深蓝学院
shenlanxueyuan.com

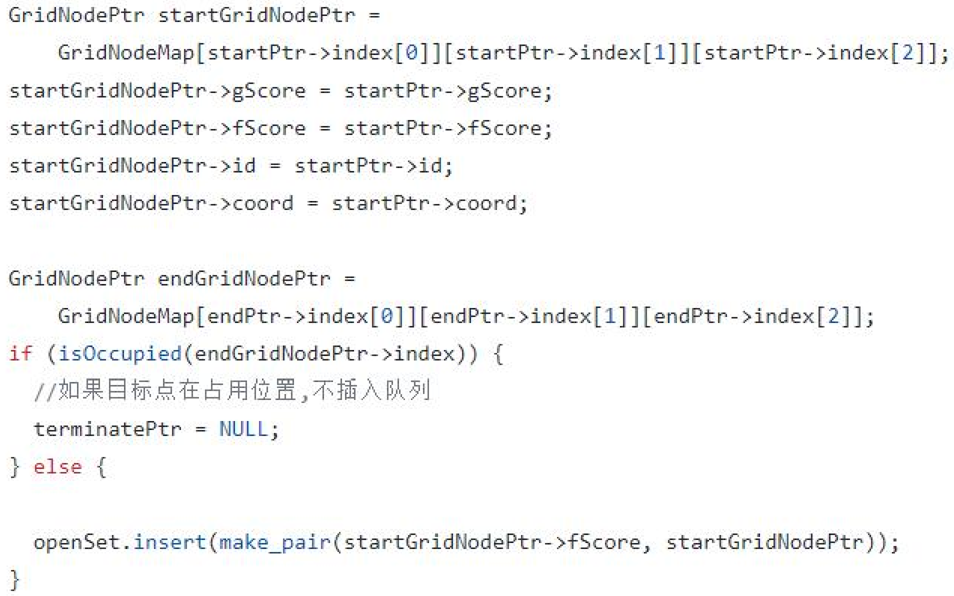
第二章作业思路

主讲人 刘武

纲要

- 作业思路
- 实验环境
- 实验验证

作业思路

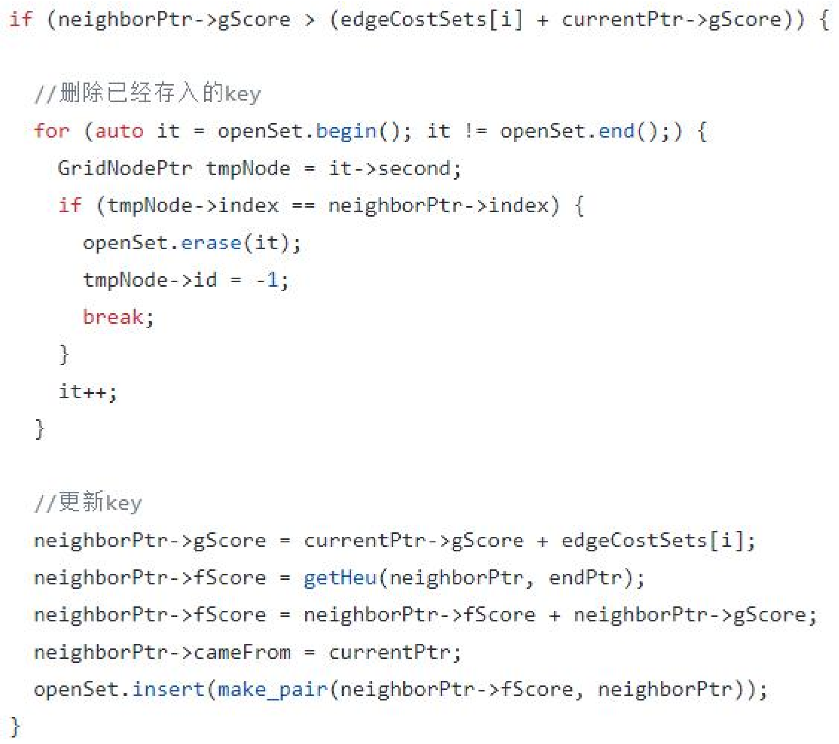


```
GridNodePtr startGridNodePtr =
    GridNodeMap[startPtr->index[0]][startPtr->index[1]][startPtr->index[2]];
startGridNodePtr->gScore = startPtr->gScore;
startGridNodePtr->fScore = startPtr->fScore;
startGridNodePtr->id = startPtr->id;
startGridNodePtr->coord = startPtr->coord;

GridNodePtr endGridNodePtr =
    GridNodeMap[endPtr->index[0]][endPtr->index[1]][endPtr->index[2]];
if (isOccupied(endGridNodePtr->index)) {
  //如果目标点在占用位置,不插入队列
  terminatePtr = NULL;
} else {

  openSet.insert(make_pair(startGridNodePtr->fScore, startGridNodePtr));
}
```

