문서 이력				
Version	작성일	변경사유	작성자	변경 내용 요약
1.0	2022.11.08		고영배 외	

ADS 연동 V2X 프로토콜 검증 시뮬레이터 제작

- 프로그램 사용 설명서 및 결과서 -

목차

1. 개요

본 문서는 ADS 연동 V2X 프로토콜 검증 시뮬레이터 용역을 수행하며 제작한 시뮬레이터의 프로그램 사용 설명서 및 결과서이다. 시뮬레이터는 Python 기반으로 제작되었다. 본 사용 설명서는 PyCharm IDE를 사용하는 환경에 맞춰 제작되었다.

본 보고서의 구성은 다음과 같다. 2장에서 시뮬레이터를 구동 및 개발하기 위한 환경 설정 방법을 기술한다. 3장에서는 시뮬레이터의 사용 방법에 대해서 기술한다.

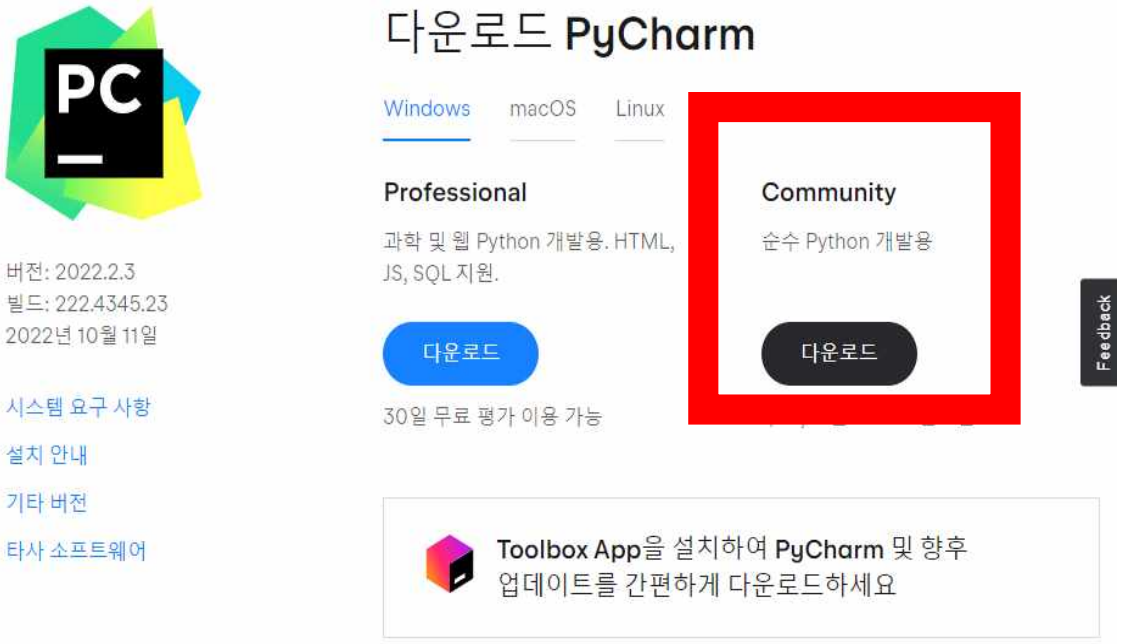
2. 시뮬레이터 구동 및 개발 환경 설정

2.1 PyCharm IDE 다운로드 및 설치

시뮬레이터 구동 및 개발을 위해 PyCharm IDE를 설치하도록 한다. 다운로드 및 설치 방법은 아래에 서술한다.

2.1.1 PyCharm 다운로드

- https://www.jetbrains.com/ko-kr/pycharm/download/#section=windows 에서 Community 버전을 선택하여 설치를 진행한다.



[그림 2.1] PyCharm 다운로드

2.1.2 설치 진행

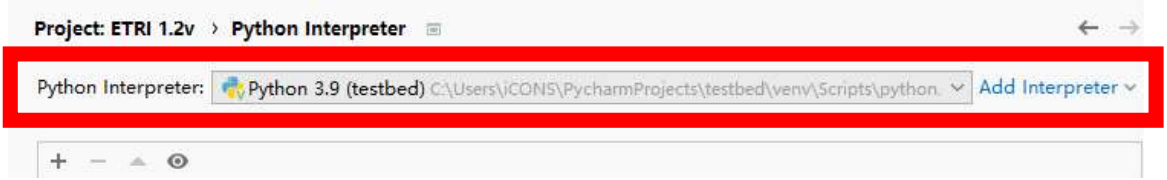
- 설치 파일을 실행시키고 진행하면 된다.
- 다른 설정은 하지 않아도 되며, 기본 설치로 설치를 진행한다.

2.2 Python 패키지 설치

- 시뮬레이터를 구동하기 위해서는 개발에 활용된 라이브러리 및 추가 패키지 설치가 필요하다. 해당 패키지들은 PyCharm을 통해 간단하게 추가 설치가 가능하며, 설치 방법 및 순서는 아래와 같다.

2.2.1 python 인터프리터 설정

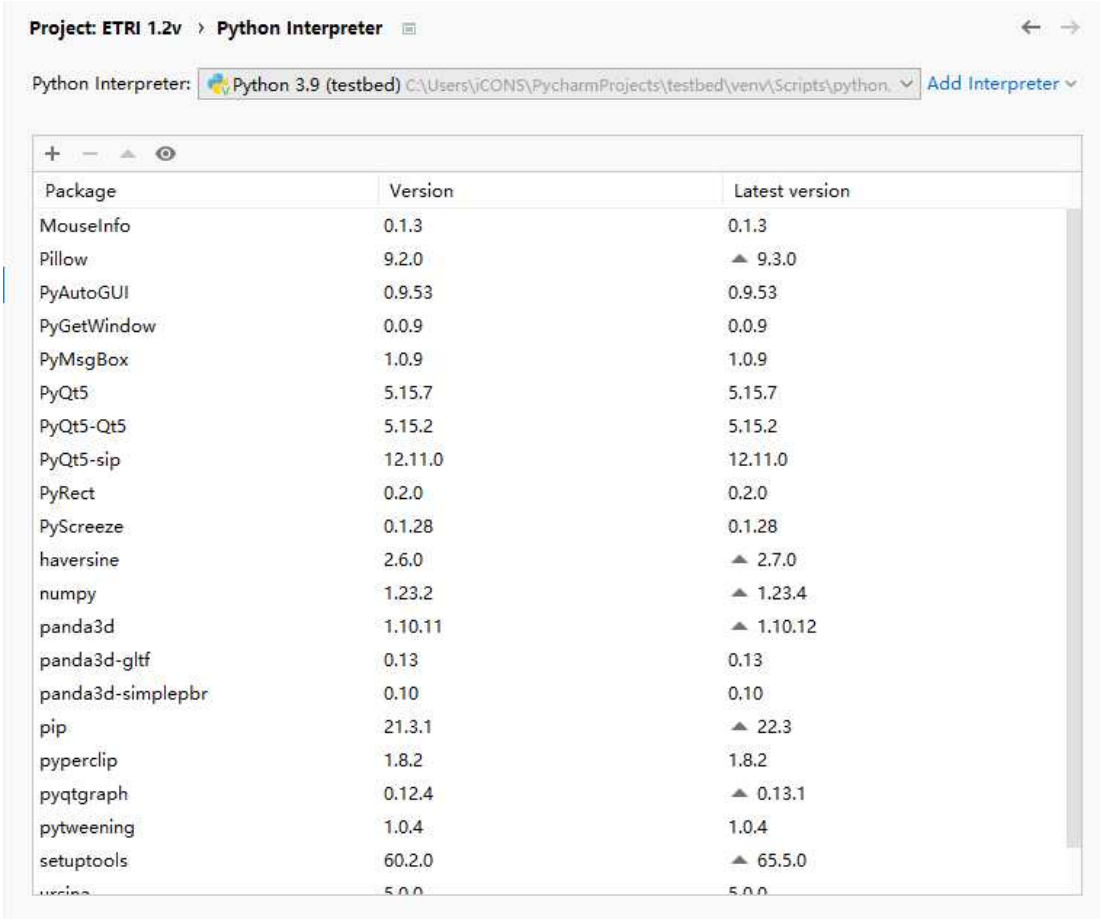
- 상단 탭 맨 왼쪽 File -> Settings -> Project : "Name" -> python Interpreter에서 아래 그림의 해당 부분에서 python Interpreter를 설정하여준다. (3.6 버전 이상의 python을 설정)



