文章编号:1671-4598(2021)02-0187-07 DOI:10.16526/j.cnki.11—4762/tp.2021.02.037 中图分类号:TP391 文献标识码:A

改进鲸鱼算法及其在路径规划的应用

杨 博,李昌华,李智杰,张 颉
(西安建筑科技大学 信息与控制工程学院,西安 710055)

摘要:针对鲸鱼优化算法(WOA)存在的收敛速度慢、收敛精度低和易陷入局部最优等问题,提出了采用非线性收敛因子、协同 a 的惯性权重、时变独立搜索概率和免疫记忆改进的鲸鱼优化算法(IWTWOA);应用非线性收敛因子、协同 a 的惯性权重和时变独立搜索概率改进 WOA 迭代模型,平衡了算法的全局搜索和局部搜索能力,有效避免了陷入局部最优的问题;引入免疫算法的免疫记忆机制,提高了算法收敛速度;选取了 15 个基准测试函数进行性能测试,结果表明 IWTWOA 算法在稳定性、计算精度和收敛速度上均有所提高;最终将其应用在路径规划问题中,获得了较好的结果。

关键词:鲸鱼优化算法;免疫记忆;非线性收敛因子;惯性权重;路径规划

Improved Whale Optimization Algorithm and Application in Path Planning

Yang Bo, Li Changhua, Li Zhijie, Zhang Jie
(College of Information and Control Engineering, Xi'an University of Architecture and Technology, Xi'an 710055, China)

Abstract: In order to improve the slow convergence, low convergence accuracy and easy to fall into local optimality of whale optimization algorithm (WOA), the improved whale optimization algorithm (Immune memory—αweight—time varying search factor improved whale optimization algorithm, IWTWOA) is proposed. The application of nonlinear convergence factors, cooperative inertial weights and time—varying independent search probabilities improves the WOA iterative model, balancing the global exploration and local search capabilities of the algorithm, effectively avoiding the problem of local optimization. The memory mechanism of immune algorithm improve the algorithm convergence speed. Selected 15 test functions for performance testing, the results show that the IWTWOA has improved stability, calculation accuracy and convergence speed. Finally, applied it to the path planning problem, and obtained the better planning.

Keywords: whale optimization algorithm; immune memory; nonlinear factor; inertia weight; path planning

0 引言

近年来,元启发式算法在解决工程、经济、信息等多个领域的问题方面逐渐开始流行。群智能算法是元启发式算法的主要门类之一,通过模拟生物的社会行为来搜索问题最优解,包括粒子群优化算法(PSO, particle swarm optimization)、布谷鸟搜索算法(CS, cuckoo search)、蜻蜓算法(DA, dragonfly algorithm)、灰狼优化算法(GWO, grey wolf optimizer)、蛾—火焰优化算法(MFO, moth—flame optimization algorithm)和鲸鱼优化算法(WOA, whale optimization algorithm)。

鲸鱼优化算法(WOA, whale optimization algorithm)是 Mirjalili 和 Lewi[1],参照鲸鱼群体习性提出的一种新的群体智能优化搜索算法。通过测试函数集和实际应用验证,表明 WOA 同其他智能优化算法有足够的竞争力[1],但 WOA 在迭代过程中仍存在收敛速度慢、收敛精度低和易陷入局部最优的问题。因此,采用了非线性收敛因子、协同 a 的惯性权重和时变独立搜索概率以及免疫算法的免疫记忆机制对 WOA 进行改进,通过 15 个基准测试函数进行性能测试验证其有效性,最终将其应用在机器人路径规划问题中。

1 相关工作

近年来,大量学者对 WOA 进行了研究,T. Xu[2]等引入惯性权重来调节最佳解对迭代的影响。龙文[3]等通过对立学习策略优化初始种群,并设计了随迭代次数非线性变化的收敛因子,提出了改进的鲸鱼优化算法(IWOA)。G. Kaur 和 S. Arora[4]提出了混沌鲸鱼算法(CWOA),将混沌理论引入了 WOA 优化过程,用混沌图调整 WOA 的主要参数,提高了 WOA 的全局收敛速度并获得了更好的性能。Y. Q. Zhou[5]等引入 Lévy 飞行轨迹来增加生物多样性,提出了基于 Lévy 飞行轨迹的鲸鱼优化算法(LWOA)。吴泽忠和宋菲[6]提出了基于改进螺旋更新位置模型的鲸鱼优化算法(IMWOA),提升了 WOA 的收敛精度和收敛速度。

此外,WOA 已被广泛应用于实际问题中。肖子雅和刘升[7]应用反向学习和黄金分割数优化 WOA,提出 EGolden

收稿日期:2020-07-03; **修回日期**:2020-07-23。
基金项目:国家自然科学基金(61373112;51878536);陕西省自然科学基金(2020JQ—687);西安建筑科技大学基础研究基金(RC1716)。
作者简介:杨 博(1994-),男,河北石家庄人,硕士,主要从事智能建筑、虚拟现实与 BIM 方向的研究。
李昌华(1963-),男,宁夏银川人,博士,教授,博士生导师,主要从事图形图像处理、模式识别、数字建筑等方向的研究。
引用格式:杨 博,李昌华,李智杰,等.改进鲸鱼算法及其在路径规划的应用[J].计算机测量与控制,2021,29(2):187-193,201.

