# 地图匹配PJ报告

杨子旭 21307130105

1. 实现思路

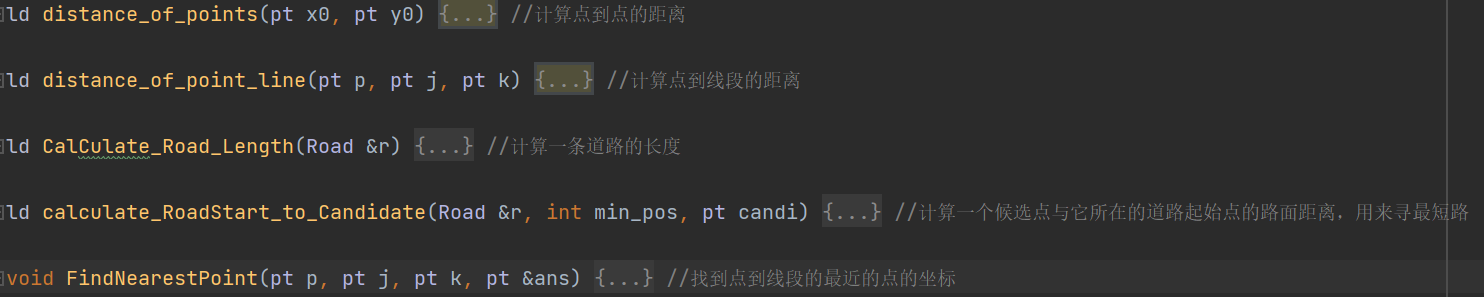
用网格化地图的方式存储路网，同时建立每个节点与有向路的联系。对于一条轨迹中的每一个轨迹点，单独的寻找它50m 内的所有路段中最近的所有点作为候选点集，以距离计算出观测概率，以两个候选点的直线距离除以最短连通路长度作为传递概率。最后用隐马尔可夫模型，对每一组候选点集计算出其当前的路径概率最大值，并记录前驱节点，直到最后一个候选点组，选择最大的概率对应的点，逆推得到路径。

1. 实现细节
2. 路网拓扑的数据结构

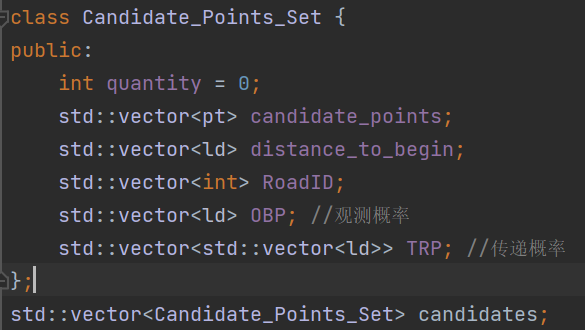
以五十米为小网格的长和宽，使用std::map<pii, std::set<int>> Grids来存储每个小网格包含的路段编号，用一个vector<Road>容器来将路段编号对应到具体的路段信息上，Road定义为一个类，包含了路段的编号、起始和终点的点编号和路段的GPS位置等信息。

使用std::set<int> Node[60010]记录每一个节点它作为路段起始点出发的道路的编号，以便于寻路算法使用。

1. 几何模型的计算函数



1. 候选点的数据结构

对于每一个候选点的位置、路段相对位置、路段编号、观测概率、传递概率的存储：

用轨迹点的编号作为candidates的下标就可以访问到这个轨迹点对应的所有候选点的信息。

1. 寻找候选点

具体寻找方法是根据轨迹点所在的网格，罗列这个网格周围3 x 3 的网格内的所有路段，每个路段计算出一个距离最近且小于50米的候选点，根据距离先计算出观测概率，将所有相关的信息先存储进std::vector<Candidate\_Points\_Set> candidates;

调试过程中发现极少数情况assert(候选点集的大小>0)会assert failed，推测某个地方道路特别稀疏，于是补充了若发现候选点太少则扩大搜索半径为75米搜索5 x 5 的网格。

曾将第一个候选点和其他候选点分开来寻找，是因为尝试在初步筛选的时候考虑轨迹点与道路的方向因素然后舍弃反向路段，第一个候选点不用考虑方向。但是在不断的参数调试中，发现只要存在这个操作，不管反向删除的程度是多少，都会大幅降低准确度，所以这两个函数可以被合并。

1. 计算传递概率

使用数据结构std::set<int> node[60010] 记录从每个节点出发的路的编号

从一个候选点所在的道路起点出发，用广度优先搜索算法寻找三层的深度，如果找到下一个候选点所在的道路了，那么更新这两个候选点间对应的最短路。

将两个候选点的距离（而不是两个轨迹点），除以上面算得的最短路得到传递概率。

调试的时候发现大部分符合条件的候选点组合传递概率都比较接近1，符合预期。尝试增加BFS的深度，但作用不大。

1. 隐马尔可夫模型计算答案

对于每一个轨迹点对应的候选点组中的每一个候选点，计算出它与前一个候选点组最有可能建立联系的点，把这个前驱节点记录入pre[][] ，把由观测概率和传递概率综合得到的这条线路独有的概率存入f[][]。最后选出概率最大的点，从后往前逆推出线路。

1. 尝试过的优化

（一） 从用结构体和二维数组存储转向类和vector、map、set容器存储，空间效率显著提升，原来超内存，现在只需100MiB以下。

（二）寻路算法从Dijkstra更换为BFS，时间效率显著提升，原来40分钟才能跑完1000条数据，现在只需3秒钟。

（三）用观测概率和传递概率计算候选点的叠加概率时使用乘法而不是加法，准确度有提升4%。

（四）查找候选点集时尽可能地预处理减少不必要的候选点数量，比如排除重复的节点，根据方向初筛等，候选点数量的减少能够提升速度近一倍，但是准确度随之降低10%~20%。

（五）调参

观测概率的计算方法，传递概率的计算方法，最终合成的计算方法，网格的大小，搜索半径等等都经过了调试，提升准确度约为5%。

(六）使用速度数据集，但是效果上不理想，推测是因为道路不确定因素多，速度随机性大。

最终的最高准确度是96.4%。我尝试从错误较多的路径寻找匹配错误的原因，但是错误较为分散，超出了我的能力范围。