논문 19-44-02-17 The Journal of Korean Institute of Communications and Information Sciences '19-02 Vol.44 No.02
https://doi.org/10.7840/kics.2019.44.2.299

모바일 크라우드소싱 기반 교통신호등 현시 및 천이시각 추정

박 재 현◆, 정 한 유°

Traffic Signal Phase and Timing Estimation based on Mobile Crowdsourcing

Jae-Hyun Park◆, Han-You Jeong°

요 약

본 논문에서는 모바일 크라우드소싱 플랫폼(MCP)을 기반 교통신호등 현시 및 천이시각(SPaT)을 추정하는 새로운 방법을 제안한다. 운전자의 크라우드내비 앱은 MCP 서버가 제공하는 공공 데이터의 신호등의 위치와 종류에 대한 정보를 기반으로 카메라 이미지에 딥러닝 인식 알고리듬과 관심영역 추정기법을 적용하여 신호등을 감지한다. 점등신호의 선형 회귀와 명도가 높은 점등신호의 상대적 위치 및 형태의 관측을 통해 신호등의 현시를 결정하고 이를 MCP 서버로 크라우드소싱한다. MCP 서버에서 신호등의 주기와 신호패턴 종류를 결정하고, 각 현시의 변경 시점과 기간을 추정하는 알고리듬을 제시한다. 부산대학교 정문 앞 교차로에서 주행실험을 수행한 결과 신호등 주기와 신호패턴 종류, 그리고 각 현시별 변경시점과 기간을 정확하게 추정할 수 있음을 보였다.

Key Words : Intelligent transportation systems, signal phase and timing (SPaT) estimation, mobile crowdsouring, public data, navigation

ABSTRACT

Intelligent transportation systems, signal phase and timing (SPaT) In this paper, we present a novel signal phase and timing estimation (SPaT) algorithm using mobile crowdsourcing platform (MCP). In the MCP, driver's CrowdNavi app detects the traffic light by running a deep-learning and a region-of-interest (ROI) algorithm based on the public data on the location and type of traffic light. It also determines the signal phase using linear regression of light color and the relative position and shape of traffic light, and then sends the phase intervals to the MCP server. We also propose a few estimation algorithms of MCP server that estimate not only the cycle period and the type of signal pattern, but also the timing information of each phase. The experimental results at the intersection in front of Pusan National University main gate show that the proposed estimation algorithms can accurately estimate the cycle period, signal pattern, and the timining of each phase. stimation, mobile crowdsouring, public data, navigation

※ 이 연구는 부산대학교 기본연구지원사업(2년)에 의하여 연구되었음.

◆ First Author : (ORCID:0000-0002-0527-9608)Department of Electrical, Electronic and Computer Engineering, Pusan National Univsersity, popwogus@gmail.com, 학생회원

° Corresponding Author : (ORCID:0000-0002-1074-8464)School of Electrical and Computer Engineering, Pusan National University, hyjeong@pusan.ac.kr, 종신회원

논문번호 : 201901-415-C-RU, Received January 14, 2019; Revised February 7, 2019; Accepted February 8, 2019

Ⅰ. 서 론

교차로에서 교통신호등은 서로 다른 방향에서 진입하는 차량들의 안전하고 원활한 소통을 가능하게 하는 장치이다. 일반적으로 운전자들은 시각을 통해 교통신호등의 현시(Phase)를 인지하며, 현시에 따라 그대로 또는 정지 후 출발(Stop and Go: SG)을 통해 교차로를 통과한다. 그러나, 현재의 교통신호등은 현시의 잔여시간 정보가 부재하여 교차로를 진입하는 차량들이 불필요하게 급제동할 수 있다. 아우디의 트라볼루션 프로젝트는 교차로 제동으로 인해 발생하는 약 15 %의 CO_2 배출을 줄일 수 있음을 보였다[1,6]. 또한, 논문 [2]는 가상의 교차로에 설치된 노변기지국이 Vehicle-to-Everything (V2X) 통신을 통해 자동차들에게 신호등의 현시와 잔여시간을 방송함으로써, 교차로 대기시간을 줄일 수 있음을 주행실험을 통해 보였다. 그러나, 국내의 교통신호제어기는 노변기지국으로 제어정보를 전달하기 위한 규격을 정의하지 않기 때문에 현시 및 잔여시간을 추정하기 위한 새로운 방안이 필요하다[7].

최근, 교통신호등 현시 및 천이시각 (Signal Phase and Timing: SPaT) 추정에 관한 다양한 연구가 진행되고 있다[3-6]. 논문 [3]은 시험차량(Probe Car)의 SG 이벤트 정보를 수집하여 교통신호등의 주기와 각 현시의 순서 및 시간을 추정할 수 있음을 보였다. 논문 [4]는 버스의 간헐적인 이동상태 정보와 간단한 이동모형을 결합하여 SPaT를 추정하는 기법을 제시하였다. 논문 [5]는 택시의 경로 데이터를 통해 교통신호등의 주기와 각 현시의 순서 및 시간을 추정하였다. 그러나, 차량의 경로 데이터 기반 SPaT 추정은 많은 데이터를 필요로 하고, 정확도가 상대적으로 낮으며, SPaT 정보 변화에 대해 기민하게 대응하지 못한다. 한편, 스마트폰 카메라의 이미지 신호처리를 통해 SPaT를 수집하고, 이를 애드 혹 네트워킹을 통해 주변 차량들과 공유하는 플랫폼인 SignalGuru를 제시하였다[6]. 이미지 신호처리를 통한 SPaT 추정은 높은 정확도를 가질 수 있으나, 관측시각, 날씨, 조도 등에 영향을 받는 문제점이 있다. 뿐만 아니라, 애드 혹 네트워킹에서 SPaT 정보는 유지/관리가 어렵고, 지리적으로 제한된 영역에서만 사용가능하다.

본 논문은 내비게이션 서비스 기반 모바일 크라우드소싱 플랫폼(Mobile Crowdsourcing Platform: MCP)을 활용한 새로운 SPaT 추정 방법을 제안한다. MCP는 안드로이드 플랫폼 기반 크라우드내비(CrowdNavi) 앱, 그리고 이와 연동하는 MCP 서버들로 구성된다. MCP 서버는 교차로에 진입하는 차량의 CrowdNavi 앱에 신호등의 위치와 종류를 제공한다. 이를 기반으로 스마트폰은 카메라 이미지의 신호등 인지와 점등신호의 색상, 위치/형태를 분석하여 SPaT을 감지하며, 이를 크라우드소싱 서버에 전달한다. 수집된 다수의 신호등 상태정보를 기반으로 MCP 서버는 교통신호의 주기, 신호등 점등패턴의 종류, SPaT을 추정한다.

