논문 2013-08-17

실시간 응용을 위한 안드로이드 플랫폼에서의 안면 검출 시스템 구현

(Implementation of Face Detection System on Android Platform for Real-Time Applications)

한 병 길*, 임 길 택
(Byung-Gil Han, Kil-Taek Lim)

Abstract : This paper describes an implementation of face detection technology for a real-time application on the Android platform. Java class of Face-Detection for detection of human face is provided by the Android API. However, this function is not suitable to apply for the real-time applications due to inadequate detection speed and accuracy. In this paper, the AdaBoost based classification method which utilizes Local Binary Pattern (LBP) histogram is employed for face detection. The face detection module has been developed by C/C++ language for high-speed image processing, and this module is included to the Android platform using the Java Native Interface (JNI). The experiments were carried out in the Java-based environment and JNI-based environment. The experimental results have shown that the performance of JNI-based is faster than Java-based method and our system is well enough to apply for real-time applications.

Keywords : Face detection, Android platform, JNI, LBP, AdaBoost

Ⅰ. 서 론

생체 인식의 한 분야인 안면 인식은 사용자에게 직접적인 접촉 등의 능동적인 요구를 필요로 하지 않기 때문에 거부감 없이 자연스러운 인식을 통한 응용이 가능하다. 이런 장점으로 인해 안면 인식 기술은 보안, 엔터테인먼트, 광고 등 다양한 분야에서 그 활용이 증가하고 있다. 최근에는 스마트폰과 같은 모바일 기기의 보급이 대폭 증가함에 따라, 기존의 고정식 PC 기반에서의 안면 인식 기술이 모바일 기기로 전이되어 활용되고 있다[1-3].

모바일 기기에 안면 인식 기술을 적용함에 있어서는 프로세서, 메모리량, 네트워크 대역폭 등 PC와는 다른 환경의 제한 조건을 고려해야하나[2], 최근 모바일 기기의 발전이 급속히 이루어짐에 따라 이러한 제한 조건이 많이 완화되고 있는 실정이다. 하지만 낮은 사양의 모바일 기기에 안면 인식 기술을 적용하기 위해서는 여전히 안면 검출 속도와 정확도가 개선되어야할 필요성이 있다.

안면을 인식하기 위해서는 일반적으로 입력 영상에서 안면의 위치를 정확하게 검출하는 것이 선행되어야 한다. 안면을 검출하기 위한 기존의 연구들로는 피부색을 이용한 방법, 주성분분석(PCA)을 이용한 방법, 신경망(Neural Network)을 이용한 방법, SVM을 이용한 방법, AdaBoost를 이용한 방법 등이 있다[4-6]. 영상 내 어딘가에 미지의 크기로 존재하는 안면을 검출하기 위해서는 영상의 전영역을 다양한 크기의 탐색창으로 스캔하면서 안면의 존재 여부를 확인하는 매우 계산 집약적인 과정을 거친다. 일반적으로 이와 같은 과정은, 영상 피라미드를 생성한 후 피라미드 내의 각 영상에 대해서 고정된 크기의 탐색 창으로 적절한 스텝사이즈로 래스터 스캔하면서 탐색 창에 안면이 존재하는지 확인하는 과정을 거치는 것으로 구현된다. 영상의 크기에 따라 다르겠지만 많게는 수백만 번의 탐색이 이루어지므로 안면 검출의 고속화를 위해서는 각 탐색에서의 안면 확인을 고속화하는 방법 또는 탐색 횟수를 적게 하는 방법을 고려할 수 있을 것

*Corresponding Author (kilyhan@etri.re.kr)
Received: 15 Feb. 2013, Revised: 12 Mar. 2013,
Accepted: 02 Apr. 2013.
B.G. Han, K.T. Lim: ETRI

ISSN : 1975-5066

이다.

