控制与决策
Control and Decision

基于改进蚁群算法的水面无人艇路径规划

孙功武， 苏义鑫， 顾轶超， 谢基榕， 王俊轩

引用本文:
孙功武, 苏义鑫, 顾轶超, 等. 基于改进蚁群算法的水面无人艇路径规划[J]. 控制与决策, 2021, 36(4): 847−856.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2019.0839

基于改进蚁群算法的水面无人艇路径规划

孙功武[1†], 苏义鑫[2], 顾轶超[1], 谢基榕[1], 王俊轩[1]

(1. 中国船舶科学研究中心 深海载人装备国家重点实验室，江苏 无锡 214082；
2. 武汉理工大学 自动化学院，武汉 430070)

摘 要: 针对水面无人艇路径规划问题,提出一种改进蚁群算法进行求解. 该算法建立作用时效不同的局部禁忌表和全局禁忌表,实现对蚂蚁途经栅格的分类存储,在蚂蚁发生障碍死锁和自死锁时分别采取不同的死锁处理策略,从而降低无效蚂蚁产生的概率,提高解的多样性;引入当前蚂蚁所处栅格与终点栅格之间的欧式距离,设计自适应启发函数,以避免蚂蚁路径搜索的初期盲目性与后期单一性;适时采用历史最优路径替换本轮迭代中的最差路径,保证已搜索到的最优路径不会丢失. 在不同规模、不同复杂度地图中的仿真结果表明,所提出改进算法能够大幅度提高搜索过程中有效蚂蚁的数量,其收敛速度与精度两方面性能均优于未改进算法. 在规模较大、复杂度较高的地图中,更能体现应用改进算法的优越性.

关键词: 水面无人艇；路径规划；蚁群算法；死锁；自适应启发函数；最优路径

中图分类号: TP242.2 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2019.0839 **开放科学(资源服务)标识码(OSID):**

引用格式: 孙功武,苏义鑫,顾轶超,等. 基于改进蚁群算法的水面无人艇路径规划[J]. 控制与决策, 2021, 36(4): 847-856.

Path planning for unmanned surface vehicle based on improved ant colony algorithm

SUN Gong-wu[1†], *SU Yi-xin*[2], *GU Yi-chao*[1], *XIE Ji-rong*[1], *WANG Jun-xuan*[1]

(1. State Key Laboratory of Deep-sea Manned Vehicles，China Ship Scientific Research Center，Wuxi 214082，China；
2. School of Automation，Wuhan University of Technology，Wuhan 430070，China)

Abstract: An improved ant colony algorithm for USV (unmanned surface vehicle) path planning is presented. A local tabu list and a global tabu list with different effect duration are set up to catalog the ants passed grids.Various deadlock processing strategies are introduced upon barrier deadlock and self deadlock situation in order to reduce the number of invalid ants and to improve the diversity of solutions. An adaptive heuristic function is designed by adopting the Euclidean distance between the ant and the destination to avoid the initial blindness and later singleness of ant path searching. The current worst path would be superseded by the historical best path when appropriate to retain the previous effort. Simulation results under different maps show that the improved algorithm considerably increases the number of effecive ants during the searhing process and the probability to find the optimal path, as well as the search speed. The improved algorithm performs even better in larger and more complex grid maps.

Keywords: unmanned surface vehicle；path planning；ant colony algorithm；deadlock；adaptive heuristic function；optimal path

0 引 言

路径规划关系到水面无人艇(unmanned surface vehicle, USV)的任务执行效率和航行安全,是USV的关键技术之一. 传统的路径规划方法主要有人工势场法[1]、Dijkstra算法[2]、A* 算法[3-4]等. 路径规划属于NP-hard问题,因此很多学者采用遗传算法[5-6]、粒子群算法[7]、蚁群算法[8]等智能仿生算法进行求解. 其中,蚁群算法是一种模拟生物群体觅食的正反馈、启发式随机搜索算法,具有鲁棒性好、全局搜索能力强、环境约束表达方便等优势.

文献[8]将蚁群算法应用于车辆路径规划中,并通过仿真验证了该方法的有效性. 为进一步提高蚁群算法的性能,文献[9]将改进烟花算法与蚁群算法相结合,通过烟花算法生成路径上的初始信息素;文

收稿日期: 2019-06-11；**修回日期:** 2019-10-09.
基金项目: 江苏省自然科学基金项目(K20170217)；七〇二所自主培育项目(51811001K1741SK).
†通讯作者. E-mail: sungongwu@126.com.

献[10]则采用粒子群算法对蚁群算法的参数进行优化,这两种方法分别能够使蚁群算法具有较好的信息素初始分布和参数配置,避免依靠人工经验选取算法参数的弊端,但也在一定程度上增加了路径规划过程的复杂度. 在栅格地图中,考虑到相邻栅格之间的距离无明显差异,采用其倒数作为启发函数对路径搜索的引导作用不大,因而文献[11]改用蚂蚁周围栅格到终点栅格欧式距离的倒数作为启发函数,使蚂蚁以更大的概率向终点移动,该方法在一定程度上避免了算法初期路径搜索的盲目性,但忽略了由当前栅格转移到下一栅格所付出的代价. 文献[12]采用的启发函数中则同时含有本次转移经过路径的距离以及转移后与终点栅格之间的距离,使算法的全局搜索能力更强,但该启发函数的作用强度受距离影响较大. 针对蚂蚁每次只能在相邻栅格之间转移,从而限制了算法收敛速度和精度的问题,文献[13]提出了一种转移不局限于周围栅格的多步长蚁群算法,该算法能够搜索出一条更短、转移步数更少的路径,但在大规模地图中的路径搜索过程较繁琐. 文献[8-13]中,蚁群算法的信息素存储在栅格间的连接路径上,当地图规模较大时,则需要占用大量内存空间来存储信息素分布矩阵. 因此,文献[14]针对栅格地图中路径规划的特点,提出了一种信息素存储于栅格中的蚁群算法,这种信息素存储方式在一定程度上减小了算法的复杂度. 将蚁群算法应用于栅格地图中,在进行路径规划时存在蚂蚁死锁问题,从而降低了算法中有效蚂蚁的数量,影响算法的搜索效率. 文献[15-16]对障碍死锁现象进行分析,并提出了数种解决措施. 其中,文献[15]在蚂蚁发生障碍死锁时,先对当前栅格的信息素进行惩罚,再令蚂蚁回退一步,该方法仅能降低蚂蚁在同一位置陷入死锁的概率. 文献[16]建立了死锁禁忌表,用于存储蚂蚁发生障碍死锁所处栅格,并禁止后续蚂蚁通行,从而彻底克服障碍死锁的问题. 除障碍死锁,蚂蚁仍可能被自身经过的路径锁死,发生自死锁,该类死锁现象在大规模地图中出现的概率较大,但目前尚未见研究此问题的相关文献.

本文采用信息素存储于栅格中的蚁群算法实施USV路径规划,对其中的死锁问题进行深入分析,并提出相应的死锁处理策略. 同时,在启发函数设计与最优路径处理两方面作出改进. 最后,通过仿真测试对改进算法的有效性进行验证.