본 논문의 주요 기여사항은 다음과 같다. 첫째, 오픈 디지털 지도를 기반으로 모바일 크라우드소싱 기능을 제공하는 MCP에 공공 데이터 포털에서 수집한 신호등의 위치와 종류를 구조화한다. 둘째, 교차로에 진입하는 CrowdNavi 앱은 관심영역에서의 신호등 인지, Hue, Saturation, Value (HSV) 색상공간에서 선형회귀(Linear Regression)와 점등신호의 형태 및 상대적 위치 정보를 고려한 신호등의 현시를 결정하는 SPaT 감지 방안을 제시한다. 셋째, 수집된 녹색점등 시작 시간의 최대공약수를 최소제곱오차 (Least Square Error) 기반으로 추정하는 신호등 주기 추정 기법과 현시 천이패턴의 결정트리 기반 신호등 패턴을 인지 기법, 동일 교차로 내의 신호등들 간 점등 제약을 고려한 신호등 SPaT 추정 알고리듬을 제시한다. 넷째, 부산대학교 정문 앞의 교차로에서 주행실험을 통해 제안하는 MCP 기반 SPaT 추정 알고리듬의 성능을 검증한다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 모바일 크라우드소싱 플랫폼을 설명한다. Ⅲ장과 Ⅳ장에서는 SPaT 정보의 감지와 추정 기법을 각각 소개한다. Ⅴ장에서는 도로주행 실험결과를 논의하고, Ⅵ장에서는 본 논문의 결론을 제시한다.

Ⅱ. 모바일 크라우드소싱 플랫폼

그림 1은 운전자에게 내비게이션 기능을 제공하기 위해 CrowdNavi 앱과 서버들로 구성된 MCP를 나타낸다[8,9]. 운전자가 출발지와 목적지 정보를 입력하면 CrowdNavi 앱은 경로요청(Route REQ) 메시지를 MCP 내비게이션 서버에 전달한다. 오픈스트리트맵(OSM)에 OSRM[1] 내비게이션 엔진을 구축한 MCP 내비게이션 서버는 경로응답(Route RSP) 메시지를 통해 두 지점 간 최단경로를 CrowdNavi 앱에 전달한다. 그림 1의 왼편에 보인 바와 같이 OSM의 경로는 연결된 선분들로 구성된 폴리라인(Polyline)인 Way들

1) Open Shortest Routing Machine

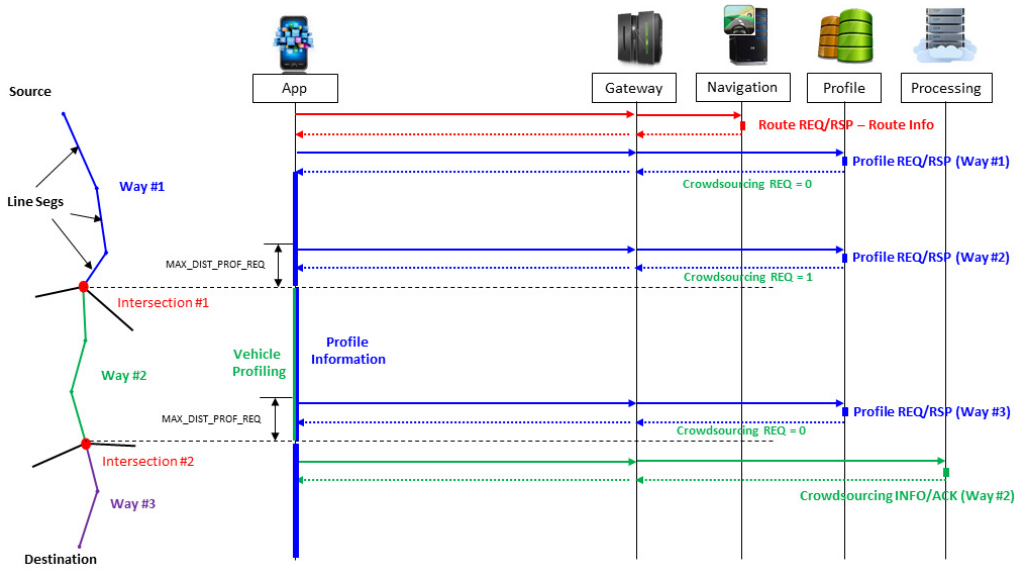

그림 1. 모바일 크라우드소싱 플랫폼에서 메시지 교환 절차 [9]
Fig. 1. Message exchanges in the mobile crowdsourcing platform [9]

과 이들을 서로 연결하는 교차로(Intersection)들의 목록으로 구성된다.

MCP 프로파일 (Profile) 서버는 주행방향, 과속방지턱 등의 정적 프로파일 정보와 Way 통과시간, 주행평균속력 등의 동적 프로파일 정보, 그리고 교통사고, 포트홀 등의 이벤트 프로파일 정보를 관리한다. CrowdNavi 앱은 각 Way에 진입하기 전의 일정한 거리(MAX_DIST_PROF_ REQ = 200 m)에서 다음 Way에 대한 프로파일 요청(Profile REQ) 메시지를 MCP 프로파일 서버로 전달한다. MCP 프로파일 서버는 프로파일 응답(Profile RSP) 메시지를 통해 정적/동적/이벤트 프로파일 정보를 CrowdNavi 앱에 전달한다.

이 때, 추가적인 크라우드소싱이 필요하면 (Way #2), Profile RSP 메시지의 크라우드소싱 요청 필드를 1로 설정한다. CrowdNavi 앱은 스마트폰 센서들을 초기화하여 Way #2에서 교통정보를 수집한다. 차량이 Way #2를 통과하여 Way #3에 진입하면 수집한 센서 측정값들을 크라우드소싱 정보 (Crowdsourcing INFO) 메시지에 담아 MCP 처리 (Processing) 서버로 전달한다. MCP 처리 서버는 크라우드소싱 응답 (Crowdsourcing ACK) 메시지를 보내 메시지 수신 여부를 통지한다. MCP 플랫폼은 도로 인프라 구축 없이 CrowdNavi 앱을 활용하여 도로와 교통정보를 효과적으로 수집할 수 있다.

