

zqchen -第 2 章作业

MATLAB 版本

C++ 版本

在 WSL2 中出现的出现的找不到头文件的问题

DELL-Ubuntu : 把新的三个功能包拷贝到新的工作空间

最终效果截图演示

保存障碍物地图到 .pcd 文件中

算法运行效率的比较以及是否加入 Tie Breaker 产生的影响

A* 算法采用 Euclidean 和 Diagonal 的效率对比

实验1: (25, 25, 0) ----> (29, 25, 0)

实验2: (25, 25, 0) ----> (49, 32, 0)

加入 Tie Breaker 对效率的影响

实验3: Euclidean + p=0.001

实验4: Diagonal + p=0.001

实验5: Diagonal + h*(1 + p=0.001/0.005) 或 Diagonal + h +

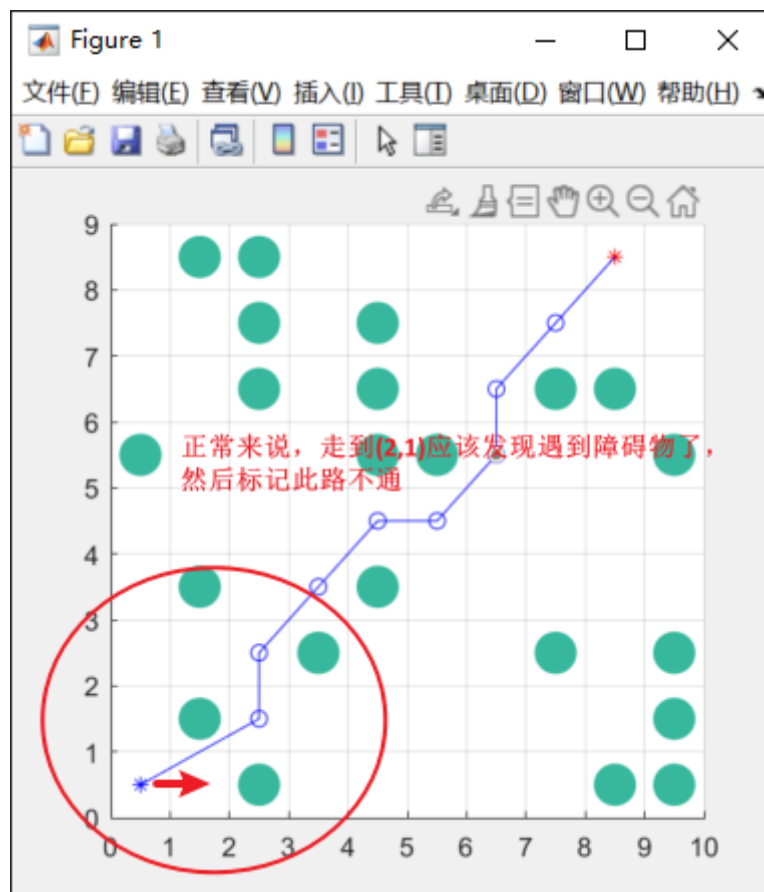
0.001*dot_product

对算法的理解和编程实现上的心得体会

zqchen -第 2 章作业

MATLAB 版本

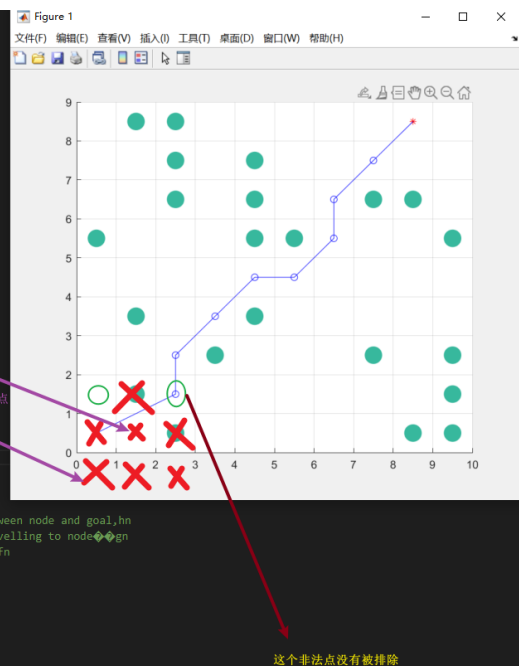
最开始 2022-03-22 周二开始做 MATLAB 版本的 A-star 算法，最后花了 5 个多小时调试成功，但是发现了有一个作业内置函数实现的问题，这个问题我尝试修复了好几个小时仍然没有成功，时间紧张最后就没有继续深究，暂时知道这个问题是存在的，以后有时间再试一下。



```

A_star > if expand_array.m > expand_array.m > {k > 0} > {i > 1} > {c1 > 1} > {flag}
1 function exp_array=expand_array(node_x,node_y,gn,xTarget,yTarget,CLOSED,MAX_X,MAX_Y)
2 %Function to return an expanded array
3 %This function takes a node and returns the expanded list
4 %of successors,with the calculated fn values.
5 %The criteria being none of the successors are on the CLOSED list.
6 %
7 %Copyright 2009-2010 The MathWorks, Inc.
8
9 %EXPANDED ARRAY FORMAT
10 %-----
11 %|X val |Y val ||h(n) |g(n)|f(n)|
12 %-----
13
14 exp_array=[];
15 exp_count=1;
16 c2=size(CLOSED,1);%Number of elements in CLOSED including the zeros
17 for k= 1:-1:-1
18     for j= 1:-1:-1
19         if (k==j || k==0) %The node itself is not its successor
20             s_x = node_x+k;
21             s_y = node_y+j;
22             if( (s_x > 0 && s_x <=MAX_X) && (s_y > 0 && s_y <=MAX_Y))%node within array bound
23                 flag=1;
24                 for c1=1:c2
25                     if(s_x == CLOSED(c1,1) && s_y == CLOSED(c1,2))
26                         flag=0;
27                     end
28                 end%End of for loop to check if a successor is on closed list.
29                 if (flag == 1)
30                     exp_array(exp_count,1) = s_x;
31                     exp_array(exp_count,2) = s_y;
32                     exp_array(exp_count,3) = distance(xTarget,yTarget,s_x,s_y);%distance between node and goal,hn
33                     exp_array(exp_count,4) = gn+distance(node_x,node_y,s_x,s_y);%cost of travelling to node,g
34                     exp_array(exp_count,5) = exp_array(exp_count,3)+exp_array(exp_count,4);%fn
35                     exp_count=exp_count+1;
36                 end%Populate the exp_array list!!!
37             end% End of node within array bound
38         end%End of if node is not its own successor loop
39     end%End of j for loop
40 end%End of k for loop

