KIPS Trans. Softw. and Data Eng.
Vol.8, No.10 pp.391~396 pISSN: 2287-5905

https://doi.org/10.3745/KTSDE.2019.8.10.391

File Database and Search Algorithm for Efficient Search of Car Number

Sim Chul Jun† · Yoo Sang Hyun†† · Kim Won Il†††

ABSTRACT

Researches for image processing have been actively progress due to the development of various hardware. For example, in order to prevent various types of crime by a vehicle, there is a method of detecting the location of a criminal vehicle using the existing CCTV in real time. However, certain types of systems and high-performance system requirements make it difficult to apply to existing equipment. In this paper proposes a search algorithm that construct a file database of Korean standard license plate information so that specific vehicles can be quickly searched using existing equipment. In order to evaluate the performance of the file database and the search algorithm proposed in this paper, we set up the search targets at various locations and the results showed that the search algorithm could always check the information by searching the vehicle within a certain time.

Keywords : Image Processing, Search Algorithm, File Database

차량번호의 효율적 탐색을 위한 파일 데이터베이스와 탐색 알고리즘

심 철 준† · 유 상 현†† · 김 원 일†††

요 약

다양한 하드웨어의 발전으로 영상처리를 이용한 연구가 활발하게 진행되고 있다. 예를 들어 차량에 의한 각종 범죄를 예방하기 위해 기존에 설치된 CCTV를 이용하여 실시간으로 범죄 차량의 위치를 탐색하여 단속하는 방법이 있다. 그러나 특정 시스템과 높은 시스템 요구 사항으로 인해 기존 장비들에 적용하기 어려운 문제점 있다. 본 논문에서는 이러한 문제점을 해결하기 위해 기존 장비들에서도 차량의 탐지를 수행할 수 있도록 한국 표준 차량 번호판 정보를 파일 데이터베이스로 구성하고, 특정 차량에 대한 정보를 빠르게 탐색할 수 있는 탐색 알고리즘을 제안하였다. 본고에서 제안한 파일 데이터베이스와 탐색 알고리즘의 성능 평가를 위해 다양한 위치에 탐색 대상을 설정하고 실험한 결과 탐색 알고리즘은 항상 특정 시간 안에 차량을 탐색하여 정보를 확인할 수 있었다.

키워드 : 영상처리, 탐색 알고리즘, 파일 데이터베이스

1. 서 론

최근 영상처리 연구에서는 이미지 처리를 통해 획득된 정보를 이용하여 범죄를 예방하거나, 현재 발생되고 있는 범죄 또는 사고를 인식하는 것으로 이미지 처리 정보의 활용 범위가 확대되고 있는 추세이다[1]. 특히 과속 차량 단속과 주차 관리 그리고 신호위반 단속 등에 적용할 수 있는 다양한 연구와 제품이 개발되고 있으며, 추가적으로 위치 정보를 이용한 범죄 차량 추적이나 탐지에 대한 연구가 활발하게 진행되고 있다[2-4].

최근에는 CCTV을 이용한 차량 위치 추적 시스템이 개발되었는데, 이 시스템은 범죄/도난/대포 차량 등의 불법 차량 번호를 중앙 관리 센터에 등록하면, 중앙 관리 센터에 연결된 CCTV들로부터 전송 받은 영상 분석을 통해 차량번호를 획득하고, 이 가운데 등록된 불법 차량을 판별하여 실시간 범죄 차량 추적 기능을 제공한다[5]. 이러한 실시간 차량 추적 기능은 각종 범죄 차량을 효율적으로 단속하고 관련 범죄율도 감소시키는 등 긍정적 영향을 미치는 것으로 나타났다[5]. 그러나 해당 시스템은 현재 경찰차에만 도입되었고, 높은 하드웨어 사양을 요구하기 때문에 기존 장비들을 활용할 수 없다는 문제점이 있다.

