目录

LGSVL

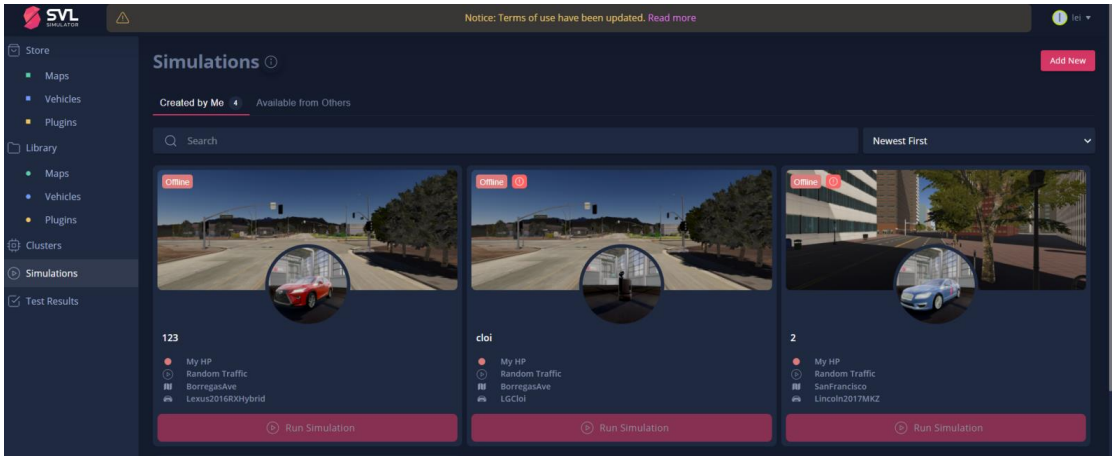
系统整体叙述

开源软件，免费使用。 LGSVL 是基于游戏引擎 Unity 开发的一款主要用于自动驾驶开发和测试，支持包括仿真环 境、传感器以及通讯内容的自定义。LGSVL 由模拟软件、软件工具、支持定制用例的内容和插件生态系统以及支持大规模模拟和 场景测试的云环境组成，允许开发人员进行调试、执行模块化测试和执行集成测试。

各模块功能性需求分析

界面分析

SVL Web 界面



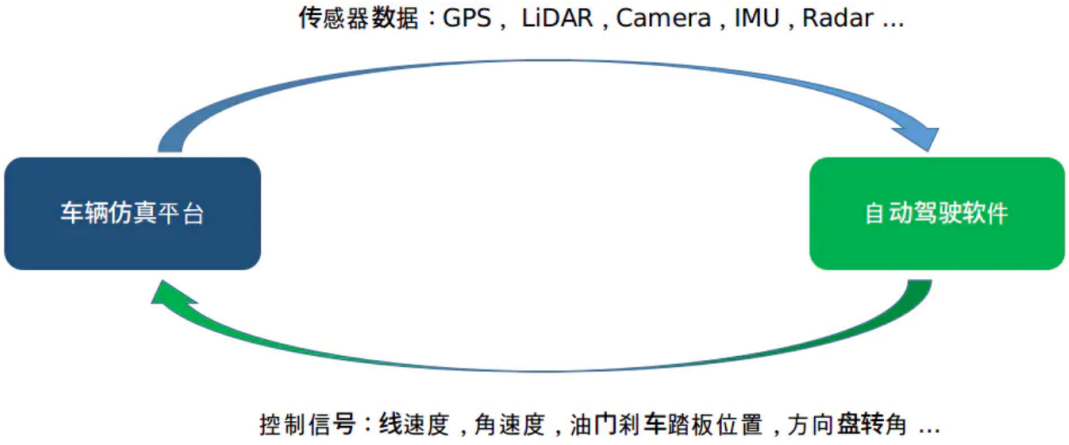
SVL 软件模拟界面

闭环仿真测试

支持与其他自动驾驶软件进行闭环仿真测试

- Apollo
- Autoware

这里所谓的“闭环”，就是测试平台中的仿真车辆将传感器检测数据发送给自动驾驶软件，自动驾驶软件经过一系列感知、定位、决策、控制环节，将最终的控制信号传递回仿真车辆，驱动汽车按照期望的路线行驶。来自车辆传感器的检测数据和来自自动驾驶软件的控制信号构成了一个闭合的信息流。闭环测试框架如下图所示。



