doi: 10.3969/j.issn.0490-6756.2020.04.013

基于记忆矩阵 A* 引导域的 VFH 算法改进策略

周　俊，庄宇辉，严　华
(四川大学电子信息学院，成都 610065)

摘　要：针对机器人采用 VFH 算法避障容易陷入局部死区的问题，提出为机器人增加记忆地图的方法. 在面对未知环境时，采用了一种新的自适应阈值策略，在一定程度上避开局部死区抵达目标点，同时在首次避障时生成记忆地图. 在二次避障过程中通过记忆地图产生低分辨率 A* 引导域，使用改进的代价函数，使 A* 引导域与 VFH 算法有效结合，以较优的路径完成避障过程，同时能实时适应周围环境的变化. 最后，在 MATLAB 上针对不同环境进行算法仿真对比. 实验结果表明，使用新的自适应阈值策略能在未知环境下完成避障过程并产生记忆地图；在记忆地图的基础上使用 A* 域的引导，能使机器人适应周围环境变化的同时以较优的路径顺利到达目标位置.

关键词：VFH；记忆地图；A* 引导域；自适应阈值；代价函数
中图分类号：TP312　　**文献标识码**：A　　**文章编号**：0490-6756(2020)04-0704-07

An improved strategy of VFH algorithm based on memory matrix and A* guided domain

ZHOU Jun, *ZHUANG Yu-Hui*, *YAN Hua*
(College of Electronics and Information Engineering, Sichuan University, Chengdu 610065, China)

Abstract: To solve the problem of easily falling into the local dead zone when the robot avoids obstacles with VFH algorithm, a method is proposed to add memory maps for robots. When facing the unknown environment, the improved strategy of using adaptive threshold is used to avoid the local dead zone and reach the target point, meanwhile, a memory map is generated on which a low-resolution A* guided domain is then produced and the cost function is improved to effectively combine the A* guided domain with the VFH algorithm to complete obstacle avoidance with a better path and adapt to changes in the surrounding environment in real time. Finally, the algorithm is verified in different environments. The experimental results show that the strategy of using adaptive threshold can complete obstacle avoidance and generate memory map in unknown environment; based on memory map and the guidance of the A* domain, the robot can adapt to the changes of the surrounding environment and smoothly reach the target.

Keywords: VFH; Memory map; A* Guided Domain; Adaptive threshold; Cost function

收稿日期：2019-05-10
基金项目："973"计划科研项目(2013CB328903)
作者简介：周俊(1994－)，女，重庆江津人，硕士研究生，研究方向为智能控制. E-mail：894712466@qq.com
通讯作者：严华. E-mail：4429175@qq.com

1　引　言

机器人的自主避障问题一直是一个热点研究问题. 针对此问题，目前已有许多成熟的理论，如可视图法[1]、栅格法[2]、虚拟势场法 VFF(Virtual Force Field)[3] 等，向量场直方图 VFH(Vector Field Histogram)[4]算法亦是其中之一.

VFH 算法是由密歇根大学的 Johann Borenstein 和 Yoram Koren 于 1991 年针对 VFF 算法易陷入局部最小以及震荡等问题改进而来，但该算法未考虑机器人的机械特性和运动特性，因此该团队提出了 VFH+算法[5]进行改进，而后又结合启发式函数提出了 VFH* 算法[6]，使机器人在运动过程中更趋于目标方向. 此后，有许多国内外学者在此基础上对 VFH 系列算法进行了改进，包括阈值敏感问题[7]、直方图构建问题[8]、动态环境问题[9]等.

但是，针对使用 VFH 算法避障易陷入局部死区的问题，一直没有很好的解决办法. VFH* 算法提出使用启发式函数引导机器人选择方向，可在一定程度上避开死区，但随之而来的是每次方向选择时时间代价的大量增加. 文献[7]提出了自适应阈值的方法，同样能在一定程度上避开死区，但也增加了时间代价. 此外，以上方法能避开死区的前提是死区范围在机器人可视范围内，如果死区深度较大，机器人依然会不可避免地陷入死区. 因此，本文提出了一种增加机器人“记忆”的方法，使用改进阈值策略的 VFH 算法的机器人在首次避障过程中将周围的环境以一个较低分辨率的矩阵存储下来. 在此基础上使用 A* 算法[10]生成 A* 引导域，结合 VFH 算法，提出新的代价函数，使机器人在此后的避障过程中能提前避开在可视范围之外的死区，同时能够适应周围环境的动态变化.