본 논문에서는 CCTV나 차량 블랙박스와 같은 열악한 기

† 정 회 원 : ㈜아이지코 연구소 소장
†† 비 회 원 : 경민대학교 융합소프트웨어학과 조교수
††† 비 회 원 : ㈜마크애니 기술연구소 책임연구원
Manuscript Received : May 14, 2019
First Revision : July 24, 2019
Accepted : August 12, 2019
* Corresponding Author : Kim Won Il(wikim@markany.com)

존 영상 장비에서도 차량 추적이 가능하도록 지원하는 것을 목표로 한다. 이를 위해 운영체제를 설치할 수 없거나 지원되지 않는 하드웨어 환경에서도 차량 탐색이 가능한 파일 데이터베이스 구조를 이용하여 특정 차량의 범죄 차량 여부를 빠르게 판단할 수 있는 효율적인 탐색 알고리즘을 제안한다.

2. 관련 연구

2.1 차량 번호판 인식

차량 번호판 인식 시스템은 이동식 CCTV 카메라 또는 고정 CCTV 카메라에서 획득한 차량 번호판 영상에서 번호판 정보를 실시간으로 인식하는 기술을 의미한다. 차량 번호판 인식 기술은 다른 숫자인식 및 문자인식 기술과는 다르게 번호판의 크기 및 문자와 숫자의 형식이 정해져 있기 때문에, CCTV 카메라에서 획득된 영상에서 번호판을 추출하고 추출된 번호판에서 개별 문자 추출을 수행한 후 문자 및 숫자를 인식하게 한다[6-8].

인식된 차량 번호는 각종 이미지 처리 알고리즘을 이용하여 처리를 위한 전처리 과정이 수행되고, 정확한 정보의 획득을 위해 추가적으로 이미지 필터를 적용한다. 이러한 전처리 과정을 거친 후, 최종적으로 차량 번호판의 정보를 사람 또는 컴퓨터가 인식할 수 있는 형태의 문자열로 추출한다. 이렇게 추출된 정보를 기반으로 번호판 정보를 획득하고, 각종 판별을 위한 정보로 사용할 수 있다. 즉, 차량 번호의 인식은 판별을 위한 기반 정보를 획득하고 인지할 수 있는 형태로 추출하는 것을 의미한다.

2.2 비트맵 트리를 이용한 블록 관리 기법

비트맵(bitmap)을 이용한 정보의 관리는 특정 정보에 대한 단일 상태 정보 또는 설정의 적용 여부를 가장 작은 데이터 표현인 비트(bit)로 관리하는 방법이다[8]. 관리 대상에 대한 단일 상태 정보를 한 비트로 표현하기 때문에 대량의 상태 정보를 표현하기 위해 큰 저장 공간을 필요로 하지 않는다. 또한 정보에 대한 접근과 확인을 위한 연산이 일반 연산에 비해 빠르기 때문에 정보에 대한 정확한 분류, 값의 확인과 설정 처리를 위한 효율적인 알고리즘이 수반되면 하드웨어 사양이 낮은 시스템에서도 충분히 사용이 가능하다. 또한 관리 대상의 수가 매우 많은 경우에도 관리 정보가 비트로 구성되기 때문에 정보의 양이 표현하려는 정보와 동일한 양의 비트 값만을 갖는다는 장점이 있다.

2.3 SQLite

SQLite는 2000년에 시작된 오픈소스 프로젝트로 고사양의 하드웨어 사양을 요구하지 않으면서도 기존의 DBMS와 같은 구조화된 질의 언어(SQL)를 지원하여 데이터에 대해 다양한 접근을 수행할 수 있는 경량화 된 데이터베이스 관리 기능을 제공한다. 기본적으로 속도가 빠르고 Windows, mac, 그리고 안드로이드에서도 동작할 수 있기 때문에 다양한 응용 프로그램에서 활용되고 있다[9]. 또한 데이터베이스의 트랜잭션 처리를 위한 ACID의 지원으로 프로그램 오동작으로 인한 데이터의 손실을 방지할 수 있고, 파일 형식을 BLOB으로 저장하기 때문에 서로 다른 운영체제 간의 데이터베이스 호환에도 문제가 없다는 장점이 있다. 무엇보다도 소스 코드가 공개되어 있기 때문에 프로젝트에 직접적으로 코드를 추가하여 데이터베이스를 활용할 수 있는 장점이 존재한다[9].

그러나 전술한 장점들을 제공하기 위해 SQLite는 하나의 파일을 데이터베이스로 사용하는 단점이 존재하며, 범용성을 목적으로 두고 있기 때문에 단일 종류의 데이터를 관리하기 위한 특수 목적용으로 사용하기에는 높은 비용을 필요로 한다. 또한 기능과 내용에 제약이 있지만, 내부에 DBMS를 포함하고 있어 그 비용이 적다고 보기 어렵기 때문에 열악한 하드웨어 환경에서 사용하기에는 적합하다고 보기 어렵다.

