DOI:10.19651/j.cnki.emt.2210814

基于双向 F-RRT* 算法的移动机器人路径规划*

杨敏豪[1,2] 张国良[1,2] 李德胜[1,2]
(1.四川轻化工大学自动化与信息工程学院 宜宾 644000; 2.人工智能四川省重点实验室 宜宾 644000)

摘 要: 针对 F-RRT* 算法在狭窄环境和多障碍物复杂环境下搜索效率低的问题,提出一种基于双向搜索的 F-RRT* 算法(BF-RRT*)。以 F-RRT* 算法为基础,首先采用双向搜索结构,双树从起点和终点轮流扩展,使用贪婪启发式引导随机树生长;其次,针对连续扩展过程中产生的冗余点进行消除处理,快速获得低成本路径,有效提高了规划速度;然后引入启发式函数,并对连接点进行优化以提高路径整体质量。最后分别基于 MATLAB 和 Gazebo 仿真平台将改进算法进行了对比实验,结果表明在不同环境下,该算法相较于原算法在迭代次数上平均降低 63.5%,在规划时间上平均降低 88.41%以上,有效提高了规划效率。

关键词: 移动机器人;路径规划;F-RRT*;贪婪扩展
中图分类号: TP242.6 **文献标识码:** A **国家标准学科分类代码:** 520.6

Path planning of mobile robot based on bidirectional F-RRT* algorithm

Yang Minhao[1,2] Zhang Guoliang[1,2] Li Desheng[1,2]
(1. School of Automation and Information Engineering, Sichuan University of Science & Engineering, Yibin 644000, China;
2. Artificial Intelligence Key Laboratory of Sichuan Province, Yibin 644000, China)

Abstract: Aiming at the problem of low search efficiency of F-RRT* algorithm in narrow environment and complex environment with multiple obstacles, a F-RRT* algorithm based on bidirectional search (BF-RRT*) is proposed. Based on the F-RRT* algorithm, firstly, a two-way search structure is adopted, the double tree is expanded from the starting point and the ending point in turn, and the greedy heuristic is used to guide the random tree growth. Secondly, the redundant points generated in the continuous expansion process are eliminated. A low-cost path is obtained, which effectively improves the planning speed; then a heuristic function is introduced, and the connection points are optimized to improve the overall quality of the path. Finally, the improved algorithm is compared based on MATLAB and Gazebo simulation platforms. The results show that compared with the original algorithm, the algorithm reduces the number of iterations by 63.5% on average and the planning time by more than 88.41% in different environments. Improved planning efficiency.

Keywords: mobile robot; path planning; F-RRT*; greedy expansion

0 引 言

移动机器人已经广泛应用于人类生产生活的各个方面,如仓储物流、医疗卫生、迎宾陪护等。作为移动机器人导航中的一项关键技术,路径规划参照某一指标(时间最少、路径最短、能量消耗最低等),在任务区域规划出一条从起点连接到终点的最优或次优的无碰路径[1-2]。根据对环境信息的侧重点不同,可分为全局路径规划和局部路径规划。前者根据全局环境模型的先验信息,在静态环境中找到一条满足最优性的无碰路径;后者侧重于动态避障,在满足机器人非完整约束条件下,利用自身传感器信息,在动态环境中找到一条尽可能符合全局路线的局部路径。全局算法可分为基于几何模型搜索的算法、基于采样的算法和基于生物智能的算法[3]。基于网格分辨率完整性的启发式搜索算法,需要在一个确定性空间中对障碍物进行建模,计算复杂度高,不适用于多障碍物且分布不均匀的复杂环境下的机器人规划问题求解[4]。基于采样的方法可以定义为一种自由空间中构造隐式随机几何图和显式生成树的算法,典型代表有概率路线图(probabilistic roadmaps, PRM)和快速扩展随机树(rapidly exploring random trees, RRT)。相对传统路径规划算法,基于采样的方法具有建模时间短、

收稿日期:2022-07-25
* 基金项目:四川省应用基础研究项目(2019YJ0413)资助

搜索能力强、方便添加非完整约束等优点[5]。

作为一种简单、快速的求解单查询路径规划问题的方法，RRT 算法通过对状态空间中的采样点进行碰撞检测，可以避免构型空间的精确几何建模[6]，具备良好的可扩展性。但上述算法仍然存在缺陷：1)采用随机采样策略，产生大量无用节点，造成计算机资源的浪费。2)产生的路径不够平滑，不满足机器人微分约束条件，导致规划出的路径不能很好地被执行。3)生成的路径非全局最优。

