*Transactions of KSAE, Vol. 27, No. 6, pp.487-494 (June, 2019)*




pISSN 1225-6382 / eISSN 2234-0149
**DOI http://dx.doi.org/10.7467/KSAE.2019.27.6.487**

# Vehicle In the Loop의 모듈화 구조 연구

박 창 우[1) ] · 이 형 철[*2)]

한양대학교 전기공학과[1)] · 한양대학교 전기생체공학부[2)]

# Modular Vehicle In the Loop

Changwoo Park[1)] • Hyeongcheol Lee[*2)]

[1)]*Department of Electric Engineering, Hanyang University, Seoul 04763, Korea*
[2)]*Division of Electrical and Biomedical Engineering, Hanyang University, Seoul 04763, Korea*
***(Received 20 February 2019 / Revised 26 March 2019 / Accepted 3 April 2019)***

**Abstract** : Most vehicle controllers are developed and validated with V-model. The model in the automotive industry has some traditional methods called “X-in-the-loop(XIL).” The validation of the advanced driver assistance system(ADAS) controller, however, is more complicated and needs more environmental resources because the controller interacts with the external environment of the vehicle. Vehicle-in-the-loop(VIL) is a recently developed approach for ADAS vehicle simulation that ensures the safety of critical test scenarios in real-world testing with a virtual environment. This paper presents a new architecture for modular vehicle-in-the-loop(M-VIL), which contains four parts: a sync module, a virtual environment, sensor emulators, and visualizers. Each part can be developed and modified separately and in combination with the other parts. The structure of M-VIL is expected to save on maintenance time and cost. This paper also shows its acceptability by testing ADAS on both a real system and the VIL system.

**Key words** : Vehicle in the loop(가상환경 기반 검증), Advanced driver assistant system(첨단운전자보조시스템), Autonomous driving(자율주행), Validation method(검증 기법), Virtual test(가상 시험)

### Subscripts

ADAS : advanced driver assistance system
AD : autonomous driving
MIL : model in the loop
SIL : software in the loop
HIL : hardware in the loop
VTHIL : vehicle traffic hardware in the loop
VIL : vehicle in the loop

## 1. 서 론

최근 차량에 장착되는 운전자 보조시스템의 종류가 급격하게 증가하고 있다. 이것은 운전자의 편의 및 안전을 증대시킴과 더불어 자동차산업의 회전율을 증가시키는 중요한 역할을 하고 있다.[1,2)] 대부분의 운전자 보조시스템의 경우 기존의 시스템과 복잡하게 얽혀있으며, 매우 빠른 순간에 대한 대처가 필요하다. 운전자와 탑승자의 안전과 밀접한 연관이 있는 운전자 보조시스템의 개발은 보다 신중할 필요가 있을 것이다.[3)]

차량 제어기의 각 개발 단계별로 효과적인 개발도구들이 존재한다. 모델 기반 검증(MIL)은 제어기를 컴퓨터 시뮬레이션 모델로 구현하여 알고리즘을 빠르게 검증하고 개발하는데 매우 유용하게 활용되고 있다. 소프트웨어 기반 검증(SIL)은 대체로 실제 자동차 제어기 (ECU)에 담길 수 있는 형태의 의사코드(Pseudo code)에 대하여 검증하게 된다. 이 두 가지 검증은 대부분 실시간보다 빠른 시간에 시뮬레이션이 가능하여 짧은 시간에 많은 시나리오를 검증할 수 있다. 다음으로 하드웨어 기반 검증(HIL)에서는 실제 ECU의 설계와 동작특성에 대한 전반적인 하드웨어 레벨의 검증을 연구실 환경에서 수행할 수 있다. 마지막으로 실차 단계에서 완성된 시스템에 대한 검증을 수행하게 된다. 실차 단계에서는 차량의 동특

*Corresponding author, E-mail: hclee@hanyang.ac.kr

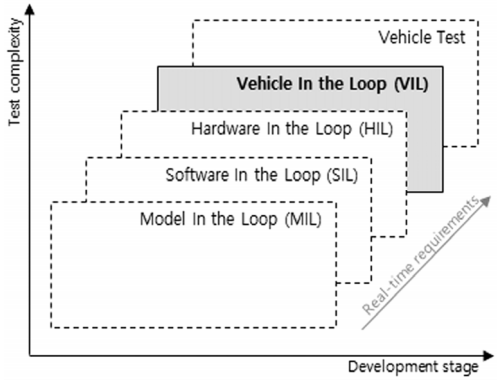


Fig. 1 MIL-SIL-HIL-Vehicle Test

성을 포함하여 가장 신뢰도 높은 검증이 가능하다. 각 단계는 검증에 소요되는 시간과 신뢰도는 트레이드오프(Trade-off) 관계로 볼 수 있다.[3,4]

차량의 Head lamp, Wiper, Smart key 등의 바디 제어 시스템이나, ABS(Anti-lock Brake System), ESC(Electronic Stabilize Control), MDPS(Motor Driven Power Steering) 등과 같은 샤시 제어 시스템 등 전통적인 차량 제어기의 경우 MIL-SIL-HIL 과정을 거친 후 실차 검증단계에서 대상 차량 단독으로 수행하는데 무리가 없다. 그러나 첨단운전자보조시스템(ADAS)이나 자율주행자동차(AD)의 경우 주변의 차량이나 보행자, 차선과 같은 외부적 요인이 검증에 한 요소로 수반되어야 하며, 기존의 실차 검증 방법으로는 단독 수행이 불가능하다.

