	[1.2566]	[12]
'user004_2.minpoints'	'user005_1.minpoints'	[6]	[-327]	[-3.0020]	[13]
'user004_2.minpoints'	'user005_2.minpoints'	[-40]	[22]	[0.0524]	[13]
'user005_1.minpoints'	'user005_2.minpoints'	[-75]	[97]	[0]	[23]

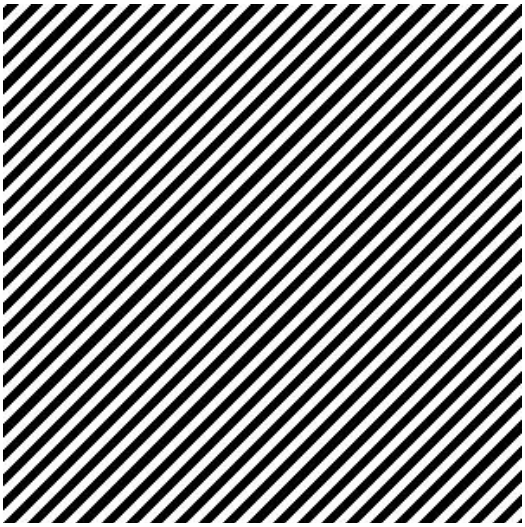
3. [10 points] The ridge pattern in a local area of a finger can be approximated by a cosine wave:
$w(x, y) = A\cos[2\pi f0(x \cos\theta + y \sin\theta)]$.
Here, w(x, y) denotes the pixel intensity at location (x, y). Generate and display ridge patterns, each of size 300 × 300, at the following orientation (θ) values: 0○ , 45○ , 90○ , 135○ . You may set A = 128 and

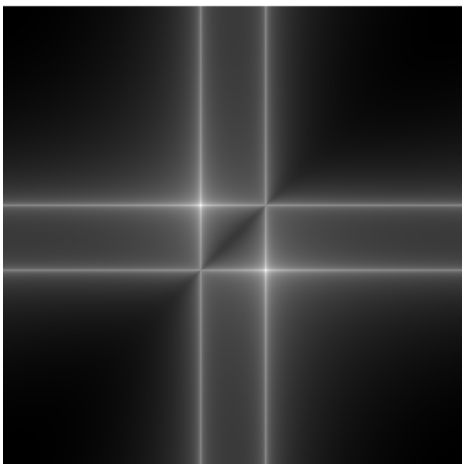
f0 = 0.1. For each ridge pattern, also plot the normalized magnitude of the Fast Fourier Transform next to it.

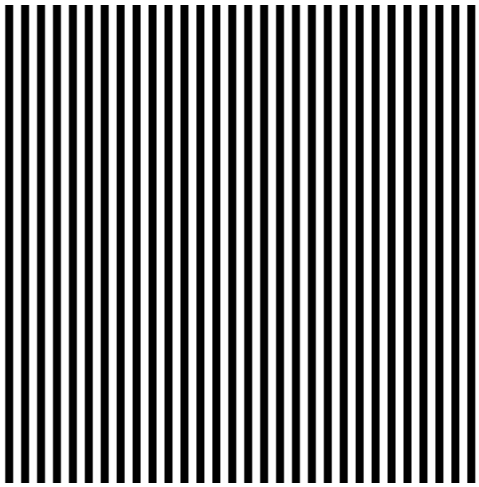
```
function[] = ridgepattern(theta)

for i=1:length(theta)
   theta(i)
   t = deg2rad(theta(i));
   for x=1:300
      for y=1:300
         image(x,y) =  128 * cos( 2*pi*0.1*(x*cos(t) + y*sin(t)) );
      end
   end

figure;
imshow (image(x,y));

fft_img = fft2(image);
imshow (fftshift(log(abs(fft_img)+1)));

   end
end
```

