Journal of KIIT. Vol. 16, No. 1, pp. 53-60, Jan. 31, 2018. pISSN 1598-8619, eISSN 2093-7571 53
http://dx.doi.org/10.14801/jkiit.2018.16.1.53

FCWS 검출속도 향상을 위한 시스템 설계 및 구현

유 환 신*

System Design and Implementation with Improved FCWS Detection Speed

Hwan-Shin Yu*

본 논문은 2017년도 호원대학교 교내학술연구비의 지원에 의한 연구결과임

요 약

최근 차량의 안전운행을 보조하고 운전자의 편의성을 향상시키기 위해 ADAS(Advanced driver-assistance systems) 시스템이 장착되고 있다. LDWS(Lane departure warning system)과 FCWS(Forward collision warning system)이 그 기술의 핵심이다. 이 중 FCWS는 차량의 충돌사고를 방지하기 위한 핵심 보조 기술로 평가받고 있다. 이에 검출속도의 향상을 위한 많은 알고리즘들이 개발되고 실험되고 있으며, 실제 검출 알고리즘들이 상용화 되고 있다. 실제 영상의 입력부터 최종 인식까지의 임베디드 시스템 전체의 설계가 고민되어야 하며, 처리속도가 같이 향상 되어야 한다. 이는 하드웨어, BSP(Board Support Package) driver와 알고리즘이 결합하여 최상의 효과를 본다. 본 논문에서는 임베디드 시스템의 OS별 특성을 파악하고, 하드웨어 설계 구조를 분석하여, FCWS 속도를 최적화한 시스템의 설계를 제안한다.

Abstract

Recently, ADAS (Advanced Driver-Assistance Systems) system has been installed to assist the safe operation of the vehicle and improve the driver's convenience. LDWS (Lane departure warning system) and FCWS (Forward collision warning system) are the core of the technology. Among these, FCWS has been evaluated as a key assistive technology to prevent vehicle collision. Therefore, many algorithms for improving the detection speed have been developed and tested, and actual detection algorithms have been commercialized. The design of the entire embedded system from the input of the actual image to the final recognition must be contemplated and the processing speed should be improved as well. It has the best effect by combining hardware and BSP driver (Board Support Package) and algorithm. In this paper, we propose the design of a system that optimizes the FCWS speed by analyzing the hardware structure of the embedded system,

Keywords
FCWS, automotive, algorithm, anti-collision, system, design

* 호원대학교 자동차기계공학과
- ORCID: http://orcid.org/0000-0002-4037-2592

· Received: Nov. 29, 2017, Revised: Dec. 18, 2017, Accepted: Dec. 21, 2017
· Corresponding Author: Hwan-Shin Yu
Dept. of Automotive Mechanical Engineering, Howon University, 64 Impi, Gunsan-si, Korea
Tel.: +82-61-450-7110, Email: hsyu@howon.ac.kr

Ⅰ. 서 론

차량의 안전운행을 지원하기 위한 ADAS (Advanced Driver-Assistance Systems) 기술은 빠르게 발전하며 상용화되고 있다. 관련된 기술은 지능형-커넥티트 카 기술의 가장 기본이 되는 기술[1]이다.

그 중에서 카메라로 받아들이는 차량의 주행영상을 기준으로 전방 차량을 인식하고 거리를 측정하여 충돌위험을 알리는 FCWS(Forward Collision Warning System) 기술은 가장 중요한 사고예방 기술이다.

최근 차량의 내구성 및 성능의 향상과 더불어 고속도로의 최고 속도가 상향 조정되어, 고속도로에서의 차량간 속도차가 발생하여 그 위험의 예방과 감지가 매우 중요한 상황이다. 위험의 인지로부터 경보의 전파까지 시스템 상에서 빠르게 처리해야 하는데, 현재의 임베디드시스템은 구조가 복잡하여 실시간 경보가 어렵다.

