GRAIC 2026 Installation Guide (ECE484 Version)

Yan Miao

GRAIC 2026 Overview:

You will be provided with all Ground Truth Perception data(E.g waypoints, lane information, obstacle information). The task is to focus on planning + control, so that your controller can navigate the vehicle through different scenarios(E.g. car cutting in lane) in different maps. Your controller will be scored on how fast you can finish one lap.

Demo

Below is Yan’s Simple Baseline Controller’s Demo Video on the Shanghai Track:
https://drive.google.com/file/d/1u4rj-sxMd6fuTnt_hS9PL_gUbfWdRjyO/view?usp=sharing

Objectives for ECE484 Students

Benchmark: **Beat Yan's naive basic controller** in terms of score(score is calculated as time to finish, so lower score indicates better performance) and # collisions(the lower, the better):

Track	**Score w/o scenarios**	**Score w/ scenarios**	**# Collisions**
t1_triple	45	70	0
t2_triple	74	97	0
t3	82	105	0
t4	57	82	0
shanghai_intl_circuit	122	156	0

Final Submission:
It is recommended to also submit the following:

1. Link to your controller code
    a. Both ground truth and vision controllers
2. Your Score for the two race configurations & compare / analysis with the baseline benchmark:
    a. Tracks with scenarios
    b. Tracks without scenarios.
3. Whether you collide or not in the above two race configurations
4. Special Instructions / steps to run your code if necessary(E.g. Install certain libraries)

Submission Files

agent.py
requirements.txt

- Any python package you used in agent.py has to be included here
- E.g. numpy==1.23.5

Dependencies:

- Carla 0.9.13
- Scenario Runner 0.9.13
- **Python3.7+**

If you are running on the ECEB5072 lab computers, please create a venv with python 3.7. Do to this, run the following commands

1. *python3.7 -m venv <env_name>*
2. *source <env_name>/bin/activate*

Install the Carla 0.9.13 Simulator

(You can find the official installation guide here: https://carla.readthedocs.io/en/latest/start_quickstart/#carla-installation)*

But I'll give you a simpler installation walkthrough:

1. Download Carla 0.9.13 simulator here: https://carla-releases.s3.eu-west-3.amazonaws.com/Linux/CARLA_0.9.13.tar.gz
2. Extract it into a path you favored (In my case: /home/yanmiao2/ws/CARLA_0.9.13)
3. (ECE484 students do not need to execute this step in lab machines)
    a. *sudo apt install libomp5* per this git issue

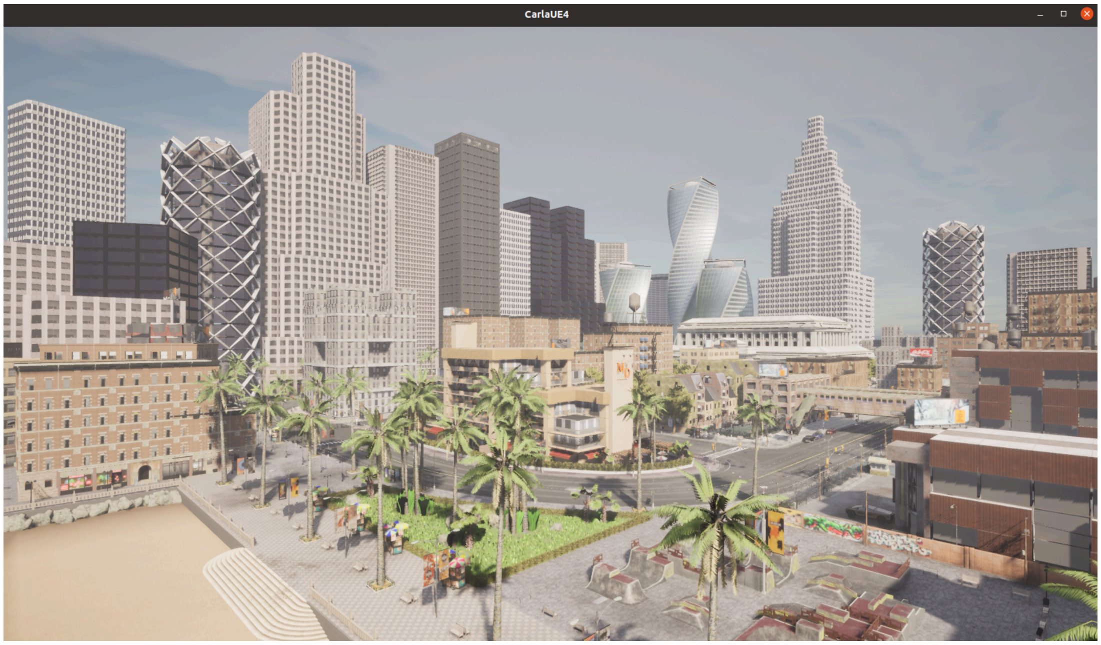
4. Add Carla API to your own python path by appending the below lines into your bashrc file(~/.bashrc) and source the file.
    a. *export PYTHONPATH=$PYTHONPATH:/home/yanmiao2/ws/CARLA_0.9.13/PythonAPI/carla/*