[그림 2.2] Python Interpreter 설정

2.2.2 ursina 설치

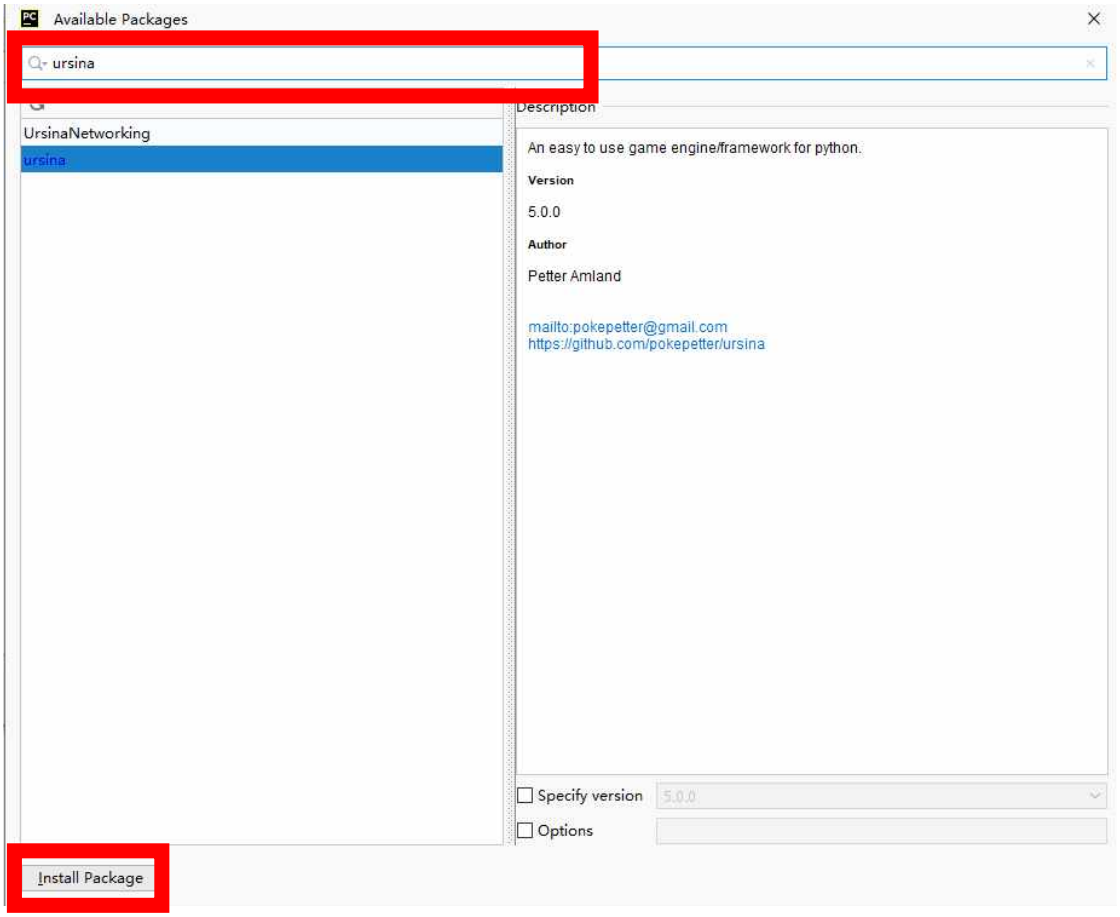
- 시뮬레이터에서 GUI를 제공하기 위한 엔진 라이브러리로, 해당 패키지는 오픈 소스 게임 엔진 패키지로 시뮬레이션의 핵심적인 오브젝트와 도로를 구현한다.



Package	Version	Latest version
MouseInfo	0.1.3	0.1.3
Pillow	9.2.0	▲ 9.3.0
PyAutoGUI	0.9.53	0.9.53
PyGetWindow	0.0.9	0.0.9
PyMsgBox	1.0.9	1.0.9
PyQt5	5.15.7	5.15.7
PyQt5-Qt5	5.15.2	5.15.2
PyQt5-sip	12.11.0	12.11.0
PyRect	0.2.0	0.2.0
PyScreeze	0.1.28	0.1.28
haversine	2.6.0	▲ 2.7.0
numpy	1.23.2	▲ 1.23.4
panda3d	1.10.11	▲ 1.10.12
panda3d-gltf	0.13	0.13
panda3d-simplepbr	0.10	0.10
pip	21.3.1	▲ 22.3
pyperclip	1.8.2	1.8.2
pyqtgraph	0.12.4	▲ 0.13.1
pytweening	1.0.4	1.0.4
setuptools	60.2.0	▲ 65.5.0
ursina	5.0.0	5.0.0

