基于改进A*算法的机器人路径规划方法研究

刘 钰[1],陆建峰[2],蔡海舟[1]
(1.金陵科技学院 信息技术学院,江苏 南京 211169;
2.南京理工大学 计算机科学与技术学院,江苏 南京 210094)

摘 要:在机器人智能控制的研究中,路径规划是移动机器人研究的重要内容。为提高常规路径规划方法中执行效率和稳定性,采用头尾双向搜索法对普通A*算法进行优化,即分别从起始节点和目标节点开始扩展,直到在中途有相同的临界子节点。同时改进节点h值的计算方式,以减少扩展节点的规模,并在仿真平台上进行机器人路径规划仿真,改进算法效果得以优化验证。仿真实验结果表明,该方法的寻优能力及稳定性均优于普通A*算法,可使智能机器人更高效地进行自主导航。

关键词:A*算法;机器人;路径规划;栅格
中图分类号:TP301.6 **文献标识码**:A **文章编号**:1673-629X(2012)12-0108-04

Research on Path Planning Method of Robot Based on Improved A* Algorithm

LIU Yu[1], LU Jian-feng[2], CAI Hai-zhou[1]
(1. School of Information Technology, Jinling Institute of Technology, Nanjing 211169, China;
2. School of Computer Science & Technology, Nanjing University of Science & Technology, Nanjing 210094, China)

Abstract: Path planning is important subjects in research of mobile robots control. In order to increase the efficiency and stability of usual ways used in path planning, the searching method of double direction has been used to optimize common A* algorithm. Such method may expand the starting node and the goal node at the same time, and if a same adjacency child node was found in the way, the algorithm would be terminated. In the meanwhile, the calculating way of the h value of a node was also improved to reduce the size of the extended nodes. By the simulation of path planning on the virtual platform, the results of simulating experiments prove that the ability of finding the best solution and the stability of this method are greatly improved compared with common A* method, consequently the planning path of intelligent robots can be much efficient.

Key words: A* algorithm; robot; path planning; grid

0 引 言

路径规划是移动机器人自主导航的关键技术之一。所谓路径规划是指移动机器人按照某一性能指标(如距离、时间、能量等)搜索一条从起始状态到目标状态的最优或次优路径[1]。用于路径规划的算法有很多种,A*算法就是其中一种比较常用和高效的算法[2,3]。

1980年Nilsson提出了A*算法,A*算法属于启发式搜索中的最佳优先搜索方法,是一种基于面积的全局规划算法,是在图中给定节点间根据估价函数搜寻较优路径的方法[4]。A*算法在人工智能领域中有所应用,由于其效率高、速度快而主要用于路径的搜索。启发函数$h(n)$告诉A*从任意节点n到目标节点的最小代价评估值[5,6]。根据不同的地形,需要选取不同的启发函数,以此来提高A*算法的效率。

文中对A*算法进行性能改进,从目标节点和起始节点同时进行扩展,并且利用一种新的方法计算各节点的h值,从算法运行速度和扩展节点规模大小来评估A*算法的性能。

1 A*算法原理

A*算法是一种快速、高效,能够应用于大多地形的启发式算法。文中采用二维栅格空间来模拟仿真实现A*算法。

A*算法的估价算式为$f(n)=g(n)+h(n)$,其中

收稿日期:2012-03-03;**修回日期**:2012-07-02
基金项目:国家自然科学基金(61005008,60803049)
作者简介:刘 钰(1980-),女,江苏南京人,讲师,硕士,研究方向为人工智能与模式识别;陆建峰,教授,博士,博士生导师,研究方向为人工智能与模式识别。

$f(n)$ 是节点 n 的估价函数,$g(n)$ 是在状态空间中从初始节点到 n 节点的实际代价,$h(n)$ 是从 n 到目标节点最佳路径的估计代价。显然 $g(n)$ 是实际存在,确定无疑的,而 $h(n)$ 则只是一个估计函数,也称为启发函数[7]。