```



这张图片对应的随机障碍物地图我用 MATLAB 保存到了 `map_wrong_1.mat` 文件中, 想要复现我的问题, 只需把作业的 `main.m` 中随机生成障碍物地图的那行注释掉, 然后新加一行把 `map` 变量加载进来即可。

```

% Used for Motion Planning for Mobile Robots
% Thanks to HKUST ELEC 5660
close all; clear all; clc;
addpath('A_star')

```

```
% Environment map in 2D space
xStart = 1.0;
yStart = 1.0;
xTarget = 9.0;
yTarget = 9.0;
MAX_X = 10;
MAX_Y = 10;

%如果想要复现我的问题，只需把下面这行注释掉，然后新加一行把map变量加载进来即可
%map = obstacle_map(xStart, yStart, xTarget, yTarget, MAX_X, MAX_Y);
load("C:\Users\zhuoqun.chen\Desktop\2022Spring\深蓝学院-路径规划课程\hw_2\map_wrong_1.mat", "map")

% Waypoint Generator Using the A*
path = A_star_search(map, MAX_X,MAX_Y);

% visualize the 2D grid map
visualize_map(map, path, []);

% save map
% save('Data/map.mat', 'map', 'MAX_X', 'MAX_Y');
```

C++ 版本

刚开始本来是想在 Thinkpad-Win10 中的 WSL2 中运行的，但是配置完后发现居然突然打开不了 RVIZ 了，所以这次只能在 DELL-Ubuntu20.04 中测试了。做第 1 章作业的时候起的是 catkin_ws，这次第 2 章新建一个 catkin_ws_ch2。

在 WSL2 中出现的出现的找不到头文件的问题

首先 catkin_make 编译的时候出现了警告，但是还是编译通过了，不知道对后续有无影响（做完作业后发现并无影响）：

```
[ 59%] Building CXX object grid_path_searcher/ObstacleFiles/demo_node.dir/src/read_only/3PS_utils.cpp.o
[ 57%] Building CXX object grid_path_searcher/ObstacleFiles/demo_node.dir/src/Astar_searcher.cpp.o
[ 64%] Building CXX object grid_path_searcher/ObstacleFiles/demo_node.dir/src/read_only/3PS_searcher.cpp.o
In file included from /opt/ros/noetic/include/ros/ros.h:40,
                 from /home/zqchen/catkin_ws_ch2/src/grid_path_searcher/include/Astar_searcher.h:5,
                 from /home/zqchen/catkin_ws_ch2/src/grid_path_searcher/src/Astar_searcher.cpp:1:
/home/zqchen/catkin_ws_ch2/src/grid_path_searcher/src/Astar_searcher.cpp: In member function 'std::vector<Eigen::Matrix<double, 3, 1>> AstarPathfinder::getVisitedNodes()':
(aka 'long unsigned int') [-Wformat=]
 82 |     ROS_WARN("visited nodes size : %d", visited_nodes.size());
    |     ~~~~~^~~~~~
    |               |
    |               | std::vector<Eigen::Matrix<double, 3, 1>>::size_type (aka long unsigned int)
/home/zqchen/catkin_ws_ch2/src/grid_path_searcher/src/Astar_searcher.cpp:82:37: note: format string is defined here
 82 |     ROS_WARN("visited nodes size : %d", visited_nodes.size());
    |     ~~~~~^~~~~~
    |               |
    |               | int
    |               | int
[ 71%] Linking CXX shared library /home/zqchen/catkin_ws_ch2/devel/lib/librviz_plugins.so
[ 78%] Built target rviz_plugins
[ 85%] Linking CXX executable /home/zqchen/catkin_ws_ch2/devel/lib/grid_path_searcher/random_complex
```

编译完成后发现各个文件的最开始的包含头文件的地方一直有红色波浪线的报错，参考了一些 CSDN 上的文章，发现是可以通过在编译的时候额外添加一个 CMAKE 选项

```
catkin_make -DCMAKE_EXPORT_COMPILE_COMMANDS=ON
```

使得生成一个 `compile_commands.json` 的文件，把这个文件加入到 VS Code 的 C/C++ 扩展插件的配置中就可以不报错了，至于这个是不是一个最优的方案再另说，首先保证解决当下的问题。

