논문 2021-2-4 http://dx.doi.org/10.29056/jsav.2021.12.04

정보기기 소스코드 유사성 분석에서 목적물 검증

남상엽*, 김도현**, 이규대***†

Comparison procedure in evaluation analysis of source code comparison on Embedded system

SangYep Nam*, Do-Hyeun Kim**, Kyu-Tae Lee***†

요 약

소스코드 감정목적물의 유사성을 판단하는 경우, 양측의 비교대상 소스코드는 컴파일과 실행이 가능 해야 한다. 또한 시스템 소스의 경우에는 하드웨어와의 인터페이스가 일치하는지 확인이 되어야 한다, 그러나 현실적으로 분쟁당사자의 소스코드가 완전하지 않은 상태로 제공되는 경우가 발생하고 있다. 분쟁을 제기하는 측에서는 동작 특성이 자사의 기능과 유사하고, 출력되는 표현과 기능이 유사한 이유로 소스코드의 핵심부분이 유사한 것으로 판단하고, 감정을 요청하고 있다. 분석과정에서 소스코드의 컴파일 오류 발생 시, 감정인은 소스코드의 흐름도를 작성하고, 함수별 코드의 흐름을 추적하는 방법을 적용하게 된다. 그러나 이 방법은 간접적이고, 감정인의 주관적인 판단이 적용되어 유사성 분석결과에 객관성에 대한 다툼이 우려된다. 본 연구에서는 임베디드 시스템에 적용된 소스코드분쟁의 감정사례를 통해 검증되지 않은 소스코드 유사성 도출의 문제점과 개선 방향을 다룬다.

Abstract

In order to analyze the similarity of the source code object material, the source code on both sides must be able to be compiled and executed. In particular, in the case of hardware-integrated software, it is necessary to check whether the hardware interface matches. However, currently, the source code is provided in an incomplete state which is not original of source code used in developing steps. The complainant confirms that the executing characteristics are similar to their own in the expression and function of the output, and request an evaluation. When a source code compilation error occurs during the evaluation process, the experts draw a flowchart of the source code and applies the method of tracing the code flow for each function as indirect method. However, this method is indirect and the subjective judgment is applied, so there is concern about the contention of objectivity in the similarity evaluation result. In this paper, the problems of unverified source code similarity analysis and improvement directions are dealt with, through the analysis cases of source code disputes applied to embedded systems.

한글키워드 : 소스코드, 목적물, 감정사례, 유사도, 라인비교

keywords : source code, object material, verification cases, similarity, line comparison

* 국제대학교 의료IT전공
** 국립제주대학교 컴퓨터공학과
*** 국립공주대학교 정보통신공학부
† 교신저자: 이규대(email: ktlee@kongju.ac.kr)
접수일자: 2021.11.27. 심사완료: 2021.12.15.
게재확정: 2021.12.20.

1. 서 론

임베디드 상용화 보드를 사용하는 정보기기는 소프트웨어와 하드웨어가 연동되는 구조로 개발된다. 개발자는 하드웨어의 신호체계에 따르는 프로그램과 데이터 처리에 사용되는 사용자 프로그램의 두 종류의 프로그램을 작성하여 임베디드 보드에서 함께 실행되도록 작성된다. 이러한 정보기기의 감정이 수행되는 경우에는 두 종류의 개발 프로그램 소스코드와 회로도가 감정 목적물로 함께 제공되어야 한다. 일반적으로 프로그램 소스코드 감정의 경우, 운영 소스코드만으로 감정 분석하는 것과 달리, 하드웨어관련 정보가 필요하고, 프로그램 작업도 하드웨어와 연동되는 소스코드로 개발되는 특징이 있다[1].

감정목적물의 분석은 제공된 자료를 중심으로 수행되기 때문에, 목적물은 개발자가 개발과정에서 적용하였던 소스코드와 회로보드가 제공되어야 하며, 유사성 분석을 하는 전문가의 분석과정에서 컴파일이 가능하고, 시스템보드 상에서 실행되는 조건을 만족해야 객관적인 감정분석이 가능하다.

그러나 기존의 감정사례에서와 같이 감정을 의뢰하는 고소인과 저작물 도용으로 의심되는 피고소인 측에서 제공하는 목적물을 보면, 개발시점에서 수년이 지난 후 도용이 의심되는 저작물이 확인되는 이유로, 원 개발자와 도용이 의심되는 개발자의 자료가 분실되거나, 보관상태가 불완전하여 프로그램 소스코드의 컴파일이 가능하지 않거나, 실행코드만 보관 되어, 소스코드 목적물이 제공되는 않는 경우가 발생하고 있다[2].