1 问题描述

1.1 环境建模

栅格法建模是采用相同大小的栅格对平面环境进行划分,并对划分后的栅格进行统一编号形成栅格地图. 采用该方法建立规模为$M\times M(M=10)$的栅格地图如图1所示.

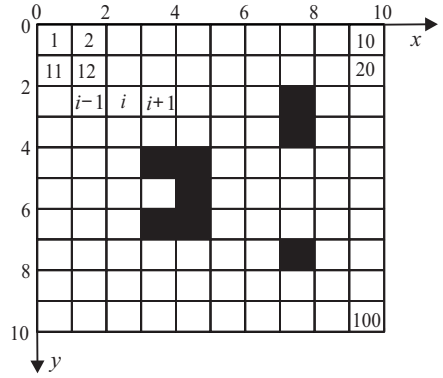


图1 栅格地图

图1中: 白色栅格表示允许通行的自由栅格;黑色栅格表示禁止通行的障碍栅格. 栅格通行属性用Map()函数表示为

$$\mathrm{Map}(i)=\begin{cases}1,\ \text{第}i\text{个栅格为障碍栅格};\\ 0,\ \text{otherwise}.\end{cases}\tag{1}$$

栅格i与其中心坐标(x_i,y_i)的对应关系为

$$\begin{cases}x_i=(i-1)\mathrm{mod}M+0.5,\\ y_i=\mathrm{int}((i-1)/M)+0.5.\end{cases}\tag{2}$$

其中: mod为取余运算; int为取整运算.

从USV的航行安全性方面考虑,对划分地图的栅格有最小尺寸要求. 在USV设计阶段,根据其水动力参数和本体航行运动控制策略,评估出最大航行工况(风、浪、流)下定点定位的控制性能.假设定位控制精度为A;根据USV的应用场合,选择合适的船舶碰撞领域模型(一般是以船舶为中心的椭圆)和模型尺寸,假设椭圆的长半轴为L. 结合USV的碰撞领域模型尺寸和环境影响下的定位精度,要求栅格尺寸应大于$2(L+A)$.

1.2 最优路径模型

在栅格地图中,假设USV位于所在栅格的中心,每次转移只选择其相邻的自由栅格作为目标. 当USV进行上、下、左、右4个方向进行转移时,转移前后所在的栅格应满足

$$\begin{cases}\mathrm{Map}(g)=0,\\ d_{gc}=\sqrt{(x_g-x_c)^2+(y_g-y_c)^2}=1.\end{cases}\tag{3}$$

其中: c为USV当前所在栅格, g为USV转移后所在栅格, d_{gc}为栅格g与c之间的距离.

由于USV并非质点,其向左上、左下、右上、右下栅格进行斜线转移时,还要求斜线两侧的栅格为自由栅格. 当USV向左上或者左下转移时,转移前后所在的栅格应满足

$$\begin{cases} \mathrm{Map}(g)=0, \\ d_{gc}=\sqrt{(x_g-x_c)^2+(y_g-y_c)^2}=\sqrt{2}, \\ (|g-c+M|)\mathrm{mod}M=1, \\ \mathrm{Map}(g-1)=0, \\ \mathrm{Map}(c+1)=0; \end{cases} \tag{4}$$

当USV向右上或者右下转移时,转移前后所在栅格应满足

$$\begin{cases} \mathrm{Map}(g)=0, \\ d_{gc}=\sqrt{(x_g-x_c)^2+(y_g-y_c)^2}=\sqrt{2}, \\ (|g-c+M|)\mathrm{mod}M=M-1, \\ \mathrm{Map}(g+1)=0, \\ \mathrm{Map}(c-1)=0. \end{cases} \tag{5}$$

式(3)～(5)为USV转移约束条件,是判断其相邻栅格是否为可行栅格的依据. 假设USV由起点栅格S,在满足转移约束条件下,经过N次转移后到达终点栅格E,则其经过的路径长度为

$$L=\sum_{i=1}^{N} d_{o_i o_{i+1}}. \tag{6}$$

其中: $\{o_1,o_2,\ldots,o_{N+1}\}$为组成路径的栅格集合,且$o_1=S, o_{N+1}=E$; L为路径长度.

USV有多条从起点到达终点的可行路径,路径规划即求解出一条最优或者次优的可行路径. 本文主要研究最短路径,其数学模型可表示为

$$L_{\mathrm{opt}}=\min(L). \tag{7}$$

其中min()表示取最小值运算.

1.3　信息素存储于栅格的蚁群算法

采用蚁群算法在栅格地图中进行路径规划与求解TSP问题的原理类似,但问题的约束条件与算法中蚂蚁路径搜索的终止条件有所不同,主要体现在:

1) 蚂蚁由起点转移到终点不必遍历地图中所有栅格;

2) 蚂蚁在转移过程中可能会出现死锁,无法成功搜索到终点而成为无效蚂蚁.

基于上述第1个不同点,可考虑将信息素存储于各个栅格中,且规定蚂蚁更容易向信息素浓度高的相邻栅格转移. 该信息素存储方式可降低算法对内存空间的要求,降低信息素更新过程的计算量,提高算法的执行效率. 此时,状态转移概率取

$$P_{ij}^{k}(t)=\begin{cases} \dfrac{\tau_j^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\limits_{v\in \mathrm{allow}d_{ik}} \tau_v^{\alpha}(t)\eta_{iv}^{\beta}(t)}, & j\in \mathrm{allow}d_{ik}; \\ 0, & \text{otherwise}. \end{cases} \tag{8}$$

其中: $P_{ij}^{k}(t)$为t时刻蚂蚁k由栅格i转移到栅格j的概率; $\tau_j(t)$为栅格j的信息素浓度; α为信息启发式因子,表征轨迹的相对重要性; $\eta_{ij}(t)$表示启发函数,为$1/d_{ij}$; β为期望启发式因子; $\mathrm{allow}d_{ik}$为第k只蚂蚁位于栅格i时,能够转移到下一栅格组成的集合,该集合为

$$\mathrm{allow}d_{ik}=\mathrm{allow}d_i-\mathrm{Path}. \tag{9}$$

其中: $\mathrm{allow}d_i$是栅格i周围8个栅格中的可行栅格的集合; Path为第k只蚂蚁经过栅格的集合.

算法每次迭代中,让所有蚂蚁各自完成一次路径搜索,并在每只到达终点或出现死锁时停止搜索. 每次迭代结束后,根据各蚂蚁搜索到的路径,对各栅格的信息素浓度作如下调整:

$$\tau_j(t+1)=(1-\rho)\tau_j(t)+\sum_{k=1}^{m}\Delta\tau_j^k. \tag{10}$$

其中: m为蚂蚁总数量; $\Delta\tau_j^k$为本次迭代中,第k只蚂蚁给栅格j带来的信息素浓度增量. 只有成功搜索到终点的蚂蚁对群体有贡献,其释放的信息素才具有借鉴意义. 因此,信息素增量取

$$\Delta\tau_j^k=\begin{cases} \dfrac{Q}{L_k}, & \text{蚂蚁}k\text{到达终点,且经过栅格}j; \\ 0, & \text{otherwise}. \end{cases} \tag{11}$$

其中: Q为常数, L_k为蚂蚁搜索到的路径长度.

