# 무안경식 3차원 모니터용 실시간 눈 추적 알고리즘

# A Real-time Eye Tracking Algorithm for Autostereoscopic 3-Dimensional Monitor

임 영 신, 김 준 식*, 주 효 남
(Young-Shin Lim, Joon-Seek Kim, and Hyonam Joo)

**Abstract:** In this paper, a real-time eye tracking method using fast face detection is proposed. Most of the current eye tracking systems have operational limitations due to sensors, complicated backgrounds, and uneven lighting condition. It also suffers from slow response time which is not proper for a real-time application. The tracking performance is low under complicated background and uneven lighting condition. The proposed algorithm detects face region from acquired image using elliptic Hough transform followed by eye detection within the detected face region using Haar-like features. In order to reduce the computation time in tracking eyes, the algorithm predicts next frame search region from the information obtained in the current frame. Experiments through simulation show good performance of the proposed method under various environments.

**Keywords:** eye tracking, face detection, hough transform, haar-like features, edge detection, partial search method

## I. 서론

실시간 영상에서 얼굴과 눈을 찾아내어 분석, 인식하는 기술은 많이 연구 되고 있으며, 이러한 기술은 의료, 로봇, 전자, 제어분야 등에서 사용 되고 있다. 또한 입체영상을 볼 수 있게 하는 3D 모니터 분야에서도 무안경 시스템을 이용한 방법을 연구하고 있다. 3D 모니터란 입체영상을 볼 수 있게 해주는 모니터로써 지금까지 많이 개발되고 있다. 3D 모니터에서 3차원 입체영상을 보는 방법은 사람의 눈을 추적하여, LCD의 barrier들을 제어시켜 사람이 보는 방향에 따라 3차원 입체영상을 볼 수 있게 해준다. 3차원 입체영상을 정확하게 보려면 빠르게 얼굴과 눈을 검출하고 추적해야 한다. 얼굴을 검출하는 기술은 그림 1과 같이 분류되어진다[1].

특징 기반 방법은 얼굴의 특징 성분들 즉, 불변하는 얼굴의 구조적 특징을 이용하여 위치를 찾는 방법이며, 얼굴의 색상과 질감, 요소들의 혼합된 형태의 정보를 이용한다. 특히, 얼굴색을 이용한 방법이 최근 많이 사용되고 있으며, 얼굴의 이동 및 회전, 크기변화에 강인한 장점이 있지만, 영상에서 조명 조건이 제각각이기 때문에 색상 정보를 찾기가 어렵다는 단점이 있다. 또한 영상에서 얼굴이 아닌데 비슷한 색상이 있는 영역을 얼굴로 검출할 수 있으므로 다른 방법과 혼합하여 사용한다[2-4].

지식 기반 방법은 얼굴 영상에서 이마, 눈, 코, 입, 턱의 예상 위치를 확인하고 이들의 상대적인 위치를 나타내는 특징 벡터를 모델 정보로 이용한다. 하지만, 인간의 얼굴에 대한 지식을 정확하게 정의하기 어렵고, 다양한 얼굴포즈 마다 재정의 되어야 하는 단점이 있다[5].

템플릿 기반은 얼굴의 표준 템플릿을 생성하여 저장하고, 입력 영상에서 비교하여 얼굴을 검출하는 방법이다. 하지만

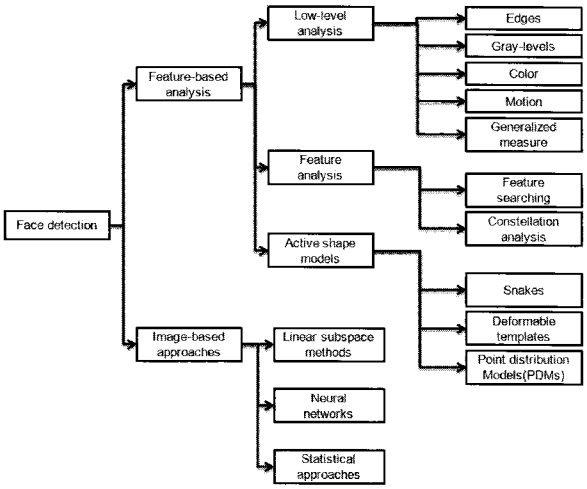



그림 1. 얼굴 검출 기술 분류.
Fig. 1. Face detection techniques.

