신뢰도를 적용한 특징점 기반의 지문인식 알고리즘 제안 및 구현

박연규*, 김호일**, 이필중*

*포항공과대학교 정보통신대학원, **포항공과대학교 전자전기공학과

Proposal and Implementation of Minutiae-based Fingerprint Identification and Authentication using Confidence Concept

Yeon Kyu PARK*, Ho il KIM**, Pil Joong LEE*

*GSIT of POSTECH, **Dept. EEE of POSTECH

요 약

컴퓨터 및 네트워크의 기술 발전과 더불어 보안에 대한 필요성이 증가하고 있다. 이를 해결하기 위한 한 방법으로 지문이 개인의 식별 및 인증 수단으로 사용되고 있다. 본 논문에서는 **신뢰도**를 적용하여 보다 효율적이고, 믿을 수 있는 특징점 기반의 지문인식 시스템을 제안 및 구현한다. 마지막으로 제안된 시스템에 대한 성능 및 안전성을 분석을 한다.

I. 서론

컴퓨터 및 네트워크의 기술 발전과 더불어 보안에 대한 필요성이 증가하고 있다. 이를 해결하기 위한 방법으로 생체측정학을 이용한 기술 분야가 발전하고 있다. 특히 지문을 이용한 개인 인증 및 식별 방법은 지문이 개개인마다 그 형태가 다르며 일반적으로, 태어날 때부터 평생 동안 거의 변하지 않는 고유한 특성 때문에 신뢰성과 안정도에 있어서 다른 생체측정학(예, 얼굴, 홍채, 망막, 손, DNA 등)을 이용한 수단보다 높이 평가되고 있다. 또한 지문인식 시스템은 다른 생체인식 시스템에 비해 구축비용이 저렴하며 속도가 빠르며 크기가 작다는 장점을 가지고 있다. 현재 지문인식은 빌딩, 사무실, 연구실 등의 출입을 제한하거나 정부 기관에서 신원을 확인하는데 사용된다. 최근엔 금융 거래나 컴퓨터 시스템의 보안 등에도 사용되고 있으며, 그 응용분야는 기술이 발전함에 따라 확대되어 것으로 전망된다.

본 논문은 신뢰도라는 개념을 적용하여 보다 효율적이고 믿을 수 있는 지문인식 시스템을 제안하고 구현한다. II장에서는 일반적인 지문인식 시스템의 개요에 대해 살펴보고, III에서는 신뢰도를 적용한 지문인식 시스템을 제안하고 구현 방법에 대해 언급한다. IV장에서는 제안된 지문인식 시스템의 구현 결과를 분석하고, V장에서 결론을 맺는다.

II. 지문인식 시스템의 개요

지문을 이용한 개인의 인증은 오래 전부터 사용된 생체측정학이다. 지문은 16세기 후반부터 학문적 접근이 시작되었으며, 근대 지문인식 시스템은 19세기 후반 Galton[1]과 Henry[2]에 의해 정립되었다.

지문은 크게 돌출된 부분과 그렇지 않은 부분으로 나뉘는데 전자를 융선(Ridge, 땀샘이 융기한 부분)이라 하고 후자를 골(Valley)라 한다. Galton은 지문 융선의 일정한 흐름이 깨지는 비연속점(Local Discontinuities)들이 지문 형상에 많이 존재한다고 밝혔으며 이를 특징점(Minutiae)라 한다. 특징점은 개개인마다 다르며 타인의 지문과 구별하는 중요한 요소가 된다. Galton은 특징점을 Ridge, Forks, Islands, Enclosures의 4가지로 분류하였으나 이후 Dot, Ridge End, Island, Bifurcation, Short Ridge, Crossover, Bridge, Spur의 8가지로 확장되었다. 그러나 특징점을 기반을 기반으로 하는 대부분의 현재 지문인식 시스템에서는 대표적인 두 특징점, 단점(Ridge End, 이후 *END*로 표기)과 분기점(Bifurcation, 이후 *BIF*로 표기)만을 사용한다. F.B.I.의 기술 보고서[3]에서는 개인을 식별하기 위해서 *END*와 *BIF*만으로 충분하다고 발표했다.

특징점 외에도 중심점(지문 형상에서 방향의 변화가 가장 급격한 곳, 이후 *CORE*로 표기)과 삼각주(융선의 흐름이 세 방향으로 나타나는 곳, *DELTA*로 표기)라고 불리는 특이점(Singular Point)이 존재한다. 특이점은 주로 지문을 분류하거나 지문의 기준점(Reference Point)을 설정하는데 사용한다.

일반적으로 하나의 지문에는 적게는 50개에서 많게는 150개 정도의 특징점이 존재한다고 알려져 있으며, 이중에서 최소 10개만 있으면 개인을 식별하는데 충분하다고 한다[4].

지문인식 시스템은 매칭(Matching)에 사용하는 특징에 따라 크게 Galton Features를 사용하는 시스템과 다른 특징점을 사용하는 시스템으로 나눌 수 있다. 전자의 경우를 특징점 기반의 지문인식 시스템(Minutiae Based Fingerprint Identification System)이라 하며 현재 가장 널리 사용되고 있는 방식이다. 개인 인식에 필요한 개인별 특징 추출에 사용되는 방법으로는 크게 세선화를 이용한 특징점 추출, 주파수 공간에서의 Fourier, Wavelet변환, 또는 신경회로망이나 퍼지이론에 의한 것 등이 있으나 대부분 잡음을 줄이는 전처리 과정(Pre-Processing)이 필요하다[5].

