恒润科技
HIRAIN TECHNOLOGIES

CARLA 场景制作说明书

北京经纬恒润科技有限公司

恒润科技
HIRAIN TECHNOLOGIES

文件状态： [√] 草稿 [ ] 正式发布 [ ] 正在修改

编制： 陈涵之	签名：	日期：
审核：	签名：	日期：
批准：	签名：	日期：

恒润科技
HIRAIN TECHNOLOGIES

更改历史

版本	更改描述	更改日期	更改人
V1.0	初始版本		

恒润科技
HIRAIN TECHNOLOGIES

目录

CARLA 场景制作说明书

本说明书用于 CARLA 测试场景的搭建。大体上说，该场景的搭建运用到了 RoadRuner 与 UE4 这两款软件，其中，RoadRunner 仅用于道路的搭建，而诸如交通灯、路标等交通信号的配置则是在 UE4 中完成的，除此之外，交通车的运行配置也是在 UE4 完成的。

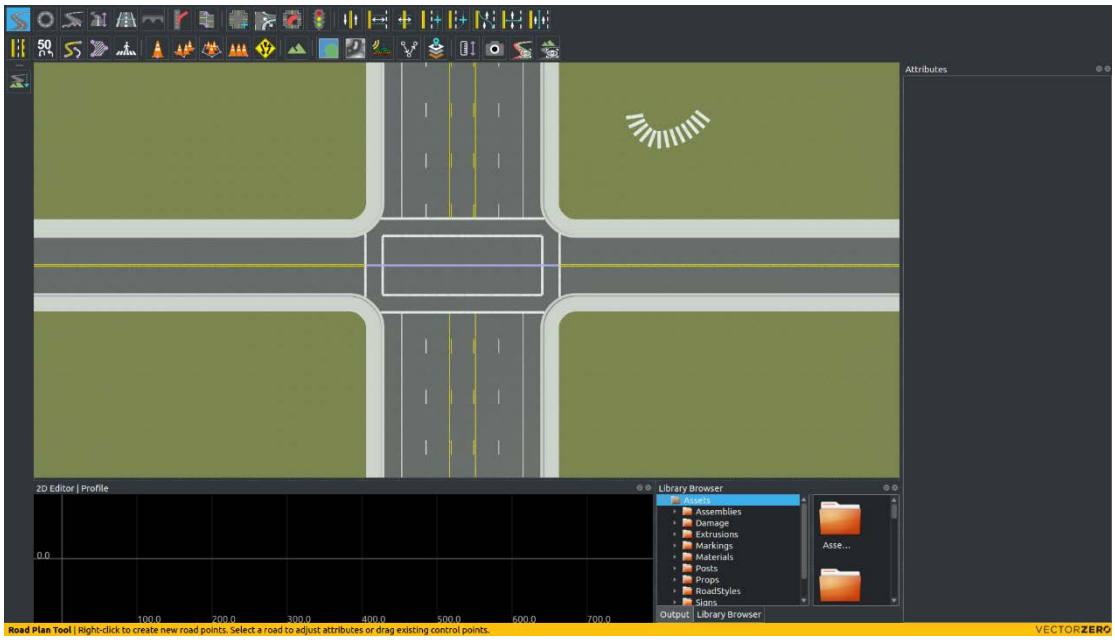
1. 路网搭建 (RoadRunner)

1.1. RoadRunner 的道路搭建

该部分的具体内容可以参见该软件的说明书，本说明书以一个交通十字路口为例。首先，在 RoadRunner 中搭建一个小型十字路口，如下图所示。

1.2. 导出道路信息文件

如果想要将搭建的地图导入 UE4，那么需要 fbx 与 xodr 文件。fbx 类似于一个 3D 模型，能够可视化我们搭建的路网；而 xodr 文件则提供路网的各部分信息。因此，该场景的导出环节如下所示。

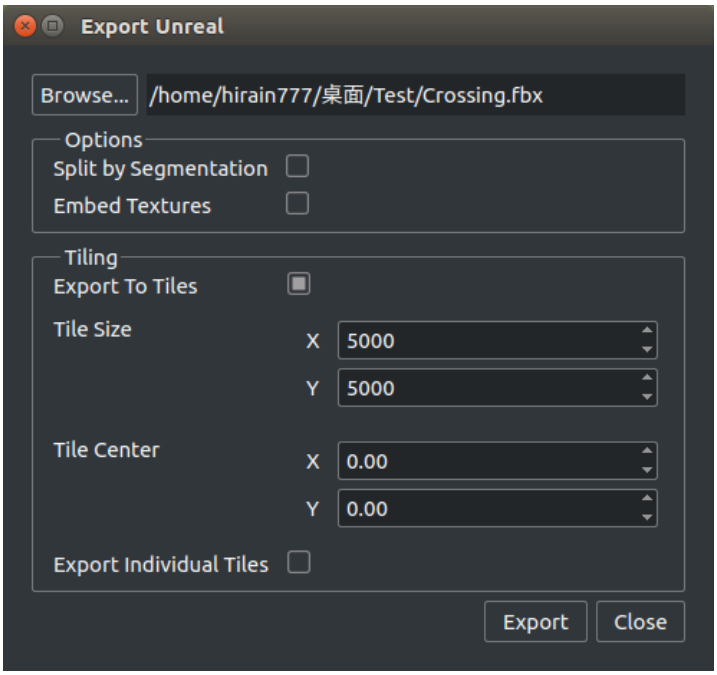


(图 1)

- 导出 fbx 文件

  选中 File 中的 Export，其中有一个 Unreal (.fbx + .xml)选项，将其选中。

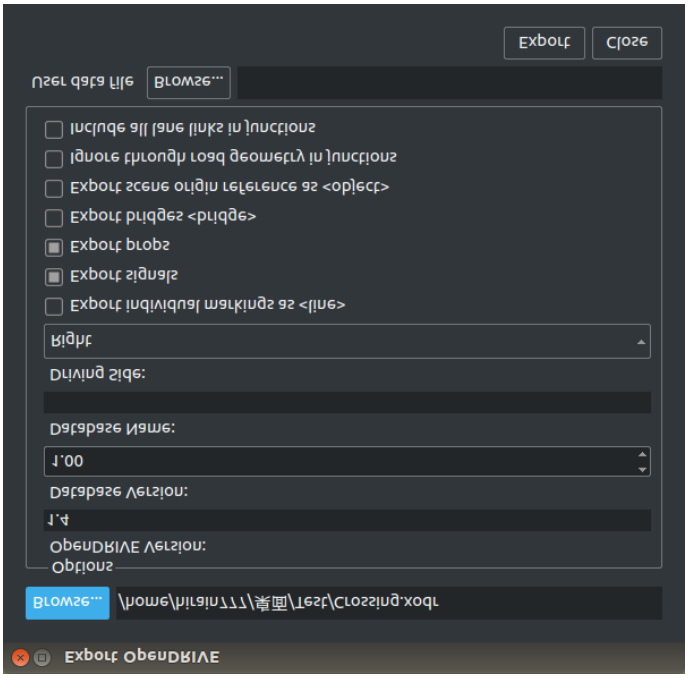
注意：导出时的 Tile Size 要和搭建的路网尺寸相符，尽可能不要过小，否则导入 UE4 的地图是不完整的。



（图 2）

- 导出 xodr 文件

  选中 File 中的 Export，其中有一个 OpenDRIVE (.xodr)选项，将其选中。



（图 3）

1.3. 导入 UE4

在完成 1.2 步骤后，我们获取了两个文件，即.fbx 文件与.xodr 文件，在着这一个步骤中，我们将搭建的道路模型导入 UE4。

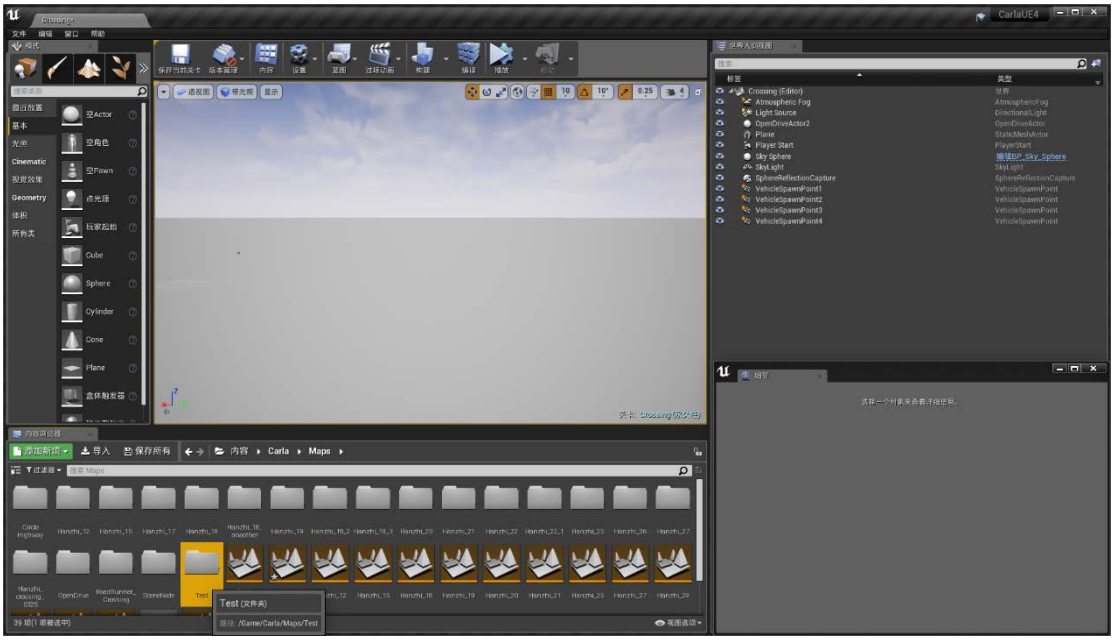
- 复制地图模板

  由于官方已经对光源等进行了设置与限定，因此在搭建地图模板的时候，可以直接复制官方的地图模板 testmaptruck，再在此基础上搭建我们的测试场景，除此之外，我们还需要对复制文件重命名，此处将该生成的场景命名为 Crossing。

