임베디드 리눅스 기반의 지문 인식 시스템 구현

Implementation of Fingerprint Cognition System Based on Embedded LINUX

*배 은 대, **김 정 하, ***남 부 희
(Eun Dae Bae · Jeoung Ha Kim · Boo Hee Nam)

Abstract - In this paper, we have designed a fingerprint cognition system based on the embedded Linux. The proposed algorithm in this paper use the wavelet transform to derive the special feature vector from the captured fingerprint and a probabilistic neural network is used to compare the feature vectors for the fingerprints. The system consists of server PC based on the Linux and the client based on the embedded Linux. The client is a Tynux box-x board using the PXA-255 CPU. For the acquisition of the fingerprint image, we used the AS-S2 semiconductor sensor. The system is likely to be used to develop a police inspection system.

Key Words : Embedded System, Fingerprint, Wavelet transform, Probabilistic neural network

1. 서 론

최근 들어 급격히 발전하고 있는 임베디드 시스템을 이용한 시스템들은 기존의 PC보다 저하되는 환경을 가진다.

이러한 임베디드 시스템 기반의 지문인식 시스템을 구축하는데 있어서의 어려운 점은 지문 데이터의 방대한 양을 보다 빠르게 획득하고, 보다 정확하게 비교하는데 있어서의 문제점에 기인하다. 본 논문의 목적은 휴대가 가능한 임베디드 시스템에서, 지문을 보다 빠르게 인식, 비교할 수 있도록 하는 것이다. 본 논문에서는 이러한 조건을 만족시키기 위해, 두가지 방법을 채택했다. 첫째는 Wavelet Transform 이고, 둘째는 Probabilistic Neural Network 이다. 그림 1은 본 논문의 전체적인 구조를 나타낸 것이다.

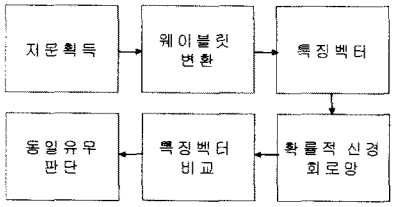

그림 1 시스템 구조
Fig 1. System architecture

시스템의 하드웨어적인 구성은 크게 LINUX 기반의 서버와, Embedded LINUX 기반의 클라이언트로 나뉜다[2]. 지문 이미지를 받아들이기 위하여, AF-S2 반도체식 센서를 사용하였다[3]. 센서로부터 받아들인 128*128 Byte 크기의 지문 이미지의 특징벡터를 축출하기 위해 Wavelet Transform이 사용된다[4]. 마지막으로, 획득한 특징벡터와 기존의 지문 이미지의 특징 벡터를 비교하기 위하여 Probabilistic Neural Network를 채택하였다[5].

2. 하드웨어 구성

본 논문에서는 사용된 시스템의 하드웨어 구성은 크게, 서버와 클라이언트로 나뉜다. 서버로는, Pentium-3 프로세서를 사용하는 PC를 사용하며, Wow LINUX 7.3을 OS로 사용한다. 서버에서는 개발보드의 환경과 같이 동작하는 에뮬레이터가 설치되어 있어 또 다른 클라이언트로 사용될 수도 있다. 클라이언트로는 PXA255 프로세서를 사용하는 Tynux사의 Box-x 개발보드를 사용한다. 클라이언트의 OS로는 Embedded LINUX를 사용하며, 그래픽 라이브러리인 Qtopia를 사용한다. 클라이언트로 사용되는 Box-x는 터치스크린 방식의 LCD를 내장하고 있어, 프로그램의 실행을 보여주며, 획득한 지문이미지를 보여주는 역할을 할 수 있다.