논문 [8,9]의 MCP를 기반으로 스마트폰 카메라의 영상 인식을 통해 SPaT 정보의 정확하게 추정하는 것이 본 논문의 목표이다. 이를 위해, 신호등 공공데이터를 MCP 프로파일 서버의 데이터베이스에 저장하기 위한 방안과 CrowdNavi 앱과 MCP 처리 서버 간 현시구간정보의 구조를 소개한다.

2.1 공공 데이터 기반 교통신호등 정보의 수집

정부에서는 공공기관이 생성 또는 취득하여 관리하는 공공 데이터를 국민이 쉽고 편리하게 이용할 수 있도록 공공 데이터의 제공 및 이용 활성화에 관한 법률을 통해 공공 데이터 포털을 운영하고 있다. 본 절에서는 공공 데이터 포털을 통해 부산광역시 내 19,955개의 차량용 신호등에 대한 공공 데이터를 수집하였다[10]. 차량용 신호등 정보는 신호등의 (위도, 경도) 좌표와 신호등의 종류를 포함한다. 차량용 신호등은 횡형 3/4/5색, 종형 3/4/5색, 가변형 가변등의 7종류가 있다.

일반적으로, 교차로의 중심을 기준으로 신호등의 위치는 진입하는 Way의 반대편에 위치하기 때문에, 신호등, 교차로 중심, 진입 Way의 (위도, 경도) 좌표들을 비교하여 교차로의 정적 프로파일에 반대편 신호등의 (위도, 경도) 좌표와 종류를 추가하였다.

2.2 신호등 정보 수집을 위한 메시지 속성 정의

그림 1에서 CrowdNavi 앱이 수신하는 프로파일 응답 메시지의 교차로 정적 프로파일에 Way를 주행

표 1. 횡형사색등의 현시, 이미지, 인덱스
Table 1. Phase, view, and index of horizontal traffic light

Phase	Image	Index
Detection Failure	-	0
Red		1
Yellow		2
Left Turn		3
Green		4
Red - Yellow		5
Red - Left Turn		6
Yellow - Left Turn		7
Yellow - Green		8
Green - Left Turn		9

한 후 진입하는 교차로 교통신호등의 (위도, 경도) 좌표와 신호등의 종류를 전달한다.

표 1은 횡형사색등이 표시할 수 있는 현시들의 이미지를 표시하였다[11]. 방향 d에서 교차로로 진입하는 차량의 위치와 교차로 중심과의 거리가 80 m 이내이면 CrowdNavi 앱은 교통신호등 감지 알고리듬을 실행한다. 감지된 신호등이 이들 중 하나와 최초로 매칭되면, 표 1의 현시 인덱스 p와 최초 감지시각 t_s, 그리고 최초 감지요인을 저장한다. 표 2에서 최초 감지요인은 교통신호등의 최초감지 또는 현시의 천이로 구분된다. 차량이 교차로를 통과하거나 현시가 변경되면, 최종 감지시각t_e와 최종 감지요인을 저장한다. 수

표 2. 감지 코드 c와 최초/최종 현시의 감지요인
Table 2. Detection code c by start/end of phase detect

Detection Code	Start of Phase Detect	End of Phase Detect
00	First Phase Detect	Passing through Intersection
01	First Phase Detect	Phase Change
10	Phase Change	Passing through Intersection
11	Phase Change	Phase Change

집된 정보를 기반으로 CrowdNavi 앱은 최초/최종 감지요인들의 조합인 감지코드 c와 현시 인덱스, 최초/최종 현시시각을 포함하는 현시구간정보 $M_{p,d}=(d,p,t_s,t_e,c)$ 를 생성한다. 마지막으로, CrowdNavi 앱은 크라우드소싱 정보 메시지를 통해 현시구간정보를 MCP 처리 서버에 전달한다.

현시구간정보를 수신한 MCP 처리 서버는 내부에서 관리하는 SPaT 정보와 비교하여 임계값을 초과하는 오차가 발생하면 해당 신호등의 크라우드소싱 요청 필드를 1로 설정하여 크라우드소싱 정보를 추가로 수집한다. 일정한 개수의 크라우드소싱 정보가 수집되면 MCP 처리 서버는 SPaT 정보를 추정한다.

Ⅲ. 교통신호등 현시 및 천이시각 감지

운전자의 CrowdNavi 앱은 카메라 이미지 처리를 통해 교통신호등과 현시를 인지하는 기능이 필요하다. 이를 위해, 스마트폰 GPS 수신기의 측위 좌표와 공공데이터 기반 신호등 위치 좌표를 고려하여 카메라 이미지 상의 신호등 위치에 대한 관심영역을 설정한다. 관심영역에서 딥러닝 기반 신호등 인식과 이전 프레임의 신호등 위치를 고려하여 현 프레임의 신호등 영역을 추정한다. 점등 신호의 HSV 색상에 대한 선형회귀(Linear Regression)와 형태, 신호등 내 점등신호의 상대적 위치 분석을 통해 현시를 감지한다. 각 현시의 시간구간이 결정되면 현시구간정보 $M_{p,d}$를 생성하여 MCP 서버로 이를 전달한다.

3.1 신호등 영역 추정

자동차가 교차로에 접근하면, 프로파일 응답 메시지에 포함된 신호등의 (위도, 경도) 좌표와 스마트폰 GPS 수신기의 측위 좌표를 활용하여 신호등의 관심영역을 설정한다. 측정된 스마트폰의 거치 높이(h)와 신호등의 최소 높이(4.5 m)가 주어지면, 자동차와 신호등의 (위도, 경도) 좌표를 통하여 상대거리(D)를 구하고, 방향 센서(Orientation Sensor)를 통해 스마트폰의 기울어진 각도(θ)를 측정한다. 그림 2에서 f는 화

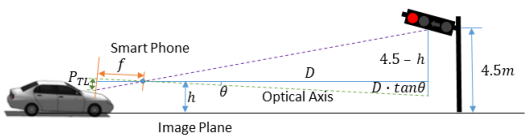

그림 2. 신호등 거리 기반 관심영역 설정
Fig. 2. Region of interest from the distance to traffic light

소로 환산한 초점 거리(Focal Length)이며, P_{TL}은 계산을 통해 얻는 카메라 이미지 중심으로부터의 신호등 위치의 높이 픽셀 수이다. 바닥이 평평한 도로를 가정하면, Pinhole 카메라 모형 기반의 비례식을 통해 P_{TL}을 추정할 수 있다.

$$D : 4.5 - h + D\tan\theta = f : P_{TL} \tag{1}$$

추정된 신호등의 위치를 중심으로 카메라 이미지 폭과 높이의 1/3의 마진을 두고 관심영역을 설정한다.