[그림 2.3] 추가 패키지

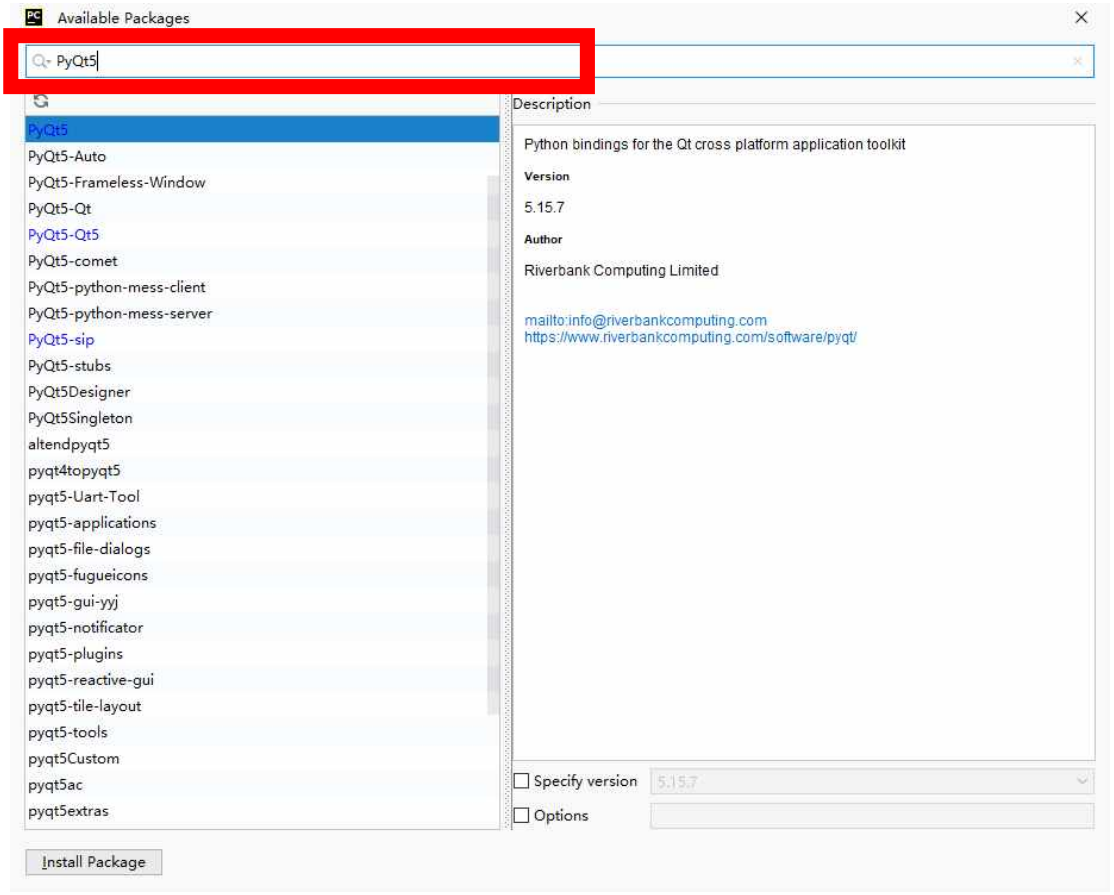
- Interpreter 설정 창에서 “+” 표시를 누른 후, 패키지 목롱 창이 떳으면, 검색창에 ursina를 검색한 후, 해당되는 패키지를 선택 후 좌측 하단의 Install Package를 누르면 설치가 진행된다.



[그림 2.4] Ursina 라이브러리 설치

2.2.3 PyQt5 설치

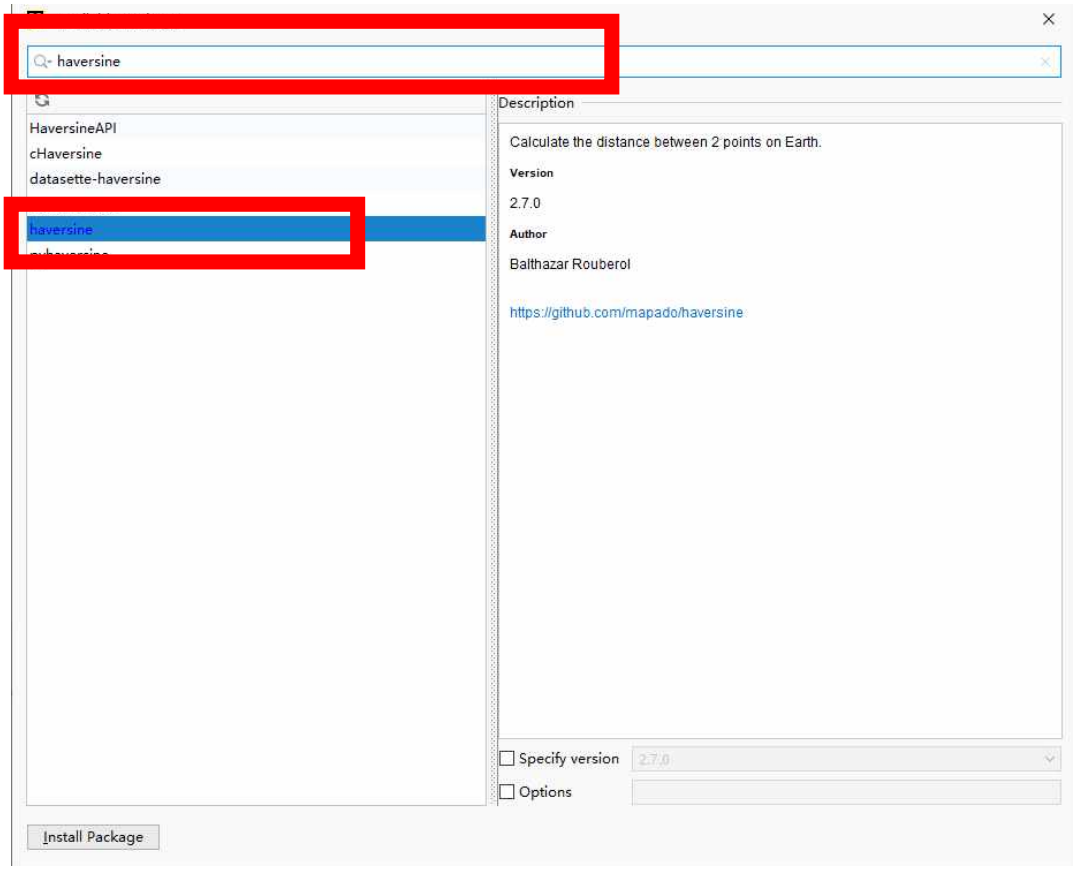
- 시뮬레이션에서 차량들의 속도와 Log를 위한 기본 패키지이다. 해당 패키지를 통해 컨트롤 패널을 다양하게 구현할 수 있다.
- ursina 설치 과정과 동일하게 설치를 진행하면 된다.