소프트웨어 소스코드 감정은 기본적으로 라인단위의 비교분석방법이 사용되는데, 이는 코딩과정에서 변수의 이름과 각종 명령문의 사용방법이 개발자 마다 고유한 특성으로 작성되기 때문에 라인단위 비교방법으로 원 개발자의 소스코드가 도용되었는지, 독자적으로 개발되었는지 판별이 되기 때문이다. 또한 공개 소프트웨어 소스코드나 함수가 사용되는 경우에도 이를 변형하여 사용하는 개발자의 창작성을 확인할 수 있다. 이러한 이유로 도용이 의심되는 피고소인 측에서는 자료제출에 비협조적이고, 소스코드의 변형 및 실행코드를 제출하는 방법으로 완전한 소스코드 제출을 회피하는 것으로 의심되고 있다.

감정수행은 위와 같이 완전한 소스코드 목적물이 제공되어야하지만, 실제적으로 부족한 자료에 의한 분석이 수행되어야 하는 상황이 있으며, 이 경우 분쟁 제품의 특성과 개발자의 입장에서 구현 가능한 과정을 추적함으로써 도용의 의심이 되는 부분을 추적하는 방법으로 분석이 진행되며, 기존의 라인단위 비교 분석은 객관적인 분석결과로 활용하는데 어려움이 있다.

본 연구에서는 감정목적물의 분석과 컴파일이 가능하지 않은 소스코드의 감정사례를 통해 소스코드의 분석과 소스코드의 유사성 분석 결과를 예시하고, 제공된 소스코드의 실행 흐름과정을 추적하는 방법으로 유사성을 분석하는 과정을 개선점으로 제시하였다.

2. 프로그램 소스코드 목적물

프로그램 소스코드는 개발자의 저작권이 포함되는 핵심자료로서, 일반적인 운영체제를 기본으로 하는 컴파일러 환경에서 개발되는 코딩결과를 의미하지만, 임베디드 시스템과 같이 하드웨어와 소프트웨어가 연동되는 프로그램 소스코드는 하드웨어 센서, 입출력신호체계와 인터페이스 되는 디바이스 드라이버 단계의 소스코드작성을 필요로 한다. 제품의 개발 목적에 부합되는 기능상의 블록도가 작성되고, 소프트웨어 흐름도가 그림 1과 같이 작성되면, 개발자는 특정 프로그램언어

를 사용하여 소스개발의 결과물을 작성한다. 이 과정에서 개발자가 사용하는 프로그램언어의 선택 및 하드웨어로 선택한 개발보드의 프로세서 종류 및 인터페이스 신호선택도 개발자의 고유 저작물 특성에 해당된다. 선택한 프로그램 언어와 컴파일러가 정해지면, 개발 제품의 기능 흐름도(flowchart)에 따라 단계별로 소스코딩 작업을 진행하고, 디버깅 작업으로 프로그램 오류를 정정하고, 실행코드로 완성되면, 하드웨어 보드에 저장(download)으로 개발이 종료된다. 이후 사용자는 제품의 화면에 나타나는 디스플레이 상태 및 입출력 버튼의 조작 순서, 출력되는 신호의 관찰 등으로 제품의 성능을 확인한다[3][4].

