# AdaBoost 기반의 실시간 고속 얼굴검출 및 추적시스템의 개발

# AdaBoost-based Real-Time Face Detection & Tracking System

김 정 현, 노 태 정*, 김 진 영, 홍 영 진, 권 장 우, 강 동 중
(Jeong-Hyun Kim, Tae-Jung Lho, Jin-Young Kim, Young-Jin Hong, Jang-Woo Kwon, and Dong-Joong Kang)

**Abstract** : This paper presents a method for real-time face detection and tracking which combined Adaboost and Camshift algorithm. Adaboost algorithm is a method which selects an important feature called weak classifier among many possible image features by tuning weight of each feature from learning candidates. Even though excellent performance extracting the object, computing time of the algorithm is very high with window size of multi-scale to search image region. So direct application of the method is not easy for real-time tasks such as multi-task OS, robot, and mobile environment. But CAMshift method is an improvement of Mean-shift algorithm for the video streaming environment and track the interesting object at high speed based on hue value of the target region. The detection efficiency of the method is not good for environment of dynamic illumination. We propose a combined method of Adaboost and CAMshift to improve the computing speed with good face detection performance. The method was proved for real image sequences including single and more faces.

**Keywords** : face detection, AdaBoost, CAMShift, real-time processing, computer vision

## I. 서론

얼굴은 사람의 인식을 위한 중요한 정보로서 최근 들어 컴퓨터 비전에서는 얼굴인식, 표정인식 등의 연구가 활발하게 진행되고 있다. 특히, 얼굴 검출은 입력 영상이 얼굴인지 아닌지를 구별하는 이진분류의 문제로 인식에 앞서 선행되어야 할 작업이다. 얼굴 검출 및 추적 시스템은 영상감시, 원격회의, 얼굴인식 시스템, 그리고 최근 많은 관심을 불러 모으고 있는 지능형 홈이나 인간형 로봇에 활용되는 등 다양한 분야에 응용될 수 있는 기술이다. 하지만 얼굴 검출은 얼굴의 움직임, 크기 변화, 얼굴 표정, 인종, 나이, 성별, 헤어스타일, 타 객체와의 겹침, 조명, 카메라의 기계적인 특성 등에 따라 다양하게 나타날 수 있기 때문에, 얼굴 검출 기술에 대한 상당한 연구가 진행되었지만 아직도 많은 문제점을 가지고 있다. 기존의 얼굴 검출 방법은 지식기반(knowledge-based) 방법[1], 특징 기반(feature-based) 방법[2], 템플릿 정합(template matching) 방법[3], 외형 기반(appearance-base) 방법[4] 등이 있다. 첫 번째 지식기반 방법은 얼굴에 대한 인간의 일반적인 지식을 기반으로 하는 방법으로 얼굴을 구성하는 눈, 코, 입 간의 위치, 거리 등이 이에 해당한다. 두 번째로 특징기반 방법은 자세나 조명 등의 변화에서도 얼굴 검출에 용이한 구조적 특징을 찾는 방법이다. 세 번째 템플릿 정합 방법은 몇몇 얼굴의 기본 형태를 통해 일정 수준 이상의 상관관계를 가질 경우 얼굴로 판단하는 방법이다. 네 번째 유형은 최근 가장 널리 연구되고 있는 외형기반 방법으로 나머지 세 가지 유형들과 달리 학습을 통해 얼굴을 모델링 한다는 점에서 큰 차이를 가지는데 최근 강력하고 우수한 알고리즘들이 많이 소개 되고 있다. 대표적인 방법으로 신경회로망(Neural Network: NN)[5], SVM(Support Vector Machine: SVM)[6], AdaBoost (Adaptive Boosting)[7] 등이 있다.

최근의 얼굴 추적의 방법으로 Wang과 Brandstein[8]은 머리의 타원경계와 2계 도함수가 목 부분에서 크다는 가정으로 얼굴을 추적한다. 그러나 카메라 운동 등 배경에 움직임에 의하여 추적이 실패할 가능성이 크다. Hager와 Belhumeur[9]는 베이시스(basis) 이미지(image)들을 이용해서 여러 조명 상황을 모델링하여 상관계수법에 의한 추적 알고리즘을 제안하였다. 그러나 추적을 위한 초기 위치를 설정하여야 하며 베이시스 이미지는 얼굴 방향과 가려짐에 대하여 민감하다. 또한 McKenna와 Gong[10]은 움직임이 있는 영역을 검출하여 정면 얼굴을 찾아내는 방법을 제안하였다. 하지만 카메라의 움직임에 의해 배경영역의 움직임과 얼굴이 부분적으로 가려지는 것에 영향을 많이 받는다.

본 논문은 얼굴검출 방법으로서 Viola와 Jones[11]가 제안한 방법을 이용한다. Viola와 Jones는 AdaBoost 알고리즘을 실시간에서 적용할 수 있도록 인티그럴 영상(integral image)과 cascade 특징 집합을 도입하여 복잡한 연산을 줄이고 동시에 높은 검출 성능을 보이는 방법을 제안하였다. Viola와 Jones는 200개의 가장 좋은 이미지 특징을 사용하여 95% 검출율을 보였고 700Mhz *Pentium-III* processor 환경에서 384x288 크기의 이미지를 0.067초(초당 약 14 frame)의 속도로 검출하였다. 하지만 로봇, 모바일 기기와 같은 임베디드(embedded) 시스템과 다중작업(multi-task)을 수행하는 실시간 시스템에서는 여전히 만족할 만한 충분한 속도가 아니다. Viola와 Jones의 얼굴검출 방법은 매우 뛰어나지만 매 프레임(frame) 마다 다양한 크기의 얼굴을 찾기 위해서