템플릿 방법은 얼굴의 다양한 크기 와 자세, 형태를 효과적으로 처리하기 위해서는 다양 한 패턴을 미리 정의하여야 하므로 시간과 비용이 매우 높다는 단점이 있다[6-8].

눈 검출 방법은 Beymer가 템플릿 매칭을 이용하여 눈을 검출하였다. 이 방법은 템플릿 영상과 입력 영상 사이의 유사성을 이용한 것이며, 템플릿 영상에 대한 초기 위치정보에 매우 큰 영향을 받는다[9]. 또한 Rizon는 입력 영상의 명암도와 가장자리 정보를 통하여 눈동자를 찾는 방법을 시도하였다[10].

최근에는 Adaboost training을 이용한 Haar-like features방법이 사용되고 있다[11,12]. Adaboost는 약한 분류기를 조합하여 강한 분류기로 만드는 boosting 알고리즘의 일종이다. 또한 이 방법은 연산속도가 빠르고 눈 검출에도 사용가능 한 장점이 있으며, 얼굴 검출 시 정면 얼굴만 검출 가능하고, 눈 검출 시에는 조명에 약한 단점이 있다.

본 논문에서는 3D 모니터용 실시간 눈 추적 알고리즘을 제안하며, 허프변환(Hough Transform), Haar-like features 기반의

* 책임저자(Corresponding Author)
논문접수: 2009. 3. 26., 수정: 2009. 4. 21., 채택확정: 2009. 5. 27.
임영신, 주효남: 호서대학교 디지털디스플레이공학과
(seagull6090@hanmail.net/hnjoo@hoseo.edu)
김준식: 호서대학교 전자공학과(joonskim@hoseo.edu)
※ 본 연구는 교육과학기술부와 한국산업기술재단의 지역혁신인력양성사업으로 수행된 연구결과임.

눈 검출 및 추적 알고리즘이다. 그리고 사람의 얼굴을 정확히 검출하여야 하며, 실시간 적으로 3차원 정보를 3D 모니터에 정확하게 표시하기 위해서는 실시간 눈 추적이 가능해야 하므로 제안한 알고리즘을 사용하였다. 제안한 알고리즘은 초기 전처리 과정을 통하여 빠르고, 정확하게 얼굴 영역을 검출할 수 있다. 또한 눈 검출과 추적을 동시에 할 수 있는 장점이 있다. 영상에서 사람의 얼굴은 타원형이기 때문에 허프변환을 사용하였으며, 찾아진 얼굴 영역에서 Haar-like features를 이용하여 눈을 찾았다. 또한 실시간 검출 속도를 올리기 위하여 다음 프레임에서는 이전에 찾았던 얼굴영역을 기준으로 탐색영역을 설정하여 눈을 찾았다.

II 장에서는 제안한 알고리즘을 자세하게 설명한다. 실험 방법과 결과는 III 장에서 그리고 결론은 IV 장에서 설명 한다.

## II. 제안한 알고리즘

그림 2는 제안한 알고리즘의 단순한 순서도이다. 전처리의 목적은 연속적인 영상에서의 최소 edge 점들을 찾기 위해서 한다. 이유는 허프변환 시 계산 속도가 느리기 때문이다. 허프변환은 얼굴에서 (타)원을 추출하기 위하여 거치는 단계이다. 마지막으로 Haar-like features로 얼굴 영역을 찾은 범위에서 두개의 feature들을 합쳐 눈을 찾는다. 눈 추적 단계에서는 partial search 방법을 사용하여, 새로운 ROI를 설정하여 눈을 추적하였다.

### 1. 전처리

전처리 과정의 목적은 허프변환 과정에서 사용하는 edge 픽셀을 찾는 것이다. 그림 3은 전처리 과정의 순서도다. 처음으로 가우시안 필터(gaussian filter)를 이용하여 고주파 잡음을 제거하여 주며, 그 후 소벨(sobel edge)검출기로 edge 성분들을 찾아내고, non maximum edge pixel suppression[13]을 이용하여 edge 픽셀의 개수를 줄였다.

