JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 21, No. 5, pp. 1009-1016, May 2020

Check for updates

라즈베리 파이를 이용한 차량 추적 시스템에 관한 연구

송 주 환
전주대학교 스마트미디어학과 교수

A Study on Vehicle Tracking System using Raspberry Pi

Ju-Whan Song
Professor, Department of Smart Media, Jeonju University, Jeonju 55069, Korea

[요 약]

본 연구는 수배된 차량을 자동으로 찾아서 차량의 위치와 사진 등을 서버에 전송하는 시스템을 구현하였다. 우리는 라즈베리 파이를 사용하여 차량 수배 시스템을 만들었고 이를 이용하여 수배된 차량을 매우 빠르게 찾을 수 있을 것이다. 이는 향후 차량의 블랙박스에 수배 차량 추적 기능을 추가하면 수배 차량을 찾는 데 도움이 된다. 제안된 시스템은 차량 수배 서버에서 수배 차량 번호 목록을 내려받아 해당 차량을 실시간으로 찾고, 찾았다면 차량 번호, 사진, 위치정보 등을 서버에 전송하는 과정으로 수행된다. 카메라에서 촬영된 영상에서 차량 번호판을 찾는 부분은 OpenCV를 이용한 영상처리 기법을 사용하여 처리하였다. 번호판에서 차량 번호를 찾는 부분은 영상을 전처리한 후 tesseract 라이브러리를 이용하였다. 차량 번호판 인식과 번호 인식은 각각 96%, 64%의 인식률로 좋은 결과를 가져왔다.

[Abstract]

This study implemented a system that automatically finds the wanted vehicle and sends the vehicle's location and photos to the server. We used Raspberry Pi to create a system to find the wanted vehicle, and we can find the wanted vehicle very quickly. In the future, we will add it to the vehicle's black box so that we can find the wanted vehicle. The proposed system is performed by downloading the list of wanted vehicle numbers from the vehicle wanted server, finding the vehicle in real time, and if it is found, sending the vehicle number, photo, location information, etc. to the server. Finding the vehicle license plate from the image captured by the camera was processed using an image processing technique using OpenCV. To find the vehicle number on the license plate, the tesseract library was used after preprocessing the image. The license plate recognition and number recognition resulted in good results with recognition rates of 96% and 64%, respectively.

색인어 : 사물 인터넷, 번호판 인식, 라즈베리 파이, OpenCV, 차량 위치 추적

Key word : IoT(Internet of Things), License Plate Recognition, Raspberry Pi, OpenCV, Vehicle Location Tracking

http://dx.doi.org/10.9728/dcs.2020.21.5.1009

Received 20 April 2020; **Revised** 15 May 2020
Accepted 25 May 2020

***Corresponding Author; Ju-Whan Song**

Tel: +82-63-220-2912
E-mail: jwsong@jj.ac.kr

Ⅰ. 서 론

국정 모니터링지표 e-나라 지표에 따르면 2019년 대한민국에 등록된 차량은 2367만 7천 대로 나타난다[1], 국민 두 사람당 1대의 차량이 운행되고 있다는 의미다. 차량도 많고 인구도 많다 보니 사건 사고가 자주 발생한다. 그중 차 사고를 낸 후 도주를 하거나, 또는 범죄에 사용된 차량을 수배하는 경우를 많이 볼 수 있다.

이를 위하여 차량 번호판 인식 기술이 필요하다. 차량 번호판 인식 시스템은 2011년 5월 행정안전부의 'CCTV 종합대책' 발표 이후 CCTV 시장이 활성화되어 지자체별로 다양한 분야에서 적용되고 있으며, 문제 차량 자동 감지, 수배 차량 추적, 주정차 단속 등에 관한 기술이 꾸준히 개발되고 있다[2].

본 논문에서는 수배된 차량을 자동으로 찾아서 차량의 위치와 사진 등을 서버에 전송하는 시스템을 구현하는데 목표를 두었다. 수배된 차량을 찾는 방법으로 사람이 직접 찾는 대신에 사진을 찍어 사진에서 차량의 번호판을 인식하여 수배 차량 여부를 판단하는 차량 수배 시스템이 필요하다. 우리나라의 대부분 차량에 블랙박스가 설치되어있는 현실이고, 이 블랙박스를 이용하여 수배된 차량을 추적할 수 있는 시스템을 추가한다면 매우 빨리 수배 차량을 찾을 수 있을 것이다. 이를 위해 본 논문에서는 라즈베리 파이를 이용하여 차량을 찾을 수 있는 블랙박스를 구성해보았다.

