Manotej Meka

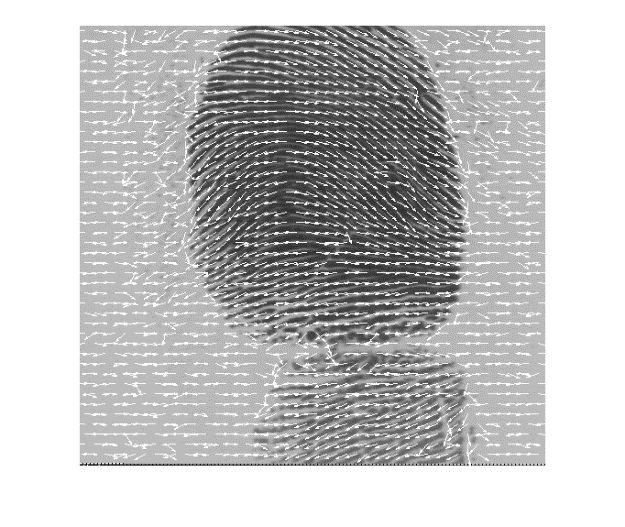
CSE 491 – Biometrics

Homework 3

1) Orientation Fields

 Image 1 Image 2 Image 3



 Image 4 Image 5 Image 6

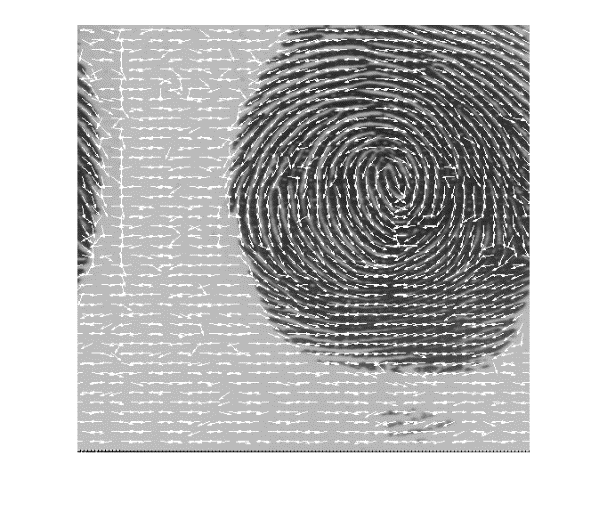


Image 7 Image 8

Image 9 Image 10



Code:

% Question 1 Orientation Filed

imageArray = {'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'};

sizeArray = size(imageArray);

sizeArray = sizeArray(1,1);

for i= 1:sizeArray

Image = imread(char(imageArray(i)));

ImageOrientationPicture(Image);

end

function [ Ori\_Image ] = ImageOrientationPicture( Image )

%UNTITLED2 Summary of this function goes here

% Detailed explanation goes here

Image = double(Image);

sobel\_x\_filter = [-1 0 1;-2 0 2;-1 0 1];

sobel\_y\_filter = fspecial('sobel');

sobel\_x\_img = imfilter(Image,sobel\_x\_filter);

sobel\_y\_img = imfilter(Image,sobel\_y\_filter);

size\_img = size(Image);

row = size\_img(1,1);

column = size\_img(1,2);

row = row - 9;

column = column - 9;

orientation\_array = zeros(size\_img);

for i = 1:(row)

for j = 1:(column)

sobel\_x\_t = sobel\_x\_img(i:(i+8),j:(j+8));

sobel\_y\_t = sobel\_y\_img(i:(i+8),j:(j+8));

theta = Orientation(sobel\_x\_t,sobel\_y\_t);

orientation\_array(i+4,j+4) = theta;

end

end

drawOrientation(Image,orientation\_array);

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]);