[그림 2.5] PyQt5 라이브러리 설치

2.2.4 haversine 설치

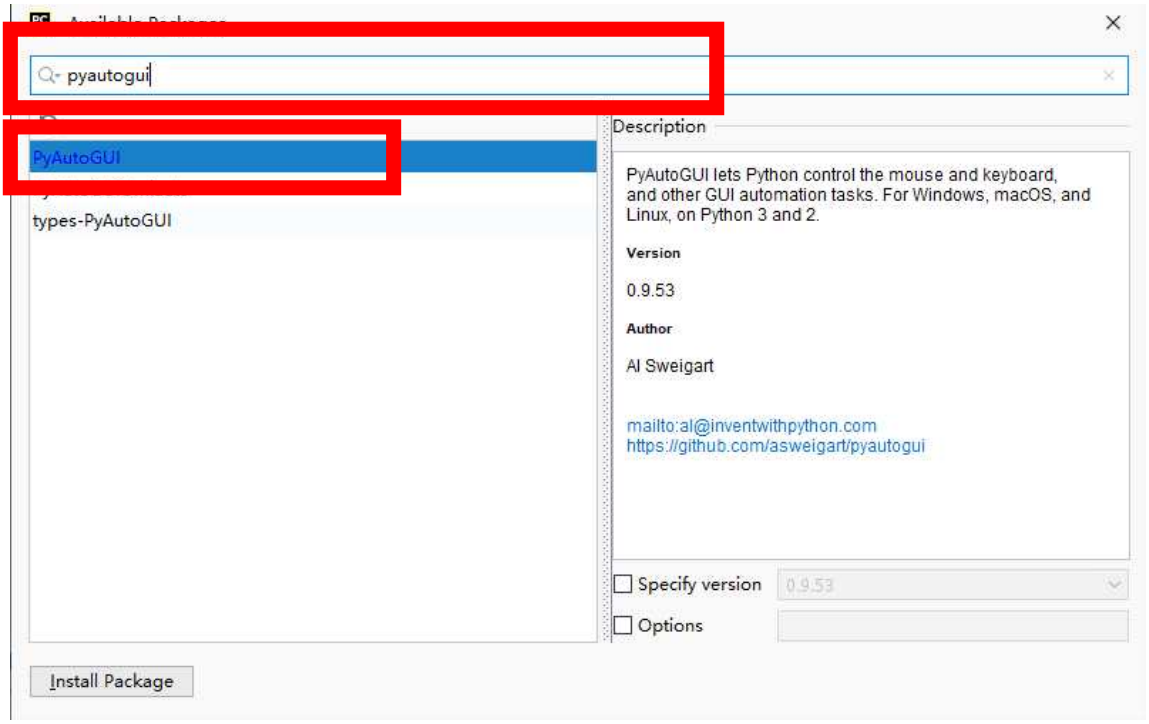
- 차량의 이동거리를 더욱 정확하고 정밀하게 표현하기 위해 사용되는 패키지이다. 위경도 값 데이터를 이용해서 두 지점 간 거리를 구할 때 유용하게 쓰인다.
- ursina 설치 과정과 동일하게 설치를 진행하면 된다.



[그림 2.6] haversine 라이브러리 설치

2.2.5 pyautogui 설치

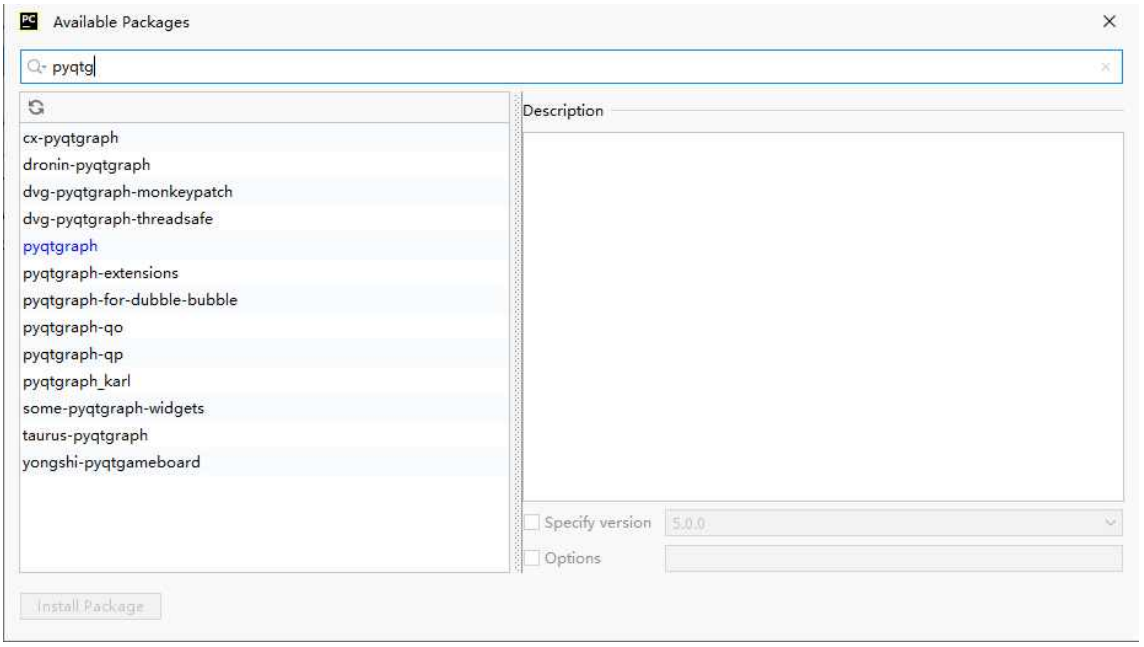
- 시뮬레이터의 화면 조절을 위하여, 사용한다.
- 모니터의 해상도를 이용하여 화면 크기 조절을 한다.



[그림 2.7] pyautogui 설치

2.2.6 pyqtgraph 설치

- PyQT5와 numpy를 기반으로 만들어진 GUI 라이브러리
- 차량의 속도를 나타내는 그래프로 사용한다.



[그림 2.8] pyqtgraph 설치

3. 시뮬레이터 사용 설명서

먼저 시뮬레이터를 구동 및 개발하기 위해 PyCharm IDE를 실행한다. 또는 소스코드 변경없이 실행만을 위해서는 제공된 exe 파일을 실행한다.

