DOI：10.13973/j.cnki.robot.2015.0443

复杂环境下基于 RRT 的智能车辆运动规划算法

杜明博[1,2]，梅 涛[2]，陈佳佳[2]，赵 盼[2]，梁华为[2]，黄如林[1,2]，陶 翔[2]
（1. 中国科学技术大学自动化系，安徽 合肥 230027； 2. 中国科学院合肥物质科学研究院应用技术研究所，安徽 合肥 230027）

摘 要：在存在大量无规则障碍物且障碍物分布不均匀的复杂环境下，现有规划算法不能很好地解决智能车辆的运动规划问题．为此，本文提出了一种简单实用的基于 RRT（快速搜索随机树）的运动规划算法——连续曲率 RRT 算法．该算法在 RRT 框架中结合了环境约束以及车辆自身的约束．它首先引入了目标偏向采样策略以及合理的度量函数，大大地提高了规划速度和质量；接着提出了一种基于最大曲率约束的后处理方法以生成平滑的且曲率连续的可执行轨迹．通过仿真实验和实车测试，证实了该算法的正确性、有效性和实用性．

关键词：运动规划；智能车辆；快速搜索随机树；曲率约束
中图分类号：TP242 **文献标识码**：A **文章编号**：1002-0446(2015)-04-0443-08

RRT-based Motion Planning Algorithm for Intelligent Vehicle in Complex Environments

DU Mingbo[1,2]，MEI Tao[2]，CHEN Jiajia[2]，ZHAO Pan[2]，LIANG Huawei[2]，HUANG Rulin[1,2]，TAO Xiang[2]
(1. *Department of Automation*, *University of Science and Technology of China*, *Hefei* 230027, *China*;
2. *Institute of Applied Technology*, *Hefei Institutes of Physical Science*, *Chinese Academy of Sciences*, *Hefei* 230027, *China*)

Abstract: The existing planning algorithms can not properly solve the motion planning problem of intelligent vehicle in complex environments with many irregular and random obstacles. To solve the problem, a simple and practical RRT-based algorithm, continuous-curvature RRT algorithm, is proposed. This algorithm combines the environmental constraints and the constraints of intelligent vehicle with RRTs. Firstly, a goal-biased sampling strategy and a reasonable metric function are introduced to greatly increase the planning speed and quality. And then, a post-processing method based on the maximum curvature constraint is presented to generate a smooth, continuous-curvature and executable trajectory. Simulation experiments and real intelligent vehicle test verify the correctness, validity and practicability of this algorithm.

Keywords: motion planning; intelligent vehicle; RRT (rapidly-exploring random tree); curvature constraint

1 引言（Introduction）

智能车辆（intelligent vehicle），作为一种智能轮式移动机器人，目前吸引了国内外众多的研究机构和专家学者的关注，并且也在军事、交通运输等领域得到了实际应用．在智能车辆的相关技术研究中，运动规划是其最为重要的核心技术之一．所谓运动规划是指移动机器人在位姿空间中找到一条从初始位姿点到目标位姿点的连续无碰撞路径，同时这条路径还要满足环境约束、时间约束以及机器人本身的动力学约束等约束条件 [1].

过去的 50 多年，运动规划技术发展迅速，涌现出了许多规划算法．传统的路径规划算法如遗传算法、模拟退火算法、蚁群优化算法等，这些算法在解决一般的规划问题时有其优越性，但由于它们需要在一个确定性空间中对障碍物进行确定的建模，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的求解 [2]．另外，前向图搜索算法，如 A*、D* 和人工势场法等，在规划时虽然能满足最优性和实时性的要求，但是其并未考虑车辆的非完整性约束的限制，使得规划的路径不一定可执行 [3].

LaValle 提出的快速搜索随机树（rapidly-exploring random tree，RRT）[18] 算法，由于其采用随机采样的规划方法，不需要对状态空间进行预处理，搜索速度快，而且在搜索的过程中还考虑了机器人客观存在的约束（非完整性约束、动力学约束、运动学约束），从而能够有效地解决复杂环境

基金项目：国家自然科学基金重大研究计划重点项目（91120307）；国家自然科学基金重大研究计划集成项目（91320301）；国家自然科学基金青年基金资助项目（61304100）．
通信作者：梅涛，tmei@iim.ac.cn 收稿/录用/修回：2015-02-10/2015-04-27/2015-04-28

下的运动规划问题，使得该算法近年来在机器人运动规划领域得到了广泛的应用和研究 [1]．但是其本身也存在一些缺陷：(1) 全局的均匀随机采样策略，导致算法无谓地耗费较大代价，使得收敛速度慢 [2]；(2) 度量函数（最近邻算法）在解决复杂约束的机器人规划问题时可能是制约算法有效性的一个瓶颈 [4]；(3) 算法的随机性会导致生成的路径不平滑，无法被非完整性约束机器人直接执行．

针对基本 RRT 算法的上述不足，国内外学者也对该算法进行了不断地改进，以适应不同的应用环境．为了提高搜索效率，Kuffner 和 LaValle 提出了 Bi-RRT（双向搜索树）[13]，从初始状态和目标状态并行生成两棵树，加快算法收敛；随后他们又提出了 RRT-connect [14]，以提高节点扩展效率；为了使 RRT 算法能有效地适用于具有狭窄通道的复杂环境，基于障碍物边界的 RRT 算法 [12] 以及偏向狭窄空间的 RRT 算法 [10-11] 相继被提出．在 RRT 运动规划中，度量函数非常重要但同时又难以准确定义，Cheng [17] 提出了在 RRT 搜索过程中让度量函数不断地学习以降低对于环境的敏感度．对于 RRT 算法的随机性所造成的路径不平滑问题，文 [5] 引入了一种 Dubins 路径，但是由于是直线和圆弧拼接产生的路径，因此路径曲率不连续；Fraichard 和 Scheuer [6] 提出了用回旋曲线（clothoid）来作平滑处理，但是回旋曲线没有闭合形式解，不能够实时准确地得到；Lau 等人 [15] 采用了 5 次贝塞尔曲线，但是没有考虑路径曲率的连续性和机器人的自身约束；Elbanhawi 等人 [7-8] 提出了简单有效的 3 次 B 样条平滑算法，在保证曲率连续的同时又满足机器人的非完整性约束．