同时,A * 算法主要用 2 个队列 OPEN 队列和 CLOSED 队列来存储相关节点信息[8]。OPEN 队列主要保存已经生成而未被访问的节点信息,而 CLOSED 队列主要保存已被访问过的节点信息。通过遍历当前节点周围 8 个方向的子节点,可以选取生成节点,生成节点应该要通过以下规则选取:

(1)若子节点是障碍点,则忽略该节点。

(2)若子节点已经在 OPEN 队列,则需进行判断,如果通过当前节点(A * 算法选择当前节点的原则是从 OPEN 队列中选取估值函数值最小的节点)到达子节点的路径代价要小于通过子节点的父节点到达子节点的路径代价,则将当前节点设置为该子节点的父节点,并且修改该子 节点的 g 和 f 值,否则忽略该子节点[7]。

(3)若非上述情况,则找到了一个新的生成节点,插入 OPEN 队列即可。

本规则可用表 1 表示:

表 1　某情况下的生成节点示意

NW (都不在)	N (已在 CLOSED)	NE (g=14)(已在 OPEN)
W (都不在)	当前节点 (g=10)	E (障碍点)
SW (都不在)	S (g=9)(已在 OPEN)	SE (障碍点)

假设在遍历当前节点的子节点时出现表 1 情况。NW、W、SW 符合第(3)种情况,是新的生成节点,插入 OPEN 队列。N 在 CLOSED 中,E、SE 是障碍点,可以忽略。NE,S 虽然都已经在 OPEN 中,但是 NE 新的 g 值约为 11.4 < 原来的 g 值 14,所以通过当前节点到达 NE 子节点可能获得较短路径,所以将 NE 子节点的父节点改为当前节点,并且修改 NE 的 g 和 f 值。对于 S,计算其新 g 值为 11 > 原来的 g 值 9,所以忽略该节点。每个子节点的 g 值计算是不一样的,对于对角线上的子节点,其 g 值应该为当前节点的 g 值 + 根号 2,对于非对角线上的子节点,其 g 值应该为当前节点的 g 值加 1。

2　基于头尾双向搜索 A * 算法原理

以往的 A * 算法只是从起始节点出发,开始探索周围节点,并逐步延伸,直到将目标节点加入 OPEN 队列[9,10]。文中考虑同时从起始节点和目标节点探索,当在某块区域有相同的延伸子节点时,结束算法。改进算法需要建立 4 个队列,分别是 SOPEN、EOPEN、SCLOSED 和 ECLOSED,需要 2 个当前节点,分别是 S_Current 和 E_Current,其中,S_Current 表示从起始节点出发的当前节点,E_Current 表示从目标节点出发的当前节点。以往的算法在计算子节点的 h 值时,采用的是子节点和目标节点的距离[11],文中拟采用子节点和另外一个方向上的当前节点的距离作为该子节点的 h 值。比如,若要计算从起始节点延伸出的子节点的 h 值,就要计算该子节点和由目标节点出发的当前节点的距离。以期望从两个方向上出发的子节点尽快相遇,减少扩展节点的规模。

3　基于头尾双向搜索 A * 算法的路径规划算法设计

建立 SOPEN 队列、SCLOSED 队列、EOPEN 队列和 ECLOSED 队列,并将 OPEN 队列和 CLOSED 队列设计为矩阵形式,矩阵的每一行存储一个节点的基本信息。SOPEN 和 EOPEN 队列的数据结构是:1/0 | xval | yval | parnet xval | parent yval | g | h | f |,其中第一个元素标记该节点是否已经在相应的 CLOSED 队列中,1 代表不在,0 代表在,xval 代表该节点的 x 坐标,yval 代表该节点的 y 坐标,parnet xval 代表该节点父节点的 x 坐标,parnet yval 代表该节点父节点的 y 坐标,g、h、f 值分别代表了该节点和起始节点的路径消耗,和另一方向当前节点的路径消耗,$f = g + h$。在 OPEN 队列中设置标志位是为了避免大量数据的移动。

假设当前节点的坐标为(xCurrent,yCurrent),目标节点的坐标为(xTarget,yTarget),则以往当前节点的 h 值的计算公式为:

$$h = \text{abs}(\text{xCurrent} - \text{xTarget}) + \text{abs}(\text{yCurrent} - \text{yTarget}) \quad (1)$$

