임베디드 시스템을 위한 영상객체의 검출방법

Image Objects Detection Method for the Embedded System

김 연 일, 노 승 용*
(Yun-Il Kim and Seung-Ryong Rho)

Abstract : In this paper, image detection and recognition algorithms are studied with respect to embedded carrier system. There are many suggested techniques to detect and recognize objects. But they have the propensity to need much calculation for high hit rate. Advanced and modified method needs to study for embedded systems that low power consumption and real time response are requested. The proposed methods were implemented using Intel® Open Source Computer Vision Library provided by Intel Corporation. And they run and tested on embedded system using a ARM920T processor by cross-compiling. They showed 1.6sec response time and 95% hit rate and supported the automated moving carrier system smoothly.

Keywords : object detection, image recognition, embedded system, template matching, CBCH (Cascade of Boosted Classifier using Haar-like features)

I. 서론

영상 객체의 인식 기술은 지난 수년 간 많은 진보를 하였으나, 단순한 운반 시스템의 응용에 대해서 아직도 지능형 임베디드를 위한 소규모의 하드웨어 구성과 다양하게 변화하는 환경의 적응성에서는 문제점을 갖고 있다. 따라서 강인하고 빠른 3차원 환경의 감지, 모델링, 검출과 인식을 위한 비전 시스템의 연구는 현재 활발히 진행되고 있다. 그러나 간단한 운반 시스템에서는 영상처리에 필요한 막대한 컴퓨터 비용 때문에 실시간 지능형 시스템의 구현에 많은 어려움이 따른다. 특히 객체 검출이나 추적에 많은 연산을 필요로 하는 알고리즘을 적용하면 검출율에서는 뛰어난 성능을 보이지만 실시간 객체 추적에는 부적합하며, 실용화하기 위해서는 많은 컴퓨팅 파워가 필요하다. 기존의 객체 추출과 인식의 기법에서 우수한 성능을 보이는 알고리즘들은 상대적으로 처리 속도가 떨어진다. 또한, 반대로 처리 속도가 빨라 많은 프레임을 처리할 수 있는 알고리즘은 객체 검출과 인식률이 낮아진다. 보통 객체 추적에 있어서 정지 영상이나 동영상에서 사람이나 사물과 같은 객체를 찾아내어 인식하고 판별하는 과정을 자동화시키기 위해서는 먼저 입력 영상에서 찾고자 하는 객체가 어디에 위치하는지를 알아내기 위하여 객체의 검출이 필요하다[1,2]. 모든 객체검출 기법은 회전과 크기(Size)라는 두 가지 문제를 가지고 있다. 회전은 두 종류가 있는데 시계방향, 반시계 방향으로 돌아가는 것을 화면 내-회전(in-plane rotation)이라 하고 그 외의 회전을 화면 외-회전(out-of-plane rotation)이라 한다. 크기는 비전의 아직 안 풀린 문제 중 하나인데 같은 모양이라도 화면에서 나타나는 크기가 모두 다를 수 있으며, 이것을 어떻게 처리하느냐가 객체 검출의 주요 연구 대상이다. 본 논문에서는 기억 용량과 연산 능력이 제한된 임베디드 시스템에서 최소의 연산으로 객체를 검출하고 인식할 수 있도록 CBCH 기법을 통해 객체의 위치와 크기를 알아내고, 다시 그의 정보를 템플릿 매칭에 의해 유사도를 비교하여 객체를 판별하는 방법을 제안한다. 따라서 주행하는 운반 시스템에서 실시간으로 영상 객체의 검출과 인식이 용이하도록 삼성의 ARM 프로세서인 S3C2440 프로세서가 장착된 임베디드 보드에 표준형 Qplus를 porting하여 실험한다. 또한 타깃 보드인 SMDK-2440 보드에서 제안된 알고리즘과 기존의 알고리즘들에 대하여 검출율과 실시간 처리성능을 비교 분석한다.

* 책임저자(Corresponding Author)
논문접수 : 2009. 2. 27., 채택확정 : 2009. 3. 16.
김연일, 노승용 : 서울시립대학교 전자전기컴퓨터 공학부
(kimyunil@gmail.com/syrho@uos.ac.kr)
※ 본 논문은 2005년도 서울시립대학교 학술연구조성비에 의하여 연구되었으며, 이에 감사드립니다.

II. 영상 객체의 검출과 인식

1. 템플릿 매칭을 이용한 객체 인식

입력 영상에서 찾고자 하는 객체를 탐색하는 방법 중 가장 단순한 방법은 템플릿 매칭(template matching)기법 이다. 이 방법은 객체 검출 방법론 중에서 영상에 기초한 접근 방법으로 추적하는 객체를 템플릿으로 정의하고 목표 영상에서 템플릿의 위치를 찾는다. 즉, 그림 1에서와 같이 영상에서 찾으려는 객체를 템플릿으로 정의하고 영상 내에서의 위치를 찾으려고 할 때에 사용하는 방법이다[3]. 템플릿에서는 입력 영상의 전 영역 혹은 객체가 있을 것이라 추측되는 관심 영역(region of interest)을 설정하고 그 영역에 대해서 이동하면서 템플릿 영상과 원 영상을 비교한다. 비교 방법은 주로 두 영상의 유사도를 계산하는 상관 계수(correlation coefficient)를 주로 사용한다[6]. 즉, 유사도가 높을수록 객체일 확률이 높다는 것을 의미하며, 이것을 이용해서 객체의 위치를 검출(object detection)한다.

