Journal of The Korea Society of Computer and Information
Vol. 20 No. 11, pp. 39-46, November 2015

www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.11.039

Performance Tradeoff Between Control Period and Delay: Lane Keeping Assist System Case Study

Hyun-Jun Cha*, Seong-Woo Park**, Woo-Hyuk Jeong***, Jong-Chan Kim****

Abstract

In this paper, we propose a performance-aware workload model for efficient implementation of control systems. When implementing a control algorithm as an embedded computer system, the control code executes periodically. For such systems, its control performance depends on not only the accuracy of the control algorithm itself but also temporal parameters such as control period and sensing to actuation delay. In this regard, this paper studies the relation between control period and delay by measuring and analyzing the control performance of LKAS (Lane Keeping Assist System) with varying period and delay combinations. Through this experimental study, this paper shows that the two timing parameters, i.e.,control period and delay, has a tradeoff relation in terms of control performance.

▸ Keyword : Period, Delay, Scheduling, Control Performance, LKAS

I. Introduction

자동차는 더 이상 기계적인 부품으로만 구성되지 않는다. 최근의 자동차는 내부에 70대 이상의 제어 컴퓨터 즉 ECU (Electronic Control Unit)가 5개 이상의 네트워크로 연결된 복잡한 컴퓨터 시스템이다 [1-2]. ECU는 다양한 차량 센서로부터 수집된 정보를 바탕으로 차량을 제어하기 위한 판단을 하고 그 결과를 알맞은 엑추에이터를 통해 차량의 움직임에 반영한다. ECU 내부의 제어 소프트웨어는 차량용 실시간 운영체제 위에서 실행된다 [3]. 실시간 운영체제의 태스크로 구현된 제어 알고리즘은 일반적으로 주기적으로 실행되며 "센싱-판단-엑추에이션" 작업을 반복적으로 실행한다.

제어 시스템이 원하는 제어 성능을 만족하기 위해서는 제어 알고리즘의 정확성이 필수적이지만 그 외에도 제어 태스크의 제어 주기 역시 적절하게 설정해야 한다. 즉, 제어 주기가 짧을수록 제어 성능은 좋지만 제어 태스크를 자주 실행함으로 인한 하드웨어 자원 요구량이 커지기 때문에 제어 주기를 결정할 때는 원하는 제어 성능과 하드웨어 자원 요구량 사이의 트레이드오프 (tradeoff) 관계를 고려해야 한다 [4]. 제어 주기에 더해서 제어 성능에 큰 영향을 미치는 요소로 지연시간이 있다. 제어 태스크의 지연시간은 센서 정보가 취득된 시점부터 실제 엑추에이션까지의 시간 간격으로 정의된다 [5]. 지연시간이 짧은 경우, 즉 빠르게 판단이 끝난 경우, 엑추에이션이 최근의 센서 정보를 바탕으로 하기 때문에 정확한 제어값이 반영된다. 반대로 지연시간이 긴 경우는 엑추에이션 값이 근거로 하는 센서 정보가 이미 상당히 과거의 정보이기 때문에 제어의 성능이 떨어지게 된다 [6].

이처럼 제어 시스템의 성능은 제어 알고리즘 자체의 정확도 외에도 실제 구현 시 결정되는 타이밍 요소인 제어 주기와 지연시간에 의해서도 많은 영향을 받게 된다. 즉, 똑같은 제어 알고리즘을 사용하더라도 제어 태스크의 제어 주기가 짧을수록 그리고 지연시간이 짧을수록 더 높은 제어 성능을 보이게 된다. 특히 복수의 제어 태스크가 하나의 하드웨어 자원을 공유하는

• First Author: Hyun-Jun Cha, Corresponding Author: Jong-Chan Kim
*Hyun-Jun Cha (chahjun@kookmin.ac.kr), Graduate School of Automotive Engineering, Kookmin University
**Seong-Woo Park (qkrtjddn938@kookmin.ac.kr), Department of Computer Science, Kookmin University
***Woo-Hyuk Jeong (jwh4807@kookmin.ac.kr), Department of Computer Science, Kookmin University
****Jong-Chan Kim (jongchank@kookmin.ac.kr), Graduate School of Automotive Engineering, Kookmin University
• Received: 2015. 10. 24, Revised: 2015. 11. 02, Accepted: 2015. 11. 10.
• This work was supported by the new faculty research program "Component-based Control Kernel Development" of Kookmin University in Korea.

멀티태스킹 환경의 경우 제어 태스크의 지연시간은 해당 태스크가 실제 하드웨어 자원을 사용하는 실행시간 외에도 다른 태스크의 실행에 의해서 대기되는 시간까지 포함하기 때문에 무시하지 못할 정도로 커질 수 있다.

본 연구는 위에서 설명한 제어 주기와 지연시간이 제어 시스템의 성능에 미치는 영향을 정량적 측정을 통해 파악하고 그 둘 사이의 관계를 분석하고자 한다. 이를 통해 원하는 제어 성능을 달성하기 위하여 제어 주기와 지연시간을 어떻게 조절해야 하는지 파악해 본다.

본 논문의 구성은 다음과 같다. 2장에서 관련 연구를 제시하고 3장에서는 제어 주기와 지연시간 그리고 둘 사이의 관계를 설명한다. 4장에서는 실험 환경 구현에 대해서 설명하고 5장에서는 실험 결과를 제시한다. 마지막으로 6장에서는 논문의 결론을 내리고 향후 연구 과제를 제시한다.

