컬러정보와 오류역전파 알고리즘을 이용한 교통표지판 인식

방 걸 원[†] · 강 대 욱[††] · 조 완 현[†††]

요 약

본 논문에서는 컬러정보를 이용하여 교통표지판 영역을 추출하고, 추출된 이미지의 인식을 위해 오류 역전파 학습알고리즘을 적용한 교통표지판 인식시스템을 제안한다. 제안된 방법은 교통표지판의 컬러를 분석하여 영상에서 교통표지판의 후보영역을 추출한다. 후보영역을 추출하는 방법은 RGB 컬러 공간으로부터 YUV, YIQ, CMYK 컬러 공간이 가지는 특성을 이용한다. 형태처리는 교통표지판의 기하학적 특성을 이용하여 영역을 분할하고, 교통표지판 인식은 학습이 가능한 오류역전파 학습알고리즘을 이용하여 인식한다. 실험결과 제안된 시스템은 다양한 크기의 입력영상과 조명의 차이에 영향을 받지 않고 후보영역 추출과 인식에 우수한 성능이 입증되었다.

키워드 : 교통표지판, 인식, 컬러정보, 컬러공간, 오류역전파알고리즘, 다층퍼셉트론

Traffic Sign Recognition Using Color Information and Error Back Propagation Algorithm

BANG GUL WON[†] · KANG DEA WOOK[††] · CHO WAN HYUN[†††]

ABSTRACT

In this thesis, the color information is used to extract the traffic sign territory, and for recognizing the extracted image, it proposes the traffic sign recognition system that applies the error back propagation algorithm. The proposed method analyzes the color of traffic sign to extract and recognize the possible territory of traffic sign. The method of extracting the possible territory is to use the characteristics of YUV, YIQ, and CMYK color space from the RGB color space. Morphology uses the geometric characteristics of traffic sign to make the image segmentation. The recognition of traffic signs can be recognized by using the error back propagation algorithm. As a result of the experiment, the proposed system has proven its outstanding capability in extraction and recognition of candidate territory without the influence of differences in lighting and input image in various sizes.

Key Words : Traffic Sign, Recognition, Color Information, Color Space, Error Back Propagation Algorithm, Multi-layer Perceptron

1. 서 론

교통표지판의 자동 인식에 관한 연구는 교통표지판 상황을 인식하여 운전자에게 인지시켜줌으로써 교통사고를 사전에 방지할 수 있고 자동주행시스템을 실현할 수 있다. 교통표지판은 도로안전시설 중 없어서는 안 될 중요한 부분으로 기호, 문자, 색으로 구성되어 있으며 도로구조의 보전과 안전하고 원활한 교통소통을 위하여 운전자에게 목적지까지의 방향, 거리, 등의 지점 및 경로 안내를 위한 다양한 정보를 제공한다[2]. 그러므로 자동주행시스템에서 교통표지판의 인식은 필수적으로 해결해야 할 연구과제이다.

일반적인 크기의 도로표지판 식별거리는 200m 정도로 알려져 있다[4]. 가령 고속도로 상에서 100Km/h 속도로 달린다고 가정하면 차량이 200m 전방의 교통표지판을 지나는데 소요되는 시간은 약 7.5초 이다. 이렇게 짧은 순간에 운전자의 부주의는 대형 사고를 유발시키기에 충분한 시간이 된다. 교통표지판은 매우 짧은 가시거리에서도 노선에 대한 대략의 정보를 제공할 수 있다. 사람의 시각은 자연 또는 인공의 배경으로부터 쉽게 교통표지판을 구별할 수 있지만 신체적, 정신적 환경에 따라 크게 영향을 받으므로 교통표지판 자동 인식시스템은 운전자나 원격조정 자동차의 보조수단으로 사용될 수 있다.

기존 교통표지판 인식 시스템을 고찰해보면 다음과 같다. 첫 번째로 컬러비와 거리비를 이용한 교통표지판 영역추출 방법은 하나의 픽셀이 가지는 컬러 성분들의 컬러비와 코너점, 중암점 거리비등의 형태정보를 기반으로 추출하는 방법으로, 입력영상에서 교통표지판 영역의 크기가 너무 작은

† 정 회 원 : (주)위너테크 기술연구소 소장
†† 종신회원 : 전남대학교 컴퓨터정보학부 교수
††† 정 회 원 : 전남대학교 통계학과 교수
논문접수 : 2007년 6월 4일, 심사완료 : 2007년 8월 27일

경우에 코너점 검출 마스크를 이용한 코너점 검출이 불가능하고, 컬러 성분의 밝기 값의 변화가 많은 경우에도 정확한 교통표지판 영역 검출이 불가능하다[1]. 두 번째로 웨이블릿(Wavelet)변환과 형태정보를 이용한 교통표지판 인식 방법은 추출된 표지판 영역에 대해서 웨이블릿 변환을 적용하여 얻은 고주파 및 저주파정보를 기반으로 모멘트, 에지 코렐로그램(edge correlgram), 동심 원형 패턴 정보를 추출하고 사전에 구축한 데이터베이스와의 유사도측정에 의해 인식하는 방법으로, 입력 영상을 128×128픽셀의 크기로 정규화하는 전처리과정이 필요하므로 처리 시간이 길어지고, 입력영상이 너무 작은 경우에 크기 정규화를 위해 이용한 선형보간법은 많은 정보의 손실을 유발하여 인식률이 저하되고, 실영상에서도 인식률이 저하된다[6]. 세 번째로 Walsh함수 모델을 이용한 교통표지판 인식 방법은 Walsh Mask와 내적 하여 특성 치를 얻어 추출하고, 인식은 Walsh 스펙트럼 특성 치를 사용하여 인식하는 방법으로, 교통표지판 영상의 이동, 회전, 크기에 민감하고, Walsh Mask를 작게 생성하면 처리 속도는 빠르지만 인식률이 저하된다[5]. 네 번째로 불변 모멘트 법을 이용한 도로 교통 표지판 인식 방법은 표지판 영상의 이동 회전 시 획득되는 불변 특성을 이용하여 인식하는 방법으로 모멘트 값이 세선화 될수록 크기가 커지며, 불변 모멘트의 회전 불변 특성으로 좌회전, 우회전 구분이 어렵다[7].

이에 본 논문에서는 YUV컬러공간, YIQ컬러공간, CMYK컬러공간의 컬러 채널을 이용하여 노란색은 흰색으로 적색은 검정색으로 변환하여 이진영상의 생성속도를 높여서 처리속도를 단축하고, 미리 학습을 하는 오류역전파 학습알고리즘을 적용하여 인식하는 시스템을 제안한다.

