



도 있지만 자칫 큰 교통 사고로 이어질 수 있기에 실제 교통 상황을 재현한 시뮬레이터 환경에서 실험이 진행되고 있다. 하지만 기존 자율주행 시뮬레이터는 고비용으로 일반 연구자가 접근하는데 제약이 있다. 따라서 본 연구에서는 오픈 소스 플랫폼인 CARLA 를 활용한 저비용 자율주행 자동차 시뮬레이터를 제안하고, 타당성과 사용성을 검증하고자 한다.

## 2. 자율주행 시뮬레이터 개발

### 2.1 조건부 자율주행 시뮬레이터

본 연구에서는 조건부 자율주행(SAE 3 단계) 시뮬레이터를 제작하고자 한다. 조건부 자율주행 환경에서는 차량이 제어할 수 없는 돌발상황 발생 시 운전자가 제어권을 전환 받아 수동 운전을 수행해야하기 때문에 ‘제어권 전환 기능’이 필수적으로 제공되어야 한다. 제어권 전환 기능은 자율주행 중 운전자가 운전대/브레이크페달/가속페달을 조작 시 수동으로 운전을 할 수 있도록 하는 기능을 말한다. 제어권 전환 기능 외에도 ‘자율주행 기능’, ‘돌발상황 시나리오 기능’, ‘정보 로깅 기능’이 제공되어야 한다. 본 연구는 상기한 네가지 기능을 제공하는 조건부 자율주행 시뮬레이터를 제작하기 위해 CARLA 를 활용하였다. CARLA [7]는 헤르만 로스와 안토니오 로페즈를 주축으로 이루어진 연구팀에서 개발한 오픈소스 플랫폼으로 자율주행 알고리즘 테스트를 목적으로 하기 때문에 본 연구에서 개발하고자 하는 조건부 자율주행 시뮬레이터의 필수 기능을 모두 지원하지 않는다.

### 2.2 자율주행 기능

본 기능은 실험 설계자가 정한 경로대로 차량이 자동으로 주행하는 기능을 말한다. 본 기능을 구현하기 위해서 CARLA 의 플러그인 모듈인 Leaderboard [8]를 활용하였다. 이 모듈은 원래 다양한 교통 상황에서 자율주행 에이전트의 운전 능력을 평가하기 위해 개발되었다. 본 연구에서는 이 모듈의 기능 중 하나인 자율주행 경로 설정을 사용하기 위해 이 모듈을 활용했다. 이 모듈에서는 xml 형식으로 실험 설계자가 자동차의 자율주행 경로를 직접 설정할 수 있다. 경로를 설정하기 위해서는 먼저 <routes> 태그를 생성한 후 태그 내에 주행 경로 정보를 담고 있는 <route> 태그를 생성하면 된다. <route> 태그 생성 시 어떤 지도의 경로인지 나타내는 town 속성을 추가하여야 한다(예: <route town="Town06">). 다음으로 <route> 태그 내에 <waypoint> 태그를 활용하여 주행 경로의 좌표를 이동 순서대로 추가하면 된다. 추가한 좌표는 CARLA 내부에서 waypoint 객체로 읽어 들이게 되며, 첫 번째 좌표는 자

율자동차가 생성되는 위치, 마지막 좌표는 시뮬레이터가 종료되는 위치이다. CARLA 는 OpenDrive 형식의 지도를 이용하는데, OpenDrive 에는 일방통행과 같이 특정 도로에서 특정 방향으로만 이동할 수 있도록 설정하는 것이 가능하기 때문에 지도의 설정을 자세히 알고 있다면 효율적으로 경로를 설정할 수 있다.

```
<?xml version="1.0" encoding="UTF-8"?>
<routes>
  <route id="0" town="Town06">
    <waypoint pitch="0.0" roll="0.0" x="-164" y="48" yaw="0" z="0.0" />
    <waypoint pitch="0.0" roll="0.0" x="-67" y="48" yaw="0" z="0.0" />
    <waypoint pitch="0.0" roll="0.0" x="100" y="48" yaw="0" z="0.0" />
  </route>
</routes>
```

