第二章作业（ROS 版本）

1. 算法流程

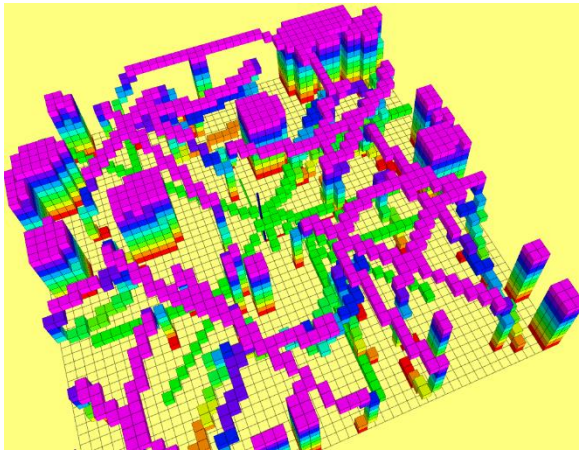
1.若现有点不为终点，则将该点从 openset 里删除，并加入 closedset

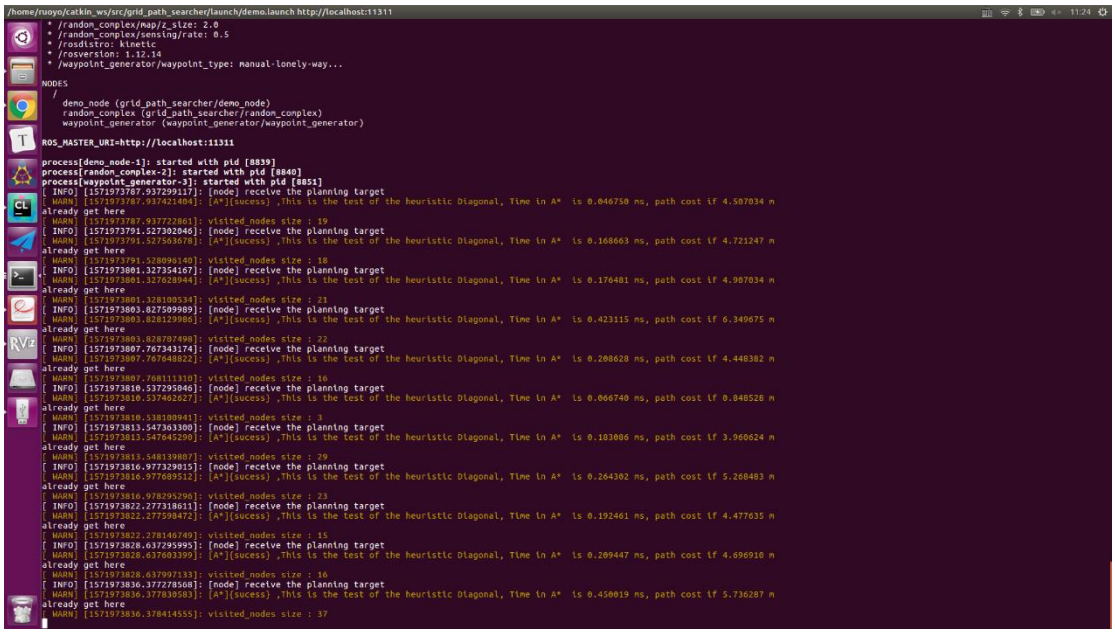
2.对该点周围 26 个点进行扩展，若未扩展过，则加入 openset；若扩展过了，则判断若现有点的 gn 小于之前的 gn，则覆盖原有信息，重新构建 nodeMaplt 的映射关系。

3.查找到终点后，利用 restrieve 函数通过 cameFrom 父节点指针信息依次将点加入到 gridPath 中

4.getPath 函数将 gridPath 中的各指针对应的点的 coord 坐标录入到 grid 数组内，并返回给 demo_node.cpp 中去，由此获得路径。

2. 运行结果：






3. 不同启发式函数对 A*运行效率的影响（均带有 tie_breaker）：

- Diagonal：平均用时：0.1117ms，平均 path_cost：3.685m 速度：32.99m/ms
- Manhattan:平均用时：0.2271ms， 平均 path_cost: 4.827m 速度：21.25m/ms
- Euclidean: 平均用时：0.3435ms，平均 path_cost: 5.195m 速度：15.12m/ms

由此可见该程序效率：diagonal>manhattan>Euclidean

4. 对比是否加入 tie_breaker 对 A*效率的影响
diagonal 未加入 tie_breaker 时：
平均用时：0.2068ms ，平均 path_cost：5.54m 速度：26.79m/ms

与问题 3 中的结果对比，可发现加入了 tie_breaker 的 diagonal 比未加入 tie_breaker 的效率更高

4.遇到的问题：

1.作为一个小白，c++有很多东西没有接触过，例如 Eigen 库、make_pair、vector、push_back、*it->first 之类的用法都不明白，因此花了很多时间来看懂给的代码。

2.ros 上运行程序出现死循环时，ubuntu 卡死了，难以进行检查。同时修改代码后重新运行所花费的时间很长。

3.不过同时，通过利用 multimap 自动排序和 vector 的函数，能够使得代码的编写易于 matlab 版本的代码编写。