第 17 卷第 2 期 2019 年 4 月
1672-6553/2019/17(2)/104-8

动力学与控制学报
JOURNAL OF DYNAMICS AND CONTROL

Vol.17 No.2
Apr. 2019

一种改进蚁群算法的移动机器人快速路径规划算法研究*

谭会生[1†]　廖雯[2]　贺迅宇[3]

(1.湖南工业大学 交通工程学院, 株洲　412007) (2.湖南工业大学 电气与信息工程学院, 株洲　412007)

(3.湖南工业大学 研究生院, 株洲　412007)

摘要　以 Dijkstra 算法求解移动机器人路径规划(mobile robot path planning,MRPP)问题已得到广泛的应用,但在复杂工况下无法保证求解的正确性和全局最优性.而基于蚁群算法的移动机器人路径规划模型,在一定条件下能可靠地获得全局最优解,但存在求解时间过长的问题.因此,提出一种结合 Dijkstra 算法和蚁群算法模型两者优势求解 MRPP 问题的融合优化方法,以实现在短时间内获得全局最优解的目标.首先,应用 Dijkstra 快速算法在机器人工作环境中粗略寻迹得到最短路径次优解,然后,在次优解路径附近进行工作环境的精确划分;最后,利用蚁群算法在次优解附近精确寻迹,使最终的寻迹结果无限逼近最短路径.仿真结果表明,该融合优化方法既克服了经典蚁群算法求解时间过长的缺点,又能无限逼近全局最优解,寻迹时间较蚁群算法可缩短 90%以上.

关键词　移动机器人,　路径规划,　Dijkstra 算法,　蚁群算法,　融合优化算法

DOI:　10.6052/1672-6553-2018-057

引言

移动机器人的路径规划是机器人研究领域的核心内容之一,同时也是机器人导航技术的一个重要环节和课题[1].它的任务就是在具有障碍物的环境内按照一定的评价标准,寻找到一条从起始状态到目标状态的无碰撞路径[1~4].

传统移动机器人路径规划方法有:人工势场法、自由空间法、人工神经网络算法、遗传算法、Dijkstra 算法、蚁群算法等.这些机器人路径规划算法都存在不同程度上的不足.人工势场法结构简单,对低层的实时控制十分容易实现,在实时避障和平滑的轨迹控制方面应用较多,但是容易产生局部最优解问题和死锁现象.自由空间法存在计算效率较低,存储空间大的缺陷.以 Dijkstra 算法求解移动机器人路径规划(mobile robot path planning,MRPP)问题已得到广泛的应用,但在复杂工况下无法保证求解的正确性和全局最优性.而基于蚁群算法的移动机器人路径规划模型,在一定条件下能可靠地获得全局最优解,但存在求解时间过长的问题[5,6].

本文提出了一种结合 Dijkstra 算法和蚁群算法模型两者优势求解 MRPP 问题的融合优化方法,以实现在短时间内获得全局最优解的目标.首先,应用 Dijkstra 快速算法在机器人工作环境中粗略寻迹得到最短路径次优解,然后,在次优解路径附近进行工作环境的精确划分;最后,利用蚁群算法在次优解附近精确寻迹,使最终寻迹结果无限逼近最短路径.

1　Djikstra 算法与蚁群算法融合的移动机器人路径规划

关于最短路径问题,目前所公认的最好的方法是由 F. W. Dijkstra 提出的方法,即 Dijkstra 算法.Dijkstra算法按路径长度的递增次序,逐条产生最短路径.在仿生寻迹算法方面,意大利学者 M. Dorigo 提出了蚂蚁算法.几乎不具备视觉的蚂蚁之所以能够找到蚁巢到食物之间的最短路径,是靠其在走过

2017-12-29 收到第 1 稿,2018-05-19 收到修改稿.

* 国家自然科学基金资助项目(6167224),湖南省自然科学基金资助项目(2016JJ6036)

† 通讯作者 E-mail:huisheng21nd@163.com

的路上释放一种挥发性分泌物(即 pheromone,信息素)来实现的.蚂蚁群体选择该路径的概率与当时这条路径上信息素的强度成正比.当某路径上通过的蚁群越来越多时,在此路径上留下的信息素也越来越强,使得后来的蚁群选择此路径的概率更高,从而更增加了此路径上的信息素强度,可吸引更多的蚁群选择此路径,这样就形成了一种正反馈机制,使蚁群最终可发现最短路径[7,8].

在对 Dijkstra 算法与蚁群算法的研究与实现发现,蚁群算法在探索初期由于节点众多以及缺乏信息素,使得搜索速度缓慢;当信息素累计到一定强度后即探索空间向着信息素较浓的节点靠拢后,最优解收敛的速度迅速提高.而 Dijkstra 算法在节点较少的情况下,时间复杂度不高,能迅速搜索到最优解.Dijkstra 算法与蚁群算法动态结合的基本思想就是:在探索初期采用 Dijkstra 算法将探索空间向着最优解附近靠拢,在锁定最优解的大致区域后采用蚁群算法求取最优解[9~11].融合算法流程如图 1 所示.

