제 19 회 한국정보처리학회 춘계학술발표대회 논문집 제 10 권 제 1 호 (2003. 5)

특징점 배치도 알고리즘을 이용한 지문 정합

임재영*, 장경식*
*한국기술교육대학교 정보기술공학부
e-mail : {micro_op, ksjang}@kut.ac.kr

Fingerprint Matching Using Minutiae Constellation Algorithm

Jae-Young Lim*, Kyung-Sik Jang*
* School of Information Technology, Korea University of Technology and Education

요 약

지문 정합을 할 때 특징점 사이의 거리와 각도가 유사한 순서대로 있는가를 비교하는 알고리즘을 제안한다. 한 점에서 가장 가까운 특징점을 찾고 다시 찾은 점에서 가장 가까운 특징점을 찾는데 이러한 세 개의 특징점들 사이의 거리와 끼인 각도를 기본 요소로 하여 이들의 순차를 특징점 배치도로 정의하여 등록지문에도 유사한 순차가 있는지를 검사한다. 정합 시에 특징점 사이의 거리, 각도 순차가 있는가를 검사하기 때문에 중심점을 찾지 않아도 되며 지문의 이동, 회전에 영향을 받지 않는다.

1. 서론

지문을 이용한 사용자 인증은 생체 정보를 이용한 개인 인증 방식 중 가장 오래된 것일 뿐만 아니라 현재 가장 널리 사용되는 방식이다. 지문 인식 알고리즘은 일반적으로 특징 추출과 지문 정합의 2 단계를 거치게 된다. 지문 특징 추출 단계에서는 지문 정합 단계에서 사용할 특징점(minutiae) 데이터 파일을 구성하는데, 일반적으로 전처리를 통한 지문 융선 방향성을 사전 정보로 이용하여 특징점 추출, 후처리의 단계로 진행된다[1][2][3]. 지문 정합 방식은 크게 특징점 정보를 이용한 정합과 전역 정합 방식[4]이 있다. 특징점 정합 방식은 모든 특징점을 연결하는 거리의 합이 최소가 되도록 Minimal Spanning Tree(MST)로 구성하여 유사성을 찾는 방법[5]과, 각 특징점의 방향을 기준으로 국부 좌표계(local coordinate)를 설정한 다음 사분면상에 존재하는 가장 가까운 특징점과의 거리, 각도등을 정합에 이용하는 방법이 있었다[6]. 또한 대용량 특징점 패턴 정합의 속도 향상을 위해 특징점 벡터를 병렬화하여 LUT에서 찾는 시도가 있었다[7].

본 논문에서 제안하는 방법은 세 특징점 사이의 거리와 각도를 기본 요소로 하여 특징점 배치도를 만드는데 정합시에 이러한 배치도의 유사성을 비교하기 때문에 지문의 중심점을 찾을 필요가 없으며, 지문 영상의 이동, 회전에 영향을 받지 않는다.

본 논문의 구성은 2 장에서 관련 연구 및 동기를 설명하고 3, 4 장에서 특징점 배치도를 정의하고 정합 방법을 설명하며 5 장에서는 실험 결과와 6 장에서 결론, 향후 과제를 기술하도록 한다.

2. 관련 연구 및 동기

특징점 정보를 직교 좌표계[7] 또는 극 좌표계[2]를 이용해 표현할 경우 기준이 되는 중심점이 필요하다. 중심점은 코어(Core)와 델타(Delta)로 정의 할 수 있다. 그러나 모든 지문이 중심점을 가지고 있다고는 할 수 없기 때문에 그 사용이 제한적이며 중심점을 잘못 찾았을 경우에 정합율이 낮아지는 문제가 발생한다.