Maio와 Maltoni[6][7]는 기존의 이진화와 세선화 같은 전처리 과정에서는 가짜 특징점이 발생할 수 있다는 단점을 보안하기 위해 Gray-Level 이미지에서 직접 특징점을 추출하는 방식을 제안했다. 이 방식은 Ridge-Line Following을 통해 Line이 끊어지거나 교차하는 부분을 탐지하여 특징점을 추출한다.

III. 신뢰도를 적용한 지문인식 시스템

1. 제안된 지문인식 시스템의 개요

본 장에서는 신뢰도를 적용한 지문인식 시스템을 제안 및 구현한다. 여기서 말하는 신뢰도는 지문 이미지의 신뢰도, 방향성의 신뢰도, 그리고 이 두 가지의 신뢰도가 적용된 특징점의 신뢰도이다. 먼저 이미지의 신뢰도는 시스템에 입력된 이미지의 질(Quality)을 측정한 것으로 이미지의 질이 높을수록(융선의 골과 마루가 선명할수록) 높은 수준의 신뢰도를 갖는다. 방향성의 신뢰도는 주변 방향성과의 일치도(Consistence)를 측정한 것으로 주변 방향성과의 차이가 적을수록 높은 수준의 신뢰도를 갖는다. 예를 들면, *CORE*나 *DELTA* 같은 특이점이나 이미지의 신뢰도가 낮은 곳에서는 낮은 수준의 신뢰도를 갖게 되며 이미지 신뢰도가 높고 융선이 한 방향으로만 흐른다면 높은 신뢰도를 갖게 된다. 특징점의 신뢰도는 이미지 신뢰도와 방향성 신뢰도가 적용된 전처리 과정을 거쳐 추출된 특징점에 부여하는 것으로 탐지된 특징점을 얼마나 믿을 수 있는지에 대한 측정값이다. 탐지된 특징점의 이미지 신뢰도와 방향성 신뢰도가 높은 경우는 진짜 특징점이라고 믿을 수 있으나 어느 한쪽만 높은 경우는 가짜 특징점일 수 있다. 예를 들면 이미지 신뢰도가 높고 방향성 신뢰도가 낮은 경우는 *CORE*나 *DELTA* 부근을 의미하므로 이 영역에서 탐지된 특징점은 전처리 과정에서 잘못 처리되어 잘못된 값(좌표값이나 방향성)을 갖는 특징점일 수 있다는 것이다.

특징점의 신뢰도는 매칭에 있어 중요한 역할을 한다. 특징점의 신뢰도는 매칭 스코어 계산에 해당 신뢰도만큼의 가중치로 작용하게 된다. 템플릿(Template, 신원의 특징점 및 특이점 데이터[22])의 특징점 신뢰도는 입력 지문이미지에서 탐지된 특징점과 비교하여 매칭되는 특징점이 있는 경우 신뢰도가 높아지지만 그렇지 않은 경우 신뢰도는 낮아진다. 다시 말해 템플릿의 특징점의 신뢰도가 높다는 것은 항상 탐지되는 특징점이라는 것을 의미한다. 반대로 신뢰도가 낮다는 것은 잘못 탐지된 특징점이거나, 사용자의 지문 입력 습관의 변화로 템플릿에 존재하는 특징점이 매칭시 사용되지 않는다는 것을 의미한다. 그리고 신뢰도가 하한 임계치 밑으로 낮아지면 템플릿의 용량, 템플릿을 저장하는 저장 매체의 용량, 특징점의 개수 등을 고려하여 템플릿에서 해당 특징점을 삭제한다.

특징점과 마찬가지로 특이점 역시 신뢰도를 가진다. 특이점은 매칭에 사용된 횟수를 통해 신뢰도를 판단한다. 특이점 신뢰도의 업데이트는 신뢰도를 이진 표기했을 때, 먼저 좌측이동(Left Shift)을 한 후 매칭에 사용된 경우는 *1*을, 그렇지 않은 경우 *0*을 *LSB*(Least Significant Bit)에 설정한다. 즉, *1*은 매칭시 사용되었음을 의미하며, *LSB*쪽의 *1*은 이전 매칭 시에 사용되었음을 의미한다. 특이점의 신뢰도는 템플릿에 2개의 *CORE*가 있고 입력 지문이미지에서 1개의 *CORE*가 탐지된 경우, 템플릿에서 신뢰도가 높은 특이점을 선택하는데 사용된다.

그림 1은 제안한 지문인식 알고리즘의 대략적인 흐름도를 보여주고 있다. 각 단계는 하나의 프로세스를 의미하며 하나의 프로세스는 여러 서브루틴을 포함한다. 단계 2는 입력된 지문이미지의 신뢰도를 부여하는 과정이다. 단계 3은 단계 2의 이미지 신뢰도를 적용하여 방향성 필드를 계산한다. 계산된 방향성 필드를 기반으로 방향성의 신뢰도를 계산하고 특이점을 추출한다. 제안된 알고리즘에서는 방향성 신뢰도가 1인 경우에서만 특이점이 탐색되므로 특이점 탐색에 있어 효율적이다. 단계 4 이후의 처리과정은 단계 2,3에서 계산된 신뢰도를 바탕으로 각 단계를 처리한다.