—SWOA 应用于压力容器和蝶形弹簧设计优化问题。Elham 和 Farnaz[8]将 WOA 算法与邻域搜索算法结合应用在城市固体废物回收车辆路径规划。Hany 和 Hasanien[9]采用基于 WOA 的 PI 控制器分别对直流斩波器和电网侧逆变器进行控制，以实现最大功率点跟踪运行，改善了光伏系统的动态电压响应。徐继亚等[10]采用基于冯诺依曼拓扑结构鲸鱼算法优化正交匹配追踪算法和优化小波核极限学习机的参数，有效提高滚动轴承故障的诊断精度。谢建群[11]等提出了一种混合小波包变换和正余混沌双弦鲸鱼优化（CSC-WOA）算法优化多层感知器神经网络（MLP）的短期云计算资源负载预测方法，提高了负载序列高频分量的预测精度和泛化能力。

2 鲸鱼优化算法（WOA）

鲸鱼优化算法（WOA）是 Mirjalili 和 Lewi[1]，参照鲸鱼群体的觅食习性（图 1），提出的群体智能优化搜索算法。

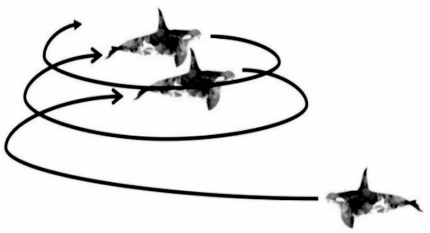

图 1 鲸鱼觅食行为

WOA 算法包含 3 种不同的搜索方式，分别模拟鲸鱼群的收缩包围机制、螺旋捕食机制和个体独立搜索机制。算法假设种群规模 N，第 i 只鲸鱼在 d 维空间中的位置为 $\boldsymbol{X}_i=(X_i^1,X_i^2,\ldots,X_i^d),i=1,2,\ldots,n$，猎物的位置对应问题的全局最优解。

$$\boldsymbol{X}_t=\begin{vmatrix} X_1^1,X_1^2,\cdots,X_1^d \\ X_2^1,X_2^2,\cdots,X_2^d \\ \cdots \\ \cdots \\ X_n^1,X_n^2,\cdots,X_n^d \end{vmatrix}\Rightarrow \boldsymbol{X}^*=\begin{vmatrix} X_1^{*1},X_1^{*2},\cdots,X_1^{*d} \\ X_2^{*1},X_2^{*2},\cdots,X_2^{*d} \\ \cdots \\ \cdots \\ X_n^{*1},X_n^{*2},\cdots,X_n^{*d} \end{vmatrix} \tag{1}$$

2.1 收缩包围

WOA 的第一种搜索方式模拟鲸鱼的收缩包围机制，即种群向最优位置的包围，见图 2，根据式（2）更新种群位置：

$$X(t+1)=X^*(t)+A\mid C*X^*(t)-X(t)\mid \tag{2}$$

其中：t 为当前迭代次数，$X^*(t)$ 为目标位置，迭代过程中 $X^*(t)$ 为当前最优个体，$A*\mid C*X^*(t)-X(t)\mid$ 为收敛幅度，A 和 C 定义如式（3）、（4）所示：

$$A=2a*r-a \tag{3}$$

$$C=2r \tag{4}$$

r 为 $[0,1]$ 上的随机向量，通过 r 种群个体更新在图 2 所示的当前最优点周边空间中的任何位置，来模拟对猎物的包围。a 为收敛因子，与 t 的关系见式（5）：

$$a=2-\frac{2t}{t_{\max}} \tag{5}$$

其中：$t_{\max}$ 为最大迭代次数。

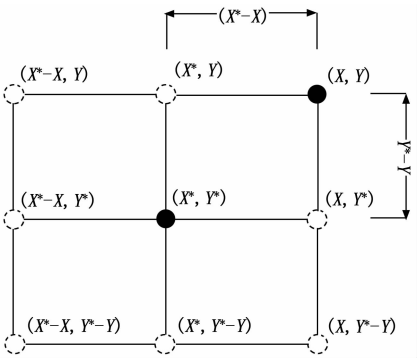

图 2 包围机制

2.2 螺旋捕食

WOA 的第二种搜索方式模拟鲸鱼螺旋觅食机制，螺旋行进更新位置，其数学模型如式（6）：

$$X(t+1)=X^*(t)+D*e^{br}*\cos(2\pi r) \tag{6}$$

其中：$D=\mid X^*(t)-X(t)\mid$ 表示第 i 只鲸鱼和目标之间的距离，b 为用于限定对数螺旋形状的常数，r 为 $[-1,1]$ 上的随机数。