DELL-Ubuntu：把新的三个功能包拷贝到新的工作空间

在 DELL-Ubuntu20.04 下：

把三个功能包拷贝到 `/usr/local/catkin_ws_ch2/src/` 路径下（`/home` 目录内存不够了，我以前记录过这个问题），进入到 `src` 目录下 `catkin_init_workspace` 后返回到 `catkin_ws_ch2` 根目录 `catkin_make`，这时会报错，错误和第 1 章的作业一样的，只需要设置 C++ 标准为 14 并且把分布的 `topic` 由 `"/world"` 改为 `"world"` 即可。

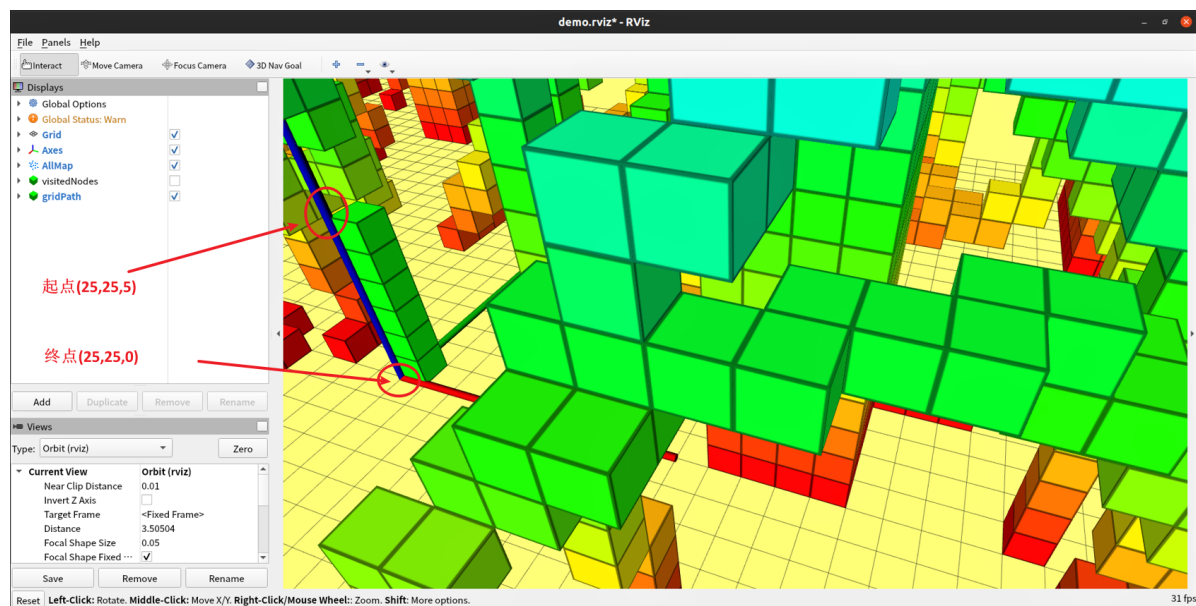
编译成功后先不要急着运行，因为肯定是无法运行的，此时我还没有开始实现 A-star 算法呢。但是此时 `roscore` 后 `RVIZ` 是能正常进去的。

使用上面相同的步骤，不知道为什么在 `Horizon-Thinkpad` 上出现 `RVIZ` 不能正常启动的情况。

最终效果截图演示

2022-03-24周四晚 ROS上的 C++ 版本的代码终于调试成功了！超级鸡冻！

在 `RVIZ` 中点击两次鼠标左键，生成目标节点的位置：



运行 A-star 算法成功的命令行：

```
process[waypoint_generator-3]: started with pid [224424]
process[rviz-4]: started with pid [224430]
[ WARN ] [1648122637.388206657]: 3D Goal Set
[ INFO ] [1648122637.399178437]: [node] receive the planning target
[ INFO ] [1648122637.402522494]: start node:(25, 25, 5)
[ INFO ] [1648122637.402604048]: end node:(25, 25, 0)
[ WARN ] [1648122637.402677312]: The closedSet has been initiated!
[ WARN ] [1648122637.402789650]: Currently the startPtr in the openSet has the smallest f(n)!
[ INFO ] [1648122637.402880446]: WHILE COUNT = 1
[ WARN ] [1648122637.402959878]: The current find_goalIdx_flag = 0
[ WARN ] [1648122637.403053675]: One record has been erased from openSet, now has 0 keys in it.
[ INFO ] [1648122637.403454367]: WHILE COUNT = 2
[ WARN ] [1648122637.403508625]: The current find_goalIdx_flag = 0
[ WARN ] [1648122637.403563239]: One record has been erased from openSet, now has 24 keys in it.
[ INFO ] [1648122637.403839634]: WHILE COUNT = 3
[ WARN ] [1648122637.403910030]: The current find_goalIdx_flag = 0
[ WARN ] [1648122637.403976400]: One record has been erased from openSet, now has 33 keys in it.
[ INFO ] [1648122637.404199307]: WHILE COUNT = 4
[ WARN ] [1648122637.404254918]: The current find_goalIdx_flag = 0
[ WARN ] [1648122637.40432273]: One record has been erased from openSet, now has 41 keys in it.
[ INFO ] [1648122637.404595513]: WHILE COUNT = 5
[ WARN ] [1648122637.404672212]: The current find_goalIdx_flag = 0
[ WARN ] [1648122637.404725659]: One record has been erased from openSet, now has 49 keys in it.
[ INFO ] [1648122637.404945751]: WHILE COUNT = 6
[ WARN ] [1648122637.405019200]: The current find_goalIdx_flag = 1
[ WARN ] [1648122637.405090484]: One record has been erased from openSet, now has 57 keys in it.
[ WARN ] [1648122637.405167519]: [A*]{success} Time in A* is 2.698191 ms, path cost is 0.200000 m
[ WARN ] [1648122637.406664548]: visited_nodes size : 5
```