本文针对存在大量无规则障碍物且障碍物分布不均匀的复杂环境下的智能车辆运动规划问题，提出一种连续曲率 RRT（continuous-curvature RRT）运动规划算法，下文中简称为 CC-RRT．它采用了目标偏向采样策略和基于合理度量函数的节点连接机制以及满足车辆运动约束的后处理方法，来解决现有 RRT 算法在这方面的不足．通过仿真实验以及复杂停车场环境下的实车实验，证实了该算法的优越性、有效性和实用性．

2 非完整性约束车辆模型（Nonholonomic constraint vehicle model）

非完整性约束是指含有系统广义坐标导数且不可积分的约束 [4]．智能车辆是一个典型的非完整性约束系统，其简化运动模型如图 1 所示．在状态空间中车辆状态 $\boldsymbol{X}=(x,y,\theta)$，其中 (x,y) 是车辆在系统坐标系下的位置，θ 是车辆纵轴与坐标系之间的夹角，φ 为前轮转角且 $|\varphi|\leqslant\varphi_{\max}$．由于存在非完整性约束，车轮只能在平面内做纯滚动、无滑动的运动．在任何运动瞬间，车体的速度必然指向机器人的主轴 [4]，即满足：

$$\begin{cases}\mathrm{d}x\cdot\sin\theta-\mathrm{d}y\cdot\cos\theta=0\\ K_{\max}=\dfrac{\tan\varphi_{\max}}{L}=\dfrac{1}{R_{\min}}\end{cases}\tag{1}$$

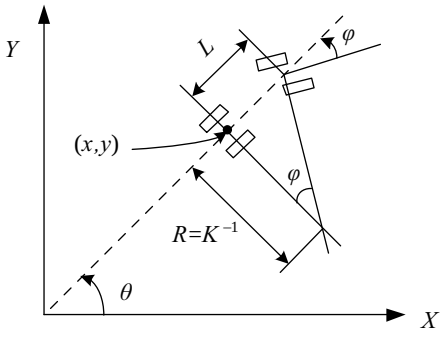


图 1 简化的车辆运动模型

Fig.1 A simplified vehicle kinematic model

根据非完整性约束的定义及车辆模型的分析可以发现，传统的运动规划算法并不适用于智能车辆的运动规划问题，它们不考虑车辆约束，所得到的路径不能应用于实际环境中．

3 连续曲率 RRT 算法（Continuous-curvature RRT algorithm）

3.1 基本 RRT 算法

RRT 算法是一种随机采样的单查询树状结构的搜索算法．它以状态空间 C 中的起始状态 $\boldsymbol{q}_{\text{start}}$ 为根节点构建随机搜索树 T，通过迭代地随机采样方式选择状态节点 $\boldsymbol{q}_{\text{rand}}$，遍历 T 找出距离 $\boldsymbol{q}_{\text{rand}}$ 最近的节点 $\boldsymbol{q}_{\text{near}}$，然后通过控制输入集 U 选择输入 $\boldsymbol{u}$，驱动机器人沿着 $\boldsymbol{q}_{\text{near}}$ 到 $\boldsymbol{q}_{\text{rand}}$．根据状态转换方程在一定时间 Δt 上进行积分便可得到 $\boldsymbol{q}_{\text{new}}$，并将其作为新的叶节点添加到随机扩展树上．重复上述过程，直到目标状态 $\boldsymbol{q}_{\text{goal}}$ 也成为叶节点或者超过最大迭代次数时才结束搜索．然后由目标节点 $\boldsymbol{q}_{\text{goal}}$ 反向追溯到根节点 $\boldsymbol{q}_{\text{start}}$，就可以得到规划路径．其搜索过程如 Basic RRT Algorithm 所示．

3.2 CC-RRT 算法

通过上述对车辆运动规划问题的分析以及基本 RRT 算法的介绍，本文基于 RRT 框架，提出了一套适用于智能车的运动规划算法，具体算法步骤如 CC-RRT Algorithm 所示．

Basic RRT Algorithm

1. T.init($\boldsymbol{q}_{\text{start}}$);
2. **for** k =1 to K do
3. $\boldsymbol{q}_{\text{rand}} \leftarrow$ RANDOM_STATE();
4. $\boldsymbol{q}_{\text{near}} \leftarrow$ NEAREST_NEIGHBOR($\boldsymbol{q}_{\text{rand}}$, T);
5. $\boldsymbol{u}_{\text{best}}$, $\boldsymbol{q}_{\text{new}}$, *success*$\leftarrow$ CONTROL($\boldsymbol{q}_{\text{near}}$, $\boldsymbol{q}_{\text{rand}}$, T);
6. **if** success
7. T.Add_Node($\boldsymbol{q}_{\text{new}}$);
8. T.Add_Edge($\boldsymbol{q}_{\text{near}}$, $\boldsymbol{q}_{\text{new}}$, $\boldsymbol{u}_{\text{best}}$);
9. **end if**
10. **end for**
11. **Return** T