관심영역 내에서 신호등의 위치 추정을 위해 합성곱 신경망(Convolution Neural Network)의 계산 복잡도를 크게 낮춘 딥러닝 기반 객체 인식 솔루션인 MobileNet을 활용한다[12]. 신호등 인식률을 향상하기 위해 이전 프레임에서 신호등이 인식되면 그 위치에서 각 방향으로 100개의 화소를 마진으로 추가한 영역을 MobileNet의 입력 데이터로 설정한다. MobileNet은 입력 이미지 내에서 인식된 객체의 영역을 사각형으로 표시한다.

인접한 두 프레임에서 신호등의 이미지 내 위치는 차량의 이동 방향과 속력 등에 따라 달라진다. 60 Km/h 속력의 차량에 장착된 10 FPS의 비디오를 지원하는 CrowdNavi 앱에서 한 프레임 시간 동안 차량은 약 1.7 m를 이동한다. 만약 딥러닝 기반 신호등 검출에 실패하면, 차량의 이동으로 인한 이미지 내 신호등 위치 변경을 반영하기 위해 이전 프레임의 신호등 위치를 중심으로 신호등 영역을 1.2 배 확장하여 현재 프레임의 신호등의 영역으로 사용한다.

3.2 신호등 현시 감지

그림 3은 크라우드소싱 내비게이션 앱에서 추정된 신호등 영역을 청색 사각형으로 표시하였다. 신호등 영역은 신호등의 일부분만을 포함하거나, 신호등 영역 밖의 배경까지 포함하는 경우가 발생한다. 이를 해결하기 위해 신호등의 검정색 배경을 감지하여 신호등의 정확한 위치를 추정한다. 우선, 신호등 영역의 폭과 높이의 비율을 통하여 감지된 영역이 신호등 일부 또는 전체인지 판단한다. 만약 신호등의 일부분만 감지되었다면, 점등 신호의 색상을 기반으로 좌우로 신호등 초기 영역을 확장한다.

RGB 색상 공간은 각 원색 별 빛의 세기를 표시하는 반면 HSV 색상 공간은 색조(Hue), 채도(Saturation), 명도(Value)를 구분하기 때문에 순수 원색을 판별할 수 있는 신호등 현시 결정에 보다 적합하다[13]. 색조는 가시광선 스펙트럼을 고리모양으로 배

그림 3. 크라우드소싱 내비게이션 앱에서 신호등 영역 추정
Fig. 3. Traffic signal area estimation in crowdsourcing navigation app

치한 색상환에서 가장 파장이 긴 적색을 0°로 하였을 때, 상대적인 배치 각도[0°, 360°]를 2로 나눈 값이다. 채도는 색상의 가장 선명한 상태를 100 %로 하였을 때 상대적인 선명함의 정도를 나타내며, 명도는 검은색을 0 %, 흰색을 100 %로 하였을 때 상대적인 밝기를 나타낸다. 색조는 [0, 180], 채도와 명도는 [0, 255]의 구간에서 정수로 나타낸다.

점등 신호의 색은 조도, 태양 및 주변 광원의 간섭 등에 의해 영향을 받는다. 점등 신호의 명도는 주변 환경보다 높기 때문에 x_V가 임계값 V_L보다 큰 화소들에 대하여 HSV 색상값의 평균 $\boldsymbol{x} = (x_H, x_S, x_V)^\top$을 입력벡터로 하는 선형 회귀를 통해 점등 신호의 색상을 구별한다. 이 때, 적색의 색조(x_H) 영역은 0과 180의 경계를 포함하기 때문에 x_H가 160을 초과하면 선형회귀를 위해 $x_H - 180$을 입력으로 대입한다. 점등신호의 색상 C의 출력변수 $y_{C_1C_2}$를 아래와 같이 정의하면

$$y_{C_1C_2} = \begin{cases} 0, & \text{if } C = C_1 \\ 1, & \text{if } C = C_2 \end{cases} \tag{2}$$

출력벡터 $\boldsymbol{y} = (y_{RY}, y_{RG}, y_{YG})^\top$ 는 선형회귀를 통해 다음과 같이 나타낸다.

$$\begin{gathered} \boldsymbol{y} = \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b} \\ \boldsymbol{W} = \begin{bmatrix} 0.0349 & 0.0123 & 0.0156 \\ 0.00229 & 0.000852 & -0.000339 \\ -0.00211 & -0.000412 & -0.00107 \end{bmatrix}, \\ \boldsymbol{b} = \begin{bmatrix} 0.05 \\ -0.079 \\ -0.027 \end{bmatrix} \end{gathered} \tag{3}$$

$\boldsymbol{W}$와 $\boldsymbol{b}$의 값은 예비 실험을 통해 제곱오차를 최소화하는 값으로 설정하였다. 점등 신호의 색상 C는 출력벡터 $\boldsymbol{y}$의 원소들 중에서 가장 많이 선택되는 색상으로 결정한다.

표 1에서 현시는 두 개의 신호등이 동시에 점등될 수 있기 때문에, 신호등 내의 상대적 위치와 점등 신호의 형태를 분석하여 현시를 결정해야 한다. MCP 서버에서 수신한 신호등의 종류에 따라 신호등 내부의 각 신호 영역을 구분하고, 해당 영역 내에서 점등 신호가 존재하는지를 판단한다. 점등 신호의 형태는 명도의 임계값 V_L을 초과하는 화소들을 라인 스캔하여 기하학적 특징이 원 또는 화살표 모양인지 판단한다.

Ⅳ. 교통신호등 현시 및 천이시각 추정

본 장에서는 크라우드소싱을 통해 수집한 사거리 교차로 신호등의 현시구간정보들을 활용하여 신호등 주기 P, 신호등 패턴 종류 T_d, 각 현시 별 시간 $D_{p,d}$를 추정하는 알고리듬을 제시한다.

