논문 2007-44CI-1-13

SoC 하드웨어 설계를 위한 얼굴 인식 알고리즘의 고정 소수점 모델 구현 및 성능 분석

(Fixed-Point Modeling and Performance Analysis of a Face Recognition Algorithm For Hardware Design)

김 영 진*, 정 용 진**

(Young-jin Kim and Yong-jin Jeong)

요 약

본 논문에서는 얼굴 인식 알고리즘을 하드웨어로 설계하여 임베디드 시스템에 적용하기 위해 고정 소수점 모델을 구성하고 그에 근거한 하드웨어 구조를 제안하였다. 얼굴 인식 알고리즘은 학습된 데이터를 사용하여 입력 영상에서 얼굴을 검출하고 검출된 얼굴 영상에서 두 눈을 찾아 얼굴 검증 단계를 거치며, 얼굴 검증단계에서 얻어진 두 눈의 위치를 이용하여 얼굴 인식 단계에서 필요한 얼굴 특징 벡터를 연산하고 저장 또는 비교를 통하여 얼굴 인식을 수행한다. 부동 소수점 모델과 고정 소수점 모델의 유사성은 부동 소수점 모델에서 검출된 영상을 고정 소수점 모델에서 수행하여 비교하였으며 성능이 0.2% 오차 범위 안에서 일치하는 고정 소수점 모델을 구성하였다.

Abstract

This paper includes an analysis of face recognition algorithm to design hardware and presents fixed point model in accordance with it. Face recognition algorithm detects the positions of face and eyes to make use of their feature data to detect and verify human faces. It distinguishes a particular user by means of comparing them with registered face features. To implement the face recognition algorithm into hardware, we developed its fixed point model by analyzing face feature parameters, face acquisition data, and feature detection parameters and operation structure.

Keywords: 얼굴 검출, 눈 검출, 얼굴 인식, 생체 인식, fixed modeling

Ⅰ. 서 론

얼굴 인식은 중요 생체 인식 기술 중의 하나로서, 지문 인식, 홍채 인식 등의 다른 생체 인식 기술과는 달리 비접촉식 방법이라는 장점을 가지고 있으며, 차세대 인식기술로 관심이 집중 되고 있다. 하지만 대부분의 얼굴 인식 알고리즘은 부동 소수점 모델을 적용하여 구성되어 있어서 임베디드 시스템이나 하드웨어 설계에 곧바로 적용하기가 쉽지 않다.

본 논문에서는 부동 소수점 모델로 설계된 얼굴 인식 알고리즘을 고정 소수점 모델로 구현하고, 성능을 분석하여 임베디드 시스템에의 구현과 하드웨어 설계에 적용할 수 있도록 하였다. 내용의 구성은 Ⅱ장에서 얼굴 인식 알고리즘에 대하여 설명하고, Ⅲ장에서 얼굴 인식 알고리즘의 부동 소수점 모델과 고정 소수점 모델의 유사성을 판단하기 위해 FRR(False Rejection Rate)을 선택하여 성능을 분석하고 고정 소수점 모델을 구현하였으며, Ⅳ장에서 얼굴 인식 하드웨어를 위한 구조를 제안한다. 마지막으로 Ⅴ장에서 결론을 맺는다.

* 학생회원, ** 정회원, 광운대학교 전자통신공학과
(Dept. of Electronics and Communication Engineering Kwangwoon University)

※ 본 연구는 광운대학교 2006 교내학술 연구 및 정보통신부의 선도기반기술 개발 사업의 지원으로 이루어졌습니다.

접수일자: 2006년11월13일, 수정완료일: 2007년1월4일

Ⅱ. 얼굴 인식 알고리즘

본 논문에서 사용한 얼굴 인식 알고리즘은 영상의 픽셀 값을 누적시키는 인테그럴 이미지(Integral Image), 얼굴 영역을 추정하는 얼굴 검출(Face Detection), 눈 후보 위치를 검출하기 위한 전처리 단계인 이진영상화/침식/확장, 커넥티드 컴포넌트(Connected Component)[7]와 눈 후보 위치 검출(Eye Detection), 얼굴로 추정된 영상을 정규화된 영상으로 변환시키는 어파인 트랜스폼(Affine Transform)과 바이리니어 인터폴래이션(Bilinear Interpolation), 너무 밝거나 어두운 영상의 조명 특성을 재편성하는 히스토그램 평활화(Histogram Equalization), 추정된 얼굴 영역과 눈 후보 위치로서 얼굴인지 확인 하는 얼굴 검증단계(Face Verification)와 얼굴 등록/인증을 처리하기 위한 전처리 작업인 영상 회전/확대/축소, 그래디언트 앵글(Gradient Angle)이 있으며, 얼굴을 등록하고 인증 하는 얼굴 인식단계(Face Recognition)로 구성되어 있다.

얼굴 인식 알고리즘 중 엄굴 검출 알고리즘은 Paul Viola와 Michael가 제안한 Robust Real-Time Object Detection 방법[1]으로 얼굴 검출단계를 구성하였으며, 눈 검출 알고리즘으로서는 커넥티드 컴포넌트, 얼굴 검증 알고리즘은 SVM(Support Vector Machine)[2], 얼굴 인식 알고리즘은 LFA(Local Feature Analysis)[10]를 사용하였다. 본 논문에서 사용된 얼굴 인식 알고리즘은 2차원의 그레이(Gray)영상을 사용하여 학습된 얼굴 특징 데이터로 비교 연산하여 얼굴을 검출하는 알고리즘이며, 하드웨어로 구현하였을 경우 다른 알고리즘보다 구조 및 성능면에서 더 효율적이기 때문에 선택하였다. 본 논문에서 얼굴 인식 단계는 등록자의 수가 가변적이므로 얼굴을 인식하기 위해 사용되는 얼굴의 특징벡터를 계산하는 부분까지 하드웨어로 수행하기 위해 고정 소수점 모델을 구현하고 얼굴 특징 벡터를 비교하는 부분은 소프트웨어로 처리하였다.

1. 인테그럴 이미지

인테그럴 이미지는 얼굴 검출에서 사용되는 8가지 형태의 사각형으로 수행되는 특징 연산을 고속으로 수행하기 위한 전처리 단계로서 그림 1과 같이 입력된 영상의 픽셀 값을 누적 시키는 단계이다.

$i(x,y)$: 원 영상
$s(x,y)$: 열의 누적 값
$ii(x,y)$: 인테그럴 이미지

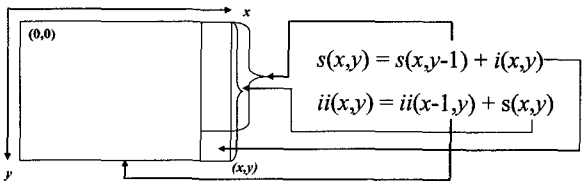


그림 1. 인테그럴 이미지 수행과정
Fig. 1. Operation of integral image process.

