doi: 10.7690/bgzdh.2018.04.022

多关节鱼水中救援策略

刘艳红，陈伟豪，张一大
（山西农业大学信息科学与工程学院，山西 太谷 030801）

摘要：针对多关节鱼水中救援项目，在单关节鱼的基础上，对多关节鱼在尽可能短的时间内完成救援任务提出了一种救援策略。策略在多关节鱼能够全部撞击到目标物的前提下，保证了在规定时间内即使多关节鱼未能完成任务，也可以取得一个相对高效的救援效果。策略中设计出了迪杰斯特拉路径规划方案的救援路线，采用了直接撞击函数和鱼头甩动函数相结合的救援方法。实验结果表明：该策略能够灵活调用多关节鱼，大大缩短了救援用时，增加了策略的稳定性，提高了救援效率，在实际应用中取得了较好的效果。

关键词：多关节鱼；最短路径；鱼头甩动；水中救援
中图分类号：TP242　　**文献标志码：**A

Strategy of Multi-joint Fish Underwater Rescue

Liu Yanhong, Chen Weihao, Zhang Yida
(*College of Information Science & Engineering*, *Shanxi Agricultural University*, *Taigu* 030801, *China*)

Abstract: In the multi-joint fish underwater rescue project, on the basis of the simple-joint fish, a rescue strategy is put forward. By this strategy the multi-joint fish can accomplish the rescue mission within the shortest possible time. In the case of hitting the entire target, strategy can achieve relative high rescue efficiency, though the multi-joint fish failed to finish the task within the allotted time. In this strategy, the rescue route plan scheme on the basis of Dijkstra thought is designed. Use the combination method of the direct impact function and the fish head swinging function. The experiment results show that this strategy can call the multi-joint fish flexibly. It shortens the total time of rescue obviously, increases the stability and improves the rescue efficiency. The strategy has achieved good result in practical application.

Keywords: multi-joint fishs; the shortest path; the fish head swinging; underwater rescue

0　引言

近年来，我国水中机器人发展迅猛。2016 年，哈尔滨工程大学乔钢创新水下通信方法，将水波通信的方式扩展到全双工、多用户。全局视觉的水中救援作为水下机器人的一个研究方向，目前的策略主要有曹培等提出的弦端点法控制算法[1]，周伟诚、夏庆锋[2]提出的关于单关节鱼的鱼尾甩动算法。在原有单关节鱼水中救援算法上，文中实现了多关节鱼能够全部撞击到目标物，在规定时间内如果多关节鱼未能完成任务，策略也可以取得一个相对高效的救援效果；因此，笔者从救援路线规划和撞击算法上进行了设计与改进，规划出了水中救援的最短路径，改进了鱼头甩动函数 Hammer，同时提出了捕获目标点的策略。该方法在一定程度上有效地提高了救援速度。

1　项目内容

如图 1 中所示，水中救援场地包括电脑一台、长方形水池、水池架，摄像头、无线发射模块、多关节鱼、操作软件。其中操作软件包括 MURobotFish – V1.1 – x64（大平台）、Fish Controller – V1.0.4 – x86（小平台）[3]。

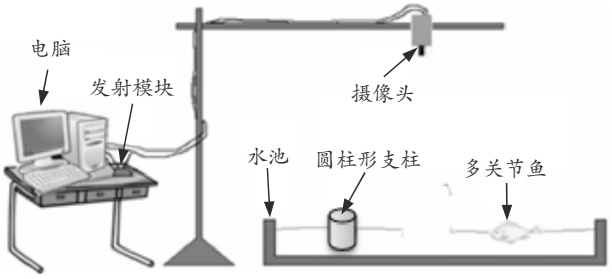

图 1　水中救援场地

圆筒形支柱（以下简称支柱）由笔筒（直径 8 cm±0.5 cm，高度 19 cm±1 cm）制成，根据支柱难度分为 3 个等级，最低难度的支柱由接口不进行固定的 2 个笔筒叠制而成，中等难度的支柱由接口粘合的 2 个笔筒叠制而成，最高难度的支柱由笔筒和配重物构成。5 个笔筒上方均安放规格大小相同的盒子[3]。