CC-RRT Algorithm

1. T.init($\boldsymbol{q}_{\text{start}}$);
2. **Repeat**
3. $p \leftarrow$ rand(0,1);
4. **if** $(p > p_{\text{goal}})$
5. $\boldsymbol{q}_{\text{rand}} \leftarrow$ RANDOM_STATE();
6. **else**
7. $\boldsymbol{q}_{\text{rand}} = \boldsymbol{q}_{\text{goal}}$
8. **end if**
9. $\boldsymbol{q}_{\text{near}} \leftarrow$ Effective_Nearest($\boldsymbol{q}_{\text{rand}}, T$)
10. $\boldsymbol{u}_{\text{best}}$, $\boldsymbol{q}_{\text{new}}$, *success* $\leftarrow$ CONTROL($\boldsymbol{q}_{\text{near}}$, $\boldsymbol{q}_{\text{rand}}$, T)
11. **if** *success*
12. T.Add_Node($\boldsymbol{q}_{\text{new}}$)
13. T.Add_Edge($\boldsymbol{q}_{\text{near}}$, $\boldsymbol{q}_{\text{new}}$, $\boldsymbol{u}_{\text{best}}$)
14. **end if**
15. **Until** find a collision-free path from $\boldsymbol{q}_{\text{start}}$ to $\boldsymbol{q}_{\text{goal}}$
16. **Return** $S \leftarrow$ Post_Processing(T)

Function Effective_Nearest($\boldsymbol{q}_{\text{rand}}, T$)

17. $C_{\max} \leftarrow -\infty$
18. **for** all $\boldsymbol{q}_i$ in T
19. $C \leftarrow C(\boldsymbol{q}_i, \boldsymbol{q}_{\text{rand}}) = w_1 * \text{dis}(\boldsymbol{q}_i, \boldsymbol{q}_{\text{rand}}) + w_2 * \text{head}(\boldsymbol{q}_i, \boldsymbol{q}_{\text{rand}})$
20. **if** $C > \boldsymbol{C}_{\max}$
21. $C_{\max} \leftarrow C$; $\boldsymbol{q}_{\text{near}} \leftarrow \boldsymbol{q}_i$;
22. **end if**
23. **end for**
24. **Return** $\boldsymbol{q}_{\text{near}}$

3.3　目标偏向采样策略

根据基本 RRT 算法流程的描述可知，其采用的是对整个状态空间进行均匀随机搜索的方式．这种方式虽然便于向未知的空间进行搜索，但是其在许多不必要的区域进行采样，浪费了大量的计算时间和成本，大大降低了算法收敛速度，无法满足智能车运动规划的实时性需求．

为了克服这一缺陷，本文采用了一种目标偏向采样策略 [4,14]，就是提前设定一个目标偏向概率阈值 p_{goal} （一般不超过 10%），在进行随机采样时根据均匀概率分布随机获得一个概率值 p，如果 $p > p_{\text{goal}}$，则 $\boldsymbol{q}_{\text{rand}}$ 根据函数 RANDOM_STATE() 获得；否则 $\boldsymbol{q}_{\text{rand}} = \boldsymbol{q}_{\text{goal}}$．这种方式既保持了算法的随机特性同时又加快了向目标状态收敛的速度 [7]．具体算法步骤见 CC-RRT Algorithm 中的第 3 ～ 8 行．

3.4　合理的度量函数

度量函数在 RRT 算法中是用来寻找“距离”随机采样节点 $\boldsymbol{q}_{\text{rand}}$ “最近”的树节点的函数．它往往决定着节点选取的合理性以及算法的效率，是制约 RRT 算法实际应用的一个主要环节．基本 RRT 算法中，采用的是欧氏距离作为度量函数．但是由于是应用于智能车辆，所以在选择最近邻节点时应该考虑节点间的夹角关系，即新增节点对于整体路径曲折程度的影响．鉴于车辆的转向能力，节点的选择应该更倾向于平滑而非欧氏距离最短．如图 2 所示，在选择节点时，虽然 $\boldsymbol{q}_j$ 与 $\boldsymbol{q}_{\text{new}}$ 之间欧氏距离更近，但是由于 $\theta_1 > \theta_2$，所以选择 $\boldsymbol{q}_i$ 作为最近邻节点，所得到的路径更为平缓．因此，本文提出了一个更合理也更符合智能车辆实际应用的度量函数 $C(\boldsymbol{q}_i, \boldsymbol{q}_j)$，即

$$\begin{cases} C(\boldsymbol{q}_i, \boldsymbol{q}_j) = w_1 \cdot D(\boldsymbol{q}_i, \boldsymbol{q}_j) + w_2 \cdot H(\boldsymbol{q}_i, \boldsymbol{q}_j) \\ D(\boldsymbol{q}_i, \boldsymbol{q}_j) = N_1(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}) \\ H(\boldsymbol{q}_i, \boldsymbol{q}_j) = N_2(|\theta_i - \theta_j|) \end{cases} \tag{2}$$

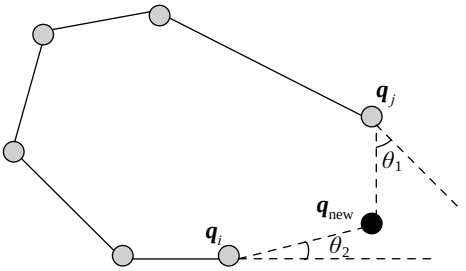


图 2　度量函数的节点选择示意图

Fig.2　The node selection of metric function

由于距离和角度是两个不同的量纲，为了能把这两个不同来源的数据统一到同一个参考系下，需要对这两个变量进行归一化处理．在这里本文采用了简单有效的离差归一化法对这两个变量进行归一化处理．式 (2) 中的 $D(\boldsymbol{q}_i, \boldsymbol{q}_j)$ 表示两节点间的欧氏距离经过归一化处理后的值，$N_1(d)$ 是距离对应的归一化函数；$H(\boldsymbol{q}_i, \boldsymbol{q}_j)$ 表示两节点夹角经过归一化处理后的值，$N_2(\theta)$ 是角度对应的归一化函数．离