更新GridNodePtr中起始节点的信息

校验目标Node是否被占用

作业思路

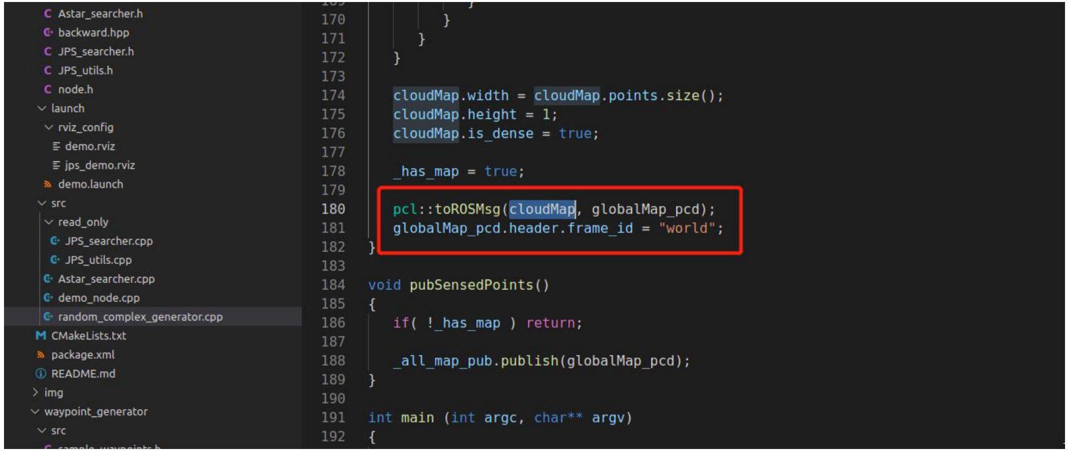


```
if (neighborPtr->gScore > (edgeCostSets[i] + currentPtr->gScore)) {

  //删除已经存入的key
  for (auto it = openSet.begin(); it != openSet.end();) {
    GridNodePtr tmpNode = it->second;
    if (tmpNode->index == neighborPtr->index) {
      openSet.erase(it);
      tmpNode->id = -1;
      break;
    }
    it++;
  }

  //更新key
  neighborPtr->gScore = currentPtr->gScore + edgeCostSets[i];
  neighborPtr->fScore = getHeu(neighborPtr, endPtr);
  neighborPtr->fScore = neighborPtr->fScore + neighborPtr->gScore;
  neighborPtr->cameFrom = currentPtr;
  openSet.insert(make_pair(neighborPtr->fScore, neighborPtr));
}
```

https://cplusplus.com/reference/map/

实验环境

将这次生成点云数据保存下来后，修改代码，以后直接加载固定的点云数据

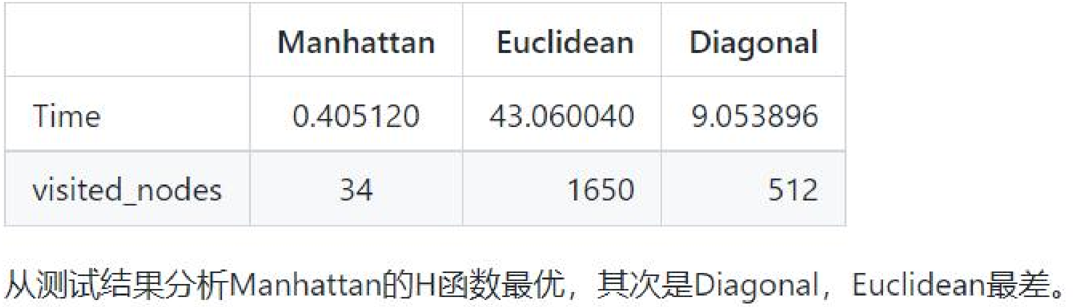


```
                }
            }
        }
    }

    cloudMap.width = cloudMap.points.size();
    cloudMap.height = 1;
    cloudMap.is_dense = true;

    _has_map = true;

    pcl::toROSMsg(cloudMap, globalMap_pcd);
    globalMap_pcd.header.frame_id = "world";
}

void pubSensedPoints()
{
    if( !_has_map ) return;

    _all_map_pub.publish(globalMap_pcd);
}

int main (int argc, char** argv)
{
```

```
rostopic pub -1 /goal
geometry_msgs/PoseStamped "header:
  seq: 0
  stamp:
    secs: 0
    nsecs: 0
  frame_id: ''
pose:
  position:
    x: 4.5936794281
    y: -4.07747602463
    z: 0.0
  orientation:
    x: 0.0
    y: 0.0
    z: 0.0
    w: 0.0"
```