본 논문의 구성은 다음과 같다. 2장은 관련연구 기술을 살펴보고, 응용 사례 및 최근의 최근 시스템 동향을 살펴본다. 3장에서 본 논문이 제안하는 알고리즘 최적화 방법 및 구현방법에 대해 자세하게 기술하며, 시스템의 설계 방식을 제안한다. 4장은 실험결과를 확인하고, 5장에서 연구에 대한 결론을 기술한다.

Ⅱ. 관련 연구

2.1 FPGA기반 FCWS-LDWS 알고리즘 구현

전방의 물체를 인식하고 가장 빠른 경보를 울리는 방법은 인식알고리즘을 탑재한 FPGA기반의 하드웨어 시스템[2]이다. FPGA기반의 방법은 영상의 관심영역이 끝나는 시점에 하드웨어 알고리즘이 즉시 동작하여 매우 빠르게 경보를 처리할 수 있다. 상용화시 고려할 부분이 영상처리와 녹화까지 처리해야 하는 하드웨어 시스템으로 설계해야한다. 그림 1과 같이 FPAG는 I/O가 많아 PCB의 크기와 비용이 증가하는 문제가 발생하게 된다. 또한 처리알고리즘의 업그레이드 및 제품 유지보수의 난점이 있어 제조사들은 Arm SoC(System On Chip)기반의 기능구현을 선호한다.

2.2 레이더와 비전센서 융합 FCWS 알고리즘

비전센서로 거리를 정확하게 측정하는 오류를 개선하기위해 레이더를 결합한 방식[3]의 알고리즘은 레이더가 갖는 거리의 정확성을 영상에서 차량을 검출해낸 데이터와 결합하여 그림 2와 같이 오차를 보정하는 방식이다.

이 방식은 악천후 영상 발생시 동작오류를 개선하는데 효과적이다. 그러나 난반사 신호와 도로면 오차는 극복해야하며, 레이더 시스템의 장착 차량간 오류가 발생할 수 있다.

기술의 발전으로 저가형 초음파 센서를 활용하여 차량을 추적하는 방식의 알고리즘[4]이 개선되어 정확도가 98% 수준으로 높아지고 있으나, 이 또한 영상과 실시간으로 결합되어 최종 판정을 내려야 한다.

그림 1. FPGA 모듈
Fig. 1. Module of FPGA

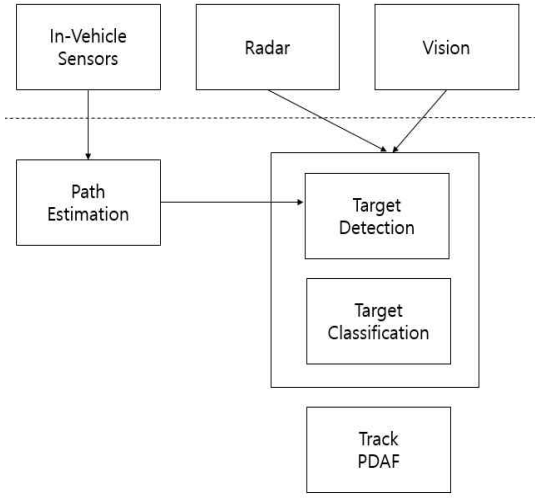

그림 2. 센서퓨전 아키텍처
Fig. 2. Sensor fusion architecture

2.3 스테레오-비전 FCWS 알고리즘

영상을 판독하여 전방차량의 물체를 식별하고 거리를 측정하는 방법에 있어, 오차를 개선하고 인식 속도를 향상하기 위한 방법으로 스테레오-비전 카메라를 활용하는 알고리즘[5]이 그림 3과 같이 제안되었다.

제안된 알고리즘은 92%~99%의 정확도를 보이며, 영상의 S/W 기반 판독기술만을 가지고 충돌예측을 한다. 인식 속도에 대한 성능을 향상하기 위해 관심영역을 지정하고, 차선과 차량의 형태소 및 그림자를 추적하는 알고리즘[6]이 있다.