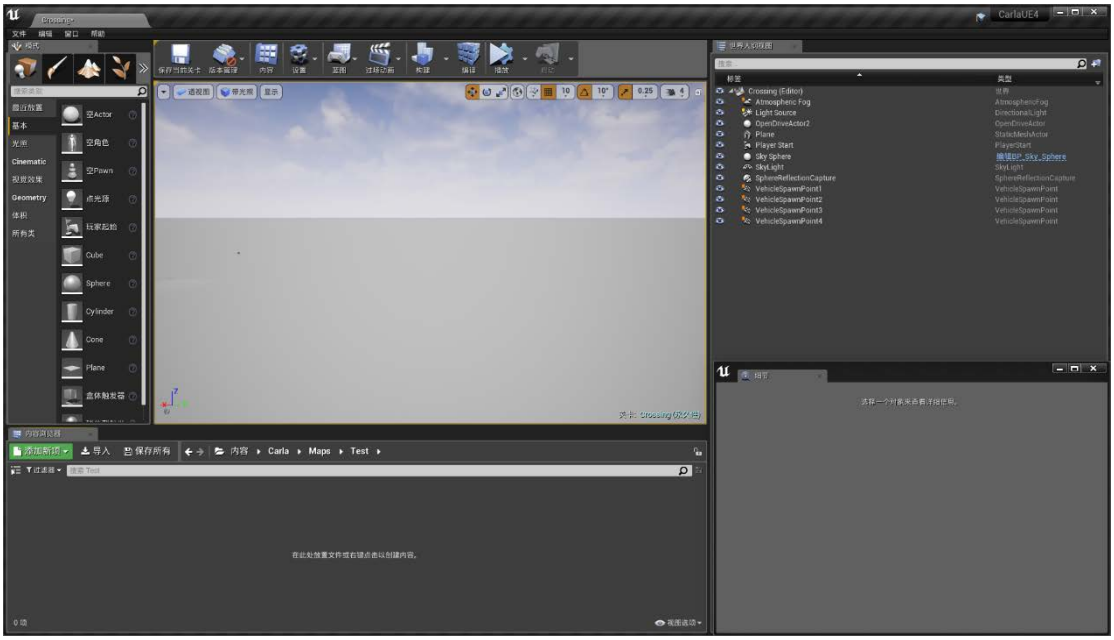
(图 4)

- 导入.fbx 文件

  在我们新生成的场景 Crossing 中，导入刚刚搭建的.fbx 文件，以获取路网模型，建议在此时先新建一个文件夹用于存放导入的材质，打开该文件夹后，再点击导入。

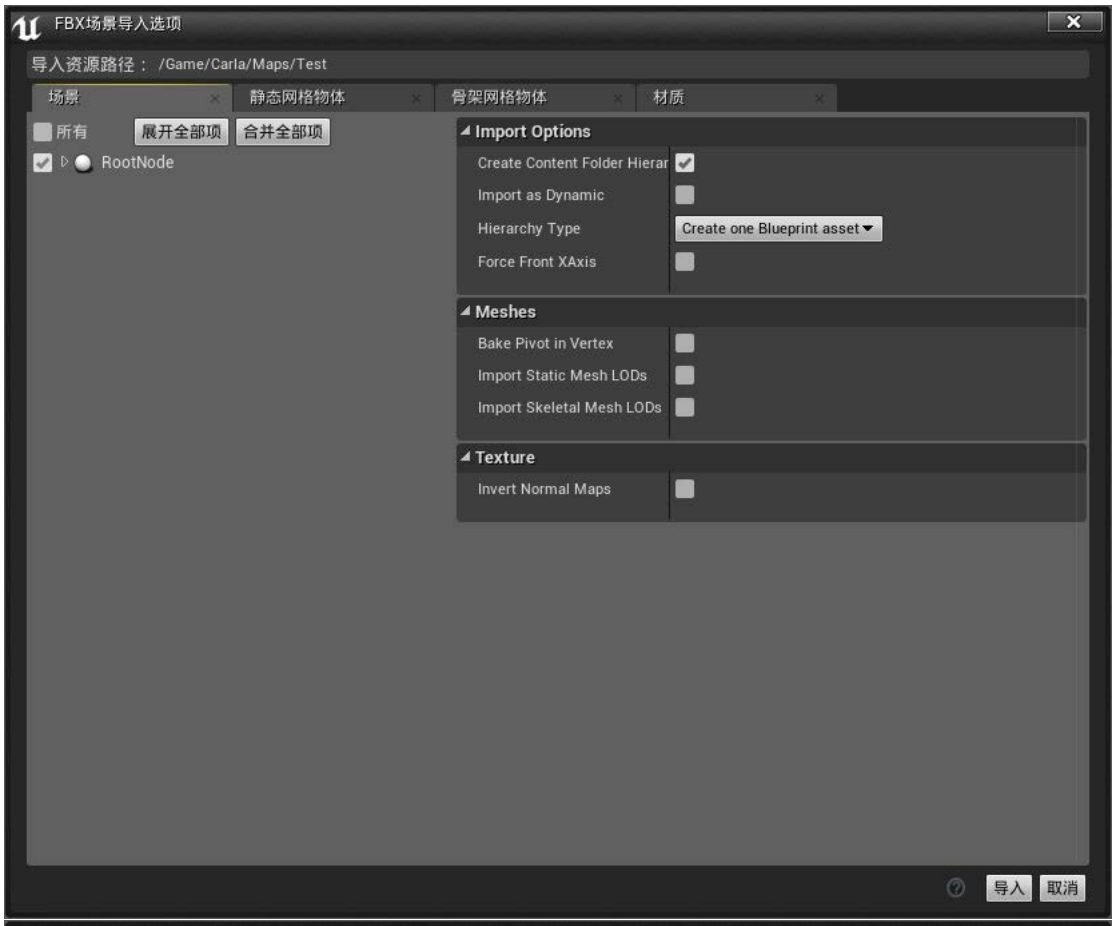


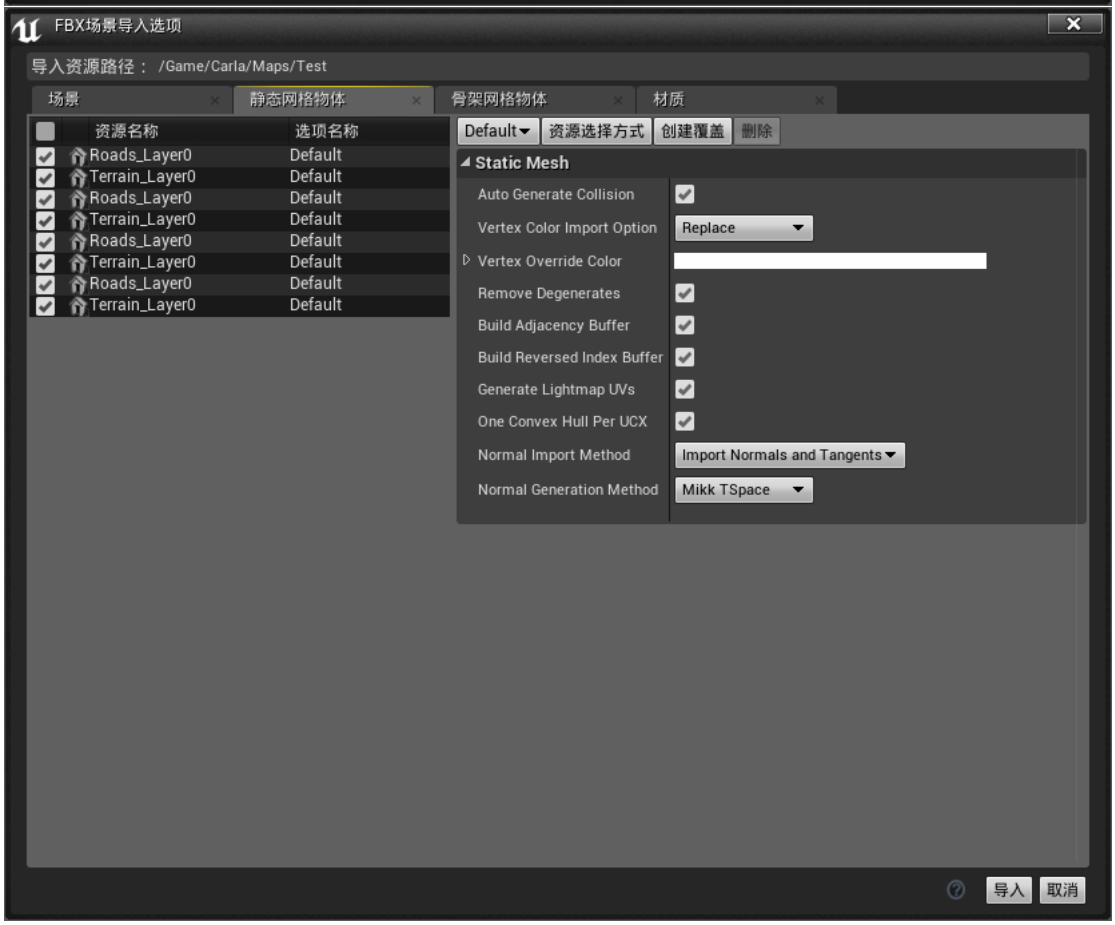
(图 5)