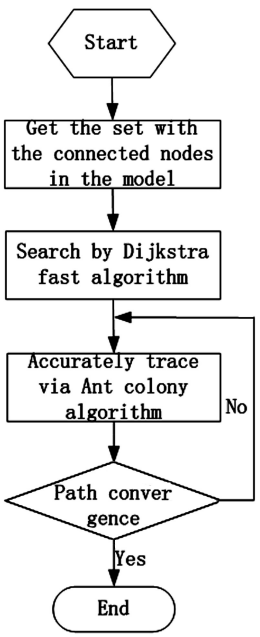

图 1　融合算法流程图

Fig.1　Flow diagram of fusion algorithm

2　环境抽象建模

机器人寻迹空间目前最为广泛采用的是栅格法:将机器人寻迹空间按照某种方式划分为一系列的网格单元[12].而当抽象出来的寻迹节点较多时,建模出来的栅格数将急剧膨胀,导致寻迹耗时大幅度增加,这在实际应用中是不允许的.为了解决此问题,在寻迹的预备阶段采用将非重点寻迹区域的抽象节点精简的策略,目的是将蚁群集中布置在障碍物集中的重点寻迹区域.

改进算法的关键在于如何将寻迹蚁群布置在重点寻迹区域.为此,对机器人工作的环境用节点连线法进行重新抽象划分:将障碍物抽象为不规则棱角分明的不规则多边形,把相邻多边形障碍物的顶点用线段连接,然后将各个线段细分成线段,将寻迹节点集中在这些细线段上,重新抽象划分后的寻迹环境如图 2 所示.

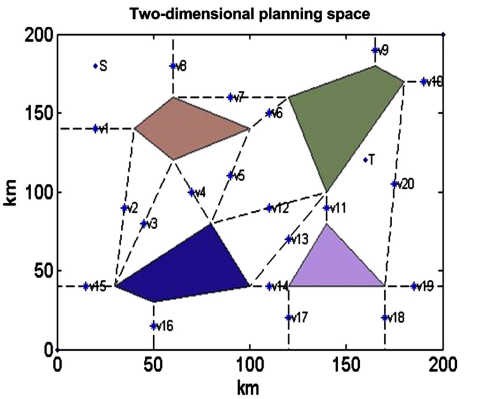

图 2　机器人工作环境抽象建模图

Fig.2　Abstract modeling diagram of the robot work environment

将机器人工作环境进行重新划分后,如前所述的栅格图将大幅简化,寻迹节点大大减少,并集中分布在障碍物周围.即使如此,如直接求取最短路径,过程仍然较为复杂,因为此时节点连线仍较多,所以实际上我们首先采取一种进一步锁定最短路径所在连线区域的方法.先将抽象出来的连线中点(即集合 V)看作有向图集合,以起始点为中心向外层层扩展,直到扩展到终点为止.由于连线中点进一步减少,这样的扩展搜索会变得更加快速高效.

3　Djikstra 算法与和蚁群算法融合的 MRPP 算法

融合算法的思想为:首先,需要对复杂的机器人工作环境进行精简;其次,应用 Dijkstra 快速算法在机器人工作环境中粗略寻迹得到寻迹路径次优解;然后,在次优解路径附近进行工作环境的精确划分;最后,利用蚁群算法在次优解附近精确寻迹,使最终的寻迹结果无限逼近最短路径.

3.1　机器人工作环境的抽象建模

对机器人工作的环境用节点连线法进行重新

抽象划分:将障碍物抽象为不规则棱角分明的不规则多边形,把相邻多边形障碍物的顶点用线段连接,然后将所有的连线进行编号.设顶点连线集合为 S,连线编号为 $s_1,s_2,s_3\cdots$,则 $S=\{s_1,s_2,s_3\cdots\}$.再设所有连线中点集合为 V,连线中点编号为 $v_1,v_2,v_3\cdots$,则 $V=\{v_1,v_2,v_3\cdots\}$.把集合 V 中的元素设为有向图节点.得到多边形端点的连线后,再将集合 S 中的所有线段进行细分,细分后集合 S 中的所有元素将变成细线段,原则上分段越细则寻迹越精确.将每一段虚线编号 $p_1,p_2,\cdots$,设虚线段集合为 $P=\{p_1,p_2,p_3\cdots\}$,即作为寻迹节点.