차량 수배 시스템은 라즈베리 파이에 카메라를 설치하였고, 카메라로 촬영된 영상을 다양한 영상처리를 통하여 번호판을 추출하고, 번호판에서 다시 차량 번호를 추출하는 과정으로 구성된다. 이 과정에서의 프로그램은 OpenCV와 파이썬을 사용하였다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 본 논문에서 구현한 차량 번호 인식 시스템에 필요한 차량 번호판과 OpenCV에 대해 기술하고, Ⅲ장에서는 시스템의 구성과 차량 인식 알고리즘의 동작 과정을 설명하고, Ⅳ장에서는 시스템의 실험 및 평가결과를 나타내고, Ⅴ장에서는 결론 및 향후 연구 방향을 서술하였다.

Ⅱ. 관련 연구

2-1 대한민국 차량 번호판

대한민국의 번호판은 1946년, 1950년, 1968년에 개정되었고 각각의 번호판은 흰색 바탕에 검은 글씨로 이루어졌다. 1973년 개정판에서는 자가용, 영업용, 전세용으로 나누어 녹색 바탕에 흰색 글씨, 흰색 바탕에 녹색 글씨, 그리고 노랑 바탕에 흰색 글씨로 2줄로 이루어졌으나, 자가용과 렌터카는 2006년 개정판부터 흰색 바탕에 검은 글씨로 1줄로 구성이 되었다. 2019년엔 차종 기호를 숫자 3자리로 확장하여 현재는 다양한 번호판들이 이용되고 있다[3]. 번호판의 크기는 52cm × 11cm이고, 각 숫자 및 문자는 5cm × 7.5cm 안에 포함된다. 인접한 두 숫자의 중심 간격은 5.4cm이다[4]. 그림 1에 2003년 이후 대한민국에서 사용하는 차량 번호판의 몇 가지 예를 나타냈다.

그림 1. 2003년 이후 대한민국 차량 번호판 예시
Fig. 1. Example of vehicle license plate in Korea since 2003

자동차 번호판의 숫자 두 자리는 자동차의 종류를, 한글 글자는 자동차의 용도를, 마지막 숫자 네 자리는 등록 차량의 일련번호를 뜻한다. 한글 글자 1글자로 구성되며 군용과 외교용 등을 제외하고 비사업용과 영업용으로 나누어서 지정되어 있다[5]. 총 40개의 글자가 존재하고 받침이 없는 한글을 사용한다. 표 1에 번호판에서 사용되는 한글을 나타냈다.

표 1. 번호판에서 사용하는 한글
Table 1. Text used in license plates

분류		기호
비사업용(관용포함)		가, 나, 다, 라, 마, 거, 너, 더, 러, 머, 버, 서, 어, 저, 고, 노, 도, 로, 모, 보, 소, 오, 조, 구, 누, 두, 루, 무, 부, 수, 우, 주
영업용	일반	아, 바, 사, 자
	택배	배
	렌터카	하, 허, 호

2-2 영상처리와 OpenCV

OpenCV는 공개 소프트웨어로 누구나 자유로운 수정, 변경, 재배포가 가능하고 영상처리와 컴퓨터 비전 분야에서 많이 사용되고 있다. 초기에는 C/C++로 개발되었고, 윈도우즈와 리눅스, 맥 등의 운영체제에서 사용할 수 있다[6]. 프로그램의 호환성이 좋아 C/C++, C#, Python 등에서 사용할 수 있다.

OpenCV의 주요 라이브러리 기능은 CXCORE, CV, HIGHGUI, ML, CVAUX 등이 있으며 CV 라이브러리에 영상처리 및 컴퓨터 비전에 관련된 많은 함수가 있어 영상의 특징 추출, 필터링, 영상 분할, 움직임 분석, 물체추적, 패턴인식, 카메라 캘리브레이션, 3D 복원 등을 할 수 있도록 한다. 영상처리 관련 함수들을 직접 구현하기는 매우 어렵지만 이를 OpenCV 라이브러리에 구현된 함수를 사용하면 간단히 영상처리를 할 수 있어 매우 편리하다.

본 논문에서는 영상의 색상 모델을 변경하고, 영상을 색상별

로 분할, 가우스 필터링, 캐니 에지 검출, 윤곽선 추출, 결과 출력 등에서 OpenCV 라이브러리 함수를 사용하여 매우 편하게 연구를 할 수 있었다.

Ⅲ. 시스템 설계 및 구현

3-1 시스템 구성 및 개발환경

본 논문을 통해 제안하고자 하는 시스템의 구조는 그림 2와 같다.