* 책임저자(Corresponding Author)
논문접수 : 2007. 4. 26., 채택확정 : 2007. 9. 14.
김정현, 강동중 : 부산대학교 기계공학부
(feelmare@pusan.ac.kr/djkang@pusan.ac.kr)
노태정, 김진영 : 동명대학교 메카트로닉스공학과
(tjlho@tu.ac.kr/kjy97@tu.ac.kr)
홍영진 : 동명대학교 전기전자공학과(gryjhong@tu.ac.kr)
권장우 : 동명대학교 컴퓨터공학과(jwkwon@tu.ac.kr)
※ 본 논문은 정보통신부 u-Port ITRC 연구비 및 SK Telecom 연구비 지원에 의하여 연구하였음.

멀티스케일(multi-scale)의 윈도우를 이용하여 이미지 상의 모든 픽셀(pixel)을 이동(shift) 하면서 스캔(scan)을 수행하는데 이는 많은 연산을 필요로 한다. 예를 들어 320x240의 이미지에서 검색 윈도우를 20x20에서 40x40의 크기로 증가시키면서 모든 위치를 검색한다면 총 25,849,530회의 검사를 수행하므로 실시간 처리에 상당한 속도 저하를 일으킨다.

Bradski는 CAMShift법을 제안하였는데 이 방법의 얼굴 추적 알고리즘은 색상을 기반으로 고속의 객체 추적을 수행한다[12]. CAMShift 알고리즘은 MeanShift[13] 알고리즘을 스트리밍(streaming) 환경에서 사용할 수 있도록 개선한 것으로 검출된 객체 영역의 색상(hue) 값의 분포를 이용하여 변화될 위치를 예측하고 탐지한 후 중심을 찾아 객체를 추적하게 된다. 그러나 객체의 색 정보에 의존하는 추적방법을 사용하므로 조도변화, 잡음이 많은 배경에서는 성능이 좋지 못하다.

본 논문은 AdaBoost 알고리즘을 이용하여 빠르게 얼굴을 검출하고 검출된 영역에 대해서 CAMShift 알고리즘을 이용하여 다음 프레임에서 얼굴 영역을 예측하여 작은 검색 영역을 제시함으로서 전체를 스캔하는 기존 방법을 효율적으로 개선하였다. 작은 검색 영역에 대해서 AdaBoost 알고리즘은 얼굴을 검출하고 얼굴이 있을 경우는 CAMShift 알고리즘으로 추적을 계속하고 얼굴이 아닌 경우는 다시 검색 영역을 전체로 하여 얼굴을 찾는다. 본 시스템은 *Window-XP* 의 *Intel Core(TM)2* CPU 6400 2.13GHz의 PC 환경에서 1 프레임당 0.016초의 처리 속도(초당 62.5 frames)로 우수한 성능을 보였다.

## II. 실시간 얼굴 검출 및 추적 시스템

### 1. AdaBoost를 이용한 얼굴 검출

#### 1.1 특징 모델

얼굴 검출을 위한 특징으로 Papageoriou *et al.*[14]에 의해 제안된 간단하면서도 연산이 빠른 Harr-like 특징(그림 1)을 객체검출을 위한 특징 집합으로 사용하였다. Harr-like 특징은 인티그럴 영상을 이용하여 빠르게 연산할 수 있다. 이러한 Harr-like 특징과 인티그럴 영상의 사용은 복잡한 연산을 줄이고, 높은 검출 성능이 요구되는 실시간 객체 검출에 효과적으로 적용된다. Harr-like 특징 계수를 이용해서 얼굴영역을 검출하면 유동적인 얼굴의 특성에도 강인한 검출 성능을 보인다. 그림 1은 Harr-like 특징의 대표적인 4종류를 보인다. 그림 1(a)와 (b)는 2개의 사각형 모양으로서, 특징 값은 밝은 부분의 모든 픽셀 값의 합과 어두운 부분의 모든 픽셀 값의 합을 구하고 두 합의 차로서 특징 값을 구한다. 그림 1(c)는 바깥쪽의 밝은 부분의 모든 픽셀 값의 합과 가운데 어두운 부분의 모든 픽셀 값에 합의 차로서 구하며, 그림 1(d)는 대각의 방향으로 위치한 각 밝은 부분과 어두운 부분의 모든 픽셀 값의 합의 차로서 구한다. 학습 영상의 크기는 24x24이며 Harr-like 특징의 최소 크기는 8x8이다. Harr-like 특징은 다양한 크기로 구성할 수 있는데, 8x8의 크기에서 가로, 세로 크기를 1씩 증가시키면서 24x24 영상 크기내의 모든 곳을 검사하는 특징을 만든다면 (1)에 의하여 총 7,140개의 특징 집합으로 구성할 수 있다.

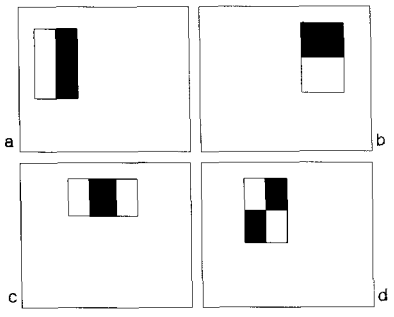



그림 1. Harr-like 특징 집합. (a)와 (b)는 2개의 사각형으로 구성되며, (c)는 3개의 사각형, (d)는 4개의 사각형으로 구성된다. 각 특징의 값은 검은색 사각형 내의 픽셀 합과 흰색 사각형 내의 픽셀 합의 차로서 특징 값을 구함.