实验验证

对比不同启发式函数（Manhattan、Euclidean、Diagonal Heuristic）对A*运行效率的影响

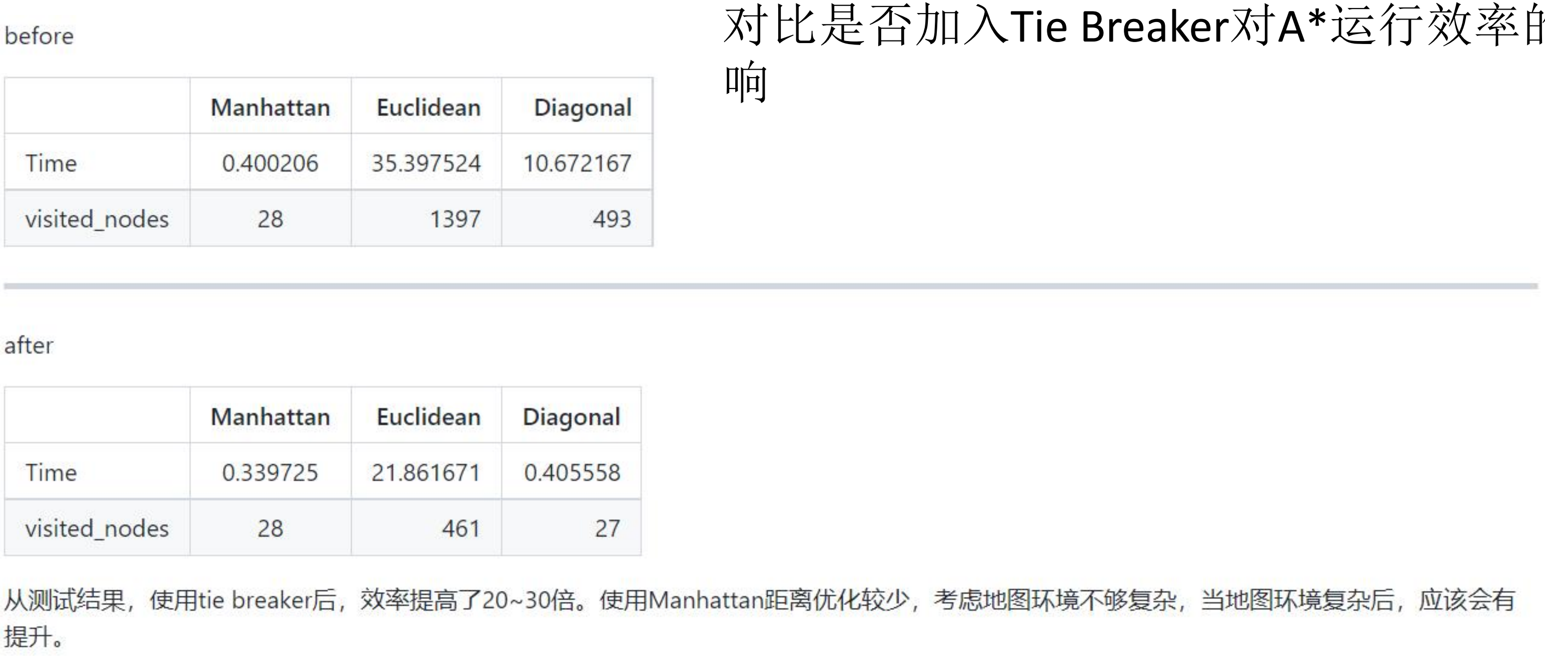


	Manhattan	Euclidean	Diagonal
Time	0.405120	43.060040	9.053896
visited_nodes	34	1650	512