템플릿 매칭 기법을 이용하여 영상 비교를 하기위해서는, 비교하려는 템플릿의 크기가 영상 혹은 영상의 관심 영역의 크기보다 작거나 같아야 한다. 즉, m x n 픽셀의 입력 영상에 대해서 p x q 픽셀의 객체를 검출하려면, $m \geq p$, $n \geq q$ 이어야 한다. 한 번의 템플릿 매칭으로 하나의 유사도 값이 나오게 된다. 따라서 최종적으로 $(m-p+1) \times (n-q+1)$

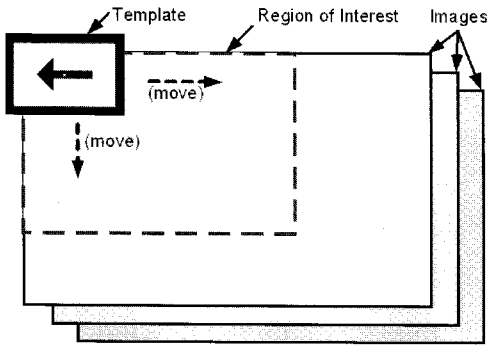


그림 1. 템플릿 매칭.

Fig. 1. Template matching.

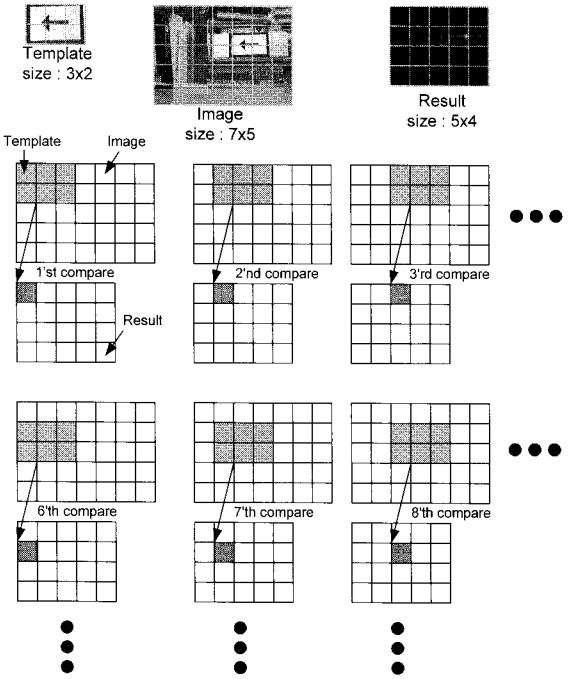


그림 2. 템플릿 비교 과정.

Fig. 2. Procedure of template matching.

크기의 유사도 배열이 나오게 된다. 그림 2에서는 7 x 5 픽셀의 영상에서 3 x 2 픽셀의 템플릿을 검출하는 과정을 보였고 5 x 4의 유사도 배열 결과를 나타내는 예를 보여주고 있다. 템플릿을 목표 영상의 전 영역에 대해서 이동하면서 템플릿 영상과 비교를 한다. 또한 이 영상 검출에서는 총 20번의 템플릿 매칭이 이루어지며, 유사도 배열은 시각화를 위해서 비교 값을 0에서 255 사이의 값으로 다시 스케일링 하여 흑백 영상화하였다. 따라서 가장 밝게 나타나는 부분이 찾고자하는 객체의 좌 상단 위치가 된다.

2. 유사도 측정 방법에 따른 성능 비교

템플릿 매칭을 이용한 객체 검출에서 성능을 평가할 때 주의할 부분은 바로 유사도 측정 방법이다. 템플릿 매칭은 전체 영상에 대해서 비교 연산이 반복적으로 이루어지므로 어떤 비교 방법을 사용하는지가 성능을 좌우 한다. 대표적인 유사도 측정 방법으로는 유클리디언(euclidean)거리를 이용하는 방법, 상관도(correlation)를 이용하는 방법 그리고 상관 계수(correlation coefficient)를 이용하는 방법 등이 있다 [6]. 절대적인 비교 값 보다는 다른 위치에서의 값과 비교되는 상대적인 유사도가 중요하므로 정규화된(normalized) 유사도를 이용하여 처리하게 된다. 따라서 유사도는 0과 1 사이의 값을 가지게 된다. 1에 근사한 값일수록 템플릿 영상과 유사도가 높고 찾고자 하는 객체 영상일 확률이 높은 것이다. 반대로 0에 가까울수록 상관성이 없는 것이다.

