Edge Detection and Feature Extraction in Automated Fingerprint Identification Systems

Mike Boldischar and Cha Poa Moua
Undergraduate Students, Applied Mathematics and Computer Science

Key Words: Edge detection, thinning, feature extraction, afis, perceptron, mlp, neural network, laplace

Abstract
As a means of access control and criminal identification, Automated Fingerprint Identification Systems (AFIS) are widely utilized. In many areas of business, these systems are entrusted to verify identities of personnel before allowing access to restricted information or facilities. In the area of criminal investigation, these same systems are entrusted to find, match, and identify criminals. Obviously, these systems are given critical tasks and are performing them unsupervised most of the time. Although most steps in the process are procedural and can be automated, there are two critical phases that need to be performed intelligently and reliably. These two phases are edge detection and feature extraction. In order to enhance important features accurately in the fingerprint image, methods in edge detection are applied. Once the important features are exposed and artifacts are removed, feature extraction takes place. In this phase, the print is characterized by the extracted features for matching later. This article looks at the theoretical foundations and practical aspects of these two phases to understand their automation.

Introduction

Fingerprint technology is the most widely used form of biometric technology. The most popular method for acquiring fingerprints is by collecting inked impressions. There are recent technologies which allow the collection of fingerprints using scanning tools. Automated Fingerprint Identification Systems (AFIS) have now been developed to improve accuracy and efficiency over the previous fingerprint matching solutions. With the increased concern for identity theft and other privacy issues, fingerprint recognition systems and other biometric authentication systems are necessary to address security needs in the international community. AFIS is now used internationally for tracking criminals and preventing fraudulent voting. Many police departments use these systems to capture, search, and store fingerprint images from crime scenes. Some AFIS are capable of searching through fingerprints at a rate of 4,000 comparisons per second.

Overview of Automated Fingerprint Identification Systems

Each vendor offers different versions of AFIS. However, every system has the same fundamental parts: image recording and image matching. Both processes acquire and extract features from fingerprint images. The image recording process has the responsibility of enrolling fingerprints into the database. The image matching process compares candidate prints to the enrolled prints as part of the verification process in the feature matcher. The feature

matcher will produce a binary result of either a successful match or an unsuccessful match. See Figure 1 for a representation of an AFIS.

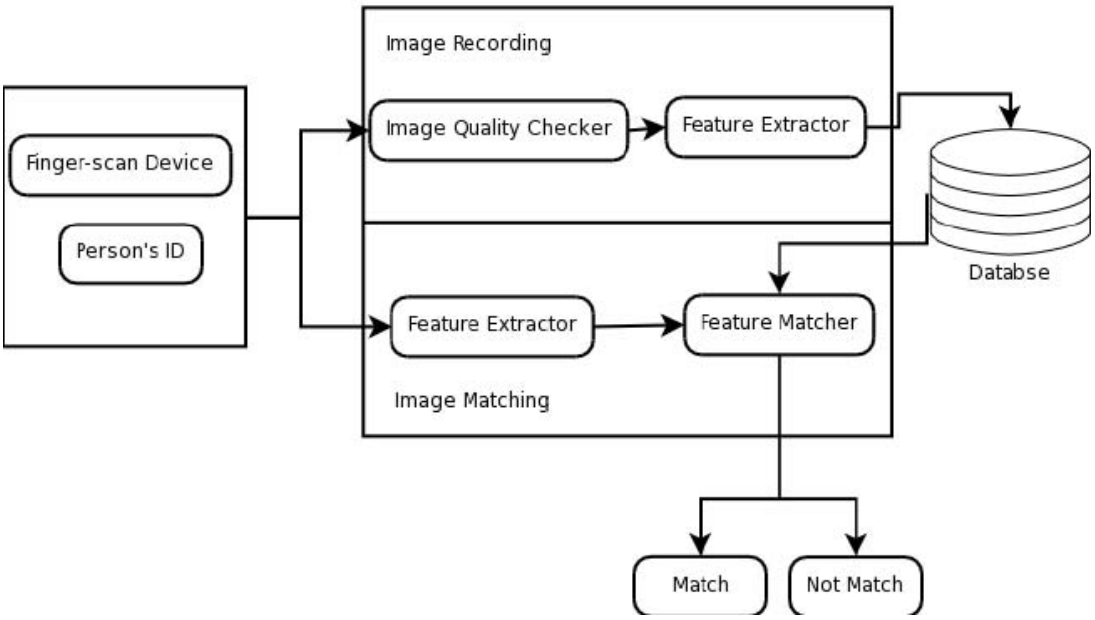

Figure 1. A typical Automated Fingerprint Identification System.