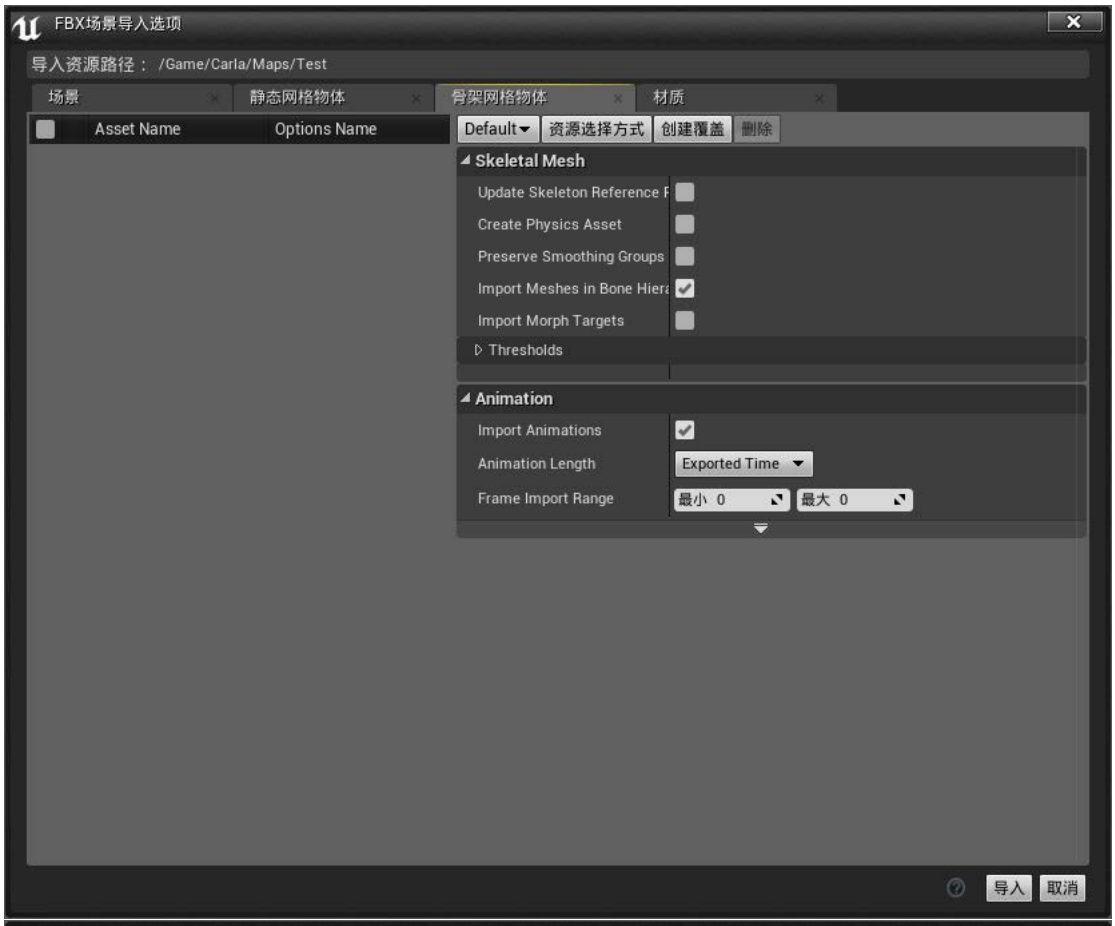


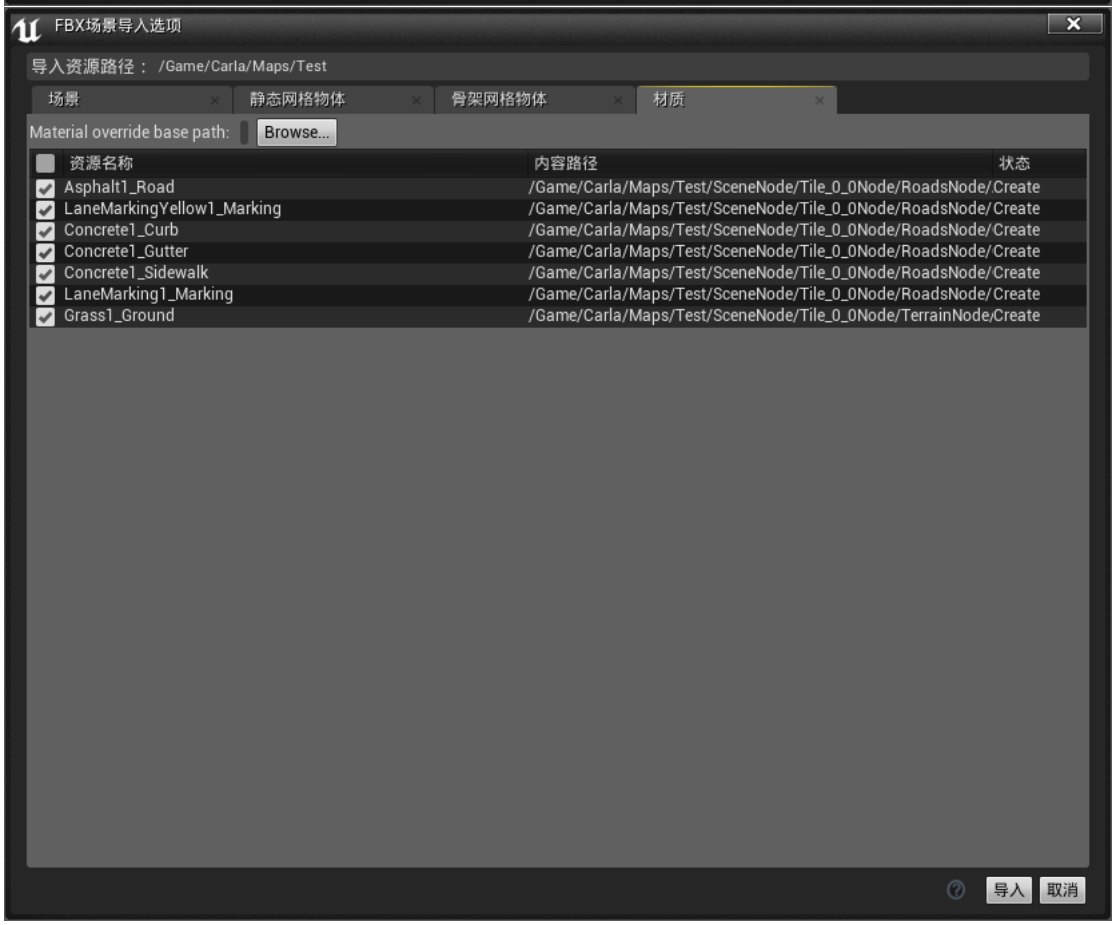
(图 6)

进入 RoadRunner 导出的.fbx 文件的文件夹，选择刚刚导出的.fbx 文件，选择打开。之后的设置，如下图所示。


FBX场景导入选项
导入资源路径： /Game/Carla/Maps/Test
场景
静态网格物体
骨架网格物体
材质
所有
展开全部项
合并全部项
RootNode
Import Options
Create Content Folder Hierar
Import as Dynamic
Hierarchy Type
Create one Blueprint asset
Force Front XAxis
Meshes
Bake Pivot in Vertex
Import Static Mesh LODs
Import Skeletal Mesh LODs
Texture
Invert Normal Maps
导入
取消

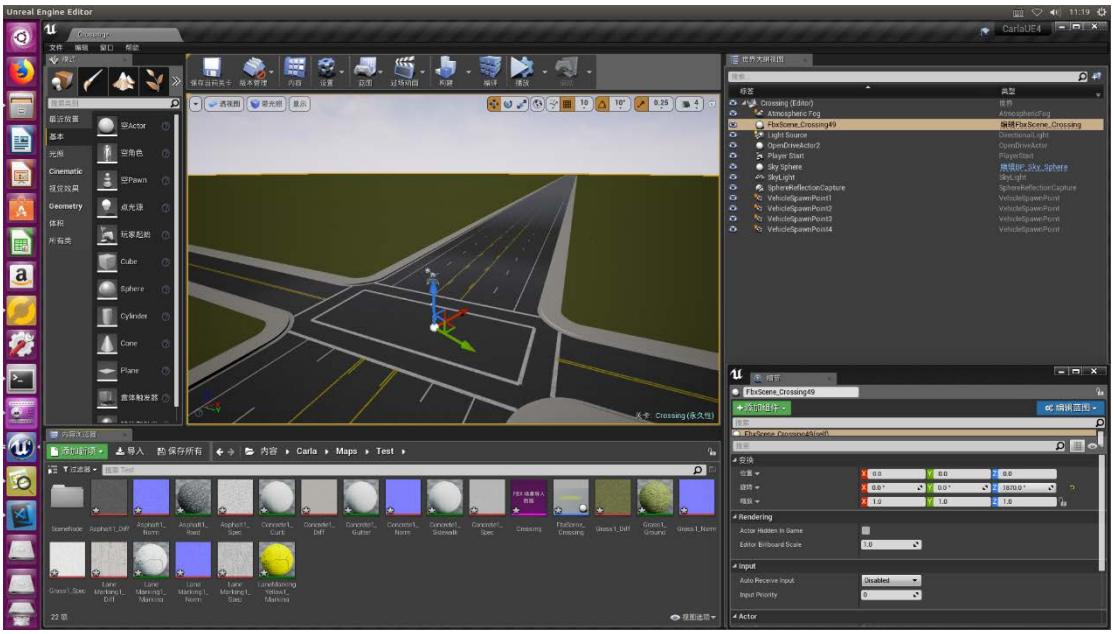

FBX场景导入选项
导入资源路径： /Game/Carla/Maps/Test
场景
静态网格物体
骨架网格物体
材质
资源名称
选项名称
Roads_Layer0 Default
Terrain_Layer0 Default
Roads_Layer0 Default
Terrain_Layer0 Default
Roads_Layer0 Default
Terrain_Layer0 Default
Roads_Layer0 Default
Terrain_Layer0 Default
Default
资源选择方式
创建覆盖
删除
Static Mesh
Auto Generate Collision
Vertex Color Import Option
Replace
Vertex Override Color
Remove Degenerates
Build Adjacency Buffer
Build Reversed Index Buffer
Generate Lightmap UVs
One Convex Hull Per UCX
Normal Import Method
Import Normals and Tangents
Normal Generation Method
Mikk TSpace
导入
取消


FBX场景导入选项
导入资源路径： /Game/Carla/Maps/Test
场景
静态网格物体
骨架网格物体
材质
Asset Name
Options Name
Default
资源选择方式
创建覆盖
删除
Skeletal Mesh
Update Skeleton Reference F
Create Physics Asset
Preserve Smoothing Groups
Import Meshes in Bone Hier
Import Morph Targets
Thresholds
Animation
Import Animations
Animation Length
Exported Time
Frame Import Range
最小 0
最大 0
导入
取消