3.2 应用 Dijkstra 算法的一次加速过程

① 初始化集合 X,D:$X=\{v_0\}$,$D[i]=\text{Arcs}[0][i]$,$i=0,1,\cdots,n-1$;其中 X 为已求出最短路径的顶点集合,D 为当前求得的从起始点 v_0 到每个终点 v_i 的最短路径长度;

② 选择 v_j,使得:$D[j]=\min\{D[i]\mid v_i\in X-Y\}$,$i=0,1,\cdots,n-1$;

③ 修改 v_0 出发到集合 $X-Y$ 上任一节点 v_k 的最短路径长度.若 $D[k]>D[j]+\text{Arcs}[j][k]$,则修改 $D[k]$ 为:$D[k]=D[j]+\text{Arcs}[j][k]$;

④ 重复②和③操作 $n-1$ 次,求得集合 V 中从 v_0 到终点 T 的最短路径长度.记录下此最短路径集合 v_k 所在的线段集合 S_k.

3.3 蚁群算法二次加速精确寻迹

① 初始化蚁群算法控制参数:设置 $\alpha=1$,$\beta=2$,循环次数 $NC=500$,蚁群种群数量 $m=10$,信息素选择阈值 $q=0.8$;初始化启发信息函数.

② 设置蚁群寻迹节点数组 $vv[i][j]$,该节点数组为包含起始点和终点以及 Dijkstra 算法中集合 S_k 中的虚线段节点.

③ 进行第一代蚁群搜索,从起点 v_0 开始,根据下一步可到达节点,计算蚁群所在节点与可到达节点间的延时 $\text{delay}[i][j]$ 和寻迹费用 $\text{cost}[i][j]$.

④ 根据已得到的延时数组 $\text{delay}[i][j]$ 和寻迹费用数组 $\text{cost}[i][j]$,由轮盘赌法选择蚁群的下一个节点.单个蚂蚁依据启发式信息,从节点 i 移动到相邻节点 j 的转移概率计算公式为:

$$p_{ij}^k(t)=\begin{cases}\dfrac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}, & j\in\text{可行域}\\ 0, & \text{其他}\end{cases}\tag{1}$$

式中,η_{ij} 代表路径 ij 上的可见度,可见度定义为 i,j 两点距离的倒数,即 $\eta_{ij}=|d_{ij}|^{-1}$;τ_{ij} 代表蚂蚁路径上节点 i 和 j 之间信息素浓度.

⑤ 每一代蚂蚁结束搜索以后更新路径上的信息素.信息素更新采用带精英策略的更新公式:

$$\tau_{ij}(t+1)=\rho\tau_{ij}(t)+\Delta\tau_{ij}(t,t+1)\tag{2}$$

$$\Delta\tau_{ij}(t,t+1)=\sum_{k=1}^{m}\Delta\tau_{ij}^k(t,t+1)\tag{3}$$

$$\Delta\tau_{ij}^k(t)=\begin{cases}Q/L_k, & \text{蚂蚁 } k \text{ 经过 } ij\\ 0, & \text{蚂蚁 } k \text{ 不经过 } ij\end{cases}\tag{4}$$

式中 Q 代表信息素强度因子,表示蚂蚁循环一次所释放的信息素总量,它在一定程度上影响着算法的收敛速度;L_k 表示第 k 只蚂蚁在本次循环中所走过的路径长度总和.

⑥重复步骤③~⑤,进行下一代蚂蚁搜索,直至所有蚂蚁完成搜索,得到最优路径.

4 仿真与分析

为了验证本算法的有效性,将融合算法的机器人路径规划系统在 Matlab 软件平台下进行了仿真实验研究.为方便说明,将障碍物抽象成为棱角分明的不规则多边形,暂不考虑机器人大小问题.仿真时,设置蚁群算法中的各项参数为:迭代次数 $NC=500$,种群数量 $m=10$,启发因子 $\alpha=1$,$\beta=2$,信息素衰减因子 $\rho=0.9$.每进行一组寻迹实验,则增加一块障碍物,以此来检验算法的有效性,机器人位于起始点(20,180),移动到目标点(180,90).

仿真实验 1:机器人寻迹空间中有 5 块障碍物,机器人位于起始点 S(20,180),对障碍物各顶点进行编号,蚁群经过 500 次迭代以后其实验结果如图 3、图 4 所示.