基于上述原因，国内外学者对该算法做出了改进。Karaman 等[7]引入随机几何图理论和剪枝优化，提出渐进最优的 RRT* 算法，但随机采样的不确定性导致收敛缓慢。为解决该问题，Nasir 等[8]提出 RRT*-smart 算法，在关键点周围划定采样区域，Gammell 等[9]将采样区域定义为椭圆子集，缩小采样范围；Jeong 等[10]提出 Quick-RRT*，将三角不等式引入父节点重选和重布线过程。为提高搜索树连接效率，Klemm 等[11]提出基于贪婪搜索的 RRT*-connect 算法；林依凡等[12]采用碰撞风险评估函数代替碰撞检测，提高避碰效率；Lucas 等[13]提出快速行进树(fast marching tree，FMT*)，使用步进的方法处理单组样本，比 RRT* 和 PRM* 更快收敛到最优解；Gammell 等[14]提出 BIT*，统一图搜索和采样技术，实现分批次采样。为改善狭窄通道等环境下的规划效率，张伟民等[15]引入目标偏置采样，结合人工势场法思想，采用三次 B 样条曲线进行平滑处理，使路径更符合非完整约束。为避免不必要的冗余探索，张建冬等[16]引入障碍物因子，吴铮等[17]提出基于方向选择的启发函数，使算法向更有期望的区域扩展。为提升初始解的质量，Liao 等[18]提出偏向障碍物边界的 F-RRT* 算法，相对传统 RRT* 算法，能更快到达高质量解，但随着分辨率 $D_{dichotomy}$ 的提高，计算复杂度增加，一定程度上限制了算法的探索效率，单纯地增加分辨率不能带来收敛速度的有效提升，这种情况在狭窄、复杂环境较为明显。

本文主要针对 F-RRT* 在狭窄环境和多障碍物复杂环境下搜索效率低的问题，提出一种基于双向搜索结构的改进 F-RRT* 算法(bidirectional fast RRT*，BF-RRT*)，采用双向搜索技术结合贪婪扩展，并在扩展过程中剔除产生的冗余节点，引入启发式函数，最后对连接点进行优化处理。通过以上改进，进一步提高找到初始解的速度。最后通过 MATLAB 和 Gazebo 仿真实验验证了改进后算法的有效性、实时性。

1　RRT* 与 F-RRT* 算法

1.1　RRT* 算法

RRT* 算法在 RRT 算法基础上增加了重选父节点和重布线操作，使得算法具备渐近最优性。基本流程为：给定一个状态空间 $C \in R^n$，以起点 X_{start} 作为树的根节点进行初始化，T 代表随机树，V 和 E 分别表示随机树的顶点集和边缘集，N 表示迭代次数。在状态空间 C 内调用 *Sample*() 进行随机采样，得到采样点 X_{rand}，调用 *Nearest*() 函数在搜索树上寻找距离 X_{rand} 最近的节点 $X_{nearest}$，然后连接两点，基于步长 σ 进行碰撞检测，若无碰撞则在连线上以步长 σ 扩展生成新节点 X_{new}。调用 *Near*() 函数以 X_{new} 为圆心，R 为半径，寻找邻域内所有点的集合 X_{near}，在 X_{near} 中重新选取使 X_{start} 到 X_{new} 路径代价最低且无碰撞的父节点 X_{parent}，接着启用 *Rewire*() 函数，对 X_{near} 中的每个节点逐一判断，是否可以将其父节点更新为 X_{new}，如果路径距离减小则对父节点进行更新，否则不变。当树扩展到目标区域 X_{goal} 时，生成一条可行路径。算法原理如图 1 所示。