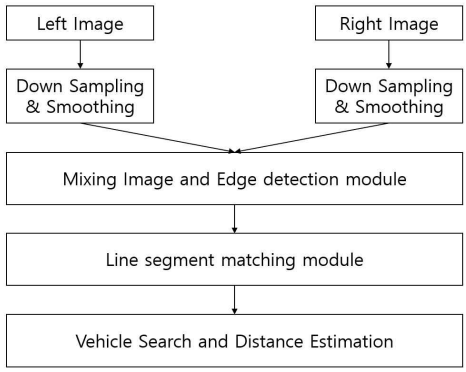

그림 3. 센서퓨전 아키텍처
Fig. 3. Sensor fusion architecture

III. 시스템 분석 설계 및 구현

3.1 임베디드 시스템 하드웨어 구조

S/W 기반 판독 알고리즘을 초소형 시스템 및 차량용 블랙박스에 최적화하는 방식[7]으로 하드웨어를 설계 및 구현해야 한다.

상용 블랙박스 시스템을 구현하는데 사용되는 SoC를 검토한 결과, 영상처리 및 녹화성능의 향상을 위하여 싱글 코어(Single-Core) 또는 듀얼 코어(Dual-Core) 기반의 SoC가 많이 사용되고 있다. 본 논문에서는 듀얼 코어 기반의 그림 4의 RDK를 활용하여, FCWS 검출엔진의 최적화 알고리즘을 구현한다.

그림 4. 듀얼 코어 SoC 보드
Fig. 4. Mainboard of dual-core SoC

표 1. 테스트 RDK의 사양
Table 1. Specification of testing RDK

테스트 RDK 사양	
SoC	AIT8428P, ARM9, Dual-Core
Memory	Embedded DDR3, 400MHz, 16Bit, 64MBytes

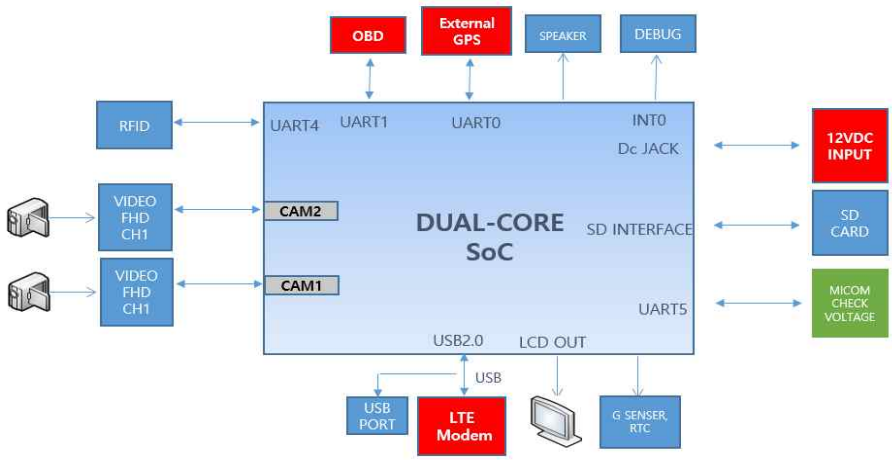

그림 5. 시스템 블록다이어그램
Fig. 5. Block diagram of system

표 1의 제원을 가지는 시스템을 선정하여, 리눅스 OS를 기반으로 일반적으로 알고리즘을 처리하는 방법과 제안하는 알고리즘의 처리방식을 비교해 본다. 그림 5와 같이 카메라와 기타 장치를 연결하여 시스템을 구성한다.

내부에 구현되는 FCWS 검출엔진은 그림 3의 방식인 스테레오-비전 카메라를 활용하는 알고리즘을 적용한다.