Fig. 1. A set of Harr-like features. (a) and (b) consist of two rectangles, (c) consist of three rectangles and (d) consist of four rectangles. Their feature values can be derived by solving the sums of all pixel values in the white and dark region, respectively, and finding the remainder of these two values.

$$f=(\sum_{i=8}^{24}(24-i+1)^2)\times 4 \tag{1}$$

#### 1.2 인티그럴 영상

특징 값 계산이 대부분 반복적으로 일어나기 때문에 입력 영상을 그림 2의 인티그럴 영상으로 변환한다.

(2)에서 $ii(x,y)$는 인티그럴 영상이고 $i(x,y)$는 원영상의 밝기 값이다. 인티그럴 영상에서 점 $(x, y)$의 값은 (2)와 같이 가로 방향으로 $x$좌표까지, 세로 방향으로 $y$좌표까지의 픽셀 값을 모두 누적시키고 나면 구할 수 있다. 입력된 영상에 대하여 인티그럴 영상을 만들어 놓으면 특징 값을 계산하기 위해서 사각형 내의 모든 픽셀 값을 매번 더해야 하는 과정을 줄일 수 있다.

그림 2에서 점 $(x, y)$에서의 값은 (2)로 구할 수 있다.

$$ii(x,y)=\sum_{x'\le x,y'\le y} i(x'y') \tag{2}$$

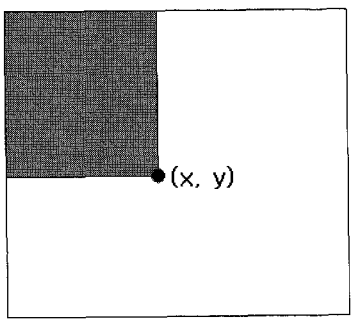



그림 2. 인티그럴 영상에서 점 $(x, y)$의 값은 어두운 색의 사각형 내의 모든 픽셀 밝기의 합.

Fig. 2. The value of point $(x,y)$ in the integral image is sum of pixel values in a dark rectangle.

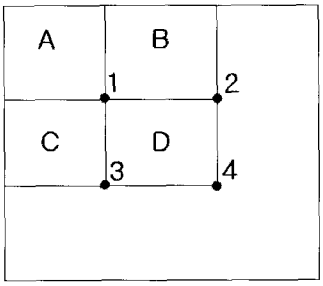



그림 3. 인티그럴 영상에서 D영역의 픽셀 값의 합은 점4 + 점1 -(점2+점3) 으로 쉽게 구함.

Fig. 3. In the integral image, the sum of pixel value in area D can be easily found by the calculation of point 4 + point 1 - (point 2+ point 3).

그림 3의 인티그럴 영상에서 $D$ 영역의 픽셀 값의 합을 구한다면 점4 + 점1 - (점2 + 점3)의 계산으로 쉽게 값을 얻을 수 있다.

1.3 학습 방법

AdaBoost 학습 알고리즘의 기본 개념은 약한 분류기(weak classifier)를 선형적으로 결합하여 최종적으로 높은 검출 성능을 가진 강한 분류기(strong classifier)를 생성하는 것이다[15]. AdaBoost 알고리즘은 반복적인 계산에 의해서 학습을 수행하며 특징 선택과 분류기 학습의 2가지 작업을 수행한다. 매 반복마다 간단한 학습 알고리즘의 분류 능력이 강해지는데 이는 최종적으로 약한 분류기의 결합에 의해서 강한 분류기를 만들어 내기 때문이다. 약한 분류기라고 부르는 이유는 1개의 분류기로 가장 최선의 분류를 기대하기 힘들기 때문이다. 학습의 첫 번째 단계 후에는 이전의 약한 분류기가 잘못 분류한 것을 강조하기 위하여 웨이트(weight)를 재조정한다. 학습의 마지막으로 만들어지는 강한 분류기는 2단계 레이어(layer)의 퍼셉트론[16]의 형태가 된다. AdaBoost 알고리즘의 학습은 Freund와 Schapire에 의해서 반복의 횟수에 따라 강한 분류기의 에러가 0에 지수적으로 접근됨을 증명하였다[17]. 약한 분류기는 얼굴과 배경 영상을 가장 잘 나누는 하나의 특징을 선택하는 역할을 한다. 약한 분류기는 (3)과 같이 정의 된다.

$$h_j(x)=\begin{cases}1 \text{ if } f_j(x)<\theta_j \\ -1 \quad otherwise\end{cases} \tag{3}$$

(3)에서 $x$는 24x24 크기의 이미지에서의 서브 윈도우(sub window)이며 $f_j(x)$는 $x$에 대한 $j$번째 특징값을 나타내는 함수이다. $\theta_j$는 임계값(threshold value) 이다. 일반적으로 임계값은 훈련영상 집합에 대한 Harr-like 특징값의 평균으로 결정한다.

강한 분류기는 약한 분류기의 선형적 결합 형태로 여러 개의 특징을 결합해서 실질적으로 얼굴의 패턴을 구별하는 역할을 한다. 표 1은 AdaBoost 전체 알고리즘이다.

