지문을 이용한 자바카드 인증 시스템의 설계 및 구현

배재형[0] 김증섭 배인구 조병호 이주형[1] 유기영
경 북 대 학 교 컴 퓨 터 공 학 과
{bezant, dambi, demon, bhcho}@purple.knu.ac.kr, yook@knu.ac.kr

The Java Card Authentication System Design and Implementation Using Fingerprint Image

Jae-Hyoung Bae[0] Jeung-Seop Kim In-Gu Bae Byoung-Ho Cho
Joo-Hyoung Lee[1] Kee-Young Yoo
Dept. of Computer Engineering, Kyungpook National University
Korea Information System[1]

요 약

지문은 사람에 있어서 유일한 특징과 변하지 않는 특성으로 인하여 개인의 인증이나 식별에 많이 사용되고 있다. 이러한 특성을 가진 지문을 스마트 카드의 인증과정에 이용함으로써 보안성을 높이고 사용에 편리함을 추가할 수 있다. 본 논문에서는 지문의 특징점 추출 과정과 매칭 방법 그리고 자바카드에 지문을 등록하는 프로토콜과 인증하는 프로토콜을 제시하고 구현하였다.

1. 서론

스마트 카드는 프랑스에서 1974년 최초로 개발된 이후 프랑스를 중심으로 유럽에서는 은행카드로 사용되기 시작하여 전세계적으로 스마트카드의 시장이 날로 늘어나고 있고, 그 사용도 매년 15%씩 증가하는 추세에 있다[1]. 스마트 카드를 사용하기 위해 사용자 인증 과정에서 PIN(personal identification number)을 이용한 기존의 방법은 사람의 망각이나 부 주위로 인하여 인증에 많은 위험과 문제점을 가지고 있다[2]. 카드 사용에 대한 보안을 높이고 사용하기 편리한 방법이 필요하다.

사용자 인증과정에서 더 높은 보안성과 편리성을 제공하기 위해 사람의 신체 특징을 이용하고 있다. 신체 특징을 이용한 인증 방법에는 지문, 손 모양, 얼굴, 목소리, 망막, 서명 등이 있으며, 특히 지문은 유일성과 불변성으로 인하여 이미 개인 인증이나 식별에 널리 사용되고 있다. 지문에서 특징 정보를 추출하여 카드의 인증과정에 이용하면 보안성 뿐만 아니라 사용의 편리함을 더 높일 수 있다.

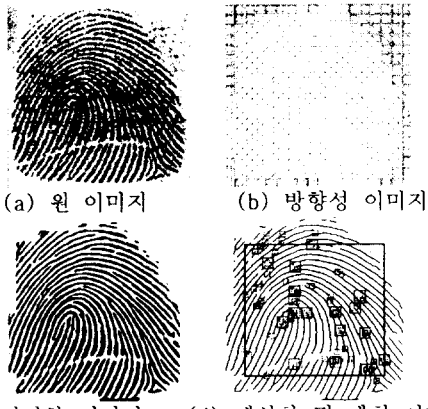
본 논문에서는 지문에서 특징점을 추출하여 스마트 카드인 자바카드 내에 저장하고, 사용자 인증에 그 정보를 이용하는 방법과 그 과정을 제시하고 구현한다.

본 논문의 구성은 다음과 같다. 2장에서는 지문에서 특징점을 추출하는 과정과 매칭 방법에 대해서 설명하고, 3장에서는 자바카드를 이용하는 등록 프로토콜과 인증 프로토콜을 설명한다. 4장에서는 실험 및 결과에 대해서 설명하고 5장에서 결론과 향후 연구 과제에 대해서 설명한다.

2. 지문에서 특징점 추출과 매칭

일반적으로 지문이미지는 Gray level 로 입력되고 이 이미지를 지문 매칭 정보로 사용하기에는 부적절하다. 정보의 양이 적으면서 지문 매칭에 사용할 특징점을 추출하기 위해서 전처리 과정이 필요하다.

전처리 과정에는 방향성 추출, 이진화, 세선화 단계로 구성되고 세선화된 이미지에서 지문의 특징점을 추출하고, 그 정보를 가지고 지문 매칭 단계를 거쳐 사용자 인증을 한다. 그림 1은 원 이미지와 방향성과 전경배경을 분리한 이미지, 이진화 이미지, 세선화 후 특징점을 찾고 매칭한 이미지이다.