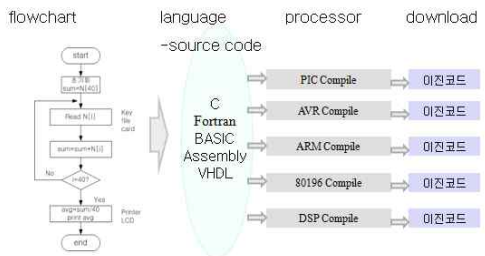

그림 1. 시스템 프로그램 작업 흐름도
Fig .1. Flowchart of system program

감정사례의 감정목적물은 STM(stm.com)의 8비트 프로세서 칩을 제어기로 사용하는 보일러용 온도 및 장치 제어용 프로그램이다. 프로그램 작성은 IAR Embedded workbench for STM8(iar.com)을 사용하여 소스코드 개발 및 컴파일과 실행코드로 개발된다[5]. 프로그램은 그림 2와 같이 제어기의 순서도를 작성하고, 흐름도에 따른 소스코드를 IAR 개발도구에서 C언어로 작성하여 개발되었다. STM8 프로세서용 개발도구에서 작성한 소스코드를 컴파일 결과로 실행코드를 작성하고, 제어기 보드로 다운로드(download)하면 제어기 패널 조작에 의해 감정목적물에 적합한 보일러 제어기로 작동하게 된다. 개발환경에서 작성된 실행코드는 개발목적의 보일러제어기 보드에 삽입되어 동작한다. 제어기는 그림 2와 같이 디스플레이 화면과 입력용 버튼, 제어대상 방의 선택과 온도설정용 버튼 등이 위치하는 구조로 개발되었다.

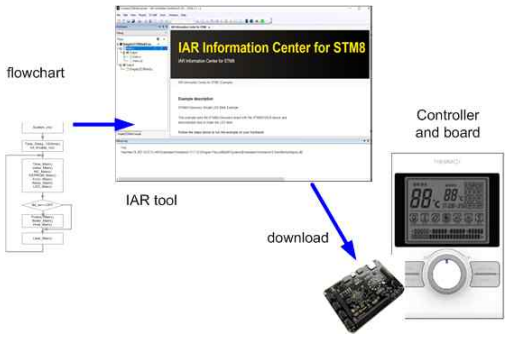

그림 2. 정보기기 내부 구성도
Fig. 2. Development flow of Embedded System

감정목적물로 제공된 소스코드는 프로그램의 사양서나 명세서가 포함된지 않은 상태로 제공되었다. 본 사례로 사용된 감정목적물은 A사가 보관중인 소스프로그램(A), A사로부터 B사가 개발중에 제공받은 것으로 제공된 소스코드(B), B사가 C사와 공동으로 협력하여 작성한 소스프로그램(C)의 3종류 소스프로그램이 감정목적물로 지정되었다. C 프로그램은 B사와 C 사가 공동 개발한 소스코드로 C사가 개발한 소스프로그램부분은 감정에 포함되지 않은 것으로 결정되어, C 프로그램 소스는 B사가 개발한 것으로 지정된 소스코드가 제공되었다. 목적물의 확보 단계에서 나타나는 문제점으로는 감정목적물 지정 시점에서 분쟁 대상 업체의 개발자와 참여자가 퇴사나 이직 등으로 존재하지 않거나, 개발과정에서 작성된 상세 문서가 보관되지 않아 소스프로그램 모듈에 대한 설명과 기능에 대한 자료가 보관되어 있지 않는 경우가 발생하고 있다[6].

3. 목적물의 분석

감정목적물은 원고가 초기 개발한 프로그램 소스코드(A), 원고가 피고에게 제공한 프로그램 소스코드(B), 피고가 자체 개발한 프로그램 소스코드(C)의 3부분으로 제공되었다.

1) 원고측에서 제공한 목적물은 그림 3과 같이 [1.감정목적물(원고제출)] 폴더에 포함되었으며, 여러 종류의 파일이 포함되어 있었지만, 감정에 필요한 C언어의 확장자 파일(*.c)과 헤더파일(*.h)만을 정리하여 그림과 같이 6개 폴더에 91개 파일로 정리하여 총 41411개 소스 라인으로 재구성하였다.

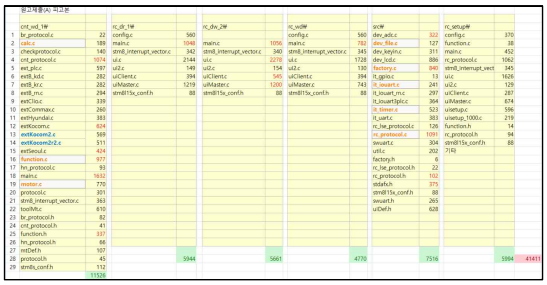

그림 3. 프로그램 파일(A)의 파일목록
Fig. 3. File structure of progam(A)

2) 원고가 피고에게 제공한 소스코드로 제공된 목적물은 그림 4와 같이 [2.감정목적물(피고제출1)] 폴더에 포함되었으며, 감정에 필요한 C언어의 확장자 파일(*.c)과 헤더파일(*.h)만을 정리하여 그림과 같이 5개 폴더에 72개 파일로 정리하여 총 33796개 소스 라인으로 재구성하였다.

