The University of Calgary
Department of Electrical and Computer Engineering

ENCM 509 - Fundamentals of Biometric Systems Design

Laboratory Experiment #4

Fingerprint Biometrics part I: Preprocessing and feature extraction

1 Introduction

The purpose of this laboratory exercise is to investigate the accuracy of the data acquisition process. In automatic fingerprint acquisition, the sensor can introduce distortion; the dryness of the skin, shallow/worn-out ridges (due to aging/genetics), skin disease, sweat, dirt, humidity in the air all confound the non-ideal contact with the sensor.

The fingerprint processing and feature (minutia) extraction involves:

- Orientation estimation,
- Ridge orientation,
- Applying Gabor filters on the fingerprint image which have frequency-selective and orientation-selective properties. This enhances the ridges.

In this lab, we will perform fingerprint acquisition, as well as pre-processing of the collected fingerprints. These steps will prepare data for fingerprint matching (Lab 5 exercise). The `.m`-files for fingerprint image processing and plotting can be found on N drive in *\ENCM\509\lab4*.

2 The laboratory procedure

2.1 Fingerprint acquisition

The acquisition of the fingerprints will be performed on the USB Digital Persona U.are.U 4500 fingerprint sensors (reader), and the software developed in the Biometric Technologies Laboratory at UofC using the Digital persona SDK. The software creates "raw" data in the `.bmp` (or other) format. Connect the reader it to the computer using a USB port. The device will be ready once red light flashes. Open the program `digitalPersona.exe` on drive N in *\ENCM\509\DigitalPersona* (press "Run"). The acquisition can be performed by placing your finger on the sensor surface, and it can be done as many times as you want to get the good quality print shown in the interface window. To save the image, choose option "File" and then "Save" (save on your drive H). You can clean the reader with a Scotch tape (yes, press the scotch tape on the top of the sensor and lift it, – as good as new).

To create a database, collect:

- 30 of your left (or right) thumb or other finger images,
- 30 of your other thumb or the other finger.

In this lab, we will need one image of good quality and one of poor quality; the remaining fingerprints will be used in Lab 5.

2.2 Reading images in Matlab

Data can be read into Matlab using the `imread` operator. For example, you can read a bmp file (usually, in general RGB form) and convert it to a gray-scale image.

```
img = imread('YourImage');
img=rgb2gray(img);
```

2.3 Fingerprint image processing

The fingerprint processing is implemented in the demo `Lab4Fingerprint1.m` file, which calls other functions available on the N drive: $\backslash ENCM \backslash 509 \backslash lab4$.

The below algorithm can be traced using the demo (`file Lab4Fingerprint1.m`). After selecting the fingerprint image, it is segmented by isolating the foreground from the background. Then, the ridge orientation is calculated , and the singularity points (core and delta) are localized.

The fingerprint image processing procedure as well as the minutiae extraction involves the following steps:

- Segmentation (`segmentimage.m`): extracts fingerprint area from background and finds the contour.
- Orientation estimation : computes orientation array at every pixel at the original image using (`computeorientationarray.m`).
- Find singularity points : finds core, delta, etc. (`findsingularitypoint.m`)
- Ridge frequency estimation : computes local frequencies (`computelocalfrequency.m` using `frequest.m`) ,
- Gabor filtering : implements directional filters to enhance ridges (`enhance2ridgevalley.m` using `choosefrequency.m`).
- Thinning and skeleton cleaning : (`cleanskeleton.m` and `Thin.m`), and
- Find minutiae : `findminutia.m`.

2.4 Image pre-processing exercise

Explore Matlab possibilities to enhance the image.

2.4.1 Intensity enhancement

Modify Lab4Fingeprint1.m demo by adding histogram equalization or modified (adaptive) one:

- `histeq()` ,
- `adapthisteq()`,

To create the pixel intensity histogram for visual inspection, use

- `imhist(img)`

2.4.2 De-noising

Modify Lab4Fingeprint1.m demo by adding de-noising filters such as Matlab functions from Image Processing Toolbox, for example:

- `imadjust()`,
- `wiener2()`, Wiener filter,
- `medfilt2()`, median filter.