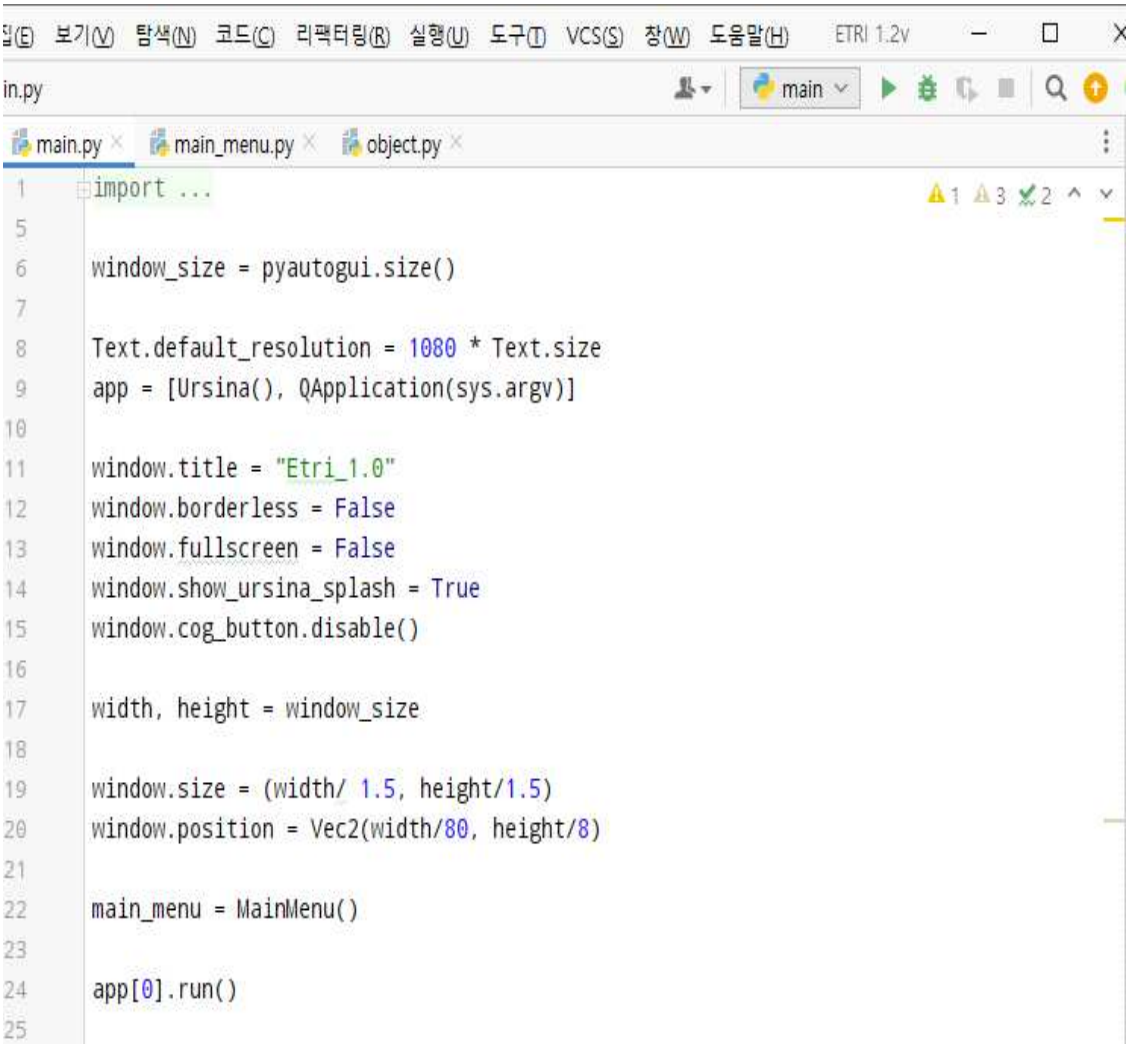
3.1 시뮬레이터 실행

1) 실행 환경 설정

- 압축 파일을 압축 해제한 후, 해당 폴더를 PyCharm에서 열어준다.
- 2장의 과정을 수행하여 패키지와 실행 환경을 설정한다.
- 해당 폴더의 “objects” 폴더가 사라질 경우, 차량의 이미지와 메인 화면의 로고 글씨가 나타나지 않으니 해당 폴더의 관리는 더욱 주의해야 한다.

2) PyCharm 에서 실행

- main.py 파일을 실행
- 실행파일을 main.py로 설정
- ▶ 버튼을 클릭하여 실행



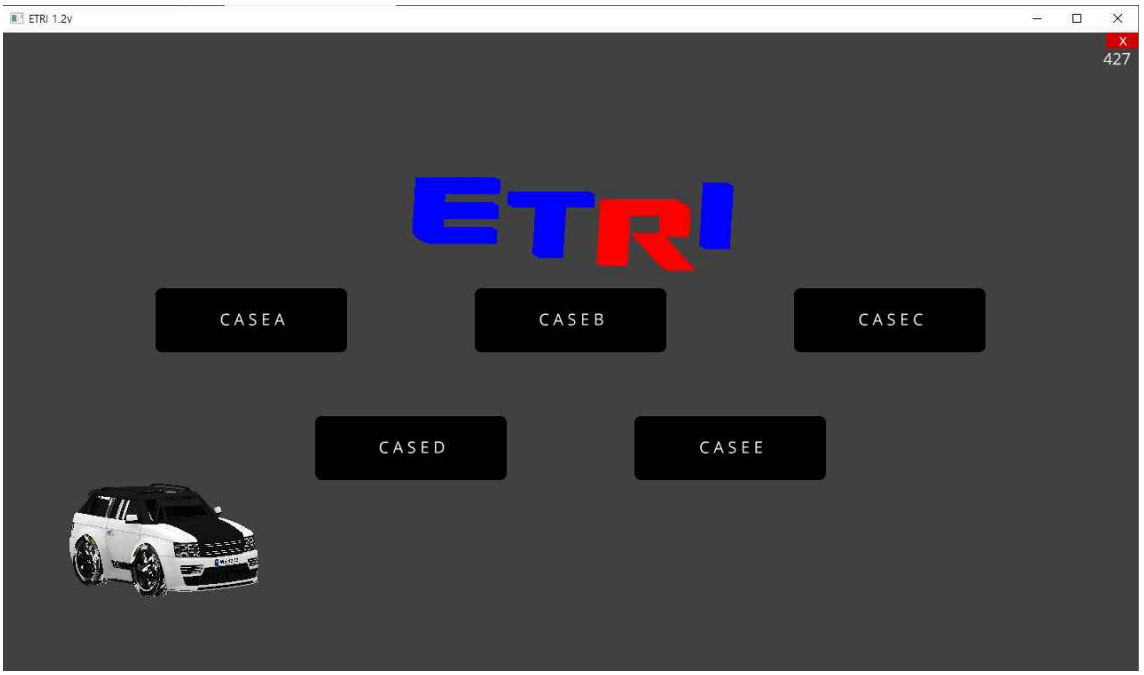
[그림 3.1] PyCharm을 이용한 시뮬레이터 실행 예시

3.2 시뮬레이터 사용 방법

1) 메인 화면

- CASE A,B,C,D,E 각 버튼을 누르면 해당 CASE에 대하여, 시뮬레이션을 진행할 수 있다.
- CASE A = 우합류
- CASE B = 끼어들기
- CASE C = 추월
- CASE D = 긴급양보