Viola와 Jones[6]는 안면 검출을 고속화하기 위해 AdaBoost 기반의 인식기들을 다단계의 캐스캐이드 형태로 적용하여 각 탐색에서의 안면 확인을 고속화하였다. 탐색 창 내의 영상은 대다수가 안면이 아니므로, 통계적으로 고속 처리하기위해서는 안면이 아닌 것을 빨리 기각하는 것이 올바른 전략이다. 처리 대상 영상의 분류 난이도에 따라 해당 인식기들을 캐스캐이드 방식으로 배치하여, 앞단에서 안면이 아닌 것으로 쉽게 판단되는 배경 영상을 빠르게 배제하고, 분류하기가 어려운 것은 뒷단의 보다 복잡한 인식기로 처리하였다. 안면과 배경에 대해 동일한 방식으로 처리하는 기존의 방식[NN방법, Rowely]에 비해 AdaBoost 기반 캐스캐이드 방식이 15배 이상 빠른 것으로 나타났으며[6], 이후 캐스캐이드 방식은 다양한 안면 고속 검출 연구 에 사용되었다[7, 8]. 그러나 이 방식은 학습 과정이 너무 복잡하고 매우 긴 시간동안 학습해야하는 단점이 있다. 캐스캐이드 단계의 수, 각 단계의 강분류기를 구성하는 약분류기의 수, 최소 검출률, 최대 부스팅 라운드 등의 파라메터를 설정하는 것은 전체 검출 성능에 중대한 영향을 끼치며 최적 값의 선택이 매우 어렵다. 앞 단계의 인식기의 임계값에 따라 다음 단계로 전달되는 탐색 창의 수가 결정되므로 탐색 정확도와 속도가 설정된 파라메터에 결정적으로 영향을 받게 된다.

동영상에서 안면을 검출할 경우에는 캐스캐이드 방식뿐만 아니라 다른 유용한 방법, 즉 배경 모델링[9], 슬라이싱[8], 트랙킹[8, 9] 등의 방법을 도입하여 단일 영상에서의 안면을 검출하는 것보다 훨씬 빠르게 처리할 수 있다. 배경 영역 모델링 기법을 도입하면, 고정 카메라 입력에서 검출의 대상이 되는 안면을 움직임 객체로 가정하고 고정된 배경 영역을 탐색 영역에서 제외함으로써 보다 더 빠르게 객체를 검출할 수 있다. 슬라이싱 기법은 연속되는 영상들의 각 피라미드를 모두 처리하지 않고 매 프레임 시각마다 피라미드 내의 영상 중 하나씩만을 처리하여 속도를 빠르게 하는 방법이다. 검출 대상 안면이 일단 검출되고 나면, 이 안면 영상에 대해 검출보다 훨씬 빠르게 동작하는 추적 기법을 적용함으로써 고속 처리가 가능해진다.

본 연구는 안드로이드 플랫폼의 모바일 기기에서 동작하는 단일 영상 안면 검출기의 구현에 관한 것으로, 고속으로 안면을 검출하기위해 다수의 배경 영상을 고속으로 기각함으로써 안면을 검출하는 캐스캐이드 방식의 안면 검출 방법에 대해 제안한

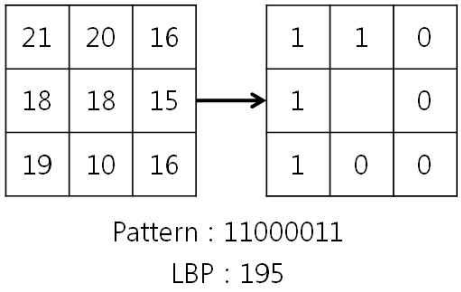

그림 1. LBP 계산
Fig. 1 Calculation of LBP

다. 안면 검출 알고리즘은 LBP 특징[10] 기반의 AdaBoost 방법을 사용하였다. AdaBoost 알고리즘으로 LBP 히스토그램 벡터 기반의 약분류기를 선택하여 강분류기를 구성하였다. 기존의 다단계 캐스캐이드 방법과 다르게 본 연구에서는 단일 단계의 캐스캐이드 방법을 적용한다.

일반적인 구현에 있어 안드로이드 플랫폼 응용은 자바를 기반으로 하지만 안면 검출과 같이 고속의 영상처리를 요구하는 응용은 자바 가상 머신의 한계로 인해 실시간 응용에 적합한 속도를 구현하기 어렵다. 따라서 본 연구에서는 고속의 영상처리를 위해 안면 검출 모듈 부분을 C/C++로 구현하고, 구현된 검출 모듈은 JNI (Java Native Interface)를 이용하여 안드로이드 플랫폼에 사용되었다.

본 논문의 2장에서는 안면 검출에 사용된 LBP 변환과 특징 후보 블록 추출, 그리고 AdaBoost 학습 알고리즘에 대해 설명하고 속도 향상을 위한 캐스캐이드 기법에 대해 논의한다. 3장에서는 제안한 안면 검출 알고리즘을 자바 언어 기반과 JNI 기반으로 구현하여 실제 스마트폰에서 안면 검출 속도를 비교한 결과에 대해 논의하고, 마지막으로 4장에서 결론 및 향후 과제에 대해 이야기 한다.