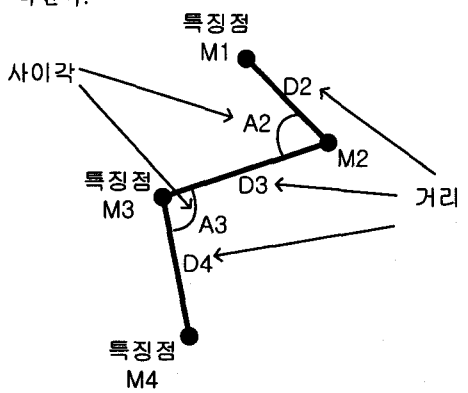
MST(Minimal Spanning Tree)를 이용하는 방법[5]은 지문상에 존재하는 특징점의 위치를 최소의 비용으로 연결할 수 있는 트리를 작성한 다음 정합에 이용하는 방법이다. 하나의 특징점이 누락 혹은 첨가되는 경우 트리의 형태가 바뀌게 되며 계산복잡도가 높다는 단점이 있다.

본 논문에서는 특징점 정보를 중심점을 찾지 않고 표현하는 방법을 제안한다. MST 와는 다르게 왼쪽 위에서부터 가장 가까운 특징점을 잇는 특징점 배치도를 만들어서 정합시에 등록된 지문에서 특징점 배치도 형태가 유사한 부분을 찾는 방법을 고안했다. 각 지문의 특징점 배치도는 고유한 거리와 각도로 구성되어 있으며 그러한 순차들은 동일지문이 회전, 이동될 경우에도 불변하는 요인이라는 점에 착안하였다.

3. 특징점 배치도 정의

특징점 추출 후에 특징점 배치도를 만들어 템플릿(template)에 저장한다. 그림 1 과 같이 특징점 배치도는 거리와 사이각으로 표현할 수 있다. 특징점 배치도를 만드는 순서는 다음과 같다.

1. 특징점을 제일 위, 왼쪽에서부터 시작점으로 하여 제일 가까운 점을 찾는다.
2. 찾은 점을 시작점으로 하여 다시 제일 가까운 특징점을 찾는다.
3. 모든 특징점들을 2 와 같은 방법으로 찾아가면서 각각의 거리와 사이각을 구한다.
4. 거리는 이전 점과의 거리이며, 사이각은 현재점을 중심으로 하여 이전 점과 다음 점을 잇는 선분을 그었을 때 180° 보다 작은 값으로 정의한다.
5. 템플릿 저장 순서는 이전점과의 거리, 사이각 순으로 저장한다.
6. 한번 찾은 점들은 다음에 찾지 않는다.
7. 모든 특징점들을 다 찾을때까지 2,3,4,5,6 을 반복한다.

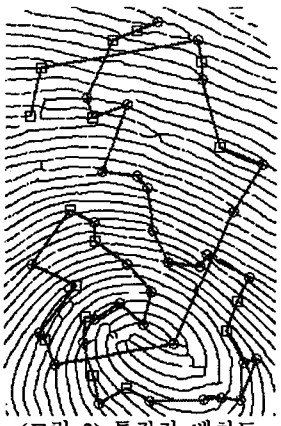


(그림 1) 특징점 배치도의 거리와 각도

사이각이 생성되기 위해서는 최소한 세 개의 특징점이 필요한데 이러한 세 점과 사이각 그리고 두 개의 거리 정보는 특징점 배치도 순차의 기본 단위가 된다. 예를 들면 그림 1 에서 D2, A2, D3 의 특징점 배치도 순차는 특징점 M1, M2, M3 가 반드시 존재하여야 생성될 수 있다. 템플릿에는 특징점 배치도의 기본 요소인 거리, 각도가 저장되는데 이러한 거리, 각도의 순차를 정합의 키로 이용한다. 그림 1 과 같이 각 특징점에서 이전 점과의 거리와 사이각을 정의하는데 템플릿의 저장 내용은 M1 을 시작점으로 했을 때 다음과 같다.

0, 0, D2, A2, D3, A3, D4

시작점에는 거리와 사이각이 없으므로 0, 0 이 된다. D2, A2, D3, A3, D4 는 하나의 패턴을 이루며 이러한 순차는 지문이 이동, 회전하더라도 변하지 않는 특징점 배치도의 특성이 된다. 그림 2 에 특징점 배치도를 표시하였다. 원으로 표시된 것은 분기점(bifurcation)이고 사각형으로 표시된 것은 끝점(ending point)이다.