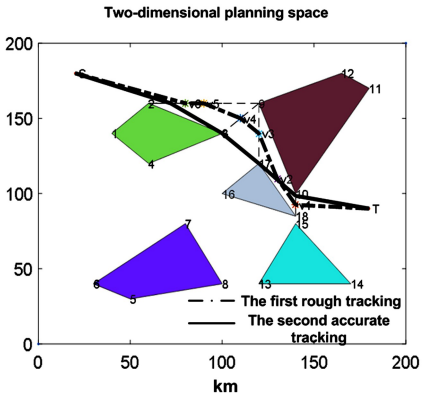

图 3 5 块障碍物融合算法寻迹结果图

Fig.3 Tracking results of fusion algorithm with 5 obstacles

从环境中含 5 块障碍物的实验结果图 3 可以

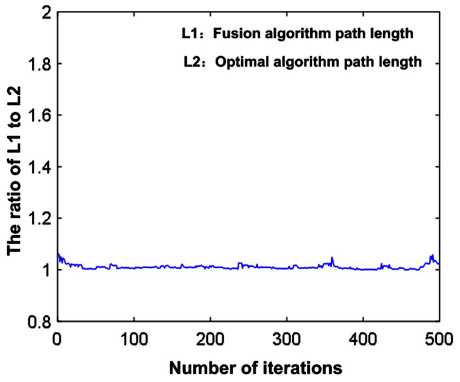

图 4　5 块障碍物两种算法寻迹路径长度对比图

Fig.4　Comparison of tracing path length for two algorithms with 5 obstacles

看出,虚线部分为 Dijkstra 算法的一次粗略寻迹路径,经过的路径点依次为障碍物各顶点连线的中点(环境边界也可以抽象为障碍物),寻找到中点集后,再将中点集所在的连线进行细分,细分所得到的线段集合即抽象为蚁群寻迹的节点集.实线部分为在一次粗略寻迹的范围内,进行蚁群二次精确搜索的路径,其路径与单独使用栅格法-蚁群算法搜索的路径几乎重合.实验事先并未考虑机器人大小,所以寻迹路径几乎与障碍物边缘重合而过.图 4 为融合算法寻迹路径抗度($L1$)与最优路径长度($L2$)的百分比,迭代过程中有小幅波动,但迭代到不同的阶段,此波动均在智能算法可接受的误差范围内.曲线显示,迭代到 20 次左右时融合算法已经无限逼近最优路径,算法运行效率较高.

仿真实验 2:在实验 1 原有障碍物不变的基础上,增加一块障碍物,以此来检验 Dijkstra-蚁群算法对环境改变的适应性.蚁群经过 500 次计算后其实验结果如图 5、图 6 所示.

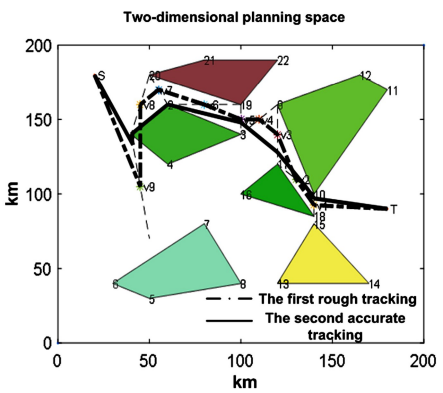

图 5　6 块障碍物融合算法寻迹结果图

Fig.5　Tracking results of fusion algorithm with 6 obstacles

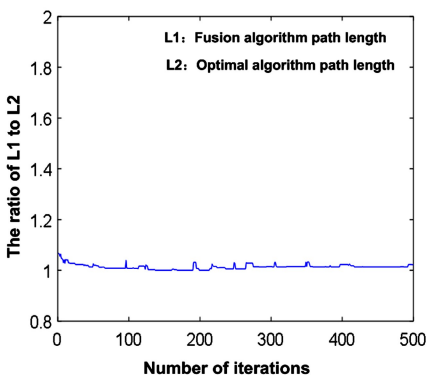

图 6　6 块障碍物两种算法寻迹路径长度对比图

Fig.6　Comparison of the tracing path lengths for two algorithms with six obstacles

寻迹结果显示,Dijkstra 算法所选取的障碍物中点连线已经发生改变,原因是在蚂蚁寻迹路径上增加了一块障碍物,环境障碍物改变时,一次粗略寻迹可自动适应环境的变化,算法鲁棒性得到提升.图 6 显示的路径长度百分比曲线表明,融合算法迭代次数同样在 20 次左右时已逼近最优解,随着迭代的继续进行,百分比曲线有小幅波动,均在可接受误差范围内.

仿真实验 3:在实验 2 所得到的路径基础上,再次增加障碍物,以 23 ~ 27 为顶点的障碍物将 26 顶点伸入已得到的最佳路径中,如图 7 所示.Dijkstra一次粗略寻迹继续自动做出顶点连线选择调整,使寻迹路径继续逼近于最佳路径.图 8 路径长度的百分比曲线显示,融合算法仍可以快速、高效地逼近最佳路径.

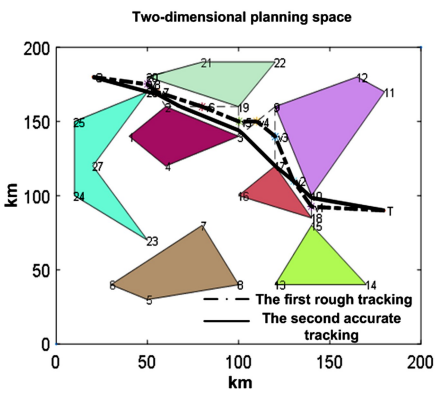

图 7　7 块障碍物融合算法寻迹结果图

Fig.7　Tracing results of fusion algorithm with 7 obstacles

仿真实验 4:在上一次实验的基础上,继续增加障碍物的复杂性,增加寻迹难度,以 28 ~ 31 为顶

点的障碍物用最大限度限制寻迹路径,如图 9 所示.寻迹结果显示,融合算法仍可以在极限中绕过障碍物并靠近最佳路径,且能自适应变化较大的障碍物情况.图 10 所显示路径长度的百分比曲线表明蚁群算法仍可准确、高效的迭代寻迹.

