یادگیری ماشین

پروژه اول:

شبیه‌سازی خودرو خودران با استفاده از

CARLA و ذخیره‌سازی داده‌های سنسورها

دانشجو: فروغ کوهی

شماره دانشجویی: ۴۰۳۷۲۳۱۵۱

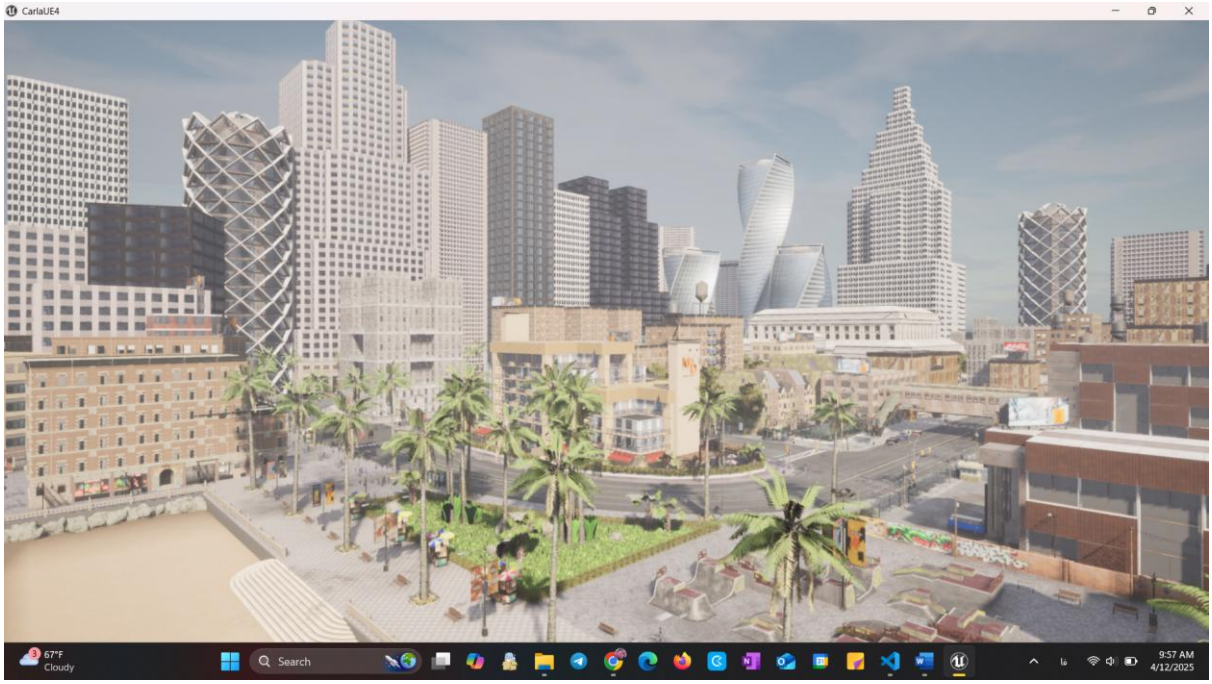
فهرست

۱. مقدمه و توضیح پروژه

این پروژه با هدف شبیه‌سازی یک خودرو خودران در محیط CARLA و جمع‌آوری داده‌های سنسورهای LiDAR، RADAR و Collision طراحی شده است. داده‌های جمع‌آوری شده در پوشه‌های مشخصی ذخیره می‌شوند و می‌توانند برای آموزش مدل‌های یادگیری ماشین یا تحلیل رفتار خودرو مورد استفاده قرار گیرند.

۲. مراحل اجرا

در ابتدا برنامه کارلا را اجرا می‌کنیم و در حالی که برنامه آماده به کار است کد مربوطه را جهت اتصال به سرور کارلا و جمع آوری داده ها را اجرا می‌کنیم. عکسی از محیط شبیه سازی در کارلا ۹.۱۵:



۲.۱. اتصال به سرور CARLA و ایجاد خودرو

با استفاده از کتابخانه carla، یک کلاینت به سرور CARLA روی localhost و پورت ۲۰۰۰متصل شده و محدودیت زمانی ۱۰ ثانیه‌ای برای اتصال تعیین شده است. یک خودروی Tesla Model ۳ از کتابخانه بلوپرینت‌ها انتخاب شده و در یکی از نقاط از پیش تعیین شده (Spawn Points) نقشه قرار می‌گیرد.

```python
client = carla.Client("localhost", 2000)
client.set_timeout(10.0)
world = client.get_world()
blueprint_library = world.get_blueprint_library()

vehicle_bp = blueprint_library.filter("vehicle.tesla.model3")[0]
spawn_point = world.get_map().get_spawn_points()[0]
vehicle = world.spawn_actor(vehicle_bp, spawn_point)
```
Python

