# 概述

路径规划是根据给定的环境模型，在一定的约束条件下，规划出一条从车辆当前位置到目标位置的最优路径，最优常常指路径长度最短或者时间最短。常用的路径规划算法主要分为两类：第一类是基于采样的路径规划算法，如概率图算法（PRM）,快速随机扩展树算法（RRT）；另一类则是基于地图的路径搜索算法，如Dijkstra算法， A\*算法等。本项目主要采用基于地图搜索的A\*算法，它以一种启发性的思想去沿着目标的方向进行搜索，避免了盲目、费时的无用搜索，有效的提高了算法的实时执行效率，这种启发式搜索算法现如今已广泛应用于路径规划中。

# 方案设计

路径规划主要包含两个步骤：一为建立包含障碍区域与自由区域的环境地图；而则是给定起点与终点的条件下，在环境地图中选择合适的路径搜索算法，快速实时地搜索可行路径。如图所示，曲线表示从车辆起始位置到目标位置的一条有效路径。



图2-1 路径规划图

## 环境地图

环境地图是真实环境中物体空间、位置等信息抽象在一张地图模型上，常规的表示方法有度量地图表示法和拓扑地图表示法。度量地图表示法采用坐标系中栅格是否哦被障碍物所占据的方式来描述环境特征；而拓扑地图是选用节点来表示道路上的特定位置，用节点与节点之间的关系来表示道路的联系。因后者包需要路网数据实现比较复杂，且对特定环境的描述能力不强，因此本项目采用基于度量地图的均匀空间分解法来对真实环境进行地图建模。

**均匀空间分解法**

该方法把环境分解为一系列离散的栅格节点，所有栅格节点大小统一、均匀分布，根据它们是否被障碍物所占据来进行状态描述。如果栅格单元被障碍物所占据，则为障碍栅格，反之则为自由栅格。实际中可用二值法进行表示，1表示障碍栅格，禁止通行；0表示自由栅格，可通行。如图2-2为采用上述方法得到的栅格地图，黑色区域可不同行，白色区域可通行，由图中起始栅格与目标栅格都是自由栅格，每个栅格都对应相应的坐标值，坐标值就表示小车在栅格地图中的当前位置。

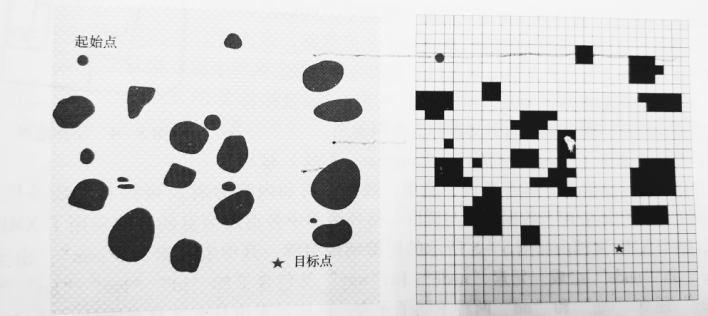


图2-2 栅格障碍地图

环境信息用均匀栅格表示后，栅格节点之间需要建立一定的连接关系才能保证能从起始点搜索到目标点的有效路径。如图2-3，为栅格节点之间的典型连接关系，图（a）表示的是八连接，它表示从当前栅格可以到达与之相邻的八个栅格节点；图（b）表示的是十六连接，它表明可以从当前栅格到达与之相近的十六个栅格。

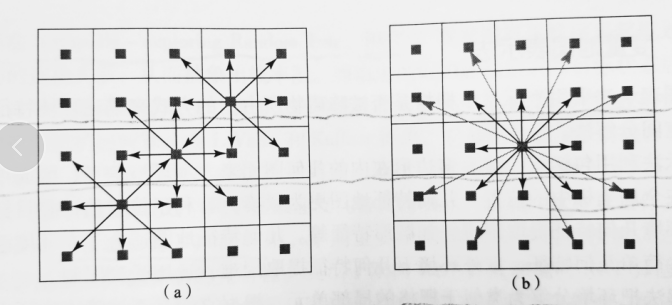


图2-3 八连接和十六连接

**高尔夫球场环境地图构建**

高尔夫球场的环境模型构造思路主要如下，首先在球场的四个角上标识锚点，利用RTK设备采集它们的真实位置信息，然后利用航拍技术来获取一张具有高分辨率的环境图片；通过锚点利用插值算法可以获得环境中每一个点的真实位置信息，然后与高分辨率的环境图片一一匹配，最后建立二维栅格地图，该地图不仅具有环境的特定信息，且每一个栅格都有真实的位置信息与之匹配，极大地有利于路径规划算法以及收球车的自动控制。如图1所示为75x90平方米的高尔夫球场航拍图，图2为间隔为1m抽象出来的二维栅格地图，白色为可行区域，红色为障碍物。

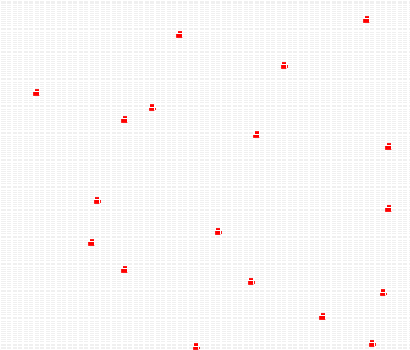
 

图1 高尔夫球场航拍图 图2 二维栅格地图

## A\*算法

A\*算法是一种启发式的搜索算法，和广度优先搜索（BFS）、深度优先搜索（DFS）类似，都是按照一定的原则确定如何展开需要搜索的节点树状结构，A\*可以认为是一种基于“优点”的搜索算法，其核心部分是对每一个道路节点均设计了一个估计函数，如式（2-1）所示：

