基于变异策略的自适应七星瓢虫优化算法*

魏锋涛，卢凤仪，郑建明
（西安理工大学 机械与精密仪器工程学院，西安 710048）

摘　要：针对七星瓢虫优化算法易陷入局部最优、求解精度不高的缺陷，提出基于变异策略的自适应七星瓢虫优化算法。为提高算法求解质量，在每次迭代搜索时利用柯西变异策略增加解的多样性，引入竞争淘汰机制淘汰适应度值较差的个体；同时，为了提高算法的收敛性能，在算法搜索后期利用混沌变异策略对种群中最优和较优个体进行混沌变异操作，并对学习因子进行自适应更新调整。利用标准测试函数进行实验仿真，结果表明改进算法不仅提高了求解精度，同时有效避免了局部收敛问题。

关键词：七星瓢虫优化算法；柯西变异策略；混沌变异策略；自适应学习因子；函数优化
中图分类号：TP301.6　　**文献标志码**：A　　**文章编号**：1001-3695(2018)08-2320-03
doi:10.3969/j.issn.1001-3695.2018.08.020

Adaptive seven-spot ladybird optimization based on mutation strategy

Wei Fengtao, Lu Fengyi, Zheng Jianming
(*School of Mechanical & Precision Instrument Engineering, Xi'an University of Technology, Xi'an* 710048, *China*)

Abstract: Considering the problems of falling into the local optimum and low solving precisions of seven-spot ladybird optimization(SLO), this paper proposed an adaptive seven-spot ladybird optimization algorithm based on mutation strategy. In order to improve the quality of solutions, it applied Cauthy mutation to increase the diversity of population and competition mechanism to eliminate bad individual according to its fitness in every iteration. To improve convergence performance of the SLO, this paper varied the best and the second-best based on chaos mutation strategy in the late iterations. Meanwhile, it adjusted the acceleration coefficient adaptively. Simulation experimental results based on a set of widely used benchmark functions show that the improved algorithm can yield solutions with higher precision, while effectively avoiding local convergence problems.

Key words: seven-spot ladybird optimization; Cauthy mutation strategy; chaos mutation strategy; adaptive learning factor; function optimization

近年来，通过模拟研究某一自然现象或生物某一行为发展而来的仿生学智能算法受到了人们的广泛关注。在现有仿生学算法中，文献[1]通过模拟自然界鸟群聚集觅食行为提出了粒子群优化算法(partical swarm optimization, PSO)；文献[2]通过模拟蜂群繁殖和采蜜行为提出了人工蜂群算法(artificial bee colony, ABC)；还有学者基于某些动物习性大胆提出鸡群算法[3]、仿生蚊子追踪算法[4]等。该类算法与传统优化算法相比，在求解时不依赖于目标函数性质与搜索空间的限制，且易于编程实现，适用于求解传统方法解决不了的非线性问题，已被广泛应用于工程领域[5~7]。

七星瓢虫优化算法(seven-spot ladybird optimization, SLO)是根据瓢虫捕食习性提出的一种新型智能仿生算法，具有参数简单、鲁棒性高等优点。该算法与其他智能算法类似，在处理多维、高峰的复杂问题寻优时，通常存在着求解质量不高、早熟等问题[8,9]。鉴于此，本文研究一种改进七星瓢虫优化算法(improved seven-spot ladybird optimization, ISLO)，在算法每次迭代时随机抽取部分瓢虫对其进行柯西变异，并在循环末尾淘汰适应度较差的瓢虫，以增强种群的多样性，提高求解精度；在算法搜索后期对适应度值最优及较优个体进行混沌变异操作，防止算法出现早熟收敛现象；同时为提高算法的收敛性，引入社会认知与自我认知学习因子项，并自适应调节学习因子大小。利用测试函数对改进七星瓢虫算法的有效性进行验证。

1　七星瓢虫优化算法

七星瓢虫优化算法是通过模拟七星瓢虫捕食行为而构造出的一种仿生智能优化算法[10]。其核心思想是模拟七星瓢虫在搜索过程中采用的分区精细搜索机制与局域广域结合的搜索方式，在初始化种群前先分区产生若干子空间，个体通过自身当前位置 present 与子空间历史最优 l_{best}、个体历史最优 p_{best}、全局最优 g_{best} 的先后比较而进行下一步的更新，最终实现寻优目的。