This article focuses on the feature extractor component shown in Figure 2. There are three main processes inside the feature extractor. The first process consists of gray scale normalization and edge detection. Secondly, the image is thinned to reduce artifacts and anomalies. Finally, the actual feature extraction takes place and fingerprint minutiae are identified.

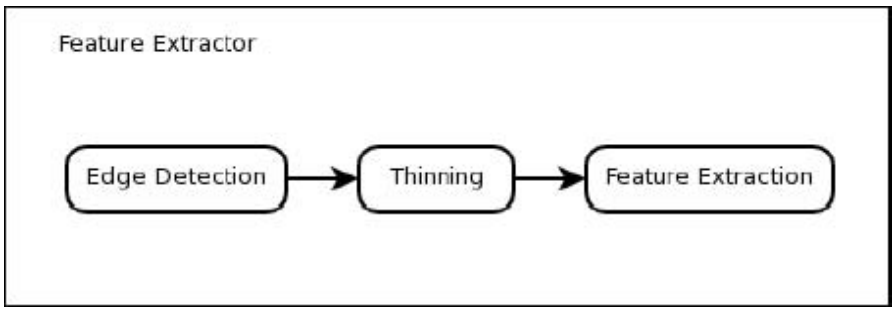

Figure 2. Expanded Feature Extractor Module from Figure 1.

Image Acquisition

There are many ways to acquire fingerprints for processing, matching, and storage. One way to acquire fingerprints is through finger-scan technology. Finger-scan devices can now be purchased by consumers or for business enterprise solutions. There are many forms of finger-scan devices which offer a variety of packages including hardware, software, and stand-alone

solutions. Many finger scan systems include image acquisition hardware, image processing components, matching components, and storage components. Each finger-scan device is different, and each of the components may be located in different places. Finger-scan devices are often integrated within other technologies such as software applications, automatic door locks, and mobile phones, but they can also be used to create databases for law enforcement and national security organizations.

It is important to have high quality fingerprint images in order to accomplish proper edge detection, thinning, feature extraction, and matching. Poor-quality of acquired samples greatly reduces the accuracy of the AFIS. Fingerprint scanning devices are capable of capturing images of over 500 dots per inch (DPI) which is the standard for forensic-quality fingerprint identification, but many devices only produce images around 350-400. This produces a problem since captured images from different devices often need to be referenced using the same algorithms and technologies. One way to implement quality control in the AFIS is by including multiple samples in hopes that many of the problems associated with poor-quality samples in the system will be alleviated.

Edge Detection

The purpose of edge detection in AFIS is to significantly reduce the amount of data found in a fingerprint image and leave only the most important information. Edge detection works by finding points on an image where the gray scale value changes greatly between pixels. If we consider edge detection in a one-dimensional array of gray scale values where an edge is present, it might look like Figure 3. This illustrates an obvious contrast in gray scale intensity. The darker pixels have low gray values while the lighter have high gray values.

Figure 3. Example of One-dimensional Gray-scale Image to Illustrate an Edge.

One method of performing edge detection is based on convolution. Convolution is a mathematical way of blending one function with another to produce a result expressing the amount of overlap the functions have on one another. Two of the most common edge detection filters are the Laplacian and the Canny operators. The Laplacian operator is a method of edge detection based on taking the second derivatives of the gray intensity (in the Cartesian coordinate system) while the Canny operator uses the first derivative of the intensity. The Canny operator is the most commonly used method for edge detection in AFIS since there are no significant advantages in other systems. The Laplacian function is shown in Equation 1.

$$L(x,y)=\frac{\partial^2 f(x,y)}{\partial x^2}+\frac{\partial^2 f(x,y)}{\partial y^2}$$

Equation 1. The Laplacian function which expresses the second derivative of a function.