起点坐标
终点坐标

A*使得只走了5个点就直接规划到终点，使用最原始的迪杰斯特拉算法需要多走300多个点

如果保持 `getHeu()` 函数内部实现为空，即使得 `h` 值一直为 0，那么实现的就是 Dijkstra 算法，没有欧几里得距离对应的 `h` 值作为贪心策略的指引，效率比 A* 要慢很多，经过测试，即使是最简单的点也需要多走几百个点才能找到最优路径。

保存障碍物地图到 .pcd 文件中

为了更好的比较几种基于图搜索的算法的优劣，有必要控制变量进行比较：控制相同的起点并且使用同一个随机生成的地图。

首先打开 RVIZ，对 `rcvPointCloudCallback()` 函数进行一下修改，使得把 ROS 传来的点云 `sensor_msgs::PointCloud2` 先转化为 `pcl` 库的点云格式 `pcl::PointCloud<pcl::PointXYZ>` 后存储到 `/usr/local/catkin_ws_ch2/test_temp_pcd.pcd` 文件中。

注意不要直接写 `test_temp_pcd.pcd`，参考 [Using savePCDFileASCII, PCD file isn't created \(no errors though\) - ROS Answers: Open Source Q&A Forum](#)，如果直接写会在当前目录中找不到这个生成的文件的，因为默认会放在 `./ros` 目录下。之后执行 `catkin_make` 并运行 `roslaunch grid_path_searcher demo.launch`。

记得在 `demo_node.cpp` 的开头额外添加一个头文件（参考：[pcl-ros-tutorial/PCL Reference with ROS.md at master · methylDragon/pcl-ros-tutorial](#)）：

```
#include <pcl/io/pcd_io.h>
```

```
void rcvPointCloudCallback(const sensor_msgs::PointCloud2 & pointcloud_map)
{
    if(!_has_map ) return;

    pcl::PointCloud<pcl::PointXYZ> cloud;
    pcl::PointCloud<pcl::PointXYZ> cloud_vis;
    sensor_msgs::PointCloud2 map_vis;

    pcl::fromROSMsg(pointcloud_map, cloud);

    if( (int)cloud.points.size() == 0 ) return;
```

```

pcl::PointXYZ pt;
for (int idx = 0; idx < (int)cloud.points.size(); idx++)
{
    pt = cloud.points[idx];

    // set obstacles into grid map for path planning
    // 将障碍物信息设置进入栅格化地图，为后续路径规划做准备
    _astar_path_finder->setObs(pt.x, pt.y, pt.z);
    _jps_path_finder->setObs(pt.x, pt.y, pt.z);

    // for visualize only
    // 可视化地图部分
    Vector3d cor_round = _astar_path_finder-
>coordRounding(Vector3d(pt.x, pt.y, pt.z));
    pt.x = cor_round(0);
    pt.y = cor_round(1);
    pt.z = cor_round(2);
    cloud_vis.points.push_back(pt);
}

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

/****
//added by zqchen to save the point_cloud to a .pcd file
//to get a fixed map to compare the performance among different
algorithms.
pcl::io::savePCDFileASCII ("/usr/local/catkin_ws_ch2/test_temp_pcd.pcd",
cloud_vis);
ROS_WARN("Generate .pcd Success!");
****

pcl::toROSMsg(cloud_vis, map_vis);

map_vis.header.frame_id = "world";
_grid_map_vis_pub.publish(map_vis);

_has_map = true;
}

```

这样点云就保存在该文件中了。

之后我们可以一直使用这个地图，只需加载进来即可，还需修改 `rcvPointCloudCallback()` 函数，两段 `****` 注释行中间的是新增的和注释掉的部分：

```

void rcvPointCloudCallback(const sensor_msgs::PointCloud2 & pointcloud_map)
{
    if(_has_map ) return;

```

```

pcl::PointCloud<pcl::PointXYZ> cloud;
pcl::PointCloud<pcl::PointXYZ> cloud_vis;
sensor_msgs::PointCloud2 map_vis;

//把这行注释掉，可以下次不使用新的随机生成的地图，而是使用我之前保存的.pcd文件载入
// pcl::fromROSMsg(pointcloud_map, cloud);

/*****
if (pcl::io::loadPCDFile<pcl::PointXYZ>
("/usr/local/catkin_ws_ch2/test_temp_pcd.pcd", cloud) == -1)
{
    ROS_WARN("Couldn't read file test_temp_pcd.pcd!");
    return;
}
ROS_WARN("Load test_temp_pcd.pcd Success!");//点数应该为63059
*****/

if( (int)cloud.points.size() == 0 ) return;

pcl::PointXYZ pt;
for (int idx = 0; idx < (int)cloud.points.size(); idx++)
{
    pt = cloud.points[idx];

    // set obstacles into grid map for path planning
    // 将障碍物信息设置进入栅格化地图，为后续路径规划做准备
    _astar_path_finder->setObs(pt.x, pt.y, pt.z);
    _jps_path_finder->setObs(pt.x, pt.y, pt.z);

    // for visualize only
    // 可视化地图部分
    Vector3d cor_round = _astar_path_finder-
>coordRounding(Vector3d(pt.x, pt.y, pt.z));
    pt.x = cor_round(0);
    pt.y = cor_round(1);
    pt.z = cor_round(2);
    cloud_vis.points.push_back(pt);
}

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

/*****
//added by zqchen to save the point_cloud to a .pcd file
//to get a fixed map to compare the performance among different
algorithms.
*****/