    b. *export PYTHONPATH=$PYTHONPATH:/home/yanmiao2/ws/CARLA_0.9.13/PythonAPI/carla/dist/carla-0.9.13-py3.7-linux-x86_64.egg*

5. Install Individual Packages
    a. Pygame: *python3 -m pip install --user pygame numpy*
    b. Networkx: *python3 -m pip install --upgrade networkx*

If you finish this section correct, then the below 2 steps should get you open the simulator correctly:

a. *cd /home/yanmiao2/ws/CARLA_0.9.13/*
b. *./CarlaUE4.sh*



Install GRAIC Customized Maps for Carla 0.9.13

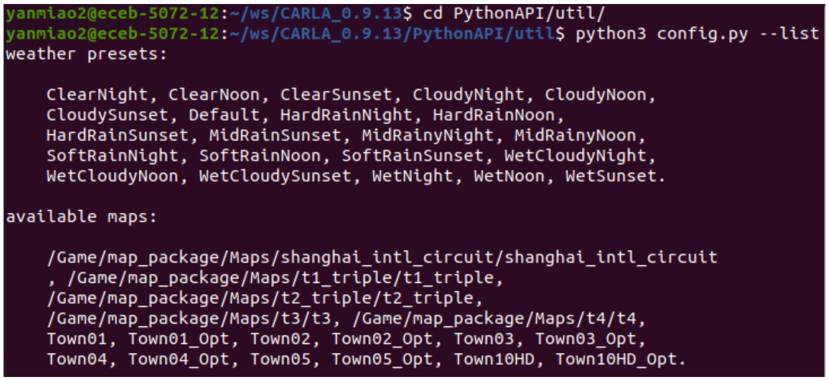
(You can find the official installation guide here: https://carla.readthedocs.io/en/0.9.13/tuto_M_add_map_package/)*

But I'll give you a simpler installation walkthrough:

1. Download the 5 raw maps in my google drive
2. Place the 5 tar.gz file in your corresponding import path (In my case, /home/yanmiao2/ws/CARLA_0.9.13/Import)
3. Import the map using the below 2 commands
    a. *cd /home/yanmiao2/ws/CARLA_0.9.13*
    b. *./ImportAssets.sh*

If you finish this section correct, then the below 4 steps should get you open the simulator and switch to any of the 5 maps:

a. *cd /home/yanmiao2/ws/CARLA_0.9.13/*
b. *./CarlaUE4.sh*
c. (new terminal) *cd /home/yanmiao2/ws/CARLA_0.9.13/PythonAPI/util*
d. *python3 config.py --list*
    i. You should now see 5 maps' name(Shanghai, t1, t2, t3, t4) you just imported



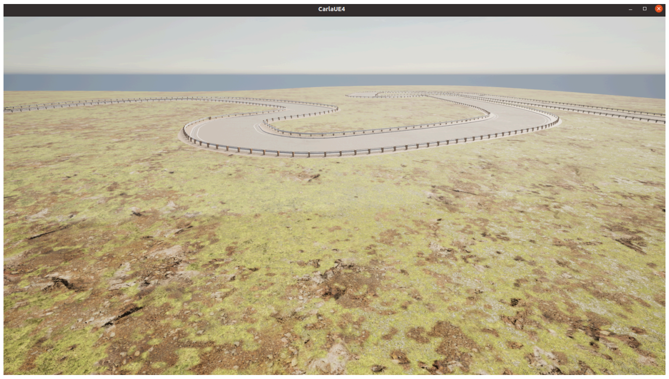
```
yanmiao2@eceb-5072-12:~/ws/CARLA_0.9.13$ cd PythonAPI/util/
yanmiao2@eceb-5072-12:~/ws/CARLA_0.9.13/PythonAPI/util$ python3 config.py --list
weather presets:

    ClearNight, ClearNoon, ClearSunset, CloudyNight, CloudyNoon,
    CloudySunset, Default, HardRainNight, HardRainNoon,
    HardRainSunset, MidRainSunset, MidRainyNight, MidRainyNoon,
    SoftRainNight, SoftRainNoon, SoftRainSunset, WetCloudyNight,
    WetCloudyNoon, WetCloudySunset, WetNight, WetNoon, WetSunset.

available maps:

    /Game/map_package/Maps/shanghai_intl_circuit/shanghai_intl_circuit
    , /Game/map_package/Maps/t1_triple/t1_triple,
    /Game/map_package/Maps/t2_triple/t2_triple,
    /Game/map_package/Maps/t3/t3, /Game/map_package/Maps/t4/t4,
    Town01, Town01_Opt, Town02, Town02_Opt, Town03, Town03_Opt,
    Town04, Town04_Opt, Town05, Town05_Opt, Town10HD, Town10HD_Opt.
```

