

基于改进BIT*的路径规划算法

彭亮,代越洋,曾志鹏

(长沙理工大学电气与信息工程学院,湖南长沙410114)

摘要:为解决BIT*算法在路径规划中出现的高采样复杂度、高采样随机性、过多的迭代次数以及对样本数量的依赖等问题,提出一种改进的BIT*算法。该算法在初始阶段采用区域限制以获取高质量的采样点,提高采样点质量,降低算法对样本数量的依赖。此外,引入动态概率机制,允许在启发式搜索策略和目标导向策略之间进行适应性转换,从而增强树的智能并减少无效扩展,极大提高了搜索效率。为了进一步节省时间,设置迭代停止条件,减少无用迭代。最后,通过结合路径回溯、线性插值和均值滤波等方法以平滑路径。实验结果表明,改进后的BIT*算法相比传统的BIT*算法运行时间减少48.77%~50.71%,初始路径减少10.33%~13.56%,最终路径减少0.29%~0.88%,在路径搜索效率和收敛速度上明显优于原有算法。

关键词: BIT*算法;区域限制;动态概率;目标导向;平滑路径

DOI: 10.11907/rjdk.242057

扫描二维码阅读全文:



中图分类号: TP18;TP242

文献标识码: A

文章编号: 1672-7800(2026)002-0119-07

Path Planning Algorithm Based on Improved BIT*

PENG Liang, DAI Yueyang, ZENG Zhipeng

(School of Electrical and Information Engineering, Changsha University of Science and Technology, Changsha 410114, China)

Abstract: To address the issues of high sampling complexity, high sampling randomness, excessive iteration counts, and dependence on sample quantity in the BIT* (Batch Informed Tree) algorithm for path planning, an improved BIT* algorithm is proposed. In the initial stage, the algorithm employs regional restrictions to obtain high-quality sampling points, enhance the quality of sampling points, and reduce the dependence of the algorithm on sample quantity. Additionally, a dynamic probability mechanism is introduced to enable adaptive transitions between heuristic search strategies and goal-oriented strategies, thereby enhancing the intelligence of the tree and reducing ineffective expansions, significantly improving the search efficiency. To further save time, iteration stop conditions are set to reduce futile iterations. Finally, the path is smoothed by combining methods such as path backtracking, linear interpolation, and mean filtering. Experimental results show that compared with the traditional BIT* algorithm, the running time of the improved BIT* algorithm is reduced by 48.77%~50.71%, the initial path is reduced by 10.33%~13.56%, and the final path is reduced by 0.29%~0.88%, it significantly outperforms the original algorithm in terms of path search efficiency and convergence speed.

Key Words: BIT* algorithm; regional restrictions; dynamical probability; goal-oriented; path smoothing

0 引言

在科技日新月异的时代,机器人已广泛应用于交通导航、工业生产、拍照摄影等行业,其发展态势备受人们关注。其中,路径规划技术是机器人研究领域的重要组成部分^[1]。路径规划算法主要有三大类:基于搜索的Dijkstra算法^[2]、A*算法^[3]等,基于采样的RRT^[4]、RRT*^[5]等算法,

以及基于智能算法的麻雀算法^[6]和蚁群算法^[7]等。

RRT*算法通过重写父节点的机制实现了渐进最优性,有效提升了路径质量但效率较低^[8]。文献[9]提出了Informed-RRT*算法,引入了椭圆区域的约束,提升了算法效率。然而,由于每次迭代需要额外计算和优化,算法耗时较多^[10]。文献[11]提出了BIT*(Batch Informed Tree)算法,它采用Informed RRT*中的采样方法,对多批点进行采样,被视为越来越密集的隐式随机几何图。之后,BIT*使

收稿日期:2025-01-13

作者简介: 彭亮(1976-),男,博士,长沙理工大学电气信息工程学院讲师、硕士生导师,研究方向为控制理论与控制工程、模式识别与智能系统;代越洋(2000-),男,长沙理工大学电气信息工程学院硕士研究生,研究方向为无人机路径规划;曾志鹏(2000-),男,长沙理工大学电气信息工程学院硕士研究生,研究方向为无人机路径规划。本文通讯作者:彭亮。

用增量搜索技术处理按潜在解决方案质量排序的可能边,其中包括可接受的启发式方法。然而,当缺乏有效路径时,BIT*的全局批量采样仍然存在一定的盲目性,在路径优化阶段也需要进行大量迭代才能获得最优解。文献[12]提出了GBIT*算法,该算法在BIT*算法中引入贪婪搜索,具有更快的初始解和收敛,但在小样本采样时,BIT*算法的成功率相对较低。文献[13]提出膨化障碍物并对路径进行优化,但对BIT*本身算法性能提高较少。文献[14]提出将随机批量采样改为均匀等间距采样,提高了算法成功率,但对于节点数量没有加以控制,导致生成无用节点较多,采样复杂度提高。文献[15]在Informed-RRT*中融合APF算法,进一步优化了采样点的质量,加快了收敛速度。文献[16]提出BIT*算法结合A*算法和Informed-RRT*算法,可以在更短的时间内找到更短的路径,且长度变化稳定,但是算法在小样本的情况下成功概率较低。文献[17]通过改进批量采样的位置改进BIT*算法,提高了路径规划效率,缩短了路径长度,但没有考虑批量采样数量对算法过度采样的影响。文献[18]提出了实时避障的BIT*,并将其应用于采摘机器人,有效实现了实时避障功能。文献[19]提出了RRT算法与A*算法的融合算法,利用RRT算法快速搜索路径,再使用A*算法的局部优化路径,有效缩短了路径长度并减少了消除拐点,其虽然提升了搜索速度,但RRT的随机性导致路径并非最优。

