A FINGER PRINT RECOGNISER USING FUZZY EVOLUTIONARY PROGRAMMING

Author1:
K.Raghu Ram
4th E.C.E
raghuram.kolipaka@gmail.com
Newton's Institute of Engg.,
Macherla,Guntur(Dist.)

Author2:
K.Krishna Chaitanya
4th E.C.E
chaitu_kolluri@yahoo.com
Newton's Institute of Engg.,
Macherla,Guntur(Dist.)

ABSTRACT:

A fingerprint recognizing system is built with two principal components: the fingerprint administrator and the fingerprint recognizer. Fingerprints are identified by their special features such as ridge endings, ridge bifurcation, short ridges, and ridge enclosures, which are collectively called the minutiae. It explains the finger print characteristics that are used to identify individuals and the process of minutiae extraction. The fingerprint administrator uses the method of gray scale ridge tracing backed up by a validating procedure to extract the minutiae of fingerprints. The fingerprint recognizer employs the technique of fuzzy evolutionary programming to match the minutiae of an input fingerprint with those from a database.

1. INTRODUCTION:

Fingerprints of an individual are unique and are normally unchanged during the whole life. This method has been widely used in criminal identification, access authority verification, financial transferring confirmation, and many other civilian applications. In the old days, fingerprint recognition was done manually by professional experts. But this task has become more difficult and time consuming

In this paper, we explain the method of direct gray scale minutiae detection proposed in improved by a backup validating procedure to eliminate false minutiae. As for minutiae matching, we employed the technique of ***fuzzy evolutionary programming***, which has been used successfully in speaker identification, ***images clustering***, and ***fuzzy algebraic operations***.

2. FINGER PRINT CHARACTERSTICS:

A fingerprint is a textural image containing a large number of ridges that form groups of almost parallel curves (Figure 1). It has been established that fingerprint's ridges are individually unique and are unlikely to change during the whole life.

Figure 1. A fingerprint

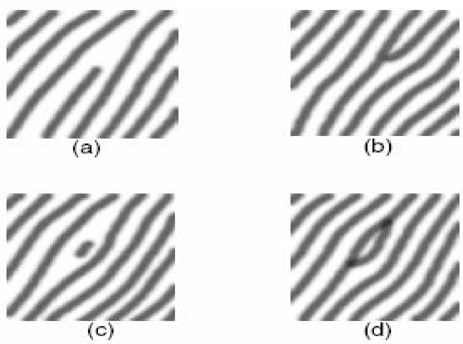


Figure 2. (a) Ridge ending; (b) Ridge bifurcatio (c) short ridge; (d) Ridge enclosure

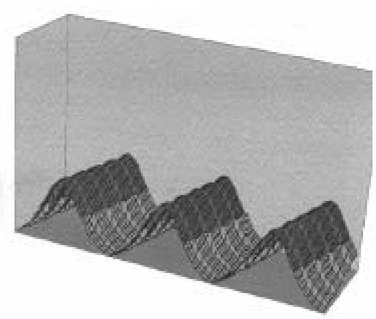
Although the structure of ridges in a fingerprint is fairly complex, it is well known that a fingerprint can be identified by its special features such as:

Ridge endings*:* The ending of the ridges takes place at the middle as shown in fig 2(a) . ***Ridge bifurcation:*** The division of the ridges in the middle as shown in fig 2(b).

Short ridges: The small lines present in between two ridges as shown in fig2(c). and ***Ridge enclosures****:* These are the loops formed between the ridges as shown in fig 2(d). These ridge features are collectively called the *minutiae* of the fingerprint. A full fingerprint normally contains 50 to 80 minutiae. According to the Federal Bureau of Investigation, it suffices to identify a fingerprint by matching 12 minutiae.

3. MINUTAE EXTRACTION;

For convenience, we represent a fingerprint image in reverse gray scale. That is, the dark pixels of the ridges are assigned high values where as the light pixels of the valleys are given low values. Figure 3 shows a section of ridges in this representation.



