概述

Simulation of Urban Mobility（SUMO）作为一款交通仿真软件，其通过结合 OpenStreetMap 官网提供的真实道路网可以生成逼真的车辆交通轨迹。OPNET 是一款流行的网络仿真软件，它通过建立网络设备、通信链路和协议模型并模拟流量的传输，来评估网络性能并对其进行设计优化，可以被很好应用于车辆网通信仿真。然而，SUMO 生成的轨迹文件不直接提供对 OPNET 仿真的支持。为此，本项目采用 JAVA 语言开发了一款可以实现 SUMO 生成轨迹转换为 OPNET 仿真轨迹的工具，旨在实现二者的联合仿真以为车联网的研究提供更加真实的仿真场景。

部署环境

Win 7 + java-1.8.0-openjdk-devel.x86_64
SUMO-1.6.0
OpenStreetMap
OPNET 14.5

部署步骤

整体的部署过程如图 1 所示，详细的步骤如下所述：

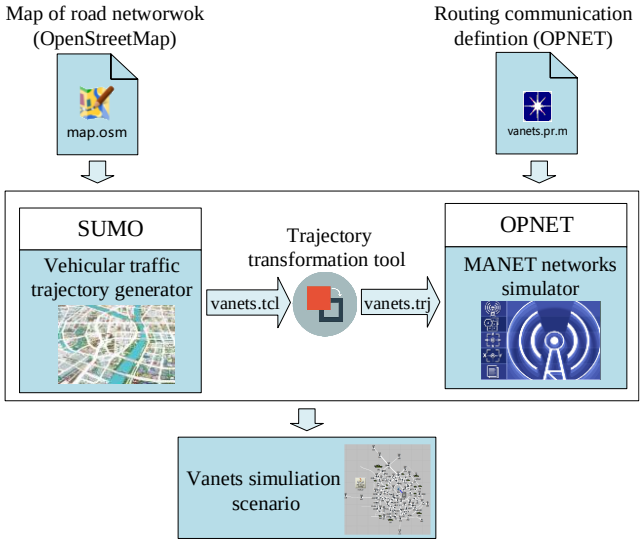

图 1.基于 SUMO 和 OPNET 联合的车联网仿真

Step 1：OSM 真实地图文件获取

打开 openstreetmap 官网（https://www.openstreetmap.org），点击右上角的“导出”，在 4 个空白框中输入需要选择的区域经纬度范围，在次点击蓝色背景框的“导出” 生成该区域的 OSM 地图文件 ***map.osm***，具体如图 2 所示。

图 2.OSM 真实地图文件获取

官网下载的 osm 除了路网信息还包含别的东西，需要做简单处理：手动打开 sumo\doc\userdoc\Networks\Import\OpenStreetMap.html 文件，然后在打开的网页中间部分 Importing additional Polygons (Buildings, Water, etc.)的下面有一大段 xml 的代码，复制这些代码，然后在 notepad++中级新建文件并把复制的代码粘贴到里面，在本实验中因为不需要，所以删除了下面的 id=“power”的那一行代码，然后保存为 ***typemap.xml***，把这个文件保存到上一步骤建的 map 文件中去，和 osm 地图文件放一起。

Step2：生成 ***map.net.xml*** 和 ***map.poly.xml*** 文件

在 cmd 命令框中打开 ***map*** 文件夹，然后依次运行如下两条命令（如图 3 所示）：

(1) netconvert --osm-files map.osm -o map.net.xml

(2) polyconvert --net-file map.net.xml --osm-files map.osm --type-file typemap.xml -o map.poly.xml

