

双向 ACO 算法应用于静态机器人全局路径规划研究

徐义晗, 程 乐, 宋艳红, 郭艾华

(淮安信息职业技术学院 计算机科学与工程系, 江苏 淮安 223003)

摘 要: 提出双向蚁群算法并应用于静态环境下的机器人全局路径规划问题, 对栅格法环境建模进行改进, 将传统的栅格法改进为膨胀栅格法; 使用双向蚁群算法在出发点和目标点设置带有不同标记的两族蚂蚁相向爬行完成搜索, 启发信息主要通过目标点、出发点和蚂蚁的当前位置二维坐标值计算得出; 信息素存储采用方向信息素矩阵。仿真实验证明: 即使在障碍物非常复杂的地理环境, 用本算法也能迅速规划出最优路径。

关键词: 蚁群算法; 路径规划; 信息素矩阵; 膨胀栅格法

中图分类号: TP301

文献标识码: A

文章编号: 1000-7180(2012)01-0097-05

Ant Colony Algorithm for Mobile Robot Path Planning

XU Yi-han, CHENG Le, SONG Yan-hong, GUO Ai-hua

(Department of Computer Science and Engineering, Huaian College of Information Technology, Huaian 223003, China)

Abstract: In this paper, we proposed a Two Directions ant colony algorithm and apply it to robot path planning in static environment. We changed grid method into expanded grid method for the environmental models. The optimal path search is finished by two tribes' ants opposite crawling from starting point and ending point. Heuristic information is calculated by the coordinate of Starting point, ending point and the position of ant. Pheromone is reserved by direction pheromone matrix. Results of simulation experiments demonstrate that even in obstacles very complicated geographical environment the best path can be found in short time.

Key words: ant colony algorithm; path planning; pheromone matrix; expanded grid method

1 引言

全局路径规划是指: 机器人在已知整个环境障碍物信息的前提下, 通过自主思考与计算规划出一条安全的、最优或相对最优的路径。这里的安全主要指不与边界或障碍物发生碰撞, 最优路径则主要指路径长度最短。目前, 已经有学者将 Dijkstra、GA^[1-2] (Genetic Algorithm, GA) 等算法应用于全局机器人路径规划问题, 取得了不错的效果。

蚁群算法 (Ant Colony Algorithm, ACO)^[3] 由意大利学者 Marco Dorigo 于 1992 年提出的一种优化算法。ACO 算法的主要模拟了蚁群的集体觅食行为, 寻优机制包括启发信息、信息素、正反馈、轮盘赌、禁忌表等等。算法设计之初被视为是一种在图中

寻找优化路径的机率型技术, 用来解决 TSP (Traveling Salesman Problem) 问题。随着 ACO 算法研究的深入该算法已经被应用到许多科研与工程领域的实际问题, 例如: 工作调度问题、网络设计问题等^[4-6]。合理的环境表示是实现机器人路径规划的基础, 也是选择合适算法的前提。栅格法是机器人路径规划领域中常用的一种方法。该方法具有信息量小、思想简单、编程容易实现、所规划处的路径信息容易表示等特点。但是, 其最终规划出的路径为折线而非曲线; 同时, 栅格密度越高也就意味着路径越接近曲线。所以, 能求解复杂的、高栅格密度的路径规划问题也就成为该领域研究的重点^[7]。

本文首先将传统的栅格法环境进行改进为膨胀栅格法, 该方法简化了传统栅格法对环境边界的判

收稿日期: 2011-06-04; **修回日期:** 2011-07-09

基金项目: 国家自然科学基金项目 (60673102); 江苏省自然科学基金项目 (BK2006218)

断过程,重点是简化程序编写,提高了算法执行效率.在此基础上,本文提出了一种基于膨胀栅格法双向蚁群算法(Two Directions ACO, TDACO). TDACO 算法在机器人行走的出发点和目标点各设置一群蚂蚁,蚂蚁在启发信息和信息素综合作用下相向爬行完成搜索.蚂蚁个体增加了感知和识别异群蚂蚁的功能. TDACO 算法可以在彻底避免回路的基础上提高了算法搜索最优路径的成功率;信息素存储采用方向信息素矩阵,方向信息素矩阵的引

入使算法的信息素浓度存储计算更加合理,查找和更新信息素速度更快,避免同群蚂蚁在同一路径过于拥挤,增加了搜索路径的多样性.

