文章编号:1671-4598(2018)07-0282-05 DOI:10.16526/j.cnki.11—4762/tp.2018.07.062 中图分类号: 文献标识码:A

基于改进 A* 算法机器人路径规划研究

王小红, 叶 涛
(青海民族大学计算机学院, 青海 西宁 810008)

摘要: 针对移动机器人全局路径规划问题提出一种改进 A* 算法; 首先建立栅格地图, 基于传统 A* 算法, 进行邻域扩展, 将传统 8 邻域扩展到 24 邻域, 使路径方向具有更多选择, 减少不必要的转折点; 优化改进 A* 算法的启发式函数, 不再采用单一的曼哈顿距离或者欧几里得距离, 将其进行融合改进, 剔除路径中冗余节点和多余转折点; 最后将全局路径与动态窗口法相结合, 结合各自的优点, 充分考虑到机器人全局最优路径的同时能安全避开障碍物, 得到一条平滑轨迹; 各个算法进行验证之后采用 ROS 平台对系统进行仿真分析, 实验结果表明, 改进后算法具有更优秀的路径规划能力。

关键词: 邻域扩展, 启发式函数, A* 算法, 动态窗口法

Research on Robot Path Planning Based on Improved A* Algorithm

Wang Xiaohong, Ye Tao
(College of Computer Science, Qinghai Nationalities University, Qinghai city of Xining Province 810008, China)

Abstract: In this paper, An improved A* algorithm for mobile robot global path planning is improved. We first set up a grid map and extend the neighborhood based on the traditional A* algorithm, extending the traditional 8 neighborhood to the 24 neighborhood, making the path direction more selective and reducing the unnecessary turning point. The heuristic function of the A* algorithm is optimized, and the single Manhattan distance or Euclidean distance is no longer used, and it is fused and improved to eliminate the redundant nodes and the superfluous turning points in the path. In the end, we combine the global path with the dynamic window method and combine the advantages of each of them, and take full consideration of the global optimal path of the robot to avoid the obstacles and get a smooth trajectory. After each algorithm is verified, the ROS platform is used to simulate the system. The experimental results show that the improved algorithm has better path planning ability.

Keywords: neighborhood extension; heuristic function; A* algorithm; Dynamic window method

0 引言

导航是智能机器人非常重要的功能, 也是机器人领域的一个研究重点。路径规划技术作为导航的核心, 其主要任务是通过机体搭载的传感器获取周围环境信息, 根据已知信息规划出一条从起始点到目标点的最优路径[1]。近些年来, 针对机器人路径规划, 诸多学者通过展开大量研究, 提出各种面向不同场景的路径规划技术。其中比较著名的有快速搜索随机树[2-3], 概率路图法[4], 也有基于节点搜索的 A* 算法和 D* 算法[5-6], 基于启发式的遗传算法, 粒子群算法和蚁群算法等[7-9], 还有应用于局部路径规划的动态窗口法和人工势场法等[10-12]。全局路径规划的前提是机器人需要有完整地图信息, 而局部路径规划中机器人只需要知道局部信息, 即通过传感器获得的自身周边障碍物信息。

收稿日期:2018-05-13; **修回日期**:2018-05-23。
基金项目:2017 青海省科技计划项目(2017—ZJ—912); 国家自然科学基金项目(2014JK1160); XX 省自然科学基金项目(2014sky007); XX 省教育厅基金项目(2014jyjx209)。

作者简介:王小红(1977-),女,硕士,讲师,主要从事软件工程、嵌入式系统,数据库方向的研究。
叶 涛(1972),男,四川隆昌人,在读博士,副教授,主要从事网络安全方向的研究。

在静态环境中 A* 算法是搜索最短路径最有效的算法, 因此被广泛应用于实际路径规划中。由于该算法本身搜索原理的限制, 规划出来的路径虽然效率高, 但是转折点较多, 对于移动机器人来说不利于直接执行。文献[13]提出一种平滑 A* 算法, 旨在解决栅格环境下移动机器人 A* 算法规划出的路径折线多, 转折次数多和转折角大的问题, 最终获得了较优路径, 该改进算法只是优化了路径, 并没有减少路径长度, 机器人执行时所付出的代价相差不大, 同时也不具备很好的动态避障能力。文献[14]将人工势场法和蚁群算法进行融合, 在栅格环境中, 以人工势场法的规则信息作为蚁群算法寻优的基础, 引入势场合力作为蚂蚁搜索路径点的部分启发信息, 该方法解决了人工势场法容易陷入极值点的缺点, 但是计算复杂度太大。动态窗口法实时性好, 具有良好的躲避障碍物能力, 但是该方法只是一种局部路径规划, 并没有考虑到机器人的全局路径规划信息, 使得最后规划出来的全局路径并不是最优。