差归一化函数如式 (3) 所示. 参数 w_1 和 w_2 是对应的权值，可以根据实际应用进行设定（本文根据实验的经验数据设定 $w_1 = w_2 = 0.5$）.

$$\begin{cases} N_1(d) = \dfrac{d_{\max} - d}{d_{\max}} \\ N_2(\theta) = \dfrac{\theta_{\max} - \theta}{\theta_{\max}} \end{cases} \tag{3}$$

参照 5.1 节中仿真实验，其中图 5(a) 和 5(c) 分别对应着基本 RRT 与 CC-RRT 的生成路径结果（红色折线），对比两者可以很容易发现：采用本文的合理度量函数所得到的路径，其整体的平滑度有了很大提高.

Post-Processing Method

1. $Q \leftarrow$ Pruning(T)
2. $Q \leftarrow$ Insert_MidNode(Q)
3. $S \leftarrow$ Cubic_Bspline(Q)
4. **Return** S

Function Pruning(T)

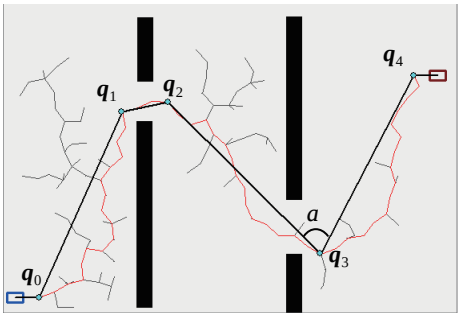
5. $T \leftarrow$ obtained from CC-RRT
6. Var Q_1, Q_2: Path
7. $Q_1(\boldsymbol{q}_0, \boldsymbol{q}_1, \boldsymbol{q}_2, \cdots, \boldsymbol{q}_n) =$ Path(T)
8. $\boldsymbol{q}_{\text{temp}} \leftarrow \boldsymbol{q}_0$; Q_2.Add_Node($\boldsymbol{q}_0$)
9. **while** $\boldsymbol{q}_{\text{temp}} != \boldsymbol{q}_n$ do
10. **for** each node $\boldsymbol{q}_i \in Q_1$
11. **if** Collision($\boldsymbol{q}_{\text{temp}}, \boldsymbol{q}_i$)
12. $\boldsymbol{q}_{\text{temp}} \leftarrow \boldsymbol{q}_i$;
13. Q_2.Add_Node($\boldsymbol{q}_{\text{temp}}$); **break**
14. **end if**
15. **end for**
16. Q_2.Add_Node($\boldsymbol{q}_n$)
17. **end while**
18. **for** each node $\boldsymbol{q}_k \in Q_2$
19. **if** Angle($\overrightarrow{\boldsymbol{q}_{k+1}\boldsymbol{q}_k}, \overrightarrow{\boldsymbol{q}_{k+1}\boldsymbol{q}_{k+2}}$) $< \alpha_{\min}$
20. Q_2.Insert_Node($\boldsymbol{q}_k, \boldsymbol{q}_{\text{insert}}, \boldsymbol{q}_{k+1}$)
21. **end if**
22. **end for**
23. **Return** Q_2

4 后处理方法（Post-processing method）

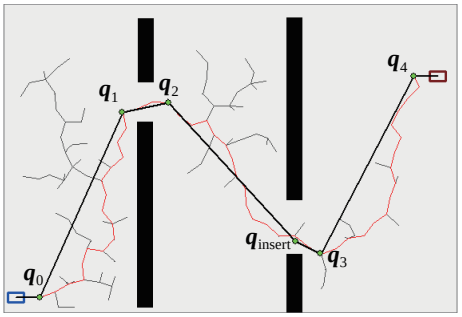
鉴于 RRT 算法随机采样的特性，它生成的路径常常是不自然的、抖动的，其中包含很多不需要的折点. 尤其是在复杂的环境下，障碍物约束使得生成的折点更多，这对于智能车辆来说很难跟踪，而且会对车辆造成严重的机械磨损. 因此，本文针对复杂环境下智能车辆的实际应用提出了高效的后处理方法，从而与 RRT 算法相结合生成曲率连续同时又满足车辆行驶约束的平滑路径. 具体的后处理方法伪代码如 Post-Processing Method 所示.

大体的操作流程是首先采用基于最大曲率约束的剪枝函数 Pruning(T) 对整棵树 T 进行处理，删除不必要的节点，插入必要的节点；接着使用三次 B 样条曲线插值对剩余的节点进行平滑处理，生成可执行轨迹.