由式(10)和(11)可以得出,信息素存储于栅格中的蚁群算法并不适用于求解TSP问题,否则求解过程中的所有节点的信息素增量始终相同.

2　蚁群算法改进

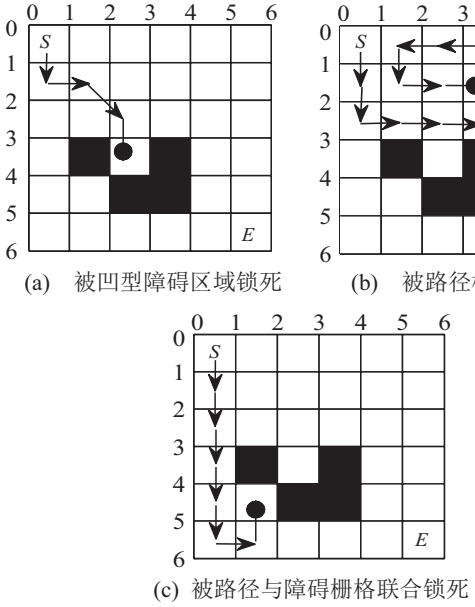
2.1　死锁处理策略

死锁是指蚂蚁在路径搜索过程中,转移到某个非终点栅格后,无满足转移条件的下一栅格可选择,导致路径搜索被迫终止的现象. 发生死锁的蚂蚁并未到达终点,其搜索的路径无效,等效于减少了算法中有效蚂蚁的数量,不利于算法的收敛速度和精度. 死锁出现的原因有两种:

1) 地图中存在凹型障碍区域,如图2(a)中的蚂蚁

由15号栅格转移到21号栅格后,无法继续转移到下一可行栅格,即在21号栅格发生障碍死锁.

2) 蚂蚁被自身经过的栅格锁死,或者被自身经过的栅格与障碍栅格组成区域锁死,分别如图2(b)和图2(c)所示,本文将两种情况统称为自死锁.



(a) 被凹型障碍区域锁死 (b) 被路径栅格锁死

(c) 被路径与障碍栅格联合锁死

图 2 发生死锁的情况

图2(a)中,任何蚂蚁转移到凹型障碍区域都必然会发生障碍死锁,目前已经有较多文献对该死锁情况进行研究,提出的解决方案主要是将此时蚂蚁所在的栅格进行信息素清除或者加入禁忌表,以避免其他蚂蚁再次进入该栅格. 障碍死锁发生的形式单一,问题的分析处理较为简单,而图2(b)和图2(c)中发生自死锁的主要原因是被自身路径包围,不同路径对同一栅格的影响不同,其分析处理复杂得多. 目前,尚未见到有文献对自死锁问题进行分析且采取有效的措施,而自死锁现象在某些地图中出现的概率较大,可能对算法的性能有较严重的影响.

本文针对两种死锁类型的特点,建立全局禁忌表和局部禁忌表,分别用于存储不同类型的栅格. 全局禁忌表G_Tab在算法初始化时为空,并在整个算法执行过程中不断加入蚂蚁发生障碍死锁时所在的栅格,该禁忌表对算法每次迭代中的所有蚂蚁均有约束作用. 局部禁忌表L_Tab在每只蚂蚁开始路径搜索时清空,并仅用于存储该蚂蚁经过且不属于全局禁忌表中的栅格,L_Tab仅对本次迭代中的当前蚂蚁有约束作用.

处于栅格i的蚂蚁k在转移时,首先判断当前是否发生死锁. 若满足下式则表示未发生死锁:

$$(\mathrm{allow}d_i \bigcap (L\_\mathrm{Tab} \bigcap G\_\mathrm{Tab})) \neq \mathrm{allow}d_i. \tag{12}$$

未发生死锁时,则将栅格i加入L_Tab,并令蚂蚁按式(8)继续转移至下一栅格,此时式(9)应修正为

$$\mathrm{allow}d_{ik} = \mathrm{allow}d_i - (L\_\mathrm{Tab} \bigcap G\_\mathrm{Tab}). \tag{13}$$

若蚂蚁位于栅格i时发生死锁,则需进一步判断发生死锁的类型,若为障碍死锁,应满足

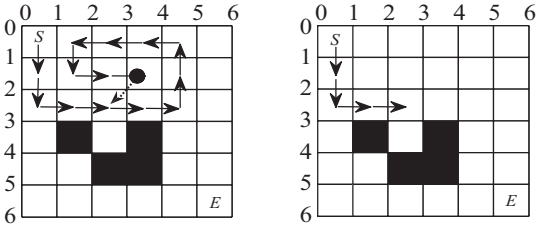
$$\mathrm{Card}(\mathrm{allow}d_i \bigcap L\_\mathrm{Tab}) = 1, \tag{14}$$

其中Card()函数表示取集合中元素个数. 此时,将栅格i加入G_Tab,再将蚂蚁回退至上一栅格. 若栅格i已处于L_Tab,则将其从L_Tab中移除.

若发生死锁且不满足式(14),则表明蚂蚁发生自死锁. 自死锁的情况较为复杂,多数情况下回退一步对跳出死锁的帮助有限,而直接回到起点等效于增加了算法中蚂蚁的总数量,本质上并未提升算法的性能. 因此,本文采用一种折中的转移策略,即蚂蚁发生自死锁时,让其转移至其周围8个栅格中最先经过的栅格,并从L_Tab中移除路径中构成环形部分的栅格. 实现时,假设L_Tab中含有m个元素$\{o_1, o_2, \ldots, o_m\}$,首先取集合$A$为

$$A = \mathrm{allow}d_i \bigcap L\_\mathrm{Tab}; \tag{15}$$

然后,将L_Tab中元素按顺序与集合A中的元素进行对比,直至在L_Tab中找到第1个属于集合A的元素,假设为$o_n (n < m)$;最后,将蚂蚁转移到地图中第o_n号栅格,同时将L_Tab中第$n+1$至第m个元素清除,即自死锁处理后的$L_\mathrm{Tab} = \{o_1, o_2, \ldots, o_n\}$. 该策略的应用效果如图3所示.



(a) 发生自死锁及转移 (b) 自死锁转移后效果

图 3 自死锁时的转移处理策略

图3(a)中,蚂蚁经过1-7-13-14-15-16-17-11-5-4-3-2-8-9-10号栅格后,在10号栅格发生自死锁. 依据本文自死锁转移策略,首先判断10号栅格周围的8个栅格分别为3、4、5、9、11、15、16、17号栅格;然后对比蚂蚁经过的栅格,得到周围栅格中的15号栅格是蚂蚁最初经过的,则令蚂蚁转移到15号栅格;最后,将构成环形的15、16、17、11、5、4、3、2、8、9、10号栅格从L_Tab中删除,让蚂蚁从15号栅格继续进行路径搜索,如图3(b)所示.