基于以上各方法的特点, 本文提出一种 A* 算法的改进方法, 对邻域进行扩展, 并且将启发式函数进行修改以适应具体环境需求, 使传统 A* 算法能够更加灵活, 规划出来的路径转折点少, 同时全局路径更加平滑。将改进 A* 算法和动态窗口法进行结合, 将两种算法的优点结合起来, 既能进行全局最优路径的搜索, 也让算法具备良好的

避障能力，躲避障碍物。

1　改进 A * 算法

1.1　传统 A * 算法介绍

传统 A * 算法是一种标准的启发式函数，是静态环境中寻找最优路径的有效方法，其核心在于一个估价函数。从初始节点开始，按照启发函数对周围的节点开始搜索，选取周围最优的一个节点进行扩展，重复这一过程，直到搜索到目标点，然后由目标点回溯到起始点形成一条全局规划轨迹。其估价函数为：

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

其中：$f(n)$ 是估计函数，表示的是从起始点到目标点的评价估计量，$g(n)$ 表示的是初始点到当前节点的实际代价值，而 $h(n)$ 则表示当前节点到目标点的估计代价值。若将启发式函数设置成零，即 $h(n)=0$，估价函数就完全由代价函数 $g(n)$ 决定，此时 A * 算法就退化为基于贪心策略的 Dijkstra 算法，搜索效率大打折扣。$h(n)$ 启发函数直接决定了 A * 算法是否高效，$h(n)$ 中包含越多的启发式信息，A * 算法效率就高，在搜索较少节点的情况下就可以找到最优全局路径，反之，若启发式信息量越少，规划出的轨迹就离最优轨迹相差越远。

1.2　邻域扩展

传统的 A * 算法只能向节点周围 8 邻域进行扩展，这种情况下机器人只能沿着八个方向运动，每个方向之间至少有四十五度的夹角，这就限制了机器人的运动，直接导致最终规划出的全局路径转折节点多，轨迹不够平滑。如图 1 所示，本文将 8 邻域搜索扩展到 24 邻域，机器人能够以更小的角度行进，从而使轨迹更加平滑。

图中中心黑色圆点表示目前机器人所在位置，传统方法中，机器人只能向 8 个方向扩展，即扩展到图中小圆点所在位置，扩展后一共可以向 16 个方向进行搜索，扩展点为图中 24 个箭头所在位置。传统 A * 算法中，由于搜索的节点数目较少，很可能传统 A * 算法中，由于搜索的节点数目较少，在前期就将更优秀的节点从列表中删除，从而只能找到次优路径，扩展后的算法由于搜索的节点更多，在很大程度上可以避免这种情况的发生，进一步优化了路径。

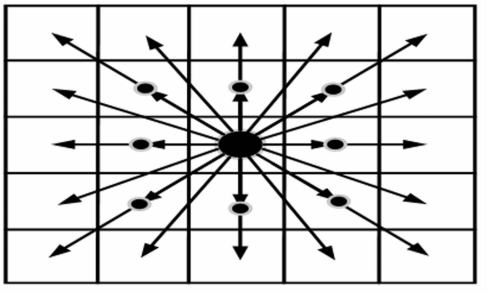


图 1　24 邻域扩展节点示意图

1.3　启发式函数的优化

在 A * 算法中，传统的启发式函数采用曼哈顿距离或者欧式距离。曼哈顿距离是指两个坐标点之间横轴和竖轴绝对值之和，假设每个相邻单位之间的路径代价为 C，n 表示当前点，goal 表示目标点，可以得到基于曼哈顿距离的启发函数为：

$$h(n)=C*(abs(n_x-goal_x)+abs(n_y-goal_y)) \tag{2}$$

若机器人单位可以进行任意方向的移动，我们就可以用欧式距离来表示对应的启发函数。欧氏距离指的是两点之间的直线距离，假设机器人经过单位长度的路径的代价为 C，则欧式距离的启发函数为：