테스트를 위한 하드웨어는 실시간 파일저장 기술[8]을 내장한다. 상용화된 제품은 녹화기능을 기본 탑재하기 때문에 시스템이 효율적으로 설계되고 충실하게 최적화 되는지 확인하기 위해서 필요하다.

어라운드 뷰 모니터링 전용의 시스템과 같은 경우는 LDWS(Lane Departure Wanrnning System) 알고리즘[9] 또는 FCWS 알고리즘의 적용시 단일한 기능을 수행하기 때문에 보편적인 성능을 보이게 된다.

3.2 리눅스 OS 기반 알고리즘 구현

리눅스 OS 기반 시스템으로 S/W를 구성할 경우, 영상처리를 위해서 처리되는 BSP는 그림 6과 같이 처리된다. 그림 6은 영상이 한 장 캡쳐되어 FCWS 알고리즘이 처리되기까지 과정을 보여준다. (1) 분석하고자 하는 영상이 한 장 캡쳐되어야 하고 (2) 영상의 캡쳐가 끝나는 시점에 비디오 동기 신호에 따른 인터럽트 서비스 처리를 진행해야 한다.

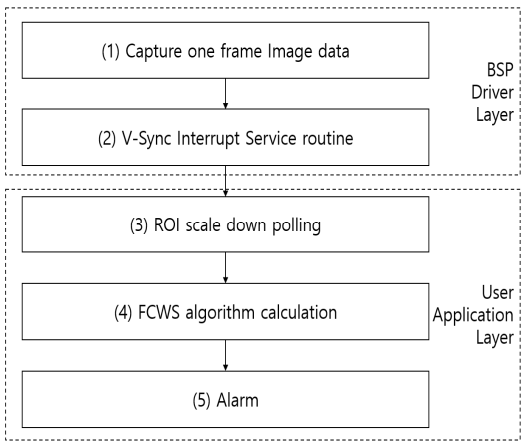

그림 6. 리눅스 OS상에서 영상처리하는 방법
Fig. 6. Image processing method based on linux OS

이 때, V4L2(Video for Linux Version 2) 규격에 따르면, 버퍼 포인터를 다음으로 옮기는 동작을 수행하고 어플리케이션의 IoC시 함수에 폴링된 동작이 있으면, 드라이버에서 풀어주는 웨이크-업 쓰레드(Wake-up Thread) 동작이 발생하게 된다. 상기의 BSP(Board Support Package) 드라이버의 작업이 완료되면, 사용자 응용프로그램 영역의 프로세서는 (3) 대기상태에서 이미지 버퍼 포인터 수신을 확인하는 드라이버의 응답을 받아서, 영상의 RAW DATA(YUV 4:2:2 포맷의 데이터)를 얻게 된다. 이 데이터는 Full HD급의 영상이거나 고해상도의 영상이기 때문에, 영상처리가 가능한 수준의 크기로 축소(Scale-Down) 처리되어야 하며, 특히 처리와 관련된 관심영역을 추출하여 스케일러를 가동하게 된다. (4) 영상처리를 위한 데이터가 준비되어 FCWS 실제알고리즘을 판독하는 루틴을 수행하게 된다. (5) 충돌이 검출되는 상황일 때, 알람을 울리게 된다.

리눅스 OS에서 알고리즘을 구현하기 위해서 그림 6과 같은 절차가 필요하며, 드라이버 영역과 응용 프로그램 영역에서 리눅스 커널 드라이버와 쓰레드, 이벤트 폴링이 사용되게 된다. 이는 알고리즘 구현시, 녹화 동식동작 또는 추가 드라이버 동작 또는 내부 커널 쓰레드 동작시 시스템의 지연을 발생시킬 가능성이 있다. 이러한 구현 방식의 문제점은 실험을 통해서 확인할 수 있다.