该策略既能快速跳出当前死锁,又可在一定程度上利用该蚂蚁已经搜索到的部分路径. 若蚂蚁在路

径搜索过程中,出现自死锁的次数达到设定的机会次数,则表明其路径质量可能较差,所提策略对其作用有限,应结束该蚂蚁的路径搜索,视为无效蚂蚁. 增加机会次数,可以在一定程度上减少无效蚂蚁的产生概率,但也可能会降低算法的执行效率.

2.2　自适应启发函数设计

TSP问题中的路径搜索需遍历所有节点,无法估计下一节点离终点的最短距离,其启发函数仅采用相邻节点之间距离的倒数. 而栅格地图中,相邻栅格的距离d_{ij}为1或1.414,在蚁群算法执行初期的信息素浓度分布均匀时,采用式(8)中的启发函数无法发挥作用. 在蚂蚁转移过程中,一般希望以最短的直线距离移动至终点,因此可以将转移后的栅格j与终点栅格E之间的欧式距离也作为转移过程的参考因素,取启发函数[12]为

$$\eta_{ij}(t)=\frac{1}{d_{ij}+d_{jE}}. \tag{16}$$

基于式(16),当地图规模较大且蚂蚁距离终点较远时,选择不同栅格所对应的启发函数值相差极小,路径选择仍具有很高的随机性;当蚂蚁接近终点时,选择不同栅格的启发函数值相差太大,会使蚂蚁在接近终点区域内路径选择趋于单一. 因此,本文在式(16)基础上,引入当前栅格与终点栅格的距离,设计自适应启发函数为

$$\eta_{ij}(t)=\frac{1}{d_{ij}+d_{jE}-d_{iE}+C}, \tag{17}$$

其中C为常数. 式(17)可使蚂蚁选择不同栅格的启发函数值之间比值不会随所处位置出现较大变化,克服式(16)中启发函数的缺陷. 通过C可以设置启发函数值之间比值的范围.

2.3　最优路径保留

蚁群算法是一种依据概率的随机搜索算法. 该算法在路径迭代搜索过程中可能会丢失已搜索到的最优路径,即算法上一次迭代中搜索到的最优路径,在本次迭代中并未出现,并且本次迭代中搜索到的所有路径长度均大于上一次最优路径长度. 图4为在某栅格地图中采用常规蚁群算法进行路径规划的最优路径收敛曲线,算法在第3、6、12、14次迭代搜索时均出现了最优路径丢失的情况.

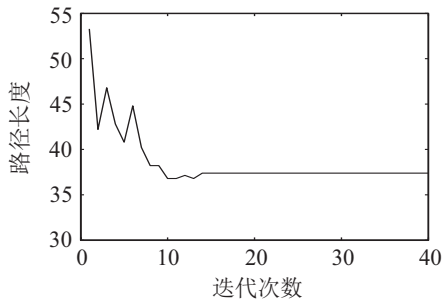


图 4　最优路径长度

由于信息素的挥发效应,最优路径丢失会导致该路径上信息素浓度降低,不利于算法的收敛. 针对该现象,本文首先将本次迭代中搜索到的最优路径与历史最优路径进行比较,若长度大于历史最优路径的长度,则用历史最优路径替代本次迭代中搜索到的最长路径或者任意一条无效路径,使得历史最优路径在算法迭代过程中得以保留,引导算法向已搜索到的最优路径收敛.

2.4　改进蚁群算法实现

step 1:　初始化算法参数,确定最大迭代次数$\mathrm{NC}_{\max}$,蚂蚁数量$M_{\max}$,令变量$k=1,\mathrm{nc}=1$,清空全局禁忌表G_Tab.

step 2:　令死锁计数器$\mathrm{dead_num}=0$,清空局部禁忌表L_Tab,执行以下步骤实现蚂蚁k的路径搜索.

step 2.1: 依据式(12)判断当前时刻的蚂蚁k是否发生死锁,若未发生死锁,则将当前栅格加入L_Tab中,并依据式(8)、(13)和(17)进行转移,然后跳转至step 2.4;否则,继续执行step 2.2.

step 2.2: 根据式(14)判断发生死锁的类型,若是障碍死锁,则将当前栅格加入G_Tab,并将蚂蚁回退至上一栅格,再回跳转至step 2.1;否则,继续执行step 2.3.

step 2.3: 依据所提出的自死锁转移策略进行处理,同时自死锁计数器$\mathrm{dead_num}=\mathrm{dead_num}+1$,再继续执行step 2.4.

step 2.4: 若蚂蚁k未到达终点且$\mathrm{dead_num}$小于设定的机会次数,则回跳至step 2.1;否则,终止该蚂蚁的路径搜索,继续执行step 4.

step 3: 令$k=k+1$,若$k\leqslant M_{\max}$,则跳转至step 2;否则,继续执行step 3.

step 4: 若搜索到的$M_{\max}$条路径中可行路径长度均大于上轮搜索到的最优路径,则将本轮迭代中的任一无效路径或最长路径用上轮最优路径替代.

step 5: 根据式(10)和(11)进行信息素的更新.

step 6: 令$\mathrm{nc}=\mathrm{nc}+1$,若$\mathrm{nc}\leqslant\mathrm{NC}_{\max}$,则令$k=1$,跳转至step 2;否则,算法迭代结束,输出最优路径.

3　仿真研究

为验证改进蚁群算法的有效性,采用该算法进行多组路径规划仿真研究. 仿真计算机配置为Intel i7双核处理器,8 G内存,64位Win7操作系统,Matlab 2014仿真软件. 算法的参数取值如表1所示.