```

```

// pcl::io::savePCDFileASCII
("/usr/local/catkin_ws_ch2/test_temp_pcd.pcd", cloud_vis);
// ROS_WARN("Generate .pcd Success!");
/*****/

pcl::toROSMsg(cloud_vis, map_vis);

map_vis.header.frame_id = "world";
_grid_map_vis_pub.publish(map_vis);

_has_map = true;
}

```

再次编译运行后可以看到地图还是原来生成的那个地图。之后我们就可以在同一个点云障碍物地图中做测试了。

算法运行效率的比较以及是否加入 Tie Breaker 产生的影响

A* 算法采用 Euclidean 和 Diagonal 的效率对比

在做实验的过程中，对 A* 算法分别采用了两种启发式函数：Euclidean 和 Diagonal。实验结果证明：二者产生的路径都是最优的，但是在三维空间规划时 Diagonal 的效率更高，因为其约束更紧凑。课程中只讲解了二维的 Diagonal，三维的情况类似，参考[python - Calculating 'Diagonal Distance' in 3 dimensions for A* path-finding heuristic - Stack Overflow](#)可以获得三维中 h 的表达式。

实验1: (25, 25, 0) ----> (29, 25, 0)

```

[WARN] [1648212851.597931963]: Load test_temp_pcd.pcd Success!
[WARN] [1648212864.844443533]: 3D Goal Set
[INFO] [1648212864.855135940]: [Node] receive the planning target
[INFO] [1648212864.855252633]: Vector3d_start_pt: (0.000000, 0.000000, 1.000000)
[INFO] [1648212864.855311262]: Vector3d_target_pt: (0.855865, 0.138823, 0.000000)
[INFO] [1648212864.855442384]: start node:(25, 25, 5)
[INFO] [1648212864.855535474]: end node:(29, 25, 0)
[WARN] [1648212864.855639899]: The ClosedSet has been initiated!
[WARN] [1648212864.855756626]: Currently the startPtr in the openSet has the smallest f(n)!
[INFO] [1648212864.855837543]: WHILE COUNT = 1
[WARN] [1648212864.855924239]: The current find_goalIdx_flag = 0
[WARN] [1648212864.856017547]: One record has been erased from openSet, now has 0 keys in it.
[INFO] [1648212864.856068661]: WHILE COUNT = 2
[WARN] [1648212864.856164289]: The current find_goalIdx_flag = 0
[WARN] [1648212864.856292978]: One record has been erased from openSet, now has 25 keys in it.
[INFO] [1648212864.857052751]: WHILE COUNT = 3
[WARN] [1648212864.857117819]: The current find_goalIdx_flag = 0
[WARN] [1648212864.857178918]: One record has been erased from openSet, now has 40 keys in it.
[INFO] [1648212864.857462943]: WHILE COUNT = 4
[WARN] [1648212864.857511687]: The current find_goalIdx_flag = 0
[WARN] [1648212864.857552193]: One record has been erased from openSet, now has 54 keys in it.
[INFO] [1648212864.857799712]: WHILE COUNT = 5
[WARN] [1648212864.857845721]: The current find_goalIdx_flag = 0
[WARN] [1648212864.857885824]: One record has been erased from openSet, now has 68 keys in it.
[INFO] [1648212864.858128769]: WHILE COUNT = 6
[WARN] [1648212864.858185630]: The current find_goalIdx_flag = 1
[WARN] [1648212864.858219132]: One record has been erased from openSet, now has 76 keys in it.
[WARN] [1648212864.858265725]: [A*]{success} Time in A* is 2.873831 ms, path cost is 0.266274 m
[WARN] [1648212864.859568566]: Visited_nodes size : 5