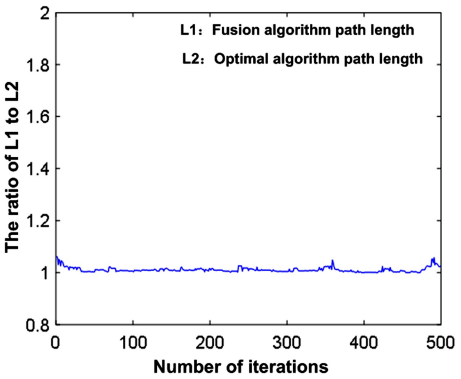

图 8 7 块障碍物两种算法寻迹路径长度对比图

Fig.8 Comparison of the tracing path lengths for two algorithms with 7 obstacles

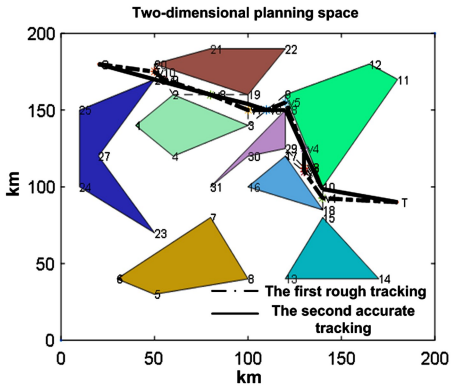

图 9 8 块障碍物融合算法寻迹结果图

Fig.9 Tracing results of fusion algorithm with 8 obstacles

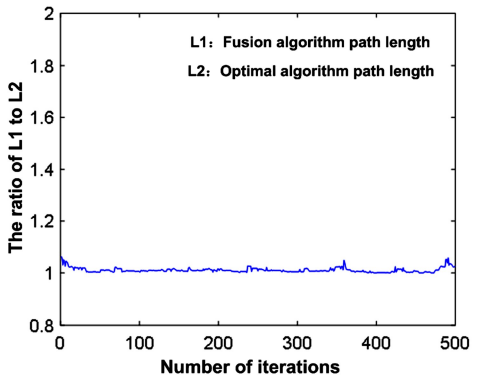

图 10 8 块障碍物两种算法寻迹路径长度对比图

Fig.10 Comparison of the tracing path lengths for two algorithms with 8 obstacles

仿真实验 5:继续检测融合算法的自适应性和鲁棒性,在寻迹路径上增加两块含尖点的障碍物,如图 11 和图 13 所示.寻迹结果显示,融合算法在增加极限尖点障碍物的情况下,均能稳定地绕过尖点,没有发生寻迹偏离和陷入次优解的情况.融合算法的自适应性和鲁棒性,在复杂工况下的综合寻迹得到了很好的发挥.图 12 和图 14 的路径长度的百分比曲线显示,在复杂工况下的寻迹中,随着迭代的进行,路径总长度的收敛情况较简单障碍物的相比出现了小幅波动,但波动幅度均大部分控制在 3%以内,对于复杂情况的智能寻迹结果来说,此波动均在可接受误差范围内.

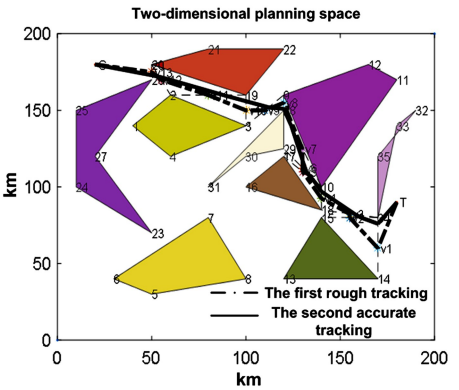

图 11 9 块障碍物融合算法寻迹结果图

Fig.11 Tracing results of fusion algorithm with 9 obstacles

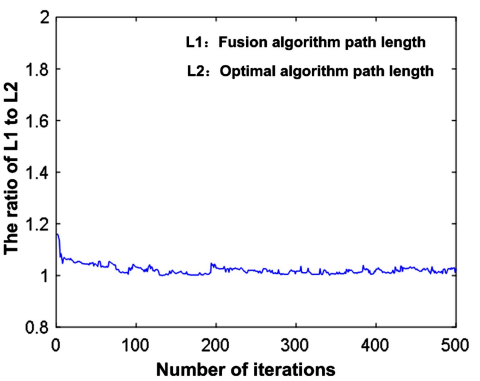

图 12 9 块障碍物两种算法寻迹路径长度对比图

Fig.12 Comparison of the tracing path lengths for two algorithms with 9 obstacles

仿真实验 6:如图 15 所示的复杂环境,改变寻迹的起始点,再次对算法的稳定性进行检验.设置寻迹起点为(20,20),终点为(180,180),算法其他参数不变,寻迹结果如图 15 所示,路径长度的百分比曲线如图 16 所示.