FBX场景导入选项
导入资源路径： /Game/Carla/Maps/Test
场景
静态网格物体
骨架网格物体
材质
Material override base path:
Browse...
资源名称
内容路径
状态
Asphalt1_Road /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/RoadsNode/ Create
LaneMarkingYellow1_Marking /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/RoadsNode/ Create
Concrete1_Curb /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/RoadsNode/ Create
Concrete1_Gutter /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/RoadsNode/ Create
Concrete1_Sidewalk /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/RoadsNode/ Create
LaneMarking1_Marking /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/RoadsNode/ Create
Grass1_Ground /Game/Carla/Maps/Test/SceneNode/Tile_0_0Node/TerrainNode/ Create
导入
取消

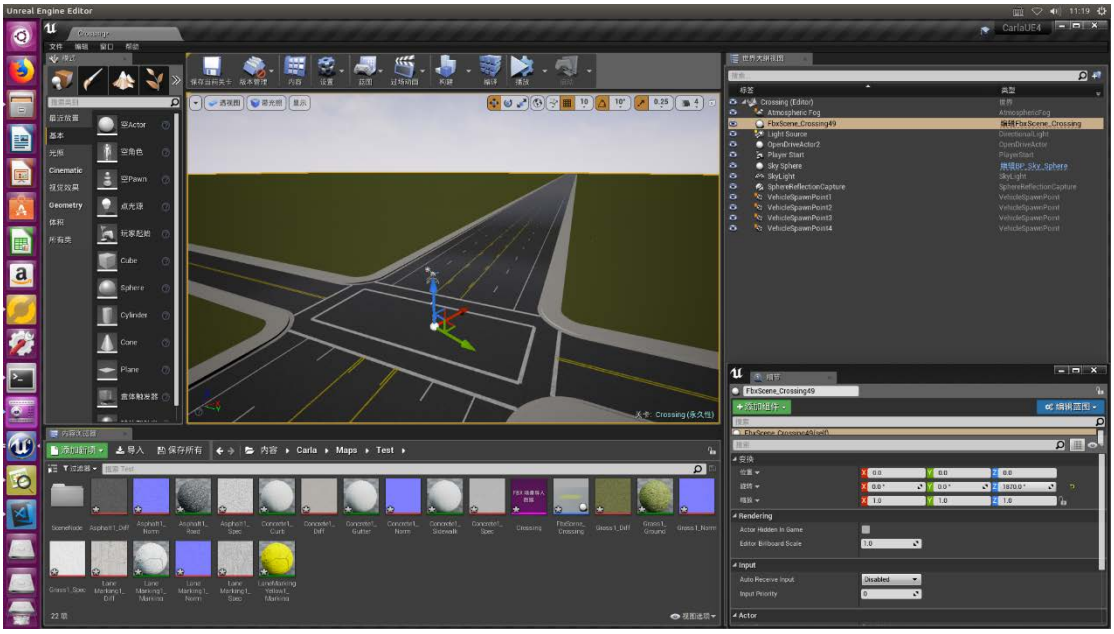
(图 7，8，9，10)

确认无误后，点击导入，我们就成功导入了刚刚搭建的十字路口路网。



(图 11)

注意，在导入成功后，一定要记得选中整个路网地图（单个模型）来改变他的“变换”值，该值的设置情况如图 13 所示，改变这些值是为了让该道路模型和 OpenDRIVE 文件的道路信息相匹配。在该例中路网模型的标签名字是 FbxScene_Crossing49，如图 12 所示。



(图 12)

(图 13)

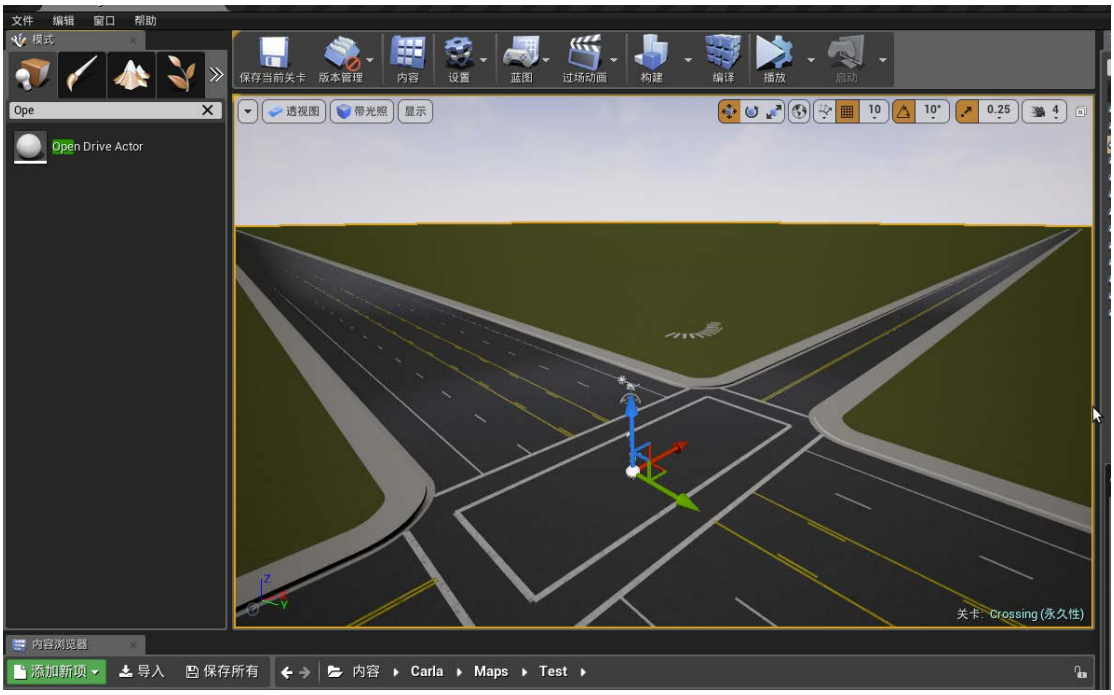
- 导入.xodr 文件

在完成了.fbx 文件的导入后，将.xodr 文件复制存入文件夹 OpenDrive。他的地址及打开方式如图 14 所示。注意，该文件的名字一定要和刚刚创建的场景名字一致，由于场景名叫做 Crossing，故相同的，也将该.xodr 文件取名为 Crossing，如图 14 所示。

(图 14)

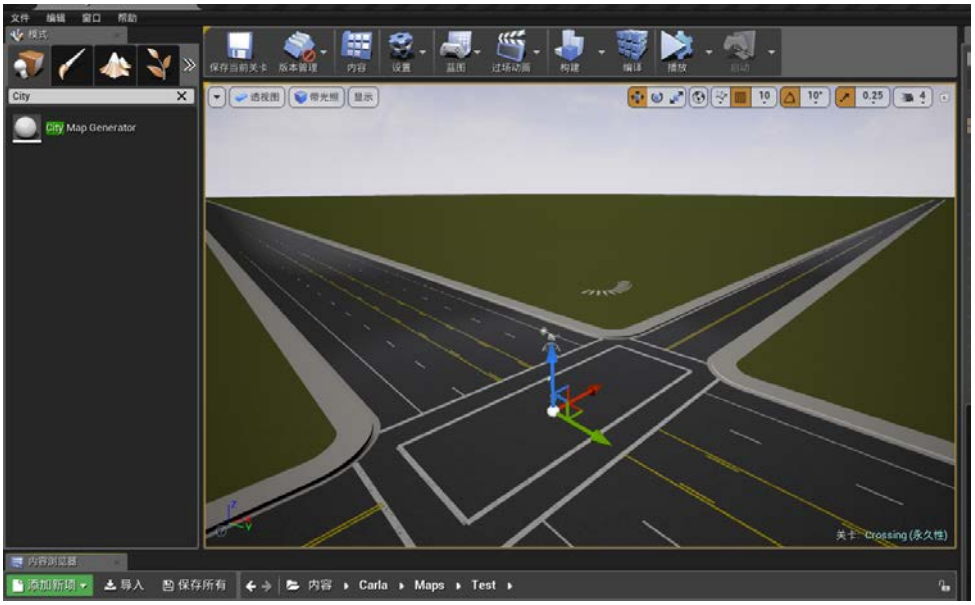
- 放置必备 Actor

  首先，如图 15 所示，放置 Actor – OpenDriveActor，该 Actor 的目的是让该地图能够读取存放的同名.xodr 文件。



(图 15)

其次，如图 16 所示，放置 Actor – CityMapGenerator，该 Actor 的目的是让该场景中的交通车能够开启自动驾驶模型以方便交通流的设定。注意，该 Actor 的变换也需要进行设置，需要和导入的.fbx 路网地图完全一致，如图 17 所示。



(图 16)

(图 17)

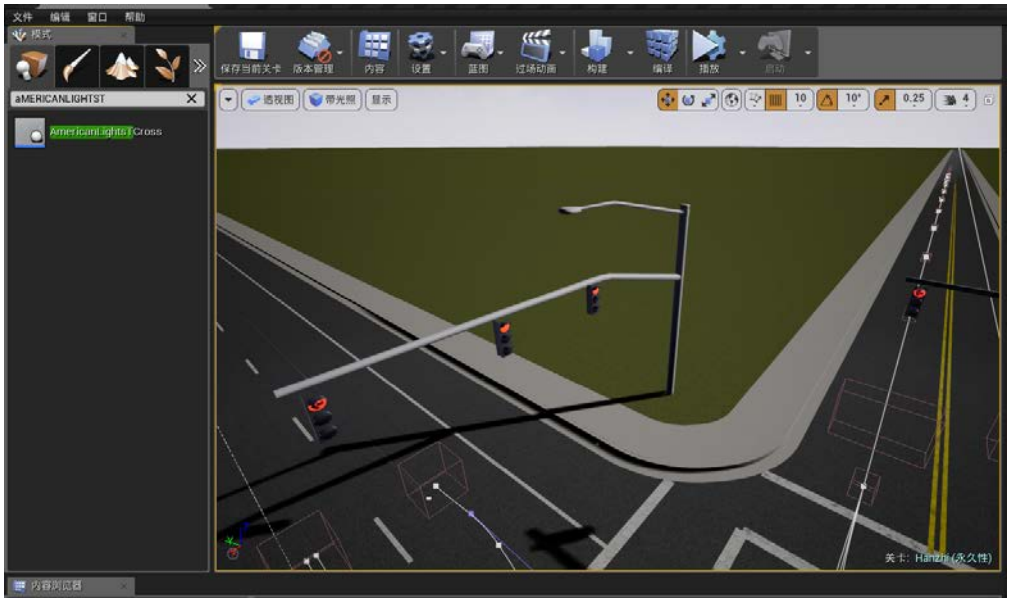
2. 交通信号配置

在 CARLA 的测试场景中，有两样交通信号最为重要，一个是交通灯（群），一个是限速牌，我们在该场景中将放置这两样交通信号。

2.1. 交通灯（群）配置

- 单个交通灯的放置

  同样的，选中 CARLA 自带的交通灯素材，在此处我们以美式交通灯为例，放置了该交通灯实例，具体的设置可以在该 Actor 中的细节界面内进行设定，在此处我们就使用其原自带的相关细节。