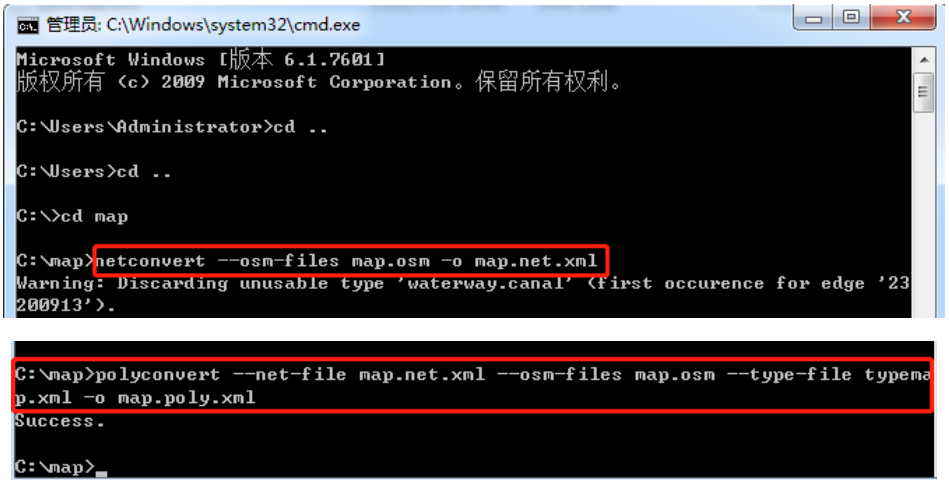

图 3.生成 map.net.xml 和 map.poly.xml 文件

Step3：生成 ***map.rou.xml*** 文件

在 cmd 命令框中，继续依次运行如下两个命令，如图 4 所示.

(1) python C:/sumo/tools/randomTrips.py -n map.net.xml -e 100 –l

(2) python C:/sumo/tools/randomTrips.py -n map.net.xml -r map.rou.xml -e 400 -l

(命令 2 中 400 表示场景中设定的车辆数量，使用者可以结合仿真场景自行设

定)

```
C:\map>python C:/sumo/tools/randomTrips.py  -n map.net.xml -e 100  - l

C:\map>python C:/sumo/tools/randomTrips.py -n map.net.xml -r map.rou.xml -e 400
-l
calling  C:\Sumo\bin\duarouter -n map.net.xml -r trips.trips.xml --ignore-errors
 --begin 0 --end 400.0 --no-step-log --no-warnings -o map.rou.xml
Success.

C:\map>
```

图 4.生成 map.rou.xml 文件

Step4：编辑配置文件 ***myConfig.sumocfg***

可以在 sumo 文件夹中搜索 test.sumocfg 然后更改 input 文件，便可以得到如下图 5 所示文件，然后把文件命名为 ***myConfig.sumocfg*** 并保存到 map 文件夹中

```
<?xml version="1.0" encoding="UTF-8"?>

<!-- generated on Tue Apr 28 00:06:20 2020 by Eclipse SUMO Version 1.6.0
This data file and the accompanying materials
are made available under the terms of the Eclipse Public License v2.0
which accompanies this distribution, and is available at
http://www.eclipse.org/legal/epl-v20.html
SPDX-License-Identifier: EPL-2.0
-->

<configuration xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="http://sumo.dlr.de
    <input>
        <net-file value="map.net.xml"/>
        <route-files value="map.rou.xml"/>
        <additional-files value="map.poly.xml"/>
    </input>
    <time>
        <begin value="0"/>
        <end value="10000"/>
    </time>

</configuration>
```

图 5. 编辑配置文件 myConfig.sumocfg

Step5：生成 SUMO 轨迹文件 ***ns2mobility.tcl***

在 cmd 命令框中，依次运行下面两条命令，如图 6 所示。

(1) sumo -c myConfig.sumocfg --fcd-output sumoTrace.xml

(2) traceExporter.py --fcd-input sumoTrace.xml --ns2mobility-output ns2mobility.tcl

```
C:\map>sumo -c myConfig.sumocfg --fcd-output sumoTrace.xml
Loading configuration ... done.
Step #0.00 (7ms ~= 142.86*RT, ~142.86UPS, vehicles TOT 1 ACT 1 BUF 0)
Step #100.00 (3ms ~= 333.33*RT, ~30666.67UPS, vehicles TOT 94 ACT 92 BUF 0)
Step #200.00 (4ms ~= 250.00*RT, ~36500.00UPS, vehicles TOT 189 ACT 146 BUF 0)
Step #300.00 (6ms ~= 166.67*RT, ~30333.33UPS, vehicles TOT 280 ACT 182 BUF 0)
Step #400.00 (4ms ~= 250.00*RT, ~56250.00UPS, vehicles TOT 372 ACT 225 BUF 0)

C:\map>traceExporter.py --fcd-input sumoTrace.xml --ns2mobility-output ns2mobil
ty.tcl
```