II. Related Work

제어 주기와 제어 성능의 연관 관계를 이용하여 [7-8]은 복수개의 제어 태스크에 대해서 전체적인 제어 성능을 최적화하는 태스크 주기를 결정하는 방법을 제시하였다. [9]는 [7-8]을 확장하여 실질적인 크기의 태스크 집합에 대해 적용 가능한 제어 주기 최적화 방법을 제안하였다. [10]은 분산 시스템의 종단간 (End-to-End) 트랜잭션으로 정의된 제어 시스템의 주기를 최적화 하였고 [11]은 태스크 주기를 최적화하여 하이퍼주기 (Hyperperiod)를 최적화 하였다. 하지만 [10]과 [11]은 제어 성능을 최적화에서 고려하지 않았다. [5]는 스케쥴링으로 인한 지연시간이 제어 성능에 미치는 영향을 같이 고려하여 제어 주기를 최적화 하였으나 제어 주기와 지연시간의 상호 관계를 고려하진 않았다.

[6]은 제어 주기와 지연시간을 같이 고려하여 제어 성능을 분석적으로 예측할 수 있는 시뮬레이션 소프트웨어를 제안하였다. [12]는 태스크 주기와 지연시간의 관계가 미분 가능 함수로 주어졌을 때 빠르게 스케쥴링 가능한 (주기, 지연시간) 쌍의 집합을 찾는 방법을 제안하였다. [13]은 지연시간이 제어 성능에 미치는 영향을 분석적 방법으로 결정한 후 먼저 최적의 제어 주기를 찾고 이를 기반으로 태스크의 최적 마감시간을 찾는 두 단계의 휴리스틱 알고리즘을 제안하였다. 앞의 연구들은 제어 주기와 지연시간이 제어 성능에 미치는 영향을 분석하였지만 제어 주기와 지연시간이 제어 성능에 미치는 영향을 정량적으로 분석하고 모델링 하지는 못하였다.

앞에서 설명한 기존 연구와는 달리 본 연구의 목적은 제어 주기와 지연시간이 제어 성능에 미치는 영향을 정량적으로 측정하여 그 관계를 밝히고 이 관계를 이용하여 원하는 제어 성능을 얻을 수 있는 방법을 파악하는 것이다.

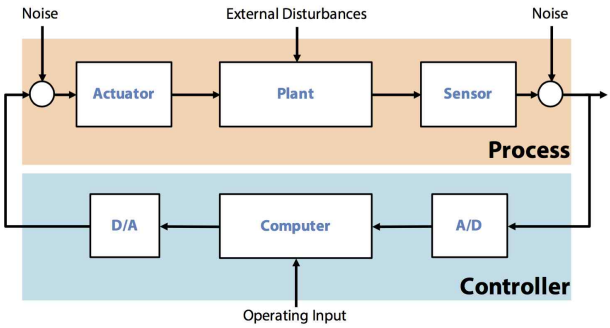


Fig. 1. General Control System Overview

III. Tradeoff Between Control Period and Delay

1. Control Period and Delay

그림 1은 일반적인 제어 시스템의 구조를 나타낸다. 상단의 Process 부분의 Plant가 제어 대상에 해당한다. 제어 대상과 그 주변 상황에 대한 정보는 센서에 의해서 수집되어 A/D 변환을 통해 하단의 Controller 부분의 컴퓨터 (Computer)에 디지털 신호로 전달된다. 이를 바탕으로 컴퓨터가 결정한 제어 값은 D/A 변환 후 엑추에이터를 통해 제어 대상에 반영된다. 센서와 엑추에이터의 작동에는 외부 노이즈가 개입할 수 있으며 Plant는 외부 환경의 변화에 의해서 직접적인 영향을 받는다. 또한 시스템 이용자는 사용자 입력을 통해 제어 방식을 운영 중에 변경할 수 있다.

이와 같은 제어 시스템에서 컴퓨터의 역할은 센서 값의 도착부터 시작하여 제어 판단에 의해 엑추에이션 값을 결정하면서 끝나게 된다. 이 과정은 일반적으로 일정한 주기를 가지고 반복적으로 실행된다. 그림 2는 이와 같은 주기적인 제어를 나타낸다. 일정한 주기 (Period)마다 센서 값이 컴퓨터에 도착하면

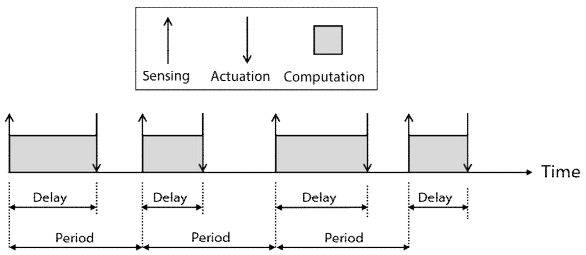


Fig. 2. Control Period and Delay (Single-Tasking)

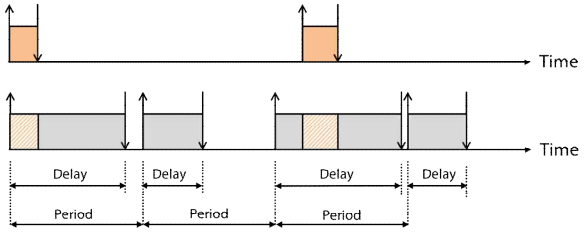


Fig. 3. Control Period and Delay (Multi-Tasking)

(Sensing) 이를 바탕으로 계산 과정을 거쳐 (Computation) 엑추에이션 값이 제어 대상에 반영되는 (Actuation) 과정이 반복적으로 실행된다. 결과적으로 각 제어 작업은 센싱부터 엑추에이션까지 지연시간 (Delay)을 겪게 된다.

그림 2와 같이 CPU가 하나의 태스크를 실행하는 싱글태스킹 환경의 경우 지연시간은 온전히 해당 태스크가 CPU를 차지하는 시간만으로 구성되지만 그림 3과 같이 복수개의 태스크가 하나의 CPU에서 실행되는 멀티태스킹 환경에서의 지연시간은 다른 태스크에 의해서 지연되는 시간까지도 포함한다. 그림 3의 경우 두 개의 태스크가 하나의 CPU를 공유하는 시나리오를 보여주고 있는데 하단의 태스크의 지연시간은 회색으로 표현된 실행시간에 더해서 상단의 태스크에 의해서 지연되는 빗금 친 부분까지 포함한다. 주기적으로 실행되는 태스크에 대해서 이를 모두 고려하여 최악의 지연시간을 계산하는 방법은 이미 밝혀져 있다 [14].