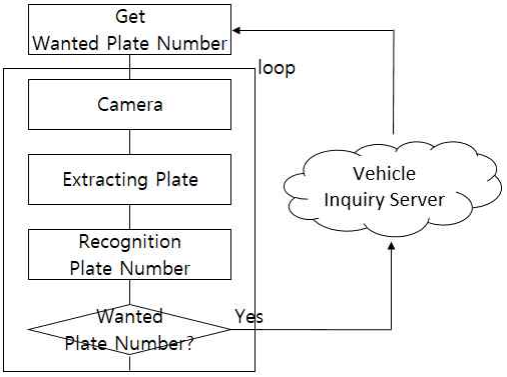


그림 2. 시스템 구조도
Fig. 2. System framework

시스템을 운영하기 위해서 첫 번째 라즈베리 파이에는 운영체제 라즈비안을 설치하고, 원격 접속이 가능하도록 ssh 데몬을 설치한다. 스마트폰을 핫 스폿(hot spot)으로 인터넷에 접속할 수 있도록 와이파이를 설정한다. 그리고 차량 수배정보를 배포하고 수배된 차량에 대한 정보를 받을 수 있는 서버를 Vehicle Inquiry Server라는 이름으로 구축하였다.

라즈베리 파이에 전원이 들어와서 자동으로 부팅되면 논문에서 구현한 차량 수배 시스템이 실행되고, 인터넷을 통하여 서버에서 수배된 차량 번호를 내려받는다. 라즈베리 파이는 5초 간격으로 사진을 찍고 그 사진에서 차량 번호판을 찾고 사진에 차량 번호판이 포함되어 있으면 차량 번호판 부분만 잘라내기를 한다. 잘라내기 한 번호판 사진에서 차량 번호와 문자를 인식한 후 수배 차량 번호이면 이 번호를 차량의 GPS(global positioning system) 위치정보와 사진, 시간 정보 등을 서버로 보내도록 구성되었다.

3-2 차량 추적 서버 구축

차량 추적 시스템 구축을 위해서는 라즈베리 파이와 인터넷에 연결되는 스마트폰, 인터넷에 연결된 서버가 필요하다. 라즈베리 파이는 스마트폰의 핫 스폿을 통하여 서버에 접속하여 수배 차량에 대한 정보를 받고, 카메라와 GPS 정보를 핫 스폿을 통하여 서버에게 전송하는 구조로 되어있다. 이를 그림 3에 나타냈다.

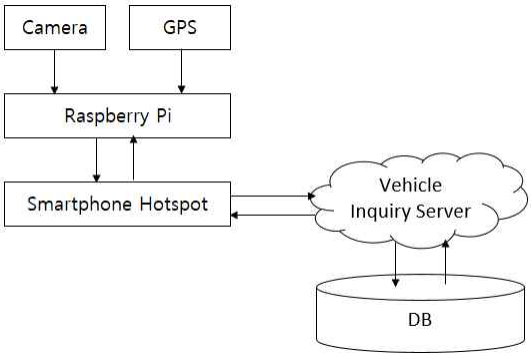


그림 3. 네트워크 구조
Fig. 3. Network diagram

1) Vehicle Inquire Server 구축

라즈베리 파이와 서버 간 통신은 중간에 스마트폰 핫 스폿을 통하여 이루어지고, 둘 사이의 연결은 HTTP 프로토콜을 사용하였다. 실시간으로 연결되어 있을 필요가 없고 필요할 때 잠깐 접속해서 데이터를 전송받거나 전송하기 때문에 웹프로그램으로 서버를 구축하여 사용하는 것이 좋다.

안정적으로 인터넷에 연결되어 있어야 하므로 직접 서버를 운영하는 대신 웹 호스팅 업체에서 1개의 호스트를 운영하여 구축하였다. 서버의 운영환경은 Linux 64bit 운영체제를 사용하고, 사용 프로그램으로는 웹서버로는 Apache 2를, 언어는 UTF-8, 데이터베이스는 MariaDB 10.X 버전이 설치되어있다.

본 논문에서 필요한 서버 프로그램은 3개의 역할에 필요한 부분만 개발하였다. 첫 번째는 수배 차량 정보를 관리하는 프로그램, 두 번째로는 라즈베리 파이가 접속하면 현재 등록되어있는 수배 차량 목록을 라즈베리 파이에 전송해주는 프로그램, 마지막으로는 라즈베리 파이에서 수배 차량을 찾았을 때 차량에 대한 정보와 영상을 전송받아서 처리하는 프로그램을 만들었다.

2) 수배 차량 목록 배포

서버에 있는 데이터베이스인 MariaDB에는 수배 차량 정보를 저장할 wanted_vehicle이란 이름의 테이블을 만들었다. 이 테이블의 구성은 표 2에 나타냈다. 실제로는 더 많은 필드가 있으나 최소 필요한 것들만 표현한 것이다.

