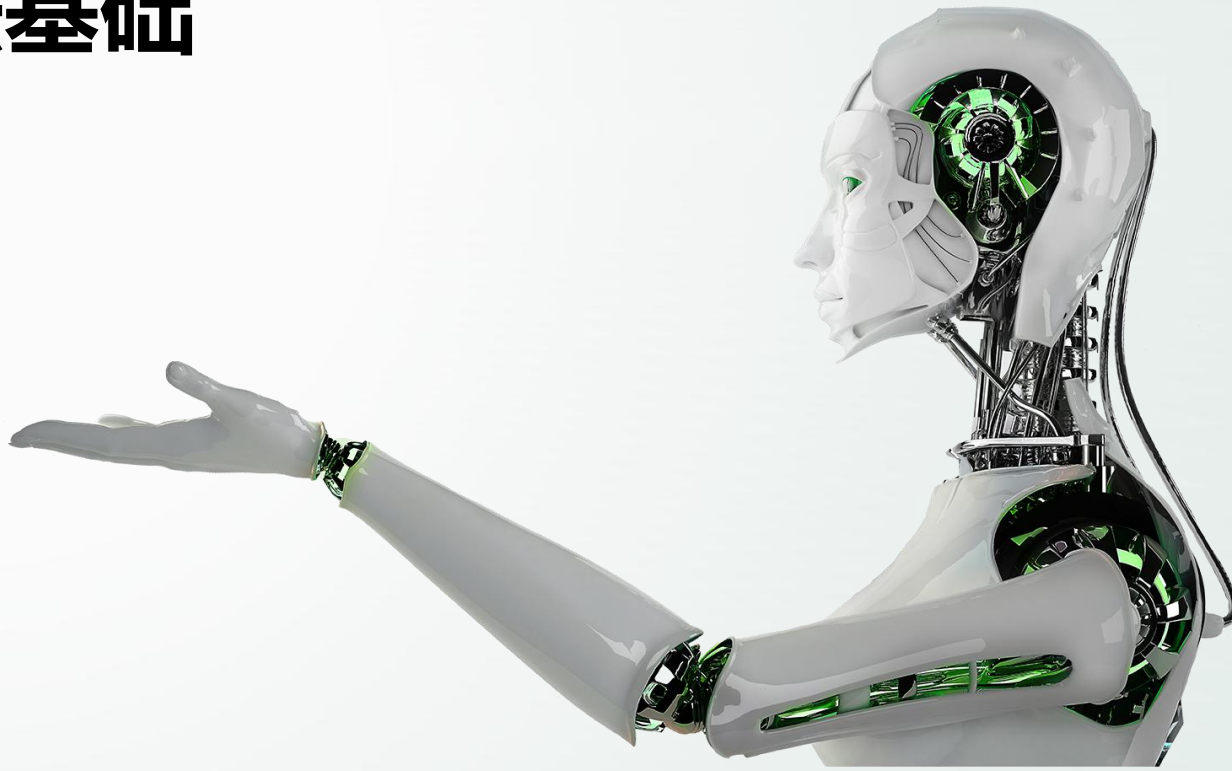
睿慕课 AIIMOOC.COM

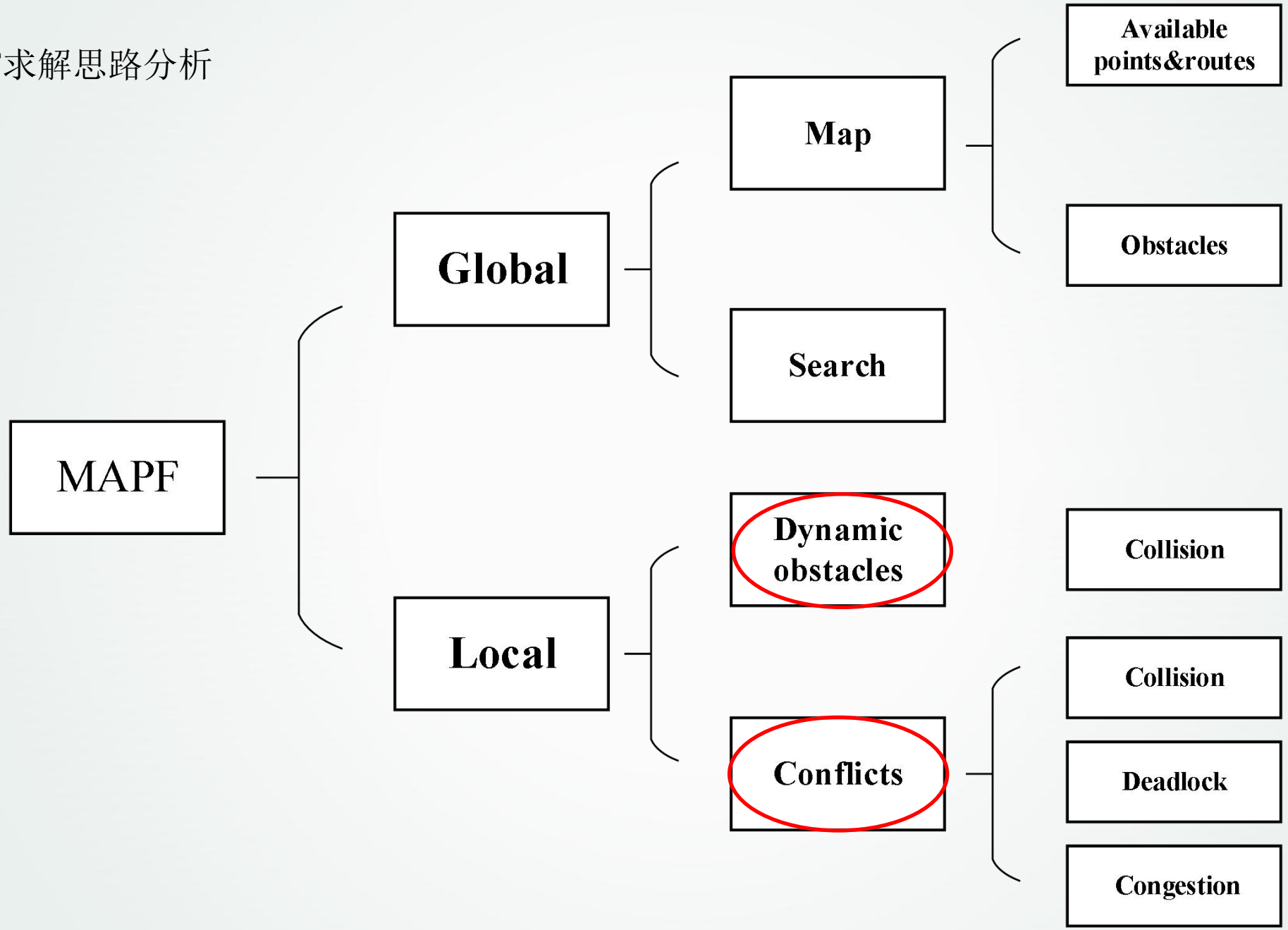
第三章

MAPF全局规划算法基础

1. MAPF求解思路分析
2. 基于采样的全局算法-RRT
3. 基于图论的全局算法-BFS
4. 基于强化学习的全局算法-QL



1. MAPF求解思路分析



睿慕课
AIIMOOC.COM

1. MAPF求解思路分析

全局路径规划算法的类型

基于图论的全局路径规划(深度/广度优先搜索、贝尔曼福德算法、迪杰斯特拉算法、A-star)