이러한 한계를 극복하고자, ADAS/AD의 효과적인 실차 검증을 위해 새로운 방안들이 제시되고 있다. 시험차량을 샤시 다이나모미터에서 동작하며 주변 환경을 로봇으로 모사하는 VTHIL(Vehicle Traffic Hardware In the Loop)의 경우 센서를 포함한 ADAS/AD 검증은 적합하지만 차량의 동역학적 특성을 반영하기에는 한계가 있다.[5] 또한 구축비용 및 유지비용이 매우 높으며, 최초 구축된 환경에 부합하지 않은 검증일 경우 자원의 재활용이 어렵다. 다른 방법으로는 ADAS 전용 시험도로를 구축하여 로봇이나 더미(Dummy)로 주변 객체를 모사할 수 있다. 센서의 인지·판단과 동시에 차량의 동특성을 반영하여 실제 환경과 매우 유사하게 검증이 가능하지만, 시험도로의 구축과 개별 시험 비용이 매우 높게 발생한다. 또한 시험 도중 주변 객체와 충돌할 수 있는 위험성이 높게 존재한다.

이러한 문제를 해결하고자 가상환경을 구축하고 이를 활용하는 VIL(Vehicle In the Loop) 시험 기법이 대두되고 있다. VIL은 Fig. 2와 같이 현실 세계의 시험차량과 가상으로 존재하는 외부환경이 융합된 시험환경이다. 실차검증과 동일한 수준의 차량 동특성을 반영할 수 있으며, 시

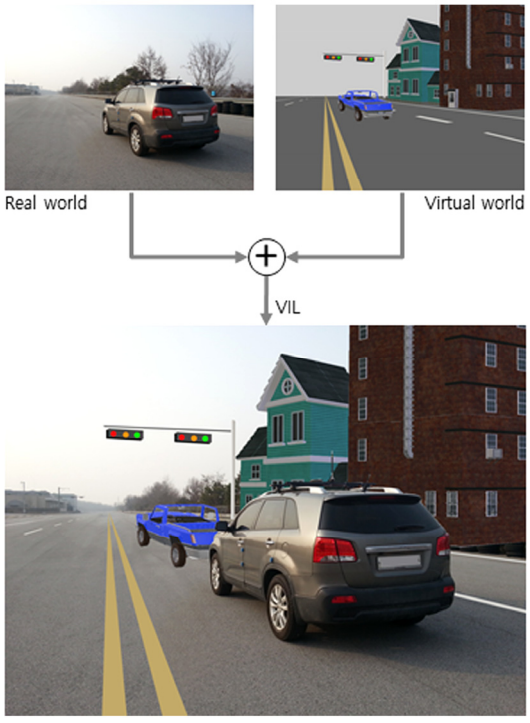


Fig. 2 VIL example

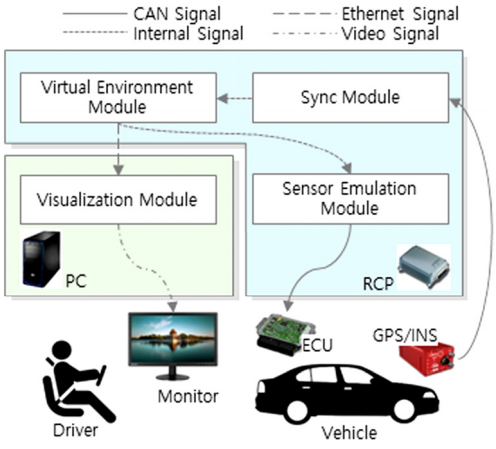


Fig. 3 VIL structure

험에 포함되는 외적요인을 구성하는데 드는 비용을 절약할 수 있다. 동시에 시험 중 발생할 수 있는 외부 객체들과의 충돌 위험을 제거할 수 있다. Thomas Bock은 기존 차량 시뮬레이터에 증강현실(Augmented reality)을 연계하는 하드웨어 구성을 제안하였다. Yvonne Laschinsky는 Active safety light 개발에 적용하여 낮 시간에도 시험을 수행할 수 있음을 보였다. Marius Feilhauer는 가상환경을 인지·판단할 수 있는 센서 구성을 하드웨어와 소프트웨어의 두 가지 방법으로 구분하여 제안하였다.[3,6-8]

다만, 선행 연구들에서는 이미 존재하는 시뮬레이션 시스템에 VIL 환경을 접목하기 위한 방안을 제시하였으

며, 이는 VIL의 핵심적 요소를 정확하게 설명하지 않으며, 시험 환경의 세부내용을 포함하고 있지 않다. 특히 기존 시뮬레이터 환경에만 제한적으로 구현됨을 보였으며, 이는 시험자의 필요에 따라 시험 환경을 수정·변경하는데 한계를 갖는다. 또한 효과적으로 VIL의 개발방안을 제안하는 연구가 아직 존재하지 않는다. 본 논문에서는 VIL의 구성요소를 분리하여 모듈화 함으로써, 각 모듈별로 독립적인 개발 및 변경이 가능한 구조를 제안하고자 한다.

본 논문은 2장에서 VIL을 4가지 모듈의 기본 구조와 각 모듈의 필수 인자를 제시한다. 3장에서는 각 모듈을 통합하는 방법과 실제 ADAS 차량과 VIL의 유사성 비교 실험을 통해 제안된 시스템이 VIL 목적에 부합하는지를 확인한다. 마지막으로 5장에서는 제안된 VIL 시스템의 모듈 가변성을 서술한다.

