深蓝学院
shenlanxueyuan.com

第二章作业思路分享

主讲人 郭定

作业思路

算法流程

- 创建一个优先级队列存储所有要被扩展的节点
- 设计一个启发式函数 H(n)计算节点到目标点的预测 cost
- 将初始节点放入优先级队列，设置初始节点的 G(s)为 0
- loop
  - 如果优先级队列为空，程序异常，退出
  - 移出优先级队列中 F(n) = H(n) + G(n)最小的节点 n 作为扩展节点
  - 如果 n 为目标节点 goal，运行成功，退出
  - 计算所有节点 n 的邻居节点。
  - 对于所有邻居节点 m
  - 如果 m 是一个新节点
    - 计算H(n)， G(m) = G(n) + Cnm ， F(m) = H(m) + G(m)
    - 将 m 存入优先级队列
  - 如果 m 已经在优先级队列中并且 G(m) > G(n) + C nm
    - G(m) = G(n) + Cnm
- end

step 1

深蓝学院
shenlanxueyuan.com

```
double AstarPathFinder::getHeu(GridNodePtr node1, GridNodePtr node2)
{
    /*
    choose possible heuristic function you want
    Manhattan, Euclidean, Diagonal, or 0 (Dijkstra)
    Remember tie_breaker learned in lecture, add it here ?
    *
    *
    *
    STEP 1: finish the AstarPathFinder::getHeu , which is the heuristic function
    please write your code below
    *
    *
    */

    return 0;
}
```

STEP 1：这一步是计算节点node1到目标点node2的启发式函数的值。

先算dx dy dz。

若选择Manhattan：

```
hScore = dx + dy + dz
```

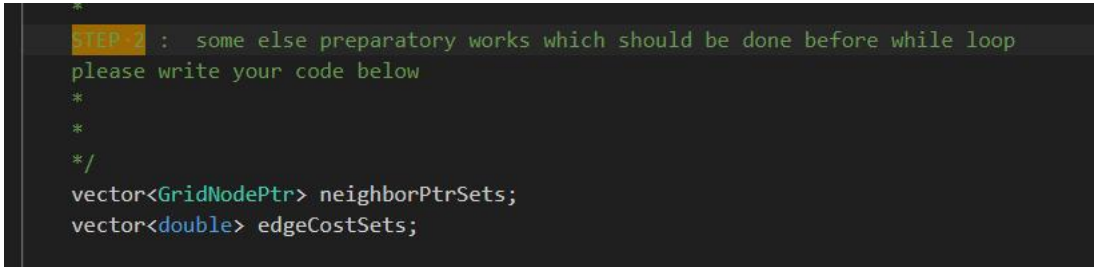
若选择Euclidean：

```
hScore = std::sqrt(std::pow(dx,2) +
std::pow(dy,2) + std::pow(dz,2))
```

若选择Diagonal Heuristic：

```
double min_xyz = std::min({dx, dy, dz});
hScore = dx + dy + dz + (std::sqrt(3.0)
-3) * min_xyz;
```

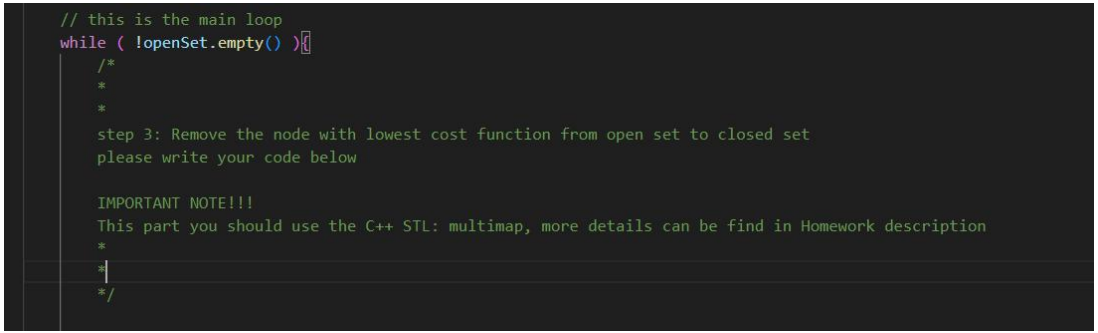
step 2



STEP 2：这一步需要熟悉整个算法的流程中用到的变量。有什么步骤需要在循环前运行。

将GridNodeMap中的初始节点对象指向startPtr。
GridNodeMap[start_idx[0]][start_idx[1]][start_idx[2]] = startPtr

step 3



```
// this is the main loop
while ( !openSet.empty() ){
    /*
    *
    *
    step 3: Remove the node with lowest cost function from open set to closed set
    please write your code below

    IMPORTANT NOTE!!!
    This part you should use the C++ STL: multimap, more details can be find in Homework description
    *
    *
    */
```

STEP 3：取出优先级队列里面cost最小的节点。

```
currentPtr = openSet.begin()->second
openSet.erase(openSet.begin())
```