鲸鱼捕猎时对猎物的包围和螺旋上升的过程是同时的，因此假设两种机制都有 50% 的可能性发生。因此给定 $\rho\in[0,1]$，鲸鱼的位置更新表达式为式（7）：

$$X(t+1)=\begin{cases} X^*(t)+A*\mid C*X^*(t)-X(t)\mid, & \rho\leqslant 0.5 \\ X^*(t)+D*e^{br}*\cos(2\pi r), & \rho>0.5 \end{cases} \tag{7}$$

2.3 独立搜索

除上述两种搜索方式外，WOA 中还存在着鲸鱼个体根据自身位置随机搜索猎物的行为，这种方式提高了算法的全局搜索能力，其数学模型表述为式（8）：

$$X(t+1)=X_{\text{random}}(t)-A*\mid C*X_{\text{random}}(t)-X(t)\mid \tag{8}$$

其中：$X_{\text{random}}(t)$ 为群体中随机选取的鲸鱼个体位置。

综上所述，WOA 算法流程如图 3 所示。

3 改进的鲸鱼优化算法

3.1 非线性收敛因子

WOA 主要通过 A 的值确定算法进行全局搜索或是局部搜索，A 的大小由收敛因子 a 决定，较大的 a 值算法全局搜索能力越强，越小的 a 值则局部搜索能力越强。在 WOA 算法中，收敛因子 a 的取值由 2 线性递减至 0，算法在迭代中后期易陷入局部最优的状况。为此，设计了一种非线性收敛因子，函数模型如图 4 所示，在初期保持在相对较高的数值一段时间，后迅速减少至低水平，并在迭代后期维持低数值，表达式如式（9）所示：