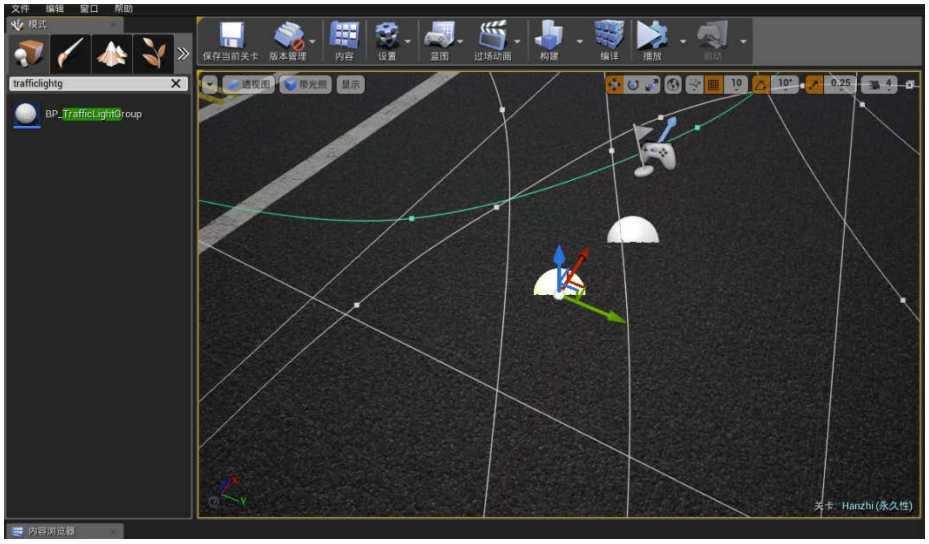


(图 18)

- 交通灯群的放置

相同的，我们在四个路口均放置了一个美式交通灯，如图 19 所示。在安放了四个交通灯之后，我们需要协调各个交通灯的状态以避免冲突，因此，在该环节需要使用重要的 Actor – BP_TrafficLightGroup 来囊括该十字路口的四个交通灯，如图 20 所示。

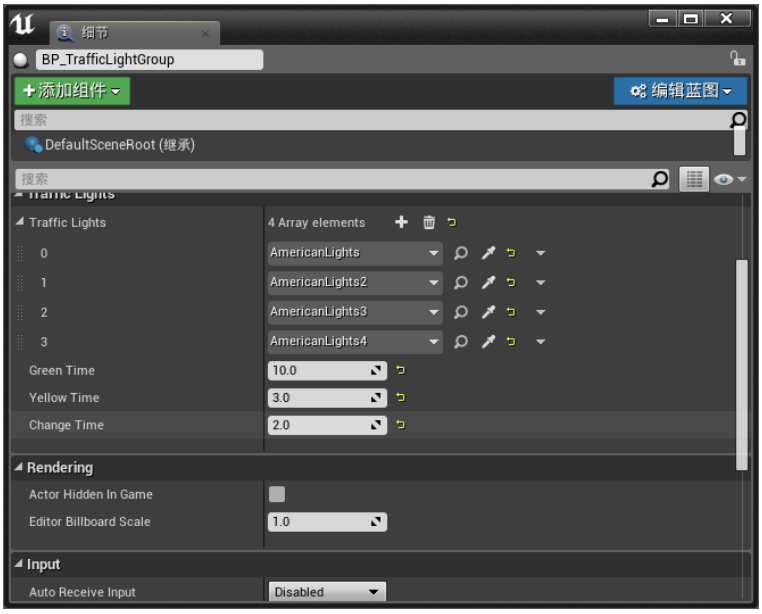
（图 19）



(图 20)

- 交通灯群的逻辑配置

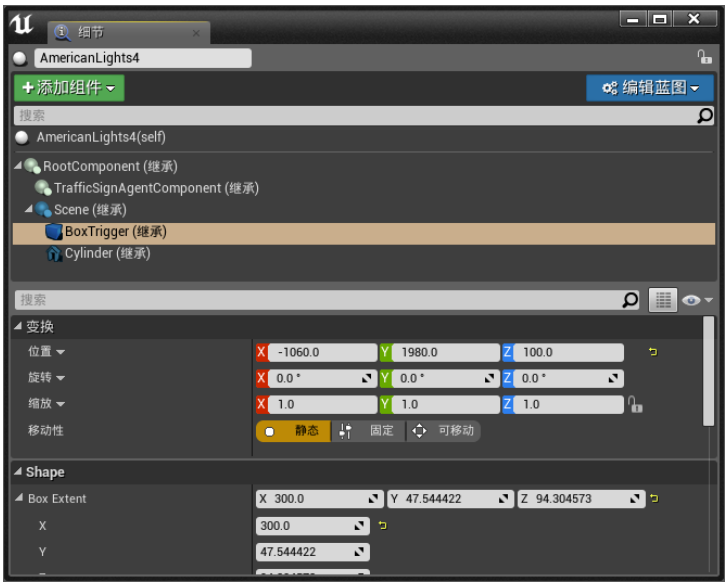
  在放置了交通灯群 Actor - BP_TrafficLightGroup 后，首先我们需要将需要设置逻辑的交通灯放入，并且设置绿灯时间、黄灯时间以及切换时间，在这之后，就可以生成一个不会发生逻辑冲突的交通灯组。该设置如图 21 所示。



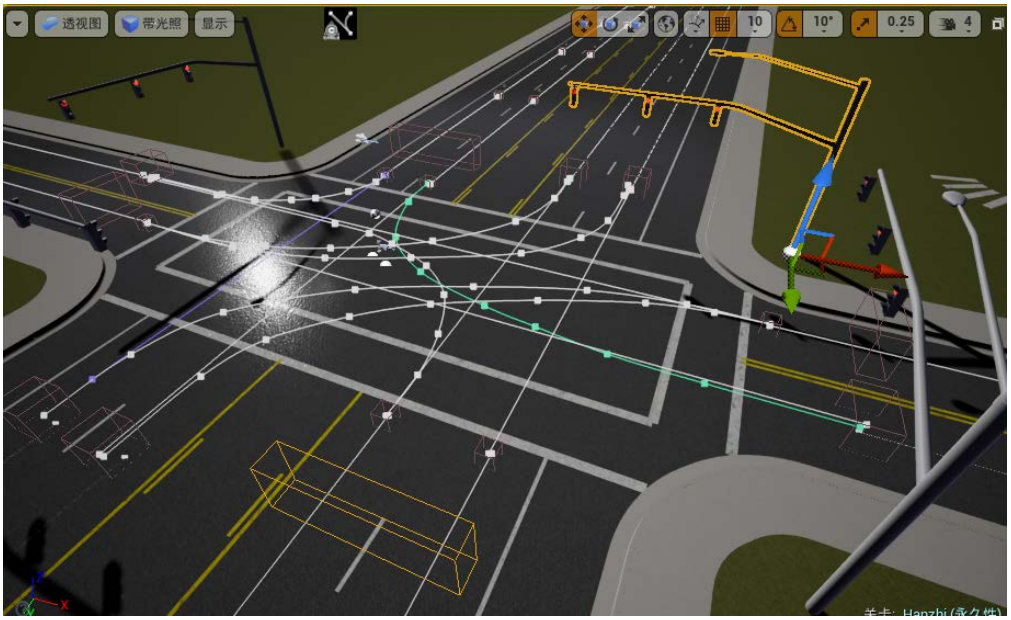
（图 21）

- 单个交通灯 BoxTrigger 的放置

BoxTrigger 是单个交通灯所附着的一个组件，他的作用是能够将交通灯的信息传递给自动驾驶模式下碰到 BoxTrigger 的交通车，从而做出相应的反馈动作（红灯停、绿灯行）。该 BoxTrigger 的选中方式如图 22 所示，在选中该 BoxTrigger 后，我们可以对其位置（变换）、尺寸（Box Extent）等参数进行修改，，它的位置可以在界面内进行拖拽，其位置通常在交通灯的马路对面，并且其宽度要包括受该交通灯影响的车道，因此其布局方式如图 23 所示。其他交通灯的 BoxTrigger 布局也同理。



(图 22)



(图 23)