Ⅱ. 안면 검출 알고리즘

1. Local Binary Pattern(LBP)

Local Binary Pattern(LBP)은 영상의 질감을 표현하기 위해 디자인된 서술방법이다[10]. LBP 변환은 방법이 간단하고 변환에 필요한 계산량이 적을 뿐 아니라 흑백 영상에서 조명의 변화에 강한 특징을 가진다.

기본적인 LBP 변환 방법은 그림 1과 같이 3×3

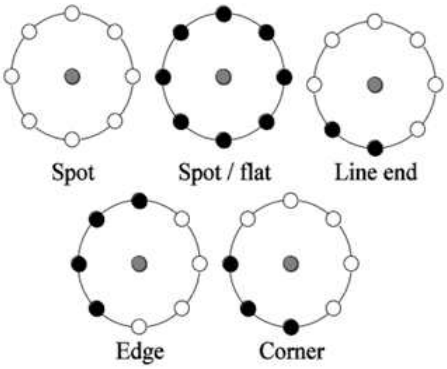

그림 2. Uniform LBP의 예
Fig. 2 Example of Uniform LBP

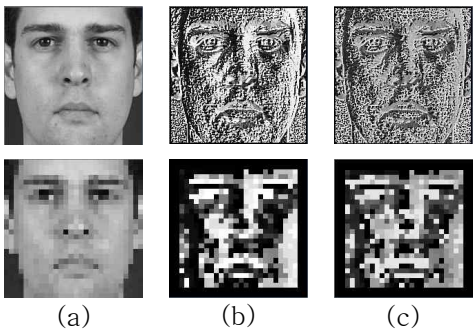

그림 3. LBP와 ULBP 변환 영상 (a)원본 영상 (b)LBP 영상 (c)ULBP 영상
Fig. 3. Image of Converted LBP and ULBP (a)Original Image (b)LBP Image (c)ULBP Image

블록에서 중심 화소를 기준으로 주변 8개 화소와의 비교를 통해 8개의 이진값이 계산되고, 계산된 이진값은 0~255 사이의 LBP 패턴값으로 표현된다. 3×3 블록뿐 아니라 5×5 이상의 블록에서 반지름 1 이상의 주변 8개 화소와 비교하는 확장된 버전의 LBP 변환도 가능하다. 또 다른 LBP의 확장으로 256개의 패턴 중, 0에서 1 혹은 1에서 0으로의 변화가 2번 이하인 58개의 패턴만을 의미있는 패턴값으로 보고 나머지 패턴은 59번째 패턴으로 하는 Uniform LBP(ULBP) 변환도 가능하다. ULBP는 그림 2와 같이 영상의 국부적 특징 중 점, 선, 면 등 관심 객체를 구분하는데 유용한 특징을 나타낸다. 그림 3은 200×200 크기 영상과 본 논문에서 학습 영상으로 사용된 28×28 크기 영상을 5×5 블록의 반경 2인 8화소 LBP와 ULBP 변환 영상을 보여준다.

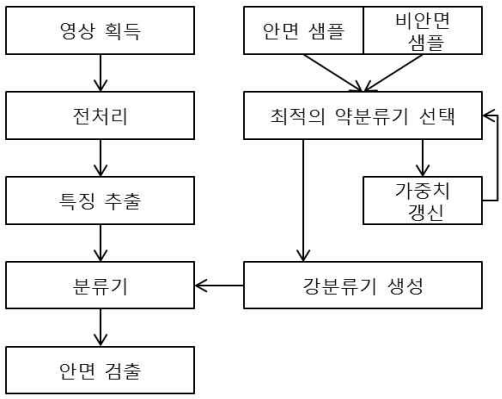

그림 4. 안면 검출 과정
Fig. 4 Process of face detection

2. 특징 후보 블록 추출

일반적으로 LBP 영상에서 특징을 추출하기 위해서는 W×H 크기의 원영상(탐색 창)을 N×M 개의 서브 블록으로 나누고 각각의 서브 블록에서 LBP 히스토그램을 계산하는 방법을 사용하는데, 본 논문에서 제안하는 방법은 1×1에서 W×H 크기의 가능한 모든 서브 블록에 대해 ULBP 히스토그램 벡터를 계산하여 서브 블록 분류기의 특징 벡터로 사용하였다. 예를 들어 28×28 크기의 템플릿 영상인 경우 ULBP 변환으로 인해 외곽의 2픽셀씩을 제외하고 총 90,000개의 서브 블록 및 ULBP 히스토그램 벡터가 추출된다.