2 膨胀栅格法环境建模

2.1 传统栅格法环境建模

栅格法^[7]主要是将环境分解成一系列具有二值信息的网格单元,如图 1 所示.

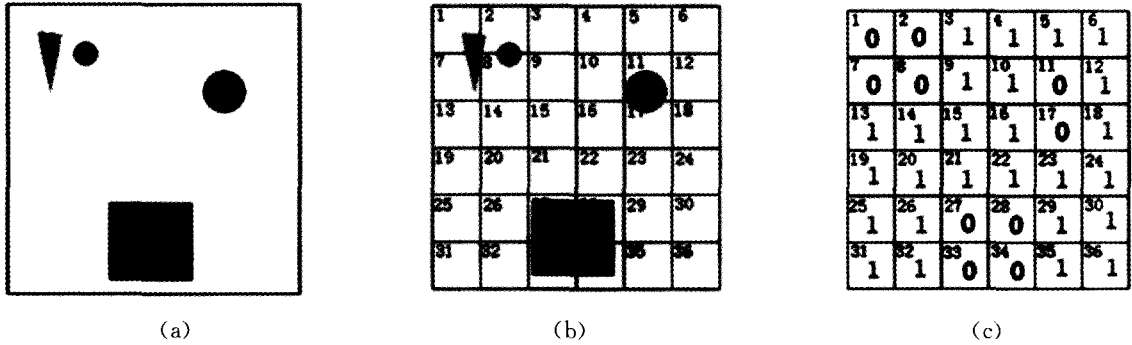


图 1 栅格法环境建模

图 1(a)表示具有障碍物的环境区域(R);图 1 (b)表示用栅格法对环境建模,其中栅格被从左至右,从上至下的连续标号标记 $R(r_1, r_2 \cdots r_{36})$;图 1 (c)表示建模后生成的机器人可存储的环境信息. 由图 1 可以看出,栅格法最终得到的环境信息是一个只包含“0”和“1”的二维矩阵. 其中“0”代表该处栅格为不可行,即该处栅格涉及到障碍物;“1”代表该处栅格为可行. 为了便于计算蚂蚁在栅格间爬行的距离和方向,每个栅格还应具有一个二维空间坐标 (x, y) ,设栅格标号用 τ 表示,每行(列)的栅格数用 μ 表示. 则由栅格标号到栅格坐标的转换公式如下:

$$x_{\tau} = ((\tau - 1) \bmod \mu) + 1 \quad (1)$$

$$y_{\tau} = (\text{int})((\tau - 1) / \mu) + 1 \quad (2)$$

int 为取整运算,mod 为求余运算. 栅格法建模后,栅格的长宽并没有实际意义,最后评价路径的长度主要和最终规划路径上的“垂直”和“斜向”栅格数量有关. 垂直可设距离为 1,斜向可设距离为 1.414. 栅格法具有思想简单,存储信息量小、无需对障碍物膨胀等特点,因此被广泛使用于机器人路径规划研究.

2.2 膨胀栅格法环境建模

传统的栅格法环境建模存在对边界判断复杂的缺欠. 如 2.1 节所述,栅格法最终存入机器人将是一组栅格标号,可用一个简单的一维数组存放. 在实际

应用时可以使用公式(1)、(2)对一维的栅格标号进行计算,将栅格标号转换为二值坐标. 但是无论是一维标号还是二值坐标都给判断环境边界带来困难. 在机器人路径规划领域中与边界发生碰撞被认为是不安全的. 传统的栅格法是利用栅格的一维标号通过复杂的转换和定义禁忌集合等方式来保证机器人不与边界发生碰撞. 本文提出一种膨胀栅格法可以巧妙地解决此问题. 如图 2 所示.