3.3 듀얼 코어 기반 BSP 설계 최적화

논문에서 제안하는 알고리즘을 처리하는 방식은 듀얼 코어 BSP를 다르게 동작시키는 것이다. 1번 CPU는 일반적인 U/I 처리를 위하여 리눅스 OS로 녹화 및 저장처리를 실행한다. 2번 CPU는 부트로더에서 활성화하며, 전용의 RTOS로 구현한다. 1번 CPU 리눅스 OS 가동 시 캡쳐버퍼의 포인터를 공유하되 그림 7과 같이 서로 완벽하게 다른 병렬로 동작하게 한다.

그림 7은 듀얼 코어에서 다중 OS를 가동하여, FCWS알고리즘을 동작시키는 방법을 기술한다. (1) 암 코어(Arm Core) 제조사별로 부팅환경을 맞추기 위한 1st 부팅코드를 동작시킨 후, (2) 리눅스 OS 적재를 위해서, U-Boot를 1번 CPU를 통해 가동한다.

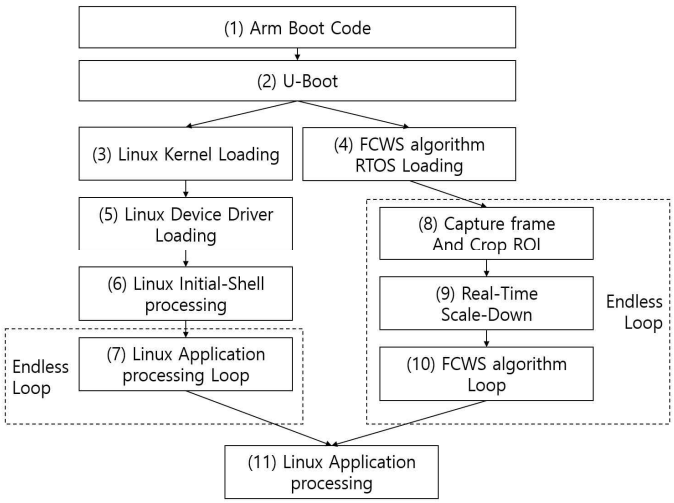

그림 7. 듀얼 코어 다중 OS 가동 방법
Fig. 7. Running the method of multipul os on dual-core

이 때, U-Boot에서 리눅스 커널의 적재동작을 1번 CPU에 서 수행하고, 2번 CPU에서는 RTOS의 적재동작을 호출한다. 여기서 OS적재 동작을 진행하는 이유는 U-Boot에서 물리적 메모리 맵을 가상 메모리 맵으로 변환할 때, 옵션설정이 가능하다. 캡쳐 버퍼의 크기와 위치를 이곳에서 분리하여 포인터 및 맵처리를 진행하여, 코드-콘플리트(Code-Conflict) 또는 데이터-어보트(Data-Abort) 동작을 방지하기 위해서이다.

리눅스 OS의 로딩과정은 (3) 커널 (5) 드라이버 (6) 초기화 쉘 스크립트 (7) 응용 프로그램 실행으로 전통적인 구조로 이루어 진다.

알고리즘 동작 전용의 2번 CPU는 (4) 알고리즘 실시간처리를 위한 최소화된 RTOS가 적재된다. (8) 영상의 캡쳐 및 (9) 관심영역 축소 그리고 (10) FCWS 알고리즘의 동작이 처리되게 된다. 이 후 (8), (9), (10)의 동작만 지속적으로 반복하게 한다.

2번 CPU의 알고리즘 구현부가 완전히 독립되었고, U/I를 담당하는 1번 CPU에 의해 프로세스의 우선순위가 변동되지 않기 때문에, 실기간 처리가 보장된다. 이를 기반하여, FPGA와 유사하게 S/W 처리 블록을 구현한다. 다만 FPGA 방식과의 차이점은 블록데이터 조합 연산이 불가능한 부분에서 성능의 차이를 보이게 된다.