4.1 基于最大曲率约束的剪枝函数



(a)



(b)

图 3 剪枝算法示意图

Fig.3 Illustration of the pruning algorithm

本文提出的基于最大曲率约束的剪枝函数见 Post_Processing Method 中 Function Pruning(T) 第 5 ～ 23 行. 首先，通过上述 CC-RRT 中的树 T 可以得到一系列从起始状态到目标状态的有效路径点集 Q_1，然后从第 1 个路径点（即初始状态）开始依次连接后序路径点，只要路径点间连线与障碍物空间无交集，那么它们之间的路径点就可以删去，直接以一条直线连接两点. 依次类推，直到碰撞发生，那么此时就以碰撞点的父节点作为新起点，再次执行上述操作，直至到达目标状态. 并将后处理第一阶段得到的路径点存储在 Q_2 中. 接着再根据车辆最大曲率约束条件，对 Q_2 中的相邻路径线段间夹角小于 $\alpha_{\min}$ 的情况，在该线段间基于 $\alpha_{\min}$ 插入一个路径点，从而使尖锐的夹角变得平缓，以便进行 B 样条拟合时生成的轨迹曲率不超过最大曲率的约束. 具体说明如图 3(a) 所示：红色折线为 CC-RRT

算法生成的路径，天蓝色节点 $\boldsymbol{q}_0$、$\boldsymbol{q}_1$、$\boldsymbol{q}_2$、$\boldsymbol{q}_3$ 和 $\boldsymbol{q}_4$ 之间由于与障碍物区域（黑色区域）没有交集，因此可以用直线连接从而删除它们之间冗余的节点；虽然 $\boldsymbol{q}_2$、$\boldsymbol{q}_3$ 和 $\boldsymbol{q}_4$ 之间也可直接连接，但那是由于 $\angle\alpha < \alpha_{\min}$（即 $\angle\boldsymbol{q}_2\boldsymbol{q}_3\boldsymbol{q}_4 < \alpha_{\min}$），所以需要基于 $\alpha_{\min}$ 插入一个节点 $\boldsymbol{q}_{\text{insert}}$ 对这样的尖锐夹角进行处理，使得 $\angle\boldsymbol{q}_{\text{insert}}\boldsymbol{q}_3\boldsymbol{q}_4 = \alpha_{\min}$，如图 3(b) 所示（这里 $\alpha_{\min}$ 为路径最小转角，实验中设定为 90°），最终得到相对更为平缓的路径点序列，如图 3(b) 中绿色点所示.

4.2　B 样条平滑函数

B 样条曲线具有连续性和局部性等优点，使得其在运动规划中的应用比较广泛 [8-9,16]. 因此本文利用它的这些优点来对剪枝处理过的路径点进行曲率连续的拟合，以生成智能车辆可执行的平滑路径.

K 阶 B 样条曲线的表达形式为

$$\boldsymbol{C}(\boldsymbol{u}) = \sum_{i=0}^{n} \boldsymbol{N}_{i,k}(\boldsymbol{u}) \cdot \boldsymbol{P}_i \tag{4}$$

这里 P_i 为控制点，B 样条的基函数可以由 Cox-de Boor 递推关系式得到：

$$N_{i,0}(\boldsymbol{u}) = \begin{cases} 1, & \boldsymbol{u}_i \leqslant \boldsymbol{u} \leqslant \boldsymbol{u}_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$\boldsymbol{N}_{i,k}(\boldsymbol{u}) = \frac{\boldsymbol{u}-\boldsymbol{u}_i}{\boldsymbol{u}_{i+k}-\boldsymbol{u}_i}\boldsymbol{N}_{i,k-1}(\boldsymbol{u}) + \frac{\boldsymbol{u}_{i+k+1}-\boldsymbol{u}}{\boldsymbol{u}_{i+k+1}-\boldsymbol{u}_{i+1}}\boldsymbol{N}_{i+1,k-1}(\boldsymbol{u}) \tag{6}$$

对于 K 阶 B 样条曲线和 n 个控制点，其节点向量为 $\boldsymbol{U} = [\boldsymbol{u}_0, \boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_m]$，其中 $m = n + K$. 为了满足起始状态和目标状态的约束（使得曲线经过起始点和目标点），并且与控制点组成的各个控制边相切，可以采用 K 重节点向量，即节点向量满足：

$$\begin{cases} \boldsymbol{u}_0 = \boldsymbol{u}_1 = \cdots = \boldsymbol{u}_K \\ \boldsymbol{u}_{m-K} = \boldsymbol{u}_{m-K+1} = \cdots = \boldsymbol{u}_m \end{cases} \tag{7}$$

5　实验与分析（Experiment and analysis）

为了验证 CC-RRT 算法的优越性、有效性和实用性，本文分别进行仿真实验和实车实验. 通过在仿真实验中与基本 RRT、双向 RRT（Bi-RRT）算法进行对比，验证了该算法的正确性和优越性. 接着，基于真实的实验平台“智能先锋 II”（如图 4），在一个复杂的露天停车场环境下，验证了该算法的有效性和实用性.

图 4　智能车：“智能先锋 II”

Fig.4　Intelligent vehicle: “Intelligent Pioneer II”

5.1　仿真实验

设定车辆为一个矩形框，整个状态空间尺寸为 580 × 360，X 坐标范围是 [20, 600]，Y 坐标范围是 [40, 400]，障碍物区域为黑色，任意设定车辆的起始位置（蓝色框）和目标位置（红色框），进行运动规划. 图 5(a) ～ (c) 中的红色折线分别为基本 RRT、双向 RRT 和 CC-RRT 三种算法对同一规划问题的求解结果；图 5(d1) 为 CC-RRT 的剪枝算法首先剪枝得到无碰撞的序列点（蓝色圆点），图 5(d2) 为接着对其中的尖锐夹角通过插入节点进行处理（绿色实心圆点）；图 5(e) 为 CC-RRT 算法的剪枝处理结果（绿色实心圆点）和最终生成的平滑路径（天蓝色曲线）；图 5(f) 为 CC-RRT 算法最终规划路径的曲率变化图，可以看出其曲率是连续变化的. 另外，考虑到 RRT 算法的随机性，为了客观评价算法性能的优劣，针对图 5(a) 的实验场景分别对 3 种算法进行了 20 次规划，并对它们各自的平均搜索时间和采样的节点数以及成功搜索次数进行了记录，如表 1 所示.