鉴于此,本文提出了一种改进的BIT*算法。首先,通过引入区域批量采样,替代了全局批量采样,显著提升了采样质量,减少了因随机性带来的无用节点生成,同时降低了对采样数量的依赖。这一改变使得算法在复杂环境中能够保持较高的效率和鲁棒性。其次,在节点扩展过程中引入了基于启发式函数和目标导向的随机选择机制,从而加快了收敛速度。此外,针对算法迭代过程中的时间消耗问题,本文方法将限制优化次数,减少了无用迭代的发生,提高了算法整体效率。

1 基本算法

1.1 Informed-RRT* 算法

Informed-RRT*算法在RRT*算法的基础上,通过引入椭圆区域约束以优化采样过程,从而提高采样点质量,并增强路径收敛速度和效果。其核心理念是限制采样区域,提升路径规划效率与质量。在二维空间中,椭圆区域的定义需要满足3个条件:焦点、长轴和短轴,如图1所示。其中,以路径起点与路径终点的直线距离 $c_{\min}$ 作为椭圆的焦距;以当前路径长度 c_{best} 作为椭圆的长轴;短轴 c_{minaxis} 则由 $c_{\min}$ 与 c_{best} 共同决定,如式(1)所示。

$$c_{\text{minaxis}} = \sqrt{c_{\text{best}}^2 - c_{\min}^2} \quad (1)$$

在路径回溯过程中,路径长度 c_{best} 会逐渐变小,因此 c_{minaxis} 也会随之减小,整个椭圆空间会随着 c_{best} 的变小而不

断压缩,收敛速度会随之加快。

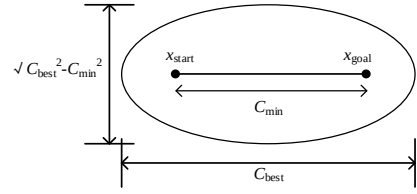


Fig. 1 Elliptic sampling space

图1 椭圆采样空间

Informed-RRT*算法的采样是在椭圆空间中,世界坐标系下椭圆内采样样本 $\mathbf{x}_{\text{ellipse}} \sim U(\mathbf{X}_{\text{ellipse}})$ 是由单位圆内的采样样本 $\mathbf{x}_{\text{ball}} \sim U(\mathbf{X}_{\text{ball}})$ 经过转换矩阵、旋转矩阵和平移矩阵而来。转换公式如式(2)所示。

$$\mathbf{x}_{\text{ellipse}} = \mathbf{C}\mathbf{L}\mathbf{x}_{\text{ball}} + \mathbf{x}_{\text{center}} \quad (2)$$

其中, $\mathbf{L}$ 是变换矩阵,其作用是将单位圆内的采样点转换成椭圆坐标系下的采样点; $\mathbf{C}$ 是旋转矩阵, $\mathbf{x}_{\text{center}}$ 是平移矩阵,也是椭圆的中心坐标矩阵,两个矩阵的作用是将椭圆坐标系下的采样点转换成世界坐标系下椭圆内采样点的坐标^[20]。

1.2 BIT*算法

初始化定义 $\mathbf{x} \in \mathbf{X}_{\text{sample}}$ 为在自由空间中的点。定义起始点和目标点 $\mathbf{x}_{\text{start}} \cup \mathbf{x}_{\text{goal}} \in \mathbf{X}_{\text{sample}}$ 。定义树 $T = (\mathbf{V}, \mathbf{E})$,其中 $\mathbf{V} \in \mathbf{X}_{\text{sample}}$ 为树中的树节点, $\mathbf{E} = (\mathbf{v}_1, \mathbf{v}_2)$ 为每两个树节点的连线,即树边。 Q_v 用于存储当前待扩展的树中节点, Q_e 用于存储当前连接信息。

BIT*算法的伪代码如算法1所示。

算法1: BIT* ($\mathbf{x}_{\text{start}}, \mathbf{x}_{\text{goal}}$)
1 $\mathbf{V} \leftarrow \mathbf{x}_{\text{start}}; \mathbf{E} \leftarrow \phi; \mathbf{X}_{\text{samples}} \leftarrow \mathbf{x}_{\text{goal}};$
2 $Q_e \leftarrow \phi; Q_v \leftarrow \phi; r \leftarrow \infty;$
3 **REPEAT**
4 **IF** $Q_e \equiv \phi$ **AND** $Q_v \equiv \phi$ **THEN**
5 $\text{Prune}(g_T(\mathbf{x}_{\text{goal}}));$
6 $\mathbf{X}_{\text{samples}} \xleftarrow{+} \text{Sample}(m, g_T(\mathbf{x}_{\text{goal}}));$
7 $\mathbf{V}_{\text{old}} \leftarrow \mathbf{V};$
8 $Q_v \leftarrow \mathbf{V};$
9 $r \leftarrow \text{radius}(|\mathbf{V}| + |\mathbf{X}_{\text{samples}}|);$
10 **WHILE** $\text{BestQV}(Q_v) \leq \text{BestQE}(Q_e)$ **DO**
11 $\text{ExpandVertex}(\text{BestInQueue}(Q_v));$
12 $(\mathbf{v}_m, \mathbf{x}_m) \leftarrow \text{BestInQueue}(Q_e);$
13 $Q_e \xleftarrow{-} \{(\mathbf{v}_m, \mathbf{x}_m)\};$
14 **IF** $g_T(\mathbf{v}_m) + c(\mathbf{v}_m, \mathbf{x}_m) + h(\mathbf{x}_m) < g_T(\mathbf{x}_{\text{goal}})$ **THEN**
15 **IF** $g_T(\mathbf{v}_m) + c(\mathbf{v}_m, \mathbf{x}_m) < g_T(\mathbf{x}_m)$ **THEN**
16 **IF** $\mathbf{x}_m \in \mathbf{V}$ **THEN**
17 $\mathbf{E} \xleftarrow{-} \{(\mathbf{v}, \mathbf{x}_m) \in \mathbf{E}\};$
18 **ELSE**
19 $\mathbf{X}_{\text{samples}} \xleftarrow{-} \{\mathbf{x}_m\};$
20 $\mathbf{V} \xleftarrow{+} \{\mathbf{x}_m\}; Q_v \xleftarrow{+} \{\mathbf{x}_m\};$
21 **END**
22 $\mathbf{E} \xleftarrow{+} \{(\mathbf{v}_m, \mathbf{x}_m)\};$

```

23   $Q_E \leftarrow \{(v, x_m) \in Q_E\};$ 
24  END
25  ELSE
26   $Q_E \leftarrow \phi; Q_V \leftarrow \phi;$ 
27  END
28  UNTIL STOP;
29  RETURN T