```

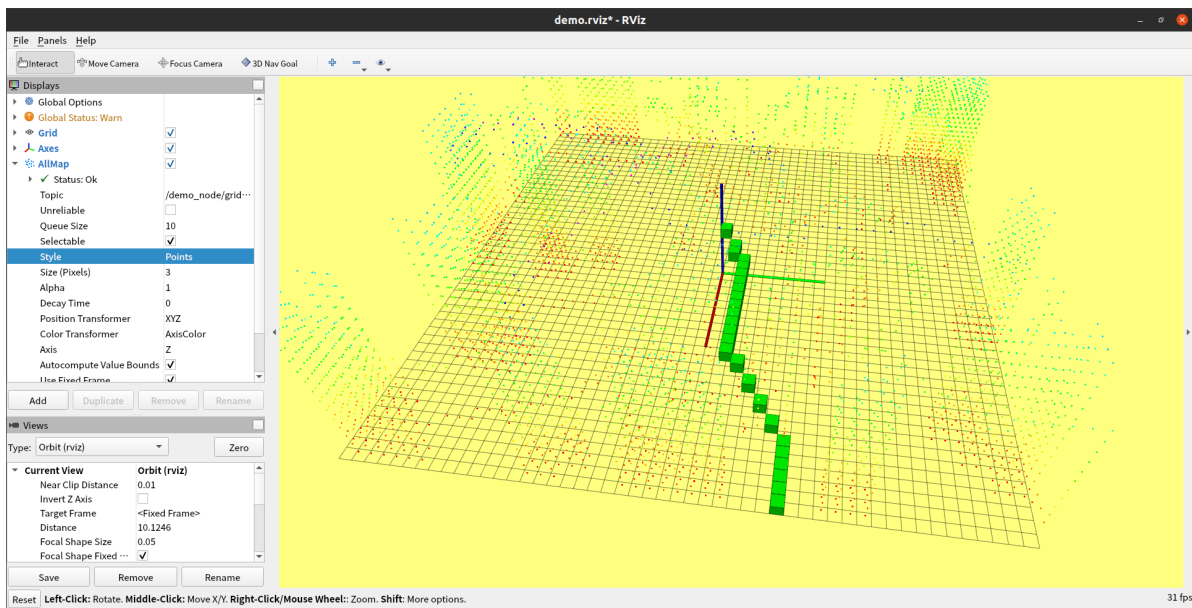


```
roscore http://zqchen-1 x zqchen@zqchen-Inspir x /usr/local/catkin_ws_ch x zqchen@zqchen-Inspir x zqchen@chenzhuoqu x Windows PowerShell x + v - _ □ x

[ INFO ] [1648206294.860338267]: Vector3d _start_pt: (0.000000, 0.000000, 1.000000)
[ INFO ] [1648206294.860392496]: Vector3d target_pt: (0.866474, 0.116800, 0.000000)
[ INFO ] [1648206294.860513466]: start node:(25, 25, 5)
[ INFO ] [1648206294.860569104]: end node:(29, 25, 0)
[ WARN ] [1648206294.860620262]: The ClosedSet has been initiated!
[ WARN ] [1648206294.860683813]: Currently the startPtr in the openSet has the smallest f(n)!
[ INFO ] [1648206294.860736967]: WHILE COUNT = 1
[ WARN ] [1648206294.860783273]: The current find_goalIdx_flag = 0
[ WARN ] [1648206294.860837447]: One record has been erased from openSet, now has 0 keys in it.
[ INFO ] [1648206294.861162486]: WHILE COUNT = 2
[ WARN ] [1648206294.861218671]: The current find_goalIdx_flag = 0
[ WARN ] [1648206294.861261028]: One record has been erased from openSet, now has 25 keys in it.
[ INFO ] [1648206294.861547894]: WHILE COUNT = 3
[ WARN ] [1648206294.861607945]: The current find_goalIdx_flag = 0
[ WARN ] [1648206294.861660362]: One record has been erased from openSet, now has 40 keys in it.
[ INFO ] [1648206294.861898845]: WHILE COUNT = 4
[ WARN ] [1648206294.861952510]: The current find_goalIdx_flag = 0
[ WARN ] [1648206294.861995509]: One record has been erased from openSet, now has 54 keys in it.
[ INFO ] [1648206294.862232923]: WHILE COUNT = 5
[ WARN ] [1648206294.862287275]: The current find_goalIdx_flag = 0
[ WARN ] [1648206294.862332067]: One record has been erased from openSet, now has 68 keys in it.
[ INFO ] [1648206294.862494975]: WHILE COUNT = 6
[ WARN ] [1648206294.862551283]: The current find_goalIdx_flag = 0
[ WARN ] [1648206294.862595219]: One record has been erased from openSet, now has 70 keys in it.
[ INFO ] [1648206294.862801494]: WHILE COUNT = 7
[ WARN ] [1648206294.862873809]: The current find_goalIdx_flag = 1
[ WARN ] [1648206294.862949099]: One record has been erased from openSet, now has 78 keys in it.
[ WARN ] [1648206294.863028243]: [A*]{success} Time in A* is 2.559770 ms, path cost is 0.266274 m
[ WARN ] [1648206294.864592185]: visited_nodes size : 6 需要走6个点
```