2　算法设计

2.1　VFH 避障算法

VFH 算法将求出的环境障碍物强度值以极坐标的形式转化到机器人坐标系下生成向量场直方图，该强度值与障碍物和机器人的距离成正比. 选取合适的阈值，小于该阈值的障碍物强度值组成的连续区域为候选区域，在候选区域中选择合适的方向作为机器人的前进方向，在下一个位置重复以上行为，直至抵达目标点. 该算法的不足之处在于阈值大小的设置直接影响候选方向的选择，当阈值较大时，机器人可选方向较多，但容易陷入死区，如图 1(a)所示；当阈值较小时，机器人能提前发现死区，从而提前转向，如图 1(b)所示.

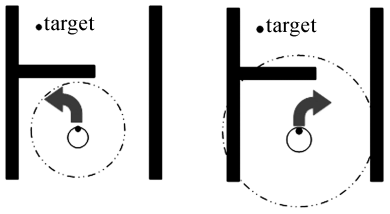


图 1　死区环境方向选择示意图

Fig. 1　Schematic diagram of direction selection in dead zone environment

但是，当目标点与障碍物距离较近时，若选择较小阈值，机器人无法靠近目标点，如图 2 所示.

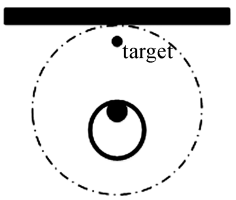


图 2　目标点环境示意图

Fig. 2　Schematic diagram for a target environment

因此，采用合适的阈值选择策略对机器人的避障过程有极为重要的影响. 文献[11]对阈值的选择策略进行了改进，使用动态阈值的方法，当机器人与目标点间存在障碍物时，以最小阈值获得机器人的目标航向；当机器人与目标点间不存在障碍物时，使用以最大阈值获取目标航向，在一定程度上能达到自适应的效果. 但在图 2 目标环境下，该文献中的方法仍设置固定阈值，会因为自适应程度不够导致机器人无法到达目标点.

2.2　改进的阈值策略

结合以上分析，本文提出一种新的自适应阈值策略，根据机器人所处环境阶段的不同，分为发现目标点前和发现目标点后两种情况.

(1) 发现目标点前的策略.

① 设置机器人最小阈值 $H_{th\min}$ 和最大阈值 $H_{th\max}$.

② 以一定步长 Δth 对此范围内的阈值进行遍历，对应每一个阈值，使用改进的 VFH 代价函数计算候选方向，如下式.

$$h(c)=\frac{1}{H_{th\max}-k*H_{th}}*g(c) \tag{1}$$

其中，$g(c)=\mu_1\Delta(c,k_t)+\mu_2\Delta(c,k_{pre})$，对应不同候选方向的代价值，$\Delta(c,k_t)$ 表示候选方向与目标方向的角度差，$\Delta(c,k_{pre})$ 表示候选方向与当前前进方

向的角度差. μ_1,μ_2 需满足 $\mu_1>\mu_2$,μ_1 越大,机器人选择更偏向目标方向的候选方向的可能性更高;μ_2 越大,机器人选择偏向目标方向的可能性更高.

由于在阈值越大的情况下,候选方向越多,更容易找到偏向目标的方向,因此计算出的 $g(c)$ 值越小,若直接根据不同阈值下的 $g(c)$ 值进行方向选择,机器人总会选择阈值较大情况下的前进方向. 因此提出 $g(c)$ 前面的自适应阈值系数项,当阈值越大时,该系数越大,$h(c)$ 越大;当阈值越小时,该系数越小,$h(c)$ 越小. 以此达到在可行条件下更倾向选择较小阈值下的候选方向的目的. $k\in[0,1)$ 为安全指数,k 越大,系数项的作用越明显,当 $k=0$ 时,系数项成为固定值,$h(c)$ 退化为 $g(c)$. 在较宽阔环境下,可选择较大的 k 值;在障碍物较多、较密集的环境下,可选择较小的 k 值.