表1 改进蚁群算法参数

参数	数值	参数	数值
迭代次数	50	β	8
蚂蚁数量	30	ρ	0.3
机会次数	3	Q	30
α	2	C	10

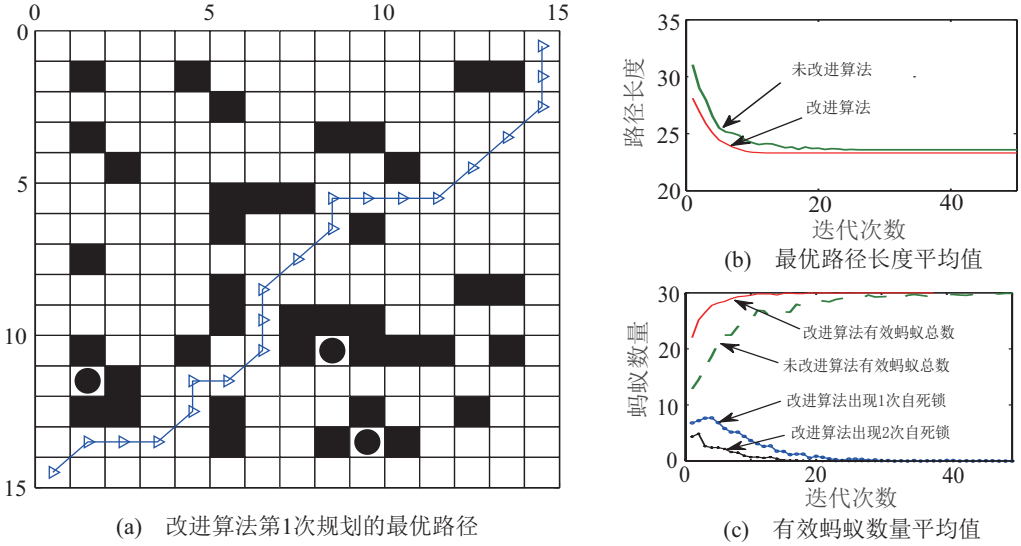
3.1 不同规模地图中的仿真研究

3.1.1 测 试 1

采用本文改进算法和未改进算法在地图1(规模为15 × 15)、地图2(规模为30 × 30)和地图3(规模为50 × 50)[9]中各进行20次重复路径规划, 对比两种算法历次迭代中的最优路径长度平均值和有效蚂蚁数量平均值. 为便于统计计算,若算法某次迭代中没有蚂蚁成功到达终点,则将该次迭代的最优路径长度置为1 000. 仿真结果分别如图5~图7所示.

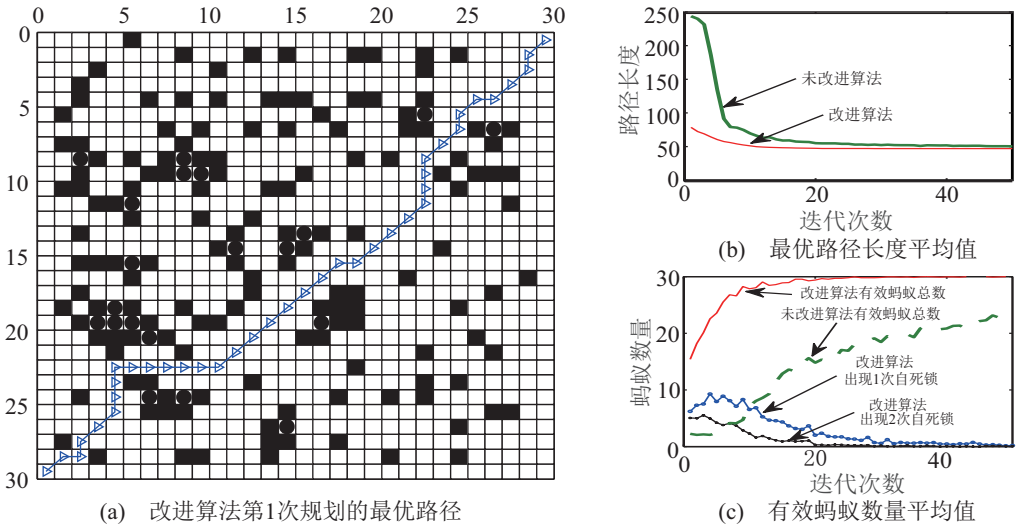
图5~图7的仿真结果表明,改进蚁群算法在3种地图中经过一定的迭代计算后,均能够搜索到一条较(最)优路径,并且在路径搜索过程中能够成功辨识导致障碍死锁的栅格,如图5(a)、图6(a)和图7(a)中的黑色圆圈所示,避免其他蚂蚁再次发生相同的障碍死锁. 由于机会次数限定为3次,改进算法在进行路径搜索时,仍有部分蚂蚁因出现多次自死锁而成为无效蚂蚁,并且在算法初期信息素分布比较均匀的情况下,无效蚂蚁出现的频率更高. 同时,3种地图中的地图规模越大,改进算法初期出现有效蚂蚁的概率越低,且有效蚂蚁中能够不经过自死锁直接到达终点的比例更低. 特别是在地图3中,改进算法前几次迭代中的所有有效蚂蚁均经历过自死锁.

两种算法的仿真对比结果表明,改进算法在3种地图中均能大幅度提高有效蚂蚁数量,历次迭代过程中搜索到的最优路径长度平均值更小、收敛速度更快. 在地图3中,未改进算法出现多次迭代中均无蚂蚁能够搜索到终点的现象,经过50次的迭代后,其产生有效蚂蚁数量的均值才达到改进算法的初期水平,



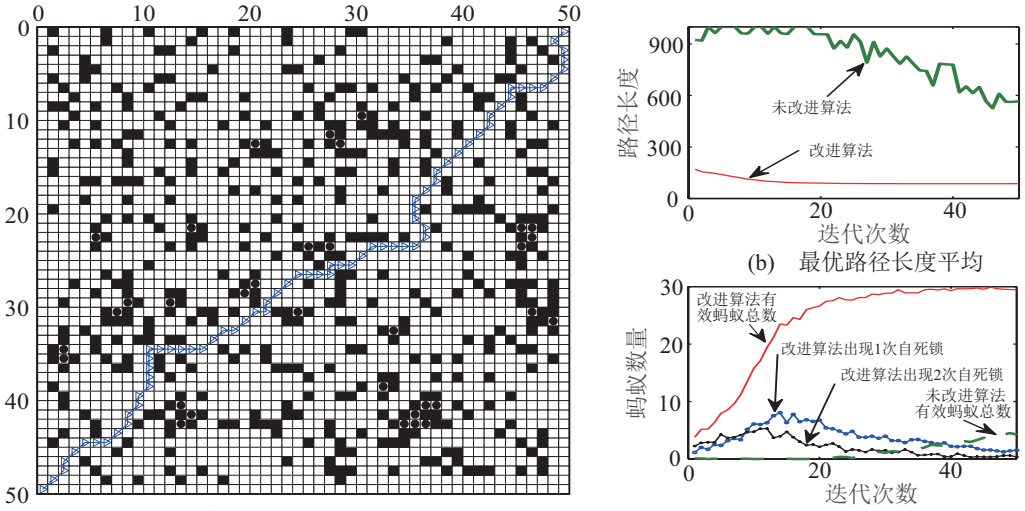
(a) 改进算法第1次规划的最优路径　(b) 最优路径长度平均值　(c) 有效蚂蚁数量平均值

图5 地图1中2种算法仿真对比



(a) 改进算法第1次规划的最优路径　(b) 最优路径长度平均值　(c) 有效蚂蚁数量平均值

图6 地图2中2种算法仿真对比



(a)　改进算法第1次规划的最优路径　　(b)　最优路径长度平均　　(c)　有效蚂蚁数量平均

图 7　地图3中2种算法仿真对比

表 2　3种地图中的2种算法性能对比

性能指标	地图1			地图2			地图3		
	未改进算法	改进算法	性能提升/%	未改进算法	改进算法	性能提升/%	未改进算法	改进算法	性能提升/%
路径长度最小值	23.317	23.317	0	45.941	45.113	1.8	114.073	83.256	27.0
路径长度平均值	23.589	23.317	1.2	49.315	46.993	4.7	581.276	85.674	85.3
算法平均耗时/s	0.729	0.682	6.4	9.713	5.778	40.5	64.020	34.792	42.5