```

其中, $\text{Prune}(g_T(x_{\text{goal}}))$ 是修剪模块, 用来去除无用的点和边; $\text{Sample}(m, g_T(x_{\text{goal}}))$ 是批采样功能函数, 在未找到可行路径状态下, 在全地图自由空间中进行批量采样, 在通过搜索树成功找到路径后, 则缩小采样范围, 转变为在椭圆区域的自由空间中进行采样; $\text{BestQV}(Q_v)$ 和 $\text{BestQE}(Q_E)$ 分别返回 Q_v 、 Q_E 队列中最小代价值及其对应的顶点和边, 只要满足循环条件, 则持续对顶点队列中代价最小的点进行扩展 $\text{ExpandVertex}(\text{BestInQueue}(Q_v))$, 不满足则提取边队列中的最优边 (v_m, x_m) , 并对这条边进行比较判断, 若有利于路径则对其进行添加和修剪, 若不利则返回。若已经没有点能够对当前路径进行优化, 则将 Q_v 、 Q_E 清空, 重新采样。

2 改进的BIT*算法

2.1 区域批量采样策略

BIT*算法对采样数量具有极高的依赖性, 当初始样本采用小批量采样时, 由于全局采样导致的随机性较大, 采样点分布散乱, 在搜索范围 r 中找不到样本点的可能性较大, 路径生成的成功率较低, 如图2所示。当初始样本采用大批量采样时, 由于采用全局采样, 故所需采样面积较大, 所需样本数量巨大, 并且大部分的采样点都难以对生成路径具有正面作用, 既消耗了内存又造成计算时间损失, 如图3所示。为了解决该问题, 本文提出了区域批量采样, 该方法能够更集中地针对与起始点和目标点相关的区域进行搜索, 利用较少的样本点得到更高的样本密度, 和更多高质量的样本点, 并减少无效采样点的产生, 显著减少在无关或不可达区域内的无效探索, 提高路径生成的成功率和路径质量。

图2、图3中, 左下角为(0,0), 右上角为(50,30), 黑色

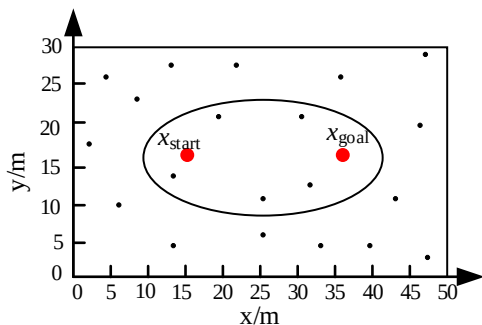


Fig. 2 Small sample sampling

图2 小样本采样

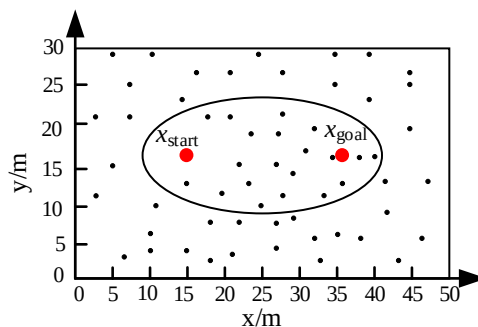


Fig. 3 Large sample sampling

图3 大样本采样

点为样本点, 图2的样本点数量为20, 图3样本点数量为60, x_{start} 为起始点, x_{goal} 为目标点, 椭圆为本文的限制区域。根据 Informed-RRT* 算法中椭圆的定义, 在整个采样空间中构建椭圆区域, 采样示意图如图4所示, 图中左下角为(6,6), 右上角为(45,30), 样本点数量为40。

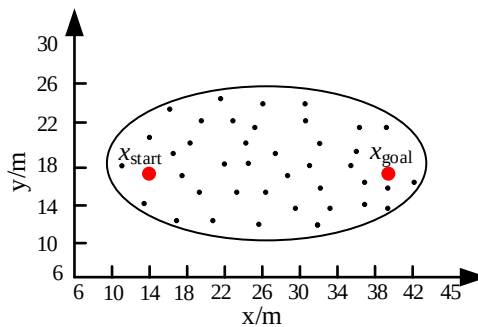


Fig. 4 Schematic of batch sampling of the region

图4 区域批量采样示意图

由此可知, 有椭圆区域限制的采样相比小样本更能搜索范围 r 内找到样本点, 相比大样本区域限制能够以更少的样本得到更高的样本密度, 并且在该区域内的样本点质量更高。

在空间中无法直接得到椭圆内的采样点, 根据式(2)与已知条件可得式(3), 该式可将单位圆内的采样点转化成世界坐标系下的椭圆坐标。

$$\mathbf{x}_{\text{ellipse}} = \begin{bmatrix} a * x_u \cos(\theta) - b * y_u \sin(\theta) + c_x \\ a * x_u \sin(\theta) + b * y_u \cos(\theta) + c_y \end{bmatrix} \quad (3)$$