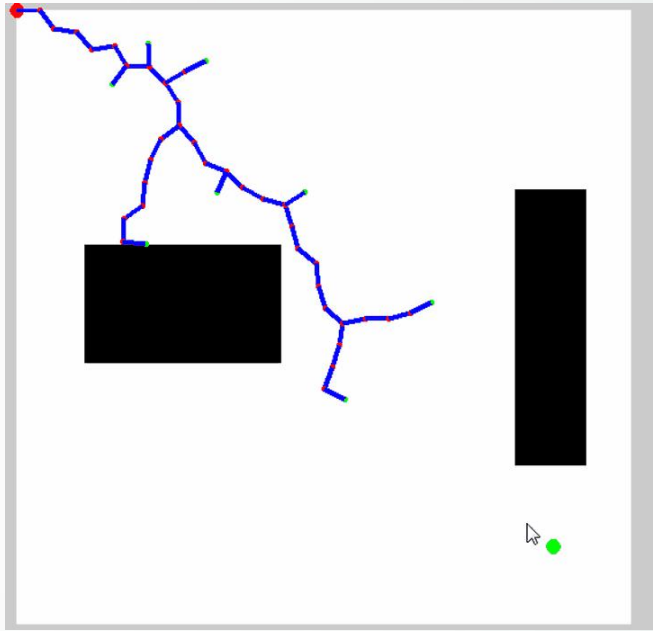
基于采样的全局路径规划（RRT算法）

基于机器学习的全局路径规划（QL）

睿慕课
AIIMOOC.COM

2. 基于采样的全局算法-RRT

快速搜索随机树（**RRT**）算法从起始点开始，在地图上进行随机采样（如**Monte Carlo Sample**）,或者说随机撒点，进而，根据采样点信息，结合障碍物检测等约束条件，构建一颗**搜索树**，直到树的枝叶延伸至目标点或达到预设的采样次数为止。

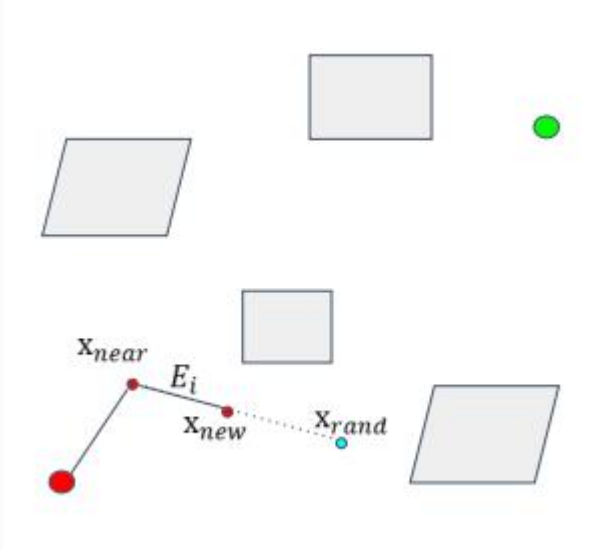


睿慕课
AIIMOOC.COM

2. 基于采样的全局算法-RRT

搜索机制

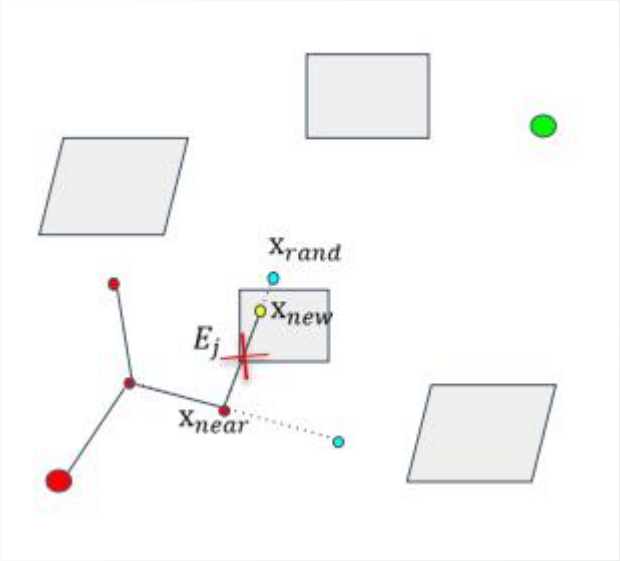
1. 随机采样点 xrand
2. 与xrand距离最近的点 xnear
3. 在xnear和xrand的连线上向前延伸一个stepsize
4. 得到xnew