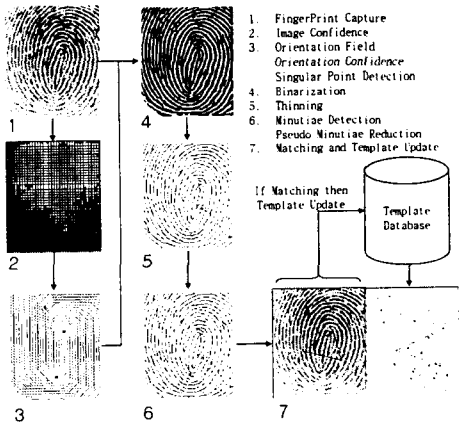


그림 1 제안된 지문인식 알고리즘 흐름도

2. Image Confidence

이미지 신뢰도는 입력된 지문이미지의 질을 판단하는 과정이다. 신뢰도는 지문이미지 전체가 아닌 일정 크기의 블록단위로 부여된다. 최하위 수준의 신뢰도는 지문이미지 중 여백(Background, 지문이 찍히지 않은 부분)을 의미하며 여백이 아닌 부분은 임계치를 적용하여 신뢰도의 수준을 세분화할 수 있다. 이미지 신뢰도는 실제 지문 영역을 대략적으로 계산할 수 있으므로 일정 지문 영역이 탐지되지 않는다면 지문을 재입력하도록 한다.

1) 지문이미지를 겹치지 않게 $w \times w$ 픽셀 크기의 블록으로 분할한다.

2) 블록 $[u,v]$ 내의 각 픽셀 $[i,j]$에 대해 변화도(Gradient) g_{xy}를 계산한다[11].

3) 블록 $[u,v]$ 내의 각 픽셀의 변화도의 합을 계산한 후, 블록 $[u,v]$의 이미지 신뢰도를 부여한다.

$$G[u,v] = \sum_k \sum_l g_{xy}[X_u+k, Y_v+l]$$

$$ImgConf[u,v] = \begin{cases} 0, & \text{if } G[u,v]/MAX(G) < T_{G1} \\ 1, & \text{if } G[u,v]/MAX(G) < T_{G2} \\ 2, & \text{if } G[u,v]/MAX(G) < T_{G3} \\ 3, & \text{otherwise} \end{cases}$$

k, l은 w의 크기에 따라 조절할 수 있으며, 제안된 시스템에서는 $-w \times (3/2) \le k,l \le w \times (3/2)$의 범위로 실정하였다. X_u, Y_v는 블록 $[u,v]$의 중앙 픽셀을 나타내며, $MAX(G)$는 $G[u,v]$ 중 가장 큰 값을 나타낸다. T_G는 이미지 신뢰도 부여를 위한 임계치이며, 제안된 시스템에서는 4등급의 신뢰도를 부여하였다.

3. Orientation Field

지문의 융선은 일정한 방향의 흐름을 가진다. *CORE*나 *DELTA*의 주변인 경우, 방향의 흐름은 다른 영역에 비해 급격히 변하게 된다. 융선의 흐름은 지문을 분류하거나 매칭 시에 중요한 요소로 사용되며, 일부 지문인식 시스템은 융선의 흐름을 이용한 매칭 알고리즘을 제안하기도 했다. 이는 입력 지문이미지의 방향성과 템플릿의 방향성의 상관관계를 계산하여 매칭을 결정하는 방식이다. 방향성을 계산하는 방법으로 많은 알고리즘[8] [9][10]이 제시되었으나, 제안된 시스템에서는 Hong, Wan, Jain[11]의 알고리즘을 사용했다. 다음은 방향성 계산과 방향성의 신뢰도 부여, 그리고 특이점 탐지에 대한 단계별 실명이다.

1) 지문이미지를 겹치지 않게 $w \times w$ 픽셀 크기의 블록으로 분할한다.

2) 블록 $[u,v]$ 내의 각 픽셀 $[i,j]$에 대해 *Sobel* 연산을 이용하여 변화도 $G_x[i,j]$, $G_y[i,j]$를 각각 계산한다(*Sobel* 연산이 아닌 다른 복잡한 미분 연산을 사용하여도 된다[16]).

3) 블록 $[u,v]$의 중심 픽셀 (X_u, Y_v)를 기준으로 $V_y[u,v]$, $V_x[u,v]$ 그리고 $dir[u,v]$를 계산한다.

$$dir[u,v] = \tan^{-1}(V_y[u,v], V_x[u,v])/2$$

4) 3)과정에서 구해진 방향성을 이용하여 전체 방향성 필드를 다시 보정하여 $G'_y[u,v]$, $G'_x[u,v]$, V'_y, V'_x와 최종 방향성 $Dir[u,v]$를 계산한다.

$$Dir[u,v] = \tan^{-1}(V'_y, V'_x)/2$$

이렇게 계산된 방향성은 *180*도를 *16*방향으로, 즉 *11.25*도씩 나누어 정수화 한다. 예를 들면 방향성 3인 경우는 다음의 범위를 갖게 된다.

$$if\ (11 \times PI)/32 > Dir[i][j] \ge (9 \times PI)/32)$$

$$Dir_I[i][j] = 3$$

여기서 PI는 *3.14152*의 값을 갖고 Dir_I는 Dir을 16방향(*0~15*)의 정수로 표현한 값을 갖는다. 이 과정에서는 음의 방향(*0~-180*도)을 고려하지 않으나 특징점 제거 과정에서 음의 방향을 고려한다.

5) 방향성에 대한 신뢰도를 계산한다. 중심 블록 $[u,v]$와 주변 인접한 $(w'' \times w''-1)$ 블록간의 방향성 차의 합을 계산한다. 융선이 일정한 방향으로 흐른다면 방향성 차의 합은 낮게 나오지만 *CORE*나 *DELTA* 영역에서는 높게 나오게 된다.

$$DD[u,v] = \sum_k \sum_l (|Dir[u+k,v+l]-Dir[u,v]|^2)$$

$$DirConf[u,v] = \begin{cases} 0, & \text{if } ImgConf[u,v]=0 \\ 1, & \text{if } DD[u,v] < T_{D1} \\ 2, & \text{if } DD[u,v] < T_{D2} \\ 3, & \text{otherwise} \end{cases}$$

k, l은 $(-w''/2) \le k, l \le w''/2$의 범위를 가지며 w''는 방향성 신뢰도를 계산하기 위한 윈도 크기를 말한다. T_D는 방향성 신뢰도 부여를 위한 임계치이며 제안된 시스템에서는 4단계의 신뢰도로 실정하였다.