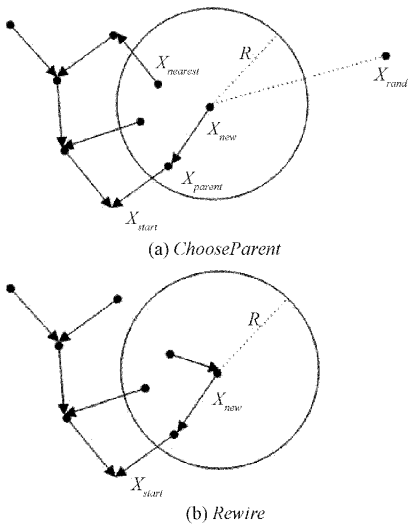

图 1　RRT* 算法优化过程

1.2　F-RRT* 算法

RRT* 算法可以在有限时间内生成一条最优路径，然而，由于随机采样的特性，算法很难采样到障碍物边界点附近，这些点通常需要经过算法的大量迭代才能找到，而最优路径点通常位于该区域，因此 RRT* 算法的收敛速度极为缓慢。可以通过增加半径参数 R 的方法来提升解的质量，因为较大的半径 R 更容易降低成本，然而这也会成倍地增加附近的顶点数和计算时间。

基于上述原因，F-RRT* 算法被提出，该算法基于三角不等式优化原理和二分法。在选择父节点之前，F-RRT* 算法的工作方式与 RRT* 相同。选择父节点时，该算法将父节点重选过程的 *ChooseParent* 函数用 *FindReachest* 和 *CreateNode* 函数替代，具体优化过程如图 2 所示。

受到 Quick-RRT* 算法的启发，*FindReachest* 函数从 $X_{nearest}$ 的 ancestor 节点中寻找候选节点，记为 $X_{reachest}$。显然，基于三角不等式原理，$X_{reachest}$ 和 X_{new} 的连接成本低于连接 $X_{nearest}$ 和 X_{new}。与 Quick-RRT* 不同的是，该算法只搜索 $X_{nearest}$ 的 ancestor，而不是考虑邻域内的所有节点，搜索节点数量的减少使 F-RRT* 有更高的规划效率。选出候选节点 $X_{reachest}$ 后，以参数 $D_{dichotomy}$ 作为终止条件，运用二分

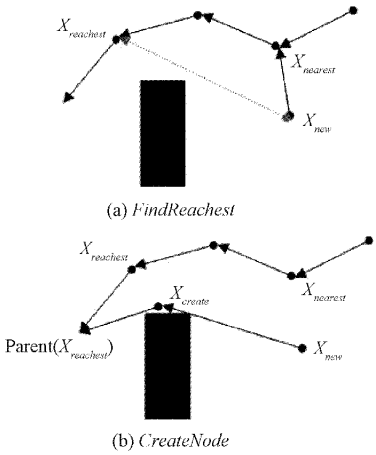

图 2 F-RRT* 优化过程示意图

法在有限时间内创建出一个收敛于障碍物边界的节点,在提高初始解质量的同时,有效缩短了收敛所需时间。

1.3 问题描述

F-RRT* 能够获得比 RRT* 更好的初始解和更快的收敛速度,但单向搜索的结构限制了算法在狭窄以及复杂环境的探索速度。与单向搜索相比,双向方法分别在起始点和目标点构建两棵并行的随机树,在每次迭代过程中,两棵树分别朝着彼此的方向交替扩展,从而降低扩展的盲目性,能够在狭窄、复杂环境中更快地脱离局部困境。在此基础上,RRT-connect 采用一种贪婪扩展策略,实践证明,该方法极大地加快了算法的连接速度,但 RRT-connect 算法缺乏优化过程,无法提供路径的最优解。为提高 F-RRT* 算法在狭窄和复杂环境的表现,提出一种基于双向搜索的改进 F-RRT* 算法(bidirectional fast RRT*,BF-RRT*)。

2 BF-RRT* 算法

2.1 去冗余点的贪婪扩展

将 F-RRT* 算法与双向快速搜索融合的关键在于 $connect(T,new)$ 函数,考虑到 F-RRT* 算法的自身特性,对贪婪扩展方法做出如下改进:

先从树 b 中找到距离 X_{new} 最近的节点 $X'_{nearest}$,由 $X'_{nearest}$ 开始,以每次前进固定步长 σ 的方式向 X_{new} 连续扩展,直到遇到障碍物或到达 X_{new}。不同在于,RRT*-connect 每进行一步扩展都需要完成父节点重选和重连接,该算法则不考虑将扩展过程中的节点加入到树结构进行优化,原因如下:记树 b 的扩展终点为 X'_{new},由于 *FindReachest* 函数尝试向 X'_{new} 的祖节点连接,而贪婪扩展得到的点位于同一条直线,X'_{new} 和 $X'_{nearest}$ 的连线必然不会发生碰撞,过程点冗余,去除后不会对 *CreateNode* 函数的结果产生影响,因此选择跳过冗余点的优化并从树结构中剔除以减少不必要的重连接和碰撞检测过程,如图 3 所示。

2.2 启发式函数

由于随机采样的不确定性,随机树可能不会向提供更好解决方案的方向扩展,此时需要引入合理的启发式函数避免不必要的冗余探索。假设已找到新的节点 X_{new},在进行碰撞检测之前,先使用启发式函数对节点进行判断,以保证当前节点可以改善路径质量。启发式函数如式(1)所示。

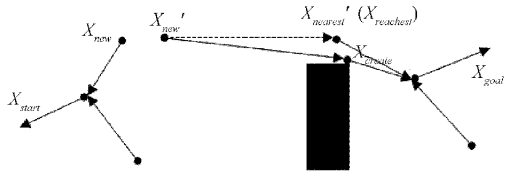

图 3 扩展过程示意图

$$f(n)=g(n)+h(n) \tag{1}$$

路径的启发式代价估计被划分为两部分,其中,$g(n)$ 是由起始点 X_{start} 到达 X_{new} 的当前实际代价,$h(n)$ 是从 X_{new} 到目标点 X_{goal} 的最小代价估计,采用欧氏距离进行度量。令当前最短可行路径的代价值为 $cbest$,当 $f(n)>cbest$,意味着当前节点 X_{new} 不能带来更好的解决方案,该节点被拒绝,跳过碰撞检测并重新采样。通过以上的启发式判断可以减少不必要的碰撞检测,加快算法的收敛过程。

2.3 连接点处理

在两棵树产生相遇时,如果不对连接点进行处理,产生的节点远离障碍物,所得到的路径非全局最优。因此,对连接点做如下优化:首先针对树 a,令 X_{new} 为父节点,子节点选取 X'_{new},启用 *FindReachest* 和 *CreateNode* 函数对 X_{new} 所在位置进行优化;然后针对树 b,令 X_{new} 为子节点,X'_{new} 选为父节点,做相同操作,使最终生成的整条路径在同伦类中达到最优。优化前和优化后结果如图 4 和 5 所示。

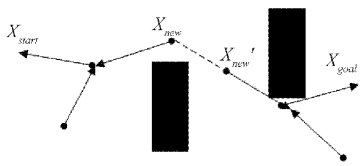

图 4 连接点优化前

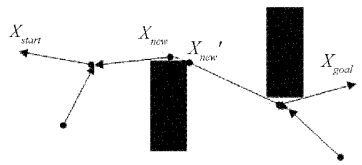

图 5 连接点优化后

BF-RRT* 算法伪代码如算法 1 所示。

算法 1 BF-RRT*

$Va \leftarrow X_{start}$;$Vb \leftarrow X_{goal}$;$Ea \leftarrow \emptyset$;$Eb \leftarrow \emptyset$;$cbest \leftarrow \infty$;
for $i=1$ to N **do**
  $X_{rand} \leftarrow Sample(i)$;
  $N_{nearest} \leftarrow N_{earest}(Ta, X_{rand})$;
  $N_{new} \leftarrow steer(N_{nearest}, X_{rand})$;

if $f(N_{new}) < cbest$ && $CollisionFree(N_{nearest}, N_{new})$ **then**
　$X_{near} \leftarrow Near(Ta, N_{new})$;
　$X_{reach} \leftarrow FindReachest(Ta, N_{nearest}, N_{new})$;
　$X_{create} \leftarrow CreatNode(Ta, X_{reach}, N_{new}, Ddichotomy)$;
　if $Xcreate \neq \emptyset$ **then**
　　$Va \pm X_{create}, N_{new}$;
　　$Ea \pm (Parent(X_{reach}), X_{create}), (X_{create}, N_{new})$;
　else
　　$Va \leftarrow Va \cup N_{new}$;
　　$Ea \leftarrow Ea \cup (X_{reach}, N_{new})$;
　end if
　$Ta \leftarrow Rewire(Ta, N_{new}, X_{near})$;
　$T \leftarrow connect(T, N_{new})$;
　$swap(Ta, Tb)$;
end if
end for
return $T = (V, E)$;