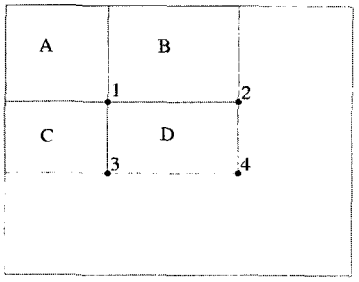


그림 2. 인테그럴 이미지를 이용한 D 영역 계산
Fig. 2. Integral image for D area.

이 과정을 수행하여 얻은 인테그럴 이미지 값은 얼굴 검출에서 필터링을 수행하기 위한 영역간의 뺄셈인 특징 연산의 수행시간을 줄일 수 있는 장점을 가져온다. 예를 들어, 그림 2에서 D의 영역을 수식 (1)과 같이 구하면 계속적으로 바뀌는 영역에 포함된 모든 값을 누적하는 연산을 인테그럴 이미지 값으로 대체 할 수 있다.

$$D = ii(4) + ii(1) - ii(2) - ii(3) \quad (1)$$

2. 얼굴 검출

얼굴 검출의 특징은 수식 (2)에서와 같이 각각의 필터에 의해 얻어지는 값과 각각의 필터의 가중치 ($alpha$)의 곱에 의해 얻어지는 값들의 선형조합에 의해 최종 얻어지는 값이 최종 임계치보다 크면 얼굴로 판단하는 구조로 되어 있다. 이것은 일정한 비율을 갖는 윈도우의 크기를 바꿔 가며, 윈도우 내부에 존재하는 674개의 Feature를 Cascade 단계별(12단계)로 비교하고, 해당 윈도우에 얼굴이 존재할 수 있는 확률 값을 연산하여, 최종적으로 얼굴이 존재하는 윈도우를 찾아 해당 윈도우의 좌표를 출력한다.

$alpha$: 가중치
$threshold$: 임계치
$direction$: 부호
S : 해당 영역 픽셀의 총합

$$\begin{aligned} &\text{if } (direction \times |S_2 - S_1| > \\ &direction \times threshold) \ then \ acc += alpha \\ &\text{if } (acc > threshold_\alpha) \ then \ face \end{aligned} \quad (2)$$

비교 윈도우의 최소 크기는 30×35픽셀이며, 입력 영상의 크기에 따라 최대 크기는 달라진다. 최소 크기의 윈도우 크기를 기준으로 비교할 Feature의 좌표 정보를 입력하고, 최소 크기를 기준으로 0.09375(3/32)배씩 윈도우 크기를 늘여가며, 최대 크기의 윈도우 크기가 될 때 까지 비교 연산을 수행한다.

3. 히스토그램 평활화

히스토그램 평활화는 얼굴 인식 알고리즘 중 얼굴 검증에서 조명에 의해 너무 밝거나 어두운 영상을 보정하기 위한 과정이며, 이를 수행하여 영상의 보다 세밀한 부분을 표현할 수 있다.

입력된 25×20픽셀 크기의 8비트 그레이 영상의 밝기를 보정하기 위하여 영상의 각 픽셀 값의 특성을 누적 분포하고 이를 정규화 하여, 정규화 된 값을 참조하여 픽셀 값을 보정한다. 히스토그램 평활화를 수행하기 위해서 다음의 네 가지 단계를 거친다.

① 영상을 한 픽셀씩 불러들여 각 레벨의 개수를 누적하여 0부터 255까지의 누적 합을 구함으로써 픽셀의 확률 밀도 함수를 구한다.

② 픽셀 레벨 개수의 0레벨부터 255레벨 까지 누적하여 적립 밀도 함수를 구한다.

③ 픽셀 레벨 누적 값을 정규화 하기 위하여, $\frac{Maxium\ Level\ Value}{Number\ of\ Pixel\ Image}$ 를 곱한다.

④ 픽셀 레벨 누적 값을 정규화하고, 영상을 한 픽셀씩 레벨에 맞는 정규화 값으로 치환한다.

4. 이진영상화/침식/확장

본 논문에서 사용된 이진영상화/침식/확장은 그레이 영상을 이진 영상으로 변환 한 뒤 침식과 확장을 사용하여 눈 후보 위치 검출을 하기 위한 전처리 과정으로 사용된다. 이진영상화는 특정한 임계치를 기준으로 그 값보다 크면 흰색, 작으면 검정색으로 바꾸는 간단한 방법을 사용하였다.

침식과 확장은 서로 반대되는 기능으로서 사용하는 순서에 따라 발생하는 현상이 다르게 나타난다. 본 얼굴 인식 알고리즘에서는 "침식 -> 확장"을 통하여 영상에 발생된 잡음을 제거하기 위해 사용하였다. 그림 3의 (b)는 (a)를 침식한 그림이며, (c)는 (b)를 확장한 그림이다.

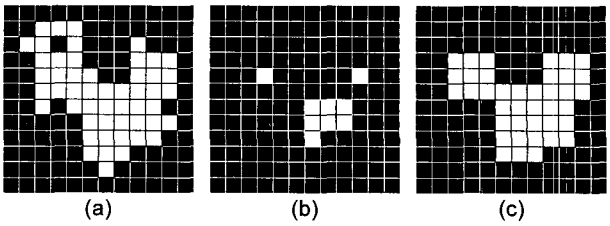


그림 3. 영상의 침식/확장
Fig. 3. Erode/Dilation.

침식 영상을 얻기 위해서는 수식 (3)와 같이 기준 위치의 픽셀 값과 주위 8개의 픽셀 값을 비교하여 최소값을 구하면 된다. 확대 영상은 반대로 수식 (4)과 같이 기준 위치의 픽셀 값과 주위 8개의 픽셀 값 중 최대값을 취하면 된다.

$$\begin{aligned} p'(x,y) = \min(&p(x-1,y-1), p(x,y-1), \\ &p(x+1,y-1), p(x-1,y), \\ &p(x,y), p(x+1,y), \\ &p(x-1,y+1), p(x,y+1), \\ &p(x+1,y+1)) \end{aligned} \quad (3)$$

$$\begin{aligned} p'(x,y) = \max(&p(x-1,y-1), p(x,y-1), \\ &p(x+1,y-1), p(x-1,y), \\ &p(x,y), p(x+1,y), \\ &p(x-1,y+1), p(x,y+1), \\ &p(x+1,y+1)) \end{aligned} \quad (4)$$

5. 커넥티드 컴포넌트

영상안에서 물체의 위치나 방향외에 각각의 물체를 구분하기 위하여, 픽셀들이 서로 연결되어 있는가 또는 떨어져 있는가를 판단해야 한다[4]. 이를 위하여 그림 4와 같이 화상 안에서의 각각 덩어리에 라벨을 붙이고 구분하게 된다.