There are some problems with using the Laplacian, since it is especially prone to picking up features which are not actually edges in the image. The Laplacian operator results in incorrect

readings where the gray value changes in small amounts consistently over part of the image. To reduce this noise in an image, a Gaussian blur is often applied before the Laplacian operator.

After the second derivative is found, a threshold must be applied to determine actual edges. More noise is produced with a lower threshold while a high threshold may miss some edges. Figure 4 shows the application of the Laplacian operator after a Gaussian blur has been applied. This example was created using the CIMP image editing software.

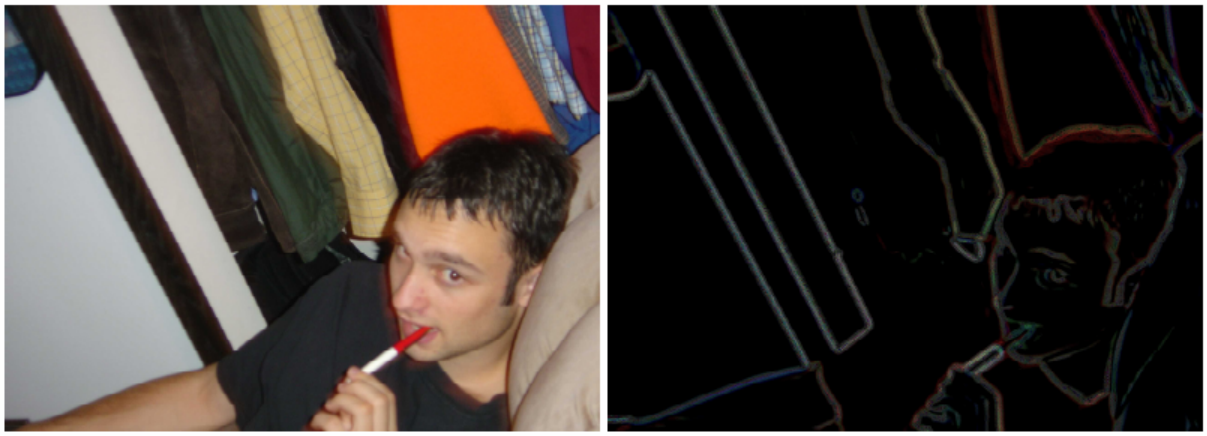

Figure 4. Example of the Laplacian operator applied to image after Gaussian blur.

Thinning

The purpose of thinning is to preserve the fingerprint minutiae shape while eliminating the extra information, which is of no use. After applying the thinning filter, the skeletal remains of the curves, only one pixel wide, are left. There are four main conditions dictating which pixels can be neglected. The neighborhood of a pixel *w* is defined as a 3x3 pixel area surrounding pixel *w*. Thinning is accomplished by using an iterative algorithm to turn off pixels by checking the neighborhood of the pixel in question and verifying that the four conditions are met. See Figure 5 for an illustration of thinning.

Figure 5. Artificial example to illustrate thinning where thick lines are condensed into thin "skeleton-like" lines.

A summary of the necessary conditions to negate a pixel will be provided based on Definitions 1 and 2; the actual mathematical conditions are shown in Conditions 1. The first condition eliminates the possibility of negating an isolated pixel or one that is completely

surrounded. The second condition makes sure the object being thinned does not get shorter. The third and fourth conditions guarantee the algorithm does not disconnect skeletal segments within objects. These conditions all ensure that the sample is properly thinned without losing valuable information.

Definition 1: ZNZT *(x).* Pick any pixel *p* on the boundary of *x*. Move along the boundary in the sequence P_{NW}, P_W, P_{SW}, P_S, P_{SE}, P_E, P_{NE}, P_N, P_{NW} where the subscript denotes given direction of the desired neighboring pixel. Count the number of times the boundary pixel switches from on to off.

Definition 2: NZN *(x)* is the number of neighbors of the pixel *x* which are on.