허프변환은 계산하는 시간이 오래 걸리기 때문에 ROI영역을 설정하여 계산 시간을 줄였다. 또한 눈을 찾고 실시간으로 추적하기 위해서 움직임이 있을 경우와 사람의 이동을 탐지했을 때 ROI를 설정하고, 만약 그렇지 않으면 미리 탐지한 얼굴 지역으로 검색 지역을 제한한다.

식 (1)은 움직임 정보를 이용하여 차영상을 구하는 식이며, 차영상은 ROI를 지정하는데 사용된다. 여기서 $f_t(x,y)$는 시간 t에 대한 입력 영상이다. $f_{t-2}(x,y)$는 현재 영상에서 2 프레임

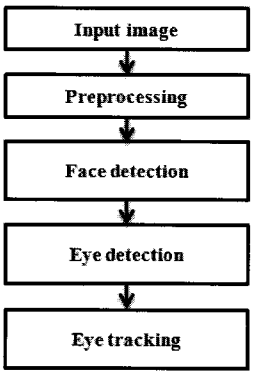



그림 2. 제안한 알고리즘 순서도.
Fig. 2. Proposed algorithm flowchart.

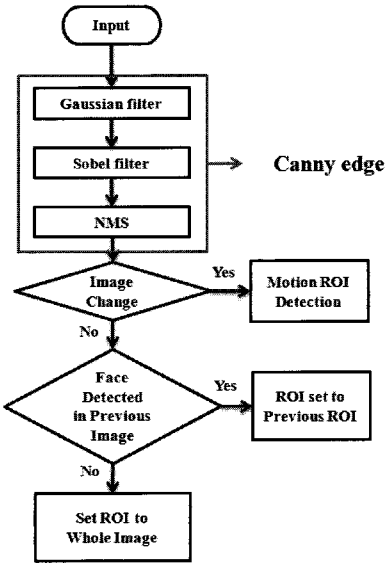



그림 3. 전처리 순서도.
Fig. 3. Preprocessing step flowchart.

전 영상이며 $f_{t-1}(x,y)$은 1 프레임 전 영상이다. $f_{t-2}(x,y)$, $f_{t-1}(x,y)$ 영상의 평균과 현재 프레임 $f_t(x,y)$을 이용하여 차영상 $Diff_t(x,y)$를 정의 하였다.

$$Diff_t(x,y)=\left|\frac{f_{t-2}(x,y)+f_{t-1}(x,y)}{2}-f_t(x,y)\right| \tag{1}$$

$Diff_t(x,y)$는 얼굴 검출을 위한 ROI를 설정하기 위해 얼굴 움직임을 나타내는 차영상이다. 만약 현재 영상에 변화가 없다면, 이전 영상에서 영역을 재설정 하거나 전체 영상에서 ROI 영역을 찾는다.

### 2. 허프변환을 이용한 얼굴 검출

허프변환은 직선, 원, 타원 그리고 일반적인 모양들을 찾아내고 많은 영상처리 응용분야에서 사용 중이다. 본 논문에서는 사람의 얼굴 형태가 (타)원에 가깝기 때문에 타원의 방정식을 이용한 허프변환을 이용 하였다. 타원의 방정식에서 중심($C_x$, $C_y$)와 semi-axis $a$와 $b$의 관계는 식 (2)와 같으며,

$$\frac{(x-C_x)^2}{a^2}+\frac{(y-C_y)^2}{b^2}=1 \tag{2}$$

이를 변환하면,

$$\begin{aligned} x&=r\cos\theta+C_x \\ y&=r\sin\theta+C_y \end{aligned} \tag{3}$$

극 좌표 방법을 사용하여 식 (4)와 같이 변환 시키며,

$$r^2=\frac{a^2b^2}{a^2\sin^2\theta+b^2\cos^2\theta} \tag{4}$$

ab_ratio(=$b/a$)는 장축과 단축의 비율로, 원과 타원에 적응적으로 동작하기 위한 파라미터이다.

$$r^2=\frac{a^2}{\cos^2+\dfrac{\sin^2\theta}{(ab\_ratio)^2}} \tag{5}$$

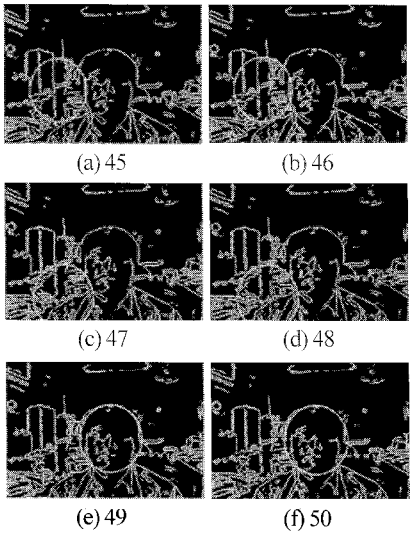



그림 4. 여러 개의 a 값을 이용한 얼굴 검출 결과.
Fig. 4. Face detection using different values of "a".