커넥티드 컴포넌트를 구현하기 위해 먼저 라벨링(labeling)을 하게 된다. 라벨링은 첫 번째 픽셀부터 마

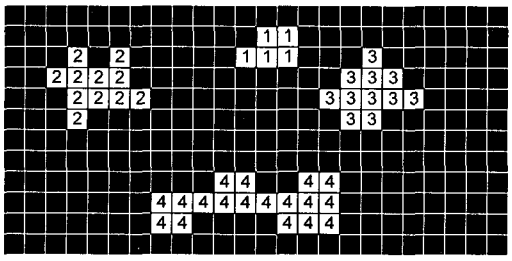


그림 4. 커넥티드 컴포넌트 라벨링
Fig. 4. Labeling of connected component.

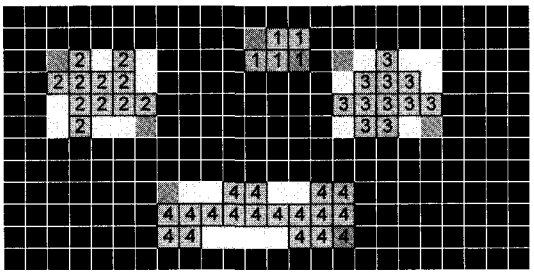


그림 5. 커넥티드 컴포넌트 위치 선정
Fig. 5. Center point of connected component.

지막 픽셀까지 검색해 나가면서 픽셀의 연결을 확인해 가며 라벨을 붙이게 된다. 라벨을 붙이는 방법에서도 상하좌우로 연결된 픽셀 외에 대각선으로 연결된 픽셀에 대해서 같은 물체인지 서로 다른 물체인지 판단하는 방법이 있다. 후자의 경우 물체를 더욱 잘게 찾을 수 있으므로 눈을 찾을 확률이 더욱 커지기 때문에 본 알고리즘에서는 대각선으로만 연결된 물체는 서로 다른 물체로 간주 하였다.

라벨링이 끝난 각각의 물체의 위치는 그림 5와 같이 검색된 물체의 외각으로 사각형을 그리고 사각형의 좌측상단, 우측하단의 좌표를 표시하는 방법이 물체의 중심을 가장 근접하게 나타내기 때문에 사용하였다.

6. 눈 검출

눈 검출을 하기 위해서 그레이 영상을 특정한 임계치를 기준으로 이진영상을 만들고, 침식, 확장을 통해 잡음을 없앤 뒤, 커넥티드 컴포넌트를 사용하여 눈 후보 위치를 얻어낸다. 임계치를 30에서 110까지 10씩 증가시키며 이진영상화, 침식, 확장과 커넥티드 컴포넌트를 거쳐 눈 후보 위치를 얻고, 이를 이용하여 눈 후보 위치의 쌍을 조합하게 된다.

눈 후보 위치의 쌍을 조합함에 있어서 여러 가지 조건을 갖게 되는데, 크기가 가로 15픽셀이상 또는 세로 10픽셀이상이면 눈 크기의 범주를 벗어나므로 눈 후보에서 제외하도록 하였다. 또한, 이전 임계치에서 얻어낸 눈 후보 위치와 진행 단계의 임계치에서 얻어낸 눈 후보 위치가 같으면 진행 단계의 눈 후보 위치를 제외한다.

눈 검출을 통하여 얻어낸 조합된 눈 후보 위치 쌍을 기준으로 얼굴 검증을 하여 나오는 값 중 가장 큰 값을 갖는 좌표를 입력된 영상의 눈 위치라고 판단하게 된다.

7. 어파인 트랜스폼

어파인 트랜스폼은 원영상의 좌표를 (x,y), 정규화영상의 좌표를 (x',y')라 표시할 때, 수식 (5)에서 a_{ij}의 해를 구함으로써 수식 (6)와 같이 원영상과 정규화영상과의 변환으로 얼굴 검증 처리에 필요한 정규화영상을 만들 때 사용된다.

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x'_1 & x'_2 & x'_3 \\ y'_1 & y'_2 & y'_3 \\ 1 & 1 & 1 \end{bmatrix} \tag{5}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \tag{6}$$

수식 (5)의 a_{ij}는 수식 (7)으로 나타낼 수 있고, 이때 (x_i,y_i)는 이미 알고 있는 원본 영상의 위치며, (x'_i,y'_i)는 정규화 영상의 정해진 위치이므로 역행렬 연산을 통해 계산되어 a_{ij}를 구할 수 있다. 수식 (8)의 (0, 0)은 정규화 영상의 좌측 상단의 모서리 위치이며, (5, 7), (19, 7)은 눈의 위치이다.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x'_1 & x'_2 & x'_3 \\ y'_1 & y'_2 & y'_3 \\ 1 & 1 & 1 \end{bmatrix}^{-1} \tag{7}$$

$$\begin{bmatrix} x'_1 & x'_2 & x'_3 \\ y'_1 & y'_2 & y'_3 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 5 & 19 \\ 0 & 7 & 7 \\ 1 & 1 & 1 \end{bmatrix} \tag{8}$$

$$\begin{bmatrix} x'_1 & x'_2 & x'_3 \\ y'_1 & y'_2 & y'_3 \\ 1 & 1 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 0 & -0.1428 & 1 \\ -0.0714 & 0.1938 & 0 \\ 0.0714 & -0.0510 & 0 \end{bmatrix} \tag{9}$$

그림 6에서 (x_2,y_2), (x_3,y_3)는 검출된 눈의 좌표이며, (x_1,y_1)는 이 눈 좌표에 의해서 구하여진다.