对于 D 维空间中某一瓢虫 i，假设每维搜索空间被分为 n 个子空间，该瓢虫位置为 $X_i=(X_{i1},X_{i2},X_{i3},\cdots,X_{iD})$，速度为 $V_i=(V_{i1},V_{i2},V_{i3},\cdots,V_{iD})$。瓢虫 i 的位置更新计算公式为

$$X_i(t+1)=X_i(t)+V_i(t) \tag{1}$$

其中：t 表示当前迭代次数；$V_i(t)$ 表示当前速度；$X_i(t)$ 表示当前位置。

由于瓢虫的搜索方式主要分为局域搜索和广域搜索，如果瓢虫之前进行了广域搜索，那么这次循环时它将会先进行局域搜索；在局域搜索完成后，瓢虫将转换成广域搜索。瓢虫在局域搜索中的速度更新公式为

$$V_i(t)=c\times r_1\times(\mathrm{pbest}_i(t)-X_i(t))+\varepsilon_1 \tag{2}$$

瓢虫在广域搜索中的速度更新公式为

$$V_i(t)=c\times r_2\times(\mathrm{lbest}_i(t)-X_i(t))+\varepsilon_2 \tag{3}$$

其中：pbest_i、lbest_i 分别为个体历史最优位置、子空间历史最优位置；r_1、r_2 为在[0,1]服从均匀分布的随机数，以增加搜索的随机性；c 为学习因子，用于调节步长与方向；ε_1、ε_2 为两个相对小的随机数。

若一定循环后瓢虫的位置仍不改善，则根据式(4)在全局最优解附近产生一个新解替代该位置。

收稿日期：2017-03-31；**修回日期**：2017-05-08　**基金项目**：国家自然科学基金资助项目(51575443,51475365)；西安理工大学博士启动基金资助项目(102-451115002)；陕西省自然科学基础研究计划资助项目(2017JM5088)

作者简介：魏锋涛(1976-)，男，陕西合阳人，副教授，博士，主要研究方向为现代优化设计理论与方法、复杂产品多目标多学科设计优化(weifengtao@xaut.edu.cn)；卢凤仪(1994-)，女，河南信阳人，硕士研究生，主要研究方向为现代优化设计方法；郑建明(1968-)，男，四川仁寿人，教授，博士，主要研究方向为优化设计理论及工程应用.