약한 분류기는 표 1과 같이 가중치($w_t$)를 이용하여 학습을 한다. 입력 받은 훈련 영상을 잘못 분류하면 $w_t(i)$를 증가시키고, 옳게 분류하면 $w_t(i)$를 감소시킨다. 이 때 가장 낮은 에러 율을 가지는 Harr-like 특징을 선택한다. 하나의 특징이 선택되면 다시 훈련 영상을 입력으로 받아들이고 이와 같은 과정을 $T$회 반복하여 가중치를 변환하고 에러율이 가장 낮은 특징을 선택한다. 표 1(a)에서는 변화된 가중치에 대하여 웨이트를 (4)로 정규화 한다.

표 1. AdaBoost 학습 알고리즘.

Table 1. The learning algorithm of AdaBoost.

1. Given N examples $(x_1,y_1),...,(x_N,y_N)$ with $x\in R^k$ , $y_i\in$ {-1, 1}
2. Initialize weight $w_1(i)=\frac{1}{N}$, $i=1,...,N$
3. Repeat for t = 1,...,T
(a) Normalize the weights
(b) Get weak hypothesis $h_t=X\rightarrow$ {-1, +1} with error $\epsilon_j=\sum_i w_i(|h_j(x_i)-y_i|)$
(c) Choose the classifier, $h_t$, with the lowest error $\varepsilon_t$
(d) weight update :
$$w_{t+1,i}=w_{t,i}\beta_t^{1-e_i}$$
where $e_i=0$ if example $x_i$ is classified correctly,
$e_i=1$ otherwise, and $\beta_t=\frac{\epsilon_t}{1-\epsilon_t}$
4. The final strong classifier is:
$$h(x)=\begin{cases}1, \sum_{t=1}^{T}\alpha_t h_t(x)\geq \frac{1}{2}\sum_{t=1}^{T}\alpha_t \\ -1, otherwise\end{cases}$$
where $\alpha_t=\log\frac{1}{\beta_t}$

$$w_{t,i}=\frac{w_{t,i}}{\sum_{j=1}^{n}w_{t,j}} \tag{4}$$

표 1(b)에서는 $j$번째 약한 분류기로 $i$개의 학습 데이터에 대한 분류 결과를 현재 웨이트에 곱하여 그 합을 구한다. 이것은 $j$번째 약한 분류기에 대한 에러 값이 된다. 표 1(c)는 $j$개의 에러 값 중에서 가장 작은 값을 선택하여 $t$번째 반복에서 최적의 약한 분류기 $h_t$를 선택하게 된다. 즉, 얼굴과 배경을 가장 잘 분리하는 하나의 특징이 최적 특징으로 선택된다. 이 최적특징이 하나의 약한 분류기가 된다. 표 1(d)는 웨이트를 재조정한다. 어떤 학습후보 $x_i$가 최적으로 선택된 특징 $h_t$에 의해 잘못 분류된 결과를 주면 $w_i$는 변화 없이 유지되고 정확하게 분류되면 $w_i$는 감소하게 된다. 결국 이러한 결과는 다음 반복에서 $\epsilon_j$의 계산에 있어 전단계의 $h_t$에 의해 잘못 분류 되었던 학습후보를 더 강조하는 결과를 가져온다. 따라서 다음 단계의 최적 분류기는 앞 단계의 $h_t$에 의해 잘 분류되지 않은 후보들을 잘 분리하는 또 다른 특징을 선택하게 만든다. 그 결과 약한 분류

기를 단계적으로 생성하게 되고 이것들의 선형적인 결합으로 강한 분류기를 생성한다.

2. CAMShift을 이용한 얼굴 추적

CAMShift(Continuous Adaptive Mean Shift) 알고리즘은 MeanShift의 정지영상-색 분할 알고리즘을 비디오 영상으로 확장한 알고리즘이다. MeanShift는 초기의 검색 영역의 크기와 위치를 지정하여, 반복되는 색 분할 계산에 의해서 색상 클러스터가 발생되고 초기 지정한 색 영역에 기반 하여 경계를 결정하여 관심 물체를 추출할 수 있다. 그러나 MeanShift 알고리즘은 그 복잡한 계산량에 의해서 실시간처리에서는 부적합하다. 이 와 같은 문제를 해결하기 위해서 CAMShift 방법이 사용된다.

2.1 CAMShift 알고리즘 구성

CAMShift 알고리즘의 전체 구성은 그림 4와 같다.

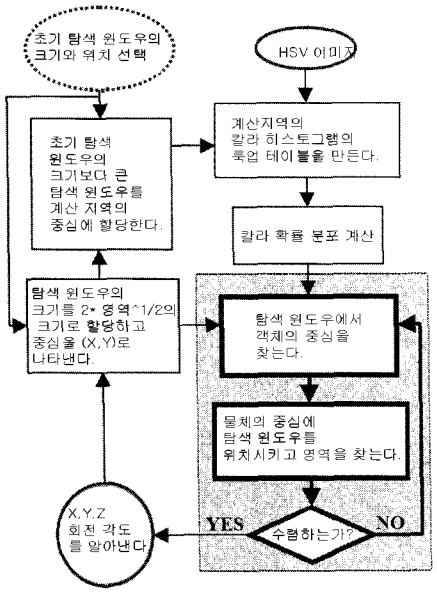



그림 4. CAMShift 알고리즘.

Fig. 4. CAMShift algorithm.