## 2. Modular Vehicle In the Loop

본 논문에서 제안하는 VIL은 Fig. 3과 같이 4가지의 모듈로 구성된다. VIL의 모듈화를 위하여 시스템의 구성요소를 특성에 맞게 분리하고 각 모듈이 유기적으로 동작하기 위해 필요한 인자들의 정의가 필요하다. 먼저 주변의 차량, 보행자 등의 거동과 도로 설치물, 차선 등의 정보를 포함하는 가상 주행환경 모듈(Virtual environment module), 현실의 환경과 가상 주행환경을 동기화 시켜주는 동기화 모듈(Sync module), 그리고 가상 주행환경의 정보를 ADAS 제어기에 전달하는 센서 에뮬레이션 모듈(Sensor emulation module), 마지막으로 운전자에게 시각적으로 주행 환경을 제공할 수 있는 운전자 시각제공 모듈(Visualization module)로 VIL을 분리·구성할 수 있다. 각 모듈이 포함하는 인자들은 아래 2.1절~2.4절에서 다루도록 하며, 논문에서 제안하는 개별 모듈은 연구자의 요구환경에 따라 변형하거나 상용 툴로 대체하여 유연하게 활용할 수 있을 것으로 기대한다.

### 2.1 Virtual Environment

가상 주행환경 모듈은 ADAS/AD 시험을 위한 외부 환경의 구성이다. 예를 들어, AEB(Advanced Emergency Brake)와 ACC(Adaptive Cruise Control) 제어기는 주변의 차량이나 보행자와 유기적인 관계를 맺으며, LKAS(Lane Keeping Assistance System)제어기의 경우 주행도로의 차선이 주요한 요인이 된다. 또한 높은 수준의 자율주행차량의 경우 신호등이나 표지판, 건물과 같은 외적 요인들에 영향을 받게 된다. 본 논문에서는 이러한 객체들을 시간에 따라 변화가 없는 시불변 객체(Time invariant object)와 시간에 따라 변하는 시변 객체(Time varying object)로 분리하였다. 이를 다시 각 객체가 ADAS/AD 제어기에 미치는 특성에 따라 Table 1과 같이 구분하여 각 객체의 의미 있는 정보를 정의하였다. 공통으로 활용되는 정보는 위치(x, y, z), 자세(yaw, pitch, roll)가 있다.

Table 1 Object database

| | | |
|---|---|---|
| Time invariant object | | |
| ㄴBuilding, Sign | Object information, Location, Rotation | |
| ㄴRoad line | Line information, Location, Rotation | |
| Time varying object | | |
| ㄴSignal light | (T/I) | Location, Rotation |
| | (T/V) | Light information |
| ㄴMoving object | (T/I) | Object information |
| | (T/V) | Location, Rotation |

*T/I: Time Invariant
*T/V: Time Varying

- 시불변 객체 : 건물, 표지판, 기타 구조물 및 도로(차선) 등 시간에 따른 변화가 없는 객체
- 시변 객체 : 신호등의 점등 정보나 주변의 차량, 자전거, 보행자 등 위치 등 시간에 따라 정보가 변하는 객체

Table 1은 시불변 객체와 시변 객체를 구분하여 저장되는 정보 구조이다. 먼저 시불변 객체 중, 건축물이나 표지판 등은 그 종류·형태·크기와 같은 객체정보(Object Information)와 가상환경 공간 내부에서 존재하는 위치(Location)와 자세(Rotation)정보를 포함한다. 차선은 Table 2와 같이 실선, 점선 등의 모양($LT_n$)과 흰색, 노란색 등의 색상($LC_n$) 및 차선 곡률($C_n$)·곡률 변화율($D_n$)이 포함된다. 그리고 차선의 위치($x_n$, $y_n$)·자세($\theta_n$) 정보가 있다. 연속된 차선을 표현하기 위하여 각 정보는 $n$개의 차선 세트로 이루어져 있다.

시변 객체 중, 신호등은 시불변 정보인 위치·자세와 시간에 따라 변하는 점등 시변정보로 구성할 수 있다. 이동형 객체(차량, 보행자 등)는 Table 3과 같이 종류($type$)와 객체의 형태를 표현할 수 있는 형태 노드($n_x$, $n_y$)를 시불변 정보로 정의하고, 이동하는 위치($x_0$, $y_0$)와 자세($yaw$)는 시계열 정보로 구성하게 된다.

Table 2 Example of road line database

| Time invariant | $L_1$ | $L_2$ | … | $L_n$ |
|---|---|---|---|---|
| Line type | $LT_1$ | $LT_2$ | … | $LT_n$ |
| Line color | $LC_1$ | $LC_2$ | … | $LC_n$ |
| Curvature | $C_1$ | $C_2$ | … | $C_n$ |
| ⊿Curvature | $D_1$ | $D_2$ | … | $D_n$ |
| Location | $x_1$, $y_1$ | $x_2$, $y_2$ | … | $x_n$, $y_n$ |
| Rotation | $\theta_1$ | $\theta_2$ | … | $\Theta_n$ |

Table 3 Example of moving object database

| Time invariant | Static data | | | |
|---|---|---|---|---|
| Object type | type - passenger car, truck, pedestrian | | | |
| Shape node | $(n_{x1}, n_{y1})$, $(n_{x2}, n_{y2})$ ⋯ $(n_{xk}, n_{yk})$ | | | |
| Time varying | t(1) | t(2) | ⋯ | t(n) |
| Location | $x_1, y_1$ | $x_2, y_2$ | ⋯ | $x_n, y_n$ |
| Rotation | $yaw_1$ | $yaw_2$ | ⋯ | $yaw_n$ |

가상 주행환경의 각 객체 정보는 매 단위시간마다 업데이트되어 다른 모듈에서 활용 할 수 있도록 한다.

