Fall 2019

California State University, Northridge

Department of Electrical & Computer Engineering

ECE 595: Image Processing
Final Project Report
Thumbprint Matching Experiment in Software

Juan Silva & Christian Rodas

December 16, 2019

1.1 INTRODUCTION

Image matching is an application of pattern recognition where a pattern can be described via decision-theoretic principles. Quantitative descriptors such as cross correlation are used for determining how correlated a pixel is to its neighbor over an entire image. A correlation coefficient is generated and compared within a range of values from -1 to 1. Two images are said to be similar if the correlation coefficient is close to the perfect positive value 1. That being said, an experiment is conducted using two thumbprint images and checking how similar they are to each other.

KEYWORDS:

Cross Correlation, Image Processing, Grayscale, Convolution, Thinning, Erosion, Dilation, Matlab

1.2 PROCEDURE

In this experiment, two thumbprint images are created using ink and paper. These images are scanned and served as inputs in Matlab. A top-level block diagram of the experiment is shown in *Fig. 1.1*. Matlab will process these images and generate a co-occurrence matrix to be used for a high-level correlation function available in Matlab. The output of this function is plotted in order to visualize the correlation between the two input images.

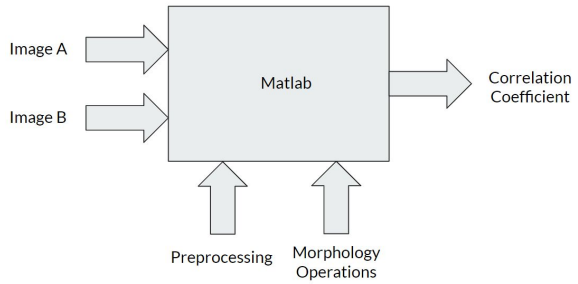

Fig 1.1: Experiment Top Level Diagram

Matlab has a function called **xcorr2** that calculates the cross correlation of two image matrices. According to the documentation, this calculation is done by applying convolution of the first image with another image kernel shown in *Fig. 1.2*. This is represented as a double summation shown in *Eq. 1.1*.

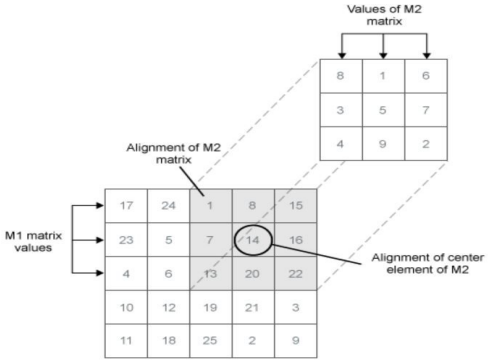

Fig 1.2: Convolution Visualized

$$C(k,l)=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}X(m,n)\,\overline{H}(m-k,n-l), \qquad \begin{matrix}-(P-1)\le k\le M-1,\\ -(Q-1)\le l\le N-1,\end{matrix}$$

Eq. 1.1: 2-D Cross Correlation

1.3 TESTING STRATEGY

Three cases are defined to test for different correlation cases. The first case will produce auto-correlation because the two inputs will be the same one-to-one image. This case will serve as a control for the experiment. The second case will produce cross-correlation because two completely different images. However, we expect the correlation coefficient result in a lower correlation value.The third case will also produce cross-correlation between two similar images.These inputs are the same thumbprint except one of them has more ink than the other. We expect the correlation coefficient result in a higher correlation value than Case 2.