6) *PoinCare Index*[9]를 이용하여 특이점을 탐지한다.

그림 2 인접한 *CORE*와 그렇지 않은 *CORE*

7) 6)에서 계산된 특이점을 보정한다. 6)에 사용된 알고리즘을 적용하면 *Whole CORE*에 대해서 *CORE*가 2개로 탐지되는 경우가 있다. 이 경우 *CORE*를 하나로 합치는 보정 과정을 거친다. 만약 인접한 두 *CORE* 사이에 *DirConf*가 1인 블록이 존재한다면 하나의 *CORE*로 인정한다. 그림 4는 2개의 *CORE*가 탐지된 경우 이를 하나의 *CORE*로 간주할 수 있는 경우와 그렇지 못한 경우를 보여주고 있다. 그림 4에서 두 번째 경우 역시 하나의 *CORE*로 간주할 수 있다. 즉 탐지된 두 *CORE*의 중간을 하나의 *CORE*로 간주할 수 있으나 거리가 떨어진 두 *CORE*를 가진다는 중요한 정보를 잃을 수 있기 때문에 바람직하지 못하다.

4. Binarization

이진화는 지문이미지를 흑(*0, BLACK*)과 백(*255, WHITE*)의 이진 이미지로 변경하는 과정을 말한다. 이진화 과정 이전에 전처리 단계로 이미지 보정 작업[11][21]이 선행되어야 한다. 이미지 보정 작업은 질이 낮은 이미지에서 잘못 탐지되는 가짜 특징점의 수를 줄여 준다. 그러나 잘못된 이미지 보정은 가짜 특징점의 수를 증가시킬 수도 있으며 중용한 정보를 손실시킬 수도 있다. 이미지 보정에는 Low Pass Filter, Wavelet Transform, Fast Fourier Transform등을 이용할 수 있다. 제안된 시스템에서는 Histogram Equalization, 3×3 픽셀 Normalization을 사용하였다.

제안된 시스템에서는 융선의 골과 마루의 그레이 레벨 피크를 탐지하여 이진화하는 방법을 사용하였다. 즉 골의 피크와 마루의 피크를 계산하여 두 피크 사이의 중앙을 기준으로 마루의 피크에 가까운 픽셀은 *BLACK*으로, 골의 피크에 가까운 픽셀은 *WHITE*로 변경하는 방법이다.

1) 지문이미지를 겹치지 않게 $w \times w$ 픽셀 크기의 블록으로 분할한다.

2) 블록 $[u,v]$에 대한 방향성을 검사한다.

3) 블록 $[u,v]$의 방향성을 고려하여 다음과 같이 선분 L을 형성한다.

$L[i]=OImage[T_x(i)\ T_y(i)],\ -3w/2 \le i \le 3w/2$

$T_x(\cdot)$와 $T_y(\cdot)$는 블록 $[u,v]$의 방향성과 수직 선분상의 X좌표와 Y좌표를 반환하는 함수이다.

4) L 상의 픽셀 중 마루와 골의 피크를 찾는다.

$P_{low}[i]$ = L상의 i 번째 *Low Peak*의 위치,

$P_{high}[j]$ = L상의 j 번째 *High Peak*의 위치,

여기서 i, j는 $i, j = 0, 1, 2, \ldots$의 범위를 가지며, $L[P_{low}[i]]$는 *0xFF*를, $L[P_{high}[j]]$는 *0xEF*를 설정한다.

5) P_{low}, P_{high}를 통해 L을 이진화 한다.

```
b = 0, w = 0
for i = from -3w/2 to 3w/2
    if(L[i] = 0xFF)
        P_B[b] = (P_low[b]+P_high[w])/2
        for i = from P_low[b] to P_B[b]-1
            L_bin[i] = BLACK
        for i = from P_B[b] to P_high[w]
            L_bin[i] = WHITE
        b++
```

*L[i]=0xEF*에 대해서도 위와 같은 식의 이진화를 진행한다. b와 w는 각각 P_{low}와 P_{high}의 인덱스이다.

6) L_{bin}을 이용하여 *RImage*를 생성한다.

```
if(L_bin[i]=BLACK && IsInW(T_x^-1(i), T_y^-1(i)))
    RImage(T_x^-1(i), T_y^-1(i)) = BLACK
if(L_bin[i]=BLACK && IsInW(T_x^-1(i), T_y^-1(i)))
    RImage(T_x^-1(i), T_y^-1(i)) = WHITE
```

여기서 $T_x^{-1}(\cdot)$와 $T_y^{-1}(\cdot)$는 L 상의 좌표를 *OImage*의 좌표로 변환하여 반환하는 함수이다. *IsInW*$(\cdot)$는 해당 픽셀이 $w \times w$의 범위 내에 속한 픽셀인지를 검사하는 함수이다.

그림 3과 그림4는 이진화에 과정과 결과를 보여주고 있다.

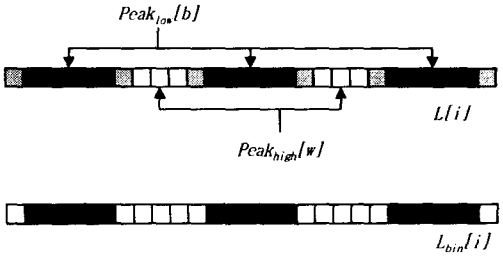


그림 3 w=9, *Dir*=8인 경우의 이진화

그림 4 이진화 처리 결과

5. Thinning

이진 이미지에서 *BLACK* 픽셀을 한 픽셀 두께로 세선화 하는 과정이다. 이진 이미지를 세선화 하는 방법은 여러 가지가 있으나[12][13], 제안한 시스템에서는 가중치를 이용한 병렬 알고리즘[14]을 사용했다.