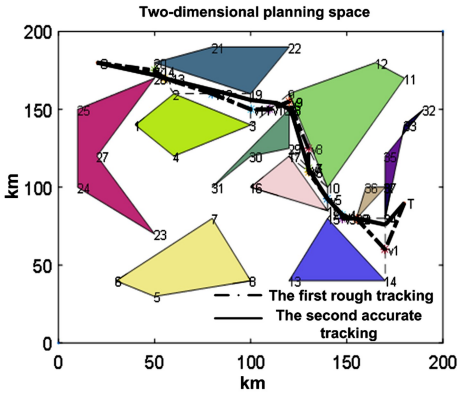

图 13　10 块障碍物融合算法寻迹结果图

Fig.13　Tracing results of fusion algorithm with 10 obstacles

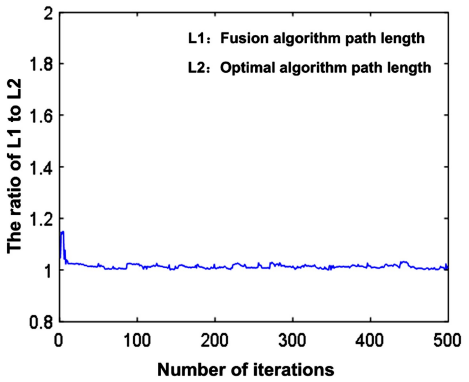

图 14　10 块障碍物两种算法寻迹路径长度对比图

Fig.14　Comparison of the tracing path length for two algorithms with 10 obstacles

图 15 的寻迹结果表明改变起始点后,融合算法仍可以对复杂障碍物环境做出及时反应,图 16 显示在迭代到 40 次左右时,已收敛到最佳路径附近.多次实验的结果表明,融合算法的自适应性和鲁棒性较好,在实际工程中有一定的使用价值.

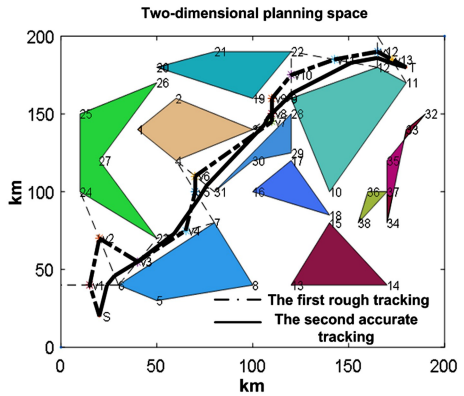

图 15　改变起始点融合算法寻迹结果图

Fig.15　Tracing results of fusion algorithm with the changing starting point

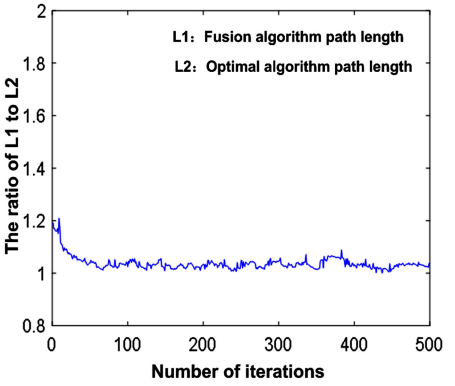

图 16　改变起始点两种算法寻迹路径长度对比图

Fig.16　Comparison of the tracing path length for two algorithms with the changing start point

综合以上各障碍物下的实验,经过优化后的 Dijkstra-蚁群融合算法运行效率大幅提高,并且依然能保持寻迹的高精度.融合算法寻迹路线长度与最优路线长度百分比曲线在探索初期阶段有小幅波动,但随着搜索的进行,波动趋于平缓,最后无限靠近最优路径.对上述 6 种障碍物环境(分别为 5 块障碍物~10 块障碍物的情形),在相同的 Matlab 环境中对 Dijkstra-蚁群融合算法与蚁群算法进行运行时间的对比测试. 蚁群算法参数默认设置为:$\alpha=1,\beta=2,\rho=0.9$,每组不同的障碍物环境测试实验进行 10 次,运行时间取平均值.表 1 和表 2 为 10 次仿真实验运行时间的结果,数据单位为秒(S),表 3 为融合算法与蚁群算法平均运行时间对比表,数据单位为秒(S).