$$h(n)=C*\sqrt{(n_x-goal_x)^2+(n_y-goal_y)^2} \tag{3}$$

但是由于 A * 算法基于栅格地图，机器人不能全向移动，所以在实际应用中，这种启发式函数需要进行转化，速度较曼哈顿慢，但是路径更短。

基于上述 24 邻域优化，本文提出一种新的启发函数，更真实的描述节点扩展之后的代价，假设单位节点代价函数还是 C，改进后的启发式函数如下。

若：$|n_y-goal_y|\geq|n_x-goal_x|$：

$$h(n)=C*((\sqrt{5}-1)*abs(n_x-goal_x)+abs(n_y-goal_y)) \tag{4}$$

若：$|n_y-goal_y|<|n_x-goal_x|$：

$$h(n)=C*((\sqrt{5}-1)*abs(n_y-goal_y)+abs(n_x-goal_x)) \tag{5}$$

1.4　轨迹平滑

采用全局路径规划算法得到的路径有时候会歪歪扭扭，不利于机器人的执行，因此需要对路径进行处理，例如本文中对路径中某些冗余节点进行删除，并将更优的路径选择出来。轨迹平滑示意图如图 2 所示。

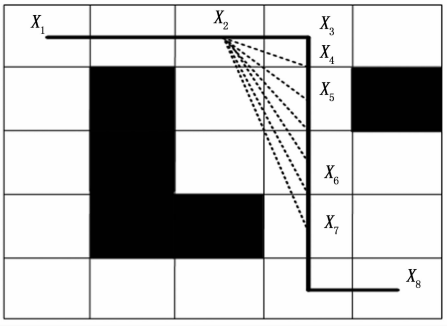


图 2　剔除冗余节点示意图

图中黑色部分表示障碍物，白色部分为可通行空白区域，黑实线表示未经优化的路径，X_1 到 X_8 代表路径上的不同节点，虚线表示的是机器人可经过的路径。以 X_2 节点为例，按照节点优化策略，X_2 节点跳过相邻节点，连接至 X_4，比较 X_2X_4 之间是否经过障碍物，即机器人能否顺利通过，若可以经过，再检查距离和之前相比是否变短，路径确实变短后，检测能否被机器人执行，能否执行主要查看 24 扩展邻域搜索范围，若在 24 扩展邻域范围内，则可以顺利进行节点优化并将该路径加入到候选表中。检查完 X_4 节点，我们继续检查 X_5 节点，重复上述流程，若 X_5 也符

合上述检测标准，则将 X_4 和 X_5 两个节点进行对比，由分析可知，X_2X_5 路线所优化的路程长度大于 X_2X_4，因此将 X_5 替换掉候选表中的 X_4。由于机器人搜索区域的限制（24 邻域），上图中最佳的路线为 X_2X_6。从头到尾重复上述流程，最后将选出一条更为平滑的路径。