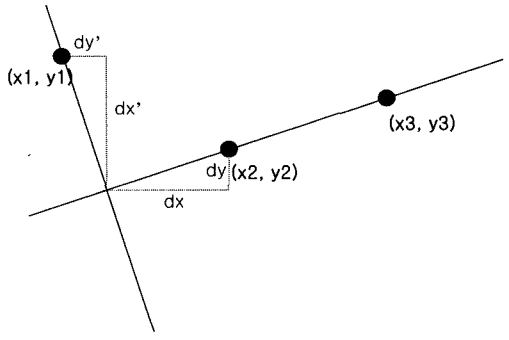


그림 6. 눈 위치를 통한 기준좌표 계산[3]
Fig. 6. Calculation of affine coordinate.

$$\begin{cases} dx = \frac{5}{14}(x_3 - x_2) \\ dy = \frac{5}{14}(y_3 - y_2) \end{cases} \quad (10)$$

$$\begin{cases} dx' = \frac{x_3 - x_2}{2} \\ dy' = \frac{y_3 - y_2}{2} \end{cases} \quad (11)$$

$$\begin{cases} x_1 = x_2 - (dx + dy') \\ y_1 = y_2 + (dy - dx') \end{cases} \quad (12)$$

8. 바이리니어 인터폴래이션

어파인 트랜스폼을 이용해서 정규화영상의 좌표 (x'_i, y'_i)에 대응하는 원영상의 좌표 (x_i, y_i)를 구하면 그림 7의 왼쪽 그림에서 검은 점과 같이 실수 값으로 나온다. 이때 수식 (13)와 같이 주변 4개의 픽셀과의 거리의 상관관계를 이용하여 (x_i, y_i)에 대응하는 픽셀 값을 구하는 것이 바이리니어 인터폴래이션이다.

$$\begin{aligned} f(x,y) = {} & g(l,k) + a(g(l+1,k) \\ & - g(l,k)) + b(g(l,k+1) \\ & - g(l,k)) + ab(g(l,k) \\ & - g(l+1,k+1) \\ & - g(l+1,k) - g(l,k+1)) \end{aligned} \quad (13)$$

where $l = floor(x), a = x - l, k = floor(y), b = y - k$

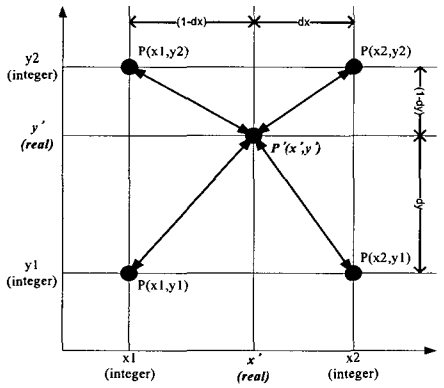


그림 7. 바이리니어 인터폴래이션
Fig. 7. Bilinear interpolation.

9. 얼굴 검증

얼굴 검증은 SVM 파라메터와 25×20픽셀의 정규화 영상의 행렬 곱에 가중치를 두어 누적 시킨 값이 임계치보다 큰지 비교하여 얼굴이 맞는지 검증하게 된다. 수식 (14)와 수식 (15)는 얼굴 검증에서 영상의 행렬 연산식을 나타낸 것이다.

α_i : 가중치
S_{ij} : Support Vector Machine Parameter
I_j : 정규화 된 영상(25×20픽셀)

$$I^n = \frac{I}{255} \quad (14)$$

$$\sum_i^{1669} \alpha_i^n [\sum_j^{500} S_{ij}^n I_j^n + 1]^2 > threshold \quad (15)$$

10. 영상 회전/확대/축소

얼굴 인식을 하기 위해서는 얼굴 영상을 정규화 된 영상으로 바꾸어 주어야 한다. 얼굴 검증을 통해 얻어진 눈의 위치를 이용하여 그림 8의 (a)와 같이 얼굴 인식에 필요한 영상의 영역을 정하고, 기울어진 얼굴 영상을 수평이 되도록 그림 8의 (b)와 같이 수식 (16), (17)을 통해 회전한다.

eye_w : 눈 위치의 x좌표의 거리
eye_h : 눈 위치의 y좌표의 거리

$$\theta = \tan^{-1}(\frac{eye_h}{eye_w}) \quad (16)$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad (17)$$

영상 회전을 통해 수평이 된 얼굴 영상은 확대나 축소를 하여 50×40픽셀의 영상으로 정규화 하여야 한다. 이 단계는 수식 (18)과 같이 얼굴 검증에서 사용된 어파인 트랜스폼의 영상을 같은 비율로 확대/축소하는 방법이 아닌 가로 세로 비율이 1:1인 정사각형의 영상을 5:4의 직사각형의 영상으로 변환하는 방법을 사용한다.

$width$: 원 영상의 가로 크기
$height$: 원 영상의 세로 크기

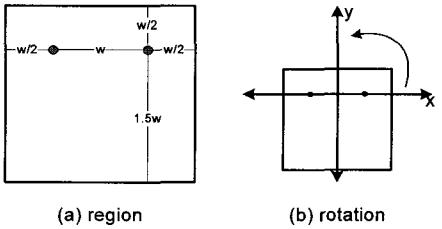


그림 8. 영상 회전
Fig. 8. Image rotation.

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \frac{width}{40} & 0 \\ 0 & \frac{height}{50} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad (18)$$

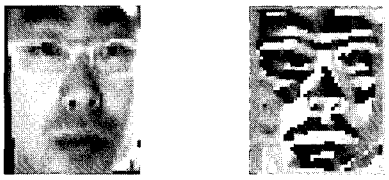
11. 그래디언트 앵글

그래디언트 앵글은 영상을 굴곡에 따라 입체적으로 표현하는 영상 처리 과정이다. 이 단계를 수행하면 그림 9의 (a)의 영상이 그림 9의 (b)처럼 굴곡을 표현하는 영상이 된다. 이는 전체적으로 조명의 변화가 있더라도 얼굴의 윤곽은 크게 변하지 않으므로, 상대적인 조명의 변화에 대해서 추출된 데이터의 신빙성을 보장하기 위함이다. 수식 (19), (20), (21)은 그래디언트 앵글의 수식을 나타내고 있다.

$$\begin{aligned} P(x,y) = & -I(x-1,y-1) - I(x-1,y) \\ & - I(x-1,y+1) + I(x+1,y-1) \\ & + I(x+1,y) + I(x+1,y+1) \end{aligned} \quad (19)$$

$$\begin{aligned} Q(x,y) = & +I(x-1,y-1) - I(x-1,y+1) \\ & + I(x,y-1) - I(x,y+1) \\ & + I(x+1,y-1) - I(x+1,y+1) \end{aligned} \quad (20)$$

$$GA(x,y) = \frac{255}{360}[180 + \tan^{-1}\frac{Q(x,y)}{P(x,y)}] \quad (21)$$



(a) normalized image (b) gradient angle

그림 9. 그래디언트 앵글 영상
Fig. 9. Gradient angle image.