收稿日期：2017-12-20；**修回日期：**2018-02-01
作者简介：刘艳红（1979—），女，山西人，硕士，从事 VC++编程及计算机图像处理、2D 仿真类算法研究。

2　项目要求

水池中安放 5 个圆筒形支柱，其中编号 1、2、3 的支柱为最低难度的支柱，编号 4 为中等难度的支柱，编号 5 为最高难度的支柱。救援人员采用控制平台进行图像处理和目标识别，应用策略驱动多关节鱼逐一冲撞支柱，使得支柱顶部的盒子落入水中，从而实现救援目的，如图 2 所示。

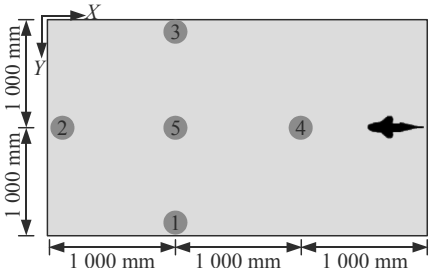

图 2　全局视觉水中救援示意图

假设 B_i 为各个目标点，i=1, 2, 3, 4, 5；“–”表示到达。由此可得目标点之间的距离：B_1–B_5=1 m，B_2–B_5=1 m，B_5–B_4=1 m，B_5–B_3=1 m。

救援人员要在 5 min 内完成 5 个支柱的救援，因此救援人员要根据实际情况设计救援路线，当多关节鱼到达支柱时要通过相应的救援策略撞击支柱，用时越短，效率越高。如果救援过程超过 5 min，则救援暂停，以救援支柱的难易度来评价救援的效果。

3　策略实现

3.1　系统运行模式

应用策略驱动的前提是必须获取支柱和机器鱼的位置信息，整个过程如图 3 所示。

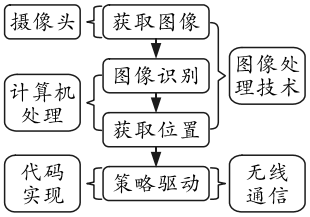

图 3　系统框架

计算机将摄像头获取到的图像传送到救援操作软件平台，救援人员在平台中利用鼠标或键盘捕获到目标点和多关节鱼，并采用加载策略的方式通过无线模块传递给多关节鱼指令，进而驱动多关节鱼进行救援。救援期间需不断捕获多关节鱼和支柱的位置，确保不会丢点。其中涉及图像传输、处理和无线通信等技术，这些技术均由操作软件实现。

3.2　确定最短救援路线

迪杰斯特拉算法是一种按照路径长度递增的次序产生最短路径的算法，用于有向网中求从某个源点到其余各顶点的最短路径。给定有向网 G 和源点 v，求从 v 到 G 中其余各顶点的最短路径[4]。这里采用迪杰斯特拉算法求取最短路径。

方法：首先，引进一个辅助向量 $\boldsymbol{D}$。它的每一个分量 $D[i]$表示当前所找到的从始点 v 到每个终点 v_i 的最短路径的长度。

$D[i]$的初态：若从 v 到 v_i 有弧，则 $D[i]$为弧上的权值；否，则 $D[i]$为∞[4]。

因此：

$$D[j]=\min\{D[i]|v_i\in v\}。\tag{1}$$

显然这条路径就是从 v 到 v_j 的长度最短的路径，此路径为(v, v_j)。由此 v 到 v_j 的最短路径找到。

修改从 v 出发到其他任一顶点 v_k 的最短路径长度。如果

$$D[j]+(v_j,v_k)\text{的长度}<D[k],\tag{2}$$

则修改 $D[k]$。它的长度是从 v 到 v_j，v_j 到 v_k 的弧上的权值之和[4]。

重复以上操作，可以求得从 v 到图 G 中其余各顶点的最短路径。

水中救援的路线可以用图 4 表示。为达到最佳救援目的，最短路径的设计与救援目标支柱的难易程度和到达目标支柱的距离有关，这里假设撞击目标点最低难度的权重为 1，中等难度的权重为 2，最高难度的权重为 3。图中边的权值=救援目标支柱的权重×到达目标支柱的距离(距离单位：m)。其中 0 号代表机器鱼，1–5 号顶点代表目标救援点。

