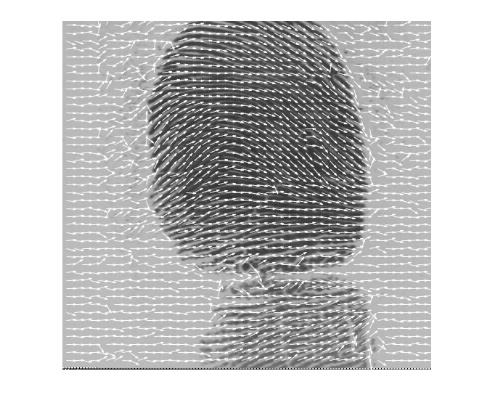
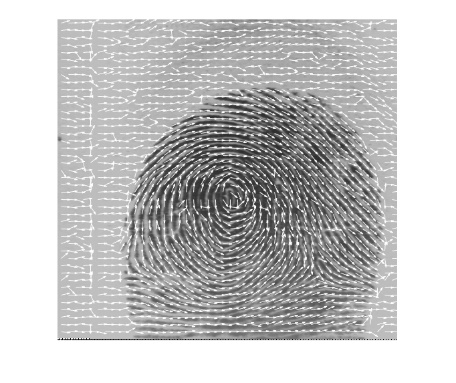
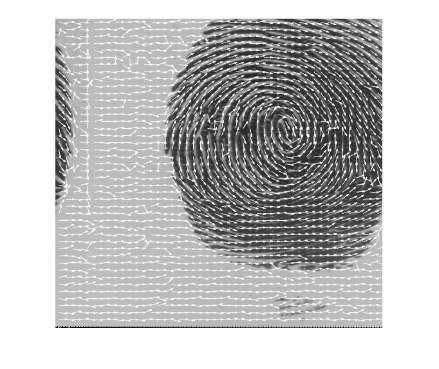
Project 02

CSE 402

Kunyu Chen

1) question 1



Code:

Question1:

listings = {'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'};

for i = 1:10

image = imread(char(listings(i)));

ImageOrientation(image);

end

function [ theta ] = Orientation(SobelX, SobelY)

numerator = 0;

denominator = 0;

pi=3.1415926;

for i=1:9

for j=1:9

numerator = numerator + ( 2\*SobelX(i,j)\*SobelY(i,j));

denominator = denominator + ((SobelX(i,j))^2-(SobelY(i,j))^2);

end

end

theta =(1/2)\*atan2(numerator,denominator)+pi/2; %function

end

function ImageOrientation(Image )

Image = double(Image);

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

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

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

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

size\_img = size(Image);

array = zeros(size\_img);

for i = 1:(size\_img(1,1)-9) %row

for j = 1:(size\_img(1,2)-9) %column

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

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

array(i+4,j+4) = Orientation(sobelX,sobelY);

end

end

drawOrientation(Image,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) question 2

Results:

'File 1' 'File 2' 'tx' 'ty' 'theta' 'matching pairs'

'user001\_1.minpoints' 'user001\_2.minpoints' [ -65] [ -6] [ 0.0524] [ 19]

'user001\_1.minpoints' 'user002\_1.minpoints' [-192] [ -80] [ 5.3756] [ 9]

'user001\_1.minpoints' 'user002\_2.minpoints' [ -85] [ -49] [-0.1396] [ 10]

'user001\_1.minpoints' 'user003\_1.minpoints' [ 50] [-294] [ 3.2987] [ 13]

'user001\_1.minpoints' 'user003\_2.minpoints' [-246] [-187] [ 4.2586] [ 10]

'user001\_1.minpoints' 'user004\_1.minpoints' [-241] [ -63] [ 4.1364] [ 11]

'user001\_1.minpoints' 'user004\_2.minpoints' [ -36] [ -46] [-0.4887] [ 12]

'user001\_1.minpoints' 'user005\_1.minpoints' [ 134] [ 5] [-3.0369] [ 9]

'user001\_1.minpoints' 'user005\_2.minpoints' [ -91] [ 21] [ 0.1222] [ 11]

'user001\_2.minpoints' 'user002\_1.minpoints' [ -86] [-196] [ 5.3233] [ 12]

'user001\_2.minpoints' 'user002\_2.minpoints' [-105] [ 10] [ 5.6549] [ 10]

'user001\_2.minpoints' 'user003\_1.minpoints' [-112] [-113] [-0.3142] [ 12]

'user001\_2.minpoints' 'user003\_2.minpoints' [ 39] [ -28] [-1.5184] [ 11]

'user001\_2.minpoints' 'user004\_1.minpoints' [ -96] [ 30] [-0.1222] [ 11]

'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' [ -27] [ 30] [-0.1396] [ 11]

'user002\_1.minpoints' 'user002\_2.minpoints' [ 15] [ -2] [ 0] [ 38]

'user002\_1.minpoints' 'user003\_1.minpoints' [ 15] [ -84] [-0.2443] [ 16]

'user002\_1.minpoints' 'user003\_2.minpoints' [ -9] [ 33] [ 1.2392] [ 15]

'user002\_1.minpoints' 'user004\_1.minpoints' [ 16] [ 18] [-0.8552] [ 12]

'user002\_1.minpoints' 'user004\_2.minpoints' [ 21] [ -24] [-3.8746] [ 15]