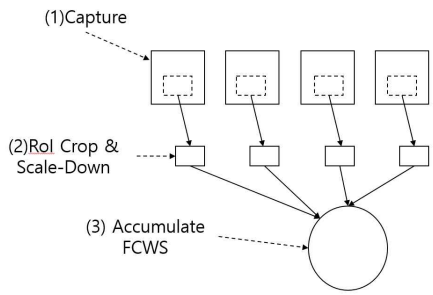

그림 8. FCWS 알고리즘 가속화 방법
Fig. 8. Accelation method of FCWS algorithm

그림 8과 같이 영상이 한 장씩 프레임 단위로 캡쳐된다. 영상은 처리되는 프레임을 기준으로 30Hz로 입력된다. 이 시점을 기준으로 VSync 인터럽트 시점에서 타이머 인터럽트를 발생시켜, ROI이미지의 영역이 지나가는 시점에 즉시, 축소 동작함수를 호출하여, 남은 데이터가 DMA에 쌓이는 동안의 대기 시간을 줄여준다. 이후, 축소 동작 이후에 FCWS 알고리즘 처리 동작을 수행한다. 알고리즘의 수행 연산시간은 축소된 영역의 크기에 비례하게 된다. RTOS이기 때문에, 전용의 알고리즘을 처리수행하기 때문에 축소 수행된 시간과 알고리즘의 처리 수행시간을 합쳐 33ms이내에 수행될 때, 실시간으로 지속적으로 알고리즘의 판독 및 알람처리가 가능하게 된다.

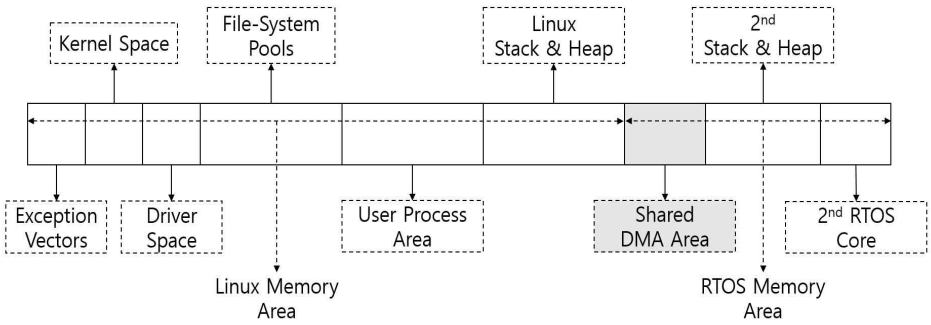

그림 9. 듀얼 코어 다중 OS 메모리 공유 방법
Fig. 9. Sharing the momory of multipul os on dual-core

이런 동작방식의 경우 영상의 캡쳐 이후, 관심영역의 스케일처리와 연산처리가 별도의 인터럽트 지연없이 즉시 수행되기 때문에, 알고리즘의 최적화 정도에 따른 반응속도를 보일 수 있게 된다.

그림 9와 같이 리눅스 OS 메모리는 예외처리 영역, 커널 영역, 드라이버 영역, 파일시스템 풀, 사용자 프로세스 공간과 스택 & 힙으로 구성된다. 2번 CPUdml 알고리즘 구현부는 전용의 RTOS 공간으로 구성되어, 코어 코드 여역과 스택 & 힙으로 구성이 된다. 이 때, 리눅스의 가상 메모리 영역에 의해 조각나지 않는 전용의 대용량 DMA버퍼를 설정함으로써, 그림 8의 알고리즘을 동작시킨다. 리눅스 OS가 비디오 이미지의 캡쳐와 스케일 인터럽트에 관여하지 않음으로써, OS 태스크 스위칭에 따른 과부화와 시간지연을 최소화할 수 있게 된다.