对比表 1 和表 2 可知,改进前蚁群算法的寻迹运行耗时达到 170 秒左右,这对于实际工作中的机器人寻迹系统来说是非常不利的,耗时过长使得工

表 1　蚁群算法寻迹运行时间表

Table 1　Tracing time of ant colony algorithm

Times	Obstacle 1	Obstacle 2	Obstacle 3	Obstacle 4	Obstacle 5	Obstacle 6
1	173.29	169.50	168.92	174.36	175.64	171.49
2	175.51	165.08	169.74	171.21	176.19	175.97
3	172.43	168.28	171.35	177.65	179.27	172.45
4	169.52	170.24	169.66	169.40	177.67	173.78
5	165.32	171.63	167.19	172.71	175.49	171.53
6	171.85	169.27	172.82	172.25	176.20	170.53
7	176.27	169.31	170.27	173.24	178.34	176.73
8	165.28	167.15	169.01	175.54	178.58	171.42
9	177.91	175.80	171.43	172.58	172.17	174.87
10	170.39	166.92	170.55	170.57	177.36	177.04
Average (s)	171.777	169.318	170.094	172.951	176.691	173.581

表 2 融合算法寻迹运行时间表

Table 2 Tracing time of fusion algorithm

Times	Obstacle 1	Obstacle 2	Obstacle 3	Obstacle 4	Obstacle 5	Obstacle 6
1	1.36	1.64	1.92	2.36	2.64	3.12
2	1.21	1.74	1.74	2.21	2.74	3.04
3	1.65	1.49	1.35	2.65	2.49	2.95
4	1.40	1.83	1.66	2.40	2.83	2.66
5	1.71	1.39	1.19	1.71	2.39	3.19
6	1.25	1.50	1.82	2.25	2.50	2.82
7	1.24	1.56	1.27	2.24	2.56	3.27
8	1.54	1.96	1.01	2.54	2.96	3.01
9	1.58	1.92	1.43	2.58	2.92	3.43
10	1.57	1.35	1.55	2.57	2.35	3.05
Average (s)	1.451	1.638	1.494	2.351	2.638	3.054

表 3 融合算法与蚁群算法平均运行时间对比表

Table 3 Average running time of fusion algorithm and ant colony algorithm and the time comparison of two algorithms

	Obstacle 1	Obstacle 2	Obstacle 3	Obstacle 4	Obstacle 5	Obstacle 6
Fusion $T_1(s)$	1.451	1.638	1.494	2.351	2.638	3.054
Ant colony $T_2(s)$	171.777	169.318	170.094	172.951	176.691	173.581
$T_1(s)/T_2(s)$	0.008	0.009	0.008	0.013	0.014	0.017

作效率低,同时也会导致工作任务的失败.改进后的融合算法运行时耗如表 2 所示,复杂环境中的寻迹运行时间降到了 2 秒左右,由表 3 可知,融合算的寻迹时间较蚁群算法缩短了 90%以上,它使得实时寻迹搜索成为可能,在实际工程中有很大的应用价值.这是因为基于 Dijkstra 算法的一次寻迹大幅缩小了搜索范围,与此同时 Dijkstra 的高效一次寻迹也大幅度减少了寻迹时间,由此为蚁群算法二次精确搜索提供了一个节点足够小的节点空间样本,在小样本的情况下,蚁群算法发挥了精确寻迹的特性.两种算法的有机结合使得寻迹过程既高效、又准确,为机器人的实时寻迹提供了一种高效解决方案.

5 结论

本文提出了结合 Dijkstra 算法和蚁群模型迭代求解 MRPP 问题的融合优化算法,兼具 Dijkstra 算法的快速性,和蚁群算法能可靠获取全局最优解的优点.通过算法仿真分析可以得到以下结论:融合优化算法较蚁群算法可以有效地缩短寻迹时间,效率提升可达 90%以上,在移动机器人实际应用中有很好的研究价值.研究蚁群算法的优化问题还处在探索阶段,实验仿真结果显示优化后的算法寻迹准确,效率大幅提升,这在解决移动机器人路径规划问题上有着良好的前景.

参 考 文 献

1 田富洋,曹东,董小宁等. 主被动关节柔性树形机器人系统动力学建模与仿真. 动力与控制学报, 2017,15(1):59~63 (Tian F Y, Cao D, Dong X N, et al. The dynamic modeling and simulation for flexible tree robots system with active-passive joint. *Journal of Dynamics and Control*, 2017,15(1):59~63 (in Chinese))

2 李魁,徐鉴. 压电谐振驱动三足机器人的平面运动. 动力与控制学报, 2015,13(6):454~458 (Li K, Xu J. Locomotion of three-legged vibration-driven robot using piezoelectric actrator. *Journal of Dynamics and Control*, 2015,13(6):454~458 (in Chinese))

3 李士勇,陈用强,李研. 蚁群算法及其应用. 哈尔滨:哈尔滨工业大学出版社, 2004:22~41 (Li S Y, Chen Y Q, Li Y. Ant colony algorithm and its application. Harbin:Harbin Institute of Technology Press, 2004:22~41 (in Chinese))