3　仿真实验与分析

为验证 BF-RRT* 算法的有效性，分别在 MATLAB 和 Gazebo 仿真环境中进行了实验测试，将该算法与 RRT*、RRT*-connect、F-RRT* 算法进行对比，并考虑到移动机器人的自身半径，对障碍物进行了膨化处理，将机器人看作一个质点。仿真实验平台及配置为：MATLAB R2018b，Gazebo，64 位 Windows10，处理器 AMD Ryzen 5 3500U，CPU 主频 2.10 GHz，内存 8 GB。

3.1　MATLAB 实验验证

实验设计了 3 种特殊环境进行仿真和分析，所有的地图规模均为[100，100]。考虑到 RRT 系列算法的随机性，在每个环境中对每种算法采用相同的伪随机数，分别执行 50 次重复实验。参数设置如下：步长 $\sigma=3$，搜索半径 $R=10$，$D_{dichotomy}=1$。三种环境下的规划结果如图 6～8 所示。

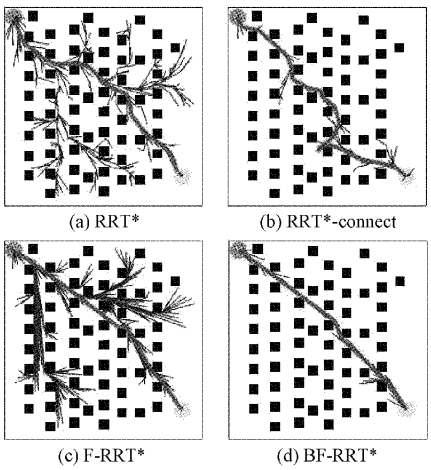
(a) RRT*　(b) RRT*-connect
(c) F-RRT*　(d) BF-RRT*

图 6　环境 1 规划结果

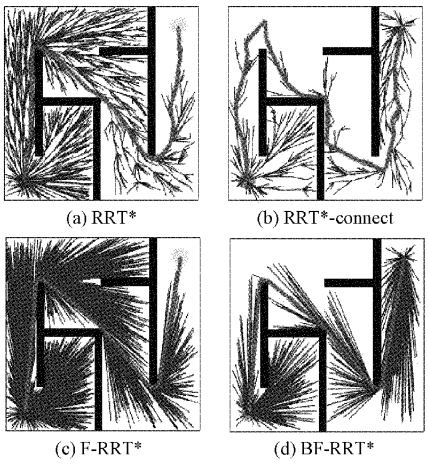
(a) RRT*　(b) RRT*-connect
(c) F-RRT*　(d) BF-RRT*

图 7　环境 2 规划结果

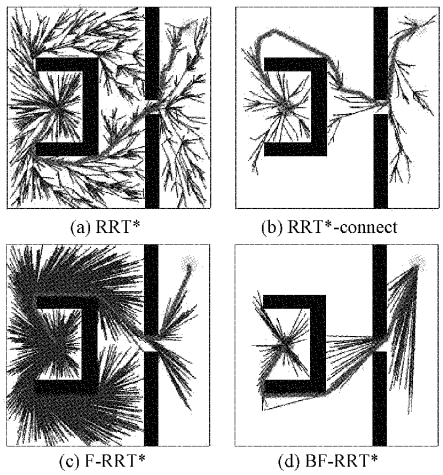
(a) RRT*　(b) RRT*-connect
(c) F-RRT*　(d) BF-RRT*

图 8　环境 3 规划结果

图 6～8 中紫色且较粗的线条为算法规划后产生的路径，蓝色较细的线条为扩展树的分支。综合 3 种环境下的结果可以看出，F-RRT* 和 BF-RRT* 生成的路径都与障碍物边缘紧密贴合，此时路径解收敛于最优；BF-RRT* 在保持和 RRT*-connect 的节点数量接近的同时，也更靠近最优解。