$$x'_{i,j} = x_{\text{gbest},j} + \varphi\omega \tag{4}$$

其中:ω 是全局最优解 gbest 的邻域;φ 为修正量,是[-1,1]的随机数。

七星瓢虫优化算法基本流程如图 1 所示。

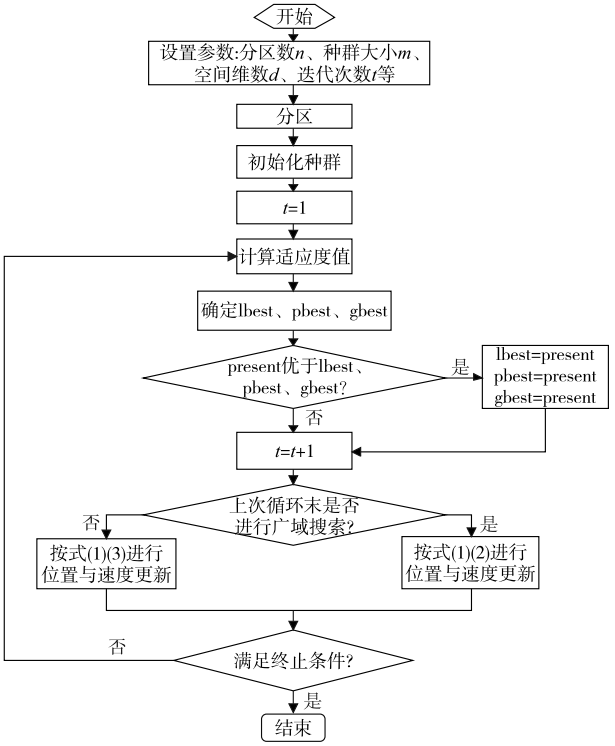

图1　七星瓢虫优化算法基本流程

2　基于变异策略的自适应七星瓢虫优化算法

在 SLO 算法中瓢虫通常追随当前最优值进行全局寻优,但由于其缺乏良好的变异机制,随着迭代次数增加,种群个体间的差异会逐渐减小,所以种群出现聚集现象致使算法易陷入局部最优且求解精度不高。此外,在该算法中用于控制自身与种群信息的学习因子相同且取值固定,这种做法忽略了学习因子的重要性,也势必会影响算法的收敛能力。鉴于此,本文引入两种变异策略,即柯西变异策略与混沌变异策略,并结合动态自适应调整学习因子,提出一种基于变异策略的自适应七星瓢虫优化算法。

2.1　柯西变异策略

在算法每次迭代伊始采用柯西变异策略,以保证种群的多样性;同时为提高求解精度,在算法每次迭代末尾引入竞争淘汰机制。柯西分布 Cauthy(θ,α)是概率与统计中的函数分布,其函数表达式如式(5)所示。当 $\theta=0$,$\alpha=1$ 时,就构成了标准柯西分布(Cauchy(0,1))。

$$f(x) = \frac{1}{\pi}\arctan(x) + \frac{1}{2} \quad x \in [-\infty, \infty] \tag{5}$$

在算法每进行一次寻优迭代开始,随机抽取各子空间中的一部分对其进行柯西变异操作,产生的变异个体位置如式(6)所示。

$$x'_i = x_i + \text{Cauchy}(0,1) \times \text{rand} \tag{6}$$

其中:Cauchy(0,1)为标准柯西分布;rand 是 0 ~ 1 的随机数;x_i 是当前被选择的瓢虫个体。

由于柯西变异操作产生部分新个体,种群规模势必超过原有种群数,所以在算法每完成一次搜索时,根据竞争淘汰机制,按照适应度值淘汰掉各子空间排名靠后的个体,直到种群达到原有规模为止。这样,算法在每次寻优迭代后都能不断进行更新,从而可以更好地提升种群个体的品质,保证求解质量。

2.2　混沌变异策略

为避免算法在后期陷入局部最优,引入混沌变异策略。根据混沌序列的随机性、遍历性,对陷入局部最优的解进行混沌变异。由于在算法后期,每个子空间的较优解比最优解可能存在更大的改进空间,所以两者一起进行变异操作,使算法更有效地跳出局部最优。本文采用混沌映射 logistic 迭代方程为

$$ch_i = 4 \times ch_i \times (1 - ch_i) \quad i = 2, \cdots, k \tag{7}$$

其中:$ch_i \in (0,1)$;k 表示混沌序列的长度。

具体操作过程为:首先,对每个子空间中的瓢虫进行适应度值排序,从排名前 5% 的个体中随机抽取一个瓢虫 l'_{best},与子空间最优解 l_{best}分别按式(7)生成混沌变量 ch_i,按照式(8)将混沌变量映射为混沌变量 CH_i,按照式(9)将 CH_i 与 l_{best}线性叠加,生成混沌变异向量 X_{best}'与 X_{best}。从 X_{best}'与 X_{best}中选优与 l_{best}比较,若变异解优于子空间最优解,则替换;否则将继续进行变异操作,直到变异解优于子空间最优解或混沌序列长度达到 k,则变异操作结束。

$$CH_i = LB + ch_i \times (UB - LB) \quad i = 2, \cdots, k \tag{8}$$

其中:UB、LB 表示混沌向量定义域的上、下限。

$$x = (1-\delta) l_{\text{best}} + \delta \times CH_i \quad i = 1, \cdots, k \tag{9}$$