- CASE E = 긴급양보

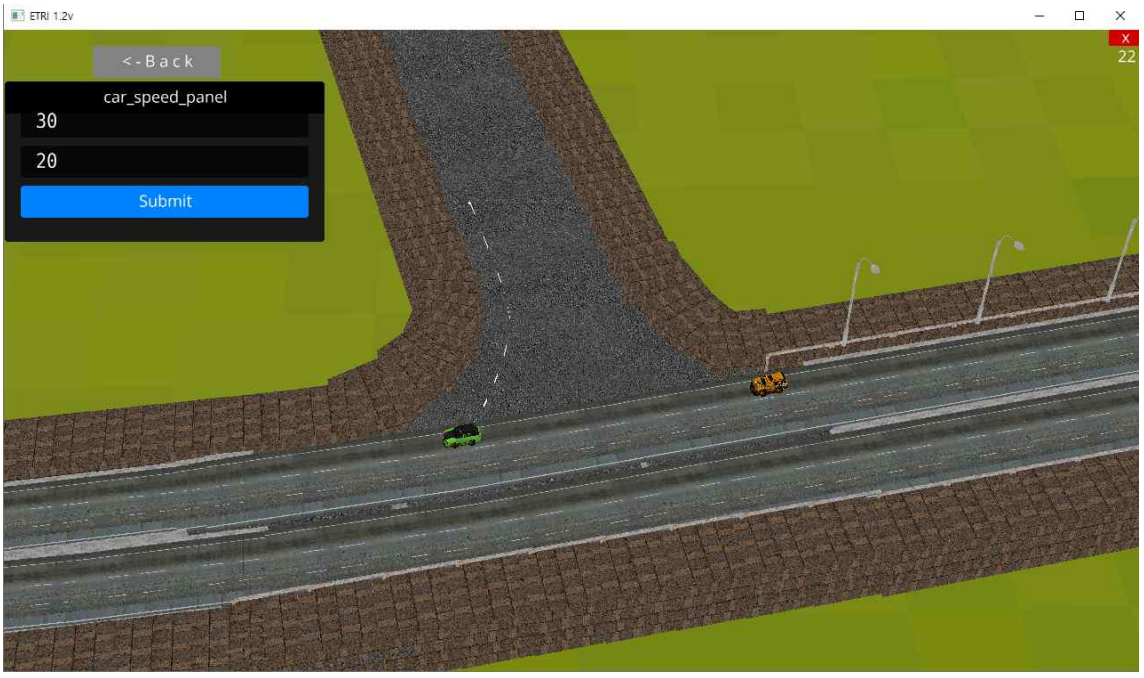


[그림 3.2] 시뮬레이터 Class 선택 화면

2) 속도 변경

- 각 CASE를 눌러서 시작하게 되면, 해당 패널이 왼쪽 상단에 나타나게 된다.
- 해당 패널을 통해 차량의 속도를 변경할 수 있다.
- 패널의 위의 값은 녹색 차의 속도를, 아래 값은 노란 차의 속도를 나타냅니다.
- 해당 칸의 값을 입력한 후, Submit 버튼을 눌러 차량들의 속도를 변경할 수 있다.
- 또한 변화된 속도의 정보를 PYQT 창에서 속도 그래프가 바뀌는 것을 확인할 수 있다.

- 그래프의 순서는 위에서부터 ADS, 녹색차, 노란차 순의 속도 정보를 나타내는 그래프이다.



[그림 3.3] 시뮬레이터 동작 화면

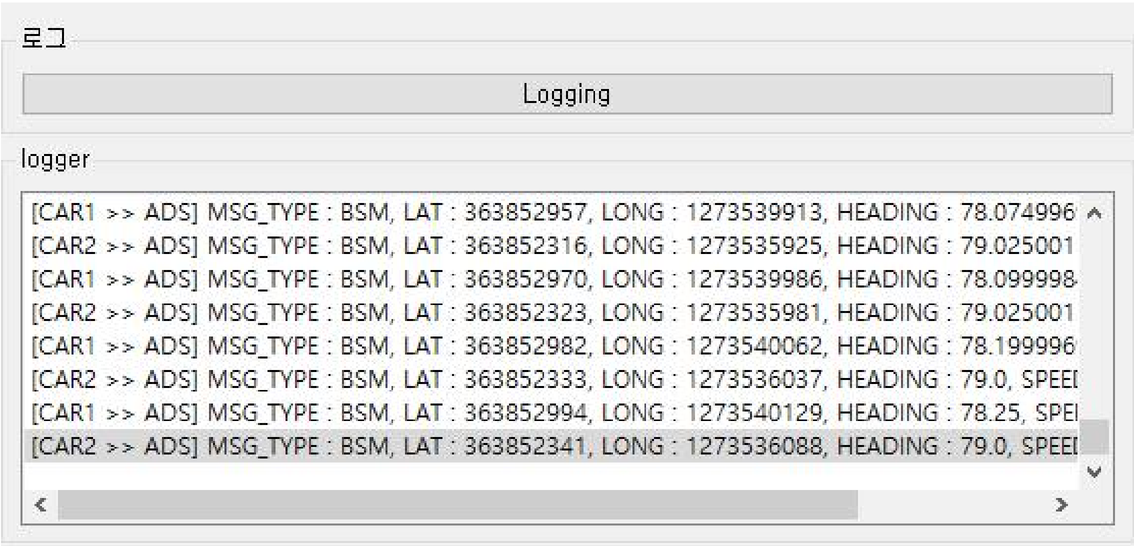
3) 키 입력 제어

- 해당 시뮬레이션에서는 차량의 움직임을 키보드의 입력을 통해 제어할 수 있다.
- “s”키를 누르게 되면, 해당 차량이 주행을 정지한 상태로 멈추게 되고, 해당 위치에서의 정보를 보낸다.
- “d”키를 누르게 되면, 해당 차량이 다시 주행을 시작하게 된다.
- “r”키를 누르게 되면, 시뮬레이션의 차량들이 초기 위치에서 다시 주행을 시작하게 된다.

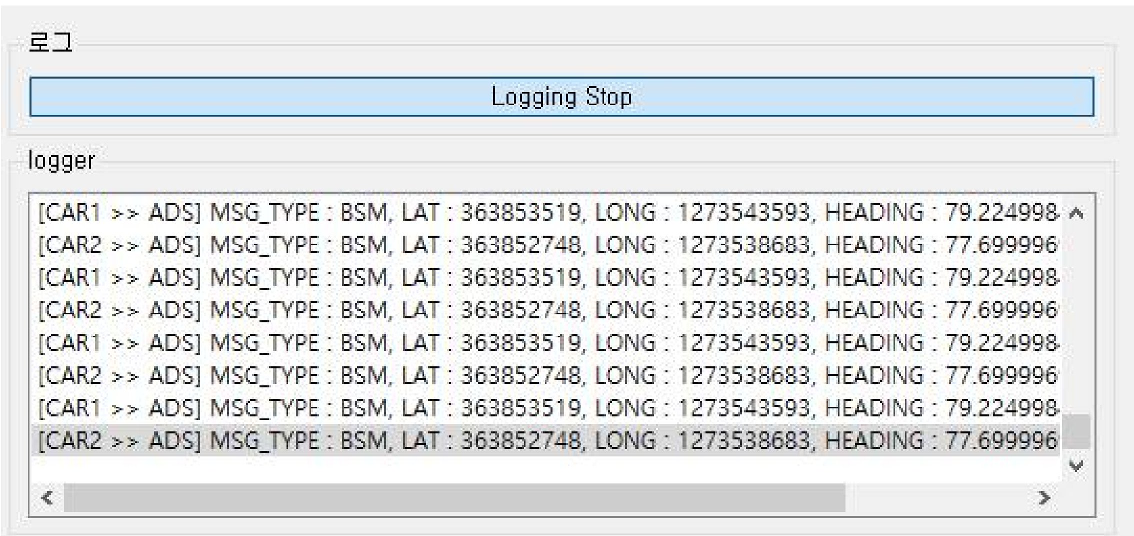
[그림 3.4] 실시간 속도 모니터링 패널

4) Log 창 제어

- Log 창에 시뮬레이터의 데이터가 출력된다.
- 버튼을 통하여, Logging 하거나 멈출 수 있다.
- 버튼에 Log창 작동 정보가 나온다.
- Logger에 나오는 데이터는 Log 파일에 따로 저장된다.



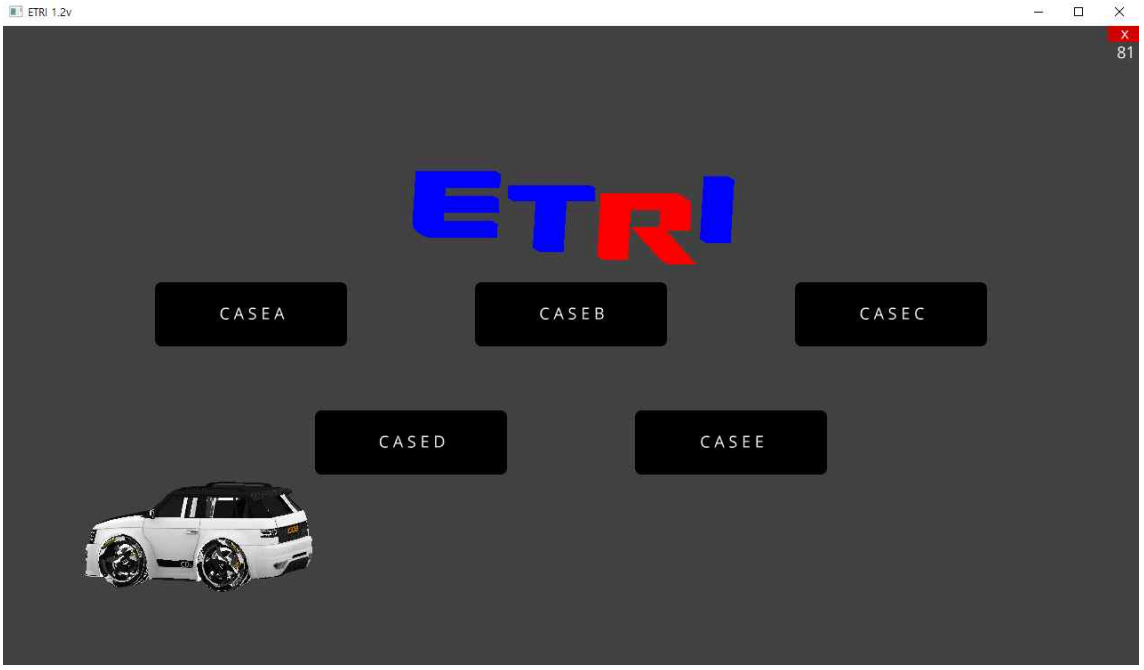
[그림 3.6] 작동중인 Log 창



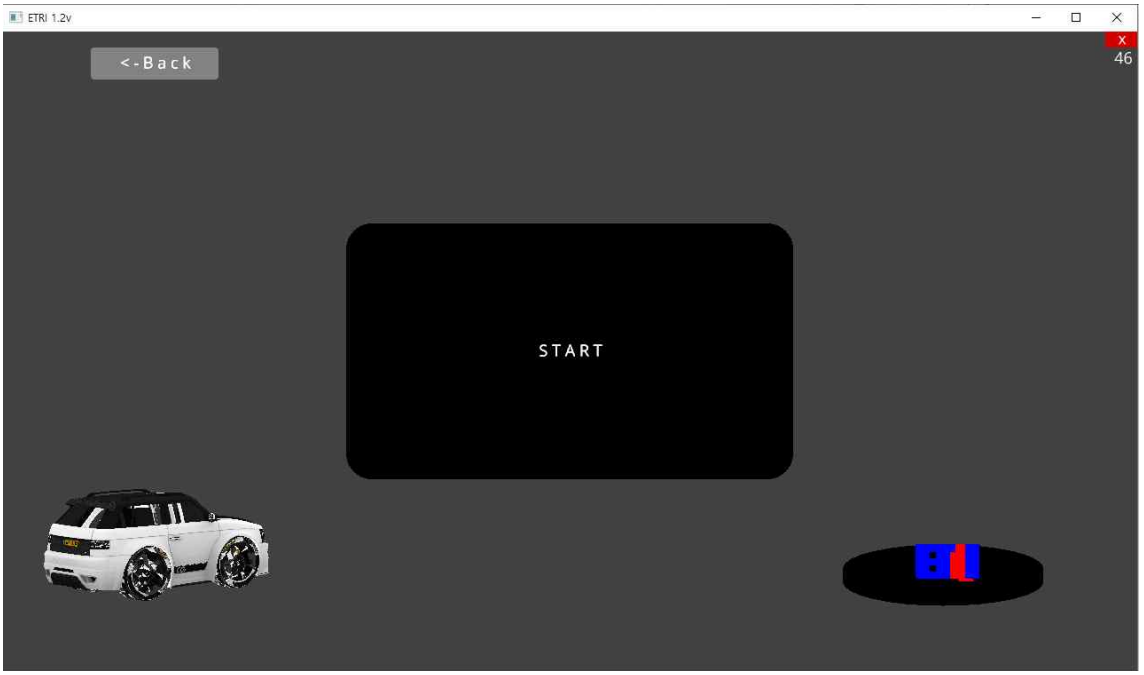
[그림 3.7] 멈춰있는 Log 창

3.3 시뮬레이터 작동 화면

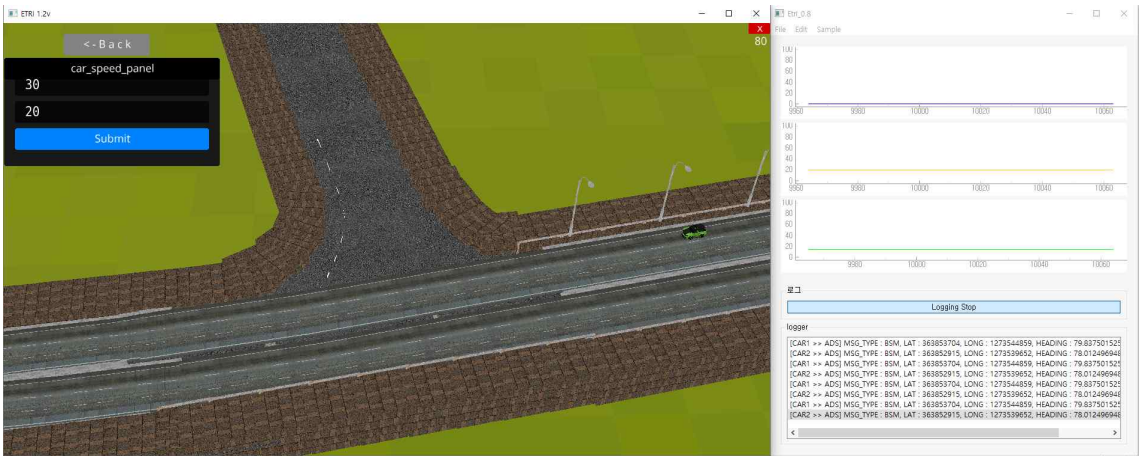
1) 메인 화면



2) case 버튼 선택 후



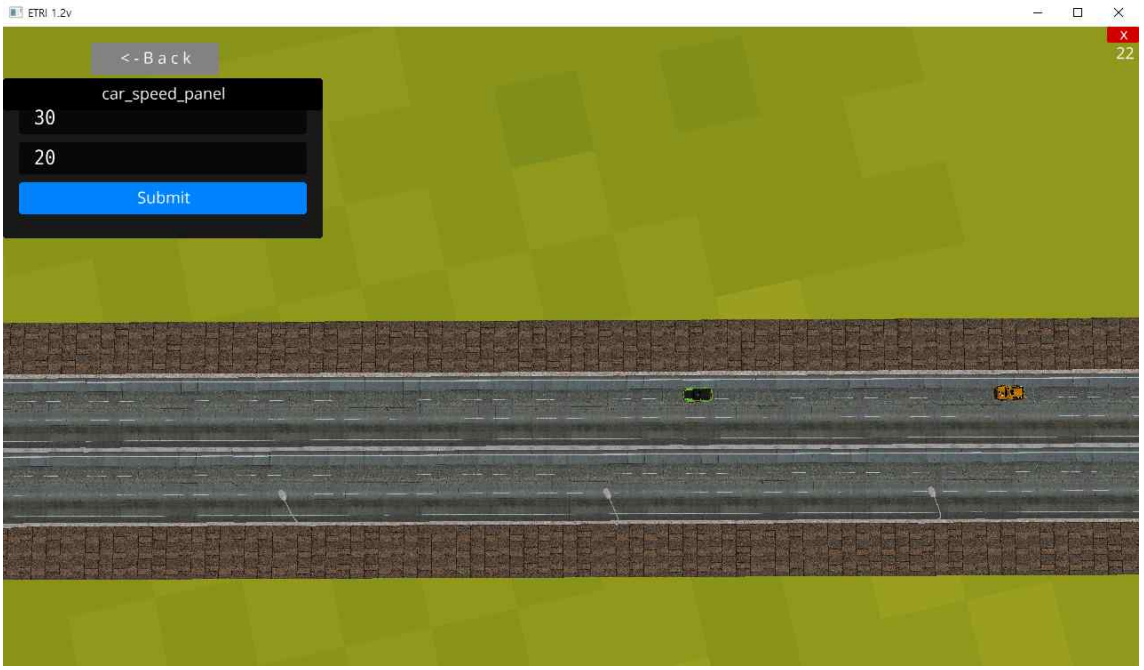
3) 시작된 시뮬레이터 화면



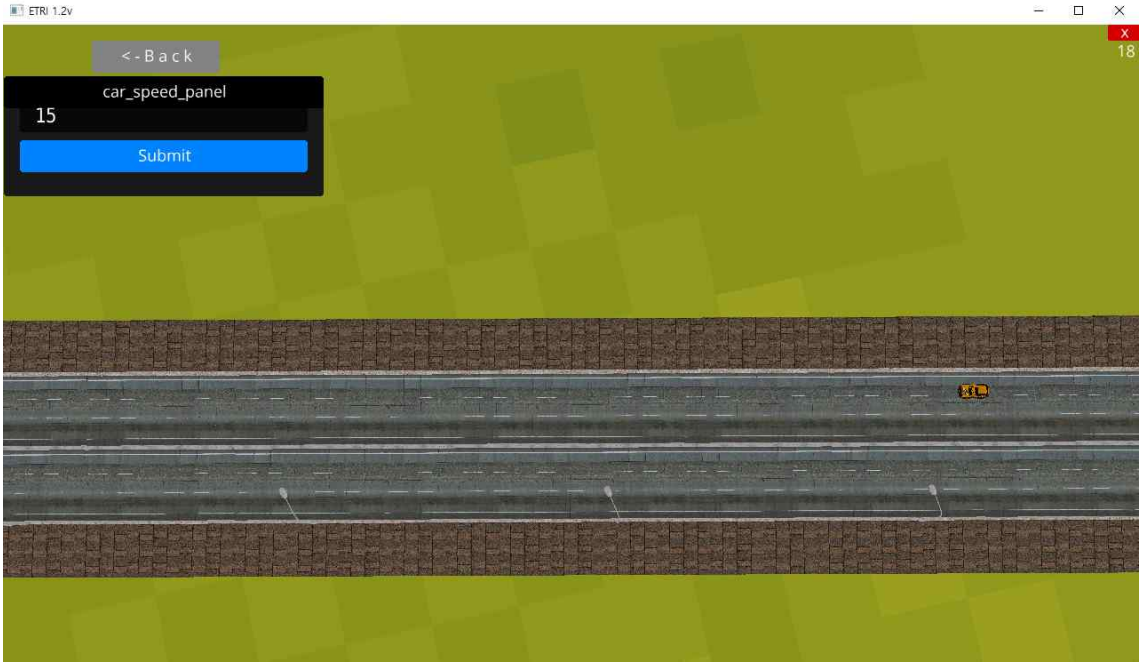