假设从起始节点出发的当前节点的坐标为(S_xCurrent,S_yCurrent),从目标节点出发的当前节点的坐标为(E_xCurrent,E_yCurrent),则计算从起始节点出发的当前节点的 h 值的计算公式为:

$$h = \text{abs}(\text{S\_xCurrent} - \text{E\_xCurrent}) + \text{abs}(\text{S\_yCurrent} - \text{E\_yCurrent}) \quad (2)$$

SCLOSED 队列和 ECLOSED 队列只需存储每个节点的坐标值,其数据结构为 xval | yval |。

采用基于栅格的路径规划算法仿真,以曼哈顿距离[12]作为估价函数。因为对每个节点要进行 8 个方向的遍历,所以用 ajda_array 来存储每个节点周围节点的信息,其数据结构为:xval | yval | g | h | f |。由于从 2 个方向进行探索,故采取了 Sadja_array 和 Eadja_array 对各方向上的扩展子节点进行处理。

基于头尾搜索 A * 算法可以使系统很快结束路径

规划，并找到一条理想的路径，如果当从起始节点开始的扩展子节点和从目标节点开始的扩展子节点是同一个节点的话，算法就应该终止，该节点的个数可能大于 1。取相同的第一个节点作为终止节点，并将该终止节点分别添加到 SOPEN、EOPEN、SCLOSED 和 ECLOSED 中。

4 基于头尾双向搜索 A * 算法实现

A * 路径规划仿真算法具体由 5 大模块组成，分别是主模块 main 和四个子模块（邻接点选取模块 adjacent_array、最小 F 值选取模块 min_open、节点索引返回模块 node_index 和 open 队列插入模块 open_insert）。每个子模块都可以封装为一个函数供主模块调用。

现将每个子模块做简单的介绍：

（1）邻接点选取模块 adjacent_array。

本模块用于返回当前节点匹配邻接点的信息，按照东北、正东、东南、正北、正南、西北、正西、西南八个方向遍历邻接点，若该邻接点是障碍点，则忽略该节点，若该邻接点已经在 CLOSED 中，则忽略该节点，若该邻接点不可以忽略，将该节点添加到 adja_array 中，并给该邻接点赋 g，h，f 值。

本模块函数可如下表示：

```
Function adja_array = adjacent_array(xCurrent,yCurrent,gvalue,xTarget,yTarget,CLOSED,MAP,MAX_X,MAX_Y)
```

（2）最小 F 值选取模块 min_open。

本模块用于返回 OPEN 队列中 F 值最小节点的索引值，也用于寻找目标节点。

本模块函数可如下表示：

```
function min_index = min_open(OPEN,OPEN_NUM,xTarget,yTarget)
```

（3）节点索引返回模块 node_index。

本模块用于返回坐标值为 xval，yval 节点在 OPEN 中的索引值。

本模块函数可如下表示：

```
function node_index = node_index(OPEN,xval,yval)
```

本模块的代码如下表示：

```
i=1;
while(OPEN(i,2) ~= xval || OPEN(i,3) ~= yval )
i=i+1; % 检测 OPEN 队列，用于索引值的返回
node_index=i;
```

（4）open 队列插入模块 open_insert。

本模块用于将当前节点插入 OPEN 队列中，插入节点的 x 坐标，y 坐标，父节点的 x 坐标，父节点的 y 坐标，以及该节点的 g，h，f 值。

本模块的函数可以如下表示：

```
Function new_row=open_insert(xval,yval,parent_xval,parent_yval,gvalue,hvalue,fvalue)
```

本模块的代码如下所示：

```
new_row=[1,8];定义一个 1 维数组，有 8 个元素
new_row(1,1)=1;是否在 CLOSED 队列，标志置 1，表示不在
new_row(1,2)=xval;当前节点的 x 坐标
new_row(1,3)=yval;当前节点的 y 坐标
new_row(1,4)=parent_xval;当前节点父节点的 x 坐标
new_row(1,5)=parent_yval;当前节点父节点的 y 坐标
new_row(1,6)=gvalue;当前节点的 g 值
new_row(1,7)=hvalue;当前节点的 h 值
new_row(1,8)=fvalue;当前节点的 f 值
```