③ 根据式(1)计算出机器人在当前位置下所有阈值及候选方向对应的代价值 $h(c)$ 后,选择出 $h(c)$ 值最小的方向作为前进方向.

(2) 发现目标点后的策略. 采用动态阈值 H_t. 根据机器人距离目标点的距离 d_t 求出对应障碍物强度.

$$H_t=cv^2(a-bd_t^2) \tag{2}$$

式中,随着 d_t 的不断变化,H_t 也将不断改变,此值作为动态阈值可使机器人顺利到达目标点.

2.3 记忆地图的生成

2.2 节对 VFH 算法的阈值选择策略进行了改进,可以在一定程度上提前发现死区并进行规避,但若环境中死区的深度较大,在机器人的可视范围之外,那么机器人无法及时转向,仍有很大概率选择死区方向,如图 3 所示.

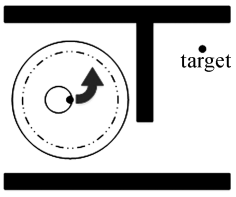


图 3 机器人方向选择示意图

Fig. 3 Schematic diagram of direction selection

基于此种情况,本文将全局环境路径规划和动态避障相结合,提出"记忆矩阵"的概念,让机器人在首次避障时将周围的障碍物信息用一个较低分辨率的矩阵存储下来,在二次避障过程中提前进行全局规划,对后续避障过程起到引导作用.

(1) 障碍物点的坐标转换. 机器人观察到的障碍物点(x,y)是相对于机器人坐标系的,需将其转化到世界坐标下,坐标转换图如图 4 所示.

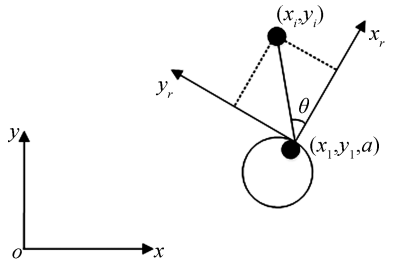


图 4 坐标转换示意图

Fig. 4 Schematic diagram of coordinate transformation

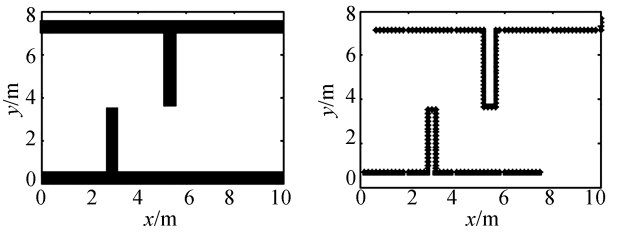
设机器人的世界坐标为(x_1,y_1,α),该障碍物在机器人坐标系下的坐标为(x_i,y_i),世界坐标下的坐标为

$$\begin{cases}x_w=x_1+R\cos(\theta+\alpha)\\ y_w=y_1+R\sin(\theta+\alpha)\end{cases} \tag{3}$$

$$R=\sqrt{x_i^2+y_i^2},\theta=\arctan(y_i/x_i) \tag{4}$$

其中,x_w 为障碍物点在世界坐标下的横坐标;y_w 为障碍物坐标点在世界坐标下的纵坐标;α 为当前机器人在世界坐标下的位姿角.

(2) 记忆地图的创建. 对应实际的世界坐标系创建一个大小为 $m*n$ 的矩阵,如图 5 所示. 相对实际距离的缩放比例为 s,将转换后的障碍物坐标点按比例缩放后存入对应的矩阵栅格中.



(a) 原始地图 (b) 记忆地图

图 5 记忆地图示意图

Fig. 5 Memory map

其中,s 的大小直接影响后续全局路径规划的结果. s 较小时,地图分辨率较高,规划出的路径就越精确,但耗时越长;s 较大时,地图分辨率越低,规划出的路径较粗糙,但耗时越短. 因此,需要选择一个合理的取值 s,平衡规划路径好坏与时间的关系.

如图 5(a)所示,原始地图为 10 m * 8 m 的二维地图,对应分辨率为 500 * 500,当 $s=10$ 时,得到如图 5(b)所示的记忆地图.

2.4 低分辨率 A* 引导域生成

根据机器人的记忆地图,以及起始点与目标点的坐标,可提前进行全局路径规划[12]. 在众多的路

径规划算法，包括无人机航路规划[13]、智能车路径规划算法中，A* 算法能保证在地图中找到最短路径，经过冗余节点去除后，生成 A* 引导域[14]以引导机器人的避障过程.