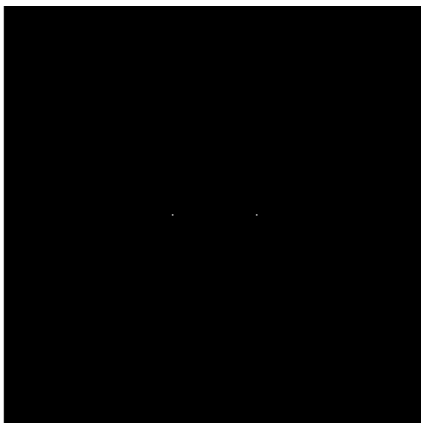


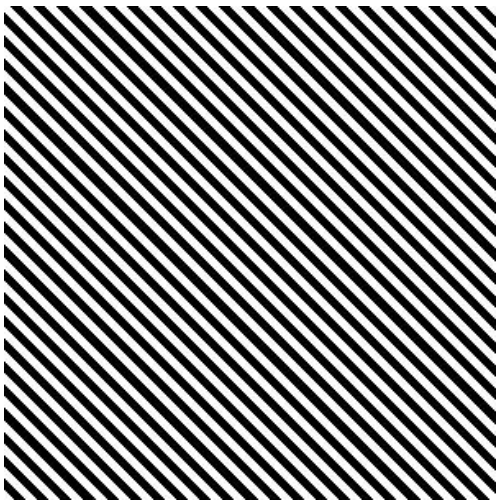


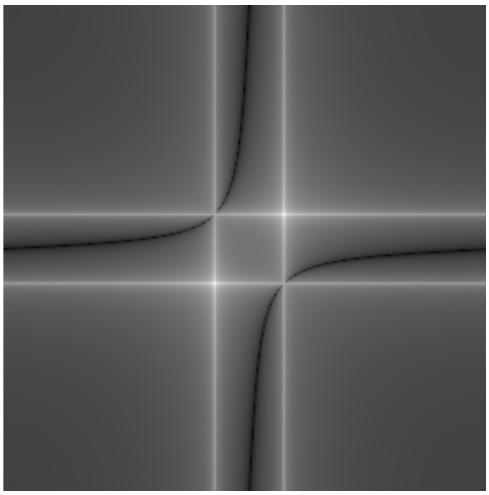












```
images = {'user001_1.gif';'user002_1.gif';'user003_1.gif';'user004_1.gif';
    'user005_1.gif';'user006_1.gif';'user007_1.gif';'user008_1.gif';
    'user009_1.gif';'user010_1.gif'};

size = length(images);

for i= 1: size
    img = imread(char(images(i)));
    ofield(img);
end



function drawOrientation(img, ofield, varargin)
%% Call this function as drawOrientation(img, ofield)
%% where 'img' is the image matrix and 'ofield' is the
%% orientation field matrix. This function displays
%% 'ofield' as a set of quivers on image 'img'.
%%
%% Author: Arun Ross
%% Last Modified: 10 Oct 2006

if (nargin==2)
    blksz = 11;
else
    blksz = varargin{1};
end

hblksz = round(blksz/2);
r = hblksz;

[nr nc] = size(ofield);
u_ofield = r*cos(ofield);
v_ofield = r*sin(ofield);

[X, Y] = meshgrid(hblksz:blksz:nr-hblksz, hblksz:blksz:nc-hblksz);
X = X(:);
Y = Y(:);
for i=1:size(X)
    U(i) = u_ofield(X(i), Y(i));
    V(i) = v_ofield(X(i), Y(i));
end
figure;
imshow(img,[]);
hold on;
```

```
h=quiver(Y, X, V', U');
set(h,'Color',[1 1 1]);

function [img] = ofield( image )

sobel_x = [-1 0 1;-2 0 2;-1 0 1];
sobel_y= fspecial('sobel');
sobel_x_img = imfilter(double(image),sobel_x);
sobel_y_img = imfilter(double(image),sobel_y);
size = size(image);

height = size(1,1) - 9;
width = size(1,2)- 9;
of= zeros(size);

for i = 1:(height)
   for j = 1:(width)
      sobel_x_of= sobel_x(i:(i+8),j:(j+8));
      sobel_y_of= sobel_y(i:(i+8),j:(j+8));
      theta = th(sobel_x_of,sobel_y_of);
      of(i+4,j+4) = theta;
   end
end

drawOrientation(image,of);

end

function [ theta ] = th(x,y )
numerator = 0;
denominator = 0;

for i=1:9
   for j=1:9
      numerator = numerator+(2*x(i,j)*y(i,j));
      denominator = denominator + (x.^2(i,j)-y.^(i,j));
   end
   end
theta = (atan2(numerator,denominator)) * 0.5 ;

end




function[] = ridgepattern(theta)
```

```
for i=1:length(theta)
   theta(i)
   t = deg2rad(theta(i));
   for x=1:300
      for y=1:300
         image(x,y) =  128 * cos( 2*pi*0.1*(x*cos(t) + y*sin(t)) );
      end
   end

figure;
imshow (image(x,y));

fft_img = fft2(image);
imshow (fftshift(log(abs(fft_img)+1)));

   end
end
```