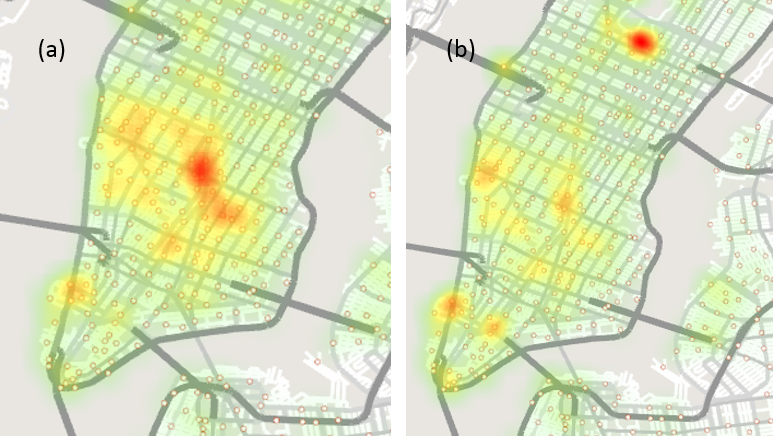
实习3 空间索引编程

**实习目的：**了解空间数据类型层次结构，熟悉包围盒Envelope在空间查询中作用，熟悉常见的空间计算方法，掌握Z-Curve、Hilbert Curve和四叉树的创建，掌握四叉树的区域查询和最邻近查询方法，理解空间数据查询的过滤和精炼步。

**实习内容：**

纽约市的自行车站点和出租车上车地点，如下图为自行车站点在休息日租车数量对应的热力图。



(a) 个体休息日租车热力图 (b) 群体(>2人)休息日租车热力图

数据在data目录下，程序代码包含以下文件：

|  |  |
| --- | --- |
| 文件名 | 说明 |
| Shapelib | Shapefile文件读取类库，包含shapefil.h, shpopen.cpp, safileio.cpp和dbfopen.cpp，无需修改任何代码 |
| Common.h | 符号定义，无需修改任何代码 |
| Curve.h/cpp | Z-Curve和Hilbert Curve构建，需要修改从zorder函数代码（存在错误）和实现izorder函数代码，实现二维坐标与Z值的相互转换，需要补全horder和ihorder函数代码，实现二维坐标与H值的相互转换 |
| Geometry.h/cpp | Envelope, Geometry, Point, LineString, Polygon类，需要实现Envelope空间关系判断代码，补全Point和LineString、Point和Polygon距离计算代码 |
| QuadTree.h/cpp | Feature类封装Geometry类，QuadNode四叉树节点类，QuadTree四叉树类，需要实现QuadNode和QuadTree的创建和查询代码 |
| KMeans.h/cpp | Cluster和KMeans类 |
| Lab3.cpp | 界面和数据读入等函数，无需修改任何代码 |
| Test.cpp | 测试代码 |

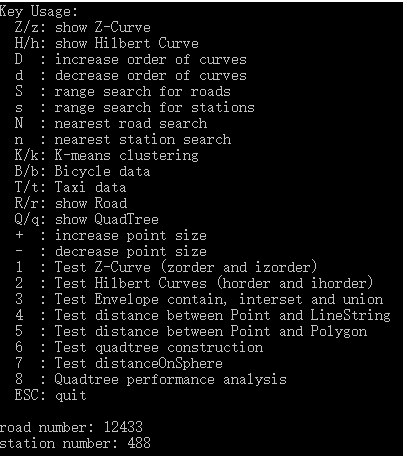
绘制使用OpenGL和freeglut(http://www.transmissionzero.co.uk/software/freeglut-devel/)，根据帮助文档将类库文件拷贝到相应的目录，Windows建议使用Visual Studio 2012及以上版本。建议先浏览每个类的功能和已有的函数，再填充相应函数的C++代码，代码写在“Write your code here”处，其他代码无需修改。实习总分50分，其中1-7是必做题，8-14是选做题，选做题最多加10分。