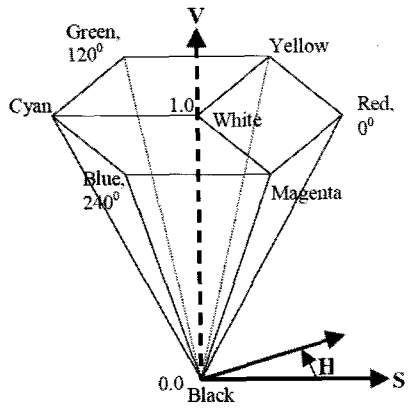



그림 5. HSV 색 모델(H: 색상(Hue), S: 채도(Saturation), V: 명도(Value/Intensity)).

Fig. 5. HSV Color model(The HSV represents Hue, Saturation, and Value(Intensity)).

최초 관심 객체의 영역이 주어지면 이 영역의 칼라 값은 그림 5의 HSV 색모델 hue값으로 변환된다. hue값을 사용하면 조명의 영향에 덜 민감하며 hue값에서 아주 작은 값(어두움)이나 큰 값(밝음)은 알아볼 수 없으므로 배제 할 수 있다. 이를 바탕으로 얼굴 영역을 초기 설정하여 고유색을 나타내는 hue값으로 0~255 사이의 값으로 정규화하여 1차원 히스토그램(histogram)을 생성한다. 피부영역 검출을 위한 룩업테이블(Look-Up Table: LUT)은 히스토그램을 기반으로 (5)를 통해 얻어진 확률 값으로 구성한다. 여기서 $h$는 0~255 사이의 인덱스(index)를 표시한다. $h$값이 피부색에 가깝다면 높은 확률 값을 가질 것이고 피부색에서 멀어지면 LUT의 확률 값은 낮아질 것이다.

$$L(h) = \frac{H(h)}{\sum_{i=0}^{255} h(i)} \tag{5}$$

일단 LUT가 만들어지고 나면 빠른 속도로 피부색 영역만을 검출하는 것이 가능하다. 비디오의 영상이 입력되면 각 영상 내의 개개의 픽셀에서 hue값이 추출되고 이 값은 LUT의 인덱스로 사용되어 직접적으로 확률 값을 나타내게 된다. 만일 h값이 피부색에 가깝다면 LUT 내의 높은 확률 값이 반환될 것이고 배경영역은 낮은 확률 값이 반환된다. 따라서 추가적인 연산 없이 LUT만을 사용하여 피부색의 확률 값으로 구성된 확률분포영상(probability distribution image)을 고속으로 얻는 것이 가능하다. LUT의 값을 적용하여 입력영상의 픽셀이 피부에 포함할 확률 분포를 계산하여 $I(x,y)$를 구한다. CAMShift 알고리즘은 다음과 같이 동작한다.

① 초기에 탐색창의 크기를 정하고, 탐색창을 초기 중심점(mean location)의 중앙에 오도록 위치시킨다.

② 새로운 중심점은 다음의 알고리즘으로 구해진다. 즉, 먼저 아래 (6)에 의해 0차 moment를 구한다.

$$M_{00} = \sum_x \sum_y I(x,y) \tag{6}$$

또, 아래 (7)에 의해 x와 y에 대한 1차 moment를 각각 구한다.

$$M_{10} = \sum_x \sum_y x\, I(x,y), \quad M_{01} = \sum_x \sum_y y\, I(x,y) \tag{7}$$

이때, 새로운 탐색창의 중심점의 좌표는 아래 (8)에 의해 구할 수 있다.

$$x_c = \frac{M_{10}}{M_{00}}, \; y_c = \frac{M_{01}}{M_{00}} \tag{8}$$

③ 새로운 탐색 창에 대하여 위의 과정을 MeanShift 알고리즘이 수렴할 때까지 반복한다. MeanShift는 밀도구배함수(density gradient function) $f(x)$가 최대값을 향해 올라가므로 알고리즘이 수렴하게 되면 모드(mode) 부근에서 $f'(x) \approx 0$가 된다. 중심점 좌표 값의 변화가 정해진 임계 값보다 작으면 알고리즘이 반복을 끝낸다.

### 3. AdaBoost와 CAMShift를 결합한 시스템

Viola와 Jones의 얼굴 검출 방법은 매우 뛰어나지만 매 프레임마다 다양한 크기의 얼굴을 찾기 위해서 멀티스케일의 윈도우를 이용하여 이미지 상의 모든 픽셀을 이동하면서 스캔을 수행하는데 이는 많은 연산을 필요로 한다. 예를 들어, 320x240의 이미지 크기에서 얼굴을 검색하는 윈도우를 20x20에서 40x40까지 늘리면서 영상내의 모든 영역을 검사한다면, 1장의 이미지에서 (9)에 의하여 총 25,849,530회의 검사를 수행하여야 한다.

$$t=\sum_{i=20}^{40}\sum_{j=20}^{40}(320-i+1)\times(240-j+1) \qquad (9)$$

일반적으로 이러한 연산을 줄이기 위하여 검사할 이미지를 1.3배 축소하여서 검사 영역을 작게 만들거나, 검색 윈도우의 최소 사이즈를 조금 크게 하거나, 최소 검색 윈도우 크기에서 최대 검색 윈도우 크기까지의 증가를 +1이 아닌 +2 또는 +3으로 하여 검색 속도를 높인다. 하지만 이러한 방법들은 검출 성능의 손실을 감수하는 것으로 근본적인 해결 방법이라 볼 수 없다.