3. AdaBoost 학습

AdaBoost 알고리즘은 Viola와 Jones에 의해 안면 검출에 성공적으로 적용된 부스팅 방법의 하나로써 다수의 약분류기를 결합하여 하나의 강분류기를 생성하는 방법이다[6]. 약분류기 는 식 1과 같이 나타내고, 강분류기 는 식 2와 같이 나타낸다. 여기서 x는 입력 영상을 의미하며, f는 서브 블록의 ULBP 특징벡터를 입력으로 하는 선형분류기, θ는 f의 임계값, p는 방향을 나타내기 위한 부호값이다. 즉 약분류기 h는 서브 블록의 ULBP 히스토그램 벡터기반 선형분류기 f, 임계값 θ와 방향 부호 p로 정의된다. T, t, α 등은 [6]의 표기법을 따른다.

$$h(x,f,p,\theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

$$C(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2}\sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

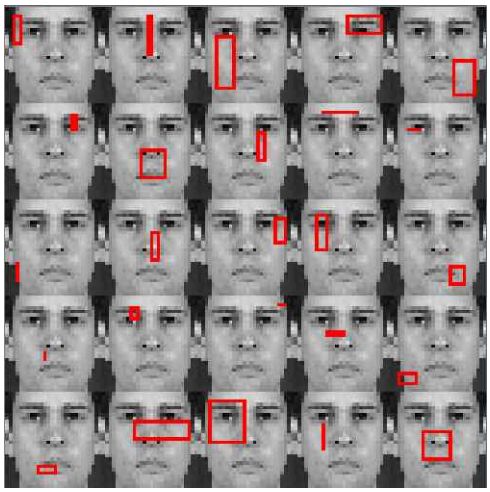

그림 5. 상위 25개의 특징 블록
Fig. 5 Top 25 Feature Blocks

그림 4는 AdaBoost를 이용한 학습을 통해 최적의 약분류기를 선택하고 선택된 약분류기를 결합한 강분류기를 이용하여 안면을 검출하는 과정을 보여준다. AdaBoost 학습을 위해 안면 영상과 비안면 영상을 학습 샘플로 하고 추출된 특징 후보 블록 중 학습 샘플을 가장 잘 분류하는 최적의 특징 서브 블록을 선택한다. 첫 번째 특징 블록이 선택되면 선택된 특징 블록의 분류 성능에 따라 특징 후보 블록의 가중치를 갱신하고 다음 특징 블록을 선택하는 작업을 반복한다. 반복적인 가중치의 갱신과 최적 특징 블록의 선택을 통해 필요한 수의 약분류기가 선택되면, 선택된 약분류기를 결합하여 하나의 강분류기를 생성한다. 그림 5는 90,000개의 특징 후보 블록 중 AdaBoost 학습을 통해 선택된 가장 성능이 좋은 상위 25개의 특징 블록을 보여준다.

4. 캐스캐이드를 이용한 안면 검출

생성된 강분류기를 이용하여 안면을 검출하기 위해서는 영상이 입력되면 흑백 변환, 리사이즈 등의 전처리를 수행한 후 강분류기와 같은 영역의 서브 블록에서 ULBP 히스토그램 특징 벡터을 추출한다. 서브 블록에서 추출된 특징값과 강분류기의 특징값을 비교하여 안면인지 여부를 판단한다.

템플릿 매칭 기반의 안면 검출의 경우 입력 영상에 존재하는 안면의 크기가 템플릿 영상의 크기와 일치해야 함으로 다양한 크기의 안면을 검출하기 위해서는 입력 영상의 크기를 변환하면서 안면을 검출하는 피라미드 방법을 사용한다. 피라미드