2) RANSAC Algorithm

Table In the Next page

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **'File 1'** | **'File 2'** | **'Delta X'** | **'Delta Y'** | **'Theta'** | **'Total'** |
| 'user001\_1.minpoints' | 'user001\_2.minpoints' | -65 | -6 | 0.052359877559829 | 19 |
| 'user001\_1.minpoints' | 'user002\_1.minpoints' | -195 | -56 | 5.75958653158129 | 12 |
| 'user001\_1.minpoints' | 'user002\_2.minpoints' | -29 | 6 | -1.15191730631626 | 13 |
| 'user001\_1.minpoints' | 'user003\_1.minpoints' | -96 | 117 | -2.89724655831059 | 14 |
| 'user001\_1.minpoints' | 'user003\_2.minpoints' | -117 | -247 | 4.55530934770520 | 11 |
| 'user001\_1.minpoints' | 'user004\_1.minpoints' | -241 | -63 | 4.13643032722656 | 12 |
| 'user001\_1.minpoints' | 'user004\_2.minpoints' | -36 | -46 | -0.48869219055841 | 13 |
| 'user001\_1.minpoints' | 'user005\_1.minpoints' | 65 | 1 | 0.994837673636768 | 10 |
| 'user001\_1.minpoints' | 'user005\_2.minpoints' | 6 | -241 | 3.26376570122940 | 12 |
| 'user001\_2.minpoints' | 'user002\_1.minpoints' | -86 | -196 | 5.32325421858271 | 13 |
| 'user001\_2.minpoints' | 'user002\_2.minpoints' | -105 | 10 | 5.65486677646163 | 12 |
| 'user001\_2.minpoints' | 'user003\_1.minpoints' | -112 | -113 | -0.31415926535897 | 13 |
| 'user001\_2.minpoints' | 'user003\_2.minpoints' | 39 | -28 | -1.51843644923507 | 13 |
| 'user001\_2.minpoints' | 'user004\_1.minpoints' | -96 | 30 | -0.12217304763960 | 12 |
| 'user001\_2.minpoints' | 'user004\_2.minpoints' | 28 | -38 | -0.43633231299858 | 12 |
| 'user001\_2.minpoints' | 'user005\_1.minpoints' | -85 | -305 | 3.08923277602996 | 10 |
| 'user001\_2.minpoints' | 'user005\_2.minpoints' | -72 | -56 | 3.54301838154849 | 11 |
| 'user002\_1.minpoints' | 'user002\_2.minpoints' | 15 | -2 | 0 | 38 |
| 'user002\_1.minpoints' | 'user003\_1.minpoints' | 15 | -84 | -0.24434609527920 | 18 |
| 'user002\_1.minpoints' | 'user003\_2.minpoints' | -9 | 33 | 1.23918376891597 | 16 |
| 'user002\_1.minpoints' | 'user004\_1.minpoints' | 55 | -76 | 0.785398163397448 | 13 |
| 'user002\_1.minpoints' | 'user004\_2.minpoints' | 63 | -14 | -1.18682389135614 | 15 |
| 'user002\_1.minpoints' | 'user005\_1.minpoints' | 125 | -214 | 4.66002910282486 | 14 |
| 'user002\_1.minpoints' | 'user005\_2.minpoints' | -10 | 218 | -1.20427718387609 | 14 |
| 'user002\_2.minpoints' | 'user003\_1.minpoints' | 69 | 45 | -2.72271363311115 | 18 |
| 'user002\_2.minpoints' | 'user003\_2.minpoints' | 6 | -94 | 0.087266462599716 | 16 |
| 'user002\_2.minpoints' | 'user004\_1.minpoints' | 12 | 26 | -4.29350995990605 | 13 |
| 'user002\_2.minpoints' | 'user004\_2.minpoints' | 34 | 17 | -0.68067840827778 | 17 |
| 'user002\_2.minpoints' | 'user005\_1.minpoints' | 61 | 1 | -0.06981317007977 | 14 |
| 'user002\_2.minpoints' | 'user005\_2.minpoints' | 22 | 9 | -5.25344104850293 | 12 |
| 'user003\_1.minpoints' | 'user003\_2.minpoints' | -1 | 17 | 0.069813170079773 | 27 |
| 'user003\_1.minpoints' | 'user004\_1.minpoints' | -208 | -78 | 2.51327412287183 | 16 |
| 'user003\_1.minpoints' | 'user004\_2.minpoints' | 232 | -6 | 2.93215314335047 | 20 |
| 'user003\_1.minpoints' | 'user005\_1.minpoints' | 259 | 203 | -2.44346095279206 | 14 |
| 'user003\_1.minpoints' | 'user005\_2.minpoints' | 31 | 86 | 0.558505360638186 | 18 |
| 'user003\_2.minpoints' | 'user004\_1.minpoints' | 162 | -161 | -1.37881010907552 | 15 |
| 'user003\_2.minpoints' | 'user004\_2.minpoints' | 142 | -43 | -5.65486677646163 | 17 |
| 'user003\_2.minpoints' | 'user005\_1.minpoints' | -21 | 75 | -1.76278254451427 | 15 |
| 'user003\_2.minpoints' | 'user005\_2.minpoints' | 54 | -54 | 4.90437519810407 | 13 |
| 'user004\_1.minpoints' | 'user004\_2.minpoints' | 64 | -44 | 0 | 30 |
| 'user004\_1.minpoints' | 'user005\_1.minpoints' | 49 | -3 | -0.05235987755982 | 11 |
| 'user004\_1.minpoints' | 'user005\_2.minpoints' | 127 | 42 | 1.25663706143592 | 12 |
| 'user004\_2.minpoints' | 'user005\_1.minpoints' | 6 | -327 | -3.00196631343025 | 13 |
| 'user004\_2.minpoints' | 'user005\_2.minpoints' | -40 | 22 | 0.052359877559829 | 13 |
| 'user005\_1.minpoints' | 'user005\_2.minpoints' | -75 | 97 | 0 | 23 |