3) 피고가 자체개발 소스코드로 제공된 목적물은 [2.감정목적물(피고제출2)] 폴더에 포함되었으며, 감정에 필요한 C언어의 확장자 파일(*.c)과 헤더파일(*.h)만을 정리하여 36개 파일로 정리하여 총 19469개 소스 라인으로 재구성하였다.

그림 4. 프로그램 파일(B) 파일목록
Fig. 4. File structure of progam(B)

감정목적물은 표1과 같이 3가지 종류의 프로그램 소스코드에 대해 폴더 수, 파일수, 소스라인 수를 분석한 내용으로 정리되었다.

표 1. 감정목적물의 소스라인 수
Table 1. Source lines of programs

	폴더수	파일수	소스라인 수
프로그램A	6	91	41411
프로그램B	5	72	33796
프로그램C	1	36	19469

4. 유사성 분석

프로그램 소스코드의 유사성 분석은 프로그램 언어가 라인단위의 문서 형식을 작성되는 특징이 있고, 라인단위로 표현되는 변수명과 명령어의 사용방법이 프로그램 개발자의 고유 기술로 차별되기 때문에 저작권의 성격을 갖게 된다. 따라서 유사성 비교의 기본은 양측의 유사라인을 분석하여, 전체 프로그램 소스라인 대비 유사하거나 동일하게 분석되는 라인수를 비율로 계산하여 유사성을 판별하는 방법을 적용한다. 비교대상의 목적물 소스코드가 폴더의 구성과 파일의 이름 및 파일 수가 매칭되는 경우에는 일대일 비교방법을

적용하게 되며, 그림 5와 같은 상용프로그램을 사용하여 유사성을 분석한다[7].

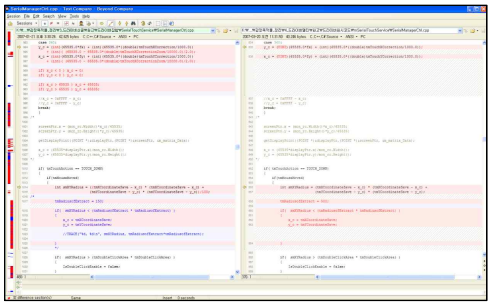

그림 5. Beyond Compare 실행 화면
Fig. 5. Beyond Compare window

파일의 이름과 개수가 상이하여, 일대일 매칭에 의한 비교가 불가능한 경우에는 다대다 비교 방식으로 원본 화일 1개에 대해 비교본 파일 전체를 비교하는 과정으로 가장 유사한 파일을 선별하여, 일대일 비교를 하는 방법을 사용한다[8][9]. 주로 운영체제를 기반으로 하는 프로그램 소스코드와 같이 대규모 개발프로그램의 경우에 해당되며, 저작권위원회에서 개발한 exEyes 도구로 그림 6과 같이 유사화일 검출이 가능하다.

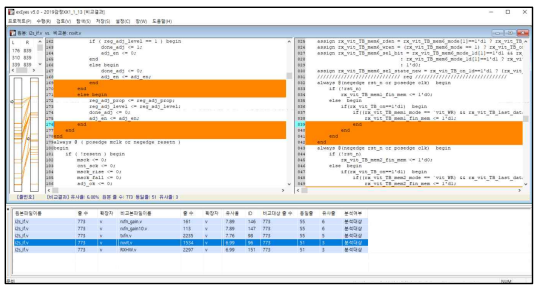

그림 6. exEyes v5.0 실행화면
Fig. 6. exEyes v5.0 window

4.1 유사도 분석 결과

유사도 분석은 프로그램코드 A와 B의 동일 유사성 비교를 수행하였으며, 이는 양측의 보관 소스코드의 변형 및 동일성을 확인하는 작업으로 선행되었다. 동일한 소스코드에 대한 변형가능성을 검사하는 과정으로 파일의 개수와 이름이 이 동일한 구조로 되어있고, 원본A와 비교본B에 대해 각 폴더에 포함된 파일마다 구성된 라인수를 확인하고, 비어있는 라인은 제거하여, '공백 제거 줄 수'로 정제한 결과로 비교분석을 진행하였다. 표 2와 같이 97.6%의 결과로 두 프로그램 소스코드는 유사성이 높은 것으로 판단되었다.