4.1 신호등 주기 추정

본 절에서 제안하는 신호등 주기 추정 알고리듬은 신호등 주기의 세가지 특성들을 활용한다. 첫째, 일반적으로 신호등의 주기는 1초 단위의 정수로 설정된다. 둘째, 교차로 신호등의 주기 P는 모든 진입방향에서 동일하며, 각 진입방향 d에서 측정된 각 현시시간 최대값의 합인 $\sum_{\forall p}\max(t_e - t_s)$ 보다 항상 크거나 같다. 셋째, 특정 진입방향 d의 두 시점에서 측정된 특정 현시 p의 변경시각들의 차이($t_s' - t_s$ 또는 $t_e' - t_e$)는 신호등 주기의 정수배로 표현된다.

그림 4는 현시구간정보의 집합 $\{M_{p,d}\}$를 입력으로 하여 신호등 주기 P를 추정하는 **estimatePeriodOfTrafficLights()** 함수를 나타낸다. 첫 for 루프에서는 각 방향별 현시시간 최대값들의 합 $\sum_{\forall p}\max(t_e - t_s)$의 최대값인 maxPeriod와 각 방향별로 가장 빈번하게 측정되는 적색점등 종료시점들의 시간($t_e' - t_e$)을 나타내는 배열 diffPeriod[]를 생성한다 (**Lines 2-13**). 두 번째 for 루프에서는 maxPeriod부터 경찰청 교통신호등 설치관리 매뉴얼 [11]의 주기 최대값인 MAX_PEROID (180 초)까지 1초 간격으로 주기를 변경하면서 제곱오차의 합을 최소화하는 신호등 주기 P의 정수값을 결정한다 (**Lines 14-21**).

Algorithm 1. estimatePeriodOfTrafficLights()

INPUT: List of phase interval messages $\{M_{p,d}\}$
OUTPUT: Estimated period of traffic light P
PROCEDURE:

```
1  maxPeriod = 0, minSquaredSumError = INFINITY;
2   for each direction d
3     directionalPhaseSum = 0, lastRedEndTime = NIL;
4     for each phase p
5       directionalPhaseSum += phaseMaxDuration(p, d);
6       if (p is RED & e = X1)
                  // If end time of RED is detected
7         if (lastRedEndTime != NIL)
8             diffPeroid[i++] = round(t_e - lastRedEndTime);
9         lastRedEndTime = t_e;
                  // Set new last RED end time
10    end for
11    if (maxPeriod<directionalPhaseSum) // If new max period
12       maxPeriod = directionalPhaseSum;      // is obtained
13  end for
14  for each integer j in interval (maxPeriod, MAX_PERIOD]
15    curSquaredSumError = 0; // Initialize the current SSE
16    for each i       // Add each squared error of period j
17          curSquaredSumError += pow(diffPeriod[i] % j, 2);
18    end for
19    if (curSquaredSumError < minSquaredSumError)
            // If new squared sum error is found
20        minSquaredSumError = curSquaredSumError;
21        P = j;
22  end for
23  return P;
```

그림 4. 신호등 주기 추정 알고리듬
Fig. 4. Algorithm for estimating the period of traffic lights

4.2 결정트리 기반 신호등 패턴 종류 추정

국내 횡형사색등은 8가지의 신호등 패턴 종류를 가질 수 있다[11]. 그림 5는 횡형사색등에서 적색점등 현시를 루트 노드로 정했을 때, 신호등 현시 패턴들을 결정트리로 나타내었다. 결정트리에서 각 노드는 현시를 나타내며 에지로 연결된 부모 노드와 자식 노드는 해당 현시들의 인접한 순서를 나타낸다. 예를 들어, 적색점등 현시(루트 노드) 이후에는 녹색점등 현시, 녹색과 좌회전 동시점등 현시, 그리고 적색과 좌회전 동시점등 현시가 가능하다. 리프 노드에 표시한 숫자는 신호패턴의 식별자를 나타낸다. 또한, 리프 노드의 깊이(Depth)는 신호패턴의 현시 개수를 나타낸다. 예를 들어, 패턴 2와 4는 각각 3개와 6개의 현시들을 가진다. 신호패턴 4는 한 주기 내에서 적색신호등이 두 번 점등되어 두 개의 리프(Leaf) 노드로 표시하였다.

한편, 경찰청 교통신호등 설치관리 매뉴얼에 따르면 교차로에서 마주보는 양방향 신호등은 동일한 패

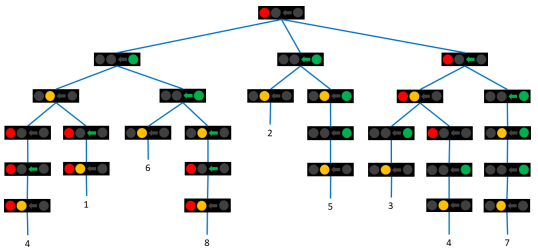

그림 5. 횡형사색등 패턴 종류에 대한 결정 트리
Fig. 5. Decision tree for 4-color horizontal traffic light pattern

턴 또는 일부 다른 패턴만을 허용한다[11]. 표 3은 매뉴얼의 내용을 분석하여 그림 5의 각 신호등 패턴 식별자에 대하여 함께 사용이 가능한 반대방향 신호등 패턴 식별자들의 목록을 나타낸다.

그림 6은 현시구간정보의 집합 $\{M_{p,d}\}$와 신호등의 주기 P를 입력으로 받아서 각 방향의 신호등 패턴을 추정하는 **estimateSignalPattern()** 함수를 나타낸다. 각 방향에 대하여 모든 현시들을 한 주기 내에 순서대로 정렬하고, 적색점등 현시(루트 노드)를 기준으로 다음 현시에 따라 결정 트리를 이동하면서 신호등의 패턴 종류를 추정한다 (**Lines 1-3**). 만약 수집된 모든 현시들을 따라 이동했을 때 현재 노드의 서브트리에 두 개 이상의 리프 노드가 존재하면, 표 4에서 나타낸 반대편 신호등 패턴 종류를 통하여 해당 신호등의 패턴 종류를 추정한다 (**Lines 4-5**). 현시구간정보가 부족하여 유일한 신호등 패턴이 도출되지 않으면, 추가로 현시구간정보를 크라우드소싱하여 신호등 패턴을 결정한다 (**Lines 6-7**).