2. 교통표지판 인식 시스템의 구성 요소

2.1 교통표지판 구성

교통표지판은 각 나라별로 통일된 규격은 아니지만 형태나 색상, 크기, 기호 등이 유사하다. 우리나라의 교통표지판의 구조는 기능면에서 주의표지판, 규제표지판, 지시표지판, 보조표지판으로 크게 4가지로 분류된다. 주의표지판은 (그림 1)과 같은 구조로 41종, 규제표지판은 (그림 2)와 같은 구조로 32종, 지시표지판은 30종, 보조표지판은 26종으로 총 129종이다. 또한 교통표지판의 설치는 운전자에게 잘 보이도록 도로상에서 주의표지판과 규제표지판은 지상 100㎝~210㎝, 지시표지판과 보조표지판은 지상 100㎝ 이상으로 설치한다[2].

(그림 1) 주의표지판

(그림 2) 규제표지판

2.2 컬러 영상모형

영상처리에서 사용되고 있는 컬러공간(Color Space)은 컬러들과 다른 컬러들과의 관계를 표현하는 방법이고 영상처리의 다양한 분야에서 서로 다른 컬러 공간을 사용한다. 일반적으로 사진이나, 그림을 출력할 경우에는 CMY(Cyan, Magenta, Yellow)컬러 공간을 사용하고 컴퓨터의 모니터나 컴퓨터 그래픽카드들은 RGB컬러공간을, 컬러TV에서는 YUV컬러공간과 YIQ컬러공간을 사용하며 인간의 시각을 기반으로 한 색상(Hue), 채도(Saturation), 명도(intensity)를 사용하는 HSI 컬러공간과 Hue, Saturation, Value를 사용하는 HSV컬러공간, JPEG압축용으로 YC_bC_r컬러공간도 사용한다[7, 8, 9]. 이와 같이 각각의 컬러모델들은 RGB컬러모델이 기본이 되며 상호변환이 가능하다.

2.3 형태연산(morphology)

모폴로지(morphology)는 어떤 영상의 형태적인 면을 사용한다. 모폴로지 알고리즘은 침식 연산, 팽창 연산, 열림 연산, 닫힘 연산으로 분류되고, 각 연산은 독립적으로 사용되지만 일반적으로 원하는 결과 영상을 얻기 위해 여러 개의 연산을 병행하는 경우가 많다.

경계, 골격 볼록면의 경계(convex hull)와 같은 영역의 형태를 표현하거나 서술하는데 있어서 유용한 영상 요소들을 추출하기 위한 도구로 수학적 형태론의 개념이 사용된다.

2.4 컬러영상의 분할(Color Image Segmentation)

컬러영상 분할이란 영상을 명암이나 색채가 같은 픽셀 성질의 분포를 기준으로 영역을 나누는 것이다[3, 10].

컬러영상의 분할 방법은 영역의 병합(Merge), 영역의 분리(Splitting), 그리고 병합과 분리의 병행이다. 영역의 병합이란 영상을 기본적인 단위로 나누고, 특정 성질의 유사성 판정에 따라 같은 부류를 합쳐가는 방법이다. 기본적인 단위는 픽셀(Pixel)로 정할 수 있다. 픽셀은 한 점에서의 빛의 세기를 나타내는 화상의 기본 요소이다. 영역의 분리는 주어진 범주의 특성 값에 의하여 영역을 두 부분으로 나눈다. 병합과 분리의 병행이란 주어진 특성 값이 현저하게 다르면 분리과정을 행하고, 특성 값이 근접하면 병합과정을 행하는 방법이다. 영역 분할의 방법에는 히스토그램(Histogram)을 사용하는 방법이 있는데, 분할의 기준이 되는 한계치(threshold value)를 구하는 방법으로 히스토그램(Histogram) 곡선에 대한 극대-극소 분석법을 이용한다. 영역 분할의 결과는 2차원적 영역 구획 영상인데 각 영역에는 고유한 레이블이 붙여진다. 각 영역은 폐곡선으로 둘러싸이게 되고 둘러싸인 부분에서 영상 기본요소의 특정 값은 일정한 범위 안에 있어야한다. 특성 값은 스펙트럼적 성질, 모양에 관한 특성 값, 경계선의 성질 등으로 구분 지을 수 있다. 여기에서 스펙트

럼적 성질은 빛의 세기, 색상(Color), 무늬모양(Texture)등이며, 모양에 관한 특성 값은 면적, 이심율(Eccentricity), 밀집도(Compactness)등이고, 경계선의 성질은 굴곡도(Curvature), 대비(Contrast) 등이 있다[9].

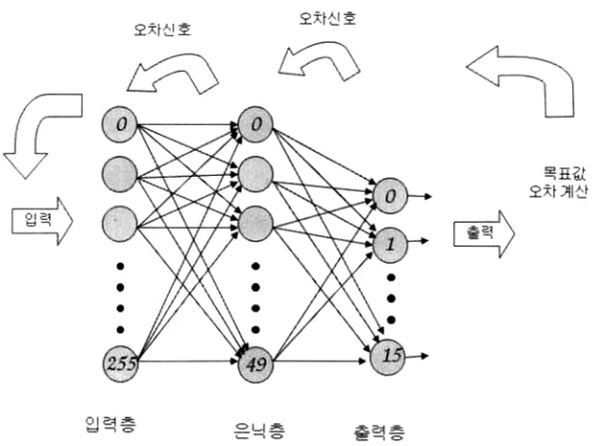
영상 영역 분할은 명암도의 불연속성과 명암도의 유사성으로 분할할 수 있다. 명암도의 불연속성으로 영역을 분할한다면 점 검출(point detection), 선 검출(line detection), 에지 검출(edge detection) 등의 방법이 있고 명암도의 유사성으로 분할한다면 영역 확장(region growing), 영역 분할과 병합(region splitting and merging), 한계치에 의한 차단법(Thresholding) 등의 방법이 있다.