۲.۲. افزودن سنسورها

❖ سنسور LiDAR

- نوع: sensor.lidar.ray_cast
- محدوده تشخیص: **۵۰ متر**
- موقعیت نصب: مرکز خودرو با ارتفاع ۲ متر((Location(۰, ۰, ۲)
- داده‌ها به فرمت ply. ذخیره می‌شوند.

```python
# LiDAR sensor
lidar_bp = blueprint_library.find('sensor.lidar.ray_cast')
lidar_bp.set_attribute('range', '50')
lidar_location = carla.Location(0, 0, 2)
lidar_rotation = carla.Rotation(0, 0, 0)
lidar_transform = carla.Transform(lidar_location, lidar_rotation)
lidar = world.spawn_actor(lidar_bp, lidar_transform, attach_to=vehicle)
```
Python

❖ سنسور RADAR

- نوع: sensor.other.radar
- موقعیت نصب: جلوتر از مرکز خودرو((Location(۰٫۲, ۰, ۱)
- داده‌ها به فرمت باینری (radar.) ذخیره می‌شوند.

```python
# RADAR sensor
radar_bp = blueprint_library.find('sensor.other.radar')
radar_location = carla.Location(0.2, 0, 1)
radar_rotation = carla.Rotation(0, 0, 0)
radar_transform = carla.Transform(radar_location, radar_rotation)
radar = world.spawn_actor(radar_bp, radar_transform, attach_to=vehicle)
```
Python

❖ **سنسور تصادف(Collision)**

- نوع: sensor.other.collision
- بدون نیاز به تنظیم موقعیت خاص، به خودرو متصل می‌شود.
- اطلاعات تصادف در فایل‌های متنی ذخیره می‌شود.

```
# Collision sensor
collision_bp = blueprint_library.find('sensor.other.collision')
collision_sensor = world.spawn_actor(collision_bp, carla.Transform(), attach_to=vehicle)
```

۲.۳. ذخیره سازی داده‌ها

سه پوشه خروجی output/lidar، output/radar، output/collision ایجاد شده و برای هر سنسور، یک تابع callbackتعریف شده است که داده‌ها را براساس شماره فریم (Frame) نام‌گذاری و ذخیره می‌کند.

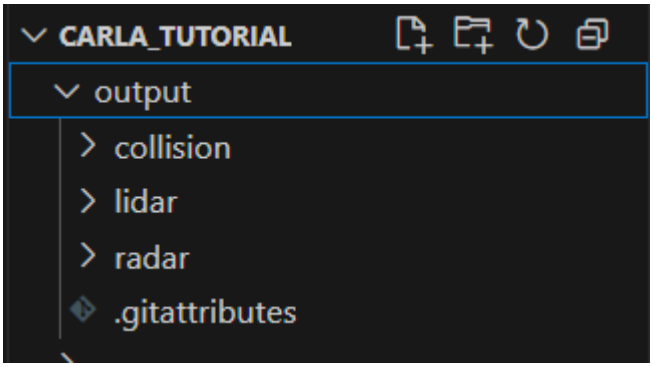
```
# Saving data
def save_lidar_data(data):
    filename = f"output/lidar/{data.frame}.ply"
    data.save_to_disk(filename)

def save_radar_data(data):
    filename = f"output/radar/{data.frame}.radar"
    with open(filename, 'wb') as f:
        f.write(data.raw_data)

def save_collision_data(event):
    filename = f"output/collision/{event.frame}.txt"
    with open(filename, 'w') as f:
        f.write(str(event))
```

```
# Create Folders
os.makedirs('output/lidar', exist_ok=True)
os.makedirs('output/radar', exist_ok=True)
os.makedirs('output/collision', exist_ok=True)
```

نتیجه اجرای کد:



۲.۴. اجرای شبیه‌سازی

سنسورها به مدت **۳۰ ثانیه** فعال می‌شوند، داده‌ها را جمع‌آوری می‌کنند و پس از اتمام زمان، سنسورها و خودرو از محیط شبیه‌سازی حذف می‌شوند.

```python
lidar.listen(lambda data: save_lidar_data(data))
radar.listen(lambda data: save_radar_data(data))
collision_sensor.listen(lambda event: save_collision_data(event))

time.sleep(30)
```

Python

```python
# Cleaning
lidar.stop()
radar.stop()
collision_sensor.stop()
vehicle.destroy()
lidar.destroy()
radar.destroy()
collision_sensor.destroy()
```

Python