표 2. 프로그램(A-B) 비교결과
Table 2. Comparison result with(A-B)

폴더명(B)	파일수	라인수
B_1	27	94.6%
B_2	8	99.9%
B_3	22	98.52%
합계	57	97.6%

유사성 검출 도구(exEyes)를 사용하여 비교분석 결과는 표 3과 같이 동일 또는 유사한 것으로 추출된 3,615 라인이 확인되어, 11.92%의 유사성을 갖는 것으로 분석되어 유사성이 없는 것으로 판단되었다.

표 3. 프로그램(A-C) 비교결과
Table 3. Comparison result with(A-C)

폴더명(A)	C	유사도
A_1	36개 파일	9.3%
A_2	36개 파일	15.4%
A_3	36개 파일	11.07%
합계		11.92%

4.2 프로그램 순서도 분석

저작물의 기능과 사용 순서 및 디스플레이 상태가 유사함에도 유사성 분석결과가 위와 같이 10%내외로 나타나는 경우에는 소스코드의 함수별 분석으로 실행 흐름도를 작성할 수 있다. 그림 7과 같이 추가적으로 목적물에 포함된 소스코드를 함수별 분류하고, main() 함수를 시작으로 함수호출관계도를 추적하는 것이 가능하다. 그림 7은 프로그램 소스코드(A)에 해당되는 흐름도를 추적한 것으로, 비교대상인 목적물 소스코드(C)는 전체프로그램이 제공되지 않아 상대적 비교가 가능하지 않았으나, 이 방법으로 간접적인 저작권 분석에 활용가능하다.

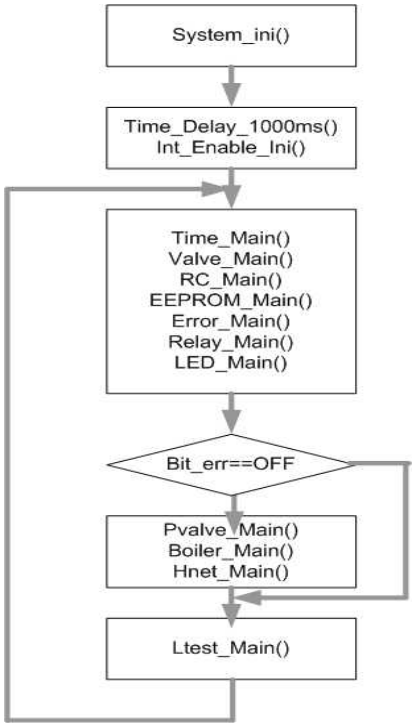

그림 7. 프로그램(A) 순서도
Fig. 7. Flow diagram of program (A)

사례분석 결과는 프로그램(A-B)는 유사하고, 프로그램(A-C)는 11.92%의 유사성 분석 결과로 상이한 것으로 판단되었다. 또한 상이한 결과의 보완 작업으로 소스코드 흐름도를 작성하여, 라인단위 비교결과의 오류를 보완하는 과정을 보였으며, 소스코드 비교감정에 활용이 가능할 것으로 보인다.

저작권 보호를 위한 프로그램 소스코드 감정은 복사와 도용을 방지하고, 불법 사용자에 대해서는 객관적인 증거자료로 활용되는 결과물로 효과가 있다. 따라서 객관적이고, 정확한 분석 데이터는 분쟁의 해결에 도움이 되고, 불법 사용자를 방지하는 예방효과를 동반하는 장점이 있다. 본 연구 결과와 같이 라인단위 검증에 추가하여, 함수 흐름도 분석의 병행으로 객관적인 감정분석 결과의 도출이 기대된다.

5. 결 론

임베디드 보드를 사용해서 제작되는 정보기기는 하드웨어와 소프트웨어가 연동되어 실행되는 특징으로 감정목적물 제공시 하드웨어관련 자료와 소프트웨어의 개발단계 소스코드가 포함되어야 한다. 또한 소스코드의 유사성 분석을 위해서는, 양측의 비교대상 소스코드가 감정인의 분석환경에서 컴파일과 실행이 가능 해야 한다. 그러나 현실적으로 분쟁당사자의 소스코드가 완전하지 않은 상태로 제공되는 경우가 발생하고 있다. 분쟁을 제기하는 측에서는 동작 특성이 자사의 기능과 유사하고, 출력되는 표현과 기능이 유사한 이유로 소스코드의 핵심부분이 유사한 것으로 판단하고, 감정을 요청하고 있다. 본 연구에서는 감정사례를 통하여 소스코드의 컴파일 오류 발생시, 소스코드의 흐름도를 작성하고, 함수별 코드의 흐름을 추적하는 방법을 제시하였다. 그러나 이 방법은 간접적이고, 불완전한 소스코드로 알고리즘의 진행을 완전하게 확인할 수 없는 한계를 갖는다. 이 경우 감정인의 주관적인 판단이