5 仿真实验和对比分析

图 1 和图 2 分别为改进前后的算法路径图。分别对改进前和改进后的 A * 算法进行仿真，通过算法运行速度和扩展节点规模大小来评估 A * 算法的性能。仿真结果见表 2，其中 SPEED 表示核心算法运行的时间（单位是秒），EXP_NODE 代表扩展节点的个数。

表 2 算法性能分析表

编号	改进前的 A * 算法		改进后的 A * 算法	
	SPEED	EXP_NODE	SPEED	EXP_NODE
1	18.22	1527	2.38	653
2	10.21	765	1.16	371
3	4.31	1548	3.52	1087
4	0.92	336	0.73	221

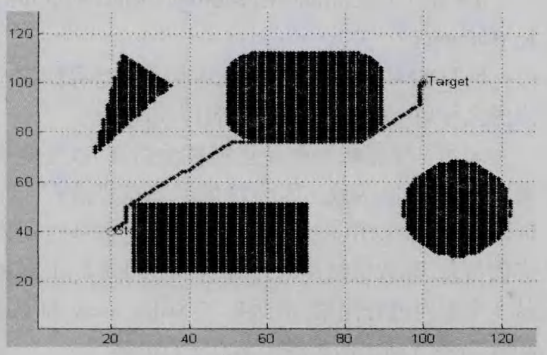

图 1 改进前算法路径图

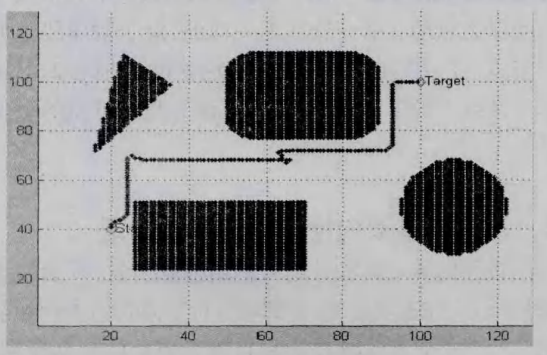

图 2 改进后算法路径图

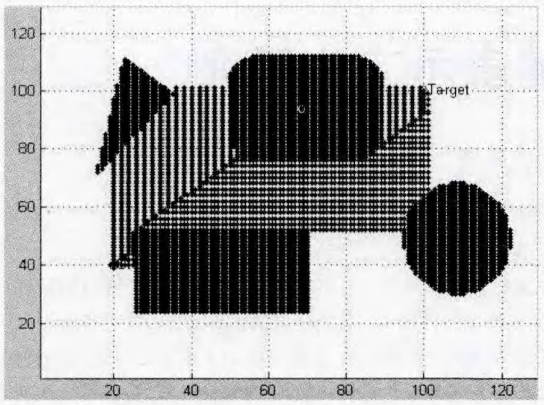

图 3　改进前算法扩散图

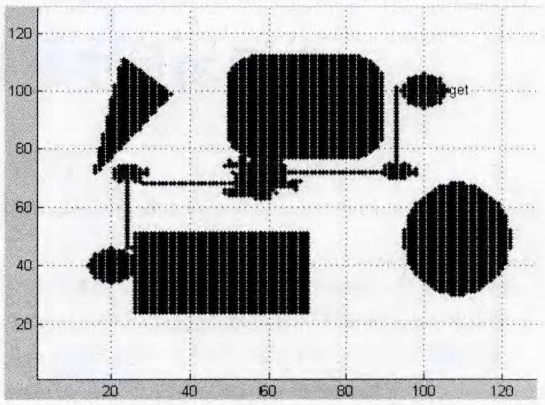

图 4　改进后算法扩散图

通过分析表 2 的数据可以发现,改进后的 A * 算法在速度上要快于原 A * 算法,扩展节点个数少于原 A * 算法。通过比较图 3 和图 4 发现,改进后的算法的扩散区域较改进前明显减少,结合核心算法的速度来看,改进后算法的效率更高。当区域环境越复杂,改进后算法的性能更加优越。