同时取出后的节点要归入closed 队列

```
currentPtr->id = -1
```

step 4

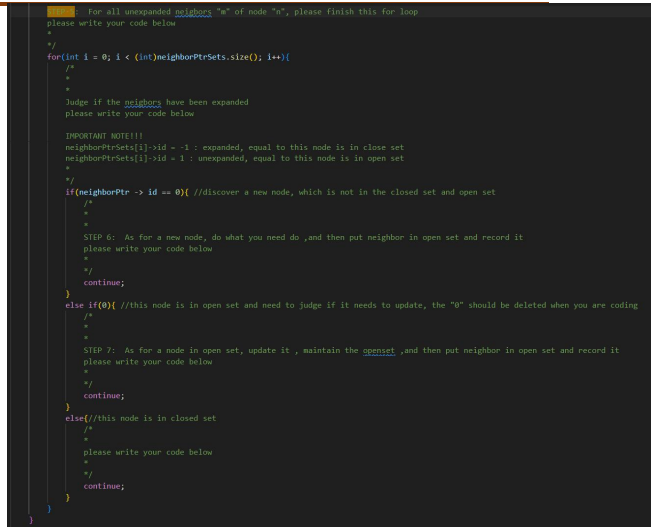
深蓝学院 shenlanxueyuan.com

```
inline void AstarPathFinder::AstarGetSucc(GridNodePtr currentPtr, vector<GridNodePtr> & neighborPtrSets, vector<double> & edgeCostSets)
{
    neighborPtrSets.clear();
    edgeCostSets.clear();
    /*
    *
    STEP 4: finish AstarPathFinder::AstarGetSucc yourself
    please write your code below
    *
    *
    */
}
```

STEP 4：获取currentPtr节点的邻居节点，及其edgeCost，也就是算法流程里的Cnm

```
for(int i=x-1;i<=x+1;i++){
    for(int j=y-1;j<=y+1;j++){
        for (int k=z-1;k<=z+1;k++){
            #此处要考虑细节删除一些节点，包括超出地图的，在障碍物中的以及自身。
            .....
            tempPtr = GridNodeMap[i][j][k];
            ........
           edgeScore = std::sqrt(std::pow(dx,2) + std::pow(dy,2) + std::pow(dz,2));
           neighborPtrSets.push_back(tempPtr);
           edgeCostSets.push_back(edgeScore);
}
```

step 5 6 7

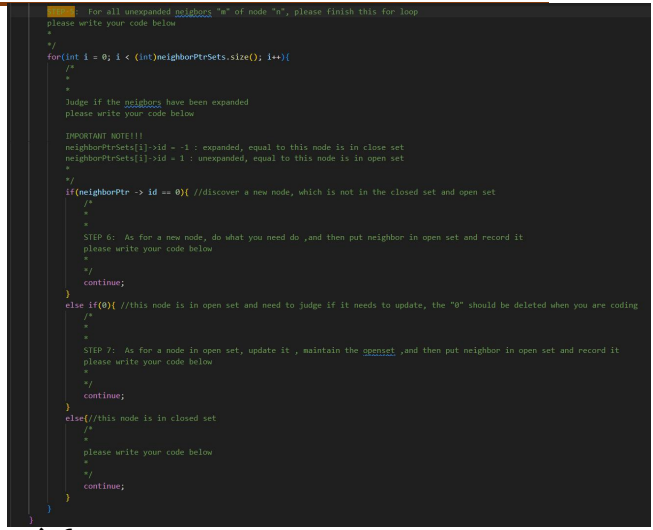


Step5 6 7处理每一个邻居节点。按照算法流程，
只需处理两种情况。
第一种是邻居节点m是新节点（对应Step 6）

```
neighborPtr->cameFrom = currentPtr;
neighborPtr->gScore = currentPtr->gScore + edgeCostSets[i];
neighborPtr->fScore = neighborPtr->gScore +
getHeu(neighborPtr,endPtr);
neighborPtr->id = 1;
openSet.insert(make_pair(neighborPtr->fScore,neighborPtr))
```

step 5 6 7

深蓝学院
shenlanxueyuan.com



Step5 6 7处理每一个邻居节点。按照算法流程，
只需处理两种情况。
第二种是邻居节点m已经在openset,并且新计算的gScore
比原来小（对应Step 7），则更新gScore以及fScore。

```
if(currentPtr->gScore + edgeCostSets[i] < neighborPtr->gScore){
    neighborPtr->cameFrom = currentPtr;
    neighborPtr->gScore = currentPtr->gScore + edgeCostSets[i];
    neighborPtr->fScore = neighborPtr->gScore + getHeu(neighborPtr,endPtr);
}
```

step 8

深蓝学院
shenlanxueyuan.com

```
vector<Vector3d> AstarPathFinder::getPath()
{
    vector<Vector3d> path;
    vector<GridNodePtr> gridPath;
    /*
    *
    *
    STEP 8:  trace back from the curretnt nodePtr to get all nodes along the path
    please write your code below
    *
    */

    for (auto ptr: gridPath)
        path.push_back(ptr->coord);

    reverse(path.begin(),path.end());

    return path;
}
```

Step 8 返回path
将终止节点的cameFrom一直连起来形成一条路径并返回。

```
GridNodePtr temPtr = terminatePtr;
while (temPtr->cameFrom != nullptr){
   gridPath.push_back(temPtr);
   temPtr = temPtr->cameFrom;
}
```

深蓝学院
shenlanxueyuan.com

感谢各位聆听！

Thanks for Listening