유클리디언 방법은 식 (1)과 같이 표현되며, 템플릿 영상 $T(x,y)$와 원영상 $I(x,y)$과의 차이값을 제곱하여 합한 값이 결과 $R(x,y)$가 된다.

$$R(x,y)=\sum_{x',y'}^{X,Y}[T(x',y')-I(x+x',y+y')]^2 \tag{1}$$

따라서 식 (1)의 정규화 표현식은 식 (2)이다.

$$R(x,y)=\sum_{x',y'}^{x,y}[T(x',y')-I(x+x',y+y')]^2/\sqrt{\sum T(x',y')^2\bullet\sum I(x+x',y+y')^2} \tag{2}$$

유사도를 구하는 다른 방법으로 상관도(correlation)를 이용하는 방인으로 두 신호의 상관 함수를 이용하여 유사도를 구하게 된다. 표현식은 아래 식 (3)과 같다.

$$R(x,y)=\sum_{x',y'}^{x,y}[T(x',y')\bullet I(x+x',y+y')]^2 \tag{3}$$

마찬가지로 다른 위치에서의 상관도와의 상대 비교를 위해 그 값을 0과 1 사이의 값으로 정규화하면 식 (4)와 같다.

$$R(x,y)=\sum_{x',y'}^{x,y}[T(x',y')\bullet I(x+x',y+y')]^2/\sqrt{\sum T(x',y')^2\bullet\sum I(x+x',y+y')^2} \tag{4}$$

마지막으로 가장 널리 사용되는 방법이 상과 계수(correlation coefficient)를 이용하는 방법이다.

$$R(x,y)=\sum_{x',y'}^{x,y}[T'(x',y')\bullet I'(x+x',y+y')]^2 \tag{5}$$

여기서 $T'(x',y')$는

$$=T(x',y')-1/(w\bullet h)\bullet\sum_{x'',y''}T(x'',y'')I'(x+x',y+y')$$
$$=I(x+x',y+y')-1/(w\bullet h)\bullet\sum_{x'',y''}x,yI(x+x'',y+y'')$$

이다. 정규화 된 상관 계수 비교 방법은 식 (6)과 같다.

$$R(x,y)=\sum_{x',y'}^{x,y}[T'(x',y')\bullet I'(x+x',y+y')]^2/\sqrt{\sum T'(x',y')^2\bullet\sum I'(x+x',y+y')^2} \tag{6}$$

여기서 $T'(x',y')$는

$$=T(x',y')-1/(w\bullet h)\bullet\sum_{x'',y''}T(x'',y'')I'(x+x',y+y')$$
$$=I(x+x',y+y')-1/(w\bullet h)\bullet\sum_{x'',y''}x,yI(x+x'',y+y'')$$

이다. 이미 설명한 바와 같이, 템플릿 매칭은 영상에서 물체의 위치를 찾을 때 사용하는 가장 기본적인 방법이다. 템플릿 매칭은 구현이 간단하고 비교적 정확한 유사도를 얻을 수 있다는 장점이 있다. 그러나 객체의 크기 변화와 회전, 빛의 밝기에 민감하게 반응하는 등의 단점이 있다. 따라서 이러한 단점을 극복하기 위해서는 multi-resolution, multi-scale 또는 sub-templates 및 deformable templates와 같은 방법들을 적용할 수 있다[1].

3. CBCH 기법을 이용한 객체 검출

객체 검출은 영상처리 중 패턴 인식 분야에 해당하며 영상의 모든 영역을 비교하여 객체인지 아닌지를 판단하는 방법이다. 물체를 찾는 간단한 방법은 템플릿 매칭이다. 템플릿 매칭에서는 유사도를 계산하는 수식을 통해 영상을 구분하지만, 패턴 인식에서는 영상의 종류를 판별하는 분류기(classifier)를 학습시켜서 원하는 패턴인지 아닌지 구별한다. CBCH(cascade of boosted classifiers working with haar-like features)를 이용한 검출 방법의 특징은 분류기의 연산이 단순하다는 것이다. 즉, Haar 분류기는 검정색 영역의 합과 흰색 영역의 합의 차이를 특징 값으로 하며 이 값을 비교하여 객체 영상인지 아닌지를 판별한다. 따라서 연산이 단순한 만큼 사물의 검출율은 높지 않다. 그러나 Haar 분류기를 100개 혹은 1000개 이상을 적절히 조합함으로써 분류기의 성능을 높일 수 있다는 것이 CBCH의 핵심이다[4]. 또한, 이 Haar 분류기의 특징 값은 누적된 밝기 값을 갖는 영상(integral image)을 이용하기 때문에 검정색 영역의 합과 흰색 영역의 합의 차는 몇 번의 덧셈 뺄셈으로 이뤄질 수 있어서 다른 패턴 인식 방법에 비해 계산 속도가 매우 빠르다. 종래에 사물의 검출에 적절한 Haar 분류기의 조합을 찾기 위해서는 아다부스트(Adaboost)라는 알고리즘을 사용하였다. CBCH의 Boosted는 바로 아다부스트 알고리즘과 같이 여러 개의 분류기를 조합해서 사용하는 부스트(Boost) 알고리즘을 사용하기 때문이다[5]. 아다부스트는 사물의 영상에서 가능한 모든 형태의 Haar 분류기에 대해서 사물의 판별능력이 뛰어난 순서대로 Haar 분류기를 추출해준다. 각각의 Haar 분류기의 성능은 다르기 때문에 가중치도 함께 계산되어 나온다. 그리고 추출된 분류기들은 임의의 영상에 대해서 각각의 사물 영역인지 아닌지를 판별하게 되고 다수결에 따라서 사물의 영상인지 아닌지 판별되게 된다. 예를 들면, 100개의 분류기가 있다면, 임의의 영상에 대해서 60개의 분류기가 사물의 대상이라고 판단하고 40개가 사물이 아니라고 판단하면 다수결에 따라 사물이 아니라고 판단하게 된다. 여기서 적절한 분류기의 조합을 찾기 위해서 적게는 수 백 장에서 많게는 수 천 장의 테스트 대상의 사물 영상이 필요하며, 이런 과정을 바로 분류기를 학습(learning 혹은 training)시킨다고 한다[7].