Code:

% RANSAC Algorithm

minutiaArray = {'user001\_1.minpoints','user001\_2.minpoints','user002\_1.minpoints',...

'user002\_2.minpoints','user003\_1.minpoints','user003\_2.minpoints','user004\_1.minpoints',...

'user004\_2.minpoints','user005\_1.minpoints','user005\_2.minpoints'};

lenMin = size(minutiaArray);

lenMin = lenMin(1,2);

valueTable = {'File 1', 'File 2', 'Delta X', 'Delta Y', 'Theta', 'Total'};

for i = 1:lenMin

for j = i+1: lenMin

file1 = char(minutiaArray(1,i));

file2 = char(minutiaArray(1,j));

values = RANSAC(file1,file2);

valueTable = [valueTable;values];

%fprintf('i: %d, j: %d\n',i,j);

end

j = j + 1;

end

disp('The Results are: \n');

disp(valueTable);

function [ ans] = RANSAC( f1, f2 )

%UNTITLED Summary of this function goes here

% Detailed explanation goes here

arrayM = OpenFile(f1);

arrayN = OpenFile(f2);

lenM = length(arrayM);

lenN = length(arrayN);

Pprime = arrayM;

size = lenM \* lenN;

Cor = zeros(size,10);

pos = 1;

for i = 1:lenM

for j = 1:lenN

Px = arrayM(i,1);

Py = arrayM(i,2);

Po = arrayM(i,3);

Qx = arrayN(j,1);

Qy = arrayN(j,2);

Qo = arrayN(j,3);

changeX = Qx - Px;

changeY = Qy - Py;

changeT = Qo - Po;

changeT = deg2rad(changeT);

for k = 1:lenM

% X Prime values

Pprime(k,1) = (arrayM(k,1)-Px)\*cos(changeT) +...

(arrayM(k,2)-Py)\*sin(changeT)+Px+changeX;

Pprime(k,2) = -(arrayM(k,1)-Px)\*sin(changeT) +...

(arrayM(k,2)-Py)\*cos(changeT)+Py+changeY;

end

% total count

count = 0;

% Tables values

Cor(pos,1) = Px;

Cor(pos,2) = Py;

Cor(pos,3) = Qx;

Cor(pos,4) = Qy;

Cor(pos,5) = changeX;

Cor(pos,6) = changeY;

Cor(pos,7) = changeT;

sizeP = length(Pprime);

for m = 1:lenN

% Radius values

tolerance = 11;

% Only one in the small box

onevalue = -1;

distance = 0;

index = 0;

for n = 1:sizeP

x1 = arrayN(m,1);

x2 = Pprime(n,1);

y1 = arrayN(m,2);

y2 = Pprime(n,2);

distance = sqrt((x2 - x1)^2 + (y2 - y1)^2);

if distance < tolerance && onevalue ~= -1 && ...

distance < onevalue

onevalue = distance;

index = n;

elseif distance < tolerance

onevalue = distance;

index = n;

end

end

Cor(pos,8) = distance;

if onevalue ~= -1

Pprime(index,:) = [];

sizeP = sizeP - 1;

count = count + 1;

end

Cor(pos,9) = count;

Cor(pos,10) = (count^2)/size \* 100;

end

pos = pos + 1;

end

end

CorSort = sortrows(Cor,9);

delX = CorSort(size,5);

delY = CorSort(size,6);

delT = CorSort(size,7);