### 2.2 Sync

현실에서 주행하는 시험차량을 2.1절의 객체들이 존재하는 가상환경과 융합하기 위하여 동기화 모듈이 사용된다. 이를 위해 GPS/INS(Global Positioning System / Inertial Navigation System) 장치를 시험차량에 장착하여 현실에서의 시험차량 위치와 자세를 취득하도록 한다. GPS 장치에서 계측되는 차량의 위치는 일반적으로 위·경도 좌표계인 WGS84의 형태를 하고 있다. 하지만 시스템 개발에 직관적이고 사용자의 수용성 향상을 위해 직각 좌표계로 변환이 유용하며, 해당 알고리즘이 동기화 모듈에 포함 되어있다. 본 논문에서는 아래의 방법을 사용하여 GPS에서 취득한 위·경도 좌표인 WGS84(World Geodetic System 1984) 좌표를 평면 직각좌표계 TM (Transverse Mercator)으로 변환하였다.

$$x = (R_N + h)(\phi - \phi_0) + \Delta X$$
$$y = (R_E + h)(\lambda - \lambda_0) + \Delta Y \quad (1)$$

여기에서($\phi_0$, $\lambda_0$)는 각각 위·경도의 원점상수며, 대한민국의 중부원점은(38, 127)이다. $\phi$, $\lambda$, $h$는 GPS에서 취득한 위치의 위·경도 및 고도이다. ($\Delta X$, $\Delta Y$)은 원점 가산값으로 대한민국은 국토지리정보원에서 정한 (2,000,000 6,000,000)으로 계산한다. 다만 본 연구에서는 비교적 좁은 영역에서 수행되는 차량 시험을 목적으로 하므로 원점 가산 값은 시험 위치에 부합하도록 수정할 수 있다. $R_N$과 $R_E$은 북쪽/동쪽 방향 곡률 반경을 의미하며, 지구타원의 장축 ($R_{major}$)과 이심률 ($e$)을 이용하여 아래와 같이 계산한다.

$$R_N = \frac{R_{major}(1-e^2)}{(1-e^2\sin^2\phi)^{3/2}},\ R_E = \frac{R_{major}}{(1-e^2\sin^2\phi)^{1/2}} \quad (2)$$

이심률 ($e$)과 편평률 ($f$)관계는 아래와 같으며 지구의 장축 ($R_{major}$)과 단축 ($R_{minor}$)이 계산에 사용된다.

$$e = \sqrt{1-(1-f)^2} = \sqrt{1-\left(1-\frac{R_{major}-R_{minor}}{R_{major}}\right)^2} \quad (3)$$

차량의 자세는 INS 장치에서 취득한 yaw, pitch, roll 정보를 바로 활용할 수 있다.

### 2.3 Sensor Emulator

현실의 ADAS/AD 제어기는 레이더나 비전 센서 등을 통해 인지·판단된 대상의 정보를 CAN(Control Arear Network) 통신으로 수신하여 목적에 맞는 제어를 수행하게 된다. 다만, VIL 환경에서는 현실의 센서로 인지할 수 있는 외부환경이 존재하지 않는다. 따라서 가상 주행환경의 객체들의 정보와 동기화된 시험차량의 위치를 이용해 센서 정보를 가공하여 ADAS/AD 제어기로 송신하도록 한다. 여기에서는 센서 에뮬레이션 모듈이 위의 역할을 한다. 예를 들어, 레이더 센서 에뮬레이터는 가상 주행환경의 차량이나 보행자 객체와 시험차량과의 상대적인 거리·속도·방위각 등의 정보를, 비전 센서 에뮬레이터는 시험차량을 기준으로 좌·우 차선정보를 가공하여 정해진 규약에 따라 송신하게 된다.

먼저 레이더 센서 에뮬레이터는 가상주행환경의 객체의 위치·자세·형태(크기) 정보와 시험차량의 위치·자세를 통해 가공된 신호를 생성한다. 임의의 객체에 대하여 평면상 중심위치와 선수각이 각각 ($x_0$, $y_0$)와 $yaw$이고, 객체 중심위치로부터 객체의 형태를 표현하는 노드($n$)까지 거리가 ($n_x$, $n_y$)일 때 Fig. 4의 객체의 각 노드의 좌표 ($x_n$, $y_n$)은 아래와 같이 정의된다. 이때, 객체는 고도좌표 ($z$)와 롤·피치 ($roll$, $pitch$)는 0으로 가정한다.

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} \cos(-yaw) & -\sin(-yaw) \\ \sin(-yaw) & \cos(-yaw) \end{bmatrix} \begin{bmatrix} n_x \\ n_y \end{bmatrix} \quad (4)$$

시험차량의 위치와 해당 차량에 장착된 가상센서의 거리를 [$n_x$, $n_y$]로 본다면, 식 (4)를 동일하게 활용하여 현재 센서의 위치좌표를 구할 수 있다. 그리고 센서가 장착된 위치좌표와 각 노드의 위치좌표의 관계를 통해 객체

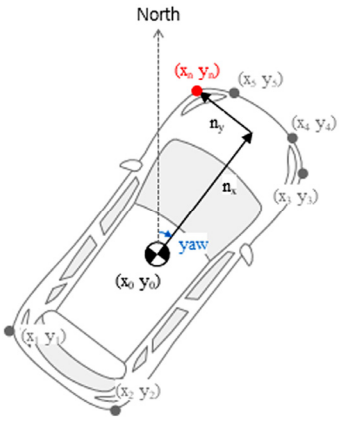


Fig. 4 Node coordination

와의 상대거리·방위각은 유클리드 기하학을 이용하여 Fig. 5로 도식화하며 아래와 같이 계산할 수 있다.