1.5 改进 A * 算法实验结果

本文选用 60 * 60 的栅格对 A * 算法和改进 A * 算法分别进行仿真验证，实验结果如下图所示。

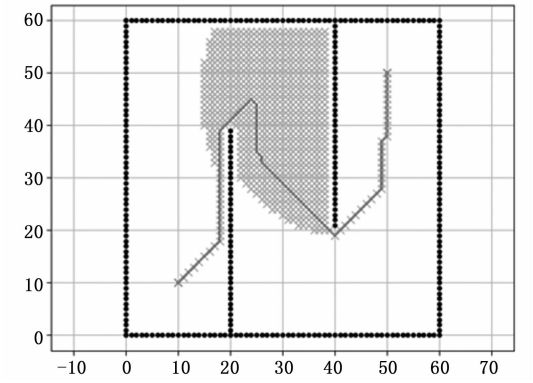


图 3 原始 A * 算法

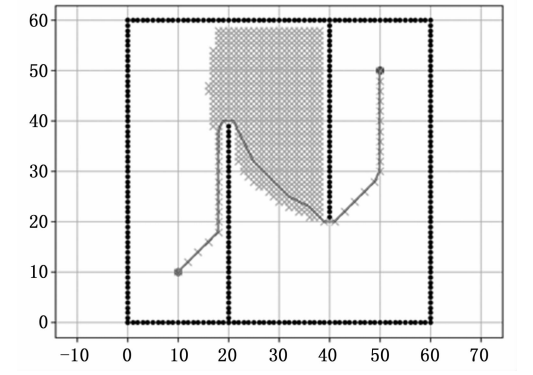


图 4 改进 A * 算法

图中位于坐标（10，10）处的红点为机器人起始点，位于坐标（50，50）处的点为机器人要到达的目标点，实心点表示的是障碍物，白色区域为空白地带，浅色区域为算法搜索过的点，曲线表示最终规划出的去全局路径。由上述两个图可以直观的看出，改进后的 A * 算法能够以更少的代价到达目标点，而且轨迹更加平滑，更加理想。

2 动态窗口法

动态窗口法的基本思路是：在速度空间中，根据自身运动模型，对多组速度进行采样，分析出在各组不同速度下，一段时间内机器人的运动轨迹，采用一定的评价函数对该组轨迹进行评价，选择出评价最高的一组来执行，直到下一执行时间的到来。动态窗口法示意图如图 5 所示。

如图 5 所示，矩形表示机器人，物体表示障碍物，虚线表示的是动态窗口法规划出来的下一时刻内的多组轨迹，由图可知，有三组轨迹将碰到障碍物，因此将这三条轨迹

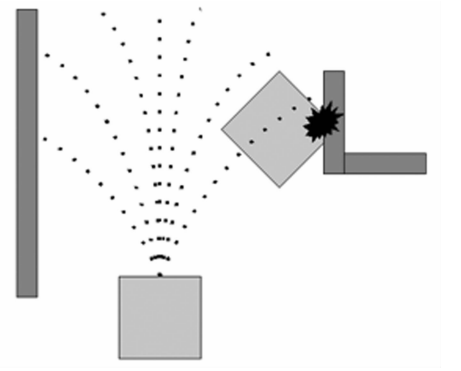


图 5 动态窗口法示意图

舍弃，然后将剩下的轨迹通过其他评价函数进行评分，最终选择出最适合的一条轨迹执行。

2.1 机器人运动模型

在动态窗口法中，对机器人建立运动模型是最基本的步骤。

假设机器人在间隔时间内沿着直线运动。若机器人是非完整约束的，即不能进行全向移动，只能进行前进和旋转。我们先计算机器人在相邻时刻的轨迹。假设机器人在该相邻时刻之间是匀速运动的，由于相邻时间间隔很短，我们将其进行近似处理成一段直线。此时可以得到机器人在世界坐标系中的位移：

$$\Delta x = \upsilon \Delta t \cos(\theta_t) \tag{6}$$

$$\Delta y = \upsilon \Delta t \sin(\theta_t) \tag{7}$$

由上述位移公式我们可以得到机器人在一段时间内的轨迹：

$$\left.\begin{aligned} x &= x + \upsilon \Delta t \cos(\theta_t) \\ y &= y + \upsilon \Delta t \sin(\theta_t) \\ \theta_t &= \theta_t + \omega \theta_t \end{aligned}\right\} \tag{8}$$

若机器人能够进行全向运动，我们需要另外将机器人在纵轴移动的距离投影到世界坐标系。此时我们推导：

$$\left.\begin{aligned} x &= x + \upsilon \Delta t \cos(\theta_t) - \upsilon_y \Delta t \sin(\theta_t) \\ y &= y + \upsilon \Delta t \sin(\theta_t) + \upsilon_y \Delta t \cos(\theta_t) \\ \theta_t &= \theta_t + \omega \Delta_t \end{aligned}\right\} \tag{9}$$

一段相邻时间内假设机器人的轨迹是直线是不准确的，若要得到更精确的结果，我们需要将轨迹假设为曲线，最终轨迹是由许多段圆弧构成。假设机器人不能进行全向运动，那么轨迹中圆弧的半径为：

$$\gamma = \upsilon / \omega \tag{10}$$

机器人坐标为：

$$\left.\begin{aligned} x &= x - \gamma \sin(\theta_t) + \gamma \sin(\theta_t + \omega \Delta t) \\ y &= y - \gamma \cos(\theta_t) - \gamma \cos(\theta_t + \omega \Delta t) \\ \theta_t &= \theta_t + \omega \Delta_t \end{aligned}\right\} \tag{11}$$