其中, $\mathbf{x}_{\text{ball}} = (x_u, y_u)$ 是单位圆内的采样点坐标; a 与 b 分别为椭圆区域的半长轴与半短轴, 这里 $a = p * C_{\min}$, $b = \sqrt{4a^2 - C_{\min}^2}$, p 根据实际情况调节; $\mathbf{x}_{\text{center}} = (c_x, c_y)$ 是椭圆中心点坐标; θ 为起始点指向目标点向量夹角。

2.2 基于动态概率的搜索策略

在真实状态空间中, 障碍物的分布位置与密集程度各异, 这对搜索树的生长有较大影响。因此, 本文提出了动态概率策略。该策略能够根据障碍物的密集程度自动调节概率因子, 在不同概率区间内搜索树并采用相应的生长

方式,以提高搜索的适应性和效率。

为了减少算法的盲目性,并加快树的收敛,本文引入了目标导向策略,其伪代码如算法2所示。当满足条件后,节点 v 向终点生长固定步长,通过碰撞检测后,将边和点分别放入 Q_E 与 Q_V 中。当终点与 v 的距离小于步长时,通过碰撞检测后,直接将其与终点相连。

算法2: ExpandGoal($v, \mathbf{x}_{\text{goal}}$)
 1 $step; \mathbf{M}; \mathbf{d} = \mathbf{x}_{\text{goal}} - \mathbf{v}; D \leftarrow \|\mathbf{d}\|;$
 2 **IF** $D \leq step$ **THEN**
 3 $\mathbf{E}_i \leftarrow \text{Edge}(\mathbf{v}, \mathbf{x}_{\text{goal}});$
 4 **IF** $\text{CollisionFree}(\mathbf{M}, \mathbf{E}_i)$ **THEN**
 5 $\mathbf{E} \leftarrow \mathbf{E} \cup (\mathbf{v}, \mathbf{x}_{\text{goal}}); \mathbf{V} \leftarrow \mathbf{V} \cup \mathbf{x}_{\text{goal}};$
 6 $g_T(\mathbf{x}_{\text{goal}}) = g_T(\mathbf{v}) + D;$
 7 $Q_E \leftarrow \phi; Q_V \leftarrow \phi;$
 8 **END**
 9 **ELSE**
 10 $\mathbf{x}_{\text{new}} = \mathbf{v} + \mathbf{d} * step / D$
 11 $\mathbf{E}_i \leftarrow \text{Edge}(\mathbf{v}, \mathbf{x}_{\text{new}});$
 12 **IF** $\text{CollisionFree}(\mathbf{M}, \mathbf{E}_i)$ **THEN**
 13 $\mathbf{E} \leftarrow \mathbf{E} \cup (\mathbf{v}, \mathbf{x}_{\text{new}}); \mathbf{V} \leftarrow \mathbf{V} \cup \mathbf{x}_{\text{new}};$
 14 $g_T(\mathbf{x}_{\text{goal}}) = g_T(\mathbf{v}) + D;$
 15 $Q_E \leftarrow (\mathbf{v}, \mathbf{x}_{\text{goal}}); Q_V \leftarrow \mathbf{x}_{\text{new}};$
 16 **END**
 17 **END**

综上可得完整的拓展策略,如式(4)所示。

$$\text{ExpandTree} = \begin{cases} \text{ExpandVertex}(\mathbf{v}), \text{rand} < q \\ \text{ExpandGoal}(\mathbf{v}, \mathbf{x}_{\text{goal}}), \text{rand} \geq q \end{cases} \quad (4)$$

其中, $\text{ExpandVertex}(\mathbf{v})$ 是启发式生长, $\text{ExpandGoal}(\mathbf{v}, \mathbf{x}_{\text{goal}})$ 是目标偏向生长, rand 是(0,1)之间的随机数, q 为概率因子。

本文采用区域批量采样,障碍物的密集程度仅需考虑区域内的障碍物,故将该区域命名为“障碍物检测区”。本文将障碍物检测区定义为以 $c_{\min}$ 为长轴、 $c_{\min\text{axis}}$ 为短轴、 $\mathbf{x}_{\text{center}}$ 为中心点的椭圆区域,椭圆旋转角度为 θ ,可得该区域的表达式如式(5)所示。

$$\frac{((x - c_x) \cos \theta + (y - c_y) \sin \theta)^2}{a^2} + \frac{((x - h) \sin \theta + (y - c_y) \cos \theta)^2}{b^2} = 1 \quad (5)$$

其中, $\mathbf{x}_{\text{center}} = (c_x, c_y)$ 是椭圆中心点坐标, a 为判断区域的半长轴, b 为判断区域的半短轴。当障碍物的坐标代入式(5)中小于1,那么说明障碍物在障碍物检测区内,反之则不在。影响概率因子因素不仅体现在障碍物的数量上,还体现在障碍物的面积上。障碍物的面积对概率因子的影响很大,因而需要计算障碍物的面积在判断区域的占比情况。考虑到障碍物的数量与面积,其计算如式(6)所示。

$$q = w_1 \frac{I}{I_{\max}} + w_2 \frac{S_{\text{obs}}}{S_{\text{jr}}} \quad (6)$$

其中: w_1 和 w_2 分别为数量和面积的影响权重,相加为

1; I 为障碍物的数量, $I_{\max}$ 为障碍物的最大数量; $S_{\text{obs}} = \sum_{i=0}^I S_i$, S_{obs} 为障碍物的面积之和, S_i 为各障碍物的面积; $S_{\text{jr}} = 0.75 ab\pi$, 为障碍物的最大面积。