1000개의 Haar 분류기를 사용한다고 했을 때도 한 번에 1000개 모두를 비교하는 것이 아니라 처음엔 1개, 그 다음 10개, 25개, 25개 50개 등 그 개수를 증가시키면서 차례로 비교하는 방법을 사용한다. 이렇게 단계를 나누어서 비교하는 것을 cascade 라고 하며 사물의 속도를 가속화시킬 수 있다. CBCH 함수를 사용하기 위해서는 CBCH의 객체 검출

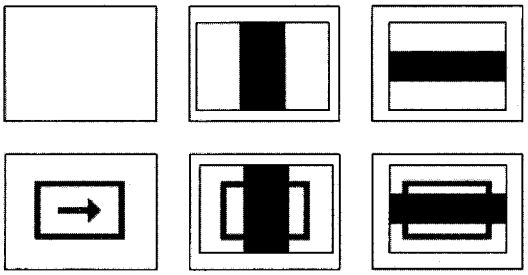


그림 3. Haar 특징 분석에 대한 예.
Fig. 3. The example of Haar features analysis.

방법에 대하여 고찰할 필요가 있다. CBCH와 같은 객체 검출 방법은 그림 3에서 보이는 사각형 영역의 크기를 변화시키면서 영상의 전 영역에 대해서 객체인지 아닌지 검출한다. 이 사각형 영역을 search window라고 한다. search window의 크기가 변화하기 때문에 작은 객체부터 큰 개체까지 검출할 수 있으며, 영상의 전 영역을 검사하기 때문에 객체가 어느 위치에 있어도 검출이 가능하다. Scale factor는 이 search window의 확대 비율이며, 1.1이라면 search window를 1.1배씩 확대시키면서 객체를 검출한다. 물론 함수 내에서는 search window 대신 영상의 크기를 변화시키는 기술을 다루어 연산량을 줄이게 한다.

4. 임베디드 시스템을 위한 객체의 인식과 검출

영상에서 객체를 찾아 판별하는 패턴 인식에는 많은 알고리즘들이 있으나, 인식률이 높으면 연산량이 많아지는 경향이 있다. 저전력과 실시간 응답특성이 요구되는 임베디드 시스템을 위해서는 그 특성에 맞게 수정 보완된 알고리즘 개발이 필요하다. 임베디드 시스템에 적용되어 사용될 수 있는 적은 연산량으로 정확한 객체 인식이 가능한 고속의 객체 인식이 가능한 알고리즘을 제안하고자 한다. 영상에서 객체를 검출하는 방법은 템플릿 매칭과 같은 단순한 기법이 있지만, 크기 변화와 배경 영상 노이즈에 취약하다. 또, 한 영상에서 둘 이상의 객체 검출이 불가능하고 연산량이 많다는 단점이 있다. 특히 연산량이 많다는 것은 곧 많은 컴퓨팅 파워를 소모하여 저전력 설계의 장애 요인이 될 수 있다. 이를 극복하기 위해서 본 논문에서는 특징 해석 기법 중의 하나인 CBCH알고리즘[5]을 사용하였다. CBCH기법은 정확한 객체 검출과 적은 연산량을 특징으로 한다. 그러나 이 방법은 확률적 접근 방법이기 때문에 정확한 객체 인식은 불가능하다. 따라서 이를 보완하기 위해 검출된 객체 영상을 다시 템플릿 매칭기법을 적용하여 객체 인식을 가능하게 하였다. 이를 통해 영상에서 적은 연산량으로 정확한 객체의 검출과 인식이 가능하게 하였다. 입력 영상으로부터 들어오는 객체의 위치와 크기는 랜덤 확률적이어서 예측할 수 없다. 이미 설명된 CBCH 기법을 통해 다양한 위치, 크기와 형태를 가지는 객체를 검출 할 수 있다. 또한 템플릿 매칭을 통해서 영상의 유사도를 구할 수 있다. 본 연구에서 제안 하는 방법은 그림 4(b)와 같은 과정을 거치게 되는데 CBCH 기법을 통해 객체의 위치와 크기를 알아내고 다시 그 정보를 받아 객체를 판별하기 위해서 템플릿 매칭을 사용하여 유사도를 비교한다.