1. 实现Z-curve曲线的二维坐标与Z值的相互转换(Curve.cpp，5分)

void zorder(int order, int& value, int coor[2])

void izorder(int order, int value, int coor[2])

zorder函数代码存在错误，基于bit-shuffling思想，修改代码，完成从二维坐标(coor)到Z值(value)的计算，实现izorder函数实现从Z值(value)到二维坐标(coor)的转换。order为Z-Curve的阶数，order为4时，平面网格大小为(2^4, 2^4)，即16x16。



2. 实现Hilbert Curve的二维坐标与H值的相互转换(Curve.cpp，5分)

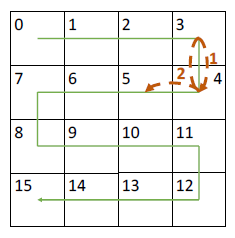
void horder(int order, int& value, int coor[2])

void ihorder(int order, int value, int coor[2])

基于linear-quadtrees递归填充思想，补全horder和ihorder函数代码，实现二维坐标(coor)到H值(value)的相互转换。order为Hilbert Curve的阶数，order为4时，平面网格大小为(2^4, 2^4)，即16x16。

zdist和hdist函数实现了二维空间填充曲线的pair-distance，其定义为：对于每个位置pi，考虑其曲线上接下来的pi+1和pi+2，两点pi=(xi, yi)和pi+1=(xi+1, yi+1)的曼哈顿顿距离为d(pi, pi+1)=| xi - xi+1| + | yi - yi+1|，包含n个点的曲线pair-distance-sum定义为

忽略最后两个点。下图节点3的pair-distance为1+2=3。



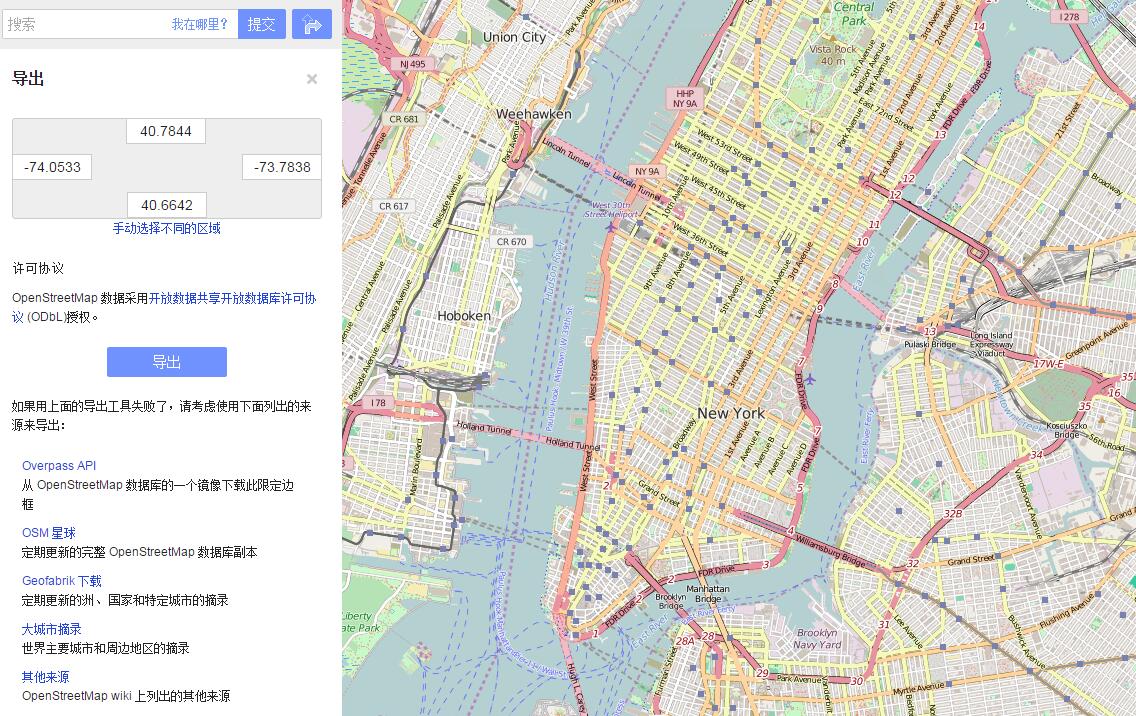
3. 包围盒空间关系判断(Geometry.cpp，6分)