2.3 路径优化与平滑处理

为提高基于搜索树生成路径的优化效果,本文算法实施了路径修剪。通过路径回溯,从起点出发逐级判断路径节点:若当前节点与后续节点相连且未发生碰撞,则将后续节点的父节点更改为当前节点;若发生碰撞,则用前一个节点替代当前节点,并继续此过程,直至到达目标点。尽管经过修剪,路径仍可能存在一定数量的拐点,优化后的路径需具备平滑性。因此,本文采用线性插值和平均滤波方法对路径节点进行平滑处理。鉴于修剪后数据点间距不均且节点分布疏散,首先使用线性插值在相邻节点之间插入新节点,以确保节点等距分布,从而提升路径的连续性和平滑性。

$$\begin{aligned} x_i &= x_{\text{node1}} + \frac{i}{\text{num} - 1} \times (x_{\text{node2}} - x_{\text{node1}}) \\ y_i &= y_{\text{node1}} + \frac{i}{\text{num} - 1} \times (y_{\text{node2}} - y_{\text{node1}}) \end{aligned} \quad (7)$$

其中, $i = [0, \text{num} - 1]$, num 为插入节点数量, $(x_{\text{node1}}, y_{\text{node1}})$ 和 $(x_{\text{node2}}, y_{\text{node2}})$ 是两个相邻节点。

线性插值生成的高密度节点可以更好地保留路径特性,在这样的条件下,平均值滤波可以更有效地识别和消除拐点,实现数据的平滑。将经过插值所得的节点序列设为 $\mathbf{p} = [p_0, p_1, \dots, p_{N-1}]$, 滤波窗口大小为 $n = \text{cnt} + 1$, 为了方便描述,将扩展后的序列定义为 $\mathbf{p}' = [p_0, p_1, \dots, p_{N+\text{cnt}+1}]$, 对于 $i = [0, N + \text{cnt} - 1]$, 如果 $i < N$, 则 $p' = p_i$, 如果 $i \geq N$, 则 $p' = p_{N-1}$ 。之后,对 $\mathbf{p}'$ 进行移动平均滤波,当 $i \leq \text{cnt}$ 时,窗口大小为 $i + 1$, 当 $i > \text{cnt}$ 时,窗口大小固定为 n 。平均值滤波公式如式(8)所示。

$$pt_i = \frac{1}{n_i} \sum_{k=0}^{n_i-1} p'_{i-k} \quad (8)$$

其中, $n_i = \min(i + 1, n)$, p'_{i-k} 为第 $i - k$ 个元素。

最终处理结果如图5所示,其中黑色线条是原始路径,红色线条是修剪后得到的新路径,蓝色线条是对拐点的平滑处理,绿色虚线是有碰撞的路线,黄色虚线是非最优路线。该方法不仅可以减少路径拐点,还能够减小路径长度以起到优化作用。

2.4 算法流程

由上述可得,本文算法的伪代码如算法3所示。 $\text{Factor}(\mathbf{x}_{\text{start}}, \mathbf{x}_{\text{goal}}, \mathbf{M})$ 是根据起始点、目标点和地图计算概

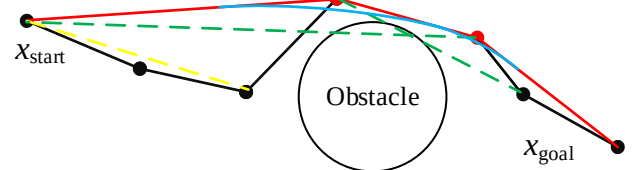


Fig. 5 Schematic diagram of path optimization process

图5 路径优化过程示意图

率因子; $ModifyPath()$ 的作用是通过回溯路径减少拐点并提取路径; $Sample(m, 2pC_{min})$ 的作用是在为找到路径时利用已知条件进行区域批处理采样; $Index$ 用来记录优化次数,在优化阶段, BIT*算法通过逐步缩小采样空间以改善路径,每次路径优化都要消耗大量的计算资源与时间,所以本文以优化次数为迭代限制条件,该方法能够节省大量无用迭代损失的时间; $LinearInterpolation()$ 和 $AveragerFilter()$ 分别是线性插值和平均值滤波,用于平滑路径; $ExpandGoal(v, x_{goal})$ 是向目标点生长的方法; Algorithm 1(12~20)为本文算法1的12~20行。

算法3: 改进BIT* (x_{start}, x_{goal})

```

1  $V \leftarrow x_{start}; E \leftarrow \phi; X_{samples} \leftarrow x_{goal}; M$ 
2  $Q_E \leftarrow \phi; Q_V \leftarrow \phi; r \leftarrow n; Index = 0;$ 
3  $q \leftarrow Factor(x_{start}, x_{goal}, M)$ 
4 REPEAT
5 IF  $Q_E \equiv \phi$  AND  $Q_V \equiv \phi$  THEN
6  $X_{samples} \leftarrow InEllipse(g_T(x_{goal}))$ ;
7  $ModifyPath()$ ;
8 IF  $g_T(x_{goal}) \neq \infty$ 
9  $X_{samples} \leftarrow Sample(m, g_T(x_{goal}))$ ;
10 ELSE
11  $X_{samples} \leftarrow Sample(m, 2pC_{min})$ ;
12 END
13  $Index++$ ;
14 IF  $Index = 3$ 
15  $LinearInterpolation()$ ;
16  $AveragerFilter()$ ;
17 BREAK;
18 END
19  $V_{old} \leftarrow V$ ;
20  $Q_V \leftarrow V$ ;
21 WHILE  $BestQV(Q_V) \leq BestQE(Q_E)$  DO
22 IF  $rand < q$ 
23  $ExpandVertex(BestInQueue(Q_V))$ ;
24 ELSE
25  $ExpandGoal(v, x_{goal})$ ;
26 END
27 Algorithm 1(12~20);
28 END
29 ELSE
30  $Q_E \leftarrow \phi; Q_V \leftarrow \phi$ ;
31 END
32 UNTIL STOP;
33 RETURN  $T$ 