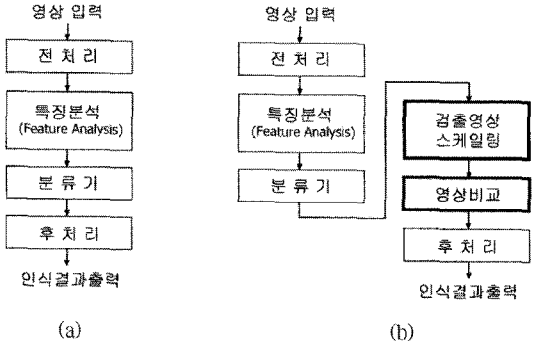


그림 4. (a) CBCH 객체인식 과정과 (b) 제안된 알고리즘을 이용한 객체인식의 과정의 흐름도.

Fig. 4. The flow of CBCH object recognition (a) and suggested object recognition (b).

따라서 최종적으로 나온 유사도를 비교하여 객체 인식을 한다. CBCH 를 통해서 검출된 영상은 다양한 크기를 갖는다. 크기가 다양한 것은 영상 객체들 자체의 크기가 다를 경우도 있고, 관측되는 거리에 따라서 크기가 달라지기도 한다. 이러한 영상들에 대해서 유사도를 구하기 위해서는 입력 영상을 비교 영상의 크기로 확대 혹은 축소하는 과정이 필요하다. 영상의 확대는 과-표본화(over-sampling)라 할 수 있고 축소는 저-표본화(under-sampling)라고 할 수 있다. 이를 보간법(interpolation)과 선택법(decimation)이라고 하기도 하는데 이는 보다 일반적인 크기로의 변화를 위해 넓은 의미로 보간법으로 간주하기도 한다[8].

III. 실험 및 고찰

본 논문의 실험은 고정된 카메라를 장착한 임베디드 시스템에서 배경 영상의 변화를 일정하게 유지한 환경으로 제약하고, 입력 영상을 실시간으로 받아들여 객체의 위치를 검출하고, 인식하는 것을 실험하였다. 본 연구에서 제안한 방법의 구현을 위해 Intel 사에서 오픈 소스 형식으로 제공하는 영상 처리 라이브러리[9]인 OpenCV b5(open source computer vision library beta 5)를 사용하였으며, 영상 실험은 임베디드 시스템과의 교차 개발 환경을 고려하여 Linux Fedora Core 4 환경에서 GNU C++ Compiler를 가지고 구현을 하였다. 또한 windows GUI 환경에서도 결과물이 동작되고, 실험결과들을 확인 할 수 있도록 구현하였다. 실험에 사용 된 호스트 컴퓨터는 Intel Dual Core 2.66 GHz, 2GB RAM에서 Microsoft Windows XP SP4 운영체제하에서 운영되었다. 일반적인 USB PC 카메라로부터 받아들이는 배경 영상과 입력 영상의 크기는 QVGA 표준을 맞추기 위해 320 X 240 pixel의 24bit 의 컬러 영상을 이용하였다. 초기 과정에서는 알고리즘 동작의 구현과 확인을 위해서 정지 영상만을 가지고 실험하였으며, 구현을 마치고 성능 평가부분에서 성능 측정을 위해서 실시간으로 들어오는 동영상 데이터에서 프레임을 capture하여 처리 할 수 있도록 하였다. 알고리즘을 테스트하기 위해서 범용적인 목적을 갖는 삼성 제품 ARM920T프로세서인 S3C2440으로 동작하는 SMDK2440보드 상에서 동작을 검증하였다. 그림 5는 본 연구

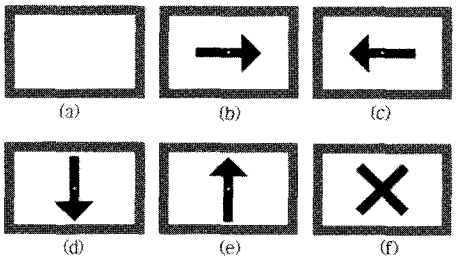


그림 5. 영상 객체 (a) 방향표시 공간, (b) 우 방향, (c) 좌방향, (d) 후 방향, (e) 직진, (f) 일시 정지.

Fig. 5. Image object (a) Flag, (b) Right direction, (c) Left direction, (d) Backward, (e) Forward, (f) Stop.