본 논문에서는 차량 번호판의 인식을 통해 획득되는 문자열 정보와 한국의 표준 차량 번호판 정보를 비트맵 파일로 구성하여 다양한 사양의 시스템에 적용할 수 있는 차량 번호 파일 데이터베이스를 구성하고, 이에 대한 빠른 접근을 수행할 수 있는 효율적인 탐색 알고리즘을 제안한다.

3. 파일 데이터베이스 설계와 탐색 알고리즘

3.1 차량 번호판의 구성

한국에 등록된 모든 차량은 전면과 후면에 직사각형 모양(520mm X 110mm)의 금속판으로 해당 차량을 구분할 수 있는 차량번호가 부착되어 있으며, 일반적으로 Fig. 1과 같은 형태로 구성되어 있다.

Fig. 1. Car License Plate

차량번호를 구성하는 각 숫자와 한글은 차량의 종류와 용도 그리고 등록번호로 구성되어 있으며, 차량번호만으로도 차량에 대한 대략적인 정보를 획득할 수 있다. 현재 법으로 규정되어 있는 차량번호판의 구성은 Table 1과 같다[4].

Table 1. Car License Plate Configuration

Car license plate		
Car type	Purpose	Registration Number
01	가	1000

차량번호판은 차종을 나타내는 2자리 숫자와 용도를 나타내는 한글 한 문자 그리고 등록번호로 구성된다. 차종의 경우 차량 종류에 대한 정보이며, 01부터 시작하여 99까지 총 99가지의 차량 종류를 표시한다. 차량 종류는 승용차, 승합차, 화물차 및 특수차로 구성되며, 각 차종에 따른 번호 부여는 Table 2와 같다[10].

Table 2. Vehicle Type Number

Car type	
A car	01 ~ 69
A van of van	70 ~ 79
A freight car	80 ~ 97
A special car	98 ~ 99

Table 1에서 용도를 나타내는 한글 한 문자는 차량의 용도를 의미하며, 받침 없이 사용된다. '가~마', '거~저', '고~조', '구~주'는 비사업용 차량을 의미하며 일반적으로 승용차나 승합차에 부여된다. '바', '사', '아', '자'는 자동차 사업용 차량 가운데 운수 목적을 갖는 차량인 택시와 버스에 부여된다. '배'는 사업용 택배 차량에 부여되며, '허', '하', '호'는 대여 사업용 차량에 부여된다[10]. 이러한 차량 용도는 총 40가지가 존재하며 이를 정리하면 Table 3과 같다.

Table 3. Vehicle Classification According to Purpose

Purpose		Letter in Korea
Non-business use (32 kinds)		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
Business use	Taxi, Bus (4 kinds)	아,바,사,자
	Delivery Car (1 kinds)	배
	Rental Car (3 kinds)	하,허,호

차량 번호판에 부여된 마지막 4자리 숫자는 차량 구분 용도로 사용되며, 첫 자리는 반드시 0을 제외한 번호로 구성되어야 한다. 이후 세 자리는 0을 포함하여 사용될 수 있기 때문에 일련번호는 1000번부터 9999까지의 범위를 갖게 되며, 차종과 용도가 동일한 경우 최대 8999대의 차량 등록이 가능하다. 또한 모든 차량 번호는 중복되지 않는다.

3.2 파일 데이터베이스 구성

3.2.1 파일명과 정보의 구성

차량 번호는 구성 방법과 범위, 그리고 의미가 부여된 글자로 구성되어 있기 때문에 파일 데이터베이스로 구성하는 것이 용이하다. 먼저 차종에 따라 99개의 파일로 구성할 수 있는데, 각 차종 별로 한 글자가 부여되므로 99 × 40, 즉 3,962개의 파일로 전체 차량 번호의 구성이 가능하다.

