CARLA传感器资源占用测试

测试环境：

笔记本：

CPU: 13th Gen Intel® Core™ i9-13980HX × 32
GPU: NVIDIA 4070 8G显存
MEM: 64G

CARLA：

版本：源码编译、0.9.14

仿真传感器参数：

cam_front：'cam_front', (1.5, 0.0, 1.6), width=1920, height=1080, fov=90

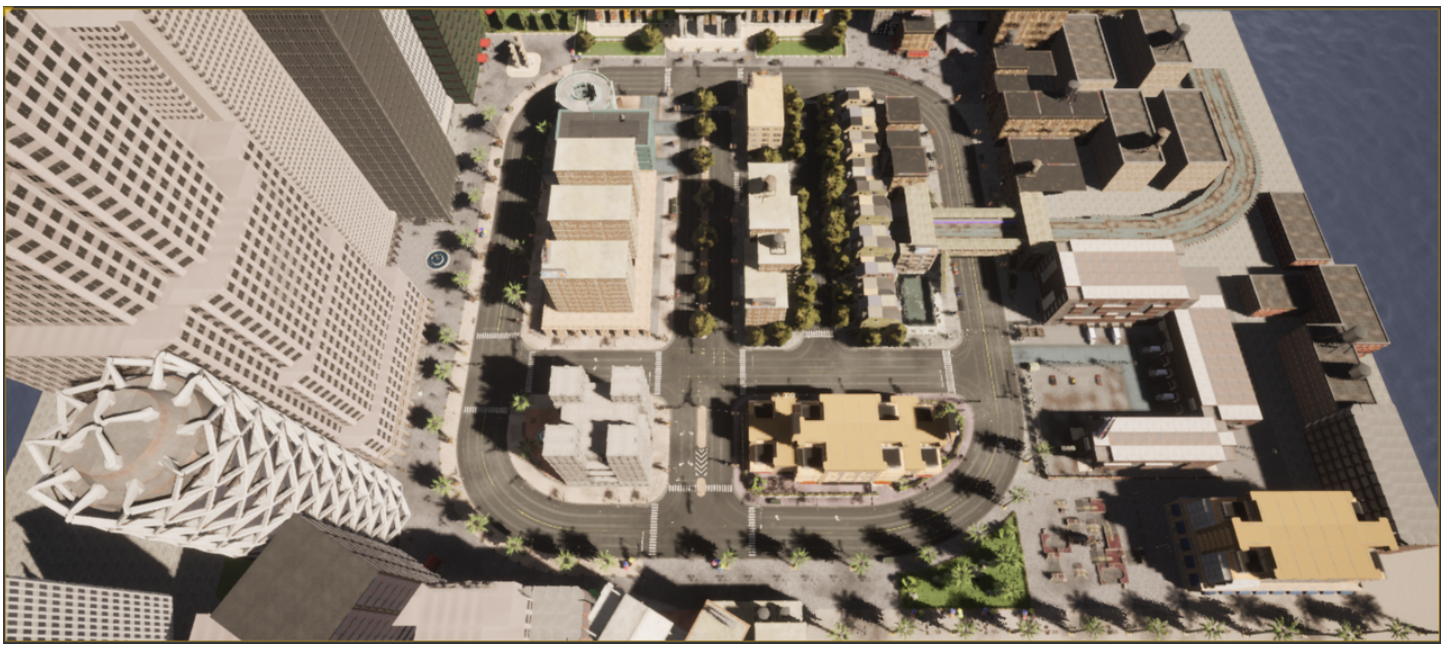
cam_right: 'cam_right', (1.5, 0.5, 1.6), width=1920, height=1080, fov=90