2. Impact of Control Period and Delay on Control Performance

제어 시스템의 성능은 여러 가지 방법으로 정의할 수 있다. 제어의 안정성 (stability), 외란 (disturbance)에 대한 강인성 (robustness), 과도 응답 (transient response) 시간, 안정 상태 정확도 (steady-state accuracy) 등이 모두 제어 성능의 지표로 사용할 수 있는 항목들이다 [15]. 위와 같은 성능 지표 외에도 제어 대상의 상태가 얼마나 원하는 제어 목표 값에 가까운지를 나타내는 시스템 에러 (system error) 또한 가장 중요한 제어 성능 항목이다 [15]. 본 논문에는 시스템 에러를 제어 시스템의 성능 지표로 삼아서 연구를 진행하였다. 다른 성능 항목 역시 같은 방법으로 분석이 가능하다.

위에서 설명한 제어 성능을 차량의 조향을 제어하여 차량을 차선 중앙으로 주행하도록 횡방향 제어를 하는 LKAS의 경우를 예로 들어 설명해 보겠다. LKAS 시스템은 그림 4와 같이 카메라를 통해 차선을 인식하여 상대적인 차량의 횡방향 위치를 파악한 후 차량이 차선을 지키기 위한 조향각을 결정하고 MDPS (Motor Driven Power Steering) 엑추에이터를 통해 조향각을 조절한다. 이때 제어 성능은 차량의 중심점 (Vehicle Center Point)을 얼마나 차선의 중앙선 (Lane Center Line)에 가까이 유지하느냐로 정의할 수 있다. 이때 이 둘 사이의 최소 직선 거리를 횡방향 에러 (Lateral Error)라 하고 이 값의 최대 측정값이 LKAS 시스템의 제어 성능으로 정의된다.

III.1 절에서 설명한대로 일반적으로 제어 알고리즘은 주기적으로 실행되는데 주기가 짧을수록 더 좋은 제어 성능을 유지할 수 있다. 예를 들어 LKAS의 경우 주기적으로 횡방향 에러를 줄이기 위한 조향각을 결정하여 반영하는데 만약 제어 주기가 너무 길게 되면 횡방향 에러가 0이 된 후에도 계속 같은 방향으로 진행하여 오히려 횡방향 에러가 커질 수 있다. 그렇기 때문에 일정 수준에서 제어 주기를 짧게 유지해야 횡방향 에러를 원하는 수준 이하로 유지할 수 있다. 그림 5는 같은 제어 알고리즘을 (a) 짧은 주기로 실행한 경우와 (b) 긴 주기로 실행한 경우를 비교하고 있다. (a)의 경우 첫 번째 제어 결과 우측으로 조향을 하게 되는데 차량이 우측 차선으로 빠져나가기 전에 다시 제어 결정을 내리기 때문에 차선을 이탈하지 않으려고 두