4 刘建华,杨建国,刘华平等. 基于势场蚁群算法的移动机器人全局路径规划方法. 农业机械学报, 2015,46(9):18~23 (Liu J H, Yang J G, Liu H P, et al. Robot global path planning based on ant colony optimization with artificial potential Field. *Transactions of the Chinese Society for Agriculture*, 2015,46(9):18~23 (in Chinese))

5 王沛栋,冯祖洪,黄新. 一种改进的机器人路径规划蚁群算法. 机器人, 2008,30(6):544~560 (Wang P D, Feng Z H, Huang X. An improved ant algorithm for mobile robot path planning. *Robot*, 2008,30(6):554~560 (in Chinese))

6 康冰,王曦辉,刘富. 基于改进蚁群算法的搜索机器人路径规划. 吉林大学学报(工学版), 2014,44(4):1063~1068 (Kang B, Wang X H, Liu F. Path planning of searching robot based on improved and ant colony algorithm. *Journal of Jilin University:Engineering and Technology Edition*, 2014,44(4):1063~1068 (in Chinese))

7 孙纯哲,林巨广,楼赣菲等. 凹形障碍全局路径规划的双蚁群完全交叉算法. 农业机械学报, 2008,39(7):149~153 (Sun C Z, Lin J G, Lou G F, et al. An double

ant colony algorithm in concave obstacle environment for global path planning. *Transactions of the Chinese Society for Agriculture*, 2008,39(7):149~153 (in Chinese))

8 Shuang B, Chen J P, Li Z B. Study on hybrid PS-ACO algorithm. *Applied Intelligence*, 2011,34(1):64~73

9 刘朝华,张英杰,章兢等. 蚁群算法与免疫算法的融合及其在 TSP 中的应用. 控制与决策, 2010,25(5):695~700 (Liu Z H, Zhang Y J, Zhang J, et al. Using combination of ant algorithm an immune algorithm to solve TSP. *Control and Decision*, 2010,25(5):695~700 (in Chinese))

10 Manabu H, Kosei I, Mitsou W. Adaptive attitude control with Lyapunov stability of nonholonomic space robot under the unstructured dynamics. Transaciton of the Japan Society of Mechanical Engineers:Part C, 2005,71(708):2586~2592

11 张银玲,牛小梅. 蚁群算法在移动机器人路径规划中的仿真研究. 计算机仿真, 2011,28(6):231~234 (Zhang Y L, Niu X M. Simulation research on mobile robot path planning based on ant colony optimization. *Computer Simulation*, 2011,28(6):231~234 (in Chinese))

12 金飞虎,高会军,钟啸剑. 自适应蚁群算法在空间机器人路径规划中的应用. 哈尔滨工业大学学报, 2010,42(7):1015~1018 (Jin F H, Gao H J, Zhong X J. Research on path planning of robot using adaptive ant colony system. *Journal of Harbin Institute of Technology*, 2010,42(7):1015~1018 (in Chinese))

A RAPID PATH PLANNING ALGORITHM FOR MOBILE ROBOT WITH IMPROVED ANT COLONY ALGORITHM*

Tan Huisheng[1†]　Liao Wen[2]　He Xunyu[3]

(1.*School of Traffic Engineering*, *Hunan University of Technology*, *Zhuzhou*　412007, *China*)

(2.*School of Electrical and Information Engineering*, *Hunan University of Technology*, *Zhuzhou*　412007, *China*)

(3.*The Graduate School*, *Hunan University of Technology*, *Zhuzhou*　412007, *China*)

Abstract　Dijkstra algorithm has been widely used for the mobile robot path planning(MRPP). However, under the complex conditions it can not make sure that the results are correct or optimal. Though with the ant colony algorithm, the mobile robot path planning model can give the global optimal solution under certain conditions, but it needs a long time for the solution. Therefore, a fusion algorithm combining Dijkstra algorithm with ant colony algorithm is proposed. This hybrid algorithm contains the advantage of Dijkstra and ant colony algorithm to obtain the optimal MRPP in shorter time. Firstly, a suboptimal solution of the shortest path can be found with the Dijkstra fast algorithm when the robot is working. Secondly, near the suboptimal solution the working environment is accurately divided. Finally, the shortest path can be gained by the ant colony algorithm. From the simulation results, more than 90% of the tracing time of the fusion algorithm can be reduced when compared with ant colony algorithm. It is apparent that this fusion optimization algorithm not only reduces the time consumption of the ant colony algorithm, but also obtains the reliable global optimal solution.

Key words　mobile robot, path planning, Dijkstra algorithm, ant colony algorithm, fusion optimization algorithm

Received 29 December 2017, revised 19 May 2018.

* The project supported by the National Natural Science Foundation of China(6167224) and the Natural Science Foundation of Hunan Province of China(2016JJ6036).

† Corresponding author E-mail:huisheng21nd@163.com