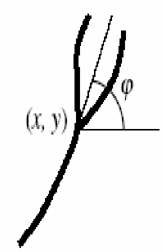


Figure 4. A minutia's attributes

Figure 3. A section of ridges.

In a fingerprint, each minutia is represented by its location (x, y) and the local ridge direction φ Figure 4 shows the attributes of a fingerprint's minutia. The process of minutiae detection starts with finding a summit point on a ridge, and then continues by tracing the ridge until a minutia, which can be either a ridge ending or bifurcation, is encountered.

3.1 FINDING A RIDGE SUMMIT POINT:

To find a summit point on a ridge, we start from a point $\boldsymbol{x} = (x1, x2)$ and compute the direction angle φ by using the gradient method. Then the vertical section orthogonal to the direction φ is constructed. The point in this section with maximum gray level is a summit point on the nearest ridge. The direction angle φ at a point $\boldsymbol{x}$

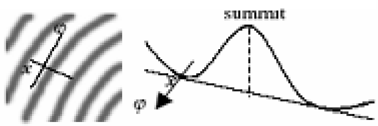


Figure 5. A summit point on a ridge

NW	N	NE
W	(x,y)	E
SW	S	SE

(a)

mentioned above is computed as follows. A 9×9 neighborhood around $\boldsymbol{x}$ is used to determine the trend of gray level change. At each pixel $\boldsymbol{u} = (u1, u2)$ in this neighborhood, a gradient vector $\boldsymbol{v}(\boldsymbol{u}) = (v1(\boldsymbol{u}), v2(\boldsymbol{u}))$ is obtained by applying the operator $\boldsymbol{h} = (h1, h2)$ with

$$h_1 = \frac{1}{4}\begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \qquad h_2 = \frac{1}{4}\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & -1 \end{bmatrix}$$

to the gray levels in a neighborhood of $\boldsymbol{u}$. That is,

$$v_1(u) = \sum_y g(y) h_1(y-u), \qquad v_2(u) = \sum_y g(y) h_2(y-u)$$

Where $\boldsymbol{y}$ runs over the eight neighboring pixels around $\boldsymbol{u}$ and $g(\boldsymbol{y})$ is the gray level of pixel $\boldsymbol{y}$ in the image. The angle φ represents the direction of the unit vector $\boldsymbol{t}$ that is (almost) orthogonal to all gradient vectors $\boldsymbol{v}$. That is, $\boldsymbol{t}$ is chosen so that $\sum_u (v \cdot t)^2$ is minimum.

3.2 TRACING A RIDGE:

The task of tracing a ridgeline to detect minutiae is described in the following algorithm. This algorithm also constructs a traced image of the fingerprint. Every time a new summit point of the ridge is found, its location in the traced image is assigned a high gray value and the surrounding pixels are given lower gray levels if they have not been marked.

Algorithm 1 (*Ridge tracing*): Start from a summit point $\boldsymbol{x}$ of a ridge.

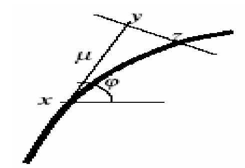


Figure 6. Ridge tracing

Repeat

Compute the direction angle φ at $\boldsymbol{x}$;

Move $\propto$ pixels from $\boldsymbol{x}$ along the direction φ to another point $\boldsymbol{y}$;

Find the next summit point $\boldsymbol{z}$ on the ridge, which is the local maximum of the section orthogonal to direction φ at point $\boldsymbol{y}$; Set $\boldsymbol{x} = \boldsymbol{z}$;

Until point $\boldsymbol{x}$ is a termination point (i.e. a minutia or off valid area).

Determine if the termination point $\boldsymbol{x}$ is a valid minutia, if so record it.