1. ZNZT *(x)*=1

2. 2≤NZN *(x)*≤6

3. (ZNZT **(**P_N)≠ - 1) OR (P_N, P_W, P_E are all on)

4. (ZNZT (P_W)≠ - 1) OR (P_N, P_W, P_S are all on)

Conditions 1. The above are the conditions necessary to turn a pixel off during thinning

Feature Extraction

Once the fingerprint image is thinned, artifacts are removed from it to improve extraction accuracy. Artifact removal algorithms typically look for anomalies like adjacent minutiae, intersecting perpendicular edges, and statistically impossible minutiae. After artifacts are detected and removed, the extraction of certain minutiae can begin.

As many as 150 minutiae types exist. The most common types are (1) ridge ending, (2) bifurcation, (3) lake, (4) dot or island, (5) crossover, (6) spur, (7) and independent ridge. Even with this limited set, minutiae recognition and extraction becomes difficult. Therefore, extraction is sufficiently limited to only two features as also dictated by practicality: ridge endings and bifurcations. The ridge ending is an abrupt stop to the ridge. The bifurcation is one ridge forking into two ridges. These two features are simply line contours that are easier to pick out accurately. In order to automate the recognition of these two features, the recognition program must be taught to accept an input pattern as a feature or a non-feature. This training process is done through the simulation of a neural network comprising of layers of perceptrons. In effect, this network of multilayered perceptrons guides a minutiae recognition program to make desirable scans.

The Rosenblatt Perceptron

The perceptron model grew from the need to extend simple neuron systems. Most limiting was the fact that the early neuron models accepted only binary inputs. In addition, the input weights were excluded from the evaluation of the function. Thus, in 1958 Rosenblatt introduced the concept of a single perceptron, in order to resolve the issues of limited input and unused weights.

Anatomy of a Perceptron

A perceptron is an automic system which can be taught to understand a concept. It learns by being exposed repeatedly to examples of what a concept is and what a concept is not. More

thoroughly, the teaching process requires the input of examples paired with desired outputs. The input examples are real values that signify a meaning in some context. Then, the training set can be defined as *Dm={(e1,d1), (e2,d2), ..., (em,dm)}*, where *m* denotes the number of examples, *e1* through *em* are the input examples and *d1* through *dm* are the associated desired outputs. Furthermore, a vector of weights is defined as *w1*, *w2* through *wm*. See a representation of a perceptron in Figure 6.

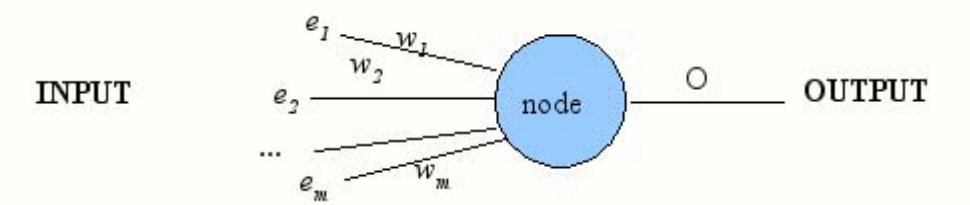

Figure 6. An *m*-variable Input Perceptron Model.

To create a mathematical equation, a linear combination of the inputs and weights occur. The weights are applied to the corresponding input values. The products of weighted inputs are then summed. This summation can be represented by Equation 2.

$$\sum_{i=1}^{m} e_i * w_i$$

Equation 2. Summation of Perceptron Inputs.

If the sum exceeds a set threshold, then the output is true. Otherwise the output is false. As indicated, the perceptron is a function which accepts real-valued inputs and evaluates to a binary output.

How Multilayered Perceptrons Recognize Minutiae

The perceptron network used in minutiae recognition consists of 3 layers. There are nine neurons in the first layer, each processing an associated element from the input vector. This first layer is simply an input layer, where no computation takes place. The second layer, also known as the hidden layer, consists of five neurons. This second layer evaluates the results of the first nine neurons. Finally, the last layer has only one neuron, acting as a single output.

The MLP is trained in off line mode, because training only needs to occur once. A set of known desirable and undesirable patterns for both minutiae are provided to the program simulating the network. Figure 7 contains training examples for the bifurcation minutia.