허프변환 테이블은 식 (5)와 같이 ($C_x$, $C_y$, $a$)와 ab_ratio의 변화에 따라 사람 얼굴의 형태에 알맞은 값을 찾아낸다. 허프변환 테이블의 최고 포인트의 숫자를 분류하고 가장 얼굴과 비슷한 값을 찾아내어 얼굴 위치 후보를 검출한다. 우리가 사용한 ab_ratio값은 1~1.5값을 선택하여 실험하였으며, 가장 적합한 값을 찾아내었다. 또한 a값은 원의 크기를 나타낸다. 3D 모니터에서 입체영상을 보려면 450mm거리에서 보면 가장 잘 보인다. 그러므로 허프변환 시 얼굴의 크기는 고정하여 실험을 하였다. 하지만 사람의 얼굴의 크기는 틀리므로 a 값을 변경해가며 실험을 하였다. 그림 4는 설정한 a값(45~50)의 크기를 늘려가며 얼굴을 찾는 방법의 예다.

### 3. Haar-like features를 사용한 눈 검출

Haar-like features는 많은 응용분야에서 눈 검출을 성공적으로 찾아낸다.

우리가 사용한 Haar-like fatures[9,10]들은 그림 5와 같으며, 이 중에서 눈을 찾아낼 수 있는 가장 적합한 feature들을 선택하였다. Haar-like features는 검은색 부분의 합과 흰색부분의 합의 차를 이용하며, 우리가 사용한 feature는 (b), (e)다. 눈의 위치를 찾는 방법은 눈 위치에 feature의 흰 부분은 눈동자와 눈썹을 포함한 부분에 위치해 있고, 검은 부분은 바로 밑에

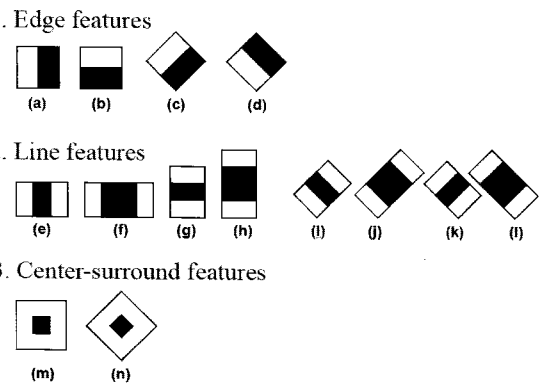
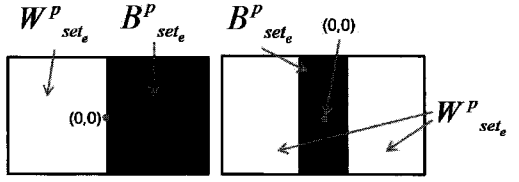
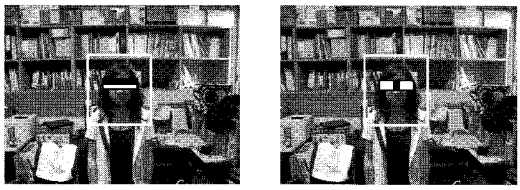



그림 5. Haar-like features.
Fig. 5. A set of Haar-like features.




(a) Edge feature (b) Line feature

그림 6. 눈 검출에 사용한 Haar-like features의 구조.
Fig. 6. Structures of Haar-like features for eye detection.



(a) (b)

그림 7. 눈 영역으로 이동한 Haar-like features.
Fig. 7. Example of Haar-like features mask positioned at eye location.