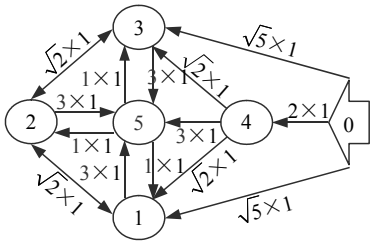

图 4　水中救援带权有向网

表 1 为该带权有向图的邻接矩阵，表 2 为求最短路径的过程表。

表 1　有向网的带权邻接矩阵

多关节鱼支柱顶点	0	1	2	3	4	5
0	∞	$\sqrt{5}$	∞	$\sqrt{5}$	2	∞
1	∞	∞	$\sqrt{2}$	∞		3
2	∞	$\sqrt{2}$	∞	$\sqrt{2}$	∞	3
3	∞	∞	$\sqrt{2}$	∞	∞	3
4	∞	$\sqrt{2}$	∞	$\sqrt{2}$	∞	3
5	∞	1	1	1	∞	∞

表 2　最短路径寻径表

终点	从 v_0 出发遍历所有点得到的最短路径				
	i=1	i=2	i=3	i=4	i=5
0	$\sqrt{5}$	$\sqrt{2}+2$	∞	$3\times\sqrt{2}+2$	–
1	∞	∞	$\sqrt{2}\times2+2$	–	–
2	$\sqrt{5}$	$\sqrt{2}+2$	–	–	–
3	2	–	–	–	–
4	∞	5	$\sqrt{2}+5$	$\sqrt{2}\times2+5$	$3\times\sqrt{2}+5$
j	4	1	2	1	5
S	{0, 4}	{0, 4, 3}	{0, 4, 3, 2}	{0, 4, 3, 2, 1}	{0, 4,3, 2, 1, 5}

综上所述，多关节鱼救援的最短路径的路线为{0, 4, 3, 2, 1, 5}。这个路线既考虑了距离，又考虑了每一个目标点的救援难易程度。两者结合起来作为路径的权值，这样求得的最短路径，既能有效地缩短多关节鱼到达各目标点的时间，又可以有效地保证如果未能在规定时间内完成任务，也能取得相对高效的救援效果。

3.3　救援方法

通过实验可知，鱼由静止到最大速度所花的时间非常短，这个时间耗费在整个救援过程中可忽略不计，由此判定鱼在很短的位移内能够达到最大速度。从鱼的起始位置到 4 号球的距离内，鱼必定能达到最大速度。

1) 鱼头直接撞击圆筒形支柱。

① 最大速度 v 的测量。用鱼的头部直接撞击，需考虑的因素有：鱼由静止状态变为最大速度所需的时间 t_0(可忽略不计)，撞击力 F，最大速度 v，鱼的质量 m_1。

② 已证明机器鱼由静止到达最大速度所需的距离很短，忽略那一段距离。从长方形水池的一侧到达另一侧的位移 S(255 cm)作为鱼的位移，用秒表计时得到时间 T，那么 $v=S/T$，近似得出鱼的最大速度 v，通过多次实验，得出 10 次数据，如表 3 所示。

表 3　鱼游(225 cm)次数与时间　　s

次数	1	2	3	4	5
时间	5.80	6.89	6.05	5.95	6.60
次数	6	7	8	9	10
时间	6.27	5.86	5.74	5.80	6.00