객체를 이루는 노드 A와 B, 센서의 위치를 C, 센서와 객체의 최단거리에 위치한 객체위의 점 H를 아래와 같이 정의할 수 있다.

$$A=\begin{bmatrix}x_1\\y_1\end{bmatrix}, B=\begin{bmatrix}x_2\\y_2\end{bmatrix}, C=\begin{bmatrix}x_s\\y_s\end{bmatrix}, H=\begin{bmatrix}x_h\\y_h\end{bmatrix} \tag{5}$$

이때 센서에서 인지되는 객체와의 최단거리는 아래와 같이 놈(Norm)을 통해 구할 수 있다.

센서와 두 노드를 지나는 선분의 최단거리는 식 (6)과 같이 놈(Norm)을 통해 구할 수 있으며, 위치는 식 (7)로 계산된다.

$$\| \overrightarrow{CH} \| = \frac{\overrightarrow{AB}}{\| \overrightarrow{AB} \|} \cdot \overrightarrow{AC} \tag{6}$$

$$\begin{bmatrix}x_h\\y_h\end{bmatrix} = H = A + \| \overrightarrow{CH} \| \times \frac{\overrightarrow{AB}}{\| \overrightarrow{AB} \|} \tag{7}$$

단, 객체와의 최단거리는 두 노드사이에 존재하여야 한다. 따라서 최단거리의 점 H의 위치에 따라 아래와 같이 최종 최단거리를 선택할 수 있다.

$$\begin{cases} d = \| \overrightarrow{CH} \| & , condition\ CH \\ d = \min( \| \overrightarrow{AC} \|, \| \overrightarrow{BC} \| ) & , else \end{cases} \tag{8}$$
$$condition\ CH: x_h \in [x_1, x_2] \wedge y_h \in [y_1, y_2]$$

식 (8)에서의 조건 CH를 만족하는 경우 인지되는 객체의 방위각은

$$\theta = \tan^{-1}\left(\frac{x_h - x_s}{y_h - y_s}\right) - yaw \tag{9}$$

이로 볼 수 있으며, 이외의 경우에는 만족하는 노드 A 또는 B의 좌표를 이용해 연산하게 된다.

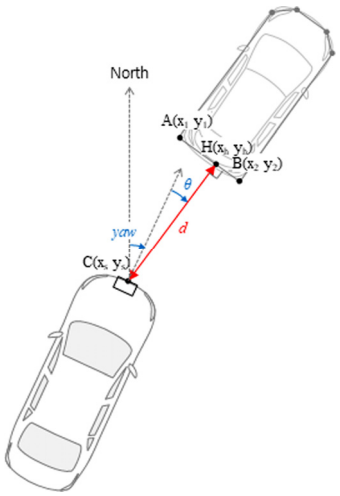


Fig. 5 Calculation of a detected object information

비전 센서 에뮬레이터는 차량에서 인지하는 좌·우 차선에 대한 정보나 영상으로부터 판단되는 외부 환경 정보를 송신하는 역할을 한다. 본 논문에서는 차선에 대해서만 고려하고자 한다. 차선의 정보는 일반적으로 시험차량과의 거리($A$), 입사각($B$), 차선의 곡률($C$)과 곡률의 변화율($D$) 등으로 정의할 수 있다.

Fig. 6에서 차량과 인접한 차선($L$)의 위치정보 ($x_L$, $y_L$)와 자세정보($\theta_L$)를 이용해 차선과 차량과의 거리($A_L$)와 입사각($B_L$)을 식 (10)과 식 (11)과 같이 계산할 수 있다. 여기에서 ($x_0$, $y_0$)은 차선과 차량과의 거리관계를 정의하는 차량내의 위치이며, 일반적으로 전륜의 중심축으로 본다. $yaw$는 차량의 선수각이다.

$$A_L = \sqrt{(x_L - x_0)^2 + (y_L - y_0)^2} \tag{10}$$

$$B_L = yaw - \theta_L \tag{11}$$

또한 미리 저장된 차선의 곡률($C_L$)과 곡률의 변화율($D_L$)을 호출하여 센서 정보를 효과적으로 이용할 수 있다. 차선의 곡률과 곡률의 변화율은 Fig. 6에서 보인 것처럼, 다항식 곡선 접합(Polynomial curve fitting)을 통해 시험 전 도출이 가능하다. 검증환경에 따라 차선의 정보 ($LT_L$, $LC_L$)를 사용할 수도 있다.

레이더와 비전 센서 에뮬레이션 모듈에서 생성한 인지·판단 정보는 실제 존재하지 않는 객체에 대하여 실제 환경 센서를 대체할 수 있다.