$$a=1.5+\frac{-1.2}{1+e^{-\zeta(2t-t_{\max})/2t_{\max}}} \tag{9}$$

其中：ζ 为调控因子，影响收敛因子 a 的变化速率，经实验 ζ 取值定为 20。

该非线性收敛因子在迭代初期保持在较高数值，提升

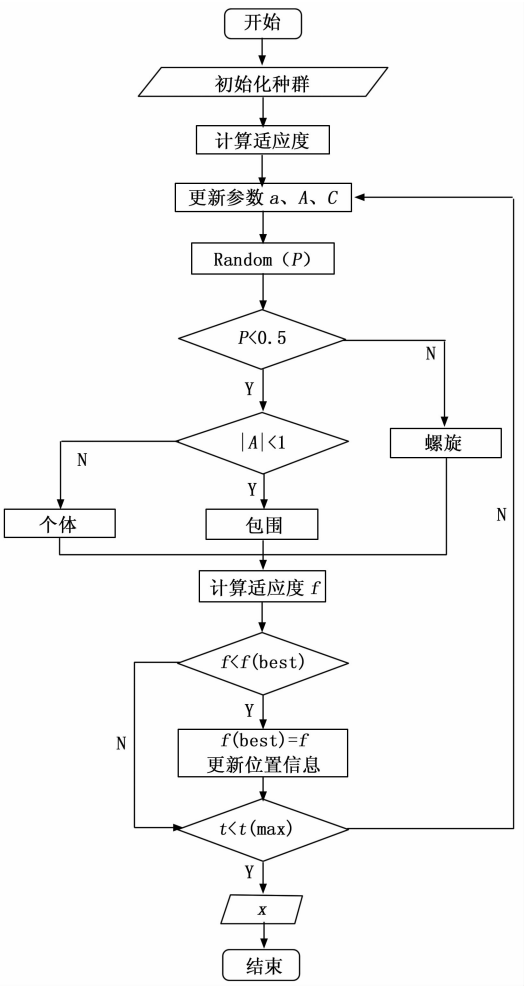

图 3　WOA 算法流程

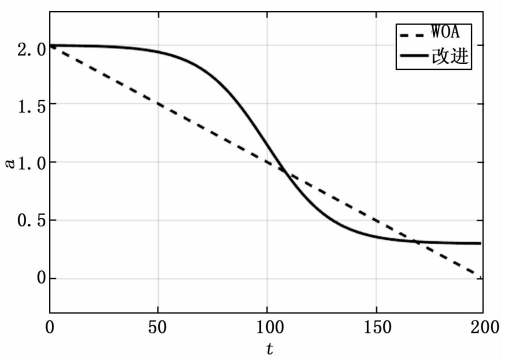

图 4　收敛因子

算法的全局搜索能力和收敛速度；在迭代中期迅速由较高数值衰减至较低数值，实现由全局搜索向局部搜索的快速过度；在迭代后期保持维持较低数值，保障算法的局部搜索性能。

3.2　协同 a 的惯性权重

惯性权重 w 是粒子群算法（PSO）的重要参数，对算法收敛性起很大作用，在 PSO 中 w 值越大，全局搜索能力越强；反之，w 值越小，则局部搜索能力越强，全局搜索能力越弱[12]。引入 PSO 算法的惯性权重 w 作为引导者权重，来提高收敛精度，同时更好地调节算法的全局搜索能力和局部搜索能力。协同 a 的惯性权重 w 值随迭代次数的增加而减小，并受收敛因子 a 控制，使 w 值与 a 值正相关，定义见公式（10）：

$$w=\frac{a}{(a_{\max}-a_{\min})}\left(w_{\max}-\frac{t(w_{\max}-w_{\min})}{t_{\max}}\right) \tag{10}$$

其中：$a_{\max}$ 为收敛因子 a 的最大值，$a_{\min}$ 为收敛因子 a 的最小值；$w_{\max}$ 为惯性权重的最大值，$w_{\min}$ 为惯性权重的最小值；$t_{\max}$ 为最大迭代次数。

引入惯性权重 w 对算法改进后，WOA 的 3 种搜索方式分别由公式（7）、（8）变更为公式（11）、（12）：

$$X(t+1)=\begin{cases}wX^{*}(t)+A*\mid C*X^{*}(t)-X(t)\mid, & \rho\leqslant 0.5\\ wX^{*}(t)+D*e^{br}*\cos(2\pi r), & \rho>0.5\end{cases} \tag{11}$$

$$X(t+1)=wX_{\text{random}}(t)-A*\mid C*X_{\text{random}}(t)-X(t)\mid \tag{12}$$

总体上，w 随迭代次数的增加而减小，同时受 a 的控制 w 也具备非线性变化，具有更强的适应性。经过 a 的系数放大，使算法在迭代初期具备更强的全局搜索能力，提高收敛速度；迭代中期随着 a 的急速下降，w 同时大幅减小，使算法在迭代中后期局部搜索能力大幅提升，提高收敛精度。

3.3　免疫记忆

Jangir[13]等的研究指出，在元启发式算法中，随机选择在勘探和开发过程中都具有非常重要的作用。但参数的随机性无法保证每一次迭代效果都是有效的，随机的搜索方式和移动步长很可能出现效果的衰退而停滞不前。因此，需要对随机的搜索方式进行一定的干预。

免疫记忆是免疫算法（IS）[14]的重要控制策略，可以加速优化搜索过程，提高优化效率和效果。IS 通过保存最优个体记忆信息，用于加速局部搜索或者抑制早熟收敛，从而使算法快速收敛到全局最优解。WOA 中每一代都在随机的迭代搜索，没有记忆单元。因此将 IS 的免疫记忆机制引入 WOA，为 WOA 添加记忆库 M，记忆各个体的适应度以及执行动作的各参数（w、A、C 和 r）。在下一次优化搜索中，比较当次迭代更新后的适应度和根据记忆库中的参数更新的适应度，若据记忆库中的参数更新效果好，则使用记忆库中的参数进行位置更新，否则更新 M。添加记忆单元后，改进 WOA 每次迭代都在与记忆单元操作比较的基础上运行，确保了算法快速收敛至全局最优解。

3.4　时变独立搜索概率

WOA 具有独立搜索的机制，由 $|A|\geqslant 1$ 控制，通过公式（3）以及公式（5）、（9）可知，无论是原始的 a 还是改进后的 a，在迭代中后期都无法触发独立搜索机制，导致鲸鱼算法更难跳出局部最优的情况。因此保持独立搜索机制在迭代后期仍有一定的发生概率，可以在一定程度上提高鲸鱼算法跳出局部最优的能力。定义一种时变独立搜索机制发生概率为 η，表达式见式（13）：

$$\eta=\begin{cases}|A| & t<\dfrac{t_{\max}}{2}\\ random[0,1.5] & t\geqslant\dfrac{t_{\max}}{2}\end{cases}\tag{13}$$

时变独立搜索概率 η 使得算法迭代中后期，仍具有一定的全局搜索能力，有效提高了算法后期跳出局部最优的能力，尤其是在密集多峰函数问题中具有较好的效果。

通过以上方式对鲸鱼算法进行改进后，基于免疫记忆、协同 a 的惯性权重和时变独立搜索概率的改进鲸鱼优化算法（IWTWOA，Immune memory－α Weight－Time varying search factor Improved Whale Optimization Algorithm）的具体流程，见算法 1。