제안된 시스템에서는 융선의 골과 마루의 정보를 이용하여 보다 정확한 특징점을 탐지하고자 한다. 따라서 이진화 이미지를 융선의 골과 마루에 대해 각각 세선화 하여 다음 과정의 입력으로 사용한다. 그림 5은 그림 4의 좌측 이미지를 융선의 골과 마루에 대해 각각 세선화한 결과를 보여주고 있다.

가. 융선의 마루 세선화 나. 융선의 골 세선화

그림 5 융선과 마루와 골의 세선화 결과

6. Minutiae Detection

지문인식을 위해서는 앞에서 언급한 바와 같이 *END*와 *BIF* 특징점을 추출해야 한다. 세선화된 이미지에서는 *CN*(*Crossing Number*)를 이용하여 쉽게 추출할 수 있다.

*CN*에 의해 추출된 특징점은 다음과 같은 형태의 구조를 가진다.

MP[i] = (type, x, y, theta, conf)

*type*은 *END*와 *BIF*를 구분하고, *x, y*는 탐지된 특징점의 좌표 값이다. *theta*는 탐지된 특징점에서의 방향성을 의미하며, *conf*는 추출된 특징점의 신뢰도를 의미한다.

MP[i].conf = ImgConf + DirConf

7. Minutiae Reduction

특징점 탐지에 의해 최초로 추출된 특징점에는 실제 지문에 존재하는 특징점 외에 손상된 지문 이미지와 이미지 처리 과정에서의 오류에 의한 가짜 특징점이 포함되어 있다. 이러한 가짜 특징점들은 매칭에 있어 성능과 소요시간에 있어 나쁜 영향을 미친다. 따라서 가짜 특징점은 반드시 제거해야 하며 제안된 시스템에서는 가짜 특징점 제거를 크게 5단계로 구성하였다.

1) 신뢰도가 낮은 영역의 특징점을 제거한다.

```
for i from 0 to nMP
    C_I=Σ_kΣ_lImgConf[MP[i]가 속한 블록]
    C_D=Σ_kΣ_lDirConfMP[i]가 속한 블록]
    if ( ImgConf[MP[i]가 속한 블록] = 0 ||
         DirConf[MP[i]가 속한 블록] = 0 ||
         (C_I+C_D) < T_C )
         MP(i) = false
    else MP(i) = ture
```

여기서 w_I와 w_D는 지문이미지 신뢰도와 방향성 신뢰도 계산에 사용된 블록의 크기를 나타내며, C_I와 C_D는 *ImgConf*와 *DirConf*의 합을 나타낸다. k, l은 $-w/2 \le k,l \le w/2$이며 w는 *MP* 주변의 인접한 신뢰도 블록의 개수를 나타낸다. T_C는 특징점으로 인정할 최소 신뢰도 합의 임계치이다. *nMP*는 특징점 탐지 과정에서 계산된 특징점의 개수이다.

2) 서로 인접한 특징점은 제거한다. 일정 거리 이내에 있는 두 특징점은 실제 지문 상의 특지점이 아닌 이미지 처리 과정에서 잘못 탐지된 특징점을 가능성이 높다.

```
for i from 0 to nMP
    if(DistanceCheck(MP[i])
        MP[i] = true
    else MP[i] = false
```

*DistanceCheck(·)*는 특징점 주변의 일정 거리 내에 다른 특징점이 있는지를 검사하여 일정 거리 내 다른 특징점이 없는 경우 *true*를 반환하는 함수이다.

3) Tico, Kuosmanen[15]의 특징점 제거 알고리즘을 적용한다.

4) 융선의 끊어짐에 의해 형성된 단점을 제거한다.

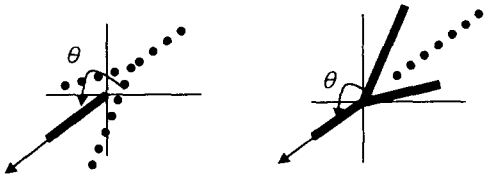
5) 골과 마루에서 탐지한 특징점을 이용하여 최종 가짜 특징점을 제거한다. 지금까지의 알고리즘을 융선의 골에 대해서도 적용한다. 이는 마루에서 탐지된 특징점은 골에서 탐지된 특징점과 쌍을 이루게 되므로 보다 신뢰도 있는 특징점을 탐지하는데 효율적이다. 즉 마루에서 *BIF* 타입의 특징점이 탐지되었다면 골에서는 *END* 타입의 특징점이 탐지되어야 한다.

```
for i from 0 to nMP
    if(ExistPairMP(MP[i]) &&
      MP[i].type != FindPairMP(MP[i]) )
      MP[i] = true
    else MP[i] = false
```

여기서 *ExistPairMP(·)*는 마루에서 탐지된 특징점 주변에 쌍을 이루는 골에서 탐지된 특징

점이 있는지를 검사하고 있는 경우 *true*를 반환하는 함수이다. *FindPairMP(·)*는 마루에서 탐지된 특징점과 쌍을 이루는 골에서 탐지된 특징점의 타입을 반환하는 함수이다.

융선의 골과 마루에 의해 탐지된 특징점 쌍을 이용하면 음의 방향을 고려하여 특징점의 방향을 설정할 수 있다. 먼저 *END*와 *BIF*는 다음과 같이 방향을 설정한다. 실선은 융선의 마루, 점선은 융선의 골을 의미하며 θ는 특징점이 갖는 각을 의미한다.



