节点移动模型仿真程序 (C++ 语言版) 使用说明

May 19, 2019

1 基本情况

本节点移动模型仿真程序使用 C++ 语言编写，现已实现仿真四种节点移动模型——随机走动 (Random Walk) 模型、随机路点 (Random Waypoint) 移动模型、随机方向 (Random Direction) 移动模型、随机高斯-马尔科夫 (Random Gauss-Markov) 移动模型。

2 程序文件结构

main.cpp 主函数，用来调用 mobilitymodel() 函数;

mobilitymodel.h 节点移动模型头文件

mobilitymodel.cpp 节点移动模型函数, 根据参数不同可以使用不同的模型。

publicvar.h 公共变量

node.h 节点类头文件

node.cpp 节点类，四种模型的节点都继承于这个类，实现了一些最基本的操作

random_walk_node.h 随机走动模型节点类头文件

random_walk_node.cpp 随机走动模型节点类

random_waypoint_node.h 随机路点移动模型节点类头文件

random_waypoint_node.cpp 随机路点移动模型节点类

random_direction_node.h 随机方向移动模型节点类头文件

random_direction_node.cpp 随机方向移动模型节点类

gauss_markov_node.h 随机高斯-马尔科夫移动模型节点类头文件

gauss_markov_node.cpp 随机高斯-马尔科夫移动模型节点类

glm OpenGL Mathematics(GLM) 库，用到了它提供的 vec3，即三维向量

3 使用说明

使用方法主要就是调用 mobilitymodel() 函数，如同主函数中的例子所示。

在 mobilitymodel.h 中声明了三个重载函数，如下：

- void mobilitymodel(string str, int nodes, double xmin, double xmax, double ymin, double ymax, double zmin, double zmax, double vmin, double vmax, double m_interval=2.0, bool isGeo = true);
- void mobilitymodel(string str, int nodes, double xmin, double xmax, double ymin, double ymax, double zmin, double zmax, double vmin, double vmax, double m_vmean, double m_dmean, double m_pmean, double m_alpha, bool isGeo = true);
- void mobilitymodel(...);

第一个参数 str 用来指定选用的节点移动模型，有四种选项：

str	含义
“rw”	随机走动模型
“rwp”	随机路点移动模型
“rd”	随机方向移动模型
“gm”	随机高斯-马尔科夫移动模型

第二个参数 nodes 表示仿真节点的数目；

上述函数中，xmin, xmax, ymin, ymax, zmin, zmax, vmin, vmax 均分别定义了仿真节点运动区域的位置范围和速度范围。

注意，当最后一个参数 isGeo 为真（默认就是真）时，表示前面输入的节点位置范围的参数均为经纬高坐标（这里使用的是 WGS84 坐标，如有需要可修改参数，在 node::calc() 函数中），则 xmin, xmax, ymin, ymax, zmin, zmax 分别表示经度的最小值和最大值、纬度的最小值和最大值、高度的最小值和最大值。

m_interval 在不同模型中有不同的含义，在随机走动模型中表示节点更新速度和方向的时间间隔，在随机方向移动模型中表示节点更新运动方向的时间间隔，在随机路点移动模型中表示节点运动到目标地点后停留的时间，在高斯-马尔科夫模型则没有此参数。默认值为 2.0 秒。

m_vmean、m_dmean、m_pmean、m_alpha 为随机高斯-马尔科夫移动模型的特有参数，分别表示速度均值、方位角均值、仰角均值和 α。

最后一个重载函数相当于默认函数，当输入参数有误时会执行这一个函数，报错并退出。

4 数据输出

程序主要维护以下两个数组（指针）变量：

```
extern glm::vec3* points; //存储笛卡尔坐标表示的所有节点位置
extern glm::vec3* points_wgs84; //存储 WGS84 坐标表示的所有节点位置
```

程序写成了多线程程序，每一个线程对应一个节点和一个编号，编号从 0 开始到 n-1，n 为节点数目。节点移动模型执行时会改变这两个数组中相应位置的数据，达到更新数据的目的。这两个数组被声明在 publicvar.h 中，在 mobilitymodel.cpp 中被初始化，可以通过访问这两个数组来实时获取所有节点的位置信息。

另外为了编写时调试方便，在 mobilitymodel 中定义了一个额外的线程 pointsDisplayer，用来每隔一秒输出编号为 0 的节点的位置信息。

5 其他说明

- 本程序编写时使用的是 Linux，因此在代码开头中可以看到 #include <unistd.h>，如果在 Windows 下编译，则需要替换为 #include <win-

dows.h>，请留意。

- 程序中个别部分的代码使用了 Lambda 语句，因此最好在支持 C++11 以上的编译器中编译，否则还需要一些更改。
- 为了使用三维向量 vec3 而使用了 GLM 库，这里简单介绍一下 vec3 的读取方法，下面的示例代码可以输出一个 vec3 三维向量的每一维数据：

```
glm::vec3 example;
cout<<example.x<<endl;
cout<<example.y<<endl;
cout<<example.z<<endl;
```

- GLM 库的官方网站：https://glm.g-truc.net
- 作者：武尚玮　微信：heartofrainbow