算法 1：IWTWOA

$t=1$，$f_{best}=f_{op}$；

WHILE($t<t_{\max}$) DO

FOR $i=1$ TO n

FOR $j=1$ TO d

更新 a,A,C,r,ρ；

IF ($\rho<0.5$)

IF ($\eta<1$) THEN

包围机制更新位置；

ELSE IF($\eta\geqslant1$) THEN

随机个体机制更新位置；

END IF

ELSE IF($\rho\geqslant0.5$) THEN

螺旋机制更新位置；

END IF

END FOR

计算新的目标值 f_{new}；

与记忆库 M 中参数操作对比；

IF $f_{new}<f_M$ THEN

$f_{new}=f_M$；

END IF

表 1 测试函数

No.	函数	表达式	维数	范围	最小值
F1	Rastrigin	$F_1(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[−10,10]	0
F2	Ackley	$F_2(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i}x_j)^2$	30	[−100,100]	0
F3	Griewank	$F_3(x)=max_i\{\lvert x_i\rvert,1\leq i\leq n\}$	30	[−100,100]	0
F4	Sphere	$F_4(x)=\sum_{i=1}^{n}x_i^2$	30	[−100,100]	0
F5	Quadric Noise	$F_5(x)=\sum_{i=1}^{n}\lvert x_i\sin(x_i)+0.1x_i\rvert$	30	[−10,10]	0
F6	Generalized Schwefel	$F_6(x)=0.5+\dfrac{\sin^2(\sum_{i=1}^{n}x_i^2)-0.5}{(1+0.001\sum_{i=1}^{n}x_i^2)^2}$	30	[−100,100]	0
F7	Generalized Rastrigin	$F_7(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	30	[−5.12,5.12]	0
F8	Generalized Penalized	$F_8(x)=\dfrac{\pi}{n}\{10\sin(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+10\sin^2(\pi y_{i+1})]+(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)$ $y_i=1+\dfrac{x_i+1}{4}u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m,x_i>a\\0,-a<x_i<a\\k(-x_i-a)^m x_i<-a\end{cases}$	30	[−50,50]	0
F9	Ackley	$F_9(x)=-20\exp(-0.2\sqrt{\dfrac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp(\dfrac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i))+20+e$	30	[−32,32]	0
F10	Griewank	$F_{10}(x)=\dfrac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\dfrac{xi}{\sqrt{i}})+1$	30	[−60,60]	0
F11	Six－Hump Camel－Back	$F_{11}(x)=4x_1^2-2.1x_1^4+\dfrac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	4	[−5,5]	−1.03162
F12	Brainin	$F_{12}(x)=(x^2-\dfrac{5.1}{4\pi^2}x_1^2+\dfrac{5}{\pi}x_1-6)^2+10(1-\dfrac{1}{8\pi})\cos x_1+10$	2	[−5,5]	0.398
F13	Goldstein Price	$F_{13}(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]\times[30+(2x_1-3x_2)^2\times(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	[−2,2]	3
F14	Sum squares	$F_{14}(x)=\sum_{i=1}^{n}ix_i^2$	30	[−100,100]	0
F15	Zakharov	$F_{15}(x)=\sum_{i=1}^{n}x_i^2+(0.5\sum_{i=1}^{n}ix_i)^2+(0.5\sum_{i=1}^{n}ix_i)^4$	30	[−10,10]	0