bool Envelope::contain(const Envelope& envelope) const

bool Envelope::intersect(const Envelope& envelope) const

Envelope Envelope::unionEnvelope(const Envelope& envelope) const

空间索引是基于几何特征的包围盒创建，首先通过判断几何特征的包围盒是否和查询区域相交，降低复杂的几何特征空间关系计算的次数。contain含义和PostGIS中的contain不同，可以是相同的包围盒。



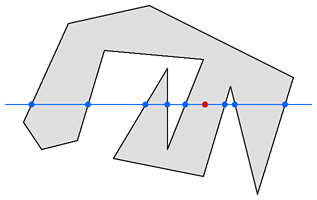
4. Point到LineString和Polygon距离计算(Geometry.cpp，6分)

double Point::distance(const LineString\* line) const

double Point::distance(const Polygon\* polygon) const

Point到LineString的距离计算分解为Point到每个线段的距离计算，即点P(x, y)到线段[P1(x1, y1), P2(y2, y2)]在二维笛卡尔空间的最短距离。基本思路是通过(P2-P1)归一化向量和(P-P1)向量的内积，计算P在直线上的投影点，判断该投影点是否在线段上，如果在线段上，计算投影点到P的距离，不在线段上，计算P到线段端点距离的最小值。

Point到Polygon的距离计算关键是判断Point是否在Polygon内部，可以通过射线法判断，参考<https://www.cnblogs.com/luxiaoxun/p/3722358.html>，如果Point在Polygon内部，距离为0，否则计算Point与Polygon边界距离。Polygon定义与PostGIS相同，外环第一个点和最后一个点为同一个点。



5. 四叉树构建(QuadTree.cpp，8分)

bool QuadTree::constructQuadTree(vector<Feature>& features)

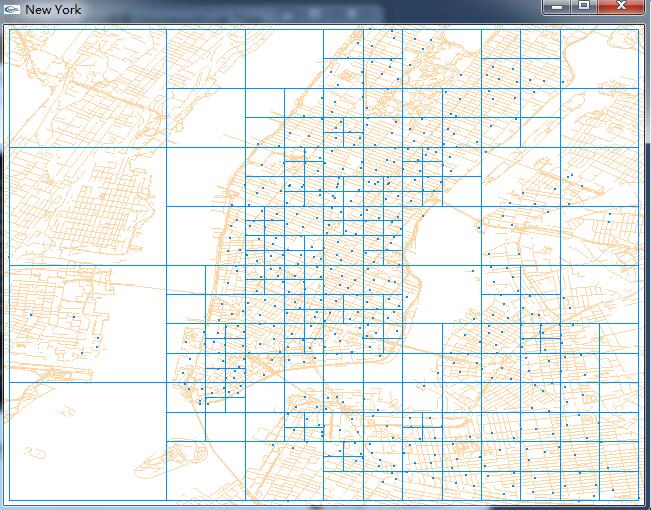
void QuadNode::split(size\_t capacity)

四叉树创建输入一组几何特征，将节点分裂为四个子节点，

每个特征加到包围盒重叠的子节点中（即一个特征可能在多个节点中），

删除当前节点的几何特征记录（即所有特征只存储在叶节点中），

如果子节点的几何特征个数大于capacity，递归分裂子节点。



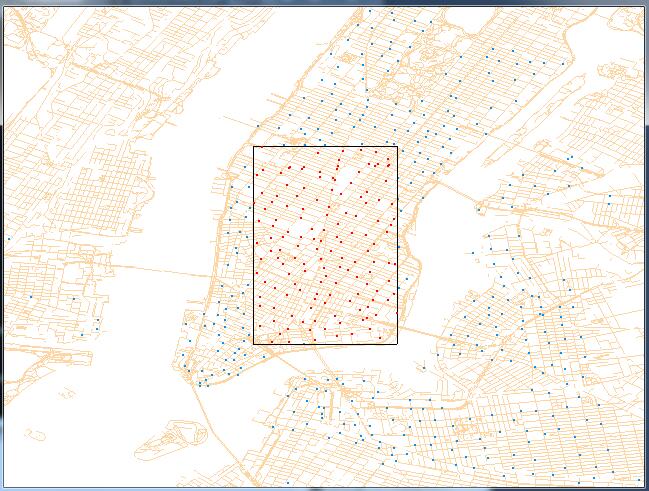
6. 基于四叉树的区域查询(QuadTree.cpp，10分)