2.5 다층퍼셉트론(Multi-layer Perceptron)

다층퍼셉트론은 입력층(input layer)과, 은닉층(hidden layer)을 가지고 있으며, 각층은 다수의 뉴런으로 이루어져 있다. 특정 층을 이루고 있는 각각의 뉴런이 바로 윗층의 모든 뉴런과 연결되어(fully-connected) 있으면, 뉴런들 사이의 연결링크는 각각의 가중치를 갖는다. 그러나 다층 퍼셉트론은 각 층 내의 뉴런들 사이의 연결과 출력층에서 입력층으로의 직접적인 연결은 존재하지 않는 전방향(feedforward) 신경망이다. 대부분의 다층 퍼셉트론의 학습은 역전파 알고리즘을 사용하여 수행할 수 있다. (그림 3)은 역전파학습 알고리즘의 구조이다. 이 알고리즘은 주어진 입력에 대해 원하는 출력결과를 학습시키고자 할 때 사용하며, 출력층의 각 뉴런에서 발생되는 출력오차를 각 층의 역으로 전파시켜 나가면서 연결링크의 가중치 수정을 통해 오차를 최소화시킨다[12].

3. 교통표지판 인식 시스템의 구조

제안하는 교통표지판 인식 시스템은 여러 가지 도로 시설물 중 교통표지판이 포함된 영상에서 교통표지판의 위치를 구하여 영역을 추출하고, 추출된 이미지를 인식하여 교통표지판의 종류를 판별하는 알고리즘이다. 제안된 교통표지판 인식 시스템의 처리 과정은 (그림 4)과 같이 크게 전처리, 분할, 인식의 3단계로 나눌 수 있다.



(그림 3) 오류역전파 학습알고리즘

제1단계 전처리과정은 이진 영상을 생성하기 위한 전처리 단계로 도로상에 있는 교통표지판을 디지털카메라로 촬영하고, 촬영된 영상을 입력한다. 입력된 영상에 컬러 균일화를 하고, 컬러를 검색하여 노란색은 흰색으로, 빨간색은 검정색으로, 검정색은 RGB 값이 0인 검정색으로 변환하여 이진 영상을 생성한다.

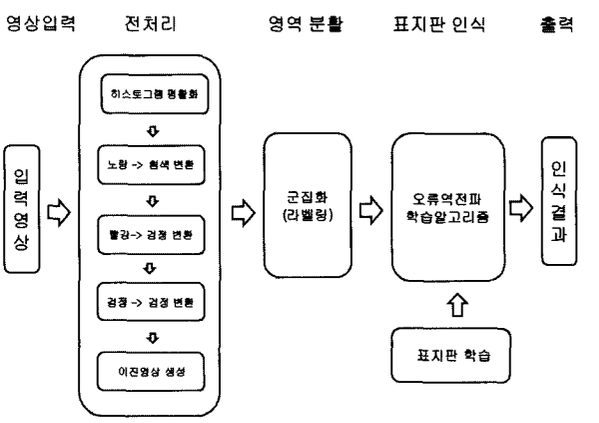
제2단계 영역분할은 배경과 표지판의 이미지를 분할하는 단계로 변환된 이진영상을 군집화과정을 거쳐 교통표지판 영역으로 추출한다. 군집화 과정은 영상의 연결성분을 찾기 위한 방법으로 스택을 이용한 라벨링 알고리즘을 사용하였다.

제3단계 표지판인식은 교통표지판을 미리 학습하고 신경망 이론인 다중 퍼셉트론의 오류역전파 학습알고리즘을 이용하여 학습된 데이터를 기초로 교통표지판을 인식한다.

3.1 히스토그램 평활화

영상 평활화를 하기 위한 알고리즘은 먼저 (그림 5)과 같이 RGB 영상을 Gray 영상으로 변환한 후 히스토그램(Histogram) 값을 구하고 구해진 히스토그램 값을 이용하여 히스토그램 배열을 생성한다. 경계를 검출하기 위한 임계값은 백분율을 사용한다.

히스토그램의 최저값은 0에서부터 255까지의 단계로 명암값을 증가하면서 빈도수의 누적 합을 구하고, 빈도수와 임계값을 비교하여 빈도수가 임계값보다 작은 빈도수의 누적 합을 구한다. 히스토그램의 최고값은 히스토그램의 최저값을 구하는 방법과 반대로 명암값을 255에서 0까지의 단계로 감소시켜서 빈도수의 누적 합을 구하고, 빈도수의 누적 합



(그림 4) 교통표지판 인식시스템

```
for(y=0; y < 이미지높이; y++){
    for(x=0; x < 이미지넓이; x++){
        color = GetPixelColor(ppImg);
        히스토그램값 = RGB2GRAY(Red, Green, Blue);
          histogram[히스토그램값]++;    배열
        ppImg += 3;
  } }
   임계값 = ( 이미지 높이 * 이미지 넓이) / 100;
//히스토그램 경계검출 1% 단계로 설정
```

(그림 5) 히스토그램 값 구하는 알고리즘

은 빈도수와 임계값을 비교하여 빈도수가 임계값보다 작은 경우의 빈도수만 누적 합을 한다. 최저값과 최고값이 같은 경우에 임계값을 0으로 설정하고 빈도수의 최저값의 누적합과 최고값의 누적합을 다시 구한다. 이런 과정을 반복하여 최저값과 최고값이 같지 않을 때까지 위 과정을 반복 수행한 후, 표준화된 배열을 생성한다.

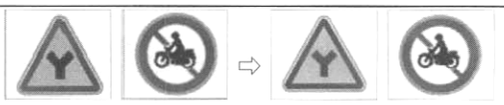
평활화 마지막 단계로 원래의 RGB 영상을 YUV 영상으로 변환하여 밝기(Luminance)인 Y성분(Brightness)을 이용하여 평활화를 수행한 후 다시 YUV영상을 RGB 영상으로 변환한다. 여기에 적용한 알고리즘과 영상은 (그림 6)과 같다.

3.2 컬러 검색 (Color Detection)

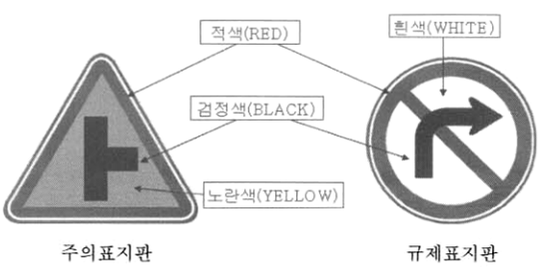
교통표지판의 주의, 규제표지판은 (그림 7)과 같이 적색, 노란색, 흰색, 검정색의 4가지 컬러로 구성되어 있다.

지시표지판은 파란색 바탕에 흰색으로 구성되며, 보조표지판은 흰색바탕에 검은색으로 구성되어 앞의 3가지 표지판과 함께 사용된다. 이와 같이 3가지 이상의 컬러로 구성된 표지판을 대상으로 교통표지판추출과 인식을 위해 컬러를 검색하여, 적색과 검정색은 검정색으로 변환하고, 노란색은 흰색으로 변환한다. 적색에서 검정색으로의 변환은 YIQ컬러의 I 값을 이용하고, 노란색에서 흰색으로의 변환은 YUV컬러의 V 값을 이용하여 변환한다. 교통표지판의 내부 기호가 시각적으로 검정색처럼 보이지만 실제로는 검정에 가까운 색으로 RGB 값이 0이 되는 검정색으로 변환하기 위해 CMYK 컬러의 C값을 이용하여 변환한다. 변환된 이미지에서 RGB 값이 0이외의 값이면 모두 흰색으로 변환하고 0의 값은 검정색으로 변환하면 빠른 처리 속도로 이진영상으로 변환 된다.

```
ppImg = pImage + 이미지size ;
  for( y = 0; y < 이미지높이; y++ ){
  for( x = 0; x < 이미지넓이; x++ ){
        color = GetPixelColor(ppImg);
        yuvClr = RGBtoYUV( color );
  YUV적색 = (BYTE)normalize_map[YUV적색];
                  color = YUVtoRGB( yuvClr );
                  SetPixelColor(ppImg,color);
                  ppImg += 3;}
 }
 return ; }
```



(그림 6) Y값을 이용한 평활화 알고리즘과 영상



(그림 7) 교통표지판 컬러 구성

컬러 검색은 교통표지판의 영상을 분석하여 검정색과 흰색으로 변화하는 과정이다. 교통표지판의 주의표지는 노란색 바탕과 적색 테두리에 검정색 기호로 구성되어 있다. 노란색은 흰색으로 변환하고 적색은 검정색으로 변환하며 규제표지는 흰색 바탕과 적색 테두리에 검정색 기호로 표시되어 있어 적색을 검정색으로 변환하는 알고리즘을 이용한다.

3.2.1 노란색(Yellow)을 흰색(White)로 변환

노란색을 흰색으로 변환하기 위해서 RGB 영상을 YUV 영상을 변환하고, 변환된 YUV 영상에서 V성분(Saturation) 값을 이용하며, V성분 값이 임계값보다 작은 픽셀은 모두 흰색으로 변환한다. (그림 8)는 노란색을 흰색으로 변환하는 알고리즘과 영상이다.

3.2.2 적색(Red)을 검정색(Black)으로 변환

적색을 검정색으로 변환하기 위해서 RGB 영상을 YIQ 영상으로 변환하고, 변환된 YIQ 영상에서 I(in-phase) 값이 임계값 보다 작은 픽셀을 모두 검정색(black)으로 변환하였다. (그림 9)는 적색을 검정색으로 변환하는 알고리즘과 영상으로 컬러값이 0이면 검정색이다.

3.2.3 검정색을 검정색으로 변환

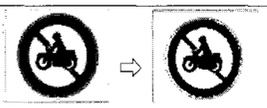
검정색(Black)에서 검정색으로의 변환은 사람의 시각으로 보면 검정색으로 보이지만 실제로는 검정색에 가까운 색으로 RGB 컬러 값이 0인 검정색이 아니므로 RGB 컬러값이 0인 검정색 영상으로 변환하여야 한다. 변환하는 과정은 RGB 영상을 CMYK 영상으로 변환하고 변환된 CMYK 영

```
* RGB를 YUV로 변환 알고리즘
        for(y=0; y < psbi->이미지높이; y++)
{
        for(x=0; x < 이미지넓이; x++)
{
                color = GetPixelColor(ppImg);
                yuvClr = RGBtoYUV( color );
        RGB 적색  = YUV 녹색 ;
        RGB 녹  색  = YUV 녹색 ;
        RGB 파란색  = YUV 녹색 ;
                SetPixelColor(ptimg,color);
                        ppImg += 3;
                        ptimg += 3;

}
}
* YUV에서 V가 임계값보다 작으면 흰색으로 변환하는 알고리즘
  for( y = 0; y < 이미지높이; y++ ){
  for( x = 0; x < 이미지높이; x++ ){
  color = GetPixelColor(ptimg);
    if (RGB 파란색 < 임계값){
     RGB 파란색  = 255; // 255는 흰색
     RGB 녹  색  = 255;
     RGB 적색  = 255;
        SetPixelColor(ppImg,color);
 }
```

(그림 8) 노란색을 흰색으로 변환 알고리즘과 영상

```
* RGB를 YIQ로 변환 알고리즘
        for(y=0; y < 이미지높이; y++)
{
                    for(x=0; x < 이미지넓이; x++){
                    color = GetPixelColor(ppImg);
                    hslClr = RGBtoYIQ( color );
                      RGB 적색  = HSI 녹색 ;
                  RGB 녹  색  = HSI 녹색 ;
                 RGB 파란색  = HSI 녹색 ;
                        SetPixelColor(ptimg,color);
                    ppImg += 3;
                                ptimg += 3; }
}
* YIQ에서 I가 임계값보다 크면 검정색으로 변환하는 알고리즘
        for( y = 0; y < 이미지높이; y++ ){
        for( x = 0; x < 이미지넓이; x++ ){
                    color = GetPixelColor(ptimg);
  if (RGB 녹색 > 임계값) //임계값보다 크면 검정색
{
          RGB 파란색  = 0; 0은 검정색
          RGB 녹  색  = 0;
          RGB 적색  = 0;
        SetPixelColor(ppImg,color); }
          ptimg = ptimg +3;
        ppImg = ppImg +3;
}
}
```



(그림 9) 적색을 검정색으로 변환 알고리즘과 영상

```
for( y = 0; y < 이미지높이; y++ ){
    for( x = 0; x <이미지넓이; x++ ){
        color = GetPixelColor(ppImg);
        cmyk = RGBToCMYK(color);
// RGB영상을 CMYK 영상으로 변환
        maxb = min(min(RGB적색,RGB녹색),RGB파란색);
        minsum = (RGB적색-minb)+(RGB녹색-minb)+
                 (RGB파란색-minb) ;
if ( CMYK의 C > 150 ) { //임계값 150
                    RGB 녹  색  = 0; //검정색
                    RGB 파란색  = 0; //검정색
                    RGB 적색  = 0; //검정색

SetPixelColor(ppImg,color);
                                }
                                ptimg = ptimg +3;
                                ppImg = ppImg +3; }      }
```

(그림 10) 검정색을 검정색으로 변환 알고리즘

상의 C (Cyan)값이 임계값과 비교하여 임계값보다 크면 검정색으로 변환한다. 실제 구현에서는 임계값을 150으로 정하고 C 값이 150이상이면 검정색으로 변환하였다. (그림 10)은 검정색(Black)을 검정색으로 변환하는 알고리즘이다.

3.3 이진영상 생성

이진 영상은 (그림 11)과 같이 쉽고 빠르게 생성할 수 있다. RGB값이 0인 픽셀은 검정색으로 그 외의 픽셀은 모두 흰색으로 변환하면 이진 영상이 생성된다. 이미 컬러검색 단계에서 이진화를 고려하여 컬러변환을 하였기 때문이다. 노란색은 흰색으로, 적색은 검정색으로 변환하였으므로 이진 영상의 생성이 빠르게 처리가 된다.

```
for( y = 0; y < 이미지 높이; y++ ){
        for( x = 0; x < 이미지넓이 ; x++ ){
          color = GetPixelColor(pImg) ;
      if (color.rgbRed > 1){  // 1 이외 값은 모두 흰색 변환
      color.rgbBlue   = 255 ;
      color.rgbGreen  = 255 ;
       color.rgbRed    = 255 ;
        }else{        // 1 이면 검정색으로 변환
                    color.rgbBlue   = 0 ;
                    color.rgbGreen  = 0 ;
                    color.rgbRed    = 0 ;
        }
        SetPixelColor(pImg,color );
        pImg        = pImg       +3;
    }
}
```

(그림 11) 이진 영상 생성 알고리즘과 영상

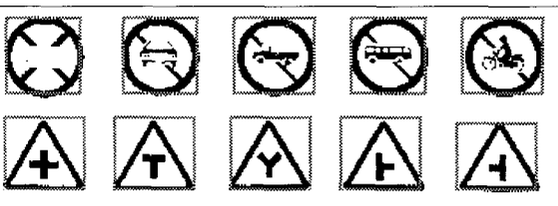
```
void BYMFILL_byte2D( int x, int y, BYTE Col )
{
   int     i,t,b,l,r,area;
   int     su,sd,stp, End, lx;
   register BYTE   *temp1;
   BYTE    *tempU,*tempD,*tempT;
   stacks *tempstack;
   t=y; b=y; l=x; r=x; area = 0;
   stp = 1;  stack[0].y = y; stack[0].x = x;
   while(  stp > 0 ){ End = stp;   stp = 0;
tempstack = tempX;  tempX = stack;  stack = tempstack;
for(i=0;i<End;i++){ y = tempX[i].y;  x = tempX[i].x;
//표지판 Pixel이 있는 위치 찾기

  temp1 = String+(y*oWString)+x;  //표지판 Pixel이 있는 경우
 if( *temp1 == 1 ){  do{  temp1--;  x--; }while( *temp1 == 1 );
 lx = x+1; tempT = temp1+1;
 tempU = temp1-oWString;  tempD = temp1+oWString; su = sd = 0;

do{ if ( *(tempU) != 1 )   su = 0;
   else if( su == 0)
    {su = 1; stack[stp].y = y-1; stack[stp++].x = x;}
   if( *(tempD) != 1 )   sd =0;  else if( sd == 0)
  {sd = 1; stack[stp].y = y+1; stack[stp++].x = x;}
    tempU++; tempD++;  x++; temp1++;
  }while( *temp1 == 1 );
  if( *(tempU)==1 && su==0 )
  {  stack[stp].y = y-1;   stack[stp++].x = x; }
  if( *(tempD)==1 && sd==0)
   {stack[stp].y = y+1;   stack[stp++].x = x; }
//표지판 Pixel을 다른 Color로 채워기

   memset(tempT,Col,x-lx);   //표지판 크기 누적
f(t> y) t=y;  if(b<y) b=y; if(l>lx) l=lx;
              if(r<x-1) r=x-1; area += x-lx; }
          }
} /* End of While */

if(TextB->num>=500)  return;
TextB->bi[TextB->num].t = t;
TextB->bi[TextB->num].b = b;
TextB->bi[TextB->num].l = l;
TextB->bi[TextB->num].r = r;
TextB->bi[TextB->num].col = Col;
TextB->bi[TextB->num].br = (float)area/((b-t+1)*(r-l+1));
TextB->bi[TextB->num].np=area;
TextB->num++;
}
```



(그림 12) 스택을 이용한 라벨링 알고리즘과 추출된 영상

```
void LearningNet()
{pNeuralNetBuf->pIvalue = pTrainBuf[nPcnt].pIvalue;  //입력
ComputeNeuralNet();
pLerr = pLayerErrBuf->pOerr;
pValue = pTrainBuf[nPcnt].pOvalue;                    //출력
pValue2 = pNeuralNetBuf->pOvalue;
for(i = 0; i < pNeuralNetBuf->nOutputCnt; i++)  //step 3
*pLerr = (*pValue - *pValue2) * (*pValue2) * (1 - *pValue2);
*pLerr = HArctan((*pValue - *pValue2)) * ((*pValue2)
        *(1 - *pValue2) + 0.1);
diff += F_ABS((*pValue - *pValue2));
for(i = 0; i < pNeuralNetBuf->nHiddenCnt; i++)  //step 4
pLerr2 = pLayerErrBuf->pOerr;

for(i = 0; i < pNeuralNetBuf->nHiddenCnt; i++)  //step 5
pLerr = pLayerErrBuf->pOerr;
pOffset = pNeuralNetBuf->pOoffset;
for( i = 0; i < pNeuralNetBuf->nInputCnt; i++)  //step 6
pOffset = pNeuralNetBuf->pHoffset;
pLerr = pLayerErrBuf->pHerr;
{if((nLearnCnt%100) == 0) printf("Diff
Value[%f] %d\n",diff,nLearnCnt);
nLearnCnt++;} while((diff > DIFF_MIN) && (nLearnCnt < ll));
```

(그림 13) 학습알고리즘

3.4 군집화

영상의 연결 성분을 찾기 위한 방법으로 스택을 이용한 라벨링 알고리즘을 사용하였다. 라벨링에서 스택을 이용하면 순차 연결성분 알고리즘보다 쉽고 수행시간도 짧다. 스택을 이용한 라벨링 알고리즘은 (그림 12)와 같이 제1단계로 입력된 영상에서 픽셀을 검색하고 원하는 영상의 픽셀이 최초로 검색되면 일단 검색을 정지하고 알고리즘을 수행한다. 발견된 픽셀의 위치 값을 스택 1에 저장한 후 제2단계로 4-이웃 픽셀을 검색하여 검색된 픽셀의 위치를 스택2에 저장한다. 제2단계를 반복적으로 수행하여 픽셀이 없을 때까지 알고리즘을 수행한다. 제2단계가 모두 끝나면 스택에 저장되었던 값들은 영상을 구성하는 모든 픽셀의 위치 값이 된다. 이렇게 구해진 위치 값을 이용하여 교통표지판 영역의 위치를 찾는다. 교통표지판의 후보영역은 5:5의 비율이다. 후보 영역에서 교통표지판을 위치 값과 크기를 이용하여 5:5 비율로 형태정규화하면 교통표지판이 추출된다.

3.5 교통표지판 인식

교통표지판 인식을 위해 표준 샘플의 특징점을 추출하여 학습을 시킨다. (그림 13)은 학습을 하는 알고리즘이다.

(그림 14)는 실제 구현한 인식알고리즘으로 하나의 은닉층을 가지는 3층 구조로 구성하였다. 이미지입력은 입력층으로서 학습을 위한 데이터를 대입하는 부분으로, 본 논문에서는 교통표지판의 학습과 식별을 위해 주의표지판 8종와 규제표지판 8종, 총 16종의 교통표지판을 학습알고리즘의 학습데이터로 사용하고, 입력층 노드 수를 256으로 설정하였다. 출력층의 노드 수는 초기 학습에서 출력층의 결과를 미리 결정해 주고 학습을 통하여 각 층별로 가중치를 얻어낼 수 있게 학습을 수행하고, 16종의 교통표지판을 이용하기 때문에 출력노드수를 16으로 설정하였다. 은닉층은 은닉층의 개수에 의하여 인식률과 학습하는데 걸리는 시간이 좌우된다. 실제구

```
fp=fopen("olsr.bwt","rb"); //학습된 파일열기
        if(fp==NULL)
RaiseException(EXP_CODE(OEX_NO_WEIGHT),0,0L,NULL);
for(n=0;n<NOUT;n++) fread(outwgt[n],sizeof(float),NHDN+1,fp);//0-30
for(n=0;n<NHDN;n++) fread(hiwgt[n],sizeof(float),NIN+1,fp);//0-255
        fclose(fp);
int  out_layer_forward(int type) { //출력층
 register int j; int nout,nhidden;  int winner; float *wpt,net,max;
 float  sum0,sum1,sum2,sum3,sum4;
 int k,ked;              char *charset; FILE *pFile;
 wpt= outwgt[0]; nout=NOUT; nhidden=NHDN;
 for(j=0;j<nout;j++){ out[j] =0.0; net = (float)0.0;

if(nhidden>50) ked = 60; //은닉층 50
 else if(nhidden>40) ked = 50;  else if(nhidden>30) ked = 40;
 else if(nhidden>20) ked = 30; ked = 50; for (k=0;k<=ked;k++)
{net+=hdnout[k]*wpt[k];              } wpt += nhidden+1;
    out[j] = 1.0/(1.0+exp(-net));  }
 max=-10000.;
 for(j=0;j<nout;j++)  {if(max<out[j])  {max=out[j];  winner=j; }  }
    if (( type!=0 && f==1) ) {     switch(type)
{           case 1: charset= han1code; break;
          case 2: charset= han2code; break;
          case 3: charset= han3code; break;
          case 4: charset= han4code; break;
          case 5: charset= han5code; break;
          case 6: charset= han6code; break;
          case 7: charset= outdata; break; } return(winner); }

void  in_layer_forward(int type)  //입력층
{  int h,nhidden;  float net,sum1,sum2,sum3,*wpt; int k;
    nhidden=NHDN;
    for(h=0;h<nhidden;h++)
    { net =0.0; wpt= hiwgt[h];
        for (k=0;k<NIN;k++)
        { net += wpt[k] * inout[k]; }
        net += 9.0*wpt[256];
        net /= (float)9.0;  wpt += NIN+1;
       hdnout[h] = (float)(1.0/(1.0+exp(-net)));  }
    hdnout[nhidden]=(float)1.0; }
int forward(int type)
{   int index;
    in_layer_forward(type);
    index=out_layer_forward(type);
    return(index); }
```

(그림 14) 교통표지판 인식 알고리즘

현에서 은닉층의 노드수를 50으로 설정하였다. 초기 연결 가중치는 1보다 작은 소수점 셋째 자리까지의 랜덤한 수로 정해지고, 역치와 학습률은 0.1로 설정 하였다. 같은 방법으로 교통표지판 16종 이외의 표지판도 약간의 학습과정을 통하면 인식할 수 있다.

이와 같이 교통표지판의 영역을 추출하고 추출된 교통표지판을 인식하여 어떤 종류의 표지판인지 판별하는 시스템으로 운전자뿐만 아니라, 컴퓨터나 텔레메틱스 등에서 응용이 가능한 시스템을 구축하였다.

4. 실험 및 결과

4.1 실험진행 단계

본 실험은 (그림 15)과 같이 7단계로 진행된다. (1)영상입력은 이미지 읽기를 수행하고, 입력된 이미지의 영상 밝기 구성이나, 명암대비 등의 정보를 분석하고 분석된 정보와

(1) 입력영상

(2) 평활화 영상

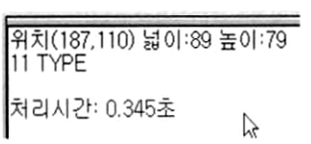
(3) 노랑→흰색 변환

(4) 빨강→검정 변환

(5) 검정→검정 변환

(6) 이진 영상

(7) 추출된 영상



(8) 인식결과

(그림 15) 교통표지판 인식 시스템

RGB 영상을 Gray 영상으로 변환하여 히스토그램 값을 구하고 그 값을 이용하여 임계값을 구한다. 영상의 화질 개선을 위한 히스토그램 평활화는 RGB 영상을 YUV 영상으로 변환 후 Y성분을 이용하여 수행한다. (2)는 평활화를 수행하는 과정이다. 평활화가 끝나면 컬러를 검색하여 주의표지판의 컬러 특성에 의해 노란색을 흰색으로 변환한다. 노란색을 흰색으로 변환은 YUV 영상으로 변환한 후에 V 값을 이용하여 변환한다. (3)은 노란색을 흰색으로 변환 하는 과정이다. 다음 단계는 (4)와 같이 적색을 검정색으로 변환하는 과정으로 교통표지판의 테두리를 검정색으로 변환한다. 이 단계는 YIQ 영상으로 변환 I 값을 이용하여 변환한다. (5)는 검정색을 검정색으로 변환하는 단계로 교통표지판의 내부 기호는 시각적으로 검정색처럼 보이지만 실제로는 검정에 가까운 색으로 RGB 값이 0이 되는 검정색으로 변환되어야 한다. 변환은 CMYK 영상으로 변환한 후에 C(Cyan) 값이 임계값과 비교하여 임계값보다 크면 검정색으로 변환