2.2. 限速牌配置

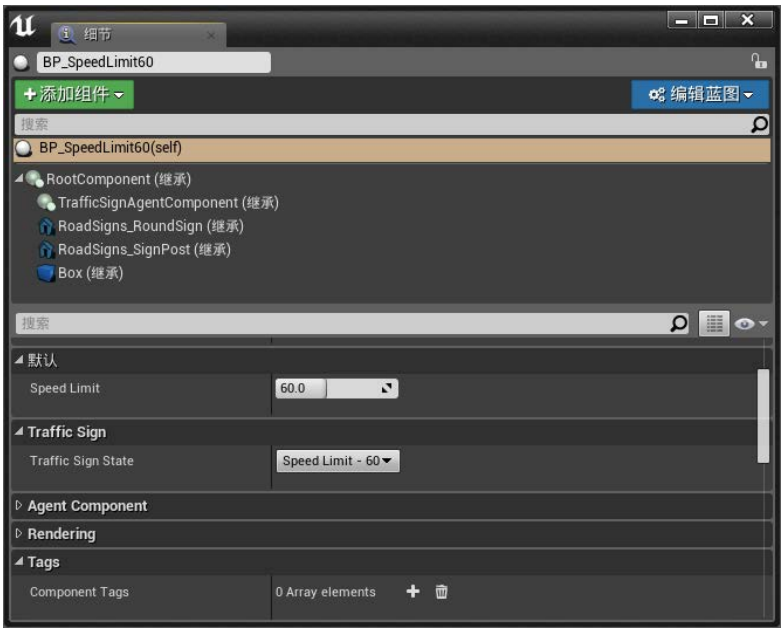
- 单个限速牌的放置

  同样的，选中 CARLA 自带的限速牌素材，在此处我们以 BP_SpeedLimit60 为例，放置了该交通灯实例，具体的设置可以在该 Actor 中的细节界面内进行设定。

(图 24)

- 具体参数配置

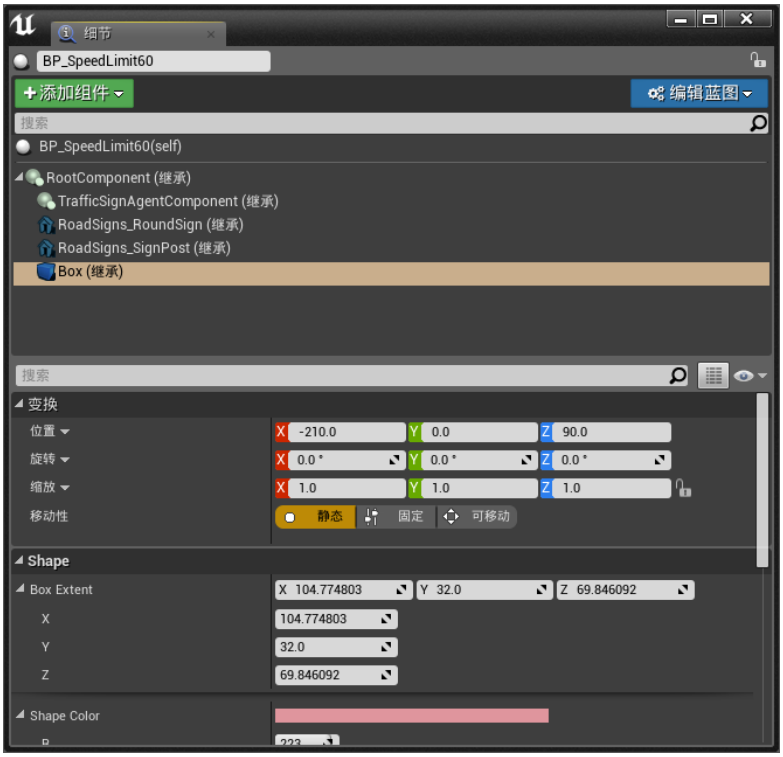
在细节窗口内，也可以对该限速牌的限制速度进行设置，如图 25 所示。



(图 25)

- BoxTrigger 的放置

  与交通灯相同，限速牌也需要利用 BoxTrigger 将其限速的信息传递给过往的交通车，因此，如图 26 所示，同样的也可以对该 Box 类的位置，尺寸等参数进行设置，它的位置可以在界面内进行拖拽，该设置与交通灯的 BoxTrigger 配置方法相同。



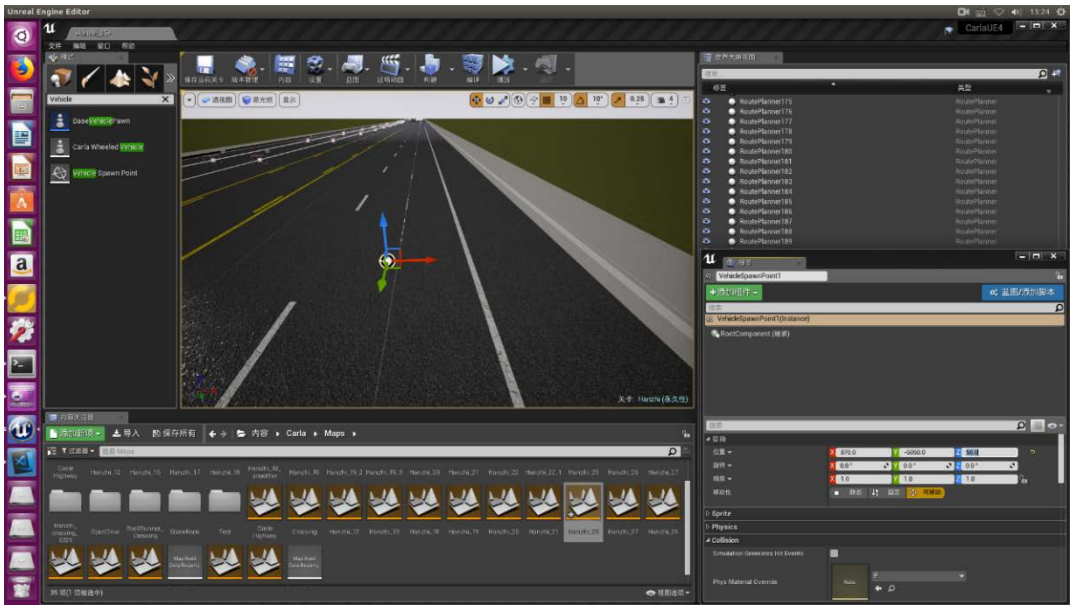
(图 26)

3. 交通车的运行配置

在配置交通车时，有两个重要的 Actor 会被使用到，即 VehicleSpawnPoint 和 RoutePlanner,。前者用于设立交通车的投放点，后者则能够让交通车在场景中按照一定的轨迹，并且遵守交规的运行。

3.1. VehicleSpawnPoint 的配置

该 Actor 将某一点作为可投放点添加到该场景的投放点列表中，以使得后期投放交通车时能够选用该点作为投放点。他的位置，旋转均可以在细节栏的变换中进行设置。值得注意的是，最好让该投放点的位置离地约 50cm (即设置位置中的 z 为 50)。该配置环节如图 27 所示。



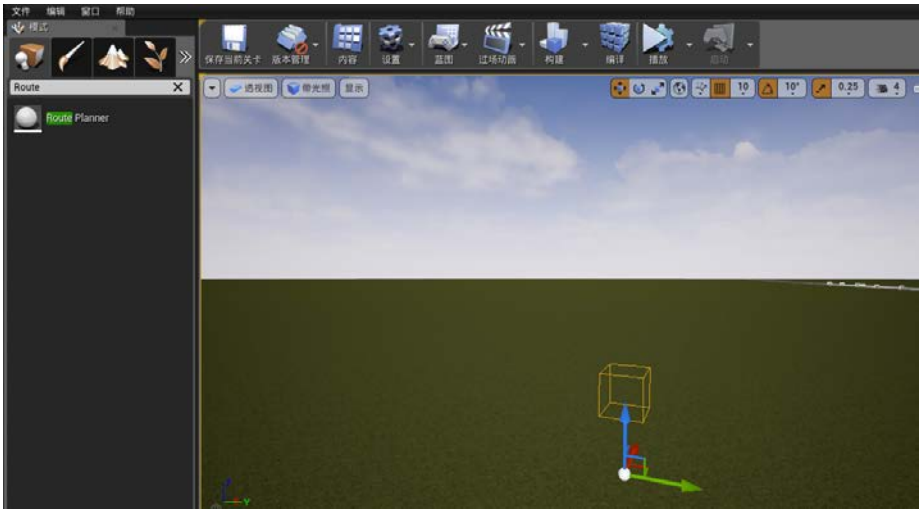
（图 27）

3.2. RoutePlanner 的配置

RoutePlanner 是一种用来规划车辆运行路径的 Actor，与前面提到的交通灯及限速牌相同，他也需要利用 BoxTrigger 将规划的路径信息传递给车辆，以使得车辆能够按照既定路径来运行。

- RoutePlanner 的选择与放置

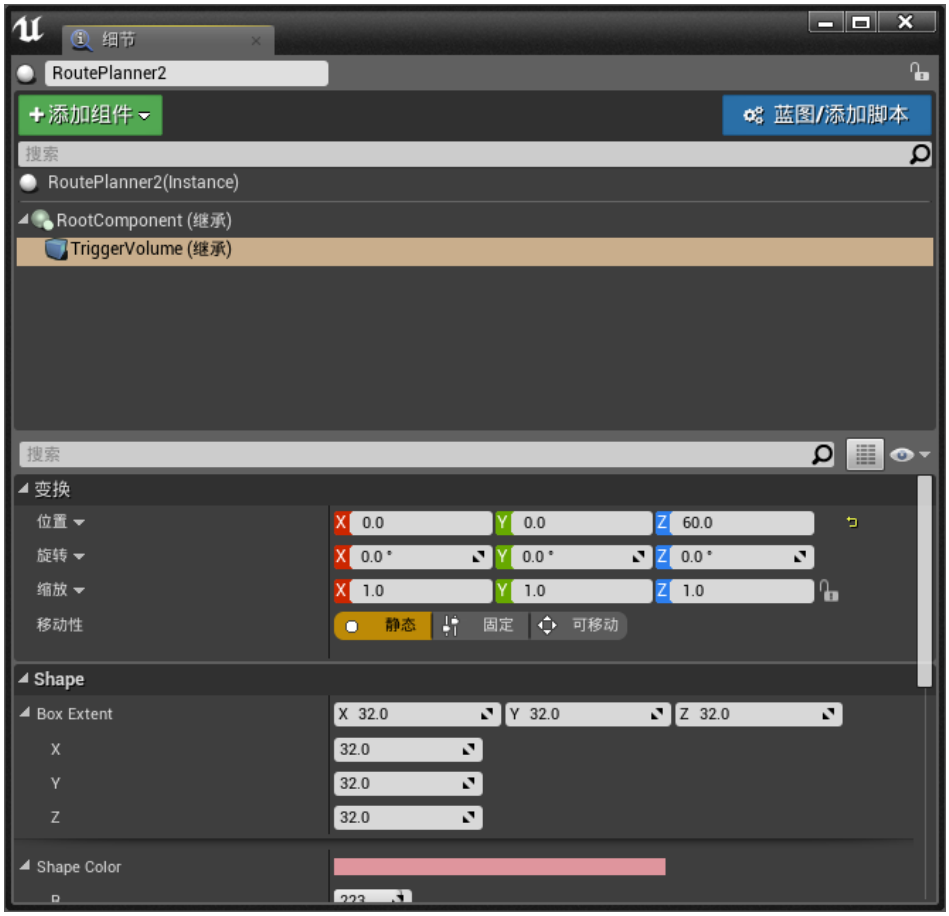
  同理，在 CARLA 素材库中找到 RoutePlanner，将其放置到场景中。如图 28 所示。