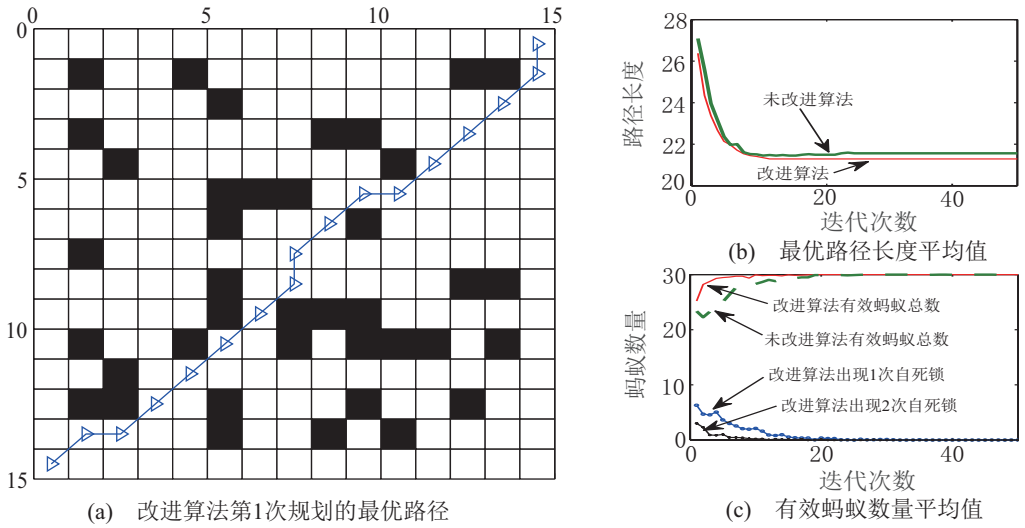
如图7(c)所示. 再结合图7(b)和图7(c)可以看出,未改进算法经过50次迭代后,其搜索到可行路径的几率仍低于改进算法第1次迭代搜索到可行路径的几率.

对两种算法在不同地图中重复执行得到的20条路径长度和算法耗时进行统计对比,如表2所示. 由表2可以看出,改进算法搜索到20条路径长度的最小值和平均值均更优,算法运行的平均耗时更短. 在规模较小的地图1中,两种算法都极易搜索到最优解,搜索到的路径平均值和耗时两方面相差微小. 随着地图规模扩大,改进算法的各项性能优势逐渐显现.

3.1.2　测 试 2

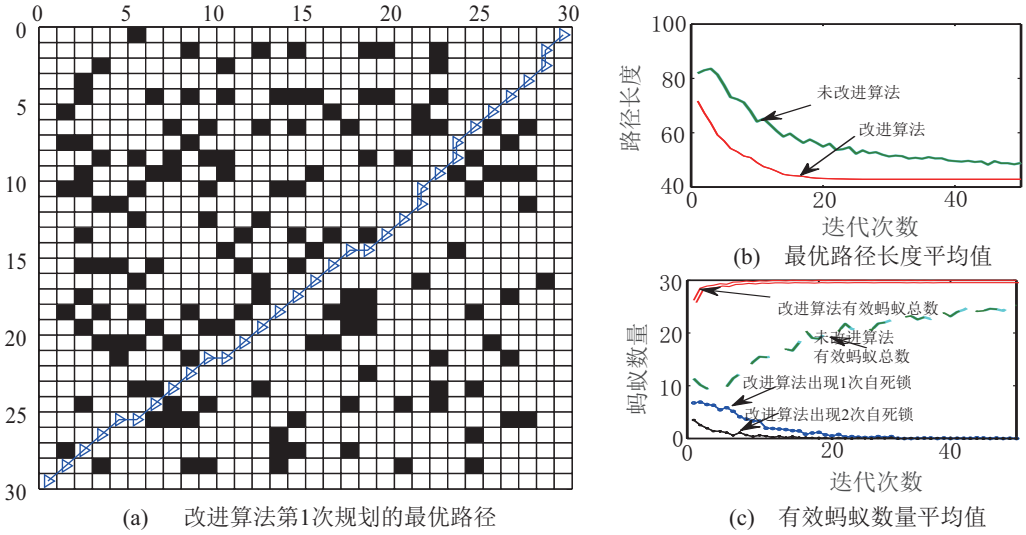
为与文献[9]中的仿真结果进行对比,将本文算法中转移约束条件修改为与文献[9]中一致,即蚂蚁在其相邻栅格中转移只需满足$\mathrm{Map}(g) = 0$即可. 采用修改转移约束条件的改进算法和未改进算法再次在3.1.1节的3种地图中各进行20次的重复路径规划,得到仿真结果如图8～图10所示.

图8～图10的仿真结果表明,修改转移约束条件后,改进算法在3种地图中仍保持其优越性. 与3.1.1节的测试结果相比,由于修改后的转移约束条件更为



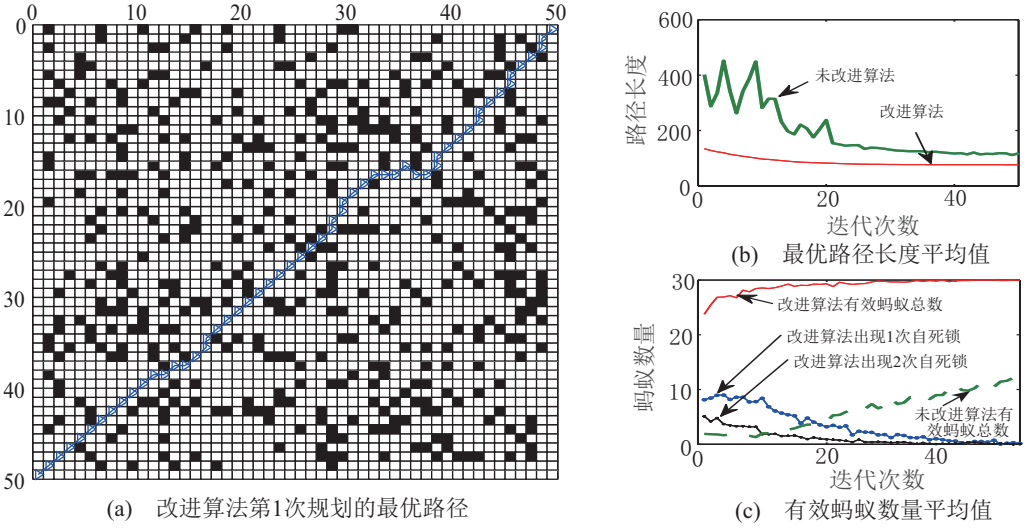
(a)　改进算法第1次规划的最优路径　　(b)　最优路径长度平均值　　(c)　有效蚂蚁数量平均值