파일명 생성 규칙은 차종과 용도를 조합하여 "01가", "36노", "43주"와 같은 형태로 생성한다. 이는 이미지 처리를 통해 번호판 정보가 획득될 때 대부분 데이터베이스에 접근을 통한 정보 획득을 위해 번호판 자체를 문자열로 구성하기 때문이다. 즉, Fig. 1의 번호판을 이미지 처리를 통해 인식하게 되면, "33", "호", "5598"의 문자와 문자열 값이 획득된다. 파일명은 차종과 용도를 이용하여 구성되기 때문에 실제로 파일의 내용은 등록번호의 정보로만 구성하면 된다. 즉, 차량 번호를 구성하는 3개의 구성 요소 가운데 2개를 파일명으로 사용하고, 나머지 1개로 파일 내용을 구성하는 형태로 파일 데이터베이스를 구성하면, 등록번호에 대한 파일 정보 구성에 대한 접근만으로 대상 차량에 대한 정보를 획득할 수 있다. 그러므로 파일 데이터베이스에 기록하는 정보는 단순히 대상 차량의 범죄 차량 여부만을 기록하고, 이에 대한 정보 확인을 통해 차량을 판별하도록 구성하였다.

3.2.2 파일 형식과 정보 구조

차종과 용도를 이용한 파일명은 총 3,960개가 존재하고, 파일에 기록되는 범죄 차량 여부 정보를 아스키 값으로 기록한다면, 파일 하나당 8,999개의 차량 정보가 표현되어야 하므로 71,992byte의 크기를 갖게 된다. 따라서 전체 파일 크기는 71,992 × 3,960로 약 285메가가 넘는 크기의 용량이 되는데, 최근 디지털 장치의 저장소 크기는 최소 기가 단위로 구성되어 있기 때문에 크기 자체가 부담이 되지는 않는다. 그러나 차량을 인식하는 기본적인 정보로 사용하는 데는 용량이 적은 편이 아니라고 할 수 있으며, 다양한 형태의 CCTV나 카메라 그리고 동작 환경을 고려한다면 더 작은 파일로 구성된 파일 데이터베이스가 저장 공간 활용과 관리에 유리하다. 따라서 적용 가능한 환경을 다양화하기 위해 정보를 경량화하여 단일 비트(bit)로 범죄 차량 정보를 판별하도록 구성하고 관리하는 파일 형태를 제안한다.

파일명을 이용한 데이터베이스 형태에 차량번호 정보 데이터를 단일 비트로 구성하면, 파일명은 차량번호 확인을 위한 주 키(Primary Key) 형태로 동작하게 되며 해당 파일 내에 등록번호 위치의 값이 범죄 차량 여부를 판별하기 위한 값이 된다. 즉, 파일의 시작부터 마지막까지의 모든 비트는 차량 등록번호를 나타내며, 등록번호는 1000부터 9999까지의 값을 가지므로 8,999비트로 구성되어 약 1,124바이트의 크기를 갖게 된다. 따라서 한국에서 등록할 수 있는 특수 목적 차량을 제외한 일반 차량 번호 전체를 비트로 구성하면 약 5메가 이내로 구성할 수 있으므로 다양한 환경이나 하드웨어 사양이 낮은 시스템에서도 사용할 수 있는 경량화 된 파일 데이터베이스를 구성하였다고 볼 수 있다.

3.3 차량 탐색 알고리즘

3.3.1 알고리즘 용어 정의

차량 탐색 알고리즘에서 사용하는 용어들을 Table 4와 같다.

Table 4. Clossary of Terms

Purpose	Meaning
BC	Registration of crime/unauthorized/stolen car
CN	Vehicle main key file name
CRN	Registration number of a vehicle
CV	Byte Position of BL
CR	Bit Postion of BL

BC(Black Car)는 범죄, 대포 또는 도난 등의 범죄에 이용된 차량의 번호를 추적 대상으로 하여 파일 데이터베이스 위

치를 1로 등록한 차량을 의미한다.

CN(Car Number)는 자동차의 번호판의 차종과 용도를 조합한 번호이며, 파일명으로 생성하여 사용할 주 키를 의미한다.

CRN(Car Register Number)는 차량의 등록번호 4자리를 말하며, 파일 데이터베이스에서 차량 정보 변위를 의미한다.

CV(Car Value)는 차량 등록번호를 8로 나누어 나온 몫으로, 비트로 구성된 차량의 byte 위치 계산을 위해 사용된다.