A*算法使用Euclidean Heuristic

实验2: (25, 25, 0) ----> (49, 32, 0)



```
roscore http://zqchen-1 x zqchen@zqchen-Inspir x /usr/local/catkin_ws_ch x zqchen@zqchen-Inspir x zqchen@chenzhuoqu x Windows PowerShell x + v - _ □ x

[ INFO ] [1648213006.807683394]: WHILE COUNT = 42
[ WARN ] [1648213006.807762316]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.807805492]: One record has been erased from openSet, now has 270 keys in it.
[ INFO ] [1648213006.807984034]: WHILE COUNT = 43
[ WARN ] [1648213006.808065664]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.808128313]: One record has been erased from openSet, now has 269 keys in it.
[ INFO ] [1648213006.808363824]: WHILE COUNT = 44
[ WARN ] [1648213006.808454747]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.808544849]: One record has been erased from openSet, now has 272 keys in it.
[ INFO ] [1648213006.808877967]: WHILE COUNT = 45
[ WARN ] [1648213006.808963348]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.809026289]: One record has been erased from openSet, now has 283 keys in it.
[ INFO ] [1648213006.809287228]: WHILE COUNT = 46
[ WARN ] [1648213006.809394348]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.809462481]: One record has been erased from openSet, now has 292 keys in it.
[ INFO ] [1648213006.809688385]: WHILE COUNT = 47
[ WARN ] [1648213006.809769843]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.809834858]: One record has been erased from openSet, now has 297 keys in it.
[ INFO ] [1648213006.810056282]: WHILE COUNT = 48
[ WARN ] [1648213006.810131824]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.810190886]: One record has been erased from openSet, now has 302 keys in it.
[ INFO ] [1648213006.810414142]: WHILE COUNT = 49
[ WARN ] [1648213006.810487501]: The current find_goalIdx_flag = 0
[ WARN ] [1648213006.810560737]: One record has been erased from openSet, now has 307 keys in it.
[ INFO ] [1648213006.810738627]: WHILE COUNT = 50
[ WARN ] [1648213006.810801281]: The current find_goalIdx_flag = 1
[ WARN ] [1648213006.810859103]: One record has been erased from openSet, now has 312 keys in it.
[ WARN ] [1648213006.810899008]: [A*]{success} Time in A* is 15.551463 ms, path cost is 1.139547 m
[ WARN ] [1648213006.812151379]: visited_nodes size : 49 A* - Diagonal-Heuristic 到(49, 32, 0)需要visit 49个点
```

```
roscore http://zqchen-1 x zqchen@zqchen-inspir x /usr/local/catkin_ws_ch x zqchen@zqchen-inspir x zqchen@chenzhuoqun x Windows PowerShell x
[ INFO] [1648213523.981826248]: WHILE COUNT = 921
[ WARN] [1648213523.981842425]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.981861245]: One record has been erased from openSet, now has 888 keys in it.
[ INFO] [1648213523.981916126]: WHILE COUNT = 922
[ WARN] [1648213523.981926064]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.981945106]: One record has been erased from openSet, now has 888 keys in it.
[ INFO] [1648213523.981990547]: WHILE COUNT = 923
[ WARN] [1648213523.982006873]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.982025311]: One record has been erased from openSet, now has 888 keys in it.
[ INFO] [1648213523.982069944]: WHILE COUNT = 924
[ WARN] [1648213523.982086089]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.982105188]: One record has been erased from openSet, now has 887 keys in it.
[ INFO] [1648213523.982146296]: WHILE COUNT = 925
[ WARN] [1648213523.982163317]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.982181836]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648213523.982230997]: WHILE COUNT = 926
[ WARN] [1648213523.982248029]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.982266927]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648213523.982314600]: WHILE COUNT = 927
[ WARN] [1648213523.982331666]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.982350177]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648213523.982389908]: WHILE COUNT = 928
[ WARN] [1648213523.982406041]: The current find_goalIdx_flag = 0
[ WARN] [1648213523.982424498]: One record has been erased from openSet, now has 885 keys in it.
[ INFO] [1648213523.982474518]: WHILE COUNT = 929
[ WARN] [1648213523.982491335]: The current find_goalIdx_flag = 1
[ WARN] [1648213523.982509734]: One record has been erased from openSet, now has 890 keys in it.
[ WARN] [1648213523.982536212]: [A*]{success} Time in A* is 110.721355 ms, path cost is 1.139547 m
[ WARN] [1648213523.983417093]: visited_nodes size : 928
```

最短路径长度和Diagonal Heuristic是一样的，都是最优的，路径长度都相同，只是有可能中途经过的路径不同。

A* - Euclidean Heuristic 到(49, 32, 0)需要visit 928 个点

加入 Tie Breaker 对效率的影响

实验3: Euclidean + $p=0.001$

```
roscore http://zqchen-1 x zqchen@zqchen-inspir x /usr/local/catkin_ws_ch x zqchen@zqchen-inspir x zqchen@chenzhuoqun x Windows PowerShell x
[ INFO] [1648214513.750917107]: WHILE COUNT = 917
[ WARN] [1648214513.750927680]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.750937455]: One record has been erased from openSet, now has 887 keys in it.
[ INFO] [1648214513.750973228]: WHILE COUNT = 918
[ WARN] [1648214513.750983673]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.750993362]: One record has been erased from openSet, now has 887 keys in it.
[ INFO] [1648214513.751027350]: WHILE COUNT = 919
[ WARN] [1648214513.751037496]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.751047231]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648214513.751083933]: WHILE COUNT = 920
[ WARN] [1648214513.751094309]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.751104224]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648214513.751141749]: WHILE COUNT = 921
[ WARN] [1648214513.751152205]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.751161705]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648214513.751194248]: WHILE COUNT = 922
[ WARN] [1648214513.751205266]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.751215302]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648214513.751249235]: WHILE COUNT = 923
[ WARN] [1648214513.751259824]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.751269252]: One record has been erased from openSet, now has 886 keys in it.
[ INFO] [1648214513.751298824]: WHILE COUNT = 924
[ WARN] [1648214513.751309158]: The current find_goalIdx_flag = 0
[ WARN] [1648214513.751317287]: One record has been erased from openSet, now has 885 keys in it.
[ INFO] [1648214513.751354581]: WHILE COUNT = 925
[ WARN] [1648214513.751365311]: The current find_goalIdx_flag = 1
[ WARN] [1648214513.751376538]: One record has been erased from openSet, now has 890 keys in it.
[ WARN] [1648214513.751393538]: [A*]{success} Time in A* is 66.324517 ms, path cost is 1.139547 m
[ WARN] [1648214513.752290538]: visited_nodes size : 924
```