③ 求速度的平均值时，去除第 2 组最长和第 8 组最短数据，对余下 8 组数据取平均值，得：

t_0=6.04，v=0.422 m/s。

④ 在撞击的一瞬间，速度由 v 变为 0，得出撞击力 F 为：

$$F=1/2\times m_1\times v^2=0.089m_1\ \text{N}。$$

注：距离测量全部以鱼尾点进行测量。

2) 鱼头甩动撞击圆筒形支柱。

① 鱼游到圆柱形支柱附近，所用变量：鱼头甩动的力 F_2，鱼头部分的质量 m_2，鱼最大旋转程度所得的半径 r，旋转的角速度 w，转过的角度 θ，所用的时间 t。

注：角度测量值为鱼头相对鱼身偏转的角度。

② $w=\theta\pi/(180°t)$。

时间 t 的测量，由于转速太快，不能直接测出精确时间，故采用估计最短时间 t=1/20 s。

w=114°/180°π/0.05=14.915 1 (rad/s)。

③ $F_2=m_2rw^2$。

其中，r 的测量采用三点确定一个圆，获得圆心，测出半径的方法，单位为 m。

$F_2=m_2\times0.099\times(14.915)^2=22.023m_2$ N。

3) 2 种方法比较。

鱼头部分的质量为 m_2(kg)，鱼的质量为 m_1(kg)。

假设：$m_2=a\times m_1(0<a<1)$。

由此可得：$F_2/F_1=247.449a$。

若 $F_2/F_1>1$ 则只需 $m_2/m_1>0.004$。

实际测得，$m_2/m_1>0.04$。

所以 $F_2>F_1$。

结论：在其他因素确定的情况下，只比较撞击力时，采用 F_2 撞击的力更大一些。但采用鱼头甩动撞击时，鱼到达目标位置调整姿势需要耗费一定的时间，比采用鱼头直接撞击耗费时间要多。所以根据各目标点的难易度采用不同的撞击方式实现，会更省时间。

3.4　函数介绍

1) Strategy0 函数，可以同时捕获 5 个目标点的函数。

BOOL CStrategy::Strategy0(CFishAction m_action[],CFishInfo m_FishInfo[],CBallInfo m_goalinfo[], OBSTAINFO m_obst[],CHANNEL m_Channel[])。

m_action[]:鱼的动作列表，主要包含速度和角度 2 个属性，用于策略运行时调整角度与速度；参数类型：CFishAction；

m_fishInfo[]:鱼的信息列表，主要包含鱼体相对位置；参数类型：CFishInfo；

m_goalinfo[]:目标点信息列表，主要使用其中的目标点位置；参数类型：CBallInfo；

m_obst[]:障碍物信息列表，在水中救援项目中没有用到该参数；参数类型：OBSTAINFO；

m_Channel[]:鱼的接收频率列表，由于只使用到一只鱼，不使用该参数；参数类型：CHANNEL；

以往策略中通过获取目标点信息列表，一次标出一个目标点的位置，待目标点成功撞击后，再标注下一个目标点。此函数实现了一次性标注出 5 个目标点位置，不仅可以减少标注过程的时间，还在减小操作误差，提高策略效率等方面有了显著改善。

2) Point2point 函数，直接撞击函数。

Void CStrategy::Point2point(CPoint aimpt, CFishAction &action, int fishID, CFishInfo m_FishInfo)(机器鱼精确点到点函数)。

aimpt: 目标点，函数类型-CPoint，包含了 x 坐标与 y 坐标；

action 目标动作，函数类型-CFishAction，包含了要设置的角度信息与速度信息；

FishID 执行动作的鱼的编号，函数类型-int，要执行该动作的机器鱼 id(可选 0,1)；

m_Fishinfo: 机器鱼的信息，函数类型-CFIshinfo，包含了目标鱼当前的位置信息、角度和速度信息等。

该函数主要通过获得要执行动作的目标鱼的信息和目标点信息，计算出多关节鱼当前位置与目标点的距离和角度差，从而调整方向和速度。若角度差绝对值(角度差范围在-180°～180°之间)大于 70°时，速度值为最低，角速度值为最高，否则根据距离差将速度调至最大，在鱼的行进过程中不断进行角度微调，根据角度差值将角速度分为 11 个档位)。