API 调用

可以通过 Python API 来操纵场景中的环境布置、车辆移动、传感器配置、天气和时间状态等。

配置管理

- 4GHz四核CPU
- Nvidia GTX 1080 (8GB memory)
- Windows 10 64 bit
- 也可在Linux系统运行，效率可能稍低

与 Apollo 或者 Autoware 在一台主机上运行:

- 至少10GB以上的GPU

Apollo 或者 Autoware 在不同系统上运行:

- 两个系统分别安装LGSVL和自动驾驶软件
- 两个系统网络连接状态良好

其他功能

- 端到端模拟
- 实时高性能模拟
- 逼真的环境模拟
- 多车模拟
- 可拓展性和可定制性
- 程序化道路网络生成和环境创建工具
- 高清地图导入、导出和注释工具
- 具有丰富的模型库：
- 地图
- 车辆（轿车、赛车、机器人）
- 插件（激光雷达、GPS、摄像机等）

Carla

系统整体叙述

CARLA 是一个开源的自动驾驶模拟器。它是从零开始构建的，作为一个模块化和灵活的 API 来解决一系列涉及到自动驾驶问题的任务。 Carla 模拟器由可扩展的客户端-服务器架构组成。服务器负责与模拟本身相关的所有事情： 传感器渲染、物理计算、世界状态及其参与者的更新等等。客户端由一组客户端模块组成， 这些模块控制场景中行为者的逻辑并设置世界条件。这是通过利用 CARLA API（在 Python 或 C++ 中）来实现的。

各模块功能性需求分析

Traffic manager

Carla 专门构造了 Traffic Manager 这个模块来模拟类似现实世 界负责的交通环境。通过这个模块，用户可以定义不同车型、不同行为模式、不同 速度的车辆在路上。

Sensors

Carla 里面有各种各样模拟真实世界的传感器模型，包括相机、激光雷 达、声波雷达、IMU、GNSS 等等。为了让仿真更接近真实世界，它里面的相机拍 出的照片甚至还有畸变和动态模糊效果。用户一般将这些 Sensor attach 到不同的 车辆上来收集各种数据。

Open assets

CARLA 为城市环境提供了不同的地图，可以控制天气条件，并提 供一个包含大量要使用的参与者的蓝图库。这些元素可以自定义，并且可以按照简 单的指南生成新元素。

World and client

客户端和世界是 CARLA 的两个基本要素。 客户端是 CARLA 架构中的主要元素之一。它们连接到服务器、检索信息和命令更改。这是 通过脚本完成的。客户端对象的主要目的是获取或改变世界，并应用命令。 世界主要有如下几个部分 Actors in the simulation and the spectator Blueprint library Map Simulation settings Snapshots Weather and light manager ，可以通 过编程来生成行为者，自定义天气状况，改变地图上的灯光以及访问一些用于模拟的高级配置等。

Actors and blueprints

行为者不仅包括车辆和步行者，还包括传感器、交通标志、交通信号灯和观众。行为者是在模拟中扮演角色的任何东西。蓝图允许用户将新行为者顺利地合并到模拟中。它们是带有动画和一系列属性的已经制作好的模型。其中一些是可以修改的，而另一些则不是。这些属性包括车辆颜色、激光雷达传感器中的通道数量、步行者的速度等等。 官方文档中有简单介绍对应传感器、观众、交通标志和信号灯、车辆以及行人的生成和处 理。

Maps and navigation

地图主要是代表模拟世界的物体，城镇。有八张地图。地图包括城镇的 3D 模型及其道路定义。地图的道路定义基于 OpenDRIVE 文件，这是一种标准化的、带注释的道路定义格式。