A* Euclidean加入 $p=0.001$ 的Tie Breaker时间从100多ms降低到60多ms，经过的节点少了4个

实验4: Diagonal + $p=0.001$

```
[ WARN] [1648214887.640174698]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.640228675]: One record has been erased from openSet, now has 277 keys in it.
[ INFO] [1648214887.640405730]: WHILE COUNT = 76
[ WARN] [1648214887.640478637]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.640532391]: One record has been erased from openSet, now has 277 keys in it.
[ INFO] [1648214887.640690877]: WHILE COUNT = 77
[ WARN] [1648214887.640759232]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.640823994]: One record has been erased from openSet, now has 279 keys in it.
[ INFO] [1648214887.641060742]: WHILE COUNT = 78
[ WARN] [1648214887.641116096]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.641157061]: One record has been erased from openSet, now has 287 keys in it.
[ INFO] [1648214887.641366138]: WHILE COUNT = 79
[ WARN] [1648214887.641454756]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.641502431]: One record has been erased from openSet, now has 294 keys in it.
[ INFO] [1648214887.641669276]: WHILE COUNT = 80
[ WARN] [1648214887.641722660]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.641770520]: One record has been erased from openSet, now has 299 keys in it.
[ INFO] [1648214887.641924110]: WHILE COUNT = 81
[ WARN] [1648214887.641975749]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.642021130]: One record has been erased from openSet, now has 304 keys in it.
[ INFO] [1648214887.642177828]: WHILE COUNT = 82
[ WARN] [1648214887.642227264]: The current find_goalIdx_flag = 0
[ WARN] [1648214887.642266300]: One record has been erased from openSet, now has 309 keys in it.
[ INFO] [1648214887.642426390]: WHILE COUNT = 83
[ WARN] [1648214887.642477818]: The current find_goalIdx_flag = 1
[ WARN] [1648214887.642521791]: One record has been erased from openSet, now has 314 keys in it
[ WARN] [1648214887.642587549]: [A*]{sucess} Time in A* is 26.222877 ms, path cost is 1.172684 m
[ WARN] [1648214887.644183499]: Visited_nodes size : 82
```

A* Diagonal加了p=0.001 性能反而变差了

实验5: $\text{Diagonal} + h \cdot (1 + p=0.001/0.005)$ 或 $\text{Diagonal} + h + 0.001 \cdot \text{dot_product}$

下表通过固定地图，详细分析了从起点 (25, 25, 0) ----> 终点 (49, 7, 0) 使用基于 A*-Diagonal Heuristic 的算法是否加入 Tie Breaker 以及加的 Tie Breaker 种类对算法效率的影响。

是否加 Tie Breaker 以及加的 Tie Breaker 种类	visited nodes	path cost	time	备注
h	157	1.347228m	37ms	无
$h - 0.001 \cdot \text{dot_product}$	172	1.345233m	43ms	按理来说这个应该是更快的呀
$h + 0.001 \cdot \text{dot_product}$	118	1.351083m	13ms	不知为何符号反过来更优(?)
$h \cdot (1 + p=0.001)$	124	1.354938m	13ms	无
$h \cdot (1 + p=0.005)$	113	1.354938m	12.99ms	无

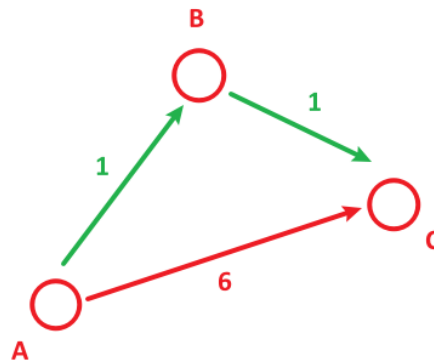
对算法的理解和编程实现上的心得体会



Dijkstra's Algorithm

- Maintain a **priority queue** to store all the nodes to be expanded
- The priority queue is initialized with the start state X_s
- Assign $g(X_s)=0$, and $g(n)=\text{infinite}$ for all other nodes in the graph
- Loop
 - If the queue is empty, return FALSE; break;
 - Remove the node "n" with the lowest $g(n)$ from the priority queue
 - Mark node "n" as expanded
 - If the node "n" is the goal state, return TRUE; break;
 - For all unexpanded neighbors "m" of node "n"
 - If $g(m) = \text{infinite}$
 - $g(m) = g(n) + C_{nm}$
 - Push node "m" into the queue
 - If $g(m) > g(n) + C_{nm}$
 - $g(m) = g(n) + C_{nm}$
- end
- End Loop

我觉得，在这个地方在把 **expanded nodes** 压入 **priority queue** 的时候，不光要更新这个数值本身，也需要更新更小的 $g(m)$ 的 **parent** 变成了谁（即当前更小的是经过了哪个新节点才到这个节点的），这样才可以在最终找到目标终点后往前回溯时找到正确的父节点，不然路径就是错乱的。



举个例子，比如说从 **A** 节点出发，第一次会把 **B** 和 **C** 都压入到优先级队列中，由于 **B** 和 **C** 都是第一次被探索到，所以 g 被更新为 1 和 6，然后下一次 **B** 节点被弹出，然后发现了在 **open_list** 中的 **C**，这时 $1+1$ 显然比 6 要更优，那么不光要更新 **C** 节点的 g 值，还要记录下来：**C** 现在不应该 **cameFrom** 节点 **A**，而是更新为 **cameFrom** 节点 **B**，这样才是更优的。这样做，才能使得最终路径回溯的时候找到最优路径。

后来，我发现有个结论，参考[Heuristics](#)这篇文章：**GridMap** 上根据移动方向局限程度的不同，使用何种 **Heuristic** 函数是最优的，有一些通用的标准结论的。在我们的实验中所有方向都是 8 自由度的，因此不难理解为何 **Diagonal** 会比 **Euclidean** 收敛速度更快。

On a grid, there are well-known heuristic functions to use.

Use the distance heuristic that matches the allowed movement:

- On a square grid that allows **4 directions** of movement, use **Manhattan distance** (L_1).
→
- On a square grid that allows **8 directions** of movement, use **Diagonal distance** (L_∞).
→
- On a square grid that allows **any direction** of movement, you might or might not want **Euclidean distance** (L_2). If A* is finding paths on the grid but you are allowing movement not on the grid, you may want to consider **other representations of the map**.
→
- On a hexagon grid that allows **6 directions** of movement, use Manhattan distance **adapted to hexagonal grids**.