분포하고 있다고 한다면, 흰 부분은 값이 적을 것이고, 검은 부분은 흰 부분에 비해 높을 것이다. 이유는 흰 부분에서는 눈동자와 눈썹 등 화소값이 작은 영역이 많이 포함되어 있고, 검은 부분은 적기 때문에 이 두 영역의 값을 합하면 얼굴의 다른 영역에서의 합의 값보다 크기 때문에 눈의 위치를 찾을 수 있다. 이 방법을 이용하여 feature들의 최대값을 구하고, 두 feature의 최대값들을 합하여 최종 눈 위치를 찾는다.

Edge와 line feature를 구하기 위한 2개의 Haar-like 마스크는 그림 6과 같으며, 영상의 화소 좌표 p에서 그림 6(a)의 edge feature 값 ($f_e(p)$) 와 그림 6(b)의 line feature 값 ($f_l(p)$) 은 식 (6)과 같이 계산한다.

$$f_e(p) = \sum_{p \in W^p{}_{set_e}} f(p) - \sum_{p \in B^p{}_{set_e}} f(p)$$
$$f_l(p) = \sum_{p \in W^p{}_{set_l}} f(p) - \sum_{p \in B^p{}_{set_l}} f(p) \quad (6)$$

where, for k=e, l,

$W^p{}_{set_k}$ : $W_{set_k}$ translated at p

$B^p{}_{set_k}$ : $B_{set_k}$ translated at p

그림 7은 알고리즘에서 사용한 두개의 Haar-like features로 찾아진 눈 영역의 위치에 놓여진 것을 보여 준다.

### 4. Partial search 방법을 이용한 눈 추적

우리가 눈을 추적하기 위해서는 사람의 움직임이 계속적으로 있어야 하며, 빠르게 눈을 검출하고 추적할 수 있어야 한다. Partial search 방법은 계산시간을 줄여 눈을 추적하고, 부분적으로 검출하는 방법이다.

우리는 연속적인 프레임에서 사람이 얼굴을 움직이는 공간은 상하좌우로 10~20pixle정도로만 움직이고 속도는 빠르지 않다고 가정 하였다.

그림 8은 우리가 제안한 partial search 방법이다. 처음에 설정된 ROI영역에서 얼굴 검출 시 타원의 중심 $f_{t-1}(C_x, C_y)$에 대

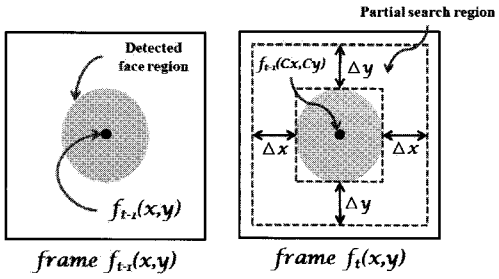



그림 8. Partial search 방법의 search region.
Fig. 8. Search region of partial search method.

한 값을 구할 수 있기 때문에 이 중심에서 상하좌우로 10~20 pixel로 새로운 ROI를 설정한다. 우리는 이 값을 이용하여 계산시간을 줄이고, 사람이 이동할 경우 적절하게 눈을 추적할 수 있다. 그러므로, 사람의 움직임이 관찰될 경우 부분적으로 허프변환이 가능하다. 또한 추적필터는 사용하지 않았다. 이 유는 사람이 움직이는 속도는 느리다고 가정하였으므로 추적필터 없이 눈을 추적할 수 있다. 탐색영역 설정 시 그림 9의 수평과 수직의 변위는 각각 Δx와 Δy이다.

## III. 실험 방법과 결과

이 장에서는 우리가 실험한 방법과 제안한 알고리즘의 결과를 설명한다.

### 1. 실험 조건

USB 카메라의 해상도는 320 x 240이고, 수평 방향의 FOV (Field of View)는 70°이다. 카메라와 사람과의 거리는 450mm 이며, 이 거리에서 카메라로 찍은 얼굴이 영상에서 차지하는 pixel 영역은 180~200 pixel이다. 또한 모니터를 바라보는 사람이 움직이는 각은 35°로 제한하고 실험하였다.

이 실험에서 a 값은 45~55로 정하였으며, ab_ratio는 1.2~1.5로 값을 찾아 정하였다. 그리고 partial search방법의 Δx와 Δy의 값을 각각 ±20과 ±10 pixel로 설정하였다.