라즈베리 파이가 실행되면 가장 먼저 웹을 통하여 Vehicle Inquire Server의 GetWantedList.php에 접속한다. 이 프로그램은 wanted_vehicle 테이블에 저장된 수배 차량 번호 목록을 보내주도록 하였다. 이때 데이터는 JSON 포맷으로 변환하여 전송한

표 2. wanted_vehicle 테이블
Table 2. wanted_vehicle table

field	type	memo
_id	int(11)	AUTO_INCREMENT
plate_no	varchar(10)	car plate number
etc	text	memo
insert_date	datetime	insert time

다. 라즈베리 파이는 이 번호 목록을 등록하여 차량을 찾게 된다.

3) 찾은 차량 정보 획득

라즈베리 파이는 찾은 차량에 대한 정보를 서버의 SendTrackedLocation.php에 접속하여 전송하고 전송받은 서버는 DB의 found_vehicle 테이블에 저장하도록 프로그램을 만들었다. 라즈베리 파이에서도 데이터는 JSON 포맷으로 변환하여 전송한다. found_vehicle 테이블의 구성은 표 3에 나타냈다.

표 3. found_vehicle 테이블
Table 3. found_vehicle table

field	type	memo
client_id	varchar(100)	
plate_no	varchar(10)	car plate number
img_name	varchar(100)	car image name
lat	float	latitude
lng	float	longitude
insert_date	datetime	insert time

라즈베리 파이에서는 차량 정보와 차량 영상을 전송하는데 차량 영상은 별도의 디렉터리에 저장하고, 이 영상의 저장된 파일명을 img_name 필드에 저장한다.

3-3 차량 추적 클라이언트 구축

차량 추적 클라이언트는 라즈베리 파이를 이용하여 구축하였고, 사용 프로그래밍 언어는 파이썬이다. 라즈베리 파이에 GPS와 카메라를 추가 장착하였고, 스마트폰의 핫 스폿에 와이파이 연결된 상태로 운영된다. 라즈베리 파이가 처음 시작되면 서버에 접속해서 수배 차량 목록을 JSON으로 받아와서 이를 파싱하여 차량 목록을 얻는다.

차량이 운행하는 동안 5초 간격으로 사진을 찍고, 사진에 차량 번호가 포함된 것으로 판단되면 차량 번호를 인식한다. 인식된 차량 번호가 수배 차량 목록에 포함되어 있으면 이 정보를 서버에 전송한다. 추출된 차량의 번호와 처음 찍었던 차량의 사진, 라즈베리 파이에 연결된 GPS로부터 얻은 위치정보, 그리고 현재 시각을 서버에 전송한다. 차량 번호 인식은 3-4와 3-5에서 설명한다.

3-4 라즈베리 파이에서 차량 번호 인식

라즈베리 파이는 5초 간격으로 사진을 촬영하여 그 사진에서 차량 번호판을 추출하는 부분이다. 이 부분의 처리 과정은 그림 4와 같다. 카메라에서 촬영된 영상을 받아 색상 모델을 변환하고 가우스 필터링(Gaussian filtering), 캐니 에지(Canny edge) 검출을 거쳐 윤곽선을 추출(find contours)한 뒤 윤곽선을 이용하여 번호판 영역에 대한 정보를 얻는다. 처음 영상에서 번호판 영역을 찾고 이를 회전 변환하여 잘라내기를 하면 차량 번호판 영역에 대한 영상이 만들어진다.

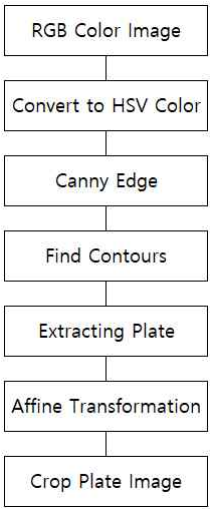


그림 4. 번호판 추출 알고리즘
Fig. 4. Extracting vehicle plate

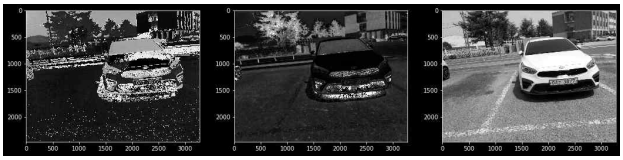
1) HSV 컬러 영상으로 변환

라즈베리 파이에서 촬영된 영상은 RGB 색상으로 표현된다. 번호판 추출하기 위해서는 RGB 색상보다는 회색 음영(gray scale color) 또는 HSV 색상이 더 효율적이기 때문에 본 논문에서는 HSV 색상으로 변환해서 사용한다. HSV는 색상을 나타내는 H(hue)와 채도를 나타내는 S(saturation)와 밝기를 나타내는 V(value)로 구성되어 있고, 이중 V는 회색 음영으로 사용할 수 있다[7]. H는 0~360도의 각도로 표현되는데 1바이트에 값 360을 저장할 수 없어 일반적으로 0~180 사이의 정수로 표현하고, S와 V는 0~255 사이의 정수로 표현한다. 색상의 변환은 직접 계산하지 않고 OpenCV의 함수 cvtColor()를 사용하면 쉽게 변환되고, 이를 split() 함수를 사용하면 H와 S, V값을 각각의 영상으로 나누어준다. 그림 5에 RGB 영상과 HSV로 변환한 영상을 나타냈다.