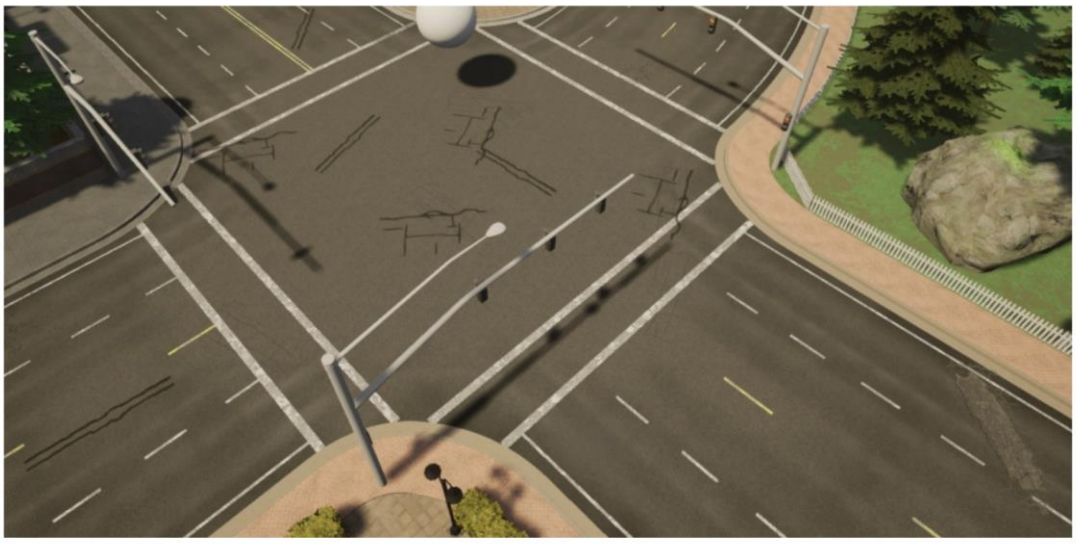
OpenDRIVE 标准 1.4 规定了道路、车道、路口等。 同样，官方文档中也提供了简单的实例。针对地图的修改，车道、路口、环境对象的定义等。

Sensors and data

传感器等待某个事件发生，然后从模拟中收集数据。它们需要一个定义如何管理数据的函数。根据具体情况，传感器检索不同类型的传感器数据。 官方文档给出了参考资料。其中 Python API 定义了许多对于世界中的元素改变、控制的方 法。包括行为者的控制、道路控制、灯光管理等。官方文档给出简单的例子关于设置传感器 的属性，详细的属性设置可以在 sensors reference 中查看。

自定义地图

官方文档中有一部分是关于地图的自定义。其中有提到对于交通灯和交通标志的自定义，建筑的自定义，天气的自定义以及道路的自定义。Carla 允许使用不同的材料、纹理以及添加网格和贴纸以获得你想要的外观。如下图所示:



AirSim

系统整体叙述

AirSim 是微软研究院开源的一个建立在虚幻引擎（Unreal Engine）上的无人机以及自动驾驶模拟研究项目。 AirSim 实现为一个虚幻引擎的插件，它充分利用了虚幻引擎在打造高还原的逼真虚拟环境的能力，可以模拟阴影、反射等现实世界中的环境，以及虚拟环境可以方便产生大量标注数据的能力，同时提供了简单方便的接口，可以让无人机和自动驾驶的算法接入进行大量的训练。 AirSim 的主要目标是作为 AI 研究的平台，以测试深度学习、计算机视觉和自主车辆的端到端的强化学习算法。最新的 AirSim 也提供了 Unity 引擎的版本，添加 了激光雷达的支持。