12. 얼굴 인식

얼굴 인식은 수식 (22)와 같이 정규화된 얼굴 영상을 그래디언트 앵글을 수행하여 얻은 영상(1640)과 LFA 파라메터(1640×466)의 행렬 곱을 통하여 구한 얼굴 특징 벡터를 저장 하거나 이전에 저장된 얼굴 특징 벡터와의 거리 비교를 통해 수행 된다. 수식 (23)은 저장된 얼굴 특징 벡터와의 거리를 구하는 식이다. 거리가 가장 낮은 영상을 인식하게 된다.

FV_i : 얼굴 특징 벡터
SFV_i : 저장된 얼굴 특징 벡터
L_{ij} : Local Feature Analysis Parameter
I_i : 입력 영상
fr : 얼굴 특징 벡터의 거리

$$FV_i = \sum_{j}^{1640} L_{ij} I_i (0 \le i \le 466) \quad (22)$$

$$fr = \sum_{i}^{466} |FV_i - SFV_i| \quad (23)$$

Ⅲ. 고정 소수점 모델 구현 및 성능 분석

본 논문에서 사용한 얼굴 인식 알고리즘에서 부동소수점이 사용된 부분은 수식 (2)의 얼굴 검출의 특징 파라메터 중 가중치인 alpha 값과 임계치 threshold 값, 어파인 트랜스폼에서 수식 (9), (10), 얼굴 검증의 수식 (15)가 있으며, 수식 (16), (17)의 영상 회전과 수식 (19)의 그래디언트 앵글을 수행하기 위해 사용되는 삼각함수, 수식 (18)의 영상 확대/축소를 하기위한 나눗셈, 얼굴 확대/축소, 얼굴 인식을 위한 LFA 파라메터가 있다.

부동 소수점 모델과 고정 소수점 모델의 유사성은 포항공과대학교의 IMLAB에서 제공한 얼굴 영상을 기준으로 각 단계별로 고정소수점 사용 비트수에 따른 FRR 비교를 통해 측정 하였다. 여기서 사용된 FRR은 본 논문에서 사용된 얼굴 인식 알고리즘의 부동 소수점 모델과 고정 소수점 모델의 성능 차이를 알아보기 위한 지표로 사용하였다.

1. 얼굴 검출

얼굴 검출 단계에서 특징 파라메터의 조건식을 살펴보면 수식 (2)와 같다. 여기서 alpha는 0.05에서 1.7사이

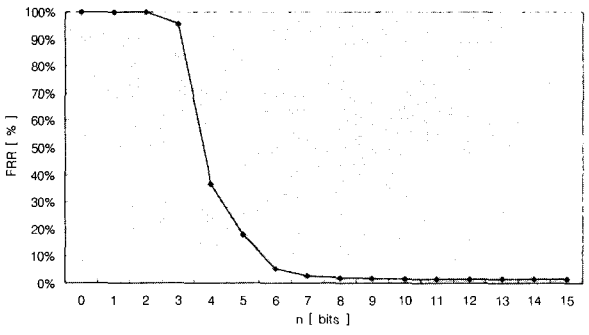


그림 10. 특징 파라메터의 소수점 사용 비트수(n)에 따른 얼굴 검출의 FRR
Fig. 10. Decimal fraction for feature parameter.

의 값이며, threshold는 1.6에서 16.0까지의 실수를 갖는다. 이 값은 이미 정해진 파라메터로서 소수점의 유효 자리를 변화시키며 얼굴 영상으로 FRR을 측정하여 그림 10에 나타내었다. 고정 소수점 모델이 alpha와 threshold 값을 소수점을 13비트 사용했을 때 부동 소수점 모델과 성능이 같아진다. 하지만 소수점 사용 비트수가 7비트 일 때 적은 자원으로 최대의 성능을 얻을 수 있었다.

2. 어파인 트랜스폼

얼굴 검증에 필요한 정규화 영상을 얻기 위해서 눈 검출을 통해 나온 눈 좌표를 기준으로 얼굴을 기울어지지 않은 영상으로 정규화 한다. 여기서 그림 6과 같이 검출 되어진 눈 좌표를 통해서 나머지 한 개의 좌표를 수식 (10), (11), (12)을 통해서 계산해 낸다.

어파인 트랜스폼에서 고정소수점 모델로의 변환을 위한 곳은 두 군데가 있다. 이중 한 부분인 수식 (10)에서 5/14의 소수점 사용 부분에 대해서 비트수를 변화시키며 FRR을 측정 하여 그 결과를 그림 11에 나타내

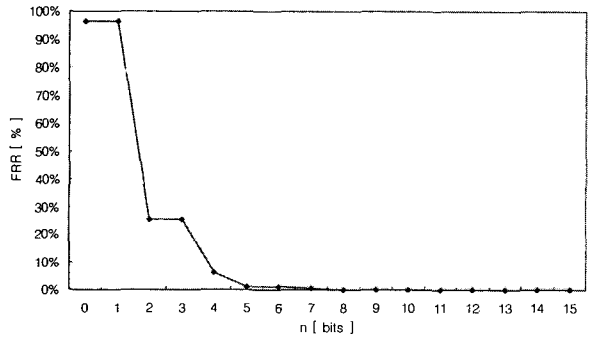


그림 11. 5/14의 소수점 사용 비트수(n)에 따른 어파인 트랜스폼의 FRR

Fig. 11. Decimal fration for 5/14.

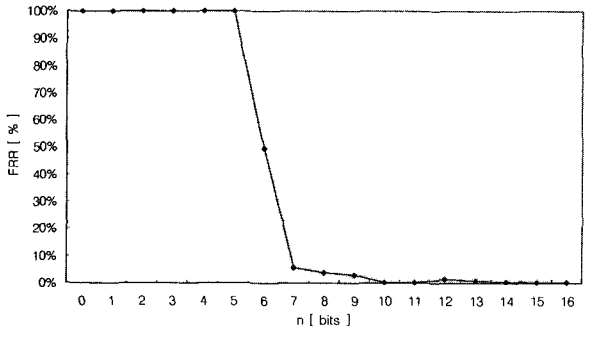


그림 12. 어파인 트랜스폼 계수의 소수점 사용 비트수(n)에 따른 FRR

Fig. 12. Decimal fraction for affine transform coefficient.

었다. FRR이 10비트이상부터 같아지는 것을 볼 수 있다. 그림 11에서 0과 1, 2와 3, 5와 6비트에서 FRR이 같게 나오는데 이는 수를 표현하는데 있어 세밀하게 표현하지 못하는 경우가 있기 때문이다. 결국, 0과 1, 2와 3, 5와 6비트를 사용하여 5/14를 표현한 값은 같다.