(1) A* 寻路算法. A* 算法是在 Dijkstra 算法的基础上发展起来的，是一种启发式搜索算法，结合了 Dijkstra 算法和最佳优先搜索(Best-First Search，BFS)算法的优点，用于在全局静态路径规划中寻找最短路径. A* 算法通过一个代价函数来确定搜索方向，从起点开始向周围扩展，通过估价函数计算得到周围每个节点的代价值，选择最小代价节点作为下一个扩展节点，重复这一过程直到抵达目标点，生成最终路径. 其代价函数表达式为

$$f(n)=g(n)+h(n) \tag{5}$$

式中，$f(n)$是从起始点经由当前点 n 到目标点的移动代价估计；$g(n)$是起始点当前点 n 的实际移动代价值；$h(n)$是当前点 n 到目标点的估计代价值.

本文采用欧几里得距离度量两点之间的移动代价，如下式所示.

$$d=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2} \tag{6}$$

其中，(x_1,y_1)、(x_2,y_2)分别表示两节点的坐标. 对于 $g(n)$，(x_2,y_2)为当前节点时，(x_1,y_1)为起始节点；对于 $h(n)$，(x_2,y_2)为当前节点时，(x_1,y_1)为目标节点.

A* 算法在进行搜索时会创建一个 open 表和一个 close 表，当到达某一节点时，该节点会被添加到 open 表中，选择 open 表中代价最小的点作为下一个扩展节点，同时将该节点加入到 close 表中，重复该过程，直到目标节点被添加到 open 表中，此时 close 表中的路径为最终路径.

(2) 碰撞检测法去冗余. A* 算法在搜索过程中是基于固定大小栅格的，最终的路径只能保证栅格相加路径最短，但可能存在大量折点，因此，本文采用碰撞检测法对路径进行去冗余处理. 去冗余的思想为：对于路径中的每一个点，依次连接后面的点，进行碰撞检测，如果连接点间没有障碍物，则删除冗余节点，如图 6 所示.

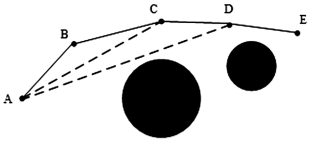


图 6　去冗余节点示意图

Fig. 6 Schematic diagram of removing redundant nodes

假设路径中有 A、B、C、D、E 共 5 个节点，连接 (A,C)，检测是否发生碰撞，没有则删除 B 节点；连接 (A,D)，检测是否发生碰撞，没有则删除 C 节点；连接 (A,E)，检测发生了碰撞，则开始从 D 节点进行搜索，直到删除所有冗余节点.

(3) 低分辨率 A* 引导域. A* 算法对地图进行路径规划时，先将地图栅格化，栅格化大小代表了地图分辨率的不同. 在起始点和目标点相同的情况下，当地图分辨率越高时，机器人规划的步数越多，相应的时间也越长；当地图分辨率越低时，机器人规划的步数越少，相应的时间也越短. 针对图 7 所示的环境，分别进行 260 * 200、120 * 100、60 * 50 和 40 * 35 的栅格化，规划结果分别如图 7 中(a)、(b)、(c)和(d)所示.