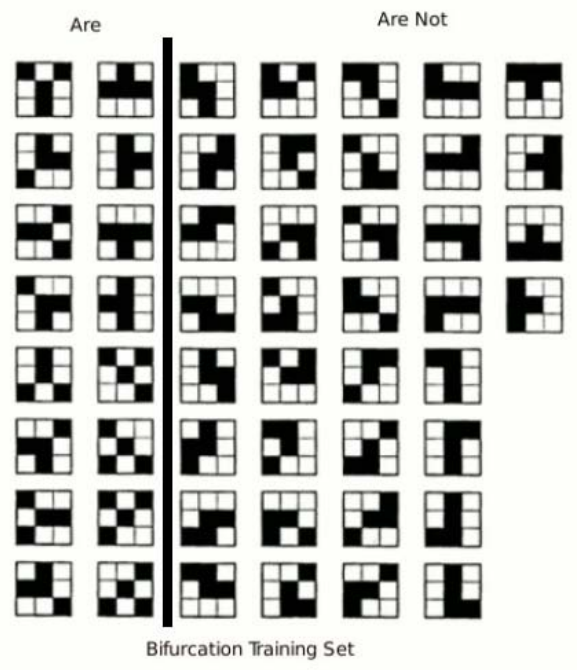

Figure 7. A bifurcation training set grouped by desired output (American University of Beirut, 2005).

Once the recognition program has been fully trained, each 3x3 pixel scan of the fingerprint image is passed to the program to identify as either an accepted minutia or not. If the 3x3 scan has centered onto an accepted minutia, an output of "1" results, otherwise an output of "0"results. For each input scan that resembles a minutiae, its position in relation to the core point of the fingerprint is recorded. This additional information, after some transformations, will be useful in the fingerprint matching process.

Conclusion

Although automated fingerprint identification systems come in various forms, there are fundamental parts that remain the same. Edge detection is vital in enhancing the quality of a fingerprint. After this enhancement is applied and the edges become distinct, the print is further thinned. Recognition works best on a thinned image, because the recognition program has sufficiently limited the domain of its parameters. Once features are recognized, they are extracted and stored with relevant information. The extracted features are later used to compare a new fingerprint with a set of enrolled fingerprints already stored in a database. If the recognized features match, the new fingerprint can be identified and associated with an individual.

References

American University of Beirut (2005, November 1). *Fingerprint identification – project 2*. Retrieved January 17, 2006, from http://webfea-lb.fea.aub.edu.lb/dsaf/labs/projectv1.1.pdf

Blais, A. (2001, July 1). *An introduction to neural networks*. Retrieved March 2, 2006, from http://www-128.ibm.com/developerworks/library/l-neural/

Honkela, A. (2001, May 30). *Multilayer perceptrons*. Retrieved March 15, 2006, from

http://www.cis.hut.fi/ahonkela/dippa/node41.html.

Nanavati, S., Thieme, M., & Nanavati, R. (2002). *Biometrics: identity verification in a networked world.* New York: John Wiley & Sons, 2002.

Noble, S. G. (2001, December). *Turning images into simple line-art.* Retrieved March 3, 2006, from http://www.reed.edu/~nobles/thesis/node3.html.

Ratha, N., & Bolle, R. (2004). *Automatic fingerprint recognition systems*. New York: Springer-Verlang, 2004.

Weisman, O., & Pollack, Z. (1995, August 13). *The perceptron*. Retrieved February 22, 2006, from http://www.cs.bgu.ac.il/~omri/Perceptron/

Weisstein, E. W. (2003). *Convolution*. *MathWorld*--A Wolfram Web Resource: http://mathworld.wolfram.com/Convolution.html.

Wikipedia (2006, March 2). *Automated fingerprint identification system*. Retrieved March 3, 2006, from http://en.wikipedia.org/wiki/Automated_Fingerprint_Identification_System

Wikipedia (2006, March 13). *Biometrics*. Retrieved March 3, 2006, from http://en.wikipedia.org/wiki/Biometrics.

Wikipedia (2006, March 14). *Perceptron*. Retrieved March 20, 2006, from http://en.wikipedia.org/wiki/Biometrics.