Algorithm 2. estimateSignalPattern()

INPUT: List of phase interval messages $\{M_{p,d}\}$, and period P
OUTPUT: Pattern type T_d
PROCEDURE:
1 **for** each direction d
2 Collapse all phase intervals into a single period
3 Match the sequence of phases along decision tree
4 **if** (Number of leaf nodes in the subtree > 1)
5 Estimate T_d using the opposite-direction pattern;
6 **if** (There is still an ambiguity in signal pattern)
7 Request a new crowdsourcing $M_{p,d}$;
8 **end for**
9 **return** T_d;

그림 6. 신호등 패턴 추정 알고리듬
Fig. 6. Algorithm for estimating the pattern of traffic lights

표 3. 패턴과 함께 사용 가능한 반대방향 신호등 패턴들
Table 3. Feasible opposite-direction patterns

Pattern Type	Feasible Opposite-Direction Patterns			
1	1			
2	2			
3	3			
5	6			
7	4	6	8	
8	4	7		

4.3 각 현시구간 추정

본 절에서는 신호등 패턴의 특징을 활용하여 크라우드소싱을 통해 각 방향별 각 현시의 시간을 추정하는 알고리듬들을 제시한다.

이를 위해, 각 방향별 각 현시에 대한 다수의 시간정보들을 통합하는 것이 필요하다. 그림 7은 각 방향 각 현시별로 수집된 다수의 현시구간들을 결합하여 통합 현시구간을 생성하는 **unionPhase Intervals()** 함수를 나타낸다. 우선, 크라우스소싱된 모든 현시구간들을 보행자 횡단 신호 후 첫 적색점등 종료시점을 기준으로 한 주기 내로 통합한다 (**Line 1**). 각 방향의 각 현시에 대하여 가장 긴 점등시간을 가지는 통합현시구간 $I^{+}_{p,d}=[t^{+}_{s},t^{+}_{e}]$과 감지코드 c^{+}를 추출한다 (**Lines 2-4**). 통합현시구간을 기준으로 나머지 현시구

Algorithm 3. unionPhaseIntervals()

INPUT: List of phase interval messages $\{M_{p,d}\}$, and period P
OUTPUT: Unioned phase intervals $\{I^{+}_{p,d}\}$ and detection codes $\{c^{+}\}$
PROCEDURE:
1 Collapse all phase intervals into a single period by setting the start of the period to the earliest end time of all-direction RED phase;
2 **for** each direction d
3 **for** each phase p
4 Get the longest $I^{+}_{p,d}=[t^{+}_{s},t^{+}_{e}]$ and its code c^{+};
5 **for** each phase interval $I_{p,d}$
6 Depending on c, allign $I_{p,d}$ to $I^{+}_{p,d}$;
7 $I^{+}_{p,d}=I^{+}_{p,d}\bigcup I_{p,d}$;// Union the phase intervals
8 $c^{+}=c^{+}|\,c$; // Combine using bit-wise OR
9 **end for**
10 **end for**
11 **end for**
12 **return** $\{I^{+}_{p,d}\}$ and $\{c^{+}\}$

그림 7. 현시구간 통합 알고리듬
Fig. 7. Algorithm for merging phase intervals

간들은 감지코드가 동일 현시변경인 시각에 맞추어 정렬하고, 이들을 모두 결합한다 (**Lines 5-9**). 마지막으로 통합현시구간 $\{I^{+}_{p,d}\}$와 감지코드 $\{c^{+}\}$를 리턴한다 (**Line 12**).

통합현시구간과 감지코드를 입력으로 받아 최종 현시구간을 리턴하는 **growPhaseIntervals()** 함수를 그림 8에 나타내었다. 경찰청의 교통신호등 설치관리 매뉴얼에 의하면 황색점등시간은 구간 [MIN_YLW = 3, MAX_YLW = 5] (초) 내에서 결정된다[11]. 본 논문에서는 황색점등시간은 교차로 진입 방향과 상관없이 일정하다고 가정한다. 만약, 현시 변경을 통해 황색점등의 시작/종료 시점이 측정 가능하면, 이들 간 시간을 교차로의 황색점등시간으로 설정한다. 그렇지 않으면, 황색점등구간을 유효한 범위 내에서 임의로 설정한다 (**Lines 1-4**). 통합현시구간을 각 방향별 현시변경감지 회수의 합을 기준으로 내림차순으로 정렬하고, 동일 방향 내에서는 황색, 적색, 녹색 및 좌회전 점등 순으로 나열한다 (**Line 5**). 만약 통합현시구간의 감지코드가 00이면 현시구간 결정을 위해 추가로 크라우드소싱이 필요하다 (**Lines 8-9**). 반면, 감지코드가 11이면, 현시구간의 시작 또는 종료 시점을 감지하였기 때문에 현시구간 결정이 불필요하다. 그 이외의 경우에는 다른 방향의 교통흐름과 충돌(Conflict)이 발생하지 않도록 현시구간을 최대로 확장한다 (**Lines 10-14**). 마지막으로 최종현시구간 $\{I^{*}_{p,d}\}$를 리턴한다 (**Line 17**).

Algorithm 4. growPhaseIntervals()

INPUT: Unioned phase intervals $\{I^{+}_{p,d}\}$ and detection codes $\{c^{+}\}$
OUTPUT: Filled phase intervals $\{I^{*}_{p,d}\}$
PROCEDURE:
1 **if** (YELLOW phase interval $I^{+}_{Y,d}$ with c^{+} equal to 11)
2 $D_{Y,-} = I^{+}_{Y,d}$;
3 **else**
4 $D_{Y,-}$ = getRandom($\max$(MIN_YLW, $\max_{\forall d}(I^{+}_{Y,d})$), MAX_YLW];
5 Sort direction in the descending order of detected phase changes, and inside each direction, sort phases to RED → YELLOW → GREEN;
6 **for** each direction d
7 **for** each phase p
8 **if** (c^{+} is 00) // If both ends are not detected
9 Request a new crowdsourcing $M_{p,d}$
10 **if** (c^{+} is 01 or 10) // Only one end is detected
11 **if** (p is YELLOW) $I^{*}_{Y,d} = D_{Y,-}$;
12 **if** (p is RED) Increase $I^{*}_{R,d}$ until conflict-free;
13 **if** (p is GREEN and/or LEFT_TURN)
14 Increase $I^{*}_{G,d}$ and/or $I^{*}_{LT,d}$ until it minus $D_{Y,-}$ is conflict-free;
15 **end for**
16 **end for**
17 **return** $\{I^{*}_{p,d}\}$