본 논문에서는 AdaBoost의 검출 성능과 CAMShift의 추적 방법을 결합하여 고속의 얼굴 추출 및 추적이 가능한 시스템을 제안한다. 그림 6은 제안 알고리즘의 전체 흐름도이다. 본 논문에서 제안한 시스템은 AdaBoost로 현재 프레임에 대한 영상에서 얼굴을 검출한다. 얼굴이 검출되면 다음 프레임에서 기존의 AdaBoost 검출 방법처럼 더 이상 전체 영역을 검출하지 않고 CAMShift 를 이용하여 다음 프레임에서 얼굴 영역을 예측하여 검출할 후보 영역을 찾아준다. AdaBoost로 검출된 영역에 대하여 CAMShift의 추적에 필요한 확률분포영상을 만들고 다음 프레임에서 얼굴 검출 기록이 있다면 CAMShift 알고리즘은 이 확률분포영상을 이용하여 현재 프레임에서 고속으로 얼굴의 후보 영역을 제시한다. 그러므로 AdaBoost는 다음 프레임에서 얼굴 검출 시 보다 작은 스캔영역으로 고속의 얼굴 추적이 가능해 진다. ① 최초 프레임에서는 AdaBoost로 전체 영역을 검출하고 ② 얼굴이 검출되면 FaceDetected 값을 true로 설정하고 검출된 얼굴 영역을 저장한다. ④ 다음 프레임에서 FaceDetected 값이 true이면 ⑤ CAMShift 알고리즘으로 저장한 얼굴 영역을 이용하여 후보 영역을 예측하고 false이면 다시 전체 영역을 검출한다. 얼굴이 1개 이상 검출 되었더라도 그 얼굴 영역들을 저장하여 각 영역에 대한 후보영역을 CAMShift 알고리즘을 이용하여 예측한다. ③본 시스템의 초기에는 전 프레임과 현재 프레임의 차가 임계치를 넘는지를 검사하고 있는데 얼굴 검출 및 추적동안 영상에 새로운 객체가 포함되거나 입력 프레임이 전혀 다른 영상으로 바뀌었다면 FaceDetected를 false로 설정하여 다시 전체를 검사하게 된다.

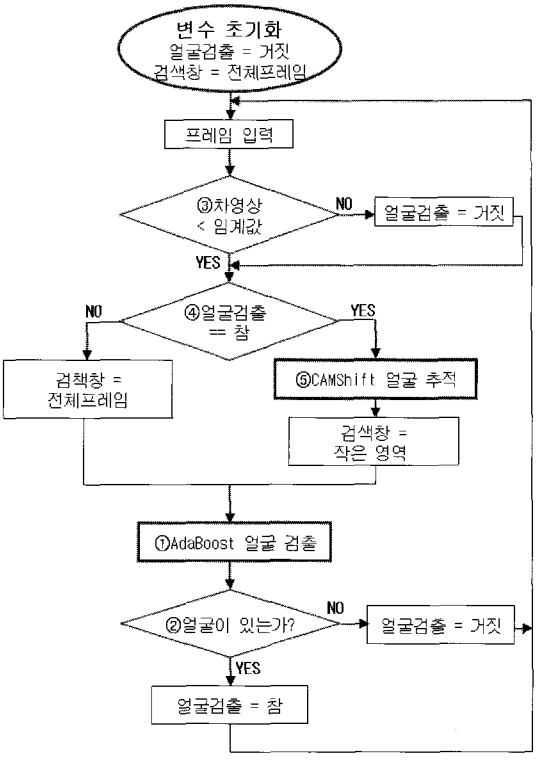



그림 6. 전체 AdaBoost와 CAMShift를 결합한 고속 얼굴 검출 및 추적 시스템.

Fig. 6. A combined algorithm of AdaBoost and CAMShift.

## III. 실험

### 1. AdaBoost 얼굴 검출과 CAMShift 얼굴 추적 비교 실험

그림 7은 AdaBoost와 CAMShift 알고리즘을 6가지 특별한 환경에서 실험한 결과이다. 그림 7(a)~(f)는 각각 다음을

(a) 정상 (b) 조명변화 (c) 어두움

(d) 피부색 잡음 포함 (e) 물체 가림 (f) 2개 이상의 얼굴

그림 7. AdaBosst와 CAMShift의 몇 가지 환경에 대한 비교실험.

Fig. 7. Experiments for several cases on AdaBoost and CAMShift algorithm.

표 2. AdaBoost와 제안 알고리즘의 처리속도 비교.
Table2. Comparison of computing time for AdaBoost algorithm and proposed one.

| Method | Minimum Face Size | Computing time per Frame[ms] |
|---|---|---|
| AdaBoost | 20 * 20 | 47.3 |
| | 40 * 40 | 23.4 |
| 제안 알고리즘 | 20 * 20 | 16.5 |
| | 40 * 40 | 15.8 |

표 3. UMPC(Ultra Mobile Personal Computer)에서 AdaBoost 알고리즘과 제안 알고리즘의 처리 속도 비교.
Table3. Comparison of computing time for the proposed and AdaBoost algorithm in UMPC.

| Method | Minimum Face Size | Computing time per Frame[ms] |
|---|---|---|
| AdaBoost | 20 * 20 | 3,104 |
| | 40 * 40 | 1,703 |
| | 60 * 60 | 901 |
| 제안 알고리즘 | 20 * 20 | 932 |
| | 40 * 40 | 551 |
| | 60 * 60 | 221 |

나타낸다. 그림 7의 상부는 AdaBoost의 실험결과이고 아래는 CamShift의 실험 결과이다. 특징 기반인 AdaBoost는 얼굴의 특징이 있는 곳을 잘 찾아내지만 검출한 객체에 대한 추적 기능이 없어 검출 표시가 깜빡거리며 배경에 대해서 오검출 되는 경우가 발생하였고, CAMShift는 색기반 추적 알고리즘으로 조명이나 주위 얼굴색에 민감하게 반응하였다.