为了使最后结果更加精确，本文使用的是第二种模型，即假设机器人在相邻时间内的轨迹是圆弧。

2.2 机器人速度采样

得到了机器人运动模型之后，下一步就是确定机器人

的速度，这一步是动态窗口法的核心，只有得到了机器人速度才能对机器人的轨迹进行预测。在速度空间内，机器人可以在当前时刻理论上可以由无穷多组，但是由于各种条件限制，可以将速度限制在某些范围内。

机器人受到自身极限速度限制：

$$V_m = \{v \in [v_{\min}, v_{\max}], \omega \in [\omega_{\min}, \omega_{\max}]\} \tag{12}$$

机器人受到电机性能制约：由于不同电机具有不同的性能，提供给机器人的最大加减速也不一样，因此在一个动态窗口内显示的速度就是机器人能够实际达到的速度，设 v_c 和 ω_c 分别为机器人当前速度和角速度，可得机器人在电机性能约束下的速度范围：

$$V_d = \{(v,\omega) \mid v \in [v_c - \dot{v}_b \Delta t, v_c + \dot{v}_a \Delta t], \omega \in [\omega_c - \dot{\omega}_b \Delta t, \omega_c + \dot{\omega}_a \Delta t]\} \tag{13}$$

机器人受障碍物约束：进行局部轨迹规划最重要的一点就是让机器人能够避开障碍物，动态窗口法刚开始并不知道障碍物的位置，在进行速度的不断选择和轨迹评价后，若有轨迹碰到障碍物，此时就可以计算出机器人到障碍物的距离，然后根据当前机器人本身性能，看是否可以以最大减速度在障碍物之前停下，如果计算出无法在障碍物之前停下，则将该轨迹删除。该条件下速度范围可以由下面的公式来进行约束：

$$V_a = \{(v,\omega) \mid v \leqslant \sqrt{2 \cdot dist(v,\omega) \cdot \dot{v}_b}, \omega \leqslant \sqrt{2 \cdot dist(v,\omega) \cdot \dot{\omega}_b}\} \tag{14}$$

2.3　评价函数

动态窗口法最后一步就是对预测的轨迹进行评价，我们采用三个不同的方面共同对轨迹进行评价，客观的得到最优路径。

第一个评价函数的方位角评价函数，方位角指的是机器人到达规划出的轨迹末端时，当前朝向和目标点朝向的角度差。在这里用 $180^\circ - \theta$ 来表示，即角度差越小，评价函数值越大，接受程度越高，方位角示意图如图 6 所示。

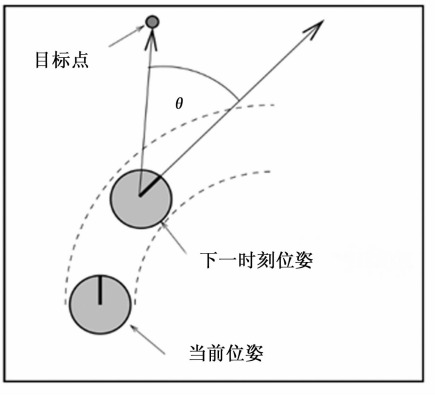


图 6　动态窗口法方位角示意图

第二个评价函数为距离评价函数，也就是轨迹末端距离障碍物的远近程度，若轨迹与障碍物相交，则将次轨迹舍弃，将其设定为一个常数。

第三个评价函数为速度评价函数，用来评价当前机器人速度大小，速度间接影响到机器人距离函数。

由上述三个评价函数得到的总评价函数为：

$$G(v,\omega) = \sigma(\alpha \cdot heading(v,\omega) + \beta \cdot dist(v,\omega) + \gamma \cdot velocity(v,\omega)) \tag{14}$$

上述公式中 $heading\ (v,\ \omega)$，$dist\ (v,\ \omega)$，$velocity\ (v,\ \omega)$ 分别表示机器人的方位角评价函数，距离评价函数和速度评价函数。为了使轨迹更加平滑，要对评价函数进行归一化处理，以方位角评价函数为例，归一化指的就是每一项除以每一项的总和：

$$normal_{heading}(i) = \frac{heading(i)}{\sum_{i=1}^{n} heading(i)} \tag{15}$$

同理，其他两个评价函数也可用相同方法进行归一化平滑处理。

2.4　动态窗口法验证

如图 7 所示，为了验证动态窗口法的性能，我们对该算法进行仿真验证，机器人需要绕过障碍物到达目标点。一连串线条表示动态窗口法所评价的轨迹，在这些轨迹中选择一条最优轨迹执行。