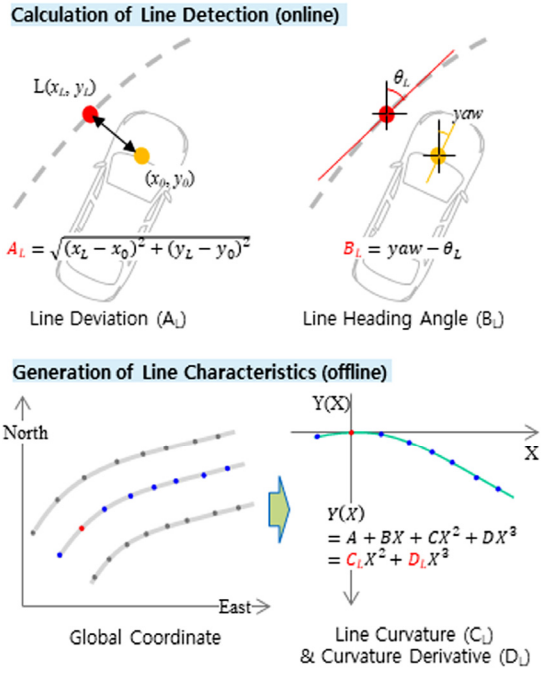


Fig. 6 Calculation of road line information

### 2.4 Visualization

가상 주행환경의 실시간 정보는 차량 제어기와 더불어 운전자에게 제공함으로써 시험수행의 편의성을 높이고, 운전자의 주행감성 평가도 가능하도록 한다. 이를 위해 가상 주행환경을 시각화할 수 있는 모듈이 필요하며 본 연구에서는 Unreal Engine 4를 기반으로 3D 시각화 프로그램을 구현하였다. Unreal Engine 4는 효율적으로 게임을 개발할 수 있도록 하는 Epic Games 社의 소프트웨어이다. 최근에는 게임 외의 분야에서 다양하게 활용되고 있으며, 특히 컴퓨터 시뮬레이션 데이터를 시각화 하는 연구에 널리 활용되고 있다.

가상 주행환경 모듈의 각 객체들과 현실의 시험차량의 위치 및 자세 데이터를 수신하여 이를 시각화하여 운전자에게 제공하게 된다. PC환경에서 수행되는 프로그램에 유리하도록 Ethernet 통신을 기반으로 해당 정보를 공유하였다. 특히 UDP 방식을 통한 빠른 데이터 전송과 Broadcast를 통한 다수의 장치에서 정보를 동시에 취득하는 확장성을 기대하도록 하였다. 가상 주행환경으로부터 낮은 전송주기로 수신한 객체들의 실시간 위치·자세 정보를 객체에 적용하면 연속적으로 이동하는 효과를 볼 수 있다. 이때 시험차량 객체 종속되는 가상의 카메라가 존재하며, 여기서 인지되는 영상을 외부로 출력하게 된다. 해당 영상을 모니터 등의 디스플레이 장치를 통해 운전자가 가상 주행환경의 실시간 정보를 인지할 수 있도록 한다. Fig. 7은 Unreal Engine 4를 통한 운전자 시각화 제공 모듈의 구조를 보이고 있다.

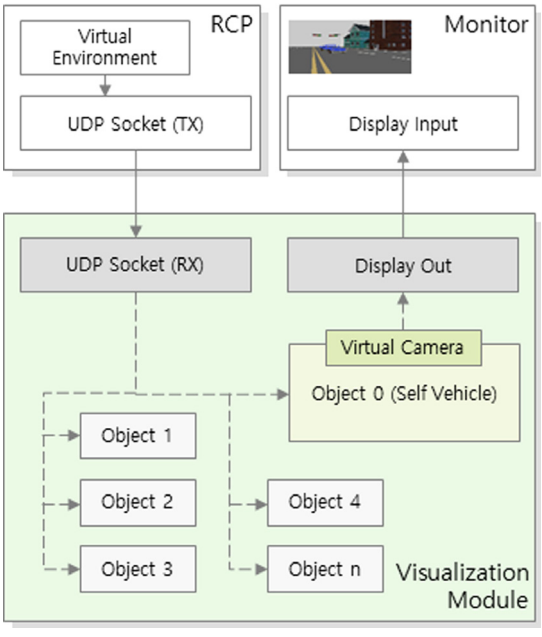


Fig. 7 Visualization module structure

## 3. Field Test

2장에서 제안한 VIL의 각 모듈은 운전자 시각제공 모듈을 제외하고 모두 MATLAB/Simulink 기반으로 구현할 수 있으며 이는 차량 제어기 개발에 널리 활용되는 환경으로, VIL 개발의 진입장벽을 낮출 수 있다. 운전자 시각제공 모듈도 비교적 개발의 편의성이 높으며 MATLAB/Simulink와 유사한 Unreal Engine 4의 Blueprint 개발도구를 이용하므로 개발 진입성에 장점을 지닌다.

본 논문에서는 VIL의 실차평가 환경 구성을 위해 OxTS社의 GPS/INS(RT3100), dSPACE社의 RCP(Micro-AutoBox) 및 Lenovo社의 노트북컴퓨터(T470p)를 이용하였다. 차량검증을 위한 VIL은 실시간성과 안정성이 보장되어야 한다. 이를 위해, 가상 주행환경 모듈, 동기화 모듈, 센서 에뮬레이션 모듈은 차량용 RCP(Rapid Control Prototyping) 장비에서 동작하도록 하였다. 일반적으로 차량용 RCP의 경우 MATLAB/Simulink와 연계성이 매우 높기 때문에 개발비용을 단축하는데 유리한 점이 있다. Unreal Engine 4를 기반으로 구현한 운전자 시각화 모듈은 비교적 저사양의 노트북 컴퓨터나 모니터가 연결된 베어본 PC에서 동작할 수 있음을 확인하였다. 시험에 사용된 노트북컴퓨터는 Intel i5-7440HQ CPU, 16GB RAM와 GeForce 940MX GPU를 탑재하고 있으며, 데이터 송신에 따른 이미지의 변화를 60 FPS 카메라로 촬영하여 실시간성을 확인하였다. 촬영한 영상을 분석한 결과 최대 1 Frame의 차이가 있었으며, 이는 약 1/60초로 실제 시험에 참여한 운전자는 지연을 인지하지 못하였다.