그림 8. 현시구간 성장 알고리듬
Fig. 8. Algorithm for growing phase intervals

V. 도로주행 실험결과

그림 9는 공공데이터를 활용하여 부산대학교 정문 앞 사거리 교차로의 7 개의 신호등의 위치를 표시하였다. 서쪽 도로는 일방통행이어서 차량이 진입할 수 없기 때문에 동, 남, 북쪽 도로에서 진입하는 차량이 바라보는 반대편(서, 북, 남쪽)에 위치한 신호등의 SPaT 정보를 추정하는 것이 본 실험의 목표이다.[2] 주행 실험은 CrowdNavi 앱이 동작하는 Galaxy Note8 스마트폰을 장착한 차량이 각 방향에서 5 번 교차로를 통과하며 현시구간들을 크라우드소싱하였다. 그림 3의 CrowdNavi 앱은 약 10 FPS로 동작하며, 신호등 현시를 89.92 %의 정확도로 감지하였다. CrowdNavi 앱은 GPS 수신기를 통해 동기화되어 있으므로 본 논문에서 제시하는 시각 정보의 오차는 최대 0.1초 범위 내에서 존재한다.

우선, 그림 4에서 나타낸 알고리듬을 활용하여 신호등 주기를 추정한다. 남쪽 신호등에서 maxPeriod가 149.93 초로 계산되었고, 적색점등 종료시점들 간의 차이 diffPeriod[]는 각각 1500.20, 900.77, 599.90, 1950.13, 600.00, 600.08 초로 측정되었다. 그림 10은 **Algorithm 1**에 의해 계산된 신호등 주기 P에 따른 제곱근 오차 (Root-Mean- Square Error: RMSE)를 나타낸다. 그림에서 주기 $P = 150$ 초에서 최적 RMSE를 가지는 것을 확인할 수 있다. 크라우드소싱 주행 실험을 진행하는 동안에 측정한 24 주기 시간이 3600.28초로 측정되었고, 이를 통해 평균신호주기는 $\overline{P} = 150.01$ 초로 계산되었다. 이를 통해 제안하는 신호 주기 추정 알고리듬이 신호등 주기를 매우 정확하게 추정할 수 있음을 보였다.

그림 11은 각 방향에서 신호등을 5 번 통과했을 때 수집한 현시구간들을 북쪽 신호등 녹색점등현시 시작점을 기준으로 한 주기 내에 표시하였다. 그림에서 $C_{d,n}$은 방향 d의 신호등을 n 번째 통과할 때 크라우

2) 동쪽 신호등 한 개와 서쪽 신호등 2개가 동일한 교통신호등 정보를 동쪽에서 진입하는 차량들에게 제공한다.

드소싱하는 현시구간들을 나타내며, 경계점에서 현시

그림 9. 교차로 내 7 개 신호등
Fig. 9. 7 traffic lights in an intersection

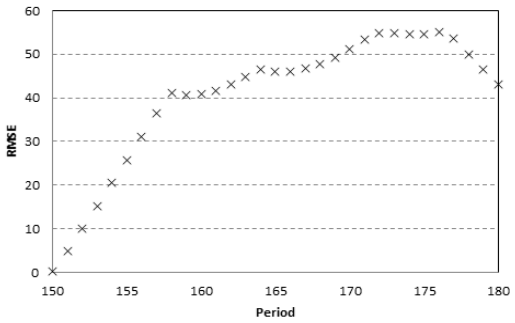

그림 10. 신호등 주기에 따른 제곱근오차
Fig. 10. RMSE vs Period of traffic lights

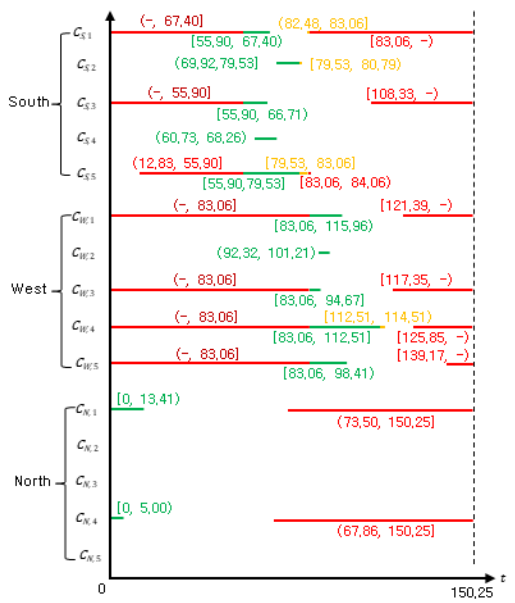

그림 11. 실험을 통해 수집한 현시구간들의 한주기 내 중첩
Fig. 11. A single period representation of all phase intervals

변화를 감지하면 대괄호([,])를 사용하고 그 이외는 소괄호((,))를 사용했다. 한 주기의 경계를 이어서 나타나는 현시구간은 양 경계에 빼기표(-)와 소괄호를 함께 사용하였다. 1회 통과 시 서쪽과 남쪽 신호등의 현시구간이 각각 평균 2.4회와 2.0회 검출되었으나, 태양이 반사되는 북쪽 신호등의 현시구간은 평균 0.8회밖에 검출되지 않아 교차로 진입방향에 따라 신호등 감지 성능이 크게 차이가 나는 것을 확인할 수 있다.

서쪽과 남쪽의 신호등은 표 1의 현시 인덱스 순서가 1→9→2로 나타나기 때문에 그림 6의 **Algorithm 2**를 통해 결정트리를 따라가면 그림 5의 횡형사색등 패턴 2임을 알 수 있다. 그러나, 북쪽 신호등의 현시 인덱스 순서 1→9를 만족하는 횡형사색등 패턴은 2와 5가 가능하다. 표 4에서 반대방향(남쪽)의 패턴이 2이기 때문에 **Algorithm 2**는 북쪽 횡형사색등의 패턴을 2로 최종 판단함으로써 모든 방향의 신호등 패턴을 정확하게 결정한다.