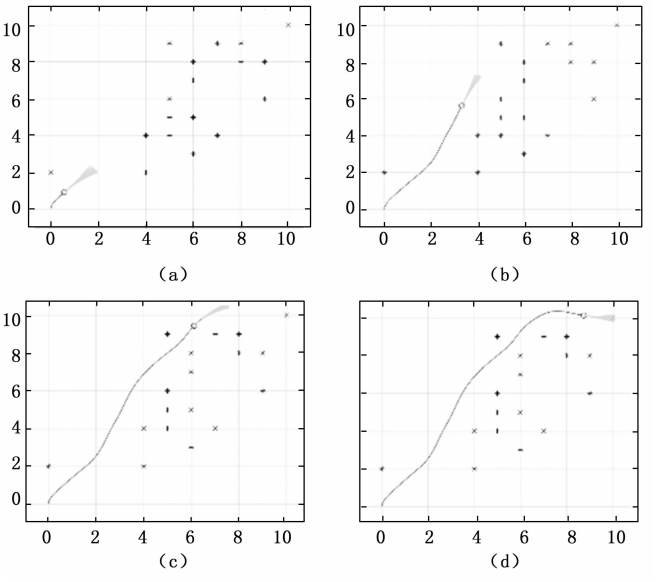


图 7　动态窗口法的 MATLAB 仿真结果

3　系统综合实验

3.1　实验仿真平台介绍

本文仿真采用的是 Ubuntu 系统下的机器人操作系统（ROS，Robotic Operating System），该操作系统于 2007 年诞生于斯坦福大学人工智能实验室，虽然被称为操作系统，但它只是借鉴了操作系统的精华，提供给用户一系列方便的服务。作为一个开源操作系统，ROS 为广大研发人员提供了大量接口，支持多种编程语言，模块化编程，大大提高开发效率。

ROS 还为我们提供了大量软件接口，例如广泛使用的 Gazebo 物理仿真平台，Rviz 数据可视化工具等，我们可以用这些工具很方便的进行开发。

3.2 实验结果

本实验在 Gazebo 平台下搭建了轮式机器人以及具体的 3D 仿真环境，同主题将 Gazebo 与 Rviz 连接起来，这样我们就可以将机器人在仿真环境中将具体的机器人采集到的数据进行直观展示。物理仿真结果如图 8 所示。

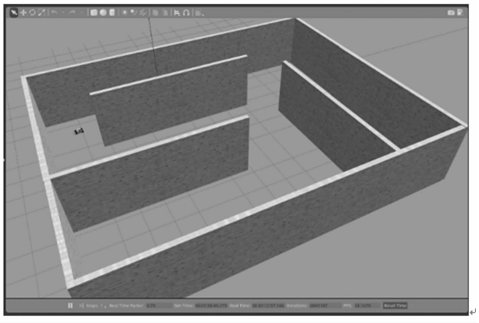
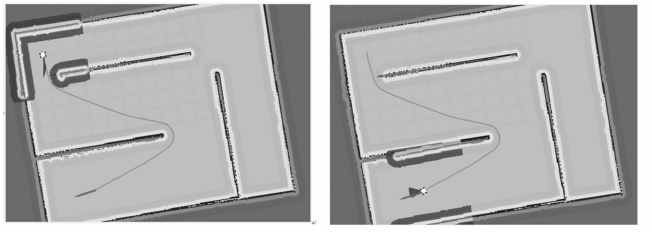


图 8 Gazebo 物理仿真环境搭建

在 Rviz 中设定机器人初始点和目标点，橙色曲线表示为改进 A * 算法规划出的全局路径，红色箭头表示设定的机器人最终位姿，机器人前方一系列绿色箭头指的就是利用动态窗口法计算出的诸多预测轨迹，在其中找到最优的一条执行。为验证机器人避障性能，设置机器人最大线速度为 2 m/s，最大线加速度为 0.8 m/s^2，评价函数参数 $\alpha=0.1$，$\beta=0.3$，$\gamma=0.2$。从仿真结果可以看出机器人能够很顺利的从初始点沿着全局路径前进，并按照自身机械结构限制进行路径的改良。