4) 각 case 별 화면

- case A

- case B


ETRI 1.2v
<-Back
car_speed_panel
30
20
Submit
22

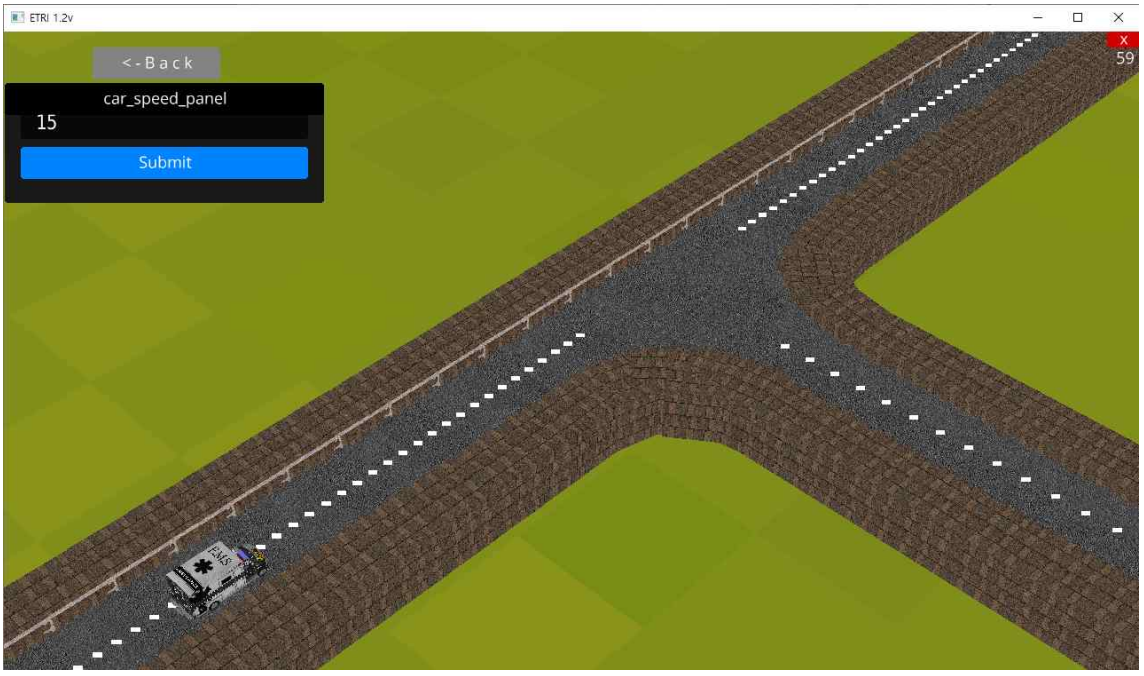
- case C


ETRI 1.2v
<-Back
car_speed_panel
15
Submit
18

- case D


ETRI 1.2v
<-Back
car_speed_panel
15
Submit
22

- case E


ETRI 1.2v
<-Back
car_speed_panel
15
Submit
59