CR(Car Remainder)는 차량 등록번호를 8로 나누어 나온 나머지로, 최종적으로 차량의 BC 값 획득을 위해 사용된다.

3.3.2 차량 탐색 알고리즘

본 논문이 제안하는 알고리즘은 인식된 차량 번호의 범죄, 대포 또는 도난 차량 등록 여부를 파일 데이터베이스로 탐색하여 정보를 확인할 수 있는 알고리즘으로, 탐색을 위해 정보를 획득하고 판별하는 과정은 다음과 같다.

먼저 전달된 차량 번호를 구분하여 주 키로 사용할 차종과 용도 그리고 등록번호를 구분하여, 파일명과 탐색을 위한 차량 위치를 획득한다. 다음으로 획득한 정보를 이용하여 접근할 파일과 차량 위치 등록번호를 계산한다. 마지막으로 탐색 알고리즘을 이용하여 현재 차량의 실제 등록 값을 획득하여 탐색 차량이 BC 인지 여부를 판단한다. 이렇게 특정 차량 번호를 이용하여 차량의 BC 차량 여부를 확인하는 알고리즘은 Fig. 2와 같다.

```
01  Int function car_bl_search
02      Transform digit from string CRN
03      set CV to (CRN-999)/8
04      set CR to (CRN-999)%8
05      move to cv position and set the value of file read to RD
06      if (BCD(Black car Data 1byte) is shifted by (CR-1)) and compared with 0x01
07          print "black list car"
08      else
09          print "not black list car"
```

Fig. 2. BC Vehicle Detect Algorithm

Fig. 1의 차량 등록 번호 "5598"을 Fig. 2에 적용한 처리 순서는 다음과 같다. 02 라인에서는 문자열로 전달된 "5598"을 연산이 가능하도록 숫자 5598로 변환한다. 03 라인은 차량 정보가 위치한 574번째 바이트 위치를 획득하고, 04 라인은 해당 바이트 내의 차량 정보 비트의 위치인 7을 획득한다. 05 라인에서는 차량 정보의 비트 값을 획득하기 위해 574 번째 바이트 값을 획득한다. 마지막 06 라인부터는 획득 바이트에서 7번째 비트를 획득하여 BC 차량 여부를 판단하고, 판단 결과에 따른 정보를 출력한다.

4. 실 험

4.1 실험 구성과 절차

차량 탐색을 위한 실험의 구성과 절차는 다음과 같다. 먼저 차량 탐색 알고리즘의 탐색 속도를 체크하기 위해 차종에 따른 파일 3,960개를 전체를 생성하였다. 다음으로 각 파일의 초, 중, 후반부에 각각 BC 차량을 하나씩 총 3개를 등록하고, 단일 차량 탐색 시간과 위치에 따른 차량 탐색 시간을 구하여, 탐색 시간 성능과 파일 데이터베이스 구성의 효율성을 실험하였다. 마지막으로 전체 탐색 수행 횟수는 3,960 × 3으로 총 11,880번의 탐색을 수행하였고, 이에 따른 결과를 측정하였다. 탐색 성능 평가는 다음과 같이 3가지 다른 실험 구성으로 진행하였다.

- EX-A: 탐색 요청 시, 파일을 개방하고, 정보를 읽어 탐색 수행
- EX-B: 초기화 시점에 모든 파일을 개방한 상태에서 요청 시, 탐색 수행
- EX-C: 초기화 시점에 모든 정보를 메모리에 적재하고, 요청 시에 탐색 수행

이러한 실험 환경을 구성한 이유는 다양한 하드웨어 환경에서 제안 알고리즘을 이용하여 차량 탐색 알고리즘의 적용 가능성을 확인하기 위해서이다. 실험을 수행한 하드웨어 환경은 다음과 같다.

- CPU: i5-82520U 1.60GHz
- Memory: 4G
- OS: Linux mint 19.1

4.2. 실험 결과

실험에서 구성한 3가지 성능 실험의 차량 번호판 요청 시간부터 탐색 완료 시점까지의 탐색 시간 측정 결과는 Fig. 3과 같다.