(a) 机器人位于起始点 (b) 机器人到达终点

图 9 Rviz 数据可视化结果

通过统计仿真过程中路径节点，运行时间等数据，我们可以更加直观的看出改进 A * 算法和传统 A * 算法之间的差别，统计表如表 1 所示。

表 1 两种算法对比

	节点数	运行时间	平均速度
传统算法	129	117.5 s	1.3 m/s
改进算法	86	106.8 s	1.4 m/s

从表中可以看出，改进后的 A * 算法搜索的节点数目更少，轨迹也更短，直接导致效率的增加，机器人行动速度变快。

4 结论

本文针对移动机器人路径规划，提出一种改进 A * 算法，将算法搜索区域扩展到 24 邻域，克服传统 A * 算法路径转折点多，不够平滑，不易被机器人执行的缺点，并设计了一种节点优化策略，遍历规划好的路径节点，剔除冗余节点，使路径更优。最后加入动态窗口法，极大提高了机器人避障能力，在保证路径全局最优的同时，可使机器人平滑的到达目标点。

本文设计的改进 A * 算法在小型地图上效果显著，但是在大规模地图中，由于计算量过大，导致机器人实时性能低下，效果不如传统算法，需要进行进一步优化。

参考文献：

[1] Eele A J, Richards A. Path－Planning with Avoidance Using Nonlinear Branch－and－Bound Optimization [J]. Journal of Guidance Control & Dynamics, 2015, 32 (2): 384－394.

[2] Bekris K E, Chen B Y, Ladd A M, et al. Multiple query probabilistic roadmap planning using single query planning primitives [A]. Ieee/rsj International Conference on Intelligent Robots and Systems [C]. IEEE, 2003 (1): 656－661.

[3] Akinc M, Bekris K E, Chen B Y, et al. Probabilistic Roadmaps of Trees for Parallel Computation of Multiple Query Roadmaps [M]. Robotics Research. The Eleventh International Symposium. Springer Berlin Heidelberg, 2005: 80－89.

[4] Kavraki L E, Latombe J C, Overmars M H. Probabilistic Roadmaps for Path Planning in High－Dimensional Configuration Spaces [A]. IEEE International Conference on Robotics and Automation [A]. 1996: 566－580.

[5] 张 彪，曹其新，王雯珊. 使用三维栅格地图的移动机器人路径规划 [J]. 西安交通大学学报，2013，47 (10)：57－61.

[6] 史久根，李凯业. 基于分层改进 D～ * 算法的室内路径规划 [J]. 计算机应用研究，2015，32 (12)：3609－3612.

[7] 吴晓涛，孙增圻. 用遗传算法进行路径规划 [J]. 清华大学学报：自然科学版，1995 (5)：14－19.

[8] 李淑红，张巧荣. 二进制粒子群算法在路径规划中的应用 [J]. 计算机工程与设计，2009，30 (21)：4953－4955.

[9] 朱庆保，张玉兰. 基于栅格法的机器人路径规划蚁群算法 [J]. 机器人，2005，27 (2)：132－136.

[10] 张建英，赵志萍，刘暾. 基于人工势场法的机器人路径规划 [J]. 哈尔滨工业大学学报，2006，38 (8)：1306－1309.

[11] Fox D, Burgard W, Thrun S. The dynamic window approach to collision avoidance [J]. IEEE Robotics & Automation Magazine, 2002, 4 (1): 23－33.

[12] Min G P, Jeon J H, Min C L. Obstacle avoidance for mobile robots using artificial potential field approach with simulated annealing [A]. IEEE International Symposium on Industrial Electronics [C]. 2001. Proceedings. ISIE. IEEE, 2001 (3): 1530－1535.

[13] 王红卫，马 勇，谢 勇，等. 基于平滑 A～ * 算法的移动机器人路径规划 [J]. 同济大学学报 (自然科学版)，2010，38 (11)：1647－1650.

[14] 董 晔，吴丽娟. 基于混合人工势场－蚁群算法的机器人避障 [J]. 辽宁科技大学学报，2015 (3)：212－216.