从测试结果分析Manhattan的H函数最优，其次是Diagonal，Euclidean最差。

实验验证

对比是否加入Tie Breaker对A*运行效率的影响



before

	Manhattan	Euclidean	Diagonal
Time	0.400206	35.397524	10.672167
visited_nodes	28	1397	493

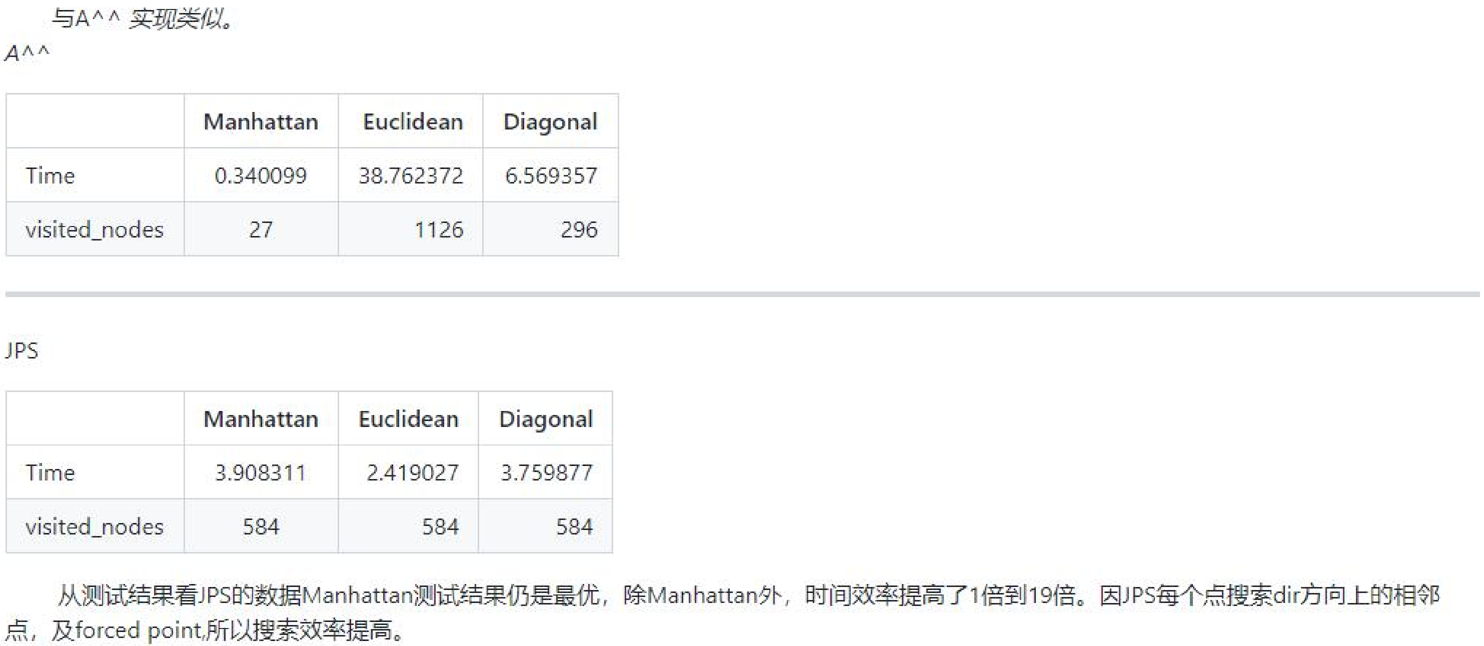
after

	Manhattan	Euclidean	Diagonal
Time	0.339725	21.861671	0.405558
visited_nodes	28	461	27

从测试结果，使用tie breaker后，效率提高了20~30倍。使用Manhattan距离优化较少，考虑地图环境不够复杂，当地图环境复杂后，应该会有提升。

实验验证

A*和JPS算法效率



与A^^ 实现类似。

*A^^*

	Manhattan	Euclidean	Diagonal
Time	0.340099	38.762372	6.569357
visited_nodes	27	1126	296

JPS

	Manhattan	Euclidean	Diagonal
Time	3.908311	2.419027	3.759877
visited_nodes	584	584	584

从测试结果看JPS的数据Manhattan测试结果仍是最优，除Manhattan外，时间效率提高了1倍到19倍。因JPS每个点搜索dir方向上的相邻点，及forced point,所以搜索效率提高。

在线问答

Q&A

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening