자동차 내부정보를 다루는 안드로이드 기반 스마트 자동차 블랙박스 어플리케이션 개발

김민영* · 남재현** · 장종욱***

A Development of Android-based Smart Car Black Box Application Using Inside Car Information

Minyoung Kim* · Jae-hyun Nam** · Jong-wook Jang***

본 연구는 교육과학기술부와 한국연구재단의 2012년도 지역혁신인력양성사업으로 수행된 연구결과임

요 약

현재 우리나라에서는 일반 자동차 부품처럼 보급되고 있는 자동차 블랙박스는 하드웨어 기반의 장비로, 추후 애로사항 수정 및 추가기능 업데이트가 불가능하다. 또한 여러 가지 물리적 문제로 인하여 많은 운전자들이 불편을 겪고 있다. 현재 출시된 블랙박스는 GPS와 영상정보만 수집하여 추후 저장된 사고기록을 바탕으로 사고원인의 규명 할 때 신뢰성이 낮은 결과가 나올 수 있다.

본 논문에서는 앞에서 서술한 기존 자동차 블랙박스의 문제점을 해결하고자 기존의 블랙박스에서 취급하는 정보 이외에 OBD프로토콜을 통해 수집되는 자동차 내부정보를 다루는 안드로이드 OS용 자동차 블랙박스 어플리케이션을 개발하고자 한다.

ABSTRACT

South Korea's black box made of the hardware-based has problem that is modify program update and Physical errors. This problem makes the driver uncomfortable. And it is save to built-in memory, after collect with GPS and Video Data. However, These data are the information used for accident analysis lacks credibility.

In this paper, to solve this problem, android Black Box application has been developed. This application is collecting OBD protocol(information inside the car) and Existing data.

키워드

자동차, 블랙박스, OBD, 안드로이드

Key word

Car, BlackBox, OBD, Android

* 준회원 : 동의대학교 컴퓨터공학과

** 정회원 : 신라대학교 IT학과

*** 종신회원 : 동의대학교 컴퓨터공학과 (jwjang@deu.ac.kr)

접수일자 : 2012. 06. 01

심사완료일자 : 2012. 06. 01

Open Access http://dx.doi.org/10.6109/jkiice.2012.16.6.1167

Ⅰ. 서 론

2010년 기준 30만대의 자동차에 탑재된 자동차 블랙박스는 자동차 충돌 사고의 전·후 상황을 기록하여 신속한 사고 후 처리 및 과학적인 사고 해석을 위한 자동차 전장품이며, 향후 에어백, ABS에 이어 대중화 가능성이 높은 차량의 장비로 예상된다.[1][2]

현재 국내에 출시된 자동차 블랙박스는 카메라, GPS 그리고 가속도센서가 내장된 하드웨어 기반 제품이다. 하드웨어 기반 자동차 블랙박스 제품은 제조단가가 낮아 운전자가 저렴한 비용으로 구입이 가능하지만 추후 추가기능 적용 및 에로사항을 수정 한 프로그램 업데이트가 힘들다. 또한, 자동차의 시거 잭에서 전원을 받는 블랙박스 제품은 시동 시 본체의 전원 공급 시 발생하는 과전류 인가 문제와 블랙박스에 기록된 정보를 확인하기 위해 매번 본체 또는 메모리 탈·부착 시 발생하는 물리적 오류는 현재 자동차 블랙박스의 문제점으로 꼽을 수 있다. 기존의 자동차 블랙박스가 사고 기록 시 영상 및 GPS 정보를 저장한다. 이 정보만 으로 사고 당시 상황을 재구성 하여 사고 원인을 규명하기엔 조금 미흡한 면이 있다. 이러한 기존의 블랙박스의 문제점을 해결하고자 본 논문에서는 자동차 내부정보까지 수집이 가능한 안드로이드 OS용 자동차 블랙박스 어플리케이션을 개발하고자 한다.

Ⅱ. 관련연구

2.1. 시스템 환경

본 논문에서 개발하고자 하는 자동차 블랙박스 어플리케이션은 Google社의 모바일 운영체제(OS)인 안드로이드에서 실행되도록 개발하였다.