如图 5 所示：以鱼头方向为 y 轴建立直角坐标系，圆代表目标点，X(m)代表目标点到鱼的距离，α(°)为鱼头方向和目标点的角度差，角速度值为 v。

当多关节鱼的 v<7 时，鱼头左偏；当 v>7 时，鱼头右偏。

当-5°<α<5°时，置 v=7，即沿当前方向前进；

当-10°<α<-5°时，置 v=5；

当-30°<α<-10°时，置 v=4；

当-50°<α<-30°时，置 v=2；

当-70°<α<-50°时，为置 v=1；

当 α<-70°时，置 v=0；

当 5°<α<10°时，置 v=9；

当 10°<α<30°时，置 v=10；

当 30°<α<50°时，置 v=12；

当 50°<α<70°时，置 v=13；

当 α>70°时，置 v=14。

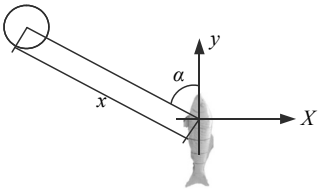

图 5 撞击目标点

Point2point 函数中通过计算角度的差值不断对目标角度进行判断，基本实现了机器鱼以较小误差和较快的速度移动至目标点的功能。

3) Hammer 函数，鱼头甩动函数。

void Hammer(CFishInfo fish,CPoint aim,CFishAction &action, int fishId, long AttackRange)，

其中 AttackRange 参数为自定义阈值[3]。

fish:鱼的信息，参数类型：CFishInfo；

aim:目标点，参数类型：CPoint；

&action: 鱼头与目标点的最小距离，参数类型：CFishAction；

fishId:鱼的 ID，参数类型：int；

AttackRange：自定义阈值，参数类型：long；

新版多关节鱼由 3 块舵机组成，其中鱼头部分的舵机(第一关节舵机)在由角速度值达到最大后再甩直所得到的力(鱼头甩动的力)大于直接撞击的力，因此可采用机器鱼“甩头”的方法撞击装有配重物的 5 号目标物和将罐子粘合在一起的 4 号目标物，完成救援。

具体过程分为 3 步：

① 以最快速度到达目标点附近；

② 将速度置为 1，不断靠近目标点，并在到达 attackrange 范围时将鱼的角速度值置为 0 或 15，速度值置为 0；

③ 将角速度值置为 7，实现“头部快速撞击”之后进行目标物是否存在的判断，选择前往下一目标点或者重复②的动作。

该 Hammer 函数的实现，在单关节鱼的鱼尾甩动函数基础上[5]，根据多关节鱼的实际情况，将原先单关节鱼的“甩尾”算法改为“甩头”算法。与原本单纯的点到点算法相比，大大缩短了撞击 5 号目标物所需的时间。

(下转第 96 页)

续将对这些参数值进一步优化。

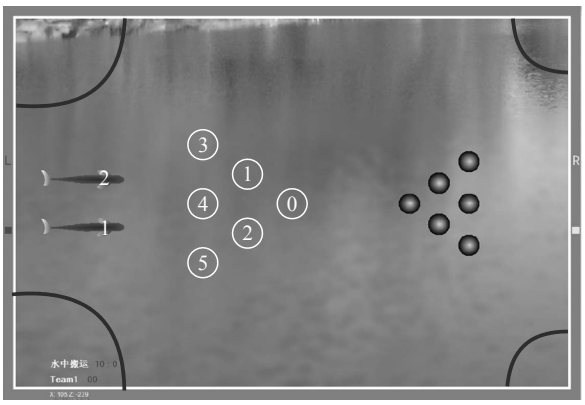

图 8　规定的禁区

2.2.2　补救策略

对于顶球时的失误，笔者准备了后续的策略来补救。即设当球在球洞附近，且球心坐标与球洞坐标所连成的直线，与坐标系的 X 轴成 alpha 角。对球洞到球心所连成的直线做反向延长，在反向延长线上某点处设定一个 A 点，此时令鱼头游到 A 点，鱼身与坐标系的 X 轴的角度成 alpha 角。此时令仿真鱼做直线高速运动，即可将球顺利补进球洞。补救方法如图 9 所示。