가. *END*의 경우 나. *BIF*의 경우

그림 6 *END*와 *BIF*의 방향각

방향성 필드에서 계산된 16방향에 음의 방향까지 고려하여 특징점에서의 방향은 32방향으로 세분화 된다. 이러한 특징점에서의 각은 제안된 시스템에서 가짜 삼각형을 탐지하는데 중요한 역할을 한다.

8. Matching

매칭은 입력 지문이미지에서 추출한 특징점을 템플릿에 저장된 특징점과 비교하여 같은 지문인지를 결정하는 단계이다. 매칭에 관한 내용을 여러 곳에서 제안하고 있으나[5][6][17][18][19] [20], 크게 특징점을 기반으로 하는 매칭 방법과 상관관계(Correlation)를 기반으로 하는 매칭 방법[16]이 있다. 제안된 시스템에서는 삼각형 기법[19][20]과 선분을 이용한 기법[24]을 응용하여 특징점의 일치 정도를 스코어로 계산하여 매칭여부를 결정하는 방식을 사용하였다.

1) 먼저 삼각형을 형성하기 위해 3개의 특징점을 선택한다.

가. 특이점이 두 개 이상인 경우, 첫 번째 특징점(q_1)은 두 특이점을 축으로 좌측 또는 우측에 위치한 특징점이 되도록 선택한다. 두 번째 특징점(q_2)은 특이점을 축으로 해서 q_1과 반대 사분면의 특징점이 되도록 선택한다. 세 번째 특징점(q_3)은 q_1과 q_2를 축으로 좌측 또는 우측에 위치한 특징점이 되도록 선택한다.

나. 특이점이 한 개인 경우, q_1은 특이점을 중심으로 일정 거리 떨어진 위치에 있는 특징점을 선택한다. q_2는 특이점을 중심으로 q_1과 반대 방향에 위치한 특징점이 되도록 선택한다. q_3는 q_1과 q_2를 축으로 좌측 또는 우측에 위치한 특징점이 되도록 선택한다.

다. 특이점이 없는 경우, q_1은 특징점들의 무게중심에 위치한 특징점이 되도록 선택한다. q2는 q_1과 일정 거리 떨어진 위치에 있는 특징점이 되도록 선택한다. q_3는 q_1과 q_2를 축으로 좌측 또는 우측에 위치한 특징점이 되도록 한다.

특징점은 신뢰도가 높은 것에 대해서만 우선 선택이 되며 해당 위치에 특징점이 없는 경우는 신뢰도가 낮은 특징점에 대해서 선택을 한다.

2) 1)에서 찾은 세 특징점을 이용하여 삼각형의 합동 여부를 검사한다.

3) 2)에 의해 찾아진 매칭되는 특징점 쌍을 중심으로 스코어를 계산한다.

MC_Q는 템플릿에 있는 특징점과 매칭되는 쌍이 있는 입력 지문이미지의 특징점 신뢰도의 합, MC_T는 입력 지문이미지의 특징점과 매칭되는 쌍이 있는 템플릿 특징점 신뢰도의 합, NMC_Q는 템플릿에 있는 특징점과 매칭되는 쌍이 없는 입력 지문이미지의 특징점 신뢰도의 합, NMC_T는 입력 지문이미지의 특징점과 매칭되는 쌍이 없는 템플릿 특징점 신뢰도의 합을 나타낸다. 실제 겹치는 영역에 속한 특징점만이 스코어 계산에 사용된다. *MC*는 매칭된 쌍에 대해서는 양의 스코어를, 그렇지 않은 쌍에 대해서는 음의 스코어를 부여하여 계산한다. *TC*는 공통영역의 모든 특징점의 신뢰도 합을 나타낸다. 따라서 *MC*와 *TC*는 다음과 같이 계산된다.

$$MC = a \times MC_Q + b \times MC_T - c \times NMC_Q - dNMC_T$$

$$TC = a \times MC_Q + b \times MC_T + c \times NMC_Q + dNMC_T$$

$$SCORE = (MC/TC) \times 100$$

*a, b, c, d*는 가중치로 적절한 값을 설정한다. 제안된 시스템에서는 *(a,b,c,d)=(8,4,-2,-1)*로 설정했다.

4) 계산된 스코어에 의해 매칭 여부를 결정한다.

9. Template Updating

매칭의 결과가 *ACCEPT*라면 특징점과 특이점의 신뢰도를 바탕으로 템플릿 특징점과 특이점을 업데이트 한다.

1) 입력 지문이미지에서 탐지된 특징점 중 템플릿 특징점과 매칭되는 쌍이 없는 특징점은 템플릿에 추가한다.

$$newMP_T.conf = NPMP_Q.conf \;/\; 2$$

$newMP_T$는 템플릿에 새로 추가되는 특징점이고 $NPMP_Q$는 매칭되는 쌍이 없는 입력 지문이미지의 특징점이다.

2) 템플릿 특징점들 중 입력 지문이미지의 특징점과 매칭되는 쌍이 있는지의 여부에 따라 신뢰도 및 좌표를 업데이트 한다.

$$X = PMP_T.conf \times PMP_T.x + PMP_Q.conf/2 \times PMP_Q.x$$

$C=PMP_{T}.conf+PMP_{Q}.conf/2$

$updatedMP_{T}.x = X / C$

$updatedMP_{T}$는 업데이트한 템플릿 특징점이며 PMP_{T}와 PMP_{Q}는 각각 템플릿과 입력 지문이미지의 매칭되는 쌍의 특징점이다. 위와 같은 방식으로 $PMP_{T}.y$, $PMP_{T}.theta$를 업데이트 한다. 특징점의 신뢰도는 다음과 같이 업데이트 한다.