2) 캐니 에지(Canny edge)

영상에서 에지는 어떤 객체의 가장 바깥의 둘레를 의미하며

(a) RGB Color



(b) HSV color(hue, saturation, value)

그림 5. RGB 색상을 HSV 색상으로 변환
Fig. 5. RGB color to HSV color

객체의 테두리로 볼 수 있다. 에지는 영상의 객체들을 구분하는 지점이며 픽셀의 밝기가 작은 값에서 큰 값으로 또는 큰 값에서 작은 값으로 변하는 지점을 말한다. 즉 픽셀의 밝기 변화율이 높은 지점이다. 에지를 찾기 위해서는 미분과 기울기 연산을 수행하며 픽셀의 밝기 변화율이 높은 경계선을 찾는다. 에지 검출은 주로 1차 미분이나 2차 미분을 이용해서 구하는데, 잡음(noise)에 대한 영향이 커서 흐림 효과(blurring)를 진행하여 잡음의 영향을 줄인 후 검출한다. 대표적인 방법으로는 소벨 미분(Sobel derivative), 샤로 필터(Scharr filter), 라플라시안(Laplacian), 캐니 에지(Canny edge) 등이 있다.

캐니 에지는 라플라스 필터 방식을 캐니(J. Canny)가 개선한 방식으로 x 와 y에 대해 1차 미분을 한 다음, 네 방향으로 미분하는 것이다. 네 방향으로 미분한 결과로 극댓값을 갖는 지점들이 에지가 되고, 이 알고리즘은 다른 에지 검출기보다 성능이 월등히 좋으며 잡음에 민감하지 않아 강한 에지를 검출할 수 있다. 캐니 에지 알고리즘의 동작은 다음 네 가지 순서로 진행된다.

① 잡음 제거를 위해 가우스 필터링하여 흐림 효과
② 기울기 값이 큰 지점을 검출
③ 최댓값이 아닌 픽셀을 0으로 변경하여 명백한 에지가 아닌 값 제거
④ 히스테리시스 임곗값(Hysteresis threshold)을 적용해 윤곽 생성

HSV로 변환된 영상에서 밝기 정보인 value의 영상을 사용하여 가우스 필터링하여 영상에 있는 잡음을 줄인다. 그림 6에 원본 value 영상과 가우스 필터링한 영상을 보인다.

가우스 필터링을 통과한 영상에 대해 기울기가 큰 값을 검출한다. 검출된 값이 매우 많아 전체를 에지로 판단하는 대신, 그 중 히스테리시스 임곗값을 적용해 일부만 추출한다. 히스테리시스 임곗값은 상위 임곗값과 하위 임곗값 2가지로 구성된다.

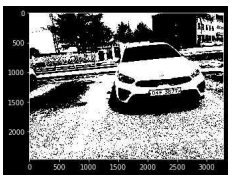
(a) Before Filtering

(b) After Filtering

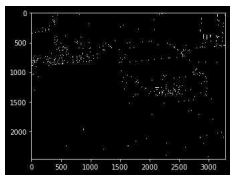
그림 6. 가우스 필터링 전과 후
Fig. 6. Before and after Gaussian filtering

상위 임곗값보다 낮으면서 하위 임곗값보다 높은 경우 에만 상위 임곗값에 연결된 에지로 판단한다.

일반적으로 이진화는 하나의 임곗값(threshold)을 영상 전체에 적용하여 임곗값보다 크면 흰색, 작으면 검은색으로 바뀌는 것이다. 이는 영상의 밝고 어두움을 임곗값을 기준으로 나누게 되는 결과가 될 뿐이다. 캐니 에지 알고리즘은 영상에서 에지로 판단되는 부분을 흰색으로 그렇지 않은 부분은 검은색으로 이진화한다. 그림 7에 HSV 색상의 밝기(value) 영상을 단순 이진화한 것과 캐니 에지를 적용한 결과를 보인다.