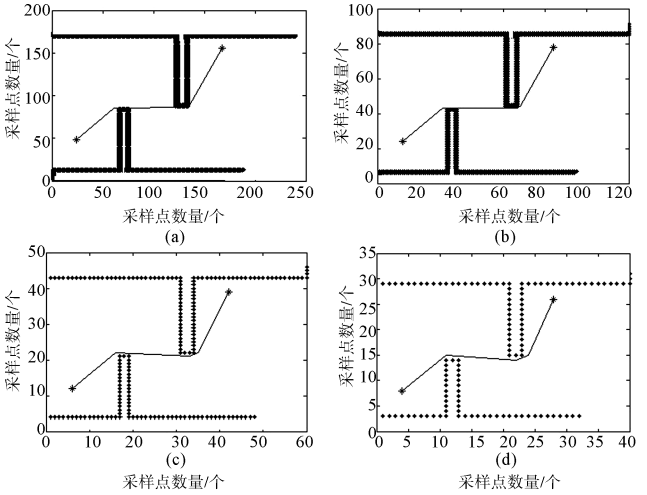


图 7　不同分辨率地图下 A* 算法规划结果

Fig. 7　A* algorithm planning results

表 1　不同分辨率地图下 A* 算法规划时间

Tab. 1　A* algorithm planning time

不同分辨率种类	图 7 (a)	图 7 (b)	图 7 (c)	图 7 (d)
时间/s	134.176	12.433	1.9188	0.79566

由图 7 和表 1 中结果可知，在不同分辨率下 A* 算法规划路径趋势基本相似，但时间却成倍数增长. 因此本文采用较低分辨率下的 A* 规划结果作为引导域，但前提是保证障碍物轮廓形态不变且满足以下公式.

$$d/f<S_{\min} \tag{7}$$

式中，d 表示地图中的实际距离；f 表示采样点数量，对应图像分辨率；$S_{\min}$ 表示机器人最小可通行通道宽度. 对地图进行低分辨率处理时，其采样间

距需小于机器人的最小可通行通道宽度.

2.5 机器人的方向选择

结合 A* 引导域和 VFH 算法,对机器人在每一步进行方向选择时的多个候选方向提出改进的代价函数,将 A* 引导域的每一个局部目标点加入到代价函数中. 定义代价函数表达式为:

$$g(c)=\mu_1\Delta(c,k_t)+\mu_2\Delta(c,k_{sub\_t})+\mu_3\Delta(c,k_{pre}) \tag{8}$$

式中,$\Delta(c,k_t)$为候选方向与全局目标方向的夹角;$\Delta(c,k_{sub_t})$为候选方向与局部目标方向的夹角,局部目标为离机器人当前位置最近的 A* 引导域的目标点;$\Delta(c,k_{pre})$为候选方向与机器人当前行驶方向的夹角. μ_1,μ_2,μ_3 均为常数,为保证机器人向目标点前进,需满足 $\mu_1>\mu_2,\mu_2>\mu_3$. μ_1 越大,机器人选择更偏向目标方向的候选方向的可能性更高;μ_2 越大,机器人选择偏向局部目标方向的可能性更高;μ_3 越大,机器人选择与当前行驶方向最接近的方向的可能性越高. 在本文的实验中,选择 $\mu_1=7,\mu_2=6,\mu_3=4$.

3 实 验

为验证本文所提出方法的有效性,在配置 Intel(R) Core(TM) i5-3460 CPU @ 3.20 GHz 处理器,4.00 GB RAM 台式机中使用 MATLAB R2014a 进行仿真.

仿真环境是机器人容易陷入死区的 10 m * 10 m的不同场景二维地图,VFH 算法各固定参数设置如表 2 所示.

表 2 VFH 算法参数

Tab.2 VFH algorithm parameters

参数名称	参数值	参数说明
$d_{\max}$	3.5	机器人视野
f	5°	角分辨率
b	2.5	常数
cv	10	常数
$d_{\min}$	0.6	机器人安全距离

3.1 自适应阈值避障结果

此阶段机器人无记忆地图信息,采用自适应阈值策略进行避障. 首先在一种极易陷入死区的地图环境下进行实验,自适应阈值算法各参数为:$H_{th\min}=600$,$H_{th\max}=2800$,$k=0.9$,$\Delta th=1100$. 在场景 1 下实验结果如图 8 所示.

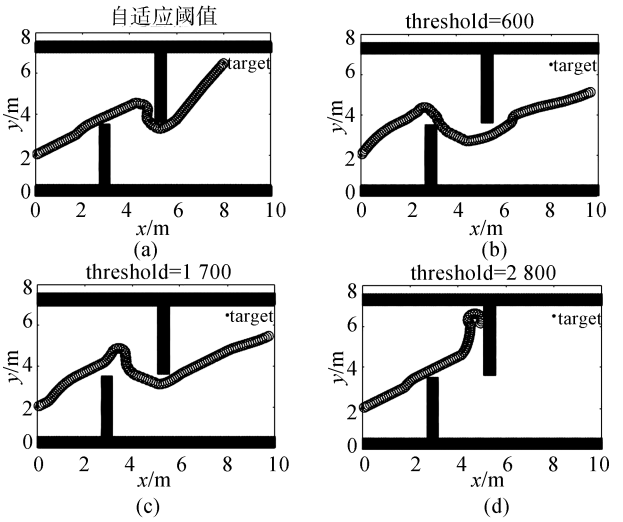


图 8 避障结果(场景 1)