```

3 仿真实验与分析

为了验证算法的有效性,进行仿真实验并分析。仿真计算机CPU是AMD Ryzen 5 3500U,主频为2.10 GHz,运行内存16 GB,使用PyCharm进行仿真。

实验中,将地图大小设置为50×30(单位:m),起点坐

标为(8,8),目标点坐标为(42,23)。本文算法中,将第一次批量采样点数量设置为160,后续批采样个数为100,搜索半径为6,步长为2,迭代次数为1200次。为了充分检验本文算法的有效性,将两种障碍物密度设置不同的环境,黑色矩形框为地图边界,灰色矩阵与圆形物体为已知的静态障碍物,白色空间为自由空间。

在1200次迭代下,在初始采样数量为150,优化采样为90的10次实验优化效果如图6所示。可以看出,本文算法在找到路径并优化两次后的效果会减弱,在这之后的迭代次数属于可舍迭代,故本文迭代限制条件为优化两次。

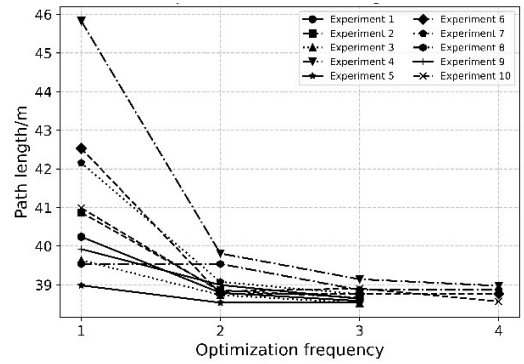


Fig. 6 Optimization effect diagram

图6 优化效果示意图

将本文改进算法与 Informed-RRT*算法、BIT*算法和文献[13]所提出的改进算法进行对比实验,将该算法记为 BITs*算法,并将不同算法的初始路径时间、最终路径时间、初始路径长度、最终路径长度、迭代次数和失败率进行对比分析。为了减小实验的偶然性,每组实验都需要运行20次并取平均值。图7和图8分别为算法在简单环境和复杂环境下的运行结果,左侧为算法初始路径结果图,右侧为算法最终路径结果图。其中,橙色虚线为椭圆采样范围边界,红色实线为算法规划出的路径。表1和表2分别为算法在简单环境和复杂环境下的算法性能。

算法在简单环境下的效果如图7和表1所示。初始阶段,本文算法在初始路径耗时方面相较于传统的 Informed-RRT*算法至少减少了74.51%以上,相比 BIT*算法和 BITs*算法分别减少了60.06%和58.06%以上;初始路径长度相较于 Informed-RRT*算法至少减少了23.90%以上,相比 BIT*算法和 BITs*分别减少了7.21%和10.33%以上。这验证了区域限制采样与动态概率选择策略在算法初始阶段有着显著优化作用。在优化阶段中,由于加入了迭代限制条件,故本文算法平均只需迭代470次便达到预期效果,时间大幅减少。实验表明,本文算法最终路径耗时相较于传统的 Informed-RRT*算法至少减少了75.98%以上,相比 BIT*算法和 BITs*算法分别减少了50.71%和49.27%以上;最终路径长度相较于 Informed-RRT*算法至少减少了20.23%以上,相比 BIT*算法和 BITs*算法至少减少了0.29%和0.23%以上。由于 BIT*算法本身的优越性,故最终路径

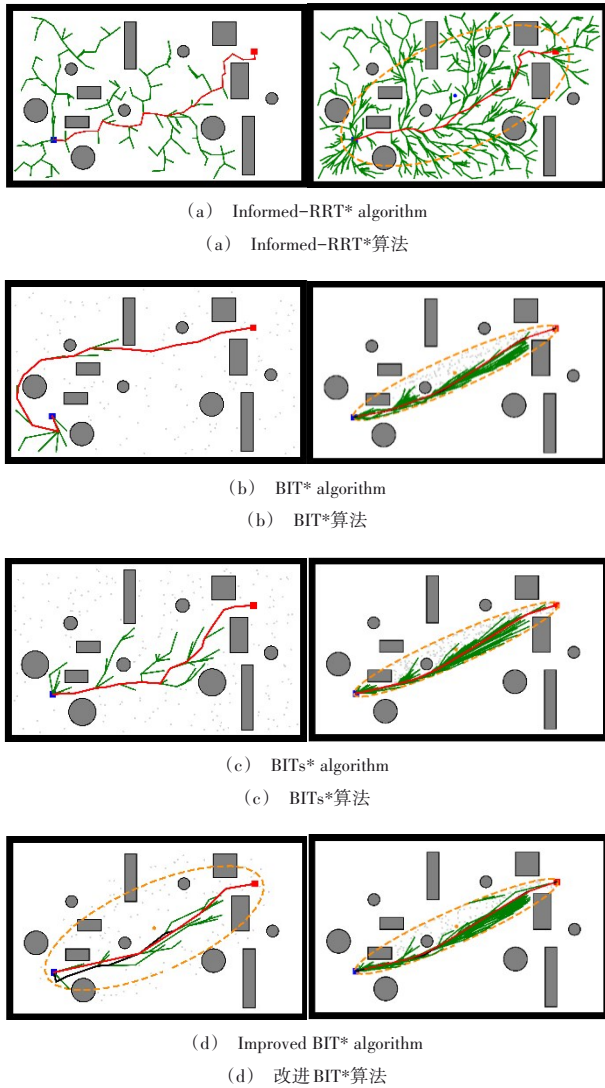


Fig. 7 Operation results of the algorithms in the simple environment

图7 简单环境下算法运行结果

Table 1 Performance comparison of the algorithms in the simple environment

表1 简单环境下算法性能比较

算法	初始耗时	初始长度	最终耗时	最终长度	失败率	迭代次数
Informed-RRT*	0.51	51.42	4.33	46.92	0	1 200
BIT*	0.33	43.64	2.11	37.73	30	1 200
BITs*	0.31	42.17	2.05	37.52	20	1 200
本文算法	0.13	39.13	1.04	37.43	0	470