(b) Threshold



(b) Canny edge

그림 7. 이진화 비교
Fig. 7. Threshold vs. Canny edge

3) 윤곽선 검출(contour find)

앞에서 구한 결과는 전체 영상에서 각각의 에지들을 구한 것이지 이 에지들의 관계를 확인할 수 없다. 하나의 객체를 둘러싼 에지들을 모아 세그먼트로 구성하고 구성된 세그먼트들 중에서 번호판의 에지들로 구성된 것이 있으면 그 세그먼트의 영역을 번호판으로 판단할 수 있다. 윤곽선 검출 알고리즘은 캐니 에지의 결과 영상에서 에지로 검출된 픽셀을 대상으로 세그멘테이션(segmentation) 작업을 진행하는 것이다. 검출된 윤곽선은 형상의 분석과 물체 감지 및 인식에 가장 효과적인 방법 가운데 하나다.

본 논문에서는 윤곽선에 해당하는 에지들을 수평과 수직, 그리고 대각선 부분을 압축해서 각 끝점만 반환하는 방법을 사용하였다.

4) 번호판 추출

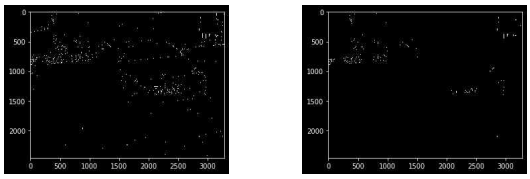
번호판 추출은 2단계로 작업을 한다. 1단계에서는 전체 윤곽

선 중 문자일 가능성이 있는 윤곽선만 남기고, 2단계에서는 남은 윤곽선 중 번호판 모양의 윤곽선만 찾는다.

먼저 1단계는 추출된 윤곽선에 대하여 각각의 윤곽선이 문자일 가능성이 있는 윤곽선만 선택하고 나머지는 모두 제거한다. 문자일 가능성을 판단하는 기준은 윤곽선을 둘러싼 바운딩 박스(bounding box)를 가지고 기준을 잡고, 그 기준은 다음과 같다.

① 바운딩 박스의 넓이가 기준값(예:100) 이상
② 바운딩 박스의 너비가 기준값(예:5) 이상
③ 바운딩 박스의 높이가 기준값(예:5) 이상
④ 바운딩 박스의 너비 높이 비율이 기준값(예:1.0) 이하

캐니 에지를 구한 결과에 윤곽선 검출한 결과를 그림 8에 나타냈다. 결과를 보면 매우 많았던 윤곽선에서 문자 후보가 되는 윤곽선만 남기고 나머지는 사라졌음을 알 수 있다.



(a) Canny edge (b) Possible characters

그림 8. 1단계 전과 후
Fig. 8. Before and after step 1

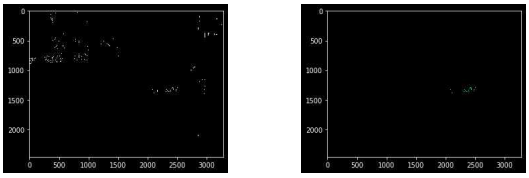
다음으로 2단계에서는 남아있는 윤곽선들, 즉 문자라고 할 수 있는 윤곽선들을 서로 조합하여 번호판처럼 여러 개의 문자의 조합인 문자열이 생성되는 문자열 후보를 추출하는 것이다. 이때 여러 개의 문자를 하나의 문자열로 묶을 수 있다고 판단하는 기준은 다음과 같다.

① 바운딩 박스의 넓이 차이가 기준값 이하
② 바운딩 박스의 높이 차이가 기준값 이하
③ 바운딩 박스의 너비 차이가 기준값 이하
④ 바운딩 박스 사이의 각의 차이가 기준값 이하
⑤ 바운딩 박스 사이의 거리 차이가 기준값 이하

문자 후보 윤곽선 중 번호판 문자열의 후보가 되는 윤곽선만을 남기고 모두 제거한 결과를 그림 9에 나타냈다. 결과를 보면 전체 윤곽선 중 한 곳에만 모여있는 윤곽선만 선택되어 1개의 번호판 후보가 나타났음을 알 수 있다

5) 아핀변환(affine transformation)

추출된 번호판은 직사각형 모양일 경우도 있지만, 대부분 차량의 위치와 방향에 따라 왜곡된 모양으로 나타난다. 왜곡된 번호판 영상을 아핀변환을 사용하여 직사각형으로 변환한다. 아



(a) Possible Characters (b) Possible Plate

그림 9. 2단계 전과 후
Fig. 9. Before and after step 2

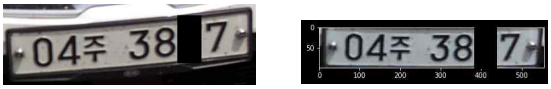
핀변환은 영상을 확대/축소(scaling), 평행이동(translation), 회전(rotation), 대칭(reflection), 밀림(shearing) 등을 의미하고,번호판 영상에는 회전과 밀림을 주로 사용한다. 아핀변환의 대표적인 변환 3개를 행렬식으로 표현하면 그림 10과 같다.