Fig.8 Obstacle avoidance results in scenario 1

图 8 是在场景 1 环境下采用本文提出的自适应阈值策略和采用较小阈值、中间阈值和较大阈值的实验结果对比图. 可以看出,采用自适应阈值策略时,机器人会根据不同的环境综合考虑不同阈值下的方向,从中选出最好的可行方向,因此可以较好地避开死区,且能顺利抵达距离障碍物较近的目标点. 采用较小固定阈值时,虽能避开死区,但由于目标点离障碍物太近,目标方向不在可通行扇区内,无法抵达终点. 采用中间阈值时,机器人发现死区的时机比前者晚,因此转弯时机稍晚,且同样由于目标点离障碍物太近,目标方向不在可通行扇区内,无法抵达终点. 采用较大固定阈值机器人无法提前发现进入死区,陷入徘徊,无法抵达终点.

为了验证算法的有效性,接下来在一种障碍物更多更复杂的环境下进行实验,如图 9 所示.

图 9 是在场景 2 环境下采用本文提出的自适应阈值策略和采用较小阈值、中间阈值和较大阈值的实验结果对比图. 从实验结果可以看出,在存在狭小通道且障碍物较密集的情况下,采用自适应阈值策略可以顺利抵达目标点. 采用较小固定阈值时,机器人容易忽略狭小的可行通道从而错失抵达目标点的机会. 采用中间阈值进行避障时,机器人虽然经过一番徘徊找到了狭小的可行通道,但最终由于障碍物太过密集因此陷入徘徊,未能抵达终点. 采用较大阈值进行避障时,机器人可以顺利抵达目标点,与自适应阈值的路径相似;这也说明了在自适应阈值策略中,机器人经过决策在此环境下自适应地选择选择了较大阈值进行避障.

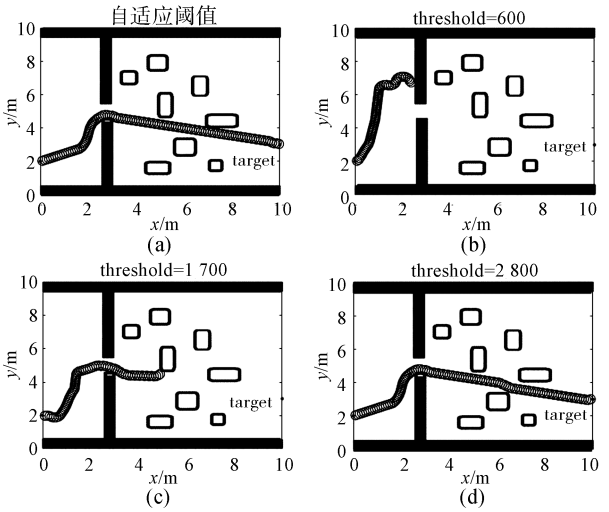


图 9　避障结果(场景 2)

Fig. 9　Obstacle avoidance results in scenario 2

上述结果表明，采用本文提出的自适应阈值策略可使机器人在一定程度上避开死区，且能适应障碍物密集的区域，顺利到达目标点，自适应能力较强.

3.2　基于 A* 域引导的避障结果

使用自适应阈值进行首次避障产生记忆地图后，采用低分辨率 A* 算法的规划结果作为引导域，并在不同环境下进行实验对比. 图 10 是机器人在场景 1 下增加 A* 引导域的前后对比结果. 由图中结果可知，增加 A* 引导域后机器人能够更早地避开死区，提前转弯，以更优的路径到达目标点.