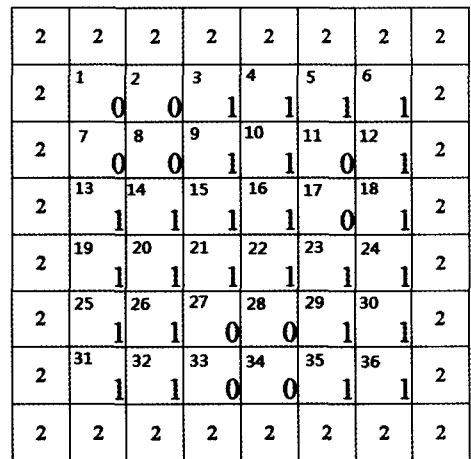


图 2 膨胀栅格法环境建模

图 2 是将图 1(c)进行膨胀. 膨胀后环境信息由原来的 6×6 栅格增加到 8×8 栅格,存储信息稍有增加,但是判断环境边界却变得极为简单. 只要搜索

到栅格中的存放值为“2”即可判断为环境边界。

3 双向蚁群算法

TDACO 算法是在 ACO 算法基础上加以改进得到的新算法。TDACO 算法采用的是两群蚂蚁在启发信息和信息素的综合作用下分别从出发点和目标点开始搜索,蚂蚁具有识别相遇蚂蚁是否来至本群的功能,如果其中一个蚂蚁感知到来至另外一个群的蚂蚁,则可建立一条路径。区别于传统 ACO 算法,TDACO 算法的信息素不是用于正反馈,而是为了避免同族蚂蚁在一条路径上过于拥挤,增加蚂蚁搜索路径的多样性。

3.1 相关定义

定义 1 TDACO 算法把机器人出发点栅格号 R_{begin} 和机器人目标点栅格号 R_{end} 作为两个种族蚂蚁的巢穴。例如机器人从 r_3 行走到 r_{36} 则 $R_{begin} = r_3$; $R_{end} = r_{36}$ 。

定义 2 为了蚂蚁能够识别被感知蚂蚁是否来至双向,每个蚂蚁都设有种群标记 Ant_{flag} , TDACO 中约定来至 R_{begin} 巢穴的蚂蚁 $Ant_{flag} = 0$, 来至 R_{end} 巢穴的蚂蚁 $Ant_{flag} = 1$ 。

定义 3 感知域(Sense Region, SR),假设蚂蚁爬行到某一栅格 r_i , 则包围 r_i 的临近可行栅格(图 1(c)中标记为“1”的栅格)组成 SR。如图 1 中 r_9 的感知域为: $SR_9 = \{r_3, r_4, r_{10}, r_{14}, r_{15}, r_{16}\}$ 。

定义 4 方向信息素矩阵(Phoneme Matrix, PM)是一个二维矩阵,记录链接 r_i, r_j 路径被蚂蚁爬过的次数(浓度)。 $PM(i, j)$ 表示从 r_i 到 r_j 路径上的信息素浓度,而 $PM(j, i)$ 则表示从 r_j 到 r_i 路径上的信息素浓度。PM 记录的值与蚂蚁爬行次数和方向有关,当 $PM(i, j)$ 值超过预先设定浓度最大值(PM_{max})时, $PM(i, j)$ 被清零($PM(i, j) = 0$)。

定义 5 禁忌表 $tabu$, 由于记录蚂蚁已爬行经过的栅格标号,例如,图 1 中从 r_3 到 r_{36} 可以在禁忌表中记录为 $tabu = \{r_3, r_{20}, r_{16}, r_{22}, r_{23}, r_{30}, r_{36}\}$ 。 $Tabu_{max}$ 表示禁忌表中可存放的最大栅格数量,即一只蚂蚁可行走的最多栅格数。TDACO 算法对 $Tabu$ 有如下约定:

约定 1 如果 $Ant_{flag} = 0$, 则 $tabu[1] = R_{begin}$; 如果 $Ant_{flag} = 1$, 则 $tabu[1] = R_{end}$ 。