그림 12는 수식 (9)에서 계수 값의 소수점 사용 비트수를 바꾸어가며 FRR을 측정한 것이다. 소수점 사용 비트가 많아지면서 FRR이 대체적으로 높아지고 있으나, 안정적이지 못하다. 이것은 수식 (9)의 계수인 6개의 값이 서로 연관되어 어파인 트랜스폼 영상을 결정한다는 것과 그림 11에서 이야기 했던 수를 표현하는데 있어 세밀하게 표현하지 못한다는 것이 서로 작용하여 나타나는 현상이다.

3. 얼굴 검증

얼굴 검증에서 부동 소수점이 사용된 부분은 수식 (15)에서 a_i와 S_{ij}이다. 수식 (15)를 수식 (24)로 변환하기 위해 다음과 같은 실험을 하였다.

첫 번째, 하드웨어에서 255로 나누는 것 보다 256으로 나누는 것이 구조상 간단해 지기 때문에 255로 나누어 얼굴 검증 한 것과 256으로 나누어 얼굴 검증 한 것의 FRR을 측정하였다. 두 결과가 동일하여 두 모델은 동일한 성능을 갖는다는 것을 알 수 있었다.

두 번째, 제곱 안에 있는 '+1'의 존재이다. 이것은 없어도 성능의 변화에 미치지 않을 것 같다는 판단아래 FRR을 측정 하였다. 그 결과 FRR이 같게 나와서 '+1'의 존재는 성능에 영향을 미치지 않는 것을 알게 되었다.

세 번째, S_{ij}의 소수점 사용 비트수이다. S_{ij}는 얼굴 검증 알고리즘에서 사용되는 파라메터로서 0에서 1사이의 값이다 .그림 13에서 보는바와 같이 소수점을 7비트

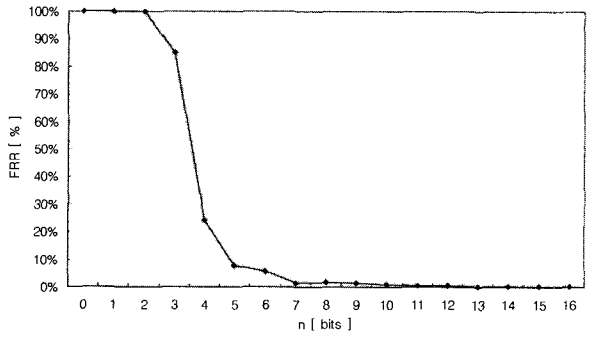


그림 13. S_{ij}의 소수점 사용 비트수(n)에 따른 FRR

Fig. 13. Decimal fraction for S_{ij}.

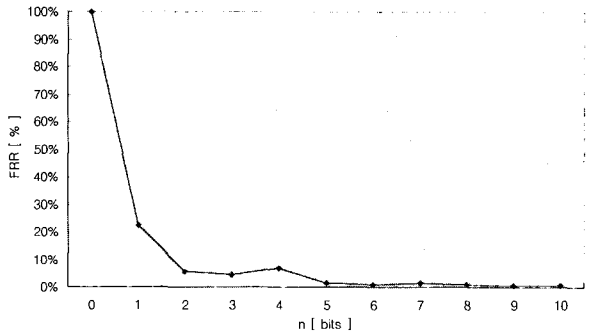


그림 14. a_i의 소수점 사용 비트수(n)에 따른 FRR
Fig. 14. Decimal fraction for a_i.

사용 할 때부터 부동 소수점 모델과 성능이 비슷해지는 것을 알 수 있다.

네 번째, a_i의 값이다. 이것은 -0.000037에서 0.000037까지의 수이다. 그림 14에서 5비트부터 FRR이 부동 소수점 모델과 비슷해지는 것을 볼 수 있다.

네 가지의 실험을 통하여 얻어진 결과를 바탕으로 수식 (15)를 수식 (24)로 변환하였다.

$$\sum_{i}^{1669}\alpha_i[\sum_{j}^{500}S_{ij}I_j]^2 > threshold \tag{24}$$

4. 영상 회전/확대/축소

얼굴 검증에서 얻어진 눈 좌표를 이용하여 얼굴 영상을 회전시키기 위해서는 수식 (16)과 수식 (17)을 사용하게 된다. 삼각함수는 임베디드 시스템이나 하드웨어에 적용시키기 어렵다. 그러므로 테이블 형식으로 메모리에 저장하여 사용하는 방식을 선택하였다.

eye_y/eye_w의 소수점 사용 비트수에 따른 부동 소수점 모델과 고정 소수점 모델과의 FRR을 그림 15에 나

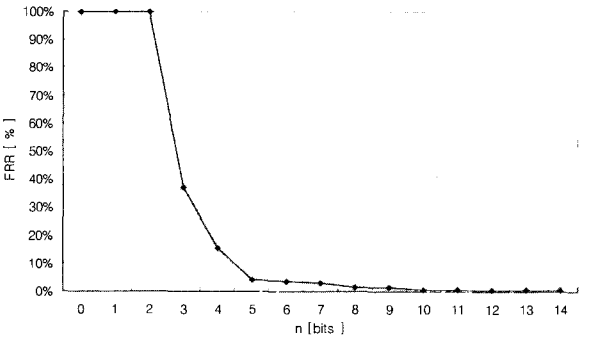


그림 15. eye_y/eye_w의 소수점 사용 비트수(n)에 따른 FRR
Fig. 15. Decimal fraction for eye_y/eye_w.

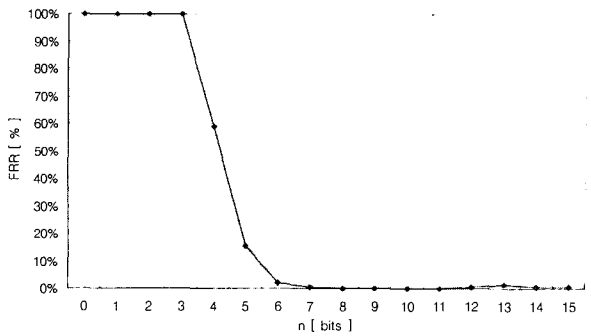


그림 16. $\cos\theta$, $\sin\theta$의 소수점 사용 비트수(n)에 따른 FRR
Fig. 16. Decimal fraction for $\cos\theta$, $\sin\theta$.

타내었다. 소수점을 10비트 사용했을 경우 부동 소수점 모델과 비슷해지는 것을 볼 수 있다.

$\cos\theta$, $\sin\theta$의 소수점 사용 비트수에 따른 부동 소수점 모델과 고정 소수점 모델과의 FRR을 그림 16에 나타내었다. 소수점을 8비트 사용했을 경우 부동 소수점 모델과 비슷해지는 것을 볼 수 있다.