End Algorithm 1

There are three criteria used to terminate tracing a ridge. The first stopping condition is that when the current point is out of the area of interest. That is, the current point is within 10 pixels from the border, as experiments show that there are rarely any minutiae close to the edges of the image. The second criterion determines a ridge ending: the section at the current point contains no pixels with gray levels above a pre-specified threshold. In this case, the previous point on the ridge is recorded as a ridge endpoint. The last stopping condition corresponds to the case of a possible bifurcation: the current point is detected to be on another ridge that has been marked on the traced image. Algorithm 1 is backed up by a checking procedure that determines if a termination point is a valid minutia. The procedure is expressed as follows.

Algorithm 2 (*Elimination of false minutiae*):

- If the current ridge end is close to another ridge end with almost opposite direction, then delete both of them and join the gap, as they are simply broken ends of the same ridge.
- If the current bifurcation point is close to the end of one of its branch, then delete both of them, as short branch of a bifurcation is normally a result of light noise in the image.
- If the current termination is close to more than two other terminations, then delete all of them, as they are likely caused by damaged ridges in the image.

End of algorithm2

The above algorithms form one major component of our fingerprint recognizing system, called the *Fingerprint*. The *Fingerprint Administrator* is used to extract minutiae of known fingerprints and store them in a database. It is also used to extract minutiae of an input fingerprint for the purpose of identification.

4. FINGER PRINT RECOGNITION:

The primary purpose of our fingerprint recognizing system is to calculate the matching degree of the target fingerprint with the images in a database and to decide if it belongs to a particular individual. A fingerprint is said to match one image in the database if the degree of matching between its minutiae and that of the image in the database is higher than some pre-specified acceptance level. The method of calculating this matching degree is based on our fuzzy evolutionary programming technique, which is described below.

Consider two fingerprints that are represented by their sets of minutiae P= {P1, K, Pm}, Q={Qq1, K, Qn} where $p_i = (x_i, y_i, \alpha_i)$ and $q_j = (u_j, v_j, \beta_j)$ for $1 \le i \le m,\ \ 1 \le j \le n$. Observe that the two sets may not have the same number of points, and that the order of the points in each set is possibly arbitrary.

The principal task is to find a transformation F = (*s*, θ, δ*x*, δ*y*) that transforms the set of minutiae *P* into the set *Q* Here, *s* represents a scaling factor, θ an angle of rotation, and $(\delta x,\ \delta y)$ a translation in the *x y*-plane. Thus, the transform $F(\boldsymbol{p}) = (x', y', \alpha')$ of a minutiae p = (x, y, α) is defined by:

$$\begin{bmatrix} x' \\ y' \\ \alpha' \end{bmatrix} = s \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & \frac{1}{s} \end{bmatrix} \begin{bmatrix} x \\ y \\ \alpha \end{bmatrix} + \begin{bmatrix} \delta x \\ \delta y \\ \theta \end{bmatrix}$$

Also, in order to calculate the degree of matching, we associate with each minutia $F(\boldsymbol{p})$ a fuzzy set (also denoted by $F(\boldsymbol{p})$ for convenience) the membership function of which is definedby:

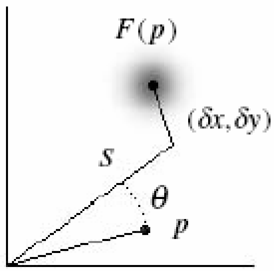


Figure 8. $F(\boldsymbol{p})$ is transform of $\boldsymbol{p}$

$$\mu_{F(p)}(q) = \min(\mu_{(0,\varepsilon)}(d(F(p),q)), \mu_{(0,\varepsilon')}(|\alpha - \beta + \theta|)),$$

where $d(F(p),q) = \sqrt{(x-u)^2 + (y-v)^2}$ and $\mu_{(0,\varepsilon)}$ represents a fuzzy subset of the real line defined as follows:

$$\mu_{(0,\varepsilon)}(x) = \begin{cases} \dfrac{e^{-p\left(\frac{x}{\varepsilon}\right)^2} - e^{-p}}{1 - e^{-p}} & \text{if } |x| \le \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

Intuitively, we allow some degree of tolerance in matching the minutiae $F(\boldsymbol{p})$ and $\boldsymbol{q}$, but this tolerance decreases rapidly when the two minutiae are far apart. The matching degree between two sets $F(P)$ and Q is defined as:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \mu_{F(p_i)}(q_j).$$

The task of finding a transformation F to match two sets of minutiae P and Q consists of two phases. First the rotation angle θ is estimated by the following algorithm.