### 2. 실험 결과

그림 9는 전처리 과정의 결과값이다. (a)는 입력 영상이며, (b)는 edge성분들을 찾아내고 non maximum suppression을 이용하여 edge의 폭을 한 픽셀로 가늘게 한 결과 영상이다.

그림 10은 전처리 과정에서 움직임을 이용한 ROI 영역을 설정한 결과이다. 영상에 움직임이 있을 때 차영상을 구하여, 초기 ROI를 설정한다. 이를 이용하여 얼굴 검출 시간을 단축시킬 수 있다.

그림 11(a)와 (b)는 허프변환 테이블 결과이며, 영상에서

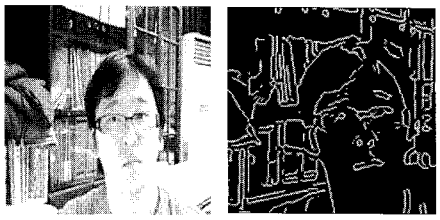



그림 9. 전처리 과정의 결과.
Fig. 9. Result of preprocessing stage.

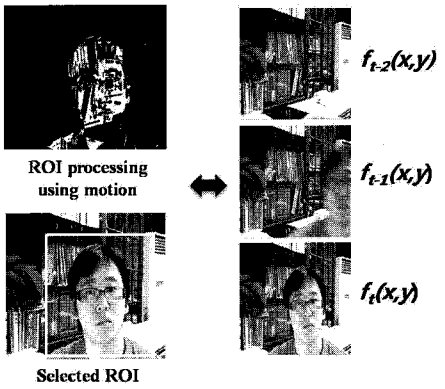



그림 10. 차영상을 이용한 ROI 영역 설정 결과.
Fig. 10. Result of ROI processing using difference image.

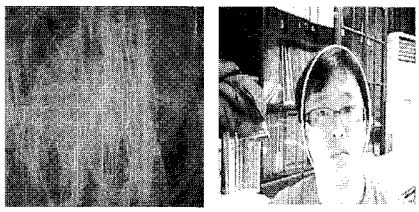



그림 11. 허프변환 테이블과 얼굴검출 결과.
Fig. 11. Hough transform result and the detected ellipse.

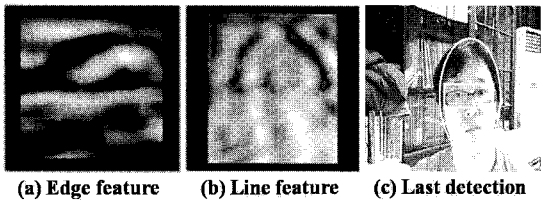



그림 12. Haar-like feature 영상과 눈 검출 결과.
Fig. 12. Haar-like feature images and the detected eye location.

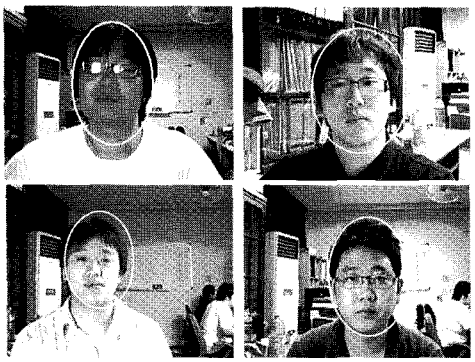
얼굴을 검출한 결과이다. 이 결과를 보면 타원이 얼굴형태와 비슷하다는 것을 알 수 있다.

그림 12(a), (b)는 edge feature와 line feature들의 계산 결과 영상이며 눈 부분에서 큰 값을 갖는 것을 볼 수 있다. 그림 12(b)는 이 feature들을 이용하여 얼굴 영역에서의 눈을 검출한 결과 영상이다.

제안한 알고리즘이 다양한 조건에서 강인한지 실험하기 위해 우리는 다른 스케일과 다양한 자세와 조명 변화, 배경이 복잡한 상황을 만들어 실험을 하였다. 그림 13은 여러 조건에서의 얼굴 검출 결과 영상이며, 다양한 조건에서도 얼굴을 검출 하는 것을 보였다. 그림 14는 눈 검출 결과 영상이며, 성공적으로 검출된 것을 확인 하였다.

그림 15는 연속적인 영상에서의 얼굴 검출 결과이다. 사람이 움직일 때 마다 얼굴의 ROI 영역이 partial search 방법으로 빠르게 추적하고 있음을 알 수 있다.