2. AdaBoost와 제안시스템 처리속도 실험

표 2는 *Window XP* OS의 *Intel Pentium-4 Core(TM)2* CPU 6400 2.13G processor의 PC 환경에서 AdaBoost 알고리즘과 제안 시스템의 처리속도를 비교한 것이다. 한 프레임의 영상 크기는 320x240 이고 1분 동안 한 프레임의 얼굴 검출 속도를 측정하여 평균하였으며 속도 측정은 *WIN API*의 *GetTickCount* 함수로 측정하였다. 측정 대상은 한 사람의 얼굴이며, 큰 움직임이 없이 영상 안에 포함되어 있을 경우이다. 표 3은 UMPC(Ultra Mobile Personal Computer)에서 측정한 결과이다. UMPC는 라온디지털의 *VEGA*제품으로 *Window XP* OS의 *AMD Geode TM LX800* @0.9W CPU 이다. 다른 측정 환경은 표 2와 같다.

표 2와 3에서 minimum face size의 의미는 AdaBoost의 검색 윈도우의 최소 사이즈를 의미한다. AdaBoost 알고리즘은 최소 검색 윈도우의 크기가 커지면 스캔할 대상이 작아져 처리 속도가 증가되는 반면에 제안 알고리즘은 AdaBoost 알고리즘으로 얼굴이 한번 검색되면 CamShift 알고리즘으로 추적을 수행하기 때문에 스캔 윈도우의 크기에 상관없는 빠른 속도를 보인다.

표 4는 제안 알고리즘의 사람 얼굴의 개수에 따른 속도 비교이다. 실험 환경은 표 2의 실험 환경과 같으며 최소 얼굴의 크기는 20x20으로 하였다. AdaBoost 알고리즘은 스캔 윈도우가 전 영역을 검사하면서 얼굴일 때 사각형을 그려주는 방법으로 처리속도는 얼굴의 개수에 상관이 없지만 제안알고리즘은 얼굴이 많아질수록 검출된 얼굴 영역에 대한 CAMShift의 추적을 위한 색분포를 저장하고 있어야 하며, 각 얼굴마다 추적을 위한 처리시간이 소요되기 때문에 얼굴의 개수에 따른 속도 차이를 보인다. 그림 8은 실험을 위한 프로그램 실행의 예이다.

표 4. 제안 알고리즘의 얼굴 개수에 따른 처리 속도.
Table4. Computing time according to the number of face.

| Face Count | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Computing time per Frame [ms] | 16.5 | 20.2 | 23.6 | 38.4 | 39 |

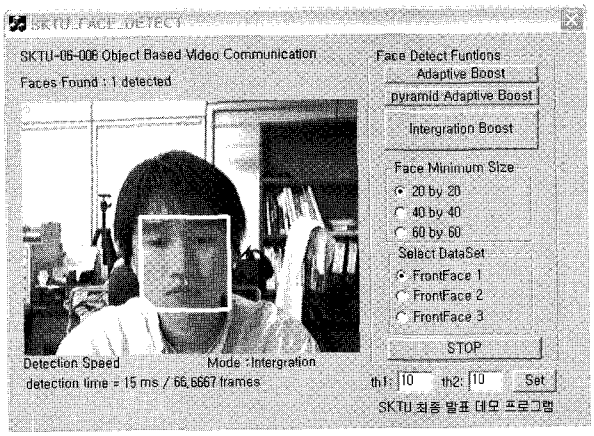



그림 8. AdaBoost와 제안알고리즘의 실험 프로그램.
Fig. 8. A test program of the proposed algorithm.

## IV. 결론

본 논문에서는 multi-task OS, 로봇, 모바일 환경의 실시간 시스템에서 얼굴 검출 및 추적을 고속으로 수행하는 시스템을 제안하였다. 제안된 시스템은 Viola와 Jones가 제안한 실시간 얼굴 검출 AdaBoost 알고리즘과 CAMShift의 고속의 객체 추적 알고리즘을 결합하였다. 기존의 얼굴 추적 방법은 얼굴의 경미한 특징으로 추적에 관하여 많은 연구가 진행되었다. 따라서 얼굴의 특징이 잘 반영되지 않아 여러 가지 잡음에 의하여 추적에 실패하는 경우가 많다. 하지만 AdaBoost알고리즘은 얼굴 검출에 대하여 성능이 매우 우수하여 영상에서 얼굴을 정확하게 검출하며 검출된 영역을 특징을 이용하여 추적함으로써 얼굴 추적의 성능을 높인다. 최초에 AdaBoost 알고리즘을 이용하여 빠르게 얼굴을 검출하고 검출된 영역에 대해서 CAMShift 알고리즘을 이용하여 다음 프레임에서 얼굴 영역을 예측하여 작은 검색 영역을 제시함으로서 전체를 스캔하는 기존 방법을 효율적으로 개선하였다. 본 시스템은 multi-task OS 환경에서 성공적인 얼굴추적속도를 보임을 실험을 통해 확인하였다. 국내 본격적인 화상전화 서비스에 따라 얼굴 검출, 인식에 대한 관심은 더욱 높아 질 전망이므로 관련 연구의 중요성은 점점 더 커질 것이다.