적용되어 유사성 분석결과에 대한 객관성이 부족할 수 있다. 결과적으로 소스코드 감정의 경우, 양측의 목적물은 개발단계에서 최종적으로 작성된 것으로 검증된 소스코드 및 제출 단계에서 컴파일과 실행이 확인된 것을 입증하는 자료나, 영상으로 함께 제출하는 제도상의 보완이 선행되어야 할 것으로 보인다.

참 고 문 헌

[1] Rajesh K. Gupta, “Introduction to Embedded system”, ICS 212, 2002 winter workshop

[2] Kyu-Tae Lee, Hyun-Chang Lee, Jang-Geun Ki, “Establishment of the Subtitle on Materials for Evaluating Intellectual Ownership”, International Journal of Signal Processing, Image Processing and Pattern Recognition. 2017 Sep; 10(9): 79-88. http://www.sersc.org/journals/IJSIP/vol10_no9/IJSIP%20ToC%20September%202017.pdf

[3] Do-Hyeun Kim, KyuTae Lee, “Management of Reliability and Delivery for Software Object Material”, Journal of Software Assessment and Valuation (JSAV), vol.15, No.2, pp.51-57. Dec. 2019. http://dx.doi.org/10.29056/jsav.2019.12.07.

[4] 이규대, “유사성 비교에서 세부항목 설정 기준”, 한국소프트웨어감정평가학회 논문지, 12권1호, pp21-26, June, 2016. http://www.i3.or.kr/html/paper/2016-1/(3)2016-1.pdf

[5] 이규대, “임베디드시스템의 이진코드 추출 및 분석”, 한국소프트웨어감정평가학회 논문지, 5권1호, pp27-38, May, 2009.

[6] 이규대, 권기영, “정보기기 감정에서 세부항목 설정 사례”, 한국소프트웨어감정평가학회 논문지, 12권2호, pp9-14, Dec. 2016. http://www.i3.or.kr/html/paper/2016-2/(2)2016-1.pdf

[7] Compare files and folders [Internet], 2018 [updated 2018 Oct 10; cited 2018 Oct 10]. http://www.scootersoftware.com/features.php?zz=features_focused (website)

[8] Raj Kamal, Embedded systems Architecture Programming and Design. 2nd ed. MacGraw Hill Companies; 2015. p.5.(Book) ISBN 0-07-049470-3

[9] Ali, Safdar, and DoHyeun Kim, “Building power control and comfort management using genetic programming and fuzzy logic”, Journal of Energy in Southern Africa 26.2 (2015): 94-102. http://www.scielo.org.za/scielo.php?script=sci_arttext&pid=S1021-447X2015000200010&lng=en&nrm=iso

[10] 김도현, 이규대, “실행코드 비교 감정에서 주변장치 분석의 유효성”, 한국소프트웨어감정평가학회 논문지, 16권1호, pp.37-44, June, 2020. http://dx.doi.org/10.29056/jsav.2020.06.05.

저 자 소 개

남상엽(SangYep Nam)

2002 단국대 전자공학과 박사
2013-2014 대한전자공학회 산업전자(소) 회장
1998~현재 국제대학교 의료IT전공 교수.
<주관심분야> 임베디드 소프트웨어, 로봇 시스템 프로그램

김도현(Do-Hyeun Kim)

2000.8 경북대 전자공학과 박사
2004.9~현재 국립제주대학교 컴퓨터공학 전공 교수.
<주관심분야> 사물인터넷, 예측 및 최적 제어, 모바일 컴퓨팅, 임베디드 소프트웨어

이규대(Kyu-Tae Lee)

1991 고려대 전자공학과 박사
1992~현재 국립공주대학교 정보통신공학부 교수
<주관심분야> 신호처리, VLC, 저작권보호, 임베디드 시스템, 상황인식 및 학습