以找到初始解时的迭代次数作为评价指标，如表 1 所示。可以看出，在 3 种环境下，BF-RRT* 算法在该指标上均优于 RRT* 和 F-RRT* 算法，迭代次数分别下降 66% 和 63.5%，数量上与 RRT*-connect 算法接近，但从图 6～8 的实验结果可以看出，本文提出的 BF-RRT* 算法所获得的路径质量远高于 RRT*-connect 算法。

一般而言，算法找到初始解的速度越快，说明算法的探索效率越高；其次，初始解的质量对后续的收敛速度也有一定的影响。因此，选取初始路径长度 l_{init} 和初始路径时间 t_{final} 作为初始解的评价指标，以找到 1.05 倍最优路径的时间 $t_{5\%}$ 作为收敛速度的衡量标准。实验结果如表 2 所示。

表 1　4 种算法在不同环境下的平均迭代次数

场景	算法	平均迭代次数
环境 1	RRT*	545
	RRT*-connect	59
	F-RRT*	482
	BF-RRT*	65
环境 2	RRT*	2 889
	RRT*-connect	1 390
	F-RRT*	2 893
	BF-RRT*	1 393
环境 3	RRT*	1 327
	RRT*-connect	530
	F-RRT*	1 164
	BF-RRT*	556

表 2　4 种算法在不同环境下的性能比较

场景	算法	l_{init}/m	t_{find}/s	$t_{5\%}$/s
环境 1	RRT*	134.08	0.93	14.06
	RRT*-connect	125.98	0.09	1.91
	F-RRT*	130.71	0.69	9.45
	BF-RRT*	121.23	0.08	0.08
环境 2	RRT*	237.41	13.66	28.61
	RRT*-connect	260.72	0.83	12.29
	F-RRT*	223.98	13.62	13.62
	BF-RRT*	224.10	0.82	0.82
环境 3	RRT*	161.14	3.09	36.99
	RRT*-connect	169.76	0.24	21.76
	F-RRT*	139.14	2.46	2.46
	BF-RRT*	139.14	0.26	0.26

由表 2 可知,环境 1 中,BF-RRT* 算法相较于 F-RRT*,初始路径长度缩短了 9.48 m,找到初始路径的时间减少了 88.41%,找到次优路径的时间减少了 99.15%;对比 RRT* 算法,路径长度缩短了 12.85 m,初始路径时间降低了 91.39%,次优路径时间降低了 99.43%;相比 RRT*-connect 算法,路径长度缩短了 4.75 m,次优解时间减少了 95.81%。

对比各算法在环境 2 中的表现,由表 2 可知,相比 F-RRT* 算法,BF-RRT* 搜索时间减少了 93.98%;对比 RRT*-connect 算法,BF-RRT* 初始路径长度缩短了 36.74 m,次优解时间降低了 97.13%;相较于 RRT* 算法,BF-RRT* 长度缩短了 13.43 m,初始解和次优解时间分别减少 94%和 97.13%。

分析表 2 关于环境 3 的数据不难发现,BF-RRT* 在狭窄地图和 U 形环境中同样表现良好,和 F-RRT* 相比,在初始路径长度相同的基础上,时间缩短了 90.24%。对比其余算法,路径长度分别减少 22 m 和 30.62 m,初始时间分别下降 92.23%和 0.08%,次优解时间分别降低 99.3%以及 98.81%。