(a) 원 이미지 (b) 방향성 이미지

(c) 이진화 이미지 (d) 세선화 및 매칭 이미지

그림 1. 특징점 추출과 매칭

2.1 전처리 과정

방향성 추출 단계는 지문 이미지를 8x8 픽셀 블록으로 나누고, 각 블록의 대표방향을 구하기 위해서 그림 2(a)의 Sobel연산자를 이용한다. 각 블록의 대표 방향은 그림 2(b)의 식으로 구할 수 있고[3], 대표 방향의 값은 8방향(0~7)이다. 인접한 대표 방향의 평균이 임계값 보다 작으면 배경으로 하고 아니면 전경이다.

$$Sobel_x = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}, \quad Sobel_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}$$

(a) Sobel 연산자

$$V_y(i,j) = \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} 2\partial_x(u,v)\partial_y(u,v)$$

$$V_x(i,j) = \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} \left(\partial_x^2(u,v) - \partial_y^2(u,v)\right)$$

$$\theta(i,j) = \frac{1}{2}\tan^{-1}\left(\frac{V_y(i,j)}{V_x(i,j)}\right)$$

(b) 방향성 추출 식

$$M(i,j) = \frac{1}{w \times w} \sum_{u=i-\frac{w}{2}}^{i+\frac{w}{2}} \sum_{v=j-\frac{w}{2}}^{j+\frac{w}{2}} \sqrt{\partial_x^2(u,v) + \partial_y^2(u,v)}$$

(c) 전경 배경 분리 식

그림 2. 방향성 추출, 전경배경 분리(w=8)

구해진 방향성 정보는 잡음이나 상처로 인하여 잘 못된 방향을 나타낼 수도 있다. 이것을 보정하기 위하여 그림 3의 식을 이용한다.

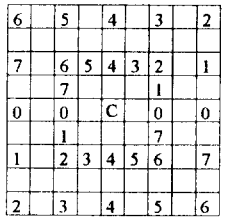
$$S_{\sin} = \sum_{u=i-1}^{i+1} \sum_{v=j-1}^{j+1} \sin 2\theta(u,v)$$

$$S_{\cos} = \sum_{u=i-1}^{i+1} \sum_{v=j-1}^{j+1} \cos 2\theta(u,v)$$

$$O(i,j) = \frac{1}{2}\tan^{-1}\left(\frac{S_{\sin}}{S_{\cos}}\right) \text{ if } O(i,j)<0 \text{ then } O(i,j) += \pi$$

그림 3. 방향성 보정

이진화 단계는 Gray level 값을 이진화 값으로 바꾸는 단계이다. 그림 4 의 이진화 마스크를 사용하여 해당 픽셀을 중심으로 8 방향 32 픽셀의 평균값과 S_{min}과 S_{max}를 구하고 해당 픽셀 값이 평균값보다 크면 흰색으로 아니면 검은색으로 한다[4].



$$4C + s_{\min} + s_{\max} > \frac{3}{8}\sum_{i=0}^{7} s_i$$

그림 4. Binarization Mask 와 계산식

세선화 단계는 융선의 두께를 1 픽셀 두께로 만드는 과정이다. 그림 5 와 같이 픽셀(i,j)의 주위 픽셀들의 패턴을 하나의 바이트로 맵핑할 수 있다. 이렇게 하면 이진화 이미지에서 있는 모든 3x3 블록들의 256 가지 패턴들을 미리 테이블로 만들 수 있다. 이렇게 만들어진 테이블에서 미리 지울 수 있는 경우와 지울 수 없는 경우를 계산하여 참조표를 구성하여 놓고, 실제 세선화를 수행할 때에는 단지 중심 픽셀을 제외한 주위 8 픽셀을 보고 맵핑한 값과 참조표를 보고 지울 것인지 아닌지를 판단할 수 있다.

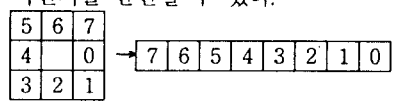


그림 5. 3x3 블록 맵핑

2.2 특징점 추출

지문 이미지에서 특징점은 융선의 단점과 분기점이다. 그림 6(a)의 3x3 마스크를 적용하여 중심점 C에 대해 시계 반대 방향으로 X_1~ X_8에 위치한 픽셀을 참조하여 그림 6 (b),(c)와 같이 계산한다.