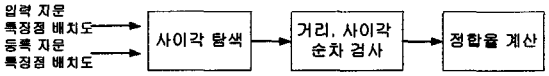


(그림 2) 특징점 배치도

4. 특징점 배치도 알고리즘을 이용한 지문 정합

본 장에서는 특징점 배치도 찾기 방법을 이용한 정합 알고리즘을 기술한다. 그레이 영상의 지문에서 추출된 특징점 집합을 입력으로 하여 3 장에서 기술한 방법으로 특징점 배치도 데이터를 만들어 템플릿에 저장한다. 그림 3 에 정합과정을 나타내었다. 등록된 지문은 템플릿에 저장되며 요구인 지문(query fingerprint)으로부터 얻은 템플릿과 비교한다. 두 특징점 배치도의 사이각 탐색을 하여 유사한 사이각을 찾은 후에 거리, 사이각 순차를 검색한다. 입력 특징점 배치도를 C^q 라고 하고 등록된 특징점 배치도를 C^r 이라 할 때 특징점 배치도의 집합은 식(1)로 표현 할 수 있으며 N 과 M 은 입력, 등록 지문의 특징점 개수와 같다. d_i 는 거리이며 a_i 는 사이각을 나타낸다.

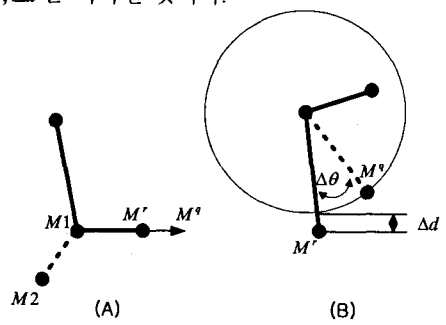
$$C^q = \{d_1^q, a_1^q, ..., d_i^q, a_i^q\} \quad i = 1...N$$
$$C^r = \{d_1^r, a_1^r, ..., d_j^r, a_j^r\} \quad j = 1...M \qquad \text{(식 1)}$$



(그림 3) 특징점 배치도를 이용한 정합 과정

시작점의 위치나 특징점의 이동, 의사 특징점(false minutiae) 등으로 인해 특징점 배치도의 순차는 같으나 저장 순서가 반대로 되는 경우가 있다. 그림 1 에서 예를 들면 D2, A2, D3 A3 D4 으로 특징점 배치도를 찾은 경우와 D4, A3, D3, A2 D2 순으로 찾은 경우가 있다. 그러므로 비교시에 사이각을 찾은 후에 양쪽 거리를 둘다 비교해본다. 이 과정을 그림 5 에 특징점 배치도 알고리즘으로 나타내었다. 결국은 두 지문을 비교할 때 얼마나 많은 특징점 배치도의 기본단위($d_i a_i d_{i+1}$ 의 순차)가 있는지 조사하는 것이다. 지문 입력 시 압력의 차이나 오염, 습도, 세선화 과정에서 끝점이 분기점으로 붙는 경우와 같은 요인으로 의사 특징점(false minutiae)이 나타날 수 있는데 그림 4 와 같이 사이각, 거리가

변하게 된다. 이러한 $\Delta\theta, \Delta d$ 에 대해 임계값을 T_a, T_d 로 설정했다. 그림 4 (A)에서 나타나듯이 M^q 로 특징점을 추출한다면 주변에 M^r 보다 가까운 $M2$ 가 있기 때문에 특징점 배치도의 순차가 $M1 \rightarrow M2$ 로 변화하여 정합율을 낮추는 요인이 된다. 그림 4 (B)에서는 등록 지문과 비교 지문의 $\Delta\theta, \Delta d$ 를 나타낸 것이다.