Apply them after reading and converting the image to gray-scale (`img=rgb2gray(img)`), for example, using

- `img = imadjust(img);`

Apply various filters and compare the results of minutiae extraction (last step in the demo) for the fingerprint images with and without pre-processing.

2.4.3 Segmentation

Run demo Lab4Fingerprint1.m to investigate the results of segmentation. Note that `segmentimage.m` uses filter `bw = medfilt2(bw, [16 16]);` with parameters [16 16] and `gradDev > 30`. What happens if those parameters are changed?

2.4.4 Orientation and singularity points

Demo Lab4Fingerprint1.m calls `computeorientationarray.m` that uses parameters `blkSize = 10`. What happens if this block size changes?

It also calls `findsingularitypoint.m`. How many singularity points are found in your fingerprint?

2.4.5 Frequency of ridges

Demo Lab4Fingerprint1.m calls `computelocalfrequency.m` that uses parameters `border = 30`. Will decreasing of this parameter influence the outcomes of the further steps (minutia finding)?

2.4.6 Ridge enhancement and skeleton building

Demo Lab4Fingerprint1.m calls `enhance2ridgevalley.m` that calls Gabor filter and also create skeleton used for the next step (cleaning and thinning). The same Gabor filter is also used for feature extraction in Lab 5. Mode details on the Gabor are studied in the next Section of this lab. `cleanskeleton.m` uses different block sized in its morphological operation blkSize to detect holes

```
   skeleton = Cs.skeleton;
for i = 1:5
blkSize = round(Cs.holeBlkSize/i);
skeleton = blkproc(skeleton, [blkSize blkSize],@ removehole);
skeleton = Thin(imcomplement(skeleton), 'Inf');
end
```

Will reducing the size (by using `i` up to 6 or more) improve the skeleton quality?

2.5 Gabor filter

Step 5 of the demo involves ridge detection using a 2D Gabor filter. Let us explore this special filter in detail.

A Gabor filter belongs to a tuned band pass filter bank. It has a Gaussian transfer function in the frequency domain. Thus, the Gabor filter is a Gaussian, and in case of 2D, it is modulated by a complex sinusoid (with centre frequencies along x and y-axes respectively), and is characterized by the frequency, `f` and angle, `angle`. Gabor filters are used mostly for shape detecting and feature extracting in image processing.

Example:

```
function gabor = GaborFilter(img, tx, ty, angle, f)
x = [-16:16]; y = [-16:16];
[x, y] = meshgrid(x,y);
xp = sin(angle) .* x + cos(angle) .* y;
yp = sin(angle) .* y - cos(angle) .* x;
gabor = exp(-0.5 * ((xp.^2 / tx^2) + (yp.^2/ty^2))).* cos(2 * pi * f .* xp);
```

Here, the filter is applied to the image `img`, the size of the filter is specified by the "window" we choose for the image processing, `tx` and `ty`, are filter parameters,

`angle` specifies an orientation (usually, one of 8), and frequency `f` (computed using local ridge frequencies for the given image).

Function `imfilter` below applies the filter, `gabor`, to the image, `img`:

```
imgabor = imfilter(img, gabor,'replicate','same');
```

The filter itself can be illustrated by plotting the 2D image for various (8) orientations.

Example:

```
 norient=8;
xsize=32;  % filter size on x
ysize=32;  % filter size on y
dx=8; dy=4; % stabdard deviation of gaussian envelope
f = 0.1;  % frequency
angle=[0:pi/8:pi-pi/8];
n=1;  %initiate the number of filters
figure, hold on
for a=angle
f{n}=GaborFilter(32,32,8, 4, a, f); %build the filter
subplot(1,norient,n),imagesc(f{n}), colormap gray
n=n+1;
end
```

Function `GaborFilter` can be found in the directory on the N drive, *\ENCM\509\lab4*. Apply the Gabor filter to your fingerprint image. Note that the image must be read first (see Lab4Fingeprint1.m demo) and converted to gray-scale double-precision format.