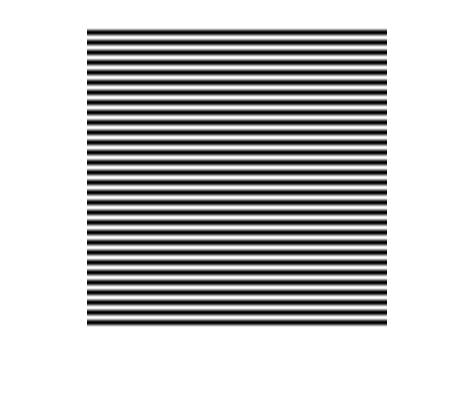
num = CorSort(size,9);

ans = {f1, f2, delX, delY, delT, num};

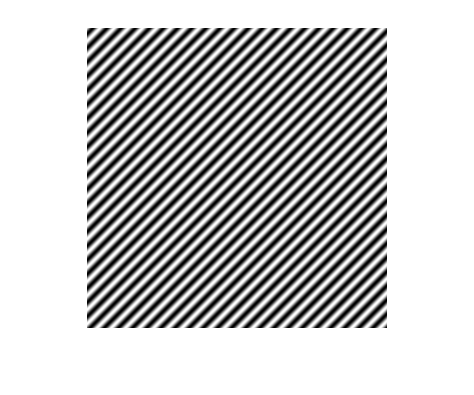
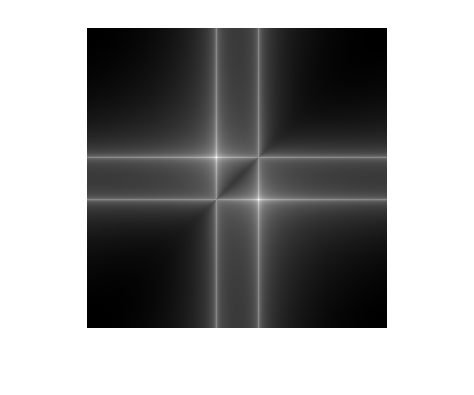
end

3) FFT and Ridges

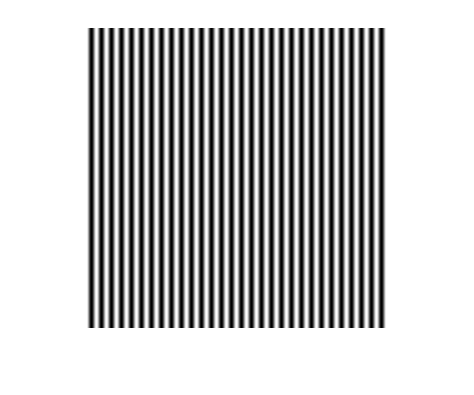
@ 0



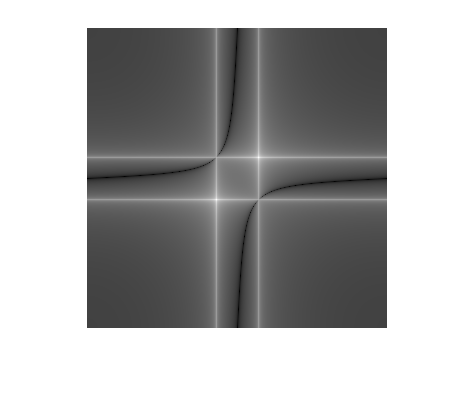
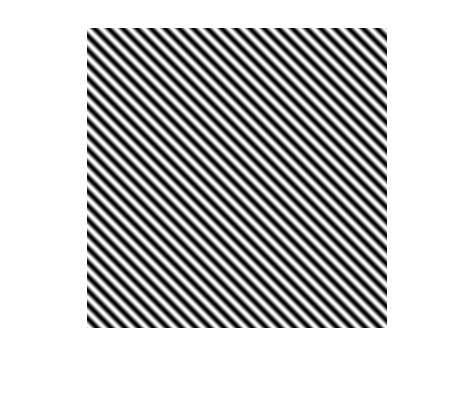
@ 45



@ 90



@ 135



Code:

% Ridge pattern

angles = [0, 45, 90, 135];

lenAng = length(angles);

for z = 1:lenAng

Ridge = zeros(300,300);

for x = 1:300

for y = 1:300

Ridge(x,y) = RidgePattern(x,y,deg2rad(angles(z)));

end

end

figure;

imshow(Ridge,[]);

figure;

fft\_imgh = fft2(Ridge);

imshow(fftshift(log(abs(fft\_imgh) + 1)), []);

end

function [ w ] = RidgePattern( x, y, theta )

%UNTITLED Summary of this function goes here

% Detailed explanation goes here

w = 128\*cos(2\*pi\*0.1\*(x\*cos(theta) + y\*sin(theta)));

end