表 1　仿真实验数据对比

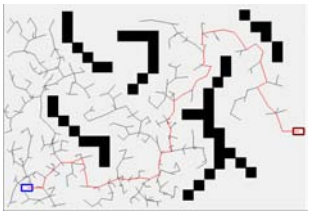
Tab.1　The data comparison of simulation experiment

20 次实验	基本 RRT	双向 RRT	CC-RRT
平均时间 /ms	161.4	74.7	31.6
平均采样节点数	316.8	139.2	68.9
成功次数	15	20	20

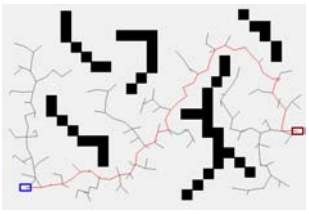
通过图 5 和表 1 的仿真实验相关数据可以看出，CC-RRT 算法在搜索速度和搜索效率上显著提高，搜索的路径也相对其他两种算法更为平缓，并且最终生成的可执行路径曲率也是连续变化的，这满足了车辆的运动约束要求. 真实的实车验证下面一节会具体介绍.

5.2　实车实验

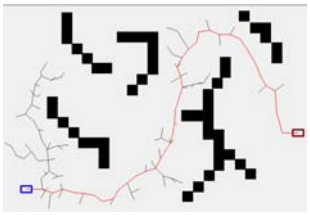
在进行实车验证前，这里先给出智能车“智能先锋 II”的相关参数，如表 2 所示.



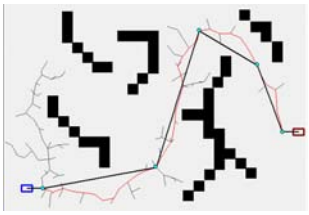
(a) 基本 RRT 构造的随机树



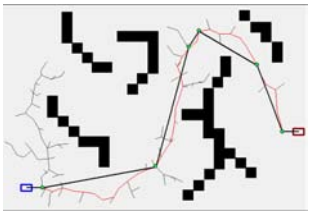
(b) 双向 RRT 构造的随机树



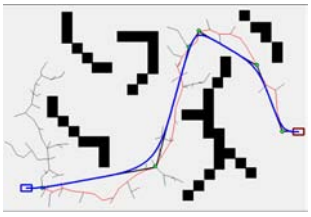
(c) CC-RRT 构造的随机树



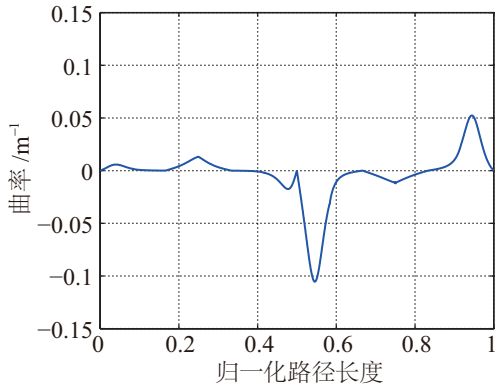
(d1) CC-RRT 后期处理过程 1



(d2) CC-RRT 后期处理过程 2



(e) CC-RRT 生成路径



(f) CC-RRT 生成路径的曲率变化图

图 5 3 种算法的对比及 CC-RRT 算法的结果

Fig.5 The comparison among three algorithms and the results of CC-RRT

表 2 “智能先锋 II”参数

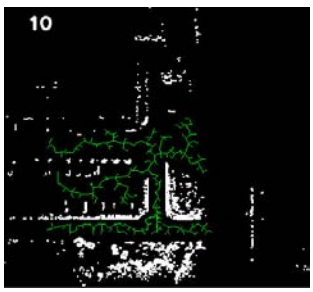
Tab.2 “Intelligent Pioneer II” parameters

最小转弯半径	6.0 m
最大曲率	0.16 m^{-1}
轴距	2.51 m
最大转向角	30°
最大角速度	0.2183 rad/s
最大速度	12 m/s

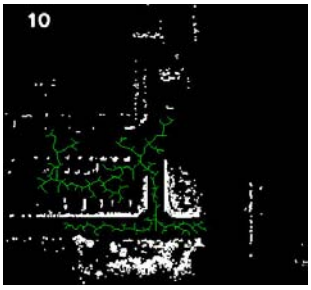
实验场景为一个环境复杂的室外停车场，如图 6 所示. “智能先锋 II”在此环境中，执行一个进入停车场并停在指定车位的任务，以此来验证本文算法的性能.