各功能模块需求分析

模拟完成车辆建模测试

AirSim 包含车辆模拟、城市道路场景，还提供可以简化 编程的 API 以及即插即用的代码，可以很快上手并用于自己的自动驾驶项目。

快速构建丰富场景

AirSim 提供了详细的 3D 城市街景，以及包括交通信号灯、公 园、湖泊、工地等丰富的场景。开发者可以在各种不同的场景下测试他们的系统， 无论是在市中心，还是在城乡道路、郊野和工业区。开发者还可以利用 AirSim 的 拓展性添加新的传感器、车辆，甚至使用不同的物理引擎。

一站式 AI 研究平台

AirSim 提供包括 C++和 Python 等多语言的 API 接口，使用者 可以十分容易地将 AirSim 和众多机器学习工具共同使用。例如，开发者可以使用 微软认知工具包(CNTK)和 AirSim 进行深度增强学习。同时，我们也看到使用新型 的对数据量需求很大的机器学习算法进行训练时，在基于 Microsoft Azure 的 AirSim 下运行多个实例具有很大的潜力。

核心 API

- 图像类**API**：获取各种类型的图像、控制云台等
- 控制仿真运行：可以控制仿真暂停或继续
- 碰撞**API**：获取碰撞信息，包括碰撞次数、位置、表面信息、渗透深度等
- 环境天气**API**：控制环境中的天气：下雨、雪、灰尘、雾气、道路湿滑程度等
- 环境风**API**：控制环境中的风速和风向等
- 雷达**API**：添加和控制环境中的雷达传感器
- 汽车**API**：控制汽车的行动及各种运动学参数

PanoSim

系统整体叙述

PanoSim 是一款面向汽车自动驾驶技术与产品研发的一体化仿真与测试平台，集高精度车

辆 动力学模型、高逼真汽车行驶环境与交通模型、车载环境传感器模型和丰富的测试场景于一 体，支持与 Matlab/Simulink 联合无缝仿真，提供包括离线仿真、实时硬件在环仿真（MIL/SIL/HIL/VIL)和驾驶模拟器等在内的一体化解决方案；支持包括 ADAS 和自动驾驶环境感知、决策规划与控制执行等在内的算法研发与测试。PanoSim 具有很强的开放性与拓展性，支持定制化开发，操作简便友好。 PanoSim 软件中的驾驶环境为仿真系统提供场景要素，其包括三部分：静态场景（道路、建 筑物、信号灯、障碍物等）、动态交通（随机车、干扰车、干扰行人）、天气环境（气象、 光照）,提供车辆行驶所必须的场所。

各模块功能性需求分析:

场景编辑器

主要用于创建或编辑仿真实验所需三维虚拟场景 和环境（包括道路和道路网络结构、道路路面和车道信息、地形、周边建筑和交通 设施等）还支持对车道线、道路路面纹理、附着属性、周边环境、地形高度等信息 的灵活定制和同步预览，兼容多种格式复杂的路面特征信息，自动生成所见即所得 的逼真三维虚拟实验场景。

车辆编辑器

主要用于创建或编辑仿真实验所用车辆三维外 形、车辆动力学和汽车结构参数。
实验设置运行

实验设置运行

承担 PanoSim 核心操作枢纽角色，涵盖实验分组管理、 选择实验场景、设置环境条件（例如天气等）、摆放实验车辆、安装车载传感（例 如像机、雷达等）、部署交通元素（例如路障、交通标志等）、设置交通模型（例 如干扰交通、随机交通等）、设置实验参数、设置驾驶参数、实验运行跟踪（与 Matlab/Simulink 无缝集成）等多项子模块。

实验数据分析

主要用于对实验过程 数据进行后处理和分析报告（例如多维度图表数据分析、实验动画的回放等）

仿真关键步骤

场景构建（Scenario Building）：通过场景编辑工具，完成测试所需要的场景。 测试场景的多样性、覆盖性、典型性等都能够影响到测试结果的准确性，从而影响 自动驾驶的安全与质量
传感器系统建模（Adding model sensor systems）：PanoSim 提供多种传感器 模型，可以灵