6　结束语

文中提出一种基于头尾双向搜索的改进 A * 算法进行智能机器人的路径规划控制。采取曼哈顿估值函数,从两个方面对改进的 A * 算法进行性能评估。通过对同一张模拟图进行多次仿真的试验结果分析表明,基于改进后的 A * 算法路径规划系统运行稳定,生成路径速度加快,搜索区域明显减少,各性能指标均优于普通 A * 算法。

参考文献:

[1] Stentz A. A real-time resolution optimal re-planning for globally constraint problem[C]//The 18th National Conf on Artificial Intelligence. Cambridge, MA: MIT Press, 2002: 1088-1096.

[2] Likhachev M. Search-based Planning for Large Dynamic Environments[D]. Pennsylvania: CMU, 2004.

[3] Tovey C, Greenberg S, Koenig S. Improved analysis of A * [C]//Proceedings of the International Conference on Robotics & Automation. Taiwan: [s. n.], 2003: 14-19.

[4] Koenig S, Likhachev M. A * Lite[C]//Proceedings of the AAAI Conference of Artificial Intelligence. [s. l.]: [s. n.], 2002: 476-483.

[5] Whangbo Taeg-Keun. Efficient Modified Bidirectional A * Algorithm for Optimal Route-finding[C]//IEA/AIE 2007. [s. l.]: [s. n.], 2007: 344-353.

[6] Koenig S, Likhachev M, Furcy D. Lifelong planning A * [J]. Artificial Intelligence Journal, 2004, 155(1-2): 93-146.

[7] van den Berg J, Overmars M. Roadmap-based Motion Planning in Dynamic Environments[J]. IEEE Transactions on Robotics, 2005, 21(5): 885-897.

[8] Gelperin D. On the Optimality of A * Artificial Intelligence [J]. IEEE Transactions on Robotics, 2000, 8(1): 60-90.

[9] Han K H, Park K H, Lee C H, et al. Parallel quantum-inspired genetic algorithm for combinatorial optimization problem[C]//Proceedings of the 2001 Congress on Evolutionary Computation. USA: IEEE Press, 2001: 1422-1429.

[10] 戴　博,肖晓明,蔡自兴. 移动机器人路径规划技术的研究现状与展望[J]. 控制工程,2005,12(3):198-202.

[11] 马雪英,何臻峰,林兰芬. 人工智能技术在机器人运动规划中的应用[J]. 计算机应用研究,2004(4):56-61.

[12] Han Kuk-Hyun, Kim Jong-Hwan. Genetic quantum algorithm and its application to combinatorial optimization problem [C]//IEEE Proceedings of the 2000 Congress on Evolutionary Computation. San Diego, USA: IEEE Press, 2000: 1354-1360.

(上接第 107 页)

社,1998.

[5] Barnsley M F, Sloan A D. A Better Way to Compress Images [J]. BYTE, 1988, 13(1): 215-223.

[6] Jacobs E W, Fisher Y, Boss R D. Image compression: a study of the iterated transform method[J]. Signal Processing, 1992, 29(3): 251-263.

[7] 何传江,杨　静. 基于形态特征的快速分形图像编码[J]. 中国图象图形学报,2005,10(4):410-414.

[8] Fisher Y. Fractal image compress: theory and application of fractals[M]. [s. l.]: [s. n.], 1995: 49-51.

[9] Jacquin A E. A Novel Fractal Block-coding Technique for Digital Image[C]//Proceedings of ICASSP. [s. l.]: [s. n.], 1990: 2225-2228.

[10] 陈衍仪. 图像压缩的分形理论和方法[M]. 北京:国防工业出版社,1997.

[11] 何传江,刘维胜,申小娜. 基于行列式的快速分形图像编码算法[J]. 中国图象图形学报,2008,13(3):435-439.

[12] 何传江,蒋海军,黄席樾. 基于平均偏差排序的快速分形图像编码[J]. 中国图象图形学报,2004,9(9):1130-1134.