$updatedMP_{T}.conf=Max(PMP_{T}.conf,PMP_{Q}.conf/2)+1$

3) 특이점을 업데이트 한다. 특이점 역시 특징점과 같은 방식으로 템플릿에 없는 특이점은 템플릿에 추가하며 매칭되는 특이점 쌍이 있는 경우 템플릿의 특이점을 업데이트 한다. 최초 *0x1*이라는 신뢰도를 갖게 된다. 템플릿 특이점 업데이트는 매칭되는 쌍의 유무에 따라 다르다.

가) 매칭되는 특이점 쌍이 있는 경우는 다음과 같이 업데이트 한다.

$W_{T} = CalcHW(MSP_{T}.conf)$

$W_{Q} = CalcHW(MSP_{Q}.conf)$

$updatedSP_{T}.x=(W_{T}\times MSP_{T}.x+W_{Q}\times MSP_{Q}.x)/(W_{T}+W_{Q})$

$updatedSP_{T}.y=(W_{T}\times MSP_{T}.y+W_{Q}\times MSP_{Q}.y)/(W_{T}+W_{Q})$

$updatedSP_{T}.conf=(MSP_{T}.conf<<1)+1 \bmod S_{c}$

$CalcHW(\cdot)$는 신뢰도를 이진 표기 시 *1*이 나타나는 위치에 가중치는 곱하여 계산한 값을 반환하는 함수이다. 즉 *LSB*에 가중치 *1*을 곱하고 *MSB*에 이진 표기 시 자릿수만큼의 가중치를 곱한다. S_{c}는 특이점 신뢰도의 크기를 나타낸다.

나) 매칭되는 특이점 쌍이 없는 경우는 특이점의 신뢰도만 감소시킨다.

IV. 제안된 지문인식 시스템의 결과 및 분석

1. 구현된 시스템의 속도

제안된 시스템은 C언어를 기반으로 작성되었으며 MS C++ 6.0에서 컴파일 되었다. 테스트 환경은 Intel Pentium II 300MHz Processor에서 수행하였으며 입력 지문이미지의 크기는 292*248 픽셀이다. 각 프로세스의 평균 수행시간은 표 1와 같다.

Process	시간(sec)	비율(%)
Image Confidence	0.081	3.9
Orientation Field	0.240	11.4
Image Enhancement	0.070	3.3
Binarization	1.192	56.7
Thinning	0.450	21.4
Minutiae Detection	0.030	1.4
Minutiae Reduction	0.020	1.0
Matching	0.100	4.8
Total	2.183	104

표 1 프로세스별 수행속도 및 비율

표 1에 제시된 프로세스 수행속도에서 *Thinning, Minutiae Detection & Reduction*은 융선의 골과 마루에 대해 각각 1번씩 수행한 시간을 기록한 것이므로 실제 알고리즘의 수행시간은 제시된 시간의 절반이다. *Matching*의 경우 템플릿과 입력 지문이미지에 따라 시간차를 보이고 있으나 *ACCEPT*해야 할 매칭 시도인 경우 평균 28ms 정도의 시간이, *REJECT*해야 할 매칭 시도인 경우 평균 64ms 정도의 시간이 소요된다. 매칭 소요 시간의 효율성을 고려하여 *TIMEOUT*을 설정하여 매칭에 사용되는 시간을 제안하였다. 표 1은 *TIMEOUT*을 100ms로 설정한 경우이다. 따라서 실제 매칭 소요시간은 약 45ms 정도 소요된다고 볼 수 있다. *FA*(False Accept)인 경우, *ACCEPT* 결과를 내기위해 매칭에 많은 시간을 소요하는데 이를 *TIMEOUT*을 설정함으로써 효율적으로 *REJECT*할 수 있다. *TIMEOUT*의 크기에 따라 *FAR/FRR*의 결과가 다소 차이를 보이고 있다.

2. 구현된 시스템의 안전성

지문인식 시스템을 평가하는 검사목록은 여러 가지가 있으나[22][23], 제안된 시스템에서는 크게 두 가지, *FAR*(False Accept Ratio)과 *FRR*(False Reject Ratio) 측면에서만 분석한다.

$FRR = Prob(number\ of\ FRs\ /\ A_{0})$

$FAR = Prob(number\ of\ FAs\ /\ A_{1})$

*FR*은 인가 받은 사용자가 *ACCEPT* 되지 않은 경우, *FA*는 인가 받지 않은 사용자가 *ACCEPT*된 경우이다. A_{0}은 인가 받은 사용자가 *ACCEPT*를 획득하기 위해 매칭을 시도한 횟수이며 A_{1}은 인가 받지 않은 사용자가 *ACCEPT*를 획득하기 위해 매칭을 시도한 횟수이다.

제안된 시스템에서 사용한 지문이미지는 상업용 광학 센서(NITGEN, FDU01A 모델)를 이용하여 직접 지문을 입력받아 사용하였다.

1) *FRR* 테스트

조 건	*FRs/A0*	*FRR(%)*
CASE 1	*23/2450*	*0.98*
CASE 2	*76/2450*	*3.10*
CASE 3	*131/2450*	*5.35*

표 2 *FRR* 테스트 결과 (*TIMEOUT*:100ms)

표 2에서 CASE는 다음과 같다. *nMT*는 합동인 삼각형의 개수를 의미하며 $SCORE_{Ave}$는 합동인 삼각형에 의해 산출된 *SCORE*의 평균을 의미한다.