其中:δ 是调整系数,根据算法迭代信息确定,具体表达式如式(10)所示。

$$\delta = \frac{t\text{Max} - t + 1}{t\text{Max}} \tag{10}$$

其中:t 是当前迭代次数;tMax 是最大迭代次数。

2.3　自适应调整学习因子

为平衡算法在全局探索和局部开采的能力,提高算法的收敛性能,引入自适应调整学习因子策略。根据自身信息、种群信息分别设置学习因子 c_1、c_2,并在此基础上,根据种群迭代信息动态调整学习因子。具体更新公式如式(11) ~ (14)所示。

$$V_i(t) = c_1 \times r_1 \times (\text{pbest}_i(t) - X_i(t) + \varepsilon_1) \tag{11}$$

$$V_i(t) = c_2 \times r_2 \times (\text{lbest}_i(t) - X_i(t) + \varepsilon_2) \tag{12}$$

$$c_1 = 2 \times (1 - (t + 2\cos(t))/(t\text{Max} + 1)) \tag{13}$$

$$c_2 = 2 - c_1 \tag{14}$$

其中:t 是当前迭代次数;tMax 是最大迭代次数。

由式(13)(14)可知,在算法迭代前期,c_1 取较大值,c_2 取较小值,希望瓢虫多向自身学习,增强种群的多样性;而在搜索后期,两者取值刚好相反,希望瓢虫向社会最优学习,以增强局部开采能力,提高算法的收敛速度。

2.4　改进算法的求解步骤

改进的七星瓢虫优化算法具体求解步骤如下:

a)种群初始化及参数设置,包括分区数 n、七星瓢虫种群规模 N、迭代次数 t、空间维度 D。

b)分区操作,按照种群规模与分区数进行分区,每个子空间大小为 $m = \frac{N}{n}$。

c)随机抽取每个子空间 5% 的个体按照式(5)(6)进行柯西变异操作、

d)计算适应度值,比较当前最优值与子空间最优值、个体历史最优值。若当前值优于个体历史最优值则进行置换,否则保持不变。

e)按照式(11) ~ (14)对七星瓢虫优化算法进行自适应更新操作。

f)当完成一次寻优后,对所有子空间个体进行评价;淘汰适应度值较差个体,保留前 m 个个体。

g)若子空间最优解连续两次没有改进,则对最优解及适应度值排名前 5% 的较优解按照式(7) ~ (10)进行混沌变异操作产生新解。若优于最优解则替换;否则将继续循环变异操作,直到满足条件或混沌序列长度达到最大。

h)若达到最大迭代次数则停止迭代,输出全局最优解;否则转向 c)。

3　仿真实验与分析

为了更好地分析 ISLO 算法的收敛性能与寻优能力,本文

选取常用的四个典型测试函数 $f_1 \sim f_6$，如表 1 所示。其中 f_1、f_3、f_5 函数是多峰高维函数，具有众多局部最优值；f_2、f_4、f_6 函数是高维单峰函数，极难搜索到全局最优点。利用测试函数对 ISLO 进行仿真测试，并与标准七星瓢虫优化算法（SLO）和粒子群优化算法（PSO）求解结果进行对比。其基本参数设定如表 2 所示。此外 ISLO 与 SLO 算法在空间维度为 10、30 和 50 维下的分区数对应为 2、3 和 5。实验仿真平台为 Windows XP，仿真所用软件为 MATLAB 2014（a）。

表 1 典型测试函数

名称	表达式	类型	搜索范围	理论最小值
Griewank	$f_1(x)=\frac{1}{400}\sum_{i=1}^{n}[x_i^2-\prod_{i=1}^{n}\cos(\frac{x^2}{\sqrt{i}})+1]$	多峰	$[-600,600]$	0
Rosenbrock	$f_2(x)=\sum_{i=1}^{n}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	单峰	$[-30,30]$	0
Rastrigin	$f_3(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)^2+10]$	多峰	$[-5.12,5.12]$	0
Schwefel's	$f_4(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	单峰	$[-500,500]$	0
Alpine	$f_5(x)=\sum_{i=1}^{n}\lvert x_i\sin x_i+0.1x_i\rvert$	多峰	$[-10,10]$	0
Sumsquares	$f_6(x)=\sum_{i=1}^{n}ix_i^2$	单峰	$[-10,10]$	0