안드로이드는 리눅스 커널 기반의 OS라 임베디드 장비에 적합하며, SDK에서 제공하는 API를 통해 각 통신 및 추가적인 하드웨어를 손쉽게 사용할 수 있다.[3]

본 논문의 자동차 블랙박스 어플리케이션은 안드로이드 장비에 추가된 Bluetooth, 카메라, 가속도센서, GPS 모듈들을 사용한다.

2.2. 사고 감지/판단

자동차 블랙박스의 사고 감지하는 기술은 에어백 전개신호 감지 또는 가속도 센서를 이용한 감지 방법이 있다.[4] 가속도 센서를 이용해 사고 감지할 때, 가속도 센서의 측정값이 변화가 있을 경우 특정 임계값 이상 일 때 사고를 판단한다.[1] 블랙박스는 운전자의 수동 조작으로 인한 사고를 감지해야 한다. 이는 사고의 정도가 경미한 경우 운전자가 직접 사고 전후 사정을 저장 할 수 있는 예외 상황을 감안해야 한다.[1] 본 논문의 사고 감지 및 판단은 가속도 센서와 사용자 조작 사고를 감지하는 것을 채택하였다.

2.3. 정보 수집 및 저장

본 논문에서는 안드로이드 장비에 내장된 카메라, Bluetooth, GPS를 이용하여 기존 자동차 블랙박스 보다 사고기록내용의 신뢰성을 높여 사고 원인을 보다 정확하게 규명하기 위해 영상 및 GPS 정보 이외에 자동차 내부정보를 수집하여 함께 저장한다. 자동차 내부정보는 Bluetooth 통신을 지원하는 OBD Scanner를 이용하여 안드로이드 장비와 Bluetooth 통신을 통해 실시간으로 원하는 OBD 프로토콜 정보 조회 및 결과를 수신 할 수 있다. 여기서 사용하는 프로토콜 규격인 OBD(On- Board Diagnostics)는 자동차 ECU들로 연결된 각종 센서 값을 진단 할 수 있는 직렬통신으로[5], 본 논문에서는 사고 당시 운전자의 자동차 조작 상태 및 차량의 거동을 기록하는 것 용도로 사용한다.

사고정보의 기록은 안드로이드 OS의 특성상 외장 플래시 메모리에 저장한다. 외장 플래시 메모리는 다른 어플리케이션도 함께 사용하는 영역이라 항상 여유용량을 일정 정도 남겨둬야 한다.

Ⅲ. 어플리케이션 설계

본 논문의 자동차 블랙박스 어플리케이션은 크게 사고를 감지해 각종정보를 기록하는 '블랙박스 프로세스' 부분과 기록된 정보관리 및 실행환경을 설정하는 기능을 가진 '블랙박스 관리' 부분으로 두 개의 부분으로 나누어 설계되었다(그림 1,2).

3.1. 사고감지 및 가속도센서 모듈

'사고감지'모듈은 '가속도센서'모듈과 상호 유기적 관계를 가진다. '가속도센서'모듈에서 안드로이드 SDK

의 API를 통해 100ms 주기로 가속도센서의 측정값을 추출 후 이전 가속도와 현재의 가속도를 계산하여 차이를 계산한 값을 '사고감지'모듈로 전달한다. '가속도센서' 모듈에서 해당 값을 전달 받은 '사고감지모듈'은 환경설정에서 설정 해 둔 임계값과 비교 해 전달 받은 값이 높으면 '사고기록'모듈에 사고발생 한 내용을 알리도록 설계 되었다. 또한, 사용자가 수동으로 사고를 저장 할 수 있도록 예외사고감지 부분도 구현되어 있다.