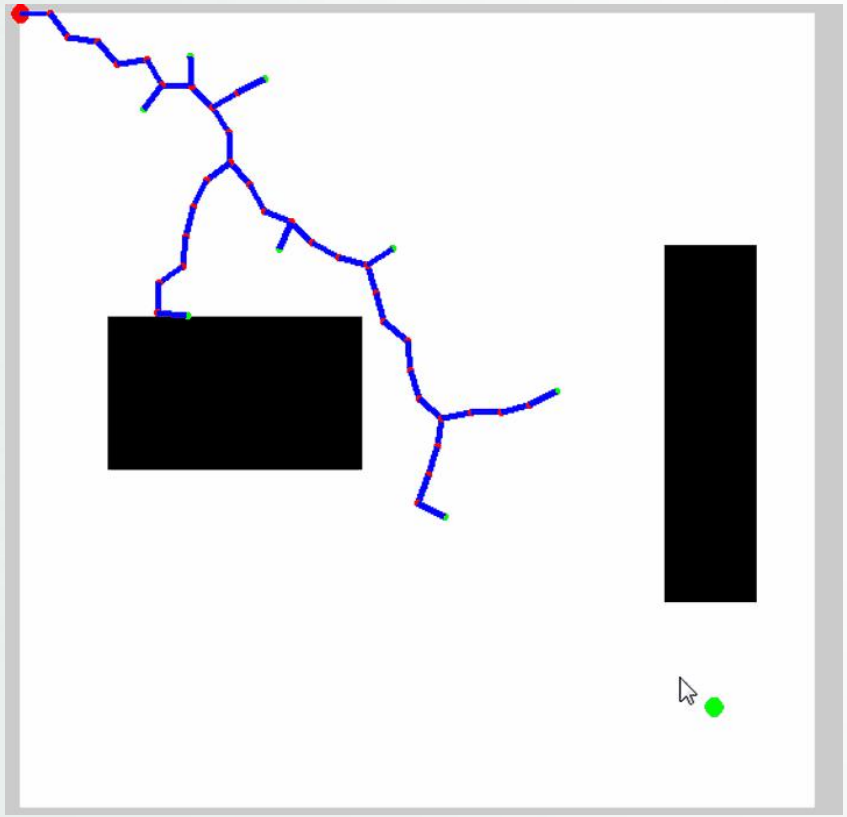
睿慕课
AIIMOOC.COM

碰撞检测

1. 检查xnear和xnew之间的路径Ei
2. 是否是collision free的？
3. 若是，xnew和Ei加入到搜索树中
4. 若否，放弃xnew和xrand，以及路径Ei



迭代，找到目标点

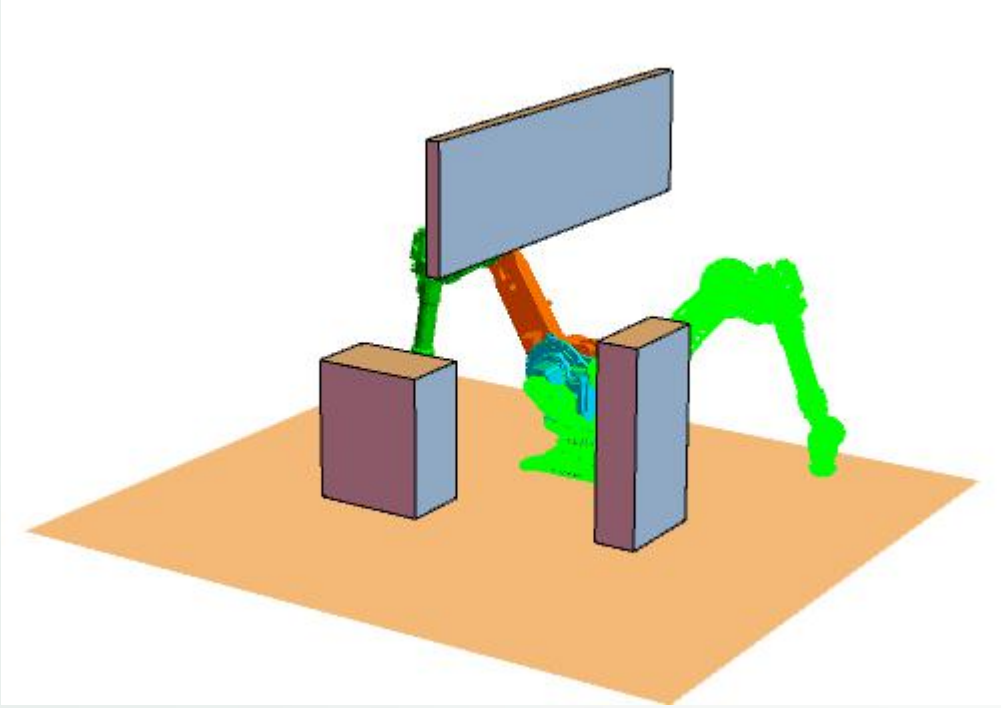


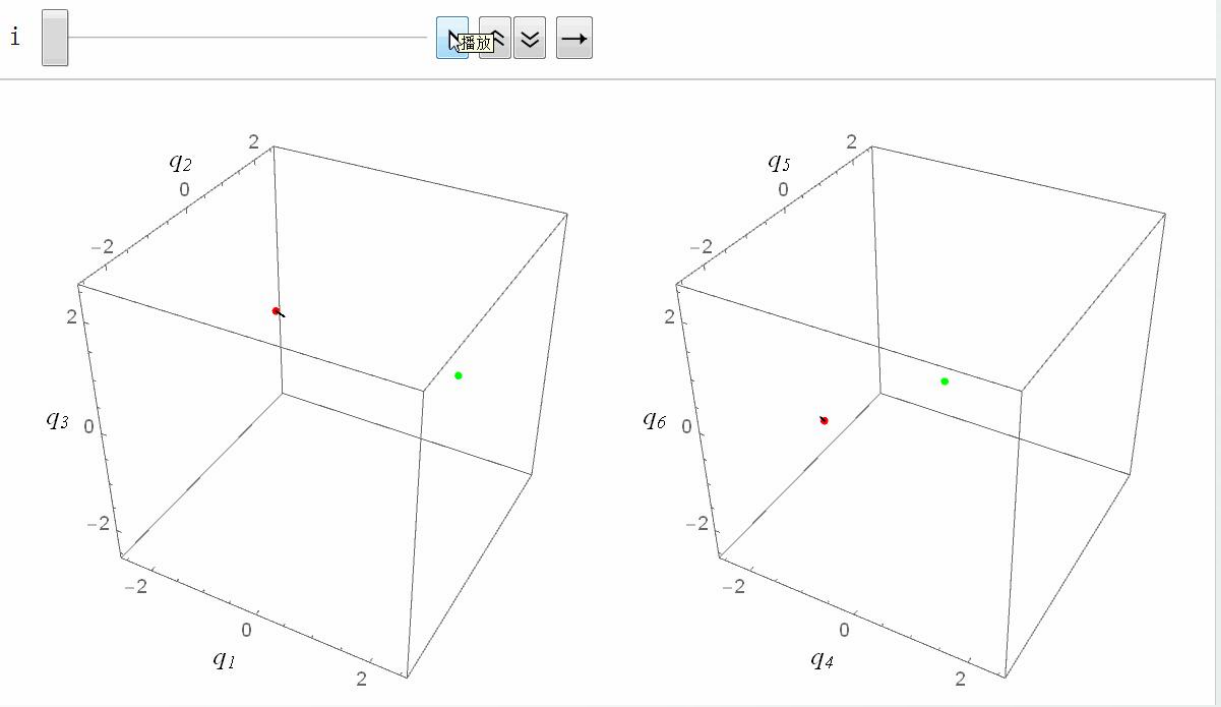
RRT算法的优势

1. 不仅在二维平面上适用，在高维空间也能使用 e.g. robotic arm
2. 是一种通用的方法，不管什么机器人类型、不管自由度是多少、不管约束有多复杂都能用
3. 原理简单
4. 搜索速度快，在大规模地图上优势明显