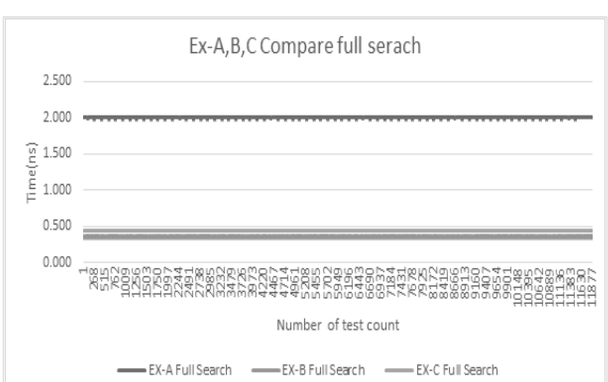


Fig. 3. Comparison of EX-A, EX-B, and EX-C in Terms of Detection Time

전술한 실험 구성으로 11,880번의 BC를 탐색한 결과는 모두 Fig. 3과 같이 특정 시간 안에 처리가 완료되었음을 확인할 수 있다. 세부적인 실험 결과는 아래와 같다.

EX-A 실험 결과, 차량 번호 탐색을 요청한 후 대상 파일을 개방한 시간이 평균 1,538ns로 측정되었고, 요청 차량 위치에 해당하는 1byte를 읽고 파일을 닫을 때까지는 평균 2,698ns의 시간이 소비되어 파일 I/O 처리 시간은 평균 4,236ns로 나타났

다. 그리고 파일에서 차량 위치 계산 시간은 평균 400ns, BC 탐색 시간은 평균 73ns로 측정되어 요청 차량 탐색을 위한 알고리즘 자체 동작 시간은 평균 473ns로 측정되었다.

EX-B 실험 결과, 파일을 개방한 시간은 평균 1,550ns로 요청 차량 위치에서 1byte를 읽고 닫을 때까지 평균 2,724ns로 측정되어 파일 I/O 시간은 총 평균 4,274ns로 측정되었다. 차량 탐색 알고리즘에서 파일 위치 계산 시간과 BC 차량 탐색 시간은 339ns로 측정되었고 알고리즘 동작 시간은 평균 4.613ns로 측정되었다.

EX-C 실험 결과, 단일 파일 개방 시간이 평균 1,650ns로 측정되었고, 파일 전체(8999byte)를 읽고 닫을 때까지 평균 558,233ns가 소비되었다. 즉, 3,960개의 파일 전체를 개방하고 읽어 들인 다음 닫을 때까지의 총 시간은 2.210sec로 측정되었다. 총 시간에서 단일 파일 평균값을 산출하면 평균 559,883ns의 시간이 소비되었다고 볼 수 있다. 요청 차량 위치 계산과 BC 차량 탐색 시간은 다른 실험 결과들과 거의 비슷하게 129ns로 측정되었다.

각 실험 결과 정리는 Table 5와 같다.

Table 5. Full Experimental Results

Classification		EX-A (ns)	EX-B (ns)	EX-C (ns)
File I/O	Open	1,538	1,550	Avg. 1,650
	Read/Close	2,698	2,724	Avg. 558,233
	Total	4,236	4,274	Avg. 559,883
Algorithm	Location Calculation	400	294	313
	BC Search	1,602	65	129
	Total	2,002	359	442

파일 I/O 처리 실험 결과에서 파일 개방에 소비된 시간은 대체적으로 비슷하였으나, EX-C의 경우 연속적인 파일의 개방과 읽기 및 닫기로 인해 시스템의 다른 실험들에 비해 단일 파일당 처리 시간이 오래 걸린 것으로 나타났다. 그러나 EX-C의 경우, 다른 실험의 단일 바이트 읽기와 달리 전체 파일 내용을 읽어 메모리에 적재하여 시간이 더 소비되기 때문에 이러한 현상이 발생한다. 따라서 파일 I/O에서 발생한 개방 시간 차이는 최대 1,000ns 정도이므로 차이가 없다고 볼 수 있다. 읽기와 닫기에서 EX-A, EX-B는 거의 차이가 발생하지 않았으나, EX-C의 경우 전체 파일을 읽어 메모리에 적재하는 시간으로 인해 다른 실험에 비해 긴 시간이 소비되었다. 그러나 이 시간 차이도 최대 555,609ns 정도로 큰 차이가 발생하지 않았다고 볼 수 있다.