表 2　实验结果

函数	算法	最小值	平均值	最大值	标准差
F1	WOA	1.58E−55	3.55856E−55	1.63653E−54	5.5502E−62
	IWOA	3.22E−170	1.65E−169	9.031E−169	1.63E−188
	IWTWOA	0	2.8413E−194	7.83E−193	3.1499E−206
F2	WOA	5.24E−24	1.76E−23	9.2947E−23	4.11077E−32
	IWOA	1.15E−69	6.08E−69	3.33E−68	4.44E−83
	IWTWOA	0	0	0	0
F3	WOA	26.354 988 51	35.809 061 65	79.809 903 81	0.069 251 404
	IWOA	0	2.1412E−185	5.069E−184	4.16E−201
	IWTWOA	0	4.5853E−195	1.35E−193	3.56E−206
F4	WOA	3.10E−71	1.67E−70	9.13E−70	3.65E−86
	IWOA	5.49E−86	1.83E−85	7.18E−85	1.21E−107
	IWTWOA	0	0	0	0
F5	WOA	1.313 522 018	6.185 921 655	33.578 354 34	1.53E−57
	IWOA	1.13E−86	3.58E−86	1.69E−85	3.63E−95
	IWTWOA	0	9.12E−187	2.4047E−185	1.21E−198
F6	WOA	0.002 498 383	0.001 902 146	0.006 224 216	0
	IWOA	0	0	0	0
	IWTWOA	0	0	0	0
F7	WOA	0	0	0	0
	IWOA	0	0	0	0
	IWTWOA	0	0	0	0
F8	WOA	0.112 218 454	0.127 327 999	0.583 769 443	0.037 039 101
	IWOA	0.018 085 144	0.010 345 462	0.054 163 285	0.004 928 509
	IWTWOA	0.009 873 548	0.003 741 839	0.018 431 802	0.002 532 6
F9	WOA	2.41E−15	3.61E−15	7.99E−15	8.88178E−16
	IWOA	0	8.88178E−16	8.88178E−16	8.88178E−16
	IWTWOA	0	8.88178E−16	8.88178E−16	8.88178E−16
F10	WOA	3.70E−18	2.03E−17	1.11022E−16	0
	IWOA	0	0	0	0
	IWTWOA	0	0	0	0
F11	WOA	−1.031 628 453	−1.022 884 448	−0.995 570 948	0.006 583 167
	IWOA	−1.031 628 453	−1.030 426 525	−0.995 806 891	0.014 161 225
	IWTWOA	−1.031 628 453	−1.031 628 178	−1.031 627 1	3.56E−07
F12	WOA	0.397 887 358	0.401 176 797	0.424 167 904	0.007 635 092
	IWOA	0.397 889 45	0.555 415 666	5.023 177 988	0.843 852 408
	IWTWOA	0.397 887 358	0.397 915 238	0.398 096 928	5.91434E−05
F13	WOA	3.000 000 013	11.150 662 86	84.000 002 51	21.475 428 63
	IWOA	3.000 002 527	3.973 292 458	30.024 091 87	4.921 420 253
	IWTWOA	3.000 000 378	3.002 698 888	3.030 888 299	0.005 888 78
F14	WOA	1.60731E−84	8.80201E−84	4.82E−83	1.44E−109
	IWOA	2.65E−99	1.44481E−98	7.91E−98	1.17E−134
	IWTWOA	0	0	0	0
F15	WOA	505.024 430 3	105.090 857	812.748 79	318.712 225 6
	IWOA	4.50592E−98	1.80E−97	8.81E−97	2.70E−130
	IWTWOA	0	0	0	0

IF $f_{new} < f_{best}$ THEN
$f_{best} = f_{new}$；
更新引导者位置 X_{best}；
END IF
END FOR
更新记忆库 M；
$t = t + 1$；
END WHILE

4 测试与对比

4.1 测试函数

为了验证 IWTWOA 算法的性能，选取了 15 个基准函数进行对比实验，基准函数详见表 1。这些函数分为具有少量局部最小值的单峰函数和具有大量局部最小值的多峰函数以及固定维函数。函数 F1－F4、F14、F15 是单峰函数，函数 F5－F10 是多峰函数，函数 F11－F13 是固定维函数。

4.2 实验结果与分析

将 IWTWOA 算法与 WOA 算法和文献 [3] 中所提到的 IWOA 算法进行对比试验，各进行 30 次实验，分别记录每种算法实验结果的最小值、平均值、最大值和标准差，实验结果见表 2。

由表 2 的实验数据可知，IWTWOA 算法在对测试函数 F2、F4、F6、F7、F10、F14、F15 的测试中都收敛到了最小值 0，标准误差为 0，其中 F2、F4、F14、F15 为单峰函数，F6、F7、F10 为多峰函数，证明了算法的适应性和稳定性。对测试函数 F6、F7、F10 的测试中，WOA、IWOA、IWTWOA 算法均得到了最优解，但在收敛速度上 IWTWOA 明显领先，收敛代数详见表 3。

F1－F4 和 F14、F15 的结果表明，在单峰函数问题中，IWOA、IWTWOA 算法相较于 WOA 算法优化效果和收敛精度均有明显提升，相对 IWOA 而言 IWTWOA 算法改进效果更为显著。F5－F10 的多峰函数测试说明，鲸鱼算法在多峰函数问题的求解上优势巨大，除在 F8 函数的测试中 IWTWOA 算法的计算结果误差相对较大以外，对其余函数的计算结果或达到最优，误差为 0 或误差十分接近 0，对 F9 函数的测试中 3 种算法效果相同。F11－F13 的固定维函数测试中，IWOA、IWTWOA 算法较 WOA 算法在计算精度上均有所提升，F13 函数测试中 WOA 算法所得的最优值更接近目标值，IWTWOA 算法具有更好平均效果和稳定性，F11 函数测试中 3 种算法都得到过最优解，但 IWTWOA 算法稳定性更高。

总体而言，IWTWOA 算法对 WOA 算法在计算精度和收敛速度上均有所提高，尤其是在 F1、F5 的计算中，IWTWOA 算法相较于改进的 IWOA 算法也有较大幅的性能提升。为更直观展示优化效果，选择测试函数中较典型的几个进行绘图展示，如图 5 所示。