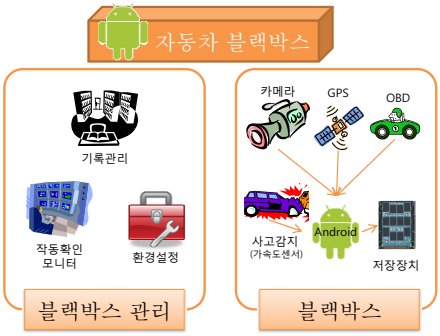

그림 1. 자동차 블랙박스 어플리케이션 개념구성도
Fig. 1 Car blackbox applications Concept Diagram

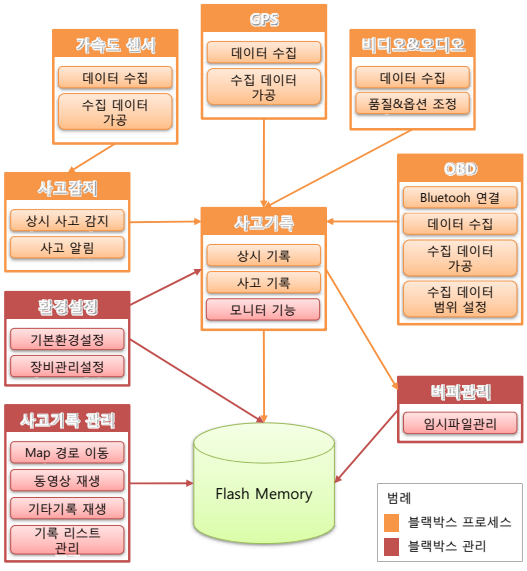

그림 2. 어플리케이션 기능 모듈별 설계구성도
Fig. 2 Module-specific application functionality Design configuration Diagram

3.2. GPS 및 비디오&오디오 모듈

'가속도센서'모듈과 동일하게 안드로이드 SDK의 API를 통해 각 센서들의 측정값을 받는다.

'GPS'모듈은 정확한 사고 기록을 위해 1초 또는 1m 거리이동시 좌표 값을 받을 수 있게 설정하였고, 좌표 값이 변경될 경우 '사고기록모듈'에 그 값을 전달한다.

'비디오&오디오'모듈은 사고 기록 시 환경설정에 따라 음성녹음여부와 비디오의 녹화품질이 결정된다. 비디오 녹화품질은 저화질(320×240), 중화질(600×480), 고화질(800×480) 등 3가지 중 하나만 선택할 수 있도록 설계 하였다.

3.3. OBD모듈

'OBD'모듈은 'GPS'모듈과 '비디오&오디오'모듈과 다르게 외부장비와 통신을 해야 한다.

OBD Sanner는 질의응답 통신방식이기 때문에 OBD Sanner에 통신을 통해 명령어로 질의해야 해당 명령어의 결과 값을 응답 받을 수 있다. 본 논문의 어플리케이션은 Bluetooth통신으로 OBD Scanner과 연결 후 정보요청 및 결과 값 수신하도록 설계하였다. 이때 안드로이드 SDK의 API를 사용하여 Bluetooth 통신을 가능하도록 실행한다. 6개의 명령어(표 1)를 요청하며 200ms 간격으로 한 명령어씩 보낸다.

표 1. 해당 어플리케이션 요청 OBD 명령 목록
Table. 1 This application requests a list of OBD commands

명령 요청 내용	명령 코드	명령 결과 값	결과 값 추출 공식	결과 값 범위
Speed (km/h)	01 0D	41 0D *x1*	*x1*	0 ~ 255
RPM (rpm)	01 0C	41 0C *x1 x2*	[(*x1* × 256) + *x2*] ÷ 4	0 ~ 16,383.75
AIR FLOW SENSOR (%)	01 11	41 11 *x1*	(*x1* × 100) ÷ 4	0 ~ 100
Engine coolant temperature (°C)	01 05	41 05 *x1 x2*	*x1* - 40	-40 ~ 215
MAF air flow rate (grams/sec)	01 10	41 10 *x1*	[(*x1* × 256) + *x2*] ÷ 100	0 ~ 655.35
Engine oil temperature (°C)	01 5C	41 5C *x1*	*x1* - 40	-40 ~ 210

또한 OBD Sanner에서 질의에 대한 결과 메시지를 해석하는 기능도 추가로 설계되어 있다. OBD Sanner는 OBD 표준에 따라 모든 송·수신 메시지는 16진수 ASCII로 구성되었으며, 일부 특정 값은 일정 공식으로 계산해야 값을 해석 할 수 있다(표 1). 메시지 수신 후 해석이 다되는 경우 '사고기록모듈'에 그 값을 전달한다.