알고리즘의 성능을 확인하기 위하여 검출률을 구하였다. 연속적인 영상에서 얼굴과 눈이 검출된 영상을 저장하여, 검출된 영상에 전체영상을 나누어 주었다. 검출률을 구하는 식은 (7)과 같다.



(a) Frontal face detection

(b) Side face detection

(c) Face detection among many faces

그림 13. 여러 환경에서의 검출 결과.

Fig. 13. Face detection on various environment conditions.

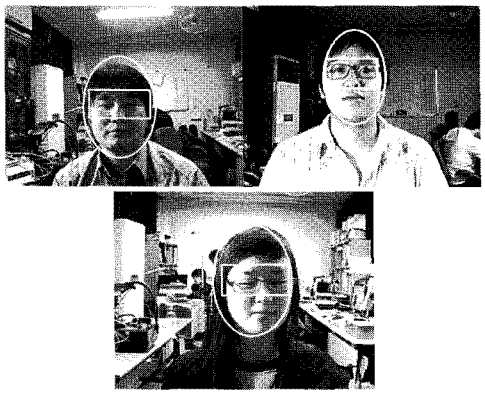


그림 14. 얼굴영역에서의 눈 검출 결과.

Fig. 14. Eye detection from the detected face area.

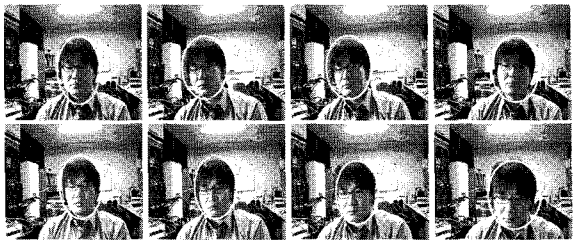


그림 15. 얼굴 추적 결과.

Fig. 15. Result of face tracking.

$$\text{얼굴(눈)검출률}=\frac{\text{얼굴(눈) 검출 영상수}}{\text{전체 입력 영상수}}\times 100(\%) \quad (7)$$

표 1. 얼굴 검출률.

Table 1. Face detection ratio.

| 입력 영상 | 정면 검출률 | 측면 검출률 |
|---|---|---|
| 400 프레임 | 95% | 84% |

그림 16. 눈 추적 결과.

Fig. 16. Result of eye tracking.

표 2. 눈 검출률.

Table 2. Eye detection ratio.

| 입력 영상 | 정면 검출률 | 측면 검출률 |
|---|---|---|
| 400 프레임 | 95% | 80% |

표 1은 얼굴 검출률을 표로 나타낸 것이다. 정면은 0~10° 내에서 검출 된 영상이며, 측면은 10~35° 내에서 검출 된 영상이다. 카메라로부터 입력되는 영상신호로부터 400개의 프레임에 대하여 실험한 결과 정면은 95% 이상의 검출률을 얻었으며, 측면에서는 84% 이상의 검출률을 얻었다.

그림 16은 연속적인 영상에서의 눈 검출 결과이다.

Partial search 방법을 이용하여 얼굴 추적을 하고, 얼굴을 찾은 영역 안에서 눈을 찾고, 추적하였다.

표 2는 눈 검출 결과를 표로 나타낸 것이다. 정면에서는 95%이상의 검출률을 보였다. 측면에서는 80%의 검출률을 얻었다.

3D 모니터에 적용하여 실행시간을 측정한 결과 프레임당 7ms로 추적하는 것을 보였으며, 입체영상을 정확하게 보는 것을 확인하였다.

## IV. 결론

타원형 허프변환과 Haar-like features들의 방법에 기반을 둔 얼굴 및 눈 검출 방법을 제안한다. 우리는 허프변환이 성공적으로 얼굴의 타원형태를 찾는 것을 보고, Haar-like features가 정확하게 눈의 위치를 찾는 것을 확인하였다.

제안한 알고리즘은 성공적으로 edge점들을 줄였으며, 이 결과로 인하여 속도도 많이 향상 되었다. 또한 실시간으로 눈을 찾는 방법도 partial search 방법을 이용하여 좋은 결과를 얻었다. 그래서 제안한 알고리즘은 실시간 영상에서 빠르게 눈을 추적할 수 있다는 결과를 확인하였다.