알고리즘의 처리 결과, 범죄/대포/도난 차량을 판단할 때 차량 정보를 이용하여 차량 정보가 위치하는 변위와 해당 값을 실제 획득하여 판단하기까지 각각 2,000ns로 측정되었다. 즉, 본 논문이 제안한 알고리즘의 수행 시간은 항상 일정한 시간 내에 처리가 되었고, 각 연산은 400ns 내에 처리가 가능하며 최대 1,600ns 내에 알고리즘 전체가 동작하고 있는 것을 확인하였다. 모든 실험에서 요청 차량에 대한 탐색은 특정 시간 안에 탐색이 완료되었음을 Fig. 3에서 확인하였으며, 파일 I/O에 관련된 시간을 제외하면 각 실험 단위에서 탐색 알고리즘이 항상 동일한 시간에 처리 및 완료되었음을 알 수 있었다.

4.3 SQLite 성능 비교

SQLite는 자체 성능의 우수함을 증명하기 위해 MySQL과 PostgreSQL 간의 각종 트랜잭션 처리 속도를 비교한 내용을 [9] 제공하고 있다. 성능 평가는 오래전에 수행되었고, 실험 당시의 하드웨어 환경과 실험 환경을 동일하게 구성할 수 없기 때문에 운영체제와 메모리를 동일하게 구성한 가상 머신으로 구성하고 실험을 수행하였다. 성능 평가 항목으로는 INSERT, UPDATE, DELETE와 같은 DML과 테이블 제거와 같은 DDL도 포함되어 있으나, 제안 파일 데이터베이스는 전체 번호판을 모두 파일로 구성한 상태이므로 INSERT 연산을 수행하지 않으며, 모든 번호판 정보를 가지므로 DELETE 연산도 발생하지 않는다. 또한 생성된 파일 구조 또는 파일을 삭제하지 않고 데이터의 조작만을 수행하기 때문에 SELECT, UPDATE 연산의 속도만을 비교 대상으로 정하였다. 성능 비교 결과는 Table 6과 같다.

Table 6. Speed comparison with SQLite

TEST	SQLite (sec)	EX-A (sec)	EX-B (sec)	EX-C (sec)
100 SELECT	2.494	0.000304	0.000304	0.000003
5000 SELECT	1.121	0.020852	0.020852	0.000237
1000 UPDATE	0.637	0.009244	0.004415	0.001595
25000 UPDATE	3.520	0.142084	0.095982	0.025542

Table 6에서 확인할 수 있듯이 SQLite의 데이터 처리 속도보다 제안 파일 데이터베이스를 활용한 알고리즘의 구성이 최소 10,000배에서 최대 100,000배 빠른 탐색과 정보 처리가 가능함을 확인할 수 있다. 이는 2.3에서 서술한 바와 같이 SQLite는 범용적인 사용을 지원하기 위한 기능들, 그리고 용량과 기능은 적지만 범용적인 DBMS의 기능을 포함하고 있기 때문이다. 제안 구조는 범용적인 목적이 아닌 차량 번호판만을 효율적으로 관리하고 접근하기 위한 최적의 구조를 가지므로 SQLite보다 빠르게 정보 획득을 위한 효율적인 탐색과 정보 변경이 가능하였다. 즉, 제안 파일 데이터베이스와 알고리즘은 범용 목적으로 사용될 수 없지만, 전체 차량 번호 정보에서 범죄 차량의 정확하고 빠른 탐색과 확인에서는 매우 뛰어난 성능을 보였다고 판단된다.

실험 결과, 제안 차량 탐색 알고리즘은 빠른 차량 탐색이 가능하였고, 차종과 용도를 주 키로 구성하여 등록번호를 비트로 구성한 파일 데이터베이스 구조가 차량 번호 정보의 보관과 관리에 적합하다는 것이 증명되었다. 또한, 전체 실험에서 파일 위치 계산과 탐색 시간이 실험 종류와 관계없이 일정한 시간 내에 완료되었으므로 시간 복잡도 O(1) 역시 만족하였다.