그림 1 xml 형식 예시

### 2.3 돌발상황 시나리오 기능

본 기능은 자율주행 차량이 제어할 수 없는 상황, 예컨대 주행중인 차량 앞에 보행자가 갑자기 무단횡단 하는 상황 등을 발생시키는 기능을 말한다. 본 기능은 Carla 의 플러그인 모듈인 Scenario Runner [9]를 활용하여 구현되었다. 해당 모듈은 CARLA 시뮬레이터에 대한 교통 시나리오를 정의하고 실행하기 위해 개발되었다. Scenario Runner 는 13 가지의 돌발상황 시나리오를 기본으로 제공하고 있다: (1) 앞선 차량의 속도 감속 후 정차 시나리오, (2) 앞선 차량의 급정차 시나리오, (3) 앞선 차량의 속도 변화 시나리오, (4-5) 좌/우회전 후 자전거 무단횡단 시나리오, (6) 보도에 있던 보행자의 무단횡단 시나리오, (7) 보이지 않던 보행자의 무단횡단 시나리오, (8) 교차로에서 신호 위반 차량을 맞닥뜨리는 시나리오, (9-10) 교차로 좌/우회전 시나리오, (11) 신호 없는 교차로 시나리오, (12-13) 비보호 좌/우회전 시나리오. 이 시나리오들은 Leaderboard 에서 지정한 경로와 함께 사용할 수 있다. 기본으로 제공되는 시나리오 외에도 파이썬 코드 형식으로 실험 설계자가 시나리오를 직접 제작할 수도 있다. 시나리오를 제작하기 위해서는 Scenario Runner 모듈의 내부에 있는 srunnerWscenarios 디렉토리에 BasicScenario 를 상속한 클래스를 생성한 후 시나리오에 필요한 Actor 와 돌발상황 정보를 정의하면 된다. Actor 는 시나리오에 출현하는 자율주행 자동차 외에 다른 차량/보행자/자전거 등을 의미하는데, Actor 의 종류 및 생성 위치가 정의되어야 한다. 돌발상황 정보로는 돌발상황 발생 조건, 발생 시 Actor 행동, 종료 조건이 정의되어야 한다. 돌발상황 발생 조건으로 자율주행차와 Actor 까지의 거리를 설정하면 된다. 해당 거리안에 자율주행 자동차가 들어오면 돌발상황이 발생하게 된다. 다음으로는 행동 트리에 Actor 의 행동, 돌발상황 발생 조건, 시나리오 종료

조건을 추가하면 시나리오가 완성된다. 행동 트리는 인공지능의 행동을 정의하는 트리이다.

#### 2.4 제어권 전환 기능

본 기능은 자율주행 중 운전자가 운전대/브레이크페달/가속페달을 조작 시 수동으로 운전을 할 수 있도록 하는 기능을 말한다. 수동 운전 기능을 제공하기 위해서는 시뮬레이션 시스템과 운전대 및 페달 컨트롤러의 연동이 필요하다. CARLA는 pygame을 통해서 컨트롤러의 입력을 받을 수 있다. 하지만 Leaderboard는 키보드 입력만 지원하기 때문에 운전대 및 페달 컨트롤러의 입력 값을 받기 위해서는 HumanAgent 클래스의 수정이 필요하다. HumanAgent 클래스는 CARLA의 DualControl 클래스를 참조하여 수정하면 된다. 이 때 기본으로 제공되는 기능은 Logitech사의 G29를 기준으로 구현되어 있으므로, 실험 설계자가 보유한 컨트롤러에 따라서 버튼을 다시 매핑해야 한다. 버튼을 매핑하기 위해서는 시뮬레이터를 실행하는 로컬 디렉토리에 있는 wheel.ini 파일을 수정하면 된다. HumanAgent 클래스 수정 시 KeyboardControl 클래스의 자율주행 관련 코드를 가져와 추가한 후 컨트롤러의 입력이 들어오면 자율주행 모드가 해제되도록 코드를 작성해서 제어권 전환 기능을 구현할 수 있다. 자율주행 모드 해제 기능을 구현할 때 운전자가 컨트롤러에 손과 발을 올려놓는 동안 발생하는 미세한 노이즈 값에 제어권이 전환될 수 있으므로 0.5의 임계값을 넘어야 전환되도록 설정한다. 해당 임계값은 실험설계자가 보유한 컨트롤러에 따라 적절하지 않을 수 있기 때문에 테스트를 통해서 조정할 필요가 있다.