1.4 RESULTS

Fig 1.3: 260x340 Grayscale

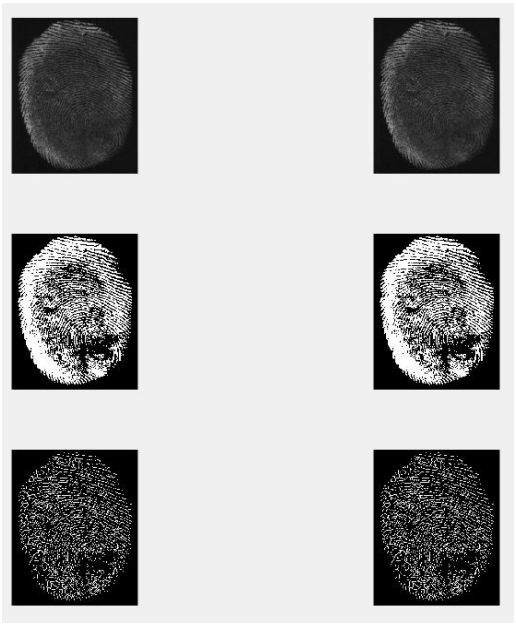

Fig 1.4: Preprocessing for Case 1

Fig 1.5: Preprocessing for Case 2

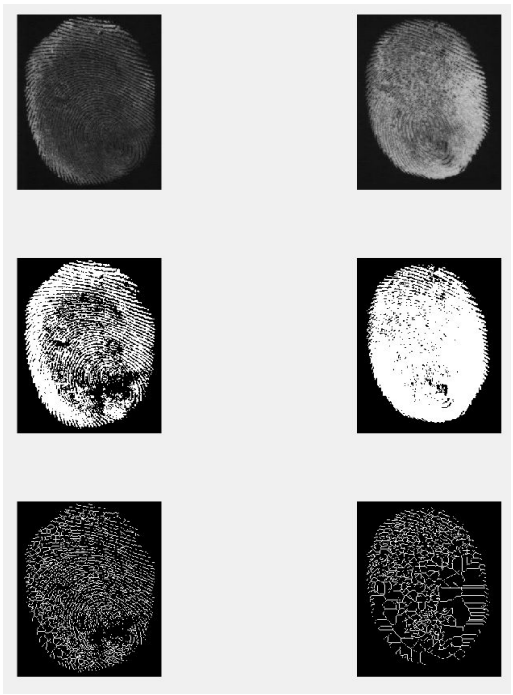

Fig 1.6: Preprocessing for Case 3

Fig 1.7: Inputs for Case 1.

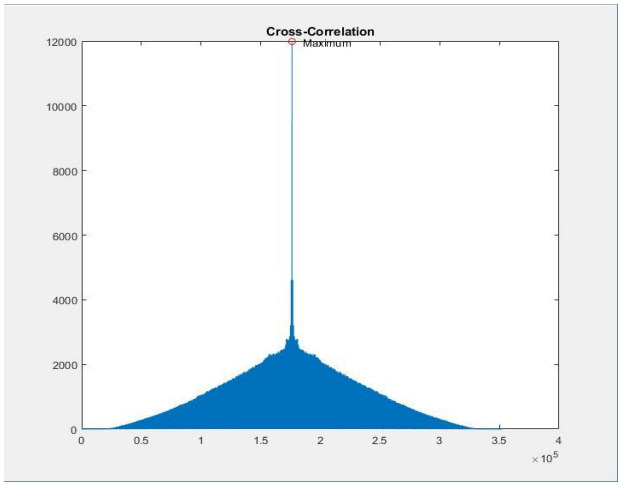

Fig 1.8: Results from Case 1.

Fig 1.9: Inputs for Case 2.

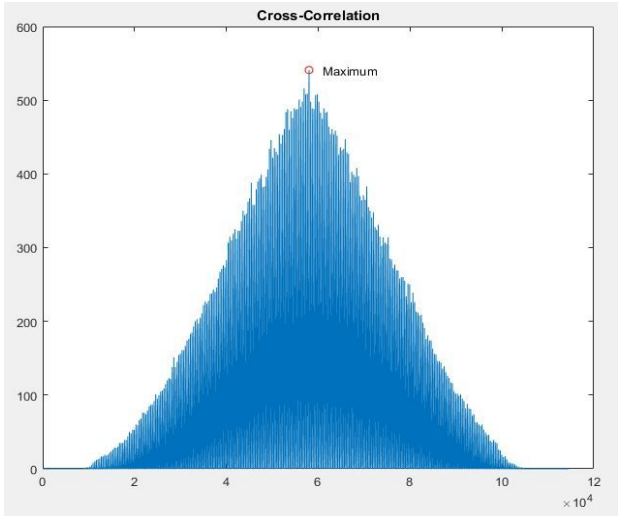

Fig 1.10: Results from Case 2.