（图 28）

- TriggerVolum 的设置

  同理交通灯的 BoxTrigger 与限速牌的 Box，TriggerVolum 的位置、尺寸也能够进行设置，设置的方法如图 29 所示。

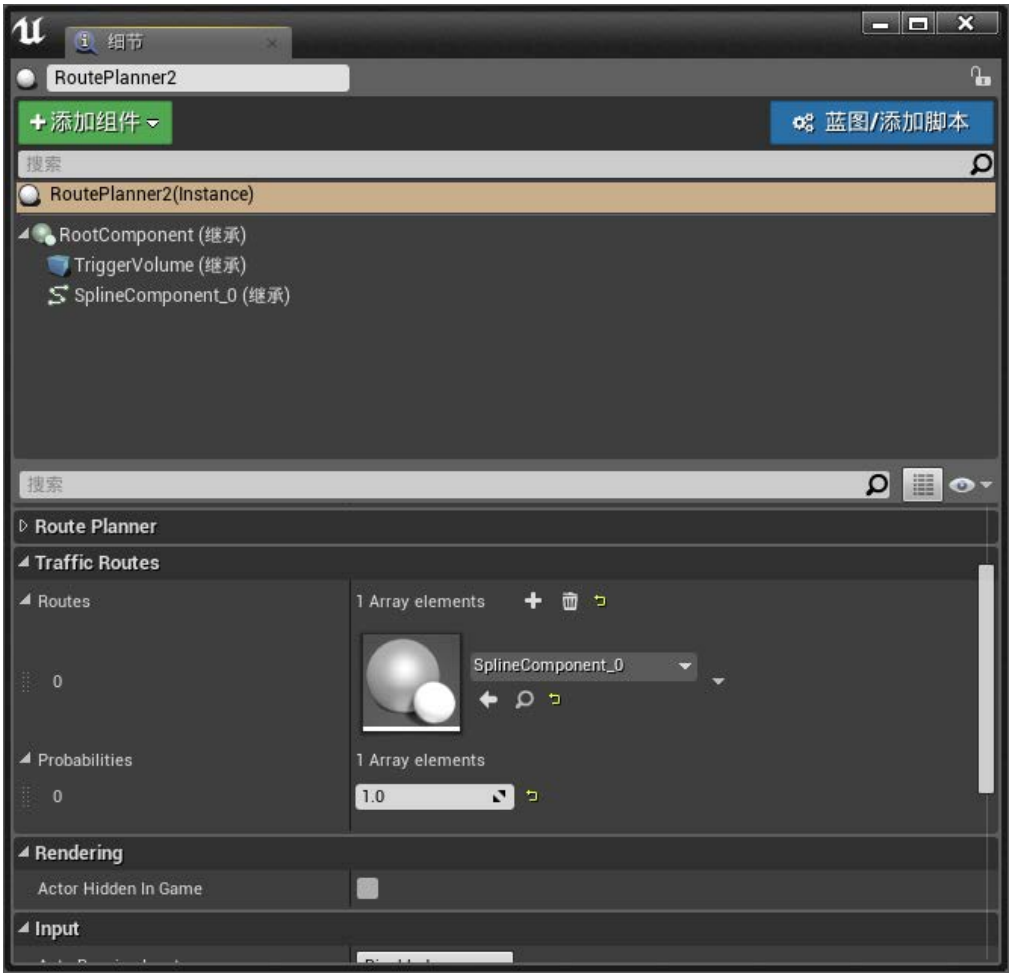


（图 29）

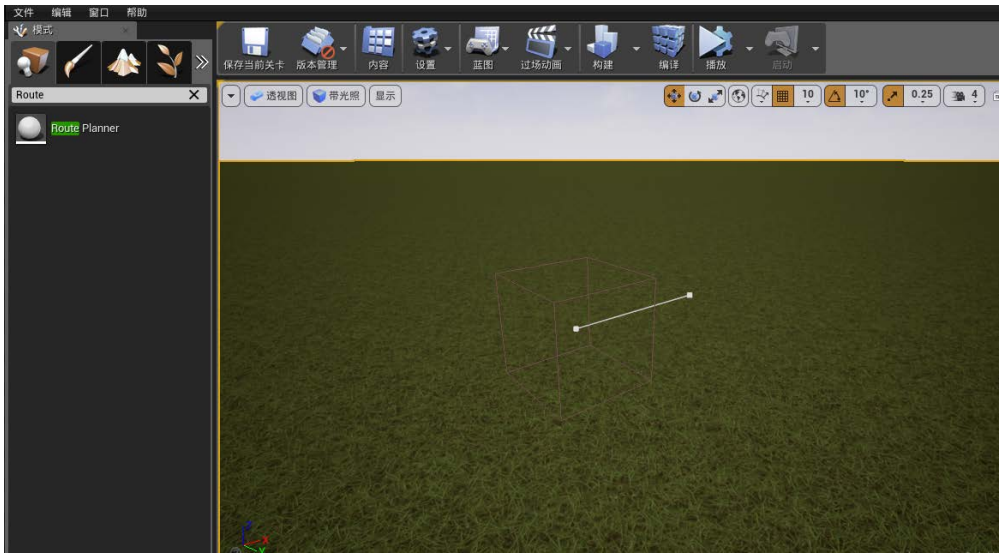
- RoutePlanner 规划路径的添加和绘制

  选中 RoutePlamner，在 Traffic Routes 内添加一条路径，如图 30 所示，此时

就会在界面内生成一个很短的路径。如图 31 所示。



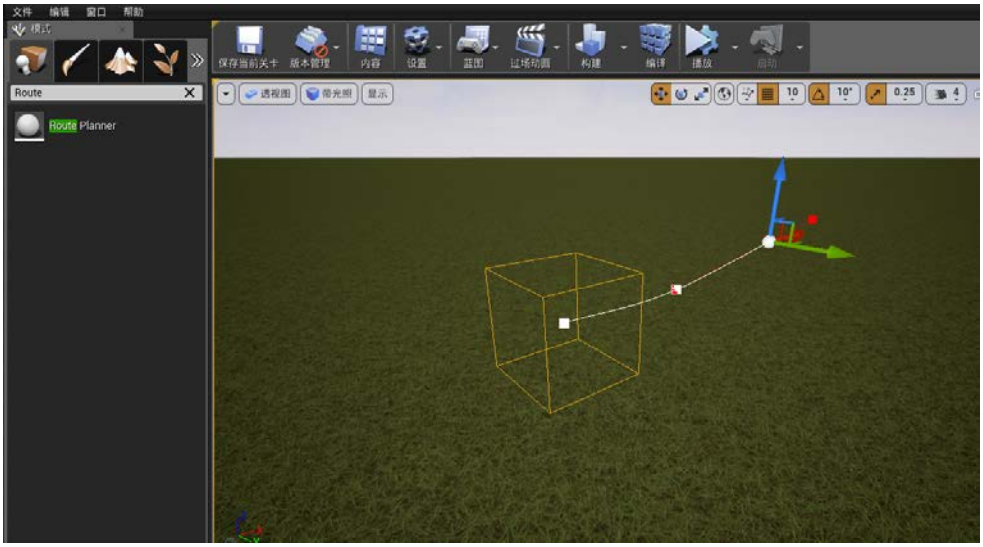
（图 30）



（图 31）

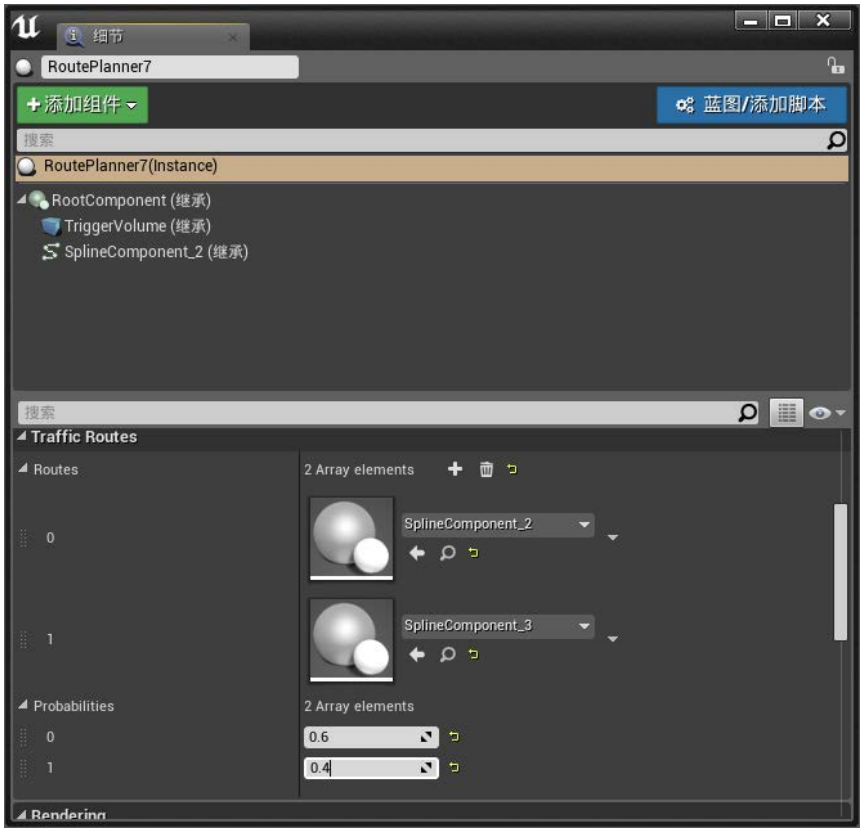
此时，左键选中生成路径的小白点，之后右键点击生成路径的小白点，选中“克隆样条曲线点”，就可以再次新生成一曲线节点，通过移动该点就可以绘