1. 不仅在二维平面上适用，在高维空间也能使用

2. 是一种通用的方法，不管什么机器人类型、不管自由度是多少、不管约束有多复杂都能用

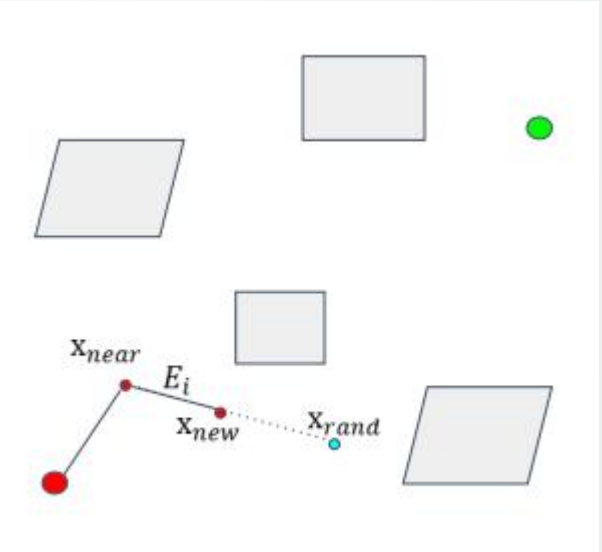




3. 原理简单

RRT 的基本步骤是：

1. 起点作为一颗种子，从它开始生长；
2. 在机器人的搜索空间中生成一个随机点 ；
3. 在树上找到距离最近的点；
4. 朝着最近点的方向生长，如果没有碰到障碍物就把生长后的树枝和端点添加到树上，返回 2；



4. 搜索速度快，在大规模地图上优势明显

睿慕课
AIIMOOC.COM

RRT算法的缺陷

1. 难以在有狭窄通道的环境找到路径

2. 基于采样的算法，本质上还是基于概率的，如果选择了错误的搜索方向，会有找不到目标点的概率。

以上是RRT算法的相关讲解，基于它的优点和缺点，已经有多种改进的RRT算法

RRT-Connect算法

基于RRT搜索空间的盲目性，节点拓展环节缺乏记忆性的缺点

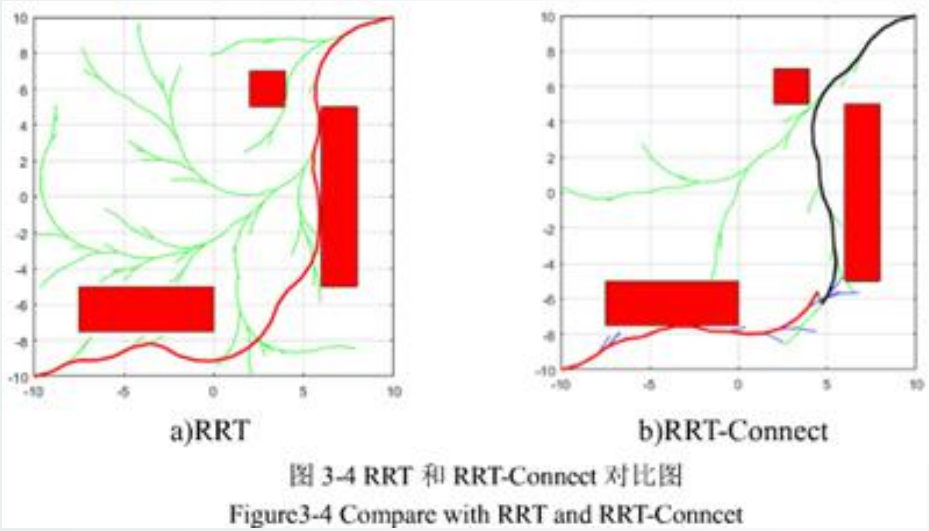
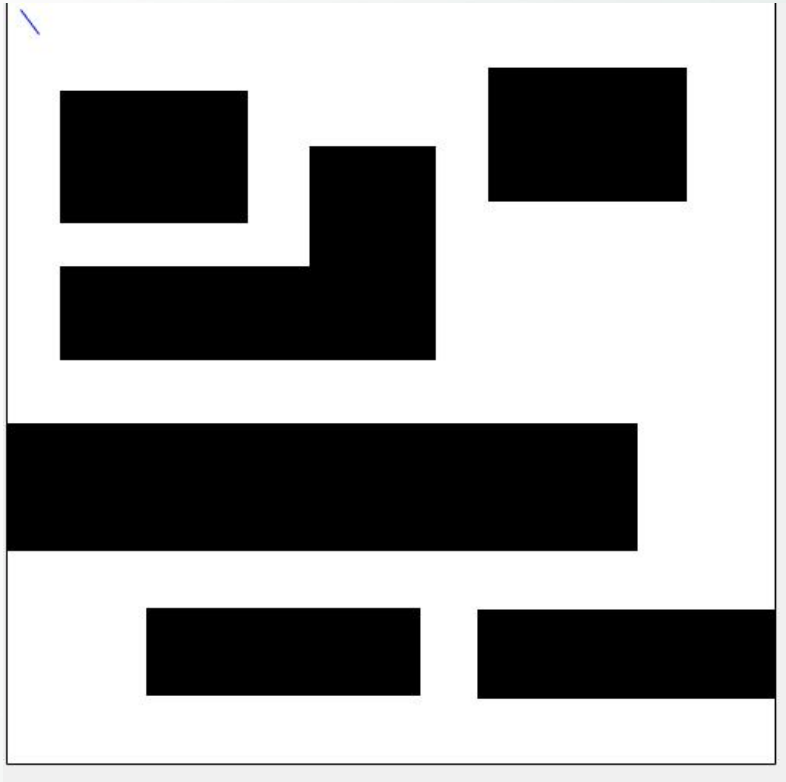