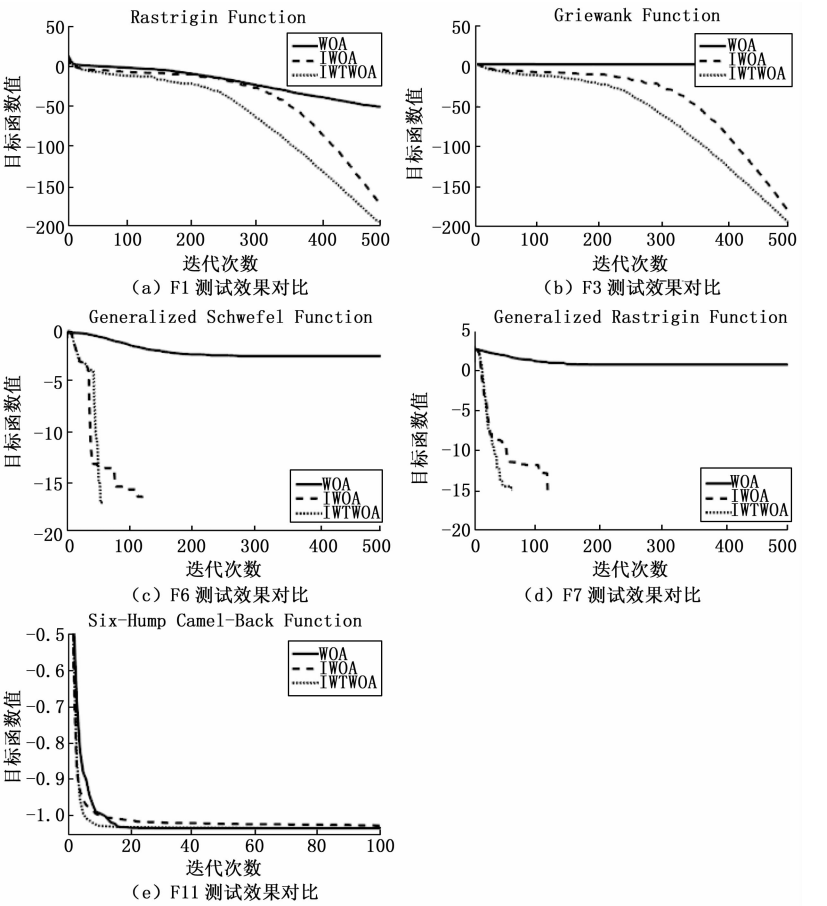

图 5 部分测试效果对比

表 3 平均迭代次数

函数	算法		
	WOA	IWOA	IWTWOA
F6	412.9	100.6	59.1
F7	327.3	78.133	55.633
F10	222.133	71.833	56.3

5 IWTWOA 在路径规划的应用

路径规划是机器人领域的热门研究问题，在机器人、飞行器、导航、物流及通信等诸多高新科技领域都有广泛应用。路径规划就是在具有障碍物的环境内按照一定的评价标准，寻找一条从起始位置到达目标位置的无碰路径[15]。

近年来随着群智能算法的发展，基于群智能优化的路径规划研究层出不穷。秦元庆[16]等采用链接图进行环境建模，使用粒子群算法对 Dijkstra 算法求得的最短路径进行优化，得到全局最优路径。李珣[17]等结合三次样条插值方法、选取罚函数作为适应度函数等对 PSO 进行了算法改进，提高了室内环境中路径规划的实时性和有效性。朱庆保[18]采用了概率搜索策略、最近邻策略和目标引导函数改进蚁群算法，搜索过程极为迅速高效。吴坤[19]等将 WOA 算法引入无人机航路规划研究，获得了代价最优的、有效的航路结果。

5.1 环境建模

栅格法是路径规划环境建模公认最成熟的技术[20]。栅格法将规划空间划分为数个格网状区块，将空间环境转化为一个由“0”，“1”信息表示的矩阵 *grid*，*grid* 中“0”表