表 2 算法测试参数取值

算法	种群规模（N）	空间维度（D）	最大迭代次数（M）
PSO	50	10、30、50	$D=10, M=900$ $D=30, M=1200$ $D=50, M=1500$
SLO	20		
ISLO	20		

为充分体现 ISLO 算法的性能，三种算法针对六个测试函数在空间维度 10 维、30 维和 50 维下各运行 30 次，利用算法求得的平均值、最优值和方差来考察算法的寻优特性。具体仿真结果如表 3 ~ 5 所示。为能直观体现 ISLO 算法的收敛性能，图 2 给出了六个函数在 50 维时采用三种算法求解得到的适应度进化曲线。其中，迭代次数为横坐标，适应度值的对数（以 10 为底）为纵坐标。

表 3 10-D 仿真测试结果

测试函数	维数	统计	PSO	SLO	ISLO
f_1	10	best	5.1062E-2	4.5009E-3	9.5456E-12
		mean	5.5244E-3	6.5548E-3	2.8459E-3
		std	4.1985E-4	6.6035E-6	3.1205E-5
f_2	10	best	3.8085E+0	1.5666E-1	1.9370E-6
		mean	3.9868E+0	3.1036E-1	5.6732E-2
		std	5.4924E-1	4.9953E-2	4.8516E-2
f_3	10	best	3.2354E+1	5.0953E+0	1.0890E-1
		mean	3.3248E+1	5.8153E+0	1.4212E-0
		std	1.3655E+1	1.5743E+1	8.1218E+0
f_4	10	best	6.4065E+0	4.1637E+0	5.4246E-7
		mean	6.5049E+1	4.1685E+3	5.1406E+0
		std	2.0839E+5	1.8636E+1	1.2833E+1
f_5	10	best	3.9755E-1	1.6728E+0	2.5620E-3
		mean	7.6223E-1	4.2946E+0	2.5796E-2
		std	2.5899E+3	4.0257E+1	1.7798E-2
f_6	10	best	3.4736E-1	1.7822E+0	5.4691E-2
		mean	3.2334E+0	5.3728E+0	8.7219E-1
		std	9.2734E+3	4.5734E-1	3.5731E-1

表 4 30-D 仿真测试结果

测试函数	维数	统计	PSO	SLO	ISLO
f_1	30	best	1.3591E-1	4.1181E-2	3.8255E-9
		mean	1.7365E-1	6.8784E-2	9.1530E-3
		std	1.0918E-3	2.6170E-4	5.2869E-4
f_2	30	best	1.8200E+2	3.4968E+1	2.8051E-1
		mean	1.8441E+2	5.7325E+1	3.9783E+0
		std	8.8502E+1	4.7435E+2	4.5770E+1
f_3	30	best	2.7493E+1	2.9648E+0	2.8051E-1
		mean	2.7957E+1	4.2769E+0	1.0983E+0
		std	7.4174E+0	9.8847E-1	2.1991E-1
f_4	30	best	1.9683E+1	6.2518E+0	1.5265E-7
		mean	1.5288E+2	1.2526E+4	1.6757E+1
		std	1.5061E+6	7.4626E+1	1.7522E+2
f_5	30	best	5.4512E-1	2.2697E+0	2.7466E-2
		mean	7.8924E+0	3.5609E+1	3.7596E-1
		std	3.1673E+4	3.6206E+1	1.7420E-1
f_6	30	best	1.3782E+0	1.4763E+1	2.5442E-2
		mean	9.2347E+1	4.3852E+2	5.3846E-1
		std	1.5428E+3	2.4838E-1	4.4823E-1