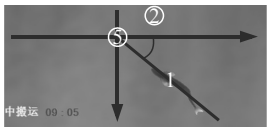

图 9　补救方法

3　结束语

笔者主要针对水中搬运比赛中出现的稳定性问题进行了研究，并给出了相应的解决方案和几种路径规划[6]。实验结果表明：在优化了稳定性问题并使用最终确定的路径规划后，水中搬运的 2 条仿真鱼可以非常稳定地完成得分的工作，运用该策略，北京信息科技大学北信守夜人队在 2016 年 10 月 21 日的国际水中机器人大赛取得了二等奖，证明了该策略的有效性。目前本文的策略中仍存在两鱼互相干扰的情况，为此，如何避免该情况成为笔者接下来的研究重点。

参考文献：

[1] 国际水中机器人联盟. 国际水中机器人联盟官方网站, http://www.ilur.org[Z].
[2] 包华, 李淑琴, 郭琴琴. URWPGSim2D 仿真平台之花样游泳比赛项目的设计与实现[J]. 北京信息科技大学学报, 2011, 26(5): 84-88.
[3] 胡庆松, 徐立鸿, ERIK G. 基于 NSGA-Ⅱ的 IPMC 机器鱼动态多目标相容优化控制[J]. 系统仿真学报, 2011, 25(9): 1925-1931.
[4] 王扬威, 王振龙, 李健. 仿生机器鱼研究进展及发展趋势[J]. 机械设计与研究, 2011, 27(2): 22-25.
[5] 朱毅, 张涛, 程农. 动态环境下基于子目标的移动机器人路径规划方法[J]. 系统仿真学报, 2010(S1): 254-257.
[6] 王梅娟, 王楠, 瞿逸洲, 等. 基于胸鳍辅助推进的仿生机器鱼[J]. 兵工自动化, 2016, 35(5): 80-83.

****

（上接第 88 页）

3.5　最优解

经过实验得知：多关节鱼以正常前进游动的方式能将 1、2、3 号目标点撞倒，4 号目标离多关节鱼起始位置最近但难度中等，并且被多关节鱼以正常直线游动的方式撞倒的概率较小，5 号目标难度最大，通过直接撞击很难完成；因此，在 1、2、3 号目标点采用直接撞击救援，而 4、5 号目标点采用鱼头甩动撞击救援。实验中，按照给出的最短路径选择撞击鱼的顺序，既实现了缩短鱼到达各目标点的时间，又保证了如未能在规定时间完成任务，也可得到高效的救援效果。2 种撞击方式的结合能快速有效地完成撞击，从而一定程度上缩短了救援时间，提高了救援效率。

4　结束语

在多关节鱼水中救援项目中，通过大量的实验研究表明，在函数中封装迪杰斯特拉思想实现多关节鱼的救援路径规划，采用鱼头甩动和直接撞击相结合的算法大大缩短了救援时间。但笔者研究的策略仍有不足，如度量的误差难以避免、鱼头甩动的速度过快，采用人眼可以识别的速度，误差也会较大。另外，在 Point2Point 函数中如果增加对距离差值的判断，将会避免机器鱼以较小误差“错过”目标点的情况，进而提高策略的准确性。

参考文献：

[1] 曹培, 姜飞, 许晓霞, 等. 水中机器鱼全局救援策略优化[J]. 福建电脑, 2015(11): 11-12.
[2] 周伟诚, 夏庆峰. 一种全局“水中救援”比赛策略[J]. 兵工自动化, 2015, 34(5): 93-96.
[3] 谢广明. 2016 国际水中机器人大赛全局视觉组规则[M]. 北京: 北京大学, 2016: 1-6, 13.
[4] 严蔚敏, 吴伟民. 数据结构(C 语言版)[M]. 北京: 清华大学出版社, 2007: 187-190.
[5] 孙琴, 武海健, 夏庆锋. 机器鱼游动性能改善方法[J]. 兵工自动化, 2014, 33(12): 67-71.