'user002\_1.minpoints' 'user005\_1.minpoints' [ 178] [ -23] [-0.3316] [ 13]

'user002\_1.minpoints' 'user005\_2.minpoints' [ 36] [ 30] [-1.2043] [ 13]

'user002\_2.minpoints' 'user003\_1.minpoints' [ 121] [ -3] [ 1.7628] [ 16]

'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] [ 16]

'user002\_2.minpoints' 'user005\_1.minpoints' [ 21] [ -96] [-1.5708] [ 12]

'user002\_2.minpoints' 'user005\_2.minpoints' [ 82] [ 55] [ 0.6632] [ 11]

'user003\_1.minpoints' 'user003\_2.minpoints' [ -1] [ 17] [ 0.0698] [ 26]

'user003\_1.minpoints' 'user004\_1.minpoints' [ 104] [-146] [ 5.1487] [ 13]

'user003\_1.minpoints' 'user004\_2.minpoints' [ 232] [ -6] [ 2.9322] [ 17]

'user003\_1.minpoints' 'user005\_1.minpoints' [ 259] [ 203] [-2.4435] [ 13]

'user003\_1.minpoints' 'user005\_2.minpoints' [ 31] [ 86] [ 0.5585] [ 17]

'user003\_2.minpoints' 'user004\_1.minpoints' [ 162] [-161] [-1.3788] [ 13]

'user003\_2.minpoints' 'user004\_2.minpoints' [ -80] [-142] [ 2.2515] [ 16]

'user003\_2.minpoints' 'user005\_1.minpoints' [ -21] [ 75] [-1.7628] [ 14]

'user003\_2.minpoints' 'user005\_2.minpoints' [ 40] [ 94] [-0.1222] [ 12]

'user004\_1.minpoints' 'user004\_2.minpoints' [ 64] [ -44] [ 0] [ 29]

'user004\_1.minpoints' 'user005\_1.minpoints' [ 264] [ 86] [-2.9147] [ 11]

'user004\_1.minpoints' 'user005\_2.minpoints' [ -94] [ -51] [ 4.3284] [ 10]

'user004\_2.minpoints' 'user005\_1.minpoints' [ -49] [ 163] [-4.3110] [ 13]

'user004\_2.minpoints' 'user005\_2.minpoints' [ -92] [ 96] [ 0.3665] [ 13]

'user005\_1.minpoints' 'user005\_2.minpoints' [ -75] [ 97] [ 0] [ 23]

Code:

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'};

Table = {'File 1', 'File 2', 'tx', 'ty', 'theta', 'matching pairs'};

for i = 1:10

for j = i+1: 10

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

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

values = RANSAC(file1,file2);

Table = [Table;values];

end

j = j + 1;

end

disp('Results:');

disp(Table);

function [values] = RANSAC( f1, f2 )

arrayM = load(f1);

arrayN = load(f2);

lenM = length(arrayM);

lenN = length(arrayN);

prime = arrayM;

size = lenM \* lenN;

array = zeros(size,0);

position = 1;

for i = 1:lenM

for j = 1:lenN

Px = arrayM(i,1); %xj

Py = arrayM(i,2); %yj

Pt = arrayM(i,3); %thetaj

Qx = arrayN(j,1); %xi

Qy = arrayN(j,2); %yi

Qt = arrayN(j,3); %thetai

deltaX = Qx - Px;

deltaY = Qy - Py;

THETA = Qt - Pt;

Theta = deg2rad(THETA);

for k = 1:lenM

prime(k,1) = (arrayM(k,1)-Px)\*cos(Theta) +(arrayM(k,2)-Py)\*sin(Theta)+Px+deltaX;

prime(k,2) = -(arrayM(k,1)-Px)\*sin(Theta) +(arrayM(k,2)-Py)\*cos(Theta)+Py+deltaY;

end

% total count

number\_pairs = 0;

array(position,1) = deltaX;

array(position,2) = deltaY;

array(position,3) = Theta;

sizeprime = length(prime);

for m = 1:lenN

% check when it is matching minutiae pairs

tolerance = 10;

onevalue = -4;

distance = 0;

index = 0;

for n = 1:sizeprime

x1 = arrayN(m,1);

x2 = prime(n,1);

y1 = arrayN(m,2);

y2 = prime(n,2);

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

%if it is true, it is matching minutiae pairs

if distance < tolerance

onevalue = distance;

index = n;

end

end

array(position,4) = distance;

%when distance changed, it is matching minutiae pairs

if onevalue ~= -4

prime(index,:) = [];

sizeprime = sizeprime - 1;

%number of pairs add 1

number\_pairs = number\_pairs + 1;

end

array(position,5) = number\_pairs;

end

position = position + 1;

end

end

%sort the matrix

arraySort = sortrows(array,5);

tx = arraySort(size,1);

ty = arraySort(size,2);

theta = arraySort(size,3);

count = arraySort(size,5);

values = {f1, f2, tx, ty, theta, count};

end

3) question 3

Left middle finger



Left index finger



Left ring finger



Left pinky finger



Right ring finger



Right pinky finger



Right middle finger



Right index finger



(a) I put graphite on my finger and use tape to get the fingerprint.

(c) I made the image contrast much higher and lower the brightness as well in photoshop. That it.