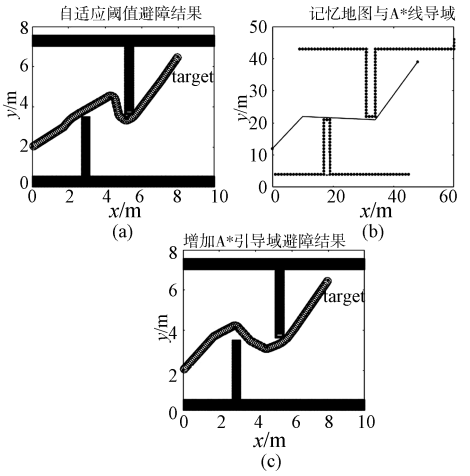


图 10　避障结果(场景 1)

Fig. 10　Obstacle avoidance results in scenario 1

为进一步验证增加 A* 引导域后的算法的有效性，接下来进一步增加死区深度进行实验对比，实验结果如图 11 所示. 同时，为了验证算法适应动态环境的能力，在场景 3 的基础上动态增加障碍物进行实验对比，如图 12 所示.

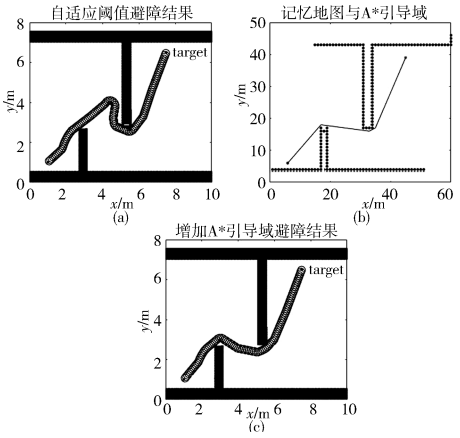


图 11　避障结果(场景 3)

Fig. 11　Obstacle avoidance results in scenario 3

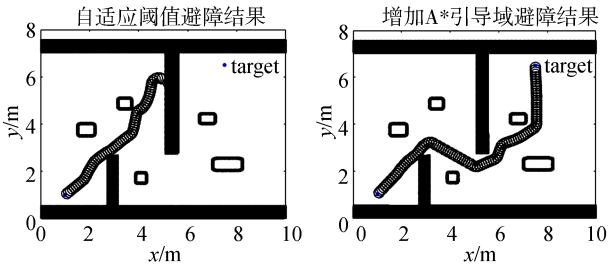


图 12　动态增加障碍物后的避障结果

Fig. 12　Obstacle avoidance results after dynamically increasing obstacles

由图 11 可知，在首次避障过程中单独使用自适应阈值进行避障时，由于死区深度较大，受目标方向的影响，机器人会优先选择死区方向，然后再返回，从而出现一个较大的折点. 增加 A* 引导域后，机器人能够根据引导域的指引，提前转弯，以更短更优的路径抵达目标点. 由图 12 结果可知，当动态增加障碍物后不使用 A* 引导域进行二次避障时，由于死区深度过大和障碍物环境过于复杂，因此即使是自适应程度较高的机器人也会陷入死区；当二次避障过程中采用 A* 引导域后，机器人既能够避开死区，也能够动态地避开障碍物到达目标点. 到达目标时的路径点数如表 3 所示. 路径点数为“－”表示机器人未到达目标，路径点数越少表示路径越短.

表 3　算法效果对比表

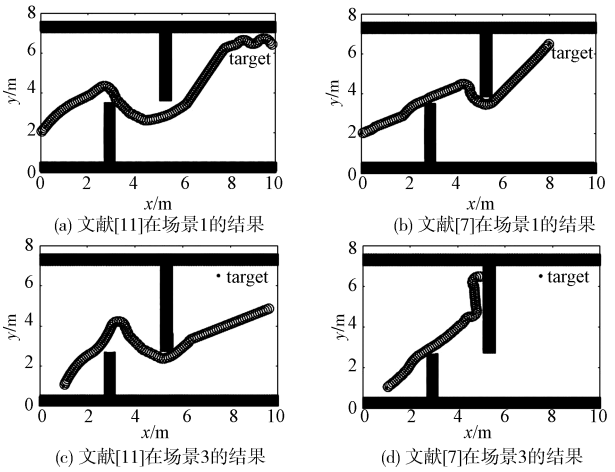
Tab. 3　Comparison of algorithm effect

图	无 A* 引导域	有 A* 引导域
图 10	111	107
图 11	115	100
图 12	－	110

结合表 3 及上述实验结果图可知，增加 A* 域的引导可以使机器人到达目标点的路径更短. 同

时,在动态新增障碍物的环境下,使用 A* 域引导可使机器人提前转向避开死区,且保留了 VFH 算法实时动态避障的特性,最终以较短的路径到达目标.