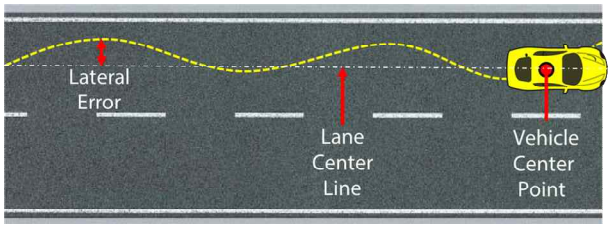


Fig. 4. LKAS System

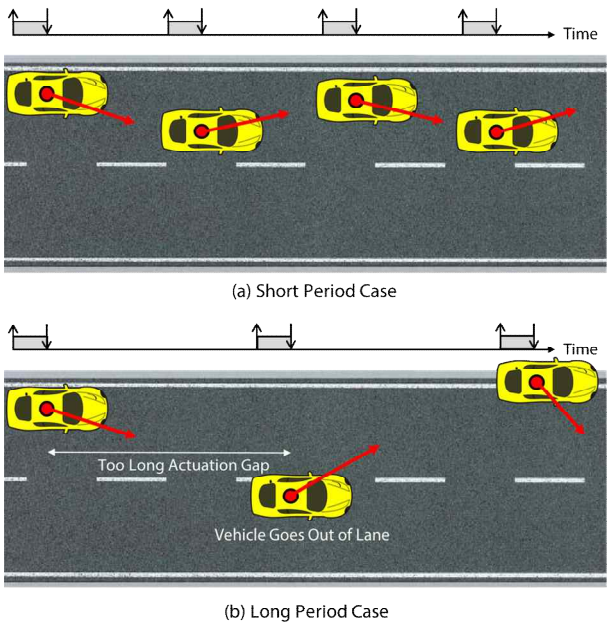


Fig. 5. Performance Impact of Control Period

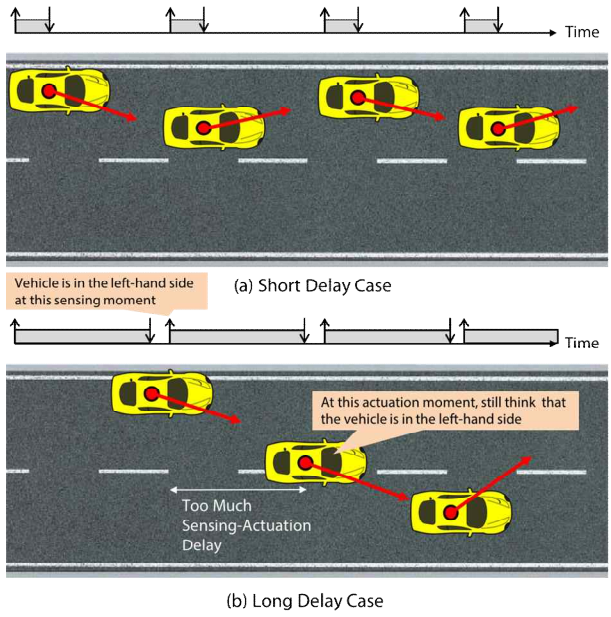


Fig. 6. Performance Impact of Sensing-Actuation Delay

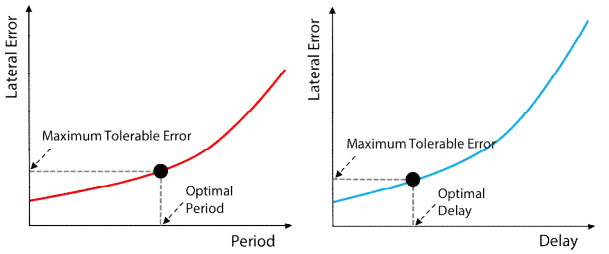
번째 제어는 좌측으로 조향을 하게 된다. 마찬가지로 세 번째와 네 번째 제어도 역시 적절하게 이루어지면서 차량이 차선을 벗어나지 않는다. 반면에 (b)의 경우 첫 번째 제어는 똑같이 우측으로 조향을 하지만 제어 주기가 너무 길기 때문에 두 번째 제어가 이루어지기 전에 차량이 이미 우측 차선을 지나쳐서 주행 차선을 이탈하게 된다. 뒤늦게 좌측으로 조향을 하지만 세 번째 제어도 너무 늦어지면서 이번에는 왼쪽 차선을 지나쳐서 차량이 차선의 좌측으로 이탈하게 된다. 그림 5의 예제가 보여주는 것처럼 제어 주기는 제어 성능에 직접적인 영향을 끼친다.

이에 더해 센싱에서 엑추에이션까지의 지연시간도 제어 성능에 큰 영향을 미친다. 지연시간이 짧으면 엑추에이션 시점의 제어값이 최근의 센서값을 기초로 하기 때문에 정확한 제어가 이루어지는 반면에 지연시간이 길어지면 엑추에이션 시점의 제어값이 먼 과거의 센서값을 근거로 하기 때문에 잘못된 제어가 이루어질 확률이 높다. 그림 6은 이와 같은 상황을 제어 주기는 동일하지만 (a) 지연시간이 짧은 경우와 (b) 지연시간이 긴 경우를 비교해서 보여준다. (a)의 경우 센서 정보 취득부터 엑추에이션까지의 지연시간이 짧기 때문에 차선 안에서 정상적으로 제어가 이루어지지만 (b)의 경우 두 번째 제어 시점을 보면 센서 정보가 취득되는 시점에는 차량이 차선의 좌측에 위치했기 때문에 계속해서 우측으로 조향할 것을 엑추에이터에 전달하지만 센싱부터 엑추에이션까지 지연시간이 너무 길기 때문에 엑추에이션 시점에는 이미 차량이 우측 차선을 넘어가고 있다. 세 번째 제어에서 이 사실을 인지하고 조향을 좌측으로 변경하지만 이미 차선을 넘어간 이후이다.

그림 7은 앞에서 LKAS의 예를 들어 설명한 제어 주기와 지연시간이 제어 성능에 미치는 영향을 일반화된 그래프로 보여준다. 그림 7(a)에 나타난 것처럼 제어 주기가 증가하면 그에 따라 제어 에러값이 증가하게 된다. 즉 제어 성능이 저하된다. 이 관계를 이용하면 한계 제어 성능이 주어졌을 때 그에 따라 적절한 최적 제어 주기를 결정할 수 있다. 그림 7(b)는 지연시간이 제어 성능에 미치는 영향을 보여준다. 지연시간이 증가하면 그에 따라 제어 에러가 증가할 것이며 주어진 한계 제어 에러에 따른 최적 지연시간을 찾을 수 있다.

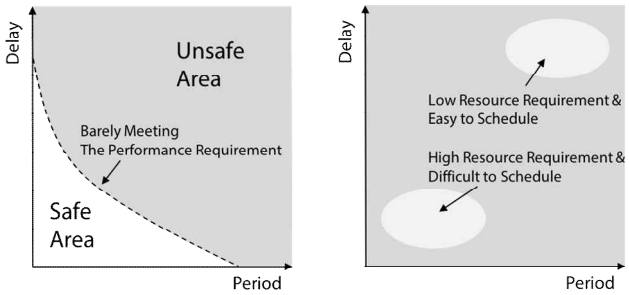
3. Tradeoff Between Period and Delay

이와 같이 제어 주기와 지연시간은 각각 제어 시스템의 성능에 직접적인 영향을 미친다. 이 둘을 종합적으로 고려하면, 즉 제어 주기와 지연시간을 동시에 조절할 수 있다면 제어 성능을 이 둘의 함수로 정의할 수 있다. 거꾸로 특정 제어 시점을 만족시키기 위한 제어 주기와 지연시간의 관계를 파악할 수도 있는데 그림 8(a)는 특정 제어 성능을 만족시키는 제어 주기와 지연시간을 이차원 평면상의 곡선으로 보여준다. 결과적으로 전체 문제공간에서 우상단의 회색 영역은 원하는 제어 성능을 만족시키지 못하는 위험 지역 (Unsafe Area)이고 좌하단의 흰색 영역은 제어 성능을 만족시키는 안전 영역 (Safe Area)에 해당한다.