活设置传感器参数等
添加控制系统（Adding control systems）：根据车辆状态及传感器信号可在MATLAB/Simulink 中建立自定义反馈控制算法设计
运行仿真实验

CarMaker

系统整体叙述

CarMaker 是德国 IPG 公司旗下的一款专注于乘用车的动力学仿真软件，它包括了精准的车辆 本体模型(发动机、底盘、悬架、传动、转向等)，同时也可以提供包括车辆，驾驶员，道路，交通环境的闭环仿真系统。CarMaker 既可针对 V 开发流程前期进行的模型在环、软件在 环等离线仿真，也可以接入 ECU、子系统总成、网络等做硬件在环测试。 CarMaker 和同为 IPG 公司开发的分别针对多轴车辆和摩托车的动力学仿真软件 TruckMaker 和 MotorcycleMaker 一样，包含了道路三维模型，交通环境模型以及适应不同驾驶环境以及 不同驾驶策略的驾驶员模型。
CarMaker 具有支持高精地图的导入与导出，支持在高性能计算 (HPC) 集群上并行执行大量测试目录，支持在 Docker 容器中运行，具有良好的可移植性和可扩展性等特点。作为平台软件，CarMaker 可以与很多第三方软件进行集成， 如 ADAMS、 AVLCruise、 rFpro 等，可 利用各软件的优势进行联合仿真。同时 CarMaker 配套的硬件，提供了大量的板卡接口，可以方便的与 ECU 或者传感器进行 HIL 测试。

各模块功能性需求分析

IPG Road

可以模拟多车道、 十字路口等多种形式的道路， 并可通过配置 GUI 生 成锥形、 圆柱形等形式的路障。可对道路的几何形状以及路面状况(不平度、 粗糙 度) 进行任意定义。

IPG Traffic

是交通环境模拟工具， 提供丰富的交通对象（车辆、行人、路标、 交通灯、道路施工建筑等）模型。可实现对真实交通环境的仿真。测试车辆可识别 交通对象并由此进行动作触发（如限速标志可触发车辆进行相应的减速动作）。

IPG Driver

先进的、 可自学习的驾驶员模型。可控制在各种行驶工况下的车 辆，实现诸如上坡起步、入库泊车以及甩尾反打方向盘等操作。并能适应车辆的动 力特性（驱动形式、 变速箱类型

等)、道路摩擦系数、风速、交通环境状况，调 整驾驶策略。

与 Simulink 联合仿真

CarMaker 自带与 Simulink 联合仿真的功能，CarMaker for Simulink 即是将 IPG 的车辆动力学 仿真软件 CarMarker 完全集成到 MathWorks 建模和仿真环境 Matlab/Simulink 中。使用 S 函数 以及 MATLAB 和 Simulink API 函数将 CarMaker 功能添加到 Simulink 环境中。CarMaker 与 Simulink 的集成不是松耦合的协同仿真，而是两个应用程序的紧密联系组合，从而形成 了兼具性能和稳定性的仿真环境。

在 Simulink 环境中使用 CarMaker 与使用标准 S 函数模块或内置 Simulink 模块没有什么不 同。CarMaker 模块的连接方式与其他 Simulink 模块的连接方式一样，只需单击几下鼠标 即可将现有 Simulink 模型添加到 CarMaker 车辆模型中。在集成的过程中， CarMaker 功 能也得以保留，而且可以与 Simulink 结合使用。

使用 CarMaker GUI 进行模拟控制和参数调整，以及定义机动和道路配置。IPGControl 可用于数据分析和绘图。 IPGMovie 在三维空间中提供逼真的动画和多体车辆模型渲染，使车辆模型栩栩如生。

CarMaker 的仿真结果可以使用 MATLAB cmread 实用程序访问。此实用程序将任何 CarMaker 仿真结果文件的数据加载到 MATLAB 工作区中。一旦数据在工作区中可用，MATLAB 的所有功能，例如图表绘制和平滑功能，都可以用于后处理和绘制仿真结果。使用 C 接口时需要 MATLAB Coder 和 Simulink Coder。