<u>Algorithm 3 (*Estimation of rotation*):</u>

- Divide the interval [- π, π] into K subintervals $[\theta k, \theta k+1]$, k = 1 to K.
- Set up an integer array $c[1, K]$ and a real array $\Theta[1, K]$ and initialize them to 0.
- For each i = 1 to m and each j = 1… n do Find an index k such that $\theta k \le \beta j - \alpha I < \theta k+1$. Increment $c[k]$ by 1 and increment $\Theta[k]$ by $(\beta j - \alpha i)$. Find the index k^* such that $c[k^*]$ is maximum. Let θ be defined by $$\theta = \frac{\Theta[k^*]}{c[k^*]}.$$

End of algorithm3

Having established the rotation angle θ, the remaining parameters s and δx, δy of the transformation F are estimated by the following algorithm.

<u>Algorithm4(*Fuzzyevolutionary programming*):</u>

- Generate a population of m chromosomes $F_k = (s_k, \theta, \delta x_k, \delta y_k)$, $k = 1, \ldots, m$, *where* the parameter values are randomly taken from appropriate intervals.
- For each k = 1…m, compute the fitness:

$$f_k = \sum_{i=1}^{m} \sum_{j=1}^{n} \mu_{F_k(p_i)}(q_j)$$

Repeat

- For each k = 1… m, generate an offspring $F_{m+k} = (s_{m+k}, \theta, \delta x_{m+k}, \delta y_{m+k})$ as follows:

$$s_{m+k} = s_k + \rho_1(s_k)N(0,\sigma(f_k)),$$
$$\delta x_{m+k} = \delta x_k + \rho_2(\delta x_k)N(0,\sigma(f_k)),$$
$$\delta y_{m+k} = \delta y_k + \rho_3(\delta y_k)N(0,\sigma(f_k)),$$

Where $N(0,\sigma(f_k))$ is a normal distribution with variance $\sigma(f_k)$ inversely proportional to the fitness f_k, and the factors $\rho_1(s_k)$, $\rho_2(\delta x_k)$ and $\rho_3(\delta y_k)$ are used to ensure that the new values are within the predetermined intervals.

- Compute the fitness f_{m+k} of the new offspring using formula (1). For each k = 1… 2m, select a random set U of c indices from 1 to 2m, and record the number w_k of $h \in U$ such that $f_h \le f_k$. Select m chromosomes from the set $\{F_1, \ldots, F_{2m}\}$ that have highest scores w_k to form the next generation of population.

Until the population is stabilized.

End of algorithm4

6. Conclusion:

We have presented a fingerprint recognizing system that uses the method of gray scale ridge

o Algorithms, McGraw-Hill, New York, 1990.

tracing backed up by a validating procedure to detect fingerprint's minutiae and that employs the technique of fuzzy evolutionary programming to match two sets of minutiae in order to identify a fingerprint. The experimental results show that the system is highly effective with relatively clean fingerprints. However, for poorly linked and badly damaged fingerprints, the system appears to be not so successful. In order to handle those bad types of fingerprints, the addition of a preprocessing component that also adopts the fuzzy evolutionary approach to reconstruct and enhance the fingerprints before they are processed by the system. Also, it is possible to connect the system with a live fingerprint scanner that obtains a person's fingerprint directly and sends it to the system for identification.

7. References

[1] Arcelli, C., and Baja, G.S.D. "A Width Independent Fast Thinning Algorithm", *IEEE Trans. Pattern Analysis Machine Intelligence,* Vol. 7, No. 4, 1984, pp. 463-474.

[2] Baruch, O. "Line Thinning by Line Following", *Pattern Recognition Letters,* Vol. 8, No. 4, 1988, pp. 271-276.