Ⅳ. 실험 결과

FCWS 알고리즘의 처리속도 향상을 위한 처리방식의 개선동작을 통한 결과를 살펴본다. 그림 10, 그림 11과 그림 12는 동일한 FCWS 처리 알고리즘을 사용하며, 가속 알고리즘 적용 전은 리눅스 OS 단독의 운영환경을 의미한다. 가속 알고리즘 적용 후는 리눅스 OS와 전용의 RTOS 병합하여 시스템의 소프트웨어 처리방식을 최적화 한 방식이다.

FCWS알고리즘 가속 적용전의 라이브 화면 송출시 21%~26%의 CPU를 점유하고 녹화 병행시 32%~47%로 CPU를 점유한다. 이때, 리눅스 OS 내부의 시스템 부하 및 태스크 스위치의 영향을 받아 내부 CPU의 점유율의 변동이 많이 발생하게 된다.

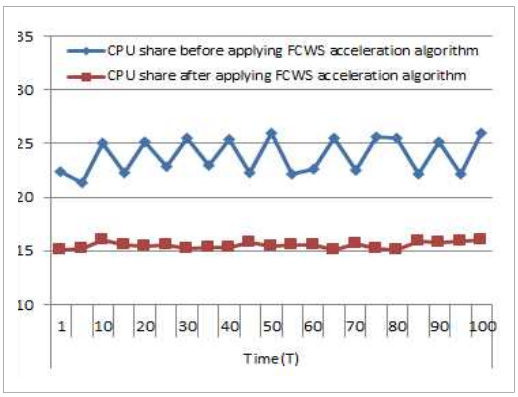

그림 10. 녹화 동작을 제외한 FCWS 알고리즘 구현 성능 비교
Fig. 10. Performance comparison of FCWS algorithm implementation except recording operation

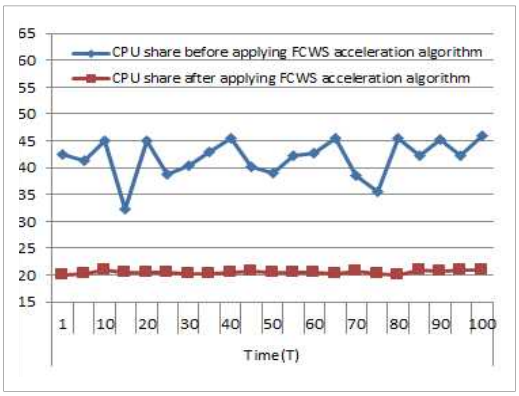

그림 11. 녹화 동작 중 FCWS 알고리즘 구현 성능 비교
Fig. 11. Performance comparison of implementation of FCWS algorithm during recording

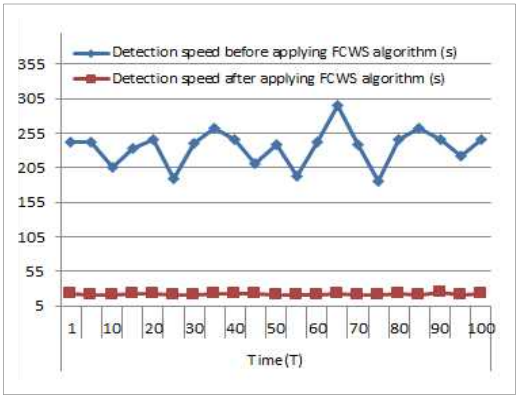

그림 12. FCWS 알고리즘 검출 속도 비교
Fig. 12. FCWS algorithm detection speed comparison

FCWS 알고리즘 가속 적용후의 라이브 화면 송출시 15%~17%의 CPU를 점유하고 녹화 병행시 20%~22%로 CPU를 점유한다. FCWS 알고리즘의 처리 동작이 분산됨에 따라서, RTOS에서 실시간으로 균등하게 CPU를 점유하고, 리눅스 OS의 자체 U/I 처리 동작도 안정화 되어 전체 시스템의 CPU 점유율이 평준화 된다.