영상 확대/축소를 하기위해 사용되는 수식 (18)과 같이 나눗셈이 포함되어 있다. 여기서 사용된 나눗셈 중 $width/40$은 수식 (25)로 $height/50$은 수식(26)으로 대체한 결과 고정 소수점 모델과 0.2%의 FRR의 차이를 보였다.

$$width \gg 6 + width \gg 8 \\ + width \gg 10 + width \gg 12 \tag{25}$$

$$height \gg 6 + height \gg 7 \\ + height \gg 10 + height \gg 11 \tag{26}$$

5. 그래디언트 앵글

수식 (21)에는 나눗셈과 아크탄젠트 함수가 사용된다. 이중 아크탄젠트 함수를 테이블 형태로 메모리에 저장하여 사용한다. 단순히 아크탄젠트 함수만을 테이블 형태로 저장하는 것이 아닌 그래디언트 앵글을 수행하는 연산을 모두 계산하여 테이블 형태로 저장한다. 수식 (21)에서 $GA(x,y)$는 픽셀 값으로서 0부터 255의 값을 갖는다. 이것을 이용하여 $GA(x,y)$을 0부터 255까지 변화 시키며 아크탄젠트의 값을 구하여 테이블을 구성 하였다. 이는 부동 소수점 모델과 FRR이 0.2%의 차이가 있었다.

6. 얼굴인식

얼굴 인식 단계에서 부동 소수점이 사용된 곳은 수식

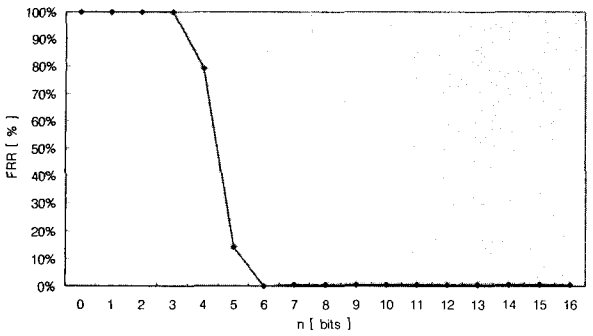


그림 17. LFA 파라메터의 소수점 사용 비트수(n)에 따른 FRR

Fig. 17. Decimal fraction for LFA parameter.

(22)의 LFA 파라메터이다. LFA 파라메터는 -0.2에서 0.2까지의 실수로서 그림 17에서 보는 바와 같이 소수점을 6비트 사용하였을 때 부동 소수점 모델과 성능이 비슷해진다.

Ⅳ. 하드웨어 설계 구조

Ⅲ장에서 얻어낸 본 얼굴 인식 알고리즘의 고정 소수점 모델을 바탕으로 얼굴 검출, 얼굴 검증 그리고 얼굴 인식의 하드웨어 설계를 위한 구조를 제안 하였다.

1. 얼굴 검출

그림 18은 카메라에서 입력된 160×120픽셀의 그레이 영상이 저장된 Image buffer로부터 얼굴의 위치를 찾아 출력하는 과정을 하드웨어로 설계하기 위한 블록 다이어그램이다. 이 구조는 입력영상으로부터 얼굴의 좌표가 추출되기까지의 전체 알고리즘을 하드웨어에서 구현이 편리하도록 각 기능별로 Integral image calculator, Feature coordinate calculator, Feature difference calculator, Cascade calculator, Window detector등의 5개 모듈로 분류하였다[11].

2. 얼굴 검증

얼굴 검출의 출력으로 30비트의 얼굴 좌표를 얻게 되면 Eye detection에서 여러 개의 눈 위치 후보 쌍을 찾게 된다. 각각의 눈 위치 후보 쌍으로 Affine transform을 수행하여 얼굴 영상을 얼굴 검증의 정규화 영상으로 변환한다. 얼굴 검증을 하기 위한 정규화 영상과 SVM 파라메터의 행렬 곱을 통하여 신뢰값을 구한 후 특정 임계치보다 높으면 얼굴 영상임을 확신하며, Eye detection에서 얻은 눈 위치 후보 쌍 중 가장 높은 신뢰

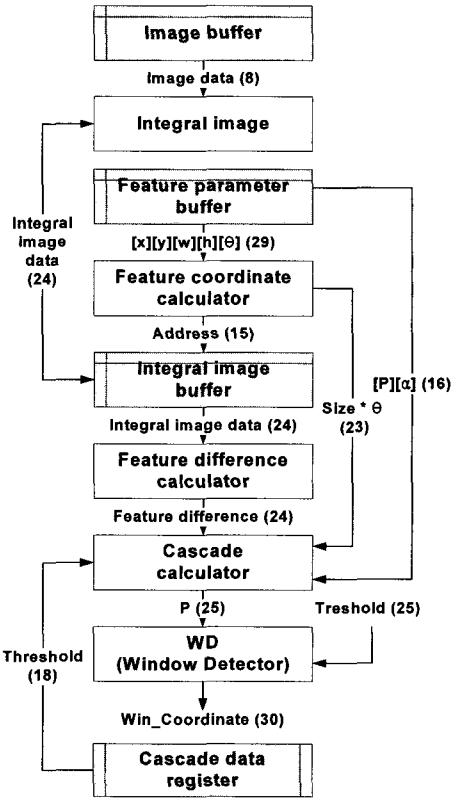


그림 18. 얼굴 검출 블록 다이어그램[11]

Fig. 18. Face detection block diagram.

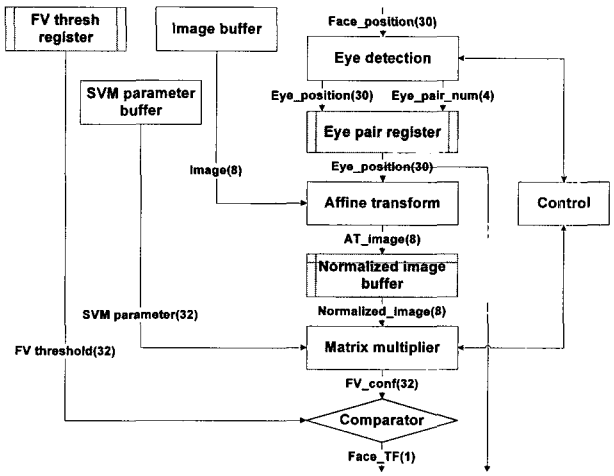


그림 19. 얼굴 검증 블록 다이어그램

Fig. 19. Face verification block diagram.

값이 구해진 눈 후보 쌍의 위치가 눈 위치로 된다. 그림 19에서 이러한 구조를 나타내고 있다.