그림 12는 그림 7의 **Algorithm 3**을 사용하여 그림 11의 현시구간들을 결합한 통합현시구간 $\{\Gamma^{+}_{p,d}\}$를 나타낸다. 그림 8의 **Algorithm 4**에서 남쪽 신호등은 모든 구간들의 현시변경이 감지되어 각 현시들의 SPaT이 모두 결정된다. 서쪽 신호등은 황색점등종료시점이 측정되지 않았지만, 남쪽 신호등의 황색점등시간(3.53초)을 이용하면 각 현시들의 SPaT이 모두 결정된다. 북쪽 신호등은 다른 방향의 교통류의 충돌을 피하기 위해서는 적색점등구간이 55.90 초 이하로 확장되어야 한다. 뿐만 아니라, 녹색/좌회전 점등구간이 55.90 − 3.53 (황색점등시간) = 52.47초까지 확장되어 모든 신호등의 SPaT을 결정할 수 있다. 마지막으로 보행자 통행 시간은 모든 방향이 적색점등구간인 $[117.35, 150.25]$로 추정 가능하다.

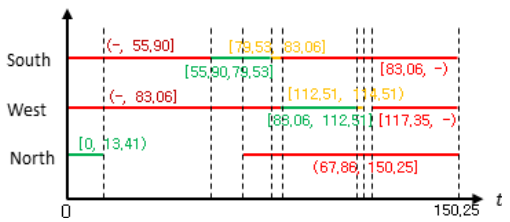

그림 12. 각 방향별 통합현시구간들
Fig. 12. Unioned phase intervals at each direction

Ⅵ. 결 론

본 논문은 교차로를 통과하는 차량들의 스마트폰

카메라 영상처리를 통해 신호등 현시구간을 MCP 서버로 크라우드소싱하고, 이를 처리하여 신호등 주기, 신호등 패턴 종류, 각 현시의 SPaT을 추정하는 알고리듬을 제시하였다. 제안하는 알고리듬은 적은 수의 크라우드소싱 정보들이 존재하더라도 결정 트리와 신호등의 양방향 제약을 고려하여 신호등의 주기, 패턴 종류, 그리고 각 현시를 정확하게 추정할 수 있음을 교차로 실험을 통해 보였다. 향후, 다양한 신호등 패턴을 가지는 교차로들에서 크라우드소싱과 SPaT 추정 알고리듬의 긴밀한 연동을 통해 성능을 최적화하기 위한 연구를 진행할 예정이다.

References

[1] R. Lamb, *How Audi's travolution device will work*, Retrieved Feb. 19, 2019, from https://electronics.howstuffworks.com/gadgets/automotive/audi-travolution.htm

[2] H.-Y. Jeong, T. A. Suramardhana, and H.-H. Nguyen, "Design and implementation of green light optimal speed advisory based on reference mobility models (GLOSA-RMM) in cyber-physical intersection systems (CPIS)," *J. KICS*, vol. 39B, no. 8, pp. 544-554, Aug. 2014.

[3] Y.-T. Chuang, C.-W. Yi, Y.-C. Tseng, C.-S. Nian, and C.-H. Ching, "Discovering phase and timing information of traffic light systems by stop-go shockwaves," *IEEE Trans. Mobile Comput.*, vol. 14, no. 1, pp. 58-71, Jan. 2015.

[4] S. A. Fayazi, A. Vahidi, G. Mahler, and A. Winckler, "Traffic signal phase and timing estimation from low-frequency transit bus data," *IEEE Trans. Intell. Transpt. Syst.*, vol. 16, no. 1, pp. 19-28, Feb. 2015.

[5] J. Yu and P. Lu, "Learning traffic signal phase and timing information from low-sampling rate taxi GPS trajectories," *Elsevier Knowledge-Based Syst.*, vol. 110, pp. 275-292, Jul. 2016.

[6] E. Koukoumidis, M. Martonosi, and L.-S. Peh, "Leveraging smartphone cameras for collaborative road advisories," *IEEE Trans. Mobile Comput.*, vol. 11, no. 5, pp. 707-723, May 2012.

[7] KOROAD, *Specification of traffic light controlle*r, Retrieved Feb. 19, 2019, from https://www.koroad.or.kr/cmm/fms/epkoroadFileDown.do?board_code=DTBBS_020&board_num=100319&file_num=177289

[8] H.-Y. Jeong, H.-H. Nguyen J.-H. Park, and J. Kwon, "One-way road information update based on a smartphone navigation system in the OpenStreetMap," *KIISE Database Res.*, vol. 33, no. 2, pp. 16-25, Aug. 2017.

[9] H.-Y. Jeong, "Design and implementation of mobile crowdsourcing-based driver assistance systems (MC-DAS)," *J. IKEEE*, vol. 22, no. 1, pp. 29-37, Mar. 2018.

[10] Open Data Portal, *Current state of traffic lights in Busan*, Retrieved Feb. 19, 2019, from https://www.data.go.kr/dataset/3079345/fileData.do

[11] Korean National Police Agency, *Installation and management manual of traffic lights in 2011*, Retrieved Feb. 19, 2019, from https://www.police.go.kr/portal/bbs/view.do?nttId=71293&bbsId=B0000136&searchCnd=1&searchWrd=&section=005008&sdate=&edate=&useAt=&replyAt=&menuNo=200525&viewType=&delCode=0&option1=005008005&option2=&option4=&option5=&deptId=&larCdOld=&midCdOld=&smCdOld=&orderType=&pageUnit=10&pageIndex=1

[12] A. G. Howard, et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *Comput. Sci.*, *arXiv: 1704.04861*, 2017.

[13] Wikipedia, *HSL and HSV*, Retrieved Feb. 19, 2019, from https://en.wikipedia.org/wiki/HSL_and_HSV

박 재 현 (Jae-Hyun Park)

2018년 2월 : 부산대학교 전기공학과 졸업
2018년 3월~현재 : 부산대학교 전기전자컴퓨터공학과 석사과정
<관심분야> 모바일 컴퓨팅

정 한 유 (Han-You Jeong)

1998년 2월 : 서울대학교 전기공학부 학사
2000년 2월 : 서울대학교 전기컴퓨터공학부 석사
2005년 2월 : 서울대학교 전기컴퓨터공학부 박사
2005년 3월~2007년 7월 : 삼성전자 책임연구원
2008년 1월~2008년 8월 : 미네소타대학교 박사후 연구원
2015년 3월~2016년 2월: 파더본대학교 방문교수
2008년 9월~현재 : 부산대학교 전기컴퓨터공학부 교수
<관심분야> 자동차 통신망, 운전지원 시스템