CASE 1: nMT $\geq$ *3 && SCORE*$_{Ave}$ $\geq$ *30*

CASE 2: nMT $\geq$ *3 && SCORE*$_{Ave}$ $\geq$ *50*

CASE 3: nMT $\geq$ *5 && SCORE*$_{Ave}$ $\geq$ *50*

2) *FAR* 테스트

조 건	*FAs/A1*	*FAR(%)*
CASE 1	*4/2450*	*0.16*
CASE 2	*130/2450*	*5.31*

표 3 *FAR* 테스트 결과 (*TIMEOUT*:100ms)

CASE 1: nMT $\geq$ *3 && SCORE*$_{Ave}$ $\geq$ *30*

CASE 2: nMT $\geq$ *3 && SCORE*$_{Ave}$ $\geq$ *10*

V. 결론

본 논문에서는 신뢰도 개념을 적용한 지문인식 알고리즘을 제안 및 구현했다. 제안된 지문인식 시스템은 *Galton feature*를 사용하여 특징점 기반의 지문인식 시스템으로 *Binarizaiton, Thinning*의 이미지 처리 과정과 *Minutiae Detection*, Minutiae *Reduction, Singular Point Detection, Singular Point Reduction* 과정을 통하여 매칭에 필요한 특성을 추출하였다. *Image Confidence*와 *Direction Confidence*의 개념을 적용하였고, 특징점 탐지 시 융선의 골과 마루를 이용하여 보다 효율적, 믿을 수 있는 처리 과정을 구현하였다. 신뢰도의 개념을 *SCORE* 계산 및 *Template Update* 과정에 반영함으로써 보다 신뢰성 있는 지문인식 시스템을 구현하였다.

참고문헌

[1] F. Galton, "Finger Prints", Macmillan, London, 1892

[2] E. R. Henry, "Classification and Uses of Finger Prints", Routledge, London, 1990

[3] Federal Bureau of Investigation, "The Science of Fingerprints: Classification and Uses", Washington D.C., 1984

[4] P. Baldi, Y. Chauvin, "Neural Networks for Fingerprint Recognition", Neural Computation, 1993, pp. 402~418

[5] A. Jain, L. Hong, S. Pankanti, R. Rolle, "An identity-authentication system using fingerprints", Processdings of IEEE, vol. 85, pp. 1365~1388, September 1997.

[6] D. Maio, D. Maltoni, "Direct Gray-Scal Minutiae Detection in Fingerprints", IEEE Transactions on Pattern Analysis Machine Intelligence, Vol.19, No.1, pp.27~40, 1997

[7] D. Maio, D. Maltoni, "Minutiae extraction and filtering from gray-scale images", Intelligent Biometric Techniques in Fingerprint & Face Recognition, CRC Press, 1999

[8] M. Kass, A. Witkin, "Analyzing Oriented Patterns", Computer Vision, Graphics and Image Processing, Vol 37, No. 4, pp.362~385, 1987

[9] M. Kawagoe, A. Tojo, "Fingerprint Pattern Classification", Pattern Recognition Vol. 17, No. 3, pp.295~303, 1984

[10] A. R. Rao, "A Taxonomy for Texture Description and Identification", SpringerVerlag, New York, 1990

[11] L. Hong, Y. Wan, A. K. Jain, "Fingerprint Image Enhancement: Algorithm and Performance Evaluation", IEEE Trans. Pattern Anal. and Machine Intell., Vol. 20, No. 8, pp.777~789, 1998

[12] Zhang, Suen, "A Fast Parallel Algorithm for Thinning Digital Patterns" Commun. ACM Mar 1984 Vol 27 No 3, pp 236-239

[13] Koushik Das, "Design and Implementation of an Efficient Thinning Algorithm", 2000

[14] 한낙희, 이필규, "가중치를 이용한 병렬세선화 알고리즘", 인지과학회, pp.5-35, Jan, 1996.

[15] Marius Tico, Pauli Kuosmanen, "An Algorithm for Fingerprint Image Postprocessing", Proceedings of the Conference record of The Thirty-Fourth Asilomar Conference on Signals, Systems & Computers - Volume 2 , 1735-1739 , 2000. 10

[16] Salil Prabhakar. "Fingerprint Classification and Matching Using a Filterbank", Ph.D. Thesis, 2001

[17] B. Chatterjee, B. Mehtre, "Automatic fingerprint identification", Journal of the Institution of Electronics and Telecom., 37(5/6):493, 1991

[18] A. K. Jain, S. Prabhakar, L. Hong, S. Pankanti, "Filterbank-based Fingerprint Matching", IEEE Trans. Image Processing, Vol. 9, No. 5, pp.846~859, May 2000

[19] B. Bhanu and X. Tan, "A triplet based approach for indexing of fingerprint database for identification," International Conference on Audio- and Video-Based Biometric Person Authentication, pp.205-210, 2001.

[20] Zsolt Miklos Kovacs-Vajna, "A Fingerprint Verification System Based on Triangular Matching

and Dynamic Time Warping," IEEE Transaction on Pattern Analysis and Machine Intelligence, VOL. 22, No. 11, pp 1266 - 1276, November 2000.

[21] L. Hong, A.K. Jain, S. Pankanti and R. Bolle, "Fingerprint Enhancement", Proc. IEEE Workshop on Applications of Computer Vision, Sarasota, Fl, pg. 202-207, Dec. 1996.

[22] ANSI X9.84, "Biometric Information Management and Security", 2001

[23] Maio Maltoni Cappelli, "FVC2002: Second Fingerprint Verification Competition", 2002

[24] Raghavendra Udupa, Gaurav Garg and Pramod Kumar Sharma "Fast and Accurate Fingerprint Verification" Audio and Video Based Biometric Person Authentication (AVBPA) June 2001