〈표 1〉 인식 속도

영상 크기(픽셀)	소요시간 (sec)
2048 × 1360	6.156
1024 × 680	1.5
720 × 478	0.782
640 × 425	0.609
480 × 318	0.344
400 × 265	0.235
320 × 212	0.156

한다. 임계값은 150으로 설정하였다. (6)은 이진화 단계로 이미지 추출의 정밀성과 신속성을 위해서 1 bit 영상으로 변환하는 과정이다. 이진영상은 RGB 값이 0 인 픽셀은 검정색으로 변환하고 그 외의 RGB 값은 모두 흰색으로 변환한다. 교통표지판 추출단계는 이진영상에서 표지판 영역을 라벨링 알고리즘을 적용하여 위치를 추출하고, 크기를 5:5 비율로 후보영역을 설정하면 교통표지판이 추출된다. 인식단계는 교통표지판의 256개의 특징점 추출하고 16종의 표지판을 학습 시켜 인식을 위한 준비 과정을 수행한다. 학습된 자료를 바탕으로 교통표지판을 type 0부터 type 15까지 16종류의 type으로 인식한다. (7)과 (8)은 교통표지판 추출과 인식된 결과이다.

4.2 실험결과

구축된 교통표지판 인식 시스템을 이용하여 주의표지판 8종과 규제표지판 8종의 영상 190개 이용하여 교통표지판 추출과 인식 실험을 수행하였다. 16종의 교통표지판은 <표 2>과 같이 type 0부터 type 15까지 분류하여 인식을 위한 학습 데이터로 사용하였다. 나머지 지시표지판과 보조표지판은 2가지 컬러로 구성 되어 있기 때문에 HSI 컬러공간으로 변환하여 이진 영상을 쉽게 생성할 수 있으므로 본 논문의 실험에서 제외 했다. 실험은 CPU Intel Pentium IV 2.8Ghz, Memory 1GB, WindowsXP의 환경에서 Visual C++ 6.0으로 구축하여 실험하였다.

실험결과에 총 190종의 실제 교통표지판을 디지털 카메라로 촬영하여 획득한 영상을 크기 별로 실험하여 인식속도를 측정하였다. 원 영상은 2240*1488 픽셀이며 JPG 영상이다. 이 영상을 BMP 영상으로 변환하고 <표 1>과 같이 여러 가지 이미지 크기로 변환하여 처리 시간을 측정하였다. 실험 결과에 의하면 480*318 픽셀이 최적의 영상이다. 영상의 크기가 너무 크면 인식 속도가 저하되고 영상크기가 작으면 인식 속도는 빠르지만 인식하는데 오류가 발생할 가능성이 높다. 인식률은 <표 2>에 의하면 인식률은 95.5%로 인식률이 매우 높다. 실제 인식 실험에 사용한 입력 영상은 13종이었다. 또한 다양한 영상을 입력 자료영상으로 사용하여 실험을 수행한 결과는 다음과 같다. (그림 16)은 교통표지판이 겹쳐있으나 형태가 완전하지 않은 표지판은 추출과 인식을 하지 않고 완전한 형태를 갖는 교통표지판만 추출하여

〈표 2〉 교통표지판 학습을 위한 분류 및 인식 결과

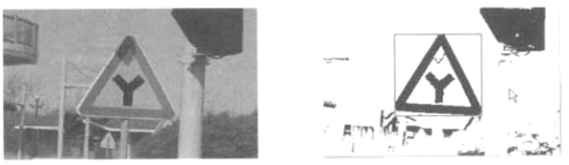
구분	분 류	표지판	표지판 명	실험	인식
규제표지판	type0		통행금지	0	0
	type1		승용차통행금지	0	0
	type2		화물차통행금지	15	14
	type3		이륜차통행금지	15	15
	type4		유턴금지	0	0
	type5		직진금지	15	15
	type6		우회전금지	15	14
	type7		좌회전금지	15	13
주의표지판	type8		+형교차로	16	16
	type9		ㅓ형교차로	15	14
	type10		ㅏ형교차로	15	14
	type11		T자형교차로	16	16
	type12		Y자형교차로	15	15
	type13		과속방지턱	18	18
	type14		우합류도로	15	14
	type15		좌합류도로	15	13
합 계				185	178

인식한 결과이다. (그림 17)은 잡음이 많은 영상에서 인식한 결과이다. (그림 18)는 교통표지판이 2개인 영상에서 인식한 것이다. (그림 19)는 역광 또는 어두운 영상에서 교통표지판을 인식한 것이다. (그림 20)은 위치와 크기가 같지 않는 2개의 교통표지판을 인식한 영상이다. 본 실험에서 인식률뿐만 아니라 입력영상이 크기와 상관없이 인식 되었다.

4.3 실험 결과 분석

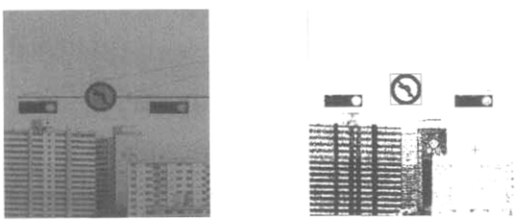
본 실험에서 도로에 설치되어 있는 교통표지판을 대상으로 실험 자료를 만들었다. 실제 교통표지판을 촬영하여 사용하였기 때문에 주위의 환경에 영향을 받을 수 있다. 또한 야간에 촬영된 영상은 조명의 영향을 받을 수 있다. 이런 문제점들을 히스토그램을 평활화 단계를 거쳐 인식률이 높은 이미지로 변환하여 문제를 해결하였다. 학습은 <표 2>와 같은 교통표지판 이용했다. 학습데이터를 생성하는데 수분이 소요되었다. 이 과정은 미리 학습함으로써 인식 속도에 영향을 주지 못한다. 실제 교통표지판을 취득하지 못한 영상은 실험에서 제외하였다. 특히 좌우가 대칭인 교통표지판은 인식률이 낮은 결과를 보였다.

