1 Matlab 部分

运行截图：

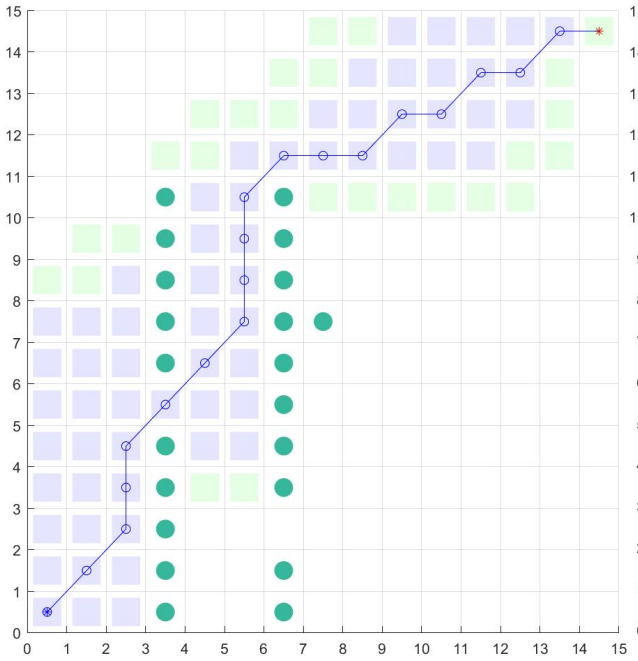

图 1.1 手动添加障碍

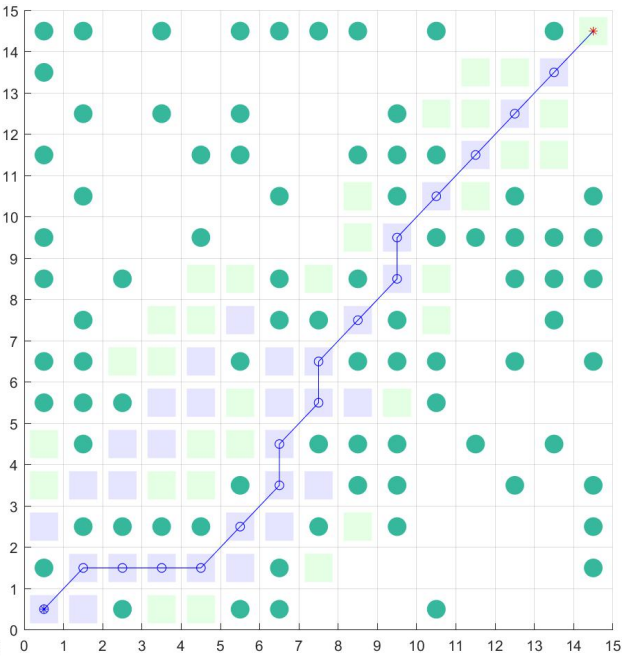

图 1.2 15x15 地图

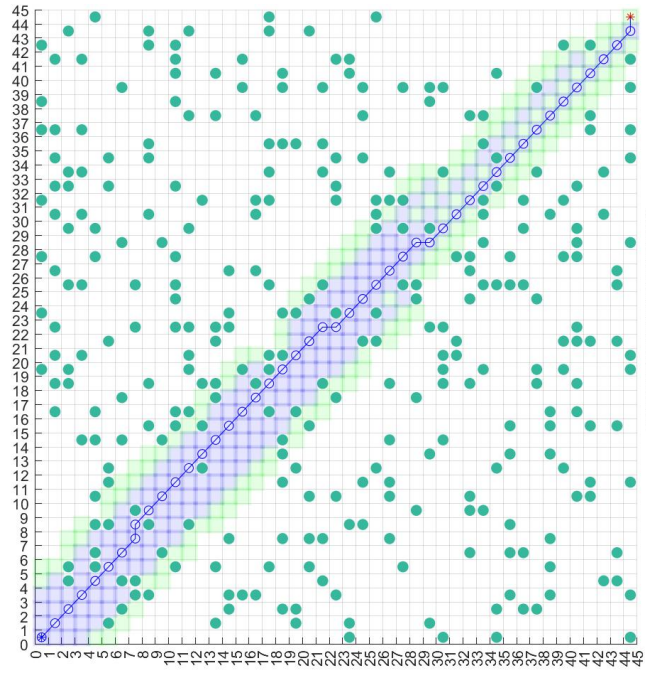

图 1.3 45x45 地图

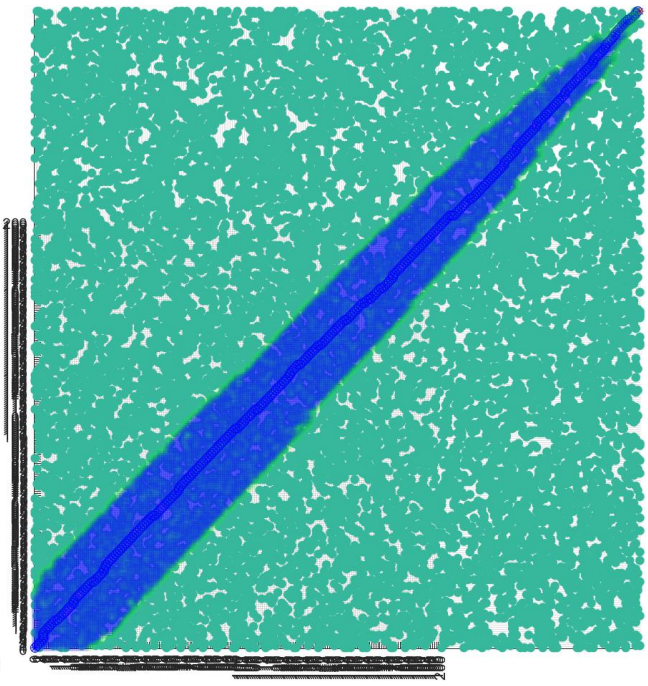

图 1.4 300×300 地图

结果分析见 ROS 部分

2、ROS 部分

2.1 运行结果

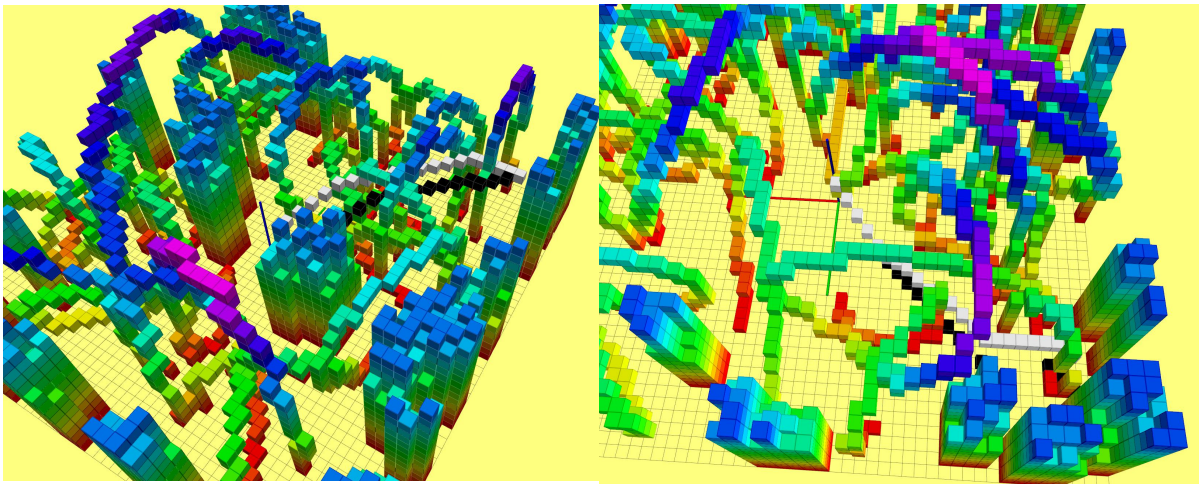

图 2.1 运行截图，其中白色轨迹为 A*，黑色轨迹为 JPS

	Different Algorithm						
	0 (Dijkstra)	Euclidean	Manhattan	Diagonal	Diagonal with Tie Breaker	JPS	
Time (ms)	33.834	0.459	0.07	0.392	0.224	0.113	
Path cost (m)	6.293	6.293	6.429	6.293	6.293	6.293	test1
Visited nodes	48882	220	22	94	21	18	
Time (ms)	28.158	1.174	0.102	0.299	0.141	0.081	
Path cost (m)	6.429	6.53	6.429	6.429	6.429	6.429	test2
Visited nodes	51570	566	22	127	31	19	
Time (ms)	23.87	1.309	0.077	0.421	0.091	0.082	