为直观地比较各算法生成的路径长度随时间的变化,实验以 50 次重复规划产生的平均值作为最终结果,相同时间下的路径长度如图 9～11 所示。

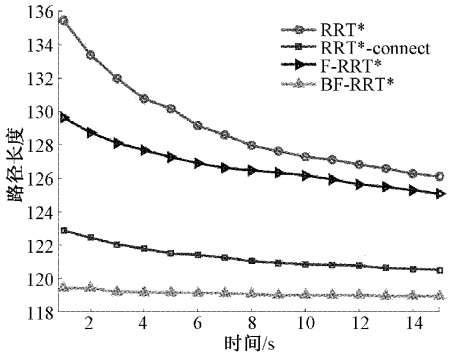

图 9　环境 1 路径长度对比

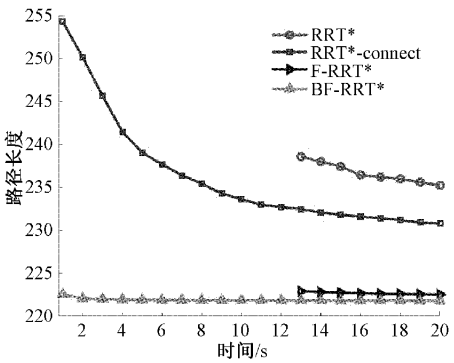

图 10　环境 2 路径长度对比

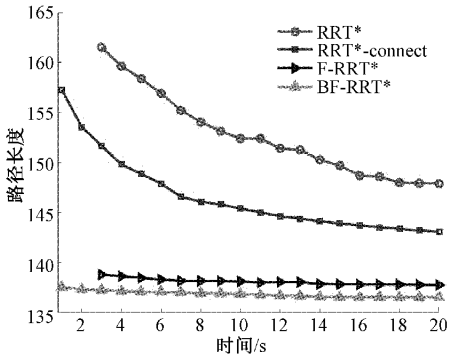

图 11　环境 3 路径长度对比

从图 9～11 中可以看出,BF-RRT* 能够很快地生成初始路径,并且在任意时间,BF-RRT* 算法得到的路径长度均优于其余 3 种算法。

综上所述,针对不同环境,相比其他 3 种算法,BF-RRT* 有效提高了搜索和采样效率,能够在狭窄、复杂环境中更快地找到更好的解决方案。

3.2　Gazebo 实验验证

为进一步验证 BF-RRT* 算法,本文采用 Gazebo 物理

仿真平台，运行在 Ubuntu16.04 操作系统下，ROS 版本为 Kinetic。首先搭建图 12 所示的仿真环境，模拟了一个边长为 10 m 的正方形房间。实验中依次启动 Gazebo 物理仿真环境和二维代价地图，利用 Rviz 可视化工具为移动机器人设置目标点并显示规划结果。

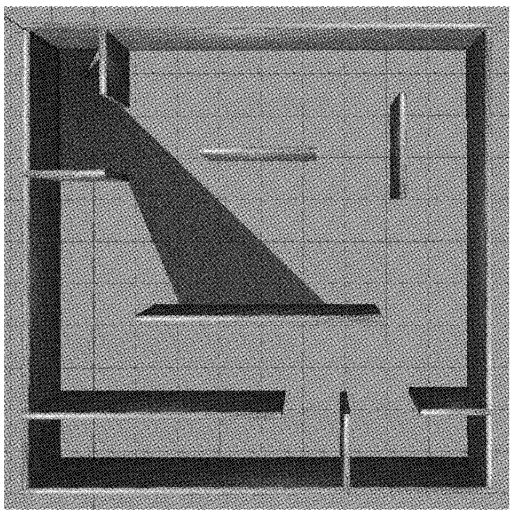

图 12　Gazebo 仿真环境

以左上角为起点，右下角为终点，针对图 12 场景分别对 4 种算法进行了 10 次重复规划，并对规划时间和路径长度做了记录，取平均值进行分析。参数设置如下：步长 $\sigma=0.5$ m，半径 $R=1$ m，分辨率 $D_{dichotomy}=0.1$ m。结果如表 3 所示。

表 3　Gazebo 仿真结果

算法	规划时间/s	路径长度/m
RRT*	15.02	15.92
RRT*-connect	0.59	16.22
F-RRT*	10.86	12.83
BF-RRT*	0.57	12.80

通过表中的相关数据可以看出，相比 F-RRT* 算法，BF-RRT* 路径长度基本相同，而规划路径的时间缩短了 94.75%；对比 RRT* 算法，规划时间缩短 96.21%，路径长度减少 19.6%；对比 RRT*-connect 算法，规划时间和路径长度分别下降 0.03%和 21.09%。

4　结　　论

针对 F-RRT* 在狭窄、复杂环境探索缓慢的问题，提出一种基于双向搜索的改进 F-RRT* 算法 BF-RRT*。首先采用双向搜索结构，双树从起点和终点轮流扩展，使用贪婪启发式引导随机树生长；其次针对连续扩展过程中产生的冗余点进行消除处理，加快连接速度；最后引入启发式函数，并对连接点进行优化，缩短路径长度。经过 MATLAB 和 Gazebo 实验验证，证明相对 F-RRT* 算法，BF-RRT* 算法有效提高了搜索效率，具有高效的计算速度和良好的路径质量。后续工作将更多地考虑机器人自身非完整约束，将该算法与局部规划算法配合，以便更好地满足工程实践的需要。

参考文献

[1] 霍凤财，迟金，黄梓健，等. 移动机器人路径规划算法综述[J]. 吉林大学学报(信息科学版)，2018，36(6)：639-647.