(a) Control Performance Impact of Control Period (b) Control Performance Impact of Delay

Fig. 7. Control Performance Impact of Period and Delay



(a) Tradeoff in terms of Control Performance (b) Tradeoff in terms of Resource Requirement and Schedulability

Fig. 8. Tradeoff Relation Between Control Period and Sensing-Actuation Delay

그림 8(b)는 같은 문제공간에 대해서 하드웨어 자원 요구량과 스케쥴링의 난이도를 개략적으로 표현하고 있다. 문제공간의 우상단의 경우 주기가 길고 견딜 수 있는 지연시간도 여유가 있기 때문에 단위 시간당 필요로 하는 계산량, 즉 하드웨어 자원 요구량이 작고 그에 따라 쉽게 스케쥴링하여 지연시간을 원하는 수준으로 줄일 수 있다. 반면에 좌하단의 경우는 주기가 짧고 지연시간도 짧게 유지해야 하기 때문에 단위 시간당 자원 요구량이 크고 스케쥴링이 어려워 한정된 자원으로 실현하지 못할 가능성이 높다.

결과적으로 한정된 하드웨어 자원을 가지고 제어 주기와 지연시간을 모두 원하는 만큼 줄이는 것은 현실적으로 불가능할 수 있다. 이 경우 그림 8(a)의 곡선은 원하는 제어 성능을 만족시키면서 자원 요구량이 주변에 비해 최소화되는 점들의 집합이 된다. 즉, 곡선상의 한 점에서 우상단으로 이동하면 위험 지역이 되고 좌하단으로 이동하면 자원 요구량이 커지게 된다. 즉, 원하는 제어 성능을 만족시키면서 최소 자원 요구량을 가지는 최적 (주기, 지연시간) 쌍은 곡선상의 한 점이 된다.

멀티태스킹 환경에서는 그림 8(a)와 같은 그래프가 태스크의 수만큼 존재하게 된다. 이때는 한정된 자원으로 모든 태스크를 스케쥴링하여 제어 성능과 자원 요구량 두 개의 관점에서 모두 수용할 수 있는 경우는 수는 각 그래프의 곡선 상에 존재하게 된다.

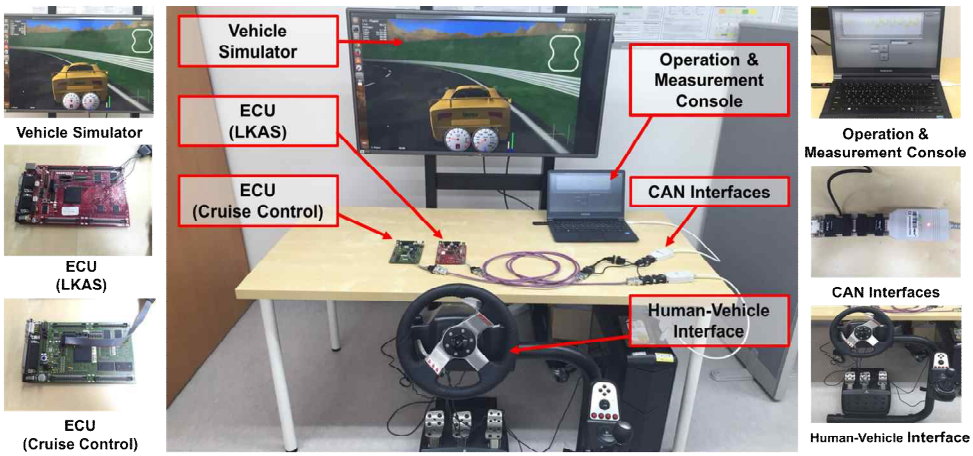


Fig. 9. Real-Time Simulation Environment

IV. Real-Time Simulation Environment

제어 주기와 지연시간이 제어 성능에 미치는 영향을 정량적으로 측정하여 분석하기 위한 실시간 시뮬레이션 환경을 구현하였다. 그림 9는 실험을 위해 구성한 전체 시뮬레이션 환경을 보여준다. 시뮬레이션 환경은 차량의 거동을 실시간으로 시뮬레이션하는 Vehicle Simulator, LKAS와 Cruise Control 알고리즘이 각각 탑재된 두 개의 ECU, 실험 파라메터 조작과 측정을 담당하는 Operation & Measurement Console, 실시간 네트워킹을 담당하는 CAN Bus Interfaces, 마지막으로 시뮬레이션 차량의 거동을 제어하기 위한 Human-Vehicle Interface로 구성된다. 각각의 구성 요소들에 대한 설명은 아래와 같다.

• **Vehicle Simulator:** 차량의 거동을 실시간으로 시뮬레이션하기 위해 오픈소스 차량 시뮬레이터인 TORCS [16]을 기반으로 차량 시뮬레이터를 구현하였다. TORCS는 자체적인 동역학 엔진을 보유하고 있고 3차원 그래픽으로 차량의 거동을 가시화할 수 있으며 소스가 모두 공개되어 있다. 하지만 외부 ECU와 연동하는 기능은 없기 때문에 TORCS의 센서 정보(e.g., 차량 속도, 조향각, Yaw, 양쪽 차선까지의 거리)를 CAN Interface를 통해서 외부로 뽑아내고 반대로 제어 값(e.g., 조향각, 쓰로틀, 브레이크)를 CAN Interface를 통해 외부에서 받아서 차량 동역학 엔진에 반영하도록 소스 코드를 수정하였다. 개발된 시뮬레이터는 PC 기반의 Ubuntu-14.04 환경에서 실행된다.

• **ECU (LKAS):** CAN 버스를 통해 Vehicle Simulator에 연결되어 차량의 조향을 제어한다. 이를 위해 LKAS 알고리즘이 필요로 하는 센서 정보들 (e.g., 차량 속도, 조향각, Yaw, 양쪽 차선까지의 거리)을 CAN 버스에서 취득하여 조향각을 결정하여 CAN 버스를 통해 Vehicle Simulator에 전달한다. ECU 하드웨어는 Infineon TC1797 MCU를 사용하였다. CPU 클락은 180MHz이고 4MB의 플래시와 1MB의 램을 탑재하고 있다. 조향각 결정을 위한 알고리즘은 [16]의 동역학 알고리즘을 기반으로 한다. 실험을 위해 제어 주기와 지연시간을 원하는 대로 조절할 수 있는 기능이 구현되어 있어서 Operation & Measurement Console에서 설정한 제어 주기와 지연시간을 지키면서 제어 알고리즘을 실행한다.