图 6.生成 SUMO 轨迹文件 ns2mobility.tcl

Step6：SUMO 轨迹文件 ***ns2mobility.tcl*** 转换为 OPNET 仿真软件支持的文件格式 (*.***trj***)

(1) 在 Java IDE 中新建工程 **SUMO_Pretreatment**，并运行代码 ***SUMO_Pretreatment.java***，将 SUMO 轨迹预处理的结果按单个车辆分类写入文件夹 SUMO_F。

(2) 在 Java IDE 中再次新建工程 **SUMO_To_OPNET_Trj**，并运行代码 ***SUMO_To_OPNET_Trj.java***，将 SUMO 轨迹预处理的结果集逐一转化为 OPNET 轨迹文件

Step7：OPNET 轨迹文件导入车联网仿真场景

将第 6 步生成的车辆轨迹文件复制到 OPNET 安装文件夹 op_models 中，新建 OPNET 工程，添加节点并关联移动轨迹。OPNET 的一大优点是提供了方便的图形化建模，但是有时候图形化建模方式并不能满足我们的需求，比如需要部署几千个节点，如果一个个手工部署这些节点，则太过烦琐，用程序随机化部署有可能导致节点位置随机化一部分节点簇拥到一起，而有些地方又无法部署到。我们在随机部署节点时，随机生成下一网络节点的坐标；在生成节点坐标的过程中，要避免让新的节点坐标距离已有的节点坐标太近。针对这个需求采用了文本方式建模(external model access, EMA) , EMA 可以精确定义节点属性，可以用循环语句来刻画多个特定规格的节点，其具体步骤如下：

（1）在图形界面下做一个简单的网络模型，只包含一个移动节点，如下图 7(a)所示。.

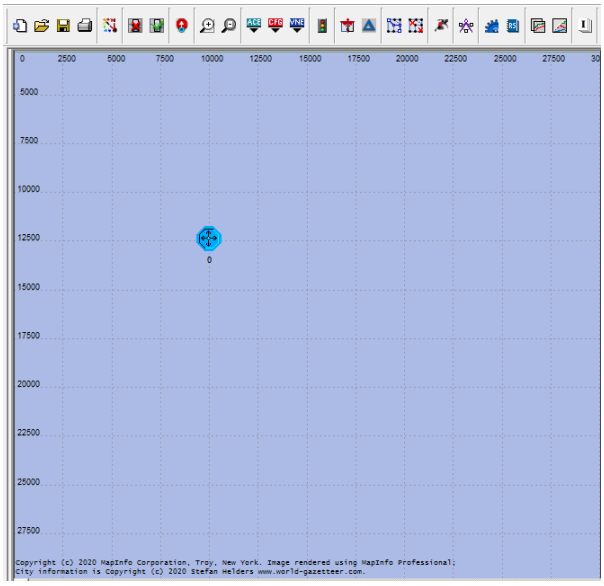

图 7(a).单节点网络模型

（2）选择菜单**Topology->Export Topology->To EMA** 生成EMA文件(***.em.c***)，观察EMA文件，查找name属性为"0"的节点，代码如下图7(b)所示。

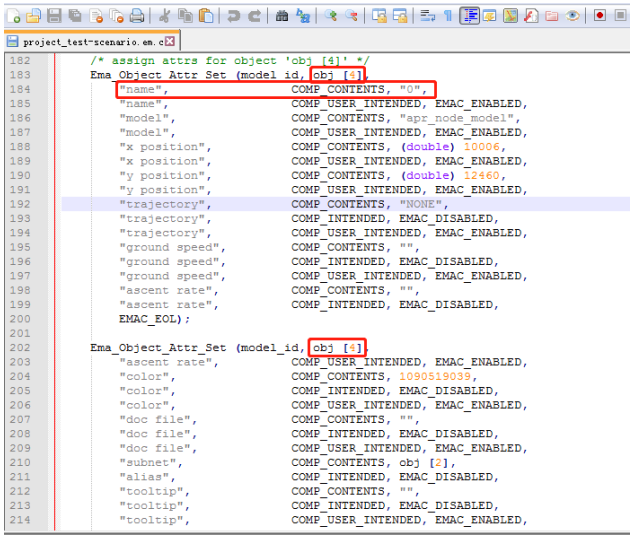

图 7(b). EMA

从上面可以看出，对象数组下标 4 为 0 号节点（注意在你的场景中不一定是

obj[4])。在后面用 for 循环将这些关于 obj[4]的语句括起来。

(3) 添加要用到的头文件。

a.因为要用到 rand()函数，因此需要添加头文件 stdlib.h，即#include <stdlib. h>。

b.在 main()函数前添加需要用到的变量，如下图 8 所示。

```
#include <opnet.h>
#include <ema.h>
#include <opnet_emadefs.h>
#include <opnet_constants.h>
# include <stdlib.h>


/* array for all textlist attributes in model */
Prg_List*       prg_lptr [6];

/* array for all objects in model */
EmaT_Object_Id      obj [13];

# define NODE_NUMBER 800                /*节点数量*/

# define X_MIN 9467.75                  /*场景大小，可以采用SUMO.java输出区域的范围*/
# define X_MAX 13216.94
# define Y_MIN 12293.16
# define Y_MAX 16287.15
# define X_LENGTH 3749.19
# define Y_LENGTH 3993.99

# define MIN_DISTANCE_2 400             /*节点最小距离的平方，这个值越大，节点生成越均匀，这个值理论最大为25 x 25 x 1.414，值越大生成的时间越长*/

int PosChooseAgain;                     /*是否重新生成*/
int NodeNumber;                         /*节点数*/
int i, k;                               /*循环变量*/
char node_name[5];                      /*节点名称*/
char tra[20];                           /*轨迹名称*/
double x_pos,y_pos;                     /*节点坐标*/
double NodeXList[800] , NodeYList[800]; /*节点坐标数组*/
EmaT_Object_Id wsn_node_objid;          /*对象变量*/
```

图 8.添加头文件

(4) 接下来设置节点的属性（如图 9 所示）：将 obj [4]的代码（主要是***ema_obj_attr_set()***函数）手动放到一个 for 循环语句里，替换其中的节点名称属性和坐标(x_postion 和 y_position)为变量，通过循环生成其他节点，在 for 循环中随机生成节点的纵横坐标，将下面代码放到文件的末尾。

```
NodeNumber=0;
for (k=1; k <= NODE_NUMBER; k++){
sprintf(node_name, "%d", k);
sprintf(tra, "10ms_2_Real_SUMO_%d", k);
PosChooseAgain = 1;
while(PosChooseAgain == 1)
{
   x_pos= ((double)rand()/((double)(RAND_MAX) + (double)(1)))*(double)X_LENGTH + X_MIN;
   y_pos= ((double)rand()/((double)(RAND_MAX) + (double)(0)))*(double)Y_LENGTH + Y_MIN;
  for (i=0; i< NodeNumber; i++)
  {
    if (((NodeXList[i]- x_pos)*(NodeXList[i]- x_pos)+(NodeYList[i]- y_pos)*(NodeYList[i]- y_pos))< MIN_DISTANCE_2)
    {
      break;
    }
  }

    if(i == NodeNumber) {
      PosChooseAgain = 0;
    }
}

NodeNumber++;
NodeXList[NodeNumber-1] = x_pos;
NodeYList[NodeNumber-1] = y_pos;
wsn_node_objid= Ema_Object_Create(model_id, OBJ_NT_NODE_MOBILE);

/* assign attrs for object 'obj [4]' */
Ema_Object_Attr_Set (model_id, wsn_node_objid,
    "name",                 COMP_CONTENTS, name,
    "name",                 COMP_USER_INTENDED, EMAC_ENABLED,
    "model",                COMP_CONTENTS, "apr_node_model",
    "model",                COMP_USER_INTENDED, EMAC_ENABLED,
    "x position",           COMP_CONTENTS, (double) x_pos,
    "x position",           COMP_USER_INTENDED, EMAC_ENABLED,
    "y position",           COMP_CONTENTS, (double) y_pos,
    "y position",           COMP_USER_INTENDED, EMAC_ENABLED,
    "trajectory",           COMP_CONTENTS, tra,
    "trajectory",           COMP_INTENDED, EMAC_DISABLED,
    "trajectory",           COMP_USER_INTENDED, EMAC_ENABLED,
    "ground speed",         COMP_CONTENTS, "",
    "ground speed",         COMP_INTENDED, EMAC_DISABLED,
    "ground speed",         COMP_USER_INTENDED, EMAC_ENABLED,
    "ascent rate",          COMP_CONTENTS, "",
    "ascent rate",          COMP_INTENDED, EMAC_DISABLED,
    EMAC_EOL);

Ema_Object_Attr_Set (model_id, wsn_node_objid,
    "ascent rate",          COMP_USER_INTENDED, EMAC_ENABLED,
    "color",                COMP_CONTENTS, 1090519039,
    "color",                COMP_INTENDED, EMAC_DISABLED,
    "color",                COMP_USER_INTENDED, EMAC_ENABLED,
    "doc file",             COMP_CONTENTS, "",
    "doc file",             COMP_INTENDED, EMAC_DISABLED,
    "doc file",             COMP_USER_INTENDED, EMAC_ENABLED,
```