5. 결 론

본 논문은 한국의 전체 표준 차량 번호판의 차종, 용도의 조합을 주 키(Primary Key)로, 등록번호 데이터를 비트로 활용한 파일 데이터베이스를 구성하고, 빠른 정보 접근을 수행할 수 있는 알고리즘을 제안하였다. 제안 알고리즘의 성능 실험은 파일 상태의 정보, 파일 개방 상태 그리고, 모든 정보의 메모리 적재 상태로 구분하여 다양한 하드웨어 또는 시스템 사양과 환경에 적용할 수 있는지 여부를 확인하기 위해 진행하였다. 그 결과 모든 경우 최대 20ms 내에 번호판을 탐색하고, 상태 정보를 확인할 수 있었다. 또한 비트 단위 정보로 구성된 차량 상태 정보에 대한 빠른 접근은 카메라를 통해 인식된 차량 번호판에 대한 빠른 정보 확인이 가능함을 증명하였다.

제안 파일 데이터베이스 구성은 한국에서 등록할 수 있는 현재 표준인 일반 목적 차량의 모든 번호를 보관 및 관리하기 위해 3,960개라는 대량의 파일을 생성하고, 이에 대한 탐색을 수행하였는데, 구성된 대량의 파일은 관리가 매우 어렵다는 단점이 있다. 향후 효율적으로 파일의 개수를 줄여 관리와 성능 및 효율성을 향상할 수 있는 방법을 연구해야 할 것이다.

References

[1] J. Y. Chio, H. N. Yim, and Y. H. Lee , “Review of Theoretical Approach to Efficacy of CCTV,” *Journal of Police*, Vol.15, No.4, pp.31-57, 2015.

[2] J. S. Lee and C. W. Lee, “An Effect Evaluation of Automated Illegal Parking Control System,” *Korean Studies*, Vol.8, No.3, pp.43-55, 2007.

[3] S. Y. Ju, “Study on Improvement of Operation Plan for Unmanned Traffic Enforcement Equipment,” Korea Road Traffic Authority Traffic Science Institute, 2012-0119-122, 2012.

[4] Y. B. Sim, “Vehicle search and notification system for crime using CCTV,” *Chungbuk National University Graduate School of Digital Information Convergence*, 2013.

[5] E. S. Kim, and J. H. Go, “Development of CCTV Real-Time Vehicle Tracking System,” Research Report, 2007.

[6] J. G. Jang, “Real-time Vehicle License Plate Recognition Method using Vehicle-loaded Camera,” *Journal of TIIS*, Vol.6, No.3, pp.147-158, 2005.

[7] K. Y. Lim, H. R. Byun, and Y. W. Choi, “Vehicle License Plate Detection in Road Images,” *Journal of KIISE*, Vol.43, No.2, pp.186-195, 2016.

[8] T. I. Jeong, I. H. Ra, and S. J. Kim, “A Large Multimedia File Management Mechanism using the Bitmap Tree,” *Journal of KIISE*, Vol.21, No.5, pp. 874-886, 2005.

[9] SQLite [Internet], https://www.sqlite.org/

[10] E. Y. Noh, “A Study on Typography of Vehicle License Plate in Korea,” *Society of Korea Design Trend*, Vol.32, pp.487-496, 2011.

심 철 준

https://orcid.org/0000-0002-7643-0344

e-mail : spsj1004@gmail.com

2003년~2018년 아이지코 연구소 소장

2017년 건국대학교 컴퓨터·정보통신공학과 (박사)

2018년~2019년 건국대학교 SW중심대학 산학협력중점교수

2019년~현 재 ㈜아이지코 연구소 소장

관심분야 : 시스템 프로그래밍, 시스템보안, 영상처리, 인공지능, 증강현실

유 상 현

https://orcid.org/0000-0002-1819-9252

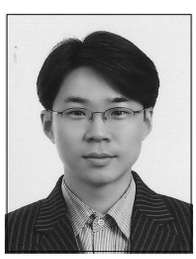
e-mail : yoosh22@gmail.com

2014년 건국대학교 컴퓨터·정보통신공학과 (박사)

2016년~2017년 건국대학교 상허교양대학 초빙교수

2018년~현 재 경민대학교 융합소프트웨어학과 조교수

관심분야 : 영상처리, 인공지능, 시스템보안



김 원 일

https://orcid.org/0000-0002-3689-2851

e-mail : wikim@markany.com

2015년 건국대학교 컴퓨터·정보통신공학과 (박사)

2015년~현 재 ㈜마크애니 기술연구소 책임연구원

관심분야 : 보안, 시스템 프로그래밍, 운영체제