约定 2 $tabu$ 中记录的栅格标号不重复。

约定 3 如果栅格数等于 $Tabu_{max}$ 则清空禁忌表,并按约定 1 完成 $tabu[1]$ 的赋值。

3.2 蚂蚁搜索

本节用一个类来描述 TDACO 算法中的蚂蚁个体,类中相关的成员与方法构成了 TDACO 搜索最优路径所需的元素。结合相关定义,描述如下:

```
Class Ant
{
    int tabu[]; // 禁忌表,最多存储栅格数为  $Tabu_{max}$ 
    int R_end; // 出发点栅格号
    int R_begin; // 目标点栅格号
    int Ant_flag; // 种族标记,取值为 0 或 1
    int position; // 记录当前蚂蚁当前位置,值为一栅格标号
    int nextPosition; // 记录由 NextStep() 计算得出的下一步蚂蚁行进的栅格号
    int NextStep(); // 计算得到下一个预行进的栅格
}
```

在栅格法建模的环境中,最重要的问题是蚂蚁在当前栅格(position)处,如何选择下一个栅格(nextPosition)。TDACO 算法通过启发信息和信息素共同作用实现 nextPosition 计算。

(1) 启发信息计算

TDACO 中启发信息主要是蚂蚁与预行进终点的距离,如 $Ant_{flag} = 0$ 则行进终点为 R_{end} ; $Ant_{flag} = 1$, 则预行进终点为 R_{begin} 。

以 $Ant_{flag} = 0$ 为例,在栅格总数为 n 的环境中,栅格 $r_i (i = 1, 2, \dots, n)$ 为蚂蚁当前感知域 SR 中的任意一个栅格,根据公式(1)、(2)将 r_i 和 R_{end} 转换成二维空间坐标 (x_{ri}, y_{ri}) 和 (x_{Rend}, y_{Rend}) 。

启发信息 D 计算公式为:

$$D = \sqrt{(x_{ri} - x_{Rend})^2 + (y_{ri} - y_{Rend})^2} \quad (3)$$

(2) 信息素存储与查找

TDACO 中信息素主要记录某一方向上,某条路径上蚂蚁爬行过的次数。因此需要建立一个方向信息素矩阵 PM 来存放某条路径上的信息素浓度。如定义 4 所述,蚂蚁从 Position 栅格选择 nextPosition 栅格爬行:

当 $PM(posion, nextPosion) < PM_{max}$ 时, PM 的计算公式如下:

$$PM(posion, nextPosion) = PM(posion, nextPosion) + 1 \quad (4)$$

当 $PM(posion, nextPosion) \geq PM_{max}$ 时, PM 的计算公式如下:

$$PM(posion, nextPosion) = 0 \quad (5)$$

在实际算法设计中 PM 可用一个二维数组实现。查找和更新 PM 是 TDACO 中最频繁的操作。

如果栅格总数为 n , 则 PM 的密度应为 n^2 因此用遍历的方法访问 PM 将消耗大量时间. 但是 TDACO 中的算法机制使得每次访问 PM 都无需遍历整个数组, 只需通过当前的栅格号与 SR 中的栅格号来访问 PM.

(3) 蚂蚁选择下一个节点策略

蚂蚁具体选择那一个栅格作为自己的 next-Position, 主要通过 NextStep() 方法实现. 该方法主要运用以下公式计算:

$$\begin{cases} \max\{(1/D)^\alpha (1/PM)^\beta\}_{\text{nextPosition} \notin \text{tabu}} & \text{If } q \leq q_0 \\ \text{rand_SR}() & \text{else} \end{cases} \quad (6)$$

式中, $\text{nextPosition} \notin \text{tabu}$ 表示 nextPosition 在感知域中, 且不在禁忌表中; $\text{rand_SR}()$ 表示随机得到一个不属于 tabu 的栅格号; 为了增加搜索多样性, TDACO 算法设定阈值 q_0 , 当随机数 $q \leq q_0$ 时根据公式 (6) 中 $\max\{(1/D)^\alpha (1/PM)^\beta\}$ 计算 nextPosition; 当 $q > q_0$ 时, 则利用 $\text{rand_SR}()$ 在 SR 中随机选择一个栅格作为 nextPosition. $\max\{(1/D)^\alpha (1/PM)^\beta\}$ 体现了算法倾向于查找 SR 中启发信息值尽量最小、信息素浓度尽量低的栅格作为 nextPosition, α, β 为重要性比重因子, 通过 α, β 调节启发信息和信息素浓度的重要性比重.