[2] 李志锟，黄宜庆，徐玉琼. 改进变步长蚁群算法的移动机器人路径规划[J]. 电子测量与仪器学报，2020，34(8)：15-21.

[3] 谭宝成，宋沽. 蚁群算法在无人驾驶智能车中的应用及改进[J]. 国外电子测量技术，2012，31(9)：15-17，30.

[4] 杜明博，梅涛，陈佳佳，等. 复杂环境下基于 RRT 的智能车辆运动规划算法[J]. 机器人，2015，37(4)：443-450.

[5] 陈秋莲，蒋环宇，郑以君. 机器人路径规划的快速扩展随机树算法综述[J]. 计算机工程与应用，2019，55(16)：10-17.

[6] WANG J，MENG M Q H，KHATIB O. EB-RRT：Optimal motion planning for mobile robots[J]. IEEE Transactions on Automation Science and Engineering，2020，17(4)：2063-2073.

[7] KARAMAN S，FRAZZOLI E. Sampling-based algorithms for optimal motion planning[J]. The International Journal of Robotics Research，2011，30(7)：846-894.

[8] NASIR J，ISLAM F，MALIK U，et al. RRT*-SMART：A rapid convergence implementation of RRT[J]. International Journal of Advanced Robotic Systems，2013，10(7)：299，DOI：10.5772/56718.

[9] GAMMELL J D，SRINIVASA S S，BARFOOT T D. Informed RRT*：Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic[C]. 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems，IEEE，2014：2997-3004，DOI：10.1109/IROS.2014.6942976.

[10] JEONG I B，LEE S J，KIM J H. Quick-RRT*：Triangular inequality-based implementation of RRT* with improved initial solution and convergence rate[J]. Expert Systems with Applications，2019，123：82-90，DOI：10.1016/j.eswa.2019.01.032.

[11] KLEMM S，OBERLÄNDER J，HERMANN A，et al. Rrt-connect：Faster，asymptotically optimal motion planning[C]. 2015 IEEE International Conference on Robotics And Biomimetics(ROBIO)，IEEE，2015：1670-1677，DOI：10.1109/ROBIO.2015.7419012.

[12] 林依凡，陈彦杰，何炳蔚，等. 无碰撞检测 RRT* 的移动机器人运动规划方法[J]. 仪器仪表学报，2020，

41(10):257-267.

[13] LUCAS J, EDWARD S, ASHLEY C, et al. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions[J]. The International Journal of Robotics Research, 2015, 34(7): 883-921.

[14] GAMMELL J D, SRINIVASA S S, BARFOOT T D. Batch informed trees (BIT): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs[C]. 2015 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2015: 3067-3074, DOI: 10.1109/ICRA.2015.7139620.

[15] 张伟民,付仕雄.基于改进 RRT* 算法的移动机器人路径规划[J].华中科技大学学报(自然科学版),2021,49(1):31-36.

[16] 张建冬,王东,马立东,等.基于改进 RRT 算法的移动机械臂路径规划[J].电子测量技术,2021,44(23):48-53.

[17] 吴铮,陈彦杰,何炳蔚,等.基于方向选择的移动机器人路径规划方法[J].计算机集成制造系统,2021,27(3):672-682.

[18] LIAO B, WAN F, HUA Y, et al. F-RRT*: An improved path planning algorithm with improved initial solution and convergence rate[J]. Expert Systems with Applications, 2021, 184: 115457, DOI: 10.1016/j.eswa.2021.115457.

作者简介

杨敏豪,硕士研究生,主要研究方向为移动机器人路径规划。

E-mail:ymh1664312751@163.com

张国良(通信作者),工学博士,教授,主要研究方向为先进控制理论、组合导航、机器人技术。

E-mail:zhgl@sohu.com