FCWS 알고리즘 가속 적용전 검출속도는 결국 리눅스 OS의 동작속도에 영향을 받아 185~300ms의 지연이 발생하게 된다. FCWS 알고리즘 가속 적용후 검출속도는 20ms 내외의 균일한 속도로 처리되어 본 논문에서 예측한 바와 가이 실시간성을 보장하고 반응속도가 향상 된다.

V. 결 론

차량용 블랙박스 시스템과 같은 초소형 임베디드 시스템에 FCWS 알고리즘을 최적화해 구현하였다. 사고 발생시 경보동작에 대한 최적화가 구현되었다. 시스템의 녹화 동작 전 10~15%, 녹화 동작 후 20~25%정도 점유율 개선이 이루어 진다. 또한 알고리즘의 검출속도가 200~250ms 정도 향상되는 효과를 가져오고 경보에 대한 처리가 실시간으로 유효하게 활용이 가능하다.

본 논문을 확장 발전시킨다면, 쿼드 코어(Quad-Core) 기반의 SoC를 통해서, LDWS, FCWS과 FVSA(Forwad Vehicle Start Alert) 알고리즘을 각각의 CPU에 병렬로 구성하고, U/I 처리를 나머지 CPU에서 구현시킨다면, 효율적인 ADAS 알고리즘의 처리와 경고의 실시간성을 확보할 수 있을 것으로 보인다. 이러한 설계 및 구현 검증을 본 논문의 향후 과제로 한다.

References

[1] Dong-Gyu Jeong and Seon-Hyung Kim, "Intelligent Vehicle Status and Future Outlook", Journal of KIIT, Vol. 14, No. 2, pp. 21-26, Dec. 2016.

[2] Hyo-Kyun Jeong, Myung-Jin Lee, and Yong-Jin Jeong, "FPGA iimplementation of AVM-based lane departure warning system", Journal of KIIT, Vol. 12, No. 11, pp. 59-68, Nov. 2014.

[3] Seunghan Yang, Bongsob Song, and Jaeyong Um, "Radar and vision sensor fusion for primary vehicle detection", Journal of Institute of Control, Robotics and Systems, Vol. 16, No. 7, pp. 639-645, Jul. 2010.

[4] Felipe Jiménez, José E. Naranjo, Oscar Gómez, and José J. Anaya, "Vehicle tracking for an evasive manoeuvres assistant using low-cost ultrasonic sensors", Sensors 2014, Vol. 13, No. 10, pp. 22689-22705, Nov. 2014.

[5] Chung-Cheng Chiu, Meng-Liang Chung, and Wen-Chung Chen, "Real-Time front vehicle detection algorithm for an asynchronous binocular system", Journal of information science and engineering, Vol. 26, No. 3, pp. 735-752, May 2010.

[6] Longhui Gang, Mingheng Zhang, Xiudong Zhao, and Shuai Wang, "Improved genetic algorithm optimization for forward vehicle detection problems", Information 2015, Vol. 6, No. 3, pp. 339-360, Jul. 2015.

[7] Hwan-Shin Yu, "The lane departure warning algorithm optimized for automotive black-boxes and compact system", Journal of KIIT, Vol. 13 No. 10, pp. 9-15, Oct. 2015.

[8] Hwan-Shin Yu and Eui-Bung Jeoung, "The

optimized file system designed for vehicle black box system", Journal of KIIT, Vol. 14, No. 2, pp. 1-6, Feb. 2016.

[9] Hwan-Shin Yu and Eui-Bung Jeoung, "The lane recognition enhancement algorithms of around view monitoring system based on automotive black boxes", Journal of KIIT, Vol. 15, No. 1, pp. 45-53, Jan. 2017.

저자소개

유 환 신 (Hwan-Shin Yu)

1993년 2월 : 동국대학교 전자공학과(공학사)
2006년 2월 : 국민대학교 자동차전자제어(공학박사)
2006년 3월 ~ 현재 : 호원대학교 자동차기계공학과 교수
관심분야 : 무인자율차량, 센서시스템, 영상처리