3.3 蚂蚁感知功能

TDACO 中最终通过蚂蚁感知功能建立机器人可行路径, 感知功能可以简单理解为: 算法中初始化了 m 只蚂蚁 (Ant), 当某只 $\text{Ant}_i (i=1, 2, \dots, m)$ 爬行到某个栅格 posion 位置后, 那么 Ant_i 能够感知到 SR 中的来至双向的蚂蚁. 判断蚂蚁是否来至双向主要通过 Ant_{flag} 判断. 当蚂蚁感知到来至双向的蚂蚁时, 立即建立联通路径 (Rode), 并与当前算法搜索到的当前最优路径 (Rode*) 比较, 比较后及时更新 Rode*. 如图 3 所示.

图 3 展示了在一个 6×6 栅格密度的环境中, 某一族蚂蚁感知到另外一族蚂蚁并建立联通路径的过程, 其中栅格 16 和栅格 20 为不可行栅格. 由图 3 可知: 无论栅格密度多大, 感知域的栅格数量永远小于等于 8 个. 并且感知域可通过公式表示如下:

设环境栅格长宽栅格数量均为 l , 蚂蚁当前位置为 Posion, 则集合 A 为: $A = \{1, -1, l, l+1, l-1, -l, 1-l, -(l+1)\}$, 集合 B 为 A 中不可行栅格 (栅格内存储信息为 0) 和环境边界栅格 (栅格内存储信息为 2).

则感知域的栅格公式为:

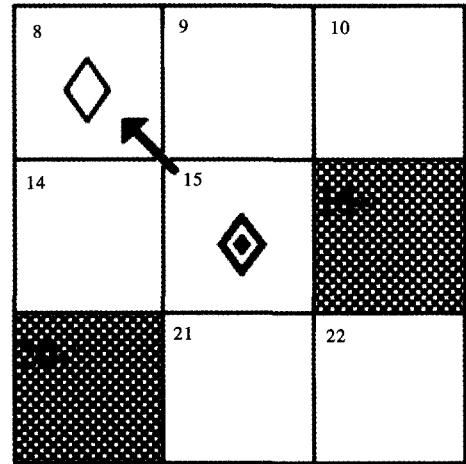


图 3 TDACO 的蚂蚁感知功能

$$\text{Posion} = \text{Posion} + \delta$$

$$\delta \in Z, Z = A - B \quad (7)$$

3.4 算法步骤

Step1: 初始化算法迭代次数 N ;

初始化蚁群规模为 m , 并将 $m/2$ 只蚂蚁的 $\text{Antflag} = 0$, 另外 $M/2$ 只 $\text{Antflag} = 1$;

初始化浓度最大值 (PM_{max});

初始化禁忌表中可存放最大栅格数 Tabu_{max} ;

初始化以及重要性比重因子 α, β ;

初始化 Road* (存放算法查找到的最优路径) 为一较大的值.

Step2: 根据公式 (6) 两族中所有蚂蚁爬行一步;

Step3: 根据定义 5 更新 tabu , 根据公式 (4)、(5) 更新 PM;

Step4: 根据公式 (7), 针对 $\text{Ant}_{\text{flag}} = 0$ 中 $m/2$ 只蚂蚁进行感知域搜索, 若搜索到来至 $\text{Ant}_{\text{flag}} = 1$ 族的蚂蚁, 则立刻建立联通路径, 计算路径长度; 若由于当前最优路径则更新 Road*.

Step5: 若总迭代次数等于 N , 则算法结束, 否则返回 Step2;

4 仿真实验

本文进行了大量仿真实验对 TDACO 算法进行性能测试, 实验仿真软件 VS2005 (Visual Studio 2005), 使用编程语言为 C Sharp. 在栅格密度为 60×60 复杂环境下对 TDACO 性能进行测试, 并与基本的 ACO 算法进行比较. 并用下式计算方差:

$$\phi = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (L_i - L)^2} \quad (8)$$

其中, N 是试验次数, L_i 是第 i 次得到的路径长度, L

是 N 次试验得到的平均长度. 取 $N=10$, 进行十组实验, 每组实验障碍物设置均不相同, 但都有一条以上的可行路径. 算法参数设置如下: $m=50$, $\alpha=1$, $\beta=2$, $Tabu_{max}=60$; $Road^*=90$. 实验结果如图 4 所示.