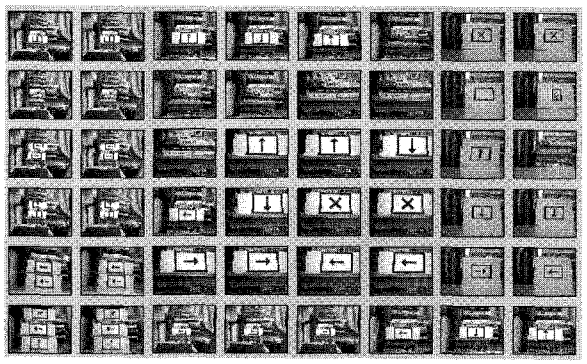


그림 6. 학습 데이터에 사용된 영상들.

Fig. 6. Example of flag images used for training.

의 실험에서 객체로 정의되어 사용되는 방향 표식들이다.

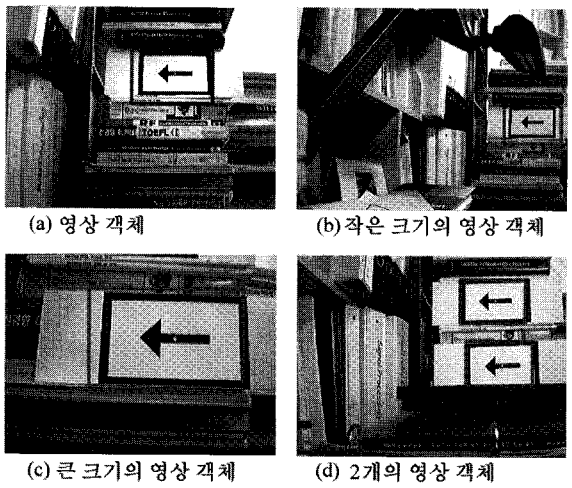
여러 가지 방향 표시를 갖는 표식들을 객체로 정의하고 영상에서 표시방향의 객체를 찾아 그 객체가 정확히 어떤 방향을 나타내는 flag 인지를 인식하는 것이 실험의 목적이다. 방향 표식들은 좌/우방향(b)(c), 직진/후진(d)(e), 일시정지(f)의 5가지의 방향 지시가 있고 일반적으로 (a)와 같은 방향 표시 객체를 정의하는 상자 안에 화살표와 다른 그림으로 정의된다. 표식들의 크기는 모두 같지만 관측거리에 따라 영상에서 여러 가지 크기와 형태로 나타나게 된다.

본 연구에서 생성된 분류기는 그림 6의 데이터 영상들을 이용해 학습되었으며, 특히 주목할 것은 객체 영상뿐만 아니라 주변 영상과 같은 잡음 영상들도 함께 사용되었다. 이것은 주변 영상 잡음에 대한 강인성을 높이기 위한 것이다.

그림 7은 CBCH기법과 템플릿 메칭을 통해서 방향 표시들을 검출한 모습이다. 그림 7(b), (c)와 같이 학습 데이터보다 작거나 큰 영상 객체에 대해서도 검출이 가능하였다. 또한 그림 7(d)과 같이 하나의 영상에 존재하는 둘 이상의 객체를 검출할 수도 있다.

표 1에 영상 크기와 템플릿 크기(size)에 따른 응답시간의 차이를 비교하였다. 응답시간은 호스트 컴퓨터와 타깃 보드에서 각각 실험하였으며, 오차를 줄이기 위해서 각 크기의 영상에 대해서 10번 수행하여 나온 결과의 평균을 취하였다. 그림 8을 통해서 입력 영상의 크기가 좌우로 2배 증가함에 따라 응답시간도 약 4배씩 증가함을 알 수 있다.

즉, 연산량이 작으면 빠른 응답시간을, 반대로 연산량이



(a) 영상 객체 (b) 작은 크기의 영상 객체

(c) 큰 크기의 영상 객체 (d) 2개의 영상 객체

그림 7. CBCH 기법과 템플릿 매칭에 의한 객체검출과 인식.

Fig. 7. Object detection using CBCH method and template matching.

표 1. 입력영상 크기와 템플릿의 크기에 따른 응답시간(ms).

Table 1. Response time on the image size and template size.

영상 크기		160x120	176x144	320x240	352x288	640x480	1280x960
응답시간	Host Comput.	188	312	437	468	1469	5662
	Target Board	3220	4130	107500	x	x	x
Template 크기		40x23	80x46	97x56	160x92	320x185	-
응답시간	Host Comput.	375	438	531	688	203	
	Target Board	3220	4130	107500		-	-

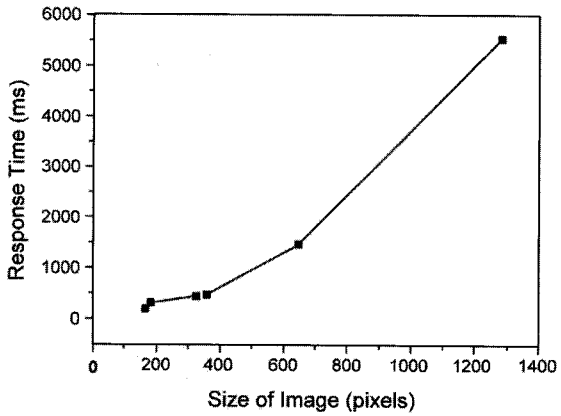


그림 8. 영상 크기에 따른 응답시간 의 변화곡선.