서버와 클라이언트 사이에는 세 가지 통신 기법이 사용된다. 첫째는 Serial Port를 이용한 통신이며, 클라이언트의 동작을 모니터링 하여 서버에서 볼 수 있도록 하기 위해 사용된다. 둘째는 Parallel Port를 이용한 통신이며, 클라이언트에 Bootloader를 올리기 위해 사용된다. 마지막으로, 이더넷을 이용한 통신이며, 서버에서 개발한 지문 인식 프로그램을 클라이언트로 전송하기 위해 사용되며, 또한 클라이언트에서 획득한 특징 벡터를 서버로 전송하기 위해서도 사용된다.

저자 소개
* 準 會 員 :강원공대 통신,멀티미디어과 석사 배 은 대
**準 會 員 :(주)KTV 연구원 김 정 하
***正 會 員 :강원공대 전기전자정보통신공학부 교수 남 부 희

전체적인 하드웨어 구성은 그림 2 에 있다.

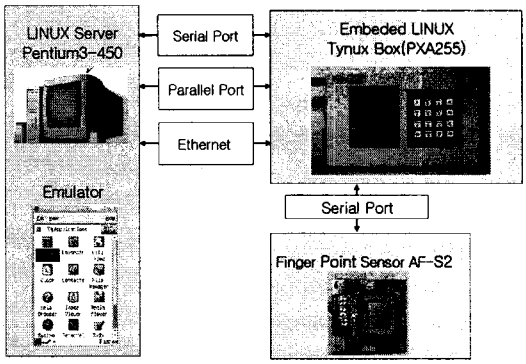

그림 2 하드웨어 구성
Fig 2. Hardware architecture

3. 지문 인식 센서

임베디드 시스템에서의 지문인식을 위한 이미지 센서는 보드와의 연동 및 확장성이 용이하고, 크기가 작고 처리속도가 빠르며, 전력 소모가 거의 없는 등의 여러 특징을 가져야 한다. 본 논문에서는 이러한 특징을 가지고 있어 임베디드 시스템에 좀 더 유리한 Authentic 사의 반도체식 센서 AF-S2를 사용한다.

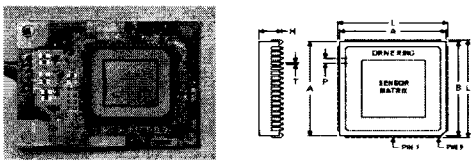

그림 3 지문 인식 센서
Fig 3. Fingerprint cognition sensor

본 논문에서는 실시간으로 지문 이미지를 획득하여야 한다. 획득한 지문 이미지는 128*128 크기의 그레이 레벨의 이미지다. 이러한 이미지를 실시간으로 획득하는데 있어서 Serial Interface를 사용한다. 본 논문에서 사용한 이미지 데이터의 양은 Serial Interface를 이용하더라도 충분히 실시간으로 획득할 수 있다. AF-S2 센서는 eight-bit data, one start bit, one stop bit 그리고, no parity를 사용한다.

4. 웨이블릿 변환

본 논문에서, 센서를 통해 획득한 지문 이미지는 8 gray level을 가지는 raw 파일로 저장이 된다. 이 영상을 가지고 이진화를 시켜 지문 알고리즘에 적용한다. 보통 영상에 생길 수 있는 잡음 등의 이유 때문에 전처리 과정을 거치게 된다. 하지만 본 논문에 사용한 AF-S2 센서를 사용할 경우 입력받은 영상은 8 gray level 이므로, 전처리 과정의 큰 효과를 기대할 수 없다. 보통 이진화를 하는 이유는 우선 gray level의 이미지의 데이터양의 축소를 위함이고, 또한 이미지에 들어 있는 필요 없는 잡음들을 제거하는데 효과적이기 때문이다. 본 논문에서는 임계값(threshold)으로 0과 255의 중간값인 127을 사용한다. 이러한 임계값 처리는 입력 영상의 각 화소에 대해 명도가 있는 일정값 이상의 경우 대응하는 출력 영상의 화소 값을 1로서, 그 외의 경우는 0으로 만드는 것이다. 임계값 처리를 식으로 나타내면 다음과 같다.