表 5 50-D 仿真测试结果

测试函数	维数	统计	PSO	SLO	ISLO
f_1	50	best	2.0835E-1	9.1047E-2	1.5486E-8
		mean	2.1347E-1	1.2910E-1	1.7066E-2
		std	9.5296E-4	1.4339E-4	8.6348E-4
f_2	50	best	5.6357E+2	7.8377E+0	1.1457E+0
		mean	5.6915E+1	1.0158E+1	1.5096E+0
		std	1.5157E+1	2.6296E+0	2.1807E+0
f_3	50	best	3.4183E+2	1.2586E+2	7.5750E-1
		mean	3.4406E+2	1.5413E+2	6.6067E+0
		std	1.2330E+2	8.0119E+2	9.1760E+0
f_4	50	best	5.5498E+2	9.0880E+0	4.1340E-6
		mean	2.3569E+2	2.0891E+4	1.4728E+1
		std	3.2918E+6	5.6265E+1	4.2121E+3
f_5	50	best	3.4273E+0	5.2642E+1	2.5526E-1
		mean	6.5223E+1	2.8925E+2	4.3723E-1
		std	4.8823E-1	2.1792E+0	5.2325E-1
f_6	50	best	4.1716E+0	2.9725E+2	6.2327E-4
		mean	2.1223E+0	5.1296E+2	2.7169E-3
		std	3.2877E+2	3.6281E-2	2.2092E-3

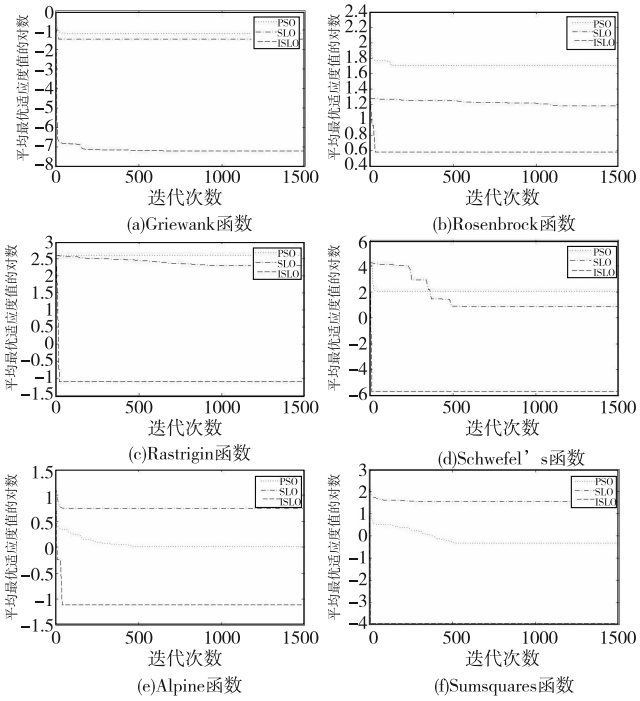

图2 测试函数迭代曲线图

表 3 ~ 5 中的最佳值 best 反映了算法的收敛精度，平均值 mean 反映了算法的寻优能力，方差 std 反映了算法的稳定性。从表 3 中可以看出，对于低维下（$D=10$）的六种函数，ISLO 算法的精度均优于其他两种算法。对于较难优化的多峰复杂函数 f_1，ISLO 算法精度达到了 E-12 以上。当维数逐渐增大时，搜索空间变得更加复杂，对算法的寻优能力更是考验。从表 4 可知，当维度增加至 30 维时，ISLO 算法在六种函数上所得的最优精度、平均最优值均比 SLO 明显提升，尤其 （下转第 2331 页）

传感器附着于一个 11 cm × 9 cm × 7 cm 尺寸的物块之上,通过串口连接至计算机。按照 3.2 节所述实验方法充分模拟货物的不同行为,传感器采集加速度及姿态角数据以文本文件保存至计算机,再导入到 MATLAB 中进行分析。

4.2　算法参数设定

在实验中对本章所述算法参数进行如下设定:a)滤波系数 $a = 0.9$,因本文采取模拟实验的环境,环境干扰较小,所以本文设定的滤波系数取值较大,灵敏性较高;b)时间窗口长度 $m = 10$,门限阈值 $\text{threshold}_{\text{begin}} = 0.10$, $\text{threshold}_{\text{end}} = 0.0005$,参数影响时间窗口的截取效果,较大的开始阈值及较小的结束阈值确保整个危险品货物活动期间的数据段被有效截取;c)决策树算法中,最大加速度阈值 $a_{th} = 1.5$,最大加速度偏移量阈值 $oa_{th} = 5$,重力阈值 $G_{th} = 75$(理论上重力方向改变时,$G_{th} = 90$,但实际实验中由于测量误差等存在,实际值往往小于 90,且在 $G_{th} > 75$ 情形下完全有理由相信货物一定发生翻倒,所以本文对重力阈值初始取值 75),较大倾角阈值 $L_{th} = 15$。