• **ECU (Cruise Control):** 차량을 미리 설정한 속도로 정속주행하게 하는 알고리즘을 탑재하고 있다. 차량 속도를 취득하여 쓰로틀 값을 결정한다. 본 논문에서는 차량의 속도를 100km/h로 유지하도록 설정되었으며 제어 주기는 1ms로 고정하였다. 본 논문은 LKAS만을 대상으로 성능 측정을 하기 때문에 Cruise Control ECU는 단지 정확히 차량의 속도를 유지하는 역할만을 수행하였다. ECU 하드웨어는 Infineon TC1796 MCU를 사용하였다. 150MHz로 동작하고 2MB의 플래시와 512KB의 램을 탑재하고 있다.

• **Operation & Measurement Console:** LKAS ECU의 제어 주기와 지연시간을 조절할 수 있는 컨트롤 패널과 LKAS 제어 성능을 실시간으로 로깅하는 기능을 담당한다. LabView를 이용하여 PC 상에서 동작하는 컨트롤 패널을 구성하였으며 실험자는 이 UI를 통해 LKAS 제어 주기와 지연시간을 조정할 수 있다. 그 외에도 LKAS 제어 성능을 포함한 모든 센싱, 엑추에이션 값을 실시간으로 그래프로 가시화 하고 기록할 수 있는 기능을 구현하였다. ECU 및 Vehicle Simulator와의 통신은 모두 CAN 버스를 이용하였다.

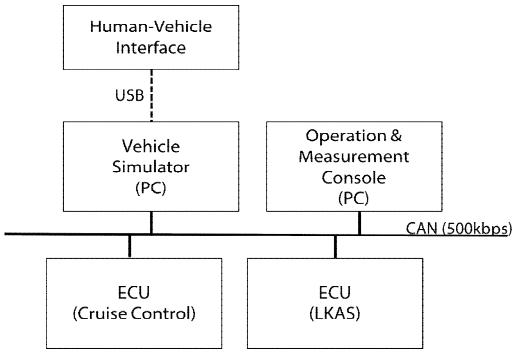


Fig. 10. Communication Architecture of Simulation Environment

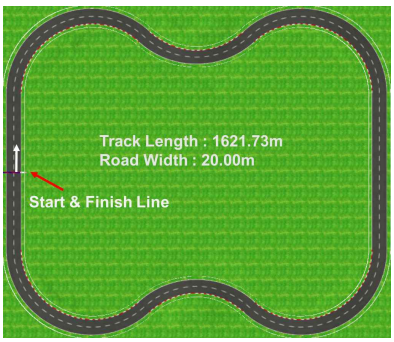


Fig. 11. Used Track for Simulation

• **CAN Bus Interfaces:** Vehicle Simulator, ECU, Operation & Measurement Console 사이의 실시간 통신을 위하여 500Kbps CAN 버스를 구성하였다. PC들은 USB 기반의 CAN 인터페이스 하드웨어를 사용하였고 ECU들은 온보드에 탑재된 CAN 컨트롤러를 사용하였다.

• **Human-Vehicle Interface:** 운전자가 직접 Vehicle Simulator 상의 차량을 조작할 경우 사용하는 드라이빙 휠과 쓰로틀, 브레이크가 Vehicle Simulator에 USB를 통해 연결되어 있다. Logitech 사의 G25 모델을 사용하였다.

시뮬레이션 환경의 전체적인 연결 구조는 그림 10과 같다. Vehicle Simulator, Operation & Measurement Console, 두 개의 ECU까지 CAN 버스로 연결되어 실시간 통신을 하고 있으며 Vehicle Simulator와 Human-Vehicle Interface는 USB로 연결되어 있다.

V. Experimental Result

IV장에서 개발한 실시간 시뮬레이션 환경을 이용하여 제어 주기와 지연시간의 변화가 제어 성능에 어떤 영향을 미치는지 LKAS 알고리즘을 대상으로 측정해 보았다. 제어 주기는 10ms에서 120ms까지 10ms 단위로 변화하고 지연시간은 각 제어 주기의 0%에서 100%까지 10% 단위로 변화하면서 LKAS 제어 성능을 측정하였다. 각 제어주기, 지연시간 조합마다 LKAS ECU와 Cruise Control ECU를 이용하여 차량의 속도를 100km/h로 유지하면서 그림 11의 트랙의 좌측 차선을 따라 10바퀴 주행한 후 최대 횡방향 에러를 측정하였다.

그림 12는 지연시간을 20ms로 고정한 상태에서 제어 주기를 20ms에서 120ms까지 변화시키면서 최대 횡방향 에러를 측정한 결과이다. 지연시간이 20ms이기 때문에 제어 주기의 최소값이 20ms이다. 실험 결과에서 보이는 것처럼 제어 주기가 증가되면 횡방향 에러가 증가하는 추세를 보인다. 지연시간이 20ms인 경우 최고 성능 0.15ms를 보이고 지연시간이 120ms인 경우 최악 성능 0.3m를 보인다. 최고 성능과 최악 성능에서 약 두 배 정도 성능 변화가 측정되었다. 결과적으로 지연시간이