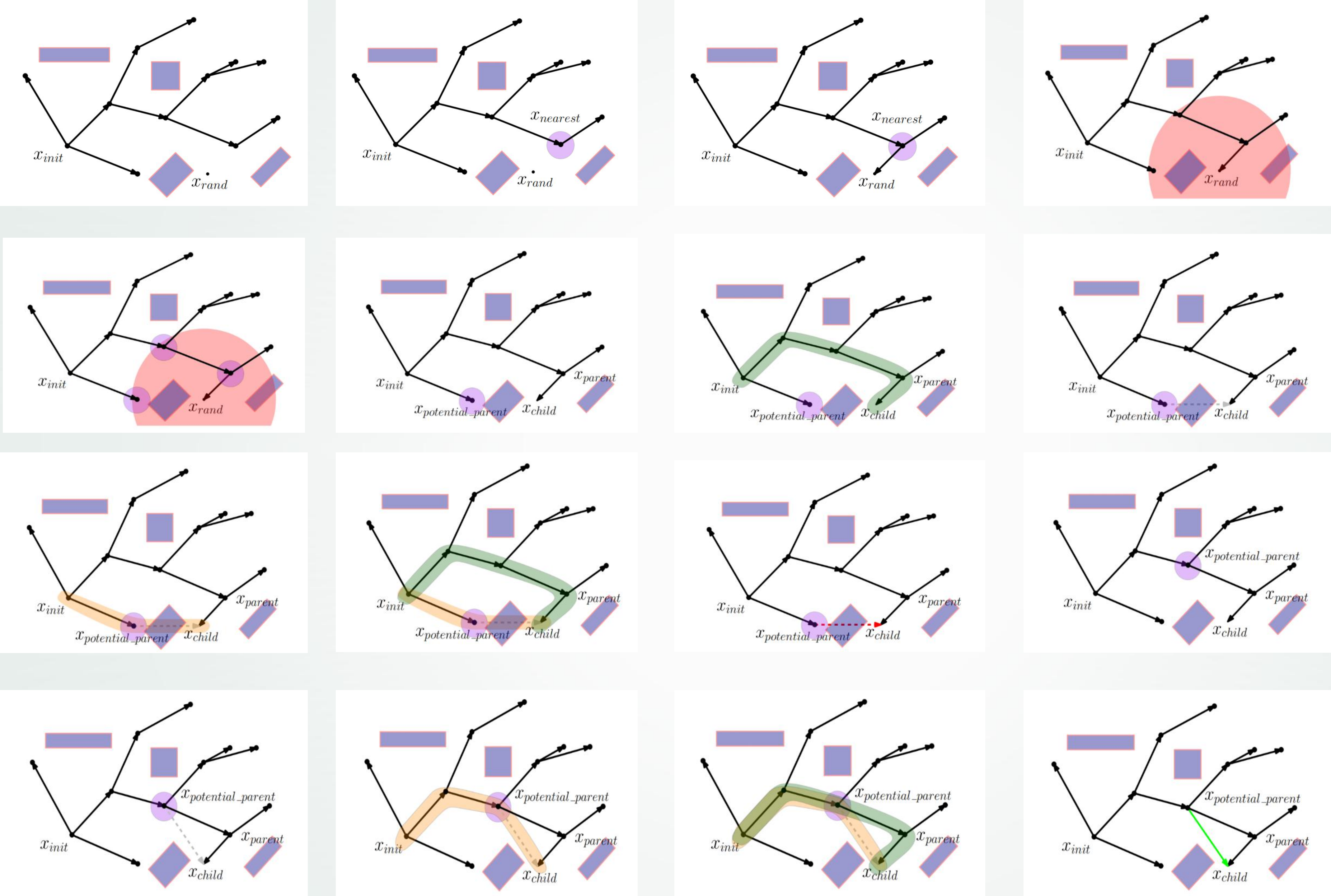
图 3-4 RRT 和 RRT-Connect 对比图

Figure3-4 Compare with RRT and RRT-Connect



RRT*算法，该算法在原有的RRT算法上，改进了父节点选择的方式，采用代价函数来选取拓展节点领域内最小代价的节点为父节点，同时，每次迭代后都会重新连接现有树上的节点，从而保证计算的复杂度和渐进最优解。

睿慕课
AIIMOOC.COM



1. Xrand作为child
2. Xneareast作为parent
3. 连接child和parent
4. 圆圈搜索
5. 潜在parents
6. 任选一个潜在parent
7. parent的cost
8. 连接child和潜在parent
9. 潜在parent的cost1
10. 比较cost和cost1
11. 碰撞检测失败
12. 另选一个潜在parent
13. 连接child和潜在parent
14. 潜在parent的cost2
15. 比较cost和cost2
16. 碰撞检测通过

睿慕课
AIIMOOC.COM

基于图论的全局算法-BFS

什么是BFS?