(그림 16) 표지판이 겹쳐있는 영상

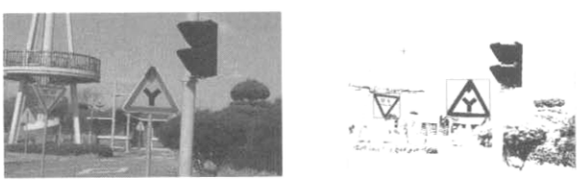


(그림 17) 잡영이 많은 영상

(그림 18) 표지판이 2개인 영상



(그림 19) 어두운 영상



(그림 20) 위치와 크기가 같지 않는 2개의 표지판영상

5. 결 론

먼저 본 논문에서 교통표지판 인식을 위하여 접근하는 방법은 인간의 시각의 기본적인 요소인 컬러정보를 이용하여 교통표지판 영역을 추출하는 방법이다. 입력된 교통표지판 영상에서 교통표지판을 추출하기 위한 컬러 분석은 교통표지판이 가지고 있는 컬러의 특성을 이용하였다. 교통표지판의 컬러 구성은 적색, 노란색, 검정색, 흰색의 4가지 컬러로 되어있다는 것에 착안해 컬러를 검색하여 노란색은 흰색으로, 적색은 검정색으로, 검정색은 RGB 값이 0인 검정색으로 변환하였다. 이진영상을 생성하기 위해서 픽셀의 RGB 값이 0 이외의 값을 가진 픽셀을 모두 흰색으로 변환하면 잡영이 제거된 이진영상이 생성되며 소요시간도 단축되었다.

두 번째 제안한 알고리즘은 기존의 교통표지판 인식시스템의 문제점을 해결하였다. 이와 관련된 기존의 연구에서 표지판영역을 추출하는 경우에 0.5초 이상 소요되고, 영상을 인식하는 경우에 10초 이상 소요되었다. 교통표지판 영역의 크기가 너무 작은 경우에 코너점 검출 마스크를 이용한 코너점 검출이 불가능하고, 컬러 성분의 밝기값의 변화가 많은 경우에도 정확한 교통표지판 영역 검출이 불가능하다. 입력 영상을 128×128픽셀의 크기로 정규화하는 전처리과정이 필요하므로 처리 시간이 길어지고, 입력영상이 너무 작은 경우에 크기 정규화를 위해 이용한 선형보간법은 많은 정보의 손실을 유발하여 인식률이 저하되고, 실영상에서도 인식률이 저하된다. 본 연구에서 관련된 연구에 비해 향상된 연구결과는 전처리 과정의 하나인 이진영상 생성이 간소화되어, 처리속도가 0.35초 이내로 향상된 것이다. 또한 입력영상의 크기에 구애받지 않으므로 형태정규화를 하지 않고 교통표지판을 추출하고 인식할 수 있었다. 다양한 컬러 모델을 이용하여 잡음 영상의 제거에 뛰어난 성능을 보였다.

향후 연구로 교통표지판 인식시스템의 실용화를 위하여 자동차가 도로 주행 중 비디오카메라로 촬영된 동영상에서 교통표지판이 포함된 프레임을 추적하여 프레임을 추출하는 과정이 요구된다. 또한 잡영에 보다 영향을 받지 않는 향상된 시스템을 위해서 계속 연구할 계획이다.

참 고 문 헌

[1] 곽현욱, 이우범, 김욱현, "컬러비와 거리비를 이용한 교통표지판 영역추출," 정보처리학회논문, pp.681-688, october, 2002.
[2] 경찰청, "교통안전 표지판 일람표" 도로교통안전협회, 내부부령 제651호, 7. 1., 1995.
[3] 김희승 "영상인식" 생능출판사, pp.15-180, 1998.
[4] 노명철, 최영우, 이성환, "자연 영상에 대한 비디오프레임에서의 자연 텍스트 추출," 컴퓨터비젼 및 패턴인식 연구회 추계워크샵, pp.161-162, 2001, 11.
[5] 박환범, 김희승, "Walsh 함수 모델을 이용한 교통안전표지 인식," 산업기술연구소 논문집 11권, pp.45-50, 2003.
[6] 오준택, 곽현욱, 김욱현, "웨이블릿 변환과 형태 정 보를 이용한 교통표지판 인식," 전자공학회논문지 제41권 제5호, pp.125-134, september, 2004.
[7] 임소영, 최태영, "불변 모멘트법을 이용한 도로 교통 표지판 인식," 제9회 신호처리 합동학술대회 논문집 제9호 1권, pp.593-596, 1999.
[8] 최형일, 이근수, 이양원, "영상처리 이론과 실제," 홍릉과학출판사, pp. 15-20, 1999.
[9] 하영호, 임재권, 남재열, 김용석, "디지털 영상처리," 서울 그린, pp.7-675. 1998.
[10] 한학용, "패턴인식 개론," 한빛미디어, 2005.
[11] H. D. Cheng, K. H. Jiang, Y. Sun, and J. Wang, "Color Image Segmentation : Advances and Prospects," Pattern Recongnition,

Vol.34, pp. 2259-2281, 2001.

[12] J. L. Mcclelland and D.E. Rumelhart, "Learning Internal Representation by Error Propagation," Parallel Distributed Processing, Vol.1, 1986.

방 걸 원

e-mail : bgw@bgcom.co.kr

1987년 광주대학교 전자계산학과(학사)

2002년 전남대학교 소프트웨어공학협동 (공학석사)

2006년 전남대학교 소프트웨어공학협동 (박사수료)

2001년~현 재 (주)위너테크 기술연구소 소장

관심분야 : 패턴인식, 멀티미디어컨텐츠, 컴퓨터비전

강 대 욱

e-mail : dwkang@chonnam.ac.kr

1982년 전남대학교 계산통계학과(이학사)

1985년 전남대학교 대학원 계산통계학과 (이학석사)

2000년 영국 Newcastle University 박사과정 수료

1984~현 재 전남대학교 컴퓨터정보학부 교수

관심분야 : Mobile Computing, distributed Computing, Network, Mobile Agent

조 완 현

e-mail : whcho@chonnam.ac.kr

1977년 전남대학교 수학교육과(학사)

1981년 전남대학교 수학과(이학석사)

1988년 고려대학교 통계학과(이학박사)

1983년~현 재 전남대학교 통계학과 교수

2006년~현 재 전남대학교 자연대학 학장

관심분야 : 패턴인식, 데이터마이닝, 인공지능