#### 2.5 정보 로깅 기능

본 기능은 차량주행 및 돌발상황 시나리오의 정보를 로그 데이터로 기록하여 추후에 활용할 수 있도록 하는 기능이다. 로깅 기능은 파이썬의 로깅 모듈을 사용해서 구현한다. 운전자 입력 정보는 운전자의 운전대 및 페달 입력과 자율주행해제시점을 말하고, 차량주행 정보는 차량의 위치, 방향, 속도 정보를 말한다. 운전대, 페달 입력 정보는 운전자가 조작하는 컨트롤러의 입력 정보를 의미하며, 자율주행해제시점은 자율주행 중 운전제어권 전환이 이루어져 운전자가 수동운전을 시작한 시점을 의미한다. 운전자 입력 정보를 기록하는 기능은 Leaderboard의 HumanAgent 클래스에 구현하면 된다. 그리고 이 기능을 활용하면 돌발상황 발생시 운전자의 반응정보도 기록할 수 있다. 차량주행 정보를 기록하는 기능은 Leaderboard의 ScenarioManager 클래스에서 구현하면 된다. 돌발상황 시나리오의 정보는 Actor의 생성 위치, 정의한 행동의 실행 시각 등의 정보를 말

한다. 이러한 정보는 Scenario Runner에서 시나리오의 행동을 정의할 때 로깅을 포함시킴으로써 구현할 수 있다.



그림 2 시뮬레이터 장비 및 개발된 시스템 사진

### 3. 자율주행 시뮬레이터의 타당성 및 사용성 스터디

#### 3.1 실험 절차

자율주행 시뮬레이터의 타당성 및 사용성을 조사하기 위해 총 10명의 참여자(나이: 평균 = 22.3, 표준편차 = 1.49, 중앙값 = 22)를 모집하여 실험을 진행하였다. 참여자 중 남자는 6명이며, 운전 면허를 소지한 사람은 5명이었다. 실험에서는 자율주행 시뮬레이터의 타당성 및 사용성을 알아보기 위해 상기한 시뮬레이터의 세 기능을 평가하였는데, 실험은 다음과 같은 순서로 진행하였다. 먼저 실험 참가자를 자리로 안내하고 운전석을 편하게 조절하도록 하였다. 다음으로 실험의 목적과 내용을 설명하고 준비된 연습용 시나리오를 1회 진행하였다. 이후 본실험용 시나리오 1, 2를 차례로 실행했는데, 실험 참가자마다 먼저 실행하는 순서를 다르게 하였다. 시나리오 1은 자율주행 알고리즘이 50km/h의 속도로 직진 주행을 하는 도중 좌측에 있는 보행자가 차도로 돌진한 뒤 차량 앞에서 멈춰 있는 시나리오이고, 시나리오 2는 시나리오 1의 위치보다 더 앞선 위치에서 우측에 있는 보행자가 차도로 돌진한 뒤 차량 앞에서 멈춰 있는 시나리오이다. 참가자로 하여금 운전대만을 조작해서 추돌을 회피하도록 하였으며, 이 때 보행자가 출발한 시각과 참가자가 운전대를 조작한 시각을 각각 기록하여 운전자의 반응 속도를 측정하였다. 마지막으로 실험 참가자를 대상으로 시뮬레이터에 대한 인터뷰를 진행하였다.

### 3.2 실험 결과

운전 제어권 전환 기능을 평가하기 위해 본 연구에서 개발한 시뮬레이터와 상용 시뮬레이터 간의 실험 데이터를 비교했다. 비교한 데이터는 돌발상황 시나리오에서 보행자가 무단횡단을 시작한 순간부터 실험 참가자가 운전대를 꺾어서 반응하기까지 걸린 시간이다. 10명의 참가자가 두 번씩 시나리오를 진행하여 총 20회의 데이터를 수집하였다. 이 데이터를 활용하여 반응 시간을 분석했을 때 참가자의 반응 시간은 0.763초부터 2.199초 사이에 분포했으며, 평균은 1.453초(중앙값: 1.360, 표준편차: 0.355)였다. 이 반응속도는 기존 상용 시뮬레이터로 진행된 연구의 결과와 비교했을 때 조금 느렸다. STISIM Drive 라는 상용 시뮬레이터에서 50km/h 의 속력으로 전방 차량이 급정지 했을 때 운전자의 반응속도는 평균 1.01 초(표준편차: 0.45 초)로 측정되었다 [10]. 이러한 차이는 돌발상황 시나리오의 차이로 야기되었다고 참작되는데, 본 연구에서는 기존 연구와는 달리 보행자가 인도에서 차도로 달리는 시나리오로 진행되었기 때문에 운전자가 주변 시야에서 발생한 돌발상황을 판단하고 반응해야 했기 때문이다.