$$\begin{pmatrix}\cos\theta & -\sin g\theta & 0\\ \sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{pmatrix} \quad \begin{pmatrix}sx & 0 & 0\\ 0 & sy & 0\\ 0 & 0 & 1\end{pmatrix} \quad \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ t_x & t_y & 1\end{pmatrix}$$

(a) rotation (b) scaling (c) translation

그림 10. 아핀 변환 행렬
Fig. 10. Affine transformation

번호판 후보 영상들의 회전각과 중심좌표를 구하여 영상을 중심좌표로 이동 후 회전각만큼 회전한 뒤 원래 위치로 이동하면 정면에서 바라보는 번호판 영상이 생성된다. 그 후 워핑 변환을 통해 직사각형 모양의 번호판 영상을 찾게 된다. 그림 11에 차량 번호판의 영상과 아핀변환 후의 영상을 나타냈다. 개인정보보호를 위해 숫자 1자는 논문에 추가하면서 가린 것이다.



(a) Before transformation (b) After transformation

그림 11. 아핀변환 전과 후
Fig. 11. Before and after affine transformation

3-5 차량 번호 인식

추출된 차량 번호판의 문자와 숫자를 인식하는 부분이다. 차량 번호판의 문자와 숫자를 인식시키기 위해 tesseract 라이브러리를 사용하였다[8]. 문자와 숫자의 인식률을 높이기 위해서는 잡음을 제거하고 이진화 영상으로 변환하는 부분이 전처리로 수행되어야 한다. 이 부분의 처리 과정은 그림 12와 같다. 번호판 영역에 대한 영상을 가우스 필터링하고 이진화를 거쳐 번호를 인식하는 순서로 진행된다.

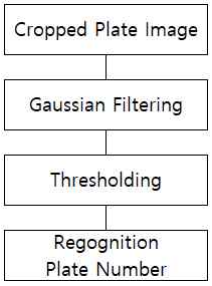


그림 12. 번호 인식 알고리즘
Fig. 12. Recognition algorithm

잡음을 제거하기 위해서 가우스 필터링을 적용하여 흐림 효과를 준 뒤 이진화 과정을 거친다. 이진화 과정에서는 오츠(Otsu) 알고리즘을 사용하는데 오츠 알고리즘은 영상의 밝기 분포를 통해 최적의 임곗값을 찾아 이진화를 적용하는 알고리즘이다. 이전 과정에서 번호판 영역만 추출하였으므로 번호판 영역에 대해 오츠 알고리즘으로 이진화하면 그림 13과 같은 영상이 나온다.

그림 13. 이진화 영상
Fig. 13. Thresholding image

tesseract 라이브러리를 사용하기 위하여 설정값으로 psm7을 설정하여 문자가 일자임을 지정해주고 번호판에는 문맥이 없으므로 oem 0을 설정하여 한 글자씩 인식하도록 지정하였다. 이 영상을 tesseract 라이브러리의 image_to_string() 함수를 사용하면 차량 번호를 돌려준다.

우리나라 번호판에서 가장 특징적이고 번호판의 종류에 영향을 받지 않는 것은 연속된 4개의 숫자가 존재한다는 점이다[9]. 결과가 차량 번호인지 판단하기 위하여 각각의 문자가 한글과 숫자로 이루어졌고, 한글 1글자가 3번째 또는 4번째에 있는지, 한글 뒤에 4개의 숫자가 있는지를 판단하여 그 조건에 만족하면 차량 번호로 인정한다. 위 영상의 인식결과는 04주38 7로 완벽하게 인식했다.

VI. 실험 및 평가

본 논문에서는 라즈베리 파이를 활용하여 실시간으로 수배 차량을 찾을 수 있는 시스템의 프로토타입을 만들어서 테스트해보았다. 시스템은 서버와 클라이언트로 구성되어 있고, 서버는 현재 상용으로 가입자를 받아 운영 중인 웹 호스팅 업체에 호스트를 운영하여 구축하였다. 서버의 운영환경은 Linux 64bit 운영체제를 사용하고, 사용 프로그램으로는 웹서버로는 Apache 2를, 언어는 UTF-8, 데이터베이스는 MariaDB 10.X 버전이 설치되어있다. 서버 프로그램은 PHP 7.2를 사용하여 코딩하였다.

클라이언트 시스템은 라즈베리 파이 3 b+에 구축하였고, 카메라는 800만 화소의 Raspberry Pi Camera Module V2를 사용하였고, Raspberry Pi GPS 모듈을 사용하였다. 라즈베리 파이에서의 프로그램은 파이썬3.7을 사용 하여 코딩하였다.