X_1	X_2	X_3
X_8	C	X_4
X_7	X_6	X_5

(a)특징점 마스크와 추출된 이미지

$$C_n = \sum_{i=1}^{8} |X_{i+1} - X_i| = 2 \text{ where } X_9 = X_1$$

(b) 단점 추출식

$$C_n = \sum_{i=1}^{8} |X_{i+1} - X_i| = 6 \text{ where } X_9 = X_1$$

(c) 분기점 추출식

그림 6. 특징점 추출

이러한 특징점을 다음과 같은 자료구조로 만들어서 사용한다.

```
typedef struct Minutia {
        BYTE x;
        BYTE y;
        BYTE angle;
        BYTE type;
} MINUTIA;
```

2.3 매칭과정

두 지문 이미지에서 추출된 특징점을 서로 비교하여 매칭하는 방법은 먼저 회전과 이동된 정도를 알아내고, 한 지문 이미지의 특징점들을 이 값으로 회전변환과 이동변환을 거쳐 특징점의 위치가 매칭이 되는지 계산하고 이렇게 구해진 특징점들을 가지고 매칭 정도(matching score)와 두 특징점이 공통적으로 분포하는 공통넓이를 계산한다[5]. 이 두 정보를 가지고 일정한 넓이에 걸쳐 매칭되고 매칭 정도가 어느 정도 높으면 두 지문이 맞는 것으로 한다.

3. 자바카드에서 등록 및 인증 프로토콜

자바카드[6]는 Java 프로그래밍 언어로 쓰여진 프로그램을 실행 할 수 있는 스마트 카드이다. Java 언어를 사용함으로써 OOP의 장점과 플랫폼 비의존성 그리고 하나의 카드 내에 여러 개의 애플릿을 넣을 수 있는 장점들을 가지고 있다. 개발과정은 javacard와 javacardx Package를 포함시켜서 원하는 어플리케이션을 작성한 후, 컴파일해서 바이트 코드로 만들고, 카드 바이너리 파일로 변환한 후,

카드에 로딩 및 인스턴스를 생성한다. 그 후 여러 가지 명령어를 사용하여 애플릿과 카드에 원하는 기능을 수행시킬 수 있다. 본 연구에서는 Schlumberger에서 개발한 Cyberflex[7]를 이용하여, 지문 이미지에서 추출한 정보를 저장, 로딩하는 기능 등을 추가 하였다.

3.1 등록 프로토콜

자바카드에 특징점 정보를 등록하기 위해서 필요한 명령어들을 추가하고, 자바카드에 정보를 보내기 위해서는 그림5와 같은 프로토콜로 전송을 한다. 이때 등록하는 주체는 공인 받은 곳에서 행해져야 한다.

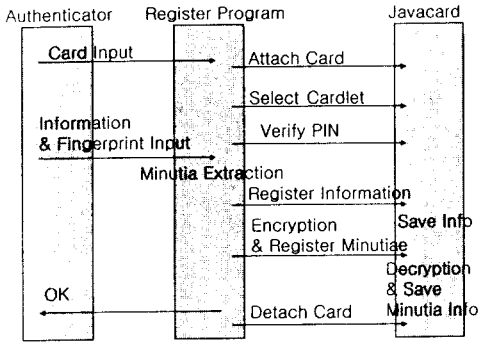


그림 5. 등록 프로토콜

3.2 인증 프로토콜

카드의 올바른 사용자 임을 인증하기 위해서 그림 6과 같이 사용자로부터 카드를 입력 받고 카드가 올바른 카드인지를 확인한 다음 사용자로부터 지문 이미지를 입력 받는다. 그리고 인증 프로그램은 자바카드로부터 이미 등록되어 있는 지문 특징점을 읽어와서 입력 받은 지문과 매칭 과정을 거친다. 올바른 사용자임이 인증되면 그 후에 카드가 할 수 있는 서비스를 제공하게 된다.