Best first search 最佳优先算法

Breadth-First-Search 广度优先算法

Bellman Ford Search 贝尔曼福德算法

Best first search 最佳优先算法

原理是每一步都选择当前最好的结点

1.建立一个open list，一个close list；将起始点s放入open list中

2.判断s是否是目标结点g：是，算法结束；否，进入3

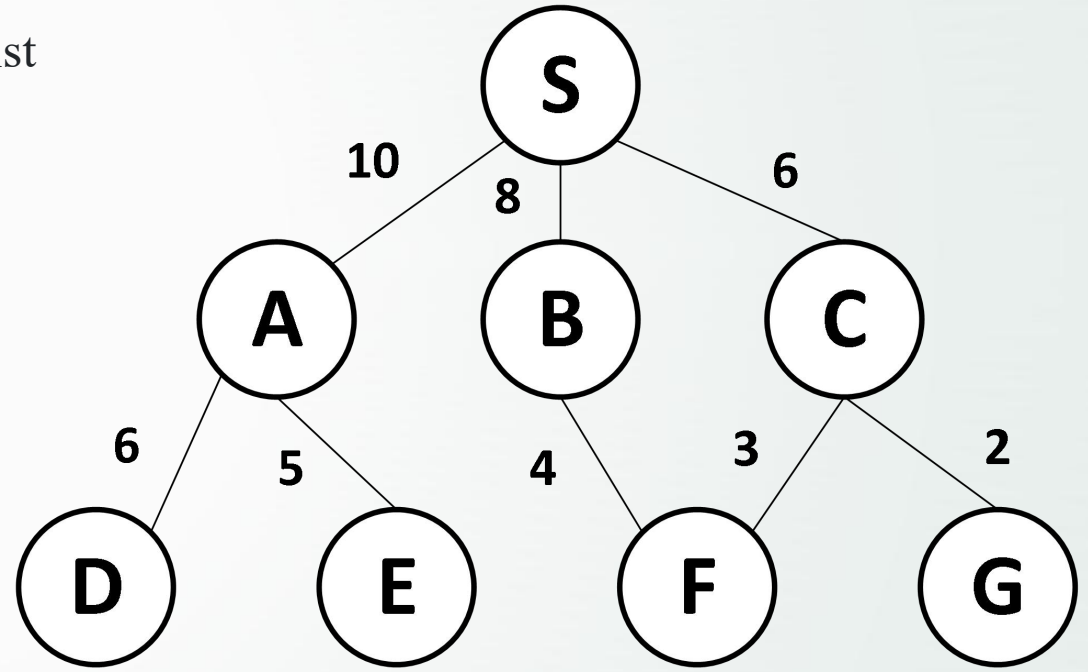
3.S从open list中移出，将s的所有子结点加入open list

4.如果open list=∅ ，输出失败，否则进入5

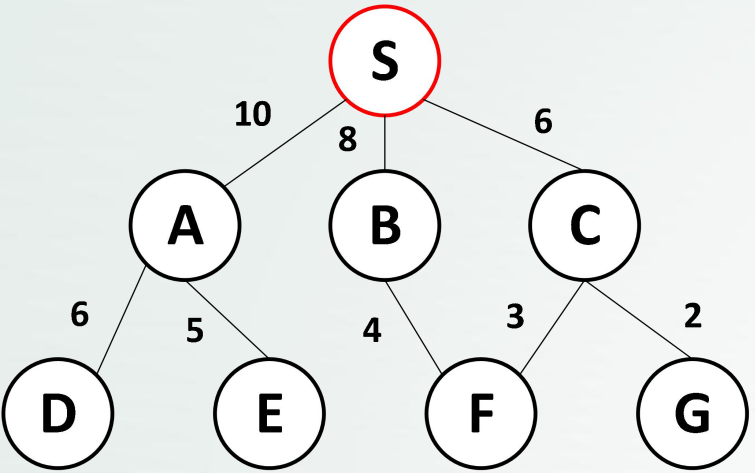
5. 选择open list中cost最小的结点M作为当前父结点，M从open list移除，转入close list，表中其他元素不变；

6. 将M的所有子结点加入open list；

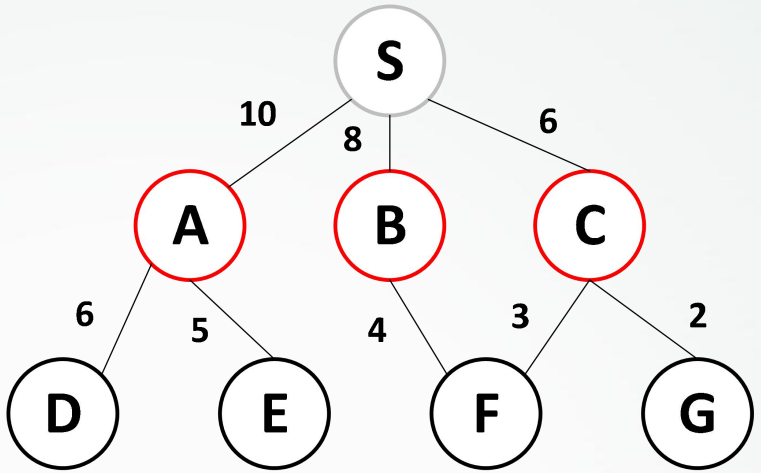
7. 判断open list中是否含有目标结点G，是，G移入close list，算法结束，输出close list即为path；否，返回5



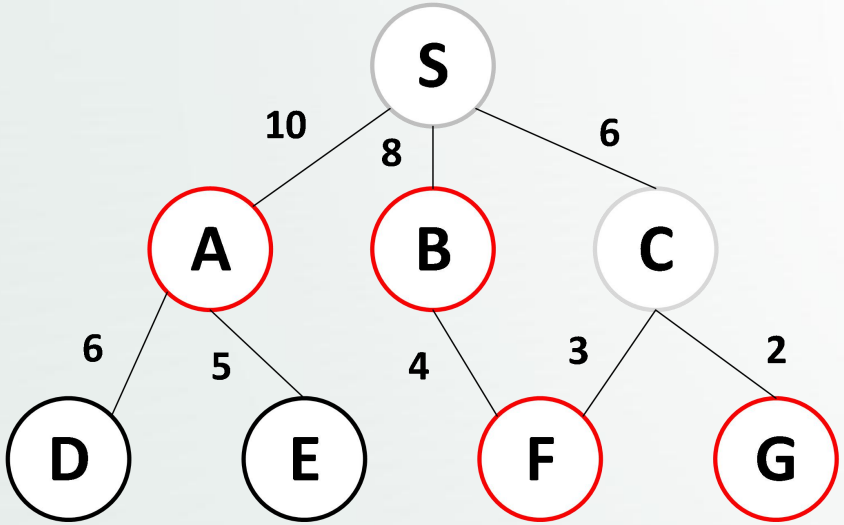
睿慕课 AIIMOOC.COM



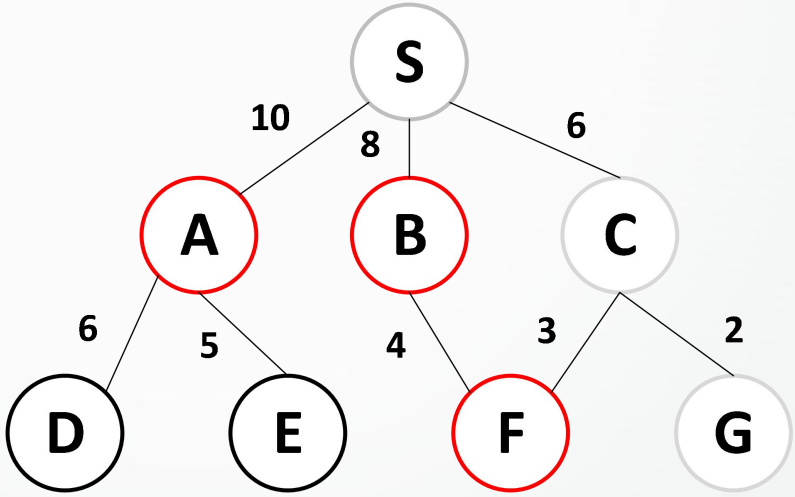
Open list= (S)
Close list= ( )



Open list= (A10, B8, C6)
Close list= (S)