示可自由通行空间单元，“1”表示被障碍物占据的单元。栅格精度取 $g=10$ cm，栅格法环境建模效果如图 6 所示。

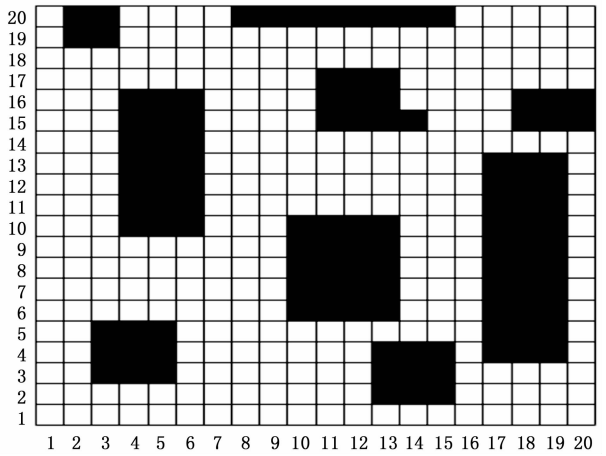

图 6　栅格法环境建模

5.2　代价函数

根据路径规划问题的描述，应满足：1）路径最短、2）和障碍物无碰撞，可以通过定义路径的代价函数获取最优路径。代价函数定义为：

$$f=\sum_{i=1}^{n}L(i)+C(i) \tag{14}$$

其中：i 为迭代次数；$L(i)$ 为每次迭代移动的直线距离，定义为：

$$L(i)=g\sqrt{(x_i-x_{i-1})^2+(y_i-y_{i-1})^2} \tag{15}$$

x_i，y_i 表示当前所在位置，x_{i-1}，y_{i-1} 表示迭代前所在位置，g 是栅格精度。

$C(i)$ 为障碍物碰撞代价，定义为：

$$C(i)=k\sum grid(p_i) \tag{16}$$

其中：p_i 为当次迭代途径的点，$grid(p_i)$ 为栅格环境下障碍物信息，k 为一个较大的常数以保障路径与障碍物无碰撞。

那么，路径规划问题就转换成求解代价函数最小值问题，即：

$$min\quad f \tag{17}$$

5.3　IWTWOA 路径规划流程

综上所述，IWTWOA 对路径规划问题的具体求解过程如下：

Step1：环境建模，确定起点与终点；

Step2：初始化 IWTWOA 参数；

Step3：根据式（14）计算代价函数，将路径规划转化为优化问题；

Step4：根据 IWTWOA 的式（11）、（12）进行位置更新，对路径规划问题进行优化求解；

Step5：与记忆库 M 中的参数操作对比，判断是否跟新记忆库，迭代次数加 1；

Step6：判断是否满足终止条件，满足则输出最优路径，否则跳转 Step4。

5.4　实验测试

基于上节算法流程描述，将 IWTWOA 算法与 WOA 算法和 PSO 算法进行仿真实验。在上述 20×20 的栅格环境下进行仿真，起点 $S=(1,1)$，终点 $D=(20,20)$，黑色区域为障碍物。3 种算法的初始种群数均为 $N=30$，最大迭代次数均为 500 次，搜索范围 $[-20,20]$。PSO 算法惯性权重 $\omega=0.78$，$c_1=1.5$，$c_2=1.5$。

路径规划结果如图 7 所示，表 4 给出了 3 种算法的代价函数最大值、最小值和平均值。由表 4 可知，改进的 IWTWOA 算法较 WOA 算法在应对路径规划问题的整体性能上有较大的的提升，虽然 PSO 算法取得了最好的优化值结果，但 IWTWOA 算法在最大值和平均值方面都取得更好的结果，说明 IWTWOA 算法能够找到相对较短的路径，且具有较好的稳定性。

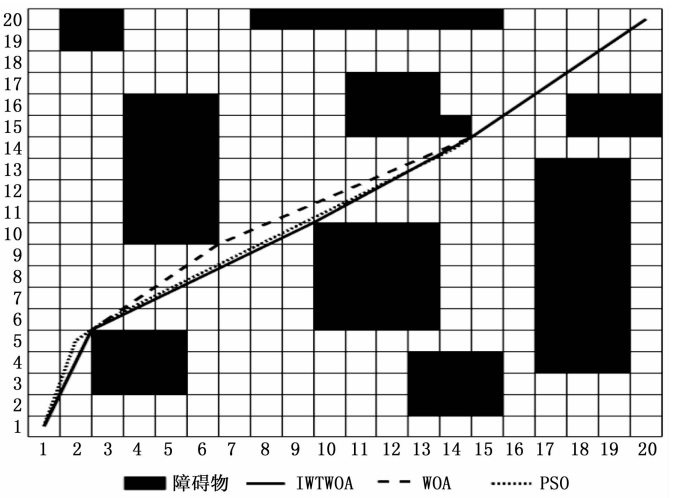

图 7　路径规划结果

表 4　代价函数比较

代价函数	算法		
	IWTWOA	WOA	PSO
最大值	304.8	312.5	306.4
最小值	293.1	293.8	292.8
平均值	301.2	307.8	302.6

经仿真实验测试，提出的 IWTWOA 算法能够规划出一条有效的、较短的路径，在稳定性和收敛精度上均有所提高。

6　结束语

本文采用非线性收敛因子、协同 a 的惯性权重和时变独立搜索概率改进鲸鱼算法迭代模型，并引入免疫算法的免疫记忆机制对 WOA 进行改进；经 15 个基准测试函数的

（下转第 201 页）