$$g(x,y)=\begin{cases}1 & f(x,y)\geq t\\ 0 & f(x,y)\leq t\end{cases} \quad (1)$$

이진화를 통해 얻은 데이터는 웨이블릿 변환을 이용하여 특징벡터로 만들어진다. 웨이블릿 이란 데이터를 다른 주파수 성분으로 분해하고, 그 크기에 맞는 해상도를 이용하여 각각의 성분을 연구하는 수학적인 함수이다. 웨이블릿 변환은 신호의 시간-주파수 표현을 제공하여 동이세 시간과 주파수에 대한 정보를 제공할 수 있다. 많은 신호에 있어서, low-frequency 성분은 중요한 데이터를 담고 있다. 본 논문에서는 이러한 이유로 low-frequency성분인 Approximation 부분만을 축출하는 방법을 사용하며, 그림 4에 나타내었다.

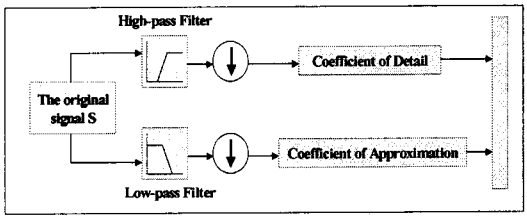

그림 4 웨이블릿 변환 과정
Fig 4. Decomposition of a signal for Wavelet Transform

본 논문에서는 지문 이미지의 128bit의 행 데이터를 7번의 웨이블릿 변환을 거쳐 하나의 데이터로 만든다. 그리고 나머지 127개의 행 데이터도 같은 과정을 거친다. 마지막으로 각 행별로 얻은 하나의 데이터를 모아 5번의 웨이블릿 변환을 수행한다. 이렇게 하여 얻어진 4개의 데이터를 특징벡터로 부르며, 지문 비교에 사용된다. 본 논문에서는 Harr 웨이블릿 변환을 사용한다. 그림 5에 특징벡터를 얻는 과정을 도식화 하였다.

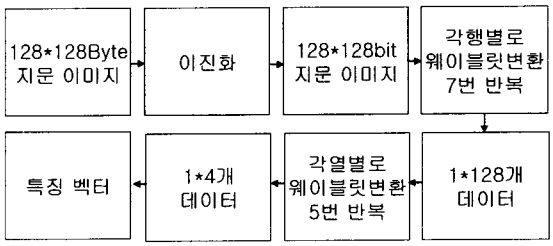

그림 5 특징 벡터 축출 방법
Fig 5. Proccdure to derive the feature vector

그림 6은 상부는 융기되어 있고, 하부는 수평에 가까운

형태를 가진 지문 이미지이다.

그리고, 그림 7은 그림 7의 지문 이미지를 행별로 웨이블릿 변환 과정을 거친 1*128 개의 데이터이다. 즉, 각 행에 저주파 필터를 통과 시키면 지문의 상부는 융선과 계곡의 변화가 심하므로 데이터의 변화가 크지만, 하부는 그 변화가 상부에 비해서 크지 않다는 것을 알 수 있다.

그림 6 원본이미지
Fig. 6 Original Image

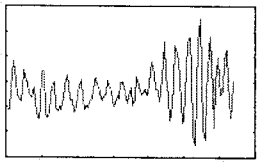

그림 7 웨이블릿 변환후
Fig. 7 Signal after one-stage Transform

그림 8는 여러 종류의 지문에 따른 특징 벡터가 다른 패턴으로 나타나는 것을 보여준다.