실험이 끝난 뒤에는 인터뷰를 진행하였다. 인터뷰에서는 시뮬레이터가 보완되어야 할 점에 대한 참가자의 의견을 수집하였다. 인터뷰에서 나온 대표적인 의견은 (1) 부자연스러운 핸들링과 (2) 부자연스러운 보행자의 행동이었다. 부자연스러운 핸들링은 실제 자동차와 운전대의 민감도가 일치하지 않은 것을 의미한다. 참가자 10명 중 7명이 “핸들을 조금만 꺾어도 차가 많이 움직여서 어색했어요. (참가자#4)”라는 의견을 제시하였다. 그리고 참가자#9 는 “지면과의 마찰이 적은 것 같아요.”라는 의견을 제시하였다. 부자연스러운 보행자의 행동은 보행자가 무단횡단을 하는 속도가 지나치게 빠르다는 것을 의미한다. 참가자 10명 중 8명이 “사람이 뛰쳐나오는 속도가 너무 빨라요. (참가자#6)”라는 의견을 제시하였다.

표 1. 실험 정보 요약

	반응 속도(초)	참가자 나이(만)
최댓값	2.199	25
최솟값	0.763	20
평균	1.453	22.3
중앙값	1.359	22
표준편차	0.355	1.49

### 4. 결론

본 연구에서는 오픈 소스 플랫폼인 CARLA 를 활용하여 조건부 자율주행 환경(SAE 3 단계)을 모의실험 할 수 있는 시뮬레이터를 개발하였다. 시뮬레이터의 기능을 실험한 결과 보완이 필요하지만 모의실험에 사용하기에 문제가 없음을 알 수 있었다. 따라서 비싼 시뮬레이터를 구매하지 않더라도 CARLA 를 활용해서 충분히 연구에 활용가능한 시뮬레이터를 개발할 수 있음을 알 수 있다. 실험 과정에서 지적인 보완사항은 차후 수정하여 소스 코드를 공개할 예정이다.

### 사사의 글

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구이다(No. NRF-2021R1F1A1063337).

### 참고 문헌

1. Global Semi-Autonomous Vehicle Market Size, Share & Trends Analysis Report by Level of Automation (Level 1, Level 2, Level 3), by Vehicle Type (Passenger Car, Commercial Vehicle), by Region, and Segment Forecasts, 2021-2028. Grand View Research, 2021.
2. Nelson, Gabe. "Tesla beams down 'autopilot' mode to Model S". Automotive News. <https://www.autonews.com/article/20151014/0EM06/151019938/tesla-beams-down-autopilot-mode-to-model-s>. November 19, 2021.
3. (R) Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. SAE. J3016. 2021.
4. Pashler, Harold. Dual-task interference in simple tasks: data and theory. Psychological bulletin. 116.2: 220. 1994.
5. Wickens, Christopher D. Multiple resources and mental workload. Human factors. 50.3: 449-455. 2008.
6. Sanbonmatsu, David M., et al. Why drivers use cell phones and support legislation to restrict this practice. Accident Analysis & Prevention. 92: 22-33. 2016.

7. Dosovitskiy, Alexey, et al. CARLA: An open urban driving simulator. In: Conference on robot learning. PMLR. p1–16. 2017.
8. CARLA Autonomous Driving Leaderboard. <https://leaderboard.carla.org>. November 19, 2021.
9. CARLA ScenarioRunner. <https://carla-scenariorunner.readthedocs.io/en/latest/>. November 19, 2021.
10. Winter, JCF De, et al. The effects of driving with different levels of unreliable automation on self-reported workload and secondary task performance. International journal of vehicle design. 70.4: 297–324. 2016.