M^r :등록지문의 특징점 M^q :비교 지문의 특징점

$\Delta\theta$:사이각의 차 Δd :거리의 차

(그림 4) 의사 특징점으로 인한 사이각, 거리의 차

```
for (i = 1 to N - 1){
  for (j = 1 to M - 1){
    if (i, j is not in history[]){
      if(|a_i^r - a_j^q| < T_a){
      if(|d_i^r - d_j^q| < T_d && |d_{i+1}^r - d_{j+1}^q| < T_d){
      score++;
      i, j is registered in history[]
      p = i + 1; q = j + 1; r = i + 2; s = j + 2;
      while(Search(p, q, r, s)){p++, q++, r++, s++;}
      else if(|d_i^r - d_{j+1}^q| < T_d && |d_{i+1}^r - d_j^q| < T_d){
      score++;
      i, j is registered in history[]
      p = i + 1; q = j - 1; r = i + 2; s = j - 1;
      while(Search(p, q, r, s)){p++, q--, r++, s--;}}}}}
}
BOOL Search(int i, int j, int l, int m){
    if (|a_i^r - a_j^q| < T_a && |d_l^r - d_m^q| < T_d){
      score++;
      i, j is registered in history[]
      retrun TRUE;}
    else return FALSE;
}
```

(그림 5) 특징점 배치도 알고리즘

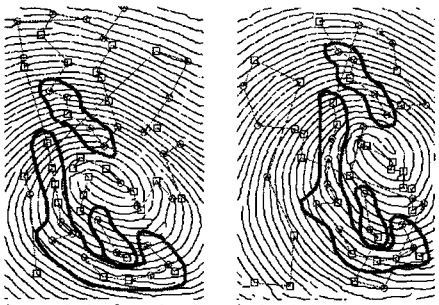
그림 5 와 같이 알고리즘을 적용하면 검사한 특징점 배치도의 기본 단위는 각각 C^q 에서 $N-2$, C^r 에서 $M-2$ 개의 거리, 사이각의 쌍이 된다. 그리하여 정합율을 식(2)와 같이 정의한다. 그림 6 의 찾은 특징점 배치도를 보면 MS 가 크지 않음을 알 수 있다. 그림 4 의 의사 특징점이나 시작점의 위치, 지문의 이동, 회전으로 특징점이 추가, 삭제되어 나타나므로 MS 는 작게 된다. 그러나 그림 6 과 같이 찾은 특징점 배치도의 순차가 연속되어 유사하게 나타나므로 본인으로 판단하기에 충분하다.

$$MS = \frac{m^2}{(N-2)\times(M-2)} \qquad \text{(식 2)}$$

N : 비교 지문의 거리, 사이각 쌍의 개수

M : 등록 지문의 거리, 사이각 쌍의 개수

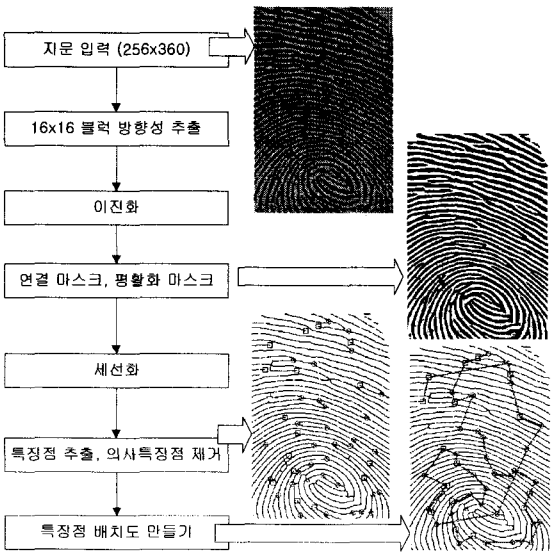
m : 찾은 별자리 기본 단위 개수



(그림 6) 회전,이동된 지문에서 찾은 특징점 배치도

5. 실험 결과

실험에 사용된 지문 DB 는 Cecrop 사의 CFS-3001UA 반도체식 센서(508dpi)로 획득된 5000(5 지문× 10 장× 100 명)장의 그레이레벨을 갖는 256× 360 화소 크기의 지문 영상으로 구성하였다.



