# Software Installation

## Requirements

- Windows 10 (this may work on Ubuntu as well, but I did not run my software on Ubuntu, since I do not have an Ubuntu machine with a GPU)
- A GPU supported by Carla (most modern GPUs are supported)
- Carla software version 0.9.11
- Python 3.7 (the only version of Python supported by Carla's Python API library on Windows as of Carla version 0.9.11)
- OpenCV module for Python

## Installation

- Install Carla  0.9.11. Following the instructions here:
    - Download from here: https://carla-releases.s3.eu-west-3.amazonaws.com/Windows/CARLA_0.9.11.zip
    - https://carla.readthedocs.io/en/0.9.11/start_quickstart/

      Note: Carla 0.9.12 was released on 8/3/21. I did all of my development and testing with Carla version 0.9.11, referenced above.

- Install my software in a directory within PythonAPI directory under Carla root

This is what Carla installation looks like:

- carla ← the root directory of the installation
    - PythonAPI
        - carla ← location of Carla API modules
        - *my-project* ← a directory where my project should be installed

An easy way to achieve this is to go into the "PythonAPI" directory and from there run 'git clone' command like this (in this case, I installed Carla into E:\carla):

E:\carla\PythonAPI> git clone
git@github.com:berratin123/DGMD-S17-submission-Benjamin-Ratin.git

After this, you can cd into the cloned directory and run my script from there, see below.

Note:

When my main.py starts it needs to find an ‘egg’ file with the Carla API. It’s looking for the file in the ‘../carla/dist’ directory. This means that when my software is run, the working directory needs to be a “sibling” directory of ‘carla’. The easiest way to achieve this is to install the software in the sibling directory and run it from there.

In addition, the only ‘egg’ file available in Carla’s installation for Windows is for Python 3.7, which is why all Carla tools need to run under Python 3.7. If you have multiple version of Python installed you can run like this ‘py -3.7 *script*.py’

# Running the Software

## Carla Server

Carla has a client-server architecture. The first thing that you need to start is the Carla server, which is achieved by:

- Go into the top directory of Carla installation (E:\carla in my case)
- Run .\CarlaUE4.exe [parameters]

I run my Carla server with the following parameters to make things run faster:

E:\carla> .\CarlaUE4.exe -quality-level=Low -fps=15

## My project

As mentioned above, my software needs to be started from a directory that’s a “sibling” of ‘carla’ within the Python API. Change directory into the cloned directory, and start it like this:

E:\carla\PythonAPI\DGMD-S17-submission-Benjamin-Ratin> **py -3.7 .\main.py**

As stated above, Python version 3.7 is a requirement for Carla Python API on Windows.

**Note**: the client/server communication timeout is set to 10 seconds. Still, when the machine is busy, it sometimes takes more than 10 seconds to load the initial map. If you get an error from main.py that it times out:

- Make sure the Carla server is running
- Try again. The second time it does not need to reload the map and the connection is significantly faster.

If you want the project vehicle to **ignore traffic lights** to make it run faster, you would need to modify the BasicAgent class that is responsible for it within Carla’s library. You can open:

E:\carla\PythonAPI\carla\agents\navigation\basic_agent.py file and add a line

    light_state = False

On line 107 (before “if light_state:” line)

You can stop my software by pressing Ctrl-C.

## Other Carla Tools

There are a number of useful examples inn the examples directory here:

E:\carla\PythonAPI\examples>

The most interesting tool for my project is the one that allows to swans other vehicles and pedestrians within the simulation: You can run it like this:

PS E:\carla\PythonAPI\examples> py -3.7 .\spawn_npc.py -n 50

Where ‘50’ is the number of actors you want to spawn.
Just like with my software, the examples rely on the ‘egg’ file to be in the ‘../carla/dist’ directory, so I run the examples after going into the directory where they are located.