图 6 实验场景图

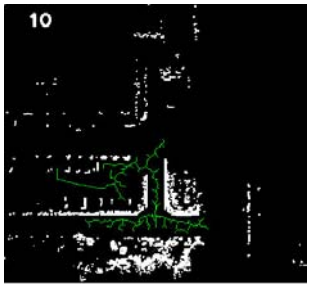
Fig.6 Experimental environment



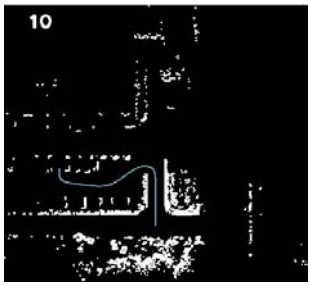
(a) 基本 RRT 算法的搜索结果



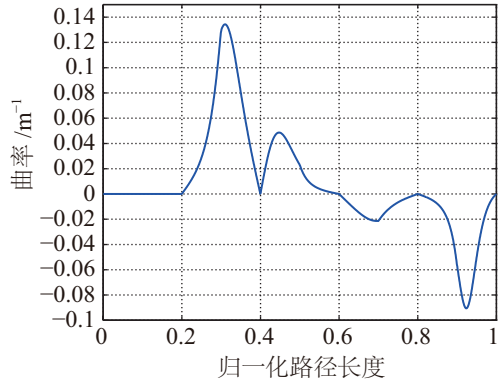
(b) 双向 RRT 算法的搜索结果



(c) CC-RRT 算法的搜索结果



(d) CC-RRT 算法的规划结果



(e) CC-RRT 最终规划路径的曲率变化

图 7 3 种算法的实车实验对比及 CC-RRT 算法的结果

Fig.7 The intelligent vehicle experiment comparison of three algorithms and the planning results of CC-RRT

“智能先锋 II”采用 3 维激光雷达 Velodyne 对周围环境进行感知，并根据感知的结果实时建立一个 512 × 512 的障碍物栅格图，栅格分辨率为 20 cm，最后进行的规划就是基于这个栅格地图. 图

7(a) ～ (c) 分别为基本 RRT、双向 RRT 和 CC-RRT 算法对这一实际场景的搜索结果，其中白色表示障碍物区域，黑色为可通行区域，绿色的细线表示其构建的随机树；图 7(d) 为 CC-RRT 经过后处理方法的最终规划结果（浅蓝色曲线）；图 7(e) 给出了对应的曲率变化图. 为了进一步验证本文算法的优越性，采集了现场环境数据，对这 3 种算法进行了 20 次统计实验，并记录了相关数据于表 3 中. 此外，CC-RRT 算法在“智能先锋 II”上实际规划的性能参数记录在表 4 中.

表 3　实车实验数据对比

Tab.3　The data comparison of the actual experiment

20 次实验	基本 RRT	双向 RRT	CC-RRT
平均时间 /ms	485.4	217.2	72.8
平均采样节点数	901.7	435.8	193.1
成功次数	12	16	20

表 4　CC-RRT 算法的实车实验数据

Tab.4　The intelligent vehicle experimental data of CC-RRT algorithm

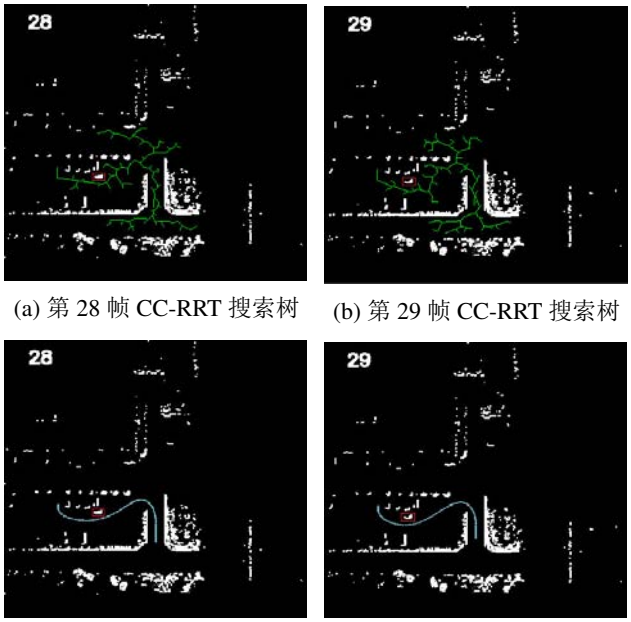
总规划时间		最大曲率
78 ms		
搜索时间	后处理时间	$0.1343\,\mathrm{m^{-1}} < K_{\max} = 0.16\,\mathrm{m^{-1}}$
56 ms	22 ms	

对比表 3 中的数据可以看出，CC-RRT 算法在真实的复杂环境下，无论是在搜索效率和成功率上依然优于另外两种算法. 此外，根据表 4 中记录的实验数据可以看出，经过后处理算法，最终规划出的路径不仅曲率连续而且满足车辆本身的曲率约束条件（小于 $K_{\max}$），而且其规划周期小于感知系统的更新周期 120 ms，也符合智能车辆对实时性的要求，进一步验证了该算法的有效性和实用性.

5.3　动态环境处理方法

根据上文分析可以看出，CC-RRT 算法规划速度满足实时性要求，在动态环境下对每一帧都重新规划，将动态环境当作静态环境处理，以完成动态环境的实时规划. 但是由于算法的随机性，两个相邻规划周期生成轨迹的差异性，容易造成车辆行驶过程中抖动. 为了解决这一问题，借鉴文 [5] 的思想：每个新的规划周期开始，均以前一个规划周期生成的轨迹中距离根节点等于 D 的位置作为该规划周期新的搜索树的根节点（由于是在低速环境下，试验中设定 $D = 1$ m 的阈值距离能保持车辆整体的平稳性），从而保证规划的轨迹总是连续的，便于车辆执行.

图 8 中给出了 CC-RRT 连续两帧的动态规划过程. CC-RRT 采用的是实时重规划，对于环境中的动态障碍物按照静态障碍物进行处理. 从图 8(a) 和图 8(b) 可以看出，搜索过程中避开了动态障碍物（由于动态障碍物是按照静态障碍物处理的，所以动态障碍物在感知栅格图中仍为白色区域，为了能叙述清楚，图 8 中动态障碍物被作者用红色矩形框标出）；从图 8(c) 和图 8(d) 可以看出，所生成的最终轨迹也都成功地绕开了移动障碍物. 因此，该算法对于动态环境也是适用的.