3D 모니터에 적용하여 실험한 결과 사람이 보는 방향에 따라 입체영상이 보여짐을 확인하였으며, 빠르게 눈을 추적하였다.

## 참고문헌


[1] E. Hjelmås, "Face detection: a survey," *Comput-er Vision and Image Understanding*, vol. 83, pp. 236-274, 2001.

[2] G. Xu and T. Sugimoto, "Rits eye: a software-based syst-em for real-time face detection and tracking using pan-ti-lt-zoom controllable camera," *in Proc. of International Conference on Pattern Recognition*, vol. 2, pp. 1194-1197, 1998.

[3] K.-C. Jin and J-H Cho, "Automatic face feature tracking," P-*roceedings of the 1999 IEEE Region 10 Conference TE-NCON'99*, vol. 1, pp. 68-71, Cheju, Korea, 1999.

[4] 신윤희, 강신국, 김은이, "얼굴 특징 추적을 이용한 인터페이스 구현," 한국정보과학회 학술발표논문집, pp. 274-276, 2006.

[5] L. Harmon & W. Hunt, "Automatic recognition of human face profile," *CVGIP*, vol. 6, pp. 135-156, 1977.

[6] 이경미, "이질적 템플릿 매칭의 융합을 이용한 얼굴 영역 검출," 한국콘텐츠학회논문지, 제7권 제12호, pp. 311-321, 2007.

[7] I. Craw, D. Tock, and A. Bennett, "Finding face features," *In proc. ECCV*, pp. 92-96, 1992.

[8] A. Lanitis, C. J. Taylor, and T. F. Cootes, "An automatic face identification system using flexible appearance models," *Image and Vision Computing*, vol. 13, no. 5, pp. 393-401, 1995.

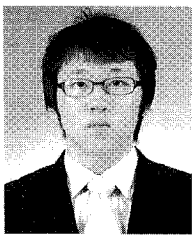
[9] D. Beymer, "Face recognition under varying pose," *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 756-761,1994.

[10] T. Kawaguchi and M. Rizon, "Iris detection using intensity and edge information," *Pattern Recognition*, vol. 36, no. 22, pp. 549-562, 2003.

[11] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision*, pp. 137-154, 2002.

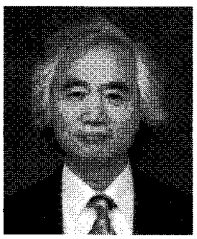
[12] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," *IEEE ICIP 2002*, vol. 1, pp. 900-903, 2002.

[13] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679-698, 1986.


**임 영 신**

1984년 12월 9일생. 2007년 호서대학교 기계공학과(공학사). 2008년~현재 호서대학교 대학원 디지털 디스플레이 공학과 석사과정 재학 중. 관심분야는 광학 시스템 설계, 패턴인식.


**주 효 남**

1953년 8월 2일생. 1976년 서울대학교 전기공학(공학사). 1976년~1982년 국방과학연구소(선임연구원). 1985년 Virginia Polytechnic Institute & State Univ. VA, USA 전기전자공학(공학석사). 1985년~1987년 Machine Vision International(선임연구원). 1990년~1996년 The Boeing Company(Principal Engineer). 1991년 Univ. of Washington 전기전자공학(공학박사). 1996년~2000년 삼성전자 생산기술센터 자동화연구소(연구소장). 2000년~2002년 (주)넥스트아이 연구개발 부문(사장). 2002년~현재 호서대학교 디스플레이공학부 교수. 2003년~2007년 호서대학교 반도체 제조장비 국산화 연구센터(소장). 최근 연구과제로는 반도체 소자의 결함 검사를 위한 Vision Inspection Module 개발.



**김 준 식**

1963년 4월 8일생. 1987년 2월 서강대학교 전자공학과 졸업. 1989년 2월 서강대학교 대학원 전자공학과 졸업(석사). 1993년 8월 서강대학교 대학원 전자공학과 졸업(박사). 1993년 9월~1994년 2월 서강대학교 부설산업기술연구소 박사후연구원. 1994년 3월~현재 호서 대학교 전자공학과 교수. 2007년 1월~2008년 2월 Southern Oregon University 방문교수. 관심분야는 digital image processing, machine vision, 영상 압축, 반도체/디스플레이 검사장비 등.