void QuadNode::rangeQuery(Envelope& rect, vector<Feature>& features)

void QuadTree::rangeQuery(Envelope& rect, vector<Feature>& features)

区域查询输入区域rect，查询与区域rect相交的几何特征，存储在features。区域rect如果与当前节点的包围盒bbox相交，递归遍历四叉树，查询哪些几何特征的包围盒和查询区域相交（filter）；再获得可能和查询区域相交的候选几何特征后，精确判断几何特征是否与查询区域相交（refine）。

通过鼠标选择查询区域，在站点和道路数据上，验证区域查询。



7. 基于四叉树的最邻近几何特征查询(QuadTree.cpp，10分)

bool QuadTree::NNQuery(double x, double y, Feature& feature)

QuadNode\* QuadNode::pointInLeafNode(double x, double y)

最邻近几何特征查询(K-NN)输入查询点(x, y)，返回与该点最邻近的几何特征，存储在feature。

首先，通过pointInLeafNode查询点(x, y)所在的叶节点，

计算查询点(x, y)与该叶节点内的几何特征包围盒的最大距离的最小值minDist，即通过包围盒而非原始几何加速最小距离计算；

然后，构造查询区域 (x – minDist, x + minDist, y – minDist, y + minDist)，查询几何特征的包围盒与该区域相交的几何特征（filter），再查询与查询点(x, y)距离最近的几何特征（refine）。

通过鼠标移动选择离鼠标最近的几何特征（站点和道路），验证最邻近几何特征查询。



**选做题**

8. 实现点与点之间的球面距离计算(Geometry.cpp，2分)

double Point::distanceOnSphere(const Point\* point) const

球面距离计算公式参考http://tech.meituan.com/lucene-distance.html

9. 四叉树性能分析(test.cpp，4分)

void QuadTreeAnalysis()

使用纽约taxi上车点数据分析四叉树性能。四叉树的性能决定于参数capacity，即每个叶节点最多的几何特征数目，数值太多，每个叶节点判断次数太大，数值太小，树的层次太高。当capacity在[70, 200]时，计算四叉树的高度和叶节点数目，评估100000次随机最邻近几何特征查询的时间。分析不同参数下的性能，可以得出什么结论？

10. 实现Line-Line和Line-Polygon的欧式距离计算(Geometry.cpp，4分)

double LineString::distance(const LineString\* line) const

double LineString::distance(const Polygon\* polygon) const

11. 实现Polygon与Envelope相交判断(Geometry.cpp, 3分)

bool Polygon::intersects(const Envelope& rect) const

12. 实现Polygon的内环几何数据存储，并修改欧式距离计算代码(Geometry.cpp, 4分)

13. 扩展几何类型，增加MultiPoint, MultiLineString, 和MultiPolygon类的数据存储和欧式距离计算(Geometry.cpp，12分，每个类4分)。

14. 基于空间位置聚类(KMeans.cpp，4分)

void KMeans::cluster(vector<Feature>& features, int k, int maxIterNum)

bool Cluster::resetMean()

聚类输入几何特征集合，聚类个数和最大迭代次数，其中resetMean根据Cluster中的几何特征集合，计算平均空间位置。KMeans算法参考http://baike.baidu.com/view/3066906.htm

**实习提交内容：**

1. 实习代码，删除不必要的文件，如lab3.sdf、data、Debug和Release目录，压缩后在1M以内。

2. 实习报告，第1-5题的测试截图，第5-7题的运行效果截图，实现了哪些选做题，相应的截图等内容。