优化较少,但是本文算法在时间上有了大幅度提升,而且减少了迭代次数,所消耗的计算资源与内存也会减少。在简单环境下,本文算法的失败率为0,而BIT*算法和BITs*算法的失败率分别为30%和20%,体现了本文算法的优越性。

算法在复杂环境下的实验结果如图8和表2所示。初始阶段,本文算法在初始路径耗时方面,相较于Informed-RRT*算法至少减少了62.5%以上,而相比于BIT算法和BITs*算法分别减少了61.7%和59.09%以上;初始路径长

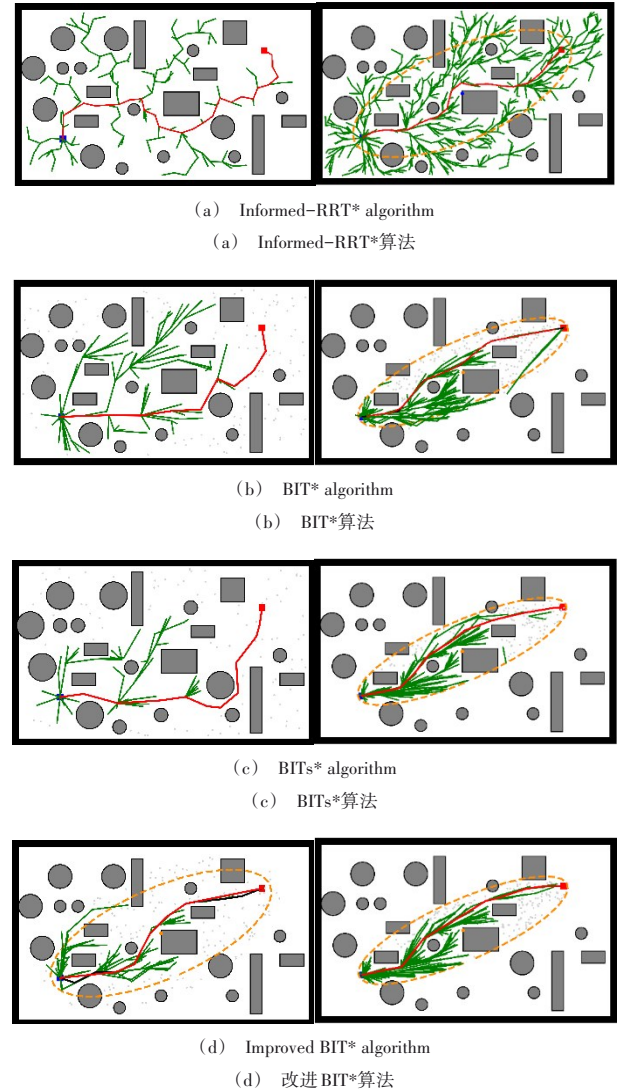


Fig. 8 Operation results of the algorithms in the complex environment

图8 复杂环境下算法运行结果

Table 2 Performance comparison of the algorithms in the complex environment

表2 复杂环境下算法性能比较

算法	初始耗时	初始长度	最终耗时	最终长度	失败率	迭代次数
Informed-RRT*	0.48	48.39	5.63	44.42	0	1 200
BIT*	0.47	46.38	2.44	38.48	20	1 200
BITs*	0.44	46.11	2.35	38.53	20	1 200
本文算法	0.18	40.09	1.25	38.14	0	610

度相比Informed-RRT*算法至少减少了17.15%以上,相比BIT*算法和BITs*算法减少了13.56%和13.06%以上。实验数据表明,区域限制采样与动态概率选择策略在复杂环境下的初始阶段仍具有显著优化效果。在路径优化阶段,本文算法平均只需进行610次迭代。实验结果表明,本文算法在最终路径耗时上,相较于Informed-RRT*算法至少减少了77.79%以上,相比于BIT算法和BITs*算法减少了48.77%和46.81%以上;最终路径长度相较于Informed-RRT算法至少减少了14.14%以上,相比BIT*算法和BITs*

算法分别减少了0.88%和1.01%。各项性能指标在复杂环境下的表现与在简单地图下的结果基本一致。在复杂环境下,本文算法的失败率为0,而BIT*算法和BITs*算法的失败率均为20%,进一步验证了本文算法的优越性。

4 结语

针对BIT*算法在路径规划中存在的问题,本文提出了改进BIT*算法,并进行了对比实验分析。与BIT*算法相比,本文算法找到初始路径的时间减少了60.6%~61.7%,初始路径减少了10.33%~13.56%,最终路径减少了0.29%~0.88%,运行时间减少了48.77%~50.71%,算法成功率提升明显。实验表明,经过改进的算法在路径搜索效率和收敛速度方面相较于原有算法有显著提升。

参考文献:

- [1] GUO X W, FU S M, WEI Y Y, et al. SQP-GPMP2 algorithm based path planning of mobile robots [J]. Electronics Optics and Control, 2024, 31 (9): 104-110.
郭希文,付世沫,魏媛媛,等. 基于SQP-GPMP2算法的移动机器人路径规划[J]. 电光与控制, 2024, 31(9): 104-110.
- [2] LYU Y Z, MAO J H, DU X P, et al. Signal laying path planning method for the civil aircraft based on improved Dijkstra algorithm [J]. Mechanical design and manufacturing engineering, 2024, 53(9): 78-82.
吕逸舟,毛建华,杜晓鹏,等. 基于改进Dijkstra算法的民机信号敷设路径规划方法[J]. 机械设计与制造工程, 2024, 53(9): 78-82.
- [3] WANG Z, YANG Y, ZHOU S, et al. Dynamic path planning algorithm of unmanned ship based on A*-dynamic window approach [J]. Journal of Naval University of Engineering, 2024, 36(2): 13-18.
王征,杨洋,周帅,等. 基于A*-动态窗口法的无人船动态路径规划算法[J]. 海军工程大学学报, 2024, 36(2): 13-18.
- [4] GU Z L, LIU Y, YUE G, et al. Fast path planning based on improved RRT algorithm [J]. Journal of Ordnance Equipment Engineering, 2022, 43(10): 294-299.
顾子吕,刘宇,岳广,等. 基于改进RRT算法的快速路径规划[J]. 兵器装备工程学报, 2022, 43(10): 294-299.
- [5] PEI Y J, YANG C J, YANG L L. Path planning algorithm for mobile robot based on improved RRT* [J]. Computer Engineering, 2019, 45(5): 285-290, 297.
裴以建,杨超杰,杨亮亮. 基于改进RRT*的移动机器人路径规划算法[J]. 计算机工程, 2019, 45(5): 285-290, 297.
- [6] TANG L J, FAN Y X, ZHOU X Y, et al. AUV path planning method based on improved sparrow search algorithm [J]. Chinese Journal of Ship Research, 2024, 20(3): 275-287.
唐李军,范云霞,周星宇,等. 基于改进麻雀搜索算法的AUV路径规划方法[J]. 中国舰船研究, 2024, 20(3): 275-287.
- [7] JIANG B H. Improvement of robot global path planning algorithm based on ant colony algorithm [J]. New Technologies and New Products of China, 2024(17): 7-10.
姜炳辉. 基于蚁群算法的机器人全局路径规划算法改进[J]. 中国新技术新产品, 2024(17): 7-10.
- [8] LUO J Y, SUN B Y. Path planning of mobile robot based on improved RRT* algorithm [J]. Computer Engineering and Design, 2024, 45(8): 2357-2363.
罗济雨,孙丙宇. 基于改进型RRT*算法的移动机器人路径规划[J]. 计算机工程与设计, 2024, 45(8): 2357-2363.
- [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]//IEEE International Conference on Intelligent Robots and Systems, 2014: 2997-3004.
- [10] LIU W Q, SHAN L, ZHANG W L, et al. Unmanned aerial vehicle path planning algorithm based on improved informed RRT* in complex environment [J]. Journal of Shanghai Jiaotong University, 2024, 58(4): 511-524.
刘文倩,单梁,张伟龙,等. 复杂环境下基于改进Informed RRT的无人机路径规划算法[J]. 上海交通大学学报, 2024, 58(4): 511-524.
- [11] 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]//IEEE International Conference on Robotics and Automation, 2015: 3067-3074.
- [12] LIU C, LIU Y, SONG L B, et al. Greedy BIT* (GBIT*): greedy search policy for sampling-based optimal planning with a faster initial solution and convergence [C]//2021 International Conference on Computer, Control and Robotics (ICCCR), 2021: 30-36.
- [13] LU Y H, LIU Y T, YU H Q, et al. Research on path optimization of agricultural robot based on improved BIT algorithm [J]. Instrumentation Customer, 2022, 29(11): 1-6.
陆宇豪,刘义亭,郝汉琪,等. 基于改进BIT算法的农业机器人路径优化研究[J]. 仪器仪表用户, 2022, 29(11): 1-6.
- [14] WANG X J, CUI X J, LI X H. Robot path planning with improved BIT* algorithm in dynamic environment [J]. Computer Engineering and Applications, 2025, 61(7): 361-369.
王晓军,崔锡杰,李晓航. 动态环境下改进BIT*算法的机器人路径规划[J]. 计算机工程与应用, 2025, 61(7): 361-369.
- [15] SHENG C H, FAN J M. Improved APF-Informed-RRT* fusion algorithm for UAV trajectory planning [J]. Electronics Optics and Control, 2023, 30(6): 1-7.
盛春红,范珈铭. 改进APF-Informed-RRT*融合算法的无人机航迹规划[J]. 电光与控制, 2023, 30(6): 1-7.
- [16] GAMMELL J D, BARFOOT T D, SRINIVASA S. Batch informed trees (BIT*): informed asymptotically optimal anytime search [J]. The International Journal of Robotics Research, 2020, 39(5): 543-567.
- [17] ZHANG B Q, LIU J D. Research on robot path planning based on improved BIT* [J]. Application Research of Computers, 2022, 39(1): 59-63.
张伯泉,刘嘉栋. 基于改进BIT*算法的机器人路径规划研究[J]. 计算机应用研究, 2022, 39(1): 59-63.
- [18] ZOU Y X. Research on real-time obstacle avoidance algorithm of harvesting robot manipulator based on joint space [D]. Changsha: Central South University of Forestry and Technology, 2018.
邹宇星. 基于关节空间的采摘机器人机械臂实时避障算法研究[D]. 长沙:中南林业科技大学, 2018.
- [19] LI M, ZHAO J L, YUAN Y P, et al. Obstacle avoidance path planning of manipulator based on RRT-A* fusion algorithm [J]. Machinery Design & Manufacture, 2023(3): 175-178, 182.
李明,赵金龙,袁逸萍,等. 基于RRT-A*融合算法的机械臂避障路径规划[J]. 机械设计与制造, 2023(3): 175-178, 182.
- [20] GE Q X, ZHANG W, CHEN H, et al. Informed-RRT* algorithm based on heuristic adaptive step size optimization [J]. Radio Engineering, 2024, 54(10): 2446-2452.
葛启兴,章伟,陈浩,等. 启发式自适应步长优化Informed-RRT算法[J]. 无线电工程, 2024, 54(10): 2446-2452.

(责任编辑:孙娟)