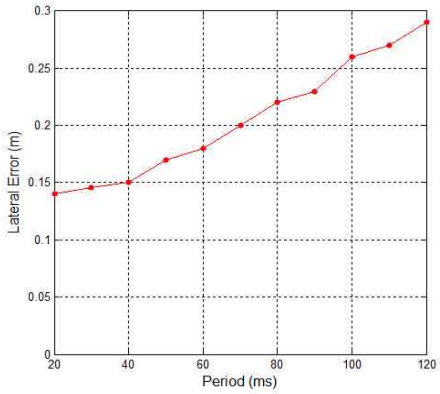


Fig. 12. Performance Impact of Control Period

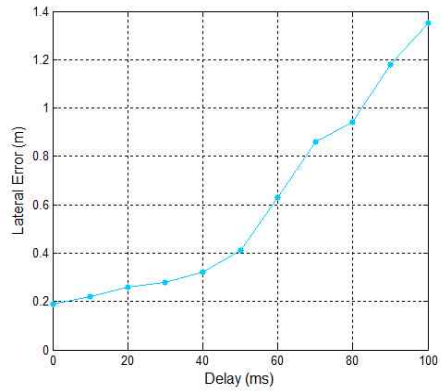


Fig. 13. Performance Impact of Sensing-Actuation Delay

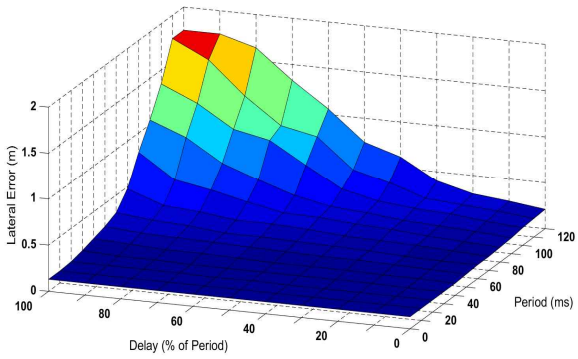


Fig. 14. Measured Control Performance of LKAS System with Varying Control Period and Delay

고정되었을 때 제어 주기가 길어지면 제어 성능이 저하된다고 판단할 수 있다.

그림 13은 반대로 제어 주기를 100ms로 고정한 상태에서 지연시간을 0ms부터 100ms까지 변화한 결과이다. 실험 편의상 지연시간이 제어 주기를 초과하는 경우는 실험 케이스에 포함시키지 않았다. 그림 13에 나타난 것처럼 제어 주기가 고정되어 있더라도 지연시간이 증가할 경우 횡방향 에러가 최소 0.2m에서 최대 1.4m로 크게 증가하는 것을 확인할 수 있다.

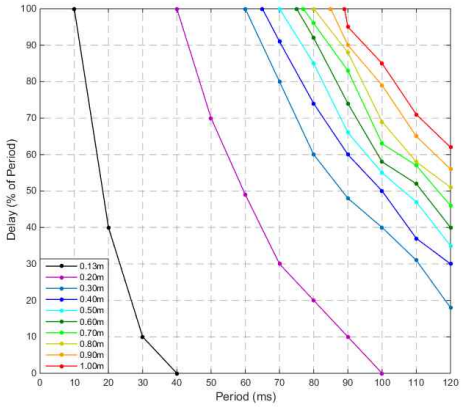


Fig. 15. Tradeoff Relation between Control Period and Delay (Proportional Delay)

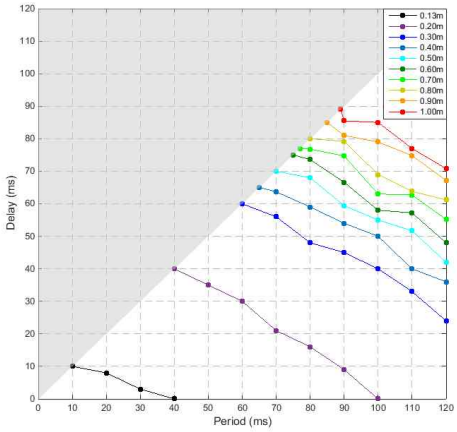


Fig. 16. Tradeoff Relation between Control Period and Delay (Absolute Delay)

그림 14는 모든 제어 주기와 지연시간에 대한 제어 성능 전체 측정 결과를 3차원으로 보여준다. 바닥 면의 두 축은 각각 제어 주기와 지연시간이고 수직 축은 횡방향 에러, 즉 제어 성능의 값을 보여준다. 여기서 지연시간은 제어 주기에 대한 상대적인 비율(%)로 나타냈다. 그래프를 보면 제어 주기와 지연시간이 동시에 제어 성능에 영향을 주는 것을 확인할 수 있다. (제어 주기, 지연시간)을 모두 최소 (10ms, 0ms)로 할 경우 횡방향 에러는 0.13m에서 모두 최대 (120ms, 120ms)로 할 경우 1.75m까지 증가한다.

그림 15는 그림 14의 결과를 바탕으로 최소 횡방향 에러인 0.13m부터 0.1m 간격으로 1m까지 열 개의 제어 성능값에 대해서 해당 제어 성능을 만족하는 제어 주기, 지연시간의 경계값을 보여준다. 그림 16은 그림 15의 결과를 Y축을 비율에서 절대값으로 변경한 결과이다. 좌상단의 회색 삼각형 영역은 지연시간이 제어 주기를 초과하는 영역으로 측정에서 제외된 영역이다. 그림 15, 16의 각 선의 좌측은 해당 제어 성능을 만족시키는 안전 영역이고 우측은 반대로 위험 영역이다.

VI. Conclusions

컴퓨터 기반의 제어 시스템에서 제어 주기와 지연시간은 제어 성능에 직접적인 영향을 미친다. 이 영향을 정량적으로 분석하기 위하여 실시간 시뮬레이션 환경을 구축하고 LKAS를 대상으로 제어 주기와 지연시간이 제어 성능에 미치는 영향을 측정 후 결과 분석을 실시하였다. 분석 결과 제어 주기와 지연시간이 길어질수록 제어 성능이 저하되는 현상이 확인되었다. 또한 특정 제어 성능을 만족하기 위한 제어 주기와 지연시간은 서로 트레이드오프 관계에 있어서 제어 주기가 증가하더라도 이를 보완하기 위해 지연시간을 줄임으로써 제어 성능을 유지할 수 있었다.