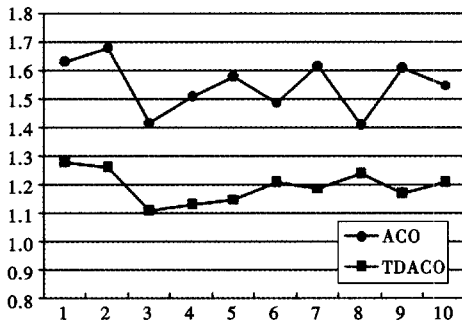


图 4 TDACO 与 ACO 的比较实验

由图 4 可知 TDACO 较基本的 ACO 算法稳定性提高明显, 求解更接近于最优解. TDACO 之所以高效主要是因为双向蚂蚁双向搜索、感知域和方向信息素矩阵的有机结合. 双向搜索和感知域的使用提高了搜索路径的速度和成功率, 而方向信息素矩阵则提高了搜索路径的多样性.

5 结束语

本文在蚁群算法基础上进行了搜索方式和信息素存储计算方面的改进, 得到了 TDACO 算法, 并改进传统的栅格法为膨胀栅格法对环境进行建模. 仿真试验证明, 改进后的算法搜索最优路径速度快, 成功率高. 各种机制的综合运用使 TDACO 算法运行的早期就能较快地收敛, 并使算法在收敛速度和执行效率整体上得到提高.

参考文献:

- [1] 孟偲, 王田苗. 一种移动机器人全局最优路径规划算法 [J]. 机器人, 2008, 30(3): 217-222.
Meng Cai, Wang Tianmiao. A global optimal path planning algorithm for mobile robot [J]. Robot, 2008, 30(3): 217-222.
- [2] Szczerba R J, Chen D Z, Uhran Jr J J. A grid based approach for finding conditional shortest paths in an unknown environment [EB/OL]. [2009-09-22]. www.cse.nd.edu/Reports/1994/tr-94-34.ps.
- [3] 孔令明, 谢圣献, 李俊青. 一种蚁群和信任结合的 P2P 节点选择机制 [J]. 微电子学与计算机, 2011, 28(3): 133-136.
- [4] 王海宁, 孙守迁. 基于优化成熟度的自适应蚁群优化算法 [J]. 微电子学与计算机, 2010, 27(8): 140-143.
- [5] Nilgün F, Celal O, Orhan E, et al. Investigation of ant system parameter interactions by using design of experiments for job-shop scheduling problems [J]. Computers and Industrial Engineering, 2009, 56(2): 538-559.
- [6] Jin Fei-hu, Gao Hui-jun, Zhong Xiao-jian. Research on path planning of robot using adaptive ant colony system [J]. Journal Of Harbin Institute Of Technology, 2010, 42(3): 1014-1017.
- [7] Elfes A. Using occupancy grids for mobile robot perception and navigation [J]. Computer, 1989, 22(6): 46-57.

作者简介:

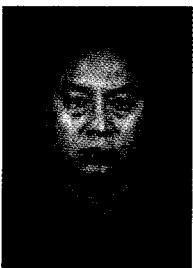
徐义晗 男, (1974—), 硕士, 讲师. 研究方向为智能控制.

程乐 男, (1979—), 双硕士, 讲师. 研究方向为人工智能与优化.

(上接第 96 页)

- [9] Mhatre V, Rosenberg C. Design guidelines for wireless sensor networks: Communication, clustering and aggregation [J]. Ad Hoc Networks, 2004, 2(1): 45-63.

作者简介:



潘中强 男(1978—), 讲师. 研究方向为计算机网络安全与网络管理.



刘亮亮 男(1983—), 助教. 研究方向为网络通信技术.



张东, 男(1988—). 研究方向为网络工程.