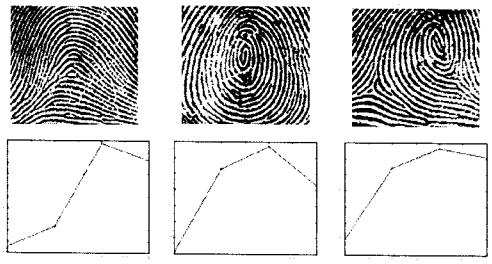

그림 8 지문의 특징 벡터
Fig 8. Feature vectors of fingerprint

5. 확률적 신경망

본 논문에서는 웨이블릿 변환을 통해 얻어진 특징벡터를 클라이언트에 저장한 후, 새롭게 입력된 지문 이미지의 데이터에 대해서 같은 방법으로 특징벡터를 구한 후 PNN을 이용하여 비교해 오차 범위내의 유사한 지문을 찾아내도록 하였다. 다음은 PNN을 구하는 식이다.

$$D_{ij}=\exp\left(\frac{1}{2\sigma^2}d_{ij}\right),\quad d_{ij}=\sum_{k=1}^{n}(x_i(k)-x_j(k))^2 \qquad (1)$$

n : size of feature vector x_i
d_{ij} : distance of the xi from x_j
σ : smoothing parameter
D_{ij} : output activation of PNN

그림 9은 5개의 서로 다른 지문의 특징 벡터를 구한 후 첫 번째 지문의 특징 벡터와의 거리를 계산하여 그래프로 그린 것이다.

그림 10은 그림 9의 벡터거리를 PNN에 입력한 결과 값을 나타낸 것이다. 첫 번째 것은 벡터거리가 0이므로 PNN 출력 값은 1이 되게 하고 정확도가 100% 라는 것을 의미한다. 다른 지문들과의 벡터거리는 그 값이 크므로 PNN 출력이 0이 됨을 알 수 있다. 식(1)의 σ의 값을 조절함으로 벡터거리의 값이 크더라도 PNN의 출력이 0이 되지 않게 조절할 수 있다. 이 값을 조절함으로 타인수락률(FAR), 본인거부율(FRR)을 조절할 수 있다.

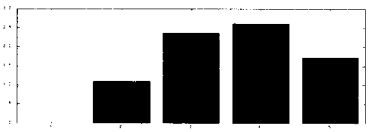

그림 9 특징벡터간의 거리
Fig 9. distance between feature vectors

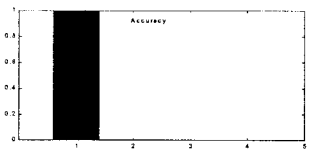

그림 10 확률적 신경망 출력값
Fig 10. Output of PNN

3. 결 론

앞의 실험 결과를 통해 실제 구현에 있어서 지문의 형태가 다르면 인식률이 높아짐을 알 수 있다. 하지만 입력된 지문이 목표지문에 비해 회전하거나 크게 이동하면, 변별력이 떨어짐을 알 수 있다. 이러한 단점을 극복하기 위해 실제 입력 장치에 손가락의 위치변화와 회전을 억제하는 가이드를 설치하는 것이 필요하다. 또한 유사한 타입의 지문이 입력되었을 경우에도 변별력이 떨어지는 경우가 있는데, 아주 유사한 모양의 궁상문 타입의 지문 입력이 그것이다. 이 경우는 PNN의 출력의 차이가 크지 않은 경우가 발생된다. 이러한 경우 DWT를 이용해 특징벡터를 추출할 때 한 단계를 줄여 특징벡터의 개수를 8개로 늘려줌으로 변별력을 높일 수 있지만 그에 따른 계산량의 증가는 불가피하다.

참 고 문 헌

[1] Henry C. LEE and R. E. Gaensslen, Advances in fingerprint Technology, CRC Press, 1994.

[2] C L. Wilson, C.I. Watson, and E. G. Paek, "Combined Optical and Neural Network Fingerprint Matching." SPIE Vol.3073, pp.373-382, 1997.

[3] Harold Szu, Charles Hsu, Joe Garcia, and Brian Telfer, "Fingerprint Data Acquisition, De-Smearing, Wavelet Feature Extraction & Identification," SPIE Vol.2491, pp.96-118, 1995.

[4] 이연조, "임베디드 리눅스 프로그래밍", PC BOOK, 2002.