(a) 第 28 帧 CC-RRT 搜索树　(b) 第 29 帧 CC-RRT 搜索树

(c) 第 28 帧 CC-RRT 规划结果　(d) 第 29 帧 CC-RRT 规划结果

图 8　动态环境中 CC-RRT 的规划结果

Fig.8　The planning results of CC-RRT in dynamic environment

6　结论（Conclusion）

以往的一些运动规划算法大多数都没考虑机器人自身的约束，这样规划出的路径虽然能避开障碍物，但是对于机器人来说不一定是可执行的. 尤其是在存在大量不规则障碍物且分布不均匀的复杂环境下，对于智能车这类具有非完整性约束的机器人，问题就变得更为严重. 对此，本文提出了一种实用的连续曲率 RRT 算法，它将环境约束以及智能车自身约束与 RRT 搜索算法和后处理优化方法结合了起来，仿真实验和实车实验同时表明该算法能有效地解决复杂环境下智能车辆的运动规划问题.

本文算法不仅仅适用于智能车，也同样适用于其他机器人运动规划问题的求解. 另外，该算法目前解决的是低速运动环境下的规划问题，下一步尝

试解决高速运动规划问题，进一步提高该算法的鲁棒性和普适性，以便更好地满足实际工程的需要.

参考文献（References）

[1] 刘华军，杨静宇，陆建峰，等. 移动机器人运动规划研究综述 [J]. 中国工程科学，2006，8(1)：85-94.

Liu H J, Yang J Y, Lu J F, et al. Research on mobile robots motion planning: A survey[J]. Engineering Science, 2006, 8(1): 85-94.

[2] 康亮，赵春霞，郭剑辉. 基于模糊滚动 RRT 算法的移动机器人路径规划 [J]. 南京理工大学学报：自然科学版，2010，34(5)：642-648.

Kang L, Zhao C X, Guo J H. Path planning based on fuzzy rolling rapidly-exploring random tree for mobile robot[J]. Journal of Nanjing University of Science and Technology: Natural Science, 2010, 34(5): 642-648.

[3] 宋金泽，戴斌，单恩忠，等. 一种改进的 RRT 路径规划算法 [J]. 电子学报，2010，38(B02)：225-228.

Song J Z, Dai B, Shan E Z, et al. An improved RRT path planning algorithm[J]. Acta Electronica Sinica, 2010, 38(B02): 225-228.

[4] 徐娜，陈雄，孔庆生，等. 非完整约束下的机器人运动规划算法 [J]. 机器人，2011，33(6)：666-672.

Xu N, Chen X, Kong Q S, et al. Motion planning for robot with nonholonomic constraints[J]. Robot, 2011, 33(6): 666-672.

[5] Kuwata Y, Teo J, Fiore G, et al. Real-time motion planning with applications to autonomous urban driving[J]. IEEE Transactions on Control Systems Technology, 2009, 17(5): 1105-1118.

[6] Fraichard T, Scheuer A. From Reeds and Shepp's to continuous-curvature paths[J]. IEEE Transactions on Robotics, 2004, 20(6): 1025-1035.

[7] Elbanhawi M, Simic M. Randomised kinodynamic motion planning for an autonomous vehicle in semi-structured agricultural areas[J]. Biosystems Engineering, 2014, 126: 30-44.

[8] Elbanhawi M, Simic M, Jazar R. Continuous-curvature bounded trajectory planning using parametric splines[M]//Frontiers in Artificial Intelligence and Applications, vol.262. Amsterdam, Netherlands: IOS Press, 2014: 513-522.

[9] Gómez-Bravo F, Cuesta F, Ollero A, et al. Continuous curvature path generation based on β-spline curves for parking manoeuvres[J]. Robotics and Autonomous Systems, 2008, 56(4): 360-372.

[10] Du M B, Chen J J, Zhao P, et al. An improved RRT-based motion planner for autonomous vehicle in cluttered environments[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2014: 4674-4679.

[11] Lee J, Kwon O, Zhang L, et al. SR-RRT: Selective retraction-based RRT planner[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2012: 2543-2550.

[12] Rodriguez S, Tang X, Lien J M, et al. An obstacle-based rapidly-exploring random tree[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2006: 895-900.

[13] LaValle S M, Kuffner J J. Rapidly-exploring random trees: Progress and prospects[C]//4th International Workshop on Algorithmic Foundations of Robotics. Wellesley, USA: A K Peters, 2000: 293-308.

[14] Kuffner J J Jr, LaValle S M. RRT-connect: An efficient approach to single-query path planning[C]//IEEE International Conference on Robotics and Automation. Piscataway, USA: IEEE, 2000: 995-1001.

[15] Lau B, Sprunk C, Burgard W. Kinodynamic motion planning for mobile robots using splines[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2009: 2427-2433.

[16] Koyuncu E, Inalhan G. A probabilistic B-spline motion planning algorithm for unmanned helicopters flying in dense 3D environments[C]//IEEE/RSJ International Conference on Intelligent Robots and Systems. Piscataway, USA: IEEE, 2008: 815-821.

[17] Cheng P. Reducing RRT metric sensitivity for motion planning with differential constraints[D]. Ames, USA: Iowa State University, 2001.

[18] LaValle S M. Rapidly-exploring random trees: A new tool path planning[R]. Ames, USA: Iowa State University, 1998.

作者简介：

杜明博（1988 –），男，博士生. 研究领域：机器人，智能车辆，运动规划.

梅 涛（1962 –），男，博士，研究员. 研究领域：机器人，传感器技术，人工智能.

陈佳佳（1986 –），男，博士. 研究领域：机器人，智能决策，机器学习.