 （2-1）

式中，表示从起始节点经过节点s到目标节点的估计长度；表示从起始节点到当前节点的路径长度，它是已知的，大小如式（2-2）所示：

 （2-2）

式（2-1）中为启发函数，是当前节点到目标节点的估计值，A\*算法一定能搜索到最优路径的前提条件：

 （2-3）

为当前节点到目标节点的最优距离，满足式（2-3）的值越大，则扩展节点越少。为了保证搜索路径的最优性，通常选择启发函数为曼哈顿距离、对角线距离或者欧几里得距离。

**A\*算法过程如下：**

**第一步：** 给定源点src和目标节点dst，并建立一个需要搜索展开的节点集合（openSet）和已经完成搜索的节点集合closedSet；

**第二步：**从openSet中选取cost最小的节点cur，即，进行8连接来扩展搜索，若扩展的节点u不在openSet集合且也不在closedSet集合中，将其加入openSet集合；

**第三步：**利用当前节点cur进行代价更新，更新准则为当满足源点src到扩展节点u的代价大于源点src到当前节点cur的代价与当前节点cur到扩展节点u的代价之和；

**第四步：**当前节点cur扩展完毕后，将其放入closedSet集合中，表明该节点已经找到最短距离，不需要再加入openSet集合进行搜索更新；

**第五步：**循环上述第二、三、四步操作，直到找到目标节点或者openSet集合为空为止。

其伪代码如下：

Table 1. A\* algorithm process.

**function:** best\_path = AStar\_Routing(MAP, src, dst)

**Input**:  and 

**Output**: Best path from src to dst

1. create vertex set closedSet // set of already visited nodes
2. create vertex set openSet // set of nodes to be expanded
3. create gScore, fScore with default value inf
4. insert src into openSet
5. gScore[src] = 0, fScore[src] = h(src, dst)
6. create pre\_path with default value **nullptr**
7. **while** openSet **is not** empty:
8. current = the node v in openSet s.t. fScore[v] is minimum in openSet
9. **if** current == dst
10. return reconstruction\_best\_path(pre\_path, current)
11. remove current from openSet
12. insert current into closedSet
13. **for each** neighbor u of current:
14. **if** u is in closeSet:
15. continue; // ignore the neighbor who has already been evaluated
16. candidate\_score = gScore[current] + h(current, u)
17. **if u not in** openSet: // discovered a new node
18. insert u into openSet
19. **else if** candidate\_score >= gScore[u]:
20. **continue;**  // this is not a better path
21. pre\_path[u] = current
22. gScore[u] = candidate\_score
23. fScore[u] = gScore[u] + h(u, dst)

# 仿真测试

## 最短路径

采集逸夫楼天台环境，构建地图如图3-1所示，障碍区域用红色正方形表示，可行区域用蓝色点表示。选择不同的起点和终点，利用A\*算法得到最优路径。图3-2、3-3、3-4分别为不同的终点得到的最优路径。

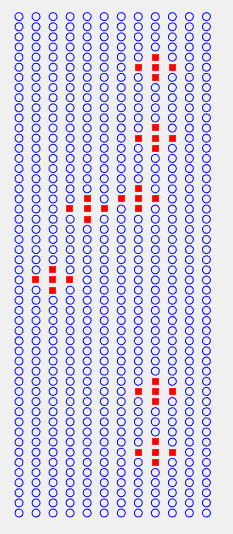
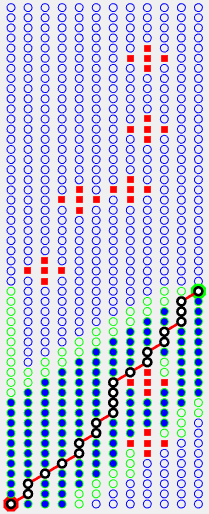
 

图3-1 逸夫楼栅格地图 图3-2 起点（1，1）和终点（12，22）

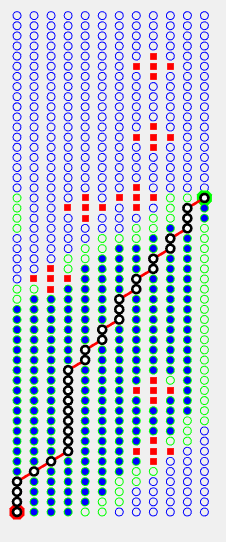
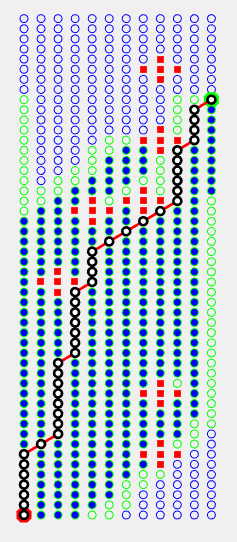
 

图3-3 起点（1，1）和终点（12，32） 图3-4 起点（1，1）和终点（12，42）

## 区域规划

给定区域的左下角和右上角坐标，对其进行区域规划。思路是以指定间隔将该区域划分为若干条直线，对每条直线进行路径规划，最后将所有直线的最短路径结果汇总，形成一条区域最优路径规划。假如给点左下角坐标（1， 1）和右上角坐标（12， 50），仿真结果如下：

图3-5 直线间隔1m 图3-6 直线间隔2m

图3-7 直线间隔3m 图3-8 直线间隔4m

# 结论

从上述仿真结果可以看出，利用A\*算法可以很好地获取任意两点之间的最短路径以及某个区域的最优路径，下一步工作可以对真实的高尔夫球场环境进行建模，获取收球车的区域最优路径规划方案。其次，在地图建模、节点数目选择、转弯半径控制方面需要进一步研究，完善路径规划方案。