개발한 시스템을 차량 조수석에서 촬영하여 그중 자동차 번호판이 완전하게 보이는 영상 100장을 선택하여 실험하였고, 번호판의 인식률과 차량 번호의 인식률을 구하여 표 4에 나타냈다.

표 4. 인식률
Table 4. Recognition rate

	success rate
recognition vehicle plate area	96%
recognition vehicle plate number	64%

번호판 영역의 인식률은 매우 좋은 결과가 나왔지만, 차량 번호에 대한 인식률은 영상의 전처리에서 완전하게 잡음을 제거하지 못해서인지 좋은 결과가 나오지 않았다.

V. 결론

본 논문에서는 수배된 차량을 찾는 방법으로 사람이 직접 찾는 대신에 사진을 찍어 사진에서 차량의 번호판을 인식하여 수배 차량 여부를 판단하는 차량 추적 시스템을 제안하였다. 제안한 시스템을 차량의 블랙박스에 추가한다면 매우 빨리 수배 차량을 찾을 수 있을 것이다.

차량 수배 서버를 구축하여 수배 차량 목록을 받고 찾은 차량에 대한 정보를 보내는 것은 오류 없이 잘 실행되었고, 차량의 번호판 영역을 찾는 것은 96%의 인식률로 잘 찾았으나, 번호판의 숫자를 인식하는 부분에서는 64%의 인식률로 조금은 부족한 결과가 나왔다. 본 논문에서 제안한 방법은 향후 실용화가 가능함을 알 수 있다.

본 논문에서는 최신의 비사업용 차량에 대한 시스템을 개발하였는데, 추후 현재 운영되는 모든 차량에 대한 시스템으로 확장하겠다.

참고문헌

[1] e-National indicators, Car Registration Status [Internet]. Available: http://www.index.go.kr/potal/main/EachDtlPageDetail.do?id

x_cd=1257

[2] Y. B. Joo, J. Y. Bang, S. E. Oh, and K. M. Huh, Robust Vehicle License Plate Recognition System Using Reinforcement Learning, *Journal of Institute of Control, Robotics and Systems,* Vol. 24, No. 4, pp. 334-339, 2018.

[3] Namu Wiki. Car Plate Number/Korea [Internet]. Available: https://namu.wiki/w/%EC%B0%A8%EB%9F%89%20%EB%B2%88%ED%98%B8%ED%8C%90/%EB%8C%80%ED%95%9C%EB%AF%BC%EA%B5%AD

[4] T. Q. Hieu, S. Yeon, and J. Kim, Korean License Plate Recognition Using CNN, *Journal of Institute of Korean Electrical and Electronics Engineers,* Vol. 23, No. 4 pp. 1337-1342, Dec. 2019.

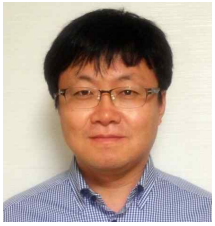
[5] Naver Knowledge Encyclopedia. Car registration plate [Internet]. Available: https://terms.naver.com/entry.nhn?docId=5702512&cid=43667&categoryId=43667

[6] S. S. Shim, A Study on Tracking System of Moving Objects Using an OpenCV Library, MS. dissertation, Mokpo National Maritime University, Mokpo, 2016

[7] J. W. Song, Content-based Image Retrieval Using HSV Color and Edge Orientation, *Journal of Korean Institute of Information Technology*, Vol. 16, No. 5, pp. 113-118, Nov. 2018.

[8] N. G. Lee and H. R. Kim, A Study on the Secure One way Local Car Number Recognition System Using Raspberry Pie under Network Separation Environment, *Journal of Digital Contents Society,* Vol. 19, No. 10, pp. 1909-1917, Oct. 2018.

[9] H. S. Choi, D. W. Kim, H. U. Chae, S. Y. Lee, and H. S. Choi, Implementation of real-time Vehicle Location Tracking System using a vehicle black box, *Journal of Digital Contents Society,* Vol. 20, No. 6, pp. 1087-1096, Jun. 2019.



송주환(Ju-Whan Song)

1997년 : 전북대학교 대학원 (이학석사)
2003년 : 전북대학교 대학원 (이학박사-컴퓨터그래픽스)

2006년～현 재: 전주대학교 스마트미디어학과 교수
2019년～현 재: 전주대학교 대학원 스마트 Agro_ICT 융합학과 교수
※관심분야 : 컴퓨터그래픽스, 영상처리, 사물인터넷(IoT), 디지털콘텐츠, 멀티미디어, 모바일 프로그래밍 등