CARLA Learning Task 1: Vehicle Data Extraction and Storage

Important concepts:

CARLA uses a client-server architecture. We are working on the client side, connecting to the server, and receiving the information from the server, and operating the simulation on the server side.

1. Client: https://carla.readthedocs.io/en/latest/core_world/#the-client
2. World: https://carla.readthedocs.io/en/latest/core_world/#the-world
3. Actors: https://carla.readthedocs.io/en/latest/core_actors/

Load the CARLA Server and Client

Server side:

1. Open the Command Prompt window, type or copy and paste the two commands and press Enter:
   ```
   >> cd C:/{Your_Directory}/CarlaSimulator
   >> CarlaUE4.exe
   ```
   Since running CARLA requires strong graphical computing power, we recommend you run CARLA with low graphics quality to speed up the simulation:
   ```
   >> CarlaUE4.exe -quality-level=Low
   ```

2. To switch to other default maps, type the following commands in the same CMD window:
   ```
   >> cd C:/{Your_Directory}/CarlaSimulator/PythonAPI/util
   >> python config.py --map Town05
   ```

3. To add vehicles and pedestrian into the world. In the same CMD window, type the following commands:
   ```
   >> cd C:/{Your_Directory}/CarlaSimulator/PythonAPI/examples
   >> python generate_traffic.py -n 80 -w 40 --safe
   ```

Client side:

1. Open a **new** CMD window, type the following commands to run the example Python client:
   ```
   >> cd C:/{Your_Directory}/PythonAPI/examples
   >> python manual_control.py
   ```

2. To run the customized code, you can type the following commands in the same way:
   ```
   >> cd C:/{Your_Directory}/PythonAPI/examples
   >> python print_actor_info.py
   ```

Procedures to generate and use actors:

Follow the procedures to generate and use the actors. You are free to change the parameters in the functions according to the documentation.

0. Set up the client and server before using the following functions.
   ```
   client = carla.Client('localhost', 2000)
   client.set_timeout(10)
   world = client.get_world()
   ```

1. Choose a blueprint
   a. Initialize carla.BlueprintLibrary class
   ```
   blueprint_library = world.get_blueprint_library()
   ```

   b. Choose a blueprint for the vehicle and the sensor. Class: carla.ActorBlueprint
   ```
   vehicle = blueprint_library.filter('model3')[0]
   vehicle = blueprint_library.find('vehicle.tesla.model3')
   sensor = blueprint_library.find('sensor.camera.rgb')
   ```

   c. Set attributes. The attributes are different for different types of ActorBlueprints.
   ```
   vehicle.set_attribute()
   sensor.set_attribute()
   ```

2. Choose a spawn point. Class: carla.Transform
   a. Manually set a spawn point
   ```
   spawn_point = carla.Transform(carla.Location(x=, y=, z=),
   carla.Rotation(pitch=, yaw=, roll=))
   ```

   b. Or get recommended spawn points
   ```
   spawn_points = world.get_map().get_spawn_points()
   ```

3. Spawn the actor
   a. Spawn the vehicle. Class: carla.Actor
   ```
   actor_vehicle = world.spawn_actor(blueprint=vehicle, transform=spawn_point)
   ```

   b. Spawn the sensor and attach it to the vehicle. Class: carla.Actor
   ```
   actor_sensor = world.spawn_actor(blueprint=sensor, transform=spawn_point,
   attach_to=actor_vehicle, attachment=carla.AttachmentType.Rigid)
   ```

4. Additional actions
   a. Get the list of existed actors
   ```
   actor_list = world.get_actors()
   ```

   b. Get the information of the actor
   ```
   speed = actor_vehicle.get_velocity()
   location = actor_sensor.get_location()
   ```

   c. Set the control mode of the vehicle
   ```
   actor_vehicle.apply_control(carla.VehicleControl(throttle=1.0, steer=0.0))
   actor_vehicle.set_autopilot(True)
   ```

d. Use the data collected by the sensor. The lambda function can be customized.

```
actor_sensor.listen(lambda image: image.save_to_disk('output/%06d.png' %
image.frame))
actor_sensor.listen(lambda data: predefined_function(data))
```

e. Destroy the actors when the program is done.

```
actor_vehicle.destroy()
actor_sensor.destroy()
```

Tasks:

1. Read and run the sample code: `print_actor_info.py, vehicle_camera.py.`
2. Generate multiple vehicles.
3. Extract and store the camera data.
   a. Attach a camera to a vehicle.
   b. Adjust the location and rotation of the camera.
   c. Save the images collected by the camera in a local folder.
4. Extract and store the information (location/speed/acceleration) of all the vehicles.
   a. Store the data in a table with the following columns: `Frame, Timestamp, Vehicle_ID, Location_X, Location_Y, Velocity_X, Velocity_Y, Acceleration_X, Acceleration_Y.`
   b. Save the table in `actor_info.csv`.
5. Submit the following files with the Google Form: https://forms.gle/vrLth6cZQ7uFfxX36
   a. Code to generate actors and sensors.
   b. Code to extract and store the information of the vehicles.
   c. The images collected by the camera.
   d. Vehicle information table `actor_info.csv`.

Helpful links:

- https://carla.readthedocs.io/en/latest/
- https://pythonprogramming.net/control-camera-sensor-self-driving-autonomous-cars-carla-python/