4.3　实验结果

将采集到的五类行为数据(每类 30 组)进行平滑滤波,并运行时间窗口检测算法,对于时间窗口没有检测到的样本数据,令 f_{SW} = false,且不再继续其他属性值计算。对其余样本进行预处理,计算出危险品货物的行为特征向量 F。计算完成后,每类行为选取 15 组特征向量作为训练样本进行训练,动态修改节点的布尔属性。训练完成后,将每类剩余的 15 组特征向量作为测试样本进行测试。测试结果如表 2 所示。

表 2　货物异常行为实验检测结果

货物行为	正确率/%	误判率/%	错误类型	货物行为	正确率/%	误判率/%	错误类型
移动	93.3	6.7	平衡	撞击	100	0	– –
晃动	86.7	13.3	撞击	翻倒	100	0	– –

在上述实验中可以发现,撞击和翻倒的识别率最高,可以实现 100% 的检测率,但是微小的移动有可能在窗口检测阶段被忽略,因此被直接判定为平衡,较大幅度、较大频率的晃动可能被判别为撞击。而相反地,撞击行为却不会被误判。通过对数据进行具体分析发现,轻微的撞击由于受力较小,不易产生姿态角的变化,所以在 LTA 节点取值为 false;而猛烈的撞击由于加速度偏移量极大,也不会被误判为晃动,所以识别效果较好。总体来看,本算法的姿态检测识别率可以达到 96%,能够有效识别危险品货物的行为姿态。

5　结束语

针对在途危险品货物状态监控问题,本文提出了一种基于六轴加速度传感器的行为姿态检测方法,实现在途危险品货物的异常行为姿态的检测。首先,本文方法的时间窗口检测算法能够有效提取关键数据段,解决加速度传感器数据采集及处理量大的问题;其次,针对危险品货物,对其行为特征及危险程度进行了定义和划分,以实现针对性的检测;最后,针对本文提出的五种异常行为,基于不同行为的运动数据特征,设计特征向量和分类器,达到准确的分类效果,对在途危险品状态的监控具有一定的实际应用价值。

参考文献:

[1] 祖绍鹏, 郭明儒, 刘会超, 等. 易损/危险品物流监测智能微系统[J]. 传感技术学报, 2012, 25(7): 1019-1022.

[2] 刘凯峥, 刘浩学, 晏远春, 等. 罐体车辆道路运输危险品事故特征分析[J]. 安全与环境学报, 2010, 10(3): 130-133.

[3] 于浚烽, 陈蔚芳, 马万太. 基于 GIS/GPS 技术的肉品冷链物流监控与调度系统[J]. 计算机应用, 2014,34(S1): 312-314.

[4] 姚振强, 王建, 胡永祥, 等. 基于 RFID/GPRS/GPS/GIS 的危险品物流智能监管系统[J]. 公路交通科技, 2013, 30(2): 147-158.

[5] 孙玉砚, 杨红, 刘卓华, 等. 基于无线传感器网络的智能物流跟踪系统[J]. 计算机研究与发展, 2011, 48(S2): 343-349.

[6] 曹丽, 刘扬, 刘伟. 利用加速度计和角速度仪的笔杆运动姿态检测[J]. 仪器仪表学报, 2008, 29(4): 831-835.

[7] Xue Yang, Jin Lianwen. A naturalistic 3D acceleration-based activity dataset & benchmark evaluations[C]//Proc of IEEE International Conference on Systems Man and Cybernetics. Piscataway, NJ: IEEE Press,2010: 4081-4085.

[8] Kratz S, Rohs M. The $3 recognizer: simple 3D gesture recognition on mobile devices[C]//Proc of the 15th International Conference on Intelligent User Interfaces. New York: ACM Press,2010: 419-420.