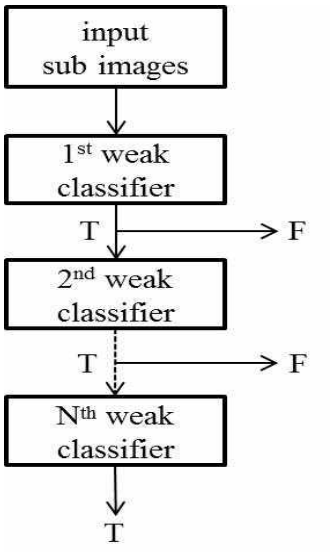

그림 6. 캐스캐이드 기반 안면 검출
Fig. 6 Cascade based Face Detection

방법을 사용할 경우 피라미드 레벨에 따라 스캔해야 하는 양이 증가하게 된다. 또한 모든 서브 영상에 대해 템플릿 매칭을 수행하면서 학습한 약분류기 모두를 사용할 경우, 검출 성능은 우수하지만 계산량이 증가하여 안면 검출 속도가 떨어지는 단점이 있다. 따라서 검출률이 약간 떨어지더라도 검출 속도를 향상시키기 위해 그림 6과 같이 캐스캐이드 방법을 이용한다. 기존의 방법에서는 캐스캐이드의 각 단계마다 강분류기를 사용하는 다단계 기법을 적용하나, 본 연구에서는 단일 단계 캐스캐이드로 하나의 강분류기를 구성하는 개별 약분류기를 차례로 적용한다. 템플릿 매칭을 위해 피라미드 영상을 스캔하면서 상위의 약분류기를 통과한 서브 영상은 다음 약분류기를 통해 계속 비교하면서 매칭율이 경계값 밑으로 떨어지면 탈락시킨다.

Ⅲ. 구현 및 성능 비교

1. JNI를 이용한 안면 검출 구현

스마트폰 운영체제 중 하나인 안드로이드 플랫폼은 자바를 기반으로 하기 때문에 자바 플랫폼의 속성을 그대로 가지고 있다. 따라서 영상 처리와 같은 고속의 성능이 필요한 부분에서는 성능이 떨어지는 단점이 있다. 안면 검출 알고리즘의 경우 입력 영상에서 안면의 위치와 크기를 알 수 없기 때문에

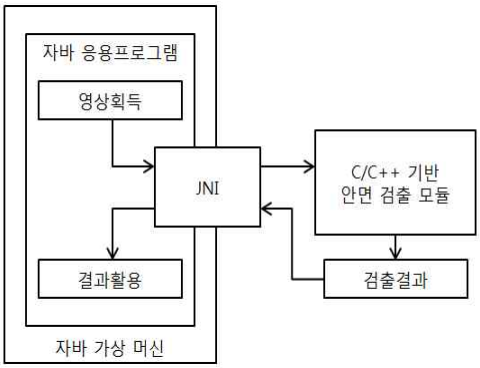

그림 7. JNI를 이용한 C/C++기반 안면 검출 모듈 사용
Fig. 7 Using C/C++ based on face detection module by JNI

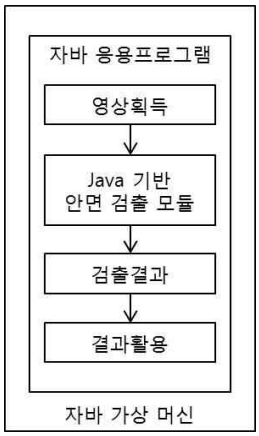

그림 8. 자바 기반 안면 검출 모듈 사용
Fig. 8 Using Java based on face detection module

피라미드 방법을 사용하여 영상의 크기를 변화시키면서, 또한 영상 전체를 순차 스캔하여야 한다. 따라서 고속의 처리 속도가 필요하고, 이런 문제점을 해결하기 위해 안면 검출 모듈은 C/C++을 이용하여 구현되었다.

자바 가상 머신에서는 C/C++로 구현된 네이티브 코드를 사용하기 위해 Java Native Interface (JNI)를 제공한다. JNI는 자바와 C/C++간에 데이터의 호환을 위한 인터페이스를 제공한다. 그림 7은 안드로이드 플랫폼에서 JNI를 이용하여 C/C++모듈을 사용하는 구조를 보여준다. 안드로이드 플랫폼에서 카메라를 통해 영상을 획득하고 획득된 영상은 JNI를 통하여 안면 검출 모듈이 인식 가능한 데이터로 변환하여 넘겨준다. 검출 모듈에서는 안면을 검출한 결과를 다시 JNI를 통하여 안드로이드 플랫폼에서 인식 가능한 데이터로 변환하여 넘겨준다.