(그림 7) 특징점 배치도 만들기 과정

실험 환경은 PC 의 Pentium -II 450MHz, RAM 192MB 이며 운영체제(OS)는 Windows 2000 환경에서 구현하였다. 그림 7 에 특징점 배치도를 만드는 과정을 나타내었다.

제안한 알고리즘에 대해 본인 거부율(FRR)과 타인 수락율(FAR)을 비교하였다. FRR 실험을 위해 한 손가락의 지문에 대해 10 번을 채취하고 서로 교차 비교하였으며 100 명 지문의 FRR 평균을 구한다. FAR 실험을 위해 모두 5000(5 지문× 10 장× 100 명)개의 지문을 등록하고 그 중 100 개의 지문에 대해 각각 비교하였다. FAR 을 0%로 고정하였을 때의 실험 결과는 표<1>과 같다.

<표 1> 실험 결과 분석표

FAR	FRR	특징점 배치도 만들기 시간	정합 시간 (Identification)
0%	8%	5.16ms	78ms

제안된 알고리즘은 특징점의 추가나 삭제, 이동 등에 민감하다는 것을 실험을 통해 확인하였다. 특징점 제거나 전처리 과정의 개선 등으로 특징점을 신뢰성 있게 추출한다면 더 좋은 결과를 얻을 수 있을 것이라 추정된다.

6. 결론 및 향후 과제

본 논문에서는 지문 정합을 할 때 특징점 사이의 거리와 각도가 유사한 순서대로 있는가를 비교하는 알고리즘을 제안했다. 한 점에서 가장 가까운 특징점을 찾고 다시 찾은 점에서 가장 가까운 특징점을 찾는데 이러한 세 개의 특징점들 사이의 거리와 끼인 각도를 기본 요소로 하여 이들의 순차를 특징점 배치도로 정의하여 등록지문에도 유사한 순차가 있는지를 검사한다. 각 지문의 특징점 배치도는 고유한 거리와 각도로 구성되어 있으며 그러한 순차들은 동일지문이 회전, 이동될 경우에도 불변한다. 두 지문 사이에 유사한 특징점 배치도가 있는가를 검사하기 때문에 중심점을 찾지 않아도 되며 지문의 이동, 회전에 영향을 받지 않는다. 그러나 의사 특징점이나 하나의 특징점이 누락 혹은 첨가되는 경우에 특징점 배치도 형태가 바뀌어서 정합율도 낮아지게 된다.

후처리 과정인 의사 특징점을 보다 엄밀히 정의하고 신뢰성 있는 특징점을 추출하는 연구를 진행하고 있으며 보다 많은 DB 와 전처리 과정의 모듈 개선 연구가 필요하다.

참고문헌

[1] 반성범, 문지현, 정용화, 김학일, "지문 인식 기술 동향", 전자통신동향분석, 제 16 권, 제 5 호, pp.46-54, 2001.10

[2] 김 현, "RSTI 불변 지문 특징량 추출 및 인식과 응용", 인하대학교 석사학위논문, 1998

[3] L. Hong, Y. Wan, A. Jain, "Fingerprint image enhancement : algorithm and performance evaluation," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 20, No. 8, pp.777-789, 1998.

[4] H. Yahagi, S. Igaki, and F. Yamagashi, "Moving-window algorithm for fast fingerprint verification," Proceedings of the IEEE Southeastcon'90, Vol. 1, pp.343-348, 1999

[5] D.Isenor and S.Zaky, "Fingerprint Identification Using Graph Matching", Pattern Recognition, Vol.19, No.2, pp.113-122, 1986

[6] Y. Hoshino and K. Asai, Identification System Employing Verification of Fingerprint, US Patent 4,944,021

[7] N. Ratha, A. Jain and D. Rover, "An FPGA-base Point Pattern Matching Processor with Application to Fingerprint Matching", CAMP '95, Italy, pp.394-401, Sept. 1995

[8] A.Jain, S.Prabhakar and A.Ross, "Fingerprint Matching: Data Acquisition and Performance Evaluation", MSU Technical Report TR99-14, 1999