3.4. 사고기록모듈

‘사고기록’모듈은 ‘사고감지’, ‘GPS’, ‘비디오&오디오’, ‘OBD’, ‘버퍼관리’의 모듈과 상호 유기적 관계를 가지며 ‘사고기록모듈’ 처음 실행할 때 각 해당 모듈을 초기화 및 사고 기록을 저장하기 위해 각 해당모듈을 컨트롤 한다. ‘사고기록’모듈은 ‘GPS’, ‘OBD’ 모듈로부터 전달 받은 내용을 사전에 정의한 필드 규격에 맞추어 레코드 형식으로 텍스트 파일로 저장한다. 영상기록은 ‘비디오&오디오’ 모듈을 통해 저장한다. 이렇게 되면 매 사고기록마다 비디오파일(mp4)과 텍스트파일(txt) 2개의 파일이 생긴다(그림 3의 상단).

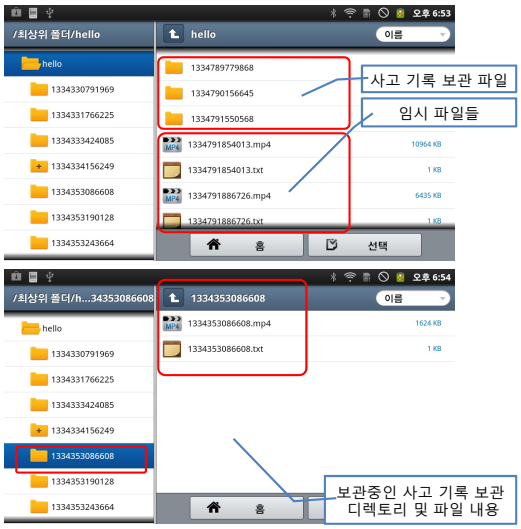

그림 3. 실제 임시파일 관리 설계부분을 구현한 수행 결과
Fig. 3 Result run that implement of the temporary file management process

‘사고기록’모듈은 1분마다 임시 파일에 저장되며 그 파일들은 생성되는 시점의 시간정보를 파일명으로 둔다. 만약 ‘사고감지’모듈에서 사고 발생 할 경우 현재 임시로 기록되는 사고 기록 파일에서 사건 발생 시점으로부터 추가로 30초 저장한다. 이렇게 사고발생 시 저장된 기록 파일은 해당 시간의 이름의 폴더를 새로 생성 후 저장된 임시파일들을 생성된 폴더에 옮긴다(그림 3의 하단). 이렇게 저장된 파일은 ‘사고기록관리 모듈’에서 기록된 내용을 확인 할 수 있다. 이이외의 임시파일들은 ‘버퍼관리’모듈에서 실시간으로 관리하도록 설계하였다.

3.5. 버퍼관리

‘버퍼관리모듈’은 항상 임시파일들이 5개미만으로 보관 할 수 있는 기능을 수행한다. 60초 마다 매번 임시파일들이 저장된 디렉터리를 검색한다. 디렉터리에서 검색한 임시파일 중 파일명이 제일 오래 된 날짜정보를 가지고 있는 파일을 자동으로 삭제하도록 설계 되었다. 사고가 발생한 시점에 저장된 파일들은 각 별도의 디렉터리에 저장되어 있어 그 부분은 검색하지 않도록 설계하였다.

3.6. 사고기록관리

‘사고기록관리’모듈은 크게 ‘기록 리스트 관리’ 기능과 ‘사고 상황 플레이어’기능 두 기능으로 나누어 설계되었다.

‘기록 리스트 관리’기능은 외부 플래시 메모리에 저장된 사고 기록 파일을 리스트형태로 사용자에게 제공한다. 또한, 기록된 정보 중 사용자가 선택한 정보를 확인 가능하도록 ‘사고 상황 플레이어’로 볼 수 있게 연결하는 기능과 사용자가 임의로 삭제할 수 있는 기능이 제공된다. ‘사고 상황 플레이어’기능은 기록 리스트에서 선택 한 사고 기록 정보를 근거로 지도를 이용한 이동경로 표시, 기록된 영상, 자동차 내부정보(OBD정보)를 함께 대입하여 사용자에게 사고 당시 상황을 보여주도록 설계하였다.

3.7. 환경설정

마지막 모듈인 ‘환경설정’모듈은 본 논문의 블랙박스 어플리케이션의 전반적인 환경설정을 담당하도록 설계되었다.