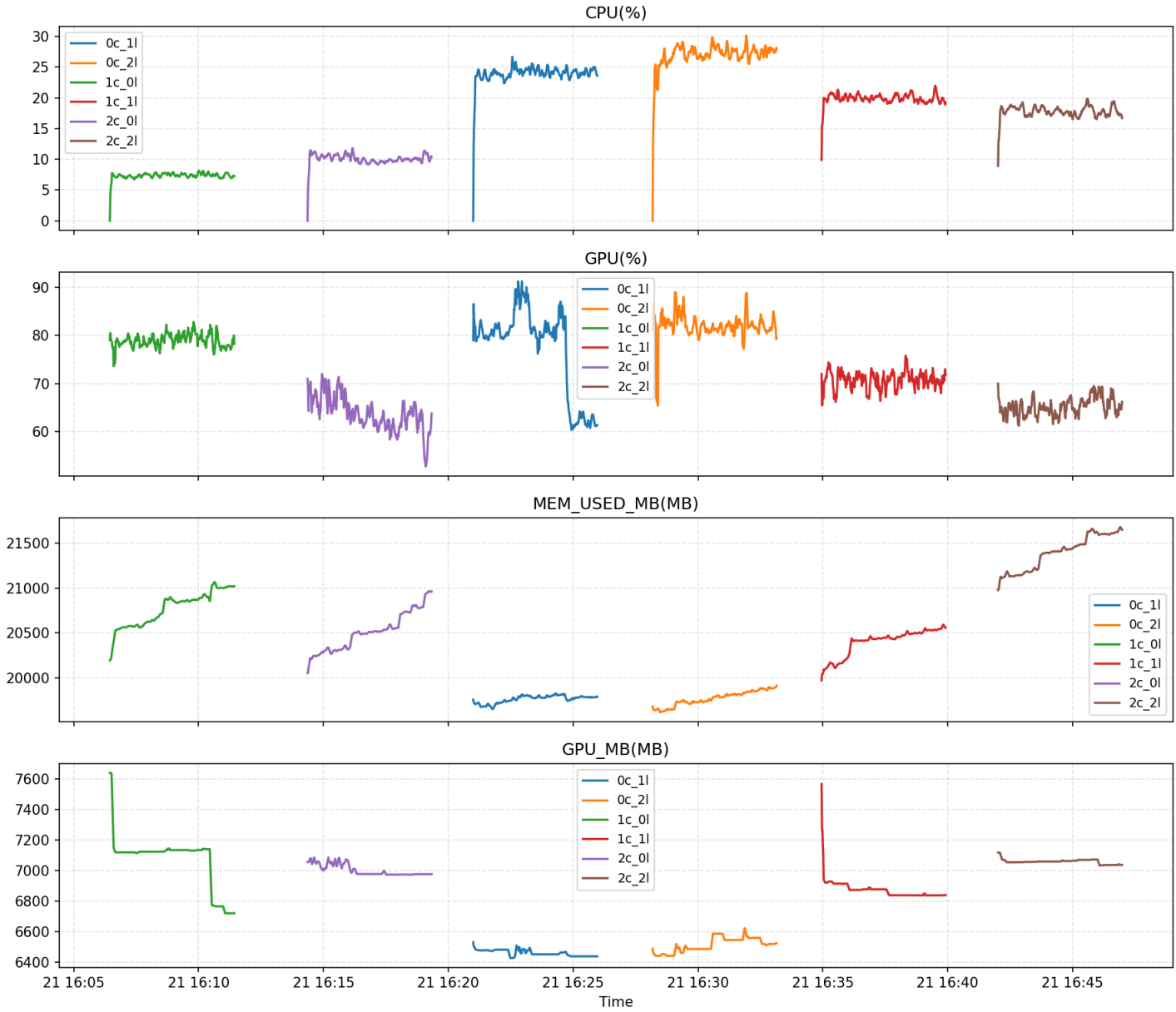
lidar_main: 'lidar_main', (0.0, 0.0, 1.8), channels=32, pps=600000, hz=10.0, range_m=100.0

lidar_aux:'lidar_aux', (0.0, 0.0, 1.9), channels=32, pps=200000, hz=10.0, range_m=80.0

测试方法：

测试脚本连接127.0.0.1:2000的carla服务端，脚本自动加载地图：Town10HD_Opt,生成一辆车，挂在不同数量的传感器，车辆自动在场景中驾驶，性能检测线程每隔1s记录一次数据。开启同步模式，便于稳定 FPS 。用 **外部 tick（定时器）20HZ 驱动** 来推进仿真和传感器更新的。





（其中xc_yl分别代表当前场景使用x个摄像头，y个激光雷达）

上述资源测试有一个反常识错觉即加1路camera反而导致带lidar的测试cpu gpu占用更低了；

代码块

```
# 每轮都 world.tick()，不做任何节拍控制
while time.time() < end_time:
    world.tick()
    frame_counter += 1
```

当只开“1 路激光雷达”时，每个 tick 很轻，1 秒里能推进更多 tick，于是**单位时间内做的工作更多** → CPU/GPU 的**平均利用率更高**。

加了“1 路相机”后，每个 tick 变重（1080p 渲染 + SceneCapture），**每秒能推进的 tick 数变少**，单位时间内做的总工作量反而下降，于是**CPU/GPU 利用率看起来更低**。这并不代表“更省算力”，而

是**被更重的每帧成本拖慢了帧率**。

为了解决上述问题：选择采用**固定墙钟节拍**

让所有组合都**严格以 20 Hz 墙钟节拍**推进（不允许更快），这样单位时间内 tick 次数一致，对比的就是“同样 20 帧/秒时的 CPU/GPU 开销”。

代码块

```
TARGET_HZ = 20.0
STEP = 1.0 / TARGET_HZ
next_t = time.time()
while time.time() < end_time:
    t0 = time.time()
    world.tick()
    # 统一节拍
    next_t += STEP
    time.sleep(max(0.0, next_t - time.time()))
```