향후 연구 과제로는 멀티태스킹 환경에서의 스케쥴링 분석 기법과 결합하여 제어 성능과 스케쥴링 두 가지 관점을 동시 고려한 트레이드오프 관계를 파악할 계획이다. 또한 제어 성능을 최적화 하거나 스케쥴링 가능성을 극대화 하는 제어 주기와 지연시간을 찾는 최적화 방법을 연구할 계획이다.

REFERENCE

[1] Manfred Broy, “Challenges in Automotive Software Engineering,” Proceedings of the 28th International Conference on Software Engineering (ICSE), pp. 33-42, 2006.

[2] H. Lee, K. Kim, G. Jung, K. Choi, S. Park, D. Kwon, "Studies of the possibility of external threats of the automotive ECU through simulation test environment," Journal of The Korea Society of Computer and Information, Vol. 18, No. 11, pp. 39-49, Nov. 2013.

[3] J. Lee, K. Kang, D. Noh, "Cyclic Executive for Autonomous Driving with Real-Time Smart Cruise Control," Journal of The Korea Society of Computer and Information, Vol. 19, No. 1, pp. 1-8, Jan. 2014.

[4] Anton Cervin, Manel Velasco, Pau Marti, and Antonio Camacho, “Optimal Online Sampling Period Assignment: Theory and Experiments,” IEEE Transactions on Control Systems Technology, Vol. 19, No. 4, pp. 902-910, July 2011.

[5] Enrico Bini and Anton Cervin, “Delay-aware Period Assignment in Control Systems,” Proceedings of the 29th IEEE Real-Time Systems Symposium (RTSS), pp. 291-300, 2008.

[6] Anton Cervin, Dan Henriksson, Bo Lincoln, Johan Eker, and Karl-Erik Arzen, “How does control

timing affect performance? Analysis and Simulation of Timing Using Jitterbug and TrueTime," IEEE Control Systems, Vol. 23, No. 3, pp. 16-30, 2003.
[7] Danbing Seto, John P. Lehoczky, Lui Sha, and Kang G. Shin, "On Task Schedulability in Real-Time Control Systems," Proceedings of the 17th IEEE Real-Time Systems Symposium (RTSS), pp. 13-21, 1996.
[8] Danbing Seto, John P. Lehoczky, Lui sha, Kang G. Shin, "Trade-Off Analysis of Real-Time Control Performance and Schedulability," Real-Time Systems, Vol. 21, No. 3, pp. 199-217, 2001.
[9] Enrico Bini and Marco Di Natale, "Optimal Task Rate Selection in Fixed Priority Systems," Proceedings of the 26th IEEE Real-Time Systems Symposium (RTSS), pp. 399-409, 2005.
[10] Abhijit Davare, Qi Zhu, Marco Di Natale, Claudio Pinello, Sri Kanajan, and Alberto Sangiovanni-Vincentelli, "Period Optimization for Hard Real-time Distributed Automotive Systems," Proceedings of the 44th Design Automation Conference (DAC), pp. 278-283, 2007.
[11] Ismael Ripoll and Rafael Ballester-Ripoll, "Period Selection for Minimal Hyperperiod in Periodic Task Systems," IEEE Transactions on Computers, Vol. 62, No. 9, pp. 1813-1822, Sep. 2013.
[12] Thidapat Chantem, Xiaofeng Wang, M.D. Lemmon, and X. Sharon Hu, "Period and Deadline Selection for Schedulability in Real-Time Systems," Proceedings of the 20th Euromicro Conference on Real-Time Systems (ECRTS), pp. 168-177, 2008.
[13] Yifan Wu, Giorgio Buttazzo, Enrico Bini, and Anton Cervin, "Parameter Selection for Real-Time Controllers in Resource-Constrained Systems," IEEE Transactions on Industrial Informatics, Vol. 6, No. 4, pp. 610-620, Nov. 2010.
[14] N. Audsley, A. Burns, M. Richardson, K. Tindell, and A.J. Wellings, "Applying new scheduling theory to static priority pre-emptive scheduling," Software Engineering Journal, Vol. 8, No. 5, pp. 284-292, 1993.
[15] Giorgio Buttazzo, Manel Velasco, and Paul Marti, "Quality-of-Control Management in Overloaded Real-Time Systems," IEEE Transactions on Computers, Vol. 56, No. 2, pp. 253-266, 2007.
[16] Bernhard Wymann, "TORCS Manual installation and Robot tutorial," http://www.berniw.org/aboutme/publications/torcs.pdf

Authors

Hyun-Jun Cha received the B.S. degree in Computer Science from Kookmin University, Korea, in 2014. He joined the Graduate School of Automotive Engineering at Kookmin University, Seoul, Korea, in 2014 where he is currently working towards his M.S. degree. His main research interests are AUTOSAR, multicore ECU scheduling, and CAN network.

Seong-Woo Park is in his senior year in the Department of Computer Science at Kookmin University, Korea. He joined the Department of Computer Science at Kookmin University, Seoul, Korea, in 2009. His research interests are AUTOSAR, software validation and verification, and ISO26262 functional safety standard.

Woo-Hyuk Jeong is in his junior year in the Department of Computer Science at Kookmin University, Korea. He joined the Department of Computer Science at Kookmin University, Seoul, Korea, in 2009. His research interests are real-time operating systems, hierarchical scheduling, multicore scheduling, AUTOSAR, and connected car services.

Jong-Chan Kim received his B.S., M.S. and Ph.D. degrees in the School of Computer Science and Engineering from Seoul National University, Korea, in 1999, 2001 and 2013, respectively. He is currently an assistant professor in the Department of Automobile and IT Convergence and the Graduate School of Automotive Engineering at Kookmin University, Seoul, Korea. His main research interests are ECU consolidation, multicore ECU, real-time operating systems, and middleware for automotive systems.