환경설정 모듈은 가속도센서 감도(임계값 설정), 카메라 영상품질, 음성 녹음여부를 설정 가능한 ‘기본설정’과 OBD 스케너를 사용할 수 있도록 Bluetooth 통신을 설정 할 수 있는 ‘외부장비설정’ 기능으로 구분된다.

Ⅳ. 구현 및 결과

실제 본 논문에서 개발된 자동차 블랙박스 어플리케이션은 3개의 탭을 가진 Tab Activity로 개발 하였다. 실제 테스트는 삼성전자의 ‘Galaxy Tab’에서 실시하였다.

4.1. 모니터

모니터 Activity는 블랙박스가 실제 동작하는 부분이다. '블랙박스 시작'버튼을 클릭하면 '블랙박스 프로세스'가 실행이 된다. 블랙박스 프로세스가 이 Activity에 포함되어 있다.

본 논문에 개발된 모니터 Activity는 블랙박스 프로세스가 수집하고 있는 영상정보, Bluetooth 통신으로 수집하는 OBD정보, 가속도센서 측정값, GPS 좌표정보를 동시에 보여준다(그림 4). 기본적으로 '사고감지'모듈은 가속도센서의 값을 측정하여 사고를 감지한다. 블랙박스 프로세스가 실행 중 '긴급(수동)저장'버튼을 클릭하면 '사고감지'모듈에서 사용자 발생 사고로 인지하여 사건을 기록하게 된다(그림 4).

그림 4. 모니터 Activity 실행화면(상), 실제 블랙박스 어플리케이션 테스트 화면(하)
Fig. 4 The screen of 'Monitor' Activity Run(Top), and the screen of the actual BlackBox application test results(Bottom)

4.2. 기록

기록 Activity는 블랙박스 프로세스가 사고 감지 후 기록한 정보를 리스트 형태로 출력한다. 이 리스트 내용 중 한 개의 값을 선택하면 바로 해당 기록정보를 바탕으로 '사고 상황 플레이어'가 실행된다. 길게 터치하면 해당 기록 정보를 삭제 할 수 있는 메뉴가 제공된다(그림 5).

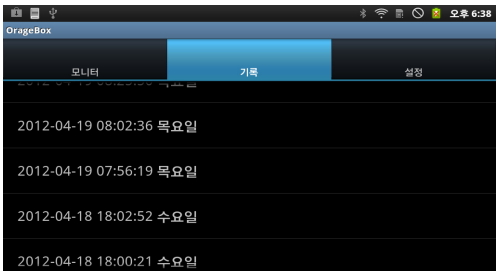

그림 5. 기록 Activity 실행화면
Fig. 5 The screen of 'Record' Activity Run

4.3. 사고 상황 플레이어

사고 상황 플레이어는 다른 Activity와 달리 독립적으로 제작되어 Intent를 통해 호출해야 한다. 이때 Intent로 넘겨주는 값은 선택한 파일이름이다. 사고 상황 플레이어 Activity는 처음 실행 하면 제일 먼저 파일이름으로 된 폴더와 영상파일 및 텍스트 파일을 검색하여 실행할 준비를 한다. 만약 없다면 실행이 자동 종료된다.

사고 상황 플레이어 Activity는 기록된 영상파일과 텍스트 파일을 근거로 영상정보, Bluetooth 통신으로 수집한 OBD정보, GPS의 좌표 값을 지도 위에 표시하여 이동경로를 보여주는 기능을 동시에 보여준다(그림 6).

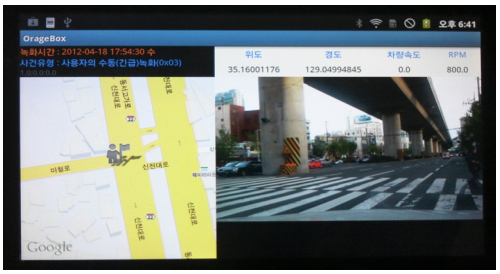

그림 6. 실제 '사고상황 플레이어'를 실행한 블랙박스 어플리케이션 테스트 화면
Fig. 6 The screen of 'Accident review player' run testing results

4.4. 설정

설정 Activity는 블랙박스 프로세스가 실행하는데 필요한 설정을 할 수 있는 곳이다(그림 7). 이 Activity는 PreferenceActivity로 구현하였다. PreferenceActivity는 환경 설정한 내용을 XML파일에 저장한다. 매번 모니터 Activity가 실행 시 XML에 저장된 환경 설정파일을 불러와 초기화 시킨다.