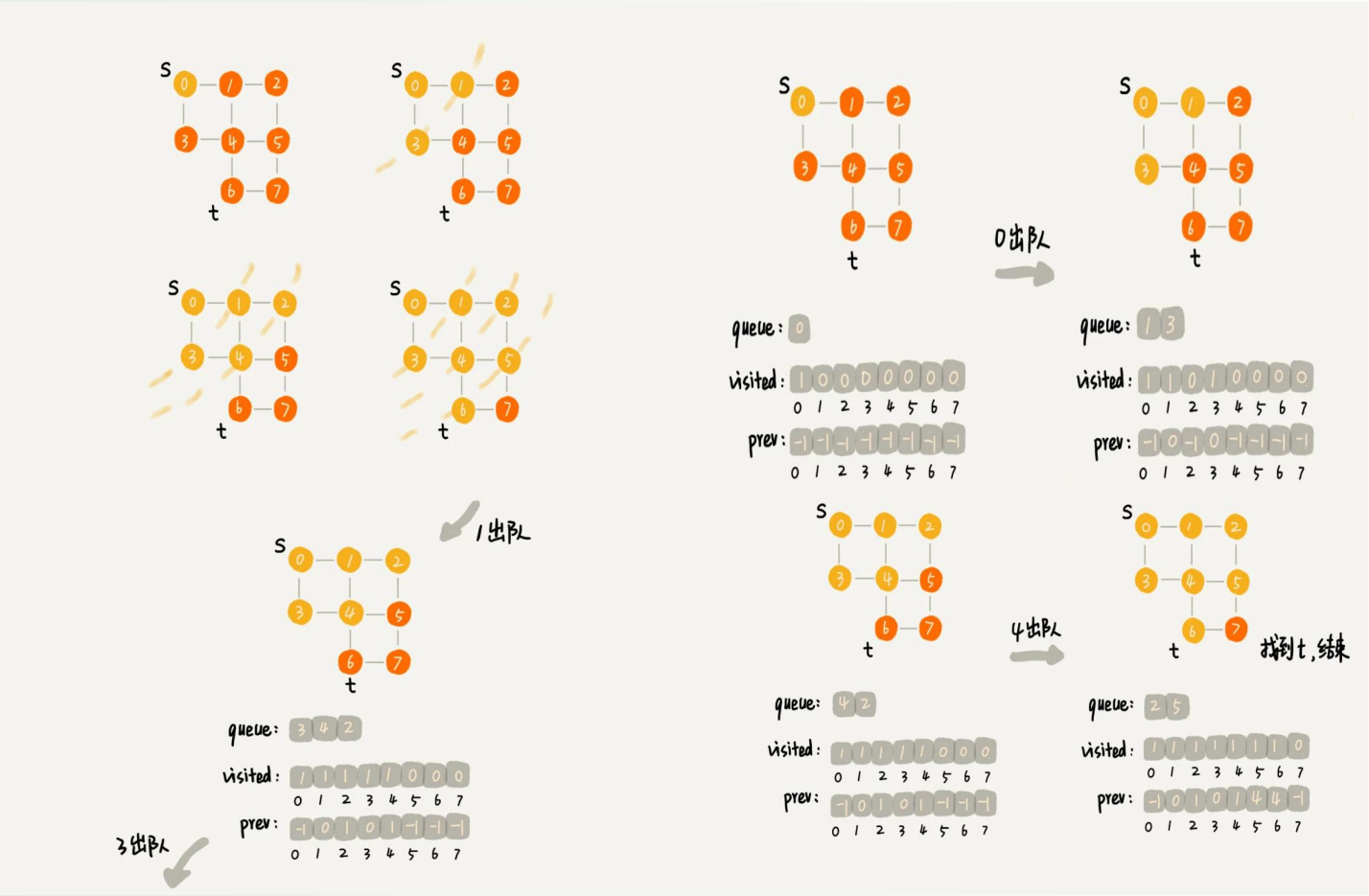


Open list= (A10, B8, F3,G2)
Close list= (S,C)

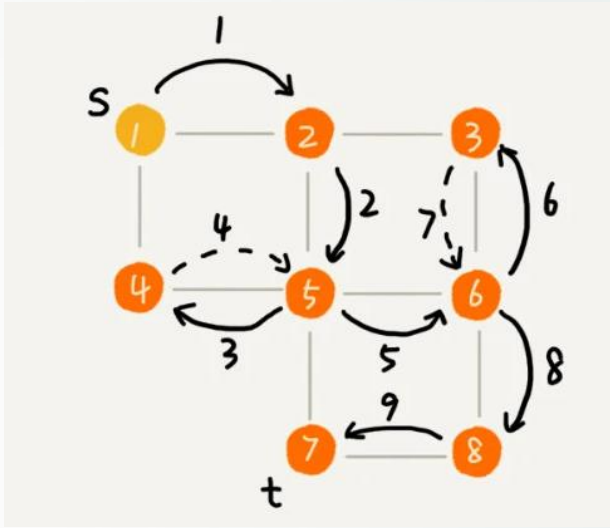


Open list= (A10, B8, F3)
Close list= (S,C,G) Cost=8

Breadth-First-Search 广度优先算法



Depth-First-Search 深度优先算法



最佳优先算法其实是广度优先和深度优先的结合

睿慕课
AIIMOOC.COM

最佳优先算法的优点

结合了广度优先的覆盖力和深度优先的穿透力

最佳优先算法的缺点

不容易找到最佳路径。原因比较明显，其搜索过程比较类似于DP，都属于回溯算法的大类。但是DP的搜索机制决定了它是一种精确算法，找到的一定是最优解；而BFS是一种可用即可行的算法，没有精确性。

Bellman ford search 是对Dijkstra算法的改进

睿慕课
AIIMOOC.COM

Q-Learning

提出： Watkins于1989年在其博士论文中提出

应用：目前应用最为广泛的强化学习算法。应用于动态系统、机器人控制、最优操作工序等领域。在机器人路径规划领域有较为广泛的应用

原因：只关心当前状态和下一状态的集合进行决策。

马尔可夫决策

贝尔曼方程和价值迭代

睿慕课 AIIMOOC.COM

马尔可夫决策

马尔科夫决策（Markov decision process, MDP）是大部分强化学习算法的基础。

MDP refers to the decision maker periodically or continuously observing stochastic dynamic systems with the Markov property, which means that agents make decisions only based on current information and do not use historical experience. According to the observed state at each moment, the agent randomly selects an action as the next action from all available operations, and it arrives at a new state after selecting the action. Then, as the new situation, it selects the next action.

MDP是指决策者周期性地或连续地观察具有马尔可夫性质的随机动态系统，这意味着Agent只根据当前信息而不使用历史经验进行决策。根据每个时刻观察到的状态，代理从所有可用操作中随机选择一个操作作为下一个操作，并在操作完成后到达新状态选择操作。然后，根据新的情况，它选择下一个动作。

睿慕课
AIIMOOC.COM

马尔可夫决策

MDP is usually used to solve sequential decision process problems. According to the degree of dependence on previous information in the decision proccess, MDP can be divided into first-order MDP, second order MDP, etc. First-order MDP means that the decision of the next state only depends on the current state and the selected action; it is independent of all previous states and actions. Correspondingly, second-order MDP means that the decision of the next state only depends on the last two states and the selected action; it is independent of other previous states and actions.

MDP通常用于解决顺序决策过程问题。根据决策过程对先前信息的依赖程度，MDP可分为一阶MDP、二阶MDP等，一阶MDP是指下一状态的决策只依赖于当前状态和所选动作；它独立于以前的所有状态和操作。相应地，二阶MDP意味着下一个状态的决定仅取决于最后两个状态和选定的操作；它独立于以前的其他状态和行动。

睿慕课 AIIMOOC.COM

贝尔曼方程和价值迭代

The dynamic programming method is the crucial technology in RL; there are two primary methods, policy iteration and value iteration, used to update the reward. The value iteration is widely used in algorithms.