## 참고문헌


[1] M. H. Yang, D. J. Kriegman, and N. Ahuja, "Detecting face in images: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 1, pp. 34-58, 2002.

[2] G. Yang and T. S. Huang, "Human face detection in a complex background," *Pattern Recognition*, vol. 27, no. 1, pp. 53-63, 1994.

[3] H. Y. Wu and Q. Chen, "Detecting human face in color images," *Proc. of the IEEE*, pp. 2332-2336, 1996.

[4] R. Brunelli and T. Poggio, "Face recognition: Features versus templates," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 15 pp. 1042-1052, 1993.

[5] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23-38, Jan. 1998.

[6] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines an application to face detection," *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Puerto Rico, June 1997.

[7] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," In computational learning Theory: *Eurocolt '95*, pp. 23-37. *Springer-Verlag*, 1995.

[8] C. Wang and M. S. Brandstein "A hybrid real-time face tracking system," *Proc. of Acoustics, Speech, and Signal Processing*, vol. 6, 1998.

[9] G. D. Hager and P. N. Belhumeur, "Real-time tracking of image regions with changes in geometry and illumination," *Proc. of Computer Vision and Pattern Recognition*, pp. 403-410, 1996.

[10] S. J. McKenna and S. Gong, "Tracking faces," *Proc. of International conference on Face and Gesture Recognition*, pp. 271-276, 1996.

[11] P. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. of Computer Vision*, vol. 57, no. 2, pp. 137-154, 2004.

[12] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," *Intel Technology Journal*, Q2, 1998.

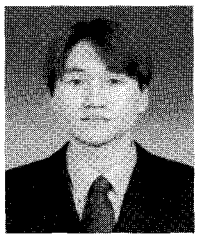
[13] D. Comaniciu and P Meer, "Robust analysis of feature spaces: color image segmentation," *Proc. of CVPR'97*, pp. 750-755. 1997.

[14] R. Brunelli and T. Poggio, "Face recognition: Features versus templates," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 15 pp. 1042-1052, 1993.

[15] Y. Freund and R. E. Schapire, "A short introduction to boosting," *J. of Japanese Society for Artificial Intelligence*, vol. 14, no. 5, pp. 771-780, 1999.

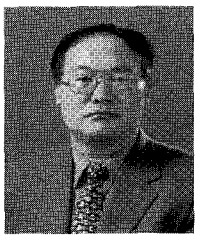
[16] R. Frank, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review(Cornell Aeronautical Laboratory)*, vol. 65, no. 6, pp. 386-408, 1958.

[17] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *Proc. of the 14th Int. Conference on Machine Learning*, 1997.

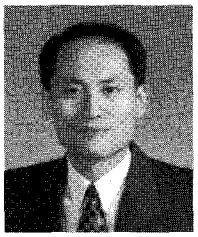

**김 정 현**

2005년 창원대 컴퓨터공학과 졸업. 2005년~2007년 동명대 메카트로닉스공학과 석사. 2007년 현재 부산대 지능기계공학과 박사과정. 관심분야는 컴퓨터 비전, 지능형 로봇, 패턴 인식.


**노 태 정**

1984년 부산대 기계설계학과 졸업. 1986년 KAIST 생산공학과 석사. 1992년 KAIST 정밀기계공학과 박사. 1986년~1999년 삼성중공업 기전연구소. 1999년~현재 동명대 메카트로닉스공학과 부교수. 관심분야는 Mechatronics, Robotics, FAB 물류 자동화, LCD 반송 자동화, 산업설비 원격운전시스템 등.

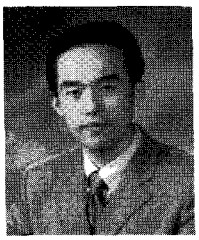

**김 진 영**

1985년 서울대 기계설계학과 졸업. 1988년 KAIST 생산공학과 석사. 1999년 KAIST 자동화 및 설계공학과 박사. 1999년~현재 동명대 메카트로닉스공학과 부교수. 관심분야는 메카트로닉스, 제어 및 자동화.



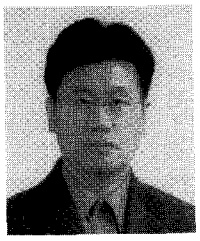
**홍 영 진**

1978년 서울대 전기공학과 졸업. 1983년 SUNY 스토니브룩 석사. 1985년 동대학 박사. 2005년~현재 동명대학교 전기전자공학과 부교수. 관심 분야는 스마트안테나, 무선통신 등.



**권 장 우**

1990년 인하대 전자공학과 졸업. 1992년 인하대학원 전자공학과 석사. 1996년 인하대학원 전자공학과 박사. 1996년 10월~1998년 2월 특허청 심사관. 2004년~현재 동명대 컴퓨터공학과 부교수. 관심분야는 재활공학, 뉴럴네트워크 신호처리, Embedded Micro-controller 시스템 등.



**강 동 중**

1988년 부산대 정밀기계공학과 졸업. 1990년 KAIST 기계공학과 석사. 1999년 KAIST 자동화 및 설계공학과 박사. 1990년~1992년 현대전자 산전연구소 연구원. 1997년~1999년 삼성종합기술원 신호처리연구실 선임연구원. 2004년 미국 Cornell Univ. 방문연구원. 2000년~2005년 동명대 메카트로닉스공학과 조교수. 2006년~현재 부산대 기계공학부 조교수. 관심 분야는 컴퓨터 비전, 이동로봇, 영상기반 검사시스템 개발.