Fig. 8. Response time curve as image size .

많으면 늦은 응답시간을 보인다. 본 논문에서 사용할 320 × 240 픽셀 크기의 QVGA영상에 대해서 호스트 컴퓨터에서는 0.4초 그리고 임베디드 타깃 보드에서는 1분이 넘는 응답시간을 보여준다. 따라서 실시간 영상 객체 인식에 적합하지 않다.

표 2에서는 CBCH 기법을 이용할 때와 제안된 방법을 적용하였을 경우 응답시간이 현저히 줄어드는 것을 확인할

표 2. CBCH와 제안된 방법에서 영상 크기와 응답시간(ms).

Table 2. Response time on the image size for the CBCH and proposal methode.

영상 크기		160x120	176x144	320x240	352x288	640x480	1280x960
CBCH 응답시간	Host Comput.	15	16	28	31	63	203
	Target Board	1800	1860	2730	2800	8260	22410
Suggest 응답시간	Host Comput.	15	16	28	31	63	203
	Target Board	1180	1420	1640	2800	8260	22410

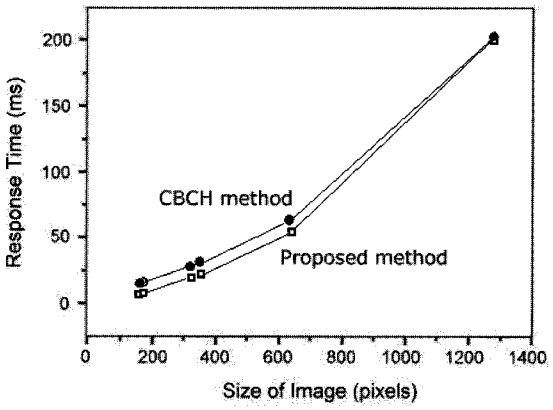


그림 9. CBCH와 제안된 방법에서 영상 크기와 응답시간의 변화 곡선.

Fig. 9. Response time on the image size for the CBCH and proposed methode.

수 있다. 호스트 환경에서는 동영상 입력에 대해서 거의 실시간 처리가 가능했으며 타깃 보드에서도 구동부의 제어를 충분히 만족시키는 응답시간을 보여주고 있다. CBCH 기법을 적용하여도 정확한 객체 검출을 위해서는 많은 연산을 필요로 한다. 실험에서는 임베디드 시스템을 타깃으로 하였기에 가장 보편적이고 단순한 형태의 Haar-Like 필터들을 사용하였다. 이 필터들은 덧셈과 뺄셈 그리고 단순한 로직 연산만을 사용하였으므로 빠른 성능을 보장 할 수 있었다.

CBCH 기법의 객체 검출율은 얼마나 많은 영상 데이터를 갖고 분류기를 학습 시켰느냐에 따라 나타난다. 실험에서는 총 100여장의 5가지 표식 촬영 사진들을 가지고 방향 표시 검출을 위한 분류기를 생성 하였다. 검출 과정에서 조명, 주변 배경화면등과 같은 요소가 고정된다면 CBCH 역시 거의 완벽한 검출 성능 보여 주었다. 그림 9에서 보는 바와 같이 제안된 방법은 CBCH 과정을 거쳐 검출된 객체 영상들에 대해서 다시 템플릿 매칭을 적용시키는 것이다. 따라서 CBCH 검출 과정에 필요한 응답시간 이외에 템플릿 영상으로 크기를 변환해 주는 스케일링 과정과 6장의 각기 다른 템플릿들과 비교하는 비교 과정에 필요한 응답시간이 추가적으로 필요하다. 그러나 추가적으로 발생되는 연산량을 줄이기 위해 템플릿 영상의 크기를 작게 조정하여서 CBCH 과정 이외의 시간들은 응답시간에 거의 영향을 주지 않게 하였다.

IV. 결론

본 논문은 기억 용량과 연산 능력이 제한된 임베디드 시스템에서 최소의 연산으로 객체를 검출하여 판별하는 패턴 인식과 검출 방법을 제안하였다. 낮은 전력과 실시간 응답 특성이 요구되는 임베디드 시스템의 자율 주행 운반체(carrier)에 이용하기 위해서는 그 특성에 맞게 연산량이 작은 수정된 알고리즘 개발이 필요하였다. 기존의 알고리즘들은 인식률에 비례하여 연산량이 증가하는 경향을 보였으나 제안된 알고리즘은 적은 연산량으로 높은 인식률을 보여주었다. 제안된 알고리즘은 인텔사에서 제공되는 open source computer vision library 영상처리 라이브러리[9]를 사용하여 구현되었으며, ARM 크로스 컴파일러를 통해서 임베디드 시스템에서 실행 가능하도록 설계하였다.