2. Java 기반 구현

본 논문에서 제안한 JNI 기반의 안면 검출 모듈 구현이 실제로 향상된 검출 속도를 보여주는지 확인하기 위해 같은 안면 검출 알고리즘을 자바 기반으로 구현하였다. 그림 8과 같이 영상의 획득 및 안면 검출, 결과의 활용까지 모든 작업이 자바 가상 머신 기반에서 이루어지게 된다.

3. 구현 및 검출 시간 비교 결과

본 논문에서는 안면 검출기의 구현 실험을 XM2VTS 데이터셋을 사용하였다. 원시 영상을 320×240의 영상으로 변환하여 검출 실험에 사용하였으며, 영상 피라미드 구성에서의 스케일은 1/1.2, 탐색 스텝은 2 픽셀로 설정하였다. 테스트 샘플의 전체 개수는 1,104개였다. 탐색 창의 크기는 28×28이며, 실험에 사용한 LBP 반경은 1과 2였다.

표 1은 PC기반에서 AdaBoost 학습을 통해 약분류기 100개를 생성한 후, 캐스캐이드 방법을 사용한 경우와 100개의 약분류기로 이루어진 하나의 강분류기를 사용한 경우의 검출 성능 및 시간을 나타낸다. 캐스캐이드 방법은 전체적인 검출률이 약 2%정도 낮지만 검출 속도는 약 30배정도 빠르게 나타났다. 캐스캐이드방법의 경우 대부분의 영상이 초반 몇 개의 약분류기를 통해 탈락되므로, 100개의 약분류기 모두를 사용하는 하나의 강분류기에 비해 상당한 속도의 향상을 달성할 수 있다. 속도가 중요한 기준인 실시간 응용에서는 다소 검출률이 저하되더라도 수십 배 빠른 캐스캐이드 방식이 보다 더 유용할 것으로 판단된다. LBP 반경에 따른 검출 성능을 살펴보면, 반경이 1인(3×3) LBP 히스토그램 벡터의 경우가 반경이 2인(5×5) 경우보다 검출 성능이 더 좋은 것으로 나타났다.

안면 검출 시스템은 안드로이드 버전 4.0.4 기반의 단말기에서 구현되었다. 실시간 입력 동영상은 안드로이드 API를 이용하여 단말기의 카메라를 통해 획득하였고, 전처리 및 영상 포맷 변환 등 기본적인 영상의 전처리를 위해 OpenCV 라이브러리를 사용하였다[11]. 그림 9는 실제 구현된 단말기에서 정지영상을 입력으로 하여 안면 검출 모듈을 테스트한 화면이다. 표 2는 JNI를 이용한 안면 검출 모

표 1. 검출률 및 검출 속도
Table 1. Detection Rate and Detection Time

Cascade	ULBP Radius	Detect (%)	False Detect (%)	Time (ms)
Use	1	97.10	2.17	21.34
	2	96.01	3.44	42.75
Not Use	1	99.00	1.72	610.69
	2	98.28	1.81	623.51

표 2. JNI와 자바 기반에서의 검출 속도 비교
Table 2. Compare detection time between JNI and Java based

JNI based	Java based
72.2ms	2040.81ms

듈과 자바 기반에서 구현한 안면 검출 모듈의 검출 시간을 비교한 결과이다. 비교 결과에서 보듯이 자바 가상 머신 기반에서는 그 한계로 인하여 다량의 고속 연산이 필요한 응용에는 적합하지 않음을 알 수 있다.

Ⅳ. 결론 및 향후과제

본 논문에서는 안드로이드 플랫폼 기반의 스마트 모바일기기에서 실시간 응용을 위한 단일 영상에서의 안면 검출 기술의 구현에 대해 논의하였다. ULBP 히스토그램 벡터를 이용해 특징 후보 블록을 추출한 후 AdaBoost 학습을 통해 100개의 약분류기를 선택하였다. 캐스캐이드 방법은, 캐스캐이드 방법을 사용하지 않는 것에 비해 검출률이 약간 저하되었지만 검출 속도는 수십 배정도 빨라져서 실시간 응용에 보다 유리하다. 안면 검출 모듈은 고속 연산을 위해 C/C++을 이용하여 구현하였으며 자바 기반의 안드로이드 플랫폼의 한계를 극복하기 위해 구현된 안면 검출 모듈은 JNI를 이용하여 자바 가상 머신 기반의 안드로이드 응용프로그램과 연결하여 최종 응용프로그램을 구현하였다. 속도 비교 결과 JNI 기반의 안면 검출 모듈이 자바 기반의 모듈에 비해 20~30배 빠른 속도를 보여주는 것을 확인하였다. 이는 동영상에서 12~15 fps 정도의 속도를 낼 수 있으므로, 제안하는 방법이 실시간 응용에 충분히 활용 가능하다는 것을 의미한다.