为进一步验证算法的先进性和有效性,接下来与文献[7]与文献[11]中提出的改进算法进行对比,分别在场景 1、场景 3 下进行实验.



(a) 文献[11]在场景1的结果 (b) 文献[7]在场景1的结果

(c) 文献[11]在场景3的结果 (d) 文献[7]在场景3的结果

图 13 避障结果

Fig. 13 Obstacle avoidance results

由图 13 结果可知,文献[11]的算法策略在场景 1 和场景 3 中均能避开死区,但无法抵达目标点.这是由于该算法在目标点附近存在障碍物时仍使用固定阈值,自适应程度不够,目标点不在可行通道内,导致其无法接近目标点.文献[7]的算法策略在场景 1 下与本文提出的自适应阈值策略避障结果相近,但在场景 3 中,由于死区深度加大,该阈值策略没有及时选择避开死区的方向,使其最终陷入死区,说明该算法存在一定局限性.

4 结 论

针对机器人采用 VFH 算法避障容易陷入局部死区的问题,在首次避障过程中,采用自适应阈值策略进行避障,同时生成记忆地图.在记忆地图的基础上采用低分辨率 A* 算法规划结果作为机器人二次避障过程中的引导域.仿真结果证明,使用该方法能够使机器人以较优的路径避开死区完成避障过程并且能够实时应对环境的变化.

参考文献:

[1] Hofner C, Schmidt G. Path planning and guidance techniques for an autonomous mobile cleaning robot [J]. Intell Robot Syst, 1995, 14: 241.

[2] Micha. Robot motion planning[J]. Commun Pur Appl Math,1995,18: 1173.

[3] Khatib O. Real-time obstacle avoidance for mani- pulators and mobile robots[J]. Int J Robot Res,1986, 5: 90.

[4] Borenstein J, Koren Y. The vector field histogram-Fast obstacle avoidance formobilerobots[J]. IEEE T Robot Autom, 1991, 7: 278.

[5] UIrich I, Borenshtein J. VFH+: Reliable obstacle avoidance for fast mobile robots[J]. IEEE Int Conf Robot Autom, 1998, 1998: 1572.

[6] UIrich I, Borenshtein J. VFH *: local obstacle avoidance with lookahead verification[J]. IEEE Int Conf Robot Autom, 2000, 2000: 2505.

[7] 庄宇辉, 赵成萍, 严华. 一种针对 VFH 系列算法阈值敏感问题的改进策略[J]. 四川大学学报:自然科学版, 2018, 55: 95.

[8] 向亚军, 严华. 基于激光雷达的移动机器人避障策略研究[J]. 四川大学学报:自然科学版, 2017, 54: 529.

[9] 章苏书, 吴敏, 曹卫华. 一种局部动态环境下的避障算法[J]. 计算技术与自动化, 2003, 22: 12.

[10] Pal A, Tiwari R , Shukla A. Modified A* algorithm for mobile robot path planning[M]// Soft Computing Techniques in Vision Science. Berlin: Springer Berlin Heidelberg, 2012.

[11] 刘杰, 闫清东, 唐正华. 基于激光雷达的移动机器人避障规划仿真研究[J]. 计算机工程, 2015, 41: 306.

[12] 吴麟麟, 杨俊辉, 汪若尘, 等. 基于混合 SA 算法的智能汽车全局路径规划[J]. 江苏大学学报:自然科学版, 2019, 40: 249.

[13] 史红玉, 刘淑芬. 基于 Voronoi 图的无人机航路改进规划[J]. 吉林大学学报:理学版, 2018, 56: 945.

[14] 冯来春, 梁华为, 杜明博, 等. 基于 A* 引导域的 RRT 智能车辆路径规划算法[J]. 计算机系统应用, 2017, 26: 127.

引用本文格式:

中 文: 周俊, 庄宇辉, 严华. 基于记忆矩阵 A* 引导域的 VFH 算法改进策略[J]. 四川大学学报: 自然科学版, 2020, 57: 704.

英 文: Zhou J, Zhuang Y H, Yan H. An improved strategy of VFH algorithm based on memory matrix and A* guided domain [J]. J Sichuan Univ: Nat Sci Ed, 2020, 57: 704.