Fig 1.11: Inputs for Case 3.

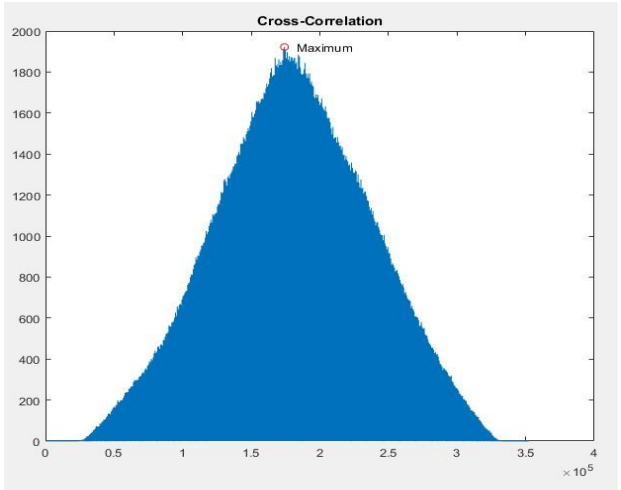

Fig 1.12: Results from Case 3.

1.4 ANALYSIS

The results generated from the cross-correlation function are graphed where the maximum peak represents a correlation coefficient. The axes for the graph are maximum points of a 3x3 neighborhood in vertical and horizontal direction.

For Case 1, where two identical fingerprint images are inputs, the expected result was a maximum peak from the graph meaning that there is a high correlation between two matched images. The maximum peak of 12,000 shown in *Fig. 1.8* supports this statement.

For Case 2, where two different fingerprint images are inputs, the expected results are a lower magnitude peak from the graph meaning that there's a lesser correlation between the two images. The maximum peak of about 500 is shown in *Fig. 1.10* supports this statement.

For Case 3, where two similar fingerprint images are inputs, the expected results are a magnitude peak higher than the peak from Case 2. The maximum peak of about 1900 is shown in *Fig. 1.12* supports this statement.

1.5 FUTURE WORKS

One way to improve this experiment would be to generate a threshold value to have a confidence interval for matching two images. This will require an increased sample size in order to generate this threshold value. Also, allowing real-time inputs corrupted with noise would allow for more inputs to be read. To remedy this, we apply additional morphologies such as opening, closing, dilation, or erosion of the image to remove said noise.

1.6 REFERENCES

[1] Digital Image Processing (p. 855)
Gonzalez / Woods, 3rd Edition

1.7 APPENDIX

thumbprint_match.m

```
clear, clc, close all
format short, format compact

%open the original image
fp = imread('fp1.JPG');
fp2 = imread('fp2.JPG');

%complements the image (for non BW images)
fp = imcomplement(fp);
fp2 = imcomplement(fp2);

%convert to grayscale
test = rgb2gray(fp);
test2 = rgb2gray(fp2);

%convert to binary
bw = imbinarize(test);
bw2 = imbinarize(test2);

%apply thinning
thin_bw = bwmorph(bw, 'thin', 'inf');
thin_bw2 = bwmorph(bw2,'thin','inf');

%convert from logical to double
thin_bw3 = double(thin_bw);
thin_bw4 = double(thin_bw2);

%apply cross-correlation (requires double)
R = xcorr2(thin_bw3, thin_bw4);

%Graph setup
[x y] = max(R(:));

%Plot the correlation function
plot(R(:))
title('Cross-Correlation');
hold on
plot(y, x, 'or')  %Wait until correlation plot complete
hold off          %plot a red circle to label maximum peak
text(y * 1.05, x, 'Maximum');

%Display the two inputs before and after image
processing
figure
subplot(3,2,1), imshow(test);
subplot(3,2,3), imshow(bw);
subplot(3,2,5), imshow(thin_bw);
subplot(3,2,2), imshow(test2);
subplot(3,2,4), imshow(bw2);
subplot(3,2,6), imshow(thin_bw2);
```