[9] Junker H, Amft O, Lukowicz P, *et al*. Gesture spotting with body-worn inertial sensors to detect user activities[J]. Pattern Recognition,2008,41(6):2010-2024.

[10] 陈丽萍, 武文波. 基于决策树 C4.5 算法的面向对象分类方法研究[J]. 遥感信息, 2013, 28(2): 116-120.

[11] 孙永健, 程臻, 付莹. 基于四元数的弹道目标微多普勒仿真[J]. 电光与控制, 2013, 20(11): 65-69.

(上接第 2322 页)在处理复杂函数如 f_3、f_5 时,ISLO 算法的稳定性也优于 SLO 及 PSO 算法。对于高维($D = 50$)问题,ISLO 算法在处理复杂函数 f_6 时,其收敛精度相对于 $D = 10$,30 得到一定的提升。对于其他函数,ISLO 算法在基本维持稳定性的同时,其搜索精度与寻优能力仍好于 SLO 和 PSO 算法。

测试函数的优化迭代曲线可以直观地反映算法在收敛性能上的表现。从图 2 可知,对于 $f_1 \sim f_6$ 而言,改进算法的收敛速度、收敛精度均优于 SLO 和 PSO 算法。从算法迭代开始,ISLO 算法就表现出相当优秀的寻优性能。在运行前期,改进算法有较快的寻优速度;在运行中后期,由于种群具有良好的多样性,算法进行更精细的局部开采,无论是对于存在大量局部最优点的多峰复杂函数或高维单峰函数,改进算法均能保持良好的收敛性。总体来说,ISLO 算法在处理多维复杂函数时具有绝对的优势,可将其适用于复杂非线性优化问题。

4　结束语

针对七星瓢虫优化算法求解精度不高、易陷入局部最优的缺陷,本文提出一种改进的七星瓢虫优化算法。在每次迭代时采用柯西变异策略,将变异产生的新个体与原种群一起进行竞争协作,并在循环末利用竞争淘汰机制淘汰适应度较差的个体;同时,在算法搜索后期引入混沌变异策略,对陷入局部最优与较优解进行优化变异,并引入自适应调整学习因子,以提高算法的收敛性。实验仿真结果表明,该算法不仅在求解精度上得到明显改善,并在一定程度上克服了原算法易早熟停滞的不足,具有良好的收敛性能。

参考文献:

[1] Kennedy J, Eberhart R. Particle swarm optimization[C]//Proc of IEEE International Conference on Neural Networks. 1995: 1942-1948.

[2] Karaboga D. An idea based on honey bee swarm for numerical optimization, TR06[R]. Kayseri: Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[3] Meng Xianbing, Liu Yu, Gao Xiaozhi, *et al*. A new bio-inspired algorithm: chicken swarm optimization[C]//Proc of International Conference in Swarm Intelligence. Cham: Springer,2014: 86-94.

[4] 冯翔,张进文,虞慧群. 仿生蚊子追踪算法[J]. 计算机学报,2014, 37(8):1794-1808.

[5] 吴意乐,何庆. 基于改进遗传模拟退火算法的 WSN 路径优化算法[J]. 计算机应用研究,2016,33(10):2959-2962.

[6] 包晓晓,叶春明,计磊,等. 改进混沌烟花算法的多目标调度优化研究[J]. 计算机应用研究,2016,33(9):2601-2605.

[7] 罗钧,林于晴,刘学明,等. 改进蜂群算法及其在圆度误差评定中的应用[J]. 机械工程学报,2016,52(16):27-32.

[8] 王娜,高学军. 一种新颖的差分混合蛙跳算法[J]. 计算机系统应用,2017,26(1):196-200.

[9] Karaboga D, Gorkemli B. A quick artificial bee colony (qABC) algorithm and its performance on optimization problems[J]. Applied Soft Computing, 2014, 23(5): 227-238.

[10] 王鹏,李洋,王昆仑. 七星瓢虫优化算法及其在多学科协同优化中的应用[J]. 计算机科学,2015,42(11):266-269,304.