Fig. 3에서 보인 바와 같이, 각 하드웨어 장치는 다수의 통신으로 연계된다. GPS/INS에서 취득한 시험차량의 위치·자세 정보는 CAN 메시지로 RCP에 전달된다. RCP에서는 해당 정보를 통해 시험차량과 가상 주행환경을 동기화 하여 센서 에뮬레이션 결과를 차량의 ADAS 제어기로 CAN 메시지 송신을 하게 된다. 이와 동시에 시험차량과 가상 주행환경 객체의 실시간 위치·자세 정보를 운전자 시각제공 모듈인 3D 시각화 프로그램이 실행되고 있는 컴퓨터로 UDP(User Datagram Protocol) 전송하여 운전자에게 가상환경을 인지하도록 하였다. Photo. 1은 차량에 장착된 VIL 하드웨어 구성과 VIL의 시험환경을 보이고 있다.

제안한 VIL 시스템의 차량시험 신뢰도를 확인하기 위해 AEB 시험평가를 실제 더미와 레이더 센서를 이용한 환경과 논문에서 제안한 VIL 환경에서 수행하였다. 10 m/s로 주행하는 시험차량 전방에 정차된 차량이 존재하는 시나리오를 동일하게 적용하여 각 5회의 시험을 반복하였다. 본 시험은 Euro NCAP에서 제안하는 도심지 CCRs


Photo. 1 VIL equipments RCP, GPS/INS and testing scene

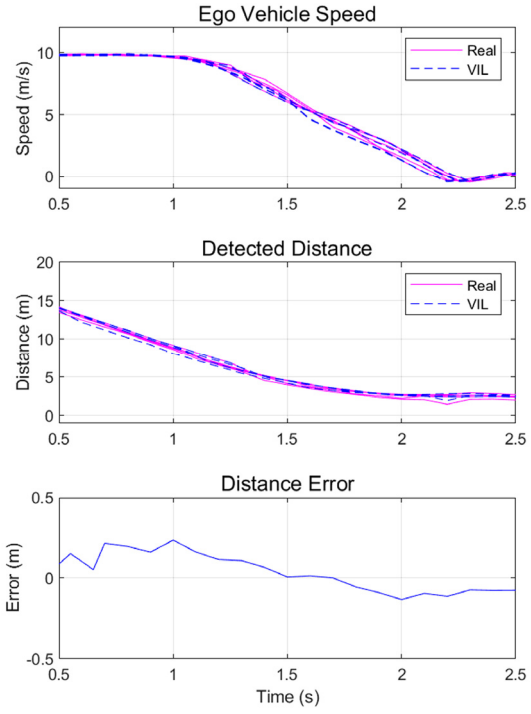


Fig. 8 Real target and virtual target testing result (Radar sensor)

(Car-to-Car Rear Stationary)를 기준으로 하였다. 시험 결과 AEB에 의해 차량 정차 후, 전방 차량과의 거리 평균은 실제 더미를 이용한 환경에서 2.456 m와 VIL을 이용한 시험에서 2.3829 m로 측정되었다. 두 시스템은 약 0.073 m의 차이를 보여 매우 신뢰도 높은 것으로 판단할 수 있다. 특히, 동일한 환경에서 수행된 5회 실험 내에서 최대 오차가 각각 1.2388 m와 0.6348 m로 발생함을 보였으며, VIL 환경의 센서 에뮬레이터가 매우 이상적이라고 할 때, 차량의 동특성에 의해 0.6348 m의 차이가 발생하였

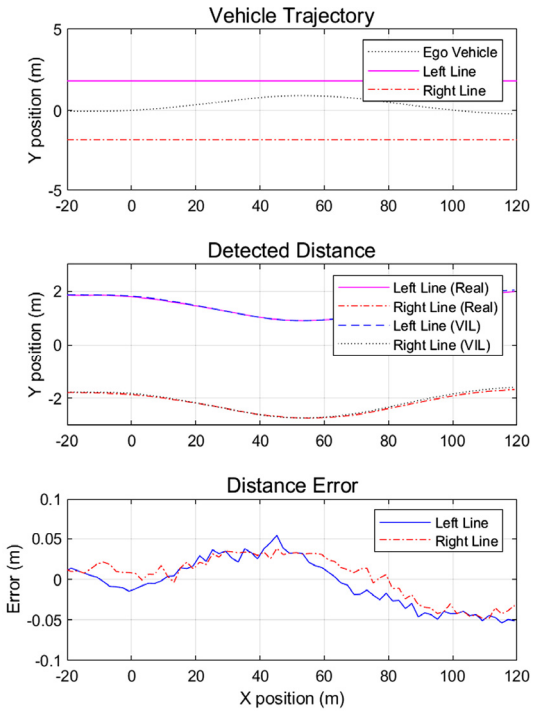


Fig. 9 Real target and virtual target testing result (Vision sensor)

고, 실제 더미의 1.2388 m 차이는 차량의 동특성과 더불어 실제 레이더 센서 오차에 의해 발생한 AEB 제어 차이를 포함하고 있음을 유추할 수 있다. Fig. 8에서는 VIL의 레이더 센서 에뮬레이터에서 인지·판단되는 차량 간 상대거리와 AEB 제어기가 장착된 차량의 거동이 매우 유사함을 보이고 있다.