Example:

```
 [namefile,pathname]=uigetfile({'*.bmp','IMAGE Files(*.bmp)'});
[img,map]=imread(strcat(pathname,namefile));
Fp.imOrig = img;
figure(1), imagesc(Fp.imOrig),colormap gray, title('Original Image');
imoriginal = Fp.imOrig;
```

You will have to convert the unsigned gray-scale representation to double-precision:

```
img = mat2gray(double(imoriginal));
```

The example below shows how to draw several images of various angles for the Gabor filter (image test1.bmp was used).

Example:

```
 [namefile,pathname]=uigetfile('test1.bmp','IMAGE Files (*.bmp)'); });
[img,map]=imread(strcat(pathname,namefile));
Fp.imOrig = img;
figure(1), imagesc(Fp.imOrig),colormap gray, title('Original Image');
xsize=32; ysize=32;
tx=4; ty=4;
f=0.11; angle=pi/4;
fi=GaborFilter(xsize,ysize,tx, ty, angle, f);
I1=Fp.imOrig;
If1(:,:)=imfilter(I1,fi);
F1(:,:)=blkproc(squeeze(If1(:,:)),...
...  [xsize/2 ysize/2],inline('abs(mean2(x)-std2(x))'));
figure(2), plot(2), imagesc(fi), colormap gray;
figure(3), plot(2), imagesc(squeeze(F1(:,:))), colormap gray;
```

This will create three figures: the original one (we used `test1.bmp` image for illustration purpose, with four sets of lines: vertical, horizontal and two diagonals), the 2D image of the filter, and the filtered image. As shown in Figure 1, given the angle $pi/4$, the filter extracts the lines oriented along with the diagonal (right-upper corner to left-lower corner).

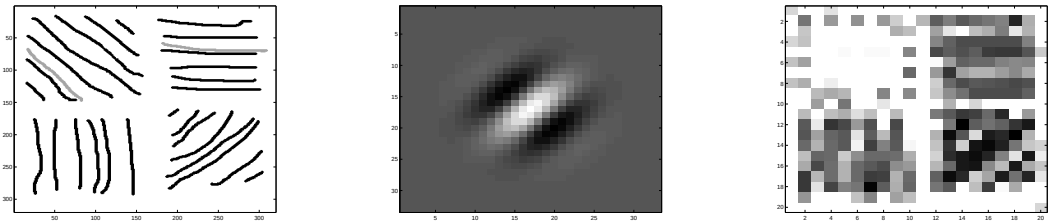

Figure 1: The original image, the filter and the filtered image.

Use the demo file Lab4Fingerprint2gab.m and modify it to perform Gabor filtering on your fingerprint image, adjust the parameters of the filter. What is the impact of changing the parameters of the Gabor filter?

3 Laboratory Report

In your report, include:

- General description of your lab exercise flow; the results of processing of the two fingerprints from your dataset: one of the better quality, and another of poor quality. Run demo file and obtain the maps of the minutiae point for each image. Include illustrations (30%).
- Answer all questions (20%).
- Perform Gabor filtering on your fingerprints using the adjusted (by you) parameters; include illustrations (30%).
- Include the created Matlab fragments (in the texts) or .m files where appropriate (20%).

4 Acknowledgments

We wish to thank research students who contributed to the projects implemented in the Biometric Technologies Laboratory. The acquisition executable `digitalPersona.exe` was created in the BT Lab based on the Digital Persona SDK. The implementation of some fingerprint processing protocols according to NIST (National Institute of Standardization, USA) are credited to K. Kryszczak, EPFL. We also wish to thank the department IT staff for setting up the software in the UofC labs, and the TA S. Samoil for verifying the lab exercise and codes.

Svetlana Yanushkevich.
October 2, 2015