3. 얼굴 인식

얼굴 검증단계가 수행되고 얼굴을 확신하면, 눈의 위치가 구해진다. 이 눈 위치를 기준으로 얼굴 인식을 하기 위한 얼굴 영역을 정한다. 이 영역은 눈 위치를 기준으로 영상 회전/확대/축소를 수행하여 40×50픽셀의 영

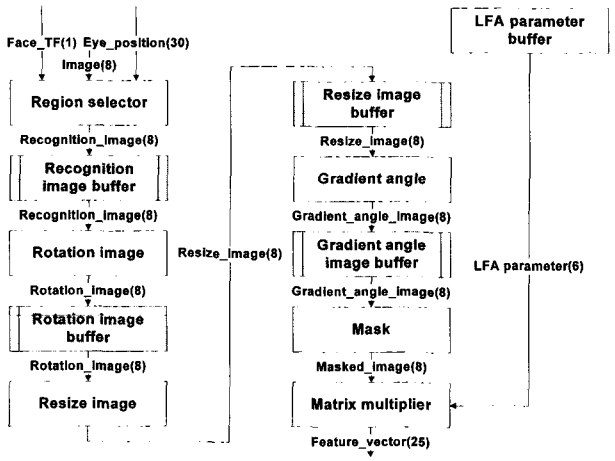


그림 20. 얼굴 인식 블록 다이어그램
Fig. 20. Face recognition block diagram.

상으로 정규화 한 후, Gradient angle이 수행된다. 마지막으로 40×50픽셀의 사각형 영상인 그래디언트 앵글 영상은 Mask를 수행하여 얼굴 모양과 같은 계란형 영상으로 변환된다. 이 영상과 LFA 파라메터의 행렬 곱연산을 통하여 466개의 Feature vector를 레지스터에 저장하게 된다. 그림 20에서는 눈 위치를 이용하여 얼굴 인식의 얼굴 특징 벡터인 Feature vector를 출력하는 구조를 나타내고 있다.

V. 결 론

본 논문에서는 얼굴 인식 알고리즘을 하드웨어로 구현하기위해 고정 소수점 모델을 구현하여 성능을 분석하였으며, 현재 본 얼굴 인식 알고리즘의 고정 소수점 모델을 사용하여 FPGA로 검증 중에 있으며, 얼굴 검출 단계는 FPGA와 ASIC으로 구현, 검증되어 얼굴 검출 성능이 고정 소수점 모델과 같음을 확인하였다.

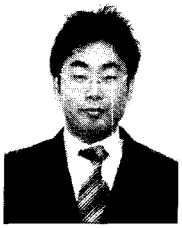
얼굴 검출단계에서 수식 (21)의 alpha 값과 threshold 값의 적절한 소수점 사용 비트 수는 그림 10을 통하여 선택 할 수 있으며, 어파인 트랜스폼에서는 수식 (10)과 수식 (5)를 위한 소수점 사용 비트수를 그림 11, 12로부터 얻을 수 있었다. 또한 얼굴 검증은 수식 (15)를 수식 (22)로 변환 한 뒤, 그림 13, 14를 참조하여 a_i, S_{ij}의 비트수를 선택하면 부동 소수점 모델의 성능을 크게 떨어뜨리지 않고 고정 소수점 모델로 변환할 수 있다. 또한, 수식 (16)의 eye_h/eye_w, 수식 (17)의 $cos\theta$, $sin\theta$의 소수점 사용 비트수를 그림 15, 16에서 선택할 수 있었으며, 수식 (18)의 나눗셈은 수식 (26), (27)과 같이 구조를 변경하였다. 그래디언트 앵글의 수식 (21)은 $GA(x,y)$의 변화에 기준을 두고 256개의 데이터로 이루어진 테이블을 사용하였다. 마지막으로 얼굴 인증에 필요한 LFA 파라메터는 그림 17에서 선택할 수 있었다.

지금 까지 했던 실험을 토대로 고정 소수점 모델을 구현하여 FRR을 측정하였다. 그 결과 부동 소수점 모델의 성능과 0.22% 차이가 나는 고정 소수점 모델과 그림 22, 23, 24와 같은 얼굴 인식 하드웨어 구조로 설계를 하고 있다.

참 고 문 헌

[1] P. Viola and M. Jones, "Robust Real-time Object Detection," Second International Workship on Statistical and Computational Theories of Vision-Modeling, Learning, Computing, and Sampling. Vancouver, Canada, July 13, 2001.
[2] M. A. Hearst, "Support Machine Vector," IEEE Intelligent Systems Magazine, pp. 18-28, July 1998.
[3] http://iria.math.pku.edu.cn/~jiangm/courses/dip/htm l/node67.html
[4] K. Jong-Hwan, 비전시스템, Intelligent Control Lab, KAIST.
[5] 김문갑, "Affine 변환과 템플릿 정합을 이용한 크기 및 회전 변화 얼굴 인식," 한국통신학회논문지 Vol.26 No.6T, 2001.
[6] L. Ji-Bum, L. Ho-Joon and K. Hyung-Hwa, "기울기 검출에 의한 얼굴영상의 인식의 개선에 관한 연구," 한국통신학회논문지 Vol.18 No.7, 1993.
[7] 정조남, 이필규, "얼굴 인식을 위한 효과적인 눈 위치 추출," 정보처리학회논문지 제12-B권 제2호 2005.
[8] http://bmrc.berkeley.edu/courseware/cs294/fall97/assignment/psnr.html
[9] 김대진, 전봉진, "SoC 칩 구현을 위한 얼굴 인식," 대한전자공학회지, 1016-9288 , 제33권 1호 , pp.56-63 , 2006.
[10] P. S. Penev and J. J. Atick, "Local feature analysis : a general statistical theory for object representation," Computation in Neural System 7, 477-500, 1996.
[11] 이수현, 정성윤, 정용진, "얼굴 검출을 위한 SoC 하드웨어 구현 및 검증," 대한전자공학회 SoC 학술 대회, 2006.

저 자 소 개



김 영 진(학생회원)

2005년 대진대학교 통신공학과 학사 졸업.

2005년 3월~현재 광운대학교 전자통신공학과 석사과정

<주관심분야 : 디지털 SoC, 영상처리, 얼굴인식, 생체인식>

정 용 진(정회원)

1983년 서울대학교 제어계측 공학과 학사 졸업.

1983년 3월~1989년 8월 한국전자통신연구원

1995년 2월 미국 UMASS 전자전산공학과 박사

1995년 4월~1999년 2월 삼성 전자 반도체 수석 연구원

1999년 3월 광운대학교 전자공학부 부교수

<주관심분야 : 무선 통신, 정보보호, SoC 설계, 영상처리 및 인식, 임베디드 시스템>