动态规划方法是RL的关键技术；有两种主要的方法，策略迭代和价值迭代，用于更新奖励。数值迭代在算法中有着广泛的应用。

值迭代法使用Bellman优化方程迭代计算最优值函数，并根据最优值函数选择相应的动作。每个状态的最优值对应一个动作，所有状态的一系列动作构成最优策略。对于固定策略Π，Bellman方程定义为

$$V^{\pi}(s) = R(s,a) + \gamma \sum_{s' \in S} P_{ss'}(a) V^{\pi}(s')$$

睿慕课 AIIMOOC.COM

贝尔曼方程和价值迭代

$$V^{\pi}(s) = R(s,a) + \gamma \sum_{s' \in S} P_{ss'}(a) V^{\pi}(s')$$

s表示当前状态；
在状态s中执行操作a后，该状态将转移到新状态s’；
R（s，a）是在状态s中执行动作a后的即时奖励；
c是贴现因子，其值介于0和1之间；
Pss’（a）是从状态s过渡到状态s'的概率。

相应地，最佳贝尔曼方程变为

$$V^{*}(s) = \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} P_{ss'}(a) V^{*}(s') \right)$$

假设在当前状态s中找到最优策略p，则最优值函数V*（s）定义为

$$V^{*}(s) = \max_{\pi} V^{\pi}(s)$$

此时得到最佳策略π*

$$\pi^{*}(s) = arg \max_{a \in A} \sum_{s' \in S} P_{ss'}(a) V^{*}(s')$$

睿慕课 AIIMOOC.COM

Algorithm 2 Value Iteration Algorithm

Input: A state transition function specifying $P_{ss'}(a)$

A reward function $R(s,a)$

A discount factor γ

Output: The value function $V(s)$

The approximately optimal policy $\pi(s)$

1: Initialize value function arbitrarily for all states.

2: **repeat** for each episode k

3: **for** each state s **do**

4: $V(s) = \max_{a \in A}[R(s,a)+\gamma \sum_{s' \in S} P_{ss'}(a)V(s')]$.

5: **end**

6: **until** $|V_k(s) - V_{k-1}(s)| < \theta$ (θ is an infinitesimal value).

7: Calculate the optimal policy by

$\pi^*(s)=arg\max_{a \in A}\sum_{s' \in S}P_{ss'}(a)V^*(s')$

自迭代

QL的思路

在马尔可夫环境中根据历史经验选择最优的行为。

QL的思想是

为每个状态-动作对学习动作值函数Q（s，a）。

Q（s，a）的值是在状态s中执行动作a之后获得的累积返回值。

agent直接从历史经验中学习，不需要完全了解环境模型。

当agent做出决策时，只需比较s状态下每个动作对应的Q（s，a）值，就可以确定s状态下的最优策略，而不考虑状态s的后续状态。

简化决策过程

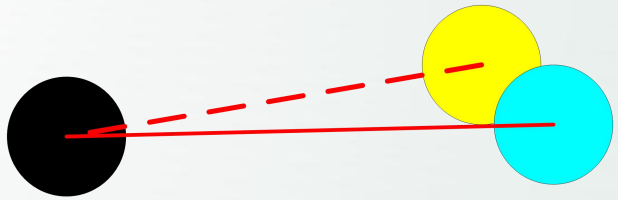
QL的思路

QL算法基于时间差分（TD）算法迭代计算MDP的动作值函数。TD指两个连续时间点之间观察到的相同变量的差异。根据不同的更新规则可以得到不同的TD算法。在QL中使用了TD（0）算法，它只表示agent迭代修改相邻状态的值函数。

TD（0）的更新公式如下

$$V(s) = V(s)+\alpha(r + \gamma V(s') - V(s))$$

其中α是学习率（0＜α＜1），γ是贴现因子。
使用最佳值r+γ V(s')和此时的估计值V（s），以修正估计值。
修正后的估计值用于替换原始估计值V（s），作为当前时刻的最佳值



对更新公式进行修改，得到QL的迭代公式

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha\left(r + \gamma \max_{a'} Q_t(s',a') - Q_t(s,a)\right)$$

Algorithm 3 Q-learning Algorithm

Input: Random state process

Output: Q-table

1: Initialize $Q(s,a)$ arbitrarily for all state-action pairs.
2: **repeat** for each episode k
3: Initialize state s.
4: **repeat** for each step of episode
5: Choose action a from s using policy derived from $Q(e.g.,\varepsilon - greedy,\varepsilon \in (0,1))$.
6: Take action a, observe r, s' .
7: Update $Q(s,a)$ with Eq. 14.
8: $s \leftarrow s'$.
9: **until** s is terminal.
10: **until** convergence($|Q_k(s,a) - Q_{k-1}(s,a)| < \theta$ (θ is an infinitesimal value)).

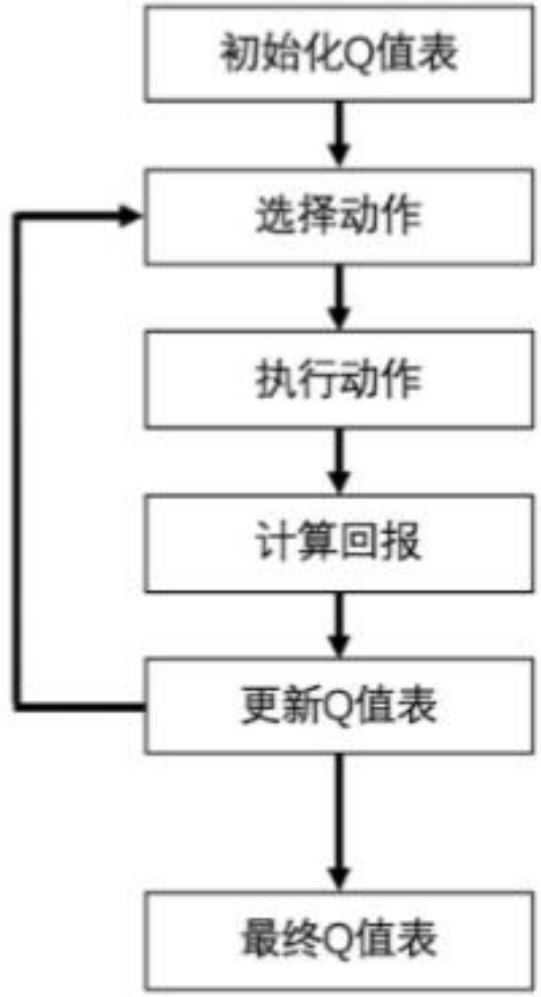


图 1 算法流程图

QL解决路径规划问题实例

睿慕课
AIIMOOC.COM

作业

本章讲解的算法中任选一种，求解单机器人迷宫问题

要求：

1. 建议RRT算法
2. 迷宫尺寸，障碍设置等不做限定，自行设置