수정 보완된 인식방법은 ARM920T 프로세서로 구동하는 Embedded System에 적용되어 실험을 하였다. 320 x 240 픽셀 크기의 QVGA영상에 대해서 제안된 방법은 1.6sec의 평균 응답 시간을 나타내었으며, 95%의 인식률로 자율 주행 운반체를 원활하게 조정 할 수 있었다. 활용 분야로는 구조적으로 잘 설계된 물류 창고의 간단한 물류 운반체에 적용되어 보다 경제적이고 편리한 운전 환경을 제공할 것이다. 만약 입력영상과 배경영상이 움직임이 있을 때와 객체의 이동이 있을 때에는 자가 학습할 수 있는 검출기를 부가하고 다중 객체 처리 방법을 보완하면 인식오류의 문제를 어느 정도 개선할 수 있을 것으로 기대한다.

참고문헌

[1] Y. Ming-Hsuan, D. J. Kriegman, and N. Ahuja, "Detecting faces in images : a survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions*, vol. 24, no. 1, pp. 34-58, Jan 2002.

[2] 박재표, 이광형, 이종희, 전문석, "객체 추출 및 추적을 이용한 실시간 웹기반 영상 감시 시스템," 전자공학회 논문지, 제41권 CI편, 제4호, pp. 85-94, 2004.

[3] B. Froba and C. Kublbeck, "Orientation template matching for face localization in complex visual scenes," *Image Processing, 2000. Proceedings. 2000 International Conference,* vol. 2, (10-13) pp. 251-254. 2000.

[4] Rainer Lienhart and Jochen Maydt. "An extended set of haar-like features for rapid object detection." *Image Processing.* 2002. *Proceedings.* 2002 *International Conference,* vol. 1, pp. 900-903, Sep. 2002.

[5] Paul Viola and Michael J. Jones. "Rapid object detection using a boosted cascade of simple features."*Conference on CVPR, IEEE,* vol. 1, pp. 511-518, 2001.

[6] G. Francois, L. Eberhard, and Takashi Iwamoto, Kazuo Kyuma, Nobuyuki Otsu, "Face recognition system using local autocorrelations and multiscale integration," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1024-1028, Oct. 1996.

[7] T. V. Pham, M. Worring, and A. W. M. Sneulders, "Face detection by aggregated bayesian network classiers," *Elsevier, Pattern Recognition Letters,* vol. 23, no. 4, pp. 451-461, Feb. 2002.

[8] Rafael C. Gonzalez, Richard E. Woods, "Digital image processing," 2nd Edition, *Prentice Hall*, 2001, Chapter 2.4.

[9] G. Bradski, A. Kaehler, and V. Pisarevsky, "Learning-based computer vision with intel's open source. computer vision library" *Intel Technology Journal*, vol. 9, no. 2, pp. 119-130, May 2005.

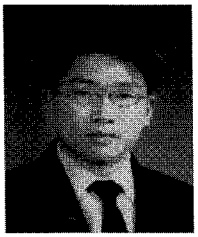
[10] E. Hjelm and B. K. Low, "Face detection: a survey," *Computer Vision and Image Understanding*, vol. 83, no. 3, pp. 236-274, 2001.

[11] F. Crow, "Summed-area tables for texture mapping."11th *Proceedings of conference on Computer Graphics and Interactive Techniques, SIGGRAPH,* vol. 18, no. 3, pp. 207-212, Jul. 1984.

[12] F. Fleuret and D. Geman. "Coarse-to-Fine face detection." *Int. J. Computer Vision*, vol. 41, no. 1-2, pp. 85-107, Jan. 2001.

[13] William T. Freeman and Edward H. Adelson. "The design and use of steerable filters." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891-906, Sep. 1991.

[14] H. Greenspan, S. Belongie, R. Goodman, P. Perona, S. Rakshit, and C.H. Anderson. "Overcomplete steerable pyramid filters and rotation invariance." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* pp. 222-228, Jun. 1994.



김 연 일

2002년 서울시립대 전자전기컴퓨터공학부 졸업. 2006년 동 대학원 전자전기컴퓨터공학부 공학석사. 2006년~현재 박사과정 중. 관심분야는 영상처리, 임베디드 시스템, SoC 설계 방법론.

노 승 용

1971년 한양대학교 전자공학과 공학사 1973년 동 대학원 전자과 공학석사. 1988년 동 대학원 전자과 공학박사. 1982년~현재 서울시립대 전자전기컴퓨터공학부 교수. 관심분야는 임베디드 시스템, SoC 설계 방법론, 영상처리.