본 연구는 단일 영상에서의 안면 고속 검출 방법에 관한 것이지만 후속 연구로서 동영상에 대해

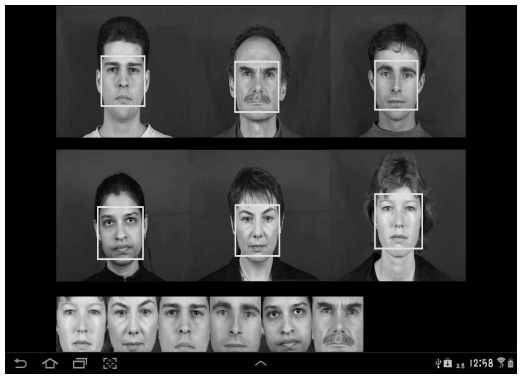

그림 9. 구현 결과 영상
Fig. 9 Image of implementation result

제안하는 방법을 적용해보고자 한다. 보다 고속의 동영상 안면 검출 시스템을 구현하기 위해서는 배경영역 모델링, 슬라이싱, 추적 기법 등 동영상의 특성을 활용하는 유용한 다른 기법과의 결합이 효율적이다. 동영상의 종류와 해상도에 따라 이들 방법과 제안한 방법을 성공적으로 결합하는 방식에 대한 연구를 향후의 연구과제로 하고자 한다. 또한 제안한 안면 검출 기법에 안면 검증 및 식별 기법을 추가하여 완전한 모바일 단말 기반 안면 인식 시스템을 개발하고자한다.

References

[1] J. Ren, X. Jiang, J. Yuan, “A complete and fully automated face verification system on mobile devices,” Pattern Recognition, Vol. 46, No. 1, pp.45-56, 2013.

[2] A. Pabbaraju, S. Puchakayala, “Face Recognition in Mobile Devices,” Electrical Engineering and Computer Science, University of Michigan, 2010.

[3] K. Lu, L. Dong, “Using LBP histogram for face recognition on Android platform,” Proceedings on International Conference of Computer Research and Development, Vol. 1, pp.266-268, 2011.

[4] M. Turk, A. Pentland, “Face Recognition Using Eigenfaces,” Proceedings on IEEE Conference of Computer Vision and Pattern Recognition, pp.586-591, 1991.

[5] H. Rowley, S. Baluja, T. Kanade, “Neural

Network-Based Face Detection," Proceedings on IEEE Conference of Pattern Recognition and Machine Intelligence, pp.1-27, 1998.

[6] P. Viola, M. Jones, "Robust Real-Time Face Detection", International Journal of Computer Vision, Vol. 57, No. 2, pp.137-154. 2004.

[7] D. Le, S. Satoh, "A multi-stage approach to fast face detection," IEICE Trans. Inf. & Syst. Vol. E89-D, No. 7, pp.2275-2285, 2006.

[8] C. Kublebeck, A. Ernst, "Face detection and tracking in video sequences using the modified census transformation," Image and Vision Computing Vol. 24, No. 6, pp. 564-572, 2006.

[9] R. Feris, Y. Tian, A. Hampapur, "Capturing People in Surveillance Video," Proceedings on IEEE Conference of Computer Vision and Pattern Recognition, pp.1-8, 2007.

[10] T. Ahonen, A. Hadid, M. Pietikainen, "Face Description with Local Binary Patterns: Application to Face Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28, pp.2037-2041, 2006.

[11] OpenCV Library, http://opencv.org

저 자 소 개

한 병 길

2005년 경북대학교 전자전기공학부 공학사.
2007년 경북대학교 전자공학과 공학석사.
현재, 한국전자통신연구원 대경권연구센터 연구원.
관심분야: 영상처리, 로봇비전, 비디오보안
Email: kilyhan@etri.re.kr

임 길 택

1993년 경북대학교 전자공학 공학사.
1995년 경북대학교 전자공학 공학석사.
1999년 경북대학교 전자공학 공학박사.
현재, 한국전자통신연구원 대경권연구센터 책임연구원.
관심분야: 컴퓨터비전, 패턴인식, 비디오보안
Email: ktl@etri.re.kr