图 8　地图1中2种算法(修改约束条件)仿真结果对比



(a) 改进算法第1次规划的最优路径 (b) 最优路径长度平均值 (c) 有效蚂蚁数量平均值

图 9 地图2中2种算法(修改约束条件)仿真结果对比



(a) 改进算法第1次规划的最优路径 (b) 最优路径长度平均值 (c) 有效蚂蚁数量平均值

图 10 地图3中2种算法(修改约束条件)仿真结果对比

表 3 与文献[9]中结果的性能对比

性能指标	地图1			地图2			地图3		
	文献[9]	本文算法	性能提升/%	文献[9]	本文算法	性能提升/%	文献[9]	本文算法	性能提升/%
路径长度	20.97	21.26	−1.5	41.59	42.51	−2.2	77.15	75.21	2.5
耗时/s	0.579	0.612	−5.7	5.42	5.16	4.8	33.042	30.658	7.2

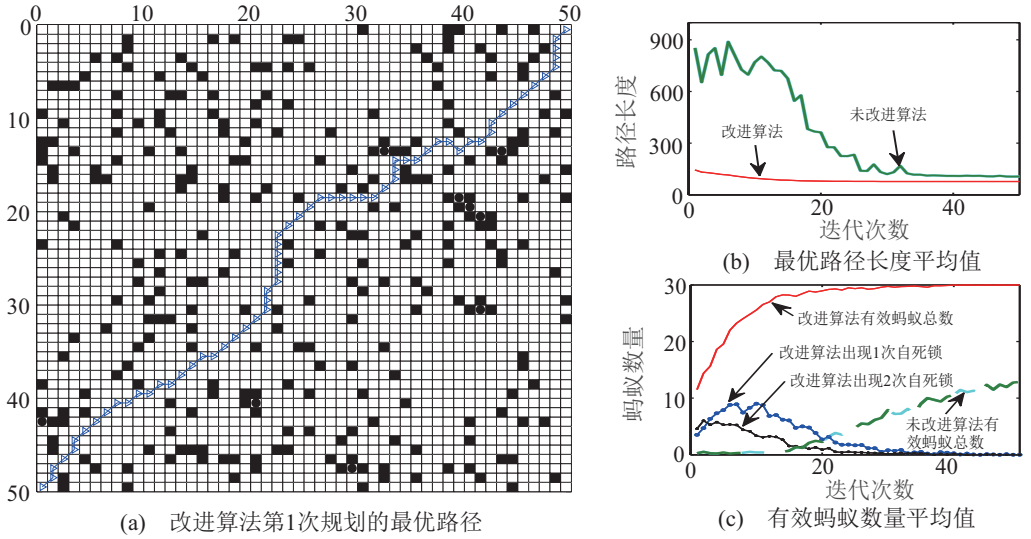
宽松,蚂蚁转移更灵活,两种算法在3种地图中均不再出现障碍死锁,且产生自死锁的概率更低,规划得出的最优路径长度更短.

统计改进算法重复执行所输出的平均路径长度与耗时,将其与文献[9]中的结果进行对比. 考虑到耗时与计算机性能相关,因此将文献[9]中的基本蚁群算法(ACA)在本仿真计算机中执行,统计其在上述3种地图进行路径规划的平均耗时并作为参考基准,再对文献[9]提出算法的耗时结果通过比例换算,得到其在本仿真计算上执行的等效耗时. 实测ACA算法的平均耗时分别为1.204 s、10.925 s、72.709 s,引用文献[9]中的路径长度并换算其等效耗时,得到与本文算法的对比结果如表3所示.

表3中,本文所提出算法在规模较小的地图1中,其规划的平均路径长度与执行耗时均略大于文献[9]的结果;在规模中等的地图2中,本文算法的耗时更短;随着地图规模扩大到50×50,本文算法在路径长度和耗时两方面性能均更优. 对比结果表明,本文算法与文献[9]中的算法在不同规模地图中的应用各具优势,本文算法在测试的大规模地图中表现出了更好的路径规划效果.

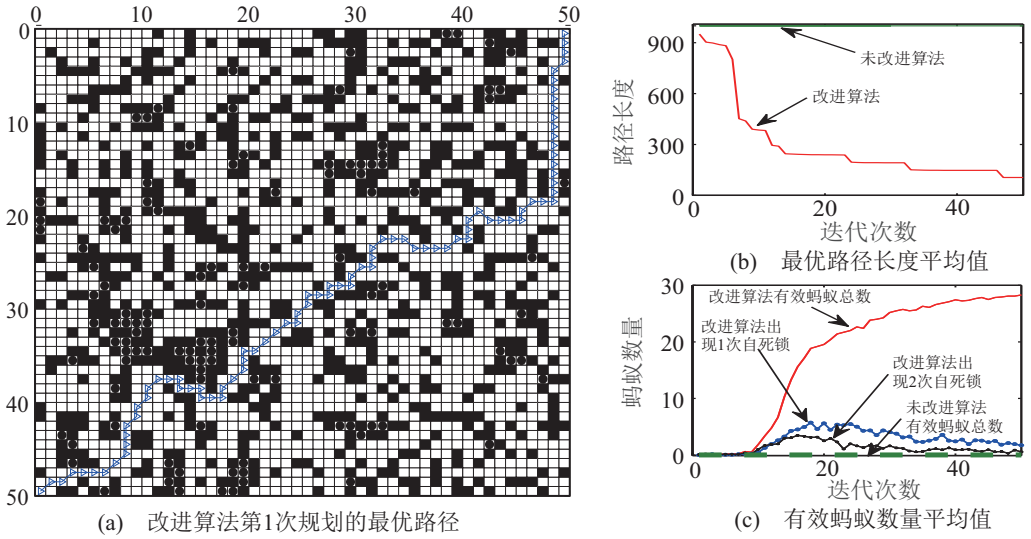
3.2 不同复杂度地图中的仿真研究

为进一步研究本文算法在相同规模、不同复杂度地图中的路径规划效果,随机产生两种规模均为450 × 50的地图4(障碍栅格比例为0.1)和地图5(障碍



(a)　改进算法第1次规划的最优路径　　(b)　最优路径长度平均值　　(c)　有效蚂蚁数量平均值

图 11　地图4中2种算法仿真对比



(a)　改进算法第1次规划的最优路径　　(b)　最优路径长度平均值　　(c)　有效蚂蚁数量平均值

图 12　地图5中2种算法仿真对比

栅格比例为0.3). 采用本文改进算法和未改进算法在两种地图中各进行20次重复路径规划,得到仿真结果如图11和图12所示.

图11中的地图复杂度较低,改进算法迭代初期的有效蚂蚁数量较多,而未改进算法直到第13次迭代开始才逐渐明显出现有效蚂蚁. 图12中的地图复杂度较高,改进算法在迭代初期的有效蚂蚁也极少,直到第7次迭代开始才逐渐明显出现有效蚂蚁,而未改进算法50次迭代中均未出现有效蚂蚁. 结合图7、图11和图12的仿真结果可知,在3种不同复杂度的栅格地图中,本文提出的改进算法均能提高有效蚂蚁数量、求取更优路径,且地图复杂度越高,该算法的性能提升效果越明显.