e. Now you can switch to any of the new map using this command
    i. *python3 config.py -m /Game/map_package/Maps/shanghai_intl_circuit/shanghai_intl_circuit*
    ii. Replace the above shanghai with t1/t2/t3/t4, then you can switch to different GRAIC maps



Install GRAIC Customized Scenario Runner

(We are using a customized Scenario Runner, so please do NOT pull from the official SR repo, but USE our GRAIC repo)*

1. Download the CUSTOMIZED 0.9.13 Scenario Runner from GRAIC public repo https://github.com/PoPGRI/scenario_runner
2. Extract SR into your workstation (in my case: /home/yanmiao2/ws/scenario_runner-0.9.13)
3. Install SR Dependencies by these 2 lines:
    a. *cd /home/yanmiao2/ws/scenario_runner-0.9.13*
    b. *python3 -m pip install -r requirements.txt*
4. Add SR API to your own python path by appending the below line into your bashrc file(~/.bashrc) and source the file.
    a. *export PYTHONPATH=$PYTHONPATH:/home/yanmiao2/ws/scenario_runner-0.9.13*

If you finish this section correctly, then executing the below scripts(E.g. copy the below 2 lines to test.py) in any path with python3(E.g. python3 test.py) should not get you any error message

a. *from scenario_runner import ScenarioRunner*
b. *print("Success")*

Install GRAIC Customized Scripts

1. Download GRAIC infrastructure scripts for 2022 https://github.com/PoPGRI/Race/tree/main
2. Place the folder into your workspace (In my case: /home/yanmiao2/ws/Race)
3. **Implement your controller in agent.py, where all your inputs are the Ground Truth Perception Data, and you just need to implement the controller (and/or planner if that's your design).**

Now you are ready to Launch the whole GRAIC:

1. (Terminal 1) Launch Simulator -- check command above
2. (Terminal 2) *python3 wrapper.py* (in the ws/Race folder)
    i. If you want to change to a different map, just modify **line 6 in wrapper.py**.
        1. 5 maps (shanghai_intl_circuit, t1_triple, t2_triple, t3, t4) have been made provided
    ii. If you would like to test your controller without the scenarios, comment out **line 10 in wrapper.py**.
    iii. You may use **pkill -9 -f python3** command to kill python if you car dies and the scenario is kept running

Note:

1. **Only File to Modify is agent.py, we will overwrite other files when grading:**
   Agent.py -- Please check the detailed comments there
2. **Carla Python API**:
   You might want to become best friends with this buddy:
   https://carla.readthedocs.io/en/0.9.13/python_api/
3. **Model + Track**:
   The code we release to you only runs Tesla Model 3 on the Shanghai Track, but during testing time we may use different models + different tracks.
4. **Scoring & Collision**
   Currently the score of your controller for each race is defined as time to finish a lap on each race. We will first rank controllers by number of collisions; if the number of collisions are the same, we will rank you score(time to finish)

   And to make life easier for you guys, your racing score will not be directly penalized by collisions, however note that:
   - Collision might lead the car into “interesting” states(E.g. Flipping upside down), as defined by the carla’s physics
   - Collision might break your controller, if your controller is using some neural network or caching previous history states (due to significant discontinuities)

Vision Method

Overview:
You will need to finish tracks using vision based(images) only.

Input: Image (720*1280*3)
- Height: 720
- Width: 1280
- Channels: RGB
- Velocity of vehicle

Output: CarlaVehicleControl

All the files are included in the **EC/ folder (https://github.com/PoPGRI/Race/tree/main/EC)** .
To run, simply run *python3 wrapper.py* (**in the EC/ folder**)