图 9.设置节点属性

(5) 打开 OPNET 控制台窗口，进入修改过的***.em.c** 所在目录，分别执行如下命令，如图 10 所示。

① 输入命令 op_mkema -m 文件名＜不加后缀＞，执行成功后生成

.dev32.i0.em.x

② 执行刚创建的可执行文件 <文件名.***i0.em.x***>, 将生成新的模型文件(****.m***)

```
C:\Users\Administrator\op_models>op_mkema -m project_test-scenario

------------------------------------------------------------
Ema executable program (project_test-scenario.dev32.i0.em.x) produced.
------------------------------------------------------------

C:\Users\Administrator\op_models>project_test-scenario.dev32.i0.em.x
```

图 10.设置节点属性

(6) 在 OPNET 菜单中单击 **File->Model Files->Refresh Model Directories** 选项刷新模型目录，否则新创建的模型文件在已启动的 OPNET 中是不可见的。生成的网络模型，需要导入项目场景中才能看到，在项目编辑器中选择 **Scenarios->Scenario Components->import** 选项，得到的模型如下图所示，可发现节点大致均匀分布，并且关联了我们采用 SUMO 生成的真实道路轨迹，如图 10 所示。

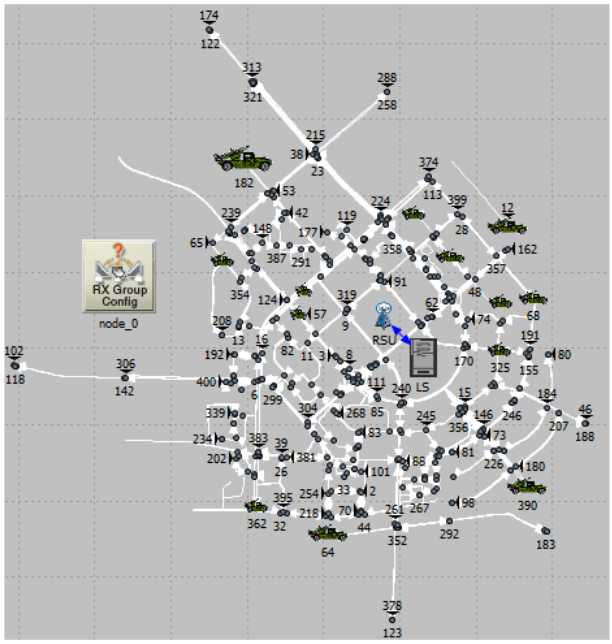

图 11.基于真实轨迹的车联网仿真