图12的仿真结果也表明,本文死锁处理策略的主要作用是提高有效蚂蚁产生的概率,而不能保证算法每次迭代中都一定有蚂蚁达到终点. 不难推论,随着地图规模和复杂度不断增大,采用本文改进算法也将出现50次迭代中均未产生有效蚂蚁的情况,增大机会次数可以在一定程度上提高有效蚂蚁产生的概率,但设定机会次数的上限仍有必要,否则蚂蚁一旦进入某些区域,如图6(a)第10行、第5列的位置,算法将会出现死循环.

4　结　论

死锁是栅格地图中采用蚁群算法进行路径规划时都存在的固有问题,本文从原理上分析了出现死锁的原因,给出了两种死锁类型的判断方法和针对性的死锁转移处理策略,可避免障碍死锁的发生,降低自死锁导致无效蚂蚁产生的概率. 同时,对蚁群算法的启发函数进行改进,并采用最优路径保留策略,以提高算法的收敛速度和精度. 在不同规模、不同复杂度地图中的仿真结果验证了本文改进算法的优越性. 改进算法实现过程简单,所提出自死锁处理策略

可以便捷地叠加在目前大多用于路径规划的蚁群算法中,而不与算法的其他改进措施相冲突,具有较高的应用价值.

改进算法中的机会次数取值应与地图规模及地图复杂度相关,如何选取适当的机会次数值得进一步探讨. 本质上,所提出自死锁处理策略是为出现自死锁蚂蚁提供新的选择机会,以提高有效蚂蚁产生的概率,研究更加灵活有效地跳出死锁策略能够增加蚁群算法在大规模、复杂地图中路径规划的适用性. 综合考虑风、浪、流等环境影响,结合无人艇水动力参数,建立含有环境因素的目标函数,可使无人艇路径规划算法的实用性更强.

参考文献(References)

[1] Luo G C, Yu J Q, Mei Y S, et al. UAV path planning in mixed-obstacle environment via artificial potential field method improved by additional control force[J]. Asian Journal of Control, 2015, 17(5): 1600-1610.

[2] Radmanesh M, Kumar M, Guentert P H, et al. Overview of path planning and obstacle avoidance algorithms for UAVs: A comparative study[J]. Unmanned Systems, 2018, 6(2): 95-118.

[3] 陈志旺, 夏顺, 李建雄, 等. 基于定向A*算法的多无人机同时集结分步策略[J]. 控制与决策, 2019, 34(6): 1169-1177.
(Chen Z W, Xia S, Li J X, et al. Serial strategy for rendezvous of multiple UAVS based on directional A* algorithm[J]. Control and Decision, 2019, 34(6): 1169-1177.)

[4] Persson S M, Sharf I. Sampling-based A* algorithm for robot path-planning[J]. The International Journal of Robotics Research 2014, 33(13): 1683-1708.

[5] Roberge V, Tarbouchi M, Labonté G. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning[J]. IEEE Transactions on Industrial Informatics, 2013, 9(1): 132-141.

[6] 何庆, 吴意乐, 徐同伟. 改进遗传模拟退火算法在TSP优化中的应用[J]. 控制与决策, 2018, 33(2): 219-225.
(He Q, Wu Y L, Xu T W. Application of improved genetic simulated annealing algorithm in TSP optimization[J]. Control and Decision, 2018, 33(2): 219- 225.)

[7] Ma Y, Hu M Q, Yan X P. Multi-objective path planning for unmanned surface vehicle with currents effects[J]. ISA Transactions, 2018, 75: 137-156.

[8] Xiong G M, Li X Y, Zhou S, et al. Incorporating bidirectional heuristic search and improved ACO in route planning[J]. International Journal of Hybrid Information Technology, 2015, 8(7): 189-198.

[9] 张玮, 马焱, 赵捍东, 等. 基于改进烟花-蚁群混合算法的智能移动体避障路径规划[J]. 控制与决策, 2019, 34(2): 335-343.
(Zhang W, Ma Y, Zhao H D, et al. Obstacle avoidance path planning of intelligent mobile based on improved fireworks-ant colony hybrid algorithm[J]. Control and Decision, 2019, 34(2): 335-343.)

[10] 柳长安, 鄢小虎, 刘春阳, 等. 基于改进蚁群算法的移动机器人动态路径规划方法[J]. 电子学报, 2011, 39(5): 1220-1224.
(Liu C A, Yan X H, Liu C Y, et al. Dynamic path planning for mobile robots based on improved ant colony optimization algorithm[J]. Acta Electronica Sinica, 2011, 39(5): 1220-1224.)

[11] 王宏健, 伍祥红, 施小成. 基于蚁群算法的AUV全局路径规划方法[J] . 中国造船, 2008, 49(2): 88-93.
(Wang H J, Wu X H, Shi X C. Global planning path method of AUV based on ant colony optimization algorithm[J]. Shipbuilding of China, 2008, 49(2): 88-93.)

[12] Châari I, Koubâa A, Trigui S, et al. SmartPATH: An efficient hybrid ACO-GA algorithm for solving the global path planning problem of mobile robots[J]. International Journal of Advanced Robotic Systems, 2014, 11(7): 94-109.

[13] 曾明如, 徐小勇, 罗浩, 等. 多步长蚁群算法的机器人路径规划研究[J]. 小型微型计算机系统, 2016, 37(2): 366-369.
(Zeng M R, Xu X Y, Luo H, et al. Research of robot path planning based on multi-step ant colony algorithm[J]. Journal of Chinese Computer System, 2016, 37(2): 366-369.)

[14] Deng X Y, Zhang L M, Luo L. An improved ant colony optimization applied in robot path planning problem[J]. Journal of Computers, 2013, 8(3): 585-593.

[15] 屈鸿, 黄利伟, 柯星. 动态环境下基于改进蚁群算法的机器人路径规划研究[J]. 电子科技大学学报, 2015, 44(2): 260-265.
(Qu H, Huang L W, Ke X. Research of improved ant colony based robot path planning under dynamic environment[J]. Journal of University of Electronic Science and Technology of China, 2015, 44(2): 260-265.)

[16] 谭覃, 刘树东, 张艳. 移动机器人路径规划仿真研究[J]. 计算机仿真, 2016, 33(8): 354-358.
(Tan T, Liu S D, Zhang Y. Simulation Study of path planning method for mobile robot[J]. Computer Simulation, 2016, 33(8): 354-358.)

作者简介

孙功武(1990−), 男, 工程师, 硕士, 从事舰船智能控制的研究, E-mail: sungongwu@126.com;

苏义鑫(1965−), 男, 教授, 博士生导师, 从事智能控制、路径规划等研究, E-mail: suyixin@whut.edu.cn;

顾轶超(1991−), 男, 工程师, 硕士, 从事舰船信息系统、智能计算的研究, E-mail: 458683767@qq.com;

谢基榕(1977−), 男, 研究员, 博士, 从事舰船信息系统、智能控制等研究, E-mail: xiejirong@126.com;

王俊轩(1986−), 男, 高级工程师, 硕士, 从事舰船信息系统等研究, E-mail: wangjunxuan@126.com.

(责任编辑：孙艺红)