그리고 약 70 km/h 속도로 폭 3.65 m의 도로를 직선 주행 중, 좌측 차선에 횡속도 0.4 m/s로 접근하는 LKAS 시험을 수행하였다. 이때 실제 비전 센서와 VIL의 비전 센서 에뮬레이터에서 측정되는 차량과 좌·우 차선의 거리를 비교하였다. Fig. 9에서 인지되는 좌·우 차선의 거리가 매우 유사함을 확인할 수 있으며, 총 140 m 구간에서 실제 비전 센서와 VIL에서 생성된 차선과의 최대 거리오차는 5.4 cm로 나타났다. 이는 약 3 %정도의 오차로 실제 비전 센서의 오차율보다 낮으므로 전체 시스템에 큰 영향을 미치지 않을 것으로 판단된다.

## 4. 결 론

본 논문에서는 제안한 VIL의 모듈화 구조를 통해 가상 주행환경의 객체를 정의하고, 동기화 모듈, 센서 에뮬레이션 모듈 그리고 운전자 시각제공 모듈과의 연계성을 보였다. 모듈화를 통해 필요에 따라 특정 모듈을 변경하게 될 경우에도 나머지 모듈의 재활용이 가능하다. 예를 들어, 주변 객체의 거동을 모사하기 위해서 차량 동역학

을 다루는 상용 소프트웨어를 활용하거나 GPS 등을 통해 취득한 차량의 실제 거동데이터를 활용할 수 있을 것이다. 또는, 직접 작성한 차량 모델을 실시간으로 적용하는 방안도 생각해 볼 수 있다. 또한, 각 센서의 특성에 따라 센서 에뮬레이션 모듈을 변경하여 적용할 수 있으며, 유사한 방법으로 라이다(Lidar)나 초음파 센서(Ultrasonic sensor)를 추가 할 수 있을 것이다. 운전자 시각화 모듈의 경우 Unity 등의 다른 플랫폼으로 변경이 가능하다. 또한 일반 모니터가 아닌 VR(Virtual Reality), AR(Augmented Reality)과 같은 HMD(Head Mounted Display) 기기를 활용하면 보다 현실감 있는 시험을 경험할 수 있도록 시험 환경을 확장할 수 있다. 이처럼, 각 모듈의 개발·수정은 서로 다른 모듈로부터 독립적으로 수행할 수 있어 시스템의 유지·보수비용을 감소시킬 수 있음을 기대할 수 있다.

마지막으로 AEB와 LKAS 시나리오에 대한 VIL 실차 시험을 진행하여 실제와의 유사성을 보였다. 다만, 연구 환경의 제약으로 보다 다양한 ADAS/AD 시나리오에 대한 검증의 한계가 있었으며, 향후에는 고도의 시나리오 검증과 함께 센서 모델의 다양성을 확보하여 시스템의 신뢰도를 확인되어야 할 것이다.

## 후 기

본 연구는 산업통상자원부 산업핵심기술개발사업 “ADAS의 시험 평가를 위한 실차 시험용 가상 주행환경 구현 및 차량 주행 동기화 장치 개발기술 개발(10052501)” 과제의 지원으로 수행되었음.

## References

1) Y. Song, “Autonomous Vehicle Technology and Trend in Europe,” Auto Journal, KSAE, Vol.40, No.6, pp.17-20, 2018.
2) Y. Song, “Trends of Autonomous Driving and Its Mobility as a Service,” Auto Journal, KSAE, Vol.40, No.10, pp.18-20, 2018.
3) M. Feilhauer, J. Haering and S. Wyatt, “Current Approaches in HiL-Based ADAS Testing,” SAE International Journal of Commercial Vehicles, Vol.9, No.2, pp.63-69, 2016.
4) K. von Neumann-Cosel, M. Dupuis and C. Weiss, “Virtual Test Drive-provision of a Consistent Tool-set for [d, h, s, v]-in-the-loop,” Proceedings of the Driving Simulation Conference Monaco, pp.1-9, 2009.
5) Y. Hwang, S. Shin and I. Yang, “Evaluation of Advanced Driver Assistance Systems with Vehicle-Traffic Hardware-in-the-Loop Simulations,” Annual Autumn Conference & Exhibition, pp.1338-1339, 2013.
6) T. Bock, K. -H. Siedersberger and M. Maurer, “Vehicle in the Loop-Augmented Reality Application for Collision Mitigation Systems,” Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR’05), pp.1-9, 2005.
7) T. Bock, M. Maurer and G. Farber, “Validation of the Vehicle in the Loop (vil); a Milestone for the Simulation of Driver Assistance Systems,” Intelligent Vehicles Symposium, IEEE, pp.612-617, 2007.
8) Y. Laschinsky, K. von Neumann-Cosel, M. Gonter, C. Wegwerth, R. Dubitzky and A. Knoll, “Evaluation of an Active Safety Light Uing Vrtual Test Drive within Vehicle in the Loop,” 2010 IEEE International Conference on Industrial Technology (ICIT), pp.1119-1122, 2010.