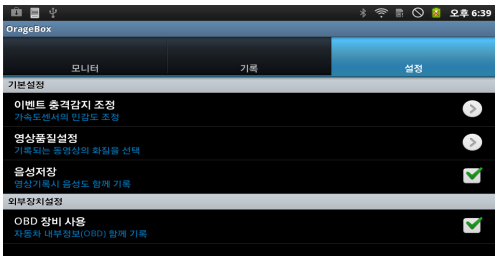

그림 7. 설정 Activity 실행화면
Fig. 7 The screen of 'Preferences' Activity Run

Ⅴ. 결론 및 고찰

본 논문은 앞서 서론에서 거론한 기존의 하드웨어 블랙박스의 문제점을 해결하고자 Android 기반 임베디드 시스템에서 실행되는 자동차 블랙박스 어플리케이션을 제작하였다. 자동차 내부정보를 기록하여 사고 재연 시 사고원인 규명에 대한 내용을 신뢰성을 높이고자 하였다. 또한 자체 버퍼 관리 프로세스를 개발 해 외장 플래시 메모리의 용량을 최소한으로 사용하면서 블랙박스 기능을 실행 할 수 있게 개발하였다.

하지만 이 어플리케이션은 개선해야 될 부분들이 있다. 먼저 사고 감지 신호를 통해 정보의 저장과 동시에 이를 원격지로 전송하는 사고 자동 통보(ACN) 기능[4]의 도입과 사고 원인 규명에 좀 더 신뢰성을 얻기 위해 사고 당시 날씨 정보 수집 및 OBD정보 통한 운전자의 운전 성향 분석 등의 기능이 필요하다.

또한 본 논문의 어플리케이션은 실행이 되어야 블랙박스가 보통 Activity이라 계속 해당 Activity를 실행 시켜야 블랙박스 기능을 실행한다. 하지만 Android 장비의 특성상 멀티태스킹이 가능하도록 하기 위해서는 본 논문의 블랙박스 어플리케이션 중 블랙박스 프로세스 부분을 Android 서비스(백그라운드) 프로그램 형식으로 변경해야 될 것이다. 그렇게 되면 다른 어플리케이션과 동시에 실행하며 블랙박스 기능을 수행 할 것이다.

감사의 글

본 연구는 교육과학기술부와 한국연구재단의 지역혁신인력양성사업(2012년)으로 수행된 연구 결과임.

참고문헌

[1] 한인환 외 1명, "차량용 블랙박스 표준화 동향과 전략", 한국표준협회, 제3회 표준화우수논문 공모전, 2005.
[2] 손해보험협회 정책연구팀, "정책연구 : 국내외 차량용 블랙박스 현황과 제도화 방향 上", 손해보험협회, 월간손해보험 2011. 5.
[3] "안드로이드 개발 문서", http://developer.android.com/
[4] 한인환, "차량용 블랙박스 기술 특허분석 및 표준화 방안", 대한교통학회지 제 25권 제 3호, 2007. 6.
[5] "OBD-II PID", http://en.wikipedia.org/wiki/OBD-II_PIDs

저자소개

김민영(Minyoung Kim)

2011년 2월 동의대 컴퓨터공학과 공학사
2011년 3월 ~ 현재 동의대 컴퓨터공학과 석서과정
※관심분야 : 자동차 네트워크, WPAN, 소프트웨어 공학

남재현(Jae-hyun Nam)

2002년 2월 부산대 컴퓨터공학과 공학박사
1993년~2002년 동주대학 네트워크 전자계열 조교수
2002년 ~ 현재 신라대학 IT학과 부교수
※관심분야 : 유뮤선통신 시스템, 자동차 네트워크

장종욱(Jong-wook Jang)

1995년 2월 부산대 컴퓨터공학과 공학박사
1987년~ 1995년 한국전자통신 연구원 연구원
2000년 UMKC Post-Doc.
1995년 ~ 현재 동의대 컴퓨터공학과 교수
※관심분야 : 유뮤선통신 시스템, 자동차 네트워크