Path cost (m)	5.561	5.561	5.561	5.561	5.561	5.561	test3
Visited nodes	39184	484	21	141	21	17	

表 2.1 三次测试结果数据

每次测试中的地图、起点、终点为同一个，仅改变启发式函数和是否使用 Tie Breaker。

2.2 不同启发式函数对 A*搜索效率的对比

（1）搜索速度:　Manhattan > Diagonal > Euclidean > Dijkstra
（2）遍历的节点数:　Dijkstra > Euclidean > Diagonal > Manhattan
（3）路径长度:　Euclidean > Manhattan > Diagonal > Dijkstra

2.3 加入 Tie Breaker 的影响

使用 Tie Breaker 明显减少了搜索过程中遍历的节点数目，同时降低了运行的时间

2.4 A*与 JPS 对比

从表 2.1 中可以看出，JPS 在实验中的地图中，障碍较多，此时无论是运行时间还是遍历节点数量都是优于 A*的，但是在开阔的环境中，

如图 2.2，将起点设置到较高的空旷区域，通过对比表 2.2 中数据可以看出，即便遍历的节点少于 A*，但是，由于周围的环境开阔，JPS 跳跃的方式会产生大量的无效搜索而浪费资源，这个时候，JPS 的效率并不比 A*要好。

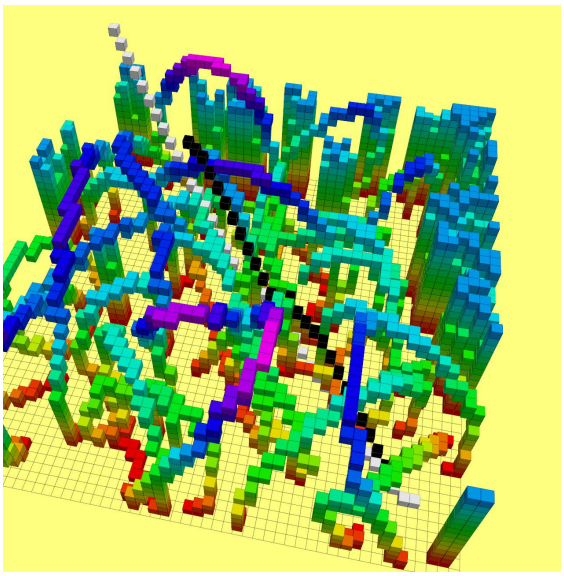

图 2.2A*与 JPS 对比 其中白色轨迹为 A*，黑色轨迹为 JPS

	Comparisons	
	A*	JPS
Time(ms)	0.312	1.119
Path cost(m)	12.907	12.673
Visited nodes	71	52

表 2.2　A*与 JPS 对比