制该曲线的轨迹，如图 32 所示。通过不断的克隆曲线点并且进行移动，就能够绘制出一条交通车运行轨迹，当自动驾驶交通车遇到 RoutePlanner 时，就可以按照该轨迹来运行。



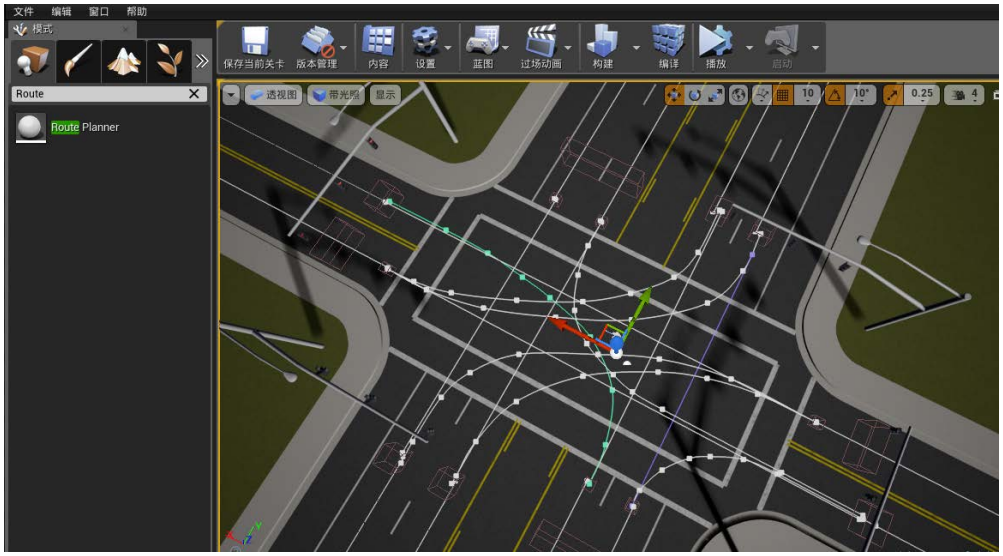
（图 32）

注意，一个 RoutePlanner 可以添加多条轨迹，但是每条轨迹都应该设置相应的概率值，如图 33 所示。添加完该轨迹后，该轨迹的绘制方法与上一条轨迹的绘制方法一致。



（图 33）

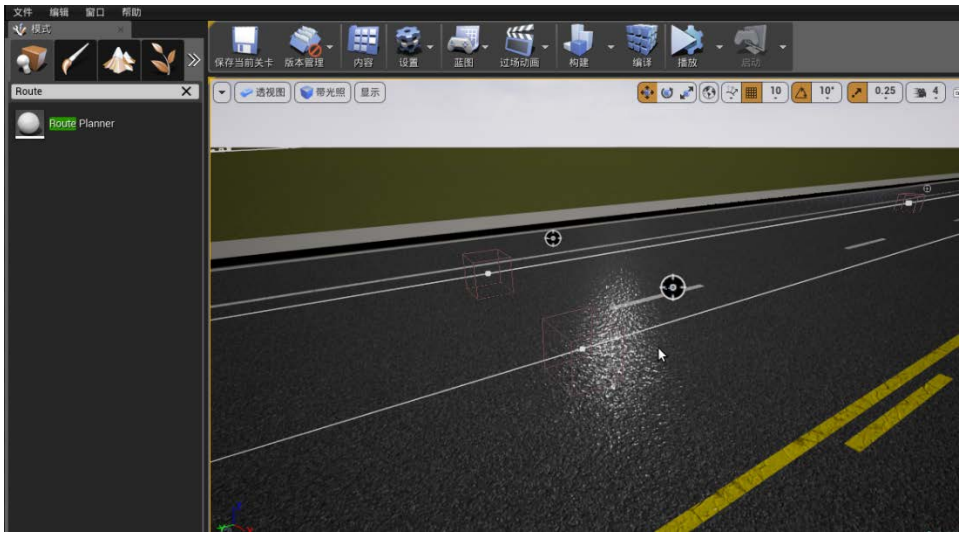
当交通车经过该 RoutePlannear 时，会根据概率值大小选择一条轨迹来行驶。而最终经过绘制及概率设置后，生成的运行路径如图 34 所示。



（图 34）

3.3. VehicleSpawnPoint 和 RoutePlanner 的联合配置

为了能够让交通车在投放后比较平稳的运行，通常可以将 VehicleSpawnPoint 放置于 RoutePlanner 后上部的地方，不用过分严格，如图 35 所示。



（图 35）

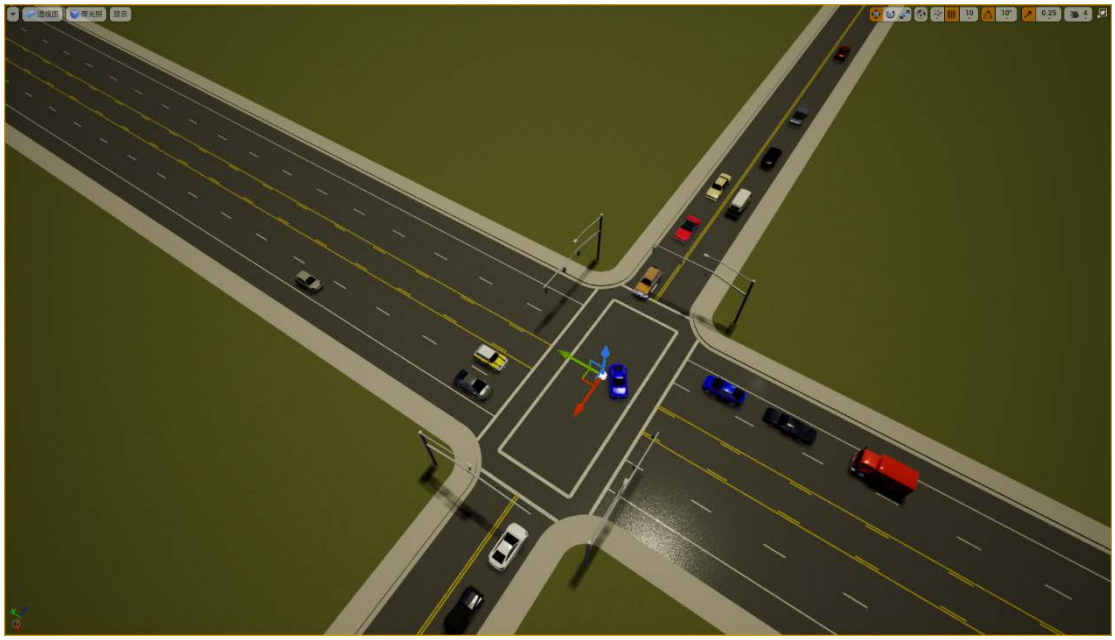
4. 交通车投放脚本运行

在 UE4 搭建好场景并且点击“播放”后，我们可以通过运行 CARLA 自带的脚本来在测试场景中投放交通车，如图 36 所示。最终的交通车运行效果如图 37

所示。

```
hirain777@hirain777-580-076cn: ~/carla-0.9.2/PythonAPI
hirain777@hirain777-580-076cn:~$ cd carla-0.9.2/PythonAPI/
hirain777@hirain777-580-076cn:~/carla-0.9.2/PythonAPI$ python spawn_npc.py --saf
e -n 30
found 122 spawn points.
spawned 'vehicle.citroen.c3' at Location(x=8.99999, y=382.5, z=0.3)
spawned 'vehicle.carlamotors.carlacola' at Location(x=-5.3, y=-281.4, z=0.8)
spawned 'vehicle.chevrolet.impala' at Location(x=-233.3, y=1.9, z=0.3)
spawned 'vehicle.tesla.model3' at Location(x=5.3, y=144.7, z=0.3)
spawned 'vehicle.ford.mustang' at Location(x=5.3, y=127.7, z=0.3)
spawned 'vehicle.citroen.c3' at Location(x=9, y=178.4, z=0.3)
spawned 'vehicle.volkswagen.t2' at Location(x=9, y=195.3, z=0.3)
spawned 'vehicle.tesla.model3' at Location(x=5.3, y=110, z=0.3)
spawned 'vehicle.ford.mustang' at Location(x=-2.1, y=-298.3, z=0.8)
spawned 'vehicle.bmw.grandtourer' at Location(x=5.29999, y=449.5, z=0.3)
spawned 'vehicle.carlamotors.carlacola' at Location(x=-5.3, y=-26.2, z=0.8)
spawned 'vehicle.toyota.prius' at Location(x=9, y=228.6, z=0.3)
spawned 'vehicle.ford.mustang' at Location(x=-2.1, y=-349.5, z=0.8)
spawned 'vehicle.audi.tt' at Location(x=154.1, y=-2.1, z=0.8)
spawned 'vehicle.toyota.prius' at Location(x=-5.3, y=-196.3, z=0.8)
spawned 'vehicle.mini.cooperst' at Location(x=5.3, y=211.9, z=0.3)
spawned 'vehicle.jeep.wrangler_rubicon' at Location(x=171.1, y=-2.1, z=0.8)
spawned 'vehicle.nissan.micra' at Location(x=8.99999, y=280.1, z=0.3)
spawned 'vehicle.carlamotors.carlacola' at Location(x=9, y=144.7, z=0.3)
spawned 'vehicle.chevrolet.impala' at Location(x=-5.3, y=-213.3, z=0.8)
```

（图 36）



（图 37）