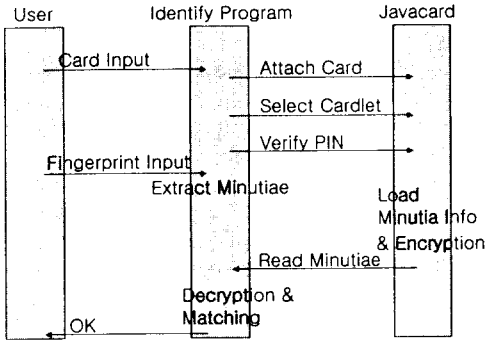


그림 6. 인증 프로토콜

4. 실험 및 결과

지문이미지는 Veridicom사의 반도체 방식의 지문입력 칩인 FPS110을 사용한 제품을 통하여 300 x 300 픽셀의 Gray level의 값을 입력 받는다[8].

지문 매칭의 FAR(false accept rate)를 측정하기 위해서 자바 카드를 이용하지 않고 지문 입력기로부터 87개의 다른 지문을 각각 1회에서 5회까지 입력 받아 377개의 지문 이미지를 파일로 저장하고 서로 간에 매칭을 수행하여 결과를 실험한다. 본 시스템은 Windows(95, 98, NT)환경에서 사용 가능하도록 마이크로소프트사의 Visual C++ 6.0으로 개발하였고, 실험은 Pentium III 450 퍼스널 컴퓨터에서 실행 되었다.

임의의 지문을 다른 사람의 지문 그룹 7개와 매칭했을 경우 매칭정도(MS)가 10퍼센트를 넘는 경우가 없었다. 표 1.은 실험한 그룹들에서 매칭한 지문의 MS값이 0~10%사이의 값을 나타낸 것의 분포를 나타내고 있다.

전처리와 특징점 추출 과정을 동시에 수행하는 시간은 약 590ms가 걸렸고, 매칭과정은 약 280ms가 걸렸다.

표1. 서로 다른 지문사이의 MS(%) 분포

	0%	1%	2%	3%	4%	5%	6%	7%	8%
그룹1	342	17	12	1	0	0	0	0	0
그룹2	345	18	4	4	0	0	1	0	0
그룹3	362	8	1	1	0	0	0	0	0
그룹4	363	7	1	1	0	0	0	0	0
그룹5	347	20	5	0	0	0	0	0	0
그룹6	366	7	0	0	0	0	0	0	0
그룹7	350	17	4	1	1	0	0	0	0
합	2475	94	27	8	1	0	1	0	0

(8%이상 모두 0)

5. 결론 및 향후 연구과제

본 논문에서는 지문이미지에서 특징점을 추출하고 자바카드 내에 그 정보를 저장하는 등록 프로토콜과 인증 프로토콜을 구현하였다. 카드에 등록한 사용자가 아닌 다른 사용자를 잘못 인증하지 못하도록 MS의 값을 조정할 수 있다. 카드의 메모리 용량과 계산 속도 때문에 지문 매칭을 터미널에서 수행했지만, 특징점 정보를 카드가 받아서 매칭하는 것이 정보 유출을 더욱 막을 수 있으므로, 카드 내에서 적은 용량으로 빠르게 수행하는 지문 매칭 방법이 향후 연구해야 할 과제이다.

6. 참고 문헌

[1] W. Rankl, and W. Effing, *Smart Card Handbook*, Chanterelle Translations, London, UK, 1997.

[2] 최정호, " 데이터보안 위한 생체측정 보안시스템", 경영과 컴퓨터, pp.96-100, 1992

[3] Lin Hong," Automatic Personal Identification using Fingerprint", Ph.D thesis, Michigan State University, 1998.

[4] R. M. Stock and C. W. Swonger. " Development and evaluation of a reader of fingerprint minutiae, " *Cornell Aeronautical Laboratory*, Technical Report CAL No. XM-2478-X-1:13-17. 1969.

[5] nalini, K.R., KARU,K., CHEN, S., and JAIN, A.K.: ' A real-time matching system for large fingerprint databases' ,IEEE Trans. Pattern Anal. Mach. Intell., 1996, 18,(8). Pp. 799-813

[6] Sun Microsystems, *Java Card 2.0 Programming Concept*, 1997

[7] Schlumberger, *Cyberflex™ Access Developer 's Series Programmer' s Guide*, 1998

[8] Veridicom, *Software Developer' s Kit(SDK) Manual* 1998.